Paper B2B-15

Looking for an Event in a Haystack?

Maya Barton, Rho, Inc.; Rita Slater, Rho, Inc.

ABSTRACT

Looking for an Event in a Haystack?

How do you identify where a specific event occurred during a visit period? Or whether a recorded event actually occurred during a particular visit? Typically Concomitant Medications (CMs) and Adverse Events (AEs) form a long list of events, that is, a dataset with each event's description, start and end dates. Typically these events appear on visit forms as a data point marking the occurrence of the event. A common programming task involves displaying any inconsistencies in a report for the data manager's perusal, such that he or she can query the site in question and clean the data.

This paper details the generation of such a report by building a dataset of the inconsistencies. In examining these inconsistencies, four case report forms (CRFs) come into play: a physical visit form (VIST), a telephone visit form (TELE), a CM form (CM), and an AE form (AE). Through a series of sorts, formats, and joins, this paper explains how these CRFs can be reconciled with one another to form a presentable dataset of inconsistencies.

INTRODUCTION

The beauty of programming in SAS is simplifying and minimizing the large number of data points that a Clinical Data Manager (CDM) has to manually sort through to clean data. A typical trial depending on the complexity of the protocol could sport anywhere from 10 visits for a several hundred spanning over several years. A data point field is captured at each of the visits to indicate if there was an occurrence of a new Adverse Event (AE) after the last visit. Another typical data point field captured is a new Concomitant Medication (CM) started after the last visit. A relevant date field will also be captured.

Meanwhile during the span between visits the site will record on an ongoing basis capturing the details of any occurrences of AEs and CMs. The AEs, CMs, site visits and the telephone visits are all captured on separate CRFs, and typically imported into different SAS datasets.

The regularly scheduled data validation, identifying inconsistencies, querying the site and eventual data cleaning requires reconciling the information captured at the visits with the ongoing set of logs for Adverse Events and Concomitant Medications which could be as large as 10,000 records each depending on the protocol. The number of data points will be orders of magnitude as the number of records. At its simplest manual level, the Data Manager would use a dump listing of AEs and CMs ongoing logs datasets sorted start dates and visually compare the visit date to the start date. This paper shows how to minimize this laborious process and take some of the human error out of the data cleaning and validation process.

A reconciling listing can be generated that only highlights the cases where there is an inconsistency, cutting the number of data points to be checked by orders of magnitude.

RECONCILING STEPS TO CREATE A QUERY LISTING

The example in this paper will address the specific case of reconciling the individual visit dates with each subject's CMs start dates. The same logic can be applied to AEs as well.

Visits have characteristics. Typical examples that will be used in this paper are regular physical visit and telephone visit. Physical visit (VPH2) and a telephone visit (TELE) are set and sorted to get our first point of reference. We only keep the records for which the data point 'Has the participant experienced new or change in Concomitant Medication(s)?' has an affirmative value.

These data sets are ordered by visits and only have that particular visit date in each record. Since we are interested in an event that occurred between this visit and the previous visit, we build a dataset that is ordered by the visit dates irrespective of whether it was a physical visit or telephone visit and we also need to impute the previous visit date in the same observation.

The program in the beginning includes the assignment of all the libraries used. Typically these include: MASTER pointing to where all the clinical datasets reside. FORMAT pointing to where study specific formatting datasets reside. All other related libraries are defined such that a specific variable from a dataset is accessible from within the program steps.

If a common sub-setting is required which apply to all individual datasets in play then it is advisable to use a macro to bring the datasets into the working environment ready to be used by the program. In this particular example we use a record status to subset out any deleted records or test records. We also format a record number to make it generic across the datasets in play. It is also a good habit to format any other variables that will be used in merges later on in the program.

