

# A SAS® MACRO FOR PRODUCING CLINICAL LABORATORY SHIFT TABLE

Shi-Tao Yeh, GlaxoSmithKline, King of Prussia, PA

## ABSTRACT

The results of all safety-related laboratory tests are collected in clinical trial studies. A shift table is a table that displays the number of subjects who are low, normal, or high at baseline and the shift or transition at selected time points or time intervals.

This table provides key information about the health and welfare of the subjects during the clinical study.

This paper provides a comprehensive, flexible, and modular SAS reporting approach for producing a laboratory shift table.

The SAS® product used in this paper is SAS BASE® on UNIX platform.

## INTRODUCTION

A clinical trial study collects all safety-related clinical laboratory (Lab) tests for safety analysis. SAS programmers in the clinical study team support the clinical data analysis and reporting process.

One of the more frequently requested reports is the Lab Shift Table (LST). It is a clinical report that summarizes the number of subjects who are low, normal, or high at

**Figure 1 Sample Output from lbsf1.sas Program**  
baseline and then at selected time points or time intervals. In other words, it displays Lab value transitions with respect to normal range and clinical concern level categories from a baseline period to a specific on-therapy visit period.

The vendor who performs the laboratory test usually provides normal range for numeric Lab parameters. The clinician may specify clinical concern level for the study. The study team uses a 'Lab Flagging' program to flag the Lab records to see whether it is above or below the concern level.

There are several types of LST presentation formats and layout. Figure 1 shows the LST layout that is adopted by GlaxoSmithKline as standard LST presentation.

## LAB FLAGGING

Prior to summarization, the raw data from a clinical laboratory require Lab Flagging. Lab flagging is a process that marks each laboratory record in the database using parameter specific clinical criteria. The criteria often include: 1) values are outside of the supplied normal range, 2) values are high or low enough to cause clinical concern, 3) values have increased or decreased by a specified amount to cause clinical concern. There are three flags from this flagging process: 1) normal range flag, 2) shift from baseline flag, and 3)

NUMBER AND PERCENTAGE OF RANDOMIZED PATIENTS BY TRANSITIONS IN LABORATORY TEST VALUES										
LAB Test=Alanine Aminotransferase										
Treatment=PLACEBO										
Time Period Value										
Planned Relative Time	Baseline n Value [1]	Reference Range	High	Normal Range	High	Within Norm Range	Normal Range	Reference Range	Low	Total
WEEK 3	71	RR High	0	0	0	0	0	0	0	0
		NR High	0	0	0	0	0	0	0	0
		W/in NR	0	0	71	100%	0	0	0	71 (100%)
		NR Low	0	0	0	0	0	0	0	0
		RR Low	0	0	0	0	0	0	0	0
	Total		0	0	71	(100%)	0	0	0	71 (100%)
WEEK 4	69	RR High	0	0	0	0	0	0	0	0
		NR High	0	0	0	0	0	0	0	0
		W/in NR	0	0	69	(100%)	0	0	0	69 (100%)
		NR Low	0	0	0	0	0	0	0	0
		RR Low	0	0	0	0	0	0	0	0
	Total		0	0	69	(100%)	0	0	0	69 (100%)

Subject count for subject has baseline and planned visit values

Subject count

Percentage count

clinical concern flag. Each flag may contain the symbols:

H for above the range, L for below the range, and I for within the range.

For the illustration purposes, the Lab flagging criteria and flags are converted to the following symbols and labels on the report:

Symbol	Label	Description
+	RR High	above high value of clinical concern
H	NR High	above high value of normal range
I	W/in NR	within normal range
L	NR Low	below low value of normal range
-	RR Low	below low value of clinical concern

LST is presented in a matrix format, shown in Figure 1. The first column is the visit or time interval label. The second column is n which is the total subjects for each visit or time interval. A subject must have both baseline and target visit/interval test values to be counted in the LST.

The baseline flagged values are shown in rows of the matrix and target visit or interval flagged values are shown in columns. Each cell represents a shift from a specific baseline flagged category to a specific target visit flagged category. The LST also provides the percentage for each cell. The percentage of the value in a single cell is the count of the cell divided by n.

