Paper CT08

BY-lieve in better BY group tables

Chris Spence, Cytel, Basel, Switzerland

ABSTRACT

When tasked with creating tables using a BY variable one of the biggest challenges is displaying correct denominator labels for each treatment within each BY group. For this, most programmers will resort to macro techniques. The most common method is to simply wrap the entire program in a macro and call it for each BY group with denominator labels retrieved from the variable labels for each run. Alternatively, only the PROC REPORT is wrapped in a macro, with the denominator labels defined using macro variables within the PROC REPORT, often with consecutive ampersand resolutions.

This paper will show an alternative method in which ACROSS variables are leveraged within a single PROC REPORT. We demonstrate with this method how denominator labels can be correctly assigned in such a report. Additionally, we show how to expand the technique to include any required spanning headers and finally how this facilitates enhanced programmatic Quality Control (QC).

INTRODUCTION

Most programmers are familiar with ACROSS variables from their training, and numerous papers discuss their use. However, these examples often involve transposing the input dataset using ACROSS variables before summarizing the transposed data. As clinical programmers, our goal is to create a final reporting dataset that closely mirrors the final output to facilitate easier programmatic QC, and as such we tend to overlook the potential of ACROSS variables.

In the following examples we will consider a simple adverse events frequency table. How can we extend it to display age group as a BY variable with the correct denominator labels for each group?

All data used in this paper is from the R *admiral* package with minor modifications for treatment groups and sex.

CREATING AN DENOMINATOR LABEL DATASET

STEP 1

We begin by creating a dataset that counts the number of subjects within each treatment and age group in the relevant analysis set. The dataset is created using a PROC SQL cartesian product to ensure that records exist for each treatment and each age group, even if some groups have no subjects.

TRT01AN	TRT01A	AGEGR1	AGEGR1N	NCOL
0	Placebo + SoC	<65	1	14
0	Placebo + SoC	65-80	2	42
0	Placebo + SoC	>80	3	30
1	Drug A + SoC	<65	1	19
1	Drug A + SoC	65-80	2	102
1	Drug A + SoC	>80	3	47

STEP 2

The data is further processed to generate labels that combine the treatment names with frequency counts (VAR1). We then transpose the data so that each treatment is represented in its own column, with the full denominator label included.

```
PROC TRANSPOSE data=pop2 out=popt1 (drop=_name_) prefix=ncol;
  BY agegr1n agegr1;
  VAR var1;
  ID trt01an;
RUN;
```

In this example we use the numeric treatment variable to identify the columns.

AGEGR1N	AGEGR1	NCOL0	NCOL1
1	<65	Placebo + SoC#(N=14)	Drug A + SoC#(N=19)
2	65-80	Placebo + SoC#(N=42)	Drug A + SoC#(N=102)
3	>80	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)

CREATING ADVERSE EVENT (AE) COUNT DATA

STEP 3

Next, we create an AE frequency count dataset grouped by treatment and age group. For simplicity, in this example we only count at the preferred term (PT) level, without including a total subject row or system organ class (SOC) totals. Using the dataset from step 1, we can calculate the percentage for each row. The partial dataset below (count2) demonstrates that percentages (NPCT) are correctly computed for each row based on the counts (NCOL) for each treatment and age group combination which were created in step 1.

