```
%macro ml_create_meta(
 meta_data =
,in_data =
,var =
,usage =
,options =
,format =
,width =
,unit =
,prcode_split =
,escapechar =
,label =
,spanning_header =
,style_header =
,style_column =
,packed = N
,page_width_chars =
,debug =
,help =
) / store  source des='V1.0.0.14';

%*****************************************************************
FILENAME:    ML_Create_Meta.sas
DEVELOPER:    (b) (6)
PLATFORM:    SAS 9.1.3 and 9.2 on PC
MACROS USED: MU_HELP_DEBUG.sas, MD_WorkInfo.sas, MU_CHECK_DATA_AND_VAR_EXIST.sas,
MU_CHECK_REQ_PARAMETERS.sas,
                         MD_Default_Check.sas, MU_VAR_ATTRIBUTES, MD_Calc_Width,
MD_Max_Component_Length

ASSUMPTIONS: First call establishes defaults.  If using SAS interactive, macro
ML_Create_Meta_Reset will reset
                        the macro environment.

DESCRIPTION: This macro is to build metadata to be used by the listing macros
ML_Process_Meta and ML_Report.
                         This macro will be called multiple times, once for each
variable needed in a listing.

USAGE NOTES:  FINAL REQUIREMENTS will be based on final FRD   (current FRD v3 dated
04Jul2013)

Meta_Data = the metadata dataset to be created and/or appended.
   OPTIONAL, default on first call to this macro is Meta_Data.
   After the first call, a value for Meta_Data is not required.  If supplied, it is
compared to the value established
   in the first call.  If different, user is ALERTED, RC set to 7 and macro aborts.

   established in the first call.
```

In_Data = the input dataset that contains the data that is being listed.
   REQUIRED on the first call, no default. If data does not exist, ALERT, MU_CHECK_DATA_AND_VAR_EXIST_RC=3, abort.
        If missing, ALERT, MU_CHECK_REQ_PARAMETERS_RC=1, abort.
   After the first call, a value for In_Data is not required. If supplied, it is compared to the value established
   in the first call. If different, user is ALERTED, RC set to 8 and macro aborts.

Var = the name of the variable for which metadata is being defined. Call ML_CREATE_META for every var needed in the
   listing, including vars that are not displayed.
   REQUIRED and must be a valid SAS variable name that is present in &In_Data.
   If the value of VAR is missing, MU_CHECK_REQ_PARAMETERS_RC is set to 1, user is ALERTED, and the missing VAR is not
   added to Meta_Data.
   If VAR does not exist in IN_DATA, MU_CHECK_DATA_AND_VAR_EXIST_RC is set to 4, user is ALERTED, and the VAR is not
   added to Meta_Data.

Usage = how the variable will be used in Proc Report.
   OPTIONAL, default is DISPLAY.
   Valid values for Usage are: BY, DISPLAY, ORDER, SKIP and SKIP=. Some multiple values are allowed, but BY and DISPLAY must
   be single values only. ORDER may be used with SKIP. If SKIP= is specified without ORDER, then ORDER will be assumed.
   Valid values for SKIP= are 0, .5, 0.5 and 1. SKIP= is used in determining line count, and the cell height in the
   compute block of Proc Report.
   If Usage value is not valid, ML_CREATE_META_RC is set to 1, user is ALERTED, the value for Usage will take on the
   default value of DISPLAY, and Meta_Data is updated.
        After checking the SKIP=x part of the USAGE string, the =x part of the string is removed from USAGE, and the numeric
        value x is stored in a variable named SKIP. If there was o SKIP=x in USAGE, then SKIP will be missing.

Options = specifies options to be used in the DEFINE statement of Proc Report.
   OPTIONAL, default is null (no options).
   Valid values for Options are: ORDER=, DESCENDING, ID, NOPRINT and PAGE. Multiple values are allowed if they do not
   conflict with one another. Conflicts are: ID and NOPRINT, <others?>
   If the value of Usage is ORDER, then the option ORDER= will be set to ORDER=INTERNAL unless otherwise specified.
   If Option value is not valid, ML_CREATE_META_RC is set to 3, user is ALERTED, the value for Options will take on the
   default value of <null>, and Meta_Data is updated.

Format = specifies the name of a format to be used on VAR.
   OPTIONAL, no default.

```
   If a Format is specified it must exist, and must be appropriate for the VAR
(numeric format for a numeric VAR,
   character format for a character VAR).
   If Format does not exist, ML_CREATE_META_RC is set to 3, user is ALERTED, and
the macro is exited without updating
   &META_DATA.
   If Format does not match VAR, ML_CREATE_META_RC is set to 4, user is ALERTED,
and the macro is exited without updating
   &META_DATA.

Width = specifies the size of the column in the listing (also see UNIT parameter,
below).
   OPTIONAL, will default to missing.
   If a Width is specified it must be numeric.
   If Width is not numeric, ML_CREATE_META_RC is set to 5, user is ALERTED, and
Width is set to missing.  Meta_Data is
   updated.

Unit = specifies the unit for Width.
   OPTIONAL, default is P, for percent.  The first time users provides Unit,
default will be set to that Unit.
   Valid values for Unit are: P and C, for percent and characters.  Lower-case also
allowed.
   If not valid, set to default (P) and alert user.  Meta_Data is updated.

PrCode_Split = specifies the value of the split character used in Proc Report to
wrap (or flow) the headers.
   REQUIRED.  Default is global value?
   Use same parameter check as in pagebreak?    Note: Incorporate code from
MA_BIGN.  This will remove quotes, if
   necessary, and if blank, will default to $.

EscapeChar = specifies the value of the ODS escape character used to identify RTF
commands.
   REQUIRED.  Default is global value?
   Use same parameter check as in pagebreak?   Note: Incorporate code from MA_BIGN.
 This will remove quotes, if
   necessary, and if blank will alert user.   Difference in MA_BIGN not having a
default macro, vs this macro does.
   Check w Phyllis on this - make sure new code (from BIGN) is correct.

Label = specifies a label to be used for the VAR in the listing.
   OPTIONAL.  If no label is specified, it will inherit the label of VAR in
IN_DATA. If the lable of VAR in IN_DATA is
        null, it will inherit the var name (VAR).
   If a blank label is desired, use value BLANK (not case-sensitive).
   If &WIDTH has been provided, assess the length of LABEL as the longest string
between &PrCode_Split(s) or
   RTF line feed(s).   If the maximum length segment is greater than column width,
ML_CREATE_META_RC is set to 6,
```

```
   user is ALERTED, WIDTH in Meta_Data is updated with the maximum length of label
segment.
   The maximum length of label segment is recorded in the &Meta_Data in
LABEL_WIDTH.

Spanning_header = specifies a header for two or more columns.
   OPTIONAL.
   Use the same text in two or more adjacent columns, to force that text to be used
as a header across those columns.

Style_Header = specifies any SAS ODS style elements for a header in Proc Report,
EXCEPT CELLWIDTH=.
   OPTIONAL.
   Example JUST=C.
   CELLWIDTH cannot be set using this parameter.  Instead, use the WIDTH parameter.
 If CELLWIDTH= appears in the
        Style_Header string, user will be alerted, and the CELLWIDTH=string will be
removed from the parameter.

Style_Column = specifies any SAS ODS style elements to be applied to the column in
Proc Report, EXCEPT CELLWIDTH=.
   OPTIONAL.
   Example JUST=C.
   CELLWIDTH cannot be set using this parameter.  Instead, use the WIDTH parameter.
 If CELLWIDTH= appears in the
        Style_Column string, user will be alerted, and the CELLWIDTH=string will be
removed from the parameter.

Packed = specifies if the &VAR is already packed.
   OPTIONAL.  Default is N for no.  Valid values are Y or N (not case sensitive).
If invalid value, default to N.

Page_Width_Chars = specifies the width of the page in characters.
   OPTIONAL.  Default is &_default_page_width_chars.
   If value is not valid, ALERT user and set to 100.

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).
   OPTIONAL.  Default is global &_default_debug.

