SDTM-ETL 5.0 User Manual and Tutorial

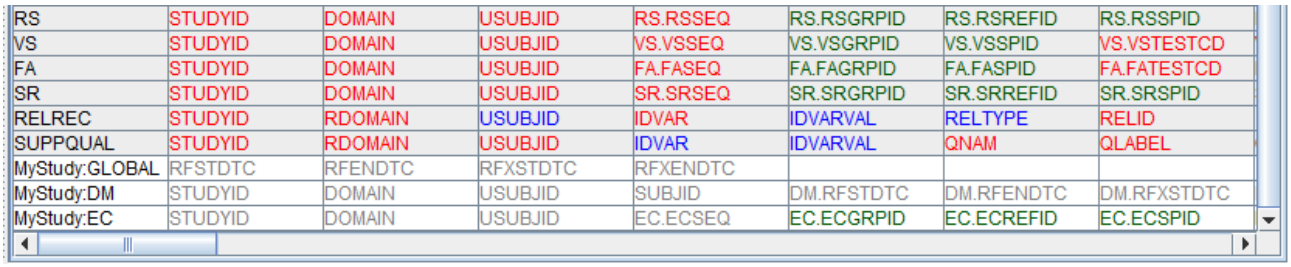
Author: Jozef Aerts, XML4Pharma

Last update: 2025-02-12

Adding mappings for the Vital Signs domain

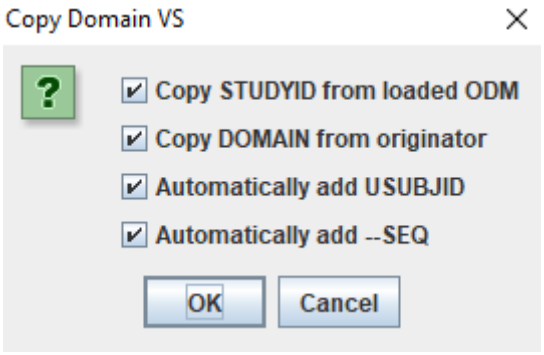
After loading and validating the ODM file with metadata, your have already loaded an existing define.xml with mappings, or created a new define.xml (using menu "File – Create define.xml").

In our case, where we already created the mappings for DM (Demographics) and EC (Exposure as collected), the right part of our screen will look like:

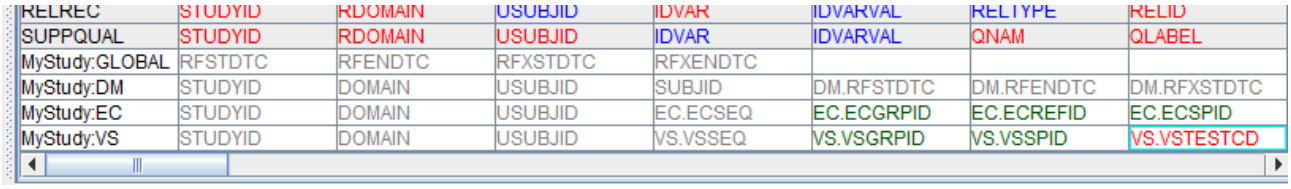


RS	STUDYID	DOMAIN	USUBJID	RS.RSSEQ	RS.RSGRPID	RS.RSREFID	RS.RSSPID
VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD
FA	STUDYID	DOMAIN	USUBJID	FA.FASEQ	FA.FAGRPID	FA.FASPID	FA.FATESTCD
SR	STUDYID	DOMAIN	USUBJID	SR.SRSEQ	SR.SRGRPID	SR.SRREFID	SR.SRSPID
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	RELTYPE	RELID
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL
MyStudy:GLOBAL	RFSTDTC	RFENDTC	RFXSTDTC	RFXENDTC			
MyStudy:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC
MyStudy:EC	STUDYID	DOMAIN	USUBJID	EC.ECSEQ	EC.ECGRPID	EC.ECREFID	EC.ECSPID

We now create a "study-specific instance" of the VS domain just by drag-and-drop of the "VS" row (from the template) to the bottom. The following dialog shows up:

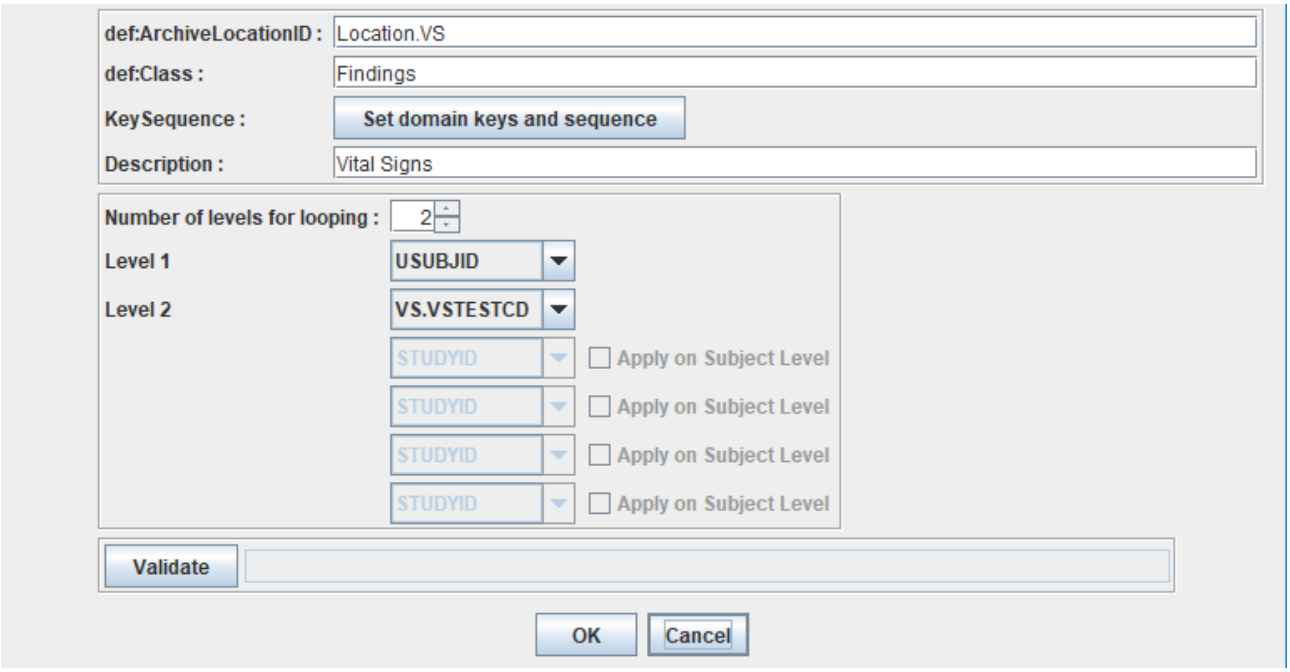


Which we all accept by clicking "OK". The result is:

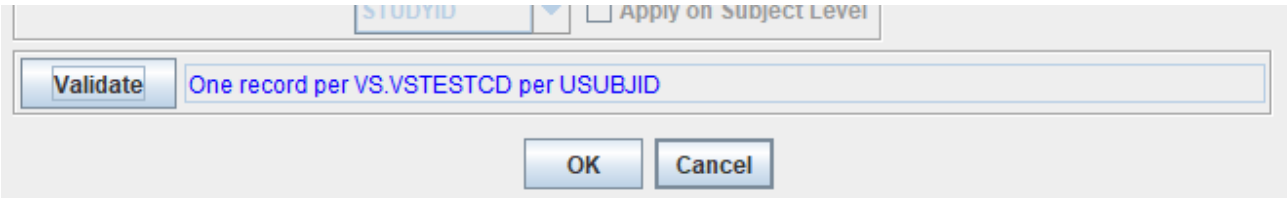


RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	RELTYPE	RELID
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL
MyStudy:GLOBAL	RFSTDTC	RFENDTC	RFXSTDTC	RFXENDTC			
MyStudy:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC
MyStudy:EC	STUDYID	DOMAIN	USUBJID	EC.ECSEQ	EC.ECGRPID	EC.ECREFID	EC.ECSPID
MyStudy:VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD

We see that mappings have already automatically be created for STUDYID, DOMAIN, USUBJID, and VSSEQ – these fields in the table have then been colored grey. We also see that the field for VSTESTCD has a blue border, meaning that it is a "looping variable", i.e. a record will be created for each vital signs test code found in the source. In most cases, the "topic variable" will also be the "looping variable".
If we double-click the first cell "MyStudy:VS", the following dialog is displayed:



Also displaying the looping structure, which essentially is "One record per subject per vital signs test code". This can be visualized by clicking the "Validate" button:

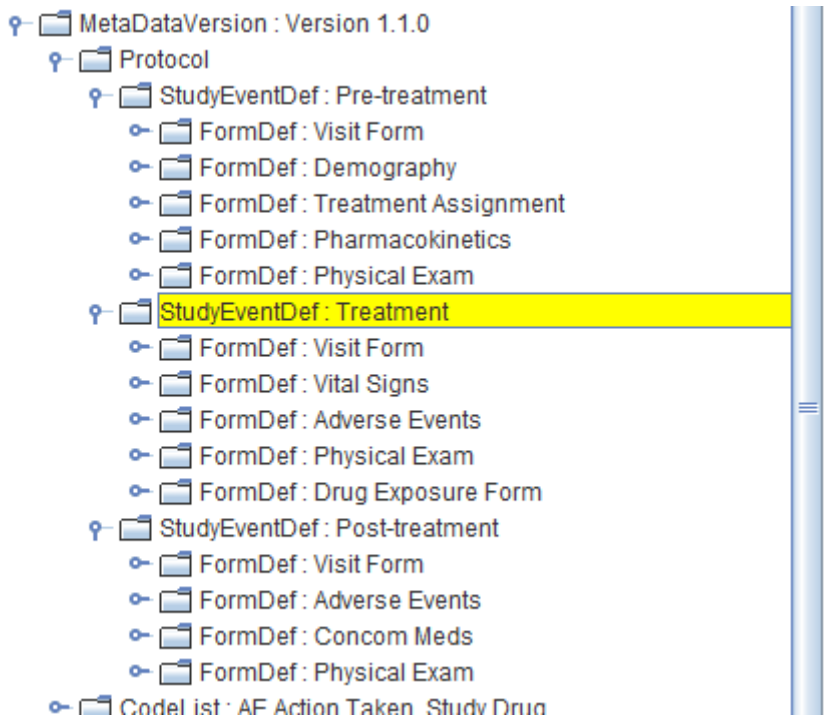


We will keep this for now, and come back later and extend this when necessary.

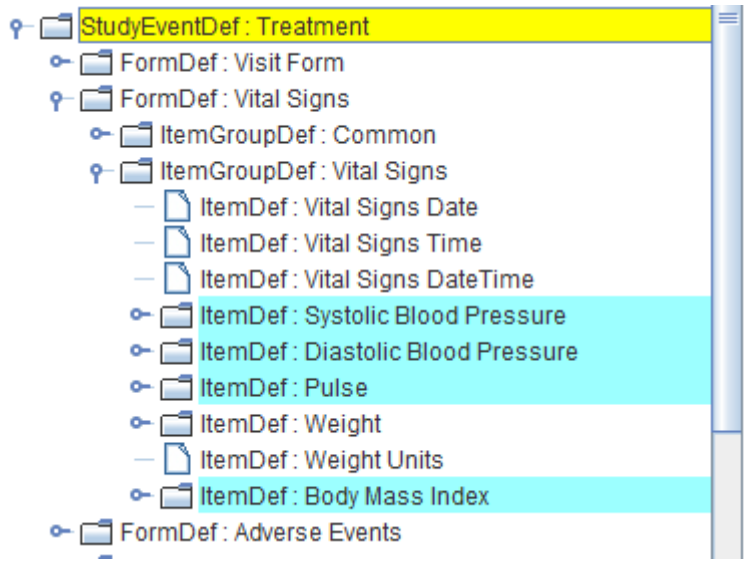
It is always a good idea to first provide the mapping for the looping variables. So we will develop the mapping for VSTESTCD first.

Generating the mapping for VSTESTCD

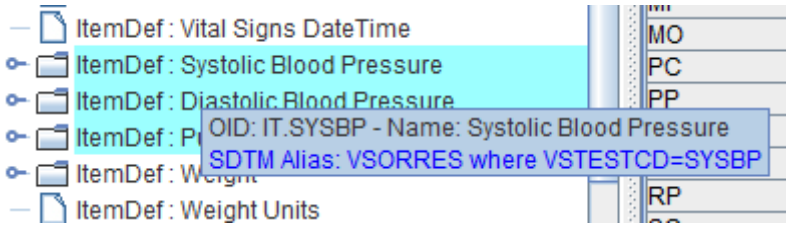
If we look into the ODM tree, we find:



and see that vital signs are only collected in the "Treatment" visit, which is a repeating visit. Further expanding the tree view leads to:

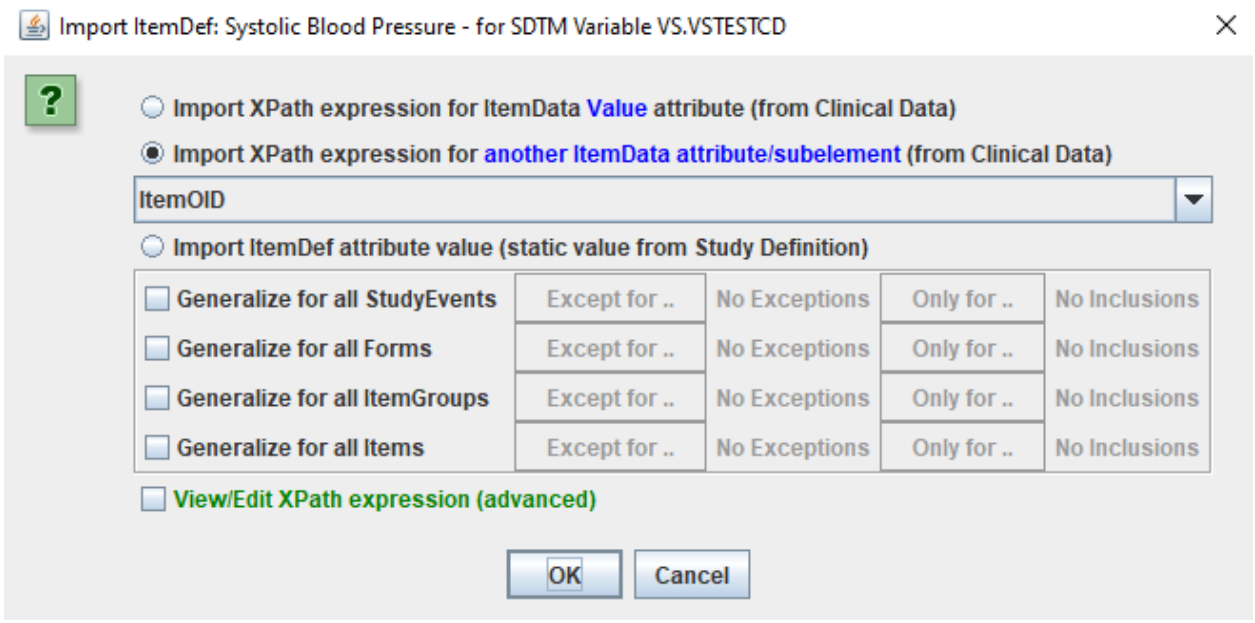


We observe that some of the fields have a cyan background color, meaning that they do already have an "SDTM annotation", i.e. the developers of the study design already annotated these with SDTM information, stating where the data point will later be used in SDTM. For example, if we hold the mouse of "Systolic Blood Pressure", a tooltip is displayed:



We can now start developing the mapping for VSTESTCD.

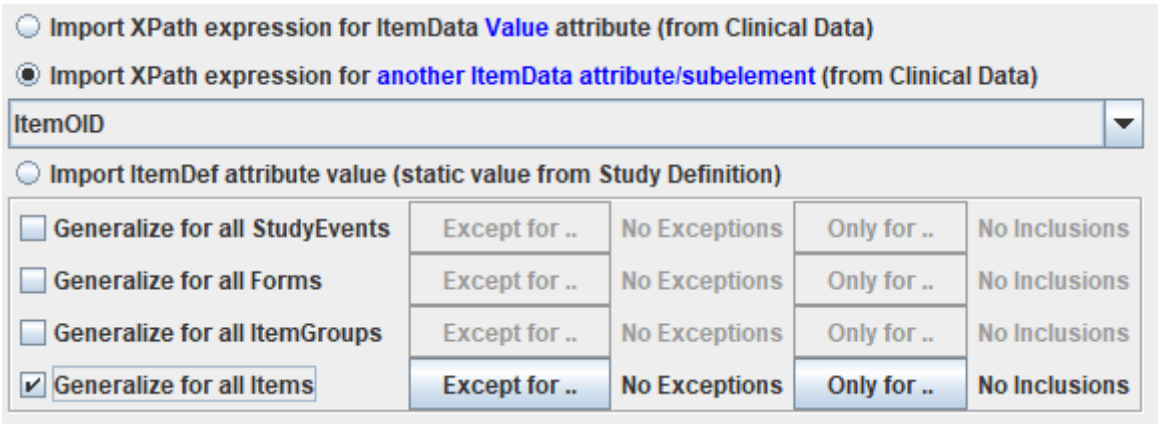
First drag-and-drop from one of the appropriate entries from the ODM tree to the cell "VS.VSTESTCD". If we e.g. take "Systolic Blood Pressure" and drag-and-drop it, the following dialog shows up:



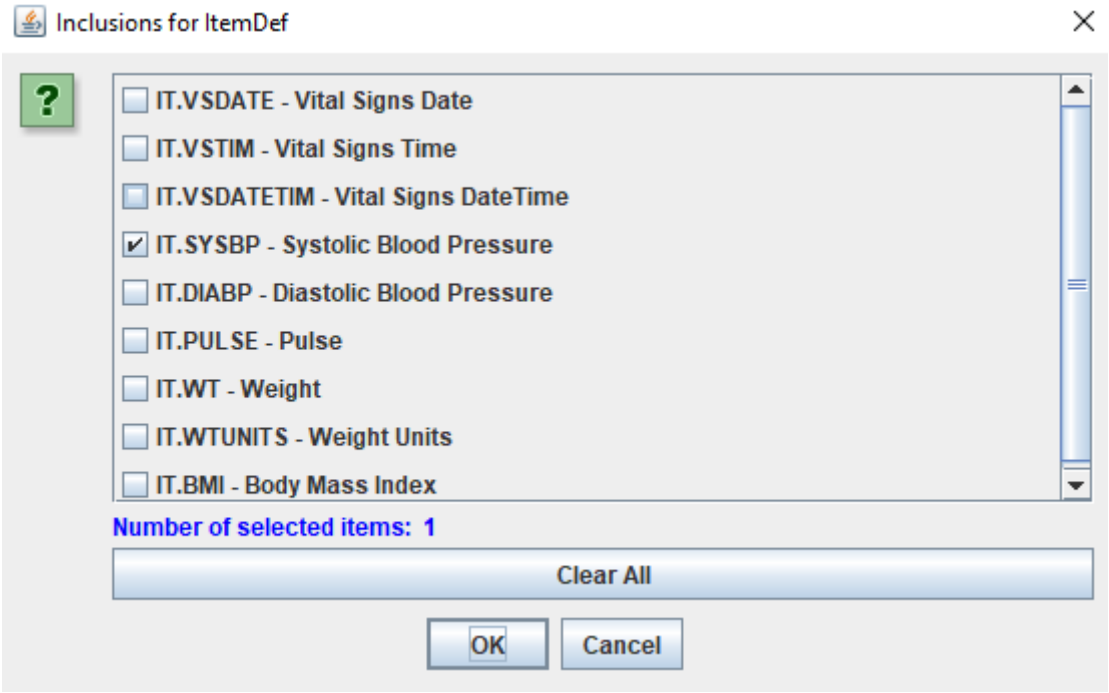
As the system knows it is about a test code, it automatically selects "Import XPath for ..." "ItemOID". If the item contained the name of the test as a value, we would then need to select the upper radiobutton "Import XPath expression for ItemData Value"

What will happing during execution of the mapping? When getting the clinical data, the software will pick up the identifier of the data point (in our case it is "IT.SYSBP") and will use that for mapping to VSTESTCD.

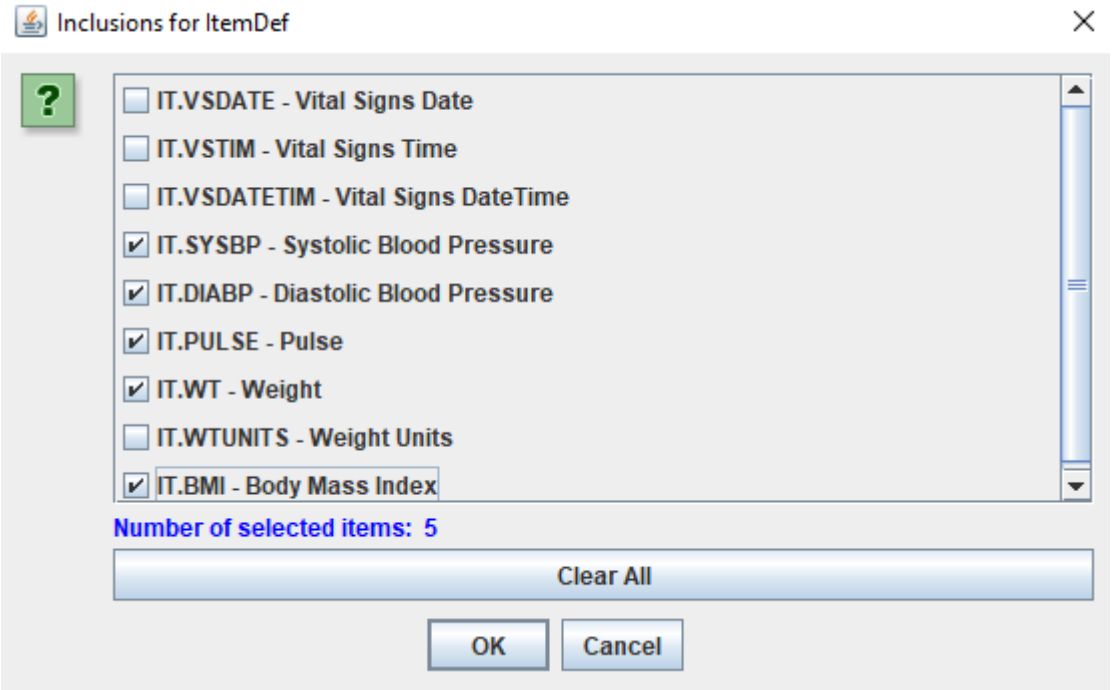
Now, we do have of course more vital signs collected than just the systolic blood pressure, and we also want to get the others too. In order to get them, click the checkbox "Generalize for all Items".



