```
%macro ma_count_one_row(
 in_data =
,in_where =

,trtvar = &_default_trtvar.
,DEFINE_TOTAL_GROUPS = &_default_define_total_groups.
,TRTVAR_PRELOADFMT = &_default_trtvar_preloadfmt.

,usubjid = &_default_usubjid.

,subgroup =


,byvars =
,byvars_preloadfmt =
,byvars_sort_order =
,byvars_sort_asc_or_desc =

,bign_in_data = BIGN

,text_for_column1 =
,indent_column1 =
,display_zero_row = Yes

,_section_ = 1
,_order1_ = 1

,header =
,header_indent = 0

,get_subj_count = Yes
,get_event_count = No

,count_subj_var = N
,count_event_var = N_EVENT

,out_data = ONE_ROW_COUNT

,debug =
,help =
) / store source des='V1.0.0.9';

%*************************************************************************
FILENAME:    MA_COUNT_ONE_ROW.SAS
DEVELOPER:   (b) (6)
PLATFORM:    SAS 9.1.3, 9.2 on PC
MACROS USED: mu_wordscan.sas, mu_get_maximum_length.sas
ASSUMPTIONS: Macro MA_BIGN must be called before calling this macro
DESCRIPTION:
```

This macro:

1. outputs a dataset as specified in parameter &OUT_DATA (default is ONE_ROW_COUNT)
containing counts categorized according to SUBGROUPS and TREATMENTS. Macro should
be used when goal is to create one row with counts per subgroup in final output.
Examples of when macro would be called:

a) For Disposition table, count number of subjects with SAFETY = 1, per treatment group
and subgroup.
b) For AE table, count number of subject with SAE = 1, per treatment group and subgroup.
c) For AE table, count number of SAE events per treatment group and subgroup.

2. Final dataset will have following variables:

&SUBGROUP: will exist only if parameter SUBGROUP is populated.

By variables: Variables specified in parameter BYVARS.

BYVARX_ORDER: order number within each BYVAR variable

BIGN_DUMMY_FLAG: actually created from macro MA_BIGN, brought in for reporting purposes.

COLUMN1: Row header.

&COUNT_SUBJ_VAR (Default = N): subject level numeric counts.

&COUNT_EVENT_VAR (Default = N_EVENT): event level numeric counts.

SECTION: section of table, used for ordering and creating blank lines between
different sections of a table.

ORDER1: order number within a section.

SKIPVAR: same as _SECTION_.

INDENT: number of spaces to indent COLUMN1 in final output.

