New Tips and Tricks for Creating a Harmonized, Report-Friendly SDTM and ADaM Lab Data for a Clinical Study Report

Xiangchen (Bob) Cui, Vertex Pharmaceuticals, Cambridge, MA
Min Chen, Vertex Pharmaceuticals, Cambridge, MA
Scott Moseley, Vertex Pharmaceuticals, Cambridge, MA

ABSTRACT
A clinical study report typically calls for programming efforts to manipulate laboratory data in order to create LB SDTM and ADLB ADaM. There are many challenges that SAS® programmers face when working through these tasks. Lab data usually is the largest data set among clinical data sets, in addition to the number of analytes. It could be from different vendors, local and/or central labs. Its values could be numbers or characters. It could have different SI units for the same lab test. Developing a CDISC compliant LB SDTM and ADLB ADaM to facilitate a Clinical Study Report adds more work and more difficulties to the process.

This paper shows a new implementation strategy that integrates the requirements of the SDTM LB and ADLB ADaM to facilitate generating tables, graphs, and listings (TFL) by creating a harmonized, report-friendly data flow. It was developed when we were preparing NDA electronic submission. The new tips and tricks provided in this paper could be time-saving ones, and assist you in your programming for Lab to improve technical accuracy and operational efficiency.

INTRODUCTION
Developing CDISC compliant LB SDTM, and ADLB ADaM to facilitate a Clinical Study Report is very challenging in SAS programming, especially in dealing with large laboratory data from different vendors, local and/or central labs. This paper will illustrate how LB and ADLB are elegantly programmed with fulfillment of the logic, and an optimal data flow from LB to ADLB to facilitate the programming for TFL. While the tips and tricks are illustrated, some sample of SAS codes are provided as your reference.

LB SDTM
This paper will show how LB and SUPPLB are developed by following CDISC standards and how SUPPLB facilitates ADLB programming. The topics covered:
1. Centralize Controlled Terminology for Lab Test and Lab Test Code, SI Units, and SI Factors in Department Tools Library
2. Basic Lab Pre-processing
3. Identify LBTESTCD with both LBORRESU and LBSTRESU with NONE Value and Set Them to NONE
4. Harmonize LBTESTCD, LBTEST and LBCAT, and Populate Lab Reporting Precision from Vertex Standards
5. Handle the Records with SI Units in Lab Data Different from Vertex Standards
6. Impute LBSTRESN if LBORRES or LBSTRESC Contains Special Characters (<, <=, >, >=)
7. Handle Computational Precision
8. Bring Back Records with Status 'NOT DONE' and Populate Other Standard Variables for LB
9. Generate SUPPLB from LB to Facilitate Programming for ADLB and TFL

ADLB ADaM
This paper will provide some tips and tricks used to derive ADLB. The goal is to facilitate TFL programming, in addition to NDA submission. The topics covered:
10. Retrieve the Central Lab Flags, Precisions, and NOTAVAL for Lab Records from SUPPLB
11. Exclude Labs with Status ‘NOT DONE’ and Ones Flagged as NOTAVAL in SUPPLB
12. Derive Lab Reporting Label for Each Parameter to Facilitate TFL Reporting
13. Add Two Flags TABLEFL and LISTNGFL to Facilitate TFL Reporting
14. Derive Baseline Values from the Median (Mean) of All Pre-treatment Records Per SAP
   • Derive LBSEQ for the Baseline Records by Adding 0.5 to LBSEQ from the Last Record of Pre-treatment
   • Populate the Normal Range (LBSTNRLO and LBSTNRHI) of the Baseline from the Last Record of Pre-treatment
   • Derive NRBL (Baseline Reference Range Indicator) for Lab Shift Table

Figure 1 shows the process flow.

CENTRALIZE CONTROLLED TERMINOLOGY FOR LAB TEST AND LAB TEST CODE, SI UNITS, AND SI FACTORS IN DEPARTMENT TOOLS LIBRARY

Figure 2 and 3 show Vertex Master standard tables LBTEST and SIFACTOR from the department tools library. They are used in SDTM LB programming for controlled terminology of LBTEST and LBTESTCD, SI units, and SI conversion factors. Centralization of these information enhances consistency across studies, improves technical accuracy, and ensures CDISC compliance for LB SDTM.

TIPS & TRICKS: Detect LBTESTCD in lab data different from Vertex Standards. It prevents from underreporting lab data due to the omission of LBTEST whose LBTESTCD were different from CDISC controlled terminology for LBTEST and LBTESTCD. Report the findings to both Librarian and DM to work out the mapping rule for these LBTESTCD or to enrich library, and document the mappings in define.pdf, if it is for the NDA submission. Please refer to the next third section for further explanation and examples.
BASIC LAB PRE-PROCESSING

TIPS & TRICKS:

1. Exclude lab records with status 'NOT DONE' before processing, and bring them back at the end. It will save a lot of program running time and avoid some unexpected programming errors.

