SDTM-ETL 5.0 User Manual and Tutorial
Tips for mapping the LB domain

Author: Jozef Aerts, XML4Pharma

Last update: 2025-03-01

Table of contents

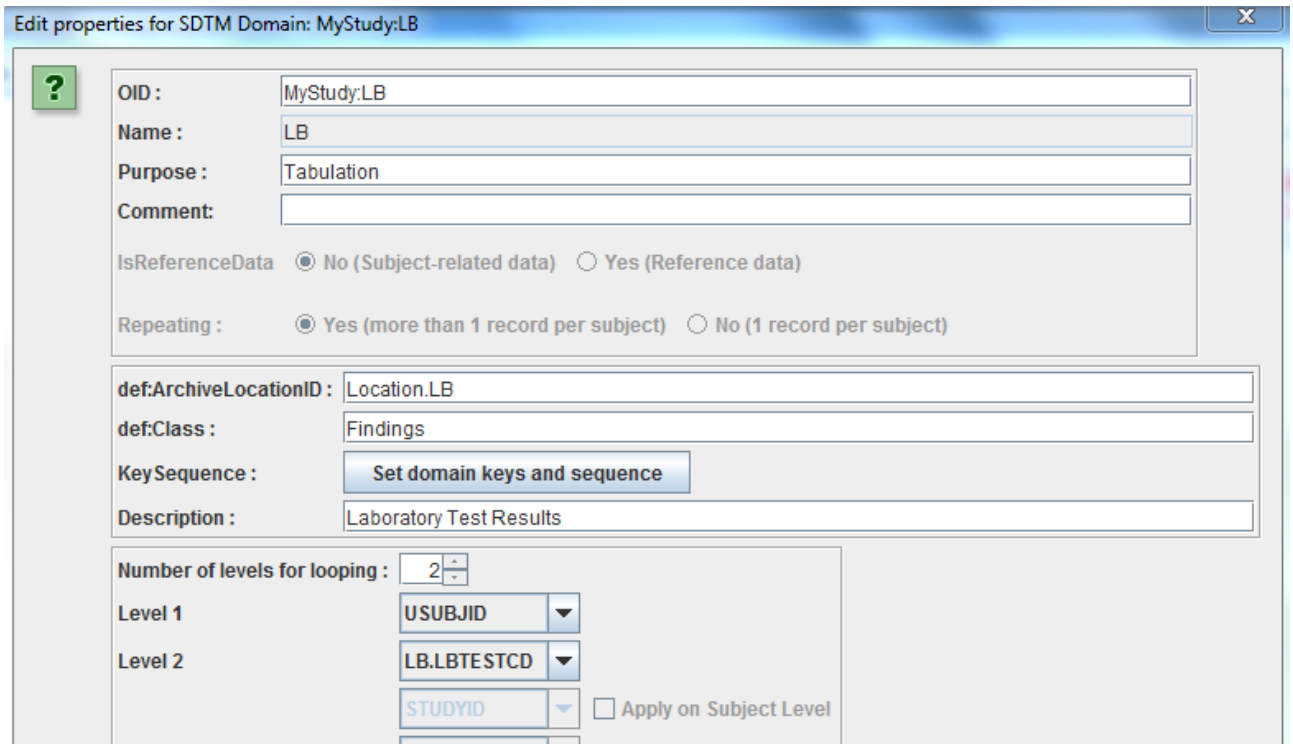
Introduction

Generating LB datasets is often one of the most difficult tasks, not only due to the large or even very large amount of data, but also because the lab data can have different providers (e.g. different labs), but also as lab data can appear in quite different forms and formats.

Instead of providing a regular tutorial, describing the mapping for one specific situation, which is probably not your situation, we choose to describe a number of worthful tips for generating LB mappings and datasets.

Tip 1: Setting the looping variable

Keep it simple: in 90% of the cases, using only 2 looping variables (USUBJID and LBTESTCD) will be sufficient. You can check this by double clicking the first cell of the „LB row“ or using the menu „Edit – SDTM domain properties":



Do not let confuse you by the SDTM-IG which states "One record per lab test per time point per visit per subject". That statement is about the result[1], not about how you come to it. If, after a drag-and-drop, you select the correct visits (using the "Generalize for" checkbox and the "Except for" and "Only for" buttons), forms and item groups, then you usually will automatically obtain the desired result. In some (but very few) cases you will want to add a third "looping variable" like "VISITNUM", but this is usually not necessary or even counterproductive.

For implementing "one record per …" in the define.xml, best is to assign "uniqueness keys" in the define.xml. This is explained in several other tutorials.

Tip 2: use several LB instances for different cases

Lab results can appear everywhere in a clinical study, and can have been generated by several labs.

This means that depending on the source (form, lab) you might need to map to a different set of LB variables. For example, you may have the case that a set of lab tests have been executed which have to do with pregnancy by lab A using form F1, and a set of blood lab tests by lab B collected on form F2.

In such a case, do not unnecessarily try to "push" everything in a single LB instance, but generate two instances of the LB domain, and generate mappings for the "pregnancy case" in one instance and mappings for the "hematology" case in another LB instance. Remember that the CDISC rules state that the dataset name may then not be longer than 4 characters, which is related to the rule that SUPP-- dataset names may not be longer than 8 characters[2]. So e.g. use "LBPR" and "LBHE" for the dataset names.
You can then use LBCAT and/or LBNAM to make clear why the data has been spread over 2 or more LB datasets.

---

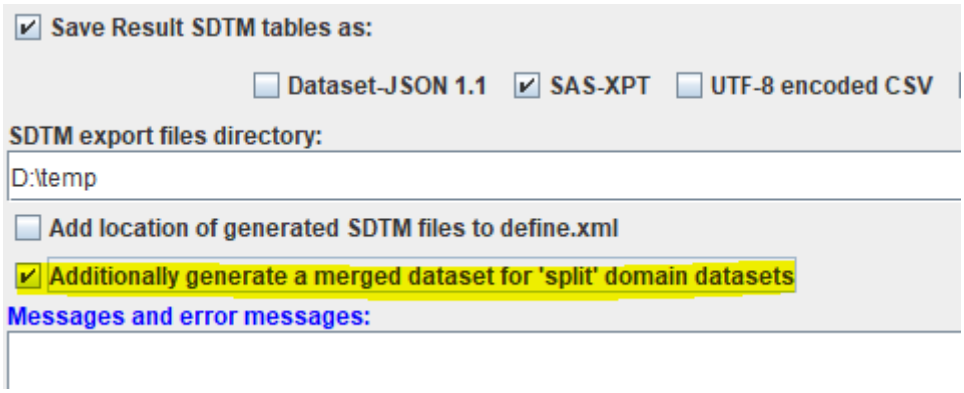
1 It doesn't state either what variable corresponds to a "time point"

2 This probably is related to SAS Transport 5. There is no other reasonable declaration ...

The SDTM-IGs and "Metadata Submission Guides" allow and even encourage such "splitting" (essentially the term in incorrect in our case, as we even never had a single dataset).

In case you see you need or want to add additional variables which do not apply for all lab data, you should definitely generate a separate LB instance for those data that you need the additional variables. By doing so, you also make it easier for the reviewer by explicitly showing the reviewer in which cases (or subset of data) you have the additional variable and in which cases you don't.

At the end, when you (also) want to obtain a single dataset for LB by merging the instances, use the checkbox "Additionally generate a merged dataset for 'split' domain datasets".



This will then take care that a single "lb.xpt" (or "lb.json" in the case of Dataset-JSON format) is generated by merging the separate datasets (e.g. lbpr.xpt and lbhe.xpt).
It will also take care that in the case of NSVs (Non-Standard Variables) a single supplb.xml (or supplb.json) file is generated, at least when the checkbox "Move non-standard SDTM variables to SUPP--" was checked.

Tip 3: reuse information

The LB table is a typical "hypervertical" table, following the "entity-attribute-value" model of databases. LBTESTCD is the "entity" LBORRES the "value", most of the others are "attributes". Depending on which test is represented in the current row, the values for attributes like LBSTNRLO (Reference Range Lower Limit) and LBSTNRHI (Reference Range Higher Limit) will very probably be different.

