**Tools Used:**

**Excel:** I used this for the initial cleaning and processing of the individual csv files, as it provided a quick and simple way to transform the data and create the new columns I needed. I also used it to analyse the average ride length per month, mode and number of rides per month.

**R Studio:** I used RStudio for the bulk of the manipulation, analysis and visualisation due to its ability to handle a large amount of data and its very logical approach to functions and syntax. I found the sqldf library to be a great resource too, as it allowed me to use SQL queries on the large dataframe.

**EXCEL Cleaning & Manipulation:**

Since the data was delivered in a .csv format, I used the “text to columns” function to format the data into rows and columns. I then saved the csv file as an EXCEL workbook. I then removed any duplicates of the data. I converted the columns “started\_at” and “ended\_at” to date time format in the dd/mm/yyyy — hh:mm format. I created a ride\_length column with a formula to subtract the started\_at value from the ended\_at value. I formatted the data in column as hh:mm:ss. I then copied and pasted the values only into a new ride\_length column. Then, I sorted the sheet by ride length in ascending order, and deleted several rows where ride length was negative (indicating an issue where the “ended at” value was before the “started at” value.) I considered deleting the rides where ride length = 0 but decided against it. I created a “day\_of\_week” column, and used the WEEKDAY function to create a numerical representation of the day of the week that each bicycle was checked out, where Sunday was represented by 1. Sorted sheet by “started at date” in ascending order.

**# Working directory**

getwd()

**# Installing Data Packages**

install.packages(“tidyverse”)

install.packages(“markdown”)

install.packages(“sqldf”)

install.packages(“maps”)

install.packages(“rgdal”)

install.packages(“ggrepel”)

library(“tidyverse”)

library(“lubricate”)

library(“markdown”)

library(“sqldf”)

library(“maps”)

library(“rgdal”)

library(“ggrepel”)

library(dplyr)

library(ggplot2)

**# Creating dataframes for each .csv file.**

Apr\_20 <- read.csv(“April20.csv”)

may\_20 <- read.csv(“May20.csv”)

jun\_20 <- read.csv(“June20.csv”)

jul\_20 <- read.csv(“July20.csv”)

aug\_20 <- read.csv(“August20.csv”)

sep\_20 <- read.csv(“September20.csv”)

oct\_20 <- read.csv(“October20.csv”)

nov\_20 <- read.csv(“November20.csv”)

dec\_20 <- read.csv(“December20.csv”)

jan\_21 <- read.csv(“January21.csv”)

feb\_21 <- read.csv(“February21.csv”)

mar\_21 <- read.csv(“March21.csv”)

**# Glimpsing a dataframe**

glimpse(dec\_20)

**# Merging all the dataframes together**

**## first, calculate the number of rows in total to verify the merge**

tot\_rows <- nrow(apr\_20) + nrow(may\_20) + nrow(jun\_20) + nrow(jul\_20) + nrow(aug\_20) + nrow(sep\_20) + nrow(oct\_20) + nrow(nov\_20) + nrow(dec\_20) + nrow(jan\_21) + nrow(feb\_21) + nrow(mar\_21)

**### then create the bind**

df\_1 <- do.call(“rbind”, list(apr\_20, may\_20, jun\_20, jul\_20, aug\_20, sep\_20, oct\_20, nov\_20, dec\_20, jan\_21, feb\_21, mar\_21))

**# checking the number of rows match up**

if (tot\_rows == nrow(df\_1)){

print(“Binding complete, data verified.”)

} else{

print(“Error, please verify your data.”)

