Paper DV07-2013

A Concise Display of Multiple Response Items

Patrick Thornton, SRI International, Menlo Park, CA

ABSTRACT

Surveys often contain multiple response items, such as language where a respondent may indicate that she speaks more than one language. In this case, an indicator variable (1=Yes, 0=No) is often created for each language category. This paper shows how a concise tabulation of the count and percent of respondents with a "Yes" on one or more indicator variables may be obtained using PROC TABULATE and a MULTILABEL format. A series of indicator variables is used to create a binary variable and its base-10 equivalent, and a MULTILABEL format is created to properly aggregate observations with a "Yes" on two or more indicator variables. The BAND function may also be used to easily subset observations with "Yes" responses on certain combinations of the indicator variables.

INTRODUCTION

Surveys often contain multiple response items, such as language where a respondent may indicate that she speaks more than one language. In this case, an indicator variable (1=Yes, 0=No) is often created for each language category. The most obvious way to obtain the count and percent of "Yes" respondents is to use SAS® PROC FREQ on each of the indicator variables, but this produces multiple tables (left side of Figure 1). A PROC TABULATE and a MULTILABEL format may also be used to produce a more concise table (right side of Figure 1).

Figure 1 A Comparison of Results from PROC FREQ and TABULATE

Spanish

e1	Frequency	Percent	Cumulative Frequency	Cumulative Percent
No	6	60.00	6	60.00
Yes	4	40.00	10	100.00

English

e2	Frequency	Percent	Cumulative Frequency	Cumulative Percent
No	4	40.00	4	40.00
Yes	6	60.00	10	100.00

Chinese

e3	Frequency	Percent	Cumulative Frequency	Cumulative Percent
No	5	50.00	5	50.00
Yes	5	50.00	10	100.00

PROC TABULATE with MULTILABEL format	N	%
Language		
Missing	1	10.0
Spanish	4	40.0
English	6	60.0
Chinese	5	50.0
Total	10	100.0

The use of the MULTILABEL format with PROC TABULATE allows the correct count of observations, 10, to be shown and used in the percentage calculations while also allowing observations to count toward more than one language category. Notice that the percentages in the table on the right match those in the PROC FREQ and sum of the counts is actually higher than 10. The following sections show how I achieved these results.

CREATING A BINARY VARIABLE FROM A SERIES OF INDICATOR VARIABLES

The following shows an example data set with a series of indicator variables (E1-E3) storing responses to an item asking "What language(s) do you speak?"

```
proc format;
   value yesno 1 = 'Yes' 0 = 'No';
   value gender 1 ='Male' 2 = 'Female';
run;
```

```
data mydata;
   /*create the variables*/
   length  e1-e3 8. language_binary $3.;
    /*Label the indicators of language spoken. Label a gender variable.*/
   label e1 = 'Spanish' e2='English' e3='Chinese' gender='Gender';
   /*read in the data*/
   input e1-e3 gender;
   /*create a binary variable representing language responses*/
   ❶language_binary = strip(catt(of e3 e2 e1));
   format gender gender. e3 e2 e1 yesno.;
cards;
0 0 0 1
0 0 1 2
0 1 0 1
1 1 0 2
1 0 1 1
0 1 1 2
0 1 0 2
1 0 0 1
0 1 1 1
1 1 1 2
;
run;
❷ proc sort data=mydata; by language_binary; run;
title1 'Indicators Variables and Binary Variable';
proc print data=mydata label;
 var e3-e1 language_binary;
run;
```

❶ A variable, LANGUAGE_BINARY, is created from the concatenation of the 3 language indicator variables in reverse order. Reverse order is uses because binary variables grow from right to left, thus allowing the addition of more indicators without changing the decimal representation of earlier indicators (see later section).

