```
*************************************************************************************************;
**  Program Name    :  adsl-fu-d21-ped6.sas                                                     **;
**  Date Created    :  15Nov2021                                                              **;
**  Programmer Name :  (b) (4), (b) (6)                                                       **;
**  Purpose         :  Create adsl-fu-d21-ped6                                                **;
**  Input data      :  adsl                                                                   **;
**  Output file     :  adsl-fu-d21-ped6.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-fu-d21-ped6;

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;

data adsl;
	set DATVPROT.ADSL;
	length FUPCAT1$ 100 FUPCAT2$ 100;

	/*  Total exposure from Dose 2 to cutoff date */
	if .<FUP2CA1N<=2 then
		do;
			FUPCAT1N=1;
			FUPCAT1="(*ESC*){unicode 003c}2 Months";
		end;

	if 2<FUP2CA1N<=4 then
		do;
			FUPCAT1N=2;
			FUPCAT1="(*ESC*){unicode 2265}2-(*ESC*){unicode 003c}4 Months";
		end;

	if 4<FUP2CA1N<=6 then
		do;
```

```
            FUPCAT1N=3;
            FUPCAT1="(*ESC*){unicode 2265}4-(*ESC*){unicode 003c}6 Months";
      end;

if 4<FUP2CA1N<=6 then
      do;
            FUPCAT1N=3;
            FUPCAT1="(*ESC*){unicode 2265}4-(*ESC*){unicode 003c}6 Months";
      end;

if 6<FUP2CA1N<=8 then
      do;
            FUPCAT1N=4;
            FUPCAT1="(*ESC*){unicode 2265}6-(*ESC*){unicode 003c}8 Months";
      end;

if 8<FUP2CA1N<=10 then
      do;
            FUPCAT1N=5;
            FUPCAT1="(*ESC*){unicode 2265}8-(*ESC*){unicode 003c}10 Months";
      end;

if 10<FUP2CA1N then
      do;
            FUPCAT1N=6;
            FUPCAT1="(*ESC*){unicode 2265}10 Months";
      end;

/*  Original vaccination period (prior to unblinding) */
if .<FUP2CA2N<=2 then
      do;
            FUPCAT2N=1;
            FUPCAT2="(*ESC*){unicode 003c}2 Months";
      end;

if 2<FUP2CA2N<=4 then
      do;
            FUPCAT2N=2;
            FUPCAT2="(*ESC*){unicode 2265}2-(*ESC*){unicode 003c}4 Months";
      end;

if 4<FUP2CA2N<=6 then
      do;
            FUPCAT2N=3;
            FUPCAT2="(*ESC*){unicode 2265}4-(*ESC*){unicode 003c}6 Months";
      end;

if 6<FUP2CA2N then
      do;
            FUPCAT2N=4;
            FUPCAT2="(*ESC*){unicode 2265}6 Months";
      end;
FUP2CUT_=FUP2CUT/28;
FUP2UNB_=FUP2UNB/28;
```

```
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;

data _subGrpData(compress=no);
    delete;
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;

data _data1;
```

```
    set g_adsl_dsin;
    where (NEWTRTN is not missing);
run;

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

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 _null_;
      set _data1 end=eof;

      if eof then
```

```
        call symput('dptlab', vlabel(FUPCAT2N));
run;

data _anal1;
    length FUPCAT2N 8;
    set _data1;
    where same and FUPCAT2N is not missing;
    _blcksrt=1;
    _cnt=1;
    _cat=1;

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

proc sort data=_anal1;
    by _datasrt _blcksrt FUPCAT2N _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 FUPCAT2N _trt USUBJID;
run;

proc freq data=_temp91;
    format FUPCAT2N;
    tables _datasrt*_blcksrt*_cat * FUPCAT2N * _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;

data _denomf1;
      _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=_denom1 out=_denomin1(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="_PCT1");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;

proc sort data=_pct1 out=_expv1 (keep=_datasrt _blcksrt FUPCAT2N) nodupkey;
      by _datasrt _blcksrt FUPCAT2N;
run;

proc sort data=_expv1;
      by _datasrt _blcksrt FUPCAT2N;
run;

proc sort data=_anal1 out=_catlabel1 (keep=_datasrt _blcksrt FUPCAT2N FUPCAT2)
            nodupkey;
