

Efficient Way to Create Adverse Event (AE) Summary Tables by System Organ Class (SOC) and Preferred Term (PT) using PROC SQL in SAS

Hardik Sheth, Catalyst Clinical Research; Nathan Riley, Catalyst Clinical Research

ABSTRACT

Adverse event (AE) summary tables by System Organ Classes (SOCs) and Preferred Terms (PTs) provide the safety of the drug to the patients; hence, they are considered key outputs for most clinical trials. The methodology proposed herein contains two straightforward steps for such AE summary table creation: 1) calculate the number of patients, percentages and number of AEs by SOC and PT for each treatment group and overall (note: at each level of summation, these counts are typically displayed as one event per subject), and 2) transpose all calculated statistics for each treatment group and the overall column. This paper will illustrate how to implement both steps to create AE summary tables by SOC and PT efficiently using the SQL procedure in SAS. Previous discussions of this topic did not transpose after the calculations were completed, so this solution provides one more option for those that frequently prepare such outputs.

INTRODUCTION

Adverse event (AE) summary tables by System Organ Class (SOC) and Preferred Term (PT) are one of the most common tables for the safety assessment of any clinical study. It is a key safety report for any phase of a clinical trial. In this paper, we are presenting the SAS program to calculate the number of patients, percentages, and number of AEs for each treatment group by SOC and PT. Also demonstrated is using PROC SQL to transpose the calculated statistics to finalize the table. The macro may also be adapted to produce similar subset tables that require summarizing based on specific types of AEs, e.g. serious AEs. Table 1 is an example table shell for the AE summary by SOC and PT.

Table 1.1.1
Treatment-Emergent Adverse Events by System Organ Class and Preferred Term
Safety Analysis Set

System Organ Class Preferred Term	TRTA	TRTB	Total
	(N=xx) n (%) m	(N=xx) n (%) m	(N=xx) n (%) m
Patients Having at Least One TEAE	xx (xx.x) xx	xx (xx.x) xx	xx (xx.x) xx
System Organ Class #1	xx (xx.x) xx	xx (xx.x) xx	xx (xx.x) xx
Preferred Term #1	xx (xx.x) xx	xx (xx.x) xx	xx (xx.x) xx
Preferred Term #2	xx (xx.x) xx	xx (xx.x) xx	xx (xx.x) xx

Table 1. Shell for AE Summary by SOC and PT

To efficiently generate these tables, we use PROC SQL. PROC SQL allows SAS users to implement structured query language in their code. Thus, with proper usage, programmers can greatly increase their efficiency. For the purposes of this paper, it is important to understand a few common practices in PROC SQL: creating macro variables, combining queries using set operators, subqueries, aggregate functions, and CASE expressions. Specifically, in relation to set operators, the OUTER UNION operator coupled with the CORR keyword functions to append the results for each level of the table, i.e., all AEs, by SOC, and by SOC and PT. Further details will be provided below.

GENERATING AE COUNTS BY TREATMENT GROUP, SOC, AND PT

The code shown below creates the %DOIT macro with the keyword parameters pop, cond, and events and completes the setup processing prior to running the SQL procedure:

```
%MACRO doit (pop=, cond=, events=Y) ;

/*****
*** Subset Data & Create Treatment Groups
*****/
data adsl0;
    set ADAM.adsl (where=(&pop.="Y"));
run;
data adsl;
    set adsl0;
    Output;
    TRT01AN = 99;
    Output;
run;

data adae;
    merge ADAM.adae (in=a) adsl0 (in=b);
    by usubjid;
    if a & b;
run;
data adae;
    set adae (where=(&cond.));

    if AEBODSYS="" then AEBODSYS="Uncoded";
    if AEDECOD="" then AEDECOD="Uncoded";

    Output;
    TRT01AN=99;
    Output;
run;
```

Table 2 provides an explanation and example of the functions of the specified macro parameters.

Parameter	Description / Example
pop	Population Flag / SAFFL
cond	Conditions used to subset the AEs / TRTEMFL="Y"
events	Y or N indicator to activate or suppress the display of total AE counts ("m" in the table shell header found in Table 1) / N

Table 2. %DOIT Macro Parameter Descriptions

CREATING MACRO VARIABLES AND COUNTING WITH PROC SQL

The following code is used to create macro variables for the total number of distinct treatment groups, the value of TRT01AN assigned to each treatment group from ADL, and the total number of subjects for each treatment group:

```
/*****
*** PROC SQL to Create AE Table by SOC & PT
*****/
proc sql noprint;
/** Assigning Macro Variable for Number of Treatment Groups **/
```

```

select count(distinct TRT01AN) into: numtrt trimmed from adsl;

/** Assigning Macro Variable for Each Treatment Group */
select distinct TRT01AN into :TRT1 - :TRT&numtrt. from adsl order by TRT01AN;

/** Denominators for each Treatment Group */
select count(distinct USUBJID) into :TOT1 - :TOT&numtrt. from adsl group by
TRT01AN;

