```
*******************************************************************************
**************
*             CLIENT:   BigPharma
*           PROTOCOL:   CDISCV2
*       PROGRAM NAME:   REVCO.SAS
*        SAS VERSION:   SAS V9 (Windows) 9.1.3 and 9.2, and 9.3
*            PURPOSE:   Merges CO back onto the specified domain dataset and
removes merged records from CO
*
*        USAGE NOTES:
*         PARAMETERS:   LIBIN:  Input libname
*                       LIBOUT: Output libname, defaults to WORK
*                       DS:     Base SDTM domain
*                       OUTDS:  Output dataset name, defaults to input dataset name
*                       COMDS:  Comments dataset name, defaults to CO
*                       DELCO:  Delete comments that have been merged to DS from
CO.  Defaulted to Y (Y,N)
*                       MOVCO:  Move comments dataset to libout if libout <> libin.
 Defaulted to Y (Y,N)
*
*             AUTHOR:   (b) (6)
*       DATE CREATED:   09/24/2009
*
*       DEPENDENCIES:
*   MODIFICATION LOG:
*******************************************************************************
**************
* DATE         BY          DESCRIPTION
*******************************************************************************
**************
*  19AUG2009  (b) (6)       Copied and modified REVSUPP Macro written by  (b) (6)
*  03JUN2011  (b) (6)       Fixed library issue when CO library different from
Libin.  Cleaned up log messages.
*  18MAY2016  (b) (6)       Handled bug where it was trying to test  &&&lds._label
for null.
*******************************************************************************
**************
* © 2016 Pharmaceutical Product Development','Inc.
* All Rights Reserved.
*******************************************************************************
**************;


%macro REVCO(libin=,libout=WORK,ds=,outds=,comds=CO,delco=Y,movco=Y) / STORE
des="V1.2";
%if &outds eq  %then %let outds = &ds;
%let id = &sysmacroname;
%let libin = %upcase(&libin);
%let ds = %upcase(&ds);
%let co = %str();
```

```
%let libout= %upcase(&libout);
%let libco = %str();
%let libco1 = %str();
%let outds =%upcase(&outds);
%let err = E%str(RROR: );
%let alrt = ALERT%str(_I);
%let sortedby = %str();
%let LABEL = %str();
%let COFND = 1;
%let cntqnam = 0;
%let cntcol = 0;
%put &id begin;
%macro DsSortLabel(base=,set=,out=,doWhat=write) /* / STORE des="V1.1" */ ;
    %let lds = %scan(%upcase(&base),-1,'.');

    %let &&lds._ds = &lds;
    %*put lds = &&&lds._ds.;
    %if &doWhat ne write %then %do;
        %let dsid = %sysfunc(open(&base,is));
        %if &DSID = 0 %then %put %sysfunc(sysmsg());
        %let &&lds._sortedby = %sysfunc(attrc(&dsid,SORTEDBY));
                /* KB - %STR( ) ENSURES AT LEAST A BLANK FOR COMPATIBILITY WITH THE
PROCESS AS DESIGNED */
        %let &&lds._label = %trim(%left(%sysfunc(attrc(&dsid,LABEL))))%str( );
        %let rc = %sysfunc(close(&dsid));
        %*put &lds._label = &&&lds._label;
        %*put &lds._sortedby = &&&lds._sortedby ;
    %end; %else %do;
        %if &&&lds._sortedby. ne %str() %then %do;
         %put ALERT_I: Applying Label and Sortedby info to %left(&set);
         proc sort data= &set out= &out(label="&&&lds._label");
           by &&&lds._sortedby;
         run;
        %end; %else %do;
         %put ALERT_I: Applying Label to %left(&set);
          data &out(label="&&&lds._label");
            set &set;
          run;
        %end;
    %end;
    %*put &lds._label = &&&lds._label;
    %*put &lds._sortedby = &&&lds._sortedby ;
%mend;

/* Ensure libraries and datasets exist */
