PharmaSUG 2012 – Paper DS01

Transform Incoming Lab Data into SDTM LB Domain with Confidence

Anthony L. Feliu, Genzyme, A Sanofi Company, Cambridge, Massachusetts

ABSTRACT

Converting incoming lab data into a submission–ready format is often challenging and stressful to the programmer. Vendors, sponsors, and collection instruments all have their own structures and nomenclature for lab data. This paper details a process to mold disparate data into a unified whole.

In brief, verbatim values of each incoming data file will be mapped one-to-one into SDTM variables. Mapped data are then compared to a dictionary of tests which is held outside the program as the "gold standard." When a match is found, verbatim values are updated to CDISC-compliant dictionary terminology. Otherwise the record is flagged for review. Next, result values for quantitative tests are compared to the "preferred units" for that test. The incoming result will either be accepted, rescaled with help from a second dictionary of conversion factors, or flagged. Similarly, qualitative test results are standardized.

Moving terminology out of program code into external dictionaries provides excellent transparency and traceability. The approach is readily implemented. Both program code and dictionaries are maintainable and extensible across protocols or product lines.

INTRODUCTION

Clinical trials are designed to evaluate the safety and effectiveness of drug products. Of all the data collected, laboratory data are among the most versatile and informative.

Teams seldom work with a single source for lab data. Although most trials require investigators to submit samples to a central lab for analysis, local labs are used also. As therapies become more sophisticated, specialty tests are increasingly part of a trial plan. Even if all samples go to a central lab, a few tests may be subcontracted out. Lab capabilities aside, evolving technology, regulatory requirements, even new preferred–provider contracts negotiated by sponsors all mean that lab data in different formats will inevitably confront programmers during the R&D lifecycle of a new product.

Working with these data is by no means easy, particularly when the assignment is to harmonize multiple studies.

That said, frontline programmers are not entirely left to their own devices when confronted with lab data:

- The CDISC SDTM implementation guide[1] (SDTMIG) prescribes the general construction of the LB domain, leaving considerable discretion to the individual sponsors.
- A recommended data model for transfer of lab data has also been published by the CDISC board[2]. This helps with, but does not eliminate, the traditional mapping exercise.
- Finally, a very helpful thesaurus for common clinical lab analytes is distributed by National Cancer Institute (NCI)[3] and regularly updated.

These resources inform and guide what to do, but offer little help how to do it. To accept and embrace different data sources, and to recognize that change is inevitable, we have developed an effective system to manage our standards and data transformations. The purpose of this paper is to inform and equip the reader on its realization.

---

[1] See http://www.cdisc.org/sdtm.

[2] See http://www.cdisc.org/lab.

[3] See NCI Enterprise Vocabulary Services at http://www.cancer.gov/cancertopics/cancerlibrary/terminologyresources/cdisc.

A STRUCTURED APPROACH

The essence of standardizing lab data is as simple as committing to segregate data from program code. How this is put into operation is remarkably simple too.

- **Planning** — Plan the layout of the LB domain and store this design as metadata. Make decisions about how to name each test, and collect the list of definitive test names.
- **Programming** — Read each source dataset and map incoming variables one–to–one with LB variables. The resulting dataset structurally resembles an SDTM, but the nomenclature is non-compliant. By comparing mapped verbatim values to the definitive list of test names, that gets resolved.
- **Process Feedback** — In the ideal case, all source data are processed and accepted within the design of the programming and associated metadata. If so, voila! More commonly, the source data will have test names or result units which were unexpected or not recognized. A mechanism to identify and evaluate these values is an essential part of the data conversion process. Valid new data call for an update to the metadata so these items will be correctly handled on the next program run. In the case of discrepant data, the temptation to modify or program around the values should be avoided in favor of reporting it to those responsible for data cleaning.

We begin with the planning discussion. It's six steps, but do not be discouraged! Most teams already make these decisions. The only change is that our discussion structures them.

Let's get started!

STEP 1: PLAN HOW TO NAME LAB TESTS

Many teams begin the design exercise by looking at source data and deciding what variables to include in their LB domain. These teams risk overlooking some requirement and generating a domain which is non-compliant. A more reliable—indeed easier—approach is to begin with the standard and build from there.

LB belongs to the **Findings** domain class. A minimum set of findings variables is shown in Table 1.

In practice, many more variables are not only required in the LB domain by the SDTM model[4] but needed to support the data.

Topic variable **LBTESTCD** and its associated synonym, **LBTEST**, carry controlled terminology in the implementation guide (SDTMIG), from which we conclude the topic is intended to be the lab analyte. But in most cases, an analyte name is not sufficient to identify a test.

Consider, for example, a test for "glucose":

Table 1. Minimum set of variables for a Findings Class domain (Many more needed for LB)

Variable	Label	Role	Core*
STUDYID	Study Identifier	Identifier	Req
DOMAIN	Domain Abbreviation	Identifier	Req
USUBJID	Unique Subject Identifier	Identifier	Req
--SEQ	Sequence Number	Identifier	Req
--TESTCD	Test or Exam Short Name	Topic	Req
--TEST	Test or Examination Name	Synonym Qualifier	Req
--ORRES	Result or Finding in Original Units	Result Qualifier	Exp
--STRESC	Character Result/Finding in Std Format	Result Qualifier	Exp
VISITNUM	Visit Number	Timing	Exp

* Required: Variable must be present and populated.
Expected: Variable must be present but may be blank.
Permitted: Optional variable

- The central lab likely runs serum through an automated analyzer.

[4] SDTM Implementation Guide 3.1.2, Chapter 6.3.3.

- At home, an insulin-dependent diabetic will draw a blood sample using a pinprick and capillary to monitor glucose levels using a test meter.
- A physician may screen for diabetes in an office setting using a patient's urine and a test strip.

To fully identify and distinguish these tests, **LBSPEC** (sample matrix) and **LBMETHOD** (test method) are useful:

```
-------------------------------------------
 LBTESTCD  LBTEST   LBSPEC  LBMETHOD
-------------------------------------------
  GLU      GLUCOSE  SERUM
  GLU      GLUCOSE  BLOOD   HOME TEST METER
  GLU      GLUCOSE  URINE   DIPSTICK
-------------------------------------------
```

The trio of variables LBTEST–LBSPEC–LBMETHOD is sufficient to tabulate the vast majority of records[5]. Be aware, however, that lab tests, particularly in the Urinalysis group, occasionally require sample collection for a defined time period, such as 24 hours. In this case, include permitted variable LBEVLINT (evaluation interval) to communicate the collection technique.

Findings class domains have two permitted variables relevant to the lab test names, "Category" and "Subcategory." Their function with the SDTM model is one of Grouping Qualifier, which means they help bundle related tests. For this reason, do not depend on them to identify specific tests.

Whether or not the product team chooses to have **LBCAT** and **LBSCAT** in the submission datasets, programmers are urged to derive these variables. (Drop them later.) The hierarchy provided by LBCAT–LBSCAT–LBTESTCD is indispensable to review the mapping. Related tests tend to have the same or similar units. Checking for units consistency is one of the fundamental quality control procedures in the programmer's armamentarium.

LBCAT and LBSCAT should carry controlled terminology. Table 2 gives an example *for illustration only*. With the understanding that the terminology for these variables is sponsor defined, programmers are advised to collaborate with statisticians and clinicians to decide what works best with their studies.

Table 2. Example lab test hierarchy

LBCAT	**LBSCAT**
CHEMISTRY	ELECTROLYTES ENZYMES LIPIDS METABOLIC PROTEINS TOXICOLOGY VITAMINS
HEMATOLOGY	COAGULATION ERYTHROCYTE COUNT ERYTHROCYTE INDICES ERYTHROCYTE MORPHOLOGY HEMOGLOBIN LEUKOCYTE COUNT LEUKOCYTE DIFFERENTIALS LEUKOCYTE MORPHOLOGY
IMMUNOLOGY	ANTIBODIES COMPLEMENT ENZYME INHIBITORS

[5] Oracle Health Sciences WebSDM™ product performs a series of data validation tests. One particular rule identifies inconsistent values for standard result units (LBSTRESU) using LBTEST–LBSPEC–LBMETHOD as the unique test identifier.

URINALYSIS	CHEMICAL PHYSICAL SEDIMENTATION

STEP 2: PLAN HOW TO REPORT RESULT VALUES

The Findings domain class provides for two result variables— an "original" result and a "standard" result. The "original" result variable is intended to hold the reported value. The "standard" result variable is then derived from it. When populating the standard result, programmers are obligated to format result values uniformly across all records.

Reporting lab results is more complex than a vanilla Findings domain. In LB, both "original" results and "standard" results are reported using sets of qualifier variables:

Table 3. Result variables in LB go beyond the minimum for a Findings class domain

Variable	Label	Role	Core
LBORRES	Result or Finding in Original Units	Result Qualifier	Exp
LBORRESU	Original Units	Variable Qualifier	Exp
LBORNRLO	Reference Range Lower Limit in Orig Unit	Variable Qualifier	Exp
LBORNRHI	Reference Range Upper Limit in Orig Unit	Variable Qualifier	Exp
LBSTRESC	Character Result/Finding in Std Format	Result Qualifier	Exp
LBSTRESN	Numeric Result/Finding in Standard Units	Result Qualifier	Exp
LBSTRESU	Standard Units	Variable Qualifier	Exp
LBSTNRLO	Reference Range Lower Limit-Std Units	Variable Qualifier	Exp
LBSTNRHI	Reference Range Upper Limit-Std Units	Variable Qualifier	Exp
LBSTNRC	Reference Range for Char Rslt-Std Units	Variable Qualifier	Perm

Generally speaking, **quantitative tests** have units and reference ranges associated with them, whilst **qualitative tests** have a result value only. Records may have no result values at all if the sample could not be analyzed. For these reasons, the Core attribute is "Expected" rather than "Required," even for LBORRES and LBSTRESC.

