# Project in R – Uber Data Analysis Project

Welcome to part 2 of R and Data Science Projects designed by DataFlair. In our series of R projects, we are trying to use all the concepts related to Machine learning, AI and Data Science.

We recommend you to follow all the steps given in the projects so that you will master the technology rapidly. In today’s R project, we will analyze the ***Uber Pickups in New York City dataset***. This is more of a data visualization project that will guide you towards using the ggplot2 library for understanding the data and for developing an intuition for understanding the customers who avail the trips. So, before we start, take a quick revision to [***data visualization concepts***](https://data-flair.training/blogs/data-visualization-in-r/).

## R Data Science Project – Uber Data Analysis

Talking about our Uber data analysis project, data storytelling is an important component of [***Machine Learning***](https://techvidvan.com/tutorials/machine-learning-tutorial/) through which companies are able to understand the background of various operations. With the help of visualization, companies can avail the benefit of understanding the complex data and gain insights that would help them to craft decisions. You will learn how to implement the ggplot2 on the Uber Pickups dataset and at the end, master the art of data visualization in R.

***You can download the dataset utilized in this project here –*** [**Uber Dataset**](https://drive.google.com/file/d/1emopjfEkTt59jJoBH9L9bSdmlDC4AR87/view)

### 1. Importing the Essential Packages

In the first step of our R project, we will import the essential packages that we will use in this uber data analysis project. Some of the ***important libraries of R*** that we will use are –

* ggplot2

This is the backbone of this project. ggplot2 is the most popular data visualization library that is most widely used for creating aesthetic visualization plots.

* ggthemes

This is more of an add-on to our main ggplot2 library. With this, we can create better create extra themes and scales with the mainstream ggplot2 package.

* lubridate

Our dataset involves various time-frames. In order to understand our data in separate time categories, we will make use of the lubridate package.

* dplyr

This package is the lingua franca of [***data manipulation in R***](https://data-flair.training/blogs/manipulating-and-processing-data-in-r/).

* tidyr

This package will help you to tidy your data. The basic principle of tidyr is to tidy the columns where each variable is present in a column, each observation is represented by a row and each value depicts a cell.

* DT

With the help of this package, we will be able to interface with the [***JavaScript***](https://data-flair.training/blogs/javascript-tutorials-home/) Library called – Datatables.

* scales

With the help of graphical scales, we can automatically map the data to the correct scales with well-placed axes and legends.

library(ggplot2)

library(ggthemes)

library(lubridate)

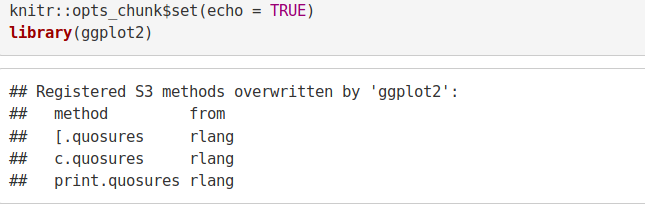
library(dplyr)

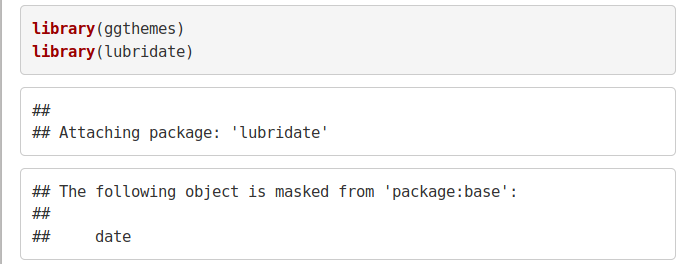
library(tidyr)

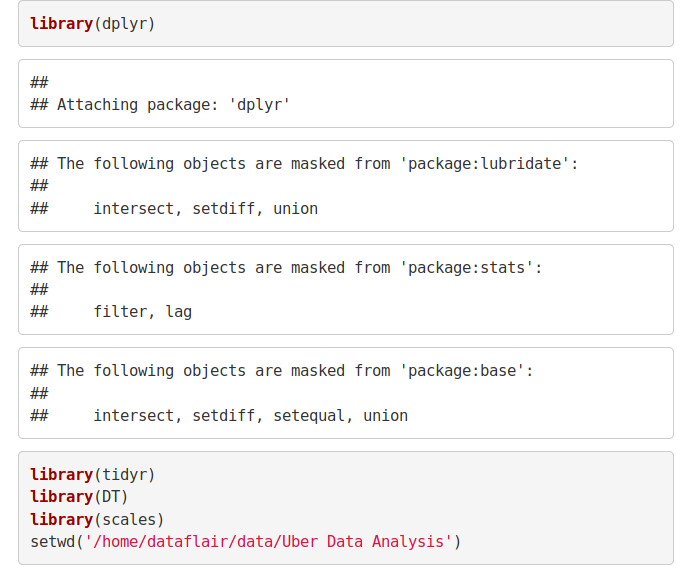
library(DT)

library(scales)

**Input Screenshot 1:**

  
**Input Screenshot 2:**

  
**Input Screenshot 3:**



### 2. Creating vector of colors to be implemented in our plots

In this step of data science project, we will create a vector of our colors that will be included in our plotting functions. You can also select your own set of colors.

**Code:**

colors = c(""#CC1011", "#665555", "#05a399", "#cfcaca", "#f5e840", "#0683c9", "#e075b0"")

**Input Screenshot 4:**

create vector of colors

### 3. Reading the Data into their designated variables

Now, we will read several csv files that contain the data from April 2014 to September 2014. We will store these in corresponding data frames like apr\_data, may\_data, etc. After we have read the files, we will combine all of this data into a single dataframe called ‘data\_2014’.

***To master this R Uber data analysis project, you need to know everything related to*** [***data frames in R***](https://data-flair.training/blogs/r-data-frame-introduction-operations/)

Then, in the next step, we will perform the appropriate formatting of Date.Time column. Then, we will proceed to create factors of time objects like day, month, year etc.

**Code:**

apr\_data <- read.csv("uber-raw-data-apr14.csv")

may\_data <- read.csv("uber-raw-data-may14.csv")

jun\_data <- read.csv("uber-raw-data-jun14.csv")

jul\_data <- read.csv("uber-raw-data-jul14.csv")

aug\_data <- read.csv("uber-raw-data-aug14.csv")

sep\_data <- read.csv("uber-raw-data-sep14.csv")

data\_2014 <- rbind(apr\_data,may\_data, jun\_data, jul\_data, aug\_data, sep\_data)

data\_2014$Date.Time <- as.POSIXct(data\_2014$Date.Time, format = "%m/%d/%Y %H:%M:%S")

data\_2014$Time <- format(as.POSIXct(data\_2014$Date.Time, format = "%m/%d/%Y %H:%M:%S"), format="%H:%M:%S")

data\_2014$Date.Time <- ymd\_hms(data\_2014$Date.Time)

data\_2014$day <- factor(day(data\_2014$Date.Time))

data\_2014$month <- factor(month(data\_2014$Date.Time, label = TRUE))

data\_2014$year <- factor(year(data\_2014$Date.Time))

data\_2014$dayofweek <- factor(wday(data\_2014$Date.Time, label = TRUE))

**Input Screenshot 5:**



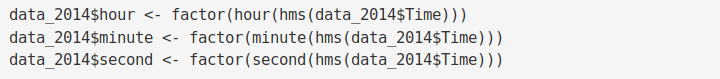
**Code:**

data\_2014$hour <- factor(hour(hms(data\_2014$Time)))

data\_2014$minute <- factor(minute(hms(data\_2014$Time)))

data\_2014$second <- factor(second(hms(data\_2014$Time)))

**Input Screenshot 6:**



### Plotting the trips by the hours in a day

In the next step or R project, we will use the ggplot function to plot the number of trips that the passengers had made in a day. We will also use dplyr to aggregate our data. In the resulting visualizations, we can understand how the number of passengers fares throughout the day. We observe that the number of trips are higher in the evening around 5:00 and 6:00 PM.

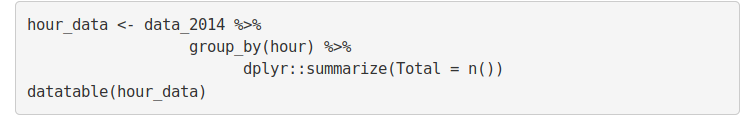
hour\_data <- data\_2014 %>%

group\_by(hour) %>%

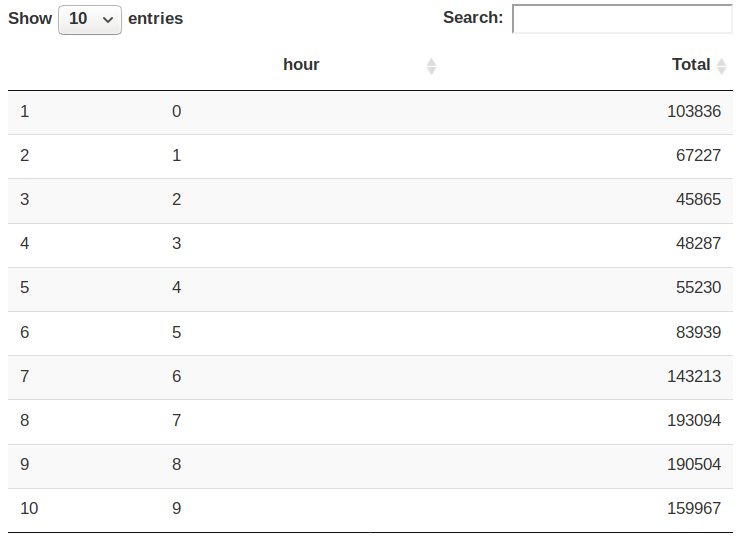
dplyr::summarize(Total = n())

datatable(hour\_data)

