Do It Yourself:
Create your own SDTM mapping framework

Bas van Bakel, OCS Consulting, Netherlands
Scope

Source data -> Mapping Specifications

Mapping Specifications -> Study programs

Study programs -> Output data

Output data -> Company standards

Company standards -> Mapping Specifications
Mapping Specification document

- Microsoft Excel spreadsheet that contains all source-to-target specifications and, directly next to them, the translation of these specifications into code or pseudo-code.
Scope

Source data → Standard Macros → Mapping Specifications → SDTM

SDTM Standards (CDISC/company)
Framework Macros

- Independent of the SDTM domain that is to be generated there will always be three macros that are executed in sequence.
  - `%mapping`: Each of the source datasets are converted into intermediate ‘mapped’ dataset(s) by using some of the (pseudo-) code as specified in the mapping specification document.
  - `%apply_poststeps`: Combines/processes ‘mapped’ dataset(s) into one single ‘near final’ intermediate SDTM dataset by using the remainder of the (pseudo-) code.
  - `%makefinaldomain`: Generates SDTM dataset and Supplemental Qualifier dataset from the ‘near final’ dataset by aligning the datasets and variables with the attributes as specified in the ‘target metadata’.
Framework Macros

Source1 → Mapped_Source1 → SDTM.AE
Source2 → Mapped_Source1 → SDTM.DM
Source3 → Mapped_Source1 → SDTM.DS
Source4 → Mapped_Source1 → SDTM.[xx]
Source5 → Mapped_Source5 → SDTM.[xx]

(Target dataset = ‘AE’)

%mapping %apply_poststeps

%makefinaldomain
Framework Macros

Source1

Source2

Source3

Source4

Source5

Mapped_Source2

Mapped_Source3

Mapped_Source5

Mapped_DM

SDTM.AE

SDTM.DM

SDTM.DS

SDTM.[xx]

%mapping

%apply_poststeps

%makefinaldomain
Framework Macros

Source1
Source2
Source3
Source4
Source5

Mapped_Source5

%mapping
%apply_poststeps

%makefinaldomain

Source[N]

SDTM.AE
SDTM.DM
SDTM.DS
SDTM.[xx]
Framework Macros

- Depending on the function that is specified in the FUNCTION column of the mapping specification the SAS code is executed either when the source data is converted to the ‘mapped’ datasets (%mapping) or when post processing and combining of the ‘mapped’ datasets take place (%apply_poststeps)


> ‘POSTSTEP1-POSTSTEP[n]’, ‘FUNCTION1-FUNCTION[n]’
%Mapping Macro

- Each source dataset for which any row is available that is marked as being needed to generate the SDTM domain will be read by the %mapping macro. Only the source variables that are marked as being needed in this process are read from the source datasets.

This process ensures that you cannot use variables that you have not marked in the specifications as being needed. "NOT MAPPED" is used to clearly indicate a variable is not used in the mapping process.

When reading the CDASH.CDASH_AE dataset, only the variables AEYN and SUBJID will be kept. When reading the MEDDRA.MEDDRA dataset, only the variables LLT_CODE, PT_CODE, SOC_CODE, HLGTCODE, HLT_CODE and PRIMARY will be kept.

