Paper 170-2009

MedDRA data as SAS formats

Jim Groeneveld, OCS Biometric Support, Leiden, the Netherlands.

ABSTRACT

To use MedDRA® data with SAS® the ascii data files can be read into SAS datasets or the available datasets can be used directly. These datasets can be merged with available MedDRA coded data, possibly using indexes. This paper describes a method to process the (generated) datasets into SAS formats that can be applied directly without the need to merge datasets. In that way terms of the lower levels, like the Lower Level Term (LLT), can be directly converted into each of their associated terms at higher levels. Formats can be made for stepping up from each code level to each higher ordered code level. Additionally all codes can be formatted with their names (descriptions), abbreviations or whatever other information is available in the datasets. This may save space in resulting datasets, as additional variables (e.g. extensive texts) don't need to be stored in all records, but just the single link to the names instead, the associated SAS format. (SAS 8/9 (for Windows), intermediate skill level)

INTRODUCTION

MedDRA® data, available as ascii files (or SAS datasets), are traditionally used with the accompanying MedDRA browser software to view them. In order to process the data with other software the ascii data can be read and linked to available MedDRA coded data. With SAS® that is also possible. The ascii data are read into SAS datasets, one for each ascii file; with the newer MedDRA distributions the already available SAS datasets may be used. In order to create reports at different levels or to group available descriptions these datasets can be merged (using indexes or not) with available user data, medical datasets. In such datasets MedDRA coding may already have been applied on descriptions of medical issues (e.g. adverse events) using Lower Level Terms (LLTs). While doing so relatively large datasets may be generated in which each record contains the linked codes or MedDRA terms (names).

This paper describes an alternative way of incorporating MedDRA data into SAS, to combine it with available user data that contains the already coded descriptions (LLTs). Instead of merging MedDRA table data with user data, the MedDRA datasets firstly are converted into SAS formats using PROC FORMAT with the CNTLIN option. Formats can be generated this way for all MedDRA terms (LLT, PT, HLT, HLGT and SOC). With each format an association can be made between the MedDRA code and its name (description) or any other higher level code. The formats between different code levels thus serve as links between two levels of codes. All resulting formats, associations can be stored in a single format catalog.

The SAS program code to create the MedDRA formats is discussed here. The code may serve as an example to create additional formats, but a user generally only has to run the code (once per MedDRA version or language) to create the formats for himself. Subsequently the formats are available to be applied as and when desired.

Striking is the fact that the MedDRA system applies the various codes, existing of some number of digits, as numerical values and not as meaningless character values (as would have been expected). That is why all digit codes in this system are also stored in numeric variables in SAS datasets and specified as such in the resulting SAS format catalog(s). As a result all formats developed here are numerical formats.

For LLT to PT and PT to PSOC translation the associations are one to one, that is for each individual LLT there is only one PT and for each PT there is only one PSOC (this is not true the other way around, from higher to lower levels of course). For PT to HLT, HLT to HLGT and HLGT to SOC translations there are multiple associations possible. E.g. for each PT there may be more than one associated HLT and so on. These SAS formats are defined as MULTILABEL formats, specifically suited for use with the PROCs MEANS (SUMMARY) and TABULATE.

Other procedures and the SAS data step, however, do not (yet?) support MULTILABELs. These produce only the first matching formatted term code of the lower term code. That is a limitation of SAS at the moment. If it is desired to produce all multiple higher level associated term codes for a lower term code the usual merging of datasets (with the desired codes) may be most feasible. The unique association between a **code** and its **name** (description) may always be efficiently arranged via formats.

The associations with other classes of codes, like those of the ICD-9-CM or WHO-ART, can be treated likewise, but these are not included in this paper.

