```
***************************************************************************************************;
**  Program Name    :  adva-s001-gmr-ped-ev-eval.sas                              **;
**  Date Created    :  23Mar2021                                          **;
**  Programmer Name :  (b) (4), (b) (6)                                   **;
**  Purpose         :  Create adva-s001-gmr-ped-ev-eval                           **;
**  Input data      :  adsl adva                                       **;
**  Output file     :  adva-s001-gmr-ped-ev-eval.html                            **;
***************************************************************************************************;
options mprint mlogic symbolgen mprint symbolgen mlogic nocenter missing=" ";
ods escapechar="~";

proc datasets library=WORK kill nolist nodetails;
quit;

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

%let outpath=&prot./analysis/esub;
%let outlog=&outpath./logs/adva-s001-gmr-ped-ev-eval.log;
%let outtable=&outpath./output/adva-s001-gmr-ped-ev-eval.html;

*********************************************************************************************;
* Clean *;
*********************************************************************************************;
options mprint mlogic symbolgen;
title;
footnote;

proc delete data=work._all_;
run;

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

proc format;
  value PARAMN 1="SARS-CoV-2 neutralization assay - NT50 (titer)";
  value visx 1="1/Prevax" 6="2/1 Month";
run;

data adsl;
  set datvprot.adsl;

  if trt01pn=8 and agegr4n=1 then
    trtprn=1;
  else if trt01pn=8 and agegr4n=2 then
    trtprn=2;
  else if trt01pn=9 and agegr4n=1 then
    trtprn=3;
  else if trt01pn=9 and agegr4n=2 then
    trtprn=4;
  trtpr=trt01p;
run;
```

```
data adva;
  set datvprot.adva;

  if trt01pn=8 and agegr4n=1 then
    trtprn=1;
  else if trt01pn=8 and agegr4n=2 then
    trtprn=2;
  else if trt01pn=9 and agegr4n=1 then
    trtprn=3;
  else if trt01pn=9 and agegr4n=2 then
    trtprn=4;
  trtpr=trt01p;
run;

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;

data g_a_dsin;
  set adva;
  analysis_subset='Y';
run;

data g_adsl_dsin;
  set adsl;
  where PEDIMMFL="Y" and HIVFL ne "Y";
run;

data g_a_dsin;
  set g_a_dsin;
  where avisitn le 6 and EVIMMFL eq 'Y' and paramn in (1) and avisitn>1 and
    PEDIMMFL="Y" and EV1MD2FL="Y" and ANL01FL="Y";
run;

proc sort data=g_adsl_dsin out=_ds1;
  by usubjid TRTPRN;
run;

proc sort data=g_a_dsin out=_ds2;
  by usubjid TRTPRN;
run;

data final_dsin;
  merge _ds1(in=d1) _ds2(in=d2);
  by usubjid TRTPRN;

  if d1;
run;

data __trtmap;
```

```
   length trtcode trtdecd $100;

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

   if vtype(TRTPRN)='C' then
      trtcode=tranwrd(compbl(quote("1")), ' ', '" "');
   else
      trtcode="1";
   trtdecd="12-15 Years";
   trtvar="TRTPRN";
   trtlbl="TRTPR";
   output;
   trtval=2;

   if vtype(TRTPRN)='C' then
      trtcode=tranwrd(compbl(quote("2")), ' ', '" "');
   else
      trtcode="2";
   trtdecd="16-25 Years";
   trtvar="TRTPRN";
   trtlbl="TRTPR";
   output;
   stop;
run;

data g_dsin;
   set final_dsin;

   if TRTPRN in (1) then
      do;
         newtrtn=1;
         newtrt=coalescec("12-15 Years", TRTPR);
         output;
      end;

   if TRTPRN in (2) then
      do;
         newtrtn=2;
         newtrt=coalescec("16-25 Years", TRTPR);
         output;
      end;
run;

proc sort data=g_dsin out=fmtdata1 (keep=PARAMN) nodupkey;
   by PARAMN;
run;

proc sort data=g_dsin out=fmtdata2 (keep=AVISITN) nodupkey;
   by AVISITN;
run;

data fin_dat1 (keep=usubjid newtrtn newtrt);
```

```
  set g_dsin (where=(newtrtn is not missing)) end=eof;

  if eof then
    output;
run;

proc sql noprint;
  create table fin_dat2 as select * from (select * from fin_dat1) , (select *
    from fmtdata1) , (select * from fmtdata2);
quit;

proc sort data=fin_dat2;
  by usubjid newtrtn newtrt PARAMN AVISITN;
run;

proc sort data=g_dsin out=tmp_gdsin;
  by usubjid newtrtn newtrt PARAMN AVISITN;
run;