Keep in mind, for quantitative tests, "standard" result does not necessarily compel SI units. It is a sponsor decision.

From the programming perspective, however, it is essential to know if a given test, identified by LBTESTCD–LBSPEC–LBMETHOD, is expected to return a quantitative or a qualitative result. And in the case of quantitative tests, it is further necessary to decide in advance the preferred units, in order to recognize those records in the incoming data stream which require transformation before populating the standard result variables.

Additional information of benefit to the programmer includes the desired precision when a result value is expressed in the preferred units, and occasionally the lower– or upper–limit of quantification.

STEP 3: PLAN THE OVERALL STRUCTURE OF THE LB DOMAIN

After discussing test names and test results, the entire structure of the LB domain can be agreed upon.

Although the SAS® language makes it easy to embed variable attributes within a data step, this temptation should be resisted. A more efficient approach is to document the variables in a spreadsheet (or database table). Then the information is reusable by multiple programs. Some teams already segregate reusable *metadata* from program code in this way. Some do not.

For teams who do not, there is no better time to begin than the present. **Appendices A-1 and A-2** offer generic metadata from the author's data definitions. These samples will not cover all possible data scenarios, but form a solid starting point.

Our process to transform source data into SDTM begins with the LB metadata, because every SDTM variable must be accounted for in the mapping. But first, let's use the tabular metadata to make quick work of a few macro variables:

- Attrib statement with variables, attributes and labels (&ATTRIB_STATEMENT);
- Variable keep list (&KEEP_VAR);
- List of key variables (&DOMAIN_KEYS);
- Dataset label (&DOMAIN_LABEL).

Please note inclusion of a non-CDISC variable, DSN, in the design of Appendix A-2. The utility of this "operational" variable for QC will be described below. This variable would be calculated and stored in the LB dataset. Later, when the dataset is split into transport files for submission, it would be dropped.

STEP 4: PREPARE A MASTER DICTIONARY OF ALL LAB TESTS

A key piece of the standardization process is the lab test dictionary. This list holds all tests using SDTM terminology and will serve as the gold–standard for how lab test results will be reported in the LB domain. Whether the dictionary is maintained in a spreadsheet or a database, this will be the "go to" resource for programming.

At minimum, the dictionary must provide for the variables to be submitted in LB. It will also need to have result data type and preferred standard units. Consider adding other fields according to project needs.

- Grouping: category, subcategory
- Topic: test code, test name
- Qualifiers: specimen, method
- Helpers: record ID, data type, preferred standard units, standard result precision
- Optional: preferred SI units, SI result precision, limits of quantification, code list for character result, etc.

See **Appendix B** for an excerpt from the author's dictionary.

Programmers do not always have advance access to a comprehensive list of tests for a given set of protocols. Accordingly, developing the master lab test dictionary may be an iterative process. Regardless of how it is compiled, all project stakeholders—statisticians, clinicians, and programmers—will need to review and approve it. Keeping this information separate from program code is the best way to assure its review and its proper use in both the production and QC programming.

STEP 5: ANTICIPATE THAT QUANTITATIVE UNITS MUST BE CONVERTED

Inevitably, source records will arrive with a mixture of units for a given test, or in units different from those agreed upon by the stakeholders. Conversion factors will be needed to go from verbatim units to preferred units, for which a lookup table is recommended.

By recognizing two kinds of conversion factors, this is easy. When switching between two gravimetric units or two molecular units, a generic scaling factor serves for any analyte. But when switching between gravimetric units (grams per liter) and molecular units (moles per liter), the factor will be specific to that analyte.

The author maintains a table with four columns: analyte test code, original unit, standard unit, and factor. For generic factors, the analyte column is populated with the keyword "(ALL)".

See **Appendix C** for an excerpt from the author's dictionary.

STEP 6: PLAN HOW INCOMING DATA WILL BE MADE STANDARDS–COMPLIANT

The mapping dictionary is the glue between verbatim terms in the source and the master list of lab tests painstakingly compiled with standard terminology. Its structure is as simple as could be:

- DSN (for QC, the source dataset name);
- Verbatim category, test code or test name, specimen and method;
- An ID variable, assigned manually, to the corresponding entry in the master lab test dictionary.

In a moment, how the mapping dictionary is compiled and maintained will be explained. See **Appendix D** for an excerpt from the author's dictionary.

AT LAST, BEGIN PROGRAMMING

The programming approach is, by design, formulaic regardless of the record layout of the source dataset(s). We've done sufficient planning that the thought process behind our program code can be easily understood.

Step 1 Preprocess each incoming dataset:

- Write a data step which begins with an attrib statement for the target LB domain.
- In the set statement, rename or drop variables of the incoming dataset to avoid clashes with LB.
- Map all LB variables one–to–one from source.
- Accept result values with any units.
- Limit conditional logic unless it is necessary to parse source variables.
- Reformat dates to ISO standard.

```
data work.dsn_src1 ;
   attrib &attrib_statement ;              /* Get attrib from metadata.     */

   set rawdata.src1 (keep = subjid visitnum plbnam lbdtm speccnd lbspec
                            battrnam tstcd tstnam cnv: si: alrtfl tstcom
                                           /* Rename to avoid clash with LB.*/
                    rename = (visitnum = src_visitnum
                              lbspec   = src_lbspec)
                    ) ;

   retain studyid "&studyid" domain "LB" ;
   retain lbseq . ;                        /* Sequence derived later.       */
   retain dsn "SRC1" ;                     /* Include DSN for traceability. */

   %ususbjid(subjid) ;

   if tststat in ('N' 'X') then do ;
      lbstat = 'NOT DONE' ;
   end ;
   else do ;
      lborres  = cnvresc ;
      lborresu = cnvu ;
      lbornrlo = cnvnrlo ;
      lbornrhi = cnvnrhi ;
      lbnrind  = put(alrtfl, $alrtfl.) ;
   end ;

   * [[ Derive all LB variables ]] ;
   * [[ Derivation logic mostly unique to the particular source dataset ]] ;

   keep &keep_var ;                        /* Get keep list from metadata.  */
run ;
```

Step 2 After all datasets are preprocessed, stack them together.

```
data work.all_dsn ;
   set work.dsn_: ;
run ;
```

Step 3 First time through, compile a unique list of the important variables to seed the mapping dictionary.

```
proc summary data = work.all_dsn nway missing ;
   class dsn lbcat lbtestcd lbtest lbspec lbmethod ;
   output out = here.unique_tests (drop = _:) ;
run ;
```

Copy these records to the (empty) mapping dictionary table. Begin matching these verbatim records to the sanctioned nomenclature of the master dictionary. Add records to the master dictionary when new tests are encountered in the data until all verbatim records can be mapped.

✓ Remember that the mapping dictionary (Appendix D) will be project specific, but the master dictionary (Appendix B) can and should be valid and usable by any project team.

Step 4 Merge the stacked datasets with the mapping dictionary. Copy source records without dictionary matches to an “orphans” table.

For matching records, update LB variables originally populated with source values to CDISC-compliant dictionary values. Updating variables can be metadata driven, as illustrated below.

At this point, the LB domain takes shape.

```
                                /* Compare variables in LB domain with the     */
                                /* master dictionary to see which variables    */
                                /* need to be updated with dictionary terms.   */
                                /* The  master dictionary may have more        */
                                /* variables than your particular protocol.    */
proc sql noprint ;
   select trim(b.var_name) || ' ' || trim(a.name)
          into :var_update separated by ' '
   from dictionary.columns a inner join metadata.variables b
   on (upcase(a.libname)  = 'METADATA' and
       upcase(a.memname)  = 'LBMAPPING' and
       upcase(b.dataset_name) = "&domain" and
       upcase(b.var_name) = upcase(tranwrd(a.name, 'DICT_', "&domain"))
      )
   order by 1 ;
quit ;
```

```
/* For example:
   %put &var_update ;
   LBCAT DICT_CAT LBMETHOD DICT_METHOD LBSCAT DICT_SCAT
   LBSPEC DICT_SPEC LBTEST DICT_TEST LBTESTCD DICT_TESTCD
*/
```

```
                                /* Combine master dictionary (Appendix B) with */
                                /* project-specific mappings (Appendix D).     */
                                /* In this paper, they are separate tables     */
                                /* (you may be storing them differently).      */
proc sort data = metadata.lbmapping out = work.lbmapping ;
   by rec_id ;
run ;

proc sort data = metadata.lbmaster out = work.lbmaster ;
   by rec_id ;
run ;

data work.lbmapping ;
   merge work.lbmapping (in = in_map)
         work.lbmaster ;
   by rec_id ;
   if in_map ;
run ;

                                /* Prepare to merge data w/ mapping-dictionary.*/
proc sort data = work.all_dsn ;
   by dsn lbcat lbtestcd lbtest lbspec lbmethod lborresu ;
run ;

proc sort data = out = work.lbmapping ;
   by dsn lbcat lbtestcd lbtest lbspec lbmethod ;
run ;
```

```
                              /* Merge data with mapping-dictionary.         */
data here.lb_all
     work.lb_orphans  (keep = dsn lbcat lbtest: lbspec lbmethod) ;

   merge work.all_dsn    (in = in_all)
         work.lbmapping (in = in_map) ;

   by dsn lbcat lbtestcd lbtest lbspec lbmethod ;

   if in_all ;               /* Keep all lab data and dictionary matches.   */

   if in_map then do ;       /* Update verbatim qualifiers from dictionary. */
      array lb (*) &var_update ;
      do i = 1 to dim(lb) by 2 ;
         lb[i] = trim(lb[i + 1]) ;
      end ;
   end ;

   else output work.lb_orphans ;     /* Remember tests not in the mappings.*/

   drop i ;

   output here.lb_all ;

run ;
```