}

**# Counting total numbers of members and casual and displaying it**

members <- sqldf (“select count(member\_casual) AS membercount

from df\_1

where member\_casual=’member’”)

members

casuals <- sqldf (“select count(member\_casual) AS casualcount

from df\_1

where member\_casual=’casual’”)

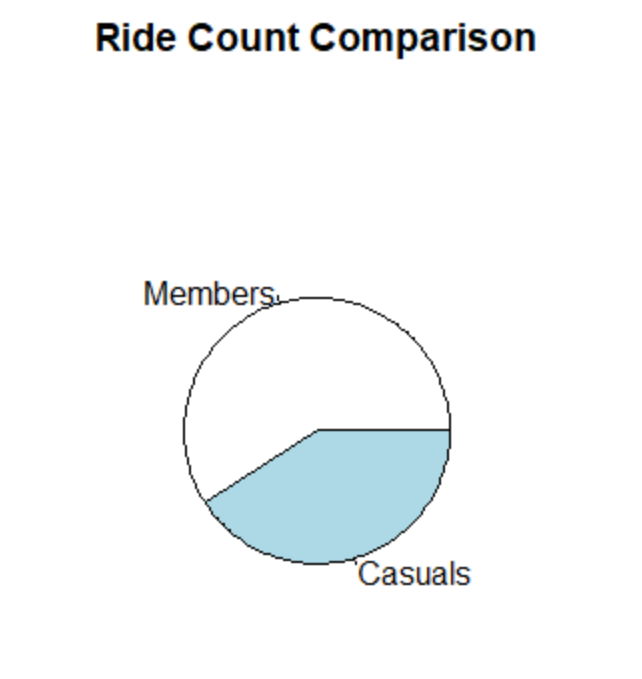
casuals

**# Pie Chart for members and casual**

count <- c (1939441, 1349341)

lbls <- c (“Members”, “Casuals”)

pie(count,labels=lbls,main=”Ride Count Comparison”)



**# changing datatypes of started\_at, ended\_at to datetime and ride\_length to time for all dataframes**

df\_1 <- df\_1 %>%

mutate(started\_at = as\_datetime(df\_1$started\_at, format = “%d/%m/%Y %H:%M”)) %>%

mutate(ended\_at = as\_datetime(df\_1$ended\_at, format = “%d/%m/%Y %H:%M”)) %>%

mutate(ride\_length = as.difftime(df\_1$ride\_length, format = “%H:%M:%S”))

**# A quick analysis to find the mean of the ride\_length column, and the max ride length**

mean\_r\_length <- as.numeric(mean(df\_1$ride\_length))/60

cat(“The average ride length over the year is:”,mean\_r\_length,”minutes”)

**# The average ride length over the year is: 24.63366 minutes**

max\_r\_length <- as.numeric(max(df\_1$ride\_length))/3600

cat("The longest ride for the year was:",max\_r\_length,"hours")

**# The longest ride for the year was: 23.99694 hours**

**# Now, going to create a new dataframe with the data I want for a visualisation.**

**# I will use sqldf to demonstrate some of my SQL abilities.**

**# creating two dataframes with top 5 start & end stations + no. of trips per mem/cas**

**#Top 5 starting geolocations for members**

mem\_start\_geo <- sqldf("SELECT member\_casual, start\_station\_name AS Start,

start\_lat AS Starting\_Latitude,

start\_lng As Starting\_Longitude, count(start\_station\_name) AS Num\_Trips

FROM df\_1

WHERE start\_station\_name IS NOT ''

AND member\_casual = 'member'

GROUP BY start\_station\_name

ORDER BY count(start\_station\_name) DESC

LIMIT 5", method = "auto")

**# Top 5 starting geolocations for casuals**

cas\_start\_geo <- sqldf("SELECT member\_casual, start\_station\_name AS Start,

start\_lat AS Starting\_Latitude, start\_lng As Starting\_Longitude,

count(start\_station\_name) AS Num\_Trips

FROM df\_1

WHERE start\_station\_name IS NOT ''

AND member\_casual = 'casual'

