WUSS 2016 – Paper XXnn

Creating Viable SAS® Data Sets From Survey Monkey® Transport Files

John R Gerlach, Dataceutics, Inc., Pottstown, PA USA

ABSTRACT

Survey Monkey is an application that provides a means for creating online surveys. Unfortunately, the transport (Excel) file from this application requires a complete overhaul in order to do any serious data analysis. Besides having a peculiar structure and containing extraneous data points, the column headers become very problematic when importing the file into SAS. In fact, the initial SAS data set is virtually unusable. This paper explains a systematic approach for creating a viable SAS data set for doing serious analysis.

INTRODUCTION

Survey Monkey is an extremely popular SAAS (Software as a Service) application, having over 15 million users, providing a means for creating all kinds of surveys at little or no cost. After initiating a user account, the application facilitates the implementation of surveys, even offering tips for writing a survey (e.g. avoiding leading questions). The easy-to-use interface allows the user to select from a vast collection of "Expert" survey templates that are organized by category, such as: Demographics, Market Research, and Political; otherwise, the user can start from scratch. In either case, the interface displays a ***Builder*** dialog box that lists several types of survey questions, including: Multiple Choice, Drop Down, Ranking, Open Text, and Date-Time (e.g. date of birth), to name several. There is even a mechanism to install a question that follows from a previous question (e.g. pregnancy status for a female). Besides survey templates and types of questions, the interface has a ***Question Bank*** that contains many typical survey questions. Another impressive feature is the ***Logic*** component that allows the user to determine the flow of the survey, such as Page Skip Logic and Question Randomization. Finally, two other features include general options (e.g. progress bar) and themes (e.g. color or patterns) that produce more attractive surveys. It's a fine application, especially for the price. Of course, obtaining professional help involving the design, implementation, collection, and analysis of a survey requires an upgrade to the user account.

Although the application greatly facilitates the design and implementation of surveys, it produces a data file that is almost worthless for serious data analysis, which is likely to be outside the scope of services provided by the SAAS application. The root cause for this problem is how the application organizes and stores the questions and responses.

A WORD ABOUT SURVEYS

Serious data analysis requires both clear objectives and well-defined data. Unfortunately, questionnaires are often replete with verbose, vague questions resulting in unreliable data. Although the scope of this paper does not include a detailed expose` about surveys, the reader should make note of common issues found in poorly designed surveys:

- The questionnaire has no clear objective.
- The questionnaire is not well-organized, lacking a natural flow, or is not aesthetically pleasing.
- The question has no purpose for the intended analysis.
- The question is ambiguous, even confusing to the respondent.
- The responses are too granular.
- The responses (i.e. values) are not well-suited for data analysis.

Despite possible flaws, given a well-designed survey that produces good data, there's still the issue of creating a viable SAS data set.

THE TRANSPORT FILE

Survey Monkey produces a transport file that contains superfluous data that may be populated during data collection, such as an IP Address, and other data points that are likely to be completely null, such as an Email address. These data points are typically not useful for the subsequent data analysis. Notwithstanding the superfluous data, the naming convention and file structure are such that a complete overhaul is absolutely necessary.

To understand the structure of the transport file, it is expedient to know the Unit of Analysis (UOA), which is simply: *All the responses given by a respondent*. In fact, the Respondent ID, which is generated by the application, explicitly identifies an observation. Besides having a rather naive UOA, the first big problem concerning the transport file concerns the column headers which contain an abridged versionof the questions from the questionnaire. Thus, for the *single-response* question below, the column header would contain the question, albeit with the underscore character supplanting the blanks and discarding other special characters, producing the variable name: **How_old_are_you_**.

1. *How old are you?*
    a. 3 – 17 years
    b. 18 – 25 years
    c. 26 – 50 years
    d. 51+ years

What if the question were verbose? In conformance with SAS syntax rules, the IMPORT procedure limits the variable names to 32 bytes. Nevertheless, the variable names are rather cumbersome. However, there's another problem that makes matters far worse – What Survey Monkey does with *Multiple-response* questions.