✓ The "orphans" table is very important. Add these "new" tests in the mapping dictionary so that the next program run will recognize them.

```
proc summary data = work.lb_orphans nway missing ;
   class dsn lbcat lbtest: lbspec lbmethod ;
   output out = here.lb_orphans (drop = _type_) ;
run ;
```

Step 5 Make a second pass through the data, this time to reconcile the collected data against the dictionary standards. Using the dictionary attribute "result data type" (REF_TYPE), logic for either quantitative or qualitative results is applied.

- For quantitative tests, verbatim units are compared to dictionary–preferred units (REF_STRESU). If different, a conversion factor is sought and the result variables are converted.
- When the verbatim units cannot be converted to preferred units, the record is saved to an "exceptions" table for review and action. Sadly common in collected data, records missing units cannot be converted and are a special case of this rule.
- For qualitative tests, only basic standardization can be performed—expand abbreviations and correct spelling.

The program code to reconcile the collected data against the dictionary standard makes dense reading. Comments with brackets show where this presentation was abbreviated.

```
data here.all_data       (keep = &keep_var ref_type)
     work.lb_exceptions (keep = lbcat lbscat lbtestcd lborresu lbstresu
                                dsn ref_stresu ref_type ;
   set here.all_data ;
   * [[ Remember to initialize working variables used below ]] ;
                            /*-----------------------------------------------*/
                            /* Standardize QUANTITATIVE result values       */
                            /* (REF_TYPE = N, numeric).                     */
                            /* Three scenarios within nested-if logic:      */
                            /* a. Verbatim matches preferred ;              */
                            /* b. Verbatim different from preferred ;       */
                            /* c. No preferred units for given test.        */
                            /*-----------------------------------------------*/
   if ref_type eq 'N' and lbstat ne 'NOT DONE' then do ;
                            /* a. Verbatim units already match preferred.   */
                            /*    Take dictionary units for capitalization. */
      if upcase(lbstresu) = upcase(ref_stresu) then do ;
         lbstresu = ref_stresu ;
         %check_precision ;    /* ... Optionally, control precision.     */
      end ;
                            /* b. Verbatim units different from preferred.  */
      else if not missing(ref_stresu) then do ;
         str1  = upcase(trim(coalescec(lbstresu, '(NONE)')) || '|'
                 || trim(ref_stresu)) ;
                               /* See Appendix C for informat definition.  */
                               /* FACT1 is test-independent scaling factor.*/
                               /* FACT2 is test-specific conversion factor.*/
         factor1 = input('(ANY)|' || str1, lbfact.) ;
         factor2 = input(trim(lbtestcd) || '|' || str1, lbfact.) ;
                               /* ... If test-independent factor found,    */
                               /* best to use that one.                    */
         if not missing(factor1) then do ;
            * [[ Convert LBST* variables using FACTOR1 ]] ;
            %check_precision ;
         end ;
                               /* ... Else, use test-specific factor.      */
         else if not missing(factor2) then do ;
            * [[ Convert LBST: variables using FACTOR2 ]] ;
            %check_precision ;
         end ;
                               /* ... If no luck, report the exception.    */
         else if not missing(lbstresn) then do ;
            output work.lb_exceptions ;
         end ;
      end ;
                            /* c. Dictionary does not have preferred units. */
      else do ;
         %check_precision ;
      end ;
   end ;
```

```
                              /*-----------------------------------------------*/
                              /* Standardize QUALITATIVE result values         */
                              /* (REF_TYPE = C, character).                    */
                              /*-----------------------------------------------*/
   if ref_type eq 'C' and lbstat ne 'NOT DONE' then do ;

                                 /* Presence of numeric value imples some-  */
                                 /* thing is amiss.  Write log note.        */
      if not missing(lbstresn) then do ;
         put "PROGRAM_" "NOTE: Unexpected numeric: " usubjid= lbcat=
             lbtestcd= ref_type= lbdtc= lborres= lborresu= lbstresn= ;

         lbstresn = . ;
      end ;

                                 /* It was not discussed in body of paper,  */
                                 /* but spelling corrections and expanding  */
                                 /* abbreviations are appropriate conver-   */
                                 /* sions for character results.            */

                                 /* This code block uses a similar approach */
                                 /* to the numeric factors, namely to have  */
                                 /* either generic translations (NEG to     */
                                 /* NEGATIVE) or test-specific ones (AA to  */
                                 /* HGB A2 for test HGBTYP).  A format      */
                                 /* works well to manage these decisions.   */
      if not missing(lbstresc) then do ;
         str1 = put('(ANY)|' || trim(upcase(lbstresc)), $lbsyn.) ;
         str2 = put(trim(lbtestcd) || '|' || trim(upcase(lbstresc)),$lbsyn.);
         if not missing(str1) then lbstresc = trim(str1) ;
         else if not missing(str2) then lbstresc = trim(str2) ;
      end ;

                                 /* Some labs will place the normal range   */
                                 /* in reported lower reference range       */
                                 /* (LBORNRLO).  If this is the case with   */
                                 /* your data, copy value from LBORNRLO to  */
                                 /* standard variable LBSTNRC and apply     */
                                 /* conversion logic along these lines.     */
      if missing(lbstnrlo) and not missing(lbornrlo) then do ;
         lbstnrc = upcase(trim(left(lbornrlo))) ;
         str1 = put('(ANY)|' || trim(lbstnrc), $lbsyn.) ;
         str2 = put(trim(lbtestcd) || '|' || trim(lbstnrc), $lbsyn.) ;
         if not missing(str1) then lbstnrc = trim(str1) ;
         else if not missing(str2) then lbstnrc = trim(str2) ;
      end ;

   end ;

   output here.lb_all ;

run ;
```

Notice how the above code will leave quantitative values in the standard result variables even if the units are non-conforming. The program logic cannot risk withholding data. Out of sight is out of mind!

```
proc summary data = work.lb_exceptions nway missing ;
   class _all_ ;
   output out = here.lb_exceptions (drop = _type_) ;
run ;
```

✓ Make careful review of exception records. Frequently, non-conforming units are resolved by adding conversion factor(s) to the dictionary. Sometimes the data are dirty and there is nothing the programmer can do except to await its cleaning by data management. Prior to database lock however, all records should be clean and conforming. If a study closes and discrepant records remain, it is then a team decision how to populate the standard result variables.

Step 6 Sort the candidate LB dataset using the natural key variable set. Compute sequence number LBSEQ and study day LBDY.

```
proc sort data = here.lb_all ;
   by &domain_keys ;        /* Get key variables from metadata.            */
run ;

                            /* An informat is built from SDTM.DM, with     */
                            /* USUBJID as start value, and a numeric date  */
                            /* from RFSTDTC as the label. A merge between  */
                            /* HERE.LB_ALL and SDTM.DM would work too.     */
%rfstdtc_informat ;

data sdtmplus.lb (label = "&domain_label") ;    /* Get label from metadata. */

   set here.lb_all ;

   length _seq _rfdt _lbdt 8 ;
   format _rfdt _lbdt date9. ;

   retain _seq ;
   if first.usubjid then _seq = 1 ;
   else _seq = _seq + 1 ;
   lbseq = _seq ;

   _rfdt = input(usubjid, ?? rf.) ;
   _lbdt = input(lbdtc, ?? yymmdd10.) ;
   if nmiss(_rfdt, _lbdt) eq 0 then lbdy = lb_dt - rf_dt + (lb_dt >= rf_dt) ;

   drop _: ;

run ;
```

Step 7 Review and resolve issues uncovered in the “orphans” and the “exceptions” tables.

Run a data quality listing (Appendix F) to assess the consistency and correctness of the mappings.

Once all issues are resolved, proceed with steps 8 and 9.

Step 8 Split SDTMPLUS.LB into its constituent deliverables as planned with data–definition attribute VAR_XPT:

- VAR_XPT = D Direct findings class variables to lab domain LB.XPT;
- VAR_XPT = S Split supplemental qualifiers into SUPPLB.XPT;
- VAR_XPT = C Comment variable(s) are split into another dataset for eventual compilation in Comments domain CO.XPT.
- VAR_XPT = O Operational variables are excluded from the deliverable.

Step 9 Enjoy !

PROCESS FEEDBACK

Integral to any business process is visibility into its performance. As byproduct of this structured programming, two tables were generated with records that were incompletely processed.

