PharmaSUG China 2015 – Paper 17

Creating Binary Tables in a Snap

Yan Liu, Sanofi R&D, Beijing, China

ABSTRCT

Summary tables for binary variables are very common in clinical trial analysis. Conventional methods usually require stacking a bunch of rows of binary variable analytical outputs to get the results desired. It is time-consuming and inefficient from the programming point of view. This paper introduces a convenient SAS macro that can create binary tables with one simple macro call and is capable of using either subject level data set (ADSL) or BDS (Basic Data Structure) ADaM data sets as inputs or a combination of these two types of data sets in case of mixed requirements.

INTRODUCTION

Binary variables are variables that only take two values, for example, Yes/No, True/False, Present/Not present etc. Figure 1 below is a simple binary variable summary table that is very common in clinical trial studies. All the variables concerned in the table are analysis population flag variables that only take Y/N values. Usually, such population flag variables are stored in the subject level data set ADSL. Figure 2 is another example of binary variable summary table. Compared with the one-record-per-subject structure in Figure 1, the data structure in Figure 2 is Multiple-record-per-subject basic data structure (BDS). Some of the variables in Figure 2 are also not obviously binary, for example, the availableness of laboratory test data. But such variables can be easily converted to a Y/N binary variable with a “if VAR1…. then BIVAR=Y/N” if statement. In Figure 2, there are also summaries for vaccinations received at different visits. Even though this is a multiple-record-per-subject information, such information usually are stored in ADSL, because it is not worthy to create a ADEX analysis data set in most cases. For such kind of tables that need to read data from both BDS and ADSL, a merge between BDS and ADSL can be performed at first to decrease the number of call macro procedures and

Figure 1: A simple binary table example

	CYD Dengue Vaccine Group	Placebo Group	All
	n (%)	n (%)	n (%)
Safety Analysis Set *	### (##.#)	### (##.#)	### (##.#)
Safety Analysis Set post-Inj 1 †	### (##.#)	### (##.#)	### (##.#)
Safety Analysis Set post-Inj 2 †	### (##.#)	### (##.#)	### (##.#)
Safety Analysis Set post-Inj 3 †	### (##.#)	### (##.#)	### (##.#)
Full Analysis Set ‡	### (##.#)	### (##.#)	### (##.#)
Per-protocol Analysis Set post-Inj 1 ‡	### (##.#)	### (##.#)	### (##.#)
Per-protocol Analysis Set post-Inj 2 ‡	### (##.#)	### (##.#)	### (##.#)
Per-protocol Analysis Set post-Inj 3 ‡	### (##.#)	### (##.#)	### (##.#)

n is the number of subjects in the specified population

* Subjects were classified as per first vaccine received

† Subjects were classified as per the vaccine they received

‡ Subjects were classified as per the vaccine to which they were randomized

Figure 2: A more complicated binary table example

Timepoint		6 to <36 Months (N=###) n (%)	3 to <9 Years (N=###) n (%)	All (N=###) n (%)
V01 (D0)	Present	### (##.#)	### (##.#)	### (##.#)
	Vaccination received	### (##.#)	### (##.#)	### (##.#)
	BL performed	### (##.#)	### (##.#)	### (##.#)
	[A (H1N1)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[A (H3N2)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[B] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
V02 (D28)	Present	### (##.#)	### (##.#)	### (##.#)
	Vaccination received (for subjects Receiving 2 Doses)	### (##.#)	### (##.#)	### (##.#)
	BL performed (for subjects Receiving 1 Dose)	### (##.#)	### (##.#)	### (##.#)
	[A (H1N1)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[A (H3N2)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[B] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
V03 (for subjects Receiving 2 Doses)	Present	### (##.#)	### (##.#)	### (##.#)
	BL performed	### (##.#)	### (##.#)	### (##.#)
	[A (H1N1)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[A (H3N2)] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)
	[B] ([HAI] - [1/dil]) available	### (##.#)	### (##.#)	### (##.#)

facilitate programming. As for this particular case, a further simple modification is also needed as the vaccination information are stored in more than one variable in ADSL, and there are subjects that are in ADSL but not in BDS.