2. Flag the records with missing Lab dates and report them to DM for resolution.

3. Identify records with duplicates and report them to DM for resolution. These "irregular" records should be cleaned before database lock (DBL). They will cause unexpected programming errors. The process should continue until the DM's resolution or the rule proposed by study statisticians. To facilitate development of SAS program for LB, temporarily choose the first of duplicates.

Sample SAS Code

```sas
data not_done;
lab2;
set lab1;
if lbstat='NOT DONE' then output not_done;
else output lab2;
runk;
```

```sas
data missing_dt; *** check missing lab dates;
set lab2(where=(lbdt=\.)) ;*** report them to DM for data issue;
runk;
```

```sas
data dupes; *** check dupes;
set lab2;
by subjid lbtestcd lab_dt labt_dt visit lbrefid lbnam lborres;
if first.labt_dt ^= last.labt_dt;
runk;
```

IDENTIFY LBTESTCD WITH BOTH LBORRESU AND LBSTRESU WITH NONE VALUE AND SET THEM TO NONE

TIPS & TRICKS: Identify LBTESTCD with none units (lborresu and lbstresu) from Vertex standards and set them to NONE in lab data. It differentiates lab records with missing units from ones without any units. The assignment of NONE to these records will prevent error messages from the report when SDTM data sets are loaded into WEBSDM for CDISC compliance checking, in addition to facilitate SDTM conversion programming. For example, PTINR (PROTHROMBIN TIME INR) does not have any units.
Sample SAS Code

```sas
proc sql noprint;
select distinct "'"||strip(lbtestcd)||"'
into: nonesiunit separated by ','
from labref.lbtest /** from Vertex Standards **/
where upcase(trim(left(lbstresu)))='NONE';
quit;
** set its original unit (lborresu) and SI unit (lbstresu) to None;
data lab;set lab;
  if lb_lbtestcd in (&nonesiunit) and
  (upcase(trim(left(lb_lborresu)))='None' or upcase(trim(left(lb_lborresu)))='NONE') then
    do;
      lb_lborresu="None";
      lb_lbstresu="None";
      ** populate SI by original results;
      if compress(lb_lborres, '.0123456789')='0' and lb_lbstresn=. then do;
        lb_lbstresn=input(lb_lborres,best.);
        lb_lbstresc=trim(left(lb_lborres));
      end;
    end;
  end;
run;
```

HARMONIZE LBTESTCD, LBTEST, AND LBCAT AND POPULATE LAB REPORTING PRECISION FROM VERTEX STANDARDS

TIPS & TRICKS: Identify LBTESTCD in lab data different from Vertex standards to report to Department Librarian, for example, “CREATPK” was not in Vertex standards, and should be mapped into “CK”; update LBTEST and LBCAT by using Vertex standard if it is in the Vertex standards; populate lab reporting precision from Vertex standards.

Sample SAS Code

```sas
data lbtestcd_bad /** list of LBTESTCD were not in the vertex standards **/
lab;
  merge lab(in=a) lbtest(in=b rename=(lbstresu=lbtest_lbstresu));** precision;
  by lb_lbtestcd; ** labref.lbtest.lbprec;
  if a and not b then output lbtestcd_bad;
  uporresu=trim(left(upcase(lb_lborresu))); *** sifactors and updating SI units;
  lbtest_lbstresu1=trim(left(upcase(lbtest_lbstresu)));  *** from vertex standards;
  if a and b then do;
    ** update lbtest by using Vertex standard lbtest and lbcat;
    if lbtest='' then lb_test=lbtest;
    else lb_test=upcase(lb_test);
    if lbcat='' then lb_cat=lbcat;
    else lb_cat=upcase(lb_cat);
  end;
  if a then output lab;
run;
data lab;
  set lab; *** remap lbestcd into department standard ones;
  if lb_lbtestcd in ("CREATPK") then lb_lbtestcd="CK";
  if lb_lbtestcd="MCHC" and upcase(lb_lborresu) in ("PERCENT","%") then do;
    lb_lbtestcd="MCHC_P";
    lb_lbstresu="PERCENT";
  if compress(lb_lborres, '.0123456789')='0' then
    lb_lbstresn=input(lb_lborres,best.);
    lb_lbstresc=trim(left(lb_lborres));
  end;
run;
```

HANDLE THE RECORDS WITH SI UNITS IN LAB DATA DIFFERENT FROM VERTEX STANDARDS

TIPS & TRICKS:

1. Get SIFACTORS and SI units from Vertex standards (labref.sifactors), identify LBTESTCD whose SI units are different from Vertex standards (LBTEST_LBSTRESU);
2. Update SI units from Vertex standards if there is only case difference
3. Update all values related to SI by using original ones if LBORRESU is same as SI unit from Vertex standards and LBORRES is a valid numeric value.