data FINAL;
  merge tmp_gdsin(in=a) fin_dat2(in=b);
  by usubjid newtrtn newtrt PARAMN AVISITN;

  if ^a and b then
    _extrabyval=1;
run;

data _data1;
  set final;
  where (NEWTRTN is not missing) and avisitn le 6 and EVIMMFL eq 'Y' and paramn
    in (1) and avisitn>1 and PEDIMMFL="Y" and EV1MD2FL="Y" and ANL01FL="Y";
run;

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;
      _str=NEWTRT;

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

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

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

data _bydat1;
  set _bydat1 end=eof;
  by _datasrt PARAMN AVISITN;
  retain _preby 0;
  drop _preby;
  length _bylab1-_bylab2 $100;
  retain _byvar1-_byvar2 0 _bylen1-_bylen2 0 _bylab1-_bylab2;

  if first.PARAMN then
    do;
      _byvar2=0;
    end;

  if first.PARAMN then
    do;
      _byvar1 + 1;
      _bylab1=put(PARAMN, PARAMN.);
      _bylen1=max(_bylen1, length(_bylab1));
    end;

  if first.AVISITN then
    do;
      _byvar2 + 1;
      _bylab2=put(AVISITN, VISX.);
      _bylen2=max(_bylen2, length(_bylab2));
    end;
  output;

  if last.AVISITN then
    do;

      if _byvar2 > _preby then
        _preby=_byvar2;
      call symput("_prebyl", compress(put(_preby, 4.)));
    end;

  if eof then
```

```
    do;
      call symput("_preby1", compress(put(_byvar1, 4.)));

      if 2=0 then
        output;
    end;
run;

data _bydat1;
  set _bydat1;
  by _datasrt;
  length _bycol _byindnt $50 _bylast $10;
  _bycol="1 2 ";
  _byindnt="0 0 ";
  _bylast=" ";
run;

proc sort data=_bydat1 out=_Byfrm2(keep=_datasrt AVISITN) nodupkey;
  by _datasrt AVISITN;
run;

data _Byfrm2;
  set _byfrm2;
  by _datasrt AVISITN;
  retain _nwbyvar2 0;

  if first.AVISITN then
    do;
      _nwbyvar2 + 1;
    end;
run;

proc sort data=_bydat1;
  by _datasrt AVISITN;
run;

data _bydat1(drop=_nwbyvar2);
  merge _bydat1 _byfrm2;
  by _datasrt AVISITN;
  _byvar2=_nwbyvar2;
run;

proc sort data=_bydat1;
  by _datasrt PARAMN AVISITN;
run;

proc print data=_bydat1;
  title 'print of _bydat1';
run;

proc sort data=_data1 out=_data1;
  by _datasrt PARAMN AVISITN;
run;
```

```
*------------------------------------------------------------------;
* Merge calculated by variables back into _DATAn dataset. ;
*------------------------------------------------------------------;

data _data1;
  merge _bydat1(keep=_datasrt _byvar1 _byvar2 PARAMN AVISITN) _data1(in=_b);
  by _datasrt PARAMN AVISITN;

  if _b;
run;

proc sort data=_data1;
  by _datasrt _byvar1 _byvar2;
run;

data _tmpdata1;
  set _data1;
  ;
  output;
run;

data _trtsubgrpframe;
  _cat=1;
  _trt=1;
  output;
  _trt=2;
  output;
run;

proc sql noprint;
  create table _fullbyvar as select * from (select distinct _trt, _cat from
    _trtsubgrpframe), (select distinct _byvar1, _byvar2 from _bydat1) order by
    _trt, _byvar1, _byvar2;
quit;

proc sql;
  create table _bydatn1 as select distinct _trt, _byvar1, count(distinct
    USUBJID) as byvar1n from _tmpdata1 group by _trt, _byvar1;
quit;

data _fullbyvar;
  merge _fullbyvar(in=a) _bydatn1;
  by _trt _byvar1;

  if a;

  if missing(byvar1n) then
    byvar1n=0;
run;

proc sql;
  create table _bydatn12 as select distinct _trt, _byvar1, _byvar2,
    count(distinct USUBJID) as byvar12n from _tmpdata1 group by _trt, _byvar1,
    _byvar2;
```

```
quit;

data _fullbyvar;
  merge _fullbyvar(in=a) _bydatn12;
  by _trt _byvar1 _byvar2;

  if a;

  if missing(byvar12n) then
    byvar12n=0;
run;

data _byvardata1;
  set _fullbyvar;

  if _trt=9999 then
    _trt=3;
run;

proc sort data=_fullbyvar out=_bylab nodupkey;
  by _byvar1 _byvar2 _trt;
run;

proc sort data=_bydat1 out=_byframe;
  by _byvar1 _byvar2;
run;

data _bylabel1;
  merge _byframe(in=a) _bylab;
  by _byvar1 _byvar2;

  if a;
run;

data _bylabel1(keep=_datasrt _trt _byvar1 _byvar2 _bylab1 byvar1n);
  set _bylabel1;
  _bylab1=strip(_bylab1)||" (N="||strip(put(byvar1n, 5.))||")";
run;

data _data12;
  set _data1;

  if aval>0 then
    do;
      log_aval=log(aval);
    end;
run;

proc sort data=_data12;
