Paper CT07

Building of SQL queries using macro patterns

Yehor Shakh, IQVIA, Kyiv, Ukraine

ABSTRACT

Creating a universal macro pattern for complicated multi-level SQL queries makes them shorter, more readable, easier to update and enables the Programmer to write code quicker.

The use of a macro pattern provides an opportunity to prepare a dataset for Proc Report by doing a large SQL query, keeping the code short, readable and easy-to update – calling the macro inside the query multiple times: sequentially or in a loop. This feature allows you to create multi-level summary reports in one step. The paper will demonstrate the use of a template to create these reports, through real life examples.

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 incapsulate the SQL queries into a pattern and use it just like a "black box" – calling it and change only the key words, making the queries shorter, more readable, and easier to update.

WHAT ARE THE MACRO PATTERNS?

The major idea is to create a macro, which contains all SQL statements and call it inside the **proc sql**.

```
%macro select(      select=      /*SELECT statement - REQUIRED*/,
                    from=        /*FROM statement - REQUIRED*/,
                    where=       /*WHERE statement - OPTIONAL*/,
                    groupby=     /*GROUP BY statement - OPTIONAL*/,
                    when=        /*WHEN statement - OPTIONAL*/,
                    orderby=     /*ORDER BY statement - OPTIONAL*/
                    );
      SELECT &select
      FROM &from
            %if &where ne %then %do; WHERE &where %end;

      %if &groupby ne %then %do; GROUP BY &groupby %end;
      %if &when ne %then %do; WHEN &when %end;
      %if &orderby ne %then %do; ORDER BY &groupby %end;
%mend select;
```

You may adjust %select macro with any SQL statements.
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.

EXAMPLES OF USING %SELECT

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

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

The result is available in Appendix 1, Table1

You may call the macro multiple times within one **proc sql** procedure to create separate datasets.

```
proc sql;
      %select(      select= %str( sex, count(Name) as Count ),
                    from= sashelp.class,
                    groupby= sex
                    );

      %select(      select= %str( age, count(Name) as Count ),
                    from= sashelp.class,
                    groupby= age
                    );

      %select(      select= %str( sex, age, count(Name) as Count ),
                    from= sashelp.class,
                    groupby= %str( sex, age )
                    );
quit;
```

The result is available in Appendix 1, Tables 2, 3, 4

Or union the queries to create a single one by calling the macro multiple times.

```
proc sql;
            %select(      select= %str("Sex" as GRP1,
                                  sex as GRP2,
                                  count(Name) as Count
                                  ),
                          from= sashelp.class,
                          groupby= %str(GRP1, GRP2)
                          )

      union  %select(     select= %str("Age" as GRP1,
                                  put(age, best.) as GRP2,
                                  count(Name) as Count
                                  ),
                          from= sashelp.class,
                          groupby= %str(GRP1, GRP2)
                          )
      ;
quit;
```

The result is available in Appendix 1, Table 5

CREATING A TABLE USING MACRO PATTERNS

CREATING A SIMPLE TABLE

Here is a standard **sashelp.heart** dataset:

Image 1. Standard 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)

Let us create a table using the following mock:

Image 2. Table 6 mock.

Table 6. Standard sashelp.heart dataset.

Variable	Value	Male	Female	Overall
Blood Pressure Status	*High*	*xxx*	*xxx*	*xxx*
	Normal	*xxx*	*xxx*	*xxx*
	Optimal	*xxx*	*xxx*	*xxx*
Cholesterol Status	...	*xxx*	*xxx*	*xxx*
Death Cause	...	*xxx*	*xxx*	*xxx*
Smoking Status	...	*xxx*	*xxx*	*xxx*
Status	...	*xxx*	*xxx*	*xxx*
Weight Status	...	*xxx*	*xxx*	*xxx*

Defining a macro:

```
*SQL query macro;
%macro select1(     grpn, grp1, grp2,
                    from= sashelp.heart,
                    where= %str( &grp2 is not NULL ),
                    groupby= %str( GRP1, GRP2 ),
                    having= %str( GRP2 is not NULL ),
                    orderby=
                    );
       select &grpn as GRPn,
              &grp1 as GRP1,
              &grp2 as GRP2,
              count(case when Sex="Male" then Sex else "" end) as Male,
              count(case when Sex="Female" then Sex else "" end) as Female,
              count(Sex) as Overall
       from &from
       %if &where ne %then %do; where &where %end;
       %if &groupby ne %then %do; group by &groupby %end;
       %if &having ne %then %do; having &having %end;
       %if &orderby ne %then %do; order by &orderby %end;
%mend select1;
```

Calling the macro in **proc sql** and **proc print** the result:

```
*run Proc SQL using the query macro;
proc sql noprint;
       create table query as
                     %select1(     1, "Status",                    Status)
              union  %select1(     2, "Death Cause",               DeathCause )
              union  %select1(     3, "Cholesterol Status",        Chol_status )
              union  %select1(     4, "Blood Pressure Status",     BP_status )
              union  %select1(     5, "Weight Status",             Weight_status )
              union  %select1(     6, "Smoking Status",            Smoking_status )

              order by GRP1
              ;
quit;

proc print data=query;
       id GRP1;
       var GRP2 Male Female Overall;
run;
```

Table 6.

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 TYPICAL AE TABLE

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

In next example we will create a typical AE table by SOC, PT and maximum toxicity grade. This table has a multi-level cascade structure.

We will use a dummy data. You can find a program for generating the dummy data and AE table in Appendix 2.

Image 3. Dummy data

	SUBJECT	AGE	TRT	AGEGRP	AEBODSYS	AEDECOD	AETOXGR
1	10001	66	TRT_A	50-75 years	EAR AND LABYRINTH DISORDERS	EAR PAIN	1
2	10002	49	TRT_B	35-50 years	CARDIAC DISORDERS	WOLFF-PARKINSON-WHITE SYNDROME	3
3	10002	49	TRT_B	35-50 years	EYE DISORDERS	CONJUNCTIVITIS	2
4	10002	49	TRT_B	35-50 years	GASTROINTESTINAL DISORDERS	CONSTIPATION	2
5	10002	49	TRT_B	35-50 years	GASTROINTESTINAL DISORDERS	FLATULENCE	1
6	10003	21	TRT_C	18-35 years	EAR AND LABYRINTH DISORDERS	CERUMEN IMPACTION	3
7	10003	21	TRT_C	18-35 years	EAR AND LABYRINTH DISORDERS	TINNITUS	3
8	10003	21	TRT_C	18-35 years	EAR AND LABYRINTH DISORDERS	VERTIGO	2
9	10003	21	TRT_C	18-35 years	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	1
10	10004	83	TRT_C	75+ years	EYE DISORDERS	VISION BLURRED	2
11	10005	61	TRT_C	50-75 years	EYE DISORDERS	EYE ALLERGY	.
12	10005	61	TRT_C	50-75 years	EYE DISORDERS	GLAUCOMA	2
13	10005	61	TRT_C	50-75 years	GASTROINTESTINAL DISORDERS	GLOSSITIS	.
14	10006	29	TRT_C	18-35 years	EAR AND LABYRINTH DISORDERS	EAR PAIN	2
15	10007	35	TRT_C	35-50 years	CARDIAC DISORDERS	CARDIAC DISORDER	3
16	10007	35	TRT_C	35-50 years	GASTROINTESTINAL DISORDERS	HIATUS HERNIA	3
17	10008	71	TRT_C	50-75 years	CARDIAC DISORDERS	SUPRAVENTRICULAR TACHYCARDIA	3
18	10008	71	TRT_C	50-75 years	EAR AND LABYRINTH DISORDERS	EAR PAIN	3
19	10008	71	TRT_C	50-75 years	EYE DISORDERS	EYE SWELLING	4
20	10009	42	TRT_C	35-50 years	EAR AND LABYRINTH DISORDERS	CERUMEN IMPACTION	2

Image 4. AE table mock.

Table 7 - Adverse Events by SOC, PT and maximum grade

Primary system organ class *Preferred term* *Maximum grade*	*TRT_A*	*TRT_B*	*TRT_C*
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>	*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>	*xxx*	*xxx*	*xxx*
<AEDECOD>	*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*

Defining a macro:

```
*SQL querry macro;
%macro select2(     grpn, grp1, grp2, grp3,
                    count= %str( distinct SUBJECT ),
                    from= dummyData,
                    where=,
                    groupby= %str( TRT, GRP1, GRP2 ),
                    having=,
                    orderby=
                    );
       select TRT,
              &grpn as GRPn,
              &grp1 as GRP1,
              &grp2 as GRP2,
              &grp3 as GRP3,
              count(&count) as CNT
       from &from
       %if &where ne %then %do; where &where %end;
       %if &groupby ne %then %do; group by &groupby %end;
       %if &having ne %then %do; having &having %end;
       %if &orderby ne %then %do; order by &orderby %end;
%mend select2;
```

Calling the macro in **proc sql:**

Image 5. Calling AE macro

```
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;
```

Transposing and outputting the table:

Image 6. Proc report.

```
proc transpose data=TOXGR1 out=TOXGR2 name=TRT;
    by GRPn GRP1 GRP2 GRP3;
    id TRT;
    var CNT;
run;

proc report data=TOXGR2;
    column GRPn GRP1 GRP2 GRP3 TRT_A TRT_B TRT_C;

    define GRPn /order order=data noprint;
    define GRP1 /order order=data display "SOC";
    define GRP2 /order order=data display "PT";
    define GRP3 /order order=data display "Grades";
    define TRT_A /order order=data;
    define TRT_B /order order=data;
    define TRT_C /order order=data;

    title1 "Table 2 - Adverse Events by SOC, PT and maximum grade";
run;
```

Table output:

Image 7. AE table output.

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

GRP1	GRP2	GRP3	TRT_A	TRT_B	TRT_C
CARDIAC DISORDERS	ATRIAL FIBRILLATION	1	1	2	1
	ATRIAL FLUTTER	1	2	1	2
		2	2	2	5
	ATRIAL HYPERTROPHY	2	2	1	2
	ATRIOVENTRICULAR BLOCK SECOND DEGREE	1	2	1	1
	BUNDLE BRANCH BLOCK LEFT	2	2	1	3
		3	1	1	1
	CARDIAC DISORDER	1	1	1	1
		2	3	1	2
	MYOCARDIAL INFARCTION	1	1	2	1
		3	1	1	1
	SUPRAVENTRICULAR TACHYCARDIA	3	2	1	1
	TACHYCARDIA	1	4	1	1
	WOLFF-PARKINSON-WHITE SYNDROME	1	1	3	2
		2	2	2	1
EAR AND LABYRINTH DISORDERS	CERUMEN IMPACTION	1	1	4	7
		2	11	5	2
		3	6	2	6
		5	1	1	1
	EAR PAIN	1	6	7	5
		2	5	8	5
		3	4	4	7
		Missing	0	0	0
	TINNITUS	1	8	4	5

CONCLUSION

The use of a macro pattern provides an opportunity to prepare a dataset for Proc Report by doing a large SQL query, keeping the code short, readable and easy-to update – calling the macro inside the query multiple times: sequentially or in a loop. This feature allows you to create multi-level summary reports in one step.

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

CONTACT INFORMATION

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

Yehor Shakh
IQVIA
12, M. Amosova Str.
Kyiv / 03038
Work Phone: +380 (44) 49 680 29
Email: yehor.shakh@iqvia.com
Web: www.iqvia.com/

