

## How to automatically cover arbitrary changes

### Using automatic code generation method in SAS Enterprise Guide

Kaiqing Fan, Mastech Digital Inc.

Jing Wang, China Minsheng Bank

#### ABSTRACT

In industry, the tremendous large data files and the file numbers from different business models are keeping updating every day; all variables, their values, and the number of variables and the number observations are keep changing; business requirements are keeping changing too. It causes lots of headaches on how to handle these changes. To use the SAS code generation method and automatically cover these arbitrary changes in SAS engines is a perfect option because the automatic code generation method not only can automatically cover all changes but also can avoid developing lots of codes and save a lot of developing time, and it makes the SAS engines readable, easy to understand, and easy to be handled and debugged too.

#### INTRODUCTION AND ITS ADVANTAGES

In banking, the tremendous large data files and the file number from different models are keeping changing; all variables, their values, and the number of observations are keep changing; business requirements are keeping changing too. It causes lots of headaches on how to handle these changes. Here I would like to introduce how to use the automatic code generation method to automatically cover these arbitrary changes in SAS engines. Its advantages are clear:

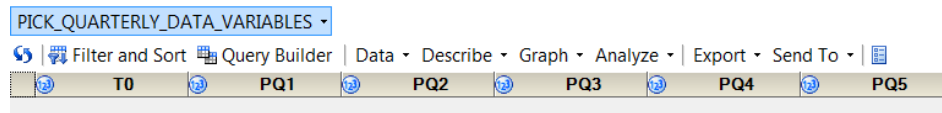
- 1) It can automatically handle the random changes of the number of variables and their values much more efficiently than traditional SAS engines, which require manual intervention to change or modify or update the code;
- 2) It can make very complex and time-consuming coding issues to become very simple and easy;
- 3) Therefore it dramatically decreases the development, modification time for SAS engines when updated requirements appear.

#### WAY1: To cover the changing of certain variables ---- A simple example

In the following PROC SQL, the highlighted yellow part currently contains PQ23, but the number of PQs is changing, it can be PQ13, PQ53 ...

```
proc sql;
create table all_summation_files
as select distinct Run_ID, Model_ID, Model Name, Mileage, Sex, State,
sum(T0) as T0, sum(PQ1) as PQ1, sum(PQ2) as PQ2, sum(PQ3) as PQ3, sum(PQ4) as PQ4,
sum(PQ5) as PQ5, sum(PQ6) as PQ6, sum(PQ7) as PQ7, sum(PQ8) as PQ8, sum(PQ9) as PQ9,
sum(PQ10) as PQ10, sum(PQ11) as PQ11, sum(PQ12) as PQ12, sum(PQ13) as PQ13, sum(PQ14) as
PQ14, sum(PQ15) as PQ15, sum(PQ16) as PQ16, sum(PQ17) as PQ17, sum(PQ18) as PQ18, sum(PQ19)
as PQ19, sum(PQ20) as PQ20, sum(PQ21) as PQ21, sum(PQ22) as PQ22, sum(PQ23) as PQ23
from all_model_files
group by Run_ID, Model_ID, Mileage, Age, Sex, State
order by Run_ID, Model_ID, Mileage, Age, Sex, State;
run; quit;
```

How to handle the change, here is the solution:



The pick\_quarterly\_data\_variables.sas7bdat file can include any number of PQs. This number is keeping changing and updating. Automatic code generation method can perfectly cover the changes.

```
/*transpose any number of quarterly variables PQs into one column*/
proc transpose data=pick_quarterly_data_variables(obs=0) out=all_files_PQs;
var T0 PQ:;
run;
```

	_NAME_
1	T0
2	PQ1
3	PQ2
4	PQ3
5	PQ4
6	PQ5
7	PQ6
8	PQ7
9	PQ8

```
/* create the sum function formula which will be used into the following codes*/
data edited_PQs;
set all_files_PQs;
sum_formula = 'sum('||strip(_NAME_)||') as '||strip(_NAME_);
run;
```

