```
%macro ma_summ_stats_display
    ( in_data                    =
    , in_where                   = /*Not used*/
    , rules_data                 =
    , rules_by                   =
    , precision_data             =
    , precision_by               =
    , precision_var              = precision
    , precision_val              =
    , bign_data                  =
    , trtvar                     = &_default_trtvar
    , trtvar_preloadfmt          =
    , subgroup                   =
    , stat_label_fmt             =
    , na_text                    = N/A
    , separator                  = %str(,)
    , display                    =
    , trigger_parentheses        = &_default_trigger_parentheses
    , trigger_space_before_bign = Y
    , trigger_align_bign_cnt     = Y
    , trigger_separate_bigncol   = N
    , repeat_if                  =
    , repeat_by                  =
    , repeat_for                 =
    , out_data                   = SUMM_DISPLAY
    , out_data_where             =
    , supress_zero_row           = Y
    , debug                      =
    , help                       =
    ) /store source des='V1.0.0.26'  ;

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

This macro:

  1. outputs a dataset as specified in parameter &OUT_DATA (default is
SUMM_DISPLAY)
     containing summary statistics specified in &STAT_LABEL_FMT, statistics order
variable STAT_ROW specified in &STAT_LABEL_FMT with NOTSTORTED
     option,  grouped by SUBGROUPS, BY_VARS

  2. Final dataset will have following variables:
```

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

SUMMARY STATS VARS: Based on &IN_DATA, &OUTDATA will contain variables STAT_VALUE_x_y

EXAMPLE 1:

when input dataset contains the following variables and treatment variable trt with value 1 and 2:

VAR1_N
VAR1_MEAN
VAR1_SD
VAR1_MED
VAR1_MIN
VAR1_MAX

The following output variables will be created:

STAT_VALUE_1 STAT_VALUE_2

if there is only one value for trt=1 then the varilabes will be STAT_VALUE_1

EXAMPLE 2:

when input dataset contains the following variables and treatment variable trt with value 1 and 2:

VAR1_N
VAR1_MEAN
VAR1_SD
VAR1_MED
VAR1_MIN
VAR1_MAX

VAR2_N
VAR2_MEAN
VAR2_SD
VAR2_MED
VAR2_MIN
VAR2_MAX

The following output variables will be created:

STAT_VALUE_1_1 STAT_VALUE_2_1 STAT_VALUE_1_2 STAT_VALUE_2_2

EXAMPLE NOTES:

The following stats will be converted as follows for standard

naming conventions on the output variables.

```
STDDEV, STD, & SD       ==> STDDEV
MEDIAN, P50, & MED      ==> MEDIAM
MINIMUM & MIN           ==> MIN
MAXIMUM & MAX           ==> MAX
SKEWNESS & SKEW         ==> SKEWNESS
KURTOSIS & KURT         ==> KURTOSIS
Q1 & P25                ==> P25
Q3 & P75                ==> P75
PROBT & PRT             ==> PROBT
```

BYVAR1_ORDER: order number corresponding to the first variable in BYVAR

BYVAR2_ORDER: order number corresponding to the second variable in BYVAR.

BYVAR3_ORDER: order number corresponding to the third variable in BYVAR

and etc.

USAGE NOTES:

IN_DATA = input data used to calculate the summary statistics. In the form LIB.DATA, where lib may be ommitted if WORK
(default = blank) REQUIRED)

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.

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

PRECISION Details:-
precision_data = Dataset that holds the precision variable.
precision_by = BY variables by which precision variable needs to be merged on STAT
dataset.
precision_var = Variable name that holds the precision value. Default value is
PRECISION.
precision_val = If static value for precision is needed, then it should be passed
in here
                instead of populating other 3 precision parameters.

ONLY FOR HORIZONTAL DISPLAY:-
Below listed 3 parameters control how the BIGN numbers in horizontal display will
be displayed in the output.

trigger_parentheses = Controls whether opening and closing parentheses will be
displayed encompassing N=xxx number.
                      Default value is &_default_trigger_parentheses.
                      Example:- If there are 2 treatments A and B and
trigger_parentheses is set to yes then output will
                                 be display as Trt. A (N=xx) and Trt. B (N=yy).

trigger_space_before_bign = Controls whether BLANK space is put before N= sign to
left align N=xxx from left side.
```

```
Example:- Consider there are 3 treatments Active Dose 1, Active Dose2, Placebo and
also Total is being displayed.
          If trigger_space_before_bign is set to Y then output will displayed as
below.
Active Dose 1 (N = 85)
Active Dose 2 (N = 86)
Placebo       (N = 90)
Total         (N = 261)

trigger_align_bign_cnt = Controls whether BLANK space is put after N= sign to align
BIGN Numbers.
Example:- Consider there are 3 treatments Active Dose 1, Active Dose2, Placebo and
also Total is being displayed.
          If trigger_align_bign_cnt is set to Y then output will displayed as
below.
Active Dose 1 (N =  85)
Active Dose 2 (N =  86)
Placebo       (N =  90)
Total         (N = 261)

trigger_separate_bigncol = controls whether separate column linlabel_bign is
created for BIGN counts or not. Default
                           value is "N".
Example:- Consider there are 2 treatments Active Dose 1, Active Dose2, Placebo and
also Total is being displayed.
          If trigger_separate_bigncol is set to "Y" then output dataset will have 2
separate variables. LINLABEL
          for Treatment Names and LINLABEL_BIGN for BIGN counts.
   LINLABEL          LINLABEL_BIGN
--------------   --------------------
Active Dose 1         (N =  85)
Active Dose 2         (N =  86)
Placebo               (N =  90)
Total                 (N = 261)

OUT_DATA = the output data set created.
    (default = SUMM_STATS) (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 &sysmacroname._RC ;
%let &SYSMACRONAME._RC=0 ;
%let mname = &sysmacroname;

%let abort = no ;
%mu_help_debug ;

%*** adding additional macro variable to retain rc valude if cleared due to calls
to other macros due to update to MU_CHECKRC **;
%let _checkrc=;

%* get current setting of mprint and if set to DEBUG mode turn mprint on ;
%* 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 );



