MS Applied Biostatistics Practical Training Experience

Sara (Shiv) Denner

August 15th, 2025

Introduction:

For my internship experience, I worked as a Statistical Data Science and Analytics (SDSA) intern at Pfizer. The internship lasted 10 weeks (June 19 – August 15) and consisted of two phases. In phase 1, I worked on generating CDISC compliant Study Data Tabulation Model (SDTM) and Analysis Data Model (ADaM) datasets from the raw collection data (Case Report Form/CDASH) and I developed submission-ready Tables, Listings, and Figures (TLFs) corresponding to different CDISC domains. For these assignments, I performed data management, cleaning, and analysis in SAS.

In phase 2, I worked with a partner on adding functionality to an RShiny app meant to give statisticians and clinicians an easy way to monitor blinded clinical trial data from two vaccine studies. Certain data related issues occurred in past studies, causing downstream analysis problems that created rework and re-analysis. These data problems included individuals being assigned to multiple sites, enrollment of individuals who did not meet baseline criteria for the study, and different individuals having identical data. The overall purpose of the RShiny Blinded Data Monitoring app is to be able to monitor data integrity of blinded clinical trial data as it continuously updates, therefore allowing statisticians and clinicians to understand data characteristics before analysis occurs.

Statistical Methods:

To create SDTM & ADaM datasets (Phase I Projects 1 & 2), I used the data step in SAS to derive and change the format of variables, SQL to join datasets and bring variables together from different datasets, and used PROC COMPARE in SAS to validate the dataset I created against CDISC SDTM & ADaM datasets. I referred to the SDTM & ADaM implementation guides, medical coding dictionaries such as MedDRA and WHODrug, and the provided domain-specific specifications. PROC TRANSPOSE was often used to derive variables that required data reshaping, since it can convert rows to columns and columns to rows. This procedure was essential for turning wide datasets into long datasets and vice versa, allowing for data manipulation and derivation required in SDTM and ADaM datasets. PROC IMPORT was used to upload Excel files into SAS, such as Per Protocol population listings, that were needed for variable derivations. I also utilized the macro programming language to automate repeated code. An example PROC COMPARE and accompanying SAS Code for an ADaM dataset are listed in the Appendix.

To generate tables and listings (Project 3), PROC FREQ was used to generate counts of subjects who belonged to certain criteria, and PROC TRANSPOSE was again used to reshape the data. SQL was used to generate counts and combine datasets. In other domains, PROC MEANS was used to generate statistics like mean, median, standard deviation, minimum, and maximum. Finally, PROC REPORT was used to format the summary statistics into tables and listings that abided by production specifications. An example table and accompanying SAS Code for are listed in the Appendix.

For the RShiny Blinded Data Monitoring dashboard in Phase 2, simulated case report form (CRF) and non-CRF datasets were created to be able to run a base app created by a Pfizer statistician, as real study data had not yet been captured. To simulate these datasets, bootstrapping was used to sample 300 unique subject IDs from the demographic information of a previous vaccine study. Base data for each subject was obtained from ADaM datasets for the previous study. The simulated data was made to exactly match the characteristics of the study data by creating variables, renaming variables, ensuring no duplicate subject ID's, and manipulating data via the tidyverse in R. To visualize each subject's viral load, symptom and viral load data were simulated up to Day 28 of follow up. Once complete, the simulated data was joined and converted to Excel files. Correctly simulated data allowed us to view and interact with the unfinished app, and at this point additional functionality was added. This experience taught me how to program in R Shiny, solidified my base-R and tidyverse coding skills, sharpened my skills in creating data visualizations using the plotly and ggplot2 packages, allowed me to get experience with collaborating with statisticians & clinicians, and exposed me to bootstrapping methods in a real-world setting.

Results:

In Phase I, I successfully mapped and validated all the required and expected variables of the SDTM AE and DM datasets with no discrepancies noted in the PROC COMPARE results. I also mapped and validated all required and expected variables from the ADaM ADAE and ADSL datasets with no discrepancies noted in the PROC COMPARE results. Similar results were found for Project 3, where my output matched the required tables and listings for the ADMH, ADVS, and ADSL domains to the production ready provided tables and listings.

In Phase 2, various functionalities and tabs were added to the existing app that enhanced the user experience and made it come alive. For example, in the "Visualizations and Summary: Viral load and Serology" tab, box plots of viral load in the patient population were enhanced by adding stratification by worst observed symptom at baseline. This feature was added to allow clinicians to understand varying quantitative measurements of infection across groups of patients who had varying severity of symptoms and visualize if any subjects had no symptoms at baseline, indicating data quality issues. A "Primary Events" tab was added, which allowed users to choose various plots from an interactive drop-down menu, each of which showed visualizations of the proportions of subjects who met the novel primary endpoint of the study. Plots generated in the "Primary Events" tab included stacked bar charts (both non-stratified and stratified by country), dynamite plots stratified by country, dot plots, and contingency tables. More plots and functionalities may be added to the app following the writeup of this report.

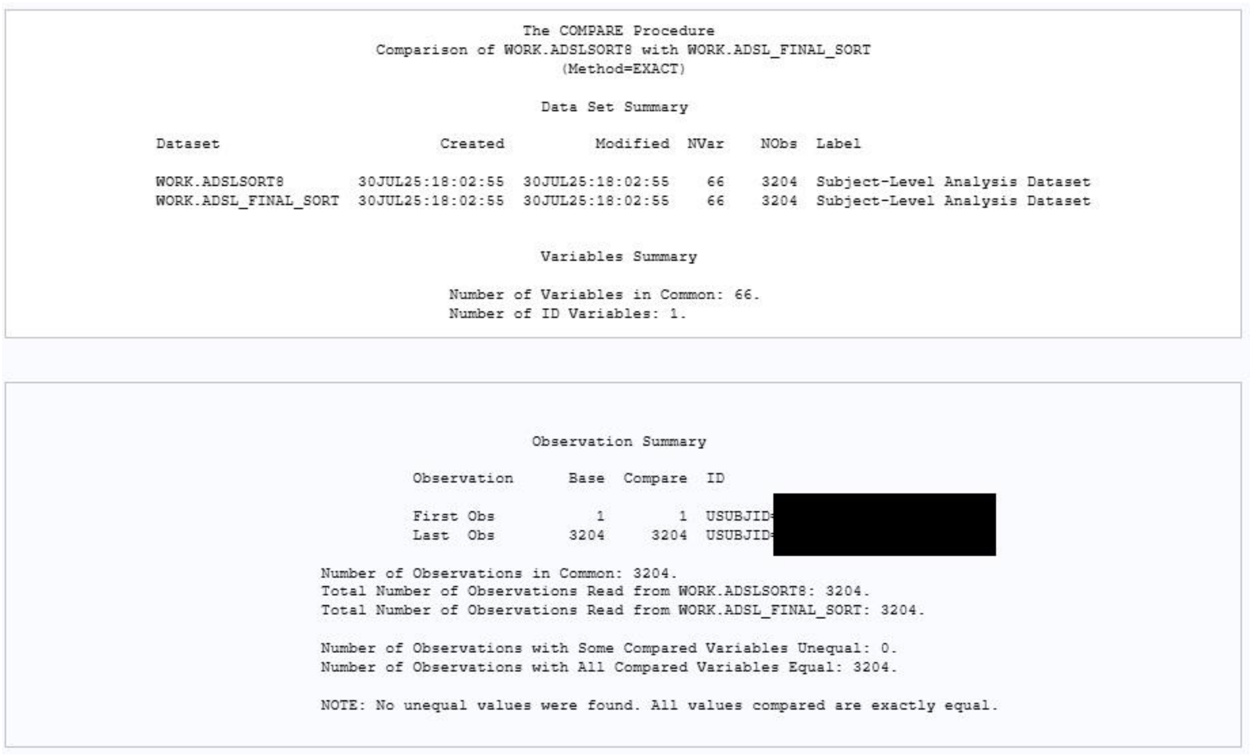
Discussion:

This first phase of this internship developed my understanding of data management and data cleaning using SAS. In the MSAB academic portion of the program, SAS was used mainly for analysis, while this experience introduced me to data manipulation in the data step, new PROCs related to reporting and validation, SQL for data extraction, and macros to automate repeated code. I also learned about CDISC standards, medical coding dictionaries, and programming in clinical trials. I also learned about Linux commands to navigate internal databases.

The second phase of the internship was helpful for developing my programming skills in R and RShiny. During the MSAB, I used mainly base R and packages suggested in classes geared towards analysis. During Phase 2, I became very comfortable with tidyverse and dplyr, and became very familiar with programming pipelines for data management, manipulation, and cleaning. I also practically applied bootstrapping methods to clinical trials datasets, which solidified what we learned about statistical simulation from the MSAB. The experience collaborating with statisticians and clinicians to develop an app helped me sharpen my skills in customer service, cross-functional collaboration, and time management. Working in a team gave me the opportunity to learn from my team member, get exposed to different ways of thinking about programming, and pushed me to perform at my best. Overall, I've had a wonderful time as an SDSA Intern at Pfizer and believe I have gained great experience that I can bring along with me as I begin my career.

Appendix: Samples of Output and SAS Code

Figure 1 – Phase I Project 2 PROC COMPARE Example



```
                                    The COMPARE Procedure
                          Comparison of WORK.ADSLSORT8 with WORK.ADSL_FINAL_SORT
                                        (Method=EXACT)

                                       Data Set Summary

        Dataset                       Created          Modified  NVar    NObs  Label

        WORK.ADSLSORT8       30JUL25:18:02:55  30JUL25:18:02:55    66    3204  Subject-Level Analysis Dataset
        WORK.ADSL_FINAL_SORT 30JUL25:18:02:55  30JUL25:18:02:55    66    3204  Subject-Level Analysis Dataset


                                       Variables Summary

                              Number of Variables in Common: 66.
                              Number of ID Variables: 1.



                                      Observation Summary

                           Observation      Base  Compare  ID

                           First Obs           1        1  USUBJID=[redacted]
                           Last  Obs        3204     3204  USUBJID=

                    Number of Observations in Common: 3204.
                    Total Number of Observations Read from WORK.ADSLSORT8: 3204.
                    Total Number of Observations Read from WORK.ADSL_FINAL_SORT: 3204.

                    Number of Observations with Some Compared Variables Unequal: 0.
                    Number of Observations with All Compared Variables Equal: 3204.

                    NOTE: No unequal values were found. All values compared are exactly equal.
```

Table 1 – Project 3 Example Table

Treatment Protocol XXX

Subject Evaluation Groups

	TRTA (N=85)	TRTB (N=71)	TRTC (N=89)	Total (N=245)
	n (%)	n (%)	n (%)	n (%)
Screened: 514				
Screen Failure: 257				
Other Screened but not Randomized: 12				
Randomized	85 (100.0)	71 (100.0)	89 (100.0)	245 (100.0)

	TRTA (N=85)	TRTB (N=71)	TRTC (N=89)	Total (N=245)
	n (%)	n (%)	n (%)	n (%)
Treated	85 (100.0)	71 (100.0)	89 (100.0)	245 (100.0)
Not Treated	0 (0.0)	0 (0.0)	0 (0.0)	0 (0.0)
Safety Population	85 (100.0)	71 (100.0)	89 (100.0)	245 (100.0)
ITT Population	85 (100.0)	71 (100.0)	89 (100.0)	245 (100.0)
Per-Protocol Population	74 (87.1)	63 (88.7)	83 (93.3)	220 (89.8)
Number of subjects[1]				
Completed Treatment Phase	72 (84.7)	63 (88.7)	81 (91.1)	216 (88.2)
Completed Safety Follow-Up	67 (78.8)	58 (81.7)	71 (79.8)	196 (80.0)
Discontinued Safety Follow-Up	5 (5.9)	5 (7.0)	10 (11.2)	20 (8.2)
Did not enter Safety Follow-Up	0 (0.0)	0 (0.0)	0 (0.0)	0 (0.0)
Discontinued Treatment Phase	13 (15.3)	8 (11.3)	8 (9.0)	29 (11.8)
Completed Safety Follow-Up	4 (4.7)	2 (2.9)	1 (1.1)	7 (2.9)
Discontinued Safety Follow-Up	5 (5.9)	4 (5.6)	1 (1.1)	10 (4.1)
Did not enter Safety Follow-Up	4 (4.7)	2 (2.9)	6 (6.7)	12 (4.9)
Completed Study	71 (83.5)	60 (84.5)	72 (80.9)	203 (82.8)
Discontinued Study	14 (16.5)	11 (15.5)	17 (19.1)	42 (17.1)
Rollover to Study XXXX	0 (0.0)	1 (1.4)	2 (2.2)	3 (1.2)