Recall the UOA of the raw file – All the responses to a set of questions. Since all the data points reside in a single row in the Excel spreadsheet, the column headers for multiple-response questions contain an abridged version of **both** the question and respective response, which might include HTML tags depending on the design of the questionnaire. The IMPORT procedure truncates enough text to assign an ordinal number representing the *i*th response. And, just to make matters worse, if the question is prefaced with instructional text (e.g. "Based on an overall scale between 1 and 7, please indicate …"), the variable identifier becomes virtually indiscernible. Moreover, imagine a questionnaire that has several verbose multiple-response questions having the same lengthy preface; whereupon, the task of producing a viable data set becomes more daunting. Consider the following question:

2. *Based on an overall scale between 1 and 7, please indicate your interest in < blah, blah, blah >.*
    a. Not at all interested
    b. Somewhat interested
    c. Interested
    d. Very interested
    e. Extremely interested

Obviously, the design of the questionnaire influences the complexity of the transport file. Certainly, more concise and distinguishable questions would help matters. Also, perhaps one could edit the Excel spreadsheet to facilitate the creation of the target SAS data set; but that would be tedious. Let's consider a more programmatic approach.

IMPORT DATA SET

The process of overhauling the transport data file into a viable SAS data set begins with the IMPORT procedure that reads an Excel spreadsheet called Survey that is stored in an Excel file having the same name, as shown below. The procedure creates the initial data set that must be transformed.

```
proc import datafile = '< Path >/Survey.xls'
            out      = survey
            dbms     = xls replace;
   sheet   = 'Survey';
   getname = yes;
run;
```

Using the CONTENTS procedure with the POSITION option offers crucial insight into the initial SAS data set, specifically the difference between single-response versus multiple-response questions. Keep in mind that the UOA denotes *all the responses from one respondent*. Observe the following abridged output from the CONTENTS procedure in Table 1.

#	Variable	Type	Len	Format	Label*
1	RepondentID	Num	8	BEST12.	RepondentID
2	CollectorID	Num	8	BEST12.	CollectorID
3	StartDate	Num	8	MMDDYY10.	StartDate
4	EndDate	Num	8	MMDDYY10.	EndDate
5	IP_Address	Char	15	$15.	IP Address
6	Email_Address	Char	15	$15.	Email Address
7	First_Name	Char	12	$12.	First Name
8	LastName	Char	12	$12.	LastName
9	What_is_your_gender_	Char	6	$6.	What is your gender?
10	What_is_your_age_range_	Char	10	$10.	What is your age range?
11	What_is_your_marital_status_	Char	10	$10.	What is your marital status?
12	What_is_your_education_level_	Char	10	$10.	What is your education level?
13	I_believe_that_our_community_nee	Char	10	$10.	I believe that our . . .
14	Lsat_year_I_attended_classical_c	Char	15	$15.	Last year I attended . . .
15	Last_year_I_attended_classical_1	Char	15	$15.	Last year I attended . . .
16	Last_year_I_attended_classical_2	Char	15	$15.	Last year I attended . . .
17	Please_select_your favorite_comp	Char	50	$50.	Please select your . . .
18	Please_select_your_favorite_com1	Char	50	$50.	Please select your . . .
19	Please_select_your_favorite_com2	Char	50	%50.	Please select your . . .
:	:	:	:	:	:
25	Please_select_your_favorite_com6	Char	50	%50.	Please select your . . .

* The ellipsis denotes the remaining text of the question as determined by the application.

Table 1: Abridged output from the CONTENTS procedure.

