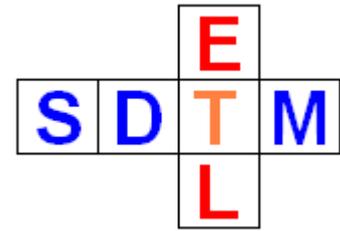


SDTM-ETL 5.0 User Manual and Tutorial

Author: Jozef Aerts, XML4Pharma

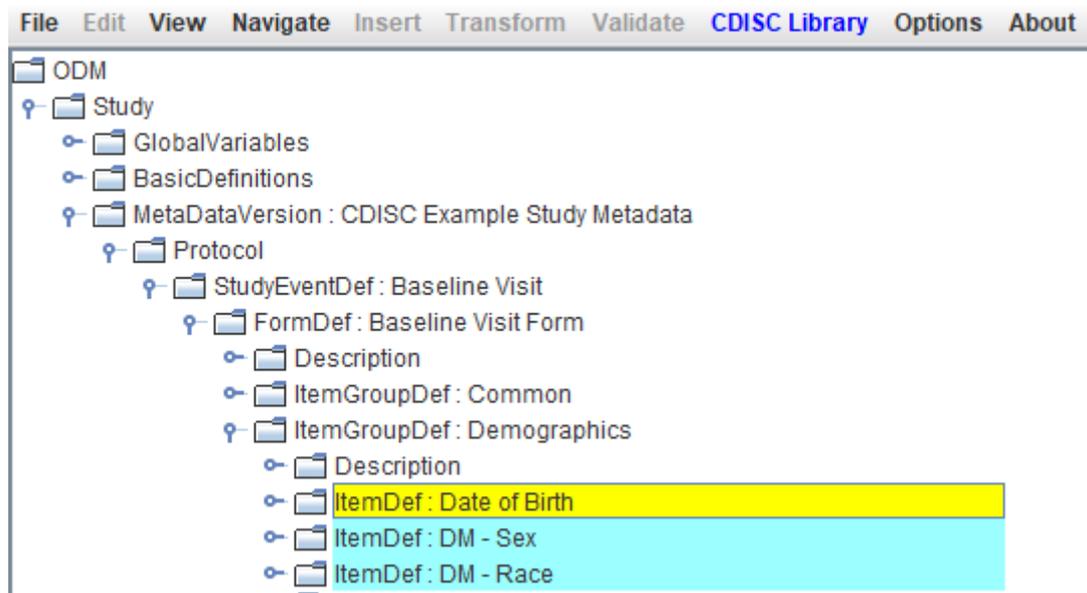
Last update: 2025-02-03



Loading an SDTM or SEND template – mappings for DM

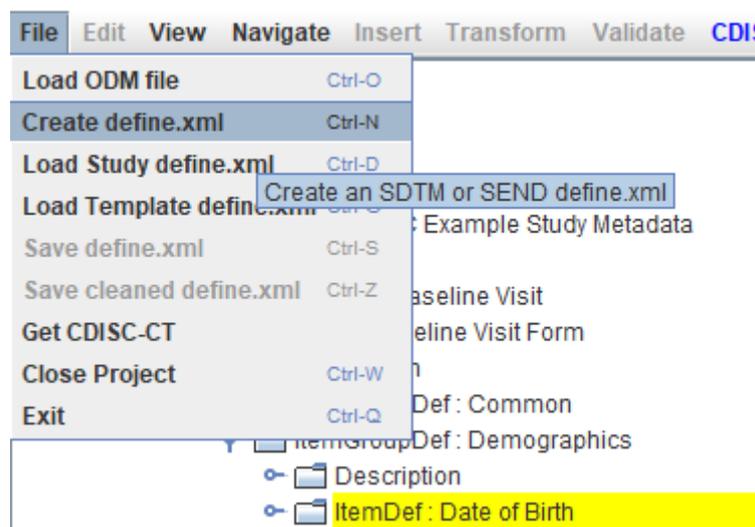
After having loaded and inspected a CDISC ODM file with the study design, we can start working on the mapping with SDTM or SEND.

At the left side of the screen, the tree view of the clinical study design is already shown, in this case of the CES study¹:



the right side of the screen being still empty.

In order to start mapping to SDTM (or SEND) a template which is implementing the SDTM-IG or SEND-IG needs to be loaded. In order to do so, use the menu "File – Create define.xml":



¹This is a study design originally developed by Dave Ibersen-Hurst for demo purposes, and later extended by others.

The reason it speaks about a define.xml is that all our mappings, and any other metadata about our SDTM or SEND will be stored in a define.xml structure, which is kept in sync with everything that we do, so that at the end, we will be able to generate a define.xml file² for our study with just a few mouse clicks.

A dialog is then presented:

SDTM/SEND Version

Do you want to work with the SDTM-IG or SEND-IG CDISC Standard?

SDTM-IG Standard SEND-IG Standard

Which version of the Standard would you like to work with?

SDTM-IG 3.1.2
 SDTM-IG 3.1.3
 SDTM-IG 3.2
 SDTM-IG 3.3
 SDTM-IG 3.4
 SDTM-IG MD.1.0
 SDTM-IG MD.1.1
 SDTM-IG AP.1.0

Define.xml version:

define.xml 2.1 define.xml 2.0

Controlled Terminology Version:

2023-06-30
2023-09-29
2023-12-15
2024-03-29
2024-09-27

Generate a 'Yes-Only' sub-codelist and assign it to all --FL variables (recommended)

Generate a STENRF sub-codelist and assign it to all --STRTP and --ENRTPT variables (recommended)

OK Cancel

The user can choose between different SDTM versions and different SEND versions³. In case there are different versions of controlled terminology for the given standard, the versions are presented and the user can decide which version of the CDISC controlled terminology should be

²For any SDTM or SEND submission, the FDA requires a define.xml file to be submitted together with the actual data sets, containing the metadata for the submission files.

³ Including SENDIG-DART 1.1

loaded⁴. Remark that each time new Controlled Terminologies are published by CDISC, these can be downloaded from our SDTM-ETL website and added to the folder "CDISC_CT", making them immediately available without the need of a software update.

Also, one can choose between using Define-XML 2.0 and 2.1 for keeping the metadata. Remark that Define-XML 1.0 is not accepted by the regulatory authorities, and is not supported anymore as of SDTM-ETL version 5.0. For new projects, it is highly recommended to use Define-XML 2.1, as it has a lot of new features that make it easier for regulatory reviewers to review the study.

Suppose we want to use Define-XML 2.1 with SDTM-IG 3.3, together with CDISC-CT version 2024-09-27⁵.

One can also come to this dialog using the keyboard combination CTRL-N.

At the bottom of the dialog, one also finds two checkboxes:

The screenshot shows a configuration dialog with the following elements:

- Radio buttons for SDTM-IG versions: SDTM-IG 3.3 (selected), SDTM-IG 3.4, SDTM-IG MD.1.0, SDTM-IG MD.1.1, SDTM-IG AP.1.0.
- Define.xml version: define.xml 2.1 (selected), define.xml 2.0.
- Controlled Terminology Version: A list box containing 2023-06-30, 2023-09-29, 2023-12-15, 2024-03-29, and 2024-09-27 (selected).
- Two checkboxes with red arrows pointing to them:
 - Generate a 'Yes-Only' sub-codelist and assign it to all --FL variables (recommended)
 - Generate a STENRF sub-codelist and assign it to all --STRTP and --ENRTPT variables (recommended)
- OK and Cancel buttons at the bottom.

When checked (highly recommended), the software will immediately and automatically generate subset codelists, one for "Yes only" and one for the -STRTP and -ENRTPT variables and assign them to the variables where this useful and necessary.

This is unfortunately needed due to the stubborn refusal of the CDISC Controlled Terminology team to publish subset codelists for these.

⁴Later we will learn how to load additional codelists when necessary.

⁵ The files with CDISC-CT are kept in the directory "CDISC-CT". In case new controlled terminology was published by CDISC, you can obtain the file for the SDTM-ETL website. If you then copy it in the "CDISC_CT" directory, it will automatically be added to the list of available versions.

After clicking "OK", the system now starts loading the template, which can take a few minutes.

One can then see that the right side of the screen is now filled with an SDTM table, containing a row for each SDTM domain in the SDTM-IG, and a cell for each SDTM variable, with the first cell containing the SDTM domain name (DM, TE, ...):

The screenshot shows the CDISC Library application. On the left, a tree view lists various SDTM domains and variables, such as 'DM', 'TE', 'AE', 'DS', 'DV', 'MH', 'DA', 'DD', 'EG', 'IE', 'IS', 'LB', 'MS', 'MI', 'CV', 'MK', 'NV', 'OE', 'RE', 'UR', 'PC', 'PP', 'PE', 'FT', 'QS', 'SC', 'SS', 'TU', 'TR', 'VS', 'FA', 'SR', 'TA', 'TE', 'TV', and 'TM'. On the right, a large grid displays the mapping of these domains to their respective variables. The grid has columns for 'Domain' and 'Variable'. The variables are color-coded: red (e.g., AE.AEMODIFY, CE.CEDECOD, DS.define.xml information:), blue (e.g., CE.AELLT, CE.AEL), and green (e.g., CE.CECAT, CE.CE). The grid is titled 'Domains (ItemGroups)'.

The division line between the two sides of the screen can be dragged, in order to see more or less of each side of the screen.

It probably has already been noticed that some of the SDTM variables are colored red, some blue and other ones green. The red ones are the ones that are designated as being "required" in the SDTM-IG, the blue ones those that are designated as being "expected", and the green ones those that are "permissible".

In order to obtain more information about a specific variable, just hover the mouse over a cell, e.g.:

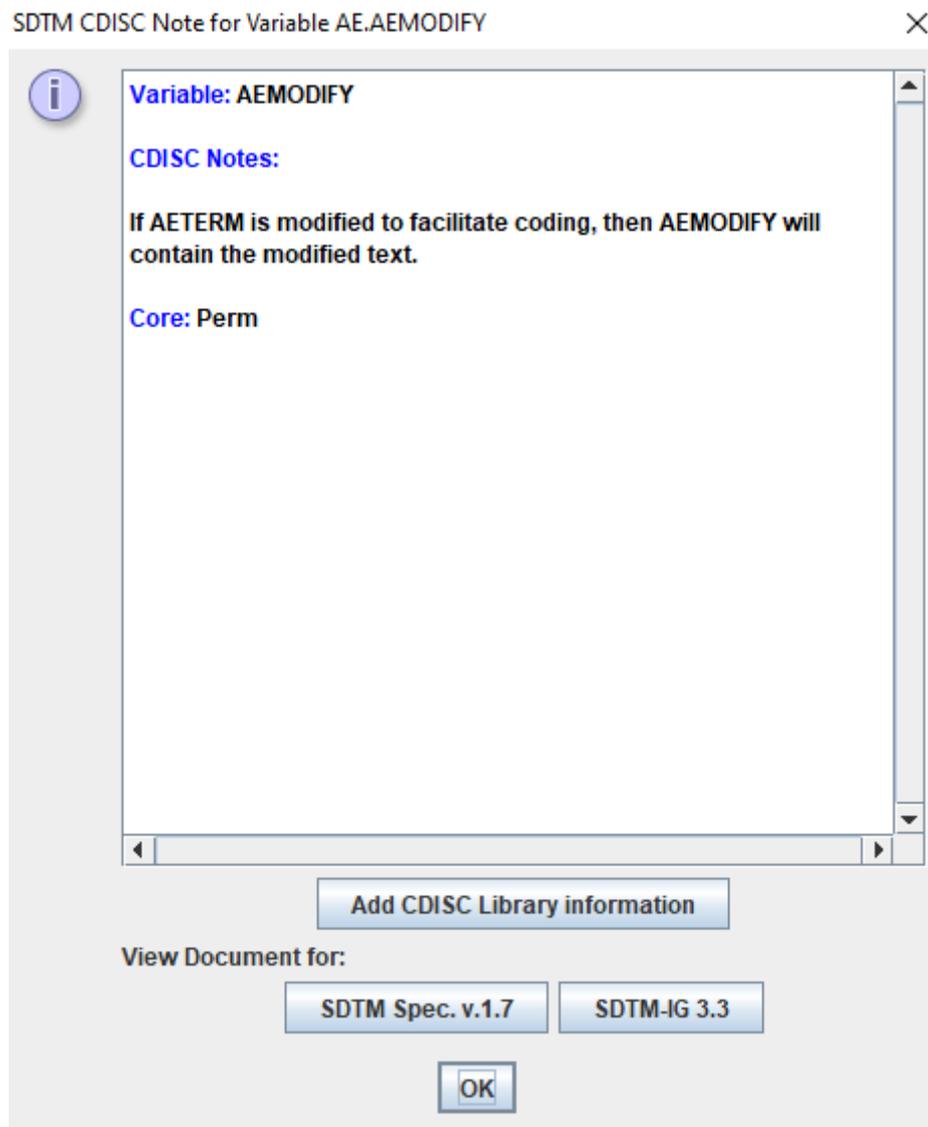
The tooltip window displays the following information for the variable AE.AEMODIFY:

- SDTM Name: AE.AEMODIFY
- OID: AE.AEMODIFY
- Mandatory: No
- OrderNumber: 9
- Role: Synonym Qualifier
- Data type: text
- Length: 80
- Description: Modified Reported Term

One also sees that currently the "maximal length" for this variable has been set to 80. Later it will be demonstrated how this value can be adapted to a more suitable value in agreement with what is in the collected data.

In order to get real in-depth information about a specific SDTM variable, select the cell, and then

use "View – SDTM CDISC Notes" or use CTRL-H. A new window is then displayed, e.g. for AEMODIFY:



One can then open the corresponding section of either the standard specification or implementation guide (SDTM-IG by either clicking the button "SDTM Spec. v.1.7" or "SDTM-IG v.3.3", as the latter documents come with the distribution⁶.

If one also wants to have the information from the "CDISC Library" being displayed, click the button "Add CDISC Library information". Remark that this requires that you have a "CDISC Library API key", which needs to be stored in the "properties.dat" file. This leads to e.g.:

⁶One only need to set the path to the favorite PDF viewer in the "properties.dat" file, as explained in the SDTM-ETL installation guide.

Variable: AEMODIFY

CDISC Notes:

If AETERM is modified to facilitate coding, then AEMODIFY will contain the modified text.

Core: Perm

CDISC Library information:

label: Modified Reported Term
description:
 If AETERM is modified to facilitate coding, then AEMODIFY will contain the modified text.
role: Synonym Qualifier
core: Perm
simpleDatatype: Char

Add CDISC Library information

Later we will also learn how to add additional standard variables, and how to add "non-standard" variables that later typically go into "SUPPQUAL".

Now have a look at the first cell in a row, representing the SDTM domain. Also here, hovering the mouse displays some more information, e.g.:

MO	STUDYID	DOMAIN	USUBJID
MI	STUDYID	DOMAIN	USUBJID
MO	STUDYID	DOMAIN	USUBJID
CV	STUDYID	DOMAIN	USUBJID
MK	define.xml information:		
NV	OID: MO		
OE	Name: MO		
RP	Domain: MO		
RE	SAS Dataset Name: MO		
UR	Purpose: Tabulation		
PC	Repeating: Yes		
PP	Is Reference Data: No		
PE	Label: Morphology		
FT	Class: FINDINGS		
QS	Structure: One record per MO.MOTESTCD per USUBJID		
RS	Archive Location ID: Location.MO		
SC	STUDYID	DOMAIN	USUBJID
SS	STUDYID	DOMAIN	USUBJID

The label for this domain is "Morphology", and it belongs to the "Findings" class. The other information will be explained later when it is explained how the domain properties can be edited.

SEND support

SDTM-ETL is very often used to generate mappings to SEND for non-clinical (non-human) studies and generating the SEND datasets. In these cases, the ODM files needed usually do not come from EDC systems, but have been generated using the "[ODMGenerator](#)" software, which allows to generate ODM files (both metadata and data) from almost any source.

When selecting "SEND" as the standard, the following dialog is displayed:

SDTM/SEND Version

Do you want to work with the SDTM-IG or SEND-IG CDISC Standard?

SDTM-IG Standard SEND-IG Standard

Which version of the Standard would you like to work with?

SEND-IG 3.0

SEND-IG 3.1

SEND-IG 3.1.1

SEND-IG GENETOX.1.0

SEND-IG AR.1.0

SEND-IG DART.1.1

Define.xml version:

define.xml 2.1 define.xml 2.0

Controlled Terminology Version:

2023-06-30

2023-09-29

2023-12-15

2024-03-29

2024-09-27

allowing to select a SENDIG-version and a SEND Controlled Terminology (SEND-CT) version. Remark that the template for SEND-IG GENETOX 1.0 includes SEND-IG 3.1.1, as GENETOX is essentially 3.1.1 with addition of the GV (Genetic Toxicology - In Vivo) domain.

