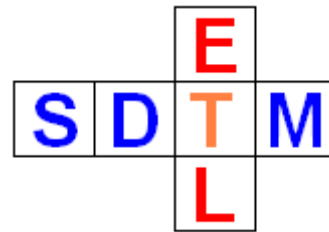


# SDTM-ETL 5.x: Derivation of data points that were not collected

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 "ODM Merger", that can be obtained together with the "ODM Generator". 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:

```
1 # Mapping using ODM element ItemData with ItemOID I_HEIGHT - value from attribu
2 # Generalized for all StudyEvents
3 # Generalized for all Forms within the StudyEvent
4 # Generalized for all ItemGroups within the Form
5 # Generalized for all Items within the ItemGroup
6 # Mapping for ODM Items [I_DIABP, I_DIZZY, I_HEIGHT, I_SYSBP, I_WEIGHT] to SDTM
7 # with CodeList OID 'CL.C66741.VSTESTCD.SUBSET'
8 $CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='BASELINE' or @StudyEventOID=
9 if ($CODEDVALUE == 'I_DIABP') {
10 $NEWCODEDVALUE = 'DIABP';
11 } elseif ($CODEDVALUE == 'I_DIZZY') {
12 $NEWCODEDVALUE = 'DIZZYNES';
13 } elseif ($CODEDVALUE == 'I_HEIGHT') {
14 $NEWCODEDVALUE = 'HEIGHT';
15 } elseif ($CODEDVALUE == 'I_SYSBP') {
16 $NEWCODEDVALUE = 'SYSBP';
17 } elseif ($CODEDVALUE == 'I_WEIGHT') {
18 $NEWCODEDVALUE = 'WEIGHT';
19 } elseif ($CODEDVALUE == '') {
20 $NEWCODEDVALUE = '';
21 } else {
22 $NEWCODEDVALUE = 'NULL';
23 }
24 $VS.VSTESTCD = $NEWCODEDVALUE;
```

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:

FA	STUDYID	DOMAIN	USUBJID	FA.FASEQ	FA.FAGRPID	FA.FASPID	FA.FATESTCD	FA.FATES
SR	STUDYID	DOMAIN	USUBJID	SR.SRSEQ	SR.SRGRPID	SR.SRREFID	SR.SRSPID	SR.SRTE
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	RELTYPE	RELID	
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL	QVAL
CES:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTCT	DM.RFENDTCT	DM.RFXSTDTCT	DM.RFXE
CES:LB	STUDYID	DOMAIN	USUBJID	LB.LBSEQ	LB.LBGRPID	LB.LBREFID	LB.LBSPID	LB.LBTE
CES:VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD	VS.VSTES

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:

PP	STUDYID	DOMAIN	USUBJID	PP.PPSEQ	PP.PPGRPID
PE	ST				PEGRPID
QS	ST				QSGRPID
RP	ST				RPGRPID
SC	ST				SCGRPID
SS	ST				SSGRPID
TU	ST				TUGRPID
TR	ST				TRGRPID
RS	STUDYID	DOMAIN	USUBJID	RS.RSSEQ	RS.RSGRPID
VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID
FA	STUDYID	DOMAIN	USUBJID	FA.FASEQ	FA.FAGRPID
SR	STUDYID	DOMAIN	USUBJID	SR.SRSEQ	SR.SRGRPID
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL
CES:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDC
CES:LB	STUDYID	DOMAIN	USUBJID	LB.LBSEQ	LB.LBGRPID
CES:VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID
CES:VSAA	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID

Do you want to create another instance of domain CES:VS ?

Yes No Cancel

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:

SR	STUDYID	DOMAIN	USUBJID	SR.SRSEQ	SR.SRGRPID	SR.SRREFID	SR.SRSPID	SI
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	RELTYPE	RELID	
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL	Q
CES:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDC	DM.RFENDTC	DM.RFXSTDC	D
CES:LB	STUDYID	DOMAIN	USUBJID	LB.LBSEQ	LB.LBGRPID	LB.LBREFID	LB.LBSPID	L
CES:VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD	V
CES:VSAA	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD	V

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:

Edit properties for SDTM dataset/domain VS with OID CES:VS

Name : VS

OID : CES:VS

Domain: VS

SAS Dataset Name: VS

Purpose : Tabulation

Comment:

External document for comment

to e.g.:

Edit properties for SDTM dataset/domainVS with OID CES:VS

**Name :** VSOR

**OID :** CES:VSOR

**Domain:** VS

**SAS Dataset Name:** VSOR

**Purpose :** Tabulation

**Comment:**

External document for comment

just by typing in the upper field ...

