ABSTRACT

IN THE REGULATORY WORLD OF PATIENT SAFETY AND PHARMACOVIGILANCE, WHETHER IT’S DURING CLINICAL TRIALS OR POST-MARKET SURVEILLANCE, SAES THAT AFFECT PARTICIPANTS MUST BE COLLECTED, AND IF CERTAIN CRITERIA ARE MET, REPORTED TO THE FDA AND OTHER REGULATORY AUTHORITIES. SAES ARE OFTEN ENTERED INTO MULTIPLE DATABASES BY VARIOUS USERS, RESULTING IN POSSIBLE DATA DISCREPANCIES AND QUALITY LOSS. EFFORTS HAVE BEEN MADE TO RECONCILE THE SAE DATA BETWEEN DATABASES, BUT THERE IS NO INDUSTRIAL STANDARD REGARDING THE METHODOLOGY OR TOOL EMPLOYED FOR THIS TASK. SOME ORGANIZATIONS STILL RECONCILE THE DATA MANUALLY, WITH VISUAL INSPECTIONS AND VOCAL VERIFICATION. NOT ONLY IS THIS LABORIOUS AND ERROR-PRONE, IT BECOMES PROHIBITIVE WHEN THE DATA REACH HUNDREDS OF RECORDS. WE DEVISED AN EFFICIENT ALGORITHM USING SAS® TO COMPARE TWO DATA SOURCES AUTOMATICALLY. OUR ALGORITHM IDENTIFIES MATCHED, DISCREPANT, AND UNPAIRED SAE RECORDS. ADDITIONALLY, IT EMPLOYS A USER-SUPPLIED LIST OF SYNONYMS TO FIND NON-IDENTICAL BUT RELEVANT MATCHES. FIRST, TWO DATA SOURCES ARE COMBINED AND SORTED BY KEY FIELDS SUCH AS “SUBJECT ID”, “ONSET DATE”, “STOP DATE”, AND “EVENT TERM”. RECORD COUNTS AND LEVENSHTEIN EDIT DISTANCES ARE CALCULATED WITHIN CERTAIN GROUPS TO ASSIST WITH SORTING AND MATCHING. THIS COMBINED RECORD LIST IS THEN FED INTO A DATA STEP TO DECIDE WHETHER A RECORD IS PAIRED OR UNPAIRED. FOR AN UNPAIRED RECORD, A STUB RECORD WITH ALL FIELDS SET AS “?” IS GENERATED AS A MATCHING PLACEHOLDER. EACH RECORD IS WRITTEN TO ONE OF TWO DATA SETS. LATER, THE DATA SETS ARE TAGGED AND PULLED INTO A COMPARISON LOGIC USING HASH OBJECTS, WHICH ENABLE FIELD-BY-FIELD COMPARISON AND DISPLAY DISCREPANCIES IN CLEAN FORMAT FOR EACH FIELD. IDENTICAL FIELDS OR COLUMNS ARE CLEARED OR REMOVED FOR CLARITY. THE RESULT IS A STREAMLINED AND USER-FRIENDLY PROCESS THAT ALLOWS FOR FAST AND EASY SAE RECONCILIATION.

INTRODUCTION

If you are a Patient Safety (PS) or Pharmacovigilance (PV) manager in charge of a safety database for clinical trials, you are likely to encounter a routine regulatory compliance task called SAE reconciliation. During a clinical trial, Adverse Events (AEs) are usually logged into a clinical database through Case Record Form (CRF) by the trial site personnel. In the meantime, Serious Adverse Events (SAEs) are reported to sponsor (or delegate such as Clinical Research Organization (CRO)) for Pharmacovigilance monitoring purposes. The SAEs are entered into a safety database by the PV specialists and assessed for reportability to regulatory bodies. So as the clinical trial proceeds, two copies of SAEs records are formed and stored in two different databases – clinical database and safety database, and it’s your responsibility to make sure the information contained in the different SAE record sets are complete, consistent and comply with all regulatory requirements.

Depending on your organization’s size, you may have various support for this activity: companies with lots of resources may have software programmers write code to compare the different SAE data sets, medium size firms may use an off-the-shelf Excel spreadsheet comparator for the job, and yet companies tight on budget may just ask their PV staff (who are usually medical doctors or trained pharmacists) to visually inspect the different data printed on paper and verify by reading them out to others!
There is no industrial standard for helping with this. I wrote a stored procedure in SAS to automate this task for the PV staff. This is designed to be used by the end user without any technical assistance from the data analysts.