  by _DATASRT _byvar1 _byvar2;
run;

ods output ConfLimits=conf Equality=equal (keep=_DATASRT _byvar1 _byvar2 ProbF);
```

```
PROC TTEST DATA=_data12 plots=none;
  TITLE "T-test Example";
  by _DATASRT _byvar1 _byvar2;
  class _trt;
  VAR log_aval;
RUN;

data conf1;
  set conf;
  where method ~=' ';
run;

data stat;
  merge conf1 equal;
  by _DATASRT _byvar1 _byvar2;
run;

data a1;
  set stat;
  where (ProbF > 0.05 and method="Pooled") or (ProbF <=0.05 and
    method="Satterthwaite");
  geomean=exp(mean);
  lcl=exp(lowerclmean);
  ci="("||strip(put(exp(lowerclmean), 8.2))||", "||strip(put(exp(upperclmean),
    8.2))||")";
  format geomean 8.2;
  keep _datasrt _byvar: geomean lcl ci;
run;

data ttest;
  set a1;
  _CVALUE8=strip(put(round(geomean, 0.01), 8.2)) ||'~-2n'||strip(ci);
  labeln=1;
  labelr=3;
  output;

  if lcl>0.67 then
    _cvalue8="Y";
  else
    _cvalue8="N";
  labeln=2;
  labelr=4;
  output;
run;

proc sort;
  by _DATASRT _byvar1 _byvar2 labeln;
run;

data _anal1;
  set _data1;
  where _trt > 0;
  _blcksrt=0;
  output;
```

```
run;

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

proc sort data=_anal1;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

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

proc univariate data=_anal1 noprint;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
  var AVAL;
  output out=_msum1 N=N;
run;

proc sort data=_anal1 out=_catby1(keep=_byvar1 _byvar2) nodupkey;
  by _byvar1 _byvar2;
run;

data _frame1;
  set _catby1(keep=_byvar1 _byvar2);
  _datasrt=1;
  _blcksrt=0;
  _catord=1;
  _trt=1;
  _cat=1;
  output;
  _trt=2;
  _cat=1;
  output;
run;

proc sort data=_frame1;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

data _msum1;
  merge _msum1 _frame1;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

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

data _result1_1;
  if 0 then
    set _basetemplate;
  set _msum1 end=eof;
  _rowsrt=0 + 1;
  _rwlabel="N ";
  _cvalue=' ';
  _nvalue=.;
  _nvalue=N;

  if N ne . then
    _cValue=strip(put(N, 5.) );
  else
    _cValue=strip(put(0, 5.) );
  output;
run;

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

data _logresult1_1;
  if 0 then
    set _basetemplate;
  stop;
run;

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

data _result2_1;
  if 0 then
    set _basetemplate;
  stop;
run;

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

data _logresult2_1;
  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 _base1;
  set _result1_1 _result2_1 _logresult1_1 _logresult2_1;
  ;

  if _trt=3 then
```

```
    _trt=9999;
  _varname="AVAL";
  _vrlabel=" ";
  _datatyp='data';
  _module='msumstat';
  _indent=0;
  _rowjump=1;
  _dptindt=0;
run;

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

proc sort data=_base1;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt;
run;

*----------------------------------------------------------------------;
* Create the _ANALxx dataset for the use of later analysis. ;
*----------------------------------------------------------------------;

data _anal2;
  set _data1;
  where _trt > 0;
  _blcksrt=0;
  output;
run;

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

proc sort data=_anal2;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

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

Data _anal2;
  set _anal2;

  if AVAL gt 0 then
    do;
      _logvar=log10(AVAL);
    end;
run;

proc univariate data=_anal2 noprint;
  ;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
```

```
   var _logvar;
   output out=_logmsum2 MEAN=MEAN;
run;

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

proc sort data=_anal2 out=_catby2(keep=_byvar1 _byvar2) nodupkey;
   by _byvar1 _byvar2;
   ;
run;

data _frame2;
   set _catby2(keep=_byvar1 _byvar2);
   _datasrt=1;
   _blcksrt=0;
   _catord=1;
   _trt=1;
   _cat=1;
   output;
   _trt=2;
   _cat=1;
   output;
run;

proc sort data=_frame2;
   by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

data _logmsum2;
   merge _logmsum2 _frame2;
