Module 7 - Final Project

DAD215

March 3, 2019

Evan Leet

I. Introduction to the Software Interface

A. Use a libname to show the creation of a data library. This library name will be used at different times in the code to reference the location of the data sets.

CODE:

```
/*The folowing code will satisfy the Introduction to Software interface portion of the final project rubric.

3/3/19
Evan Leet

input: final project folder and files
output: library


*/

libname bikes 'C:\Users\evan.leet\Documents\bikesheets1';
/*Use Libname statement. Name new library. Point SAS towards proper folder.*/
RUN;
```

LOG:

```
46   /*The folowing code will satisfy the Introduction to Software interface portion of the final
46 ! project rubric.
47
48   3/3/19
49   Evan Leet
50
51   input: final project folder and files
52   output: library
53
54
55   */
56
57   libname bikes 'C:\Users\evan.leet\Documents\bikesheets1';
NOTE: Libref BIKES was successfully assigned as follows:
      Engine:        V9
      Physical Name: C:\Users\evan.leet\Documents\bikesheets1
58   /*Use Libname statement. Name new library. Point SAS towards proper folder.*/
59   RUN;
```

II. Working With the Data: In this section, you will utilize SAS methods and data-manipulation techniques to work with the provided data. At the end, you will submit your finalized code that demonstrates your ability to do the following:

A. Utilize an appropriate method for importing and converting the data from the provided CSV file to an Excel file.

CODE:

```
/*The folowing code will use PROC IMPORT to convert the excel files into SAS files and place them in the library

3/3/19
Evan Leet

input: final project folder and files
output: SAS files

*/

PROC IMPORT out=bikes.bikes72013 datafile='C:\Users\evan.leet\Documents\bikesheets1\bikes72013.csv'
/*Use proc import to import data files. Use OUT to name the file and place it in the proper livrary
Point SAS towards the data file on the computer
*/
    DBMS=csv replace; /*original file is a CSV file*/
    getnames= yes;
RUN;
/*Run and repeat for the other files*/

PROC IMPORT out=bikes.bikes72014 datafile='C:\Users\evan.leet\Documents\bikesheets1\bikes72014.csv'
    DBMS=csv replace;
    getnames= yes;
RUN;

PROC IMPORT out=bikes.bikes72015 datafile='C:\Users\evan.leet\Documents\bikesheets1\bikes72015.csv'
    DBMS=csv replace;
    getnames= yes;
RUN;
```

LOG:

B. Create two new variables that represent the month and year of each file.

CODE:

```
/*The folowing code will use the same format for the date for each data set. It will also add a Month and Year variable.

3/3/19
Evan Leet

input: final project folder and files, SAS files created earlier
output: data sets with formatted dates

*/

DATA bikes.Bikes72013FD; /*New data set name. FD stands for formatted date*/
    set bikes.bikes72013; /*Point to original data set*/

    dateformatted = input(starttime, ANYDTDTM21.); /*Create new variable for the formatted date
                                                    Set the input as starttime variable, use ANYDTDTM21. to allow multiple date formats as input*/

    dateformatted = datepart(dateformatted); /*use datepart to keep only the date, use date9 format*/
        format dateformatted date9.;

    Month = Month(dateformatted); /*extract month from dateformatted, assign to new variable*/
    Year = Year(dateformatted);     /*exrtract year*/

RUN;
/*Run and repeat for other data sets*/

DATA bikes.Bikes72014FD;
    set bikes.bikes72014;


    dateformatted = input(starttime, ANYDTDTM21.);

    dateformatted = datepart(dateformatted);
        format dateformatted date9.;

    Month = Month(dateformatted);
    Year = Year(dateformatted);

RUN;


DATA bikes.Bikes72015FD;
    set bikes.bikes72015;

    dateformatted = input(starttime, ANYDTDTM21.);

    dateformatted = datepart(dateformatted);
        format dateformatted date9.;

    Month = Month(dateformatted);
    Year = Year(dateformatted);

RUN;

/*verify with proc print*/

PROC PRINT data=bikes.bikes72013fd (obs=5);
RUN;
PROC PRINT data=bikes.bikes72014fd (obs=5);
RUN;
PROC PRINT data=bikes.bikes72015fd (obs=5);
RUN;
```