HEADER_FLAG: created only if parameter header is populated. Indicates line is a
header.

```
USAGE NOTES:

IN_DATA = input data used to calculate counts.  In the form LIB.DATA, where lib may
be ommitted if WORK
        (default = blank) REQUIRED)
IN_WHERE = subsetting clause
        (default = blank) (OPTIONAL)

TRTVAR = the treatment group variable. Used in the CLASS statement of the PROC
MEANS.
        (default = &_default_trtvar) (REQUIRED)

DEFINE_TOTAL_GROUPS = used to define (sub)total group(s), this parameter must be
defined as follows:
     list of treatment variable values = subtotal 1 ! list of treatment variable
values = subtotal 2 ! ...
     (default = &_default_define_total_groups) (OPTIONAL)


        For example, the values of TRTVAR are 'A' 'B' and 'C' and the analysis
calls for a subtotal of 'A' and
        'B' and an overall total of all three treatments.  Set the parameter as
        define_total_groups = 'A' 'B' = 'subtotal' ! 'A' 'B' 'C' = 'total'
        The (sub)total value(s) must be of the same type as the exisiting variable
        The (sub)totals must be separated by an exclamation point (!)
        The (sub)total values are then used in the TRTVAR_PRELOADFMT (see below)


TRTVAR_PRELOADFMT = the format which assigns the order of the treatments across the
page.  If no
        format is listed, SAS will generate a warn-ing that PRELAODFMT will have no
effect,
        ie, treatment groups which are not already in the data will not be
created/dummied.
        Also, note that any (sub)total groups should be defined here.
        (default = &_default_trtvar_preloadfmt) (OPTIONAL - *but highly
recommended)

        For example, as above, the values of TRTVAR are 'A' 'B' and 'C' and the
analysis calls
        for the columns to be displayed as
                'A', 'B', 'Subtotal of A&B', 'C', 'Total'
        set the format as
          value $_trt <<<--fmtname of user choice
            'A' = '1'
            'B' = '2'
            'subtotal' = '3'
            'C' = '4'
            'total' = '5'
```

```
            NOTE - 'subtotal' and 'total' were created with DEFINE_TOTAL_GROUPS.
User can pick
            whatever makes sense.

USUBJID = the variable(s) used to uniquely identify a subject.
(default=&_default_usubjid) (REQUIRED)

SUBGROUP = subgroups variable(s) to use in the BY statement of the PROC MEANS.
  (default = blank) (OPTIONAL)

BIGN_IN_DATA =  name of "BIGN" dataset.  This dataset will be used to determine
which subgroups will
        be displayed in final output. (default = BIGN) (REQUIRED)

        For example:

        1) You are trying to create a SAE table of the Safety population and
subgrouping by COUNTRY.

        2) There are subjects from Nepal in the Safety population, but none of them
have any SAEs,
           and it is required that there be a page for Nepal in the final output
that states,
           something to the effect "No subjects met criteria" or displays zeros for
the
           "Number of Subjects with SAEs" row.



BYVARS = by variable to obtain counts (typically LBTESTCD VISIT)- space delimited.
    (default = Blank) (OPTIONAL)

BYVARS_PRELOADFMT =  the formats associated with BYVARS to preload in PROC MEANS -
space delimited.
        If do not want a format with a respective BYVARS then use the indicator
word BLANK.  For example,
        if BYVARS = LBTESTCD VISIT and there is no preload format associated with
BYVARS but there is a preload
        format associaged with VISIT, called VISFMT, then call would be:

                        ,BYVARS = LBTESTCD VISIT
                        ,BYVARS_PRELOADFMT = BLANK VISFMT

    If format is BLANK, the BYVAR will be used in the BY statement rather than a
CLASS statement in PROC MEANS.
        This format serves two main functions:

    1) it preloads, meaning that zero counts can be generated even if there are no
subjects that
    meet the criterion for a particular row.
```

```
    2) The format is used for display purposes.

        Multilabel formats are supported but user should ascertain results are as
expected.

    (default = blank) (OPTIONAL)

BYVARS_SORT_ORDER = Value should be I, F, N.
    I = sort by internal value of BY variable.
    F = sort by formatted value of BY variable.
        N = NOTSORTED option used when creating format specified in
BYVARS_PRELOADFMT - BY variable will be sorted in same order
                as when format was created.

    If parameter is F, but PRELOADFMT is BLANK, then defaults to I - an alert will
be
    issued.

        If parameter is N, but PRELOADFMT was created without using the NOTSORTED
option, then an alert
        will be issued and parameter will default to I.

    (default = I) (OPTIONAL)

BYVARS_SORT_ASC_OR_DESC = (D)escending or (A)scending, specify in which way the
    BY variable should be sorted. Default is ascending.
        (default = A(scending)) (OPTIONAL)

TEXT_FOR_COLUMN1 = text for first column associated with counts
        (default = blank) (REQUIRED)

DISPLAY_ZERO_ROW = set to N(o) to remove rows that have zero counts for all
treatment groups.
        (default = Yes) (OPTIONAL)

INDENT_COLUMN1 = number of spaces to indent first column text.
        If blank:
                - default = 0 if parameter &HEADER is not populated.
                - default = &HEADER_INDENT + 2 if parameter &HEADER is populated.

        (default = Blank) (OPTIONAL, but highly recommended)

_SECTION_ = section number of row - used to create blank lines between sections of
        final table. Should be a positive integer.
        (default = 1) (REQUIRED)

_ORDER1_ = order number within section.  Should be a positive integer.
                (default = 1) (REQUIRED)
```

```
HEADER = text of header to be displayed above count row.
        (default = blank) (OPTIONAL)

HEADER_INDENT = number of spaces to indent header.
        (default = 0) (REQUIRED)

GET_SUBJ_COUNT =  set to Y(es) if count of number of subject is desired.
                                    If set to Y(es) then variable &COUNT_SUBJ_VAR
will be in the final dataset.
        (default = Yes) (OPTIONAL)

GET_EVENT_COUNT =  set to Y(es) if count of number of events (as opposed to count
of number of
                                     subjects with event) is desired. If set to Y(es)
then variable
                                     &COUNT_EVENT_VAR will be in the final dataset.