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

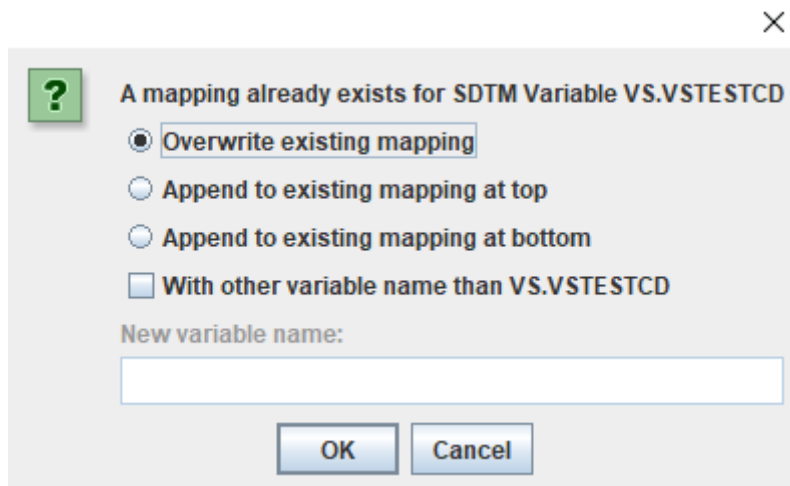
We then do something similar for VSAA" which we rename e.g. to "VSBM", standing for "VS BMI". Notice that it is "best practice" to limit data set names, even when they are just temporary, to 4 characters.

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":

