

%SHOWCOMB: a macro to produce a data set with frequency of combinations of responses from multiple-response data

Ronald Fehd, Centers for Disease Control, and Prevention, Atlanta GA

ABSTRACT

For the question "(Who, What, Why, When, Where, How)", the answer may be A, or B. Every SAS® programmer wishes the answer were A xor B, because proc FREQ would calculate the answer: A: 17(35.4%), B: 31(64.6%) N=48. Multiple-response data where questions have the phrase "check all that apply" appended to them present a challenge to the SAS programmer because the number of possible response combinations is two to the power of the number of responses. SAS programmers, according to the SASware® Ballot of 1996, desire a proc that provides the answers: A: 17(29.3%) B: 31(53.4%) AB: 10(17.2%) N=58. This paper examines the SAS proc FREQ output data set from a cross-tabulation and discusses the issues in constructing a similar data set for multiple-response data with one variable containing the combination of responses. Issue related to labeling, storage and type of multiple-response variables are discussed.

The SHOWCOMB macro takes as parameters an output data set name which is the prefix of a series of variables containing the multiple-response data. Optionally it may take a list of the multiple-response variables.

Note: macro SHOWCOMB is a one of two papers presented at SESUG that address multiple-response data: see CHECKALL. Macro ARRAY is a utility used by both of the macro procedures CHECKALL and SHOWCOMB.

INTRODUCTION

Simple questions may have complex answers when the question contains the phrase "Check all that apply". This paper reviews the output data set of a proc FREQ cross-tabulation of a series of variables. This data set is then used as a model to construct a macro routine which produces a standardized data set with the frequencies of the combinations of variables in the series of multiple-response data.

The Answers: (Check all that apply)

```
A:Apple    >----> box.for.answer(_)
B:Banana   >----> box.for.answer(_)
C:Cherry   >----> box.for.answer(_)
```

Common values used for the meaning of 'checked' include: (Y,N), (T,F), etc. The example data uses numeric (1,0). A proc FREQ cross-tabulation is the easy first step in examining multiple-response data. Our proposed process requires saving the output data set.

Program 1

```
data QUERIES;
label Q02A = 'Apple'
      Q02B = 'Banana'
      Q02C = 'Cherry';
input Q02A Q02B Q02C;
cards;
;
proc FREQ data = QUERIES;
tables Q02A * Q02B * Q02C
/ list noprint out = FREQ;

proc PRINT data = FREQ label;
```

```
SAS output
-----
Frequency      Percent of
                Total
```

OBS	Apple	Banana	Cherry	Count	Frequency
1	0	0	1	1	8.3333
2	0	1	0	2	16.6667
3	0	1	1	3	25.0000
4	1	0	0	3	25.0000
5	1	1	0	2	16.6667
6	1	1	1	1	8.3333

Our output is raw data: the values representing 'checked' and 'not checked' must be mentally translated while reading. Our next task is to replace the value for 'checked' in each variable with the variable label. This requires a new set of variables which are character with length of 40, the allowed length of labels. 'Not checked' is irrelevant and is changed to blank.

example intermediate output

OBS	Label Q02A	Label Q02B	Label Q02C	Frequency Count	Percent of Total Frequency
1			Cherry	1	8.3333
2		Banana		2	16.6667
3		Banana	Cherry	3	25.0000
4	Apple			3	25.0000
5	Apple	Banana		2	16.6667
6	Apple	Banana	Cherry	1	8.3333

The last step is to concatenate the series of variables into one variable, compress, and delimit the labels with a comma. The data is sorted by descending Count.

example desired output

OBS	Combinations of Q02	Frequency Count	Percent of Total Frequency
1	Apple	3	25.0000
2	Banana, Cherry	3	25.0000
3	Apple, Banana	2	16.6667
4	Banana	2	16.6667
5	Apple, Banana, Cherry	1	8.3333
6	Cherry	1	8.3333

Constraints of using cross-tabulation

When processing a cross-tabulation, SAS must allocate a matrix of N columns, where N is the number of variables. The number of matrix rows allocated is $O(2^{**}N)$ if data is binary-valued: (0,1) and $O(3^{**}N)$ if data contains missing (0,1,.). An additional consideration is the width of each column. The minimum space available for numeric variables is 2 or 3 bytes, depending on operating system. For multiple-response data with large numbers of variables, some optimization of both data storage and matrix size is necessary.

