STA6714 Assignment 1

Kyle Scott

9/22/2021

QUESTION ONE: SQLite

1. Setting up the climate database

– Load in necessary libraries and read in the csv file to an R dataframe

```
library('RSQLite')
library('sqldf')

# CODE CAN BE EASILY REPLICATED BY PLACING THIS CSV IN IT'S
    # DESIRED LOCATION AND USING THE setwd() FUNCTION TO SET THE
    # WORKING DIR TO THAT LOCATION

climate_df <- read.csv(file = 'DailyDelhiClimateTest.csv',
                       header = TRUE)
```

– Create an SQL database using RSQLite

```
db.climate <- dbConnect(SQLite(), dbname = 'climate')
```

– Place the climate dataframe into the database

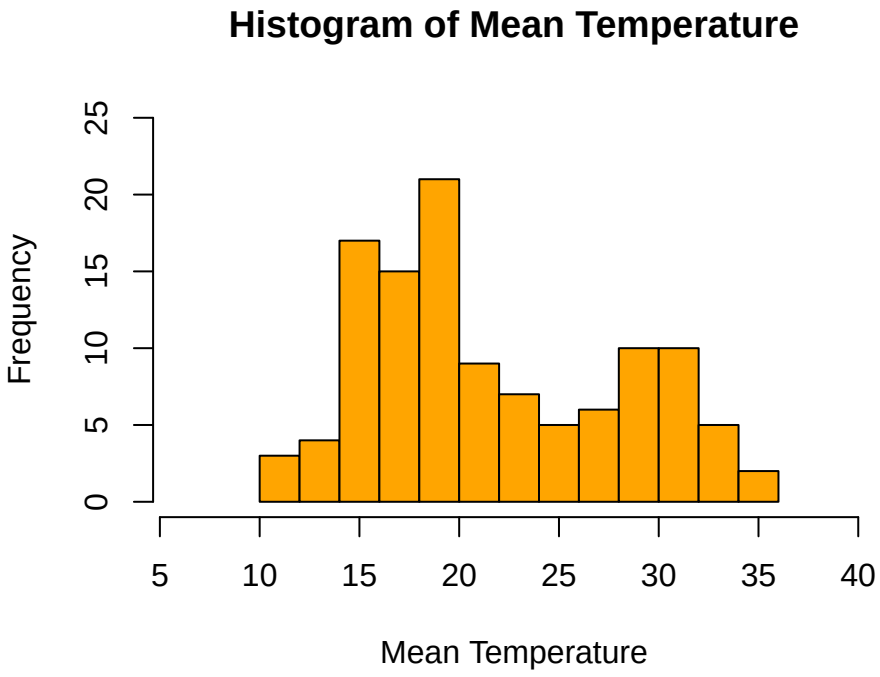
```
dbWriteTable(conn = db.climate, name = 'climate_tbl',
             value = climate_df, overwrite = TRUE)
# run the list tables function to ensure that the table was created
dbListTables(db.climate)
```

```
## [1] "climate_tbl"
```

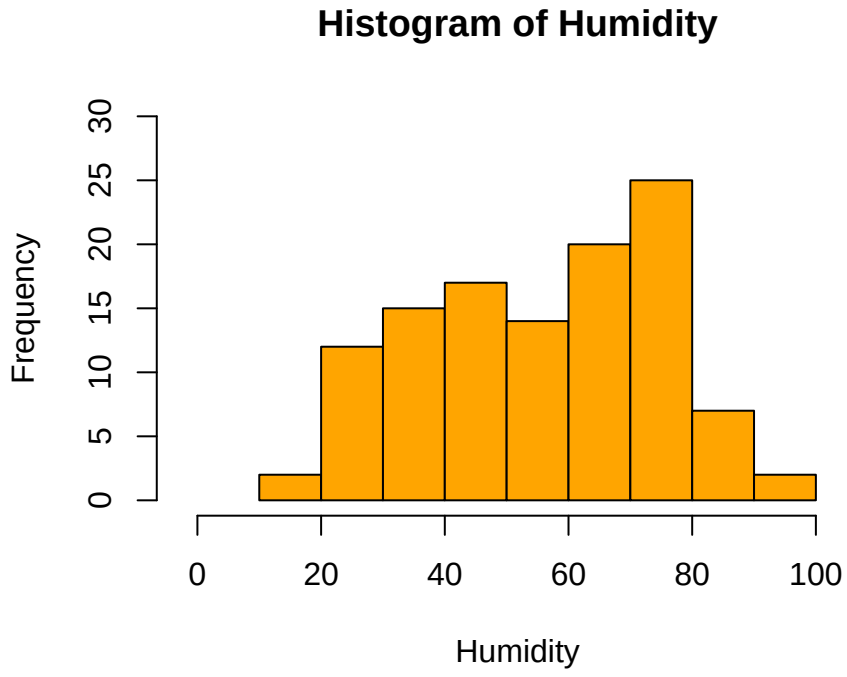
2. Produce histograms for meantemp, humidity, windspeed, and meanpressure

```
# create a tmp var and add a small number to the range to increase the readability of the graph
tmp <- range(climate_df$meantemp) + c(-5,5)
# create the histogram
```