Note the following important points about the initial data set:

1. Variable labels denote the actual questions in the survey.
2. Variable identifiers must comply with SAS syntax, thereby supplanting the space and special characters (most notably the question mark) with the underscore.
3. Item 1 uniquely identifies an observation (i.e. UOA).
4. Items 2-8 are superfluous, not needed for the intended analysis.
5. Items 9-13 represent single-response questions. Item 13 is a typical Likert Scale question, having five values: Strongly Agree to Strongly Disagree.
6. The variable identifiers for the remaining items have been truncated to the 32 byte limit.
7. Items 14-16 and items 17-25 actually represent TWO questions, one about attendance and the other about composers. Recall the UOA such that all possible responses reside in a single row.
8. Items 14-16 are identical except for the **last** character of the variable names, which makes them unique. Imagine if there were 50 composers.
9. Items 17-25 are similar to items 14-16; however, these questions are similar to items 17-25; however, this question allows the respondent to select *up to three composers*.

Concerning multiple-choice questions, the column headers in the transport file consist of both the question and the response. For example, items 17-25 that represent favorite composers, one column would have the header:

Please select your favorite composer – Beethoven

Not surprisingly, the cells for that column will contain the answer “Beethoven” for those respondents who select Beethoven as one of three favorite composers. Also, keep in mind that the column headers in the transport file are truncated, thereby often not including the response information for multiple response questions. Regardless, it becomes obvious that the initial data set generated by the IMPORT procedure needs an overhaul such that the UOA and the variable names must be changed, dramatically.

MAPPING THE DATA COLUMN TO THE QUESTION / RESPONSE

There is one very fortunate attribute about the transport file – It has order as defined by the questionnaire, which is crucial for the proposed SAS solution. However, before doing the conversion, it is necessary to have a more suitable (intuitive) naming convention for the variables, as follows:

Q<0>*integer<letter>* where

- 0 (zero) for questions 1-9
- *integer* represents the *i*th question
- *letter* represents the *j*th response (as needed)

The <letter> component is not necessary for single-response questions, obviously. So, in accordance with Table 1, the variable *What_is_your_age_range_* would have the new variable name Q02; whereas, a multiple-response question, as represented by items 14-16, would have the variable identifiers: Q06a, Q06b, Q06c.

An informat VARNUM maps the columns to the respective *i*th question, as shown below (Note: The juxtaposed comments are not part of the PROC step). The respondent identifier is assigned the value zero, which becomes variable 'Q00' in order to distinguish it from actual questions. Superfluous items (e.g. IP address) are not assigned a value and subsequently discarded in the transformation process. Notice that the single response variables, reasonably have a format range of 1; whereas, the multiple-response variables have a range from *n* to *m*, depending on the number of responses for a given question.

```
proc format;
   invalue varnum     1 -  1 =  0          ← Participant Identifier
                      9 -  9 =  1          ← Gender
                     10 - 10 =  2          ← Age range
                     11 - 11 =  3          ← Marital status
                     12 - 12 =  4          ← Education level
                     13 – 13 =  5          ← Community needs classical music
                     14 – 16 =  6          ← Attendance
                     17 – 25 =  7          ← Favorite composers

                       :      : ;

run;
```

Assume that a survey has twenty-five questions of which 10 are single response questions (e.g. gender) and the remaining are multiple response questions. If the multiple response questions each had three responses, then there would be 56 variables (1 + 10 + 45) in the target SAS data set, shown in Table 2.

Variable(s)	**Question(s)**
Q00	Participant Identifier
Q01 – Q10	Single Response Questions 1 – 10
Q11a – Q11c	Question #11
Q12a – Q12c	Question #12
:	: : :
Q24a – Q24c	Question #24
Q25a – Q25c	Question #25

Table 2: Listing of variables denoting survey questions.

As noted earlier, the content of the questions can make it more difficult to discern one question from another, for example, questions having an instructive preface, such as: "*Based on an overall scale from 1 to 7, please indicate your preference of ….* " Despite the length of the questions or its location in juxtaposition with similar looking questions, it is highly recommended to use both the physical survey and the transport file in order to guarantee a proper assignment of each column header.

