PhUSE 2013

Paper PP12

Check your Standards – An Example

Véronique Bourcier, Bayer Healthcare, Leverkusen, Germany

ABSTRACT

Companies maintain meta data and code lists for clinical studies and drug projects in different ways and systems. Meta data become more and more the key for daily business, either for project standardization and adherence or process and system automation. Therefore it's very important and essential that the meta data and code lists used and maintained within a company are a reliable source and well-structured and proofed for consistency. An example will give you an overview of development and maintenance of a compliance check package, which can be applied as concept, independent whether the source are Excel tables, SAS data sets or a database. In addition global available standards like CDISC SDTM IG, SDTM controlled terminology, etc. should be also taken into account. A well checked consistency package for standards needs to be key for a well maintained standards environment used for process automation.

INTRODUCTION

The purpose of this paper is to show the implementation of five automatic checks: definition of the check, generation of the check report, annotation of the queries. It also gives some recommendations to reduce the amount of queries by changing a single input table per standard to a set of tables with no overlap.

IMPLEMENTING 5 NEW CHECKS

In this paper five checks are implemented. They are called:

- chk_codlst_dup.
- chk_type_codmeta
- chk_extensible_ny
- chk_outform_across
- chk_type_metacod

Each check has an unique name. This check name starts with the prefix 'chk_' to facilitate the programming work (e.g. append all check outputs starting with the prefix 'chk'). The name gives some information about the meaning of the check. It is also limited to 32 characters in order to be able to use it as SAS dataset name during the programming activities.

STORE THE CHECK DEFINITION IN A SAS TABLE

The objective of these five checks can be summarized as follow:

- chk_codlst_dup lists duplicates in codelists.
- chk_type_codmeta lists any character format used by a numeric variable and any numeric format used by a character variable.
- chk_extensible_ny applies to SDTM controlled terminology (CT) based codelists. Some of this codelists are limited to the values provided by SDTM CT. The check lists any value not present in SDTM CT.
- chk_outform_across is a metadata check which lists variables defined by the same codelist but having different output format (w.d).
- chk_type_metacod is a metadata check which lists numeric variable having a character codelist for attributes and reversely character variables having a numeric codelist attached to it.

The check definition is stored in a dataset with the following structure.

Variable Name	Variable Label		Key
chk_type	Check Type	$10	
chk_std	Check applies to standard	$8	
chk_name	Check Name	$32	(1)
chk_order	Check Order	8	
chk_expval	Check Expected Value	$1000	
chk_except	Exception allowed (Y/N)	$1	
chk_rule	Check Rule	$500	
chk_stdtc	Start Date	$10	
chk_endtc	End Date	$10	

Note on chk_order: The order of the checks would be primary used to improve the readability of the check report. The order can be assigned per program to avoid renumbering every check each time a new check is added.

The chk_decision table for the five checks of our example looks as follow:

chk_definition

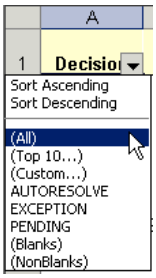
	chk_type	chk_std	chk_name	chk_order	chk_expval	chk_except	chk_rule	chk_stdtc	chk_endtc
1	codelist	all	chk_codlst_dup	1001	No duplicate	N	More than one code per codelist is not allowed in SAS.	2013-06-23	
2	codelist	all	chk_type_codmeta	1002	Same type for both codelist and variable	N	The type of a variable must match the type of the ass...	2013-06-23	
3	codelist	g	chk_extensible_ny	1003	Entry exists in SDTM CT	N	According to SDTM Controlled Terminology, the code...	2013-06-23	
4	metadata	all	chk_outform_across	2001	Common outform (w.d)	N	Variables using the same codelist should have the sa...	2013-06-23	
5	metadata	all	chk_type_metacod	2002	Same type for both codelist and variable	N	The type of a variable must match the type of the ass...	2013-06-23	

GENERATE AN .XLS OUTPUT

A standard program is called within each standard to generate an .xls check report. The structure of the SAS data set converted in .XLS file is as follow:

Variable Name	Variable Label		Key
decision	Decision	$15	(1)
decision_dtc	Decision Date	$10	(1)
decision_userid	Decision UserID	$5	(1)
decision_note	Decision Note	$200	(1)
chk_name	Check Name	$32	(1)
criteria_id1	Status	$40	(1)
criteria_id2	FMTNAME / Data Set	$40	(1)
criteria_id3	START / Variable	$40	(1)
chk_actval	Actual Value	$1000	(1)
chk_expval	Expected Value	$1000	
chk_rule	Check Rule	$1000	

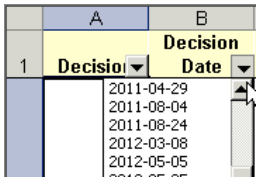
The first four columns of the .xls file are updated by the user if the entry cannot be resolved.



decision — Possible values are:

- EXCEPTION: no further action needed.
- AUTORESOLVE: query will disappear when another standard is updated.
- PENDING: the resolution of the query can be postnoted to a later point. The release of the standard can already take place.

New queries can be filtered using the Decision column searching for Decision=<blank>



decision_dtc — The decision date is stored in an ISO 8601 format (yyyy-mm-dd). Sorted dates appears in a chronological order in the Excel filter.

decision_userid — The decision user ID is the User Identifier of the reviewer.

decision_note — The decision note provides a short explanation about the decision.

Below is a codelist check report.

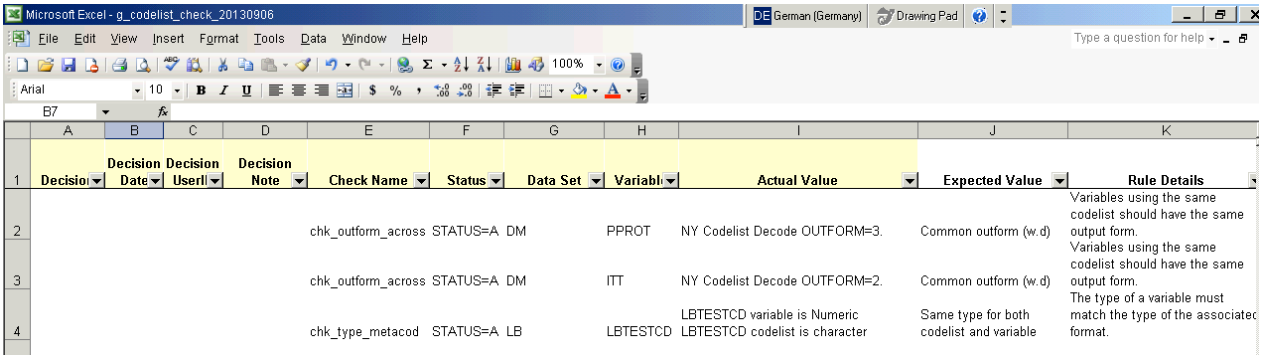
- The report shows duplicates in LBTESTCD codelist. The duplicate entry needs to be removed (chk_codlst_dup).
- The type of the LBTESTCD variable will be changed to character. The decision is to set to AUTORESOLVE. (chk_type_codemta).
- The NY codelist contains a value not allowed by SDTM controlled terminology. The codelist was deactivated, meaning that from now no study will use it. The entry is then flagged as EXCEPTION (chk_extensible_ny).

Microsoft Excel - g_codelist_check_20130906

	A	B	C	D	E	F	G	H	I	J	K
1	Decision	Decision Date	Decision UserID	Decision Not	Check Name	Status	FMTNAM	START	Actual Value	Expected Value	Rule Details
2					chk_codlst_dup	STATUS=A	LBTESTCD	A1AGLP	START=AMMONIA LABEL=Ammonia	No Duplicate	More than one code per codelist is not allowed in SAS.
3					chk_codlst_dup	STATUS=A	LBTESTCD	A1AGLP	START=AMMONIA LABEL=AMMONIA	No Duplicate	More than one code per codelist is not allowed in SAS.
4	AUTORESOLVE	2013-08-01	ABCDE	LBTESTCD variable will be corrected	chk_type_codmeta	STATUS=A	LBTESTCD	A1AGLP	Variable LBTESTCD is numeric. LBTESTCD codelist is character.	Same type for both codelist and variable	The type of a variable must match the type of the associated format.
5	AUTORESOLVE	2013-08-01	ABCDE	LBTESTCD variable will be corrected	chk_type_codmeta	STATUS=A	LBTESTCD	APOE	Variable LBTESTCD is numeric. LBTESTCD codelist is character.	Same type for both codelist and variable	The type of a variable must match the type of the associated format.
6	EXCEPTION	2013-08-10	ZZZZ	Code is desactivated and cannot be used by any new study	chk_extensible_ny	STATUS=X	NY		997 LABEL=NOT ASSESSABLE	Entry exist in SDTM CT	According to SDTM Controlled Terminology, the codelist is not extensible. Only submission values specified in SDTM CT are allowed