4. Flag the record to be updated in next data step by using SIFACTOR (siunit_update=1) if its SIFACTOR is available in Vertex standards; otherwise flag this record as "NO UPDATE" (nosiu_update=1), and flag it as "NOT IN ADAM AVAL" (not_in_adam=1). If LBSTRES or LBSTRESC is "non-missing", all values will be kept, the flag "NOT IN ADAM AVAL" will be written into SUPPLB to indicate this record will not be used for any analysis.

5. Report SI units, which were not in Vertex standards, to Librarian when one of LBSTRESU and LBTEST_LBSTRESU is not “NONE” and LBSTRESC is “non-missing”, for example, for “HCT” (HEMATOCRIT) with non-standard SI unit LBSTRESU = ‘NONE’, the librarian adds SIFACTOR=0.01 in the department tools library when LBORRESU="PERCENT" and LBTEST_LBSTRESU="PROPORTION OF 1.0".

Sample SAS Code

```sas
data lbstresu_bad;
  merge lab0(in=a) labref.sifactors(in=b);
  by lb_lbtstcd uporresu lbtest_lbstresu;
  if a; *** SI units from lab dataset(lb_lbstresu) were different from Vertex standards(lbtest_lbstresu);
    if trim(left(lb_lbstresu)) ^= trim(left(lbtest_lbstresu)) and ((upcase(trim(left(lbtest_lbstresu)))^='NONE' and upcase(trim(left(lb_lbstresu)))^='NONE') or (upcase(lb_lborresu) ^= 'NONE' and upcase(trim(left(lb_lbstresu)))='NONE')) then do;if
      upcase(trim(left(lbtest_lbstresu)))=upcase(trim(left(lb_lbstresu))) then
        lb_lbstresu=lbtest_lbstresu; *update SI units with standards for case difference;
      else
        if upcase(trim(left(lbtest_lbstresu)))=upcase(trim(left(lb_lborresu)))
          and compress(lb_lborres, .'0123456789')=''
          then do; *** use the original to populate SI's;
            lb_lbstresu=lb_lborresu;
            lb_lbstresn=input(lb_lborres,best.);
            lb_lbstreasc=trim(left(lb_lborres));
            lb_lbstnrlr=lb_lbornrlr;
            lb_lbstnhr=lb_lbornhr;
            siunit_update=0;
          end;
        else if trim(left(sifactor)) ^='' then siunit_update=1;
        else if trim(left(sifactor)) ='' and compress(lb_lbstresc, .'0123456789')=''
          then do;
            nosiunit_update=1;
            not_in_adam=1;
          end;
    end;
  end;
  if upcase(trim(left(lbtest_lbstresu)))=upcase(trim(left(lb_lbstresu))) then
    lb_lbstresu=lbtest_lbstresu; ** always use Vertex standards(lbtest_lbstresu);
  output lab1;
  ** report SI units, which were not in Vertex standards, to Librarian ** *
  if(upcase(trim(left(lb_lbstresu)))='NONE' and upcase(trim(left(lbtest_lbstresu)))='NONE') or (upcase(trim(left(lb_lbstresu)))='NONE' and upcase(trim(left(lbtest_lbstresu)))='NONE' and compress(lb_lbstreasc, .'0123456789')='') then do;
    lbstresu_bad=1;
    output lbstresu_bad;
  end;
run;
```

**IMPUTE LBSTRESN IF LBORRES OR LBSTRESC CONTAINS SPECIAL CHARACTERS (<, <=, >, >=)**

**TIPS & TRICKS:**

1. Impute LBSTRESN by multiplying 0.5 to numeric part of LBSTRESC if it contains '<' or '<=', and plus 1.0 to it if '>' or '>='; Similar to LBORRESN
2. Flag the record as "NOT IN ADAM AVAL" (not_in_adam=1) if LBSTRESU is different from LBTEST_LBSTRESU, and it will be flagged in SUPPLB
3. Impute LBSTRESN by SIFACTOR if siunit_update=1 and LBORRESN is numeric; since the records containing Special Characters (<, <=, >, >=) are imputed in Tips 1, LBSTRESN could not have been imputed by LBORRES containing Special Characters (<, <=, >, >=) had the imputation occurred before this step.

4. Impute LBSTNRLO and LBSTNRHI by multiplying SIFACTOR to LBORNRLO and LBORNRHI, respectively if it is numeric in Tips 3.

