

```

*****
** Program Name   : adcm.sas                               **
** Date Created  : 09Mar2021                               **
** Programmer Name : (b) (6)                               **
** Purpose       : Create adcm dataset                     **
** Input data    : cm suppcm adsl                          **
** Output data   : adcm.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";
libname viewpx "/Volumes/app/saseng/prod/cdisc3_0/view/" access=readonly;

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

proc sql noprint;
  select distinct(version) into :_dictver from viewpx.whodrug;
quit;

*****
* Specification 2 *;
* Merge SDTM with Supplemental dataset if Supplemental CM Dataset Exists *;
* Abort the program if SUPPCM.DICTVER does not match WHODRUG.VERSION or *;
* if SUPPCM does not have QNAM=DICTVER *;
*****
** (b) (6) modified to fix mapping issue;

data cm;
  length CMSTDTC $20;
  set dataprot.cm;

  if CMTRT=" and CMSTDTC=" then
    delete;

  if length(CMSTDTC)=10 then
    CMSTDTC=strip(CMSTDTC)||"T23:59:59";
  **to fix period issue when subject dosing 2 doses in the same day;
run;

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

```

```

*****;
*Specification 1: Reading INPUT SDTM and Supplemental Datasets *;
*Subsetting Supplemental Dataset based on _supp_subset parameter*;
*****;

data _spmdel_supp_dsin_subset;
    set dataprot.suppem;
run;

data _spmdel_sdtm_ds;
    set cm;
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="CMSEQ";
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=CMSEQ);
  set _spmdel_sdtm_ds;
run;

data _spmdel_supps1 (drop=idvar idvarval _NAME _LABEL _);
  set _spmdel_supp_dsin_idvar1_h;

  if idvar="CMSEQ";
  CMSEQ=input(idvarval, best12.);
run;

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

proc sort data=_spmdel_supps1 out=_ds2;
  by STUDYID USUBJID CMSEQ;
run;

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

  if d1;
run;

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

data _condrug;
  set _spmdel_sdtm_temp_out1;
run;

proc sql noprint;
  select distinct(dictver) into :_supp_dictver from _condrug where dictver ne "";
quit;

*****
* Specification 4 *;
* Derive the following variables: *;
* ASTDT :Analysis Start Date *;
* ASTDTF :Analysis Start Date Imputation Flag *;
* ASTTM :Analysis Start Time *;
* ASTTMF :Analysis Start Time Imputation Flag *;
* AENDT :Analysis End Date *;
* AENTM :Analysis End Time *;
* AENDTF :Analysis End Date Imputation Flag *;
* AENTMF :Analysis End Time Imputation Flag *;
* ASTDTM :Analysis Start Date/Time *;
* AENDTM :Analysis End Date/Time *;
* Note: Time variables are not derived if time is not collected for any of the records *;

```

FDA-CBER-2022-5812-0735849

*****,

```
proc sql noprint;
  select count(1) into :_start_time_exists from _condrug where
    length(compress(cmstdtc)) > 10;
  select count(1) into :_end_time_exists from _condrug where
    length(compress(cmendtc)) > 10;
quit;

data _condrug;
  set _condrug;

  if ^missing(CMSTDTC) then
    do;
      length yr $4 mm dd $2;
      yr=substr(CMSTDTC, 1, 4);
      mm=substr(CMSTDTC, 6, 2);
      dd=substr(CMSTDTC, 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(CMENDTC) then
    do;
      length yr $4 mm dd $2;
      yr=substr(CMENDTC, 1, 4);
      mm=substr(CMENDTC, 6, 2);
      dd=substr(CMENDTC, 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;
drop yr mm dd dflag newdate;
end;
*****
* SPECIFICATION 3 *;
* -Process Time (if requested in macro call) *;
* Substring isodate into hour, minutes and seconds. *;
* Based upon _imputation_rule_time (start or stop), impute missing values in time. *;
* If _imptimeflag is supplied in macro call, then the highest time imputed will *;
* be populated. *;
* If no time exists, then hour, minutes and seconds are imputed and flag will contain 'H'*;
* if only hour exists, then minutes and seconds will be imputed and flag will contain 'M'*;
* if hour and minutes exist, then seconds will be imputed and flag will contain 'S' *;
*****
if ^missing(CMSTDTC) then
do;
  length hr mn sc $2 newtime $8;
  yr=substr(CMSTDTC, 1, 4);
  hr=substr(CMSTDTC, 12, 2);
  mn=substr(CMSTDTC, 15, 2);
  sc=substr(CMSTDTC, 18, 2);
  ;

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

    if sc eq " " then
    do;
      sc='00';