      by _datasrt _blcksrt FUPCAT2N;
      ;
run;

data _expv1;
      merge _expv1 (in=_a) _catlabel1 (in=_b);
      by _datasrt _blcksrt FUPCAT2N;

      if _a;
run;

proc sql noprint;
      select put(nobs - delobs, 12.) into :_nobs from dictionary.tables
            where (libname="WORK" and memname="_PCT1");
      select setting into :miss from dictionary.options where
            upcase(optname)="MISSING";
quit;
```

```
proc sort data=_expv1;
	by _datasrt _blcksrt FUPCAT2N;
run;

data _frame1;
	set _expv1;
	by _datasrt _blcksrt FUPCAT2N;
	length _catLabl $100;
	_catLabl=FUPCAT2;

	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=_frame1;
	by _datasrt _blcksrt _cat FUPCAT2N _trt;
run;

proc sort data=_pct1;
	by _datasrt _blcksrt _cat FUPCAT2N _trt;
run;

data _pct1;
	merge _frame1(in=_inframe) _pct1;
	by _datasrt _blcksrt _cat FUPCAT2N _trt;

	if _inframe;

	if count=. then
		count=0;
run;

proc sort data=_pct1;
	by _datasrt _blcksrt FUPCAT2N;
run;

data _miss1(keep=_datasrt _blcksrt FUPCAT2N totcount);
	set _pct1;
	where FUPCAT2N=9998;
```

```
	retain totcount;
	by _datasrt _blcksrt FUPCAT2N;

	if first.FUPCAT2N then
		totcount=0;
	totcount=totcount+count;

	if last.FUPCAT2N;
run;

data _pct1(drop=totcount);
	merge _pct1 _miss1;
	by _datasrt _blcksrt FUPCAT2N;

	if totcount=0 then
		delete;
run;

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;

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

	if _a;
	_varname="FUPCAT2N ";
	_vrlabel="Original blinded placebo-controlled follow-up period ";
	_rwlabel=_catLabl;

	if FUPCAT2N=9998 then
```

```
        do;
            _rwlabel="Missing ";
            _catord=9998;
        end;
    else if FUPCAT2N=9999 then
        do;
            _rwlabel="Total ";
            _catord=9999;
        end;

    if _catord=. then
        _catord=9997;
run;

proc sort data=_pct1;
    by _datasrt _blcksrt _catord FUPCAT2N _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 FUPCAT2N _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.FUPCAT2N 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=4;
        _dptindt=0;

        if _trt=3 +1 then
                _trt=9999;

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

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

data _anal2;
        set _data1;
        where _trt > 0;
        _blcksrt=1;
        output;
run;

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

```
proc sort data=_anal2;
      by _datasrt _blcksrt _trt;
run;

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

proc univariate data=_anal2 noprint;
      ;
      by _datasrt _blcksrt _trt;
      var FUP2UNB_;
      output out=_msum2 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 _frame2;
      set _bydat1(keep=);
      _datasrt=1;
      _blcksrt=1;
      _catord=1;
      _trt=1;
      _cat=1;
      output;
      _trt=2;
      _cat=1;
      output;
      _trt=3;
      _cat=1;
      output;
run;

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

data _msum2;
      merge _msum2 _frame2;
      by _datasrt _blcksrt _trt;
run;
```

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

data _result1_2;
	if 0 then
		set _basetemplate;
	set _msum2 end=eof;
	_rowsrt=4 + 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=4 + 2;
	_rwlabel="Median ";
	_cvalue=' ';
	_nvalue=.;
	_nvalue=MEDIAN;

	if MEDIAN ne . then
		_cValue=strip(put(MEDIAN, 5.1) );
	else
		_cValue="-";
	output;
	_rowsrt=4 + 3;
	_rwlabel="Min, max ";
	_cvalue=' ';
	_nvalue=.;
	*-------------------------------------------------------------------;
	* MINMAX MINMAXC MEDIAN(MINMAX) MEDIAN(MINMAXC) ;
	*-------------------------------------------------------------------;
	_cValue=' ';

	if min ^=. & max ^=. then
		do;
			_cValue=trim(_cvalue) || ' (' || strip(put(min, 5.1)
				)|| ', ' || strip(put(max, 5.1) )||')';