TRT01AN	TRT01A	AGEGR1N	AGEGR1	AESOC	AEDECOD	NPAT	NCOL	NPCT
0	Placebo + SoC	1	<65	GENERAL DISORDERS AND ADMINISTRATION SITE CONDITIONS	ASTHENIA	1	14	1 (7.1)
0	Placebo + SoC	2	65-80	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	1	42	1 (2.4)
0	Placebo + SoC	2	65-80	PSYCHIATRIC DISORDERS	AGITATION	1	42	1 (2.4)
0	Placebo + SoC	3	>80	MUSCULOSKELETAL AND CONNECTIVE TISSUE DISORDERS	BACK PAIN	1	30	1 (3.3)
0	Placebo + SoC	3	>80	PSYCHIATRIC DISORDERS	AGITATION	1	30	1 (3.3)
1	Drug A + SoC	2	65-80	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	3	102	3 (2.9)
1	Drug A + SoC	2	65-80	MUSCULOSKELETAL AND CONNECTIVE TISSUE DISORDERS	BACK PAIN	4	102	4 (3.9)
1	Drug A + SoC	2	65-80	PSYCHIATRIC DISORDERS	AGITATION	3	102	3 (2.9)
1	Drug A + SoC	3	>80	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	1	47	1 (2.1)
1	Drug A + SoC	3	>80	GENERAL DISORDERS AND ADMINISTRATION SITE CONDITIONS	ASTHENIA	1	47	1 (2.1)

STEP 4

Once again, we transpose the data using numeric treatment identifiers as the ID variable so that each treatment is in a separate column.

```
PROC TRANSPOSE data=count2 out=rep1 prefix=nct;
  BY agegr1n agegr1 aesoc aedecod;
  VAR npct;
  ID trt01an;
RUN;
```

BRINGING IT ALL TOGETHER

STEP 5

We merge our transposed AE dataset from step 4 with the denominator labels from step 2. This prepares the data for reporting. The snippet below shows that NCOL0 and NCOL1 provide the correct denominator labels for each AGEGR1 BY variable.

AGEGR1N	AGEGR1	AESOC	AEDECOD	NCT0	NCT1	NCOL0	NCOL1
1	<65	GENERAL DISORDERS AND ADMINISTR...	ASTHENIA	1 (7.1)	0	Placebo + SoC#(N=14)	Drug A + SoC#(N=19)
2	65-80	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	1 (2.4)	3 (2.9)	Placebo + SoC#(N=42)	Drug A + SoC#(N=102)
2	65-80	MUSCULOSKELETAL AND CONNECTIVE ...	BACK PAIN	0	4 (3.9)	Placebo + SoC#(N=42)	Drug A + SoC#(N=102)
2	65-80	PSYCHIATRIC DISORDERS	AGITATION	1 (2.4)	3 (2.9)	Placebo + SoC#(N=42)	Drug A + SoC#(N=102)
3	>80	GASTROINTESTINAL DISORDERS	ABDOMINAL PAIN	0	1 (2.1)	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)
3	>80	GENERAL DISORDERS AND ADMINISTR...	ASTHENIA	0	1 (2.1)	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)
3	>80	MUSCULOSKELETAL AND CONNECTIVE ...	BACK PAIN	1 (3.3)	0	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)
3	>80	PSYCHIATRIC DISORDERS	AGITATION	1 (3.3)	0	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)

LET'S REPORT

STEP 6

```
PROC REPORT data=rep3 nowd missing split="#";
  BY agegr1n agegr1;
  COLUMNS aesoc aedecod nct0,ncol0 nct1,ncol1;

  DEFINE aesoc     / "SOC" order order=data _stylestatement;
  DEFINE aedecod   / "PT " _stylestatement;
  DEFINE nct:      / ""    _stylestatement;
  DEFINE ncol:     / "" across;

RUN;
```

For clarity, ODS style statements have been removed from the DEFINE statements above. The PROC REPORT is relatively straightforward: to assign the denominator labels to the frequency count variables we pair them in the COLUMNS statement. First, we reference the variable containing the AE frequency counts, followed by the respective variable containing the denominator label, with a comma to separate the two variables. In the DEFINE statements we set missing/null labels for both the frequency count variables (NCT) and denominator label variables (NCOL). We then apply the ACROSS function in the DEFINE statement for then NCOL variables containing denominator labels.