```

```

        tflag='S';
    end;

    if mn eq " " then
        do;
            mn='00';
            tflag='M';
        end;

    if hr eq " " then
        do;
            hr='00';
            tflag='H';
        end;
        newtime=(trim(left(hr))||':'||trim(left(mn))||':'||trim(left(sc)));
        ASTTM=input(newtime, ??is8601tm.);
        format ASTTM time8.;
        ASTTMF=tflag;
        drop tflag;
    end;
    drop yr hr mn sc newtime;
end;

if ^missing(CMENDTC) then
do;
    length hr mn sc $2 newtime $8;
    yr=substr(CMENDTC, 1, 4);
    hr=substr(CMENDTC, 12, 2);
    mn=substr(CMENDTC, 15, 2);
    sc=substr(CMENDTC, 18, 2);
    ;

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

            if sc eq " " then
                do;
                    sc='59';
                    tflag='S';
                end;

            if mn eq " " then
                do;
                    mn='59';
                    tflag='M';
                end;

            if hr eq " " then
                do;
                    hr='23';
                    tflag='H';
                end;
            newtime=(trim(left(hr))||':'||trim(left(mn))||':'||trim(left(sc)));

```

```

        AENTM=input(newtime, ??is8601tm.);
        format AENTM time8.;
        AENTMF=tflag;
        drop tflag;
    end;
    drop yr hr mn sc newtime;
end;

ASTDTM=ifn(missing(ASTTM), ASTDT, dhms(ASTDT, 0, 0, ASTTM));
format ASTDTM datetime20.;
drop AENTM AENTMF;
run;

*****
* SPECIFICATION 1: *;
* Protocol Specific Imputation Rules. *;
*****

data _ConDrug;
    set _ConDrug;
run;

*****
* Specification 2 *;
* Imputation of APxxSTM/APxxETM/APxxSDTM/APxxEDTM based on Time collected or missing. *;
*****

Data _Null_ ;
    Do i=1 to 2;
        x=Strip(Put(i, 2.));
        y=Strip(Put(i, z2.));
        Call Symput(cats("_TRSDT", x), cats("TR", y, "SDT"));
        Call Symput(cats("_TREDT", x), cats("TR", y, "EDT"));
        Call Symput(cats("_TRSTM", x), cats("TR", y, "STM"));
        Call Symput(cats("_TRETm", x), cats("TR", y, "ETM"));
        Call Symput(cats("_APSDT", x), cats("AP", y, "SDT"));
        Call Symput(cats("_APEDT", x), cats("AP", y, "EDT"));
        Call Symput(cats("_APSTM", x), cats("AP", y, "STM"));
        Call Symput(cats("_APETM", x), cats("AP", y, "ETM"));
        Call Symput(cats("_APSDTM", x), cats("AP", y, "SDTM"));
        Call Symput(cats("_APEDTM", x), cats("AP", y, "EDTM"));
        Call Symput(cats("_TRSDTM", x), cats("TR", y, "SDTM"));
        Call Symput(cats("_TREDTM", x), cats("TR", y, "EDTM"));
        Call Symput(cats("_PrdxxN", x), y);
    End;
Run;

Data work.adsl;
    Set datvprot.adsl;

    If Missing(TR01STM) then
        _apx_TR01STM="00:00:00"t;
    Else
        _apx_TR01STM=TR01STM;