Numeric binary-valued data where the data is either zero, one or missing may be more effectively stored as character in one byte. The cost of this storage efficiency is that the data must be converted to numeric for usage in many SAS procedures.

Further efficiency may be realized by converting a series of binary-valued variables into an integer where each bit represents one variable. Again, the cost is conversion from compressed data to individual variables for analysis.

A major benefit of using an integer to store multiple-response data is that the width of the cross-tabulation matrix is reduced to one column. This reduces the chance that a production routine would fail.

Data compression

Each bit in an integer can be changed from zero to one on the condition that a contributing value is true. Our macro routine has a parameter TRUE which can be changed to accommodate a value of numeric or character one. Each element in the array is tested for &TRUE and the bit changed accordingly. The exponent of 2 is the array dimension minus the index which changes bits from left to right, reflecting the left to right pass through the array. This allows the programmer to compare the variable values with the integer produced.

Macro code excerpt 1

```
%LET TRUE = 1;
%LET DIM_VAR = 3;

%*3: data: prepare subset of DATA and create Number for FREQ
which is the binary-value of all the variables with value = &TRUE;
DATA ZBINNMNR;
set DATA;
array CheckAll (")$VAR_LIST;
N = 0;
do I = 1 to dim(CheckAll);
if CheckAll{I} = &TRUE, then N = sum(N, 2**(DIM_VAR. - I));
/*do I */ end;
```

Saving the labels

In this step the labels of the series are saved to an array of macro variables.

Macro code excerpt 2

```
%*3.2 create array of Labels of series;
length Label $ 40; drop Label;
%do I = 1 %to DIM_VAR;
call label($VARLIST,Label);
call symput("LBL&I.",trim(left(Label))); %END;
```

The data is now prepared for proc FREQ of a single variable: the integer containing the data from the series of variables. An output data is saved for the decompression step: ZFRQCOMB.

Decompression

Changing the integer to a character variable of combinations of labels is a two-step process. First the integer is changed to a character variable -- BinStrng -- containing zeros and ones. Then this binary string is used to concatenate the labels into one variable -- Label -- which contains the combinations.

In order to save data for large series, where more than 200 characters are needed for the combinations, a second process is carried out. An array of Labels is prepared. Lbls{I} always contains the same variable label or is blank. A shorter array of Columns is prepared -- Cols{I} -- whose dimension is equal to the maximum number of items checked in the series. Each non-blank Lbls{I} is moved to next empty Cols{I}. Cols{I} will always contain one label, though it may be different in each combination.

Macro code excerpt 3

```
%*6. data: recode FREQ output: convert Number to Combinations;
DATA ZFRQCOMB;
array Lbls (")$ Lbl1-LblDIM_LBL.;
array Cols (")$ Col1-Col$MAXCHKD.;
set ZBINNMNR;
Delimitr = ' '; /*change to ' ' after first item*/
%*6.1 change Number to binary string
loop: change binary string to Label and Lbl*;
BinStrng = put(N,binary&DIM_LBL.);
%do I = 1 %to DIM_LBL.; /*-----*/
if substr(BinStrng,&I,1) = '1'
then do; Label = left(trim(Label) || Delimitr || "&LBL&I.");
Delimitr = ' ';
Lbls{I} = "&LBL&I."; end;
/*-----*/ %do I =1:DIM_LBL*/ %END;
if length(Label) = 200 then Label = ' ' || substr(Label,1,199);
EmptyCol = 1; /*fill Cols from Lbls*/
do I = 1 to dim(Lbls); /*-----*/
if Lbls{I} ne ' ' then do; Cols{EmptyCol} = Lbls{I};
EmptyCol = EmptyCol + 1; end;
/*-----*/ do I=1:dim(Lbls)*/ end;
```

CONCLUSION

When analyzing multiple-response data, a cross-tabulation from proc FREQ serves as a model to present the data. Users prefer to be able to read the labels of variables in each combination. Compressing the data saves system resources and allows large number of variables to be processed in a similar manner. When the data is decompressed, variable labels can be substituted for the variable values.

