

Survey Research Using SAS: Tricks of the Trade

Archer R. Gravely, UNC Asheville, Asheville, NC

Abstract

This paper will illustrate the use of SAS to conduct mail and telephone survey research projects based on principles of efficient survey practice. SAS techniques for accomplishing the following survey research tasks are discussed: (1) calculating sample size requirements and error rates; (2) techniques for generating personalized survey communications and tracking respondents; (3) character and numeric data manipulation; and (4) tips for reporting survey data. This paper is based on the book *Your Guide to Survey Research Using the SAS System* published by SAS Institute in 1998.

Introduction

In all types of organizations, survey research data often provide some of the more useful information for identifying strengths and weaknesses, developing marketing strategies, performing needs assessments, and providing the empirical evidence required to make difficult resource reallocation decisions. This paper will provide the reader with a number of SAS® based techniques for organizing and conducting mail and telephone surveys with minimal effort, determining sample size and sampling errors, manipulating both character and numeric data, and presenting survey results in an effective manner.

General Principles of Efficient Survey Practice

- Avoid asking respondents for information that is available through electronic records
- Obtain high response rates by using a personalized survey design
- Use standardized procedures for conducting surveys
- Create camera-ready tables and graphs with SAS.

Determine Sample Size/Sampling Error

One of the first steps in planning a survey project is to determine the sample size. This decision is usually a compromise between the desired

maximum sampling error and resources available to conduct the survey. To determine the sample size, the following factors should be considered:

- Population size
- Desired sampling error
- Costs/budget for data collection
- Estimate of variance for key variables

An estimate of the population variance can be obtained from previous research or estimated with a pilot study. If the key variable of interest is measured as a percentage or proportion, which is often the case in survey research, a proportion estimate of .5 can be used to calculate the sample size needed to obtain a specified sampling error. The use of $p = .5$ is a conservative approach as this value yields the *maximum* possible variance and hence a larger required sample size than would be needed for other p values.

To facilitate the survey planning process, two SAS programs (based on formulas from Sheaffer, et. al. 1979) are provided. The first program provides a procedure for calculating the *required* sample size for a specified error rate in the estimation of a proportion or percentage variable. The results of this program are shown in **Output 1**. The second program calculates the *sample error* for a given sample size, observed proportion (p), and population size (N) as shown in **Output 2**.

SAS Code for Determining Sample Size

```
data compute;
***** Define Parameter*****;
N = 2000; ** Population Size **;
p = .5; ** Estimated Sample Proportion **;
b = .05; ** Bound on Error of Estimation **;
*****;
q = 1-p;
d = (b*b)/4;
sample = (N*p*q)/(((N-1)*d)+(p*q));

proc print data = compute label; id N;
var p b sample;
label N = 'Population Size'
p = 'Estimated Proportion'
sample = 'Sample Size Needed'
b = 'Sampling Error';
title1 'Needed Sample Size';
run;
```

Output 1: Sample Size Calculations

Population Size	Needed Sample Size		Sample Size Needed
	Estimated Proportion	Sampling Error	
2000	0.5	0.05	333.472

As shown in **Output 1**, for a population size of 2,000, a sample size of 334 is needed to have a maximum sampling error of .05 in estimating a proportion. It is important to note that for mail surveys, the sample size will need to be *increased to allow for non-responses*. In this example, the 334 sample size figure is the number of *completed* surveys needed; if you send a mailed survey to 334 individuals, you will undoubtedly have a number of non-responses and thus a sampling error greater than .05. Using Dillman's (1979) survey techniques, we can usually get a response rate of 60+% to mail surveys. Therefore, using this example, we would select a random sample 334/.60 or 557 to allow for a 40% non-response resulting in 334 completed surveys. A generalized statement of this adjustment for non-response is as follows:

Mail Survey Sample Size Adjustment	
Adjusted Sample Size =	$\frac{\text{Sample Size Needed}}{\text{Estimated Response Rate}}$

It is often of interest to determine the amount of sampling error for a given sample size, observed sample proportion estimate, and population size. The SAS code to follow will compute the error when these parameters are defined in the data step as shown in the following program. The results are shown in **Output 2**.