THE MACRO

The following SAS macro is developed to create summary tables for binary variables. The macro has eleven parameters. The lib parameter specifies the library name of the input data set; dsin, the name of the input data set; dsout, the name of the output data set; pop, the population flag; trtn, the treatment group variable in the input data set; tot takes the Y/N value and indicates whether a "Total" column is needed in the output; list lists the binary variables that need to be presented in the output; text lists the corresponding labels of the binary variables listed in the list parameter; fvar lists the treatment group and the category variables of the data separated by '*'; ci takes the Y/N value and indicate if CIs need to be calculated; line lists line numbers that are blanks.

```
%macro binary(lib=work, dsin=, dsout=, pop=, trtn=, tot=N, fvar=&trtn, list=, text=,
ci=N, line=);

    proc sql noprint;
        select count(distinct &trtn) into :ngr
        from &lib..adsl;
    quit;

    %let ncol=&ngr;
```

```
%if %upcase(&tot)=Y %then %do;
    %let ncol=%eval(&ngr+1);
%end;

data &dsin;
    set &lib..&dsin.(where=(&pop));
    %if %upcase(&tot)=Y %then %do;
        output;
        &trtn=&ngr+1;
        output;
    %end;
run;

%if %upcase(&dsin)^=ADSL %then %do;
    data adsl;
        set adsd.adsl(where=(&pop));
         %if %upcase(&tot)=Y %then %do;
           output;
           &trtn=&ngr+1;
           output;
        %end;
     run;
%end;

%do a=1 %to &ncol.;
    proc sql noprint;
        select count(distinct usubjid) into :NN&a
        from adsl where &trtn=&a;
    quit;
%end;

%let bfvar=%sysfunc(tranwrd(&fvar,*, ));
%let numm=%sysfunc(countw(&bfvar));
%if &numm>1 %then %do;
    %let by=%substr(&bfvar,%sysfunc(length(%scan(&bfvar,1)))+1);
%end;
%let num=%sysfunc(countw(&list));
%do ii=1 %to &num;
    %let itm=%scan(&list, &ii);
    %let lbl=%scan(&text, &ii, |);

proc sort data=&dsin.(where=(&itm='Y')) out=_out&ii nodupkey;
    by usubjid &trtn
        %if &numm>1 %then %do; &by %end; &itm;
```

```
 run;

proc freq data=_out&ii noprint;
   table &fvar / out=out&ii;
run;

%if &numm>1 %then %do;
   proc sort data=out&ii; by &by; run;
%end;

proc transpose data=out&ii out=trnout&ii;
   id &trtn;
   %if &numm>1 %then %do; by &by;%end;
   var count;
run;

data ds&ii;
   set trnout&ii;
   labl="&lbl"; ii=&ii;
   %do n=1 %to &ncol;
      if _&n=. then _&n=0;
      nm&n=catx('/',_&n,&&NN&n);
      pct&n=compress(put(_&n*100/&&nn&n, 5.1));
      npct&n=compbl(put(_&n,best.)||" ("||compress(put(_&n*100/&&nn&n,
     5.1))||")");
      %if &ci=Y %then %do;
         %Stxt(P1cI, _&n,&&NN&n, varout=ci&n);
      %end;
   %end;
run;

    %if &ii=&num %then %do;
       data &dsout;
          length labl $100.;
          set %do jj=1 %to &num; ds&jj %end;;
        proc sort; by %if &numm>1 %then %do; &by %end;ii;
       run;

       data &dsout;
            set &dsout;
          output;
          if _n_ in (&line) then do;
             call missing(of _all_); output;
          end;
```

```
            run;
        %end;
    %end;
%mend;
```

The ***Stxt*** sub-macro is a confidence interval calculation macro of Sanofi Pasteur, which I don't have the authorization to show the contents in this paper. But it can be easily replaced by any similar CI calculation macros. CI calculation is an extended function of this binary macro. Most binary variable tables don't need to calculate CIs though and the CI parameter is set to 'N' by default. Below is a sample call macro to produce the table in Figure 2.

%***binary***(dsin=adslim, dsout=final, pop=FAS='Y', trtn=agegrpn, tot=Y, fvar=agegrpn*visitnum, list=Presn VAC LBYNA FL1 FL2 FL3, text=Present | Vaccination received|BL performed | [A (H1N1)] ([HAI] - [1/dil]) available | [A (H3N2)] ([HAI] - [1/dil]) available|[B] ([HAI] - [1/dil]) available, line=**6 12**);

CONCLUSION

In this paper, I introduced a SAS macro that can produce summary tables for binary variables. The macro is capable of creating the desired results for binary variables or variables that can be converted to binary values with a one simple call macro. For variables stored in BDS instead of ADSL or both, a merge between BDS and ADSL can be performed beforehand to avoid multiple call macro procedures. Depending on the complexity, a further data set modification can also be carried out if needed.

AUTHOR CONTACT

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

Name: Yan Liu
Enterprise: Biostatistics & Programming, Sanofi R&D China
Address: 112 Jianguo Road, Chaoyang District
City, State ZIP: Beijing, China 100022
Work Phone: +86 10 65634779
Email: chris.liu@sanofi.com

TRADEMARK INFORMATION