REFERENCES

Fehd, Ronald, %ARRAY: construction and usage of arrays of macro variables, SESUG 96 Atlanta GA

Fehd, Ronald, %CHECKALL: a macro to produce a frequency of response data set from multiple-response data, SESUG 96 Atlanta GA

ACKNOWLEDGMENTS

These routines were developed over a period of five years while I crunched the numbers of survey data collected by the Model Performance Evaluation Program (MPEP) of the Division of Laboratory Systems, Public Health Practice Program Office of the Centers for Disease Control, and Prevention, Atlanta, Georgia. John Hancock, chief, Information Services Activity, encouraged me to write up these routines. I wish to thank Sharon Blumer, David Cross, Thomas Hearn, and William Schallia of the MPEP group for repeating the questions about the data often enough and in enough variation that I was finally able to comprehend the underlying pattern.

SAS is a registered trademark of SAS Institute, Inc. In the USA and other countries, © indicates USA registration.

```
/******
* MACRO: SHOWCOMB
* USAGE: 1) %SHOWCOMB(series-name);
*          *assumes CHECKALL data set exists for this series
*          2) %SHOWCOMB(series-name,LIST=var1 var2 var3 .. varN);
*          3) %SHOWCOMB(series-name,DATA=dataset);
*          4) %SHOWCOMB(series-name,TRIMCHAR=-);
*          5) %SHOWCOMB(series-name,PRINT=1);
*          6) %SHOWCOMB(series-name,BY_VAR=var-name);
*          7) %SHOWCOMB(series-name,TRUE='Y');
* DESCRIPTION: Processing of Check-all-that-apply questions
*               output dataset contains Combinations, Count, Percent
*               and Columns containing each item of Combination
******/
```

```

* PROCESS:
* 1: macro: prepare ARRAY of variables
* 2: macro: concatenate elements into VAR_LIST
* 3: data: prepare subset of DATA and create Number for FREQ
* 4: proc: if BY_VAR present, then SORT data
* 5: proc: FREQ of Number to output dataset
* 6: data: recode FREQ output: convert Number to Combinations
* 7: data: save optimized dataset to library
* 8: proc: if wanted CONTENTS and/or PRINT of saved data set
* 9: if LIST ne CHECKALL-DATA: process as in CHECKALL macro
* NOTES: datasets created by %CHECKALL are named &SERIES.
*       these are input data sets to %ARRAY
* KEYWORDS: APPEND array %ARRAY binary-coded call label()
*           data compression dim() dimension FREQ left()
*           multiple-response data put() trim()
*****/
%MACRO SHOWCOMB(
  SERIES /* name of series of variables, output data set prefix */
  /* output data set name: &SERIES.COMB */
  LIST=DATA /* list of variables */
  /* default is DATA previously prepared */
  /* whose name is &SERIES: output from %CHECKALL macro */
