Cyclistic Data Analysis

Your Name

2024-06-17

## Data Extraction and Initial Setup

First, we need to install and load the necessary packages to extract and manipulate our data from the SQL Server database.

## Install necessary packages if not already installed

#install.packages("odbc")  
#install.packages("DBI")  
#install.packages("kableExtra")

Next, we load the required libraries. ## Load required libraries

Database Connection We need to set up the database connection details and connect to the SQL Server database.

library(odbc)  
library(DBI)  
library(tidyverse)  
library(skimr)  
library(janitor)  
library(tibble)  
library(ggplot2)  
library(knitr) # For table formatting  
library(dplyr)  
library(kableExtra) # For table styling

## Set up database connection details

server <- 'SAMMYLAYO\\SQLEXPRESS02'  
database <- 'Cyclisist'

## Connect to SQL Server database

con <- dbConnect(odbc::odbc(),  
 Driver = "SQL Server",  
 Server = server,  
 Database = database,  
 Trusted\_Connection = "Yes")

Data Retrieval After establishing the connection, we fetch the data using an SQL query and store it in a data frame.

SQL query to fetch data

query <- "SELECT \* FROM CycleData"

Execute the query and fetch the data into a data frame

df <- dbGetQuery(con, query)

Data Overview We will now display some basic information about the data to understand its structure.

## Display basic information about the data

glimpse(df) # Concise summary of the data frame structure

## Rows: 5,743,278  
## Columns: 17  
## $ ride\_id <chr> "00000065B3150FF2", "00000B15294F9057", "000010D58F…  
## $ rideable\_type <chr> "electric\_bike", "electric\_bike", "electric\_bike", …  
## $ started\_at <dttm> 2023-09-24 12:56:50, 2023-12-08 17:52:37, 2023-11-…  
## $ ended\_at <dttm> 2023-09-24 13:00:43, 2023-12-08 18:00:54, 2023-11-…  
## $ start\_station\_name <chr> "Sheridan Rd & Noyes St (NU)", NA, NA, "Sheffield A…  
## $ start\_station\_id <chr> "604", NA, NA, "TA1309000033", NA, "13160", "20246"…  
## $ end\_station\_name <chr> "University Library (NU)", NA, "Ashland Ave & Divis…  
## $ end\_station\_id <chr> "605", NA, "13061", "13243", "644", "13304", "13193…  
## $ start\_lat <dbl> 42.05820, 41.90000, 41.88000, 41.92154, 41.87126, 4…  
## $ start\_lng <dbl> -87.67743, -87.75000, -87.65000, -87.65382, -87.673…  
## $ end\_lat <dbl> 42.05294, 41.90000, 41.90345, 41.91262, 41.86856, 4…  
## $ end\_lng <dbl> -87.67345, -87.76000, -87.66775, -87.68139, -87.686…  
## $ member\_casual <chr> "casual", "casual", "member", "member", "member", "…  
## $ TripDurationMin <dbl> 3, 8, 12, 12, 7, 5, 18, 12, 15, 28, 2, 37, 3, 13, 1…  
## $ WeekDay <chr> "Sunday", "Friday", "Friday", "Wednesday", "Tuesday…  
## $ Month <chr> "September", "December", "November", "June", "May",…  
## $ DayOfMonth <chr> "24", "8", "10", "21", "7", "12", "21", "16", "19",…

str(df) # Showing the structure of the data frame

## 'data.frame': 5743278 obs. of 17 variables:  
## $ ride\_id : chr "00000065B3150FF2" "00000B15294F9057" "000010D58FFC4A2B" "0000140539E54BF2" ...  
## $ rideable\_type : chr "electric\_bike" "electric\_bike" "electric\_bike" "classic\_bike" ...  
## $ started\_at : POSIXct, format: "2023-09-24 12:56:50" "2023-12-08 17:52:37" ...  
## $ ended\_at : POSIXct, format: "2023-09-24 13:00:43" "2023-12-08 18:00:54" ...  
## $ start\_station\_name: chr "Sheridan Rd & Noyes St (NU)" NA NA "Sheffield Ave & Webster Ave" ...  
## $ start\_station\_id : chr "604" NA NA "TA1309000033" ...  
## $ end\_station\_name : chr "University Library (NU)" NA "Ashland Ave & Division St" "Milwaukee Ave & Wabansia Ave" ...  
## $ end\_station\_id : chr "605" NA "13061" "13243" ...  
## $ start\_lat : num 42.1 41.9 41.9 41.9 41.9 ...  
## $ start\_lng : num -87.7 -87.8 -87.7 -87.7 -87.7 ...  
## $ end\_lat : num 42.1 41.9 41.9 41.9 41.9 ...  
## $ end\_lng : num -87.7 -87.8 -87.7 -87.7 -87.7 ...  
## $ member\_casual : chr "casual" "casual" "member" "member" ...  
## $ TripDurationMin : num 3 8 12 12 7 5 18 12 15 28 ...  
## $ WeekDay : chr "Sunday" "Friday" "Friday" "Wednesday" ...  
## $ Month : chr "September" "December" "November" "June" ...  
## $ DayOfMonth : chr "24" "8" "10" "21" ...

