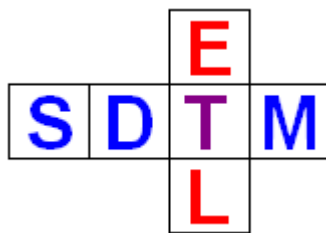


SDTM-ETL™



New features in version 3.2

Version 3.2 will become available early 2017

Table of Contents

Introduction.....3
Wizards for easier generation and editing of ValueLists.....4
Retrieving page numbers from an annotated CRF in PDF format.....18

Introduction

SDTM-ETL™ 3.2 is another "big step forward" in the development of software for the mapping between operational clinical data and the CDISC SDTM and Define-XML standard.

Although SDTM itself is under pressure, because being outdated (still using 2-dimensional tables and the "medieval" SAS-XPT format), and because more and more "derived" variables are added (see [here](#) and here), making it unnecessarily complex and error-prone, we will still need to live some time with it, also as the tools of the regulatory authorities are not very modern.

To cope with this (unnecessary) complexity, we are currently adding a good number of new features to SDTM-ETL™, making it even more user-friendly than it was, and at the same time adding new support for the new standards that are currently been developed and will soon be released by CDISC: SDTM 1.5/SDTM-IG 3.3 and Define-XML 2.1.

We are very proud to be able to present SDTM-ETL v.3.2.

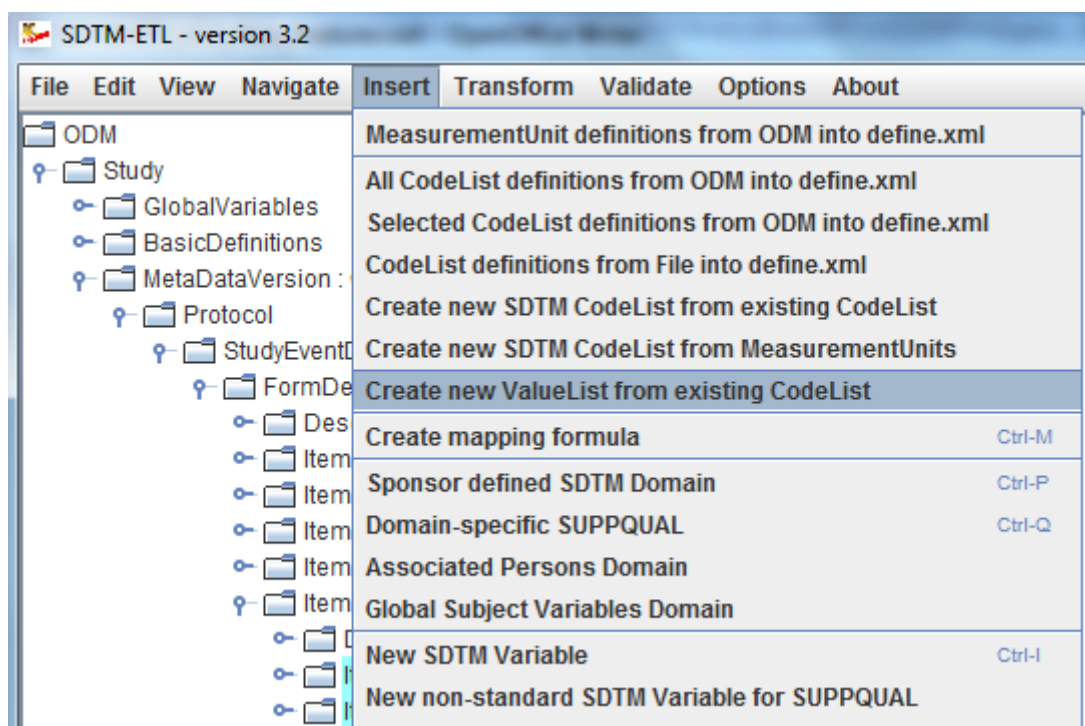
We are convinced that this is the best tool on the market for mapping operational clinical data to submission data, with the lowest total cost of ownership (TCO).

Wizards for easier generation and editing of ValueLists

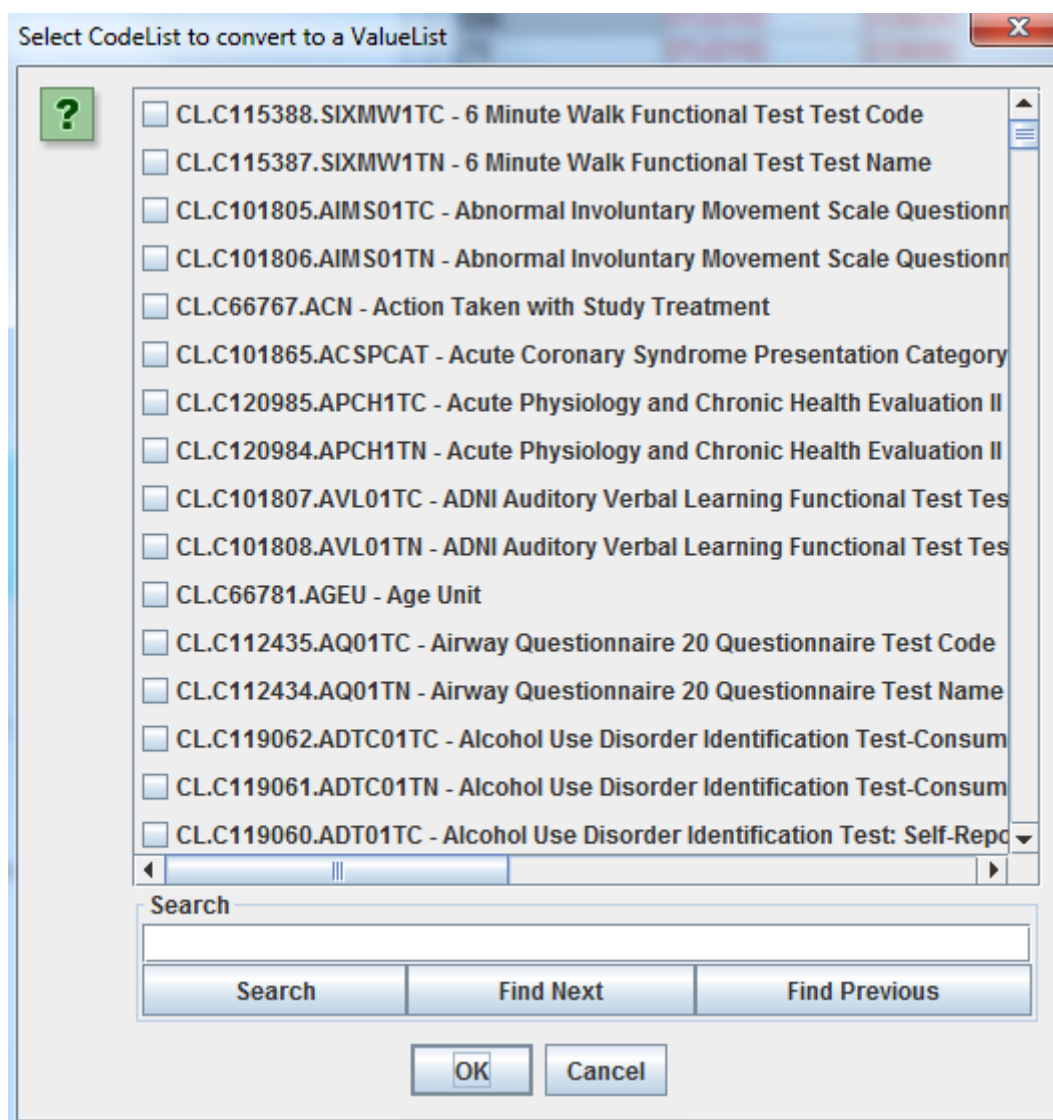
Generating and maintaining valuelists can be a lot of work. Essentially, one will need to generate a value-level variable for each test (or group of them) one has in each of the "Findings" domains. For example, if one has 50 different lab tests, each with their own metadata (datatype, max.length, number of figures after the decimal point, units used, or codelist for e.g. ordinal results), one may end with 50 different value-level variables, one for each test.

That looks like a lot of work, but we will show that this can be automated for a good part, without the user losing control and oversight.

