PharmaSUG China 2016 – Paper 40

An Effective Way to Produce Laboratory Shift Table

Amanda Yi, Roche, Shanghai, China

Abstract

'Shift table' is often required for most clinical trials, especially laboratory data analysis. The purpose of shift table is to deduce how the results are varying from the baseline to post-baseline visits in the study. This paper will start from an example and explain the anatomy, then a step-wise explanation of how to create a shift table in a wink.

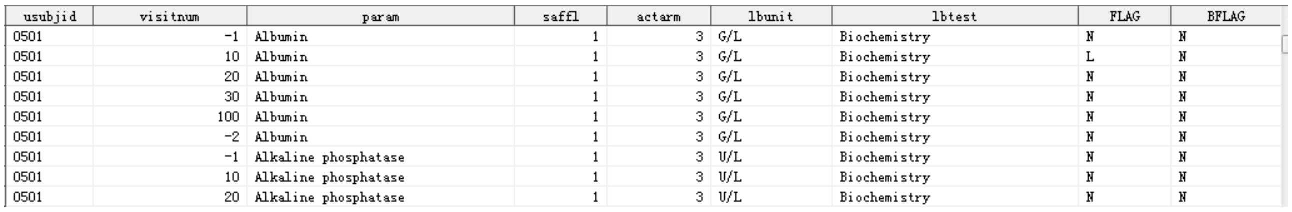
Introduction

It is important to look at the effect of the drug by comparing values at baseline and post –baseline in any clinical trials. Shift table is one of the analysis approaches suggested in E3 Guidelines to show the number of patients who are low, normal, or high at baseline and then at selected time intervals. It's easier to figure out any abnormality in data because the layout helps to get the count of subjects at two time points together. Shift tables can be created for any data like labs, vital signs and ECG…etc., as it is very common to compare results from labs, this paper will focus on how to create such a laboratory shift table in a step-wise explanation.

Data Used for Produce Shift Table

The following dataset (Figure 1) will be used for the creation of shift table and it will help understand the whole process of programming later.

Figure 1



usubjid	visitnum	param	saffl	actarm	lbunit	lbtest	FLAG	BFLAG
0501	-1	Albumin	1	3	G/L	Biochemistry	N	N
0501	10	Albumin	1	3	G/L	Biochemistry	L	N
0501	20	Albumin	1	3	G/L	Biochemistry	N	N
0501	30	Albumin	1	3	G/L	Biochemistry	N	N
0501	100	Albumin	1	3	G/L	Biochemistry	N	N
0501	-2	Albumin	1	3	G/L	Biochemistry	N	N
0501	-1	Alkaline phosphatase	1	3	U/L	Biochemistry	N	N
0501	10	Alkaline phosphatase	1	3	U/L	Biochemistry	N	N
0501	20	Alkaline phosphatase	1	3	U/L	Biochemistry	N	N

The variables used for producing the shift table is explained below,

USUBJID – Unique Subject ID
ACTARM – Actual Treatment Arm Code
SAFFL - Safety Population Flag
LBTEST – Laboratory Category (Hematology, Biochemistry…etc.)
PARAM – Laboratory Sub-Category
LBUNIT – Laboratory Standard Unit
VISITNUM – Visit Number
BFLAG – Baseline Flag of Results Out of Normal Range
FLAG – Pose-baseline Flag of Results Out of Normal Range