For example, if we select "SEND-IG GENETOX 1.0", "define.xml 2.1", as well as "2024-09-27" for the SEND-CT, the result on the right side in the main window is:

Domains (ItemGroups)								
Domain	Variable	Variable	Variable	Variable	Variable	Variable	Variable	Variable
DS	STUDYID	DOMAIN	USUBJID	DS.DSSEQ	DS.DSTERM	DS.DSDECOD	DS.DSUSCHFL	DS.VISITDY
BG	STUDYID	DOMAIN	USUBJID	BG.BGSEQ	BG.BGTESTCD	BG.BGTEST	BG.BGORRES	BG.BGORRE
BW	STUDYID	DOMAIN	USUBJID	BW.BWSEQ	BW.BWTESTCD	BW.BWTEST	BW.BWORRES	BW.BWORR
CL	STUDYID	DOMAIN	USUBJID	POOLID	CL.FOCID	CL.CLSEQ	CL.CLGRPID	CL.CLSPID
CV	STUDYID	DOMAIN	USUBJID	CV.CVSEQ	CV.CVGRPID	CV.CVSPID	CV.CVTESTCD	CV.CVTEST
DD	STUDYID	DOMAIN	USUBJID	DD.DDSEQ	DD.DDTESTCD	DD.DDTEST	DD.DDORRES	DD.DDSTRE
EG	STUDYID	DOMAIN	USUBJID	EG.EGSEQ	EG.EGGRPID	EG.EGREFID	EG.EGSPID	EG.EGTEST
FW	STUDYID	DOMAIN	USUBJID	POOLID	FW.FWSEQ	FW.FWGRPID	FW.FWTESTCD	FW.FWTEST
GV	STUDYID	DOMAIN	USUBJID	GV.GVSEQ	GV.GVGRPID	GV.GVREFID	GV.GVSPID	GV.GVTEST
LB	STUDYID	DOMAIN	USUBJID	POOLID	LB.LBSEQ	LB.LBGRPID	LB.LBREFID	LB.LBSPID
MA	define.xml information:			MA.FOCID	MA.MASEQ	MA.MAGRPID	MA.MAREFID	MA.MASPID
MI	OID: GV			MI.FOCID	MI.MISEQ	MI.MIGRPID	MI.MIREFID	MI.MISPID
OM	Name: GV			OM.OMSEQ	OM.OMTESTCD	OM.OMTEST	OM.OMORRES	OM.OMORRI
PC	Domain: GV			POOLID	PC.PCSEQ	PC.PCGRPID	PC.PCREFID	PC.PCSPID
PM	SAS Dataset Name: GV			PM.PMSEQ	PM.PMGRPID	PM.PMSPID	PM.PMTESTCD	PM.PMTEST
PP	Purpose: Tabulation			POOLID	PP.PPSEQ	PP.PPGRPID	PP.PPTESTCD	PP.PPTEST
RE	Repeating: Yes			RE.RESEQ	RE.REGRPID	RE.RESPID	RE.RETESTCD	RE.RETEST
SC	Is Reference Data: No			SC.SCSEQ	SC.SCGRPID	SC.SCTESTCD	SC.SCTEST	SC.SCORRE
TF	Label: Genetic Toxicology - In Vivo			TF.TFSEQ	TF.TFGRPID	TF.TFREFID	TF.TFSPID	TF.TFTEST
VS	Class: FINDINGS			VS.VSSEQ	VS.VSGRPID	VS.VSSPID	VS.VSTESTCD	VS.VSTEST
EX	Structure: One record per GV.GVTESTCD per USUBJID			POOLID	EX.FOCID	EX.EXSEQ	EX.EXTRT	EX.EXDOSE
POOLDE	Archive Location ID: Location.GV							
RELREC	Standard-OID: STD.SENDIG.GENETOX.1.0			POOLID	IDVAR	IDVARVAL	RELTYPE	RELID
SUPQUAL	STUDYID	RDOMAIN	USUBJID	POOLID	IDVAR	IDVARVAL	QNAM	QLABEL
CO	STUDYID	DOMAIN	RDOMAIN	USUBJID	POOLID	CO.COSEQ	CO.IDVAR	CO.IDVARV
DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC	DM.RFXEND
SE	STUDYID	DOMAIN	USUBJID	SE.SESEQ	SE.ETCD	SE.ELEMENT	SE.SESTDTC	SE.SEENDT
TA	STUDYID	DOMAIN	TA.ARMCD	TA.ARM	TA.TAETORD	TA.ETCD	TA.ELEMENT	TA.TABRAN
TE	STUDYID	DOMAIN	TE.ETCD	TE.ELEMENT	TE.TESTRL	TE.TEENRL	TE.TEDUR	
TS	STUDYID	DOMAIN	TS.TSSEQ	TS.TSGRPID	TS.TSPARMCD	TS.TSPARM	TS.TSVAL	TS.TSVALNF
TX	STUDYID	DOMAIN	TX.SETCD	TX.SET	TX.TXSEQ	TX.TXPARMCD	TX.TXPARM	TX.TXVAL

Adding new SDTM/SEND versions

CDISC regularly (every 2-5 years) publishes new versions of SDTM or SEND and Implementation Guides (IGs) for them.

When such a new standard is published and one wants to use it, it is not necessary to obtain a new version of the SDTM-ETL software. As soon as a template is available (ask us), it simply can be added to the software distribution, and registered in the file "SDTM_SEND_standards.xml" which is part of the distribution. For example, when SENDIG-GENETOX was added, the template is registered as:

D:\SDTM-ETL_5_0\SDTM_SEND_standards.xml - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

SDTM_SEND_standards.xml

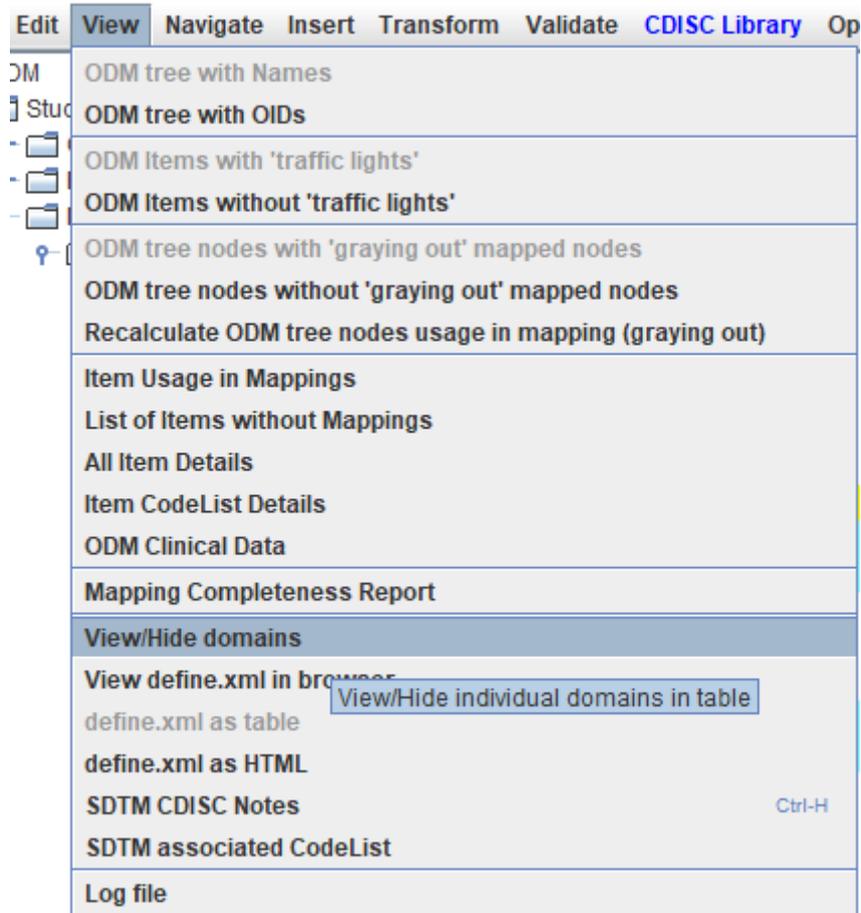
```

133
134 <!-- SEND implementations of define.xml 2.1 -->
135 <Standard Name="SEND-IG" Version="3.0" SDSVersion="1.2" DefineXMLVersion="2.1">
136   <TemplateFile xlink:href="define_2_1/define_template_SENDIG_3.0.xml"/>
137   <!--CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SEND_v3.0.xml"/-->
138   <CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SDS_v1.2.xml"/>
139   <ImplementationGuideFile xlink:href="standarddocs/SEND_Implementation_Guide_V3.0.pdf"/>
140 </Standard>
141 <Standard Name="SEND-IG" Version="3.1" SDSVersion="1.5" DefineXMLVersion="2.1">
142   <TemplateFile xlink:href="define_2_1/define_template_SENDIG_3.1.xml"/>
143   <!--CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SEND_v3.1.xml"/-->
144   <CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SDS_v1.5.xml"/>
145   <ImplementationGuideFile xlink:href="standarddocs/SEND_Implementation_Guide_V3.1.pdf"/>
146 </Standard>
147 <Standard Name="SEND-IG" Version="3.1.1" SDSVersion="1.5" DefineXMLVersion="2.1">
148   <TemplateFile xlink:href="define_2_1/define_template_SENDIG_3.1.1.xml"/>
149   <!-- SENDIG 3.1.1 and 3.1 are both based on SDTM model 1.5 -->
150   <CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SDS_v1.5.xml"/>
151   <ImplementationGuideFile xlink:href="standarddocs/SEND_Implementation_Guide_V3.1.1.pdf"/>
152 </Standard>
153 <Standard Name="SEND-IG" Version="GENETOX.1.0" SDSVersion="1.5" DefineXMLVersion="2.1">
154   <TemplateFile xlink:href="define_2_1/define_template_SENDIG_3.1.1_GENETOX.1.0.xml"/>
155   <!-- SENDIG 3.1.1 and 3.1 are both based on SDTM model 1.5 -->
156   <CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SDS_v1.5.xml"/>
157   <ImplementationGuideFile xlink:href="standarddocs/SEND_Implementation_Guide_V3.1.1.pdf"/>
158 </Standard>
159 <Standard Name="SEND-IG" Version="AR.1.0" SDSVersion="1.8" DefineXMLVersion="2.1">
160   <TemplateFile xlink:href="define_2_1/define_template_SENDIG_AR.1.0.xml"/>
161   <!--CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SEND_v3.1.xml"/-->
162   <CDISCNotesFile xlink:href="CDISC_Notes/CDISC_SDS_v1.8.xml"/>
163   <ImplementationGuideFile xlink:href="standarddocs/SENDIG-AR_v1.0-Final.pdf"/>
164 </Standard>
165 <Standard Name="SEND-IG" Version="DABP.1.1" SDSVersion="1.5" DefineXMLVersion="2.1">

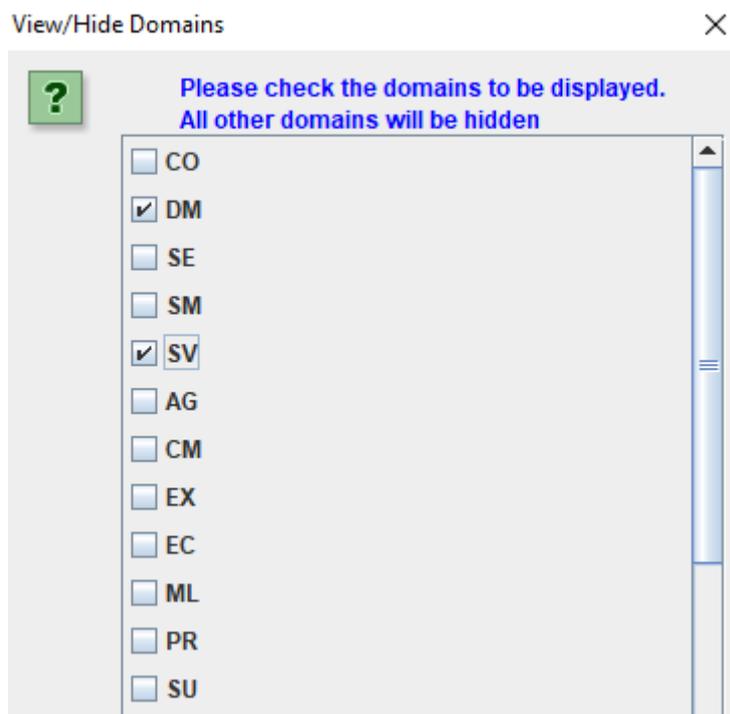
```

Viewing and hiding domains

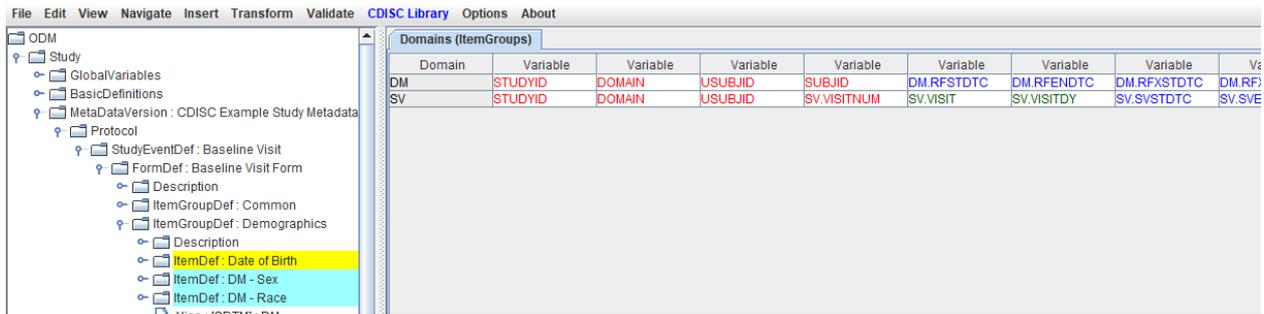
Each new SDTM-IG or SEND-IG version has a lot of new domains, and it is easy to lose overview. Therefore individual domains in the table on the right can be hidden or be displayed, so that one can concentrate on the ones that currently are of importance. To do so, use the menu "View – View/Hide domains":



A list of domains then is displayed, and we can check the ones that we want to keep displayed in the table (all others are hidden). For the moment, we just keep DM (Demography) and "SV" (Subject visits) as these can usually best be mapped first:



After clicking "OK", the table on the right reduces to:



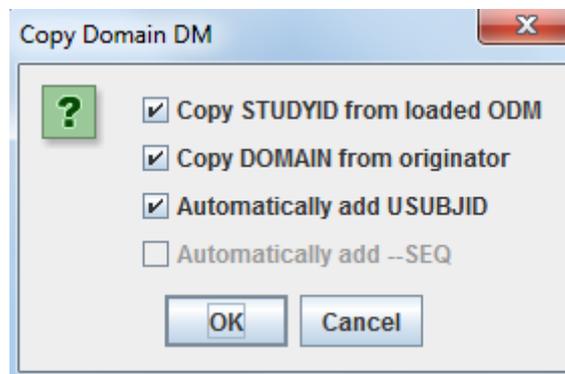
The mapping can begin...

As we do not want to edit the template domains themselves (well, it is not possible within the tool anyway), we need to create a **study-specific instance**. We will start with the DM domain.

There are two ways to do so:

- 1) drag-an-drop the "DM" row to the last row (which in our case is the "SV") using the mouse with the left mouse button down (release the left mouse button to "drop")
- 2) select one of the cells of the "DM" row and use the menu "Edit – Copy Domain/Dataset" (or use CTRL-B). Then select the last row of the table, and use the menu "Edit – Past Domain/Dataset" (or use CTRL-U)

In both cases, the following dialog is displayed:



The three first checkboxes are already checked in advance. The first means that the value for "STUDYID" in the SDTM will automatically be set to the value of the Study OID in the ODM (which is usually a wise decision).

The second will fix the value of the SDTM variable "DOMAIN" to the one from the template. This is almost always the case – later we will see in which cases one might want to make an exception. The third tells the system that for the SDTM variable USUBJID, it needs to take the value from the ODM, i.e. from the "SubjectKey" attribute of the "SubjectData" element in the ODM file with clinical data.

The fourth checkbox allows to have the --SEQ variable be calculated automatically by the system. In the "DM" domain however, there is not DMSEQ variable, so this checkbox is disabled here.

After clicking "OK", an information message is displayed:

 After having created a study-specific instance of a domain (i.e. dataset), it is always recommended to first check and/or adapt the looping structure, i.e. the structure 'One record per VARIABLE1 per VARIABLE2 per ...'. The structure suggested by the SDTM-IG is NOT always suitable for your study.

You can change the looping structure by using the menu 'Edit - SDTM Domain Properties', or by double clicking the first cell in the row designated 'CES:DM'.

Don't show me again

Stating that it is always a good idea to first look at, and often adapt the "structure" of the dataset. For this, don't automatically rely on what is provided in the SDTM- or SEND-IG: how the dataset is structured is **your choice**. What is provided in the SDTM-IG is just one of the possibilities.

The software will automatically assign the following variables for the "looping structure", which is suitable in 90% of the cases:

DM: one record per USUBJID

SV: one record per VISITNUM per USUBJID

Interventions: one record per --TRT (treatment) per USUBJID

Events: one record per --TERM (event term) per USUBJID

Findings: one record per --TESTCD (test) per USUBJID

In some cases, one can add "VISITNUM" or "VISIT" as an additional "looping variable", but this is usually not necessary. The same applies to "TPTNUM" (time point number).

Remark that this is different from the "uniqueness keys" (KeySequence attribute in the define.xml)

If one does not want to see this information message each time a new study-specific instance is created, check the checkbox "Don't show me again".

Clicking the OK button leads to our first mappings.

Domains (ItemGroups)							
Domain	Variable	Variable	Variable	Variable	Variable	Variable	
DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM
SV	STUDYID	DOMAIN	USUBJID	SV.VISITNUM	SV.VISIT	SV.VISITDY	SV
CES:DM	STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM

One sees that a new row has been created, with the name (OID in the define.xml) "CES:DM" for our study-specific DM domain. The color of three cells (STUDYID, DOMAIN and USUBJID) is changed to grey, meaning that a mapping script for these variables now exists.

Hovering the mouse over the first cell (CES:DM) shows:

Domains (ItemGroups)		
Domain	Variable	Variable
DM	STUDYID	DOMAIN
SV	STUDYID	DOMAIN
CES:DM	STUDYID	DOMAIN

OID: CES:DM
 Name: DM
 Purpose: Tabulation
 Repeating: No
 Is Reference Data: No
 Label: Demographics
 Class: Special Purpose
 Structure: One record per USUBJID
 Archive Location ID: Location.DM - 46

Later we will learn how to edit the properties of the domain instance. In the case of the DM domain there is currently no necessity to do so.

The mapping for a specific variable (e.g. "STUDYID") can be edited by double-clicking the cell. This leads to a new window that opens and shows:

Create XPath expression for new script variable

Variable Name: Clinical data
 Reference data

Variable Comment:

StudyEvent Rep.Key Form Rep.Key

Baseline Visit *

ItemGroup Rep.Key Item Attribute

* * StudyEventOID

Mapping Description and Link to external Document

Origin: No Origin has been added yet!

The Transformation Script

```
1 $STUDYID = "CES";
```

Scripting Language Functions

+	-	*	/	xpath	comment	decode
usubjid	investigator	site	name	sitename	question	alias
substring	substring-before	substring-after	concat	string-length	replace	tokenize
if	elseif	else	trim	upper-case	lower-case	
<	>	<=	>=	==	!=	
contains	starts-with	ends-with	matches	not		
abs	sqrt	log	log10	exp	exp10	
min	max	avg	sum	count	is-a-number	
ceiling	floor	round	modulus	number	string	
date	year	month-in-year	day-in-year	day-in-month	day-in-week	

This window is named the "**mapping editor**", which we will use a lot. Let us first look at the basic features of this mapping editor.

The upper panel is for advanced usage when complicated selections for items must be made. It can be hidden by using the button "Hide Upper Panel".

The smaller panel "Mapping Description" has already been prefilled. It contains a short description of the mapping. Please feel free to edit its text. Remark that for "derived" variables, the text will later flow into the "Method description" in the define.xml.

As of SDTM-ETL v.4.1, it is also indicated whether an "Origin" has been defined for the variable. Remind that in the case of a regulatory submission, the origin must be provided for each variable. It will later be explained in more detail how an origin can be assigned or edited.

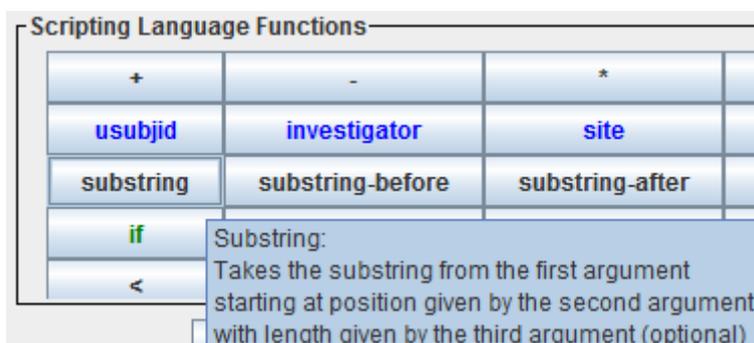
The most important panel is the panel "The Transformation Script". This is where the script is generated and/or edited. The scripting is in a special, easy-to-learn language. Although most of the scripts are generated automatically, it will be necessary to learn a bit about this scripting language, which is described in a special document "SDTM-ETL Scripting Language".

