```
%macro mu_create_format(
 in_data       =
,short_var     =
,long_var      =
,format_name   =
,add_to_format =
,other         =
,format_where  =
,format_global = Y
,format_global_name  = _default_format_name
,format_zeroobs_text = %str(No Observations)
,delim         = !
,quote_swap    = @
,allow_label_dup = N
,help          =
,debug         =
     ) / source store  des='V1.0.1.5';
/*************************************************************************
FILENAME: MU_CREATE_FORMAT.SAS
DEVELOPER: (b) (6)
PLATFORM: PC SAS 9.1.3, 9.2
DATE: 20120215

PURPOSE: This macro will create a format using two variables, the short value, and
the long value.
         It is designed to save programmers from having to type the entire format.
Please see examples
         below for clarification.

        Example: Imagine that you are asked to create a medical history table
listing the number of subjects
                 that had all their possible medical history listed in the CRF,
such as SKIN, HEENT, RESPIRATORY,
                 GASTROINTESTINAL, etc.  Assume the data has two variables, one
numeric (MHBDSYSN) and one character
                 (MHBODSYS).  The table should have MHBODSYS displayed, but in the
order found in MHBDSYSN.
                 Also, let us say out of 14 possible MHBODSYS in the CRF, 13 are
present in the data.
                 Nobody had HEPATIC, but it is to be displayed (with counts for all
treatment groups being zero).
                 Also, let's assume that there is no format provided.  This is the
type of situation where this macro
                 should prove useful and save the programmer from a lot of typing.
When using MA_COUNT_CATEGORICAL for
                 example, MHBDSYSN will be the categorical variable, and
CATVAR_PRELOADFMT will be the format created by
                 this macro. This way all the values found in the data plus the one
you add for HEPATIC by using the
```

```
                parameter ADD_TO_FORMAT (e.g: 11 = HEPATIC) will be part of the
format.  See USAGE section for examples:


MACROS USED:   MU_CHECKRC, MU_HELP_DEBUG, MD_WORKINFO, mu_check_req_parameters,
mu_check_data_and_var_exist,
               MU_WORDSCAN, md_clean_and_reset

PARAMETERS:
in_data       = Name of data set containing variable named in SHORT_VAR and/or
LONG_VAR parm  (REQUIRED)
short_var     = Variable name to be left side or start value of format  (e.g.,
MHBDSYSN)
                If no value is entered for SHORT_VAR, the value of LONG_VAR will be
used.
long_var      = Variable name to be right side or label value of format (e.g.,
MHBODSYS)
                If no value is entered for LONG_VAR, the value of SHORT_VAR will be
used.
format_name   = Name of this created format.  No need to put $ in front of name -
macro will know if it is a character format
                If no value is entered, the variable name of the SHORT_VAR will be
used and prefixed with F (F&SHORT_VAR) (OPTIONAL)
add_to_format = Add desired combination of short_var = long_var not found in data,
must be delimited with &DELIM. (OPTIONAL)
                It is not required to enter quotes.  Case value (upper/lower)
remains as entered.
                Enter the word BLANK to use a numeric or character blank value as
the short_var value.
other         = When a value is entered, will add OTHER = &other  to the format

delim         = for Usage with ADD_TO_FORMAT Parm:  Delimiter used to seperate sets
of short_var = long_var.  Default = !
quote_swap    = This parameter is no longer used.
format_where  = Enter where clause to subset data to be used in creating format.
format_global = Y- Yes, create Global macro variable holding format name (Default).
 Any other value will not create a global macro variable.