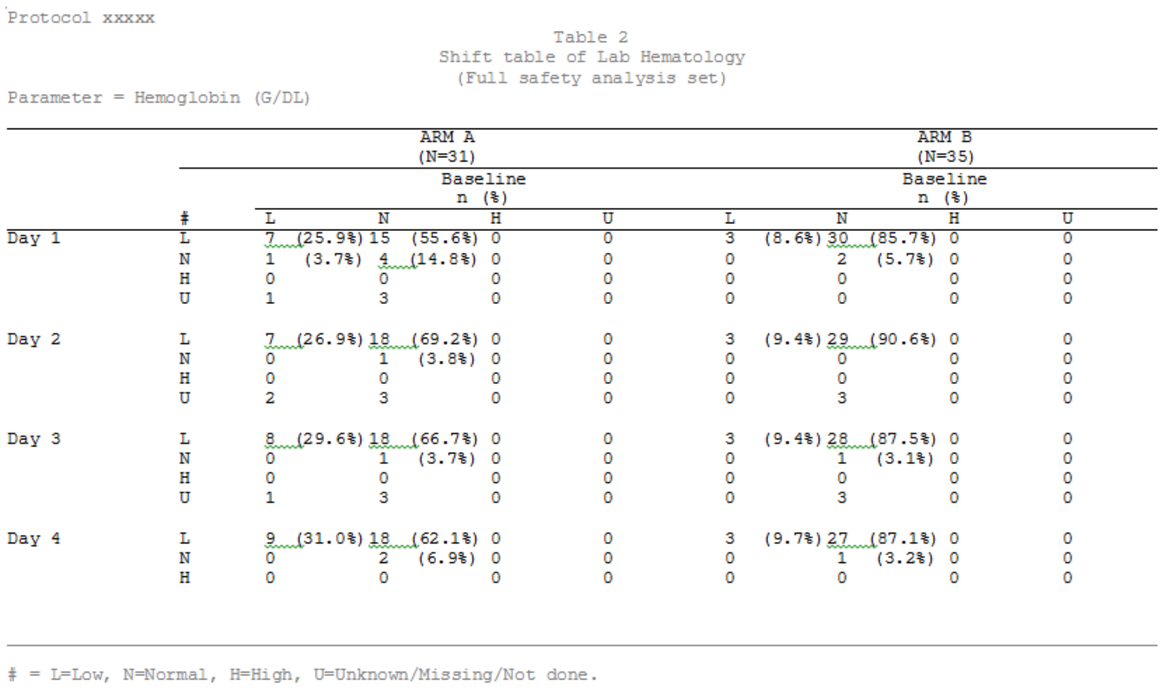
The first two variables usually come from subject level dataset like ADSL in ADaM and all the others usually come from laboratory results dataset like ADLB in ADaM.

Understanding the process

The scientist may want to see the number of subjects whose laboratory values worsened during therapy. From shift table, we can easily identify the patients who are ‘NORMAL’ at baseline and ‘ABNORMAL’ at post-baseline. And also the scientist cares about how many patients are ‘ABNORMAL’ at baseline and after treatment the laboratory values return ‘NORMAL’, which can be a very good indicator about the efficacy of the drug.

Once we are acquainted with the different values, a variable can have, in this case “Hemoglobin” can have ‘Low’, ‘Normal’, we can design the layout for the table. The basic layout can have the Treatment Groups as Columns and the post-baseline visits as rows. The figure 2 shows the details how the shift table looks like.

Figure 2



```
Protocol xxxxx
                                        Table 2
                              Shift table of Lab Hematology
                               (Full safety analysis set)
Parameter = Hemoglobin (G/DL)

                                   ARM A                                            ARM B
                                  (N=31)                                           (N=35)
                                   Baseline                                         Baseline
                                    n (%)                                            n (%)
              #      L         N         H         U         L          N         H         U
Day 1         L      7  (25.9%) 15  (55.6%) 0      0         3   (8.6%) 30  (85.7%) 0       0
              N      1   (3.7%)  4  (14.8%) 0      0         0           2   (5.7%) 0       0
              H      0           0          0      0         0           0          0       0
              U      1           3          0      0         0           0          0       0

Day 2         L      7  (26.9%) 18  (69.2%) 0      0         3   (9.4%) 29  (90.6%) 0       0
              N      0           1   (3.8%) 0      0         0           0          0       0
              H      0           0          0      0         0           0          0       0
              U      2           3          0      0         0           3          0       0

Day 3         L      8  (29.6%) 18  (66.7%) 0      0         3   (9.4%) 28  (87.5%) 0       0
              N      0           1   (3.7%) 0      0         0           1   (3.1%) 0       0
              H      0           0          0      0         0           0          0       0
              U      1           3          0      0         0           3          0       0

Day 4         L      9  (31.0%) 18  (62.1%) 0      0         3   (9.7%) 27  (87.1%) 0       0
              N      0           2   (6.9%) 0      0         0           1   (3.2%) 0       0
              H      0           0          0      0         0           0          0       0

# = L=Low, N=Normal, H=High, U=Unknown/Missing/Not done.
```