In SDTM-ETL, you can reuse variables values (but only in read-only mode) that you have defined before, i.e. more to the left in the same row. As LBTESTCD is pretty on the left, you can reuse its value in all variables that come after it (i.e. more to the right). Now suppose that you reference ranges are not the CRF, so you need to add them "manually". You can then e.g. have the following mapping script:

```
if($LB.LBTESTCD == 'GLUC') {
        $LB.LBSTNRLO = …;
} elsif ($LB.LBTESTCD == 'FRUCT') {
        $LB.LBSTNRLO =  ;
} …
```

As you have defined LBTESTCD before (more to the left) you do not need a drag-and-drop, you just can reuse the value from the mapping that you did before.

In case however that the reference range is also dependent on the method (LBMETHOD) you can not reuse the value from the LBMETHOD mapping as the latter comes after LBSTNRLO so you will probably need a drag-and-drop and create a temporary variable which has the same XPath expression as for the one you have for LBMETHOD itself.

For example:

```
$MYLBMETHOD = xpath(....);
if($LB.LBTESTCD == 'GLUC' and $MYLBMETHOD == '….' ) {
        $LB.LBSTNRLO = …;
} elsif($LB.LBTESTCD == 'GLUC' and $MYLBMETHOD == '….') {
        $LB.LBSTNRLO = …;
} elsif ($LB.LBTESTCD == 'FRUCT') {
        $LB.LBSTNRLO =  ;
} …
```

Essentially, the same applies for other defining variables such as LBSPEC.

Tip 4: Use the LOINC code when you can!

Unfortunately, CDISC still doesn't want to recognize that, due to the "not invented here syndrome", the LOINC code is a much better designator of the test. This is not only true for lab tests, but also e.g. for microbiology tests (MB domain), vital sign tests (VS) and even ECG tests (EG).
It is not a wonder at all that LOINC coding is used for all kinds of tests in Electronic Health Records (EHRs).
Also, FDA is now requiring the submission of the LOINC code (in LBLOINC) if you can get it from the lab provider[3]. Essentially, FDA should also mandate that for microbiology tests, especially for the case of COVID-19. There is however still a lot of resistance against that within CDISC.

IF you can get the LOINC codes from the lab[4], why not make this fact a blessing instead of a burden?

Essentially, as the LOINC code is a much better designator of the test than any combination of LBTESTCD, LBTEST, LBSPEC, LBMETHOD etc., it should not be necessary anymore to also submit values for the latter when one has populated the LOINC code in LBLOINC. However, CDISC thinks differently ...
Developing mappings for LBTESTCD, LBTEST, LBSPEC, LBMETHOD etc. is always a laborious task, with many pitfalls. For the mapper, it requires a very good knowledge about how lab tests and results, as essentially, the mapping is essentially a categorization step.
When one has the LOINC code however, this can be completely automated in a very reliable way.

When the FDA requirement for the LOINC code came, there was a lot of panic in the clinical research community. As a result, CDISC developed a mapping between the 1,400 most popular lab test LOINC codes, and the LB-SDTM variables. It was published as ... an Excel worksheet.

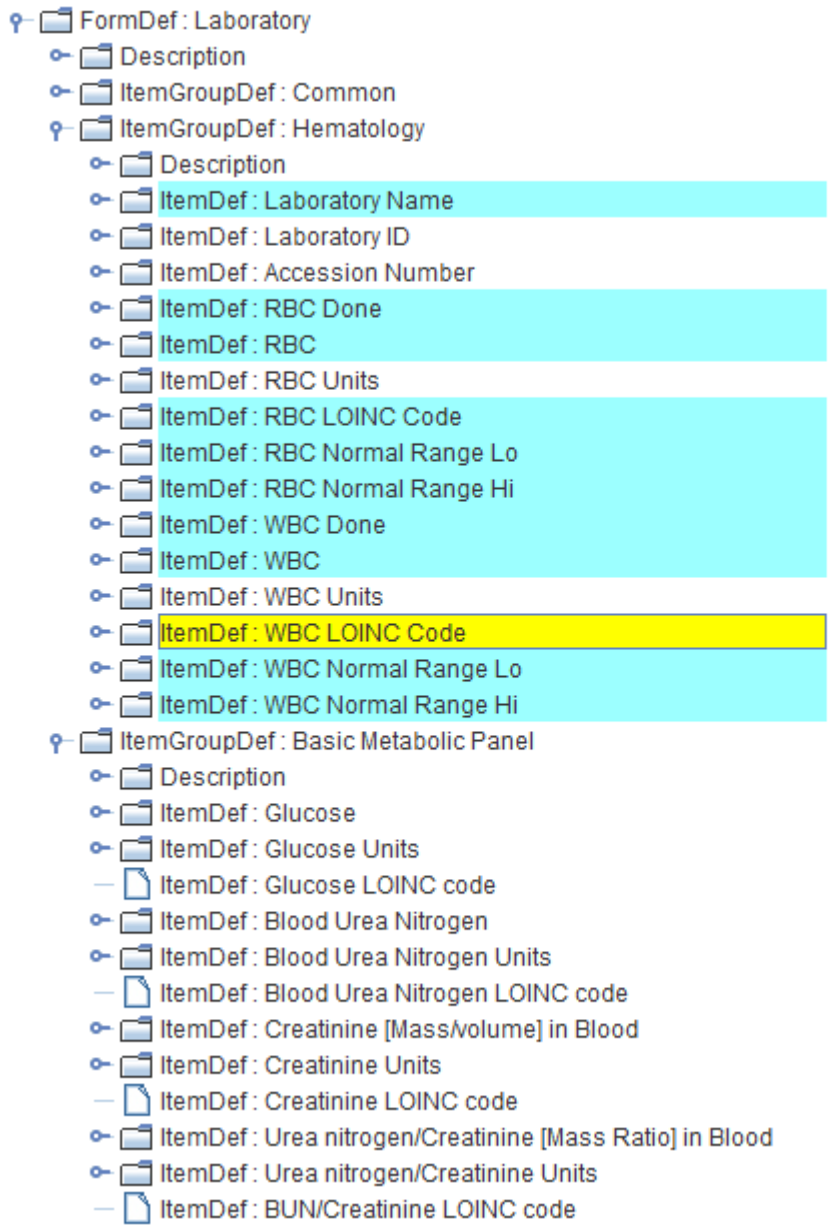
---

[3] One should however not try to deduce the LOINC code from the information obtained from the lab.
[4] If your lab provider cannot provide LOINC codes with the results, you should re-negociate that, or simply, kick them out ...

We soon found out that this (limited) mapping was far from sufficient for "real life" work, especially when EHRs are used as a source. Therefore, we extended the mapping to almost 10,000 LOINC codes, with a total of over 18,000 mappings, and installed a RESTful web service to enable applications to use them. The RESTful web service is described at:
http://xml4pharmaserver.com/WebServices/LOINC2CDISC_webservices.html