SAS Global Forum 2009

PROTOTYPE FORMATTING SYSTEM

1. SETTINGS

To create all possible formats and to activate them the following basic program is run and presented here for the understanding of the definitions inside:

```
* meddra.sas, settings, definitions, includes (all) and overview;
OPTIONS PS=63 LS=80 FORMCHAR = '|____|||___';
%LET MdraData = «directory path to MedDRA ascii data files»;
%LET MdraProg = «directory path to these SAS programs and formats»;
%LET MedDRAta = _MedDRA_; * name of format catalog, e.g. MedDRAnl ;
LIBNAME MedDRA "&MdraProg";
%INC "MedDRAll.sas"; * or: "«path_to»MedDRAll.sas";
PROC FORMAT FMTLIB LIBRARY=MedDRA.&MedDRAta; RUN; * Presenting all formats;
```

The Include statement of MedDRAll.sas includes all other (short) SAS programs together building all possible formats and storing it into the catalog MedDRA.&MedDRAta. Below some of those SAS programs are presented as examples (prototypes).

2. UNIQUE ASSOCIATIONS

Example SAS code to read the LLT data from the MedDRA ascii file and to store it in a SAS dataset (without using the available SAS dataset from the MedDRA distribution) is:

```
* llt.sas, reading llt.asc;
DATA MedDRA.llt (KEEP=llt_code llt_name pt_code llt_currency);
  INFILE "&MdraData.\llt.asc" DSD DLM='$' MISSOVER;
  LENGTH llt_name $100 llt_whoart_code $7 llt_costart_sym $21 llt_icd9_code $8
    llt_icd9cm_code $8 llt_icd10_code $8 llt_currency $1 llt_jart_code $6;
  INPUT llt_code llt_name pt_code llt_whoart_code llt_harts_code llt_costart_sym
        llt_icd9_code llt_icd9cm_code llt_icd10_code llt_currency llt_jart_code;
RUN;
```

Example SAS code to create formats (into a permanent format catalog MedDRA.&MedDRAta) out of these data is:

```
* llt_fmt.sas, translating llt dataset into llt format (codelist) and pt translation;
DATA llt;
  SET MedDRA.llt;
  Type = 'N'; * MedDRA codes always are numeric;
  LLT_fmt = 'LLT_fmt'; * Format name of llt codelist;
  llt_pt = 'llt_pt'; * Format name of llt to pt translation;
RUN;

PROC FORMAT LIBRARY=MedDRA.&MedDRAta
  CNTLIN=llt (RENAME=(llt_code=Start llt_name=Label LLT_fmt=FmtName));
RUN;

PROC FORMAT LIBRARY=MedDRA.&MedDRAta
  CNTLIN=llt (RENAME=(llt_code=Start pt_code=Label LLT_pt=FmtName));
RUN;
```

3. NON-UNIQUE ASSOCIATIONS

SAS code to read MedDRA data with multiple associations (without using the available SAS dataset) is for example:

```
* hlt_pt.sas, reading hlt_pt.asc;
DATA MedDRA.hlt_pt;
  INFILE "&MdraData.\hlt_pt.asc" DSD DLM='$' MISSOVER;
  INPUT hlt_code pt_code;
RUN;
```

Creating formats from these data needs the (example) code:

```
* pt_hlt.sas, translating hlt_pt dataset into pt to hlt translation;
DATA hlt_pt;
  SET MedDRA.hlt_pt;
  Type = 'N'; * MedDRA codes always are numeric;
  HLO = 'M'; * indicating MULTILABEL or multiple codes;
  pt_hlt = 'pt_hlt'; * Format name of pt to hlt translation;
```

SAS Global Forum 2009

```
RUN;

PROC FORMAT LIBRARY=MedDRA.&MedDRAta
  CNTLIN=hlt_pt (RENAME=(pt_code=Start hlt_code=Label pt_hlt=FmtName));
RUN;
```

In similar ways most of the other ascii files are read and converted into formats. The 8 ascii files used here are: **llt.asc, pt.asc, hlt.asc, hlgt.asc, soc.asc, hlt_pt.asc, hlgt_hlt.asc, soc_hlgt.asc**.

