```
%* ********************************************************************
%*       PROGRAM NAME:  GET_TF.sas
%*        SAS VERSION:  9.3, 9.4
%*            PURPOSE:  To create Title and Footnote statements or a TF
%*                      data set from an ASCII metadata file for RTF
%*                      output.
%*        USAGE NOTES:
%GET_TF(METADATA_FILEPATH=&g_projectpath.&g_toplevel.\Tools\TF_EXTRACT.TF,
%*                                  This is the path and filename of your
%*                                  metadata file
%*                               TLF_PROGNAME=&g_pgmname.,
%*                                  This is the name of the program to add
%*                                  the TFs to as indicated in the
%*                                  TF_EXTRACT.TF file.
%*                                                                  If value is
ALL and OUT_TF_DATASET=NO then it will output TFs for all
%*                                                                  output TFs
for all entries in the metadata file
%*                               WRITE_TITLES=Yes,
%*                                  This indicates whether or not Title
%*                                  statements should be written
%*                               WRITE_FOOTNOTES=Yes,
%*                                  This indicates whether or not Footnote
%*                                  statements should be written
%*                               OUT_TF_DATASET=Yes,
%*                                  This indicates whether or not you want
%*                                  an output data set
%*                               ESCAPECHAR=^,
%*                                  This is the ODS escape character you
%*                                  want to use for the TF statements
%*                               USE_PARSE_CHAR=Yes,
%*                                  This is the option to add 'a0'x at
%*                                  the end of a TF to pass to a
%*                                  del_substr so that the TF is removed
%*                                  from the output automatically
%*                               BLANK_LAST_TITLE=No,
%*                                  This indicates whether you want to put
%*                                  a blank title as the last title
%*                               BLANK_FIRST_FOOTNOTE=No,
%*                                  This indicates whether you want to put
%*                                  a blank footnote as the first footnote
%*                                                        New in V2:
%*                                                  FOOTNOTE_OVERLINE=Yes,
%*                                                                  Option to
add top border over the first footnote
%*
FOOTNOTE_HANGING_INDENT=Yes,
%*                                                                  Option to
apply hanging idents to footnotes triggered
%*                                                                  by a set of
```

```
phrases commonly found at the beggining of
%*                                                                    the string
%*                                                        FONT_SIZE=9,
%*                                                                    Required
parameter when using FOOTNOTE_HANGING_INDENT=Yes to determine
%*                                                                    the number
of twips to shift the text. Value
%*                                                                    should
match the font size of the output in the TLF.
%*                                                    GLOBALTFS=1,
%*                                                                    Specify
which set of global TFs to use
%*                                                        FPAGE=No,
%*                                                                    Option to
generate a footnote page
%*
FPAGE_IGNORE_LINES=1,
%*                                                                    Valid with
FPAGE=Yes, specify how many of the last remaining
%*                                                                    footnotes
to exclude from the footnote page and keep as bona
%*                                                                    fide
footnotes
%*
FPAGE_FIRST_FOOTNOTE=
%*                                                                    Text that
will as the first bona fide footnote when FPAGE=Yes
%*
USE_PROJECT_DEFAULTS=Yes
%*                                                                    Option to
use the project level defaults for get_tf parameter
%*                                                                    Use for
parameter values only if the differ from the default
%*                                                                    set in the
macro definition
%*                                                        COMBINE_TFS=Yes
%*                                                                     Option to
attempt to combine adjacent TFs when there are more
%*                                                                     then 10
%*
%*
%*        INPUT FILES: &metadata_filepath
%*       OUTPUT FILES: &tlf_progname._tf (only to the work library)
%*
%*************************************************************************
%*  © 2018 PPD
%*  All Rights Reserved.
%*************************************************************************;

%macro get_tf
```

```
  ( metadata_filepath=&g_projectpath.&g_toplevel.\Tools\TF_EXTRACT.TF
  , tlf_progname=&g_pgmname.
  , write_titles=Yes
  , write_footnotes=Yes
  , out_tf_dataset=Yes
  , escapechar=^
  , use_parse_char=Yes
  , blank_last_title=No
  , blank_first_footnote=No
  , footnote_overline=Yes
  , footnote_overline_width=10
  , footnote_hanging_indent=Yes
  , font_size=9
  , globaltfs=1
  , combine_tfs=Yes
  , fpage=No
  , fpage_ignore_lines=1
  , fpage_first_footnote=
  , fpage_suppress=Yes
  , use_project_defaults=Yes
  , debug=No
   ) /  store source des='V2.2.0.1'
   ;

%* UPCASE keyword parameters ;
%let tlf_progname=%upcase(&tlf_progname.);
%let write_titles=%upcase(&write_titles.);
%let write_footnotes=%upcase(&write_footnotes.);
%let out_tf_dataset=%upcase(&out_tf_dataset.);
%let use_parse_char=%upcase(&use_parse_char.);
%let blank_last_title=%upcase(&blank_last_title.);
%let blank_first_footnote=%upcase(&blank_first_footnote.);
%let footnote_overline=%upcase(&footnote_overline.);
%let footnote_hanging_indent=%upcase(&footnote_hanging_indent.);
%let globaltfs=%upcase(&globaltfs.);
%let combine_tfs=%upcase(&combine_tfs.);
%let fpage=%upcase(&fpage.);
%let fpage_suppress=%upcase(&fpage_suppress.);
%let use_project_defaults=%upcase(&use_project_defaults.);
%let debug=%upcase(&debug.);


 %* Identify version and entrance into the macro ;
 %local Version;
 %let Version = 2.2 ;
 %put NOTE: Entering &sysmacroname v&Version macro.;

 %* Capture value of macro options ;
 %local mprint mlogic symbolgen ;
 %let mprint=%sysfunc(getoption(mprint));
```

```
%let mlogic=%sysfunc(getoption(mlogic));
%let symbolgen=%sysfunc(getoption(symbolgen));
option nomprint nomlogic nosymbolgen ;
%if &debug.=YES or &debug.=1 or &debug.=2 or &debug.=3 %then %do ;
       option mprint ;
%end ;
%if &debug.=YES or &debug.=2 or &debug.=3 %then %do ;
       option mlogic ;
%end ;
%if &debug.=YES or &debug.=3 %then %do ;
       option symbolgen ;
%end ;

%* Capture value of QUOTELENMAX option in SAS V9 ;
%local defaultquoteoption;
%let defaultquoteoption=%sysfunc(getoption(quotelenmax));
%* Temporarily turn off the max quote length warning message ;
options NOQUOTELENMAX;


%* Set global macro var with error checking status ;
%global get_tf_rc;
%let get_tf_rc=0;

%* Declare other global macro vars ;
%global title_count footnote_count;

%* Load any project-level macro parameter settings ;
%if &use_project_defaults.=YES %then %do ;
  data proj_defaults ;
   infile "&metadata_filepath." delimiter='bb'x missover dsd lrecl=1500 ;
   attrib
        _name  length=$6       label='Macro Name'
    _param   length=$40    label='Macro Parameter Name'
    _value   length=$100   label='Project Level Parameter Value'
    ;
   input _name $ _param $ _value $ ;
       if upcase(_name)='GET_TF' ;
       __value=trim(left(upcase(_value))) ;
       call symput( _param, trim(left(__value))  ) ;
       _paramU=upcase(_param) ;
       put "%str(N)OTE-" _paramU " parameter set to " __value " in .tf file." ;
       run ;
  %end ;
  %else %if &use_project_defaults. ne NO %then %do ;
     %put ALERT_P: (&sysmacroname) Valid values for the USE_PROJECT_DEFAULTS
parameter are Yes or No;
         %put ALERT_P: (&sysmacroname) Please use a valid value and try again.