```

The first SELECT statement counts the number of distinct treatment groups from ADSL. This allows us to dynamically account for the number of treatment groups required to report in the table output. An example is found in the second SELECT statement where the created macro variable NUMTRT from SELECT statement one is used to indicate the last TRT macro variable needed for assigning macro variables to each distinct value of TRT01AN. Finally, the third SELECT statement counts the distinct number of subjects per TRT01AN and assigns the value to the corresponding macro variable with the prefix TOT.

After creating the macro variables, we use the following code to count the number of subjects and total events for each level of the table:

```

/** Counting patients for each AEBODSYS, AEDECOD */
create table FROW as
  select TRT01AN, 1 as grp, "Any Adverse Event" as text length=200,
    count(distinct USUBJID) as VAL, count(*) as AE
  from ADAE
  group by TRT01AN

  outer union corr

/** By AEBODSYS */
select TRT01AN, AEBODSYS, 2 as grp, AEBODSYS as text length=200,
  count(distinct USUBJID) as VAL, count(*) as AE,
  (select count(distinct USUBJID)*-1 from ADAE
    where TRT01AN=99 & AEBODSYS=a.AEBODSYS group by AEBODSYS)
  as sort1
  from ADAE a
  group by TRT01AN, AEBODSYS

  outer union corr

/** By AEBODSYS, AEDECOD */
select TRT01AN, AEBODSYS, AEDECOD, 2 as grp,
  " ||AEDECOD as text length=200,
  count(distinct SUBJID) as VAL, count(*) as AE,
  (select count(distinct USUBJID)*-1 from ADAE where TRT01AN=99 &
    AEBODSYS=a.AEBODSYS group by
    AEBODSYS)
  as sort1,
  (select count(distinct USUBJID)*-1 from ADAE where TRT01AN=99 &
    AEBODSYS=a.AEBODSYS & AEDECOD=a.AEDECOD group by
    AEBODSYS, AEDECOD)
  as sort2
  from ADAE a
  group by TRT01AN, AEBODSYS, AEDECOD

order by TRT01AN, sort1, AEBODSYS, sort2, AEDECOD;

```

The first SELECT statement generates the counts of subjects and number of events for the first row of the table, i.e., subjects with any adverse events by treatment group. Between SELECT statements, the OUTER UNION set operator is implemented to append the results to one dataset called FROW. Similar to specifying multiple datasets within one DATA step SET statement, the OUTER UNION operator allows us to append results of multiple queries. The CORR keyword is used to instruct SAS that variables with identical names between the queries should be overlayed, that is, displayed concatenated in the same column. The second SELECT statement counts the subjects by treatment group and SOC. Furthermore, the variable sort1 is created using a subquery where the result takes on the value of the number of subjects for the corresponding SOC from the total group, i.e., TRT01AN=99. This variable is created to aid in the final sort of the table by descending frequency of SOC and PT. The third and final SELECT statement counts the subjects by treatment group, SOC, and PT. As in the second SELECT statement, the variable sort1 is created. The third SELECT statement also creates the variable sort2, which is similar to sort1 but counts by SOC and PT for the total group.

Figure 1 displays a subset of the created FROW dataset. Also illustrated is the origin of the sort1 and sort2 variables.

	TRT01AN	GRP	TEXT	VAL	AE	AEBODSYS	SORT1	AEDECOD	SORT2
1	1	1	Any Adverse Event	6	30				
2	1	2	Gastrointestinal disorders	4	5	Gastrointestinal disorders	-7		
3	1	2	Nausea	1	1	Gastrointestinal disorders	-7	Nausea	-3
4	1	2	Vomiting	1	1	Gastrointestinal disorders	-7	Vomiting	-3
5	1	2	Constipation	2	2	Gastrointestinal disorders	-7	Constipation	-2
6	1	2	Abdominal pain	1	1	Gastrointestinal disorders	-7	Abdominal pain	-1
7	1	2	General disorders and administration site conditions	3	6	General disorders and administration site conditions	-6		
8	1	2	Fatigue	3	5	General disorders and administration site conditions	-6	Fatigue	-5
9	1	2	Chills	1	1	General disorders and administration site conditions	-6	Chills	-2
10	1	2	Investigations	1	2	Investigations	-3		
11	1	2	Electrocardiogram T wave abnormal	1	1	Investigations	-3	Electrocardiogram T wave abnormal	-1
12	1	2	Troponin I increased	1	1	Investigations	-3	Troponin I increased	-1
84	99	1	Any Adverse Event	15	61				
85	99	2	Gastrointestinal disorders	7	11	Gastrointestinal disorders	-7		
86	99	2	Nausea	3	3	Gastrointestinal disorders	-7	Nausea	-3
87	99	2	Vomiting	3	3	Gastrointestinal disorders	-7	Vomiting	-3
88	99	2	Constipation	2	2	Gastrointestinal disorders	-7	Constipation	-2
89	99	2	Abdominal pain	1	1	Gastrointestinal disorders	-7	Abdominal pain	-1
90	99	2	Gingival pain	1	1	Gastrointestinal disorders	-7	Gingival pain	-1
91	99	2	Stomatitis	1	1	Gastrointestinal disorders	-7	Stomatitis	-1
92	99	2	General disorders and administration site conditions	6	9	General disorders and administration site conditions	-6		
93	99	2	Fatigue	5	7	General disorders and administration site conditions	-6	Fatigue	-5
94	99	2	Chills	2	2	General disorders and administration site conditions	-6	Chills	-2
95	99	2	Investigations	3	8	Investigations	-3		
96	99	2	Blood bilirubin increased	2	3	Investigations	-3	Blood bilirubin increased	-2
97	99	2	Alanine aminotransferase	1	1	Investigations	-3	Alanine aminotransferase	-1
98	99	2	Aspartate aminotransferase increased	1	1	Investigations	-3	Aspartate aminotransferase increased	-1
99	99	2	Blood alkaline phosphatase increased	1	1	Investigations	-3	Blood alkaline phosphatase increased	-1
100	99	2	Electrocardiogram T wave abnormal	1	1	Investigations	-3	Electrocardiogram T wave abnormal	-1
101	99	2	Troponin I increased	1	1	Investigations	-3	Troponin I increased	-1

Figure 1. Snapshot of the Intermediate Dataset FROW

TRANSPOSING COUNTS TO FINALIZE TABLE OUTPUT

The final step in creating the dataset for the AE table is to transpose and format the counts. At this point in the code, we utilize the macro variables created in the previous step. To complete the transpose, PROC SQL's property of aggregate functions is applied. The following code demonstrates how this is done:

```
/** Transposing by Treatment Group and Assigning Percentages */
create table Final as
select sort1, AEBODSYS, sort2, AEDECOD, text, grp,
       %DO i = 1 %TO &numtrt.;
       max(case when TRT01AN = &&trt&i. then put (VAL,3.) || "(" ||
                                     put (100* (VAL/&&tot&i.),5.1) || "%) "
               %IF &events.=Y %THEN %DO; || " [" || put (AE,5.) || "]" %END;
               else put (0,3.) END) as COL&&TRT&i. length=200
       %IF &i. ^= &numtrt. %THEN %DO;
```

```

        ,
    %END;
%END;
from FROW
group by sort1, AEBODSYS, sort2, AEDECOD, text, grp;
quit;

%MEND;
%doit (pop=SAFFL, cond=TRTEMFL="Y" );

```

Firstly, a macro %DO loop is implemented to ensure that a column for each treatment group is created. The macro variable NUMTRT acts as the end point for the %DO loop. Since we group by sort1, AEBODSYS, sort2, AEDECOD, text, and grp, PROC SQL creates one record per distinct combination of these group variables when returning the result from the summary function MAX(). This is the key PROC SQL property that completes the transpose. Within the function, a CASE expression loops through each value of TRT01AN in the data, where the values are stored in the macro variables with the prefix TRT. The calculation of the percentage invokes the macro variables with the prefix TOT to capture the corresponding denominator for the treatment group. The keyword macro parameter EVENTS determines if the number of total AE events will be displayed in the output. Finally, if the value of TRT01AN is not found in the data for the current GROUP BY combination, a value of 0 is returned using the ELSE expression.

Figure 2 shows a snapshot of the resulting dataset. Under the population flag condition passed to the macro, i.e., POP equals SAFFL, there are four treatment groups in the data: TRT01AN equal to 1, 2, 3, and 8. Thus, the transpose results in the creation of COL1, COL2, COL3, and COL8. COL99 is created as the total group from the initial processing in outputting an extra observation where TRT01AN equals 99. This dataset can be used to output the report in the desired format.