%* check if required parameters TRTVAR, vars, and IN_DATA are blank ;
%mu_check_req_parameters
    ( parameters_to_check   =  trtvar in_data rules_data stat_label_fmt
    , help                  = &help
    , debug                 = &debug
    ) ;

%if &display ne %then %do;
    %let display=%qupcase(%substr(%sysfunc(compress(&display)),1,1));
    %MU_CHECKRC
    (CONDITION = %upcase(&display) ne H and %upcase(&display) ne V
    ,RC        = 1
    ,rcprefix  = ma_summ_stats_display
    ,MSG1      = &SYSMACRONAME: Parameter DISPLAY has value &display that is not a
valid value
    ,MSG2      = &SYSMACRONAME: DISPLAY will default to V
    ,MSG3      = &SYSMACRONAME: Valid values for DISPLAY parameter is H or V
    ,ALERTID   = I
    ,SETABORT  = N
    );
        %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
        %if &help = Y %then %put ffffffffffffffffffffffffff &_checkrc. (1)
ffffffffffffffffffffffffff; ;
%end;
%else %do;
```

```
    %let &SYSMACRONAME._RC=1 ;
    %put %str(ALERT_)I: &SYSMACRONAME: Parameter DISPLAY has value &display that is
not a valid value;
    %put %str(ALERT_)I: &SYSMACRONAME: DISPLAY will default to V;
    %put %str(ALERT_)I: &SYSMACRONAME: Valid values for DISPLAY parameter is H or
V;
        %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
%end;
%if %upcase(&display) ne H and %upcase(&display) ne V %then %do;
    %let display=V;
%end;

%*Get Sort order from the input dataset;
%let byvars=;
%mu_check_data_and_var_exist
( data_to_check                     = &in_data
, vars_to_check_in_all_data         =
, vars_to_check_in_respective_data  =
, abort_if_does_not_exist           = yes
, help                              = &help
) ;
%mu_get_sort_order(&in_data);
%if %qupcase(&sort_order)=_ALL_ %then %do;
    %put %str(ALERT_)P: &SYSMACRONAME - The input dataset %qupcase(&in_data) is not
sorted.;
    %put %str(ALERT_)P: &SYSMACRONAME - Macro will abort.;
    %let &SYSMACRONAME._RC=7;
        %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
    %let abort = yes ;
    %goto exit ;
%end;
%else %do;
    data _null_;
    length sort $1000;
    sort=tranwrd(upcase(strip("&sort_order")),
                 upcase(strip("&trtvar")),
                 '');
    %if &subgroup ne %then %do;
        sort=tranwrd(strip(sort),
                     upcase(strip("&subgroup")),
                     '');
    %end;
    call symputx("byvars",sort);
    run;
%end;

%* Check to see if the precision details are supplied properly or not;
%if &precision_var eq %then %do;
    %let precision_var=precision;
%end;
```

```
%if (&precision_data ne or &precision_by ne ) and &precision_val eq %then %do;

    %if %index(&byvars,%qupcase(&precision_var))>0 %then %do;
        %if %qupcase(&help)=Y %then %do;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to get the proper
precision for each statistic, macro needs to get precision variable into the input
stats dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to do this USER does not
need to include %qupcase(&PRECISION_VAR) variable into the sort order of input
dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME - By adding precision variable into
sort order of input dataset, USER might end up with unexpected results.;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to get the precision
variable in the input dataset properly USER needs to follow below listed steps.;
            %put %str(ALERT_)I: &SYSMACRONAME - There are multiple scenarios.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 1 - Constant precision
value. e.g. Demographic Table (Height, Weight, BMI);
            %put %str(ALERT_)I: &SYSMACRONAME -     For this particular case, USER
needs to populate only one parameter PRECISION_VAL with the precision that USER
wants.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Other 3 precision parameters,
PRECISION_DATA PRECISION_BY and PRECISION_VAR needs to be left blank.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 2 - Precision variable is
supplied in the original Analysis Dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME -     For this scenario, USER first
needs to populate PRECISION_DATA parameter with the original analysis dataset
name.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Second, USER needs to populate
PRECISION_BY parameter with the by variables that yield unique precision value.;
            %put %str(ALERT_)I: &SYSMACRONAME -             These by variable must
be present in the stats dataset as well.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Third, USER needs to populate
PRECISION_VAR parameter with name of the precision variable.;
            %put %str(ALERT_)I: &SYSMACRONAME -     PRECISION_VAL parameter needs
to be left blank.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 3 - Precision needs to be
derived based on some variable value.;
            %put %str(ALERT_)I: &SYSMACRONAME -     For this scenario, First, USER
needs to first run the macro MA_SUMM_STATS_PRECISION.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Second, USEER needs to populate
PRECISION_DATA, PRECISION_BY and PRECISION_VAR parameters as explained in Scenario
2.;
            %put %str(ALERT_)I: &SYSMACRONAME -     PRECISION_VAL parameter needs
to be left blank.;
            %put %str(ALERT_)P: &SYSMACRONAME - PRECISION_VAR
%qupcase(&precision_var) is also listed as;
            %put %str(ALERT_)P: &SYSMACRONAME - one of the BYVARS.;
            %put %str(ALERT_)P: &SYSMACRONAME - Macro will abort.;
        %end;
        %else %do;
```

```
            %put %str(ALERT_)P: &SYSMACRONAME - PRECISION_VAR
%qupcase(&precision_var) is also part of;
            %put %str(ALERT_)P: &SYSMACRONAME - the sort order of input dataset.;
            %put %str(ALERT_)P: &SYSMACRONAME - Macro will abort.;
        %end;
        %let &SYSMACRONAME._RC=6;
                %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
        %let abort = yes ;
        %goto exit ;
    %end;

    %mu_check_req_parameters
    ( parameters_to_check   = precision_data precision_by
    , help                  = &help
    , debug                 = &debug
    ) ;

    %if &precision_var eq %then %do;
        %let precision_var=precision;
    %end;
    %mu_check_data_and_var_exist
    ( data_to_check                     = &in_data &precision_data
    , vars_to_check_in_all_data         = &precision_by
    , vars_to_check_in_respective_data  =  ! &precision_var
    , abort_if_does_not_exist           = yes
    , help                              = no
    ) ;

    proc sort data=&precision_data(keep=&precision_by &precision_var) out=precision
nodupkey;
        by &precision_by;
    run;
    %mu_get_sort_order(&in_data);
    proc sort data=&in_data;
        by &precision_by;
    run;
    data &in_data;
        merge &in_data(in=a) precision
                                %if %qupcase(&precision_var) ne PRECISION %then
%do;
                                    (rename=(&precision_var = precision))
                                %end; ;
        by &precision_by;
    if a;
    run;
    proc sort data=&in_data;
        by &sort_order;
    run;