LOG:

```
1524  /*The following code will use the same format for the date for each data set. It will also add a
1524! Date and Year.
1525
1526  3/3/19
1527  Evan Leet
1528
1529  input: final project folder and files
1530  output: SAS files
1531
1532  */
1533

NOTE: There were 5 observations read from the data set BIKES.BIKES72015FD.
NOTE: PROCEDURE PRINT used (Total process time):
      real time             57.98 seconds
      cpu time              3.82 seconds


1534  DATA bikes.Bikes72013FD; /*New data set name. FD stands for formatted date*/
1535      set bikes.bikes72013; /*Point to original data set*/
1536
1537      dateformatted = input(starttime, ANYDTDTME21.); /*Create new variable for the formatted date
1538                                                        Set the input as starttime variable, use
1538! ANYDTDTME21. to allow multiple date formats as input*/
1539
1540      dateformatted = datepart(dateformatted); /*use datepart to keep only the date, use date9
1540! format*/
1541          format dateformatted date9.;
1542
1543      Month = Month(dateformatted); /*extract month from dateformatted, assign to new variable*/
1544      Year = Year(dateformatted);     /*exrtract year*/
1545
1546  RUN;

NOTE: There were 843416 observations read from the data set BIKES.BIKES72013.
NOTE: The data set BIKES.BIKES72013FD has 843416 observations and 18 variables.
NOTE: DATA statement used (Total process time):
      real time             1.71 seconds
      cpu time              1.15 seconds


1547  /*Run and repeat for other data sets*/
1548
1549  DATA bikes.Bikes72014FD;
1550      set bikes.bikes72014;
1551
1552      dateformatted = input(starttime, ANYDTDTME21.);
1553
1554      dateformatted = datepart(dateformatted);
1555          format dateformatted date9.;
1556
1557       Month = Month(dateformatted);
1558       Year = Year(dateformatted);
1559
1560  RUN;

NOTE: There were 968842 observations read from the data set BIKES.BIKES72014.
NOTE: The data set BIKES.BIKES72014FD has 968842 observations and 18 variables.
NOTE: DATA statement used (Total process time):
      real time             2.13 seconds
      cpu time              1.34 seconds


1561
1562
1563  DATA bikes.Bikes72015FD;
1564      set bikes.bikes72015;
1565
1566      dateformatted = input(starttime, ANYDTDTME21.);
1567
1568      dateformatted = datepart(dateformatted);
1569          format dateformatted date9.;
1570
1571      Month = Month(dateformatted);
1572      Year = Year(dateformatted);
1573
1574  RUN;

NOTE: There were 1085676 observations read from the data set BIKES.BIKES72015.
NOTE: The data set BIKES.BIKES72015FD has 1085676 observations and 18 variables.
NOTE: DATA statement used (Total process time):
      real time             2.31 seconds
      cpu time              1.68 seconds


1575
1576  /*verify with proc print*/
1577
1578  PROC PRINT data=bikes.bikes72013fd (obs=5);
1579  RUN;

NOTE: There were 5 observations read from the data set BIKES.BIKES72013FD.
NOTE: PROCEDURE PRINT used (Total process time):
      real time             0.03 seconds


1580  PROC PRINT data=bikes.bikes72014fd (obs=5);
1581  RUN;

NOTE: There were 5 observations read from the data set BIKES.BIKES72014FD.
NOTE: PROCEDURE PRINT used (Total process time):
      real time             0.02 seconds
      cpu time              0.01 seconds


1582  PROC PRINT data=bikes.bikes72015fd (obs=5);
1583  RUN;

NOTE: There were 5 observations read from the data set BIKES.BIKES72015FD.
NOTE: PROCEDURE PRINT used (Total process time):
      real time             0.01 seconds
      cpu time              0.01 seconds
```

OUTPUT:

ır	gender	dateformatted	Month	Year
	0	01JUL2013	7	2013
	0	01JUL2013	7	2013
	2	01JUL2013	7	2013
	0	01JUL2013	7	2013
	1	01JUL2013	7	2013

ır	gender	dateformatted	Month	Year
	2	01JUL2014	7	2014
	1	01JUL2014	7	2014
	2	01JUL2014	7	2014
	1	01JUL2014	7	2014
	2	01JUL2014	7	2014

ır	gender	dateformatted	Month	Year
	1	01JUL2015	7	2015
	1	01JUL2015	7	2015
	1	01JUL2015	7	2015
	1	01JUL2015	7	2015
	1	01JUL2015	7	2015

C. Before merging your data, ensure that you have sorted your data appropriately. For example, both data sets must be sorted by the same ID variable.

CODE:

```
/*The folowing code will sort data sets by BIKEID

3/3/19
Evan Leet

input: final project folder and files, SAS files created earlier
output: sorted data sets

*/

PROC SORT data=bikes.bikes72013fd; /*Use PROC SORT. Point SAS to the relevenat data file*/
    BY bikeid; /*Sort by bikeid*/
RUN;
/*Run and repeat for other datasets*/

PROC SORT data=bikes.bikes72014fd;
    BY bikeid;
RUN;

PROC SORT data=bikes.bikes72015fd;
    BY bikeid;
RUN;
```

LOG:

```
1628  /*The folowing code will sort data sets by BIKEID
1629
1630  3/3/19
1631  Evan Leet
1632
1633  input: final project folder and files, SAS files created earlier
1634  output: sorted data sets
1635
1636  */
1637
1638  PROC SORT data=bikes.bikes72013fd; /*Use PROC SORT. Point SAS to the relevenat data file*/
1639      BY bikeid; /*Sort by bikeid*/
1640  RUN;

NOTE: Input data set is already sorted, no sorting done.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.01 seconds


1641  /*Run and repeat for other datasets*/
1642
1643  PROC SORT data=bikes.bikes72014fd;
1644      BY bikeid;
1645  RUN;

NOTE: Input data set is already sorted, no sorting done.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds


1646
1647  PROC SORT data=bikes.bikes72015fd;
1648      BY bikeid;
1649  RUN;

NOTE: Input data set is already sorted, no sorting done.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.01 seconds
```

D. Merge the data from the data sets.

CODE:

```
/*The folowing code will merge the previous data sets

3/3/19
Evan Leet

input: final project folder and files, SAS files created earlier
output: one merged set
*/

DATA bikes.bikescomplete; /*create and name new dataset*/
    merge bikes.bikes72013fd bikes.bikes72014fd bikes.bikes72015fd; /*use merge, call on relevant datafiles*/
    BY bikeid; /*merge by bikeid since sets were sorted by this variable earlier*/
RUN;
```