The table below is a snapshot of a metadata check report.

- It indicates that the output format is not consistent across variables. Either the value is wrong and needs to be updated or the name of the codelist is wrong for one of the variable. Once outform attribute by PPROT variable is changed to value 2. both the queries will disappear.
- It reports inconsistencies in type: the LBTESTCD variable is numeric when the attached format LBTESTCD is character. Once the LBTESTCD variable type is changed to character the query is removed.



	A	B	C	D	E	F	G	H	I	J	K
1	Decisio	Decision Date	Decision UserI	Decision Note	Check Name	Status	Data Set	Variabl	Actual Value	Expected Value	Rule Details
2					chk_outform_across	STATUS=A	DM	PPROT	NY Codelist Decode OUTFORM=3.	Common outform (w.d)	Variables using the same codelist should have the same output form.
3					chk_outform_across	STATUS=A	DM	ITT	NY Codelist Decode OUTFORM=2.	Common outform (w.d)	Variables using the same codelist should have the same output form.
4					chk_type_metacod	STATUS=A	LB	LBTESTCD	LBTESTCD variable is Numeric LBTESTCD codelist is character	Same type for both codelist and variable	The type of a variable must match the type of the associated format.

STORE PERMANENTLY THE CHECK DEFINITIONS

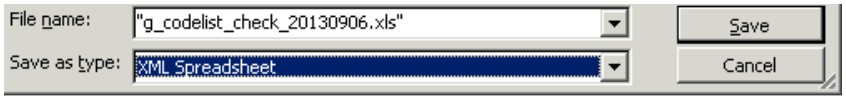
In the codelist check report, the decisions AUTORESOLVE and EXCEPTION needs to be stored for them to be available at next program run.

chk_decision

	chk_type	decision_dtc	decision_userid	decision_note	chk_name	criteria_id1	criteria_id2	criteria_id3	chk_actval
1	codelist	2013-08-01	ABCDE	Type in LBTESTCD variable will be corrected	chk_type_codmeta	STATUS=A	LBTESTCD	A1AGLP	Variable LBTESTCD is numeric. LBTESTCD codelis
2	codelist	2013-08-01	ABCDE	Type in LBTESTCD variable will be corrected	chk_type_codmeta	STATUS=A	LBTESTCD	APOE	Variable LBTESTCD is numeric. LBTESTCD codelis
3	codelist	2013-08-10	ZZZZ	Code is desactivated and cannot be used by any new...	chk_extensible_ny	STATUS=X	NY	997	LABEL=NOT ASSESSABLE

Unless a solution is found to import the .xml file generated by ODS TAGSETS.EXCELXP as SAS dataset, a solution must be found to store the decisions: either the .xml file is saved as Excel or the decisions (yellow columns) are copied and pasted in a permanent Excel file. The Excel file can then be imported as SAS dataset with proc import.

Before



After

For each standard where the checks are running, a chk_decision table is stored. If a decision for a standard can be generalized to all standards, then the chk_decision table at the upper level standard is updated and not the one where the decision was made. The next time another standard experiences the same query, the decision column will already be filled in, leaving less entries to review.

