PharmaSUG2011 - Paper CD12

ADaM Standard Naming Conventions are Good to Have

Christine Teng, Merck Sharp & Dohme Corp, Rahway, NJ

ABSTRACT

The Clinical Data Interchange Standards Consortium (CDISC) Version 1.0 Analysis Data Model Implementation Guide (ADaMIG) was released in 2009. The ADaMIG specifies ADaM standard dataset structures and variables, including naming conventions. It also specifies standard solutions to implementation issues. This paper focuses on the naming conventions (*GRy, *FL, etc.) which enable the use of wildcards in programming code when displaying the data content for quality control (QC) purposes. This paper presents code examples using SAS data dictionary tables and Perl regular expressions to output the values of key variables of interest based on naming conventions. The code examples can also be applied to the standard Basic Data Structure (BDS) required variables such as PARAM and PARAMCD. An Excel workbook (using the SAS®9 ExcelXP tagset) that contains metadata for all ADaM datasets of a mock study will be produced. Each worksheet includes information for one ADaM dataset and the associated values of specified variables based on naming conventions. This workbook is very helpful in data validation during the development process. The code can be re-used for any study that follows the ADaM standards.

SAS®9, Windows®, Intermediate Level
Key Words: CDISC, ADaM, ExcelXP, Naming Convention, Regular Expressions

INTRODUCTION

Based on the Analysis Data Model (ADaM) document - CDISC Analysis Data Model Version 2.1, it specifies the fundamental principles and standards to follow in the creation of analysis datasets and associated metadata. The purpose of ADaM is to provide a framework that enables analysis of the data, while at the same time allowing reviewers and other recipients of the data to have a clear understanding of the data's lineage from collection to analysis to results. The ADaM document also describes ADaM metadata, the subject-level dataset ADSL, and a multiple-record-per-subject data structure: the ADaM Basic Data Structure (BDS). The CDISC Version 1.0 Analysis Data Model Implementation Guide (ADaMIG) specifies ADaM standard dataset structures and variables, including naming conventions. It also specifies standard solutions to implementation issues. The ADaMIG should be used in concert with the ADaM document. Founded on section 4 of the ADaM document,

Analysis datasets must:

- include a subject-level analysis dataset named “ADSL”

- consist of the optimum number of analysis datasets needed and have enough self-sufficiency to allow analysis and review with little or no additional programming or data processing

- be named using the convention “ADxxxxxx”

- use ADaM standard variable names and naming conventions when available

- maintain the values and attributes of SDTM variables if copied into analysis datasets without renaming (i.e., adhere to the “same name, same meaning, same values” principle of harmonization

- apply naming conventions for datasets and variables consistently across studies within a given submission and across multiple submissions for a product

This paper focuses on how the naming conventions can help simplify programming for the display of data content. A simple regular expression function PRXMATCH is used to demonstrate how easy it is to display data based on certain patterns of variable names. The report of metadata and value of some variables will be displayed in an Excel workbook which is created via the SAS ExcelXP tagset.

The SAS ExcelXP tagset generates XML output that conforms to the Microsoft XML Spreadsheet Specification ("XML Spreadsheet Reference", Microsoft Corp.). It provides the functionality to create multiple worksheets in a workbook as well as multiple tables within a single worksheet. These features are very useful for creating metadata documentation where each ADaM dataset has its own worksheet with label. It enables quicker accessibility to locate the information for each ADaM dataset. With SAS DICTIONARY and PROC SQL, the metadata documentation can be created without hardcoding. This paper is not a tutorial about the ExcelXP tagset. Rather, it demonstrates another application using the ExcelXP tagset. The detailed tutorials and references for the ExcelXP tagset can be found in the References section of this paper.

SAS provides many standard style templates that allow for customization. To see a list of templates provided by SAS, (1) go to the Results windows, (2) right click on Results and select Template, (3) expand sashelp.Tmplmst (See Table-1 in Appendix). Templates for Tagsets and Styles can be found here. Templates can be customized using parent templates provided by SAS. Style templates make the output more presentable. Examples of some SAS ODS styles can be found at http://stat.lsu.edu/SAS_ODS_styles/SAS_ODS_styles.htm.

Perl regular expressions (regexp) is a language for searching text that consists of a string of literal characters and special characters called metacharacters. When searching with a regexp, a literal character in the regexp matches that character in the string being searched. For example, the regexp /abc/ will search for the characters abc in a string. All regexp begin and end with a delimiter for example, " /". The power of regexp lies with the metacharacters. The regexp metacharacters perform special actions when searching. For instance, the regexp "/gr\d/" will match the text gr followed by a digit. The metacharacter for a digit 0-9 is \d. However, \D matches non-digit.