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

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

%*** RETURN CODES:
0 = no errors
1 = Usage parameter invalid, set usage param to DISPLAY, notify user, continue
```

```
2 = Options parameter invalid, set option param to null, notify user, continue
3 = Format parameter contains name of a format that does not exist, notify user,
set format to null, continue
4 = Format parameter contains a format not appropriate for VAR, notify user, set
format to null, continue
5 = Width parameter is not numeric, notify user, set width to ., continue
6 = Label length exceeds column width, notify user, continue
7 = Meta_Data value changed from default at first call, notify user, abort
8 = IN_Data value changed from default at first call, notify user, abort
9 = PrCode_Split value changed from default at first call, notify user, continue
10= EscapeChar value changed from default at first call, notify user, continue
11= Page_Width_Chars value changed from default at first call, notify user,
continue
;

   %* Other return codes, from other macro vars:
   MU_CHECK_DATA_AND_VAR_EXIST_RC = 3 when IN_DATA does not exist
   MU_CHECK_REQ_PARAMETERS_RC = 1 when IN_DATA or VAR is missing
   MU_CHECK_DATA_AND_VAR_EXIST_RC = 4 when VAR does not exist
   ;

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

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

%mu_help_debug;
%md_workinfo(debug = &debug );

%let abort = no;

%* Requirement 1 - macro can be called multiple times in same program  (no code
required here - handled in defaults
        section) ;

%**** Requirement 2, 4, 11, 12 and 18 AT FIRST INVOCATION  for Meta_Data, IN_Data,
PrCode_Split, EscapeChar
                and Page_Width_Chars.     ;
%*  At first invocation, establish defaults that will be defaulted across calls to
this macro.
        These include: Meta_Data, IN_Data, PrCode_Split, EscapeChar and
Page_Width_Chars.
        These values should not change across calls to ML_Create_Meta.
        Check for existence of global macro variable ML_Create_Meta_First_Call.  If
it does not exist yet, then this
```

```
        is the first call to this macro, and these defaults can be established.  ;

        %if %symexist(ML_Create_Meta_First_Call) = 0 %then %do;
        %global ML_Create_Meta_First_Call ;
                %LET ML_Create_Meta_First_Call = Y;
                %PUT %STR(ALERT_I: This is the first call to ML_Create_Meta.);
        %end;
        %else %LET ML_Create_Meta_First_Call = N;

        %* If first call, establish defaults for Meta_Data, IN_Data, PrCode_Split,
EscapeChar and Page_Width_Chars ;
        %IF %str(&ML_Create_Meta_First_Call) = %str(Y)
        %THEN %DO;
                %GLOBAL ML_Create_Meta_Data_Default  ML_Create_IN_Data_Default
ML_Create_Split_Default
                                ML_Create_EscapeChar_Default
ML_Create_Page_Width_Default ;

                %* Default for Meta_Data ;
                %IF %BQUOTE(&META_DATA)  = %STR( )
                        %THEN %DO;
                                %put %STR(ALERT_I: --- &sysmacroname var=&var
---------------------------------------);
                                %put %STR(ALERT_I: Parameter Meta_Data is null, so
default is set to [Meta_Data]);
                                %put %STR(ALERT_I:
-------------------------------------------------------------------);
                                %LET Meta_Data = Meta_Data;
                                %LET Meta_Data_Level2 = Meta_Data;
                        %END;
                        %ELSE %DO;
                                %LET period_position =%index(&Meta_Data,.);
                                %LET Meta_Data_Level2 =
%trim(%substr(&Meta_Data,&period_position+1));
                                %IF %SYSFUNC ( NVALID (&Meta_Data_Level2 , V7)) = 0

                                %THEN %DO;
                                        %put %STR(ALERT_I: --- &sysmacroname
var=&var ---------------------------------------);
                                        %put %STR(ALERT_I: Parameter Meta_Data is
[&Meta_Data.] which is not a valid name.);
                                        %put %STR(ALERT_I: Meta_Data will default
to [Meta_Data]);
                                        %put %STR(ALERT_I:
-------------------------------------------------------------------);
                                        %LET Meta_Data = Meta_Data;
                                %END;
                %END;
                %LET ML_Create_Meta_Data_Default = &Meta_Data;
```

```
                %* If the dataset &ML_Create_Meta_Data_Default already exists,
alert user that data will be overwritten ;
                %let _check_meta_data_exist =
%sysfunc(exist(&ML_Create_Meta_Data_Default));
                %IF &_CHECK_META_DATA_EXIST EQ 1
                        %THEN %DO;
                                %put %STR(ALERT_I: --- &sysmacroname var=&var
---------------------------------------);
                                %put %STR(ALERT_I: The dataset
[&ML_Create_Meta_Data_Default.] already exists.);
                                %put %STR(ALERT_I: The dataset will be
overwritten.);
                                %put %STR(ALERT_I:
-------------------------------------------------------------------);

                        %END;

                %* Default for IN_Data ;
                %IF %bquote(&in_data)  = %str( )
                        %THEN %DO;
                                %put %STR(ALERT_R: --- &sysmacroname var=&var
---------------------------------------);
                                %put %STR(ALERT_R: Parameter IN_Data is null.
There is no default for IN_Data.);
                                %put %STR(ALERT_R: Macro will abort.);
                                %put %STR(ALERT_R: You may need to run
ML_CREATE_META_RESET);
                                %put %STR(ALERT_R:
-------------------------------------------------------------------);
                                %let ML_CREATE_META_RC = 12;
                                %LET abort = Y;
                                %LET ML_Create_IN_Data_Default = ;
                                %GOTO EXIT;
                        %END;

                        %* If the dataset &IN_Data does not exist, alert user  and
abort ;
                        %mu_check_data_and_var_exist
                        (data_to_check = &IN_Data
                        ,abort_if_does_not_exist = no
                        ,help = no
                        )
                        %IF &MU_CHECK_DATA_AND_VAR_EXIST_RC EQ 3
                        %THEN %DO;
                                %put %STR(ALERT_R: --- &sysmacroname var=&var
---------------------------------------);
                                %put %STR(ALERT_R: The dataset &IN_Data does not
exist.  Please check IN_Data value.);
                                %put %STR(ALERT_R: Macro will abort.);
                                %put %STR(ALERT_R: You may need to run
```

```
ML_CREATE_META_RESET);
                                %put %STR(ALERT_R:
------------------------------------------------------------------);
                                %LET abort = Y;

                                %LET ML_Create_IN_Data_Default = ;
                                %GOTO EXIT;
                        %END;
                        %LET ML_Create_IN_Data_Default = &IN_Data;

                %* Default for PrCode_Split ;
                %IF %BQUOTE(&PrCode_Split) EQ %STR( )
                        %THEN %DO;
                                %md_default_check
                                (mparm_name= PrCode_Split
                                ,md_default_mvar   = _default_prcode_split
                                ,md_default_value  = %STR($)
                                )
                        %END;

                        %LET ML_Create_Split_Default = &PrCode_Split ;

                %* Default for EscapeChar ;
                %IF %BQUOTE(&EscapeChar) EQ %STR( )
                        %THEN %DO;
                                %md_default_check
                                (mparm_name= EscapeChar
                                ,md_default_mvar = _default_escapechar
                                ,md_default_value  = %STR(^)
                                )
                        %END;
                        %LET ML_Create_EscapeChar_Default = &EscapeChar ;

                %* Default for Page_Width_Chars ;
                %IF %BQUOTE(&Page_Width_Chars) EQ %STR( )
                        %THEN %DO;
                                %md_default_check
                                (mparm_name = Page_Width_Chars
                                ,md_default_mvar= _default_page_width_chars
                                ,md_default_value  = 100
                                )
                        %END;
                        %ELSE %DO;
                                %IF  ( %sysfunc(notdigit(&Page_Width_Chars)) GT 0 )
%THEN %DO;
                                        %PUT %STR(ALERT_C: --- &sysmacroname
var=&var ----------------------------------------------------);
                                        %PUT %STR(ALERT_C: Page_Width_Chars is not
a positive integer.  It is [&Page_Width_Chars.]);
                                        %PUT %STR(ALERT_C: Page_Width_Chars will
```

```
default to 100.);
                                        %PUT %STR(ALERT_C:
----------------------------------------------------------------------------);
                                        %LET Page_Width_Chars = 100;

                                %END;
                        %END;

                        %LET ML_Create_Page_Width_Default = &Page_Width_Chars ;
        %END;
%**** END of Requirement 2, 4, 11, 12 and 18 for Meta_Data, IN_Data, PrCode_Split,
EscapeChar and
                        Page_Width_Chars AT FIRST INVOCATION. ;

%**** Requirement 2, 4, 11, 12 and 18 for Meta_Data, IN_Data, PrCode_Split,
EscapeChar and
                        Page_Width_Chars NOT FIRST INVOCATION. ;