        (default = No) (OPTIONAL)

COUNT_SUBJ_VAR = name of subject level count variable
        (default = N)

COUNT_EVENT_VAR = name of event level count variable
        (default = N_EVENT)

OUT_DATA = the output data set created.
        (default = ONE_ROW_COUNT) (REQUIRED)

HELP = set to Y(es) to output helpful hints to the log
        (Default = &_default_help)

DEBUG = set to Y(es) to save all intermediate datasets and to turn on mprint. Set
to NO to delete
        all intermediate datasets and to not turn on mprint (maintains original
option setting)
        (Default = &_default_debug)

*********************************************************************************;

%PUT ---------------------------------------------- ;
%PUT INFO: (&SYSMACRONAME);
%PUT INFO: Version 1.0;
%PUT START;
%PUT ---------------------------------------------- ;

%**** STEP 1 - PARAMETER CHECKS;

/*%global MA_COUNT_ONE_ROW_RC; */
/*%let MA_COUNT_ONE_ROW_RC = 0;*/

%global &sysmacroname._RC;
```

```
%let &sysmacroname._RC = 0;

%let abort = no;
%mu_help_debug;


%* get a list of the datasets in the WORK library before starting to enable cleanup
at
   the end of this macro;

%md_workinfo(
      debug        = &debug
    ,_workdata     = WORK_DATASETS_DATA
    ,_workview     = WORK_DATASETS_VIEW
    ,_mprinttoggle = mprint_setting
   );

%if "&bign_in_data" eq "" %then %do;
        %let bign_in_data = BIGN;
%end;

%mu_check_data_and_var_exist(
data_to_check = &bign_in_data
,vars_to_check_in_all_data =
,vars_to_check_in_respective_data = &subgroup &trtvar
,abort_if_does_not_exist = yes
,help = no
);



%* check if required parameters USUBJID TRTVAR TEXT_FOR_COLUMN1 and IN_DATA are
blank;
%mu_check_req_parameters(
 parameters_to_check = usubjid trtvar text_for_column1 in_data
,help = no
);

%* check for invalid IN_DATA or invalid variables in IN_DATA;
%mu_check_data_and_var_exist(
data_to_check = &in_data
,vars_to_check_in_all_data =
,vars_to_check_in_respective_data = &usubjid &subgroup &trtvar &byvars
,abort_if_does_not_exist = yes
,help = no
);
```

```
 %* check PARAMETERS related to byvars ;


%md_byvar_check(in_data=&in_data
               ,byvars=&byvars
               ,byvars_preloadfmt=&byvars_preloadfmt
               ,byvars_sort_order=&byvars_sort_order
               ,byvars_sort_asc_or_desc=&byvars_sort_asc_or_desc);



%** check for missing TRTVAR_PRELOADFMT;

%if &trtvar_preloadfmt eq %str( ) %then %do;
        %put
-------------------------------------------------------------------------------
-----------;
        %put ALERT_I: &SYSMACRONAME - No format has been specified in the
TRTVAR_PRELOADFMT parameter.;
        %put ALERT_I: &SYSMACRONAME - If %upcase(&TRTVAR) is not already formatted
PRELOADFMT will have no effect.;
        %put
-------------------------------------------------------------------------------
-----------;
%end;
%else %do;
        data _null_;
                call symput("trtvar_preloadfmt", compress("&trtvar_preloadfmt",
'.'));
        run;
%end;


%* populate with defaults if certain parameters are blank;

%if "&header" ne "" %then %do;
        %if "&header_indent" eq "" %then %do;
                %let header_indent = 0;
        %end;
%end;

%if "&indent_column1" eq "" %then %do;
        %if "&header" eq "" %then %do;
                %let indent_column1 = 0;
        %end;
        %else %do;
                %let indent_column1 = &header_indent + 2;
        %end;
%end;
```

```
%if "&_section_" eq "" %then %do;
        %let _section_ = 1;
%end;

%if "&_order1_" eq "" %then %do;
        %let _order1_ = 1;
%end;

%if "&display_zero_row" eq "" %then %do;
        %let display_zero_row = Y;
%end;

%if "&out_data" eq "" %then %do;
        %let out_data = ONE_ROW_COUNT;
%end;

%if "&count_subj_var" eq "" %then %do;
        %let count_var = N;
%end;

%if "&count_event_var" eq "" %then %do;
        %let count_var = N_EVENT;
%end;

%if "&get_subj_count" eq "" %then %do;
        %let get_subj_count = Y;
%end;

%if "&get_event_count" eq "" %then %do;
        %let get_event_count = N;
%end;



%global _section_int _order1_int;
%macro check_integer(param_name =, value=) ;
 %let RE1=%sysfunc(prxparse(#\D#)) ;
 %put &RE1. ;
  %if %sysfunc(prxmatch(&re1.,&value))>0 %then %do ;
          %put ALERT_P: A NON-DIGIT character was found for parameter = &param_name
Value found was &value..;
          %put ALERT_P: &param_name must be a positive integer.;
          %let &param_name.int = 1;
  %end ;
  %else %do ;
    %put INTEGER value for &param_name found! ;
        %let &param_name.int = 0;
  %end ;
%mend ;
```

```
%check_integer(param_name=_SECTION_, value=&_section_);
%check_integer(param_name=_ORDER1_, value=&_order1_);

%if &_section_int ne 0 or &_order1_int ne 0 %then %do;
        %let &sysmacroname._RC = 1;
        %put ALERT_P: A NON-DIGIT character was found for at least one of the
following two parameters:;
        %put ALERT_P:  _SECTION_ _ORDER1_. Macro will abort.;
%end;

%if &&&sysmacroname._RC = 0 %then %do;
        %if %upcase(%substr(&get_subj_count,1,1)) = Y and
                %upcase(%substr(&get_event_count,1,1)) = Y and
                %upcase(&count_subj_var) = %upcase(&count_event_var) %then %do;

                        %let &sysmacroname._RC = 2;
                        %put
------------------------------------------------------------;
                        %put ALERT_P: &SYSMACRONAME - Both parameters
COUNT_SUBJ_VAR and COUNT_EVENT_VAR;
                        %put ALERT_P: &SYSMACRONAME - HAVE THE SAME VALUE of
&COUNT_SUBJ_VAR.;
                        %put ALERT_P: &SYSMACRONAME - These two parameters must
have unique values.;
                        %put ALERT_P: &SYSMACRONAME - Program will ABORT.;
                        %put
------------------------------------------------------------;
        %end;
%end;

%if &&&sysmacroname._RC = 0 %then %do;
        %if %upcase(%substr(&get_subj_count,1,1)) = N and
                %upcase(%substr(&get_event_count,1,1)) = N %then %do;

                        %let &sysmacroname._RC = 3;
                        %put
------------------------------------------------------------;
                        %put ALERT_P: &SYSMACRONAME - Both parameters
GET_SUBJ_COUNT and GET_EVENT_COUNT are No.;
                        %put ALERT_P: &SYSMACRONAME - At least one must be Yes.;
