Paper 98-26

A SAS Macro to Isolate All Date Values from a Data Library into a SAS Dataset

Michael L. White, Kendle International, Cincinnati, OH

ABSTRACT

This paper will describe a method we devised to accumulate a dataset containing all SAS date variables' values in a SAS data library. With the resultant dataset, we were able to make comparisons with a baseline date such as randomization or date of initiation of study treatment and isolate dates that were recorded or entered incorrectly. We used SQL and SAS Macro language. The paper is appropriate for SAS users with a basic knowledge of the SAS Macro facility.

INTRODUCTION

In a large phase III clinical trial, there is a large amount of data collected on Case Report Forms (CRFs) and entered into a database. In the resultant database, there are a large number of date variables, which can be key to the final statistical analysis. Many of these dates are related to each other and need to meet a certain chronological order. For instance, laboratory collections and efficacy evaluations should be after the date of randomization or dates of death should be on or after all other dates of patient evaluation. Evaluations collected at the week 8 visit should be approximately 56 days after randomization. Typical recording and entry errors include the transposition of numbers in the dates or recording years incorrectly in January.

Our task was to devise a method to isolate these dates into one dataset in order to make it easier to identify values that were out of range when compared to a standard date for a patient, for instance date of randomization. We used the DICTIONARY.COLUMNS resource available in PROC SQL and SAS Macros in order to do this.

TECHNICAL DETAILS

First, using PROC SQL we created a data table that contained all date-related variables in defined libraries. The table contains any variable in the defined libraries, which had the word DATE in the label or had a date format applied to the variable making sure we did not include SAS date/time variables.

```
PROC SQL;
  create table dates as
  select libname, memname, name
  from dictionary.columns
  where type='num' and memtype='DATA' and
  (index(upcase(label), "DATE") > 0 or
  index(format, "DATE") > 0)) and
  index(format,"DATETIME")=0
  order by libname, memname, name;
QUIT;
```

The following macro puts all date variables from one selected dataset into a macro variable string called THESEDAT and then uses the string to create a output dataset with each date value on a separate observation. The macro N_CNTWRD is a macro that counts the number of variables in a specified list and is not presented here. Looping through each entry in the string &THESEDAT and outputting an observation for each nonmissing value of a date variable creates the dataset &OUTSET. The output dataset contains the identifier variables, extra variables deemed necessary for later reports, the dataset where the date value was stored, the variable name, and the variable's value. One of the more important macro variables defined below is EXTRAVAR. Typical values for this would be WEEK, VISITDT, or CRFPAGE. If the macro is run for one specified dataset, there are no issues. However, if the macro is run using the second macro %ALLDATES that will be described later, each variable cited in IDVARS and EXTRAVAR need to be present in all datasets in the library.

```
%MACRO DATECHK(lib=SOURCE, indata=, idvars=,
extravar=, outset=);
  PROC SQL;
    Select distinct NAME
    Into :thesedat separated by ' '
    from DATES
    where libname="%upcase(&lib)" and
          memname="%upcase(&indata)";
  quit;

  %n_cntwrd(string=&thesedat, cntvar=VARCNT)

  DATA &OUTSET;
    Set source.&indata(keep=&idvars &extravar
                            &thesedat);
    length variable dataset $ 8;

    %do i=1 %to &varcnt;
       %let thisvar =
            %upcase(%scan(&thesedat,&i));
       dataset="&indata";
       variable="&thisvar";
       value=&thisvar;
       if value ne . then output;
    %end;
    keep &idvars &extravar dataset variable
         value;
    format value date9.;
  run;
%mend;
```

Here is an example of using %DATECHK. Suppose you have a dataset called LAB, which includes date variables collection date and lab analysis date called DTCOLL and DTANA respectively and identifying variable UNIQPAT. Other variables that are considered important for subsequent reports are WEEK and a code variable for the lab test performed (LABTEST). A typical PROC PRINT would look like this.

```
                PROTOCOL XX-XXX
          PRINTOUT OF LABORATORY DATA

OBS UNIQPAT DTCOLL WEEK LABTEST DTANA
 1    1001  01JAN01  2   CALC   01JAN01
 2    1001  01JAN01 20   HEMG   01JAN01
 3    1001  10JAN01  2   PLAT   01JAN01
```

The call of the macro would look like this.