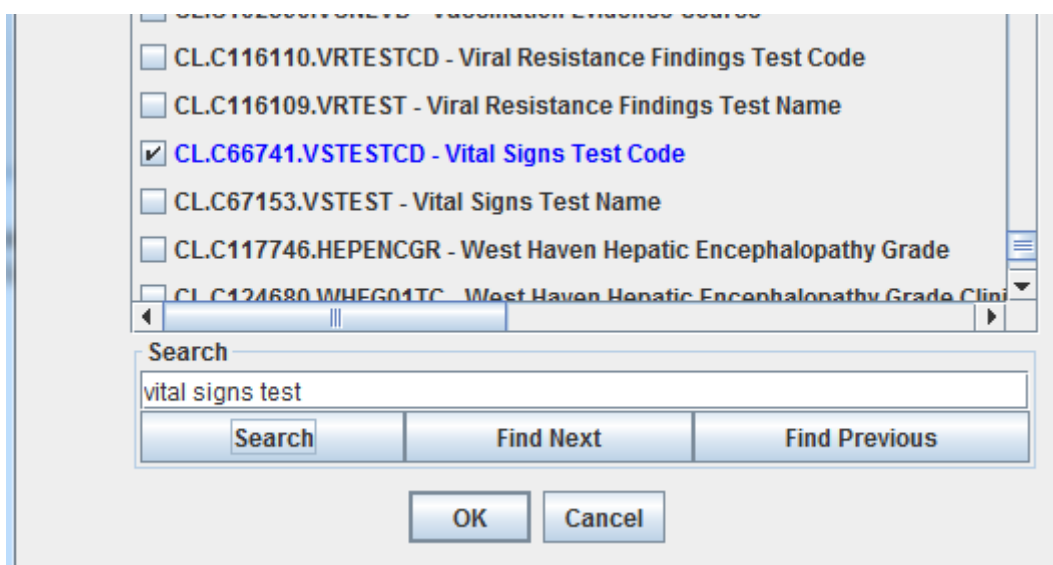
Usually, a ValueList is created started from a codelist. In SDTM-ETL™ this is done using the menu "Insert - Create ValueList from CodeList":



The user can then select one of the codelists that was loaded with the SDTM template. Usually (but not always) this will be CDISC CodeList:



The user can search for a specific codelist using the "Search" textfield and button. For example, for the "Vital Signs codes" codelist:



Clicking OK then leads to:

OID	Name	Data Type	Length	Sign.Digits	Origin	Comment	Description	def.Display...	Method	CodeList	WhereClause
IT.ABSKNF	ABSKNF						ABSKNF				WC.IT.ABSKNF
IT.BMI	BMI						BMI				WC.IT.BMI
IT.BMR	BMR						BMR				WC.IT.BMR
IT.BODLNTH	BODLNTH						BODLNTH				WC.IT.BODLNTH
IT.BODYFATM	BODYFATM						BODYFATM				WC.IT.BODYFATM
IT.BSA	BSA						BSA				WC.IT.BSA
IT.DIABP	DIABP						DIABP				WC.IT.DIABP
IT.FARMCIR	FARMCIR						FARMCIR				WC.IT.FARMCIR
IT.FRMSIZE	FRMSIZE						FRMSIZE				WC.IT.FRMSIZE
IT.HDCIRC	HDCIRC						HDCIRC				WC.IT.HDCIRC
IT.HEIGHT	HEIGHT						HEIGHT				WC.IT.HEIGHT
IT.HIPCIR	HIPCIR						HIPCIR				WC.IT.HIPCIR
IT.HR	HR						HR				WC.IT.HR
IT.IDEALWT	IDEALWT						IDEALWT				WC.IT.IDEALWT
IT.KNEEHEEL	KNEEHEEL						KNEEHEEL				WC.IT.KNEEHEEL
IT.LBM	LBM						LBM				WC.IT.LBM
IT.MAP	MAP						MAP				WC.IT.MAP
IT.MUARMCIR	MUARMCIR						MUARMCIR				WC.IT.MUARMCIR
IT.OXYSAT	OXYSAT						OXYSAT				WC.IT.OXYSAT
IT.PULSE	PULSE						PULSE				WC.IT.PULSE
IT.PULSEPR	PULSEPR						PULSEPR				WC.IT.PULSEPR
IT.RESP	RESP						RESP				WC.IT.RESP
IT.SAD	SAD						SAD				WC.IT.SAD
IT.SSKNF	SSKNF						SSKNF				WC.IT.SSKNF
IT.SYSP	SYSP						SYSP				WC.IT.SYSP

For each test code, a value-level variable has been created, and one needs to provide the datatype, in most cases the maximal length and sometimes also the number of digits after the decimal point, and sometimes an associated codelist, like for "FRMSIZE" ("Frame Size": small, medium, large).

This looks like a lot of work!

First of all, we need to remove those rows in the table for which there was no test anyway in our study. We can do this manually, but we can also do a lookup in the source ODM, to find out whether there was such a test.

Also adding the "WhereClause" (like "where VSTESTCD=DIABP") to each value-level variable looks like a lot of work. Also this process can be automated for a good part.

In the lower part of the screen, we find 3 new buttons:

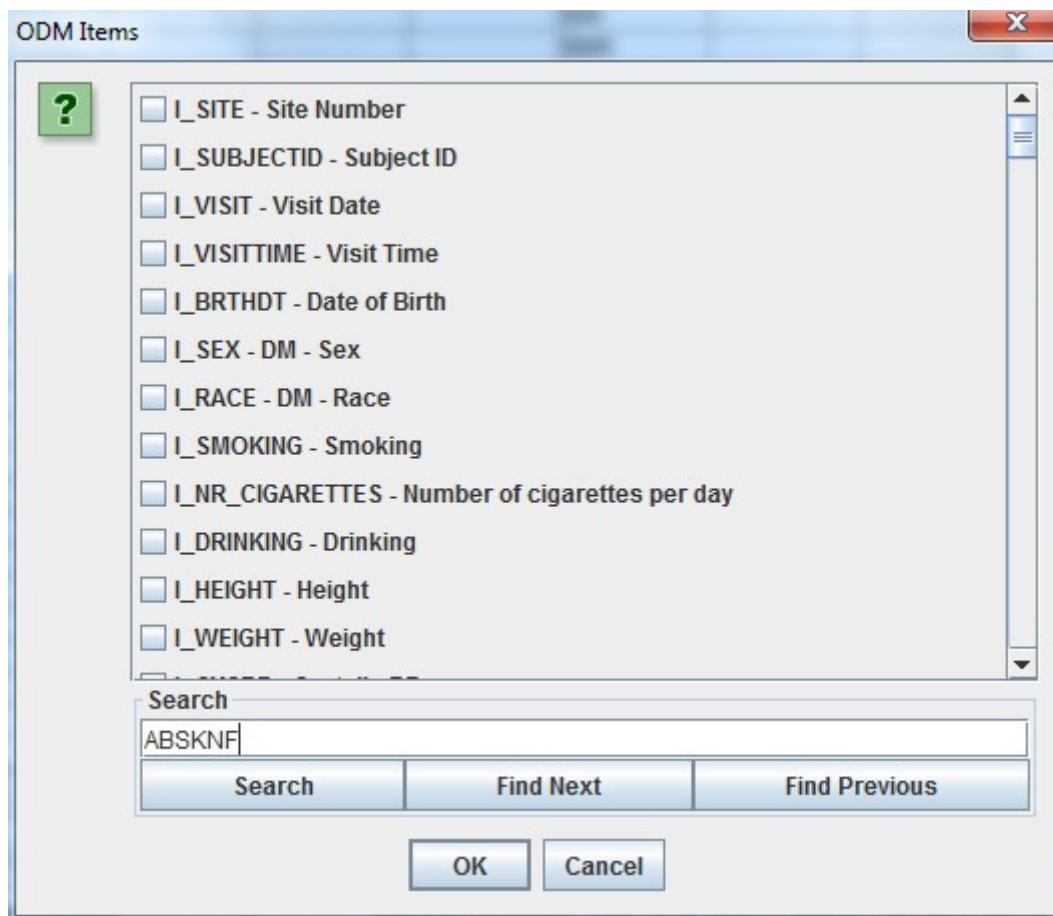
OID	Name	Data Type	Length	Sign.Digits	Origin	Comment	Description	def.Display...	Method	CodeList	WhereClause
IT.SAD	SAD						SAD				WC.IT.SAD
IT.SSKNF	SSKNF						SSKNF				WC.IT.SSKNF
IT.SYSP	SYSP						SYSP				WC.IT.SYSP

stating:

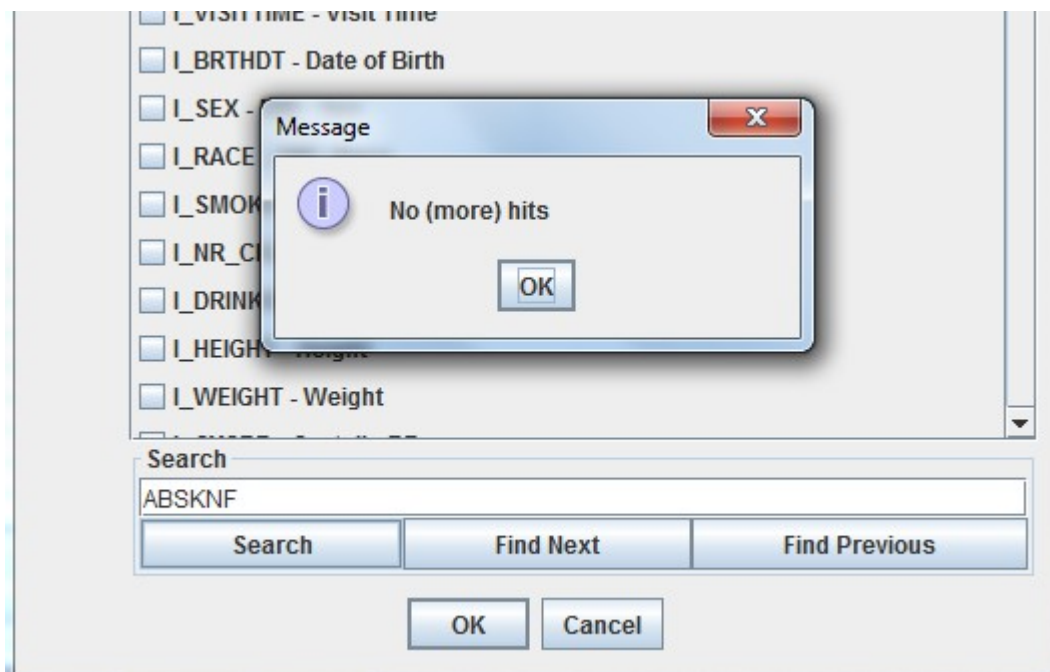
- Retrieve metadata from ODM source using SDTM annotations
- Retrieve metadata from ODM using (CRF) Item name
- Generate WhereClause automatically

First of all, let us look which vital signs were really present in the source ODM (study design). We

can easily find out using the button "Retrieve metadata from ODM using (CRF) Item name". For example, when we select the first row containing "ABSKNF", and then click the button, we are presented with the list of items (questions/datapoints) from the ODM:



and using the "Search" button:



So, the system did not find any item in which the wording "ABSKNF" appears¹. So we can assume that this test was not in the study design (it might of course be that it was named differently, but then we have a semantic problem) and so can be removed from the valuelist.

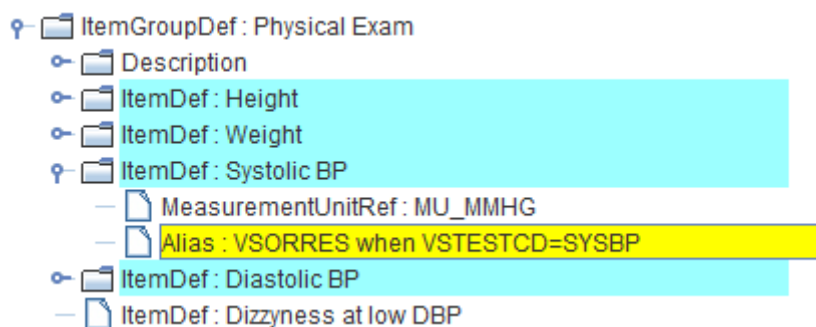
So, after removing the first row, the valuelist reduces a little bit as:

OID	Name	Data Type	Length	Sign Digits	Origin	Comment	Description	def.Display...	Method	CodeList	WhereClause
IT.BMI	BMI						BMI			WC.IT.BMI	
IT.BMR	BMR						BMR			WC.IT.BMR	
IT.BODLNTH	BODLNTH						BODLNTH			WC.IT.BODLNTH	
IT.BODYFATM	BODYFATM						BODYFATM			WC.IT.BODYFATM	
IT.BSA	BSA						BSA			WC.IT.BSA	
IT.DIABP	DIABP						DIABP			WC.IT.DIABP	
IT.FARMCIR	FARMCIR						FARMCIR			WC.IT.FARMCIR	
IT.FRMSIZE	FRMSIZE						FRMSIZE			WC.IT.FRMSIZE	
IT.HDCIRC	HDCIRC						HDCIRC			WC.IT.HDCIRC	
IT.HEIGHT	HEIGHT						HEIGHT			WC.IT.HEIGHT	
IT.HIPCIR	HIPCIR						HIPCIR			WC.IT.HIPCIR	
IT.HR	HR						HR			WC.IT.HR	
IT.IDEALWT	IDEALWT						IDEALWT			WC.IT.IDEALWT	

Finding out which tests were in the study design goes pretty fast in this way. We can then remove rows until we only have the vital signs that were really present in our study design.

Another strategy (a very good one) is to first "subset" the codelist for vital signs, using the menu "Insert - Create new SDTM CodeList from existing CodeList" and for units of measure, using the menu "Insert - Create new SDTM CodeList from MeasurementUnits".

In our valuelist table, there is a second way to find out whether a test was in the study design or not. This method is based on the annotations that were added to the ODM study design, usually already at study design time². These annotations use the "Alias" element in the ODM, usually with the "Context" being "SDTM". For example:



As the annotations (or better, their way of writing) is not standardized, some variations of it can be found, like "VSORRES where VSTESTCD = 'SYSBP'".

In our ValueList table editor, we can make advantage of this, by using the button "Retrieve Metadata from ODM Source using Annotations":

¹ We are currently working on extending this feature so that also a lookup is done in the XXTEST (Test Name) codelist. Very unfortunately, CDISC published coded test terms and their decode as two different lists. The decode of ABSKNF is "Abdominal Skinfold Thickness".

² This is an extremely good practice, explained in our "[ODM Wiki](#)"

IT.OXYSAT	OXYSAT							OXYSAT			
IT.PULSE	PULSE							PULSE			
IT.PULSEPR	PULSEPR							PULSEPR			
IT.RESP	RESP							RESP			
IT.SAD	SAD							SAD			
IT.SSSKNF	SSSKNF							SSSKNF			
IT.SYSBP	SYSBP							SYSBP			
IT.TBW	TBW							TBW			

Insert row

Retrieve Metadata from ODM source using SDTM Annotations

Retrieve Metadata from ODM source using (CRF)

Validate

For example, if we first select the row "BMI" and then click the button, we get:

'Alias' search string

Please complete/correct the following search string for searching through the SDTM/SEND annotations in the source ODM ('Alias' element). Do not mind after uppercase/lowercase, the search is case-insensitive. You may use the '*' as a wildcard, when you are not sure about the correct syntax in the ODM.

