```
*****************************************************************************************;
** Program Name    : adsl-s005-all1-ped6-saf.sas                              **;
** Date Created    : 15Nov2021                                        **;
** Programmer Name : (b) (4), (b) (6)                                          **;
** Purpose         : Create adsl-s005-all1-ped6-saf                          **;
** Input data      : adsl                                     **;
** Output file     : adsl-s005-all1-ped6-saf.html                           **;
*****************************************************************************************;
options mprint mlogic symbolgen mprint symbolgen mlogic nocenter missing=" ";
ods escapechar="~";

proc datasets library=WORK kill nolist nodetails;
quit;

**Setup the environment**;
%let
bprot=/Volumes/app/cdars/prod/sites/cdars4/prjC459/nda2_unblinded_esub/sbla1215_esub_adam/saseng/cdisc3_0/;
%let prot=/Volumes/app/cdars/prod/sites/cdars4/prjC459/nda2_unblinded_esub/sbla1215_esub_adam/saseng/cdisc3_0;
%let codename=adsl-s005-all1-ped6-saf;

libname datvprot "&bprot.data_vai" access=readonly;
%let outlog=&prot./analysis/eSUB/logs/&codename..log;
%let outtable=&prot./analysis/eSUB/output/&codename..html;


proc printto log="&outlog." new;
run;

***********************************************************************************;
* Clean *;
***********************************************************************************;

proc delete data=work._all_;
run;

proc format;
      value cov 1="Positive" 2="Negative" other="Missing";
      value sars 1="Positive(*ESC*){super c}" 2="Negative(*ESC*){super d}"
            other="Missing";
      value cd 1="<200 cells/mm(*ESC*){super 3}"
            2="200-500 cells/mm(*ESC*){super 3}" 3=">500 cells/mm(*ESC*){super 3}";
      value rna 1="(*ESC*){Unicode 003C}50 copies/mL"
            2="(*ESC*){unicode 2265}50 copies/mL";
      value sex 1='Male' 2='Female';
      value arace 1='White' 2='Black or African American'
            3='American Indian or Alaska Native' 4='Asian'
            5='Native Hawaiian or other Pacific Islander' 6='Multiracial'
            7='Not reported' 8='Unknown' 999='All others~{super c}';
      value ethnic 1='Hispanic/Latino' 2='Non-Hispanic/non-Latino' 3='Not reported'
            4='Unknown';
      value RANDAGE 1='12-15 Years' 2='16-55 Years' 3='18-55 Years' 4='65-85 Years'
            5='>55 Years';
      value Raciald 1="Indian Subcontinent Asian" 10="African Caribbean"
```

```
          11="Saudi Arabian" 12="Malay" 13="Filipino" 14="Vietnamese"
          15="Australian Aboriginal" 16="Torres Strait Islander" 17="Han Chinese"
          18="Non-Han Chinese" 19="Ashkenazi Jew" 2="Southeast Asian"
          3="Far East Asian" 4="Japanese American" 5="Japanese" 6="Korean" 7="Chinese"
          8="African" 9="African American" 999="Other";
     value BMICAT 1="Underweight ((*ESC*){Unicode 003C}18.5 kg/m~{super 2})" 2=" Normal weight ((*ESC*)
{Unicode 2265}18.5 kg/m~{super 2} - 24.9 kg/m~{super 2})"
          3="Overweight ((*ESC*){Unicode 2265}25.0 kg/m~{super 2} - 29.9 kg/m~{super 2})"
          4="Obese ((*ESC*){Unicode 2265}30.0 kg/m~{super 2})" 5="Missing";
     value obes 1="Yes" 2="No";
run;

data adsl;
     set DATVPROT.ADSL(rename=(ethnic=ethnic1));
     length ethnic $50;

     if covblst="POS" then
          do;
               covblst="Positive";
               covblstc="Positive(*ESC*){super c}";
               covblstn=1;
          end;
     else if covblst="NEG" then
          do;
               covblst="Negative";
               covblstc="Negative(*ESC*){super d}";
               covblstn=2;
          end;

     /*     else          covblstn=.;*/
     else
          do;
               covblst="Missing";
               covblstc="Missing";
               covblstn=999;
          end;

     if upcase(ethnic1)='NOT HISPANIC OR LATINO' then
          ethnic='Non-Hispanic/Non-Latino';
     else if upcase(ethnic1)='HISPANIC OR LATINO' then
          ethnic='Hispanic/Latino';
     else if upcase(ethnic1)='NOT REPORTED' then
          ethnic='Not Reported';
run;

data adsl;
     set adsl;
     length countryx  $50;

     if country='ARG' then
          countryx='Argentina';
     else if country='BRA' then
          countryx='Brazil';
     else if country='DEU' then
```

```
          countryx='Germany';
     else if country='TUR' then
          countryx='Turkey';
     else if country='USA' then
          countryx='USA';
     else if country='ZAF' then
          countryx='South Africa';
     else
          countryx='Others';
run;

data adsl;
     set adsl;

     if trt01an=8 and agegr4n=1 then
          trtarn=1;
     else if trt01an=8 and agegr4n=2 then
          trtarn=2;
     else if trt01an=9 and agegr4n=1 then
          trtarn=3;
     else if trt01an=9 and agegr4n=2 then
          trtarn=4;
     trtar=trt01a;

     if COMBODFL='Y' or OBESEFL="Y" then
          do;
               COMBODFLNX=1;
               COMBODFLX="Yes";
          end;
     else
          do;
               COMBODFLNX=2;
               COMBODFLX="No";
          end;

     if obesefl="Y" then
          do;
               obeseflc="Yes";
               obesefln=1;
          end;
     else if obesefl="N" then
          do;
               obeseflc="No";
               obesefln=2;
          end;

     if racialdn=999 then
          racialdn=.;
run;

data g_adsl_dsin;
     set adsl;
     where SAFFL eq 'Y' and AGEGR4N=1 and phasen ne 1;
run;
```

```
data __trtmap;
    length trtcode trtdecd $100;

    if 0 then
        set g_adsl_dsin(keep=TRT01AN);
    trtval=1;

    if vtype(TRT01AN)='C' then
        trtcode=tranwrd(compbl(quote("8")), ' ', '" "');
    else
        trtcode="8";
    trtdecd="BNT162b2 (30 (*ESC*){unicode 03BC}g)";
    trtvar="TRT01AN";
    trtlbl="TRT01A";
    output;
    trtval=2;

    if vtype(TRT01AN)='C' then
        trtcode=tranwrd(compbl(quote("9")), ' ', '" "');
    else
        trtcode="9";
    trtdecd="Placebo";
    trtvar="TRT01AN";
    trtlbl="TRT01A";
    output;
    trtval=3;

    if vtype(TRT01AN)='C' then
        trtcode=tranwrd(compbl(quote("8 9")), ' ', '" "');
    else
        trtcode="8 9";
    trtdecd="Total";
    trtvar="TRT01AN";
    trtlbl="TRT01A";
    output;
    stop;
run;

data g_adsl_dsin;
    set g_adsl_dsin;

    if TRT01AN in (8) then
        do;
            newtrtn=1;
            newtrt=coalescec("BNT162b2 (30 (*ESC*){unicode 03BC}g)", TRT01A);
            output;
        end;

    if TRT01AN in (9) then
        do;
            newtrtn=2;
            newtrt=coalescec("Placebo", TRT01A);
            output;
```

```
          end;

     if TRT01AN in (8 9) then
          do;
               newtrtn=3;
               newtrt=coalescec("Total", TRT01A);
               output;
          end;
run;

*-------------------------------------------------------------------;
* Initialize dataset for non-pvalue footnote queue. ;
*-------------------------------------------------------------------;

data _stdft1(compress=no);
     length model $200 mark $5;
     index=0;
     model=' ';
     mark=' ';
run;

*-------------------------------------------------------------------;
* Initialize dataset for pvalue related footnote queue.;
*-------------------------------------------------------------------;

data _stdft2(compress=no);
     length model $200 mark $5;
     index=0;
     model=' ';
     mark=' ';
run;

*-------------------------------------------------------------------;
* Initialize structure for _BASETEMPLATE dataset. ;
*-------------------------------------------------------------------;

data _basetemplate(compress=no);
     length _varname $8 _cvalue $35 _direct $20 _vrlabel $200 _rwlabel
          _colabel $800 _datatyp $5 _module $8 _pr_lbl $ 200;
     array _c _character_;
     delete;
run;

*-------------------------------------------------------------------;
* Create next _DATAn dataset ;
*-------------------------------------------------------------------;

data _data1;
     set g_adsl_dsin;
     where (NEWTRTN is not missing);
run;

*-------------------------------------------------------------------;
* Count number of treatment groups ;
```

FDA-CBER-2022-5812-0072622

```
*-------------------------------------------------------------------;

proc sql noprint;
       select count(unique NEWTRTN) into :_trtn from _data1 where NEWTRTN is not
              missing;
quit;

*-------------------------------------------------------------------;
* Generate variable _TRT. Use assigned order if applicable ;
*-------------------------------------------------------------------;

proc sort data=_data1;
       by NEWTRTN USUBJID;
run;

data _data1;
       retain _trt 0;
       length _str $200;
       _datasrt=1;
       set _data1 end=eof;
       by NEWTRTN USUBJID;
       drop _str;
       _str=' ';
       _lastby=1;
       _dummyby=0;

       if first.NEWTRTN then
              do;

                     if not missing(NEWTRTN) then
                            do;
                                   _trt=_trt + 1;
                            end;
                     *-------------------------------------------------------------------;
                     * Generate _STR as the treatment label ;
                     *-------------------------------------------------------------------;
                     _str=NEWTRT;
                     *-------------------------------------------------------------------;
                     * Update _TRTLB&n with generated treatment label ;
                     *-------------------------------------------------------------------;

                     if _trt > 0 then
                            call symput('_trtlb'||compress(put(_trt, 4.)), trim(left(_str)));
              end;
run;

*-------------------------------------------------------------------;
* Count number of patients in each treatment. ;
*-------------------------------------------------------------------;

proc sql noprint;
       select compress(put(count(*), 5.) ) into :_trt1 - :_trt3 from (select distinct
              USUBJID, _trt from _data1 where NEWTRTN is not missing) group by _trt;
       select compress(put(count(*), 5.) ) into :_trt4 from (select distinct USUBJID
          from _data1 where NEWTRTN is not missing);
quit;

*-------------------------------------------------------------------;
* Generate a dataset containing all by-variables ;
*-------------------------------------------------------------------;

proc sort data=_data1 out=_bydat1(keep=_datasrt _dummyby) nodupkey;
     by _datasrt;
run;

data _bydat1;
     set _bydat1 end=eof;
     by _datasrt;
     retain _preby 0;
     drop _preby;
     _byvar1=0;

     if eof then
          do;
               call symput("_preby1", compress(put(_byvar1, 4.)));

               if 0=0 then
                    output;
          end;
run;

data _bydat1;
     set _bydat1;
     by _datasrt;
     length _bycol _byindnt $50 _bylast $10;
     _bycol=" ";
     _byindnt=" ";
     _bylast=" ";
run;

proc sort data=_bydat1;
     by _datasrt;
run;

proc sort data=_data1 out=_data1;
     by _datasrt;
run;

data _anal1;
     length SEXN 8;
     set _data1;

     if SEXN=. then
          SEXN=9998;
     _blcksrt=1;
     _cnt=1;
     _cat=1;
```

FDA-CBER-2022-5812-0072624

```
    if _trt <=0 then
        delete;
    output;
run;

proc sort data=_anal1;
    by _datasrt _blcksrt SEXN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp1;
    set _anal1;
    output;
run;

proc sort data=_temp1 out=_temp91 nodupkey;
    by _datasrt _blcksrt _cat SEXN _trt USUBJID;
run;

proc freq data=_temp91;
    format SEXN;
    tables _datasrt*_blcksrt*_cat * SEXN * _trt / sparse norow nocol nopercent
        out=_pct1(drop=percent);
run;

proc sort data=_anal1 out=_denom1(keep=_datasrt _cat) nodupkey;
    by _datasrt _cat;
run;

data _denom1;
    set _denom1;
    by _datasrt _cat;
    label count='count';
    _trt=1;
    count=&_trt1;
    output;
    _trt=2;
    count=&_trt2;
    output;
    _trt=3;
    count=&_trt3;
    output;
run;

*-------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-------------------------------------------------------------------;

data _denomf1;
    _datasrt=1;
    set _bydat1(keep=);
    * All treatment groups ;
    _trt1=0;
```