```

```

If Missing(TR01ETM) then
  _apx_TR01ETM="23:59:59"t;
Else
  _apx_TR01ETM=TR01ETM;

If ^Missing(TR01SDT) then
  _apx_TR01SDTM=dhms(TR01SDT, 0, 0, _apx_TR01STM);

If ^Missing(TR01EDT) then
  _apx_TR01EDTM=dhms(TR01EDT, 0, 0, _apx_TR01ETM);
AP01SDT=datepart(_apx_TR01SDTM);
AP01STM=timepart(_apx_TR01SDTM);

If Missing(TR02STM) then
  _apx_TR02STM="00:00:00"t;
Else
  _apx_TR02STM=TR02STM;

If Missing(TR02ETM) then
  _apx_TR02ETM="23:59:59"t;
Else
  _apx_TR02ETM=TR02ETM;

If ^Missing(TR02SDT) then
  _apx_TR02SDTM=dhms(TR02SDT, 0, 0, _apx_TR02STM);

If ^Missing(TR02EDT) then
  _apx_TR02EDTM=dhms(TR02EDT, 0, 0, _apx_TR02ETM);
AP02SDT=datepart(_apx_TR02SDTM);
AP02STM=timepart(_apx_TR02SDTM);

if ^Missing(_apx_TR02SDTM-1) then
  do;
    _apx_TR01EDTM=min((_apx_TR02SDTM-1), (_apx_TR01EDTM+((365)*86400)));
  end;
else
  _apx_TR01EDTM=_apx_TR01EDTM+((365)*86400);
AP01EDT=datepart(_apx_TR01EDTM);
AP01ETM=Timepart(_apx_TR01EDTM);
AP01SDTM=dhms(AP01SDT, 0, 0, AP01STM);
AP01EDTM=dhms(AP01EDT, 0, 0, AP01ETM);
Attrib AP01SDT Label="Period 01 Start Date" AP01EDT Label="Period 01 End Date"
  AP01STM Label="Period 01 Start Time" AP01ETM Label="Period 01 End Time"
  AP01SDTM Label="Period 01 Start Datetime" AP01EDTM
  Label="Period 01 End Datetime";
Format AP01SDT AP01EDT date9. AP01STM AP01ETM time8. AP01SDTM AP01EDTM
  datetime20.;
_apx_TR02EDTM=_apx_TR02EDTM+((365)*86400);
AP02EDT=datepart(_apx_TR02EDTM);
AP02ETM=Timepart(_apx_TR02EDTM);
AP02SDTM=dhms(AP02SDT, 0, 0, AP02STM);
AP02EDTM=dhms(AP02EDT, 0, 0, AP02ETM);
Attrib AP02SDT Label="Period 02 Start Date" AP02EDT Label="Period 02 End Date"

