SDTM-ETL 4.x User Manual and Tutorial

Author: Jozef Aerts, XML4Pharma

Last update: 2022-06-22

Tutorial: Generating baseline flags

Table of Contents

Introduction

Until SDTM 1.4 (SDTM-IG 3.2) the concept of Baseline Flag (--BLFL variables) was poorly defined. The IG states "*Indicator used to identify a baseline value. The value should be Y or null*", but doesn't state how it should be derived or assigned. Essentially this means that sponsors can assign or derive baseline flags in any way they want.

As of SDTM 1.5, a new baseline flag variable --LOBXFL "*Last Observation before first Exposure Flag*" has been introduced with the definition "*Operationally-derived indicator used to identify the last non-missing value prior to RFXSTDTC. Should be Y or null.*"

Essentially, this variable should not be in SDTM, as such observations can easily be derived and highlighted by good review tools. See e.g. the article "Why --LOBXFL should not be in SDTM". However, this new variable has been added to SDTM on request of the FDA, probably because not all FDA reviewers have the capabilities to derive such records using their review tools.

In this tutorial, we will explain how such baseline flags can be derived or generated. We will do this using VSBLFL and VSLOBXFL as an example.

The simple and easy way: Baseline flag based on the Visit Name or Visit Number

We see that in many submissions, the baseline flag (or last observation before medication flag) is simply based on the visit (StudyEvent) name or number. This however assumes that:

- there is a single visit (e.g. "Screening") that has measurements and ends with the first study treatment (e.g. first drug intake)
- It is clearly indicated in the protocol that no measurements may be performed anymore within this visit after the subject received the first study treatment (e.g. first drug intake)
- Alternatively, there is a clearly protocol-defined visit with measurements that is the last visit before first study treatment
- The actions of the investigator conforms to the protocol and does not perform any measurements after first study treatment within the same visit of first study treatment

In such cases, the baseline flag can simply be set by a "drag-and-drop" from the ODM "StudyEvent", setting "Generalize for all StudyEvents" and getting the identifier (OID) of the visit and then adding it to a little script like:

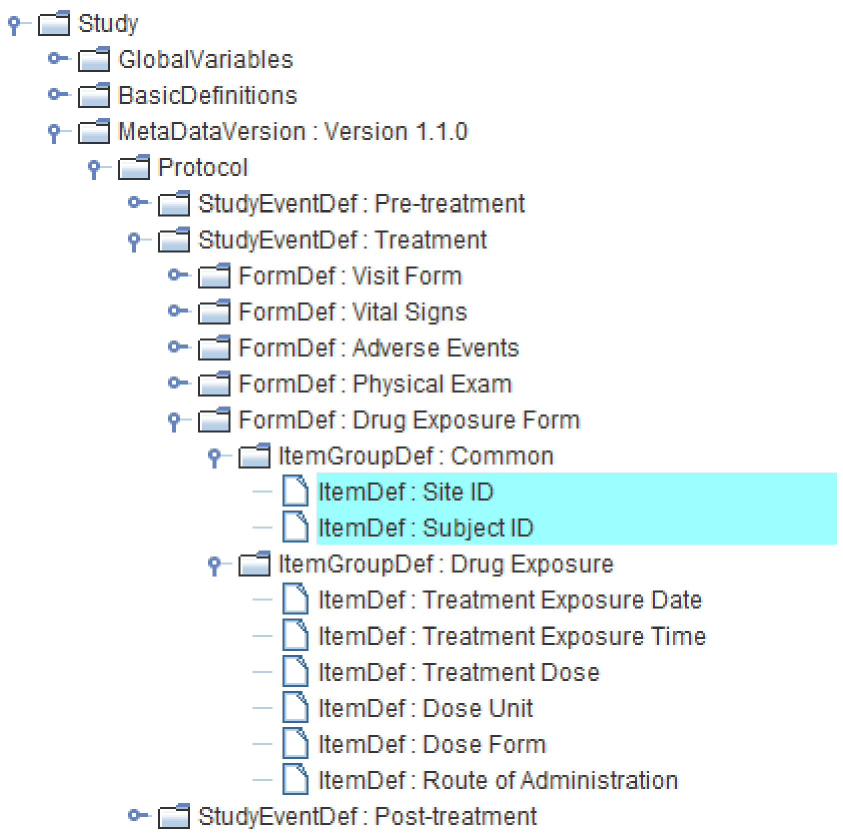
```
$VISITOID = xpath(/StudyEventData/@StudyEventOID);
if ($VISITOID == 'SE.SCREEN') {
        $VS.VSBLFL = 'Y';
} else {
        $VS.VSBLFL = '';
}
```

This is very similar to getting visit number and name, as explained in the other tutorials.

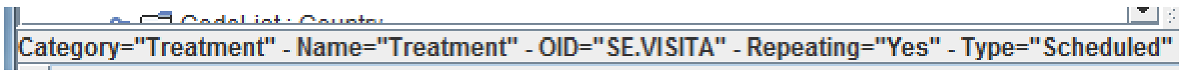
Getting the date/time of first study treatment

First of all, we need to generate RFXSTDTC in DM ("*Date/Time of First Study Treatment*") which needs to be delivered in ISO-8601 format. RFXSTDTC can either be a date or a datetime. In many clinical studies, we (still) see that first exposure is (unfortunately) captured as a date.

For this tutorial, we need the sample ODM 1.3 file "MyStudyNew_ODM_1_3.xml". When loaded into SDTM-ETL, we can see the following tree structure:

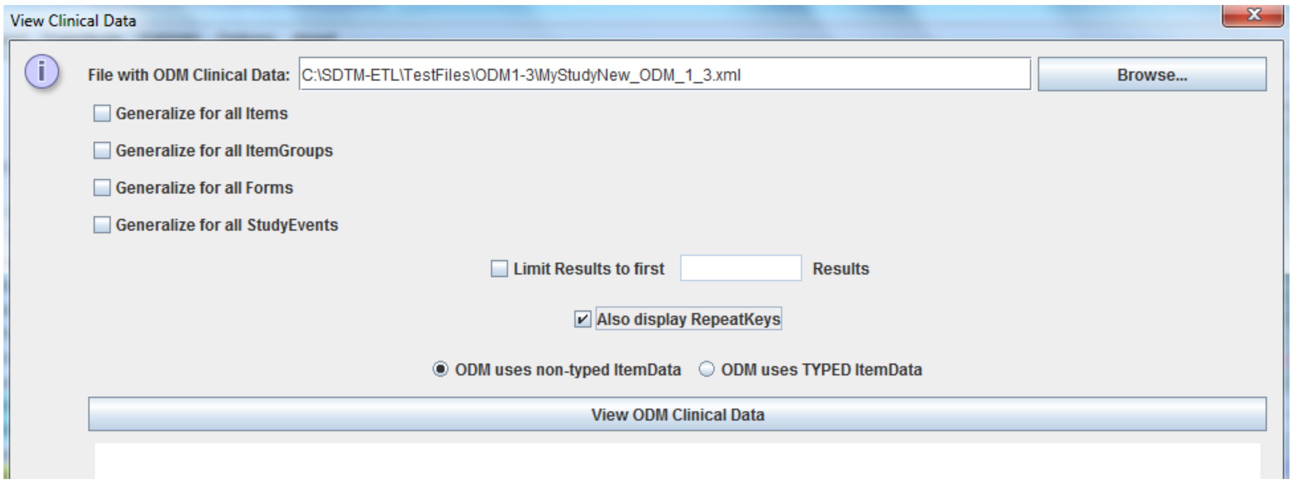

Study
GlobalVariables
BasicDefinitions
MetaDataVersion : Version 1.1.0
Protocol
StudyEventDef : Pre-treatment
StudyEventDef : Treatment
FormDef : Visit Form
FormDef : Vital Signs
FormDef : Adverse Events
FormDef : Physical Exam
FormDef : Drug Exposure Form
ItemGroupDef : Common
ItemDef : Site ID
ItemDef : Subject ID
ItemGroupDef : Drug Exposure
ItemDef : Treatment Exposure Date
ItemDef : Treatment Exposure Time
ItemDef : Treatment Dose
ItemDef : Dose Unit
ItemDef : Dose Form
ItemDef : Route of Administration
StudyEventDef : Post-treatment

We see that this simple demo study consists of 3 (types of) visits: a pre-treatment visit, a treatment visit and a post-treatment visit. When clicking on the "StudyEventDef" tree node "Treatment", we also see that the visit is "repeating", i.e. that there may be more than one occurrence of this visit:


Category="Treatment" - Name="Treatment" - OID="SE.VISITA" - Repeating="Yes" - Type="Scheduled"

Furthermore, we easily see that the form "Drug Exposure Form" has a "Treatment Exposure Date" and a "Treatment Exposure Time". These will be very useful later to populate the EX (and/or EC) dataset and for RFXSTDTC.

Before starting any mappings, we want to see what data points are available for the exposure dates and times. Select "Treatment Exposure Date" and then use the menu "View - ODM Clinical Data". Select your ODM file with clinical data. As the whole visit is repeating, also check the checkbox "Also display RepeatKeys":


View Clinical Data

File with ODM Clinical Data: C:\SDTM-ETL\TestFiles\ODM1-3\MyStudyNew_ODM_1_3.xml [Browse...]

☐ Generalize for all Items
☐ Generalize for all ItemGroups
☐ Generalize for all Forms
☐ Generalize for all StudyEvents

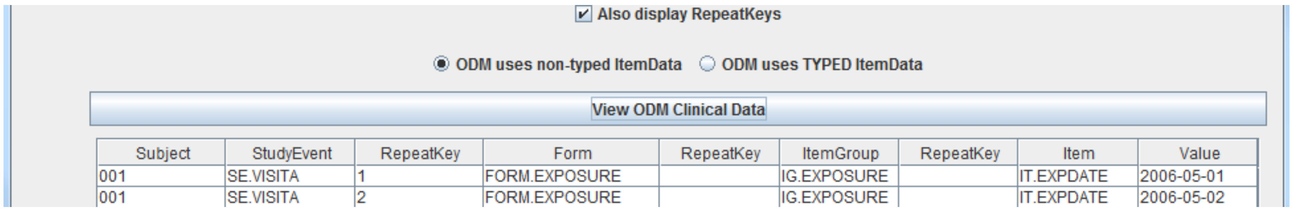
☐ Limit Results to first ___ Results

☑ Also display RepeatKeys

[✓ ODM uses non-typed ItemData] ODM uses TYPED ItemData

[View ODM Clinical Data]

After clicking the "View ODM Clinical Data" button, one gets:


☑ Also display RepeatKeys

[✓ ODM uses non-typed ItemData] ODM uses TYPED ItemData

[View ODM Clinical Data]

Subject	StudyEvent	RepeatKey	Form	RepeatKey	ItemGroup	RepeatKey	Item	Value
001	SE.VISITA	1	FORM.EXPOSURE		IG.EXPOSURE		IT.EXPDATE	2006-05-01
001	SE.VISITA	2	FORM.EXPOSURE		IG.EXPOSURE		IT.EXPDATE	2006-05-02

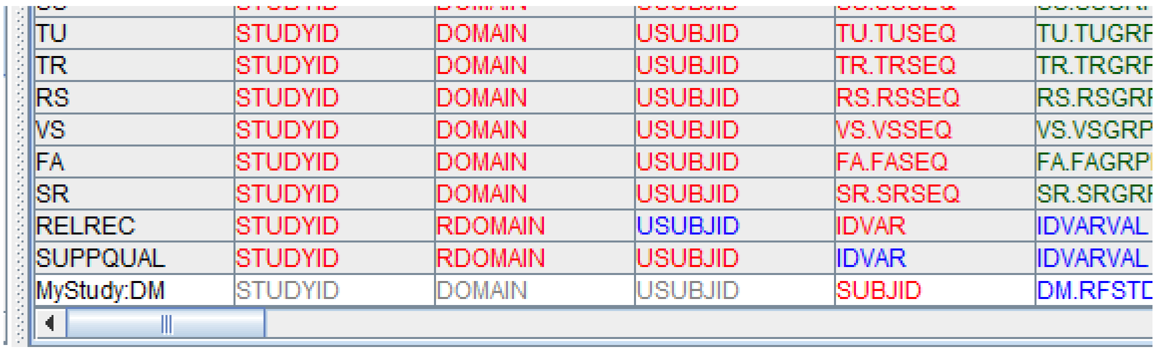
seeing that subject 001 has two exposures, one of May 1st 2006, and one on May 2nd 2006.
Also notice the third column "RepeatKey" which is the repeat number of the visit.

For RFXSTDTC, we of course only need the first.

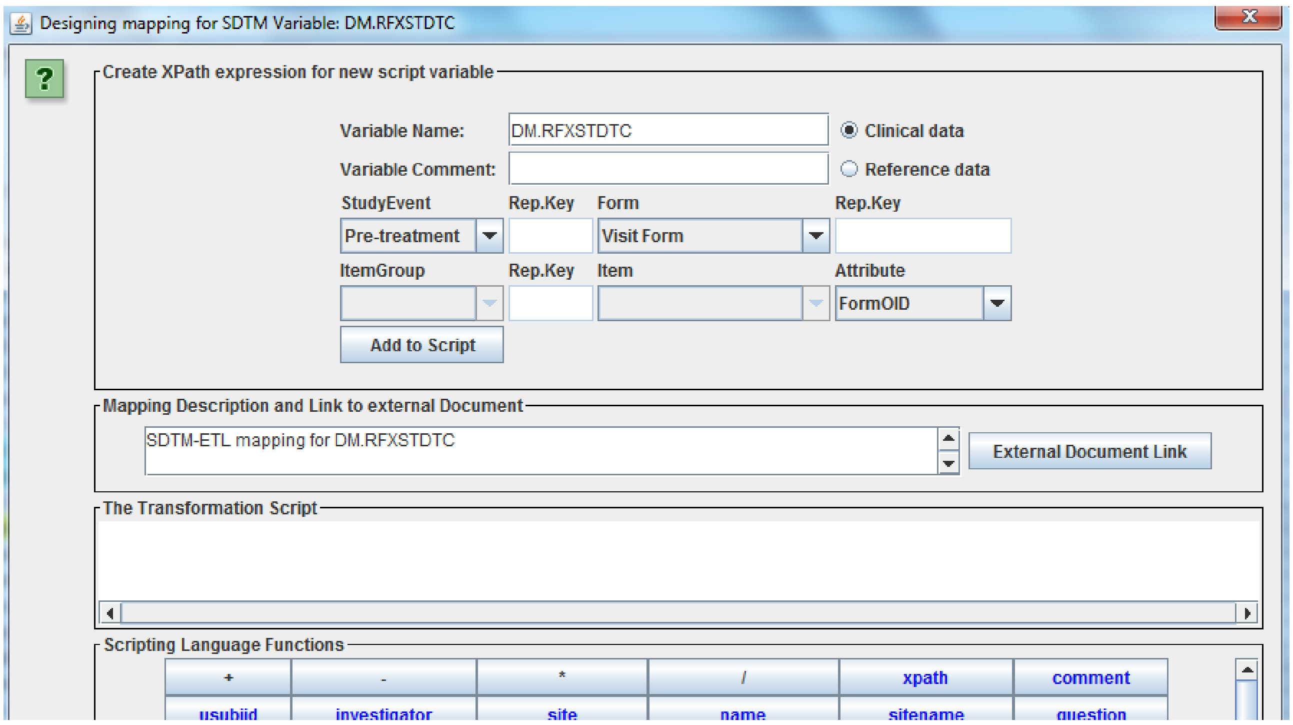
Now for generating the mapping to RFXSTDTC, there are several possibilities

A. Using the mapping wizard

After creating a study-specific copy of the DM domain (by dragging the DM row from the template to the bottom of the table):


TU	STUDYID	DOMAIN	USUBJID	TU.TUSEQ	TU.TUGRF
TR	STUDYID	DOMAIN	USUBJID	TR.TRSEQ	TR.TRGRF
RS	STUDYID	DOMAIN	USUBJID	RS.RSSEQ	RS.RSGRI
VS	STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSGRP
FA	STUDYID	DOMAIN	USUBJID	FA.FASEQ	FA.FAGRP
SR	STUDYID	DOMAIN	USUBJID	SR.SRSEQ	SR.SRGRI
RELREC	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL
SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL
MyStudy:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTD

double-click the cell "RFXSTDTC". The mapping window is displayed:


Designing mapping for SDTM Variable: DM.RFXSTDTC

Create XPath expression for new script variable

Variable Name: DM.RFXSTDTC [✓ Clinical data] Reference data

Variable Comment:

StudyEvent	Rep.Key	Form	Rep.Key
Pre-treatment		Visit Form	

ItemGroup	Rep.Key	Item	Attribute
			FormOID

Mapping Description and Link to external Document

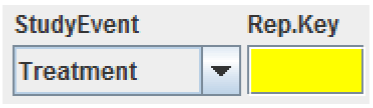
SDTM-ETL mapping for DM.RFXSTDTC

The Transformation Script

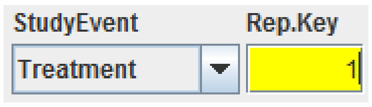
Scripting Language Functions

+	-	*	/	xpath	comment
usubjid	investigator	site	name	sitename	question

The upper part allows us to select a visit (StudyEvent), Form, ItemGroup (subform) and Item. For the date of the first exposure, we know we need to retrieve the data from the first "treatment visit", so we select:

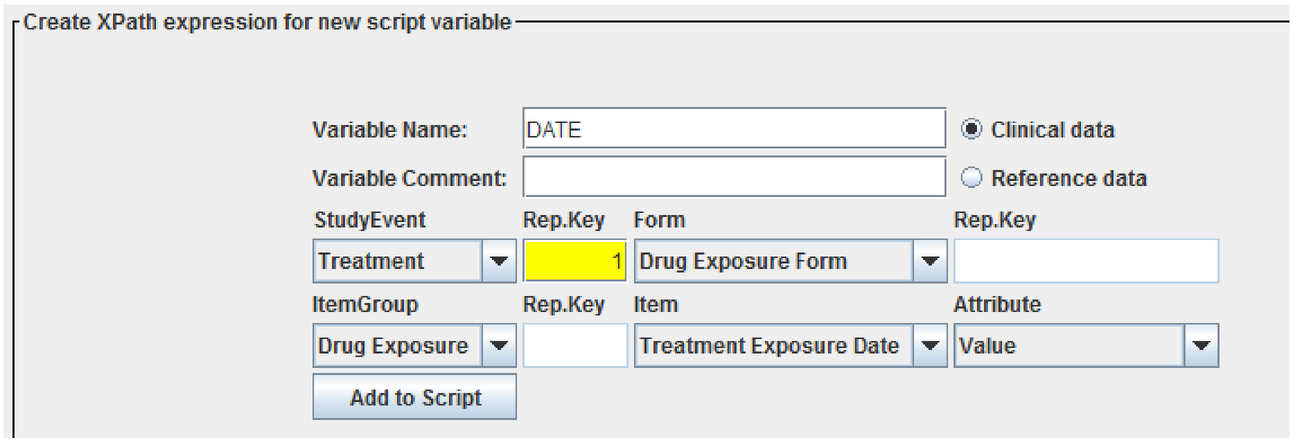


We see that the field "Rep.Key" is highlighted, as the "treatment visit" is repeating. As we need the first treatment visit, we need to fill "1" in this field:



Later we will see that there is another way of ensuring that the date of exposure from the first visit is selected.

We then continue with selecting the form, subform and item, and assign a variable name (e.g. "DATE") for this variable:

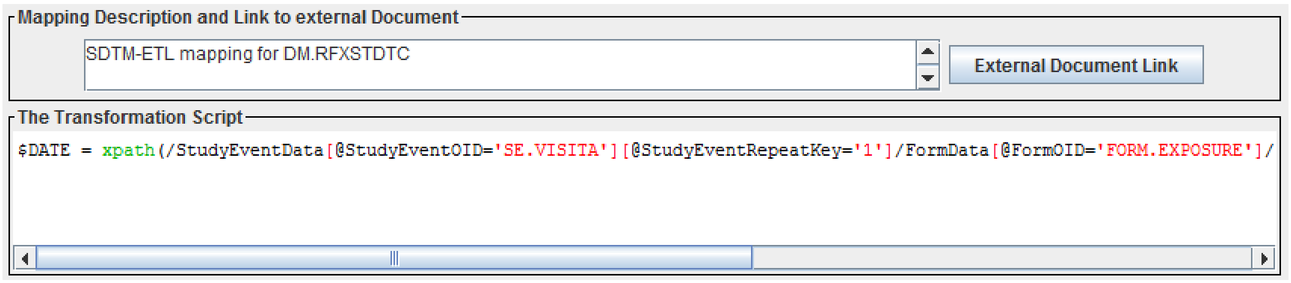

Create XPath expression for new script variable

Variable Name: DATE [✓ Clinical data] Reference data

Variable Comment:

StudyEvent	Rep.Key	Form	Rep.Key
Treatment	1	Drug Exposure Form	
ItemGroup	**Rep.Key**	**Item**	**Attribute**
Drug Exposure		Treatment Exposure Date	Value

Clicking the "Add to Script" button leads to:


Mapping Description and Link to external Document

SDTM-ETL mapping for DM.RFXSTDTC

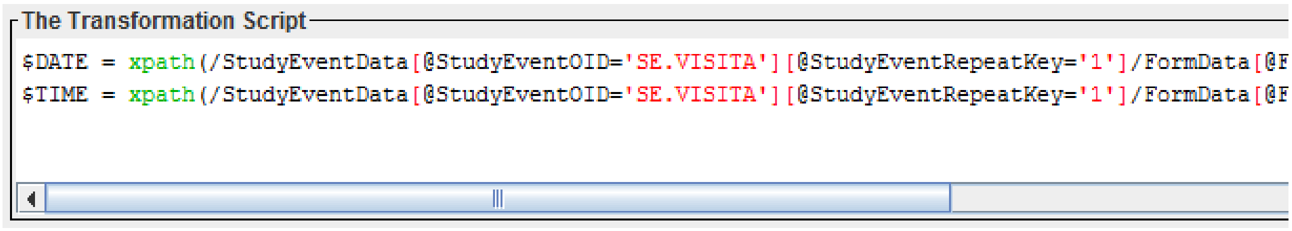
The Transformation Script

```
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][@StudyEventRepeatKey='1']/FormData[@FormOID='FORM.EXPOSURE']/
```

Notice the @StudyEventRepeatKey='1' in the XPath expression.

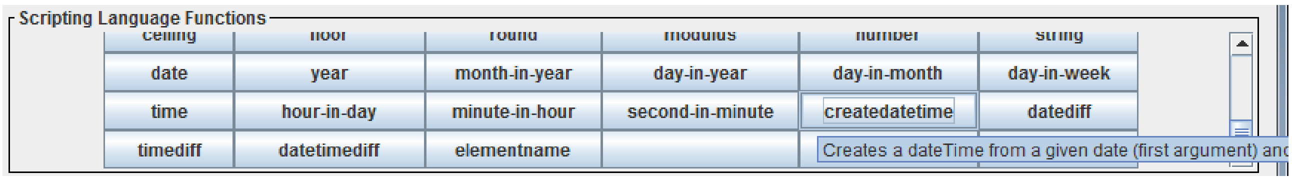
Now to the same for "Treatment Exposure Time" and name the variable "TIME":

leading to:


The Transformation Script

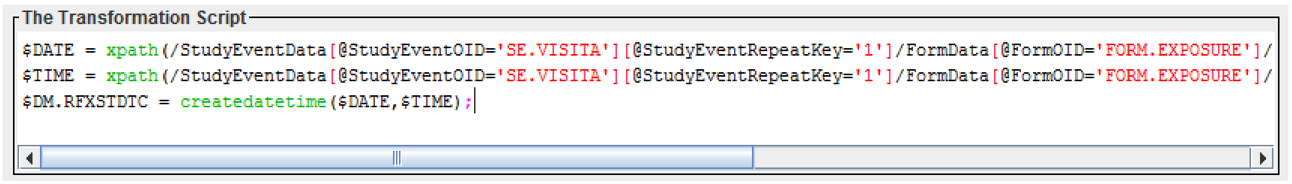
```
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][@StudyEventRepeatKey='1']/FormData[@F
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][@StudyEventRepeatKey='1']/FormData[@F
```

In the mapping script, we can now easily create a datetime in ISO-8601 format, using the function "createdatetime" (in the box with all function):


Scripting Language Functions

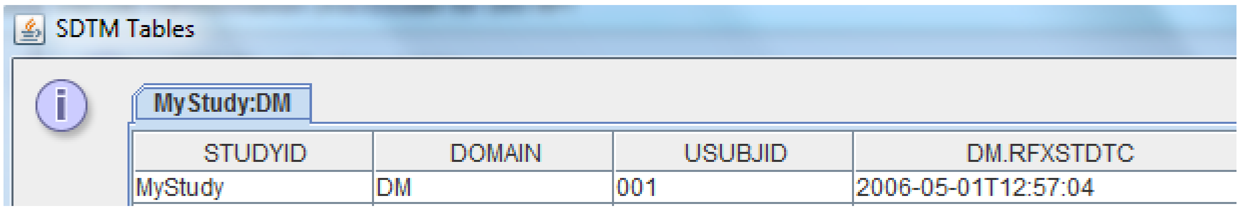
ceiling	floor	round	modulus	number	string
date	year	month-in-year	day-in-year	day-in-month	day-in-week
time	hour-in-day	minute-in-hour	second-in-minute	createdatetime	datediff
timediff	datetimediff	elementname			

Creates a dateTime from a given date (first argument) and


The Transformation Script

```
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][@StudyEventRepeatKey='1']/FormData[@FormOID='FORM.EXPOSURE']/
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][@StudyEventRepeatKey='1']/FormData[@FormOID='FORM.EXPOSURE']/
$DM.RFXSTDTC = createdatetime($DATE,$TIME);
```

and then complete the script to:
Testing leads to:



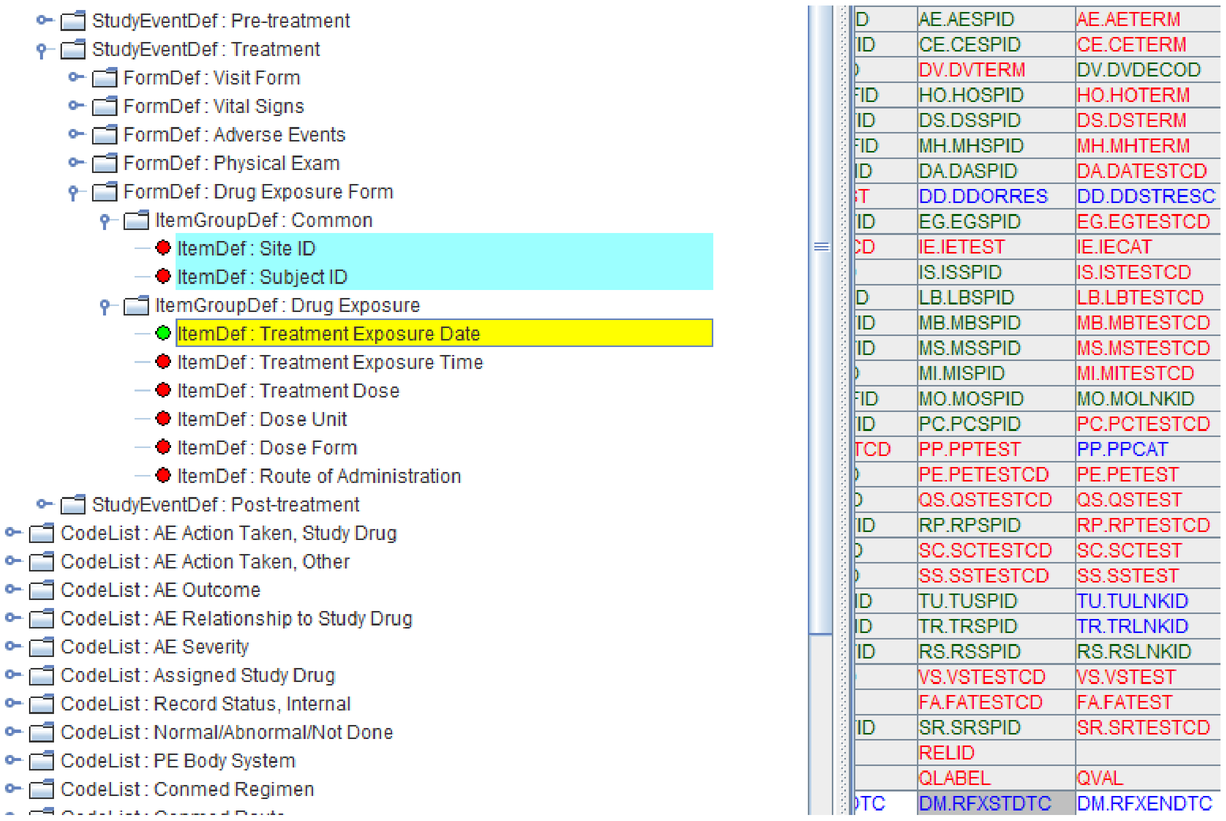
SDTM Tables

My Study:DM

STUDYID	DOMAIN	USUBJID	DM.RFXSTDTC
MyStudy	DM	001	2006-05-01T12:57:04

B. By drag-and-drop

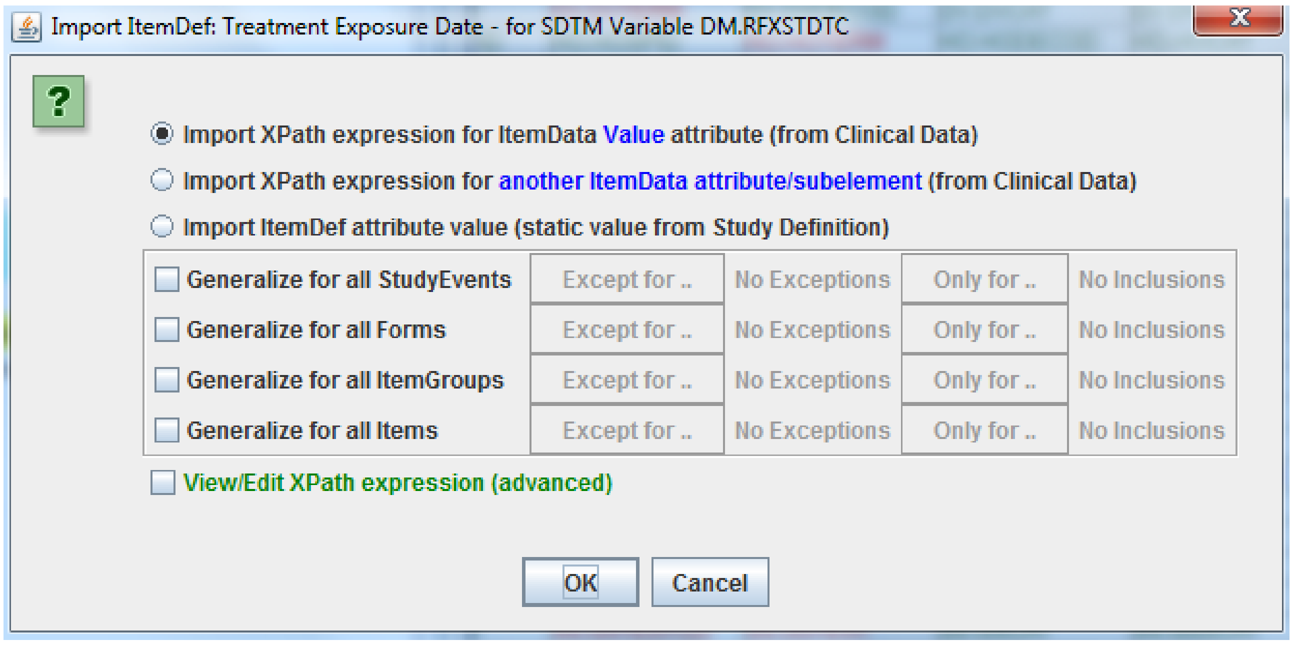
We can also use simple drag-and-drop for generating the mapping script for RFXSTDTC.