Steps to Create a Shift Table

Creation of a shift table involves realigning data, performing means to get the totals and output the results using “PROC REPORT”. Since the variable name is same in both the datasets (Baseline, Post-baseline), first step is aligning the data in such a way that each subject has base value and post-baseline value. Basically if post-baseline values are “Low” ,”High” and “Normal” then we will have three columns “Low” ,” High” and “Normal”. Now we have data aligned in such a way that we can get a column wise sum for each treatment group, baseline value and the different post-baseline values. We are basically looking for a method to facilitate the counting of subjects in the following pattern:

Table 1

Baseline Value	**Post-baseline Value**
Normal	Low
Normal	High
Normal	Normal
Normal	Missing/Unknown
Low	Low
…	…
High	Low
…	…

So basically we are creating a frequency of baseline value to different post baseline values .Now that we have looked at the process, let's try to analyze how to create a shift table in SAS .There is just one way of creating the shift table but there can be numerous ways to program.

Let's look at the SAS Steps now. The following steps are involved in creating above Shift table.

Step 1: Rearranging Data

```
*------------------------------------------------------------*;
*---bring analysis data in, remember to only keep the--------*;
*---variables required for the analysis---------------------*;
*------------------------------------------------------------*;
***baseline lab data***;
proc sort data=ana.alb (keep=usubjid actarm lbtest saffl param lbunit bflag
                               where=(lbtest="HEMO" and saffl=1 and actarm in (1,2)))
           out=base1(drop=lbtest saffl) nodupkey;
    by usubjid param lbunit actarm bflag ;
run;


**post-baseline lab data***;
proc sort data= ana.alb (keep=usubjid lbtest param visitnum flag actarm lbunit
                                where=(lbtest="HEMO" and visitnum gt 0 and actarm in (1,2)))
           out=post1(drop=lbtest actarm);
    by usubjid param visitnum;
run;
```

Subset the lab dataset for Hemoglobin Lab parameters for safety subjects where safety population flag is equal to 1 and output the baseline records to base1 dataset as baseline lab data and post-baseline records to post1 as post-baseline lab data. We only keep the necessary variables in the base1 and post1 dataset to be simple and clean to check the data in the later process.

Step 2: Ensure all subjects have all present visits populated

```
***unique subjects**;
proc sort data=post1(keep=usubjid) out=sub nodupkey;
    by usubjid;
run;

***unique labtests/visit**;
proc sort data=post1(keep=param lbunit  visitnum) out=labtest nodupkey;
    by param lbunit  visitnum;
run;

**Many to many merge, create 1 obs per subject per labtest per visit***;
proc sql noprint;
    create table allvis as
    select a.usubjid, b.*
    from   sub as a
    cross join labtest as b
    order by usubjid, param, visitnum
;
quit;
```

In real data, we probably get missing value in some visits. The above step is to ensure all subjects have all the present visits populated. If there is any missing result, we will deal with it in later step.

Step 3: Flag each missing results as 'Unknown' and get frequency count

```
***add all subjects to data and flag unknown results***;
data alllab;
    merge allvis  (in=a)
          post1   (in=b);
    by usubjid param visitnum;
   if a and not b then flag="U";
run;
```

```
***merge on baseline results***;
data incbase;
    merge base1 (in=a)
          alllab;
    by usubjid param ;
    if not a then put "ER" "ROR: Baseline not present " usubjid=;
run;
```

```
proc freq data=incbase noprint;
    table actarm*param*lbunit*visitnum*flag*bflag  / out=catfreq(drop=percent);

    table actarm*param*lbunit*visitnum*bflag       / out=cattot2(drop=percent);

    table actarm*param*lbunit*visitnum*flag        / out=cattot (drop=percent);
run;
```

In this step, we flag the missing value as ‘Unknown’ and check the data issue of missing baseline result. Then let’s use ‘PROC FREQ’ to count the number of subjects in the category in Table 1 mentioned above.