4. FORMATS

The prototype MedDRA formats system makes the following formats and associations available:

a. **llt_fmt** llt_code and name (llt_name)
b. **llt_pt** llt to pt translation, one to one (unique)
c. **pt_fmt** pt_code and name (pt_name)
d. **pt_psoc** pt to primary (psoc) translation, one to one (unique)
e. **pt_hlt** pt to hlt translation, multiple values (non-unique)
f. **hlt_fmt** hlt_code and name (hlt_name)
g. **hlt_hlgt** hlt to hlgt translation, multiple values (non-unique)
h. **hlgt_fmt** hlgt_code and name (hlgt_name)
i. **hlgt_soc** hlgt to soc translation, multiple values (non-unique)
j. **soc_fmt** soc_code and name (soc_name)

5. APPLICATION

Applying the formats (from the permanent format catalog MedDRA.&MedDRAta) in SAS programs is done in the usual way, defining the FMTSEARCH directory or catalog (OPTIONS command) and defining permanent format associations in a SAS data step while saving a new (permanent) SAS dataset, or temporary associations in procedures (FORMAT statement). Another way to apply the formats is to use them with data step functions like PUT to obtain for example the names (descriptions), and with both PUT and INPUT to translate from one level to the other. A code example to translate an available numeric LLT code (after coding of medical descriptions) into a numeric PT code is:

```
PT_code = INPUT ( PUT ( LLT_code , LLT_PT. ) , 8. ) ;
```

With higher MedDRA coding levels and MULTILABEL formats the above example would be similar. Only one, the first association is returned. Such numeric to numeric translations can be built as macros. A general example is:

```
%MACRO NumToNum (Value, Format); INPUT ( PUT ( &Value, &Format ) , 8. ) %MEND NumToNum;
```

where the applied macro call would be: `PT_code = %NumToNum ( LLT_code , LLT_PT. ) ;`

The developed formats may also be specified using the macro functions %SYSFUNC and %QSYSFUNC if MedDRA codes are available as macro variables.

In order to translate directly from one level to another one, while skipping intermediate levels, the PUT/INPUT functions or the macro calls to NumToNum have to be nested, e.g. to translate from the LLT code to the SOC code via the primary matches one could program:

```
soc_code = %NumToNum ( %NumToNum ( %NumToNum ( %NumToNum ( LLT_code , llt_pt. ), pt_hlt.),
hlt_hlgt.), hlgt_soc. ) ;
```

PRODUCTION FORMATTING SYSTEM

1. DISADVANTAGE OF PROTOTYPE

Though the prototype system technically works very well, it has a significant drawback. Using MULTILABEL formats usually only returns the first non-uniquely formatted value. It has appeared that the first specified value in the MedDRA data is not always the most important one; it is not always the value that leads from a lower level (e.g. pt) to the primary soc of the pt. In other words the most important match towards the primary soc is not always the first one of a multiple match between two adjacent level codes. This can be demonstrated quite easily by applying the nested macro call above; it most often yields the correct primary soc, but in quite some cases also yields another (not primary) soc, while comparing its result to the primary soc-s in the pt data file. So it is important to have the associated higher level codes that correspond to the **route** towards the primary soc as the first entry to a MULTILABEL SAS format. That causes in those instances that (primary) associated higher level code (leading to the primary soc) will be returned as a result of formatting.

The word **"route"** here and below refers to the associations between successive, adjacent level values that correspond to a defined relation between a pt and its primary soc or a secondary soc. A pt is uniquely associated with only one primary soc, but (non-uniquely) with zero or more secondary soc-s.