This will select all the data points in the group "Vital Signs", including "Vital Signs Date", "Vital Signs" time and so on. These two are not vital signs of course, so we will want to exclude them, or alternatively, we only want to include the data point codes that really represent.
This can easily be done using one of the buttons "Except for …" or "Only for …". If we use "Only for …" the following dialog is displayed:

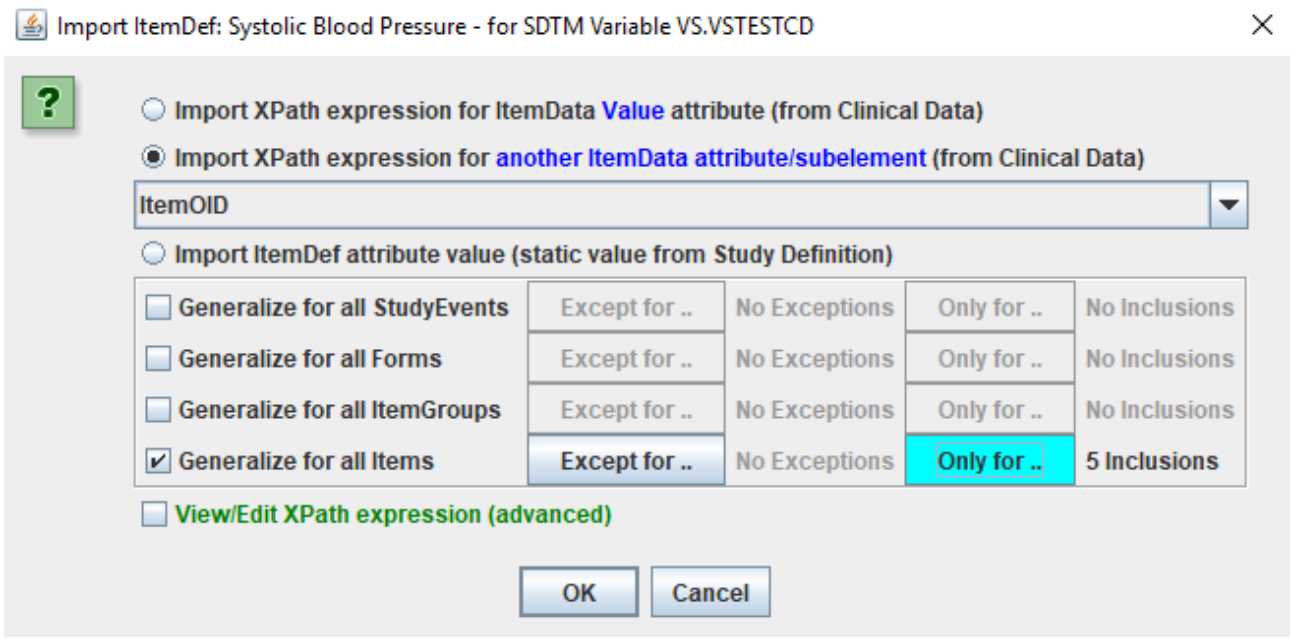


We than select only these items that really represent a vital signs test code, i.e.:



We do not select "Vital Signs Date", "Vital Signs Time", "Vital Signs DateTime", and "Weight Units" as these do not represent vital signs test codes.

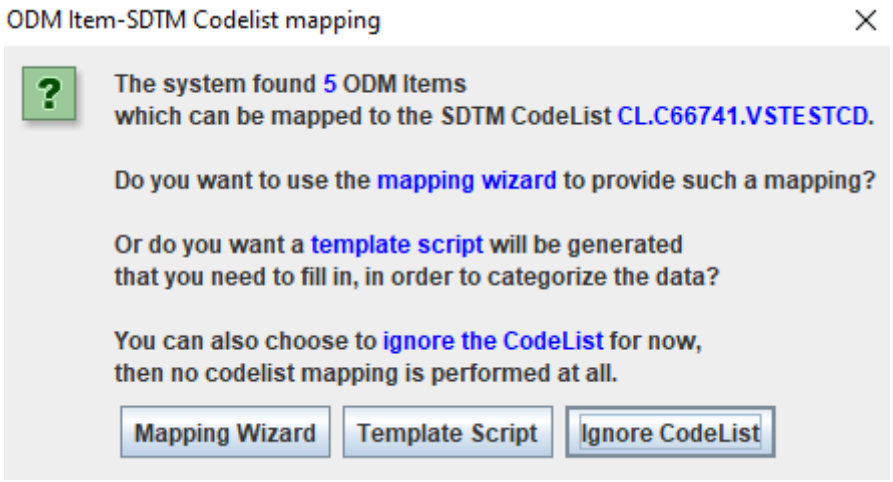
Clicking OK now leads to:



Stating that we will have 5 different vital signs test codes.

If we had the case that vital signs would also have been collected in other visits (study-events) than "treatment", we should select the checkbox "Generalize for all StudyEvents" and use the button "Except for …" or "Only for …" and then select the visits in which vital signs have been collected. This will then take care that these visits are also taken into account.
If the checkbox "Generalize for all StudyEvents" is not checked, the selection will be limited to the current visit only, which is the "treatment" visit.

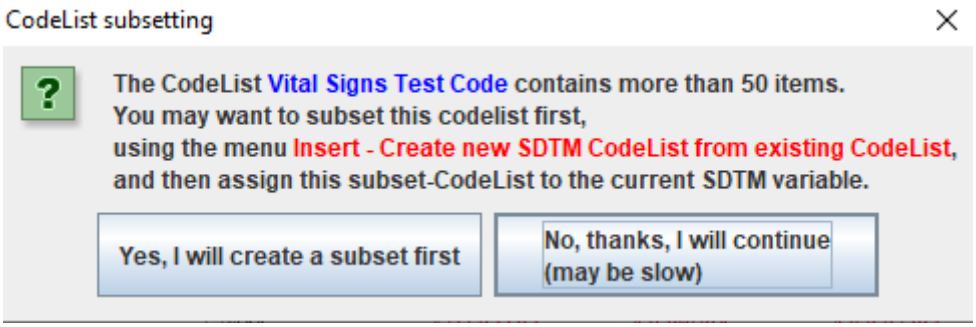
Clicking "OK" then leads to a new dialog:



Stating that there is an SDTM codelist for VSTESTCD in SDTM, and that we can map our 5 test codes in our ODM to the SDTM codelist. We can either choose for using a mapping wizard, which will often be the best choice, generation of a template script that further needs to be completed, or to ignore the SDTM codelist completely. One can already understand that in this case, this will be a bad choice. It is only interesting when the identifier of the ODM item is identical (or closely looks like) the SDTM test code value. This can e.g. be the case when using CDASH forms.

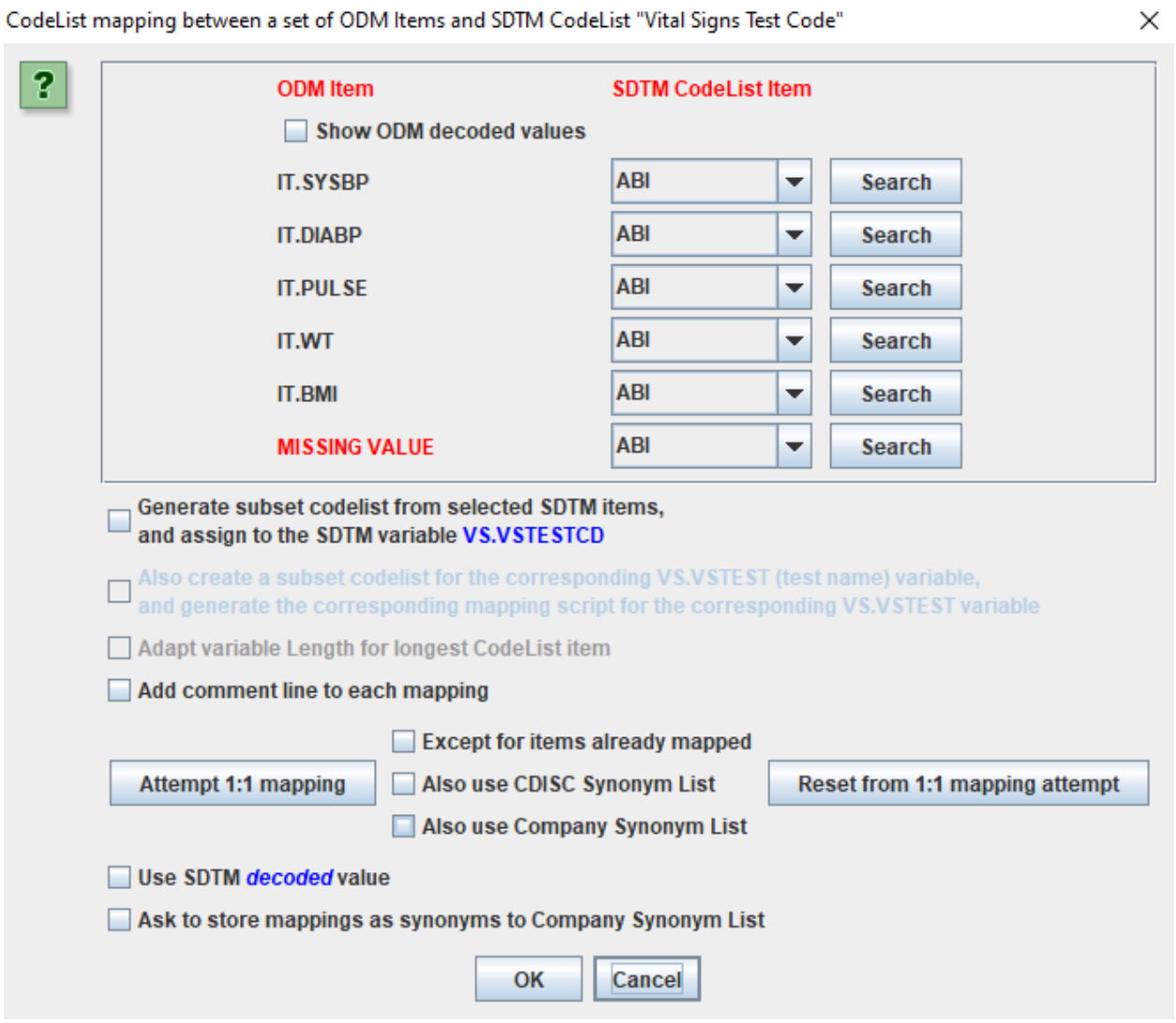
So we check "Mapping Wizard".

This leads to:

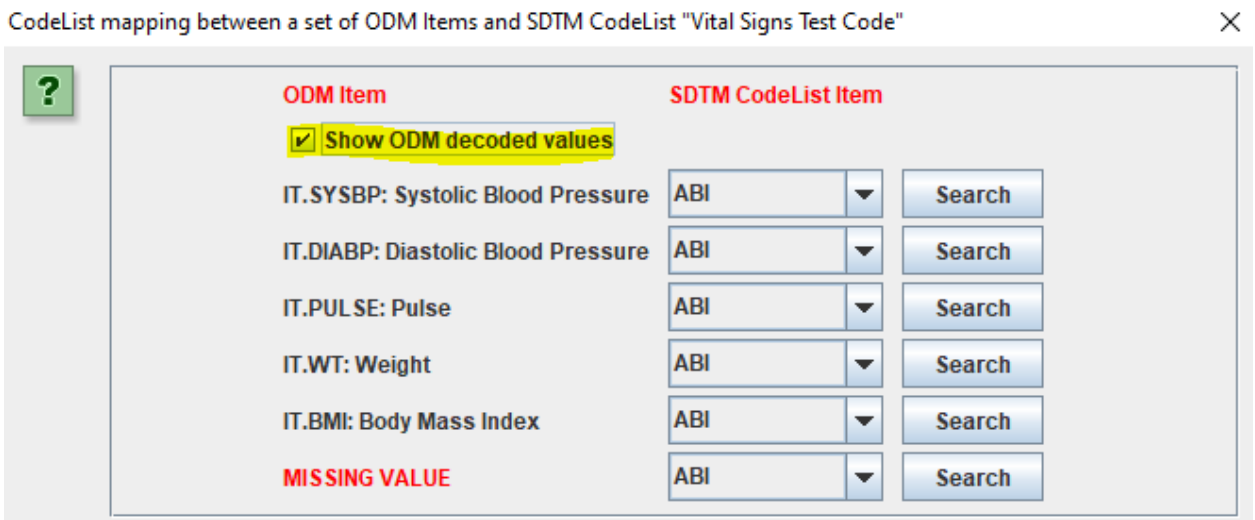


Reason that this appears is that the number of codes in the codelist is pretty large (larger than the threshold 50 - which can be changed using the menu "Options - Properties" anyway), so that it may take some time to set up the wizard when there is a large number of test codes, as is e.g. the case for LBTESTCD. As for VSTESTCD, the number of codes (somewhat more than 60) is still acceptable, we will not create a subset first. So, we click "No, thanks".
When there is a very large number of testcodes, like for LBTESTCD (almost 2,500 test codes, still growing), it may take a few minutes when not generating a subset. However, generating a subset first also takes time...

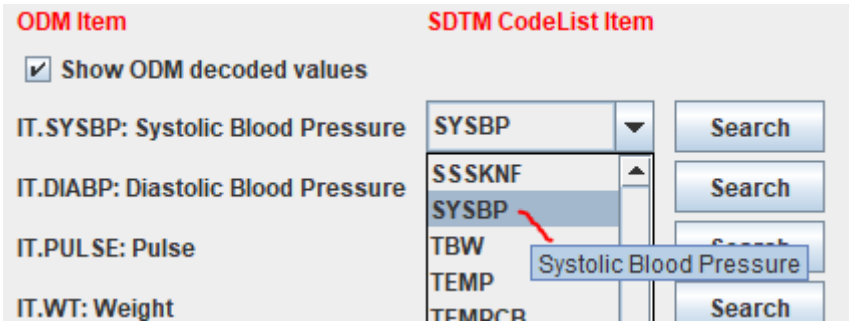
After about 5 seconds, the mapping wizard is displayed:



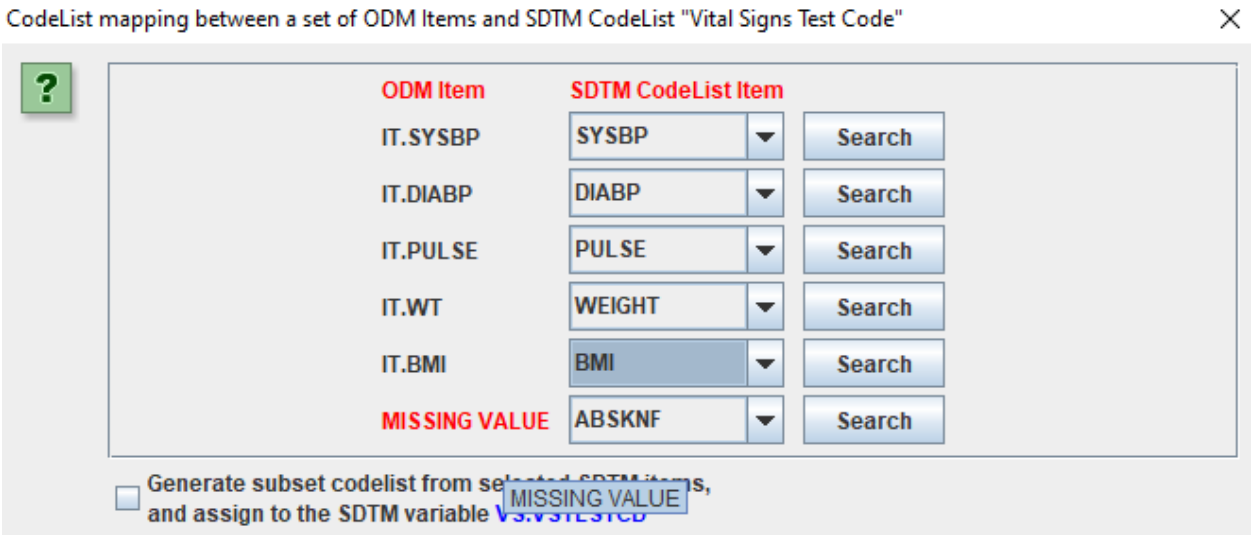
In this case, the values of the "ODM Item" are displayed as the OIDs. This will not always be self-explaining (an OID can be anything, including e.g. "550e8400-e29b-41d4-a716-446655440000" which is a "UUID"). In such a case, it is also a good idea to check "Show ODM decoded values":



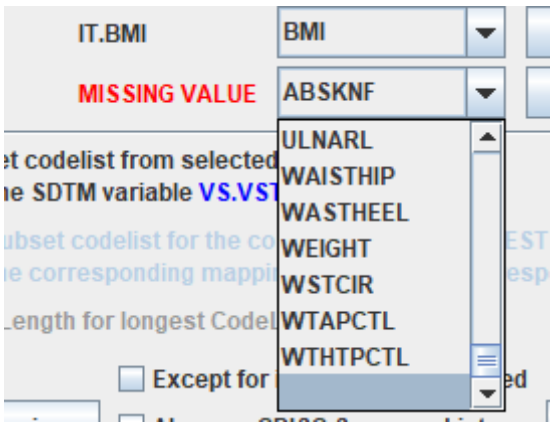
For each of the entries on the left (our ODM codes) we can now easily map to an SDTM "vitals signs test code", by a simple selection on the right. For example, for "Systolic Blood Pressure":



In many cases, just typing the first characters of the desired test code, allows to find the desired test code. This will e.g. lead to:



For the last row "MISSING VALUE", we should also make a choice. Usually, one will select the "NULL" value, i.e. scroll to the bottom of the list and select the last entry which is the empty entry:

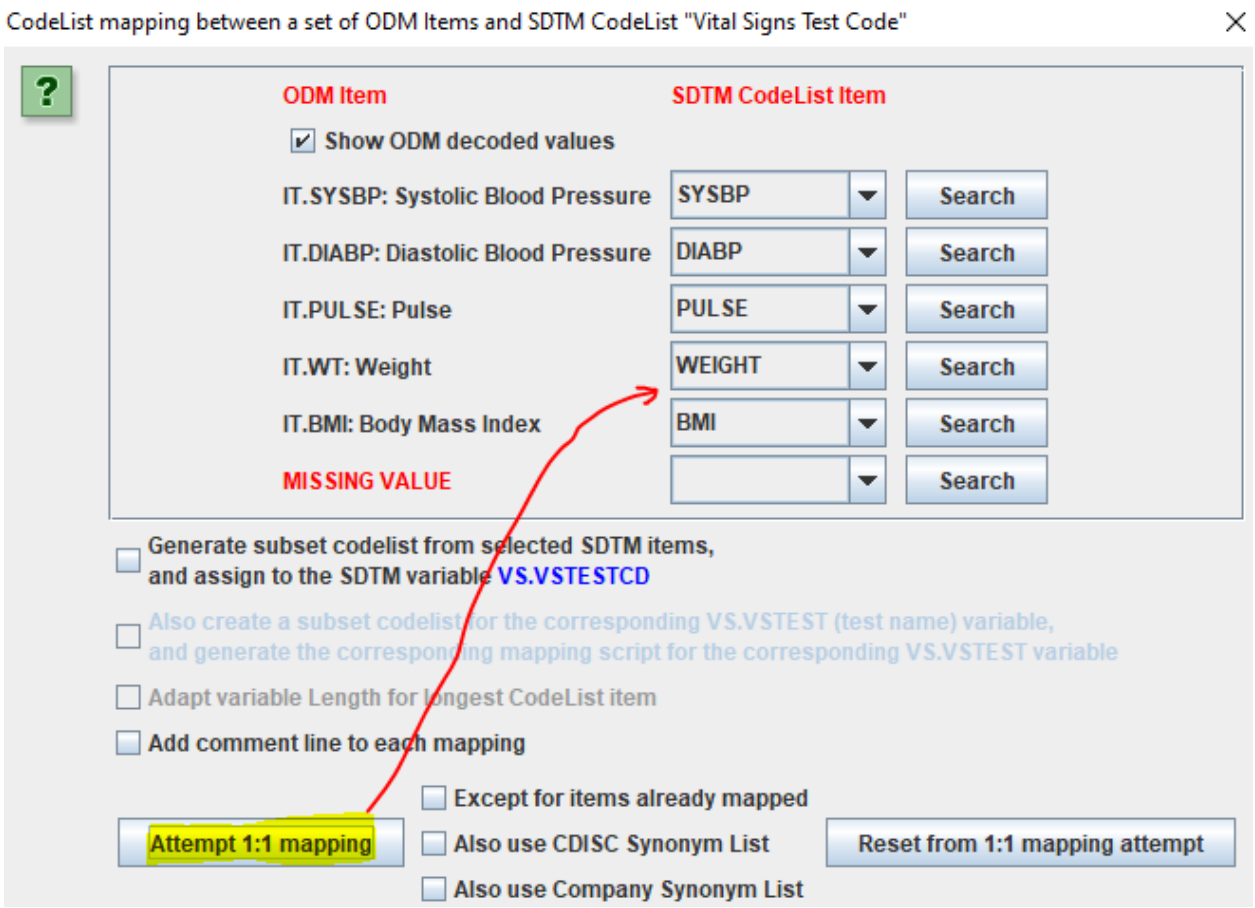


With the result:

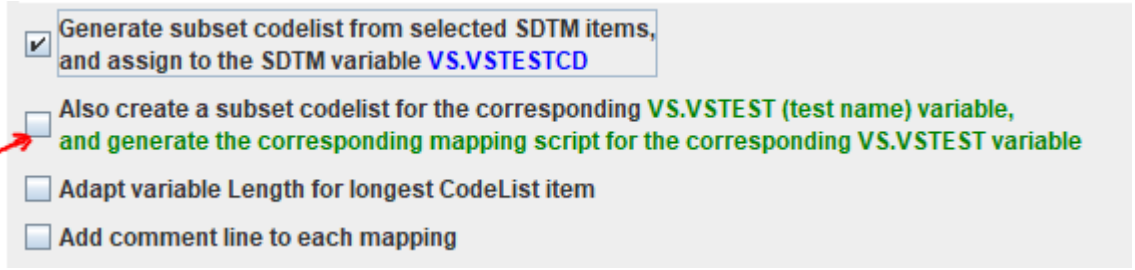
The case of a "missing value" should however never occur in practice, as VSTESTCD is a "looping

variable" and we iterate only over data points that represent one of these 5 test codes.

Another possibility is to rely on "word similarity" between the description of the ODM item and the SDTM variable name. This can be tried using the button "Attempt 1:1 mapping", leading to:



The use of "CDISC Synonym List" and "Company Synonym List" will be explained in another tutorial.
It is often also wise to check the checkbox "Generate subset codelist from selected SDTM items, and assign to the SDTM variable …", as later, for submission, we only want to have a codelist for VSTESTCD for which vitals signs tests were [1].
If we check this checkbox, another one becomes available:



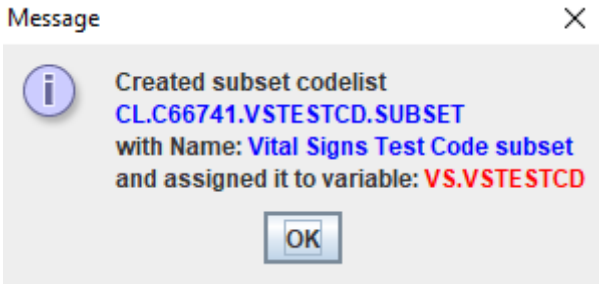
suggesting that we should also have a "subset codelist" for VSTEST (Vital Signs Test Name) too. When checked, the corresponding subset codelist for VSTEST (Vital Signs Test Name) will automatically be created. This is possible due to the 1:1 correspondence between values for -TESTCD and -TEST variables. Additionally, the mapping script for also VSTEST will be

[1] The word "planned" is important here. For example, if "weight" was planned, but never actually collected, we still would like to have it in the define.xml codelist for VSTESTCD.

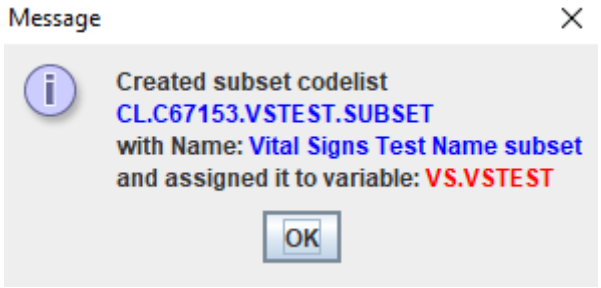
generated fully automatically behind the scene. So, one is then essentially hitting two targets with one shot! This feature should however be used with some care, and the additionally generated mapping script for VSTEST also be inspected.

We can also check the checkbox "Adapt variable length for longest codelist item". If we do so, the "Length" attribute in the define.xml will be assigned the value 6, as the longest code is "WEIGHT". We can however also do this later when generating our datasets. Assigning a good value for "Length" is especially important for the case that we want to generate SAS-XPT datasets later, as regulatory authorities want us to have the XPT files as compact as possible, as XPT is a very inefficient format for storage.