format_global_name = _default_format_name (DEFAULT).  Name of GLOBAL macro variable
to hold format name when FORMAT_GLOBAL=Y
allow_label_dup= N (Default) do not allow duplicate labels (long_var), Y = Allow
duplicate labels for unique short_var values
help          =
debug         =


USAGE:
   data a;
        mhbdsysn=1;mhchar='1';mhbodsys='A';output;
        mhbdsysn=2;mhchar='2';mhbodsys='B';output;
        mhbdsysn=3;mhchar='3';mhbodsys='C';output;
```

```
 run;

  Example 1: Creates format $MHBOD
        %mu_create_format(
            in_data     = a
           ,short_var   = mhchar
           ,long_var    = mhbodsys
           ,format_name = mhbod
              )

  Example 2: Creates format FMBOD with additional value ( ' ' = 'UNCODED TERM' )
        %mu_create_format(
            in_data       = a
           ,short_var     = mhbdsysn
           ,long_var      = mhbodsys
           ,format_name   = mhbod
           ,add_to_format = BLANK = UNCODED TERM
               )

  Example 3: Creates format $MHBOD with additional values
        %mu_create_format(
            in_data       = a
           ,short_var     = mhbdsysn
           ,long_var      = mhbodsys
           ,format_name   = mhbod
           ,add_to_format = 3 = X ! 4 = D ! 5 = EE
                )

  Example 4: Creates format $FMHCHAR from permanent sas data set and does not
create global macro variable containing format name
        %mu_create_format(
            in_data       = adb.adae
           ,short_var     =
           ,long_var      = aebodsys
           ,format_global = N
             )

  Example 5: Add OTHER to format and set to a missing value
        %mu_create_format(
            in_data       = adb.adae
            ,short_var     = xyz
            ,long_var      = xyz_text
            ,other         = BLANK
             )
  Example 6: Add OTHER to format and set to a value
        %mu_create_format(
            in_data       = adb.adae
            ,short_var     = xyz
            ,long_var      = xyz_text
            ,other         = 99
```

```
            )

(b) (6) Mods
0) Updated header
1) Updated to use md_workdata and md_clean_and_reset
2) removed short_var and Long_var from req_parms and added a separate check for
entry of either parm
3) If either short_var or long_var is empty, then set both to the one that is not
missing
4) Added usage of mu_checkrc
5) Moved build of format from data step to macro code
6) added format_where parm
7) added format_global parm and set to Y and format_global_name parm set to
_default_format_name
8) added BLANK usage for short_var value to change to blank value (char or num) to
be set to a label value (long_var)
---
9) allow macro to continue if IN_DATA data set is empty
10) create dummy format if IN_DATA data set is empty or no records meet
FORMAT_WHERE  criteria

23May2014 (b) (6) mods
Modified program to handle % and & embedded in short_var and long_var
Program uses data set to build format

06Nov2015 (b) (6) mod
Added OTHER parm and logic to add OTHER = " " to format.  When BLANK is entered
OTHER is set to missing

01Dec2015 (b) (6) upgrade to allow duplicate labels
*************************************************************************/

%local mname fprefix lib ds start label m2 dq fstart fend skipaddmore nstart
nlabel;
%let mname = &sysmacroname;

%* Alerts user to start of macro, sets up global rc variable and will initialze it
to 0;
%mu_checkrc(condition =  1=1
           ,rcprefix  = &mname
           ,msg1      = %str(&mname starts here )
             );

%mu_help_debug;

%* -- get current setting of mprint and list work data sets --;
%md_workinfo(debug = &debug);

%* --  Allow format name to be value of &SHORT_VAR with prefix of F if SHORT_VAR
was entered and is not too long --;
```

```
%if %bquote(&format_name) = %str( ) and %length(f&short_var) < 9 and
%bquote(&short_var) ne %str( )
    %then %let format_name = %bquote(&short_var);

%mu_check_req_parameters(
    parameters_to_check = in_data  format_name
   ,help = no
    );

%* -- SHORT_VAR or LONG_VAR must be entered, rc will be set to 9 ---;
 %mu_checkrc(condition = %length(&short_var) + %length(&long_var) = 0
            ,setabort  = Y
            ,rcprefix  = &mname
            ,msg1      = %str(SHORT_VAR and LONG_VAR both have null values, please
provide at least one)
             );

%* -- Verify SHORT_VAR and LONG_VAR variables are present in data set IN_DATA --;
%mu_check_data_and_var_exist(
    data_to_check                    = &in_data
   ,vars_to_check_in_all_data        = &short_var &long_var
   ,vars_to_check_in_respective_data =
   ,abort_if_does_not_exist          = yes
   ,debug                            = &debug
   ,help = no
      );

%* Determine if work data set or permanent data set was entered in IN_DATA for
MU_NOBS and SQL processing --;
%let in_data = %upcase(&in_data);
%if %index(%bquote(&in_data), %str(.)) >0 %then %do;