                        %put ALERT_P: &SYSMACRONAME - Program will ABORT.;
                        %put
------------------------------------------------------------;
        %end;
%end;

%if &&&sysmacroname._RC = 0 %then %do;

        %put ------------------------------------;
```

```
        %put ALERT_I: &SYSMACRONAME - Using dataset &IN_DATA for counting;
        %put -------------------------------------;


        %if "&subgroup" ne "" %then %do;
                %mu_wordscan(string=&subgroup,root=subgroupvar, numw=num_subgroup);
        %end;


%*** STEP 2 - GET THE DATA,CREATE THE DUMMY COUNTING VARIABLE, APPLY FORMATS,
CHECK FOR MISSING VALUES ***;

        data dsin;
                set &in_data;
                %if %sysevalf(%superq(in_where)=,boolean)=0 %then %do; where
&in_where; %end;
                retain _counter 1;
                %if "&trtvar_preloadfmt" ne "" %then %do;
                        format &trtvar &TRTVAR_PRELOADFMT..;
                %end;

                                %if "&byvars" ne "" and "&byvars_preloadfmt" ne ""
%then %do;
                                        %do i=1 %to &num_byvars;
                                                %if "&&fmt&i" ne "BLANK" %then %do;
                                                        format &&byvar&i &&fmt&i...
;
                                                %end;
                                        %end;
                                %end;
        run;

        data dsin;
                set dsin;
                %if "&subgroup" ne "" %then %do;
                        %do i = 1 %to &num_subgroup;
                                if missing(&&subgroupvar&i) then put "ALERT_I:
SUBGROUP variable %upcase(&&subgroupvar&i) is missing at record " _n_ ;
                        %end;
                %end;

                if missing(&trtvar) then put "ALERT_I: TRTVAR variable
%upcase(&trtvar) is missing at record " _n_  "Record will be removed.";
                if missing(&trtvar) then delete;
        run;


        %mu_nobs(dsin);
```

```
        %if &dsin_nobs eq 0 %then %do;

                %* get format associated with subgroup variable;

                %if "&subgroup" ne "" %then %do;
                        %mu_var_attributes(datasets=&in_data, variables=&subgroup);

                                                %put num_subgroup = &num_subgroup;

                        data _null_;
                        call symput("muva_in_data", tranwrd("&in_data", ".", "_"));
                        run;

                        %do i = 1 %to &num_subgroup;
                                %put Macro variable
&muva_in_data._&&subgroupvar&i.._FMT resolves to
&&&&&muva_in_data._&&subgroupvar&i.._FMT  ;
                        %end;
                %end;


                %* if no data after subset, create dummy data for counting and
populate with zeros;

                %put ALERT_I: Dataset &in_data contains 0 observations after
subsetting.;

                proc sort data = &bign_in_data(keep = &subgroup &trtvar
bign_dummy_flag) out = dummy;
                        by &subgroup &trtvar;
                run;

                data count;
                        set dummy;
                        _order1_ = &_order1_;
                        COL1 = "&text_for_column1";
                        _indent_ = &indent_column1;
                        %if %upcase(%substr(&get_subj_count,1,1)) = Y %then %do;
                                &count_subj_var = 0;
                        %end;
                        %if %upcase(%substr(&get_event_count,1,1)) = Y %then %do;
                                &count_event_var = 0;
                        %end;
                        %if "&subgroup" ne ""  %then %do;
                                format %do i = 1 %to &num_subgroup;

    &&subgroupvar&i. &&&&&muva_in_data._&&subgroupvar&i.._FMT

%end;
                                                                        ;
```

```
                        %end;
                run;


                                %* determine possible values of the byvar variable for
situations with 0 records.;
                                %if "&byvars" ne "" and "&byvars_preloadfmt" ne ""
%then %do;
                                                %do i=1 %to &num_byvars;

                                                %* get type (num or char) of byvars;
                                                proc contents data=&in_data
out=contents(keep=name type) noprint;
                                                run;
                                                data contents;
                                                        set contents;
                                                        where
upcase(name)=upcase("&&byvar&i");
                                                        call
symputx('vartype',compress(put(type,best.)));
                                                        if type = 1 then call
symputx('vartypfmt', 'N');
                                                        else call symputx('vartypfmt',
'C');

                                                run;

                                                        %if "&&fmt&i" ne "BLANK" %then %do;
                                                                proc format lib=work
                                                                    cntlout=byvar_format
                                                                    (where=(type =
"&vartypfmt" and fmtname=upcase(compress("&&fmt&i",'$'))));
                                                                run;


                                                                data byvar_format;
                                                                    set byvar_format;
                                                                    %if &vartype = 1 %then
%do;
                                                                                &&byvar&i =
input(compress(start), best8.);
                                                                    %end;
                                                                    %else %do;
                                                                                &&byvar&i =
start;
                                                                    %end;


for_merge=1;
                                                                                keep