%end;
```

```
%else %do;

    %if %index(&byvars,&precision_var)>0 %then %do;
        %if %qupcase(&help)=Y %then %do;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to get the proper
precision for each statistic, macro needs to get precision variable into the input
stats dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to do this USER does not
need to include %qupcase(&PRECISION_VAR) variable into the sort order of input
dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME - By adding precision variable into
sort order of input dataset, USER might end up with unexpected results.;
            %put %str(ALERT_)I: &SYSMACRONAME - In order to get the precision
variable in the input dataset properly USER needs to follow below listed steps.;
            %put %str(ALERT_)I: &SYSMACRONAME - There are multiple scenarios.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 1 - Constant precision
value. e.g. Demographic Table (Height, Weight, BMI);
            %put %str(ALERT_)I: &SYSMACRONAME -     For this particular case, USER
needs to populate only one parameter PRECISION_VAL with the precision that USER
wants.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Other 3 precision parameters,
PRECISION_DATA PRECISION_BY and PRECISION_VAR needs to be left blank.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 2 - Precision variable is
supplied in the original Analysis Dataset.;
            %put %str(ALERT_)I: &SYSMACRONAME -     For this scenario, USER first
needs to populate PRECISION_DATA parameter with the original analysis dataset
name.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Second, USER needs to populate
PRECISION_BY parameter with the by variables that yield unique precision value.;
            %put %str(ALERT_)I: &SYSMACRONAME -             These by variable must
be present in the stats dataset as well.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Third, USER needs to populate
PRECISION_VAR parameter with name of the precision variable.;
            %put %str(ALERT_)I: &SYSMACRONAME -     PRECISION_VAL parameter needs
to be left blank.;
            %put %str(ALERT_)I: &SYSMACRONAME - Scenario 3 - Precision needs to be
derived based on some variable value.;
            %put %str(ALERT_)I: &SYSMACRONAME -     For this scenario, First, USER
needs to first run the macro MA_SUMM_STATS_PRECISION.;
            %put %str(ALERT_)I: &SYSMACRONAME -     Second, USEER needs to populate
PRECISION_DATA, PRECISION_BY and PRECISION_VAR parameters as explained in Scenario
2.;
            %put %str(ALERT_)I: &SYSMACRONAME -     PRECISION_VAL parameter needs
to be left blank.;
            %put %str(ALERT_)P: &SYSMACRONAME - PRECISION_VAR
%qupcase(&precision_var) is also listed as;
            %put %str(ALERT_)P: &SYSMACRONAME - one of the BYVARS.;
            %put %str(ALERT_)P: &SYSMACRONAME - Macro will abort.;
        %end;
        %else %do;
```

```
            %put %str(ALERT_)P: &SYSMACRONAME - PRECISION_VAR
%qupcase(&precision_var) is also part of;
            %put %str(ALERT_)P: &SYSMACRONAME - the sort order of input dataset.;
            %put %str(ALERT_)P: &SYSMACRONAME - Macro will abort.;
        %end;
        %let &SYSMACRONAME._RC=6;
                %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
        %let abort = yes ;
        %goto exit ;
    %end;
    %else %do;
        %if &precision_val ne %then %do;
            data _null_;
                prec_=compress(strip("&precision_val"),'','d');
                call symput ("prec_check",strip(prec_));
            run;
            %if &prec_check eq %then %do;
                %mu_get_sort_order(&in_data);
                data &in_data;
                    set &in_data;
                retain precision &precision_val;
                run;
                proc sort data=&in_data;
                    by &sort_order;
                run;
            %end;
            %else %do;
                %put %str(ALERT_)C: &SYSMACRONAME - Value supplied in PRECISION_VAL
parameter in invalid.;
                %put %str(ALERT_)C: &SYSMACRONAME - Precision will be defaulted to
1 decimal.;
                %let &SYSMACRONAME._RC=8;
                                %if "&&&mname._rc." ne "0" %then %let
_checkrc=&&&mname._rc.;
                %mu_get_sort_order(&in_data);
                data &in_data;
                    set &in_data;
                retain precision 1;
                run;
                proc sort data=&in_data;
                    by &sort_order;
                run;
            %end;
        %end;
        %else %do;
            %put %str(ALERT_)C: &SYSMACRONAME - No value is passed in PRECISION_VAL
parameter.;
            %put %str(ALERT_)C: &SYSMACRONAME - Precision will be defaulted to 1
decimal.;
            %let &SYSMACRONAME._RC=9;
```

```
                        %if "&&&mname._rc." ne "0" %then %let
_checkrc=&&&mname._rc.;
            %mu_get_sort_order(&in_data);
            data &in_data;
                set &in_data;
            retain precision 1;
            run;
            proc sort data=&in_data;
                by &sort_order;
            run;
        %end;
    %end;

%end;
%* Convert PRECISION variable into numberic if it is character;
%let ds_id=%sysfunc(open(&in_data));
%let var_num=%sysfunc(varnum(&ds_id,precision));
%if %sysfunc(vartype(&ds_id,&var_num))=C %then %do;
    %mu_get_sort_order(&in_data);
    data &in_data;
        set &in_data(rename=(precision=_precision));
        by &sort_order;
    precision=input(strip(_precision),best.);
    drop _precision;
    run;
%end;
%let close_ds_id=%sysfunc(close(&ds_id));
%let precision_var=precision;

%* check for invalid IN_DATA or not valid variables in IN_DATA;
%mu_check_data_and_var_exist
( data_to_check                     = &in_data &rules_data
, vars_to_check_in_all_data         =  &rules_by    /**&rules_by should be in both
dataset**/
, vars_to_check_in_respective_data  = /*&subgroup &byvars &trtvar &vars*/ precision
!
, abort_if_does_not_exist           = yes
, help                              = no
) ;