   by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

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

data _result1_2;
   if 0 then
      set _basetemplate;
   stop;
run;

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

data _logresult1_2;
   if 0 then
      set _basetemplate;
   set _logmsum2 end=eof;
```

```
  _rowsrt=1 + 1;
  _rwlabel="Geometric Mean ";
  _cvalue=' ';
  _nvalue=.;

  if mean ne . then
    _cValue=strip(put(10 ** (mean), 9.1) );
  else
    _cValue="NE";
  output;
run;

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

data _result2_2;
  if 0 then
    set _basetemplate;
  stop;
run;

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

data _logresult2_2;
  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 _base2;
  set _result1_2 _result2_2 _logresult1_2 _logresult2_2;
  ;

  if _trt=3 then
    _trt=9999;
  _varname="AVAL";
  _vrlabel=" ";
  _datatyp='data';
  _module='msumstat';
  _indent=0;
  _rowjump=1;
  _dptindt=0;
run;

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

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

proc sort data=_base2;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt;
run;

options MPRINT SYMBOLGEN MLOGIC;
;

data _base2;
  set _base2;
  _cvalue2=_cvalue;
run;

proc sort data=_base1 out=_ds1;
  by _datasrt _byvar1 _byvar2 _cat _trt;
run;

proc sort data=_base2 out=_ds2;
  by _datasrt _byvar1 _byvar2 _cat _trt;
run;

data _base2;
  merge _ds1(in=d1) _ds2(in=d2 keep=_datasrt _trt _cvalue2 _cat _byvar1 _byvar2);
  by _datasrt _byvar1 _byvar2 _cat _trt;

  if d1;
run;

data _base1;
  set _base1;
  delete;
run;

*-------------------------------------------------------------------;
* Create the _ANALxx dataset for the use of later analysis. ;
*-------------------------------------------------------------------;

data _anal3;
  set _data1;
  where _trt > 0;
  _blcksrt=0;
  output;
run;

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

proc sort data=_anal3;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

ods listing close;
```

```
Data _anal3;
  set _anal3;

  if AVAL gt 0 then
    do;
      _logvar=log10(AVAL);
    end;
run;

proc univariate data=_anal3 cibasic(type=TWOSIDED alpha=0.05);
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
  var _logvar;
  ods output basicIntervals=_logmsumbi3;
run;

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

proc sort data=_anal3 out=_catby3(keep=_byvar1 _byvar2) nodupkey;
  by _byvar1 _byvar2;
  ;
run;

data _frame3;
  set _catby3(keep=_byvar1 _byvar2);
  _datasrt=1;
  _blcksrt=0;
  _catord=1;
  _trt=1;
  _cat=1;
  output;
  _trt=2;
  _cat=1;
  output;
run;

proc sort data=_frame3;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt;
run;

proc sql;
  create table __frame3 as select distinct a.parameter, b.* from _frame3 b,
    _logmsumbi3 a;
quit;

proc sort data=__frame3;
  by _datasrt _byvar1 _byvar2 _blcksrt _trt parameter;
run;

data _logmsumbi3;
```

```
   merge _logmsumbi3 __frame3;
   by _datasrt _byvar1 _byvar2 _blcksrt _trt parameter;
run;

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

data _result1_3;
   if 0 then
      set _basetemplate;
   stop;
run;

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

data _logresult1_3;
   if 0 then
      set _basetemplate;
   stop;
run;

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

data _result2_3;
   if 0 then
      set _basetemplate;
   stop;
run;

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

data _logresult2_3;
   if 0 then
      set _basetemplate;
   set _logmsumbi3;
   *---------------------------------------------------------------------;
   * Recode PARAMETER variable to match keywords used by MSUMSTAT. ;
   *---------------------------------------------------------------------;
   parameter=upcase(parameter);

   if parameter=: 'MEAN' then
      parameter='MEAN_CI';
   _rowsrt=2 + 1;
   _rwlabel="95% Confidence|Interval of Geometric Mean ";
   _cvalue=' ';
   _nvalue=.;
   *---------------------------------------------------------------------;
```