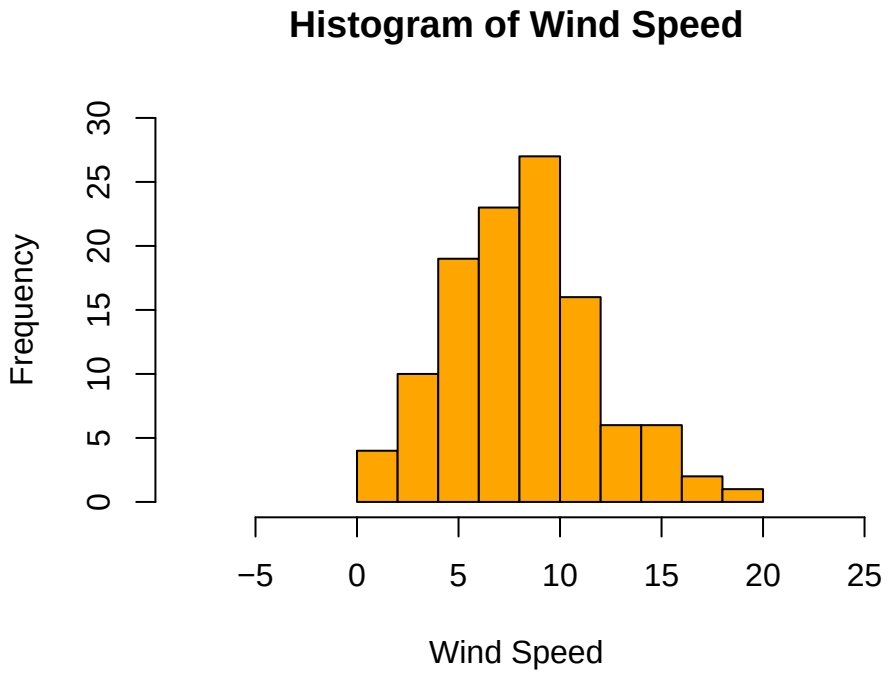
```r
hist(x = climate_df$meantemp, breaks = 10, col = 'orange',
     main = 'Histogram of Mean Temperature', xlim = tmp,
     xlab = 'Mean Temperature', ylim = c(0,25))
```




```r
tmp <- range(climate_df$humidity) + c(-20,10)
hist(x = climate_df$humidity, breaks = 10, col = 'orange',
     xlim = tmp, main = "Histogram of Humidity",
     xlab = "Humidity", ylim = c(0,30))
```

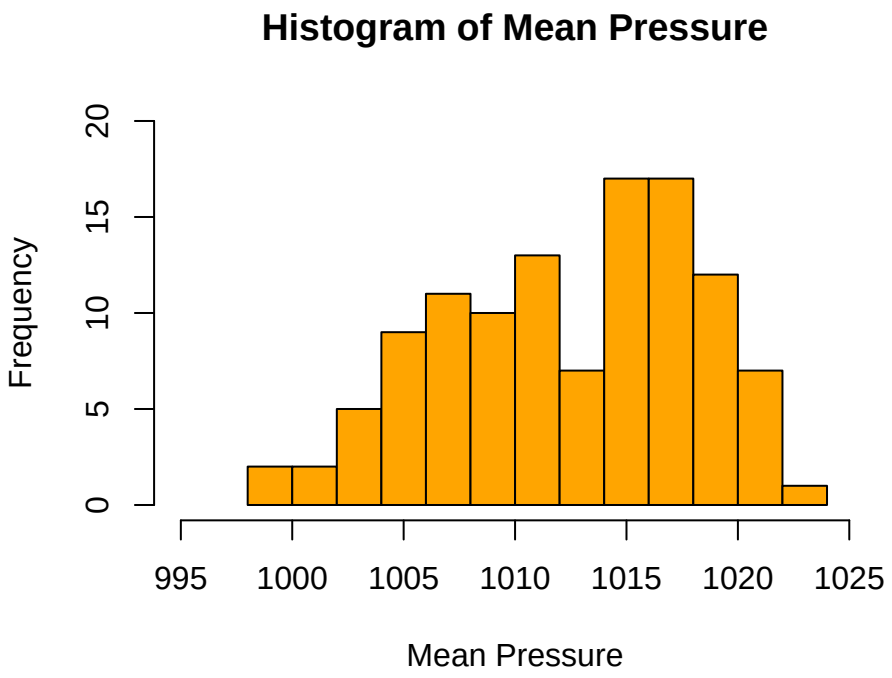


```r
tmp <- range(climate_df$wind_speed) + c(-10, 5)
hist(x = climate_df$wind_speed, breaks = 10, col = 'orange',
     xlim = tmp, main = "Histogram of Wind Speed", xlab = "Wind Speed",
     ylim = c(0,30))
```



– **This column has an extreme outlier which I have purposely ignored for this histogram**

```r
# didn't use a tmp var here due to the outlier, although it could still be done
# use a basic R function to concatenate the data frame
# the outlier is the very first date which has made this easy
hist(x = climate_df$meanpressure[2:114], breaks = 10,
     col = 'orange', xlim = c(995,1025),
     main = "Histogram of Mean Pressure",
     xlab = "Mean Pressure", ylim = c(0,20))
```



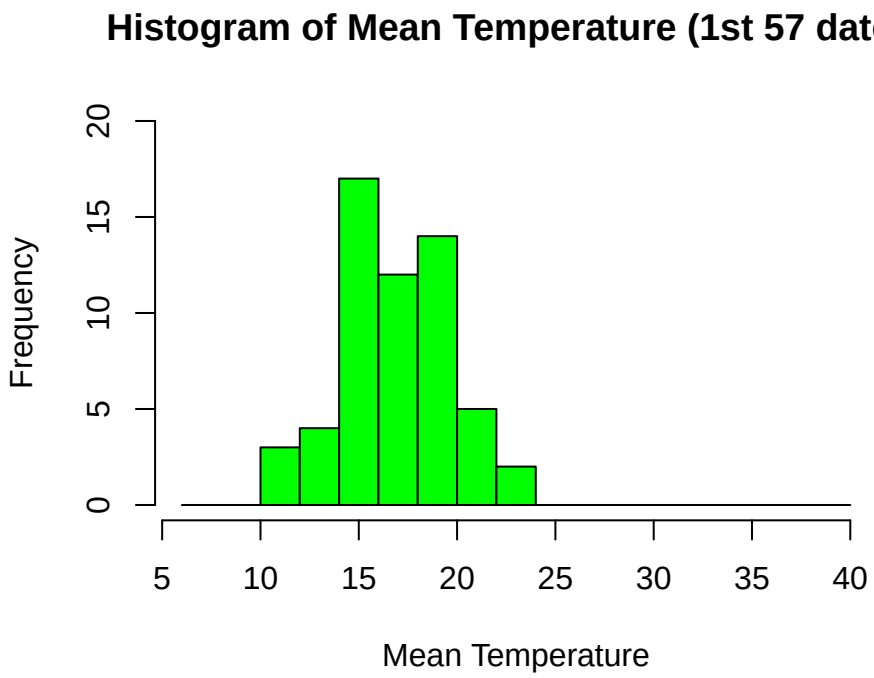
3. Split the data into two parts and produce similar histograms for each new dataset

– Luckily for us, the dataframe is already in order by date, so we don't need to deal with any additional cleaning. Here we can just split the data in two parts by row and create the new histograms from there.