SAS Code To Determine Sampling Error

```
data sample;
***** Define Parameters *****,
p = .15;      ** Sample Proportion **;
N = 3000;     ** Population Size **;
sample = 100; ** Sample size **;
*****
error = 2*(sqrt((p*(1-p))/
(sample-1))*(N-sample)/N));
error = round(error, .01);

proc print data=sample; id N;
var sample p error;
```

```
titlel 'Bound on the Error of Estimation';
run;
```

Output 2: Sampling Error Calculations

Bound on the Error of Estimation			
N	SAMPLE	P	ERROR
3000	100	0.15	0.07

Generating Random Telephone Numbers

An excellent way of developing a sample for telephone surveys of the general population is to use SAS to create a file of random numbers that can be translated into phone numbers appropriate for the survey population geographical area. The random digit dialing (RDD) approach provides two major advantages: (1) it provides a quick, and inexpensive way of developing a sampling frame; and (2) the use of randomly generated telephone numbers completely eliminates the sampling bias of not having unlisted or new numbers when the sample is taken from published sources. The first step in using this technique is to identify the area codes and telephone prefixes included in the survey population geographical area. Local telephone companies can provide this information and also identify blocks of phone numbers that are not in service or used for commercial purposes. The following SAS program creates five random phone numbers for a pre-selected '251' prefix.

SAS Code to Generate Random Phone Numbers

```
data sample;
length phone $8;
do i = 1 to 5;
x1 = ranuni(0);
x2 = x1*10000;
last4 = put(x2,z4.);
phone='251-'||last4;
output;
end;

proc sort data = sample nodupkey;
by last4;

proc print data = sample;
var x1 x2 last4 phone;
titlel 'Random Phone Numbers' ;
```

Output 3: Generate Random Phone Numbers

Random Phone Numbers

OBS	X1	X2	LAST4	PHONE
1	0.02698	269.77	0270	251-0270
2	0.06073	607.31	0607	251-0607
3	0.06807	680.68	0681	251-0681
4	0.07520	751.98	0752	251-0752
5	0.10792	1079.22	1079	251-1079

Conducting Personalized Mail Surveys

The personalization of mail surveys is one of the key elements required to obtain acceptable to high response rates. Although there are no formally acknowledged standards for defining "acceptable" response rates, Salant and Dillman (1994) suggest that mail surveys with less than 60-70% should have a "red flag" to indicate questionable validity. The SAS System provides a powerful tool for achieving a high degree of personalization with *minimal* effort.

Creating ID numbers

The use of ID numbers in survey projects provide a mechanism for tracking respondent participation, identifying survey forms for correcting keypunch errors, and combining survey data with information from other electronic sources. As shown in the following SAS code, the RETAIN statement can be used to assign a sequential number to each observation creating an ID number for each subject

```
data sample;
retain id 0;
  set pop;
id + 1;
```

Name/Address File

For mail surveys, it is useful to create a name/address file to manage the survey process. The name/address file provides a data source for:

- generating individualized form letters and envelopes
- tracking respondent participation status and identifying non-respondents who should not be sent a follow-up mailing (e.g., person responded, address information is invalid; deceased, etc.)

The SAS code to follow illustrates how to build a name/address file by merging a SAS data set (contains the survey sample) with an ASCII file of

address information. This code demonstrates the use of CONCATENATION (||), LENGTH statements, ARRAY/DO processing, and the use of the SUBSTR function. Also, note the use of a DO loop to remove periods and commas from the address fields to conform to U.S. postal regulations. The results of this program are shown in **OUTPUT 4**.

