Instructions: “*Geology rocks but geography is where it's at. . .*” (famous dad joke). In a global economy, geography has an important influence on everything from manufacturing to marketing to transportation. As a result, most data scientists will have to work with map data at some point in their careers. An add-on to the ggplot2 package, called ggmap, provides powerful tools for plotting and shading maps. Make sure to install the maps, mapproj, and ggmap packages before running the following:

**library(ggplot2); library(maps); library(ggmap); library(mapproj)**

**us <- map\_data("state")**

**us$state\_name <- tolower(us$region)**

**map <- ggplot(us, aes(map\_id= state\_name))**

**map <- map + aes(x=long, y=lat, group=group) +**

**geom\_polygon()**

**map <- map + expand\_limits(x=us$long, y=us$lat)**

**map <- map + coord\_map() + ggtitle("USA Map")**

**map**

1. Add a comment for each line of code, explaining what that line of code does.
2. The map you just created fills in the area of each county in black while outlining it with a thin white line. Use the fill= and color= commands inside the call to geom\_polygon( ) to reverse the color scheme

Now run the following code:

**ny\_counties <- map\_data("county","new york")**

**ggplot(ny\_counties) + aes(long,lat, group=group) + geom\_polygon()**

1. Just as in step 2, the map you just created fills in the area of each county in black while outlining it with a thin white line. Use the fill= and color= commands inside the call to geom\_polygon( ) to reverse the color scheme.
2. Run head(ny\_counties) to verify how the county outline data looks.
3. Make a copy of your code from step 3 and add the following subcommand to your ggplot( ) call (don’t forget to put a plus sign after the geom\_polygon( ) statement to tell R that you are continuing to build the command):  
   **coord\_map(projection = "mercator")**In what way is the map different from the previous map. Be prepared to explain what a Mercator projection is.

This is the end of the first breakout session. Please send a chat message to your instructor when you are finished.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Second Breakout Session Starts Here

1. Grab a copy of the nyData.csv data set from the same location where you obtained this handout. Read that data set into R with read\_csv(). The next step assumes that you have named the resulting data frame “nyData.”
2. Next, merge your ny\_counties data from the first breakout group with your new nyData data frame, with this code:  
   **mergeNY <- merge(ny\_counties,nyData,  
    all.x=TRUE,by.x="subregion",by.y="county")**
3. Run head(mergeNY) to verify how the merged data looks.
4. Now drive the fill color inside each county by adding the fill aesthetic *inside of* your geom\_polygon( )subcommand (fill based on the pop2000).
5. Extra (not required):
   1. Read in the following JSON datasets:  
      'https://gbfs.citibikenyc.com/gbfs/en/station\_information.json'  
      'https://gbfs.citibikenyc.com/gbfs/en/station\_status.json'
   2. Merge the datasets, based on ‘station\_id’
   3. Clean the merged dataset to only include useful information  
      For this work, you only need lat, lon and the number of bikes available
   4. Create a stamen map using ‘get\_stamenmap()’  
      Have the limits of the map be defined by the lat and lot of the stations
   5. Show the stations, as points on the map.
   6. Show the number of bikes available as a color