Processing will be stopped!;
         %goto exit ;
```

```
   %end ;

 %*** Check supported SAS versions;
 %if &sysver. ne 9.3 and &sysver. ne 9.4  %then
  %put ALERT_R: (&sysmacroname) This macro has not been tested using this version
of SAS.  Please proceed with caution.;

 %*** Check the parameter values ***;
 %if %sysfunc(fileexist(&metadata_filepath)) = 0 %then %do;
  %put ALERT_P: (&sysmacroname) Metadata file does not exist as specified
"&Metadata_filepath".;
  %put ALERT_P: (&sysmacroname) Please modify the METADATA_FILEPATH parameter to a
valid path and filename. Processing will be stopped!;
 %let get_tf_rc=1;
 %end;

 %if &write_titles. ne YES and &write_titles. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the WRITE_TITLES parameter are Yes
or No. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for WRITE_TITLES and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;
 %if &write_titles. eq YES and &tlf_progname.=ALL %then %do;
  %put ALERT_P: (&sysmacroname) The WRITE_TITLES parameter must be NO when
TLF_PROGNAME=ALL. Please...;
  %put ALERT_P: (&sysmacroname) ...update the value for WRITE_TITLES and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;


 %if &write_footnotes. ne YES and &write_footnotes. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the WRITE_FOOTNOTES parameter are
Yes or No. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for WRITE_FOOTNOTES and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;
 %if &write_footnotes. eq YES and &tlf_progname.=ALL %then %do;
  %put ALERT_P: (&sysmacroname) The WRITE_FOOTNOTES parameter must be NO when
TLF_PROGNAME=ALL. Please...;
  %put ALERT_P: (&sysmacroname) ...update the value for WRITE_FOOTNOTES and try
again. Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %if &out_tf_dataset. ne YES and &out_tf_dataset. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the OUT_TF_DATASET parameter are
Yes or No. Please use...;
```

```
  %put ALERT_P: (&sysmacroname) ...a valid value for OUT_TF_DATASET and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;
 %if &out_tf_dataset. eq NO and &tlf_progname.=ALL %then %do;
  %put ALERT_P: (&sysmacroname) The OUT_TF_DATASET parameter must be Yes when
TLF_PROGNAME=ALL. Please...;
  %put ALERT_P: (&sysmacroname) ...update the value for OUT_TF_DATASET and try
again. Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %if %length(%trim(&escapechar))>1 %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the ESCAPECHAR parameter are a
single character only. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for ESCAPECHAR and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %if &use_parse_char. ne YES and &use_parse_char. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the USE_PARSE_CHAR parameter are
Yes or No. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for USE_PARSE_CHAR and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %if &blank_last_title. ne YES and &blank_last_title. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the BLANK_LAST_TITLE parameter are
Yes or No. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for BLANK_LAST_TITLE and try
again. Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %if &blank_first_footnote. ne YES and &blank_first_footnote. ne NO %then %do;
  %put ALERT_P: (&sysmacroname) Valid values for the BLANK_FIRST_FOOTNOTE parameter
are Yes or No. Please use...;
  %put ALERT_P: (&sysmacroname) ...a valid value for BLANK_FIRST_FOOTNOTE and try
again. Processing will be stopped!;
  %let get_tf_rc=1;
 %end;

 %* Checks for GLOBALTFS parameter ;
 %local re1 ;
 %let re1=%sysfunc( prxparse(/[^0-9]/) ) ; %* Will use this for FPAGE parameter
check, too ;
 %if &globaltfs. eq YES %then %do;
    %let globaltfs=1;
```

```
  %end;
  %else %if &globaltfs. eq NO %then %do;
    %let globaltfs=0;
  %end;
  %else %if %sysfunc( prxmatch(&re1.,%bquote(&globaltfs.)) ) gt 0 %then %do;
    %put ALERT_P: (&sysmacroname) Valid values for the GLOBALTFS parameter are Yes,
No or a postive integer.;
    %put ALERT_P: (&sysmacroname) Please use a valid value for GLOBALTFS and try
again. Processing will be stopped!;
    %let get_tf_rc=1;
  %end ;

 %if &globaltfs. ne 0 and &tlf_progname.=ALL %then %do;
  %put ALERT_P: (&sysmacroname) The GLOBALTFS parameter must be No or 0 when
TLF_PROGNAME=ALL. Please...;
  %put ALERT_P: (&sysmacroname) ...update the value for GLOBALTFS and try again.
Processing will be stopped!;
  %let get_tf_rc=1;
 %end;


  %if &footnote_overline. ne YES and &footnote_overline. ne NO %then %do;
   %put ALERT_P: (&sysmacroname) Valid values for the FOOTNOTE_OVERLINE parameter
are Yes or No. Please use...;
   %put ALERT_P: (&sysmacroname) ...a valid value for FOOTNOTE_OVERLINE and try
again. Processing will be stopped!;
   %let get_tf_rc=1;
  %end;

  %if &footnote_hanging_indent. ne YES and &footnote_hanging_indent. ne NO %then
%do;
   %put ALERT_P: (&sysmacroname) Valid values for the FOOTNOTE_HANGING_INDENT
parameter are Yes or No. Please use...;
   %put ALERT_P: (&sysmacroname) ...a valid value for FOOTNOTE_HANGING_INDENT and
try again. Processing will be stopped!;
   %let get_tf_rc=1;
  %end;

 %if &footnote_hanging_indent. eq YES and %length(&font_size.)=0 %then %do;
    %put ALERT_P: (&sysmacroname) FONT_SIZE value can not be missing when
FOOTNOTE_HANGING_INDENT parameter is Yes.;
    %put ALERT_P: (&sysmacroname) Use an integer between 5 and 14 for FONT_SIZE and
try again. Processing will be stopped!;
    %let get_tf_rc=1;
  %end;
  %else %if &footnote_hanging_indent. eq YES and %index(7 8 9 10 11
12,&font_size.)=0 %then %do;
      %put ALERT_P: (&sysmacroname) FONT_SIZE value was invalid! Please use an
integer between 7 and 12 ...;
      %put ALERT_P: (&sysmacroname) for FONT_SIZE and try again. Processing will be
```

```
stopped!;
      %let get_tf_rc=1;
  %end;

  %* FPAGE checks ;
  %if &fpage. ne YES and &fpage. ne NO %then %do ;
   %put ALERT_P: (&sysmacroname) Valid values for the FPAGE parameter are Yes or
No. Please use...;
   %put ALERT_P: (&sysmacroname) ...a valid value for FPAGE and try again.