Step 4: Transpose the data to get the shift table structure.

```
*------------------------------------------------------------*;
*--------Transpose data for output--------------------------*;
*------------------------------------------------------------*;
***flip the data so that we have post baseline on each row **;
***and baseline for each column                            **;
proc transpose data=catfreq2 out=transed(drop=_name_ _label_) prefix=_;
    by actarm param lbunit visitnum flag;
    var count;
    id bflag;
run;
```

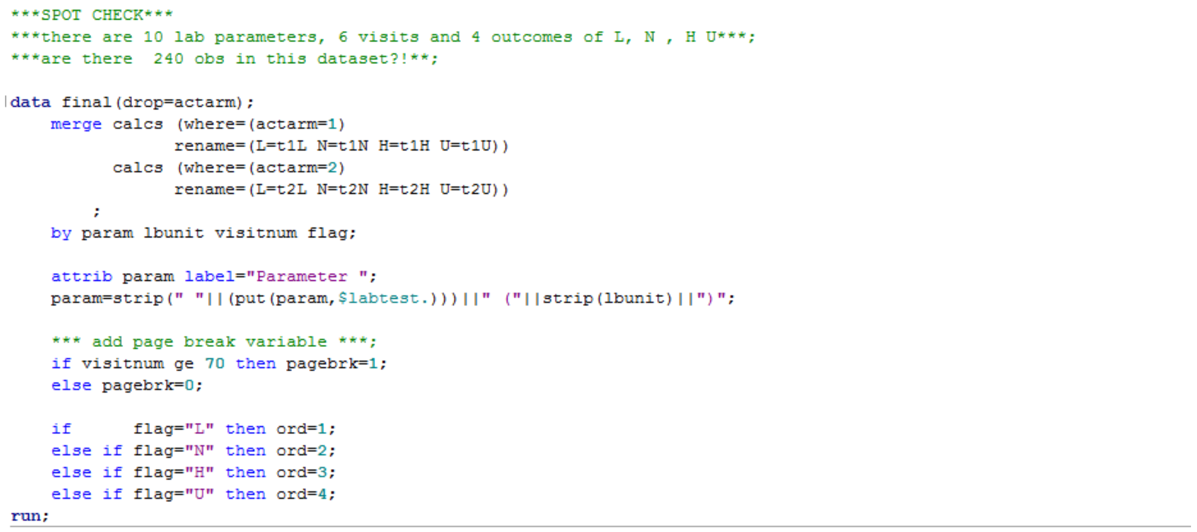
In this step, we use ‘PROC TRANSPOSE’ to get the shift table structure. Dummy dataset is used for fill up missing categories, details can be found in appendix sample code in the end of this paper.

Step 5: Calculate Percentages

```
***now calculate the percentages***;
data calcs (drop=total _U);
    set precalc;

    %macro calcit(_col=);
        if flag ne "U" then do;
            if _&_col gt .z then &_col = put(_&_col,2.)||put((_&_col/total)*100,pctfmt.);
            else &_col = put(0,2.);
        end;
        else do;
            if _&_col gt .z then &_col = put(_&_col,2.);
            else &_col = put(0,2.);
        end;

        drop _&_col;
    %mend calcit;
    %calcit(_col=L);
    %calcit(_col=N);
    %calcit(_col=H);

    if _U gt .z then U = put(_U,2.);
    else U = put(0,2.);

run;
```

In this step, let's use a simple macro to calculate the percentages of each category. We need total N as denominator for percentages, the way to calculate total N is similar to step 4, and details can be found in appendix sample code in the end of this paper.