THE OUTPUT

Age group: <65

SOC	PT	Placebo + SoC (N=14)	Drug A + SoC (N=19)
CARDIAC DISORDERS	ATRIAL HYPERTROPHY	1 (7.1)	0
	ATRIOVENTRICULAR BLOCK SECOND DEGREE	1 (7.1)	0
	SINUS BRADYCARDIA	0	1 (5.3)
GASTROINTESTINAL DISORDERS	DIARRHOEA	3 (21.4)	1 (5.3)

Age group: 65-80

SOC	PT	Placebo + SoC (N=42)	Drug A + SoC (N=102)
CARDIAC DISORDERS	ATRIAL FIBRILLATION	0	3 (2.9)
	ATRIOVENTRICULAR BLOCK FIRST DEGREE	1 (2.4)	1 (1.0)
	BRADYCARDIA	1 (2.4)	0
	CARDIAC DISORDER	0	1 (1.0)
	MYOCARDIAL INFARCTION	1 (2.4)	4 (3.9)

Age group: >80

SOC	PT	Placebo + SoC (N=30)	Drug A + SoC (N=47)
CARDIAC DISORDERS	ATRIAL FIBRILLATION	1 (3.3)	1 (2.1)
	ATRIAL FLUTTER	0	2 (4.3)
	BUNDLE BRANCH BLOCK LEFT	1 (3.3)	0
	BUNDLE BRANCH BLOCK RIGHT	1 (3.3)	1 (2.1)
	CARDIAC FAILURE CONGESTIVE	1 (3.3)	0

The snippets above show the first few observations on the first page for each AGEGR1 value used as our BY variable. We can see that with a single call to PROC REPORT, our column labels have adapted to show the correct denominator labels for each BY group. If we compare the output with the dataset created in step 5, we can see the main body of the report (AESOC, AEDECOD, NCT0, NCT1) is represented in the output as it is in the dataset. This is what we would expect from a QC point of view. If we look at the values for NCOL we see that for each variable there is a single value within each variable. As there is a 1-1 correlation between the treatment label columns and BY variable value no transposition of the report body data takes place.

WHAT IF I HAVE SPANNING HEADERS?

To incorporate spanning headers, we can use the same technique as previously demonstrated. Let's imagine we want an updated version of the above table that splits each of the treatment columns into male and female, giving us four frequency count columns in total, but we still wanted to see total number of patients treated in each treatment group spanning across the sex columns. We will show how to do this in the following steps.

STEP 7

As in step 1. we create an extended dataset containing the denominator counts for each of the treatment, age group, sex combinations within the analysis set in variable NCOL. Additionally, we create another variable, NSPAN, to hold the counts for each treatment and age group combination needed for the spanning headers.

TRT01AN	TRT01A	AGEGR1	AGEGR1N	SEX	SEXN	NCOL	NSPAN
0	Placebo + SoC	<65	1	Female	1	9	14
0	Placebo + SoC	<65	1	Male	2	5	14
0	Placebo + SoC	65-80	2	Female	1	22	42
0	Placebo + SoC	65-80	2	Male	2	20	42
0	Placebo + SoC	>80	3	Female	1	22	30
0	Placebo + SoC	>80	3	Male	2	8	30
1	Drug A + SoC	<65	1	Female	1	10	19
1	Drug A + SoC	<65	1	Male	2	9	19
1	Drug A + SoC	65-80	2	Female	1	56	102
1	Drug A + SoC	65-80	2	Male	2	46	102
1	Drug A + SoC	>80	3	Female	1	24	47
1	Drug A + SoC	>80	3	Male	2	23	47

After processing to create labels from the treatment, age group and sex information, NCOL and NSPAN are transposed separately before being merged back together to create a labeling dataset with one observation per age group.