SAS Global Forum 2009

2. ALGORITHM

The general program that configures the settings is basically the same as in the prototype system. The way the llt codes are read and processed in relation to their unique association with pt codes is unchanged too: See the SAS example code presented earlier in which:

a. LLT ascii data have been read into SAS dataset llt.sas7bdat by llt.sas, only keeping variables llt_code llt_name pt_code llt_currency;
b. LLT format (code and name) and llt-to-pt translation (llt_code and pt_code) have been used to produce SAS formats **llt_fmt** and **llt_pt**, derived from LLT dataset by llt_fmt.sas;

Instead of applying the individual data files **pt.asc**, **hlt.asc** and **htgt.asc** the data file **mdhier.asc** is being used as it contains an indication of the records with relations leading from a pt to the primary soc (Primary_SOC_fg). Besides it contains all the data from the separate files. The purpose is to arrange an order in which for each level (pt, hlt or hlgt) the record with a primary soc (Primary_SOC_fg = 'Y') is the first one for each unique value of the concerning level term. Records with Primary_SOC_fg = 'N' should be arranged below that record (in arbitrary order). This initially had been realized with SAS code.

However, an additional problem that emerged was that not all records in the data file mdhier.asc (or its deduced SAS dataset) have either a 'Y' or 'N' value for the variable Primary_SOC_fg. There are also empty, missing values. These also occurred with values at other lower levels that did not even have an associated 'Y' or 'N' in other records with the same (lower level) value. So, instead of applying the existing variable Primary_SOC_fg another strategy has been used to differentiate between (routes to) primary and secondary soc-s.

Another indication of a primary soc is present in the data file pt.asc (and its deduced SAS dataset) as pt_soc_code. Before reordering the mdhier data, as outlined above, it has been merged with the pt data (by pt_code). Another primary soc indication was generated as 'Y' if the **pt_soc_code** value from the **pt** data was equal to the **soc_code** (not pt_soc_code) value of the **mdhier** data. During the merge step some checks were done that gave the following results for the merged records where the value match was evident:

a. In hundredths of cases the value of Primary_SOC_fg was not 'Y', but missing (empty);
b. in hundredths of cases the value of pt_soc_code in the mdhier data was missing or deviant. If deviant it appeared that the present values were corresponding with the "correct" primary soc codes, but that 1 to 6 digits were missing from the end, leaving values of 7 or less digits (as numeric values).

(It looks like possible incomplete data in the MedDRA data.)
However, in far most cases there appeared consistency between the primary soc value from both merged datasets.

Having deduced the primary soc-s as good as possible this way all data was sorted according to the lower level, for which a format had to be created, and the primary soc indicator, where the positive indication ('Y') came first. This had been done separately for each lower level, thus for each separate format to generate. The result was a set of data determining the concerning format in which the "primary route" was specified as the first one of all possible, multiple routes as good as possible. Existing associations have not been changed (within), only their order (between). Using this algorithm the following MULTILABEL formats were generated: **pt_hlt**, **hlt_hlgt**, **hlgt_soc**, **pt_hlgt**, **pt_soc**, **hlt_soc**. The generated name formats **pt_fmt**, **hlt_fmt**, **hlgt_fmt** and **soc_fmt** are the same as generated in the prototype system. The unique format **pt_psoc** was generated directly and unambiguously from the data in pt.asc.

Checks have been performed that yield the following results:

a. Hundredths of specific pt_codes and their associated hlt_codes completely lacked a value of 'Y' for Primary_SOC_fg in the original mdhier data. The only occurring values for those codes were 'N' and missing. This also applies to combinations of pt_codes and hlgt_codes, pt_codes and soc_codes, hlt_codes and hlgt_codes, hlt_codes and soc_codes, and (12) hlgt_codes and soc_codes. That means that the first and most important value in the resulting MULTILABEL format does not represent a route towards a primary soc according to the information in the mdhier data (Primary_SOC_fg). It does, however, represent a route to the primary soc in agreement with the one-to-one relation between the pt and the primary soc. (This is the same finding as above);
b. Additionally there are 4 hlgt_codes that have more than one different, associated primary soc, according to both the original Primary_SOC_fg value and the deduced indication from the pt to psoc relation in the pt.asc data. This is due to different original pt-s that "pass" the same hlgt value on their "route" towards their respective, different primary soc-s and are assumed valid. These very few, associated primary soc-s hence are (legally) ambiguous and it does not matter which one of both values comes first in the MULTILABEL format **hlgt_soc**.

