SDTM-ETL 5.x: Derivation of data points that were not collected

Author: Jozef Aerts, XML4Pharma

Last update: **2026-05-26**

Introduction

Sometimes, we have the case that a data point is to be derived from other data points within the same Findings dataset. A typical example is BMI (Body Mass Index).

Normally, such calculations are done in the EDC system itself, either "on the fly", or at ODM export time. However …
SDTM should not contain "derived data" - these should go in ADaM. Unfortunately, regulatory authorities (especially FDA) have forced the CDISC SDTM development team to add more and more "derived" variables into SDTM, as their tools are too primitive to have these calculations done by the reviewers themselves. Typical examples are the --DY variables and more recently, the --LOBXFL variables.
As these are variables in SDTM itself, this is not what we mean here.
What we mean are existing SDTM variables like --ORRES that need to be populated with values that were not collected, like BMI.

We will provide a few possibilities to deal with such, some on the level of the ODM, some using SDTM-ETL. The latter are not very elegant, but they do solve the problem

A. Derivation within the ODM export

One possibility is of course to "correct" this within the ODM itself.

Suppose we have the following structure with vital signs data points:

```
<ItemGroupDef Domain="VS" Name="Physical Exam" OID="IG_PE_BASE" Repeating="No" SASDatasetName="PHYEXBAS">
    <Description>
        <TranslatedText xml:lang="en">Physical examination: Base</TranslatedText>
        <TranslatedText xml:lang="fr">Examination du corps: Base</TranslatedText>
        <TranslatedText xml:lang="de">Körperuntersuchung: Basis</TranslatedText>
        <TranslatedText xml:lang="ko">□□ □□: □□</TranslatedText>
    </Description>
    <ItemRef ItemOID="I_HEIGHT" Mandatory="Yes" OrderNumber="1"/>
    <ItemRef ItemOID="I_WEIGHT" Mandatory="Yes" OrderNumber="2"/>
    <ItemRef ItemOID="I_SYSBP" Mandatory="No" OrderNumber="3"/>
    <ItemRef ItemOID="I_DIABP" Mandatory="Yes" OrderNumber="4"/>
    <ItemRef CollectionExceptionConditionOID="COND.DBP.DIZZY" ItemOID="I_DIZZY" Mandatory="Yes" OrderNumber="5"/>
    <Alias Context="SDTM" Name="VS"/>
</ItemGroupDef>
```

where "BMI" is missing indeed.
If we want to add it, we need to edit the ODM metadata (can be with a suitable, perhaps "visual", tool, so not necessary an XML editor. The result would then e.g. be:

```
<ItemGroupDef Domain="VS" Name="Weekly Physical Exam" OID="IG_PE_WEEK" Repeating="No" SASDatasetName="PHYEX">
    <Description>
        <TranslatedText xml:lang="en">Physical Examination: Week</TranslatedText>
        <TranslatedText xml:lang="fr">Examination du corps: Semaine</TranslatedText>
        <TranslatedText>Körperuntersuchung: Woche</TranslatedText>
        <TranslatedText xml:lang="ko">□□□□:□ </TranslatedText>
    </Description>
    <ItemRef ItemOID="I_WEIGHT" Mandatory="Yes" OrderNumber="1"/>
    <ItemRef ItemOID="I_SYSBP" Mandatory="Yes" OrderNumber="2"/>
    <ItemRef ItemOID="I_DIABP" Mandatory="Yes" OrderNumber="3"/>
    <ItemRef ItemOID="I_DIABP" Mandatory="Yes" OrderNumber="4"/>
    <ItemRef ItemOID="I_BMI" Mandatory="No" OrderNumber="5" MethodOID="METH.BMI"/>
    <ItemRef CollectionExceptionConditionOID="COND.DBP.DIZZY" ItemOID="I_DIZZY" Mandatory="Yes" OrderNumber="6"/>
    <Alias Context="SDTM" Name="VS"/>
</ItemGroupDef>
```

having inserted an additional item, for which we can also add a reference to a "MethodDef" for describing the derivation, if desired.
One should then also add the corresponding "ItemDef" to the metadata part in the ODM. For example:

```
<ItemDef OID="I_BMI" DataType="float" SignificantDigits="1" Name="BMI" SDSVarName="VSORRES">
    <Alias Context="SDTM" Name="VSORRES WHERE VSTESTCD=BMI"/>
</ItemDef>
```

Notice the "annotation" "Alias" stating where this will go into in SDTM. This annotation is not really necessary, but can be helpful for unexperienced mappers.

Usually using a program (we don't want to do this manually if course), we then add the values for the BMI in the "ClinicalData" part, e.g. leading to:

```
<ItemGroupData ItemGroupOID="IG_PE_BASE">
    <ItemData ItemOID="I_HEIGHT" Value="193">
        <MeasurementUnitRef MeasurementUnitOID="MU_CM"/>
    </ItemData>
    <ItemData ItemOID="I_WEIGHT" Value="90">
        <MeasurementUnitRef MeasurementUnitOID="MU_KG"/>
    </ItemData>
    <ItemData ItemOID="I_SYSBP" Value="120">
        <MeasurementUnitRef MeasurementUnitOID="MU_MMHG"/>
    </ItemData>
    <ItemData ItemOID="I_DIABP" Value="80">
        <MeasurementUnitRef MeasurementUnitOID="MU_MMHG"/>
    </ItemData>
    <!-- Calculated BMI, not collected -->
    <ItemData ItemOID="I_BMI" Value="24.16">
        <!-- If wanted, we can also add a "MeasurementUnitRef here,
        pointing to the unit defining kg/m2-->
    </ItemData>
    <ItemData ItemOID="I_DIZZY" Value="1"/>
</ItemGroupData>
```

As we then have both the metadata as well as clinical data for BMI, we can just proceed as usual, as e.g. described in the tutorial "Adding Mappings for the Vital Signs Domain".

As this requires adding data point definitions in, and calculations on an existing ODM (probably generating a new ODM), this may not always be the most appropriate choice.

As already stated however, such calculation should normally be done within the EDC system

anyway.

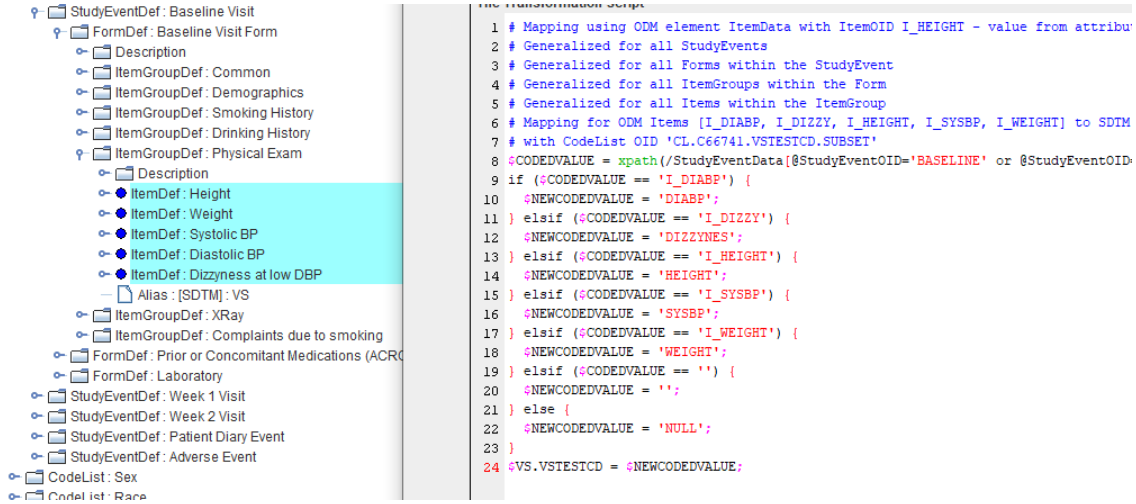
B. Derivation in SDTM-ETL

Essentially, SDTM-ETL can only treat information that is available in the ODM file. For the cases that no ODM file is available, there is the "ODM Generator", which can generate ODM files starting from spreadsheets (using a CSV export), or SAS Transport datasets, or any other "text-based" file. Such generated ODM files can also be merged into a single one using the "ODMMerger", that can be obtained together with the "ODMGenerator". If the data is however already organized per domain, merging into a single large ODM file is not always necessary.