The Version 9 Online DOC or any book on PERL programming will provide you with more details about different kind of metacharacters or wild cards to locate pattern.

ADaM NAMING CONVENTIONS

Section 3 of ADaMIG defines the required characteristics of standard variables that are frequently needed in analysis datasets. The ADaM standard requires that these variable names be used when a variable that contains the content defined in Section 3 is included in an analysis dataset. ADaMIG section 3.1 describes variables in ADSL and section 3.2 describes variables in the BDS.

Values of ADaM “Core” Attribute in ADaMIG

Req = Required. The variable must be included in the dataset.

Cond = Conditionally required. The variable must be included in the dataset in certain circumstances.

Perm = Permissible. The variable may be included in the dataset, but is not required.

Unless otherwise specified, all ADaM variables are populated as appropriate, meaning nulls are allowed.

Below are some examples used in this paper to demonstrate standard variables and naming conventions in getting data content.

Required variables:
Example 1: ADSL SEX , RACE, TRTxxP (xx refer to specific period) and ARM
Example 2: BDS PARAM and PARAMCD

Conditional variables:
Example 3: *FL (Character subject-level population flag names end in FL. Similarly, parameter-level population flag names end in PFL, and record-level population flag names end in RFL)

Permissible variables:
Example 4: *GRy (Variables whose names end in GRy are grouping variables, where y refers to the grouping scheme or algorithm)

IMPLEMENTATION

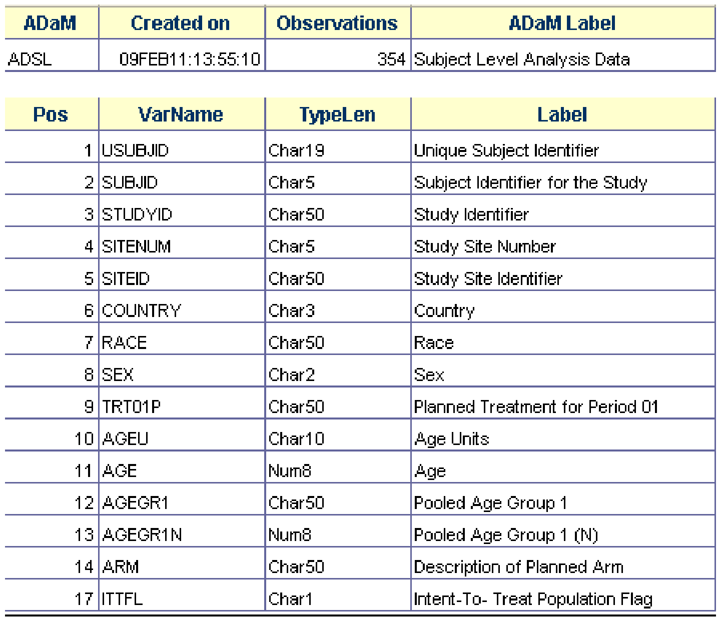
Since the ExcelXP tagset is still evolving, there are some limitations and hence its functionality may be changed in the future. It is recommended that the user always download the latest update to verify the changes and enhancements. To use the ExcelXP tagset, first download the latest ExcelXP tagset from the SAS ODS MARKUP page. This page also provides links to documentation for using and customizing tagsets. For this exercise, the ExcelXP tagset version dated 08/25/10 2010, version 1.116 was used. Details of ExcelXP syntax used will not be explained in this paper. Please refer to SAS knowledge base website for reference.

