

```

*****
** Program Name   : admh.sas                               **
** Date Created  : 09Mar2021                               **
** Programmer Name : (b) (6)                               **
** Purpose       : Create admh dataset                     **
** Input data    : mh suppmh adsl                          **
** External file : ../prjC459/nda2_unblinded_esub/euaext_esub_adam/saseng/cdisc3_0/data **
** Output data   : admh.sas7bdat                           **
*****
options mprint mlogic symbolgen mprint symbolgen mlogic nocenter missing=" ";

proc datasets library=WORK kill nolist nodetails;
quit;

**Setup the environment**
%let
oprot=/Volumes/app/cdars/prod/sites/cdars4/prjC459/nda2_unblinded_esub/bla_euaext_esub_sdtm/saseng/cdisc3_0;
%let prot=/Volumes/app/cdars/prod/sites/cdars4/prjC459/nda2_unblinded_esub/euaext_esub_adam/saseng/cdisc3_0;
libname dataprot "&oprot./data" access=readonly;
libname datvprot "&prot./data_vai";
%let expath=&prot./data;

proc printto print=&prot./analysis/esub/output/admh.rpt"
log=&prot./analysis/esub/logs/admh.log" new;
run;

*****
* Specification
* Merge MH dataset with SUPPMH dataset.
*****

proc sql noprint;
select distinct QNAM into :_suppmh_keep_vars separated by " " from
dataprot.suppmh;
quit;

*****
*Specification : Reading INPUT SDTM and Supplemental Datasets
*Subsetting Supplemental Dataset based on _supp_subset parameter*
*****

data _spmdel_supp_dsin_subset;
set dataprot.suppmh;
run;

data _spmdel_sdtm_ds;
set dataprot.mh;
run;

*****
*Specification : Supplemental Dataset will be merged with SDTM for all values*;
*of IDVAR including missing values. *;
* a. Find whether IDVAR has a missing a value *;

```

```
* b. Calculate number of non-missing values for IDVAR *;  
* c. Checking whether non-missing value of IDVAR is character or Numeric *;  
*****,
```

```
proc sql noprint;  
  select NMISS(distinct idvar) into :_cntvar from _spmdel_supp_dsin_subset;  
  select N(distinct idvar) into :_cntvar1 from _spmdel_supp_dsin_subset;  
quit;
```

```
proc sql noprint;  
  select distinct idvar into :_idvar1 - :_idvar1 from _spmdel_supp_dsin_subset  
  where idvar is not missing;  
quit;
```

```
data _spmdel_supp_dsin_subset_idvar1;  
  set _spmdel_supp_dsin_subset;  
  where idvar="MHSEQ";  
run;
```

```
*****,  
*Specification :Tranposing Supplemental Dataset *;  
*****,
```

```
proc sort data=_spmdel_supp_dsin_subset_idvar1;  
  by studyid usubjid idvar idvarval;  
quit;
```

```
proc transpose data=_spmdel_supp_dsin_subset_idvar1  
  out=_spmdel_supp_dsin_idvar1_h;  
  by studyid usubjid idvar idvarval;  
  id qnam;  
  idlabel qlabel;  
  var qval;  
quit;  
*****,  
*Specification :Creating IDVAR from IDVARVAL *;  
*****,
```

```
data _spmdel_temp(keep=MHSEQ);  
  set _spmdel_sdtm_ds;  
run;
```

```
data _spmdel_suppsds1 (drop=idvar idvarval _NAME_ _LABEL_);  
  set _spmdel_supp_dsin_idvar1_h;  
  
  if idvar="MHSEQ";  
  MHSEQ=input(idvarval, best12.);  
run;
```

```
proc sort data=_spmdel_sdtm_ds out=_ds1;  
  by STUDYID USUBJID MHSEQ;  
run;
```

```
proc sort data=_spmdel_suppsds1 out=_ds2;
```

```
by STUDYID USUBJID MHSEQ;
run;
```

```
data _spmdel_sdtm_temp_out1;
merge _ds1(in=d1) _ds2(in=d2);
by STUDYID USUBJID MHSEQ;
```

```
if d1;
run;
```

```
*****;
*Specification : Final Merged output dataset *;
*****;
```

```
data _mh;
set _spmdel_sdtm_temp_out1;
run;
```

```
*****;
* Specification *;
* If MHCAT or MHTERM have missing values, then drop record and display *;
* the dropped record in a supplemental listing. *;
*****;
```

```
data _mh_droprecs;
set _mh;

if MHCAT eq '' or MHTERM eq '' then
do;
output _droprecs;
end;
else
do;
output _mh;
end;
run;
```

```
proc sort data=_droprecs out=_droprecs(keep=STUDYID USUBJID MHSEQ MHTERM MHCAT
DICTVER MHBDSYCD MHSOC);
by usubjid;
run;
```

```
data _mh;
set _mh;

if ^missing(MHSTDTC) then
do;
length yr $4 mm dd $2;
yr=substr(MHSTDTC, 1, 4);
mm=substr(MHSTDTC, 6, 2);
dd=substr(MHSTDTC, 9, 2);

if yr ne '' then
do;
```

```

dflag=' ';

if (dd eq " " or dd eq "-T") and mm ne " " then
do;
    dd='01';
    dflag='D';
end;

if mm eq " " or mm eq "--" then
do;
    mm='01';
    dd='01';
    dflag='M';
end;
newdate=(trim(left(yr))||'-'||trim(left(mm))||'-'||trim(left(dd)));
ASTDT=input(newdate, ??is8601da.);
format ASTDT date9.;
ASTDTF=dflag;
end;
drop yr mm dd dflag newdate;
end;

if ^missing(MHENDTC) then
do;
    length yr $4 mm dd $2;
    yr=substr(MHENDTC, 1, 4);
    mm=substr(MHENDTC, 6, 2);
    dd=substr(MHENDTC, 9, 2);

    if yr ne ' ' then
    do;
        dflag=' ';

        if (dd eq " " or dd eq "-T") and mm ne " " then
        do;
            fakedate=input((((trim(left(yr))||'-'||trim(left(mm))||'-'||'01')),
                ??is8601da.);
            format fakedate date9.;
            tempdate=intnx('month', fakedate, 1)-1;
            dd=strip(put(day(tempdate), best.));
            dflag='D';
        end;

        if (dd eq " " or dd eq "-T") and mm eq " " or mm eq "--" then
        do;
            mm='12';
            dd='31';
            dflag='M';
        end;
        newdate=(trim(left(yr))||'-'||trim(left(mm))||'-'||trim(left(dd)));
        AENDT=input(newdate, ??is8601da.);
        format AENDT date9.;
        AENDTF=dflag;
        drop fakedate tempdate;
    end;
end;