```

```
&&byvar&i for_merge;
                                                        run;

                                                        data count;
                                                                set count;
                                                                for_merge=1;
                                                        run;

                                                        proc sql noprint;
                                                                create table count_
as
                                                                select a.*,
b.&&byvar&i
                                                                from count a,
byvar_format b
                                                                where a.for_merge =
b.for_merge;
                                                        quit;

                                                        data count(drop=for_merge);
                                                                set count_;
                                                        run;


                                                %* end of non-missing preloadfmt;
                                                %end;

                                                %else %do;

                                                                proc sort
data=&in_data(keep=&&byvar&i) out=orig_data nodupkey;
                                                                by &&byvar&i;
                                                                where not
missing(&&byvar&i);
                                                        run;

                                                        %mu_nobs(orig_data);

                                                        %if &orig_data_nobs eq 0
%then %do;
                                                                data count;
                                                                        set count;
                                                                        %if
&vartype = 1 %then %do;

&&byvar&i = 1;
                                                                        %end;
                                                                        %else %do;

&&byvar&i = 'A';
```

```
                                                                %end;
                                                        run;
                                                %end;
                                                %else %do;
                                                        data orig_data;
                                                                set
orig_data;

for_merge=1;
                                                        run;

                                                        data count;
                                                                set count;

for_merge=1;
                                                        run;

                                                        proc sql noprint;
                                                                create
table count_ as
                                                                select a.*,
b.&&byvar&i
                                                                from count
a, orig_data b
                                                                where
a.for_merge = b.for_merge;
                                                        quit;

                                                        data
count(drop=for_merge);
                                                                set count_;
                                                        run;

                                                %end;
                                        %end;
                                %end;
                        %end;
                        %else %if "&byvars" ne "" and "&byvars_preloadfmt" eq
"" %then %do;
                                %do i=1 %to &num_byvars;
                                        proc contents data=&in_data
out=contents(keep=name type) noprint;
                                        run;

                                        data contents;
                                                set contents;
                                                where
upcase(name)=upcase("&&byvar&i");
                                                call
symputx('vartype',compress(put(type,best.)));
```

```
                                                run;

                                                proc sort
data=&in_data(keep=&&byvar&i) out=orig_data nodupkey;
                                                        by &&byvar&i;
                                                        where not
missing(&&byvar&i);
                                                run;

                                                %mu_nobs(orig_data);


                                                %if &orig_data_nobs eq 0 %then %do;
                                                        data count;
                                                                set count;
                                                                %if &vartype = 1
%then %do;
                                                                        &&byvar&i =
1;
                                                                %end;
                                                                %else %do;
                                                                        &&byvar&i =
'A';
                                                                %end;
                                                        run;
                                                %end;
                                                %else %do;
                                                        data orig_data;
                                                                set orig_data;
                                                                for_merge=1;
                                                        run;

                                                        data count;
                                                                set count;
                                                                for_merge=1;
                                                        run;

