Paper AD04

CDISC for electronic submissions - A Table Translation Program

John H. Adams, Boehringer Ingelheim Pharmaceutical, Inc., Ridgefield, CT
Kirill Tchernakov, Boehringer Ingelheim Pharmaceutical, Inc., Ridgefield, CT

ABSTRACT

This paper outlines a SAS® program that automatically translates a pharmaceutical company's data to the CDISC submissions data standards (SDS) model format for FDA e-submissions. The program was added as a new function, called CDISCDOMAIN, to the XPDL system. A paper by John Adams, 'XPDL – An Extensible Project Database Loading and Table Translation Program', was previously presented at the NeSug 2003 and PharmaSUG 2003 conferences .

The new CDISCDOMAIN macro program , along with another XPDL function, can easily translate one or more sources of data into a CDISC standard domain. So no matter what your current data format structure or format is, you can quickly translate it into the CDISC format, without programming resources!

Generic templates for each CDISC domain were developed to reflect the SDS model. In their pure form, they contain the metadata that represents the CDISC requirements. Then by adding some additional information, such as variable attribute and sourcing information we created EXCEL template sheets that are used to directly drive the CDISCDOMAIN program. When CDISC models are updated in the future, only templates need to be updated, not the program.

INTRODUCTION

A drug project in the pharmaceutical industry typically produces a lot of data that must be analyzed for efficacy, safety, drug interactions, demographics, etc. before reports are prepared for submission to the FDA. These days, such submissions must be made electronically, sometimes call e-submissions. All data used to support these e-submissions must also be shipped to the FDA electronically.

CDISC is an industry consortium that is establishing standards for the exchange of digital information. The FDA has endorsed the CDISC standards approach to providing data in e-submissions. By submitting tabulations that conform to the standard structure, the industry can benefit by no longer having to submit separate patient profiles. Reviewers also benefit by only needing training in the principles of standard datasets and the use of standard software tools.

Unfortunately, most pharmaceutical companies do not keep their data in the CDISC standard format. Therefore, there was always a crunch at submission time to transform their internal data structure to the desired FDA format. This effort required a lot of ad-hoc programming and data verification.

XPDL, a dynamic SAS® macro program, makes this task easy (see the referenced PharmaSug 2003 publication on XPDL). The program can combine and re-map data for any number of trials to create a translated CDISC domain without custom programming. XPDL uses tables (EXCEL® spreadsheets) to control the process of combining and re-mapping of trial data. These tables contain the metadata that describes the source-to-target data relationships and any required re-mapping of variables.

The newly added XPDL function called CDISCDOMAIN is the macro program that, along with another XPDL function, translates the source data into a CDISC standard domain. So no matter what your current data format structure or format is, you can quickly translate it into the CDISC format, without programming resources!

Generic templates for each CDISC domain were developed to reflect their SDS model. In their pure form, they contain the metadata that represents the CDISC requirements. Then by adding some additional information, such as variable attribute and sourcing information we created EXCEL template sheets that are used to directly drive the CDISCDOMAIN program. When CDISC models are updated in the future, only templates need to be updated, not the program.

By using the appropriate XPDL functions a lot of programming resources will be saved, not only during the initial e-submission , but also on future filings. The savings in time and quality are immense.

The code behind this program is quite complex and lengthy and therefore will not be shown or explained in this paper.

1.0 PRELIMINARIES

This type of application is not, by the nature of the complexities, be a 'push button' operation. It will require some human intervention, choices to be made, etc. CDISCDOMAIN automates as many tasks as possible. So, before starting on a mission to establish a CDISC domain, we must first do some planning and preparation. Normally, the generic CDISC templates for the domains are prepared only once and can then be shared by all trials and/projects.

When we wish to create an actual CDISC domains for our submission, we must take a copy of it's generic template and complete the rest of the template to make them trial or drug project specific. Before starting the process for each domain, we must, of course, decide which CDISC standard domains are appropriate for our submission. Keep in mind that we may also have to provide additional data to the FDA for non-standard domains, if the submission warrants it.

For a given domain, we must first decide which of the non-core' variables need to be included in that domain in order to support the submission. We must also identify the various data sources and their locations for each of the selected domain variables (core + optional). Sources can be any combination of SAS® views/datasets at local or remote locations.

Furthermore, we need to decide on the necessary derivations for variables that are not directly available. We also must fill in the 'ACTION keywords' in the template, where necessary, to control the formation of the domain variable map. Actual SAS® code for derivations can also be entered in the appropriate field. This will drive the automatic generation of SAS® macros.

The creation of this final pre-pared template is extremely important as it represents the 'cook book', used by CDISDOMAIN, for creating a CDISC domain variable map. This CDISC domain variable map is later used by another XPDL function (PDBLOAD) to actually create the actual CDISC domain SAS® dataset. The 'road map' depicted in Fig. 2 shows this process.

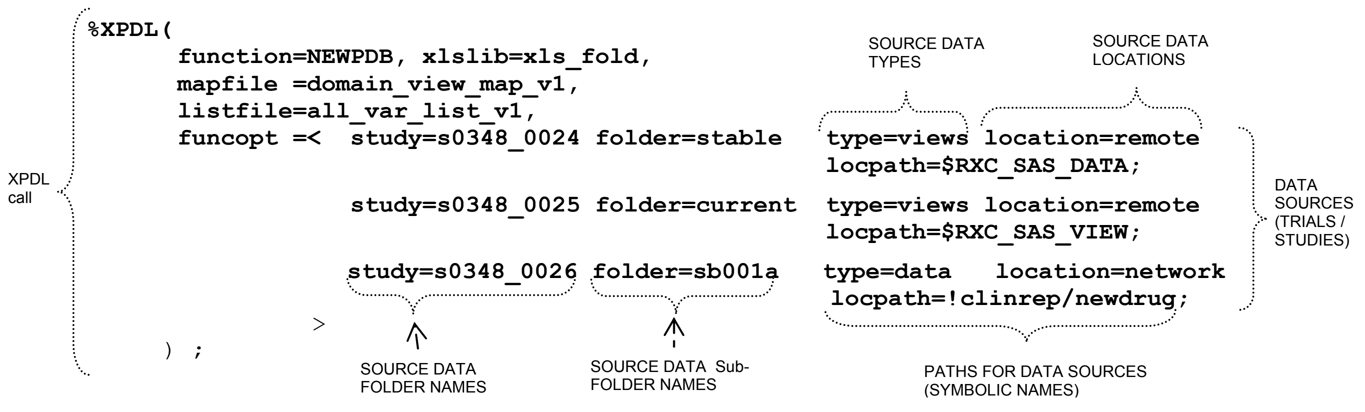
2.0 FUNCTIONALITY

XPDL has 15 major functions. The table in Fig. 1, however, only shows those functions that are germane to this paper:

Fig. 1 – XPDL Functions used for creating CDISC domains

FUNCTION	PROGRAM PURPOSE	PROGRAM FUNCTIONS
NEWPDB	Starts a new database	• compare trial views/ and variables • create an all variable list • create a domain-view map
CDISCDOMAIN	Makes a new CDISC domain	• read CDISC template • read the all variable list and domain-view map • create a CDISC domain-variable map • flag problems to be corrected / edited
LOADPDB	Loads data to PDB	• build a dynamic load pgm • run the load program to create the dataset(s)

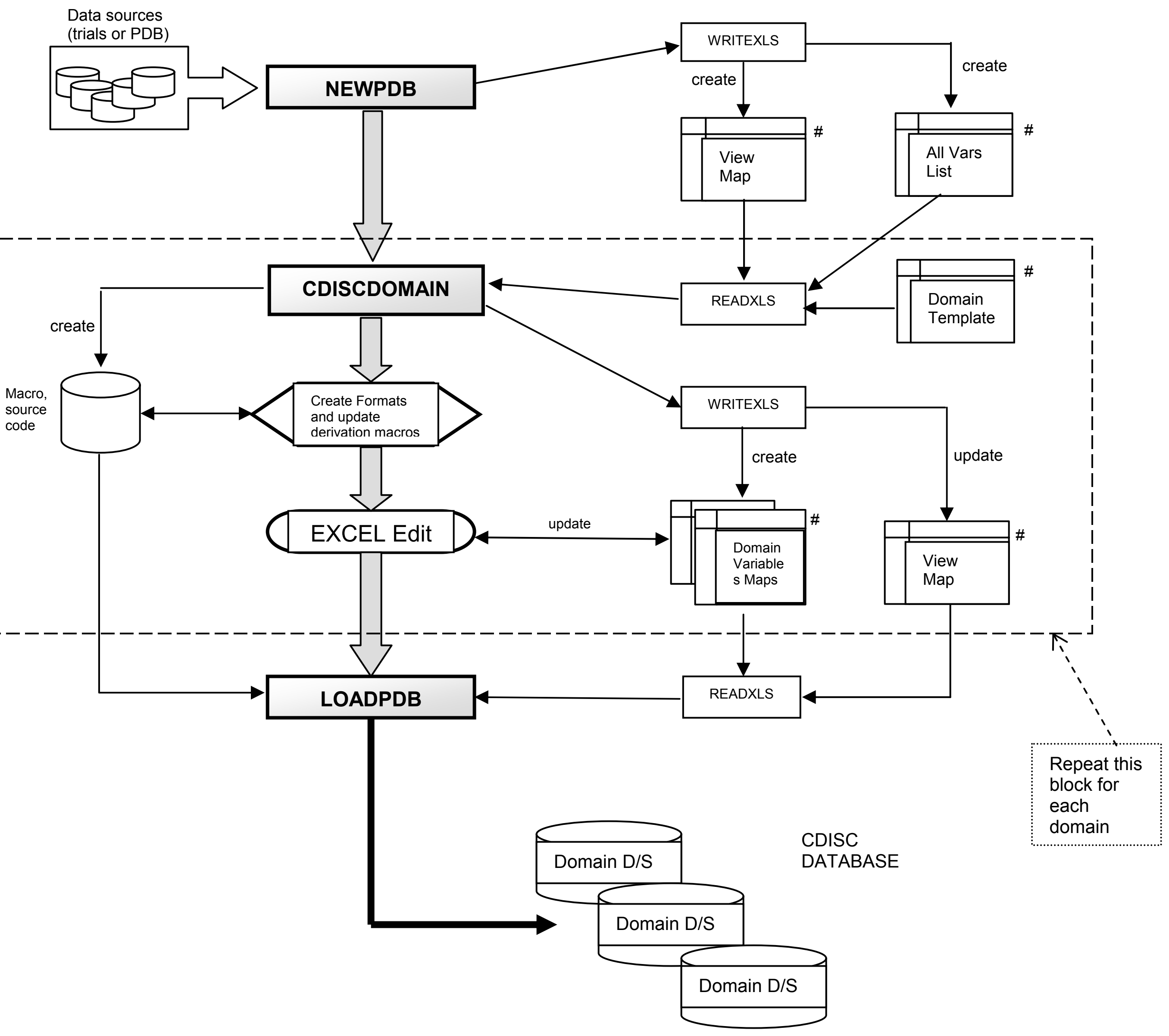
Let's look at the steps needed to create a CDISC database, as shown in roadmap of figure 2.
We must first start with a one-time call to the NEBPDB function. A sample call with the NEWPDB function is relatively simple:

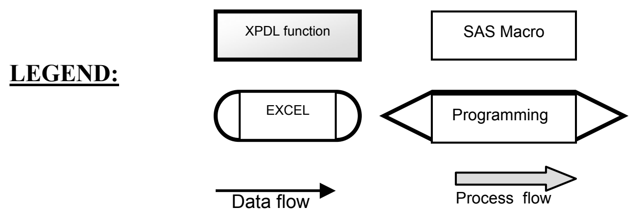


```
%XPDL(
      function=NEWPDB, xlslib=xls_fold,
      mapfile =domain_view_map_v1,
      listfile=all_var_list_v1,
      funcopt =<  study=s0348_0024  folder=stable    type=views  location=remote
                                                      locpath=$RXC_SAS_DATA;

                  study=s0348_0025  folder=current   type=views  location=remote
                                                      locpath=$RXC_SAS_VIEW;

                  study=s0348_0026  folder=sb001a    type=data   location=network
                                                      locpath=!clinrep/newdrug;
               >
      ) ;
```

SOURCE DATA TYPES
SOURCE DATA LOCATIONS
XPDL call
DATA SOURCES (TRIALS / STUDIES)
SOURCE DATA FOLDER NAMES
SOURCE DATA Sub-FOLDER NAMES
PATHS FOR DATA SOURCES (SYMBOLIC NAMES)

FIG. 2 – ROADMAP FOR CREATING CDISC DOMAINS





Note: #1, #2, #3, #4 are the sheet type numbers as described in Figure 3-5

A XPDL call with the NEWPDB function creates two types of XLS spreadsheets, as seen in Figures 3 and 4.

Fig. 3 – Layout of the 'all variables list' sheet as produced by the NEWPDB function (# 1)

Column Name	Column Description	Note
Domain	Name of project domain	Fixed column names
Viewname	Study view / dataset name	
Varname	Study variable name	
Varlabel	Study variable label	
STUDY1-STUDYn	Study1_foldername- Studyn_foldername	One set per Study (n= # studies)
VARTYP1-VARTYPn	Study1_variable_type- Studyn_variable_type	
VARFMT1-VARFMTn	Study1_variable_fmt- Studyn_variable_fmt	
VARLEN1-VARLENn	Study1_variable_length- Studyn_variable_length	
SDYLOC1-SDYLOCn	Study1- Studyn data_location [remote or network]	
SDYPAT1-SDYPATn	Path_to_Study_data_locations	
SDYLVL1-SDYLVLn	Subdirectory_folder_of_Study_folders	

<u>**NOTES:**</u>

1. This sheet contains information about all available source data variables, their attributes and their locations.
2. This sheet is used by XPDL functions and is normally not edited by the user.