		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_2;
        if 0 then
            set _basetemplate;
        stop;
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=4 then
            _trt=9999;
        _varname="FUP2UNB_";
        _vrlabel=" ";
        _datatyp='data';
        _module='msumstat';
        _indent=4;
        _rowjump=1;
        _dptindt=0;
```

```
run;

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

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

data _base2;
      set _base2;
      _vhlabel=" ";
run;

data _anal3;
      length FUPCAT1N 8;
      set _data1;
      where same and FUPCAT1N is not missing;
      _blcksrt=2;
      _cnt=1;
      _cat=1;

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

proc sort data=_anal3;
      by _datasrt _blcksrt FUPCAT1N _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 FUPCAT1N _trt USUBJID;
      ;
run;

proc freq data=_temp93;
      format FUPCAT1N;
      tables _datasrt*_blcksrt*_cat * FUPCAT1N * _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;

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;

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

proc sort data=_pct3 out=_expv3 (keep=_datasrt _blcksrt FUPCAT1N) nodupkey;
	by _datasrt _blcksrt FUPCAT1N;
run;

proc sort data=_expv3;
	by _datasrt _blcksrt FUPCAT1N;
run;

proc sort data=_anal3 out=_catlabel3 (keep=_datasrt _blcksrt FUPCAT1N FUPCAT1)
		nodupkey;
	by _datasrt _blcksrt FUPCAT1N;
	;
run;

data _expv3;
	merge _expv3 (in=_a) _catlabel3 (in=_b);
	by _datasrt _blcksrt FUPCAT1N;

	if _a;
run;
```

```
proc sort data=_expv3;
	by _datasrt _blcksrt FUPCAT1N;
run;

data _frame3;
	set _expv3;
	by _datasrt _blcksrt FUPCAT1N;
	length _catLabl $100;
	_catLabl=FUPCAT1;

	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=_frame3;
	by _datasrt _blcksrt _cat FUPCAT1N _trt;
run;

proc sort data=_pct3;
	by _datasrt _blcksrt _cat FUPCAT1N _trt;
run;

data _pct3;
	merge _frame3(in=_inframe) _pct3;
	by _datasrt _blcksrt _cat FUPCAT1N _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 FUPCAT1N;
```

```
run;

data _miss3(keep=_datasrt _blcksrt FUPCAT1N totcount);
      set _pct3;
      where FUPCAT1N=9998;
      retain totcount;
      by _datasrt _blcksrt FUPCAT1N;

      if first.FUPCAT1N then
            totcount=0;
      totcount=totcount+count;

      if last.FUPCAT1N;
run;

data _pct3(drop=totcount);
      merge _pct3 _miss3;
      by _datasrt _blcksrt FUPCAT1N;

      if totcount=0 then
            delete;
run;

*-------------------------------------------------------------------------;
* 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="FUPCAT1N ";
	_vrlabel="Total follow-up period from Dose 2 to cutoff date ";
	_rwlabel=_catLabl;

	if FUPCAT1N=9998 then
		do;
			_rwlabel="Missing ";
			_catord=9998;
		end;
	else if FUPCAT1N=9999 then
		do;
			_rwlabel="Total ";
			_catord=9999;
		end;

	if _catord=. then
		_catord=9997;
run;

proc sort data=_pct3;
	by _datasrt _blcksrt _catord FUPCAT1N _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 FUPCAT1N _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.FUPCAT1N 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=4;
    _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 _base3;
      set _base3;
      where _trt eq 1;
run;

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

data _anal4;
      set _data1;
      where _trt > 0;
      _blcksrt=2;
      output;
run;

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

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

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

proc univariate data=_anal4 noprint;
      ;
      by _datasrt _blcksrt _trt;
      var FUP2CUT_;
      output out=_msum4 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 _frame4;
      set _bydat1(keep=);
      _datasrt=1;
      _blcksrt=2;
      _catord=1;
```

```