SAS Code: Figure 1 Phase I Project 2

```
/* Define Libraries */
libname XXXXXXXXXXXXXXXX access = readonly;
libname XXXXXXXXXXXXXXXX access= readonly;
libname XXXXXXXXXXXXXXXX;
libname XXXXXXXXXXXXXXXX access = readonly;
libname XXXXXXXXXXXXXXXX;

/* Clean working directory */
/* proc datasets lib = work kill memtype = data nowarn; */
/* run; */

/*********** DM DERIVED AND DIRECT MOVES *************/

data dmvars;

      attrib STUDYID label = 'Study Identifier'
                   USUBJID label = 'Unique Subject Identifier'
                   SUBJID label = 'Subject Identifier for the Study'
                   SITEID label = 'Study Site Identifier'
                   AGE label = 'Age'
                   AGEU label = 'Age Units'
                   AGEGR1 label = 'Pooled Age Group 1' length = $100 /*SET LENGTHS
TO STOP TRUNCATION*/
                   AGEGR1N label = 'Pooled Age Group (N)' length  = 8
                   AGEGR2 label = 'Pooled Age Group 2' length = $100
                   AGEGR2N label = 'Pooled Age Group 2 (N)' length = 8
                   AAGE label = 'Analysis Age'
                   AAGEU label = 'Analysis Age Unit'
                   SEX label = 'Sex'
                   SEXN label = 'Sex (N)'
                   ETHNIC label = 'Ethnicity'
                   RACE label = 'Race'
                   RACEN label = 'Race (N)'
                   COUNTRY label = 'Country'
                   SAFFL label = 'Safety Population Flag'
                   DTHFL label = 'Subject Death Flag'
                   ARM label = 'Description of Planned Arm' length = $200
                   ARMCD label = 'Planned Arm Code' length = $20
                   ACTARM label = 'Description of Actual Arm' length = $200
                   ACTARMCD label = 'Actual Arm Code' length = $20
                   TRT01P label = 'Planned Treatment for Period 01' length = $200
                   TRT01A label = 'Actual Treatment for Period 01' length = $200
                   BRTHDT label = 'Date of Birth' format = DATE9.
                   BRTHDTF label = 'Date of Birth Imput. Flag'
                   DTHDT label = 'Date of Death'
```

```
             RFSTDT label = 'Subject Reference Start Date' format = DATE9.
             RFSTTM label = 'Subject Reference Start Time' format = TIME8.
             RFENDT label = 'Subject Reference End Date' format = DATE9.
             RFENTM label = 'Subject Reference End Time' format = TIME8.
             RFENDT label = 'Subject Reference End Date' format = DATE9.
             RFENTM label = 'Subject Reference End Time' format = TIME8.
             RFICDT label = 'Date of Informed Consent' format = DATE9.
             RFPENDT label = 'Date of End of Participation' format = DATE9.
             INVID label = 'Investigator Identifier'
             INVNAM label = 'Investigator Name'
             DMDY label = 'Study Day of Collection'
             DMDT label = 'Date of Demog Collection' format = DATE9.
             ;
set XXXXXX;

/*Direct Moves*/

STUDYID = STUDYID;
USUBJID = USUBJID;
SUBJID = SUBJID;
SITEID = SITEID;
AGE = AGE;
AGEU = AGEU;
AAGEU = AGEU;
SEX = SEX;
ETHNIC = ETHNIC;
RACE = RACE;
COUNTRY = COUNTRY;
DTHFL = DTHFL;
ARM = ARM;
ARMCD = ARMCD;
INVID = INVID;
INVNAM = INVNAM;
DMDY = DMDY;

/* If/then */

*AGEGR1;
if 0 < AGE < 18 then AGEGR1 = '<18';
      else if 18 <= AGE <= 44 then AGEGR1 = '18-44';
      else if 45 <= AGE <= 64 then AGEGR1 = '45-64';
      else if AGE > 64 then AGEGR1 = '>=65';
      else if AGE = . then AGEGR1 = 'UNSPECIFIED';

*AGEGR1N;
if AGEGR1 = "<18" then AGEGR1N = 1;
      else if AGEGR1 = "18-44" then AGEGR1N = 2;
```

```
              else if AGEGR1 = "45-64" then AGEGR1N = 3;
              else if AGEGR1 = ">=65" then AGEGR1N = 4;
              else if AGEGR1 = "UNSPECIFIED" then AGEGR1N = 5;

       *AGEGR2;
       if .<AGE<18 then AGEGR2 = '<18';
              else if 18<AGE<=44 then AGEGR2 = '18-44';
              else if 45<=AGE<= 64 then AGEGR2 = '45-64';
              else if 65 <= AGE <= 74 then AGEGR2 = '65-74';
              else if AGE > 74 then AGEGR2 = '>=75';

       *AGEGR2N;
       if AGEGR2 = "<18" then AGEGR2N = 1;
              else if AGEGR2 = "18-44" then AGEGR2N = 2;
              else if AGEGR2 = "45-64" then AGEGR2N = 3;
              else if AGEGR2 = "65-74" then AGEGR2N = 4;
              else if AGEGR2 = ">=75" then AGEGR2N = 5;

       *SEXN;
       if SEX = 'M' then SEXN = 1;
              else if SEX = 'F' then SEXN = 2;

       *RACEN;
       if RACE = "WHITE" then RACEN = 1;
              else if RACE = "BLACK OR AFRICAN AMERICAN" then RACEN = 2;
              else if RACE = "ASIAN" then RACEN = 3;
              else if RACE = "OTHER" then RACEN = 4;
              else if RACE = "" then RACEN = .;

       *SAFFL;
       if RFSTDTC NE "" then SAFFL = "Y";
              else SAFFL = "N";

       *ACTARM;
       if (RFSTDTC = "" and ACTARMCD not IN("NOTASSGN","SCRNFAIL")) then ACTARM =
"Not Treated";
              else ACTARM = ACTARM;

       *ACTARMCD;
       if (RFSTDTC = "" and ACTARMCD not in ("NOTASSGN","SCRNFAIL")) then ACTARMCD =
"NOTTRT";
              else ACTARMCD = ACTARMCD;

       *TRT01P;
       if ARM in('Screen Failure', 'Not Assigned') then TRT01P = "";
              else TRT01P = ARM;