```

```

        end;
        drop yr mm dd dflag newdate;
    end;

if ^missing(MHDTC) then
do;
    length yr $4 mm dd $2;
    yr=substr(MHDTC, 1, 4);
    mm=substr(MHDTC, 6, 2);
    dd=substr(MHDTC, 9, 2);

    if yr ne ' ' then
        do;
            dflag=' ';

            if (dd eq " " or dd eq "-T") and mm ne " " then
                do;
                    dd='01';
                    dflag='D';
                end;

            if mm eq " " or mm eq "--" then
                do;
                    mm='01';
                    dd='01';
                    dflag='M';
                end;
            newdate=(trim(left(yr))||'-'||trim(left(mm))||'-'||trim(left(dd)));
            ADT=input(newdate, ??is8601da.);
            format ADT date9.;
            ADTF=dflag;
        end;
        drop yr mm dd dflag newdate;
    end;
run;

data _tmpcol2(keep=_usrlst);
    length _usrlst $20;
    drop _string;
    _string="SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N SEX SEXN RACE RACEN ARACE ARACEN
    RANDFL SAFFL COMPLFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT
    TRTETM
    TRTEDTM TRT01A TRT01AN TRT02A TRT02AN TRT01P TRT01PN TRT02P TRT02PN TR01SDT TR01STM
    TR01SDTM TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM
    TR02EDTM
    VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX101DT VAX102DT VAX10UDT VAX201DT
    VAX202DT VAX20UDT UNBLNDDT RANDDT COHORT COHORTN DOSALVL DOSALVLN DOSPLVL
    DOSPLVLN
    DS30KFL PHASE PHASEN AGEGR4 AGEGR4N HIVFL PEDIMMFL MULENRFL PEDREAFL DS3KFL
    AGETR01
    AGETRU01 RAND1FL SAF1FL SAF2FL AP01SDT AP01STM AP01SDTM AP01EDT AP01ETM AP01EDTM
    AP02SDT AP02STM AP02SDTM AP02EDT AP02ETM AP02EDTM";