5. Update LBNRIND due to the imputation of LBSTRESN, LBSTNRLO, and/or LBSTNRHI in Tips 3 and 4.

6. Flag the record as “NOT IN ADAM AVAL” (not_in_adam=1) if siunit_update=1 and LBORRESN is missing.

Sample SAS Code

data lab3;
length tt ttns $20. lbnrind $8.;
set lab2;
   tt=compress(lb_lborres,' .0123456789');
   if length(compress(tt))=1 and tt in('<', '>') then do;
      impute_fg=1;
      if substr(tt,1,1)='<' then lb_lborres=(substr(compress(lb_lborres),2)+0.0)/2.0;
      else if substr(tt,1,1)='>' then
         lb_lborres=(substr(compress(lb_lborres),2)+1.0);
   end;
   else if length(compress(tt))=2 and tt in('<', '>') and substr(tt,1,1)='=' then do;
      impute_fg=1;
      if substr(tt,1,1)='<' then
         lb_lborres=(substr(compress(lb_lborres),3)+0.0)/2.0;
      else if substr(tt,1,1)='>' then
         lb_lborres=(substr(compress(lb_lborres),3)+1.0);
   end;
   *** get the numeric values for original results;
   if compress(tt)='' then lb_lborres=lb_lborres=0.0;
   if upcase(trim(left(lbtest_lbstresu)))=upcase(trim(left(lb_lbstresu))) then do;
      ttns=compress(lb_lbstresc,' .0123456789');
      if length(compress(ttns))=1 and ttns in('<', '>') then do;
         impute_fg=1;
         if substr(ttns,1,1)='<' then
            lb_lbstres=(substr(compress(lb_lbstresc),2)+0.0)/2.0;
         else if substr(ttns,1,1)='>' then
            lb_lbstres=(substr(compress(lb_lbstresc),2)+1.0);
      end;
      else if length(compress(ttns))=2 and ttns in('<', '>') and substr(ttns,2,1)='=' then do;
         impute_fg=1;
         if substr(ttns,1,1)='<' then
            lb_lbstres=(substr(compress(lb_lbstresc),3)+0.0)/2.0;
         else if substr(ttns,1,1)='>' then
            lb_lbstres=(substr(compress(lb_lbstresc),3)+1.0);
      end;
      if impute_fg=1 then lb_lbstres=put(lb_lbstres,best.);
   end;
   if impute_fg=1 and
      upcase(trim(left(lbtest_lbstresu)))=upcase(trim(left(lb_lbstresu))) and
      trim(left(sifactor))='0' then not_in_adam=1;
   if siunit_update=1 and lb_lborres>.Z then do; *** update the SI value ***;
      lb_lbstres=lb_lborres*(sifactor+0.0);
      if lb_lbornrlo='' then do;
         if compress(lb_lbornrlo,' .0123456789')='' then
            lb_lbornrlo=put((lb_lbornrlo+0.0)*(sifactor+0.0),best.);
         else put lb_SUBJID= lb_lbtestcd= lb_VISIT= lbdt= lb_lborres= lb_lbornrlo=;
      end;
      if lb_lbornrh='' then do;
         if compress(lb_lbornrh,' .0123456789')='' then
            lb_lbornrh=put((lb_lbornrh+0.0)*(sifactor+0.0),best.);
         else put lb_SUBJID= lb_lbtestcd= lb_VISIT= lbdt= lb_lborres= lb_lbornrh=;
      end;
      lb_lbstres=trim(left(lb_lbstres));lb_lbstresu=lbtest_lbstresu;
*** update lbnrind due to the updated SI values and normal ranges***

```sas
if lb_lbstnrl0 ne . and lb_lbstnrhi ne . then do;
    if lb_lbstresn eq . then lbnrind = ' ';
    else if lb_lbstresn le lb_lbstresn le lb_lbstnrl0 then lbnrind = 'NORMAL';
    else if lb_lbstnrl0 lt lb_lbstresn then lbnrind = 'HIGH';
    end;
else if lb_lbstnrl0 eq . and lb_lbstnrhi ne . then do;
    if lb_lbstresn eq . then lbnrind = ' ';
    else if lb_lbstnrl0 le lb_lbstresn le lb_lbstnrl0 then lbnrind = 'NORMAL';
    else if lb_lbstnrhi lt lb_lbstresn then lbnrind = 'HIGH';
    end;
else if lb_lbstnrl0 ne . and lb_lbstnrhi eq . then do;
    if lb_lbstresn eq . then lbnrind = ' ';
    else if lb_lbstnrl0 lt lb_lbstresn then lbnrind = 'NORMAL';
    else if lb_lbstnrl0 eq . and lb_lbstnrhi le lb_lbstresn then lbnrind = 'LOW';
end;
end;
if siunit_update=1 and lb_lborresn=. then not_in_adam=1;
if compress(lbnrind) ^= ' ' and compress(lb_lbnrind) ^= compress(lbnrind) and
    siunit_update=1 then lb_lbnrind=lbnrind;
*** handle computational precision;
if lbnrind = 'LOW' and abs(lbstresn-lbstnrlo)<1e-10 then lbnrind = 'NORMAL';
if lbnrind = 'HIGH' and abs(lbstresn-lbstnrhi)<1e-10 then lbnrind = 'NORMAL';
run;
```

**HANDLE COMPUTATIONAL PRECISION**

When calculated Lab SI values are used to derive variables by comparing their values with criteria, computational precision issues could occur. The examples are reference range indicators (Low, High, and Normal), PCSA (Predefined Clinical Significant Abnormal) Identifier, PCSA Identifier for Change, and DAIS (Grading the Severity of Adult and Pediatric Adverse Events).