Processing will be stopped!;
   %let get_tf_rc=1;
  %end ;

  %if &fpage.=YES %then %do ;
    %if &fpage_ignore_lines eq %str() %then %do ;
      %put ALERT_P: (&sysmacroname) Valid value for the FPAGE_IGNORE_LINES
parameter is a postive integer.;
      %put ALERT_P: (&sysmacroname) Please use a valid value for FPAGE_IGNORE_LINES
and try again. Processing will be stopped!;
      %let get_tf_rc=1;
    %end ;
    %else %if %sysfunc( prxmatch(&re1.,%bquote(&fpage_ignore_lines.)) ) gt 0 %then
%do ;
          %put ALERT_P: (&sysmacroname) Valid value for the FPAGE_IGNORE_LINES
parameter is a postive integer.;
          %put ALERT_P: (&sysmacroname) Please use a valid value for
FPAGE_IGNORE_LINES and try again. Processing will be stopped!;
          %let get_tf_rc=1;
    %end ;
  %end ;

  %if &fpage_suppress. ne YES and &fpage_suppress. ne NO %then %do ;
   %put ALERT_P: (&sysmacroname) Valid values for the FPAGE_SUPPRESS parameter are
Yes or No. Please use...;
   %put ALERT_P: (&sysmacroname) ...a valid value for FPAGE_SUPPRESS and try again.
Processing will be stopped!;
   %let get_tf_rc=1;
  %end ;

  %* COMBINE_TFS check ;
  %if &combine_tfs. ne YES and &combine_tfs. ne NO %then %do;
   %put ALERT_P: (&sysmacroname) Valid values for the COMBINE_TFS parameter are Yes
or No. Please use...;
   %put ALERT_P: (&sysmacroname) ...a valid value for COMBINE_TFS and try again.
Processing will be stopped!;
   %let get_tf_rc=1;
  %end;

 %* Jump to bottom of macro if there are failed parameter checks ;
 %if &get_tf_rc.=1 %then %goto exit ;
```

```
ods escapechar="&escapechar";


%* Read in the document properties from metadata file defined in METADATA_FILEPATH

   parmeter ;
%local last_dt_tm
       qc_dt_tm
       max_progid_lg
       max_torfseq_lg
       max_tftext1
       max_tftext2
       max_tftext3;

data _null_;
 infile "&metadata_filepath" lrecl=32767;
 input @ ;
 *** Remove the comment marks from the buffer ***;
 _infile_=compress(_infile_, "*");

 *** Get the Last run date/time into a macro variable ***;
 if index(upcase(_infile_), "LAST RUN DATE/TIME=")>0 then
   call symput("LAST_DT_TM", substr(_infile_, index(_infile_, "=")+1));

 *** Get the QC date and time into a macro variable ***;
 if index(upcase(_infile_), "QC DATE/TIME=")>0 then
   call symput("QC_DT_TM", substr(_infile_, index(_infile_, "=")+1));

 *** Get the maximum program id length into a macro variable ***;
 if index(upcase(_infile_), "MAX_PROGID=") then
   call symput("MAX_PROGID_LG", substr(_infile_, index(_infile_, "=")+1));

 *** Get the maximum T/F and Seq length into a macro variable ***;
 if index(upcase(_infile_), "MAX_TORFSEQ=") then
   call symput("MAX_TORFSEQ_LG", substr(_infile_, index(_infile_, "=")+1));

 *** Get the maximum length for the TF_TEXT variables into macro variables ***;
 if index(upcase(_infile_), "MAX_TFTEXT")>0 then
   call symput("MAX_TFTEXT"||substr(_infile_,index(_infile_ , "=")-1, 1),
               substr(_infile_, index(_infile_, "=")+1));
 if index(upcase(_infile_), "END OF DOCUMENT PROPERTIES") then stop;
 run ;

 %* Make sure text length macros have a valid value (>0) for reading in purposes
only
    other vars are checked with the capture and storage metadata QC ;
 %if &max_tftext1=0 %then %let max_tftext1=1;
 %if &max_tftext2=0 %then %let max_tftext2=1;
 %if &max_tftext3=0 %then %let max_tftext3=1;
```

```
 %* Get the maximum length of a record to make reading in more efficient
    (30 is delimiter count+user modified+justification+6 buffer space) ;
 %local lrecl_length;
 %let
lrecl_length=%eval(&max_progid_lg+&max_torfseq_lg+&max_tftext1+&max_tftext2+&max_tf
text3+30);

 %* Check LRECL_LENGTH for >32767 issue a conditional alert ;
 %if &lrecl_length>32767 %then %do;
  %put ALERT_C: (&sysmacroname) The length of the longest record is >32767 causing
possible truncation.;
  %put ALERT_C: (&sysmacroname) Careful scrutiny of the TFs is required!;
 %end;

 %*** Read in Global TFs from .TF file  ***;
 %local gtitles gfoots ;
 %if &globaltfs. ge 1 %then %do ;
 data GlobalTFs(drop=_blank flag _user_modified _torf_seq i);
  length _torf $10.;
  retain flag 0;
  infile "&metadata_filepath." lrecl=&lrecl_length;
  input @;
  if index(upcase(_infile_),trim(upcase("GLOBALTFS&globaltfs."))||"»") then do;
   flag=1;

    infile "&metadata_filepath" delimiter='bb'x missover dsd lrecl=&lrecl_length ;
    attrib
         _progid        length=$&max_progid_lg.          label='Name of the
program'
     _torf_seq      length=$4                        label='(T)Title/(F)footnote
and Sequence Number'
     _user_modified length=$&MAX_TORFSEQ_LG.         label="User Modified Y/N"
     _blank         length=$1                        label='Blank(dropped) for
visual reference only'
     _just1         length=$3                        label='Justification for first
element'
     _tftext1       length=$%eval(&max_tftext1.+100) label='Text for first element'
 informat=$char%eval(&max_tftext1.+100).