**Input Screenshot 7:**



**Output Screenshot:**



**Code:**

ggplot(hour\_data, aes(hour, Total)) +

geom\_bar( stat = "identity", fill = "steelblue", color = "red") +

ggtitle("Trips Every Hour") +

theme(legend.position = "none") +

scale\_y\_continuous(labels = comma)

month\_hour <- data\_2014 %>%

group\_by(month, hour) %>%

dplyr::summarize(Total = n())

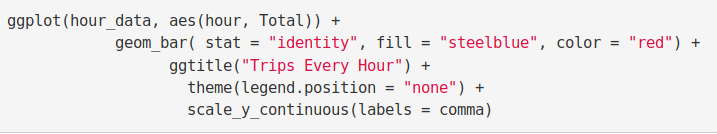
ggplot(month\_hour, aes(hour, Total, fill = month)) +

geom\_bar( stat = "identity") +

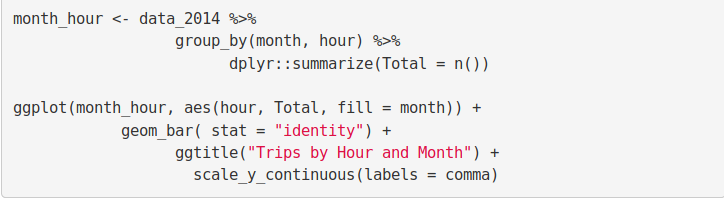
ggtitle("Trips by Hour and Month") +

scale\_y\_continuous(labels = comma)

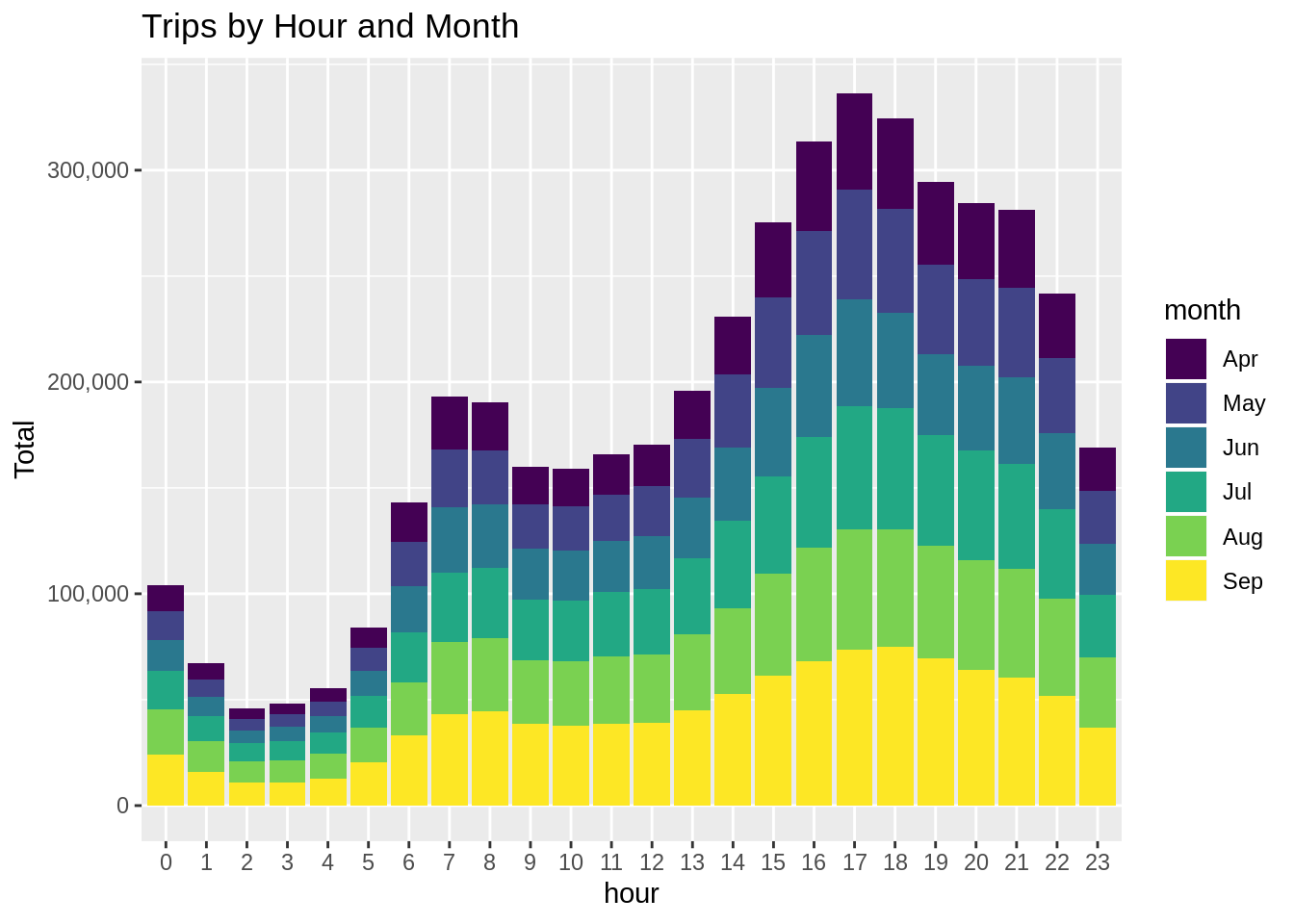
**Input Screenshot 8:**



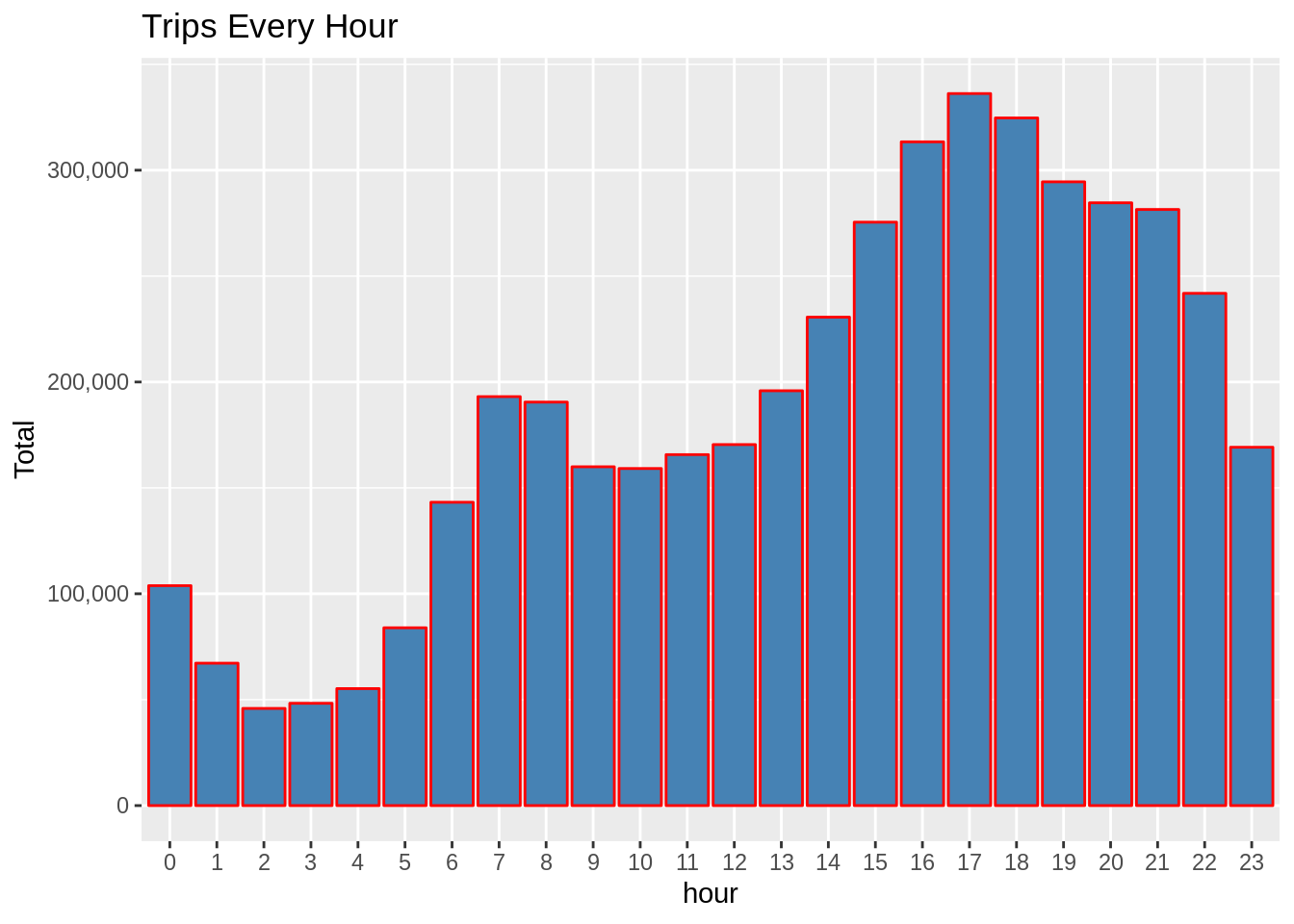
**Input Screenshot 9:**



**Output:**



**Output:**



### Plotting data by trips during every day of the month

In this section of DataFlair R project, we will learn how to plot our data based on every day of the month. We observe from the resulting visualization that 30th of the month had the highest trips in the year which is mostly contributed by the month of April.

