ABSTRACT
All submission datasets must comply with CDISC guidelines. One of the challenging tasks in following CDISC guidelines is variable text string in a submission dataset should not exceed 200 characters. If this scenario occurs in general observation class domains, the first 200 characters are stored in parent domain variable and other part of the text to be stored in supplemental qualifiers dataset as per CDISC standards. In case of TSVAL and COVAL the first 200 characters are stored in the parent domain variable, and rest of the text to be stored in additional variables (eg: COVAL1, COVAL2…) with each new variable text length less than or equal to 200.

When collecting data from subjects and/or populating trial summary parameters (TSVAL), it is very common for the values to exceed 200 characters. In fact, it is tedious and cumbersome process for a programmer to search each dataset in a library for the variables with text value greater than 200 characters. When such variables are found it is cumbersome to split them into additional variables without awkwardly breaking words. This complicated task can be done swiftly using the macro tool FINDSPLIT (by just passing two parameters) given in this paper.

INTRODUCTION
It is the FDA requirement that clinical trial submission datasets must comply with CDISC standards. One of the CDISC requirements for the submission datasets is the variable text should not exceed 200 characters, but it is most common for sponsor collecting some additional data with text greater than 200 characters in CRF and/or trial summary parameter value (TSVAL) with text string >200. Typically numerous datasets are present in a given library, and each dataset contains tens of variables. To find each variable within the data sets in a specific library that has greater than 200 characters is tedious and a time consuming process for the programmer. However, sometimes it can be overlooked by the programmer causing data truncation in submission datasets.

Even after identifying the variables with value length greater than 200 characters in each dataset of a specific library, it is a challenging task to split them into additional variables without breaking the word in a readable manner and complying CDISC standards.

In order to overcome these challenges, we created a macro tool, FINDSPLIT

Advantages of the Macro Tool FINDSPLIT are:
• It is very easy to use (just a couple of parameters to be passed).
• Finds the variables in datasets with text string greater than 200 characters in a given specific library irrespective of number of datasets.
• Provides the summary in html window.
• If any variable with text string greater than 200 characters in any dataset within a specific library is found, split them into additional variables.
• Splitting occurs in a readable manner and complies with CDISC standards for submission purpose.
• It saves a lot of programmer’s precious time and expedites the submission activities.

Tips for FINDSPLIT Macro Execution:
• Make sure that the dataset names in your library should not be same as given below.
  _TEMPLEN1,  _TEMPLEN2,  _TEMPORARY_INDSN_CONTENTS,
  _TEMPORARY_VAR_LEN_DSN,  _TEMPLCONT1, _TEMPLEN_DSN,
  _TEMPORARY_VAR_NOLEN_DSN,  SASHELPVCOLUMN

If so the macro stops executing, and will be notified in log as a WARNING message.
• Macro stops executing if any of the keyword parameters value is missing and will be notified in log as a WARNING message.
• Keyword parameter SPLIT should have a value of either Y or N.
• Macro stops executing and will be notified in log as a WARNING message, when keyword parameter SPLIT has a value of Y and the following variables preexist in your dataset: X, LENGTHA.

MECHANISM OF FINDSPLIT MACRO TOOL:

<table>
<thead>
<tr>
<th>Keyword Parameter</th>
<th>Parameter Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libname</td>
<td>The library name in which your datasets are stored (eg: WORK, Raw…)</td>
<td>Eg: %findsplit(libname=work, split=)</td>
</tr>
<tr>
<td>split</td>
<td>When keyword parameter value is N (Possible values are Y, N only)</td>
<td>Variable value length information in each dataset with more than 200 characters within the specified library will be given in html window. Eg: %findsplit (libname=work, split=N)</td>
</tr>
<tr>
<td>split</td>
<td>When keyword parameter value is Y (Possible values are Y, N only)</td>
<td>(i) Variable value length information in each dataset with more than 200 characters within specified library will be given in html window. (ii) Stores the variable long text value into additional variables in a readable way with each new variable length less than or equal to 200. The scope of this process is, all datasets within a specified library %findsplit (libname=work, split=Y)</td>
</tr>
</tbody>
</table>

Table 1. Mechanistic aspects of the macro tool FINDSPLIT.