VIEWTABLE: Work.Final											
	SORT1	AEBODSYS	SORT2	AEDECOD	TEXT	GRP	COL1	COL2	COL3	COL8	COL99
1					Any Adverse Event	1	6 (85.7%) [30]	4 (100.0%) [11]	2 (100.0%) [6]	3 (75.0%) [14]	15 (88.2%) [61]
2	-7	Gastrointestinal disorders			Gastrointestinal disorders	2	4 (57.1%) [5]	0	2 (100.0%) [4]	1 (25.0%) [2]	7 (41.2%) [11]
3	-7	Gastrointestinal disorders	-3	Nausea	Nausea	2	1 (14.3%) [1]	0	1 (50.0%) [1]	1 (25.0%) [1]	3 (17.6%) [3]
4	-7	Gastrointestinal disorders	-3	Vomiting	Vomiting	2	1 (14.3%) [1]	0	1 (50.0%) [1]	1 (25.0%) [1]	3 (17.6%) [3]
5	-7	Gastrointestinal disorders	-2	Constipation	Constipation	2	2 (28.6%) [2]	0	0	0	2 (11.8%) [2]
6	-7	Gastrointestinal disorders	-1	Abdominal pain	Abdominal pain	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]
7	-7	Gastrointestinal disorders	-1	Gingival pain	Gingival pain	2	0	0	1 (50.0%) [1]	0	1 (5.9%) [1]
8	-7	Gastrointestinal disorders	-1	Stomatitis	Stomatitis	2	0	0	1 (50.0%) [1]	0	1 (5.9%) [1]
12	-3	Investigations			Investigations	2	1 (14.3%) [2]	1 (25.0%) [2]	0	1 (25.0%) [4]	3 (17.6%) [8]
13	-3	Investigations	-2	Blood bilirubin increased	Blood bilirubin increased	2	0	1 (25.0%) [2]	0	1 (25.0%) [1]	2 (11.8%) [3]
14	-3	Investigations	-1	Alanine aminotransferase	Alanine aminotransferase	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
15	-3	Investigations	-1	Aspartate aminotransferase increased	Aspartate aminotransferase increased	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
16	-3	Investigations	-1	Blood alkaline phosphatase increased	Blood alkaline phosphatase increased	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
17	-3	Investigations	-1	Electrocardiogram T wave abnormal	Electrocardiogram T wave abnormal	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]
18	-3	Investigations	-1	Troponin I increased	Troponin I increased	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]
27	-3	Nervous system disorders			Nervous system disorders	2	1 (14.3%) [1]	1 (25.0%) [1]	0	1 (25.0%) [1]	3 (17.6%) [3]
28	-3	Nervous system disorders	-1	Disturbance in attention	Disturbance in attention	2	0	1 (25.0%) [1]	0	0	1 (5.9%) [1]
29	-3	Nervous system disorders	-1	Dizziness	Dizziness	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
30	-3	Nervous system disorders	-1	Headache	Headache	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]
44	-2	Vascular disorders			Vascular disorders	2	0	1 (25.0%) [2]	1 (50.0%) [1]	0	2 (11.8%) [3]
45	-2	Vascular disorders	-1	Hypotension	Hypotension	2	0	1 (25.0%) [2]	0	0	1 (5.9%) [2]
46	-2	Vascular disorders	-1	Orthostatic hypotension	Orthostatic hypotension	2	0	0	1 (50.0%) [1]	0	1 (5.9%) [1]
47	-1	Eye disorders			Eye disorders	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
48	-1	Eye disorders	-1	Eye haematoma	Eye haematoma	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
49	-1	Hepatobiliary disorders			Hepatobiliary disorders	2	0	0	0	1 (25.0%) [2]	1 (5.9%) [2]
50	-1	Hepatobiliary disorders	-1	Hepatomegaly	Hepatomegaly	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
51	-1	Hepatobiliary disorders	-1	Jaundice	Jaundice	2	0	0	0	1 (25.0%) [1]	1 (5.9%) [1]
52	-1	Psychiatric disorders			Psychiatric disorders	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]
53	-1	Psychiatric disorders	-1	Anxiety	Anxiety	2	1 (14.3%) [1]	0	0	0	1 (5.9%) [1]

Figure 2. Snapshot of the Transposed Dataset Final

CONCLUSION

AE summary tables by SOC and PT are valuable indicators of the safety in a clinical trial. Hence, it is highly beneficial to be able to efficiently program these tables. As illustrated in this paper, PROC SQL is one way to generate these tables with high efficiency. In just one SQL procedure, we can obtain distinct subject counts and total number of events by treatment group, SOC, and PT, transpose the data to get a column for each treatment group, format the counts to include percentages, and sort by descending frequency of SOC and PT. This is completed using the following techniques in PROC SQL: INTO clauses

to establish macro variables, OUTER UNION set operators coupled with the CORR keyword to append multiple queries, subqueries to obtain sorting variables, aggregating functions with the GROUP BY clause to transpose data, and CASE expressions to output a column for each treatment group. Incorporating all these techniques in one SQL procedure builds an efficient routine for outputting AE summary tables that requires fewer lines of code than constructing similar programs using SORT, FREQ, and TRANSPOSE procedures and DATA steps.