    %let lib = %scan(&in_data, 1, %str(.));
    %let ds  = %scan(&in_data, 2, %str(.));
%end;
%else %do;
    %let lib = WORK;
    %let ds  = &in_data;
%end;



    %* -- Verify only one variable was entered using macro variables created  --;
    %* --  in MU_CHECK_DATA_AND_VAR_EXIST when it ran mu_wordscan on the
vars_to_check_in_all_data  --;
    %mu_checkrc(condition= &numvar > 2
           ,rc       = 7
           ,rcprefix = &mname
           ,setabort = Y
           ,msg1     = %str(&mname:  Enter one (1) variable in each SHORT_VAR and
LONG_VAR)
```

```
           ,msg2     = %str(&mname:  SHORT_VAR = &short_var,   LONG_VAR =
&long_var)
                       );

%* -- Remove leading $ or ending . from format name --;
%let format_name = %sysfunc(compress(%bquote(&format_name), %str($.) ));

%* -- Check format name length --;
%mu_checkrc(condition  = %length(&format_name) > 8
           ,setabort = Y
           ,rc       = 3
           ,rcprefix = &mname
           ,msg1     = %str( Format name needs to be Eight (8) characters or less);
                );

%* -- Verify that user did not enter IF or THEN --;
%if %bquote(&format_where) ne %str( ) %then %do;
    %mu_checkrc(condition =  %sysfunc(indexw(%bquote(%upcase(&format_where)),
%str(IF) )) > 0  or
                              %sysfunc(indexw(%bquote(%upcase(&format_where)),
%str(THEN) )) > 0
               ,rc        = 8
               ,rcprefix  = &mname
               ,setabort  = Y
               ,msg1      = %str(&mname:  FORMAT_WHERE clause includes IF and/or
THEN)
               ,msg2      = %str(Remove IF/THEN from FORMAT_WHERE ==>
&format_where)
                )

%end;


%*************************************************************************;


%* -- If either short_var or long_var is missing, then set it to the other --;
%if       %bquote(&long_var)  = %str( ) %then %let long_var  = %bquote(&short_var);
%else %if %bquote(&short_var) = %str( ) %then %let short_var = %bquote(&long_var);



proc sql noprint;
   %* -- Determine short variable type and use to set format --;
     select upcase(type)
       into :type

       from dictionary.columns

      where upcase(libname) = "&lib"
```

```
        and upcase(memname) = "&ds"
        and upcase(name)    = upcase("&short_var")
           ;

  %* -- Create data set containing unique values for each &short_var &long_var
combination  --;
  %* --  Subset data set if a where clause was passed via parm
-------------------------------;

   create table makefmt as
   select distinct &short_var as start, &long_var as label

     from &in_data

    where &short_var is not null
      and &long_var  is not null

      %if %bquote(&format_where) ne %str( ) %then %do;
        and &format_where
      %end;
      ;

   select count(*)  into :nobs from makefmt;
quit;

%* -- Check that data set met any critieria entered, and there are
obs----------------------;
%mu_checkrc(condition = &nobs = 0
           ,setabort  = N
           ,rcprefix  = &mname
           ,rc        = 11
           ,alertid   = C
           ,msg1      = %str(No values exist in data set &IN_DATA for SHORT_VAR
and/or LONG_VAR)
           ,msg2      = %str(Subset processing passed was:  &format_where)
           ,msg3      = %str(A dummy format will be created)
              )



 %* -- Build Format ------------------------------------------------------;

 %if %str(&type) = CHAR %then %do;
     %let fprefix = %str($);
     %let blank   = %str( );
 %end;
 %else %let blank = %str(.);


 %if &mu_create_format_rc = 11 %then %do;   %* -- data set is  empty --;
```

```
     %if %length(&format_zeroobs_text) > 0 %then %do;
          proc format;
               value &fprefix.&format_name

                low-high = "&format_zeroobs_text" ;
          run;
     %end;

     %mu_checkrc(condition =  %length(&add_to_format) > 0
                ,rcprefix  = &mname
                ,msg1      = %str(&MNAME: ADD_TO_FORMAT values were not placed in
format)
                ,msg2      = %str(&MNAME: as they would have cause an overlap with
low-high = &format_zeroobs_text )
                 )

 %end;
 %else %do;
       %let nplus = ;

       %if %length(&add_to_format) > 0 %then %do;

           %* -- ADD_TO_FORMAT  Processing
----------------------------------------------;

           %let nplus = nplus;

           data nplus(keep = _label _start);
                length set $500 _label startx $200 %if %str(&type) = CHAR %then
%str( _start $200);;

                nsets = count("&add_to_format", "&delim") + 1;
                do i = 1 to nsets;
                   set = scan("&add_to_format", i, "&delim");

                   startx = scan(set, 1, '=');
                   if upcase(strip(startx)) = 'BLANK' then startx = ' ';

                   _start = %if %str(&type) = CHAR %then %str( strip(startx) );
                           %else                         %str( input(startx, best.)