FDA-CBER-2022-5812-0072625

```
        _trt2=0;
        _trt3=0;
        * _CAT is the subgroup variable ;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*-------------------------------------------------------------------;

proc sql noprint;
        select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
                where (libname="WORK" and memname="_DENOM1");
        select setting into :miss from dictionary.options where
                upcase(optname)="MISSING";
quit;

proc transpose data=_denom1 out=_denomin1(drop=_name_ _label_) prefix=_trt;
        by _datasrt _cat;
        var count;
        id _trt;
run;

*-------------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
*-------------------------------------------------------------------;

proc sql noprint;
        select count(distinct SEXN) into : totexpv from _anal1;
        select distinct SEXN into :expv1 - :expv2 from _anal1 order by SEXN;
quit;

*-------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*-------------------------------------------------------------------;

data _frame1;
        _datasrt=1;
        set _bydat1(keep=);
        _blcksrt=1;
        length SEXN 8;
        _catLabl=" ";
        _trt=1;
        SEXN=1;
        _catord=1;
        _cat=1;
        output;
        _trt=2;
        SEXN=1;
        _catord=1;
        _cat=1;
        output;
        _trt=3;
```

FDA-CBER-2022-5812-0072626

```
        SEXN=1;
        _catord=1;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        SEXN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=2;
        SEXN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=3;
        SEXN=2;
        _catord=2;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame1;
        by _datasrt _blcksrt _cat SEXN _trt;
run;

proc sort data=_pct1;
        by _datasrt _blcksrt _cat SEXN _trt;
run;

data _pct1;
        merge _frame1(in=_inframe) _pct1;
        by _datasrt _blcksrt _cat SEXN _trt;

        if _inframe;

        if count=. then
                count=0;
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct1;
        by _datasrt _blcksrt SEXN;
run;

data _miss1(keep=_datasrt _blcksrt SEXN totcount);
```

```
        set _pct1;
        where SEXN=9998;
        retain totcount;
        by _datasrt _blcksrt SEXN;

        if first.SEXN then
                totcount=0;
        totcount=totcount+count;

        if last.SEXN;
run;

data _pct1(drop=totcount);
        merge _pct1 _miss1;
        by _datasrt _blcksrt SEXN;

        if totcount=0 then
                delete;
run;

*****************************************************************;
*IF PCTDISP=CAT/DPTVAR then add dptvar into denomitor frame dataset;
*****************************************************************;
*-------------------------------------------------------------------;
* Merge the _DENOMIN with its frame up dataset (_denomf) ;
*-------------------------------------------------------------------;

proc sort data=_denomf1;
        by _datasrt _cat;
run;

proc sort data=_denomin1;
        by _datasrt _cat;
run;

data _denomin1;
        merge _denomf1(in=_inframe) _denomin1;
        by _datasrt _cat;

        if _inframe;
        _blcksrt=1;
run;

*-------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*-------------------------------------------------------------------;

proc sort data=_pct1;
        by _datasrt _cat;
run;

*-------------------------------------------------------------------;
* Create _VARNAME variable to hold depend variable name. ;
* Create _VRLABEL variable to display Group label. ;
* Create _RWLABEL variable to display &dptvar categories. ;
*------------------------------------------------------------------;

data _pct1;
        if 0 then
                set _basetemplate;
        merge _denomin1(in=_a) _pct1;
        by _datasrt _cat;

        if _a;
        _varname="SEXN ";
        _vrlabel="Sex ";
        _rwlabel=put(SEXN, sex.);

        if SEXN=9998 then
                do;
                        _rwlabel="Unknown ";
                        _catord=9998;
                end;
        else if SEXN=9999 then
                do;
                        _rwlabel="Total ";
                        _catord=9999;
                end;

        if _catord=. then
                _catord=9997;
run;

proc sort data=_pct1;
        by _datasrt _blcksrt _catord SEXN _trt _cat;
run;

*------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*------------------------------------------------------------------;

data _base1;
        length _catlabl $200;
        set _pct1 end=eof;
        by _datasrt _blcksrt _catord SEXN _trt _cat;
        retain _rowsrt 0 _rowmax 0;
        array _trtcnt(*) _trt1-_trt4;
        drop _rowmax _cpct;
        length _cpct $100;
        _cpct=' ';
        _module='mcatstat';

        if count > . then
                _cvalue=put(count, 5.);
        else
                _cvalue=put(0, 5.);
        *------------------------------------------------------------------;
```

```
    * Format percent to append to display value in _CVALUE ;
    *-------------------------------------------------------------------;

    if _trt ne . then
        do;

            if _trtcnt(_trt) > 0 then
                do;
                    percent=count / _trtcnt(_trt) * 100;

                    if percent > 0 then
                        do;

                            if round(percent, 0.1) GE 0.1 then
                                _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                            else
                                _cpct="(*ESC*){nbspace 1}(0.0)";
                            _cvalue=trim(_cvalue)||_cpct;
                        end;
                end;
        end;

    if length(_cvalue) < 13 then
        do;
            *-------------------------------------------------------------------;
            * Put character A0x at right most character to pad text;
            *-------------------------------------------------------------------;
            substr(_cvalue, 13, 1)='A0'x;
        end;

    if first.SEXN then
        do;
            _rowsrt=_rowsrt + 1;
            _rowmax=max(_rowsrt, _rowmax);
        end;
    _datatyp='data';
    _indent=0;
    _dptindt=0;
    _vorder=1;
    _rowjump=1;

    if upcase(_rwlabel)='_NONE_' then
        _rwlabel=' ';
    _indent=3;
    _dptindt=0;

    if _trt=3 +1 then
        _trt=9999;

    if eof then
        call symput('_rowsrt', compress(put(_rowmax, 4.)));
    _direct="TOP ";
    _p=2;
run;
```

FDA-CBER-2022-5812-0072630

```
data _anal2;
        length ARACEN 8;
        set _data1;
        where same and ARACEN is not missing;
        _blcksrt=2;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal2;
        by _datasrt _blcksrt ARACEN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp2;
        set _anal2;
        output;
run;

proc sort data=_temp2 out=_temp92 nodupkey;
        by _datasrt _blcksrt _cat ARACEN _trt USUBJID;
        ;
run;

proc freq data=_temp92;
        format ARACEN;
        tables _datasrt*_blcksrt*_cat * ARACEN * _trt / sparse norow nocol nopercent
                out=_pct2(drop=percent);
run;

proc sort data=_anal2 out=_denom2(keep=_datasrt _cat) nodupkey;
        by _datasrt _cat;
run;

data _denom2;
        set _denom2;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;
```

```
*------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*------------------------------------------------------------------;

data _denomf2;
      _datasrt=1;
      set _bydat1(keep=);
      * All treatment groups ;
      _trt1=0;
      _trt2=0;
      _trt3=0;
      * _CAT is the subgroup variable ;
      _cat=1;
      output;
run;

*------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*------------------------------------------------------------------;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_DENOM2");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

;

proc transpose data=_denom2 out=_denomin2(drop=_name_ _label_) prefix=_trt;
      by _datasrt _cat;
      var count;
      id _trt;
run;

*------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*------------------------------------------------------------------;

data _frame2;
      _datasrt=1;
      set _bydat1(keep=);
      _blcksrt=2;
      length ARACEN 8;
      _catLabl=" ";
      _trt=1;
      ARACEN=1;
      _catord=1;
      _cat=1;
      output;
      _trt=2;
      ARACEN=1;
      _catord=1;
```