LOG:

```
1650  /*The following code will merge the previous data sets
1651
1652  3/3/19
1653  Evan Leet
1654
1655  input: final project folder and files, SAS files created earlier
1656  output: one merged set
1657  */
1658
1659  DATA bikes.bikescomplete; /*create and name new dataset*/
1660       merge bikes.bikes72013fd bikes.bikes72014fd bikes.bikes72015fd; /*use merge, call on
1660! relevant datafiles*/
1661       BY bikeid; /*merge by bikeid since sets were sorted by this variable earlier*/
1662  RUN;

WARNING: Multiple lengths were specified for the variable start_station_name by input data set(s).
         This can cause truncation of data.
WARNING: Multiple lengths were specified for the variable end_station_id by input data set(s). This
         can cause truncation of data.
WARNING: Multiple lengths were specified for the variable end_station_name by input data set(s). This
         can cause truncation of data.
WARNING: Multiple lengths were specified for the variable end_station_latitude by input data set(s).
         This can cause truncation of data.
WARNING: Multiple lengths were specified for the variable end_station_longitude by input data set(s).
         This can cause truncation of data.
NOTE: MERGE statement has more than one data set with repeats of BY values.
NOTE: There were 843416 observations read from the data set BIKES.BIKES72013FD.
NOTE: There were 96884? observations read from the data set BIKES.BIKES72014FD.
NOTE: There were 1085676 observations read from the data set BIKES.BIKES72015FD.
NOTE: The data set BIKES.BIKESCOMPLETE has 1554715 observations and 18 variables.
NOTE: DATA statement used (Total process time):
      real time           3.90 seconds
      cpu time            1.35 seconds
```

E. Review your work to this point and ensure that you identify and resolve any syntax or logic errors that you have encountered. Provide at least one example of an issue you debugged or justify why no changes were needed.

The work up to this point was making a usable data set that we can manipulate and ultimately use for the purposes set out in the assignment. I had trouble converting the given excel files into SAS data sets. At first, I was treating them as xlsx files but I then realized that they were CSV files. Once I made this important distinction, incorporating the files into SAS went more smoothly. I also had a hard time converting the dates into a single format. In the 3 original files, the dates were formatted differently and included the time. I did some research via google and found a solution that easily allowed me to convert all the dates to date9 format, and then extract the month and year from that.

F. Within your data, manipulate the text so that "Gender" is represented by numbers. (Male = 1; Female = 2; Unknown = 0).

Already Complete.

G. After formatting the data, utilize a function to merge two variables together. (Station A is one variable, and Station B is one variable) = Station A – Station B (new variable).

CODE:

```
/*The folowing create a new variable which is the combination of two existing variables

3/3/19
Evan Leet

input: bikescomplete set
output: bikes complete set with an additional variable
*/

DATA bikes.bikescomplete;
    set bikes.bikescomplete;

    monthyear = catx('/', month, year); /*create new variable, use catx, determine seperator and existing variables to use*/

RUN;
```

LOG:

```
1732  /*The folowing create a new variable which is the combination of two existing variables
1733
1734  3/3/19
1735  Evan Leet
1736
1737  input: bikescomplete set
1738  output: bikes complete set with an additional variable
1739  */
1740
1741  DATA bikes.bikescomplete;
1742      set bikes.bikescomplete;
1743
1744      monthyear = catx('/', month, year); /*create new variable, use catx, determine seperator and
1744!   existing variables to use*/
1745
1746  RUN;

NOTE: There were 1554715 observations read from the data set BIKES.BIKESCOMPLETE.
NOTE: The data set BIKES.BIKESCOMPLETE has 1554715 observations and 19 variables.
NOTE: DATA statement used (Total process time):
      real time           3.30 seconds
      cpu time            1.59 seconds
```

H. Utilize conditional logic to create a new variable based on trip duration. For example, if duration is less than 5 minutes, then trip_time = “short trip.”

CODE:

```
/*The folowing code will use if/then to define a variable based on given conditions
if trip duration is greater than 5 minutes, call it long. Otherwise, call it short

3/3/19
Evan Leet

input: bikescomplete set
output: bikescomplete set with an additional variable
*/

DATA bikes.bikescomplete;
    set bikes.bikescomplete;

    tripdur = 'short'; /*define new variable*/

    if tripduration > 300 then tripdur = 'long'; /*use if then to assign new value to variable based on given condition*/

RUN;
```

LOG:

```
NOTE: DATA statement used (Total process time):
      real time           5.86 seconds
      cpu time            1.45 seconds


1781  /*The folowing code will use if/then to define a variable based on given conditions
1782  if trip duration is greater than 5 minutes, call it long. Otherwise, call it short
1783
1784  3/3/19
1785  Evan Leet
1786
1787  input: bikescomplete set
1788  output: bikescomplete set with an additional variable
1789  */
1790
1791  DATA bikes.bikescomplete;
1792      set bikes.bikescomplete;
1793
1794      tripdur = 'short'; /*define new variable*/
1795
1796      if tripduration > 300 then tripdur = 'long'; /*use if then to assign new value to variable
1796! based on given condition*/
1797
1798  RUN;

NOTE: Character values have been converted to numeric values at the places given by: (Line):(Column).
      1796:8
NOTE: There were 1554715 observations read from the data set BIKES.BIKESCOMPLETE.
NOTE: The data set BIKES.BIKESCOMPLETE has 1554715 observations and 20 variables.
NOTE: DATA statement used (Total process time):
      real time           7.11 seconds
      cpu time            1.54 seconds
```

I. From this data set, you will create multiple data sets, applying different criteria to each. You will need to create one data set for each breakout of the city (boroughs) represented in the data.

```
/*The folowing code will combine the bikeboroughs data set with the bikescomplete dataset,
it will give each observation a borough.
Afterward, it will remove observations that are missing a stationid
It will is also create a variable ageatuse which is the difference between birthyear and year of use

3/3/19
Evan Leet

input: bikescomplete set and nyc bikeboroughs set
output: bikescomplete set with borough added, other items as above
*/

PROC SORT data=bikes.nycbikeboroughs;
    by station_id;

RUN;

DATA bikes.bikescomplete;
    set bikes.bikescomplete;
    station_id = input(start_station_id, 5.);
RUN;

PROC SORT data=bikes.bikescomplete;
    by station_id;
RUN;

DATA bikes.bikescomplete;
    merge bikes.bikescomplete bikes.nycbikeboroughs;
    by station_id;

RUN;

DATA bikes.bikescomplete;
    set bikes.bikescomplete(where=(station_id ne .));
RUN;

DATA bikes.bikescomplete;
    set bikes.bikescomplete;

    ageatuse = year - birth_year;

    RUN;
```

J. Review your work to this point and ensure that you identify and resolve any syntax or logic errors that you have encountered. Provide at least one example of an issue you debugged or justify why no changes were needed.