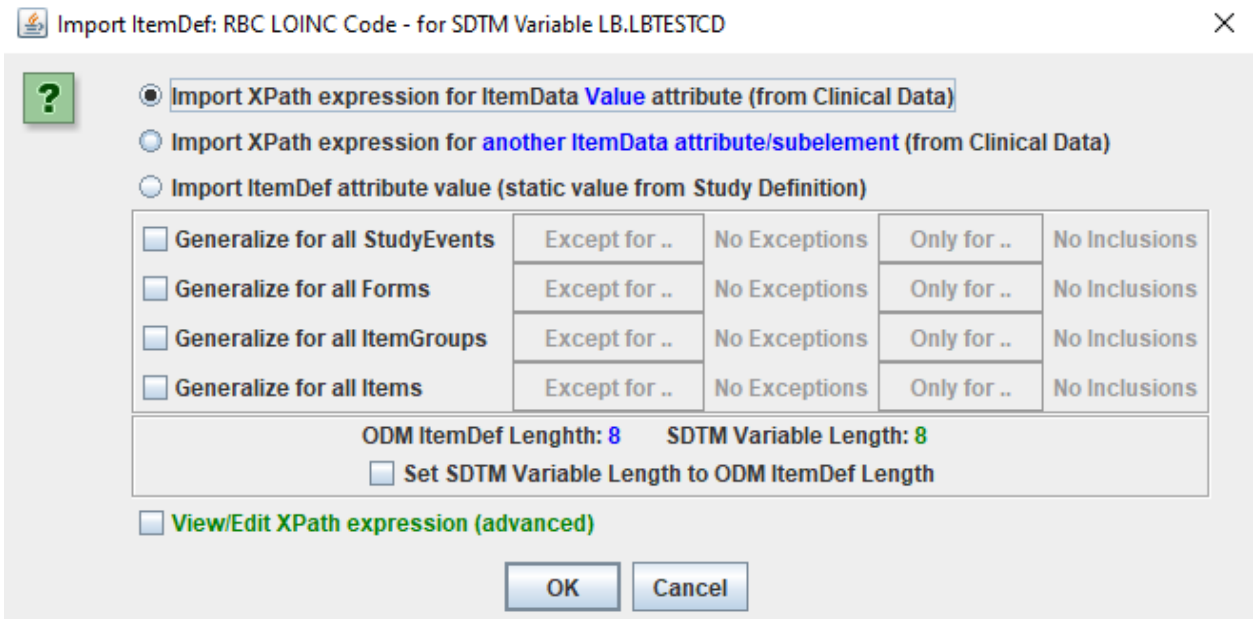
SDTM-ETL uses this RESTful web service to automate the assignment of values for LBTESTCD, LBTEST, LBSPEC, LBMETHOD, etc. in a very reliable way.

Here is an example where we indeed do have the LOINC code as a field in the CRF:

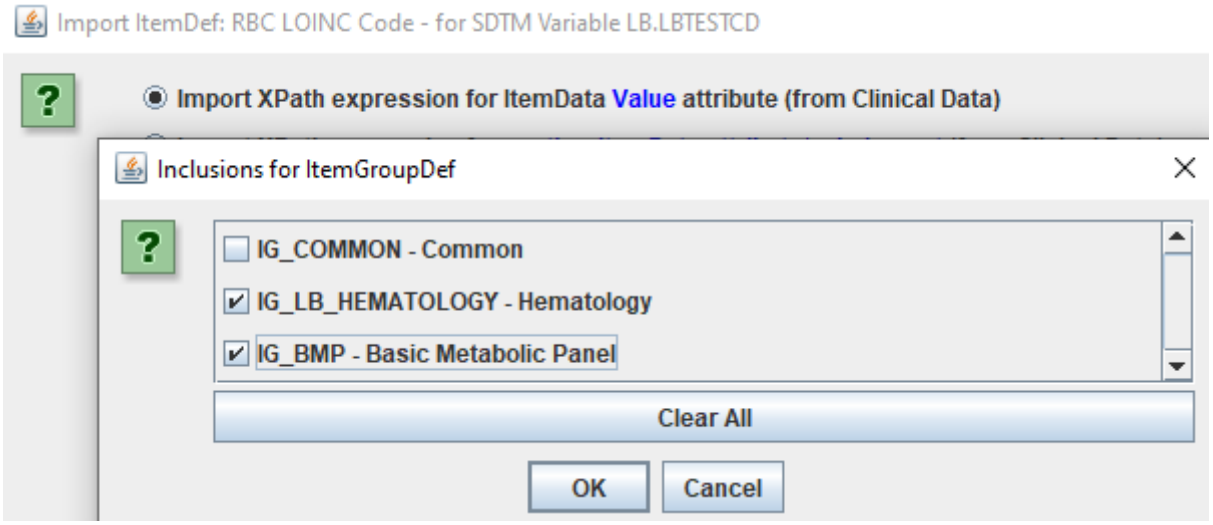


We see that for each test, besides the test result and the units, also the LOINC code of the test is provided. This should nowadays be the standard.

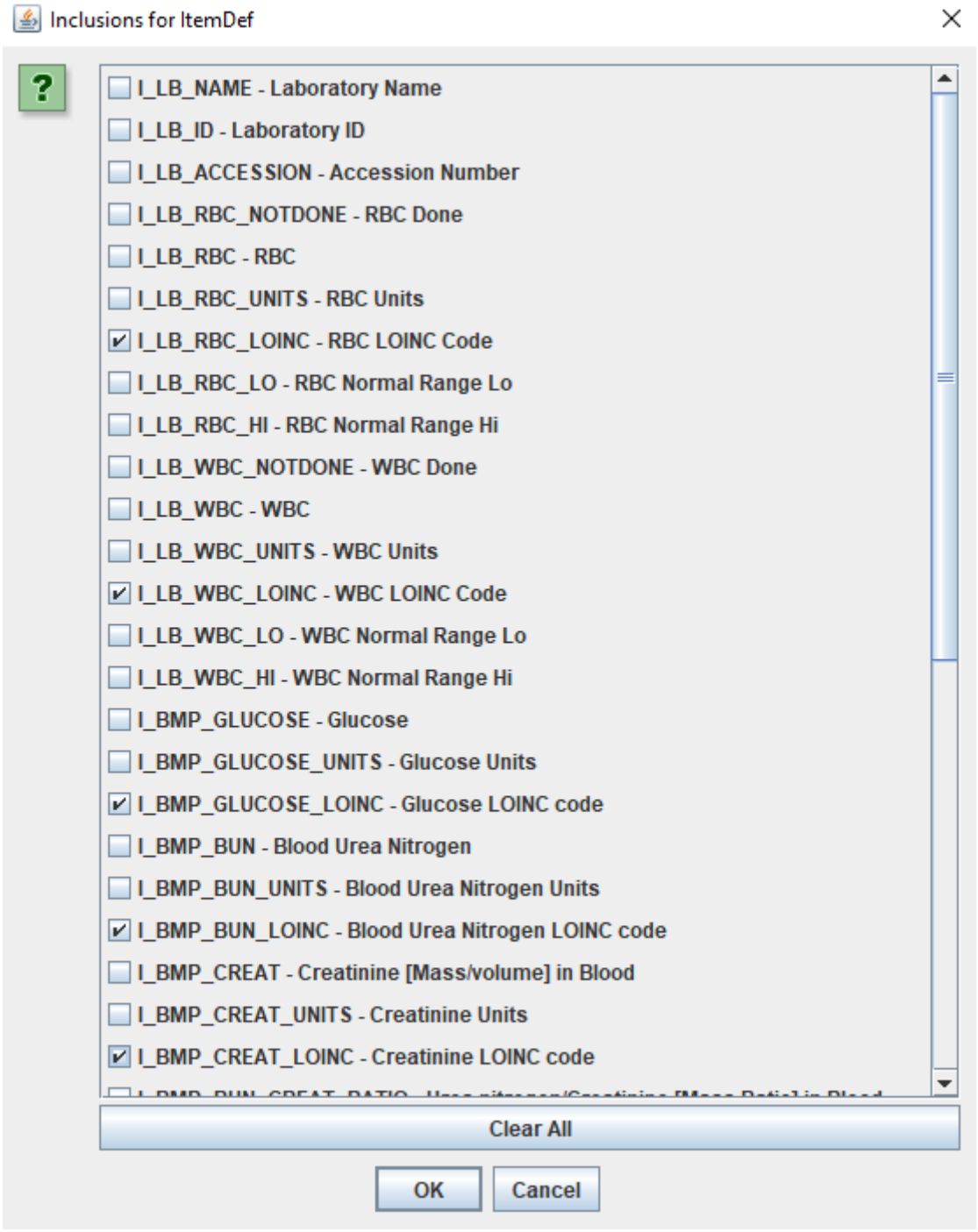
Instead of now basing our mapping on the OIDs (identifiers) of the ODM items, we will base it on the LOINC code. After loading a template define.xml (or use a define.xml containing prior mappings for other domains) we first drag-and-drop one of the "LOINC" items (e.g. "RBC LOINC code) to LBTESTCD. The usual dialog is displayed:



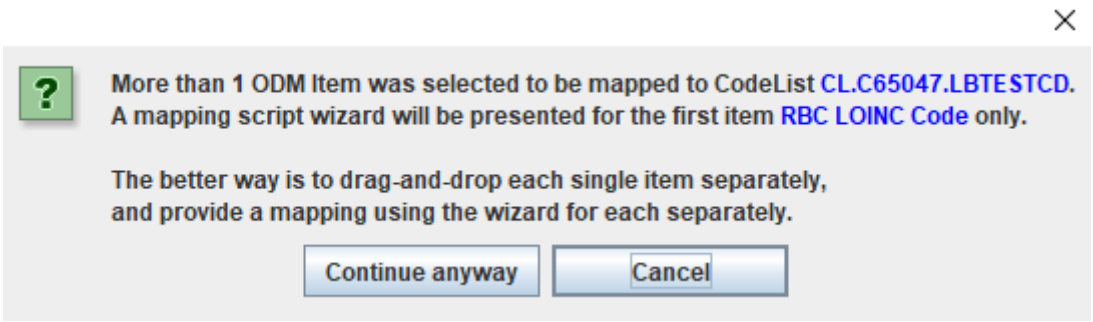
Important here is that we state that we want to use the value of the data point, not the identifier. Also, we want to obtain the lab data for any visit, so we check the checkbox "Generalize for all StudyEvents". As the lab tests come in two groups, one being "hematology" and one being "Basic metabolic panel", we also check "Generalize for all ItemGroups", and then using "only for", select the 2 groups:



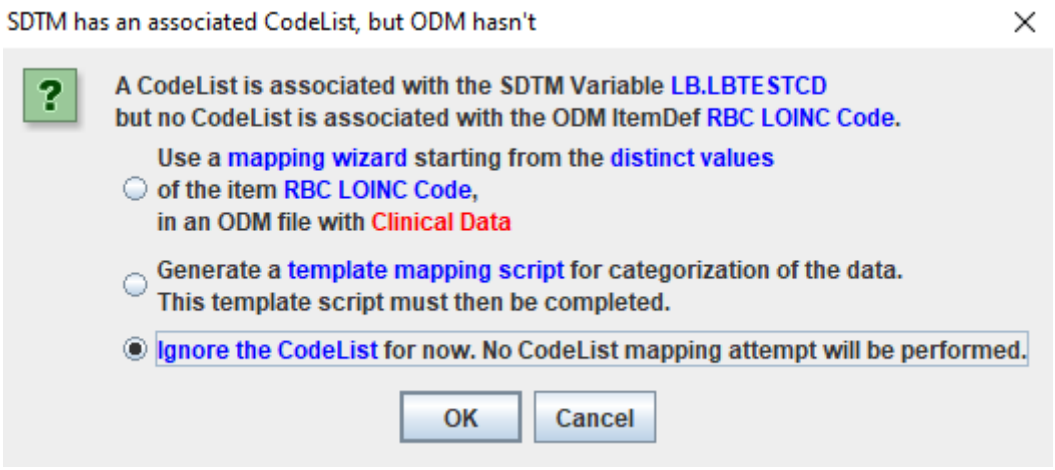
These two groups however also contain items that do not represent a LOINC code, so we also check "Generalize for all Items", and then select the ones for a LOINC code, using the "Only for" button:



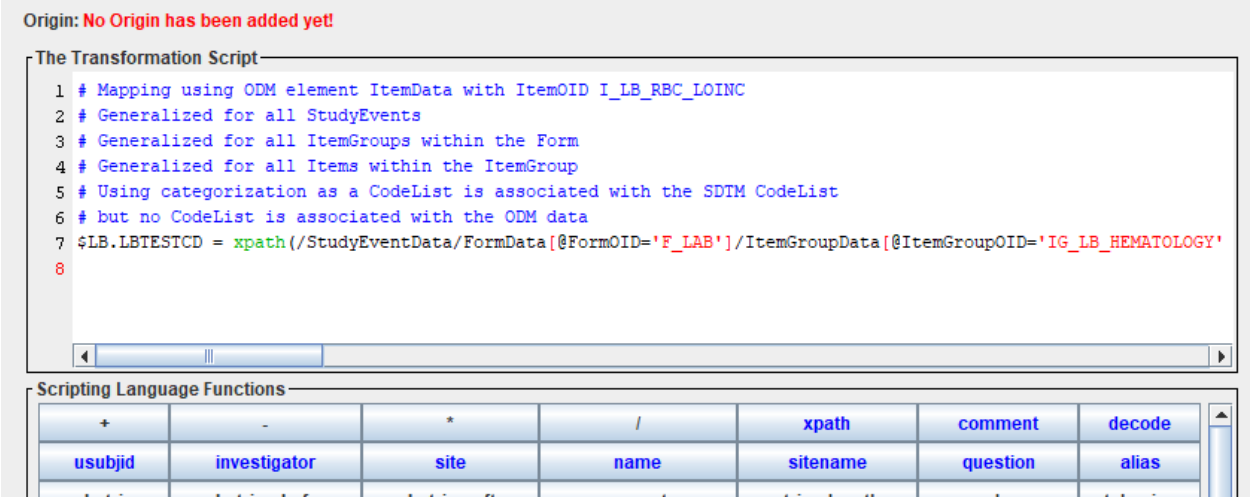
After clicking "OK" twice, we get:



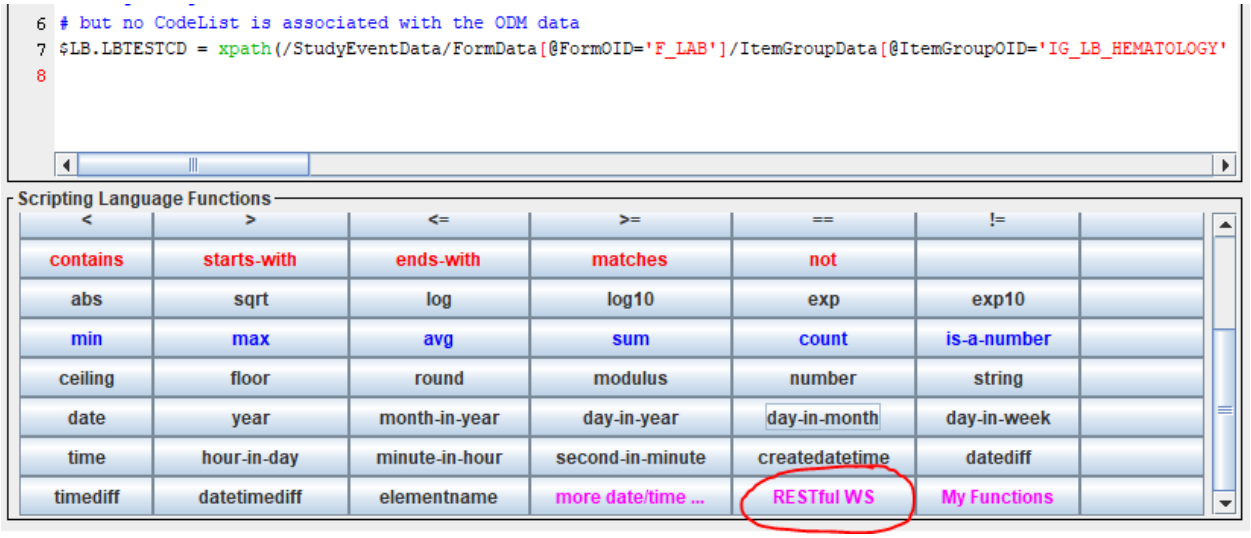
We click "Continue anyway", leading to:



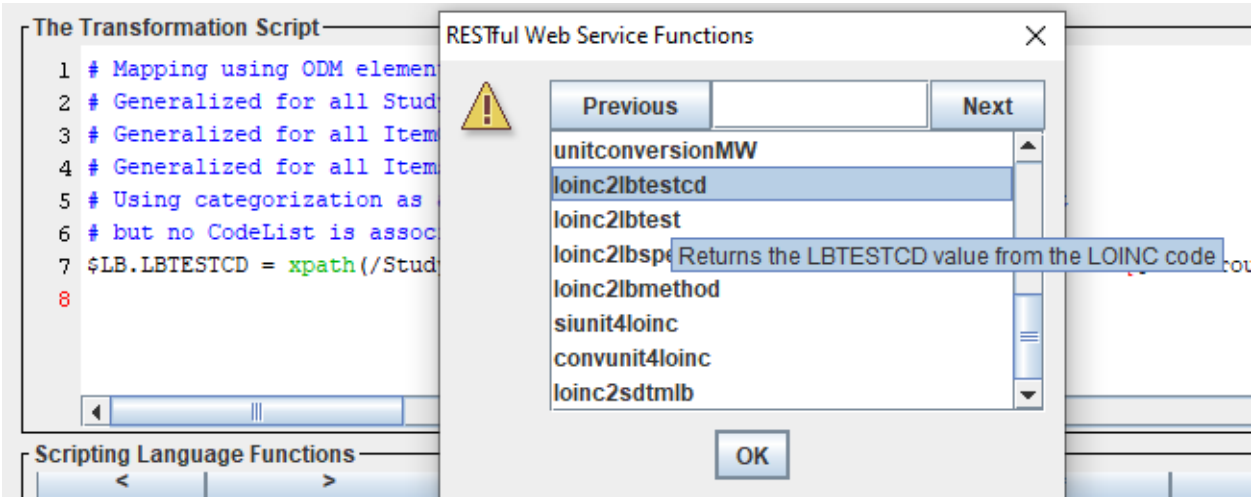
selecting "Ignore the CodeList for now ..." as we want to let the RESTful web service do the work for us. After clicking "OK", the mapping editor is shown:



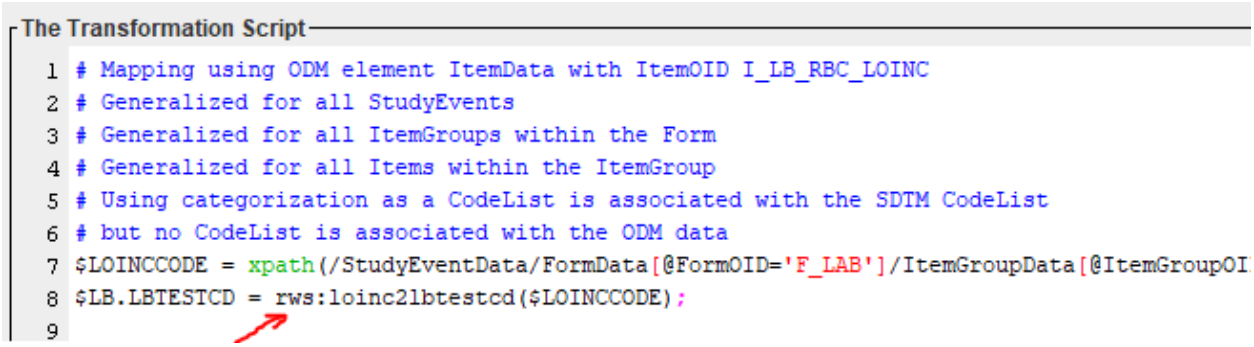
We now look for the button "RESTful WS":



and if we click it, we get all the SDTM-ETL functions that use RESTful web services:

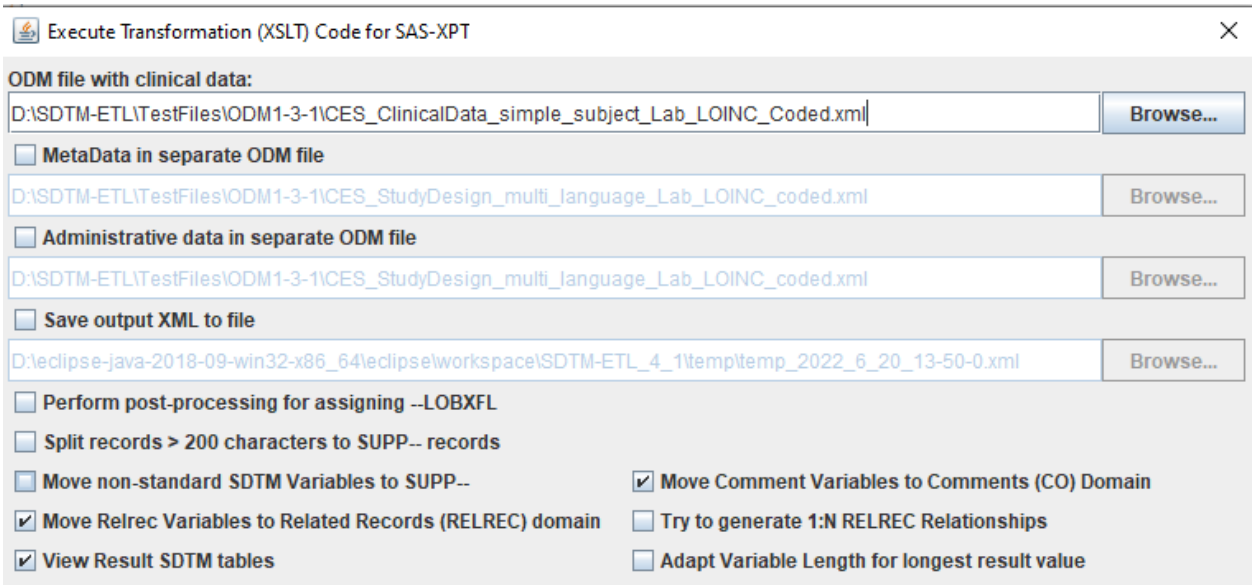


One interesting function seems to be "loinc2lbtestcd", which takes a LOINC code and returns the value for the associated LBTESTCD value. So, let's use it!
After changing the script slightly, we have:

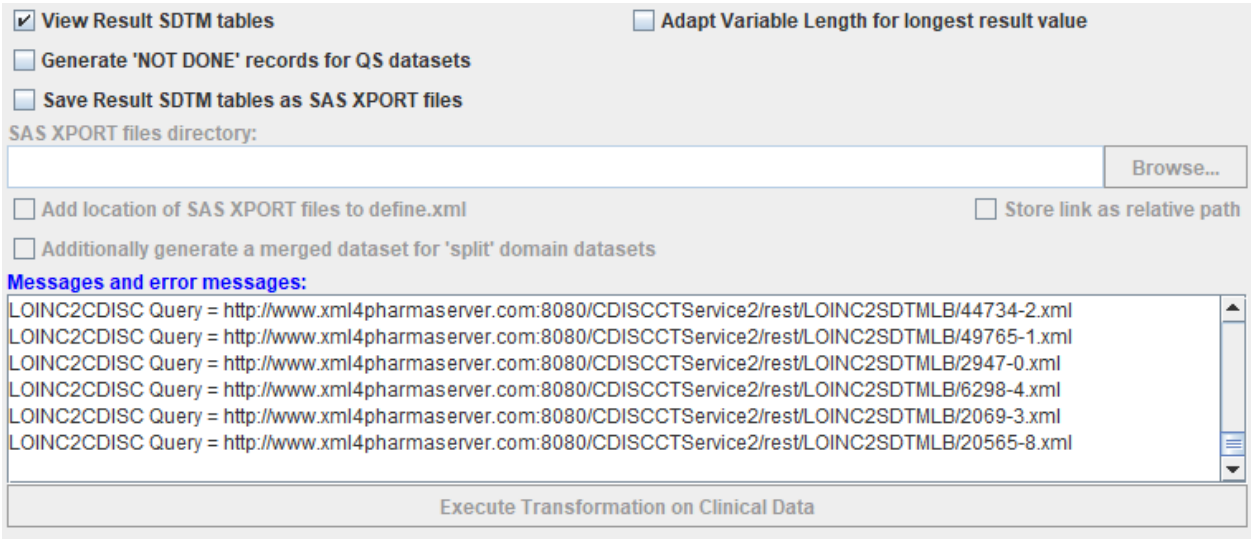


Then click "OK" to confirm the mapping script generation.
Before testing this mapping, let us first also populate LBLOINC itself. This can simply be done by repeating the "drag-and-drop", but then to the LB.LBLOINC cell. The generated script is just one line.