Fig. 4 – Layout of the 'domain-view map' sheet as produced by the NEWPDB function(# 2)

Column Name	Column Description	Note
Domain	Name of project domain	
Active	YES or NO, is domain actively loaded?	
Desc	Description of domain	
Dvarmap	Name of domain variable map spreadsheet	
Viewname	Name of study view or dataset	
SDYFOL1-SDYFOLn	Study1_foldername- Studyn_foldername	One set per Study (n= # studies)
SDYLVL1-SDYLVLn	Study1_sub_foldername- Studyn_sub_foldername	
SDYLOC1-SDYLOCn	Study1_folder_location- Studyn_folder_location	
SDYPAT1-SDYPATn	Study1_folder_path- Studyn_folder_path	

<u>**NOTES:**</u>

1. This sheet contains summary information about all available source data views/datasets and their locations.
2. This sheet is created initially by XPDL and at times user edited. It is used by XPDL functions.

The next step on our roadmap (Fig.2) to creating a CDISC domain is to run the CDISCDOMAIN function. This function will create a domain variables map based on the information in a template. Before we do that, however, we must first create a this domain template sheet (# 4). Figure 5 shows the layout of a template.

Fig. 5 – Layout of the of CDISC domain template (# 4)

Column Name	Column Description	Note
Domain[3]	Name of project domain	Domain variable attributes
Prjvar[2,3]	Name of project domain variable	
Prjvlbl	Project domain variable label	
Prjkey	INDEX, MERGE, MERGE1,etc	
Prjtyp	Project domain variable type (C,N)	
Prjfmt	Project domain variable format	
Prjlen	Project domain variable length	
Prjvdef	Action key for project domain variable	PrjVar Action
Pdyview	Study view / dataset name	Data Source Information
Pdyvar	Study view / dataset variable name	
Corevar	Is variable required? (Y,N)	General Information
Prjvcom	Comments and notes	

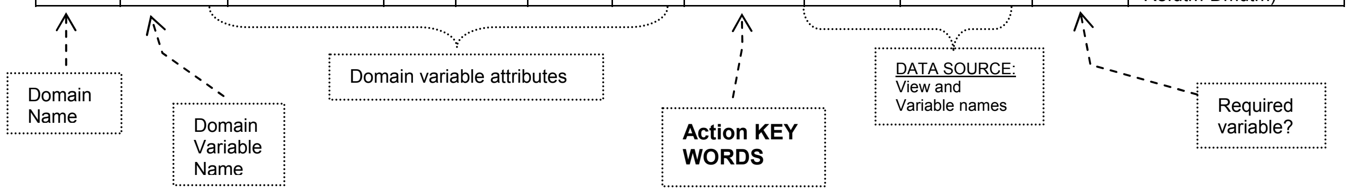
NOTES: 1. These sheet are created and heavily edited by the user. They are used by XPDL function(s).
2. Names that start with a '?' are not active and will not be included in the final domain.
3. Do not place any comments in these columns (Domain and Prjvar) below the regular row.

Let's look at an actual sample of a completed 'DM' CDISC domain template (see Fig. 6). The figure is well annotated as to what it's columns are and how they are used in the creation of a domain variable map, which will be used to translate the data.

Fig. 6 – Sample of a completed CDISC DM domain template (# 4)

DOMAIN	PRJ VAR	PRJVLBL	PRJ KEY	PRJ TYP	PRJ FMT	PRJ LEN	PRJ VDEF	PDY VIEW	PDYVAR	CORE VAR	PRJVCOM
DM	Studyid	Study Identifier		C	$15.	15		PATD	STUDY	Y	
DM	Domain	Domain Abbreviation		C	$2.	2	_DEFINE_	PATD	DM	Y	CDISC uses a 2 character domain name
DM	Usubjid	Unique Subject Identifier		C	$30.	30		PATD	UNQPTNO	Y	Unique identifier within the submission
DM	Subjid	Subject Identifier for the study		C	$30.	30		PATD	PTNO	Y	Often used as ID of the subject within the study
DM	Refdtm	Subject Reference date/time		N	8.	8		E_TRT EXP	ATRSTDT	Y	Time when subject entered trial (in seconds from 01jan1960)
DM	Refdtmp	Refdtm Precison		N	8.	8	_DEFINE_	PATD	60	Y	Precision of Refdtm= Minute (units = seconds)
DM	Siteid	Study Site Identifier		C	$10.	10		PATD	INVSITE	Y	
DM	Invade	Investigator Identifier		C	$10.	10	_DELETE_			N	Not needed if Invade=Siteid
DM	Invar	Investigator		C	$20.	20		PATD	INVNAME	N	

DOMAIN	PRJ VAR	PRJVLBL	PRJ KEY	PRJ TYP	PRJ FMT	PRJ LEN	PRJ VDEF	PDY VIEW	PDYVAR	CORE VAR	PRJVCOM
		Name									
DM	Brthdtm	Date/Time of Birth		N	8.	8		PATD	BTHDT	N	Time when subject was born (in seconds from 01Jan1960)
DM	Brthdtmp	Precision of Birthdtm		N	8.	8	_DEFINE_	PATD	86400	N	Precision of BTHDT=Day (units=seconds)
DM	Age	Age at REFDTM		N	8.	8	_CODE_	PATD	Age= REFD TM-BRTHDTM;	Y	Age at REF date/time (derived:REFDTM-BRTHDTM) in AGEU units
DM	Ageu	Age Units		C	$8.	8	_DEFINE_	PATD	YEARS	Y	Units for age [YEARS, MONTHS, DAYS]
DM	Sex	Sex		C	$1.	1		PATD	SEX	Y	Male,Female,Unknow n [M,F,U]
DM	Race	Race		C	$20.	20		PATD	RACEP	Y	May become optional in the future
DM	Ethnic	Ethnicity		C	$20.	20		PATD	MIXRAC	N	Ethneticity of subject
DM	Trtcd	Treatment Code		N	8.	8		PATD	PRJTRT	Y	Treatment code - Numeric version of Trtgrp
DM	Trtgrp	Treatment group		C	$40.	40		PATD	TPATT	Y	Treatment group
DM	Country	Country		C	$20.	20		PATD	COUNTRY	Y	Country where subject participated in trial
DM	Weight	Weight in kilograms		N	8.	8		PATD	WTSTD	N	Weight in kilograms
DM	Height	Height in centimeters		N	8.	8		PATD	HTSTD	N	Height in centimeters
DM	Complt	Completers Population		C	$1.	1		TTM	PTERM	N	Subject completed study? [Y,N]
DM	Safety	Safety Population		C	$1.	1		POPU	POPU	N	Subject included in safety population? [Y,N]
DM	Itt	Intent to treat		C	$1.	1		RAND	RANDELP	N	Subject randomized for treatment? [Y,N]
DM	Pprot	Protocol population		C	$1.	1		POPU	POPUNY	N	Subject included in protocol analysis dataset? [Y,N]
DM	Visit	Visit name		C	$20.	20		PATD	CPEVENT	N	May be dropped in the future
DM	Visitnum	Visit number		N	8.	8		PATD	ACTEVENT	Y	Use Visitnum, Visitdy or both (at least one is required)
DM	Dmdtm	Data collection Date-time		N	8.	8		PATD	VISDT	Y	SAS date-time when demo was collected (in seconds)
DM	Dmdtmp	Dmdtm precision		N	8.	8	_DEFINE_	PATD	86400	Y	Precision of Dmdtm=day (units = seconds)
DM	Dmdy	Data collection day		N	8.	8	_DERIVE_	PATD		Y	Day # relative to Refdtm (derived: Refdtm-Dmdtm)



The Action Key words in the PRJVDEF column of the template controls the process in the CDISCDOMAIN function. The summary table in Fig. 7 shows how the various Action Key words and how they affect and control the tasks of the function.

