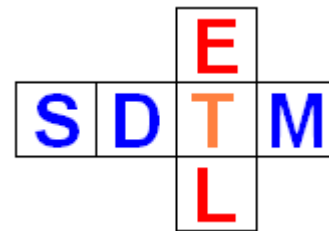# SDTM-ETL 4.x User Manual and Tutorial

Author: Jozef Aerts, XML4Pharma

Last update: 2022-06-19

Creating mappings for the SV (Subject Visits) domain

Although Fred Wood, one of the authors of the SDTM Implementation Guides has once stated "Creating Trial Design tables can be as much of an art as it is a science", it is pretty easy when using SDTM-ETL. Essentially, a "visit" is a "StudyEvent" in ODM. So if we map "StudyEvent" to "VISITNUM" (which is the key in the corresponding TV domain), we are essentially done. If there are repeating visits (as in many oncology studies), one however need to do a bit more.

For our "CES" study, the study design tree is:
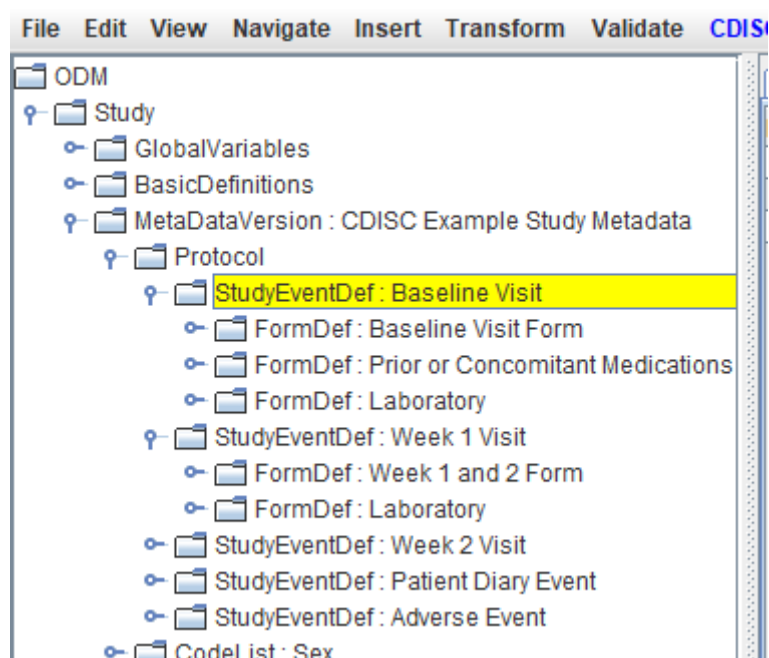
containing 5 planned visits. However, is the visit "Adverse Event" really a planned visit? In the message pane at the bottom of the screen we see that the form „Adverse Event Form (ACRO)" is "repeating" so this lets us doubt a bit. It could be that this "StudyEvent" is just an "artificial" one that is "triggered" when an adverse event is observed, and is meant to collect all adverse events independent from when they occurred. So we need to look into the protocol …
Our suspicion is further strengthened by the design of the StudyEvent:
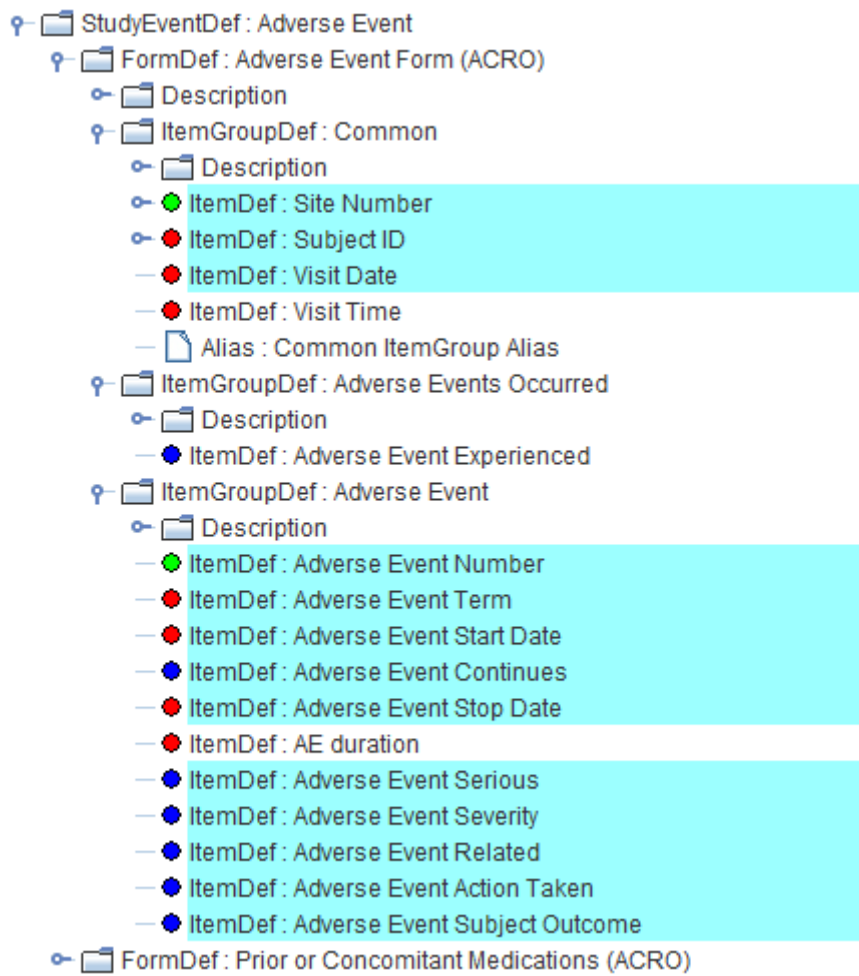
We will further develop the mapping for SV assuming that the StudyEvent "Adverse Events" is not a planned visit, and later (when generating the AE domain mapping) explain how VISITNUM must then be handled there (as an "unplanned visit").

First drag and drop the "SV" row from the template to the bottom of the existing table. The following dialog is displayed:
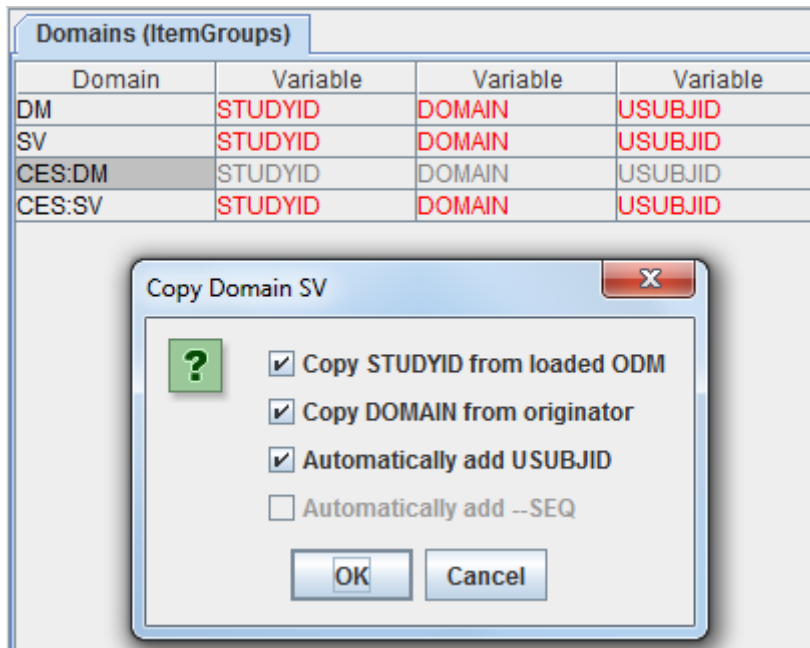
and accept the choices already checked by the wizard, as we want to have the mappings for STUDYID, DOMAIN and USUBJID be generated automatically.

Clicking "OK" results is a new row in our table[1]:

In domains where there is more than one record, we must <u>first</u> define the looping structure. This means that we need to define over which of the SDTM variables an iteration will be done in order to generate the data sets. It often corresponds to the statement in the SDTM-IG like "on record per study per visit".

Most of the domains in the template have already a proposed looping structure, but one should **always** inspect it and adapt it if necessary to its own study structure.