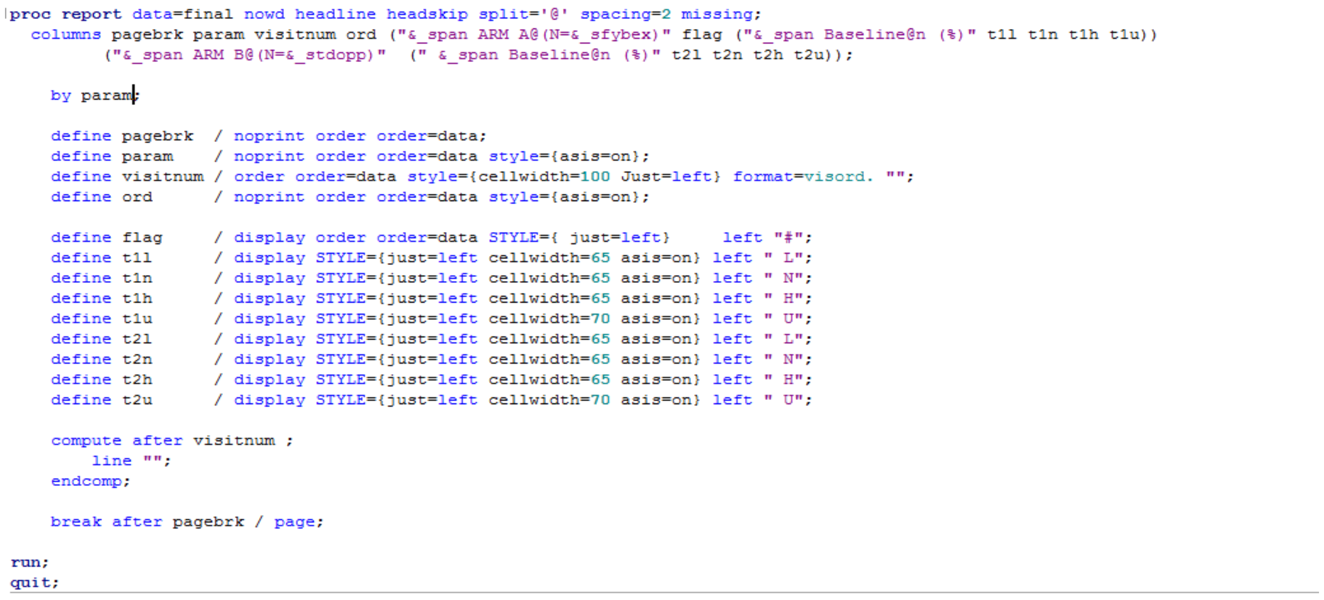
Step 6: Spot check before output



```
***SPOT CHECK***
***there are 10 lab parameters, 6 visits and 4 outcomes of L, N , H U***;
***are there  240 obs in this dataset?!**;

data final(drop=actarm);
    merge calcs (where=(actarm=1)
                rename=(L=t1L N=t1N H=t1H U=t1U))
          calcs (where=(actarm=2)
                rename=(L=t2L N=t2N H=t2H U=t2U))
        ;
    by param lbunit visitnum flag;

    attrib param label="Parameter ";
    param=strip(" "||(put(param,$labtest.)))||"  ("||strip(lbunit)||")");

    *** add page break variable ***;
    if visitnum ge 70 then pagebrk=1;
    else pagebrk=0;

    if      flag="L" then ord=1;
    else if flag="N" then ord=2;
    else if flag="H" then ord=3;
    else if flag="U" then ord=4;
run;
```

Before output to the final table, let's check whether we programs right in the above steps. The total records should be the product of the number of parameters, the number of visits and the categories of outcome to be shown in final table, which is 4(L,N,H,U) in our sample case.

Step 7: Produce final report with 'PROC REPORT'



```
proc report data=final nowd headline headskip split='@' spacing=2 missing;
  columns pagebrk param visitnum ord ("&_span ARM A@(N=&_sfybex)" flag ("&_span Baseline@n (%)" t1l t1n t1h t1u))
         ("&_span ARM B@(N=&_stdopp)"  (" &_span Baseline@n (%)" t2l t2n t2h t2u));

    by param;

    define pagebrk  / noprint order order=data;
    define param    / noprint order order=data style={asis=on};
    define visitnum / order order=data style={cellwidth=100 Just=left} format=visord. "";
    define ord      / noprint order order=data style={asis=on};

    define flag     / display order order=data STYLE={ just=left}     left "#";
    define t1l      / display STYLE={just=left cellwidth=65 asis=on} left " L";
    define t1n      / display STYLE={just=left cellwidth=65 asis=on} left " N";
    define t1h      / display STYLE={just=left cellwidth=65 asis=on} left " H";
    define t1u      / display STYLE={just=left cellwidth=70 asis=on} left " U";
    define t2l      / display STYLE={just=left cellwidth=65 asis=on} left " L";
    define t2n      / display STYLE={just=left cellwidth=65 asis=on} left " N";
    define t2h      / display STYLE={just=left cellwidth=65 asis=on} left " H";
    define t2u      / display STYLE={just=left cellwidth=70 asis=on} left " U";

    compute after visitnum ;
        line "";
    endcomp;

    break after pagebrk / page;

run;
quit;
```

Now we have our final dataset ready we can use a "PROC REPORT" to output the final shift table. The output is now produced and it looks like the one we mentioned in the beginning Figure 2.