The **"orphans" table** captured incoming data not currently mapped in the dictionaries. This table has combinations of verbatim terms (mapping was one-to-one) that had no corresponding matches in the mapping dictionary.

Appropriate corrective actions include updating the mapping dictionary to reflect new verbatim terms for a known test in the master lab test dictionary, OR defining a new test in the master dictionary followed by updating the mapping dictionary, OR reporting the bogus record to the data management group for query and correction.

Table 4. Sample Orphans Table (Verbatim terms without matches in the mapping dictionary)

DSN	LBCAT	LBTESTCD	LBTEST	LBSPEC	LBMETHOD	FREQ
LBX	SERUM CHEMISTRY		CHLORIDE (CL)			20
LBXQ		CHT368	BICARBONATE	BLOOD		16
CGX			PLATELET COUNT			3
LBX	COMPLETE BLOOD COUNT		PLATELETS			10
LAB	HEMATOLOGY		PLATELETS			5
LBX5	HEMATOLOGY		PLATELETS			100
LBX5H	HEMATOLOGY		PLATELETS			18

- In this sample print, all tests are valid data. By additions to the mapping dictionary (Appendix D) these tests can be converted to SDTM nomenclature in the next program run.

The **"exceptions" table** captured incoming data records where current values could not be converted to the master dictionary's preferred units. Corrective actions in this case include researching and adding a new conversion factor to the dictionary OR reporting discrepant data to the data management group for query and correction.

Table 5. Sample Exceptions Table (Quantitative results successfully mapped but which could not be converted to preferred units)

Lab Test Identifiers				Result Units as Collected		Preferred Units	Attempted Conversion		
LBCAT	LBSCAT	LBTEST CD	REF_ TYPE	LBORRESU	LBSTRESU	REF_ STRESU	From | To	DSN	FREQ
CHEM	METABOLIC	BUN	N	MMOL/L	MMOL/L	mg/dL	MMOL/L|MG/DL	LBXQ	14
HEMAT	COAGULATION	INR	N			RATIO	(NONE)|RATIO	LAB	18
HEMAT	COAGULATION	PLAT	N			10^9/L	(NONE)|10^9/L	LAB	28
HEMAT	ERYTHROCYTE COUNT	RBC	N			10^3/uL	(NONE)|10^3/UL	LAB	22
HEMAT	ERYTHROCYTE COUNT	RBC	N	cells/µL	cells/µL	10^3/uL	CELLS/µL|10^3/UL	LAB	6
HEMAT	ERYTHROCYTE COUNT	RBC	N	x10^12	x10^12	10^12/L	X10^12|10^12/L	LAB	97
HEMAT	ERYTHROCYTE COUNT	RBC	N	x10^6/µL	x10^6/µL	10^12/L	X10^6/µL|10^12/L	LBX	144
HEMAT	ERYTHROCYTE COUNT	RETI	N			10^3/uL	(NONE)|10^3/UL	LAB	30
HEMAT	ERYTHROCYTE COUNT	RETI	N	x10^3/µL	x10^3/µL	10^9/L	X10^3/µL|10^9/L	LAB	61
HEMAT	ERYTHROCYTE COUNT	RETI	N	x10^6/µL	x10^6/µL	10^9/L	X10^6/µL|10^9/L	LAB	16
HEMAT	ERYTHROCYTE INDICES	MCH	N	fL	fL	pg	FL|PG	LAB	10
HEMAT	ERYTHROCYTE INDICES	MCV	N	pg	pg	fL	PG|FL	LAB	10

- In this sample print, several tests do not have units, so conversion to preferred units could not be done.
- As a team decision, the one case where it might be safe to convert null units to preferred units is LBTESTCD=INR which is a unit-less ratio. A test-specific factor for this case is shown in the sample conversion factors dictionary—see last entry in Appendix C.

- Some records for LBTESTCD=MCH and MCV have units incompatible for these tests. Inference may be drawn that the collected results were inadvertently mixed. No action can be taken except to report the discrepancies to data management.
- Several other tests (BUN, RBC, and RETI) have valid collected units. After updates to the conversion factors dictionary (Appendix C), these test results can be standardized to preferred units on the next program run.

Once the "orphans" and "exceptions" tables both contain zero observations, you and your colleagues are eligible for Step 9.

When things go wrong, these tables help you. So how do you know that everything went right? While there's not yet an app for that, there are **quick diagnostics**.

An easy check (often an eye–opening one) is to verify that the key variable set is adequate to uniquely identify records:

```
proc summary data = sdtmplus.lb nway missing ;
   class &domain_keys ;
   output out = work.summary (rename = (_freq_ = rec_cnt)) ;
run ;

proc print data = work.summary width = minimum ;
   var &domain_keys rec_cnt ;
   where rec_cnt gt 1 ;
run ;
```

Alternatively …

```
%let last_key = %scan(&domain_keys, -1, %str( )) ;

data work.lb_dupe ;
   set sdtmplus.lb ;
   by &domain_keys ;
   if first.&last_key and last.&last_key then delete ;
run ;
```

Datasets with a non–unique key variable set point to underlying mapping or data issues. Such situations can only be diagnosed by review of the data in question. Caution is advised, however, about *over specifying* key variables, because this can mask dirty data. In particular, if the key variable set has two timing variables, such as LBDTC and VISITNUM, test records mistakenly reported twice will escape notice.

A second helpful diagnostic is to compute univariate statistics for each test and review for normal or skewed distributions. This check is particularly informative with data pooled from multiple protocols. Appendix G offers a ready–to–run program, LBDQ.SAS. Sample output is too wide to show here. See Appendix F. Interpretations of that sample output:

- All Cholesterol records (CNT=122) have result values (N=122). Univariates show normal distribution.
- HDL Cholesterol has six records with missing result (CNT=6), but all are accounted by status "not done" (ND=6).
- Triglycerides have six records with status not done (ND=6) also. Of 122 records, 120 have result values (N=120). Two have missing result values (NMISS=2) but the record status is not "not done" (ND=0). Probably dirty data.
- Check the Hemoglobin Sickle Cell test. Notice character and numeric data types for the same test code. Records came from three panels (LBXQ, LBX3, LBX3H) suggesting a possible mapping inconsistency.
- Mean Cell Hemoglobin and Mean Cell Volume tests have a mixture of units. The units are incompatible. Report them to data management.

Equipped with orphans, exceptions, and diagnostics, the programmer is well prepared to identify problems in the data and take appropriate actions.

CONCLUSION

The system described has proven to be very robust in cross–protocol standardization of lab data. The dictionary files make transformations and nomenclature accessible for review by subject matter experts in the data management, medical, and statistical functional areas. All corrections and updates are easily made without risk to program code, which itself is very portable. This or similar system is highly recommended to any team preparing clinical trial data for submission.

POSTSCRIPT

This paper makes extensive use of program code. To support those algorithms, sample metadata are provided in the several appendices. Both code and metadata tables are slightly simplified yet remain true to the techniques the author has found effective in multiple studies. For anyone wishing to get started with lab data, the code and accompanying metadata will prove an invaluable beginning.

If there is a **single take–away** from this paper, surely it is to highlight the importance of **metadata** in reaching a predictable and reproducible deliverable. These metadata cannot and should not be locked in program code. Rather they should be centrally maintained. All programmers can then be trained to query the metadata in their various programs.

Every task has multiple programming solutions. Teams who find improvements or alternatives to the author's approach are invited and encouraged to share their discoveries with the community of programmers at next year's conference.

CONTACT INFORMATION

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

Anthony L. Feliu, PhD,
Principal Programmer,
Genzyme Corporation
500 West Kendall Street
Cambridge, MA 02142
Tel: (617) 768-9296
E-mail: ANTHONY.FELIU @ GENZYME.COM

APPENDIX A-1: SAMPLE DATA DEFINITION TABLE FOR LAB DOMAIN DATASET

Data definition tables at the dataset–level and variable–level specify and control construction of the SDTM domains. Prior to submission, these same metadata are needed to generate define–XML.