);
                           ;

                   _label = strip(scan(set, 2, '='));
                   output;
                end;
           run;

           %* -- Review lengths of nplus versus the data --;
           proc contents data=makefmt out=orig;
```

```
           %let newstlength  =0;
           %let newlbllength =0;

           proc sql noprint;
              select max(length(_label)) into :plenlab from nplus;
              select length into :lenlab from orig where upcase(name) = 'LABEL';
              %if %str(&type) = CHAR %then %do;
                  select max(length(_start)) into :plenst from nplus;
                  select length into :lenst from orig where upcase(name) = 'START';
                  %if &plenst > &lenst %then %let newstlength = &plenst;
              %end;
              %if &plenlab > &lenlab %then %let newlbllength = &plenlab;
           quit;

           %* -- Assign label (and start if character) lengths based upon longer
values--;
           data nplus(keep = start label);
                length label $
                       %if &newlbllength = 0 %then %str(&lenlab);
                       %else                       %str(&newlbllength);
                       %if %str(&type) = CHAR %then %do; start $
                           %if &newstlength = 0 %then %str(&lenst);
                           %else                      %str(&newstlength);
                       %end; ;
            set nplus;

                start = _start;
                label = _label;
           run;

           %if &newlbllength >0 or &newstlength >0 %then %do;  %* -- add-to-format
lengths are longer than data --;
               data makefmt;
                    length %if &newlbllength >0 %then %str(label $&newlbllength);
                           %if &newstlength >0 %then %str(start $&newstlength);;

                set makefmt(rename = ( %if &newlbllength >0 %then %str(label =
_label);
                                       %if &newstlength  >0 %then %str(start =
_start);
                                       ));
                     %if &newlbllength >0 %then %str(label = _label;);
                     %if &newstlength  >0 %then %str(start = _start;);
                run;
           %end;
       %end;

       %local addoth;
       %if %md_notnull(OTHER) %then %do;
```

```
           %let addoth = other;
           data other;
              hlo = 'O';
              %if %upcase(&other) = BLANK %then %let other=;
              label = "&other";
           run;
       %end;

       data makefmt;
            retain fmtname "&fprefix.&format_name";
        set makefmt &nplus &addoth;
       run;

       %* -- Check for multiple start to different label values ---;
       proc sql;
        create table nstart as select count(distinct start) as found from makefmt
group by label;
        select found into :nstart from nstart where  found > 1 ;
        create table nlabel as select count(distinct label) as found from makefmt
group by start;
        select found into :nlabel from nlabel where found > 1 ;
       quit;
       proc delete data=nstart; run;
       proc delete data=nlabel; run;


       %if %upcase(&allow_label_dup) ne Y %then
       %mu_checkrc(condition = &nstart > 0
            ,setabort  = Y
            ,rcprefix  = &mname
            ,rc        = 7
            ,msg1      = %str( ONE (1) SHORT_VAR[&short_var] value has Multiple
LONG_VAR[&long_var] values)
            ,msg2      = %str(No format will be created)
              );

       %mu_checkrc(condition = &nlabel > 0
            ,setabort  = Y
            ,rcprefix  = &mname
            ,rc        = 7
            ,msg1      = %str(Multiple SHORT_VAR[&short_var] values exist for the
same LONG_VAR[&long_var] value)
            ,msg2      = %str(No format will be created)
              )


       proc format cntlin=makefmt ;
       run;
 %end;
```

```
 %if %str(&debug) = Y %then %do;
     proc format fmtlib lib=work;
          select &fprefix.&format_name;
     run;
 %end;

 %* -- Problem creating the format, Alert user, end macro here  --;
 %mu_checkrc(condition = &syserr > 0
            ,rcprefix  = &mname
            ,setabort  = Y
            ,msg1      = %str(&MNAME:  There was a problem creating the format
&fprefix&format_name )
            ,msg2      = %str(&MNAME:  Check the LOG for more information )
            ,msg3      = %str(&mname:  Ends here )
            )


 %* --  Create global variable and set to format name just created
--------------------------------;

 %if %bquote(&format_global) = %str( ) %then %let format_global = Y;   %* --
Default to Y if null --;

 %if %upcase(%substr(%bquote(&format_global), 1, 1)) = Y %then %do;
     %if %bquote(&format_global_name) = %str( ) %then  %let format_global_name =
_default_format_name;
     %global &format_global_name ;
     %let &format_global_name = &fprefix.&format_name;
     %let m2 = %str(Global macro variable %upcase(&format_global_name) created.
Value is %upcase(&&&format_global_name));
 %end;

 %md_clean_and_reset(
     debug       =&debug
    ,_workdata   = %str(&WORK_DATASETS_DATA)
    ,resetmprint = &mprint_setting
     );

 %* -- Alert user to end of macro, and info on format name/creation --;
 %mu_checkrc(condition =  1=1
    ,rcprefix  = &mname
    ,msg1      = %str(&MNAME:  Format %upcase(&fprefix&format_name) has been output
to the work format catalog)
    ,msg2      = %str(&m2)
    ,msg3      = %str(&mname:  Ends here )
           );

%mend mu_create_format;
```