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

  if parameter='MEAN_CI' then
    do;

      if lowerCL ne . then
        _cValue=strip(put(10 ** (lowerCL), 9.1) );
      else if lowerCL eq . then
        _cValue="NE";

      if upperCL ne . then
        _cValue=strip(_cValue) || ', ' || strip(put(10 ** (upperCL), 9.1) );
      else if upperCL eq . then
        _cValue=strip(_cValue) || ', ' ||"NE";
      _cValue='(' || strip(_cValue) || ')';
      output;
    end;
run;

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

data _base3;
  set _result1_3 _result2_3 _logresult1_3 _logresult2_3;

  if _trt=3 then
    _trt=9999;
  _varname="AVAL";
  _vrlabel=" ";
  _datatyp='data';
  _module='msumstat';
  _indent=0;
  _rowjump=1;
  _dptindt=0;
run;

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

proc sort data=_base3;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt;
run;

data _base3;
  set _base3;
  _cvalue3=tranwrd(_cvalue, ", ", ",~{nbspace 1}");
run;

proc sort data=_base2 out=_ds1;
  by _datasrt _byvar1 _byvar2 _cat _trt;
```

```
run;

proc sort data=_base3 out=_ds2;
  by _datasrt _byvar1 _byvar2 _cat _trt;
run;

data _base3;
  merge _ds1(in=d1) _ds2(in=d2 keep=_datasrt _trt _cvalue3 _cat _byvar1 _byvar2);
  by _datasrt _byvar1 _byvar2 _cat _trt;

  if d1;
run;

data _base2;
  set _base2;
  delete;
run;

data _final;
  set _base1 _base2 _base3;
run;

proc sort data=_final;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt;
run;

*-------------------------------------------------------------------;
* Collect BY-VARS ;
*-------------------------------------------------------------------;

data _bydata;
  set _bydat1;

  if _byvar1=0 then
    delete;
run;

proc sort data=_bydata;
  by _datasrt _byvar1 _byvar2;
run;

data _final;
  merge _bydata _final(in=_b);
  by _datasrt _byvar1 _byvar2;

  if _b;
run;

data _bylabeldata;
  set _bylabel1;
run;

proc sort data=_bylabeldata;
  by _datasrt _trt _byvar1 _byvar2;
```

```
run;

proc sort data=_final;
  by _datasrt _trt _byvar1 _byvar2;
run;

data _final;
  merge _final(in=_a) _bylabeldata(drop=_bylab1);
  by _datasrt _trt _byvar1 _byvar2;

  if _a;
run;

*-------------------------------------------------------------------;
* Both TRT and Stats horizonal;
*-------------------------------------------------------------------;

proc sql noprint;
  select max(_rowsrt) into :rowmax from _final where _datatyp='data';
quit;

data _final;
  set _final;
  drop __trt;

  if _datatyp='data' then
    do;

      if _trt=9999 then
        __trt=2 + 1;
      else
        __trt=_trt;

      if __trt > 1 then
        _column=_rowsrt + (__trt - 1) * 1;
      else
        _column=_rowsrt;
      _proc_report_column=_rowsrt;
    end;
  _rowsrt=1;

  if _rwlabel ne ' ' then
    _colabel=translate(trim(_rwlabel), '^', ' ');
  _rwlabel=' ';
run;

proc sort data=_final out=_final;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt _column;
run;

data _linecnt;
  set _final end=eof;
  by _datasrt _byvar1 _byvar2 _blcksrt _rowsrt _column;
  retain _totline _maxval _maxrow _rwlbtag _vrlbtag 0 _maxline _linecnt;
```