%DATECHK(LIB=SOURCE, INDATA=LAB, IDVARS=UNIQPAT, EXTRAVAR=WEEK LABTEST, OUTSET=LABDTCHK)

For each observation with complete data for DTCOLL and DTANA, two observations would be generated in LABDTCHK. Here is the PROC PRINT for LABDTCHK.

```
                PROTOCOL XX-XXX
             PRINTOUT OF LABDTCHK
            LOOK FOR DATE PROBLEMS
O
B
S DATASET UNIQPAT WEEK LABTEST VARIABLE VALUE

1   LAB      1001    2   CALC    DTCOLL 01JAN01
2   LAB      1001    2   CALC    DTANA  01JAN01
3   LAB      1001   20   HEMG    DTCOLL 01JAN01
4   LAB      1001   20   HEMG    DTANA  01JAN01
5   LAB      1001    2   PLAT    DTCOLL 10JAN01
6   LAB      1001    2   PLAT    DTANA  01JAN01
```

Suppose the patient 1001 was randomized on 18DEC2000. Then 01JAN01 would be 14 days after the date of randomization or exactly 2 weeks post randomization. Using a PROC UNIVARIATE for observations for week 2, we would hope to find the entry error for collection date of platelet s of 10JAN01. Also, observations collected at week 20 should be approximately 140 days post randomization. We should be able to find through our checks that the study week for the HEMG observation was recorded as 20 instead of the correct 2.

A second macro below was written to run the %DATECHK macro for all datasets in a selected permanent SAS library and to create one final output dataset containing all date values for those datasets. First, all datasets in the specified library are accumulated into the macro string variable &ALLDATA. Next, we count the number of datasets in that macro string using the N_CNTWRD macro and then loop through each dataset running the %DATECHK macro for each dataset. After all datasets have been run, we set all of the datasets together ready for further use.

```
%macro alldates(library=source);
  PROC SQL;
    select distinct memname
    into :alldata separated by ' '
    from dates
    where libname = "%upcase(&library)";
  quit;

  %n_cntwrd(string=&alldata, cntvar=SETCNT)

  %do j = 1 %to &setcnt;
      %let thisset =
           %upcase(%scan(&alldata,&j));
      %datechk(lib=&library, IDVARS=UNIQPAT,
               extravar=CRFPAGE,
               outset=&thisset)
  %end;
  data final;
    set &alldata;
    by uniqpat;
  run;
%mend;
```

After running the %ALLDATES macro we were able to merge the resultant dataset with a dataset containing the date of randomization and other demographic characteristics by UNIQPAT. Using basic exploratory data analysis methods like PROC UNIVARIATE, we were able to isolate those values which "stick out like a sore thumb" or other observations that possibly could have been misentered. We generated exclusion reports for submission to our data management group, and cleaned up the data for the final statistical analysis.

CONCLUSION

Our task was to isolate out of range date values in our SAS data libraries. Using a little bit of SQL coding and SAS data step programming, we were able to compare dates across multiple datasets and find those values which were recorded incorrectly. Repeated generations of our reports, along with further suggestions from our data management colleagues, allowed us to refine our extreme value ranges and get our data cleaned in time for our final statistical analysis.

Although our programming centered on the editing of date values, the methodology presented here could easily be modified to handle other types of variables.

CONTACT INFORMATION

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

Michael L. White
Kendle International
1200 Carew Tower
441 Vine Street
Cincinnati OH 45202
Work Phone: (513) 763-1916
Fax: (513) 562-1760
Email: white.michaell@kendle.com