**Code:**

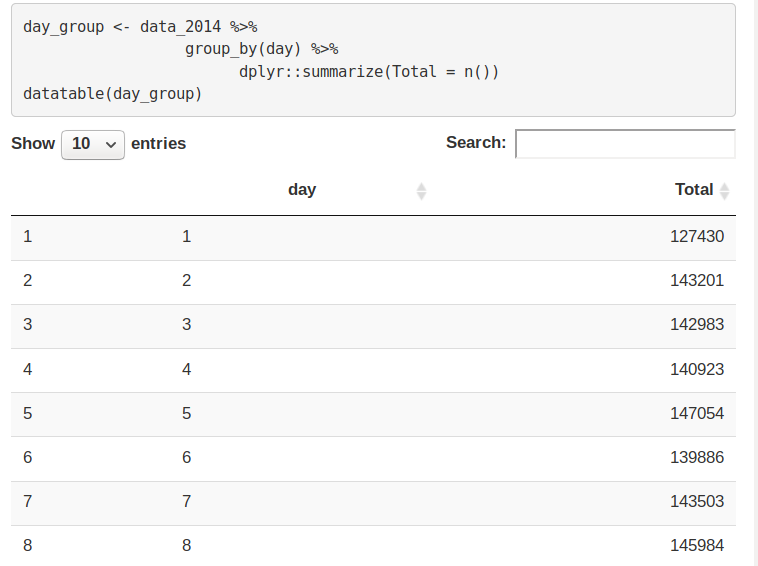
day\_group <- data\_2014 %>%

group\_by(day) %>%

dplyr::summarize(Total = n())

datatable(day\_group)

**Output Screenshot:**



**Code:**

ggplot(day\_group, aes(day, Total)) +

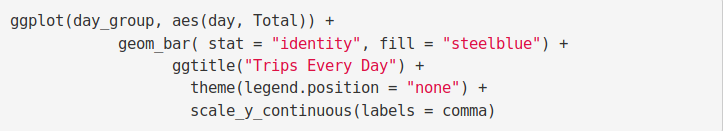
geom\_bar( stat = "identity", fill = "steelblue") +

ggtitle("Trips Every Day") +

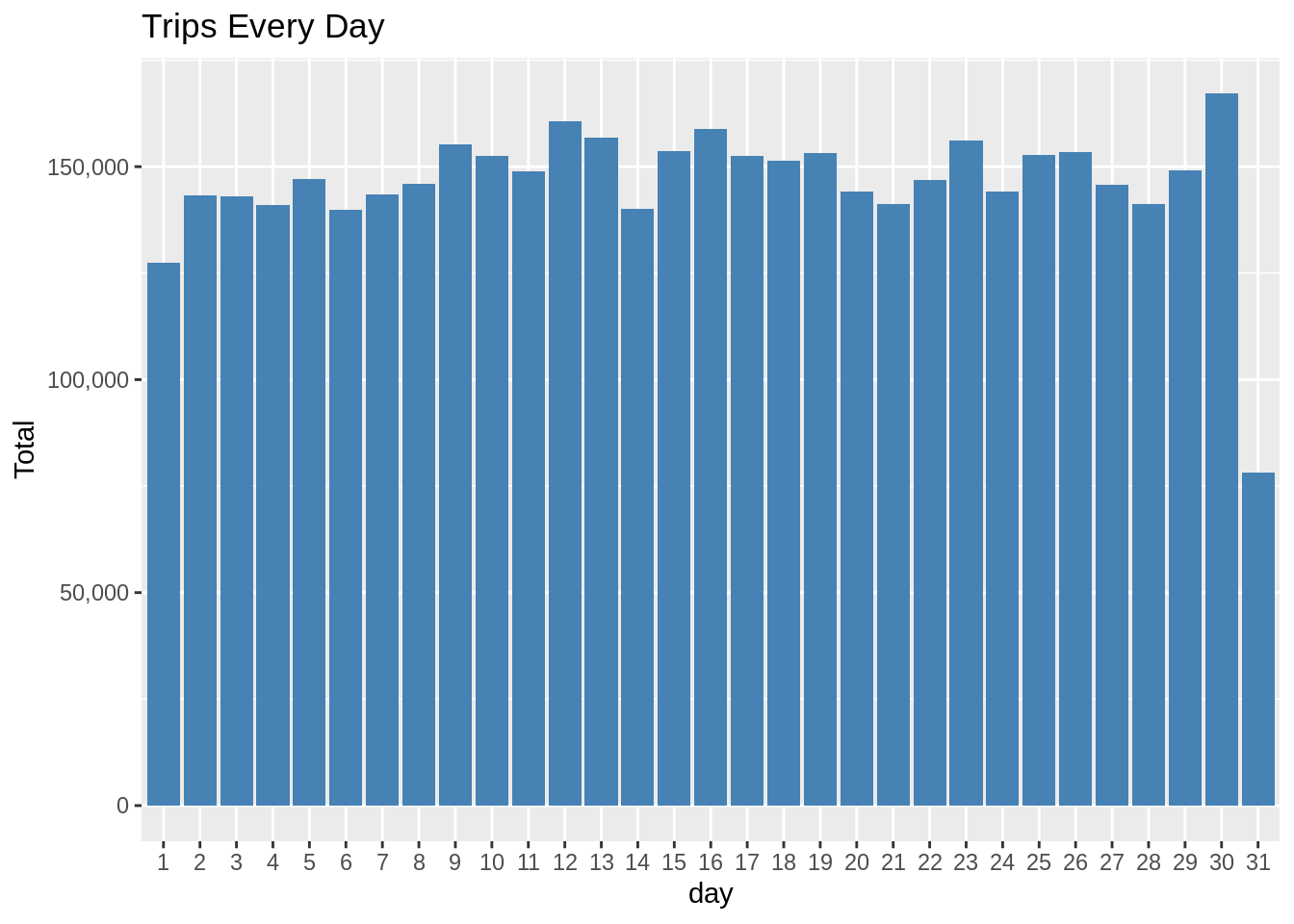
theme(legend.position = "none") +

scale\_y\_continuous(labels = comma)

**Input Screenshot 10:**



**Output:**



**Code:**

day\_month\_group <- data\_2014 %>%

group\_by(month, day) %>%

dplyr::summarize(Total = n())

ggplot(day\_month\_group, aes(day, Total, fill = month)) +

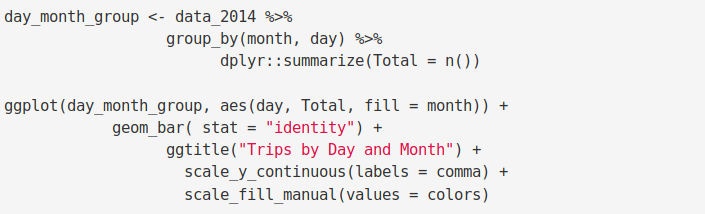
geom\_bar( stat = "identity") +

ggtitle("Trips by Day and Month") +

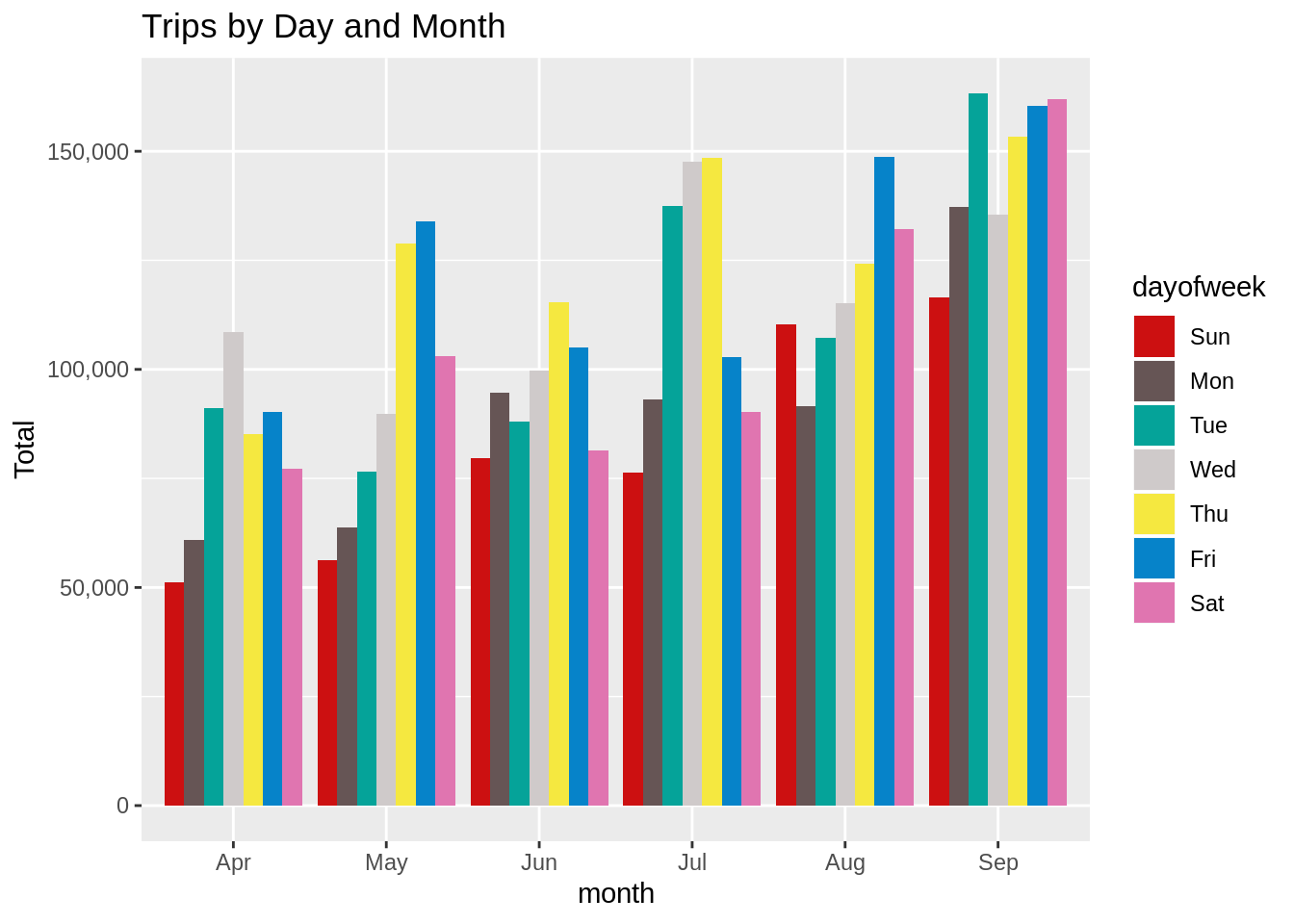
scale\_y\_continuous(labels = comma) +

scale\_fill\_manual(values = colors)

**Input Screenshot 11:**



**Output:**



### Number of Trips taking place during months in a year

In this section, we will visualize the number of trips that are taking place each month of the year. In the output visualization, we observe that most trips were made during the month of September. Furthermore, we also obtain visual reports of the number of trips that were made on every day of the week.