Personal email: yegor.shakh@gmail.com
Personal mobile: +380 (63) 25 191 77

APPENDIX 1. TABLES

TABLE 1

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

TABLES 2, 3, 4

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

TABLE 5

GRP1	GRP2	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

APPENDIX 2. AE TABLE PROGRAM

GENERATE DUMMY DATA

```
data dummyDM (keep= SUBJECT--AGEGRP);
	length SUBJECT AGE 8 TRT $5 AGEGRP $20;
	call missing(of _ALL_);

	array TRTs{3} $ ("TRT_A", "TRT_B", "TRT_C");

	do SUBJECT=10001 to 10500;
		id= rand('integer', 1, dim(TRTs));
		TRT= TRTs[id];

		AGE= rand('integer', 18, 85);
		select;
			when(18 <= AGE < 35) AGEGRP="18-35 years";
			when(35 <= AGE < 50) AGEGRP="35-50 years";
			when(50 <= AGE < 75) AGEGRP="50-75 years";
			when(AGE >= 75) AGEGRP="75+ years";
			otherwise;
			end;

		output;
		end;
run;

data dummyAE (keep= SUBJECT--AETOXGR);
	length SUBJECT 8 AEBODSYS AEDECOD $200 AETOXGR 8;
	call missing(of _ALL_);

	array a{4} $ (	"CARDIAC DISORDERS", "EAR AND LABYRINTH DISORDERS", "EYE
DISORDERS", "GASTROINTESTINAL DISORDERS");

	array b1{20} $	(	"ATRIAL FIBRILLATION",
				"ATRIAL FLUTTER",
				"ATRIAL HYPERTROPHY",
				"ATRIOVENTRICULAR BLOCK FIRST DEGREE",
				"ATRIOVENTRICULAR BLOCK SECOND DEGREE",
				"BRADYCARDIA",
				"BUNDLE BRANCH BLOCK LEFT",
				"BUNDLE BRANCH BLOCK RIGHT",
				"CARDIAC DISORDER",
				"CARDIAC FAILURE CONGESTIVE",
				"MYOCARDIAL INFARCTION",
				"PALPITATIONS",
				"SINUS ARRHYTHMIA",
				"SINUS BRADYCARDIA",
				"SUPRAVENTRICULAR EXTRASYSTOLES",
				"SUPRAVENTRICULAR TACHYCARDIA",
				"TACHYCARDIA",
				"VENTRICULAR EXTRASYSTOLES",
				"VENTRICULAR HYPERTROPHY",
				"WOLFF-PARKINSON-WHITE SYNDROME"
				);

	array b2{4} $	(	"CERUMEN IMPACTION",
				"EAR PAIN",
				"TINNITUS",
				"VERTIGO"
				);

	array b3{7} $	(	"CONJUNCTIVAL HAEMORRHAGE",
				"CONJUNCTIVITIS",
```

```
                                "EYE ALLERGY",
                                "EYE PRURITUS",
                                "EYE SWELLING",
                                "GLAUCOMA",
                                "VISION BLURRED"
                                );

        array b4{16} $      (       "ABDOMINAL DISCOMFORT",
                                "ABDOMINAL PAIN",
                                "CONSTIPATION",
                                "DIARRHOEA",
                                "DYSPEPSIA",
                                "DYSPHAGIA",
                                "FLATULENCE",
                                "GASTROINTESTINAL HAEMORRHAGE",
                                "GASTROOESOPHAGEAL REFLUX DISEASE",
                                "GLOSSITIS",
                                "HIATUS HERNIA",
                                "NAUSEA",
                                "RECTAL HAEMORRHAGE",
                                "SALIVARY HYPERSECRETION",
                                "STOMACH DISCOMFORT",
                                "VOMITING"
                                );

        do i=1 to 1000;
                call missing(SUBJECT, AEBODSYS, AEDECOD);
                SUBJECT= rand('integer', 10001, 10500);

                a_id= rand('integer', 1, dim(a));
                AEBODSYS= a[a_id];

                select(a_id);
                        when(1) AEDECOD= b1[rand('integer', 1, dim(b1))];
                        when(2) AEDECOD= b2[rand('integer', 1, dim(b2))];
                        when(3) AEDECOD= b3[rand('integer', 1, dim(b3))];
                        when(4) AEDECOD= b4[rand('integer', 1, dim(b4))];
                        end;

                AETOXGR= int(rand('GAMMA',5, 0.5));
                        if AETOXGR=0 then AETOXGR=.;
                        if AETOXGR>5 then AETOXGR=5;

                output;
                end;
run;

proc sql;
        create table dummyData as
                select dm.*,
                        ae.AEBODSYS, ae.AEDECOD, ae.AETOXGR
                from dummyDM dm
                left join dummyAE ae
                        on dm.SUBJECT = ae.SUBJECT

                order by SUBJECT, AEBODSYS, AEDECOD
                ;
quit;
```

CREATE AE TABLE

```
proc sql noprint;
create table TOXGR1 as
             %select2(     1,
                           "Number of subjects with at least one event",
                           "",
                           ""
                           )

       union  %select2(     1,
                           "Number of subjects with at least one event",
                           "",
                           tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"),

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

       union  %select2(     2,
                           AEBODSYS,
                           "",
                           "",

                           where=%str(AEBODSYS is not NULL)
                           )


       union  %select2(     2,
                           AEBODSYS,
                           "",
                           tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"),

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

       union  %select2(     2,
                           AEBODSYS,
                           AEDECOD,
                           "",

                           where=%str(AEDECOD is not NULL)
                           )


       union  %select2(     2,
                           AEBODSYS,
                           AEDECOD,
                           tranwrd(strip(put(MaxTOXGR, best.)), ".", "Missing"),

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

```
                                    group by TRT, AEBODSYS, AEDECOD, SUBJECT)),
                              groupby = %str(TRT, AEBODSYS, AEDECOD, MaxTOXGR),
                              where= %str(AEDECOD is not NULL)
                              )

      order by GRPn, GRP1, GRP2, GRP3
      ;
quit;

proc transpose data=TOXGR1 out=TOXGR2 name=TRT;
      by GRPn GRP1 GRP2 GRP3;
      id TRT;
      var CNT;
run;

proc report data=TOXGR2;
      column GRPn GRP1 GRP2 GRP3 TRT_A TRT_B TRT_C;

      define GRPn /order order=data noprint;
      define GRP1 /order order=data display "SOC";
      define GRP2 /order order=data display "PT";
      define GRP3 /order order=data display "Grades";
      define TRT_A /order order=data;
      define TRT_B /order order=data;
      define TRT_C /order order=data;

      title1 "Table 2 - Adverse Events by SOC, PT and maximum grade";
run;
```