Viewing and editing the properties (like the looping structure) of the domain can be either done by a double-click of the first cell of that domain (in this case the cell with "CES:SV") or by selecting any cell in the row and then using the menu "Edit – SDTM Domain Properties". This results in the following dialog:

---

[1]We have hidden all other rows from the template using the menu "View – View/Hide Domains"

and at the bottom, after having clicked the "validate" button:

stating "one record per VISITNUM per USUBJID".

As this is exactly the structure that is needed, it is not necessary to change the looping structure. We do however still need to set the domain keys. In the past (pre-SDTMIG-3.2), the SDTM-IG often made a proposal for the domain keys, which was copied by implementers without any considerations that the proposal might not be what is needed.

Click the button "Set domain keys and sequence". The following dialog is displayed:
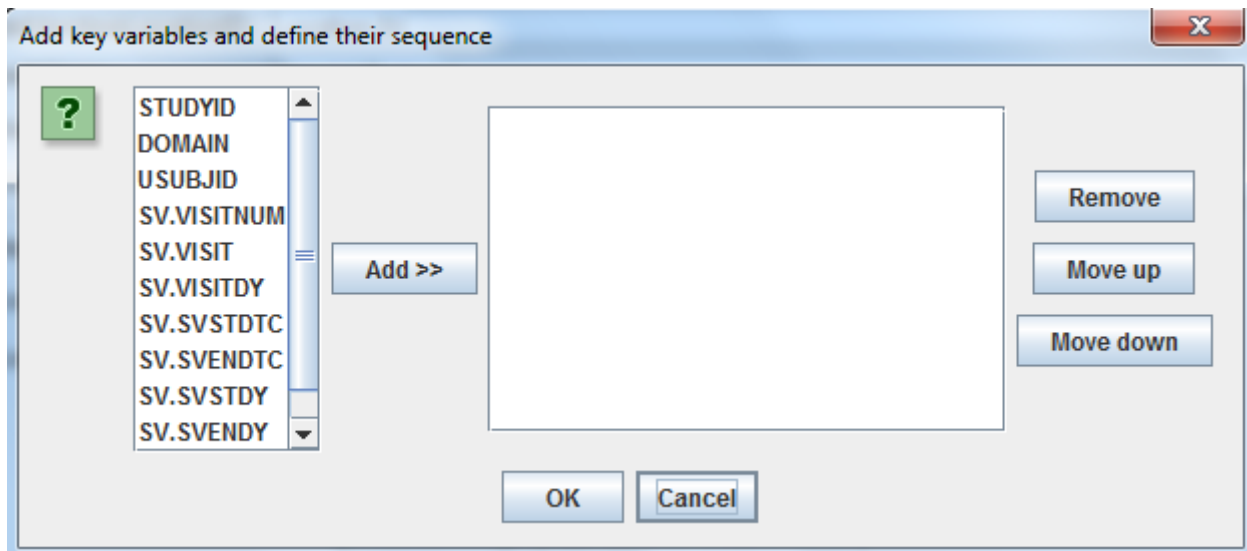
One can now add variables that are the "natural keys" of the domain (see the SDTM-IG 3.2, section "Assigning Natural Keys in the Metadata"). In our case, most logical is of course:
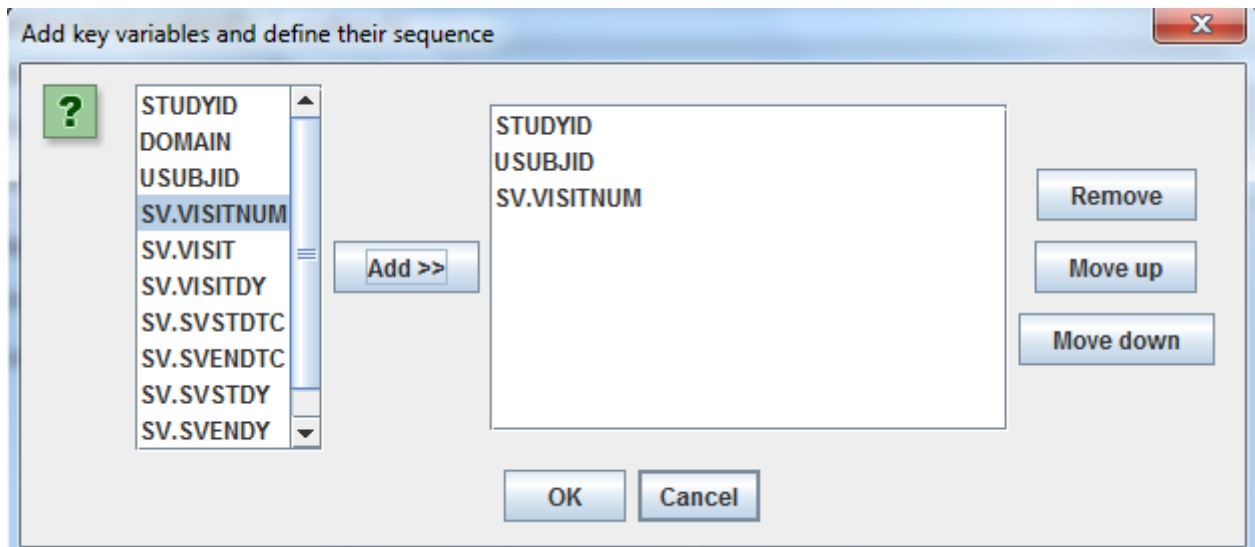

This choice will generate "KeySequence" attributes in the underlying define.xml structure for "STUDYID", "USUBJID" and "VISITNUM" ItemRef elements in the ItemGroupDef "SV".

Click "OK" until returning into the main window.