```

```
	*TRT01A;
	if ACTARM in('Screen Failure', 'Not Assigned', 'Not Treated') then TRT01A = 
"";
		else TRT01A = ACTARM;

	*DTHDT - only three instances not missing, none are partial;
	if DTHDTC ne "" then DTHDT = input(DTHDTC, IS8601DA.);

	/* DERIVED */

	*AAGE;
	AAGE = int((input(DMDTC, YYMMDD10.) - input(BRTHDTC, YYMMDD10.)+1)/365.25);

	*BRTHDT - proc freq showed no partial dates;
	BRTHDT = input(BRTHDTC, IS8601DA.);

	*BRTHDTF - since no imputation, set to missing;
	BRTHDTF = "";

	*RFSTDT - take first 10 characters from string, put into new variable, convert 
variable to date9.;
	if not missing(RFSTDTC) then
		RFSDATE = substr(RFSTDTC, 1,10);
		RFSTDT = input(RFSDATE, YYMMDD10.);
	drop RFSDATE;

	*RFSTTM - take last 8 characters from string, put into new variable, convert 
to time8.;
	if not missing(RFSTDTC) then
		RFSTIME = substr(RFSTDTC, 12, 8);
		RFSTTM = input(RFSTIME, TIME8.);
	drop RFSTIME;

	*RFENDT;
	if not missing(RFENDTC) then
		RFENDATE = substr(RFENDTC, 1, 10);
		RFENDT = input(RFENDATE, YYMMDD10.);
	drop RFENDATE;

	*RFENTM;
	if not missing(RFENDTC) then
		RFENTIME = substr(RFENDTC, 12, 8);
		RFENTM = input(RFENTIME, TIME8.);
	drop RFENTIME;

	*RFICDT;
	if not missing(RFICDTC) then
```

```
		RFICDATE = substr(RFICDTC, 1, 10);
		RFICDT = input(RFICDATE, YYMMDD10.);
	drop RFICDATE;

	*RFPENDT;
	if not missing(RFPENDTC) then
		RFPENDT = input(RFPENDTC, YYMMDD10.);

	*DMDT;
	if not missing(DMDTC) then
		DMDT = input(DMDTC, YYMMDD10.);

	/*KEEP STATEMENT*/
	keep STUDYID USUBJID SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE
AAGEU SEX SEXN ETHNIC RACE
		RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
		RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT;
run;



/********* SUPPDM AND DM VARIABLES ***********/

* Use proc transpose to derive variables with QVAL and QNAM;

proc transpose data = XXXX out = XXXX_(drop = _name_ _label_);
	by STUDYID USUBJID IDVARVAL;
	id QNAM; *becomes column name;
	var QVAL; *becomes rows;
run;

data dmmerge;
	merge dmvars(in = A) suppdm_(in = B);
	by USUBJID;
run;

/*Create SUPPDM Variables within the DM variables dataset*/

data dmmerge1;
	set dmmerge;
	attrib RACEOTH label = "Specify Other Race"
			DMPROTID label = "Protocol ID"
			DMCENTER label = "Center ID"
			RACEGR1 label = "Pooled Race Group 1" length = $100
			RACEGR1N label = "Pooled Race Group 1 (N)"
			ARACE label = 'Analysis Race'
```

```
             ;

*instead of qnam, use raceoth;

       /* Direct Moves */

       RACEOTH = RACEOTH;
       DMPROTID = DMPROTID;
       DMCENTER = DMCENTER;

       /* Derivations */

       *RACEGR1;
       if RACE = "WHITE" then RACEGR1 = "WHITE";
             else if RACE = "BLACK OR AFRICAN AMERICAN" then RACEGR1 = "BLACK";
             else if RACE = "ASIAN" then RACEGR1 = "ASIAN";
             else if RACEOTH = "HISPANIC" or ETHNIC = "HISPANIC OR LATINO"
                    then RACEGR1 = "HISPANIC";
             else if RACE = "" then RACEGR1 = "UNSPECIFIED";

       *RACEGR1N;
       if RACEGR1 = "WHITE" then RACEGR1N = 1;
                    else if RACEGR1 = "BLACK" then RACEGR1N = 2;
                    else if RACEGR1 = "ASIAN" then RACEGR1N = 3;
                    else if RACEGR1 = "HISPANIC" then RACEGR1N = 4;
                    else if RACEGR1 = "UNSPECIFIED" then RACEGR1N = 999;

       *ARACE;
       if RACE ne "" then ARACE = RACE;
                    else if RACEOTH ne "" then ARACE = "OTHER";
                    else ARACE = "UNKNOWN";

       keep STUDYID USUBJID SUBJID SITEID AGE AGEU AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE
AAGEU SEX SEXN ETHNIC RACE
             RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
             RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE;

run;


/************** EX VARIABLES ********************/

*min of EXSTDTC;
proc sql;
       create table minfirsttrt as
```

```
              select USUBJID, min(EXSTDTC) as MIN_EXSTDTC
              from sdtm.ex
              group by USUBJID;
quit;

*max of EXENDTC;
proc sql;
       create table maxlasttrt as
              select USUBJID, max(EXENDTC) as MAX_EXENDTC
              from sdtm.ex
              group by USUBJID;
quit;

*join minfirsttrt & maxlasttrt & merge with EX;

proc sort data = minfirsttrt out = min1trtsort nodupkey dupout = test;
       by USUBJID;
run;

proc sort data = maxlasttrt out = maxlasttrtsort nodupkey dupout = test;
       by USUBJID;
run;

proc sort data = XXXX out = exsort nodupkey dupout = test;
       by USUBJID;
run;

proc sql;
       create table minmax as
              select A.*, B.MIN_EXSTDTC, C.MAX_EXENDTC
              from exsort as A
              inner join min1trtsort as B
                     on A.USUBJID = B.USUBJID
              inner join maxlasttrtsort as C
                     on A.USUBJID = C.USUBJID;
quit;

data EXVARS;
       set MINMAX;
       attrib TRTSDT label = 'Date of First Exposure to Treatment' format = DATE9.
                     TRTSTM label = 'Time of First Exposure to Treatment' format =
TIME8.
                     TRTEDT label = 'Date of Last Exposure to Treatment' format =
DATE9.
                     TRTETM label = 'Time of Last Exposure to Treatment' format =
TIME8.