www.ocs-consulting.com/nl
**%Mapping Macro**

- When the function "**WHERE [statement]**" is specified the exact where clause as specified within the brackets will be applied on the source dataset.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>VARIABLE</th>
<th>SDTM SDTM_VAR</th>
<th>SPECIFICATION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDASH.CDASH AE</td>
<td>VISITNUM</td>
<td></td>
<td>Not mapped</td>
<td>NOT MAPPED</td>
</tr>
<tr>
<td>CDASH.CDASH AE</td>
<td>SUBJID</td>
<td>DM</td>
<td>Rename for processing (SDTM output variable has the same name)</td>
<td>RENAME [subjid]</td>
</tr>
<tr>
<td>CDASH.CDASH AE</td>
<td>SUBJID</td>
<td>DM</td>
<td>Store the (numeric) source SUBJID as character value in a six digit format</td>
<td>FUNCTION [subjid = STRIP(PUT(_subjid, Z6));]</td>
</tr>
<tr>
<td>CDASH.CDASH AE</td>
<td>SUBJID</td>
<td>AE</td>
<td>Set equal to the study number defined in the input program (&amp;studynum).</td>
<td>FUNCTION [studynum = STRIP(&quot;&amp;studynum&quot;);]</td>
</tr>
<tr>
<td>CDASH.CDASH AE</td>
<td>SUBJID</td>
<td>AE</td>
<td>Concatenate with a ‘</td>
<td>’ the calculated value of studynum and the (numeric) source SUBJID as character value in a six digit format</td>
</tr>
<tr>
<td>CDASH.CDASH AE</td>
<td>AEYN</td>
<td></td>
<td>Keep only records that indicate an AE occurred by keeping only records with AEYN = 'Y'</td>
<td>WHERE [aeyn = 'Y']</td>
</tr>
</tbody>
</table>

Resolves to: **WHERE=((aeyn = 'Y'))**

- If there are multiple WHERE-clauses specified they will both be applied on the source dataset so only records fulfilling both clauses will be kept when reading the source dataset.
%Mapping Macro

When the function “RENAME [newvar]” is specified the source variable will be renamed to the value within the brackets and the rename statement will be applied on the source dataset.

Resolves to: RENAME=(subjid=_subjid)
%Mapping Macro

- When the function "COPY" is specified the source variable will be copied to the SDTM target variable by generating a SAS statement in the format [target variable] = [source variable];.
- If there are multiple COPY-functions specified for a source dataset they will be concatenated.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>VARIABLE</th>
<th>SDTM</th>
<th>SDTM_VAR</th>
<th>SPECIFICATION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDAH.CDAH_DM</td>
<td>SUBID</td>
<td>DM</td>
<td></td>
<td>Rename for processing (SDTM output variable has the same name)</td>
<td>RENAME [subject]</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>STUDYID</td>
<td>DM</td>
<td></td>
<td>Set equal to the study number defined in the input program [&amp;studynum].</td>
<td>FUNCTION [studyid = STRIP(&quot;&amp;studynum&quot;)];</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>SUBID</td>
<td>DM</td>
<td></td>
<td>Store the (numeric) source SUBID as character value in a six digit format</td>
<td>FUNCTION [subject = STRIP(PUT(_subject,26.))]</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>SUBJID</td>
<td>DM</td>
<td></td>
<td>Concatenate the calculated value of STUDYID and SUBJID with a &quot;.&quot;.</td>
<td>FUNCTION [usubjind = STRIP(studyid)</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>SITEID</td>
<td>DM</td>
<td></td>
<td>Copy directly from source variable</td>
<td>COPY</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>BIRTH_DAT</td>
<td>DM</td>
<td></td>
<td>Generate ISO8601 format from CDAH --DAT variable in character format</td>
<td>FUNCTION [%isof_from_dat(outvar=birthdate, invar=birthdat)];</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>RACE</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the RACE recoding list</td>
<td>RECODE [RACE]</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>RACE</td>
<td>DM</td>
<td></td>
<td>Copy of RACE.</td>
<td>COPY</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>RACEOTH</td>
<td>DM</td>
<td></td>
<td>Copy of RACEOTH.</td>
<td>COPY</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>ETHNIC</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the ETHNIC recoding list</td>
<td>RECODE [ETHNIC]</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>SEX</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the SEX recoding list</td>
<td>RECODE [SEX]</td>
</tr>
<tr>
<td>CDAH.CDAH_DM</td>
<td>DOMAIN</td>
<td>DM</td>
<td></td>
<td>Set to 'DM' for all records.</td>
<td>FUNCTION [domain = 'DM']</td>
</tr>
</tbody>
</table>

Resolves to: SITEID = SITEID; RACEOR = RACE; RACEOTH=RACEOTH;
%Mapping Macro

- When the function “FUNCTION [statement]” is specified the exact code as specified in brackets will be executed.
- If there are multiple FUNCTION functions specified for a source dataset they will be concatenated in the order of which they are available in the mapping specification document.