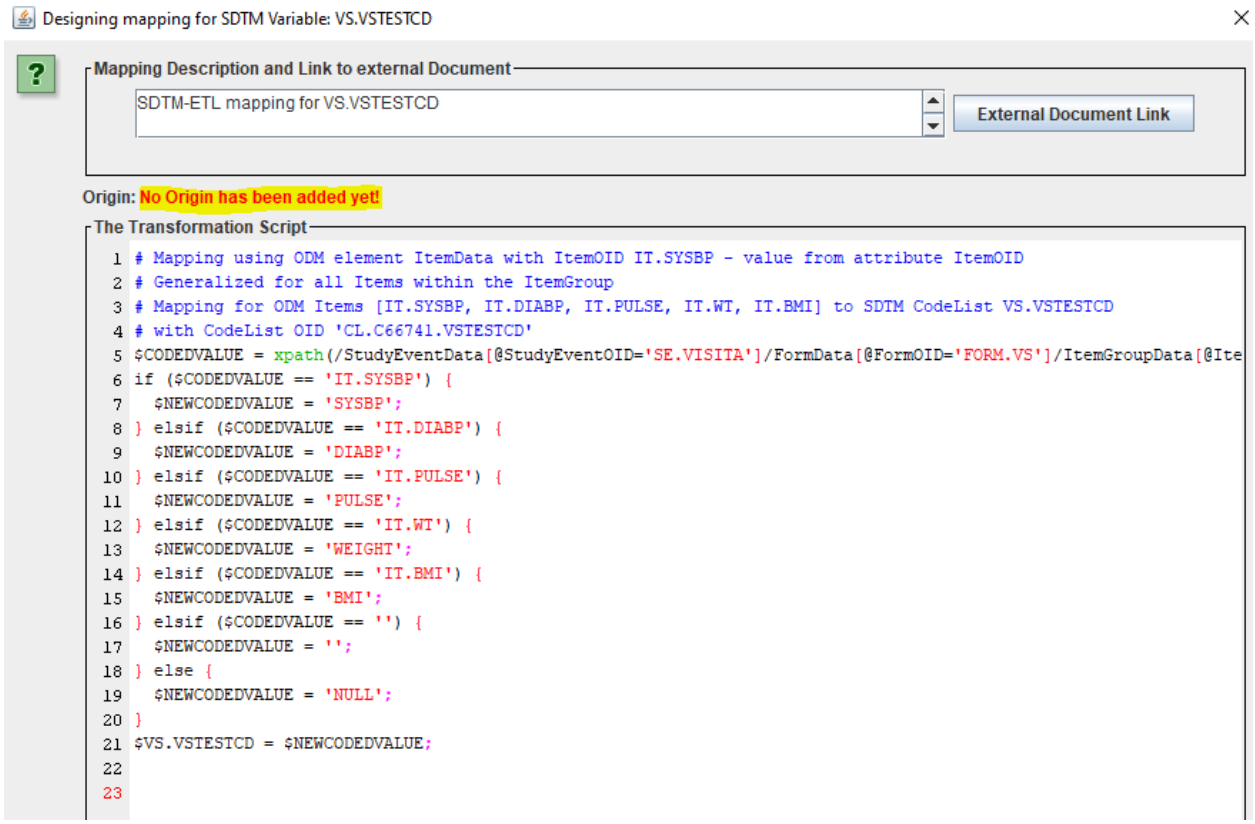
Clicking "OK" then first shows us a message:



followed by:



and then generates the mapping script:

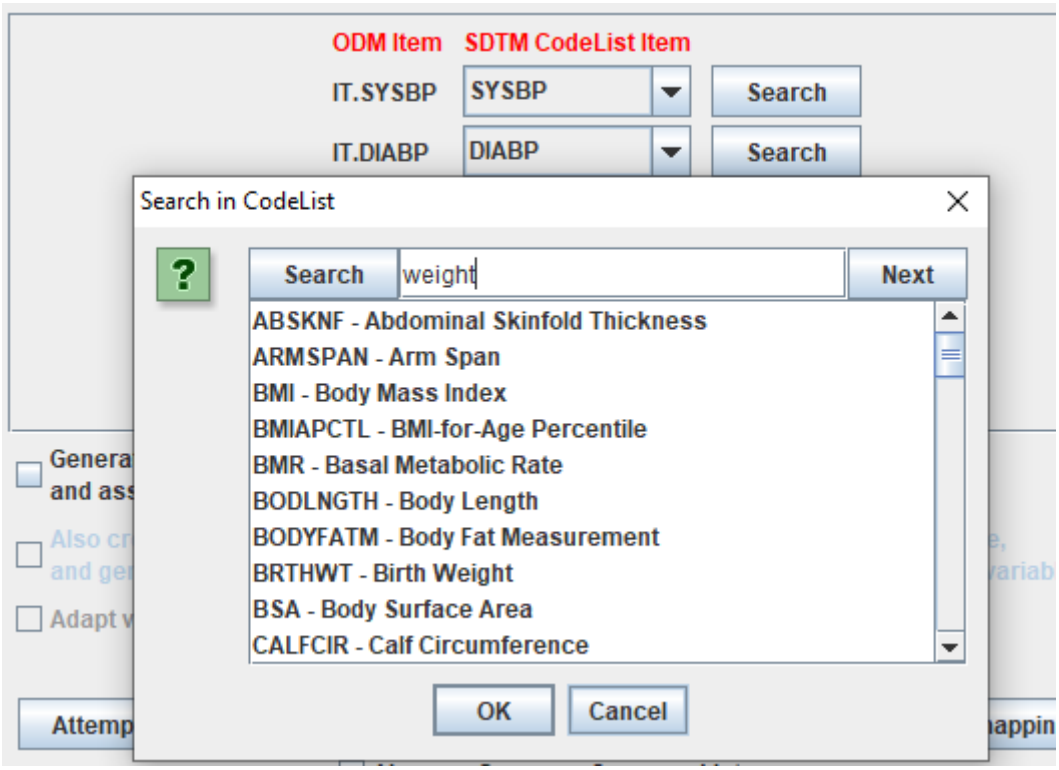


Remark the line "No origin has been added yet". This is something we will still need to do later[2].

If needed, one can then still adapt the mapping script manually. In many cases, this will not be necessary.
One then accepts the mapping script (which is then stored in the underlying define.xml) by clicking the "OK" button.

Another method to find a suitable value for VSTESTCD, is to use one of the "Search" buttons. For example for "IT.WEIGHT", clicking it shows up a "Search" window:

[2] In most cases, the "Origin" for -TESTCD and -TEST is "Assigned", and, in the case of Define-XML 2.1, the "Source" is "Sponsor", as it is the responsibility of the sponsor to do the assignment.



and when typing in "weight", followed by "Search" will also provide "WEIGHT" as the test code. This feature is especially interesting when the OIDs and even the "Name" of the item are not very meaningful, as we have seen in a number of cases of ODM exports of EDC systems", but the mapper does now (from other sources?) what the meaning of each ODM item is.

After we click "OK" after inspection of the automatically generated mapping script for VSTESTCD, we can now also check the automatically generated mapping script for VSTEST, just by a double-click on the VS.VSTEST cell:

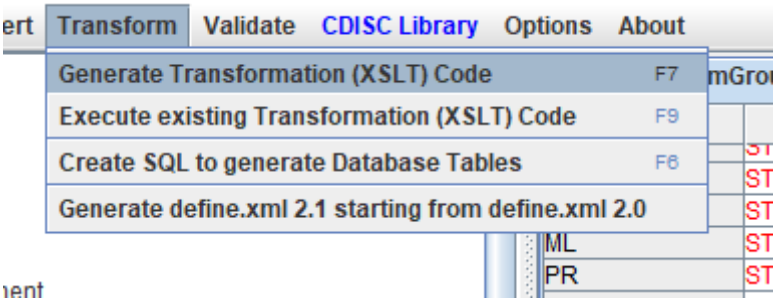
The additionally mapping script for VSTEST then looks like:

The Transformation Script

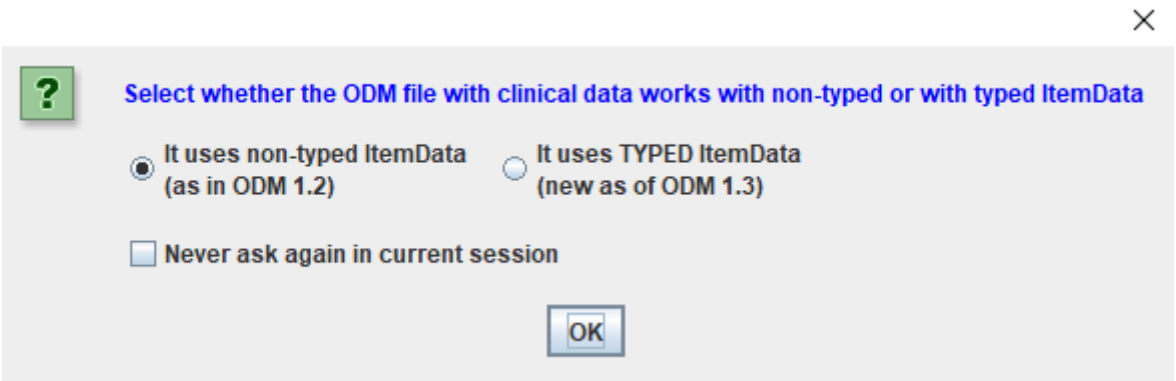
```
# Mapping using ODM element ItemData with ItemOID IT.SYSBP - value from attribute ItemOID
# Generalized for all Items within the ItemGroup
# Mapping for ODM Items [IT.SYSBP, IT.DIABP, IT.PULSE, IT.WT, IT.BMI] to SDTM CodeList VS.VSTESTCD
# with CodeList OID 'CL.C66741.VSTESTCD'
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Ite
# Mapping code for variable VS.VSTEST
# automatically generated from the mapping script for the corresponding variable null
# using decoded values of the VS.VSTESTCD codelist
if ($CODEDVALUE == 'IT.SYSBP') {
  $NEWCODEDVALUE = 'Systolic Blood Pressure';
} elsif ($CODEDVALUE == 'IT.DIABP') {
  $NEWCODEDVALUE = 'Diastolic Blood Pressure';
} elsif ($CODEDVALUE == 'IT.PULSE') {
  $NEWCODEDVALUE = 'Pulse Rate';
} elsif ($CODEDVALUE == 'IT.WT') {
  $NEWCODEDVALUE = 'Weight';
} elsif ($CODEDVALUE == 'IT.BMI') {
  $NEWCODEDVALUE = 'Body Mass Index';
} elsif ($CODEDVALUE == '') {
  $NEWCODEDVALUE = 'NULL';
} else {
  $NEWCODEDVALUE = 'NULL';
}
$VS.VSTEST = $NEWCODEDVALUE;

```

One can now do a first test, by using the menu "Transform – Generate Transformation (XSLT) Code".



This leads to:

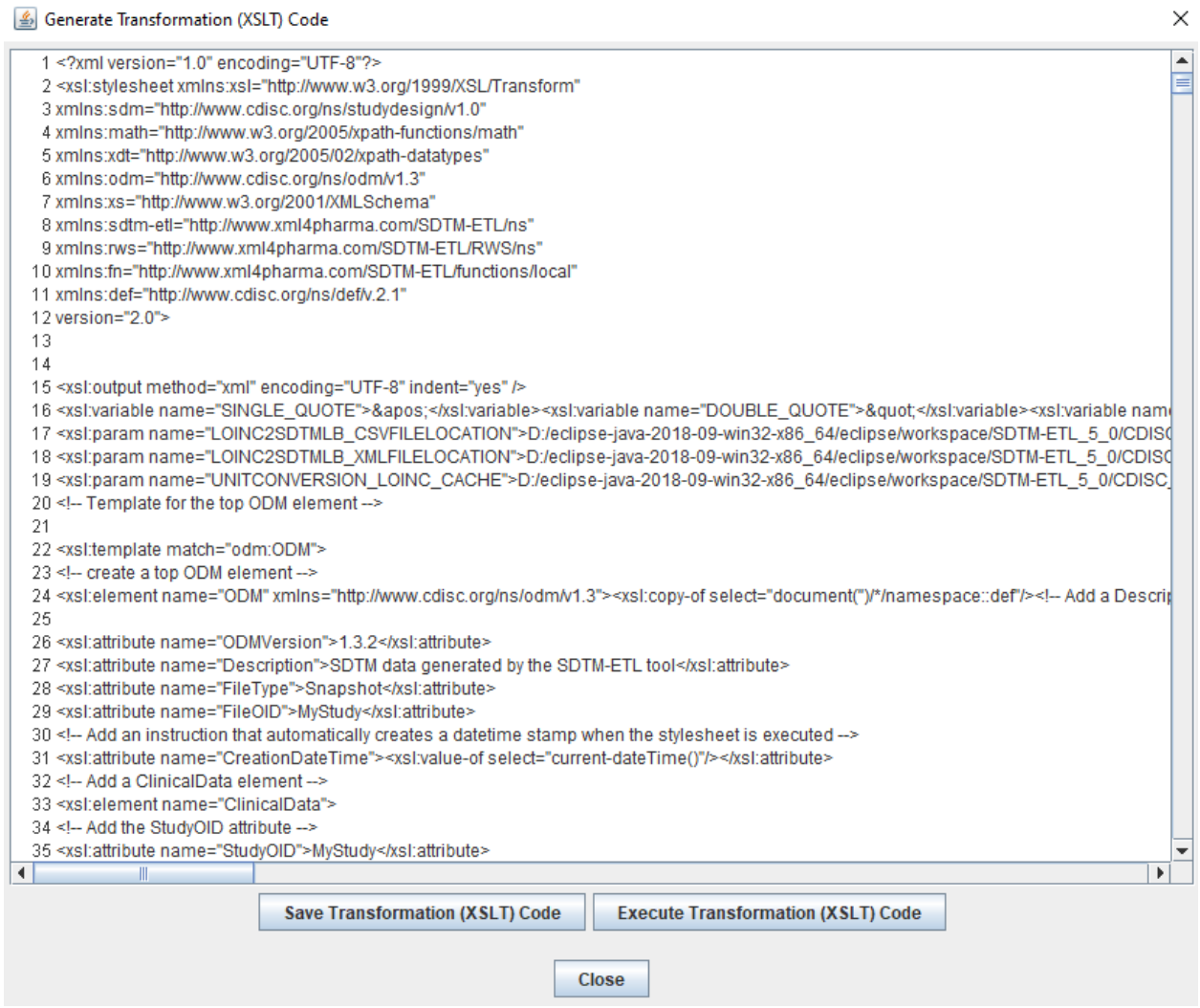


Most of the EDC vendors (>90%) use the "classic" ODM export, with "non-typed ItemData"[3]. Only a few use the "Typed ItemData" flavor of ODM. If you have doubts, ask your EDC vendor which of both is exported, or just try both here – if you have the wrong one, the output will simply be empty.

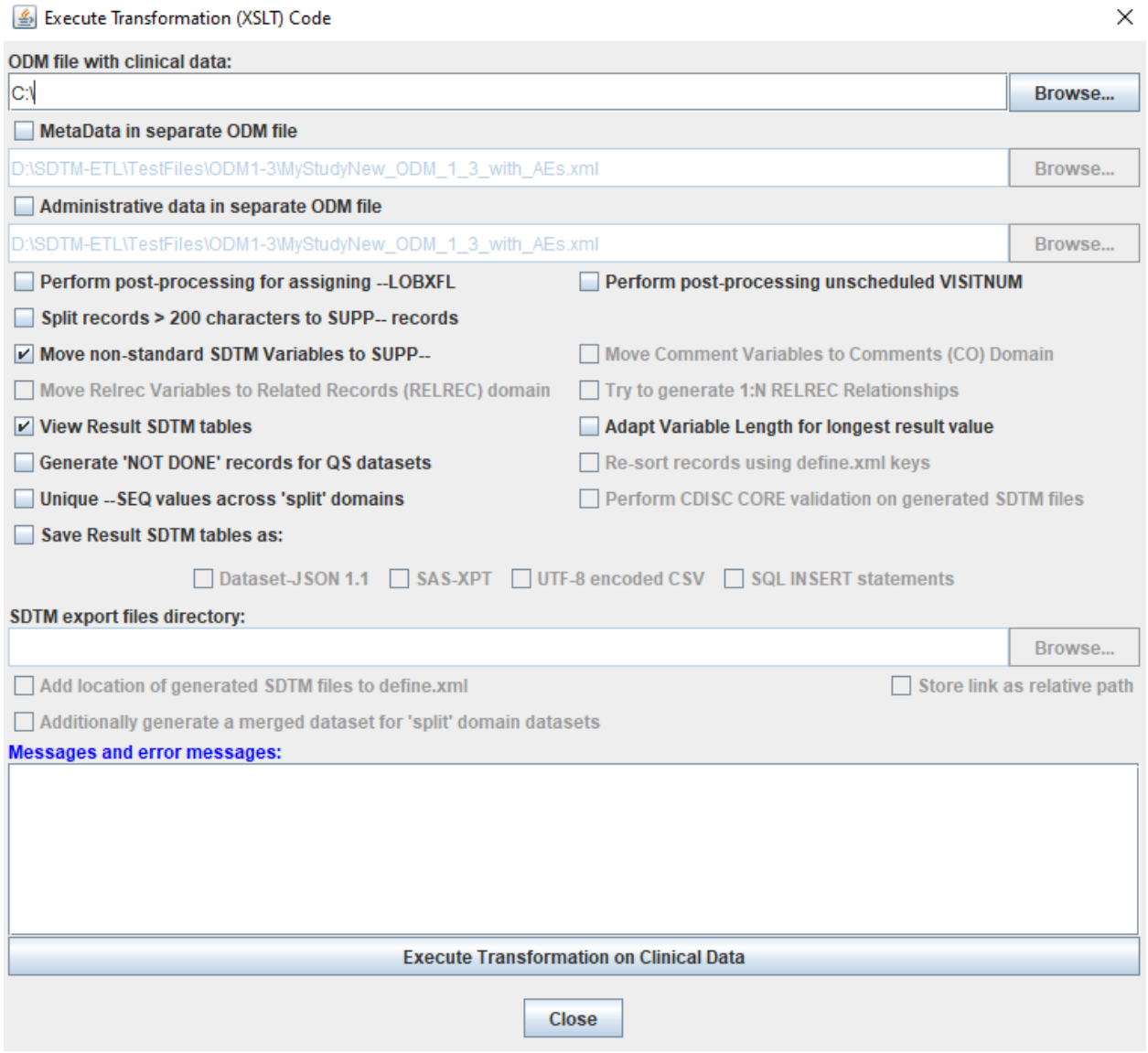
The following dialog is displayed after clicking "OK":

---

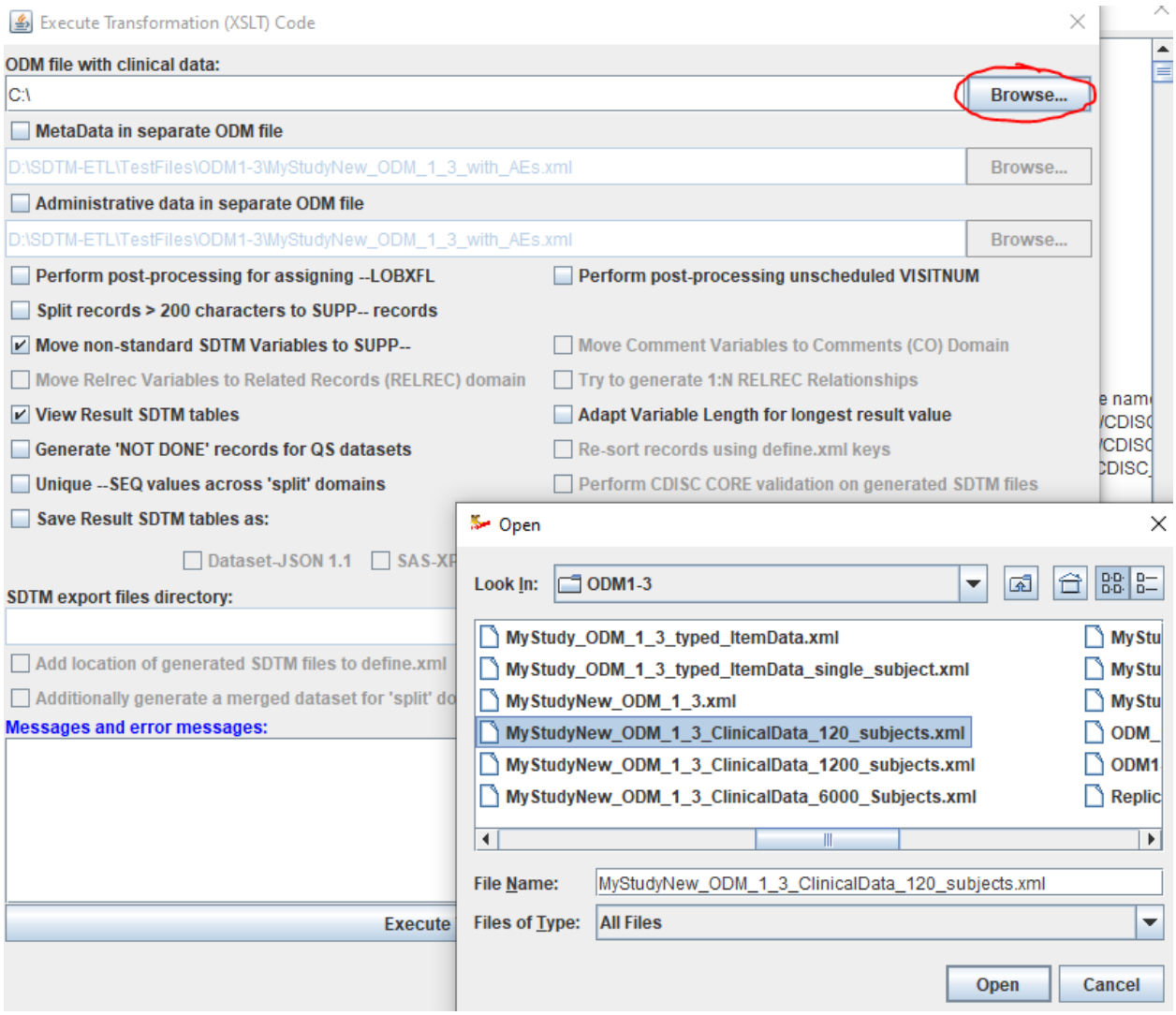
[3] If you already use the new ODMv2 format, this dialog will not appear.