**PROBLEM**

The following are some sample safety database and clinical database SAE records to be reconciled for the same clinical trial. For privacy purpose, all essential data are replaced with tokens.

Figure 1 is a partial listing of SAE data from the safety database.

Figure 2 is a partial listing of SAE data from the clinical database.

The columns / fields of the tables have been normalized to match the same from both data sources for comparison purpose.

<table>
<thead>
<tr>
<th>SID</th>
<th>Event</th>
<th>Onset Date</th>
<th>End Date</th>
<th>Outcome</th>
<th>Causality</th>
<th>LLT</th>
<th>PT</th>
<th>SOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>N001</td>
<td>Exa</td>
<td>8/30/2014</td>
<td>8/31/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>EXA</td>
<td>EXAATT</td>
<td>ENO</td>
</tr>
<tr>
<td>N002</td>
<td>xyz</td>
<td>10/7/2014</td>
<td>10/16/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>XYZTT</td>
<td>XYZ</td>
<td>II</td>
</tr>
<tr>
<td>N002</td>
<td>K</td>
<td>11/19/2014</td>
<td>11/26/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>K</td>
<td>M, MU</td>
<td>IAI</td>
</tr>
<tr>
<td>N004</td>
<td>HT BR</td>
<td>10/27/2014</td>
<td></td>
<td>Recovering</td>
<td>Not Related</td>
<td>BRON</td>
<td>BRON</td>
<td>SKTM</td>
</tr>
<tr>
<td>N005</td>
<td>MD, 39C</td>
<td>11/21/2014</td>
<td>11/22/2014</td>
<td>Recovered</td>
<td>Related</td>
<td>PY</td>
<td>FE</td>
<td>NSD</td>
</tr>
</tbody>
</table>

**Figure 1. Sample Data from Safety Database**

<table>
<thead>
<tr>
<th>SID</th>
<th>Event</th>
<th>Onset Date</th>
<th>End Date</th>
<th>Outcome</th>
<th>Causality</th>
<th>LLT</th>
<th>PT</th>
<th>SOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>N004</td>
<td>HTBR</td>
<td>27-Oct-2014</td>
<td>4-Nov-2014</td>
<td>Recovering</td>
<td>Not Related</td>
<td>BRON</td>
<td>BRON</td>
<td>SKTM</td>
</tr>
<tr>
<td>N005</td>
<td>MD, 39C</td>
<td>21-Nov-2014</td>
<td>22-Nov-2014</td>
<td>Recovered</td>
<td>Related</td>
<td>PY</td>
<td>FE</td>
<td>NSD</td>
</tr>
</tbody>
</table>

**Figure 2. Sample Data from Clinical Database**

As you can see, there are quite a few discrepancies between the two listings. It’s becoming especially difficult for human eye to discern and organize the discrepancies if there are missing / extra records.

**SOLUTION**

The complexity of the problem calls for a systemic approach for this job. The SAS stored procedure that was written uses a combination of MACRO, DATA STEP and PROC SQL to process the above illustrated SAE data sources, and output the results in a very clean format for users.

Figure 3 is the output from the program that lists the discrepancies of the two SAE data sources. Every row accommodates two text lines for easy visual comparison, one each from the two SAE data sources. Only the discrepant values of the column / fields are printed for clarity purpose. For missing or extra records, all fields are printed with ‘?’ as placeholders.

<table>
<thead>
<tr>
<th>Source</th>
<th>SID</th>
<th>Event</th>
<th>Onset Date</th>
<th>End Date</th>
<th>Outcome</th>
<th>Causality</th>
<th>LLT</th>
<th>PT</th>
<th>SOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAE</td>
<td>N001</td>
<td>Exa</td>
<td>8/30/2014</td>
<td>8/31/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>EXA</td>
<td>EXAATT</td>
<td>ENO</td>
</tr>
<tr>
<td>DCE</td>
<td>N002</td>
<td>xyz</td>
<td>10/30/2014</td>
<td>10/31/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>XYZTT</td>
<td>XYZ</td>
<td>II</td>
</tr>
<tr>
<td>DCE</td>
<td>N004</td>
<td>HT BR</td>
<td>10/27/2014</td>
<td>11/04/2014</td>
<td>Recovered</td>
<td>Not Related</td>
<td>XYZTT</td>
<td>XYZ</td>
<td>IAI</td>
</tr>
</tbody>
</table>

**Figure 3. Output Table that Shows the Discrepancies**
Figure 4 lists the two SAE datasets intertwined with each paired records, including the placeholders that are comprised of character ‘?’ . This is the ultimate reference for the end user which instructs them what action is needed to correct the discrepancies.