In the current case the mapping script is very simply:

```
$STUDYID = "CES";
```

stating the the variable STUDYID is a string (remark the quotes - it is also allowed to use single quotes) with a fixed value of "CES". Also notice the semicolon at the end marking the end of the statement.

The lower panel "Scripting Language Functions" contain a series of buttons for generating snippets of coding involving build-in functions. To get more explanation about a specific function, just hover the mouse over a button, e.g.:



We will later treat the use of functions in detail.

For very long or complicated mappings (which should be the minority – but that depends on your study design), one can "blow up" the central panel using the button "Full Screen Transformation Script Panel" which generates a full screen script editor panel.

When done editing the mapping script, click the "OK" button, or use the "Cancel" button to cancel all editing.

For the DM variables "DOMAIN" a similar mapping has already been generated automatically:



Double-clicking the cell "USUBJID" provides the mapping for the variable "USUBJID":

Mapping Description
USUBJID from ODM ClinicalData
The Transformation Script
<code>USUBJID = usubjid();</code>

The field "Mapping Description" has been pre-filled (but you can edit it) stating that the value will be taken from the ODM ClinicalData.

The transformation script itself uses a function **usubjid()**, which simply takes the value of the "SubjectKey" attribute of the SubjectData element in the ODM file with clinical data. Later in this tutorial, we will see how one can edit this, e.g. to set the value of the USUBJID as a concatenation of the study ID with the subject ID.

Let us now test this mapping on a real set of clinical data. For this, click the button "Test – Transform to XSLT". This will generate a mapping script in XSLT language⁷ (which you do not need to learn) to transform XML files or to extract information from XML files such as CDISC ODM files with clinical data.

The result of clicking the button "Test – Transform to XSLT" is a new window:

? Select whether the ODM file with clinical data works with non-typed or with typed ItemData

It uses non-typed ItemData (as in ODM 1.2) It uses TYPED ItemData (new as of ODM 1.3)

Never ask again in current session

OK

It asks you whether your ODM clinical data is "non-typed" or "typed". If you don't know, ask your EDC vendor or the source of your clinical data, or just try one of both possibilities (you will immediately find out which one applies). You can also have a quick look at a file with clinical data. In case you find a lot of "ItemData" elements with a "Value" attribute, this means that your data is "non-typed". For example:

⁷XSLT is an international standard from the W3C for transforming XML documents

```

<FormData FormOID="F_BASELINE">
  <ItemGroupData ItemGroupOID="IG_COMMON">
    <ItemData ItemOID="I_SITE" Value="23"/>
    <ItemData ItemOID="I_SUBJECTID" Value="001"/>
    <ItemData ItemOID="I_VISIT" Value="2010-02-27"/>
    <ItemData ItemOID="I_VISITTIME" Value="10:27:33"/>
  </ItemGroupData>
  <ItemGroupData ItemGroupOID="IG_DM">
    <ItemData ItemOID="I_BRTHDT" Value="1957-05-07"/>
    <ItemData ItemOID="I_SEX" Value="F"/>
    <ItemData ItemOID="I_RACE" Value="CAUCASIAN"/>
  </ItemGroupData>

```

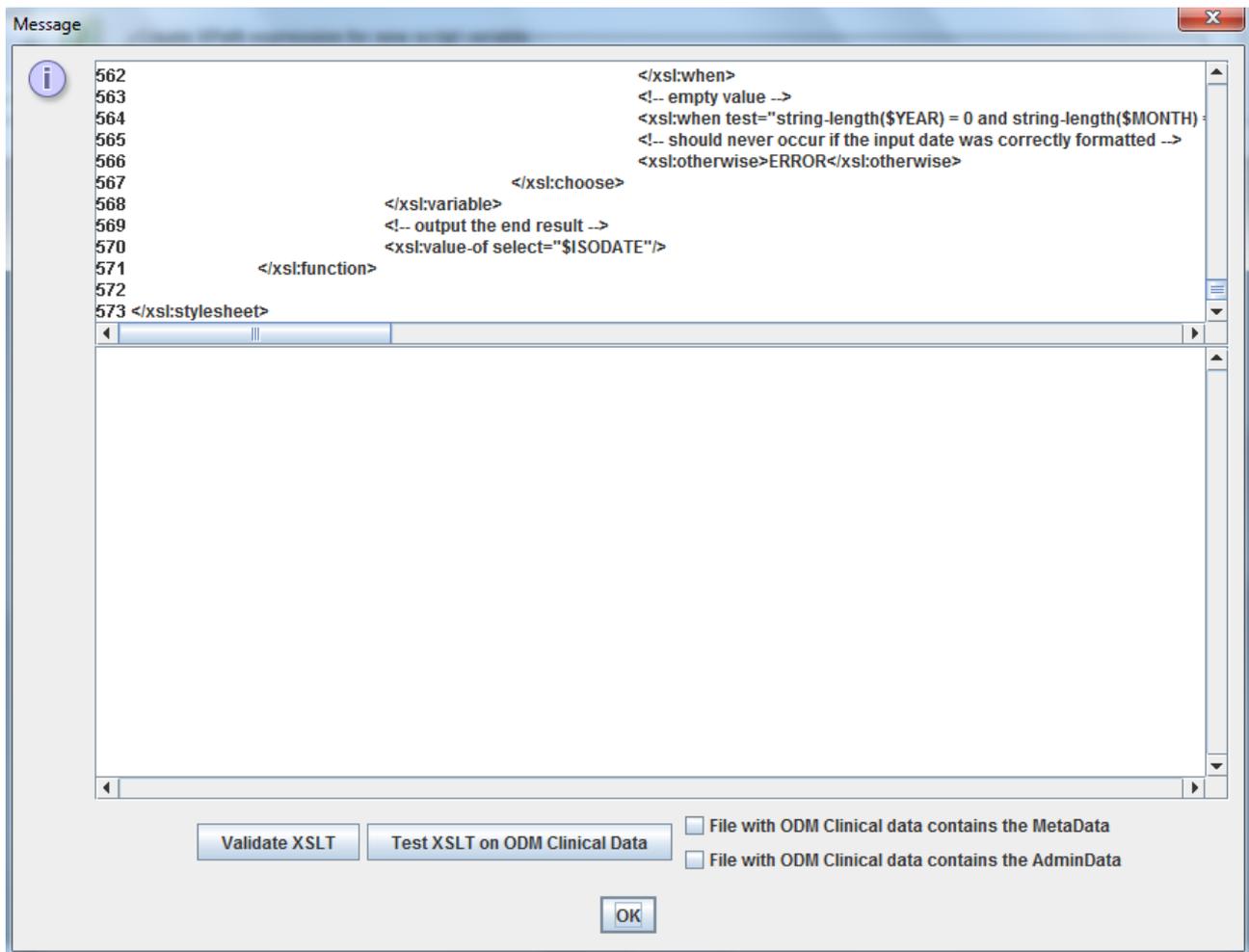
If your data however contain elements like "ItemDataString" or "ItemDataDate" and there is no "Value" attribute, this means that your data is "typed". For example:

```

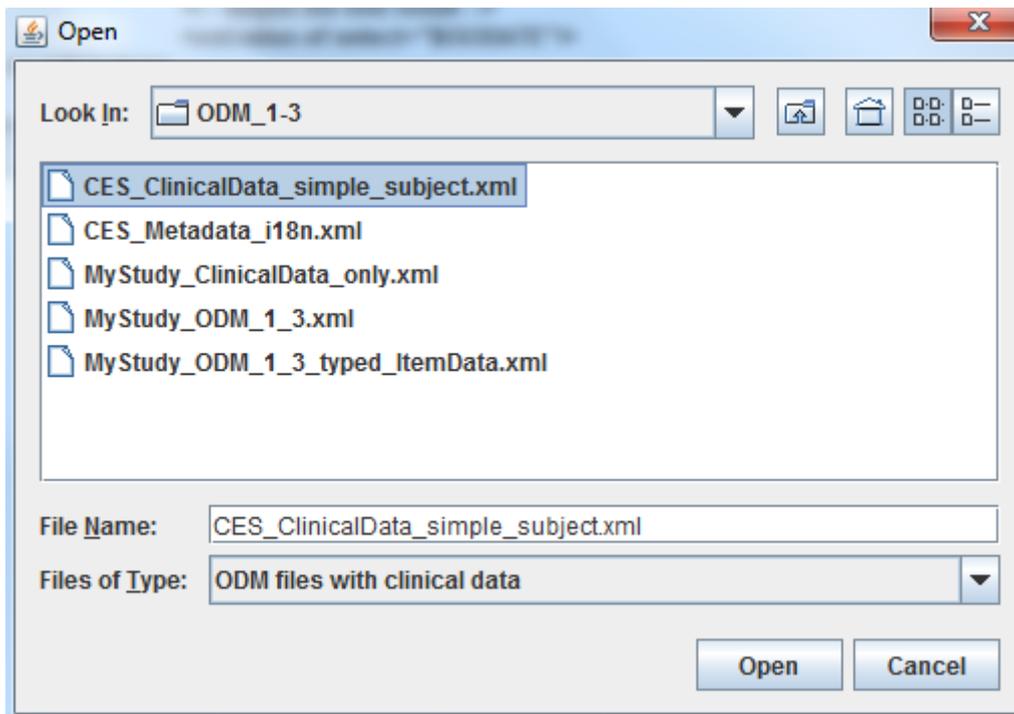
<SubjectData SubjectKey="002">
  <StudyEventData StudyEventOID="StudyEventOID" StudyEventRepeatKey="1">
    <FormData FormOID="FormOID" FormRepeatKey="1">
      <ItemGroupData ItemGroupOID="DATATYPE" ItemGroupRepeatKey="ALL ELEMENT" TransactionType="Insert">
        <ItemDataPartialDate ItemOID="ID.PD">1959-12</ItemDataPartialDate>
        <ItemDataPartialTime ItemOID="ID.PT">12</ItemDataPartialTime>
      </ItemGroupData>
    </FormData>
  </StudyEventData>

```

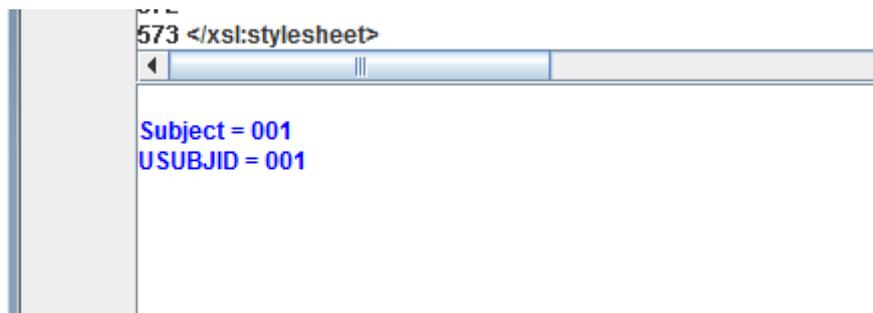
In our case, we work with "non-typed" data, so we leave the radiobutton "It uses non-typed ItemData" selected. If it is sure that your clinical data will always come as "untyped", one can check the checkbox "Never ask again in current session", and then this dialog will not show up again. Clicking "OK" leads to a dialog:



One can then validate the correctness of the generated XSLT, or just inspect it (specialists with very complicated scripts like to do so for debugging). In 99% of the cases, you will however just want to continue by clicking the "Test XSLT on ODM Clinical Data". This leads to a file-chooser allowing to pick the ODM file with clinical data. For example:

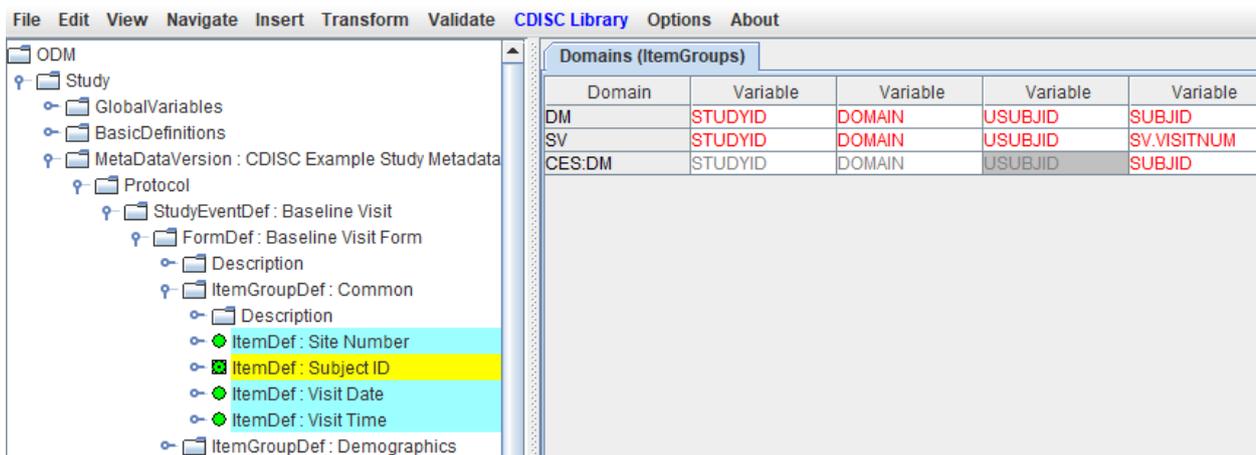


Clicking "Open" then immediately executes the script. As our file only contains the data for a single subject, the output is:



Notice that this testing mechanism only works for a single variable in a single domain, and e.g. not when the the script references other variables from the same or other domains. Later we will learn how to do more sophisticated testing.

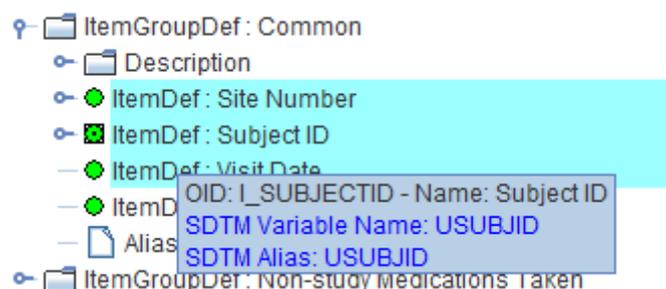
Let us now generate an alternative mapping for USUBJID. For example, we would like to have the value of USUBJID to be a concatenation of the STUDYID and of the subject ID from the "Common" section of each form. For doing so, first select the cell "USUBJID" and then expand the tree with the study design so that you see an item "Subject ID" in a group of items "Common". One can of course also do a search in the study design tree (see the document "Loading ODM"). For example:



- If one looks carefully, two important observations can be made:
- the items that are visible have a green "traffic light" in front of them
 - the item "Subject ID" has a traffic light that has a **square** around it

The green "traffic light" means that the item is of a suitable data type for mapping to the SDTM variable. For example, if one expects a datetime for an SDTM variable, the traffic light on the item "Subject ID" in the study design tree will be red⁸. The square around the green "traffic light" means that the item is a "hot candidate", i.e. has been annotated in the ODM as being ideally suited for mapping with the given SDTM variable.

This can also be seen by hovering the mouse over the item "Subject ID" in the study design tree:



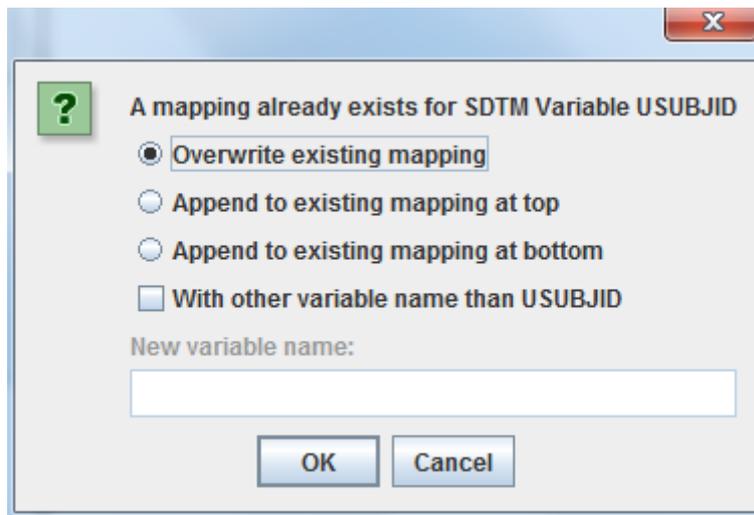
Technically, this was done by adding the attribute SDSVarName="USUBJID" in the ODM study design⁹.

To use the item "Subject ID" in the mapping for the SDTM variable "USUBJID", select the item "Subject ID" in the tree with the mouse, then drag it (keep the left mouse button down) to the cell "USUBJID" in the table on the right, then drop it by releasing the left mouse button. During the dragging, you will see a yellow "copy" symbol replacing your mouse cursor, meaning that you are in the "copy" mode.

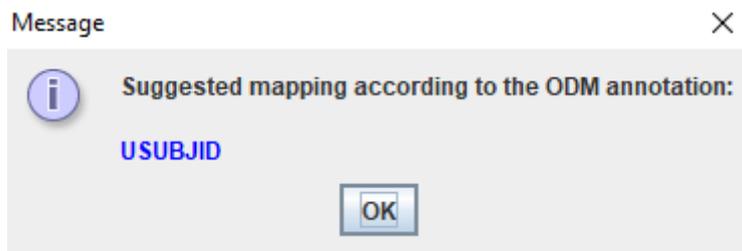
After having dropped in the "USUBJID" cell, the following dialog is displayed:

⁸Which does not mean that it cannot be used in that mapping – people drive through red traffic lights, but that is taking a big risk ...