Below is the framework that creates a workbook, where each worksheet has information for metadata about the dataset and variables as well as specific data content for each ADaM dataset defined in &datadir libname. Use %Do-%While loop to process/parse each dataset. In the outer shell for ExcelXP setup,

```
ods tagsets.ExcelXP   path = "c:\temp\excelXP"   * output location
                      file = "test.xml"          * workbook name
                      style = XLStatistical;     * customized style template;

          *Build the worksheets (see below);

ods tagsets.ExcelXP  close;
```

```
proc sql noprint;
   *dsetname contains a list of ADaM datasets in a directory;
    select memname into :dsetname separated by '+'
    from dictionary.tables
    where libname="&datadir" and memtype="DATA" ;

    *examlst contains a list of ADaM datasets that have the PARAMCD variable;
    select memname into :examlst separated by ' '
    from   dictionary.columns
    where  libname="&datadir" and memtype="DATA" and name = 'PARAMCD' ;
quit;

*--------------------------------------------------------------------------------------------------------------*;
*Initialize the loop;
%let num=1;
%let list = %upcase(%scan(&dsetname, &num, +));

*Use Do-While loop to create individual worksheet;
%do %while (&list. ne  );
   *Create worksheet with defined options for ExcelXP;
    ods &_ODSDEST options(absolute_column_width='10, 15, 23, 28, 23, 10,10,10'  autofit_height='yes'
                          sheet_interval='none'  sheet_name="&list");

   *Print ADaM name and label at the beginning of the sheet;
    proc sql;
       select memname label='ADaM', crdate label='Created on', nobs label='Observations',
              memlabel label='ADaM Label'
        from   dictionary.tables
        where  libname="&datadir" and memtype="DATA" and memname="&list";
   quit;
```

```
     *Print variables for a  given ADaM dataset and attributes information;
      proc sql;
            select int(varnum) as Pos, upcase(name) as VarName,
                       propcase(catx('',type,put(length, best5.))) as TypeLen,
                       substr(label,1) as Label
             from   dictionary.columns
             where  libname = "&datadir" and memtype = "DATA" and memname = "&list"
            order by varnum;
      quit;
```



ADaM	Created on	Observations	ADaM Label
ADSL	09FEB11:13:55:10	354	Subject Level Analysis Data

Pos	VarName	TypeLen	Label
1	USUBJID	Char19	Unique Subject Identifier
2	SUBJID	Char5	Subject Identifier for the Study
3	STUDYID	Char50	Study Identifier
4	SITENUM	Char5	Study Site Number
5	SITEID	Char50	Study Site Identifier
6	COUNTRY	Char3	Country
7	RACE	Char50	Race
8	SEX	Char2	Sex
9	TRT01P	Char50	Planned Treatment for Period 01
10	AGEU	Char10	Age Units
11	AGE	Num8	Age
12	AGEGR1	Char50	Pooled Age Group 1
13	AGEGR1N	Num8	Pooled Age Group 1 (N)
14	ARM	Char50	Description of Planned Arm
17	ITTFL	Char1	Intent-To- Treat Population Flag

```
     *Keep data in memory to save processing time;
      data &list;
       set &datadir..&list;
       run;

      Example 1;
      Example 2;
      Example 3;
      Example 4;

     *Ready to build the next worksheet;
     %let num = %eval(&num + 1);
     %let list   = %upcase(%scan(&dsetname, &num, '+'));
%end;
```

Example 1: ADSL SEX , RACE and ARM
Listing required variables directly as their names are fixed and they are also part of ADSL.

```
%if &list = ADSL %then %do;
    proc sql;
           select SEX, RACE, count(*) label='Frequency'
           from   &list
```

```
            group by SEX, RACE;

            select ARM, count(*) label='Frequency'
            from   &list
            group by ARM;
      quit;
%end;
```

Sex	Race	Frequency
F	BLACK OR AFRICAN AMERICAN	15
F	WHITE	84

Description of Planned Arm	Frequency
ABC 1 mg	32
MK 20 mg	34
Placebo	33

Example 2: BDS PARAM and PARAMCD

Using pre-stored values in macro variable &examlst which contains all BDS dataset names to determine if paramcd and param should be listed in the current parsed dataset.

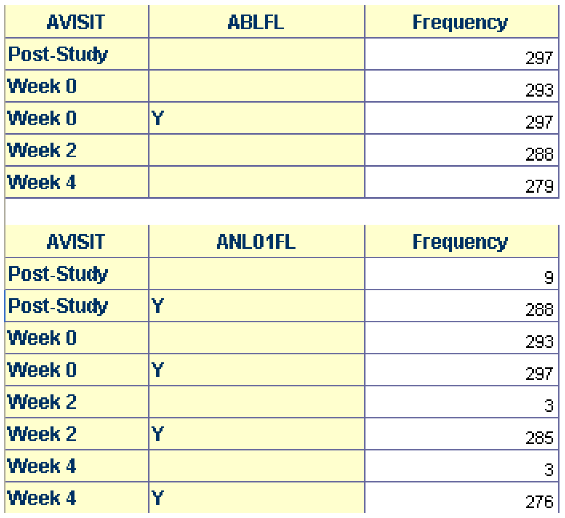
```
%if %index(&examlst., &list.) %then %do;
    proc sql;
         select distinct  paramcd, param
         from &list.;
    quit;
%end;
```