     _just2         length=$4                        label='Justification for
second element'
     _tftext2       length=$%eval(&max_tftext2.+100) label='Text for second
element' informat=$char%eval(&max_tftext2.+100).
     _just3         length=$4                        label='Justification for third
element'
     _tftext3       length=$%eval(&max_tftext3.+100) label='Text for third element'
 informat=$char%eval(&max_tftext3.+100).
     ;
    input _progid $ _torf_seq $ _user_modified $ _blank $ _just1 $_tftext1 $ _just2
$
```

```
          _tftext2 $ _just3 $_tftext3 $;

    %*** Separate the sequence and TF from _torf ***;
    if substr(compress(_torf_seq),1,1)="F" then _torf="Footnote";
    else if substr(compress(_torf_seq),1,1)="T" then _torf="Title";

    _seq=input(substr(compress(_torf_seq),2), ?best8.);

    %* Concatenate the j= text onto the justification read in ;
    if _just1 ne '' then _just1="J="||trim(_just1);
    if _just2 ne '' then _just2=" J="||trim(_just2);
    if _just3 ne '' then _just3=" J="||trim(_just3);

    %* Modify the _tftext variables for quotes and resolution needs;
    array tftext{3} _tftext1-_tftext3 ;
    do i = 1 to dim(tftext) ;

       %* Resolve the dynamic text (macro variables or calls) ;
       if indexc(tftext{i}, '%&') then tftext{i}=resolve( tftext{i} ) ;
           %* Add a DOUBLE quote to the beginning and end of the _TF_TEXT variables
;
       if ~missing( tftext{i} ) then tftext{i}='"'||trim(tftext{i})||'"';
           %* Remove those lines with the text PLACEHOLDER ;
       if index( upcase(tftext{i}), 'PLACEHOLDER' )>0 then delete ;
    end;

    output;

  end;
  %*** Stop reading in the file if you pass GLOBALTFS ***;
  if (flag=1) and
((index(upcase(_infile_),trim(upcase("GLOBALTFS&globaltfs."))||"»") ne 1) and
(index(_infile_, "***") = 0)) then stop;
  run;

    %* Get a count of global titles and footnotes ;
    proc sql noprint ;
      select  max( _seq ) into :gtitles
        from    GlobalTFs
        where   _torf eqt 'T'
        ;
      select max( _seq ) into :gfoots
        from   GlobalTFs
        where  _torf eqt 'F'
        ;
    quit ;

  %end ; %* End reading in Global TFs ;
```

```
 %*** Read in TF_EXTRACT.TF until you get to the tlf_progname, output and stop
reading ***;
 data &tlf_progname._TFx(drop=_blank flag _user_modified _torf_seq i);
  length _torf $10.;
  retain flag 0;
  infile "&metadata_filepath" lrecl=&lrecl_length;
  input @;

  %if &tlf_progname.=ALL %then %do ;
    if  index(upcase(_infile_),"GLOBALTFS")=0 and
                index(_infile_,'***')=0  and
        index(upcase(_infile_),'GET_TF')=0 then do ;
   %end ;
  %else %do ;
        if upcase(scan(_infile_,1,"»"))=trim(upcase("&tlf_progname"))  and
index(_infile_,'***')=0  then do ; %* v2.0.0.15 & V2.1.0.1 modifications;
   %end ;
   flag=1;

    infile "&metadata_filepath" delimiter='bb'x missover dsd lrecl=&lrecl_length ;
    attrib
         _progid        length=$&max_progid_lg.          label='Name of the
program'
     _torf_seq      length=$4                        label='(T)Title/(F)footnote
and Sequence Number'
     _user_modified length=$&MAX_TORFSEQ_LG.         label="User Modified Y/N"
     _blank         length=$1                        label='Blank(dropped) for
visual reference only'
     _just1         length=$3                        label='Justification for first
element'
     _tftext1       length=$%eval(&max_tftext1.+100) label='Text for first element'
 informat=$char%eval(&max_tftext1.+100).
     _just2         length=$4                        label='Justification for
second element'
     _tftext2       length=$%eval(&max_tftext2.+100) label='Text for second
element' informat=$char%eval(&max_tftext2.+100).
     _just3         length=$4                        label='Justification for third
element'
     _tftext3       length=$%eval(&max_tftext3.+100) label='Text for third element'
 informat=$char%eval(&max_tftext3.+100).;
    input _progid $ _torf_seq $ _user_modified $ _blank $ _just1 $_tftext1 $ _just2
$
              _tftext2 $ _just3 $_tftext3 $;

    %*** Separate the sequence and TF from _torf ***;
        if ~missing( _torf_seq ) then do ;
      if substr(compress(_torf_seq),1,1)="F" then _torf="Footnote";
        else if substr(compress(_torf_seq),1,1)="T" then _torf="Title";
        end ;
```

```
    %* 20110131 (b) (6) - removed logic for incremented _TorF_SEQ for
BLANK_FIRST_FOOTNOTE ;
    _seq=input( compress(_torf_seq,'TF'), ?best8. ) ;


    %*** Concatenate the j= text onto the justification read in ***;
    if _just1 ne '' then _just1="J="||trim(_just1);
    if _just2 ne '' then _just2=" J="||trim(_just2);
    if _just3 ne '' then _just3=" J="||trim(_just3);

   %* Modify the _tftext variables for quotes and resolution needs;
    array tftext{3} _tftext1-_tftext3;
    do i = 1 to dim(tftext);
       %* Resolve the dynamic text (macro variables or calls) ;
       if indexc(tftext{i}, '%&') then tftext{i}=resolve(tftext{i});

           %* Add a DOUBLE quote to the beginning and end of the _TF_TEXT variables
;
       if ~missing( tftext{i} ) then tftext{i}='"'||trim(tftext{i})||'"';
    end;
   if ~missing(_torf) then output ;
  end ;


  %* Stop reading in the file if the program name changes once you found it ;
  %if &tlf_progname. ne ALL %then %do ;
  if    (flag=1)
        and ((index(upcase(_infile_),trim(upcase("&tlf_progname"))||"»") ne 1) and
(index(_infile_, "***") = 0))
        then stop;
  %end ;
  run;

  proc sort data=&tlf_progname._TFx ;
    by _progid descending _TorF _seq ;
        run ;

  %* Adjust the TF sequence counter for global TFs ;
  data &tlf_progname._TFxx ;
   %if &globaltfs. gt 0 and &gtitles. gt 0 %then %do ;
   do until( eofT ) ;
      set globaltfs end=eofT ;
          where _torf eq: 'T' ;
          if ~missing( _tftext1 ) then __seq=_seq ;
          output ;
   end ;
   %end ;
   %else __seq=0 ; ;
   do until( last._progid ) ;
```

```
      set &tlf_progname._TFx ;
          by _progid ;
      %if &globaltfs. gt 0 %then %do ;
          if _torf eq: 'T' then _seq=sum(_seq,__seq,0);
      %end ;
          output ;
   end ;
   if _torf eq: 'F' then ___seq=_seq ; %* Number of non-global footnotes ;
      else ___seq=0 ;
   %if &globaltfs. gt 0 and &gfoots. gt 0 %then %do ;
   do n=0 by 0 until( eofF ) ;
      set globaltfs end=eofF ;
          where _torf eq: 'F' ;
          if ~missing( _tftext1 ) then do ;
            n++1 ;
                _seq=___seq+n ;
            output ;
      end ;
   end ;
   %end ;
   run ;

   %* Add markup commands for hanging indents, footnote overline and blank first
footnote ;
   %* Add 'a0'x to the first TF_TEXT if USE_PARSE_CHAR=Yes ;
   data &tlf_progname._TFxxx ;