* when CL.C66741.VSTESTCD=BMI

OK

Cancel

We can still edit the search expression, and also use a "wildcard" ("*") if we do not know part of the annotation. However, the system is very smart, it will look for sentence similarity in all the ODM-SDTM annotations, and rank them according to the similarity. For the above search term (without any editing) we get:

i

☐ Item OID = I_HEIGHT - Name = Height - DataType

☐ Item OID = I_WEIGHT - Name = Weight - DataType

☐ Item OID = I_SYSBP - Name = Systolic BP - DataT

☐ Item OID = I_DIABP - Name = Diastolic BP - DataT

☐ Item OID = I_VISIT - Name = Visit Date - DataType

☐ Item OID = I_SITE - Name = Site Number - DataTy

☐ Item OID = I_SUBJECTID - Name = Subject ID - Da

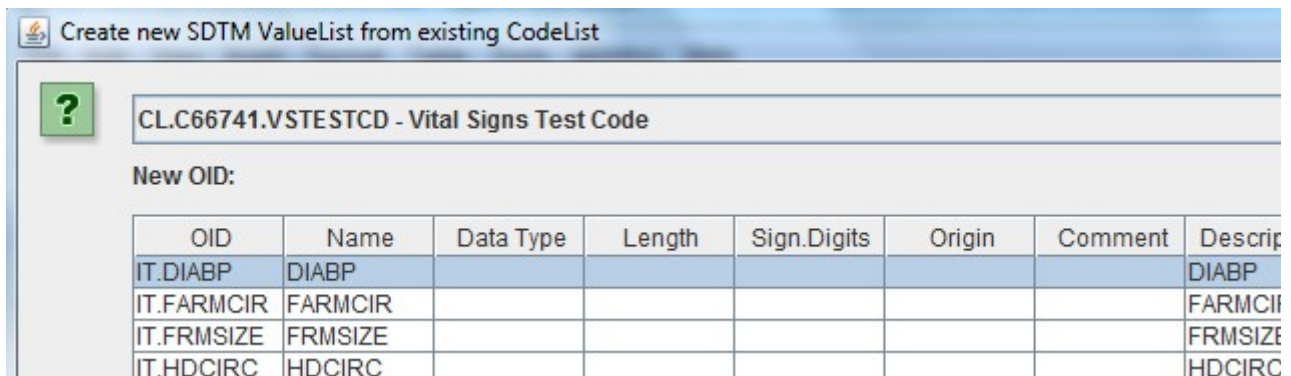
☐ Item OID = I_RACE - Name = DM - Race - DataTyp

Copy Metadata to ValueList

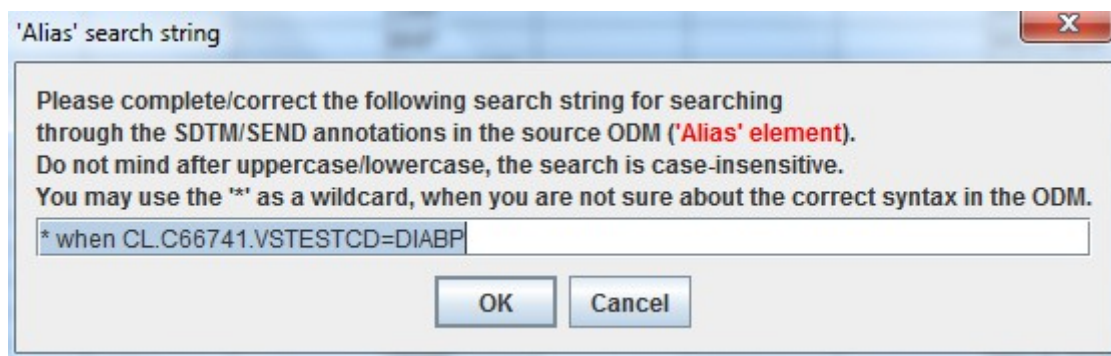
Cancel

indicating that none of the data points in the study design was annotated as being to be used later with VSTESTCD=BMI. So we can remove that row too.

If we do the same however for the row with entry "DIABP":



OID	Name	Data Type	Length	Sign.Digits	Origin	Comment	Description
IT.DIABP	DIABP						DIABP
IT.FARMCIR	FARMCIR						FARMCIR
IT.FRMSIZE	FRMSIZE						FRMSIZE
IT.HDCIRC	HDCIRC						HDCIRC

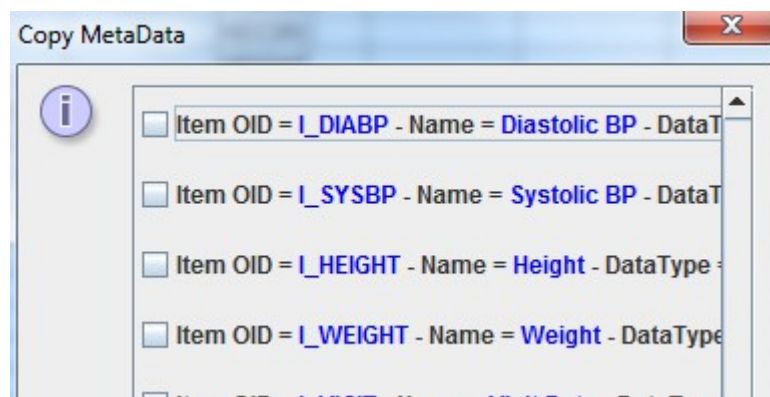


Please complete/correct the following search string for searching through the SDTM/SEND annotations in the source ODM (**Alias** element). Do not mind after uppercase/lowercase, the search is case-insensitive. You may use the ** as a wildcard, when you are not sure about the correct syntax in the ODM.

* when CL.C66741.VSTESTCD=DIABP

OK Cancel

and then executing the search, leading to:

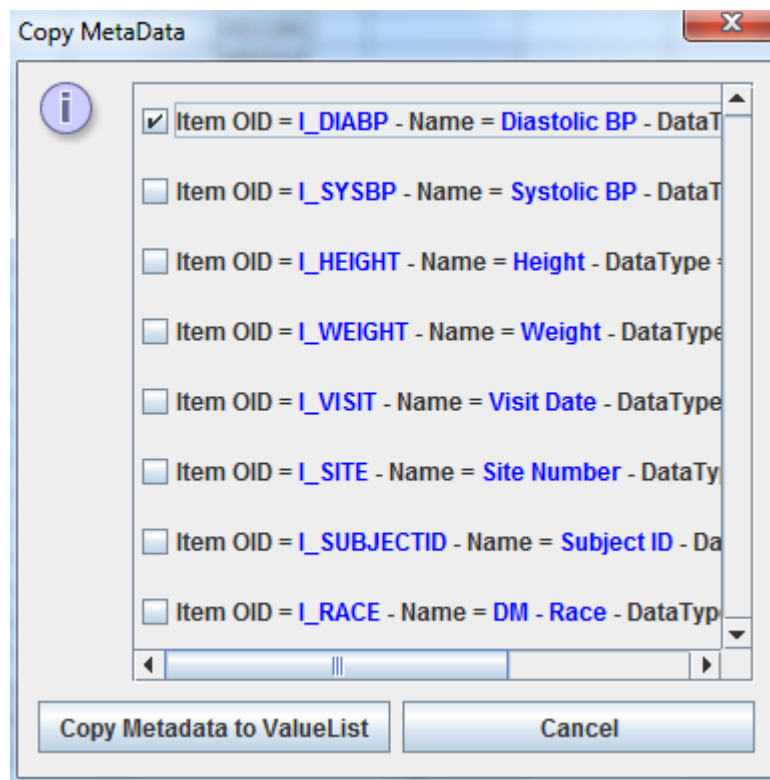


Copy MetaData

- ☐ Item OID = I_DIABP - Name = Diastolic BP - DataT
- ☐ Item OID = I_SYSBP - Name = Systolic BP - DataT
- ☐ Item OID = I_HEIGHT - Name = Height - DataType
- ☐ Item OID = I_WEIGHT - Name = Weight - DataType
- ☐ Item OID = I_VIST - Name = VIST - DataT

the items in the list being ordered by similarity with the search string, and showing that there is indeed an item in the study design that was annotated and that is a "diastolic blood pressure".