In the study design tree, collapse or expand all tree nodes until one sees the StudyEvent nodes:
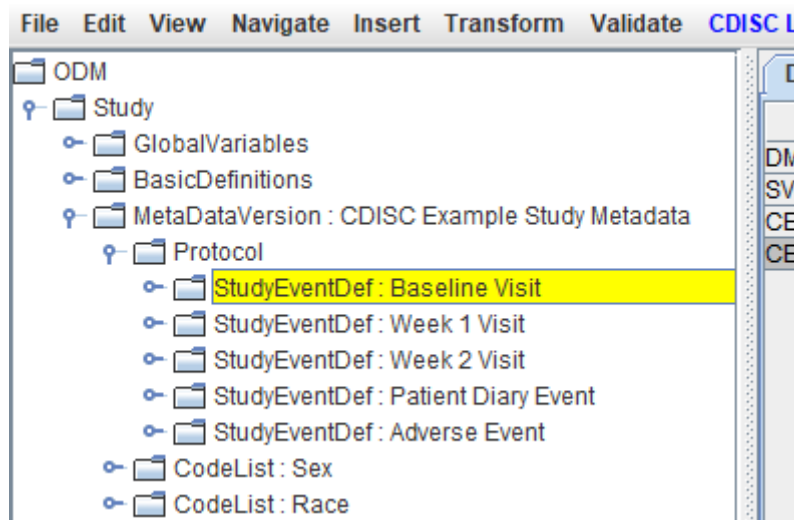
Now drag-and-drop one of the StudyEvent nodes (it doesn't matter which one. except for the "Adverse Event" one) to the cell "SV.VISITNUM". The following wizard is displayed:
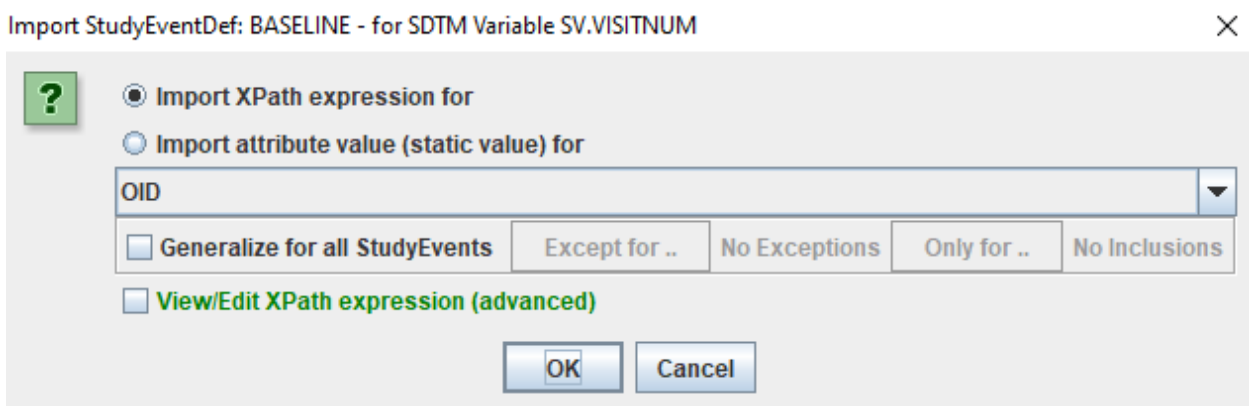
This selects a single StudyEvent, but we want the mapping to be applicable to all StudyEvents, except for the "Adverse Events" StudyEvent, as the latter is not a real planned visit in the sense of SDTM. So we check the checkbox "Generalize for all StudyEvents":
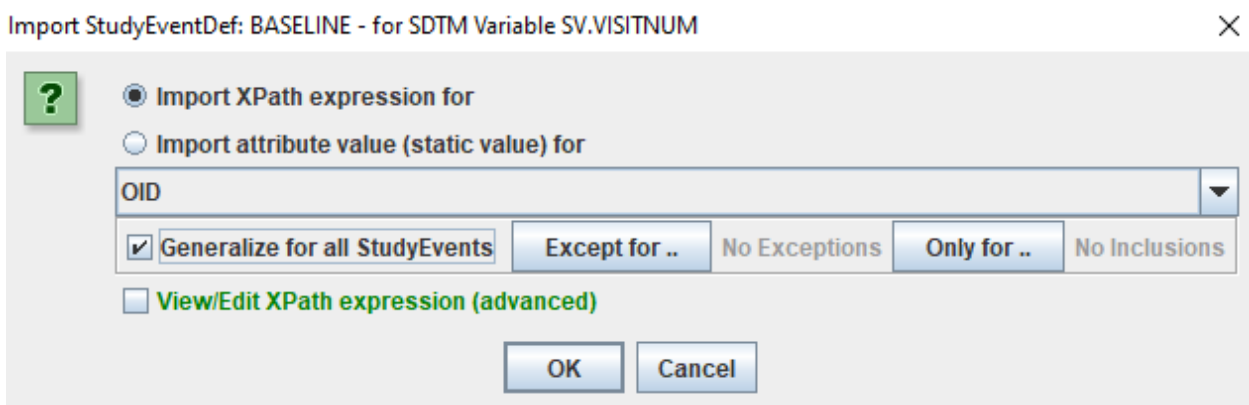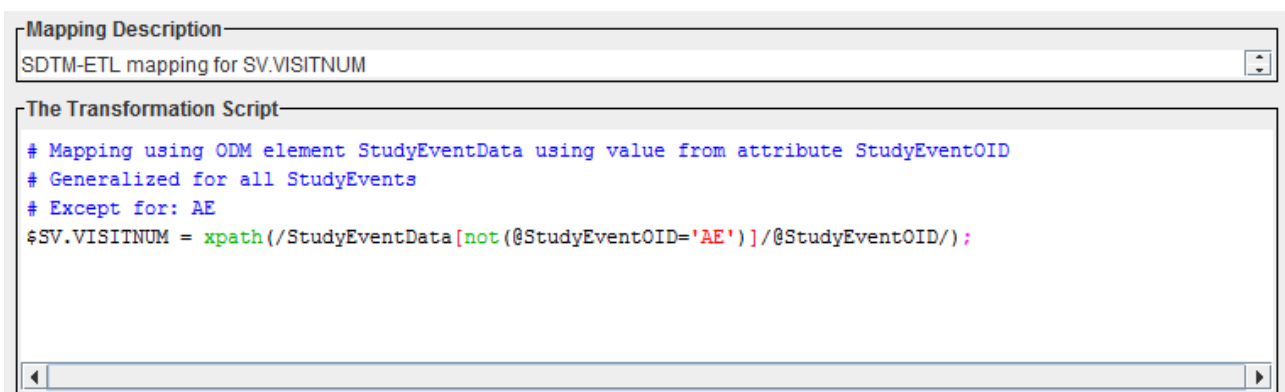

which makes the buttons "Except for …" and "Only for ..." available. As we need to exclude the "Adverse StudyEvent" StudyEvent, we need to click "Except for ..." and

check "AE- Adverse Event":
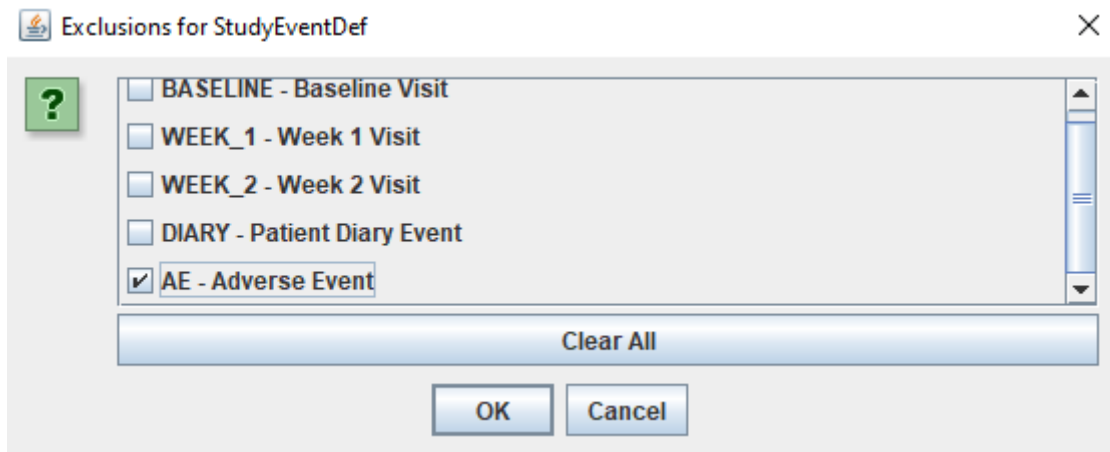
Clicking "OK" several times generates the mapping script:

stating that we will consider all "StudyEventData" in the ODM "ClinicalData" file except for the one with the StudyEventOID "AE".

Testing this out on our test set with clinical data for two subjects leads to:

STUDYID	DOMAIN	USUBJID	SV.VISITNUM
CES	SV	001	BASELINE
CES	SV	001	WEEK_1
CES	SV	001	WEEK_2
CES	SV	001	DIARY
CES	SV	002	BASELINE
CES	SV	002	WEEK_1
CES	SV	002	WEEK_2
CES	SV	002	DIARY

which is not what is wanted[2] as we need a number for VISITNUM. So we still need to assign numbers to each of our visits. This can be done by a little amount of editing of the mapping script, e.g. like:

---

[2]This would however already be a perfect mapping for the variable "VISIT" (visit name)

```
┌Mapping Description─────────────────────────────────────────────
│ SDTM-ETL mapping for SV.VISITNUM
├The Transformation Script──────────────────────────────────────
│ # Mapping using ODM element StudyEventData using value from attribute StudyEventOID
│ # Generalized for all StudyEvents
│ # Except for: AE
│ $TEMP = xpath(/StudyEventData[not(@StudyEventOID='AE')]/@StudyEventOID/);
│ if($TEMP == 'BASELINE') {
│       $SV.VISITNUM = 1;
│ } elsif ($TEMP == 'WEEK_1') {
│       $SV.VISITNUM = 2;
│ } elsif ($TEMP == 'WEEK_2') {
│       $SV.VISITNUM = 3;
│ } elsif ($TEMP == 'DIARY') {
│       $SV.VISITNUM = 4;
│ }
```

Also note that the comment lines (starting with a '#') have been generated automatically – you can of course edit them.

At this point, one may also want to change the text in the "Mapping Description", as if one later states that VISITNUM is "derived", the text from the "Mapping Description will appear in the "method description" of the define.xml. So one could e.g. change it into "Derived from the visit name".

And testing on the clinical data then leads to:

STUDYID	DOMAIN	USUBJID	SV.VISITNUM
CES	SV	001	1
CES	SV	001	2
CES	SV	001	3
CES	SV	001	4
CES	SV	002	1
CES	SV	002	2
CES	SV	002	3
CES	SV	002	4

Remark: the assignment of the "visit number" is completely arbitrary, but the order needs to correspond to the "natural order" given in the protocol, i.e. that the "baseline visit" normally comes before the "Week 1" visit etc.. However, the numbers must not be subsequent by default: so if one would assign them as being "100", "200", "300" and "400", this would be perfectly OK (also see the remark about repeating visits later in this chapter). They however MUST correspond to VISITNUM in the records of the "trial visits" (TV) domain[3].