%*Assign SUMM_DISPLAY to out_data parameter if left blank;
%if "&out_data" eq "" %then %do ;
    %let out_data = SUMM_DISPLAY  ;
%end ;

%*find variable information from SUMM_STATS**;
proc contents data = &in_data out = __summ_stats_meta__ noprint;
run;

%**find number of var set in summ_stats**;
```

```
data __summ_stats_meta1__;
  set __summ_stats_meta__;
  x=prxparse("/^(var)(\d*)(_)/i");
  if prxmatch(x,name);
  call prxposn(x,2,pos,len);
  num = input(strip(substr(name,pos,len)),best.);

  length stat $ 20;
  stat = upcase(scan(name,2, '_'));
run;

proc sql noprint;
  select count(name) into: num_of_statvar
  from __summ_stats_meta1__;

  %let num_of_statvar = &num_of_statvar;

  select name into: statvar1 - :statvar&num_of_statvar
  from __summ_stats_meta1__;

  select max(num) into :num_of_var
  from __summ_stats_meta1__;
quit;


%if &num_of_var>=3 %then %do;
     %put ALERT_I: &SYSMACRONAME - There are more than 2 sets of variables  ;
     %put ALERT_I: &SYSMACRONAME - in &IN_DATA. Please confirm ;
%end;

%put &statvar1 - &&statvar&num_of_statvar;
%put &num_of_var;


%*check &RULES_DATA**;
data __rules_data__;
  length stat $ 20;
  set &rules_data;
run;

proc sort data = __rules_data__;
  by &rules_by stat;
run;


*check if RULE exists for each stat**;
proc sort data = __summ_stats_meta1__ nodupkey out = __stat_in_data__(keep=stat);
  by  stat;
run;
```

```
%if "&rules_by" ne "" %then %do;
proc sort data = &in_data out = __by_vars__(keep = &rules_by) nodupkey;
  by &rules_by;
run;
data __stat_in_data__;
    set __by_vars__;
     do i = 1 to xnobs;
        set __stat_in_data__ nobs=xnobs point=i;
        output;
     end;
run;
%end;

proc sort data = __stat_in_data__ ;
  by &rules_by stat;
run;

proc sort data = __rules_data__ out = __rules_data_keep_one__ nodupkey;
  by  &rules_by stat;
run;

data __stat_in_data_check__;
  merge __stat_in_data__ (in=a) __rules_data_keep_one__(in=b);
  by &rules_by stat;
  if a and not b;
run;

%mu_nobs(__stat_in_data_check__);

%let stat_no_rule=;
proc sql noprint;
     select distinct stat into :stat_no_rule separated by ' '
     from __stat_in_data_check__;
quit;

%MU_CHECKRC
(CONDITION = &__stat_in_data_check___nobs > 0
,RC        = 2
,rcprefix  = ma_summ_stats_display
,MSG1      = &SYSMACRONAME: No RULE found for the following stats: &stat_no_rule
,MSG2      = &SYSMACRONAME: Program will ABORT
,MSG3      =
,ALERTID   = P
,SETABORT  = Y
);

%if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
%if &help = Y %then %put ffffffffffffffffffffffffff &_checkrc. (2)
ffffffffffffffffffffffffff; ;
```

```
**check if there is RULE with miss1ng value for STATS found in IN_DATA***;
proc sort data = __rules_data__;
  by &rules_by stat;
run;

data _null_;
  merge __rules_data__(in=a) __stat_in_data__(in=b);
  by &rules_by stat;
  if a and b and missing(rule);
  if missing(rule) then do;
     put "ALERT_P: &SYSMACRONAME - RULE is missing for STAT: " stat;
     put "ALERT_P: &SYSMACRONAME - Program will ABORT." ;
  end;
  call symputx("&SYSMACRONAME._RC","3");
run;

%if &&&SYSMACRONAME._RC=3 %then %do;
   %let abort = yes ;
   %goto exit ;
%end;



%* get stats from formats ***;
proc format lib=work
   cntlout=__stat_format__
(where=(fmtname=upcase(compress(compress("&stat_label_fmt",'.'),'$')))) ;
run ;

%mu_nobs(__stat_format__);

%MU_CHECKRC
(CONDITION = &__stat_format___nobs eq 0
,RC        = 4
,rcprefix  = ma_summ_stats_display
,MSG1      = &SYSMACRONAME: Parameter STAT_LABEL_FMT is specified as
&STAT_LABEL_FMT
,MSG2      = &SYSMACRONAME: however the format &STAT_LABEL_FMT  cannot be
found/loaded
,MSG3      = &SYSMACRONAME: Program will ABORT
,ALERTID   = P
,SETABORT  = Y
);
%if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
%if &help = Y %then %put ffffffffffffffffffffffffff &_checkrc. (4)
ffffffffffffffffffffffffff; ;

 %mu_nobs(__stat_format__,where=hlo='S');
```

```
%MU_CHECKRC
(CONDITION = &__stat_format___nobs eq 0
,RC        = 5
,rcprefix  = ma_summ_stats_display
,MSG1      = &SYSMACRONAME: Format &STAT_LABEL_FMT is not created with option
NOTSORTED
,MSG2      = &SYSMACRONAME: It has to be created with option NOTSORTED to create
STAT_ROW
,MSG3      = &SYSMACRONAME: Program will ABORT
,ALERTID   = P
,SETABORT  = Y
);
%if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
%if &help = Y %then %put ffffffffffffffffffffffffff &_checkrc. (5)
ffffffffffffffffffffffffff; ;


%let MSTE = N;
%let Q1Q3 = N;
%let P1P99 = N;
%let P5P95 = N;
%let P10P90 = N;
%let LCLMUCLM = N;
%let NUM1NUM2 = N;
%let PNUM1NUM2P = N;
%let NUM1PNUM2P = N;
data __stat_format__(keep = stat_row stat stat_label);
  set __stat_format__;
  length stat $ 20;
  stat=upcase(start);