We can then copy its metadata from the ODM to the SDTM, by selecting the item and clicking the button "Copy Metadata to ValueList":



and immediately, we see in our valuelist table:

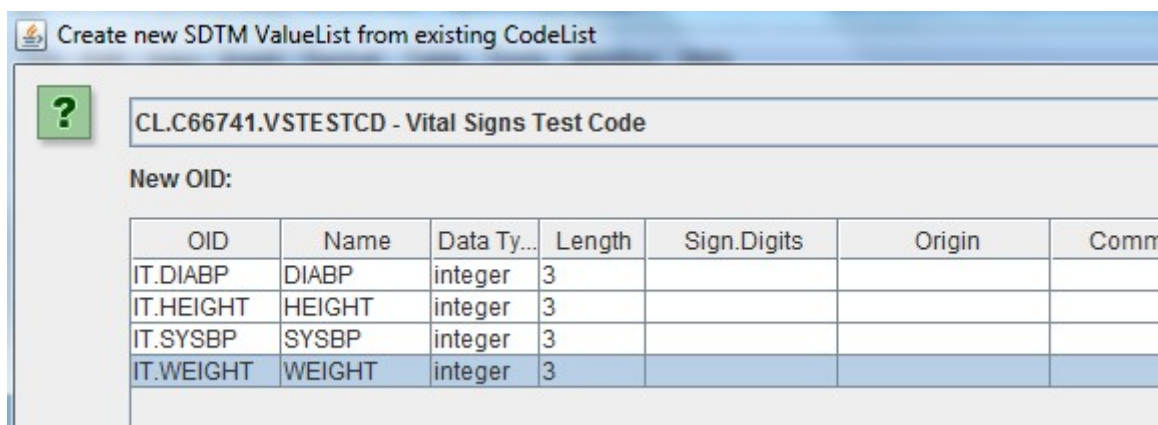
OID	Name	Data Type	Length	Sign.Digits	Origin
IT.DIABP	DIABP	integer	3		
IT.FARMCIR	FARMCIR				
IT.FRMSIZE	FRMSIZE				
IT.HDCIRC	HDCIRC				
IT.HEIGHT	HEIGHT				
IT.HIPCIR	HIPCIR				

that the datatype (in this case "integer") maximal length (in this case "3") have been copied. If a codelist would be involved, also the codelist reference would be copied in the cell "CodeList". If that codelist is not present in the loaded controlled terminology (in the underlying define.xml), the user is invited to copy the codelist from the ODM into the SDTM, or to choose another one that is already in the SDTM. The latter can easily be done by using the dropdown, e.g. for "FRMSIZE":

OID	Name	Data Ty...	Length	Sig...	Origin	Co...	Desc...	def...	Met...	CodeList	WhereClau...
IT.DIABP	DIABP	integer	3				DIABP				WC.IT.DIABP
IT.FARMCIR	FARMCIR						FAR...				WC.IT.FAR...
IT.FRMSIZE	FRMSIZE						FRM...				WC.IT.FRM...
IT.HDCIRC	HDCIRC						HDCI...			CL.C100155.SFMP2TN	WC.IT.HDC...
IT.HEIGHT	HEIGHT						HEIG...			CL.C66733.SIZE	WC.IT.HEI...
IT.HIPCIR	HIPCIR						HIPC...			CL.C76351.SKINCLAS	WC.IT.HIP...
IT.HR	HR						HR			CL.C112024.SRTE	WC.IT.HR
IT.IDEALWT	IDEALWT						IDEA...			CL.C112023.SRTE	WC.IT.IDEA...
IT.KNEEHE...	KNEEHEEL						KNE...			CL.C74561.SKINTY	WC.IT.KNE...
IT.LBM	LBM						LBM			CL.C102587.RISKS	WC.IT.LBM
IT.MAP	MAP						MAP			CL.C78733.SPECC	WC.IT.MAP
IT.MUARM...	MUARMCIR						MUA...				WC.IT.MUA...
IT.OXYSAT	OXYSAT						OXY...				WC.IT.OXY...

With these two new features, one can easily and relatively quickly find out which tests were in the study design, for which value-level metadata need to be provided, and copy them from the study design automatically³. The ones in the list that were not in the study design can then easily be removed from the valuelist table.

Doing so in our case, quickly leads to:

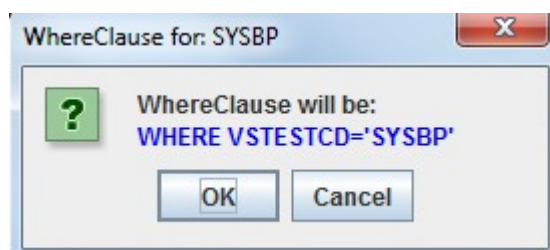


with datatype and maximal length being copied from the source ODM study design.

The third button under the table states "Generate WhereClause Automatically".

Most of our "WhereClauses" will be similar to "where VSTESTCD=DIABP" for the value-level variable "SYSBP". So this can also be automated.

When e.g. the entry "SYSBP" is selected, and the button "Generate WhereClause Automatically" is clicked, a dialog is displayed with:



You can still cancel this now. If "OK" is clicked, the "WhereClause Wizard" shows up, allowing to still make changes to our "WhereClause".

AUTOMATED "Origin" ASSIGNMENT: TODO

³ We decided not to make this a "bulk" or "blind" automatism, as that can easily lead to errors, like removing tests that were known under a (more or less slightly) different name or code in the study design. We consider it to be important that the user understands what he/she is doing, unlike in "black box" tools.

WhereClause Editor

OID: WC.IT.SYSBP

Comment: [External document for comment](#)

Show 'Where' clause

Number of RangeChecks: 1

Comparator: EQ Item OID: VS.VSTESTCD
VS.VSTEST
VS.VSCAT
VS.VSSCAT
VS.VSPOS
VS.VSORRES
VS.VSORRESU
VS.VSSTRESC
VS.VSSTRESN
VS.VSSTRESU
VS.VSSTAT

CheckValue: SYSBP

Also here, we can validate whether the underlying machine-readable expression is correct, using the "Show 'Where' Clause" button. We can however also add a comment and/or add a link to a comment to an external document (like a reviewers guide).

If you feel fine, just click the "OK" button, returning to the valuelist table. We then see that the cell on the right is highlighted (blue color). When holding the mouse over the cell, we also see the "WhereClause" as a human-readable expression:

Comment	Description	d...	...	...	WhereClause
	DIABP				WC.IT.DIABP
	HEIGHT				WC.IT.HEIGHT
	SYSBP				WC.IT.SYSBP
	WEIGHT				WC.IT.WEIGHT

where VSTESTCD EQ 'SYSBP'

Just for the exercise, let us know make a single "WhereClause" for as well the systolic blood pressure. This makes sense, as both are very similar properties, both are measured as an integer with a maximal length of 3.

First, we remove one of both entries. Let's says we just keep the first which is "DIABP", and remove "SYSBP":

OID	Name	Data Type	Length	Sign.Digits
IT.DIABP	DIABP	integer	3	
IT.HEIGHT	HEIGHT	integer	3	
IT.WEIGHT	WEIGHT	integer	3	

Then we edit the OID and the Name to reflect that this is about blood pressure. For example:

Remember that (as long as we are forced to use SAS-XPT), the value for "Name" may not be longer than 8 characters. For the "Description", we write "Systolic and diastolic blood pressure". For the same reason as before, the "Description" may not be longer than 40 characters⁴:

OID	Name	Data Type	Length	...	O...	C...	Description	d...	...	WhereClause
IT.BLOODPRESSURE	BLP	integer	3				Systolic and diastolic blood pressure			WC.IT.DIABP
IT.HEIGHT	HEIGHT	integer	3				HEIGHT			WC.IT.HEIGHT
IT.WEIGHT	WEIGHT	integer	3				WEIGHT			WC.IT.WEIGHT

The one thing we still need to adapt is the "WhereClause". So we click the corresponding cell on the right, and the "WhereClause Wizard" is displayed:

Designing/Updating WhereClause for Item: IT.DIABP

OID: WC.IT.DIABP

Comment: [External document for comment](#)