- StudyEventDef : Pre-treatment
- StudyEventDef : Treatment
  - FormDef : Visit Form
  - FormDef : Vital Signs
  - FormDef : Adverse Events
  - FormDef : Physical Exam
  - FormDef : Drug Exposure Form
    - ItemGroupDef : Common
      - ItemDef : Site ID
      - ItemDef : Subject ID
    - ItemGroupDef : Drug Exposure
      - ItemDef : Treatment Exposure Date
      - ItemDef : Treatment Exposure Time
      - ItemDef : Treatment Dose
      - ItemDef : Dose Unit
      - ItemDef : Dose Form
      - ItemDef : Route of Administration
- StudyEventDef : Post-treatment
- CodeList : AE Action Taken, Study Drug
- CodeList : AE Action Taken, Other
- CodeList : AE Outcome
- CodeList : AE Relationship to Study Drug
- CodeList : AE Severity
- CodeList : Assigned Study Drug
- CodeList : Record Status, Internal
- CodeList : Normal/Abnormal/Not Done
- CodeList : PE Body System
- CodeList : Conmed Regimen

D	AE.AESPID	AE.AETERM
ID	CE.CESPID	CE.CETERM
)	DV.DVTERM	DV.DVDECOD
FID	HO.HOSPID	HO.HOTERM
ID	DS.DSSPID	DS.DSTERM
FID	MH.MHSPID	MH.MHTERM
ID	DA.DASPID	DA.DATESTCD
T	DD.DDORRES	DD.DDSTRESC
ID	EG.EGSPID	EG.EGTESTCD
CD	IE.IETEST	IE.IECAT
	IS.ISSPID	IS.ISTESTCD
D	LB.LBSPID	LB.LBTESTCD
ID	MB.MBSPID	MB.MBTESTCD
ID	MS.MSSPID	MS.MSTESTCD
)	MI.MISPID	MI.MITESTCD
FID	MO.MOSPID	MO.MOLNKID
ID	PC.PCSPID	PC.PCTESTCD
TCD	PP.PPTEST	PP.PPCAT
)	PE.PETESTCD	PE.PETEST
)	QS.QSTESTCD	QS.QSTEST
ID	RP.RPSPID	RP.RPTESTCD
)	SC.SCTESTCD	SC.SCTEST
)	SS.SSTESTCD	SS.SSTEST
ID	TU.TUSPID	TU.TULNKID
ID	TR.TRSPID	TR.TRLNKID
ID	RS.RSSPID	RS.RSLNKID
	VS.VSTESTCD	VS.VSTEST
	FA.FATESTCD	FA.FATEST
ID	SR.SRSPID	SR.SRTESTCD
	RELID	
	QLABEL	QVAL
TC	DM.RFXSTDTC	DM.RFXENDTC

Simply drag-and-drop "Treatment Exposure Date" to the RFXSTDTC cell. This leads to:

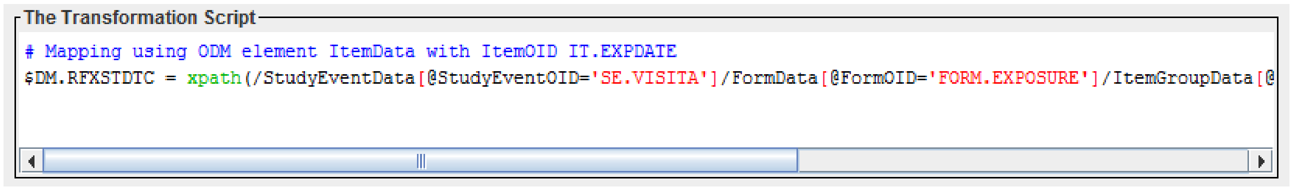

Import ItemDef: Treatment Exposure Date - for SDTM Variable DM.RFXSTDTC

[✓ Import XPath expression for ItemData Value attribute (from Clinical Data)]
Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
Import ItemDef attribute value (static value from Study Definition)

☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Items	Except for ..	No Exceptions	Only for ..	No Inclusions

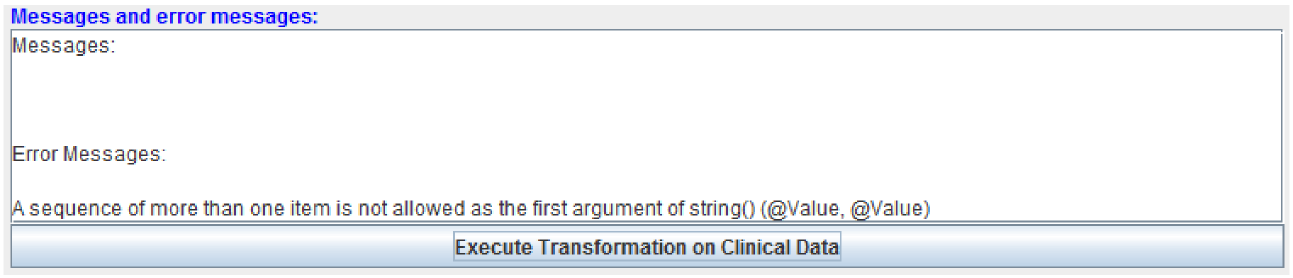
☐ View/Edit XPath expression (advanced)

and clicking "OK" to the following mapping script:


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DM.RFXSTDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@
```

When we however test this, we get:


Messages and error messages:

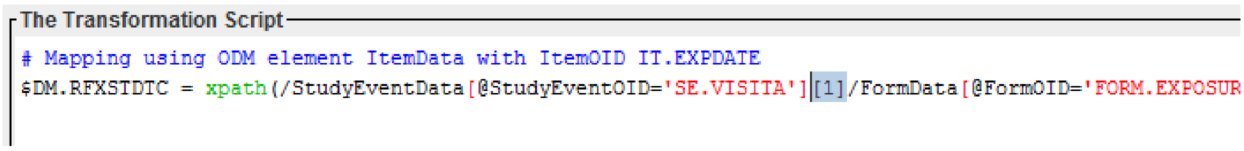
```
Messages:

Error Messages:

A sequence of more than one item is not allowed as the first argument of string() (@Value, @Value)
```

The reason is that the XPath retrieves ALL treatment dates, but we only want the first one, or better said, the one from the first visit.

We can easily solve this in several ways:

- add [@StudyEventRepeatKey='1'] to the XPath in the mapping script.
  This however already requires some knowledge of ODM and of XPath

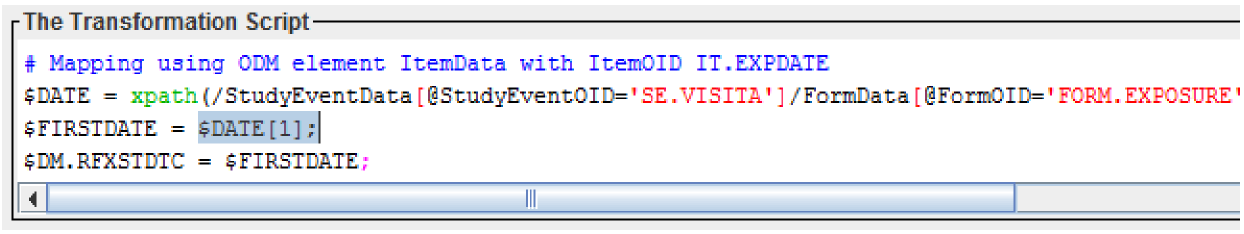
- Manually edit the XPath expression like:


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DM.RFXSTDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'][1]/FormData[@FormOID='FORM.EXPOSUR
```

Notice the [1] at the end of the "StudyEventData" part

- Use the following construct:

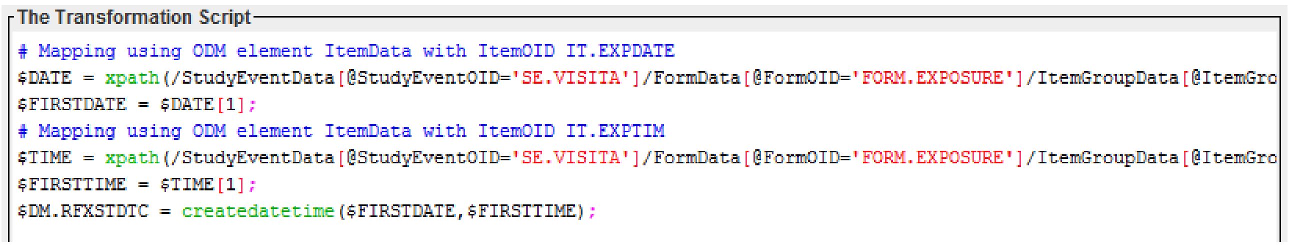


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE'
$FIRSTDATE = $DATE[1];
$DM.RFXSTDTC = $FIRSTDATE;
```

where first all treatment dates are retrieved, and then the first one encountered.

We can then do the same for the time of the first exposure, finally leading to the following script:

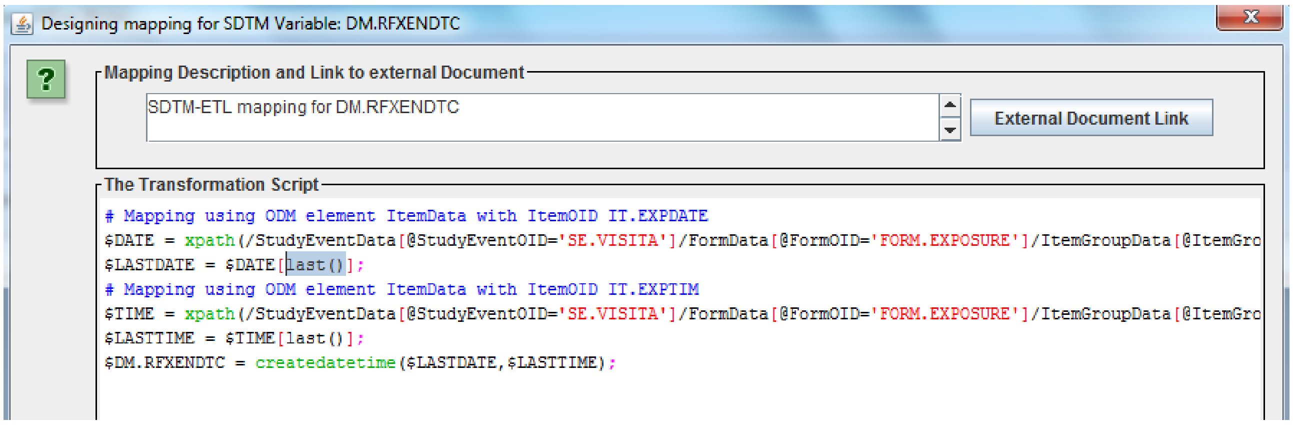


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@ItemGro
$FIRSTDATE = $DATE[1];
# Mapping using ODM element ItemData with ItemOID IT.EXPTIM
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@ItemGro
$FIRSTTIME = $TIME[1];
$DM.RFXSTDTC = createdatetime($FIRSTDATE,$FIRSTTIME);
```

Remark that this will only be correct when both date and time have been collected for the first exposure.

For RFXENDTC (last date/time) of exposure, we can use the construct:



Designing mapping for SDTM Variable: DM.RFXENDTC

Mapping Description and Link to external Document

SDTM-ETL mapping for DM.RFXENDTC

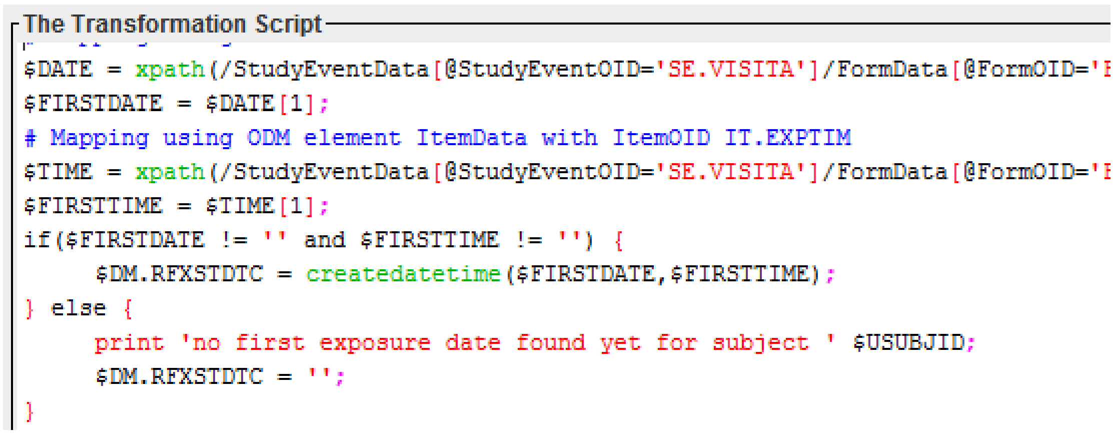
External Document Link

The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@ItemGro
$LASTDATE = $DATE[last()];
# Mapping using ODM element ItemData with ItemOID IT.EXPTIM
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@ItemGro
$LASTTIME = $TIME[last()];
$DM.RFXENDTC = createdatetime($LASTDATE,$LASTTIME);
```

Notice the "[last()] predicates. This construct is especially useful when we do not know the number of repeats of study treatments, for example when these differ between subjects.

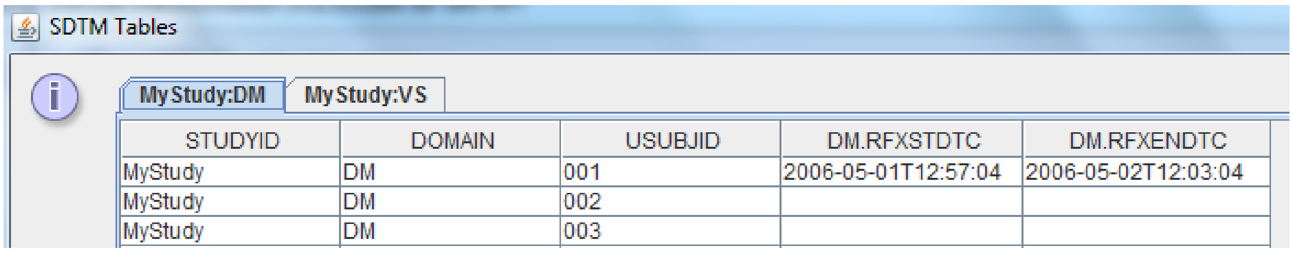
However …, suppose that we have subjects in the database (and so in the ODM) that have been included yet, but for which no exposure has been captured yet. If we keep the script as it is now, this will lead to a result of "T" for RFXSTDTC.
So we better do a check whether data is available, and if not, put an empty value.
This can e.g. be done by:



```
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='
$FIRSTDATE = $DATE[1];
# Mapping using ODM element ItemData with ItemOID IT.EXPTIM
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='
$FIRSTTIME = $TIME[1];
if($FIRSTDATE != '' and $FIRSTTIME != '') {
     $DM.RFXSTDTC = createdatetime($FIRSTDATE,$FIRSTTIME);
} else {
     print 'no first exposure date found yet for subject ' $USUBJID;
     $DM.RFXSTDTC = '';
}
```

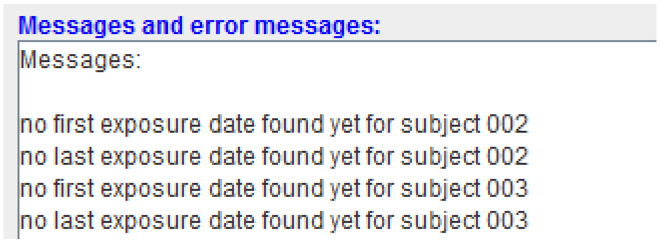
Remark the "print" statement which will generate a message during execution, and the usage of $USUBJID. The latter is possible as $USUBJID has been defined before $DM.RFXSTDTC.

The result is then e.g.:



STUDYID	DOMAIN	USUBJID	DM.RFXSTDTC	DM.RFXENDTC
MyStudy	DM	001	2006-05-01T12:57:04	2006-05-02T12:03:04
MyStudy	DM	002		
MyStudy	DM	003		

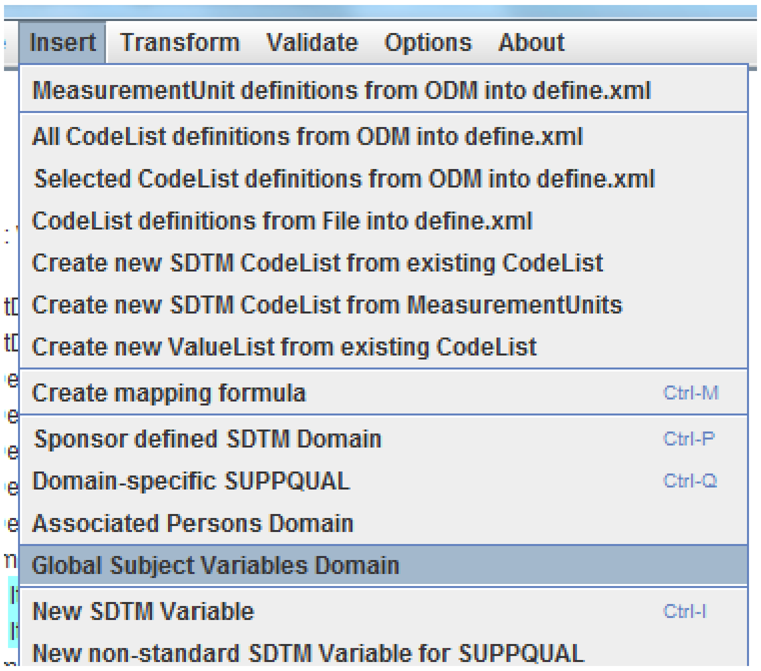
We see that for subjects 002 and 003, no first nor last exposure date/time have been found yet. In the messages box we indeed find:



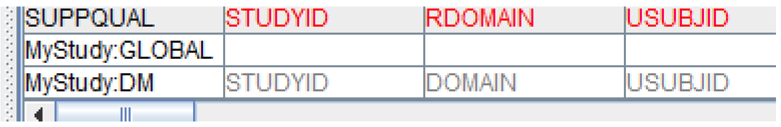
```
Messages:

no first exposure date found yet for subject 002
no last exposure date found yet for subject 002
no first exposure date found yet for subject 003
no last exposure date found yet for subject 003
```

Making RFXSTDTC a "global" variable

As we will need the values of RFXSTDC for deriving baseline flags, we need to make it a "global" variable[1]. For this, use the menu "Insert - Global Subject Variables Domain"


Insert
Transform
Validate
Options
About
MeasurementUnit definitions from ODM into define.xml
All CodeList definitions from ODM into define.xml
Selected CodeList definitions from ODM into define.xml
CodeList definitions from File into define.xml
Create new SDTM CodeList from existing CodeList
Create new SDTM CodeList from MeasurementUnits
Create new ValueList from existing CodeList
Create mapping formula Ctrl-M
Sponsor defined SDTM Domain Ctrl-P
Domain-specific SUPPQUAL Ctrl-Q
Associated Persons Domain
Global Subject Variables Domain
New SDTM Variable Ctrl-I
New non-standard SDTM Variable for SUPPQUAL

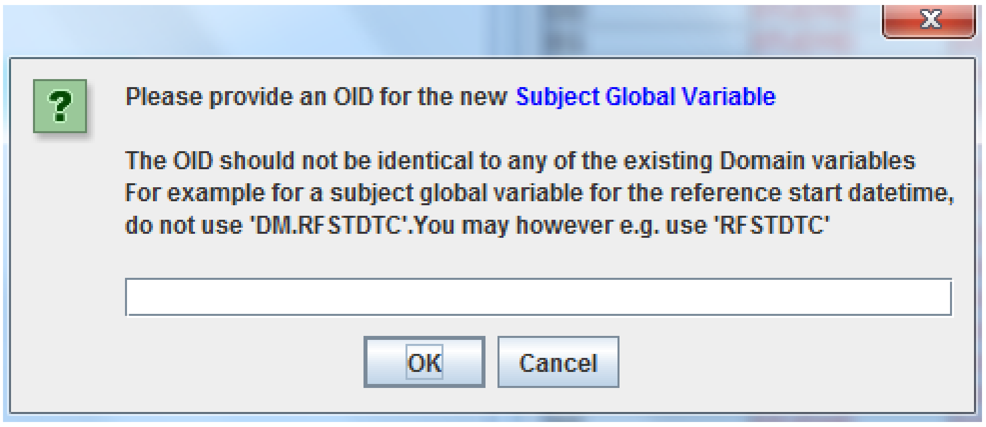
leading to:



SUPPQUAL	STUDYID	RDOMAIN	USUBJID
MyStudy:GLOBAL			
MyStudy:DM	STUDYID	DOMAIN	USUBJID

A "special" dataset has been added, allowing to define variables and mappings that can be reused in any other domain mapping. However, when executing the mappings, no "GLOBAL" dataset will be seen in the output, it is just for temporary usage.

We can now generate variables in "MyStudy:GLOBAL" like for any other domain, using the menu "Insert - New SDTM Variable", leading to:



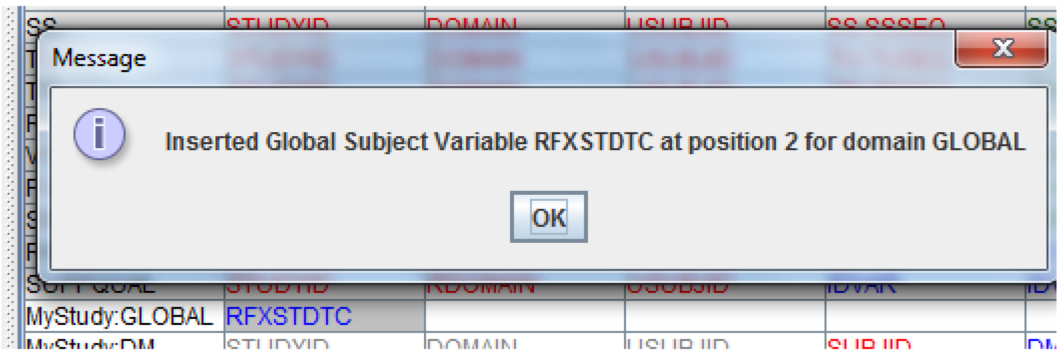
Please provide an OID for the new Subject Global Variable

The OID should not be identical to any of the existing Domain variables
For example for a subject global variable for the reference start datetime,
do not use 'DM.RFSTDTC'.You may however e.g. use 'RFSTDTC'

We give the new variable the name "RFXSTDTC". Do not name it "DM.RFXSTDTC" as the latter

[1]The reason for this is that SDTM-ETL cannot "look" in datasets that have already been created and populated, it can only look in the source data.

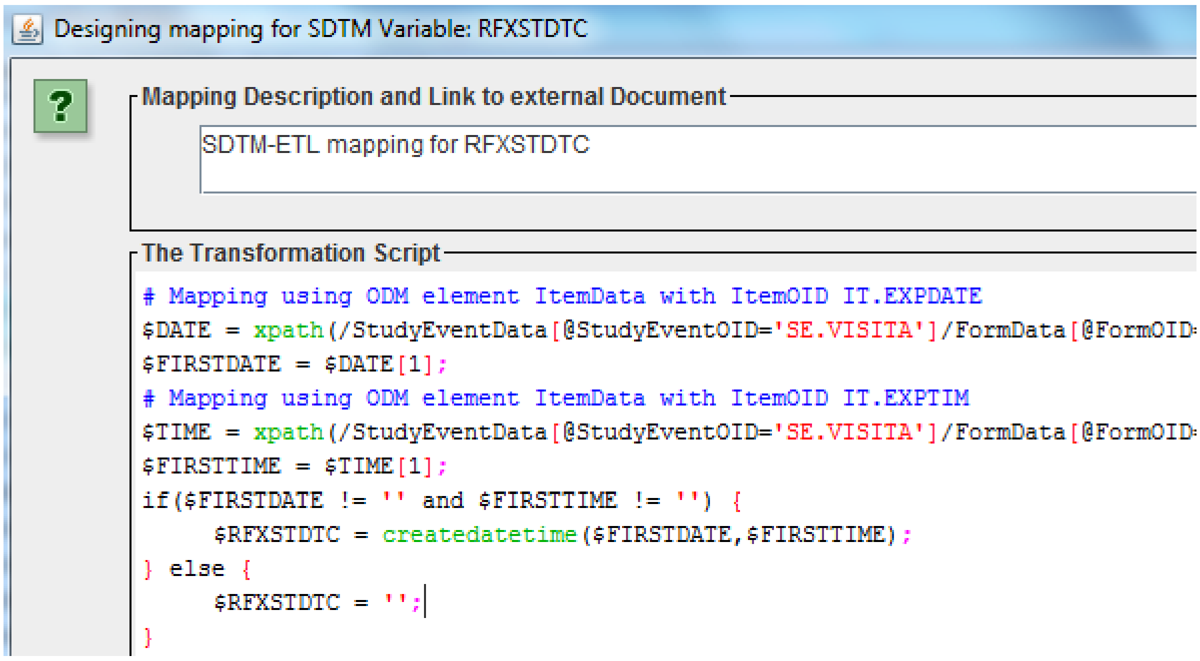
is only for DM, whereas our is global.
After filling "RFXSTDTC" in the field and clicking "OK" we get:



Inserted Global Subject Variable RFXSTDTC at position 2 for domain GLOBAL

SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR
MyStudy:GLOBAL	RFXSTDTC			

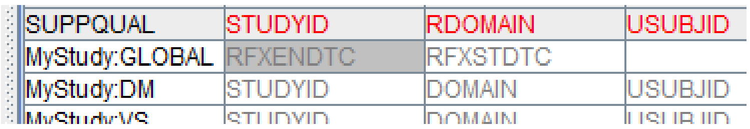
We can now just add a mapping for RFXSTDTC just like for any other variable. As we do already have a very similar mapping for DM.RFXSTDTC we can copy that one, and adapt it for the different variable name RFXSTDTC, i.e.:


Designing mapping for SDTM Variable: RFXSTDTC
Mapping Description and Link to external Document
The Transformation Script

SDTM-ETL mapping for RFXSTDTC