FDA-CBER-2022-5812-0072632

```
        _cat=1;
        output;
        _trt=3;
        ARACEN=1;
        _catord=1;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        ARACEN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=2;
        ARACEN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=3;
        ARACEN=2;
        _catord=2;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame2;
        by _datasrt _blcksrt _cat ARACEN _trt;
run;

proc sort data=_pct2;
        by _datasrt _blcksrt _cat ARACEN _trt;
run;

data _pct2;
        merge _frame2(in=_inframe) _pct2;
        by _datasrt _blcksrt _cat ARACEN _trt;

        if _inframe;

        if count=. then
                count=0;
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct2;
        by _datasrt _blcksrt ARACEN;
```

```
run;

data _miss2(keep=_datasrt _blcksrt ARACEN totcount);
      set _pct2;
      where ARACEN=9998;
      retain totcount;
      by _datasrt _blcksrt ARACEN;

      if first.ARACEN then
            totcount=0;
      totcount=totcount+count;

      if last.ARACEN;
run;

data _pct2(drop=totcount);
      merge _pct2 _miss2;
      by _datasrt _blcksrt ARACEN;

      if totcount=0 then
            delete;
run;

proc sort data=_denomf2;
      by _datasrt _cat;
run;

proc sort data=_denomin2;
      by _datasrt _cat;
run;

data _denomin2;
      merge _denomf2(in=_inframe) _denomin2;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=2;
run;

*-------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*-------------------------------------------------------------------;

proc sort data=_pct2;
      by _datasrt _cat;
run;

data _pct2;
      if 0 then
            set _basetemplate;
      merge _denomin2(in=_a) _pct2;
      by _datasrt _cat;

      if _a;
```

```
        _varname="ARACEN ";
        _vrlabel="Race ";
        _rwlabel=put(ARACEN, arace.);

        if ARACEN=9998 then
                do;
                        _rwlabel="Missing ";
                        _catord=9998;
                end;
        else if ARACEN=9999 then
                do;
                        _rwlabel="Total ";
                        _catord=9999;
                end;

        if _catord=. then
                _catord=9997;
run;

proc sort data=_pct2;
        by _datasrt _blcksrt _catord ARACEN _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base2;
        length _catlabl $200;
        set _pct2 end=eof;
        by _datasrt _blcksrt _catord ARACEN _trt _cat;
        retain _rowsrt 0 _rowmax 0;
        array _trtcnt(*) _trt1-_trt4;
        drop _rowmax _cpct;
        length _cpct $100;
        _cpct=' ';
        _module='mcatstat';

        if count > . then
                _cvalue=put(count, 5.);
        else
                _cvalue=put(0, 5.);
        *-------------------------------------------------------------------;
        * Format percent to append to display value in _CVALUE ;
        *-------------------------------------------------------------------;

        if _trt ne . then
                do;

                        if _trtcnt(_trt) > 0 then
                                do;
                                        percent=count / _trtcnt(_trt) * 100;
                            if percent > 0 then
                                do;

                                    if round(percent, 0.1) GE 0.1 then
                                        _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                    else
                                        _cpct="(*ESC*){nbspace 1}(0.0)";
                                    _cvalue=trim(_cvalue)||_cpct;
                                end;
                                end;
                end;

        /* if length(_cvalue) < 13 then do; */
        *--------------------------------------------------------------------;
        * Put character A0x at right most character to pad text;
        *--------------------------------------------------------------------;

        /* substr(_cvalue,13,1)= 'A0'x ; */
        /* end; */
        if first.ARACEN then
                do;
                        _rowsrt=_rowsrt + 1;
                        _rowmax=max(_rowsrt, _rowmax);
                end;
        _datatyp='data';
        _indent=0;
        _dptindt=0;
        _vorder=1;
        _rowjump=1;

        if upcase(_rwlabel)='_NONE_' then
                _rwlabel=' ';
        _indent=3;
        _dptindt=0;

        if _trt=3 +1 then
                _trt=9999;

        if eof then
                call symput('_rowsrt', compress(put(_rowmax, 4.)));
        _direct="TOP ";
        _p=2;
run;

data _data1;
        set _data1;

        if Aracen in (3, 4, 5, 6, 7, 8) then
                newrace="Y";
run;

data _anal3;
        length NEWRACE $4;
        set _data1;
```

```
        where same and NEWRACE is not missing;
        _blcksrt=2;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal3;
        by _datasrt _blcksrt NEWRACE _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp3;
        set _anal3;
        output;
run;

proc sort data=_temp3 out=_temp93 nodupkey;
        by _datasrt _blcksrt _cat NEWRACE _trt USUBJID;
run;

proc freq data=_temp93;
        format NEWRACE;
        tables _datasrt*_blcksrt*_cat * NEWRACE * _trt / sparse norow nocol nopercent
                out=_pct3(drop=percent);
run;

proc sort data=_anal3 out=_denom3(keep=_datasrt _cat) nodupkey;
        by _datasrt _cat;
run;

data _denom3;
        set _denom3;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;

*------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*------------------------------------------------------------------;
```

```
data _denomf3;
      _datasrt=1;
      set _bydat1(keep=);
      * All treatment groups ;
      _trt1=0;
      _trt2=0;
      _trt3=0;
      * _CAT is the subgroup variable ;
      _cat=1;
      output;
run;

*-------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*-------------------------------------------------------------------;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_DENOM3");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

proc transpose data=_denom3 out=_denomin3(drop=_name_ _label_) prefix=_trt;
      by _datasrt _cat;
      var count;
      id _trt;
run;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_PCT3");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

;

proc sort data=_pct3 out=_expv3 (keep=_datasrt _blcksrt NEWRACE) nodupkey;
      by _datasrt _blcksrt NEWRACE;
run;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_PCT3");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

;

proc sort data=_expv3;
      by _datasrt _blcksrt NEWRACE;
run;
```

```
data _frame3;
       set _expv3;
       by _datasrt _blcksrt NEWRACE;

       if first._blcksrt then
              _catord=0;
       _catord + 1;
       _trt=1;
       _cat=1;
       output;
       _trt=2;
       _cat=1;
       output;
       _trt=3;
       _cat=1;
       output;
run;

proc sort data=_frame3;
       by _datasrt _blcksrt _cat NEWRACE _trt;
run;

proc sort data=_pct3;
       by _datasrt _blcksrt _cat NEWRACE _trt;
run;

data _pct3;
       merge _frame3(in=_inframe) _pct3;
       by _datasrt _blcksrt _cat NEWRACE _trt;

       if _inframe;

       if count=. then
              count=0;
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct3;
       by _datasrt _blcksrt NEWRACE;
run;

data _miss3(keep=_datasrt _blcksrt NEWRACE totcount);
       set _pct3;
       where NEWRACE='ZZZY';
       retain totcount;
       by _datasrt _blcksrt NEWRACE;

       if first.NEWRACE then
              totcount=0;
```

```
      totcount=totcount+count;

      if last.NEWRACE;
run;

data _pct3(drop=totcount);
      merge _pct3 _miss3;
      by _datasrt _blcksrt NEWRACE;

      if totcount=0 then
            delete;
run;

*****************************************************************;
*IF PCTDISP=CAT/DPTVAR then add dptvar into denomitor frame dataset;
*****************************************************************;
*-------------------------------------------------------------------;
* Merge the _DENOMIN with its frame up dataset (_denomf) ;
*-------------------------------------------------------------------;

proc sort data=_denomf3;
      by _datasrt _cat;
run;

proc sort data=_denomin3;
      by _datasrt _cat;
run;

data _denomin3;
      merge _denomf3(in=_inframe) _denomin3;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=2;
run;

*-------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*-------------------------------------------------------------------;

proc sort data=_pct3;
      by _datasrt _cat;
run;

*-------------------------------------------------------------------;
* Create _VARNAME variable to hold depend variable name. ;
* Create _VRLABEL variable to display Group label. ;
* Create _RWLABEL variable to display &dptvar categories. ;
*-------------------------------------------------------------------;

data _pct3;
      if 0 then
            set _basetemplate;
      merge _denomin3(in=_a) _pct3;
```

```
        by _datasrt _cat;

        if _a;
        _varname="NEWRACE ";
        _vrlabel="Race ";
        _rwlabel="All others ";

        if NEWRACE='ZZZY' then
                do;
                        _rwlabel="Missing ";
                        _catord=9998;
                end;
        else if NEWRACE='ZZZZ' then
                do;
                        _rwlabel="Total ";
                        _catord=9999;
                end;

        if _catord=. then
                _catord=9997;
run;

proc sort data=_pct3;
        by _datasrt _blcksrt _catord NEWRACE _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base3;
        length _catlabl $200;
        set _pct3 end=eof;
        by _datasrt _blcksrt _catord NEWRACE _trt _cat;
        retain _rowsrt 2 _rowmax 0;
        array _trtcnt(*) _trt1-_trt4;
        drop _rowmax _cpct;
        length _cpct $100;
        _cpct=' ';
        _module='mcatstat';

        if count > . then
                _cvalue=put(count, 5.);
        else
                _cvalue=put(0, 5.);
        *-------------------------------------------------------------------;
        * Format percent to append to display value in _CVALUE ;
        *-------------------------------------------------------------------;

        if _trt ne . then
                do;

                        if _trtcnt(_trt) > 0 then
                do;
                    percent=count / _trtcnt(_trt) * 100;

                    if percent > 0 then
                        do;

                            if round(percent, 0.1) GE 0.1 then
                                _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                            else
                                _cpct="(*ESC*){nbspace 1}(0.0)";
                            _cvalue=trim(_cvalue)||_cpct;
                        end;
                end;
        end;

    /* if length(_cvalue) < 13 then do; */
    *-------------------------------------------------------------------------;
    * Put character A0x at right most character to pad text;
    *-------------------------------------------------------------------------;

    /* substr(_cvalue,13,1)= 'A0'x ; */
    /* end; */
    if first.NEWRACE then
        do;
            _rowsrt=_rowsrt + 1;
            _rowmax=max(_rowsrt, _rowmax);
        end;
    _datatyp='data';
    _indent=0;
    _dptindt=0;
    _vorder=1;
    _rowjump=1;

    if upcase(_rwlabel)='_NONE_' then
        _rwlabel=' ';
    _indent=3;
    _dptindt=0;

    if _trt=3 +1 then
        _trt=9999;

    if eof then
        call symput('_rowsrt', compress(put(_rowmax, 4.)));
    _direct="TOP ";
    _p=2;
run;

data _anal4;
    length ARACEN 8;
    set _data1;
    where same and ARACEN is not missing;
    _blcksrt=2;
    _cnt=1;
    _cat=1;
```

```
        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal4;
        by _datasrt _blcksrt ARACEN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp4;
        set _anal4;
        output;
run;

proc sort data=_temp4 out=_temp94 nodupkey;
        by _datasrt _blcksrt _cat ARACEN _trt USUBJID;
        ;
run;

proc freq data=_temp94;
        format ARACEN;
        tables _datasrt*_blcksrt*_cat * ARACEN * _trt / sparse norow nocol nopercent
                out=_pct4(drop=percent);
run;

proc sort data=_anal4 out=_denom4(keep=_datasrt _cat) nodupkey;
        ;
        by _datasrt _cat;
run;

data _denom4;
        set _denom4;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        ;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;

*-------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-------------------------------------------------------------------;

data _denomf4;
```

```
        _datasrt=1;
        set _bydat1(keep=);
        * All treatment groups ;
        _trt1=0;
        _trt2=0;
        _trt3=0;
        * _CAT is the subgroup variable ;
        _cat=1;
        output;
run;

*------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*------------------------------------------------------------------;

proc sql noprint;
        select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
                where (libname="WORK" and memname="_DENOM4");
        select setting into :miss from dictionary.options where
                upcase(optname)="MISSING";
quit;

;

proc transpose data=_denom4 out=_denomin4(drop=_name_ _label_) prefix=_trt;
        by _datasrt _cat;
        var count;
        id _trt;
run;

*------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*------------------------------------------------------------------;

data _frame4;
        _datasrt=1;
        set _bydat1(keep=);
        _blcksrt=2;
        length ARACEN 8;
        _catLabl=" ";
        _trt=1;
        ARACEN=3;
        _catord=1;
        _cat=1;
        output;
        _trt=2;
        ARACEN=3;
        _catord=1;
        _cat=1;
        output;
        _trt=3;
        ARACEN=3;
        _catord=1;
        _cat=1;
```

```
output;
_catLabl=" ";
_trt=1;
ARACEN=4;
_catord=2;
_cat=1;
output;
_trt=2;
ARACEN=4;
_catord=2;
_cat=1;
output;
_trt=3;
ARACEN=4;
_catord=2;
_cat=1;
output;
_catLabl=" ";
_trt=1;
ARACEN=5;
_catord=3;
_cat=1;
output;
_trt=2;
ARACEN=5;
_catord=3;
_cat=1;
output;
_trt=3;
ARACEN=5;
_catord=3;
_cat=1;
output;
_catLabl=" ";
_trt=1;
ARACEN=6;
_catord=4;
_cat=1;
output;
_trt=2;
ARACEN=6;
_catord=4;
_cat=1;
output;
_trt=3;
ARACEN=6;
_catord=4;
_cat=1;
output;
_catLabl=" ";
_trt=1;
ARACEN=7;
_catord=5;
_cat=1;
```

```
        output;
        _trt=2;
        ARACEN=7;
        _catord=5;
        _cat=1;
        output;
        _trt=3;
        ARACEN=7;
        _catord=5;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        ARACEN=8;
        _catord=6;
        _cat=1;
        output;
        _trt=2;
        ARACEN=8;
        _catord=6;
        _cat=1;
        output;
        _trt=3;
        ARACEN=8;
        _catord=6;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame4;
        by _datasrt _blcksrt _cat ARACEN _trt;
run;

proc sort data=_pct4;
        by _datasrt _blcksrt _cat ARACEN _trt;
run;

data _pct4;
        merge _frame4(in=_inframe) _pct4;
        by _datasrt _blcksrt _cat ARACEN _trt;

        if _inframe;

        if count=. then
                count=0;
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
```

FDA-CBER-2022-5812-0072646

```
*-------------------------------------------------------------------;

proc sort data=_pct4;
      by _datasrt _blcksrt ARACEN;
run;

data _miss4(keep=_datasrt _blcksrt ARACEN totcount);
      set _pct4;
      where ARACEN=9998;
      retain totcount;
      by _datasrt _blcksrt ARACEN;

      if first.ARACEN then
            totcount=0;
      totcount=totcount+count;

      if last.ARACEN;
run;

data _pct4(drop=totcount);
      merge _pct4 _miss4;
      by _datasrt _blcksrt ARACEN;

      if totcount=0 then
            delete;
run;

*******************************************************************;
*IF PCTDISP=CAT/DPTVAR then add dptvar into denomitor frame dataset;
*******************************************************************;
*-------------------------------------------------------------------;
* Merge the _DENOMIN with its frame up dataset (_denomf) ;
*-------------------------------------------------------------------;

proc sort data=_denomf4;
      by _datasrt _cat;
run;

proc sort data=_denomin4;
      by _datasrt _cat;
run;

data _denomin4;
      merge _denomf4(in=_inframe) _denomin4;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=2;
run;

*-------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*-------------------------------------------------------------------;
```

```
proc sort data=_pct4;
      by _datasrt _cat;
run;

*---------------------------------------------------------------------;
* Create _VARNAME variable to hold depend variable name. ;
* Create _VRLABEL variable to display Group label. ;
* Create _RWLABEL variable to display &dptvar categories. ;
*---------------------------------------------------------------------;

data _pct4;
      if 0 then
            set _basetemplate;
      merge _denomin4(in=_a) _pct4;
      by _datasrt _cat;

      if _a;
      _varname="ARACEN ";
      _vrlabel="Race ";
      _rwlabel=put(ARACEN, arace.);

      if ARACEN=9998 then
            do;
                  _rwlabel="Missing ";
                  _catord=9998;
            end;
      else if ARACEN=9999 then
            do;
                  _rwlabel="Total ";
                  _catord=9999;
            end;

      if _catord=. then
            _catord=9997;
run;

proc sort data=_pct4;
      by _datasrt _blcksrt _catord ARACEN _trt _cat;
run;

*---------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*---------------------------------------------------------------------;

data _base4;
      length _catlabl $200;
      set _pct4 end=eof;
      by _datasrt _blcksrt _catord ARACEN _trt _cat;
      retain _rowsrt 3 _rowmax 0;
      array _trtcnt(*) _trt1-_trt4;
      drop _rowmax _cpct;
      length _cpct $100;
      _cpct=' ';
```

