R Notebook

Overview

This repository contains the code and data for a data analysis project focused on exploring the ridership patterns of the bike-share scheme in Toronto. The analysis is conducted using the R programming language to gain insights into user behaviors, popular routes, and temporal trends.

Data Source

- Ref:
  - https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/ (https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/)

- Import libraries.

```
# load library
library(opendatatoronto)
library(dplyr)
```

```
##
## 载入程辑包：'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(ggplot2)
```

Download and Import data.

- Manually download data files from data source.
  - We have 52 files of 5-year data, from 2019 to 2023.
- List of all data files

```
# Define path of dataset
DATA_PATH <- "./source/"


# get the paths for each data file.
FILE_PATH_LIST <- list.files(DATA_PATH, pattern = "\\.csv$", full.names = TRUE)
FILE_PATH_LIST
```

```
##  [1] "./source/2019-Q1.csv"
##  [2] "./source/2019-Q2.csv"
##  [3] "./source/2019-Q3.csv"
##  [4] "./source/2019-Q4.csv"
##  [5] "./source/2020-01.csv"
##  [6] "./source/2020-02.csv"
##  [7] "./source/2020-03.csv"
##  [8] "./source/2020-04.csv"
##  [9] "./source/2020-05.csv"
## [10] "./source/2020-06.csv"
## [11] "./source/2020-07.csv"
## [12] "./source/2020-08.csv"
## [13] "./source/2020-09.csv"
## [14] "./source/2020-10.csv"
## [15] "./source/2020-11.csv"
## [16] "./source/2020-12.csv"
## [17] "./source/Bike share ridership 2021-01.csv"
## [18] "./source/Bike share ridership 2021-02.csv"
## [19] "./source/Bike share ridership 2021-03.csv"
## [20] "./source/Bike share ridership 2021-04.csv"
## [21] "./source/Bike share ridership 2021-05.csv"
## [22] "./source/Bike share ridership 2021-06.csv"
## [23] "./source/Bike share ridership 2021-07.csv"
## [24] "./source/Bike share ridership 2021-08.csv"
## [25] "./source/Bike share ridership 2021-09.csv"
## [26] "./source/Bike share ridership 2021-10.csv"
## [27] "./source/Bike share ridership 2021-11.csv"
## [28] "./source/Bike share ridership 2021-12.csv"
## [29] "./source/Bike share ridership 2022-01.csv"
## [30] "./source/Bike share ridership 2022-02.csv"
## [31] "./source/Bike share ridership 2022-03.csv"
## [32] "./source/Bike share ridership 2022-04.csv"
## [33] "./source/Bike share ridership 2022-05.csv"
## [34] "./source/Bike share ridership 2022-06.csv"
## [35] "./source/Bike share ridership 2022-07.csv"
## [36] "./source/Bike share ridership 2022-08.csv"
## [37] "./source/Bike share ridership 2022-09.csv"
## [38] "./source/Bike share ridership 2022-10.csv"
## [39] "./source/Bike share ridership 2022-11.csv"
## [40] "./source/Bike share ridership 2022-12.csv"
## [41] "./source/Bike share ridership 2023-01.csv"
## [42] "./source/Bike share ridership 2023-02.csv"
## [43] "./source/Bike share ridership 2023-03.csv"
## [44] "./source/Bike share ridership 2023-04.csv"
## [45] "./source/Bike share ridership 2023-05.csv"
## [46] "./source/Bike share ridership 2023-06.csv"
## [47] "./source/Bike share ridership 2023-07.csv"
## [48] "./source/Bike share ridership 2023-08.csv"
## [49] "./source/Bike share ridership 2023-09.csv"
## [50] "./source/Bike share ridership 2023-10.csv"
## [51] "./source/Bike share ridership 2023-11.csv"
## [52] "./source/Bike share ridership 2023-12.csv"
```

- Import data from data files.
    - Check the columns of each csv file for data consisitence.
    - Combine all data into a uniformed dataframe for further anaysis.

```
# # Apply read.csv, a function to import data from csv file, for each file.
# # Get a list of df
# df_list <- lapply(FILE_PATH_LIST, read.csv)
#
#
# # Union all df by row
# raw_df <- do.call(rbind, df_list)
```

- Optional
    - Export the uniformed data

```
# # Path to export
# output_file <- "./data/dataset.csv"
# # export
# write.csv(df, file = output_file, row.names = FALSE)
```

- Test using selective data

```
paths <- c(
  "./source/2019-Q1.csv",
  "./source/2019-Q2.csv",
  "./source/2019-Q3.csv",
  "./source/2019-Q4.csv",
  "./source/2020-01.csv",
  "./source/2020-02.csv",
  "./source/2020-03.csv",
  "./source/2020-04.csv",
  "./source/2020-05.csv",
  "./source/2020-06.csv",
  "./source/2020-07.csv",
  "./source/2020-08.csv",
  "./source/2020-09.csv",
  "./source/2020-10.csv",
  "./source/2020-11.csv",
  "./source/2020-12.csv"
)

df_list <- lapply(paths, read.csv)
raw_df <- do.call(rbind, df_list)
raw_df
```

Trip.Id <dbl>	Trip..Duration <int>	Start.Station.Id <chr>	Start.Time <chr>
4581278	1547	7021	01/01/2019 00:08
4581279	1112	7160	01/01/2019 00:10
4581280	589	7055	01/01/2019 00:15
4581281	259	7012	01/01/2019 00:16
4581282	281	7041	01/01/2019 00:19
4581283	624	7041	01/01/2019 00:26
4581284	604	7041	01/01/2019 00:26
4581285	416	7275	01/01/2019 00:26

Trip.Id	Trip..Duration	Start.Station.Id	Start.Time
<dbl>	<int>	<chr>	<chr>
4581286	192	7071	01/01/2019 00:34
4581287	518	7199	01/01/2019 00:38

1-10 of 10,000 rows | 1-4 of 10 columns Previous **1** 2 3 4 5 6 … 1000 Next

- Explore raw data frame

```
# Data overview
num_row <- nrow(raw_df)             # total rows
column_names <- colnames(raw_df)    # column names
cat("\n\nNumber of rows: ", "\n", num_row)
```

```
##
##
## Number of rows:
##  5350825
```

```
cat("\n\nColumn names: ", "\n", column_names)
```

```
##
##
## Column names:
##  Trip.Id Trip..Duration Start.Station.Id Start.Time Start.Station.Name End.Station.Id End.Time End.S
tation.Name Bike.Id User.Type
```

```
cat("\n\nDisplay the Structure:\n")
```

```
##
##
## Display the Structure:
```

```
str(raw_df)
```

```
## 'data.frame':    5350825 obs. of  10 variables:
##  $ Trip.Id           : num  4581278 4581279 4581280 4581281 4581282 ...
##  $ Trip..Duration    : int  1547 1112 589 259 281 624 604 416 192 518 ...
##  $ Start.Station.Id  : chr  "7021" "7160" "7055" "7012" ...
##  $ Start.Time        : chr  "01/01/2019 00:08" "01/01/2019 00:10" "01/01/2019 00:15" "01/01/2019 00:
16" ...
##  $ Start.Station.Name: chr  "Bay St / Albert St" "King St W / Tecumseth St" "Jarvis St / Carlton St"
"Elizabeth St / Edward St (Bus Terminal)" ...
##  $ End.Station.Id    : chr  "7233" "7051" "7013" "7235" ...
##  $ End.Time          : chr  "01/01/2019 00:33" "01/01/2019 00:29" "01/01/2019 00:25" "01/01/2019 00:
20" ...
##  $ End.Station.Name  : chr  "King / Cowan Ave - SMART" "Wellesley St E / Yonge St (Green P)" "Scott
St / The Esplanade" "Bay St / College St (West Side) - SMART" ...
##  $ Bike.Id           : chr  "1296" "2947" "2293" "283" ...
##  $ User.Type         : chr  "Annual Member" "Annual Member" "Annual Member" "Annual Member" ...
```

```
cat("\n\nDisplay Summaries:\n")
```

```
##
##
## Display Summaries:
```

```
summary(raw_df)
```

```
##     Trip.Id          Trip..Duration     Start.Station.Id    Start.Time
##  Min.   :4.581e+06   Min.   :       0   Length:5350825     Length:5350825
##  1st Qu.:6.090e+06   1st Qu.:     454   Class :character   Class :character
##  Median :7.606e+06   Median :     759   Mode  :character   Mode  :character
##  Mean   :9.738e+06   Mean   :    1114
##  3rd Qu.:9.118e+06   3rd Qu.:    1192
##  Max.   :1.026e+11   Max.   :12403785
##                      NA's   :16
##  Start.Station.Name End.Station.Id       End.Time         End.Station.Name
##  Length:5350825     Length:5350825     Length:5350825     Length:5350825
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    Bike.Id           User.Type
##  Length:5350825     Length:5350825
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

Data Processing

- Handle NA value that exist in the raw data

```
# Remove rows with any null values
proc_df <- raw_df[complete.cases(raw_df), ]
```

- Divide time into year, month, date, hour, and minute.