PROT_NUM	DATASET_NAME	DATASET_NUM	DATASET_LABEL	DATASET_KEYS	DATASET_DESCR	CDISC_CLASS
MYPROT1	LB	1	Laboratory Test Results	STUDYID DOMAIN USUBJID LBTESTCD LBSPEC LBMETHOD LBDTC	One record per test per sample drawn	Findings

APPENDIX A-2: SAMPLE DATA DEFINITION TABLE FOR LAB DOMAIN VARIABLES

DATA SET_ NAME	VAR_NAME	VAR_ NUM	KEY_ VAR	VAR_LABEL	VAR_ TYPE	VAR_ LEN	VAR_ XPT *	CODELIST	CDISC_ROLE	CDISC_ CORE	VAR_ORIGIN	VAR_QEVAL	DERIVATION
LB	DSN	1		Source Dataset Name	Char	20	O						
LB	STUDYID	2	1	Study Identifier	Char	50	D		Identifier	Req	Protocol		
LB	DOMAIN	3	2	Domain Abbreviation	Char	8	D		Identifier	Req	Derived		
LB	USUBJID	4	3	Unique Subject Identifier	Char	50	D		Identifier	Req	Derived		
LB	LBSEQ	5		Sequence Number	Num	8	D		Identifier	Req	Derived		
LB	LBREFID	6		Specimen ID	Char	20	D		Identifier	Perm	eDT		
LB	LBTESTCD	7	4	Lab Test or Examination Short Name	Char	8	D	(LBTESTCD)	Topic	Req	CRF; eDT		
LB	LBTEST	8		Lab Test or Examination Name	Char	40	D	(LBTEST)	Synonym Qualifier	Req	Derived		
LB	LBCAT	9		Category for Lab Test	Char	50	D	(LBCAT)	Grouping Qualifier	Exp	Assigned		
LB	LBSCAT	10		Subcategory for Lab Test	Char	50	D	(LBSCAT)	Grouping Qualifier	Perm	Assigned		
LB	LBORRES	11		Result or Finding in Original Units	Char	50	D		Result Qualifier	Exp	CRF; eDT		
LB	LBORRESU	12		Original Units	Char	50	D		Variable Qualifier	Exp	CRF; eDT		
LB	LBORNRLO	13		Reference Range Lower Limit in Orig Unit	Char	20	D		Variable Qualifier	Exp	eDT		
LB	LBORNRHI	14		Reference Range Upper Limit in Orig Unit	Char	20	D		Variable Qualifier	Exp	eDT		
LB	LBSTRESC	15		Character Result/Finding in Std Format	Char	50	D		Result Qualifier	Exp	Derived		
LB	LBSTRESN	16		Numeric Result/Finding in Standard Units	Num	8	D		Result Qualifier	Exp	Derived		
LB	LBSTRESU	17		Standard Units	Char	50	D		Variable Qualifier	Exp	Assigned		
LB	LBSTNRLO	18		Reference Range Lower Limit-Std Units	Num	8	D		Variable Qualifier	Exp	eDT		
LB	LBSTNRHI	19		Reference Range Upper Limit-Std Units	Num	8	D		Variable Qualifier	Exp	eDT		
LB	LBSTNRC	20		Reference Range for Char Rslt-Std Units	Char	50	D		Variable Qualifier	Perm	eDT		
LB	LBNRIND	21		Reference Range Indicator	Char	20	D	(NRIND)	Variable Qualifier	Exp	eDT		
LB	LBSTAT	22		Completion Status	Char	8	D	(ND)	Record Qualifier	Perm	CRF; eDT		
LB	LBREASND	23		Reason Test Not Done	Char	200	D		Record Qualifier	Perm	CRF; eDT		
LB	LBNAM	24		Vendor Name	Char	50	D	(LBNAM)	Record Qualifier	Perm	Assigned		

DATA SET_ NAME	VAR_NAME	VAR_ NUM	KEY_ VAR	VAR_LABEL	VAR_ TYPE	VAR_ LEN	VAR_ XPT *	CODELIST	CDISC_ROLE	CDISC_ CORE	VAR_ORIGIN	VAR_QEVAL	DERIVATION
LB	LBSPEC	25	5	Specimen Type	Char	50	D		Record Qualifier	Perm	CRF; eDT		
LB	LBMETHOD	26	6	Method of Test or Examination	Char	50	D		Record Qualifier	Perm	Derived		
LB	LBBLFL	27		Baseline Flag	Char	1	D	(YONLY)	Record Qualifier	Exp	Derived		
LB	VISITNUM	28		Visit Number	Num	8	D	(VISITN)	Timing	Exp	Assigned		
LB	VISIT	29		Visit Name	Char	50	D	(VISIT)	Timing	Perm	CRF; eDT		
LB	LBDTC	30	7	Date/Time of Specimen Collection	Char	20	D	ISO 8601	Timing	Exp	CRF; eDT		
LB	LBDY	31		Study Day of Specimen Collection	Num	8	D		Timing	Perm	Derived		
LB	LBCLSIG	32		Clinical Significance	Char	1	S	(NYONLY)	Result Qualifier	Perm	CRF	INVESTIGATOR	
LB	LBCOM	33		Investigator Comments	Char	200	C		Result Qualifier	Perm	CRF	INVESTIGATOR	
LB	LBLABCOM	34		Laboratory Vendor Comments	Char	200	C		Result Qualifier	Perm	eDT	CENTRAL LAB	

* VAR_XPT: D (domain variable), S (supplemental qualifier), C (comment variable), O (operational variable)

APPENDIX B: SAMPLE LAB TEST DICTIONARY

This table holds the definitive naming for all the lab tests. At minimum it will have the variables appearing in the LB domain. In this sample, preferred standard and preferred SI units are recorded. In actual practice, more variables are present, such as test name in proper case, controlled terminology for categorical tests, quantification limits, and provenance (sponsor–defined vs. CDISC).

REF_ID	DICT_CAT	DICT_SCAT	DICT_TESTCD	DICT_TEST	DICT_SPEC	DICT_METHOD	DICT_SPID	REF_TYPE	REF_STRESU	REF_STPREC	REF_SIU	more
1	CHEMISTRY	LIPIDS	CHOL	CHOLESTEROL	SERUM		CLCHOL	N	mg/dL	1	mmol/L	
2	CHEMISTRY	LIPIDS	HDL	HDL CHOLESTEROL	SERUM		CLHDL	N	mg/dL	1	mmol/L	
3	CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	CALCULATED	CLLDLC	N	mg/dL	1	mmol/L	
4	CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	MEASURED	CLLDLM	N	mg/dL	1	mmol/L	
5	CHEMISTRY	LIPIDS	TRIG	TRIGLYCERIDES	SERUM		CLTRIG	N	mg/dL	1	mmol/L	
6	HEMATOLOGY	COAGULATION	INR	PROTHROMBIN INTL. NORMALIZED RATIO	PLASMA		HCINR	N	RATIO	0.01		
7	HEMATOLOGY	COAGULATION	PLAT	PLATELETS	BLOOD		HCPLAT	N	10^9/L	0.1		
8	HEMATOLOGY	COAGULATION	PT	PROTHROMBIN TIME	PLASMA		HCPT	N	sec	0.1		
9	HEMATOLOGY	DIFFERENTIAL COUNT	BASOLE	BASOPHILS/LEUKOCYTES	BLOOD		HDBASOLE	N	%	1	PROPORTION OF 1.0	
10	HEMATOLOGY	DIFFERENTIAL COUNT	EOSLE	EOSINOPHILS/LEUKOCYTES	BLOOD		HDEOSLE	N	%	1	PROPORTION OF 1.0	
11	HEMATOLOGY	DIFFERENTIAL COUNT	LYMLE	LYMPHOCYTES/LEUKOCYTES	BLOOD		HDLYMLE	N	%	1	PROPORTION OF 1.0	
12	HEMATOLOGY	DIFFERENTIAL COUNT	RETIRBC	RETICULOCYTES/ERYTHROCYTES	BLOOD		HDRETIRB	N	%	0.1	PROPORTION OF 1.0	
13	HEMATOLOGY	ERYTHROCYTE COUNT	RBC	ERYTHROCYTES	BLOOD		HCRBC	N	10^12/L	0.01		
14	HEMATOLOGY	ERYTHROCYTE COUNT	RETI	RETICULOCYTES	BLOOD		HCRETI	N	10^9/L	0.01		
15	HEMATOLOGY	ERYTHROCYTE INDICES	HCT	HEMATOCRIT	BLOOD		HIHCT	N	%	0.1	PROPORTION OF 1.0	
16	HEMATOLOGY	ERYTHROCYTE INDICES	MCH	ERY. MEAN CORPUSCULAR HEMOGLOBIN	BLOOD		HIMCH	N	pg	0.1		
17	HEMATOLOGY	ERYTHROCYTE INDICES	MCHC	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	BLOOD		HIMCHC	N	g/dL	0.1	g/L	
18	HEMATOLOGY	ERYTHROCYTE INDICES	MCV	ERY. MEAN CORPUSCULAR VOLUME	BLOOD		HIMCV	N	fL	0.1		
19	HEMATOLOGY	ERYTHROCYTE MORPHOLOGY	ANISO	ANISOCYTES	BLOOD		HMANISO	C				
20	HEMATOLOGY	ERYTHROCYTE MORPHOLOGY	BURRCE	BURR CELLS	BLOOD		HMBURRCE	C				
21	HEMATOLOGY	ERYTHROCYTE MORPHOLOGY	CRENCE	CRENATED CELLS	BLOOD		HMCRENCE	C				
22	HEMATOLOGY	HEMOGLOBIN	FERRITIN	FERRITIN	BLOOD		HHFERRIT	N	ng/mL	1	pmol/L	
23	HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L	
24	HEMATOLOGY	HEMOGLOBIN	HGBA	HEMOGLOBIN A	BLOOD		HHHGBA	N	%	0.1	PROPORTION OF 1.0	
25	HEMATOLOGY	HEMOGLOBIN	HGBA2	HEMOGLOBIN A2	BLOOD		HHHGBA2	N	%	0.1	PROPORTION OF 1.0	
26	HEMATOLOGY	LEUKOCYTE COUNT	BASO	BASOPHILS	BLOOD		HCBASO	N	10^3/uL	0.01	10^9/L	
27	HEMATOLOGY	LEUKOCYTE COUNT	EOS	EOSINOPHILS	BLOOD		HCEOS	N	10^3/uL	0.01	10^9/L	
28	HEMATOLOGY	LEUKOCYTE COUNT	LYM	LYMPHOCYTES	BLOOD		HCLYM	N	10^3/uL	0.01	10^9/L	

