PharmaSUG 2015 - Paper QT45

A Practical Approach to Create Adverse Event Summary by Toxicity Grade Table

Zhengxin (Cindy) Yang, inVentiv Health, Princeton, NJ

ABSTRACT

Adverse event summary by toxicity grade table is a common requirement in clinical safety study. The table displays severity grade of organ toxicity per CTCAE (Common Terminology Criteria for Adverse Events) as a sub-categories in AE summary tables. Those tables can be difficult to create with added sub-categories of organ toxicity grades. The variation of grade presentation with study requirements also adds complexity to the table programming. This paper presents a practical approach for generating AE grade tables using modular macro calls. It uses modules to address various formatting issues with AE grade tables. A set of guidelines is presented to simplify the programming process.

INTRODUCTION

Adverse event summary by toxicity grade table is one of the most common table types in clinical safety reports, especially for oncology study reports. They summarize organ toxicity data in a clear manner. The grade information presented in tables use the CTCAE (Common Terminology Criteria for Adverse Events) for organ toxicity. The CTCAE defines as any abnormal clinical finding temporally associated with the use of a therapy as an adverse event, and standardizes adverse events in five grades by the National Institutes of Health. Toxicity is graded as mild (Grade 1), moderate (Grade 2), severe (Grade 3), or life-threatening (Grade 4) and Death (Grade 5). An AE by toxicity grade table is a highly effective way of presenting CTCAE data from a clinical trial.

Depending on the study design, adverse event summary by grade tables can display all five toxicity grades or only selected grades, such as only grade 2 or above inclusively or exclusively. Therefore, it is important to know what type of AE table is required, and to adjust the program according to the shell. The paper will address this in the first section. The paper will also discuss how to create modular macros to accommodate different types of AE grade table. This practical approach will make programming such tables easier.

STEPS BEFORE PROGRAMMING

A well designed program will save programming time. Several key points should be considered before creating an AE summary by grade table.

- **KNOW THE COUNT TYPE**

Shell examples 1 and 2 show the most commonly used AE summary by grade tables. Shell Example 1 is the overall summary, and shell example 2 is the summary by system organ class and preferred term. For both shells, the grade counts are in the last layer of the sub-header. However, the grade display can have variation. They can be displayed only "Grade **=** 2" or exclusively as "Grade **>=** 2". In some cases, the shell may require to display only moderate toxicity, such as grade 2 and above. This can be easily done via a modular approach.

Shell Example 1. Summary of Adverse Events by Toxicity Grade

	Treatment (N = xxx)	Placebo (N =xxx)	Total (N = xxx)
Subjects with any adverse event – n (%)	xx (xxx)	xx (xxx)	xx (xxx)
Worst grade of 1	xx (xxx)	xx (xxx)	xx (xxx)
Worst grade of 2	xx (xxx)	xx (xxx)	xx (xxx)
Worst grade of 3	xx (xxx)	xx (xxx)	xx (xxx)
Worst grade of 4	xx (xxx)	xx (xxx)	xx (xxx)
Worst grade of 5	xx (xxx)	xx (xxx)	xx (xxx)

Shell Example 2. Summary of Adverse Events by System Organ Class, Preferred Term and Grade

	Treatment (N = XXX) n (%)	Placebo (N =XXX) n (%)	Total (N = XXX) n (%)
Number of subjects reporting adverse events	xx (xx)	xx (xx.x)	xx (xxx)
Any System Organ Class	xx (xx)	xx (xx.x)	xx (xxx)
Any Preferred Term	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 2	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 3	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 4	xx (xx)	xx (xx.x)	xx (xxx)
Fatal	xx (xx)	xx (xx.x)	xx (xxx)
System Organ Class 1	xx (xx)	xx (xx.x)	xx (xxx)
Any Preferred Term	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 2	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 3	xx (xx)	xx (xx.x)	xx (xxx)
Grade ≥ 4	xx (xx)	xx (xx.x)	xx (xxx)
Fatal	xx (xx)	xx (xx.x)	xx (xxx)