Conclusion

Shift tables can be very effective in looking at changes from one point to another. We looked into logic of developing the shift tables and then the actual SAS code to accomplish the same.

References

SAS V9 Online Documentation

ICH E3

Contact Information

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

Name: Amanda (Chun) Yi
Enterprise: Roche
City, State ZIP: Shangai, China
E-mail: chun.yi@roche.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.

Appendix 1 – Sample Code

```
libname ana 'C:\shifttable';
libname fmt 'C:\shifttable\formats' access=readonly;

*-----------------------------------------------------------*;
*---bring analysis data in, remember to only keep the--------*;
*---variables required for the analysis---------------------*;
*-----------------------------------------------------------*;
***baseline lab data***;
proc sort data=ana.alb (keep=usubjid actarm lbtest saffl param lbunit bflag
                               where=(lbtest="HEMO" and saffl=1 and actarm in (1,2)))
           out=base1(drop=lbtest saffl) nodupkey;
    by usubjid param lbunit actarm bflag ;
run;


**post-baseline lab data***;
proc sort data= ana.alb (keep=usubjid lbtest param visitnum flag actarm lbunit
                                where=(lbtest="HEMO" and visitnum gt 0 and actarm in
(1,2)))
           out=post1(drop=lbtest actarm);
    by usubjid param visitnum;
run;


*-----------------------------------------------------------*;
*---Ensure all subjects have all present visits populated---*;
*---flag each missing result with a "U"---------------------*;
*-----------------------------------------------------------*;
***unique subjects**;
proc sort data=post1(keep=usubjid) out=sub nodupkey;
    by usubjid;
run;

***unique labtests/visit**;
proc sort data=post1(keep=param lbunit  visitnum) out=labtest nodupkey;
    by param lbunit  visitnum;
run;

**Many to many merge, create 1 obs per subject per labtest per visit***;
proc sql noprint;
```

```
    create table allvis as
    select a.usubjid, b.*
    from   sub as a
    cross join labtest as b
    order by usubjid, param, visitnum
;
quit;


***add all subjects to data and flag unknown results***;
data alllab;
    merge allvis (in=a)
          post1  (in=b);
    by usubjid param visitnum;
   if a and not b then flag="U";
run;

***merge on baseline results***;
data incbase;
    merge base1 (in=a)
          alllab;
    by usubjid param ;
    if not a then put "ER" "ROR: Baseline not present " usubjid=;
run;


*-------------------------------------------------------------*;
*---Shift table using proc freq/transpose-------------------*;
*-------------------------------------------------------------*;
proc freq data=incbase noprint;
    table actarm*param*lbunit*visitnum*flag*bflag  / out=catfreq(drop=percent);

    table actarm*param*lbunit*visitnum*bflag       / out=cattot2(drop=percent);

    table actarm*param*lbunit*visitnum*flag        / out=cattot (drop=percent);
run;

***create a dummy dataset before merge***;
data alloutcom;
    set labtest;
    do actarm = 1 to 2;
        do bflag =  "L", "N", "H", "U";
            do flag = "L", "N", "H", "U";
                output;
            end;
        end;
    end;
run;

proc sort data=alloutcom;
    by actarm param lbunit visitnum flag bflag;
run;

***merge on actual counts onto all possible outcomes***;
data catfreq2;
    merge catfreq
          alloutcom;
    by actarm param lbunit visitnum flag bflag;
run;


*-------------------------------------------------------------*;
*--------Transpose data for output--------------------------*;
```

```
*------------------------------------------------------------*;
***flip the data so that we have post baseline on each row **;
***and baseline for each column                            **;
proc transpose data=catfreq2 out=transed(drop=_name_ _label_) prefix=_;