APPENDIX C: SAMPLE CONVERSION FACTOR DICTIONARY

DICT_ TESTCD	UNIT_ORIG	UNIT_STD	CONV_ FACTOR
(ANY)	1	%	100
(ANY)	1	PROPORTION OF 1.0	1
(ANY)	%	PROPORTION OF 1.0	0.01
(ANY)	CELLS/µL	10^12/L	0.000001
(ANY)	CELLS/µL	10^9/L	0.001
(ANY)	CELLS/µL	10^3/UL	0.001
(ANY)	FRACTION	PROPORTION OF 1.0	1
(ANY)	G/DL	G/L	10
(ANY)	G/DL	MG/DL	1000
(ANY)	G/L	G/DL	0.1
(ANY)	MG/DL	G/L	0.01
(ANY)	MG/DL	MG/L	10
(ANY)	NG/ML	MG/L	0.001
BILDIR	MG/DL	UMOL/L	17.104
BILI	MG/DL	UMOL/L	17.104
BILIND	MG/DL	UMOL/L	17.104
BUN	MG/DL	MMOL/L	0.357
BUN	MMOL/L	MG/DL	2.80
CHOL	MG/DL	MMOL/L	0.0259
CHOL	MMOL/L	MG/DL	38.61
CL	MEQ/L	MMOL/L	1
CO2	MEQ/L	MMOL/L	1
CREAT	MMOL/L	MG/DL	11.31
CREAT	UMOL/L	MG/DL	0.01131
INR	(NONE)	RATIO	1

Stack test code, original unit, and standard unit as the starting value of an informat.
This avoids multiple sorts of a large lab dataset and keeps the standardization code concise.

```
data work.fmt ;
   set ref.lbfactors end = lastobs ;

   retain fmtname 'LBFACT' type 'I' hlo ' ' ;
   length start $100 ;
   rename conv_factor = label ;

   start = trim(dict_testcd) || '|' || trim(unit_orig) || '|' || trim(unit_std) ;

   output ;

   if lastobs then do ;
      start = '' ;
      conv_factor = . ;
      hlo = 'O' ;
      output ;
   end ;

run ;

proc format library = work cntlin = work.fmt ;
run ;
```

The author maintains this table in all-caps.
Definitive capitalization is in the preferred–units column of the master test dictionary.

APPENDIX D: SAMPLE MAPPING DICTIONARY

The mapping dictionary links to the master test dictionary using an ID variable, as shown.

	Verbatim terms from One-to-One Mapping					
DSN	LBCAT	LBTESTCD	LBTEST	LBSPEC	LBMETHOD	REF_ID
LAB	HEMATOLOGY		BASOPHILS			26
LCX	COMPLETE BLOOD COUNT	DNN	BASOPHILS			9
LBX	COMPLETE BLOOD COUNT		BASOPHILS			9
LBX	COMPLETE BLOOD COUNT		BASOPHILS ABS			26
LCX	COMPLETE BLOOD COUNT	AB4	BASOPHILS ABSOLUTE			26
LAB	HEMATOLOGY		BASOPHILS PCT			9
LAB	HEMATOLOGY		EOSINOPHILS			27
LCX	COMPLETE BLOOD COUNT	DLN	EOSINOPHILS			10
LBX	COMPLETE BLOOD COUNT		EOSINOPHILS			10
LBX	COMPLETE BLOOD COUNT		EOSINOPHILS ABS			27
LCX	COMPLETE BLOOD COUNT	AB3	EOSINOPHILS ABSOLUTE			27
LAB	HEMATOLOGY		EOSINOPHILS PCT			10
LCX	HB HPLC	HBA2	HB A2			25
LCX	COMPLETE BLOOD COUNT	HCTS1	HEMATOCRIT			15
LAB	HEMATOLOGY		HEMATOCRIT			15
LBX	COMPLETE BLOOD COUNT		HEMATOCRIT (HCT)			15
LAB	HEMATOLOGY		HEMATOCRIT PCT			15
LCX	COMPLETE BLOOD COUNT	HG9	HEMOGLOBIN			23
LBX5	HEMATOLOGY		HEMOGLOBIN			23
LBX5H	HEMATOLOGY		HEMOGLOBIN			23
LBX	COMPLETE BLOOD COUNT		HEMOGLOBIN (HGB)			23
LBXQ		RLT4557	HEMOGLOBIN A	BLOOD		24
LBXQ		RLT4558	HEMOGLOBIN A2	BLOOD		25
LBXQ		CHT298	LDL CHOLESTEROL	BLOOD		4
LBXQ		CHT307	LDL-C (CALCULATED)	BLOOD		3
LCX	CHEM PLUS	XLC	LDL-CHOL CALCULATION			3
LBX	SERUM CHEMISTRY		LOW-DENSITY LIPOPROTEIN (LDL)			4
LCX	COMPLETE BLOOD COUNT	MH9S1	MCH			16
LCX	COMPLETE BLOOD COUNT	CH9S1	MCHC			17
LAB	HEMATOLOGY		MEAN CORPUSCULAR HEMOGLOBIN			16
LBX	COMPLETE BLOOD COUNT		MEAN CORPUSCULAR HEMOGLOBIN (MCH)			16
LAB	HEMATOLOGY		MEAN CORPUSCULAR HEMOGLOBIN CONCENTRATIO			17
LBX	COMPLETE BLOOD COUNT		MEAN CORPUSCULAR HGB CONC (MCHC)			17

If dictionaries and mappings are maintained in spreadsheets, rather than use an ID variable for mapping as shown above, it will be more clear to copy/paste master dictionary records (Appendix B) next to the verbatim records (above), and work from there. For example:

Verbatim Terms from Source Data						Corresponding Records from Master Dictionary									
DSN	LBCAT	LBTESTCD	LBTEST	LBSPEC	LBMETHOD	DICT_CAT	DICT_SCAT	DICT_TESTCD	DICT_TEST	DICT_SPEC	DICT_METHOD	DICT_SPID	REF_TYPE	REF_STRESU	REF_STPREC	REF_SIU
LAB	HEMATOLOGY		BASOPHILS			HEMATOLOGY	LEUKOCYTE COUNT	BASO	BASOPHILS	BLOOD		HCBASO	N	10^3/uL	0.01	10^9/L
LCX	COMPLETE BLOOD COUNT	DIFF	BASOPHILS			HEMATOLOGY	DIFFERENTIAL COUNT	BASOLE	BASOPHILS/LEUKOCYTES	BLOOD		HDBASOLE	N	%	1	PROPORTION OF 1
LBX	COMPLETE BLOOD COUNT		BASOPHILS			HEMATOLOGY	DIFFERENTIAL COUNT	BASOLE	BASOPHILS/LEUKOCYTES	BLOOD		HDBASOLE	N	%	1	PROPORTION OF 1
LBX	COMPLETE BLOOD COUNT		BASOPHILS ABS			HEMATOLOGY	LEUKOCYTE COUNT	BASO	BASOPHILS	BLOOD		HCBASO	N	10^3/uL	0.01	10^9/L
LCX	COMPLETE BLOOD COUNT	ABA	BASOPHILS ABSOLUTE			HEMATOLOGY	LEUKOCYTE COUNT	BASO	BASOPHILS	BLOOD		HCBASO	N	10^3/uL	0.01	10^9/L
LAB	HEMATOLOGY		BASOPHILS PCT			HEMATOLOGY	DIFFERENTIAL COUNT	BASOLE	BASOPHILS/LEUKOCYTES	BLOOD		HDBASOLE	N	%	1	PROPORTION OF 1
LAB	HEMATOLOGY		EOSINOPHILS			HEMATOLOGY	LEUKOCYTE COUNT	EOS	EOSINOPHILS	BLOOD		HCEOS	N	10^3/uL	0.01	10^9/L
LCX	COMPLETE BLOOD COUNT	DLN	EOSINOPHILS			HEMATOLOGY	DIFFERENTIAL COUNT	EOSLE	EOSINOPHILS/LEUKOCYTES	BLOOD		HDEOSLE	N	%	1	PROPORTION OF 1
LBX	COMPLETE BLOOD COUNT		EOSINOPHILS			HEMATOLOGY	DIFFERENTIAL COUNT	EOSLE	EOSINOPHILS/LEUKOCYTES	BLOOD		HDEOSLE	N	%	1	PROPORTION OF 1
LBX	COMPLETE BLOOD COUNT		EOSINOPHILS ABS			HEMATOLOGY	LEUKOCYTE COUNT	EOS	EOSINOPHILS	BLOOD		HCEOS	N	10^3/uL	0.01	10^9/L
LCX	COMPLETE BLOOD COUNT	ABE	EOSINOPHILS ABSOLUTE			HEMATOLOGY	LEUKOCYTE COUNT	EOS	EOSINOPHILS	BLOOD		HCEOS	N	10^3/uL	0.01	10^9/L
LAB	HEMATOLOGY		EOSINOPHILS PCT			HEMATOLOGY	DIFFERENTIAL COUNT	EOSLE	EOSINOPHILS/LEUKOCYTES	BLOOD		HDEOSLE	N	%	1	PROPORTION OF 1
LCX	HB HPLC	HBAZ	HB AZ			HEMATOLOGY	HEMOGLOBIN	HGBAZ	HEMOGLOBIN AZ	BLOOD		HHHGBAZ	N	%	0.1	PROPORTION OF 1
LCX	COMPLETE BLOOD COUNT	HCTS1	HEMATOCRIT			HEMATOLOGY	ERYTHROCYTE INDICES	HCT	HEMATOCRIT	BLOOD		HIHCT	N	%	0.1	PROPORTION OF 1
LAB	HEMATOLOGY		HEMATOCRIT			HEMATOLOGY	ERYTHROCYTE INDICES	HCT	HEMATOCRIT	BLOOD		HIHCT	N	%	0.1	PROPORTION OF 1
LBX	COMPLETE BLOOD COUNT		HEMATOCRIT (HCT)			HEMATOLOGY	ERYTHROCYTE INDICES	HCT	HEMATOCRIT	BLOOD		HIHCT	N	%	0.1	PROPORTION OF 1
LAB	HEMATOLOGY		HEMATOCRIT PCT			HEMATOLOGY	ERYTHROCYTE INDICES	HCT	HEMATOCRIT	BLOOD		HIHCT	N	%	0.1	PROPORTION OF 1
LCX	COMPLETE BLOOD COUNT	HGB	HEMOGLOBIN			HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L
LAB	HEMATOLOGY		HEMOGLOBIN			HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L
LBX5	HEMATOLOGY		HEMOGLOBIN			HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L
LBX5H	HEMATOLOGY		HEMOGLOBIN			HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L
LBX	COMPLETE BLOOD COUNT		HEMOGLOBIN (HGB)			HEMATOLOGY	HEMOGLOBIN	HGB	HEMOGLOBIN	BLOOD		HHHGB	N	g/dL	0.1	g/L
LBXQ		RLT455	HEMOGLOBIN A	BLOOD		HEMATOLOGY	HEMOGLOBIN	HGBA	HEMOGLOBIN A	BLOOD		HHHGBA	N	%	0.1	PROPORTION OF 1
LBXQ		RLT455B	HEMOGLOBIN AZ	BLOOD		HEMATOLOGY	HEMOGLOBIN	HGBAZ	HEMOGLOBIN AZ	BLOOD		HHHGBAZ	N	%	0.1	PROPORTION OF 1
LBXQ		CHT2S2	LDL CHOLESTEROL	BLOOD		CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	MEASURED	CLLDLM	N	mg/dL	1	mmol/L
LBXQ		CHT3D7	LDL-C (CALCULATED)	BLOOD		CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	CALCULATED	CLLDLC	N	mg/dL	1	mmol/L
LCX	CHEM PLUS	XLC	LDL-CHOL CALCULATION			CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	CALCULATED	CLLDLC	N	mg/dL	1	mmol/L
LCX	CHEM PLUS	XLCSD	LDL-CHOL CALCULATION			CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	CALCULATED	CLLDLC	N	mg/dL	1	mmol/L
LBX	SERUM CHEMISTRY		LOW-DENSITY LIPOPROTEIN (LDL)			CHEMISTRY	LIPIDS	LDL	LDL CHOLESTEROL	SERUM	MEASURED	CLLDLM	N	mg/dL	1	mmol/L
LCX	COMPLETE BLOOD COUNT	MNHS1	MCH			HEMATOLOGY	ERYTHROCYTE INDICES	MCH	ERY. MEAN CORPUSCULAR HEMOGLOBIN	BLOOD		HIMCH	N	pg	0.1	
LCX	COMPLETE BLOOD COUNT	CHS61	MCHC			HEMATOLOGY	ERYTHROCYTE INDICES	MCHC	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	BLOOD		HIMCHC	N	g/dL	0.1	g/L
LAB	HEMATOLOGY		MEAN CORPUSCULAR HEMOGLOBIN			HEMATOLOGY	ERYTHROCYTE INDICES	MCH	ERY. MEAN CORPUSCULAR HEMOGLOBIN	BLOOD		HIMCH	N	pg	0.1	
LBX	COMPLETE BLOOD COUNT		MEAN CORPUSCULAR HEMOGLOBIN (MCH)			HEMATOLOGY	ERYTHROCYTE INDICES	MCH	ERY. MEAN CORPUSCULAR HEMOGLOBIN	BLOOD		HIMCH	N	pg	0.1	
LAB	HEMATOLOGY		MEAN CORPUSCULAR HEMOGLOBIN CONCENTRATION			HEMATOLOGY	ERYTHROCYTE INDICES	MCHC	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	BLOOD		HIMCHC	N	g/dL	0.1	g/L
LBX	COMPLETE BLOOD COUNT		MEAN CORPUSCULAR HGB CONC (MCHC)			HEMATOLOGY	ERYTHROCYTE INDICES	MCHC	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	BLOOD		HIMCHC	N	g/dL	0.1	g/L

APPENDIX E: DATA DEFINITIONS FOR ALL DICTIONARIES

Use these specifications to create the metadata tables presented in Appendices A–D. Once these structures are created, populate them with the sample data or your own data.

- VAR_NUM is the logical order of variables within a table.
- KEY_VAR is the subset of variables needed to identify unique records in the table. The logical order of key variables is the recommended sort order for data review.
- Attaching a CODELIST to certain variables may help with consistency of data entry.

DATA_SET	VAR_NAME	VAR_ NUM	KEY_ VAR	VAR_LABEL	VAR_ TYPE	VAR_ LEN	CODELIST	NOTES
METADATA.DATASETS	PROT_NUM	1	1	Protocol Number	Char	50		
	DATASET_NAME	2	2	Dataset Name	Char	8		
	DATASET_NUM	3		Submission Order	Num	8		
	DATASET_LABEL	4		Dataset Label	Char	40		
	DATASET_KEYS	5		Natural Key Variables	Char	200		Uniquely identifies records, domain sort order.
	DATASET_DESCR	6		Dataset Description	Char	200		
	CDISC_CLASS	7		Dataset Class	Char	50	(METCLAS)	Events, Findings, Interventions, etc.
METADATA.VARIABLES	PROT_NUM	1	1	Protocol Number	Char	50		(Foreign Key DATASETS)
	DATASET_NAME	2	2	Dataset Name	Char	8		(Foreign Key DATASETS)
	VAR_NAME	3	3	Variable Name	Char	8		
	VAR_ NUM	4		Variable Logical Order in Dataset	Num	8		
	KEY_ VAR	5		Key Order	Num	8		
	VAR_LABEL	6		Variable Label	Char	40		
	VAR_TYPE	7		Variable Data Type	Char	1	(METTYPE)	C (Char), N (Num)
	VAR_LEN	8		Variable Length	Num	8		
	VAR_XPT	9		Transport File	Char	1	(METXPT)	C (comment), D (domain), S (supp), O (operational)
	CODELIST	10		Controlled Terminology	Char	20		
	CDISC_ROLE	11		Role in CDISC Model	Char	50	(METROLE)	
	CDISC_CORE	12		Importance in CDISC Model	Char	10	(METCORE)	Exp (expected), Perm (permitted), Req (required)
	VAR_ORIGIN	13		Collection Source	Char	20		
	VAR_QEVAL	14		Evaluator	Char	50		
	DERIVATION	15		Derivation from Source Data	Char	2000		
	NOTES	16		Programmer Notes	Char	2000		
METADATA.LBMASTER	REF_ID	1		Record ID	Num	8		
	DICT_CAT	2		Standard Category	Char	50	(LBCAT)	
	DICT_SCAT	3		Standard Subcategory	Char	50	(LBSCAT)	
	DICT_TESTCD	4	1	Standard Test Code	Char	8	(LBTESTCD)	
	DICT_TEST	5		Standard Test Name	Char	40	(LBTEST)	

DATA_SET	VAR_NAME	VAR_ NUM	KEY_ VAR	VAR_LABEL	VAR_ TYPE	VAR_ LEN	CODELIST	NOTES
	DICT_SPEC	6	2	Test Specimen	Char	50	(LBSPEC)	
	DICT_METHOD	7	3	Test Method	Char	50	(LBMETHOD)	
	DICT_EVLINT	8	4	Test Evaluation Interval	Char	50	ISO 8601	
	DICT_SPID	9		ADaM Parameter Code	Char	8		
	DICT_PARAM	10		ADaM Parameter	Char	200		
	REF_TYPE	11		Test Result Data Type	Char	1	(METTYPE)	
	REF_STRESU	12		Preferred Standard Units	Char	50	(LBUNIT)	
	REF_STPREC	13		Precision of Standard Result	Num	8		
	REF_SIU	14		Preferred SI Units	Char	50	(LBUNIT)	
	REF_SIPREC	15		Precision of SI Result	Num	8		
	REF_LLOQ	16		Lower Limit of Quantification	Num	8		
	REF_ULOQ	17		Upper Limit of Quantification	Num	8		
	REF_CTLTERM	18		Char Result Controlled Terminology	Char	20		
	REF_STNRC	19		Reference Range for Char Result	Char	50		
METADATA.LBFACTORS	DICT_TESTCD	1	1	Test Code	Char	10		
	UNIT_ORIG	2	2	Original Result Unit	Char	50		
	UNIT_STD	3	3	Standard Result Unit	Char	50		
	CONV_FACTOR	4		Conversion Factor	Num	8		
METADATA.LBMAPPING	DSN	1	1	Source Dataset	Char	20		
	LBCAT	2	2	Verbatim Test Category	Char	50		
	LBTESTCD	3	3	Verbatim Test Subcategory	Char	8		
	LBTEST	4	4	Verbatim Test Name	Char	40		
	LBSPEC	5	5	Verbatim Test Specimen	Char	50		
	LBMETHOD	6	6	Verbatim Test Method	Char	50		
	REF_ID	7		Matching Dictionary Record	Num	8		(Foreign Key LBMASTER)

APPENDIX F: SAMPLE DATA QUALITY LISTING

This listing was generated using the SAS program in Appendix G. For interpretation and discussion, see page 14.