  LIBRARY=LIBRARY /* library name */
  DATA=DATASET /* DATASET is global variable, else hardcode here */
  /* master DATA set name */
  /* DO NOT USE &SERIES AS NAME OF THIS PARAM */
  LBL_LBL=Combinations of &SERIES /* label of variable label */
  LBLCOUNT=Number of Laboratories Responding /*
  /* label of Freq-Count */
  LBLPCENT=Percentage of Laboratories Responding /*
  /* label of Freq-Percent */
  BY_VAR= /* var for subsetting */
  FORMAT= /* format of single by-variable, use dot at end of format */
  CHARTYPE=GOODBARH /* Chart-Type in (barh barv tbl) */
  PRINT=1 /* WantPrint output? suppresses printing */
  TESTING=1 /* TESTING=1 enables explanatory printouts */
  TITLE=TEMP /* label of output dataset and title for graphics */
  /* TITLE cannot contain commas */
  TRIMCHAR= /* front-trim label to this char */
  /* labels expected to be in form:
  /* "Q048: Category - Specific"
  /* TRUE=1 /* variable counted with this value: numeric one
  /* TRUE=1 /* variable counted with this value: character one
  /* TRUE=Y /* variable counted with this value: character 'Yes'
  );
  %1: macro: prepare ARRAY of variables;
  %IF "&LIST"="DATA" %THEN /* use CHECKALL dataset */
    %ARRAY(VAR,DATA=&LIBRARY.&SERIES,VAR=Variable);
  %ELSE %ARRAY(VAR,&LIST);
  %2: macro: concatenate elements into VAR_LIST
  NOTE: LIST is tested below;
  %MACRO VAR_LIST; %DO I = 1 %TO &DIM_VAR; &SARV&I. %END;
  %LOCAL VAR_LIST;
  %LET VAR_LIST = %VAR_LIST;
  %IF &BY_VAR ne %THEN %DO; TITLE4 "&SERIES. by &BY_VAR."; %END;
  %ELSE %DO; TITLE4 "&SERIES."; %END;
  %3: data: prepare subset of DATA and create Number for FREQ
  which is the binary-value of all the variables with value = &TRUE;
  DATA ZBINMMBR;
  retain MaxChkd MnbrResp 0;
  do until(EndOfFile);
    set &LIBRARY.&DATASET;
    (keep = &BY_VAR. &VAR_LIST.)
    end = EndOfFile nobs = MnbrObs;
    MnbrChkd = 0;
    array CheckAll {*} &VAR_LIST;
    N = 0;
    do I = 1 to dim(CheckAll);
      if CheckAll{I} = &TRUE then do; N = sum(N, 2**(&DIM_VAR. - I));
        MnbrChkd + 1; /*do I */ end;
      if MnbrChkd then do; output;
        MaxChkd = max(MaxChkd,MnbrChkd);
        MnbrResp + 1; /*if MnbrChkd*/ end;
      /*do until(EndOfFile)*/ end;
    end;
  %3.2 create array of labels of series;
  %local DIM_LBL; %LET DIM_LBL = &DIM_VAR;
  length Label $ 40; drop Lbl;
  NwLabel = 0; /*save Maximum Width of Label*/
  %DO I = 1 %TO &DIM_VAR;
    call label(&SARV&I.,Label);
    if index(Label,"&TRIMCHAR") then Label = left(substr(Label,
      index(Label,"&TRIMCHAR")+1));
    call symput("LBL&I.",trim(left(Label)));
    NwLabel = max(NwLabel,length(Label)); %END;
  %3.3 create mac-vars of;