For the variable "VISIT" visit name, we can just copy-paste the first line of the script for VISITNUM and add an extra line for the assignment to "VISIT":

---

[3] As of SDTM-ETL v.4.1, it is possible to generate the TV dataset first with an automatically generated codelist with visit number and names. This can assure that both datasets are aligned.

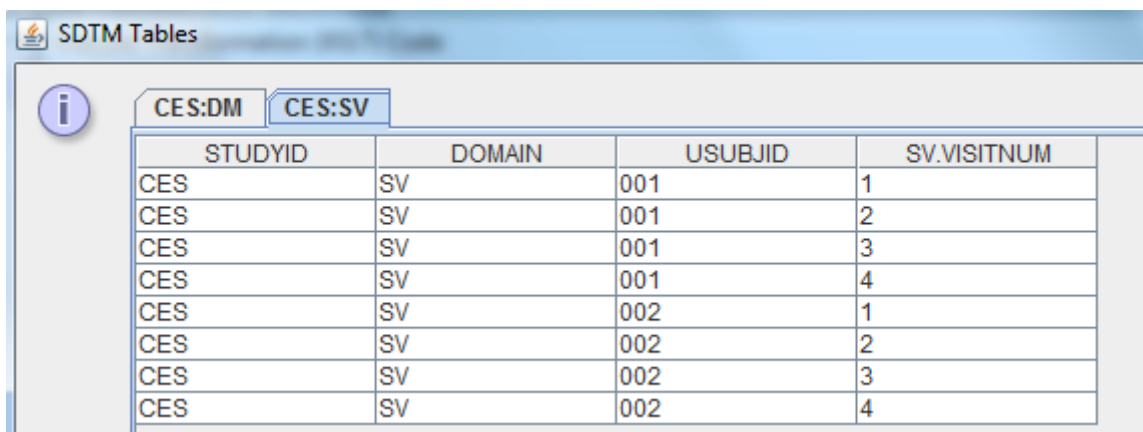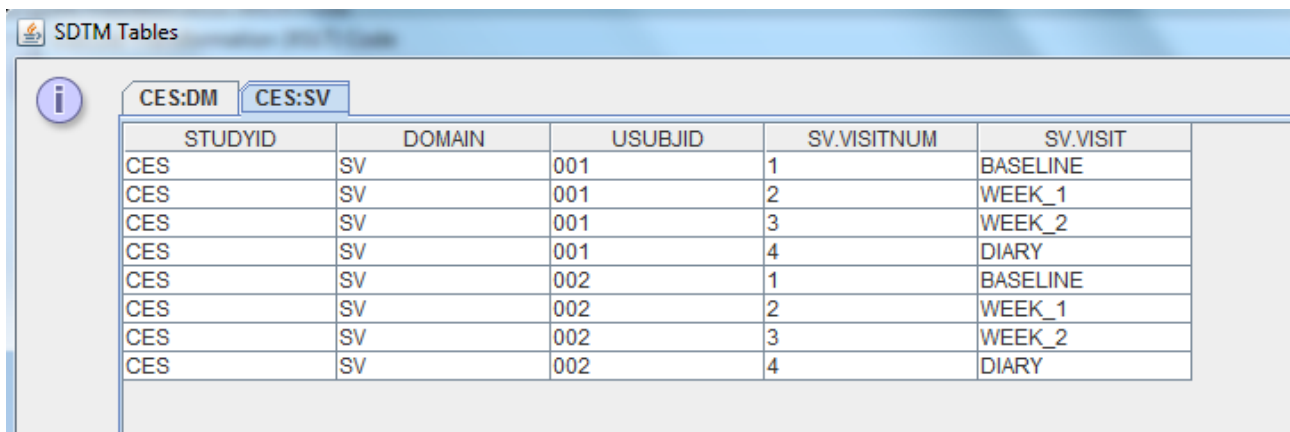**Mapping Description**

SDTM-ETL mapping for SV.VISIT

The Transformation Script

```
# Mapping using ODM element StudyEventData using value from attribute StudyEventOID
# Generalized for all StudyEvents
# Except for: AE
$TEMP = xpath(/StudyEventData[not(@StudyEventOID='AE')]/@StudyEventOID/);
$SV.VISIT = $TEMP;
```

which leads to following results:

STUDYID	DOMAIN	USUBJID	SV.VISITNUM	SV.VISIT
CES	SV	001	1	BASELINE
CES	SV	001	2	WEEK_1
CES	SV	001	3	WEEK_2
CES	SV	001	4	DIARY
CES	SV	002	1	BASELINE
CES	SV	002	2	WEEK_1
CES	SV	002	3	WEEK_2
CES	SV	002	4	DIARY

If required, one can replace the underscore in string by simply using the "replace()" function of SDTM-ETL in the mapping script.

We can now also populate SVSTDTC, SVENDTC, SVSTDY and SVENDY.
Note that these variables are permissible and should only be populated in case they were collected.
So one should not try to derive them from other information.
In our case, the visit date was however collected as "Visit Date" (OID "I_VISIT") in the ODM, and this is done in the same way in each visit:
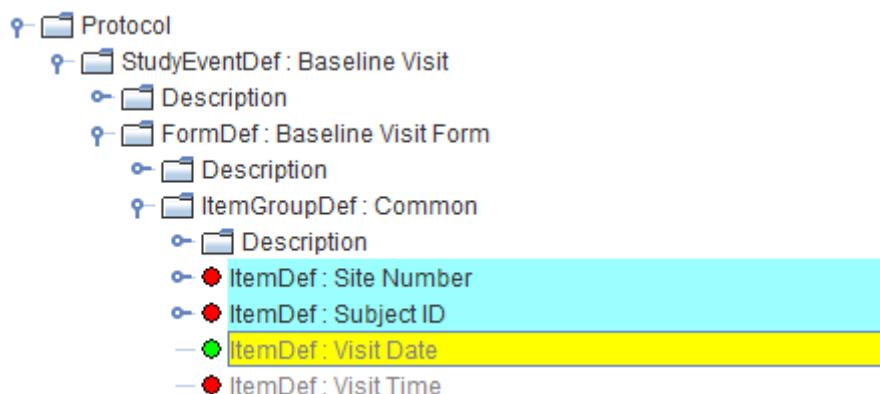
So drag-and-drop the item "Visit Date" to the cell "SV.SVSTDTC". The dialog shows up again asking what we want to do exactly:
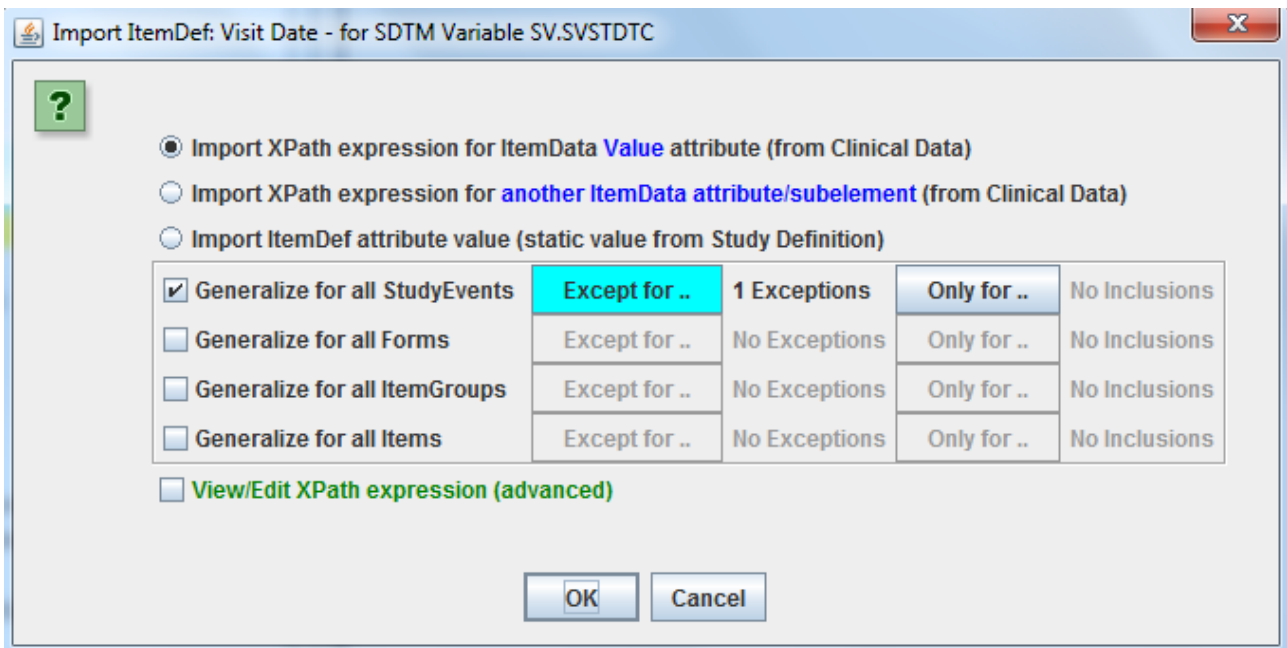
The information about "generalizing for all StudyEvents" and the exception for the "Adverse Event" is still there. If it isn't, one should add it again. This leads to the mapping:
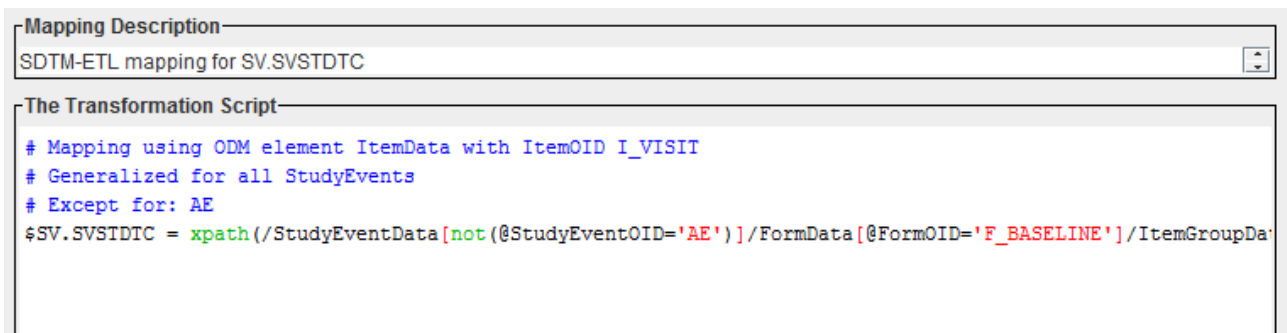