Let us now test on real clinical data, using "Transform - Generate Transformation (XSLT) Code for SAS-XPT". We then need to select an ODM file with clinical data. For example:

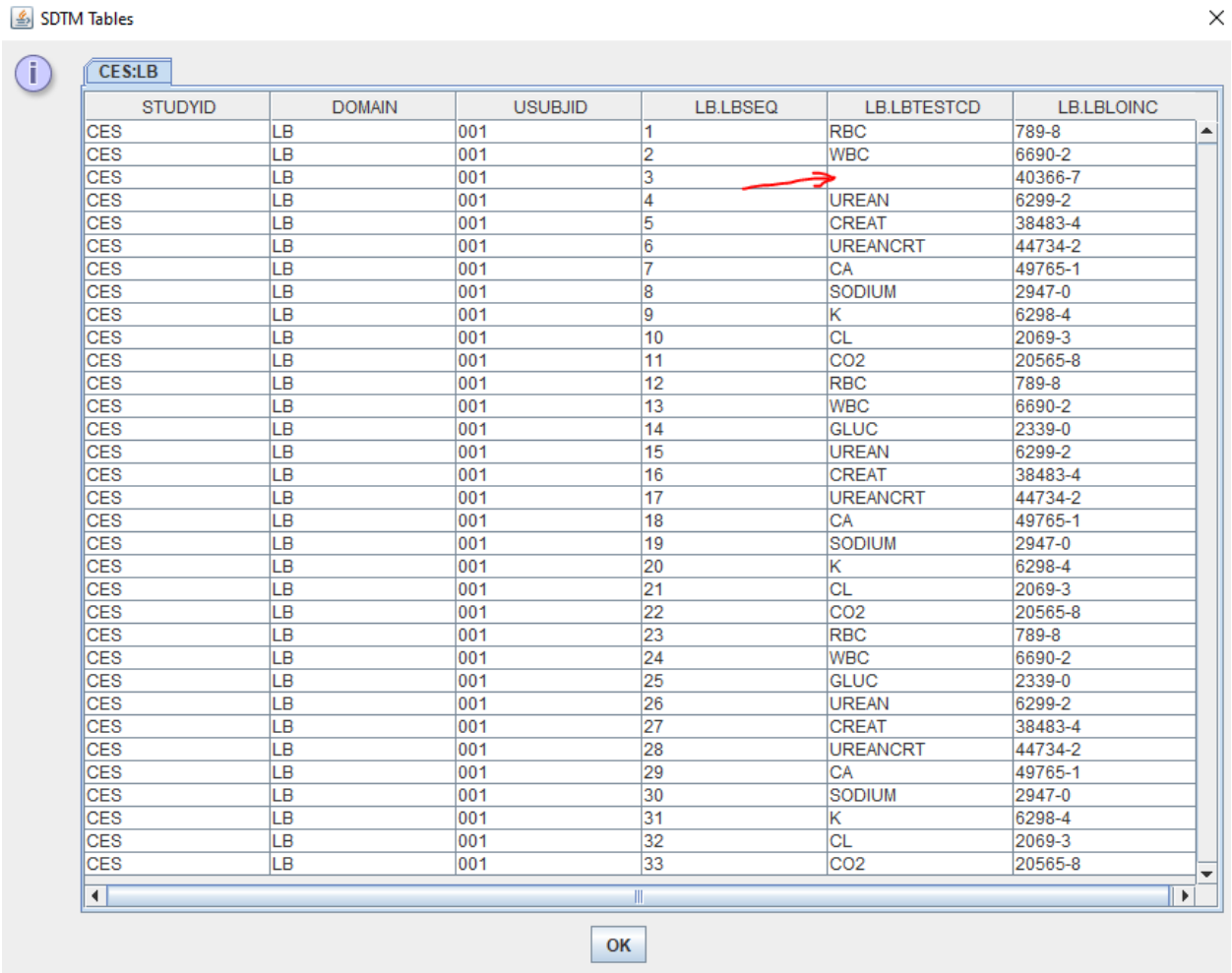


and when hitting the "Execute Transformation on Clinical Data", we see some messages passing by:



showing us that the RESTful web service is indeed called[5].

The result is:



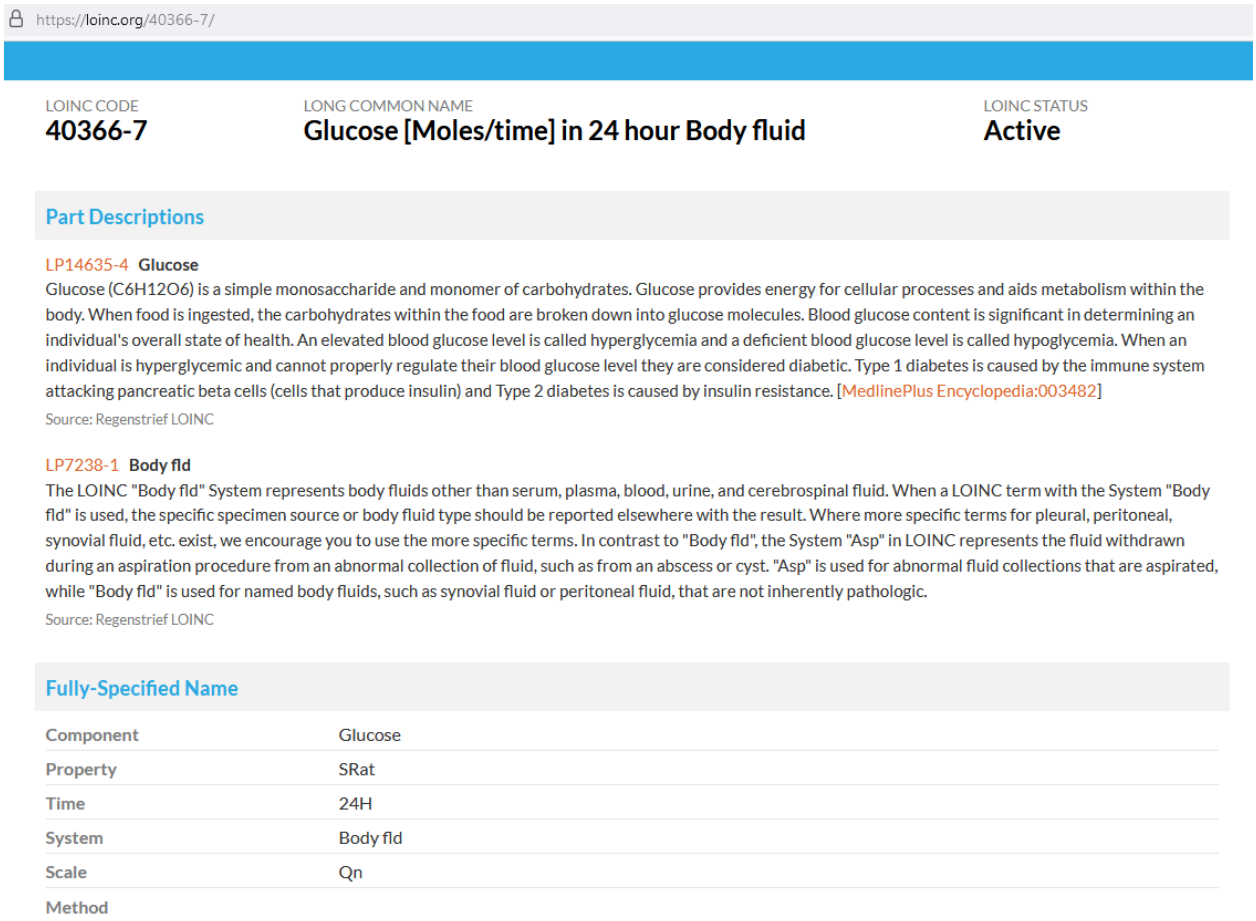
STUDYID	DOMAIN	USUBJID	LB.LBSEQ	LB.LBTESTCD	LB.LBLOINC
CES	LB	001	1	RBC	789-8
CES	LB	001	2	WBC	6690-2
CES	LB	001	3		40366-7
CES	LB	001	4	UREAN	6299-2
CES	LB	001	5	CREAT	38483-4
CES	LB	001	6	UREANCRT	44734-2
CES	LB	001	7	CA	49765-1
CES	LB	001	8	SODIUM	2947-0
CES	LB	001	9	K	6298-4
CES	LB	001	10	CL	2069-3
CES	LB	001	11	CO2	20565-8
CES	LB	001	12	RBC	789-8
CES	LB	001	13	WBC	6690-2
CES	LB	001	14	GLUC	2339-0
CES	LB	001	15	UREAN	6299-2
CES	LB	001	16	CREAT	38483-4
CES	LB	001	17	UREANCRT	44734-2
CES	LB	001	18	CA	49765-1
CES	LB	001	19	SODIUM	2947-0
CES	LB	001	20	K	6298-4
CES	LB	001	21	CL	2069-3
CES	LB	001	22	CO2	20565-8
CES	LB	001	23	RBC	789-8
CES	LB	001	24	WBC	6690-2
CES	LB	001	25	GLUC	2339-0
CES	LB	001	26	UREAN	6299-2
CES	LB	001	27	CREAT	38483-4
CES	LB	001	28	UREANCRT	44734-2
CES	LB	001	29	CA	49765-1
CES	LB	001	30	SODIUM	2947-0
CES	LB	001	31	K	6298-4
CES	LB	001	32	CL	2069-3
CES	LB	001	33	CO2	20565-8