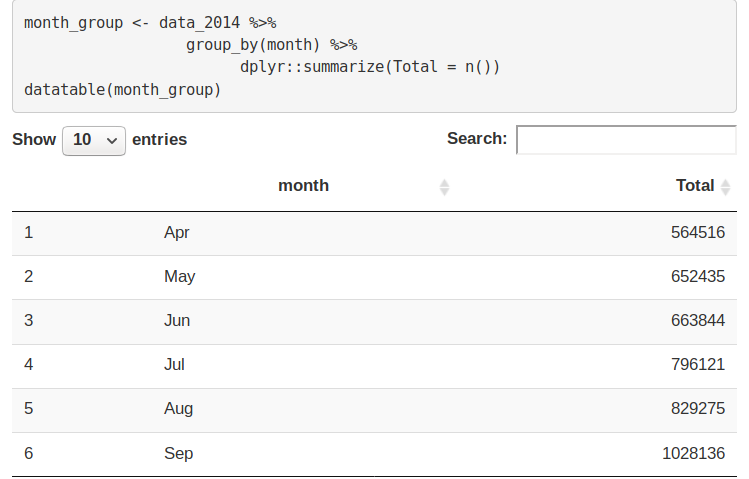
**Code:**

month\_group <- data\_2014 %>%

group\_by(month) %>%

dplyr::summarize(Total = n())

datatable(month\_group)

**Output Screenshot:**

**Code:**

ggplot( , aes(month, Total, fill = month)) +

geom\_bar( stat = "identity") +

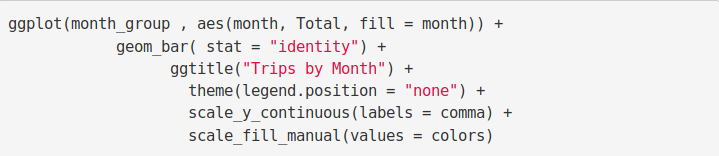
ggtitle("Trips by Month") +

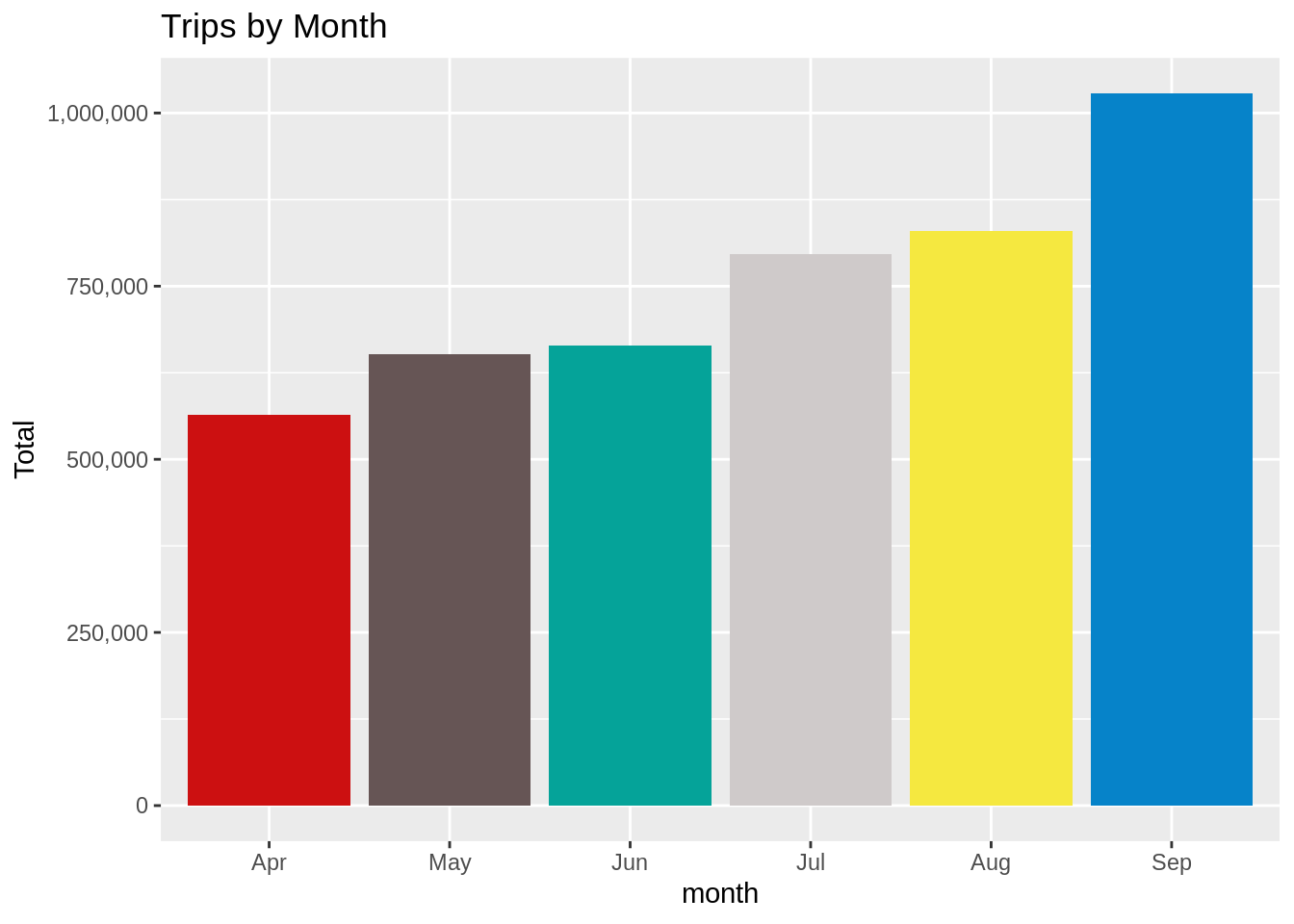
theme(legend.position = "none") +

scale\_y\_continuous(labels = comma) +

scale\_fill\_manual(values = colors)

**Input Screenshot 12:**



**Output:**

month\_weekday <- data\_2014 %>%

group\_by(month, dayofweek) %>%

dplyr::summarize(Total = n())

ggplot(month\_weekday, aes(month, Total, fill = dayofweek)) +

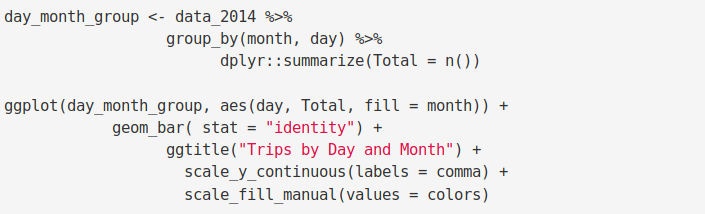
geom\_bar( stat = "identity", position = "dodge") +

ggtitle("Trips by Day and Month") +

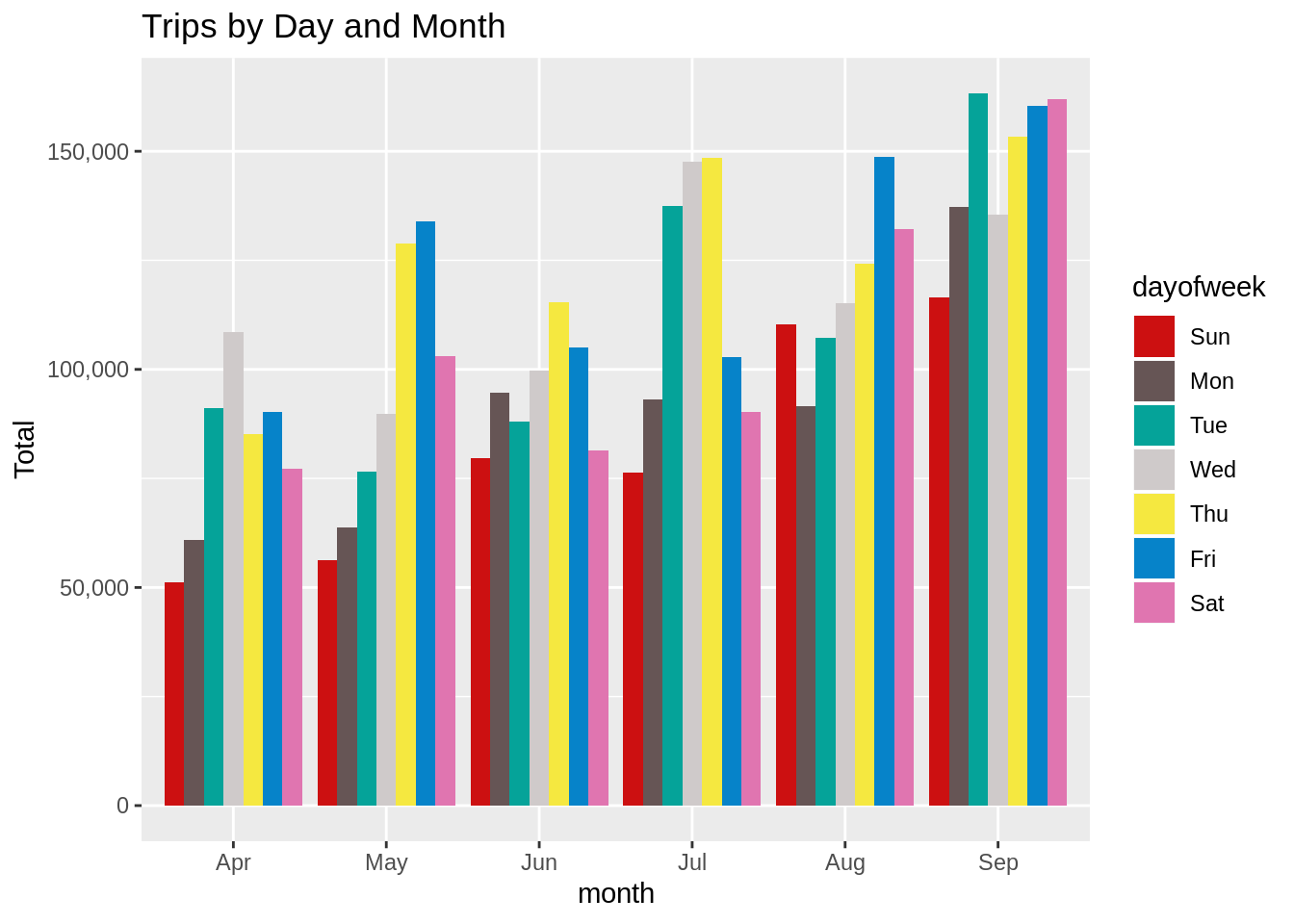
scale\_y\_continuous(labels = comma) +

scale\_fill\_manual(values = colors)