	_NAME_	sum_formula
1	T0	sum(T0) as T0
2	PQ1	sum(PQ1) as PQ1
3	PQ2	sum(PQ2) as PQ2
4	PQ3	sum(PQ3) as PQ3
5	PQ4	sum(PQ4) as PQ4
6	PQ5	sum(PQ5) as PQ5
7	PQ6	sum(PQ6) as PQ6
8	PQ7	sum(PQ7) as PQ7
9	PQ8	sum(PQ8) as PQ8
10	PQ9	sum(PQ9) as PQ9
11	PQ10	sum(PQ10) as PQ10

```
/*define the formula column as a SAS macro parameter, we can use it anywhere if needed*/
proc SQL; select sum_formula into : sum_formula separated by ',' from edited_PQs; quit;
```

OR

```
proc SQL; select 'sum('||strip(_NAME_)||') as '||strip(_NAME_) into : sum_formula separated
by ',' from all_files_PQs; quit;
%put &sum_formula;
```

```

&sum_formula
Macro variable SUM_FORMULA resolves to sum(T0) as T0,sum(PQ1) as PQ1,sum(PQ2) as PQ2,sum(PQ3) as PQ3,sum(PQ4) as
PQ4,sum(PQ5) as PQ5,sum(PQ6) as PQ6,sum(PQ7) as PQ7,sum(PQ8) as PQ8,sum(PQ9) as PQ9,sum(PQ10) as PQ10,sum(PQ11) as
PQ11,sum(PQ12) as PQ12,sum(PQ13) as PQ13,sum(PQ14) as PQ14,sum(PQ15) as PQ15,sum(PQ16) as PQ16,sum(PQ17) as
PQ17,sum(PQ18) as PQ18,sum(PQ19) as PQ19,sum(PQ20) as PQ20,sum(PQ21) as PQ21,sum(PQ22) as PQ22,sum(PQ23) as PQ23

/*use the SAS macro parameter into our code*/
proc sql;
create table all_summation_files
as select distinct Run_ID, Model_ID, Model_Name, Mileage, Sex, State,
&sum_formula
from all_model_files
group by Run_ID, Model_ID, Mileage, Age, Sex, State
order by Run_ID, Model_ID, Mileage, Age, Sex, State;
run; quit;

```

## WAY2: To cover the random changing of many macro/parameters in a table

When we use PROC TEMPLATE to plot costumed images, we have a CHART\_TYPE\_FILE.sas7bdat table which defines the parameters in PROC TEMPLATE. The chart\_type\_file.sas7bdat looks like the following:

Chart_Type	Graph_Type	Title	Task_Type	Task_Target	Situation	Line_Type	Legend_Value	Slide_Number	Data_Key	Color	font_size	thick	
1	line11	Line	Title1	Historical	Historical	Historical	solid	Historical	1	Key1	CX000000	22pt	6
2	line11	Line	Title1	aaa	REPORTa2017	A	solid	REPORTa 2017 A	1	Key1	CX006E51	22pt	6
3	line11	Line	Title1	aaa	REPORTa2017	B	solid	REPORTa 2017 B	1	Key1	CX0069AA	22pt	6
4	line11	Line	Title1	aaa	REPORTa2017	C	solid	REPORTa 2017 C	1	Key1	CXF58025	22pt	6
5	line11	Line	Title1	aaa	REPORTa2017	D	solid	REPORTa 2017 D	1	Key1	CX7030A0	22pt	6
6	line11	Line	Title1	aaa	REPORTb2017	A	Dash	REPORTb 2017 A	1	Key1	CX006E51	22pt	6
7	line11	Line	Title1	aaa	REPORTb2017	B	Dash	REPORTb 2017 B	1	Key1	CX0069AA	22pt	6
8	line11	Line	Title1	aaa	REPORTb2017	C	Dash	REPORTb 2017 C	1	Key1	CXF58025	22pt	6
9	line12	Line	Title2	aaa	REPORTa2017	A	solid	REPORTa 2017 A	2	Key2	CX006E51	22pt	6
10	line12	Line	Title2	aaa	REPORTa2017	B	solid	REPORTa 2017 B	2	Key2	CX0069AA	22pt	6
11	line12	Line	Title2	aaa	REPORTa2017	C	solid	REPORTa 2017 C	2	Key2	CXF58025	22pt	6
12	line12	Line	Title2	aaa	REPORTa2017	D	solid	REPORTa 2017 D	2	Key2	CX7030A0	22pt	6
13	line12	Line	Title2	aaa	REPORTb2017	B	Dash	REPORTb 2017 B	2	Key2	CX0069AA	22pt	6
14	line12	Line	Title2	aaa	REPORTb2017	C	Dash	REPORTb 2017 C	2	Key2	CXF58025	22pt	6
15	line13	Line	Title3	Historical	Historical	Historical	solid	Historical	3	Key3	CX000000	22pt	6
16	line13	Line	Title3	aaa	REPORTa2017	A	solid	REPORTa 2017 A	3	Key3	CX006E51	22pt	6
17	line13	Line	Title3	aaa	REPORTa2017	B	solid	REPORTa 2017 B	3	Key3	CX0069AA	22pt	6
18	line13	Line	Title3	aaa	REPORTa2017	C	solid	REPORTa 2017 C	3	Key3	CXF58025	22pt	6
19	line13	Line	Title3	aaa	REPORTb2017	A	Dash	REPORTb 2017 A	3	Key3	CX006E51	22pt	6
20	line13	Line	Title3	aaa	REPORTb2017	B	Dash	REPORTb 2017 B	3	Key3	CX0069AA	22pt	6