Parameter Code	Parameter Description
BMI	Body Mass Index (kg/m[2])
DIABP	Diastolic Blood Pressure (mmHg)
HEIGHT	Height (cm)
PULSE	Pulse Rate (beats/min)
RESP	Respiratory Rate (breaths/min)
SYSBP	Systolic Blood Pressure (mmHg)
TEMP	Temperature (C)
WEIGHT	Weight (kg)

Example 3: *FL

Using a Like clause in SQL to find all variable names ending with FL from dictionary table and list them by AVISIT.

```
proc sql noprint;
       select  name into :frqnames separated by ' avisit*'
        from dictionary.columns
        where (name like '%FL'  ) and libname="&datadir" and memname="&list";
quit;

%if &sqlobs ne 0 %then %do;
     proc freq data=&list;
          table avisit*&frqnames/list missing nopercent nocum;
     run;
%end;
```



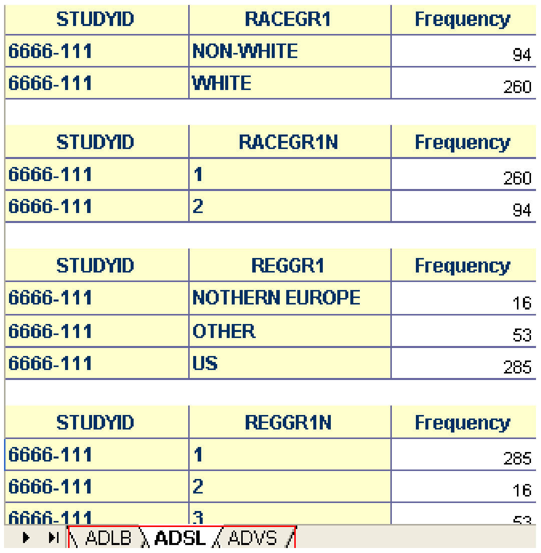
AVISIT	ABLFL	Frequency
Post-Study		297
Week 0		293
Week 0	Y	297
Week 2		288
Week 4		279

AVISIT	ANL01FL	Frequency
Post-Study		9
Post-Study	Y	288
Week 0		293
Week 0	Y	297
Week 2		3
Week 2	Y	285
Week 4		3
Week 4	Y	276

Example 4: *GRy

Using Perl regular expression to find variable names in a certain pattern. In the example below, variable names ending with both GRy and GRyN will be located and listed by studyid.

```
proc sql noprint;
       select  name into :frqnames separated by ' studyid*'
       from dictionary.columns
       where prxmatch("/[gG][rR][1-9]/",name) and libname="&datadir" and memname="&list";
quit;

%if &sqlobs ne 0 %then %do;
     proc freq data=&list;
          table studyid*&frqnames/list missing nopercent nocum;
     run;
%end;
```



STUDYID	RACEGR1	Frequency
6666-111	NON-WHITE	94
6666-111	WHITE	260

STUDYID	RACEGR1N	Frequency
6666-111	1	260
6666-111	2	94

STUDYID	REGGR1	Frequency
6666-111	NOTHERN EUROPE	16
6666-111	OTHER	53
6666-111	US	285

STUDYID	REGGR1N	Frequency
6666-111	1	285
6666-111	2	16
6666-111	3	53

As shown above, with the use of PROC SQL, SAS DICTIONARY tables, standardized ADaM structure/naming, and Perl regular expression, a workbook, that contains the metadata information for all ADaM datasets located in a directory, can be created with minimal code. This workbook is very helpful in data validation during the development process. The code can be re-used for any study that follows the ADaM standards. Customizations can also be done for study specific variables.

SUMMARY

Standardized analysis dataset structures allow the development of standard software tools that will facilitate the access, manipulation, and viewing of the analysis datasets. CDISC standardized naming conventions promote understanding and facilitate programming and analysis. By naming variables in a certain pattern, we promote more rapid comprehension of the meaning of variables in new studies. Data validation can be supported based on such predicted patterns and can be re-used from one study to another. Standards are good to have as they ultimately shorten development timelines, speed-up learning curves and save costs in resources.