colnames(df) # Listing column names of the data frame

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual" "TripDurationMin" "WeekDay"   
## [16] "Month" "DayOfMonth"

We can also use some additional functions to summarize the data without charts.

## Summarize data without charts

skim\_without\_charts(df) # Summary statistics without charts

Data summary

|  |  |
| --- | --- |
| Name | df |
| Number of rows | 5743278 |
| Number of columns | 17 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 10 |
| numeric | 5 |
| POSIXct | 2 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ride\_id | 0 | 1.00 | 8 | 17 | 0 | 5743222 | 0 |
| rideable\_type | 0 | 1.00 | 11 | 13 | 0 | 3 | 0 |
| start\_station\_name | 905237 | 0.84 | 10 | 64 | 0 | 1645 | 0 |
| start\_station\_id | 1359066 | 0.76 | 2 | 35 | 0 | 1610 | 0 |
| end\_station\_name | 956579 | 0.83 | 10 | 64 | 0 | 1658 | 0 |
| end\_station\_id | 1250160 | 0.78 | 2 | 36 | 0 | 1624 | 0 |
| member\_casual | 0 | 1.00 | 6 | 6 | 0 | 2 | 0 |
| WeekDay | 0 | 1.00 | 6 | 9 | 0 | 7 | 0 |
| Month | 0 | 1.00 | 3 | 9 | 0 | 12 | 0 |
| DayOfMonth | 0 | 1.00 | 1 | 2 | 0 | 31 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| start\_lat | 0 | 1 | 41.90 | 0.05 | 41.63 | 41.88 | 41.90 | 41.93 | 42.07 |
| start\_lng | 0 | 1 | -87.65 | 0.03 | -87.94 | -87.66 | -87.64 | -87.63 | -87.46 |
| end\_lat | 7684 | 1 | 41.90 | 0.05 | 0.00 | 41.88 | 41.90 | 41.93 | 42.18 |
| end\_lng | 7684 | 1 | -87.65 | 0.07 | -88.16 | -87.66 | -87.64 | -87.63 | 0.00 |
| TripDurationMin | 0 | 1 | 17.95 | 168.10 | 0.00 | 5.00 | 9.00 | 17.00 | 98489.00 |

**Variable type: POSIXct**

| skim\_variable | n\_missing | complete\_rate | min | max | median | n\_unique |
| --- | --- | --- | --- | --- | --- | --- |
| started\_at | 0 | 1 | 2023-06-01 00:00:44 | 2024-05-31 23:59:00 | 2023-09-28 20:55:19 | 4373744 |
| ended\_at | 0 | 1 | 2023-06-01 00:02:56 | 2024-06-02 00:56:00 | 2023-09-28 21:09:16 | 4384017 |

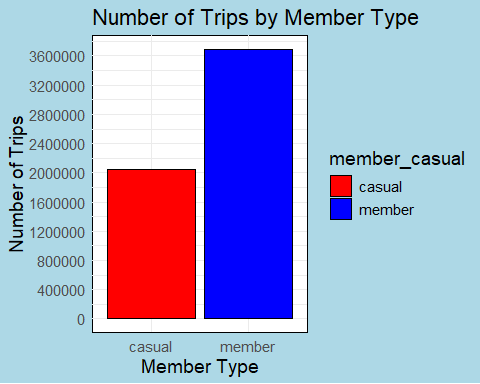
Custom Theme Definition Let’s define a custom theme for our visualizations to ensure they have a consistent appearance.