A major property of ODM is that it only contains data points for which there is data. If a data point like for "BMI" was not collected or derived with the EDC system, there will simply be no "ItemData" line for it in the ODM. Essentially, as SDTM-ETL is based on retrieval of data from the ODM, this would mean that "BMI" cannot be added if it is not in the ODM.

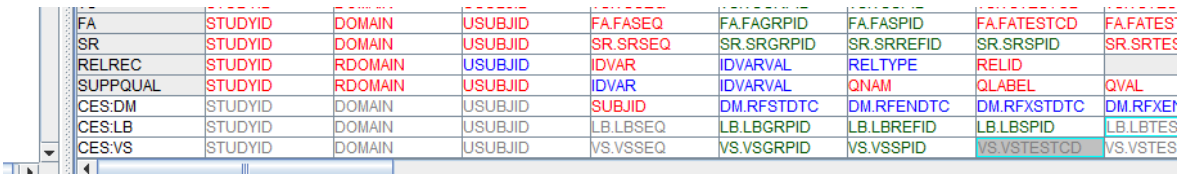
There is however a "workaround" that can be used.

Suppose we have the following mappings defined:

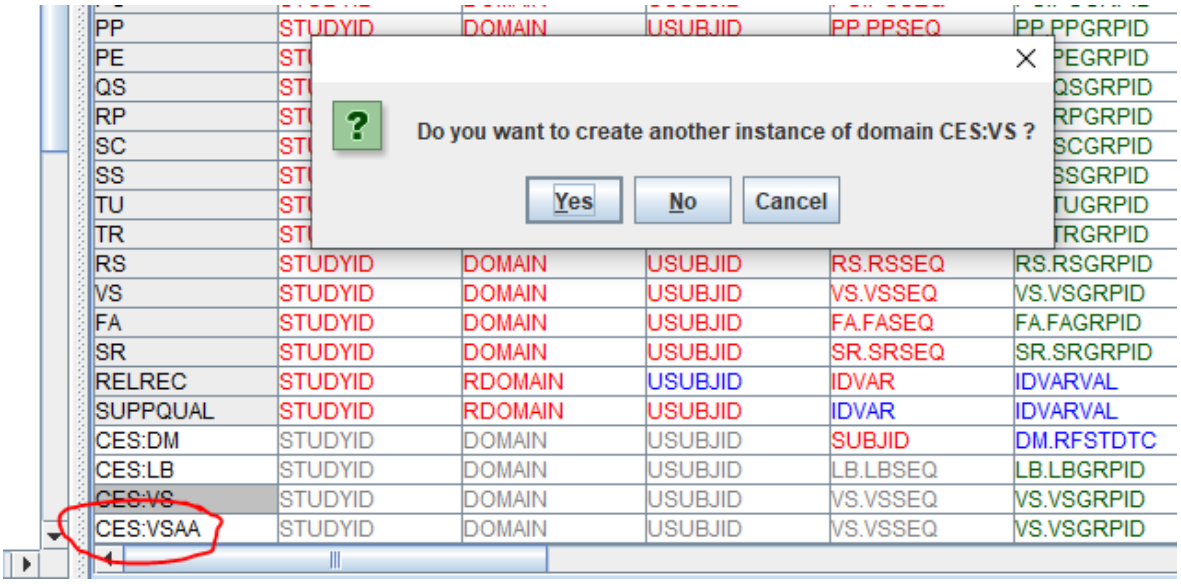


As one can see. there is nothing for "BMI".
However, as we do have "Weight" and "Height", we can treat this separately, generating a separate dataset for it, and then merge that with the one without BMI.

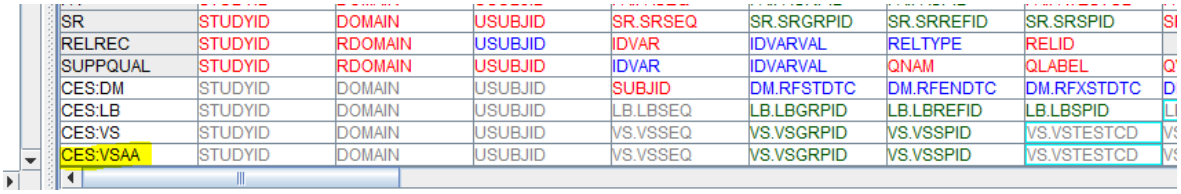
In our main GUI window, we have:



Now, we select the first cell "CES:VS" in the last row, and drag-and-drop it on itself. Alternatively, after selecting it, we use the menu "Edit - Copy Domain/Dataset" followed by "Edit - Paste Domain/Dataset". This then leads to the following dialog:

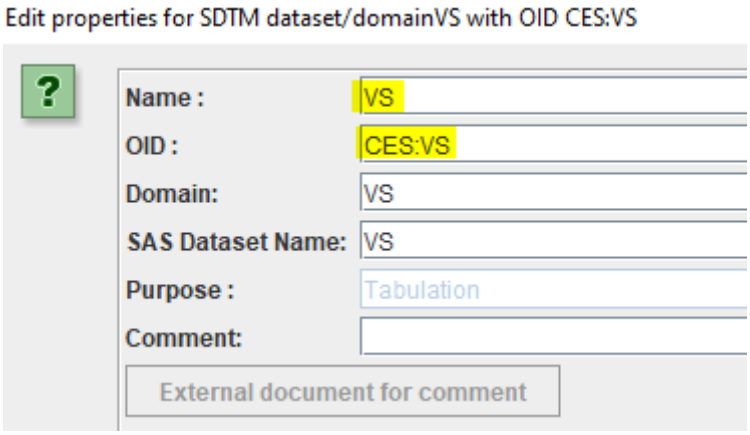


asking us whether it is OK to generate another "study-specific instance" of VS. We can already see that it is already presented as an extra row at the bottom, but we will still need to click "Yes" to confirm. When doing so, this leads to:

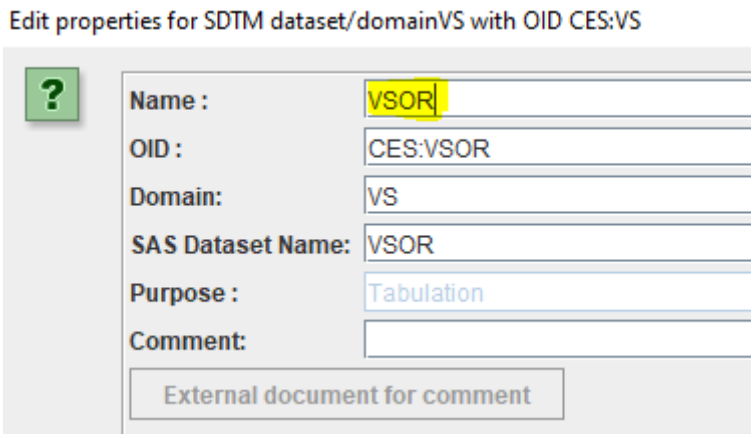


Now, as we copied the first instance "CES:VS" completely to the second (automatically assigned the ID "CES:VSAA"), when we would execute all the mappings, we would get two identical VS datasets, one named "VS", and one named "VSAA".