Show 'Where' clause

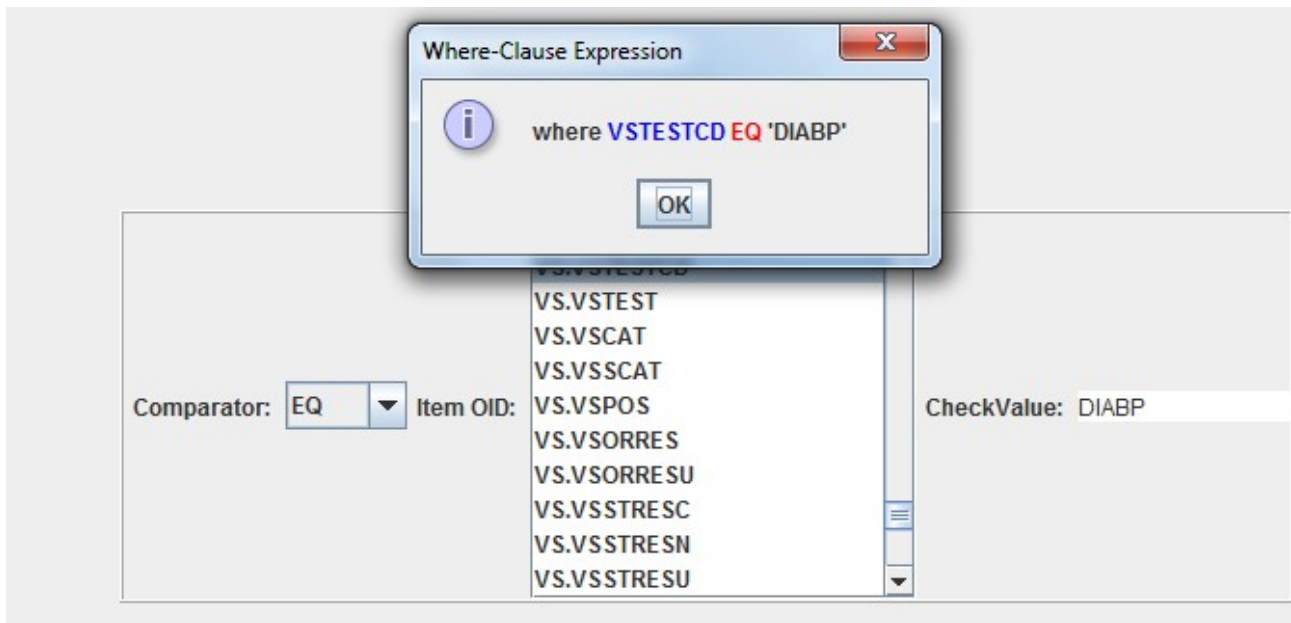
Number of RangeChecks: 1

Comparator: EQ Item OID: VS.VSSPID VS.VSTESTCD VS.VSTEST VS.VSCAT VS.VSSCAT VS.VSPOS VS.VSORRES VS.VSORRESU VS.VSSTRESC VS.VSSTRESN VS.VSSTRESU

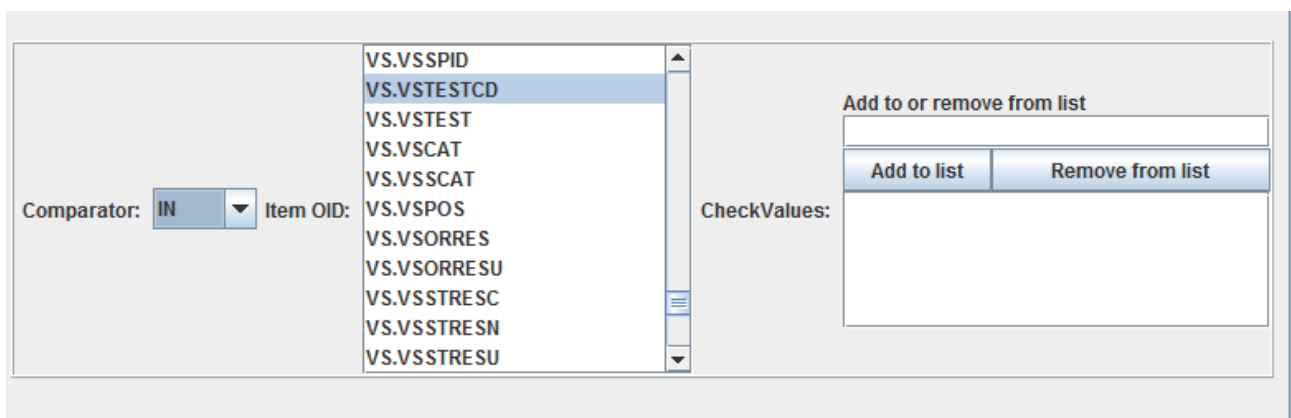
CheckValue: DIABP

using the "Show 'Where' Clause" button, we see that currently, only DIABP is covered:

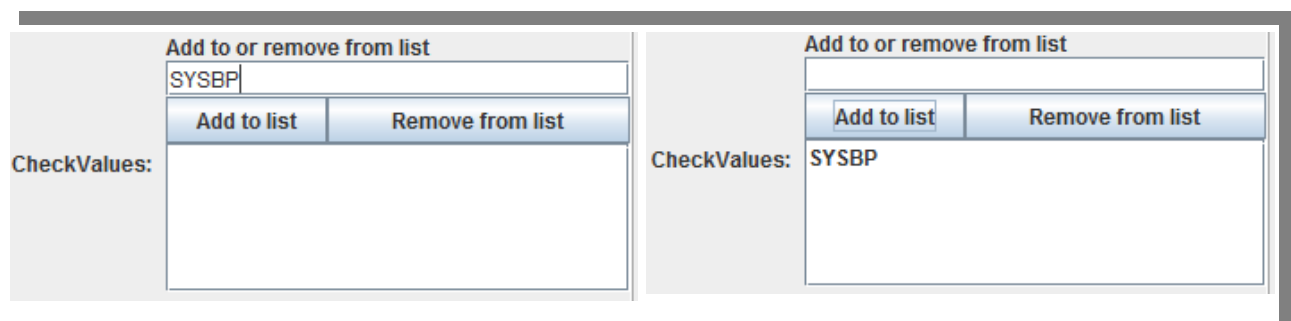
⁴ Well, isn't it time the FDA moves away from SAS-XPT?



but we of course also want it to cover SYSBP. Such an "OR" statement is essentially accomplished by providing a list, and then stating that the value must be in the list. So we set the "Comparator" to "IN" (using the dropdown), leading to:



"VSTESTCD" is still OK, and we still need to add "SYSBP" and "DIABP" to our list. We can just do by typing into the "Add to or remove from list", and then click "Add to list", like:



and again for "DIABP":

Add to or remove from list

DIABP

Add to list
Remove from list

CheckValues: SYSBP

Add to or remove from list

Add to list
Remove from list

CheckValues: SYSBP
DIABP

When then clicking the "Show 'Where' Clause again, we get:

Where-Clause Expression

where VSTESTCD IN ['SYSBP', 'DIABP']

OK

Comparator: IN

Item OID: VS.VSSPID
VS.VSTESTCD
VS.VSTEST
VS.VSCAT
VS.VSSCAT
VS.VSPOS
VS.VSORRES
VS.VSORRESU
VS.VSSTRESC
VS.VSSTRESN
VS.VSSTRESU

Add to or remove from list

Add to list
Remove from list

CheckValues: SYSBP
DIABP

which is exactly what we want.

We also change the OID (although it is nothing else than an identifier, so it is not mandatory, but we like our OIDs to be "mnemonic", so we make it:

OID: WC.IT.BLOODPRESSURE

Comment:

External document for comment

Show 'Where' clause

Number of RangeChecks: 1

Comparator: IN

Item OID: VS.VSSPID
VS.VSTESTCD
VS.VSTEST
VS.VSCAT
VS.VSSCAT
VS.VSPOS

Add to or remove from list

Add to list
Remove from list

CheckValues: SYSBP

Clicking "OK" then brings us back to the valuelist table:

OID	Name	Data Type	Length	...	O...	C...	Description	d...	...	...	WhereClause
IT.BLOODPRESSURE	BLP	integer	3				Systolic and diastolic blood pressure				WC.IT.BLOODPRESSURE
IT.HEIGHT	HEIGHT	integer	3				HEIGHT				WC.IT.HEIGHT
IT.WEIGHT	WEIGHT	integer	3				WEIGHT				WC.IT.WEIGHT where VSTESTCD IN ['SYSBP', 'DIABP']

again showing the "human-readable" expression as a tooltip.

We should not forget to validate the table, using the "Validate" button which is near the bottom. Using it leads to ... nothing, meaning that our "ValueList" table is technically valid.