REDUCE THE VOLUME OF CHECKS BY IMPROVING THE MAINTENANCE PROCESS

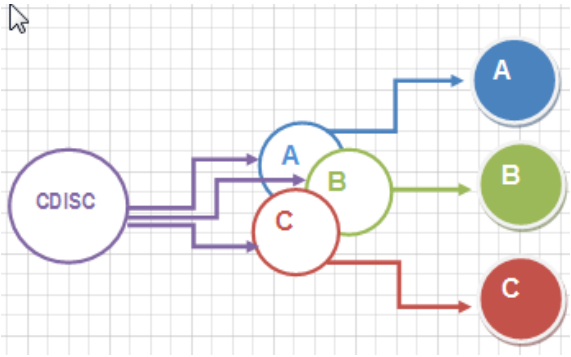
The amount of checks to program and queries to review can be reduced by splitting the input table. For metadata it can mean two things:

1. CDISC attributes and global attributes (common to all standards) are maintained by a limited number of persons in one place. The few remaining standard specific attributes are maintained locally by each standard representative (e.g. variable origin).
2. All the variables do not need to be listed in each and every standard.
    a. Variable: For example some timing variables should be present in the domain when the related date is selected for the standard e.g. –STDY when –STDTC variable is present, etc.
    b. Domain: Some domains are needed for all standards e.g. trial design domains TA, TE, TI, TS, TV but also inclusion/Exclusion criteria IE, comments CO and protocol deviations DV.

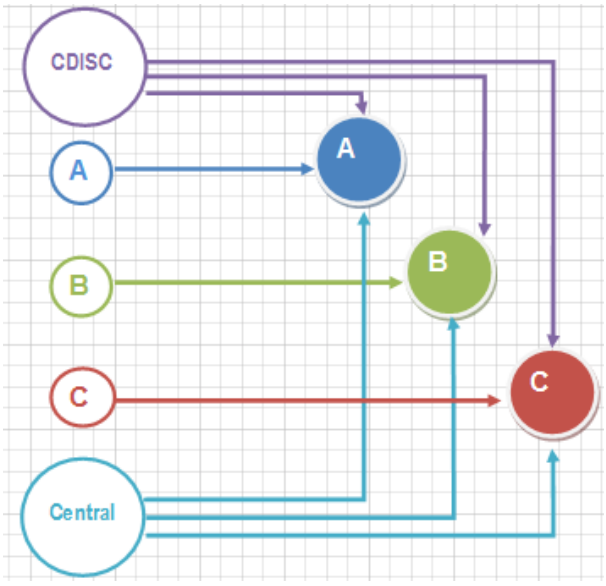
In the first scenario the three standards (A, B and C) have overlaps e.g. the label of the same variable could be maintained in standard A, standard B and standard C i.e. three times. The CDISC attributes (cdisc_notes, cdisc_role, cdisc_core) are also maintained in each of the input files.

In the second scenario the attributes common to several standards are maintained centrally (variable sequencing, label, type, outform, codelist name, etc.). The CDISC attributes are going directly from the referenced tables to the output standards. The amount of attributes/variables which are standard specific is minimized.

Scenario with Repetitive Task



Scenario without Duplicated Task



NOTE ON CODELISTS

When a new variable is created, the list of possible values are not always known upfront, onlyt the name and the type. A central table can store the codelist type of all existing codelists. The standard specific input codelist table contains the remaining information. The output file is then a merge of the standard specific codelist table and the central codelist table.

CONCLUSION

Automatic checks can be used to reduce the amount of mistakes in standards and give more time to the users for more knowledgeable tasks. The review of the generated queries is facilitated by a .xls file with filters which can be annotated. The work done in previous review is kept. There is no need to start from scratch at each new run of the program.

In this paper, checks definitions are applied to internal tables for the purpose of the demonstration assuming external sources are correct. Any source data can be controlled the same way whether it is the CDISC SDTM IG or the SDTM controlled terminology.

Checks should also cover the whole spectrum: check individual variable, check a combination of variables, check across tables and check across standards.

Controlling is still time consuming. Every effort should be put upfront to avoid the query to pop-up by blocking any possibility to enter invalid entries, auto correction of invalid entries and build a structure which require a single action when a change needs to be implemented across standards.

CONTACT INFORMATION (HEADER 1)

Your comments and questions are valued and encouraged. Contact the author at:

Véronique Bourcier
Bayer Healthcare
Building K 9, 365
D-51366 Leverkusen
veronique.bourcier@bayer.com
www.bayer.com