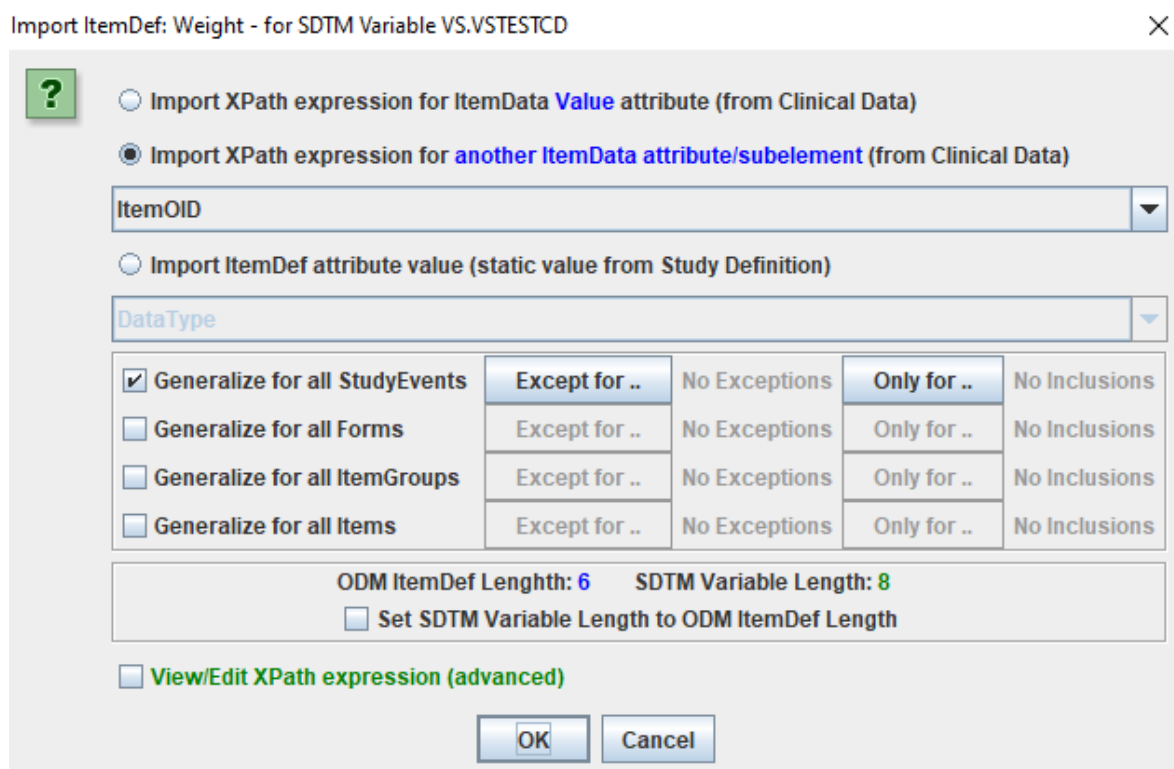
HS	STUDYID	DOMAIN	USUBJID	HS HSSEQ	HS HSGRPID	HS HSREFID	HS HSPID	HS HSTESTCD	HS HSTEST
DS	STUDYID	DOMAIN	USUBJID	DS DSSEQ	DS DSGRPID	DS DSREFID	DS DSPID	DS DSTESTCD	DS DSTEST
MH	STUDYID	DOMAIN	USUBJID	MH MHSEQ	MH MHGRPID	MH MHREFID	MH MHPID	MH MHTESTCD	MH MHTEST
DA	STUDYID	DOMAIN	USUBJID	DA DASEQ	DA DAGRPID	DA DAREFID	DA DASPID	DA DATES	DA DATES
DD	STUDYID	DOMAIN	USUBJID	DD DDSEQ	DD DDTESTCD	DD DDTEST	DD DDORRES	DD DDST	DD DDST
EG	STUDYID	DOMAIN	USUBJID	EG EGSEQ	EG EGGRPID	EG EGREFID	EG EGSPID	EG EGTESTCD	EG EGTEST
IE	STUDYID	DOMAIN	USUBJID	IE IESEQ	IE IEGRPID	IE IEREFID	IE IESPID	IE IETESTCD	IE IETEST
IS	STUDYID	DOMAIN	USUBJID	IS ISSEQ	IS ISGRPID	IS ISREFID	IS ISSPID	IS ISTESTCD	IS ISTEST
LB	STUDYID	DOMAIN	USUBJID	LB LBSEQ	LB LBGRPID	LB LBREFID	LB LBSPID	LB LBTESTCD	LB LBTEST
MB	STUDYID	DOMAIN	USUBJID	MB MBSEQ	MB MBGRPID	MB MBREFID	MB MBSPID	MB MBTESTCD	MB MBTEST
MS	STUDYID	DOMAIN	USUBJID	MS MSSEQ	MS MSGRPID	MS MSREFID	MS MSSPID	MS MSTESTCD	MS MSTEST
MI	STUDYID	DOMAIN	USUBJID	MI MISEQ	MI MIGRPID	MI MIREFID	MI MISPID	MI MITESTCD	MI MITEST
MO	STUDYID	DOMAIN	USUBJID	MO MOSEQ	MO MOGRPID	MO MOREFID	MO MOSPID	MO MOLT	MO MOLT
PC	STUDYID	DOMAIN	USUBJID	PC PCSEQ	PC PCGRPID	PC PCREFID	PC PCSPID	PC PCTESTCD	PC PCTEST
PP	STUDYID	DOMAIN	USUBJID	PP PPSEQ	PP PPGRPID	PP PPREFID	PP PPSPID	PP PPTESTCD	PP PPTEST
PE	STUDYID	DOMAIN	USUBJID	PE PESEQ	PE PEGRPID	PE PESPID	PE PETESTCD	PE PETEST	PE PETEST
OS	STUDYID	DOMAIN	USUBJID	OS OSSEQ	OS OSGRPID	OS OSSPID	OS OSTESTCD	OS OSTEST	OS OSTEST
RP	STUDYID	DOMAIN	USUBJID	RP RPSEQ	RP RPGRPID	RP RPREFID	RP RPSPID	RP RPTESTCD	RP RPTEST
SC	STUDYID	DOMAIN	USUBJID	SC SCSEQ	SC SCGRPID	SC SCREFID	SC SCSPID	SC SCTESTCD	SC SCTEST
SS	STUDYID	DOMAIN	USUBJID	SS SSSEQ	SS SSGRPID	SS SSSPID	SS SSTESTCD	SS SSTEST	SS SSTEST
TU	STUDYID	DOMAIN	USUBJID	TU TUSEQ	TU TUGRPID	TU TUREFID	TU TUSPID	TU TULN	TU TULN
TR	STUDYID	DOMAIN	USUBJID	TR TRSEQ	TR TRGRPID	TR TRREFID	TR TRSPID	TR TRLN	TR TRLN
RS	STUDYID	DOMAIN	USUBJID	RS RSSEQ	RS RSGRPID	RS RSREFID	RS RSSPID	RS RSLN	RS RSLN
VS	STUDYID	DOMAIN	USUBJID	VS VSSEQ	VS VSGRPID	VS VSPID	VS VSTESTCD	VS VSTEST	VS VSTEST
FA	STUDYID	DOMAIN	USUBJID	FA FASEQ	FA FAGRPID	FA FAREFID	FA FATESTCD	FA FATEST	FA FATEST
SR	STUDYID	DOMAIN	USUBJID	SR SRSEQ	SR SRGRPID	SR SRREFID	SR SRSPID	SR SRTESTCD	SR SRTEST
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	RELTYPE	RELID		
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL	QVAL	
CES.DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM RFXSTDC	DM RFXENDTC	DM RFXSTDC	DM RFXENDTC	DM RFXENDTC
CES.LB	STUDYID	DOMAIN	USUBJID	LB LBSEQ	LB LBGRPID	LB LBREFID	LB LBSPID	LB LBTESTCD	LB LBTEST
CES.VSOR	STUDYID	DOMAIN	USUBJID	VS VSSEQ	VS VSGRPID	VS VSPID	VS VSTESTCD	VS VSTEST	VS VSTEST
CES.VSBM	STUDYID	DOMAIN	USUBJID	VS VSSEQ	VS VSGRPID	VS VSPID	VS VSTESTCD	VS VSTEST	VS VSTEST