```

FDA-CBER-2022-5812-0735854


```

AP02STM Label="Period 02 Start Time" AP02ETM Label="Period 02 End Time"
AP02SDTM Label="Period 02 Start Datetime" AP02EDTM
Label="Period 02 End Datetime";
Format AP02SDT AP02EDT date9. AP02STM AP02ETM time8. AP02SDTM AP02EDTM
datetime20.;
run;

*****
*Specification *;
*Create a list of all Treatment Time related column names from user input of G_ADSL_VARS *;
*Create a Master List of all Treatment Time related column names from datvprot.adsl *;
*Compare User list with Master List *;
*Bailout if any of the columns of Master List are not specified by USER *;
*****

data _tmpcol2(keep=_usrlst);
    length _usrlst $20;
    drop _string;
    _string="SUBJID SITEID AGE AGEU SEX SEXN RACE RACEN ARACE ARACEN SAFFL COMPLFL ARM
ARMCD ACTARM ACTARMCD
TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM TRTEDTM V01DT V02DT V03DT V04DT VAX101DT
VAX10UDT VAX102DT VAX201DT VAX202DT COHORTN COHORT
DOSALVLN DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE UNBLNDDT TRTARN TRTAR TRTPRN
TRTPR PROCGR1 PROCGR1N MULENRFL DS30KFL HIVFL AGETR01
AGEGR1 AGEGR1N AGEGR4 AGEGR4N TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A TRT01AN
TRT01P TRT01PN TRT02A TRT02AN TRT02P TRT02PN
VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX20UDT DS3KFL RANDFL 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=work.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 (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="AP01EDT AP01EDTM AP01ETM AP01SDT AP01SDTM AP01STM AP02EDT AP02EDTM AP02ETM
AP02SDT AP02SDTM AP02STM TRTEDT TRTETM TRTSDT TRTSTM";

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

proc sql noprint;
  select distinct(a._mstrlst) into :_usrlst_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 _retlst _retlst2 $2000 _column $40;
  dsid=open(upcase("WORK.ADSL"));
  i=1;

  do while (scan("USUBJID SUBJID SITEID AGE AGEU SEX SEXN RACE RACEN ARACE ARACEN SAFFL
ARM ARMCD ACTARM
ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM TRTEDTM V01DT V02DT V03DT V04DT
VAX101DT VAX10UDT VAX102DT VAX201DT VAX202DT COHORTN
COHORT DOSALVLN DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE UNBLNDDT MULENRFL
DS30KFL HIVFL AGETR01 AGEGR1 AGEGR1N AGEGR4 AGEGR4N
TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A TRT01AN TRT01P TRT01PN TRT02A TRT02AN
TRT02P TRT02PN VAX101 VAX102 VAX10U VAX201 VAX202
VAX20U VAX20UDT DS3KFL RANDFL RAND1FL SAF1FL SAF2FL AP01SDT AP01STM AP01SDTM
AP01EDT AP01ETM AP01EDTM AP02SDT AP02STM AP02SDTM
AP02EDT AP02ETM AP02EDTM TR01SDT TR01STM TR01SDTM TR01EDT TR01ETM TR01EDTM
TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM TR02EDTM",i,)
>");
  _column = upcase(scan("USUBJID SUBJID SITEID AGE AGEU SEX SEXN RACE RACEN ARACE
ARACEN SAFFL ARM ARMCD
ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM TRTEDTM V01DT V02DT V03DT
V04DT VAX101DT VAX10UDT VAX102DT VAX201DT VAX202DT
COHORTN COHORT DOSALVLN DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE UNBLNDDT
MULENRFL DS30KFL HIVFL AGETR01 AGEGR1 AGEGR1N AGEGR4 AGEGR4N
TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A TRT01AN TRT01P TRT01PN TRT02A TRT02AN
TRT02P TRT02PN VAX101 VAX102 VAX10U VAX201 VAX202
VAX20U VAX20UDT DS3KFL RANDFL RAND1FL SAF1FL SAF2FL AP01SDT AP01STM AP01SDTM
AP01EDT AP01ETM AP01EDTM AP02SDT AP02STM AP02SDTM
AP02EDT AP02ETM AP02EDTM 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;

*****;
* SPECIFICATION - Merge the datasets. *;
* 1.Sort the datasets by key merge-by variables. *;
* 2.Merge the datasets using full join, inner join,left join or right join. *;
* Default is full join. *;
* 3.Generate a list of subjects not having any matching observations based on *;
* key-by variables. *;
*****;

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

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

data _condrug;
  merge _ds1(in=d1) _ds2(in=d2 keep= Usubjid SUBJID SITEID AGE AGEU SEX SEXN RACE RACEN ARACE
ARACEN SAFFL
  ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM TRTEDT TRTETM TRTEDTM V01DT
V02DT V03DT V04DT VAX101DT VAX10UDT VAX102DT VAX201DT
  VAX202DT COHORTN COHORT DOSALVLN DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE
UNBLNDDT MULENRFL DS30KFL HIVFL AGETR01 AGEGR1 AGEGR1N
  AGEGR4 AGEGR4N TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A TRT01AN TRT01P TRT01PN
TRT02A TRT02AN TRT02P TRT02PN VAX101 VAX102 VAX10U
  VAX201 VAX202 VAX20U VAX20UDT DS3KFL RANDFL RAND1FL SAF1FL SAF2FL AP01SDT AP01STM
AP01SDTM AP01EDT AP01ETM AP01EDTM AP02SDT AP02STM
  AP02SDTM AP02EDT AP02ETM AP02EDTM TR01SDT TR01STM TR01SDTM TR01EDT TR01ETM
TR01EDTM TR02SDT TR02STM TR02SDTM TR02EDT TR02ETM
  TR02EDTM ) ;
  by USUBJID;
  if d1;
run;

*****;
* Specification *;
* Derive the following variables: *;
* PREFL :Pre-treatment Flag *;
* ONPERFL :On-Period Flag *;