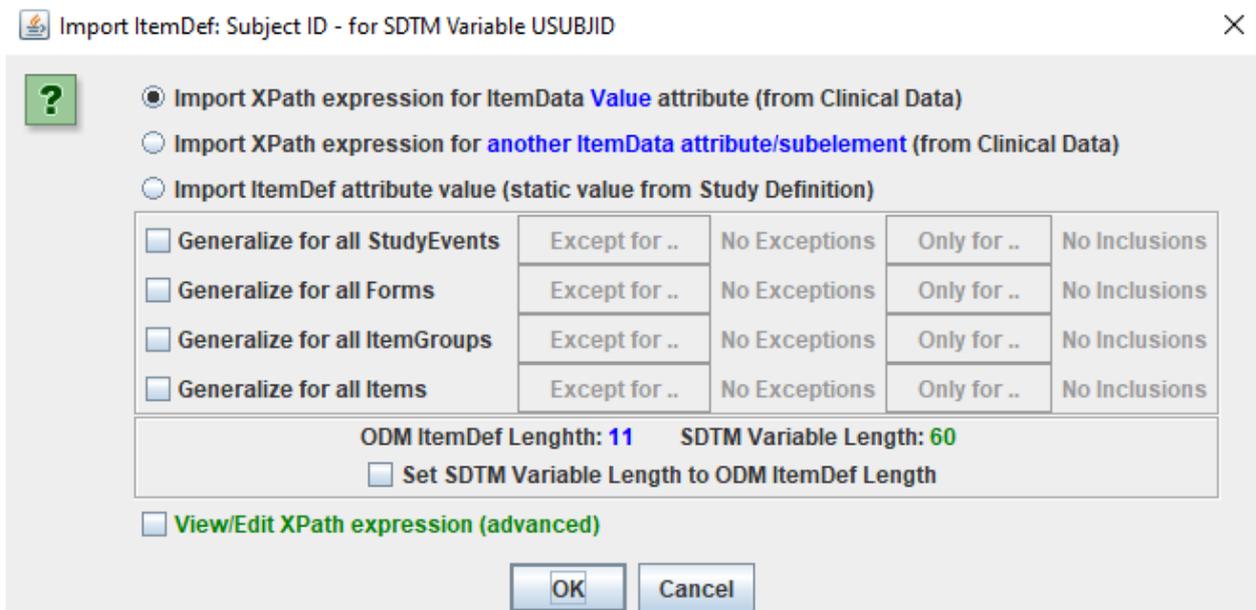
⁹ It is always a very good idea to develop the study design "with the end" (the submission) "in mind". More modern study designers such as the "ODM Designer" allow this.



as a mapping already exists for USUBJID. Select "Overwrite existing mapping" and click "OK". As the ODM ItemDef is having an SDTM annotation that it maps to "USUBJID", an information message is displayed:



This displays a new dialog:

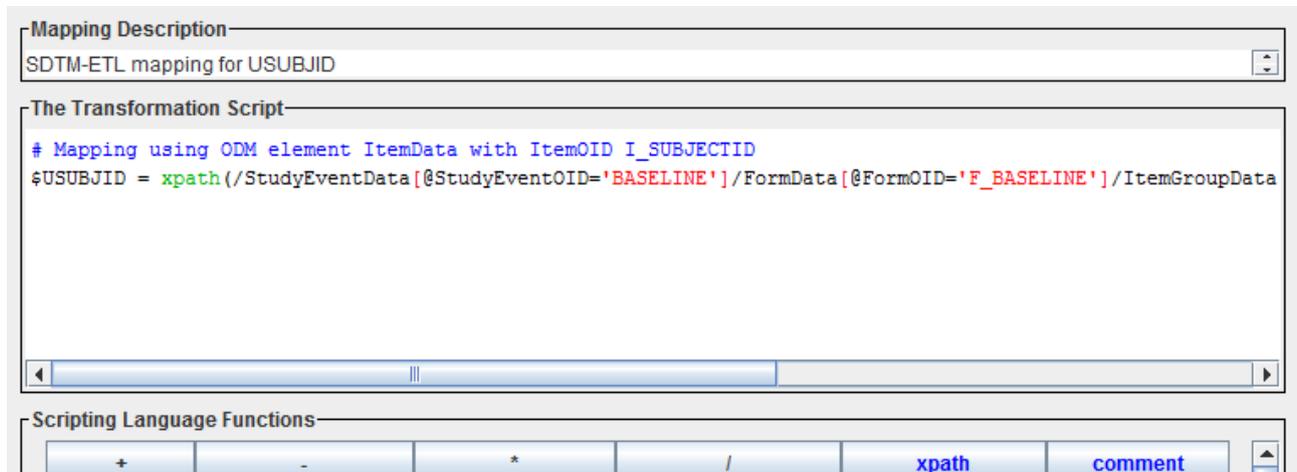


The most important radiobutton is the button "Import XPath expression for ItemData Value attribute (from Clinical Data)" meaning that we want to import a collected value (this will be >90% of the cases). We will describe the function of the other radiobuttons later.

The lower part of the dialog states that we currently have set the maximal length for USUBJID to 60 (being the default) from the template, but that the maximal length in the study was defined to be

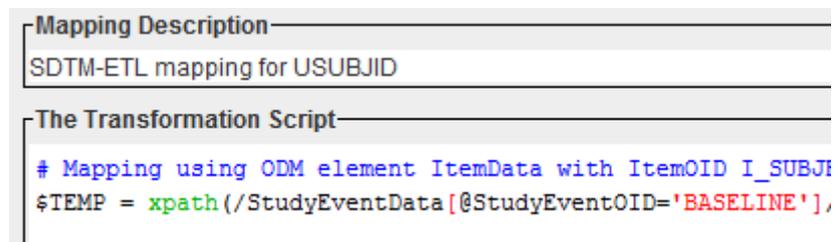
11. Checking the box "Set SDTM Variable Length to ODM ItemDef Length" allows to reduce the SDTM variable length to the one given in the study design, which is 11. Don't check the checkbox for now, as we still want to concatenate with the Study ID.

After clicking the OK button, the mapping scripting shows up:



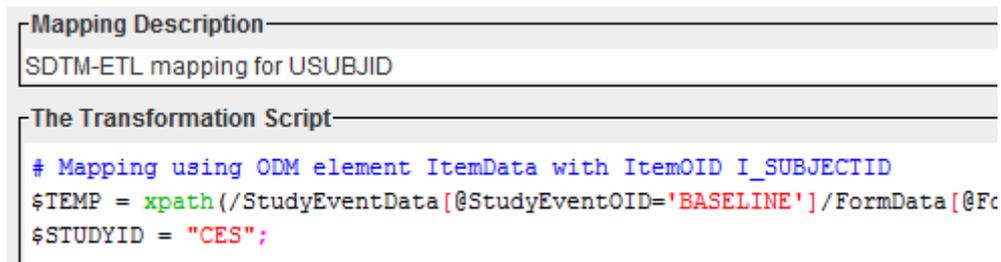
Essentially what it does, is to define a **path** to the item in the clinical data, and store the result in the variable \$USUBJID. As it is a path in XML, this is called an "XPath" expression. One can now test this script again on clinical data (as before), giving the same result as before.

Now we want to concatenate the value of STUDYID with the above result. In order to do so, we need to adapt the script slightly. First, the variable \$USUBJID is renamed into \$TEMP. We then have:



Do not change anything in the "XPath" expression¹⁰.

Now, we already have a mapping for the SDTM variable \$STUDYID. We can just copy-paste from the previous mapping which results in:



¹⁰ It will be very seldom that one needs to change something in the XPath expression. We will give some examples later though

Now have a look at the functions in the lower panel, the "Scripting Language Functions" panel. You will find a "concat" function with the following explanation:

name	sitename	question
concat	string-length	replace
Concatenates the contents of the arguments into a single string		

The "concat" function has at least two arguments, but there can be more. It is used to concatenate a set of strings into a new string.

Now in the mapping script editor, just type:

```
$USUBJID =
```

and then click the "concat" button. The string is extended with the function with empty parameters:

```
Mapping Description
SDTM-ETL mapping for USUBJID

The Transformation Script
# Mapping using ODM element ItemData with ItemOID I_SUBJECTID
$TEMP = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormD
$STUDYID = "CES";
$USUBJID = concat(, ) |
```

which can now easily be extended as:

```
Mapping Description
SDTM-ETL mapping for USUBJID

The Transformation Script
# Mapping using ODM element ItemData with ItemOID I_SUBJECTID
$TEMP = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormD
$STUDYID = "CES";
$USUBJID = concat($STUDYID,$TEMP);
```

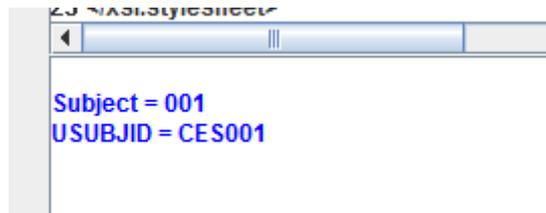
Do not forget the semicolon at the end¹¹.

You might already have noticed the coloring in the script: **comments** (starting with a "#") are colored blue¹². Functions are colored green, and strings (that are between quotes) are colored red.

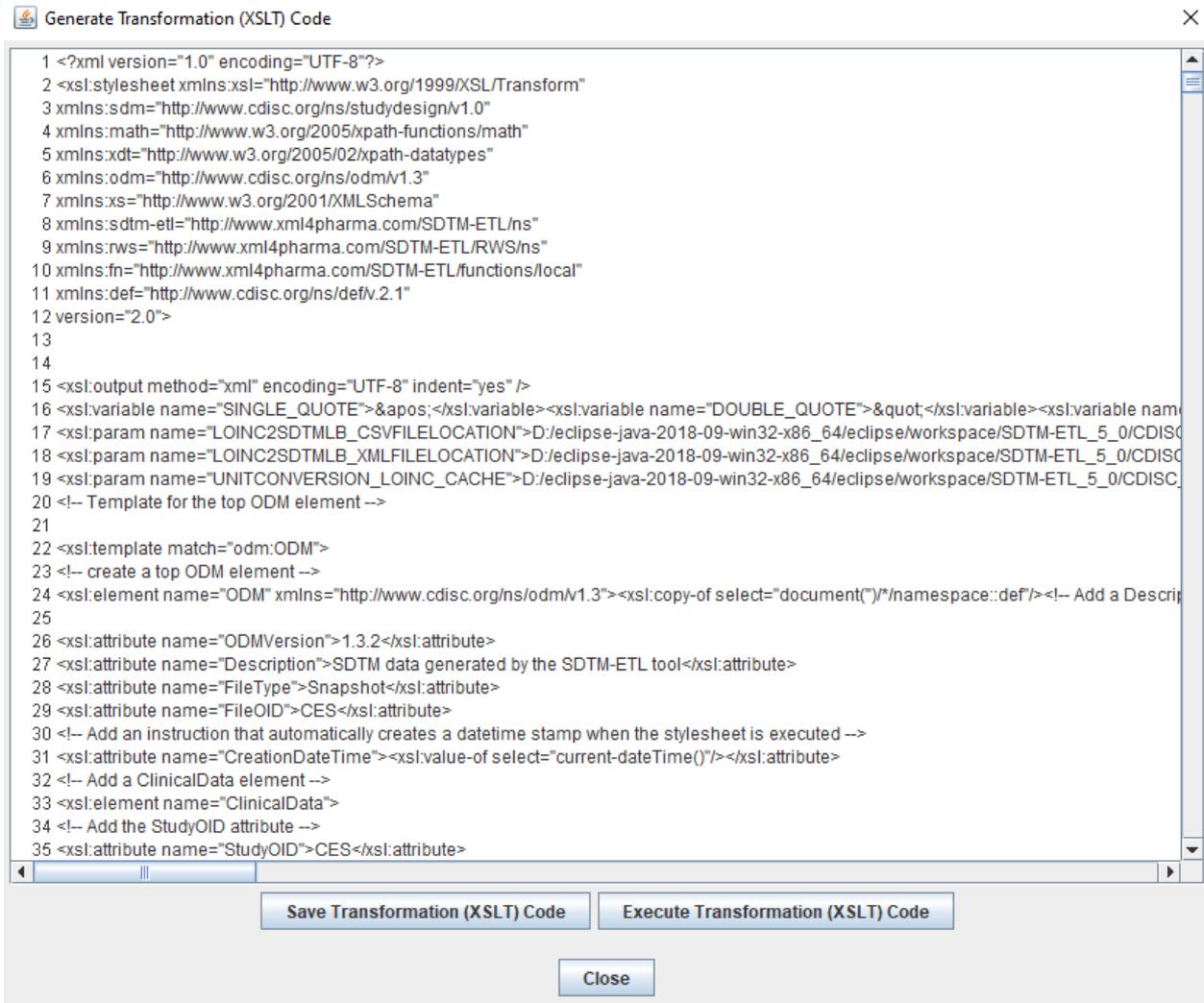
Re-executing the mapping script on real clinical data results in:

¹¹If the semicolon is forgotten, a warning message will be displayed when trying to execute the mapping.

¹²As in every programming effort, it is advised to add as many comments as possible, for a later good understanding what the intention of the statement or snippet was.



One can also execute all the available mappings together. After clicking OK for the mapping script editor, we come into the main window again. Now, use the menu "Transform – Generate Transformation (XSLT) Code". The following dialog is presented:



One can now save the transformation XSLT code to file¹³, but we will execute the code within the software itself, so click "Execute Transformation (XSLT) Code". This results in a new dialog:

¹³ This can be useful to execute the transformations off-line, as is done by some of our advanced users.

Execute Transformation (XSLT) Code X

ODM file with clinical data:

C:\ Browse...

MetaData in separate ODM file

D:\SDTM-ETL\TestFiles\ODM1-3-1\CES_Metadata.xml Browse...

Administrative data in separate ODM file

D:\SDTM-ETL\TestFiles\ODM1-3-1\CES_Metadata.xml Browse...

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

Dataset-JSON 1.1 SAS-XPT UTF-8 encoded CSV SQL INSERT statements

SDTM export files directory:

Browse...

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

Messages and error messages:

Execute Transformation on Clinical Data

The upper field allows to add the location of the ODM file with clinical data. One can use the "Browse" button to locate this file.

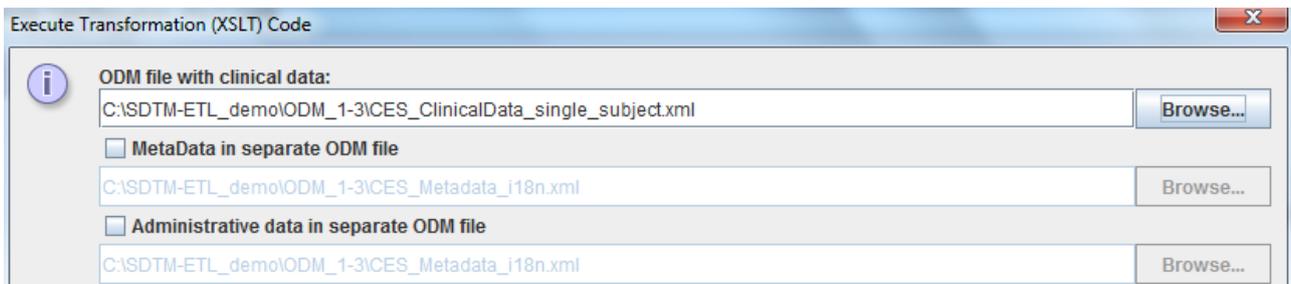
At the moment, as we are just testing we do not need to generate any files, so we leave the checkbox "Save Result SDTM tables" unchecked. The checkbox "View Result SDTM files" remains checked – this will open an own internal viewer for the results that we have so far.

One of the advantages of the SDTM-ETL software is that one can start developing the mappings even before the first subject has enrolled. But in order to test the mappings, we need some clinical data, even if it is mock data¹⁴.

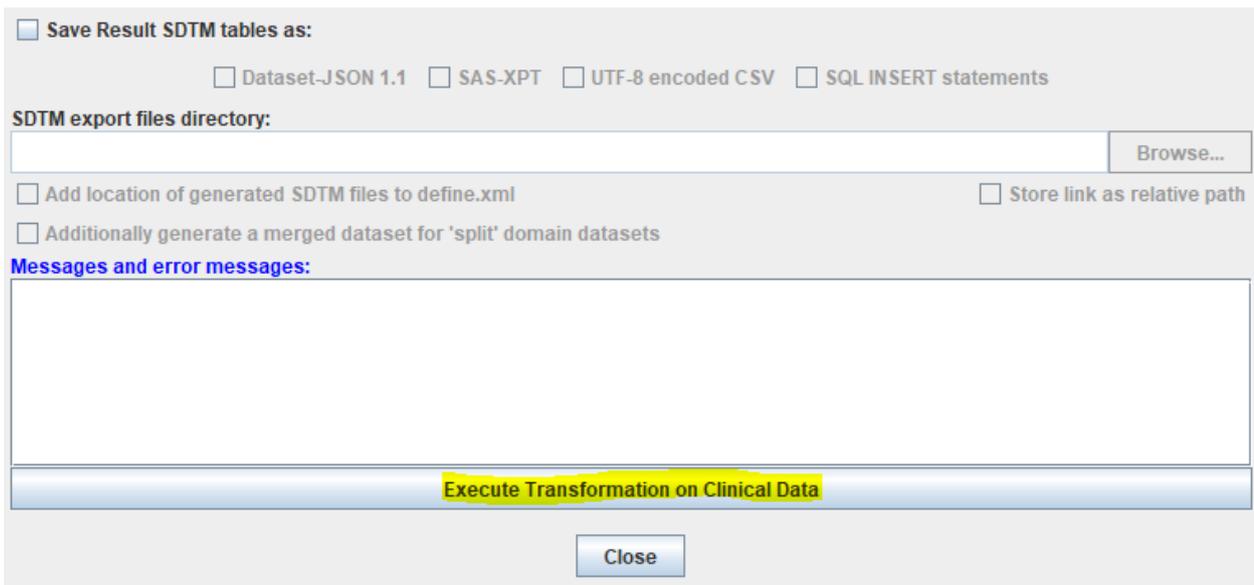
Consider the case that we already have some (but not all) collected data of a first subject. We can already use these data to test our mapping.

For example:

¹⁴ Some of our users develop and generate mock data even before the study starts, in order to test whether the (future) study data can be mapped to CDISC SDTM.



and we can now click the button "Execute Transformation on Clinical Data"



After a few seconds, the transformation has been executed, and a new window with the results (those that we have so far) is displayed:



Remarks:

If you would liked to have a dash between the study ID and the subject ID for USUBJID, you could have used: `$USUBJID = concat($STUDYID, '-', $STEMP);`

In the remaining of the tutorial, we will however use the default mapping which is:

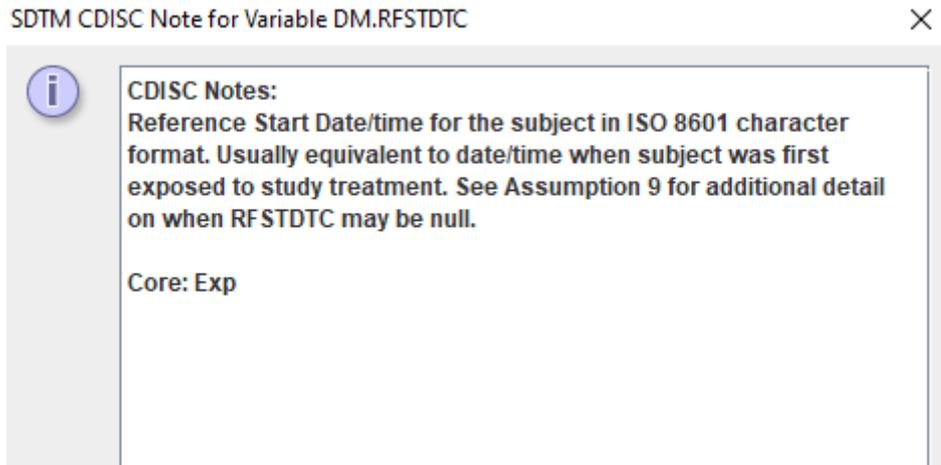
`$USUBJID = usubjid();`

taking the value of the ODM "SubjectKey" attribute of the "SubjectData" element.