  %* use standard stat name found in rules dataset**;

  if stat in ('STD','SD') then stat='STDDEV';
  else if stat in ('STE','SE') then stat='STDERR';
  else if stat in ('KURT') then stat='KURTOSIS';
  else if stat in ('SKEW') then stat='SKEWNESS';
  else if stat in ('P50','MED') then stat='MEDIAN';
  else if stat in ('P25') then stat='Q1';
  else if stat in ('P75') then stat='Q3';
  else if stat in ('MINIMUM') then stat='MIN';
  else if stat in ('MAXIMUM') then stat='MAX';
  else if stat in ('PRT') then stat='PROBT';

  if stat='MSTE' then call symputx("MSTE",'Y');
  if stat='Q1Q3' then call symputx("Q1Q3",'Y');
  if stat='P1P99' then call symputx("P1P99",'Y');
  if stat='P5P95' then call symputx("P5P95",'Y');
  if stat='P10P90' then call symputx("P10P90",'Y');
  if stat='LCLMUCLM' then call symputx("LCLMUCLM",'Y');
```

```
  if stat='NUM1NUM2' then call symputx("NUM1NUM2",'Y');
  if stat='PNUM1NUM2P' then call symputx("PNUM1NUM2P",'Y');
  if stat='NUM1PNUM2P' then call symputx("NUM1PNUM2P",'Y');
  stat_row=_n_;
  rename  label=stat_label;
run;


proc sort data = __stat_format__;
  by stat;
run;

 data __summ_stats__;
     set &in_data;
 run;


%*if supress_zero_row is Y, then BIGN_DATA must have value with valid dataset**;
%if  &supress_zero_row ne %then %do;

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

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


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

    proc sort data = &bign_data out = __bign__(keep=&subgroup &trtvar denom);
       by &subgroup &trtvar;
    run;

    data __summ_stats__(drop=denom);
       merge __summ_stats__(in=a) __bign__(in=b);
       by &subgroup &trtvar;
       *if a;
       if denom=0 or not b then do;
          %do i=1 %to &num_of_var;
            if var&i._n=0 then var&i._n=.;
          %end;
       end;
    run;
```

```
  %end;
%end;

%if &rules_by ne %then %do;
%mu_transpose(
     in_data            = __rules_data__        /* input data set */
    ,transpose_by       =  &rules_by /* BY statement in the transpose */
    ,transpose_vars     = rule fixed    /* VAR statement in the transpose */
    ,transpose_id       = stat  /* ID statement in the transpose */
    ,out_data           = __rules_transpose__   /* output data set */
    ,help               = &help
    ,debug              = &debug
     );


  proc sort data =   __summ_stats__;
     by &rules_by;
  run;
%end;
%else %do;
data __summ_stats__;
    set __summ_stats__;
retain dummy 'a';
run;
data __rules_data__;
    set __rules_data__;
retain dummy 'a';
run;
%mu_transpose(
     in_data            = __rules_data__        /* input data set */
    ,transpose_by       = dummy /* BY statement in the transpose */
    ,transpose_vars     = rule fixed    /* VAR statement in the transpose */
    ,transpose_id       = stat  /* ID statement in the transpose */
    ,out_data           = __rules_transpose__   /* output data set */
    ,help               = &help
    ,debug              = &debug
     );
%end;

data __summ_stats1__;
  merge __summ_stats__(in=a) __rules_transpose__;
  %if &rules_by ne %then %do;
  by &rules_by;
  %end;
  %else %do;
  by dummy;
  %end;


  %do i=1 %to  &num_of_statvar;
```

```
         length &&statvar&i.._c $50;
         %put &&statvar&i;
         %let leaf =  %scan(%bquote(&&statvar&i),2,%str(_)) ;
         %put leaf = &leaf;

         if  fixed_&leaf  = 1 then do;

            if not missing(&&statvar&i) then do;
               &&statvar&i =  round(&&statvar&i,10**(-rule_&leaf.));
               &&statvar&i.._c = strip(putn(&&statvar&i,catx('.',20, rule_&leaf)));
            end;

         end;
         else if fixed_&leaf  in (0,.) then do;

            if not missing(&&statvar&i) then do;
               &&statvar&i =  round(&&statvar&i,10**(-rule_&leaf.-&precision_var));

               &&statvar&i.._c = strip(putn(&&statvar&i,catx('.', 20 ,
rule_&leaf.+&precision_var)));
            end;

         end;

  %end;

  length %do i=1 %to &num_of_var;
            var&i._minmax_c var&i._mstd_c
            %if &MSTE eq Y %then %do; var&i._MSTE_c %end;
            %if &Q1Q3 eq Y %then %do; var&i._q1q3_c %end;
            %if &P1P99 eq Y %then %do; var&i._p1p99_c %end;
            %if &P5P95 eq Y %then %do; var&i._p5p95_c %end;
            %if &P10P90 eq Y %then %do; var&i._p10p90_c %end;
            %if &LCLMUCLM eq Y %then %do; var&i._lclmuclm_c %end;
            %if &NUM1NUM2 eq Y %then %do; var&i._NUM1NUM2_c %end;
            %if &PNUM1NUM2P eq Y %then %do; var&i._PNUM1NUM2P_c %end;
            %if &NUM1PNUM2P eq Y %then %do; var&i._NUM1PNUM2P_c %end;

         %end; $50;

  %do i=1 %to &num_of_var;
            var&i._minmax_c = catx("&separator ",var&i._min_c, var&i._max_c);
            if not missing(var&i._mean_c) and not missing(var&i._stddev_c) then
var&i._mstd_c = strip(var&i._mean_c)||' ('||strip(var&i._stddev_c)||')';
            else if not missing(var&i._mean_c) and missing(var&i._stddev_c) then
var&i._mstd_c = strip(var&i._mean_c)||" (&na_text)";
            %if &MSTE eq Y %then %do;
                if not missing(var&i._mean_c) and not missing(var&i._stderr_c) then
var&i._mste_c = strip(var&i._mean_c)||' ('||strip(var&i._stderr_c)||')';
                else if not missing(var&i._mean_c) and missing(var&i._stderr_c)
```