```

```

* ONTRXXFL :On-Period XX Flag *;
* ASTDY :Analysis Start Relative Day *;
* AENDY :Analysis End Relative Day *;
* Note: As per (b) (6) , If Time collected in Condrug then use DTM vars when deriving *;
* ONPERFL variable. *;
* Updated PREFL by removing the logic handling CMSTRF *;
*****;

data adcm;
  set _condrug;

  if ASTDT < AP01SDT then
    PREFL='Y';

  if n(ap01sdtm) and (((ASTDTM <=AP01EDTM) and ((AENDT >=AP01SDT)
    or (upcase(CMENRTPT)='ONGOING')))) or ((ASTDTM >=AP01SDTM)
    and (ASTDTM <=AP01EDTM))) then
    do;
      ONTR01FL='Y';
      ONPERFL='Y';
    end;
  label ONTR01FL="On-Period 01 Flag";

  if n(ap02sdtm) and (((ASTDTM <=AP02EDTM) and ((AENDT >=AP02SDT)
    or (upcase(CMENRTPT)='ONGOING')))) or ((ASTDTM >=AP02SDTM)
    and (ASTDTM <=AP02EDTM))) then
    do;
      ONTR02FL='Y';
      ONPERFL='Y';
    end;
  label ONTR02FL="On-Period 02 Flag";
  *****;

* Specification 1 *;
* Calculate the study day using the dates supplied by the user. *;
*****;

If ^Missing(AstDt) and ^Missing(TrtSdt) then
  do;

    If AstDt lt TrtSdt then
      AstDy=AstDt - TrtSdt;
    Else If AstDt ge TrtSdt then
      AstDy=AstDt - 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;

run;

```

```
*****
* Specification 7 *;
* Merge WHODRUG Dictionary with SDTM *;
* Get ATC Level Code/Names from WHODRUG *;
*****;
```

```
data whodrug;
  set viewpx.whodrug;
run;
```

```
proc sql noprint;
  select max(length(drugname)) into: _len from whodrug;
quit;
```

```
data whodrug;
  set whodrug(rename=(drugname=drugname_));
  length temp drugname drgname1 $200;
  temp=substr(drugname_, 1, 200);
  reverse=reverse(temp);
  pos=indexc(reverse, ",.");

  if substr(reverse(temp), 1, 1) in (" ", ".", ",") or substr(drugname_, 201, 1)
    in (" ", ".", ",") then
    do;
      val_1=temp;
      rval_1=substr(drugname_, 201);
    end;
  else
    do;
      val_1=substr(temp, 1, 200-pos+1);
      rval_1=substr(drugname_, 200-pos+2);
    end;
  drugname=val_1;

  if length(rval_1) le 200 then
    drgname1=rval_1;
  else if length(rval_1) gt 200 then
    drugname_=rval_1;
run;
```

```
proc sql noprint;
  create table _all as select a.*, b.drugname, b.drgname1, b.atc1name as ATC1T
    length=200 informat=$200., b.atc2name as ATC2T length=200 informat=$200.,
    b.atc3name as ATC3T length=200 informat=$200., b.atc4name as ATC4T length=200
    informat=$200., b.atc5name as ATC5T length=200 informat=$200., b.atc1code as
    ATC1CD, b.atc2code as ATC2CD, b.atc3code as ATC3CD, b.atc4code as ATC4CD,
    b.atc5code as ATC5CD from adcm as a left join whodrug as b on
    a.cmcode=b.drugcode;
quit;
```

```
*****
* Specification 7.1 *;
* STUDY SPECIFIC VARIABLES/FLAGS DERIVATION *;
```

* CMOPIDFL FLAG DERIVATION *;