    set &tlf_progname._TFxx ;

     %* V2.1.0.4 update ;      * Add a non-breaking space to first footnote if it
is null ;     if _torf eq: 'F' and _seq=1 and missing(_tftext1) then _tftext1='"'
|| "&escapechar.w" || '"' ;    %* Apply hanging indents ;
        %if &footnote_hanging_indent.=YES %then %do ;
         label len  ="# of Characters for Hanging Indent"
               twips="Twip Shift for Hanging Indent"
                   ;
         if _torf eq: 'F' then do ;
            length len 8 twips $5 ;
        if upcase(_tftext1) in: ( '"NOTE: ', '"NOTE- ' ) then do ;
                    len=6 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                   end ;
        else if upcase(_tftext1) in: ( '"NOTE - ', '"NOTES- ', '"NOTES: ' ) then do
;
                    len=7 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
        else if upcase(_tftext1) eq: '"CROSS REFERENCE(S): ' then do ;
                    len=20 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
```

```
        else if upcase(_tftext1) eq: '"SOURCE DATA: ' then do ;
                    len=13 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
        else if upcase(_tftext1) in: ( '"SOURCE: ', '"SOURCE- ' )  then do ;
                    len=8 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"[1] ', '"[2] ', '"[3] ', '"[4] ', '"[5] ',
'"[6] ',
                                                                '"[7] ',
                                                                '"[8] ', '"[9] ', '"(1)
', '"(2) ', '"(3) ', '"(4) ',
                                                                '"(5) ', '"[a] ', '"[b]
', '"[c] ', '"[d] ', '"[e] ','"[f] ',
                                                                '"(6) ',  '"(7) ', '"(8)
', '"(9) ') then do ;
                    len=4 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"[10] ', '"[12] ', '"[13] ', '"[14] ',
'"[15] ', '"[16] ',
                                                                '"[17] ', '"[11] ','"(11)
',
                                                                '"[18] ', '"[19] ',
'"(10) ', '"(12) ', '"(13) ', '"(14) ',
                                                                '"(15) ',
                                                                '"(16) ',  '"(17) ',
'"(18) ', '"(19) ') then do ;
                    len=5 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"- [1] ', '"- [2] ', '"- [3] ', '"- [4] ',
'"- [5] ',
                                                                '"- [6] ', '"- [7] ',
                                                                '"- [8] ', '"- [9] ', '"-
(1) ', '"- (2) ', '"- (3) ',
                                                                '"- (4) ', '"- (5) ',
                                                                '"- (6) ', '"- (7) ', '"-
(8) ', '"- (9) ') then do ;
                    len=6 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"- [11] ', '"- [12] ', '"- [13] ', '"- [14]
', '"- [15] ',
                                                                '"- [16] ', '"- [17] ',
'"- (10) ', '"- [10] ',
                                                                '"- [18] ', '"- [19] ',
'"- (11) ', '"- (12) ', '"- (13) ',
                                                                '"- (14) ', '"- (15) ',
```

```
                                                        '"- (16) ', '"- (17) ',
'"- (18) ', '"- (19) ') then do ;
                    len=7 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"-[1] ', '"-[2] ', '"-[3] ', '"-[4] ',
'"-[5] ',
                                                        '"-[6] ', '"-[7] ',
                                                        '"-[8] ', '"-[9] ',
'"-(1) ', '"-(2) ', '"-(3) ',
                                                        '"-(4) ', '"-(5) ',
                                                        '"-(6) ', '"-(7) ',
'"-(8) ', '"-(9) ') then do ;
                    len=5 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"-[10] ', '"-[12] ', '"-[13] ', '"-[14] ',
'"-[15] ',
                                                        '"-[16] ', '"-[17] ',
'"-[11] ', '"-(10) ',
                                                        '"-[18] ', '"-[19] ',
'"-(11) ', '"-(12) ', '"-(13) ',
                                                        '"-(14) ', '"-(15) ',
                                                        '"-(16) ', '"-(17) ',
'"-(18) ', '"-(19) ') then do ;
                    len=6 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"1     ', '"2     ', '"3     ', '"4     ',
'"5     ' , '"6     ', '"7     ', '"8     ', '"9     ',
                                        '"a     ', '"b     ', '"c     ', '"d     ',
'"e     ', '"f     ',
                                        '"g     ', '"h     ', '"i     ', '"j     ' )
then do ;
                                    len=6 ;
                                        twips=trim(left(
put(len*&font_size.*12,5.0) )) ;
                    end ;
                else if _tftext1 in: ('"1 ', '"2 ', '"3 ', '"4 ', '"5 ' , '"6 ',
'"7 ', '"8 ', '"9 ') then do ;
                    len=2 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
                else if _tftext1 in: ('"10 ', '"11 ', '"12 ', '"13 ', '"14 ', '"15
' , '"16 ', '"17 ', '"18 ', '"19 ') then do ;
                    len=3 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                  end ;
        %* This case should be listed last ;
                else if _tftext1 in: ( '"- ' ) then do ;
```

```
                    len=2 ;
                        twips=trim(left( put(len*&font_size.*12,5.0) )) ;
                   end ;
            if ~missing(twips) then do ;

_tftext1='"&escapechar.R/RTF'||"'\li"||trim(twips)||"\fi-"||trim(twips)||"'
"||substr(_tftext1,2,length(_tftext1)-2)||'"' ;
            end ;
           end ;
         %end ;

         %* Add footnote overline and/or blank first footnote AFTER hanging indent ;
         %* Do not add footnote overline if using a footnote page ;
         %if &footnote_overline.=YES or &blank_first_footnote.=YES  %then %do ;
            if _torf eq: 'F' and _seq=1 then do ;
            put 'Entering Footnote formatting section' ;
        %if &footnote_hanging_indent.=YES and &footnote_overline.=YES and
&blank_first_footnote.=NO %then %do ;
                 %* IF condition has phrase to trigger hanging indent, ELSE does not
;
                        if substr(_tftext1,14,5)='R/RTF' then
_tftext1='"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\."
||substr(trim(_tftext1),20) ;
                          else
_tftext1='"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\.'