Using the proper procedure can simplify the programming task. If the table shell requires displaying all the grades, like in shell example 1, then a PROC FREQ procedure or SQL GROUP BY statement should be used. If subsetting of the desired grades is needed, like in shell example 2, then a macro will make programming much easier.

```
** Get counts of all grades at once **;
proc freq data=adae noprint;
 table trtan*aetoxgrn/out=grade(drop=percent) nocum missing list;
run;

Proc sql;
  create table &out as
    select trtan, aetoxgrn, count(distinct usubjid) as count
      from temp
      group by trtan, aetoxgrn
  ;

** Get count of each grade **;
%macro get_data(in, out, condition);

proc freq data=&dsin(&condition.) noprint;
 table txcd/out=&dsout(drop=percent) nocum missing list;
run;
%mend get_data;

%get_data(dsin, dsout, where=(aetoxgrn >= 2));
```

Besides grade count types, program needs to consider the event count type as well. The summary of AE data can be at subject level or event level. Unless specifically mentioned, the event should be counted once per group. This desired group could be subjects only, system organ classes or preferred terms only, or a combination of system organ class and preferred terms. In this case, the duplicate events need to be eliminated at the desired group level. In SAS, the duplicate records can be removed using the NODUPKEY option in PROC FREQ, or DISTINCT argument in PROC SQL.

For the AE Summary by grade table, simply removing the duplicates is not a good practice. The program should pick the worst grade per desired group level, because it is most important to show the worst case scenario. Under this circumstance, an IF LAST statement in data step or MAX function in PROC SQL will do the trick.

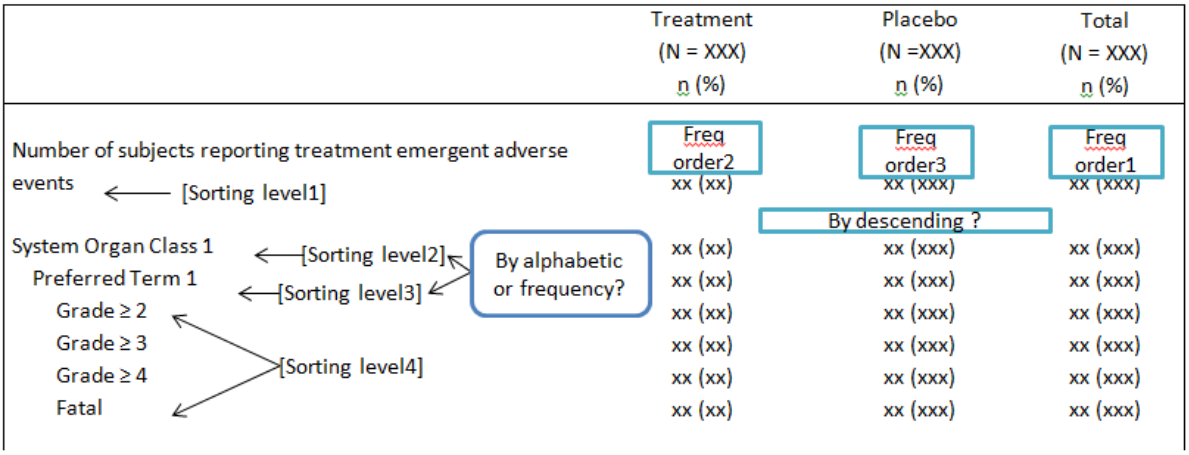
```
** Get the worst grade example 1 **;
proc sort data = ae;
    by trtan usubjid aetoxgrn;
run;

data ae1(keep=trtan usubjid aetoxgrn);
set ae;
by trtan usubjid aetoxgrn;
       if last.usubjid;
run;

** Get the worst grade example 2 **;
proc sql;
       create table ae1 as
       select distinct usubjid, trtan, max(aetoxgrn) as aetoxgrn from ae
       group by trtan, usubjid
       order by trtan, usubjid;
quit;
```

- **DISPLAY ORDERS**

Category order is also critical. In the overall AE summary by grade table, the grade counts will be displayed by the order from categories defined in the shell. It is very easy to set up the category order by marking the sorting keys and order on paper before programming.