❷ The data set is sorted by LANGUAGE_BINARY and is printed as shown in Figure 2

Figure 2 Listing of the Indicators and Binary Variable

Obs	Chinese	English	Spanish	language_binary
1	No	No	No	000
2	No	No	Yes	001
3	No	Yes	No	010
4	No	Yes	No	010
5	No	Yes	Yes	011
6	Yes	No	No	100
7	Yes	No	Yes	101
8	Yes	Yes	No	110
9	Yes	Yes	No	110
10	Yes	Yes	Yes	111

Figure 2 shows that LANGUAGE_BINARY stores the pattern of 1s and 0s that represents the “Yes” and “No” responses of all three indicator variables. Some of the patterns contain a single “1” (e.g. 001) and some contain two or three 1s (e.g. 011, 111).

OBSERVATIONS MISSING RESPONSES ON INDICATOR VARIABLES

Notice that the binary form of the indicator variables may only be created when all indicator variables have a 0 or a 1. If you have an observation where you cannot assume that a missing response is a “no,” and therefore set indicators to 0, then you would have to exclude the whole observation from the summary. If you have a lot of observations with missing values on the indicators, the decision to include or exclude may have profound effects on the percentages. Also, if you are using this technique with a series of indicator variables that do not come from the same multiple response item (e.g. 3 indicators of economic risk), then there may be the need to have a different denominator when calculating the percent for each item. In this case, you cannot use the technique shown in this paper.

The next sections shows how we can use LANGUAGE_BINARY to both summarize and subset observations based on responses to all 3 indicator variables.

SUMMARIZING: CREATING THE BASE-10 OF THE BINARY VARIABLE AND A MULTILABEL FORMAT

We could create a MULTILABEL format using the binary numbers:

```
proc format;
   value $langf (❶multilabel notsorted )
    '000' = 'Missing'
    '001','011','101','111' = 'Spanish'
     '010','011','110','111'= 'English'
     '100','101','110','111'='Chinese';
run;
```

❶ The MULTILABEL option allows us to assign the same binary number to different labels on the right, for example, '111' is assigned to Spanish, English and Chinese. The format could then be used with PROC TABULATE to produce the output in Figure 1.

```
proc tabulate data=mydata  format=10.0;
  class language_binary /ORDER=DATA PRELOADFMT ❶MLF MISSING;
   table language_binary='Language' all, (n pctn='%'*format=5.1)
  /box='One Table with MLF to Report Counts and Percents' rtspace=50;
     format language_binary $langf. ;
run;
```

❶ The option MLF allows PROC TABULATE to use the format to count observations in two or more of the formatted categories of Spanish, English and Chinese while the total shown is the count of observations, not the count of responses.

The disadvantage of using the binary numbers in the format is that changing the number of indicator variables will cause all values of the binary variable to change. For example, the following is the format that is needed in the case of having just an English and Spanish indicator.

```
proc format;
value $langf (multilabel notsorted)
'00' = 'Missing'
'01','11' = 'Spanish'
'10','11', = 'English';
Run;
```

Fortunately, we can create a decimal or base-10 equivalent of the binary variable whose values do not all change when a different number of indicator variables are involved. The following table shows that as the number of languages in the response item changes the binary representation also changes, but the decimal equivalent remains the same. Thus, for example, Spanish 'Yes" may consistently be represented as a 1, 3, 5, and 7 (shown in grey).

Table 1 Example of a Constant Decimal Representation

One Language	Binary One	Two Languages	Binary Two	Three Languages	Binary Three	Decimal Equivalent
Spanish	1	Spanish	01	Spanish	001	1
		English	10	English	010	2
		English, Spanish	11	English, Spanish	011	3
				Chinese	100	4
				Chinese, Spanish	101	5
				Chinese, English	110	6
				Chinese, English, Spanish	111	7

The following syntax is used to create the decimal equivalent of our binary variable:

```
data mydata;
  set mydata;
   /*create a base-10 equivalent representing language responses*/
   language_profile=input(language_binary,binary3.);
run;
```

I created the following MULTILABEL format to use with LANGUAGE_PROFILE.

```
Proc format;
     value langf (multilabel notsorted )
     0 = 'Missing'
     1,3,5,7 = 'Spanish (..1)'
     2,3,6,7= 'English (.1.)'
     4,5,6,7= 'Chinese (1..)';
Run;
```