Figure 4 shows that LBSTRESN and LBSTNRLO look like the same with value equal to 2.499. Hence LBNRIND should be set to 'NORMAL'. However their difference was -4.44089E-16 which derived the value of LBNRIND into 'LOW'. Figure 5 shows that LBNRIND was derived into 'NORMAL' from another SAS programmer (another user of server) when the different SAS programs were run to same dataset. It shows that the difference between LBSTRESN and LBSTNRLO was 0. The results for LBNRIND from two users were different. This situation occurs when one independent SAS programmer validates a LAB dataset.

**TIPS & TRICKS:** Add an extra condition to exclude the situation and adjust the result.

**Sample SAS Code**

The following SAS codes are added in the bottom of data step in last section for derivation of LBNRIND.

```sas
if lbnrind = 'LOW'  and abs(lbstresn-lbstnrlo)<1e-10 then lbnrind = 'NORMAL';
if lbnrind = 'HIGH' and abs(lbstresn-lbstnrhi)<1e-10 then lbnrind = 'NORMAL';
```

---

**Figure 4 Sample of LAB Records**

<table>
<thead>
<tr>
<th>USUBJD</th>
<th>LBTSTCD</th>
<th>LBTST</th>
<th>LBCAT</th>
<th>LBSTRESU</th>
<th>LBSTRESN</th>
<th>LBSTNRLO</th>
<th>LBSTNRHI</th>
<th>LBNRIND</th>
<th>LBOTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ABC-0000-101002</td>
<td>UREA</td>
<td>UREA</td>
<td>2.499</td>
<td>2.495</td>
<td>7.14</td>
<td>4.44089E-16</td>
<td>LOW</td>
<td>2006-10-07T17:23:00</td>
</tr>
<tr>
<td>2</td>
<td>ABC-0000-101002</td>
<td>UREA</td>
<td>UREA</td>
<td>2.499</td>
<td>2.495</td>
<td>7.14</td>
<td>0</td>
<td>NORMAL</td>
<td>2006-10-07T17:22:00</td>
</tr>
</tbody>
</table>

**Figure 5 Sample of LAB Records**

**BRING BACK RECORDS WITH STATUS 'NOT DONE' AND POPULATE OTHER STANDARD VARIABLES FOR LB**

**TIPS & TRICKS:** Populate other standard variables for LB to all lab data, including ones with status ‘NOT DONE’

**Sample SAS Code**
data lab4;
attrib &attrib;
set lab not_done(in=b);
........
LBTESTCD=trim(left(lb_lbstestcd));
LBTEST=trim(left(lb_lbstest));
if compress(lb_lbstnrlo,'.0123456789')='' then LBSTNRLO=1*lb_lbstnrlo;
if compress(lb_lbstnrhi,'.0123456789')='' then LBSTNRHI=1*lb_lbstnrhi;
LBNIND=trim(left(lb_lbnrind));
LBSTAT=trim(left(lb_lbstat));
LBNAM=trim(left(lb_lbnam));
LBMETHOD=trim(left(lb_lbmethod));
LBFAST=trim(left(lb_lbfast));
........
run;

SUPPLB

GENERATE SUPPLB FROM LB TO FACILITATE PROGRAMMING FOR ADLB AND TFL

TIPS & TRICKS: Populate Central Lab Flag, Precision, and NOTAVAL into SUPPLB, these flags and values will be used in ADLB programming

Sample SAS Code

```sas
data supplb(keep=&keep_supp label='Supplemental Qualifier for LB');
attrib &attrib_supp;
set lab;
rdomain="LB";qnam="CENTFLAG";qlabel="Central Lab Flag";idvar="&domain.SEQ";
idvarval=trim(left(lbseq));qorig="CRF";qeval="PRINCIPAL INVESTIGATOR";
if central=1 then qval='Y';else qval='N';output;
if lbprecn>=0 then do;
qnam="PREC";qlabel="Precision";qval=trim(left(lbprecn));qorig="ASSIGNED";
qeval="STATISTICIAN";output;
end;
if not_in_adam=1 then do;
qnam="NOTAVAL";qlabel="Will not be Derived as Analysis Value";qval='Y';
qorig="ASSIGNED";qeval="STATISTICIAN";output;
end;
run;
```

ADLB

RETRIEVE THE CENTRAL LAB FLAGS, PRECISIONS, AND NOTAVAL FOR LAB RECORDS FROM SUPPLB

TIPS & TRICKS:

1. The central lab flags (CENTFLAG) will be used to exclude local labs, for normally local laboratories will only be presented in data listings.