After completing the task of mapping the column headers to respectively named variables, a similar task must be done to revamp the labels, some of which are indiscernible due to the length of the questions. Here again, it becomes necessary to glean information from both the actual (physical) survey and transport file in order to formulate concise, clear questions. The following SAS code creates the format QTEXT, which stores the edited questions that will be used to generate the respective labels for each variable.

```
proc format;
   value $qtext
      'Q01'  = "What is your gender?"
      'Q02'  = "How is your age range?"
      'Q03'  = "What is your marital status?"
      'Q04'  = "What is your education level?"
      'Q05'  = "Our community needs a symphony orchestra."
      'Q06'  = "Last year, I attended classical concerts by: (Check all that apply)."
      'Q07'  = "Select your favorite composers (As many as three)."

         :             :             :             :

Run;
```

DEFINING NEW VARIABLE IDENTIFIERS

To begin the transformation process, it is necessary to associate one or more variables to a respective question. The SQL step obtains the necessary metadata, specifically the Variable Number (VARNUM). The subsequent Data step performs two important tasks: 1) eliminates unwanted variables (columns 2-9 in the spreadsheet) and assigns the variable QUEST its respective question number.

```
proc sql;
   create table meta01 as
   select varnum, name
      from dictionary.columns
      where libname eq 'WORK' and memname eq 'SURVEY'
      order by varnum;
quit;

data meta02;
   set meta01;
   if 2 le varnum le 9
      then delete;
   quest = input(put(varnum,3.),varnum.);
run;
```

Although the variable QUEST associates a data point to its respective question in the survey, the goal is to define the variable VAR that uses the aforementioned naming convention: **Q<0>***integer*<*letter*>, which identifies each data point more precisely. Assuming that the survey contains less than 100 questions, the following Data step performs By-group processing to accomplish the task. For single response variable, the value of VAR will be simply "Q*nn*" where *nn* denotes a zero-filled 2-digit number, for example: Q02 and Q27. For multiple-response variables, the value of VAR will have a letter-suffix denoting the *i*th possible response for that question, such as: Q11a, Q11b, and Q11c. For each by-group, the variable ITEM is initialized to the value 65, which is the ASCII value for the letter 'a'. The IF statement in the following Data step determines whether there is more than one observation in the by-group; whereupon, the variable VAR is assigned accordingly.

```
data meta03;
   retain item;
   length var $8;
   set meta02;
      by quest;
   if first.quest
      then item = 65;
      else item + 1;
   if first.quest and last.quest
      then var = 'Q' || put(quest,z2.) ;
      else var = 'Q' || put(quest,z2.) || lowcase(byte(item));
run;
```

Using the data set META03, the following Data _null_ step creates pair-wise macro variables that represent the original and new variable names.

```
data _null_;
   set meta03;
   call symput('ovar' || left(put(_n_,3.)),name);
   call symput('nvar' || left(put(_n_,3.)),var);
run;
```

Recall that the original variable name is an abridged mish-mash; whereas, the new variable name follows an intuitive naming convention, as shown in Table 3.

Original Variable			**New Variable**	
OVAR1	What_is_your_gender_	→	NVAR1	Q01
OVAR2	What_is_your_age_range_	→	NVAR2	Q02
OVAR3	What_is_your_marital_status_	→	NVAR3	Q03
OVAR4	What_is_your_education_level_	→	NVAR4	Q04
OVAR5	Our_community_needs_a_symphony_o	→	NVAR5	Q05
OVAR6	Last_year__I_attended____classic	→	NVAR6	Q06
OVAR2	Plesae_select_your_favorite_comp	→	NVAR2	Q07a
OVAR3	Plesae_select_your_favorite_com1	→	NVAR3	Q07b
:	:	:	:	

Table 3: Partial list of values of pairwise macro variables OVAR*i* and NVAR*i*.