```
then var&i._mste_c = strip(var&i._mean_c)||" (&na_text)";
            %end;
            %if &Q1Q3 eq Y %then %do;
                if not missing(var&i._q1_c) and not missing(var&i._q3_c) then
var&i._q1q3_c = catx("&separator ",var&i._q1_c, var&i._q3_c);
                                else if not missing(var&i._mean_c) and
missing(var&i._q1_c) and missing(var&i._q3_c) then var&i._q1q3_c = "&na_text";
            %end;
            %if &P1P99 eq Y %then %do;
                if not missing(var&i._p1_c) and not missing(var&i._p99_c) then
var&i._p1p99_c = catx("&separator ",var&i._p1_c, var&i._p99_c);
                                else if not missing(var&i._mean_c) and
missing(var&i._p1_c) and missing(var&i._p99_c) then var&i._p1p99_c = "&na_text";
            %end;
            %if &P5P95 eq Y %then %do;
                if not missing(var&i._p5_c) and not missing(var&i._p95_c) then
var&i._p5p95_c = catx("&separator ",var&i._p5_c, var&i._p95_c);
                                else if not missing(var&i._mean_c) and
missing(var&i._p5_c) and missing(var&i._p95_c) then var&i._p5p95_c = "&na_text";
            %end;
            %if &P10P90 eq Y %then %do;
                if not missing(var&i._p10_c) and not missing(var&i._p90_c) then
var&i._p10p90_c = catx("&separator ",var&i._p10_c, var&i._p90_c);
                                else if not missing(var&i._mean_c) and
missing(var&i._p10_c) and missing(var&i._p90_c) then var&i._p10p90_c = "&na_text";
            %end;
            %if &LCLMUCLM eq Y %then %do;
                if not missing(var&i._lclm_c) and not missing(var&i._uclm_c) then
var&i._lclmuclm_c = "("||catx("&separator ",var&i._lclm_c, var&i._uclm_c)||")";
                                else if not missing(var&i._mean_c) and
missing(var&i._lclm_c) and missing(var&i._uclm_c) then var&i._lclmuclm_c =
"&na_text";
            %end;
            %if &NUM1NUM2 eq Y %then %do;
                if not missing(var&i._NUM1_c) and not missing(var&i._NUM2_c) then
var&i._NUM1NUM2_c = catx("&separator ",var&i._num1_c, var&i._num2_c);
                                else if not missing(var&i._mean_c) and
missing(var&i._num1_c) and missing(var&i._num2_c) then var&i._num1num2_c =
"&na_text";
            %end;

            %if &PNUM1NUM2P eq Y %then %do;
                if not missing(var&i._NUM1_c) and not missing(var&i._NUM2_c) then
var&i._Pnum1num2P_c = "("||catx("&separator ",var&i._num1_c, var&i._num2_c)||")";
                                else if not missing(var&i._mean_c) and
missing(var&i._num1_c) and missing(var&i._num2_c) then var&i._Pnum1num2P_c =
"&na_text";
            %end;

            %if &NUM1PNUM2P eq Y %then %do;
```

```
                    if not missing(var&i._num1_c) and not missing(var&i._num2_c)
then var&i._num1pnum2p_c = strip(var&i._num1_c)||' ('||strip(var&i._num2_c)||')';
                    else if not missing(var&i._num1_c) and missing(var&i._num2_c)
then var&i._num1pnum2p_c = strip(var&i._num1_c)||" (&na_text)";
                    else if missing(var&i._num1_c) and missing(var&i._num2_c) then
var&i._num1pnum2p_c = "(&na_text)";
            %end;




            if var&i._n=1 then do;
                if var&i._CV_C in ('.', '') then var&i._CV_C="&na_text";
                if var&i._LCLM_C in ('.', '') then var&i._LCLM_C="&na_text";
                if var&i._UCLM_C in ('.', '') then var&i._UCLM_C="&na_text";
                if var&i._MODE_C in ('.', '') then var&i._MODE_C="&na_text";
                if var&i._PROBT_C in ('.', '') then var&i._PROBT_C="&na_text";
                if var&i._T_C in ('.', '') then var&i._T_C="&na_text";
                if var&i._VAR_C in ('.', '') then var&i._VAR_C="&na_text";
                if var&i._STDDEV_C in ('.', '') then var&i._STDDEV_C="&na_text";
                if var&i._STDERR_C in ('.', '') then var&i._STDERR_C="&na_text";
            end;
            if var&i._n lt 3 and var&i._SKEWNESS_C in ('.', '') then
var&i._SKEWNESS_C="&na_text";
            if var&i._n lt 4 and var&i._KURTOSIS_C in ('.', '') then
var&i._KURTOSIS_C="&na_text";


  %end;

run;

%if &rules_by eq %then %do;
proc sql;
    alter table __summ_stats1__
        drop dummy;
quit;
%end;

%if %upcase(&display)=H  %then %do;

 %*** for horizontal display, TRTVAR_PRELOADFMT cannot be blank**;
%MU_CHECKRC
(CONDITION = %length(&TRTVAR_PRELOADFMT)=0
,RC        = 4
,rcprefix  = ma_summ_stats_display
,MSG1      = &SYSMACRONAME - For horizontal display: TRTVAR_PRELOADFMT cannot be
blank
,MSG2      = &SYSMACRONAME - Program will ABORT
,MSG3      =
```

```
,ALERTID   = P
,SETABORT  = Y
);

%if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
%if &help = Y %then %put ffffffffffffffffffffffffff &_checkrc. (4)
ffffffffffffffffffffffffff; ;


data _null_;
    call symput("trtvar_preloadfmt", compress("&trtvar_preloadfmt", '.'));
run;

  %*find variable to be kept except for the stat*;
  proc contents data = __summ_stats1__ out = __summ_out1_meta__ noprint;
  run;

  data __summ_out1_meta__ ;
    set __summ_out1_meta__;
    x=prxparse("/^(var)(\d*)(_)/i");
    if  prxmatch(x,name) and upcase(strip(reverse(name)))=:'C_';
    call prxposn(x,2,pos,len);
    num = input(strip(substr(name,pos,len)),best.);
    length stat $ 20;
    stat = upcase(scan(name,2, '_'));
  run;

  proc sort data = __summ_out1_meta__;
    by stat;
  run;

  proc sort data =  __stat_format__;
    by stat;
  run;

  %** find variables other than stat variable, need to be in keep statment later**;
  proc sql noprint ;
     select name into :__keepvar__ separated by ' '
     from __summ_out1_meta__
     where stat ='';
  quit;

  %** find stat variable and their order**;
  data __summ_out1_meta__;
     merge __summ_out1_meta__(in=a where=(stat ne '')) __stat_format__(in=b);
     by stat;
     if a and b;
  run;