                                                        proc sql noprint;
                                                                create table count_
as
                                                                select a.*,
b.&&byvar&i
                                                                from count a,
orig_data b
                                                                where a.for_merge =
b.for_merge;
                                                        quit;

                                                        data count(drop=for_merge);
                                                                set count_;
```

```
                                                run;

                                        %end;
                                %end;
                        %end;



                %* if do not want to display rows with all zero counts then remove
all records;

                %if %upcase(%substr(&display_zero_row, 1, 1)) = N %then %do;
                        data count;
                                set count;
                                delete;
                        run;

                        %put ALERT_I: &SYSMACRONAME - Row will not be displayed
since counts for all;
                        %put ALERT_I: &SYSMACRONAME - treatment groups are zero and
parameter;
                        %put ALERT_I: &SYSMACRONAME - DISPLAY_ZERO_ROW is set to
&DISPLAY_ZERO_ROW;
                %end;

                %else %do;
                        %put ALERT_I: &SYSMACRONAME - Count for all treatment
groups will be zero.;
                %end;

        %end;


/*************************************************************************
**/


        %else %do;

%*** STEP 3 - CREATE SUBTOTALS AND TOTAL GROUPS AS REQUESTED ***;

                %* get the type and length of the treatment variable;
                %let dsid_in_data = %sysfunc(open(&in_data));
                %let trtvar_type = %sysfunc(vartype(&dsid_in_data,
%sysfunc(varnum(&dsid_in_data, &trtvar))));
                %let trtvar_length = %sysfunc(varlen(&dsid_in_data,
%sysfunc(varnum(&dsid_in_data, &trtvar))));
```

```
                %if &define_total_groups eq %str( ) or %index(&define_total_groups,
=) eq 0 %then %do;
                        %put ALERT_I:
------------------------------------------------------------------------------
--;
                        %put ALERT_I: DEFINE_TOTAL_GROUPS paramenter is missing or
does not specify a total.;
                        %put ALERT_I: No subtotal and/or total groups will be
calculated.;
                        %let define_total_groups = ;
                        %if &help eq Y %then %do;
                                %put HELP: To define subtotal and/or total groups
enter the definition of the group as:;
                                %put HELP: list of treatments = new group ! list of
treatments = new group.;
                                %put HELP: For example: TRTC is in the data with
values of 'A' 'B' and 'C' and the analysis;
                                %put HELP: calls for a subtotal of 'A' and 'B'
along with an overall total of all three treatments.;
                                %put HELP: set DEFINE_TOTAL_GROUPS = 'A' 'B' =
'subtotal' ! 'A' 'B' 'C' = 'total';
                                %put HELP: The left side of the equals sign must be
in the syntax of the actual data and will be;
                                %put HELP: used in an IN statement, ie, TRTC in
('A' 'B')  (no commas!).;
                                %put HELP: The right side of the equals sign will
be the value of the new group and must be of the same;
                                %put HELP: type as the existing treatment group
variable.;
                                %put HELP: Separate each subtotal/total definition
with the exclamation point (!);
                                %put HELP: Make sure to include these new groups in
the value statement of the format that will be ;
                                %put HELP: assigned to the TRTVAR_PRELOADFMT
parameter.;
                        %end;
                        %put ALERT_I:
------------------------------------------------------------------------------
--;
                %end;
                %else %do;
                        %mu_wordscan(string=&define_total_groups, root=total,
numw=numtotals, delim=!)
                        %do i = 1 %to &numtotals;
                                %mu_wordscan(string=&&total&i, root=side&i._,
numw=numside, delim=%str(=))
                        %end;
                %end;
```

```
                %if &define_total_groups ne %str( ) %then %do;
                %** START SUBTOTAL / TOTAL GROUPS;
                        %if &trtvar_type eq C %then %do;
                                %* may need to redefine the length of the treatment
variable - consider the new treatment group names;
                                data _null_;
                                        call symput('new_trtvar_length',
                                                trim(left(put(max(%do i = 1 %to
&numtotals; length(&&side&i._2), %end; &trtvar_length), 8.))));
                                run;
                        %end;

                        %*cycle through each definition of subtotal/total and
create new observations for each subject in each group;
                        %do i = 1 %to &numtotals;
                                %* keep one record per subject per group of
treatments;
                                proc sort data=dsin out=total&i(drop=&trtvar);
                                        by &usubjid;
                                        where &trtvar in (&&side&i._1);
                                run;

                                %* create the new treatment group;
                                data total&i;
                                        %if &trtvar_type eq C %then %do;
                                                length &trtvar $
&new_trtvar_length;
                                        %end;
                                        set total&i;
                                        retain &trtvar &&side&i._2;
                                run;
                        %end;

                        %*set the new treatment groups together with the original
data;