In the majority of the cases there are only single associations between values of different levels. If there are more of them then there is just one, single association that is part of a route to the primary soc in most cases; the only exceptions are those 4, discussed above.

3. SAS PROGRAMS

The MedDRA formats production system needs the following SAS programs, in addition to the ones already indicated (llt_sas and llt_fmt.sas):

MedDRAll.sas contains:

```
%INC "llt.sas";
%INC "llt_fmt.sas";
```

SAS Global Forum 2009

```
%INC "pt.sas";
%INC "pt_psoc.sas";
%INC "mdhier.sas";
%INC "Low_High.sas";
```

The last 4 programs in there are detailed below; the first two already have been discussed with the prototype.

pt.sas contains:

```
* pt.sas, reading pt.asc;
DATA MedDRA.pt (KEEP=pt_code pt_name pt_soc_code);
  INFILE "&MdraData.\pt.asc" DSD DLM='$' MISSOVER;
  LENGTH pt_name $100 null_field $1 pt_whoart_code $7 pt_costart_sym $21
    pt_icd9_code $8 pt_icd9cm_code $8 pt_icd10_code $8 pt_jart_code $6;
  INPUT pt_code pt_name null_field pt_soc_code pt_whoart_code pt_harts_code
        pt_costart_sym pt_icd9_code pt_icd9cm_code pt_icd10_code pt_jart_code;
RUN;
```

pt_psoc.sas contains:

```
* pt_psoc.sas, translating pt dataset into psoc translation;
DATA pt;
  SET MedDRA.pt;
  Type = 'N'; * MedDRA codes always are numeric;
  pt_psoc = 'pt_psoc'; * Format name of pt to pt translation;
RUN;
PROC FORMAT LIBRARY=MedDRA.&MedDRAta
  CNTLIN=pt (RENAME=(pt_code=Start pt_soc_code=Label pt_psoc=FmtName));
RUN;
PROC DATASETS NOLIST; DELETE pt; QUIT;
```

mdhier.sas contains:

```
* mdhier.sas, reading mdhier.asc;
DATA MedDRA.mdhier (DROP = null_field);
  INFILE "&MdraData.\mdhier.asc" DSD DLM='$' MISSOVER;
  LENGTH pt_name hlt_name hlgt_name soc_name $100 soc_abbrev $5 null_field $1
         primary_soc_fg $1;
  INPUT pt_code hlt_code hlgt_code soc_code pt_name hlt_name hlgt_name soc_name
        soc_abbrev null_field pt_soc_code primary_soc_fg;
RUN;
```

Finally, **Low_High.sas** contains the most important, most extensive code including checks, as well as the algorithms that have been described before. These have been implemented in the code and are explained in more detail in the comments inside.
In the code below the less essential, just checking code is printed in italics:

```
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
* Translating mdhier dataset into lower to higher level translation ;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;

* Combine (merge) mdhier data with pt data for primary SOC;
PROC SORT DATA=MedDRA.mdhier OUT=mdhier; BY pt_code; RUN;
PROC SORT DATA=MedDRA.pt     OUT=pt    ; BY pt_code; RUN;

DATA mdhier;
  MERGE mdhier (RENAME=(pt_soc_code=psoc_code) IN=INmdhier) pt (IN=INpt);
    BY pt_code;
  FILE 'Discrep.txt';

  IF (soc_code /*from mdhier*/ EQ pt_soc_code /*from pt*/) THEN DO;
    Prim_SOC = 'Y';
    IF (Primary_SOC_fg NE 'Y') THEN
      PUT '*** Primary_SOC-fg not Y: ' pt_code= soc_code= psoc_code= Primary_SOC_fg=;
    IF (psoc_code NE pt_soc_code) THEN
      PUT '*** psoc NE pt_soc_code: ' pt_code= soc_code= psoc_code= Primary_SOC_fg=;
  END;

  ELSE Prim_SOC = 'N';
  IF (INpt AND NOT INmdhier) THEN
```