Let us first rename both to make things more clear.
When selecting "CES:VS" and double-clicking it, or using the menu "Edit - SDTM Dataset Definition Properties", we can change the ID, from:



to e.g.:

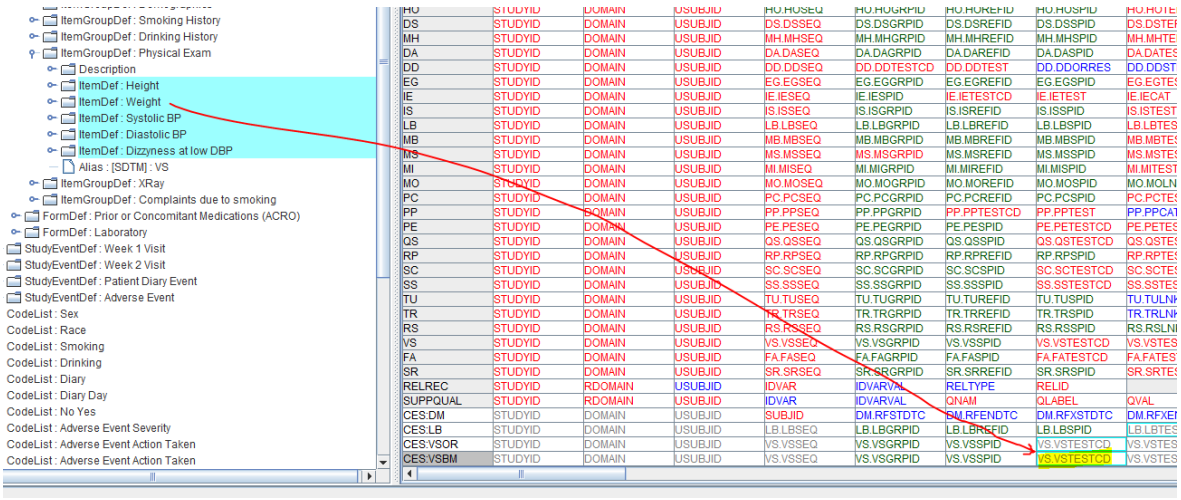


just by typing in the upper field ...

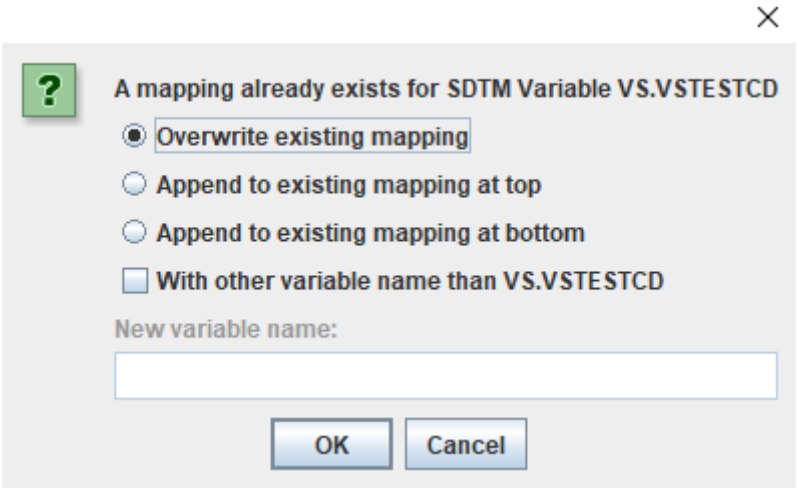
We could say "VSOR" then means "VS Original"

We then do something similar for VSAA" which we rename e.g. to "VSBM", standing for "VS BMI". Notice that it is "best practice" to limit data set names, even when they are just temporary, to 4 characters.

What we are going to do now, is to use the second one to generate a separate dataset with just the BMI values, and then merge that with the original one during execution time.
As BMI is derived from "weight" and "height", we first drag-and-drop "weight" from the ODM tree to the VSTESTCD cell of the instance "VSBM":

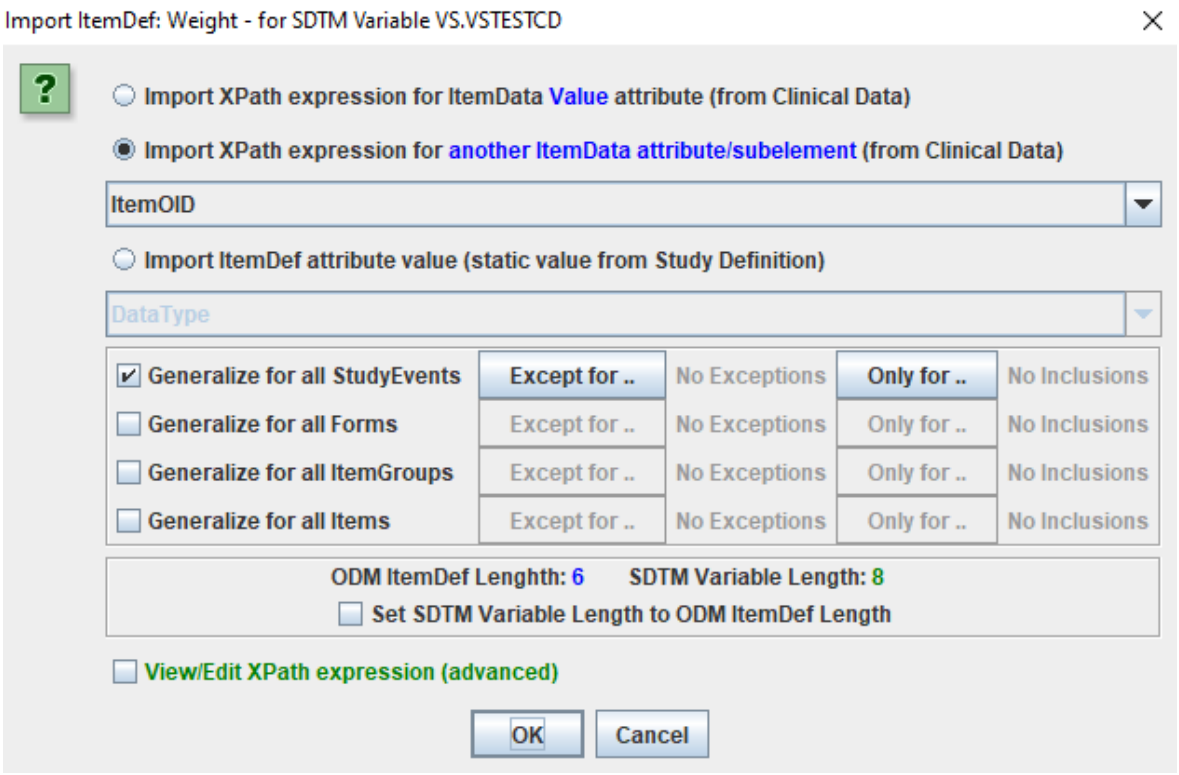


When doing the "drop", as we did already have mappings defined in that cell, the system will ask whether we want to "overwrite" that mapping, or that we want to append to it:

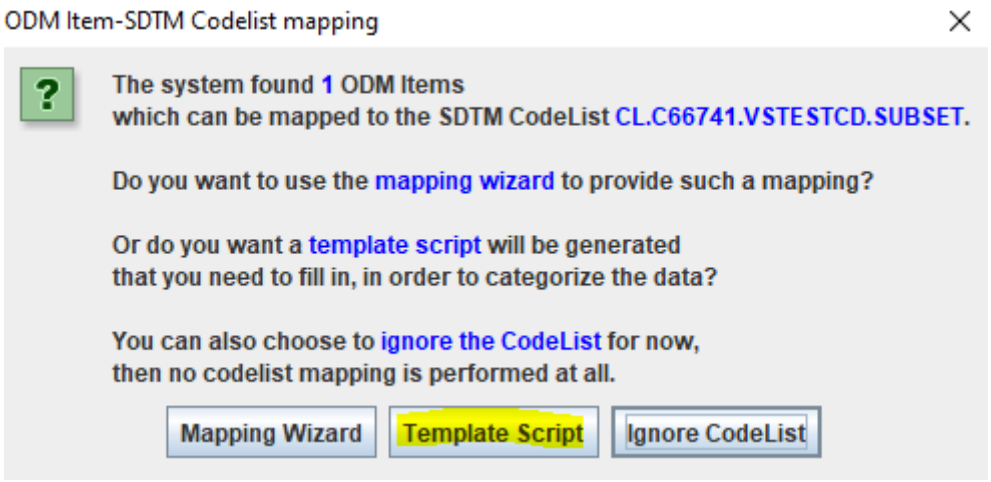


We select to "overwrite" it.