" ||substr(trim(_tftext1),2) ;
            %end ;
        %else %if &footnote_hanging_indent.=YES and &footnote_overline.=YES and
&blank_first_footnote.=YES %then %do ;
                        if index(_tftext1,"/RTF'")>0 then
_tftext1=tranwrd(_tftext1,"/RTF'","/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.
\par ") ;
                         else
_tftext1='"'||"&escapechar.R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\par' "
|| substr(_tftext1,2) ;
            %end ;
        %else %if &footnote_hanging_indent.=YES and &footnote_overline.=NO and
&blank_first_footnote.=YES %then %do ;
                        *_tftext1=tranwrd(_tftext1,"/RTF'","/RTF'\par ") ;
                   %* IF condition has phrase to trigger hanging indent, ELSE does
not ;
                        if substr(_tftext1,14,5)='R/RTF' then
_tftext1=tranwrd(_tftext1,"/RTF'","/RTF'\par ") ;
                          else _tftext1='"'||"&escapechar."||"R/RTF'\par '"
||substr(trim(_tftext1),2) ;
            %end ;
        %else %if &footnote_hanging_indent. ne YES and &footnote_overline.=YES and
&blank_first_footnote.=NO %then %do ;

_tftext1='"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\.'
```

```
" ||substr(trim(_tftext1),2) ;
           %end ;
       %else %if &footnote_hanging_indent. ne YES and &footnote_overline.=YES and
&blank_first_footnote.=YES %then %do ;

_tftext1='"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\li
ne '" ||substr(trim(_tftext1),2) ;
           %end ;
       %else %if &footnote_hanging_indent. ne YES and &footnote_overline.=NO and
&blank_first_footnote.=YES %then %do ;
                    _tftext1='"'||"&escapechar."||"R/RTF'\par '"
||substr(trim(_tftext1),2) ;
           %end ;
           put 'Exiting Footnote formatting section' ;
           end ;
        %end ;
    %*** Add 'a0'x to the first TF_TEXT if use_parse_char=Yes ***;
    %if &use_parse_char.=YES %then %do;
      _tftext1=trim(_tftext1)||" 'a0'x";
    %end;
   run ;


 %* Check for an empty data set caused by not finding the TLF_PROGNAME and
    issue a meaningful message ;
 %* Check the data set without the GLOBALTFs ;
 %local dsid bool;
 %let dsid=%sysfunc(open(&tlf_progname._tfx));
 %let bool=%sysfunc(attrn(&dsid, nobs));
 %let dsid=%sysfunc(close(&dsid));
 %if &bool=0 %then %do;
   %put ALERT_P: (&sysmacroname) &tlf_progname was not found in
&metadata_filepath.. Please enter an appropriate TLF_PROGNAME to continue.
Processing we be stopped!;
    %let get_tf_rc=2;
    %goto exit;
 %end;


 %* Sort in order to merge with the BLANK_LAST_TITLE and/or blank first footnote
(when needed) ;
 proc sort data=&tlf_progname._tfxxx ;
  by descending _torf _seq ;
  run ;

 %* Get title and footnote count ;
 %* TITLE_COUNT mac var is needed for the BLANK_LAST_TITLE in the following step ;
 proc sql noprint;
   select max(_seq) into: Title_count
   from   &tlf_progname._tfxxx
```

```
   where _torf="Title"
   ;

   select max(_seq) into: Footnote_count
   from   &tlf_progname._tfxxx
   where  _torf="Footnote"
   ;
   quit ;

 %* If needed create a blank last title and/or blank first footnote ***;
 %if &blank_last_title.=YES %then %do ;
  data &tlf_progname.blanktfs;
    %if &blank_last_title.=YES %then %do;
      _progid="&tlf_progname";
      _torf="Title";
      _seq=%eval(&title_count+1);
      call symput( 'title_count', put(_seq,best.) ) ;
      _just1="J=L";
      %* SMO added a left justify RTF control word because SAS will not output a
blank at the beginning or end ;
          _tftext1='"'||"&escapechar.R/RTF'\ql'"||'"';
      _just2='';
      _tftext2='';
      _just3='';
      _tftext3='';
      output &tlf_progname.blanktfs;
    %end;
  run;

  proc sort data=&tlf_progname.blanktfs;
    by descending _torf _seq;
  run;
 %end ;

 data &tlf_progname._tf;
  %* Set the length of tf_text to the max line length +200 for a buffer with
dynamic text ;
    length tf_text $%eval(&lrecl_length.+200);
  %if &blank_last_title.=YES %then %do;
    set &tlf_progname._tfxxx
        &tlf_progname.blanktfs;
  %end;
  %else %do;
    set &tlf_progname._tfxxx;
  %end;
    by descending _torf _seq;
  tf_text=trim(trim(left(_torf))||trim(left(put(_seq, 8.)))||" "|| _just1||" "||
          trim(_tftext1)||_just2||" "||trim(_tftext2)||_just3||"
"||trim(_tftext3))||";";
  run ;
```

```
 %* Check the title/footnote count for more than 10 TFs;
 %* If there are either more then 10 titles or footnotes then attempt to combine
    adjacent entries ;
  %if  &fpage.=NO  and &tlf_progname. ne ALL and
          (&title_count.>10 or &footnote_count.>10 ) and &combine_tfs.=YES %then
%do ;
    proc datasets library=work nolist ;
          delete &tlf_progname._tfxxxx ; %* Enusre this data set does not exist ;
          run ;
          change &tlf_progname._tf=&tlf_progname._tfxxxx ;
          run ;
          quit ;

    data &tlf_progname._tf( rename=(new_tf_text=tf_text) ) ;

          length  new_tf_text $4000  prev_just1 prev_just2 $4 ;
          prev_just1='';
          prev_just2='' ;

          do until( last._torf  ) ;
            set &tlf_progname._tfxxxx ;
                by descending _torf _seq;

                if first._torf then do ;
                       new_tf_text=substr(tf_text,1,length(tf_text)-1) ; * Remove
trailing semicolon ;
                       prev_just1=_just1 ;
                       prev_just2=_just2 ;
                       last_seq=_seq ;
                  end ;
                 else do ;
                    %* Concatenate adjacent lines if they share a single common
justification ;
                       %* Put a line break in between concatenated lines ;
                       %* Ensure hanging indent commands from previous lines do
not persist ;
                    if (prev_just1=_just1) and missing(_just2)
                               and ( missing(prev_just2) | index( new_tf_text,'"'
)>0 & last_seq ne 1)