        %* If not first call, check values of Meta_Data, IN_Data, PrCode_Split,
EscapeChar and Page_Width_Chars against
                defaults ;
        %IF %str(&ML_Create_Meta_First_Call) = %str(N)
        %THEN %DO;
                %IF %STR(&Meta_Data) EQ %STR( ) %THEN %LET Meta_Data =
&ML_Create_Meta_Data_Default;
                %LET period_position =%index(&Meta_Data,.);
                                %LET Meta_Data_Level2 =
%trim(%substr(&Meta_Data,&period_position+1));
                %IF %STR(&Meta_Data) NE %STR(&ML_Create_Meta_Data_Default)
                        %THEN %DO;
                                %put %STR(ALERT_R: --- &sysmacroname var=&var
----------------------------------------);
                                %put %STR(ALERT_R: This is not the first call to
ML_Create_Meta.  The Meta_Data dataset was);
                                %put %STR(ALERT_R: already established, in the
first call, as [&ML_Create_Meta_Data_Default.]);
                                %put %STR(ALERT_R: The parameter Meta_Data, on this
call, has value [&Meta_Data.]);
                                %put %STR(ALERT_R: Macro will abort.);
                                %put %STR(ALERT_R: You may need to run
ML_CREATE_META_RESET);
                                %put %STR(ALERT_R:
----------------------------------------------------------------);
                                %LET ML_Create_Meta_RC = 7 ;
                                %LET abort = Y;
                                %GOTO EXIT;
                        %END;
                %IF %STR(&IN_Data) EQ %STR( ) %THEN %LET IN_Data =
&ML_Create_IN_Data_Default;
                %ELSE %IF %STR(&IN_Data) NE %STR(&ML_Create_IN_Data_Default)
                        %THEN %DO;
```

```
                                %put %STR(ALERT_R: --- &sysmacroname var=&var
----------------------------------------);
                                %put %STR(ALERT_R: This is not the first call to
ML_Create_Meta.  The IN_Data dataset was);
                                %put %STR(ALERT_R: already established, in the
first call, as [&ML_Create_IN_Data_Default.]);
                                %put %STR(ALERT_R: The parameter IN_Data, on this
call, has value [&IN_Data.]);
                                %put %STR(ALERT_R: Macro will abort.);
                                %put %STR(ALERT_R: You may need to run
ML_CREATE_META_RESET);
                                %put %STR(ALERT_R:
----------------------------------------------------------------);
                                %LET ML_Create_Meta_RC = 8 ;
                                %LET abort = Y;
                                %GOTO EXIT;
                        %END;
                %IF "&PrCode_Split" EQ "" %THEN %LET PrCode_Split =
&ML_Create_Split_Default;
                %ELSE %IF "&PrCode_Split" NE "&ML_Create_Split_Default"
                        %THEN %DO;
                                %put %STR(ALERT_C: --- &sysmacroname var=&var
----------------------------------------);
                                %put %STR(ALERT_C: This is not the first call to
ML_Create_Meta.  The parameter PrCode_Split    );
                                %put %STR(ALERT_C: was already established, in the
first call, as [&ML_Create_Split_Default.]   );
                                %put %STR(ALERT_C: The parameter PrCode_Split, on
this call, has value [&PrCode_Split.]);
                                %put %STR(ALERT_C: PrCode_Split will be set to
[&ML_Create_Split_Default.]);
                                %put %STR(ALERT_C:
----------------------------------------------------------------);
                                %LET ML_Create_Meta_RC = 9 ;
                                %LET PrCode_Split = &ML_Create_Split_Default ;
                        %END;
                %IF "&EscapeChar" EQ "" %THEN %LET EscapeChar =
&ML_Create_EscapeChar_Default;
                %ELSE %IF "&EscapeChar" NE "&ML_Create_EscapeChar_Default"
                        %THEN %DO;
                                %put %STR(ALERT_C: --- &sysmacroname var=&var
----------------------------------------);
                                %put %STR(ALERT_C: This is not the first call to
ML_Create_Meta.  The parameter EscapeChar);
                                %put %STR(ALERT_C: was already established, in the
first call, as [&ML_Create_EscapeChar_Default.]);
                                %put %STR(ALERT_C: The parameter EscapeChar, on
this call, has value [&EscapeChar.]);
                                %put %STR(ALERT_C: EscapeChar will be set to
[&ML_Create_EscapeChar_Default.]);
```

```
                                %put %STR(ALERT_C:
------------------------------------------------------------------);
                                %LET ML_Create_Meta_RC = 10 ;
                                %LET EscapeChar = &ML_Create_EscapeChar_Default ;
                        %END;
                %IF %STR(&Page_Width_Chars) EQ %STR( ) %THEN %LET Page_Width_Chars
= &ML_Create_Page_Width_Default;
                %ELSE %IF %STR(&Page_Width_Chars) NE
%STR(&ML_Create_Page_Width_Default)
                        %THEN %DO;
                                %put %STR(ALERT_C: --- &sysmacroname var=&var
---------------------------------------);
                                %put %STR(ALERT_C: This is not the first call to
ML_Create_Meta.  The parameter Page_Width_Chars);
                                %put %STR(ALERT_C: was already established, in the
first call, as [&ML_Create_Page_Width_Default.]);
                                %put %STR(ALERT_C: The parameter Page_Width_Chars,
on this call, has value [&Page_Width_Chars.]);
                                %put %STR(ALERT_C: Page_Width_Chars will be set to
[&ML_Create_Page_Width_Default.]);
                                %put %STR(ALERT_C:
------------------------------------------------------------------);
                                %LET ML_Create_Meta_RC = 11 ;
                                %LET Page_Width_Chars =
&ML_Create_Page_Width_Default ;
                        %END;
        %END;
%**** END of Requirement 2, 4, 11, 12 and 18 for Meta_Data, IN_Data, PrCode_Split,
EscapeChar and
                        Page_Width_Chars  NOT FIRST INVOCATION  ;

%**** Requirement 5 - VAR ;
%* Check that the required parameter VAR has been defined.  If VAR missing, no
update. R5. ;
        %* upper case &VAR since it is used in the update, below ;
        %LET VAR = %upcase(&VAR);
        %mu_check_req_parameters(parameters_to_check = VAR,help = no) *;
                %IF &MU_Check_Req_Parameters_RC EQ  1
                        %THEN %DO;
                                %put %STR(ALERT_R: --- &sysmacroname var=&var
----------------------------------------);
                                %put %STR(ALERT_R: Parameter VAR is null.  There is
no default for VAR.);
                                %PUT %STR(ALERT_R: Meta_Data dataset will not be
updated.);
                                %put %STR(ALERT_R:
------------------------------------------------------------------);
                                %GOTO EXIT;
                        %END;
```

```
%* Check that the required parameter VAR exists in IN_DATA.  R5.   ;
        %mu_check_data_and_var_exist(data_to_check = &IN_DATA
                                    ,vars_to_check_in_all_data = &VAR
                                    ,abort_if_does_not_exist = no
                                    ,help = no
                                    )
        %IF &MU_CHECK_DATA_AND_VAR_EXIST_RC EQ 4
                %THEN %DO;
                        %PUT %STR(ALERT_R: --- &sysmacroname var=&var
-------------------------------------------);
                        %PUT %STR(ALERT_R: The IN_DATA dataset [&IN_DATA.] DOES NOT
CONTAIN A VAR NAMED [&VAR.] );
                        %PUT %STR(ALERT_R: Meta_Data dataset will not be updated.);
                        %PUT %STR(ALERT_R:
-----------------------------------------------------------------------);
                        %GOTO EXIT;
                %END;
%* END of VAR parameter checks  R5 ;

%* Check USAGE parameter.    R6.    ;
%* Several USAGE flags are used to check across params.  Set these all to 0 here ;
        %local usage_by usage_display usage_order usage_skip  ML_SKIP_VALUE
skip_val  usage_display_default ;
                        %let usage_by = 0;
                        %let usage_display = 0;
                        %let usage_order = 0;
                        %let usage_skip = 0;
                        %let ML_SKIP_VALUE = . ;
                        %let skip_val = %STR( );

        %IF %BQUOTE(&USAGE) EQ %STR( ) %THEN
                        %DO;
                                %LET USAGE = DISPLAY;
                                %let usage_display = 1;
                                %let usage_display_default = 1 ;
                                %let ML_SKIP_VALUE = . ;
                                %GOTO ENDUSAGE;
                        %END;

        %LET USAGE = %upcase(&USAGE);

%* Make sure that none of the key words exist more than once in USAGE.     ;
                %local ML_Create_Keyword_Bad ;
                %let ML_Create_Keyword_Bad = 0 ;

                %macro md_keyword_more_than_once_ (USAGE=,_KeyWord_=) / des="CALLED
FROM ML_CREATE_META";
                  %let _numtimes_ = %SYSFUNC ( COUNT ( &USAGE , %STR(&_KeyWord_) ))
 ;
```

```
                  %IF %EVAL ( &_numtimes_ GT 1) %THEN
                        %DO;
                                %PUT %STR(ALERT_R: --- &sysmacroname var=&var
----------------------------------------------);
                                %PUT %STR(ALERT_R: The USAGE parameter contains
more than one instance of &_KeyWord_..    );
                                %PUT %STR(ALERT_R: USAGE will be set to DISPLAY );
                                %PUT %STR(ALERT_R:
----------------------------------------------------------------------);
                                %LET ML_CREATE_Keyword_Bad = 1;

                        %END;