The code just above is representative of an issue encountered. It was sort of tricky to get the name of the borough alongside the other data. Luckily, we were provided the chart that provided the station ID and the borough each is located in on a separate sheet. I sorted the borough sheet

and the bike data set by station ID and merged them. This gave each observation in the bike data set a value for location. I also had to calculate the customer's age at use based on their birth year and the year they used the Citi Bike service. Luckily, the Year variable was calculated prior to merging.

K. Submit your finalized code that runs successfully without errors and yields appropriate outcomes. Submit your code and log files by copying and pasting into a Microsoft Word document from SAS.

See above and below

CODE:

```
/*The folowing code will further trim the bikescomplete set to include only the necessary data
  | 3/3/19
Evan Leet
input: bikescomplete set and nyc bikeboroughs set
output: bikescomplete set with borough added, other items as above
*/

DATA bikes.bikescompleteref;/*create and name new data set*/
    set bikes.bikescomplete (keep=tripdur bikeid  usertype birth_year gender
                                  station_id location ageatuse dateformatted year);
                            /*Use keep to keep only the required data*/


    if location = "Brooklyn" or location = "Manhattan" then output;
    /*output if location is brooklyn or manhattan only*/

RUN;
```

LOG:

```
2195
2196    /*The folowing code will further trim the bikescomplete set to include only the necessary data
2197   3/3/19
2198    Evan Leet
2199
2200    input: bikescomplete set and nyc bikeboroughs set
2201    output: bikescomplete set with borough added, other items as above
2202    */
2203
2204   DATA bikes.bikescompleteref;/*create and name new data set*/
2205       set bikes.bikescomplete (keep=tripdur bikeid  usertype birth_year gender
2206                                     station_id location ageatuse dateformatted year);
2207                               /*Use keep to keep only the required data*/
2208
2209
2210       if location = "Brooklyn" or location = "Manhattan" then output;
2211       /*output if location is brooklyn or manhattan only*/
2212
2213   RUN;

NOTE: There were 1554872 observations read from the data set BIKES.BIKESCOMPLETE.
NOTE: The data set BIKES.BIKESCOMPLETEREF has 1551326 observations and 10 variables.
NOTE: DATA statement used (Total process time):
      real time           1.19 seconds
      cpu time            1.15 seconds
```

The CONTENTS Procedure

Data Set Name	BIKES.BIKESCOMPLETEREF	Observations	1551326
Member Type	DATA	Variables	10
Engine	V9	Indexes	0
Created	03/03/2019 20:51:26	Observation Length	80
Last Modified	03/03/2019 20:51:26	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1899
First Data Page	1
Max Obs per Page	817
Obs in First Data Page	792
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\Users\evan.leet\Documents\bikesheets1\bikescompleteref.sas7bdat
Release Created	9.0401M3
Host Created	X64_8PRO

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat	Label
6	Year	Num	8			
10	ageatuse	Num	8			
1	bikeid	Char	7	$7.	$7.	
3	birth_year	Char	6	$6.	$6.	
5	dateformatted	Num	8	DATE9.		
4	gender	Char	3	$3.	$3.	
9	location	Char	9	$9.	$9.	location
8	station_id	Num	8	BEST.		station id
7	tripdur	Char	5			
2	usertype	Char	12	$12.	$12.	

CODE:

```
/*The folowing code will add age categories

 Evan Leet
 input: bikescompleteref
 output: bikescompleteref with agecat
 */

DATA bikes.bikescompleteref;
    set bikes.bikescompleteref;

    agecat = 'unknown'; /*define new variable*/

    /*Set up if then statements to change agecat as necessary*/
    if ageatuse ge 0 and ageatuse le 25 then agecat = '0-25';
    else if ageatuse ge 26 and ageatuse le 35 then agecat = '26-35';
    else if ageatuse ge 36 and ageatuse le 49 then agecat = '36-49';
    else if ageatuse ge 50 then agecat = '50+';
    else if ageatuse = . then agecat = 'unknown'; /*assign unknown if no value*/

    RUN;
```

LOG:

```
2240
2241    /*The folowing code will add age categories
2242
2243    Evan Leet
2244    input: bikescompleteref
2245    output: bikescompleteref with agecat
2246    */
2247
2248   DATA bikes.bikescompleteref;
2249       set bikes.bikescompleteref;
2250
2251       agecat = 'unknown';
2252
2253       if ageatuse ge 0 and ageatuse le 25 then agecat = '0-25';
2254       else if ageatuse ge 26 and ageatuse le 35 then agecat = '26-35';
2255       else if ageatuse ge 36 and ageatuse le 49 then agecat = '36-49';
2256       else if ageatuse ge 50 then agecat = '50+';
2257       else if ageatuse = . then agecat = 'unknown';
2258
2259       RUN;

NOTE: There were 1551326 observations read from the data set BIKES.BIKESCOMPLETEREF.
NOTE: The data set BIKES.BIKESCOMPLETEREF has 1551326 observations and 11 variables.
NOTE: DATA statement used (Total process time):
      real time           0.58 seconds
      cpu time            0.51 seconds
```

III. Understanding the Data In this section, you will write your data summary to explain your findings to stakeholders.

A. Explain how you imported the data into SAS, the id variables used to merge the data, the formats and functions used to transform the data, and the analysis done to summarize the data.