For the variable SUBJID, we can also use the same mapping `$USUBJID=usubjid();` but you can also decide otherwise.

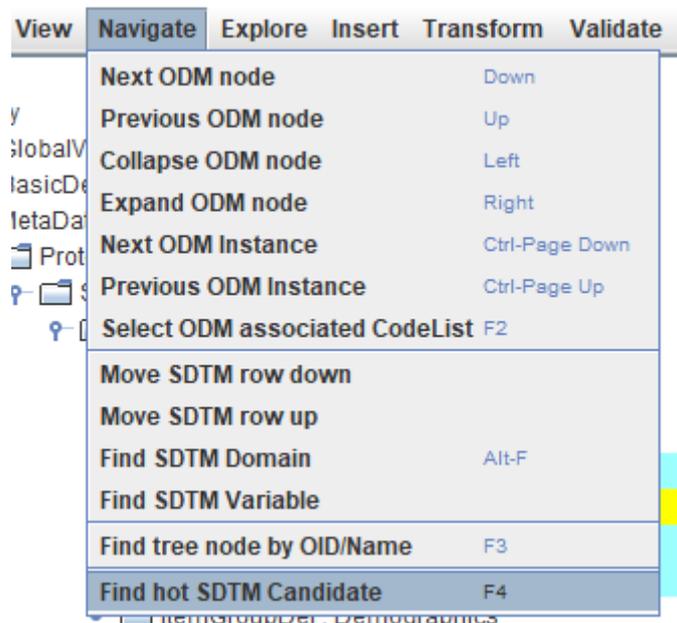
The next variable is RFSTDTC (Reference Start Date/time). In order to get more information on

this item use CTRL-H or the menu "View – SDTM CDISC Notes“. This displays the window:



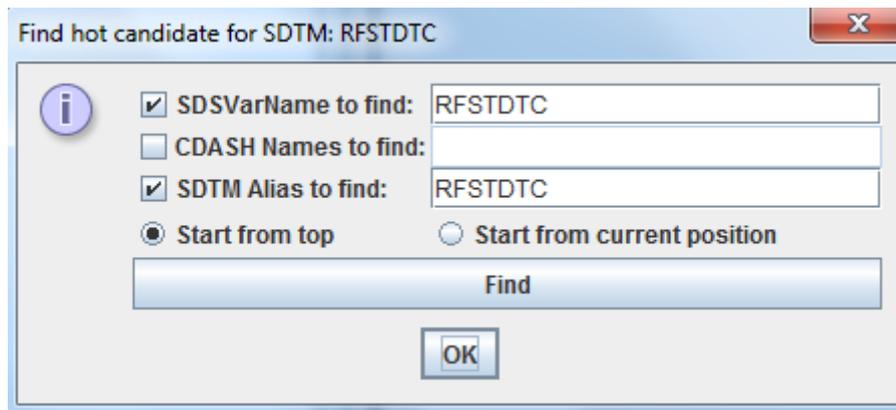
We can easily map this to the date of the first visit¹⁵.

Maybe there is a "hot candidate“ in the ODM for RFSTDTC, i.e. the ODM has been annotated that the item is ideally suited to be used for RFSTDTC. For finding out, first select the RFSTDTC cell and then use the menu "Navigate – Find hot SDTM Candidate"::



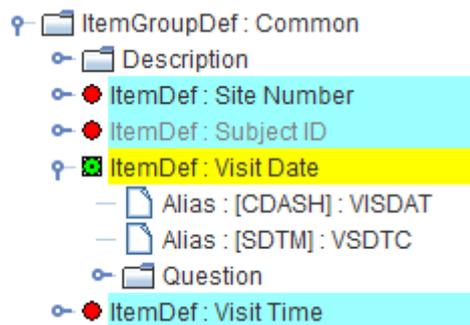
The following dialog is displayed:

¹⁵Our very simple sample study does not have a data point for "date of first study treatment“. If there is such a data point, the corresponding date can (or even is advised to) be used.

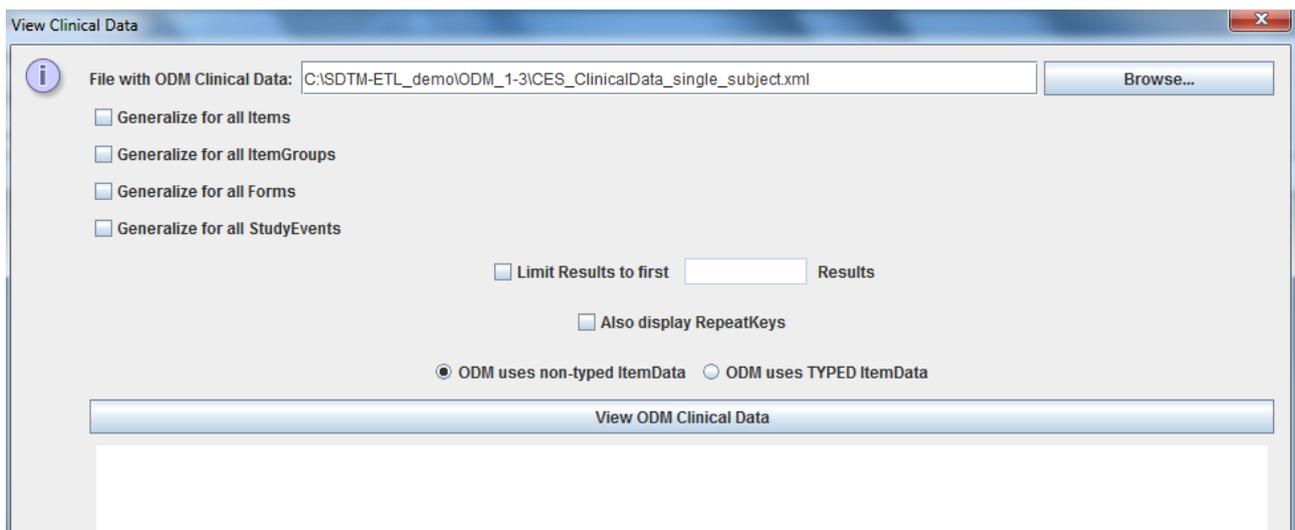


One can select to search using the "SDSVarName" (meaning "SDTM Variable Name") in the ODM, the CDASH name and/or the "SDTM Alias".

After clicking "Find", and if there is a "hot candidate" in the ODM, the tree will automatically expand, and the "hot candidate" item is displayed and selected:



But, are there already any clinical data for this data point? One can test using the menu "View – ODM Clinical Data". This shows the window:



As a file with clinical data has already been used for testing, the field "File with ODM Clinical Data" is already pre-filled. So one only need to click the button "View ODM Clinical Data" which results in:

ODM uses non-typed ItemData
 ODM uses TYPED ItemData

View ODM Clinical Data					
Subject	StudyEvent	Form	ItemGroup	Item	Value
001	BASELINE	F_BASELINE	IG_COMMON	I_VISIT	2010-02-27

The rightmost column showing the value, and the other columns the subject ID, the StudyEvent (visit), Form, and ItemGroup, as well as the current Item.

The menu "View – ODM Clinical Data" will often be extremely useful to find out whether the current item is really the one we need or want for the mapping.

The same can be applied to check whether also the time of the first visit was collected using the Item "Visit Time" (OID I_VISIT_TIME):

View ODM Clinical Data					
Subject	StudyEvent	Form	ItemGroup	Item	Value
001	BASELINE	F_BASELINE	IG_COMMON	I_VISITTIME	10:27:33

As well as a visit date as time is present, they can both be used to populate RFSTDTC. To do so, drag the item "Visit Date" to the cell "RFSTDTC", leading to:

```

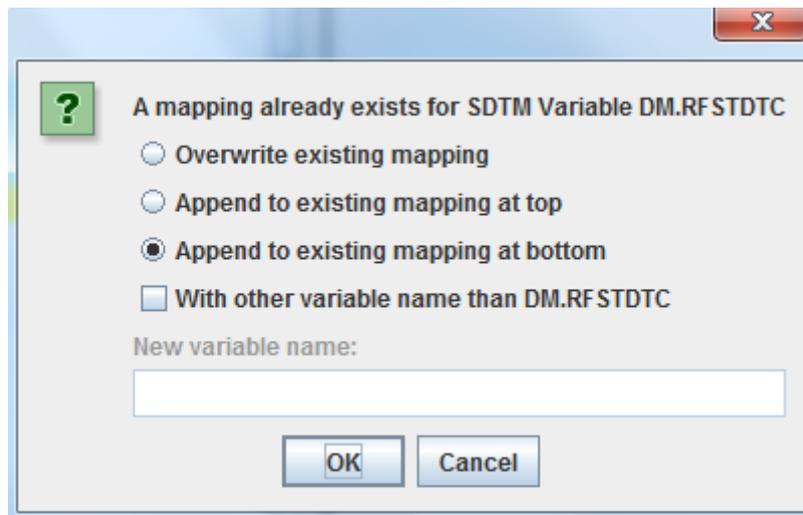
Mapping Description
SDTM-ETL mapping for DM.RFSTDTC

The Transformation Script
# Mapping using ODM element ItemData with ItemOID I_VISIT
$DM.RFSTDTC = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData[@FormOID='F_BASELINE']/ItemGroupD

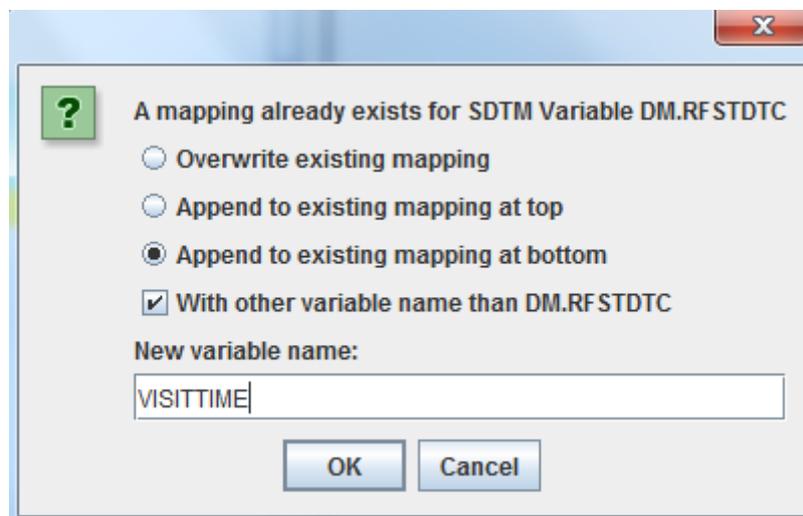
```

and rename \$DM.RFSTDTC into "\$VISITDATE".

Then drag-and-drop the item "Visit Time" to the same cell RFSTDTC. The following wizard is displayed:

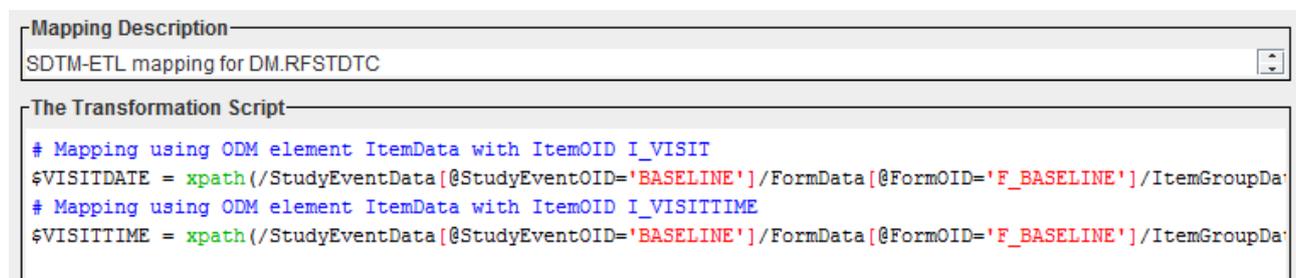


We want to append to the existing mapping, but as we still need to combine both items, we choose to rename the current one, e.g. to \$VISITTIME:



You do not need to add a "\$" in front of the new variable name, the system will take care of it.

This results in a mapping:



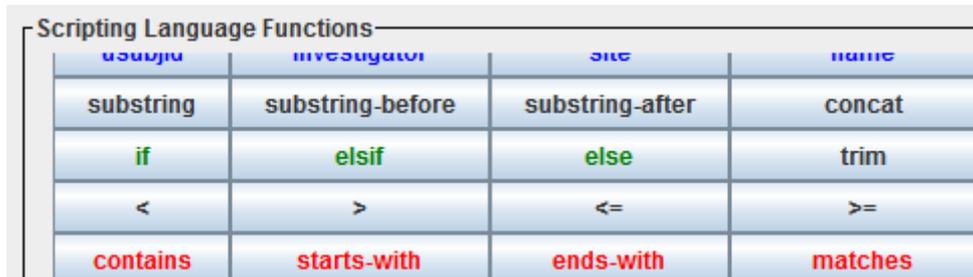
Remark that the two comment lines have been generated automatically.

The SDTM Implementation Guide explains the usage of ISO-8601 dates, times and datetimes. In case of a complete datetime, the format is: YYYY-MM-DDThh:mm:ss. The central "T" separating the date part from the time part. So for our mapping, we can use:

\$DM.RFSTDTC = concat(\$VISITDATE, 'T', \$VISITTIME);

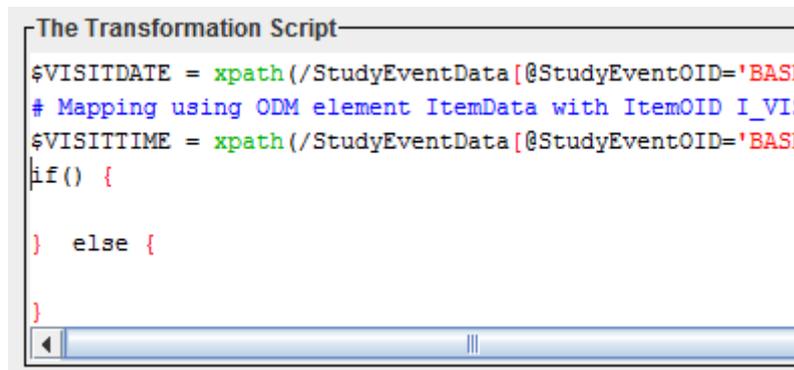
Hey, wait a minute! What in the case that the visit time was not collected? Then the central "T" should not be present! So ... **time for our first if-else statement!**

Like for the "concat" function, one can use the "if", "elsif" and "else" buttons from the "Scripting Language Functions" panel to insert snippets:



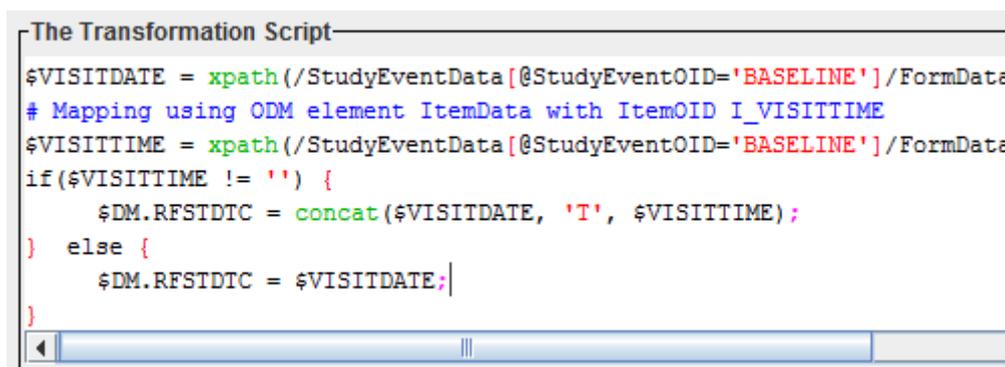
substring	substring-before	substring-after	concat
if	elsif	else	trim
<	>	<=	>=
contains	starts-with	ends-with	matches

e.g. leading to:



```
The Transformation Script
$VISITDATE = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData
# Mapping using ODM element ItemData with ItemOID I_VISITTIME
$VISITTIME = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData
if() {
} else {
}
}
```

One can then fill in the individual parts:



```
The Transformation Script
$VISITDATE = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData
# Mapping using ODM element ItemData with ItemOID I_VISITTIME
$VISITTIME = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData
if($VISITTIME != '') {
    $DM.RFSTDTC = concat($VISITDATE, 'T', $VISITTIME);
} else {
    $DM.RFSTDTC = $VISITDATE;
}
}
```

The "if" statement saying that in case the VISITTIME variable is not empty ("!=" symbol), then the value of DM.RFSTDTC is the concatenation of the visit date with the character "T" and the visit time. In any other case ("else" statement), the value of DM.RFSTDTC only consist of the date.

Testing on our single subject leads to:

```
24 </xsl:stylesheet>
```

Subject = 001
DM.RFSTDTC = 2010-02-27T10:27:33

One may ask why it is so important to have a time part for the value of RFSTDTC. The reason is that RFSTDTC will be used in other domains to assign whether the event, intervention or finding collection started before or after the formal start of the study for each subject. Think about the --STRF (Start Relative to the Reference Period) variables that are used to identify the start of an observation relative to the sponsor-defined study reference period. **In case no time was collected, and the date of the intervention, event or finding is on the same day as the study start date, there is essentially no possibility (except maybe from the protocol) to find out whether that action or event was before or after study start for the given subject.**

The next SDTM variable that needs to be mapped is RFENDTC (Reference end date/time). Using CTRL-H tells us:

SDTM CDISC Note for Variable DM.RFENDTC

CDISC Notes:
Reference End Date/time for the subject in ISO 8601 character format. Usually equivalent to the date/time when subject was determined to have ended the trial, and often equivalent to date/time of last exposure to study treatment. Required for all randomized subjects; null for screen failures or unassigned subjects.

Core: Exp

Add CDISC Library information

The use of the "Add CDISC Library information" button will further be explained in the tutorial "Working with the CDISC Library".

