BIOS 511 Lab 4
Basic Use of the DATA Step – Introduction to Merging Data Sets

Please read the following instructions carefully before beginning this lab. You will need to start by downloading the AE dataset from the LAB-04 assignment on the Sakai site. This dataset will be the basis of all activites for this lab. You will also need to use the DM dataset from earlier labs.

Instructions:

- All tasks should be completed in a single SAS program named lab-04-PID.sas where PID is your student PID number. Please make sure to include an appropriate header in your SAS program.
- All the output that your SAS program produces in this lab should be delivered to a *single HTML file* named lab-04-PID-output.HTML.
- In your code, before starting a new task, include a block comment with the task number

```
/*********************************************************************
                          SAS Code for Task # X
 *********************************************************************/
```

- Recommendation: While initially writing the SAS program, do not concern yourself with creating a permanent output file. Simply view results in the results window to verity that your program has created the desired output. Once your program is essentially complete, add the appropriate ODS statements to create the permanent HTML file based on the requirements above.
- You will upload the SAS program, SAS log, and HTML output file to document completion of the lab.

BIOS 511 Lab 4
Basic Use of the DATA Step – Introduction to Merging Data Sets

The AE data set contains ***adverse event*** data from the hypothetical ECHO clinical trial. According to the US Food and Drug Administration (FDA), an adverse event is defined as *any untoward medical occurrence associated with the use of a drug/device in humans, whether or not the event is considered drug/device related in the opinion of the investigator*.

All such events arising during a clinical trial that is under the purview of the FDA must be reported to the FDA to facilitate the FDA's evaluation of the risk-benefit profile for the drug or device.

The AE dataset contains all adverse event data from the hypothetical ECHO clinical trial. The variables contained in the AE dataset along with their types, lengths, and labels are included in the table below. A description of the values stored in the variable is also provided.

Variables contained in the AE dataset

#	Variable	Type	Len	Label	Description of Values
1	STUDYID	Char	10	Study Identifier	Values are always equal to "ECHO"
2	USUBJID	Char	30	Unique Subject Identifier	Values are of the form "ECHO-XXX-YYY" where XXX and YYY are integers
3	AETERM	Char	200	Reported Term for the Adverse Event	Investigator reported description of the adverse event – values are not standardized
4	AESOC	Char	200	Primary System Organ Class	Standardized MedDRA system organ class associated with value of AEDECOD
5	AEDECOD	Char	200	Dictionary-Derived Term	Standardized MedDRA dictionary term associated with value of AETERM
6	AESEV	Char	20	Severity/Intensity	Severity of adverse event – values are "MILD", "MODERATE", or "SEVERE"
7	AESER	Char	1	Serious Event	Indicator variable for whether the adverse event is serious – values are "Y" or "N"
8	AEOUT	Char	50	Outcome of Adverse Event	Values are standardized to one of four possible values – possible values are "RECOVERED/RESOLVED", "RECOVERED/RESOLVED WITH SEQUELAE", "RECOVERING/RESOLVING", and "NOT RECOVERED/NOT RESOLVED"
9	AESTDTC	Char	20	Start Date/Time of Adverse Event	Onset date of adverse event in YYYY-MM-DD format
10	AEENDTC	Char	20	End Date/Time of Adverse Event	Resolution date of adverse event in YYYY-MM-DD format – missing if adverse event was ongoing at the end of the trial

BIOS 511 Lab 4
Basic Use of the DATA Step – Introduction to Merging Data Sets

The structure of the AE dataset is one observation per subject per reported adverse event. The variables USUBJID, AETERM (or AEDECOD), and AESTDTC uniquely define observations in this dataset.

For your information, MedDRA is a standard dictionary of medical terms. You can read more about the MedDRA dictionary here https://en.wikipedia.org/wiki/MedDRA, but it is not necessary to do so.

The AE dataset has one observation *per subject per adverse event*. Thus, some subjects enrolled in the trial may not have any observations in the AE dataset and some subjects may have multiple observations in the AE dataset.

All subjects in the trial will have an observation in the DM dataset. This is a data standards requirement.