ACTION KEY WORD

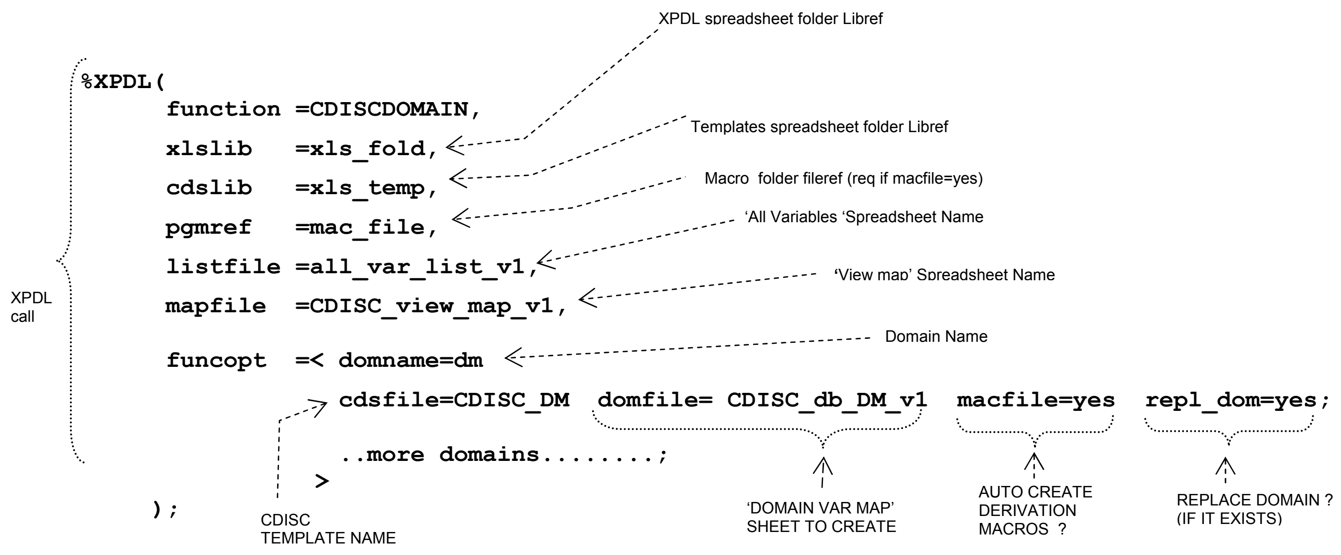
Fig. 7 – Action Key words descriptions

COLUMN NAME in the TEMPLATE SHEET				DESCRIPTION
PRJVDEF[3]	**PRJVAR[1,3,4]**	**PDYVIEW[3]**	**PDYVAR[3]**	
blank	*Project varname*	Source *viewname*	Source *varname*	A PRJVAR is directly sourced from a variable (named in PDYVAR field) in view/dataset (named in PDYVIEW field)
CODE	*Project varname*	Source *viewname*	SAS code	A 'Study-View' derivation macro will be automatically created with the actual SAS code from the PDYVAR field. (each statement ends with a ';'). This macro will be called during processing of the defined PDYVIEW.
DELETE	*Project varname*	blank	blank	PRJVAR will not be in the final Domain (Can only be used where COREVAR=N)
DEFINE	*Project varname*	Source *viewname*	blank	A 'Study-View' derivation macro will be automatically created which will set PRJVAR to a constant value. This macro will be called during processing of the PDYVIEW.
DERIVE	*Project varname*	*viewname*	blank	A 'Study-View' derivation macro shell will be automatically created (containing only a comment). You must code the actual macro logic before using the domain. This macro will be called during processing of the PDYVIEW.
NOSOURCE	*Project varname*	blank	blank	This PRJVAR will be in the final domain but will not be sourced (it will always be empty).

NOTES:

1 PRJVAR names that start with a '?' are not active and will not be included in the final domain varmap.
2 Each row in the template must have entries in all fields (PrjVdef , PrjVar, PdyView, PdyVar), unless 'blank' is shown.
3 Use only single values in these columns for each row.
4 Do not place any comments in columns Domain and Prjvar below the regular rows

So now that we have a completed template (like that in Fig. 6), we are ready to execute the CDISCDOMAIN function. Following is a sample call to create a CDISC DM domain variables map (with a layout as per Fig. 8) with a template:



```
%XPDL(
      function =CDISCDOMAIN,
      xlslib   =xls_fold,
      cdslib   =xls_temp,
      pgmref   =mac_file,
      listfile =all_var_list_v1,
      mapfile  =CDISC_view_map_v1,

      funcopt  =< domname=dm
                  cdsfile=CDISC_DM  domfile= CDISC_db_DM_v1  macfile=yes  repl_dom=yes;

                  ..more domains........;
                 >
    );
```

XPDL spreadsheet folder Libref
Templates spreadsheet folder Libref
Macro folder fileref (req if macfile=yes)
'All Variables 'Spreadsheet Name
'View map' Spreadsheet Name
Domain Name
XPDL call
CDISC TEMPLATE NAME
'DOMAIN VAR MAP' SHEET TO CREATE
AUTO CREATE DERIVATION MACROS ?
REPLACE DOMAIN ? (IF IT EXISTS)

Notes: 1. Each line in FUNCOPT defines a domain and ends with a ';' . The template specifies the structure and sourcing of the final domain.
2. CAUTION- If you use CDISCDOMAIN again for an existing domain with the same 'domain variables' map sheet name and repl_dom=yes the sheet will be overwritten.
3. This function updates the view map automatically and makes the domain 'active'.
4. This function creates the macro source file automatically and creates the macro links in the variables map (if macfile=yes).