                %mend md_keyword_more_than_once_ ;

                %md_keyword_more_than_once_ (USAGE=&USAGE,_KeyWord_ =BY)
                %md_keyword_more_than_once_ (USAGE=&USAGE,_KeyWord_ =DISPLAY)
                %md_keyword_more_than_once_ (USAGE=&USAGE,_KeyWord_ =ORDER)
                %md_keyword_more_than_once_ (USAGE=&USAGE,_KeyWord_ =SKIP)


                %IF &ML_Create_Keyword_Bad = 1 %THEN
                        %DO;
                                %LET USAGE = DISPLAY ;
                                %let usage_by = 0;
                                %let usage_display = 1;
                                %let usage_order = 0;
                                %let usage_skip = 0;
                                %let ML_SKIP_VALUE = . ;

                                %LET ML_CREATE_Meta_RC = 1;
                                %GOTO ENDUSAGE;
                        %END;

                %IF %index(&USAGE,BY) GT 0 %THEN %LET USAGE_BY = 1;
                %IF %index(&USAGE,DISPLAY) GT 0 %THEN %LET USAGE_DISPLAY = 1;
                %IF %index(&USAGE,ORDER) GT 0 %THEN %LET USAGE_ORDER = 1;
                %IF %index(&USAGE,SKIP) GT 0 %THEN %LET USAGE_SKIP = 1;

                %* eliminate multiple blanks ;
                %LET USAGE = %SYSFUNC ( COMPBL ( &USAGE) ) ;

                %* If USAGE contains SKIP but no = sign, default to SKIP=1 and
alert user ;
                %LET _x_ = %index(%STR(&USAGE),SKIP);
                %LET _y_ = %index(%STR(&USAGE),=);

                %IF &_x_ GT 0 AND &_y_ = 0 %THEN
                        %DO;
                                %LET USAGE = %SYSFUNC ( TRANWRD ( &USAGE , SKIP ,
SKIP=1)) ;
```

```
                                %PUT %STR(ALERT_I: --- &sysmacroname var=&var
----------------------------------------------);
                                %PUT %STR(ALERT_I: The keyword SKIP in parameter
USAGE was not followed by = , so SKIP    );
                                %PUT %STR(ALERT_I: will default to SKIP=1.);
                                %PUT %STR(ALERT_I:
-------------------------------------------------------------------);
                                %let ML_SKIP_VALUE = 1 ;
                                %GOTO ENDSKIP;
                        %END;

                %* The characters after the = sign must be a number ;
                %ELSE %IF  &_y_ GT  0 %THEN
                        %DO;
                                %LET Temp_Usage_Part1 = %SUBSTR(&USAGE,1,&_y_);
                                %LET Temp_Usage_Part2 =
%SUBSTR(&USAGE,%EVAL(&_y_+1));
                                %LET Skip_Val = %sysfunc ( SCAN
(&Temp_USAGE_Part2,1," )) ;

                                %LET Index3 = %INDEX (&Temp_Usage_Part2, %STR( ) )
;
                                %IF &Index3 = 0 %THEN %LET Temp_Usage_Part3 = ;
                                %ELSE %LET Temp_Usage_Part3 = %SUBSTR
(&Temp_USAGE_Part2,&Index3) ;

                                %put Skip_Val is &Skip_Val ;

                                %IF (%datatyp(&Skip_Val) NE NUMERIC) %THEN
                                    %DO;
                                            %PUT %STR(ALERT_I: ---
&sysmacroname var=&var ----------------------------------------------);
                                            %PUT %STR(ALERT_I: The characters
following SKIP= in parameter USAGE are not numeric.     );
                                            %PUT %STR(ALERT_I: They are
[&Skip_Val.] );
                                            %PUT %STR(ALERT_I: SKIP=
[&Skip_Val.] will default to SKIP = 1.);
                                            %PUT %STR(ALERT_I:
-------------------------------------------------------------------);
                                            %LET ML_CREATE_META_RC = 1;
                                            %LET USAGE = &Temp_Usage_Part1.1
&Temp_Usage_Part3. ;
                                            %let ML_SKIP_VALUE = 1 ;
                                            %GOTO ENDSKIP;
                                    %END;
                                %IF (%datatyp(&Skip_Val) EQ NUMERIC) %THEN
                                    %DO;
                                      %IF &Skip_Val LT 0 %THEN %DO;
                                            %PUT %STR(ALERT_I: ---
```

```
&sysmacroname var=&var ----------------------------------------------);
                                                %PUT %STR(ALERT_I: The number
following SKIP= in parameter USAGE is less than 0.);
                                                %PUT %STR(ALERT_I: It is
[&Skip_Val.] );
                                                %PUT %STR(ALERT_I: SKIP=
[&Skip_Val.] will default to SKIP = 1.);
                                                %PUT %STR(ALERT_I:
-------------------------------------------------------------------);
                                                %LET ML_CREATE_META_RC = 1;
                                                %LET USAGE = &Temp_Usage_Part1.1
&Temp_Usage_Part3. ;
                                                %let ML_SKIP_VALUE = 1 ;
                                                %GOTO ENDSKIP;
                                            %END;
                                            %let ML_Skip_Value = &Skip_Val ;
                                        %END;
                        %END;
%ENDSKIP:

%put USAGE is [&USAGE] ;

                        %LET USAGE_EDIT = &USAGE ;

                        %* Remove BY, DISPLAY, ORDER and SKIP=&Skip_Val.   ;
                        %* If any chars left, then those characters must be
invalid. ;
                        %LET USAGE_EDIT = %SYSFUNC ( TRANWRD ( &USAGE_EDIT , BY ,
)) ;
                        %LET USAGE_EDIT = %SYSFUNC ( TRANWRD ( &USAGE_EDIT ,
DISPLAY , )) ;
                        %LET USAGE_EDIT = %SYSFUNC ( TRANWRD ( &USAGE_EDIT , ORDER
, )) ;

                        %LET USAGE_EDIT = %SYSFUNC ( TRANWRD ( &USAGE_EDIT ,
%STR(SKIP=&ML_Skip_VALUE), )) ;

                        %IF "&USAGE_EDIT" NE "" %THEN
                                %DO;
                                        %PUT %STR(ALERT_R: --- &sysmacroname
var=&var ----------------------------------------------);
                                        %PUT %STR(ALERT_R: The USAGE parameter
contains one or more invalid words: &USAGE_EDIT    );
                                        %PUT %STR(ALERT_R: USAGE will be set to
DISPLAY);
                                        %PUT %STR(ALERT_R:
-------------------------------------------------------------------);
                                        %LET ML_CREATE_META_RC = 1;
                                        %LET USAGE = DISPLAY ;
                                        %let usage_by = 0;
```

```
                                        %let usage_display = 1;
                                        %let usage_order = 0;
                                        %let usage_skip = 0;
                                        %let ML_SKIP_VALUE = .  ;
                                        %GOTO ENDUSAGE ;
                                  %END;
%SKIPUSAGE:
                        %* Check if USAGE contains BY or DISPLAY with other words ;
                        %IF %EVAL (&USAGE_BY EQ 1 OR &USAGE_DISPLAY EQ 1)
                                        %THEN %IF  %EVAL (&usage_by +
&usage_display + &usage_order + &usage_skip ) GT 1
                                                %THEN %DO;
                                                        %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ----------------------------------------);
                                                        %PUT %STR(ALERT_R: The
USAGE parameter contains BY or DISPLAY plus other values. );
                                                        %PUT %STR(ALERT_R: In
USAGE, BY or DISPLAY can only be used as a single value.   );
                                                        %PUT %STR(ALERT_R: USAGE
will be set to DISPLAY);
                                                        %PUT %STR(ALERT_R:
----------------------------------------------------------------);
                                                        %LET ML_CREATE_META_RC = 1;
                                                        %LET USAGE = DISPLAY ;
                                                        %let usage_by = 0;
                                                        %let usage_display = 1;
                                                        %let usage_order = 0;
                                                        %let usage_skip = 0;
                                                        %let ML_Skip_Value = . ;
                                                        %GOTO ENDUSAGE ;
                                                %END;

                        %* IF BY or DISPLAY is not specified, and SKIP is specified
only once, and ORDER is not specified, ;
                        %* then assume ORDER.

                                ;
                        %IF %EVAL ( &USAGE_BY NE 1 AND &USAGE_DISPLAY NE 1 AND
&usage_skip EQ  1 AND  &USAGE_ORDER NE 1  )
                                                %THEN %DO;
                                                        %PUT %STR(ALERT_I: ---
&sysmacroname var=&var ----------------------------------------);
                                                        %PUT %STR(ALERT_I: The
USAGE parameter contains SKIP, but not ORDER.);
                                                        %PUT %STR(ALERT_I: ORDER
will be assumed.);
                                                        %PUT %STR(ALERT_I:
----------------------------------------------------------------);

                                                        %LET USAGE = &USAGE   ORDER
```

```
  ;
                                                        %LET USAGE_ORDER = 1;
                                                  %END;
%ENDUSAGE:
%* END of USAGE parameter checks  ;

%* Check OPTIONS parameter.  OPTIONS are not required.  There is no default.   R7.