We see that LBTESTCD has been fully automatically populated, except for the LOINC code

[5] The RESTful web service is very fast, returning several hundreds of responses per second.

40366-7, for which there seems to be no mapping in the database[6]. This will seldomly happen, as there are mappings for almost 10,000 LOINC codes[7].

We can however easily find out what the LOINC code 40366-7 means by visiting the LOINC website:



So that we immediately see that we can map this to LBTESTCD=GLUC, LBTEST=Glucose, LBMETHOD being empty. We will however also need to look for a CDISC code in LBSPEC for "body fluid".

So, we slightly change the mapping script into:

[6] In this case, we did this of course on purpose. If you encounter such a case that a mapping is missing, please let us know and we will try to add it to the database used by the RESTful-WS.
[7] There are over 100,000 LOINC codes (version 2.80)

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID I_LB_RBC_LOINC - value from attribute ItemOID
# Generalized for all StudyEvents
# Generalized for all Items within the ItemGroup
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$LOINCCODE =  xpath(/StudyEventData/FormData[@FormOID='F_LAB']/ItemGroupData[@ItemGroupOID='IG_LB_
# LOINC code 40366-7 is not covered by the LOINC to CDISC mapping
# LOINC code 40366-7 is for 'Glucose [Moles/time] in 24 hour Body fluid'
if($LOINCCODE = '40366-7') {
     $TEMP = 'GLUC';
} else {
     $TEMP = rws:loinc2lbtestcd($LOINCCODE);
}
$LB.LBTESTCD = $TEMP;
```

After re-running the transformation, the result is:

SDTM Tables

CES:LB

STUDYID	DOMAIN	USUBJID	LB.LBTESTCD	LB.LBLOINC
CES	LB	001	RBC	789-8
CES	LB	001	WBC	6690-2
CES	LB	001	GLUC	40366-7
CES	LB	001	UREAN	6299-2
CES	LB	001	CREAT	38483-4
CES	LB	001	UREANCRT	44734-2
CES	LB	001	CA	49765-1

We can now do something very similar for LBTEST, but this time not using the function "loinc2lbtestcd", but using the function "rws:loinc2lbtest":

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID I_LB_RBC_LOINC - value from attribute ItemC
# Generalized for all StudyEvents
# Generalized for all Items within the ItemGroup
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$LOINCCODE =  xpath(/StudyEventData/FormData[@FormOID='F_LAB']/ItemGroupData[@ItemGroupOID='I
# LOINC code 403566-7 is not covered by the LOINC to CDISC mapping
if($LOINCCODE = '403566-7') {
     $LB.LBTEST = 'Glucose';
} else {
     $LB.LBTEST = loinc2lbtest($LOINCCODE);
}

```

leading to:

The Transformation Script

```
1  # Mapping using ODM element ItemData with ItemOID I_LB_RBC_LOINC - value from attribut
2  # Generalized for all StudyEvents
3  # Generalized for all Items within the ItemGroup
4  # Using categorization as a CodeList is associated with the SDTM CodeList
5  # but no CodeList is associated with the ODM data
6  $LOINCCODE =  xpath(/StudyEventData/FormData[@FormOID='F_LAB']/ItemGroupData[@ItemGrou
7  # LOINC code 40366-7 is not covered by the LOINC to CDISC mapping
8  if($LOINCCODE = '40366-7') {
9       $LB.LBTEST = 'Glucose';
10 } else {
11      $LB.LBTEST = rws:loinc2lbtest($LOINCCODE);
12 }
13
```

and the result:

SDTM Tables

CES:LB

STUDYID	DOMAIN	USUBJID	LB.LBTESTCD	LB.LBTEST	LB.LBLOINC
CES	LB	001	RBC	Erythrocytes	789-8
CES	LB	001	WBC	Leukocytes	6690-2
CES	LB	001	GLUC	Glucose	40366-7
CES	LB	001	UREAN	Urea Nitrogen	6299-2
CES	LB	001	CREAT	Creatinine	38483-4
CES	LB	001	UREANCRT	Urea Nitrogen/Creatinine	44734-2
CES	LB	001	CA	Calcium	49765-1
CES	LB	001	SODIUM	Sodium	2947-0
CES	LB	001	K	Potassium	6298-4
CES	LB	001	CL	Chloride	2069-3
CES	LB	001	CO2	Carbon Dioxide	20565-8
CES	LB	001	RBC	Erythrocytes	789-8

We can then do similarly for LBSPEC (using the method rws:loinc2lbspec), and LBMETHOD (using the method rws:loinc2lbmethod).

For LBSPEC, we still need a code for "body fluid". We can try to find one using the menu "View - SDTM associated codelist":

CodeList: CL.C77529.SPEC

FLUID [C13236]
FLUID, ABDOMINAL [C77611]
FLUID, AMNIOTIC [C13188]
FLUID, BRONCHOALVEOLAR LAVAGE [C13195]
FLUID, CEREBROSPINAL [C12692]
FLUID, PERICARDIAL [C3319]
FLUID, PERITONEAL [C77612]
FLUID, PLEURAL [C77613]
FLUID, SYNOVIAL [C33718]
FLUID, THORAX [C125897]