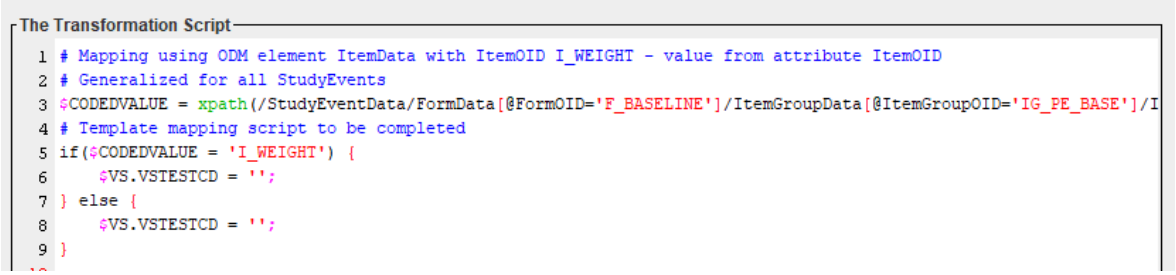
The usual "wizards" shows up, but this time, as we only want to import "weight" for the moment, we must take care that we do NOT select "Generalize for all Items":



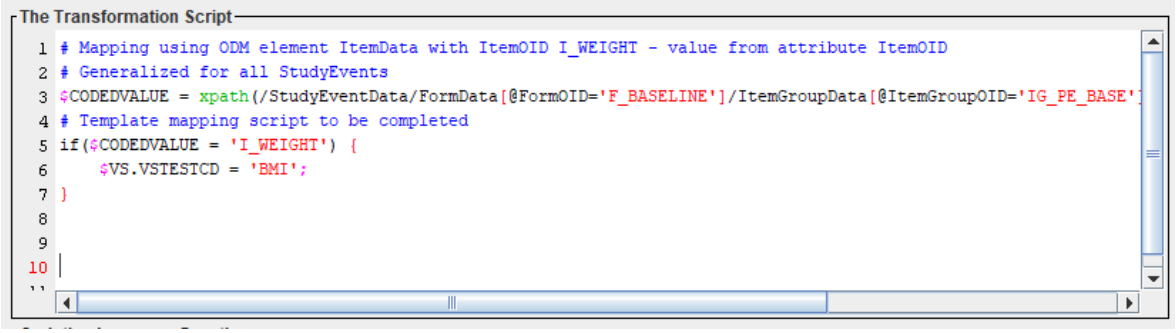
leading to:



In this case, we are NOT going to use the "Mapping Wizard" to find out how to map "Weight" to "WEIGHT", but we will map "Weight" to "BMI". So we select "Template Script", leading to:

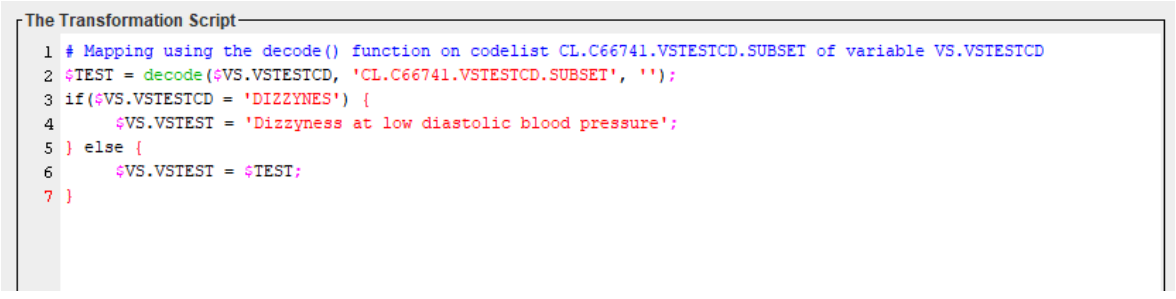


which we then change into:



so, in this "special" instance, we will generate a BMI for each data point for which we have a "Weight".

For VSTEST ("test name"), we just double click and find:



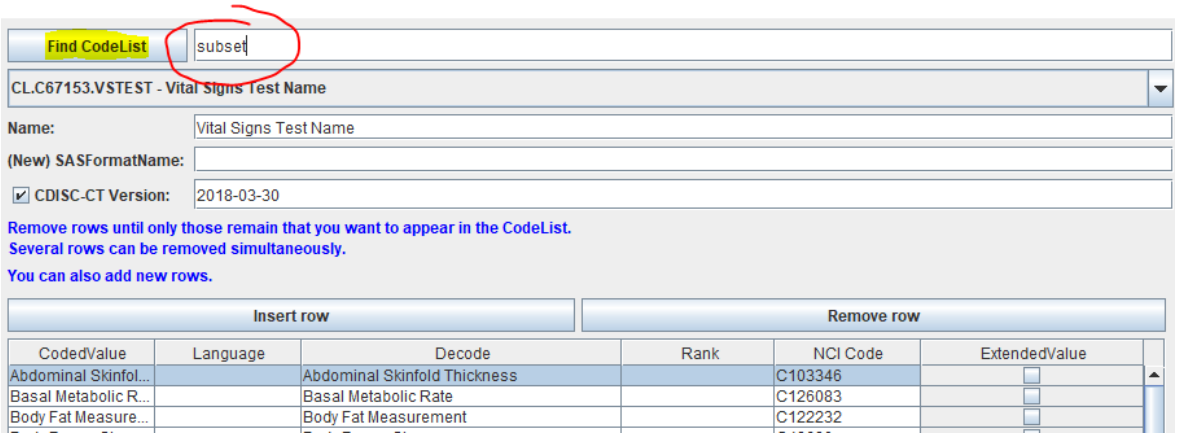
which, as we only have "BMI" now, we can reduce to:

The Transformation Script

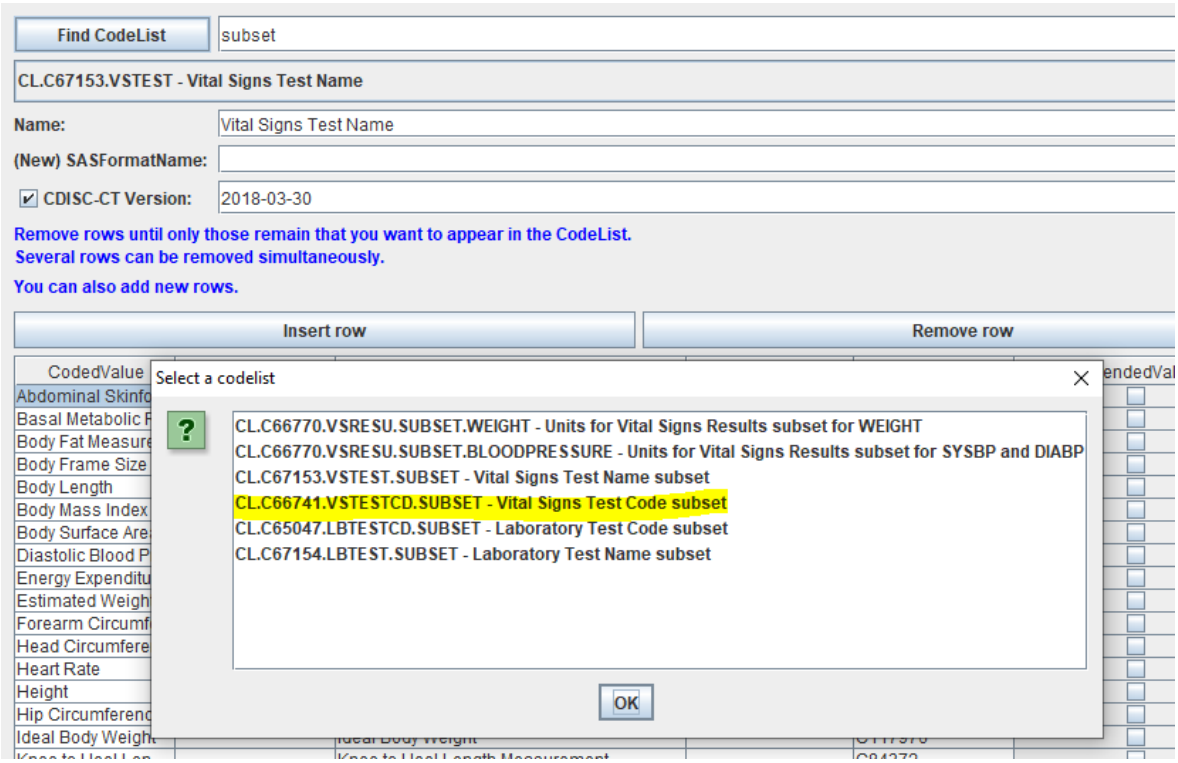
```
1 # Mapping using the decode() function on codelist CL.C66741.VSTESTCD.SUBSET of variable VS.VSTESTCD
2 $VS.VSTEST = decode($VS.VSTESTCD, 'CL.C66741.VSTESTCD.SUBSET', '');
3
```