FDA-CBER-2022-5812-0072648

```
_module='mcatstat';

if count > . then
       _cvalue=put(count, 5.);
else
       _cvalue=put(0, 5.);
*-------------------------------------------------------------------;
* Format percent to append to display value in _CVALUE ;
*-------------------------------------------------------------------;

if _trt ne . then
       do;

              if _trtcnt(_trt) > 0 then
                     do;
                            percent=count / _trtcnt(_trt) * 100;

                            if percent > 0 then
                                   do;

                                          if round(percent, 0.1) GE 0.1 then
                                                 _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                          else
                                                 _cpct="(*ESC*){nbspace 1}(0.0)";
                                          _cvalue=trim(_cvalue)||_cpct;
                                   end;
                     end;
       end;

/* if length(_cvalue) < 13 then do; */
*-------------------------------------------------------------------;
* Put character A0x at right most character to pad text;
*-------------------------------------------------------------------;

/* substr(_cvalue,13,1)= 'A0'x ; */
/* end; */
if first.ARACEN then
       do;
              _rowsrt=_rowsrt + 1;
              _rowmax=max(_rowsrt, _rowmax);
       end;
_datatyp='data';
_indent=0;
_dptindt=0;
_vorder=1;
_rowjump=1;

if upcase(_rwlabel)='_NONE_' then
       _rwlabel=' ';
_indent=6;
_dptindt=0;

if _trt=3 +1 then
       _trt=9999;
```

```
    if eof then
        call symput('_rowsrt', compress(put(_rowmax, 4.)));
    _direct="TOP ";
    _p=2;
run;

proc sql;
    create table nozero as select * , sum(count) as sum from _base4 group by
        _rwlabel having sum>0;
quit;

data _base4;
    set nozero;
run;

********************************************************************************;
*SPECIFICATION 5 -1) RACIALD - n and percent when RACIALD exists *;
********************************************************************************;

data _anal5;
    length RACIALDN 8;
    set _data1;
    where same and RACIALDN is not missing;
    _blcksrt=3;
    _cnt=1;
    _cat=1;

    if _trt <=0 then
        delete;
    output;
run;

proc sort data=_anal5;
    by _datasrt _blcksrt RACIALDN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp5;
    set _anal5;
    output;
run;

proc sort data=_temp5 out=_temp95 nodupkey;
    by _datasrt _blcksrt _cat RACIALDN _trt USUBJID;
    ;
run;

proc freq data=_temp95;
    format RACIALDN;
    tables _datasrt*_blcksrt*_cat * RACIALDN * _trt / sparse norow nocol nopercent
        out=_pct5(drop=percent);
run;
```

```
proc sort data=_anal5 out=_denom5(keep=_datasrt _cat) nodupkey;
        ;
        by _datasrt _cat;
run;

data _denom5;
        set _denom5;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;

*-----------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-----------------------------------------------------------------;

data _denomf5;
        _datasrt=1;
        set _bydat1(keep=);
        * All treatment groups ;
        _trt1=0;
        _trt2=0;
        _trt3=0;
        * _CAT is the subgroup variable ;
        _cat=1;
        output;
run;

proc sql noprint;
        select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
                where (libname="WORK" and memname="_DENOM5");
        select setting into :miss from dictionary.options where
                upcase(optname)="MISSING";
quit;

;

proc transpose data=_denom5 out=_denomin5(drop=_name_ _label_) prefix=_trt;
        by _datasrt _cat;
        var count;
        id _trt;
run;

*-----------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
```

```
*-------------------------------------------------------------------;

proc sql noprint;
      select count(distinct RACIALDN) into : totexpv from _anal5;
      select distinct RACIALDN into :expv1 - :expv1 from _anal5 order by RACIALDN;
quit;

*-------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*-------------------------------------------------------------------;

data _frame5;
      _datasrt=1;
      set _bydat1(keep=);
      _blcksrt=3;
      length RACIALDN 8;
      _catLabl=" ";
      _trt=1;
      RACIALDN=5;
      _catord=1;
      _cat=1;
      output;
      _trt=2;
      RACIALDN=5;
      _catord=1;
      _cat=1;
      output;
      _trt=3;
      RACIALDN=5;
      _catord=1;
      _cat=1;
      output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame5;
      by _datasrt _blcksrt _cat RACIALDN _trt;
run;

proc sort data=_pct5;
      by _datasrt _blcksrt _cat RACIALDN _trt;
run;

data _pct5;
      merge _frame5(in=_inframe) _pct5;
      by _datasrt _blcksrt _cat RACIALDN _trt;

      if _inframe;

      if count=. then
            count=0;
```

FDA-CBER-2022-5812-0072652

```
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct5;
      by _datasrt _blcksrt RACIALDN;
run;

data _miss5(keep=_datasrt _blcksrt RACIALDN totcount);
      set _pct5;
      where RACIALDN=9998;
      retain totcount;
      by _datasrt _blcksrt RACIALDN;

      if first.RACIALDN then
            totcount=0;
      totcount=totcount+count;

      if last.RACIALDN;
run;

data _pct5(drop=totcount);
      merge _pct5 _miss5;
      by _datasrt _blcksrt RACIALDN;

      if totcount=0 then
            delete;
run;

proc sort data=_denomf5;
      by _datasrt _cat;
run;

proc sort data=_denomin5;
      by _datasrt _cat;
run;

data _denomin5;
      merge _denomf5(in=_inframe) _denomin5;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=3;
run;

proc sort data=_pct5;
      by _datasrt _cat;
run;

data _pct5;
      if 0 then
```

```
		set _basetemplate;
	merge _denomin5(in=_a) _pct5;
	by _datasrt _cat;

	if _a;
	_varname="RACIALDN ";
	_vrlabel="Racial designation ";
	_rwlabel=put(RACIALDN, raciald.);

	if RACIALDN=9998 then
		do;
			_rwlabel="Missing ";
			_catord=9998;
		end;
	else if RACIALDN=9999 then
		do;
			_rwlabel="Total ";
			_catord=9999;
		end;

	if _catord=. then
		_catord=9997;
run;

proc sort data=_pct5;
	by _datasrt _blcksrt _catord RACIALDN _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base5;
	length _catlabl $200;
	set _pct5 end=eof;
	by _datasrt _blcksrt _catord RACIALDN _trt _cat;
	retain _rowsrt 0 _rowmax 0;
	array _trtcnt(*) _trt1-_trt4;
	drop _rowmax _cpct;
	length _cpct $100;
	_cpct=' ';
	_module='mcatstat';

	if count > . then
		_cvalue=put(count, 5.);
	else
		_cvalue=put(0, 5.);
	*-------------------------------------------------------------------;
	* Format percent to append to display value in _CVALUE ;
	*-------------------------------------------------------------------;

	if _trt ne . then
		do;
```

```
                if _trtcnt(_trt) > 0 then
                    do;
                        percent=count / _trtcnt(_trt) * 100;

                        if percent > 0 then
                            do;

                                if round(percent, 0.1) GE 0.1 then
                                    _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                else
                                    _cpct="(*ESC*){nbspace 1}(0.0)";
                                _cvalue=trim(_cvalue)||_cpct;
                            end;
                    end;
            end;

        if length(_cvalue) < 13 then
            do;
                *-------------------------------------------------------------------;
                * Put character A0x at right most character to pad text;
                *-------------------------------------------------------------------;
                substr(_cvalue, 13, 1)='A0'x;
            end;

        if first.RACIALDN then
            do;
                _rowsrt=_rowsrt + 1;
                _rowmax=max(_rowsrt, _rowmax);
            end;
        _datatyp='data';
        _indent=0;
        _dptindt=0;
        _vorder=1;
        _rowjump=1;

        if upcase(_rwlabel)='_NONE_' then
            _rwlabel=' ';
        _indent=3;
        _dptindt=0;

        if _trt=3 +1 then
            _trt=9999;

        if eof then
            call symput('_rowsrt', compress(put(_rowmax, 4.)));
        _direct="TOP ";
        _p=2;
run;

data _anal6;
        length ETHNICN 8;
        set _data1;
        where same and ETHNICN is not missing;
        _blcksrt=4;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal6;
        by _datasrt _blcksrt ETHNICN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp6;
        set _anal6;
        output;
run;

proc sort data=_temp6 out=_temp96 nodupkey;
        by _datasrt _blcksrt _cat ETHNICN _trt USUBJID;
        ;
run;

proc freq data=_temp96;
        format ETHNICN;
        tables _datasrt*_blcksrt*_cat * ETHNICN * _trt / sparse norow nocol nopercent
                out=_pct6(drop=percent);
run;

proc sort data=_anal6 out=_denom6(keep=_datasrt _cat) nodupkey;
        ;
        by _datasrt _cat;
run;

data _denom6;
        set _denom6;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;

*-------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-------------------------------------------------------------------;
```

```
data _denomf6;
        _datasrt=1;
        set _bydat1(keep=);
        * All treatment groups ;
        _trt1=0;
        _trt2=0;
        _trt3=0;
        * _CAT is the subgroup variable ;
        _cat=1;
        output;
run;

*---------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*---------------------------------------------------------------------;

proc sql noprint;
        select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
                where (libname="WORK" and memname="_DENOM6");
        select setting into :miss from dictionary.options where
                upcase(optname)="MISSING";
quit;

;

proc transpose data=_denom6 out=_denomin6(drop=_name_ _label_) prefix=_trt;
        by _datasrt _cat;
        var count;
        id _trt;
run;

*---------------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
*---------------------------------------------------------------------;

proc sql noprint;
        select count(distinct ETHNICN) into : totexpv from _anal6;
        select distinct ETHNICN into :expv1 - :expv3 from _anal6 order by ETHNICN;
quit;

*---------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*---------------------------------------------------------------------;

data _frame6;
        _datasrt=1;
        set _bydat1(keep=);
        _blcksrt=4;
        length ETHNICN 8;
        _catLabl=" ";
        _trt=1;
        ETHNICN=1;
        _catord=1;
```

FDA-CBER-2022-5812-0072657

```
        _cat=1;
        output;
        _trt=2;
        ETHNICN=1;
        _catord=1;
        _cat=1;
        output;
        _trt=3;
        ETHNICN=1;
        _catord=1;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        ETHNICN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=2;
        ETHNICN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=3;
        ETHNICN=2;
        _catord=2;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        ETHNICN=3;
        _catord=3;
        _cat=1;
        output;
        _trt=2;
        ETHNICN=3;
        _catord=3;
        _cat=1;
        output;
        _trt=3;
        ETHNICN=3;
        _catord=3;
        _cat=1;
        output;
run;

*--------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*--------------------------------------------------------------------;

proc sort data=_frame6;
        by _datasrt _blcksrt _cat ETHNICN _trt;
run;
```

```
proc sort data=_pct6;
      by _datasrt _blcksrt _cat ETHNICN _trt;
run;

data _pct6;
      merge _frame6(in=_inframe) _pct6;
      by _datasrt _blcksrt _cat ETHNICN _trt;

      if _inframe;

      if count=. then
            count=0;
run;

proc sort data=_pct6;
      by _datasrt _blcksrt ETHNICN;
run;

data _miss6(keep=_datasrt _blcksrt ETHNICN totcount);
      set _pct6;
      where ETHNICN=9998;
      retain totcount;
      by _datasrt _blcksrt ETHNICN;

      if first.ETHNICN then
            totcount=0;
      totcount=totcount+count;

      if last.ETHNICN;
run;

data _pct6(drop=totcount);
      merge _pct6 _miss6;
      by _datasrt _blcksrt ETHNICN;

      if totcount=0 then
            delete;
run;

proc sort data=_denomf6;
      by _datasrt _cat;
run;

proc sort data=_denomin6;
      by _datasrt _cat;
run;

data _denomin6;
      merge _denomf6(in=_inframe) _denomin6;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=4;
run;
```

```
proc sort data=_pct6;
      by _datasrt _cat;
run;

data _pct6;
      if 0 then
            set _basetemplate;
      merge _denomin6(in=_a) _pct6;
      by _datasrt _cat;

      if _a;
      _varname="ETHNICN ";
      _vrlabel="Ethnicity ";
      _rwlabel=put(ETHNICN, ethnic.);

      if ETHNICN=9998 then
            do;
                  _rwlabel="Missing ";
                  _catord=9998;
            end;
      else if ETHNICN=9999 then
            do;
                  _rwlabel="Total ";
                  _catord=9999;
            end;

      if _catord=. then
            _catord=9997;
run;

proc sort data=_pct6;
      by _datasrt _blcksrt _catord ETHNICN _trt _cat;
run;

*--------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*--------------------------------------------------------------------;

data _base6;
      length _catlabl $200;
      set _pct6 end=eof;
      by _datasrt _blcksrt _catord ETHNICN _trt _cat;
      retain _rowsrt 0 _rowmax 0;
      array _trtcnt(*) _trt1-_trt4;
      drop _rowmax _cpct;
      length _cpct $100;
      _cpct=' ';
      _module='mcatstat';

      if count > . then
            _cvalue=put(count, 5.);
      else
```

