Paper 10000-2016

A Macro That Can Fix Data Length Inconsistency and Detect Data Type Inconsistency

Ting Sa, Cincinnati Children's Hospital Medical Center

ABSTRACT

Common tasks that we need to perform are merging or appending SAS® data sets. During this process, we sometimes get error or warning messages saying that the same fields in different SAS data sets have different lengths or different types. If the problems involve a lot of fields and data sets, we need to spend a lot of time to identify those fields and write extra SAS codes to solve the issues. However, if you use the macro in this paper, it can help you identify the fields that have inconsistent data type or length issues. It also solves the length issues automatically by finding the maximum field length among the current data sets and assigning that length to the field. An html report is generated after running the macro that includes the information about which fields' lengths have been changed and which fields have inconsistent data type issues.

INTRODUCTION

Running the following SAS codes, you will get error and warning messages:

```
data sample1;a="abc";b=1;run;
data sample2;a="abcde";run;
data sample3;b="one";run;
data sample4;set sample1 sample2;run;
data sample5;set sample1 sample3;run;
```

Error! Reference source not found. shows the error and warning messages for the above SAS codes:

```
539  data sample4;
540       set sample1 sample2;
541  run;

WARNING: Multiple lengths were specified for the variable a by input data
set(s). This can cause
         truncation of data.
NOTE: There were 1 observations read from the data set WORK.SAMPLE1.
NOTE: There were 1 observations read from the data set WORK.SAMPLE2.
NOTE: The data set WORK.SAMPLE4 has 2 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.03 seconds
      cpu time            0.01 seconds

543  data sample5;
544       set sample1 sample3;
ERROR: Variable b has been defined as both character and numeric.
545  run;

NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set WORK.SAMPLE5 may be incomplete.  When this step was
stopped there were 0
         observations and 2 variables.
WARNING: Data set WORK.SAMPLE5 was not replaced because this step was
stopped.
```

Output 1. The Error and Warning Log Messages

The error and warning messages are caused by (1) the different lengths of field “a” in sample1 and sample2 (2) and the different data types of field “b” in sample1 and sample3.

It will not be hard for a SAS programmer to solve the above two problems. However, if the problems involve a lot of fields and data sets, it will cost a lot of time to identify those fields and write extra SAS codes to solve the issues. Using the macro in this paper, it can help you identify the fields that have inconsistent data type or length issues. It also solves the length issues automatically by finding the maximum field length among the current data sets and assigning that length to the field. An html report is generated after running the macro that includes the information about which fields’ lengths have been changed and which fields have inconsistent data type issues.

THE HELP_CONSISTENCY MACRO SAS CODES

Presented below are the SAS codes for the length_type_consistent macro:

```
%macro help_consistency(libnm=,datasets=%str());
 %let datasets=%sysfunc(upcase(&datasets.));

 proc sql noprint;
 *select all the variables from the data sets and save them in data set
 tmp1;

 %if &datasets.= %then %do;
 create table tmp1 as
 select libname,name,type,length,memname
 from dictionary.columns
 where libname=upcase("&libnm.") and memtype="DATA";
 quit;
 %end;

 %else %do;
 create table tmp1 as
 select libname,name,type,length,memname
 from dictionary.columns
 where libname=upcase("&libnm.") and upcase(memname) in (&datasets.) and
 memtype="DATA";
 %end;

*select all the variables that have different lengths for the same type;
create table tmp2 as
select * from tmp1 where name in
(select distinct name from
(select name,type,count(distinct length) as length_ct from tmp1
group by name,type having calculated length_ct >1))
and type ^="num" order by name,type,length,memname;

*find the maximum length of a variable and save the result to tmp3;
create table tmp3 as
select t.*,max_length from tmp2 as t,(select name,type,max(length) as
max_length from tmp2 group by name,type) as m
where t.name=m.name and t.type=m.type and t.length ^=m.max_length;
quit;

*prepare the SAS codes to change the length to the maximum length;
data tmp3;
set tmp3;
length codes $500.;
codes=catx(" ","proc sql;alter table",cats(libname,".",memname),
```

```
"modify",name,cats("char(",max_length,") format=$",max_length,".;quit;"));
run;

*execute the SAS codes to make the lengh consistent;
data _null_;
set tmp3;
call execute(codes);
run;

*save the length change info to the data set tmp4;
data tmp4;
set tmp3;
length comments $200.;
comments=catx(" ","the length of data field '",name,"'
in",libname,".",memname," has been changed from ",length,"to ",max_length);
keep comments;
run;

proc sql;
*find the varibles with different types;
create table tmp5 as
select * from tmp1 where name in
(select distinct name from
(select name,count(distinct type) as type_ct
from tmp1 group by name
having calculated type_ct >1))
order by name,type,memname;
quit;

data tmp5;
set tmp5;
dataset=cats(libname,".",memname);
field_name=name;
field_type=type;
keep dataset--field_type;
run;

proc sort data=tmp5;by field_name field_type dataset;

ods html;
title "Data Fields that have different data types";
proc print data=tmp5;run;
title "Data Fields whose length have been modified";
proc print data=tmp4;run;
ods html close;
ods listing;
proc datasets;delete tmp1-tmp5;run;quit;
%mend;
```

To call the macro, you just need to pass the library name of the input datasets to the macro parameter “libname” and the input SAS datasets’ names to the “datasets” macro parameter. The input data sets must save in the same library. For the example in this paper, we can call the macro using the following way:

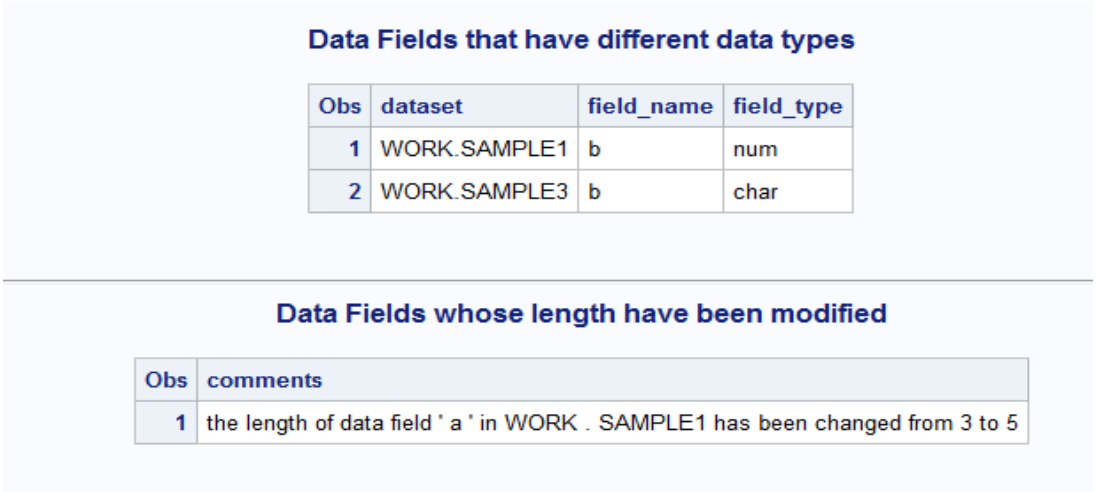
```
%help_consistency(libnm=work,datasets=%str('sample1','sample2','sample3'));
```

If you want to check all the data sets in a library, you don’t need to pass the values to the “datesets” macro variable. For e.g, if you want to check all the data sets in the “work” library, you just need to call the

macro in this way:

```
%help_consistency(libnm=work);
```

After running the macro, the macro will create an html report to summarize the field lengths that have been changed and report the data type issues if there are any. See figure1 for the sample html report.



Data Fields that have different data types

Obs	dataset	field_name	field_type
1	WORK.SAMPLE1	b	num
2	WORK.SAMPLE3	b	char

Data Fields whose length have been modified

Obs	comments
1	the length of data field ' a ' in WORK . SAMPLE1 has been changed from 3 to 5

Figure 1 the Screenshot for the HTML Report

HOW THE MACRO WORKS

Let me explain the macro step by step:

1. The following SAS codes will select all the variables in the input data sets from the dictionary.columns table and save all the things to the “tmp1” data set. If you don’t pass the data set names to the “datasets” macro variable, by default, the macro will check all the SAS data sets in the library. Figure 2 shows you the screenshot of “tmp1” data set for the “sample1”, “sample2” and “sample3” data sets:

```
%if &datasets.= %then %do;
create table tmp1 as
select libname,name,type,length,memname
from dictionary.columns
where libname=upcase("&libnm.") and memtype="DATA";
quit;
%end;

%else %do;
create table tmp1 as
select libname,name,type,length,memname
from dictionary.columns
where libname=upcase("&libnm.") and upcase(memname) in (&datasets.) and
memtype="DATA";
%end;
```

	Library Name	Column Name	Column Type	Column Length	Member Name
1	WORK	a	char	3	SAMPLE1
2	WORK	b	num	8	SAMPLE1
3	WORK	a	char	5	SAMPLE2
4	WORK	b	char	3	SAMPLE3