Resolves to:

studyid = STRIP("&studynum"); subjid = STRIP(PUT(_subjid,Z6.));
usubjid = STRIP(studyid) || "-" || STRIP(subjid);
%iso_from_dat(outvar=brthdtc, invar=brthdat); domain = ‘DM’;
**%Mapping Macro**

- When the function “**RECODE**” is specified the value specified in brackets will be used to check a specific part of the mapping specification document for a recoding list.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>VARIABLE</th>
<th>SDTM</th>
<th>SDTM VALUE</th>
<th>SPECIFICATION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SUBJID</td>
<td>DM</td>
<td></td>
<td>Rename for processing (SDTM output variable has the same name)</td>
<td>RENAME [ _subj ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SUBJID</td>
<td>DM</td>
<td></td>
<td>Set equal to the study number defined in the input program (&amp;stidym).</td>
<td>FUNCTION [ studyid = STRIP(&quot;&amp;stidym&quot;); ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SUBJID</td>
<td>DM</td>
<td></td>
<td>Store the (numeric) source SUBJID as character value in a six digit format</td>
<td>FUNCTION [ subjid = STRIP(PUT(_subjid,26)); ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SUBJID</td>
<td>DM</td>
<td></td>
<td>Concatenate the calculated value of STUDYID and SUBJID with a &quot;.&quot;.</td>
<td>FUNCTION [ usurjid = STRIP(studyid)</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SITEID</td>
<td>DM</td>
<td></td>
<td>Copy directly from source variable</td>
<td>COPY</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>BRTHDATE</td>
<td>DM</td>
<td></td>
<td>Generate ISO8601 format from CDASH --DAT variable in character format</td>
<td>FUNCTION [ %iso_from_dat(outvar=brthdatc, invar=brthdat); ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>RACE</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the RACE recoding list</td>
<td>RECODE [ RACE ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>RACE</td>
<td>DM</td>
<td></td>
<td>Copy of RACE.</td>
<td>COPY</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>ETHNIC</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the ETHNIC recoding list</td>
<td>RECODE [ ETHNIC ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>SEX</td>
<td>DM</td>
<td></td>
<td>Recode source value to target value using the SEX recoding list</td>
<td>RECODE [ SEX ]</td>
</tr>
<tr>
<td>CDASH.CDASH_DM</td>
<td>DOMAIN</td>
<td>DM</td>
<td></td>
<td>Set to 'DM' for all records.</td>
<td>FUNCTION [ domain = 'DM'; ]</td>
</tr>
</tbody>
</table>

**CODELIST**

<table>
<thead>
<tr>
<th>TYPE</th>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHNIC</td>
<td>C2C</td>
<td>HISPANIC OR LATINO</td>
</tr>
<tr>
<td>ETHNIC</td>
<td>C2C</td>
<td>NOT HISPANIC OR LATINO</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>AMERICAN INDIAN OR ALASKA NATIVE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN - CENTRAL / SOUTH ASIAN HERITAGE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN - EAST ASIAN HERITAGE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN - SOUTH EAST ASIAN HERITAGE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>BLACK OR AFRICAN AMERICAN</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>OTHER</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>WHITE - ARABIC / NORTH AFRICAN HERITAGE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>WHITE - CAUCASIAN / EUROPEN HERITAGE</td>
</tr>
<tr>
<td>SEX</td>
<td>C2C</td>
<td>F</td>
</tr>
<tr>
<td>SEX</td>
<td>C2C</td>
<td>M</td>
</tr>
</tbody>
</table>
%Mapping Macro