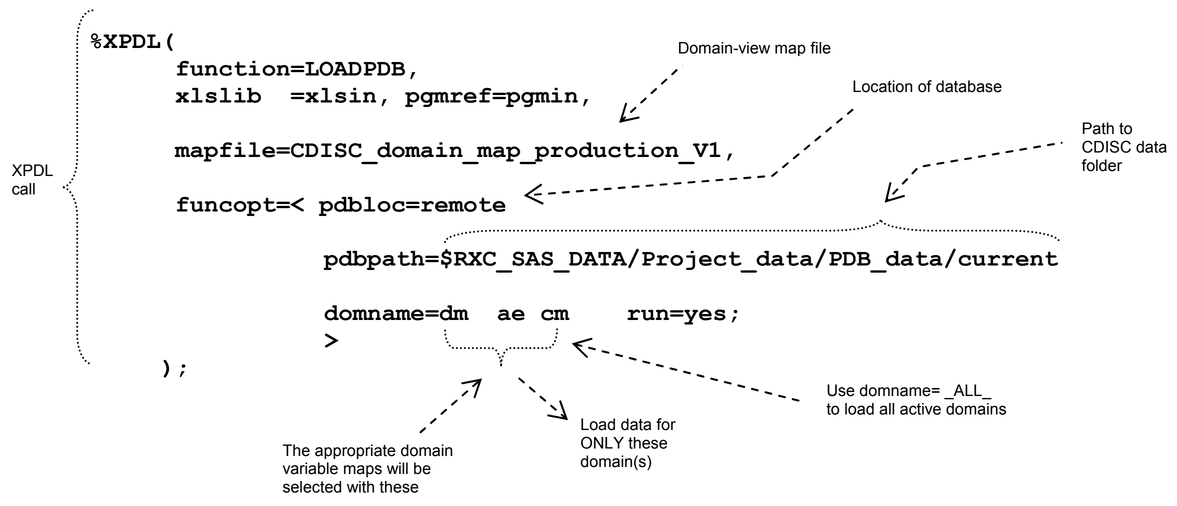
The CDISCDOMAIN function produces a domain variable map based on the specified template. Keep in mind that it is this map that will control the actual data loading of the final CDISC domain later on. Fig. 8 shows a layout of a domain variable map sheet.

Fig. 8 – layout of a domain variables map produced by CDISCDOMAIN function (# 3)

Column Name	Column Description	Note
Domain[3]	Name of project domain	Fixed column names
Prjvar[2, 3]	Name of project domain variable	
Prjkey	=INDEX if variable has needs to be indexed	
Prjvcomp	Comparison / error edit flag	
Prjvlbl	Project domain variable label	
Prjtyp	Project domain variable type	
Prjfmt	Project domain variable format	
Prjlen	Project domain variable length	
Sdyview	Study view / dataset name	
Sdyvar	Study view / dataset variable name	
Sdylbl	Study view / dataset variable label	
STUDY1-STUDYn	Study1_foldername- Studyn_foldername	One set per Study (n= # studies)
VARTYP1-VARTYPn	Study1_variable_type- Studyn_variable_type	
VARFMT1-VARFMTn	Study1_variable_format- Studyn_variable_format	
VARLEN1-VARLENn	Study1_variable_length- Studyn_variable_length	

NOTES: 1. This sheet is created initially by the CDISCDOMAIN function and then heavily edited by the user. It is later used by the PDBLOAD function to create the actual CDISC dataset.
2. Names starting with a '?' are not active and will not be included in the final domain
3. Do not place any comments in these columns (Domain and Prjvar) below the regular rows

You must edit the domain variable map (created by the CDISCDOMAIN function) to make any necessary changes in the metadata. Once the domain variable map is final and considered correct, we can make a XPDL call with the PDBLOAD function to actually create the CDISC domain dataset. Following is a sample call to load three domains:



```
%XPDL(
      function=LOADPDB,
      xlslib  =xlsin, pgmref=pgmin,

      mapfile=CDISC_domain_map_production_V1,

      funcopt=< pdbloc=remote

                 pdbpath=$RXC_SAS_DATA/Project_data/PDB_data/current

                 domname=dm  ae  cm    run=yes;
                 >
     );
```

XPDL call
Domain-view map file
Location of database
Path to CDISC data folder
Use domname= _ALL_ to load all active domains
The appropriate domain variable maps will be selected with these
Load data for ONLY these domain(s)

3.0 REAL EXAMPLES

In this section we would like to illustrate the CDISC table translation process by presenting a ‘real life’ example. It involves a recent electronic submission (including data in CDISC format) to the FDA for one of our drug projects. Since we already had our project database (consisting of multiple trials) built (with XPDL) for this drug, we decided to use it as a direct data source instead of going to each particular trial data that constitutes the project.

Our first step involved running the NEWPDB XPDL function in order to start a new CDISC database:

```
libname   xlsin  'S:\MEDICAL\data\SAS\CLINREP\IND\CDISC_pdb\data\test';

%XPDL(
      function =NEWPDB,  xlslib=xlsin, runopt = mprint debug = yes,
      mapfile  =CDISC_domain_map_v1,       listfile=CDISC_db_list_v1,
      funcopt  = <
                   study= test   folder=pdb20040127  type=data
                  location=remote
                  locpath=/u05/home/ocpaps/sas_data/ocprus/s1182_PDB;
                 >
      );
```

The above XPDL macro call caused the following (please see the referenced articles for a full NEWPDB description):

1. Connected to original data source (our existing project database) on our remote UNIX server (using locpath and location parameters).
2. Selected the folder and subfolder where the source datasets resides (by specifying study and folder parameters).
3. Specified the type of data to use (type=data means SAS datasets, whereas type=view means SAS views into Oracle Clinical tables).
4. Created maps of the data sources relationships in two Excel spreadsheets :
    a) the ‘domain-to-view’ map (CDISC_domain_map_v1)
    b) the ‘all-variables’ map (CDISC_db_list_v1). These data relationships are also referred to as metadata.

The sample XPDL call with the NEWPDB function produced a 'domain-to-view' map with a lay-out as per Fig 4. Essentially, the metadata in this map contains the data source names and their location attributes. Some of this information will be edited by the user. The contents of the partial map for this example are shown in Fig.9 .

Fig 9. 'Domain-to-view' map generated by the NEWPDB function

DOMAIN	ACTIVE	DESC	DVARMAP	VIEWNAME	SDYFOL1	SDYLOC1	SDYLVL1	SDYPAT1
ACTG	NO			ACTG	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
ACTGADQ	NO			ACTGADQ	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
ADM	NO			ADM	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
ADQ	NO			ADQ	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
AEAEA	NO			AEAEA	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
HIVAIDS	NO			HIVAIDS	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
AIDSILL	NO			AIDSILL	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
ARV_MED	NO			ARV_MED	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
BDYC	NO			BDYC	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
CTCT	NO			CTCT	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
COMP	NO			COMP	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
CPUG	NO			CPUG	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
DIARY	NO			DIARY	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
PATD	NO			PATD	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
POPU	NO			POPU	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
RAND	NO			RAND	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
TTM	NO			TTM	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
ECG	NO			ECG	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB

Please note that the DOMAIN and VIEWNAME columns in the above **'**Domain-to-view' map show all available project database (PDB) data sources (views or datasets) for our drug project. The other columns, named SDYPAT1, SDYFOL1, SDYLVL1 and SDYLOC1, represent additional information about these data sources, e.g. path, folder name, sub-folder name and data location (*remote* or *local*), respectively.