```

```

do until(_usrlst=' ');
    _count+1;
    _usrlst=scan(_string, _count);
    output;
end;
run;

proc sql noprint;
    create table _tmpcol4 as select distinct upcase(_usrlst) as _usrlst from
        _tmpcol2 where upcase(_usrlst) like 'TR%' or upcase(_usrlst) like 'AP%';
quit;

proc contents data=datvprot.adsl out=_tmpcol(keep=NAME) noprint;
proc sql noprint;
    select distinct upcase(NAME) into : _masterlist separated by " " from _tmpcol
        where (upcase(name) like 'TR%' or upcase(name) like 'AP%')
        and (prxmatch('/TR\d{2}STM\b/', Name) or prxmatch('/TR\d{2}ETM\b/', Name) or
            prxmatch('/TR\d{2}SDT\b/', Name) or prxmatch('/TR\d{2}EDT\b/', Name) or
            upcase(Name) in ('TRTEDT', 'TRTSDT', 'TRTETM', 'TRTSTM') or
            prxmatch('/AP\d{2}[EDT\b|SDT\b]/', Name) or
            prxmatch('/AP\d{2}[STM\b|ETM\b]/', Name) or
            prxmatch('/AP\d{2}[SDTM\b|EDTM\b]/', Name) );
quit;

data _tmpcol3(keep=_mstrlst);
    length _mstrlst $20;
    drop _string;
    _string="TR01EDT TR01ETM TR01SDT TR01STM TR02EDT TR02ETM TR02SDT TR02STM TRTEDT
TRTETM TRTSDT TRTSTM";

    do until(_mstrlst=' ');
        _count+1;
        _mstrlst=scan(_string, _count);
        output;
    end;
run;

proc sql noprint;
    select distinct(a._mstrlst) into : usrlist_missing separated by ' ' from
        _tmpcol3 as a where a._mstrlst not in (select _usrlst from _tmpcol4);
quit;

proc sql noprint;
    create table _list_ (name char(32));
    insert into _list_ values("G_VEXIST") values("");
    select name into:G_NOMATCH separated by ' ' from _list_ where name not
        in (select name from dictionary.macros);
    drop table _list_;
quit;

data _null_;
    length _retlist _retlst2 $2000 _column $40;
    dsid=open(upcase("DATVPROT.ADSL"));
    i=1;

```

```

do while (scan("USUBJID SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N SEX SEXN RACE RACEN
ARACE
ARACEN RANDFL SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT
TRTETM TRTEDTM
TRT01A TRT01AN TRT02A TRT02AN TRT01P TRT01PN TRT02P TRT02PN TR01SDT TR01STM TR01SDTM
TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM TR02EDTM VAX101
VAX102
VAX10U VAX201 VAX202 VAX20U VAX101DT VAX102DT VAX10UDT VAX201DT VAX202DT
VAX20UDT UNBLNDDT
RANDDT COHORT COHORTN DOSALVL DOSALVLN DOSPLVL DOSPLVLN DS30KFL PHASE PHASEN
AGEGR4 AGEGR4N
HIVFL PEDIMMFL MULENRFL PEDREAFL DS3KFL AGETR01 AGETRU01 RAND1FL SAF1FL SAF2FL
TR01SDT
TR01STM TR01SDTM TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT
TR02ETM TR02EDTM",
i, ") > ");
_column=upcase(scan("USUBJID SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N SEX SEXN RACE
RACEN ARACE ARACEN RANDFL SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM
TRTSDTM
TRTEDT TRTETM TRTEDTM TRT01A TRT01AN TRT02A TRT02AN TRT01P TRT01PN TRT02P TRT02PN
TR01SDT TR01STM TR01SDTM TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM
TR02EDT
TR02ETM TR02EDTM VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX101DT VAX102DT
VAX10UDT
VAX201DT VAX202DT VAX20UDT UNBLNDDT RANDDT COHORT COHORTN DOSALVL DOSALVLN
DOSPLVL
DOSPLVLN DS30KFL PHASE PHASEN AGEGR4 AGEGR4N HIVFL PEDIMMFL MULENRFL PEDREAFL
DS3KFL
AGETR01 AGETRU01 RAND1FL SAF1FL SAF2FL TR01SDT TR01STM TR01SDTM TR01EDT TR01ETM
TR01EDTM

TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM TR02EDTM", i, "));

if varnum(dsid, _column) then
do;
_retlist=trim(left(_retlist))||' '||_column;
_retlst2=trim(left(_retlst2))||'/'||_column;
end;
i=i+1;
end;
dsid=close(dsid);
call symput('g_vexist', trim(left(compbl(_retlist))));
call symput('g_vexist2', trim(left(_retlst2))||'/');
run;

proc sort data=_mh out=_ds1;
by USUBJID;
run;

proc sort data=datvprot.adsl out=_ds2;
by USUBJID;
run;