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:

**?** The system found **1** ODM Items which can be mapped to the SDTM CodeList **CL.C66741.VSTESTCD.SUBSET**.

Do you want to use the **mapping wizard** to provide such a mapping?

Or do you want a **template script** will be generated that you need to fill in, in order to categorize the data?

You can also choose to **ignore the CodeList** for now, then no codelist mapping is performed at all.

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:

```
The Transformation Script
1 # Mapping using ODM element ItemData with ItemOID I_WEIGHT - value from attribute ItemOID
2 # Generalized for all StudyEvents
3 $CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE']/I
4 # Template mapping script to be completed
5 if($CODEDVALUE = 'I_WEIGHT') {
6     $VS.VSTESTCD = '';
7 } else {
8     $VS.VSTESTCD = '';
9 }
10
11
```

which we then change into:

```
The Transformation Script
1 # Mapping using ODM element ItemData with ItemOID I_WEIGHT - value from attribute ItemOID
2 # Generalized for all StudyEvents
3 $CODEDVALUE = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_PE_BASE'];
4 # Template mapping script to be completed
5 if($CODEDVALUE = 'I_WEIGHT') {
6     $VS.VSTESTCD = 'BMI';
7 }
8
9
10
11
```

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:

```
The Transformation Script
1 # Mapping using the decode() function on codelist CL.C66741.VSTESTCD.SUBSET of variable VS.VSTESTCD
2 $TEST = decode($VS.VSTESTCD, 'CL.C66741.VSTESTCD.SUBSET', '');
3 if($VS.VSTESTCD = 'DIZZYNES') {
4     $VS.VSTESTI = 'Dizziness at low diastolic blood pressure';
5 } else {
6     $VS.VSTESTI = $TEST;
7 }
```

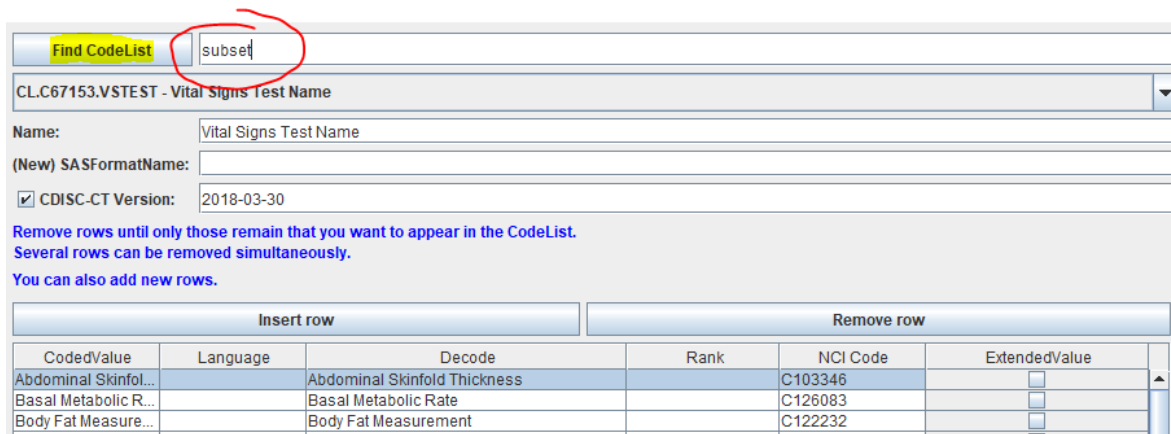
which, as we only have "BMI" now, we can reduce to:

```
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
```