```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$DATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID
$FIRSTDATE = $DATE[1];
# Mapping using ODM element ItemData with ItemOID IT.EXPTIM
$TIME = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID
$FIRSTTIME = $TIME[1];
if($FIRSTDATE != '' and $FIRSTTIME != '') {
     $RFXSTDTC = createdatetime($FIRSTDATE,$FIRSTTIME);
} else {
     $RFXSTDTC = '';
}
```

and then repeat that for the global variable RFXENDTC.

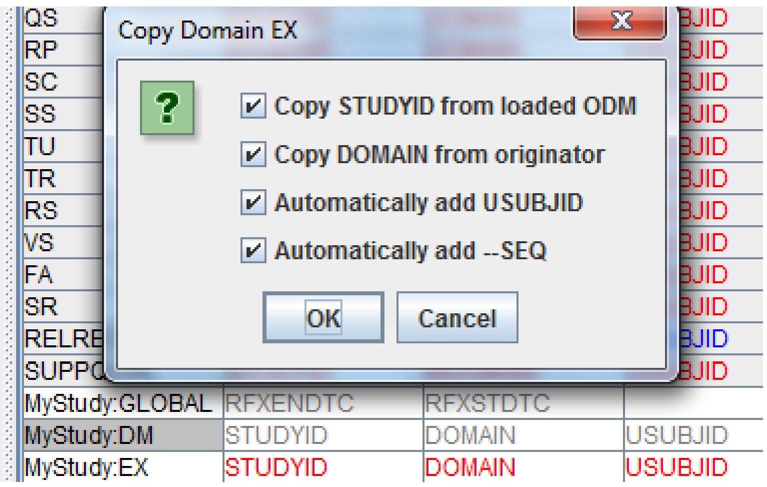


SUPPQUAL	STUDYID	RDOMAIN	USUBJID
MyStudy:GLOBAL	RFXENDTC	RFXSTDTC	
MyStudy:DM	STUDYID	DOMAIN	USUBJID
MyStudy:VS	STUDYID	DOMAIN	USUBJID

It doesn't matter here whether RFXENDTC comes before or after RFXSTDTC.

Generating the EX dataset

In order to generate the EX dataset, drag-and-drop the EX row from the template to after the "MyStudy:DM" row. This leads to:



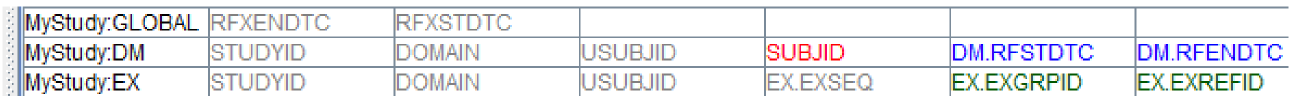
Copy Domain EX

- [✓] Copy STUDYID from loaded ODM
- [✓] Copy DOMAIN from originator
- [✓] Automatically add USUBJID
- [✓] Automatically add --SEQ

OK Cancel

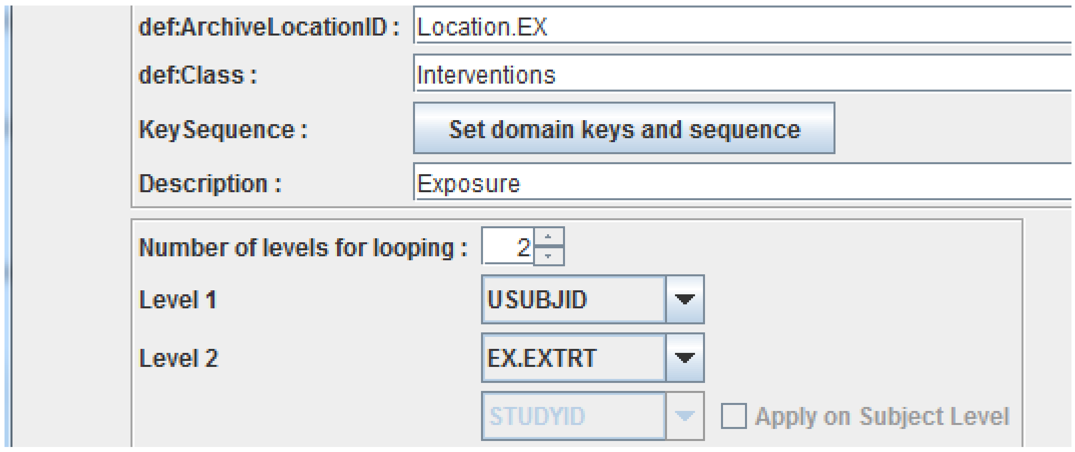
QS			BJID
RP			BJID
SC			BJID
SS			BJID
TU			BJID
TR			BJID
RS			BJID
VS			BJID
FA			BJID
SR			BJID
RELRE			BJID
SUPPQ			BJID
MyStudy:GLOBAL	RFXENDTC	RFXSTDTC	
MyStudy:DM	STUDYID	DOMAIN	USUBJID
MyStudy:EX	STUDYID	DOMAIN	USUBJID

asking us whether the mappings for STUDYD, DOMAIN, USUBJID and EXSEQ can already be automatically generated. As these are obvious, we accept. We then see that these cells in the table are already "grayed", meaning that a mapping is available:



MyStudy:GLOBAL	RFXENDTC	RFXSTDTC				
MyStudy:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC
MyStudy:EX	STUDYID	DOMAIN	USUBJID	EX.EXSEQ	EX.EXGRPID	EX.EXREFID

The most important variables in EX are "EXTRT" (the topic variable), EXDOSE (the numeric dose), and of course EXSTDTC (The date/time when administration of the treatment indicated by EXTRT and EXDOSE began) and EXENDTC (The date/time when administration of the treatment indicated by EXTRT and EXDOSE ended). The usual structure is "*One record per protocol-specified study treatment, constant-dosing interval, per subject*".
In the case we have only one constant dosing interval and one treatment (as in our study), this essentially reduces to "One record per subject". We need to indicate that in our domain structure by double-clicking the cell "MyStudy:EX"[2]:

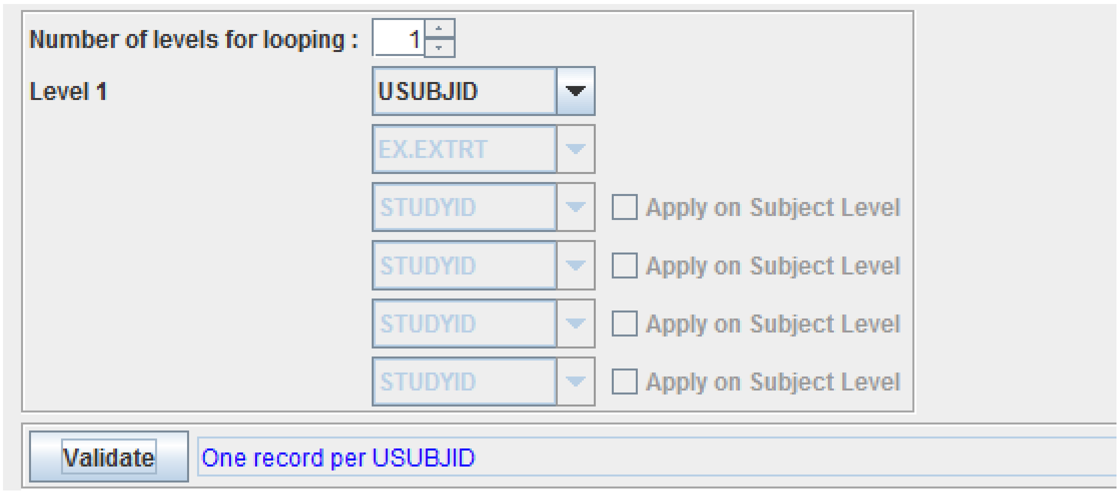


def:ArchiveLocationID :	Location.EX
def:Class :	Interventions
KeySequence :	Set domain keys and sequence
Description :	Exposure
Number of levels for looping :	2
Level 1	USUBJID
Level 2	EX.EXTRT
	STUDYID ☐ Apply on Subject Level

We can decrease the number of iteration levels to 1, which leaves us with iterating only over the

[2]Or by using the menu "Edit - SDTM Domain Properties"

subjects (i.e. one record per subject)[3]:



Number of levels for looping : 1

Level 1: USUBJID

EX.EXTRT

STUDYID ☐ Apply on Subject Level

STUDYID ☐ Apply on Subject Level

STUDYID ☐ Apply on Subject Level

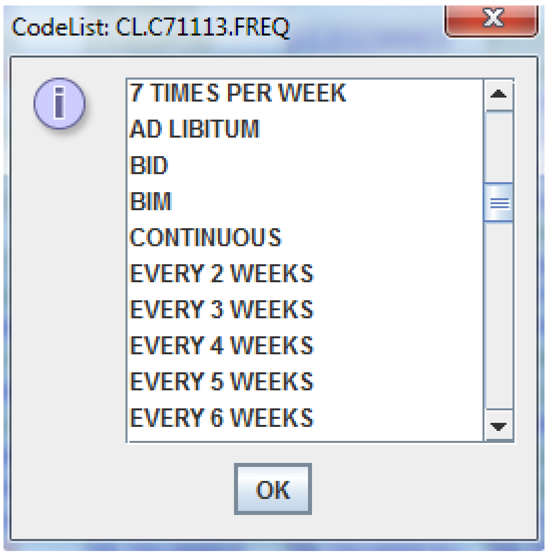
STUDYID ☐ Apply on Subject Level

One record per USUBJID

Click "OK" to confirm.

We can now start adding the mappings. EXTRT and EXDOSE (and EXDOSEU) or EXDOSETXT are obvious as they come from the protocol and are fix (text) values. You can fill in the values simply e.g. as $EX.EXTRT = 'My Study Drug;'.

For EX.EXFREQ, there is an associated codelist, and we can ask for the values using the menu "View - SDTM associated CodeList[4]":



CodeList: CL.C71113.FREQ

7 TIMES PER WEEK
AD LIBITUM
BID
BIM
CONTINUOUS
EVERY 2 WEEKS
EVERY 3 WEEKS
EVERY 4 WEEKS
EVERY 5 WEEKS
EVERY 6 WEEKS

In our case, the protocol stated "daily" so we use "QD"[5]
leading to the mapping $EX.EXDOSFRQ = 'QD';

Let us now concentrate concentrate on EXSTDC (first exposure date/time) and EXENDTC (last exposure date/time). We will do the exercise for 2 use cases:

[3]In case of multiple treatment and dosing intervals, we will probably need the looping variables USUBJID, EX.EXTRT and EX.EXSTDTC, in that order.

[4]In this case, we used CDISC SDTM controlled terminology 2016-06-24, codelist CL.71113.FREQ.

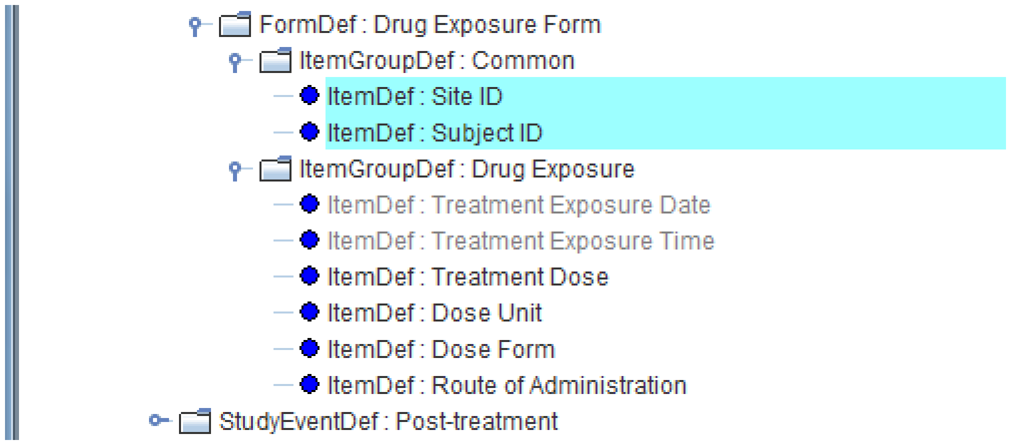
[5]Unfortunately, the CDISC controlled terminology comes without explanation of what "QD" means, so mappers do either need medical knowledge, or need to look up what these terms mean.

- only the exposure dates were captured
- both exposure dates and times were captured

Of course, in some cases the database already contains first and last date(time) of exposure, but we want to do the exercise where all the exposure dates or datetimes are known[6].

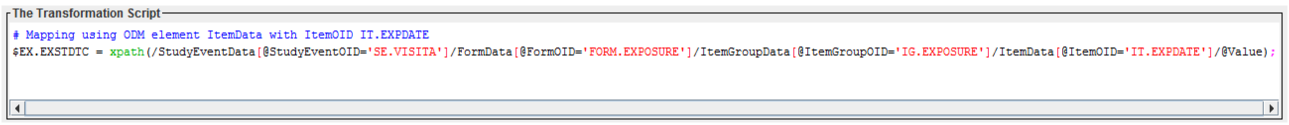
The first case (only exposure dates captured) is very common, but may lead to problems with assigning baseline flags. For example, suppose the first exposure date is 2006-05-02 and vital signs have been captured that same day, once a 09:02:55 and once at 23:12:55.
Was the last vital signs test before or after first exposure? No way to find out from the data themselves. And if no times were given for the vitals signs, but only the date, the problem even becomes more difficult. We will demonstrate this in this tutorial.

For the first case, we will just use the dates. For EX.EXSTDC we look into our ODM tree:



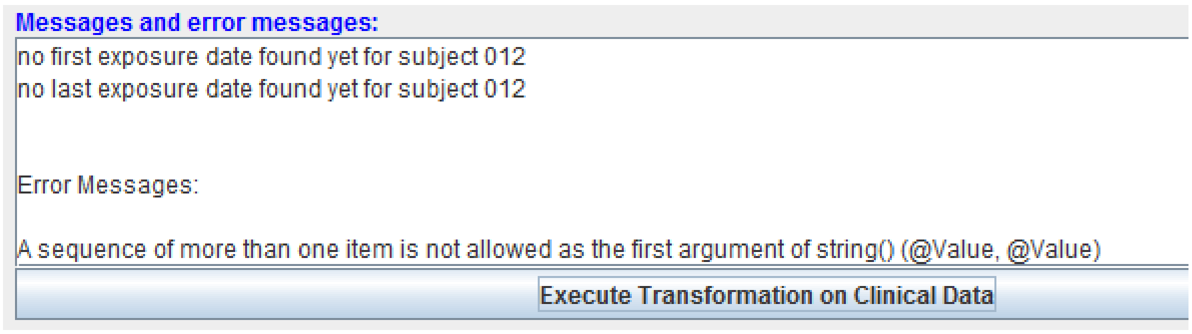
FormDef : Drug Exposure Form
ItemGroupDef : Common
ItemDef : Site ID
ItemDef : Subject ID
ItemGroupDef : Drug Exposure
ItemDef : Treatment Exposure Date
ItemDef : Treatment Exposure Time
ItemDef : Treatment Dose
ItemDef : Dose Unit
ItemDef : Dose Form
ItemDef : Route of Administration
StudyEventDef : Post-treatment

and just drag-and-drop "Treatment Exposure Date" to EX.EXSDTC, leading to:



```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$EX.EXSTDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.EXPOSURE']/ItemGroupData[@ItemGroupOID='IG.EXPOSURE']/ItemData[@ItemOID='IT.EXPDATE']/@Value);
```

Executing the mappings then results in an error:


Messages and error messages:

```
no first exposure date found yet for subject 012
no last exposure date found yet for subject 012


Error Messages:

A sequence of more than one item is not allowed as the first argument of string() (@Value, @Value)
```

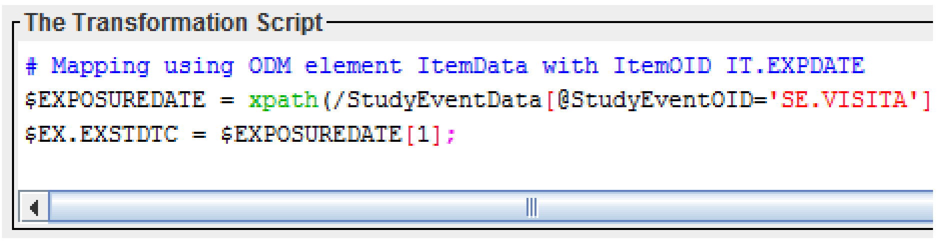
Execute Transformation on Clinical Data

as there is more than 1 exposure date, i.e. the value of EX.EXSTDTC is an array instead of a single value. We can solve this in 2 ways.

[6]Remark that this may mean that also an EC dataset (Exposure as collected) needs to be populated. This is especially important when treatments were missed or deviations from the constant dosing interval have been collected.

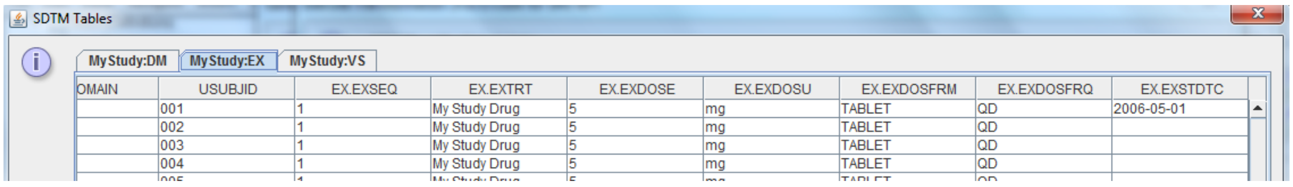
The first is that we trust that the exposure dates in the ODM file are in chronological order. This is a requirement of the ODM standard: "*Section 2.10: The elements in the data portion of an ODM file that apply to a single entity must occur in temporal order*".

In that case we can simply adapt the script to:



```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$EXPOSUREDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']
$EX.EXSTDTC = $EXPOSUREDATE[1];
```

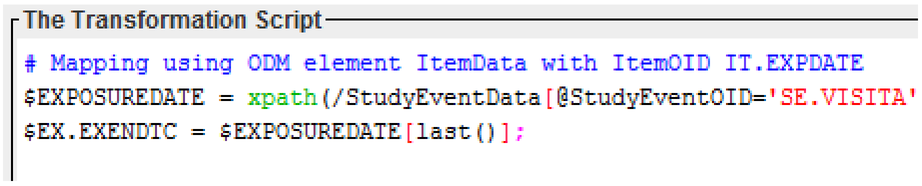
leading to:



OMAIN	USUBJID	EX.EXSEQ	EX.EXTRT	EX.EXDOSE	EX.EXDOSU	EX.EXDOSFRM	EX.EXDOSFRQ	EX.EXSTDTC
	001	1	My Study Drug	5	mg	TABLET	QD	2006-05-01
	002	1	My Study Drug	5	mg	TABLET	QD	
	003	1	My Study Drug	5	mg	TABLET	QD	
	004	1	My Study Drug	5	mg	TABLET	QD	

The "[1]" indicating the first value in the array of values.

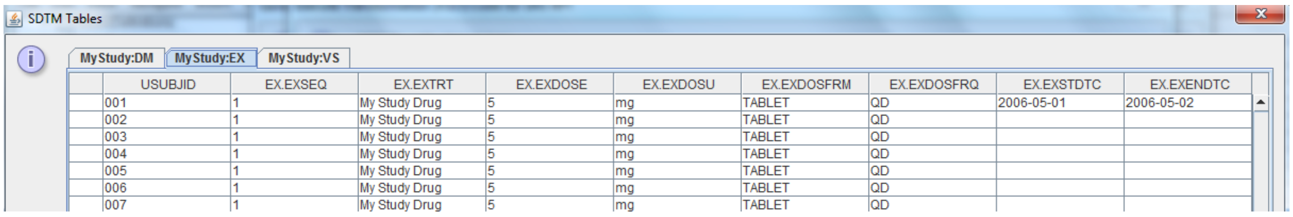
Similarly for EX.EXENDTC:



```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$EXPOSUREDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'
$EX.EXENDTC = $EXPOSUREDATE[last()];
```

with the predicate [last()] meaning: take the last value from the array.

leading to:



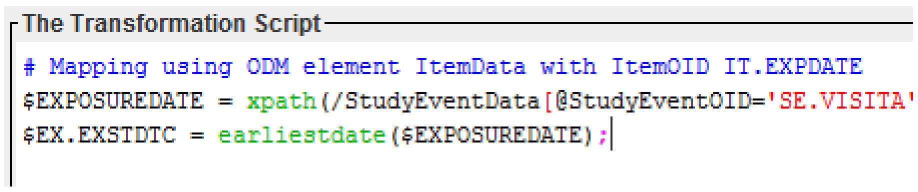
	USUBJID	EX.EXSEQ	EX.EXTRT	EX.EXDOSE	EX.EXDOSU	EX.EXDOSFRM	EX.EXDOSFRQ	EX.EXSTDTC	EX.EXENDTC
	001	1	My Study Drug	5	mg	TABLET	QD	2006-05-01	2006-05-02
	002	1	My Study Drug	5	mg	TABLET	QD		
	003	1	My Study Drug	5	mg	TABLET	QD		
	004	1	My Study Drug	5	mg	TABLET	QD		
	005	1	My Study Drug	5	mg	TABLET	QD		
	006	1	My Study Drug	5	mg	TABLET	QD		
	007	1	My Study Drug	5	mg	TABLET	QD		

As of version 3.1 of the software, we also have new functions for selecting the earliest and latest date in a series of dates. These are:

– earliestdate(...)
– latestdate(...)
– earliestdatetime(...)
– latestdatetime(...)

each taking an array of dates or datetimes respectively.

They do select the earliest or latest date or datetime in a series (array) of date or datetimes, independent of the order in which these appear.
So for example, for $EX.EXSTDTC we can use:



```
# Mapping using ODM element ItemData with ItemOID IT.EXPDATE
$EXPOSUREDATE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA'
$EX.EXSTDTC = earliestdate($EXPOSUREDATE);
```

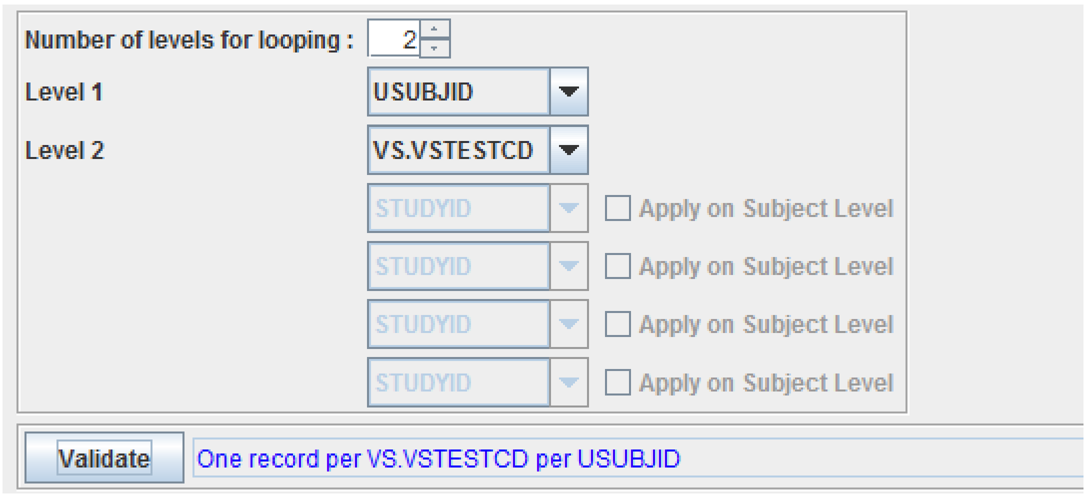
leading to exactly the same result as before, but now ensuring we really have the earliest date.

Later, we will see that some other useful new functions have been defined, like "latest date before or equal to a reference date", which will help us to generate baseline flags.

Creating the Vital Signs (VS) domain

We can start creating mappings for the VS domain by drag-and-drop from the template, this time to after the "MyStudy:EX" row.

Looking at the structure by double-clicking "MyStudy:VS" we find:

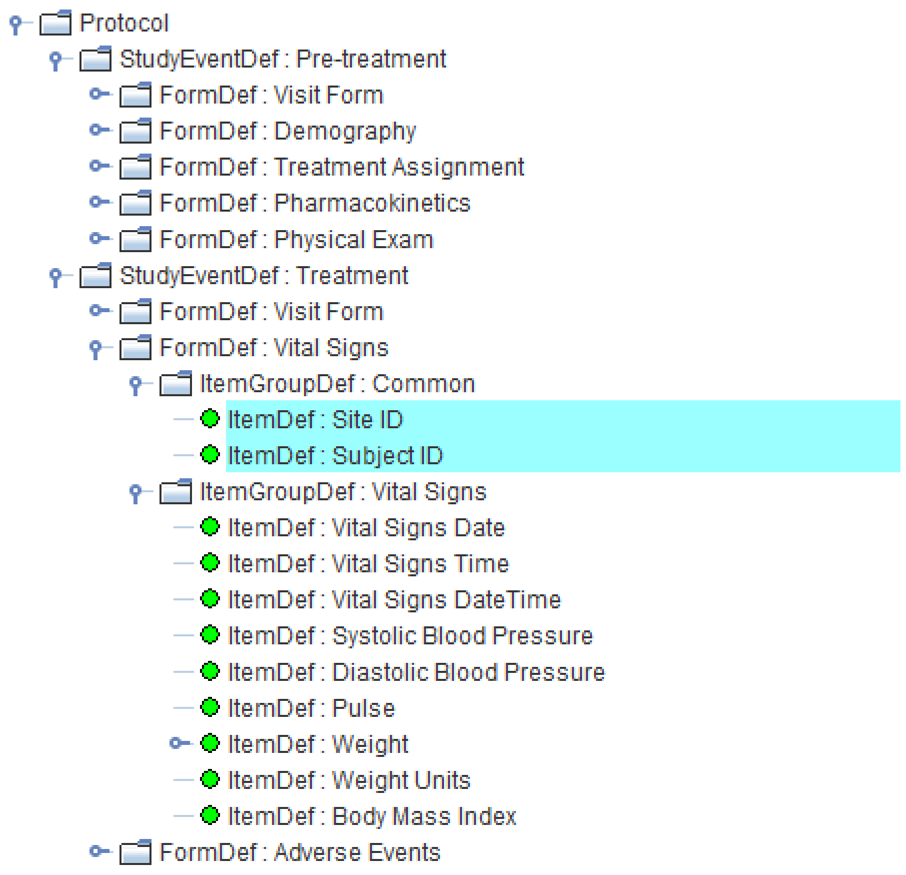


Number of levels for looping : 2
Level 1 USUBJID
Level 2 VS.VSTESTCD
STUDYID
STUDYID
STUDYID
STUDYID

One record per VS.VSTESTCD per USUBJID

This is sufficient for now. Later we might need to add an additional iteration level.

Looking at our ODM, we find:



```
Protocol
    StudyEventDef : Pre-treatment
        FormDef : Visit Form
        FormDef : Demography
        FormDef : Treatment Assignment
        FormDef : Pharmacokinetics
        FormDef : Physical Exam
    StudyEventDef : Treatment
        FormDef : Visit Form
        FormDef : Vital Signs
            ItemGroupDef : Common
                ItemDef : Site ID
                ItemDef : Subject ID
            ItemGroupDef : Vital Signs
                ItemDef : Vital Signs Date
                ItemDef : Vital Signs Time
                ItemDef : Vital Signs DateTime
                ItemDef : Systolic Blood Pressure
                ItemDef : Diastolic Blood Pressure
                ItemDef : Pulse
                ItemDef : Weight
                ItemDef : Weight Units
                ItemDef : Body Mass Index
        FormDef : Adverse Events
```

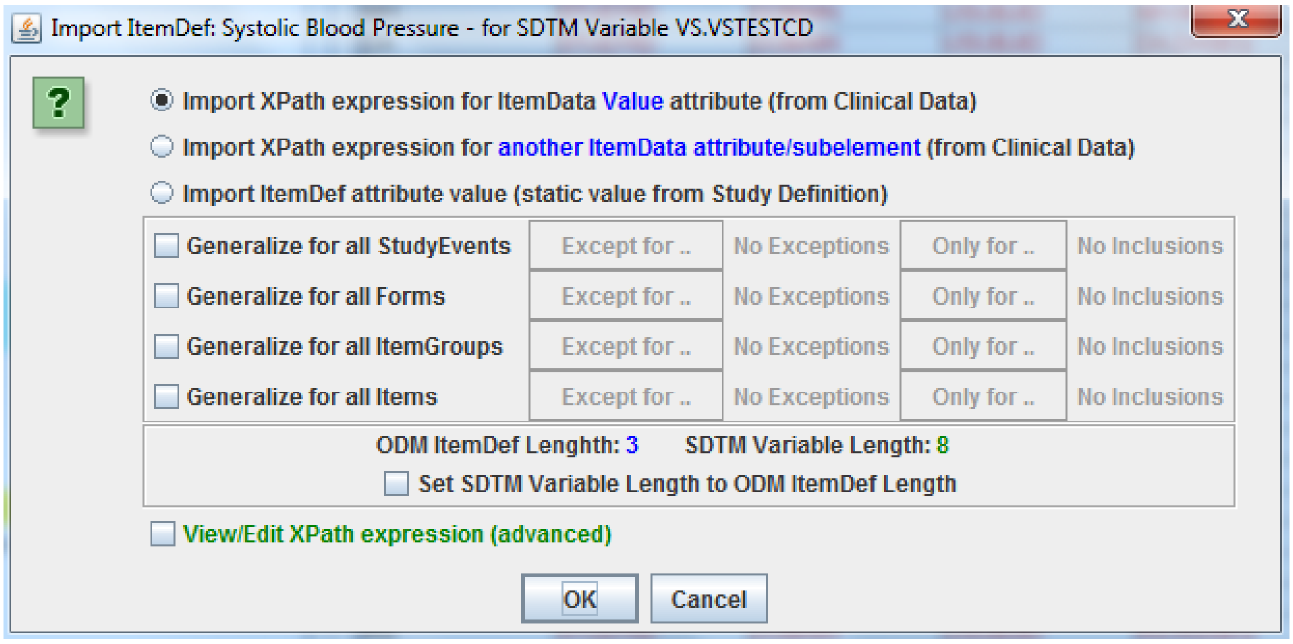
We see that no vital signs are taken during the "Pre-treatment" visit (at least not using the "Vital Signs Form"), but we also know that the "Treatment" visit is repeating. We also see however, when clicking "ItemGroupDef: Vital Signs", that this group is repeating, i.e. we can have repeats for all our vital signs within one and the same "Vital Signs Form".

We also see things like "Systolic Blood Pressure" and "Pulse" which are clearly vitals signs data points, but in the same group we also find "Vital Signs Date" which will probably later come in VSDTC.

First we will start doing the mapping for VS.VSTESTCD, as this is a "looping variable"[7].

So we need to take a bit of attention when using "drag and drop". Let us for example "drag and drop" "Systolic Blood Pressure" to VS.VSTESTCD:

[7]It is adviced to always first perform the mappings for "looping variables" as this makes testing much easier.


Import ItemDef: Systolic Blood Pressure - for SDTM Variable VS.VSTESTCD

(●) Import XPath expression for ItemData Value attribute (from Clinical Data)
( ) Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
( ) Import ItemDef attribute value (static value from Study Definition)

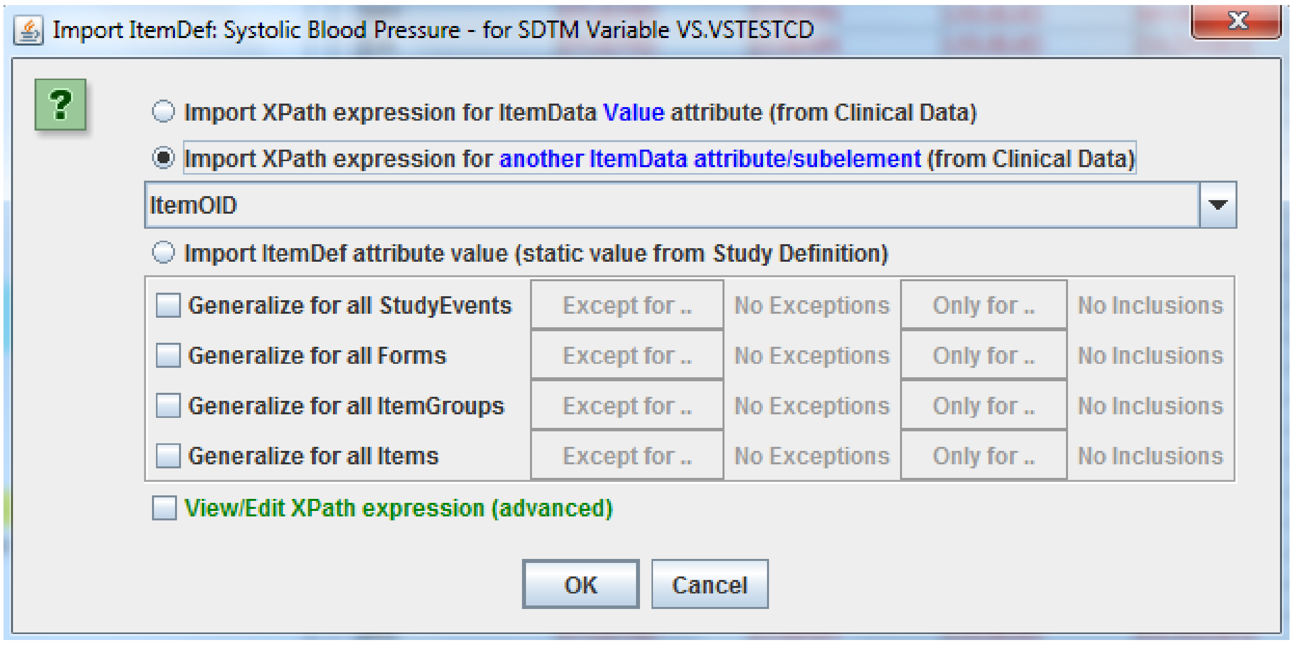
☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Items	Except for ..	No Exceptions	Only for ..	No Inclusions

ODM ItemDef Lenghth: 3 SDTM Variable Length: 8
☐ Set SDTM Variable Length to ODM ItemDef Length

☐ View/Edit XPath expression (advanced)

OK Cancel

We do not want the value from the collected data, but we want to get the identifier of the test, and map that to CDISC controlled terminology for VSTESTCD. So we check the radiobutton "Import XPath expression for another ItemData attribute/subelement):


Import ItemDef: Systolic Blood Pressure - for SDTM Variable VS.VSTESTCD

( ) Import XPath expression for ItemData Value attribute (from Clinical Data)
(●) Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
ItemOID
( ) Import ItemDef attribute value (static value from Study Definition)

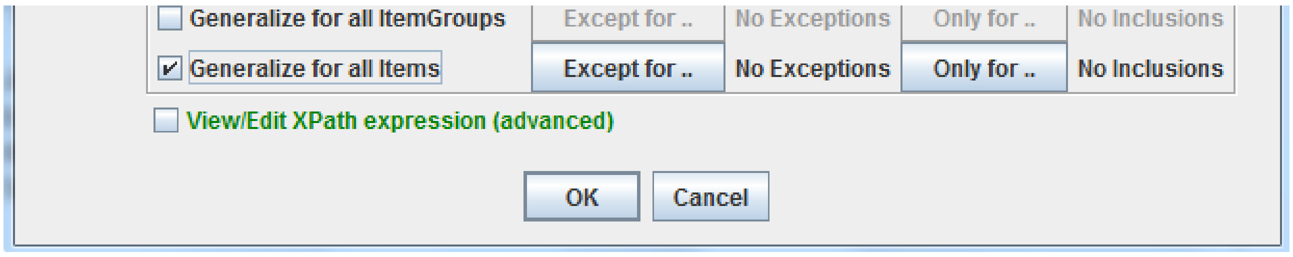
☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Items	Except for ..	No Exceptions	Only for ..	No Inclusions

☐ View/Edit XPath expression (advanced)

OK Cancel

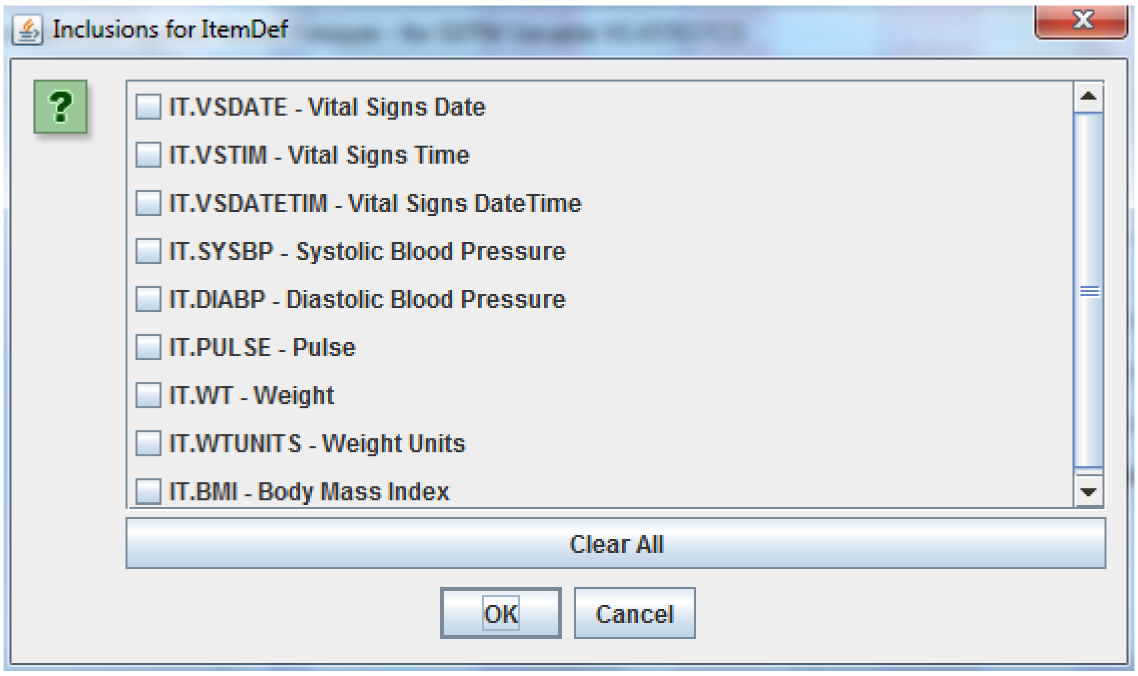
"ItemOID" seems to be a good choice, as it is the identifier for the data point

As vital signs have only been collected during treatment visits, we do not need to check "Generalize for all StudyEvents" (in case of doubt, one can check it, this does not harm). We do however have that different items contain the test name and value, but with exceptions.
So we check "Generalize for all Items":


- [ ] Generalize for all ItemGroups — Except for .. | No Exceptions | Only for .. | No Inclusions
- [✓] Generalize for all Items — Except for .. | No Exceptions | Only for .. | No Inclusions
- [ ] View/Edit XPath expression (advanced)

OK | Cancel

and then use "Except for …" or "Only for …" to select which ones we do really want. Using "Only for …" this leads to:

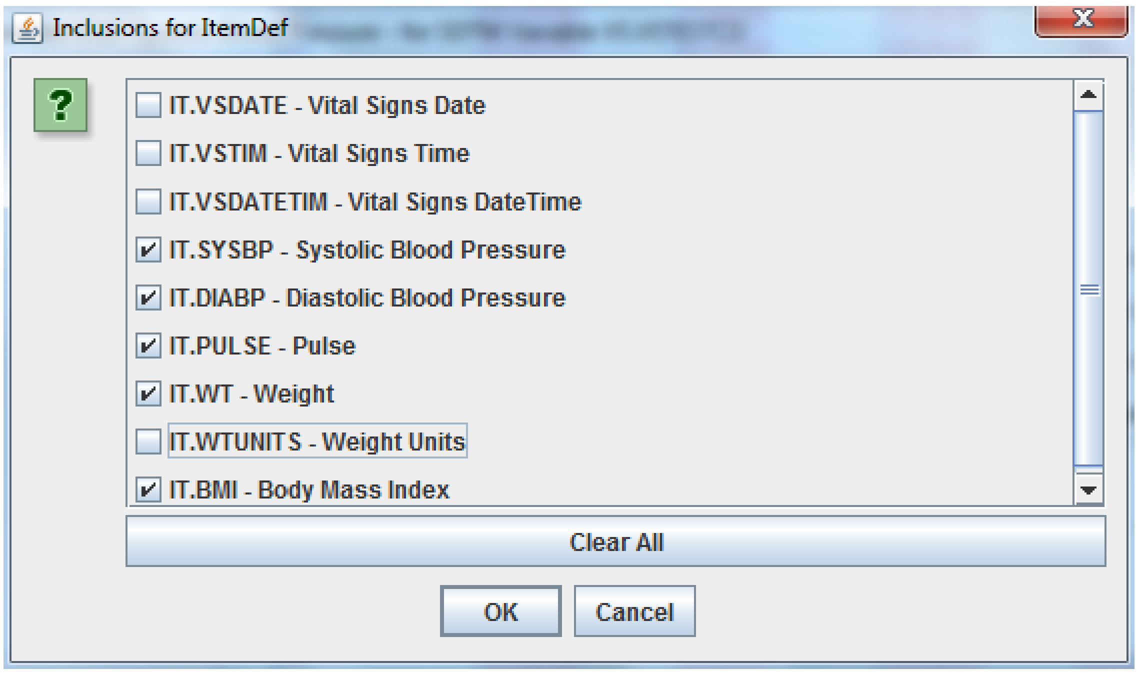

Inclusions for ItemDef

- [ ] IT.VSDATE - Vital Signs Date
- [ ] IT.VSTIM - Vital Signs Time
- [ ] IT.VSDATETIM - Vital Signs DateTime
- [ ] IT.SYSBP - Systolic Blood Pressure
- [ ] IT.DIABP - Diastolic Blood Pressure
- [ ] IT.PULSE - Pulse
- [ ] IT.WT - Weight
- [ ] IT.WTUNITS - Weight Units
- [ ] IT.BMI - Body Mass Index

Clear All

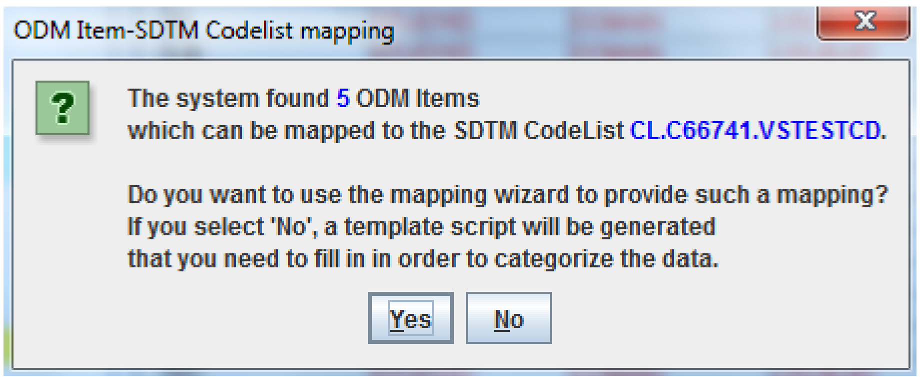
OK | Cancel

We do not want to have "IT.VSDATE", "IT.VSTIM" and "IT.VSDATETIM" as test codes, and also not "IT.WTUNITS", but we want to generate test codes for all the others ("IT.SYSBP" to "IT.BMI"). So we leave the first three and the second last unchecked , and check all other ones:


Inclusions for ItemDef

- [ ] IT.VSDATE - Vital Signs Date
- [ ] IT.VSTIM - Vital Signs Time
- [ ] IT.VSDATETIM - Vital Signs DateTime
- [x] IT.SYSBP - Systolic Blood Pressure
- [x] IT.DIABP - Diastolic Blood Pressure
- [x] IT.PULSE - Pulse
- [x] IT.WT - Weight
- [ ] IT.WTUNITS - Weight Units
- [x] IT.BMI - Body Mass Index

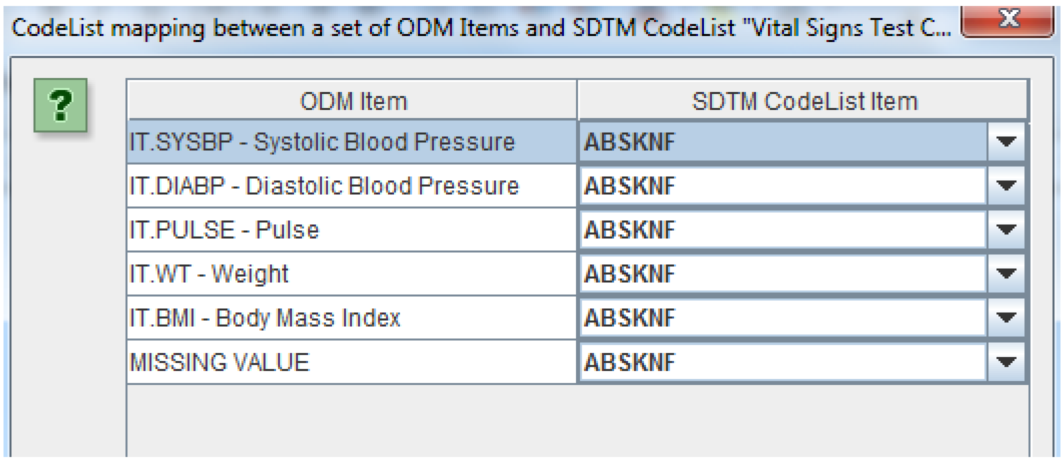
Then clicking "OK" twice leads to:


ODM Item-SDTM Codelist mapping

The system found 5 ODM Items
which can be mapped to the SDTM CodeList CL.C66741.VSTESTCD.

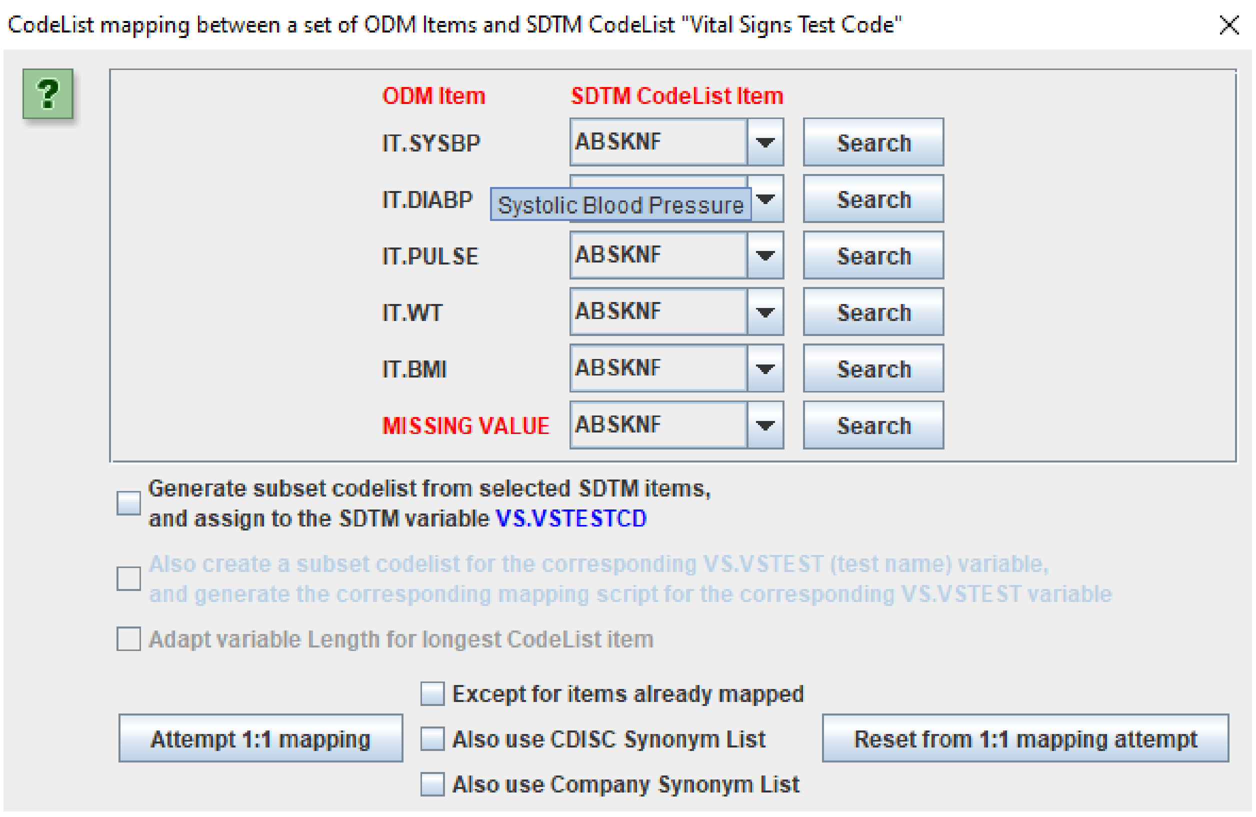
Do you want to use the mapping wizard to provide such a mapping?
If you select 'No', a template script will be generated
that you need to fill in in order to categorize the data.

This will retrieve the identifiers (ItemOIDs), but of course, these are not the test codes themselves. The system knows this and asks us whether we want to use the mapping wizard for generating the 1:1 mappings. When clicking "Yes", the wizard is displayed:


CodeList mapping between a set of ODM Items and SDTM CodeList "Vital Signs Test C...

ODM Item	SDTM CodeList Item
IT.SYSBP - Systolic Blood Pressure	ABSKNF
IT.DIABP - Diastolic Blood Pressure	ABSKNF
IT.PULSE - Pulse	ABSKNF
IT.WT - Weight	ABSKNF
IT.BMI - Body Mass Index	ABSKNF
MISSING VALUE	ABSKNF

with dropdowns on the right, which make it easy to do the mapping, e.g.:


CodeList mapping between a set of ODM Items and SDTM CodeList "Vital Signs Test Code"

ODM Item	SDTM CodeList Item	
IT.SYSBP	ABSKNF	Search
IT.DIABP	Systolic Blood Pressure	Search
IT.PULSE	ABSKNF	Search
IT.WT	ABSKNF	Search
IT.BMI	ABSKNF	Search
MISSING VALUE	ABSKNF	Search

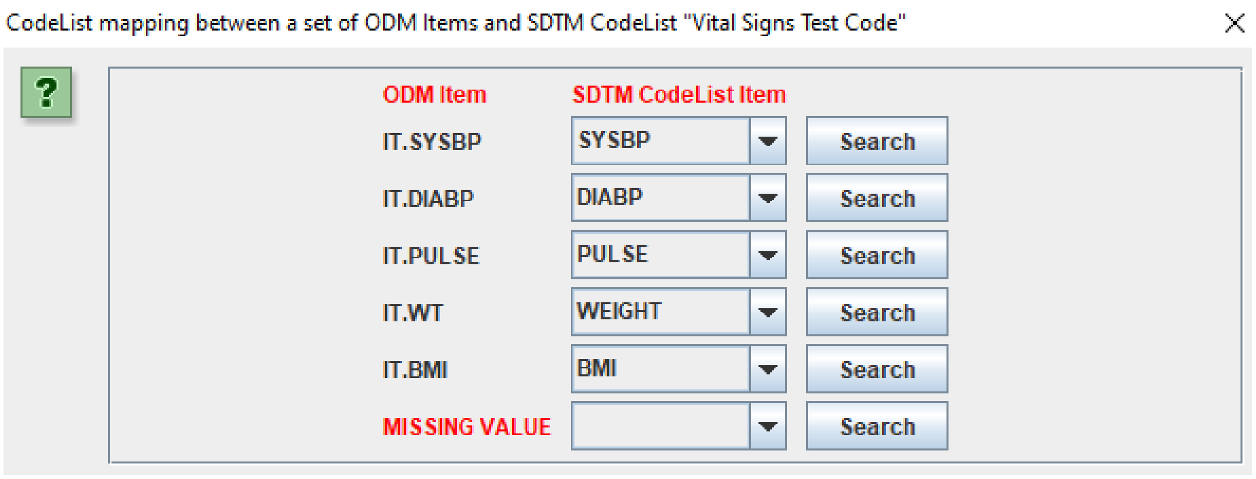
☐ Generate subset codelist from selected SDTM items, and assign to the SDTM variable VS.VSTESTCD

☐ Also create a subset codelist for the corresponding VS.VSTEST (test name) variable, and generate the corresponding mapping script for the corresponding VS.VSTEST variable

☐ Adapt variable Length for longest CodeList item

Attempt 1:1 mapping
☐ Except for items already mapped
☐ Also use CDISC Synonym List
☐ Also use Company Synonym List
Reset from 1:1 mapping attempt

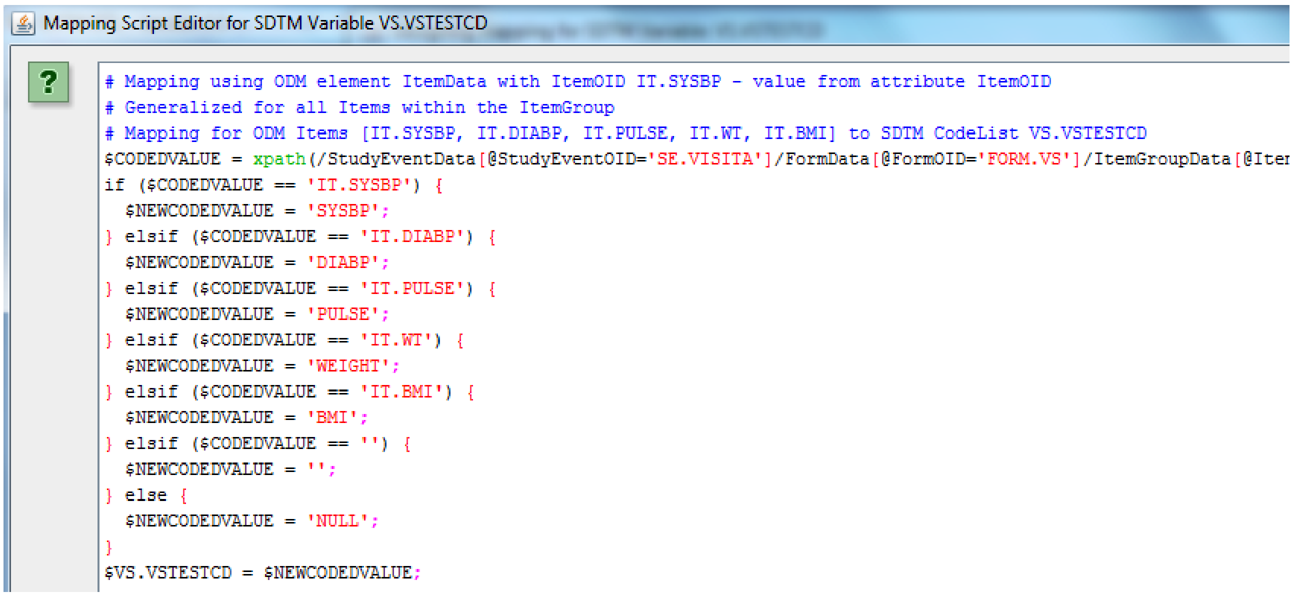
The correct mapping can be easily achieved[8]:


CodeList mapping between a set of ODM Items and SDTM CodeList "Vital Signs Test Code"

ODM Item	SDTM CodeList Item	
IT.SYSBP	SYSBP	Search
IT.DIABP	DIABP	Search
IT.PULSE	PULSE	Search
IT.WT	WEIGHT	Search
IT.BMI	BMI	Search
MISSING VALUE		Search

Clicking "OK" then leads to the mapping script:

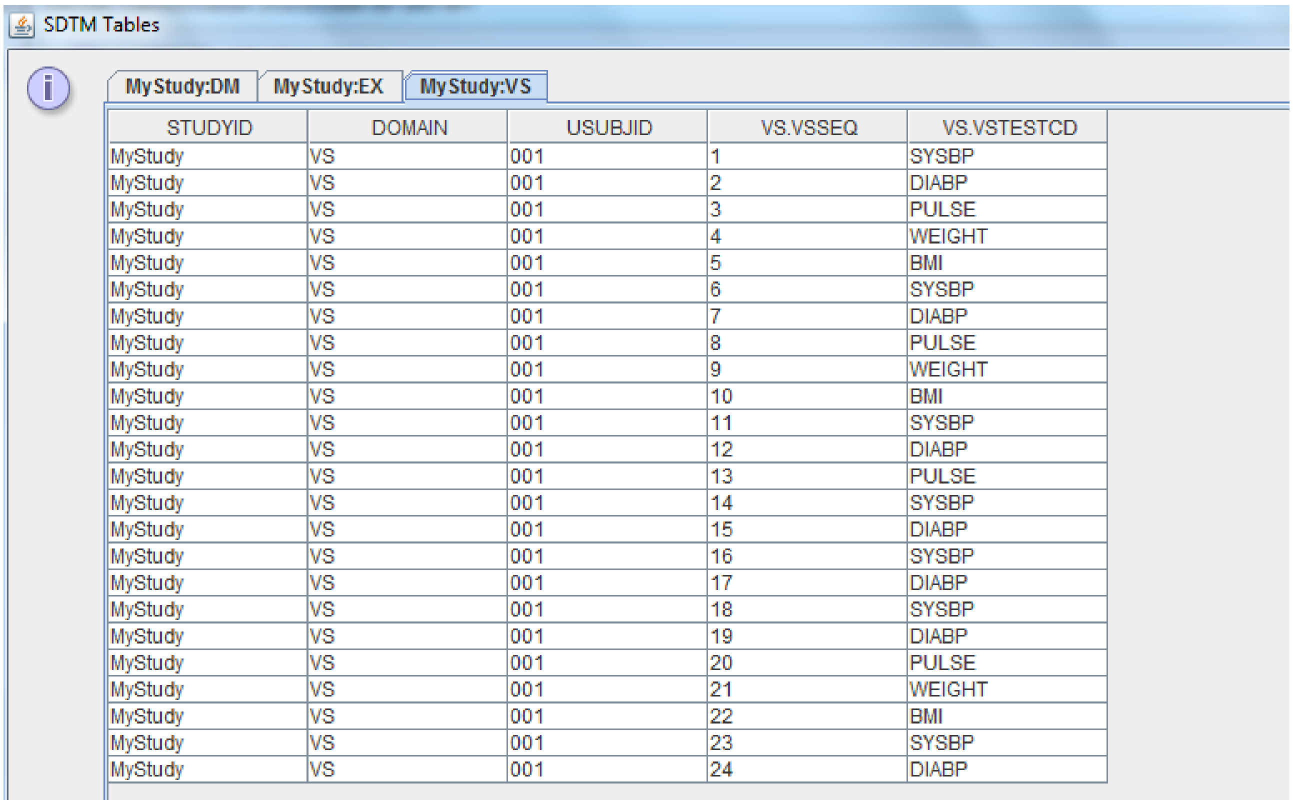
[8]When using the dropdown, typing the first character often already leads to the correct VSTESCD - no need to scroll down ... Also, in many cases, using the button "Attempt 1:1 mapping" will already do the job.



```
# 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
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Iter
if ($CODEDVALUE == 'IT.SYSBP') {
  $NEWCODEDVALUE = 'SYSBP';
} elsif ($CODEDVALUE == 'IT.DIABP') {
  $NEWCODEDVALUE = 'DIABP';
} elsif ($CODEDVALUE == 'IT.PULSE') {
  $NEWCODEDVALUE = 'PULSE';
} elsif ($CODEDVALUE == 'IT.WT') {
  $NEWCODEDVALUE = 'WEIGHT';
} elsif ($CODEDVALUE == 'IT.BMI') {
  $NEWCODEDVALUE = 'BMI';
} elsif ($CODEDVALUE == '') {
  $NEWCODEDVALUE = '';
} else {
  $NEWCODEDVALUE = 'NULL';
}
$VS.VSTESTCD = $NEWCODEDVALUE;
```

which one can still edit manually.
Remark the third-last line "$NEWCODEDVALUE = 'NULL'" which is for identifiers that were somehow missed. One can then of course still change the text 'NULL' in something else, like 'TO DO'.

Executing the mappings then gives:



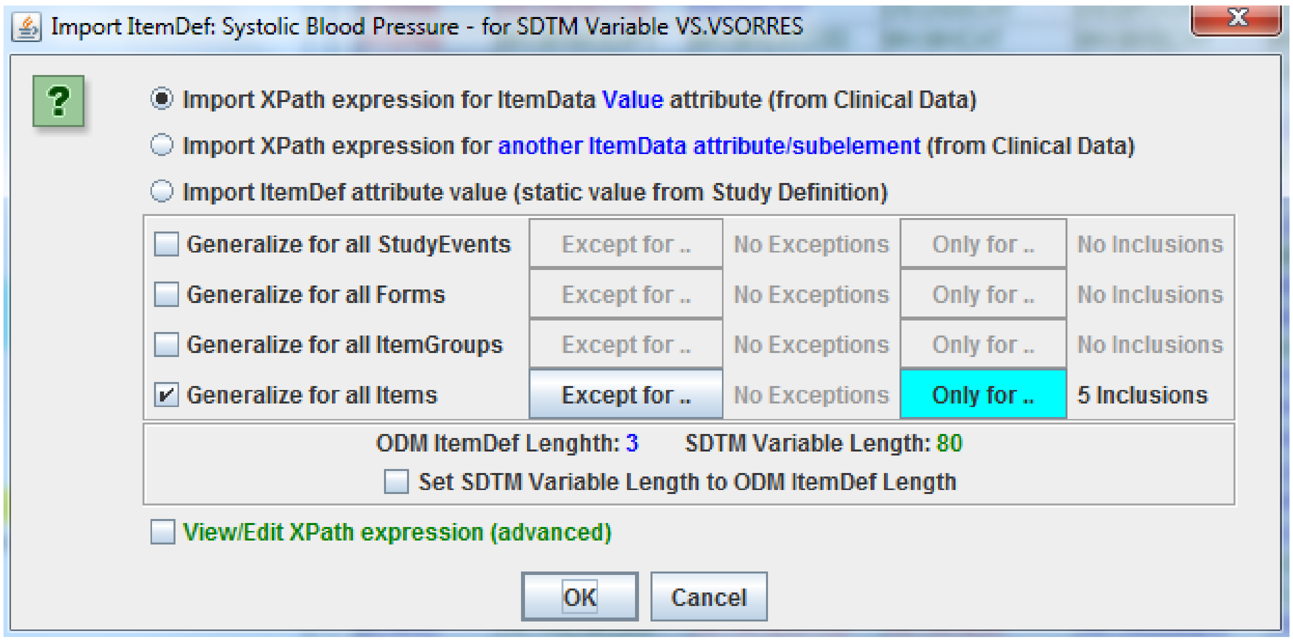
STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD
MyStudy	VS	001	1	SYSBP
MyStudy	VS	001	2	DIABP
MyStudy	VS	001	3	PULSE
MyStudy	VS	001	4	WEIGHT
MyStudy	VS	001	5	BMI
MyStudy	VS	001	6	SYSBP
MyStudy	VS	001	7	DIABP
MyStudy	VS	001	8	PULSE
MyStudy	VS	001	9	WEIGHT
MyStudy	VS	001	10	BMI
MyStudy	VS	001	11	SYSBP
MyStudy	VS	001	12	DIABP
MyStudy	VS	001	13	PULSE
MyStudy	VS	001	14	SYSBP
MyStudy	VS	001	15	DIABP
MyStudy	VS	001	16	SYSBP
MyStudy	VS	001	17	DIABP
MyStudy	VS	001	18	SYSBP
MyStudy	VS	001	19	DIABP
MyStudy	VS	001	20	PULSE
MyStudy	VS	001	21	WEIGHT
MyStudy	VS	001	22	BMI
MyStudy	VS	001	23	SYSBP
MyStudy	VS	001	24	DIABP

looking pretty good for the moment[9].

Let us now get some results "as captured" (VSORRES).

[9]Also observe that the VSSEQ values are assigned automatically, as we use the default mechanism to generate them automatically.

Just drag-and-drop from "Systolic Blood Pressure" again, but this time to VS.VSORRES, and keep the first choice "", as we now want the value of the data point, not the identifier:


Import ItemDef: Systolic Blood Pressure - for SDTM Variable VS.VSORRES

[● Import XPath expression for ItemData Value attribute (from Clinical Data)]
○ Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
○ Import ItemDef attribute value (static value from Study Definition)

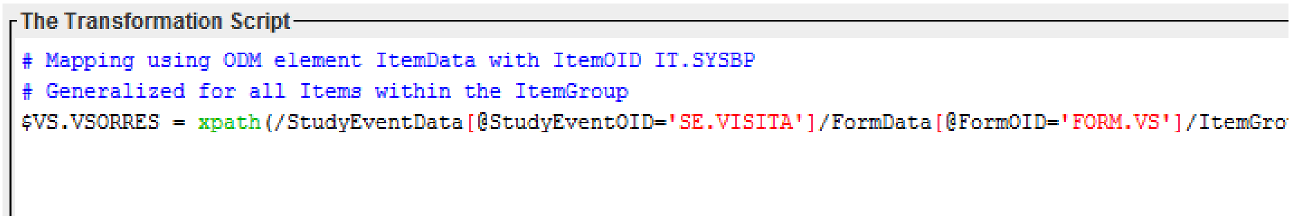
☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☑ Generalize for all Items	Except for ..	No Exceptions	Only for ..	5 Inclusions

ODM ItemDef Lenghth: 3 SDTM Variable Length: 80
☐ Set SDTM Variable Length to ODM ItemDef Length

☐ View/Edit XPath expression (advanced)

OK Cancel

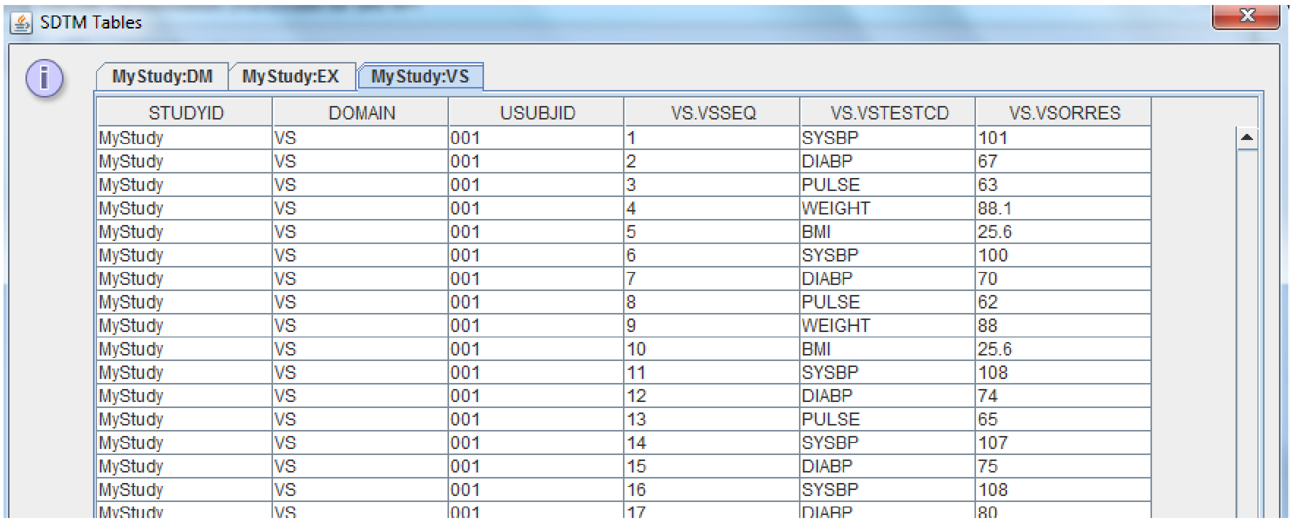
we see that the "Generalize for all Items" is still checked and the "Only for" still shows 5 inclusions. Clicking "OK" then leads to:


The Transformation Script