  %*generate rename statement for stat variable**;
```

```
  data _null_;
    set __summ_out1_meta__ end=eof;
    %* If there is only one variable for which the stats are being derived then no
need to have variable number in output var names;
    %if &num_of_var ne 1 %then %do;
        call symput(cats("__rename__",_n_),
catx('=',name,cats("stat_value_",num,"_",stat_row)));
    %end;
    %else %do;
        call symput(cats("__rename__",_n_),
catx('=',name,cats("stat_value_",stat_row)));
    %end;
    if eof then call symputx("__nstatvar__",_n_);
  run;

  data __summ_stats2__;
     set __summ_stats1__;
     rename %do i=1 %to &__nstatvar__;
                &&__rename__&i
            %end;;
  run;

  %** create new column for big N***;
  proc sort data = __summ_stats2__;
     by &subgroup &trtvar;
  run;

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

  proc sort data = &bign_data out = __bign__;
     by &subgroup &trtvar;
  run;

  proc sql noprint;
    select max(length(strip(put(denom,best.)))) into: maxlen_denom
      from __bign__;
    select max(length(strip(put(&trtvar,&trtvar_preloadfmt..)))) into: maxlen_trt
      from __summ_stats2__;
  quit;

  data __summ_stats3__;
     merge __summ_stats2__(in=a) __bign__;
     by &subgroup &trtvar;
     if a;
```

```
    length linlabel linlabel_bign $200 _trtlen _denomlen 8;

    _trtlen=length(strip(put(&trtvar,&trtvar_preloadfmt..)));
    _denomlen=length(strip(put(denom,best.)));

    if denom>.z then linlabel=strip(put(&trtvar,&trtvar_preloadfmt..))||
       %if %qupcase(%substr(&trigger_space_before_bign,1,1))=Y %then %do;
           repeat(' ',&maxlen_trt-_trtlen)||
       %end;
       %else %do;
           ' '||
       %end;
       %if %qupcase(%substr(&trigger_parentheses,1,1))=Y %then %do;
           '('||
       %end;
       'N ='||
       %if %qupcase(%substr(&trigger_align_bign_cnt,1,1))=Y %then %do;
           repeat(' ',&maxlen_denom-_denomlen)||
       %end;
       %else %do;
           ' '||
       %end;
       strip(put(denom,best.))
       %if %qupcase(%substr(&trigger_parentheses,1,1))=Y %then %do;
           ||')';
       %end;;

       %if %qupcase(%substr(&trigger_separate_bigncol,1,1))=Y %then %do;
           %if %qupcase(%substr(&trigger_parentheses,1,1))=Y %then %do;
               linlabel_bign=scan(scan(linlabel,2,"("),1,")");
               linlabel=scan(linlabel,1,"(");
           %end;
           %else %do;
               linlabel_bign='N ='||scan(linlabel,2,"N =");
               linlabel=scan(linlabel,1,"N =");
           %end;
       %end;
       %else %do;
           call missing(linlabel_bign);
       %end;
    keep &subgroup &byvars &trtvar linlabel stat_v:
         %if %qupcase(%substr(&trigger_separate_bigncol,1,1))=Y %then %do;
linlabel_bign %end;
         ;
  run;

  proc sort data=__summ_stats3__(keep=&trtvar) out=__trtvar__ nodupkey;
    by &trtvar;
  run;
  data __trtvar__;
```

```
     set __trtvar__;
  retain stat_row 0;
  stat_row+1;
  run;
  proc sort data=__summ_stats3__;
     by &trtvar;
  run;
  data __summ_stats3__;
    merge __summ_stats3__(in=a) __trtvar__;
    by &trtvar;
  if a;
  run;


  proc sort data = __summ_stats3__ out = &out_data;
    by &subgroup &byvars &trtvar stat_row;
    %if "&out_data_where" ne "" %then %do;
     where &out_data_where;
  %end;

  data &out_data;
    retain &subgroup &byvars &trtvar stat_row linlabel
                   %if %qupcase(%substr(&trigger_separate_bigncol,1,1))=Y %then
%do; linlabel_bign %end;
                   stat_v:;
    set &out_data;
    keep &subgroup &byvars &trtvar stat_row linlabel
                  %if %qupcase(%substr(&trigger_separate_bigncol,1,1))=Y %then %do;
linlabel_bign %end;
                  stat_v:;
  run;
  proc sort data = &out_data;
    by &subgroup &byvars &trtvar stat_row;
  run;

%end;
%else %if %upcase(&display)=V %then %do;

proc sort data = __summ_stats1__;
  by &subgroup &byvars &trtvar;
run;

proc transpose data = __summ_stats1__ out=__summ_stats2__ prefix=result;
  by &subgroup &byvars &trtvar;
  var %do i=1 %to &num_of_statvar ;
         &&statvar&i.._c
      %end;
      %do i=1 %to &num_of_var;
          var&i._minmax_c  var&i._mstd_c
          %if &MSTE eq Y %then %do; var&i._mste_c %end;
```

```
          %if &Q1Q3 eq Y %then %do; var&i._q1q3_c %end;
          %if &P1P99 eq Y %then %do; var&i._p1p99_c %end;
          %if &P5P95 eq Y %then %do; var&i._p5p95_c %end;
          %if &P10P90 eq Y %then %do; var&i._p10p90_c %end;
          %if &LCLMUCLM eq Y %then %do; var&i._lclmuclm_c %end;
          %if &NUM1NUM2 eq Y %then %do; var&i._NUM1NUM2_c %end;
          %if &PNUM1NUM2P eq Y %then %do; var&i._PNUM1NUM2P_c %end;
          %if &NUM1PNUM2P eq Y %then %do; var&i._NUM1PNUM2P_c %end;
      %end;;
run;

%*check if there is only one value of &trtvar in dataset**;

proc sql noprint;
  select count(distinct &trtvar) into : __trtvaln__
  from __summ_stats2__;
quit;

data __summ_stats2__;
  set __summ_stats2__;
  length name1 $50 stat $ 20;

  stat= upcase(scan(_name_,2,'_'));