## Define a custom codes

custom\_theme <- theme\_minimal() +  
 theme(  
 plot.background = element\_rect(fill = "lightblue", color = NA),  
 panel.background = element\_rect(fill = "white"),  
 text = element\_text(size = 14)  
 )  
  
angled\_text <- theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
  
color\_scale <- scale\_fill\_manual(values = c("member" = "blue", "casual" = "red"))  
  
# Set breaks for better readability  
grid\_line <- function(max) {  
 scale\_y\_continuous(  
 breaks = seq(0, max, by = max/15),  
 minor\_breaks = seq(0, max, by = max/30)  
 )  
}

## Create Histogram Showing Number of Trips by Member Type

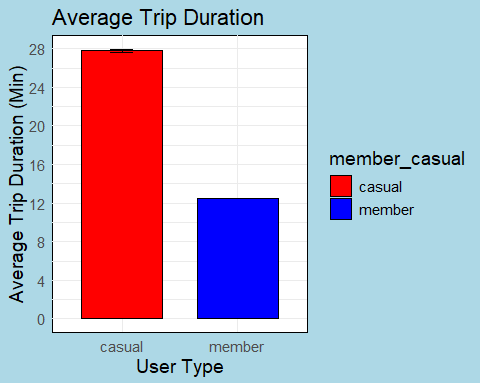
This histogram visualizes the distribution of trips by member type. The y-axis represents the number of trips, and the x-axis categorizes trips into ‘member’ and ‘casual’ types.



The bar heights for ‘casual’ members are slightly above 2 million, while for ‘member’ members, they exceed 3.6 million.

## Average Trip Duration

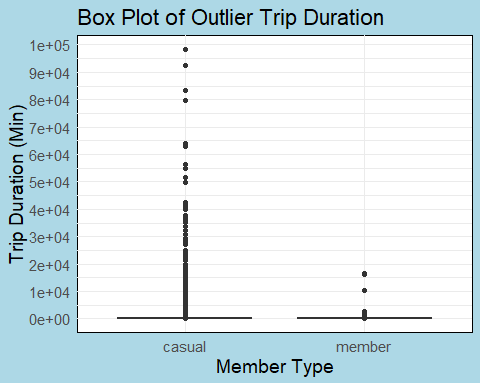
data\_summary <- df %>%  
 group\_by(member\_casual) %>%  
 summarise(  
 mean\_duration = mean(TripDurationMin),  
 se = sd(TripDurationMin) / sqrt(n())  
 )  
  
avg\_trip\_duration\_plot <- ggplot(data\_summary, aes(x = member\_casual, y = mean\_duration, fill = member\_casual)) +  
 geom\_bar(stat = "identity", color = "black", width = 0.7) +  
 geom\_errorbar(aes(ymin = mean\_duration - se, ymax = mean\_duration + se), width = 0.2) +  
 labs(title = "Average Trip Duration", x = "User Type", y = "Average Trip Duration (Min)") +  
 theme\_minimal() +  
 color\_scale +  
 custom\_theme +  
 grid\_line(max=60)  
  
print(avg\_trip\_duration\_plot)



The bar plot with error bars shows that casual users have an average trip duration of around 28 minutes, while members have a shorter average trip duration of around 12 minutes.

## Box Plot of Outlier Trip Duration

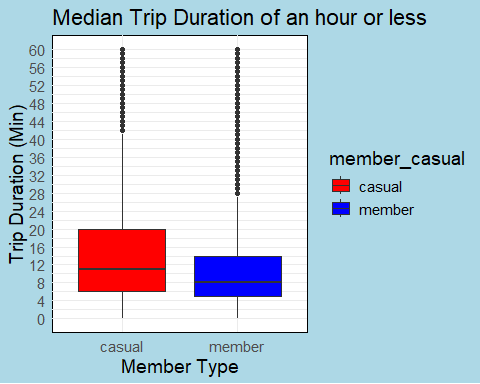
Outlier\_plot <- ggplot(df, aes(x = member\_casual, y = TripDurationMin)) +  
 geom\_boxplot() +  
 labs(title = "Box Plot of Outlier Trip Duration", x = "Member Type", y = "Trip Duration (Min)") +  
 custom\_theme +  
 grid\_line(max=150000)  
  
print(Outlier\_plot)



The box plot shows significant outliers in trip duration for both casual and member users, with casual users having more extreme outliers.

## Median Trip Duration of an Hour or Less

df\_filtered <- df %>%  
 filter(TripDurationMin <= 60)  
  
Median\_plot <- ggplot(df\_filtered, aes(x = member\_casual, y = TripDurationMin, fill = member\_casual)) +  
 geom\_boxplot() +  
 labs(title = "Median Trip Duration of an hour or less", x = "Member Type", y = "Trip Duration (Min)") +  
 theme\_minimal() +  
 color\_scale +  
 custom\_theme +  
 grid\_line(max=60)  
  
print(Median\_plot)



The plot indicates that the median trip duration for casual riders is higher than that for members. Most casual trips last around 20 minutes, while most member trips last around 10 minutes.