```
# Divide "Start.Time" into columns
proc_df$Start.Time <- as.POSIXct(proc_df$Start.Time, format = "%m/%d/%Y %H:%M")
proc_df$Start.Year <- as.factor(format(proc_df$Start.Time, "%Y"))
proc_df$Start.Month <- as.factor(format(proc_df$Start.Time, "%m"))
proc_df$Start.Date <- as.factor(format(proc_df$Start.Time, "%d"))
proc_df$Start.Hours <- as.factor(format(proc_df$Start.Time, "%H"))
proc_df$Start.Minutes <- as.factor(format(proc_df$Start.Time, "%M"))

# Divide "End" into columns
proc_df$End.Time <- as.POSIXct(proc_df$End.Time, format = "%m/%d/%Y %H:%M")
proc_df$End.Year <- as.factor(format(proc_df$End.Time, "%Y"))
proc_df$End.Month <- as.factor(format(proc_df$End.Time, "%m"))
proc_df$End.Date <- as.factor(format(proc_df$End.Time, "%d"))
proc_df$End.Hours <- as.factor(format(proc_df$End.Time, "%H"))
proc_df$End.Minutes <- as.factor(format(proc_df$End.Time, "%M"))

# factor user.type
proc_df$User.Type <- as.factor(proc_df$User.Type)

# Drop Start.Time and End.Time
proc_df <- proc_df %>% select(-Start.Time, -End.Time)
```

- Check NA value again, in case of any possible values generated during the data processing.

```
is_miss <- any(is.na(proc_df))

# if the processed_df contains missing value, drop the rows with missing values and assign to df
if (is_miss) {
  df <- proc_df[complete.cases(proc_df), ]
# otherwise, df = proc_df
}else{
  df <- proc_df
}

is_miss <- any(is.na(df))
cat("df has missing value? ", is_miss)  # output result
```

```
## df has missing value?  FALSE
```

- Data Overview afater data processing.

```
# Data overview after data processing
num_row <- nrow(df)             # total rows
column_names <- colnames(df)    # column names
cat("\n\nNumber of rows: ", "\n", num_row)
```

```
##
##
## Number of rows:
##  5350560
```

```
cat("\n\nColumn names: ", "\n", column_names)
```

```
##
##
## Column names:
##  Trip.Id Trip..Duration Start.Station.Id Start.Station.Name End.Station.Id End.Station.Name Bike.Id
User.Type Start.Year Start.Month Start.Date Start.Hours Start.Minutes End.Year End.Month End.Date End.H
ours End.Minutes
```

```
cat("\n\nDisplay the Structure:\n")
```

```
##
##
## Display the Structure:
```

```
str(df)
```

```
## 'data.frame':    5350560 obs. of  18 variables:
##  $ Trip.Id           : num  4581278 4581279 4581280 4581281 4581282 ...
##  $ Trip..Duration    : int  1547 1112 589 259 281 624 604 416 192 518 ...
##  $ Start.Station.Id  : chr  "7021" "7160" "7055" "7012" ...
##  $ Start.Station.Name: chr  "Bay St / Albert St" "King St W / Tecumseth St" "Jarvis St / Carlton St"
"Elizabeth St / Edward St (Bus Terminal)" ...
##  $ End.Station.Id    : chr  "7233" "7051" "7013" "7235" ...
##  $ End.Station.Name  : chr  "King / Cowan Ave - SMART" "Wellesley St E / Yonge St (Green P)" "Scott
St / The Esplanade" "Bay St / College St (West Side) - SMART" ...
##  $ Bike.Id           : chr  "1296" "2947" "2293" "283" ...
##  $ User.Type         : Factor w/ 3 levels "","Annual Member",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Start.Year        : Factor w/ 2 levels "2019","2020": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Start.Month       : Factor w/ 12 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Start.Date        : Factor w/ 31 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Start.Hours       : Factor w/ 24 levels "00","01","02",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Start.Minutes     : Factor w/ 60 levels "00","01","02",..: 9 11 16 17 20 27 27 27 35 39 ...
##  $ End.Year          : Factor w/ 3 levels "2019","2020",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ End.Month         : Factor w/ 12 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ End.Date          : Factor w/ 31 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ End.Hours         : Factor w/ 24 levels "00","01","02",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ End.Minutes       : Factor w/ 60 levels "00","01","02",..: 34 30 26 21 25 37 37 34 38 47 ...
```

```
cat("\n\nDisplay Summaries:\n")
```

```
##
##
## Display Summaries:
```

```
summary(df)
```

```
##     Trip.Id         Trip..Duration     Start.Station.Id   Start.Station.Name
##  Min.   : 4581278   Min.   :       0   Length:5350560     Length:5350560
##  1st Qu.: 6090112   1st Qu.:     454   Class :character   Class :character
##  Median : 7606090   Median :     759   Mode  :character   Mode  :character
##  Mean   : 7606151   Mean   :    1113
##  3rd Qu.: 9117443   3rd Qu.:    1192
##  Max.   :10644217   Max.   :12403785
##
##  End.Station.Id     End.Station.Name     Bike.Id
##  Length:5350560     Length:5350560     Length:5350560
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##          User.Type       Start.Year      Start.Month        Start.Date
##                :      0   2019:2439501   08     : 911862   23     : 197751
##  Annual Member:3731484   2020:2911059   07     : 841120   07     : 190216
##  Casual Member:1619076                  09     : 771382   21     : 188167
##                                         06     : 662389   08     : 185912
##                                         10     : 526246   05     : 185578
##                                         05     : 410837   24     : 185403
##                                         (Other):1226724   (Other):4217533
##   Start.Hours      Start.Minutes     End.Year         End.Month
##  17     : 579916   39     :  91557   2019:2439486   08     : 911890
##  18     : 494087   44     :  91322   2020:2911059   07     : 841138
##  16     : 445840   45     :  91258   2021:     15   09     : 771508
##  19     : 395269   48     :  91237                  06     : 662249
##  15     : 360974   09     :  91219                  10     : 526268
##  14     : 327624   43     :  91196                  05     : 410745
##  (Other):2746850   (Other):4802771                  (Other):1226762
##     End.Date         End.Hours        End.Minutes
##  23     : 197758   17     : 562914   55     :  95127
##  07     : 190131   18     : 515535   57     :  94398
##  21     : 188129   16     : 422382   56     :  93810
##  08     : 185999   19     : 418260   53     :  93759
##  05     : 185596   15     : 349778   51     :  93397
##  24     : 185406   20     : 325821   54     :  93247
##  (Other):4217541   (Other):2755870   (Other):4786822
```

```
df
```

	Trip.Id <dbl>	Trip..Duration <int>	Start.Station.Id <chr>	Start.Station.Name <chr>
1	4581278	1547	7021	Bay St / Albert St
2	4581279	1112	7160	King St W / Tecumseth St
3	4581280	589	7055	Jarvis St / Carlton St
4	4581281	259	7012	Elizabeth St / Edward St (Bus Terminal)
5	4581282	281	7041	Edward St / Yonge St
6	4581283	624	7041	Edward St / Yonge St
7	4581284	604	7041	Edward St / Yonge St

	Trip.Id <dbl>	Trip..Duration <int>	Start.Station.Id <chr>	Start.Station.Name <chr>
8	4581285	416	7275	Queen St W / James St
9	4581286	192	7071	161 Bleecker St (South of Wellesley)
10	4581287	518	7199	College St / Markham St