BIOS 511 Lab 4
Basic Use of the DATA Step – Introduction to Merging Data Sets

Task 1: Write a PROC CONTENTS step to print out metadata for the AE dataset.

- Use an ODS SELECT statement to select only the output object that contains the variable names, labels, types, and lengths.

- Use the VARNUM option on the PROC statement so that the variable names are ordered based on the column order. The column order of variables in a SAS dataset is the order of the column presentation when the dataset is opened for viewing (or the order in which the variables are printed when one does not use a VAR statement). By default, the ODS object produced by PROC CONTENTS lists the variables in alphabetical order based on their name.

- Hint: Using the VARNUM option changes the name of the ODS object that is created and so you will need to select a different ODS object that you have in the past! A key point here is that using certain options can change the set of OBS objects produced by a SAS procedure.

Task 2: Sort and print out select observations from the DM and AE datasets.

- Part 1:
  - Write a PROC SORT step that sorts the ECHO trial DM dataset by the unique subject ID variable and creates a new dataset named WORK.DM_SORTED.

  - **Include in output file:** Write a PROC PRINT step that prints a data listing containing all variables in the WORK.DM_SORTED dataset but only prints the first 5 observations.
- Part 2:
  - Write a PROC SORT step that sorts the ECHO trial AE dataset by the unique subject ID variable and creates a new dataset named WORK.AE_SORTED.

  - **Include in output file:** Write a PROC PRINT step that prints a data listing containing all variables in the WORK.AE_SORTED dataset but only prints the first 5 observations.

It should be apparent from the output that the subject with USUBJID = ECHO-011-001 did not experience any adverse events and that the subject with USUBJID = ECHO-011-004 experienced multiple adverse events. Make sure this is clear to you before moving on.

Task 3: For this task, you will *merge* the DM data with the AE data. The goal of this task is to add the SITEID, ARMCD, SEX, COUNTRY, RFXSTDTC, and RFXENDTC variables to the AE dataset. Since these variables are in the DM dataset, we need to merge the two datasets together to add the needed variables to the AE dataset. For this task, you will print out various data listings to help you understand the code below.

- We will learn much more about how to merge SAS datasets soon (including better ways to complete this very task)! Merge the datasets using the following DATA step:

  ```
  data work.AE2;
   merge work.DM_sorted(keep=usubjid armcd sex
                             race country rfxstdtc rfxendtc)
         work.AE_sorted;
   by usubjid;

    if (aeterm > '');
    drop studyid;
  run;
  ```

  In this DATA step, observations from the WORK.DM_sorted and WORK.AE_sorted data sets are joined together based on having a common value of USUBJID.

  This type of merge provides sensible results because the WORK.DM_sorted dataset has one observation per value of USUBJID and thus the match is well-defined. This type of merge is called a *one-to-many* merge because the WORK.DM_sorted data set has one observation per value of USUBJID and the WORK.AE_sorted dataset has more than one (at least for some values of USUBJID).

- Once you have run the DATA step code above, review the notes in the SAS log. The SAS log should tell you the number of observations read from each dataset being merged and the number of observations in the newly created AE2 dataset.

- **Include in output file: Part 1 -** Write a PROC PRINT step to print the first 15 observations in the newly created WORK.AE2 dataset. Observe that all 15 observations correspond to those printed for the WORK.AE_sorted dataset and that the newly created dataset has additional variables (compared to the WORK.AE_sorted dataset). The newly added variables are on the left because the WORK.DM_sorted dataset was listed first on the MERGE statement.

  Now, copy the DATA step above (adding a second DATA step to you SAS program), comment out the subsetting IF, and replace WORK.AE2 with WORK.AEDM on the DATA statement so that this new DATA step creates a dataset named WORK.AEDM. The IF statement used in the DATA step is called a "subsetting IF statement" and it uses a Boolean expression to determine which observations should be processed and which should be discarded.

- Run the revised DATA step code and note how many observations are in the WORK.AEDM data set. There are more observations in the WORK.AEDM data set than were in the WORK.AE2 data set. Thus, the subsetting IF statement caused some of them to be removed.