                        data dsin;
                                %if &trtvar_type eq C %then %do;
                                        length &trtvar $ &new_trtvar_length;
                                %end;
                                set dsin %do i = 1 %to &numtotals; total&i %end; ;
                        run;

                        proc sort data=dsin;
                                by &subgroup &byvars &usubjid &trtvar;
                        run;

                %*END OF SUBTOTAL / TOTAL GROUPS;
                %end;
```

```
                %let close_in_data = %sysfunc(close(&dsid_in_data));




%*** STEP 4 - COUNT;

                proc sort data=dsin out=data_for_count nodupkey;
                        by &subgroup

                                                                %if "&byvars" ne "" %then
%do;
                                                                        %do i=1 %to
&num_byvars;


&&byvar&i

                                                                        %end;
                                                                %end;
                                                        &trtvar &usubjid;
                run;

                proc means data=data_for_count noprint nway completetypes missing;
                        %if "&subgroup.&byvars" ne "" %then %do;
                            by &subgroup
                                                                %if "&byvars" ne "" %then
%do;
                                                                        %do i=1 %to
&num_byvars;
                                                                                %if
"&&fmt&i" eq "BLANK" %then %do;

&&byvar&i
                                                                                %end;
                                                                        %end;
                                                                %end;
                                                                ;
                        %end;

                                                        %if "&byvars" ne "" %then %do;
                                                                %do j=1 %to &num_byvars;
                                                                        %if "&&fmt&j" ne
"BLANK" %then %do;
                                                                            class &&byvar&j /
preloadfmt
                                                                                %if
&&byvar_mlf&j._nobs gt 0 %then %do;
```

```
                                                        mlf
                                                %end;
                                                        ;
                                                %end;
                                                %end;
                                        %end;
                            class &trtvar / preloadfmt mlf;
                        var _counter;
                        output out=count(drop=_type_ _freq_) n=n;
                run;

                %if "&subgroup" ne "" and %upcase(%substr(&get_event_count,1,1)) =
Y %then %do;

                        proc sort data=dsin;
                                by &subgroup
                                %if "&byvars" ne "" %then %do;
                                  %do i=1 %to &num_byvars;
                                   %if "&&fmt&i" eq "BLANK" %then %do;
                                    &&byvar&i
                                   %end;
                                  %end;
                                %end;
                                ;
                        run;

                %end;

                %if %upcase(%substr(&get_event_count,1,1)) = Y %then %do;

                        proc means data=dsin noprint nway completetypes missing;
                                %if "&subgroup.&byvars" ne "" %then %do;
                                    by &subgroup
                                                        %if "&byvars" ne ""
%then %do;
                                                                %do i=1 %to
&num_byvars;
                                                                        %if
"&&fmt&i" eq "BLANK" %then %do;

    &&byvar&i

%end;
                                                                %end;
                                                        %end;
                                                        ;
                                %end;


                                                %if "&byvars" ne "" %then
```

```
%do;
                                                                %do j=1 %to
&num_byvars;
                                                                    %if
"&&fmt&j" ne "BLANK" %then %do;
                                                                        class &&byvar&j
/ preloadfmt
                                                                            %if
&&byvar_mlf&j._nobs gt 0 %then %do;
                                                                    mlf
                                                                %end;
                                                                            ;
                                                                    %end;
                                                                %end;
                                                        %end;

                                class &trtvar / preloadfmt mlf;
                                var _counter;
                                output out=event_count(drop=_type_ _freq_)
n=n_event;
                            run;
                                                    %if "&byvars" ne "" %then %do;
                                                            proc sort data=count;
                                                                by &subgroup
                                                                    %do j=1 %to
&num_byvars;

&&byvar&j
                                                                    %end;
                                                                &trtvar;
                                                        run;

                                                        proc sort data=event_count;
                                                                by &subgroup