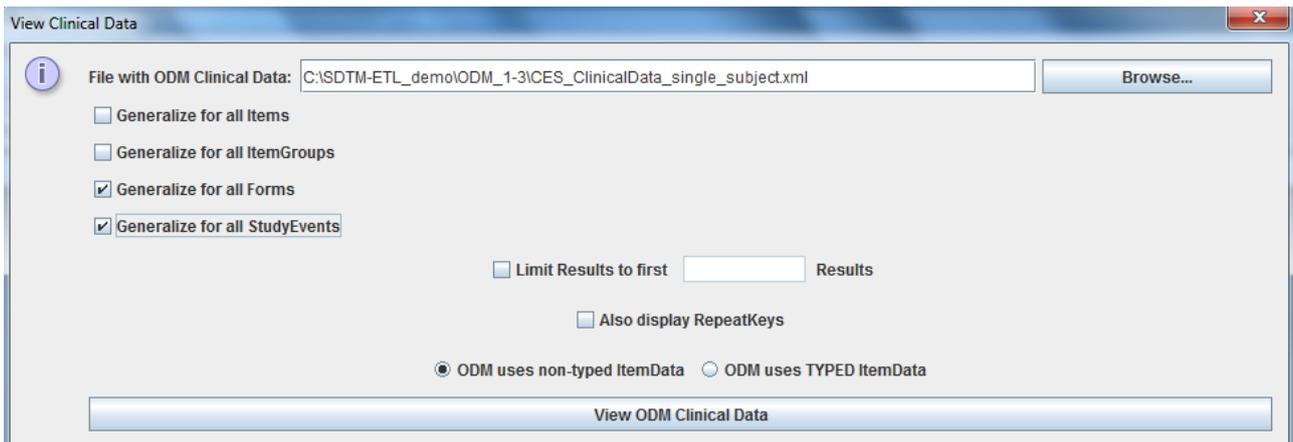
But now the question arises: what was the date the subject ended the trial? Was it the "Week 2 Visit", or was it the "Patient Diary Event", or maybe even the "Adverse Event" visit? This time the menu "Navigate – Find hot SDTM Candidate" does not give any results, so we need

to find out ourselves...

We can easily find out what the last visit date is, as it was always collected (i.e. in each visit) using the same item ("Visit date", with OID "I_VISIT"). One can easily see this by selecting the item, and then use "Navigate – Next Instance" (or use CTRL-Page-down). One will then see that it was collected for each form for each visit.

But what was the last one?

Here again, the menu "View – ODM Clinical Data" is of great help. So select an item "Visit Date" and then use the menu "View – ODM Clinical Data":



This time, check the checkboxes "Generalize for all Forms" and "Generalize for all StudyEvents". This means that we want to see each data point "Visit Date" independent from within which form and within which visit. Clicking the "View ODM Clinical Data" button leads to:

Subject	StudyEvent	Form	ItemGroup	Item	Value
001	BASELINE	F_BASELINE	IG_COMMON	I_VISIT	2010-02-27
001	BASELINE	F_CM	IG_COMMON	I_VISIT	2010-02-27
001	BASELINE	F_LAB	IG_COMMON	I_VISIT	2010-02-27
001	WEEK_1	F_WEEK_1_2	IG_COMMON	I_VISIT	2010-03-06
001	WEEK_1	F_LAB	IG_COMMON	I_VISIT	2010-03-06
001	WEEK_2	F_WEEK_1_2	IG_COMMON	I_VISIT	2010-03-13
001	WEEK_2	F_LAB	IG_COMMON	I_VISIT	2010-03-13
001	DIARY	F_DIARY	IG_COMMON	I_VISIT	2010-03-13
001	AE	F_AE	IG_COMMON	I_VISIT	2010-03-13
001	AE	F_CM	IG_COMMON	I_VISIT	2010-03-13

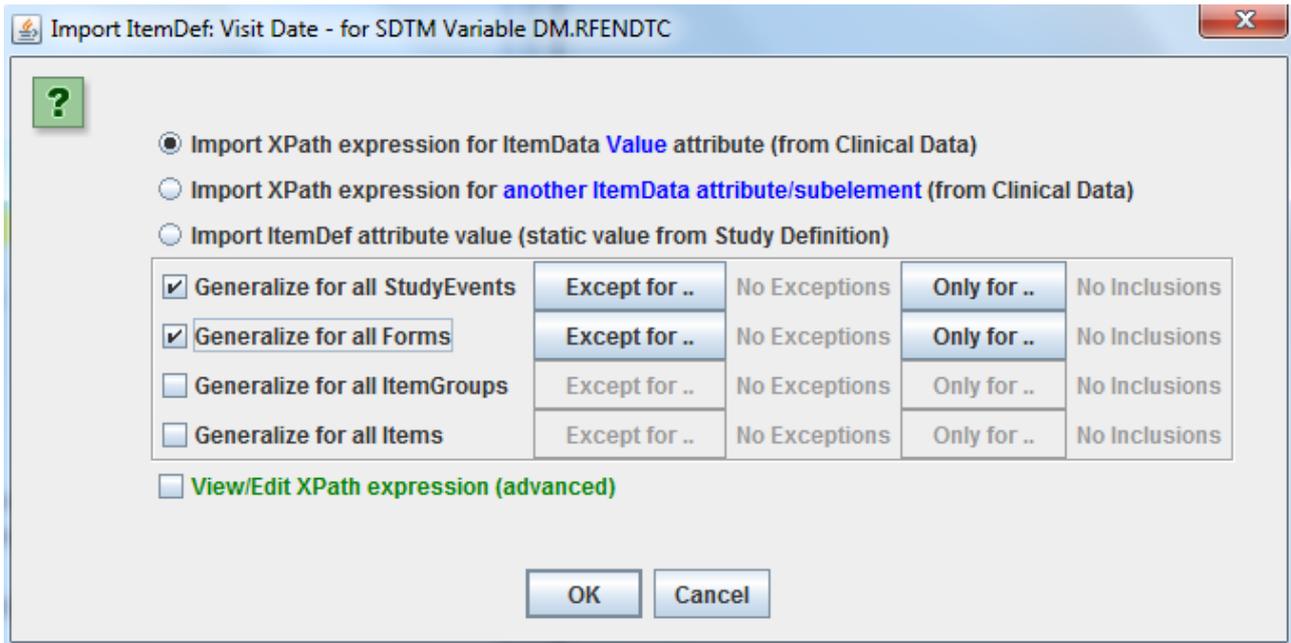
showing all the visit dates ever registered.

It looks as (at least for this subject) the last visit date was on March 13th 2010, and the visit was either "WEEK_2" or "DIARY" or "AE", which all happened on the same day. However, we cannot know whether this will apply to all subjects.

The ODM standard states that clinical data for subjects MUST come in chronological order, with earliest data first, and latest data last in the file. So we can simply look for the last occurrence of "Visit Date" for each subject in the file with clinical data.

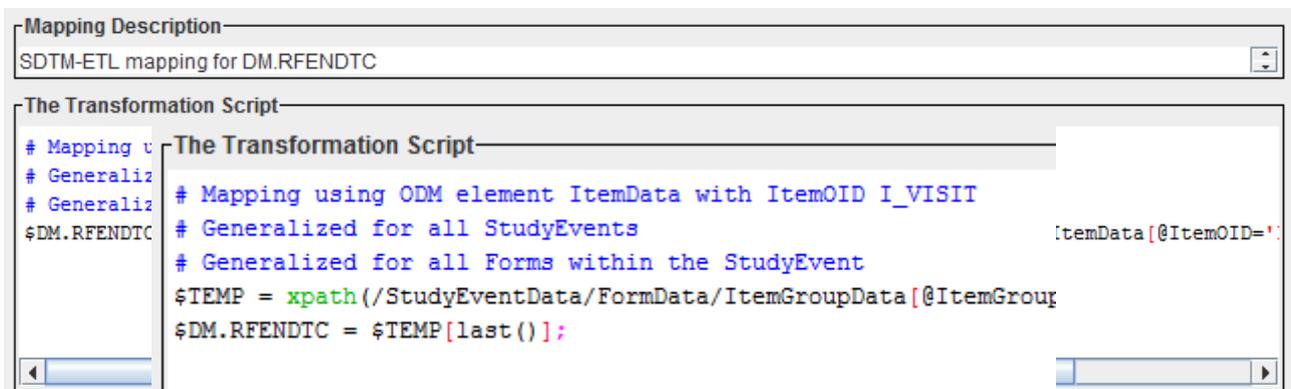
After having gone back to the main window, drag-and-drop one of the items "Visit date" from the tree with the study design (it doesn't matter which one), and drop it in the cell "RFENDTC". The

following dialog is displayed:

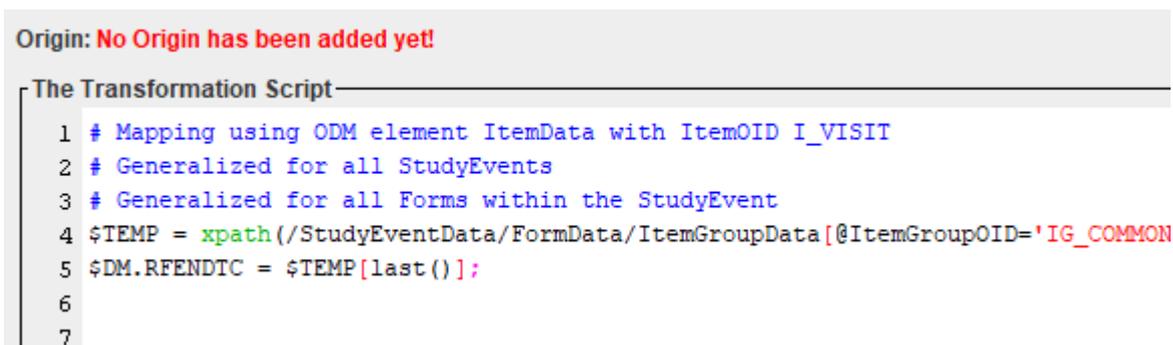


Check the checkboxes "Generalize for all StudyEvents" and "Generalize for all Forms", stating that we want to have the item independent of the form or visit¹⁶.

This leads to the mapping:



But we only want the last one, so we do a little rewrite into:



¹⁶Later we will see how to work with the buttons "Except for ..." and "Only for ..."

i.e. By generating a temporary variable, and then adding a condition [last()] to the expression¹⁷.

Another possibility is to use one of the many "datetime" functions, in this case "latestdate()":

```
Origin: No Origin has been added yet!

The Transformation Script
1 # Mapping using ODM element ItemData with ItemOID I_VISIT
2 # Generalized for all StudyEvents
3 # Generalized for all Forms within the StudyEvent
4 $TEMP = xpath(/StudyEventData/FormData/ItemGroupData[@ItemGroupOID
5 $DM.RFENDTC = latestdate($TEMP);
6
7
```

Which may be a more safe method.

The use of functions (and how they can be extended by the user) will be explained further on and in other tutorials.

Then executing the mapping script leads to:

STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC
CES	DM	001	001	2010-02-27T10:27...	2010-03-13

Remark that in many studies, there is an "End of Study" form (EOS), which collects what the last date/time was that the subject took part in the study or had contact with the study.

If such a form exists, it is of course the best to use the information from this form for mapping to RFSTDTC, but also RFPENDTC (Date/Time of End of Participation) which must correspond to the last contact date/time.

In a good number of cases, earlier defined mappings (i.e. for variables more to the left in the same domain) can easily be reused. For example, for the next variable "RFXSTDTC" in the DM domain, we can write:

```
Mapping Description
SDTM-ETL mapping for DM.RFXSTDTC

The Transformation Script
# copied from RFSTDTC
$DM.RFXSTDTC = $DM.RFSTDTC;
```

In case we select that the "Reference Start Date/Time" is the same as the "Date/Time of First Study Treatment".

¹⁷"taking the first one available" is written as "[1]"

Similarly, we can set for, when applicable, the next DM variables:

```
$DM.RFXENDTC = $DM.RFENDTC;  
$DM.RFICDTC = $DM.RFSTDTC;  
$DM.RFPENDTC = $DM.RFENDTC;
```

but of course only in the case dates were really identical to the first and last visit date correspondingly.

This leads to the following result:

STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC	DM.RFXENDTC	DM.RFICDTC	DM.RFPENDTC
CES	DM	001	001	2010-02-27T10:27:33	2010-03-13	2010-02-27T10:27:33	2010-03-13	2010-02-27T10:27:33	2010-03-13

Meanwhile we have received the data of a second subject, so we can test our mapping again:

Execute Transformation (XSLT) Code

ODM file with clinical data:
C:\SDTM-ETL_demo\ODM_1-3\CES_ClinicalData_two_subjects.xml

MetaData in separate ODM file
C:\SDTM-ETL_demo\ODM_1-3\CES_Metadata_j18n.xml

Administrative data in separate ODM file
C:\SDTM-ETL_demo\ODM_1-3\CES_Metadata_j18n.xml

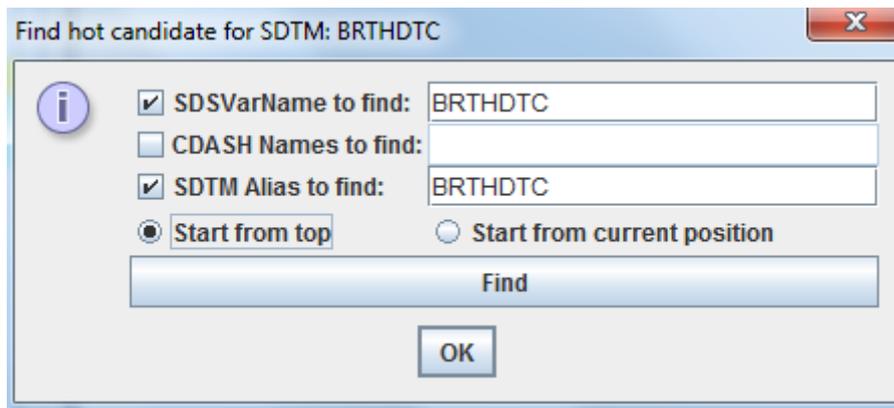
Split records > 200 characters to SUPP-- records

Resulting in:

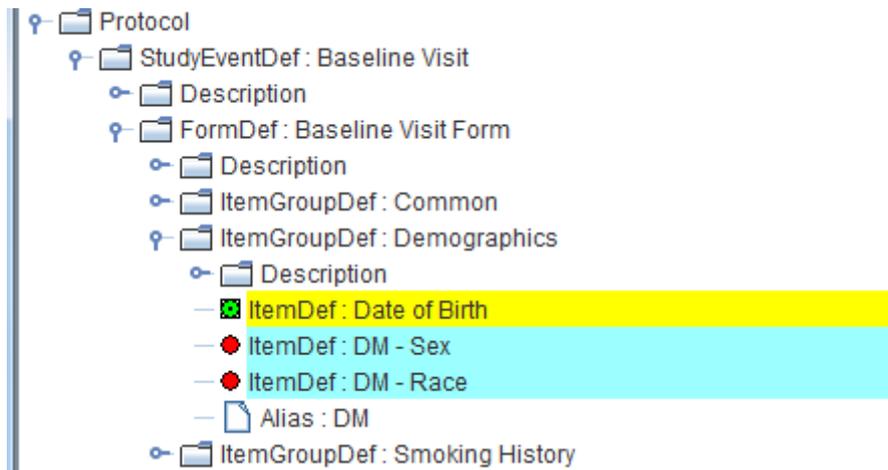
STUDYID	DOMAIN	USUBJID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC	DM.RFXENDTC	DM.RFICDTC	DM.RFPENDTC
CES	DM	001	001	2010-02-27T10:27:33	2010-03-13	2010-02-27T10:27:33	2010-03-13	2010-02-27T10:27:33	2010-03-13
CES	DM	002	002	2010-02-28T14:33:12	2010-03-16	2010-02-28T14:33:12	2010-03-16	2010-02-28T14:33:12	2010-03-16

In the tutorial "Creating and working with Subject Global Variables", we will learn how to make RFSTDTC and RFENDTC as "global" variable that can be used (in read-mode) over and over again, as it is used in many domains for the calculation of the "visit day" (-DY) variable and for variables that are about "relative to start ..." or "relative to end of study participation".

Let us now concentrate on two other important SDTM variables in the SDTM domain: BRTHDTC and AGE. Again we first try to find a "hot candidate" in our ODM tree.



With the result:



A view in the clinical data for this item (using "View – ODM Clinical Data") results in:

ODM uses non-typed ItemData
 ODM uses TYPED ItemData

View ODM Clinical Data					
Subject	StudyEvent	Form	ItemGroup	Item	Value
001	BASELINE	F_BASELINE	IG_DM	I_BRTHDT	1957-05-07
002	BASELINE	F_BASELINE	IG_DM	I_BRTHDT	1961-09-06

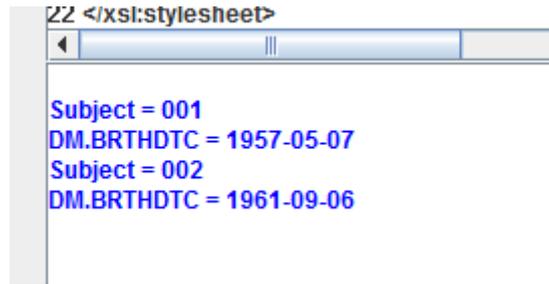
Dragging and dropping the item from the tree into the cell "DM.BRTHDTC" results in the mapping:

```

Mapping Description
SDTM-ETL mapping for DM.BRTHDTC

The Transformation Script
# Mapping using ODM element ItemData with ItemOID I_BRTHDT
$DM.BRTHDTC = xpath (/StudyEventData [@StudyEventOID='BASELINE']/FormData [@FormOID='F_BASELINE']/ItemGroupD
  
```

and doing a "local" quick test of this mapping results in:

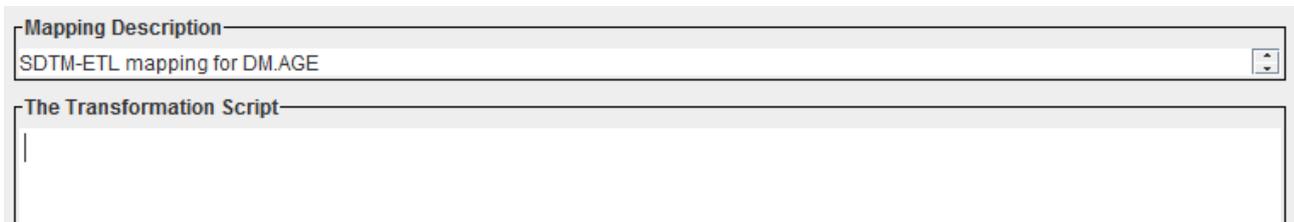


Or executing the mapping for all SDTM variables in the DM that we mapped sofar in:

JID	SUBJID	DM.RFSTDTC	DM.RFENDTC	DM.RFXSTDTC	DM.RFXENDTC	DM.RFICDTC	DM.RFENDTC	DM.BRTHDTC
	001	2010-02-27T10:27:...	2010-03-13	2010-02-27T10:27:...	2010-03-13	2010-02-27T10:27:...	2010-03-13	1957-05-07
	002	2010-02-28T14:33:...	2010-03-16	2010-02-28T14:33:...	2010-03-16	2010-02-28T14:33:...	2010-03-16	1961-09-06

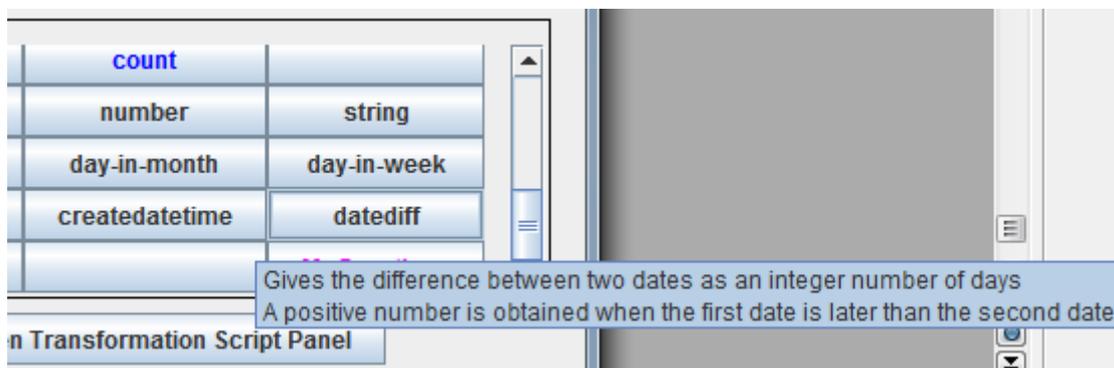
The next variable that needs to be mapped is "AGE". However, it looks as the age of the subject was not collected directly, so we need to calculate it from the birth date and the reference start date.