- Using the recode list IF-THEN-ELSE code will be generated to translate source values to SDTM values:

```plaintext
IF ETHNIC = " THEN ETHNIC = ";
ELSE IF ETHNIC = 'HISPANIC OR LATINO' THEN ETHNIC = 'HISPANIC OR LATINO';
ELSE IF ETHNIC = 'NOT HISPANIC OR LATINO' THEN ETHNIC = 'NOT HISPANIC OR LATINO';

IF RACE = " THEN RACE = ";
ELSE IF RACE = 'AMERICAN INDIAN OR ALASKA NATIVE' THEN RACE = 'AMERICAN INDIAN OR ALASKA NATIVE';
ELSE IF RACE = 'ASIAN – CENTRAL / SOUTH ASIAN HERITAGE' THEN RACE = 'ASIAN';
ELSE IF RACE = 'ASIAN – EAST ASIAN HERITAGE' THEN RACE = 'ASIAN';
ELSE IF RACE = 'ASIAN – SOUTH EAST ASIAN HERITAGE' THEN RACE = 'ASIAN';
ELSE IF RACE = 'BLACK OR AFRICAN AMERICAN' THEN RACE = 'BLACK OR AFRICAN AMERICAN';
ELSE IF RACE = 'OTHER' THEN RACE = 'OTHER';
ELSE IF RACE = 'WHITE – ARABIC / NORTH AFRICAN HERITAGE' THEN RACE = 'WHITE';
ELSE IF RACE = 'WHITE – CAUCASIAN / EUROPEAN HERITAGE' THEN RACE = 'WHITE';

IF SEX = " THEN SEX = ";
ELSE IF SEX = 'F' THEN SEX = 'F';
ELSE IF SEX = 'M' THEN SEX = 'M';
```

<table>
<thead>
<tr>
<th>CODELIST</th>
<th>TYPE</th>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHNIC</td>
<td>C2C</td>
<td>HISPANIC OR LATINO</td>
<td>HISPANIC OR LATINO</td>
</tr>
<tr>
<td>ETHNIC</td>
<td>C2C</td>
<td>NOT HISPANIC OR LATINO</td>
<td>NOT HISPANIC OR LATINO</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>AMERICAN INDIAN OR ALASKA NATIVE</td>
<td>AMERICAN INDIAN OR ALASKA NATIVE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN – CENTRAL / SOUTH ASIAN HERITAGE</td>
<td>ASIAN</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN – EAST ASIAN HERITAGE</td>
<td>ASIAN</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>ASIAN – SOUTH EAST ASIAN HERITAGE</td>
<td>ASIAN</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>BLACK OR AFRICAN AMERICAN</td>
<td>BLACK OR AFRICAN AMERICAN</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>OTHER</td>
<td>OTHER</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>WHITE – ARABIC / NORTH AFRICAN HERITAGE</td>
<td>WHITE</td>
</tr>
<tr>
<td>RACE</td>
<td>C2C</td>
<td>WHITE – CAUCASIAN / EUROPEAN HERITAGE</td>
<td>WHITE</td>
</tr>
<tr>
<td>SEX</td>
<td>C2C</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>SEX</td>
<td>C2C</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>
%Mapping Macro

- The recoding tab is also used to check that all source values are expected values by checking them against the recoding list:

IF RACE NOT IN ("", 'AMERICAN INDIAN OR ALASKA NATIVE', 'ASIAN – CENTRAL / SOUTH ASIAN HERITAGE', 'ASIAN – EAST ASIAN HERITAGE', 'ASIAN – SOUTH EAST ASIAN HERITAGE', 'BLACK OR AFRICAN AMERICAN', 'BLACK OR AFRICAN AMERICAN', 'OTHER', 'WHITE – ARABIC / NORTH AFRICAN HERITAGE', 'WHITE – CAUCASIAN / EUROPEAN HERITAGE') THEN PUT "WAR" "NING: Variable RACE has value " [...] "not in recoding list";

