



Building of SQL queries using macro patterns

Yehor Shakh

Introduction

Sometimes SQL queries may be large and complicated.

Sometimes a person needs to update this code...

After someone...

Who didn't even think about a good formatting of the code.

And it must be done ASAP.

If you can imagine such a situation - then welcome.

I suggest to encapsulate the SQL queries into a pattern and use it just like a “black box” – calling it and change only key words.
Making the queries shorter, more readable and easier to update.

Let's see how it works...

What are the macro patterns?

Time savings and consistency

```
1 %macro select( select= /*SELECT statement - REQUIRED*/,
2               from=    /*FROM statement - REQUIRED*/,
3               where=    /*WHERE statement - OPTIONAL*/,
4               groupby=  /*GROUP BY statement - OPTIONAL*/,
5               when=     /*WHEN statement - OPTIONAL*/,
6               orderby=  /*ORDER BY statement - OPTIONAL*/
7               )
8     SELECT &select
9     FROM &from
10        %if &where ne %then %do; WHERE &where %end;
11
12        %if &groupby ne %then %do; GROUP BY &groupby %end;
13        %if &when ne %then %do; WHEN &when %end;
14        %if &orderby ne %then %do; ORDER BY &groupby %end;
15 %mend select;
16
17 proc sql;
18     %select( /*<your parameters here>*/ );
19 quit;
```

You may adjust %select macro with any SQL statements.

Note! Do not close your query with a semicolon ";" inside the macro – so that you may expand the SQL code at the place of calling it

Example of using %select

```
19 proc sql;  
20     %select(select=%str(*), from=sashelp.class);  
21 quit;
```

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84
John	M	12	59	99.5
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90
Louise	F	12	56.3	77
Mary	F	15	66.5	112
Philip	M	16	72	150
Robert	M	12	64.8	128
Ronald	M	15	67	133
Thomas	M	11	57.5	85
William	M	15	66.5	112

Macro pattern is like a “black box” a tip which allows you to incapsulate SQL queries into a “black box”, just calling them with your parameters whenever you need

Using %select multiple times within a single Proc SQL

```
19 proc sql;  
20   %select(    select= %str( sex, count(Name) as Count ),  
21             from= sashelp.class,  
22             groupby= sex  
23             );  
24  
25   %select(    select= %str( age, count(Name) as Count ),  
26             from= sashelp.class,  
27             groupby= age  
28             );  
29  
30   %select(    select= %str( sex, age, count(Name) as Count ),  
31             from= sashelp.class,  
32             groupby= %str( sex, age )  
33             );  
34 quit;
```

Sex	Count
F	9
M	10

Age	Count
11	2
12	5
13	3
14	4
15	4
16	1

Sex	Age	Count
F	11	1
F	12	2
F	13	2
F	14	2
F	15	2
M	11	1
M	12	3
M	13	1
M	14	2
M	15	2
M	16	1

Using of UNION

```
20 proc sql;  
21     %select(select=%str("Sex" as GRP,  
22                     Sex,  
23                     count(Name) as Count ),  
24     from=sashelp.class,  
25     groupby= %str( GRP, Sex )  
26     )  
27  
28     union  %select(select=%str("Age" as GRP,  
29                     strip(put(Age, best.)),  
30                     count(Name) as Count ),  
31     from=sashelp.class,  
32     groupby= %str( Age )  
33     );  
34 quit;
```

GRP	Sex	Count
Age	11	2
Age	12	5
Age	13	3
Age	14	4
Age	15	4
Age	16	1
Sex	F	9
Sex	M	10

Note! You may expand the first query thanks to not ending it with a semicolon “;” inside the macro.