FDA-CBER-2022-5812-0072660

```
          _cvalue=put(0, 5.);
     *------------------------------------------------------------------;
     * Format percent to append to display value in _CVALUE ;
     *------------------------------------------------------------------;

     if _trt ne . then
          do;

               if _trtcnt(_trt) > 0 then
                    do;
                         percent=count / _trtcnt(_trt) * 100;

                         if percent > 0 then
                              do;

                                   if round(percent, 0.1) GE 0.1 then
                                        _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                   else
                                        _cpct="(*ESC*){nbspace 1}(0.0)";
                                   _cvalue=trim(_cvalue)||_cpct;
                              end;
                    end;
          end;

     /* if length(_cvalue) < 13 then do; */
     /*             *------------------------------------------------------------------; */
     /*             * Put character A0x at right most character to pad text; */
     /*             *------------------------------------------------------------------; */
     /*             substr(_cvalue, 13, 1)='A0'x; */
     /*        end; */
     if first.ETHNICN then
          do;
               _rowsrt=_rowsrt + 1;
               _rowmax=max(_rowsrt, _rowmax);
          end;
     _datatyp='data';
     _indent=0;
     _dptindt=0;
     _vorder=1;
     _rowjump=1;

     if upcase(_rwlabel)='_NONE_' then
          _rwlabel=' ';
     _indent=3;
     _dptindt=0;

     if _trt=3 +1 then
          _trt=9999;

     if eof then
          call symput('_rowsrt', compress(put(_rowmax, 4.)));
     _direct="TOP ";
     _p=2;
run;
```

```
data _anal7;
        length COUNTRYX $50;
        set _data1;
        where same and COUNTRYX is not missing;
        _blcksrt=5;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal7;
        by _datasrt _blcksrt COUNTRYX _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp7;
        set _anal7;
        output;
run;

proc sort data=_temp7 out=_temp97 nodupkey;
        by _datasrt _blcksrt _cat COUNTRYX _trt USUBJID;
run;

proc freq data=_temp97;
        format COUNTRYX;
        tables _datasrt*_blcksrt*_cat * COUNTRYX * _trt / sparse norow nocol nopercent
                out=_pct7(drop=percent);
run;

proc sort data=_anal7 out=_denom7(keep=_datasrt _cat) nodupkey;
        by _datasrt _cat;
run;

data _denom7;
        set _denom7;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;
```

FDA-CBER-2022-5812-0072662

```
*-------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-------------------------------------------------------------------;

data _denomf7;
      _datasrt=1;
      set _bydat1(keep=);
      * All treatment groups ;
      _trt1=0;
      _trt2=0;
      _trt3=0;
      * _CAT is the subgroup variable ;
      _cat=1;
      output;
run;

*-------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*-------------------------------------------------------------------;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_DENOM7");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

proc transpose data=_denom7 out=_denomin7(drop=_name_ _label_) prefix=_trt;
      by _datasrt _cat;
      var count;
      id _trt;
run;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_PCT7");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

proc sort data=_pct7 out=_expv7 (keep=_datasrt _blcksrt COUNTRYX) nodupkey;
      by _datasrt _blcksrt COUNTRYX;
run;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_PCT7");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

proc sort data=_expv7;
      by _datasrt _blcksrt COUNTRYX;
run;
```

FDA-CBER-2022-5812-0072663

```
data _frame7;
      set _expv7;
      by _datasrt _blcksrt COUNTRYX;

      if first._blcksrt then
            _catord=0;
      _catord + 1;
      _trt=1;
      _cat=1;
      output;
      _trt=2;
      _cat=1;
      output;
      _trt=3;
      _cat=1;
      output;
run;

*-----------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-----------------------------------------------------------------------;

proc sort data=_frame7;
      by _datasrt _blcksrt _cat COUNTRYX _trt;
run;

proc sort data=_pct7;
      by _datasrt _blcksrt _cat COUNTRYX _trt;
run;

data _pct7;
      merge _frame7(in=_inframe) _pct7;
      by _datasrt _blcksrt _cat COUNTRYX _trt;

      if _inframe;

      if count=. then
            count=0;
run;

proc sort data=_pct7;
      by _datasrt _blcksrt COUNTRYX;
run;

data _miss7(keep=_datasrt _blcksrt COUNTRYX totcount);
      set _pct7;
      where COUNTRYX='ZZZY';
      retain totcount;
      by _datasrt _blcksrt COUNTRYX;

      if first.COUNTRYX then
            totcount=0;
      totcount=totcount+count;
```

```
        if last.COUNTRYX;
run;

data _pct7(drop=totcount);
        merge _pct7 _miss7;
        by _datasrt _blcksrt COUNTRYX;

        if totcount=0 then
                delete;
run;

proc sort data=_denomf7;
        by _datasrt _cat;
run;

proc sort data=_denomin7;
        by _datasrt _cat;
run;

data _denomin7;
        merge _denomf7(in=_inframe) _denomin7;
        by _datasrt _cat;

        if _inframe;
        _blcksrt=5;
run;

proc sort data=_pct7;
        by _datasrt _cat;
run;

data _pct7;
        if 0 then
                set _basetemplate;
        merge _denomin7(in=_a) _pct7;
        by _datasrt _cat;

        if _a;
        _varname="COUNTRYX ";
        _vrlabel="Country ";
        _rwlabel=COUNTRYX;

        if COUNTRYX='ZZZY' then
                do;
                        _rwlabel="Missing ";
                        _catord=9998;
                end;
        else if COUNTRYX='ZZZZ' then
                do;
                        _rwlabel="Total ";
                        _catord=9999;
                end;
```

```
    if _catord=. then
        _catord=9997;
run;

proc sort data=_pct7;
    by _datasrt _blcksrt _catord COUNTRYX _trt _cat;
run;

*--------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*--------------------------------------------------------------------;

data _base7;
    length _catlabl $200;
    set _pct7 end=eof;
    by _datasrt _blcksrt _catord COUNTRYX _trt _cat;
    retain _rowsrt 0 _rowmax 0;
    array _trtcnt(*) _trt1-_trt4;
    drop _rowmax _cpct;
    length _cpct $100;
    _cpct=' ';
    _module='mcatstat';

    if count > . then
        _cvalue=put(count, 5.);
    else
        _cvalue=put(0, 5.);
    *--------------------------------------------------------------------;
    * Format percent to append to display value in _CVALUE ;
    *--------------------------------------------------------------------;

    if _trt ne . then
        do;

            if _trtcnt(_trt) > 0 then
                do;
                    percent=count / _trtcnt(_trt) * 100;

                    if percent > 0 then
                        do;

                            if round(percent, 0.1) GE 0.1 then
                                _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                            else
                                _cpct="(*ESC*){nbspace 1}(0.0)";
                            _cvalue=trim(_cvalue)||_cpct;
                        end;
                end;
        end;

    if length(_cvalue) < 13 then
        do;
            *--------------------------------------------------------------------;
```

```
                * Put character A0x at right most character to pad text;
                *------------------------------------------------------------------;
                substr(_cvalue, 13, 1)='A0'x;
            end;

        if first.COUNTRYX then
            do;
                _rowsrt=_rowsrt + 1;
                _rowmax=max(_rowsrt, _rowmax);
            end;
        _datatyp='data';
        _indent=0;
        _dptindt=0;
        _vorder=1;
        _rowjump=1;

        if upcase(_rwlabel)='_NONE_' then
            _rwlabel=' ';
        _indent=3;
        _dptindt=0;

        if _trt=3 +1 then
            _trt=9999;

        if eof then
            call symput('_rowsrt', compress(put(_rowmax, 4.)));
        _direct="TOP ";
        _p=2;
run;

data _anal8;
        length COVBLSTN 8;
        set _data1;

        if COVBLSTN=. then
            COVBLSTN=9998;
        _blcksrt=6;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
            delete;
        output;
run;

proc sort data=_anal8;
        by _datasrt _blcksrt COVBLSTN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp8;
        set _anal8;
        output;
```

```
run;

proc sort data=_temp8 out=_temp98 nodupkey;
      by _datasrt _blcksrt _cat COVBLSTN _trt USUBJID;
run;

proc freq data=_temp98;
      format COVBLSTN;
      tables _datasrt*_blcksrt*_cat * COVBLSTN * _trt / sparse norow nocol nopercent
            out=_pct8(drop=percent);
run;

proc sort data=_anal8 out=_denom8(keep=_datasrt _cat) nodupkey;
      ;
      by _datasrt _cat;
run;

data _denom8;
      set _denom8;
      by _datasrt _cat;
      label count='count';
      _trt=1;
      count=&_trt1;
      output;
      _trt=2;
      count=&_trt2;
      output;
      _trt=3;
      count=&_trt3;
      output;
run;

*-------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-------------------------------------------------------------------;

data _denomf8;
      _datasrt=1;
      set _bydat1(keep=);
      * All treatment groups ;
      _trt1=0;
      _trt2=0;
      _trt3=0;
      * _CAT is the subgroup variable ;
      _cat=1;
      output;
run;

*-------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*-------------------------------------------------------------------;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
```

```
          where (libname="WORK" and memname="_DENOM8");
     select setting into :miss from dictionary.options where
          upcase(optname)="MISSING";
quit;

proc transpose data=_denom8 out=_denomin8(drop=_name_ _label_) prefix=_trt;
     by _datasrt _cat;
     var count;
     id _trt;
run;

*----------------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
*----------------------------------------------------------------------;

proc sql noprint;
     select count(distinct COVBLSTN) into : totexpv from _anal8;
     select distinct COVBLSTN into :expv1 - :expv3 from _anal8 order by COVBLSTN;
quit;

*----------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*----------------------------------------------------------------------;

data _frame8;
     _datasrt=1;
     set _bydat1(keep=);
     _blcksrt=6;
     length COVBLSTN 8;
     _catLabl=" ";
     _trt=1;
     COVBLSTN=1;
     _catord=1;
     _cat=1;
     output;
     _trt=2;
     COVBLSTN=1;
     _catord=1;
     _cat=1;
     output;
     _trt=3;
     COVBLSTN=1;
     _catord=1;
     _cat=1;
     output;
     _catLabl=" ";
     _trt=1;
     COVBLSTN=2;
     _catord=2;
     _cat=1;
     output;
     _trt=2;
     COVBLSTN=2;
     _catord=2;
```