The following macro revises the imported data set SURVEY by renaming the original with their new respective variable name. However, first the SQL step defines a macro NVARS, which is needed for the %DO loop as it enumerates the pairwise macro variables (using deferred addressing), expanding the RENAME statement, for as many as there are variables, inside the MODIFY statement of the DATASETS procedure.

```
%macro revise;
   proc sql noprint;
      select count(*) into :nvars from meta03;
   quit;
   proc datasets library=work nolist;
      modify survey;
         rename %do i = 1 %to &nvars.; &&ovar&i.. = &&nvar&i.. %end; ;
   quit;
%mend revise;
```

RESTRUCTURING THE DATA SET

Despite a substantial change to the imported data set with respect to variable naming convention, keep in mind that the UOA has not changed at all. Therefore, for multiple-response questions, the several responses are still stored in separate variables. For example, the question concerning favorite composers would have as many variables as there are possible responses, not to mention the missing values, those variables indicating other less-favored composers per the respondent's taste.

It would be better to normalize the data set so that the single or multiple responses can be identified by one variable representing one question. Thus, the desired UOA would be:

Respondent ID, Question, ith Response

Prior to restructuring the data set, it is expedient to drop those unwanted variables (e.g. Collector_ID, IP_Address). Rather than enumerate these unwanted variables in a DROP statement (or DROP data set option), the SQL step defines a macro variable DELVARS that contains the enumerated list, as determined by the WHERE clause. Because the TRANSPOSE procedure transforms the data set by respondent, it is convenient to insert the macro variable as part of the DROP= data set option of the SORT procedure. Then, the TRANSPOSE procedure restructures the data set, transposing the Q-variables (except Q00, which is the by-group variable denoting the respondent), as shown below.

```
proc sql noprint;
   select name into :delvars separated by ' '
      from meta01
      where 2 le varnum le 9;
quit;

proc sort data=survey(drop=&delvars.) out=surv01;
   by q00;
run;

proc transpose data=surv01
   out=surv02(rename=(q00=id _name_=item  col1=response _label_=quest));
   by q00;
   var q01--q25;
run;
```

Notice the data set option for SURV02 data set that renames the by-group variable Q00 and those variables generated by the TRANSPOSE procedure. Table 4 shows a partial listing of the output data set.

ID	ITEM	QUEST	RESPONSE
1001	Q01	What_is_your_gender	Male
	Q02	What_is_your_age_range	36-50 years
	Q03	What_is_your_marital_status	Single
	Q04	What_is_your_education_level	BA
	Q05	Our_community_needs_a_symphony_o	Strongly Agree
	Q06	Last_year__I_attended __ classic	10
	Q07a	Select_as_many_as_three_favorite	
	Q07b	Select_as_many_as_three_favorit1	Beethoven
	Q07c	Select_as_many_as_three_favorit2	
	Q07d	Select_as_many_as_three_favorit3	Corelli
	:	: : :	:
	Q07m	Select_as_many_as_three_favori13	Schubert
	Q07n	Select_as_many_as_three_favori14	
	Q07o	Select_as_many_as_three_favori15	
	Q08	< Single-response question >	< Response >
	Q09a	< Multiple-response question >	
	:	: : :	:

Table 4: Partial listing of transposed data set.

The transposed data set needs more work, still. The QUEST variable needs to be standardized, thereby supplanting underscore characters and letter-suffixes. Also, the ITEM variable needs to be adjusted so that it denotes a single question, not an individual response to the question. But, there's another important issue regarding multiple-response questions – Those null values for the RESPONSE variable. Question #7 asked the respondent to select three favorite composers out of fifteen, which were listed *alphabetically* on the survey, from Johann Sabastian Bach to Giuseppe Verdi.

FURTHER ADJUSTMENTS

Recall that single response variable do not have a letter suffix; whereas, multiple-response questions can have (for this discussion) up to 26 possible responses (i.e. 26 letters in the alphabet). The task is to truncate those question identifiers, eliminating their letter suffix. However, there's another problem: the ubiquitous "Please explain" part of a question, the so-called verbatim response. For those questions having a supplemental component, the question identifier (value of the variable ITEM) will have the suffix 'x' in order to distinguish it from the set of possible responses, such as *from Bach to Verdi* for the Favorite Compose question.