When a user selects time interval with multiple visits, the worst case test value or the last test value carried forward can be selected for each subject.

This paper presents a novel and comprehensive approach that is simple, efficient, and produces a readily interpretable transition presentation in the LST. The approach consists of several SAS macros, and it is discussed in the following sections. The sample SAS codes are shown in Appendix I.

## STEP 1: SET INITIATION AND ENVIRONMENT VARIABLES

The first step sets up macro variables, computing environment variables, and formats used for reporting the LST. The input parameters for this macro are as follows:

PARAMETER	DESCRIPTION
dataset	the input dataset name
noprintv	no print variable which controls the order of display
byvars	a categorical variable list for page by variables that appeared in the display
byfmts	format name for byvars
pidvars	default=usubjid, subject variable name
colvar	a variable name for the first column in the display
colfmt	format name for colvar

baseflag	default=R, flag for baseline records
cellindx	default=labshift, cell index dataset name

## STEP 2: SET UP REPORTING MAIN PROGRAM

The main reporting program completes a number of relatively generic steps in the process.

- 1) assign the number of lab parameters and by group variables to loop index variables
- 2) subset valid lab flag values from the data
- 3) separate baseline and other visits
- 4) convert flagging values to the formatted symbols to be displayed on the matrix
- 5) merge the baseline records into the other visit records
- 6) count the number of subjects in each matrix cell and the total denominators
- 7) compute the cell percentages
- 8) determine which parameters do not have clinical concern flags and blank out the clinical concern rows for these pages

## STEP 3: CREATE A REPORTING MODULE

The reporting module produces a matrix page of output for each laboratory parameter and by group requested.

## STEP 4: CREATE A CELL INDEX MODULE

The final step produces an optional cell index to provide the detailed laboratory data that comprise each cell.

## STEP 5: CREATE SASMAN PROGRAM

The SASMAN program is a protocol-specific module that allows the other modules of the LST program to remain relatively generic. It is also the data preparation step before invoking the LST macro. The SASMAN module increases the flexibility of the LST software by performing the following tasks:

- 1) read in the flagged Lab dataset,
- 2) make any required data subset selection,
- 3) select the target Lab variable list,
- 4) define the column variables and baseline variables in the LST and feed-in dataset,
- 5) select the last value or worst case based on protocol-specific requirements,
- 6) use data steps to prepare dataset to be feed in to main program.

## INVOCATION SAMPLE CODE

The following sample code is an example of SASMAN program before calling the LST macro.

```
proc format;
  value labses
    2 = 'BASELINE'
    8 = 'WEEK 8'
    9 = 'FOLLOW-UP'
  ;

data lab;
  set _adata.lab;
  lbstxrhi = lborxrhi;
  lbstxrlo = lborxrlo;

  lab_parm = trim(left(lbtest)) || '(' ||
trim(left(lbstunit)) || ')';
  if visitnum = 1 and lbflag='B' then
visitnum=2;
  if visitnum=15 then visitnum=9;
  if visitnum=1 or (visitnum = 2 and lbflag
= 'U' ) or trtcd=1 then delete;
  if visitnum in (2,8,9);
  keep lbnrcd lbxrcd lborres lbstresn
lbflag usubjid subjid lbtest lbtestcd
lbstunit lbstxrhi lbstxrlo visit visitnum
trtcd trtgrp lab_parm;

data dem;
  set _adata.demo;
  if trtcd = 1 then delete;
  keep usubjid;
  proc sort;by usubjid;

  data final;
  set lab;
  vorder = visitnum;
  label lab_parm = 'Test'
  trtgrp = 'Treatment';

proc sort;by usubjid;

data final;
  merge dem(in=a) final;
  by usubjid;
  if a;
  %rstart(driver=titles.lis,
rptdir=$_list, suffix=a, outfile=lb_4);
  options ls=90 ps=31 missing=' ' nobyline;
  %inc 'lbsf1.sas';

  %lbsf1(dataset=final,
  noprintv=vorder,
  byvars = lab_parm trtgrp,
  colvar = visitnum,
  baseflag = B,
  cellindx = labshift,
  colfmt = labses
  );

%rstop;
  run;

  proc datasets;
  delete final lab;
```

run;

## SAMPLE OUTPUTS

Two sample pages of output are presented. The first page, shown in Figure 1, represents a laboratory parameter for which normal range and clinical concern criteria were specified. The second sample page in Figure 2, represents a laboratory parameter for which only clinical concern criteria were specified.

