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 speak more than one language. In this case, an indicator variable (1=Yes, 0=No) is 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 speak more than one language. In this case, an indicator variable (1=Yes, 0=No) is created for each language category. The most obvious way to obtain the count and percent of “Yes” respondents is use SAS® PROC FREQ on each of the indicator variables, but this produces multiple tables. A PROC TABULATE and a MULTILABEL format may also be used to produce a more concise table (Figure 1).

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

Figure 1 A Comparison of Results from PROC FREQ and TABULATE

<table>
<thead>
<tr>
<th>Language</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanish</td>
<td>6</td>
<td>60.00</td>
<td>6</td>
<td>60.00</td>
</tr>
<tr>
<td>Yes</td>
<td>4</td>
<td>40.00</td>
<td>10</td>
<td>100.00</td>
</tr>
<tr>
<td>English</td>
<td>4</td>
<td>40.00</td>
<td>4</td>
<td>40.00</td>
</tr>
<tr>
<td>No</td>
<td>1</td>
<td>10.00</td>
<td>5</td>
<td>50.00</td>
</tr>
<tr>
<td>Yes</td>
<td>5</td>
<td>50.00</td>
<td>10</td>
<td>100.00</td>
</tr>
<tr>
<td>Chinese</td>
<td>5</td>
<td>50.00</td>
<td>10</td>
<td>100.00</td>
</tr>
<tr>
<td>No</td>
<td>1</td>
<td>10.00</td>
<td>1</td>
<td>10.00</td>
</tr>
<tr>
<td>Yes</td>
<td>5</td>
<td>50.00</td>
<td>6</td>
<td>60.00</td>
</tr>
</tbody>
</table>

The final table in Figure 1 uses a MULTILABEL format so that the total observations are correctly shown as10 while observations may count toward more than one language category. Notice that the percent in the final table match those in the PROC FREQ and sum of the counts is 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?"

```sas
proc format;
  value yesno 1 = 'Yes' 0 = 'No';
  value gender 1 = 'Male' 2 = 'Female';
run;
data mydata;
  /*establish 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*/
  language_binary = strip(catt(of e3 e2 e1));
  format gender gender. e3 e2 e1 yesno.;
cards;
0 0 1 1
0 1 2 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;
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;
```

1. A variable, LANGUAGE_BINARY, is created from the concatenation of the 3 language indicator variables in reverse order. Reverse order is used because binary variables grow from right to left.

2. 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

<table>
<thead>
<tr>
<th>Obs</th>
<th>Chinese</th>
<th>English</th>
<th>Spanish</th>
<th>Language_binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>000</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>001</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>010</td>
</tr>
<tr>
<td>4</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>010</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>011</td>
</tr>
<tr>
<td>6</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>101</td>
</tr>
<tr>
<td>8</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>110</td>
</tr>
<tr>
<td>9</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>110</td>
</tr>
<tr>
<td>10</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>111</td>
</tr>
</tbody>
</table>

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 subset and summarize observations based on responses to all 3 indicator variables.

**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.

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

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

```sas
proc print data=mydata label;
  var language_binary e3-e1;
  where band(input(language_binary,3.),'101);
run;
```

BAND performs a “bitwise and” operation which compares each position of LANGUAGE_BINARY to the corresponding position of second argument, and if both positions are 1 then it returns a 1, else a 0. The end result is that all observations are returned that have a 1 in LANGUAGE_BINARY in one or more of the same positions in the second argument. Use of a binary variable and the BAND function may make it easier to select observations that meet a pattern over a large number of indicators. For example, if we had 10 language indicator variables, and were interested in listing OBS having “yes” on 7 specific variables, then we could just change the binary selector to:

```sas
where band(input(language_binary,10.),'11011011')
```

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

We could create a MULTILABEL format using the binary numbers:

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

The MULTILABEL option allows us to assign the same binary value 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.

```sas
proc tabulate data=mydata format=10.0;
  class language_binary /ORDER=DATA PRELOADFMT MLF 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;
```

The use of 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 values 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 base-10 equivalent of the binary variable whose values do not all change when a different number of indicator variables are involved.

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

The DATA step uses the LANGUAGE_BINARY to create a variable storing the base-10 equivalent called LANGUAGE_PROFILE (Figure 3). The variable is created using the SAS supplied format BINARY3.

I created the following MULTI-LABEL 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 easily be 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 format used to represent Spanish, English and Chinese. The fact that numbers 5, 6, and 7 would not found in the LANGUAGE_PROFILE variable representing only Spanish and English does not matter. The following shows the PROC TABULATE used to generate the summary in Figure 4.

```
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;
```
CONCLUSION

A series of indicator variables storing responses on a multiple response item may be used to create a binary and base-10 variable. The use of the BAND function is a scalable strategy for sub-setting, and the creation of a MULTILABEL format for use with PROC TABULATE allows a concise presentation of results.

RECOMMENDED READING


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

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ™ indicates USA registration.

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

<table>
<thead>
<tr>
<th>Language</th>
<th>Combination</th>
<th>Gender</th>
<th>Male</th>
<th>N</th>
<th>%</th>
<th>Female</th>
<th>N</th>
<th>%</th>
<th>All</th>
<th>N</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanish</td>
<td>000</td>
<td>1</td>
<td>10.0</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>010</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>All</td>
<td>2</td>
<td>20.0</td>
<td>2</td>
<td>20.0</td>
<td>4</td>
<td>40.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>English</td>
<td>010</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>All</td>
<td>2</td>
<td>20.0</td>
<td>4</td>
<td>40.0</td>
<td>6</td>
<td>60.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chinese</td>
<td>010</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>All</td>
<td>2</td>
<td>20.0</td>
<td>3</td>
<td>30.0</td>
<td>5</td>
<td>50.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>000</td>
<td>1</td>
<td>10.0</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>001</td>
<td>1</td>
<td>10.0</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>010</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>1</td>
<td>10.0</td>
<td>1</td>
<td>10.0</td>
<td>2</td>
<td>20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>All</td>
<td>5</td>
<td>50.0</td>
<td>5</td>
<td>50.0</td>
<td>10</td>
<td>100.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>