                     TR01SDTM label = 'Datetime of First Exposure in Period 01' format
```

```
= DATETIME20.
                    TR01SDT label = 'Date of First Exposure in Period 01' format =
DATE9.
                    TR01STM label = 'Time of First Exposure in Period 01' format =
TIME8.
                    TR01EDTM label = 'Datetime of Last Exposure in Period 01' format
= DATETIME20.
                    TR01EDT label = 'Date of Last Exposure in Period 01' format =
DATE9.
                    TR01ETM label = 'Time of Last Exposure in Period 01' format =
TIME8.
       ;

       *TRTSDT;
       if not missing(MIN_EXSTDTC) then
              TRTSDT0 = substr(MIN_EXSTDTC, 1, 10);
              TRTSDT = input(TRTSDT0, YYMMDD10.);
       drop TRTSDT0;

       *TRTSTM;
       if not missing(MIN_EXSTDTC) then
              TRTSTM0 = substr(MIN_EXSTDTC, 12, 8);
              TRTSTM = input(TRTSTM0, TIME8.);
       drop TRTSTM0;

       *TRTEDT;
       if not missing(MAX_EXENDTC) then
              TRTEDT0 = substr(MAX_EXENDTC, 1, 10);
              TRTEDT = input(TRTEDT0, YYMMDD10.);
       drop TRTEDT0;

       *TRTETM;
       if not missing(MAX_EXENDTC) then
              TRTETM0 = substr(MAX_EXENDTC, 12, 8);
              TRTETM = input(TRTETM0, TIME8.);
       drop TRTETM0;

       *TR01SDTM;
       if not missing(EXSTDTC) then
              TR01SDTM = input(MIN_EXSTDTC, E8601DT.);

       *TR01SDT;
       if not missing(EXSTDTC) then
              TR01SDT = TRTSDT;

       *TR01STM;
       if not missing(EXSTDTC) then
```

```
            TR01STM = TRTSTM;

      *TR01EDTM;
      if not missing(EXENDTC) then
            TR01EDTM = input(MAX_EXENDTC, e8601dt.);

      *TR01EDT;
      if not missing(EXENDTC) then
            TR01EDT = TRTEDT;

      *TR01ETM;
      if not missing(EXENDTC) then
            TR01ETM = TRTETM;

      keep USUBJID TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM
TR01EDT TR01ETM;

run;


/* Merge DMMERGE1 and EXVARS */

data dmex;
      merge dmmerge1(in = A) exvars(in = B);
      by USUBJID;
run;

proc sort data = dmex out = dmexsort;
      by USUBJID;
run;

/************** DS & SUPPDS VARIABLES ***************/

* Create a copy of DS dataset and keep only variables required for derivation;
data ds0;
      set XXXX;
      format ;
      informat ;
      keep DSDECOD USUBJID DSCAT DSSTDTC;
run;

*RANDFL;
proc sql;
      create table randfl0 as
            select USUBJID,
                  max(case when DSDECOD = "RANDOMIZED" then "Y"
                  else "N"
```

```
                    end) as RANDFL
             from ds0
             group by USUBJID;
quit;

proc sort data = randfl0 out = randfl0sort;
      by USUBJID;
run;

data dmexds1;
      merge randfl0sort(in = A) dmexsort(in = B);
      by USUBJID;
run;

* Derive ITTFL;
data dmexds2;
      set dmexds1;
      attrib RANDFL label = "Randomized Population Flag"
                    ITTFL label = "Intent-To-Treat Population Flag"
      ;

      /* If/Then */

      *ITTFL;
      if (RANDFL = "Y" and SAFFL = "Y") then ITTFL = "Y";
             else ITTFL = "N";
run;

proc sort data = dmexds2 out = dsdmex_sort_ (keep = STUDYID USUBJID SUBJID SITEID AGE
AGEU AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE AAGEU SEX SEXN ETHNIC RACE
             RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
             RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE
             TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM TR01EDT
TR01ETM RANDFL ITTFL);
      by USUBJID;
run;

* Load "Per Protocol Population" Sheet;

options validvarname = v7;
proc import
      datafile = "XXXXXXXXXXXXXX"
      dbms = xlsx
      out = work.ppp
      replace;
```

```
       getnames = yes;
       range = "Per Protocol Population$A4:D89";
run;

data _ppp;
       attrib STUDYID length = $8 label = "Study Identifier"
                     USUBJID length = $22 label = "Unique Subject Identifier"
                     SUBJECT_ID length = $8 format = $9. informat = $10. label =
"Subject Identifier for the Study"
                     CENTER_ID length = $4 label = "Study Site Identifier"
                     PD_CATEGORY length = $19 format = $19. informat = $19.
                     PD_SUB_CATEGORY length = $135 format = $135. informat = $135.
                     ;
       set ppp;

       *STUDYID;
       STUDYID = "XXXXXXX";

       *SITEID;
       SITEID = CENTER_ID;

       *SUBJID;
       SUBJID = SUBJECT_ID;

       *PD_CAT;
       PD_CAT = PD_CATEGORY;

       *PD_SUBCAT;
       PD_SUBCAT = PD_SUB_CATEGORY;

       *USUBJID;
       USUBJID = catx(" ", STUDYID, SITEID, SUBJID);

       keep STUDYID USUBJID SITEID SUBJID PD_CAT PD_SUBCAT;
run;


* Sort by USUBJID and retain only USUBJID;
proc sort data = _ppp (keep = USUBJID) out = pppsort nodupkey dupout = ppptest;
       by USUBJID;
run;

proc sort data = dsdmex_sort_ out = dsdmex_sort;
       by USUBJID;
run;

proc sql;
```

```
	create table dmexds_PPROTFL as
	select a.*,
		case
			when a.ITTFL = "N" then "N"
			when a.ITTFL = "Y" and b.USUBJID is not null then "N"
			when a.ITTFL = "Y" and b.USUBJID is null then "Y"
			else ""
		end as PPROTFL
	from dsdmex_sort as a
	left join pppsort as b
	on a.USUBJID = b.USUBJID;
quit;