```
keep _datasrt _byvar1 _byvar2 _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;
  *--------------------------------------------------------------------;
  * 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;

      if _vrlabel ne ' ' then
        _totline=_totline - 1;
      output;
    end;
run;

data _final;
  length _direct $20;
  _direct=' ';
  merge _final _linecnt;
  by _datasrt _byvar1 _byvar2 _blcksrt;
run;

data _final;
  length _cvalue4 $100.;
  set _final;
  _cvalue4=strip(_cvalue2) ||'~-2n'||strip(_cvalue3);
run;

proc sort data=_final out=_final_ods;
  by _trt _byvar1 _bylab1 _byvar2 _bylab2;
run;
```

```
data _trt;
  attrib _trtlbl length=$50.;
  _trtn=1;
  _trtlbl="12-15 Years";
  output;
  _trtn=2;
  _trtlbl="16-25 Years";
  output;
run;

data _trt_fmt;
  set _trt;
  by _trtn;
  FMTNAME="TREAT";
  call symputx('_trt'||strip(put(_trtn, best.)), _trtlbl, 'G');
run;

data _trt_fmt(keep=fmtname _trtnum _trtlbl rename=(_trtnum=start
    _trtlbl=label)) _trt_map(keep=_trtn _trtnum rename=(_trtn=_trt) );
  set _trt_fmt;
  by _trtn;
  _trtnum=_N_;
run;

proc format cntlin=work._trt_fmt noprint;
  select treat;
run;

data _final;
  set _final_ods;
run;

data outdata (drop=_trt rename=(_trtnum=_trt));
  merge _final _trt_map(in=a);
  by _trt;

  if a;
  _fixvar=1;
  _fix2var=1;
  _dummy=1;
run;

proc sort data=outdata;
  by _FIXVAR _FIX2VAR _DATASRT _byvar1 _bylab1 _byvar2 _bylab2 _BLCKSRT _ROWSRT
    _TRT _DUMMY;
run;

proc transpose data=outdata out=trans;
  by _FIXVAR _FIX2VAR _DATASRT _byvar1 _bylab1 _byvar2 _bylab2 _BLCKSRT _ROWSRT
    _TRT _DUMMY;
  var _CVALUE _cvalue4;
run;

data trans;
```

```
  length label $100.;
  set trans;

  if upcase(_name_)="_CVALUE" then
    do;
      labeln=1;
      label="n~{super c}";
    end;
  else if upcase(_name_)="_CVALUE4" then
    do;
      labeln=2;
      label="GMT~{super d} ~-2n (95%(*ESC*){nbspace 1}CI~{super d})";
    end;
run;

proc format;
  value par 1="n~{super b}"
    2="GMT~{super c} ~-2n (95%(*ESC*){nbspace 1}CI~{super c})"
    3="GMR~{super d} ~-2n (95%(*ESC*){nbspace 1}CI~{super d})"
    4="Met Noninferiority Objective~{super e} ~-2n (Y/N)";
run;

proc sort data=trans;
  by _FIXVAR _FIX2VAR _DATASRT _byvar1 _bylab1 _byvar2 _bylab2 _BLCKSRT _ROWSRT
    labeln label _DUMMY;
run;

proc transpose data=trans out=final prefix=TRT;
  by _FIXVAR _FIX2VAR _DATASRT _byvar1 _bylab1 _byvar2 _bylab2 _BLCKSRT _ROWSRT
    labeln label _DUMMY;
  id _trt;
  var col1;
run;

proc sort data=final;
  BY _DATASRT _byvar1 _byvar2 labeln;
run;

data merg;
  merge final ttest;
  BY _DATASRT _byvar1 _byvar2 labeln;
run;

data final;
  set merg;
run;

ods escapechar="~";
ods html file="&prot./analysis/esub/output/adva-s001-gmr-ped-ev-eval.html";
title1 "Summary of Geometric Mean Ratio (*ESC*){Unicode 2013} NT50 (*ESC*){Unicode 2013}";
title2 "Comparison of Subjects 12 Through 15 Years of Age to Subjects 16 Through 25 Years of Age (Immunogenicity
Subset) (*ESC*){Unicode 2013}";
title3 "Subjects Without Evidence of Infection up to 1 Month After Dose 2 (*ESC*){Unicode 2013} Dose 2 Evaluable
Immunogenicity Population";
```