;
%* Several OPTIONS flags are used to check across params.  Set these all to 0 here
;
        %local  options_descending options_id options_noprint options_page
                options_orderdata options_orderinternal options_orderformatted
                options_edit  options_temp ;
        %let options_descending = 0;
        %let options_id = 0;
        %let options_noprint = 0;
        %let options_page = 0 ;
        %let options_orderdata = 0;
        %let options_orderinternal = 0;
        %let options_orderformatted = 0;
        %LET OPTIONS = %upcase(&OPTIONS);

        %IF "&OPTIONS" = "" %THEN %GOTO SKIPOPTIONS;

        %* eliminate multiple blanks ;
        %LET OPTIONS = %SYSFUNC ( COMPBL ( &OPTIONS) ) ;
        %* eliminate blanks between ORDER and = if any ;
        %LET OPTIONS = %SYSFUNC ( TRANWRD ( &OPTIONS , ORDER = , ORDER=)) ;
        %* eliminate blanks between = and any valid text for ORDER  ;
        %LET OPTIONS = %SYSFUNC ( TRANWRD ( &OPTIONS ,= DATA, =DATA)) ;
        %LET OPTIONS = %SYSFUNC ( TRANWRD ( &OPTIONS ,= INTERNAL, =INTERNAL)) ;
        %LET OPTIONS = %SYSFUNC ( TRANWRD ( &OPTIONS ,= FORMATTED, =FORMATTED)) ;

        %* Before using the macro _valid_parts_, below, to check each piece of
OPTIONS, make sure that none of  ;
        %* the key words exist more than once in OPTIONS.

;
                %macro md_keyword_more_than_once_ (OPTIONS=,_KeyWord_=) /
des="CALLED FROM ML_CREATE_META";
                  %let _numtimes_ = %SYSFUNC ( COUNT ( &OPTIONS , &_KeyWord_ ))  ;

                  %IF %EVAL ( &_numtimes_ GT 1)
                        %THEN %DO;
                                %PUT %STR(ALERT_R: --- &sysmacroname var=&var
-----------------------------------------------);
                                %PUT %STR(ALERT_R: The OPTIONS parameter contains
more than one instance of &_KeyWord_..  );
                                %PUT %STR(ALERT_R: OPTIONS will be set to null);
                                %PUT %STR(ALERT_R:
```

```
-------------------------------------------------------------------------);
                                %LET ML_CREATE_META_RC = 2;
                                %let options_descending = 0;
                                %let options_id = 0;
                                %let options_noprint = 0;
                                %let options_page = 0 ;
                                %let options_orderdata = 0;
                                %let options_orderinternal = 0;
                                %let options_orderformatted = 0;
                        %END;
                %mend md_keyword_more_than_once_ ;
                %md_keyword_more_than_once_ (OPTIONS=&OPTIONS,_KeyWord_ =ORDER)
                %md_keyword_more_than_once_ (OPTIONS=&OPTIONS,_KeyWord_
=DESCENDING)
                %md_keyword_more_than_once_ (OPTIONS=&OPTIONS,_KeyWord_ =ID)
                %md_keyword_more_than_once_ (OPTIONS=&OPTIONS,_KeyWord_ =NOPRINT)
                %md_keyword_more_than_once_ (OPTIONS=&OPTIONS,_KeyWord_ =PAGE)


                %IF &ML_CREATE_META_RC EQ 2
                        %THEN %DO;
                                %LET OPTIONS = SETNULL;
                                %let options_descending = 0; %let options_id = 0;
                                %let options_noprint = 0;  %let options_page = 0 ;
                                %let options_orderdata = 0; %let
options_orderinternal = 0;
                                %let options_orderformatted = 0;
                                %GOTO ENDOPTIONS;
                        %END;


                %LET OPTIONS_EDIT = &OPTIONS ;
                %macro md_valid_parts_ (ValidString=,FlagVar=) / des="CALLED FROM
ML_CREATE_META";
                  %* Remove valid words, one at a time, storing into options_edit.
If, after all calls to the _valid_parts_      ;
                  %* macro, options_edit has any characters left, then those
characters must be invalid.                                          ;
                  %* When OPTIONS_TEMP is different from OPTIONS_EDIT, set the
corresponnding flag - for checking status          ;
                  %* across multiple valid parts, below.

                          ;
                  %LET OPTIONS_TEMP = &OPTIONS_EDIT ;
                  %LET OPTIONS_EDIT = %SYSFUNC ( TRANWRD ( &OPTIONS_EDIT ,
&ValidString , )) ;
                  %IF "&OPTIONS_EDIT" NE "&OPTIONS_TEMP" %THEN %LET &FlagVar = 1;
                %mend md_valid_parts_ ;
                %md_valid_parts_(ValidString=DESCENDING,FlagVar=OPTIONS_DESCENDING)
                %md_valid_parts_(ValidString=ID,FlagVar=OPTIONS_ID)
```

```
                %md_valid_parts_(ValidString=NOPRINT,FlagVar=OPTIONS_NOPRINT)
                %md_valid_parts_(ValidString=PAGE,FlagVar=OPTIONS_PAGE)
                %md_valid_parts_(ValidString=ORDER=DATA,FlagVar=OPTIONS_ORDERDATA)

%md_valid_parts_(ValidString=ORDER=INTERNAL,FlagVar=OPTIONS_ORDERINTERNAL)

%md_valid_parts_(ValidString=ORDER=FORMATTED,FlagVar=OPTIONS_ORDERFORMATTED)


                %IF "&OPTIONS_EDIT" NE ""
                        %THEN %DO;
                                %PUT %STR(ALERT_R: --- &sysmacroname var=&var
----------------------------------------------);
                                %PUT %STR(ALERT_R: The OPTIONS parameter contains
one or more invalid words: &OPTIONS_EDIT );
                                %PUT %STR(ALERT_R: OPTIONS will be set to null.);
                                %PUT %STR(ALERT_R:
------------------------------------------------------------------);
                                %LET ML_CREATE_META_RC = 2;
                                %LET OPTIONS = SETNULL;
                                %let options_descending = 0; %let options_id = 0;
%let options_noprint = 0;
                                %let options_page = 0 ; %let options_orderdata = 0;
%let options_orderinternal = 0;
                                %let options_orderformatted = 0;
                                %GOTO ENDOPTIONS ;
                        %END;


                %* IF DESCENDING was specified in &OPTIONS, and &USAGE is display
only by default, change USAGE to ORDER.;
                %IF ( "&OPTIONS_DESCENDING" = "1") AND ("&USAGE_DISPLAY_DEFAULT" =
"1")  AND ("&USAGE_BY" = "0")
                        %THEN %DO;
                                %PUT %STR(ALERT_I: --- &sysmacroname var=&var
--------------------------------------------);
                                %PUT %STR(ALERT_I: Since OPTIONS contains the
parameter DESCENDING, and USAGE was null,  );
                                %PUT %STR(ALERT_I: USAGE will be set to ORDER.);
                                %PUT %STR(ALERT_I:
------------------------------------------------------------------);
                                %LET USAGE = &USAGE ORDER   ;
                                %LET USAGE_ORDER = 1 ;
                        %END;

                %* IF DESCENDING was specified in &OPTIONS, then &USAGE should also
include ORDER.  If not there, notify user ;
                %* and append ORDER to &USAGE.