```
head(df, 10)
```

	Trip.Id <dbl>	Trip..Duration <int>	Start.Station.Id <chr>	Start.Station.Name <chr>	End.Sta <chr>
1	4581278	1547	7021	Bay St / Albert St	7233
2	4581279	1112	7160	King St W / Tecumseth St	7051
3	4581280	589	7055	Jarvis St / Carlton St	7013
4	4581281	259	7012	Elizabeth St / Edward St (Bus Terminal)	7235
5	4581282	281	7041	Edward St / Yonge St	7257
6	4581283	624	7041	Edward St / Yonge St	7031
7	4581284	604	7041	Edward St / Yonge St	7031
8	4581285	416	7275	Queen St W / James St	7041
9	4581286	192	7071	161 Bleecker St (South of Wellesley)	7311
10	4581287	518	7199	College St / Markham St	7252

1-10 of 10 rows | 1-6 of 19 columns

Temporal analysis

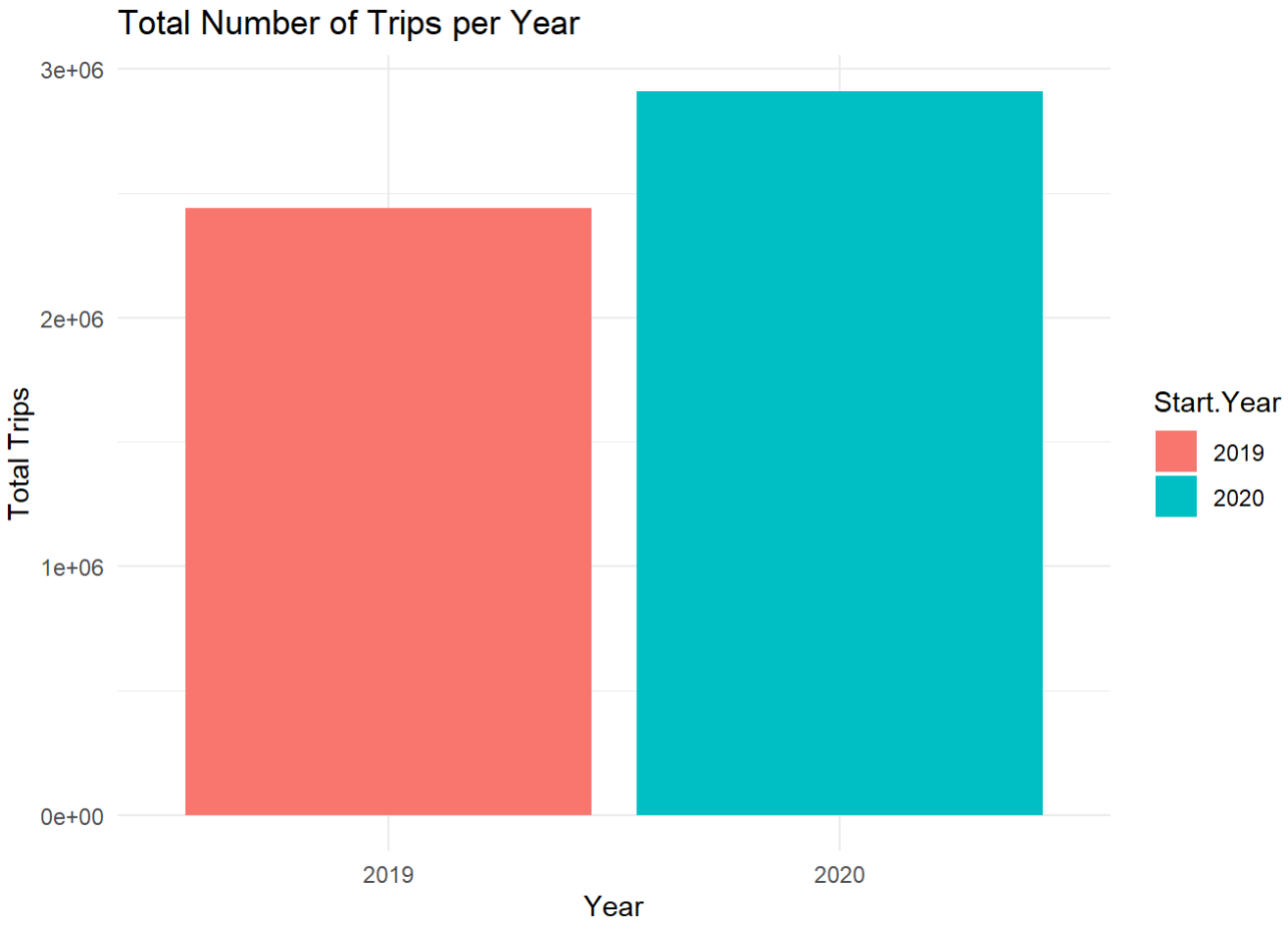
Yearly Trip Trends

```
total_trip_by_year <- df %>%
  group_by(Start.Year) %>%
  summarize(Total_Trips = n())
total_trip_by_year
```

Start.Year <fct>	Total_Trips <int>
2019	2439501
2020	2911059

2 rows

```
# Create a bar plot
ggplot(
  data = total_trip_by_year,
  mapping = aes(
      x = Start.Year,
      y = Total_Trips,
      fill = Start.Year
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of Trips per Year",
    x = "Year",
    y = "Total Trips"
  ) +
  theme_minimal()
```



Monthly Total Trip Distribution

- Comparing monthly trip to unveil patterns over the years.

```
monthly_trip_by_year <- df %>%
  group_by(Start.Year, Start.Month) %>%
  summarize(Trips = n())
```

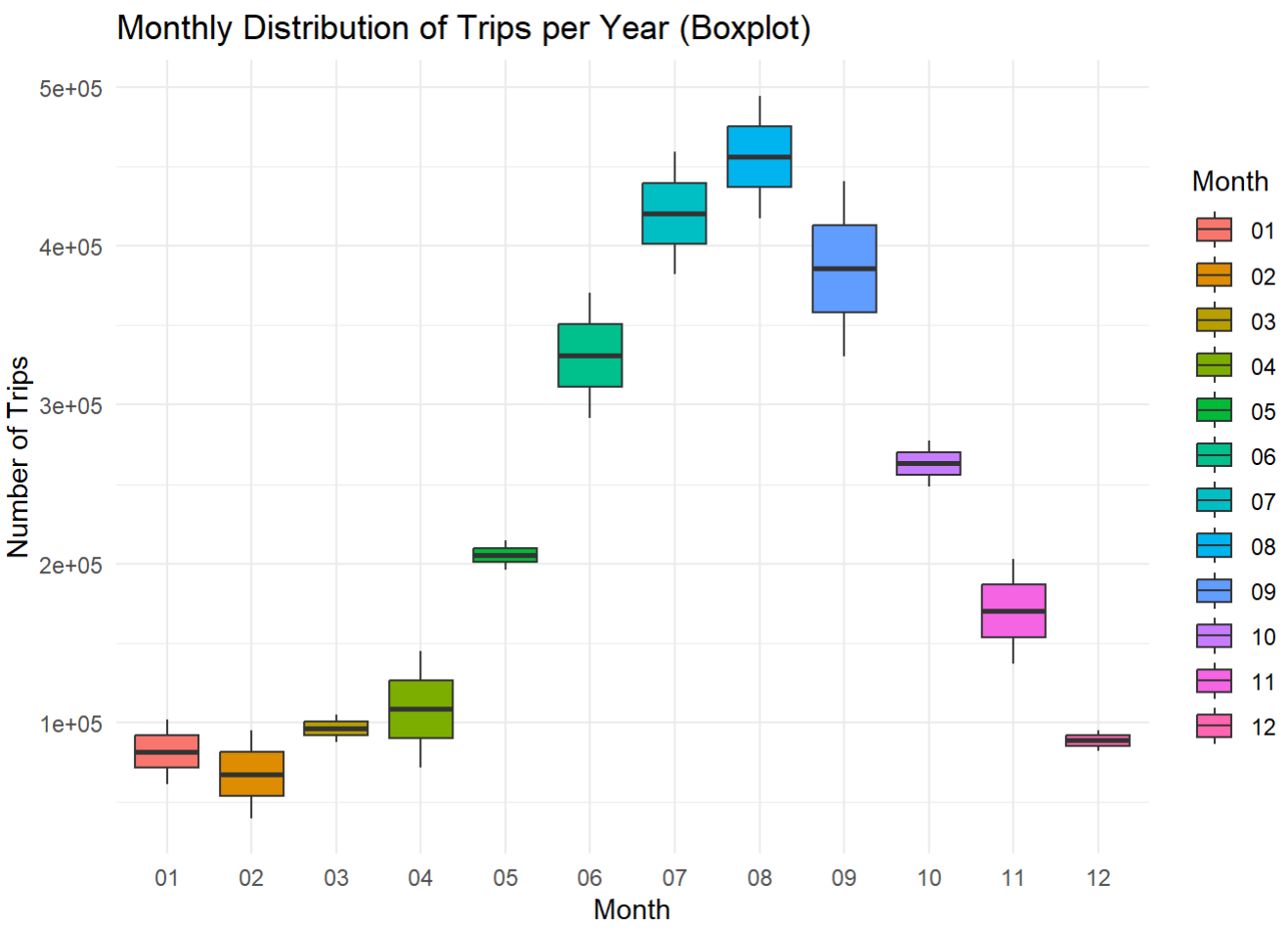
```
## `summarise()` has grouped output by 'Start.Year'. You can override using the
## `.groups` argument.
```