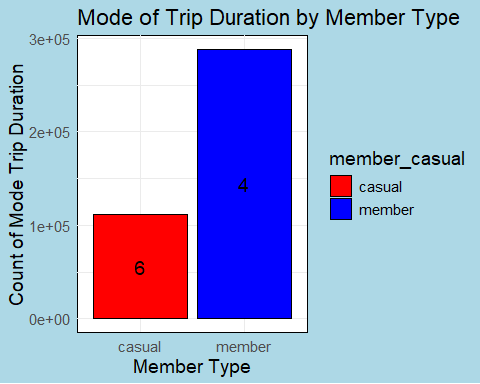
## Mode of Trip Duration

Now, let’s create a histogram to display the mode of trip duration by member type. We’ll make sure the mode values are displayed inside the bars.

# Calculate the mode for each group  
mode\_function <- function(x) {  
 ux <- unique(x)  
 ux[which.max(tabulate(match(x, ux)))]  
}  
  
modes <- df %>%  
 group\_by(member\_casual) %>%  
 summarise(mode\_duration = mode\_function(TripDurationMin))  
  
# Count the occurrences of the mode for each group  
mode\_counts <- df %>%  
 filter(TripDurationMin %in% modes$mode\_duration) %>%  
 group\_by(member\_casual, TripDurationMin) %>%  
 summarise(count = n()) %>%  
 ungroup() %>%  
 inner\_join(modes, by = c("member\_casual", "TripDurationMin" = "mode\_duration"))

## `summarise()` has grouped output by 'member\_casual'. You can override using the  
## `.groups` argument.

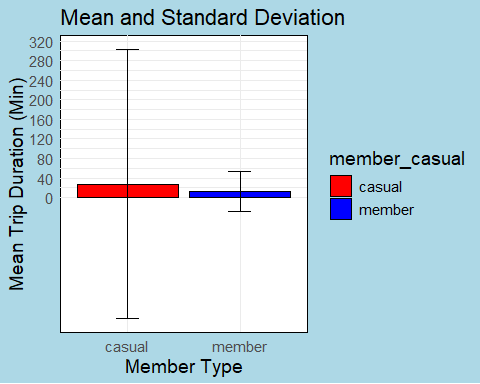
# Create Mode for the Trips  
mode\_histogram <- ggplot(mode\_counts, aes(x = member\_casual, y = count, fill = member\_casual)) +  
 geom\_bar(stat = "identity", color = "black") +  
 geom\_text(aes(label = TripDurationMin), position = position\_stack(vjust = 0.5), size = 5) +  
 labs(title = "Mode of Trip Duration by Member Type", x = "Member Type", y = "Count of Mode Trip Duration") +  
 custom\_theme +  
 color\_scale  
  
print(mode\_histogram)



The plot shows that there are more members with a trip time of 4 minutes, while the casual riders, though have a lesser count, spend more time.

## Mean and Standard Deviation of Trip Durations

# Calculate mean and standard deviation for each member type  
mean\_stdev <- df %>%  
 group\_by(member\_casual) %>%  
 summarise(Mean = mean(TripDurationMin), sd = sd(TripDurationMin))  
  
# Plot mean with error bars for standard deviation  
ggplot(mean\_stdev, aes(x = member\_casual, y = Mean, fill = member\_casual)) +  
 geom\_bar(position = position\_dodge(), stat = "identity", colour = 'black') +  
 geom\_errorbar(aes(ymin = Mean - sd, ymax = Mean + sd), width = 0.2) +  
 labs(title = "Mean and Standard Deviation", x = "Member Type", y = "Mean Trip Duration (Min)") +  
 theme\_minimal() +  
 custom\_theme +  
 color\_scale +  
 grid\_line(max = 600)

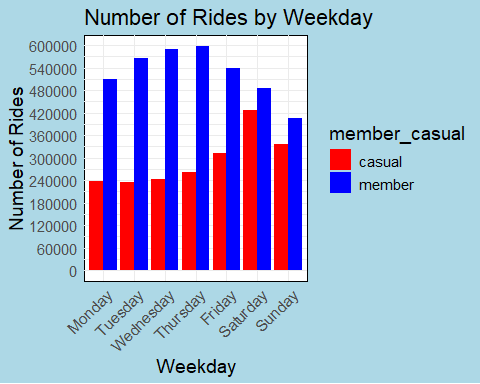
 The mean trip duration for casual riders is significantly higher than for members. Casual riders also have a larger standard deviation, indicating more variability in trip durations.