As one sees, in the original mapping without BMI, we generated a "subset codelist", that we use for "decoding" the value of VSTESTCD to VSTEST, but of course, it does not contain "BMI", as we did not foresee that at that time.

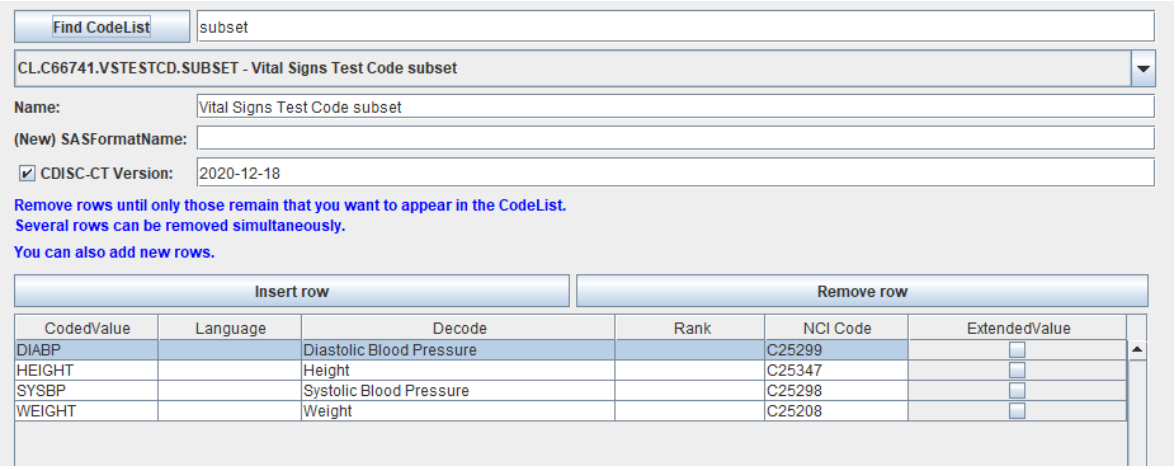
So, we need to extend that codelist.
First we click "OK" for the mapping and then use the menu "Edit - SDTM CodeList" and then search for "subset":



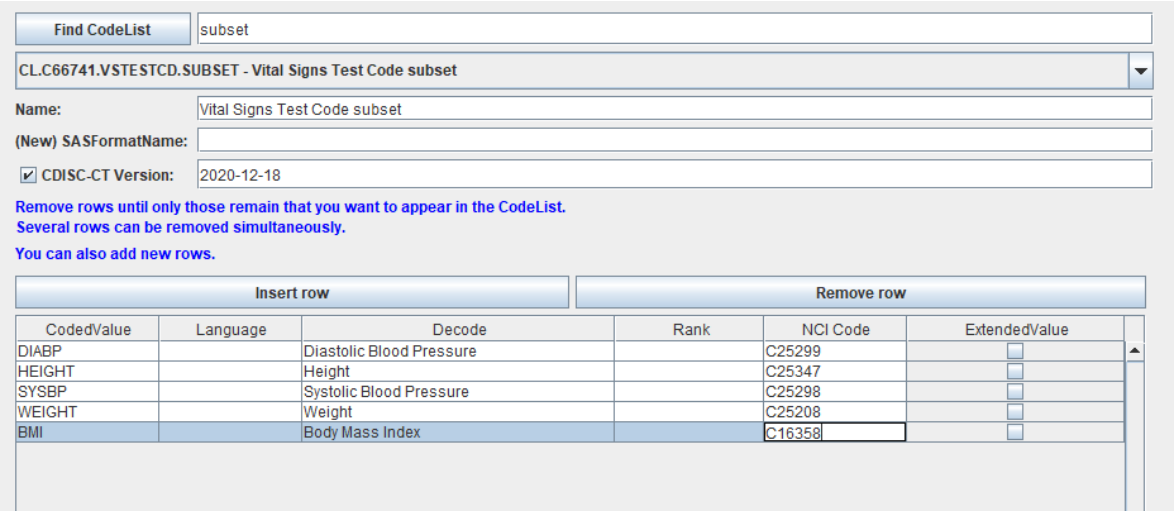
When then clicking "Find CodeList" we can select the appropriate one:



and after "OK", we can start editing it, adding "BMI":



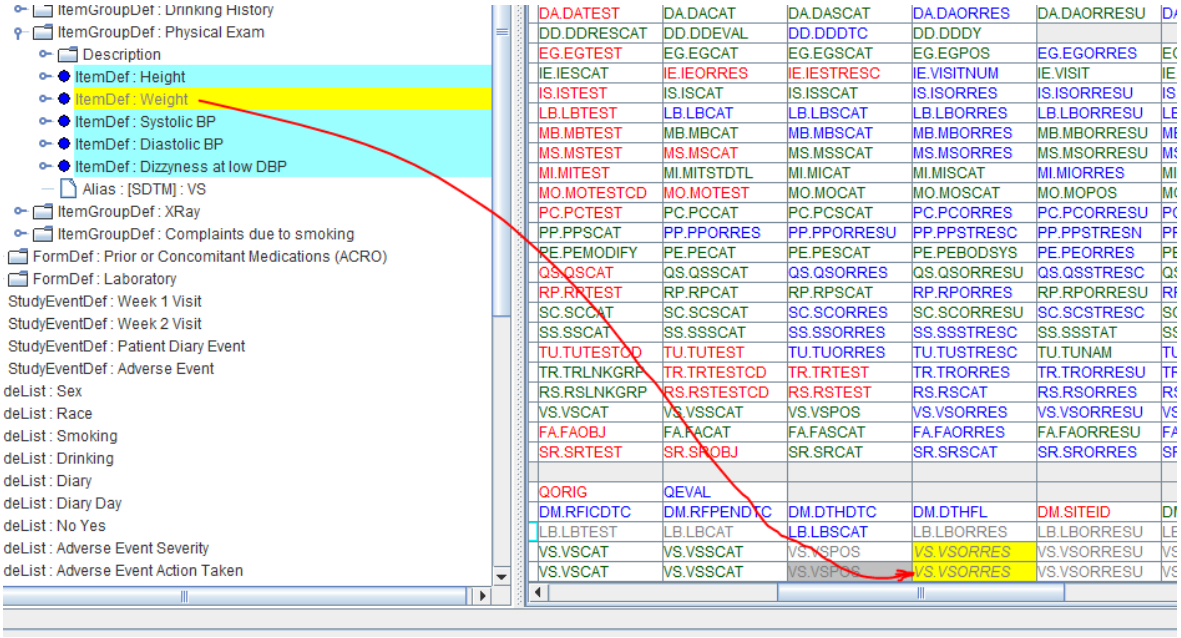
adding a row at the bottom (but essentially the order is unimportant) and filling it with the information:



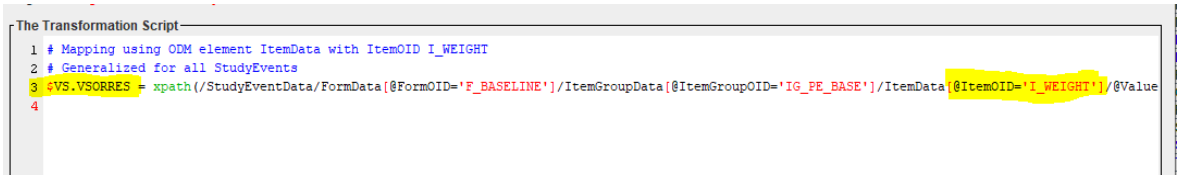
The NCI code for BMI can be found from the CDISC publications of the Controlled Treminology, e.g. using the "CDISC Library Browser". As "BMI" is a formal member of the codelist for Vital Signs Test Codes, do NOT check the checkbox "Extended Value". The latter is only to be used when you want to add an additional term, not published by CDISC, to the codelist.