proc sort data = dmexds_PPROTFL out = pprotfl_sort (keep = STUDYID USUBJID SUBJID
SITEID AGE AGEU AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE AAGEU SEX SEXN ETHNIC RACE
		RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
		RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE
		TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM TR01EDT
TR01ETM RANDFL ITTFL PPROTFL);
	by USUBJID;
run;

*** Merge DS and SUPPDS;

data suppds;
	set sdtm.suppds;
run;

proc transpose data=suppds out=suppds_(drop=_name_ _label_);
	by STUDYID USUBJID IDVARVAL;
	id QNAM; *becomes column name;
	var QVAL; *becomes rows;
run;

*Rename IDVARVAL to DSSEQ;
data suppds1;
	attrib DSSEQ length = 8;
	set suppds_;

	*rename IDVARVAL to DSSEQ;
	DSSEQ = IDVARVAL;

	drop IDVARVAL;
run;
```

```
data suppds1;
      merge suppds1(in = A) sdtm.ds(in = B);
      by USUBJID STUDYID DSSEQ;
run;

* COMPFL;
data suppds2(keep = USUBJID COMPLFL COMPLDT) suppds3(keep = USUBJID RANDDT RANDNO)
     suppds4(keep = USUBJID EOSDCDT EOSDCRS) suppds5(keep = USUBJID EOTDCDT EOTDCRS);

      attrib COMPLFL label = "Completers Population Flag"
                    COMPLDT label = "Date of Completion/Withdrawal" format = date9.
                    RANDDT label = "Date of Randomization" format = date9.
                    EOSDCDT label = "End of Study Date" format = date9.
                    EOSDCRS label = "End of Study Reason"
                    EOTDCDT label = "End of Treatment Date" format = date9.
                    EOTDCRS label = "End of Treatment Reason"
      ;
      set suppds1;

      *Derive COMPFL & COMPLDT;
      if (DSCAT = "DISPOSITION EVENT" and DSPHASE = "STUDY"
             and DSDECOD = "COMPLETED") then do;
                    COMPLFL = "Y";
                    COMPLDT = input(DSSTDTC, YYMMDD10.);
             end;

      *Keep only non-missing records for easy merging;
      if not missing(COMPLFL) then output suppds2;

      *RANDDT;
      if DSDECOD = "RANDOMIZED" then do;
             RANDDT = input(DSSTDTC, YYMMDD10.);

      *RANDNO;
             if DSCAT = "PROTOCOL MILESTONE" and DSDECOD = "RANDOMIZED" then
                    RANDNO = DSREFID;
                    output suppds3;
      end;

      *EOSDCDT;
      if DSPHASE = "STUDY" then do;
             if DSSTDTC ne "" then EOSDCDT = input(DSSTDTC, YYMMDD10.);
             else if DSDTC ne "" then EOSDCDT = input(DSDTC, YYMMDD10.);

      *EOSDCRS;
             EOSDCRS = DSDECOD;
```

```
			output suppds4;
		end;

		*EOTDCDT;
		if DSPHASE = "TREATMENT" then do;
			if DSSTDTC NE "" then EOTDCDT = input(DSSTDTC, yymmdd10.);
			else if DSDTC NE "" then EOTDCDT = input(DSDTC, yymmdd10.);

		*EOTDCRS;
			EOTDCRS = DSDECOD;
			output suppds5;
		end;

run;

** Merge;

data adsl;
		merge pprotfl_sort(in = A) suppds2(in = B) suppds3(in = C)
					suppds4(in = D) suppds5(in = E);
		by USUBJID;

		*Finish COMPLFL;
		if missing(COMPLFL) then
			COMPLFL = "N";
run;

* Sort production dataset;

proc sort data = adsl out = adsl_sort (keep = STUDYID USUBJID SUBJID SITEID AGE AGEU
AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE AAGEU SEX SEXN ETHNIC RACE
			RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
			RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE
			TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM TR01EDT
TR01ETM RANDFL ITTFL PPROTFL
			COMPLFL COMPLDT RANDDT RANDNO EOSDCDT EOSDCRS EOTDCDT EOTDCRS);
		by USUBJID;
run;


/************** Cleaning the final Dataset ********************/

data adsl_final(label = "Subject-Level Analysis Dataset");
		attrib RACEGR1 length = $100
					ARACE length = $200
```

```
                    PPROTFL label = "Per-Protocol Population Flag"
                    RANDNO label = "Randomization Number"
                    EOSDCRS length = $100
                    EOTDCRS length = $100
                    DTHDT format = DATE9.
      ;
      set adsl_sort;
run;


/************** Proc Compare ******************/

proc sort data = adsl_final out = adsl_final_sort (keep = STUDYID USUBJID SUBJID
SITEID AGE AGEU AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE AAGEU SEX SEXN ETHNIC RACE
             RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
             RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE
             TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM TR01EDT
TR01ETM RANDFL ITTFL PPROTFL
             COMPLFL COMPLDT RANDDT RANDNO EOSDCDT EOSDCRS EOTDCDT EOTDCRS);
      by USUBJID;
run;

proc sort data = XXXX out = adslsort8 (keep = STUDYID USUBJID SUBJID SITEID AGE AGEU
AGEGR1 AGEGR1N AGEGR2 AGEGR2N AAGE AAGEU SEX SEXN ETHNIC RACE
             RACEN COUNTRY SAFFL DTHFL ARM ARMCD ACTARM ACTARMCD TRT01P TRT01A BRTHDT
BRTHDTF DTHDT RFSTDT RFSTTM
             RFENDT RFENTM RFENDT RFENTM RFICDT RFPENDT INVID INVNAM DMDY DMDT
RACEOTH RACEGR1 RACEGR1N DMPROTID DMCENTER ARACE
             TRTSDT TRTSTM TRTEDT TRTETM TR01SDTM TR01SDT TR01STM TR01EDTM TR01EDT
TR01ETM RANDFL ITTFL PPROTFL
             COMPLFL COMPLDT RANDDT RANDNO EOSDCDT EOSDCRS EOTDCDT EOTDCRS);
      by USUBJID;
run;

proc compare base = adslsort8 compare = adsl_final_sort listobs listvar listall
                    out = compFINAL_ADSL outnoeq outcomp outbase;
                    id USUBJID;
RUN;
```