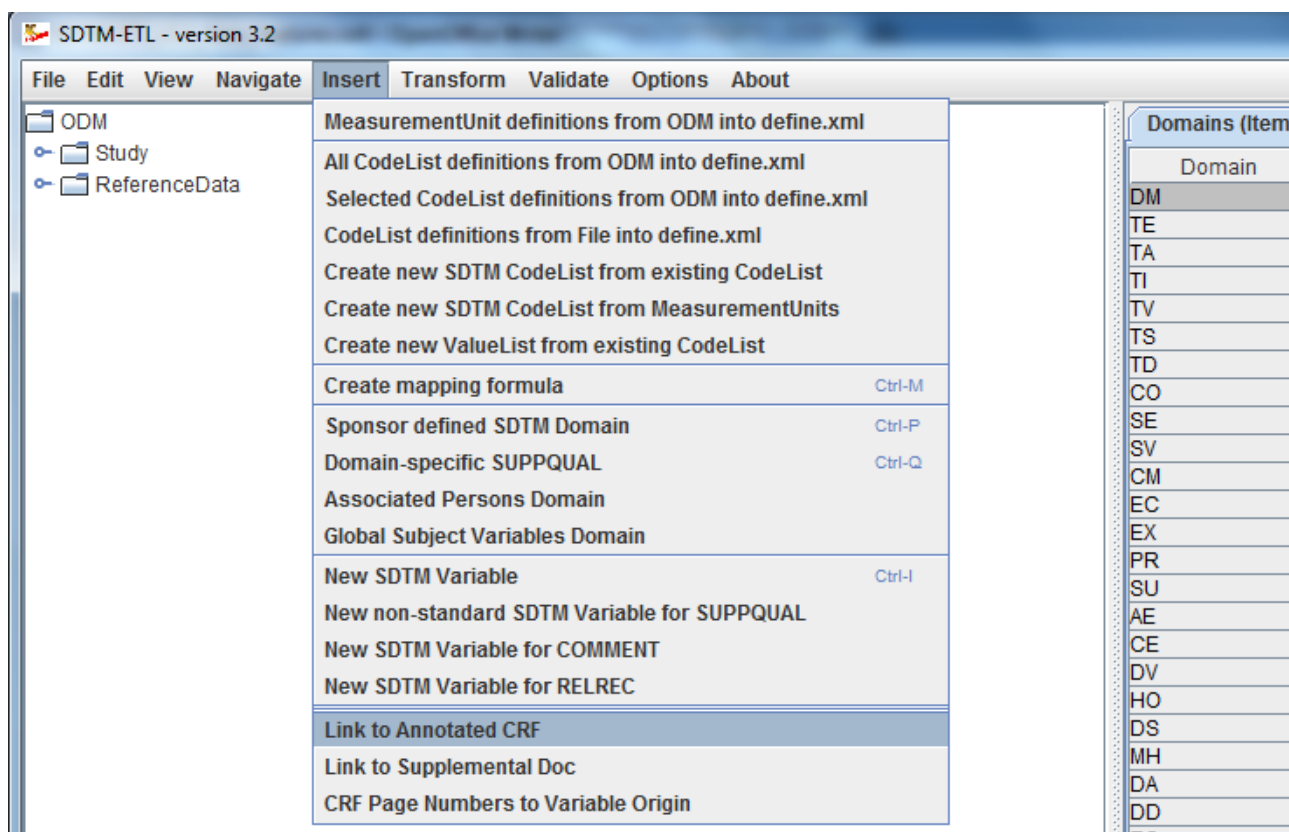
We could still add the "Origin", but we could also do that at the higher level, when all vital signs were collected on the same "page"⁵. We will describe how to semi-automatically add page numbers in the next section.

5 When using EDC, this essentially does not make sense. As we start from a study design in ODM, our mapping script IS the description of the origin. However, the FDA still want sponsors to generate a paper-like annotated CRF artificially, with page numbers, although the EDC screens do not have such at all.

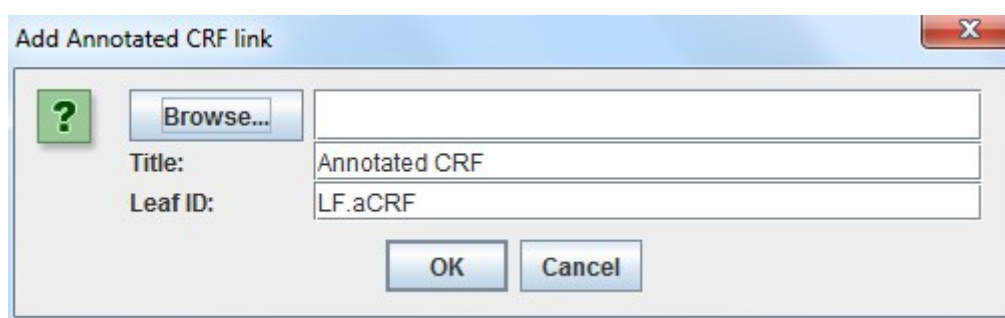
Retrieving page numbers from an annotated CRF in PDF format

Retrieving page number information from annotated Case Report Forms (aCRFs) has now become very easy using SDTM-ETL.

First, one of course needs to provide the location of the aCRF, which can be done using the menu "Insert - Link to annotated CRF":



leading to:



One gives the link a title (e.g. "Annotated CRF"), an ID (like "LF.aCRF"), and can then use the "Browse" button to add the location of the annotated CRF.

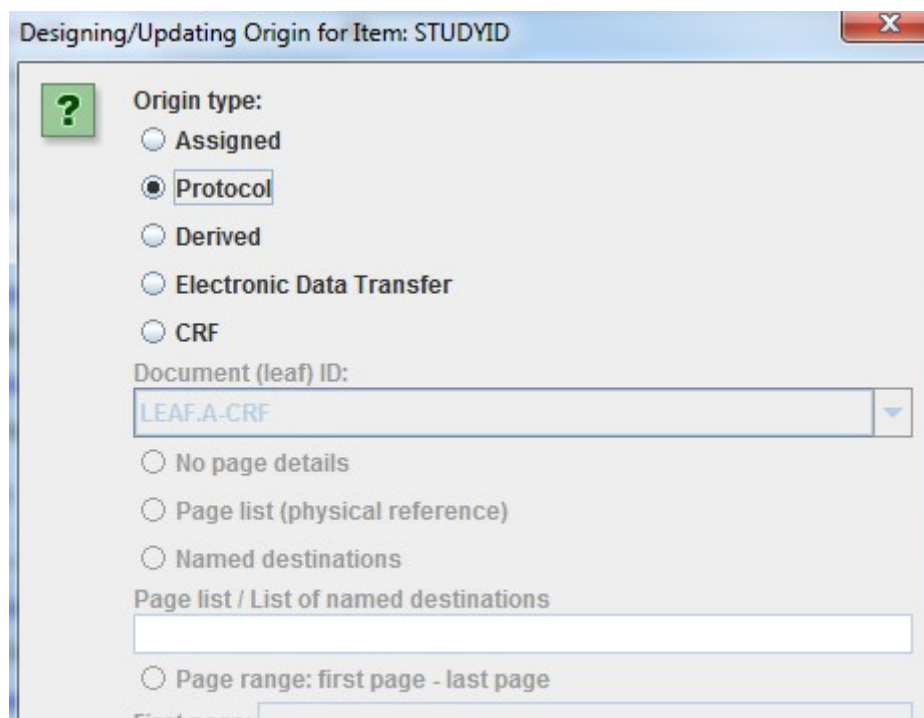
One can then later always change the information using the menu "Edit - Links to External Documents".

We have already learned how to add an "Origin" to an SDTM/SEND/ADaM variable, by selecting the cell in the table, and then using the menu "Edit - SDTM Variable Properties"⁶. For example for "STUDYID":

When then checking the "Edit Origin" checkbox, followed by clicking the "Edit" button, the "Origin" wizard will appear:

⁶ In case of using SEND, the menu is automatically adapted to "SEND Variable Properties".