SAS Code for Creating Name/Address File

```
libname database '[gravely]';
filename address 'company.dat';
filename saveit 'address.dat';

data sample;
  set database.survey;

data mail;
length zip $10;
  infile address;
input
@001 ssn      09. @013 name      $32.
@045 firname  $20. @065 lasname  $20.
@121 street1  $32. @153 street2  $32.
@185 city     $20. @205 state     $02.
@209 zip5     $05. @214 zip_ext  $04.
@261 mi       $01. ;
  if zip_ext > ' ' then
    zip = zip5||'-'||zip_ext;
  else zip = zip5;
proc sort data = sample; by ssn;
proc sort data = mail;   by ssn;

data update;
  merge sample(in=a) mail; by ssn;
if a;
if sex = 'M' then title = 'Mr';
else if sex = 'F' then title = 'Ms';
do p = 1 to 32;
if substr(street1,p,1) = '.' then
  substr(street1,p,1) = ' ';
if substr(street2,p,1) = '.' then
  substr(street2,p,1) = ' ';
if p < 21 and substr(city,p,1) = '.'
  then substr(city,p,1) = ' ';
end;

proc sort data = update; by id;

data _null_ ;
  set update;
file saveit recfm=f lrecl=83;
put
@02 id          z4. @06 title $2.
@09 firname $char20. @29 mi     $1.
@31 lasname  $char20.
@51 street1  $char32.
#2
@01 street2  $char32. @33 city  $char20.
@54 state    $2. @57 zip  $char10.;
```

Note that each record is written on two lines and that the first column of the first line (before ID) is left blank. This space is used to manually input a flag to indicate respondent participation status. We use the following coding system to track respondents:

- 1 = Completed survey
- 2 = Bad address
- 9 = Remove from survey sample

Once the address file is built, we are ready to create personalized form letters. A print merge operation is performed on the sample document shown in **Figure 1** to generate personalized cover letters. The name1, name2, street1, street2, city, state, and zip variables are obtained from a SAS created text file during the word processing print merge operation. The SAS code to create this file follows the sample cover letter with the resulting text file shown in **Output 4**.

Figure 1: Survey Letter - Print Merge

<<DATA letter.lis>>

September 24, 1998

```
<<name1>>
<<street1>>
<<if street2>><<street2>>
<<ENDIF>><<city>> <<state>> <<zip>>
```

Dear <<name2>> :

{ body of survey cover letter}

Sincerely,

Archer R. Gravely
Director

SAS Code to Create Letter File

```
filename addr 'address.dat';
filename output 'letter.lis';

data address;
length street1 $40 street2 $40 address $100;
infile addr recfm=f lrecl=90;
input
@1 flag $1.
@02 id $4. @06 title $2.
@09 firname $char20.
@29 mi $1. @31 lasname $char20.
@51 street1 $char32.
#2
@01 street2 $char32. @33 city $char20.
@54 state $2. @57 zip $char10.;
if flag in ('1','2','9') then delete;
name1 = title||'. '||trim(firname)||
' '||trim(lasname);
name2 = title||'. '||trim(lasname);
street1 = trim(street1);
street2 = trim(street2);
city = ' '||trim(city)||', '||trim(zip);
address=trim(name1)||', '||trim(name2)||', '||
```

```
trim(street1)||', '||trim(street2)||', '||
trim(city)||', '||trim(state)||', '||trim(zip);

proc sort data = address; by zip;

data _null_ ;
set address;
file output recfm=v;
if _n_ = 1 then put @1
'name1,name2,street1,street2,city,state,zip';
put @1 address;
```

Output 4: Letter File

```
NAME1,NAME2,STREET1,STREET2,CITY,STATE,ZIP
Mr. Ryan McCall,Mr. McCall,6514 Brockstone Drive, "Stoney Creek",NC,27377-9221
Ms. Berit Bjorklund,Ms. Bjorklund,9438 St. Mary's Street, "Raleigh",NC,27605
Ms. Danielle Claycomb,Ms. Claycomb,1916 Thopshire Dr., "Raleigh",NC,27615
Mr. David Finnan,Mr. Finnan,302 Hopewell Road #C, "Morganton",NC,28655
Mr. Marc Ludvigsen,Mr. Ludvigsen,156 Monte Vista Road, "Statesville",NC,28677
Ms. Sandi Roper,Ms. Roper,168 Pinecrest Drive, "Alexander",NC,28701
Mr. Timothy Means,Mr. Means,222 Glenn Bridge Road B-16, "Arden",NC,28704
Mr. John Elliott,Mr. Elliott,18 Willowcreek Lane, "Arden",NC,28704
Ms. Deborah Sheppard,Ms. Sheppard,7 N. Reece Drive, "Arden",NC,28704
Mr. Spurgeon Gilbert,Mr. Gilbert,PO Box 960, "Arden",NC,28704
Ms. Carole Cogburn,Ms. Cogburn,37 Morgan St., "Canton",NC,28716
Ms. Monica Jemigan,Ms. Jemigan,PO Box 617 Ridge Road, "Dana",NC,28724-0617
Ms. Joy Ciuro,Ms. Ciuro,23 Spring Mtn Rd., "Fairview",NC,28730
Mr. Matthew Simmons,Mr. Simmons,111 Fairview Hills Dr., "Fairview",NC,28730-9777
Mr. George Wagner III,Mr. Wagner III,160 Poplar Loop, "Flat Rock",NC,28731
```

Survey Data Manipulation Techniques

The SAS system provides a powerful set of tools for handling survey research data and variable manipulation tasks. The use of these tools is an important component of efficient survey practice.

Reversing Scale Values

In the analysis and reporting of survey data, it is often necessary to reverse item scale values. This process can be accomplished with ARRAY statements or by using a simple mathematical trick of subtracting the original variable from a constant that is equal to the number of scale values plus one. To illustrate, we will use a typical Likert survey item named Q01 with the following scale values:

- 1 Strongly Disagree
- 2 Disagree
- 3 Agree Somewhat
- 4 Strongly Agree

To reverse this scale the following code, a new variable R01 is created:

```
r01 = 5 - q01;
```

A comparison of Q01 and R01 values is shown in the following table.

Q01 Values	R01 Values
1	4
2	3
3	2
4	1

If the original scale has 5 values, then you would subtract the variable from 6. This logic continues for any number of scale values. A generalized statement of this scale reversal technique is as follows:

recoded variable = $(n + 1) - \text{original variable}$
 where n = number of scale values

Create Variable Based on Format Value

It is sometimes useful to sort or print observations by *formatted* values or labels rather than the *actual* variable values. In the following example, we have a variable named DEPTCODE that is represented by a two digit numerical code. A sort/print of this variable as currently defined will not produce an alphabetized list of DEPTCODES by formatted label. One solution to this problem is to create a new variable to represent the user defined value labels created with PROC FORMAT. This conversion can be accomplished with the PUT function as shown in the following SAS program with the results shown in **Output 5**.

Create New Vars With Format Values

```
proc format;
    value $code '01' = 'Service'
               '02' = 'Finance'
               '03' = 'Marketing'
               '04' = 'Sales';

data report;
    input @1 deptcode $2. @4 rating 1.;
    dept = put(deptcode,$code.);
drop deptcode;
cards;
01 3
02 4
03 2
04 4
;
proc sort data = report; by dept;
proc print; id dept;
var rating;
title1 'Department Ratings';
```

Output 5: Create Values With Format Values

Department Ratings	
DEPT	RATING
Finance	4
Marketing	2
Sales	4
Service	3

Advanced Character Manipulation Techniques

This section will illustrate the use of the SAS programming language to perform a number of character manipulation tasks that are sometimes required to implement a personalized mail survey project. In this example, we want to conduct a personalized mail survey, but the sample name information is stored as a one variable, all upper case character string. To generate personalized form letters, we need to separate the combined first name, middle name, and last name field into *three* distinct variables. Additionally, we need to translate the all uppercase characters into the appropriate upper/lower case configuration required for personalized correspondence. The original data set is shown in **Figure 2**.

Figure 2: NAME.DAT File

```
THOMAS REAZER COCHRAN
CAROLINE B. MILLER
KERN ARTHUR PARKER
SHIRLEY S. BROWNING
JAMES R. PITTS
ELAINE A. FOX
MALCOLM E. BLOWERS
MERRITT W. MOSELEY
```

Our approach to converting the all uppercase single variable name field into three separate name variables with appropriate upper/lower case characters is based on a four step process.

1. Convert uppercase text to all lowercase text.
2. Identify the character position number of the blank spaces that separate first name, middle name, and last name in the original name field.
3. Use the position numbers of these blanks or separators as a guide or pointer to extract the three distinct name elements.
4. Convert the first character of each newly created name field to upper case.

The following SAS program will make these data conversions. The program results are shown in **Output 6**.

Convert All Uppercase Single Name Field to Separate Upper/Lower Case Name Variables

```
data test;
  infile 'name.dat';
  input name $35. ;
  space1 = 0;
  space2 = 0;
  space3 = 0;
  name = lowercase(name);
  do x = 1 to 35;
    if substr(name,x,1) = ' ' then do;
      if space1 = 0 then space1 = x;
      else if space2 = 0 then space2 = x;
      else if space3 = 0 then space3 = x;
    end;
  end;
  fname = substr(name,1,space1-1);
  mname = substr(name,space1+1,((space2-1)-
  space1));
  lname = substr(name,space2+1,((space3-1)-
  space2));
  if lname = ' ' then do;
    lname=mname;
    mname = ' ';
  end;
  substr(fname,1,1) = upcase(substr(fname,1,1));
  substr(mname,1,1) = upcase(substr(mname,1,1));
  substr(lname,1,1) = upcase(substr(lname,1,1));
  proc print; var fname mname lname;
  title1 'Creation of New Name Variables';
run;
```

Output 6: Name Conversion

Creation of New Name Variables			
OBS	FNAME	MNAME	LNAME
1	Thomas	Reazer	Cochran
2	Caroline	B.	Miller
3	Kern	Arthur	Parker
4	Shirley	S.	Browning
5	James	R.	Pitts
6	Elaine	A.	Fox
7	Malcolm	E.	Blowers
8	Merritt	W.	Moseley

It should be noted that name conversions of this type usually do not work perfectly for several reasons. One, you will sometimes encounter records with four names when your code assumes three names; and two, you may find inconsistent spacing between names in the original data file. The previous example can be augmented with additional SAS code to handle names with mixed upper/lower case (e.g. McDowell). When using conversions of this type, it is very important to *proof* the name/address before printing individualized letters/envelopes. Personalization features poorly handled may have a more deleterious effect on response rates than would

be obtained by a mistake free but crude "DEAR ALUMNI" or "DEAR CUSTOMER" letter.

Reshaping Data - Convert Vars to Obs

In the analysis and reporting of survey results, it is often necessary to reshape your data to create multiple observations from existing observations based on responses to one or more variables. There are two general applications for this type of data manipulation in survey work.

1. A common survey technique is to have multiple response type questions where respondents are instructed to *check all that apply*. While this type of question may appear on the survey form as a single question with *n* response categories, it is usually coded as *n* variables with dichotomous response coding (Y/N, 1/0, etc.).
2. In the analysis of a series of questions related to a single dimension, an effective presentation of the results may be made by reporting the data in a single table or graph rather than multiple tables or graphs. With the SAS system, this type of presentation is facilitated by reshaping multiple variables into a single variable whose values are represented by the creation of new observations.

The process for reshaping data under either of these two scenarios is based on using the OUTPUT statement. The SAS System executes an implicit OUTPUT command by default at the end of every DATA step to write observations to one or more SAS data sets. When specified by a user, the OUTPUT statement directs SAS to write the current observation to the SAS data set immediately rather than at the end of the data step. A good practice is to use DROP/KEEP statements to eliminate unnecessary variables as part of the data reshaping process. The example to follow is based on the multiple response type question shown in Figure 3.

Figure 3: Multiple Response Example Survey Items

The following statements reflect the goals of many students. Which of these goals are important to you? (Circle all that apply)

01 To Prepare for a career or good job
 02 To take part in college social life
 03 To dev an appreciation & enjoyment of art, music, & lit.
 04 To dev an understanding & appreciation of science & tech

In the original data set, each ID or case has one observation with four goal variables coded as 1 or 0.

The SAS code to follow reshapes the multiple response survey items (GOAL01-GOAL04) into a pair of new variables named GOAL and FREQ. GOAL captures the original GOAL01-GOAL04 label information and FREQ represents the response (1/0) for each goal. In the transformed data set, the observation count for each ID is expanded to four so as to have a separate GOAL value for each record. The information previously contained in four variables per record now resides on two variables stored on four records. The new data set is not efficient for storage purposes, but has advantages for generating custom tables and graphs where we need to treat multiple variables as a single dimension to facilitate the reader's perceptual ability to make quick comparisons. For this example, PROC TABULATE is used to report a frequency count of the new variable GOAL in a single table as shown in **Output 7**.

SAS Code for Reshaping Data

```
libname library 'pb hd:new student svy';

data survey(keep=goal freq);
length goal $40;
set library.db966(keep=goal01-goal04);

goal = 'Prepare For Career or Good Job';
if goal01 = '1' then freq = 1;
else freq = 0;
output;

goal = 'Take Part in College Social Life';
if goal02 = '1' then freq = 1;
else freq = 0;
output;

goal = 'Dev Apprec, Enjoy of Art, Music, & Lit';
if goal03 = '1' then freq = 1;
else freq = 0;
output;

goal = 'Dev Understand of Science and Tech';
if goal04 = '1' then freq = 1;
else freq = 0;
output;

proc tabulate data = survey;
class goal;
var freq;
table goal= ' ', freq = 'Frequency'*sum=
'f=11.0/
box = 'Goals for Attending College' RTS=42;
title1 'Reshaping Data Example Output';
run;
```

Output 7: Reshaping Vars into Obs

Reshaping Data Example Output

Goals for Attending College	Frequency
Dev Apprec, Enjoy of Art, Music, & Lit	280
Dev Understand of Science and Tech	242
Prepare For Career or Good Job	580
Take Part in College Social Life	380

Data Analysis and Reporting

The researcher has three main tools to convey the message of his or her work: words, tables, and graphs. Like a woodworker's chisels, saws, and planes, these tools are best used to perform distinct functions, but all are usually required in combination to produce a quality piece of work. Some general guidelines (Tufte, 1983) for the effective use of tables, graphs, and words are:

- Tables are the media of choice for conveying numerical *values*;
- Graphs are best suited for depicting *relationships* among variables
- When small numerical differences have important implications, graphs tend to do a poor job of communicating the findings
- Tables do a much better job than words in comparing more than two numbers
- For reporting small data sets, tables are more effective than graphs
- Sentences are used to *summarize* information

Mathematical Trick for Calculating Percentages

In many survey research projects, percentages are the *statistic of choice* for summarizing and reporting the results. In the measurement of perceptions and attitudes, percentages have two major advantages over the use of mean scores:

- For interval level measurements, the use of means, while frequently reported, may not be statistically appropriate
- Summary measures such as the percentage of respondents that "agree" or are "satisfied" with some dimension has a much more *intuitive* meaning to to readers than is conveyed by a mean scores for a typical Likert type scale.

PROC FREQ provides an easy to use procedure for generating percentages in one-way and n-way tables, but the output can be lengthy and not reader friendly. PROC TABULATE provides an enhanced capability to report percentages in custom tables, but this procedure can involve considerable complexity. A quick and easy way of computing percentages is based on the mathematical "trick" of creating a new variable with two possible values: 100 to represent the presence of some characteristic and 0 to indicate the absence of the characteristic. The *mean* of this new variable is the *percentage* of observations with the characteristic represented by the value of 100. If a proportion is desired, the new variable can be represented by a 1 or 0 coding scheme.

In the SAS code to follow, the two survey items shown in **Figure 4** will be recoded as PERCENT1 and PERCENT2. Respondents who indicated "Yes" to these items will have a PERCENT1/PERCENT2 value of 100 and the "No" responses will be coded as 0. These operations are shown in the highlighted text of the SAS code to follow. The TABULATE analysis will include academic discipline (DISCIP) and YEAR as CLASS variables or subgroups. The results are shown in **Output 8**.

Figure 4: Calculate Percentages With Means Survey Item

4. (a) If you could begin again, would you choose to attend UNCA?
- 1 Yes
 - 2 No
- (b) Would you choose the same major field of study?
- 1 Yes
 - 2 No