```

```

data _mh;
  merge _ds1(in=d1) _ds2(in=d2 keep=Usubjid SUBJID SITEID AGE AGEU AGEGR1
    AGEGR1N SEX SEXN RACE RACEN ARACE ARACEN RANDFL SAFFL ARM ARMCD ACTARM
    ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM TRTEDTM TRT01A TRT01AN TRT02A
    TRT02AN TRT01P TRT01PN TRT02P TRT02PN TR01SDT TR01STM TR01SDTM TR01EDT
    TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM TR02EDTM VAX101
    VAX102 VAX10U VAX201 VAX202 VAX20U VAX101DT VAX102DT VAX10UDT VAX201DT
    VAX202DT VAX20UDT UNBLNDDT RANDDT COHORT COHORTN DOSALVL DOSALVLN DOSPLVL
    DOSPLVLN DS30KFL PHASE PHASEN AGEGR4 AGEGR4N HIVFL PEDIMMFL MULENRFL PEDREAFL
    DS3KFL AGETR01 AGETR01 RAND1FL SAF1FL SAF2FL TR01SDT TR01STM TR01SDTM
    TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM TR02EDTM);
  by USUBJID;

  if d1;
run;

data _mh;
  merge _mh (in=a) datvprot.adsl (in=b keep=usubjid);
  by usubjid;

  if a and b;
run;

*****
* Specification *;
* Calculate Study Day *;
* If MHCAT is Primary Diagnosis or Secondary Diagnosis then derive ADURN *;
* (b) (6): add GENERAL MEDICAL HISTORY for covid study *;
*****

data _mh;
  length ADURU $10.;
  set _mh;

  If ^Missing(ASDT) and ^Missing(TRTSDT) then
    do;

      If ASDT lt TRTSDT then
        ASTDY=ASDT - TRTSDT;
      Else If ASDT ge TRTSDT then
        ASTDY=ASDT - TRTSDT + 1;
    end;

  If ^Missing(AENDT) and ^Missing(TRTSDT) then
    do;

      If AENDT lt TRTSDT then
        AENDY=AENDT - TRTSDT;
      Else If AENDT ge TRTSDT then
        AENDY=AENDT - TRTSDT + 1;
    end;

  if upcase(MHCAT) in ('PRIMARY DIAGNOSIS', 'SECONDARY DIAGNOSIS',
    'GENERAL MEDICAL HISTORY') then

```



```

do;
  _utilflg='N';

if ((_utilflg eq 'Y') and (MHDUR ne ' ')) then
  do;

    if MHDUR ne ' ' then
      do;
        call is8601_convert('du', 'du', MHDUR, dur1);
        ADURN=round((dur1/(24*60*60*365.25)), 0.01);
        drop dur1;
      end;
    else
      do;
        ADURN=.;
      end;

    if ADURN ne . then
      do;
        ADURU="YEARS";
      end;
  end;
else
  do;
    ADURN=round(((ADT - ASTDT + 1)/365.25), 0.01);

    if ADURN ne . then
      do;
        ADURU="YEARS";
      end;
  end;
drop _utilflg;
end;
run;

*****
* Specification 7 *;
* Attach attributes to all variables as per ADaM Spec *;
*****
*** (b) (6) 06Mar2021-modified variable name ACATn to CATn to align with EUA per request;

proc import file="&expath./Comorbidity_Categories.xlsx" out=_comobi dbms=xlsx
  replace;
  RXLX;
  getnames=yes;
run;

proc import file="&expath./Report_CCI_AIDS_HIV.xlsx" out=_hiv dbms=xlsx replace;
  RXLX;
  datarow=17;
  getnames=no;
run;

data _hiv;