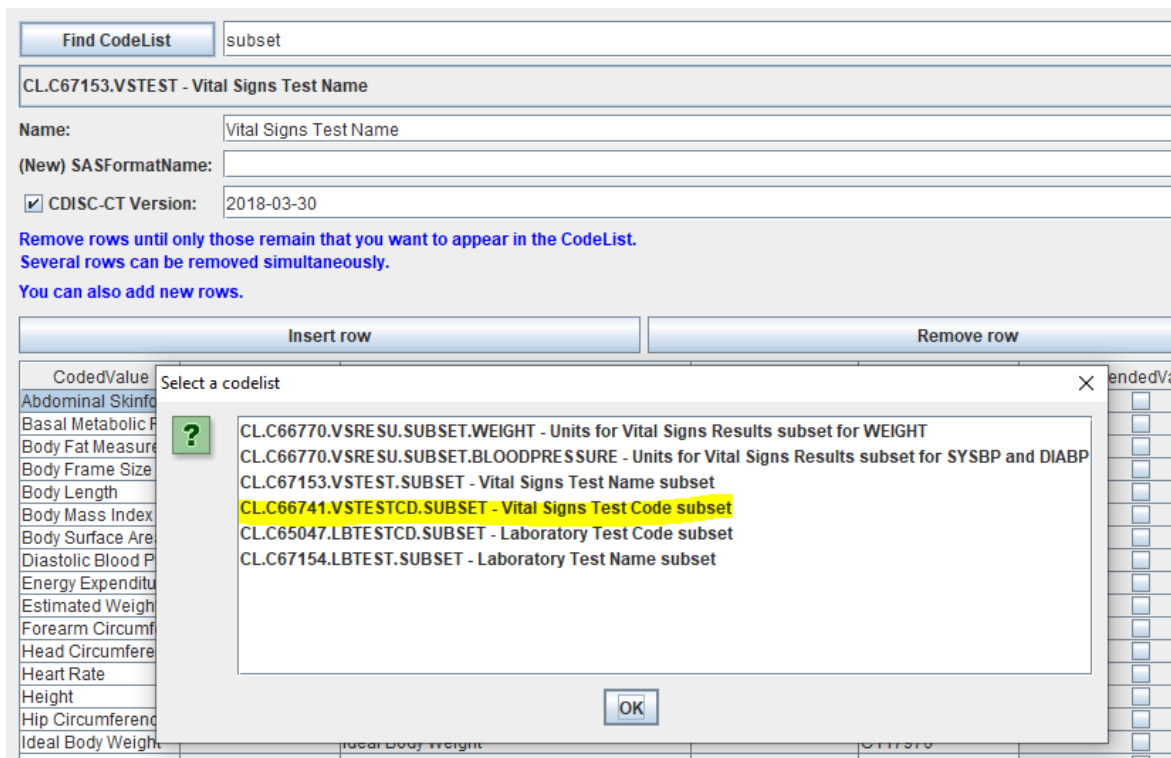
As one sees, in the original mapping without BMI, we generated a "subset codelist", that we use for "decoding" the value of VSTESTCD to VSTEST, but of course, it does not contain "BMI", as we did not foresee that at that time.

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 Terminology, 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:

The screenshot shows a software interface with a tree view on the left and a table on the right. The tree view is expanded to 'ItemGroupDef: Physical Exam' and includes 'ItemDef: Height' and 'ItemDef: Weight'. A red arrow points from 'ItemDef: Weight' to a row in the table. The table contains various codes and their corresponding values.