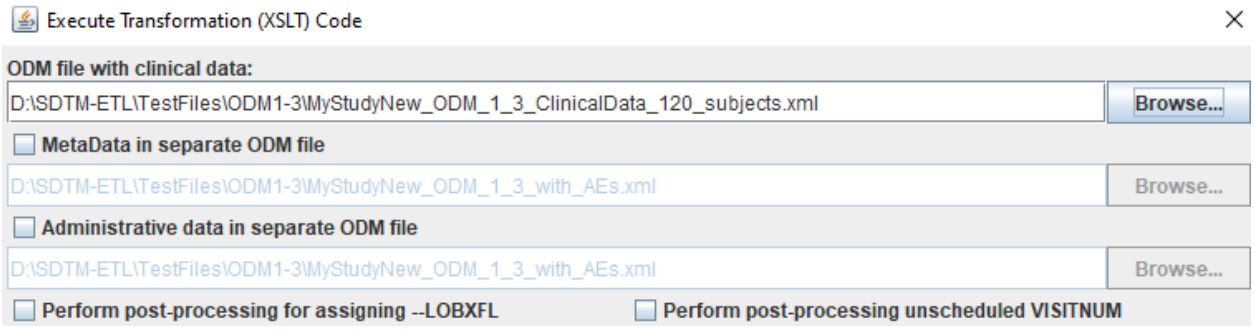
Showing you the generated XSLT. If you want to use it for offline SDTM generation, you can save it now. You can also skip this step by using the menu "Options – Settings" and checking "Skip display of generated XSLT".
Then click the button "Execute Transformation (XSLT) Code", leading to:



Now select an input file with ODM clinical data (as exported from your EDC system) by using the first "Browse" button. For example:



leading to:



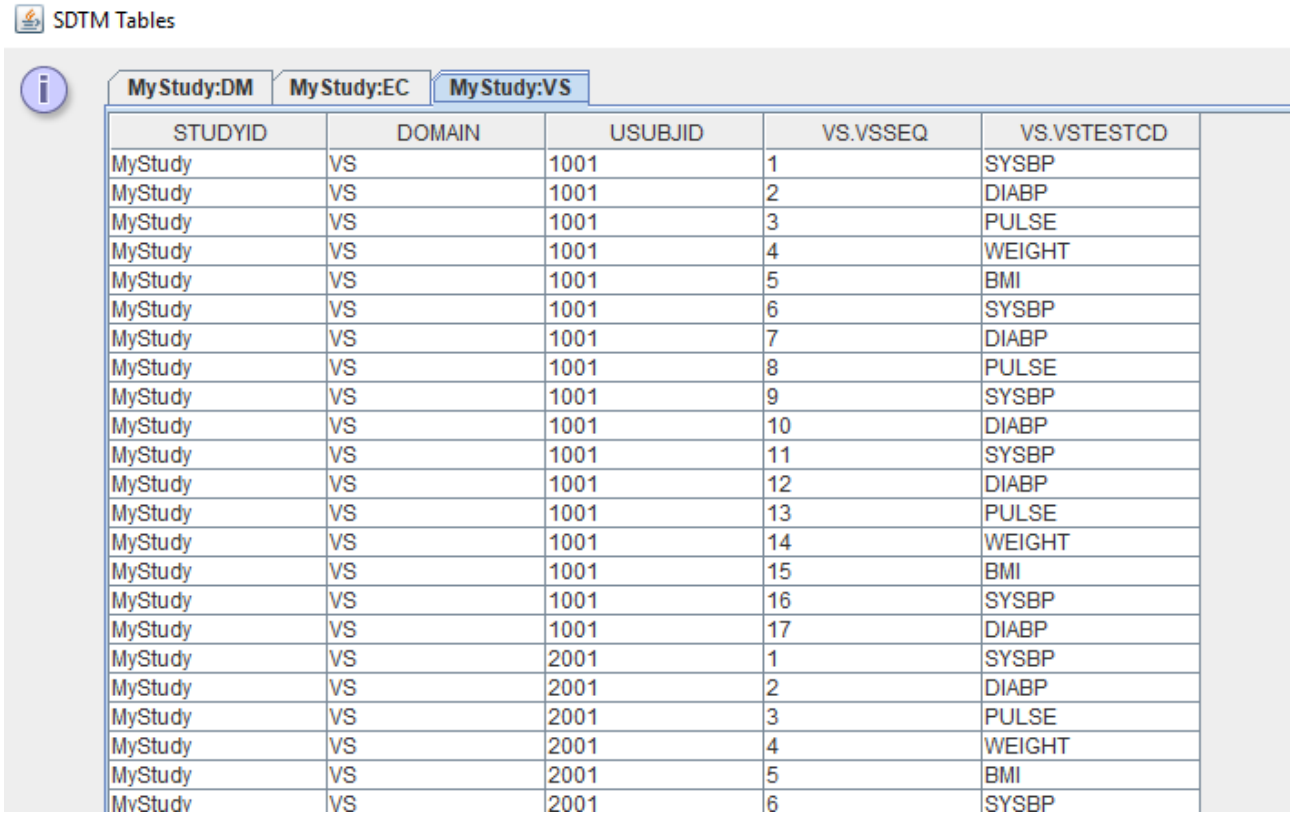
In most cases, you will not need to check the checkbox "Metadata in separate ODM file". This will only be the case e.g. when you need "decoded" codelist values from the ODM, or the "description" or "question text" of a data point[4].
If you already want to generate SAS-XPT datasets, also check the checkbox "Save Result SDTM tables as:" and as "SAS-XPT", and then select a directory where these need to be written to. During testing however, this will usually not be necessary, as the generated SDTM tables will be displayed by the software itself.

All the other options will be explained later.

[4] This will often be the case for questionnaires, that are mapped to the QS domain.

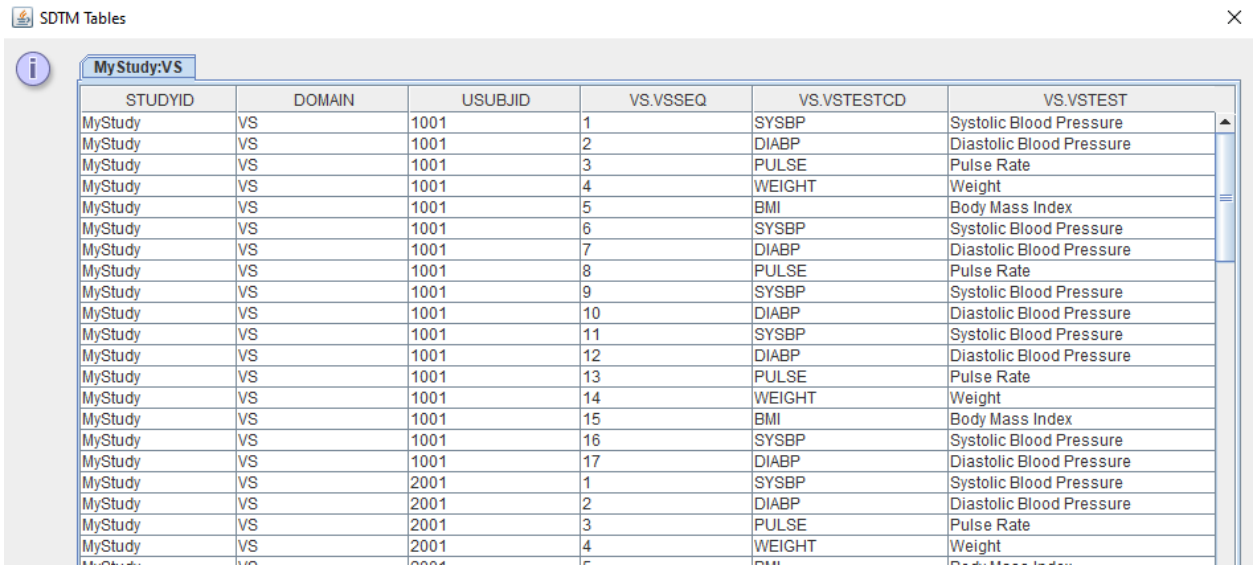
Clicking "Execute Transformation on Clinical Data" starts the transformation. In case of very complicated mappings and large amounts of data this can take a few minutes.

In our case, the results display after a few seconds:



STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD
MyStudy	VS	1001	1	SYSBP
MyStudy	VS	1001	2	DIABP
MyStudy	VS	1001	3	PULSE
MyStudy	VS	1001	4	WEIGHT
MyStudy	VS	1001	5	BMI
MyStudy	VS	1001	6	SYSBP
MyStudy	VS	1001	7	DIABP
MyStudy	VS	1001	8	PULSE
MyStudy	VS	1001	9	SYSBP
MyStudy	VS	1001	10	DIABP
MyStudy	VS	1001	11	SYSBP
MyStudy	VS	1001	12	DIABP
MyStudy	VS	1001	13	PULSE
MyStudy	VS	1001	14	WEIGHT
MyStudy	VS	1001	15	BMI
MyStudy	VS	1001	16	SYSBP
MyStudy	VS	1001	17	DIABP
MyStudy	VS	2001	1	SYSBP
MyStudy	VS	2001	2	DIABP
MyStudy	VS	2001	3	PULSE
MyStudy	VS	2001	4	WEIGHT
MyStudy	VS	2001	5	BMI
MyStudy	VS	2001	6	SYSBP

Showing that 17 vital signs data points have been collected for the first subject (1001). Remark that "VSSEQ" has been generated automatically, this does not need to be programmed by the user.

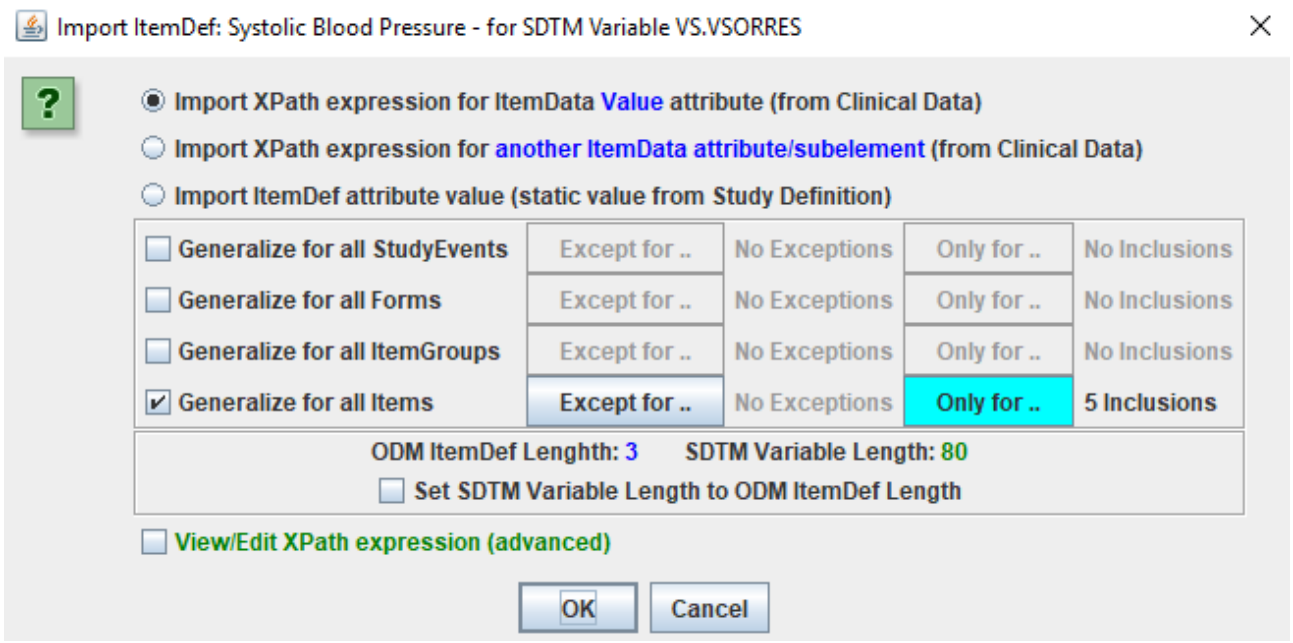


STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST
MyStudy	VS	1001	1	SYSBP	Systolic Blood Pressure
MyStudy	VS	1001	2	DIABP	Diastolic Blood Pressure
MyStudy	VS	1001	3	PULSE	Pulse Rate
MyStudy	VS	1001	4	WEIGHT	Weight
MyStudy	VS	1001	5	BMI	Body Mass Index
MyStudy	VS	1001	6	SYSBP	Systolic Blood Pressure
MyStudy	VS	1001	7	DIABP	Diastolic Blood Pressure
MyStudy	VS	1001	8	PULSE	Pulse Rate
MyStudy	VS	1001	9	SYSBP	Systolic Blood Pressure
MyStudy	VS	1001	10	DIABP	Diastolic Blood Pressure
MyStudy	VS	1001	11	SYSBP	Systolic Blood Pressure
MyStudy	VS	1001	12	DIABP	Diastolic Blood Pressure
MyStudy	VS	1001	13	PULSE	Pulse Rate
MyStudy	VS	1001	14	WEIGHT	Weight
MyStudy	VS	1001	15	BMI	Body Mass Index
MyStudy	VS	1001	16	SYSBP	Systolic Blood Pressure
MyStudy	VS	1001	17	DIABP	Diastolic Blood Pressure
MyStudy	VS	2001	1	SYSBP	Systolic Blood Pressure
MyStudy	VS	2001	2	DIABP	Diastolic Blood Pressure
MyStudy	VS	2001	3	PULSE	Pulse Rate
MyStudy	VS	2001	4	WEIGHT	Weight

Continuing with VSORRES

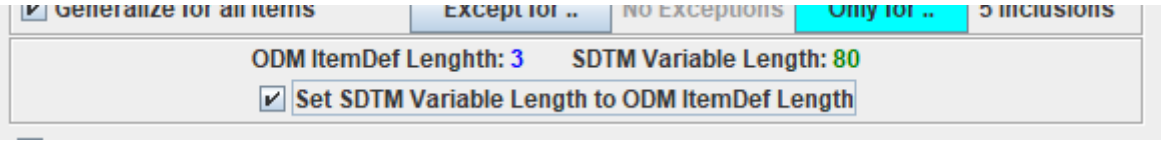
Once the mapping for VSTESTCD has been developed (which can be a bit more challenging when the tests are divided over different forms), generating the mapping for VSORRES is pretty easy.

Select the cell "VS.VSORRES" and then drag-and-drop one of the tree nodes representing a vital signs test to the cell "VS.VSORRES". Again, a dialog is displayed.

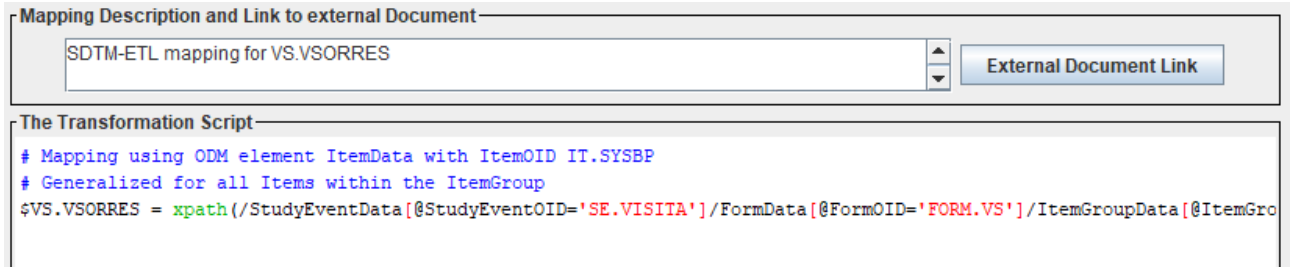


This time we must import the value of the data point, not the test code. So, we let the checkbox "Import … for ItemData Value attribute …" selected. We also see that the system remembers the "generalization" with the 5 inclusions, so there is nothing to be changed there.

Also notice the line "ODM ItemDef Length" (near the bottom) stating that in the ODM the value has been declared as of a maximum length of 3, whereas in SDTM the value from the template is 80. So, not a bad idea to check the checkbox "Set SDTM Variable Length to ODM ItemDef Length" which will set the value for the maximum length in SDTM to 3 too:



Clicking "OK" leads to:



After accepting the mapping script, testing on our clinical data using the menu "Transform - Generate …", and executing the steps as before, leads to:

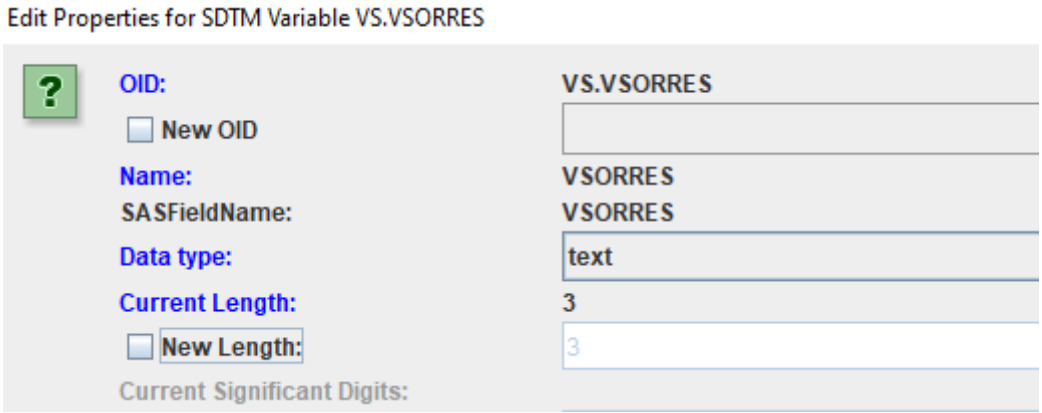
SDTM Tables

MyStudy:VS

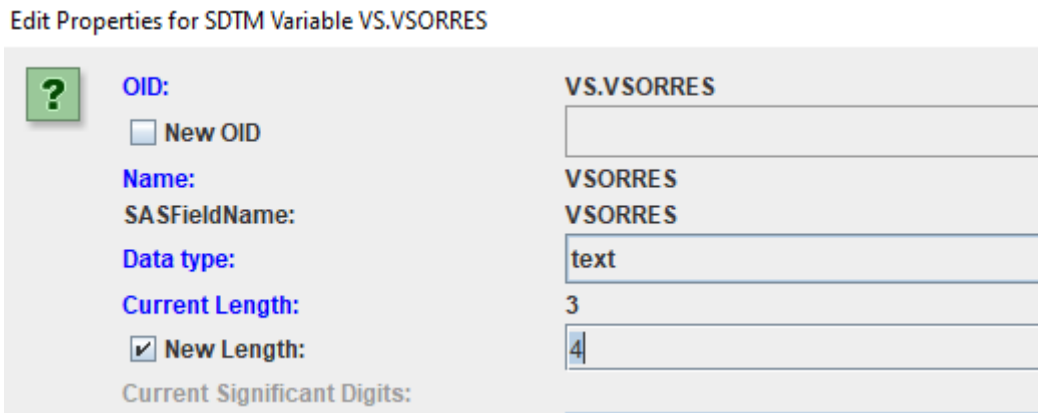
STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSORRES
MyStudy	VS	1001	1	SYSBP	Systolic Blood Press...	100
MyStudy	VS	1001	2	DIABP	Diastolic Blood Pres...	70
MyStudy	VS	1001	3	PULSE	Pulse Rate	62
MyStudy	VS	1001	4	WEIGHT	Weight	88
MyStudy	VS	1001	5	BMI	Body Mass Index	23
MyStudy	VS	1001	6	SYSBP	Systolic Blood Press...	108
MyStudy	VS	1001	7	DIABP	Diastolic Blood Pres...	74
MyStudy	VS	1001	8	PULSE	Pulse Rate	65
MyStudy	VS	1001	9	SYSBP	Systolic Blood Press...	107
MyStudy	VS	1001	10	DIABP	Diastolic Blood Pres...	75
MyStudy	VS	1001	11	SYSBP	Systolic Blood Press...	105
MyStudy	VS	1001	12	DIABP	Diastolic Blood Pres...	76
MyStudy	VS	1001	13	PULSE	Pulse Rate	63
MyStudy	VS	1001	14	WEIGHT	Weight	88.2
MyStudy	VS	1001	15	BMI	Body Mass Index	23
MyStudy	VS	1001	16	SYSBP	Systolic Blood Press...	108
MyStudy	VS	1001	17	DIABP	Diastolic Blood Pres...	74
MyStudy	VS	2001	1	SYSBP	Systolic Blood Press...	100
MyStudy	VS	2001	2	DIABP	Diastolic Blood Pres...	70
MyStudy	VS	2001	3	PULSE	Pulse Rate	62
MyStudy	VS	2001	4	WEIGHT	Weight	88
MyStudy	VS	2001	5	BMI	Body Mass Index	22

We also see that something is not entirely ok. In the ODM, it was defined that the maximum length (as text) is 3, but we see that there are values that are of length 4. Although this can be automatically corrected when generating the SAS-XPT files, it is not a bad idea to set the maximum length for VSORRES to 4 already now in the underlying define.xml.

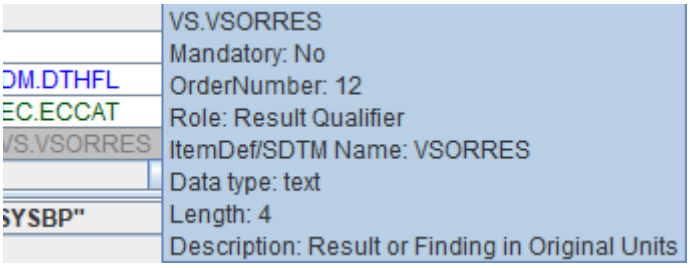
In order to do so, select the cell VS.VSORRES and then use the menu "Edit – SDTM Variable Properties", or use the shortcut "Ctrl-E". This leads to:



Checking the box "New Length" allows us to set the new maximum length to 4:



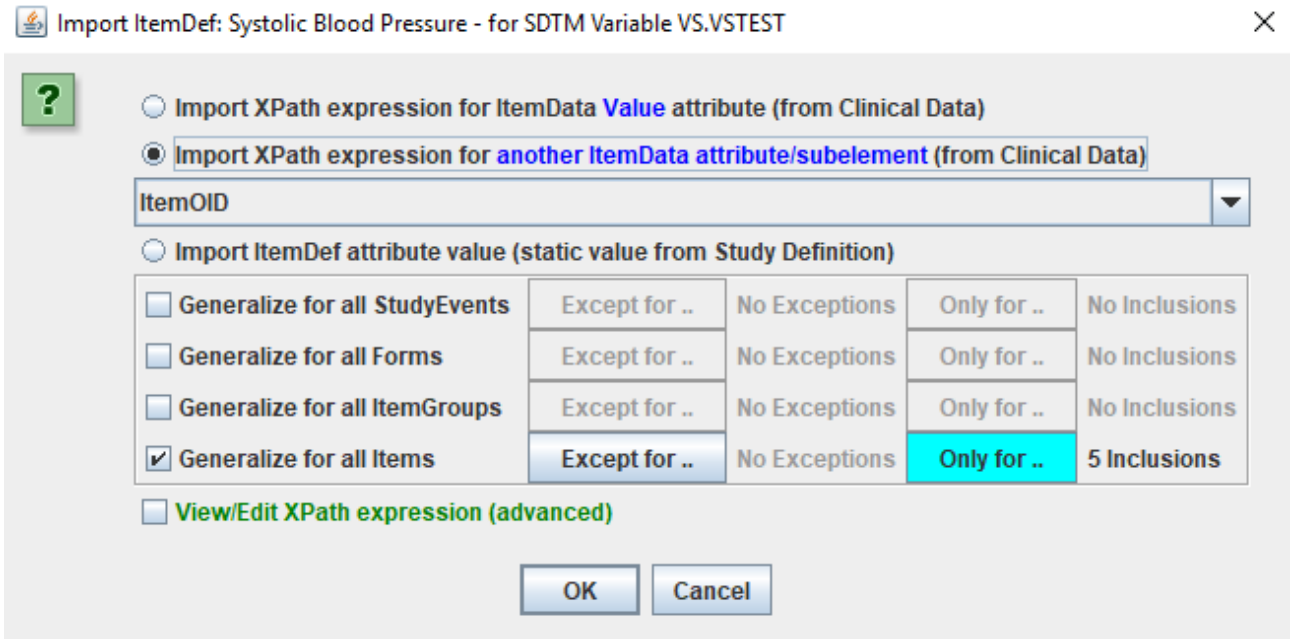
Clicking "OK" updates the variable properties.
When we then hold the mouse over the cell "VS.VSTESTCD" we see:



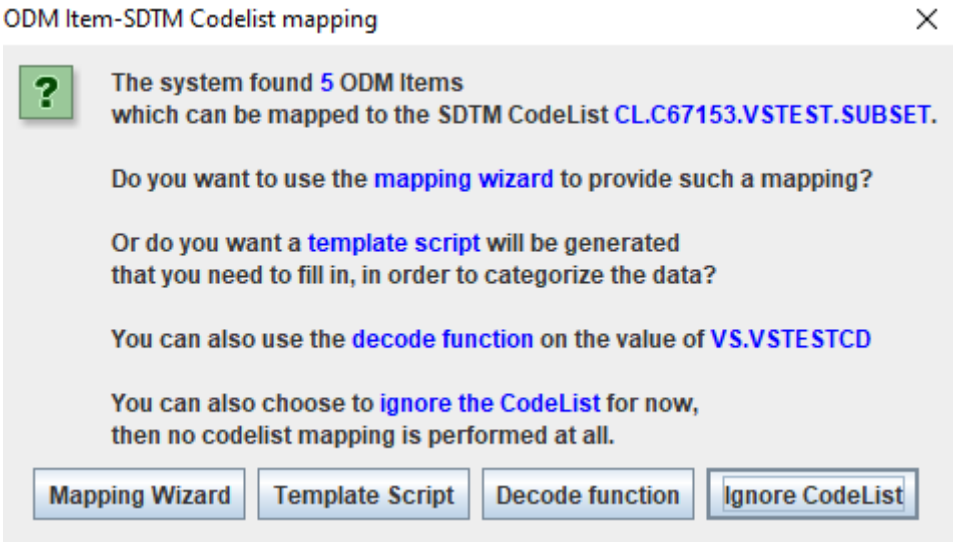
And see that the maximum length has indeed been updated.