GROUP BY start\_station\_name

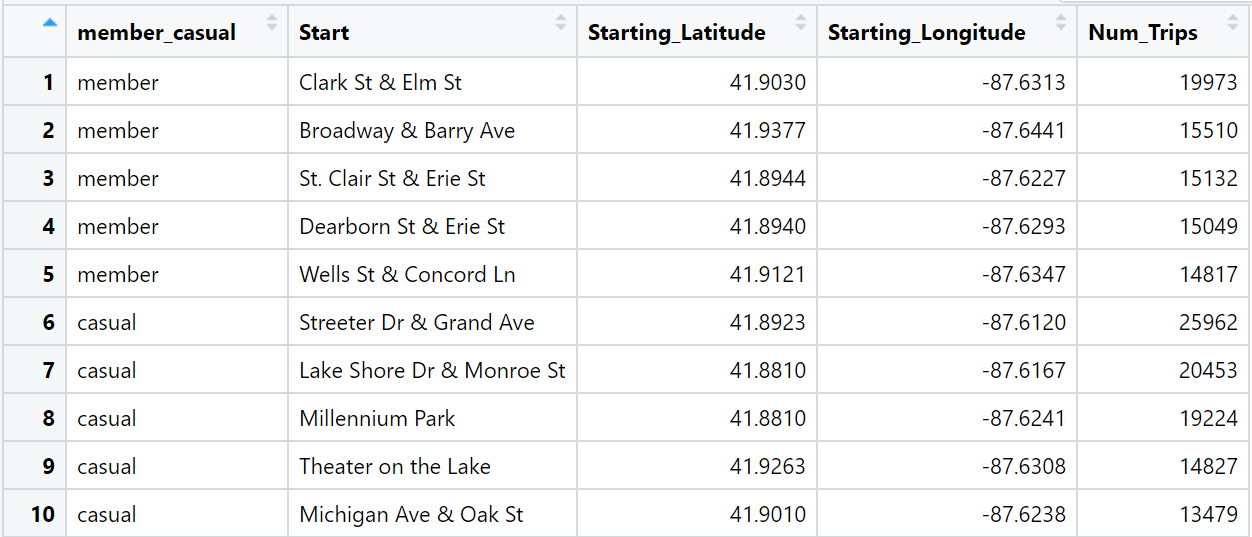
ORDER BY count(start\_station\_name) DESC

LIMIT 5", method = "auto")

**# Binding the two tables into a dataframe, and viewing it**

start\_geo <- rbind(mem\_start\_geo, cas\_start\_geo)

View(start\_geo)



**####Top 5 ending geolocations for members**

mem\_end\_geo <- sqldf("SELECT member\_casual, end\_station\_name AS End,

end\_lat AS Ending\_Latitude,

end\_lng As Ending\_Longitude, count(end\_station\_name) AS Num\_Trips

FROM df\_1

WHERE end\_station\_name IS NOT ''

AND member\_casual = 'member'

GROUP BY end\_station\_name

ORDER BY count(end\_station\_name) DESC

LIMIT 5", method = "auto")

**####Top 5 ending geolocations for casuals**

cas\_end\_geo <- sqldf("SELECT member\_casual, end\_station\_name AS End,

end\_lat AS Ending\_Latitude, end\_lng As Ending\_Longitude,

count(end\_station\_name) AS Num\_Trips

FROM df\_1

WHERE end\_station\_name IS NOT ''

AND member\_casual = 'casual'