AGEGR1N	AGEGR1	NCOL01	NCOL02	NCOL11	NCOL12	SPAN0	SPAN1
1	<65	Sex: Female#(N=9)	Sex: Male#(N=5)	Sex: Female#(N=10)	Sex: Male#(N=9)	Placebo + SoC#(N=14)	Drug A + SoC#(N=19)
2	65-80	Sex: Female#(N=22)	Sex: Male#(N=20)	Sex: Female#(N=56)	Sex: Male#(N=46)	Placebo + SoC#(N=42)	Drug A + SoC#(N=102)
3	>80	Sex: Female#(N=22)	Sex: Male#(N=8)	Sex: Female#(N=24)	Sex: Male#(N=23)	Placebo + SoC#(N=30)	Drug A + SoC#(N=47)

Again here we have used the numeric treatment variable to identify our spanning headers. For the individual columns, we have concatenated the numeric treatment variable with the numeric sex variable. For example, NCOL01 contains the denominator counts for Female (SEXN=1) subjects assigned to Placebo + SoC (TRT01AN=0).

ALL TOGETHER NOW

STEP 8

After merging the AE frequency dataset with our denominator labels from step 7, our data is ready for reporting. In the snippet below we can see that the various NCOLxx columns display the correct denominator labels for each AGEGR1 BY variable, while the SPANx columns show the correct labels for each of our spanning headers.

AGEGR1N	AGEGR1	AESOC	AEDECOD	NCT01	NCT02	NCT11	NCT12	NCOL01	NCOL02	NCOL11	NCOL12
1	<65	GENER..	ASTHENIA	1 (11.1)	0	0	0	Sex: Female#(N=9)	Sex: Male#(N=5)	Sex: Female#(N=10)	Sex: Male#(N=9)
2	65-80	GASTR..	ABDOMINAL PAIN	1 (4.5)	0	2 (3.6)	1 (2.2)	Sex: Female#(N=22)	Sex: Male#(N=20)	Sex: Female#(N=56)	Sex: Male#(N=46)
2	65-80	MUSCU..	BACK PAIN	0	0	2 (3.6)	2 (4.3)	Sex: Female#(N=22)	Sex: Male#(N=20)	Sex: Female#(N=56)	Sex: Male#(N=46)
2	65-80	PSYCH..	AGITATION	1 (4.5)	0	3 (5.4)	0	Sex: Female#(N=22)	Sex: Male#(N=20)	Sex: Female#(N=56)	Sex: Male#(N=46)
3	>80	GASTR..	ABDOMINAL PAIN	0	0	1 (4.2)	0	Sex: Female#(N=22)	Sex: Male#(N=8)	Sex: Female#(N=24)	Sex: Male#(N=23)
3	>80	GENER..	ASTHENIA	0	0	1 (4.2)	0	Sex: Female#(N=22)	Sex: Male#(N=8)	Sex: Female#(N=24)	Sex: Male#(N=23)
3	>80	MUSCU..	BACK PAIN	1 (4.5)	0	0	0	Sex: Female#(N=22)	Sex: Male#(N=8)	Sex: Female#(N=24)	Sex: Male#(N=23)
3	>80	PSYCH..	AGITATION	1 (4.5)	0	0	0	Sex: Female#(N=22)	Sex: Male#(N=8)	Sex: Female#(N=24)	Sex: Male#(N=23)

LET'S REPORT (AGAIN)

STEP 9

```
PROC REPORT data=rep3 nowd missing split="#";
  BY agegr1n agegr1;
  COLUMNS aesoc aedecod (span0,(nct01,ncol01 nct02,ncol02)) gap
                        (span1,(nct11,ncol11 nct12,ncol12));

  DEFINE gap       / display " " computed style(column)=[cellwidth=1% padding=0];

  DEFINE aesoc     / "SOC" order order=data _stylestatement;
  DEFINE aedecod   / "PT " _stylestatement;
  DEFINE nct:      / ""    _stylestatement;

  DEFINE span:     / "" across style(header)=[borderbottomwidth=0.5];
  DEFINE ncol:     / "" across;
RUN;
```

Again the ODS style statements have been removed from the DEFINE statements above for clarity. This time, our PROC REPORT is somewhat more complex.

To add spanning headers, we enclose the columns we want to span in brackets in the COLUMNS statement, similar to how we would normally add a spanning header. However instead of placing a text label inside the brackets, we specify the variable that contains the spanning header. Follow this with a comma, then list the pairs of denominator labels and frequency count variables. Since we are displaying two frequency count variables under one spanning header, we enclose both variables in a single pair of brackets. As before, we separate our frequency count variable and its corresponding denominator label variable with a comma.

Within the DEFINE statements, we apply the ACROSS function to both the spanning headers and individual column labels, applying a missing/null label to all. For the spanning headers, we include a style attribute to add a border under the header that spans the entire cell. Since we have two separate spanning headers, we introduce a computed gap variable to create a small space between them.

THE OUTPUT

Age group: <65

		Placebo + SoC (N=14)		Drug A + SoC (N=19)	
		Sex: Female	Sex: Male	Sex: Female	Sex: Male
SOC	PT	(N=9)	(N=5)	(N=10)	(N=9)
CARDIAC DISORDERS	ATRIAL HYPERTROPHY	0	1 (20.0)	0	0
	ATRIOVENTRICULAR BLOCK SECOND DEGREE	0	1 (20.0)	0	0
	SINUS BRADYCARDIA	0	0	0	1 (11.1)
GASTROINTESTINAL DISORDERS	DIARRHOEA	2 (22.2)	1 (20.0)	1 (10.0)	0

Age group: 65-80

		Placebo + SoC (N=42)		Drug A + SoC (N=102)	
		Sex: Female	Sex: Male	Sex: Female	Sex: Male
SOC	PT	(N=22)	(N=20)	(N=56)	(N=46)
CARDIAC DISORDERS	ATRIAL FIBRILLATION	0	0	2 (3.6)	1 (2.2)
	ATRIOVENTRICULAR BLOCK FIRST DEGREE	1 (4.5)	0	0	1 (2.2)
	BRADYCARDIA	1 (4.5)	0	0	0
	CARDIAC DISORDER	0	0	0	1 (2.2)

Age group: >80

		Placebo + SoC (N=30)		Drug A + SoC (N=47)	
		Sex: Female	Sex: Male	Sex: Female	Sex: Male
SOC	PT	(N=22)	(N=8)	(N=24)	(N=23)
CARDIAC DISORDERS	ATRIAL FIBRILLATION	1 (4.5)	0	0	1 (4.3)
	ATRIAL FLUTTER	0	0	0	2 (8.7)
	BUNDLE BRANCH BLOCK LEFT	1 (4.5)	0	0	0
	BUNDLE BRANCH BLOCK RIGHT	1 (4.5)	0	0	1 (4.3)

Reviewing the snippets above, which show the first few observations on the first page for each AGEGR1 value used as our BY variable, we can confirm that the correct subject counts are shown in each column for each value of treatment and sex within each BY variable. Furthermore, the overall treatment numbers spanning the individual male and female columns are also correctly displayed.

TIDYING THINGS UP

Although throughout this paper we have used the numerical treatment group values as column IDs, in practice, we typically use a consistent naming convention such as COL1-COLx for our reporting output datasets. To align with this approach, we should adopt a similar naming convention for our ACROSS variable. For example, we could use HEADx, where x corresponds to the column number it is being used to label. In our initial example, the reporting dataset would then have the columns COL1, COL2, COL3, COL4, HEAD3 and HEAD4. Similarly, for spanning headers, we could use names like SPAN34 (or SPAN3_4) to represent the spanning header for COL3 and COL4.

CONCLUSION

In this paper, we have demonstrated how using ACROSS variables to manage the denominator labels we can simplify the creation of BY group tables. This approach eliminates the need for multiple PROC REPORT calls or repeated macro variables resolutions. Additionally, incorporating denominator labels as variables in the reporting dataset facilitates a more thorough QC process, allowing programmers to programmatically verify that the correct denominator are being displayed for each BY group.