```

```

    set _hiv;
    CAT='AIDS/HIV';
run;

proc import file="&expath./Report_CCI_Any malignancy.xlsx" out=_mali dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _mali;
    set _mali;
    CAT='Any Malignancy';
run;

proc import file="&expath./Report_CCI_Cerebrovascular.xlsx" out=_cere dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _cere;
    set _cere;
    CAT='Cerebrovascular Disease';
run;

proc import file="&expath./Report_CCI_CHF.xlsx" out=_chf dbms=xlsx replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _chf;
    set _chf;
    CAT='Congestive Heart Failure';
run;

proc import file="&expath./Report_CCI_Dementia.xlsx" out=_deme dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _deme;
    set _deme;
    CAT='Dementia';
run;

proc import file="&expath./Report_CCI_Diabetes with comp.xlsx" out=_diabe_c
    dbms=xlsx replace;
    RXLX;

```

```

    datarow=17;
    getnames=no;
run;

data _diabe_c;
    set _diabe_c;
    CAT='Diabetes With Chronic Complication';
run;

proc import file="&expath./Report_CCI_Diabetes without comp.xlsx" out=_diabe
    dbms=xlsx replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _diabe;
    set _diabe;
    CAT='Diabetes Without Chronic Complication';
run;

proc import file="&expath./Report_CCI_Hemiplegia.xlsx" out=_hemip dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _hemip;
    set _hemip;
    CAT='Hemiplegia or Paraplegia';
run;

proc import file="&expath./Report_CCI_Leukemia.xlsx" out=_leuk dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _leuk;
    set _leuk;
    CAT='Leukemia';
run;

proc import file="&expath./Report_CCI_Lymphoma.xlsx" out=_lymph dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _lymph;
    set _lymph;

```

```

    CAT='Lymphoma';
run;

proc import file="&expath./Report_CCI_Metastatic tumour.xlsx" out=_metas
    dbms=xlsx replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _metas;
    set _metas;
    CAT='Metastatic Solid Tumor';
run;

proc import file="&expath./Report_CCI_MI.xlsx" out=_mi dbms=xlsx replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _mi;
    set _mi;
    CAT='Myocardial Infarction';
run;

proc import file="&expath./Report_CCI_Mild liver.xlsx" out=_mild dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _mild;
    set _mild;
    CAT='Mild Liver Disease';
run;

proc import file="&expath./Report_CCI_Mod sev liver.xlsx" out=_modsev dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _modsev;
    set _modsev;
    CAT='Moderate or Severe Liver Disease';
run;

proc import file="&expath./Report_CCI_Peptic ulcer.xlsx" out=_peptic dbms=xlsx
    replace;
    RXLX;
    datarow=17;

```

```

    getnames=no;
run;

data _peptic;
    set _peptic;
    CAT='Peptic Ulcer Disease';
run;

proc import file="&expath./Report_CCI_Periph_vasc.xlsx" out=_peri dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _peri;
    set _peri;
    CAT='Peripheral Vascular Disease';
run;

proc import file="&expath./Report_CCI_Pulmonary.xlsx" out=_pulm dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _pulm;
    set _pulm;
    CAT='Chronic Pulmonary Disease';
run;

proc import file="&expath./Report_CCI_Renal.xlsx" out=_renal dbms=xlsx replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _renal;
    set _renal;
    CAT='Renal Disease';
run;

proc import file="&expath./Report_CCI_Rheumatic.xlsx" out=_rheuma dbms=xlsx
    replace;
    RXLX;
    datarow=17;
    getnames=no;
run;

data _rheuma;
    set _rheuma;
    CAT='Rheumatic Disease';
run;

```

```

data _pt;
  length a b $ 200 c CAT $ 100;
  set _hiv _mali _cere _chf _deme _diabe _c _diabe _hemip _leuk _lymph _metas _mi
    _mild _modsev _peptic _peri _pulm _renal _rheuma;
  drop d e f;
run;

```

```

data report _cci;
  set _pt;
  mhptcd=input(b, best.);
  rename a=term;

```

```

proc sort;
  by mhptcd;
run;

```

```

proc transpose data=report _cci out=t _cci prefix=CAT;
  by mhptcd term;
  var CAT;
run;

```

```

proc sort data=_mh;
  by mhptcd;
run;