Just double-click the cell "AGE" to start the mapping process:



As the birth date (\$DM.BRTHDTC) and the reference start date (\$DM.RFSTDTC) were already mapped before, we can reuse them, but in only in "read mode".

Now look into the lower part of the mapping screen, where the "Scripting Language Functions" are displayed. If we scroll down, we find:



So we can use the function "datediff()" to calculate the difference (in number of days) between

reference start date and the birth date. If the result is then divided by 365.2 (the average number of days in a year), then the age in years is obtained. So the mapping script becomes:

```

Mapping Description
SDTM-ETL mapping for DM.AGE

The Transformation Script
$DM.AGE = datediff($DM.RFSTDTC,$DM.BRTHDTC) / 365.2;

```

and executing the mapping for the whole domain¹⁸ results in:

DM.RFICDTC	DM.RFPENDTC	DM.BRTHDTC	DM.AGE
2010-02-27T10:27:...	2010-03-13	1957-05-07	52.8176341730558...
2010-02-28T14:33:...	2010-03-16	1961-09-06	48.48576122672508

which is ... not entirely what we want, as we would like to obtain an integer number.

If we look again to the available functions, we find:

Scripting Language Functions					
min	max	avg	sum	count	
ceiling	floor	round	modulus	number	string
date	year	returns the largest integer that is less than or equal to the numeric value of the argument			
time	hour-in-day	minute-in-hour	second-in-minute	createdatetime	datediff

with the "floor()" function delivering what we want. So the mapping is adapted to:

```

The Transformation Script
$TEMP = datediff($DM.RFSTDTC,$DM.BRTHDTC) / 365.2;
$DM.AGE = floor($TEMP);

```

Resulting in:

DM.RFICDTC	DM.RFPENDTC	DM.BRTHDTC	DM.AGE
2010-02-27T10:27:...	2010-03-13	1957-05-07	52
2010-02-28T14:33:...	2010-03-16	1961-09-06	48

which is exactly what we want.

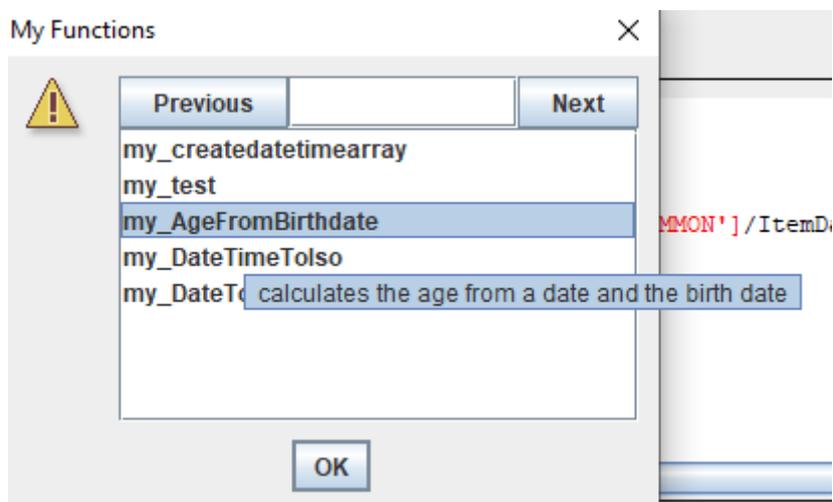
This is where we come to "user functions". How to develop these is explained in another tutorial. These functions reside in the file "functions.xsl" in the folder "stylesheets" and need to start with "my_". The distribution already comes with an example user function "my_AgeFromBirthdate". It can be found under the button "My Functions" in the mapping editor":

¹⁸We can not do a "local" testing, as the variables "DM.RFSTDTC" and "DM.BRTHDTC" are out of scope, as they have been defined in previous mappings.

<	>	<=	>=	==	!=	
contains	starts-with	ends-with	matches	not		
abs	sqrt	log	log10	exp	exp10	
min	max	avg	sum	count	is-a-number	
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	more date/time ...	RESTful WS	My Functions	

Test - Transform to XSLT Hide Upper Panel Full-screen Transformation Script Panel

when clicked, we find:



And we can use it in the mapping script, e.g.:

```

The Transformation Script
1 $DM.AGE = my_AgeFromBirthdate($DM.RFSTDTC,$DM.BRTHDTC);
2
3

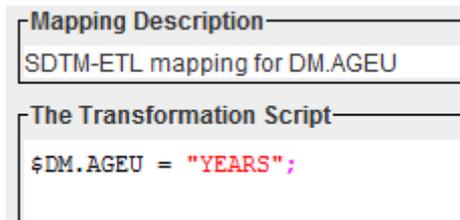
```

Development of user functions is relatively easy, and is very frequently done by our customers¹⁹.

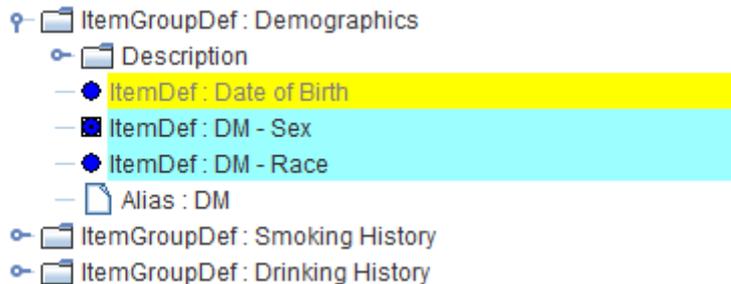
This kind of calculations should be the exception in SDTM, as SDTM is about collected data and not about derived data. Unfortunately, derivations have sneaked in into SDTM in the last years, as the tools of the regulatory authorities are not able to calculate them "on the fly" from the already available data. A typical example are all the --DY variables.

The next SDTM variable is "AGEU". In our case it just is the string "YEARS". So the mapping is:

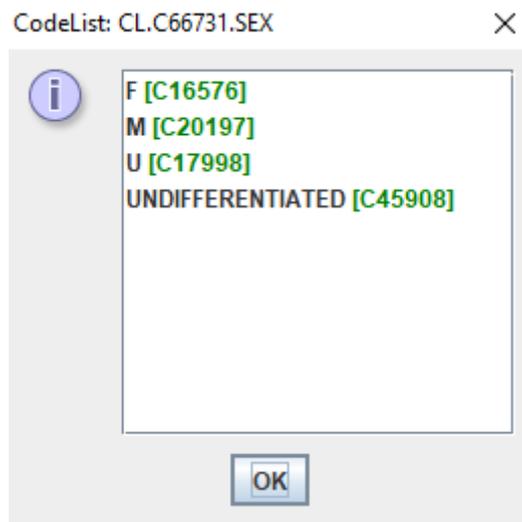
¹⁹ We do of course deliver services for development of user functions.



For "SEX", we once again first look for a "hot candidate" and find:



It is seen that the "traffic light" is blue, meaning that the variable is under controlled terminology. The information about the SDTM controlled terminology can be obtained using the menu "View – SDTM associated codelist" which delivers:

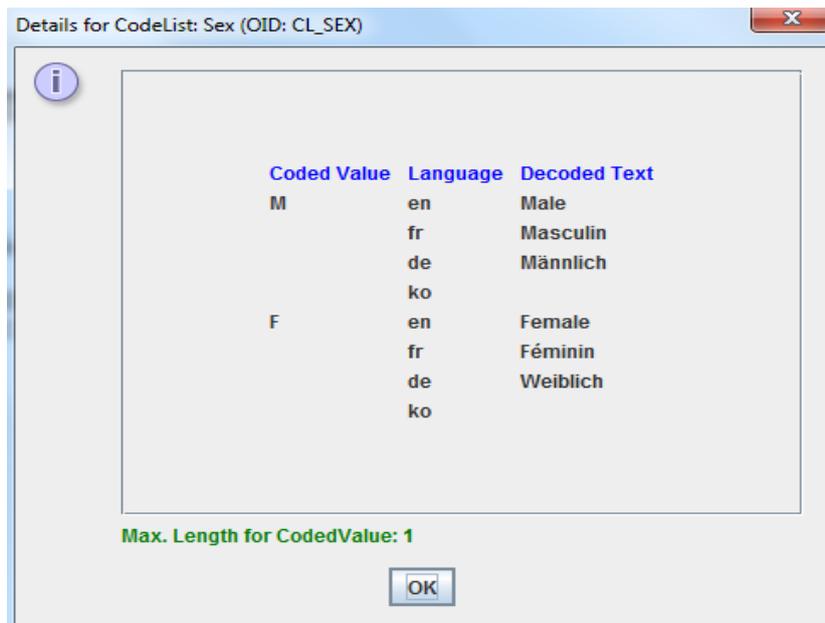


standing for "female", "male", "unknown" and "undifferentiated" (intersex)²⁰.

The codes in brackets in green provide the CDISC/NCI codes, which become ever more important in how CDISC maintains its codelists.

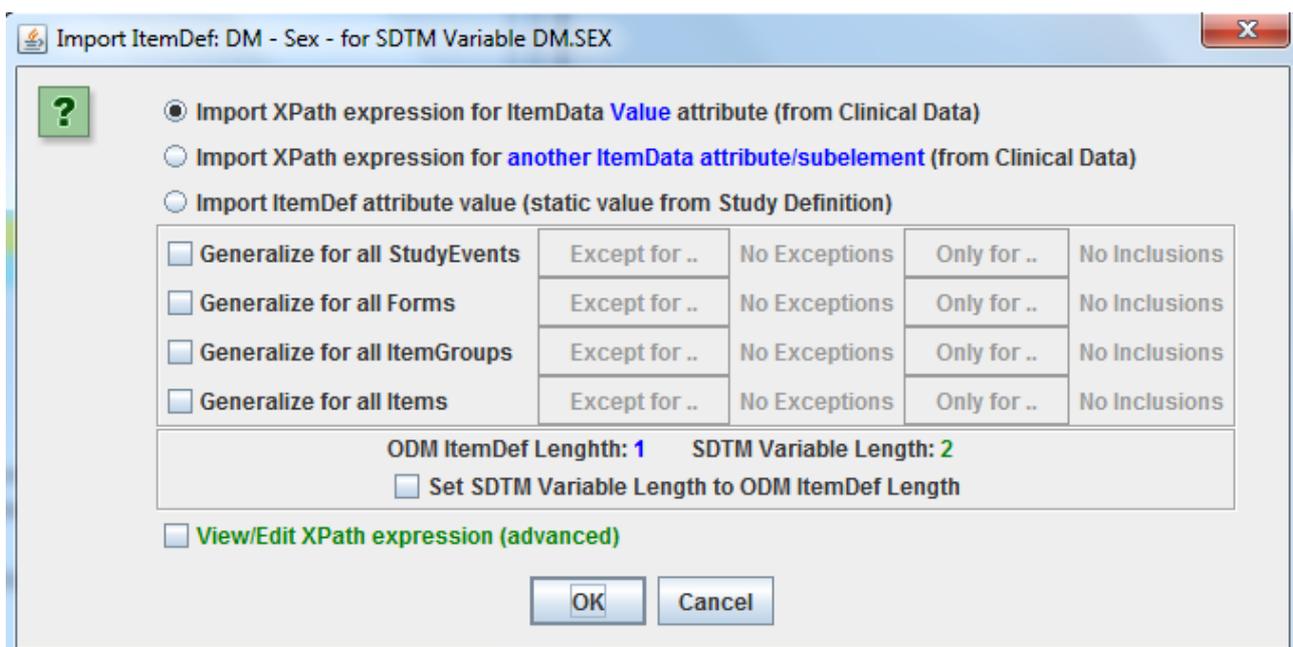
Also on the ODM side, there is an associated codelist. Selecting the item "Sex" and using the menu "View – Item CodeList details" provides a dialog:

²⁰See the published CDISC controlled terminology lists published by NCI

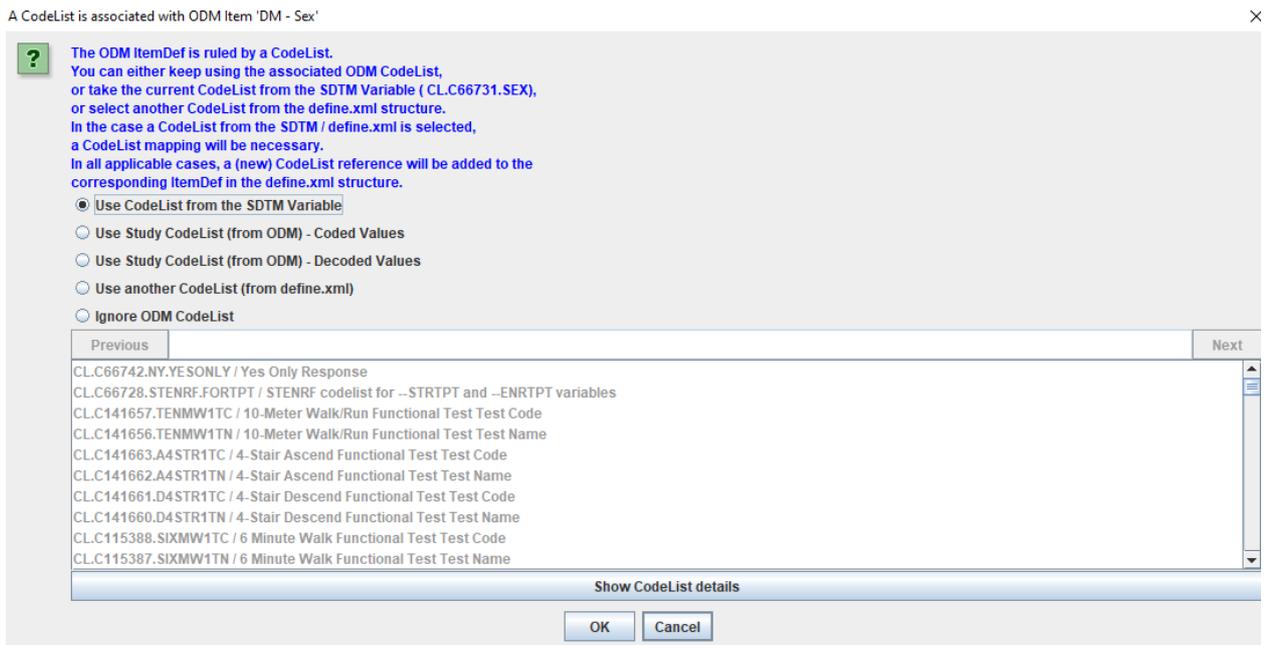


stating that in the ODM, only the values "M" and "F" are foreseen.

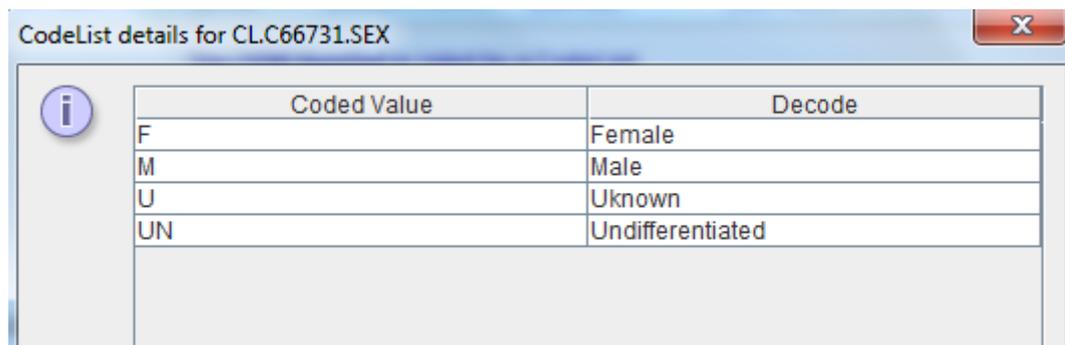
Drag-and-drop from the item "Sex" in the study design tree to the SDTM cell "DM.SEX" displays the wizard:



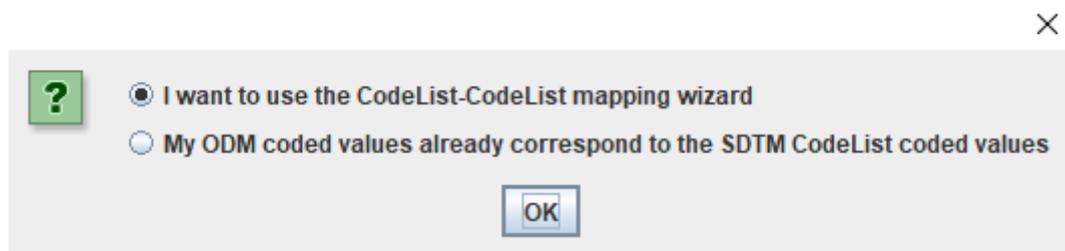
and then clicking "OK" leads to the following dialog:



asking whether we want to use the ODM codelist (coded or decoded values), the currently to DM.SEX associated codelist, or another list from the SDTM define.xml list. We want to use the SDTM codelist, so we select "Use codelist from the SDTM Variable". We can quickly inspect that codelist using the "Show CodeList Details" button:



After clicking the "OK" button in the "A CodeList is associated" wizard, a choice dialog²¹ is displayed:



Mostly, you will want to use the "mapping wizard", unless your (coded) values in the ODM are already "F", "M", etc.. This will especially be the case when CDASH forms have been used with CDISC controlled terminology already in the study design.

Not only just for the exercise, but as we also want to have "U" (Unknown) in the result, which is not in the ODM, we want to use the "mapping wizard", leading to:

²¹ This is new as of SDTM-ETL version 4.1.