Generating the mapping for VSTEST

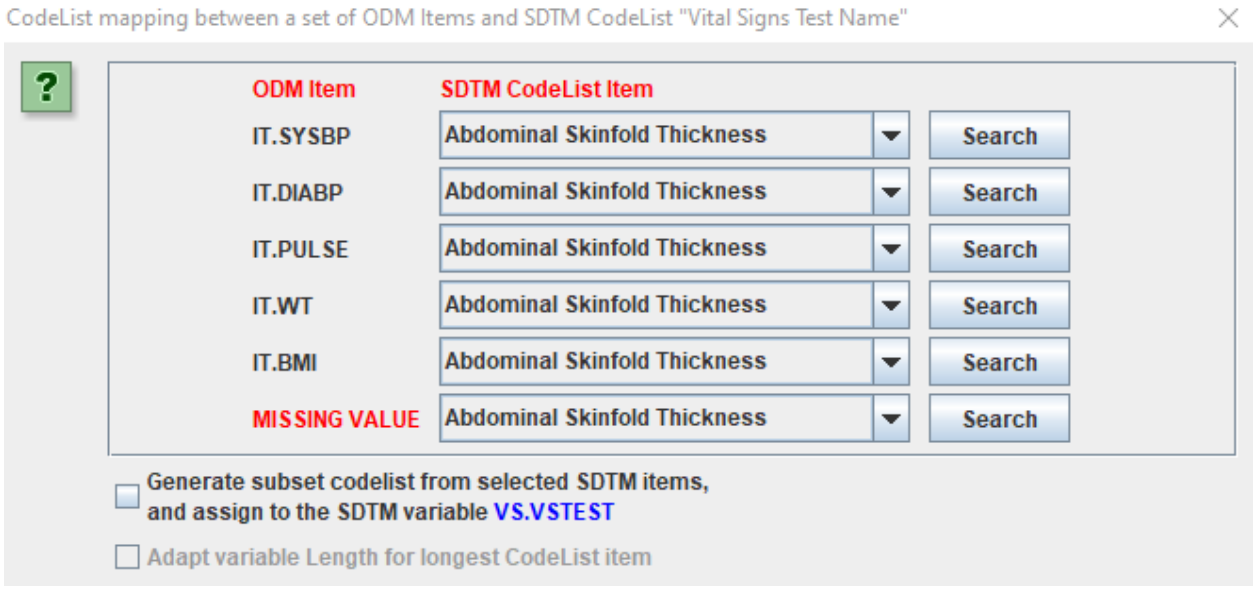
If we do not want to use the new feature of generating the mapping for VSTEST simultaneously with the mapping for VSTESTCD, generating the mapping for VSTEST is almost identical to the one for VSTESTCD.
Drag-and-drop one of the vital sign test entries from the ODM tree to the VS.VSTEST cell, then select to use the "ItemOID" from the clinical data:



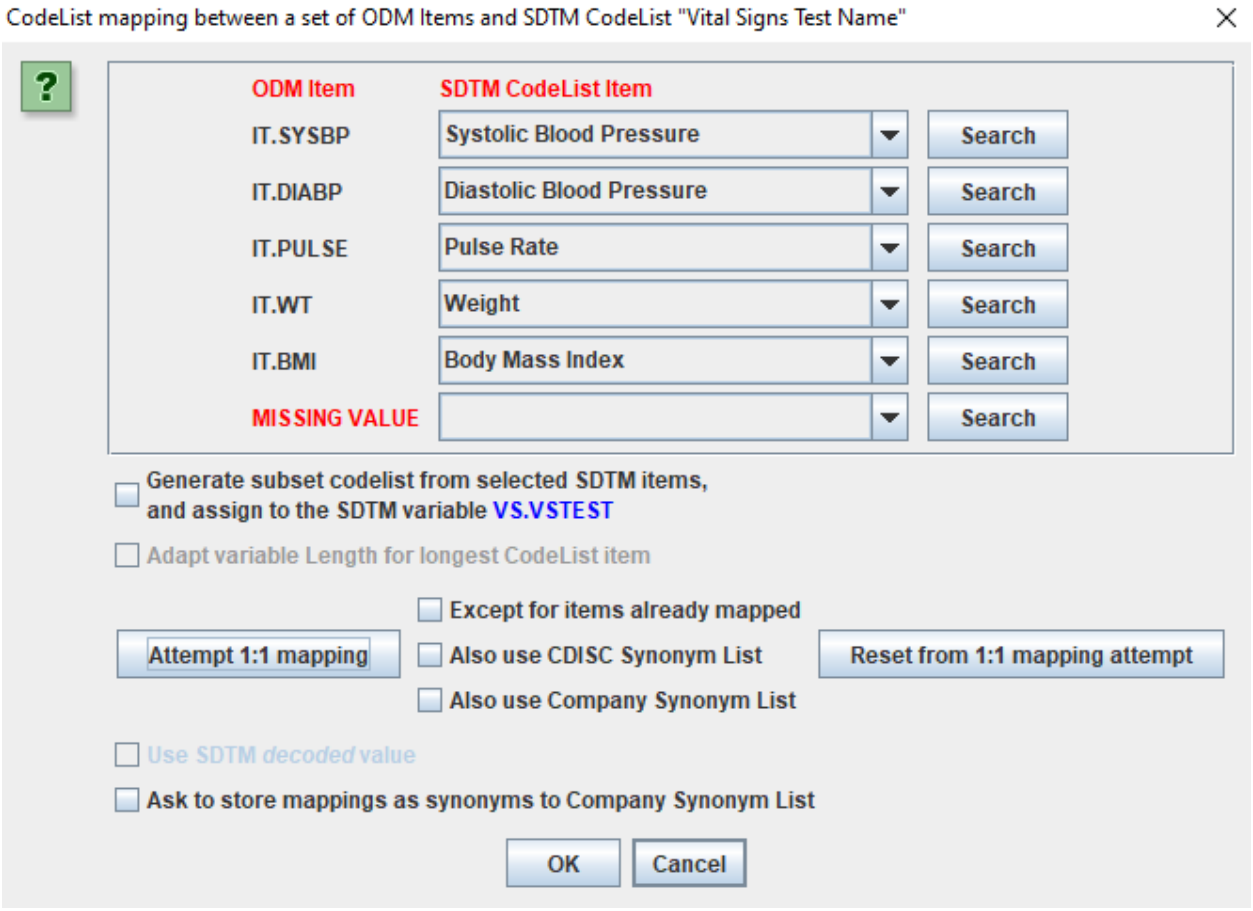
The system still knows about our 5 different vital sign test, so we can just continue, leading to:



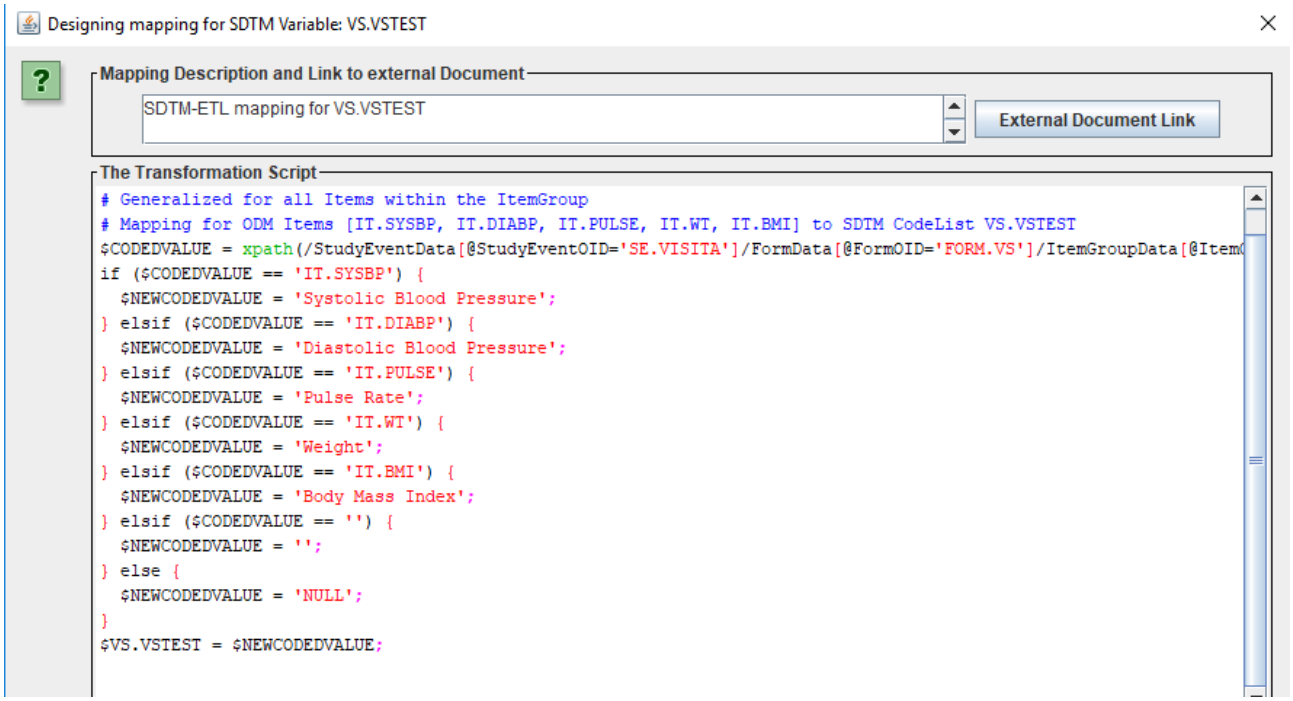
We can once again use the "mapping wizard", leading to:



Which can then easily be completed, using one of the methods described before, to:



Which then automatically generates the mapping script:



And after testing using Transform – Generate Transformation …", leads to:

SDTM Tables

MyStudy:DM | MyStudy:EC | MyStudy:VS

STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSORRES
MyStudy	VS	1001	1	SYSBP	Systolic Blood Pressure	100
MyStudy	VS	1001	2	DIABP	Diastolic Blood Pressure	70
MyStudy	VS	1001	3	PULSE	Pulse Rate	62
MyStudy	VS	1001	4	WEIGHT	Weight	88
MyStudy	VS	1001	5	BMI	Body Mass Index	23
MyStudy	VS	1001	6	SYSBP	Systolic Blood Pressure	108
MyStudy	VS	1001	7	DIABP	Diastolic Blood Pressure	74
MyStudy	VS	1001	8	PULSE	Pulse Rate	65
MyStudy	VS	1001	9	SYSBP	Systolic Blood Pressure	107
MyStudy	VS	1001	10	DIABP	Diastolic Blood Pressure	75
MyStudy	VS	1001	11	SYSBP	Systolic Blood Pressure	105
MyStudy	VS	1001	12	DIABP	Diastolic Blood Pressure	76
MyStudy	VS	1001	13	PULSE	Pulse Rate	63
MyStudy	VS	1001	14	WEIGHT	Weight	88.2
MyStudy	VS	1001	15	BMI	Body Mass Index	23
MyStudy	VS	1001	16	SYSBP	Systolic Blood Pressure	108
MyStudy	VS	1001	17	DIABP	Diastolic Blood Pressure	74
MyStudy	VS	2001	1	SYSBP	Systolic Blood Pressure	100
MyStudy	VS	2001	2	DIABP	Diastolic Blood Pressure	70
MyStudy	VS	2001	3	PULSE	Pulse Rate	62

Nice! We are making good progress …

Generating other mappings for Vital Signs - VSORRESU

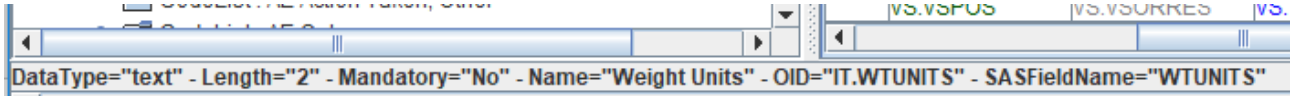
If we now look at our table again, we see

DM.DTHDTC	DM.DTHFL	DM.SITEID	DM.INVID	DM.INVNAM	DM.BRTHDTC	DM.AGE	DM.AGEU
EC.ECMOOD	EC.ECCAT	EC.ECSCAT	EC.ECPRESP	EC.ECOCCUR	EC.ECDOSE	EC.ECDOSTXT	EC.ECDO
VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC	VS.VSSTRESN	VS.VSSTRESU	VS.VSSTAT	VS.VSRE

That we still need to add the information for "VSORRESU" (original result units) and for VSSTRESC, VSTRESN (standardized results, character and numeric) and VSSTRESU (standardized result units). We will start with VSORRESU (original units).

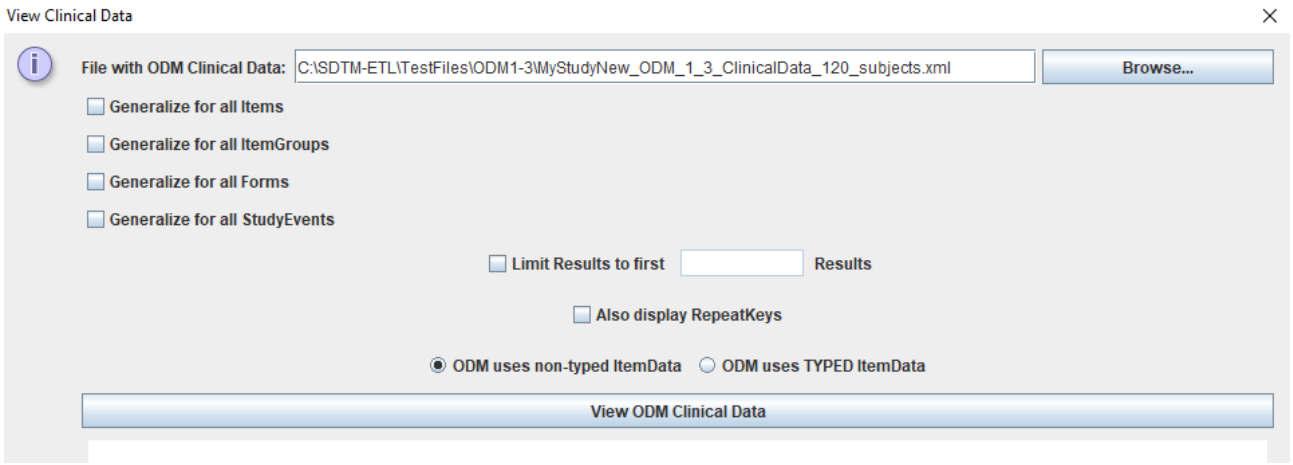
We see at the ODM side that for systolic and diastolic blood pressure, the units are not provided. The reason is that the value is always mmHg (UCUM notation[5]: mm[Hg]). For "Weight" however, the field in the EDC system was essentially a free text field, as we can see from the ODM:

And when we ask for the associated codelist using the menu "View – Item CodeList Details" we get … nothing. We can also see that there was no codelist, by selecting "Weight Units" and inspecting the line at the bottom:

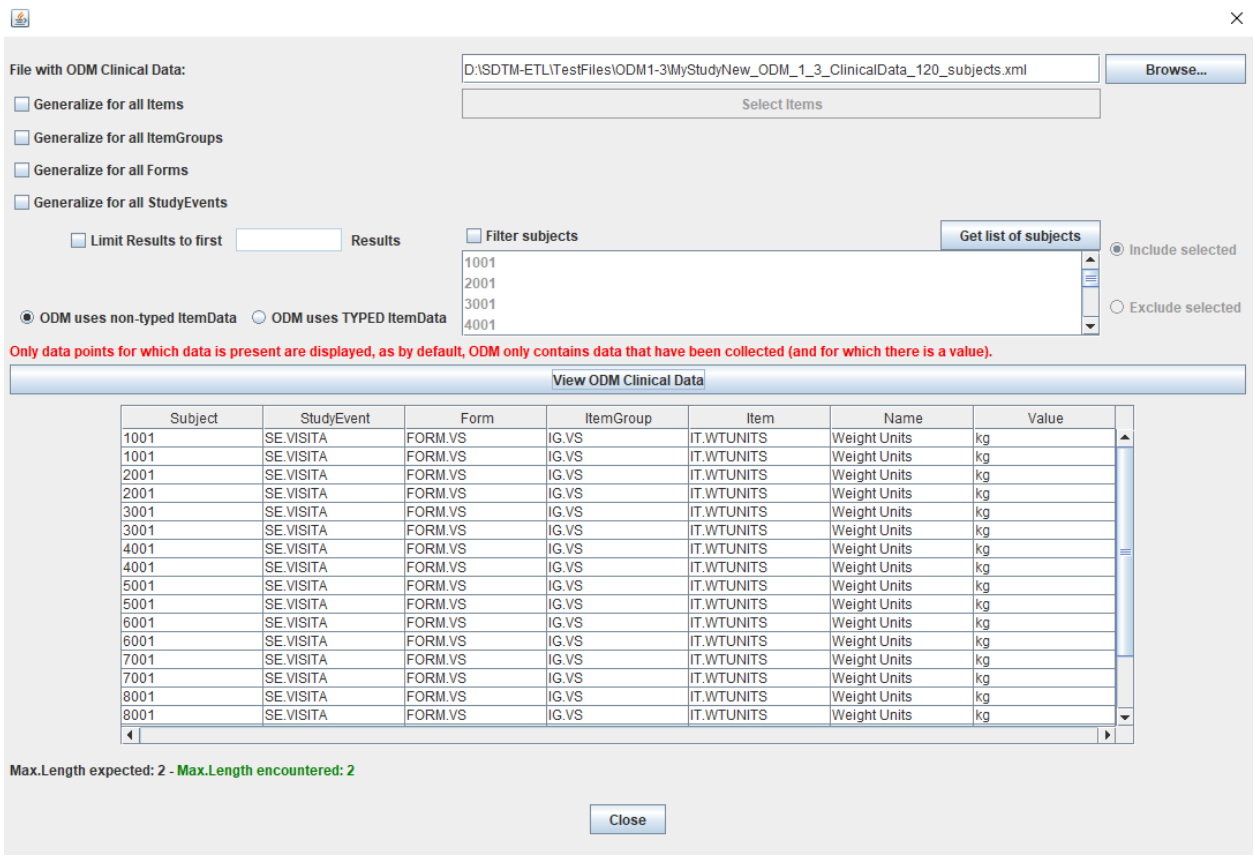


So, what were the reported weight units? In order to see them, use the menu "View – ODM ClinicalData". This leads to:

[5] Unfortunately, CDISC and FDA do not allow units in UCUM notation yet, which would simplify many things, such as automated unit conversions.



And clicking "View ODM Clinical Data" then shows us a table:



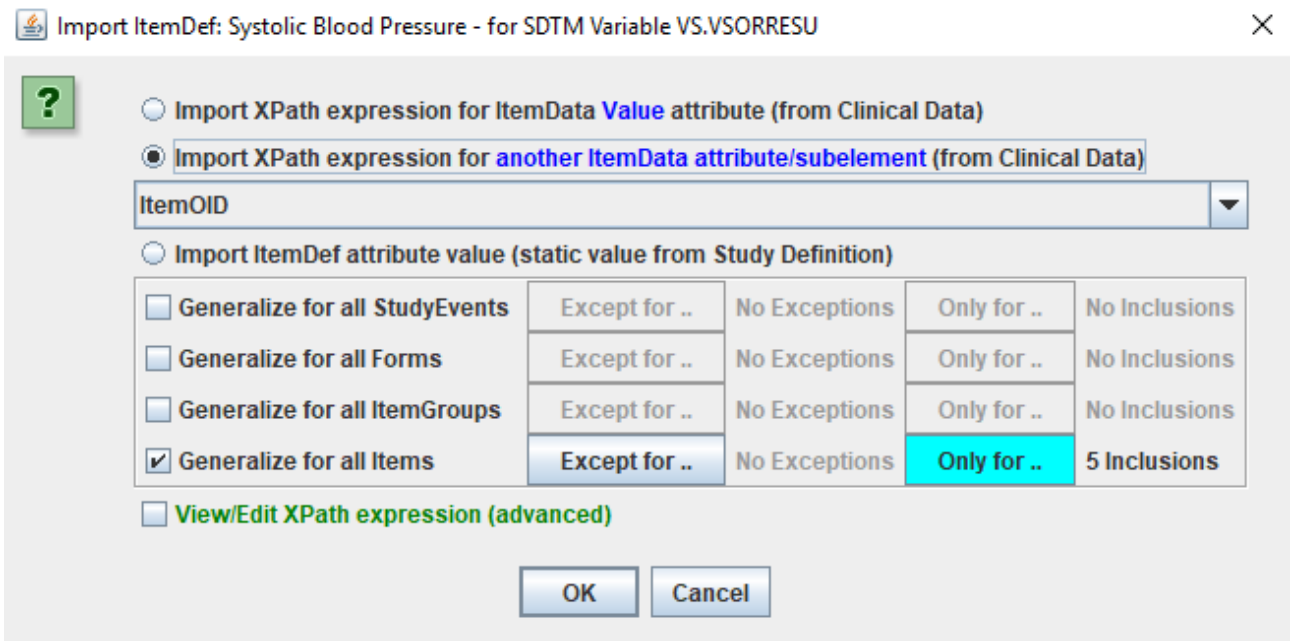
And we see that "kg" has always been used. If we had found a multitude of different units here, we would have had to add some extra code, e.g. taking into account the different ways of writing "kilogram" and "pounds".

Inspecting the collected data is always a good idea when the item was a free text item and needs to be standardized.

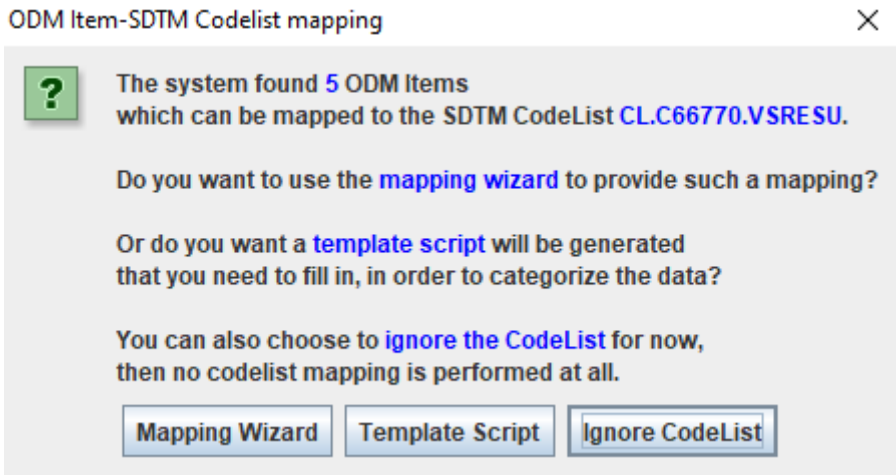
In order to provide the mapping for VSORRESU, again drag-and-drop one of the vital signs entries from the ODM to the VS.VSORRESU cell, and select to import the "ItemOID"[6], leading to:

---

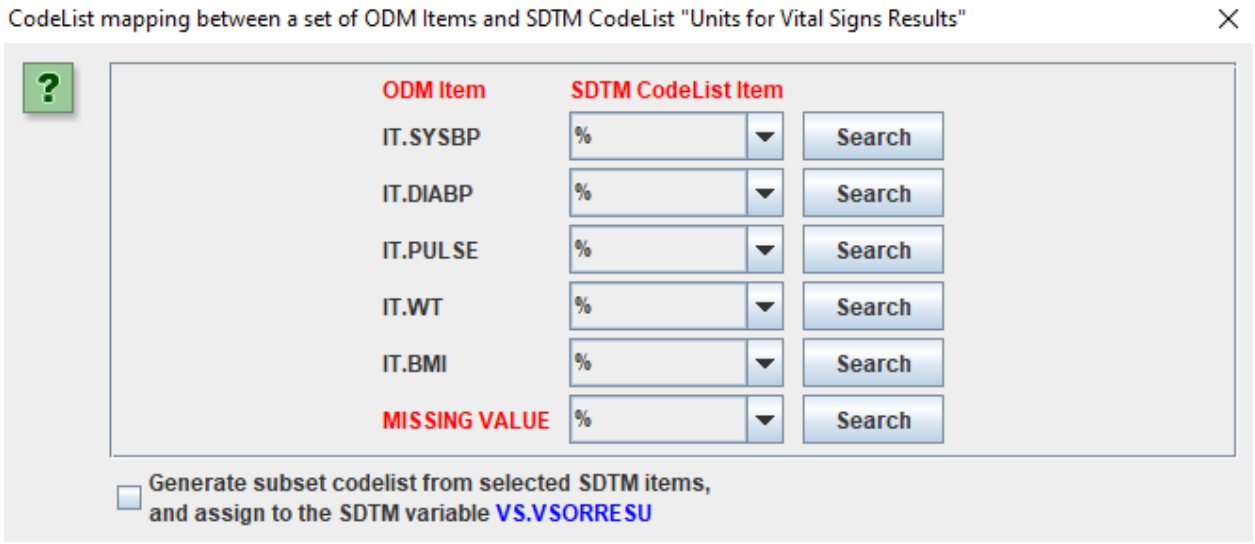
[6] Depending on the situation, the system may first propose "MeasurementUnit", which is a way of providing units of measurements in ODM that is often used. In this case however, it wasn't.