**Input Screenshot 13:**



**Output:**



### Finding out the number of Trips by bases

In the following visualization, we plot the number of trips that have been taken by the passengers from each of the bases. There are five bases in all out of which, we observe that B02617 had the highest number of trips. Furthermore, this base had the highest number of trips in the month B02617. Thursday observed highest trips in the three bases – B02598, B02617, B02682.

**Code:**

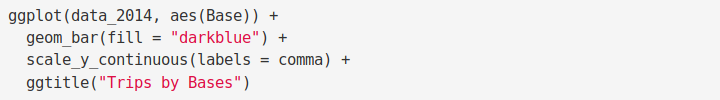
ggplot(data\_2014, aes(Base)) +

geom\_bar(fill = "darkred") +

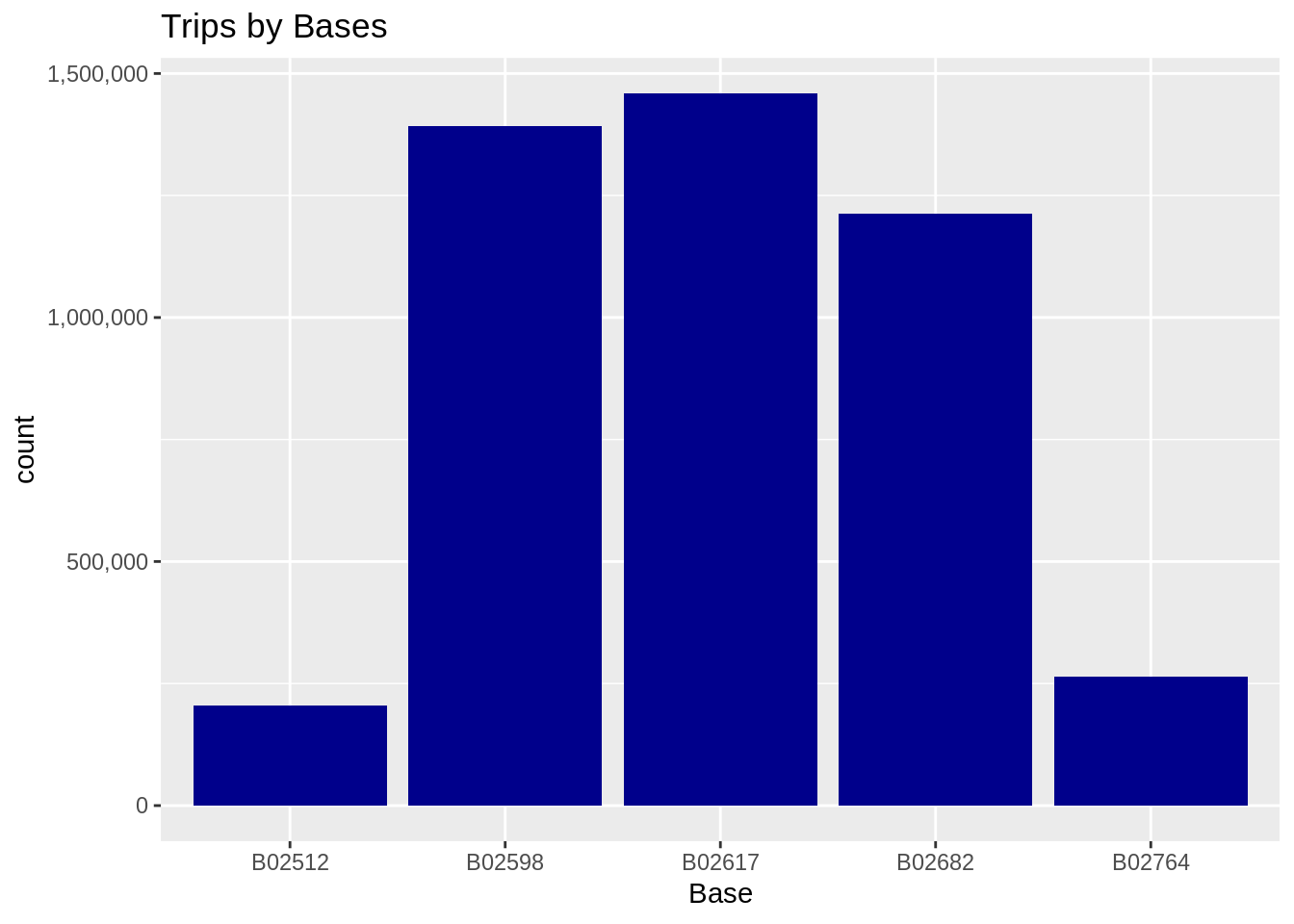
scale\_y\_continuous(labels = comma) +

ggtitle("Trips by Bases")

**Input Screenshot 14:**



**Output:**



**Code:**

ggplot(data\_2014, aes(Base, fill = month)) +

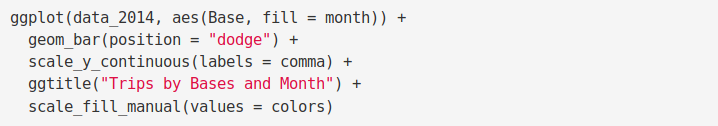
geom\_bar(position = "dodge") +

scale\_y\_continuous(labels = comma) +

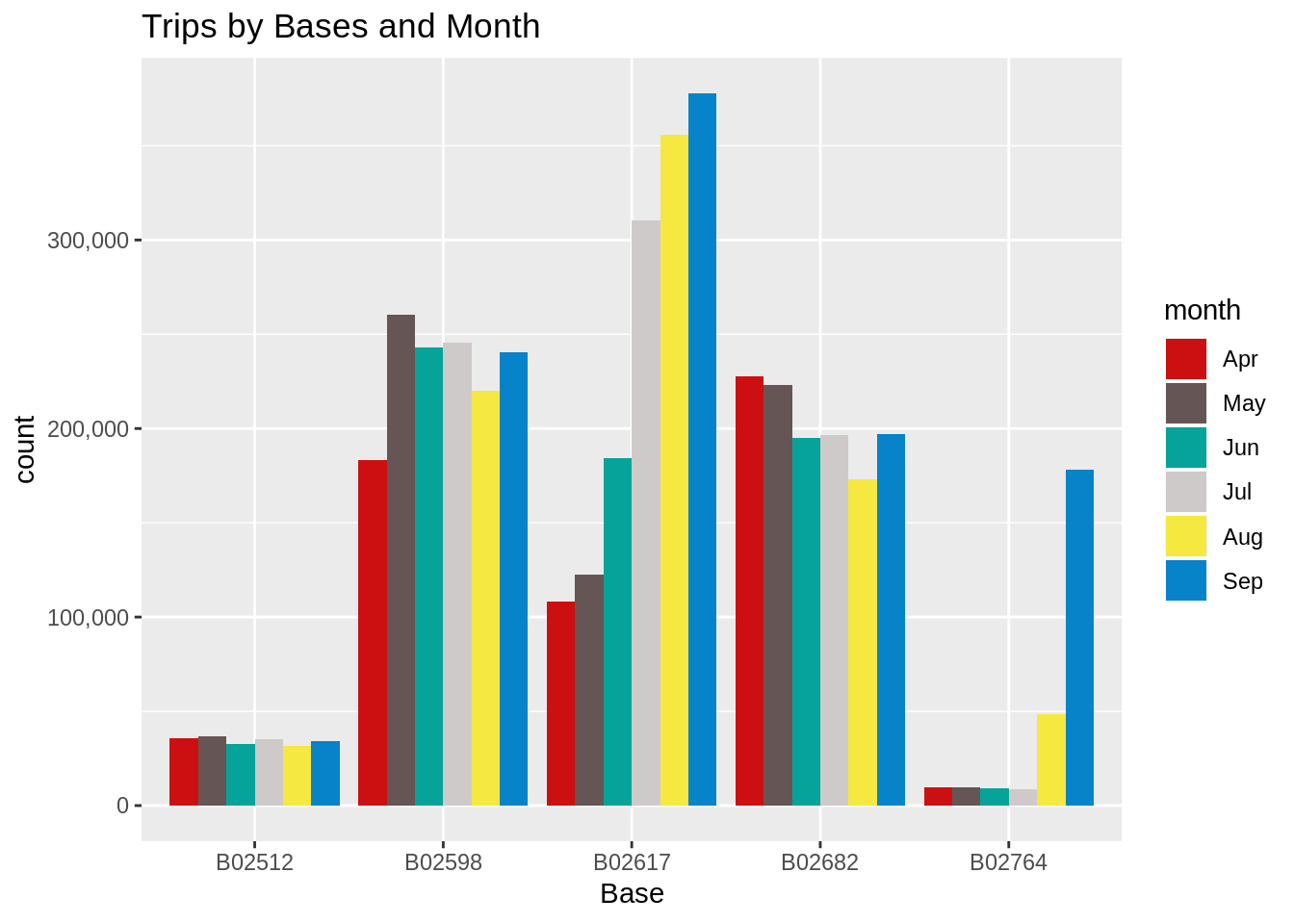
ggtitle("Trips by Bases and Month") +

scale\_fill\_manual(values = colors)