GROUP BY end\_station\_name

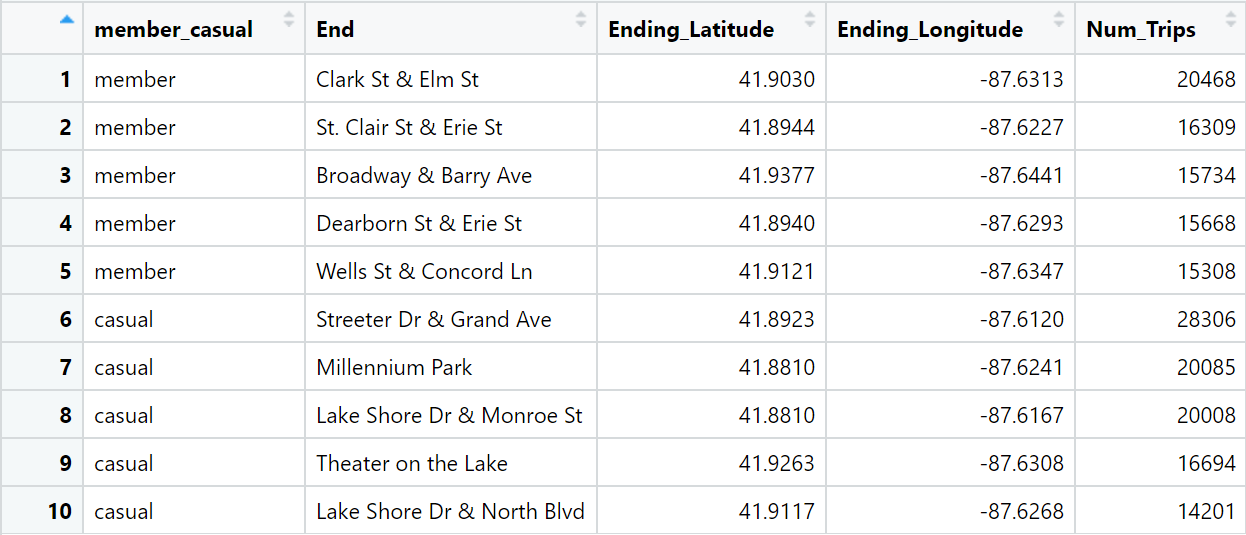
ORDER BY count(end\_station\_name) DESC

LIMIT 5", method = "auto")

**# Binding the two tables into a dataframe, and viewing it**

end\_geo <- rbind(mem\_end\_geo, cas\_end\_geo)

View(end\_geo)



**# SQL Queries for the yearly Mode of day\_of\_week (total, members, casuals)**

mode\_t <- sqldf("SELECT day\_of\_week, member\_casual, COUNT(day\_of\_week) AS Total

FROM df\_1

GROUP BY member\_casual, day\_of\_week

ORDER BY day\_of\_week DESC", method = "auto")

**# Replacing the numerical values with names of weekdays**

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "1"] <- "Sunday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "2"] <- "Monday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "3"] <- "Tuesday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "4"] <- "Wednesday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "5"] <- "Thursday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "6"] <- "Friday"

mode\_t$day\_of\_week[mode\_t$day\_of\_week == "7"] <- "Saturday"

**#Plotting the Modes**

**# This function locks in the order I established so that x axis isn't sorted**

mode\_t$day\_of\_week <- factor(mode\_t$day\_of\_week, levels = rev(unique(mode\_t$day\_of\_week)), ordered=TRUE)

**# This function finds the sum of casual and member riders, to be used to plot labels**

**# in the middle of each bar.**

mode\_t <- mode\_t %>%

arrange(day\_of\_week, rev(member\_casual)) %>%

group\_by(day\_of\_week) %>%

mutate(GTotal = cumsum(Total) - 0.5 \* Total)

**# A stacked bar plot with the yearly modes for all riders**

Mode\_plot <- ggplot(data = mode\_t, aes(x = day\_of\_week, y = Total, fill = member\_casual)) +