DA.DATEST	DA.DACAT	DA.DASCAT	DA.DAORRES	DA.DAORRESU	DA.DAORRESUR
DD.DDRESCAT	DD.DDEVAL	DD.DDDTC	DD.DDDY		
EG.EGTEST	EG.EGCAT	EG.EGSCAT	EG.EGPOS	EG.EGORRES	EG.EGORRESU
IE.IEASCAT	IE.IEORRES	IE.IESTRESC	IE.VISITNUM	IE.VISIT	IE.VISITRESU
IS.ISTEST	IS.ISCAT	IS.ISSCAT	IS.ISORRES	IS.ISORRESU	IS.ISORRESUR
LB.LBTEST	LB.LBCAT	LB.LBSCAT	LB.LBORRES	LB.LBORRESU	LB.LBORRESUR
MB.MBTEST	MB.MBCAT	MB.MBSCAT	MB.MBORRES	MB.MBORRESU	MB.MBORRESUR
MS.MSTEST	MS.MSCAT	MS.MSSCAT	MS.MSORRES	MS.MSORRESU	MS.MSORRESUR
MI.MITEST	MI.MITSTDTL	MI.MICAT	MI.MISCAT	MI.MIORRES	MI.MIORRESU
MO.MOTESTCD	MO.MOTEST	MO.MOCAT	MO.MOSCAT	MO.MOPOS	MO.MOPOSU
PC.PCTEST	PC.PCCAT	PC.PCSCAT	PC.PCORRES	PC.PCORRESU	PC.PCORRESUR
PP.PPSCAT	PP.PPORRES	PP.PPORRESU	PP.PPSTRESC	PP.PPSTRESN	PP.PPSTRESUR
PE.PEMODIFY	PE.PECAT	PE.PESCAT	PE.PEBODSYS	PE.PEORRES	PE.PEORRESU
QS.QSCAT	QS.QSSCAT	QS.QSORRES	QS.QSORRESU	QS.QSSTRESC	QS.QSSTRESN
RP.RRTEST	RP.RPCAT	RP.RPSCAT	RP.RPORRES	RP.RPORRESU	RP.RPORRESUR
SC.SCAT	SC.SCSCAT	SC.SCORRES	SC.SCORRESU	SC.SCSTRESC	SC.SCSTRESN
SS.SSCAT	SS.SSSCAT	SS.SSORRES	SS.SSSTRESC	SS.SSSTAT	SS.SSSTATU
TU.TUTESTCD	TU.TUTEST	TU.TUORRES	TU.TUSTRESC	TU.TUNAM	TU.TUNAMU
TR.TRLNKGRP	TR.TRTESTCD	TR.TRTEST	TR.TRORRES	TR.TRORRESU	TR.TRORRESUR
RS.RSLNKGRP	RS.RSTESTCD	RS.RSTEST	RS.RSCAT	RS.RSORRES	RS.RSORRESU
VS.VSCAT	VS.VSSCAT	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSORRESUR
FA.FAOBJ	FA.FACAT	FA.FASCAT	FA.FAORRES	FA.FAORRESU	FA.FAORRESUR
SR.SRTEST	SR.SROBJ	SR.SRCAT	SR.SRSCAT	SR.SRORRES	SR.SRORRESU
QORIG	QVAL				
DM.RFICDTC	DM.RFPENDTC	DM.DTHDTC	DM.DTHFL	DM.SITEID	DM.SITEIDU
LB.LBTEST	LB.LBCAT	LB.LBSCAT	LB.LBORRES	LB.LBORRESU	LB.LBORRESUR
VS.VSCAT	VS.VSSCAT	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSORRESUR
VS.VSCAT	VS.VSSCAT	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSORRESUR

This leads to:

```
The Transformation Script
1 # Mapping using ODM element ItemData with ItemOID I_WEIGHT
2 # Generalized for all StudyEvents
3 $VS.VSORRES = xpath(/StudyEventData/FormData[@FormOID='F_BASELINE']/ItemGroupData[@ItemGroupOID='IG_FE_BASE']/ItemData[@ItemOID='I_WEIGHT']/@Value)
4
```

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:

```
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_FE_BASE']/ItemData[@ItemOID='I_WEIGHT']/@Value);
4
```

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

A dialog box titled "A mapping already exists for SDTM Variable VS.VSORRES". It contains three radio buttons: "Overwrite existing mapping", "Append to existing mapping at top", and "Append to existing mapping at bottom". The "Append to existing mapping at bottom" option is selected. There is a checked checkbox for "With other variable name than VS.VSORRES". Below this is a text field for "New variable name:" containing the text "HEIGHT". There are "OK" and "Cancel" buttons at the bottom.

leading to:

```

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']/ItemData[@ItemOID='I_WEIGHT']/@Value);
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']/ItemData[@ItemOID='I_HEIGHT']/@Value);
7