This method is not limited to BY group processing, it can be applied to all table programming. In addition to enabling programmatic QC of the denominator labels, it makes programs more adaptable for future BY group outputs. If introduced in tables created with macros or template programs, a null BY variable can be added, so the only update required for BY group outputs is to populate the null BY variable with the appropriate BY values.

ACKNOWLEDGMENTS

Special thanks to Anna L for reviewing my first draft and providing suggestions to clarify various passages of text and my colleagues Arnold van Aswegen and Kath Wright for their reviews and support throughout this process.

CONTACT INFORMATION

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

Chris Spence
Cytel Inc
Email: chris.spence@cytel.com
Linkedin: https://linkedin.com/in/christopher-spence-1bbb6b67
Website: https://cytel.com/

APPENDIX I

Code used for creation of outputs without spanning headers
Data source: https://github.com/cdisc-org/sdtm-adam-pilot-project/tree/master/updated-pilot-submission-package/900172/m5/datasets/cdiscpilot01/analysis/adam/datasets

```
**----------------------------------------------------------------**;
** Get required data
**----------------------------------------------------------------**;
DATA adsl;
  LENGTH sex $10.;
  SET adam.adsl;
  WHERE saffl="Y";

  KEEP usubjid trt01a trt01an agegr1 agegr1n sex sexn;
  IF trt01an ne 0 THEN trt01an=1;
  IF trt01an=0 THEN trt01a="Placebo + SoC";
    ELSE trt01a="Drug A + SoC";

  IF sex="F" THEN sexn=1;
  ELSE IF sex="M" THEN sexn=2;
  IF sex="F" THEN sex="Female";
  ELSE IF sex="M" THEN sex="Male";
RUN;

DATA adae;
  SET adam.adae;
  KEEP usubjid aedecod aesoc trtemfl;
  WHERE trtemfl="Y";
RUN;

DATA all;
  MERGE adsl (in=inadsl) adae (in=inadae);
  BY usubjid;
  IF inadsl and inadae;
RUN;

**----------------------------------------------------------------**;
** Create shell of all possible by groups and treatments, then get patient counts.
**----------------------------------------------------------------**;
PROC SQL;
  CREATE TABLE pop0 AS SELECT
  distinct a.trt01an, a.trt01a,
  b.agegr1, b.agegr1n
  FROM (select distinct trt01an, trt01a from adsl) a,
       (select distinct agegr1, agegr1n from adsl) b;

  CREATE TABLE pop1 AS SELECT
  a.*, b.ncol
  FROM pop0 a
  LEFT JOIN (select distinct trt01an, agegr1, count(distinct(usubjid)) as ncol
```

```
            from adsl group by trt01an, agegr1) b
    ON a.trt01an=b.trt01an and a.agegr1=b.agegr1;
QUIT;

**----------------------------------------------------------------**;
** Create labels for treatments and across labels
**----------------------------------------------------------------**;
DATA pop2;
  SET pop1;
  IF ncol=. THEN ncol=0;
  var1=trim(trt01a)||"#(N="||compress(put(ncol, best.))||")";
RUN;

**----------------------------------------------------------------**;
** Transpose N counts by AGE group so we have one column per
** treatment, matching columns presented in report
**----------------------------------------------------------------**;
PROC SORT data=pop2;
 BY agegr1n;
RUN;

PROC TRANSPOSE data=pop2 out=popt1 (drop=_name_) prefix=ncol;
  BY agegr1n agegr1;
  VAR var1;
  ID trt01an;
RUN;

**----------------------------------------------------------------**;
** Count adverse events, by age, sex and treatment
** Merge N count
** Create n (%) column
**----------------------------------------------------------------**;
PROC SQL;
  CREATE TABLE count0 AS SELECT
  distinct aesoc, aedecod, trt01an, trt01a, agegr1, agegr1n,
  count(distinct(usubjid)) as npat
  FROM all
  GROUP BY aesoc, aedecod, trt01an, trt01a, agegr1, agegr1n;
QUIT;

PROC SQL;
  CREATE TABLE count1 AS SELECT
  a.*, b.ncol,
  put(npat, 3.)||" ("||put(100*npat/ncol, 5.1)||")" as npct
  FROM count0 a
  LEFT JOIN pop1 b
    ON a.trt01an=b.trt01an and a.agegr1n=b.agegr1n
  ORDER BY a.agegr1n, aesoc, aedecod;
QUIT;

**----------------------------------------------------------------**;
** Transpose to report orientation one col per treatment by age group
**----------------------------------------------------------------**;
PROC TRANSPOSE data=count1 out=rep1 (drop=_name_) prefix=nct;
  BY agegr1n agegr1 aesoc aedecod;
  VAR npct;
  ID trt01an;
RUN;

**----------------------------------------------------------------**;
** Add all treatment label variables
**----------------------------------------------------------------**;
DATA rep2;
  MERGE rep1 (in=inrep) pop4;
  BY agegr1n agegr1;
RUN;
```

```
**----------------------------------------------------------------**;
** Zero fill missing dataset point
**----------------------------------------------------------------**;
DATA rep3;
  SET rep;
  ARRAY zero(*) nct:;
  DO i=1 TO dim(zero);
    IF zero(i)="" THEN zero(i)="  0";
  END;
  DROP i;
RUN;

**----------------------------------------------------------------**;
** Run report and close rtf destination
**----------------------------------------------------------------**;
```