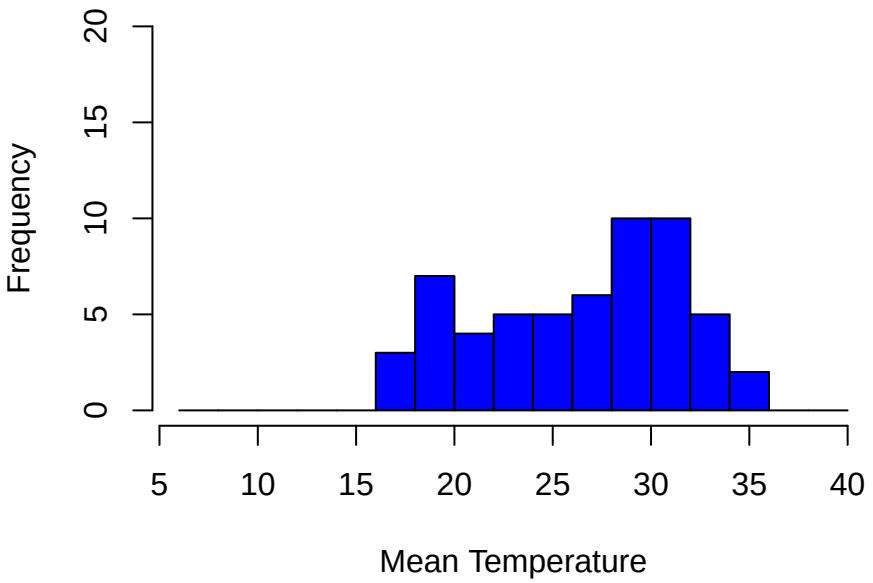
```
climate_first <- climate_df[1:57,]
climate_second <- climate_df[58:114,]
```

– I wanted to choose the same scale for each of the plots to allow for easier comparison

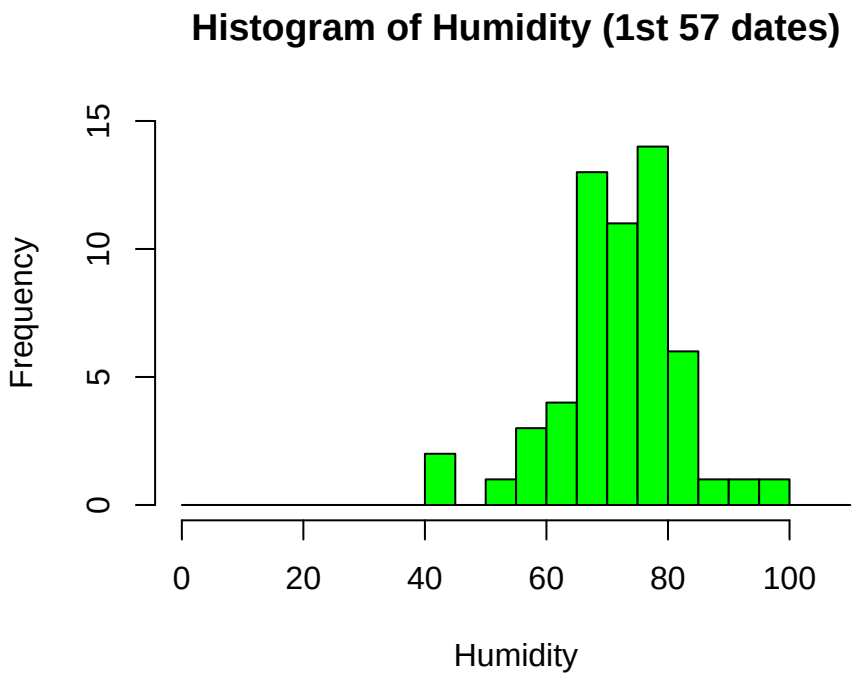
```
# create a sequence based upon the range found in our histogram from #2
# can be used to keep the scale constant between data splits
sqnc <- seq(6,40,2)
hist(x = climate_first$meantemp, breaks = sqnc, col = 'green',
     main = 'Histogram of Mean Temperature (1st 57 dates)',
     xlim = c(6,40), xlab = 'Mean Temperature', ylim = c(0,20))
```



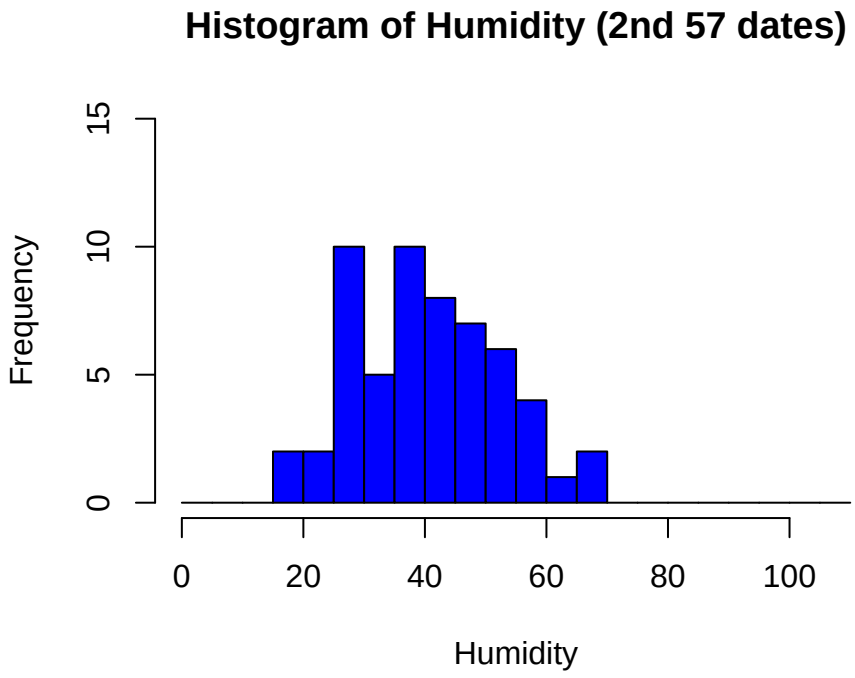

```
hist(x = climate_second$meantemp, breaks = sqnc, col = 'blue',
     main = 'Histogram of Mean Temperature (2nd 57 dates)',
     xlim = c(6,40), xlab = 'Mean Temperature', ylim = c(0,20))
```