SAS Code for Computing Percentages With Means

```
libname svy 'pb hd:gss';

data report;
  set svy.data95(keep=a4_a a4_b year discip)
      svy.data96(keep=a4_a a4_b year discip);

  if a4_a = '1' then percent1 = 100;
  else if a4_a = '2' then percent1 = 0;

  if a4_b = '1' then percent2 = 100;
  else if a4_b = '2' then percent2 = 0;

proc format;
  value $discip 'NS' = 'Natural Sciences'
               'SS' = 'Social Sciences'
               'HM' = 'Humanities'
               'PR' = 'Professional';
```

```
value $year '95' = '1995'
            '96' = '1996';
```

```
proc tabulate data = report noseps;
  class discip year;
  var percent1 percent2;
  table discip*(Year ALL='Discip Total') year
all='Total',
      N*F=5. percent1 = 'Attend UNCA
Again' *mean='Percent'*F=12.1
      Percent2 = 'Same Major Again' * mean=
'Percent'*F=12.1/
rts=23 indent=3;
FORMAT discip $discip. year $year.;
title1 'If You Could Begin Again, Would You
Choose To: ';
title2 '(1) Attend UNCA Or (2) Choose Same
Field Of Study?';
run;
```

Output 8: Calculate Percentages With Means

If You Could Begin Again, Would You Choose To:
(1) Attend UNCA Or (2) Choose Same Field Of Study?

	N	Attend UNCA Again	Same Major Again
		Percent	Percent
Humanities			
1995	55	85.2	90.6
1996	38	94.4	88.9
Discip Total	93	88.9	89.9
Natural Science			
1995	75	89.7	88.9
1996	59	83.1	84.7
Discip Total	134	86.6	87.0
Professional			
1995	122	89.1	86.8
1996	90	85.2	89.9
Discip Total	212	87.4	88.1
Social Sciences			
1995	117	83.3	83.0
1996	92	83.0	81.9
Discip Total	209	83.2	82.6
1995	369	86.8	86.6
1996	279	85.2	86.1
Total	648	86.1	86.4

TABULATE Supertables

The *supertable* concept was developed by Tufte (1983) in his classic work *The Presentation of Quantitative Information*. The basic idea of the supertable is to provide the reader with a comprehensive table of multiple data comparisons clustered around a common dimension. This data rich unified table is a far better presentation than would be obtained by the tedious repetition of numerous small tables or histograms related to a common dimension. Supertables have a reference like quality based on the amount of information provided in an easy to read, attractive format.

The PROC TABULATE supertable is simply a concatenated table with numerous row variables. Since TABULATE, by default, *excludes* all observations with one or more missing CLASS variables, it is important to use the **MISSING** option

on the PROC statement. Given the frequency of missing data in most survey research applications, the **MISSING** option will prevent the unnecessary loss of observations when respondents provide valid responses to some but not all CLASS variables. Based on the previous example (Figure 4), the SAS code to follow produces a *preliminary* supertable show in **Output 9**.

SAS Code for TABULATE Supertable

```
libname library 'pb hd:gss';

data supertab;
length year $4;
set library.data95
    library.data96;
year= '19'||year;
if a4_a = '1' then percent = 100;
else if a4_a = '2' then percent = 0;

proc tabulate data= supertab missing noseps;
var percent;
class discip admtype cumgpa race sex year;
table discip='Discipline:'
    admtype = 'Admission''s Type:'
    cumgpa = 'Cumulative GPA:'
    race = 'Race:'
    sex = 'Sex:'
    all = 'Total',
    year='Year Graduated'*(n*f=5.0
percent=' '*mean='Pct'*f=5.1)
all='2-Year Total'*(n*f=5.0
    percent=' '*mean='Pct'*f=5.1)/
    rts=23 box='Dimension' ;
format discip $discip. admtype $admtype. cumgpa
cumgpa. race $race.
sex $sex.;
title1 'If You Could Begin Again, Would You
Choose to Attend UNCA';
title2 '(Percent Responding "Yes")';
run;
```

Output 9: TABULATE Supertable

If You Could Begin Again, Would You Choose to Attend UNCA
(Percent Responding "Yes")