```

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
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']/ItemData[@ItemOID='I_WEIGHT']/@Value);
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']/ItemData[@ItemOID='I_HEIGHT']/@Value);
7 # BMI calculation, taking into account that "height" is provided in centimeters
8 $VS.VSORRES = 10000 * $WEIGHT / ($HEIGHT * $HEIGHT);
9

```

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
1 $VS.VSPOS = '';

```

and for VSORRESU, we also must change into:

```

The Transformation Script
1 $VS.VSORRESU = 'kg/m2';
2
3

```

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:

CES:DM	CES:LB	CES:VSOR	CES:VSMB								
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		23.3886865818771	kg/m2	23.3886865818771		
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	Returns the result of rounding the value in the first argument to number of decimals in the second argument		in-week	
time	hour-in-day	minute-in-h			atediff	
timediff	datetimediff	elementname	more date/time ...	RESTful WS	My Functions	

and after executing again, we get for CES:VSMB

CES:DM	CES:LB	CES:VSOR	CES:VSMB								
STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSPOS	VS.VSORRES	VS.VSORRESU	VS.VSSTRESC	VS.VSSTRESC	VS.VSSTRESC
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	Dizziness 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	Dizziness 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):

Execute Transformation (XSLT) Code

ODM file with clinical data:  
 D:\SDTM-ETL\TestFiles\ODM1-3-1\CES\_ClinicalData\_LOINC\_more\_subjects.xml Browse...

MetaData in separate ODM file  
 D:\SDTM-ETL\TestFiles\ODM1-3-1\CES\_Metadadata\_LOINC.xml Browse...

Administrative data in separate ODM file  
 D:\SDTM-ETL\TestFiles\ODM1-3-1\CES\_Metadadata\_LOINC.xml Browse...

Perform post-processing for assigning --LOBXFL       Perform post-processing unscheduled VISITNUM  
 Split records > 200 characters to SUPP-- records  
 Move non-standard SDTM Variables to SUPP--       Move Comment Variables to Comments (CO) Domain  
 Move Relrec Variables to Related Records (RELREC) domain       Try to generate 1:N RELREC Relationships  
 View Result SDTM tables       Adapt Variable Length for longest result value  
 Generate 'NOT DONE' records for QS datasets       Re-sort records using define.xml keys  
 Unique --SEQ values across 'split' domains       Perform CDISC CORE validation on generated SDTM files  
 Save Result SDTM tables as:  
 Dataset-JSON 1.1     SAS-XPT     UTF-8 encoded CSV     SQL INSERT statements

SDTM export files directory:  
 D:\temp Browse...

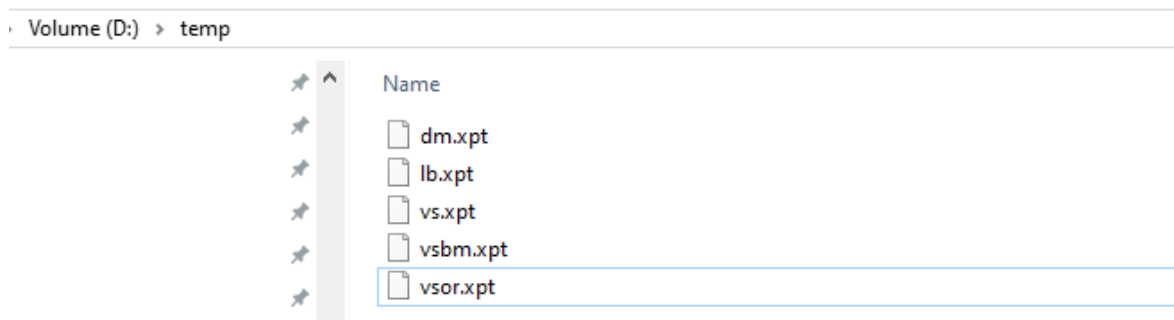
Add location of generated SDTM files to define.xml       Store link as relative path  
 Additionally generate a merged dataset for 'split' domain datasets

Messages and error messages:  
 Messages:

**Execute Transformation on Clinical Data**

Close

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:

SAS Universal Viewer - [vs.xpt]

File Tools Window Help

Address

Library Properties VS

Freeze Hide Show... Format Filter... Font... Find

Table View

	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	Dizziness 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	Dizziness at low ...		No	

Rows 1-120 of 120 Filter: Off Sort: none 1,1

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.