Since our data source was a project database, the above table only shows a single data source like a single trial (we only have column names with a 1 suffix). Please note that none of the CDISC domains are active yet (active=NO) in the above example, as have not created any of the required 'domain variable maps' yet (this will be done later with the CDISCDOMAIN function).

The second Excel® spreadsheet of metadata that was created simultaneously by the NEWPDB call is the 'all-variables' map (CDISC_db_list_v1.xls). Essentially, the metadata in this map contains the 'variables' attribute information about all variable in the data sources. FIG. 10 shows the partial content for the example.

Please note that the first two columns on the left (VIEWNAME and VARNAME) show the original PDB variable and domain names. The last four columns on the right represent one study set (study=test, in this case) and carry variable attribute information about each variable (type, format, length). There can be many study sets (our sample only shows one because we used a PDB as source).

This limited metadata example shows a number of source variables from two domains (PKBLD and POPU PDB). For instance, variable TMDIF comes from the PKBLD domain and has the following attributes: numeric type with a format of '20.' and length of '8' . Variable ATPATT is sourced from the POPU domain and has the following attributes : character type, '$8.' as a format and '8' as a variable length. Please note that this map is used by other XPDL function and should never edited by the user.

Fig 10. ‘All-variables’ map generated by the NEWPDB function.

VIEWNAME	VARNAME	VARLABEL	STUDY1	VARTYP1	VARFMT1	VARLEN1
PKBLD	TMDIFF	Time Difference: Actual-Planned	test	N	20.	8
PKBLD	TMDIFFR	Time difference actual - planned RTV	test	N	20.	8
PKBLD	TMDIFFT	Time Difference Actual - Planned TPV	test	N	20.	8
PKBLD	TMDIFFU	Time difference unit	test	N	UNIT2F.	8
PKBLD	TMDIFFV	Time Difference Actual Planned- Vitals	test	N	20.	8
PKBLD	TMDIFUV	Time Difference Unit- Vitals	test	N	UNIT2F.	8
PKBLD	TPATT	Project Study Treatment group	test	C	$PRJTRT.	15
PKBLD	UNQPTNO	Universal Patient ID	test	C	$16.	16
PKBLD	USUBJID	Universal Patient ID	test	C	$30.	30
PKBLD	VISDT	Visit date	test	N	DATE8.	8
POPU	ADMDT	Treatment Start Date	test	N	DATE9.	8
POPU	ATPATT	Actual Treatment Group	test	C	$8.	8
POPU	ATPATTDC	Actual Treatment Group Decode	test	C	$60.	60
POPU	ATPATTSR	Actual Treatment Group Sort Code	test	C	$8.	8
POPU	ATRSTDT	Actual Treatment Start Date	test	N	DATE9.	8
POPU	DRGSTPDT	Drug Discontinuation Date	test	N	DATE9.	8
POPU	POPU	Population	test	C	$16.	16
POPU	POPUDC	Population Decode	test	C	$60.	60
POPU	POPUNY	Included in Population	test	N	YN1F.	8
POPU	POPUX	Population Comment	test	C	$200.	200
POPU	PTNO	Patient Number	test	N	10.	8
POPU	STUDY	Trial Number	test	C	$15.	9
POPU	TERML	Reason for Withdrawal	test	N	TERMC1F.	8

The next step in the CDISC process (refer to the roadmap in Fig.2) is to create a new CDISC domain variable map. This is accomplished by using the newly introduced CDISCDOMAIN XPDL function.

As it was mentioned before, the CDISCDOMAIN function reads a previously established CDISC domain template to get it’s ‘instructions’ on how to create a CDISC domain variable map. It then reads the ‘all-variables’ and domain-to-view maps to create a CDISC domain variable map, using the ‘instructions’ from the template.

This function also verifies the metadata and flags problems that need to be corrected by user editing.

Below is an example of our AE (Adverse Events) CDISC domain creation with the CDISCDOMAIN function:

```
libname  xlsin 'K:\MEDICAL\data\MEDDAT95\SAS\CLINREP\Tipranavir\CDISC_pdb\data\test';
libname  cdsin 'K:\MEDICAL\data\MEDDAT95\SAS\CLINREP\Tipranavir\CDISC_Templates';
filename pgmin 'K:\MEDICAL\data\MEDDAT95\SAS\CLINREP\Tipranavir\CDISC_pdb\macros';


%XPDL(
       function=CDISCDOMAIN, xlslib=xlsin, cdslib=cdsin, pgmref=pgmin,
       listfile=CDISC_db_list_v1,
       mapfile=CDISC_domain_map_v1,
       funcopt =< domname=ae
                  domfile=CDISC_db_AE_v1
                  cdsfile=CDISC_AE
                  macfile=yes;
                 >
      );
```

In the above call, xlslib, cdslib and pgmref are XPDL's spreadsheet folder, template folder and macro folder library references, respectively. Listfile and mapfile are the file names for our 'all-variables' and 'domain-view' maps (CDISC_db_list_v1.xls and CDISC_domain_map_v1, respectively) that were created in a previous example for the NEWPDB function. Parameter Macfile=yes means to auto-create the derivation macros for the AE domain during the XPDL macro call.

Shown in FIG. 11 below is the partial AE domain CDISC template that was used in the above call. This template is critical in the running of the CDISCDOMAIN function.

As you can notice from this figure, there are several ACTION KEYS in the AE template to control the formation of the domain variable map. For instance, in rows where PRJVAR= Domain, AeStDtm and AeEndTm the corresponding PRJVDEF values are equal to _DEFINE_. That means that constants should be establish for these three variables (as specified in PDYVAR column: ' AE', '66400' and '66400', respectively).

In other rows, you also see _DERIVE_, _CODE_ and _NO_SOURCE_ action keys specified (in the PRJVDEF column) for those project variables.

An example where PRJVDEF = _DERIVE_ (variables AeStDy and AeEnDy) means that CDISCDOMAIN will code a derivation macro for each in the SAS® AE_macro file (if macfile=yes).

The code for each derivation is defined in the PDYVAR column and described in the PRJVCOM column (if PDYVAR= empty, then an empty shell macro will be coded).