A library for the Citi Bike data was created in SAS. The excel files were converted into SAS files and were incorporated into this library. The date of use for each entry was created and the month and year of use were extracted from this date. Each of the 3 files was sorted by the same variable (bikeid) and then the three files were combined into one dataset. The age at use was calculated for each subscriber based on the year of use and the subcriber's year of birth – a new variable was created and each entry was the result of the calculation of the difference between age at use and birth year. Bike users were further placed in age groups. Unknown entries for age and gender come from non subscribers. Each entry included a station ID and this was used to merge the bike use data set with a key that included the borough in which each station ID is located. This merging process ensured that each entry would have the appropriate borough assigned to it. These three categories - age group, gender, and location -were the basis for the following analyses.

B. Use SAS graphing procedures to create graphs of your outcome data.

CODE:

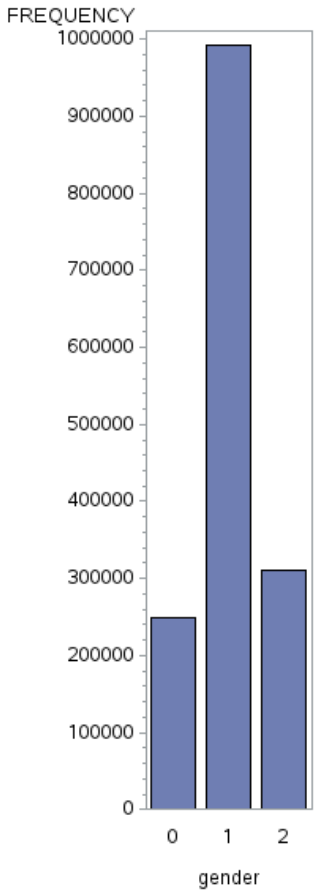
```
/*The folowing code will create various graphs

 Evan Leet
 input: bikescompleteref
 output: graphs
 */

 title "Total Frequency of Gender"; /*assign title*/

 PROC GCHART data=bikes.bikescompleteref; /*Use PROC GCHART, point SAS to the proper file*/
    vbar gender; /*Use vertical bar graph for the variable gaender*/

    RUN;

title "Total Frequency of Age by group";

 PROC GCHART data=bikes.bikescompleteref;
    vbar agecat; /*Use vertical bar graph for the variable agecat*/

    RUN;


     title "Frequency of Gender By Year and Location";

 PROC GCHART data=bikes.bikescompleteref;
    vbar gender /
    subgroup=location /*create a verticle chart of gender, subgrouped by location*/
    GROUP=year;         /*further group results by year*/

    RUN;

     title "Frequency of Age Group By Year and Location";

 PROC GCHART data=bikes.bikescompleteref;
    vbar agecat /
    subgroup=location
    GROUP=year;

    RUN;
```