stating: "take all StudyEventData elements (except for the one with StudyEventOID "AE"), and of these, take the FormData element with OID "F_BASELINE" …
Wait a minute: there is no form "F_BASELINE" in the StudyEvent "WEEK_1"! So we need to find something better.
If we use the menu "Navigate – Next Instance" we see that "Visit Date" occurs in every visit, but in different forms, but always in the ItemGroup "Common". However, it can also occur several times in different forms in the same visit, so we must always take the first occurrence.

We do start again from the drag-and-drop and get:
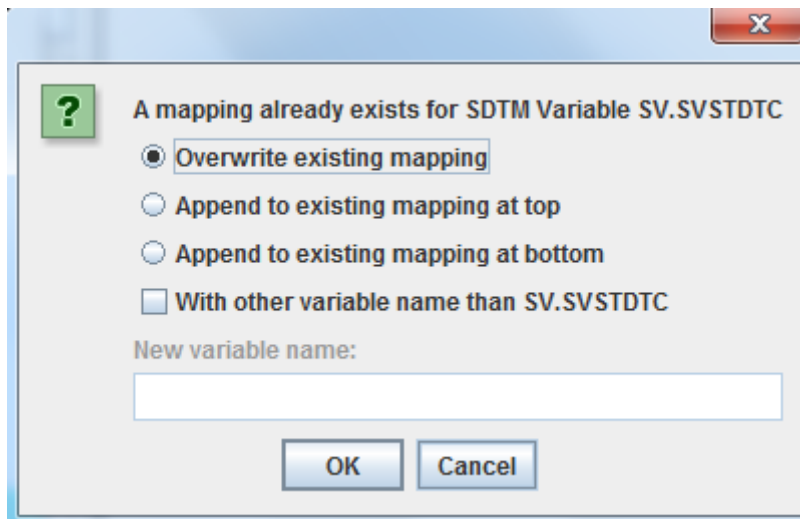
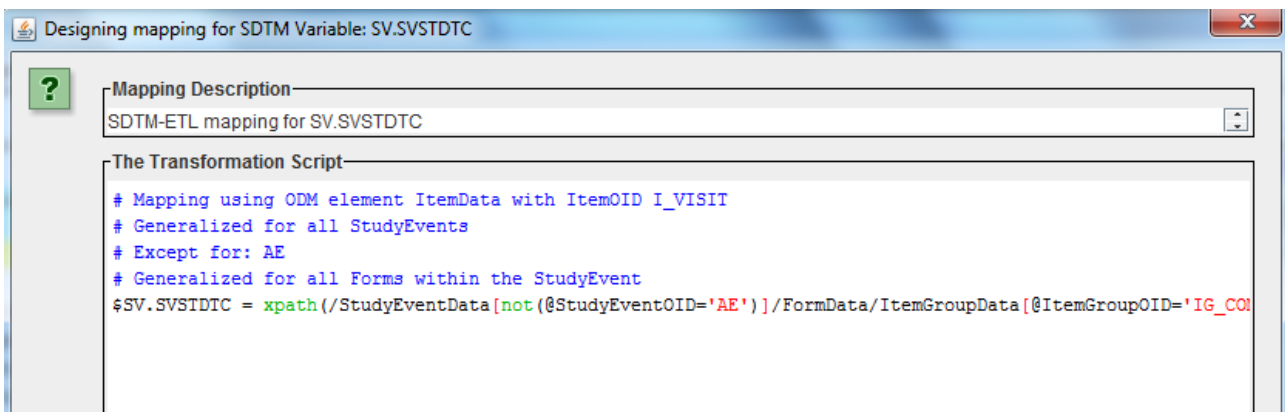Check the checkbox "Overwrite existing mapping". This leads again to:
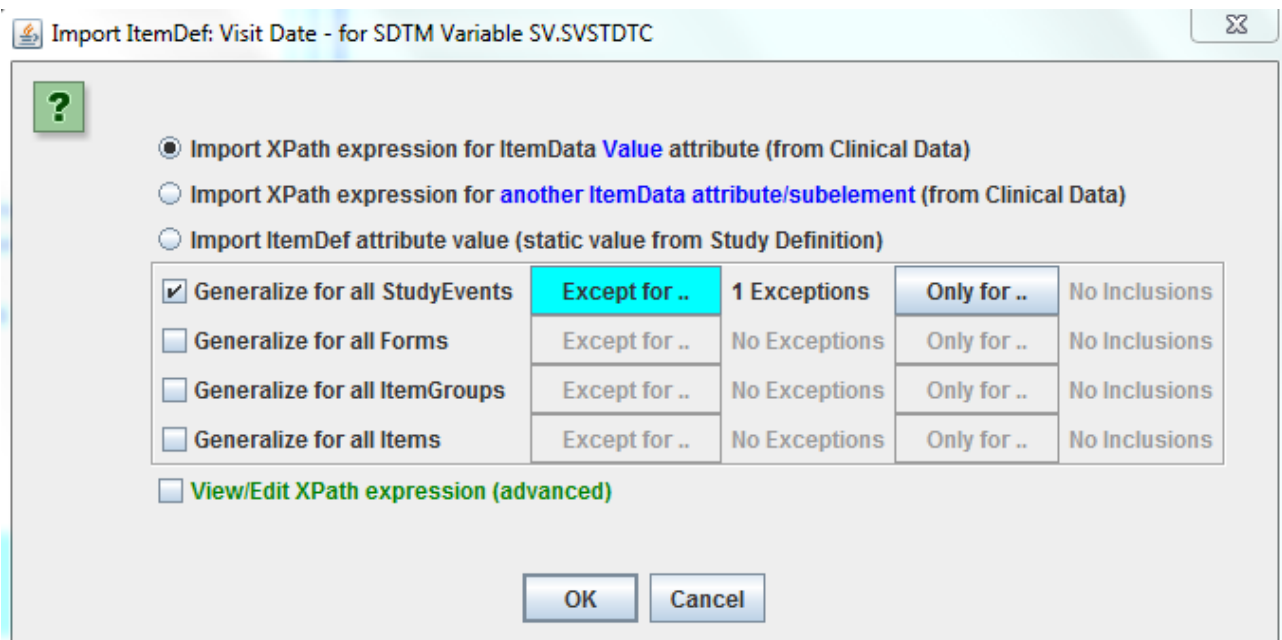
and now also check the checkbox "Generalize for all forms", stating that the "Visit date" can occur in any form[4]:

---

[4]If there is no "Visit Date" in every form, we should here again exclude the forms that haven't got the "Visit date", or using the button "Only for" include the ones that have.
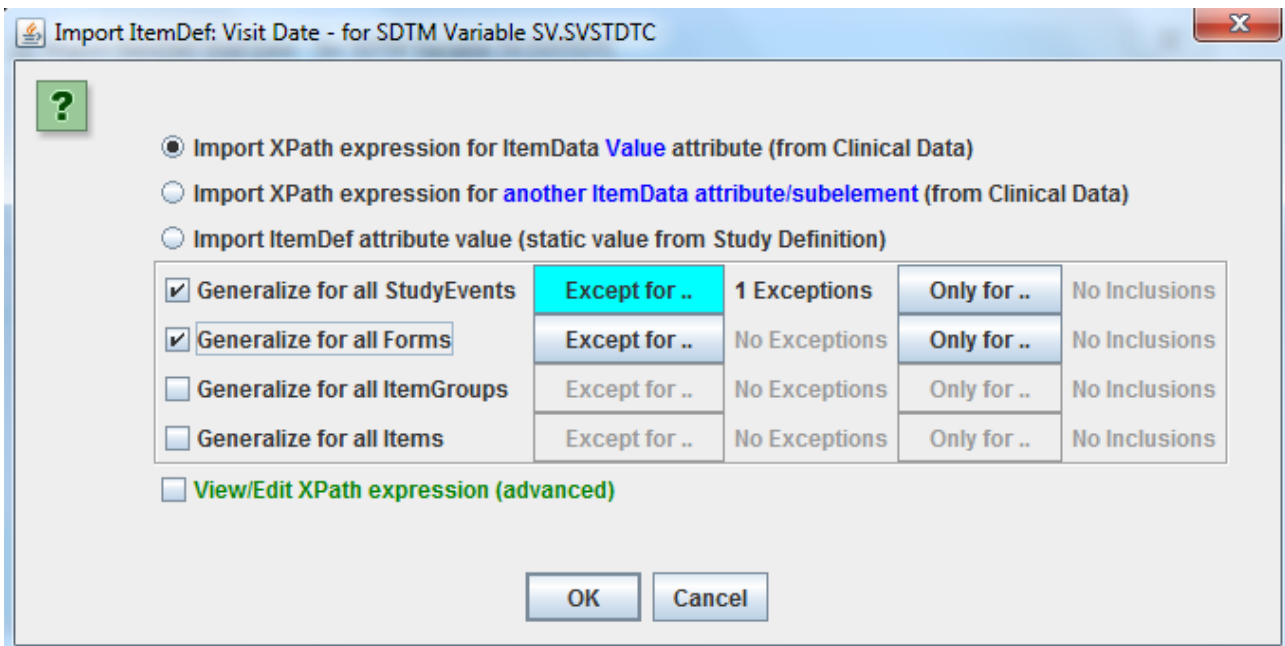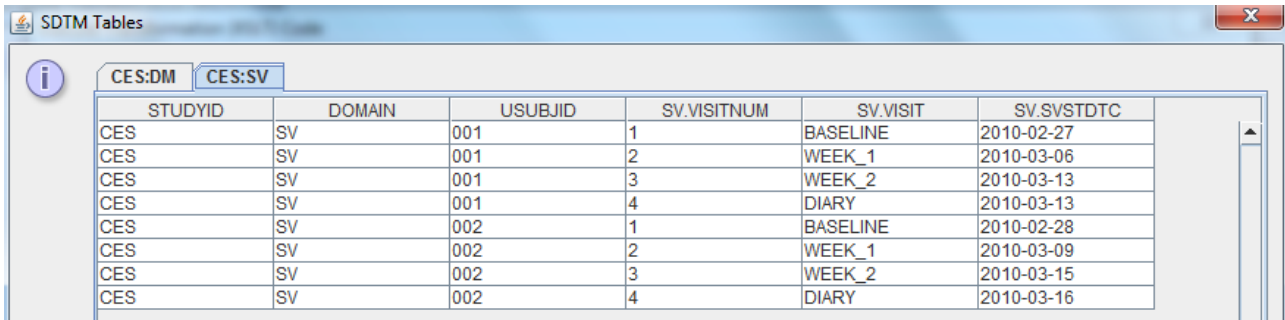
leading to:

If we would now execute all the mappings for the domain, we would get an error ("A sequence of more than one item is not allowed as the first argument of string()"). Reason for this is that the visit date, as we have defined it here, occurs several times in the same visit (i.e. on different forms). We do however only want the first occurrence. So we rewrite the mapping script as:
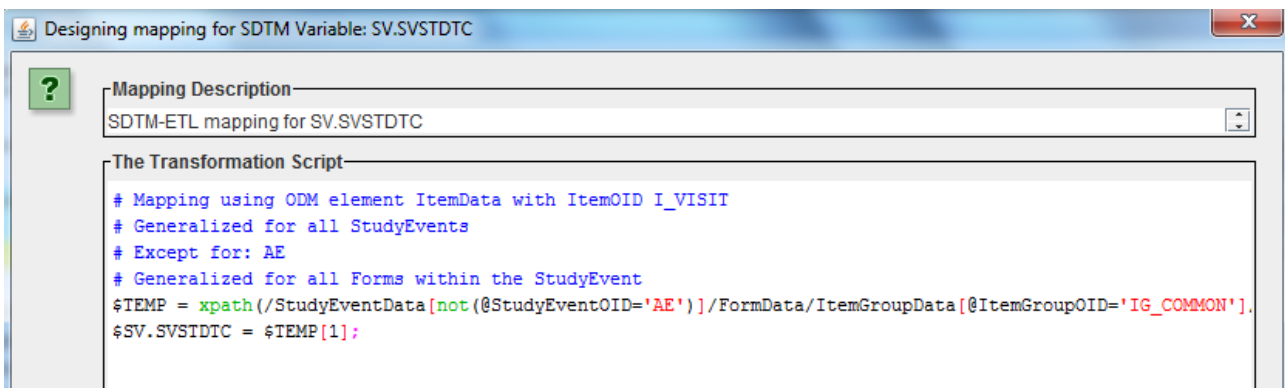
In our scripting language, a pair of square brackets means a condition. The condition [1] means "take the first one", [2] means "take the second one" and [last()] means "take the last occurring
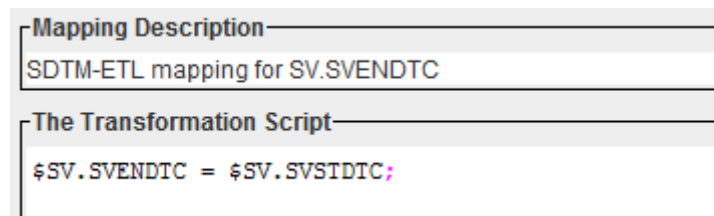
one".

This leads to the following results:

The same can now be done for SVENDTC, as in our case, start date and end date of the visit are always identical, or we can just use:

as SVSTDTC has been defined before (i.e. more to the left in the table).

The variable SVSTDY and SVENDY can be mapped using SVSTDTC and SVENDTC, and using "datediff" in the same way as was done in the mappings for "AGE" and "DMDY" in the demographics domain. For this, we need the value for "RFSTDTC" (reference start date again). We can either copy it from the mapping for RFSTDTC in the domain, or drag-and-drop it from the "baseline visit – Visit date" again (attention: one should not have any "Generalize for ..." checkboxes checked in this case).