                        ;
```

```
                %IF ( "&OPTIONS_DESCENDING" = "1") AND ("&USAGE_ORDER" NE "1") AND
("&USAGE_BY" = "0")
                        %THEN %DO;
                                %PUT %STR(ALERT_I: --- &sysmacroname var=&var
-----------------------------------------------);
                                %PUT %STR(ALERT_I: Since OPTIONS contains the
parameter DESCENDING, USAGE must also);
                                %PUT %STR(ALERT_I: contain the option ORDER.);
                                %PUT %STR(ALERT_I: ORDER will be appended to
USAGE.);
                                %PUT %STR(ALERT_I:
-----------------------------------------------------------------------);
                                %LET USAGE = &USAGE  ORDER   ;
                                %LET USAGE_ORDER = 1 ;
                        %END;


%SKIPOPTIONS:
                %* IF &USAGE is ORDER and no ORDER= has been defined in &OPTIONS,
then append ORDER=INTERNAL to options ;
                %IF ("&USAGE_ORDER" = "1") AND %EVAL(&options_orderdata +
&options_orderinternal + &options_orderformatted ) EQ  0

                        %THEN %DO;
                                %PUT %STR(ALERT_I: --- &sysmacroname var=&var
-----------------------------------------------);
                                %PUT %STR(ALERT_I: Since USAGE contains the
parameter ORDER and no ORDER= was specified   );
                                %PUT %STR(ALERT_I: in OPTIONS, ORDER=INTERNAL will
be appended to OPTIONS.);
                                %PUT %STR(ALERT_I:
-----------------------------------------------------------------------);
                                %LET OPTIONS = &OPTIONS  ORDER=INTERNAL   ;
                                %LET OPTIONS_ORDERINTERNAL = 1 ;
                        %END;


                        %* Check if OPTIONS contains ID and NOPRINT - these would
be in conflict  ;
                        %IF "&OPTIONS_ID" EQ "1" AND  "&OPTIONS_NOPRINT" EQ "1"

                                                %THEN %DO;
                                                        %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ----------------------------------------);
                                                        %PUT %STR(ALERT_R: The
OPTIONS parameter contains ID and NOPRINT.);
                                                        %PUT %STR(ALERT_R: These
OPTIONS are in conflict with each other.);
                                                        %PUT %STR(ALERT_R: These
OPTIONS will be removed.);
```

```
                                                    %PUT %STR(ALERT_R:
----------------------------------------------------------------);
                                                    %LET ML_CREATE_META_RC = 2;
                                                    %let options =
%sysfunc(tranwrd(&options, ID,%str( ) ));
                                                    %let options =
%sysfunc(tranwrd(&options, NOPRINT,%str( ) ));
                                                    %let options_id = 0;
                                                    %let options_noprint = 0;

                                                    %GOTO ENDOPTIONS ;
                                            %END;

                        %* Check if OPTIONS contains ID and PAGE - these would be
in conflict  ;
                        %IF "&OPTIONS_ID" EQ "1" AND  "&OPTIONS_PAGE" EQ "1"

                                            %THEN %DO;
                                                    %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ------------------------------------------);
                                                    %PUT %STR(ALERT_R: The
OPTIONS parameter contains ID and PAGE.);
                                                    %PUT %STR(ALERT_R: These
OPTIONS are in conflict with each other.);
                                                    %PUT %STR(ALERT_R: These
OPTIONS will be removed.);
                                                    %PUT %STR(ALERT_R:
----------------------------------------------------------------);
                                                    %LET ML_CREATE_META_RC = 2;
                                                    %let options =
%sysfunc(tranwrd(&options, ID,%str( ) ));
                                                    %let options =
%sysfunc(tranwrd(&options, PAGE,%str( ) ));
                                                    %let options_id = 0;
                                                    %let options_page = 0 ;
                                                    %GOTO ENDOPTIONS ;
                                            %END;

                        %* Check if OPTIONS contains PAGE and NOPRINT - these would
be in conflict  ;
                        %IF "&OPTIONS_PAGE" EQ "1" AND  "&OPTIONS_NOPRINT" EQ "1"

                                            %THEN %DO;
                                                    %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ------------------------------------------);
                                                    %PUT %STR(ALERT_R: The
OPTIONS parameter contains PAGE and NOPRINT.);
                                                    %PUT %STR(ALERT_R: These
OPTIONS are in conflict with each other.);
                                                    %PUT %STR(ALERT_R: OPTIONS
```

```
will be set to null.);
                                                        %PUT %STR(ALERT_R:
----------------------------------------------------------------------);
                                                        %LET ML_CREATE_META_RC = 2;
                                                        %let options =
%sysfunc(tranwrd(&options, PAGE,%str( ) ));
                                                        %let options =
%sysfunc(tranwrd(&options, NOPRINT,%str( ) ));
                                                        %let options_noprint = 0;
                                                        %let options_page = 0 ;
                                                        %GOTO ENDOPTIONS ;
                                                  %END;



%ENDOPTIONS:
%* END of OPTIONS parameter checks  ;

%* Check FORMAT parameter.  FORMAT is not required.  There is no default.  R8. ;
        %* Use dictionary.formats to include SAS defined formats in the check.   ;

        %IF %STR(&FORMAT)  NE %STR( )
                %THEN %DO;
                        %LET FORMAT = %upcase(&FORMAT);
                        %* Edit the format string to remove . ;
                        %LET _temp_format = %SYSFUNC ( translate
(%STR(&FORMAT),%STR( ),%STR(.)  )) ;

                        %* Remove trailing numbers that qualify the length of the
format.  Any number after the last non-number
                                should be removed to match the name of the format
in dictionary.formats ;
                        %LET _temp_format = %SYSFUNC ( reverse (%STR(&_temp_format)
)) ;

                        %LET _first_non_digit = %SYSFUNC ( notdigit
(%STR(&_temp_format) ));
                        %if &_first_non_digit ne 0 %then %do;
                          %LET _temp_format =
%SUBSTR(%STR(&_temp_format),&_first_non_digit);
                        %end;
                        %LET _temp_format = %SYSFUNC ( reverse (%STR(&_temp_format)
)) ;

                        proc sql noprint;
                                create table allformats as
                                select *
                                from dictionary.formats
                                        where upcase(fmtname) = "&_temp_format";
```

```
                        quit;

                        %IF &sqlobs EQ 0 and %verify(&format, 0123456789.) ne 0

                                %THEN %DO;
                                                %PUT %STR(ALERT_)R: ---
&sysmacroname var=&var ----------------------------------------------;
                                                %PUT %STR(ALERT_)R: The FORMAT
[&_temp_format.] from the FORMAT parameter value ;
                                               ;
                                                %PUT %STR(ALERT_)R: [&format.] does
not exist.  Format will be set to null.;
                                                %PUT %STR(ALERT_)R:
-------------------------------------------------------------------;
                                                %LET ML_CREATE_META_RC = 3;
                                                %LET FORMAT = SETNULL;
                                                %GOTO ENDFORMAT;
                                %END;

                        %* Check that a numeric format is being used for a numeric
var, or a character format is being used ;
                        %* for a character var.

                                ;
                        %* In case IN_DATA is 2 level name, translate period to _
to work with mu_var_attributes macro vars ;
                        %LET IN_Data_Level2 = %SYSFUNC ( translate
(%STR(&IN_DATA),%STR(_),%STR(.)  ));
                        %mu_var_attributes (datasets = &In_Data , variables = &VAR
, debug=n)
                        %LET VarType = &&&In_Data_Level2._&VAR._VARTYP;


                        %if "&VARTYPE" = "C"
                                %THEN %DO;
                                %IF %SUBSTR(&FORMAT,1,1) NE $
                                        %THEN %DO;
                                                %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ----------------------------------------------);
                                                %PUT %STR(ALERT_R: The variable
[&VAR.] is character, but the format [&FORMAT.] is not.);
                                                %PUT %STR(ALERT_R: Format will be
set to null.);
                                                %PUT %STR(ALERT_R:
-------------------------------------------------------------------);
                                                %LET ML_CREATE_META_RC = 4;
                                                %LET FORMAT = SETNULL;
                                                %GOTO ENDFORMAT;

                                        %END;
```

```
                                %END;
                        %if "&VARTYPE" = "N"
                                %THEN %DO;
                                %IF %SUBSTR(&FORMAT,1,1) EQ  $
                                        %THEN %DO;
                                                %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ----------------------------------------------);
                                                %PUT %STR(ALERT_R: The variable
[&VAR.] is numeric, but the format [&FORMAT.] is not.   );
                                                %PUT %STR(ALERT_R: Format will be
set to null.);
                                                %PUT %STR(ALERT_R:
-------------------------------------------------------------------);
                                                %LET ML_CREATE_META_RC = 4;
                                                %LET FORMAT = SETNULL;
                                                %GOTO ENDFORMAT;

                                        %END;
                                %END;
                %END;
%ENDFORMAT:
%* END of FORMAT parameter checks    R8 ;

%* Check UNIT parameter.  UNIT shoud start with a P or C.  Default is P for
percent.  R10.  ;
        %* R10, UNIT code is before R9 so UNIT will be set for R9 ;
                        %IF %STR(&UNIT)  = %STR( ) %THEN %LET UNIT = P;
                        %LET Unit = %upcase(%substr(&UNIT, 1,1)) ;
                        %if ("&UNIT" NE "P" AND "&UNIT" NE "C")
                                %THEN %DO;
                        %PUT %STR(ALERT_I: --- &sysmacroname var=&var
-----------------------------------------------);
                                        %PUT %STR(ALERT_I: The parameter UNIT does
not start with P for Percent or C for Character.);
                                        %PUT %STR(ALERT_I: It is [&UNIT.]);
                    %PUT %STR(ALERT_I: UNIT will default to P for percent.);
                    %PUT %STR(ALERT_I:
-------------------------------------------------------------------);
                                %END;
%* END of UNIT parameter check     R10  ;

%* Check WIDTH parameter.  WIDTH is not required.  Default is null.   R9. ;
                        %* Check that WIDTH is numeric. ;
                        %IF %STR(&WIDTH)  = %STR( ) %THEN %GOTO ENDWIDTH;

                        %IF (%datatyp(&WIDTH) NE NUMERIC)
                                        %THEN %DO;
                                                %PUT %STR(ALERT_R: ---
&sysmacroname var=&var ----------------------------------------------);
                                                %PUT %STR(ALERT_R: The parameter
```

```
WIDTH is not numeric.  WIDTH = [&WIDTH.]);
                                                %PUT %STR(ALERT_R: WIDTH will be
set to missing.);
                                                %PUT %STR(ALERT_R:
----------------------------------------------------------------------);
                                                %LET ML_CREATE_META_RC = 5;
                                                %LET WIDTH =  ;
                                                %GOTO ENDWIDTH;
                                 %END;
                             %* IF UNIT is C (character) make sure that (numeric) WIDTH
is a whole number. ;
                             %IF "&UNIT" = "C" AND  %EVAL( %SYSFUNC( CEIL (&WIDTH )) NE
&WIDTH )
                                           %THEN %DO;
                                                %PUT %STR(ALERT_I: ---
&sysmacroname var=&var ----------------------------------------------);
                                                %PUT %STR(ALERT_I: UNIT is C
(character), but WIDTH is not a whole number. WIDTH = [&WIDTH.]);
                                                %PUT %STR(ALERT_I: WIDTH will be
rounded up to %SYSFUNC( CEIL (&WIDTH )));
                                                %PUT %STR(ALERT_I:
----------------------------------------------------------------------);
                                                %LET WIDTH = %SYSFUNC( CEIL (&WIDTH
));
                                                %GOTO ENDWIDTH;
                                 %END;
%ENDWIDTH:
%* END of WIDTH parameter check    R9  ;

%* R10, UNIT  appears before R9 so UNIT is set for R9 ;

%* R11, PRCODE_SPLIT is covered in set-up defaults, above.  ;

%* R12, ESCAPECHAR is covered in set-up defaults, above.   ;

%* R13, Label parameter ;
%* Check LABEL parameter.  LABEL is not required.  If LABEL = BLANK (not
case-sensitive) then set LABEL to null.    ;
%* If label is not specified, it will inherit the label from &VAR in &IN_DATA.