SAS Global Forum 2009

```
    PUT '*** not in mdhier: ' pt_code= pt_soc_code=;
  ELSE IF (NOT INpt AND INmdhier) THEN
    PUT '*** not in pt: ' pt_code= soc_code= psoc_code= Primary_SOC_fg=;
RUN;

* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
* Use mdhier data with primary soc indication from pt data, because ;
* Primary_SOC_fg is not always 'Y' where it is expected to be so,   ;
* to extract all level names and all lower to adjacent higher level ;
* codes for creating (multilabel) formats, in which the first match ;
* of the multilabel formats correspond with the path to the primary ;
* SOC. This match is the only one obtained outside MEANS, TABULATE. ;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
* Yet, there are more solutions upward leading to different primary ;
* SOCs, dependent on the pt of departure. E.g. for a given hlgt_code;
* there may occasionally be more than one upward primary SOC. That  ;
* means that there may be routes from lower to higher levels, which ;
* are ambiguous. Anyway, it is the intention to have at least one   ;
* adjacent higher level for a given lower level that leads to a     ;
* primary SOC as the first of multiple records for that lower level.;
* Furthermore, going from a lower level to a higher one, while      ;
* skipping one or more intermediate levels can be done by going via ;
* the adjacent intermediate formats, ending up at the higher level, ;
* but this does not always correspond to the best solution if going ;
* to the higher level directly via the direct format.               ;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;

%MACRO Low_High (FromTo=, Lower=, Higher=, LowName=, LowFmt=, PrimSOC=Prim_SOC);
 %IF (%LENGTH(&Higher)) %THEN %DO;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
* Firstly for each pair of adjacent or remote codes for which a     ;
* format has to be created the possibly multiple occurring records  ;
* (for hlt levels and up) have to be reduced to single records,     ;
* unique pairs, including the possible Primary SOC indicators Y/N.  ;
* During the reduction the indicator value Y is kept, if present,   ;
* while removing the indicator value N. If Y is not present (first) ;
* then only the unique record with the indicator value N is kept.   ;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;

  PROC SORT DATA=mdhier (KEEP=&Lower &PrimSOC &Higher Primary_SOC_fg)
            OUT=&FromTo NODUPKEY;
    BY &Lower &Higher DESCENDING &PrimSOC;
  RUN;

  DATA &FromTo;
    SET &FromTo;
    BY &Lower &Higher DESCENDING &PrimSOC;
    IF (FIRST.&Higher); * keep only first occurrences of Higher value;
* This Higher value has a corresponding indication of a primary SOC ;
* if it is present, otherwise it has the indication of its absence. ;
  RUN;

* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
* Now resort the data such that the records with a primary SOC      ;
* indication come before the ones that don`t have that indication.  ;
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;

  PROC SORT DATA=&FromTo NODUPKEY;
    BY &Lower DESCENDING &PrimSOC &Higher;
  RUN;

  DATA &FromTo;          /* &FromTo is format name, also used as dataset name */
    SET &FromTo;
    BY &Lower DESCENDING &PrimSOC &Higher;
    FILE "&FromTo..txt";

    IF (NOT FIRST.&Lower AND /*&PrimSOC*/Primary_SOC_fg EQ 'Y') THEN
      PUT '*** Non-unique Primary_SOC_fg=Y for ' &Lower= &Higher=;
                                                    /* more 'Y' per pair */
    ELSE IF (FIRST.&Lower AND /*&PrimSOC*/Primary_SOC_fg NE 'Y') THEN
      PUT '*** No Primary_SOC_fg=Y at all for ' &Lower= &Higher=;
```