For the AE summary table by system organ class and preferred term, the sorting key could be in either alphabetical order or in frequency order by columns. If the sorting is based on frequency count (descending order), the columns are also need to be considered as well. The programmer need to know which column is headmost for sorting. For example, last "Total" column could be the first, "Treatment" column could be the second and "Placebo" column could be the last in case of equal count. In this situation, it is better to first transpose the frequency count from vertical to horizontal data structure to get the correct sorting sequence. However, the sorting order for grade display is usually in the same order, as grade 1 to 5 regardless the frequency count.

Treatment Emergent Adverse Events by System Organ Class, Preferred Term and Grade



MODULAR MACROS

Modular macros can be used to efficiently create multiple, different tables for a study. It can also be used to create AE summary by grade tables.

Many programmers think modular macros are complicated one-size-fit-all macro that integrates many complex codes and ampersands together. Actually, modular macro programs do not have to be very complicated. A practical way is to simplify a modular macro is to build several small macros within the main macro. Since the small macros only serve a single task, only small changes are needed to adapt these macros for your data or study requirement. So

building a large macro using many simple, small macros greatly simplifies the programming. This practical approach will make program easy to understand and modify, so that only minimum effort is needed to meet the different study requirements.

Below is an example of a simple macro. This macro's purpose is to get the counts for each grade per subsetting logic by passing parameters through macro calls. So, a programmer can easily change the logic from grade '>=' to grade '=' or add more grade count requests by adding a simple macro call. It is convenient and flexible.

```
** Create a macro to get count for each grade **;
%macro get_data(out, condition);
proc freq data=ae&in.(&condition.) noprint;
 table txcd&key./out=&out(drop=percent) nocum missing list;
run;
%mend get_data;

%get_data(gd4&in.,%str(where=(aetoxgrn >= 4)));
%get_data(gd5&in.,%str(where=(aetoxgrn = 5)));
```

Another example below is to call macro for AE event count. After the display and sorting order are decided, a simple macro can be constructed to count the desired summary of AE grade, which can be the overall summary, the summary by preferred AE term only, or by system organ class and preferred AE term.

```
** get ae any count **;
%aegrd_cnt(any,all1,,,,2,1);
** get ae system organ class count **;
%aegrd_cnt(soc,all2,*aesoc,aesoc,,3,2);
** get ae system organ class, aept count **;
%aegrd_cnt(pt,all3,*aesoc*aept,aesoc,aept,3,3);
```