    R13.                                                          ;
%*  (but if the label of &VAR in &IN_DATA is null, set Label to &VAR (name of
variable).                                                          ;
        %put var=&var label1 = &label;
        %LET TEMP_LABEL = %superq(LABEL);
        %IF %bquote(&TEMP_LABEL) EQ BLANK %THEN %DO;
            %LET LABEL = SETNULL;
            %GOTO ENDLABEL ;
        %END;
        %put >> boolean = %sysevalf(%superq(LABEL)=,boolean);
        %IF %sysevalf(%superq(LABEL)=,boolean) ne 0  %THEN %DO;
```

```
        %put label is missing - getting label;
            %* Set Label to the label attribute of &VAR in &IN_DATA.  If that is
null set label to &VAR ;
            %LET IN_Data_Level2 = %SYSFUNC ( translate
(%STR(&IN_DATA),%STR(_),%STR(.)  ));
            %mu_var_attributes (datasets = &In_Data , variables = &VAR , debug=n) ;

            %LET Label = &&&In_Data_Level2._&VAR._LBL;
            *** ADDED *** ;  %let label=%superq(label) ;
            %put label2 = &label;
            %put >> boolean = %sysevalf(%superq(LABEL)=,boolean);
            %IF %sysevalf(%superq(LABEL)=,boolean) ne 0  %THEN %LET LABEL = %STR
(&VAR );
            %put label3 = &label;

        %END;

        %LET LABEL4 = %BQUOTE(&LABEL);
        %put >>>> VAR=&VAR LABEL=&LABEL;
%ENDLABEL:
%* END of 1st part LABEL parameter checks  ;

        %let label2=%sysfunc( tranwrd(%superq(label), %str(%'), K) ) ;

        %md_max_component_length(string_to_check = %bquote(&label2.)  ,

max_component_length_mvar=MAX_LABEL_LENGTH,
                                                        string_delimiters =
%STR(&prcode_split) %STR(&escapechar.n) %STR(&escapechar.\line)

            %STR(&escapechar.\par)   );
                %if "&Max_Label_Length" = "" %THEN %LET Max_Label_Length = 0 ;

        %* If width is not null and label is not null check length of label versus
width. ;
        %IF "&WIDTH" NE "" AND "&LABEL" NE ""
        %THEN %DO;

        %MD_Calc_Width (MD_Col_Width = &Width,
                                        MD_Col_Width_Unit = &UNIT,
                                        MD_Page_Width_Chars = &Page_Width_Chars ) ;

                %IF &MAX_LABEL_LENGTH GT &MD_Out_Var_Width_Chars
                %then %do;
                        %PUT %STR(ALERT_I: --- &sysmacroname var=&var
------------------------------------------------------------);
                        %PUT %STR(ALERT_I: The maximum length of a LABEL segment
(appearing on 1 line) exceeds WIDTH.);
                        %if %STR(&UNIT) = %STR(C) %then %do;
                                %PUT %STR(ALERT_I: WIDTH was %left(&WIDTH) (in
```

```
characters), but maximum length of LABEL segment    );
                                %PUT %STR(ALERT_I:  is %left(&MAX_LABEL_LENGTH).);
                                %PUT %STR(ALERT_I:  Consider inserting a split
character or escape character in the label.);
                                %PUT %STR(ALERT_I:
--------------------------------------------------------------------------------);
                                %LET ML_CREATE_META_RC = 6;
                        %end;
                        %if %STR(&UNIT) = %STR(P) %then %do;
                                %PUT %STR(ALERT_I: WIDTH was %left(&WIDTH) (in
percent).  Given the Page_Width_Chars of %left(&Page_Width_Chars),);
                                %PUT %STR(ALERT_I:  this is equal to
%left(&MD_Out_Var_Width_Chars) (in characters).);
                                %PUT %STR(ALERT_I:  Consider inserting a split
character or escape character in the label.);
                                %PUT %STR(ALERT_I:
--------------------------------------------------------------------------------);
                                %LET ML_CREATE_META_RC = 6;
                        %end;
                %end;
        %END;
%* END of 2nd part LABEL parameter checks    R13.  ;

%* R14, SPANNING_HEADER - no code required at this time. ;

%* R15, STYLE_HEADER.  No version of the word WIDTH can be specified here (instead
use WIDTH parameter).
        Do not allow brackets.   ;
        %IF %BQUOTE(&STYLE_HEADER)  EQ %STR( ) %THEN %GOTO ENDSTYLEHEADER;
                %ELSE %DO;
                        %LET STYLE_TEMP = %upcase(&STYLE_HEADER);
                        %IF %index(&STYLE_TEMP,WIDTH) GT 0
                        %THEN %DO;

                                %PUT %STR(ALERT_C: --- &sysmacroname var=&var
-----------------------------------------------------) ;
                                %PUT %STR(ALERT_C: The STYLE_HEADER parameter
contains the word WIDTH.  Neither CELLWIDTH nor   ) ;
                                %PUT %STR(ALERT_C:  WIDTH is allowed in
STYLE_HEADER.  Instead use the WIDTH parameter.) ;
                                %PUT %STR(ALERT_C: STYLE_HEADER will be set to
null.) ;
                                %PUT %STR(ALERT_C:
-------------------------------------------------------------------------) ;
                                %LET STYLE_HEADER = SETNULL;
                                %GOTO ENDSTYLEHEADER;
                        %END;
                        %IF  %sysfunc( indexc(&STYLE_TEMP,[]{}) ) GT 0
                        %THEN %DO;
```

```
                                %PUT %STR(ALERT_C: --- &sysmacroname var=&var
----------------------------------------------------) ;
                                %PUT %STR(ALERT_C: The STYLE_HEADER parameter
contains brackets.  Brackets are inserted in the  ) ;
                                %PUT %STR(ALERT_C:  reporting step.  Please remove
the brackets, either [] or {} ) ;
                                %PUT %STR(ALERT_C: STYLE_HEADER will be set to
null.) ;
                                %PUT %STR(ALERT_C:
----------------------------------------------------------------------------) ;
                                %LET STYLE_HEADER = SETNULL;
                        %END;
                %END;
%ENDSTYLEHEADER:
%* END of STYLE_HEADER parameter checks  ;

%* R16, STYLE_COLUMN.  No version of the word WIDTH can be specified here (instead
use the WIDTH parameter).
        Do not allow brackets.  ;
        %IF %BQUOTE(&STYLE_COLUMN)  EQ  %STR( ) %THEN %GOTO ENDSTYLECOLUMN;
                %ELSE %DO;
                        %LET STYLE_TEMP = %upcase(&STYLE_COLUMN);

                        %IF %index(&STYLE_TEMP,WIDTH) GT 0
                        %THEN %DO;

                                %PUT %STR(ALERT_C: --- &sysmacroname var=&var
----------------------------------------------------) ;
                                %PUT %STR(ALERT_C: The STYLE_COLUMN parameter
contains the word WIDTH.  Neither CELLWIDTH nor   ) ;
                                %PUT %STR(ALERT_C:  WIDTH is allowed in
STYLE_COLUMN.  Instead use the WIDTH parameter.) ;
                                %PUT %STR(ALERT_C: STYLE_COLUMN will be set to
null.) ;
                                %PUT %STR(ALERT_C:
----------------------------------------------------------------------------) ;
                                %LET STYLE_COLUMN = SETNULL;
                                %GOTO ENDSTYLECOLUMN;
                        %END;
                        %IF  %sysfunc( indexc(&STYLE_TEMP,[]{}) ) GT 0
                        %THEN %DO;

                                %PUT %STR(ALERT_C: --- &sysmacroname var=&var
----------------------------------------------------) ;
                                %PUT %STR(ALERT_C: The STYLE_COLUMN parameter
contains brackets.  Brackets are inserted in the  ) ;
                                %PUT %STR(ALERT_C:  reporting step.  Please remove
the brackets, either [] or {}) ;
                                %PUT %STR(ALERT_C: STYLE_COLUMN will be set to
null.) ;
```

```
                                %PUT %STR(ALERT_C:
----------------------------------------------------------------------------) ;
                                %LET STYLE_COLUMN = SETNULL;
                        %END;
                %END;
%ENDSTYLECOLUMN:
%* END of STYLE_COLUMN parameter checks  ;

%* Check PACKED parameter.  PACKED is not required.  If not  Y,y,N or n default to
N.  R17. ;
        %IF %BQUOTE(&PACKED)  = %STR( ) %THEN %LET PACKED = N;
        %LET PACKED = %upcase(%substr(&PACKED, 1,1)) ;
                        %if ("&PACKED" NE "Y" AND "&PACKED" NE "N")
                                %THEN %DO;
                        %PUT %STR(ALERT_I: --- &sysmacroname var=&var
-------------------------------------------------);
                                        %PUT %STR(ALERT_I: The parameter PACKED
does not start with Y or N. It is [&PACKED.]);
                    %PUT %STR(ALERT_I: PACKED will default to N (No).);
                    %PUT %STR(ALERT_I:
-----------------------------------------------------------------------);
                                        %LET PACKED = N ;
                                %END;
%* END of PACKED parameter check    R17  ;

%* R18, Page_Width_Chars is covered in set-up defaults, above.  ;

%**** END of PARAMETER CHECKS;


%**** save meta_data ;
        %* Note: The variable SKIP is saved in the Meta_Data.  That value comes
from the x in SKIP=x in USAGE.  That was
                saved in zzz above.    ;

        %IF %BQUOTE(&Width) = %STR( ) %THEN %LET Width = . ;

        %* Create the dataset _OneObs_ capturing the values in this call, as
modified above.    ;
                Data _OneObs_ ;
                  length IN_DATA $ 32  VarNum 8  Var $ 32  Usage Options  $ 200
Format $ 15  Width 8  Unit $ 1
                                    Label $ 256  Label_Width 8  Spanning_Header
Style_Header Style_Column $ 200  Packed $ 1
                                    SKIP 8   PrCode_Split EscapeChar $ 1
Page_Width_Chars 8 ;
                        IN_Data = "&IN_Data";
                        VarNum =  . ;
                        Var = "&Var";
                        Usage = "&Usage";
```

```
                        Options = "&Options";
                        Format = "&Format";
                        Width = &Width ;
                        Unit = "&Unit";
                        Label = "&Label";
                        Label_Width = &Max_Label_Length ;
                        Spanning_Header = "&Spanning_Header";
                        Style_Header = "&Style_Header";
                        Style_Column = "&Style_Column";
                        Packed = "&Packed";
                        Skip   = &ML_SKIP_VALUE ;
                        PrCode_Split = "&PrCode_Split";
                        EscapeChar   = "&EscapeChar";
                        Page_Width_Chars = &Page_Width_Chars;
                run;

%GLOBAL &META_DATA_Level2._NOBS ;
%LET &META_DATA_Level2._NOBS = 0;

%IF %str(&ML_Create_Meta_First_Call) NE %str(Y) %THEN %mu_nobs (In_Data =
&Meta_Data ) ;

        %* If first call, create &Meta_Data from _OneObs_ and set VARNUM to 1 ;
        %IF %str(&ML_Create_Meta_First_Call) = %str(Y)
                %THEN %DO;
                                Data &Meta_Data ;
                                        set _OneObs_ ;
                                        VarNum = 1 ;
                                        %IF "&Usage"    = "SETNULL" %THEN %DO;
USAGE   = ' '; %END;
                                        %IF "&OPTIONS"  = "SETNULL" %THEN %DO;
OPTIONS = ' '; %END;
                                        %IF "&FORMAT"   = "SETNULL" %THEN %DO;
FORMAT  = ' '; %END;
                                        %IF "&WIDTH"    = "."           %THEN %DO;
WIDTH   =  . ; %END;
                                        %IF "&LABEL"    = "SETNULL" %THEN %DO;
LABEL   = ' '; %END;

                                        %IF "&SPANNING_HEADER" = "SETNULL" %THEN
%DO; SPANNING_HEADER = ' '; %END;
                                        %IF "&Style_Header"    = "SETNULL" %THEN
%DO; Style_Header    = ' '; %END;
                                        %IF "&STYLE_COLUMN"    = "SETNULL" %THEN
%DO; Style_Column    = ' '; %END;
                                run;
                        %END;
                %* If not first call, &MU_NOBS_RC will be 0 and
&&&META_DATA_Level2._NOBS will be number of obs in &META_DATA. ;
                %* Update &Meta_Data with _OneObs_ by VAR.  If VARNUM is missing,
```

```
set to &&&META_DATA_Level2._NOBS + 1 ;
                %ELSE %DO;

                        %IF %BQUOTE(&Usage)     = %STR( ) %THEN %LET USAGE   =
SETNULL;
                        %IF %BQUOTE(&OPTIONS)   = %STR( ) %THEN %LET OPTIONS =
SETNULL;
                        %IF %BQUOTE(&FORMAT)    = %STR( ) %THEN %LET FORMAT  =
SETNULL;
                        %IF %BQUOTE(&LABEL)     = %STR( ) %THEN %LET LABEL   =
SETNULL;
                        %IF %BQUOTE(&SPANNING_HEADER) = %STR( ) %THEN %LET
SPANNING_HEADER = SETNULL;
                        %IF %BQUOTE(&STYLE_HEADER) = %STR( ) %THEN %LET
STYLE_HEADER = SETNULL;
                        %IF %BQUOTE(&STYLE_COLUMN) = %STR( ) %THEN %LET
STYLE_COLUMN = SETNULL;

                                Proc Sort Data = &Meta_Data ;
                                        by VAR;
                                Data &Meta_Data ;
                                        update &Meta_Data   _OneObs_  (in=in_one);
                                        by VAR ;
                                        if VarNum = . then VarNum =
&&&META_DATA_Level2._NOBS + 1 ;

                                        if in_one then do;
                                        %IF "&Usage"    = "SETNULL" %THEN %DO;
USAGE   = ' '; %END;
                                        %IF "&OPTIONS"  = "SETNULL" %THEN %DO;
OPTIONS = ' '; %END;
                                        %IF "&FORMAT"   = "SETNULL" %THEN %DO;
FORMAT  = ' '; %END;
                                        %IF "&WIDTH"    = "."           %THEN %DO;
WIDTH   =  . ; %END;
                                        %IF "&LABEL"    = "SETNULL" %THEN %DO;
LABEL   = ' '; %END;

                                        %IF "&SPANNING_HEADER" = "SETNULL" %THEN
%DO; SPANNING_HEADER = ' '; %END;
                                        %IF "&Style_Header"    = "SETNULL" %THEN
%DO; Style_Header    = ' '; %END;
                                        %IF "&STYLE_COLUMN"    = "SETNULL" %THEN
%DO; Style_Column    = ' '; %END;
                                        end;
                                run;
                        %END;


                Proc Sort Data = &Meta_Data ;
```

```
                        by VARNUM;
                run;

%EXIT:

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

options &mprint_setting;

%if %upcase(%substr(&abort, 1,1)) = Y  %then %do;
  %if &ML_Create_Meta_First_Call = Y %then %do;
        %put ALERT_R:  The first call to &sysmacroname was unsuccessfull. ;
        %ml_create_meta_reset *;
  %end;
  %ABORT ;

%end;

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

%mend ML_CREATE_META;
```