OK

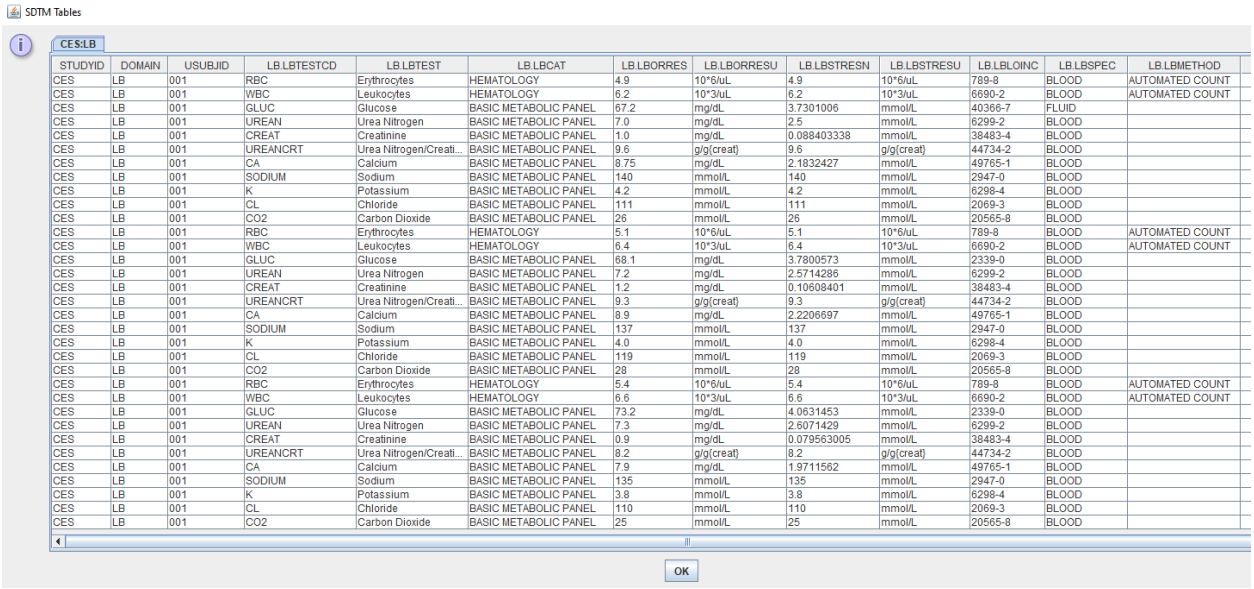
The best hit still seems to be "FLUID" (CDISC-NCI code C13236).
So we will use the following mapping script for LBSPEC:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID I_LB_RBC_LOINC - value from attr
# Generalized for all StudyEvents
# Generalized for all Items within the ItemGroup
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$LOINCCODE =  xpath(/StudyEventData/FormData[@FormOID='F_LAB']/ItemGroupData[@Item
# LOINC code 40366-7 is not covered by the LOINC to CDISC mapping
if($LOINCCODE = '40366-7') {
      $LB.LBSPEC = 'FLUID';
} else {
      $LB.LBSPEC = rws:loinc2lbspec($LOINCCODE);
}
```

Once we have the LOINC code and mappings to at least LBTESTCD and LBTEST, we can also start populating variables such as LBORRES and LBORRESU. As we do already have LBTESTCD. we can now choose whether to base our selection for the value either on the value of LBTESTCD, or on basis of the value of the LOINC code. The first can however be tricky when we have more than one type of test for the same analyte, e.g. "glucose in blood" and "glucose in urine". In our case, we have as well "glucose in blood" (LOINC code 2339-0) as well as "glucose in body fluid" (LOINC code 40366-7), so we must be careful, and better base the selection of which data point to take based on the LOINC code.

The final result looks like:


SDTM Tables

CES:LB

STUDYID	DOMAIN	USUBJID	LB.LBTESTCD	LB.LBTEST	LB.LBCAT	LB.LBORRES	LB.LBORRESU	LB.LBSTRESN	LB.LBSTRESU	LB.LBLOINC	LB.LBSPEC	LB.LBMETHOD
CES	LB	001	RBC	Erythrocytes	HEMATOLOGY	4.9	10*6/uL	4.9	10*6/uL	789-8	BLOOD	AUTOMATED COUNT
CES	LB	001	WBC	Leukocytes	HEMATOLOGY	6.2	10*3/uL	6.2	10*3/uL	6690-2	BLOOD	AUTOMATED COUNT
CES	LB	001	GLUC	Glucose	BASIC METABOLIC PANEL	67.2	mg/dL	3.7301006	mmol/L	40366-7	FLUID	
CES	LB	001	UREAN	Urea Nitrogen	BASIC METABOLIC PANEL	7.0	mg/dL	2.5	mmol/L	6299-2	BLOOD	
CES	LB	001	CREAT	Creatinine	BASIC METABOLIC PANEL	1.0	mg/dL	0.088403338	mmol/L	38483-4	BLOOD	
CES	LB	001	UREANCRT	Urea Nitrogen/Creati...	BASIC METABOLIC PANEL	9.6	g/g{creat}	9.6	g/g{creat}	44734-2	BLOOD	
CES	LB	001	CA	Calcium	BASIC METABOLIC PANEL	8.75	mg/dL	2.1832427	mmol/L	49765-1	BLOOD	
CES	LB	001	SODIUM	Sodium	BASIC METABOLIC PANEL	140	mmol/L	140	mmol/L	2947-0	BLOOD	
CES	LB	001	K	Potassium	BASIC METABOLIC PANEL	4.2	mmol/L	4.2	mmol/L	6298-4	BLOOD	
CES	LB	001	CL	Chloride	BASIC METABOLIC PANEL	111	mmol/L	111	mmol/L	2069-3	BLOOD	
CES	LB	001	CO2	Carbon Dioxide	BASIC METABOLIC PANEL	26	mmol/L	26	mmol/L	20565-8	BLOOD	
CES	LB	001	RBC	Erythrocytes	HEMATOLOGY	5.1	10*6/uL	5.1	10*6/uL	789-8	BLOOD	AUTOMATED COUNT
CES	LB	001	WBC	Leukocytes	HEMATOLOGY	6.4	10*3/uL	6.4	10*3/uL	6690-2	BLOOD	AUTOMATED COUNT
CES	LB	001	GLUC	Glucose	BASIC METABOLIC PANEL	68.1	mg/dL	3.7800573	mmol/L	2339-0	BLOOD	
CES	LB	001	UREAN	Urea Nitrogen	BASIC METABOLIC PANEL	7.2	mg/dL	2.5714286	mmol/L	6299-2	BLOOD	
CES	LB	001	CREAT	Creatinine	BASIC METABOLIC PANEL	1.2	mg/dL	0.10608401	mmol/L	38483-4	BLOOD	
CES	LB	001	UREANCRT	Urea Nitrogen/Creati...	BASIC METABOLIC PANEL	9.3	g/g{creat}	9.3	g/g{creat}	44734-2	BLOOD	
CES	LB	001	CA	Calcium	BASIC METABOLIC PANEL	8.9	mg/dL	2.2206697	mmol/L	49765-1	BLOOD	
CES	LB	001	SODIUM	Sodium	BASIC METABOLIC PANEL	137	mmol/L	137	mmol/L	2947-0	BLOOD	
CES	LB	001	K	Potassium	BASIC METABOLIC PANEL	4.0	mmol/L	4.0	mmol/L	6298-4	BLOOD	
CES	LB	001	CL	Chloride	BASIC METABOLIC PANEL	119	mmol/L	119	mmol/L	2069-3	BLOOD	
CES	LB	001	CO2	Carbon Dioxide	BASIC METABOLIC PANEL	28	mmol/L	28	mmol/L	20565-8	BLOOD	
CES	LB	001	RBC	Erythrocytes	HEMATOLOGY	5.4	10*6/uL	5.4	10*6/uL	789-8	BLOOD	AUTOMATED COUNT
CES	LB	001	WBC	Leukocytes	HEMATOLOGY	6.6	10*3/uL	6.6	10*3/uL	6690-2	BLOOD	AUTOMATED COUNT
CES	LB	001	GLUC	Glucose	BASIC METABOLIC PANEL	73.2	mg/dL	4.0631453	mmol/L	2339-0	BLOOD	
CES	LB	001	UREAN	Urea Nitrogen	BASIC METABOLIC PANEL	7.3	mg/dL	2.6071429	mmol/L	6299-2	BLOOD	
CES	LB	001	CREAT	Creatinine	BASIC METABOLIC PANEL	0.9	mg/dL	0.079563005	mmol/L	38483-4	BLOOD	
CES	LB	001	UREANCRT	Urea Nitrogen/Creati...	BASIC METABOLIC PANEL	8.2	g/g{creat}	8.2	g/g{creat}	44734-2	BLOOD	
CES	LB	001	CA	Calcium	BASIC METABOLIC PANEL	7.9	mg/dL	1.9711562	mmol/L	49765-1	BLOOD	
CES	LB	001	SODIUM	Sodium	BASIC METABOLIC PANEL	135	mmol/L	135	mmol/L	2947-0	BLOOD	
CES	LB	001	K	Potassium	BASIC METABOLIC PANEL	3.8	mmol/L	3.8	mmol/L	6298-4	BLOOD	
CES	LB	001	CL	Chloride	BASIC METABOLIC PANEL	110	mmol/L	110	mmol/L	2069-3	BLOOD	
CES	LB	001	CO2	Carbon Dioxide	BASIC METABOLIC PANEL	25	mmol/L	25	mmol/L	20565-8	BLOOD	

OK

Conclusion: using the LOINC code as the driver for the mapping has enormous advantages: in most cases, all necessary mappings can be developed within an hour, whereas using the classic approach, looking at the (local) codes and result data provided by the lab, this usually takes days, and is error prone.

Further methods for using LOINC-CDISC mappings are explained in the tutorial "Using the LOINC-SDTM-LB Mapping and similar functions" including storing mappings to a local file for reuse, which in some cases can be faster than when using the RESTful web services.