                               %if &sysver.=8.2 %then and
sum(length(new_tf_text),length(_tftext1))<1000;
                             then do ;
                            if index( new_tf_text,'"' )>0 then
                                       new_tf_text=trim( new_tf_text ) ||' "' ||
"&escapechar.R/RTF'\par\li0\fi0' " ||
                                                  '" ' || trim(_tftext1 ) ;
                                 else new_tf_text=trim( new_tf_text ) || ' ' ||
trim(_tftext1 ) ; %* No blank line for V8 ;
```

```
                          end ;
                        %* This condition is for beginning a new TF statement - no
line break needed ;
                     else if (prev_just1=_just1) and missing(_just2) and
~missing(prev_just2)
             & last_seq ne 1 then do ;
                       new_tf_text=left( trim( new_tf_text ) ||' ' || trim(_tftext1
) );
                          end ;
                        %* This condition is for beginning a new TF statement b/c
the justification changed from previous ;
                        else if (prev_just1 ne _just1) and missing(_just2) then do
;
                               new_tf_text=trim( new_tf_text ) || ' ;' ;
                               if index( new_tf_text,'"' )>0 then output ;
                               last_seq=last_seq++( index( new_tf_text,'"' )>0 ) ;
                            new_tf_text=trim(_torf) ||
trim(left(put(last_seq,best.))) || ' ' ||
                                                        _just1 || ' ' || trim(_tftext1
) ;
                            prev_just1=_just1 ;
                               prev_just2=_just2 ;
                         end ;
                        else if (prev_just1=_just1) and ( ~missing(_just2) or
~missing(prev_just2)
             %if &sysver.=8.2 %then or sum(length(new_tf_text),length(_tftext1)) >=
1000; )
              then do ;
                               new_tf_text=trim( new_tf_text ) || ' ;' ;
                               if index( new_tf_text,'"' )>0 then output ;
                               last_seq=last_seq++( index( new_tf_text,'"' )>0 ) ;
                            new_tf_text=trim(_torf) ||
trim(left(put(last_seq,best.))) || ' ' ||
                                                        _just1 || ' ' || trim( _tftext1
) ||
                                                            _just2 || ' ' || trim(
_tftext2 ) ||
                                                            _just3 || ' ' || trim(
_tftext3 ) || ';' ;
                            prev_just1=_just1 ;
                               prev_just2=_just2 ;
                               if index( new_tf_text,'"' )>0 then output ;
                               last_seq=last_seq++( index( new_tf_text,'"' )>0 ) ;
                               new_tf_text=trim(_torf) ||
trim(left(put(last_seq,best.))) ||
                                                        ' ' || _just1 || ' '  ;
                         end ;
                  end ;
           end ;
      new_tf_text=trim( new_tf_text ) || ' ;' ;
```

```
          if index( new_tf_text,'"' )>0 then output ;
          drop tf_text ;
          run ;

         %*Re-check if there is more than 10 TFs after combining ;
         %*local title_count2 footnote_count2 ;
         proc sql noprint;
           select max(last_seq) into: Title_count
           from   &tlf_progname._tf
           where _torf="Title"
           ;

           select max(last_seq) into: Footnote_count
           from   &tlf_progname._tf
           where  _torf="Footnote"
           ;
           quit ;

      %if &title_count.>10 or &footnote_count.>10 %then %do;
        %put ALERT_P: (&sysmacroname) Algorithm to consolidate TFs was
unsuccessful.;
        %put ALERT_P: (&sysmacroname) Consider using OUT_TF_DATASET=Yes,
WRITE_TITLES=No and WRITE_FOOTNOTES=No .;
        %put ALERT_P: (&sysmacroname) Processing will be stopped!;
                %let get_tf_rc=5;
                %goto exit ;
      %end ;
          %else %do ;
        %put ALERT_I: (&sysmacroname) Algorithm to consolidate TFs was successful,
but ;
        %put ALERT_I: (&sysmacroname) you should review the output to ensure that
truncation did not occur;
          %end ;

  %end ;
  %else %if &fpage.=NO and
          (&title_count.>10 or &footnote_count.>10 ) and &combine_tfs.=NO %then %do
;
        %put ALERT_P: (&sysmacroname) There are more than 10 titles or 10
footnotes;
        %put ALERT_P: (&sysmacroname) Consider using COMBINE_TFS=Yes. Processing
will be stopped!;
                %let get_tf_rc=6;
                %goto exit ;
      %end ;

  %* For data only mode, put an alert and stop processing ;
  %if &write_titles. eq NO and &write_footnotes. eq NO %then %do ;
    %put Note: (&sysmacroname) The &sysmacroname macro is being run in data set
only mode - no title/footnote statments will be generated.;
```

```
    %goto exit ;
  %end ;


  %* Section for creating footnote page ;
  %if &fpage.=YES %then %do ;
     %put %str(N)OTE-Entering Footnote Page section ;

         data fpage0 ; %* Resolve function in SQL truncates to 200 characters ;
           set &tlf_progname._tf( rename=(_tftext1=__tftext1) ) ;
           length _tftext1 $3000 ;
           _tftext1=resolve( compress(__tftext1,'"') ) ;
        * Replace the parse_char character with its BYTE equiv so it is not viewable
;
           _tftext1=tranwrd( _tftext1, "'a0'x", byte(160) )     ;
           * Remove top border markup command ;
           _tftext1=tranwrd( _tftext1, "\brdrt\brdrs\brdrw10", "\ql " ) ;
           drop __tftext1 ;
           run ;

     proc sql noprint ;
           * Read in the lines for the footnote page ;
           create table fpage1 as
           select   *
           from     fpage0
           where    _torf eqt 'F'
                  & _seq le (&footnote_count.-&fpage_ignore_lines.)