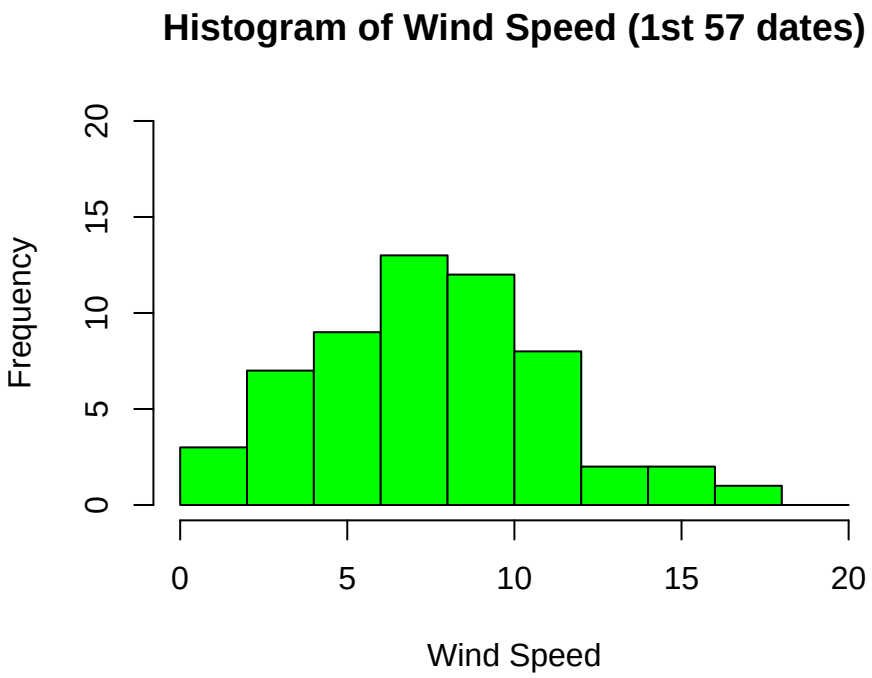

```
sqnc <- seq(0,110,5)
hist(x = climate_first$humidity, breaks = sqnc, col = 'green',
     main = 'Histogram of Humidity (1st 57 dates)',
     xlim = c(0,110), xlab = 'Humidity', ylim = c(0,15))
```



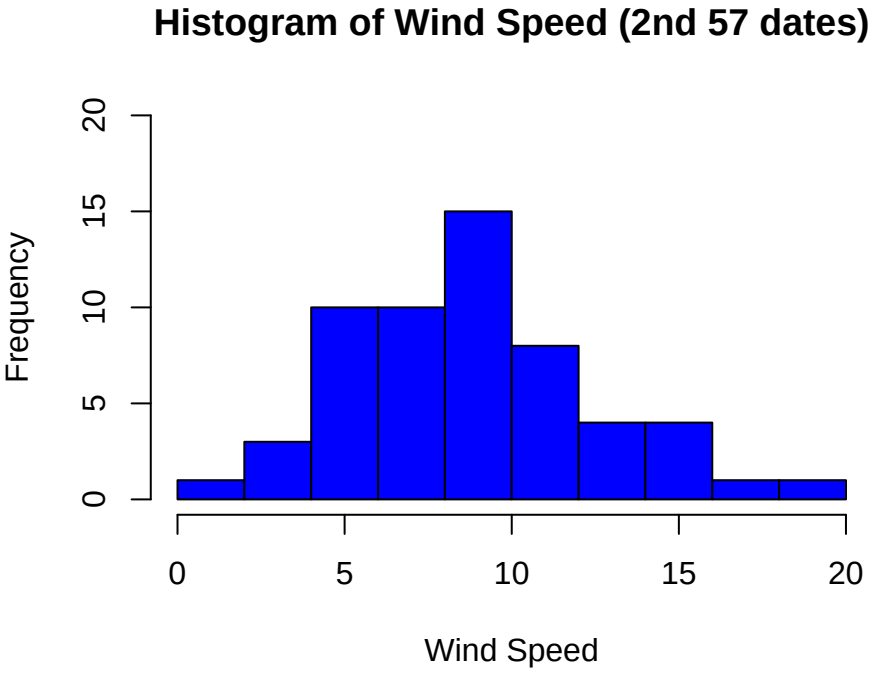

```
hist(x = climate_second$humidity, breaks = sqnc, col = 'blue',
     main = 'Histogram of Humidity (2nd 57 dates)',
     xlim = c(0,110), xlab = 'Humidity', ylim = c(0,15))
```




```
sqnc <- seq(0,20,2)
hist(x = climate_first$wind_speed, breaks = sqnc, col = 'green',
     main = 'Histogram of Wind Speed (1st 57 dates)',
     xlim = c(0,20), xlab = 'Wind Speed', ylim = c(0,20))
```



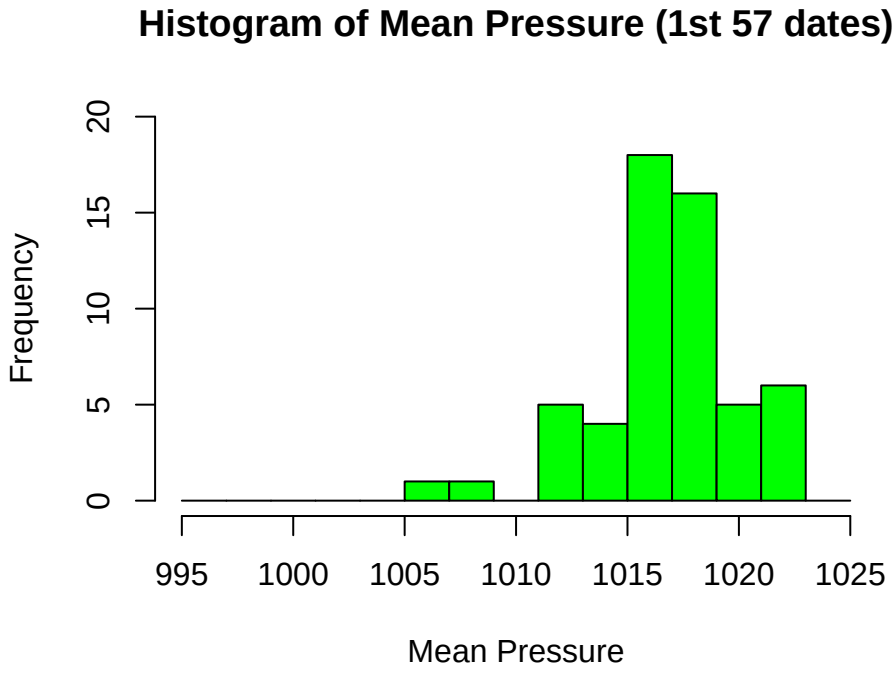

```
hist(x = climate_second$wind_speed, breaks = sqnc, col = 'blue',
     main = 'Histogram of Wind Speed (2nd 57 dates)',
     xlim = c(0,20), xlab = 'Wind Speed', ylim = c(0,20))
```