FDA-CBER-2022-5812-0072669

```
        _cat=1;
        output;
        _trt=3;
        COVBLSTN=2;
        _catord=2;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        COVBLSTN=999;
        _catord=3;
        _cat=1;
        output;
        _trt=2;
        COVBLSTN=999;
        _catord=3;
        _cat=1;
        output;
        _trt=3;
        COVBLSTN=999;
        _catord=3;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame8;
        by _datasrt _blcksrt _cat COVBLSTN _trt;
run;

proc sort data=_pct8;
        by _datasrt _blcksrt _cat COVBLSTN _trt;
run;

data _pct8;
        merge _frame8(in=_inframe) _pct8;
        by _datasrt _blcksrt _cat COVBLSTN _trt;

        if _inframe;

        if count=. then
                count=0;
run;

*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct8;
        by _datasrt _blcksrt COVBLSTN;
```

FDA-CBER-2022-5812-0072670

```
run;

data _miss8(keep=_datasrt _blcksrt COVBLSTN totcount);
      set _pct8;
      where COVBLSTN=9998;
      retain totcount;
      by _datasrt _blcksrt COVBLSTN;

      if first.COVBLSTN then
            totcount=0;
      totcount=totcount+count;

      if last.COVBLSTN;
run;

data _pct8(drop=totcount);
      merge _pct8 _miss8;
      by _datasrt _blcksrt COVBLSTN;

      if totcount=0 then
            delete;
run;

*******************************************************************;
*IF PCTDISP=CAT/DPTVAR then add dptvar into denomitor frame dataset;
*******************************************************************;
*-------------------------------------------------------------------;
* Merge the _DENOMIN with its frame up dataset (_denomf) ;
*-------------------------------------------------------------------;

proc sort data=_denomf8;
      by _datasrt _cat;
run;

proc sort data=_denomin8;
      by _datasrt _cat;
run;

data _denomin8;
      merge _denomf8(in=_inframe) _denomin8;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=6;
run;

proc sort data=_pct8;
      by _datasrt _cat;
run;

data _pct8;
      if 0 then
            set _basetemplate;
      merge _denomin8(in=_a) _pct8;
```

FDA-CBER-2022-5812-0072671

```
        by _datasrt _cat;

        if _a;
        _varname="COVBLSTN ";
        _vrlabel="Baseline SARS-CoV-2 status ";
        _rwlabel=put(COVBLSTN, sars.);

        if COVBLSTN=9998 then
                do;
                        _rwlabel="Missing ";
                        _catord=9998;
                end;
        else if COVBLSTN=9999 then
                do;
                        _rwlabel="Total ";
                        _catord=9999;
                end;

        if _catord=. then
                _catord=9997;
run;

proc sort data=_pct8;
        by _datasrt _blcksrt _catord COVBLSTN _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base8;
        length _catlabl $200;
        set _pct8 end=eof;
        by _datasrt _blcksrt _catord COVBLSTN _trt _cat;
        retain _rowsrt 0 _rowmax 0;
        array _trtcnt(*) _trt1-_trt4;
        drop _rowmax _cpct;
        length _cpct $100;
        _cpct=' ';
        _module='mcatstat';

        if count > . then
                _cvalue=put(count, 5.);
        else
                _cvalue=put(0, 5.);
        *-------------------------------------------------------------------;
        * Format percent to append to display value in _CVALUE ;
        *-------------------------------------------------------------------;

        if _trt ne . then
                do;

                        if _trtcnt(_trt) > 0 then
```
FDA-CBER-2022-5812-0072672

```
                    do;
                         percent=count / _trtcnt(_trt) * 100;

                         if percent > 0 then
                              do;

                                   if round(percent, 0.1) GE 0.1 then
                                        _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                   else
                                        _cpct="(*ESC*){nbspace 1}(0.0)";
                                   _cvalue=trim(_cvalue)||_cpct;
                              end;
                    end;
          end;

     if length(_cvalue) < 13 then
          do;
               *-------------------------------------------------------------------;
               * Put character A0x at right most character to pad text;
               *-------------------------------------------------------------------;
               substr(_cvalue, 13, 1)='A0'x;
          end;

     if first.COVBLSTN then
          do;
               _rowsrt=_rowsrt + 1;
               _rowmax=max(_rowsrt, _rowmax);
          end;
     _datatyp='data';
     _indent=0;
     _dptindt=0;
     _vorder=1;
     _rowjump=1;

     if upcase(_rwlabel)='_NONE_' then
          _rwlabel=' ';
     _indent=3;
     _dptindt=0;

     if _trt=3 +1 then
          _trt=9999;

     if eof then
          call symput('_rowsrt', compress(put(_rowmax, 4.)));
     _direct="TOP ";
     _p=2;
run;

data _anal9;
     length COMBODFLNX 8;
     set _data1;

     if COMBODFLNX=. then
          COMBODFLNX=9998;
```

FDA-CBER-2022-5812-0072673

```
        _blcksrt=7;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal9;
        by _datasrt _blcksrt COMBODFLNX _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp9;
        set _anal9;
        output;
run;

proc sort data=_temp9 out=_temp99 nodupkey;
        by _datasrt _blcksrt _cat COMBODFLNX _trt USUBJID;
        ;
run;

proc freq data=_temp99;
        format COMBODFLNX;
        tables _datasrt*_blcksrt*_cat * COMBODFLNX * _trt / sparse norow nocol
                nopercent out=_pct9(drop=percent);
run;

proc sort data=_anal9 out=_denom9(keep=_datasrt _cat) nodupkey;
        ;
        by _datasrt _cat;
run;

data _denom9;
        set _denom9;
        by _datasrt _cat;
        label count='count';
        _trt=1;
        count=&_trt1;
        output;
        _trt=2;
        count=&_trt2;
        output;
        _trt=3;
        count=&_trt3;
        output;
run;

*------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*------------------------------------------------------------------;
```

```
data _denomf9;
        _datasrt=1;
        set _bydat1(keep=);
        * All treatment groups ;
        _trt1=0;
        _trt2=0;
        _trt3=0;
        * _CAT is the subgroup variable ;
        _cat=1;
        output;
run;

*-----------------------------------------------------------------------;
* Transpose _DENOM into _DENOMIN to get COUNT as _TRTn columns ;
*-----------------------------------------------------------------------;

proc sql noprint;
        select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
                where (libname="WORK" and memname="_DENOM9");
        select setting into :miss from dictionary.options where
                upcase(optname)="MISSING";
quit;

;

proc transpose data=_denom9 out=_denomin9(drop=_name_ _label_) prefix=_trt;
        by _datasrt _cat;
        var count;
        id _trt;
run;

*-----------------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
*-----------------------------------------------------------------------;

proc sql noprint;
        select count(distinct COMBODFLNX) into : totexpv from _anal9;
        select distinct COMBODFLNX , COMBODFLX into :expv1 - :expv2 ,
                :catlab1 - :catlab2 from _anal9 order by COMBODFLNX;
quit;

*-----------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*-----------------------------------------------------------------------;

data _frame9;
        _datasrt=1;
        set _bydat1(keep=);
        _blcksrt=7;
        length COMBODFLNX 8;
        length _catLabl $7;
        _catLabl=' ';
        _catLabl="Yes ";
        _trt=1;
        COMBODFLNX=1;
        _catord=1;
        _cat=1;
        output;
        _trt=2;
        COMBODFLNX=1;
        _catord=1;
        _cat=1;
        output;
        _trt=3;
        COMBODFLNX=1;
        _catord=1;
        _cat=1;
        output;
        _catLabl="No ";
        _trt=1;
        COMBODFLNX=2;
        _catord=2;
        _cat=1;
        output;
        _trt=2;
        COMBODFLNX=2;
        _catord=2;
        _cat=1;
        output;
        _trt=3;
        COMBODFLNX=2;
        _catord=2;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame9;
        by _datasrt _blcksrt _cat COMBODFLNX _trt;
run;

proc sort data=_pct9;
        by _datasrt _blcksrt _cat COMBODFLNX _trt;
run;

data _pct9;
        merge _frame9(in=_inframe) _pct9;
        by _datasrt _blcksrt _cat COMBODFLNX _trt;

        if _inframe;

        if count=. then
                count=0;
run;
```

```
*-------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*-------------------------------------------------------------------;

proc sort data=_pct9;
      by _datasrt _blcksrt COMBODFLNX;
run;

data _miss9(keep=_datasrt _blcksrt COMBODFLNX totcount);
      set _pct9;
      where COMBODFLNX=9998;
      retain totcount;
      by _datasrt _blcksrt COMBODFLNX;

      if first.COMBODFLNX then
            totcount=0;
      totcount=totcount+count;

      if last.COMBODFLNX;
run;

data _pct9(drop=totcount);
      merge _pct9 _miss9;
      by _datasrt _blcksrt COMBODFLNX;

      if totcount=0 then
            delete;
run;

*****************************************************************;
*IF PCTDISP=CAT/DPTVAR then add dptvar into denomitor frame dataset;
*****************************************************************;
*-------------------------------------------------------------------;
* Merge the _DENOMIN with its frame up dataset (_denomf) ;
*-------------------------------------------------------------------;

proc sort data=_denomf9;
      by _datasrt _cat;
run;

proc sort data=_denomin9;
      by _datasrt _cat;
run;

data _denomin9;
      merge _denomf9(in=_inframe) _denomin9;
      by _datasrt _cat;

      if _inframe;
      _blcksrt=7;
run;
```

```
*-------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*-------------------------------------------------------------------;

proc sort data=_pct9;
      by _datasrt _cat;
run;

*-------------------------------------------------------------------;
* Create _VARNAME variable to hold depend variable name. ;
* Create _VRLABEL variable to display Group label. ;
* Create _RWLABEL variable to display &dptvar categories. ;
*-------------------------------------------------------------------;

data _pct9;
      if 0 then
            set _basetemplate;
      merge _denomin9(in=_a) _pct9;
      by _datasrt _cat;

      if _a;
      _varname="COMBODFLNX ";
      _vrlabel="Comorbidities(*ESC*){super e} ";
      _rwlabel=_catLabl;

      if COMBODFLNX=9998 then
            do;
                  _rwlabel="Missing ";
                  _catord=9998;
            end;
      else if COMBODFLNX=9999 then
            do;
                  _rwlabel="Total ";
                  _catord=9999;
            end;

      if _catord=. then
            _catord=9997;
run;

proc sort data=_pct9;
      by _datasrt _blcksrt _catord COMBODFLNX _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base9;
      length _catlabl $200;
      set _pct9 end=eof;
      by _datasrt _blcksrt _catord COMBODFLNX _trt _cat;
      retain _rowsrt 0 _rowmax 0;
```

```
array _trtcnt(*) _trt1-_trt4;
drop _rowmax _cpct;
length _cpct $100;
_cpct=' ';
_module='mcatstat';

if count > . then
      _cvalue=put(count, 5.);
else
      _cvalue=put(0, 5.);
*-------------------------------------------------------------------;
* Format percent to append to display value in _CVALUE ;
*-------------------------------------------------------------------;

if _trt ne . then
      do;

            if _trtcnt(_trt) > 0 then
                  do;
                        percent=count / _trtcnt(_trt) * 100;

                        if percent > 0 then
                              do;

                                    if round(percent, 0.1) GE 0.1 then
                                          _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                    else
                                          _cpct="(*ESC*){nbspace 1}(0.0)";
                                    _cvalue=trim(_cvalue)||_cpct;
                              end;
                  end;
      end;

if length(_cvalue) < 13 then
      do;
            *-------------------------------------------------------------------;
            * Put character A0x at right most character to pad text;
            *-------------------------------------------------------------------;
            substr(_cvalue, 13, 1)='A0'x;
      end;

if first.COMBODFLNX then
      do;
            _rowsrt=_rowsrt + 1;
            _rowmax=max(_rowsrt, _rowmax);
      end;
_datatyp='data';
_indent=0;
_dptindt=0;
_vorder=1;
_rowjump=1;

if upcase(_rwlabel)='_NONE_' then
      _rwlabel=' ';
```

```
        _indent=3;
        _dptindt=0;

        if _trt=3 +1 then
                _trt=9999;

        if eof then
                call symput('_rowsrt', compress(put(_rowmax, 4.)));
        _direct="TOP ";
        _p=2;
run;

data _anal10;
        length OBESEFLN 8;
        set _data1;

        if OBESEFLN=. then
                OBESEFLN=9998;
        _blcksrt=8;
        _cnt=1;
        _cat=1;

        if _trt <=0 then
                delete;
        output;
run;

proc sort data=_anal10;
        by _datasrt _blcksrt OBESEFLN _trt _cat;
run;

*--- Counts for each by-sequence, dependant var, and treatment combination ---*;

data _temp10;
        set _anal10;
        output;
run;

proc sort data=_temp10 out=_temp910 nodupkey;
        by _datasrt _blcksrt _cat OBESEFLN _trt USUBJID;
        ;
run;

proc freq data=_temp910;
        format OBESEFLN;
        tables _datasrt*_blcksrt*_cat * OBESEFLN * _trt / sparse norow nocol nopercent
                out=_pct10(drop=percent);
run;

proc sort data=_anal10 out=_denom10(keep=_datasrt _cat) nodupkey;
        ;
        by _datasrt _cat;
run;
```

```
data _denom10;
      set _denom10;
      by _datasrt _cat;
      label count='count';
      _trt=1;
      count=&_trt1;
      output;
      _trt=2;
      count=&_trt2;
      output;
      _trt=3;
      count=&_trt3;
      output;
run;

*-----------------------------------------------------------------------;
* Create _DENOMF a frame dataset for the denominators ;
*-----------------------------------------------------------------------;

data _denomf10;
      _datasrt=1;
      set _bydat1(keep=);
      * All treatment groups ;
      _trt1=0;
      _trt2=0;
      _trt3=0;
      * _CAT is the subgroup variable ;
      _cat=1;
      output;
run;

proc transpose data=_denom10 out=_denomin10(drop=_name_ _label_) prefix=_trt;
      by _datasrt _cat;
      var count;
      id _trt;
run;

*-----------------------------------------------------------------------;
* VALRANGE=FULL. Create full rank categories WITHOUT using where. ;
*-----------------------------------------------------------------------;

proc sql noprint;
      select count(distinct OBESEFLN) into : totexpv from _anal10;
      select distinct OBESEFLN into :expv1 - :expv2 from _anal10 order by OBESEFLN;
quit;

*-----------------------------------------------------------------------;
* Create _FRAME dataset using all combinations of category variable ;
*-----------------------------------------------------------------------;

data _frame10;
      _datasrt=1;
      set _bydat1(keep=);
      _blcksrt=8;
```

```
        length OBESEFLN 8;
        _catLabl=" ";
        _trt=1;
        OBESEFLN=1;
        _catord=1;
        _cat=1;
        output;
        _trt=2;
        OBESEFLN=1;
        _catord=1;
        _cat=1;
        output;
        _trt=3;
        OBESEFLN=1;
        _catord=1;
        _cat=1;
        output;
        _catLabl=" ";
        _trt=1;
        OBESEFLN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=2;
        OBESEFLN=2;
        _catord=2;
        _cat=1;
        output;
        _trt=3;
        OBESEFLN=2;
        _catord=2;
        _cat=1;
        output;
run;

*-------------------------------------------------------------------;
* Merge the _PCT dataset with its frameup dataset(_FRAME) ;
*-------------------------------------------------------------------;

proc sort data=_frame10;
        by _datasrt _blcksrt _cat OBESEFLN _trt;
run;

proc sort data=_pct10;
        by _datasrt _blcksrt _cat OBESEFLN _trt;
run;

data _pct10;
        merge _frame10(in=_inframe) _pct10;
        by _datasrt _blcksrt _cat OBESEFLN _trt;

        if _inframe;

        if count=. then
```

```
          count=0;
run;

*------------------------------------------------------------------;
* Delete Zero filled MISSING category rows for each combination of;
* _datasrt &_byvar _blcksrt;
*------------------------------------------------------------------;

proc sort data=_pct10;
     by _datasrt _blcksrt OBESEFLN;
run;

data _miss10(keep=_datasrt _blcksrt OBESEFLN totcount);
     set _pct10;
     where OBESEFLN=9998;
     retain totcount;
     by _datasrt _blcksrt OBESEFLN;

     if first.OBESEFLN then
          totcount=0;
     totcount=totcount+count;

     if last.OBESEFLN;
run;

data _pct10(drop=totcount);
     merge _pct10 _miss10;
     by _datasrt _blcksrt OBESEFLN;

     if totcount=0 then
          delete;
run;

proc sort data=_denomf10;
     by _datasrt _cat;
run;

proc sort data=_denomin10;
     by _datasrt _cat;
run;

data _denomin10;
     merge _denomf10(in=_inframe) _denomin10;
     by _datasrt _cat;

     if _inframe;
     _blcksrt=8;
run;

*------------------------------------------------------------------;
* Merge in _PCT(counts) with the _DENOMIN(denominator for percents) ;
*------------------------------------------------------------------;

proc sort data=_pct10;
```

```
      by _datasrt _cat;
run;

data _pct10;
      if 0 then
            set _basetemplate;
      merge _denomin10(in=_a) _pct10;
      by _datasrt _cat;

      if _a;
      _varname="OBESEFLN ";
      _vrlabel="Obese(*ESC*){super f} ";
      _rwlabel=put(OBESEFLN, obes.);

      if OBESEFLN=9998 then
            do;
                  _rwlabel="Missing ";
                  _catord=9998;
            end;
      else if OBESEFLN=9999 then
            do;
                  _rwlabel="Total ";
                  _catord=9999;
            end;

      if _catord=. then
            _catord=9997;
run;

proc sort data=_pct10;
      by _datasrt _blcksrt _catord OBESEFLN _trt _cat;
run;

*-------------------------------------------------------------------;
* Create _CVALUE variable to display results. ;
* Create _ROWSRT variable to order results. ;
*-------------------------------------------------------------------;

data _base10;
      length _catlabl $200;
      set _pct10 end=eof;
      by _datasrt _blcksrt _catord OBESEFLN _trt _cat;
      retain _rowsrt 0 _rowmax 0;
      array _trtcnt(*) _trt1-_trt4;
      drop _rowmax _cpct;
      length _cpct $100;
      _cpct=' ';
      _module='mcatstat';

      if count > . then
            _cvalue=put(count, 5.);
      else
            _cvalue=put(0, 5.);
      *-------------------------------------------------------------------;
```