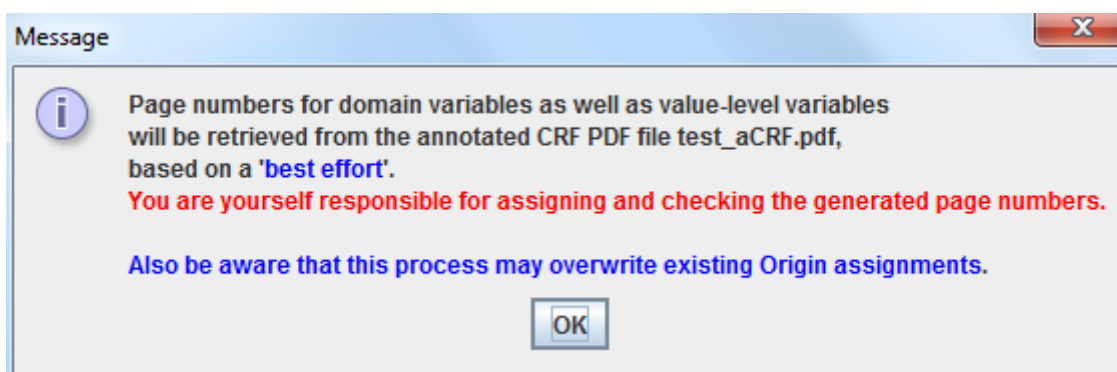
Depending on the origin type (Assigned, Protocol, Derived, Electronic Data Transfer, CRF), a number of fields will become disabled/enabled and to be filled or not. For example, when we state that the STUDYID is taken from the protocol, we just select "Protocol" and all other fields get disabled:



This also allows to add page numbers in the case of the CRF being the origin, and this in a very user-friendly way. Doing so for each variable (we have hundreds of them) is still a cumbersome task, which we want to automate when possible.

If we do already have an annotated CRF, the page numbers can be semi-automatically retrieved and the "Origin" elements populated and assigned to the variables, this as well to the normal domain variables as well as to value-level variables.

To do so, use the menu "Insert - CRF Page Numbers to Variable Origin". The following information is displayed:



Unlike other (black box) tools on the market, you will be guided through the process, allowing you to decide yourself which page numbers will be copied to the variables, and which not. Normally, PDF page numbers will only be retrieved for variables for which you already provided a

mapping. If your mapping is not complete yet, you can later repeat the process if you want - the software will tell you for which variables you do have already the page numbers assigned and for which not.

After clicking OK, the system is retrieving all information (this may take a few seconds), and the following dialog is displayed:

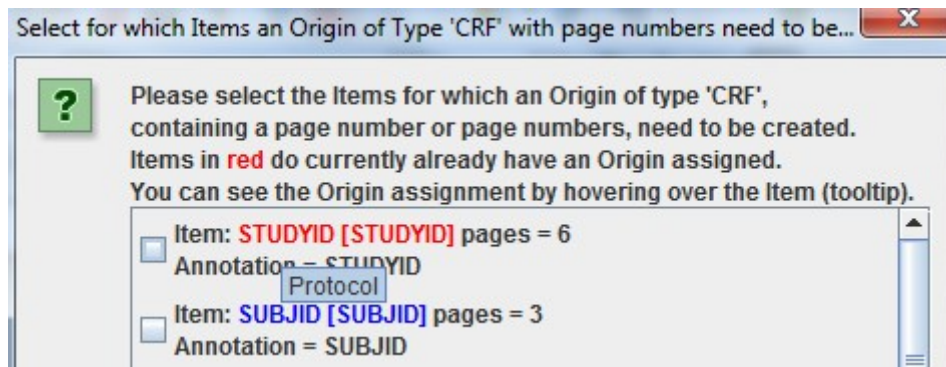
Select for which Items an Origin of Type 'CRF' with page numbers need to be...

? Please select the Items for which an Origin of type 'CRF', containing a page number or page numbers, need to be created. Items in **red** do currently already have an Origin assigned. You can see the Origin assignment by hovering over the Item (tooltip).

☐	Item: STUDYID [STUDYID] pages = 6 Annotation = STUDYID
☐	Item: SUBJID [SUBJID] pages = 3 Annotation = SUBJID
☐	Item: SITEID [DM.SITEID] pages = 3 Annotation = SITEID
☐	Item: BRTHDTC [DM.BRTHDTC] pages = 6 Annotation = BRTHDTC
☐	Item: SEX [DM.SEX] pages = 6 Annotation = SEX
☐	Item: RACE [DM.RACE] pages = 6 Annotation = RACE
☐	Item: QSTEST [QS.QSTEST] pages = (13,14,15,17) Annotation = QSTEST
☐	Item: QSORRES [QS.QSORRES] pages = (13,14,15,17) Annotation = QSORRES
☐	Item: VISIT [QS.VISIT] pages = (4,5) Annotation = VISIT
☐	Item: QSDTC [QS.QSDTC] pages = (13,14,15,17)

OK Cancel

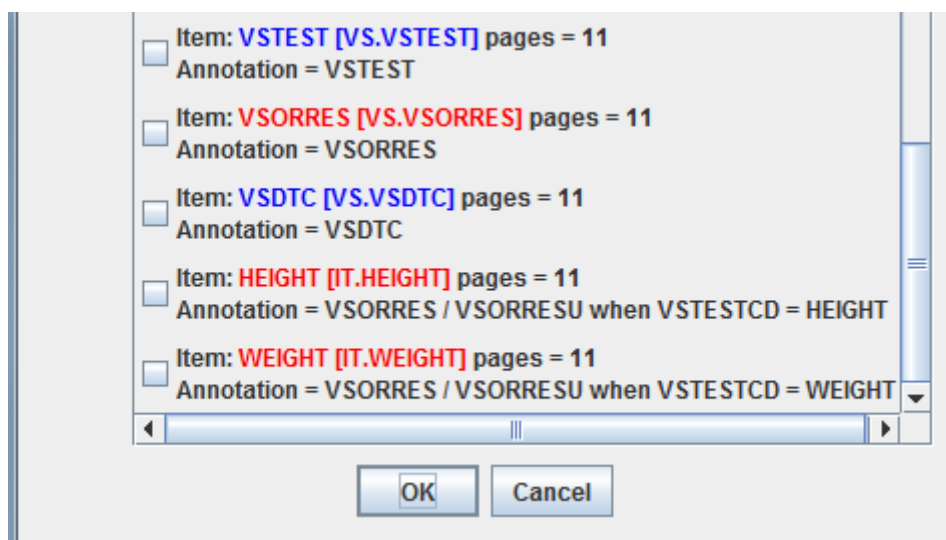
In each row, one sees the suggested variable, the page number or numbers, and the PDF annotation itself. Rows with a red text indicate that there is an "Origin" assignment yet, and one can easily find out what assignment is already present, by hovering the mouse over the row. A tooltip is then displayed:



If the assignment were a CRF page or pages, also the latter will be displayed in the tooltip (see later).

As a user, you can now decide yourself which page numbers must be copied to the variable "Origin". For example, an annotation of "STUDYID" was found on page 6, but you want to keep the old assignment, stating that it comes from the protocol, so you would in this case not check the checkbox.

Suggestions are also generated for value-level variables. For example, near the bottom of the list we find:



stating that annotations were found on page 11 about value-level variables "HEIGHT" and "WEIGHT". It is also indicated that an "Origin" assignment was already present, so if we check these checkboxes, the old assignments will be overwritten.

One can now decide for which variables (domain or value-level variables) the page numbers are copied. For example, we check all of them, except for "STUDYID". The assignments are then automatically performed, but only for the variables for which the checkbox was checked.

If we later than inspect the "Origin" for e.g. the variable "QSORRES" (using the menu "Edit - SDTM Variable Properties" again, we find:

Designing/Updating Origin for Item: QSORRES

? Origin type:

☐ Assigned
☐ Protocol
☐ Derived
☐ Electronic Data Transfer
☒ CRF

Document (leaf) ID:

LEAF.A-CRF

☐ No page details
☒ Page list (physical reference)
☐ Named destinations

Page list / List of named destinations

13 14 15 17

☐ Page range: first page - last page
 First page:
 Last page:

OK Cancel

We can then still change this, e.g. add page numbers, of changing the list into a "first page - last page" in this case (just as an example) set it to pages 13-17:

☐ No page details
☐ Page list (physical reference)
☐ Named destinations

Page list / List of named destinations

13 14 15 17

☒ Page range: first page - last page
 First page:
 Last page:

OK Cancel

When one later then repeats the whole process, all the variables that already were assigned an origin are marked as such, for example:

