



ABSTRACT

Every time someone validates a new SDTM domain through *Pinnacle-21*, the following recurrent error and notice are usually found in *Pinnacle-21* Validator Report: “**Variable length is too long for actual data (SD1082 error)**” and “**Permissible variable with missing values for all records (SD1078 notice)**”. A SAS script has been developed in order to remove these errors and notices in one click. The SAS script could be run as a SAS code or as a SAS macro within SAS base or submitted as a Job within SAS Clinical Data Integration (CDI). The SAS script goes through all variables in all SDTM domains, detecting the maximum length of their values for every single variable. Afterwards, the length is adjusted for every character variable or the variable is dropped when its length is null. These two actions can be performed on a single step or separate steps.

PROGRAMMING STEPS

- ✓ **STEP 1:** Variable name, type and length will be read from SDTM domains and stored in *ALLDS* SAS dataset
- ✓ **STEP 2:** Previous dataset will be split into three datasets:
 - ✓ *ALLDSC*: character variables in main domains
 - ✓ *ALLDSP*: character variables in supplemental domains
 - ✓ *ALLDSN*: numeric variables
- ✓ **STEP 3:** The longest length will be read for every character variable in the previous datasets (Steps 3a and 3b for *ALLDSC* and *ALLDSP* datasets, respectively)
- ✓ **STEP 4:** Maximum value for numeric variables is calculated
- ✓ **STEP 5:** SAS “Length” statement will be generated for every domain with the updated lengths.
- ✓ **STEP 6:** SAS “Drop” statement will be generated for those variables with null value (character and numeric)
- ✓ **STEP 7:** SAS “Retain” statement will be generated in order to maintain the same variable order within the new domain dataset
- ✓ **STEP 8:** *FINAL* dataset will be created with the previous statements as column variables
- ✓ **STEP 9:** Go through all the domains applying the statements detailed in *FINAL* dataset.

PROGRAMMING STEPS - SAS SCRIPTS

```
Proc sort data=sashelp.vcolumn  
(where=(libname='STD_SDTM' and  
mentype='DATA'))  
keep=libname mentype memname  
name length type varnum)  
out=ALLDS nodupkey;  
by memname name;  
run;
```

Variable attributes
are captured

1

```
data ALLDSC ALLDSP ALLDSN;  
set ALLDS;  
If  
index(uppercase(memname), 'SUPP')>0  
then output ALLDSP;  
else if type^='char' then output  
ALLDSN;  
else output ALLDSC;  
run;
```

Dataset split into
ALLDSC, ALLDSP
and ALLDSN

2

```
data _null_;  
set ALLDSC;  
call execute ("Proc sql;create table len1 as  
select domain as memname,'"||compress(name)||"' as  
name, max(length('||compress(name)||')) as  
maxleng from STD_SDTM."||compress(memname)||  
" where "||compress(name)||" ^=''  
group by domain order by memname,name;quit; data  
allds; merge alls len1;by memname name;run;");  
run;
```

The longest length is read for
every character variable in
ALLDSC dataset

3a

```
data _null_;  
set ALLDSP;  
call execute ("Proc sql;create table len1 as  
select compress('SUPP'||rdomain) as  
memname,'"||compress(name)||"' as  
name,max(length('||compress(name)||')) as  
maxleng from STD_SDTM."||compress(memname)||  
" where "||compress(name)||" ^=''  
group by rdomain order by  
memname,name;quit; data alls; merge alls  
len1;by memname name;run;");  
run;
```

The longest length is read for
every character variable in
ALLDSP dataset

3b

```
data _null_;  
set alldsn;  
call execute ("Proc sql;  
create table drop2 as select  
domain as memname,  
'"||compress(name)||"' as name,  
max('||compress(name)||') as  
maxleng from  
STD_SDTM."||compress(memname)||  
" group by domain  
order by memname,name;quit;  
data alldsn;  
merge alldsn drop2;  
by memname name;run;");  
run;
```

Maximum value
for numeric
variables

4

```
data CHANGELENGTH;  
length _length $2000.;  
set allds  
(where=(.<maxleng<length));  
by memname name;  
aux1=compbl(name)||compress('$'|  
|maxleng||'.');  
if first.memname then do;  
_length=compbl('length'  
'|aux1);  
retain _length;  
end;  
else do;  
_length=compbl(_length||aux1);  
retain _length;  
end;  
if last.memname;  
keep memname _length;  
run;
```

SAS “Length”
statement creation

5

```
data dropvars;  
length _drop $2000.;  
set allds  
(where=(index(memname, 'SUPP')=0  
and maxleng= . and  
type='char')) alldsn  
(where=(maxleng= .));  
by memname name;  
if first.memname then do;  
_drop=compbl('drop '|name);  
retain _drop;  
end;  
else do;  
_drop=compbl(_drop||name);  
retain _drop;  
end;  
if last.memname;  
keep memname _drop;  
run;
```

SAS “Drop”
statement creation

6

```
Proc sort data=allds  
(keep=memname name varnum)  
out=allds_order;  
by memname varnum;  
run;  
data allds_order2;  
length _order $2000.;  
set allds_order;  
by memname varnum;  
if first.memname then do;  
_order=compbl('retain '|name);  
retain _order;  
end;  
else do;  
_order=compbl(_order||name);  
retain _order;  
end;  
if last.memname;  
keep memname _order;  
run;  
Proc sort;by memname;run;
```

SAS “Retain”
statement creation

7

```
data FINAL;  
merge CHANGELENGTH (in=a)  
dropvars (in=b)  
allds_order2;  
by memname;  
if a or b;  
run;
```