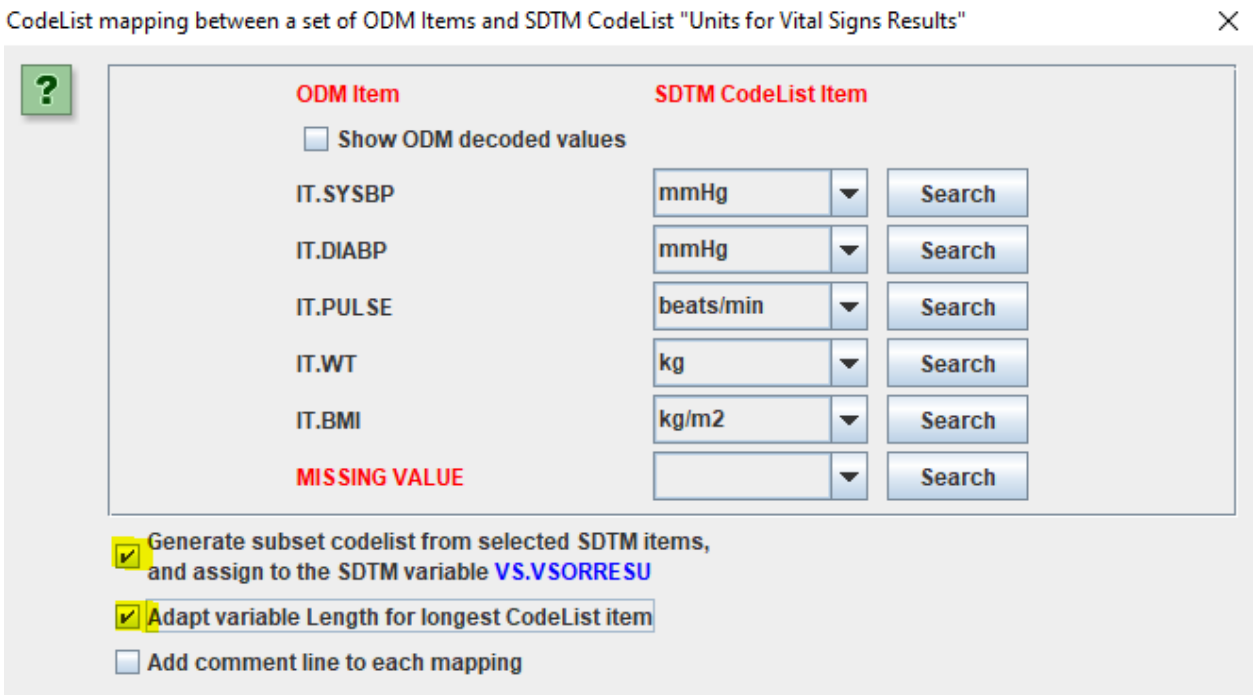
Continued by the dialog:



And when clicking "Mapping Wizard", the mapping wizard is displayed:

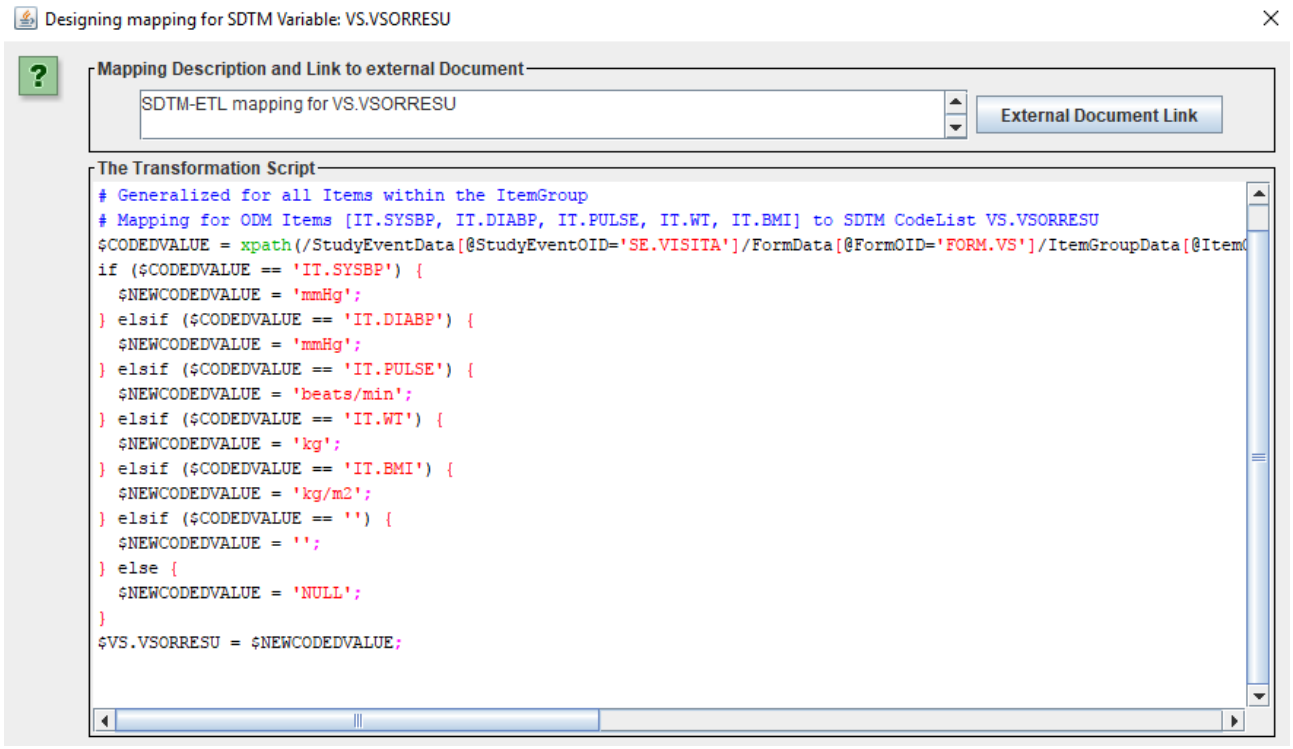


Which we can easily complete, as typing the first character already leads us to the unit value that we need. The result will be:



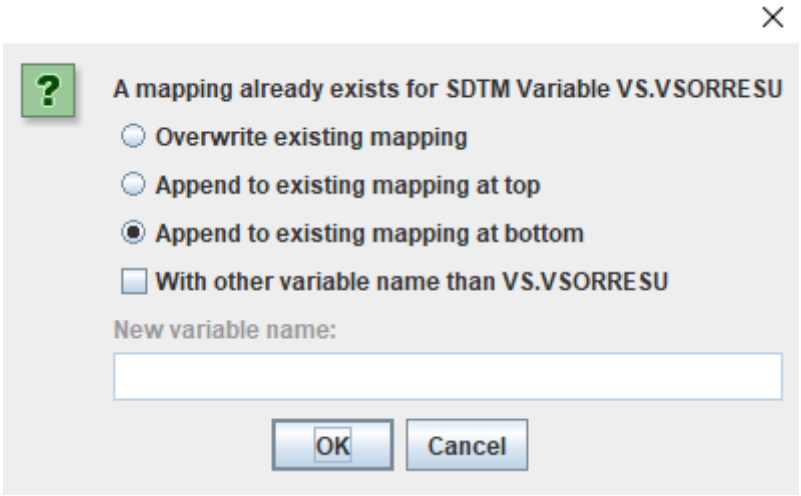
Also here it can be of advantage to check the checkbox "Generate subset codelist", as the "unit" codelist for Vital Signs is rather large, and we probably do not want to submit the whole list.

This is then leading to the mapping script:

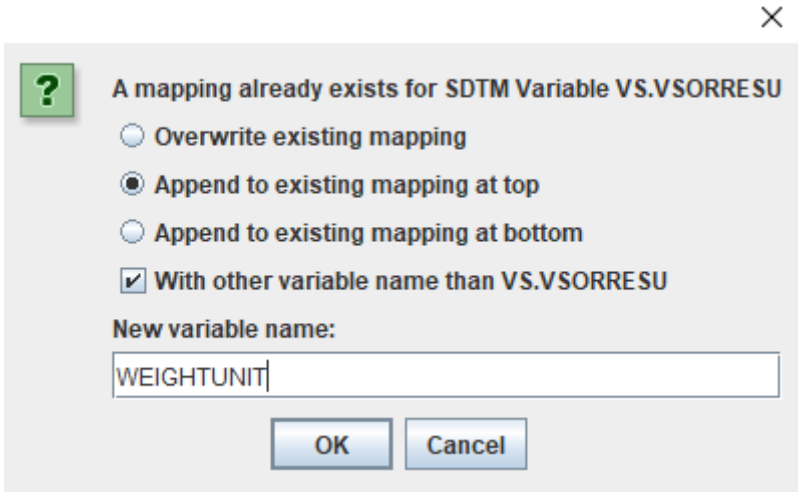


Suppose now that we want to read the value for "weight unit" from the ODM, and add the value here. We can easily do so by adding some extra scripting. We first drag-and-drop "Weight Units" to

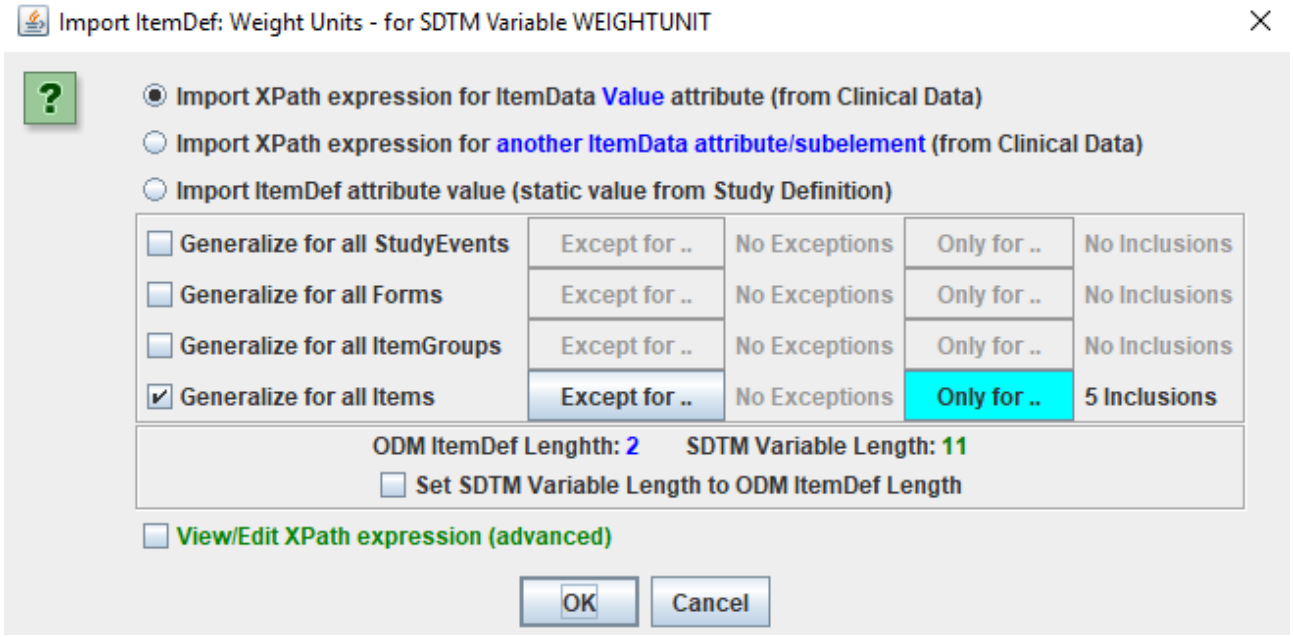
"VSORRESU". As there is already a mapping present, the system asks us:



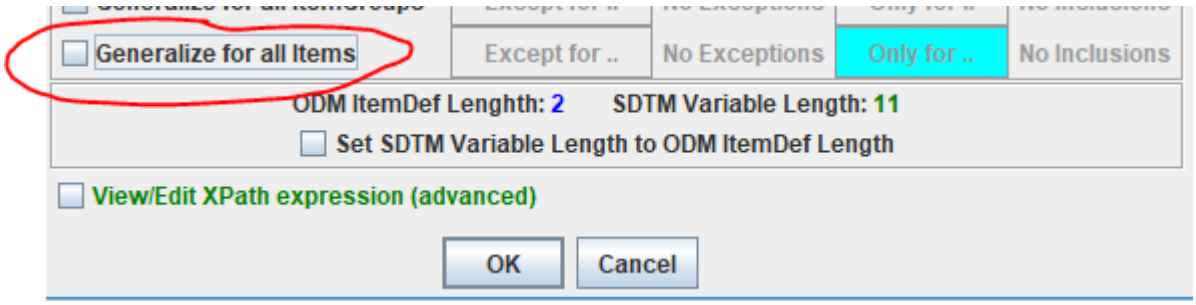
In our case, it is a good idea to append the mapping to the top of our script, so we select "Append to existing mapping at top". We can also give a new name to the new variable that is then created, e.g.:



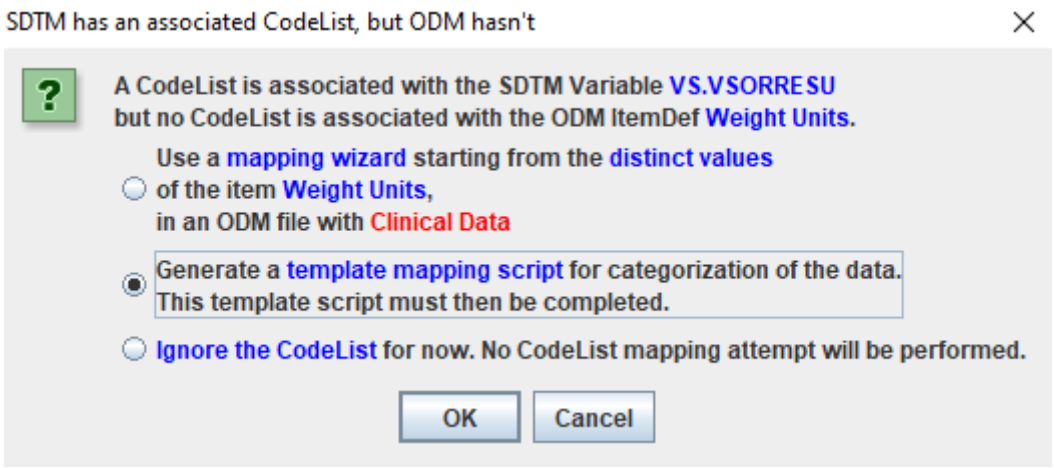
After clicking "OK", we get:



However, this time, we only want to import the value for "Weight Units", so do not want to "generalize for all items" (within the group), so we must unselect that checkbox:



And after clicking "OK" we get:



Allowing us to choose between starting a mapping wizard that takes the distinct values of "Weight Units" from the clinical data, generating a template mapping script that needs to be completed, and just ignoring the codelist for now. The first can be very useful, but also tricky when the clinical data is not complete yet.
Which will first try to generate a template script that we will need to complete. Clicking "OK" leads to:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.WTUNITS
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$WEIGHTUNIT = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/Item

if( ) {
     $VS.VSORRESU = '%';
} elsif( ) {
     $VS.VSORRESU = 'beats/min';
} elsif( ) {
     $VS.VSORRESU = 'breaths/min';
} elsif( ) {
     $VS.VSORRESU = 'C';
} elsif( ) {
     $VS.VSORRESU = 'cm';
} elsif( ) {
     $VS.VSORRESU = 'F';
} elsif( ) {
     $VS.VSORRESU = 'g';
} elsif( ) {
     $VS.VSORRESU = 'in';
```

Such a "template" script is extremely useful when one needs to "categorize" collected data that comes as free text into categories or coded values, as is the case here.
In our case however, it is only about "weight", so we can remove that template script again, as we will just "pick up" the value as it was collected, i.e. "ignoring the codelist". This leads to:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.WTUNITS
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$WEIGHTUNIT = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Item

# Mapping using ODM element ItemData with ItemOID IT.SYSBP - value from attribute ItemOID
# Generalized for all Items within the ItemGroup
# Mapping for ODM Items [IT.SYSBP, IT.DIABP, IT.PULSE, IT.WT, IT.BMI] to SDTM CodeList VS.VSORRESU
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Item
if ($CODEDVALUE == 'IT.SYSBP') {
  $NEWCODEDVALUE = 'mmHg';
} elsif ($CODEDVALUE == 'IT.DIABP') {
  $NEWCODEDVALUE = 'mmHg';
} elsif ($CODEDVALUE == 'IT.PULSE') {
  $NEWCODEDVALUE = 'beats/min';
} elsif ($CODEDVALUE == 'IT.WT') {
  $NEWCODEDVALUE = 'kg';
} elsif ($CODEDVALUE == 'IT.BMI') {
  $NEWCODEDVALUE = 'kg/m2';
} elsif ($CODEDVALUE == '') {
  $NEWCODEDVALUE = '';
```

The first set of lines takes the collected value for weight unit, the other lines "hard-code" the value, as it was not collected, but e.g. prescribed by the protocol.
There is now still one line we need to correct, which is the line in the "if" statements that is about weight units. We change it into:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.WTUNITS
# Using categorization as a CodeList is associated with the SDTM CodeList
# but no CodeList is associated with the ODM data
$WEIGHTUNIT = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Item

# Mapping using ODM element ItemData with ItemOID IT.SYSBP - value from attribute ItemOID
# Generalized for all Items within the ItemGroup
# Mapping for ODM Items [IT.SYSBP, IT.DIABP, IT.PULSE, IT.WT, IT.BMI] to SDTM CodeList VS.VSORRESU
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Item
if ($CODEDVALUE == 'IT.SYSBP') {
  $NEWCODEDVALUE = 'mmHg';
} elsif ($CODEDVALUE == 'IT.DIABP') {
  $NEWCODEDVALUE = 'mmHg';
} elsif ($CODEDVALUE == 'IT.PULSE') {
  $NEWCODEDVALUE = 'beats/min';
} elsif ($CODEDVALUE == 'IT.WT') {
  $NEWCODEDVALUE = $WEIGHTUNIT;
} elsif ($CODEDVALUE == 'IT.BMI') {
  $NEWCODEDVALUE = 'kg/m2';
} elsif ($CODEDVALUE == '') {
  $NEWCODEDVALUE = '';
```

Saying that for "Weight", we will take the collected value, and not the hard-coded value.

Running the scripts then leads to:

SDTM Tables

My Study:DM | My Study:EC | My Study:VS

AIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSORRES	VS.VSORRESU
	1001	1	SYSBP	Systolic Blood Pres...	100	mmHg
	1001	2	DIABP	Diastolic Blood Pre...	70	mmHg
	1001	3	PULSE	Pulse Rate	62	beats/min
	1001	4	WEIGHT	Weight	88	kg
	1001	5	BMI	Body Mass Index	23	kg/m2
	1001	6	SYSBP	Systolic Blood Pres...	108	mmHg
	1001	7	DIABP	Diastolic Blood Pre...	74	mmHg
	1001	8	PULSE	Pulse Rate	65	beats/min
	1001	9	SYSBP	Systolic Blood Pres...	107	mmHg
	1001	10	DIABP	Diastolic Blood Pre...	75	mmHg
	1001	11	SYSBP	Systolic Blood Pres...	105	mmHg
	1001	12	DIABP	Diastolic Blood Pre...	76	mmHg
	1001	13	PULSE	Pulse Rate	63	beats/min
	1001	14	WEIGHT	Weight	88.2	kg
	1001	15	BMI	Body Mass Index	23	kg/m2
	1001	16	SYSBP	Systolic Blood Pres...	108	mmHg
	1001	17	DIABP	Diastolic Blood Pre...	74	mmHg
	2001	1	SYSBP	Systolic Blood Pres...	100	mmHg
	2001	2	DIABP	Diastolic Blood Pre...	70	mmHg
	2001	3	PULSE	Pulse Rate	62	beats/min
	2001	4	WEIGHT	Weight	88	kg
	2001	5	BMI	Body Mass Index	22	kg/m2

Later, we will also learn how to create "Value lists" for VSORRES for the individual tests. See the tutorial "Working with the WhereClause in define.xml 2.0". With SDTM-ETL, generating value lists is extremely easy, whereas it is a nightmare with some of our competitor software offerings.

Remark that "ignoring the codelist" is only a good idea when one is sure that the value for the unit already follows the CDISC notation, which is often true in the "paper world", but surely not when the data is collected electronically or from electronic health records, where the much better UCUM notation is used. Unfortunately, CDISC still does not allow UCUM notation[7] ("not invented here syndrome")

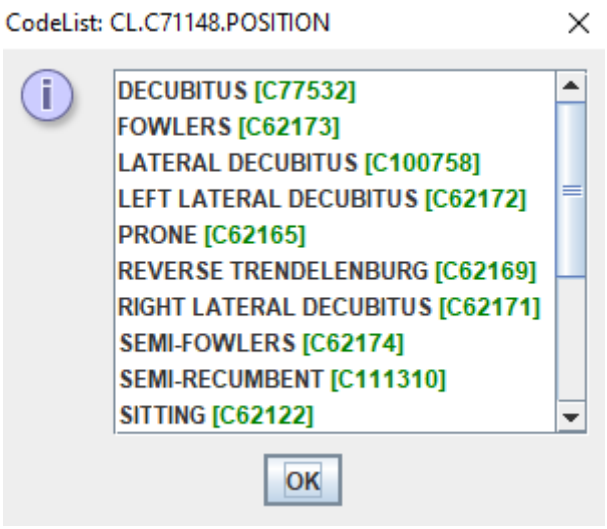
[7] There is however some overlap between UCUM notation and CDISC notation, especially for concentrations.

Creating the mapping for VSPOS

VSPOS (Vital Signs Position) is a "permissible" variable, meaning that one can omit it when no data was collected for it, which is the case here. We will however still add it to demonstrate the use of "Origin" in define.xml.

Suppose that it was stated in the protocol that the vital sign tests blood pressure and pulse must be taken in sitting position. For "weight", "height" (which is not collected here) this of course doesn't make sense. So we will create a small mapping script in which we state that in the former case, the value of VSPOS is "SITTING", whereas it is empty ("null") in the latter case.

Let us first have a look at the allowed values for VSPOS.
In order to do this, select the cell VS.VSPOS and use the menu "View - SDTM Associated CodeList". The following dialog is displayed:



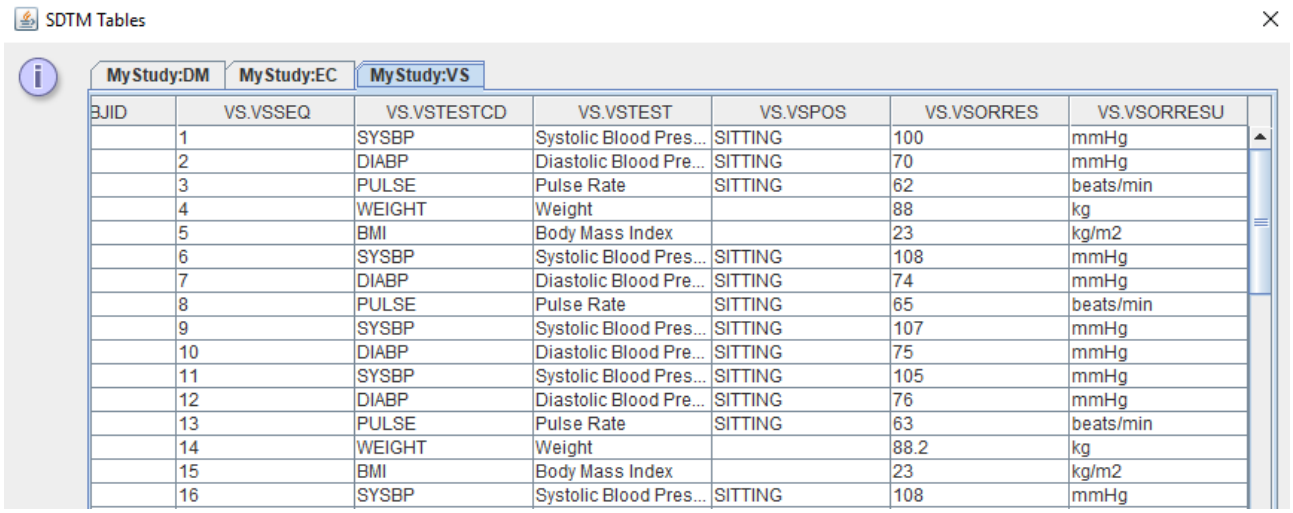
In SDTM-ETL, one can always reuse variables in the same row that come before the current variable and for which a mapping exists. As VSPOS comes after VSTESTCD, we can easily reuse it in a set of one or more "if" statements.

So we can write:

The Transformation Script