Joining of datasets

```
19 %macro join(    DSName /*Name of the dataset to join - REQUIRED*/,
20                where= /*WHERE statement - OPTIONAL*/,
21                join=%str(left join)/*Type of JOIN - OPTIONAL, 'left join' by default*/,
22                on= /*ON statment of JOIN - REQUIRED*/
23                );
24    &join &DSName
25    %if &where ne %then %do; where &where %end;
26    on &on
27 %mend join;
```

Using %select multiple times within a single Proc SQL

```
50 proc sql noprint;
51   create table Height as
52     %select( select=%str( Name, Sex, Age, Height ), from=sashelp.class, where= %str( age=12 ) );
53
54   create table Weight as
55     %select( select=%str( Name, Sex, Age, Weight ), from=sashelp.class, where= %str( age=12 ) );
56 quit;
57
58 proc sql;
59   %select( select=%str( t1.Name, t1.Sex, t1.Age, t1.Height, t2.Weight ),
60     from=%str( Height as t1 ) )
61     %join( %str( Weight as t2 ),
62       on=%str( t1.Name = t2.Name ) );
63   ;
64 quit;
```

Name	Sex	Age	Height	Weight
James	M	12	57.3	83
Jane	F	12	59.8	84.5
John	M	12	59	99.5
Louise	F	12	56.3	77
Robert	M	12	64.8	128

Creating a table

Let's create a table from *sashelp.heart* dataset

Obs	Status	DeathCause	AgeCHDdiag	Sex	AgeAtStart	Height	Weight	Diastolic	Systolic	MRW	Smoking	AgeAtDeath	Cholesterol	Chol_Status	BP_Status	Weight_Status	Smoking_Status
1	Dead	Other	.	Female	29	62.50	140	78	124	121	0	55	.		Normal	Overweight	Non-smoker
2	Dead	Cancer	.	Female	41	59.75	194	92	144	183	0	57	181	Desirable	High	Overweight	Non-smoker
3	Alive		.	Female	57	62.25	132	90	170	114	10	.	250	High	High	Overweight	Moderate (6-15)
4	Alive		.	Female	39	65.75	158	80	128	123	0	.	242	High	Normal	Overweight	Non-smoker
5	Alive		.	Male	42	66.00	156	76	110	116	20	.	281	High	Optimal	Overweight	Heavy (16-25)
6	Alive		.	Female	58	61.75	131	92	176	117	0	.	196	Desirable	High	Overweight	Non-smoker
7	Alive		.	Female	36	64.75	136	80	112	110	15	.	196	Desirable	Normal	Overweight	Moderate (6-15)
8	Dead	Other	.	Male	53	65.50	130	80	114	99	0	77	276	High	Normal	Normal	Non-smoker
9	Alive		.	Male	35	71.00	194	68	132	124	0	.	211	Borderline	Normal	Overweight	Non-smoker
10	Dead	Cerebral Vascular Disease	.	Male	52	62.50	129	78	124	106	5	82	284	High	Normal	Normal	Light (1-5)
11	Alive		.	Male	39	66.25	179	76	128	133	30	.	225	Borderline	Normal	Overweight	Very Heavy (> 25)

Table 1

Variable	Value	Male	Female	Overall
<i>Blood Pressure Status</i>	<i>High</i>	XXX	XXX	XXX
	<i>Normal</i>	XXX	XXX	XXX
	<i>Optimal</i>	XXX	XXX	XXX
<i>Cholesterol Status</i>	...	XXX	XXX	XXX
<i>Death Cause</i>	...	XXX	XXX	XXX
<i>Smoking Status</i>	...	XXX	XXX	XXX
<i>Status</i>	...	XXX	XXX	XXX
<i>Weight Status</i>	...	XXX	XXX	XXX

Creating a table

6 unions in 6 rows

```
1 *SQL query macro;
2 %macro select1(grpn, grp1, grp2,
3   from= sashelp.heart,
4   where= %str( &grp2 is not NULL ),
5   groupby= %str( GRP1, GRP2 ),
6   having= %str( GRP2 is not NULL ),
7   orderby=
8   );
9   select  &grpn as GRPn,
10          &grp1 as GRP1,
11          &grp2 as GRP2,
12          count(case when Sex="Male" then Sex else "" end) as Male,
13          count(case when Sex="Female" then Sex else "" end) as Female,
14          count(Sex) as Overall
15   from &from
16   %if &where ne %then %do; where &where %end;
17   %if &groupby ne %then %do; group by &groupby %end;
18   %if &having ne %then %do; having &having %end;
19   %if &orderby ne %then %do; order by &orderby %end;
20 %mend select1;
```

“Black box”

```
22 *run Proc SQL using the query macro;
23 proc sql noprint;
24   create table query as
25     %select1( 1, "Status",          Status)
26     union %select1( 2, "Death Cause", DeathCause )
27     union %select1( 3, "Cholesterol Status", Chol_status )
28     union %select1( 4, "Blood Pressure Status", BP_status )
29     union %select1( 5, "Weight Status", Weight_status )
30     union %select1( 6, "Smoking Status", Smoking_status )
31
32     order by GRPn
33   ;
34 quit;
35
36 proc print data=query;
37   id GRP1;
38   var GRP2 Male Female Overall;
39 run;
```

GRP1	GRP2	Male	Female	Overall
Status	Alive	1241	1977	3218
Status	Dead	1095	896	1991
Death Cause	Cancer	283	256	539
Death Cause	Cerebral Vascular Disease	179	199	378
Death Cause	Coronary Heart Disease	396	209	605
Death Cause	Other	181	176	357
Death Cause	Unknown	56	56	112
Cholesterol Status	Borderline	902	959	1861
Cholesterol Status	Desirable	600	805	1405
Cholesterol Status	High	781	1010	1791
Blood Pressure Status	High	1081	1186	2267
Blood Pressure Status	Normal	977	1166	2143
Blood Pressure Status	Optimal	278	521	799
Weight Status	Normal	626	846	1472
Weight Status	Overweight	1643	1907	3550
Weight Status	Underweight	65	116	181
Smoking Status	Heavy (16-25)	707	339	1046
Smoking Status	Light (1-5)	157	422	579
Smoking Status	Moderate (6-15)	236	340	576
Smoking Status	Non-smoker	819	1682	2501
Smoking Status	Very Heavy (> 25)	398	73	471

Creating a table

The more complicated is structure of SQL query – the more effective is using of SQL patterns

Table 2 - Adverse Events by SOC, PT and maximum grade

Primary system organ class Preferred term	TRT_A	TRT_B	TRT_C
Maximum grade			
Number of subjects with at least one event	xxx	xxx	xxx
Grade 1	xxx	xxx	xxx
Grade 2	xxx	xxx	xxx
Grade 3	xxx	xxx	xxx
Grade 4	xxx	xxx	xxx
Grade 5	xxx	xxx	xxx
Missing	xxx	xxx	xxx
<AEBODSYS>			
Grade 1	xxx	xxx	xxx
Grade 2	xxx	xxx	xxx
Grade 3	xxx	xxx	xxx
Grade 4	xxx	xxx	xxx
Grade 5	xxx	xxx	xxx
Missing	xxx	xxx	xxx
<AEBODSYS>			
<AEDECOD>			
Grade 1	xxx	xxx	xxx
Grade 2	xxx	xxx	xxx
Grade 3	xxx	xxx	xxx
Grade 4	xxx	xxx	xxx
Grade 5	xxx	xxx	xxx
Missing	xxx	xxx	xxx

```

1 proc sql noprint;
2   create table TOXGR1 as
3     %select2(1, "Number of subjects with at least one event", "", "")
4     union %select2(1, "Number of subjects with at least one event", "",
5       tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"), count=MaxTOXGR,
6       from= %nrstr( ( select TRT, SUBJECT, max(AETOXGR) as MaxTOXGR
7         from dummyData where AEDECOD is not NULL
8         group by TRT, SUBJECT ) ),
9       groupby = %str(TRT, MaxTOXGR))
10
11   union %select2(2, AEBODSYS, "", "", where=%str(AEBODSYS is not NULL))
12   union %select2(2, AEBODSYS, "",
13     tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"), count=MaxTOXGR,
14     from= %nrstr( ( select TRT, AEBODSYS, SUBJECT, max(AETOXGR) as MaxTOXGR
15       from dummyData where AEDECOD is not NULL
16       group by TRT, AEBODSYS, SUBJECT ) ),
17     groupby = %str(TRT, AEBODSYS, MaxTOXGR), where=%str(AEBODSYS is not NULL))
18
19   union %select2(2, AEBODSYS, AEDECOD, "", where=%str(AEDECOD is not NULL))
20   union %select2(2, AEBODSYS, AEDECOD,
21     tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"), count=MaxTOXGR,
22     from= %nrstr( ( select TRT, AEBODSYS, AEDECOD, SUBJECT, max(AETOXGR) as MaxTOXGR
23       from dummyData where AEDECOD is not NULL
24       group by TRT, AEBODSYS, AEDECOD, SUBJECT ) ),
25     groupby = %str(TRT, AEBODSYS, AEDECOD, MaxTOXGR), where=%str(AEDECOD is not NULL))
26
27   order by GRPn, GRP1, GRP2, GRP3
28   ;
29   quit;
30
31
32 proc transpose data=TOXGR1 out=TOXGR2 name=TRT;
33   by GRPn GRP1 GRP2 GRP3;
34   id TRT;
35   var CNT;
36 run;
37
38 proc print data=TOXGR2;
39   var TRT_A TRT_B TRT_C;
40   id GRPn GRP1 GRP2 GRP3;
41   title2 "Table 2 - Adverse Events by SOC, PT and maximum grade";
42 run;

```

```

from= %nrstr( ( select TRT, AEBODSYS, AEDECOD, SUBJECT, max(AETOXGR) as MaxTOXGR
  from dummyData where AEDECOD is not NULL
  group by TRT, AEBODSYS, AEDECOD, SUBJECT ) ),

```

Table 2 - Adverse Events by SOC, PT and maximum grade

GRP1	GRP2	GRP3	TRT_A	TRT_B	TRT_C
Number of subjects with at least one event			165	178	157
Number of subjects with at least one event		1	27	28	20
Number of subjects with at least one event	2		47	48	42
Number of subjects with at least one event		3	44	44	44
Number of subjects with at least one event	4		11	23	18
Number of subjects with at least one event		5	12	13	5
Number of subjects with at least one event		Missing	0	0	0
CARDIAC			62	77	70
CARDIAC		1	21	21	23
CARDIAC		2	15	27	22
CARDIAC		3	16	13	16
CARDIAC		4	4	10	5
CARDIAC		5	4	3	2
CARDIAC		Missing	0	0	0
CARDIAC	ATRIAL F		4	9	8
CARDIAC	ATRIAL F	1	1	3	2
CARDIAC	ATRIAL F	2	1	4	1
CARDIAC	ATRIAL F	3	1	.	3
CARDIAC	ATRIAL F	4	1	.	1
CARDIAC	ATRIAL F	Missing	.	0	0
CARDIAC	ATRIAL H		5	7	7
CARDIAC	ATRIAL H	1	1	3	3
CARDIAC	ATRIAL H	2	2	3	2
CARDIAC	ATRIAL H	3	1	1	2
CARDIAC	ATRIAL H	5	1	.	.
CARDIAC	ATRIOVEN		6	9	10
CARDIAC	ATRIOVEN	1	.	2	5
CARDIAC	ATRIOVEN	2	4	4	4
CARDIAC	ATRIOVEN	3	1	3	1
CARDIAC	ATRIOVEN	4	1	.	.
CARDIAC	BRADYCAR		.	4	5
CARDIAC	BRADYCAR	1	.	1	3
CARDIAC	BRADYCAR	2	.	2	2
CARDIAC	BRADYCAR	5	.	1	.
CARDIAC	BUNDLE B		10	6	9
CARDIAC	BUNDLE B	1	2	2	3
CARDIAC	BUNDLE B	2	2	1	3
CARDIAC	BUNDLE B	3	6	.	2

In this example we fill the UNION of 6 queries, 3 of which include subqueries just in 20 rows of code - keeping it readable and easy-to-update



Let your programs be short, readable
and easy-to-update

Yehor Shakh
Sr. Statistical Programmer