## Number of Rides by Weekday

# Plot number of rides by weekday  
df\_aggregated <- df %>%  
 group\_by(WeekDay, member\_casual) %>%  
 summarize(Count = n()) %>%  
 ungroup()

## `summarise()` has grouped output by 'WeekDay'. You can override using the  
## `.groups` argument.

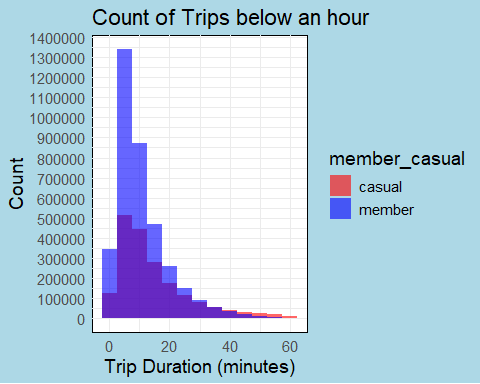
df\_aggregated$WeekDay <- factor(df\_aggregated$WeekDay, levels = c("Monday","Tuesday","Wednesday","Thursday","Friday", "Saturday", "Sunday"))  
  
ggplot(df\_aggregated, aes(x = WeekDay, y = Count, fill = member\_casual)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 scale\_fill\_manual(values = c("casual" = "red", "member" = "blue")) +  
 labs(title = "Number of Rides by Weekday", x = "Weekday", y = "Number of Rides") +  
 theme\_minimal() +  
 custom\_theme +  
 angled\_text +  
 grid\_line(max = 900000)



The plot shows the distribution of rides throughout the week, with both casual and member riders having more rides on weekends compared to weekdays.

## Distribution of Ride Durations Below 60 Minutes

# Plot distribution of ride durations below 60 minutes  
ggplot(df\_filtered, aes(x = TripDurationMin, fill = member\_casual)) +  
 geom\_histogram(binwidth = 5, alpha = 0.6, position = "identity") +  
 color\_scale +  
 labs(title = "Count of Trips below an hour", x = "Trip Duration (minutes)", y = "Count") +  
 theme\_minimal() +  
 custom\_theme +  
 grid\_line(max = 1500000)



The histogram reveals that the majority of trips for both casual and member riders are clustered between 5 to 20 minutes, with casual riders having a higher count of longer trips compared to members.

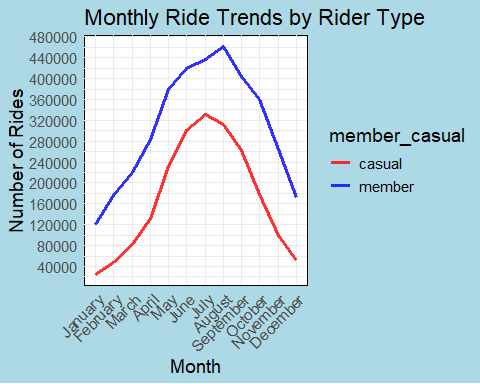
## Monthly Ride Trends by Rider Type

# Plot monthly ride trends by rider type  
df <- df %>%  
 mutate(Month = factor(month(started\_at, label = TRUE, abbr = FALSE), levels = month.name)) # Order months by name  
  
monthly\_rides <- df %>%  
 group\_by(Month, member\_casual) %>%  
 summarize(Count = n()) %>%  
 ungroup()

## `summarise()` has grouped output by 'Month'. You can override using the  
## `.groups` argument.

ggplot(monthly\_rides, aes(x = Month, y = Count, color = member\_casual, group = member\_casual)) +  
 geom\_line(size = 1.2, alpha = 0.8) +  
 scale\_color\_manual(values = c("casual" = "red", "member" = "blue")) +  
 labs(title = "Monthly Ride Trends by Rider Type", x = "Month", y = "Number of Rides") +  
 theme\_minimal() +  
 custom\_theme +  
 grid\_line(max = 600000) +  
 angled\_text

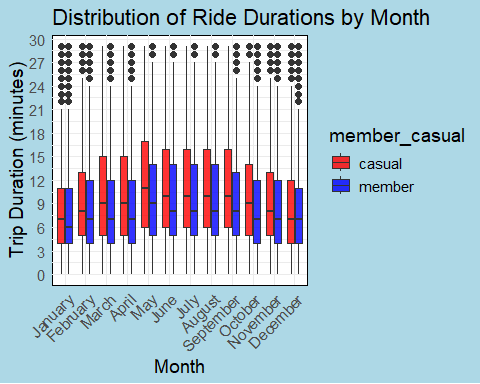
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



The line plot shows the monthly ride trends for both casual and member riders, indicating seasonal variations in ridership with peaks during the summer months and declines during the winter months