```

```

data _mh;
  merge _mh(in=a) t _cci(in=b);
  by mhptcd;
  if a & b then
    COMORBFL='Y';
  else
    COMORBFL='N';
  if a;
  if cat1 ne "AIDS/HIV" then do;
    cat1=upcase(substr(cat1,1,1))||substr(lowcase(cat1),2,length(cat1)-1);
  end;
  if cat2 ne "AIDS/HIV" then do;
    cat2=upcase(substr(cat2,1,1))||substr(lowcase(cat2),2,length(cat2)-1);
  end;
  drop _NAME_ term;
run;

```

```

data admh;
  retain STUDYID USUBJID SUBJID SITEID MHSEQ MHTERM MHDECOD MHPTCD MHBODSYS
    MHBDSYCD MHLTCD MHLTCD MHPTCD MHHLT MHHLTCD MHHLGT MHHLGTCD MHSOC
MHSOCCD
  MHCAT MHSTDTC MHENDTC MHENRTPT MHENTPT DICTVER MHSPID ASTDT ASTDTF ASTDY
  AENDT AENDTF AENDY COMORBFL CAT1 CAT2 ADT ADTF ADURN ADURU DICTVER USUBJID
  SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N SEX SEXN RACE RACEN ARACE ARACEN RANDFL
  SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM
  TRTEDTM
  TRT01A TRT01AN TRT02A TRT02AN TRT01P TRT01PN TRT02P TRT02PN TR01SDT TR01STM
  TR01SDTM TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM

```

FDA-CBER-2022-5812-0735996

```

TR02EDTM VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX101DT VAX102DT VAX10UDT
VAX201DT VAX202DT VAX20UDT UNBLNDDT RANDDT COHORT COHORTN DOSALVL DOSALVLN
DOSPLVL DOSPLVLN DS30KFL PHASE PHASEN AGEGR4 AGEGR4N HIVFL PEDIMMFL MULENRFL
PEDREAFL DS3KFL AGETR01 AGETRU01 RAND1FL SAF1FL SAF2FL;
attrib ASTDT length=8. label="Analysis Start Date" COMORBFL length=$1.
  label="Comorbidity Flag" CAT1 length=$100.
  label="Charlson Comorbidity Index Category 1" CAT2 length=$100.
  label="Charlson Comorbidity Index Category 2" ASTDTF length=$1.
  label="Analysis Start Date Imputation Flag" ASTDY
  label="Analysis Start Relative Day" AENDT length=8. label="Analysis End Date"
  AENDTF length=$1. label="Analysis End Date Imputation Flag" AENDY
  label="Analysis End Relative Day" ADT length=8. label="Analysis Date" ADTF
  length=$1. label="Analysis Date Imputation Flag" ADURN length=8.
  label="Analysis Duration (N)" ADURU length=$10.
  label="Analysis Duration Units";
set _mh(keep=STUDYID USUBJID SUBJID SITEID MHSEQ MHTERM MHDECOD MHPTCD
  MHBODSYS MHBDSYCD MHLT MHLTCD MHPTCD MHLT MHLTCD MHLGT MHLGTCD
MHSOC
  MHSOCCD MHCAT MHSTDTC MHENDTC MHENRTPT MHENTPT DICTVER MHSPID ASTDT ASTDTF
  ASTDY AENDT AENDTF AENDY COMORBFL CAT1 CAT2 ADT ADTF ADURN ADURU DICTVER
  USUBJID SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N SEX SEXN RACE RACEN ARACE
  ARACEN RANDFL SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT
  TRTETM TRTEDTM TRT01A TRT01AN TRT02A TRT02AN TRT01P TRT01PN TRT02P TRT02PN
  TR01SDT TR01STM TR01SDTM TR01EDT TR01ETM TR01EDTM TR02SDT TR02STM TR02SDTM
  TR02EDT TR02ETM TR02EDTM VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX101DT
  VAX102DT VAX10UDT VAX201DT VAX202DT VAX20UDT UNBLNDDT RANDDT COHORT COHORTN
  DOSALVL DOSALVLN DOSPLVL DOSPLVLN DS30KFL PHASE PHASEN AGEGR4 AGEGR4N HIVFL
  PEDIMMFL MULENRFL PEDREAFL DS3KFL AGETR01 AGETRU01 RAND1FL SAF1FL SAF2FL);
run;

proc sort data=admh out=datvprot.admh(label='Medical History Analysis Dataset');
  by MHCAT USUBJID MHSPID MHTERM;
run;

proc printto;
run;

```