The above GRAPHIC\_PARAMETERS\_FILE.sas7bdat contains variables such as Chart\_Type, Graph\_Type, Situation, Task\_Type, Task\_Target, Color, Line\_Type, THICK, Font\_Size, Legend\_Value, Title .... All of them will be used into PROC TEMPLATE.

- 1) Each value of Chart\_Type would be one image, such as line11,...,bar11, ..., pie12, ...
- 2) Each row of Chart\_Type would be a line in the line graphics.
- 3) The number of lines in each image like line11 with 7 rows (lines) can be **random** changing. For example, line11 image can have any number of lines.
- 4) For each line, its values of each variable can be **random** changing too.

In this table, there are many variables with many rows, to define them as macro parameters would be a very complex, will cause lots of headaches.

In the following PROC TEMPLATE code, the highlighted parts are keeping changings.

```

proc template;
define statgraph class1;   beginingraph;
entrytitle "Title1" / border=false textattrs=(size=30pt);
discreteattrvar attrvar=classfill var=fore_sce attrmap='colors';

legendItem type=Line name="Historical"/lineattrs=(color=CX000000

```

```

pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="Historical";
legendItem type=Line name="REPORTa 2017 A"/lineattrs=(color=CX006E51
pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 A";
legendItem type=Line name="REPORTa 2017 B"/lineattrs=(color=CX0069AA
pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 B";
legendItem type=Line name="REPORTa 2017 C"/lineattrs=(color=CXF58025
pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 C";
legendItem type=Line name="REPORTa 2017 D"/lineattrs=(color=CX7030A0
pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 D";
legendItem type=Line name="REPORTb 2017 A"/lineattrs=(color=CX006E51
pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 A";
legendItem type=Line name="REPORTb 2017 B"/lineattrs=(color=CX0069AA
pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 B";
legendItem type=Line name="REPORTb 2017 C"/lineattrs=(color=CXF58025
pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 C";

legendItem type=line name="blank"/lineattrs=(color=white)
labelattrs=(size=&font_size) label=" ";
layout overlay /border=false WALLCOLOR=white WALLDISPLAY=(FILL)
xaxisopts=(display=all linearopts=(origin=0 THRESHOLDMIN=1 THRESHOLDMAX=1)
label=' ' griddisplay=off labelattrs=(weight=bold size=&font_size)
tickvalueattrs=(weight=bold size=&font_size)
discreteopts=(tickdisplaylist=(&xaxis_tickdisplaylist)
tickvaluelist=(&xaxis_tickvaluelist) TICKVALUEFITPOLICY=ROTATEALWAYS
TICKVALUEROTATION=VERTICAL))
yaxisopts=(display=(label ticks tickvalues) label=' ' labelattrs=(weight=bold
size=&font_size) griddisplay=on tickvalueattrs=(weight=bold size=&font_size)
linearopts=(integer=FALSE origin=&lower_band
viewmin=&lower_band tickvaluelist=(&yaxis_tickvaluelist) TICKVALUEPRIORITY=TRUE
tickvalueformat=DATA) gridattrs=(pattern=dash THICKNESS=2 color=darkgrey));

seriesplot x=xaxis y=Historical Historical Historical/LEGENDLABEL="Historical"
PRIMARY=TRUE lineattrs=(color=CX000000 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_A_aaa/LEGENDLABEL="REPORTa 2017 A"
PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_B_aaa/LEGENDLABEL="REPORTa 2017 B"
PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_C_aaa/LEGENDLABEL="REPORTa 2017 C"
PRIMARY=TRUE lineattrs=(color=CXF58025 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_D_aaa/LEGENDLABEL="REPORTa 2017 D"
PRIMARY=TRUE lineattrs=(color=CX7030A0 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_A_aaa/LEGENDLABEL="REPORTb 2017 A"
PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=Dash THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_B_aaa/LEGENDLABEL="REPORTb 2017 B"
PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=Dash THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_C_aaa/LEGENDLABEL="REPORTb 2017 C"
PRIMARY=TRUE lineattrs=(color=CXF58025 pattern=Dash THICKNESS=6);

discretelegend "REPORTa 2017 A" "REPORTa 2017 B" "REPORTa 2017 C" "REPORTa 2017 D"
"REPORTb 2017 A" "REPORTb 2017 B" "REPORTb 2017 C" "Historical"
/border=false across=4 autoalign=(BOTTOMLEFT) ORDER=ROWMAJOR;
referenceline y=0/ curvelabel="&curvelb." curvelabelattrs=(size=&font_size weight=bold)
clip=true curvelabelposition=min lineattrs=(pattern=dash THICKNESS=2 color=darkgrey) ;
endlayout;
entryfootnote halign=left TEXTATTRS=(size=20pt weight=bold) "Type=&Type"
halign=right TEXTATTRS=(size=20pt weight=bold) "Publish=&No.";

endgraph;
end;
run;
proc sgrender data=D_TEMP.Line&j._data template=class1; run;

```

Using the following method, it perfectly covers these changings, makes our coding very clear and simple because we don't need to define these values as macro parameters anymore. New variables `Line_legendItem`, `Line_discretelegend`, and `Line_seriesplot` are the exact codes in **PROC TEMPLATE**.

```
data line_legenditem_seriesplot;
```

```

retain Line_legendItem Line_seriesplot Chart_Type
Line_discretelegend Title Legend_Value Situation;
set Graphic_Parameters_file(where=(Slide_Number = 1));
length Line_legendItem $500. Line_seriesplot $500.;
Line_legendItem='legendItem type='||strip(Graph_Type)||' name='
||strip(Legend_Value)||'"/lineattrs=(color='||strip(Color)
||' pattern='||strip(Line_Type)||' THICKNESS='||strip(thick)
||') labelattrs=(size='||strip(font_size)||' label='
||strip(Legend_Value)||'';';
Line_seriesplot='seriesplot x=xaxis||' y='||strip(Task_Target)||'_'
||strip(Situation)||'_'||strip(Task_Type)||'/LEGENDLABEL='
||strip(Legend_Value)||''||' PRIMARY=TRUE lineattrs=(color='||strip(Color)
||' pattern='||strip(Line_Type)||' THICKNESS='||strip(thick)||'';';
Line_discretelegend='''||strip(Legend_Value)||'';
run;

```

LINE\_LEGENDITEM\_SERIESPLOT ▾

Filter and Sort Query Builder Where Data Describe Graph Analyze Export Send To

Chart_Type	Line_legendItem
line11	legendItem type=Line name="Historical"/lineattrs=(color=CX000000 pattern=solid THICKNESS=6) labelattrs=(size=font_size) label="Historical".
line11	legendItem type=Line name="REPORTa 2017 A"/lineattrs=(color=CX006E51 pattern=solid THICKNESS=6) labelattrs=(size=font_size) label="REPORTa 2017 A".
line11	legendItem type=Line name="REPORTa 2017 B"/lineattrs=(color=CX0069AA pattern=solid THICKNESS=6) labelattrs=(size=font_size) label="REPORTa 2017 B".
line11	legendItem type=Line name="REPORTa 2017 C"/lineattrs=(color=CXF58025 pattern=solid THICKNESS=6) labelattrs=(size=font_size) label="REPORTa 2017 C".
line11	legendItem type=Line name="REPORTa 2017 D"/lineattrs=(color=CX7030A0 pattern=solid THICKNESS=6) labelattrs=(size=font_size) label="REPORTa 2017 D".
line11	legendItem type=Line name="REPORTb 2017 A"/lineattrs=(color=CX006E51 pattern=Dash THICKNESS=6) labelattrs=(size=font_size) label="REPORTb 2017 A".
line11	legendItem type=Line name="REPORTb 2017 B"/lineattrs=(color=CX0069AA pattern=Dash THICKNESS=6) labelattrs=(size=font_size) label="REPORTb 2017 B".
line11	legendItem type=Line name="REPORTb 2017 C"/lineattrs=(color=CXF58025 pattern=Dash THICKNESS=6) labelattrs=(size=font_size) label="REPORTb 2017 C".

LINE\_LEGENDITEM\_SERIESPLOT ▾

Filter and Sort Query Builder Where Data Describe Graph Analyze Export Send To

Chart_Type	Line_seriesplot
line11	seriesplot x=xaxis y=Historical_Historical_Historical/LEGENDLABEL="Historical" PRIMARY=TRUE lineattrs=(color=CX000000 pattern=solid THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTa2017_A_aaa/LEGENDLABEL="REPORTa 2017 A" PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=solid THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTa2017_B_aaa/LEGENDLABEL="REPORTa 2017 B" PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=solid THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTa2017_C_aaa/LEGENDLABEL="REPORTa 2017 C" PRIMARY=TRUE lineattrs=(color=CXF58025 pattern=solid THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTa2017_D_aaa/LEGENDLABEL="REPORTa 2017 D" PRIMARY=TRUE lineattrs=(color=CX7030A0 pattern=solid THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTb2017_A_aaa/LEGENDLABEL="REPORTb 2017 A" PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=Dash THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTb2017_B_aaa/LEGENDLABEL="REPORTb 2017 B" PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=Dash THICKNESS=6).
line11	seriesplot x=xaxis y=REPORTb2017_C_aaa/LEGENDLABEL="REPORTb 2017 C" PRIMARY=TRUE lineattrs=(color=CXF58025 pattern=Dash THICKNESS=6).

LINE\_LEGENDITEM\_SERIESPLOT ▾

Filter and Sort Query Builder Where Data Describe Graph

Chart_Type	Line_discretelegend	Title	Legend_Value
line11	"Historical"	Title1	Historical
line11	"REPORTa 2017 A"	Title1	REPORTa 2017 A
line11	"REPORTa 2017 B"	Title1	REPORTa 2017 B
line11	"REPORTa 2017 C"	Title1	REPORTa 2017 C
line11	"REPORTa 2017 D"	Title1	REPORTa 2017 D
line11	"REPORTb 2017 A"	Title1	REPORTb 2017 A
line11	"REPORTb 2017 B"	Title1	REPORTb 2017 B
line11	"REPORTb 2017 C"	Title1	REPORTb 2017 C

```

proc sql;
select distinct Line_legendItem      into :Line_legendItem      separated by ' ' from
line_legenditem_seriesplot;

select distinct Line_seriesplot      into :Line_seriesplot      separated by ' ' from
line_legenditem_seriesplot;

select distinct Line_discretelegend  into :Line_discretelegend separated by ' ' from
line_legenditem_seriesplot;
quit;

```

Here Chart\_Type is the group variable with values as line11, line12 ...; bar11, bar12...; pie11, pie12, ... . Each group of Chart\_Type contains a random number of lines, bars, pies, ... with different features in the graph image. This method perfect covers the random changing of lines.

```

Line_legendItem
legendItem type=Line name="Historical"/lineattrs=(color=CX000000 pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="Historical";
legendItem type=Line name="REPORTa 2017 A"/lineattrs=(color=CX006E51 pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 A";
legendItem type=Line name="REPORTa 2017 B"/lineattrs=(color=CX0069AA pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 B";
legendItem type=Line name="REPORTa 2017 C"/lineattrs=(color=CF58025 pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 C";
legendItem type=Line name="REPORTa 2017 D"/lineattrs=(color=CX7030A0 pattern=solid THICKNESS=6) labelattrs=(size=22pt) label="REPORTa 2017 D";
legendItem type=Line name="REPORTb 2017 A"/lineattrs=(color=CX006E51 pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 A";
legendItem type=Line name="REPORTb 2017 B"/lineattrs=(color=CX0069AA pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 B";
legendItem type=Line name="REPORTb 2017 C"/lineattrs=(color=CF58025 pattern=Dash THICKNESS=6) labelattrs=(size=22pt) label="REPORTb 2017 C";

```

Page Break

```

Line_seriesplot
seriesplot x=xaxis y=Historical_Historical_Historical/LEGENDLABEL="Historical" PRIMARY=TRUE lineattrs=(color=CX000000 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_A_aaa/LEGENDLABEL="REPORTa 2017 A" PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_B_aaa/LEGENDLABEL="REPORTa 2017 B" PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_C_aaa/LEGENDLABEL="REPORTa 2017 C" PRIMARY=TRUE lineattrs=(color=CF58025 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTa2017_D_aaa/LEGENDLABEL="REPORTa 2017 D" PRIMARY=TRUE lineattrs=(color=CX7030A0 pattern=solid THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_A_aaa/LEGENDLABEL="REPORTb 2017 A" PRIMARY=TRUE lineattrs=(color=CX006E51 pattern=Dash THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_B_aaa/LEGENDLABEL="REPORTb 2017 B" PRIMARY=TRUE lineattrs=(color=CX0069AA pattern=Dash THICKNESS=6);
seriesplot x=xaxis y=REPORTb2017_C_aaa/LEGENDLABEL="REPORTb 2017 C" PRIMARY=TRUE lineattrs=(color=CF58025 pattern=Dash THICKNESS=6);

```

Line_discretelegend
"Historical"
"REPORTa 2017 A"
"REPORTa 2017 B"
"REPORTa 2017 C"
"REPORTa 2017 D"
"REPORTb 2017 A"
"REPORTb 2017 B"
"REPORTb 2017 C"

Page Break

Title
Title1

Replace the hard coding part with the macro parameters, we got:

```

proc template;
define statgraph class1;
begingraph;
entrytitle "&Title" / border=false textattrs=(size=30pt);
discreteattrvar attrvar=classfill var=fore_sce attrmap='colors';
&Line_legendItem; /*legendItem statement*/
legendItem type=line name="blank" /lineattrs=(color=white)
  labelattrs=(size=&font_size) label=" ";
layout overlay /border=false WALLCOLOR=white WALLDISPLAY=(FILL)
xaxisopts=(display=all linearopts=(origin=0 THRESHOLDMIN=1 THRESHOLDMAX=1)
  label=' ' griddisplay=off labelattrs=(weight=bold size=&font_size)
  tickvalueattrs=(weight=bold size=&font_size)
  discreteopts=(tickdisplaylist=(&xaxis_tickdisplaylist)
  tickvaluelist=(&xaxis_tickvaluelist) TICKVALUEFITPOLICY=ROTATEALWAYS
  TICKVALUEROTATION=VERTICAL))
yaxisopts=(display=(label ticks tickvalues) label=' ' labelattrs=(weight=bold
  size=&font_size) griddisplay=on tickvalueattrs=(weight=bold size=&font_size)
  linearopts=(integer=FALSE origin=&lower_band viewmin=&lower_band
  tickvaluelist=(&yaxis_tickvaluelist) TICKVALUEPRIORITY=TRUE
  tickvalueformat=DATA) gridattrs=(pattern=dash THICKNESS=2 color=darkgrey));
&Line_seriesplot; /*seriesplot statement*/
discretelegend &Line_discretelegend /*discretelegend values*/
  /border=false across=4 autoalign=(BOTTOMLEFT) ORDER=ROWMAJOR;

referenceline y=0/ curvelabel="&curvelb." curvelabelattrs=(size=&font_size
  weight=bold) clip=true curvelabelposition=min
  lineattrs=(pattern=dash THICKNESS=2 color=darkgrey) ;
endlayout;
entryfootnote halign=left TEXTATTRS=(size=20pt weight=bold) "Type=&Type"
  halign=right TEXTATTRS=(size=20pt weight=bold) "Publish=&No";

```

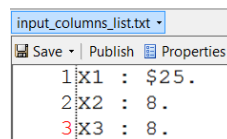


```
endgraph;
end;
run;
```

Other options such as [xaxisopts](#), [yaxisopts](#), and statements like [referenceline](#), [entryfootnote](#) can also be generated by SAS codes too if they have changing parameters.

### WAY3: To automatically cover the random changing of the number of variables in large data files

We can easily use data step to create data step input columns with their length, format, informat, retain column order, then proc export as txt file like `input_columns_list.txt` file in `&D_TYPE` folder:



```
input_columns_list.txt
Save Publish Properties
1X1 : $25.
2X2 : 8.
3X3 : 8.
```

Then using the following example SAS code to read them into data step input and retain statements:

```
/*path of data order and type used for reading .txt files into SAS as codes*/
FILENAME COLLST "&D_TYPE/" ;
%macro read_retain_txt; %include COLLST("retain_columns_list.txt"); %mend read_retain_txt;
%macro read_input_txt; %include COLLST("input_columns_list.txt"); %mend read_input_txt;

options mprint mlogic symbolgen ;
data O_DATA.&Date_Prefix._mll_raw_infile;
retain %read_retain_txt; ;
infile "&I_DATA/&data_filename..txt" dlm='|' dsd missover trunccover firstobs=2 lrecl=32767;
input %read_input_txt; ;
run;
```

This way is extremely helpful for us to handle the large data files with hundreds or thousands variables with different format, informat, input columns with their lengths.

In my paper name as *Comparisons of two large complex data sets with matching common dimensions but altered data to catch any changes using SAS Enterprise Guide*, we can see another similar SAS code generation example:

```
proc export data=cur_prv_diff(obs=0 drop=T0 PQ: _TYPE_ _OBS_)
outfile="&D_TEMP./used_for_srted.txt" dbms=csv replace;
delimiter=" "; putnames=YES;
run;

/*define the txt file inside with common columns as sort keys using macro*/
filename COLLST "&D_TEMP./";
%macro used_for_srted_txt;
%include COLLST("used_for_srted.txt");
%mend used_for_srted_txt;

/*find common columns and compare them*/
data find_commonkeys;
retain %used_for_srted_txt; ;
merge cur_srt(drop=T0 PQ: in=a) prv_srt(drop=T0 PQ: in=b);
by _ALL_ ; if a=1 and b=1 ;
run;

proc sort data=find_commonkeys;
by %used_for_srted_txt;
run;
```

## CONTACT INFORMATION

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

Kaiqing Fan  
Sr. Data Scientist  
Sr. SAS Developer Lead  
Sr. Risk Predictive Modeler  
Mastech Digital Inc.  
Address: 6750 Miller Road, Brecksville, OH-44141  
Mobile: 504.344.7267  
Email: fankaiqinguw@gmail.com  
Linkedin: <https://www.linkedin.com/in/fan-kaiqing-81776940/>

Jing Wang  
Sr. SAS Programmer  
Sr. Data Scientist  
China Minsheng Bank

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.