CONTACT INFORMATION

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

Hardik Sheth
Catalyst Clinical Research
hardik.sheth@catalystcr.com

Nathan Riley
Catalyst Clinical Research
nathan.riley@catalystcr.com

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.

APPENDIX 1: FULL %DOIT MACRO CODE TO GENERATE AE SUMMARY TABLES BY SOC AND PT USING PROC SQL

```
%MACRO doit(pop=,cond=,events=Y);

/*****
*** Subset Data & Create Treatment Groups
*****/
data adsl0;
    set ADAM.adsl (where=(&pop.="Y"));
run;
data adsl;
    set adsl0;
    Output;
    TRT01AN = 99;
    Output;
run;

data adae;
    merge ADAM.adae(in=a) adsl0(in=b);
    by usubjid;
    if a & b;
run;
data adae;
    set adae(where=(&cond.));

    if AEBODSYS="" then AEBODSYS="Uncoded";
    if AEDECOD="" then AEDECOD="Uncoded";

    Output;
    TRT01AN=99;
    Output;
run;

/*****
*** PROC SQL to Create AE Table by SOC & PT
*****/
proc sql noprint;
/** Assigning Macro Variable for Number of Treatment Groups **/
select count(distinct TRT01AN) into: numtrt trimmed from adsl;

/** Assigning Macro Variable for Each Treatment Group **/
select distinct TRT01AN into :TRT1 - :TRT&numtrt. from adsl order by TRT01AN;

/** Denominators for each Treatment Group **/
select count(distinct USUBJID) into :TOT1 - :TOT&numtrt. from adsl group by
TRT01AN;

/** Counting patients for each AEBODSYS, AEDECOD **/
create table FROW as
    select TRT01AN, 1 as grp, "Any Adverse Event" as text length=200,
        count(distinct USUBJID) as VAL, count(*) as AE
    from ADAE
    group by TRT01AN
```

```

outer union corr

/** By AEBODSYS */
select TRT01AN, AEBODSYS, 2 as grp, AEBODSYS as text length=200,
       count(distinct USUBJID) as VAL, count(*) as AE,
       (select count(distinct USUBJID)*-1 from ADAE
        where TRT01AN=99 & AEBODSYS=a.AEBODSYS group by AEBODSYS)
       as sort1
from ADAE a
group by TRT01AN, AEBODSYS

outer union corr

/** By AEBODSYS, AEDECOD */
select TRT01AN, AEBODSYS, AEDECOD, 2 as grp,
       " "||AEDECOD as text length=200,
       count(distinct SUBJID) as VAL, count(*) as AE,
       (select count(distinct USUBJID)*-1 from ADAE where TRT01AN=99 &
        AEBODSYS=a.AEBODSYS group by
        AEBODSYS)
       as sort1,
       (select count(distinct USUBJID)*-1 from ADAE where TRT01AN=99 &
        AEBODSYS=a.AEBODSYS & AEDECOD=a.AEDECOD group by
        AEBODSYS, AEDECOD)
       as sort2
from ADAE a
group by TRT01AN, AEBODSYS, AEDECOD

order by TRT01AN, sort1, AEBODSYS, sort2, AEDECOD;

/** Transposing by Treatment Group and Assigning Percentages */
create table Final as
select sort1, AEBODSYS, sort2, AEDECOD, text, grp,
       %DO i = 1 %TO &numtrt.;
       max(case when TRT01AN = &&trt&i. then put(VAL,3.)||" ("||
                                               put(100*(VAL/&&tot&i.),5.1)||"%) "
               %IF &events.=Y %THEN %DO; ||" ["||put(AE,5.)||"]" %END;
               else put(0,3.) END) as COL&&TRT&i. length=200

       %IF &i. ^= &numtrt. %THEN %DO;
       ,
       %END;
       %END;
from FROW
group by sort1, AEBODSYS, sort2, AEDECOD, text, grp;
quit;

%MEND;
%doit(pop=SAFFL,cond=TRTEMFL="Y");

```