SAS Global Forum 2009

```
                                                                    /* no 'Y' at all */
    IF (NOT FIRST.&Lower AND &PrimSOC EQ 'Y') THEN
      PUT "*** Non-unique &PrimSOC=Y for " &Lower= &Higher=;
                                                              /* more 'Y' per pair */

   Type = 'N';                              /* MedDRA codes always are numeric */
    HLO = 'M';                         /* indicating MULTILABEL or multiple codes */
    &FromTo = "&FromTo";          /* Format name of Lower to Higher translation */
  RUN;

  PROC FORMAT LIBRARY=MedDRA.&MedDRAta
    CNTLIN=&FromTo (RENAME=(&Lower=Start &Higher=Label &FromTo=FmtName));
  RUN;
 %END;

* - - - - - - - - - - - - - - - - - - - -;
* Create format for term description ;
* - - - - - - - - - - - - - - - - - - - -;
 %IF (%LENGTH(&LowName))%THEN %DO;
  PROC SORT DATA=mdhier (KEEP=&Lower &LowName )
            OUT=&FromTo NODUPKEY;
    BY &Lower;
  RUN;

  DATA &FromTo;
    SET &FromTo;
    BY &Lower;
    Type = 'N'; * MedDRA codes always are numeric;
    &LowFmt = "&LowFmt";
  RUN;

  PROC FORMAT LIBRARY=MedDRA.&MedDRAta
    CNTLIN=&FromTo (RENAME=(&Lower=Start &LowName=Label &LowFmt=FmtName));
  RUN;
 %END;

  PROC DATASETS NOLIST; DELETE &FromTo; QUIT;
%MEND Low_High;

* Create format(s) pt_hlt, pt_fmt;
%Low_High (FromTo=pt_hlt, Lower=pt_code, Higher=hlt_code, LowName=pt_name, LowFmt=pt_fmt);

* Create format(s) hlt_hlgt, hlt_fmt;
%Low_High (FromTo=hlt_hlgt, Lower=hlt_code, Higher=hlgt_code, LowName=hlt_name,
LowFmt=hlt_fmt);

* Create format(s) hlgt_soc, hlgt_fmt;
%Low_High (FromTo=hlgt_soc, Lower=hlgt_code, Higher=soc_code, LowName=hlgt_name,
LowFmt=hlgt_fmt);

* Create format(s) soc_fmt;
%Low_High (FromTo=soc, Lower=soc_code, Higher=/*empty*/, LowName=soc_name,
LowFmt=soc_fmt);

* Create format(s) pt_hlgt;
%Low_High (FromTo=pt_hlgt, Lower=pt_code, Higher=hlgt_code, LowName=, LowFmt=);

* Create format(s) pt_soc;
%Low_High (FromTo=pt_soc, Lower=pt_code, Higher=soc_code, LowName=, LowFmt=);

* Create format(s) hlt_soc;
%Low_High (FromTo=hlt_soc, Lower=hlt_code, Higher=soc_code, LowName=, LowFmt=);

PROC DATASETS NOLIST; DELETE mdhier pt; QUIT;
```

4. FORMATS

In this formatting system MedDRA table data (ascii code lists) have been processed, converted into SAS datasets and formats by way of the following programs (note that older MedDRA versions did not yet have accompanying SAS datasets next to the ascii files. That is why the ascii files are to be read initially in these examples, but the currently existing SAS datasets, delivered with the MedDRA data can be used instead):