*****;

```
data _all;
  set _all;
  length VPHASE $100.;
  label VPHASE="Vaccine Phase" VPHASEN="Vaccine Phase(N)"
        VAXNO="CM Occured after Which Vaccination";

  if (ASTDT<UNBLNDDT or UNBLNDDT=.) and not (UNBLNDDT=. and ASTDT>=VAX201DT>.)
  then
    do;

      if (prefl='Y' and vax101dt ne .) or (astdt ne . and vax101dt=.) then
        VPHASE='Pre-Vaccination';
      else if (VAX101DT>. and ((VAX101DT<=ASTDT<=V01DT and coalesce(vax102dt,vax10udt)=.)
              or (VAX101DT<=ASTDT<coalesce(vax102dt,vax10udt)))) then
        VPHASE='Vaccination 1';
      else if .<coalesce(vax102dt,vax10udt)<=ASTDT<=V01DT then
        VPHASE='Vaccination 2';
      else if .<V01DT<ASTDT<=V02DT then
        VPHASE='Follow Up 1';
      else if ASTDT>V02DT>. then
        VPHASE='Follow Up 2';
    end;
  else
    do;

      if (astdt ne . and vax201dt=.) or (unblnddt<=astdt<vax201dt) then
        VPHASE='After unblinding and before Vaccination 3';
      else if (VAX201DT>. and ((VAX201DT<=ASTDT<=V03DT and vax202dt=.)
              or (VAX201DT<=ASTDT<vax202dt))) then
        VPHASE='Vaccination 3';
      else if .<VAX202DT<=ASTDT<=V03DT then
        VPHASE='Vaccination 4';
      else if .<V03DT<ASTDT<=V04DT then
        VPHASE='Follow Up 3';
      else if ASTDT>V04DT>. then
        VPHASE='Follow Up 4';
    end;

  if CMSTDTC=" then
    VPHASE=";

  if VPHASE='Pre-Vaccination' then
    VPHASEN=0;
  else if VPHASE='Vaccination 1' then
    VPHASEN=1;
  else if VPHASE='Vaccination 2' then
    VPHASEN=2;
  else if VPHASE='Follow Up 1' then
    VPHASEN=3;
  else if VPHASE='Follow Up 2' then
    VPHASEN=99;
```

```

else if VPHASE='After unblinding and before Vaccination 3' then
  VPHASEN=4;
else if VPHASE='Vaccination 3' then
  VPHASEN=5;
else if VPHASE='Vaccination 4' then
  VPHASEN=6;
else if VPHASE='Follow Up 3' then
  VPHASEN=7;
else if VPHASE='Follow Up 4' then
  VPHASEN=100;

if vphasen in (3, 99) then
  do;

    if vax102dt ne . then
      VAXNO=2;
    else if vax101dt ne . then
      VAXNO=1;
  end;
else if vphasen in (7, 100) then
  do;

    if vax202dt ne . then
      VAXNO=4;
    else if vax201dt ne . then
      VAXNO=3;
  end;
else if vphasen in (5, 6) then
  do;
    VAXNO=vphasen-2;
  end;
else if vphasen in (1,2) then VAXNO=vphasen;
  else if vphasen in (4) then do;
    if .<coalesce(vax102dt,vax10udt)<=ASTDT then VAXNO=2;
    else if .<vax101dt<=ASTDT then VAXNO=1;
  end;
if PREFL='N' and vphasen=0 then
  do;
    vphasen=1;
    vphase='Vaccination 1';
    VAXNO=1;
  end;

if vphase='After unblinding and before Vaccination 3' and vax201dt ne . then
  do;
    ONTR01FL="";
    ONTR02FL="";
  end;
else if vphasen>=99 then
  do;
    ONTR01FL="";
    ONTR02FL="";
    ONPERFL="";
  end;