        _trt=1;
        _cat=1;
        output;
        _trt=2;
        _cat=1;
        output;
        _trt=3;
        _cat=1;
        output;
run;

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

data _msum4;
        merge _msum4 _frame4;
        by _datasrt _blcksrt _trt;
run;

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

data _result1_4;
        if 0 then
                set _basetemplate;
        set _msum4 end=eof;
        _rowsrt=6 + 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=6 + 2;
        _rwlabel="Median ";
        _cvalue=' ';
        _nvalue=.;
        _nvalue=MEDIAN;

        if MEDIAN ne . then
```

```
        _cValue=strip(put(MEDIAN, 5.1) );
    else
        _cValue="-";
    output;
    _rowsrt=6 + 3;
    _rwlabel="Min, max ";
    _cvalue=' ';
    _nvalue=.;
    *-------------------------------------------------------------------;
    * MINMAX MINMAXC MEDIAN(MINMAX) MEDIAN(MINMAXC) ;
    *-------------------------------------------------------------------;
    _cValue=' ';

    if min ^=. & max ^=. then
        do;
            _cValue=trim(_cvalue) || ' (' || strip(put(min, 5.1)
                )|| ', ' || strip(put(max, 5.1) )||')';
        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_4;
    if 0 then
        set _basetemplate;
    stop;
run;

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

data _result2_4;
    if 0 then
        set _basetemplate;
    stop;
run;

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

data _logresult2_4;
    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 _base4;
      set _result1_4 _result2_4 _logresult1_4 _logresult2_4;
      ;

      if _trt=4 then
            _trt=9999;
      _varname="FUP2CUT_";
      _vrlabel=" ";
      _datatyp='data';
      _module='msumstat';
      _indent=4;
      _rowjump=1;
      _dptindt=0;
run;

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

proc sort data=_base4;
      by _datasrt _blcksrt _rowsrt;
run;

data _base4;
      set _base4;
      where _trt eq 1;
run;

*******************************************************************************;
*SPECIFICATION 3 -1) titles and footnotes *;
* 2) display *;
*******************************************************************************;
title1 "Follow-up Time After Dose 2 (*ESC*){Unicode 2013} Phase 2/3 Subjects 12 Through 15 Years of Age (*ESC*)
{unicode 2013} Safety Population";
footnote1 "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.";
footnote2
      "b.(*ESC*){nbspace 5}n = Number of subjects with the specified characteristic.";

data _final;
      set _base1 _base2 _base3 _base4;
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;
    *-------------------------------------------------------------------;
    * 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;

proc sql noprint;
    create table hspon as select distinct _trt, _column , _vhlabel as _rwlabel ,
        _datasrt, _blcksrt, (min(_rowsrt)-0.9) as _rowsrt , _dptindt as _indent , 0
        as _dptindt from _final(where=(_vhlabel^=' ')) group by _trt, _column ,
        _datasrt, _blcksrt, _vhlabel;
quit;

data ADSL_FU_D21_PED6;
    length _rvalue $800;
    set _final rspon hspon end=eof;
    _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_FU_D21_PED6;
	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;

data outdata1;
	set ADSL_FU_D21_PED6;

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

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;

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

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

ods html file="&outtable.";

proc report data=report nowd list missing contents="" split="|"
        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;
    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 " " 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;
```