  %local NWLABEL MAXCHKD MMBROBS MMBRRESP PCHTRESP;
  call symput("NWLABEL",trim(left(put(NwLabel,2.))));
  call symput("MAXCHKD",trim(left(put(MaxChkd,8.))));
  call symput("MMBROBS",trim(left(put(MnbrObs,8.))));
  call symput("MMBRRESP",trim(left(put(MnbrResp,8.))));
  stop; run;
  %put NWLABEL = <&NWLABEL.>;
  %put MAXCHKD = <&MAXCHKD.>;
  %put MMBROBS = <&MMBROBS.>;
  %put MMBRRESP = <&MMBRRESP.>;
  %LET PCHTRESP = %val(100* &MMBRRESP / &MMBROBS);
  %put PCHTRESP = <&PCHTRESP.>;
  %4. proc: if BY_VAR present, then SORT data;
  %IF &BY_VAR. ne %THEN %DO;
    proc SORT data = ZBINMMBR;
      by &BY_VAR.; %END;
  %5. proc: FREQ of Number to output dataset;
  proc FREQ data = ZBINMMBR order = FREQ;
    tables N / out = ZFRCOMBS noprint;
    %IF &BY_VAR. ne %THEN %DO;
      by &BY_VAR.; %END;
  %6. data: recode FREQ output: convert Number to Combinations
  convert Number to binary string
  convert binary string to user-readable string in var Label $ lbl
  NOTE: long combinations may be truncated: max SAS char width is 200
  noted with asterisk in front of combinations;
  DATA ZFRCOMBS;
  drop Delimtr EmptyCol I
  Lbl1-LblDIM_LBL.
  %IF not &TESTING %THEN N BinStrng;
  WLabel WCount WPercent
  WCol1-WCol&MAXCHKD.;
  length Label $ 200
  Delimtr $ 2
  BinStrng $ &DIM_LBL.
  %DO I = 1 %TO &DIM_LBL.; Lbl&I. %END;
  %DO I = 1 %TO &MAXCHKD.; Col&I. %END;
  Col&val(&MAXCHKD. + 1) $ %length(&MMBRRESP.) /*: Count */
  Col&val(&MAXCHKD. + 2) $ 5 /*: Percent*/ ;
  array Lbls {*} $ Lbl1-LblDIM_LBL.;
  array Cols {*} $ Col1-Col&MAXCHKD.;
  array WCol {*} WCol1-WCol&MAXCHKD.;
  /* save to create mac-vars */
  retain WLabel WCount WPercent
  %DO I = 1 %TO &MAXCHKD.; WCol&I. %END;
  0;
  do until(EndOfFile); /*-----*/
    set ZFRCOMBS end = EndOfFile;
    /* initialize arrays */
    do I = 1 to dim(Lbls); Lbls{I} = ' '; end;
    do I = 1 to dim(Cols); Cols{I} = ' '; end;
    Label = ' ';
    Delimtr = ' '; /*change to '.' after first item*/
    %6.1 change Number to binary string
    loop: change binary string to Label and Lbl;
    BinStrng = put(N,Binary&DIM_LBL.);
    %DO I = 1 %TO &DIM_LBL.; /*-----*/
      if substr(BinStrng,&I,1) = '1'
        then do; Label = left(trim(Label)!! Delimtr !! "LBL&I.");
          Delimtr = ' ';
          Lbl&I = "LBL&I."; end;
    /*-----*/
    %DO I = 1 %TO &DIM_LBL.; %END;
    if length(Label) = 200 then Label = ' ' !! substr(Label,1,199);
    EmptyCol = 1; /*fill cols from Lbls*/
    do I = 1 to dim(Lbls); /*-----*/
      if Lbls{I} ne ' ' then do; Cols{EmptyCol} = Lbls{I};
        WCol{EmptyCol} = max(WCol{EmptyCol},
          length(Cols{EmptyCol})); %END;
    /*avoid error msg: */