STEP 1:

```
  * Library names and file names-------------------------------------------------------;
             libname FORMAT      "&ROOT\Format"        ACCESS = READONLY;
             libname LIBRARY     "&ROOT\Format\"       ACCESS = READONLY;
             libname MASTER      "&ROOT\Data\Master"   ACCESS = READONLY;

   *Macro to get Clinical data to work with from the Masters library-----------------;
   %MACRO GetData( ds, keep=);
   DATA &ds;
        set master.&ds(keep=ID Recstat &keep) ;
        if recstat in ('X','T') then delete;
        recnumber = input(seqno, $3.) + 1;
        recnum = put(recnumber,2.);
        drop recnumber;
   RUN;

   PROC sort data = &ds;
        by ID Phase recnum;
   RUN;
   %MEND GetData;

   *Call macro with all the visit datasets to be pulled in---------------------------;
   %GetData( tele, keep= Site ID datastr Phase seqno NEWCM TELEDT);
   %GetData( vph2, keep= Site ID datastr phase seqno VISDT);

   *rename the variable to set the different visit datasets appropriately------------;
   proc sort data= tele(rename= (teledt=visitdate))
               out= teledates(keep= id datastr phase seqno visitdate);
               by id phase seqno visitdate;
               where newcm eq '1';
   run;

   proc sort data= vph2(rename= (visdt=visitdate))
               out= vph2dates(keep= id datastr phase seqno visitdate);
               by id phase seqno visitdate;
   run;

   Data all_visits;
        set teledates vph2dates;
   run;

   proc sort data= all_visits; by id phase visitdate datastr; run;
   proc sort data= tele out= teleNEWCM_IDs(keep= ID teledt); by id; where newcm eq
   '1'; run;


   Data all_visits;
        merge all_visits teleNEWCM_IDs (in= newcm) ;
        by id;
        if newcm;
        lag_date= lag(visitdate);
        lag_ds= lag(datastr);
        if first.id then do; previous_date = visitdate; previous_form= datastr; end;
        else if not (first.id) then do; previous_date= lag_date; previous_form=
   lag_ds; end;
        format previous_date date9.;
        drop lag_date;
        if visitdate eq previous_date then delete;
        if teledt ne visitdate then delete;
run;
```

In step 1 we bring all the visit datasets and keep only the relevant records. Now we have a clean dataset that has all the visits with variable 'NEWCM' checked for the field 'Has the participant experienced new or change in Concomitant Medication(s)?' We also have the previous visit's date which will enable us to validate that a particular CM with a start date between these two visit dates has occurred. We keep the previous form and date such that we can list this for the CDM to query and clean the data.

ID	PHASE	visitdate	previous_date	previous_form
AA1A11	9101	3/15/2008	9/17/2007	VPH2
AA1A12	3023	9/7/2010	4/9/2010	VPH2
AA1A14	3021	12/1/2010	8/17/2010	VPH2
AA1A15	9107	12/16/2009	10/2/2009	VPH2
AA1A1A	9022	2/13/2012	10/24/2011	VPH2
AA1AA1	9014	8/17/2010	6/11/2010	VPH2
AA1AA2	9020	2/21/2011	12/20/2010	VPH2
AA1AA4	9002	10/9/2009	1/28/2013	TELE
AA1AA8	9016	3/11/2011	11/22/2010	VPH2
AA2AA1	9014	9/20/2010	7/29/2010	VPH2
AA2AA4	5020	2/4/2013	10/8/2012	TELE
AA2AA5	9105	10/10/2007	6/11/2007	TELE
AA2AA8	9101	3/29/2009	12/18/2008	VPH2
AA2AAB	3013	10/23/2009	9/22/2009	VPH2
AA3AA1	9111	4/7/2010	1/26/2010	TELE
AA3AA2	9107	9/20/2009	9/30/2009	VPH2

Table1: Visits with Subject ID, current visit date, and previous visit date

STEP 2:

```
%GetData( cmed, keep= site ID datastr phase seqno CMVTM CMSTDT CMSTD CMSTM CMSTY
CMENDT CMEND CMENM CMENY);
proc sort data= cmed; by id cmstdt; run;

data cmed_dates(keep= id seqno cmstdt);
     merge teleNEWCM_IDs(in= newcm) cmed;
        by id;
        if newcm;
run;
```

In step 2 we bring in the CM dataset and extract the relevant records, basically selecting the records for just the subjects from the dataset built in step 1. A Sample of the dataset is shown below, since this can be a very large dataset.

ID	RECNUM	CMSTDT
AA1A11	000	5/18/2007
AA1A11	001	5/18/2007
AA1A11	007	5/18/2007
AA1A11	002	6/29/2007
AA1A11	003	6/29/2007
AA1A11	004	8/8/2007

Table 2: Concomitant Medications with Subject, record number, and start date.

STEP 3:

```
proc sql noprint feedback;

create table temp_dates as
  select cmed_dates.*, all_visits.previous_date, all_visits.visitdate
    from cmed_dates a left join all_visits b on b.id=a.id
   order by a.id;

create table yes_dates as
  select *
    from temp_dates
   where cmstdt between previous_date and visitdate
   order by id;

 quit;
```

In Step 3 we find subjects that have consistent data. We found the program more readable when implemented in PROC SQL. We left joined the CM dataset with the visit dataset for all the subjects, and then selected the records where the CM start date was between the visit date and previous date to create the tables shown below. A small subset has been shown as examples.