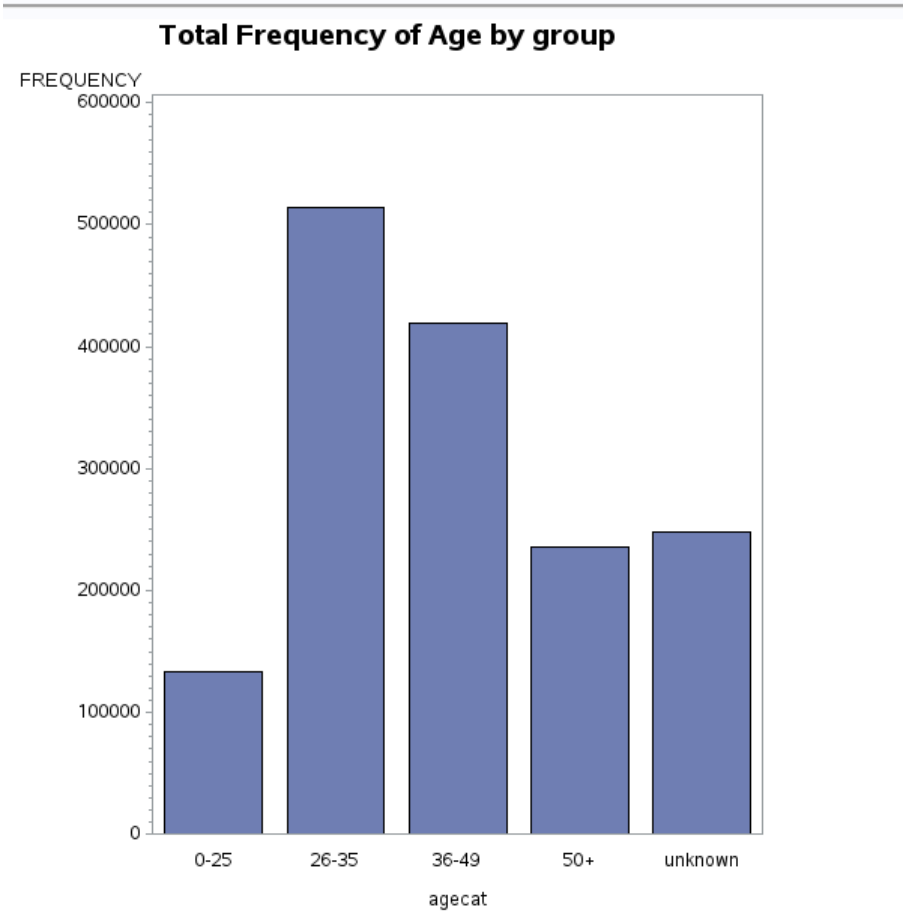
LOG:

```
2411    title "Total Frequency of Gender";
2412
NOTE: There were 1551191 observations read from the data set BIKES.BIKESCOMPLETEREF.
NOTE: PROCEDURE GCHART used (Total process time):
      real time           3:35.41
      cpu time            9.85 seconds


2413    PROC GCHART data=bikes.bikescompleteref;
2414       vbar gender;
2415
2416       RUN;

NOTE: 12773 bytes written to C:\Users\EVAN~1.LEE\AppData\Local\Temp\SAS Temporary
      Files\_TD10132_STEM-SAS199_\gchart10.png.
2417  title "Total Frequency of Age by group";
2418

NOTE: There were 1551191 observations read from the data set BIKES.BIKESCOMPLETEREF.
NOTE: PROCEDURE GCHART used (Total process time):
      real time           10.85 seconds
      cpu time            2.09 seconds


2419    PROC GCHART data=bikes.bikescompleteref;
2420       vbar agecat;
2421
2422       RUN;

NOTE: 13570 bytes written to C:\Users\EVAN~1.LEE\AppData\Local\Temp\SAS Temporary
      Files\_TD10132_STEM-SAS199_\gchart11.png.
2423        title "Frequency of Gender By Year and Location";
2424

NOTE: There were 1551191 observations read from the data set BIKES.BIKESCOMPLETEREF.
NOTE: PROCEDURE GCHART used (Total process time):
      real time           10.94 seconds
      cpu time            2.04 seconds


2425    PROC GCHART data=bikes.bikescompleteref;
2426       vbar gender /
2427       subgroup=location
2428       GROUP=year;
2429
2430       RUN;

NOTE: 16195 bytes written to C:\Users\EVAN~1.LEE\AppData\Local\Temp\SAS Temporary
      Files\_TD10132_STEM-SAS199_\gchart12.png.
2431        title "Frequency of Age Group By Year and Location";
2432

NOTE: There were 1551191 observations read from the data set BIKES.BIKESCOMPLETEREF.
NOTE: PROCEDURE GCHART used (Total process time):
      real time           11.09 seconds
      cpu time            3.17 seconds


2433    PROC GCHART data=bikes.bikescompleteref;
2434       vbar agecat /
2435       subgroup=location
2436       GROUP=year;
2437
2438       RUN;

NOTE: 17499 bytes written to C:\Users\EVAN~1.LEE\AppData\Local\Temp\SAS Temporary
      Files\_TD10132_STEM-SAS199_\gchart13.png.
```

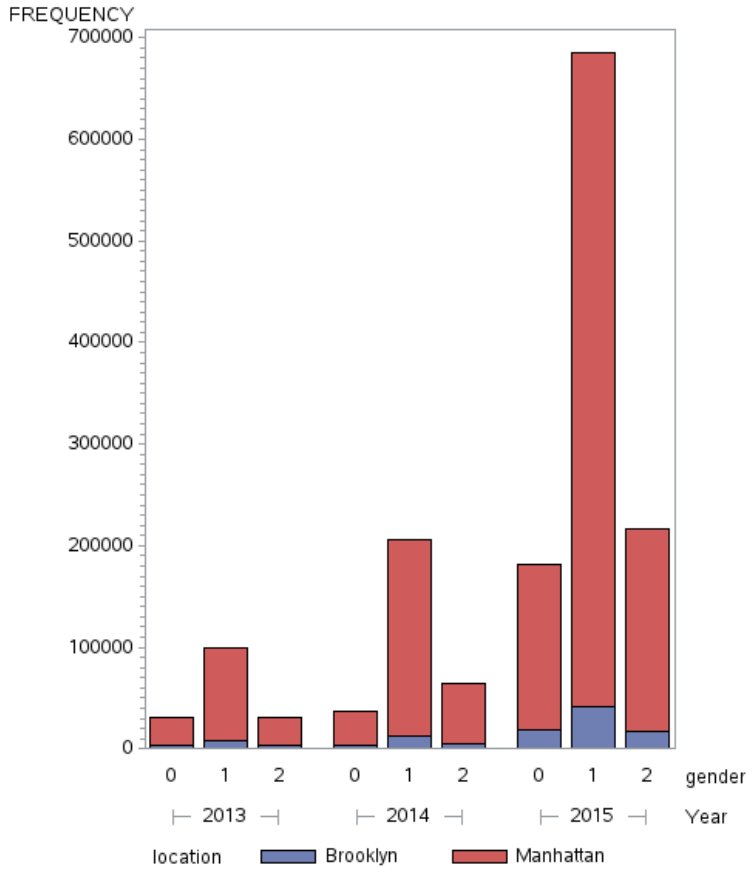
RESULTS:

Total Frequency of Gender

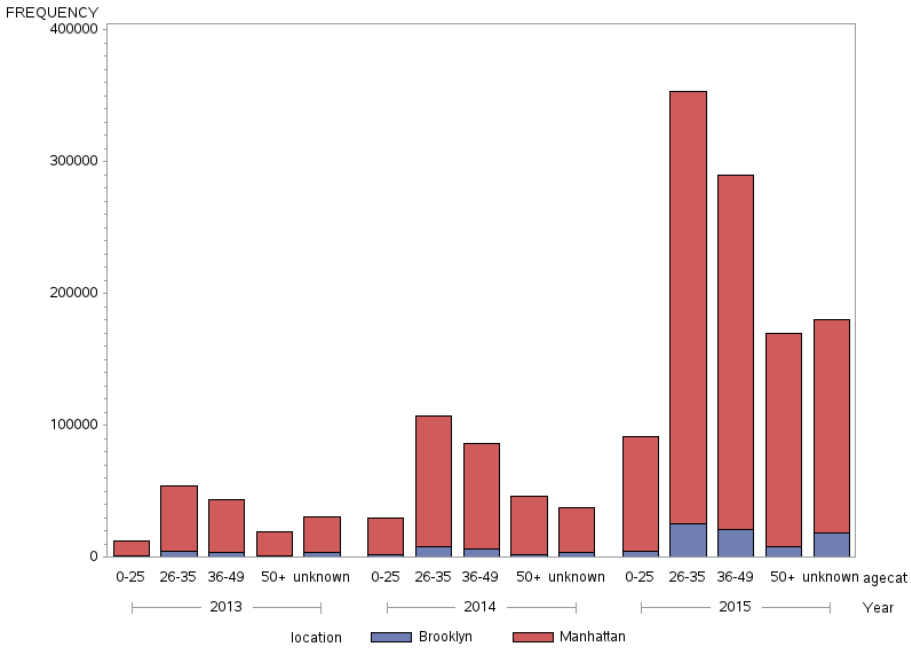


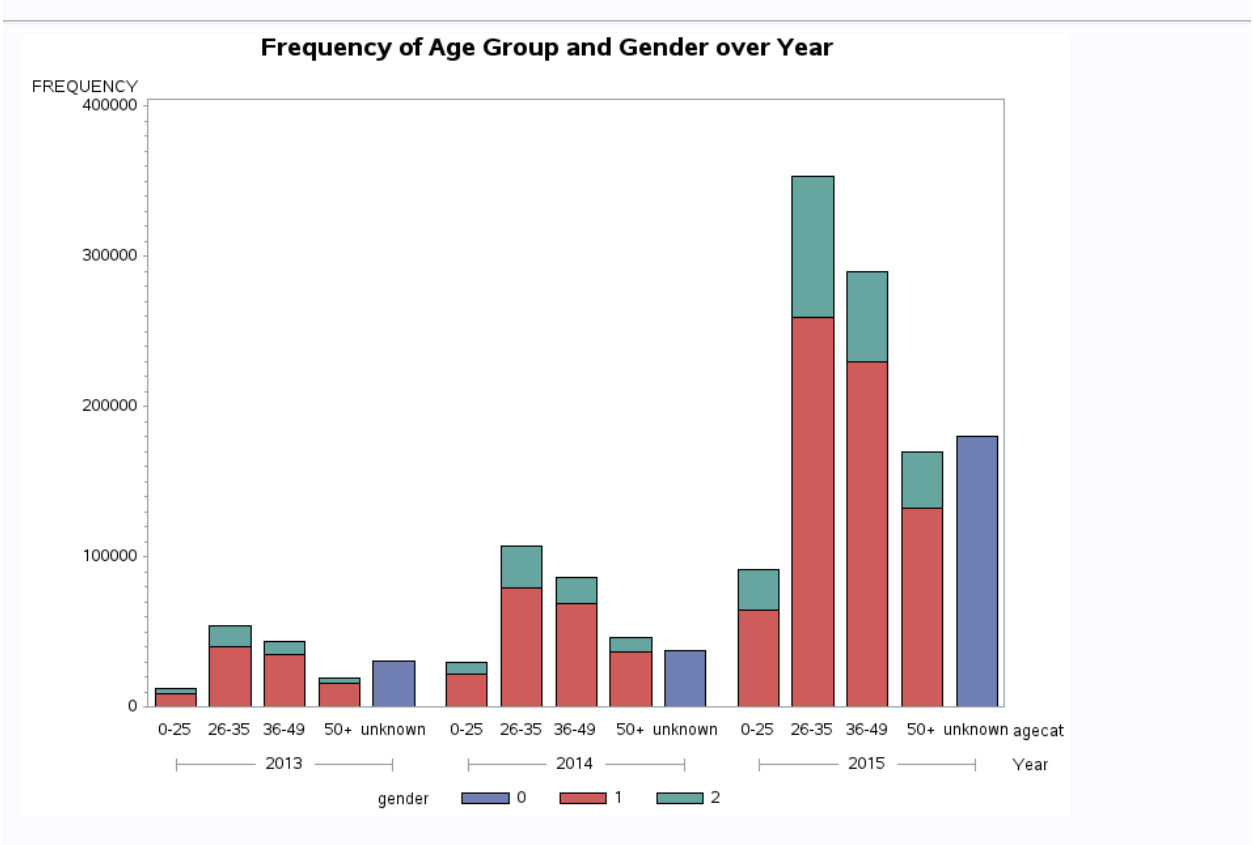

Total Frequency of Age by group
FREQUENCY
600000
500000
400000
300000
200000
100000
0
0-25
26-35
36-49
50+
unknown
agecat

Frequency of Gender By Year and Location

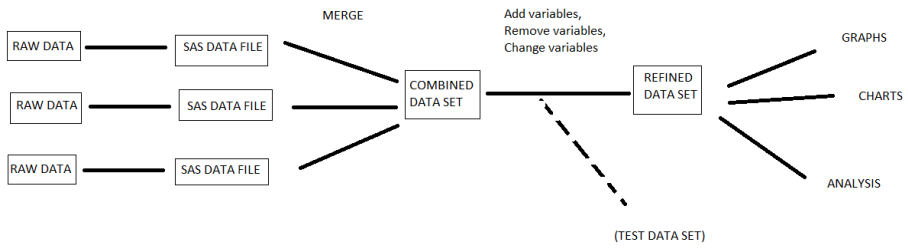


Frequency of Age Group By Year and Location





C. Create a visual illustration of the data workflow, from import to multiple data set creation.



D. Discuss the errors or difficulties encountered during the coding process, including how you resolved these issues.

The majority of my difficulties came from creating and manipulating the data sets. I frequently created duplicate data sets so that I could experiment with them and not worry about ruining something important. In terms of graphs, I tried to learn from my classmates and included a graph which I learned from the discussion assignment in previous weeks.