```
$VS.VSPOS = '';
if($VS.VSTESTCD='DIABP' or $VS.VSTESTCD='SYSBP' or $VS.VSTESTCD='PULSE' ) {
     $VS.VSPOS = 'SITTING';
}  else {
     $VS.VSPOS = '';
}
```

leading to:

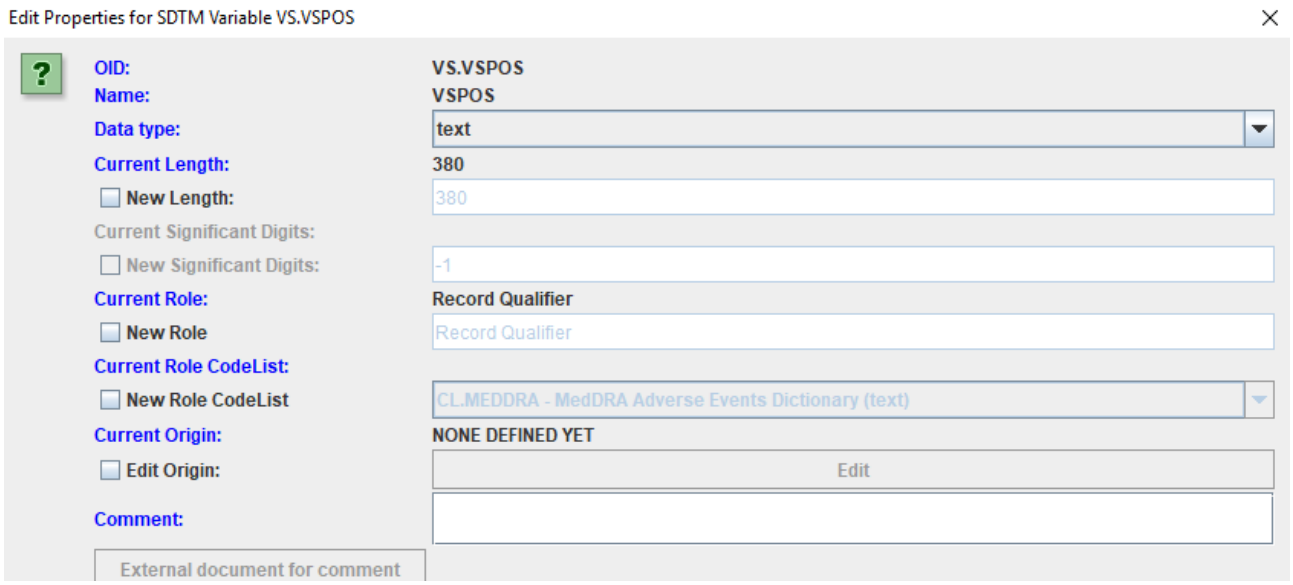


BJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU
	1	SYSBP	Systolic Blood Pres...	SITTING	100	mmHg
	2	DIABP	Diastolic Blood Pre...	SITTING	70	mmHg
	3	PULSE	Pulse Rate	SITTING	62	beats/min
	4	WEIGHT	Weight		88	kg
	5	BMI	Body Mass Index		23	kg/m2
	6	SYSBP	Systolic Blood Pres...	SITTING	108	mmHg
	7	DIABP	Diastolic Blood Pre...	SITTING	74	mmHg
	8	PULSE	Pulse Rate	SITTING	65	beats/min
	9	SYSBP	Systolic Blood Pres...	SITTING	107	mmHg
	10	DIABP	Diastolic Blood Pre...	SITTING	75	mmHg
	11	SYSBP	Systolic Blood Pres...	SITTING	105	mmHg
	12	DIABP	Diastolic Blood Pre...	SITTING	76	mmHg
	13	PULSE	Pulse Rate	SITTING	63	beats/min
	14	WEIGHT	Weight		88.2	kg
	15	BMI	Body Mass Index		23	kg/m2
	16	SYSBP	Systolic Blood Pres...	SITTING	108	mmHg

If the "position" was collected, one can of course generate the mapping using simple drag-and-drop, and using the codelist mapping wizard when necessary.

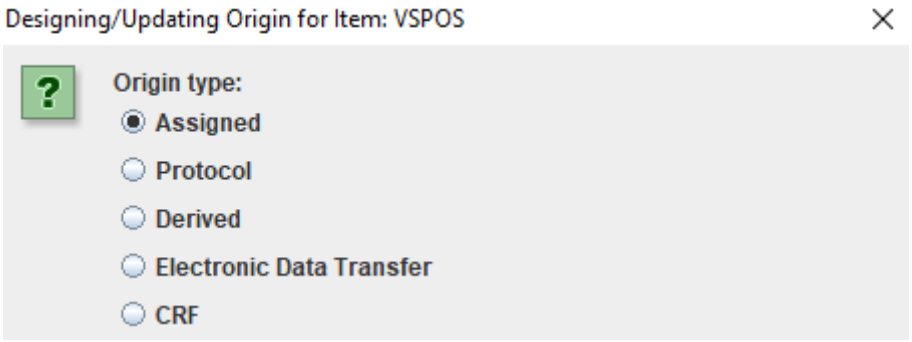
As the subject's position for vital signs measurement was defined in the protocol, we should add this information in the (underlying) define.xml. We can do this at the variable level (VSPOS) or at the value level, i.e. individually for SYSBP, DIABP, WEIGHT, … The mechanism is however essentially the same.
Just for the sake of the tutorial, we will define the "origin" at the variable level here.

Select the VS.VSPOS cell, and then use the menu "Edit – SDTM Variable Properties". The following dialog is displayed:

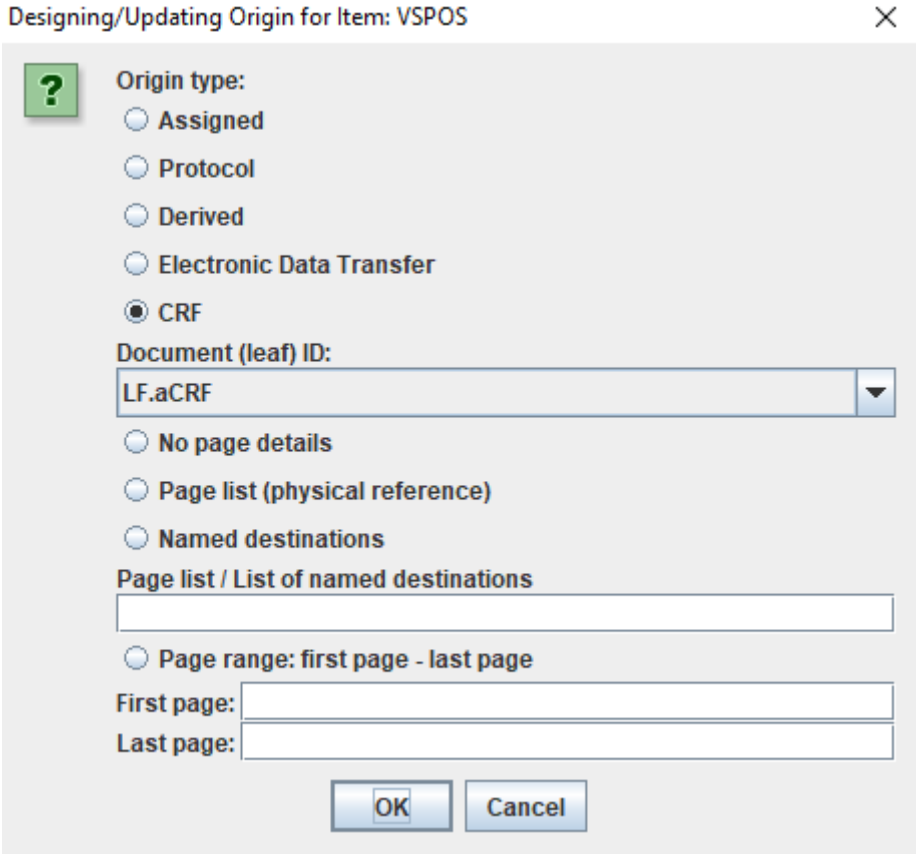


One sees that no "origin" has been defined yet.
So, check the checkbox "Edit Origin", and then click the button "Edit" on the right of it. The following dialog is displayed[8]:

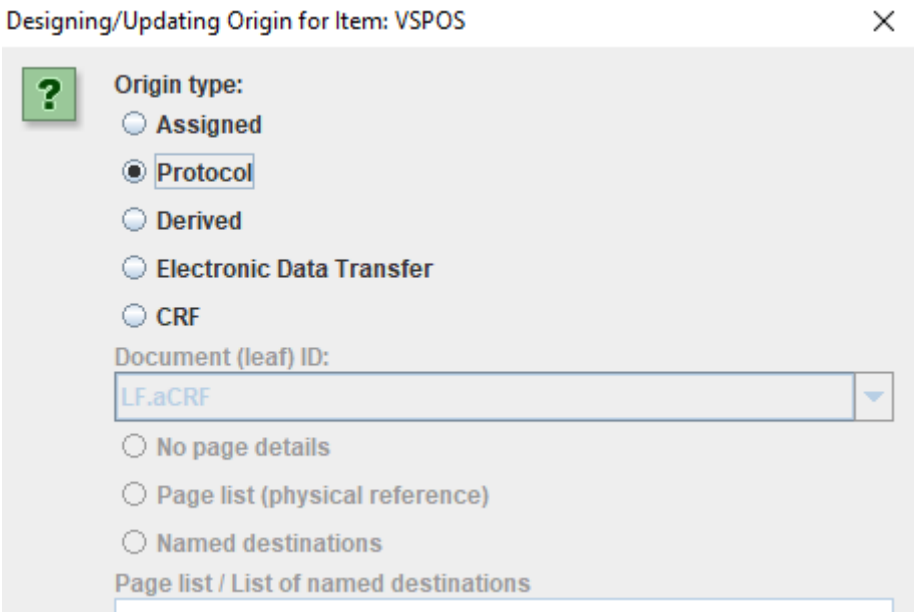
[8] In the case of Define-XML 2.1, the dialog will be different.



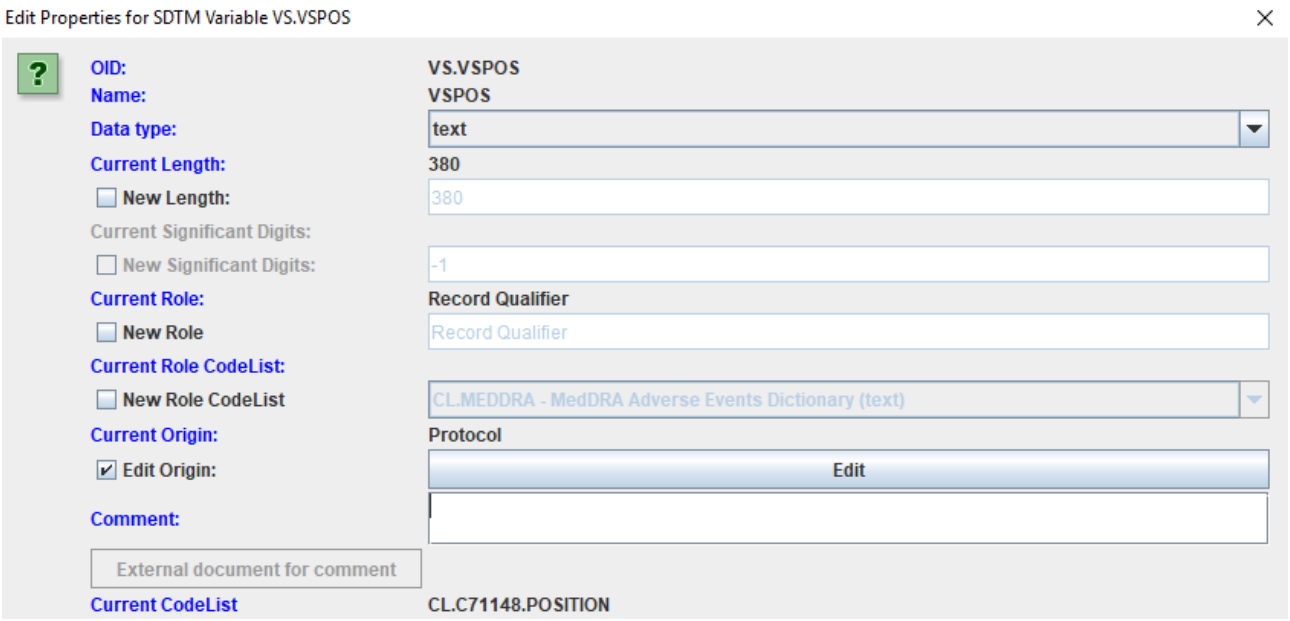
If we select "CRF", more fields become available:



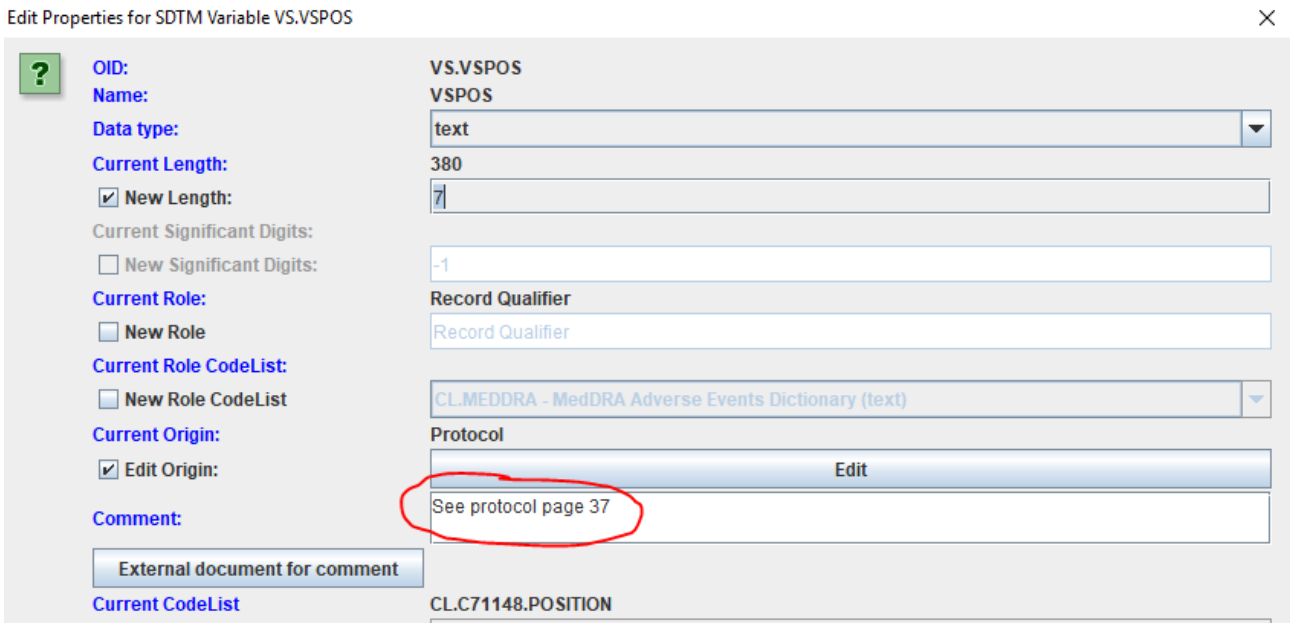
If the subject's position would have been collected, we would set the "origin" to "CRF", and then add the page numbers in the annotated CRF. If the subject's position would have been assigned by us (not recommended), we would set the origin to "Assigned". In our case however, as it comes from the protocol, we select the radiobutton "Protocol":



One also sees that the fields about page numbers are then disabled. Click "OK" to confirm. This leads to the prior dialog:

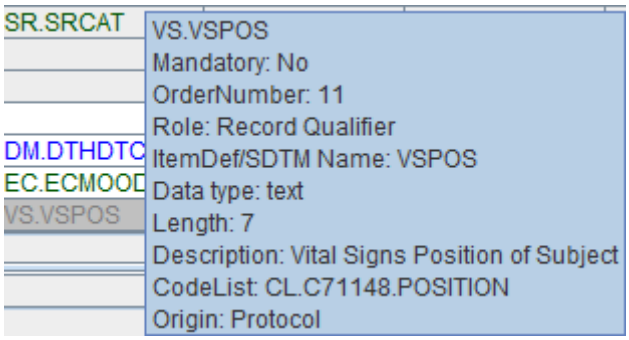


We also see that somehow, the maximum length was set to 380, which just doesn't make sense. We set it to 7 which is the length of the word "SITTING".
We can also add a comment to the "origin" statement, and even make a reference to an external document if our comment is too long for keeping it in the define.xml. Just let us add a short statement that one can find the instruction about taking the vital signs in a sitting position on page 37 of the protocol. For example:



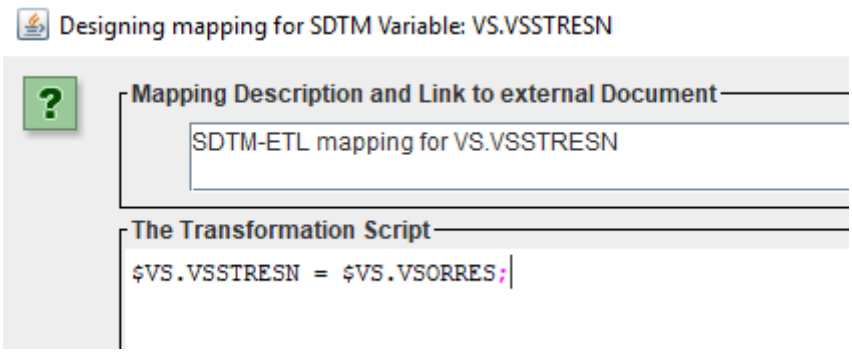
Now click "OK" to confirm.

When we move the mouse over the cell "VS.VSPOS" a tooltip is displayed:



Mappings for VSSTRESN, VSSTRESC and VSSTRESU

In our case, all collected values are already in standard units, so there is not much to do.
Also, as VSSTRESN, VSSTRESC and VSSTRESU come after VSORRES and VSORRESU, we can reuse the values. For example for VSSTRESN, we can simply state:



And similar for VSSTRESC and VSSTRESU (which can be copied from VSORRESU).

Just for the sake of the tutorial, suppose that we also had collected "frame size" (VSTESTCD=FRMSIZE) which has values "SMALL", "MEDIUM" and "LARGE" and thus no units, and as it is not a numeric value, VSSTRESN ("Numeric Result/Finding in Standard Units"), we could easily write the script as:

The Transformation Script

```
# When the test code is FRMSIZE, there is no numeric value
if($VS.VSTESTCD != 'FRMSIZE') {
     $VS.VSSTRESN = $VS.VSORRES;
}  else {
     $VS.VSSTRESN = '';
}
```

With a similar script for VSSTRESU

In SDTM-ETL, we usually do not have to care whether the value is numeric or text as long as we do not perform calculations[9]. This is taken care of by the software itself.

Just as an example, let us suppose that some of the weight data was collected in (US) pounds. We know that 1 pound (UCUM notation: [lb_av]) = 0.4536 kg. If CDISC would allow UCUM notation, this conversion factor and the conversion itself could be done by a RESTful web service as e.g. delivered by the National Library of Medicine. However, this is not the case yet, and we need to program this manually. For VSSTRESN the script could then looks like:

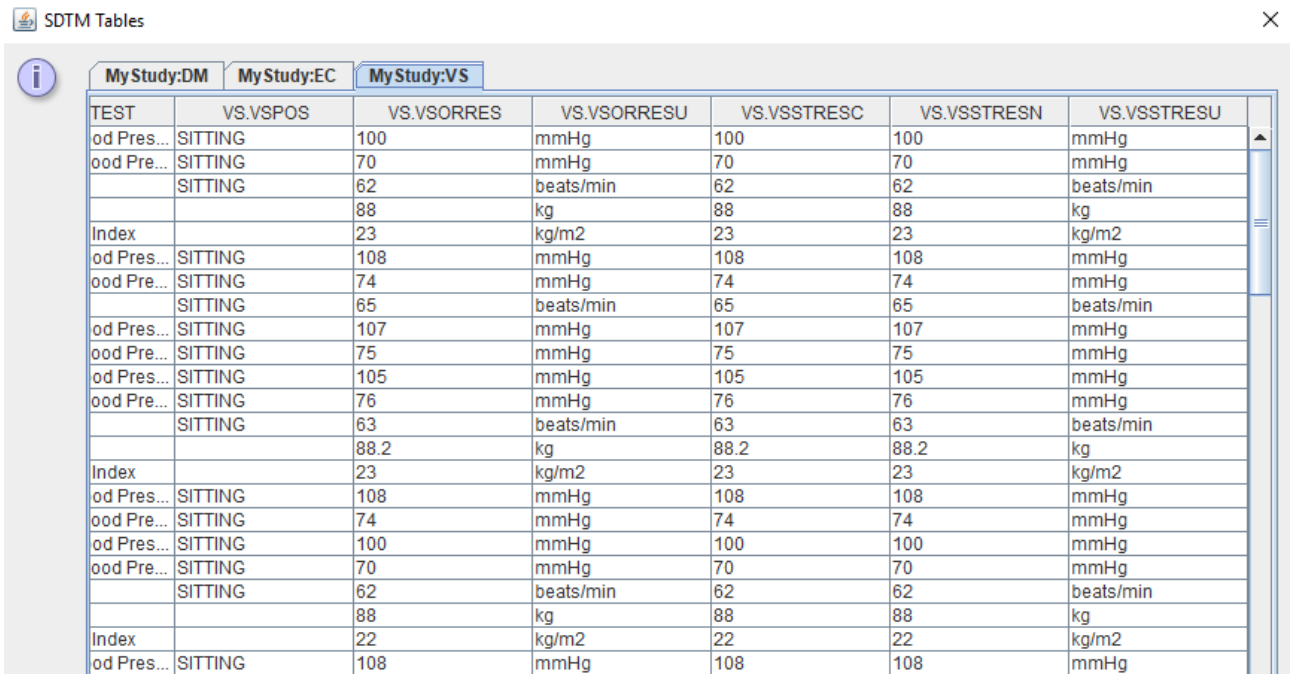
The Transformation Script