Planned Relation Time	Baseline n Value(%)	Time Period Values						Total
		Reference Range	Normal High	Within Norm Range	Normal Range Low	Reference Range Low		
WEEK 2	68	BP High	1	0	1	0	0	2
		BP Low	0	0	0	0	0	0
		RR High	2	0	65	0	0	67
		RR Low	0	0	0	0	0	0
		RR Low	0	0	0	0	0	0
		Total	3	0	65	0	0	68
WEEK 8	64	BP High	0	0	1	0	0	1
		BP Low	2	0	63	0	0	65
		RR High	0	0	0	0	0	0
		RR Low	0	0	0	0	0	0
		RR Low	0	0	0	0	0	0
		Total	2	0	64	0	0	66

Figure 2. Sample Output

## CONCLUSIONS

This LST approach provides a framework that is simple, comprehensive, efficient, and produces a readily interpretable transition presentation in the LST. This modular design makes the best use of the SAS system capability of providing robust generic code, and customizable data processing components.

In summary, this approach

- \* Provides a framework for the rapid development and modification of a LST program.
- \* Provides a compact and easy-to-read matrix format that allows one page per laboratory parameter.
- \* Allows lab parameters with and without missing values and clinical concern levels to be presented in the same set of pages.

## APPENDIX I: MACRO CODE

```
%macro lbsf1 (
  dataset=,
  noprintv=,
  byvars=,
  byfmts=,
  pidvars=usubjid,
  colvar=,
  colfmt=,
  baseflag=R,
  cellindx=labshift
);
proc format ;
```

```

value labrow 1 = 'RR High'
              2 = 'NR High'
              3 = 'W/in NR'
              4 = 'NR Low '
              5 = 'RR Low '
              6 = 'Total  ';

picture pp (round)
          0 = ' ' (noedit)
          other = '0009%)' (prefix='(');

%* Separate all by variable information ;
%let numbys=1;
%do %while (%scan(&byvars, &numbys) ne )
;
  %let byvar&numbys = %scan(&byvars,
&numbys) ;
  %let byfmt&numbys = %scan(&byfmts,
&numbys,#) ;
  %if &&byfmt&numbys = ! %then %let
byfmt&numbys =;
  %let numbys = %eval(&numbys + 1) ;
%end ;
%let numbys = %eval(&numbys - 1) ;

%* Sort the input dataset ;
%* Extract only flag values that have
values we can plot in the ;
%* shift table ;
proc sort data=&dataset
          out=table ;
  by &pidvars &byvars &colvar ;
  where
    lbncrd in ('H', 'L', 'I');
run ;
%proc print data=table;run;
%* Separate baseline and other visits ;
data baseline (keep=&pidvars &byvars
lbncrd lbxrcd lborres lbstresn lbflag
              rename=(lbncrd= bf1
lbxrcd = bf3 lbstresn=base_val))
  other (keep=&byvars &pidvars
&colvar lbncrd lbxrcd lbstresn lborres ) ;
  set table ;
  if lbflag = "&baseflag"
  then output baseline ;
  else if lbflag ^= "&baseflag" then
output other ;
run ;

proc datasets nolist nofs ;
  delete table ;
run ;

%* Get latest assessment with flags within
each visit for each pid ;
%* Set the row and column co-ordinate ;
data baseline ;
  set baseline ;
  by &pidvars &byvars ;

%if %length(&byvars) gt 0 %then %do ;
  if last.&&byvar&numbys ;
%end ;

if bf3 = 'H' then _row = 1 ;
else if bf3 = 'L' then _row = 5 ;