									SDTM LB Record Counts			
Obs	LBCAT	LBSCAT	LBTEST	LBTESTCD	LBSPEC	LBMETHOD	REF_ TYPE	LBSTRESU	CNT	N	NMISS	ND
1	CHEMISTRY	LIPIDS	CHOLESTEROL	CHOL			N	mg/dL	122	122	0	0
2	CHEMISTRY	LIPIDS	HDL CHOLESTEROL	HDL			N		6		6	6
3	CHEMISTRY	LIPIDS	HDL CHOLESTEROL	HDL			N	mg/dL	122	122	0	0
4	CHEMISTRY	LIPIDS	LDL CHOLESTEROL	LDL		CALCULATED	N	mg/dL	120	120	0	0
5	CHEMISTRY	LIPIDS	LDL CHOLESTEROL	LDL		MEASURED	N	mg/dL	2	2	0	0
6	CHEMISTRY	LIPIDS	TRIGLYCERIDES	TRIG			N		6	0	6	6
7	CHEMISTRY	LIPIDS	TRIGLYCERIDES	TRIG			N	mg/dL	122	120	2	0
8	HEMATOLOGY	HEMOGLOBIN	HEMOGLOBIN SICKLE CELL	HGBS	BLOOD		N	%	56	56	0	0
9	HEMATOLOGY	HEMOGLOBIN	HEMOGLOBIN SICKLE CELL	HGBS	BLOOD	QUALITATIVE	C		56	0	48	8
10	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR HEMOGLOBIN	MCH	BLOOD		N		16	16	0	0
11	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR HEMOGLOBIN	MCH	BLOOD		N	FL	10	10	0	0
12	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR HEMOGLOBIN	MCH	BLOOD		N	pg	96	96	0	0
13	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	MCHC	BLOOD		N		16	16	0	0
14	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR HGB CONCENTRATION	MCHC	BLOOD		N	g/dL	106	106	0	0
15	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR VOLUME	MCV	BLOOD		N		16	16	0	0
16	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR VOLUME	MCV	BLOOD		N	fL	96	96	0	0
17	HEMATOLOGY	ERYTHROCYTE INDICES	ERY. MEAN CORPUSCULAR VOLUME	MCV	BLOOD		N	pg	10	10	0	0

	Univariate Statistics											Unique Character Results	Record Counts in Source Datasets			
Obs	MEAN	STD	MIN	P5	P10	P25	P50	P75	P90	P95	MAX	STRESC	DSN_ LBXQ	DSN_ LAB	DSN_ LBX3	DSN_ LBX3H
1	126.381	27.1524	56	84	93	109	123	142	160	175	236		122			
2													6			
3	28.2425	9.25932	10	16	18	22	26.5	33	41	46	73		122			
4	70.7857	23.4426	20	34	43	55	67.5	86	103	111	144		120			
5	47	38.1838	20	20	20	20	47	74	74	74	74		2			
6													6			
7	140.149	67.3733	49	69	78	98.5	125.5	165	222	282	589		122			
8	0	0	0	0	0	0	0	0	0	0	0		56			
9												NEGATIVE; POSITIVE			48	8
10	28.525	3.96022	19	19	23	26.65	28.95	30.7	32.7	35.9	35.9			16		
11	85.5313	10.1915	62.6	62.6	71	78.2	87.85	92.95	98	100.8	100.8			10		
12	29.2738	4.45631	22.4	25	26.2	27.5	28.8	30.4	32.6	33.7	86			96		
13	71.6688	105.149	28.9	28.9	30.6	32.8	33.4	35	341	341	341			16		
14	33.209	2.77536	2.9	29.2	30.2	32	33.95	34.9	35.3	35.7	36.5			106		
15	85.5313	10.1915	62.6	62.6	71	78.2	87.85	92.95	98	100.8	100.8			16		
16	86.6282	5.33449	73	78.4	80	83.1	86.4	90.2	94.5	96	100.6			96		
17	28.525	3.96022	19	19	23	26.65	28.95	30.7	32.7	35.9	35.9			10		

APPENDIX G: DATA QUALITY LISTING

This program generates to listing shown in Appendix F. Just update the library names and let-statements. It's ready to run.

```
/*-------------------------------------------------------------------------------*/

   libname sdtmplus '..\..\..\data\sdtmplus' access = readonly ;
   libname here '.' ;

/*-------------------------------------------------------------------------------*/

   %let mysdtm   = sdtmplus.lb ;

                         /*-------------------------------------------------------*/
                         /* If the SDTM dataset includes a temporary variable     */
                         /* identifying the originating raw dataset for each SDTM */
                         /* record, it is informative to tabulate the source      */
                         /* record counts. This allows suspect summary results to */
                         /* be traced to their source.                            */
                         /*-------------------------------------------------------*/

   %let dsn = ;
   %let dsn = dsn ;

                         /*-------------------------------------------------------*/
                         /* If the SDTM dataset temporarily keeps the dictionary  */
                         /* variable with result datatype (C or N), it is helpful */
                         /* to include this in the data quality report.           */
                         /* -- Of course, a final SDTM dataset would have neither */
                         /*    raw dataset name nor datatype variables.           */
                         /*-------------------------------------------------------*/

   %let ref_type = ;
   %let ref_type = ref_type ;

                         /*-------------------------------------------------------*/
                         /* A typical variable set for summarization:             */
                         /* -- STUDYID  (keep when comparing multiple protocols)  */
                         /* -- LBCAT, LBSCAT (bundle together related tests)      */
                         /* -- LBTESTCD, LBTEST, LBSPEC, LBMETHOD (full test name) */
                         /*-------------------------------------------------------*/

   %let by_var = studyid lbcat lbscat lbtest lbtestcd lbspec lbmethod &ref_type ;
```

```
/*-----------------------------------------------------------------------------*/
/* For records with numeric result, derive a full set of univariate metrics.  */
   data work._data ;
      set &mysdtm (keep = &by_var lbstat lbstres: &dsn) ;
      nd = (lbstat eq 'NOT DONE') ;
   run ;

   proc summary data = work._data nway missing ;
      class &by_var lbstresu ;
      var lbstresn ;
      output out = work._lbdq (drop = _type_ rename = (_freq_ = CNT))
             n = N
             nmiss = NMISS
             sum(nd) = ND
             mean = MEAN
             std = STD
             min = MIN
             p5  = P5
             p10 = P10
             p25 = P25
             p50 = P50
             p75 = P75
             p90 = P90
             p95 = P95
             max = MAX ;
   run ;
/*-----------------------------------------------------------------------------*/
/* For records with non-numeric result, collect a list of unique values to check   */
/* for abbreviations, misspellings, or other inconsistencies.                      */
   proc sort data = work._data
             out  = work._lbstres nodupkey ;
      by &by_var lbstresu lbstresc ;
      where missing(lbstresn) and (lbstresc ne '') ;
   run ;

   data work._lbstres ;
      set work._lbstres ;
      by &by_var lbstresu lbstresc ;

      length STRESC $300 ;
      retain stresc ;

      if first.lbstresu then stresc = '' ;
      stresc = trim(stresc) || '; ' || strip(lbstresc) ;
      if last.lbstresu then do ;
         stresc = substr(stresc, 3) ;
         output ;
      end ;

      keep &by_var lbstresu stresc ;
   run ;

   data work._lbdq ;
      merge work._lbdq
            work._lbstres ;
      by &by_var lbstresu ;
   run ;
```

```
/*-----------------------------------------------------------------------------*/
/* Count source records, if possible.                                          */
   %macro dsn ;
      %if %str(&dsn) eq %str() %then %goto nodsn ;

      proc summary data = work._data nway missing ;
         class &by_var lbstresu &dsn ;
         output out = work._dsn (rename = (_freq_ = cnt)) ;
      run ;

      proc transpose data   = work._dsn
                     out    = work._transpose (drop = _name_)
                     prefix = DSN_ ;
         by &by_var lbstresu ;
         var cnt ;
         id &dsn ;
      run ;

      data work._lbdq ;
         merge work._lbdq work._transpose ;
         by &by_var lbstresu ;
      run ;

      %nodsn:
   %mend ;
   %dsn ;
/*-----------------------------------------------------------------------------*/
/* Save result as permanent dataset and as Excel workbook.                     */
   proc datasets library = work nolist ;
      modify _lbdq (label = "%upcase(&mysdtm) data quality report") ;
      run ;
      copy out = here ; select _lbdq ;
      run ;
      delete _: ;
   quit ;

   proc export data = here._lbdq
               outfile = ".\_lbdq.xls"
               dbms = excel
               replace ;
      sheet = "%upcase(&mysdtm) &sysdate9" ;
   run ;
/*-----------------------------------------------------------------------------*/
```