2. The precisions (LBPREC) will used in lab summary tables to determine the number of decimal places for each analyte. The example below shows that LBPREC was used in a macro call. Note: The source of precisions is from Vertex standards. Hence the consistency is ensured across all studies, in addition to efficiency. `LBRPTLBL` (Laboratory Test Label for Reports) in the macro will be introduced in the next second section.

```sas
%summaryStatsIncolumns *** Get Lab Summary Statistics ***;
( data = _adlb,
  outcolm = stats,
  bytitles = lbcat paramcd LBRPTLBL &_trtcd.,
  byrows = ablfl avisitn,
  usubjid = &_usubjid.,
  continuousVariables = lbstresn chg,
  variableOrderList = lbstresn chg,
  format = avisitn visitf.,
  precision = lbprec);
```

3. NOTAVAL flags lab records with non-missing LBSTRESN, but LBSTRESU different from Vertex standards. These records should not be included in any analysis.
Sample SAS Code

```sas
proc sort data=lb; by usubjid lbseq; run;
data supplb;
set supplb;
  lbseq = input(idvarval, best.);
  where qnam in ("CENTFLAG", "PREC", "NOTAVAL");
run;
proc sort data=supplb; by usubjid lbseq qnam; run;
proc transpose data=supplb out=supplb(drop=_name_ _label_);
  by usubjid lbseq id qnam idlabel qlabel var qval;
run;
proc sort data=supplb; by usubjid lbseq; run;
```

**EXCLUDE LABS WITH STATUS ‘NOT DONE’ AND ONES FLAGGED AS NOTAVAL IN SUPPLB**

**TIPS & TRICKS:** Exclude labs with status ‘NOT DONE’ and NOTAVAL='Y' to facilitate the process, and bring them back at the end. It will save a lot of program running time and avoid some unexpected programming errors.

Sample SAS Code

```sas
data labextra lb1;
  set lb_t in=a supplb in=b;
  if lbstat='NOT DONE' or NOTAVAL='Y' then output labextra;
  else output lb1;
run;
```

The followings are from the ADLB.log of one of our studies. Excluding the records of LABEXTRA with 401666 observations and 46 variables will speed up the running of the program and prevent the errors.

NOTE: There were 1070415 observations read from the data set WORK.LB_T.

NOTE: The data set WORK.LABEXTRA has 401666 observations and 46 variables.

NOTE: The data set WORK.LB1 has 668749 observations and 46 variables.

**DERIVE LAB REPORTING LABEL FOR EACH PARAMETER TO FACILITATE TFL REPORTING**

**TIPS & TRICKS:** LBRPTLBL (Laboratory Test Label for Reports) will be used for the titles in TFL programs.

<table>
<thead>
<tr>
<th>LAB Test or Examination Short Name</th>
<th>LAB Test or Examination Name</th>
<th>Standard Units</th>
<th>LBRPTLBL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ALB</td>
<td>ALBUMIN</td>
<td>g/L</td>
<td>Albumin (g/L)</td>
</tr>
<tr>
<td>2 ALP</td>
<td>ALKALINE PHOSPHATASE</td>
<td>U/L</td>
<td>Alkaline Phosphatase (U/L)</td>
</tr>
<tr>
<td>3 ALT</td>
<td>ALANINE AMINOTRANSFERASE</td>
<td>U/L</td>
<td>Alanine Aminotransferase (Sgpt) (U/L)</td>
</tr>
<tr>
<td>4 APOTAT</td>
<td>APOLIPOPROTEIN A1</td>
<td>g/L</td>
<td>Apolipoprotein A1 (g/L)</td>
</tr>
<tr>
<td>5 AST</td>
<td>ASPARTATE AMINOTRANSFERASE</td>
<td>U/L</td>
<td>Aspartate Aminotransferase (Sgot) (U/L)</td>
</tr>
<tr>
<td>6 BASO_A</td>
<td>BASOPHILS [ABSOLUTE]</td>
<td>X10E9/L</td>
<td>Basophils (Absolute) X10E9/L</td>
</tr>
</tbody>
</table>
Sample SAS Code
*** get the lab tests and their SI units;
proc freq data=lb0 noprint;
tables lbstresu/out=_out1(drop=percent);
tables lbcat*lbtestcd*lbtest*lbstresu/out=lb_cd_su(drop=percent);
run;
data lb_cd_su;
   set lb_cd_su;
   if upcase(left(trim(lbstresu)))='NONE' then lbstresu='';
run;
proc sort data=lb_cd_su;by lbcat lbtestcd lbstresu;run;
data lb_cd_su2;
   set lb_cd_su;
   by lbcat lbtestcd lbstresu;
   if last.lbtestcd; *** keep the one with SI unit if it has;
run;
*** get variables PARAM and LBRPTLBL;
data lb_cd_su3;
   length PARAM LBRPTLBL lbtest2 $80.;
   set lb_cd_su2;
   if findc(lbtestcd,'_') and scan(trim(left(lbstresu)),1,'_')='P' then PARAM = upcase(compbl(lbtest));
   else if lbstresu='' then PARAM = upcase(compbl(lbtest)||"("||trim(left(lbstresu))||"";
   else if lbstresu='PERCENT' then PARAM = upcase(compbl(lbtest));
   if lbstresu='PERCENT' then lbtest2=tranwrd(lbtest,"(PERCENT)",'');
   else lbtest2=lbtest;
   if lbstresu='' then LBRPTLBL=propcase(compbl(lbtest2)||"("||trim(left(lbstresu))||"";
   else LBRPTLBL=propcase(compbl(lbtest));
run;
proc sort data=lb_cd_su3;by lbtestcd;run;