```

```

else if bf1 = 'H' then _row = 2 ;
else if bf1 = 'L' then _row = 4 ;
else _row = 3 ;

run ;
%* Only take the latest information within
the column variable ;
data other ;
  set other ;
  by &pidvars &byvars &colvar ;
  if last.&colvar ;

  if lbxrcd = 'H' then _col = 1 ;
  else if lbxrcd = 'L' then _col = 5 ;
  else if lbncrd = 'H' then _col = 2 ;
  else if lbncrd = 'L' then _col = 4 ;
  else _col = 3 ;
run ;

%* Merge the baseline into the other visit
;
data tablek ;
  merge other(in=a) baseline(in=b) ;
  by &pidvars &byvars ;
  if a and b ;
run ;
%proc print data=tablek;run;
data tablek;
  set tablek;
  output;
  _row = 6;
  output;
proc sort data=tablek;by &byvars &colvar
_row _col;
proc datasets nolist nofs ;
  delete other baseline ;
quit ;

%* Count the number of patients ;
proc summary data=tablek nway missing ;
  class &byvars &colvar _row _col ;

output out=table(rename=_freq_=num) ;
run ;

%* Count again for totals ;

proc summary data=table nway missing ;
  class &byvars &colvar _row ;
  var num ;
  output out=tablet sum= ;
run ;
proc summary data=table nway missing ;
  class &byvars &colvar;
  var num ;
  output out=allpid sum=allpid ;
run ;
%proc print data=allpid;run;
data allpid;
  set allpid;
  %do i=1 %to 6;
  _row = &i;
  output;
%end;
%proc print ;run;
proc sort;
  by &byvars &colvar _row ;

```

```

data table ;
  set table(in=a) tablet(in=b) ;
  by &byvars &colvar _row ;
  if b then _col = 6 ;
run ;
*proc print data=table;run;
%* Calculate the percentages now ;
data table ;
  merge table(in=a) tablet(in=b
rename=(num=tnum)) allpid;
  by &byvars &colvar _row ;
  output ;
  _row = _row + 6 ;
  if tnum ne . then num = 200 * num /
allpid;
  output ;
run ;

proc datasets nolist nofs ;
  delete tablet ;
run ;

%* Generate a template ;
proc sort data=table (keep=&byvars
&colvar) out=tmp11 nodupkey ;
  by &byvars &colvar ;
run ;

data template ;
  set tmp11 ;
  by &byvars &colvar ;
  do _row = 1 to 12 ;
    do _col = 1 to 6 ;
      output ;
    end ;
  end ;
run ;

proc datasets nolist nofs ;
  delete tmp11 ;
quit ;

%* Merge template into counts ;
proc sort data=table ;
  by &byvars &colvar _row _col ;
run ;

data table ;
  merge table template ;
  by &byvars &colvar _row _col ;
  if num = . then num = 0 ;
  * if _row = 7 or _col = 8 then num = .
;
  retain pg cnt ;
  if _n_ = 1 then cnt = 0 ;
  %if %length(&byvars) gt 0 %then %do ;
    if first.&&byvar&numbys then cnt = 0
;
  %end ;
  if first.&colvar then cnt + 1 ;
  if cnt = 4 then do ;
    pg + 1 ;
    cnt = 1 ;
  end ;
  %do i = 1 %to &numbys ;

```

```

%if %length(&&byfmt&i) gt 0 %then
%do ;
  format &&byvar&i &&byfmt&i... ;
  %end ;
%end ;
%if %length(&colfmt) gt 0 %then %do ;
  format &colvar &colfmt.. ;
  %end ;
run ;

%let byvars=&byvars ;

proc datasets nolist nofs ;
  delete template ;
quit ;

%* Row format for table ;
run ;

%* Transpose data to get one line per row
shift ;
proc sort data=table ;
  by &byvars _row &colvar _col ;
run ;
*proc print data=table;run;
proc transpose data=table out=table
prefix=collet ;
  by &byvars _row &colvar ;
  var num ;
  id _col ;
run ;

data f3;
  set &dataset;
  if lbxrcd = 'I';
  keep %scan(&byvars , 1);
  proc sort data=f3 nodupkey;by %scan(
&byvars ,1);
  *proc print data=f3;run;
  *proc print data=table;run;
  proc sort data=table; by %scan( &byvars ,
1);

data tn tp;;
  merge table(in=a) f3(in=b);
  by %scan( &byvars , 1) ;
  if a;
  if not b and _row in (1,5,7,11) then do;
    %do i=1 %to 6;
      col&i=.;
    %end;
  end;
  if not b and _row in (2,3,4,6,8,9,10,12)
then do;
    col1=.;
    col5=.;
  end;
  if _row in (1,2,3,4,5,6) then output
tn;
  else if _row in (7,8,9,10,11,12) then
output tp;

data tall;
  set tn;
  tall = col6;
  if _row=6 ;
  keep &byvars &colvar tall;