Notice that the format can be easily reused for the case of reporting only Spanish and English indicators by simply removing the Chinese category. There is no need to change the number used for the Spanish and English categories. The numbers used to indicate Spanish and English in a LANGUAGE_PROFILE variable would be 1, 2, 3 and those numbers are already in the previous format used to represent Spanish, English and Chinese. The fact that numbers 5, 6, and 7 would not be found in the current LANGUAGE_PROFILE variable that represents only Spanish and English does not matter. The following shows the PROC TABULATE syntax used to generate the summary in Figure 3.

```
proc tabulate data=mydata noseps;
  class language_profile /ORDER=DATA PRELOADFMT MLF MISSING;
  class language_binary gender;
   table (language_profile='Language' all)*(language_binary='Combination' all),
   (gender all)*(n*format=5.0 pctn='%'*format=5.1)
  /box='MLF with Response Combinations' rtspace=50;
     format language_profile langf. ;
run;
```

Figure 3 Count and Percent of Language Spoken by Gender

MLF with Response Combinations		Gender				All	
		Male		Female			
		N	%	N	%	N	%
Language	Combination						
Missing	000	1	20.0	.	.	1	10.0
	All	1	20.0	.	.	1	10.0
Spanish (..1)	Combination						
	001	1	20.0	.	.	1	10.0
	011	.	.	1	20.0	1	10.0
	101	1	20.0	.	.	1	10.0
	111	.	.	1	20.0	1	10.0
	All	2	40.0	2	40.0	4	40.0
English (.1.)	Combination						
	010	1	20.0	1	20.0	2	20.0
	011	.	.	1	20.0	1	10.0
	110	1	20.0	1	20.0	2	20.0
	111	.	.	1	20.0	1	10.0
	All	2	40.0	4	80.0	6	60.0
Chinese (1..)	Combination						
	100	.	.	1	20.0	1	10.0
	101	1	20.0	.	.	1	10.0
	110	1	20.0	1	20.0	2	20.0
	111	.	.	1	20.0	1	10.0
	All	2	40.0	3	60.0	5	50.0

Figure 3 shows not only the overall within language n and percent that is shown on the rows labeled “All,” but also uses the binary variable to show the various combinations of languages spoken among respondents endorsing each language. For example, two men indicated they spoke Spanish, one spoke Spanish only and one also spoke Chinese.

SUBSETTING OBSERVATIONS USING THE BAND FUNCTION

If we wanted to subset our observations based on some combination of responses on the three indicator variables we could specify the indicator variables in a WHERE statement. For example, here we are listing all observations with a "Yes" to speaking Chinese (E3) or Spanish (E1):

```
proc print data=mydata label;
  var language_binary language_profile e3-e1;
  where e3 = 1 or e1 = 1;
run;
```

Since we have the LANGUAGE_PROFILE variable we can also use the BAND function to accomplish the same thing.

```
proc print data=mydata label;
  var language_binary language_profile  e3-e1;
  where ❸band(language_profile , ❷ input(❶'101',binary3.));
run;
```

❶ We can type in a binary number that has "1" in the 1st and 3rd positions. A "1" would be in one or both of these positions if a respondent indicated "Yes" to speaking Chinese (E3) and/or Spanish (E1).

❷ An INPUT is used to convert our binary criteria (101) to a base-10 number (5)

❸ BAND function compares the LANGUAGE_PROFILE with the base-10 version of our criteria using a "bitwise and" operation. BAND returns a 0 if a 1 is not found in either position of the binary version of LANGUAGE_PROFILE.

DYNAMICALLY CREATING THE MULTILABEL FORMAT

I created a macro program to enable production reporting of multiple response items (see Appendix). The macro program dynamically builds a MULTILABEL format by knowing the total possible combinations of 0 and 1 that may exist given the number of variables. Combinations of 0 and 1 are equal 2 to the power of the number of variables. The macro program uses the indicator variable labels as format labels. After creating the format, the macro program creates variables _BINARY and _PROFILE and displays the data using the dynamically generated MULTILABEL format as shown in the this paper. Note that this approach has limitation to the number of indicator variables. For example, with 10 indicator variables the macro program will build a multi label format that represents 2 to the 10th power or 1024 combinations. Each time an indicator variable is added the number of combinations doubles, so I would not recommend using this approach with over 1o indicator variables.