RTF options, titles, footnotes and open RTF code;

```
OPTIONS nobyline;

PROC REPORT data=rep3 nowd missing split="#";
  BY agegr1n agegr1;
  COLUMNS  aesoc aedecod nct0,ncol0 nct1,ncol1;


  DEFINE aesoc     / "SOC" order order=data _stylestatement_;
  DEFINE aedecod   / "PT" _stylestatement_;
  DEFINE nct:      / ""   _stylestatement_;

  DEFINE ncol:     / "" across;

RUN;
```

Close RTF code;

APPENDIX II

Code used for creation of outputs with spanning headers. Source as per Appendix I.

```
**----------------------------------------------------------------**;
** Get required data
**----------------------------------------------------------------**;
```

This section as per Appendix I

```
**----------------------------------------------------------------**;
** Create shell of all possible by groups and treatments, then get patient counts.
**----------------------------------------------------------------**;
PROC SQL;
  CREATE TABLE pop0 AS SELECT
  distinct a.trt01an, a.trt01a,
  b.agegr1, b.agegr1n,
  c.sex, c.sexn
  FROM (select distinct trt01an, trt01a from adsl) a,
       (select distinct agegr1, agegr1n from adsl) b,
       (select distinct sex, sexn from adsl) c;

  CREATE TABLE pop1 AS SELECT
  a.*, b.ncol, c.nspan
  FROM pop0 a
  LEFT JOIN (select distinct trt01an, sex, agegr1, count(distinct(usubjid)) as ncol
             from adsl group by trt01an, sex, agegr1) b
    ON a.trt01an=b.trt01an and a.agegr1=b.agegr1 and a.sex=b.sex
  LEFT JOIN (select distinct trt01an, agegr1, count(distinct(usubjid)) as nspan
             from adsl group by trt01an, agegr1) c
    ON a.trt01an=c.trt01an and a.agegr1=c.agegr1;
QUIT;

**----------------------------------------------------------------**;
** Create labels for treatments and across labels
**----------------------------------------------------------------**;
```

```
DATA pop2;
  SET pop1;
  IF ncol=.  THEN ncol=0;
  IF nspan=. THEN nspan=0;
  var1="Sex: "||trim(sex)||"#(N="||compress(put(ncol, best.))||")";
  var2=trim(trt01a)||"#(N="||compress(put(nspan, best.))||")";
RUN;

**------------------------------------------------------------------**;
** Transpose N counts by AGE group and sex so we have one column per
** treatment per sex, matching columns presented in report
**------------------------------------------------------------------**;
PROC SORT data=pop2;
 BY agegr1n;
RUN;

PROC TRANSPOSE data=pop2 out=popt1 (drop=_name_) prefix=ncol;
  BY agegr1n agegr1;
  VAR var1;
  ID trt01an sexn;
RUN;

**------------------------------------------------------------------**;
** Transpose N counts by AGE group so we have one column per treatment
** matching spanning headers in report
**------------------------------------------------------------------**;
PROC SORT data=pop2 out=pop3 nodupkey;
  BY agegr1n trt01an var2;
RUN;