For instance, variables *AeStDy* and *AeEnDy* will be created with manually coded derivation macros, defined as follows in our AE macro file:

```
%macro DER_0001;
     AeStDy = AEONDT-REFDTM;
%mend;

%macro DER_0002;
   AeEnDy = AEENDDT-REFDTM;
%mend;
```

On the other hand, in rows (of the AE template in Fig.11) where PDYVAR=AeStDtm and AeEndTm, _CODE_ was specified in the PRJVDEF column. The actual corresponding derivation code for each PDYVAR was provided in the PDYVAR column. Thus, when running, the CDISCDOMAIN function automatically created the following two macros:

```
%macro DER_003 ;                        /** Code the Derivation for : AeStDtm  **/ ;
       hr=hour(aeontm);
       mn=minute(aeontm);
     if aeontm eq . then   do;
       hr=0;
       mn=0;
     end;
       AeStDtm=dhms(aeondt,hr,mn,0);
%mend ;

%macro DER_004 ;                        /** Code the Derivation for : AeEndTm  **/ ;
       hr=hour(aeendtm);
       mn=minute(aeendtm);
     if aeendtm eq . then do;
       hr=0;
       mn=0;
     end;
       AeEndTm=dhms(aeenddt,hr,mn,0);
%mend ;
```

As you can see, the feature for automating the task of SAS® macro code generation is very powerful and convenient. Please see Fig. 7 for a description on all other action keys.

The final product of CDISCDOMAIN function call was a domain variable map according to the CDISC specifications(defined in domain template). By looking at Fig 12. we can see partial listing of the final AE domain

Fig 11. CDISC AE domain template. nple.

DOMAIN	PRJVAR	PRJVLBL	PRJKEY	PRJTYP	PRJFMT	PRJLEN	PRJVDEF	PDYVIEW	PDYVAR	COREVAR	PRJVCOM
AE	Studyid	Study Identifier		C	$15.	15		AEAEA	STUDY	Y	
AE	Domain	Domain Abbreviation		C	$2.	2	_DEFINE_	AEAEA	AE	Y	CDISC uses a 2 character domain name
AE	Usubjid	Unique Subject Identifier		C	$30.	30		AEAEA	UNQPTNO	Y	
AE	AeSeq	Sequence Number		N	8.	8	_DERIVE_	AEAEA		Y	Sequence number to ensure uniqueness in domain (Derive)
AE	AeTerm	Reported Term for Adverse Event		C	$50.	50		AEAEA	AEMNM	Y	The verbatim term of the event
AE	AeStDtm	Start Date/time of Event		N	12.	12	_CODE_	AEAEA	Hr=hour(aeontm); Mn=minute(aeontm); If aeontm=. Then do; Hr=o; mn=0; Aestdtm=dhmr(hr,mn,0); End;	Y	Start date/time for an adverse event in seconds from 01/01/1960 (DERIVE AS AESTDTM = AEONYMD| AEONTM)
AE	AeStDtmp	Precision of AESTDTM		N	8.	8	_DEFINE_	AEAEA	86400	Y	Precision of AeStDtm in seconds
AE	AeEndTm	End date/time of Event		N	12.	12	_CODE_	AEAEA	Hr=hour(aeendtm); Mn=minute(aeendtm); If aeendtm=. Then do; Hr=o; mn=0; Aeendtm=dhmr(hr,mn,0); End;	Y	End date/time of adverse event in seconds from 01/01/1960 (DERIVE AS AEENDTM =AEEENDYMD | AEENDTM))
AE	AeEndTmp	Precision of AEENDTM		N	8.	8	_DEFINE_	AEAEA	86400	Y	Precision of AeEndTm in seconds
AE	AeStDy	Start Day of Event		N	8.	8	_DERIVE_	AEAEA		Y	Day of start of adverse event relative to REFDTM (AESTDY = AEONDT-REFDTM)
AE	AeEnDy	End Day of Event		N	8.	8	_DERIVE_	AEAEA		Y	Day of end of adverse event relative to REFDTM (AEENDY = AEENDDT-REFDTM)
AE	Visitnum	Visit Number		N	8.2	8		AEAEA	ACTEVENT	N	Added to domain since AEs are not all collapsed on a period x period basis
AE	AeModify	Modified reported Term		C	$30.	30	_DELETE_			N	If AETERM is modified as part of procedure , then modified text goes here
AE	AeDecod	Dictionary-Derived Text Description		C	$50.	50	_DERIVE_	AEAEA	MPT	Y	Dictionary-derived text description of AETERM or AEMODIFY
AE	AeBodSys	Body System or Organ Class		C	$200.	200	_DERIVE_	AEAEA	MSOC	Y	Body system or organ class (primary SOC) for the adverse event (from MEDDRA) (DERIVE from %XMEDTRM macro)
AE	AeTrtEm	Treatment Emergent		C	$2.	2	_DERIVE_	AEAEA	AETRT	Y	Was the event emergent ? [Y, N] (DERIVE as 'Y' if AETRT not "Screening", "Off-drug period", "Post-study" else 'N' when not blank)
AE	AeSev	Severity/Intensity		C	$10.	10		AEAEA	AEINT	Y	The severity of the event [MILD, MODERATE, SEVERE]
AE	AeSer	Serious Criteria		C	$1.	1		AEAEA	AESERA	Y	Is this a serious event? [Y, N]
AE	AeAcn	Action Taken with Study Treatment		C	$50.	50		AEAEA	AEACTA	Y	Describes changes to study treatment as a result of the event
AE	AeAcnOth	Other Action Taken		C	$50.	50	_DELETE_			N	Describes other action taken as a result of the event
AE	AeRel	Causality		C	$40.	40	_CODE_	AEAEA	AEREL = put(aereln,aerelf.);	Y	Investigator's opinion to the causality of the event to treatment [DEFINITELY NOT RELATED, POSSIBLY RELATED, PROBABLY RELATED, etc]
AE	AeRelOth	Relationship to OTHER (NON-STUDY) TREATMENT		C	$20.	20	_DELETE_			N	Investigator's opinion to the causality of the event to non-study treatment [DEFINITELY NOT RELATED, POSSIBLY RELATED, PROBABLY RELATED, etc]
AE	AeOut	Outcome of Event		C	$20.	20	_CODE_	AEAEA	AEOUT= put(aeoutn,aeoutf.);	Y	Description of the outcome of the event [RECOVERED, RESOLVED, FATAL, etc] (E2b values)

Fig. 12. The AE CDISC domain variables map, as produced by the CDISCDOMAIN function