**Input Screenshot 15:**



**Output:**



**Code:**

ggplot(data\_2014, aes(Base, fill = dayofweek)) +

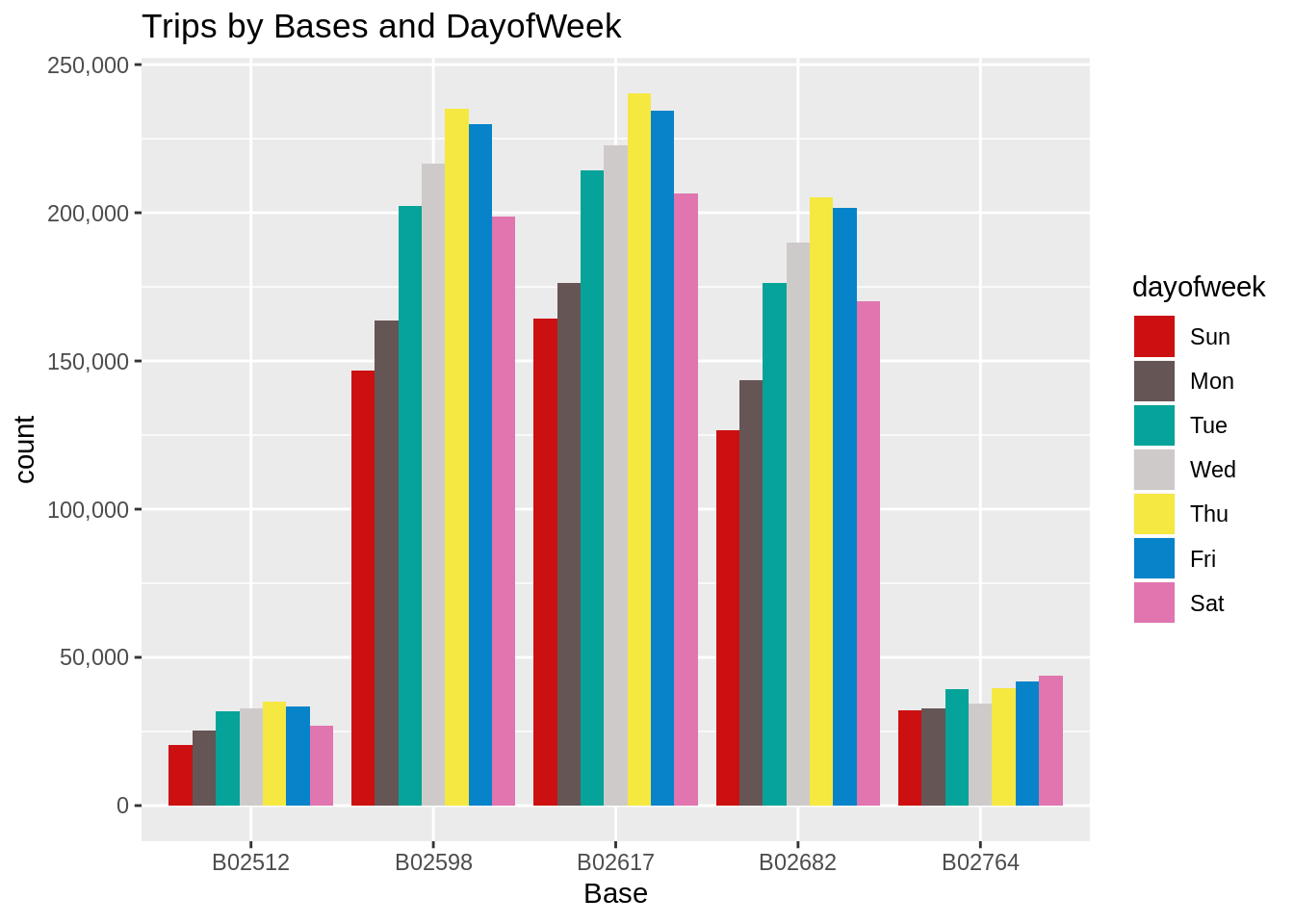
geom\_bar(position = "dodge") +

scale\_y\_continuous(labels = comma) +

ggtitle("Trips by Bases and DayofWeek") +

scale\_fill\_manual(values = colors)

**Output:**



### Creating a Heatmap visualization of day, hour and month

In this section, we will learn how to plot heatmaps using ggplot(). We will plot five heatmap plots –

* First, we will plot [Heatmap](https://en.wikipedia.org/wiki/Heat_map) by Hour and Day.
* Second, we will plot Heatmap by Month and Day.
* Third, a Heatmap by Month and Day of the Week.
* Fourth, a Heatmap that delineates Month and Bases.
* Finally, we will plot the heatmap, by bases and day of the week.

**Code:**

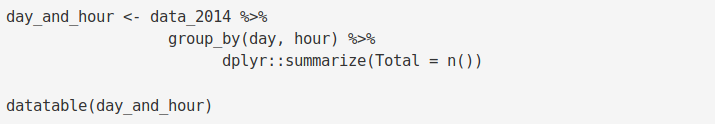
day\_and\_hour <- data\_2014 %>%

group\_by(day, hour) %>%

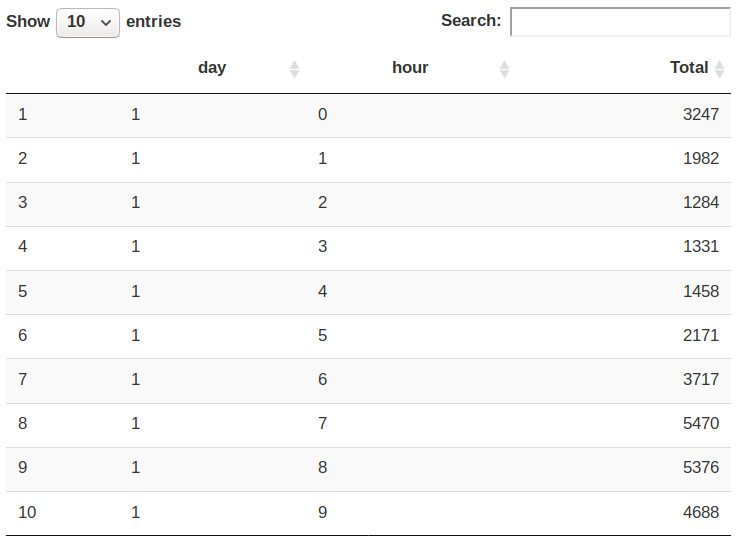
dplyr::summarize(Total = n())

datatable(day\_and\_hour)

**Input Screenshot 16:**



**Output Screenshot:**



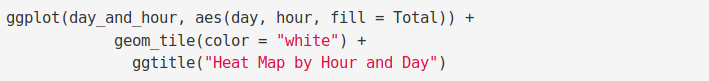
**Code:**

ggplot(day\_and\_hour, aes(day, hour, fill = Total)) +

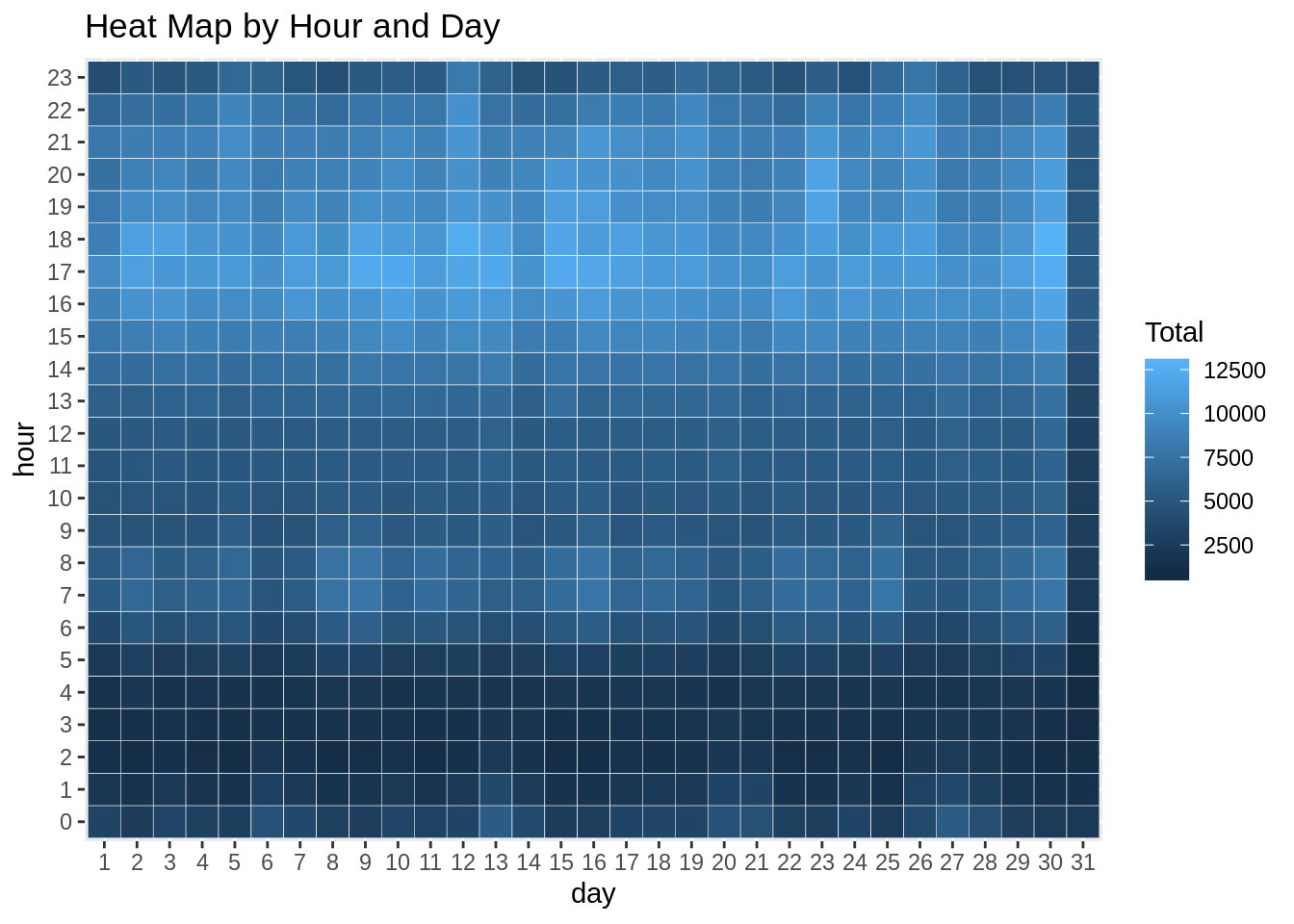
geom\_tile(color = "white") +

ggtitle("Heat Map by Hour and Day")