```
monthly_trip_by_year
```

Start.Year <fct>	Start.Month <fct>	Trips <int>
2019	01	61461
2019	02	40055
2019	03	87540
2019	04	145150
2019	05	214613
2019	06	291918
2019	07	382236
2019	08	417394
2019	09	330720
2019	10	248656

1-10 of 24 rows Previous **1** 2 3 Next

```
ggplot(
  data = monthly_trip_by_year,
  mapping = aes(
    x = Start.Month,
    y = Trips,
    fill = Start.Month
    )
  ) +
  geom_boxplot() +
  labs(
    title = "Monthly Distribution of Trips per Year (Boxplot)",
    x = "Month",
    y = "Number of Trips",
    fill = "Month"
  ) +
  theme_minimal()
```



Hourly Total Trip Distribution

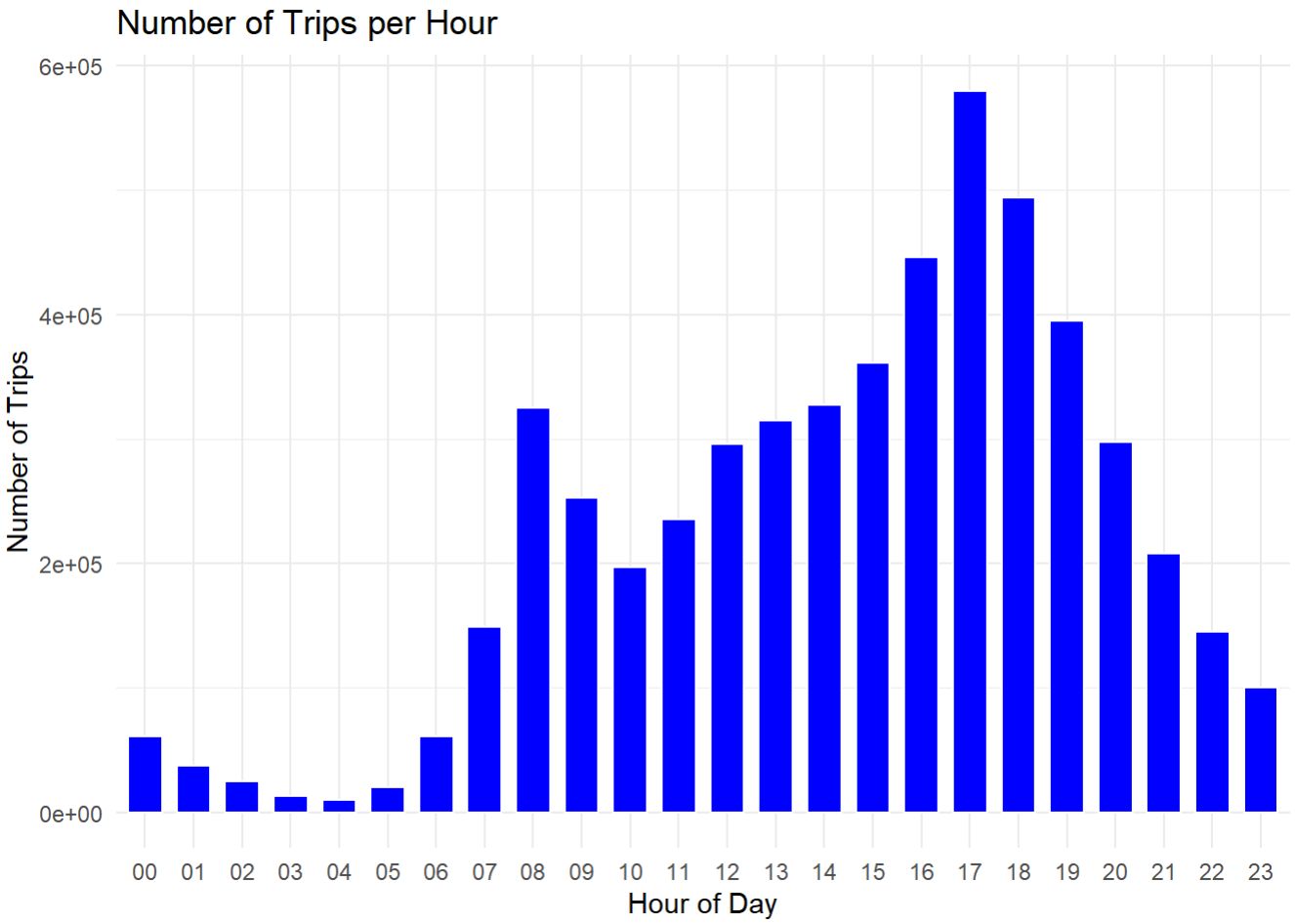
- Exploring Patterns Throughout the Day

```
trips_per_hour <- df %>%
  group_by(Start.Hours) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips))
trips_per_hour
```

Start.Hours <fct>	**Total_Trips** <int>
17	579916
18	494087
16	445840
19	395269
15	360974
14	327624
08	325202
13	314638
20	297264
12	295533

1-10 of 24 rows Previous **1** 2 3 Next

```
# Plot the data
ggplot(
    data = trips_per_hour,
    mapping = aes(
    x = Start.Hours,
    y = Total_Trips
      )
  ) +
  geom_bar(
    stat = "identity",
    fill = "blue",
    color = "white",
    width = 0.7
    ) +
  labs(
    title = "Number of Trips per Hour",
    x = "Hour of Day",
    y = "Number of Trips") +
  theme_minimal()
```



Monthly Trip Distribution Across Months

- Seeking patterns and variations among months