FDA-CBER-2022-5812-0072684

```
        * Format percent to append to display value in _CVALUE ;
        *--------------------------------------------------------------------;

        if _trt ne . then
              do;

                    if _trtcnt(_trt) > 0 then
                          do;
                                percent=count / _trtcnt(_trt) * 100;

                                if percent > 0 then
                                      do;

                                            if round(percent, 0.1) GE 0.1 then
                                                  _cpct="(*ESC*){nbspace 1}("||strip(put(percent, 5.1))||")";
                                            else
                                                  _cpct="(*ESC*){nbspace 1}(0.0)";
                                            _cvalue=trim(_cvalue)||_cpct;
                                      end;
                          end;
              end;

        if length(_cvalue) < 13 then
              do;
                    *--------------------------------------------------------------------;
                    * Put character A0x at right most character to pad text;
                    *--------------------------------------------------------------------;
                    substr(_cvalue, 13, 1)='A0'x;
              end;

        if first.OBESEFLN then
              do;
                    _rowsrt=_rowsrt + 1;
                    _rowmax=max(_rowsrt, _rowmax);
              end;
        _datatyp='data';
        _indent=0;
        _dptindt=0;
        _vorder=1;
        _rowjump=1;

        if upcase(_rwlabel)='_NONE_' then
              _rwlabel=' ';
        _indent=3;
        _dptindt=0;

        if _trt=3 +1 then
              _trt=9999;

        if eof then
              call symput('_rowsrt', compress(put(_rowmax, 4.)));
        _direct="TOP ";
        _p=2;
run;
```

FDA-CBER-2022-5812-0072685

```
data _anal11;
       set _data1;
       where _trt > 0;
       _blcksrt=9;
       output;
run;

*----------------------------------------------------------------------;
* Make sure data is sorted by groups ;
*----------------------------------------------------------------------;

proc sort data=_anal11;
       by _datasrt _blcksrt _trt;
run;

*----------------------------------------------------------------------;
* Call PROC UNIVARIATE to generate all possible statistics plus any ;
* Percentiles or Confidence Intervals. ;
*----------------------------------------------------------------------;

proc univariate data=_anal11 noprint;
       ;
       by _datasrt _blcksrt _trt;
       var AGETR01;
       output out=_msum11 CSS=CSS CV=CV KURTOSIS=KURTOSIS MAX=MAX MEAN=MEAN N=N
              MIN=MIN MODE=MODE RANGE=RANGE NMISS=NMISS NOBS=NOBS STDMEAN=STDMEAN
              SKEWNESS=SKEWNESS STD=STD USS=USS SUM=SUM VAR=VAR MEDIAN=MEDIAN P1=P1
P5=P5
              P10=P10 P90=P90 P95=P95 P99=P99 Q1=Q1 Q3=Q3 QRANGE=QRANGE GINI=GINI MAD=MAD
              QN=QN SN=SN STD_GINI=STD_GINI STD_MAD=STD_MAD STD_QN=STD_QN
              STD_QRANGE=STD_QRANGE STD_SN=STD_SN NORMAL=NORMAL PROBN=PROBN
MSIGN=MSIGN
              PROBM=PROBM SIGNRANK=SIGNRANK PROBS=PROBS T=T PROBT=PROBT;
run;

*---------------------------------------------------------------------;
*Create Frame dataset when user requested Subgrouping as well as set;
*sparsesgrpyn to Y to sparse subgrp categories of a format.;
*----------------------------------------------------------------------;

data _frame11;
       set _bydat1(keep=);
       _datasrt=1;
       _blcksrt=9;
       _catord=1;
       _trt=1;
       _cat=1;
       output;
       _trt=2;
       _cat=1;
       output;
       _trt=3;
       _cat=1;
```

FDA-CBER-2022-5812-0072686

```
        output;
run;

proc sort data=_frame11;
        by _datasrt _blcksrt _trt;
run;

data _msum11;
        merge _msum11 _frame11;
        by _datasrt _blcksrt _trt;
run;

*-------------------------------------------------------------------;
* Generate _result1 from OUT= dataset of PROC UNIVARIATE ;
*-------------------------------------------------------------------;

data _result1_11;
        if 0 then
                set _basetemplate;
        set _msum11 end=eof;
        _rowsrt=0 + 1;
        _rwlabel="Mean (SD) ";
        _cvalue=' ';
        _nvalue=.;
        *-------------------------------------------------------------------;
        * MEAN(STD) ;
        *-------------------------------------------------------------------;

        if mean ne . and std ne . then
                do;
                        _cValue=strip(put(mean, 5.1) ) || ' (' || strip(put(std, 5.2) ) || ')';
                end;
        else if mean eq . then
                _cValue="-" || ' (' || "-" || ')';
        else if std eq . then
                do;
                        _cValue=strip(put(mean, 5.1) ) || ' (' || "NE" || ')';
                end;
        output;
        _rowsrt=0 + 2;
        _rwlabel="Median ";
        _cvalue=' ';
        _nvalue=.;
        _nvalue=MEDIAN;

        if MEDIAN ne . then
                _cValue=strip(put(MEDIAN, 5.1) );
        else
                _cValue="-";
        output;
        _rowsrt=0 + 3;
        _rwlabel="Min, max ";
        _cvalue=' ';
        _nvalue=.;
        *------------------------------------------------------------------;
        * MINMAX MINMAXC MEDIAN(MINMAX) MEDIAN(MINMAXC) ;
        *------------------------------------------------------------------;
        _cValue=' ';

        if min ^=. & max ^=. then
                do;
                        _cValue=trim(_cvalue) || ' (' || strip(put(min, 5.0)
                                )|| ', ' || strip(put(max, 5.0) )||')';
                end;
        else if min=. & max=. then
                do;
                        _cValue=trim(_cvalue) || ' (' || "-" || ', ' || "-" ||')';
                end;
        _cValue=compbl(_cValue);
        output;
run;

*--------------------------------------------------------------------;
* Generate _logresult1 from OUT= dataset of PROC UNIVARIATE for log stats;
*--------------------------------------------------------------------;

data _logresult1_11;
        if 0 then
                set _basetemplate;
        stop;
run;

*------------------------------------------------------------------;
* Generate _result2 from confidence interval output dataset ;
*------------------------------------------------------------------;

data _result2_11;
        if 0 then
                set _basetemplate;
        stop;
run;

*----------------------------------------------------------------------;
* Generate _logresult2 from confidence interval output dataset for log stats;
*----------------------------------------------------------------------;

data _logresult2_11;
        if 0 then
                set _basetemplate;
        stop;
run;

*------------------------------------------------------------------;
* Combine to form one result dataset. Set variables that do not depend ;
* on the statistic. Sort the result. ;
*------------------------------------------------------------------;

data _base11;
```