```

```

if ONTR01FL='Y' or ONTR02FL='Y' then
  ONPERFL='Y';
else
  ONPERFL="";

proc sort;
  by USUBJID CMSEQ;
run;

data _all;
  set _all;
  cmstdtc=strip(scan(cmstdtc, 1, "T"));
run;

proc sort data=_all;
  by usubjid cmseq;
quit;

data datvprot.adcm(label='Concomitant Medications Analysis Dataset');
retain STUDYID USUBJID SUBJID SITEID CMSEQ CMCAT CMSCAT CMSTDTC CMENDTC CMTRT
CMDECOD CMCODE
PREFL ONPERFL ONTR01FL ONTR02FL ASTDT ASTDTF ASTTM ASTTMF ASTDTM ASTDY AENDT
AENDTF AENDY
CMCLAS CMCLASCD DRUGNAME DRGNAME1 ATC1CD ATC1T ATC2CD ATC2T ATC3CD ATC3T
ATC4CD ATC4T ATC5CD
ATC5T CMSTDY CMENDY CMDOSE CMROUTE CMDOSTXT CMDOSU CMDOSFRQ CMENRTPT
CMENPT EPOCH DICTVER
CMCLAS1 CMCLAS2 CMCLSCD1 CMCLSCD2 CMCODE DICTVER USUBJID SUBJID SITEID AGE AGEU
SEX SEXN RACE
RACEN ARACE ARACEN SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM TRTSDTM
TRTEDT TRTETM TRTEDTM
V01DT V02DT V03DT V04DT VAX101DT VAX10UDT VAX102DT VAX201DT VAX202DT COHORTN
COHORT DOSALVLN
DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE UNBLNDDT MULENRFL DS30KFL HIVFL AGETR01
AGEGR1 AGEGR1N
AGEGR4 AGEGR4N TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A TRT01AN TRT01P TRT01PN
TRT02A
TRT02AN TRT02P TRT02PN VAX101 VAX102 VAX10U VAX201 VAX202 VAX20U VAX20UDT DS3KFL
RANDFL
RAND1FL SAF1FL SAF2FL;
set _all(keep = STUDYID USUBJID SUBJID SITEID CMSEQ CMCAT CMSCAT CMSTDTC CMENDTC CMTRT
CMDECOD
CMCODE PREFL ONPERFL ONTR01FL ONTR02FL ASTDT ASTDTF ASTTM ASTTMF ASTDTM ASTDY
AENDT AENDTF
AENDY CMCLAS CMCLASCD DRUGNAME DRGNAME1 ATC1CD ATC1T ATC2CD ATC2T ATC3CD ATC3T
ATC4CD ATC4T
ATC5CD ATC5T CMSTDY CMENDY CMDOSE CMROUTE CMDOSTXT CMDOSU CMDOSFRQ CMENRTPT
CMENPT EPOCH
DICTVER CMCLAS1 CMCLAS2 CMCLSCD1 CMCLSCD2 CMCODE DICTVER USUBJID SUBJID SITEID AGE
AGEU SEX
SEXN RACE RACEN ARACE ARACEN SAFFL ARM ARMCD ACTARM ACTARMCD TRTSDT TRTSTM
TRTSDTM TRTEDT
TRTETM TRTEDTM V01DT V02DT V03DT V04DT VAX101DT VAX10UDT VAX102DT VAX201DT

```

FDA-CBER-2022-5812-0735862


```

VAX202DT COHORTN
COHORT DOSALVLN DOSALVL DOSPLVLN DOSPLVL PHASEN PHASE UNBLNDDT MULENRFL
DS30KFL HIVFL
AGETR01 AGEGR1 AGEGR1N AGEGR4 AGEGR4N TR01SDTM TR01EDTM TR02SDTM TR02EDTM TRT01A
TRT01AN
TRT01P TRT01PN TRT02A TRT02AN TRT02P TRT02PN VAX101 VAX102 VAX10U VAX201 VAX202
VAX20U
VAX20UDT DS3KFL RANDFL RAND1FL SAF1FL SAF2FL VPHASE VPHASEN VAXNO);
label DRUGNAME = 'WHODrug preferred name' DRGNAME1 = "WHODrug preferred name1" ATC1CD = 'ATC
Level 1 Code' ATC1T = 'ATC Level 1 Text' ATC2CD = 'ATC Level 2 Code' ATC2T = 'ATC Level 2 Text' ATC3CD =
'ATC Level 3 Code' ATC3T = 'ATC Level 3 Text' ATC4CD = 'ATC Level 4 Code' ATC4T = 'ATC Level 4 Text'
ATC5CD = 'ATC Level 5 Code' ATC5T = 'ATC Level 5 Text' ASTDT = 'Analysis Start Date' ASTTM = 'Analysis Start
Time' ASTDTM = 'Analysis Start Date/Time' ASTTMF = 'Analysis Start Time Imputation Flag' ASTDTF = 'Analysis
Start Date Imputation Flag' ASTDY = 'Analysis Start Relative Day' AENDT = 'Analysis End Date' AENDTF =
'Analysis End Date Imputation Flag' AENDY = 'Analysis End Relative Day' PREFL = 'Pre-treatment Flag' ONPERFL =
'On-Period Flag' CMCODE = 'Standardized Medication Code' ;
drop ASTDTM ASTTM ASTTMF;
run;

proc printto;
run;

```