FINDSPLIT Macro Code:

```sas
%macro findsplit (libname=, split=) ;
%if &libname= %then %do;
%put WARNING: The Name of the Library to be Assigned for the Keyword Parameter LIBNAME to Execute this Macro;
%abort cancel;
%end;
%if &split= %then %do;
%put WARNING: Value N to be Assigned to the Keyword Parameter split: if you want to find only the Variable Length > 200 in each dataset in %upcase(&libname) library;
%put WARNING: Value Y to be Assigned to the Keyword Parameter split: if you want to find Variable Length > 200 in each dataset in %upcase(&libname) library and Split them into multiple Variables;
%put WARNING: Otherwise the macro findsplit STOPS EXECUTING.;
%abort cancel ;
%end;
%macro lenfind ;
%if %sysfunc(exist(&libname..sashelpvcolumn)) eq 1 %then %do;
```

2
%put WARNING: In %upcase(&libname) Library, Datasets name should not be identical with the following names:
SASHELPVCOLUMN;
%abort cancel;%end;

data sashelpvcolumn;
set sashelpvcolumn;
if upcase (libname) eq "%upcase(&libname)"
   then chak_var=1;
else chak_var=0;
run;
proc sort data=sashelpvcolumn nodupkey;
by chak_var;
run;
data _null_; 
set sashelpvcolumn;
if chak_var eq 0 then call symput("same_name", "NO");
else call symput("same_name", "YES");
run;
%if &same_name eq YES %then %do;
%put WARNING: In %upcase(&libname) Library, Datasets name should not be identical with the following names:
   "_TEMPCONT1", "_TEMPLEN1", "_TEMPLEN2", "_TEMPLEN_DSN", 
   "_TEMPORARY_VAR_LEN_DSN", "_TEMPORARY_VAR_NOLEN_DSN",
   "_TEMPORARY_INDSN_CONTENTS" then chak_var=1;
%end;

data _null_; 
set sashelpvcolumn;
if upcase (memname) eq "%upcase(memname)"
   then chak_var=1;
else chak_var=0;
run;
proc sort data=sashelpvcolumn nodupkey;
by chak_var;
run;
data _null_; 
set sashelpvcolumn;
if chak_var eq 0 then call symput("same_name", "NO");
else call symput("same_name", "YES");
run;
%if &same_name eq YES %then %do;
%put WARNING: In %upcase(&libname) Library, Datasets name should not be identical with the following names:
   "_TEMPCONT1", "_TEMPLEN1", "_TEMPLEN2", "_TEMPLEN_DSN", 
   "_TEMPORARY_VAR_LEN_DSN", "_TEMPORARY_VAR_NOLEN_DSN", "_TEMPORARY_INDSN_CONTENTS" then chak_var=1;
%end;

data _null_; 
set sashelpvcolumn;
if upcase (memname) eq "%upcase(memname)"
   then chak_var=1;
else chak_var=0;
run;
%if %errorlevel eq 0 %then %do;
%errorlevel=%errorlevel+1;
%end;