So, we now added "BMI" to the codelist, so the "decode" function in the mapping script for VSTEST should now work.

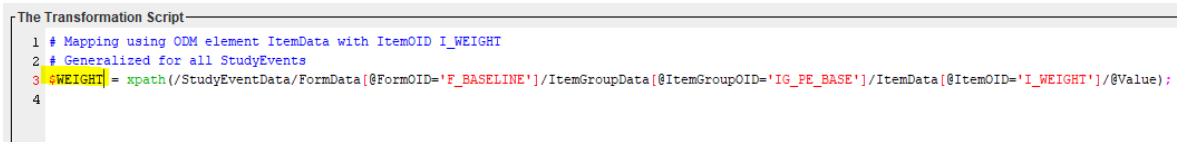
After clicking "OK", we then go back to the main screen, as we want to start adding the derivation of the value of the BMI in VSORRES.
We first drag-and-drop "Weight" to "VSORRES", and ask to "overwrite" the existing mapping:



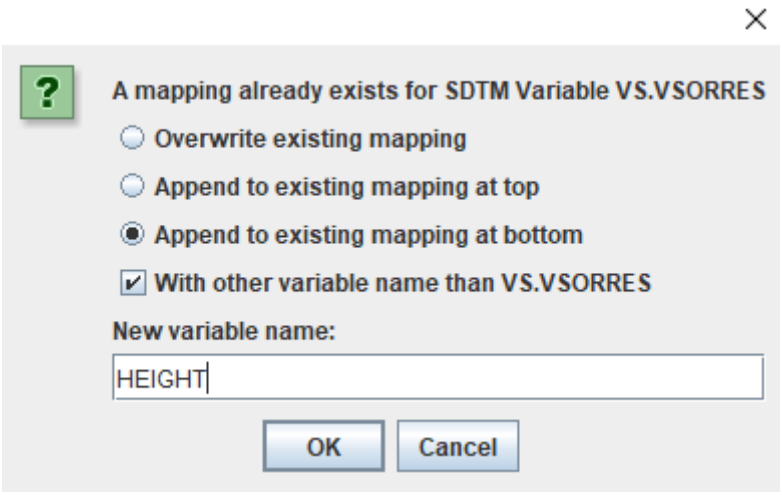
This leads to:



but we do not want to get the value of the "weight" to go into VSORRES, so we change this e.g. in "WEIGHT" as a temporary variable, like:



After clicking "OK", we then also drag-and-drop "Height", and select to "Append" with another variable name:



leading to:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID I_WEIGHT
# Generalized for all StudyEvents
$WEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemData[@ItemOID='I_WEIGHT']/@Value);
# Mapping using ODM element ItemData with ItemOID I_HEIGHT
# Generalized for all StudyEvents
$HEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemData[@ItemOID='I_HEIGHT']/@Value);

```

So we have now retrieved the values for "HEIGHT" and "WEIGHT" from which we can generate the BMI which will be assigned to VSTESTCD=BMI and VSTEST="Body Mass Index".

So we add:

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID I_WEIGHT
# Generalized for all StudyEvents
$WEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemData[@ItemOID='I_WEIGHT']/@Value);
# Mapping using ODM element ItemData with ItemOID I_HEIGHT
# Generalized for all StudyEvents
$HEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemData[@ItemOID='I_HEIGHT']/@Value);
# BMI calculation, taking into account that "height" is provided in centimeters
$VS.VSORRES = 10000 * $WEIGHT / ($HEIGHT * $HEIGHT);
```

One may now think we are done, but for the other variables in this row, we must still do a clean-up. For example, having a value for VSPOS does not make sense for "BMI", and also we may have some other retrievals which may not be correct anymore, as they retrieve more than one value, due to the "generalize for" mappings we did for the original mappings without BMI in "CES:VSOR". So often, the best thing will be to "empty" these, and then re-add them later. For example, for VSPOS, we can set:

The Transformation Script

```
$VS.VSPOS = '';
```

and for VSORRESU, we also must change into:

The Transformation Script

```
$VS.VSORRESU = 'kg/m2';

```

Essentially, we must revisit all other mappings in "CES:VSBM" (our "BMI-specific instance) and check whether they are still adequate. This especially considers the ones for which we used "Generalize for all Items" in the past, as we know have just one item …
This especially applies to date/time variables like VSDTC …

Our almost-final result for CES:VSBM then becomes:

SDTM Tables

CES:DM | CES:LB | CES:VSOR | CES:VSBM

STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC
CES	VS	001	12	BMI	Body Mass Index		24.16172246234798	kg/m2	24.16172246234798
CES	VS	002	12	BMI	Body Mass Index		25.05930702662969	kg/m2	25.05930702662969
CES	VS	003	12	BMI	Body Mass Index		23.38868655818771	kg/m2	23.38868655818771
CES	VS	004	12	BMI	Body Mass Index		24.22145328719723	kg/m2	24.22145328719723
CES	VS	005	12	BMI	Body Mass Index		26.31578947368421	kg/m2	26.31578947368421
CES	VS	006	12	BMI	Body Mass Index		24.158817968558523	kg/m2	24.158817968558523
CES	VS	007	12	BMI	Body Mass Index		26.296566837107378	kg/m2	26.296566837107378
CES	VS	008	12	BMI	Body Mass Index		24.691358024691358	kg/m2	24.691358024691358
CES	VS	009	12	BMI	Body Mass Index		27.700831024930746	kg/m2	27.700831024930746
CES	VS	010	12	BMI	Body Mass Index		23.875114784205692	kg/m2	23.875114784205692

Ok, but we do not want to that many characters after the decimal point, but e.g. only 2.
So we edit the mapping for VSORRES again, and use the "round()" function:

The Transformation Script

```
1 # Mapping using ODM element ItemData with ItemOID I_WEIGHT
2 # Generalized for all StudyEvents
3 $WEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemD
4 # Mapping using ODM element ItemData with ItemOID I_HEIGHT
5 # Generalized for all StudyEvents
6 $HEIGHT = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/ItemD
7 # BMI calculation, taking into account that "height" is provided in centimeters
8 $CALCULATED = 10000 * $WEIGHT / ($HEIGHT * $HEIGHT);
9 $VS.VSORRES = round($CALCULATED,2);
```

Scripting Language Functions

contains	starts-with	ends-with	matches	not		
abs	sqrt	log	log10	exp	exp10	
min	max	avg	sum	count	is-a-number	ucum2cdisc
ceiling	floor	round	modulus	number	string	
date	year	month-in-y			in-week	
time	hour-in-day	minute-in-h			atediff	
timediff	datetimediff	elementname	more date/time ...	RESTful WS	My Functions	

Returns the result of rounding the value in the first argument to number of decimals in the second argument

