PharmaSUG 2017 - Paper QT09

Basic SDTM House-Keeping

Emmy Pahmer, inVentiv Health Clinical

ABSTRACT

When creating SDTM datasets, there are a few simple checks that can be performed to help spot potential problems. These checks will be presented here. They are generally for common programming errors, not for checking the validity of source values or cross-checking certain values against others. In other words, basic house-keeping.

INTRODUCTION

Creating CDISC SDTM (Study Data Tabulation Model) datasets tends to be an iterative process, with several rounds of creation/correction. Our goal is avoid repeating errors by creating checks to spot them as early as possible in the process. We'll look at some examples and underscore how this type of checking can be beneficial in various circumstances.

SOME EXAMPLES

Here are some cases you might come across. They are provided as examples in the case of SDTM data, but the concept can be transferred to different types of data as well.

IS THE VARIABLE POPULATED?

Very simple! Yet how often do we run domains and end up with missing data?

```
if missing(LBTEST) then put 'W' 'ARNING: LBTEST is not populated';
```

CHECK THAT KEY VARIABLES IDENTIFY UNIQUE RECORDS

Key variables can change from one study to another, for a particular domain, depending on the design. So, to avoid having duplicates, and potential problems with your sequence (--SEQ) variable, this is important to check. Methods to check this can include PROC FREQ, PROC SQL and PROC SORT. Here's an example using PROC SORT with the DUPOUT option.

```
%macro check_for_dups (domain = , out_libname =);

/* PUT KEY VARS INTO MACRO VARIABLE */
data _null_;
    set int.domains (where = (domain = "&domain")) ;
    call symputx('keyvars', keyvars);
run;

/* CHECK THAT THERE ARE NO DUPS WITH KEY VARS */
proc sort data = &out_libname..&domain. out = temp_key nodupkey dupout =
temp.&domain._key;
    by &keyvars ;
run;
```

```
data _null_;
    set temp.&domain.&no._key;
    if _n_ = 1 then
    do;
        put "W" "ARNING: Key variables for &domain do not identify unique
records.";
         put "W" "ARNING- (%trim(&keyvars.))";
        put "W" "ARNING- See DATA\TEMP\&domain.&no._key.sas7bdat for
duplicates.";
        stop;
    end;
run;

%mend check_for_dups;
```

CHECK THAT THERE ARE NO VARIABLES POPULATED WITH "."

When converting from numeric to character and not taking missing values into consideration, we can end up with a value of "." in a character field. This can also happen when source data formats change and there are automatic conversions. It can be checked for all character variables in one shot using an array.

```
   array charvars[*] _character_;   /* this array used in examples below as
well */
   do k=1 to dim(charvars);
        if charvars[k]= '.' then put 'W' 'ARNING: Check value of . -> '
_n_= charvars[k]=;
    end;
```

CHECK IF --DTC VARIABLES HAVE A TIME PART OF 00:00:00

This is more often than not a sign that only the date part was available, therefore only the date should be presented. Depending on the circumstances, it is possible that the time part represents exactly midnight so it should be verified.

Using the same array method as above:

```
        if length(charvars[k]) = 16 and find(charvars[k], 'T00:00') then
put 'W' 'ARNING: time part is 0. ' _n_= charvars[k]=;

        if length(charvars[k]) = 19 and find(charvars[k], 'T00:00:00') then
put 'W' 'ARNING: time part is 0. ' _n_= charvars[k]=;
```

CHECK THAT UNWANTED VALUES FROM THE SPECIFICATIONS ARE REMOVED

You may be getting certain values from your database specifications, with values such as "N/A", "Empty" or "Null". Since they should not be presented in the data, make sure that they have been removed.

Using the same array method as above:

```
       if upcase(charvars[k]) = '[EMPTY]' then put 'W' 'ARNING: remove
[Empty]';
```

CHECK THAT TEXT ISN'T BEING CUT OFF

If a variable has a length of 200 and the 200th character is not blank, check that data isn't being cut off.

```
  %let var = COVAL;
```

In data step:

```
    last_words = substr(&var, findc(&var, ' ', ,170)); /* get the last few
words to help see where data is being cut off */
    last_char = substr(&var, 200, 1);
    if last_char ne ' ' then put 'W' "ARNING: Check if &var is being cut off.
" last_words=;
```

ARE CHARACTER VARIABLES LEFT ALIGNED?

I can't think of any circumstances where a character variable should not aligned to the left. When a numeric value is automatically converted to character, it floats somewhere in the middle of the field, and it's not easy to see with PROC PRINT or using a PUT statement.

Using the same array method as above:

```
if not missing(charvars[k]) and substr(charvars[k],1,1) = ' ' then
put 'W' 'ARNING: char var not left aligned. ' _n_= charvars[k]=;
```

CONCLUSION

Programmers can produce cleaner data and catch errors before the data leaves their hands with a few easy-to-code checks. Sometimes everything is correct and then there's a minor change to the source data or database specifications that can lead to data that was previously imported as numeric, to be imported as character, or vice versa. These examples can be used for SDTM data but the concept of basic checking can be applied to any kind of data, sometimes based on the data standards, or just common sense. Any time there's an error in the data, we can ask ourselves if we can create a check for this to avoid it in the future.

ACKNOWLEDGMENTS

Thanks to Sophie Arlix, Senior Statistical Programmer/Senior Clinical Data Manager, inVentiv Health.

CONTACT INFORMATION

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

Emmy Pahmer
InVentiv Health Clinical, Montréal, Canada
emmy.pahmer@inventivhealth.com