```
trips_per_month_hour <- df %>%
  group_by(
    Month = Start.Month,
    Hour = Start.Hours) %>%
  summarize(Trip = n())
```

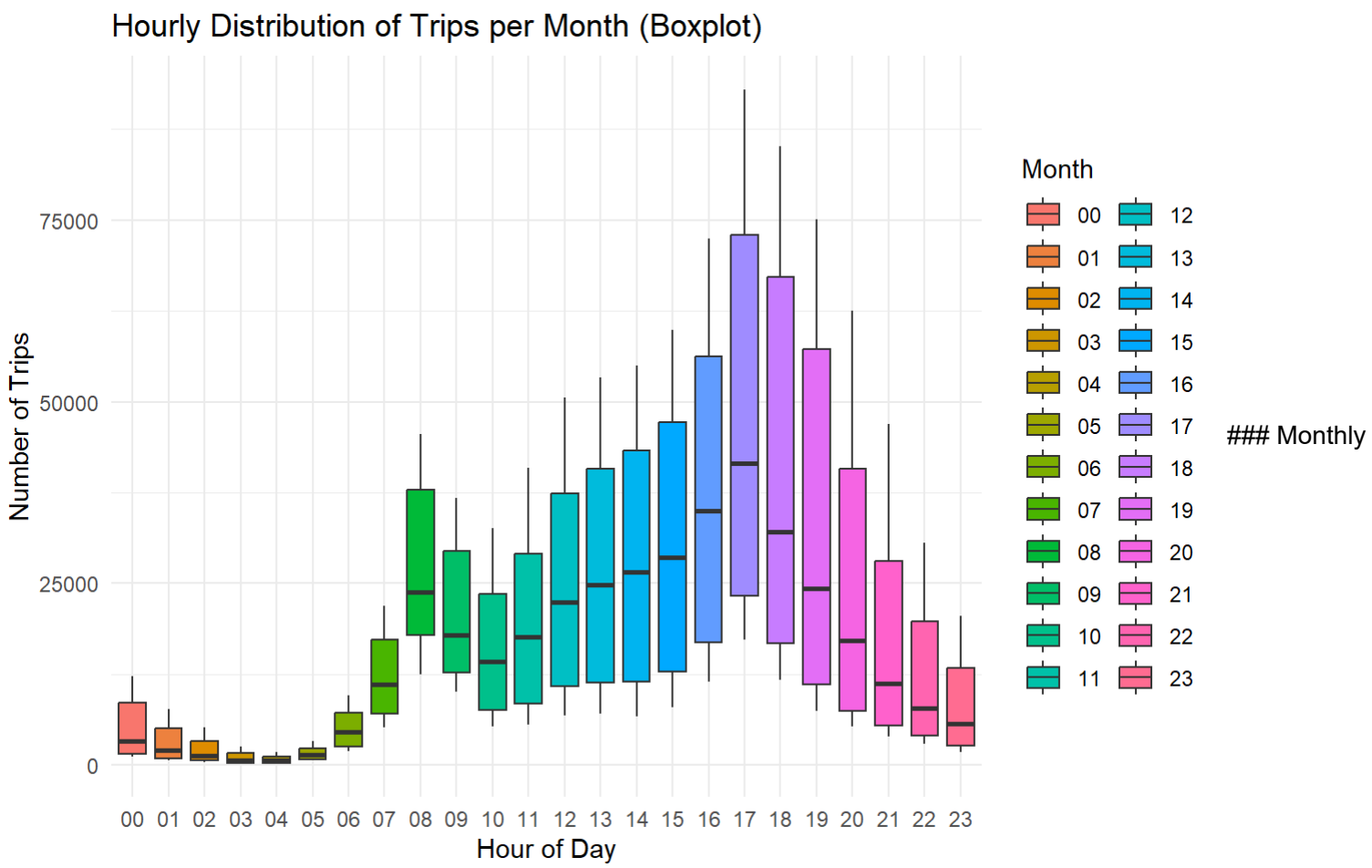
```
## `summarise()` has grouped output by 'Month'. You can override using the
## `.groups` argument.
```

```
trips_per_month_hour
```

Month <fct>	Hour <fct>	Trip <int>
01	00	1137
01	01	796
01	02	564
01	03	286
01	04	270
01	05	772
01	06	2512
01	07	6964
01	08	19157
01	09	13858

1-10 of 288 rows Previous **1** 2 3 4 5 6 … 29 Next

```
ggplot(
  trips_per_month_hour,
  aes(
    x = Hour,
    y = Trip,
    fill = Hour
    )
  ) +
  geom_boxplot() +
  labs(
    title = "Hourly Distribution of Trips per Month (Boxplot)",
    x = "Hour of Day",
    y = "Number of Trips",
    fill = "Month") +
  theme_minimal()
```



Patterns of Rush Hour

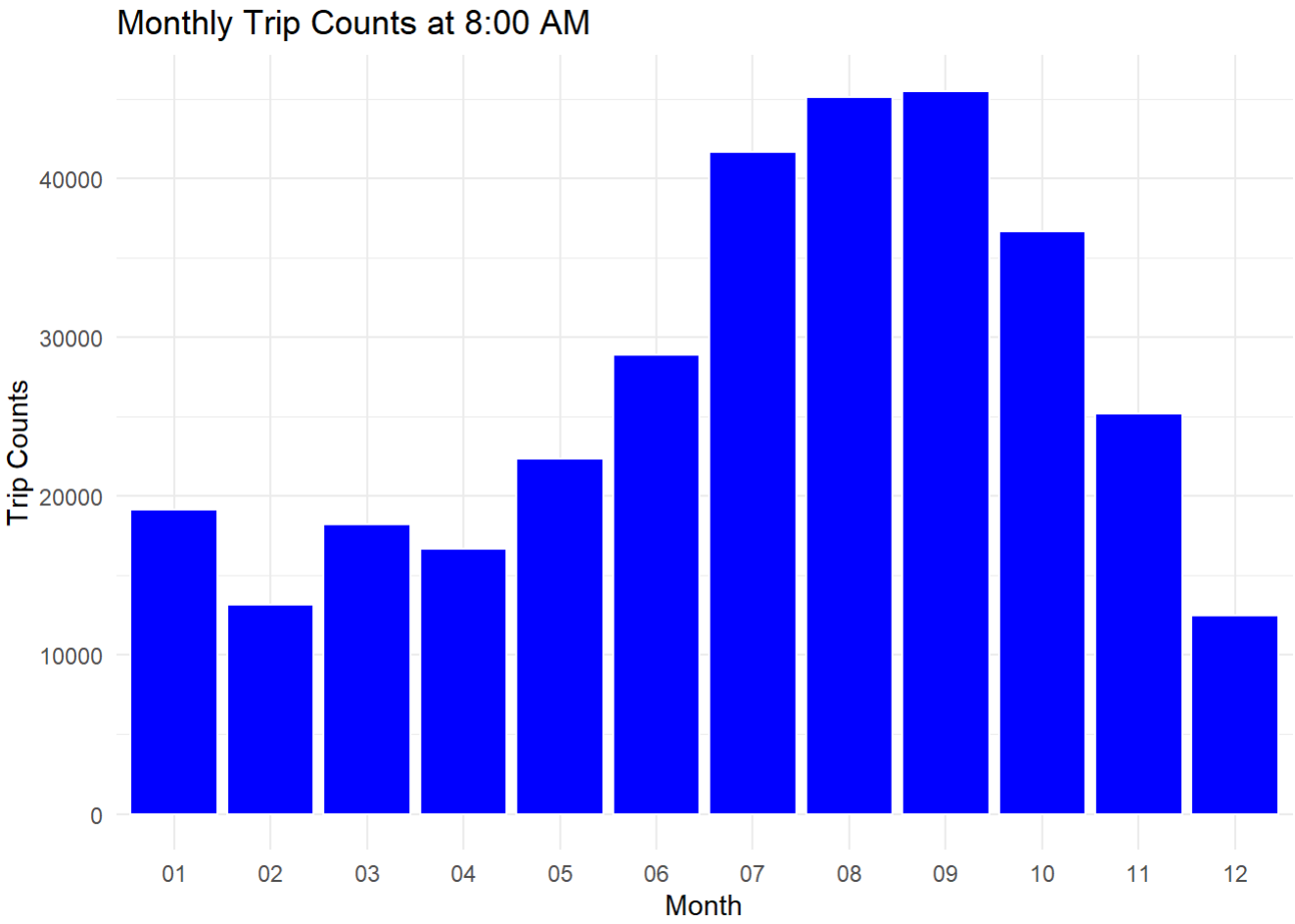
```
rush_hour_data <- df[df$Start.Hours == "08", ]

rush_hour_by_month <- rush_hour_data %>%
  group_by(Month = Start.Month) %>%
  summarize(Trip = n())

rush_hour_by_month
```

Month <fct>	**Trip** <int>
01	19157
02	13199
03	18205
04	16717
05	22343
06	28912
07	41659
08	45127
09	45516
10	36649