scale\_fill\_manual(values=c(member = 'orange', casual= 'blue')) +

geom\_col() +

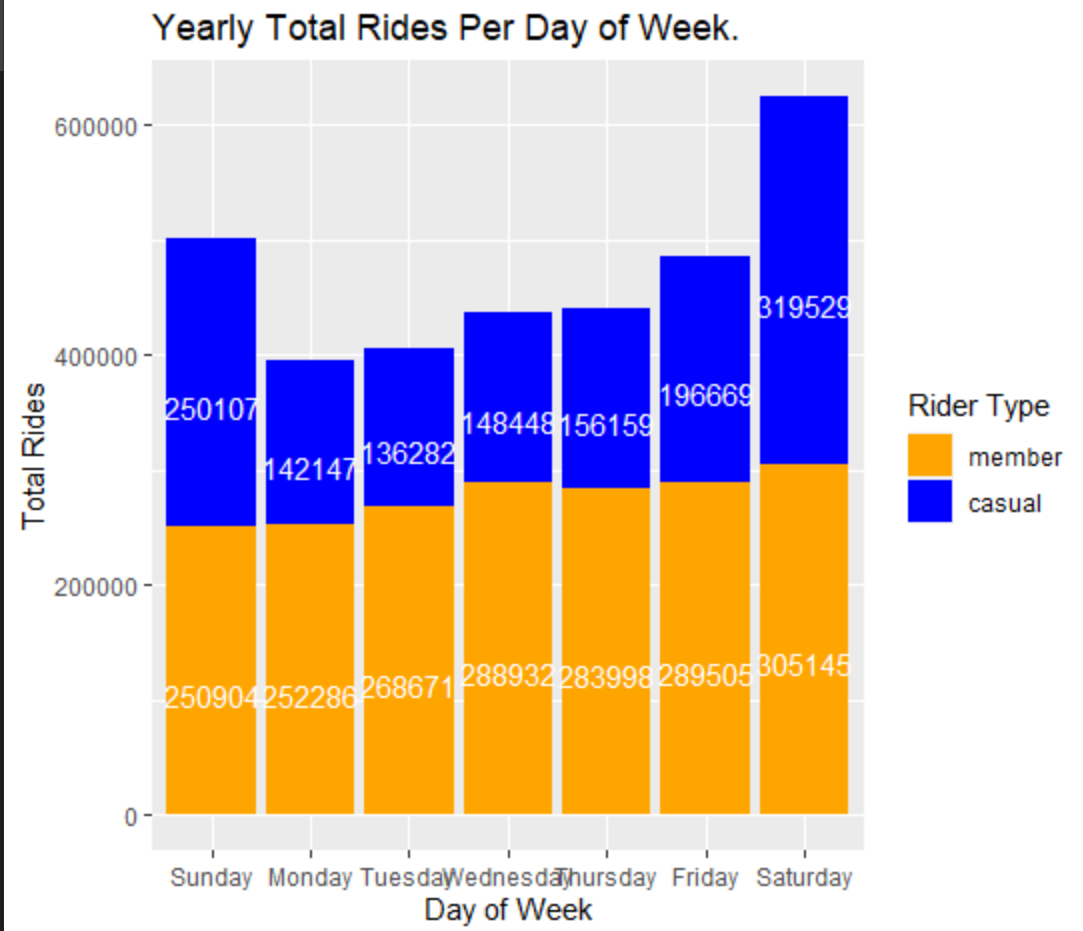
geom\_text(aes(y = GTotal, label = Total), vjust = 1.5, colour = "white") +

labs(title = "Yearly Total Rides Per Day of Week.", x = "Day of Week",

y = "Total Rides", fill = "Rider Type") +

scale\_y\_continuous(labels = function(x) format(x, scientific = FALSE))

Mode\_plot



**# A query to return results related to rideble types used by members**

bike\_df <- sqldf("SELECT rideable\_type, member\_casual, count(rideable\_type) as number\_of\_uses

FROM df\_1

GROUP BY member\_casual, rideable\_type

ORDER BY count(rideable\_type) DESC", method = "auto" )

**# Changing the names of the rideable type to remove the underscore**

bike\_df$rideable\_type[bike\_df$rideable\_type == "classic\_bike"] <- "Classic Bike"

bike\_df$rideable\_type[bike\_df$rideable\_type == "docked\_bike"] <- "Docked Bike"

bike\_df$rideable\_type[bike\_df$rideable\_type == "electric\_bike"] <- "Electric Bike"

**# A side by side bar plot with the yearly count of rideablet for all riders**

bike\_plot <- ggplot(data = bike\_df, aes(x = rideable\_type, y = number\_of\_uses, fill = member\_casual)) +

scale\_fill\_manual(values=c(member = 'orange', casual= 'blue')) +

geom\_col(position = "dodge") +

geom\_text(aes(label = number\_of\_uses), vjust = -0.3 ,colour = "black",

position = position\_dodge(.9)) +

labs(title = "Yearly Total Rides Per Rideable Type.", x = "Rideable Type",

y = "Total Rides", fill = "Rider Type") +

scale\_y\_continuous(labels = function(x) format(x, scientific = FALSE))

bike\_plot