IF ETHNIC NOT IN ("", 'HISPANIC OR LATINO', 'NOT HISPANIC OR LATINO') THEN PUT "WAR" "NING: Variable ETHNIC has value " [...] "not in recoding list";

IF SEX NOT IN ("", 'F', 'M') THEN PUT "WAR" "NING: Variable SEX has value " [...] "not in recoding list";

---

<table>
<thead>
<tr>
<th>CODELIST</th>
<th>TYPE</th>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETHNIC C2C</td>
<td>HISPANIC OR LATINO</td>
<td>HISPANIC OR LATINO</td>
<td></td>
</tr>
<tr>
<td>ETHNIC C2C</td>
<td>NOT HISPANIC OR LATINO</td>
<td>NOT HISPANIC OR LATINO</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>AMERICAN INDIAN OR ALASKA NATIVE</td>
<td>AMERICAN INDIAN OR ALASKA NATIVE</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>ASIAN – CENTRAL / SOUTH ASIAN HERITAGE</td>
<td>ASIAN</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>ASIAN – EAST ASIAN HERITAGE</td>
<td>ASIAN</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>ASIAN – SOUTH EAST ASIAN HERITAGE</td>
<td>ASIAN</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>BLACK OR AFRICAN AMERICAN</td>
<td>BLACK OR AFRICAN AMERICAN</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>OTHER</td>
<td>OTHER</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>WHITE – ARABIC / NORTH AFRICAN HERITAGE</td>
<td>WHITE</td>
<td></td>
</tr>
<tr>
<td>RACE C2C</td>
<td>WHITE – CAUCASIAN / EUROPEAN HERITAGE</td>
<td>WHITE</td>
<td></td>
</tr>
<tr>
<td>SEX C2C</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>SEX C2C</td>
<td>M</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>
%Mapping Macro

- When the function “STACK1”, “STACK2” or “STACK[n]” is specified the exact function as specified in brackets will be executed.
- If there are multiple STACK[x] functions specified for a source dataset they will be concatenated in the order of which they are available in the mapping specification document.
- In principle STACK works in the same way as FUNCTION except that after each set of statements with the same stack number (STACK[x]) an **OUTPUT statement** will be executed.
- In practice the stack functions will therefore be used to **generate multiple output records from a single source record**.
%Mapping Macro

<table>
<thead>
<tr>
<th>DATASET</th>
<th>VARIABLE</th>
<th>SDTM</th>
<th>SDTM_VAR</th>
<th>SPECIFICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMOG</td>
<td>HT</td>
<td>VS</td>
<td>VTESTCD</td>
<td>For each record in DEMOG generate one 'height' record: VTESTCD = 'HEIGHT';</td>
</tr>
<tr>
<td>DEMOG</td>
<td>HT</td>
<td>VS</td>
<td>VTEST</td>
<td>For each record in DEMOG generate one 'height' record: VTEST = 'Height';</td>
</tr>
<tr>
<td>DEMOG</td>
<td>HT</td>
<td>VS</td>
<td>VSORRES</td>
<td>For each record in DEMOG generate one 'height' record: VSORRES = HT stored as character.</td>
</tr>
<tr>
<td>DEMOG</td>
<td>HT</td>
<td>VS</td>
<td>VSSTAT</td>
<td>For each record in DEMOG generate one 'height' record: VSSTAT = 'NOT DONE' when HT is missing. Otherwise VSSTAT = missing.</td>
</tr>
<tr>
<td>DEMOG</td>
<td>VISNUM</td>
<td>VS</td>
<td>VISITNUM</td>
<td>Copy from VISNUM</td>
</tr>
<tr>
<td>DEMOG</td>
<td>WT</td>
<td>VS</td>
<td>VTESTCD</td>
<td>For each record in DEMOG generate one 'weight' record: VTESTCD = 'WEIGHT';</td>
</tr>
<tr>
<td>DEMOG</td>
<td>WT</td>
<td>VS</td>
<td>VTEST</td>
<td>For each record in DEMOG generate one 'weight' record: VTEST = 'Weight';</td>
</tr>
<tr>
<td>DEMOG</td>
<td>WT</td>
<td>VS</td>
<td>VSORRES</td>
<td>For each record in DEMOG generate one 'weight' record: VSORRES = WT stored as character.</td>
</tr>
<tr>
<td>DEMOG</td>
<td>WT</td>
<td>VS</td>
<td>VSSTAT</td>
<td>For each record in DEMOG generate one 'weight' record: VSSTAT = 'NOT DONE' when WT is missing. Otherwise VSSTAT = missing.</td>
</tr>
</tbody>
</table>