CONCLUSION

A series of indicator variables storing responses (1=Yes, 0=No) on a multiple response item may be used to create two variables, storing respectively, the binary and base-10 representation of the responses. A MULTILABEL format may be created using, preferably, the base-10 variable values to label all values having a 1 on any indicator variable. PROC TABULATE and a MULTILABEL format may be used to produce the count and percent of observations with a "Yes" on any and/or all item response options where observations are counted toward more than one option.

RECOMMENED READING

Chapman , T. (2008). Making the Most Out of Multilabel Formats. Proceedings of the Pacific Northwest SAS Users Group Annual Conference, Seattle, WA

Fehd, R. (2004). CheckAll: a Routine to Produce A Frequency of Response Data Set from Multiple-Response Data. PharmaSUG 2004

Hinson, J. & Coughlin, M. (2012). Deciphering PROC COMPARE Codes: The Use of the bAND Function. Proceedings 2012 SAS Global Forum, Orlando, FL.

Luo, S. & Lin, X. (1997) Using SAS® Bitwise Functions to Scramble Data Fields with Key. Proceedings of the Twenty-Second Annual SAS Users Group International Conference, San Diego, CA.

Pasta, D.J. & Elkin, E. (2008). PcBFF: Binary Flags Forever (Or, Boole's Paradise) SAS Global Forum 2008

APPENDIX

```
%macro chooseany(lib,d,series,serieslabel);
    data vars (keep=&series); /*indicator variables in order to use the labels*/
      set &lib..&d; if _n_ = 1;
    run;
    %let seriescnt = %sysfunc(countw(&series)); /*counts the indicator variables*/
    data combos;
    length _binary $&seriescnt..; /*creates a character to store binary*/
    %let dec = %eval(%eval(2**&seriescnt)-1); /*number of combinations 0 and 1*/
    %do k = 0 %to &dec;
           _profile = &k; /*create base-10 representation of the binary*/
           _binary = put(&k, binary&seriescnt..); /*create the binary*/
            output;
    %end;
    run;
    data formats; /*assign labels to the base-10 representation of the binary*/
      set combos;
      if _n_ = 1 then set vars; /*get indicators from original data set*/
      length hlo start end $16. fmtname $32. label $256.;
      array es(*) &series; /*array of indicator variables*/
      hlo = 'M'; /*makes format multilabel*/
      fmtname = 'mymulti'; /*name the format*/
      if _profile = 0 then do;
           /*creates default format values*/
           start="."; end="."; label='Missing'; output;
            start ="0"; end="0"; label='No Response'; output;
      end;
      else do;
      i = 1;
      do until(i > lengthc(_binary)); /*go through each position of the binary*/
       if substr(_binary,i,1) = 1 then do; /*if a 1 is found*/
              start=strip(_profile);  /*assign the decimal to start and end*/
              end= start;
              label = vlabel(es(i));   /*format label gets indicator var label*/
              output;
          end;
          i = i + 1;
       end;
       end;
    run;
    proc format lib=work cntlin=formats; /*create the multilabel format*/
    run;
    data c_dum; /*using the data to summarize*/
       set &lib..&d;
       array es(*) &series;
       length _binary $&seriescnt..;
       _binary = strip(catt(of es(*))); /*create binary*/
       _profile=input(_binary,binary&seriescnt..); /*create base-10*/
        format _profile mymulti.; /*assign the format create above*/
    run;
    proc tabulate data=c_dum noseps format=6.0 missing;
      class _profile / PRELOADFMT MLF order=data;
       table (_profile="" all),(n pctn='%'*format=5.1)
       /box="Table 1 &serieslabel" rtspace=50 printmiss;
    run;
    proc delete data=c_dum vars combos formats; run;
%mend;
title1 "MultiLabel Format Generated Dynamically";
%chooseany(work,mydata,%str(e1 e2 e3),Languages Spoken);
```

CONTACT INFORMATION

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

S. Patrick Thornton Ph.D., Principal. Scientific Programmer
SRI International, Center for Education and Human Services
Phone: 650 859-5583
Email: patrick.thornton@sri.com