```
ggplot(
  data = rush_hour_by_month,
  mapping = aes(
    x = Month,
    y = Trip
    )
  ) +
  geom_bar(
    stat = "identity",
    fill = "blue",
    color = "white"
  ) +
  labs(
    title = "Monthly Trip Counts at 8:00 AM",
    x = "Month",
    y = "Trip Counts") +
  theme_minimal()
```



Duration Analysis

Mean Duration Yearly Trend

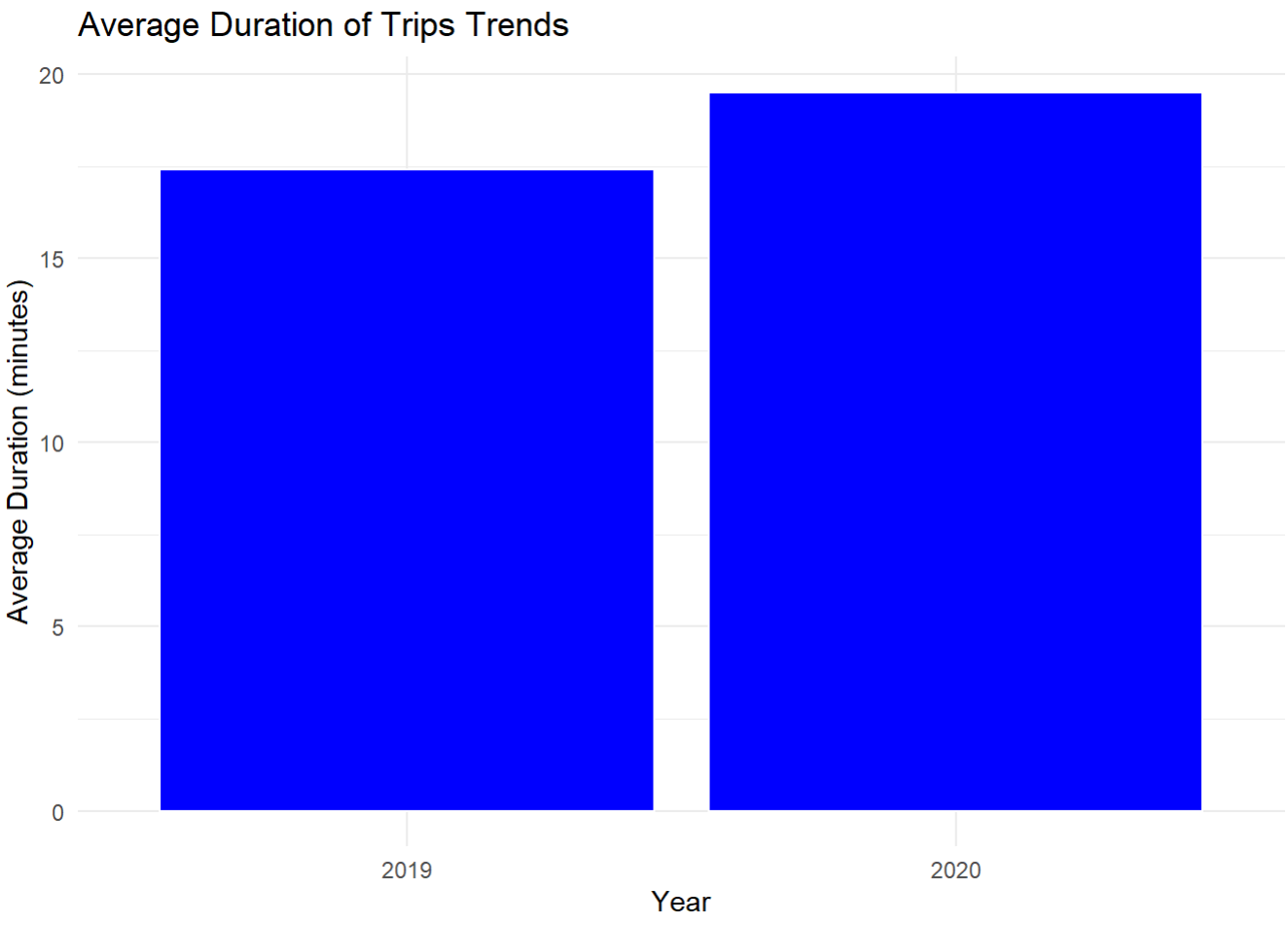
```
mean_duration_by_year <- df %>%
  group_by(Year = Start.Year) %>%
  summarize(Mean_Duration = mean(Trip..Duration) / 60)  # Duration in minutte

mean_duration_by_year
```

Year <fct>	Mean_Duration <dbl>
2019	17.41422
2020	19.51386

2 rows

```
ggplot(
  data = mean_duration_by_year,
  mapping = aes(
    x = Year,
    y = Mean_Duration
    )
  ) +
  geom_bar(
    stat = "identity",
    fill = "blue",
    color = "white"
  ) +
  labs(
    title = "Average Duration of Trips Trends",
    x = "Year",
    y = "Average Duration (minutes)") +
  theme_minimal()
```



Monthly Mean Duration Distribution acorss months

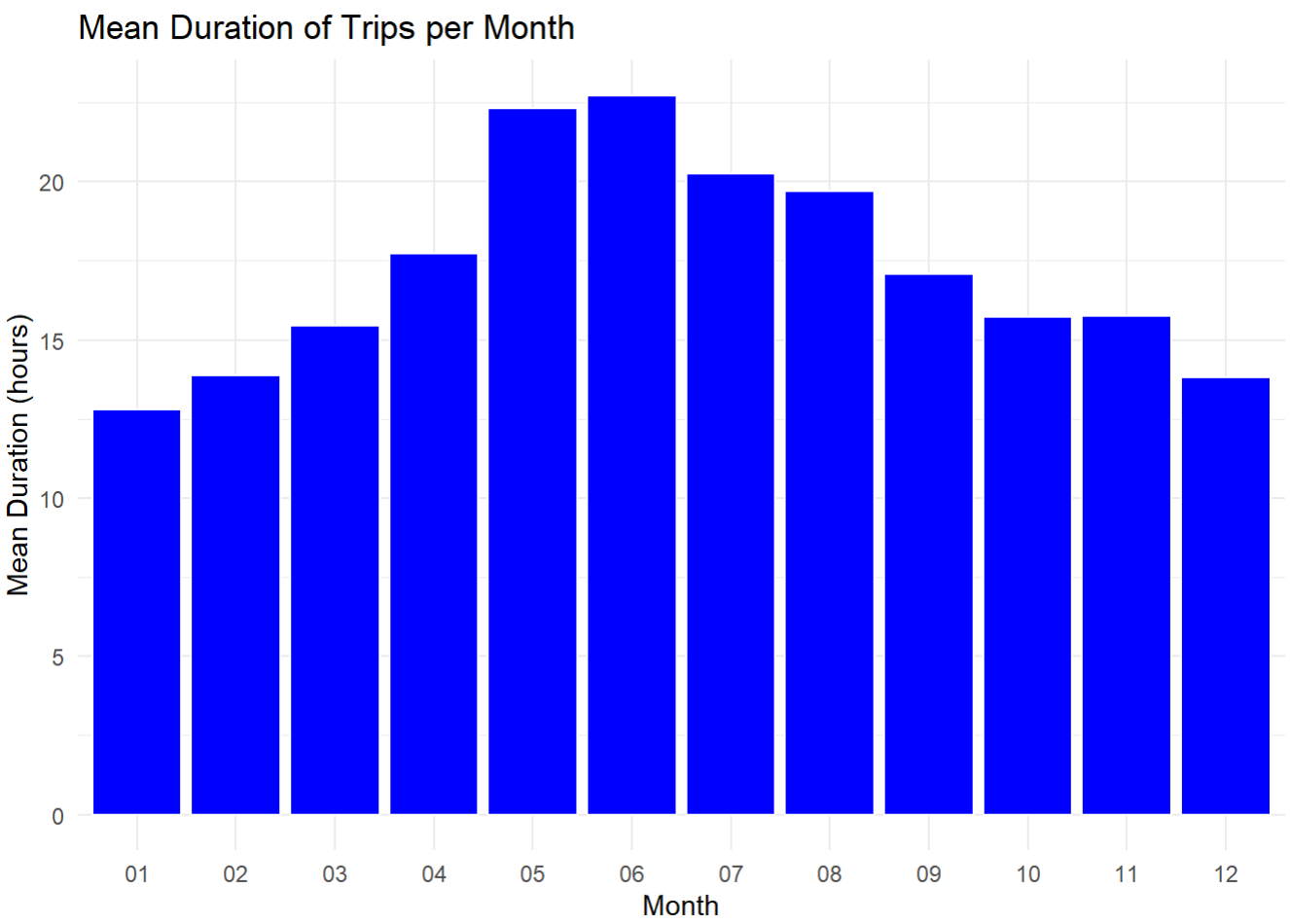
```
mean_duration_by_month <- df %>%
  group_by(Start.Month) %>%
  summarize(Average_Duration = mean(Trip..Duration) / 60)  # Convert mean to hours

mean_duration_by_month
```

Start.Month <fct>	Average_Duration <dbl>
01	12.80625
02	13.87830
03	15.43512
04	17.73152
05	22.31656
06	22.73143
07	20.24864
08	19.68579
09	17.09506
10	15.73864

1-10 of 12 rows Previous **1** 2 Next