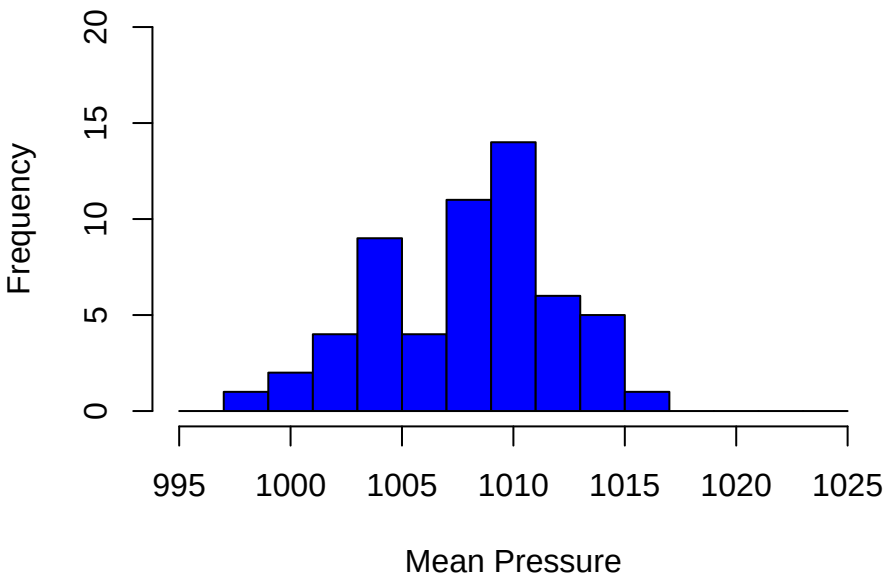
– Like before, the first data point for Mean Pressure is an extreme outlier and probably a mistabulation, therefore I will ignore it for this histogram.

```
sqnc <- seq(995,1025,2)
hist(x = climate_first$meanpressure[2:57], breaks = sqnc,
```

```
     col = 'green', main = 'Histogram of Mean Pressure (1st 57 dates)',
     xlim = c(995,1025), xlab = 'Mean Pressure', ylim = c(0,20))
```




```
hist(x = climate_second$meanpressure, breaks = sqnc, col = 'blue',
     main = 'Histogram of Mean Pressure (2nd 57 dates)',
     xlim = c(995,1025), xlab = 'Mean Pressure', ylim = c(0,20))
```



4. Run T-tests on all 4 columns, comparing the data from climate_first and climate_second

General Hypothesis Test Discussion

1. We will be using a Hypothesis Test called *Welch two sample t-test*. This test takes the *Null Hypothesis*, H_0 and tests it against the *Alternative Hypothesis*, H_1.

2. The *Null Hypothesis* generally represents the case where two populations display no statistical difference within their data. In our case:

$$H_0 : \mu_{first} = \mu_{second}$$

3. The *Alternative Hypothesis* represents the case where the opposite of the null hypothesis is accepted when the null is rejected. For example, if the test is $H_0 : \mu < 10$, then we would display the alternative as $H_1 : \mu \geq 10$. In our case:

$$H_1 : \mu_{first} \neq \mu_{second}$$

4. When the p-value of the test is less than the signficance level, known as α, the null hypothesis is *rejected* in favor of the alternative. In layman's terms, this is because the probability of the null being true is incredible low, therefore it is safe to assume that the null is in fact not the proper fit for the data.

```
# use the t test function from R to compare the means from climate_first and climate_second
temp_ttest <- t.test(climate_first$meantemp, climate_second$meantemp)
temp_ttest
```

```
##
##  Welch Two Sample t-test
##
## data:  climate_first$meantemp and climate_second$meantemp
## t = -13.013, df = 86.403, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.326030  -8.324308
## sample estimates:
## mean of x mean of y
##  16.80049  26.62566
```

OUR RESULTS: With a p-value of essentially 0, it is clear that we will reject the null hypothesis that the sample means of the meantemp variable across the first and second half of the data are the same. As shown in our results, the sample mean of climatefirst is 16.8 while the sample mean of climatesecond is 26.6. Based on our low p-value we know that this difference is statistically significant (I'll use the usual α of 0.05). This certainly could make some sense given that climatefirst covers January and February while climatesecond covers March and April. It is reasonable for one to expect the temperatures to be lower in the Winter months vs the Spring months.

```
# use the t test function from R to compare the means from climate_first and climate_second
temp_ttest <- t.test(climate_first$humidity, climate_second$humidity)
temp_ttest
```

```
##
##  Welch Two Sample t-test
##
## data:  climate_first$humidity and climate_second$humidity
## t = 14.719, df = 108.79, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  26.67663 34.97867
## sample estimates:
## mean of x mean of y
##  71.67219  40.84454
```

OUR RESULTS: With a p-value of essentially 0, it is clear that we will reject the null hypothesis that the sample means of the humidity variable across the first and second half of the data are the same. For humidity, the first half of the data displays a sample mean of 71.7 while the second half of the data displays a sample mean of 40.8. After looking up the Delhi climate on google, it appears that the city experiences its dry climate in March and April, whereas January can be a very wet month. With that said, it is not surprising that the humidity within the first half of the data has been shown to be significantly higher than the second half.

```
# use the t test function from R to compare the means from climate_first and climate_second
temp_ttest <- t.test(climate_first$wind_speed, climate_second$wind_speed)
temp_ttest
```

```
##
##  Welch Two Sample t-test
##
## data:  climate_first$wind_speed and climate_second$wind_speed
## t = -2.0821, df = 111.62, p-value = 0.03962
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.69167033 -0.06665232
## sample estimates:
## mean of x mean of y
##  7.454343  8.833505
```

OUR RESULTS: In a slight diversion from the first two variables, our p-value is not essentially 0. However, the p-value still comes out to be 0.04 which does not exceed our α level of 0.05. This means we still must reject the Null Hypothesis stating that the mean wind speed of each half of the data are equal. We should conclude that the difference between the two means is in fact different! The mean of the first half came out to be 7.45 while the mean of the second half came out to be 8.83.