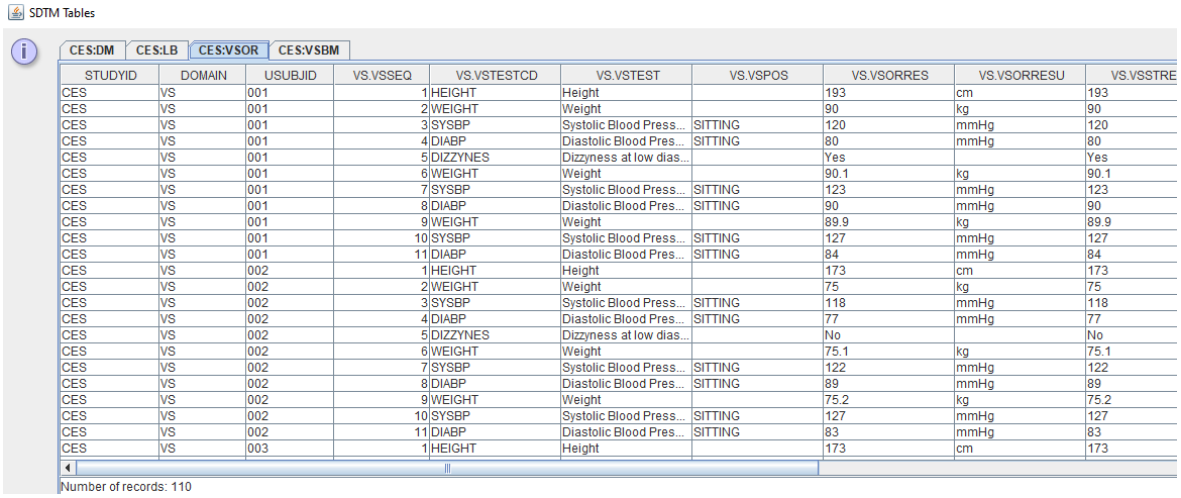
and after executing again, we get for CES:VSMB

SDTM Tables

CES:DM | CES:LB | CES:VSOR | CES:VSBM

STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC	VS.VSST
CES	VS	001	12	BMI	Body Mass Index		24.16	kg/m2	24.16	
CES	VS	002	12	BMI	Body Mass Index		25.06	kg/m2	25.06	
CES	VS	003	12	BMI	Body Mass Index		23.39	kg/m2	23.39	
CES	VS	004	12	BMI	Body Mass Index		24.22	kg/m2	24.22	
CES	VS	005	12	BMI	Body Mass Index		26.32	kg/m2	26.32	
CES	VS	006	12	BMI	Body Mass Index		24.16	kg/m2	24.16	
CES	VS	007	12	BMI	Body Mass Index		26.3	kg/m2	26.3	
CES	VS	008	12	BMI	Body Mass Index		24.69	kg/m2	24.69	
CES	VS	009	12	BMI	Body Mass Index		27.7	kg/m2	27.7	
CES	VS	010	12	BMI	Body Mass Index		23.88	kg/m2	23.88	

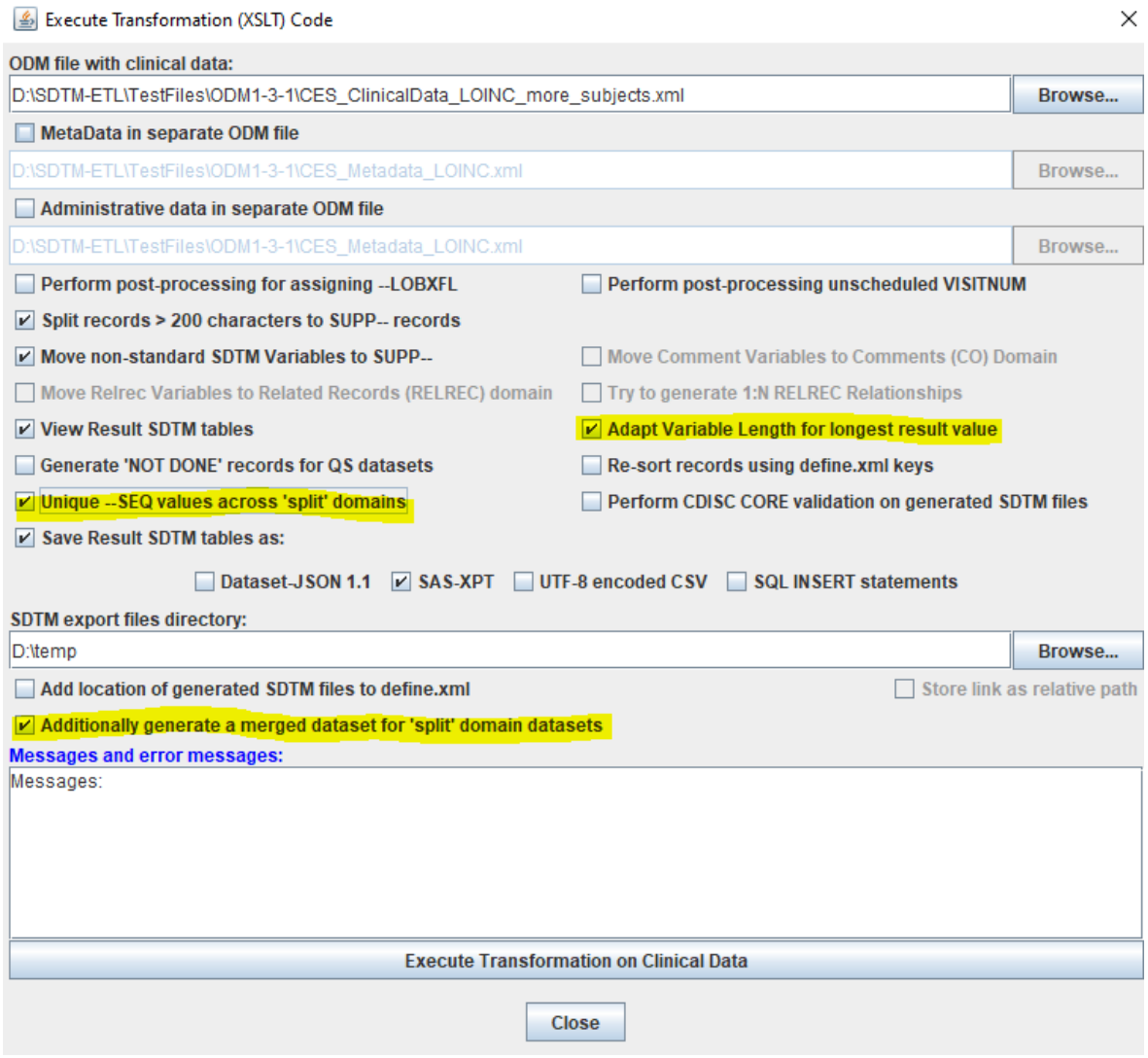
and for the original CES:VSOR, we get:



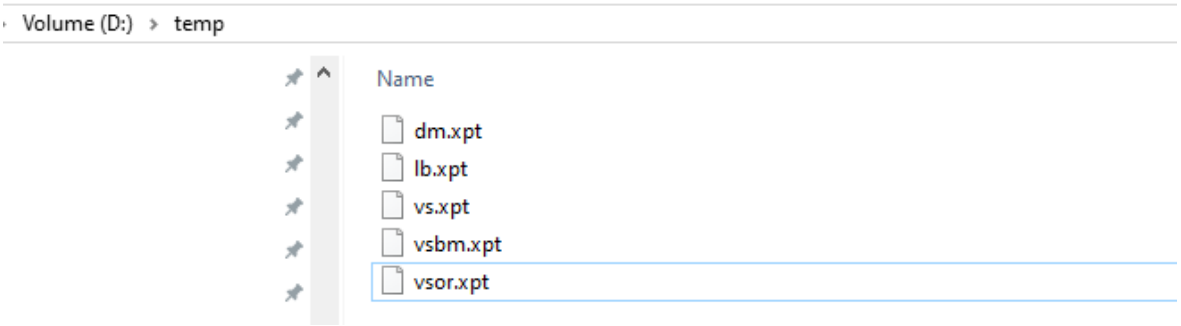
STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRE
CES	VS	001	1	HEIGHT	Height		193	cm	193
CES	VS	001	2	WEIGHT	Weight		90	kg	90
CES	VS	001	3	SYSBP	Systolic Blood Press...	SITTING	120	mmHg	120
CES	VS	001	4	DIABP	Diastolic Blood Pres...	SITTING	80	mmHg	80
CES	VS	001	5	DIZZYNES	Dizzyness at low dias...		Yes		Yes
CES	VS	001	6	WEIGHT	Weight		90.1	kg	90.1
CES	VS	001	7	SYSBP	Systolic Blood Press...	SITTING	123	mmHg	123
CES	VS	001	8	DIABP	Diastolic Blood Pres...	SITTING	90	mmHg	90
CES	VS	001	9	WEIGHT	Weight		89.9	kg	89.9
CES	VS	001	10	SYSBP	Systolic Blood Press...	SITTING	127	mmHg	127
CES	VS	001	11	DIABP	Diastolic Blood Pres...	SITTING	84	mmHg	84
CES	VS	002	1	HEIGHT	Height		173	cm	173
CES	VS	002	2	WEIGHT	Weight		75	kg	75
CES	VS	002	3	SYSBP	Systolic Blood Press...	SITTING	118	mmHg	118
CES	VS	002	4	DIABP	Diastolic Blood Pres...	SITTING	77	mmHg	77
CES	VS	002	5	DIZZYNES	Dizzyness at low dias...		No		No
CES	VS	002	6	WEIGHT	Weight		75.1	kg	75.1
CES	VS	002	7	SYSBP	Systolic Blood Press...	SITTING	122	mmHg	122
CES	VS	002	8	DIABP	Diastolic Blood Pres...	SITTING	89	mmHg	89
CES	VS	002	9	WEIGHT	Weight		75.2	kg	75.2
CES	VS	002	10	SYSBP	Systolic Blood Press...	SITTING	127	mmHg	127
CES	VS	002	11	DIABP	Diastolic Blood Pres...	SITTING	83	mmHg	83
CES	VS	003	1	HEIGHT	Height		173	cm	173

Number of records: 110

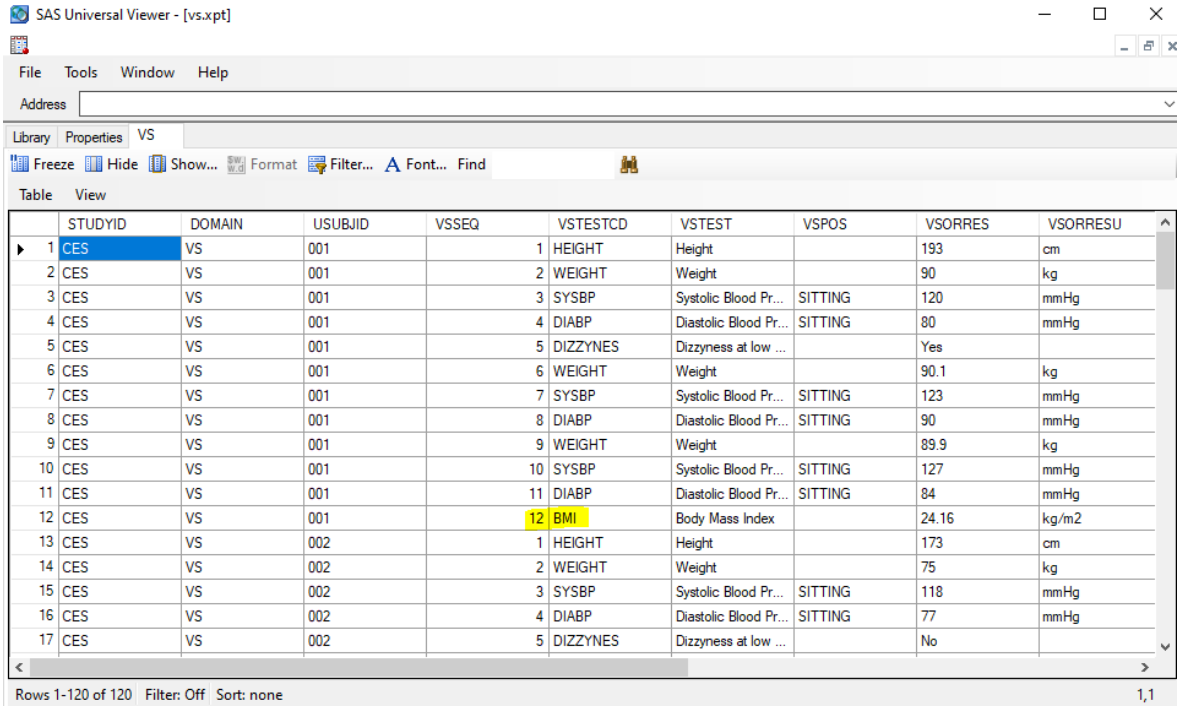
Now that we gave 2 result sets, we want to merge them into one. We can best do that during SAS-XPT generation, so when setting up the mapping executions, we select "Save SDTM Result tables as:" and check the checkbox "SAS-XPT". We also select "Adapt Variable Length for longest result value".
Very important now is to also check the checkbox "Additionally generate a merged dataset for 'split' domain datasets", and the checkbox "Unique --SEQ values across 'split' domains:
(when checking one of these checkboxes, the system may ask for a little bit more detail):



And when then clicking "Execute Transformation on Clinical Data", the usual tables will be generated, but also the SAS-XPT datasets, in our case in the D:\temp directory:



Essentially, as we only want to keep the "merged" VS dataset, we can omit the "vsbm.xpt" and "vsor.xpt" dataset. The "merged" vs.xpt dataset then, when opened in the "Universal SAS Viewer", looks like:



	STUDYID	DOMAIN	USUBJID	VSSEQ	VSTESTCD	VSTEST	VSPOS	VSORRES	VSORRESU
1	CES	VS	001	1	HEIGHT	Height		193	cm
2	CES	VS	001	2	WEIGHT	Weight		90	kg
3	CES	VS	001	3	SYSBP	Systolic Blood Pr...	SITTING	120	mmHg
4	CES	VS	001	4	DIABP	Diastolic Blood Pr...	SITTING	80	mmHg
5	CES	VS	001	5	DIZZYNES	Dizzyness at low ...		Yes	
6	CES	VS	001	6	WEIGHT	Weight		90.1	kg
7	CES	VS	001	7	SYSBP	Systolic Blood Pr...	SITTING	123	mmHg
8	CES	VS	001	8	DIABP	Diastolic Blood Pr...	SITTING	90	mmHg
9	CES	VS	001	9	WEIGHT	Weight		89.9	kg
10	CES	VS	001	10	SYSBP	Systolic Blood Pr...	SITTING	127	mmHg
11	CES	VS	001	11	DIABP	Diastolic Blood Pr...	SITTING	84	mmHg
12	CES	VS	001	12	BMI	Body Mass Index		24.16	kg/m2
13	CES	VS	002	1	HEIGHT	Height		173	cm
14	CES	VS	002	2	WEIGHT	Weight		75	kg
15	CES	VS	002	3	SYSBP	Systolic Blood Pr...	SITTING	118	mmHg
16	CES	VS	002	4	DIABP	Diastolic Blood Pr...	SITTING	77	mmHg
17	CES	VS	002	5	DIZZYNES	Dizzyness at low ...		No	

also containing the BMI records.

Remark: things may get a bit more complicated when e.g. the "height" is only measured once during the screening visit (as it is not expected to change during the course of the study) and the weight is measured during each visit.
In such a case one may think about making "Height" a "Global Variable", so that it can be reused in all the visits. For more information about "Global Variables", see the tutorial "Generating mappings for the SV domain & creating global variables for reuse"

IMPORTANT: Essentially, data points like for BMI should be calculated in the EDC system itself. The presented solution here to do it in SDTM-ETL itself, is just a "workaround". Reason is that SDTM-ETL uses ODM as the source, which normally only contains data points for which there are values.