ID	RECNUM	CMSTDT	previous_date	visitdate
AA1A11	002	6/29/2007	9/17/2007	3/15/2008
AA1A11	001	5/18/2007	9/17/2007	3/15/2008
AA1A11	009	10/24/2007	9/17/2007	3/15/2008
AA1A11	003	6/29/2007	9/17/2007	3/15/2008
AA1A11	004	8/8/2007	9/17/2007	3/15/2008
AA1A11	008	10/24/2007	9/17/2007	3/15/2008
AA1A11	006	12/11/2007	9/17/2007	3/15/2008
AA1A11	000	5/18/2007	9/17/2007	3/15/2008
AA1A11	005	8/8/2007	9/17/2007	3/15/2008
AA1A11	007	5/18/2007	9/17/2007	3/15/2008
AA1A12	007	8/31/2010	4/9/2010	9/7/2010
AA1A12	001	10/10/2007	4/9/2010	9/7/2010
AA1A12	008	3/18/2011	4/9/2010	9/7/2010
AA1A12	005	9/24/2007	4/9/2010	9/7/2010
AA1A12	004	8/5/2008	4/9/2010	9/7/2010
AA1A12	000	10/10/2007	4/9/2010	9/7/2010
AA1A12	006	1/8/2010	4/9/2010	9/7/2010
AA1A12	002	10/10/2007	4/9/2010	9/7/2010
AA1A12	003	8/5/2008	4/9/2010	9/7/2010

Table 3: A join table of the Concomitant start dates, previous visit date, and current visit date by Subject ID.

ID	RECNUM	CMSTDT	previous_date	visitdate
AA1A11	009	10/24/2007	9/17/2007	3/15/2008
AA1A11	008	10/24/2007	9/17/2007	3/15/2008
AA1A11	006	12/11/2007	9/17/2007	3/15/2008
AA1A12	007	8/31/2010	4/9/2010	9/7/2010
AA1A14	011	11/21/2010	8/17/2010	12/1/2010
AA1A15	024	10/21/2009	10/2/2009	12/16/2009
AA1A1A	016	2/8/2012	10/24/2011	2/13/2012
AA1AA1	022	8/16/2010	6/11/2010	8/17/2010
AA1AA1	021	8/15/2010	6/11/2010	8/17/2010
AA1AA1	014	6/11/2010	6/11/2010	8/17/2010
AA1AA2	029	2/21/2011	12/20/2010	2/21/2011
AA1AA2	028	2/20/2011	12/20/2010	2/21/2011
AA1AA2	027	2/20/2011	12/20/2010	2/21/2011
AA1AA2	026	2/20/2011	12/20/2010	2/21/2011
AA1AA2	023	2/19/2011	12/20/2010	2/21/2011
AA1AA2	024	2/19/2011	12/20/2010	2/21/2011
AA1AA2	022	2/18/2011	12/20/2010	2/21/2011
AA1AA2	021	2/18/2011	12/20/2010	2/21/2011
AA1AA2	020	2/18/2011	12/20/2010	2/21/2011
AA1AA2	025	2/19/2011	12/20/2010	2/21/2011

Table 4: Subject IDs with Concomitant start dates between the previous visit date, and current visit date.

STEP 4:

```
proc sort nodupkey data= yes_dates
out= yes_dates1(drop= seqno); by ID previous_date visitdate;
run;

proc sort data= all_visits out= chk_dates; by id previous_date visitdate; run;

data no_dates;
       merge yes_dates1 (in= problem) chk_dates;
       if not problem;
       by id;
       run;
```

In Step 4 we use the data that we have for subjects that have consistent data to find the problem subjects with inconsistent data.

ID	previous_date	visitdate	PHASE	previous_form
AA1AA8	11/22/2010	3/11/2011	9016	VPH2
AA2AA4	10/8/2012	2/4/2013	5020	TELE
AA2AAB	9/22/2009	10/23/2009	3013	VPH2
AA3AA2	9/30/2009	9/20/2009	9107	VPH2

Table 5: Problem Subject IDs identified.

STEP 5:

```
* Bring in CM records for subjects with inconsistent data----------------------------;

proc sort nodupkey data= no_dates out= prob_IDs(keep= ID); by id; run;
 proc sort data= cmed
 out= cmed_temp(keep= id recnum site cmvtm cmstdt cmendt);
 by id  recnum;
 run;

data CMED_info;
        merge prob_IDs (in=want) cmed_temp;
         if want;
        by id;
        run;


* Format numbers to CRF Form names and Visit names-----------------------------------;

data final_1;
     length visit $40;
        set no_dates;
        Form1    = put(strip(datastr), $dsfmt.) || " (" || strip(datastr) || ")";
        Form2    = put(strip(previous_form), $dsfmt.) || " (" || strip(previous_form)
|| ")";
        run;

proc sort data= final_1 out=final_2; by id phase previous_date visitdate; run;

data CMED_info; set CMED_info; by id recnum; if first.id then rec=0; rec+1; run;
data final_2; set final_2; by id phase previous_date visitdate; if first.id then
rec=0; rec+1; run;

data final(keep= rec id visit form1 visitdate form2 previous_date recnum cmvtm cmstdt
cmendt);
        merge final_2 CMED_info;
        by id rec;
        run;
```