**Input Screenshot 17:**



**Output:**



**Code:**

ggplot(day\_month\_group, aes(day, month, fill = Total)) +

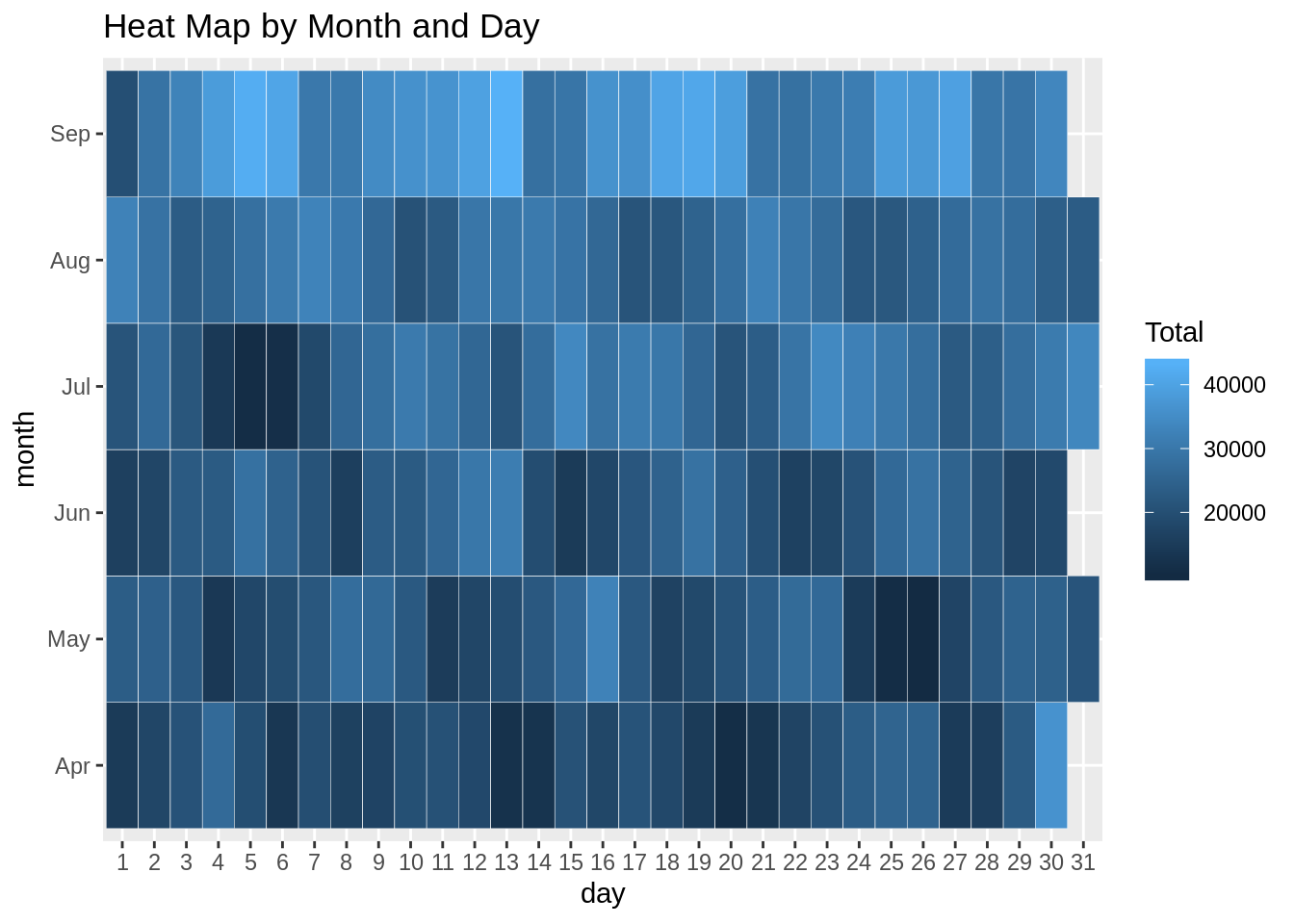
geom\_tile(color = "white") +

ggtitle("Heat Map by Month and Day")

**Input Screenshot 18:**



**Output:**



**Code:**

ggplot(month\_weekday, aes(dayofweek, month, fill = Total)) +

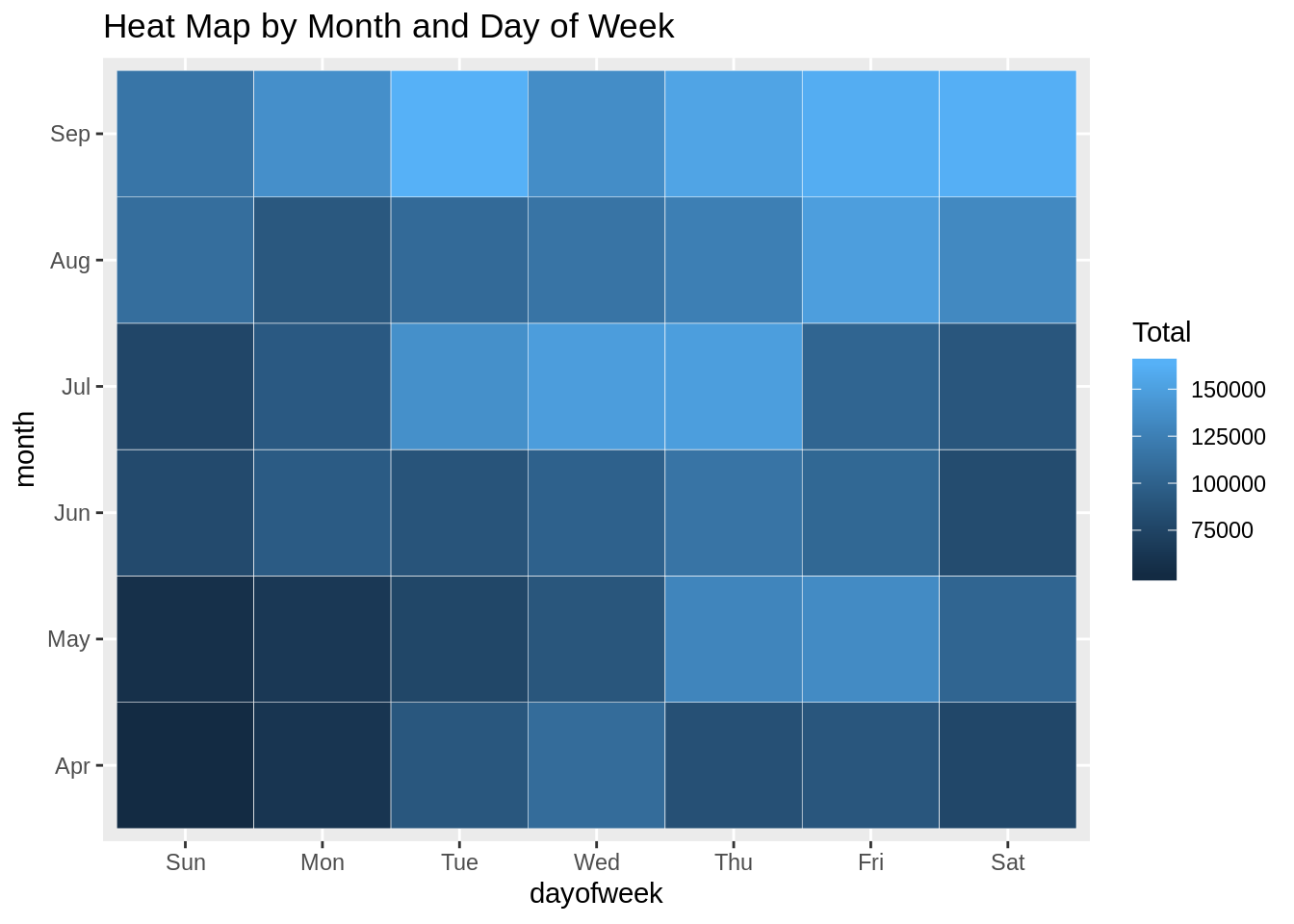
geom\_tile(color = "white") +

ggtitle("Heat Map by Month and Day of Week")

**Input Screenshot 19:**



**Output:**



**Code:**

month\_base <- data\_2014 %>%

group\_by(Base, month) %>%

dplyr::summarize(Total = n())

day0fweek\_bases <- data\_2014 %>%

group\_by(Base, dayofweek) %>%

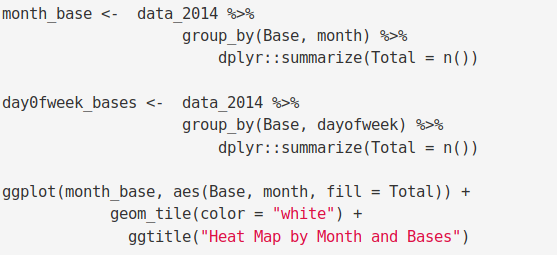
dplyr::summarize(Total = n())