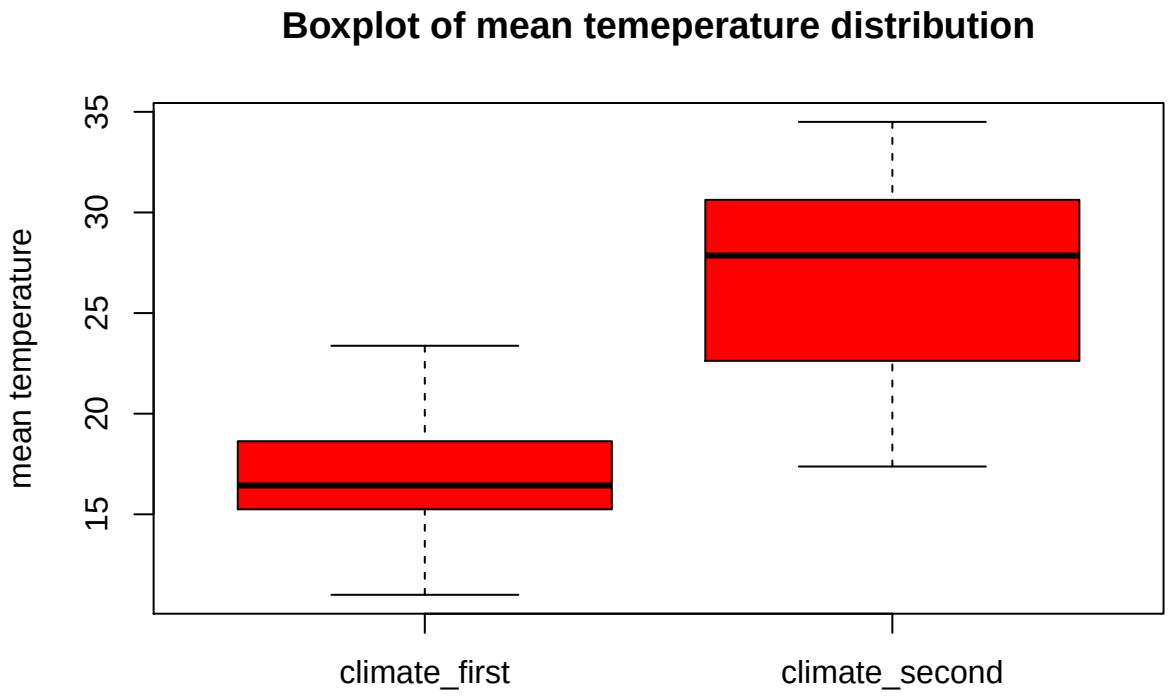
```
# use the t test function from R to compare the means from climate_first and climate_second
temp_ttest <- t.test(climate_first$meanpressure, climate_second$meanpressure)
temp_ttest
```

```
##
##  Welch Two Sample t-test
##
## data:  climate_first$meanpressure and climate_second$meanpressure
## t = -0.48295, df = 56.115, p-value = 0.631
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -41.80954  25.56562
## sample estimates:
## mean of x mean of y
##  999.9741 1008.0961
```

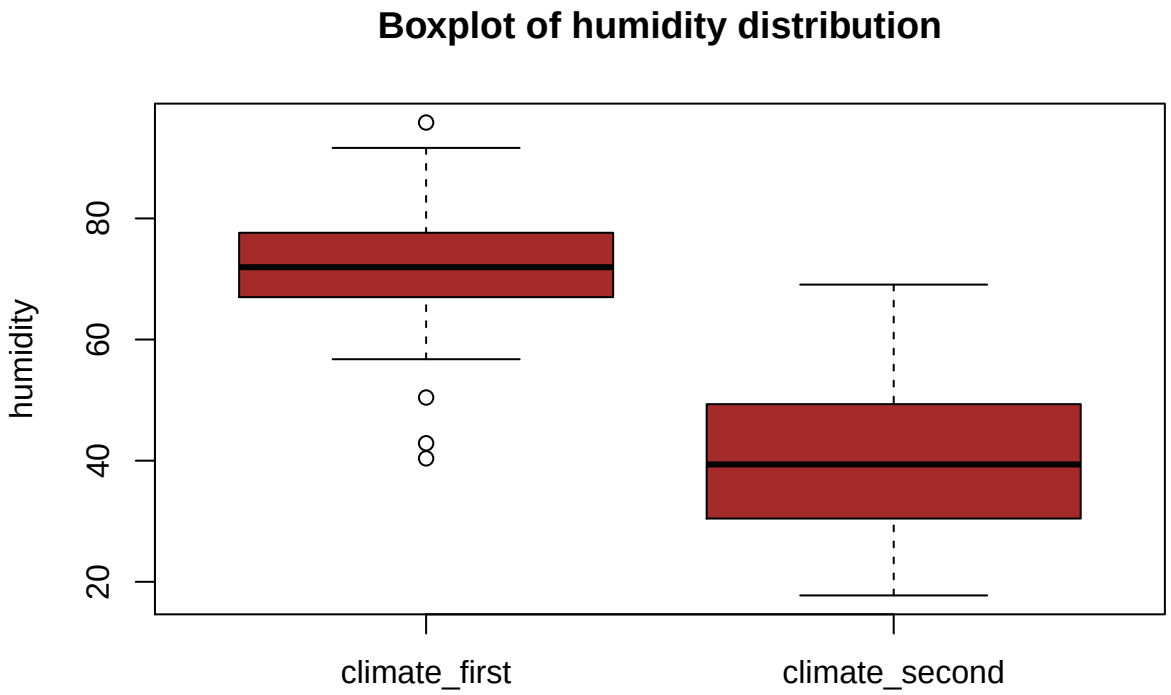
OUR RESULTS: In our t test for mean pressure, we have found a p-value of 0.631, which far exceeds our α level of 0.05. This means we still do not have sufficient evidence to reject the Null Hypothesis stating that the mean mean pressure of each half of the data are equal. In this test, we have found the mean mean pressure of climate first to be 999.97 (This is even skewed by the first outlier but did not impact the results of the test. The p-value likely would have been even higher without it.) and the mean mean pressure of climate second to be 1008.10.