    by actarm param lbunit visitnum flag;
    var count;
    id bflag;
run;

*------------------------------------------------------------*;
*---Calculate percentages out of n---------------------------*;
*------------------------------------------------------------*;

proc freq data=incbase(where=(flag ne "U" and bflag ne "U")) noprint;
    table actarm*param*lbunit*visitnum  / out=totfreq(drop=percent rename=(count=total));
run;


***merge in n***;
data precalc;
    merge transed
          totfreq;
    by actarm param lbunit visitnum;
run;


***now calculate the percentages***;
data calcs (drop=total _U);
    set precalc;

    %macro calcit(_col=);
        if flag ne "U" then do;
            if _&_col gt .z then &_col = put(_&_col,2.)||put((_&_col/total)*100,pctfmt.);
            else &_col = put(0,2.);
        end;
        else do;
            if _&_col gt .z then &_col = put(_&_col,2.);
            else &_col = put(0,2.);
        end;

        drop _&_col;
    %mend calcit;
    %calcit(_col=L);
    %calcit(_col=N);
    %calcit(_col=H);

    if _U gt .z then U = put(_U,2.);
    else U = put(0,2.);

run;


***SPOT CHECK***
***there are 10 lab parameters, 6 visits and 4 outcomes of L, N , H U***;
***are there  240 obs in this dataset?!**;

data final(drop=actarm);
    merge calcs (where=(actarm=1)
                rename=(L=t1L N=t1N H=t1H U=t1U))
          calcs (where=(actarm=2)
                rename=(L=t2L N=t2N H=t2H U=t2U))
        ;
    by param lbunit visitnum flag;
    attrib param label="Parameter ";
    param=strip(" "||(put(param,$labtest.))||" ("||strip(lbunit)||")");

    *** add page break variable ***;
    if visitnum ge 70 then pagebrk=1;
    else pagebrk=0;

    if      flag="L" then ord=1;
    else if flag="N" then ord=2;
    else if flag="H" then ord=3;
    else if flag="U" then ord=4;
run;

proc sort data=final;
    by param param visitnum ord;
run;


*-----------------------------------------------------------*;
*-------Output the results----------------------------------*;
*-----------------------------------------------------------*;

title2 j=c "Table 2";
title3 j=c "Shift table of Lab Hematology";
title4 j=c &_saffl.;
title5 j=l "Parameter = #byval2";
footnote2 "# = L=Low, N=Normal, H=High, U=Unknown/Missing/Not done.";

proc report data=final nowd headline headskip split='@' spacing=2 missing;
  columns pagebrk param visitnum ord ("&_span ARM A@(N=&_sfybex)" flag ("&_span
Baseline@n (%)" t1l t1n t1h t1u))
         ("&_span ARM B@(N=&_stdopp)"  (" &_span Baseline@n (%)" t2l t2n t2h t2u));

    by param;

    define pagebrk  / noprint order order=data;
    define param    / noprint order order=data style={asis=on};
    define visitnum / order order=data style={cellwidth=100 Just=left} format=visord. "";
    define ord      / noprint order order=data style={asis=on};

    define flag     / display order order=data STYLE={ just=left}     left "#";
    define t1l      / display STYLE={just=left cellwidth=65 asis=on} left " L";
    define t1n      / display STYLE={just=left cellwidth=65 asis=on} left " N";
    define t1h      / display STYLE={just=left cellwidth=65 asis=on} left " H";
    define t1u      / display STYLE={just=left cellwidth=70 asis=on} left " U";
    define t2l      / display STYLE={just=left cellwidth=65 asis=on} left " L";
    define t2n      / display STYLE={just=left cellwidth=65 asis=on} left " N";
    define t2h      / display STYLE={just=left cellwidth=65 asis=on} left " H";
    define t2u      / display STYLE={just=left cellwidth=70 asis=on} left " U";

    compute after visitnum ;
        line "";
    endcomp;

    break after pagebrk / page;

run;
quit;

ods rtf close;

proc datasets nolist lib=work memtype=data kill;
```

```
quit;
title;
footnote;
```