## Distribution of Ride Durations by Month and Rider Type

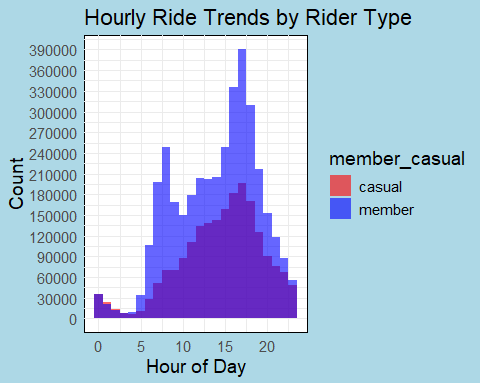
# Plot distribution of ride durations by month and rider type  
df\_filter <- df %>%  
 filter(TripDurationMin < 30)  
  
ggplot(df\_filter, aes(x = Month, y = TripDurationMin, fill = member\_casual)) +  
 geom\_boxplot(alpha = 0.8, outlier.size = 2) +  
 color\_scale +  
 labs(title = "Distribution of Ride Durations by Month", x = "Month", y = "Trip Duration (minutes)") +  
 theme\_minimal() + custom\_theme + angled\_text + grid\_line(max=45)



This plot shows that casual riders (red) generally have longer ride durations compared to members (blue), with notable consistency across the months. The median trip duration for casual riders is higher throughout the year.

## Hourly Ride Trends by Rider Type

# Plot hourly ride trends by rider type  
df <- df %>%  
 mutate(Hour = hour(started\_at))  
  
ggplot(df, aes(x = Hour, fill = member\_casual)) +  
 geom\_histogram(binwidth = 1, alpha = 0.6, position = "identity") +  
 color\_scale +  
 labs(title = "Hourly Ride Trends by Rider Type", x = "Hour of Day", y = "Count") +  
 theme\_minimal() + grid\_line(max=450000) + custom\_theme



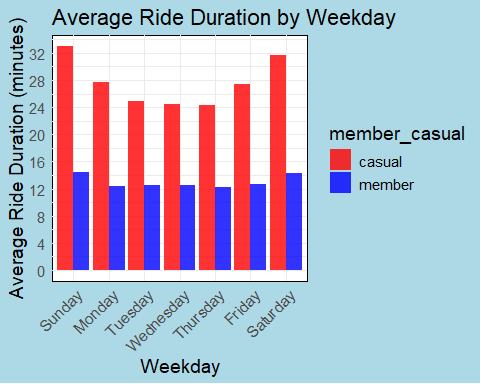
The histogram highlights peak ride hours, showing that members (blue) have significant ride counts during morning and evening hours, indicating commuting patterns. Casual riders (red) have more scattered ride times throughout the day, with a peak around the afternoon.

## Average Ride Duration by Weekday and Rider Type

# Calculate average ride duration by weekday and rider type  
df <- df %>%  
 mutate(RideDurationMin = as.numeric(difftime(ended\_at, started\_at, units = "mins")))  
  
avg\_duration <- df %>%  
 group\_by(WeekDay, member\_casual) %>%  
 summarize(AvgDuration = mean(RideDurationMin, na.rm = TRUE))

## `summarise()` has grouped output by 'WeekDay'. You can override using the  
## `.groups` argument.

# Define the order of weekdays  
weekday\_order <- c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")  
  
avg\_duration$WeekDay <- factor(avg\_duration$WeekDay, levels = weekday\_order, ordered = TRUE)  
  
ggplot(avg\_duration, aes(x = WeekDay, y = AvgDuration, fill = member\_casual)) +  
 geom\_bar(stat = "identity", position = "dodge", alpha = 0.8) +  
 color\_scale +   
 labs(title = "Average Ride Duration by Weekday", x = "Weekday", y = "Average Ride Duration (minutes)") +  
 theme\_minimal() + custom\_theme + angled\_text + grid\_line(max=60)



This bar plot shows that casual riders consistently have longer average ride durations compared to members. The longest average durations for casual riders are on Sundays and Saturdays, indicating more leisurely rides during weekends.