E. Explain what the data tells you about the utilization of the bike-sharing program. Support your response with examples from the data. This support could be presented in table form but should come from the descriptive analysis.

F. Summarize what you learned about the bike-sharing program and its users. Support your response with examples from the data. (Remember this should be the final summary of the outcomes discovered from the analysis and should be directed toward a professional audience.)

In looking at the graphs created by the data set, it looks like use of the Citi Bikes has been increasing year to year. This is true among genders and across the age categories. Usage in Manhattan is greater than usage in Brooklyn, and usage in Manhattan has also been increasing at a much greater rate. It seems that males are more commonly users than females, and the most frequent age group is the 25-36 group. Usage among males in the 25-36 year old age group is increasing, and usage among females in this category is increasing as well. In fact, usage among females in general appears to have tripled or more from 2014 to 2015.

In terms of location, Manhattan locations are clearly more frequently used. Growth in this location is also more apparent. In both locations and across all years, males are the most common gender group and 25-36 is the most common age group. Usage in Brooklyn is increasing over the years (21,496 users in 2014 to 77,754 users in 2015), but the numbers from Manhattan are far greater – 285,752 users in 2014 to 1,005,661 in 2015. The following charts further demonstrate these findings:

3 Way Chart of Gender and Location by Year

The FREQ Procedure

Table 1 of gender by location
Controlling for Year=2013

Frequency / Percent / Row Pct / Col Pct	location(location)		
gender	Brooklyn	Manhattan	Total
0	3473 2.16 11.36 25.53	27101 16.87 88.64 18.43	30574 19.03
1	7368 4.59 7.40 54.16	92168 57.38 92.60 62.69	99536 61.97
2	2762 1.72 9.05 20.30	27756 17.28 90.95 18.88	30518 19.00
Total	13603 8.47	147025 91.53	160628 100.00

Table 2 of gender by location
Controlling for Year=2014

Frequency / Percent / Row Pct / Col Pct	location(location)		
gender	Brooklyn	Manhattan	Total
0	4094 1.33 10.97 19.05	33227 10.81 89.03 11.63	37321 12.15
1	12498 4.07 6.06 58.14	193828 63.09 93.94 67.83	206326 67.15
2	4904 1.60 7.71 22.81	58697 19.10 92.29 20.54	63601 20.70
Total	21496 7.00	285752 93.00	307248 100.00

Table 3 of gender by location
Controlling for Year=2015

Frequency / Percent / Row Pct / Col Pct	location(location)		
gender	Brooklyn	Manhattan	Total
0	18669 1.72 10.31 24.01	162354 14.99 89.69 16.15	181023 16.71
1	42022 3.88 6.13 54.04	643760 59.43 93.87 64.02	685782 63.30
2	17063 1.58 7.88 21.94	199447 18.41 92.12 19.83	216510 19.99
Total	77754 7.18	1005661 92.82	1083315 100.00

3 Way Chart of Age Category and Location by Year

The FREQ Procedure

Frequency
Percent
Row Pct
Col Pct

Table 1 of agecat by location

Controlling for Year=2013

agecat	location(location) Brooklyn	Manhattan	Total
0-25	795 0.49 6.47 5.84	11489 7.15 93.53 7.81	12284 7.65
26-35	4705 2.93 8.69 34.59	49446 30.78 91.31 33.63	54151 33.71
36-49	3427 2.13 7.78 25.19	40614 25.28 92.22 27.62	44041 27.42
50+	1204 0.75 6.15 8.85	18378 11.44 93.85 12.50	19582 12.19
unknown	3472 2.16 11.36 25.52	27098 16.87 88.64 18.43	30570 19.03
Total	13603 8.47	147025 91.53	160628 100.00

Frequency
Percent
Row Pct
Col Pct

Table 2 of agecat by location

Controlling for Year=2014

agecat	location(location) Brooklyn	Manhattan	Total
0-25	1568 0.51 5.20 7.29	28608 9.31 94.80 10.01	30176 9.82
26-35	7796 2.54 7.26 36.27	99614 32.42 92.74 34.86	107410 34.96
36-49	5879 1.91 6.82 27.35	80270 26.13 93.18 28.09	86149 28.04
50+	2173 0.71 4.70 10.11	44059 14.34 95.30 15.42	46232 15.05
unknown	4080 1.33 10.94 18.98	33201 10.81 89.06 11.62	37281 12.13
Total	21496 7.00	285752 93.00	307248 100.00

Frequency
Percent
Row Pct
Col Pct

Table 3 of agecat by location

Controlling for Year=2015

agecat	location(location) Brooklyn	Manhattan	Total
0-25	4626 0.43 5.08 5.95	86472 7.98 94.92 8.60	91098 8.41
26-35	25527 2.36 7.23 32.83	327436 30.23 92.77 32.56	352963 32.58
36-49	21100 1.95 7.29 27.14	268418 24.78 92.71 26.69	289518 26.73
50+	7939 0.73 4.68 10.21	161817 14.94 95.32 16.09	169756 15.67
unknown	18562 1.71 10.31 23.87	161418 14.90 89.69 16.05	179980 16.61
Total	77754 7.18	1005561 92.82	1083315 100.00