5. Compare the distributions of climate first and climate second

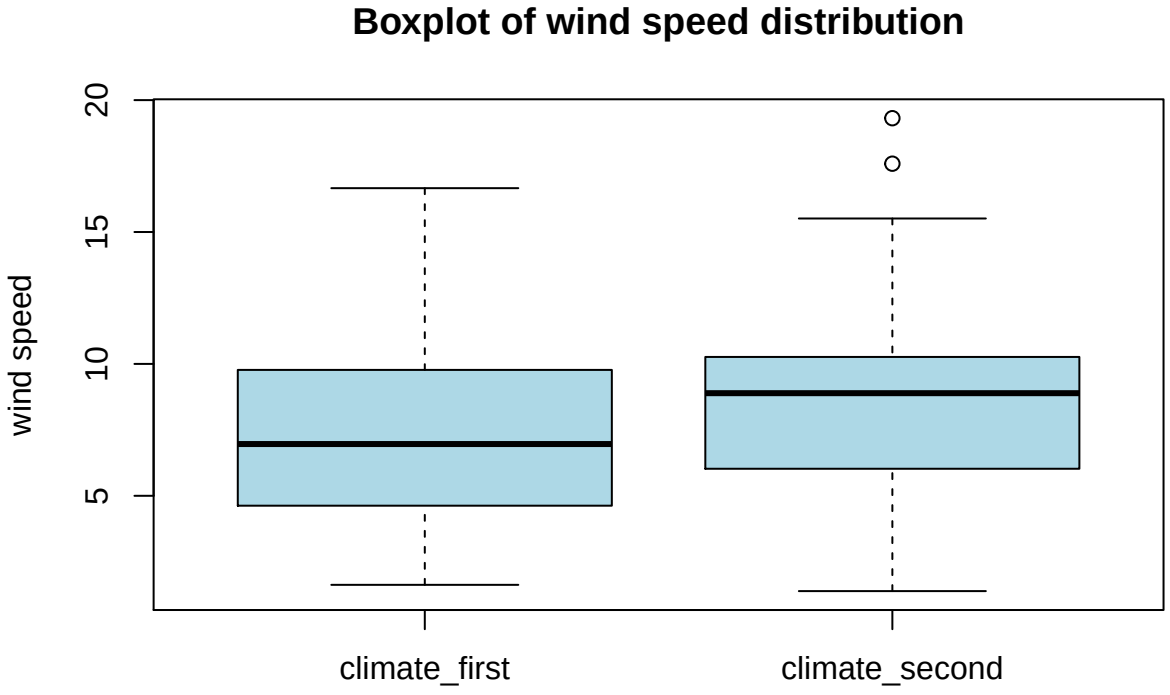
```
boxplot(climate_first$meantemp, climate_second$meantemp,
        names  = c("climate_first", "climate_second"),
        ylab = "mean temperature",
        main = "Boxplot of mean temeperature distribution",
        col = "red")
```



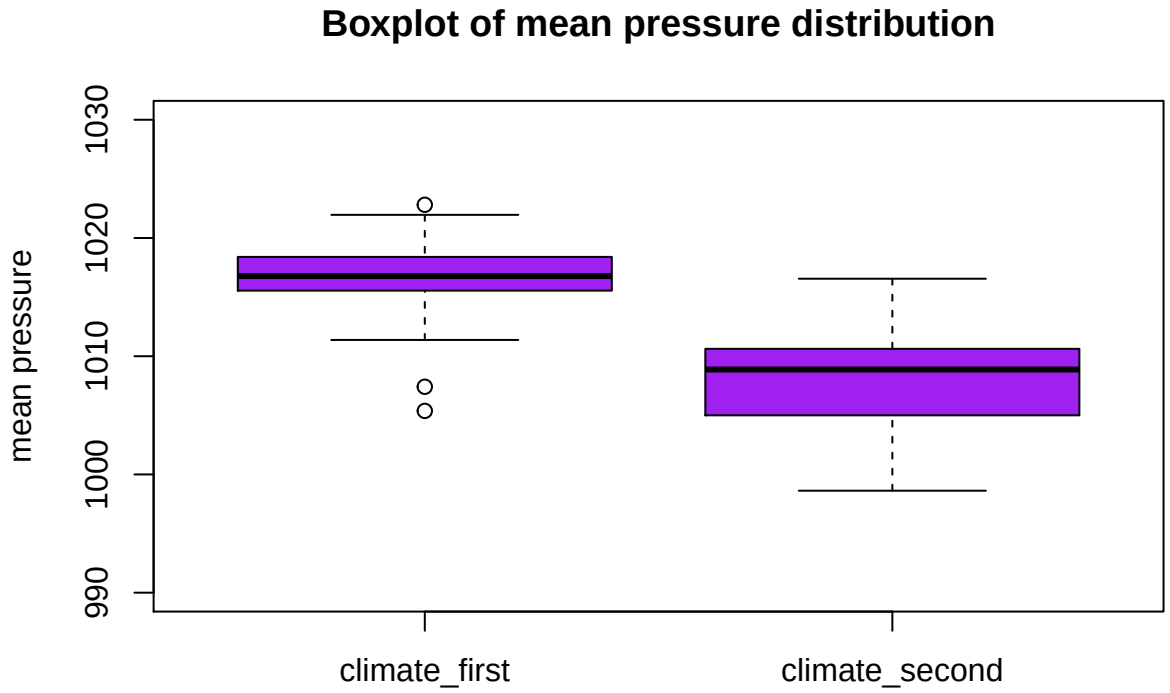

```
boxplot(climate_first$humidity, climate_second$humidity,
        names  = c("climate_first", "climate_second"),
        ylab = "humidity",
        main = "Boxplot of humidity distribution",
        col = "brown")
```




```
boxplot(climate_first$wind_speed, climate_second$wind_speed,
        names  = c("climate_first", "climate_second"),
        ylab = "wind speed",
        main = "Boxplot of wind speed distribution",
        col = "light blue")
```



```
boxplot(climate_first$meanpressure, climate_second$meanpressure,
        ylim = c(990, 1030),
        names  = c("climate_first", "climate_second"),
        ylab = "mean pressure",
        main = "Boxplot of mean pressure distribution",
        col = "purple")
```



We know that the t test is a test of means whereas these boxplots show us the comparison between the climate first and climate second distributions. As it turns out, despite having a statisitically significant difference in means, the distributions of wind speed are remarkably similar. This tells me that the means were likely skewed by a few outliers.