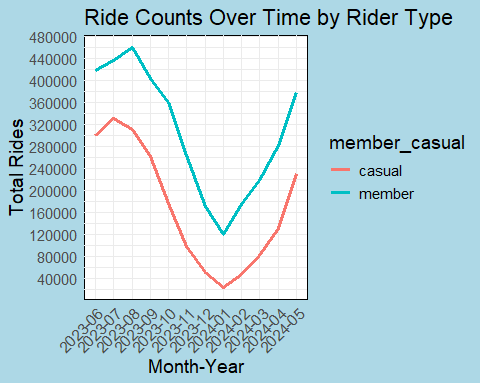
## Ride Counts Over Time by Rider Type

# Plot ride counts over time by rider type  
df <- df %>%  
 mutate(MonthYear = format(started\_at, "%Y-%m"),  
 Month = month(started\_at, label = TRUE, abbr = FALSE),  
 Year = year(started\_at))  
  
ride\_counts <- df %>%  
 group\_by(MonthYear, member\_casual) %>%  
 summarize(TotalRides = n())

## `summarise()` has grouped output by 'MonthYear'. You can override using the  
## `.groups` argument.

ggplot(ride\_counts, aes(x = MonthYear, y = TotalRides, color = member\_casual, group = member\_casual)) +  
 geom\_line(size = 1.2) +  
 labs(title = "Ride Counts Over Time by Rider Type", x = "Month-Year", y = "Total Rides") +  
 theme\_minimal() + custom\_theme + angled\_text + color\_scale + grid\_line(max=600000)

## Warning: No shared levels found between `names(values)` of the manual scale and the  
## data's fill values.



The line chart illustrates trends in ride counts over time, revealing that member rides (blue) consistently outnumber casual rides (red). Both types show seasonal variation, with peaks in summer and declines in winter. However, the fluctuation is more pronounced for casual riders.

## Mapping Station Locations on a World Map

First, we need to install and load additional packages for spatial data manipulation and mapping.

# Install additional packages for mapping  
#install.packages("sf")  
#install.packages("maps")  
  
library(sf)

## Linking to GEOS 3.12.1, GDAL 3.8.4, PROJ 9.3.1; sf\_use\_s2() is TRUE

library(maps)

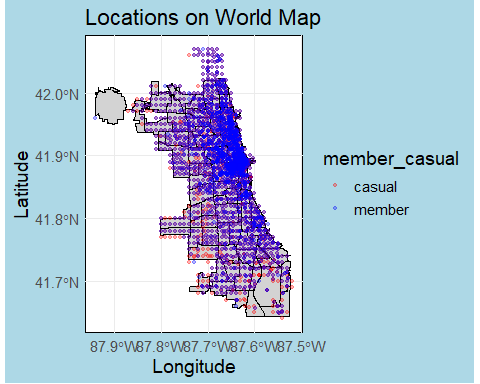
##   
## Attaching package: 'maps'

## The following object is masked from 'package:purrr':  
##   
## map

# Read the path to your shapefile  
shapefile\_path <- "C:/Users/ADEBAYO/Videos/Boundaries/chicago.dbf"  
shapefile\_data <- st\_read(shapefile\_path)

## Reading layer `chicago' from data source   
## `C:\Users\ADEBAYO\Videos\Boundaries\chicago.dbf' using driver `ESRI Shapefile'  
## Simple feature collection with 77 features and 9 fields  
## Geometry type: MULTIPOLYGON  
## Dimension: XY  
## Bounding box: xmin: -87.94011 ymin: 41.64454 xmax: -87.52414 ymax: 42.02304  
## Geodetic CRS: WGS 84

# Query to get location data  
LocationQuery <- "Select member\_casual, start\_lat, start\_lng, count(\*) as Trips  
 from CycleData  
 group by member\_casual, start\_lat, start\_lng  
 having count(\*) > 1"  
  
Location <- dbGetQuery(con, LocationQuery)  
  
# Create spatial points data frame  
coordinates\_sf <- st\_as\_sf(Location, coords = c("start\_lng", "start\_lat"), crs = 4326)  
  
# Create world map data  
world\_map <- map\_data("world")  
chicago\_map <- subset(world\_map, region %in% c("illinois", "chicago"))  
  
# Plot locations on world map  
ggplot() +  
 geom\_sf(data = shapefile\_data, fill = "lightgrey", color = "black") + # Plot the shapefile  
 geom\_sf(data = coordinates\_sf, aes(color = member\_casual), size = 1, alpha = 0.3) + # Plot the points  
 scale\_color\_manual(values = c("member" = "blue", "casual" = "red")) + # Define colors for each subcategory  
 theme\_minimal() +  
 labs(title = "Locations on World Map", x = "Longitude", y = "Latitude") + custom\_theme

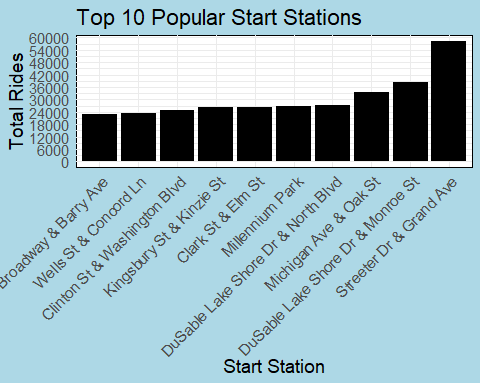


This script maps the station locations onto a world map. Points are colored based on the rider type (member or casual). The shapefile data for Chicago is used to provide a detailed map background.