ADD TWO FLAGS TABLEFL AND LISTNGFL TO FACILITATE TFL REPORTING

TIPS & TRICKS: TABLEFL (Selected Analysis Flag for Tables) and LISTNGFL (Selected Analysis Flag for Listings) are specified by SAP. The study statistician provides the spreadsheet, which is converted into a SAS dataset and merged into lab data set. These two flags are used to subset ADLB at the beginning of SAS programs for TFLs. They facilitate the TFL programming.

DERIVE BASELINE VALUES FROM THE MEDIAN (MEAN) OF ALL PRE-TREATMENT RECORDS PER SAP

TIPS & TRICKS: The definition of baseline values varies from study to study. The easiest one will be the last assessment before the first treatment, which is same as one in SDTM LB. The more complex one is the median or mean of all pre-treatment measurements.

1. Derive LBSEQ for the Baseline Records by Adding 0.5 to LBSEQ from the Last Record of Pre-treatment
   The baseline record is newly derived. To keep the sorting order of ADLB by USUBJID LBSEQ, one for baseline will be derived by adding 0.5 to LBSEQ from the last record of pre-treatment

2. Populate the Normal Range (LBSTNRLO and LBSTNRHI) of the Baseline from the Last Record of Pre-treatment
   The normal ranges (LBSTNRLO and LBSTNRHI) of pre-treatment measurements could be different from each other due to the timing. Among them, one must be chosen to derive variable: NRBL (Baseline Reference Range Indicator) for lab shift table. The last record of pre-treatment is the best candidate in term of timing.

3. Derive NRBL (Baseline Reference Range Indicator) for Lab Shift Table
   The lab shift table needs LBNRIND (Reference Range Indicator) from the newly derived baseline record. LBNRIND should be derived based on the normal range (LBSTNRLO and LBSTNRHI) of the last record of pre-treatment and the newly derived baseline value.
Figure 8 shows an example of lab records with the derived baseline records and variables NRBL below.

<table>
<thead>
<tr>
<th>USUBJID</th>
<th>PARAMCD</th>
<th>LBRPTLBL</th>
<th>PHASEFL</th>
<th>LBSEQ</th>
<th>AVISIT</th>
<th>ADTM</th>
<th>LRNRFND</th>
<th>NRBL</th>
<th>LBSTNRLO</th>
<th>LBSTNRHI</th>
<th>LBSTNRLD</th>
<th>LISTNGFL</th>
<th>AVAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>Pre-Treatment Phase (Actual)</td>
<td>997</td>
<td>Screening</td>
<td>11/23/2008 14:03:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.69</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>Pre-Treatment Phase (Actual)</td>
<td>930</td>
<td>Day 1 PreDose</td>
<td>09/30/2008 10:45:00</td>
<td>NORMAL</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>4.85</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>Pre-Treatment Phase (Actual)</td>
<td>3015</td>
<td>Baseline</td>
<td>NORMAL</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>4.07</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>910</td>
<td>Week 1</td>
<td>12/09/2008 09:15:00</td>
<td>NORMAL</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>4.22</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>910</td>
<td>Week 2</td>
<td>12/09/2008 10:20:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.83</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>911</td>
<td>Week 3</td>
<td>12/09/2008 11:30:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.56</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>912</td>
<td>Week 4</td>
<td>12/09/2008 12:45:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.28</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>913</td>
<td>Week 6</td>
<td>12/09/2008 14:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.06</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>914</td>
<td>Week 8</td>
<td>12/09/2008 14:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.16</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>916</td>
<td>Week 12</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.14</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>917</td>
<td>Week 16</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.16</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>918</td>
<td>Week 20</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.34</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>919</td>
<td>Week 24</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.03</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>920</td>
<td>Week 28</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>2.93</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>921</td>
<td>Week 36</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>2.92</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>922</td>
<td>Week 40</td>
<td>12/09/2008 15:00:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>2.92</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>ADC-miss-001001</td>
<td>BRC</td>
<td>Electrolytes (010.12.1)</td>
<td>On-Treatment Phase (Actual)</td>
<td>923</td>
<td>Safety Follow-Up</td>
<td>04/03/2009 09:40:00</td>
<td>LOW</td>
<td>NORMAL</td>
<td>3.0</td>
<td>5.4</td>
<td>3.77</td>
<td></td>
</tr>
</tbody>
</table>