```

```

*proc print data=tn;run;

proc sort data=tall;by &byvars &colvar;

data tp;
  set tp;
  _row = _row - 6;
  %do i=1 %to 6;
  col&i.p=col&i;
  %end;
drop col1 col2 col3 col4 col5 col6;

proc sort data=tn;by &byvars _row
&colvar;
proc sort data=tp;by &byvars _row
&colvar;

*proc print data=tn;run;
*proc print data=tp;run;
data table;
merge tn tp;
by &byvars _row &colvar;
acvar=' ';
%if %length(&colfmt) eq 0 %then %do;
  time1=&colvar;
%end;
%else %do;
  time1=put(&colvar , $&colfmt.);
%end;
proc sort data=table;by &byvars &colvar;
*proc print data=table;run;
*proc print data=tall;run;

data table;
merge table tall;
by &byvars &colvar;
run;
%if %length(&noprntv) gt 0 %then %do;
data noprint;
set &dataset;
keep &byvars &colvar &noprntv;
proc sort data=noprnt nodupkey;by
&byvars &colvar &noprntv;
data table;
merge table(in=a) noprint;
by &byvars &colvar;
if a;
run;
%end;

%* Create the table ;
proc report data=table nowd headline
missing split='^' spacing=1;
  %if %length(&byvars) gt 0 %then %do ;
  by &byvars ;
  %end ;
  columns ( &noprntv time1 tall _row
acvar, ( '--' ('Reference^Range High'
(col1 col1p))

('Normal^Range High' (col2 col2p))

('Within^Norm Range' (col3 col3p))

('Normal^Range Low' (col4 col4p))

('Reference^Range Low' (col5 col5p))

```

```

(' Total
' (col6 col6p)))
);
%if %length(&noprntv) gt 0 %then %do;
define &noprntv / order noprint;
%end;
define time1 /'Planned^Relative^Time
' width=10 order flow ;
define tall /'n' width=4 order
spacing=1 ;
define _row / group
'Baseline^Value[1]' order=data width=8
left format=labrow. ;
define acvar / across order=data
'Time Period Value';
%do i=1 %to 6;
define col&i / sum ' ' f=3. width=3 ;
define col&i.p / sum ' ' f=pp.
width=6;
%end;
break after tall/skip;
run ;

proc datasets nolist nofs ;
delete table tp tn f3 &dataset;
quit ;

%* Create dataset for the cellindex ;
%* NOTE: The cell index is not generated
here. ;
data &cellindx ;
set tablek ;
baseline =
substr(put(_row,labrow.),1,1) ;
interval =
substr(put(_col,labrow.),1,1) ;
keep &byvars &pidvars &colvar baseline
interval ;
run ;
proc datasets nolist nofs ;
delete tablek ;
quit ;
%mend lbsf1;

```

## ACKNOWLEDGMENTS

The author would like to thank Jitendra Shah for his earlier work on this macro and Randall Austin from GSK for his suggestion of this reporting format.

## TRADEMARKS

SAS is a registered trademark of SAS Institute Inc., in the USA and other countries.

® indicates USA registration.

## AUTHOR CONTACT INFORMATION

Shi-Tao Yeh, Ph. D.  
(610)787-3856(W)  
E-mail: shi-tao\_yeh-1@gsk.com