- **Include in output file: Part 2 -** Write a PROC PRINT step to print the first 15 observations in the newly created WORK.AEDM dataset. Observe that the first observation corresponds to the subject that did not have any adverse events and so all the variables coming from the AE data are missing for that subject.

  In the DATA step that created the WORK.AE2 data set, such observations were discarded because the Boolean expression (AETERM > '') is FALSE for those observations. When a subsetting IF statement is used, only observations where the Boolean expression is TRUE are processed beyond the subsetting IF statement and subsequently written to the new dataset (in this case the WORK.AE2 dataset). By using the subsetting IF statement in the DATA step that creates the WORK.AE2 data set, we are effectively asking SAS to keep all the observations in the AE data, to merge on the needed variables from the DM, and to discard any observations in the DM data set that do not have a match (based on USUBJID) in the AE dataset.

- **Temporary modification to code:** Try changing the subsetting IF statement to a WHERE statement in the DATA step that creates the WORK.AE2 data set and rerun the code. Carefully read the ERROR message in your SAS log. This illustrates one virtue of the subsetting IF statement compared to a WHERE statement. That is, the subsetting IF statement can be used regarding of whether the variables used in the evaluated expression (e.g., AETERM) are in the input datasets (or dataset).

  In other words, you can create a variable in a DATA step and then use it in a subsetting IF statement to discard unwanted observations. Only variables that exist on the input data set can be used in a WHERE statement and, when merging data, those variables must be in *all* datasets being merged.

- Make sure you revert your code to use a subsetting IF statement once you understand why the WHERE statement does not work in this case.

Task 4: For this task, you will use the WORK.AE2 dataset as input and create a dataset named WORK.TEAE using a DATA step.

The main purpose of this DATA step is create a new variable name TEAEFN. This variable name is based on a common naming convention. The prefix TEAE stands for *treatment emergent* *adverse event*, the FN suffix indicates that the variable is a *flag* variable which means that is binary variable with yes/no interpretation (F=flag, N=numeric).

The variable TEAEFN should have a value of 1 if the adverse event is *treatment-emergent* and a value of zero if the adverse event is *non-treatment-emergent*. The term treatment-emergent means that the adverse event began while the subject was on treatment. In most studies, this definition is operationalized by comparing the subject's treatment start date (RFXSTDTC) to the adverse event onset date (AESTDTC) to ensure the onset date for the event is on or after the date treatment started. In other words, TEAEFN has a value of 1 if the adverse event onset date is on or after the date treatment started.

The derivation is often complicated by the fact that the onset date for the adverse event is sometimes not fully known. When the adverse event onset *day* is not fully known, it is common to assume the latest plausible day for this derivation to ensure that events are conservatively attributed to the treatment period, if plausible.

The following steps should guide you through the creation of the TEAEFN variable and several intermediate variables used to create it.

Step 1: Using the SCAN function, create character variables AESTYR, AESTMN, and AESTDY, each with the minimum necessary length to store the AE start year, month, and day, respectively. You will need to write assignment statements such as the following:

```
AESTDY = scan(AESTDTC,3,'-');
```

Note that the scan() function requires the first argument to be a character expression in quotes or a character variable. The adverse event onset data AESTDTC is a character variable. The SDTM (Study Data Tabulation Model) standard for storing date data is to do so as a character variable. This is so partial date information can be stored effectively. For example, if an event began in February of 2016 but it was the case that the day precise day was unknown, the value of AESTDTC would be "2016-02". In this case, since there is no third "word" for the AESTDTC variable (with a word being any text separated by a "-"), the value of AESTDY would be missing.