Resolves to:

vstestcd = 'HEIGHT';
vstest = 'Height';
if missing(ht) then vsorres = ";
else vsorres = STRIP(PUT(ht,best.));
if missing(ht) then vsstat = 'NOT DONE';
else vsstat = ";
OUTPUT;

vstestcd = 'WEIGHT';
vstest = 'Weight';
if missing(wt) then vsorres = ";
else vsorres = STRIP(PUT(wt,best.));
if missing(wt) then vsstat = 'NOT DONE';
else vsstat = ";
OUTPUT;
%Mapping Macro

- If you specify the function 'KEEP' the variable from the source dataset will be retained in the 'mapped' version of the dataset and can be referenced in the code that is used by the %apply_poststeps macro.
%Mapping Macro

- A simplified version of the single datastep that is generated by the %mapping macro:

```
DATA mapped_currlib. &currd (KEEP = &manualkeepstr &tovarkeepstr);
  SET work.&currds (KEEP = &keepstr);
  %IF %BQUOTE(&wherestr) NE %THEN %DO;
  %IF %BQUOTE(&renstr) NE %THEN %DO;
    WHERE=((&wherestr));
    RENAME=(&renstr);
  %END;
  %END;
  /*Copy*/
  &copystr;
  /*Apply recordings (also generate warn when value not available*)/
  &check;
  &recostr;
  /*Apply the functions*/
  &execfunc;
  /*Apply Stacking (and apply an OUTPUT after each set of stack functions)*/
  %IF &diffstack > 0 %THEN %DO;
    %DO curstack = 1 %TO &diffstack;
      &stack&curstack OUTPUT;
    %END;
  %END;
RUN;
```

The mapping macro generates a simplified version of the datastep that is generated by the mapping macro. When reading the source dataset, only the variables that are needed (as specified in the mapping specification document) are kept. The WHERE-clause(s) is/are applied to select only the appropriate records and the RENAME func+on(s) is/are applied to rename variables. The RECODE func+ons (and the code to check for unexpected values) are executed in the order as available in the mapping specification document and afterwards an OUTPUT statement is executed. If available the STACK2 func+on is executed in the order as available in the mapping specification document and afterwards an OUTPUT statement is executed. (same logic applies for STACK3 up to STACK[n])
Framework Macros

Source1 \rightarrow Mapped_Source1
Source2 \rightarrow Mapped_Source2
Source3 \rightarrow Mapped_Source3
Source4 \rightarrow Mapped_Source4
Source5 \rightarrow Mapped_Source5

Mapped_Source1 \rightarrow SDTM.AE
Mapped_Source2 \rightarrow SDTM.DM
Mapped_Source3 \rightarrow SDTM.DS
Mapped_Source5 \rightarrow SDTM.[xx]

%makefinaldomain

%mapping
WHERE, RENAME, COPY, FUNCTION, RECODE, STACK1-STACK[n], KEEP.