PROC TRANSPOSE data=pop3 out=popt2 (drop=_name_) prefix=span;
  BY agegr1n agegr1;
  VAR var2;
  ID trt01an;
RUN;

**------------------------------------------------------------------**;
** Merge all N counts
**------------------------------------------------------------------**;
DATA pop4;
  MERGE popt1 popt2;
  BY agegr1n agegr1;
RUN;

**------------------------------------------------------------------**;
** Count adverse events, by age, sex and treatment
** Merge N count
** Create n (%) column
**------------------------------------------------------------------**;
PROC SQL;
  CREATE TABLE count0 AS SELECT
  distinct aesoc, aedecod, sex, sexn, trt01an, trt01a, agegr1, agegr1n,
           count(distinct(usubjid)) as npat
  FROM all
  GROUP BY aesoc, aedecod, sex, sexn, trt01an, trt01a, agegr1, agegr1n;
QUIT;

PROC SQL;
  CREATE TABLE count1 AS SELECT
  a.*, b.ncol,
  put(npat, 3.)||" ("||put(100*npat/ncol, 5.1)||")" as npct
  FROM count0 a
  LEFT JOIN pop1 b
    ON a.trt01an=b.trt01an and a.sexn=b.sexn and a.agegr1n=b.agegr1n
  ORDER BY a.agegr1n, aesoc, aedecod;
QUIT;
```

```
**----------------------------------------------------------------------**;
** Transpose to report orientation one col per treatment and sex by age group
**----------------------------------------------------------------------**;
PROC TRANSPOSE data=count1 out=rep1 (drop=_name_) prefix=nct;
  BY agegr1n agegr1 aesoc aedecod;
  VAR npct;
  ID trt01an sexn;
RUN;

**----------------------------------------------------------------------**;
** Add all counts
**----------------------------------------------------------------------**;
DATA rep2;
  MERGE rep1 (in=inrep) pop4;
  BY agegr1n agegr1;
RUN;

**----------------------------------------------------------------------**;
** Zero fill missing dataset point
**----------------------------------------------------------------------**;
DATA rep3;
  SET rep2;
  BY agegr1n;
  ARRAY zero(*) nct:;
  DO i=1 TO dim(zero);
    IF zero(i)="" THEN zero(i)="  0";
  END;
  DROP i;
RUN;

**----------------------------------------------------------------------**;
** Run report and close rtf destination
**----------------------------------------------------------------------**;
```

RTF options, titles, footnotes and open RTF code;

```
OPTIONS nobyline;

PROC REPORT data=rep3 nowd missing split="#";
  BY agegr1n agegr1;
  COLUMNS aesoc aedecod (span0,(nct01,ncol01 nct02,ncol02)) gap
                        (span1,(nct11,ncol11 nct12,ncol12));

  DEFINE gap  / display " " computed style(column)=[cellwidth=1% padding=0];

  DEFINE aesoc     / "SOC" order order=data _stylestatement_;
  DEFINE aedecod   / "PT" _stylestatement_;
  DEFINE nct:      / ""   _stylestatement_;

  DEFINE span:     / "" across style(header)=[borderbottomwidth=0.5];
  DEFINE ncol:     / "" across;

RUN;
```

Close RTF code;