SAS Global Forum 2009

a. LLT ascii data have been read into SAS dataset llt.sas7bdat by llt.sas, only keeping variables llt_code llt_name pt_code llt_currency;
b. LLT code and name and llt-to-pt translation (llt_code and pt_code) have been used to produce SAS formats **llt_fmt** and **llt_pt**, derived from LLT dataset by llt_fmt.sas;
c. PT ascii data have been read into SAS dataset pt.sas7bdat by pt.sas, only keeping variables pt_code pt_name pt_soc_code;
d. pt-to-primary-soc translation (pt_code and pt_soc_code) have been used to produce SAS format **pt_psoc**, derived from PT dataset by pt_psoc.sas;
e. all terms, excluding llt (and its name) have been read into SAS dataset mdhier.sas7bdat by mdhier.sas. These terms include the pt, hlt, hlgt, soc, their names (descriptions), the Primary SOC_fg, and the pt_soc_code;
f. all codes and names, pt_code, pt_name, hlt_code, hlt_name, hlgt_code, hlgt_name, soc_code, soc_name from the mdhier dataset have been used to generate the descriptive formats **pt_fmt**, **hlt_fmt**, **hlgt_fmt** and **soc_fmt** by the program Low_High.sas;
g. all possible translations from lower to higher level codes, pt_code, hlt_code, hlgt_code and soc_code, have been used to generate pairwise MULTILABEL formats from every lower to every higher level code in the mdhier dataset by the program Low_High.sas. These formats include **pt_hlt**, **hlt_hlgt**, **hlgt_soc**, **pt_hlgt**, **pt_soc** and **hlt_soc**.

5. APPLICATION

The code presented above directly generates formats between all lower and higher level term combinations from the pt level. That means that deducing formatted values by nested %NumToNum macros are not applicable (and might be somewhat deviating). The only case in which a nested macro specification is valid and needed, is while relating a higher level value to an llt, e.g.:

```
soc_code = %NumToNum ( %NumToNum ( LLT_code , llt_pt. ), pt_soc.) ;
```

In order to use the generated format catalog &MedDRAta from any user program one basically needs the program code:

```
OPTIONS FMTSEARCH=(MedDRA.&MedDRAta); * with defined macro variable MedDRAta;
```

where the LIBNAME MedDRA already should have been assigned.

ADVANTAGES

The advantages of applying MedDRA formats over merging of various MedDRA datasets with user datasets are:
a. No need to merge datasets all the time, just create the format catalog once and apply the formats from then onwards;
b. Less space consuming datasets as the formatted values are not necessarily part of the datasets themselves;
c. Central MedDRA formats, created only once. Just recreate the format catalog once MedDRA has been updated;
d. Ease of use with different languages: create language specific format catalogs to apply as desired.

SUPPORTED SAS VERSIONS AND OPERATING SYSTEMS

The prototype system has initially been developed for SAS version 8.x and tested with MedDRA versions 5 and 6; it was also tested with SAS version 9.1.3 and (English language) MedDRA versions 9, 10 and 11. The production system has run with SAS version 9.1.3 and tested with (English language) MedDRA versions 9, 10 and 11. The size of resulting files in terms of records and formatted values has been compared to those reported with the MedDRA documentation and found to match. The formatting system probably will run with any SAS version from version 8, supporting MULTILABEL formats, as well as with future MedDRA versions. The operating system used was initially MS Windows 2000 and is currently MS Windows XP, but the system is expected to operate correctly with many other operating systems and platforms too.
The SAS code of the production system can be downloaded from: http://home.hccnet.nl/jim.groeneveld/meddrafmt

REFERENCES

1) SAS Institute Inc. 2004. Base SAS® 9.1.3 Procedures Guide. Cary, NC: SAS Institute Inc.
2) MedDRA® is a registered trademark of the International Federation of Pharmaceutical Manufacturers and Associations.

CONTACT INFORMATION

Author Name : Y. (Jim) Groeneveld
Company : OCS Biometric Support
Address : PO BOX 490
5240 AL Rosmalen
The Netherlands
Work Phone : +31 (0)71 572 1828
Fax : +31 (0)71 576 5040
Email : questions@ocs-biometricsupport.com
Web : www.ocs-biometricsupport.com
Author's website : home.hccnet.nl/jim.groeneveld