FINAL dataset
creation

8

```
data _null_;  
set FINAL;  
call execute  
("data STD_SDTM."||compress(memname)||";"  
||compbl(_order)||";"||compbl(_length)||";"  
set STD_SDTM."||compress(memname)||";"  
||compbl(_drop)||";"  
run;");  
run;
```

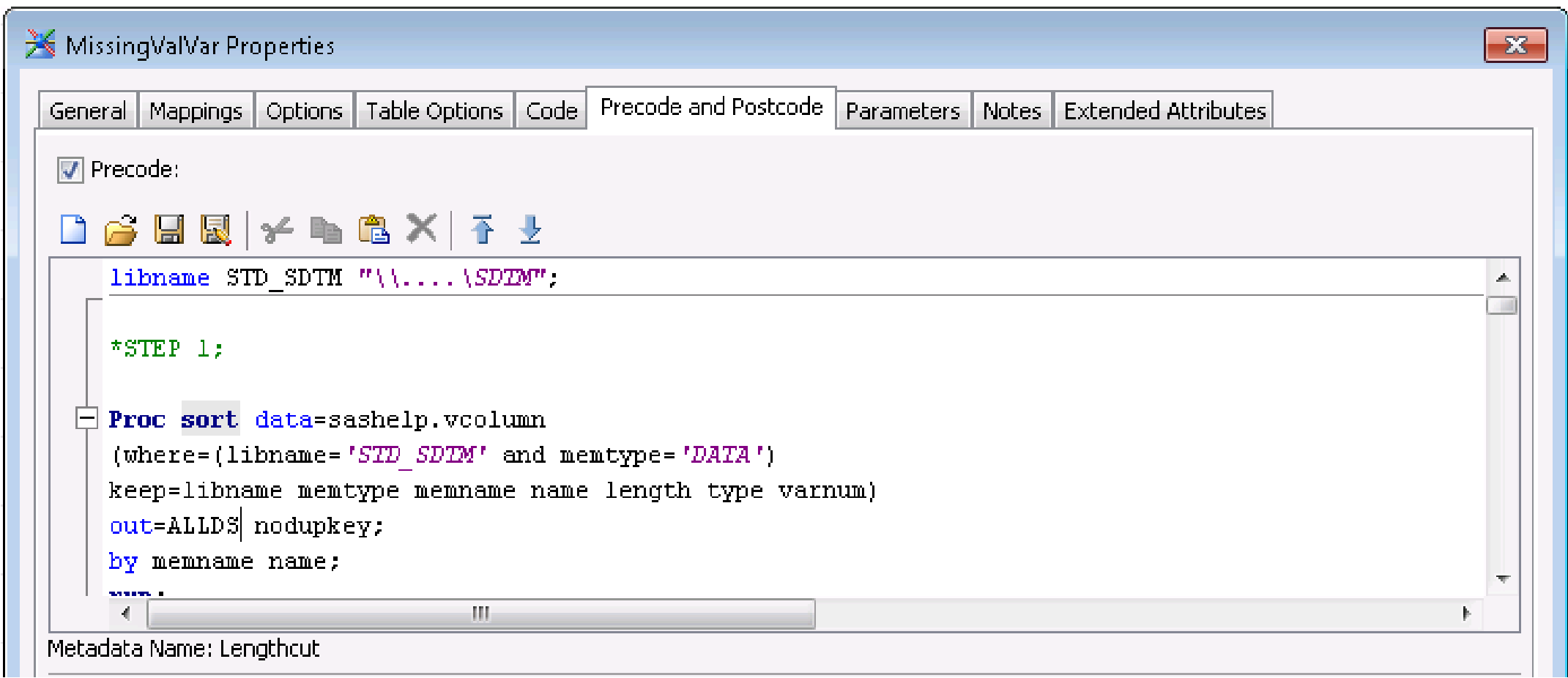
Go through all domains applying the
statements detailed in *FINAL* dataset

9

memname	_length	_drop	_order
AE	length AEACN \$16. AEACNOTH \$20. AEBODSYS \$67. AEDECOD \$62. AEENDTC \$10. AEENRPT \$7.....	drop AEBDSYCD AECAT AECONTRT AEDUR AEENDY AEENRF AEGRPID AEHLGTCD AEHLTCD ...	retain STUDYID DOMAIN USUBJID AESEQ AEGRPID AEREFID AESPID AETERM AEMODIFY AELLT...
CM	length CMCAT \$32. CMCLAS \$63. CMCLASCD \$1. CMDECOD \$45. CMDOSTXT \$3. CMDOSU \$14...	drop CMDOSE CMDOSFRM CMDOSRGM CMDOSTOT CMDUR CMENDY CMENRF CMMODIFY ...	retain STUDYID DOMAIN USUBJID CMSEQ CMGRPID CMSPID CMTRT CMMODIFY CMDECOD...
...	...	...	...

SAS CLINICAL DATA INTEGRATION JOB

The previous SAS script could be also submitted as a Job within SAS Clinical Data Integration (CDI) by using the transformation “Code entered by the user” as a “Pre-code”:



Keep in mind that metadata has to be updated after running the job, therefore the length will be already cut for further versions. It is strongly recommended to use this JOB in CDI only if the final data is used.

CONCLUSIONS

New tools, macros and programs are needed in order to minimize and mitigate the number of issues risen by Pinnacle 21. This code will help us to reduce the number of errors and notices in order to optimize the time for SDTM mapping as an additional programming step.

Your comments and questions are valued and encouraged:
sextremera@pharmamar.com
anieto@pharmamar.com
jgomez@pharmamar.com