SAS Code: Example Phase I Project 3 Table 1

```
/* Define Libraries */
libname XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;

/* Clean working directory */
/* proc datasets lib = work kill memtype = data nowarn; */
/* run; */

/************* make dataset *******************/

/***** First Chunk ****/
*Screened;

proc sql noprint;
        create table scrn as
                select count(distinct USUBJID) as count
                from xxxx;
quit;

*Screen Failure;

proc sql noprint;
        create table scrnf as
                select count(distinct USUBJID) as count
                from xxxx
                where ARMCD = 'SCRNFAIL';
quit;

*Other Screened but not Randomized;

proc sql noprint;
        create table othscrn as
                select count(distinct USUBJID) as count
                from adam.adsl
                where ARMCD = 'NOTASSGN';
quit;

*join;
data top;
        set scrn scrnf othscrn;
run;

*Format the table;
data first;
        length Summary $50;
        set scrn(in = a)
                scrnf(in = b)
                othscrn(in = c);

        if a then Summary = cats("Screened: ", count);
        else if b then Summary = cats("Screen Failure: ", count);
        else if c then Summary = cats("Other Screened but not Randomized: ", count);

        keep Summary;

run;


/* SUBSET DATA to only those randomized */
data treated;
        set adam.XXXX;
        if RANDFL = 'Y';
run;

/* For MSAB Report - Subset dataset for random numbers */

proc surveyselect data = treated out = treated
        method = srs
        samprate = 0.35;
run;

/**** chunk 2 dataset ****/

*Randomized;
proc freq data = treated noprint;
        tables ARM*RANDFL/out = mc1;
        where RANDFL = 'Y';
run;

proc transpose data = mc1 out = a(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data alab;
        attrib Summary length = $50;
        set a;
        Summary = "Randomized";
run;

*Treated;
proc freq data = treated noprint;
        tables ARM*SAFFL/out = mc2;
        where SAFFL = 'Y';
run;

proc transpose data = mc2 out = b(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data blab;
        attrib Summary length = $50;
        set b;
        Summary = "Treated";
run;

*Not Treated;
proc freq data = treated noprint;
        tables ARM*SAFFL/out = mc3 ;
        where SAFFL = 'N';
run;

proc transpose data = mc3 out = c(drop = _name_ _label_);
        id ARM;
        var COUNT;
```

```
run;

data clab;
        attrib Summary length = $50;
        set c;
        Summary = "Not Treated";
        if 'TRTB'n = . then 'TRTB'n  = 0;
run;
*Join;

data d;
        attrib Summary length = $50;
        set alab blab clab;
        Total = sum(of _NUMERIC_);
run;

data second;
        set first d;
run;

/**** Chunk 3 Dataset ****/

*Safety Population;
proc freq data = treated noprint;
        tables ARM*SAFFL/out = c31;
        where SAFFL = 'Y';
run;

proc transpose data = c31 out = e(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data elab;
        attrib Summary length = $50;
        set e;
        Summary = "Safety Population";
run;

*ITT Population;
proc freq data = treated noprint;
        tables ARM*ITTFL/out = c32;
        where ITTFL = 'Y';
run;

proc transpose data = c32 out = f(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data flab;
        attrib Summary length = $50;
        set f;
        Summary = "ITT Population";
run;
```

```
*per protocol population;
proc freq data = treated noprint;
        tables ARM*PPROTFL/out = c33;
        where PPROTFL = 'Y';
run;

proc transpose data = c33 out = g(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data glab;
        attrib Summary length = $50;
        set g;
        Summary = "Per-Protocol Population";
run;

*Join;

data h;
        attrib Summary length = $50;
        set elab flab glab;
        Total = sum(of _NUMERIC_);
run;

data third;
        set second h;
run;

/***** Chunk 4 Dataset *****/

*Merge treated with ADDS.xxxx & ADDS.xxxx;

*Completed Treatment Phase;
proc freq data = treated noprint;
        tables ARM/out = c41;
        where EOTDCRS = "COMPLETED" and SAFFL = 'Y';
run;

proc transpose data = c41 out = aa(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data aalab;
        attrib Summary length = $50;
        set aa;
        Summary = "Completed Treatment Phase";
run;

*Completed Treatment Phase & Completed Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM*ORLDCRSN/out = c42;
        where EOTDCRS = "COMPLETED" and
```

```
                ORLDCRSN = 1 and SAFFL = 'Y';
run;

proc transpose data = c42 out = ab(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data ablab;
        attrib Summary length = $50;
        set ab;
        Summary = "Completed Safety Follow-Up";
run;

*Completed Treatment Phase & Discontinued Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM*ORLDCRSN/out = c43;
        where EOTDCRS = "COMPLETED" and
                ORLDCRSN = 2 and SAFFL = 'Y';
run;

proc transpose data = c43 out = ac(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data aclab;
        attrib Summary length = $50;
        set ac;
        Summary = "Discontinued Safety Follow-Up";
run;

*Completed Treatment Phase & Did not enter Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM*ORLDCRSN/out = c44;
        where EOTDCRS = "COMPLETED" and
                ORLDCRSN = 3 and SAFFL = 'Y';
run;

proc transpose data = c44 out = ad(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;


data adlab;
        attrib Summary length = $50;
        set ad;
        Summary = "Did not enter Safety Follow-Up";
        if 'TRTC'n = . then 'TRTC'n  = 0;
        if 'TRTB'n = . then 'TRTB'n  = 0;
run;

*join;
```

```
data j;
        attrib Summary length = $50;
        set aalab ablab aclab adlab;
        Total = sum(of _NUMERIC_);
run;

data third;
        set third j;
run;

*Discontinued Treatment Phase;
proc freq data = treated noprint;
        tables ARM/out = c45;
        where EOTDCRS ne "COMPLETED" and SAFFL = 'Y';
run;

proc transpose data = c45 out = ae(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data aelab;
        attrib Summary length = $50;
        set ae;
        Summary = "Discontinued Treatment Phase";
run;

*Discontinued Treatment Phase & Completed Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM/out = c46;
        where EOTDCRS ne "COMPLETED" and SAFFL = 'Y' and ORLDCRSN = 1;
run;

proc transpose data = c46 out = af(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data aflab;
        attrib Summary length = $50;
        set af;
        Summary = "Completed Safety Follow-Up";
run;

*Discontinued Treatment Phase & Discontinued Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM/out = c47;
        where EOTDCRS ne "COMPLETED" and SAFFL = 'Y' and ORLDCRSN = 2;
run;

proc transpose data = c47 out = ag(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;
```

```
data aglab;
        attrib Summary length = $50;
        set ag;
        Summary = "Discontinued Safety Follow-Up";
run;

*Discontinued Treatment Phase & Did not enter Safety Follow-Up;
proc freq data = treated noprint;
        tables ARM/out = c48;
        where EOTDCRS ne "COMPLETED" and SAFFL = 'Y' and ORLDCRSN = 3;
run;

proc transpose data = c48 out = ah(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data ahlab;
        attrib Summary length = $50;
        set ah;
        Summary = "Did not enter Safety Follow-Up";
run;


*join;

data l;
        set aelab aflab aglab ahlab;
                Total = sum(of _NUMERIC_);
run;

data fourth;
        set third l;
run;

/***** Chunk 5 Dataset *****/

*Completed Study;
proc freq data = treated noprint;
        tables ARM*COMPLFL/out = c51;
        where COMPLFL = "Y";
run;

proc transpose data = c51 out = ba(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data balab;
        attrib Summary length = $50;
        set ba;
        Summary = "Completed Study";
run;

*Discontinued Study;
```

```
proc freq data = treated noprint;
        tables ARM*COMPLFL/out = c52;
        where COMPLFL = "N" and SAFFL = 'Y';
run;

proc transpose data = c52 out = bb(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data bblab;
        attrib Summary length = $50;
        set bb;
        Summary = "Discontinued Study";
run;

*Rollover to Study XXXXXXX;
proc freq data = treated noprint;
        tables ARM*RO1064FL/out = c53;
        where RO1064FL = "Y" and SAFFL = 'Y';
run;

proc transpose data = c53 out = bc(drop = _name_ _label_);
        id ARM;
        var COUNT;
run;

data bclab;
        attrib Summary length = $50;
        set bc;
        Summary = "Rollover to Study XXXX";
run;

*join;

data n;
        set balab bblab bclab;
                Total = sum(of _NUMERIC_);
run;

data table;
        set fourth n;
run;
```

```
/**** Add Percentages ****/

*per the footnote, percentages of first two blocks are based on 698 while
last two blocks are based on safety population (696). So create macro variables based
```