Creating global variables

There is however a smarter way...
We will need RFSTDTC and RFENDTC over and over again in many domains for deriving either -DY values, or "BEFORE", "AFTER" values for -STRF ("Start relative to reference period") and -ENRF ("End relative to reference period) variables.
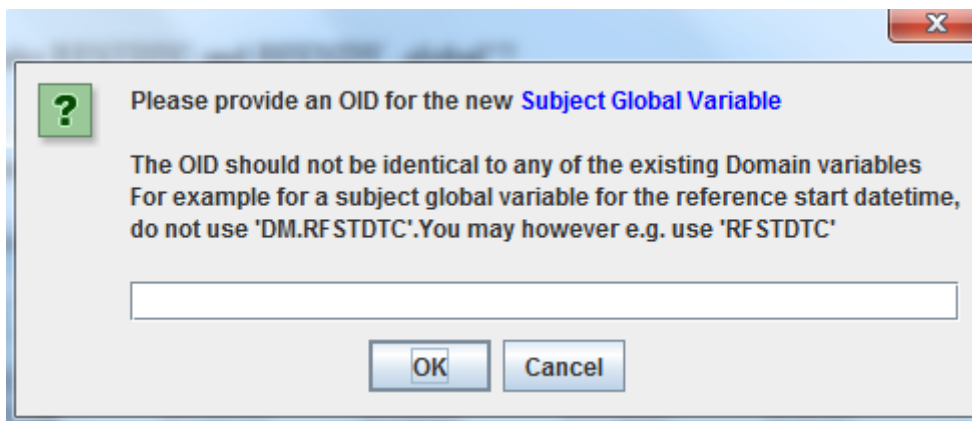
So why not make RFSTDTC and RFENDY "global"?

In SDTM, one can set global variables for reuse in any domain using the menu "Insert – Global Subject Variables Domain". This leads to:
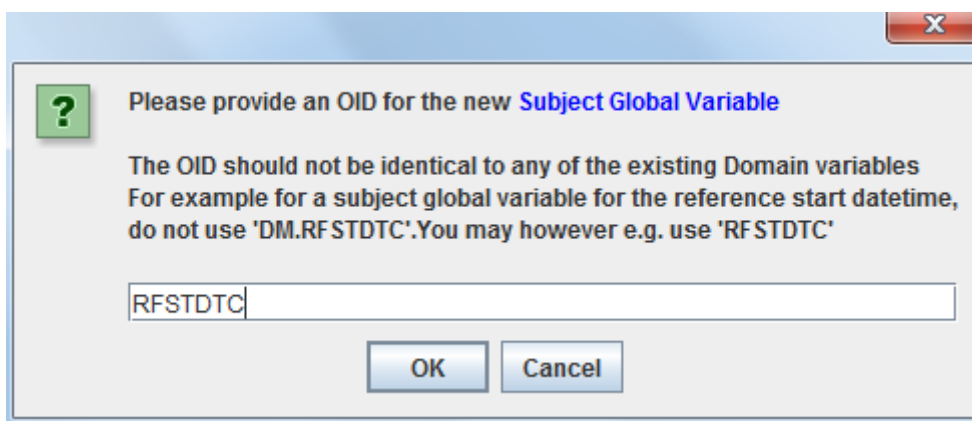
Domains (ItemGroups)				
Domain	Variable	Variable	Variable	Variable
DM	STUDYID	DOMAIN	USUBJID	SUBJID
SV	STUDYID	DOMAIN	USUBJID	SV.VISITNUM
CES:GLOBAL				
CES:DM	STUDYID	DOMAIN	USUBJID	SUBJID
CES:SV	STUDYID	DOMAIN	USUBJID	SV.VISITNUM

This is a very special domain (it will be removed from the "clean" define.xml when generating a "submission-ready" define.xml) to store global variables that we can then use in "read-only" mode in any of the mappings of our domains.
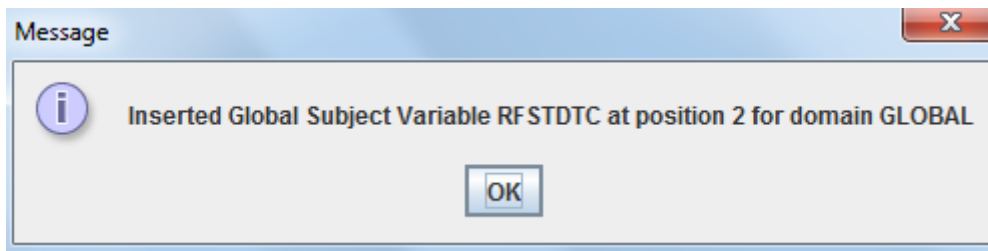
This "Global" domain typically contains things like mappings for RFSTDTC (reference start date) and RFENDTC (reference end date). After selecting any cell in the "CES:GLOBAL" domain, one can add variables by using the menu "Insert – New SDTM Variable", leading to the wizard:

> **?** 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'
>
> [   ]
>
> [OK]   [Cancel]

It explains that the identifier (OID) should be unique, i.e. it should be different from any OID already in the SDTM. So we can use "RFSTDTC", but not "DM.RFSTDTC".
For a global variable "Reference Start Date", we choose "RFSTDTC". So:

> **?** 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'
>
> [RFSTDTC]
>
> [OK]   [Cancel]

and click OK to finalize the process. This leads to:
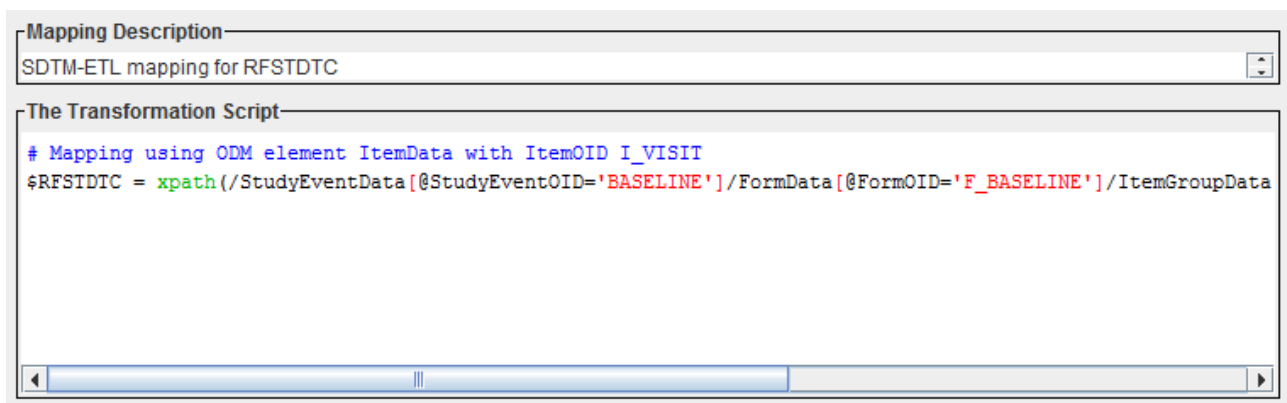
and:

The global variable can now be mapped in the same way as any SDTM variable, e.g. using drag-and-drop from the first visit date of our study design tree:
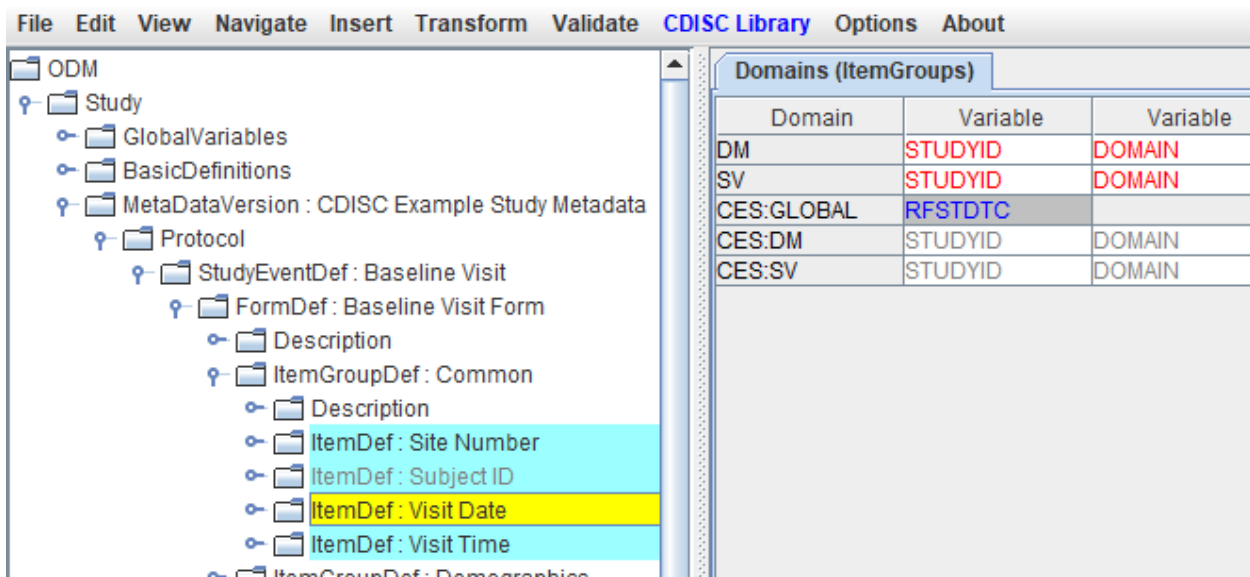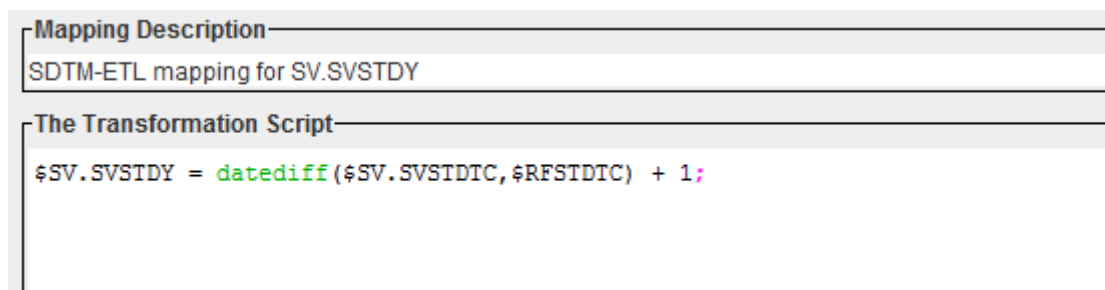