data _null_; 
set sashelpvcolumn;
if upcase (memname) eq "%upcase(memname)"
   then chak_var=1;
else chak_var=0;
run;
%if &same_name eq YES %then %do;
%put WARNING: In %upcase(&libname) Library, Datasets name should not be identical with the following names:
   "_TEMPCONT1", "_TEMPLEN1", "_TEMPLEN2", "_TEMPLEN_DSN", 
   "_TEMPORARY_VAR_LEN_DSN", "_TEMPORARY_VAR_NOLEN_DSN", "_TEMPORARY_INDSN_CONTENTS" then chak_var=1;
%end;
keep dsnilb dsn variable max_length_var;
run;
proc append base=_temporary_var_len_dsn data=_TEMPLEN_DSN;
run;
quit;
%end;
%end;
%end;
%if %sysfunc(exist(_temporary_var_len_dsn)) eq 1 %then %do;
ods listing close;
ods html ;
title1 "Below is the Variable Information With Length Greater than 200 in %upcase(&libname) Library Datasets";
proc print data=_temporary_var_len_dsn split="-" NOOBS;
label dsnilib="Library~ Name" dsn="Dataset~ Name" variable="Variable~Name" max_length_var="Variable~Maximum Length" ;
run;
ods html close;
ods listing;
%end;
%else %do;
data _temporary_var_nolen_dsn;
Result= "None of the variable in the %upcase(&libname) Library Datasets with Length Greater Than 200 ";
run;
ods listing close;
ods html ;
proc print data=_temporary_var_nolen_dsn noobs;
run;
ods html close;
ods listing;
%end;
title1 "";
proc datasets lib=work memtype=data nolist;
delete _temporary_var_nolen_dsn _TEMPCONT1 _TEMPLEN1 _TEMPLEN2 _TEMPLEN_DSN sashelpvcolumn;
quity;
%mend lenfind;
%macro splitvar (aval=, indsn=);
%local var1 maxlen x;
proc contents data=&indsn noprint out=._temporary_indsn_contents (keep=name); /"---Check for variables present in the dataset---"/
run;
data _temporary_indsn_contents;
set _temporary_indsn_contents;
if upcase(name) in ( "LENGTHA", "X") then chk_var= 1;
else if upcase(name) not in ( "LENGTHA", "X") then chk_var=0;
run;
proc sort data=_temporary_indsn_contents nodupkey;
by chk_var;
run;
data _null_
set _temporary_indsn_contents;
if chk_var eq 0 then call symput ("samevarname","NO");
else call symput ("samevarname","YES");
run;
proc datasets lib=work memtype=data nolist;
delete _temporary_indsn_contents ;
quit;
%if &samevarname eq YES %then %do;
%put WARNING: %upcase(&INDSN) DATASET HAS THE FOLLOWING VARIABLE NAME/NAMES X,LENGTHA. SO THE MACRO findsplit STOPS EXECUTING.;
%abort cancel;
%end;
proc sql noprint;
select max (length(&aval)) into: maxlen
from &indsn;
quit;
%let var1 = %sysfunc(compress(%sysfunc(ceil(%sysvalf(%sysfunc(divide(&maxlen, 200))+3)))))
proc sql noprprint;
alter table &indsn modify &aval char (%sysvalf(%sysfunc(compress(&maxlen)) +4));
quit;
data &indsn;
set &indsn;
if length(&aval) gt 200 then &aval=strip(&aval)||" ###";
run;
data &indsn;
set &indsn;
lengtha=lengthn(&aval);
array varx (*) $200 &aval.1 - &aval&var1;
do i=1 to dim(varx);
if i = 1 then do ;varx(i)= substr(&aval, 1, ifn(index(reverse(substr(&aval, 1, 200)), " ") ne 0 and
index(reverse(substr(&aval, 1, 200)), ".") ne 0 , ( 200- min(index(reverse(substr(&aval,1,200)), " ")),
index(reverse(substr(&aval,1,200)), ".")));
( 200- max(index(reverse(substr(&aval,1,200)), " "), index(reverse(substr(&aval,1,200)), "."))) ) ;
x=sum(x, length(varx(i)));
end;
if i gt 1 and x< lengtha then do;
varx(i)= substr(&aval, x+1 , ifn(index(reverse(substr(&aval, x+1,min(lengtha-x-1, 200)))) , " ") ne 0 and
index(reverse(substr(&aval, x+1, min(lengtha-x-1, 200))), ".") ne 0 ,
( min(lengtha-x-1, 200) - min(index(reverse(substr(&aval+x,1,min(lengtha-x-1, 200))))) , " "),
index(reverse(substr(&aval+x,1,min(lengtha-x-1, 200))), "."))) ));
( min(lengtha-x, 200)- max(index(reverse(substr(&aval+x,1,min(lengtha-x, 200))))) , " "),
index(reverse(substr(&aval+x,1,min(lengtha-x-1, 200))), "."))) ) ;
x=sum(x, length(varx(i)));
if x eq lengtha then x=x+1;
if varx(i) = "###" then call missing(varx(i));
end;
end;
drop i x;
run;
%do i=1 %to &var1 ;
proc sql noprprint;
select (max(lengthn(&aval.&i))) into: x
from &indsn;
quit;
data &indsn;
set &indsn;
&aval.&i=left (&aval.&i);
rename &aval.&i = &aval.%sysfunc( compress(%eval(&i,-1)));
run;
%if &x eq 0 %then %do;
data &indsn;
set &indsn;
drop &aval.%sysfunc( compress(%eval(&i,-1)));
run;
%end;
%end;
data &indsn;
set &indsn;
drop &aval lengtha;
rename &aval.0=&aval;
run;
%mend splitvar;
%local tolit;
%lenfind;
%if %sysfunc(exist(_temporary_var_len_dsn)) eq 1 and %upcase(&split) eq N %then %do;
As per the CDISC guidelines, the submission datasets must be in Version 5 SAS transport file format. In fact V5 transport file is compatible for the variable with value length less than or equal to 200 characters. So using the macro tool FINDSPLIT, length of the variable with text string greater than 200 characters can be identified and/or split into additional variables in a readable way complying CDISC standards. The scope of FINDSPLIT macro tool is for all datasets within specified library.

REFERENCES
CDISC SDTM/G V3.2
HTTPS://WWW.FDA.GOV/DOWNLOADS/DRUGS/DEVELOPMENTAPPROVALPROCESSES/FORMSSUBMISSION
ACKNOWLEDGEMENTS
I would like to thank Malla Reddy Boda, Associate Director, Shionogi Inc, for his review and providing invaluable comments.

CONTACT INFORMATION
Venkata N Madhira
E-mail: venkatanmadhira@gmail.com
Harish Yeluguri
E-mail: harish.yeluguri@gmail.com

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.