Figure 8 Sample of Lab Records with the Derived Baseline Records and Variables NRBL

Sample SAS Code

```sas
proc sort data=lb3 out=lb_pre;
  by PARAMCAT PARAMCD PARAM LBRPTLBL STUDYID usubjid lbcat lbtestcd lbtest lbstresss
  phasefn lbd t lbmt lbmsf f1 TABLEFL LISTNGFL;
  where phasefn=10 and trtstdt>.Z and lbstress>.Z;
run;
data last_pre;set lb_pre;
  by PARAMCAT PARAMCD PARAM LBRPTLBL STUDYID usubjid lbcat lbtestcd lbtest lbstresss
  phasefn lbd t lbmt lbmsf f1 TABLEFL LISTNGFL;
  if last.phasefn;lbseq=lbseq +0.5;
run;
proc sort data=last_pre; by PARAMCAT PARAMCD PARAM TABLEFL LISTNGFL LBRPTLBL STUDYID lbprec
  usubjid lbcat lbtestcd;run;
*** get the median for all pre-treatment records for baseline values;
proc univariate data=lb_pre nograph noprint;
  by PARAMCAT PARAMCD PARAM TABLEFL LISTNGFL LBRPTLBL STUDYID lbprec usubjid lbcat
  lbtestcd lbtest lbstresss phasefn;
  var lbstress;
  output out=lb_base n=n median=BASE;
run;
data lb_base;
  length ABLFL $1. BASEC avisit $40.;
  set lb_base;
  avisitn = 950; avisit = 'Baseline'; lbstress=trim(left(put(BASE,best.)));
  lbstress=BASE; ABLFL = 'Y'; BASEC=trim(left(put(BASE,best.)));;
run;
proc sort data=lb_base; by PARAMCAT PARAMCD PARAM LBRPTLBL STUDYID usubjid lbcat lbtestcd;run;
data lb_base;
  merge lb_base(drop=lbstress);
  last_pre(keep=PARAMCAT PARAMCD PARAM TABLEFL LISTNGFL LBRPTLBL STUDYID
  usubjid lbcatlbtestcd lbstress lbstnrlo lbstnrhi lbnrind lbseq lbfast);
  by PARAMCAT PARAMCD PARAM LBRPTLBL STUDYID usubjid lbcat lbtestcd;
  lbstress=BASE;
  if lbstnrlo ne . and lbstnrhi ne . then do;
    if lbstress eq . then lbnrind = ' ';
    else if lbstnrlo le lbstress le lbstnrhi then lbnrind = 'NORMAL';
    else if lbstress lt lbstnrlo then lbnrind = 'LOW';
    else if lbstnrhi lt lbstress then lbnrind = 'HIGH';
    end;
    else if lbstnrlo eq . and lbstnrhi ne . then do;
      if lbstress eq . then lbnrind = ' ';
      else if lbstres le lbstress then lbnrind = 'NORMAL';
      else if lbstress lt lbstres then lbnrind = 'HIGH';
      end;
    end;
    else if lbstnrlo ne . and lbstnrhi eq . then do;
      if lbstress eq . then lbnrind = ' ';
      else if lbstress le lbstres then lbnrind = 'NORMAL';
      else if lbstress lt lbstres then lbnrind = 'LOW';
      end;
    end;
  end;
NRBL=lbnrind; label NRBL='Baseline Reference Range Indicator';
r
```

11
CONCLUSION
This paper introduces new tips and tricks to develop LB SDTM and ADLB ADaM to facilitate generation of tables, graphs, and listings (TFL). Some sample of SAS codes are provided as your reference. The tips and SAS codes in this paper will assist you in your lab programming for a clinical study report and further NDA submission, and we hope that they can make your life a little easier when you are working on lab data for a clinical study report.

REFERENCES

ACKNOWLEDGEMENTS
Appreciation goes to Dean Gittleman and Kelly Blackburn for their review and comments.

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

Name: Xiangchen (Bob) Cui, Ph.D.
Enterprise: Vertex Pharmaceuticals, Inc.
Address: 88 Sidney Street
City, State ZIP: Cambridge MA, 02139
Work Phone: 617-444-6069
Fax: 617-460-8060
E-mail: xiangchen_cui@vrtx.com

Name: Min Chen, Ph.D.
Enterprise: Vertex Pharmaceuticals, Inc.
Address: 88 Sidney Street
City, State ZIP: Cambridge MA, 02139
Work Phone: 617-444-7134
E-mail: min_chen@vrtx.com

Name: Scott Moseley, M.S.
Enterprise: Vertex Pharmaceuticals, Inc.
Address: 88 Sidney Street
City, State ZIP: Cambridge MA, 02139
Work Phone: 617-444-6162
E-mail: scott_moseley@vrtx.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.