For this hypothetical survey, questions 10, 14, 15, 16, and 17 have a supplemental component, requesting more information with respect to the responses. Question 10 is technically a single response question; however, it is processed as if it were a multiple response question because of its supplemental component. The other aforementioned questions have several responses. Questions 10 and 14 have as many as five responses (denoted by suffix letters *a, b, c, d, e*) with an additional "Please comment" component, identified by the letter *f*. However, questions 16 and 17 have up to three responses (denoted by suffix letters *a, b, c*), along with a "Please explain" component. The objective is to associate the several discrete responses with a single question identifier (e.g. Q10) and to distinguish those responses from the verbatim component. The Data step accomplishes this task for all questions by processing the supplemental questions, then assigning the variable ITEM, accordingly.

```
data surv03;
   set surv02;
   if substr(item,1,3) eq 'Q10' and item ne 'Q10b'
      then item = substr(item,1,3);
   if substr(item,1,3) eq 'Q14' and item ne 'Q14f'
      then item = substr(item,1,3);
   if substr(item,1,3) eq 'Q15' and item ne 'Q15f'
      then item = substr(item,1,3);
   if substr(item,1,3) eq 'Q16' and item ne 'Q16d'
      then item = substr(item,1,3);
   if substr(item,1,3) eq 'Q17' and item ne 'Q17d'
      then item = substr(item,1,3);
   select(item);
      when('Q10b') item = 'Q10x';
      when('Q14f') item = 'Q14x';
      when('Q15f') item = 'Q15x';
      when('Q16d') item = 'Q16x';
      when('Q17d') item = 'Q17x';
      otherwise;
      end;
   if length(item) gt 3 and not index(item,'x')
      then item = substr(item,1,3);
run;
```

The data set SURV03 has an improved (normalized) structure with a more revised question identifying (ITEM) variable, such as the multiple-response Question 7 (selecting as many as three favorite composers). However, there are still some issues. One obvious issue concerns the text of the survey question, which still distinguishes the individual multiple responses. Referring back to Table 4, the variable QUEST needs to be standardized, similar to the variable ITEM. For example, Question 7 should simply read:

Select as many as three favorite composers.

rather than the text:

Select_as_many_as_three_favori…

The Data step below standardizes the text by using the revised variable ITEM and a format that maps the question identifier (e.g. Q07) to a more readable statement. Unfortunately, there are several more issues to resolve.

```
data surv04;
   length quest $200;
   set surv03;
   quest = put(item,$qtext.);
   if strip(response) in('.')
      then response = "  ";
run;
```

EXTRANEOUS CHARACTERS

For whatever reason, a response might contain extraneous characters (e.g. HTML code). Fortunately, the revised structure of the data set makes it easy to correct the data, if necessary. Upon inspection of a report generated by the FREQ procedure, it becomes obvious that Question 5 has data issues, specifically the Likert scale responses to: *Our community needs a symphony orchestra*. The following Data step uses a SELECT / WHEN statement that can focus on any question having data issues.

```
data surv05;
   set surv04;
   select(item);
      when('Q05') do;
         select(response);
            when('Strongly <br>Agree')      response = 'Strongly Agree';
            when('Somewhat<br>Agree')       response = 'Somewhat Agree';
            when('Somewhat <br>Disagree')   response = 'Somewhat Disagree';
            when('Strongly <br>Disagree')   response = 'Strongly Disagree';
            otherwise;
            end;
         end;
               :      < Other corrections >         :
      otherwise;
      end;
run;
```

EXTRANEOUS OBSERVATIONS

There's one more issue concerning the normalized data set. Certainly, it is conceivable that a respondent did not answer a question on the survey. Moreover, concerning multiple response questions, the respondent chose only a subset of possible answers; thereby, leaving the other component variables null. For example, out of the fifteen variables representing favorite composers, a respondent will have at most three variables populated, having chosen three composers, as shown in Table 5. Although the variables and ITEM and QUEST have been standardized, the extraneous observation having a null response must be deleted.