Step 2: Only the day is sometimes missing in this dataset and so we will write programming statements that only deal with a missing day. If the value of AESTDY is missing, *impute* its value to the 28th day of the month. This approach is not ideal but avoids the complication of dealing with the fact that month may have 28, 29 (leap year), 30, or 31 days. You will need to use a conditional assignment statement similar to what is given below:

```
if <write code to see if AESTDY is missing>
 then <assign it a value of 28>;
```

Step 3: Use the MDY function to create the numeric AESTDTI (numeric, imputed onset date for AE). The MDY function expects arguments that are numeric (either numeric values or numeric variables), but will *auto-convert* character arguments to numeric arguments as long as the auto-conversion is straightforward (e.g., “28” -> 28). You will need to write assignment statements such as the following:

```
AESTDTI = MDY(AESTMN,AESTDY,AESTYR);
```

Carefully read the messages in your SAS log when the DATA step containing this code is run. The SAS log will provide notes about auto-conversion like this anytime you use a function that expects arguments of a certain type that do not match the arguments provided. On occasion, auto-conversion can produce unexpected results so it is always best to avoid it if you can. Here, it works just fine.

Step 4: Since the adverse event onset date was sometimes missing, we had to impute the missing part before we could construct a numeric SAS date variable (i.e., AESTDTI). For this step we want to create a numeric treatment start date variable (TRTSTDTN) from the character version RFXSTDTC. Do this using the same approach as outlined in Steps 1-3.

Step 5: Write a conditional assignment statement to create the TEAEFN variable based on the AESTDTI and TRTSTDTN variables. You will need to write a conditional assignment statement like the following:

```
if <adverse event began on/after treatment start date> then TEAEFN = 1;
else TEAEFN = 0;
```

Step 6: Add the following FORMAT statement to control the displayed values for the two numeric data variables just created.

```
format AESTDTI TRTSTDTN date9.;
```

Step 7: Add a DROP statement to remove the variables AESTYR, AESTMN, AESTDY, TRTSTYR, TRTSTMN, and TRTSTDY from the TEAE dataset. These were temporary variables that are generally not kept.

Step 8: Add a LABEL statement that sets the label for the AESTDTI and TRTSTDTN variables to *Imputed AE Onset Date (Numeric)* and *Treatment Start Date (Numeric)*, respectively.

A template DATA step is provided below. You will need to replace the comments with the required code.

```
data work.TEAE;
```

```
 set work.AE2;

 ** Insert LENGTH statement for date component variables;
 ** Insert FORMAT statement for numeric date variables;
 ** Insert LABEL statement for AESTDTI and TRTSTDTN variables;

 ** Insert assignment statements for AESTYR, AESTMN, and AESTDY;

 ** Insert conditional assignment statement to modify missing AESTDY;

 ** Insert assignment statements for TRTSTYR, TRTSTMN, AND TRTSTDY;

 ** Insert assignment statements to create numeric AESTDTI and
    TRTSTDTN variables using auto-conversion of character arguments
    to numeric arguments for the MDY() function;

 ** Insert assignment statements to create the TEAEFN variable
    from the AESTDTI and TRTSTDTN variables;

 ** Insert a DROP statement to remove temporary date component variables;

run;
```

Task 5: Write a PROC PRINT step to print only observations for *treatment-emergent infections and infestations where the onset date for the adverse event was imputed*. The goal for this task is to inspect the accuracy of the derivation for TEAEFN as well as to practice PROC PRINT.

- Use PROC statement options to suppress default observation numbering and to ensure that labels are printed.

- Use a WHERE statement to ensure that only treatment-emergent infections and infestations are included and only those observations where imputation took place.

  - To select infections and infestations, use the AESOC variable. You will need to look up the precise value (i.e. if it is upper case, lower case, etc) or you could use a DATA step function (i.e., uppercase()) to ensure the case of the variable value is as expected.

  - To select observations where imputation took place, it is easy enough to check the length *of the values* of the AESTDTC variable. Refer to the online SAS documentation for information on the LENGTH function. Feel free to use another technique if you want.

  - Because there are three conditions that need to be TRUE for this WHERE statement (i.e., the adverse event is treatment-emergent, the adverse event class is “infections and infestations” and the adverse event onset date was imputed) this WHERE statement will need to use the AND operator and should resemble the following:

    WHERE (Boolean expression 1) AND (Boolean expression 2) AND (Boolean expression 3)

- Use a VAR statement to ensure that only the data for variables USUBJID, AETERM, AEDECOD, AESTDTC, AESTDTI, and TRTSTDTN.