Code Generator to Verify Implementation of Cut-off Date in the SDTM Domains

Christine Teng
Merck & Co., Inc.
Outline

- Background
- Approach
- PROC SQL
- SAS Code
The Clinical Data Interchange Standards Consortium (CDISC) has developed and published recommendations for data standards for Biopharmaceutical Companies.

Standard naming conventions are provided in guidance document (ex: xxTEST, xxTESTCD, xxFL, xxFN, xxDTC etc)
Background

- ISO 8601 is a widely used data standard for dates, times, durations, and intervals. The values are stored as text strings. They are formatted in a way that ensures that all of the components are always unambiguous.
Background

➢ Purpose of Cut-off date
  ▪ Common practice for interim-analysis in clinical trials.
  ▪ Data management cleans data at least up to the cut-off date before database lock.
  ▪ Correct implementation of data cut-off greatly impacts the analysis results and interpretations.

➢ QC of cut-off date
  ▪ To create a report to verify the proper cut-off date for the projects you manage in-house or from CRO.
Approach

Verify the cut-off dates were applied as expected

- Identify all date variables from SDTM domains based on naming pattern with suffix ‘DTC’
- Use PROC SQL and Data Dictionary tables to generate the scripts.
- Display the domains, date variable names and associated maximum date value within the study domains
Approach

Identify all date variables from SDTM domains based on naming pattern with suffix ‘DTC’

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Column Name</th>
<th>Column Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE</td>
<td>AEENDTC</td>
<td>End Date/Time of Adverse Event</td>
</tr>
<tr>
<td>AE</td>
<td>AESTDTC</td>
<td>Start Date/Time of Adverse Event</td>
</tr>
<tr>
<td>BS</td>
<td>BSDTC</td>
<td>Date/Time of Collection</td>
</tr>
<tr>
<td>CM</td>
<td>CMENDTC</td>
<td>End Date/Time of Medication</td>
</tr>
<tr>
<td>CM</td>
<td>CMSTDTC</td>
<td>Start Date/Time of Medication</td>
</tr>
<tr>
<td>CO</td>
<td>CODTC</td>
<td>Date/Time of Comment</td>
</tr>
</tbody>
</table>
**Approach**

Find maximum date value for each DTC variable within the domain

<table>
<thead>
<tr>
<th>Domain</th>
<th>Date Variable</th>
<th>Max Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE</td>
<td>AEEN DTC</td>
<td>2018-03-24</td>
</tr>
<tr>
<td>AE</td>
<td>AEST DTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>BS</td>
<td>BSDTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CM</td>
<td>CMEN DTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CM</td>
<td>CMST DTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CO</td>
<td>CODTC</td>
<td>2018-03-26T00:00:00</td>
</tr>
</tbody>
</table>
There are two kinds of Macro Variables in PROC SQL we can use.

- User Defined Macro Variables using “INTO: ” Clause.
- Automatic Macro Variables provided by SAS.
The Macro Variables that you create and name are called user-defined Macro Variables.

```
SELECT <column name in a table>
INTO :<Macro Variable name1> -
    :<Macro Variable name99*> 
FROM < table>

Store row values into a list of Macro Variables

*Only the needed number of macro variables will be created. You want to specify a number that is big enough to hold the number of observations returned from the SELECT statement.
Those variables that are defined by SAS are called **automatic Macro Variables**. PROC SQL produces several automatic Macro Variables (ex: SQLOBS, SQLRC, and SQLOOPS).

In this exercise, SQLOBS is used.
## PROC SQL

### Macro Variables

<table>
<thead>
<tr>
<th>Macro Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLOBS</td>
<td>Contains the number of rows or observations executed by a SQL statement.</td>
</tr>
<tr>
<td>SQLRC</td>
<td>Contains the return code from a SQL statement. For return codes, see SAS SQL documentation.</td>
</tr>
<tr>
<td>SQLOOPS</td>
<td>Contains the number of iterations the inner loop of PROC SQL processes.</td>
</tr>
</tbody>
</table>
SAS Code – Identify Date Variables

- From the data dictionary table only look for **date variables** (which should end with **DTC** per standard naming convention)

- Dataset “**datetest**” is created with SDTM domain name, date variable name and associated variable label

```sas
create table datetest as
select memname, name, label as domain, variable, variable label
from dictionary.columns
where (name like '%DTC') and libname='DATADIR'
order by memname, name;
```
SAS Code – Construct Scripts

```
select "select " || quote(strip(memname)))|| ',' ||
     quote(strip(name)) || ", MAX(" ||(strip(name) || ")
from datadir:" || strip(memname) ||
" where " || strip(name) || " ne " ;"
into :select1- :select999
from datetest ;
```

- Construct script into each macro variable name: &select1 to &selectn, example of &select1 is as follows:

```
select “AE”, “AEENDTC”, MAX(AEENDTC)
from datadir.AE where AEENDTC ne " ";
```
SAS Code – Execute Scripts

%let cnt=&sqlobs;
create table T1 (ds_name char(5), date_var char(10), maxdtc char(20) );
%do i=1 %to &cnt;
   insert into T1
      &&select&i;
%end;

• An example of macro variable &select1 has:

  Insert into T1
  select “AE”, “AEENDTC”, max(AEENDTC)
  from datadir.AE where AEENDTC ne ‘‘;
SAS Code – Display T1 Table

The cut-off date in this example is 3/26/2018
The below example shows that date variables in DM domain did not have the rule applied.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Date Variable</th>
<th>Max Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE</td>
<td>AEENDTTC</td>
<td>2018-03-24</td>
</tr>
<tr>
<td>AE</td>
<td>AESTDTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>BS</td>
<td>BSDTTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CM</td>
<td>CMENDTTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CM</td>
<td>CMSTDTTC</td>
<td>2018-03-26</td>
</tr>
<tr>
<td>CO</td>
<td>CODTTC</td>
<td>2018-03-26T00:00:00</td>
</tr>
<tr>
<td>DD</td>
<td>DDDTTC</td>
<td>2018-03-24</td>
</tr>
<tr>
<td>DM</td>
<td>BRTHDTTC</td>
<td>1997-02</td>
</tr>
<tr>
<td>DM</td>
<td>DISCNDTTC</td>
<td>2018-03-27</td>
</tr>
<tr>
<td>DM</td>
<td>DTHDTTC</td>
<td>2018-04-12</td>
</tr>
<tr>
<td>DM</td>
<td>RFENDTTC</td>
<td>2018-03-27</td>
</tr>
<tr>
<td>DM</td>
<td>RFICDTTC</td>
<td></td>
</tr>
</tbody>
</table>
Summary

- CDISC variables naming standard enables pattern search for checking common logic

- Macro programming, macro variables, data dictionary tables and PROC SQL can simplify and automate processes easily
Contact

Please forward questions and comments to

Christine Teng
Merck & Co., Inc.
126 E. Lincoln Avenue
Rahway, NJ 07065
Christine_teng@merck.com