FDA-CBER-2022-5812-0072688

```
        set _result1_11 _result2_11 _logresult1_11 _logresult2_11;
        ;

        if _trt=4 then
                _trt=9999;
        _varname="AGETR01";
        _vrlabel="Age at vaccination (years) ";
        _datatyp='data';
        _module='msumstat';
        _indent=5;
        _rowjump=1;
        _dptindt=0;
run;

*------------------------------------------------------------------;
* merge ISAM subgroup variables _SUBCAT _COLABEL ;
*------------------------------------------------------------------;

proc sort data=_base11;
        by _datasrt _blcksrt _rowsrt;
run;

********************************************************************************;
*SPECIFICATION 8 -2) AgeTR0x (Age at Each Dose) - descriptive statistics *;
********************************************************************************;
********************************************************************************;
*SPECIFICATION 10 -1) titles and footnotes *;
* 2) display *;
********************************************************************************;

proc sql noprint;
        select max(_trt) into :maxtrt from _base1;
quit;

data _base1;
        set _base1;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base2;
quit;
```

```
data _base2;
        set _base2;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base3;
quit;

data _base3;
        set _base3;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base4;
quit;

data _base4;
        set _base4;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
```

```
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base5;
quit;

data _base5;
        set _base5;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base6;
quit;

data _base6;
        set _base6;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
        select max(_trt) into :maxtrt from _base7;
quit;

data _base7;
        set _base7;

        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";

    if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
    select max(_trt) into :maxtrt from _base8;
quit;

data _base8;
    set _base8;

    if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";

    if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";

    if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
    select max(_trt) into :maxtrt from _base9;
quit;

data _base9;
    set _base9;

    if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";

    if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";

    if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
        sum=.) then
            _cvalue="(*ESC*){nbspace 5}";
run;

proc sql noprint;
    select max(_trt) into :maxtrt from _base10;
quit;

data _base10;
    set _base10;
```

```
        if (_module="mcatstat" and _trt=1 and _trt1=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=2 and _trt2=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";

        if (_module="mcatstat" and _trt=3 and _trt3=0) or (_module="msumstat" and
                sum=.) then
                        _cvalue="(*ESC*){nbspace 5}";
run;

data _final;
        set _base1 _base2 _base3 _base4 _base5 _base6 _base7 _base8 _base9 _base10
                _base11;
run;

proc sort data=_final;
        by _datasrt _blcksrt _rowsrt;
run;

*-------------------------------------------------------------------;
* At least one of TRT and STAT is vertical;
*-------------------------------------------------------------------;

data _final;
        set _final;
        drop __trt;

        if _trt=9999 then
                __trt=3 + 1;
        else
                __trt=_trt;

        if __trt=. then
                __trt=1;
        _column=_trt;

        if _column=9999 then
                _column=3 + 1;
run;

proc sort data=_final out=_final;
        by _datasrt _blcksrt _rowsrt _column;
run;

data _linecnt;
        set _final end=eof;
        by _datasrt _blcksrt _rowsrt _column;
        retain _totline _maxval _maxrow _rwlbtag _vrlbtag 0 _maxline _linecnt;
        keep _datasrt _blcksrt _totline _linecnt _maxrow;
```

```
if _rowjump=. then
      _rowjump=1;

if first._blcksrt then
      do;
            *-------------------------------------------------------------------;
            * Count words inside DATA step ;
            *-------------------------------------------------------------------;
            _token=repeat(' ', 99);
            _count=1;
            _token=scan(_vrlabel, _count, "|");

            if _token=: '_' then
                  _tag=1;
            else
                  _tag=0;

            do while(_token ^=' ');
                  _count=_count + 1;
                  _token=scan(_vrlabel, _count, "|");
            end;
            _linecnt=_count - 1 + _tag;
            ;
            _totline=_linecnt;

            if _vrlabel ne ' ' and _vrlabel ne '^' & _datatyp='data' then
                  _vrlbtag=1;
      end;

if first._rowsrt then
      do;
            *-------------------------------------------------------------------;
            * Count words inside DATA step ;
            *-------------------------------------------------------------------;
            _token=repeat(' ', 99);
            _count=1;
            _token=scan(_rwlabel, _count, "|");

            if _token=: '_' then
                  _tag=1;
            else
                  _tag=0;

            do while(_token ^=' ');
                  _maxrow=max(_maxrow, length(_token) + _indent);
                  _count=_count + 1;
                  _token=scan(_rwlabel, _count, "|");
            end;
            _maxline=_count - 1 + _tag;

            if _rwlabel ne ' ' then
                  _rwlbtag=1;
            _totline + _rowjump - 1;
      end;
```

FDA-CBER-2022-5812-0072694

```
        *------------------------------------------------------------------;
        * Count words inside DATA step ;
        *------------------------------------------------------------------;
        _token=repeat(' ', 99);
        _count=1;
        _token=scan(_cvalue, _count, "|");

        if _token=: '_' then
                _tag=1;
        else
                _tag=0;

        do while(_token ^=' ');
                _maxval=max(_maxval, length(_token));
                _count=_count + 1;
                _token=scan(_cvalue, _count, "|");
        end;
        _ccnt=_count - 1 + _tag;
        _maxline=max(_maxline, _ccnt);

        if last._rowsrt then
                _totline=_maxline + _totline;

        if last._blcksrt then
                do;
                        _totline=_totline - _rowjump + 1;
                        output;
                end;

        if eof then
                do;
                        call symput('_valwid', compress(put(_maxval, 3.)));
                        call symput('_rwlbtag', put(_rwlbtag, 1.));
                        call symput('_vrlbtag', put(_vrlbtag, 1.));
                end;
run;

data _final;
        length _direct $20;
        _direct=' ';
        merge _final _linecnt;
        by _datasrt _blcksrt;
run;

proc sql noprint;
        create table rspon as select distinct _trt, _column , _vrlabel as _rwlabel ,
                _datasrt, _blcksrt, (min(_rowsrt)-0.5) as _rowsrt , _dptindt as _indent , 0
                as _dptindt from _final(where=(_vrlabel^=' ')) group by _trt, _column ,
                _datasrt, _blcksrt, _vrlabel;
quit;

data ADSL_S005_ALL1_PED6_SAF;
        length _rvalue $800;
        set _final rspon end=eof;
```

FDA-CBER-2022-5812-0072695

```
        _rwindt=sum(_indent, _dptindt);

        if _rwindt <=0 then
                _rvalue=_rwlabel;

        /* else _rvalue=repeat(byte(160),_rwindt-1)||_rwlabel; */
        else
                _rvalue=repeat("~{nbspace 1}", _rwindt-1)||_rwlabel;
        _dummy=1;

        if _trt=. then
                _trt=1;
run;

proc sort data=ADSL_S005_ALL1_PED6_SAF;
        by _datasrt _trt _blcksrt _rowsrt;
run;

data treat;
        length FMTNAME $8 start 8 label $200;
        fmtname='TREAT';

        do start=1 to 3 + ("N"="Y");
                label=symget('_TRTLB'|| compress(put(start, 4.)));
                label=trim(label)
                        || "|   (N~{super a}=" || compress(symget("_TRT" || compress(put(start,
                        4.)))) || ")"
|| "|n~{super b}      (%)";
                output;
        end;
run;

proc format cntlin=treat;
run;

options orientation=LANDSCAPE papersize="LETTER";
ods escapechar="~";
title1 "Demographic Characteristics (*ESC*){unicode 2013} Phase 2/3 Subjects 12 Through 15 Years of Age (*ESC*)
{unicode 2013} Safety Population ";
footnote1
        "Abbreviation: SARS-CoV-2 = severe acute respiratory syndrome coronavirus 2. ";
footnote2 "a.(*ESC*){nbspace 5}N = number of subjects in the specified group, or the total sample.  This value is the
denominator for the percentage calculations. ";
footnote3 "b.(*ESC*){nbspace 5}n = Number of subjects with the specified characteristic. ";
footnote4 "c.(*ESC*){nbspace 5}Positive N-binding antibody result at Visit 1, positive NAAT result at Visit 1, or
medical history of COVID-19. ";
footnote5 "d.(*ESC*){nbspace 5}Negative N-binding antibody result at Visit 1, negative NAAT result at Visit 1, and no
medical history of COVID-19. ";
footnote6 "e.(*ESC*){nbspace 5}Number of subjects who have 1 or more comorbidities that increase the risk of severe
COVID-19 disease: defined as subjects who had at least one of the Charlson comorbidity index category or BMI
(*ESC*){unicode 2265}95(*ESC*){super th} percentile. ~n f.(*ESC*){nbspace 5}Obese is defined as BMI (*ESC*)
{unicode 2265}95(*ESC*){super th} percentile from the growth chart. Refer to the CDC growth charts at
https://www.cdc.gov/growthcharts/html_charts/bmiagerev.htm. ";
```

```
data outdata1;
      set ADSL_S005_ALL1_PED6_SAF;

      if upcase(_module)='MCATSTAT' then
            _cvalue=transtrn(compress(_cvalue), '(', ' (');
      _fixvar=1;
      _fix2var=1;
run;

option nobyline;

proc sort data=outdata1;
      by _datasrt _trt _blcksrt _rowsrt;
run;

proc sql noprint;
      select distinct start, label into :start1, :_trlbl1 - :_trlbl99 from treat
            order by start;
quit;

proc sort data=outdata1 out=_pre_transposed;
      by _fixvar _fix2var _datasrt _blcksrt _rowsrt _rvalue _trt;
run;

data _pre_transposed;
      set _pre_transposed;

      if _trt=9999 then
            _trt=3 +1;
run;

proc transpose data=_pre_transposed out=_column_transposed (drop=_name_)
            prefix=TRT;
      by _fixvar _fix2var _datasrt _blcksrt _rowsrt _rvalue;
      var _cvalue;
      id _trt;
run;

ods html file="&outtable.";

data REPORT;
      set _column_transposed;
      _dummy=1;
run;

proc sort data=report;
      by _datasrt _blcksrt _rowsrt _dummy;
run;

proc report data=report nowd list missing contents="" split="|" spanrows
            style(report)={} style(header)={} style(column)={};
      column _fixvar _fix2var _datasrt _blcksrt _rowsrt ("" _rvalue)
            ("Vaccine Group (as Administered)~{line}" ("" TRT1 TRT2) TRT3 _dummy);
      define _fixvar / group noprint;
```

FDA-CBER-2022-5812-0072697

```
     define _fix2var / group noprint;
     define _datasrt / group order=internal noprint;
     define _blcksrt / group order=internal noprint;
     define _rowsrt / group order=internal noprint;
     define _rvalue / group id " " order=data style(column)={just=left width=60mm
          rightmargin=18px} style(header)={just=left} left;
     ;
     define _dummy / sum noprint;
     define TRT1 / group nozero "&_trlbl1." spacing=2 style(column)={width=35mm
          leftmargin=12px} style(header)={just=center} center;
     define TRT2 / group nozero "&_trlbl2." spacing=2 style(column)={width=35mm
          leftmargin=12px} style(header)={just=center} center;
     define TRT3 / group nozero "&_trlbl3." spacing=2 style(column)={width=35mm
          leftmargin=12px} style(header)={just=center} center;
     break before _fixvar / contents="" page;
     compute before _fix2var;
          line @1 " ~n ";
     endcomp;
     compute after _blcksrt;
          line " ~n ";
     endcomp;
run;

ods HTML close;

proc printto;
run;
```