DOMAIN	PRJVAR	PRJVLBL	PRJVCOMP	PRJKEY	PRJTYP	PRJFMT	PRJLEN	SDYVIEW	SDYVAR	STUDY1	VARFMT1	VARLEN1	VARTYP1
AE	Studyid	Study Identifier	Ok		C	$15.	15	AEAEA	STUDY	test	$15.	9	C
AE	Domain	Domain Abbreviation	Ok		C	$2.	2	AEAEA	%DER_016	test			
AE	Usubjid	Unique Subject Identifier	Ok		C	$16.	16	AEAEA	UNQPTNO	test	$16.	16	C
AE	AeSeq	Sequence Number	Ok		N	8.	8	AEAEA	%DER_011	test			
AE	AeTerm	Reported Term for Adverse Event	Match		C	$50.	50	AEAEA	%DER_019	test	$50.	50	C
AE	AeStDtm	Start Date/Time of Event	Ok		N	12.	12	AEAEA	%DER_012	test			
AE	AeStDtmp	Precision of AESTDTM	Ok		N	8.	8	AEAEA	%DER_013	test			
AE	AeEndTm	End Date/Time of Event	Ok		N	12.	12	AEAEA	%DER_003	test			
AE	AeEndTmp	Precision of AEENDTM	Ok		N	8.	8	AEAEA	%DER_004	test			
AE	AeStDy	Start Day of Event	Ok		N	8.	8	AEAEA	%DER_014	test			
AE	AeEnDy	End Day of Event	Ok		N	8.	8	AEAEA	%DER_002	test			
AE	Visitnum	Visit Number	Added		N	8.2	8	AEAEA	ACTEVENT	test	8.	8	N
AE	AeDecod	Dictionary-Derived Text Description	Ok		C	$50.	50	AEAEA	%DER_023	test	$200.	200	C
AE	AeBodSys	Body System or Organ	Ok		C	$200.	200	AEAEA	%DER_001	test			
AE	AeTrtEm	Treatment Emergent	Ok		C	$2.	2	AEAEA	%DER_015	test			
AE	AeSev	Severity/Intensity	Ok		C	$10.	10	AEAEA	AEINT	test	AEINTF	8	N
AE	AeSer	Serious Criteria	Ok		C	$1.	1	AEAEA	AESERA	test	YNS1F	8	N
AE	AeAcn	Action Taken with Study Treatment	Ok		C	$50.	50	AEAEA	AEACTA	test	AEACT	8	N
AE	AeRel	Causality	Ok		C	$40.	40	AEAEA	%DER_018	test	YN1F.	8	N
AE	AeOut	Outcome of Event	Ok		C	$20.	20	AEAEA	%DER_017	test	AEOUT	8	N
AE	?AeDur	Duration of Event	Ok		N	8.2	8	AEAEA	AEDUR	test	11.	8	N
AE	?AeDurU	Units of Time for AEDUR	Ok		C	$10.	10	AEAEA	%DER_022	test			
AE	AeOngo	Ongoing Adverse Event?	Ok		C	$2.	2	AEAEA	%DER_005	test			
AE	AeSCong	Congenital Anomaly or Birth Defect	Ok		C	$2.	2	AEAEA	%DER_006	test			
AE	AeSDisab	Permanent/Serious/Disable/Incapacitating	Ok		C	$2.	2	AEAEA	%DER_007	test			
AE	AeSDth	Results in Death	Ok		C	$2.	2	AEAEA	%DER_008	test			
AE	AeSHosp	Requires or Prolongs Hospitalization	Ok		C	$2.	2	AEAEA	%DER_009	test			
AE	AeSLife	Is Life Threatening	Ok		C	$2.	2	AEAEA	%DER_010	test			
AE	AeSOth	Other Medically Important Serious Event	Match		C	$2.	2	AEAEA	%DER_021	test			
AE	AeSOthC	Dscr of Other Med Impt Serious Event	Added		C	$25.	25	AEAEA	%DER_020	test			
AE	AeConTrt	Concomitant or Additional Trtmnt Given	Ok		C	$1.	1	AEAEA	AETHPA	test	YN1F.	8	N
AE	AeCom	Comment	Match		C	$200.	200	AEAEA	AEX	test	$200.	200	C

As you can see from the structure of the above domain variable map, it looks just like a conventional domain variable map (as created by the NEWDOMAIN and MODDOMAIN functions). However in this case, all project level variables and their attributes correspond to CDISC established standards. Some minor editing of this map was required (The editing process for domain variable maps is described in detail in the referenced articles) .

Now we were ready to upload the actual data into the CDISC domain, the final step on the roadmap in Fig.2. That is accomplished with the standard XPDL LOADPDB function (see the referenced paper), which is no different from the conventional PDB loading process. So what we have done, in essence, is to use the CDISCDOMAIN function with a template (that defines what the CDISC domain should like) to create a domain variable map (that defines the data translation) and use that variable map to create or load our final domain dataset.

The following XPDL call was used to actually load the data into the final AE CDISC domain dataset:

```
%XPDL(function=LOADPDB,                xslib=xlsin,
      mapfile=CDISC_domain_map_v1,     pgmref=pgmin,
      funcopt=< pdbloc =local
                pdbpath=K:\MEDICAL\data\MEDDAT95\SAS\CLINREP\IND\CDISC_pdb\Test_Data
                domname=ae   sdy_var=studyid    saspgm=c:\windows\temp\ae.pgm
                run    =y    debug  =Y;
             >
      );
```

The major parameters in the above call are the same as described in section 2 (FUNCTIONALITY). Parameter **mapfile** refers to the same domain-to-view map we created with NEWPDB function and subsequently edited (as described in the referenced paper). So, after the final editing of our *CDISC_domain_map_1.xls* , the CDISC 'domain-to-view' sheet looked like below the sample in Fig. 13 (partial, only the AE domain associated records are shown).

Fig. 13 – Partial Sample 'domain-to-view' sheet

DOMAIN	ACTIVE	DESC	DVARMAP	VIEWNAME	SDYFOL1	SDYLO(	SDYLVL1	SDYPAT1
AE	YES		CDISC_db_AE_v1	ADQ	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB
AE	YES		CDISC_db_AE_v1	AEAEA	test	remote	nda20040524	$RXC_SAS_DATA/s1182_PDB

Additional spreadsheet editing may be required after initial runs of the PDBLOAD function in order to produce correct results (as described in the referenced paper). After completing all of the editing, the final PDBLOAD run will produce an actual CDISC_AE domain dataset, which is subjected to a rigorous QC process. The final dataset (not shown here) is then converted to a SAS transport file before submitting to FDA.

The main advantage of the described above process is that even if the CDISC standards change in the future, say a new version, there is no need to change the XPDL macro coding. All the changes could be quickly applied to CDISC domain templates directly, without direct involvement of programming resources. The savings of time and resources for our process were tremendous and greatly helped us in keeping our submission to FDA ahead of target..

4.0 CONCLUSION

A powerful generic CDSIC Domain translator / loader has been created by combining the power of a SAS® 'data driven' program with the power of EXCEL® spreadsheets. The templates and spreadsheets, maintained by non-programmers, provide the documentation and also drive the CDISC domain creation and loading process.

XPDL with the new CDISCDOMAIN function has been be a tremendous success. A recent multi-trial e-submission was accomplished with XPDL and a dedicated team in record time. The cost savings for programming and data management resources, lower maintainability costs, higher quality of database / translation documentation all combine to make this macro a 'must have' addition to the software toolkit in the pharmaceutical industry.

REFERENCES

1. PharmaSug 2003 paper : XPDL – An Extensible Project Database Loading and Table translation Program' by John Adams.
2. NeSug 2003 paper : XPDL – An Extensible Project Database Loading and Table translation Program' by John Adams.

CONTACT INFORMATION

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

John H. Adams
Boehringer Ingelheim Pharmaceutical, Inc.
900 Ridgebury Road
Ridgefield, CT, 06877-0368
Work Phone: 203-778-7820
Fax: 203-837-4413
Email: jadams@rdg.boehringer-ingelheim.com
adamsjh@mindspring.com