In summary, modular macro design is a very practical way to generate AE summary by grade table. By assembling easily understandable small macros as objects in the main macro, the program will fit a variety of commonly used situations, like displaying each grade inclusively or exclusively, or displaying entire grade or displaying only requested grade. With a few simple macro calls, a programmer can easily create many different kinds of grade tables. Finally, an example program to generate AE summary by toxicity grade table data set by using this approach is shown below.

```
*************************************************************************
** Sample Program
** Create Adverse Event Summary by Toxicity Grade table output data sets

** The sorting order:
** Descending order by total count(col99), 1st treatment count(col1),
** 2nd treatment count(col2) and then alphabetical order
**
** Macro variable:
** dsin1 - input dataset to get the total count for N
** dsin2 - AE input dataset
** out   - output dataset ready to be plotted
** pop1  - subset statement for dsin1 if any
** pop2  - subset statement for dsin2 if any
*************************************************************************;

%macro creat_aegrd(dsin1=, dsin2=, outds=, pop1=, pop2=);

** Create macro variables for Overall patient count for each treatment group **;
data dm(keep=usubjid trtan: saffl);
   set &dsin1;
   &pop1 ;
run;

data trtcount(where=(txcd ne .));
   set dm;
   txcd = trtan;
```

```
   output;
   txcd = 99;
   output;
run;

proc sql noprint;
  select count(distinct usubjid) as count into :cnt1-:cnt3
      from trtcount
      group by txcd
  ;
quit;


%put &cnt1 &cnt2 &cnt3;


************************************************************
Count Total Number of Subjects per Treatment Group/Cohort
************************************************************;
data ae(where=(txcd ne .));
set &dsin2(rename=(aedecod =aept);
   &pop2;

   txcd = trtan;
   output;
   txcd = 99;
   output;
run ;

**************************************************
** Get Maximum Grade for All level
**************************************************;
proc sql;
** Any AE **;
create table aeany as
select distinct usubjid, txcd, max(aetoxgrn) as aetoxgrn from ae
group by txcd, usubjid
order by txcd, usubjid ;
** AE bodysys **;
create table aesoc as
select distinct usubjid, txcd, aesoc, max(aetoxgrn) as aetoxgrn from ae
group by txcd, usubjid, aesoc
order by txcd, usubjid, aesoc ;
** AE aesoc aept **;
create table aept as
select distinct usubjid, txcd, aesoc, aept, max(aetoxgrn) as aetoxgrn from ae
group by txcd, usubjid, aesoc, aept
order by txcd, usubjid, aesoc, aept ;

quit;


%macro aegrd_cnt(in,dsout,key, key1,key2,grp,type);

%macro get_data(out, condition);

proc freq data=ae&in.(&condition.) noprint;
 table txcd&key./out=&out(drop=percent) nocum missing list;
run;
%mend get_data;

%get_data(aae&in.,%str());
%get_data(gd2&in.,%str(where=(aetoxgrn >= 2)));
```

```
%get_data(gd3&in.,%str(where=(aetoxgrn >= 3)));
%get_data(gd4&in.,%str(where=(aetoxgrn >= 4)));
%get_data(gd5&in.,%str(where=(aetoxgrn = 5)));

data all(where=(grp ne .));
set aae&in.(in=a) aae&in.(in=b) aae&in.(in=c) gd2&in.(in=d) gd3&in.(in=e) gd4&in.(in=f)
gd5&in.(in=g);

%if &type = 1 %then %do;
      if a then do; grp=1; ord=1; end; ** num subj **;
%end;
%if &type = 2 or &type = 1 %then %do;
      if b then do; grp=&grp; ord=0; end; ** soc **;
%end;
      if c then do; grp=&grp; ord=1; end;  ** aept **;
      if d then do; grp=&grp; ord=2; end;  ** grade >=2 **;
      if e then do; grp=&grp; ord=3; end;  ** grade >=3 **;
      if f then do; grp=&grp; ord=4; end;  ** grade >=4 **;
      if g then do; grp=&grp; ord=5; end;  ** grade =5 **;

run;

proc sort data=all; by grp ord &key1 &key2; run;

proc transpose data=all out=&dsout(drop=_name_ _label_);
      by grp  ord &key1 &key2;
      id txcd;
      var count;
run;

** Get descending order for all subjects, treatment1, treatment2 **;
%if &type ne 1 %then %do;
%if &type = 2 %then %let ord = 0;
%else %let ord = 1;
proc sort data=&dsout (where=(ord=&ord)) out=ord&in.;
by descending _99 descending _1 descending _2 ae&in.;
run;

data ord&in.(keep=ae&in. sort&ord.);
      set ord&in.;
      by descending _99 descending _1 descending _2 ae&in.;
      sort&ord. + 1;
run;
%end;

%mend aegrd_cnt;

** get ae any count **;
%aegrd_cnt(any,all1,,,,2,1);
** get ae system organ class count **;
%aegrd_cnt(soc,all2,*aesoc,aesoc,,3,2);
** get ae system organ class, aept count **;
%aegrd_cnt(pt,all3,*aesoc*aept,aesoc,aept,3,3);

************************************************
** Combine all count
************************************************;

data combine(drop=i);
set all1 all2 all3;

array denm{3} cnt1 cnt2 cnt3 (&cnt1 &cnt2 &cnt3) ;
array c{3} _1 _2 _99 ;
```

```
array col{3} $20 col1 col2 col99;
length  aetxt $100;

do i=1 to 3;
      if c{i} ne . then col{i}=strip(put(c{i},8.0))||'
('||strip(put(c{i}/denm{i}*100,5.1))||')';
      else col{i}='0 (0.0)';
end;

 if grp=1 then aetxt = "Number of subjects reporting treatment-emergent adverse
events";
 else if grp=2 then do;
      if ord=0 then aetxt= "Any System Organ Class";
      else if ord=1 then aetxt= "Any Preferred Term";
      else aetxt = put(ord, grdfmt.);
 end;
 else if grp=3 then do;
      if ord=0 then aetxt = trim(left(aesoc));
      else if ord=1 then do;
            if aept='' then aetxt = "Any Preferred Term";
            else aetxt = trim(left(aept));
      end;
      else aetxt = put(ord, grdfmt.);
 end;

run;

** Merge sorting order **;
proc sort data=ordsoc; by  aesoc; run;
proc sort data=combine; by  aesoc; run;

data combine;
merge combine(in=a) ordsoc(keep=aesoc sort0);
by  aesoc;
if a;
run;

proc sort data=ordpt; by  aept; run;
proc sort data=combine; by  aept; run;

data &outds;
merge combine(in=a) ordpt(keep=aept sort1);
by  aept;
if a;
run;

proc sort data=&outds;
by grp sort0 sort1 aept ord;
run;

** Check final output dataset which is ready to be plotted **;
title "Output Dataset for Adverse Event Summary by Toxicity Grade table";
proc print data=&outds;
var aetxt col1 col2 col99;
run;

%mend creat_aegrd;
```