 %if %sysfunc(exist(&libin..&ds,DATA)) ne 1 %then %do;
   %put &err. %left(&libin..&ds) is invalid. Make sure LIBIN and DS both exist and
are correct.;
   %goto ERR;
%end;
```

```
%* Check to see if the comment libref and comments dataset exist and is valid;
%if %index(&comds,.) = 0 %then %do;
     %put &alrt.: Libref not stated for %left(&comds) assuming same as libin.;
     %if %sysfunc(exist(&libin..&comds,DATA)) ne 1 %then %do;
       %put &alrt.: The %left(&comds) dataset does not yet exist.;
       %let COFND = 0;
    %end; %else %do;
        %let co = %upcase(&comds);
        %let libco = &libin;
    %end;
%end;  %else %do;
     %if %sysfunc(exist(&comds,DATA)) ne 1 %then %do;
       %put &alrt.: The %left(&comds) dataset does not yet exist.;
       %let COFND = 0;
    %end; %else %do;
        %let co = %scan(%upcase(&comds),-1,'.');
        %let libco = %scan(%upcase(&comds),1,'.');
    %end;
%end;
%*put co = &co;
%*put libco = &libco;
%let libco1 = &libco;
%global &ds._label &ds._sortedby &co._label &co._sortedby ;

%DsSortLabel(base=&libin..&ds,set=&libin..&ds,out=&libin..&ds,doWhat=);

%if %sysfunc(libref(&libout)) ne 0 %then %do;
    %put &err. LIB=&libout is not assigned;
    %goto ERR;
%end;

 %if %sysfunc(exist(&libout..&outds,DATA)) ne 0 %then %do;
   %put &alrt.: The %left(&libout..&outds) already exists and will be overwritten;
%end;


%if &COFND ne 0 %then %do;

%DsSortLabel(base=&libco..&co,set=&libco..&co,out=&libco..&co,doWhat=);

   %if %upcase(&movco) = Y and &libout ne &libco and
%sysfunc(exist(&libco..&co,DATA)) = 1 %then %do;
       %if %sysfunc(exist(&libout..&co,DATA)) ne 0 %then %do;
           %put &alrt.: The %left(&libout..&co) dataset already exists and will be
overwritten;
       %end;
       proc datasets;
           copy in=&libco out =  &libout.;
```

```
          select &co. ;
       quit;run;
       %let libco1 = &libout.;
   %end;

   /* Find out if there is an ID variable or not */
   proc sql noprint;
     select count(*) into: idpres
       from &libco1..&CO where idvar ne '' and
            upcase(rdomain) = upcase("&ds") and
            COREF is not null; /*macro can only be used for those studies that
assign COREF with the original Variable Name */
     select usubjid,coref,count(coref),idvar,idvarval into:
subj,:coref,:cntcor,:idvr,:idvarvl
       from (select upcase(left(trim(coref))) as coref, usubjid,idvar,idvarval
              from &libco1..&CO where upcase(rdomain) = upcase("&ds")) as a
      group by usubjid, coref,idvar,idvarval
     having count(coref)>1;
   quit;
   %*put sqlobs ==== &sqlobs;
   %if &sqlobs ^= 0 %then %do;
        %put &err. Duplicate Comment Column Reference (%left(&coref.)) values
associated with Subject "%left(&subj.)" and %left(&idvr.) = "%left(&idvarvl.)" in
%left(&co.) database for the %left(&ds.) Domain.;
        %goto ERR;
   %end;
   proc sql noprint;
        select a.usubjid,a.idvar,a.idvarval, a.coref,count(a.coref) as cnt into:
subj1,:idvr1,:idvarvl1,:coref1,:cntcoref1
        from &libco1..&CO as a,
            (select usubjid,count(idvar),idvar,idvarval
               from &libco1..&CO where upcase(rdomain) = upcase("&ds")
              group by usubjid, idvar,idvarval
             having count(idvar)>1) as b
        where a.usubjid=b.usubjid
          and a.idvar=b.idvar
          and a.idvarval=b.idvarval
        group by a.usubjid,a.idvar,a.idvarval, a.coref
        having count(coref)>1;
   quit;
   %*put sqlobs ==== &sqlobs;
   %if &sqlobs ^= 0 %then %do;
        %put &err. There are more than 1 %left(&idvr1.) = "%left(&idvarvl1.)"