           ;

        * Make a cleaned up version of the fpage ;
            create table fpage as
                select   _tftext1 as text length=1024
                           , _seq
                from     fpage1
                order by _seq
                ;


           %local has_tftext2 ;
           select  count(*) into :has_tftext2
           from    fpage1
           where   _tftext2 is not missing
                        & _torf eqt 'F'
           ;

           drop table fpage0 ;

           %if &has_tftext2.>0 %then %do ;
              %put ALERT_P: (&sysmacroname) FPAGE does not support lines with
multiple justfications ;
```

```
                %let get_tf_rc=4;
                %goto exit ;
          %end ;

          quit ;

          * Remove the lines from the footnote page ;
          data &tlf_progname._tf ;
          %* Set ___SEQ=1 if FPAGE_FIRST_FOOTNOTE populated, else set to 0 ;
           ____seq=( length("&fpage_first_footnote.")>1 ) ;
           do until( eof ) ;
            set &tlf_progname._tf( in=T where=(_torf eq: 'T') )
                    &tlf_progname._tf( in=F where=(_torf eq: 'F' and
                                                                    _seq gt
(&footnote_count.-&fpage_ignore_lines.)) )
                         end=eof
              ;

                  if F then do ;
                         ____seq++1 ;
                     ___footseq='Footnote' || put(____seq,best2.-l) || '' ;
             tf_text=___footseq || substr( tf_text,index(tf_text,' ')) ;
                         %if &footnote_overline.=YES %then %do ;
                         %* Add top border markup command if it is the new first
footnote ;
                           if ____seq=1 then do ;
                                  put tf_text= ;
                                  __firstquote=index( tf_text, '"' ) ;
                                      put  __firstquote= ;
                                      %if &blank_first_footnote.=NO %then
                                         tf_text=substr(tf_text,1,__firstquote)
|| "&escapechar.R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\.'" ||
substr(tf_text,__firstquote+1) %str(;);
                                      %else %if &blank_first_footnote.=YES %then
                                         tf_text=substr(tf_text,1,__firstquote)
|| "&escapechar.R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\par '" ||
substr(tf_text,__firstquote+1) %str(;);
                                      drop __firstquote ;
                                  end ;
                           %end ;
                        end ;
                   output ;
                end ;

          %if %length(&fpage_first_footnote.)>0 %then %do ;
                   * Update FOOTNOTE_COUNT macro variable ;
                   call symput( 'footnote_count', put(____seq,best2.-l) ) ;
          _torf='Footnote' ;
                  _seq=1 ;
                  %if &footnote_overline.=YES and &blank_first_footnote.=NO %then
```

```
                        tf_text='Footnote1 J=L
"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\.'" ||
"&fpage_first_footnote." || '"' %str(;) ;
                    %else %if &footnote_overline.=YES and &blank_first_footnote.=YES
%then
                        tf_text='Footnote1 J=L
"'||"&escapechar."||"R/RTF'\brdrt\brdrs\brdrw&footnote_overline_width.\par '" ||
"&fpage_first_footnote." || '"' %str(;) ;
                    %else tf_text='Footnote1 J=L "' || "&fpage_first_footnote." ||
'"' %str(;) ;
                  %if &use_parse_char.=YES %then tf_text=trim(left(tf_text)) || ' '
|| "'a0'x" || ';' %str(;) ;
                    %else tf_text=trim(left(tf_text)) || ';' %str(;) ;
                  output ;
                 %end ;
            run ;

                proc sort data=&tlf_progname._tf ;
                  by _torf _seq ;
                  run ;

              %put %str(N)OTE-Exiting Footnote Page section ;

          %end ;  %* End FPAGE section - Render footnote page after the TF
statements are generated ;

 %* Clean up TLF_PROGNAME._TF data set ;
 data &tlf_progname._tf ;
   set &tlf_progname._tf( %if &debug.=NO or &debug.=0 or &debug.=1 %then  drop= __:
 ; ) ;
   label _torf='Type'
         _seq='Relative Sequence Order'
   ;
   run ;


 %* Generate the Title statements ;
 data _null_;
  %if &write_titles.=YES %then %do;
    set &tlf_progname._tf(where=(_torf="Title"));
  %end;
  %else %if &write_titles.=NO and &write_footnotes.=YES %then %do;
    set &tlf_progname._tf(where=(_torf="Footnote"));
  %end;
  call execute( trim(tf_text) ) ;
 run ;

 %* Generate the Footnote statements ;
 data _null_;
  %if &write_footnotes.=YES %then %do;
```

```
   set &tlf_progname._tf(where=(_torf="Footnote"));
 %end;
 %else tf_text='*;' %str(;) ;
 %if &fpage.=YES and &fpage_suppress.=YES %then %do ;
   if index( tf_text, 'Footnote1' )>0
              then tf_text=tranwrd( tf_text, "&fpage_first_footnote.", "" ) ;
 %end ;
 call execute( trim(tf_text) ) ;
run ;

%* Generate the footnote page(s) ;
%if &fpage.=YES %then %do ;
   proc report data=fpage nowd ;
    column text ;
        define text / '' display width=98 style=[just=l asis=on] ;
        run ;
%end ;

%* Re-generate all footnote statements, if needed ;
%if &fpage.=YES and &fpage_suppress.=YES %then %do ;
data _null_;
 %if &write_footnotes.=YES %then %do;
   set &tlf_progname._tf(where=(_torf="Footnote"));
 %end;
 %else tf_text='*;' %str(;) ;
 call execute( trim(tf_text) ) ;
run ;
%end ;

%*** Exit is the stop point for data set only mode ***;
%exit:

%* reset sort order ;
%if %sysfunc( exist(&tlf_progname._tf) )=1 %then %do ;
proc sort data=&tlf_progname._tf ;
  by _progid descending _torf _seq ;
run ;
%end ;


%*** Clean up ***;
%if  &blank_last_title.=YES %then %do;
 proc datasets nolist nodetails;
   delete &tlf_progname.blanktfs ;
 run;
 quit;
%end;
%if &out_tf_dataset.=NO and ( &debug.=NO or &debug.=0 or &debug.=1 ) %then %do ;
 proc datasets nolist nodetails ;
   delete &tlf_progname._tf: %if &globaltfs. ge 1 %then globaltfs ; %str(;)
```

```
  run;
  quit ;
  %end ;
  %else %if &out_tf_dataset.=NO and ( &debug.=YES or &debug.=2 or &debug.=3 ) %then
%do ;
     %put ALERT_P: (&sysmacroname) Intermediate data sets created by this macro
call have NOT been deleted to facilitate debugging. ;
     %put ALERT_P: (&sysmacroname) Set DEBUG parameter to NO or 0 to turn off macro
debugging options or 1 ( MPRINT only ) ;
         %put ALERT_P: (&sysmacroname) prior to delivery execution.  ;
  %end ;
 %if &out_tf_dataset.=YES and ( &debug.=NO or &debug.=0 or &debug.=1 ) %then %do ;
  proc datasets nolist nodetails ;
    delete &tlf_progname._tfx: ;
  run;
  quit ;
 %end ;

 %* Restore macro debugging settings ;
 option &mprint. &mlogic. &symbolgen. %str(;) ;

 %* Restore the QUOTELENMAX option ;
 options &defaultquoteoption;

 %put NOTE: Leaving &sysmacroname v&Version macro.;

%mend GET_TF;
```