```
ggplot(data = mean_duration_by_month, aes(x = Start.Month, y = Average_Duration)) +
  geom_bar(stat = "identity", fill = "blue", color = "white") +
  labs(title = "Mean Duration of Trips per Month",
       x = "Month",
       y = "Mean Duration (hours)") +
  theme_minimal()
```




```
monthly_mean_duration_by_year <- df %>%
  group_by(
    Year = Start.Year,
    Month = Start.Month) %>%
  summarize(Mean_Duration = mean(Trip..Duration) / 60)  # Convert mean to hours
```

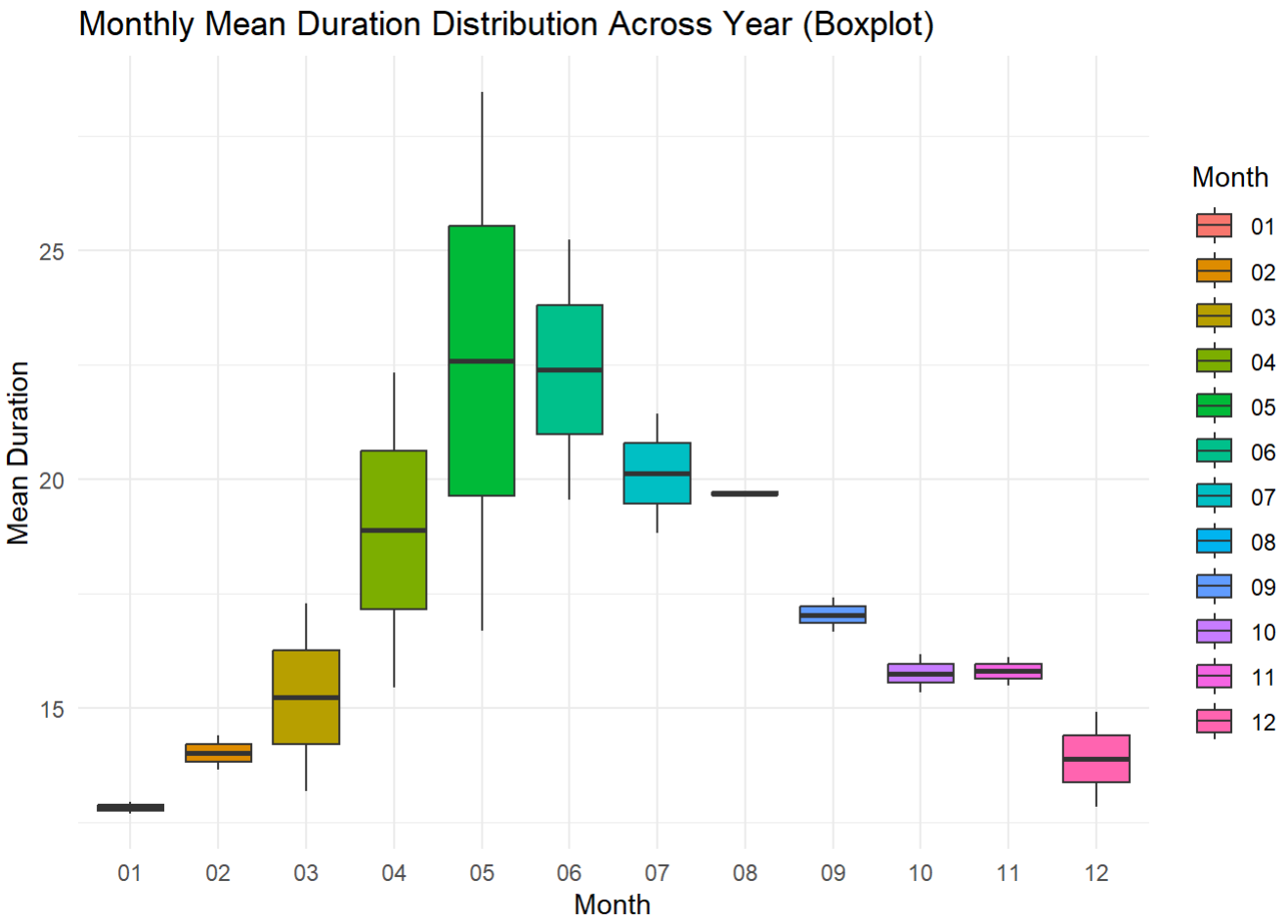
```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
monthly_mean_duration_by_year
```

Year <fct>	Month <fct>	Mean_Duration <dbl>
2019	01	12.96360
2019	02	14.41481
2019	03	13.20129
2019	04	15.45447
2019	05	16.69236
2019	06	19.55907
2019	07	18.81917
2019	08	19.74806
2019	09	16.67274
2019	10	16.17545

1-10 of 24 rows Previous **1** 2 3 Next

```
ggplot(
  data = monthly_mean_duration_by_year,
  mapping = aes(
    x = Month,
    y = Mean_Duration,
    fill = Month
    )
  ) +
  geom_boxplot() +
  labs(
    title = "Monthly Mean Duration Distribution Across Year (Boxplot)",
    x = "Month",
    y = "Mean Duration",
    fill = "Month"
  ) +
  theme_minimal()
```



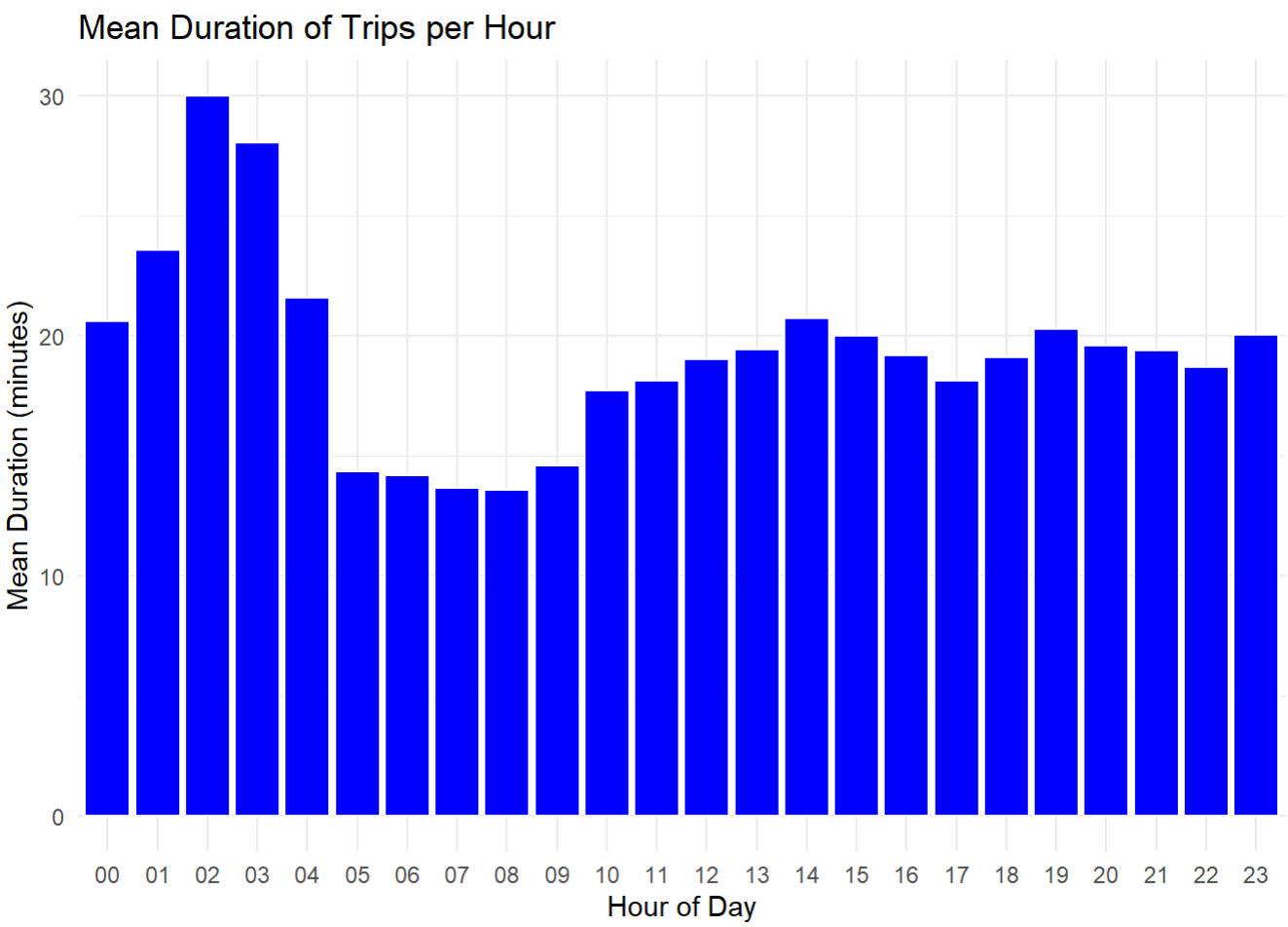
Hourly Mean Duration Distribution