values associated with Subject "%left(&subj1.)" in %left(&co.) database for the
%left(&ds.) Domain.;
        %goto ERR;
   %end;
   %if &idpres = 0 %then %do;
         %let idcnt = 0; %let id1 = ;
         %PUT &alrt.: THERE ARE NO COMMENT RECORDS FOR THIS DOMAIN;
```

```
         data &libout..&outds;
            set &libin..&ds;
         run;
         %DsSortLabel(base=&libin..&ds,set=&libin..&ds,out=&libout..&outds);
   %end; %else %do;
       /* Determain how many ID vars the Comments file has */
       proc sql noprint;
         select count(distinct idvar) into: idcnt
         from &libco1..&CO where upcase(rdomain) = upcase("&ds") and idvar is not
null;
         %let idcnt = %trim(%left(&idcnt));
         select distinct upcase(idvar) into: id1 -: id&idcnt
         from &libco1..&CO where upcase(rdomain) = upcase("&ds") and idvar is not
null;
       /* Determain how many COREV vars the Comments file has */
         select count(distinct COREF) into: cntCOREF
         from &libco1..&CO where upcase(rdomain) = upcase("&ds") and COREF is not
null;
         %let cntCOREF = %trim(%left(&cntCOREF));
         select distinct upcase(COREF) into: COREF1 -: COREF&cntCOREF
         from &libco1..&CO where upcase(rdomain) = upcase("&ds") and COREF is not
null;
       quit;
       %*put idcnt=&idcnt;

   %end;
   %do i = 1 %to &idcnt;
      %*put id=&&id&i..;
      %if &&id&i.. ne  %then %do;
         /* Find the length and type of the ID variable in the original DS */
         proc sql noprint;
           select length, type,label into: idlen, :idtype, :idlabel
           from dictionary.columns
           where upcase(libname) = upcase("&libin") and
                 upcase(memname) = upcase("&ds") and
                 upcase(name) = upcase("&&id&i");
           select length, type into: usubjlen, :usubjtype  /*make sure usubjid var
has same attributes as */
           from dictionary.columns                          /*domain dataset  (b) (6)
20AUG2009*/
           where upcase(libname) = upcase("&libin") and
                 upcase(memname) = upcase("&ds") and
                 upcase(name) = upcase("usubjid");
           /*CDISC spec states COVAL can be CVAL - CVALn so there can be more than
one comment columns*/
           /*find out how many here*/
           %let cntcol = 0;
           select count(name) into :cntcol
           from dictionary.columns
           where upcase(libname) = upcase("&libco1") and
```

```
                upcase(memname) = upcase("&co") and
                upcase(name) like ('COVAL___');
          %let cntcol = %trim(%left(&cntcol));
             create table __TMPSRT as
                 select distinct name,input(srt,best.) as srtn
                 from (
                 select name,case srt when '' then '0' else srt end as srt
                 from (
                    select distinct upcase(name) as name,

left(trim(translate(upcase(name),'','ABCDEFGHIJKLMNOPQRSTUVWRXYZ'))) as srt
                     from dictionary.columns
                     where upcase(libname) = upcase("&libco1") and
                           upcase(memname) = upcase("&co") and
                           upcase(name) like ('COVAL___')))
                    order srtn
                  ;
             select name into: CVAL1 -: CVAL&cntcol
             from __TMPSRT;

         /* Create a dataset that contains a single ID value */
           create table __TOTRAN as
           select  %if %upcase(&usubjtype) = NUM %then %do;
                       input(usubjid,best.) as usubjid %end;
                     %else %do;
                       usubjid length=%trim(%left(&usubjlen.)) as usubjid  %end;,
                    %if %upcase(&idtype) = NUM %then %do;
                      input(idvarval,best.) as &&id&i label="&idlabel" %end;
                     %else %do;
                      idvarval length=%trim(%left(&idlen.)) as &&id&i
label="&idlabel" %end;
                    %do z =  1 %to &cntCOREF;
                      %do y = 1 %to &cntcol;
                           ,
                           %if &y = 1 %then %do;
                           &&CVAL&y as &&COREF&z.. label = ""
                           %end; %else %do;