ODM CodeList Item	SDTM CodeList Item	
<input type="checkbox"/> Show ODM decoded values		
M	F	Search
F	F	Search
MISSING VALUE	F	Search

Generate subset codelist from selected SDTM items, and assign to the SDTM variable **DM.SEX**
 Adapt variable Length for longest CodeList item
 Add comment line to each mapping
 Except for items already mapped
 Also use CDISC Synonym List
 Also use Company Synonym List

Use SDTM decoded value
 Ask to store mappings as synonyms to Company Synonym List

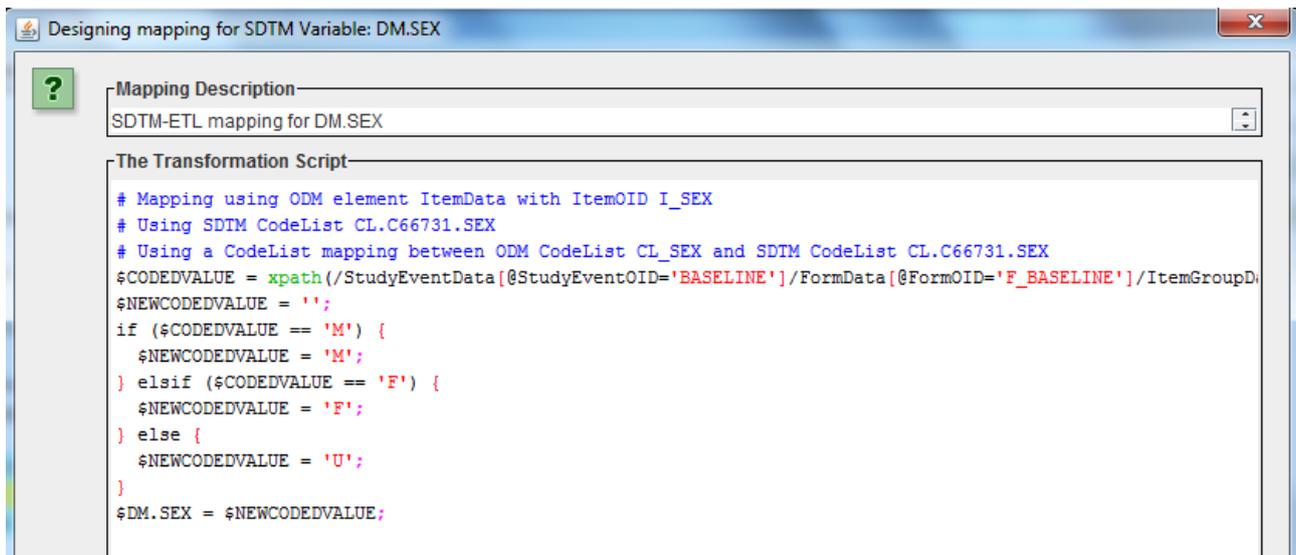
In this case it is easy, and even the wizard will have an easy task finding out – so click the "Attempt 1:1 mapping" button, resulting in a proposal mapping:

ODM CodeList Item	SDTM CodeList Item	
M	M	Search
F	F	Search
		Search

For the empty value, we can add that we want to map this to "U" (for "Unknown"), so we add that by using the dropbox:

ODM CodeList Item	SDTM CodeList Item	
M	M	Search
F	F	Search
	U	Search

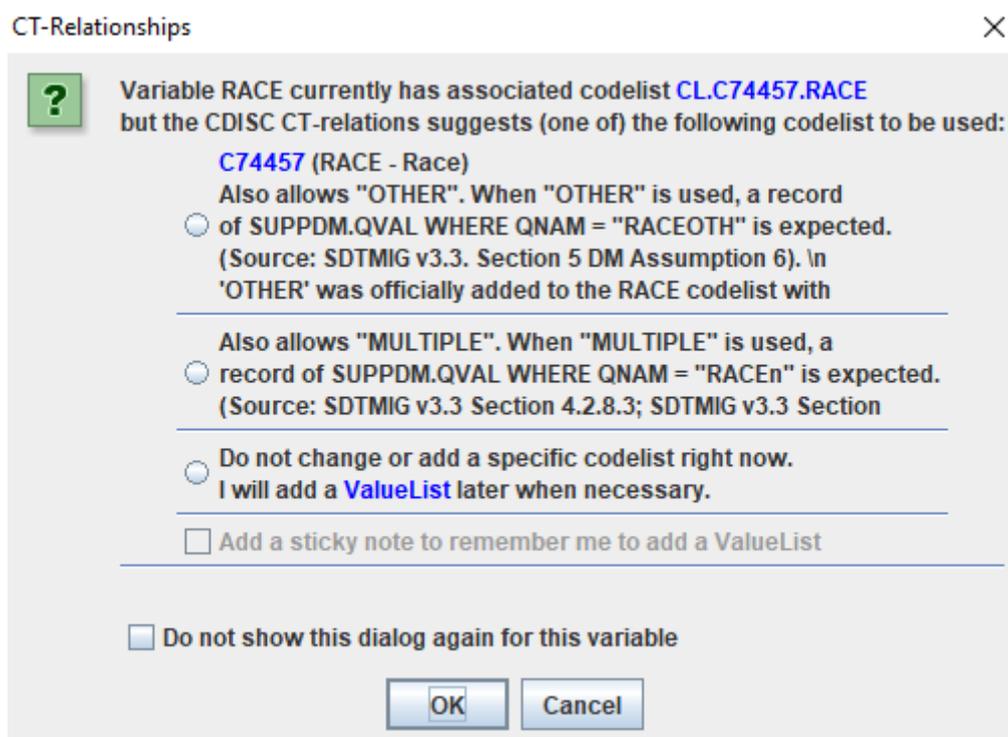
After clicking "OK" the mapping script is completely automatically created:



the "if-elseif-else" construct being generated automatically.

In many cases, wizards will create mapping scripts completely automatically, but the user can always further enhance or change the mapping script manually.

A similar mapping needs to be done for "RACE". Using the menu "Navigate – Find hot SDTM Candidate", the ODM item "Race" is quickly found in the study design tree, and the subsequent drag-and-drop leads to:



and showing some additional possibilities. The reason is that, as of SDTMETL 5.0, the published "[CDISC-CT Relations](#)" has been implemented in the software, pointing the user to "special" situations.

If we do not expect any "MULTIPLE" races, as in the ODM, the value for "Race" was enumerated, we can select the radiobutton "C74457 (RACE - Race) which means that we will use the default codelist for "RACE". Selecting it and then clicking "OK" then leads to the codelist mapping wizard:

?

ODM CodeList Item	SDTM CodeList Item
<input type="checkbox"/> Show ODM decoded values	
CAUCASIAN	AMERICAN INDIAN OR ALASKA NATIVE <input type="button" value="Search"/>
BLACK	AMERICAN INDIAN OR ALASKA NATIVE <input type="button" value="Search"/>
ASIAN	AMERICAN INDIAN OR ALASKA NATIVE <input type="button" value="Search"/>
OTHER	AMERICAN INDIAN OR ALASKA NATIVE <input type="button" value="Search"/>
MISSING VALUE	AMERICAN INDIAN OR ALASKA NATIVE <input type="button" value="Search"/>

Generate subset codelist from selected SDTM items, and assign to the SDTM variable **DM.RACE**

Adapt variable Length for longest CodeList item

Add comment line to each mapping

Except for items already mapped

Also use CDISC Synonym List

Also use Company Synonym List

Use SDTM *decoded* value

Ask to store mappings as synonyms to Company Synonym List

which is easily mapped to:

ODM CodeList Item	SDTM CodeList Item	
<input type="checkbox"/> Show ODM decoded values		
CAUCASIAN	WHITE	<input type="button" value="Search"/>
BLACK	BLACK OR AFRICAN AMERICAN	<input type="button" value="Search"/>
ASIAN	ASIAN	<input type="button" value="Search"/>
OTHER	OTHER	<input type="button" value="Search"/>
MISSING VALUE	UNKNOWN	<input type="button" value="Search"/>

Generate subset codelist from selected SDTM items, and assign to the SDTM variable **DM.RACE**
 Adapt variable Length for longest CodeList item
 Add comment line to each mapping

Except for items already mapped
 Also use CDISC Synonym List
 Also use Company Synonym List

Use SDTM *decoded* value
 Ask to store mappings as synonyms to Company Synonym List

This leads to the automatically generated mapping script:

```

The Transformation Script
1 # Mapping using ODM element ItemData with ItemOID I_RACE
2 # Using SDTM CodeList CL.C74457.RACE
3 # Using a CodeList mapping between ODM CodeList CL_RACE and SDTM CodeList CL.C74457.RACE
4 $CODEDVALUE = xpath(/StudyEventData[@StudyEventOID='BASELINE']/FormData[@FormOID='F_BASELINE']/ItemGroupData[@I
5 if ($CODEDVALUE == 'CAUCASIAN') {
6   $NEWCODEDVALUE = 'WHITE';
7 } elseif ($CODEDVALUE == 'BLACK') {
8   $NEWCODEDVALUE = 'BLACK OR AFRICAN AMERICAN';
9 } elseif ($CODEDVALUE == 'ASIAN') {
10  $NEWCODEDVALUE = 'ASIAN';
11 } elseif ($CODEDVALUE == 'OTHER') {
12  $NEWCODEDVALUE = 'OTHER';
13 } else {
14  $NEWCODEDVALUE = 'UNKNOWN';
15 }
16 $DM.RACE = $NEWCODEDVALUE;
17
18

```

and test the mapping for the whole domain, the following result is obtained (partial view):

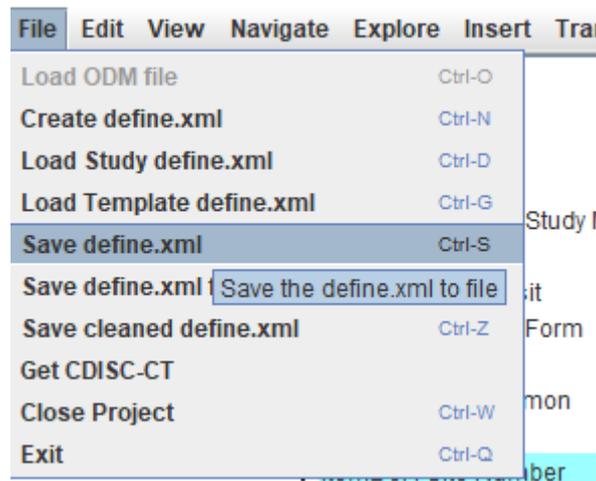
DM.RFPENDTC	DM.BRTHDTC	DM.AGE	DM.AGEU	DM.SEX	DM.RACE
-03-13	1957-05-07	52	YEARS	F	WHITE
-03-16	1961-09-06	48	YEARS	M	WHITE

Remark that we can of course edit/adapt the mapping script in the "mapping editor" even when it was generated automatically.

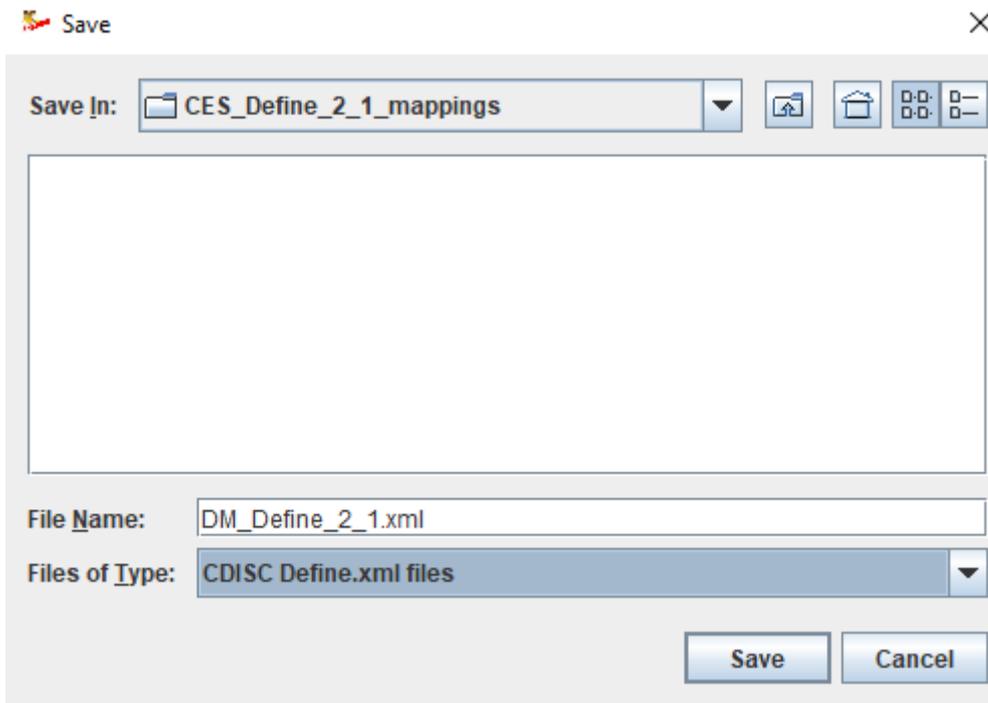
Also remark that "Attempt 1:1 mapping" would not lead to good results, as it is based on word similarity, and as "Caucasian" is similar to "Asian", "Caucasian" would then be mapped to "ASIAN", which is of course not correct.

When in most cases "Caucasian" was used in the CRFs, one can also check the checkbox "ask to store mappings as synonyms to Company Synonyms list". This allows to store the current mappings for reuse in other studies, by using (in the other study) the checkbox "Also use Company Synonyms list" and then use "Attempt 1:1 mapping".

It's not a bad idea to save all the work done so far. This is accomplished by using the menu "File – Save define.xml" (or using CTRL-S):



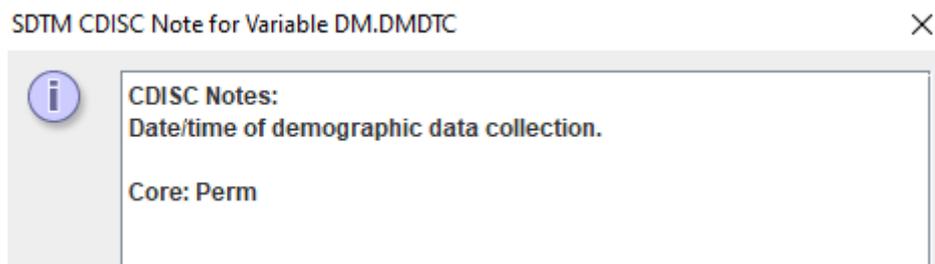
and selecting a location and name for our file, e.g. "DM_define_2_1.xml":



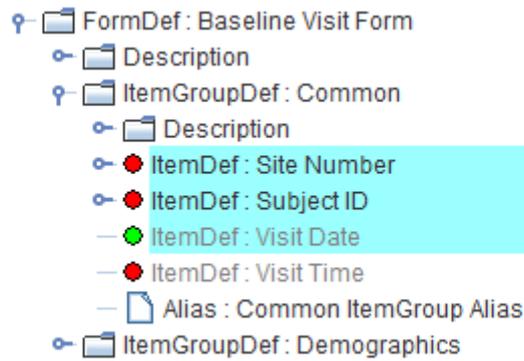
Remark that all the current work is automatically saved every 15 minutes as a define.xml in the folder "define_autosave". The amount of time between such "backups" can be changed by using the "Options" menu. The auto-saved define.xml files can help in recovering e.g. after a computer crash, or when one made mistakes and wants to revert to a prior situation.

The "Country" is fixed in this study. So one can just add \$DM.COUNTRY = 'xxx' where 'xxx' is the three character code in [ISO-3166](#) notation. Examples are: USA (United States of America), CAN (Canada), GER (Germany), AUT (Austria), AUS (Australia).
One should

The next variable is DMDTC. When using CTRL-H, more information is displayed:

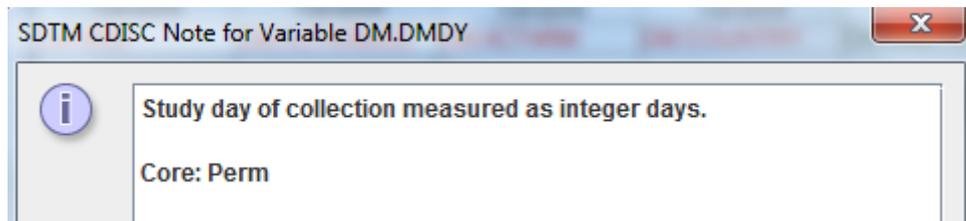


We can just take it as the "Visit Date" for the form where also the demographics data was collected:



In this case, a simple drag-and-drop from the item "Visit date" is all that is needed.

The next one is DMDY:



There is something very special here (see the SDTM-IG). In SDTM, the day the study starts for a specific subject has $xxDY = 1$ (and not 0 as one might think). The day before the study starts however is then not day 0, as one might think, but day -1. So in SDTM, there is no day "0", and $xxDY$ can never have the value "0". Logical, isn't it²²?

So when calculating $xxDY$, we must always add logic in our script to avoid that a value "0" is given as the result. In this case, it is pretty simple – we can even reuse variables that were defined before. For DMDY, we write the mapping:

```
The Transformation Script
$DM.DMDY = datediff($DM.DMDTC, $DM.RFSTDTC);
```

DM.DMDTC and DM.RFSTDTC have been defined before (i.e. more to the left), so we can reuse them in "read only" mode. The "datediff()" function delivers the difference in days. In case the first parameter value is later than the second, a positive (or better said, non-negative) result is obtained. One immediately sees that this can lead to a DMDY=0 result when DMDTC and RFSTDTC are identical (as is the case)²³. So we adapt the mapping to:

²²That was meant in a sceptical way ...

²³Essentially, DMDY should never appear in SDTM, as SDTM is about collected data, not about derived data. The tools of the FDA should do these kind of calculations.

```

Mapping Description
SDTM-ETL mapping for DM.DMDY

The Transformation Script

$TEMP = datediff($DM.DMDTC,$DM.RFSTDTC);
$TEMPNUMBER = number($TEMP);
if($TEMPNUMBER >= 0) {
    $DM.DMDY = $TEMPNUMBER + 1;
} else {
    $DM.DMDY = $TEMPNUMBER;
}

```

There is one peculiarity in this script: the "datediff" function essentially returns a string²⁴, which need to be transformed into a number (kind of casting) in order to do mathematical calculations with it.

The result for our two subjects is:

DM.AGE	DM.AGEU	DM.SEX	DM.RACE	DM.DMDTC	DM.DMDY
	YEARS	F	WHITE	2010-02-27	1
	YEARS	M	WHITE	2010-02-28	1

In the next tutorial, we will work on the SV (subject visits) domain, and also introduce alternative output formats.

²⁴The reason for this is that in XSLT, a datediff returns a duration, e.g. "P1D" meaning a period of 1 day.