Output Dataset for Adverse Event Summary by Toxicity Grade table

aetxt	col1	col2	col99
Number of subjects reporting treatment-emergent adverse events	163 (54.5)	164 (55.0)	327 (54.8)
Any System Organ Class	163 (54.5)	164 (55.0)	327 (54.8)
Any Preferred Term	163 (54.5)	164 (55.0)	327 (54.8)
Grade >= 2	136 (45.5)	132 (44.3)	268 (44.9)
Grade >= 3	97 (32.4)	83 (27.9)	180 (30.2)
Grade >= 4	28 (9.4)	23 (7.7)	51 (8.5)
Fatal	3 (1.0)	4 (1.3)	7 (1.2)
General disorders and administration site conditions	89 (29.8)	113 (37.9)	202 (33.8)
Any Preferred Term	89 (29.8)	113 (37.9)	202 (33.8)
Grade >= 2	50 (16.7)	69 (23.2)	119 (19.9)
Grade >= 3	18 (6.0)	27 (9.1)	45 (7.5)
Grade >= 4	1 (0.3)	1 (0.3)	2 (0.3)
Fatal	0 (0.0)	1 (0.3)	1 (0.2)
Fatigue	62 (20.7)	54 (18.1)	116 (19.4)
Grade >= 2	33 (11.0)	36 (12.1)	69 (11.6)
Grade >= 3	13 (4.3)	15 (5.0)	28 (4.7)
Oedema peripheral	13 (4.3)	51 (17.1)	64 (10.7)
Grade >= 2	4 (1.3)	22 (7.4)	26 (4.4)
Grade >= 3	0 (0.0)	6 (2.0)	6 (1.0)

CONCLUSION

Display grades in adverse event summary are very common for the drug safety study and are required in most oncology studies. Creating multiple such tables for a study report may seem challenging, but it can be easily accomplished using the modular approach outlined in this report.

ACKNOWLEDGMENTS

I would like to thank Frank Shen for proofreading my draft paper.

CONTACT INFORMATION

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

Name:	Zhengxin (Cindy) Yang
Enterprise:	inVentiv Health
Address:	504 Carnegie Center
City, State ZIP:	Princeton, NJ 08540
Work Phone:	215-860-2474
E-mail:	cindy.yang@inventivhealth.com