                           &&CVAL&y as &&COREF&z..%eval(&y-1) label = ""
                           %end;
                       %end;
                    %end;
                  ,coref
           from &libco1..&CO
           where upcase(rdomain) = upcase("&ds") and
                 upcase(idvar) = "&&id&i"
           order by usubjid, &&id&i.. ,coref;
         quit;
         data __TOTRAN;
            set __TOTRAN;
            %do z =  1 %to &cntCOREF;
```

```
               %do y = 1 %to &cntcol;
                    %if &y = 1 %then %do;
                       if upcase(coref) ne "&&COREF&z.." then &&COREF&z.. = '';
                        if length("&&COREF&z..") > 8 and &i = 1  then do;
                           put "ALERT_I: Comment variable name &&COREF&z.. greater
than 8 chars";
                        end;
                    %end; %else %do;
                       if upcase(coref) ne "&&COREF&z.." then
&&COREF&z..%eval(&y-1) = '';
                        if length("&&COREF&z..%eval(&y-1)") > 8 and &i = 1 then do;
                           put "ALERT_I: Comment variable name
&&COREF&z..%eval(&y-1) greater than 8 chars";
                        end;
                    %end;
                %end;
            %end;
         run;
         data __totran_&&id&i..;
            set __totran;
         run;
         proc sql noprint;
           create table __POSTTRAN as
             select usubjid, &&id&i..
                    %do z =  1 %to &cntCOREF;
                       %do y = 1 %to &cntcol;
                            %if &y = 1 %then %do;
                               ,
                               max(&&COREF&z..) as &&COREF&z..
                            %end; %else %do;
                               ,
                               max(&&COREF&z..%eval(&y-1)) as
&&COREF&z..%eval(&y-1)
                            %end;
                        %end;
                     %end;
             from __TOTRAN
             group by usubjid, &&id&i..;
         quit;
        /*  data __POSTTRAN_&&id&i..;
             set __POSTTRAN;
         run;*/
         %if &i = 1 %then %do;
             data _null_;
                length x $500;
                 %do z =  1 %to &cntCOREF;
                    %do y = 1 %to &cntcol;
                         %if &y = 1 %then %do;
                             x = left(trim("&&COREF&z.."));
                         %end; %else %do;
```

```
                           x = left(trim(x)) ||" &&COREF&z..%eval(&y-1)";
                        %end;
                        %if &y = &cntcol %then %do;
                           x = compbl(x);
                           put "ALERT_I: &cntcol comment colum(s) added to
&outds";
                           put 'ALERT_I: The following colums (' x ") have been
added to &outds";
                        %end;
                     %end;
                  %end;
              run;
           %end;
      %end;
      %else %do; /*This is when the idvar is missing */
         %PUT &alrt.: THERE ARE NO COMMENT RECORDS FOR THIS DOMAIN;
         *data &libout..&outds(label="&label");
         *   set &libin..&ds;
         *run;
         %DsSortLabel(base=&libin..&ds,set=&libin..&ds,out=&libout..&outds);
      %end;
      /* Determine where the input comes from based on run iterations */
      %if &i = 1 %then %do;
        %let input = &libin..&ds;  %end;
      %if &i ne 1 %then %do;
        %let input = &libout..&outds.; %end;
      /* Sort SDTM dataset by ID variable */
      proc sort data=&input out=&libout..&outds.;
        by usubjid  %if &&id&i.. ne  %then &&id&i..;;
      run;
        /*************  begin updates 19AUG2009  (b) (6)
**********************************/
        /*multiple idvar's found */
        %if &idcnt ^= 0 %then %do;
             proc sql noprint;
                     /*get a list of vars not on domain ds*/
                     %let mergecols=%str();
                     select distinct name into :mergecols SEPARATED by ' '
                     from dictionary.columns a
                     where upcase(libname) = upcase("work")
                       and upcase(memname) = upcase("__posttran")
                       and not exists (select 1 from dictionary.columns b