on that;

```
proc sql noprint;
   select count(distinct(USUBJID)) into: trt1tot trimmed from adam.adsl where ARM =
"" and RANDFL = 'Y' ;
   select count(distinct(USUBJID)) into: trt2tot trimmed from adam.adsl where ARM =
"TRTB" and RANDFL = 'Y';
   select count(distinct(USUBJID)) into: trt3tot trimmed from adam.adsl where ARM =
"TRTC" and RANDFL = 'Y' ;
      select (&trt1tot + &trt2tot + &trt3tot) into: total trimmed from adam.xxxx(obs
= 1);
   select count(distinct(USUBJID)) into: trt1saf trimmed from adam.adsl where ARM =
"TRTA" and SAFFL = 'Y';
   select count(distinct(USUBJID)) into: trt2saf trimmed from adam.adsl where ARM =
"TRTB" and SAFFL = 'Y';
   select count(distinct(USUBJID)) into: trt3saf trimmed from adam.adsl where ARM =
"TRTC" and SAFFL = 'Y' ;
                 select (&trt1saf + &trt2saf + &trt3saf) into: saftot trimmed from
adam.xxxx(obs = 1);

quit;

data table_pct;
        set table;

        /* Format: n (%) */
        length TRTA_fmt TRT1_fmt TRT2_fmt total_fmt $20;

        *in the first two blocks, use the total randomized population;
        if _N_ <= 9 then do;
                denom1 = &trt1tot;
                denom2 = &trt2tot;
                denom3 = &trt3tot;
                denom_tot = &total;
        end;
        *in the last 2 blocks, use the safety population;
        else if 10 <= _N_ <= 20 then do;
                denom1 = &trt1saf;
                denom2 = &trt2saf;
                denom3 = &trt3saf;
                denom_tot = &saftot;
        end;

        *calculate percentages;

        if 'TRTA'n = 0 then TRTA_fmt = "0";
                else if 'TRTA'n = . then TRTA_fmt = "";
                else TRTA_fmt = catt(put('TRTA'n, 3.), " (", put(100 * 'TRTA'n /
denom1, 5.1), ")");

        if 'TRTB'n = 0 then TRT1_fmt = "0";
                else if 'TRTB'n = . then TRT1_fmt = "";
                else TRT1_fmt = catt(put('TRTB'n, 3.), " (", put(100 * 'TRTB'n /
denom2, 5.1), ")");
```

```
        if 'TRTC'n = 0 then TRT2_fmt = "0";
                else if 'TRTC'n = . then TRT2_fmt = "";
                else TRT2_fmt = catt(put('TRTC'n, 3.), " (", put(100 * 'TRTC'n /
denom3, 5.1), ")");

        if 'Total'n = . then total_fmt = "";
                else total_fmt = catt(put('Total'n, 3.), " (", put(100 * 'Total'n /
denom_tot, 5.1), ")");

        drop 'TRTA'n 'TRTB'n 'TRTC'n 'Total'n denom1 denom2 denom3 denom_tot;
run;

/* add Number of subjects[1] row */

data table1;
        set table_pct;

        if _N_ = 10 then do;
                Summary = "Number of subjects[1]";
                TRTA_fmt = "";
                TRT1_fmt = "";
                TRT2_fmt = "";
                total_fmt = "";
                output;
        end;

        /* output the rest of the original table */
        set table_pct;
        output;
run;

/**** Add group and order markers ****/

data table2;
        set table1;

        length grp 8;
        length ord 8;

        *Add group markers;

        if 1 <= _N_ <= 3 then grp = 1;
        else if 4 <= _N_ <= 6 then grp = 2;
        else if 7 <= _N_ <= 9 then grp = 3;
        else if 10 <= _N_ <= 14 then grp = 4;
        else if 15 <= _N_ <= 18 then grp = 5;
        else if _N_ = 19 then grp = 6;
        else if _N_ = 20 then grp = 7;
        else if _N_ = 21 then grp = 8;

        *Add order markers;
        if 1 <= _N_ <= 10 then ord = 0;
        else if _N_ in (11, 15, 19, 20, 21) then ord = 1;

        else if (12 <= _N_ <= 14) or (16 <= _N_ <= 18) then ord = 2;
```

```
run;

/*** start of macro for applying spaces - needed to maintain the table format -
before applying
the compute line for this was adding spaces where they shouldnt be ***/


%macro shell (dsn, ordnum, grpnum);
data &dsn. ;
        ORD = &ordnum. ;
        GRP = &grpnum. ;
run;


%mend shell ;

%shell(dsn = shell1, ordnum = 0.1, grpnum = 0.1 ) ;
%shell(dsn = shell2, ordnum = 0.1, grpnum = 1.1 ) ;
%shell(dsn = shell3, ordnum = 0.1, grpnum = 2.1 ) ;
%shell(dsn = shell4, ordnum = 0.1, grpnum = 3.1 ) ;

%shell(dsn = shell6, ordnum = 1.1, grpnum = 5.1 ) ;
%shell(dsn = shell7, ordnum = 1.1, grpnum = 6.1 ) ;
%shell(dsn = shell8, ordnum = 1.1, grpnum = 7.1 ) ;
%shell(dsn = shell9, ordnum = 1.1, grpnum = 8.1 ) ;




data join;
        set shell1 shell2 shell3 shell4 shell6 shell7 shell8 shell9 table2;
run;

proc sort data = join;
        by GRP ORD;
run;

/*** end of macro ***/


/**** PROC REPORT ****/

ods listing close;
ods pdf file ="/home/dennes02/Project 3/Tables/dennes02_ADSL_s002_MSAB.pdf";
ods rtf file = "/home/dennes02/Project 3/Tables/dennes02_ADSL_s002_MSAB.rtf";

proc report data = join missing headline headskip split = "|" style(report) =
{just=left} spacing=1
        nowd style(header)=[font_face='Times New Roman' font_size=10pt]
        style(column)=[font_face='Times New Roman' font_size=10pt];
options orientation = landscape;

title1 font='Times New Roman' height=9pt bold j=left "Table XXXXXXX";
```

```
title2 font='Times New Roman' height=9pt bold j=left "Treatment Protocol XXXXXXXX";
title3 font='Times New Roman' height=9pt bold j=left "Subject Evaluation Groups";
footnote1 font='Times New Roman' height=9pt j=left "N is Number of Subjects
Randomized. Percentages are based on the number of subjects Randomized.";
footnote2 font='Times New Roman' height=9pt j=left "[1]Subjects in Safety Population.
Percentages are based on the number of subjects in Safety Population.";
footnote3 font='Times New Roman' height=9pt j=left "Treatment Period is planned for
16 weeks and Safety Follow-Up period is planned for 24 weeks.";
footnote4 font='Times New Roman' height=9pt j=left "Safety population consists of all
subjects treated with XX study medication. ITT population consists of all randomized
subjects who received at least one dose.";
footnote5 font='Times New Roman' height=9pt j=left "'Other Screened but not
Randomized' displays subjects who were screened but not randomized for a reason not
related to a specific eligibility criterion.";
footnote6 font='Times New Roman' height=9pt j=left "XXXXXXXXXXXXXXXXXX";
column Summary ("TRTA | (N=&trt1tot)"  (TRTA_fmt))
                                 ("TRTB | (N=&trt2tot)" (TRT1_fmt))
                                 ("TRT | (N=&trt3tot)" (TRT2_fmt))
                                 ("Total | (N=&total)" (total_fmt)) grp ord;

        define Summary/display " "
        style(column)=[width=2in];
        define TRTA_fmt/display "n  (%)"                  style(column)=[width=1in
just=c] style(header)=[just=c];
        define TRT1_fmt/display "n  (%)"                  style(column)=[width=1in
just=c] style(header)=[just=c];
        define TRT2_fmt/display "n  (%)"                  style(column)=[width=1in
just=c] style(header)=[just=c];
        define total_fmt/display "n  (%)"                 style(column)=[width=1in
just=c] style(header)=[just=c];
        define grp / order order = internal noprint;
        define ord / order order = internal noprint;

        compute Summary;
                if index(Summary, 'Treatment') or index(Summary, 'Study') then
                        call define('Summary', 'style', 'style = {leftmargin =
0.25in}');
                else if index(Summary, 'Follow-Up') then
                        call define('Summary', 'style', 'style = {leftmargin =
0.45in}');

        endcomp;

run;

ods pdf close;
ods rtf close;
ods listing;
```