Dimension	Year Graduated				2-Year Total	
	1995		1996			
	N	Pct	N	Pct	N	Pct
Discipline:						
	35	86.2	14	63.6	49	80.0
Humanities	55	85.2	38	94.4	93	88.9
Natural Sciences	75	89.7	59	83.1	134	86.6
Professional	122	89.1	90	85.2	212	87.4
Social Sciences	117	83.3	92	83.0	209	83.2
Admission's Type						
	37	87.1	14	63.6	51	81.0
Freshmen	172	85.7	130	88.4	302	86.9
Transfer	195	87.6	149	82.4	344	85.3
Cumulative GPA:						
	37	87.1	14	63.6	51	81.0
2.00-2.49	51	81.6	34	53.1	85	70.4
2.50-2.99	110	85.7	78	69.7	188	87.4
3.00-3.49	118	85.2	98	91.5	216	88.0
3.50-4.00	88	92.9	69	86.6	157	90.1
Race:						
	37	87.1	14	63.6	51	81.0
Other	14	85.7	8	87.5	22	86.4
Black	12	95.0	14	90.0	26	92.3
White	341	87.2	257	86.9	598	87.0
Sex:						
	37	87.1	14	63.6	51	81.0
Female	214	89.0	161	87.4	375	88.3
Male	153	83.2	118	82.1	271	82.7
Total	404	86.7	293	84.4	697	85.7

As shown in **Output 9**, the **MISSING** option generates a missing value row header as the first line for each row variable. This blank header provides little useful information and is likely to confuse readers, but the **MISSING** option is needed to include as many observations as possible in the report. This table could also be improved by the addition of row separator lines between each new row variable or data "paragraph." The missing data rows can be deleted and row variable separator lines can be added by using a text editor on the original TABULATE output file. The visual impact of these improvements on the readability of the table is shown in **Output 10**. While the use of a text editor on TABULATE output may seem awkward, it is far less work than importing the data into a spreadsheet or word processor.

Output 10: Improved Supertable After Editing

If You Could Begin Again, Would You Choose to Attend UNCA
(Percent Responding "Yes")

Dimension	Year Graduated				2-Year Total	
	1995		1996			
	N	Pct	N	Pct	N	Pct
Discipline:						
Humanities	55	85.2	38	94.4	93	88.9
Natural Sciences	75	89.7	59	83.1	134	86.6
Professional	122	89.1	90	85.2	212	87.4
Social Sciences	117	83.3	92	83.0	209	83.2
Admission's Type						
Freshmen	172	85.7	130	88.4	302	86.9
Transfer	195	87.6	149	82.4	344	85.3
Cumulative GPA:						
2.00-2.49	51	81.6	34	53.1	85	70.4
2.50-2.99	110	85.7	78	89.7	188	87.4
3.00-3.49	118	85.2	98	91.5	216	88.0
3.50-4.00	88	92.9	69	86.6	157	90.1
Race:						
Other	14	85.7	8	87.5	22	86.4
Black	12	95.0	14	90.0	26	92.3
White	341	87.2	257	86.9	598	87.0
Sex:						
Female	214	89.0	161	87.4	375	88.3
Male	153	83.2	118	82.1	271	82.7
Total	404	86.7	293	84.4	697	85.7

Conclusions

As one of the more widely used general purpose software products, the SAS system provides an ideal tool for survey research applications. SAS can be used to effectively manage all aspects of the survey process from random sampling, generating personalized correspondence, tracking respondents, data analysis, and report writing. This paper has attempted to provide the reader with a number of tips and techniques for conducting and analyzing mail and telephone survey projects. A more complete treatment of this topic is provided in Gravely's 1998 *Your Guide to Survey Research Using the SAS System* published by SAS Institute.

References

- Dillman, Don A. *Mail and Telephone Surveys: The Total Design Method*. New York: John Wiley & Son, 1978.
- Gravely, Archer R. *Your Guide to Survey Research Using the SAS System*. Cary, NC: SAS Institute Inc., 1998.
- Mahon, B.H., "Statistics and Decisions: The Importance of Communication and the Power of Graphical Presentation". *Journal of the Royal Statistical Society (A)*. 1977, 140, pp. 298-323.
- Salant, Priscilla S. and Don Dillman. *How to Conduct Your Own Survey*, New York: John Wiley & Sons, 1994.
- Scheaffer, Richard L., William Mendenhall, and Lyman Ott. *Elementary Survey Sampling*, 2nd Edition. North Scituate, MA: Duxbury Press, 1979.
- Tufte, Edward R. *The Visual Display of Quantitative Information*, Cheshire, Conn: Graphics Press, 1983.

Author Contact

Archer R. Gravely
Office of Institutional Research -CPO 1460
UNC Asheville
One University Heights
Asheville, NC 28804-8503
email: gravely@unca.edu

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.