                                       where upcase(libname) = upcase("&libout")
                                         and upcase(memname) = upcase("&outds")
                                         and upcase(a.name)=upcase(b.name));
                     /*get a count of vars on __posttran that are already on domain
dataset*/
                     select count(distinct name) into :cntupdatecols
                     from dictionary.columns a
                     where upcase(libname) = upcase("work")
```

```
                        and upcase(memname) = upcase("__posttran")
                        and upcase(name) not in ('USUBJID', "&&id&i")
                        and exists (select 1 from dictionary.columns b
                                        where upcase(libname) = upcase("&libout")
                                          and upcase(memname) = upcase("&outds")
                                          and upcase(a.name)=upcase(b.name));
                     %let cntupdatecols = %trim(%left(&cntupdatecols));
                     /*set macro vars = list of columns that are already found on
domain dataset*/
                     %if &cntupdatecols ^= 0 %then %do;
                        select distinct name into
:updatecols1-:updatecols&cntupdatecols
                         from dictionary.columns a
                         where upcase(libname) = upcase("work")
                           and upcase(memname) = upcase("__posttran")
                           and upcase(name) not in ('USUBJID', "&&id&i")
                           and exists (select 1 from dictionary.columns b
                                           where upcase(libname) =
upcase("&libout")
                                             and upcase(memname) =
upcase("&outds")
                                             and upcase(a.name)=upcase(b.name));
                     %end;
             quit;
            %*put &mergecols;
            %*put *************mergecols=&mergecols******************;
            /*Here is where the qnam's not found on domain are merged to domain*/
            %if &mergecols ^= %str() %then %do;
                 /* Now we merge together the base dataset and SUPPQUAL */
                 data &libout..&outds.;
                   merge &libout..&outds.(in=a)
                         __posttran(in=b);
                   by usubjid %if &&id&i.. ne  %then &&id&i..;;
                   if b and not a then put 'ALERT' '_I: There is data in Comments
that had no row to merge on!';
                   if a;
                 run;

%DsSortLabel(base=&libin..&ds,set=&libout..&outds,out=&libout..&outds);
            %end;
            %if &cntupdatecols ^= 0 %then %do;
               /*rare but it happens we could have multiple duplicate qnam's to
update on domain for different idvars*/
               /*whether one or many it should be handled here*/
               %do x = 1 %to &cntupdatecols;
                %if &&updatecols&x.. ne  %then %do;
                   proc sql noprint;
                       select *
                         from __posttran as a
                        where not exists
```

```
                           (select 1
                            from &libout..&outds.  as b
                            where a.usubjid  = b.usubjid
                             and a.&&id&i.. = b.&&id&i.. /*idvar */
                            )
                        and &&updatecols&x.. is not null;
                  quit;
                  %if &sqlobs ne 0 %then %do;
                     %put &alrt.: There is data in Comments that had no row to
merge on!;
                  %end;
                  %put &alrt.: updating values for %left(&&updatecols&x..) on
dataset %left(&libout..&outds.);
                  proc sql;
                     update &libout..&outds. as a
                         set &&updatecols&x.. = (select &&updatecols&x.. /*qnam[i]
*/
                            from __posttran as b
                            where a.usubjid  = b.usubjid
                             and a.&&id&i.. = b.&&id&i..  /*idvar */
                             and &&updatecols&x.. is not null
                             )
                       where exists (select 1