ID	ITEM	QUEST	RESPONSE
1001	Q07	Select as many as three favorite composers	
1001	Q07	Select as many as three favorite composers	Beethoven
1001	Q07	Select as many as three favorite composers	
1001	Q07	Select as many as three favorite composers	Corelli
1001	Q07	Select as many as three favorite composers	
		: : :	

Table 5: Extraneous observations for Question 7, Favorite Composers.

Concerning unanswered questions, it is imperative to preserve those observations; otherwise, the analysis will not reflect the survey population, that is, to know the distribution of a question with respect to everyone. Only extraneous observations should be deleted, such as the aforementioned question concerning favorite composers and unanswered supplemental (“Please explain”) components.

There are two steps needed to accomplish this task. First, the NODUPKEY of the SORT procedure eliminates any duplicate observations based on the UOA, a respondent’s answer to a particular question, thereby preserving those instances of unanswered questions. However, keep in mind that extraneous null responses are preserved, as well, hence the need for the following Data step that addresses multiple-response questions and supplemental components. It so happens that some of the responses contains a period when it was deemed null, another caveat of the transport data set.

```
proc sort data=surv04 out=surv05 nodupkey;
   by id item response;
run;

data SURVEY;
   length quest $200;
   set surv05;
      by id item;
   response = strip(response);
   if not(first.item and last.item) and response in(' ','.')
      then delete;
   if index(item,'x') and response in(' ','.')
      then delete;
run;
```

THE VIABLE DATA SET

Finally, the SURVEY data set is ready for doing serious data analysis. However, it would be wise to produce several reports to glean more about the data, as well as to make sure it’s correct. Besides the Informat VARNUM and Format $QTEXT that were used to convert survey questions into a more conventional format (pun intended), several other useful formats are defined, as shown below.

```
proc format;
   value nullf    . = ‘Null’
             other = ‘Not Null’;

   invalue varnum     1 -  1 =  0
                      9 -  9 =  1
                     10 – 10 =  2
                     11 – 11 =  3
         :      :       :      : ;

   value $nullf ‘ ‘ = ‘Null’
             other = ‘Not Null’;

   value $noresp ‘ ‘, ‘.’ = ‘< No Response >’
                   other = [$200.];

   value $qtext 'Q01'  = "What is your gender?"
                'Q02'  = "How is your age range?"
                'Q03'  = "What is your marital status?"
         :      :      :      :      :      : ;
run;
```

Preliminary analysis should include an abstract of the data set, a partial listing, and frequency reports, as shown below, that will promote greater understanding of the survey. Keep in mind the UOA and the normalized structure of the data set before proceeding with any analysis. No doubt, more formats will be required to produce good reports. Whether doing exploratory analysis or proving a hypothesis, the SURVEY data set is better suited for the task.

```
proc contents data=survey;
   title2 'Abstract of SURVEY Data Set';
run;

proc report data=survey(obs=120) nowindows headline headskip;
   column id itm quest response;
   define id       / order   width=10         'ID';       ← Respondent ID
   define item     / order   width=5          'Item';     ← Question Identifier
   define quest    / order   width=40  flow   'Question'  ← Survey Question
   define response / order   width=50  flow   'Response'  ← Specific Response
   break after id  / skip;
   title2 'Detailed Listing of SURVEY Data Set';
run;

proc freq data=survey;
   tables item * response / list missing;
   format response $nullf.;
   title2 'Distribution of Responses';
run;
```

CONCLUSION

Survey Monkey facilitates the design and implementation of surveys at a nominal cost. Unfortunately, the transport file from this SAAS application is unsuitable for doing serious data analysis. Thus, a systematic process must be implemented in order to convert the file into a viable SAS data set. This process includes: restructuring the data file, defining intuitive variables, having clear, concise questions, eliminating noise in the responses, and deleting extraneous observations. Once completed, the revised data set is much more suitable for the intended analysis.

REFERENCES

Survey Monkey web site. https://www.surveymonkey.com.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	John R. Gerlach
Affiliation:	Dataceutics, Inc.
E-mail:	JRGerlach0907@gmail.com