![Figure 4. Detailed Line by Line Merged listing of the Two SAE Data Discrepancies](image)

**SOURCE CODE**

```
/*Normalize input SAE data and put them in two tables DAE and DCE*/
/*...*/
/*include a variable 'Source' to indicate whether the data is from DAE or DCE*/

/*Mix the two datasets*/
PROC SQL;
  create table raw_combined as
  select *, count(*) as mgroup from
  (select * from dae
    union
    select * from dce)
  group by upcase(SID), 'Onset Date'n, 'End Date'n, upcase(Event)
  order by upcase(SID), 'Onset Date'n, 'End Date'n, upcase(Event),
  Outcome, Causality, LLT, PT, SOC, Source;
QUIT;
/*Calculate LEVENSHTEIN on concatenated fields*/
DATA icy_combined (drop=gfirst);
  set raw_combined;
  by SID;
  format dspedis 8. gfirst $255.;
  retain gfirst;
  if first.SID then gfirst = cats('Onset Date'n, 'End Date'n, Event);
  dspedis = complev(cats('Onset Date'n, 'End Date'n, Event), gfirst, 'il');
RUN;
/*Order by LEVENSHTEIN edit distance values*/
PROC SQL;
  create table pre_combined as select * from icy_combined
  order by upcase(SID), dspedis, 'Onset Date'n, 'End Date'n, upcase(Event), Source
desc, Outcome, Causality, LLT, PT, SOC;
QUIT;
/*Macros for field data swaps*/
%let ClearVariables = %str(
  Source = '?';
  SID = '?';
```
'Onset Date'n = '?';
'End Date'n = '?';
Event = '?';
Outcome = '?';
Causality = '?';
LLT = '?';
PT = '?';
SOC = '?';
);

%let
SaveVariables = %str(
  _cn = Source;
  _si = SID;
  _so = 'Onset Date'n;
  _ss = 'End Date'n;
  _et = Event;
  _oc = Outcome;
  _rc = Causality;
  _llt = LLT;
  _pt = PT;
  _soc = SOC;
);

%let
RestoreVariables = %str(
  Source = _cn;
  SID = _si;
  'Onset Date'n = _so;
  'End Date'n = _ss;
  Event = _et;
  Outcome = _oc;
  Causality = _rc;
  LLT = _llt;
  PT = _pt;
  SOC = _soc;
);

/*Navigate and split the list with paired records*/
DATA dae1 (drop=i cid pid gid _cn--_soc)
dce1 (drop=i cid pid gid _cn--_soc);
set pre_combined end=me;
format cid pid $255.;
retain i 1 cid pid gid;

if mod(i, 2) - 1 then
  do;
    if Source = 'DCE' then
      output dce1;
    else output dae1;

    if me then
      do;
        _cn = Source;
        &ClearVariables;
        if _cn = 'DCE' then
          output dae1;
        else output dce1;
      end;
    else
      do;
        cid = Source;
        pid = SID;
        gid = mgroup;
        i + 1;
      end;
  end;

/*even row*/
else if pid ne SID or (cid ne '' and Source ne '') or (cid = '' and Source = '') then do;
  &SaveVariables;
  &ClearVariables;

  if cid = 'DCE' then

output dael;
else output dcel;
&RestoreVariables;

/* starting with new odd line */
if Source = 'DCE' then
  output dcel;
else output dael;

if me then
do;
&ClearVariables;
  if _cn = 'DCE' then
    output dael;
  else output dcel;
end;
else
do;
  cid = Source;
  pid = SID;
  gid = mgroup;
end;
else
do;
  /* same id (more than 2) but different dates/et */
  if gid ne mgroup then
do;
    &SaveVariables;
    &ClearVariables;
      if cid = 'DCE' then
        output dael;
      else output dcel;
    &RestoreVariables;
    /* starting with new odd line */
    if Source = 'DCE' then
      output dcel;
    else output dael;
    if me then
do;
      &ClearVariables;
        if _cn = 'DCE' then
          output dael;
        else output dcel;
      end;
    else
do;
      cid = Source;
      pid = SID;
      gid = mgroup;
    end;
  else
do;
    if Source = 'DCE' then
      output dcel;
    else output dael;
    cid = Source;
    pid = SID;
    gid = mgroup;
    i + 1;
  end;
end;