                            from __posttran as b
                            where a.usubjid  = b.usubjid
                             and a.&&id&i.. = b.&&id&i.. /*idvar */
                             and &&updatecols&x.. is not null
                         );
                  quit;
                  %put &alrt.: %left(&sqlobs) records on dataset
%left(&libout..&outds.) have been updated;
               %end;

%DsSortLabel(base=&libin..&ds,set=&libout..&outds,out=&libout..&outds);
            %end;
         %end;
      /*************  end updates 19AUG2009  (b) (6)
************************************/
      /*uniqe qname across idvar's or first idvar iteration just merge*/
      %end; %else %do;
       /* Now we merge together the base dataset and Comments */
       data &libout..&outds.;
         merge &libout..&outds.(in=a)
               __posttran(in=b);
         by usubjid %if &&id&i.. ne  %then &&id&i..;;
         if b and not a then put 'ALERT' '_I: There is data in Comments that had no
row to merge on!';
         if a;
       run;
       %DsSortLabel(base=&libin..&ds,set=&libout..&outds,out=&libout..&outds);
```

```
      %end;
   %*put ****************************  &&id&i..
**************************************************;
        /* Time to remove the records from the comments dataset that have been put
back on the domain dataset*/
       /**/

        %if %upcase(&delco) ^= N %then %do;
            proc sql;
              delete
               from &libco1..&CO a
              where upcase(rdomain) = upcase("&ds")
                and a.coref is not null
                and exists (select 1
                              from __totran b
                             where a.usubjid = b.usubjid
                               and upcase(a.IDVAR) = "&&id&i"
                               and b.&&id&i.. =
                                   %if %upcase(&idtype) = NUM %then %do;
                                     input(idvarval,best.)
                                   %end; %else %do;
                                     idvarval
                                   %end;
                               and a.coref = b.coref
                               and exists (select 1 from &libout..&outds. c
                                           where b.usubjid = c.usubjid
                                             and c.&&id&i.. =
                                                 %if %upcase(&idtype) = NUM %then
%do;
                                                   input(idvarval,best.)
                                                 %end; %else %do;
                                                   idvarval
                                                 %end;
                                          )
                            );
            quit;
            /*data &libco..&CO; set &libco..&CO;run;*/
            %if &sqlobs ne 0 %then %do;
              %put &alrt.: %left(&sqlobs) records deleted from %left(&co) dataset!;
            %end;
            /*added to fix issue with obs and proc compare during testing*/
            %DsSortLabel(base=&libco..&CO,set=&libco1..&CO,out=&libco1..&CO);
        %end;
   %end;
%end; *End merging together Comments;
%if &COFND = 0 %then %do;
  data &libout..&outds.(label="&label");
    set &libin..&ds;
  run;
  %DsSortLabel(base=&libin..&ds,set=&libin..&ds,out=&libout..&outds);
```

```
%end; %else %do;

proc sql noprint; select count(*) into : cocnt from &libco1..&co;quit;
%*put cocnt = &cocnt;
%if &cocnt = 0 %then %do;
    %put &alrt.: All comment records have been merged onto %left(&ds) and there are
no more records in %left(&co).  Dataset will now be deleted;
    proc datasets library=&libco1;
       delete &co.;
    quit;run;
%end; %else %if %upcase(&movco) = Y %then %do;
    %DsSortLabel(base=&libco..&CO,set=&libco1..&CO,out=&libco1..&CO);
%end;


%end;


/* Cleanup the work directory */
proc datasets library=work nodetails nolist;
  delete  __totran: __POSTTRAN __TMPSRT;
run; quit;
%goto fin;
%err:

%if &libin ne &libout and &outds %then %do;
    proc datasets library=&libout. nodetails nolist;
      delete &outds. &co.;
    run; quit;
%end;

%fin:
%put &id: End;

%mend;
```