In Step 5 we identify these subjects and build a dataset of CMs for just the problem subjects. At this point we can just list the subjects that have the relevant variable 'NEWCM' checked along with its visit dataset information which is entirely adequate for the CDM to query the site and get the data cleaned. However at this point CDMs would like to be more proactive and check if there are CMs that could fit the situation and suggest that there may have been a typo or an obvious mistake. This step provides a more comprehensive listing for such a check. A final pretty listing provided to the CDM for their data cleaning purposes looks like the following.

ID	Form1	Form2	Previous Visit Date	Current Visit Date	rec	CMVTM	CMSTDT	CMENDT
AA1AA8	Telephone Consultation	Vital Signs	11/22/2010	3/11/2011	1	VALCYTE	11/29/2004	11/20/2006
AA1AA8					2	FENTANYL	11/15/2006	11/15/2006
AA1AA8					3	VERSED	11/15/2006	11/15/2006
AA1AA8					4	BENADRYL	11/15/2006	11/15/2006
AA1AA8					5	TYLENOL	12/18/2006	12/18/2006
AA1AA8					6	ADVIL	2/12/2007	2/16/2007
AA1AA8					7	ADVAIR	2/7/2007	2/12/2007
AA1AA8					8	ALBUTEROL	2/7/2007	2/12/2007
AA1AA8					9	VALCYTE	2/16/2007	5/21/2008

Table 6: Final dataset.

CONCLUSION

In conclusion this paper represents one way you can use SAS to take a very labor intensive process and streamline it to a more efficient and robust method for data cleaning. This process can be programmed into a generic macro to identify any event corresponding to a haystack of events. This paper shows the specific example using dates as events, but the method can be applied to other values such as for Adverse Events and Protocol Deviations. The final dataset can be printed to a pretty report with headings, page numbers, footnotes, and colors using Proc report. The reports are typically sent to sites such that the site monitors can enter relevant missing data or correct existing data. For internal Data Management purposes the final dataset can be exported to spreadsheets which can then be used to query the sites and checked off as data cleaning issues that have been addressed. Shown below is an example of a final report.

Study Name
Protocol: SESUG_2013

Page 1 of 8

Rho, Inc.
If [Has the participant experienced new or change in concomitant medication?] is yes
then a concomitant medication should be present with start date between the previous visit date and the current date of contact

Visits with NEWCM=1 but no corresponding CMEDSTDT						CONCOMITANT MEDS			
ID	VISIT	FORM	CURRENT VISIT DATE	PREVIOUS FORM	PREVIOUS VISIT DATE	LINE	MEDICATION	START DATE	END DATE
AA1AA8	M16 (9016)	Telephone Consultation (TELE)	11MAR2011	Vital Signs and Physical Exam (VPH2)	22NOV2010	1	VALCYTE	29NOV2004	20NOV2006
			.		.	2	FENTANYL	15NOV2006	15NOV2006
			.		.	3	VERSED	15NOV2006	15NOV2006
			.		.	4	BENADRYL	15NOV2006	15NOV2006
			.		.	5	TYLENOL	18DEC2006	18DEC2006
			.		.	6	ADVIL	12FEB2007	16FEB2007
			.		.	7	ADVAIR	07FEB2007	12FEB2007
			.		.	8	ALBUTEROL	07FEB2007	12FEB2007
			.		.	9	VALCYTE	16FEB2007	21MAY2008
			.		.	10	AVEENO HYDROCORTISONE	06MAR2007	17MAR2007
			.		.	11	ADVIL	14FEB2007	04JAN2008
			.		.	12	ALBUTEROL	02APR2008	.
			.		.	13	ADVAIR	02APR2008	.
			.		.	14	Zyrtec	13MAY2009	.
			.		.	15	Fentanyl	17NOV2009	17NOV2009

*****Confidential preliminary data. Based on data available in the clinical database as of 15AUG2013*****

ACKNOWLEDGMENTS

The authors acknowledge support and guidance of this Data Cleaning Process to:
LaSonia Morgan, Senior Clinical Data Project Manager, Rho, Inc.

CONTACT INFORMATION

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

Name:	Maya Barton, Rita Slater
Enterprise:	RHO, INC.
Address:	Rho Building, 6330 Quadrangle Drive,
City, State ZIP:	Chapel Hill, North Carolina, 27517
Work Phone:	(919) 408-8000
Fax:	(919) 408-0999
E-mail:	maya_barton@rhoworld.com rita_slater@rhoworld.com