RUN;

/* getting ready for comparison by adding different prefix to variables*/
%MACRO vars(dsn, out, prefix);
  %let list=;
  %let type=;
  %let dsid=%sysfunc(open(&dsn));
  %let cnt=%sysfunc(attrn(&dsid,nvars));
  %do i = 1 %to &cnt;
    %let list=&list:%sysfunc(varname(&dsid,&i));
    %let type=%sysfunc(vartype(&dsid,&i));
  %end;
  %let rc=%sysfunc(close(&dsid));

  /*Formatting code causes damages here!*/
  data &out(drop=);
    %do i = 1 %to &cnt;
      %let temp=%scan(&list,&i,:);
      "&temp"n
    %end;
  set &dsn;
  %do i = 1 %to &cnt;
    %let temp=%scan(&list,&i,:);
    "&prefix&temp"n="&temp"n;
  %end;
  run;
%MEND vars;

%vars(dae1, dae2, dae_);
%vars(dce1, dce2, dce_);

DATA frame;
  format Source SID Event 'Onset Date'n 'End Date'n Outcome Causality LLT PT SOC $255.
  set dae1;
  _OBS_ = put(_n_, 8.);
RUN;

DATA dae2;
  set dae2;
  dae__OBS_ = put(_n_, 8.);
RUN;

DATA dce2;
  set dce2;
  DCE__OBS_ = put(_n_, 8.);
RUN;

/*Compare field by field*/
DATA diff (drop= dae_Source -- dae_OBS_ DCE_Source -- DCE_OBS_ i x y flag) mtag
  (keep=flag);
  set frame;
  if _n_=1 then do;
    set dae2;
    set dce2;
    declare hash hdae(dataset: 'dae2');
    hdae.definekey('dae_OBS_');
    hdae.definedata(all: 'yes');
    hdae.definedone();
    declare hash hDCE(dataset: 'dce2');
    hDCE.definekey('dce_OBS_');
    hDCE.definedata(all: 'yes');
    hDCE.definedone();
  end;
  hdae.find(key:_OBS_);
  hDCE.find(key:_OBS_);
  array ma{*} Source -- SOC;
flag = ''; 

do i = 1 to dim(ma);
    x = strip(vvalue("dae_" || vname(ma[i])));
    if x = '' or x = '.' or x = 'A0'x then 
        x = ifc(i = 1, 'NA', 'MISSING');
    y = strip(vvalue("dce_" || vname(ma[i])));
    if y = '' or y = '.' or y = 'A0'x then y = ifc(i = 1, 'NA', 'MISSING');
    if i in (1, 2) then ma[i] = x || '0A'x || y;
    else if lowcase(x) = lowcase(y) or (i = 6 and
        /*These values meant the same thing*/
        lowcase(x) in ('resolved without sequela', 'recovered/resolved', 'recovered / resolved') and
        lowcase(y) in ('resolved without sequela', 'recovered/resolved', 'recovered / resolved')
        /*More synonyms can be added here ...*/
        then ma[i] = '';
    else
        do;
            ma[i] = x || '0A'x || y;
            flag = '*';
        end;
    end;
if flag ne '' then output diff;
output mtag;
RUN;

DATA combined;
    set mtag;
    set dael;
    output;
    set dcel;
    output;
RUN;

/* Find and drop columns that don't have discrepancies */
/*...*/
/* Output to Excel */
/*...*/

CONCLUSION
This SAS stored procedure provided the PV staff an easy to use tool with clean output of the SAE discrepancies between two different data sources. It achieved very good results – accuracy improved from about 70% to 100%, productivity soared about 80 fold! The algorithm is efficient, able to handle most of the scenarios in data discrepancy, and there is no reason it can’t be used elsewhere for other types of data.

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

<Tom Zhou Hui>
<tomzhouhui@yahoo.com>

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.