QUESTION TWO: MongoDB

1. Load the data and place it into MongoDB

```
# set up mongodb through R - start mongodb in the terminal first


#install.packages("devtools")
library(devtools)
#install_github(repo = "mongosoup/rmongodb")
#install.packages("rmongodb")
library(rmongodb)
```

```
# connect to mongodb and ensure that it is connected
mongo <- mongo.create(host = "localhost")
mongo.is.connected(mongo)
```

```
## [1] TRUE
```

We need to setup our mongo database, which I have done in the terminal by running the following:

mongoimport –type csv -d countriesDB -c countriesCSV–headerline countries.csv

The -d and -c flags are very important here because these flags determine which database and collection the csv will go in. Once we need to pull data from the csv, we will need both of these names.

```
# confirm that our database was created, we should see 'config' and our new countries DB
mongo.get.databases(mongo)
```

```
## [1] "config"      "countriesDB"
```

2. Find the country with the largest carbon footprint and the largest carbon footprint per capita

```
# create an empty vector to store the data as we iterate
carbon_footprint_vec <- c()
# create the bson from the proper namespace created above
countries_bson <- mongo.find(mongo, ns = "countriesDB.countriesCSV")
# iteratively add the carbon footprint variable to our vector
while(mongo.cursor.next(countries_bson)) {
    tmp_bson <- mongo.cursor.value(countries_bson)
    tmp_json <- mongo.bson.to.list(tmp_bson)
    tmp_footprint <- tmp_json$`Carbon Footprint`
    carbon_footprint_vec <- append(carbon_footprint_vec,tmp_footprint)
}
```

```
# create an empty vector to store the data as we iterate
countries_vec <- c()
# create the bson from the proper namespace created above
countries_bson <- mongo.find(mongo, ns = "countriesDB.countriesCSV")
# iteratively add the countries variable to our vector
while(mongo.cursor.next(countries_bson)) {
    tmp_bson <- mongo.cursor.value(countries_bson)
    tmp_json <- mongo.bson.to.list(tmp_bson)
    tmp_countries <- tmp_json$Country
    countries_vec <- append(countries_vec,tmp_countries)
}
carbon_footprint_vec <- as.numeric(carbon_footprint_vec)
```

```
# create an empty vector to store the data as we iterate
population_vec <- c()
# create the bson from the proper namespace created above
countries_bson <- mongo.find(mongo, ns = "countriesDB.countriesCSV")
```

```
# iteratively add the population variable to our vector
while(mongo.cursor.next(countries_bson)) {
    tmp_bson <- mongo.cursor.value(countries_bson)
    tmp_json <- mongo.bson.to.list(tmp_bson)
    tmp_population <- tmp_json$`Population (millions)`
    population_vec <- append(population_vec,tmp_population)
}
```

```
# create the carbon footprint per capita variable
# simply divide footprint by population
cf_percapita_vec <- carbon_footprint_vec/population_vec
```

```
# create a data frame using the 3 rows we pulled from mongo
countries_df <- data.frame(countries_vec, carbon_footprint_vec, cf_percapita_vec)
```

```
# find the max carbon footprint
max(countries_df$carbon_footprint_vec, na.rm = TRUE)
```

```
## [1] 12.65
```

```
# print the country with the max carbon footprint
countries_df$countries_vec[which.max(countries_df$carbon_footprint_vec)]
```

```
## [1] Luxembourg
## 188 Levels: Afghanistan Albania Algeria Angola Antigua and Barbuda ... Zimbabwe
```

```
# find the max carbon footprint per capita
max(countries_df$cf_percapita_vec, na.rm = TRUE)
```

```
## [1] 66.8
```

```
# print the country with the max carbon footprint per capita
countries_df$countries_vec[which.max(countries_df$cf_percapita_vec)]
```

```
## [1] Saint Kitts and Nevis
## 188 Levels: Afghanistan Albania Algeria Angola Antigua and Barbuda ... Zimbabwe
```

We can see here that the country with the largest carbon footprint is Luxemborg with 12.65 and the country with the largest carbon footprint per capita is Saint Kitts and Nevis with 66.8.

3. Find the probability of P(Region = "NorthAmerica" | Cropland-Footprint > 0.5)

```
library(readr)
countries <- read_csv("countries.csv")
```

```
# find the probability that the region is North America and the
# CroplandFootprint is greater than 0.5


NthAm_CF <- sum(countries$Region == "North America" & countries$`Cropland Footprint` > 0.5, na.rm = TRUE)
NthAm_CF
```

```
## [1] 2
```

```
# find the probability of having a Cropland Footprint of
# greater than 0.5


CF <- sum(countries$`Cropland Footprint` > 0.5, na.rm = TRUE)
CF
```

```
## [1] 89
```

```
# find the conditional probability


cond_prob <- NthAm_CF/CF
cond_prob
```

```
## [1] 0.02247191
```

The conditional probability of a country being located in the North America region given that the country has a cropland footprint of greater than 0.5 is 0.225.

4. Produce a function to return the conditional probability between variable value selections

To get the conditional probability between two variables P(var1 | var2) with one fucntion, use the following:

$P(var1 \mid var2) = \frac{P(var1 \cap var2)}{P(var2)}$

In our case, to find the conditional probability in Q3, we use the following:

$P(NA \mid CF > 0.5) = \frac{P(NA \cap CF > 0.5)}{P(CF > 0.5)}$