REFERENCES

CDISC Analysis Data Model Version 2.1 http://www.cdisc.org/adam

ADaM Implementation Guide, Version 1.0 (ADaMIG v1.0) http://www.cdisc.org/adam

SAS Knowledge Base http://support.sas.com/rnd/base/ods/odsmarkup/

A Data Step in SAS 9: What's New by Jason Secosky, SAS, Cary, NC.
http://support.sas.com/rnd/base/datastep/dsv9-sugi-v3.pdf

SAS Macro Language: Reference

SAS SQL Procedure User’s Guide

SAS Language Reference Dictionary

ACKNOWLEGEMENTS

The author would like to thank the management team for their encouragement and review of this paper.

TRADEMARKS

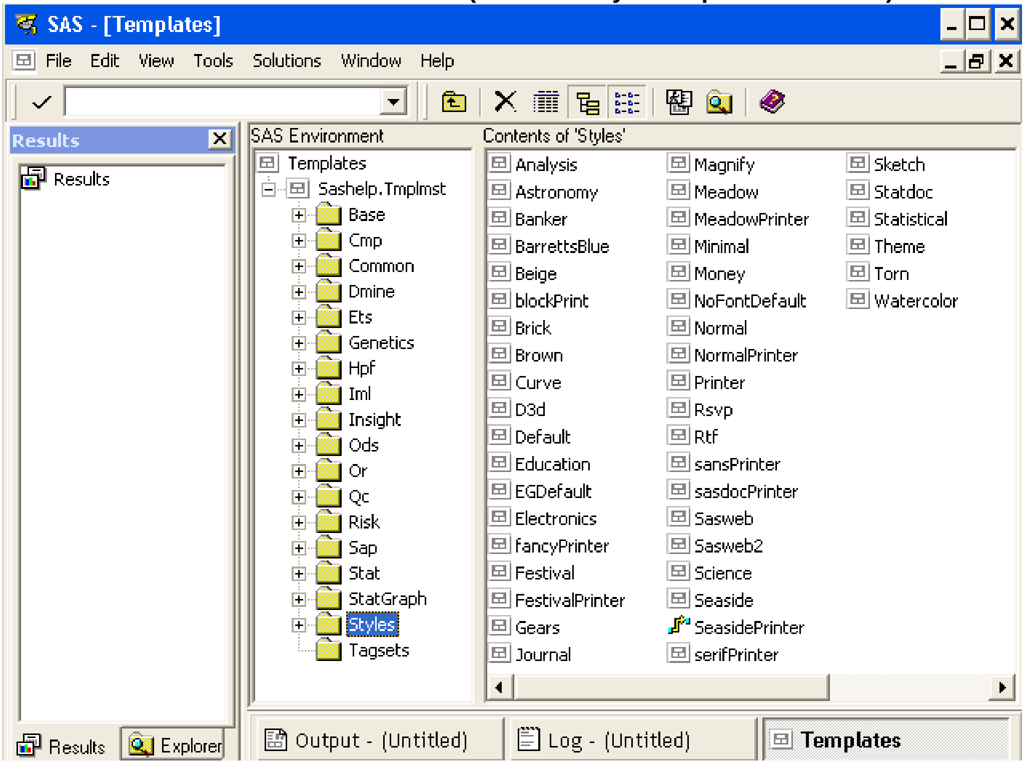
CONTACT INFORMATION

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

Christine Teng Merck Sharp & Dohme Corp Rahway, NJ 07065 christine_teng@merck.com	sas® Certified Advanced Programmer

APPENDIX

Table – 1 (Available Style Templates in SAS®9)



Templates
- Sashelp.Tmplmst
  - Base
  - Cmp
  - Common
  - Dmine
  - Ets
  - Genetics
  - Hpf
  - Iml
  - Insight
  - Ods
  - Or
  - Qc
  - Risk
  - Sap
  - Stat
  - StatGraph
  - Styles
  - Tagsets

Analysis	Magnify	Sketch
Astronomy	Meadow	Statdoc
Banker	MeadowPrinter	Statistical
BarrettsBlue	Minimal	Theme
Beige	Money	Torn
blockPrint	NoFontDefault	Watercolor
Brick	Normal	
Brown	NormalPrinter	
Curve	Printer	
D3d	Rsvp	
Default	Rtf	
Education	sansPrinter	
EGDefault	sasdocPrinter	
Electronics	Sasweb	
fancyPrinter	Sasweb2	
Festival	Science	
FestivalPrinter	Seaside	
Gears	SeasidePrinter	
Journal	serifPrinter	