Figure 2. The Screenshot for Tmp1 SAS Data Set

2. The following SAS codes will select those fields that have the same names, data types but different lengths and save the results to the "tmp2" data set. Figure 3 shows you the screenshot of "tmp2" data set:

```
create table tmp2 as
select * from tmp1 where name in
(select distinct name from
(select name,type,count(distinct length) as length_ct from tmp1
group by name,type having calculated length_ct >1))
and type ^="num" order by name,type,length,memname;
```

	Library Name	Column Name	Column Type	Column Length	Member Name
1	WORK	a	char	3	SAMPLE1
2	WORK	a	char	5	SAMPLE2

Figure 3. The Screenshot for Tmp2 SAS Data Set

3. The following SAS codes create "tmp3" data set that saves those fields whose lengths are smaller than the maximum field length. Figure 4 shows you the screenshot of "tmp3" data set:

```
create table tmp3 as
select t.*,max_length from tmp2 as t,(select name,type,max(length) as
max_length from tmp2 group by name,type) as m
where t.name=m.name and t.type=m.type and t.length ^=m.max_length;
```

	Library Name	Column Name	Column Type	Column Length	Member Name	max_length
1	WORK	a	char	3	SAMPLE1	5

Figure 4. The Screenshot for Tmp3 SAS Data Set

In this paper's example, the length of field "a" in the "sample1" data set is 3, the maximum field length for "a" is 5; therefore it has been selected.

4. The following SAS codes create a new column "codes" in the "tmp3" data set that saves the codes to change the field length. Figure 5 shows you the screenshot of the updated "tmp3" data set:

```
data tmp3;
set tmp3;
length codes $500.;
codes=catx(" ","proc sql;alter table",cats(libname,".",memname),
"modify", name,cats("char(",max_length,") format=$", max_length,
".;quit;"));
run;
```

	Library Name	Column Name	Column Type	Colum Length	Member Name	max_length	codes
1	WORK	a	char	3	SAMPLE1	5	proc sql;alter table WORK.SAMPLE1 modify a char(5) format=$5.;quit;

codes
proc sql;alter table WORK.SAMPLE1 modify a char(5) format=$5.;quit;

Figure 5. The Screenshot for Tmp3 SAS Data Set with Column "codes"

5. The following SAS codes execute the SAS codes saved in the “codes” column of the “tmp3” data set:

```
data _null_;
set tmp3;
call execute(codes);
run;
```

After this step, all the fields with different lengths will have the same length.

6. The following SAS codes create “tmp4” data set, in this data set, column “comments” contains the information about the fields that have their lengths be changed. Figure 6 shows you the screenshot of the “tmp4” data set:

```
data tmp4;
set tmp3;
length comments $200.;
comments=catx(" ","the length of data field '",name,"'
in",libname,".",memname," has been changed from ",length,"to ",max_length);
keep comments;
run;
```

	comments
1	the length of data field ' a ' in WORK . SAMPLE1 has been changed from 3 to 5

Figure 6. The Screenshot for Tmp4 SAS Data Set

7. The following SAS codes will select those fields that have the same names, but different data types and save the results to the “tmp5” data set. Figure 7 shows you the screenshot of “tmp5” data set:

```
proc sql;
create table tmp5 as
select * from tmp1 where name in
(select distinct name from
(select name,count(distinct type) as type_ct
from tmp1 group by name
having calculated type_ct >1))
order by name,type,memname;
quit;

data tmp5;
set tmp5;
dataset=cats(libname,".",memname);
field_name=name;
field_type=type;
keep dataset--field_type;
run;

proc sort data=tmp5;by dataset field_name field_type;
```

	dataset	field_name	field_type
1	WORK.SAMPLE1	b	num
2	WORK.SAMPLE3	b	char

Figure 7. The Screenshot for Tmp5 SAS Data Set

8. The following SAS codes generate the html report to summarize the field lengths that have been changed and report the data type issues if there are any:

```
ods html;
title "Data Fields that have different data types";
proc print data=tmp5;run;
title "Data Fields whose length have been modified";
proc print data=tmp4;run;
ods html close;
```

CONCLUSION

The macro presented in this paper can be used as a helpful tool to help with the data length and data type inconsistency issues especially in the situation where there are a lot of inconsistent fields in different SAS data sets.

ACKNOWLEDGMENTS

The author wishes to thank the Division of Biostatistics and Epidemiology at Cincinnati Children's Hospital Medical Center for its support.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Ting Sa
Division of Biostatistics and Epidemiology
Cincinnati Children's Hospital Medical Center
3333 Burnet Ave. Cincinnati, OH 45229
(513) 636-3674
E-mail: Ting.Sa@cchmc.org