```
# Mapping using ODM element ItemData with ItemOID IT.SYSBP
# Generalized for all Items within the ItemGroup
$VS.VSORRES = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGro
```

ending with "@Value".

Execution of the mappings then shows:

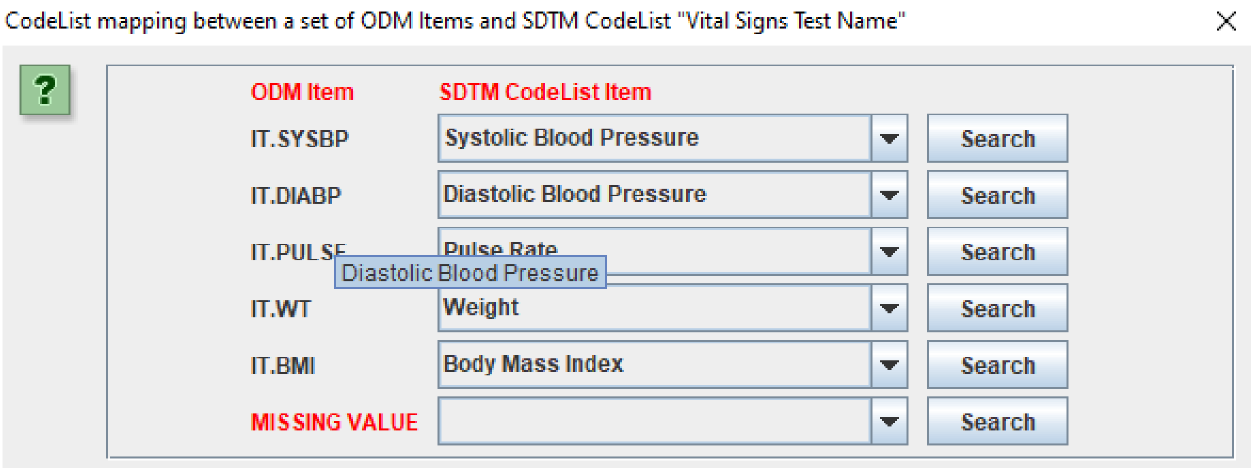

SDTM Tables

MyStudy:DM MyStudy:EX MyStudy:VS

STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES
MyStudy	VS	001	1	SYSBP	101
MyStudy	VS	001	2	DIABP	67
MyStudy	VS	001	3	PULSE	63
MyStudy	VS	001	4	WEIGHT	88.1
MyStudy	VS	001	5	BMI	25.6
MyStudy	VS	001	6	SYSBP	100
MyStudy	VS	001	7	DIABP	70
MyStudy	VS	001	8	PULSE	62
MyStudy	VS	001	9	WEIGHT	88
MyStudy	VS	001	10	BMI	25.6
MyStudy	VS	001	11	SYSBP	108
MyStudy	VS	001	12	DIABP	74
MyStudy	VS	001	13	PULSE	65
MyStudy	VS	001	14	SYSBP	107
MyStudy	VS	001	15	DIABP	75
MyStudy	VS	001	16	SYSBP	108
MyStudy	VS	001	17	DIABP	80

Let us now also add "VSTEST" and "VSORRESU".

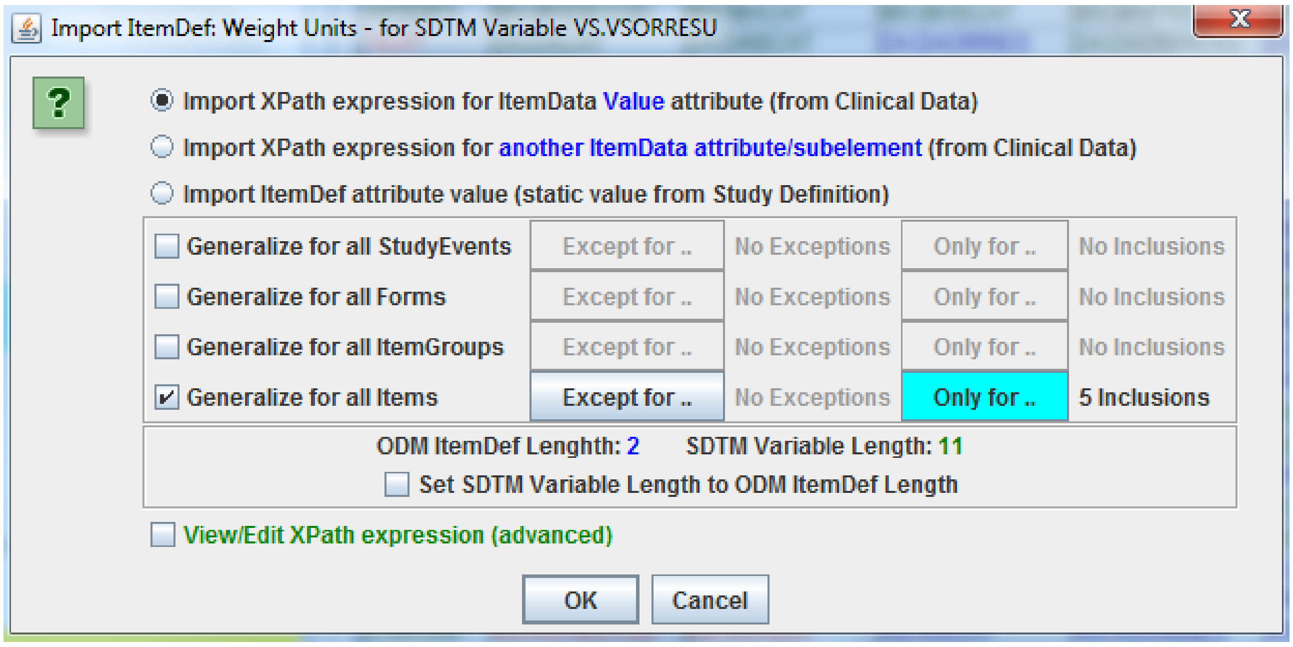
For the first, we can simply repeat the mapping as we did for VSTESTCD (using the "ItemOID"), and use the mapping wizard:


CodeList mapping between a set of ODM Items and SDTM CodeList "Vital Signs Test Name"

ODM Item	SDTM CodeList Item
IT.SYSBP	Systolic Blood Pressure
IT.DIABP	Diastolic Blood Pressure
IT.PULSE	Pulse Rate
IT.WT	Weight
IT.BMI	Body Mass Index
MISSING VALUE	

For the second (Vital Signs Original Results Unit), we need to be a little bit more careful, as we only have units in the source data for "weight" (which can be either "pounds" or "kilograms"), but of course we also need to add them for "pulse rate" and for the "blood pressure". Probably these were pre-printed on the CRF.

Drag-and-drop "Weight Units" to VS.VSORRESU. The mapping wizard again helps us:


Import ItemDef: Weight Units - for SDTM Variable VS.VSORRESU

(●) Import XPath expression for ItemData Value attribute (from Clinical Data)
( ) Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
( ) Import ItemDef attribute value (static value from Study Definition)

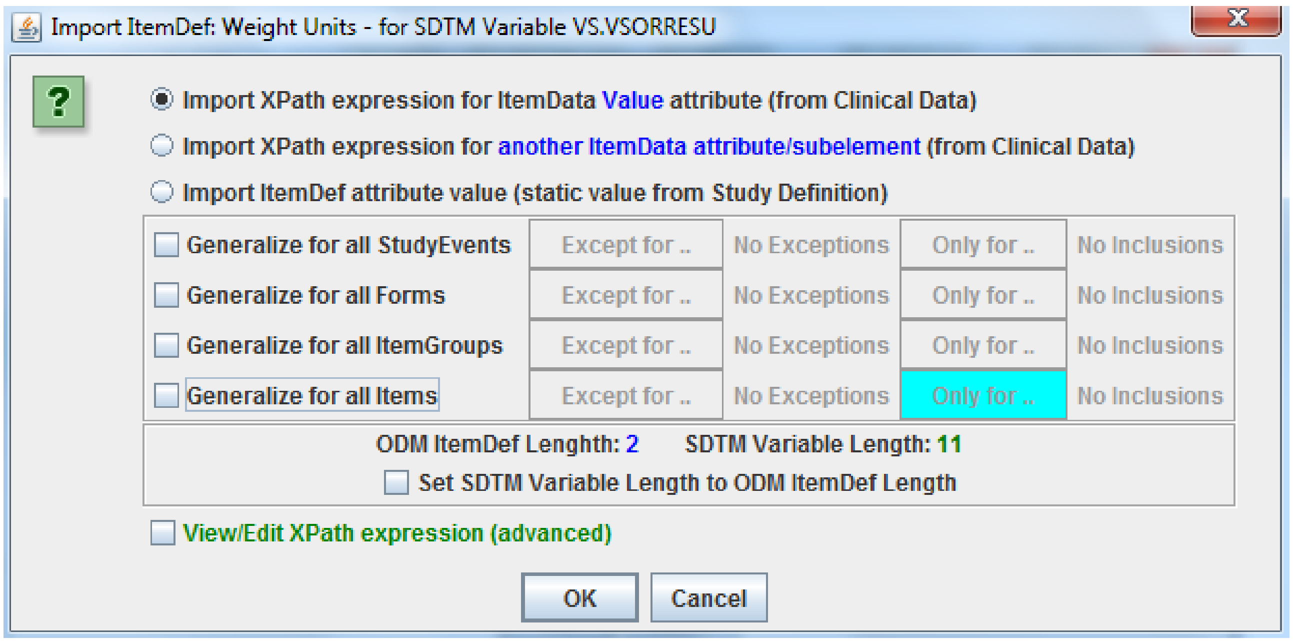
☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☑ Generalize for all Items	Except for ..	No Exceptions	Only for ..	5 Inclusions

ODM ItemDef Lenghth: 2   SDTM Variable Length: 11
☐ Set SDTM Variable Length to ODM ItemDef Length

☐ View/Edit XPath expression (advanced)

OK   Cancel

But we don't want to "generalize" anymore, as we will use "Weight units" only. So we uncheck the "Generalize for all Items" button:


Import ItemDef: Weight Units - for SDTM Variable VS.VSORRESU

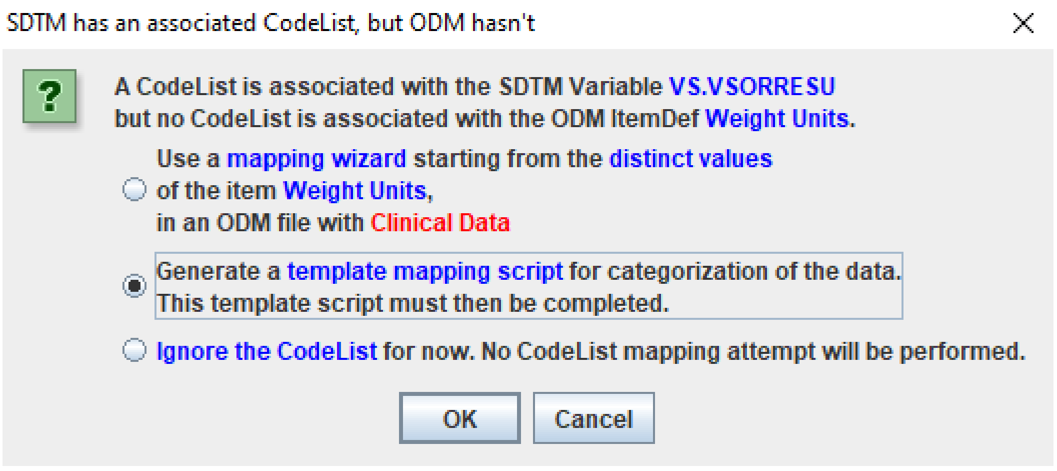
[✓ Import XPath expression for ItemData Value attribute (from Clinical Data)]
Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
Import ItemDef attribute value (static value from Study Definition)

☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Items	Except for ..	No Exceptions	Only for ..	No Inclusions

ODM ItemDef Lenghth: 2 SDTM Variable Length: 11
☐ Set SDTM Variable Length to ODM ItemDef Length

☐ View/Edit XPath expression (advanced)

and clicking "OK" leads to the following dialog:


SDTM has an associated CodeList, but ODM hasn't

A CodeList is associated with the SDTM Variable VS.VSORRESU
but no CodeList is associated with the ODM ItemDef Weight Units.

Use a mapping wizard starting from the distinct values of the item Weight Units, in an ODM file with Clinical Data

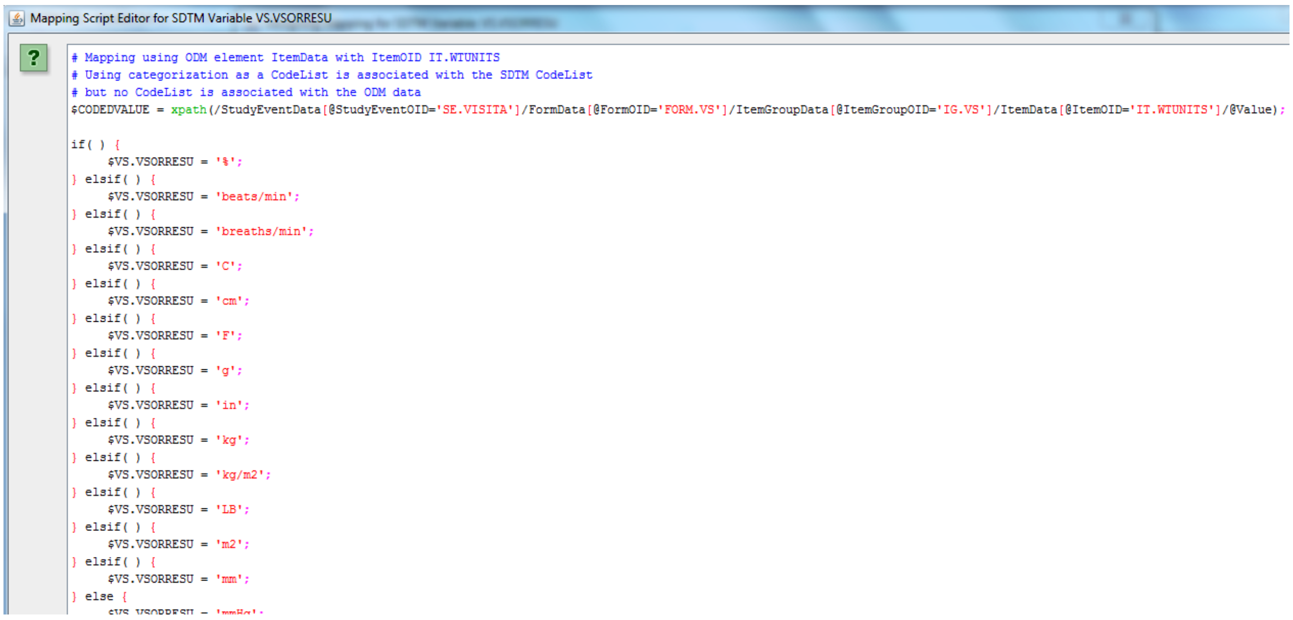
[✓ Generate a template mapping script for categorization of the data. This template script must then be completed.]

Ignore the CodeList for now. No CodeList mapping attempt will be performed.

The reason is that (unfortunately!) CDISC decided to associate an (although extensible) codelist to VSORRESU, making "original" "not original" anymore. As such, we need to map what was on the CRF as unit to a CDISC unit[10]. So we will allow the system to generate a template mapping scripr, and then fill that for the cases of "blood pressure", "pulse", "weight", and "BMI".

This leads to:

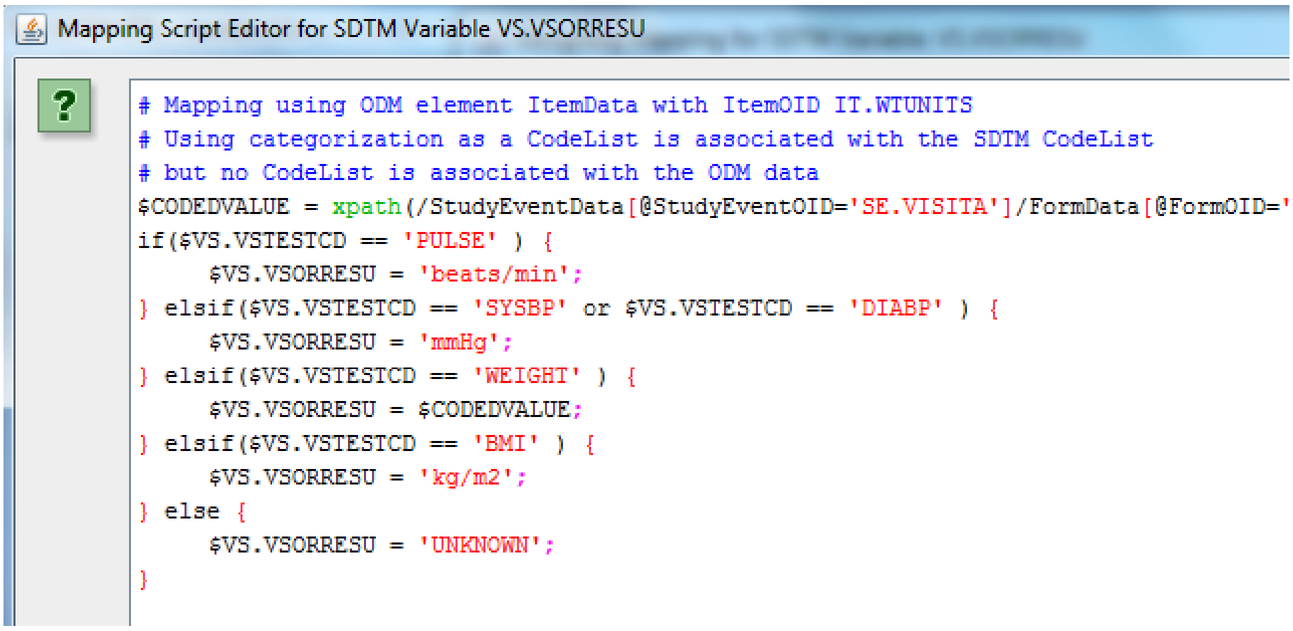
[10]This means that the reviewer can never know what unit was really on the CRF (original?). Also very unfortunately, CDISC does not allow the use of UCUM notation units, so that all units coming from electronic health records msut be mapped to the CDISC units. UCUM is an international notation for units and mandatory for the use in electronic health records.



Mapping Script Editor for SDTM Variable VS.VSORRESU

```
# 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
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.WTUNITS']/@Value);

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';
} elsif( ) {
     $VS.VSORRESU = 'kg';
} elsif( ) {
     $VS.VSORRESU = 'kg/m2';
} elsif( ) {
     $VS.VSORRESU = 'LB';
} elsif( ) {
     $VS.VSORRESU = 'm2';
} elsif( ) {
     $VS.VSORRESU = 'mm';
} else {
     $VS.VSORRESU = 'mmHg';
```

We can however reuse the value of VS.VSTESTCD in "read-mode", so that the script can be rewritten to:



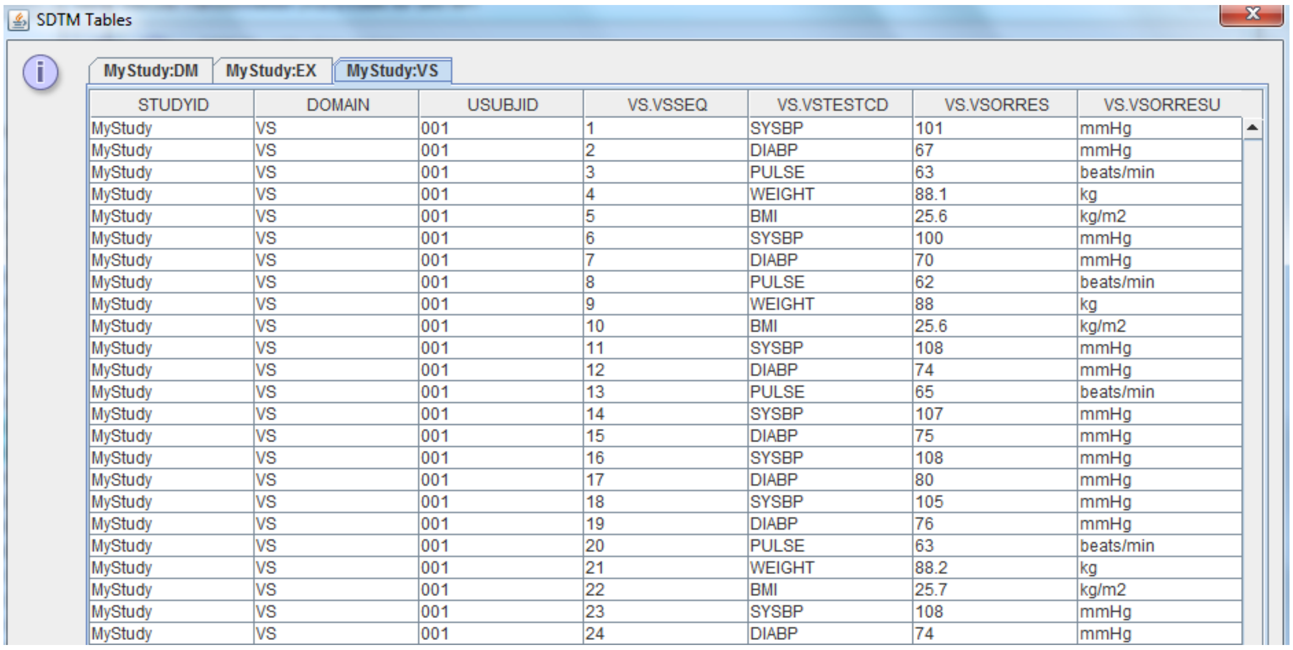
Mapping Script Editor for SDTM Variable VS.VSORRESU