```
# For WEIGHT in pounds, a conversion is necessary
# For FRMSIZE, there is no numeric value
if($VS.VSTESTCD = 'FRMSIZE') {
     $VS.VSSTRESN = '';
}  elsif($VS.VSTESTCD = 'WEIGHT' and $VS.VSORRESU = 'pounds') {
     $VS.VSSTRESN = 0.4536 * number($VS.VSORRES);
}  else {
     $VS.VSSTRESN = $VS.VSORRES;
}
```

Remark the use of the "number()" function to ensure that the value is treated as numeric.
Also remark that in "if"- statements, the comparator "==" as well as "=" can be used. Both are treated equal.

The result when executing the transformations then is:

[9] The use of having two variables -STRESC and -STRESN is an artefact of the mandated use of SAS Transport 5 by the regulatory authorities. It is something that is not of the 21st century anymore.



TEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC	VS.VSSTRESN	VS.VSSTRESU
od Pres...	SITTING	100	mmHg	100	100	mmHg
ood Pre...	SITTING	70	mmHg	70	70	mmHg
	SITTING	62	beats/min	62	62	beats/min
		88	kg	88	88	kg
Index		23	kg/m2	23	23	kg/m2
od Pres...	SITTING	108	mmHg	108	108	mmHg
ood Pre...	SITTING	74	mmHg	74	74	mmHg
	SITTING	65	beats/min	65	65	beats/min
od Pres...	SITTING	107	mmHg	107	107	mmHg
ood Pre...	SITTING	75	mmHg	75	75	mmHg
od Pres...	SITTING	105	mmHg	105	105	mmHg
ood Pre...	SITTING	76	mmHg	76	76	mmHg
	SITTING	63	beats/min	63	63	beats/min
		88.2	kg	88.2	88.2	kg
Index		23	kg/m2	23	23	kg/m2
od Pres...	SITTING	108	mmHg	108	108	mmHg
ood Pre...	SITTING	74	mmHg	74	74	mmHg
od Pres...	SITTING	100	mmHg	100	100	mmHg
ood Pre...	SITTING	70	mmHg	70	70	mmHg
	SITTING	62	beats/min	62	62	beats/min
		88	kg	88	88	kg
Index		22	kg/m2	22	22	kg/m2
od Pres...	SITTING	108	mmHg	108	108	mmHg

Generating the mappings for VSDTC and VSDY

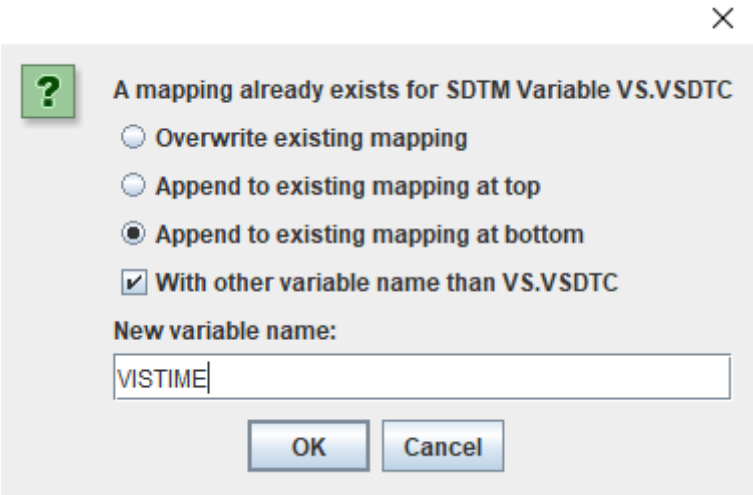
For VSDTC (Vital Signs Date/Time of collection), we can simply drag-and-drop from the ODM item "Vital Signs DateTime" or by concatenating the values of "Vital Signs Date" and "Vital Signs Time". Also here, first displaying the collected clinical data is often of help.

For the case, we need to concatenate date and time, we first drag-and-drop from "Vital Signs Date" to VS.VSDTC, and manually change the variable name in e.g. "VISDATE":

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VISDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='F(
```

Then we drag-and-drop from "Vital Signs Time" to VS.VSDTC. As there is already a mapping present, the systems asks us what to do. We state that we want to append it to the existing mapping with another variable name, e.g. "VISTIME":



Leading to:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VISDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS'
# Mapping using ODM element ItemData with ItemOID IT.VSTIM
$VISTIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS'
```

In case we are not sure that the time part has always been collected, we can take care of that in the script. For example:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VISDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/It
# Mapping using ODM element ItemData with ItemOID IT.VSTIM
$VISTIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/It
# Take into account that it might be that the visit time has not always been collected
if($VISTIME != '') {
    $VS.VSDTC = createdatetime($VISDATE,$VISTIME);
    # alternatively, we can use: $VS.VSDTC = concat($VISTDATE,'T',$VISTIME);
} else {
    $VS.VSDTC = $VISDATE;
}
```

We can either use the "createdatetime" function (which is a build-in function) or perform the concatenation ourselves using the "concat" function.

Our result is:

SDTM Tables

My Study:DM | My Study:EC | My Study:VS

VS.VSPOS	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC	VS.VSSTRESN	VS.VSSTRESU	VS.VSDTC
ITTING	SYSBP	100	mmHg	100	100	mmHg	2006-05-01T12:48:00
ITTING	DIABP	70	mmHg	70	70	mmHg	2006-05-01T12:48:00
ITTING	PULSE	62	beats/min	62	62	beats/min	2006-05-01T12:48:00
	WEIGHT	88	kg	88	88	kg	2006-05-01T12:48:00
	BMI	23	kg/m2	23	23	kg/m2	2006-05-01T12:48:00
ITTING	SYSBP	108	mmHg	108	108	mmHg	2006-05-01T12:54:08
ITTING	DIABP	74	mmHg	74	74	mmHg	2006-05-01T12:54:08
ITTING	PULSE	65	beats/min	65	65	beats/min	2006-05-01T12:54:08
ITTING	SYSBP	107	mmHg	107	107	mmHg	2006-05-01T13:07:22
ITTING	DIABP	75	mmHg	75	75	mmHg	2006-05-01T13:07:22
ITTING	SYSBP	105	mmHg	105	105	mmHg	2006-05-03T12:01:00
ITTING	DIABP	76	mmHg	76	76	mmHg	2006-05-03T12:01:00
ITTING	PULSE	63	beats/min	63	63	beats/min	2006-05-03T12:01:00
	WEIGHT	88.2	kg	88.2	88.2	kg	2006-05-03T12:01:00
	BMI	23	kg/m2	23	23	kg/m2	2006-05-03T12:01:00
ITTING	SYSBP	108	mmHg	108	108	mmHg	2006-05-03T12:07:22
ITTING	DIABP	74	mmHg	74	74	mmHg	2006-05-03T12:07:22
ITTING	SYSBP	100	mmHg	100	100	mmHg	2006-05-01T12:48:00
ITTING	DIABP	70	mmHg	70	70	mmHg	2006-05-01T12:48:00
ITTING	PULSE	62	beats/min	62	62	beats/min	2006-05-01T12:48:00
	WEIGHT	88	kg	88	88	kg	2006-05-01T12:48:00
	BMI	22	kg/m2	22	22	kg/m2	2006-05-01T12:48:00
ITTING	SYSBP	108	mmHg	108	108	mmHg	2006-05-01T12:54:08
ITTING	DIABP	74	mmHg	74	74	mmHg	2006-05-01T12:54:08
ITTING	PULSE	65	beats/min	65	65	beats/min	2006-05-01T12:54:08
ITTING	SYSBP	107	mmHg	107	107	mmHg	2006-05-01T13:07:22
ITTING	DIABP	75	mmHg	75	75	mmHg	2006-05-01T13:07:22
ITTING	SYSBP	105	mmHg	105	105	mmHg	2006-05-03T12:01:00

(remark that we moved the VSTESTCD column somewhat to the right for a better visualization of the results).

Once we have VSDTC, the derivation of VSDY is very simple. Here is the script:

The Transformation Script

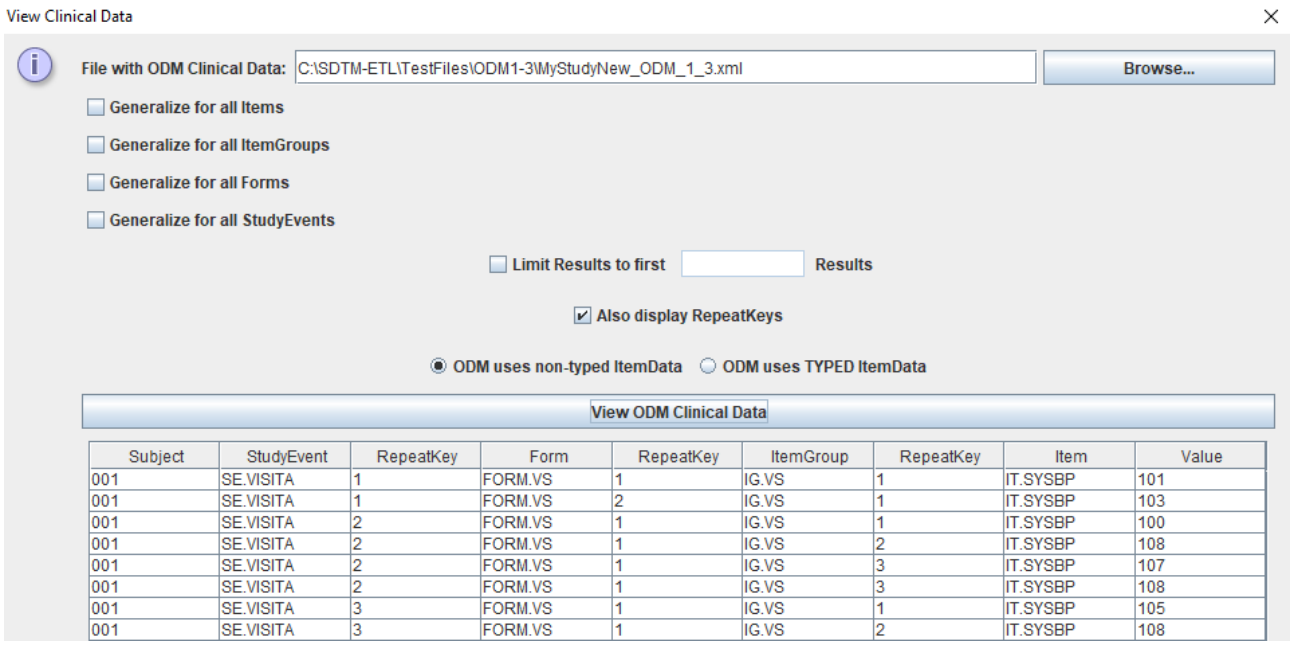
```
# difference in days beween date/time of collection and reference
$DAYS = datediff($VS.VSDTC,$RFSTDTC);
# for non-negative differences, add 1
# as VSDY is not allowed to be 0
if($DAYS >= 0) {
     $VS.VSDY = $DAYS + 1;
} else {
     $VS.VSDY = $DAYS;
}
```

We use the function "datediff" to obtain the number of days between the date/time of collection and the reference start date RFSTDTC, which was previously defined as a global variable (see the tutorial "Generating mappings for the SV domain & creating global variables for reuse"), and then add 1 when the number of days is non-negative. The reason is that the FDA considers the reference start date as day 1, and there is no day 0 according to the FDA.

Generating mappings for VISITNUM and VISIT

VISITNUM (Visit Number) is an expected variable (integer datatype) in SDTM, VISIT a permissible variable.
There is no requirement that visit numbers are subsequent. They just have to be unique for each planned visit, and need to be described in the "trial visits" (TV) domain. There is also no requirement that they are positive numbers. So, we might e.g. assign negative numbers to all visits before the reference start date and positive numbers for the visits at or after the reference date. Unscheduled visits should be given a floating point number. For example, if there are two unscheduled visits between visit 2 and 3, these can be given visit number 2.1 and 2.2.

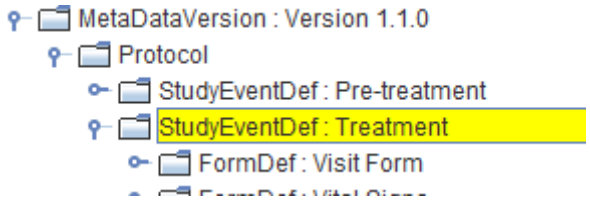
In our case, we have a single "pre-treatment" visit (ODM: Repeating=No) and several "treatment visits" (ODM: Repeating=Yes"). We will give the "pre-treatment" visit the visit number 1, and the repeating visits a number starting from 2. We could however also have started from e.g. 11 for the repeating visits, that is completely OK.
The nice thing about ODM is that in the clinical data the repeat number is already given by the attribute "RepeatKey". We can easily see this when using the menu "View – ClinicalData", and select the checkbox "Also display RepeatKeys". For example, for our data point "systolic blood pressure":



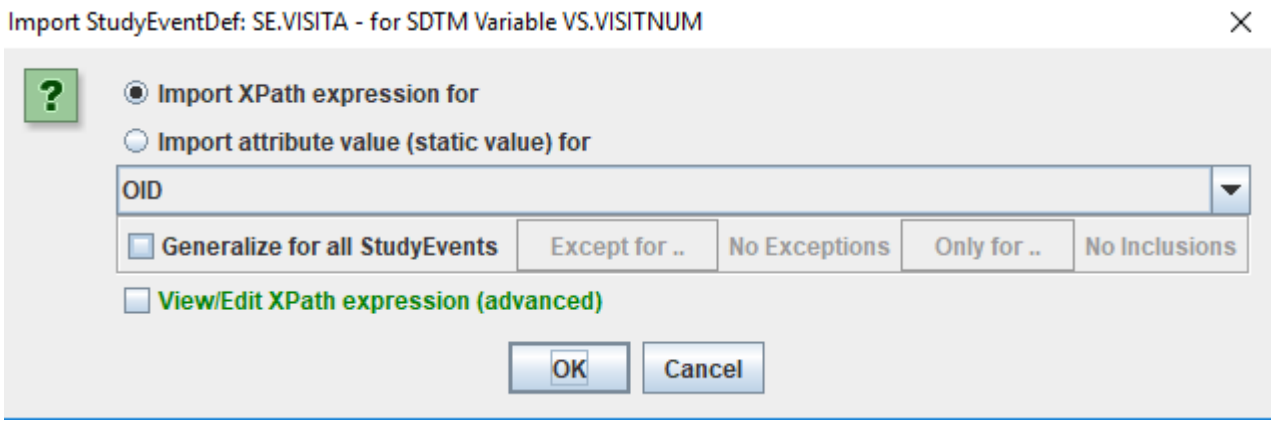
Subject	StudyEvent	RepeatKey	Form	RepeatKey	ItemGroup	RepeatKey	Item	Value
001	SE.VISITA	1	FORM.VS	1	IG.VS	1	IT.SYSBP	101
001	SE.VISITA	1	FORM.VS	2	IG.VS	1	IT.SYSBP	103
001	SE.VISITA	2	FORM.VS	1	IG.VS	1	IT.SYSBP	100
001	SE.VISITA	2	FORM.VS	1	IG.VS	2	IT.SYSBP	108
001	SE.VISITA	2	FORM.VS	1	IG.VS	3	IT.SYSBP	107
001	SE.VISITA	2	FORM.VS	1	IG.VS	3	IT.SYSBP	108
001	SE.VISITA	3	FORM.VS	1	IG.VS	1	IT.SYSBP	105
001	SE.VISITA	3	FORM.VS	1	IG.VS	2	IT.SYSBP	108

We see that for subject 001, the "treatment" visit (SE.VISITA) has been repeated 3 times (RepeatKey values from 1 to 3). So we can easily use the value of "RepeatKey" for calculating the visit number.

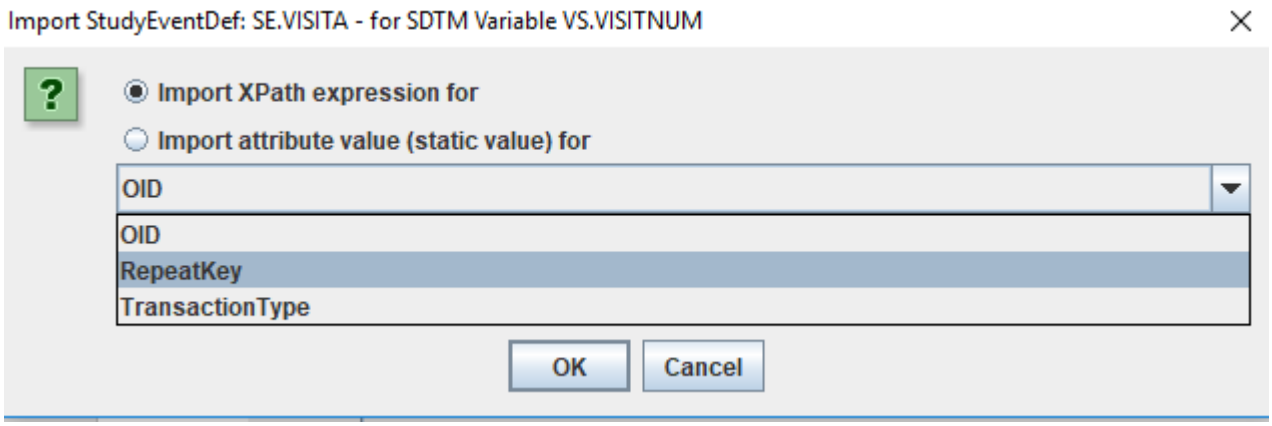
In order to do so, drag-and-drop the "StudyEventDef Treatment" to the SDTM cell "VS.VISITNUM":



A dialog shows up:



We do however not retrieve the OID (the identifier) but we want the "RepeatKey", so we use the selection box as follows:



After clicking "OK", the mapping script is generated:

The Transformation Script

```
# Mapping using ODM element StudyEventData using value from attribute StudyEventRepeatKey
$VS.VISITNUM = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/@StudyEventRepeatKey);
```

And adapt it to:

The Transformation Script

```
# Mapping using ODM element StudyEventData using value from attribute StudyEventRepeatKey
$REPEATKEY = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/@StudyEventRepeatKey);
$VS.VISITNUM = $REPEATKEY + 1;
```

For the SDTM variable "VISIT", we can reuse the value from "VISITNUM". For example, we can set it as:

The Transformation Script

```
$VS.VISIT = concat('TREATMENT VISIT ',$VS.VISITNUM);
```

Our result then becomes:

SDTM Tables

My Study:DM | My Study:EC | My Study:VS

TRESC	VS.VSSTRESN	VS.VSSTRESU	VS.VISITNUM	VS.VISIT	VS.VSDTC	VS.VSDY
	101	mmHg	2	TREATMENT VISIT 2	2006-04-30T12:48:33	30
	67	mmHg	2	TREATMENT VISIT 2	2006-04-30T12:48:33	30
	63	beats/min	2	TREATMENT VISIT 2	2006-04-30T12:48:33	30
	88.1	kg	2	TREATMENT VISIT 2	2006-04-30T12:48:33	30
	25.6	kg/m2	2	TREATMENT VISIT 2	2006-04-30T12:48:33	30
	103	mmHg	2	TREATMENT VISIT 2	2006-04-30T12:59:33	30
	68	mmHg	2	TREATMENT VISIT 2	2006-04-30T12:59:33	30
	65	beats/min	2	TREATMENT VISIT 2	2006-04-30T12:59:33	30
	100	mmHg	3	TREATMENT VISIT 3	2006-05-01T12:48:00	31
	70	mmHg	3	TREATMENT VISIT 3	2006-05-01T12:48:00	31
	62	beats/min	3	TREATMENT VISIT 3	2006-05-01T12:48:00	31
	88	kg	3	TREATMENT VISIT 3	2006-05-01T12:48:00	31
	25.6	kg/m2	3	TREATMENT VISIT 3	2006-05-01T12:48:00	31
	108	mmHg	3	TREATMENT VISIT 3	2006-05-01T12:54:08	31
	74	mmHg	3	TREATMENT VISIT 3	2006-05-01T12:54:08	31

How the visits are exactly named is of course up to the mapper. This is just an example.

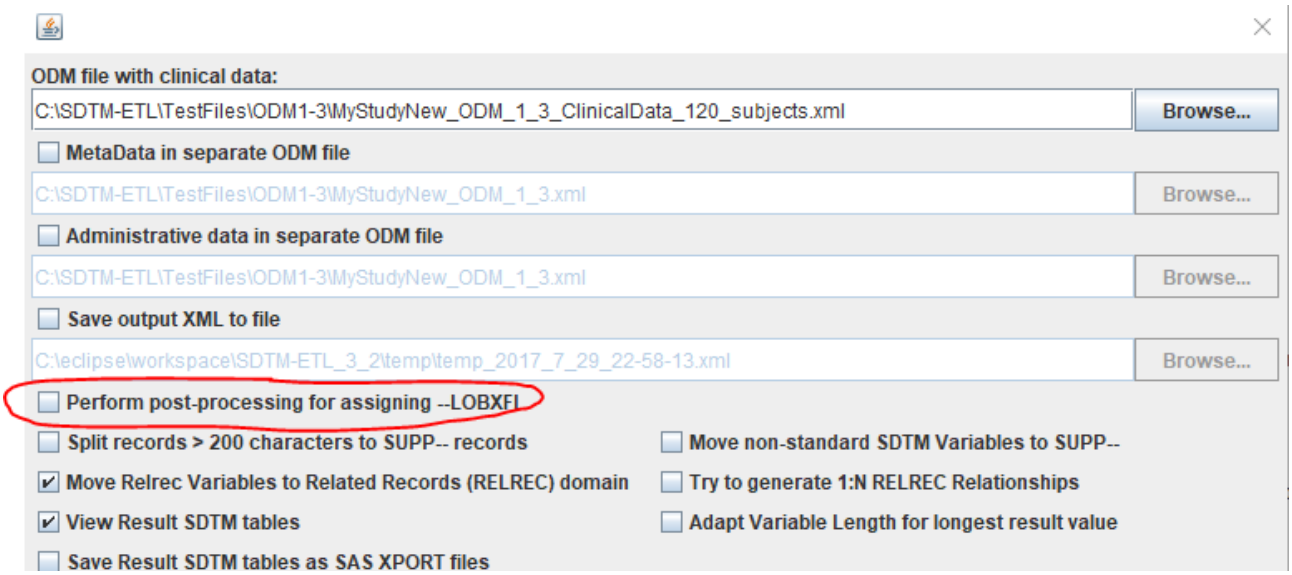
Also, do not forget to add the visit numbers and names to the TV (trial visits) dataset.

The baseline flag

The last mapping we want to do is the one for the „baseline" flag (VSBLFL). How to do that is to be decided on by the sponsor. Often, the baseline flag measurement is assigned to the last measurement for each test before first exposure to the study treatment.

As there are different ways to generate a mapping for the baseline flag, depending on how the reference start date is set and what needs to be taken as the baseline flag, we developed a separate tutorial "Generating baseline flags". It explains these different ways in great detail.
Please use this tutorial for understanding the different ways of assigning the baseline flag.

Also remark that as of SDTM-IG 3.3, a new variable –LOBXFL (Last Observation Before Exposure Flag) has been added[10]. Our SDTM-ETL software already implemented an algorithm for calculating the values for this new variable. It requires a post-processing step, which can be initiated at execution time using the checkbox "Perform post-processing for assigning "LOBXFL"



We will further discuss the automated generation of -LOBXFL variables in a separate tutorial.

[10] Also see our article „Why LOBXFL should not be in SDTM", which can be found at:
http://cdiscguru.blogspot.com/2016/08/why-lobxfl-should-not-be-in-sdtm.html