  %* If there is only one variable for which the stats are being derived then no
need to have variable number in output var names;
  %if &num_of_var ne 1 %then %do;
    name1='STAT_VALUE_'||compress(&trtvar)||'_'||substr(scan(_name_,1,'_'),4);
  %end;
  %else %do;
    name1='STAT_VALUE_'||compress(&trtvar);
  %end;
run;

proc sort data = __summ_stats2__;
  by &subgroup &byvars stat;
run;

proc transpose data = __summ_stats2__ out = __summ_stats3__;
   by &subgroup &byvars stat;
   var result1;
   id name1;
run;

proc sort data = __summ_stats3__;
  by stat;
run;

data __summ_stats3__;
   merge __summ_stats3__(in=a) __stat_format__(in=b);
```

```
   by stat;
   if a and b;
run;

proc sort data = __summ_stats3__ out = &out_data;
  by &subgroup &byvars stat_row;
  %if "&out_data_where" ne "" %then %do;
     where &out_data_where;
  %end;
run;

data &out_data;
  retain &subgroup &byvars stat_row stat_label stat_value_:;
  set &out_data;
  keep &subgroup &byvars stat_row stat_label stat_value_:;
run;
  proc sort data = &out_data;
    by &subgroup &byvars stat_row;
  run;

%end;  /*if display=V*/

%if %str(&repeat_if) ne %then %do;

   %mu_check_req_parameters
    ( parameters_to_check   =  repeat_by repeat_for
    , help                  = no
    ) ;

  %mu_check_data_and_var_exist
    ( data_to_check                     = &out_data
    , vars_to_check_in_all_data         = &repeat_by &repeat_for
    , vars_to_check_in_respective_data  =
    , abort_if_does_not_exist           = yes
    , help                              = no
    ) ;

    data __summ_if__;
      set &out_data;
      if &repeat_if;
      drop &repeat_for;
    run;

    %mu_nobs(__summ_if__);
    %if &__summ_if___nobs <= 0 %then %do;
        %let MA_SUMM_STATS_DISPLAY_RC=11;
        %put %str(ALERT_)C: &SYSMACRONAME - No observations satisfy condition
&repeat_if.;
        %put %str(ALERT_)C: &SYSMACRONAME - No observations will be repeated.;
    %end;
```

```
    proc sort data = &out_data out = __summ_repeat_for__ nodupkey;
      by &subgroup &repeat_by &repeat_for;
      where not (&repeat_if);
    run;

    proc sort data = __summ_if__;
      by &subgroup &repeat_by;
    run;

    %mu_wordscan
                ( string    = &repeat_by
                , root      = repeat_by_var
                , numw      = numrepeatby
                , delim     = %str( )
                ) ;

    %mu_wordscan
                ( string    = &repeat_for
                , root      = repeat_for_var
                , numw      = numrepeatfor
                , delim     = %str( )
                ) ;

    proc sql ;
       create table __summ_repeat__ as
       select /*%if "&subgroup" ne "" %then %do;
                  &subgroup,
              %end;*/
                    %do i=1 %to &numrepeatfor;
                        b.&&repeat_for_var&i,
                    %end;  a.*
       from __summ_if__ as a, __summ_repeat_for__ as b
       where %if "&subgroup" ne "" %then %do;
                  a.&subgroup=b.&subgroup and
              %end;
              %do i=1 %to &numrepeatby;
                a.&&repeat_by_var&i = b.&&repeat_by_var&i %if &i ne &numrepeatby
%then %do; and %end;
             %end;;
    quit;

    data __out_data__;
       set __summ_repeat__ (in=a) &out_data(in=b where=(not (&repeat_if)));
       if a then _repeat_ =1;
       else if b then _repeat_=2;
    run;

    proc sort data = __out_data__;
       by &subgroup &repeat_by &repeat_for _repeat_;
```

```
    run;

    data &out_data;
      retain &subgroup &repeat_by &repeat_for _repeat_;
      set __out_data__;
    run;
  proc sort data = &out_data;
    by &subgroup &repeat_by &repeat_for _repeat_;
  run;

%end;
%else %do;
    %if &repeat_by ne and &repeat_for ne %then %do;
        %put %str(ALERT_)C: &SYSMACRONAME - No value is supplied in REPEAT_IF
parameter.;
        %put %str(ALERT_)C: &SYSMACRONAME - But REPEAT_BY and REPEAT_FOR parameters
are populated.;
        %put %str(ALERT_)C: &SYSMACRONAME - No observations will be repeated.;
        %let &SYSMACRONAME._RC=10;
                %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
    %end;
    %else %if &repeat_by ne %then %do;
        %put %str(ALERT_)C: &SYSMACRONAME - No value is supplied in REPEAT_IF and
REPEAT_FOR parameters.;
        %put %str(ALERT_)C: &SYSMACRONAME - But REPEAT_BY parameter is populated.;
        %put %str(ALERT_)C: &SYSMACRONAME - No observations will be repeated.;
        %let &SYSMACRONAME._RC=10;
                %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
    %end;
    %else %if &repeat_for ne %then %do;
        %put %str(ALERT_)C: &SYSMACRONAME - No value is supplied in REPEAT_IF and
REPEAT_BY parameters.;
        %put %str(ALERT_)C: &SYSMACRONAME - But REPEAT_FOR parameter is populated.;
        %put %str(ALERT_)C: &SYSMACRONAME - No observations will be repeated.;
        %let &SYSMACRONAME._RC=10;
                %if "&&&mname._rc." ne "0" %then %let _checkrc=&&&mname._rc.;
    %end;
%end;


%*** STEP 6 - END OF PROCESS TASKS ;

%*** check to see if CHECK_RC has been set previously and update final macro value
reported,
     issue with checks which do not cause the macro to exit, being lost during
exection
     of later calls to MU_CHECK_RC, latest issue will be reported ***;

%if "&_checkrc" ne "" %then %let &MNAME._rc=&_checkrc;
```

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


%* Clean up;
%md_clean_and_reset(
    debug     =&debug
   ,_workdata = %str(&WORK_DATASETS_DATA &out_data)
   ,resetmprint = &mprint_setting
    );


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


%exit:

%if &&&sysmacroname._RC gt 0 and %upcase("&abort") = YES %then %do ;
    data _null_ ;
        abort ;
    run ;
%end ;


%mend ma_summ_stats_display;
```