```
# 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
$CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='
if($VS.VSTESTCD == 'PULSE' ) {
     $VS.VSORRESU = 'beats/min';
} elsif($VS.VSTESTCD == 'SYSBP' or $VS.VSTESTCD == 'DIABP' ) {
     $VS.VSORRESU = 'mmHg';
} elsif($VS.VSTESTCD == 'WEIGHT' ) {
     $VS.VSORRESU = $CODEDVALUE;
} elsif($VS.VSTESTCD == 'BMI' ) {
     $VS.VSORRESU = 'kg/m2';
} else {
     $VS.VSORRESU = 'UNKNOWN';
}
```

In the fourth line we retrieve the "weight units" from the ODM and store it in $CODEDVALUE. Then we test on the test code, and when the value of VSTESTCD is "WEIGHT", then we pick up the value from "$CODEDVALUE" again. In all other cases, we just assign the value from the CDISC controlled terminology, based on VSTESTCD. In case we cannot do such an assignment, we set "UNKNOWN". We could also assign "", meaning that in any other case, there is supposed to be no unit for the test code[11].

[11]This is often the case for lab results, e.g. for "pH" as well as for tests that do are not quantitative, like "URINE COLOR". In "Vital Signs", having no unit is rather unusual.

The result of the mapping execution is:



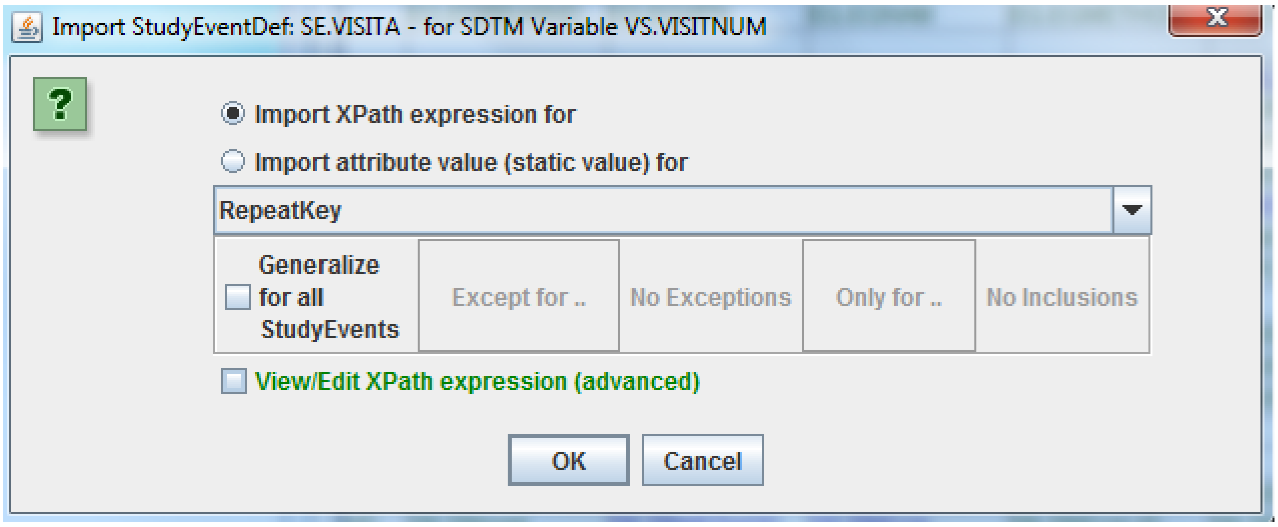
SDTM Tables

MyStudy:DM | MyStudy:EX | MyStudy:VS

STUDYID	DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU
MyStudy	VS	001	1	SYSBP	101	mmHg
MyStudy	VS	001	2	DIABP	67	mmHg
MyStudy	VS	001	3	PULSE	63	beats/min
MyStudy	VS	001	4	WEIGHT	88.1	kg
MyStudy	VS	001	5	BMI	25.6	kg/m2
MyStudy	VS	001	6	SYSBP	100	mmHg
MyStudy	VS	001	7	DIABP	70	mmHg
MyStudy	VS	001	8	PULSE	62	beats/min
MyStudy	VS	001	9	WEIGHT	88	kg
MyStudy	VS	001	10	BMI	25.6	kg/m2
MyStudy	VS	001	11	SYSBP	108	mmHg
MyStudy	VS	001	12	DIABP	74	mmHg
MyStudy	VS	001	13	PULSE	65	beats/min
MyStudy	VS	001	14	SYSBP	107	mmHg
MyStudy	VS	001	15	DIABP	75	mmHg
MyStudy	VS	001	16	SYSBP	108	mmHg
MyStudy	VS	001	17	DIABP	80	mmHg
MyStudy	VS	001	18	SYSBP	105	mmHg
MyStudy	VS	001	19	DIABP	76	mmHg
MyStudy	VS	001	20	PULSE	63	beats/min
MyStudy	VS	001	21	WEIGHT	88.2	kg
MyStudy	VS	001	22	BMI	25.7	kg/m2
MyStudy	VS	001	23	SYSBP	108	mmHg
MyStudy	VS	001	24	DIABP	74	mmHg

looking very good.

"VISIT" and "VISITNUM" can easily be generated from the "StudyEventRepeatKey" of the StudyEvent. For example for VISITNUM, drag-and-drop "StudyEvent: Treatment" to the cell VISITNUM, and select to use "RepeatKey":



Import StudyEventDef: SE.VISITA - for SDTM Variable VS.VISITNUM

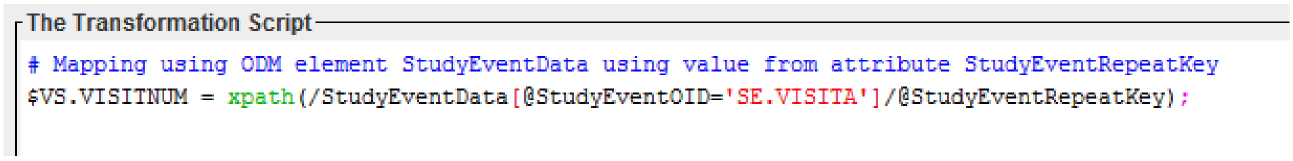
[✓ Import XPath expression for]

Import attribute value (static value) for

RepeatKey

Generalize for all StudyEvents
Except for ..
No Exceptions
Only for ..
No Inclusions
View/Edit XPath expression (advanced)
OK
Cancel

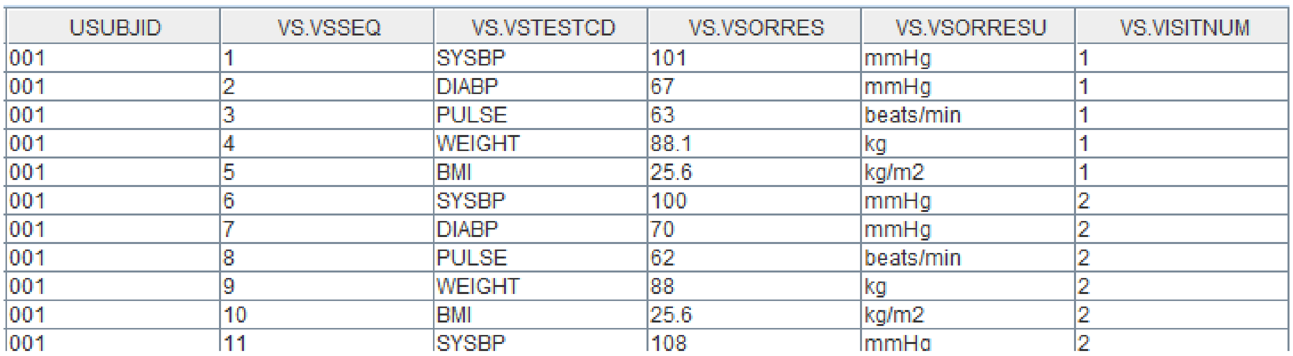
This will lead to:



The Transformation Script

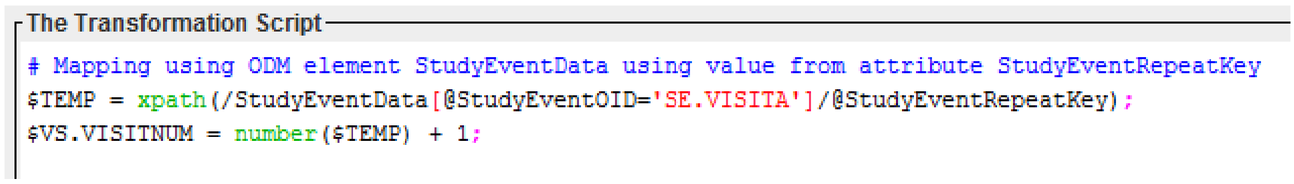
```
# Mapping using ODM element StudyEventData using value from attribute StudyEventRepeatKey
$VS.VISITNUM = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/@StudyEventRepeatKey);
```

and the result is:



USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VISITNUM
001	1	SYSBP	101	mmHg	1
001	2	DIABP	67	mmHg	1
001	3	PULSE	63	beats/min	1
001	4	WEIGHT	88.1	kg	1
001	5	BMI	25.6	kg/m2	1
001	6	SYSBP	100	mmHg	2
001	7	DIABP	70	mmHg	2
001	8	PULSE	62	beats/min	2
001	9	WEIGHT	88	kg	2
001	10	BMI	25.6	kg/m2	2
001	11	SYSBP	108	mmHg	2

if we do not want to have the "Pre-treatment" visit have VISITNUM=0 (there are essentially no rules for this), we can change our script into:

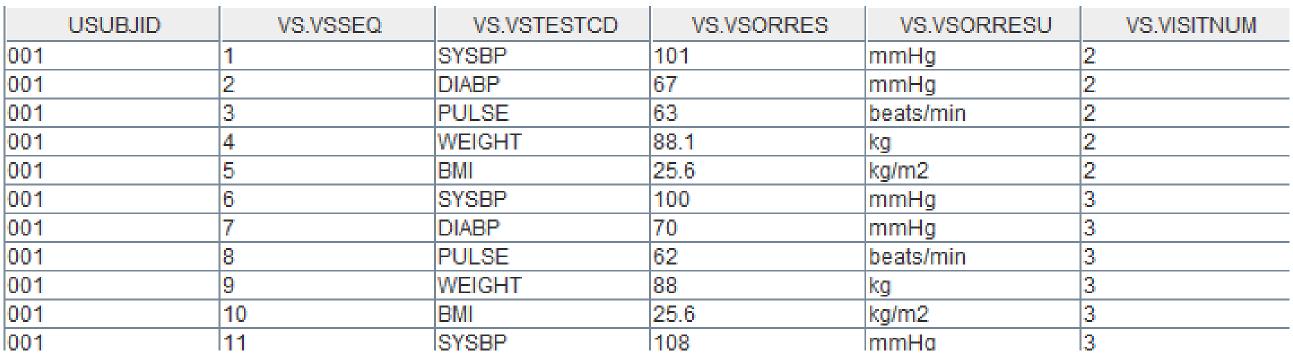


The Transformation Script

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

Do not forget the "number(..)" as the outcome of $TEMP is a text, not a number yet.

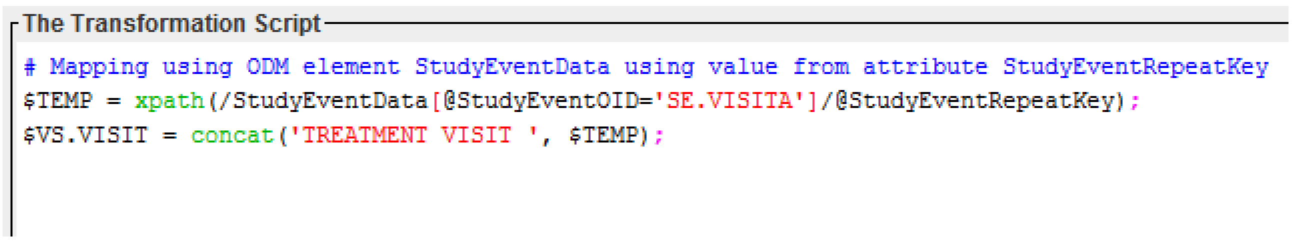
This leads to:



USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VISITNUM
001	1	SYSBP	101	mmHg	2
001	2	DIABP	67	mmHg	2
001	3	PULSE	63	beats/min	2
001	4	WEIGHT	88.1	kg	2
001	5	BMI	25.6	kg/m2	2
001	6	SYSBP	100	mmHg	3
001	7	DIABP	70	mmHg	3
001	8	PULSE	62	beats/min	3
001	9	WEIGHT	88	kg	3
001	10	BMI	25.6	kg/m2	3
001	11	SYSBP	108	mmHg	3

Unfortunately, CDISC has not published any recommendations yet for repeating visits (as e.g. very usual in oncology studies with cycles). So you will to develop your own strategy. Important to know is that the visit numbers do not need to be subsequent, so when your pre-treatment study is assigned VISITNUM=1, you are free to assign VISITNUM=101 to the first treatment visit.

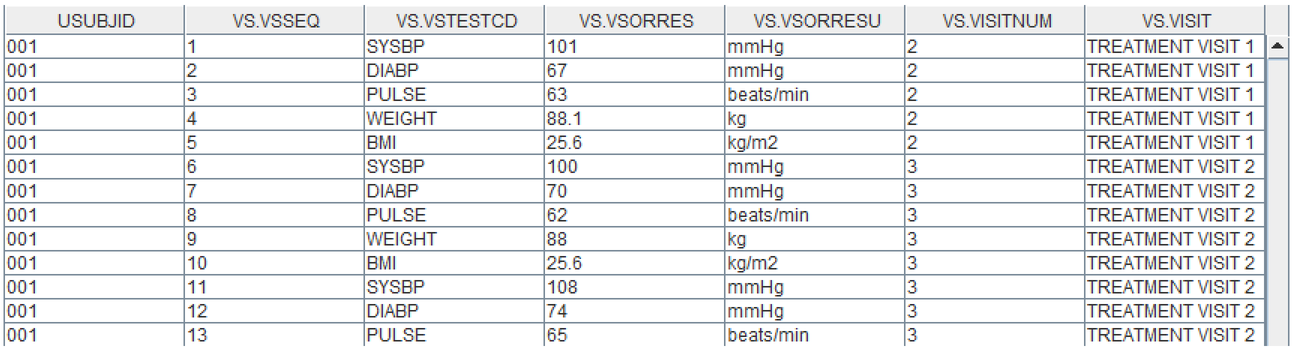
For VISIT (Visit Name) we can do something similar, like:



The Transformation Script

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

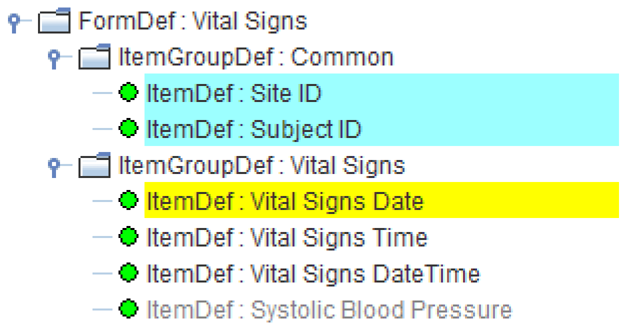
leading to:



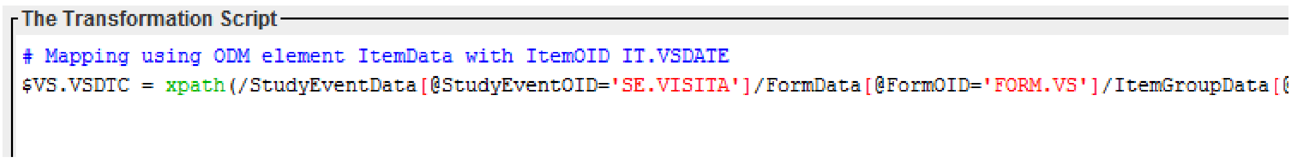
USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VISITNUM	VS.VISIT
001	1	SYSBP	101	mmHg	2	TREATMENT VISIT 1
001	2	DIABP	67	mmHg	2	TREATMENT VISIT 1
001	3	PULSE	63	beats/min	2	TREATMENT VISIT 1
001	4	WEIGHT	88.1	kg	2	TREATMENT VISIT 1
001	5	BMI	25.6	kg/m2	2	TREATMENT VISIT 1
001	6	SYSBP	100	mmHg	3	TREATMENT VISIT 2
001	7	DIABP	70	mmHg	3	TREATMENT VISIT 2
001	8	PULSE	62	beats/min	3	TREATMENT VISIT 2
001	9	WEIGHT	88	kg	3	TREATMENT VISIT 2
001	10	BMI	25.6	kg/m2	3	TREATMENT VISIT 2
001	11	SYSBP	108	mmHg	3	TREATMENT VISIT 2
001	12	DIABP	74	mmHg	3	TREATMENT VISIT 2
001	13	PULSE	65	beats/min	3	TREATMENT VISIT 2

Remark that you will later also need to add the visit numbers and names to the trial visit dataset (TV), which might be some work in the case of repeating visits and e.g. there was one subject who had 67 repeats. For SDTM, these are still 67 different visits!

Also the value for VSDTC (date/time of collection) can easily be obtained, as it was on the CRF (item "Vital Signs Date"). In this part of the exercise, we will just use the date part, demonstrating the data quality issues this can bring.

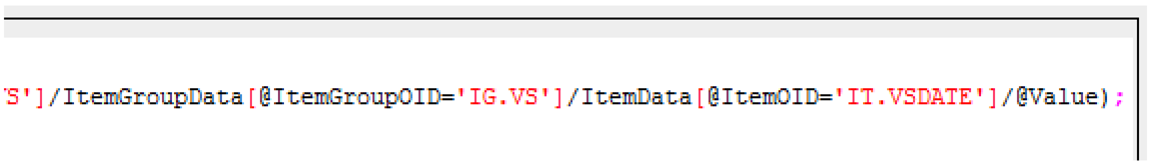

FormDef : Vital Signs
ItemGroupDef : Common
ItemDef : Site ID
ItemDef : Subject ID
ItemGroupDef : Vital Signs
ItemDef : Vital Signs Date
ItemDef : Vital Signs Time
ItemDef : Vital Signs DateTime
ItemDef : Systolic Blood Pressure

Just drag-and-drop the Item "Vital Signs Date" to the cell "VS.VSDTC". Do not check any of the "Generalization" checkboxes. This leads to the script:


The Transformation Script

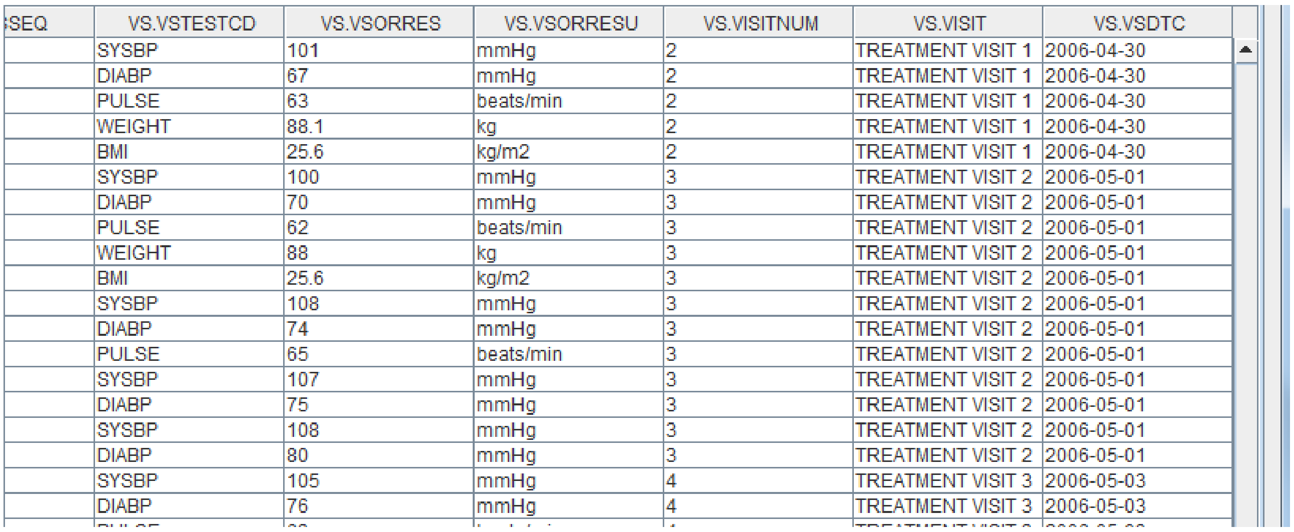
```
# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VS.VSDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@
```

ending with:



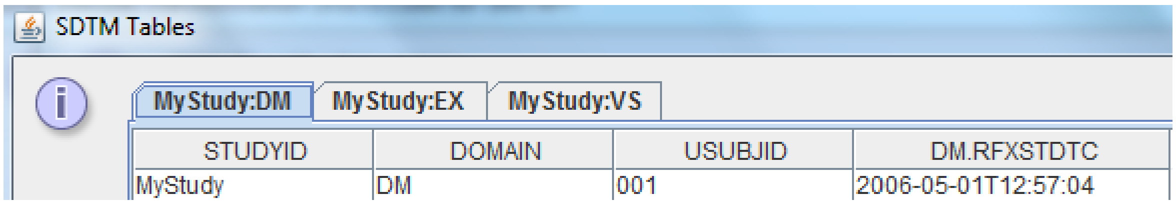
```
S']/ItemGroupData[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.VSDATE']/@Value);
```

and the execution result being:



SEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VISITNUM	VS.VISIT	VS.VSDTC
	SYSBP	101	mmHg	2	TREATMENT VISIT 1	2006-04-30
	DIABP	67	mmHg	2	TREATMENT VISIT 1	2006-04-30
	PULSE	63	beats/min	2	TREATMENT VISIT 1	2006-04-30
	WEIGHT	88.1	kg	2	TREATMENT VISIT 1	2006-04-30
	BMI	25.6	kg/m2	2	TREATMENT VISIT 1	2006-04-30
	SYSBP	100	mmHg	3	TREATMENT VISIT 2	2006-05-01
	DIABP	70	mmHg	3	TREATMENT VISIT 2	2006-05-01
	PULSE	62	beats/min	3	TREATMENT VISIT 2	2006-05-01
	WEIGHT	88	kg	3	TREATMENT VISIT 2	2006-05-01
	BMI	25.6	kg/m2	3	TREATMENT VISIT 2	2006-05-01
	SYSBP	108	mmHg	3	TREATMENT VISIT 2	2006-05-01
	DIABP	74	mmHg	3	TREATMENT VISIT 2	2006-05-01
	PULSE	65	beats/min	3	TREATMENT VISIT 2	2006-05-01
	SYSBP	107	mmHg	3	TREATMENT VISIT 2	2006-05-01
	DIABP	75	mmHg	3	TREATMENT VISIT 2	2006-05-01
	SYSBP	108	mmHg	3	TREATMENT VISIT 2	2006-05-01
	DIABP	80	mmHg	3	TREATMENT VISIT 2	2006-05-01
	SYSBP	105	mmHg	4	TREATMENT VISIT 3	2006-05-03
	DIABP	76	mmHg	4	TREATMENT VISIT 3	2006-05-03

When looking at DM, we also see:



SDTM Tables

MyStudy:DM | MyStudy:EX | MyStudy:VS

STUDYID	DOMAIN	USUBJID	DM.RFXSTDTC
MyStudy	DM	001	2006-05-01T12:57:04

with the first date of exposure being 2006-05-01.
So it will already be clear that the measurements on 2006-04-30 will <u>not</u> be baseline values.

Calculating the baseline flag

Let us now start the calculation of the baseline flag using VSBLFL. This is the most usual way when using SDTM-IG 3.2 or earlier.
For this we need to know all measurement dates for vital signs measurements, and compare these with the "date of first study treatment" for which we developed a mapping in the global variable RFXSTDTC.

So we need to find all measurements, and this for each different test code, the last measurement before the first treatment date, or on the first treatment date.

One can already see the data quality problem here: if our datetime of first exposure is 2006-05-01, we do not know whether measurements on the same date were before first exposure. Of course it can be stated that all measurements on the date of first exposure need to be done before the first exposure, but this will not be clear from the data themselves.

For VSSTDTC we had the mapping:

# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VS.VSDTC =
xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']
/ItemGroupData[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.VSDATE']/@Value);

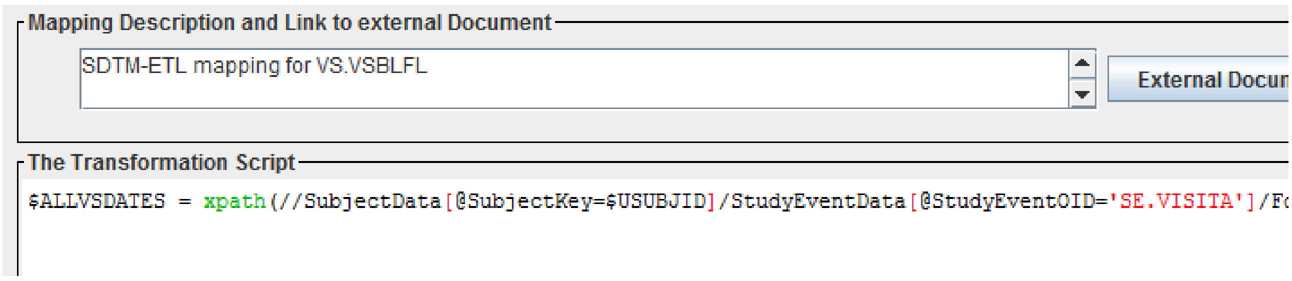
as we iterate over all tests, this gives one value per row.

In order to find ALL measurement dates however, we need to adapt this XPath expression to:

xpath(**//SubjectData[@SubjectKey=$USUBJID]**/StudyEventData[@StudyEventOID='SE.VISITA']
/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemGroupOID='IG.VS']
/ItemData[@ItemOID='IT.VSDATE']/@Value);

Remark the //SubjectData[@SubjectKey=$USUBJID] in front.
It selects all data for which the subject ID is equal to our value of USUBJID[12]. As it starts with "//", this time this is an absolute path in the ODM, and not relative to our looping variable.

We can now use this in our script for VS.VSBLFL:

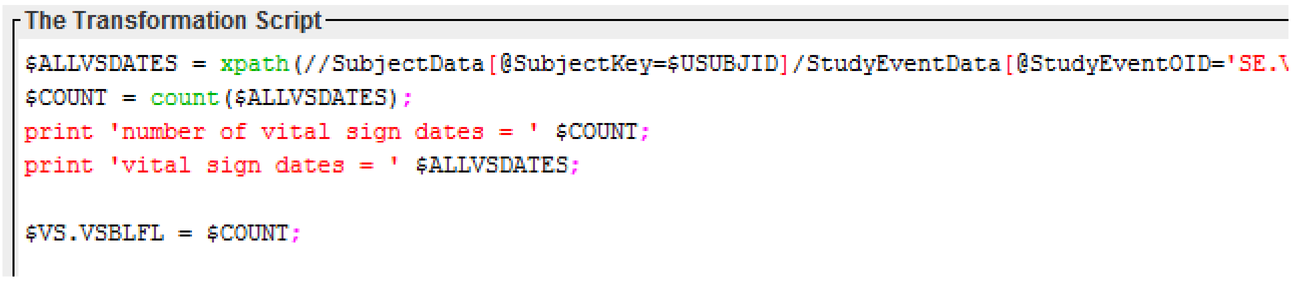

Mapping Description and Link to external Document
External Docur
The Transformation Script

SDTM-ETL mapping for VS.VSBLFL

```
$ALLVSDATES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISITA']/Fo
```

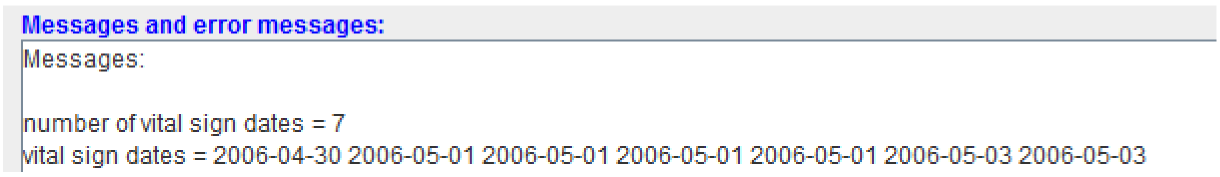
we can check that this delivers more then one date, e.g. using:

[12]In case we did not use the subject ID from the ODM for USUBJID, the expression will be a little bit more difficult, we could then e.g. create a "global" variable SUBJID for the ID in the ODM, and use that in the XPath expression instead of USUBJID.



```
$ALLVSDATES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.\
$COUNT = count($ALLVSDATES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATES;

$VS.VSBLFL = $COUNT;
```

and upon executing the mappings, we see the following message coming by:



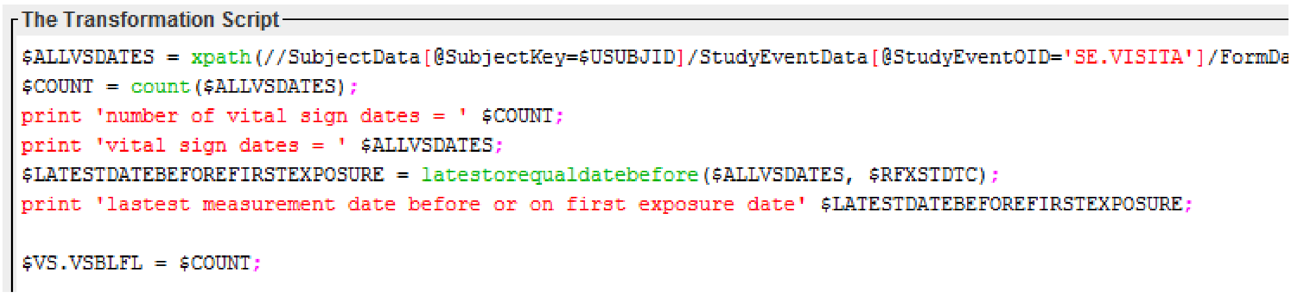
```
Messages:

number of vital sign dates = 7
vital sign dates = 2006-04-30 2006-05-01 2006-05-01 2006-05-01 2006-05-01 2006-05-03 2006-05-03
```

But in order to assign a baseline flag, we need to get the latest measurement or measurements before or on the first treatment date (which we stored in RFXSTDTC).

For this, a new function was introduced in SDTM-ETL v.3.1[13]:

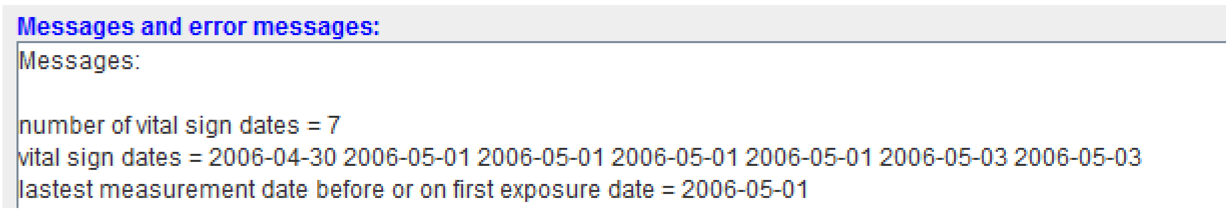
function latestorequaldatebefore($dates, $referencedate).

It takes a series of dates (array) as the first argument, and a reference date as the second argument, and returns a single date that is the lastest date before or on the reference date.
So let us use this in the mapping script:



```
$ALLVSDATES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISITA']/FormDa
$COUNT = count($ALLVSDATES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATES;
$LATESTDATEBEFOREFIRSTEXPOSURE = latestorequaldatebefore($ALLVSDATES, $RFXSTDTC);
print 'lastest measurement date before or on first exposure date' $LATESTDATEBEFOREFIRSTEXPOSURE;

$VS.VSBLFL = $COUNT;
```

and when executing the script, we see the message:

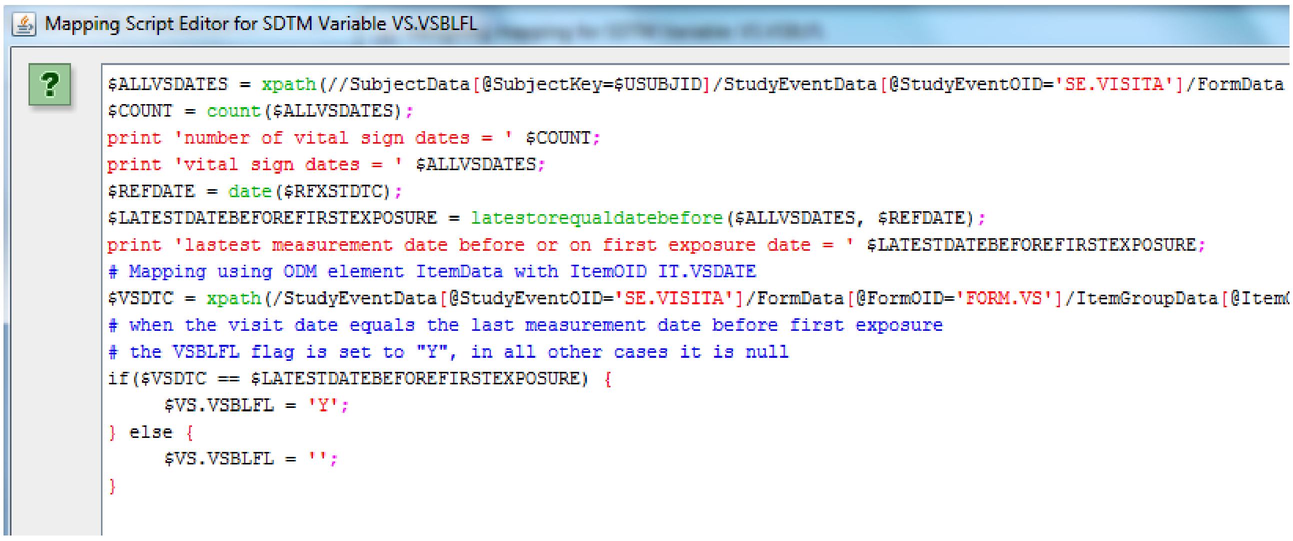


```
Messages:

number of vital sign dates = 7
vital sign dates = 2006-04-30 2006-05-01 2006-05-01 2006-05-01 2006-05-01 2006-05-03 2006-05-03
lastest measurement date before or on first exposure date = 2006-05-01
```

Unfortunately, CDISC has put the VSBLFL variable before VSDTC, not taking into account that one needs the second to calculate the first. So we cannot simply use $VS.VSDTC but need to retrieve it again. This can easily be done using "drag-and-drop" from "Vital Signs Date" again.

[13]See the document "Scripting Language Specification" v.3.1

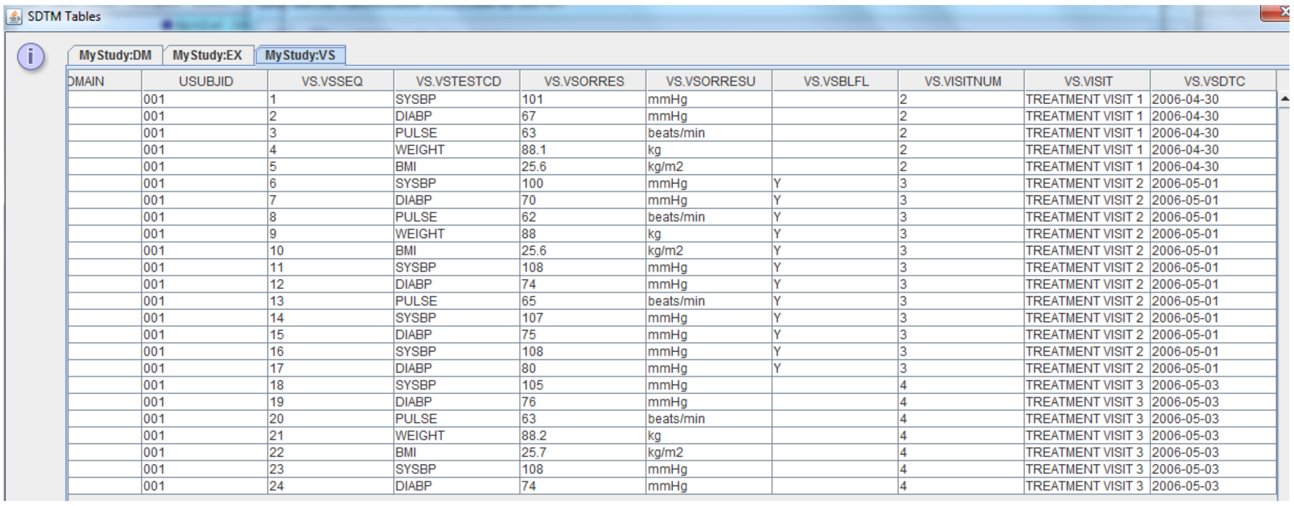
We then compare the value of VSDTC with the latest measurement date before first exposure:



```
$ALLVSDATES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISITA']/FormData
$COUNT = count($ALLVSDATES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATES;
$REFDATE = date($RFXSTDTC);
$LATESTDATEBEFOREFIRSTEXPOSURE = latestorequaldatebefore($ALLVSDATES, $REFDATE);
print 'lastest measurement date before or on first exposure date = ' $LATESTDATEBEFOREFIRSTEXPOSURE;
# Mapping using ODM element ItemData with ItemOID IT.VSDATE
$VSDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@Item
# when the visit date equals the last measurement date before first exposure
# the VSBLFL flag is set to "Y", in all other cases it is null
if($VSDTC == $LATESTDATEBEFOREFIRSTEXPOSURE) {
     $VS.VSBLFL = 'Y';
} else {
     $VS.VSBLFL = '';
}
```

Remark that we need the function "date()" in 5th line, as RFXSTDTC is a datetime, and not a date, and datetimes can essentially not be compared with dates without making assumptions for the case that the date part is equal.

The result is:



DMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VSBLFL	VS.VISITNUM	VS.VISIT	VS.VSDTC
	001	1	SYSBP	101	mmHg		2	TREATMENT VISIT 1	2006-04-30
	001	2	DIABP	67	mmHg		2	TREATMENT VISIT 1	2006-04-30
	001	3	PULSE	63	beats/min		2	TREATMENT VISIT 1	2006-04-30
	001	4	WEIGHT	88.1	kg		2	TREATMENT VISIT 1	2006-04-30
	001	5	BMI	25.6	kg/m2		2	TREATMENT VISIT 1	2006-04-30
	001	6	SYSBP	100	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	7	DIABP	70	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	8	PULSE	62	beats/min	Y	3	TREATMENT VISIT 2	2006-05-01
	001	9	WEIGHT	88	kg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	10	BMI	25.6	kg/m2	Y	3	TREATMENT VISIT 2	2006-05-01
	001	11	SYSBP	108	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	12	DIABP	74	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	13	PULSE	65	beats/min	Y	3	TREATMENT VISIT 2	2006-05-01
	001	14	SYSBP	107	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	15	DIABP	75	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	16	SYSBP	108	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	17	DIABP	80	mmHg	Y	3	TREATMENT VISIT 2	2006-05-01
	001	18	SYSBP	105	mmHg		4	TREATMENT VISIT 3	2006-05-03
	001	19	DIABP	76	mmHg		4	TREATMENT VISIT 3	2006-05-03
	001	20	PULSE	63	beats/min		4	TREATMENT VISIT 3	2006-05-03
	001	21	WEIGHT	88.2	kg		4	TREATMENT VISIT 3	2006-05-03
	001	22	BMI	25.7	kg/m2		4	TREATMENT VISIT 3	2006-05-03
	001	23	SYSBP	108	mmHg		4	TREATMENT VISIT 3	2006-05-03
	001	24	DIABP	74	mmHg		4	TREATMENT VISIT 3	2006-05-03

As one sees, all the measurements that were on the date of first exposure, have been marked with VSBLFL=Y, all earlier and later ones do not have the flag set.

One already sees the danger of such an approach: we here assume that all measurements on the date of exposure were done before the first treatment. If this was not the case, we have a serious data quality problem. Suppose e.g. that we have a blood pressure lowering agent, and there was a systolic blood pressure measurement with a value of 130 one hour after first study treatment. With our approach, this value would be marked with a baseline flag, whereas the reality is that the blood pressure was raised after the first treatment, maybe because of the treatment with the blood pressure lowering agent!

Sadly, this quality of clinical data is still often found in studies.
So, in order to have high quality data, one should not only capture the first exposure as a datetime (including the time part), but also capture the time part (and not only the date part) of any

measurement, especially vital sign measurements.

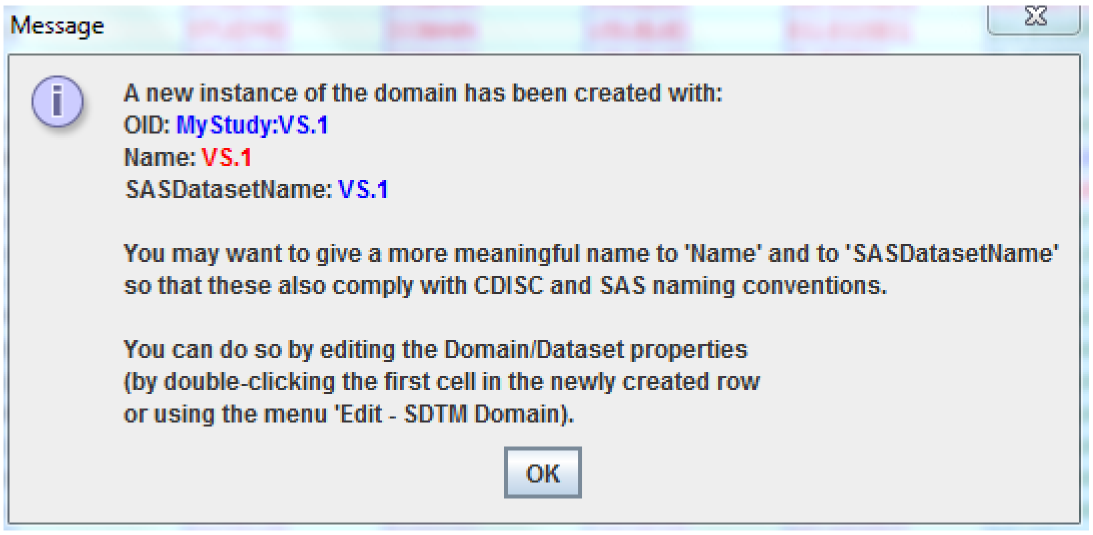
Calculating baseline flags with exact dates and times

Especially when measurements are performed on the first date of treatment or exposure, it is of utmost importance to capture the date and time of the measurement exactly. Also the first treatment has to be captured exactly, so with date and time.

This section will demonstrate how baseline flags can be generated when such exact information is available. The final results will then be compared with those of the case that only dates are used.

As we already have done a lot of mapping work, we just create a second instance of our VS domain (for demonstration purposes) by copy-and-paste the row for "MyStudy:VS". This can either be done by drag-and-dropping this row, or by selecting it and then use the menu "Edit - Copy Domain/Dataset" followed by "Edit - Paste Domain/Dataset".

In both cases, the following dialog is displayed:

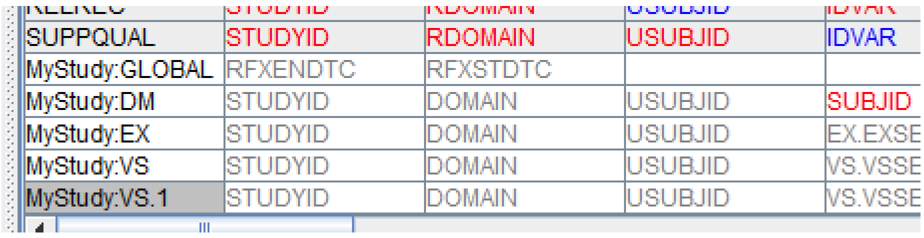


A new instance of the domain has been created with:
OID: MyStudy:VS.1
Name: VS.1
SASDatasetName: VS.1

You may want to give a more meaningful name to 'Name' and to 'SASDatasetName' so that these also comply with CDISC and SAS naming conventions.

You can do so by editing the Domain/Dataset properties
(by double-clicking the first cell in the newly created row
or using the menu 'Edit - SDTM Domain).

leading to an additional row in our SDTM table:

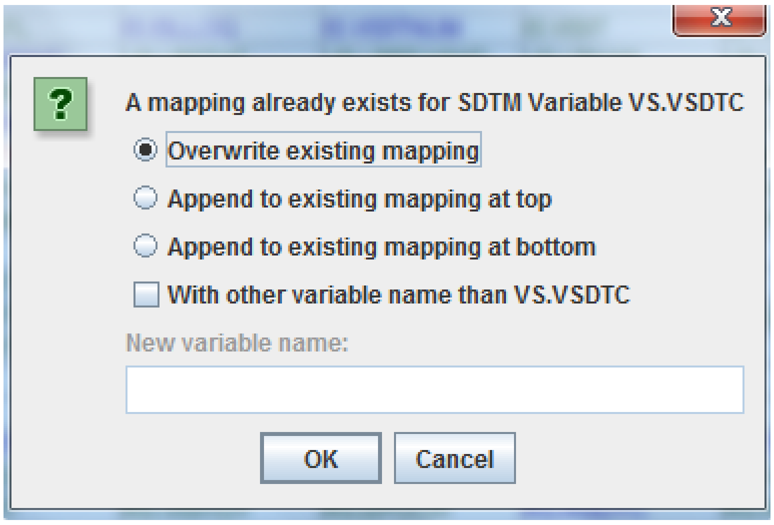


SUPPQUAL	STUDYID	RDOMAIN	USUBJID	IDVAR
MyStudy:GLOBAL	RFXENDTC	RFXSTDTC		
MyStudy:DM	STUDYID	DOMAIN	USUBJID	SUBJID
MyStudy:EX	STUDYID	DOMAIN	USUBJID	EX.EXSE
MyStudy:VS	STUDYID	DOMAIN	USUBJID	VS.VSSE
MyStudy:VS.1	STUDYID	DOMAIN	USUBJID	VS.VSSE

We will now start changing/editing some scripts for the new use case that we want to use datetime-s for all our measurements.

The first script we change is the one for VS.VSSTDTC. Instead of using item "Vital Signs Date" we use "Vital Signs DateTime".
We could also drag-and-drop both "Vital Signs Date" and "Vital Signs Time" and then combine them using the "createdatetime" function, but we will see later that for getting all measurement datetimes, we need "Vital Signs DateTime" anyway.

So, just drag-and-drop from "Vital Signs DateTime" to VS.VSDTC of row "VS.1" leads to:


A mapping already exists for SDTM Variable VS.VSDTC

[✓ Overwrite existing mapping]

Append to existing mapping at top

Append to existing mapping at bottom

☐ With other variable name than VS.VSDTC

New variable name:

where we select to "overwrite".
We will now have datetimes for the VSDTC instead of dates only.

As we do already have RFXSTDTC as **global** variable (datetime of first study treatment) as a datetime, we do not need to change anything there.

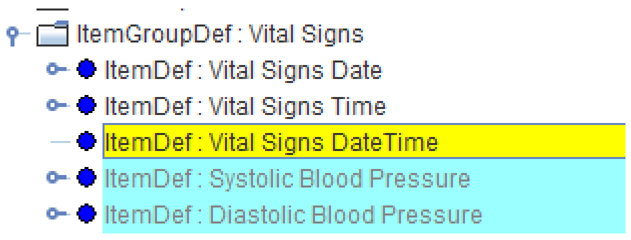
For the baseline flag, we need to make a few changes, as we do want to base it on datetimes instead of on dates alone.

First of all, we need to get all the measurement datetimes for the test corresponding to the current value of VSTESTCD. We will store these datetimes in an array variable $ALLVSDATETIMES.
So we first need to retrieve the value of VSTESTCD. The easiest is to to just copy the first line of the script for VS.VSTESTCD into VS.VSBLFL and give the variable another name, e.g. "CUROID" (as it is the current test OID):

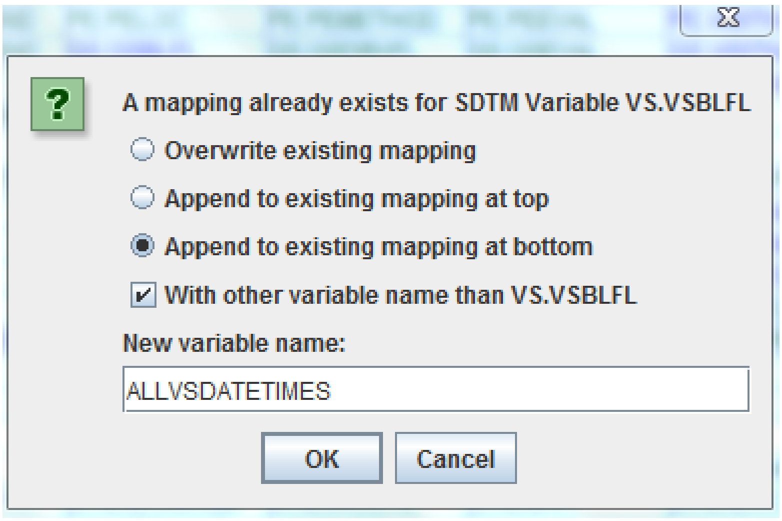
$CUROID =
xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.SYSBP' or @ItemOID='IT.DIABP' or @ItemOID='IT.PULSE' or @ItemOID='IT.WT' or @ItemOID='IT.BMI']/@ItemOID);

Remark that we select the OID (@ItemOID) at the end, and that we again only accept codes for SYSBP, DIABP, PULSE, WT and HT and BMI.

For the variable $ALLVSDATETIMES (all vital sign datetimes for the current test), we drag-and-drop "Vital Signs DateTime"


ItemGroupDef : Vital Signs
ItemDef : Vital Signs Date
ItemDef : Vital Signs Time
ItemDef : Vital Signs DateTime
ItemDef : Systolic Blood Pressure
ItemDef : Diastolic Blood Pressure

to VS.VSBLFL, and select to append it to the script with another name:



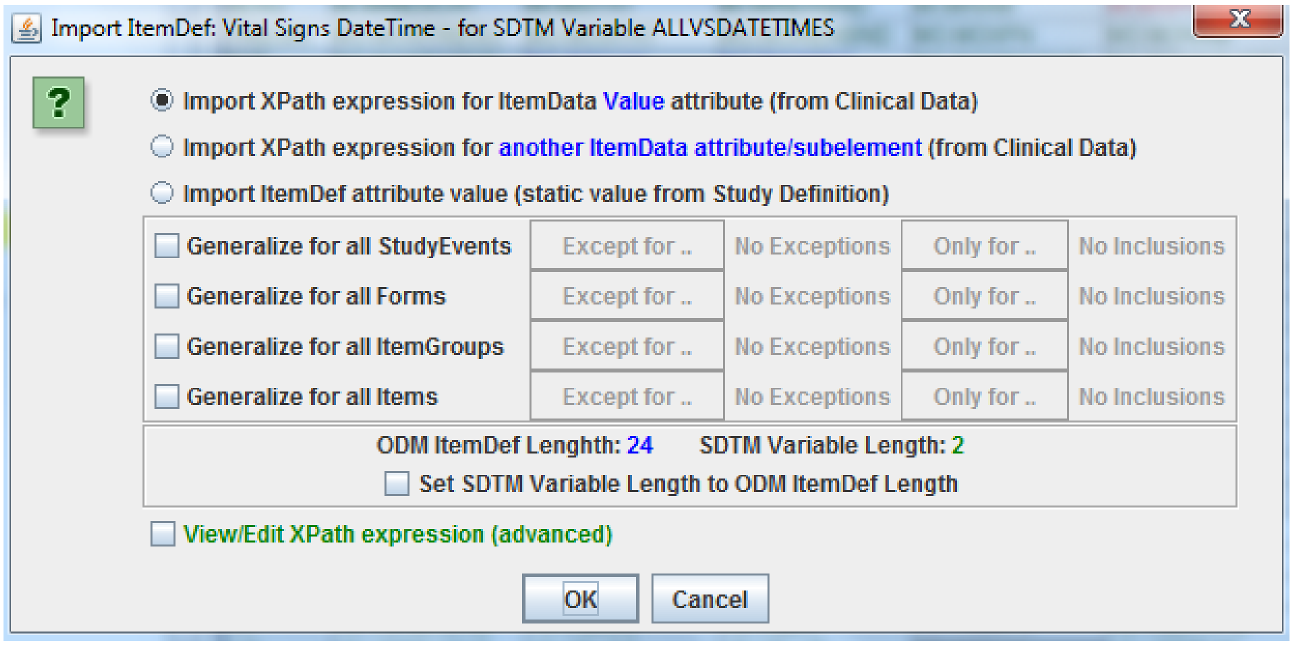
A mapping already exists for SDTM Variable VS.VSBLFL

- ○ Overwrite existing mapping
- ○ Append to existing mapping at top
- ● Append to existing mapping at bottom
- ☑ With other variable name than VS.VSBLFL

New variable name:

ALLVSDATETIMES

after clicking OK, this leads to:



Import ItemDef: Vital Signs DateTime - for SDTM Variable ALLVSDATETIMES

- ● Import XPath expression for ItemData Value attribute (from Clinical Data)
- ○ Import XPath expression for another ItemData attribute/subelement (from Clinical Data)
- ○ Import ItemDef attribute value (static value from Study Definition)

☐ Generalize for all StudyEvents	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Forms	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all ItemGroups	Except for ..	No Exceptions	Only for ..	No Inclusions
☐ Generalize for all Items	Except for ..	No Exceptions	Only for ..	No Inclusions

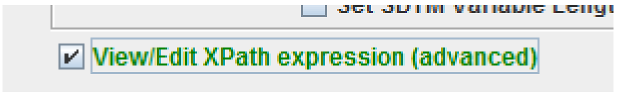
ODM ItemDef Lenghth: 24   SDTM Variable Length: 2

☐ Set SDTM Variable Length to ODM ItemDef Length

☐ View/Edit XPath expression (advanced)

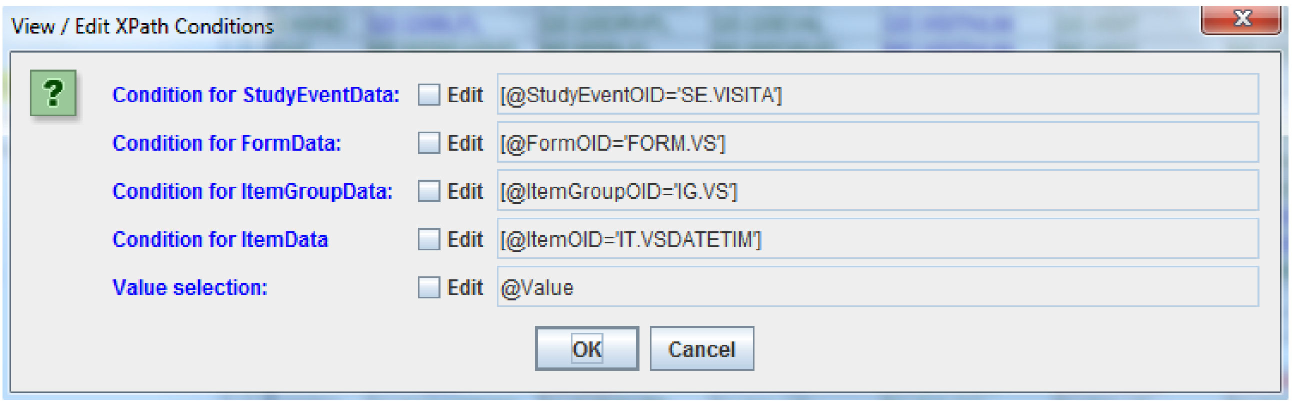
Ensure that the "Generalize for all Items" is <u>not</u> checked as we really only want to retrieve information from "Vital Signs DateTime" and not from any other source variable.

However, this would not filter on the current test code: we only want to get the datetimes for the current observation, not from other types of observations. So we need to do something special. There is a simple wizard for this, which is enabled by checking "View/Edit XPath expressions":



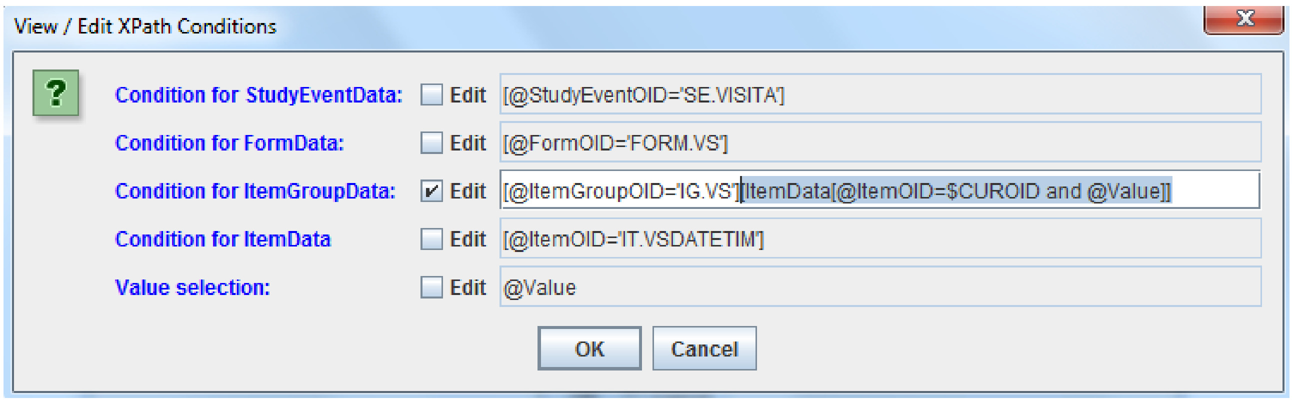
☑ View/Edit XPath expression (advanced)

This then leads to:


View / Edit XPath Conditions

Condition for StudyEventData:	☐ Edit	[@StudyEventOID='SE.VISITA']
Condition for FormData:	☐ Edit	[@FormOID='FORM.VS']
Condition for ItemGroupData:	☐ Edit	[@ItemGroupOID='IG.VS']
Condition for ItemData	☐ Edit	[@ItemOID='IT.VSDATETIM']
Value selection:	☐ Edit	@Value

where the filters are again displayed.
We do however require that we only get datetimes for which there really was a measurement for the current test (identified by CUROID). We can add this condition by editing the filter for "ItemGroupData":


View / Edit XPath Conditions

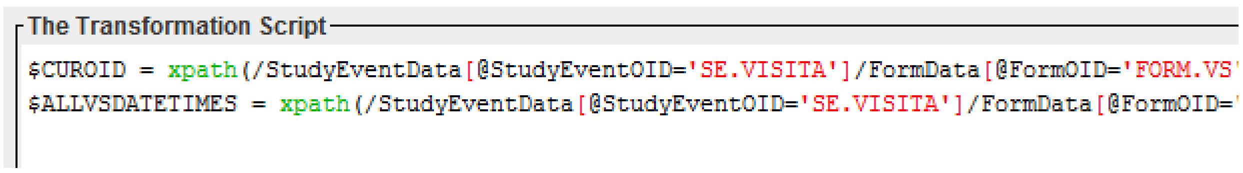
Condition for StudyEventData:	☐ Edit	[@StudyEventOID='SE.VISITA']
Condition for FormData:	☐ Edit	[@FormOID='FORM.VS']
Condition for ItemGroupData:	☑ Edit	[@ItemGroupOID='IG.VS'][ItemData[@ItemOID=$CUROID and @Value]]
Condition for ItemData	☐ Edit	[@ItemOID='IT.VSDATETIM']
Value selection:	☐ Edit	@Value

or a bit cleared:

[@ItemGroupOID='IG.VS'][ItemData[@ItemOID=$CUROID and @Value]]

what we state here is that there MUST be a data point for the current test (CUROID) in the current group, and that there MUST be a value for it[14].

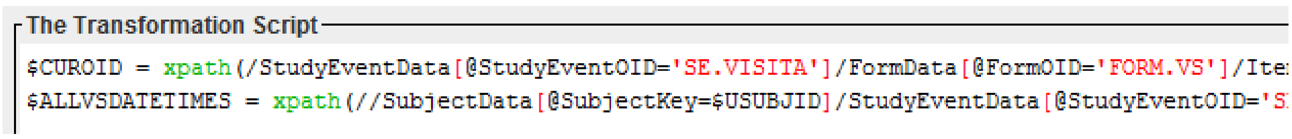
After clicking OK, we get:


The Transformation Script

```
$CUROID = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS'
$ALLVSDATETIMES = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='
```

$ALLVSDATETIMES would now only give one date as it will be interpreted against the current looping variable (which is VS.VSTESTCD). We do however want to have all datetimes for the current test and subject. We already added the condition for "the current test", but not yet for "the current subject". This can be easily done by adding "**//SubjectData[@SubjectKey=$USUBJID]**" at the beginning of the XPath expression[15]:

[14]Remark that ODM does not allow the construct Value="" (i.e. the empty value). In such a case the whole ItemData must be absent.
[15]In case one cannot use $USUBJID as it has been transformed from the original one in the ODM (e.g. study-site-comcatenation), one can always define a subject-global variable, e.g. "$ODMSUBJECTKEY", assign the ODM subject key to it, and then use that is

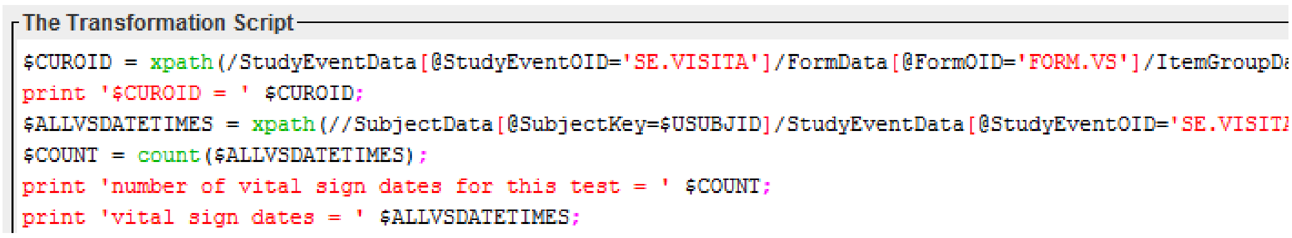


The Transformation Script

```
$CUROID = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/Ite
$ALLVSDATETIMES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='S
```

This is an important step, as one can otherwise obtain false positives.

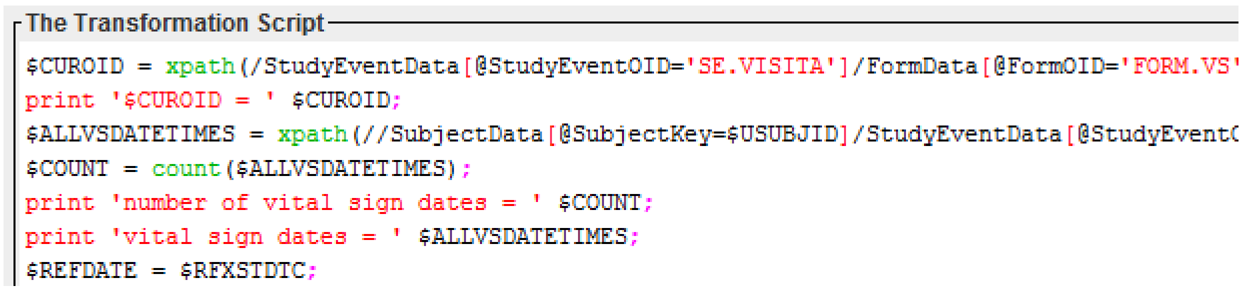
Not a bad idea either to add some "print" statements for during testing:



The Transformation Script

```
$CUROID = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupDa
print '$CUROID = ' $CUROID;
$ALLVSDATETIMES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISIT
$COUNT = count($ALLVSDATETIMES);
print 'number of vital sign dates for this test = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATETIMES;
```

We can now also add the datetime we will be comparing to. As we made RFXSTDTC a global variable, we can read it out here:

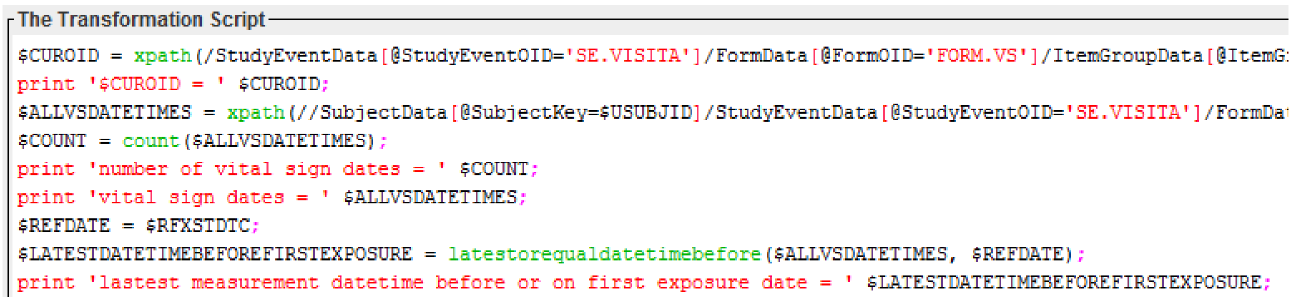


The Transformation Script

```
$CUROID = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS'
print '$CUROID = ' $CUROID;
$ALLVSDATETIMES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEvent
$COUNT = count($ALLVSDATETIMES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATETIMES;
$REFDATE = $RFXSTDTC;
```

So what we will be doing is compare the reference datatime (which is the first exposure datetime) with all the datetimes of observation for the current test, and retrieve the latest one from the array that is before or on the reference datetime.

There is a special function for this: latestorequaldatetimebefore(). It has two arguments, the first being the array with all datetimes, the second one being the reference datetime. So we add:

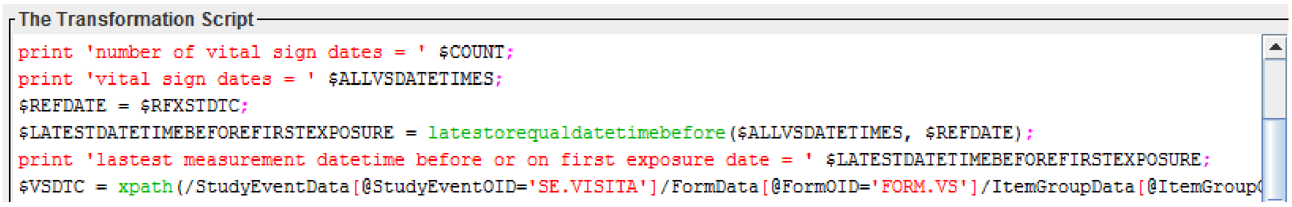


The Transformation Script

```
$CUROID = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemG
print '$CUROID = ' $CUROID;
$ALLVSDATETIMES = xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISITA']/FormDa
$COUNT = count($ALLVSDATETIMES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATETIMES;
$REFDATE = $RFXSTDTC;
$LATESTDATETIMEBEFOREFIRSTEXPOSURE = latestorequaldatetimebefore($ALLVSDATETIMES, $REFDATE);
print 'lastest measurement datetime before or on first exposure date = ' $LATESTDATETIMEBEFOREFIRSTEXPOSURE;
```

storing the latest date (for this test and subject) of observation that comes before the first exposure datetime in the variable $LATESTDATETIMEBEFOREFIRSTEXPOSURE.

We still need the current datetime of the observation, which is equal to the one from VS.VSTESTCD, so we can simply copy that mapping into our script. E.g.:

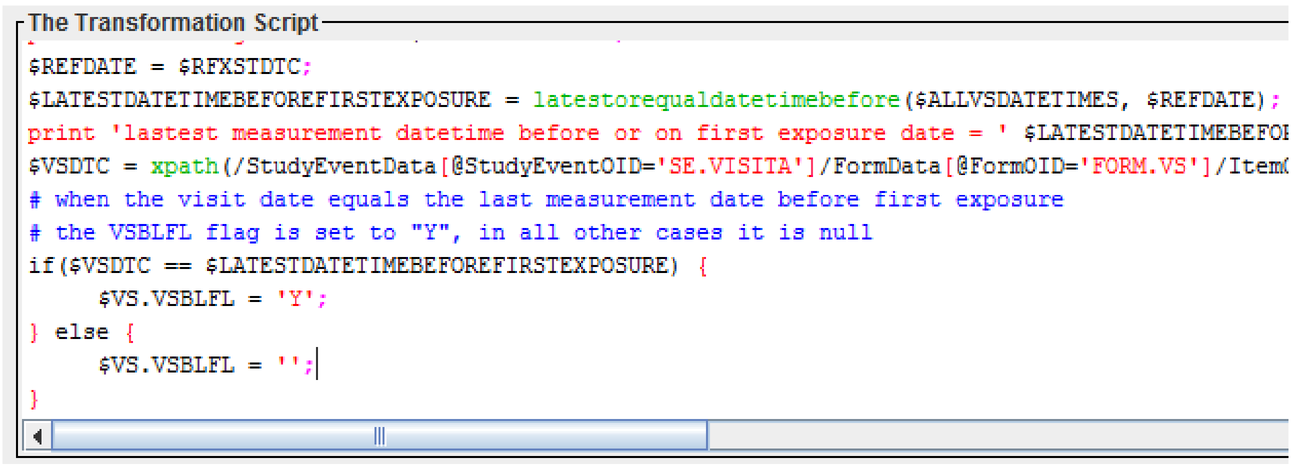


```
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATETIMES;
$REFDATE = $RFXSTDTC;
$LATESTDATETIMEBEFOREFIRSTEXPOSURE = latestorequaldatetimebefore($ALLVSDATETIMES, $REFDATE);
print 'lastest measurement datetime before or on first exposure date = ' $LATESTDATETIMEBEFOREFIRSTEXPOSURE;
$VSDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemGroup
```

Just for good reference, it is:

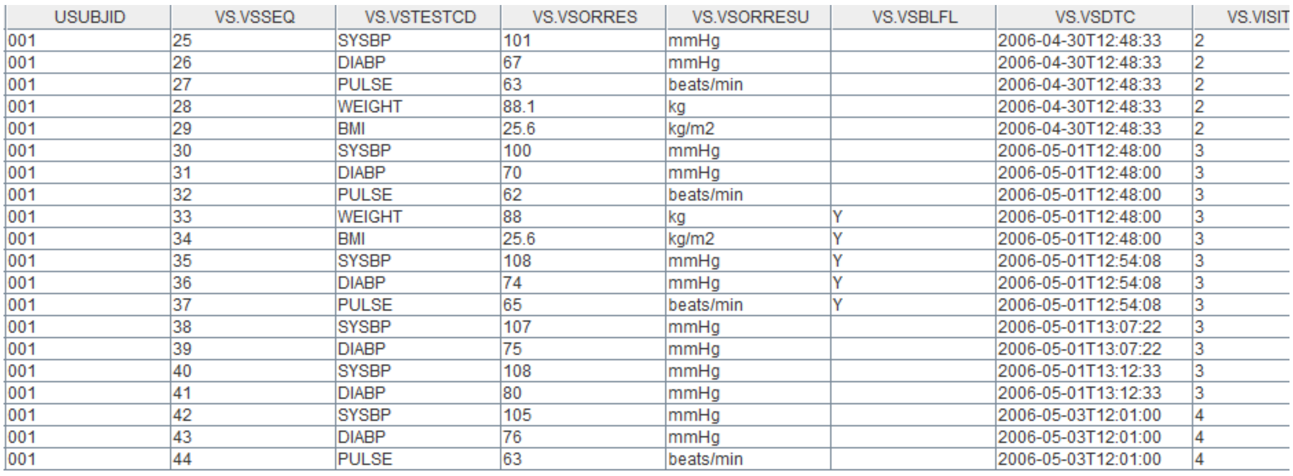
$VSDTC =
xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupData[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.VSDATETIM']/@Value);

When the last datetime of exposure (for current subject and test) is equal to our datetime (from VSDTC), then we need to set the baseline flag to "Y", in all other cases to null. So:



```
$REFDATE = $RFXSTDTC;
$LATESTDATETIMEBEFOREFIRSTEXPOSURE = latestorequaldatetimebefore($ALLVSDATETIMES, $REFDATE);
print 'lastest measurement datetime before or on first exposure date = ' $LATESTDATETIMEBEFOI
$VSDTC = xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/Item(
# when the visit date equals the last measurement date before first exposure
# the VSBLFL flag is set to "Y", in all other cases it is null
if($VSDTC == $LATESTDATETIMEBEFOREFIRSTEXPOSURE) {
     $VS.VSBLFL = 'Y';
} else {
     $VS.VSBLFL = '';
}
```

The final result when executing the script on the clinical data is:



USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSORRES	VS.VSORRESU	VS.VSBLFL	VS.VSDTC	VS.VISIT
001	25	SYSBP	101	mmHg		2006-04-30T12:48:33	2
001	26	DIABP	67	mmHg		2006-04-30T12:48:33	2
001	27	PULSE	63	beats/min		2006-04-30T12:48:33	2
001	28	WEIGHT	88.1	kg		2006-04-30T12:48:33	2
001	29	BMI	25.6	kg/m2		2006-04-30T12:48:33	2
001	30	SYSBP	100	mmHg		2006-05-01T12:48:00	3
001	31	DIABP	70	mmHg		2006-05-01T12:48:00	3
001	32	PULSE	62	beats/min		2006-05-01T12:48:00	3
001	33	WEIGHT	88	kg	Y	2006-05-01T12:48:00	3
001	34	BMI	25.6	kg/m2	Y	2006-05-01T12:48:00	3
001	35	SYSBP	108	mmHg	Y	2006-05-01T12:54:08	3
001	36	DIABP	74	mmHg	Y	2006-05-01T12:54:08	3
001	37	PULSE	65	beats/min	Y	2006-05-01T12:54:08	3
001	38	SYSBP	107	mmHg		2006-05-01T13:07:22	3
001	39	DIABP	75	mmHg		2006-05-01T13:07:22	3
001	40	SYSBP	108	mmHg		2006-05-01T13:12:33	3
001	41	DIABP	80	mmHg		2006-05-01T13:12:33	3
001	42	SYSBP	105	mmHg		2006-05-03T12:01:00	4
001	43	DIABP	76	mmHg		2006-05-03T12:01:00	4
001	44	PULSE	63	beats/min		2006-05-03T12:01:00	4

(the VS.VSDTC column has been moved to the left for better clarity).
The first datetime of exposure for subject 001 was 2006-05-01T12:57:04 so at 12:57 on the first of May 2006. As one sees, for each of the unique tests SYSBP, DIABP, WEIGHT PULSE and BMI, a single baseline value has been generated, which all (in this case) are on the same date of first exposure, but before the time of first exposure. All other measurements after 12:57 have not been marked with the baseline flag.

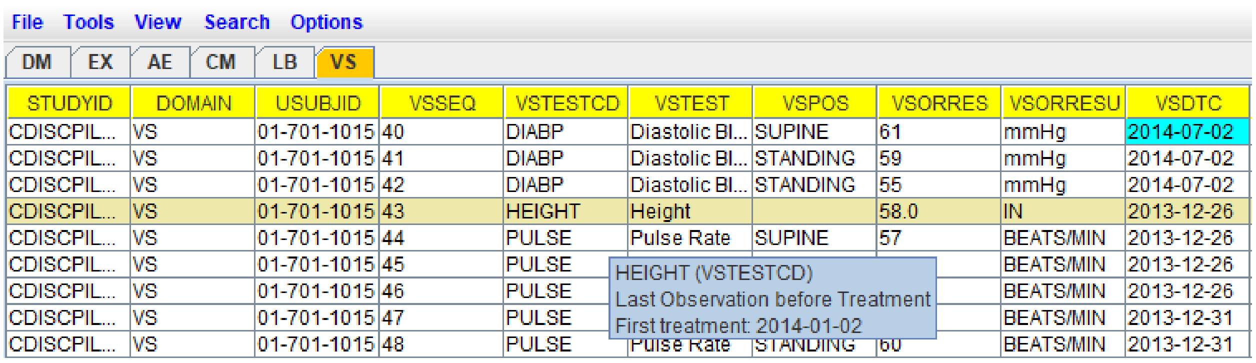
For good reference, here is the whole script again, completed with comments for better readability:

```
# The OID of the current test
$CUROID =
xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupD
ata[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.SYSBP' or @ItemOID='IT.DIABP' or
@ItemOID='IT.PULSE' or @ItemOID='IT.WT' or @ItemOID='IT.BMI']/@ItemOID);
print '$CUROID = ' $CUROID;
# all vital signs collection datetimes for the current test
$ALLVSDATETIMES =
xpath(//SubjectData[@SubjectKey=$USUBJID]/StudyEventData[@StudyEventOID='SE.VISITA']/FormDat
a[@FormOID='FORM.VS']/ItemGroupData[@ItemGroupOID='IG.VS'][ItemData[@ItemOID=$CUROID
and @Value]]/ItemData[@ItemOID='IT.VSDATETIM']/@Value);
# Just for testing: the number of datetimes for the current test
$COUNT = count($ALLVSDATETIMES);
print 'number of vital sign dates = ' $COUNT;
print 'vital sign dates = ' $ALLVSDATETIMES;
# the study start reference date (global variable)
$REFDATE = $RFXSTDTC;
# the latest datetime before first exposure, using the function latestorequaldatetimebefore()
$LATESTDATETIMEBEFOREFIRSTEXPOSURE = latestorequaldatetimebefore($ALLVSDATETIMES,
$REFDATE);
print 'lastest measurement datetime before or on first exposure date = '
$LATESTDATETIMEBEFOREFIRSTEXPOSURE;
# The datetime of collection
$VSDTC =
xpath(/StudyEventData[@StudyEventOID='SE.VISITA']/FormData[@FormOID='FORM.VS']/ItemGroupD
ata[@ItemGroupOID='IG.VS']/ItemData[@ItemOID='IT.VSDATETIM']/@Value);
# when the visit date equals the last measurement date before first exposure
# the VSBLFL flag is set to "Y", in all other cases it is null
if($VSDTC == $LATESTDATETIMEBEFOREFIRSTEXPOSURE) {
        $VS.VSBLFL = 'Y';
} else {
        $VS.VSBLFL = '';
}
```

Using -LOBXFL (as of SDTM-IG 3.3)

SDTM-IG 3.3 introduced a new variable "-LOBXFL", "Last Observation Before Exposure Flag", as reviewers wanted to have a consistent way of having a baseline flag, which is not guaranteed by the -BLFL variable.
Essentially, SDTM should not have baseline flags at all, as SDTM is meant to be "collected data only", but we have unfortunately seen that CDISC has given in to requests of reviewers even when that violates the first principles of SDTM themselves. Sadly so, also for -LOBXFL.
Also, it once again demonstrates that the tools the reviewers are using are not capable of deriving such information. Most of them still use either the "SASViewer" or the "SAS Universal Viewer", which have no intrinsic SDTM knowledge, or they load the SAS Transport files into SAS, not knowing how to program the derivation of a "last observation before exposure".

Modern review tools that are not based on SAS Transport 5, like the "Smart Submission Dataset Viewer", can assign "last observation before exposure" records fully automatically:



STUDYID	DOMAIN	USUBJID	VSSEQ	VSTESTCD	VSTEST	VSPOS	VSORRES	VSORRESU	VSDTC
CDISCPIL...	VS	01-701-1015	40	DIABP	Diastolic Bl...	SUPINE	61	mmHg	2014-07-02
CDISCPIL...	VS	01-701-1015	41	DIABP	Diastolic Bl...	STANDING	59	mmHg	2014-07-02
CDISCPIL...	VS	01-701-1015	42	DIABP	Diastolic Bl...	STANDING	55	mmHg	2014-07-02
CDISCPIL...	VS	01-701-1015	43	HEIGHT	Height		58.0	IN	2013-12-26
CDISCPIL...	VS	01-701-1015	44	PULSE	Pulse Rate	SUPINE	57	BEATS/MIN	2013-12-26
CDISCPIL...	VS	01-701-1015	45	PULSE				BEATS/MIN	2013-12-26
CDISCPIL...	VS	01-701-1015	46	PULSE				BEATS/MIN	2013-12-26
CDISCPIL...	VS	01-701-1015	47	PULSE				BEATS/MIN	2013-12-31
CDISCPIL...	VS	01-701-1015	48	PULSE	Pulse Rate	STANDING	60	BEATS/MIN	2013-12-31

HEIGHT (VSTESTCD)
Last Observation before Treatment
First treatment: 2014-01-02

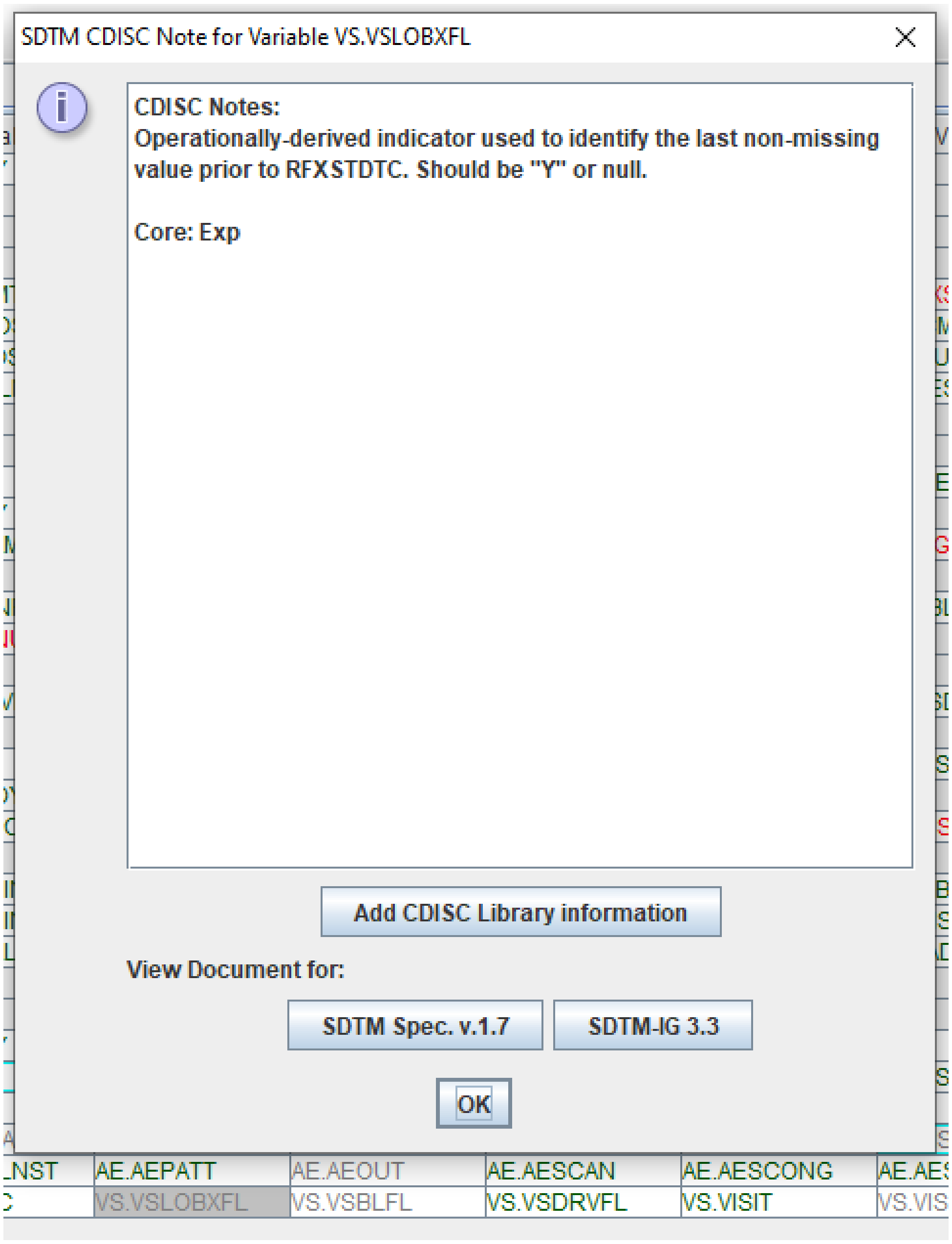
Unfortunately, also as the regulatory authorities stick to outdated SAS Transport, they do not use such modern review tools, so the leave the burden of assigning the "last observation before exposure" to the sponsor[16].

The generation of -LBOXFL however often (but now always) requires a post-SDTM processing step, i.e. generate all the SDTM datasets without the "last observation before exposure" flags, and then do a post-processing step in which the datasets are "corrected", and the baseline flags calculated from the generated datasets.

Such a post-processing step is of course completely against the spirit of SDTM ("source data - no derived variables"), but this is however a sad truth.

First, let us have a look at the "CDISC Notes" for VSLOBXFL. These can be obtained using "CTRL-H" or using the menu "View - SDTM CDISC Notes", leading to:

[16] Essentially, this is the "world upside down": reviewers should not trust such important assignments to be made by the sponsor. Even the smaller error can lead to fully incorrect review conclusions!

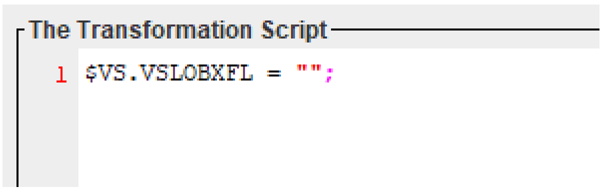

SDTM CDISC Note for Variable VS.VSLOBXFL

CDISC Notes:
Operationally-derived indicator used to identify the last non-missing value prior to RFXSTDTC. Should be "Y" or null.

Core: Exp

Although we will need a "post generation derivation", we must already set a value, which is the default value, i.e. the empty value[17].

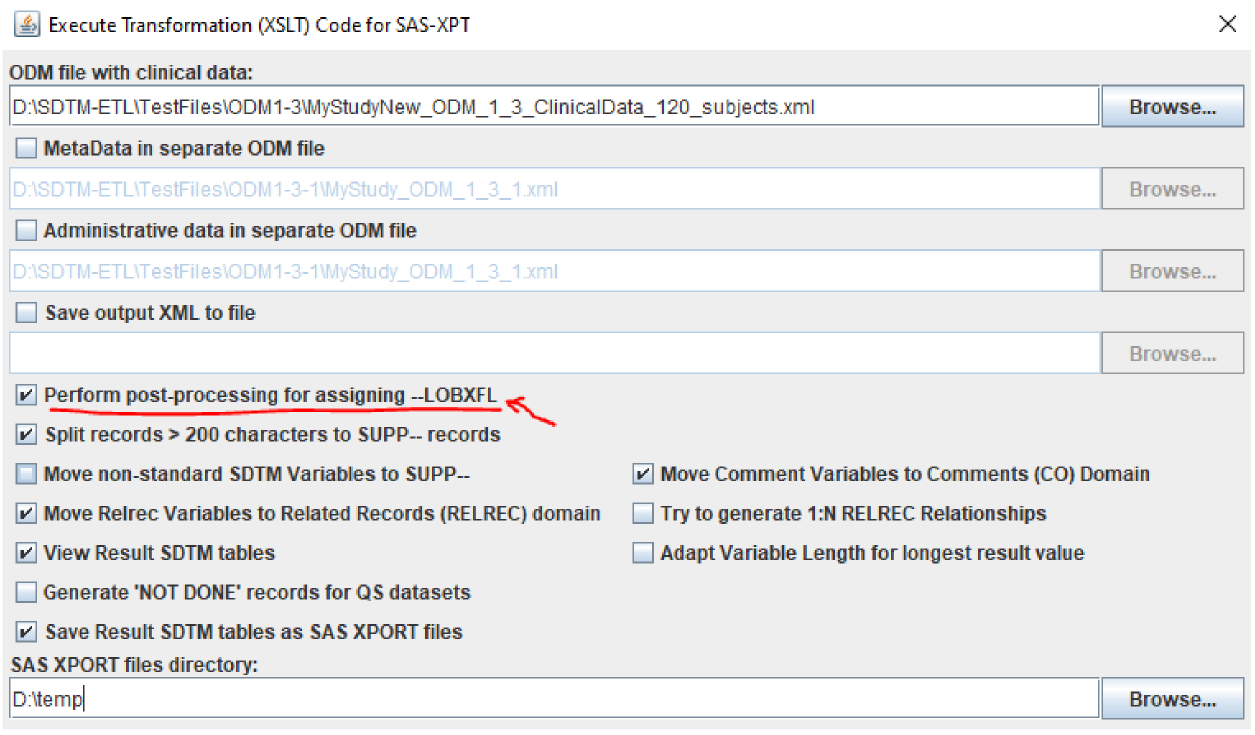
So, after double-clicking the cell for VS.VSLOBXFL, we simply add:


The Transformation Script

```
$VS.VSLOBXFL = "";
```

After having completed the mappings for all necessary variables, and then performing execution on the file with clinical data, we must also check the checkbox "Perform post-processing for assigning --LOBXFL":

[17] The designation "null" in the CDISC notes is misleading: there is no " explicit null" in SDTM. In the SAS Transport files, the value is simply filled with one or more blanks.



ODM file with clinical data: D:\SDTM-ETL\TestFiles\ODM1-3\MyStudyNew_ODM_1_3_ClinicalData_120_subjects.xml

[ ] MetaData in separate ODM file
D:\SDTM-ETL\TestFiles\ODM1-3-1\MyStudy_ODM_1_3_1.xml

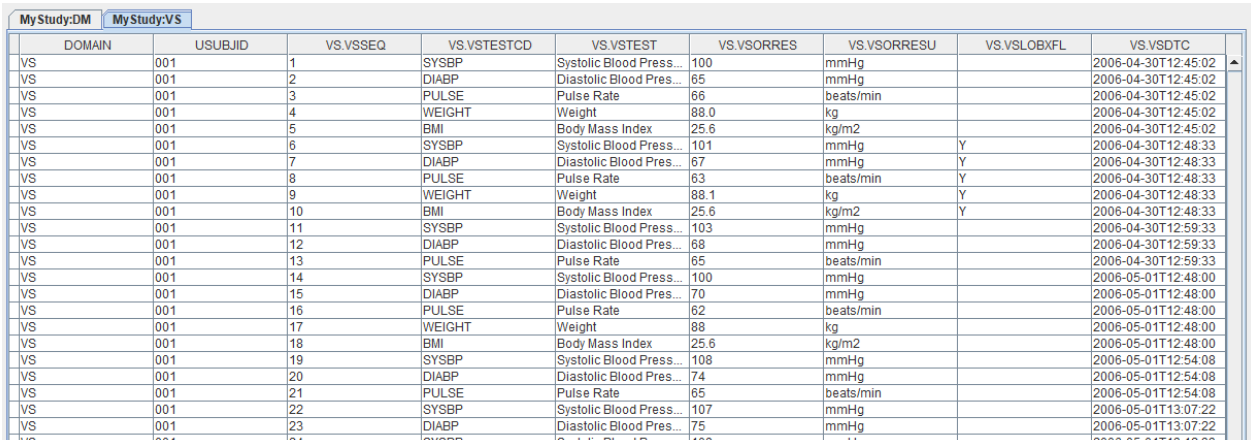
[ ] Administrative data in separate ODM file
D:\SDTM-ETL\TestFiles\ODM1-3-1\MyStudy_ODM_1_3_1.xml

[ ] Save output XML to file

[✓] Perform post-processing for assigning --LOBXFL
[✓] 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
[✓] Save Result SDTM tables as SAS XPORT files

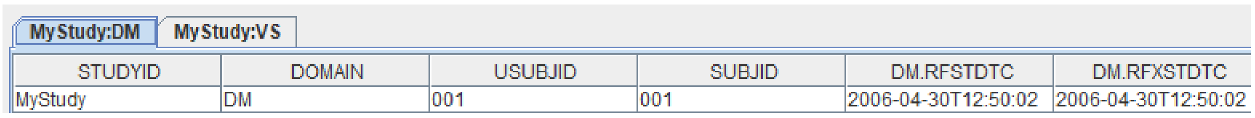
SAS XPORT files directory: D:\temp

and then starting the execution using the button "Execute Transformation on Clinical Data", leading to the result:



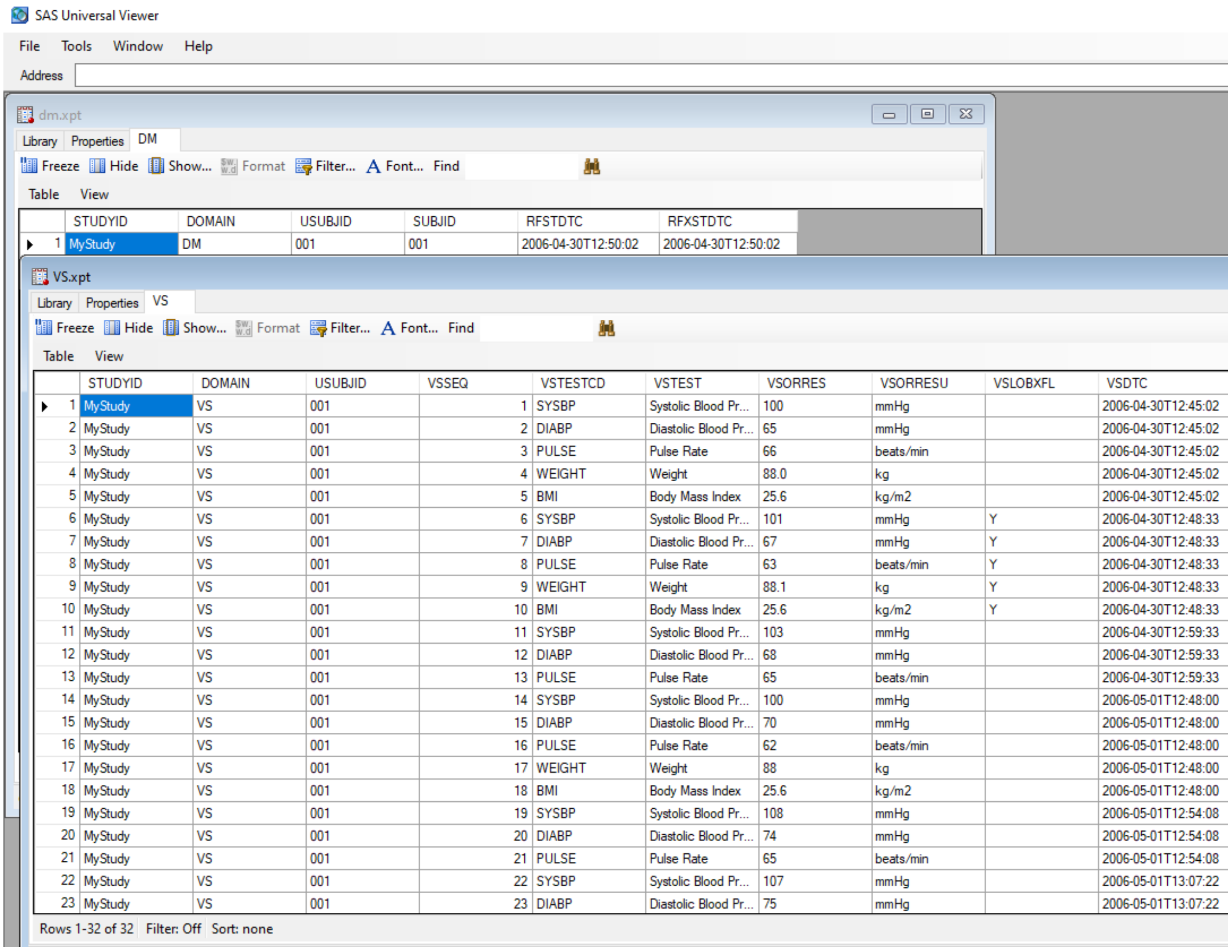
DOMAIN	USUBJID	VS.VSSEQ	VS.VSTESTCD	VS.VSTEST	VS.VSORRES	VS.VSORRESU	VS.VSLOBXFL	VS.VSDTC
VS	001	1	SYSBP	Systolic Blood Press...	100	mmHg		2006-04-30T12:45:02
VS	001	2	DIABP	Diastolic Blood Pres...	65	mmHg		2006-04-30T12:45:02
VS	001	3	PULSE	Pulse Rate	66	beats/min		2006-04-30T12:45:02
VS	001	4	WEIGHT	Weight	88.0	kg		2006-04-30T12:45:02
VS	001	5	BMI	Body Mass Index	25.6	kg/m2		2006-04-30T12:45:02
VS	001	6	SYSBP	Systolic Blood Press...	101	mmHg	Y	2006-04-30T12:48:33
VS	001	7	DIABP	Diastolic Blood Pres...	67	mmHg	Y	2006-04-30T12:48:33
VS	001	8	PULSE	Pulse Rate	63	beats/min	Y	2006-04-30T12:48:33
VS	001	9	WEIGHT	Weight	88.1	kg	Y	2006-04-30T12:48:33
VS	001	10	BMI	Body Mass Index	25.6	kg/m2	Y	2006-04-30T12:48:33
VS	001	11	SYSBP	Systolic Blood Press...	103	mmHg		2006-04-30T12:59:33
VS	001	12	DIABP	Diastolic Blood Pres...	68	mmHg		2006-04-30T12:59:33
VS	001	13	PULSE	Pulse Rate	65	beats/min		2006-04-30T12:59:33
VS	001	14	SYSBP	Systolic Blood Press...	100	mmHg		2006-05-01T12:48:00
VS	001	15	DIABP	Diastolic Blood Pres...	70	mmHg		2006-05-01T12:48:00
VS	001	16	PULSE	Pulse Rate	62	beats/min		2006-05-01T12:48:00
VS	001	17	WEIGHT	Weight	88	kg		2006-05-01T12:48:00
VS	001	18	BMI	Body Mass Index	25.6	kg/m2		2006-05-01T12:48:00
VS	001	19	SYSBP	Systolic Blood Press...	108	mmHg		2006-05-01T12:54:08
VS	001	20	DIABP	Diastolic Blood Pres...	74	mmHg		2006-05-01T12:54:08
VS	001	21	PULSE	Pulse Rate	65	beats/min		2006-05-01T12:54:08
VS	001	22	SYSBP	Systolic Blood Press...	107	mmHg		2006-05-01T13:07:22
VS	001	23	DIABP	Diastolic Blood Pres...	75	mmHg		2006-05-01T13:07:22

showing the 5 measurements done at 12:48 on date 2006-04-30 as the "last observation before exposure", as the first exposure was at 12:50 of the same day, as can be seen from RFXSTDTC in DM.



STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFXSTDTC
MyStudy	DM	001	001	2006-04-30T12:50:02	2006-04-30T12:50:02

Or when inspecting the generated SAS Transport files using the "SAS Universal Viewer":



	STUDYID	DOMAIN	USUBJID	SUBJID	RFSTDTC	RFXSTDTC
1	MyStudy	DM	001	001	2006-04-30T12:50:02	2006-04-30T12:50:02

	STUDYID	DOMAIN	USUBJID	VSSEQ	VSTESTCD	VSTEST	VSORRES	VSORRESU	VSLOBXFL	VSDTC
1	MyStudy	VS	001	1	SYSBP	Systolic Blood Pr...	100	mmHg		2006-04-30T12:45:02
2	MyStudy	VS	001	2	DIABP	Diastolic Blood Pr...	65	mmHg		2006-04-30T12:45:02
3	MyStudy	VS	001	3	PULSE	Pulse Rate	66	beats/min		2006-04-30T12:45:02
4	MyStudy	VS	001	4	WEIGHT	Weight	88.0	kg		2006-04-30T12:45:02
5	MyStudy	VS	001	5	BMI	Body Mass Index	25.6	kg/m2		2006-04-30T12:45:02
6	MyStudy	VS	001	6	SYSBP	Systolic Blood Pr...	101	mmHg	Y	2006-04-30T12:48:33
7	MyStudy	VS	001	7	DIABP	Diastolic Blood Pr...	67	mmHg	Y	2006-04-30T12:48:33
8	MyStudy	VS	001	8	PULSE	Pulse Rate	63	beats/min	Y	2006-04-30T12:48:33
9	MyStudy	VS	001	9	WEIGHT	Weight	88.1	kg	Y	2006-04-30T12:48:33
10	MyStudy	VS	001	10	BMI	Body Mass Index	25.6	kg/m2	Y	2006-04-30T12:48:33
11	MyStudy	VS	001	11	SYSBP	Systolic Blood Pr...	103	mmHg		2006-04-30T12:59:33
12	MyStudy	VS	001	12	DIABP	Diastolic Blood Pr...	68	mmHg		2006-04-30T12:59:33
13	MyStudy	VS	001	13	PULSE	Pulse Rate	65	beats/min		2006-04-30T12:59:33
14	MyStudy	VS	001	14	SYSBP	Systolic Blood Pr...	100	mmHg		2006-05-01T12:48:00
15	MyStudy	VS	001	15	DIABP	Diastolic Blood Pr...	70	mmHg		2006-05-01T12:48:00
16	MyStudy	VS	001	16	PULSE	Pulse Rate	62	beats/min		2006-05-01T12:48:00
17	MyStudy	VS	001	17	WEIGHT	Weight	88	kg		2006-05-01T12:48:00
18	MyStudy	VS	001	18	BMI	Body Mass Index	25.6	kg/m2		2006-05-01T12:48:00
19	MyStudy	VS	001	19	SYSBP	Systolic Blood Pr...	108	mmHg		2006-05-01T12:54:08
20	MyStudy	VS	001	20	DIABP	Diastolic Blood Pr...	74	mmHg		2006-05-01T12:54:08
21	MyStudy	VS	001	21	PULSE	Pulse Rate	65	beats/min		2006-05-01T12:54:08
22	MyStudy	VS	001	22	SYSBP	Systolic Blood Pr...	107	mmHg		2006-05-01T13:07:22
23	MyStudy	VS	001	23	DIABP	Diastolic Blood Pr...	75	mmHg		2006-05-01T13:07:22

Rows 1-32 of 32 Filter: Off Sort: none

This concludes the tutorial about baseline flags.