%apply_poststeps
POSTSTEP1-POSTSTEP[n], FUNCTION1-FUNCTION[n]

www.ocs-consulting.com/nl
© OCS Consulting
The `apply_poststeps` macro is used to obtain and execute the code specified within the POSTSTEP1-POSTSTEP[x] and FUNCTION1-FUNCTION[x] functions of the mapping specification document in the following order:

- When a function like “POSTSTEP[x]” is specified the exact SAS code as specified in brackets will be executed outside of a data step.
- Functions like “FUNCTION[x]” will be handled in exactly the same way as the FUNCTION used in the %mapping macro and will be executed within a DATA STEP. The only difference is the timing of when the code is executed:
%Apply_poststeps Macro

PROC SORT DATA=work.mapped_cdash_cdash_ae; BY usubjid; RUN;
DATA work.mapped_cdash_cdash_ae; MERGE work.mapped_cdash_cdash_ae(IN=a) insdtm.dm(KEEP = usubjid rfstdtc);
BY usubjid;
IF a;
RUN;
DATA work.mapped_cdash_cdash_ae;
SET work.mapped_cdash_cdash_ae;
%calculatedayno(refdate=rfstdtc, date=aestdtc, dayvar=aestdy);
%calculatedayno(refdate=rfstdtc, date=aeendtc, dayvar=aeendy);
RUN;

PROC SORT DATA=work.mapped_cdash_cdash_ae; BY aelltcd; RUN;
PROC SORT DATA=work.mapped_meddra_meddra; BY aelltcd; RUN;
DATA work.mapped_cdash_cdash_ae;
MERGE work.mapped_cdash_cdash_ae(IN=a)
work.mapped_meddra_meddra(KEEP = aelltcd aebdsycd aehltcd aehltdcd aehltdc aehltlcd aehltdcd aehltlcd)
BY aelltcd;
IF a;
RUN;
DATA work.mapped_cdash_cdash_ae;
SET work.mapped_cdash_cdash_ae;
aellt = STRIP(PUT(STRIP(PUT(aelltcd,BEST.)), $MD_LLT.));
RUN;
Framework Macros

`%mapping`:
- `WHERE`,
- `RENAME`,
- `COPY`,
- `FUNCTION`,
- `RECODE`,
- `STACK1-STACK[n]`, `KEEP`.

`%apply_poststeps`:
- `POSTSTEP1-POSTSTEP[n]`, `RECODE`,
- `FUNCTION1-FUNCTION[n]`
%Makefinaldomain

Using the near-final SDTM dataset and the ‘target metadata’ the macro %makefinaldomain will ensure that:

- Only variables that are defined in the ‘target metadata’ are kept in the output;
- Variables and domains get the attributes (type, length, label) as defined in the ‘target metadata’;
- Domains are sorted on the keys as defined in the ‘target metadata’ and the --SEQ variable is generated using those keys;
- The intermediate dataset is split into a SDTM ‘main’ domain and a Supplemental Qualifier domain (e.g. split into AE and SUPPAE).

Variables that need to go into the Supplemental Qualifier domain can be added to the intermediate ‘main’ dataset itself.
Standard Mappings

FRAMEWORK

Mapping Specifications

Standard Macros

Source data

SDTM

SDTM Standards (CDISC/company)

Standard mappings
All **standard rules** (i.e. the specification and function to convert source to target) are available in the **standard mapping specification**.

Conversion programs will use the specification and function as available in the standard mapping specification if standard rule is specified.

The specification and function specified in the study mappings specification will be compared with the standard mapping specification and warning messages are given if any difference is encountered.

More info: [PAPER CD03](www.ocs-consulting.com/nl)
Benefits

- Readability of the source-to-target mapping specification document;
- Study specific mappings never lead to adaptations of the programs, only to the mappings specification document;
- Distinction between standard and non-standard mapping rules;
- Standard metadata is used on several locations in the process which ensures all output generated is according to the defined standard. It is not possible to generate output that is not defined in the standards;
- All code that is generated and executed is SAS code and the MFILE option is used to print all executed code to a file. This is particularly useful when debugging or when the code needs to be sent to external parties;
- Warnings/errors are generated when the mapping specification document does not align with the source data or the standard mapping specification document;
- The available functions as currently available in the mapping specification document (in the column “FUNCTION”) can be extended if the need arises. For example, the STACK function was added at a later time.
Questions