                                                                    %do j=1 %to
&num_byvars;

&&byvar&j
                                                                    %end;
                                                                &trtvar;
                                                        run;

                                data count;
                                        merge count event_count;
                                        by &subgroup
                                        %if "&byvars" ne "" %then %do;
                                          %do j=1 %to &num_byvars;
                                                &&byvar&j
                                          %end;
                                        %end;
```

```
                                        &trtvar;
                                run;
                                        %end;

                                        %else %do;
                                data count;
                                        merge count event_count;
                                        by &subgroup &trtvar;
                                run;
                                        %end;

        %end;

        %if "&subgroup" ne "" %then %do;

                %* get all subgroups desired to be displayed from BIGN
dataset;

                proc sort data = &bign_in_data(keep = &subgroup &trtvar
bign_dummy_flag) out = dummy;
                        by &subgroup &trtvar;
                run;

                                        proc sort data=count;
                                                by &subgroup &trtvar;
                                        run;

                data count;
                        merge count dummy;
                        by &subgroup &trtvar;
                        if n = . then n = 0;
                        %if %upcase(%substr(&get_event_count,1,1)) = Y
%then %do;
                                if n_event = . then n_event = 0;
                        %end;
                run;

                %if %upcase(%substr(&display_zero_row, 1, 1)) = N %then
%do;
                        proc sort data=count out=non_zero(keep=&subgroup)
nodupkey;
                                by &subgroup;
                                where n ne 0;
                        run;

                        data count;
                                merge count non_zero(in=b);
                                by &subgroup;
                                if b;
                        run;
```

```
                %end;

        %end;

        data count;
                set count;
                COL1 = "&text_for_column1";
                _order1_ = &_order1_;
                _indent_ = &indent_column1;
        run;

%end;


%* order byvars;
%md_add_byvar_order(in_data = count
              ,out_data=count
              ,subgroup=&subgroup
              ,byvars=&byvars
              ,byvars_preloadfmt=&byvars_preloadfmt
              ,byvars_sort_order=&byvars_sort_order
              ,byvars_sort_asc_or_desc=&byvars_sort_asc_or_desc
              );


%*** STEP 5 - CREATE HEADERS;

%if "&header" ne "" %then %do;

        %* get maximum length of column1 from dataset COUNT, length of
header;

        %mu_get_maximum_length(
                data_to_check_length = count
                ,var_to_check_length = COL1
                ,text_to_check_length = %str(&header)
        );


        %* add header row;

        proc sort data=count out=header(drop=COL1) nodupkey;

                by &subgroup &trtvar;
        run;

        data header ;
                set header;
                COL1 = "&header";
                _order1_ = &_order1_ - 0.5 ;
```

```
                n = . ;
                %if %upcase(%substr(&get_event_count,1,1)) = Y %then %do;
                        n_event = . ;
                %end;
                header_flag=1;
                _indent_ = &header_indent;
        run;

        data count;
                length COL1 $ &max_length;
                set header count;
        run;

%end;

data count;
        set count;
        _section_ = &_section_;
        _skipvar_ = &_section_;
run;


%if &dsin_nobs ne 0 %then %do;
        %if %upcase(%substr(&get_subj_count,1,1)) = N %then %do;
                data count;
                        set count(drop=n);
                run;
        %end;

        proc sort data=count out=&out_data(rename=(
                %if %upcase(%substr(&get_subj_count,1,1)) = Y %then %do;
                        n=&count_subj_var
                %end;
                %if %upcase(%substr(&get_event_count,1,1)) = Y %then %do;
                        n_event=&count_event_var
                %end;
                ));
                by &subgroup _section_ _order1_;
        run;
%end;

%else %do;
        proc sort data=count out=&out_data;
                by &subgroup _section_ _skipvar_ _order1_;
        run;
%end;

        proc contents data=&out_data out=contents(keep=name) noprint;
        run;
```

```
                data contents;
                        set contents;
                        where index(upcase(name),'TEMPVAR') gt 0;
                run;

                proc sql noprint;
                        create table _null_ as
                        select * from contents;
                quit;

                %if &sqlobs > 0 %then %do;
                        data &out_data;
                                set &out_data(drop=tempvar:);
                        run;
                %end;



%* STEP 6 - END OF PROCESS TASKS;
         %md_clean_and_reset(
              debug      = &debug
            ,_workdata   = %str(&WORK_DATASETS_DATA &out_data)
            ,_workview   = %str(&WORK_DATASETS_VIEW)
            ,resetmprint = &mprint_setting
                );
%end;

%PUT ------------------------------------------------ ;
%PUT INFO: &sysmacroname._RC = &&&sysmacroname._RC;
%PUT ------------------------------------------------ ;

%if &&&sysmacroname._RC gt 0 %then %do;
        data _null_;
                abort;
        run;
%end;

%PUT ------------------------------------------------ ;
%PUT INFO: (&SYSMACRONAME) ;
%PUT INFO: Version 1.0 ;
%PUT -END------------------------------------------- ;

%exit:
%if &abort = yes %then %do;
    data _null_;
        abort;
    run;
%end;
```

```
%mend ma_count_one_row;
```