```
footnote1 "Abbreviations: GMR = geometric mean ratio; GMT = geometric mean titer; LLOQ = lower limit of
quantitation;";
footnote2 "NT50 = 50% neutralizing titer; SARS-CoV-2 = severe acute respiratory syndrome coronavirus 2.";
footnote3 "Note: Subjects who had no serological or virological evidence (up to 1 month after receipt of the last dose) of
past SARS-CoV-2 infection (ie, N-binding antibody [serum] negative at Visit 1 and SARS-CoV-2 not detected by
NAAT [nasal swab] at Visits 1 and 2), and had negative NAAT (nasal swab) at any unscheduled visit up to 1 month
after Dose 2 were included in the analysis.";
footnote4
  "a.(*ESC*){nbspace 5}Protocol-specified timing for blood sample collection.";
footnote5 "b.(*ESC*){nbspace 5}n = Number of subjects with valid and determinate assay results for the specified
assay at the given dose/sampling time point.";
footnote6 "c.(*ESC*){nbspace 5}GMTs and 2-sided 95% CIs were calculated by exponentiating the mean logarithm of
the titers and the corresponding CIs (based on the Student t distribution). Assay results below the LLOQ were set to 0.5
~{unicode 00D7} LLOQ.";
footnote7 "d.(*ESC*){nbspace 5}GMRs and 2-sided 95% CIs were calculated by exponentiating the mean difference of
the logarithms of the titers (Group 1 [12-15 years] (*ESC*){Unicode 2013} Group 2 [16-25 years]) and the
corresponding  CI (based on the Student t distribution).";
footnote8 "e.(*ESC*){nbspace 5}Noninferiority is declared if the lower bound of the 2-sided 95% CI for the GMR is
greater than 0.67.";

proc report data=final nowd list missing nocompletecols contents="" split="|"
    style(report)={} style(header)={} style(column)={};
  COLUMN (_FIXVAR _FIX2VAR _DATASRT _byvar1 _bylab1 _byvar2 _bylab2 _BLCKSRT _ROWSRT ("~S=
{just=center}Vaccine Group (as Randomized) ~{line}" ("BNT162b2 (30 (*ESC*){unicode 03BC}g)~{line}" (trt1,
    labeln) (trt2, labeln)) ) ("" ("" (_CVALUE8, labelr)) ) _DUMMY);
  define _fixvar / group order=internal noprint;
  define _fix2var / group order=internal noprint;
  DEFINE _DATASRT / GROUP FORMAT=BEST9. WIDTH=9 SPACING=2 NOPRINT RIGHT
    ORDER=INTERNAL "_DATASRT";
  define _byvar1/ group order=internal noprint;
  define _bylab1 / group order=internal "Assay"
    style(column)={just=left}style(header)={just=left};
  define _byvar2/ group order=internal noprint;
  define _bylab2 / group order=internal "Dose/|Sampling|Time Point~{super a}"
    style(column)={just=left}style(header)={just=left};
  DEFINE _BLCKSRT / GROUP FORMAT=BEST9. WIDTH=9 SPACING=2 NOPRINT RIGHT
    ORDER=INTERNAL;
  DEFINE _ROWSRT / GROUP FORMAT=BEST9. WIDTH=9 SPACING=2 NOPRINT RIGHT
    ORDER=INTERNAL;
  DEFINE labeln / ACROSS WIDTH=25 format=par. SPACING=2 RIGHT ORDER=internal " ";
  DEFINE labelr / ACROSS WIDTH=25 format=par. SPACING=2 RIGHT ORDER=internal " ";
  DEFINE trt1 / GROUP FORMAT=$100. WIDTH=30 SPACING=2 NOZERO center
    "12-15 Years";
  DEFINE trt2 / GROUP FORMAT=$100. WIDTH=30 SPACING=2 NOZERO center
    "16-25 Years";
  DEFINE _CVALUE8 / GROUP FORMAT=$30. WIDTH=30 SPACING=2 NOZERO CENTER
    "12-15 Years/16-25 Years";
  DEFINE _DUMMY / SUM FORMAT=BEST9. WIDTH=9 SPACING=2 NOPRINT RIGHT "_DUMMY";
  BREAK BEFORE _FIX2VAR /;
  BREAK BEFORE _FIXVAR / PAGE CONTENTS="";
  COMPUTE BEFORE _FIX2VAR;
    line @1 ' ';
  ENDCOMP;
RUN;
```

```
ods HTML close;

proc printto;
run;
```