```
mean_duration_by_hour <- df %>%
  group_by(Hour = Start.Hours) %>%
  summarize(Mean_Duration = mean(Trip..Duration) / 60)  # Convert mean to hours
mean_duration_by_hour
```

Hour <fct>	**Mean_Duration** <dbl>
00	20.57960

Hour <fct>	Mean_Duration <dbl>
01	23.57765
02	30.00683
03	28.05646
04	21.56942
05	14.32757
06	14.17900
07	13.64556
08	13.56984
09	14.59475


```
ggplot(
  data = mean_duration_by_hour,
  mapping = aes(
    x = Hour,
    y = Mean_Duration
    )
  ) +
  geom_bar(
    stat = "identity",
    fill = "blue",
    color = "white"
  ) +
  labs(
    title = "Mean Duration of Trips per Hour",
    x = "Hour of Day",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()
```



Hourly Mean Duration Distibution Pattern

```
hourly_mean_duration_by_month <- df %>%
  group_by(
    Month = Start.Month,
    Hour = Start.Hours
    ) %>%
  summarize(Mean_Duration = mean(Trip..Duration) / 60)  # Convert mean to hours
```

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

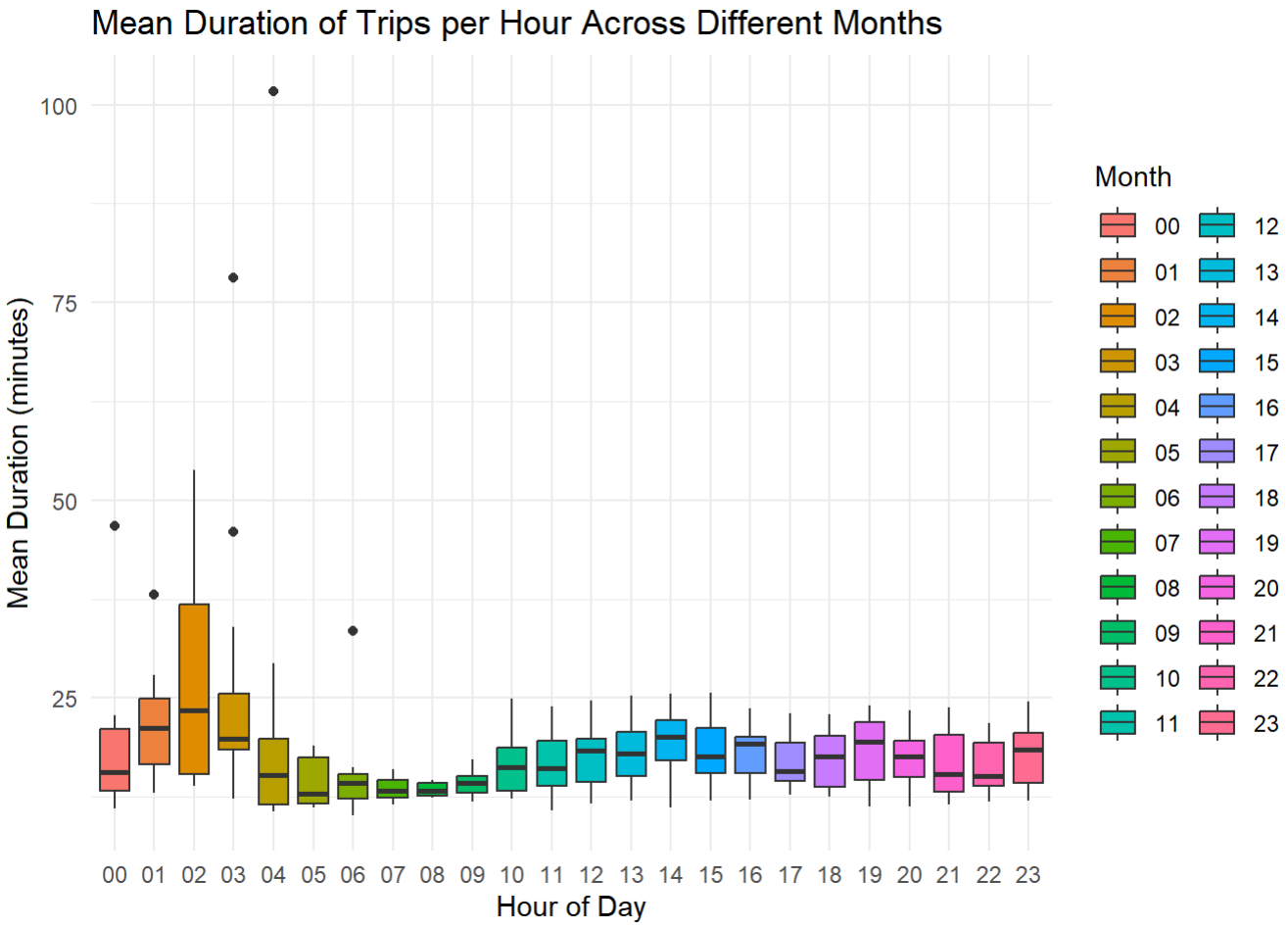
```
hourly_mean_duration_by_month
```

Month <fct>	Hour <fct>	Mean_Duration <dbl>
01	00	13.22237
01	01	14.47372
01	02	50.78118
01	03	12.29324
01	04	10.64346
01	05	17.74298
01	06	15.42632
01	07	12.66557

Month <fct>	Hour <fct>	Mean_Duration <dbl>
01	08	12.55800
01	09	13.18494

1-10 of 288 rows Previous **1** 2 3 4 5 6 ... 29 Next

```
ggplot(
  data = hourly_mean_duration_by_month,
  mapping = aes(
    x = Hour,
    y = Mean_Duration,
    fill = Hour
    )
  ) +
  geom_boxplot() +
  labs(
    title = "Mean Duration of Trips per Hour Across Different Months",
    x = "Hour of Day",
    y = "Mean Duration (minutes)",
    fill = "Month") +
  theme_minimal()
```



Geolocation Analysis

User type analysis