```

```

/* "array subscript out of bounds" */
if EmptyCol lt MAXCHKD, then EmptyCol = EmptyCol + 1; end;
/*..... do I=1:dim(Lbls)*/ end;
/*copy numeric Count, Percent to character vars: Col;
Col%val(MAXCHKD, + 1) = put(Count, length(SMMRRESP),0);
Col%val(MAXCHKD, + 2) = left(put(Percent, 5.1));
/*6.2 set-up for mac-vars: widths of variables, note all are retained;
WLabel = max(WLabel, length(Label));
WCount = max(WCount, length(trim(trim(put(Count, 8.0)))));
WPercent = max(WPercent, length(trim(trim(put(Percent, 5.1)))));
output; /*.....do until(EndOfFile)*/ end;
/*6.3 create mac-vars of widths;
%local WLabel WCount WPercent;
call symput('WLabel', trim(trim(put(WLabel, 3)))));
call symput('WCount', trim(trim(put(WCount, 8)))));
call symput('WPercent', trim(trim(put(WPercent, 5)))));
%do I = 1 %to MAXCHKD.;
%local WCOL1.;
call symput('WCOL1.', trim(trim(put(WCOL1, 2))))); %END;
stop; run;
%local LIB; %LET LIB = LIBRARY;
%IF &TESTING &THEN &DO;
%LET LIB = WORK;
%PUT WLabel = <WLabel> WCount = <WCount> WPercent = <WPercent>;
%do I = 1 %to MAXCHKD.; %PUT WCOL1. :: &WCOL1.; %END;
/*%IF &TESTING %*/ %END;
%local SAVEName; %LET SAVEName = &SERIES.CMS;
%IF &length(SAVEName) gt 8 %THEN %LET SAVEName=%substr(SAVEName,1,8);
/*7. data: save optimized dataset to library;
DATA &LIB..&SAVEName. (label =
%local LEN;
%LET LEN = %length(&TITLE.);
%IF &LEN. le 30 %THEN "SHOWCOMB ATTITLE.";
%ELSE "SHOWCOMB %substr(ATTITLE,1,30)"; );
attrb
%IF &BY_VAR ne %THEN &DO;
Subst label = "subst: &BY_VAR" %END;
Title label = "&SERIES. Title"
length = $ &LEN. format = Schar&LEN..
N_eq label =
"=SMMRRESP data:&DATA Obs:&MMRRES Resp:&PCNTRESP."
%LET LENRESP = %val(2 + %length(SMMRRESP));
length = $ LENRESP. format = Schar&LENRESP..
Chartype label = "SCHARTYPE"
length = $ 8 format = Schar8.
Label label = "LBL_LBL"
length = $ &WLabel. format = Schar&WLabel..
Count label = "BLBLCOUNT." format = &WCount..0
Percent label = "BLBLPCENT." format = &WPercent..1
%do I = 1 %to MAXCHKD.;
label = "Col &I."
length = $ &WCOL1. format = Schar&WCOL1... %END;
/*PC: three dots in ScharN. mainframe ??? ... */
Col%val(MAXCHKD, + 1)
label = "Count Col %val(MAXCHKD, + 1)"
length = $ %length(SMMRRESP)
format = Schar%length(SMMRRESP)..
Col%val(MAXCHKD, + 2)
label = "Percent Col %val(MAXCHKD, + 2)"
length = $ &WPercent. format = Schar&WPercent..
;
retain Title "&TITLE."
N_eq "N=SMMRRESP."
Chartype "SCHARTYPE";
do until(EndOfFile);/*.....*/
set ZFRCCOMB
%IF &BY_VAR ne %THEN &DO;
(rename=(&BY_VAR = Subst)) %END;
end = EndOfFile;
Label = translate(Label, ',', '');/*change <I> back to quote*/
***NmrResp = 100*Count/Percent;
output;
Title = ' ';
N_eq = ' ';
Chartype = ' ';/*.....do until(EndOfFile)*/ end;
stop;
/*8. proc: if wanted CONTENTS and/or PRINT of saved data set;
%IF &TESTING &THEN &DO; proc CONTENTS data = &SYSLAST; %END;
%IF &PRINT &THEN &DO;/*.....*/
proc PRINT data = &LIB..&SAVEName double nobbs label;
var
%IF &TESTING &THEN N BinStrng ;
%IF &WLabel gt 100 %THEN Col; ;
%ELSE Label Count Percent ; ;
%IF &BY_VAR ne %THEN &DO;
by Subst:&BY_VAR.;
id Subst:&BY_VAR.;