leading to:
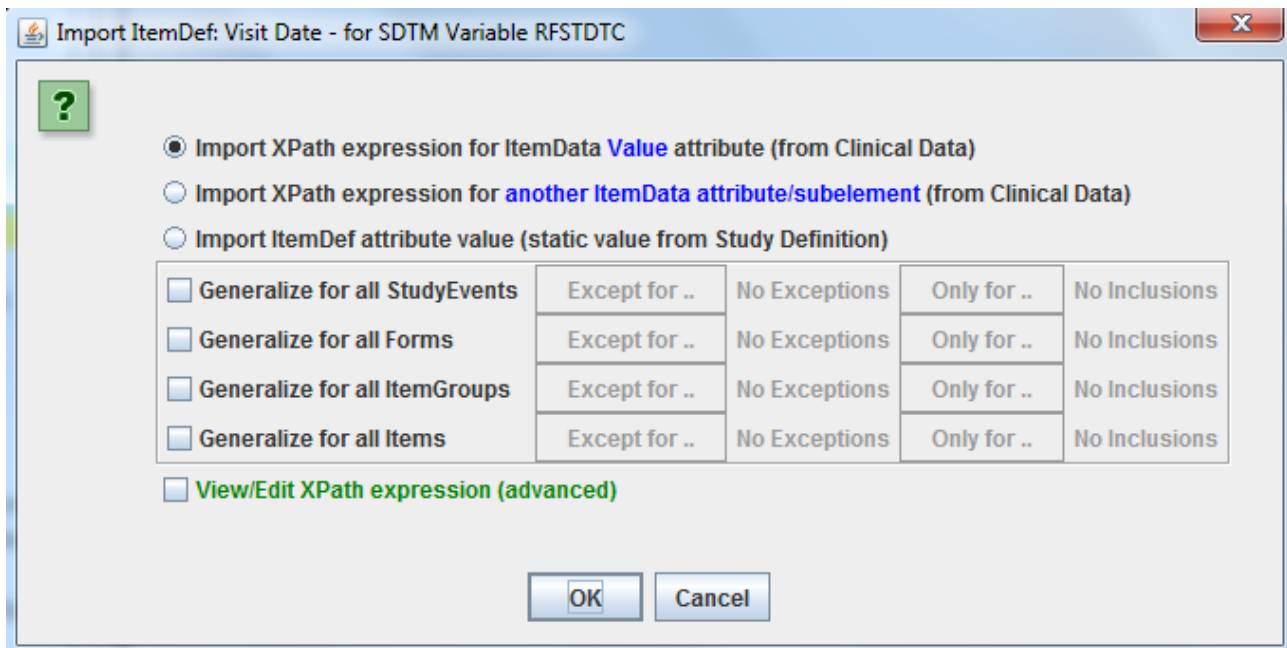
and the mapping:

We now return to our mappings for the SV domain.

For SV.SVSTDY we can now simply use:

as $SV.SVSTDC has been defined in the same domain before, and $RFSTDTC is a global variable. The "+1" is necessary as the SDTM does not allow a "day 0": the first day in a study is "day 1" and the day before it "day -1"[5].

---

[5]Conflicting with everything I learned in mathematics in primary school ;-)

This results into:

The same can now be done for the variable SVENDY.

Notice that the variable SV.VISITDY contains the "planned" day of the visit, not the actual one. So we will either need to copy it from the study design domain TV (trial visits), or hard-code it (using an if-elsif-else structure) from the information in the protocol[6].

Some important remarks

You may have noticed that when testing the mappings, we have used the menu "Transform – Generate Transformation (XSLT) Code ..." and have not performed any "local" testing using the button "Test – Transform to XSLT" in the mapping editor.
The reason is simple: when using the latter, the system is not knowing about any looping variables (in our case VISITNUM), nor is it knowing about any variables that were defined before, nor about global variables.

In our example study, we would now also like to map the "unplannned visits". The amount of collected clinical data we have so far (only one adverse event) does not provide us the necessary information yet on whether the AE was collected during a normal visit, or whether an additional (unplanned) visit was executed. Using "View – Clinical Data" for the "Visit Date" of the "Adverse Event" StudyEvent gives us:

---

[6]Essentially, VISITDY should NOT appear in "Subject Visits", as it is not "collected". If a reviewer would like to know what the planned visit day was, his review tool should enable him to jump to the TV domain.

and "2010-03-13" is also the visit date of the "DIARY" visit. If AEs were always collected during planned visits, we will not need any additional records in the SV domain.

Executing mappings for a single, selected domain

Currently we have developed mappings for as well the DM as for the SV domain. When testing, we each time see the results for both these domains. This is OK here, as both contain a small set of records, and generating the XSLT and executing it does not take much time.
For other domains however that have more records, we often would like to execute the mappings on the "current" domain only, even if mappings for other domains are also loaded.

As of version 3.0, only executing the mappings for a single domain is possible. For doing so, first use the menu "Options - Settings". This gives:

Now check the checkbox "Generate/Execute XSLT for user-selected domains only":

and click "OK". The next time one uses the menu "Transform – Generate Transformation (XSLT) Code" an additional dialog will be displayed:

allowing to select the domains for which the XSLT will be generated and the mappings executed.

In the case we select "CES:SV" this will lead to:

Important remarks:

This study did not have any unplanned visits for any of the subjects. If your study has such, you also need to have these in your SV dataset. The following rules then apply:

–   VISITNUM must be populated in such a way that it can be sorted in the correct chronological order. For example, if you have visits with VISITNUM being "3", "4" and "5" and the unplanned visit was after visit 4, you may assign a VISITNUM "4.1" for the unplanned visit. If you had planned visists "30", "40" and "50", and the unplanned visit came after visit "40", you may assign the number "41" for the unplanned visit.

We do however recommend using the former method (with the decimal point) as the tools used by the FDA do not support the second method, though it is fully legal.

–   VISIT can be left blank or can be filled with e.g. "UNSCHEDULED". If you plan to submit to the FDA, also test what you filled in here using the latest version of a good SDTM validation tool. Unfortunately, earlier (but also newer) versions of the tool used by the FDA make some invalid assumptions on how this should be dealt with[7]. If you get an error using such a validation tool, you might either want to adapt to what comes out of the validation tool (even if you are sure that you did it the right way), or make a notice about the (false positive) error in the reviewers guide. The latter is the better method: one should essentially not make adaptions in the submission due to false positives from a crappy validation software.

–   VISITDY must be left empty, as this is the <u>planned</u> study day.

–   SVSTDTC, SVENDTC, SVSTDY and SVENDY must be populated as usual.

–   SVUPDES <u>must</u> be populated with the reason for the unplanned visit.

This is well explained in the different SDTM Implementation Guides e.g. in section "Clinical Encounters and Visits"of the SDTM-IG 3.2 and higher.

–   [7]Unfortunately, the validation tool used by the FDA is well-known for generating a lot of false positives!