## Identifying Top 10 Popular Start Stations

The following script analyzes the data to find the top 10 start stations based on ride counts.

# Analyze popular start stations  
start\_station\_counts <- df %>%   
 drop\_na() %>%  
 group\_by(start\_station\_name) %>%  
 summarize(TotalRides = n()) %>%  
 top\_n(10, TotalRides) # Top 10 start stations by ride count  
  
ggplot(start\_station\_counts, aes(x = reorder(start\_station\_name, TotalRides), y = TotalRides)) +  
 geom\_bar(stat = "identity", fill = "black") +  
 labs(title = "Top 10 Popular Start Stations", x = "Start Station", y = "Total Rides") +  
 theme\_minimal() + custom\_theme + grid\_line(max=90000) + angled\_text



This bar plot highlights the top 10 most popular start stations based on the total number of rides originating from each station. The bars are ordered by ride counts and labeled for clarity.

## Conclusion

Cyclist Data Analysis: Recommendations for Converting Casual Riders to Annual Members Based on the analysis of Cyclistic bike-share data, there are significant differences in how casual riders and annual members use the service. These insights can guide the development of a targeted marketing strategy to convert casual riders into annual members, thereby maximizing the company’s future growth and profitability.

### Key Findings:

### Ride Volume and Duration:

Casual riders take fewer trips but have longer average trip durations (around 28 minutes) compared to annual members (around 12 minutes). This indicates that casual riders may be using the service more for leisure or extended activities, while members are likely using it for shorter, utilitarian purposes such as commuting. Peak Usage Times:

Casual riders show a peak in the afternoon, while members have significant ride counts during morning and evening hours, indicating commute patterns. Both groups have higher ride counts on weekends, with casual riders showing more variability in their trip durations. Seasonal Trends:

Ride counts for both casual and member riders peak during the summer months and decline in the winter, with casual riders showing more pronounced seasonal fluctuations. Geographical Distribution:

Members’ rides are more concentrated in specific areas, suggesting routine travel routes, while casual riders’ rides are more dispersed, possibly indicating exploratory or leisure rides. Top Start Stations:

The analysis identifies the top 10 most popular start stations, which can be focal points for targeted promotions and infrastructure improvements.

## Recommendations

### Promotional Offers:

Seasonal Discounts: Offer discounts on annual memberships during peak casual riding seasons (spring and summer) to leverage the higher ride volumes and encourage conversion when casual riders are most engaged. Trial Memberships: Introduce short-term trial memberships at a reduced cost to give casual riders a taste of the benefits of being a member without a long-term commitment.

### Marketing Campaigns:

Commute Incentives: Highlight the benefits of annual memberships for daily commuting, such as cost savings and convenience. Use data to show how members save money compared to frequent single-ride or full-day pass purchases. Leisure and Exploration: Promote the flexibility and freedom of unlimited rides to casual riders who use the service for leisure. Emphasize the variety of bike options available, including reclining bikes and cargo bikes, which cater to a broader audience.

### Digital Media Strategy:

Targeted Advertising: Utilize digital media channels to target casual riders with personalized ads based on their ride patterns and preferences. For example, display ads that highlight membership benefits during their typical riding hours or seasons. Engaging Content: Create engaging content, such as testimonials and success stories from current members, to build social proof and highlight the community aspect of being an annual member.

### Infrastructure and Service Improvements:

Popular Stations: Enhance facilities and services at the top 10 most popular start stations identified in the analysis. This could include better signage, additional docking stations, and promotional materials highlighting membership benefits. Exclusive Member Benefits: Introduce exclusive benefits for members, such as priority access to bikes during peak hours or special offers from partner businesses located near popular docking stations.

### Community and Events:

Member-Exclusive Events: Organize community events and rides exclusive to members, creating a sense of belonging and community. This can include guided tours, social rides, and member-only meetups. Partnerships: Partner with local businesses and attractions to offer members discounts and incentives, further enhancing the value of an annual membership.

By implementing these strategies, Cyclistic can effectively convert casual riders into annual members, driving growth and ensuring long-term profitability. The insights derived from the data analysis provide a solid foundation for designing marketing campaigns that resonate with casual riders and highlight the compelling advantages of becoming an annual member.