```

```

%IF &FORMAT. ne %THEN &DO;
format &BY_VAR. &FORMAT.; %END;
/*%IF BY-VAR not missing*/ %END;
/*..... %IF PRINT .....*/ %END;
/*9. if LIST ne CHECKALL-DATA: process as in CHECKALL macro;
%IF &LIST. = "DATA"
and &BY_VAR = %THEN &DO; %PUT EXIT: LIST = CHECKALL DATA; %END;
%ELSE &DO;%code similar to CHECKALL macro deleted for space;
/*..... %ELSE LIST ne CHECKALL-DATA %*/ %END;
%ENDSTEP: run;/*.....SHOWCOMB..*/ %END;
/*- SHOWCOMB test data .....*/
%LET DATASET = SURVEY2;
options details nprint nocenter;
%libname LIBRARY 'c:\saswipd\sasuser';/*default;
*Step 1: label the variables;
DATA LIBRARY.SURVEY2;
label
004A = "004a: Primary Classification"
004BCOM = "004b: Blood Bank - Community"
004BREG = "004b: Blood Bank - Regional"
004BPLA = "004b: Blood Bank - Blood/Plasma center"
004BANC = "004b: Blood Bank - American Red Cross"
004BPRI = "004b: Blood Bank - Privately owned"
004BMIL = "004b: Blood Bank - Military (Federal)"
004BNOS = "004b: Blood Bank - Hospital blood bank"
004BOTH = "004b: Blood Bank - Other"
004CCIT = "004c: Hospital - City"
004CCNT = "004c: Hospital - County"
004CSTA = "004c: Hospital - State"
004CDIS = "004c: Hospital - District"
004CCOM = "004c: Hospital - Community"
004CREG = "004c: Hospital - Regional"
004CMIL = "004c: Hospital - Military (Federal)"
004CNET = "004c: Hospital - Veterans Administration"
004CPRI = "004c: Hospital - Privately owned"
004CUNI = "004c: Hospital - University"
004CHMO = "004c: Hospital - HMO owned & operated"
004CREL = "004c: Hospital - Religious associated"
004COTH = "004c: Hospital - Other"
; input
P 1 004A Schar1.
S 3 004BCOM Schar1. S 4 004BREG Schar1. S 5 004BPLA Schar1.
S 6 004BANC Schar1. S 7 004BPRI Schar1. S 8 004BMIL Schar1.
S 9 004BNOS Schar1. S10 004BOTH Schar1.
S11 004CCIT Schar1. S12 004CCNT Schar1. S13 004CSTA Schar1.
S14 004CDIS Schar1. S15 004CCOM Schar1. S16 004CREG Schar1.
S17 004CMIL Schar1. S18 004CNET Schar1. S19 004CPRI Schar1.
S20 004CUNI Schar1. S21 004CHMO Schar1. S22 004CREL Schar1.
S23 004COTH Schar1.; cards;
S 10000010 00000000000000
N 00000000 00001100000000
N 00000000 11000000000000
N 00000000 00001100000000
S 11000000 00000000000000
N 00000000 00000001000000
S 11000000 00000000000000
N 00000000 00001100000000
S 10000000 00000000000000
S 10000010 00000000000000
N 00000000 00000001000000
N 00000000 00000000000000
N 00000000 00000000000000
;
/*Step 2: save CONTENTS of data set;
proc CONTENTS data = LIBRARY.&DATASET. noprint
out = LIBRARY.CONTENT(S(keep = Name));
*Step 3: create and save data sets with series of variables;
data LIBRARY.V004 LIBRARY.V004B LIBRARY.V004C;
set LIBRARY.CONTENT(S;
if substr(Name,1,3) = '004' then output LIBRARY.V004;
if substr(Name,1,4) = '004B' then output LIBRARY.V004B;
if substr(Name,1,4) = '004C' then output LIBRARY.V004C;
/* end CHECK-ALL set-up .....*/
*Step 4: run %CHECKALL:
output dataset contains only variables used in the series,
used as input for SHOWCOMB;
%CHECKALL(004); %CHECKALL(004B,TRIMCHAR=); %CHECKALL(004C,TRIMCHAR=);
/* end SHOWCOMB SGP set-up .....*/
%LET DATASET = SURVEY2;name of input data set;
*show intermediate vars;:"SHOWCOMB(004B,&BY_VAR=004A,TRIMCHAR=,TESTING=1);
*SHOWCOMB(004B,TRIMCHAR=);
*SHOWCOMB(004C,TRIMCHAR=);
*user-provided list;
*SHOWCOMB(004B,LIST = 004BCOM 004BPLA 004BPRI 004BREG,TRIMCHAR=);
*SHOWCOMB(004,TRIMCHAR=);/*.....end test data %*/

```

Author: Ronald Fehd
Centers for Disease Control
4770 Buford Hwy NE MS-G25
Atlanta GA 30341-3724

voice: 770/488-4316
fax: 770/488-7667
E-mail: RJF2@phpdLs1.em.cdc.gov