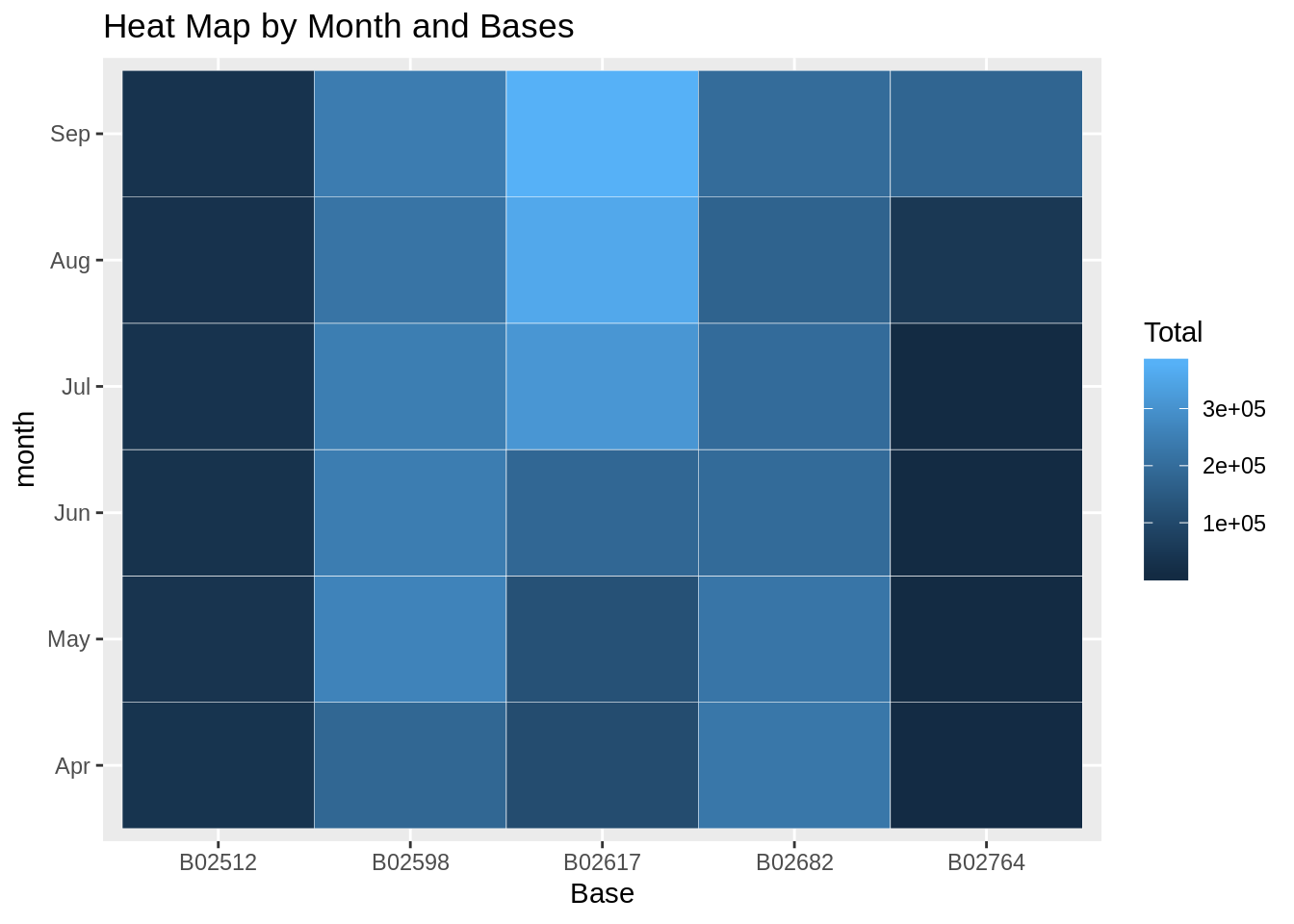
ggplot(month\_base, aes(Base, month, fill = Total)) +

geom\_tile(color = "white") +

ggtitle("Heat Map by Month and Bases")

**Input Screenshot 20:**

  
**Output:**



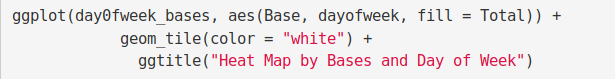
**Code:**

ggplot(day0fweek\_bases, aes(Base, dayofweek, fill = Total)) +

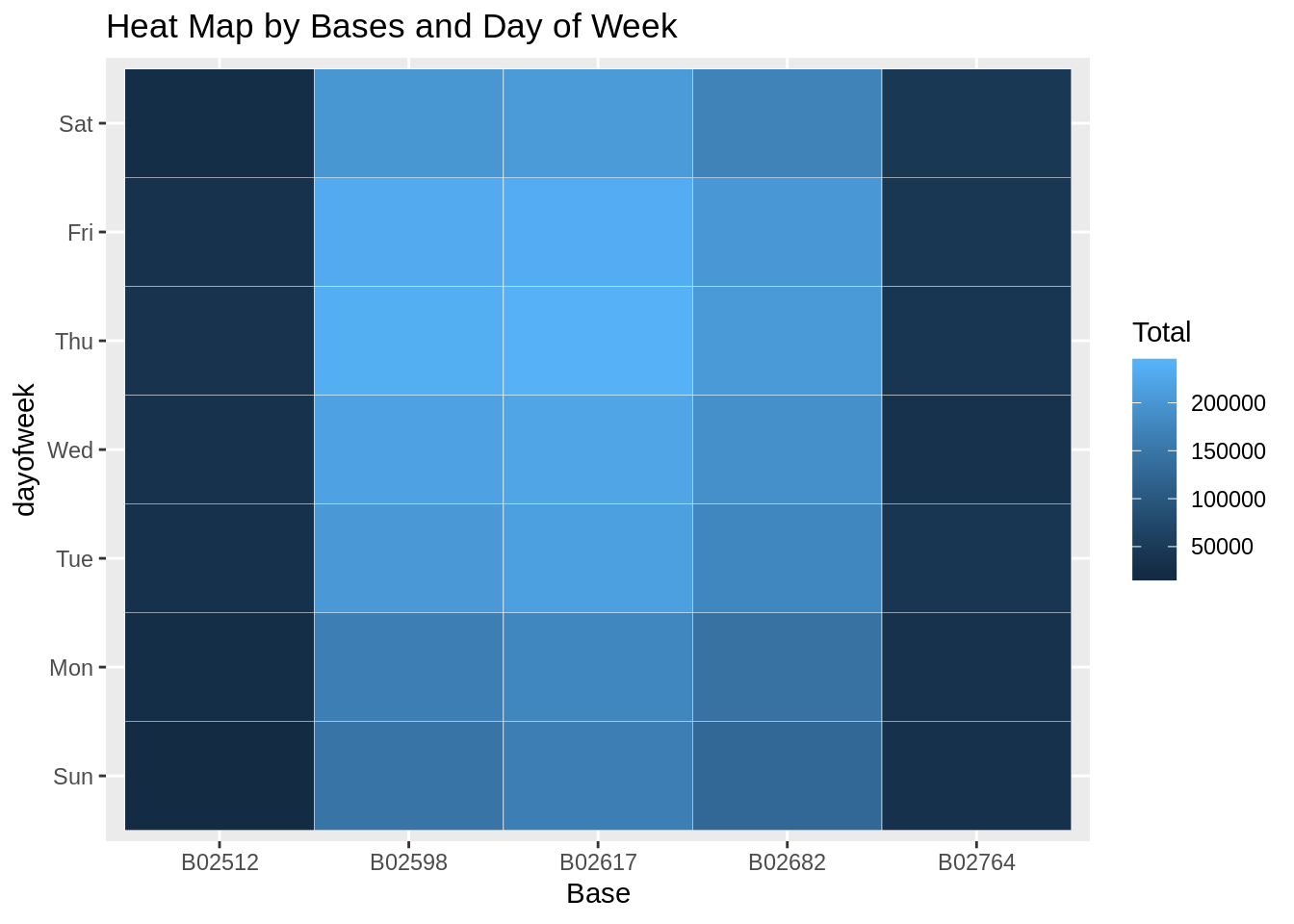
geom\_tile(color = "white") +

ggtitle("Heat Map by Bases and Day of Week")

**Input Screenshot 21:**



**Output:**



### Creating a map visualization of rides in New York

In the final section, we will visualize the rides in New York city by creating a geo-plot that will help us to visualize the rides during 2014 (Apr – Sep) and by the bases in the same period.

**Code:**

min\_lat <- 40.5774

max\_lat <- 40.9176

min\_long <- -74.15

max\_long <- -73.7004

ggplot(data\_2014, aes(x=Lon, y=Lat)) +

geom\_point(size=1, color = "blue") +

scale\_x\_continuous(limits=c(min\_long, max\_long)) +

scale\_y\_continuous(limits=c(min\_lat, max\_lat)) +

theme\_map() +

ggtitle("NYC MAP BASED ON UBER RIDES DURING 2014 (APR-SEP)")

ggplot(data\_2014, aes(x=Lon, y=Lat, color = Base)) +

geom\_point(size=1) +

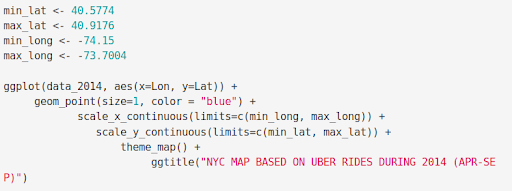
scale\_x\_continuous(limits=c(min\_long, max\_long)) +

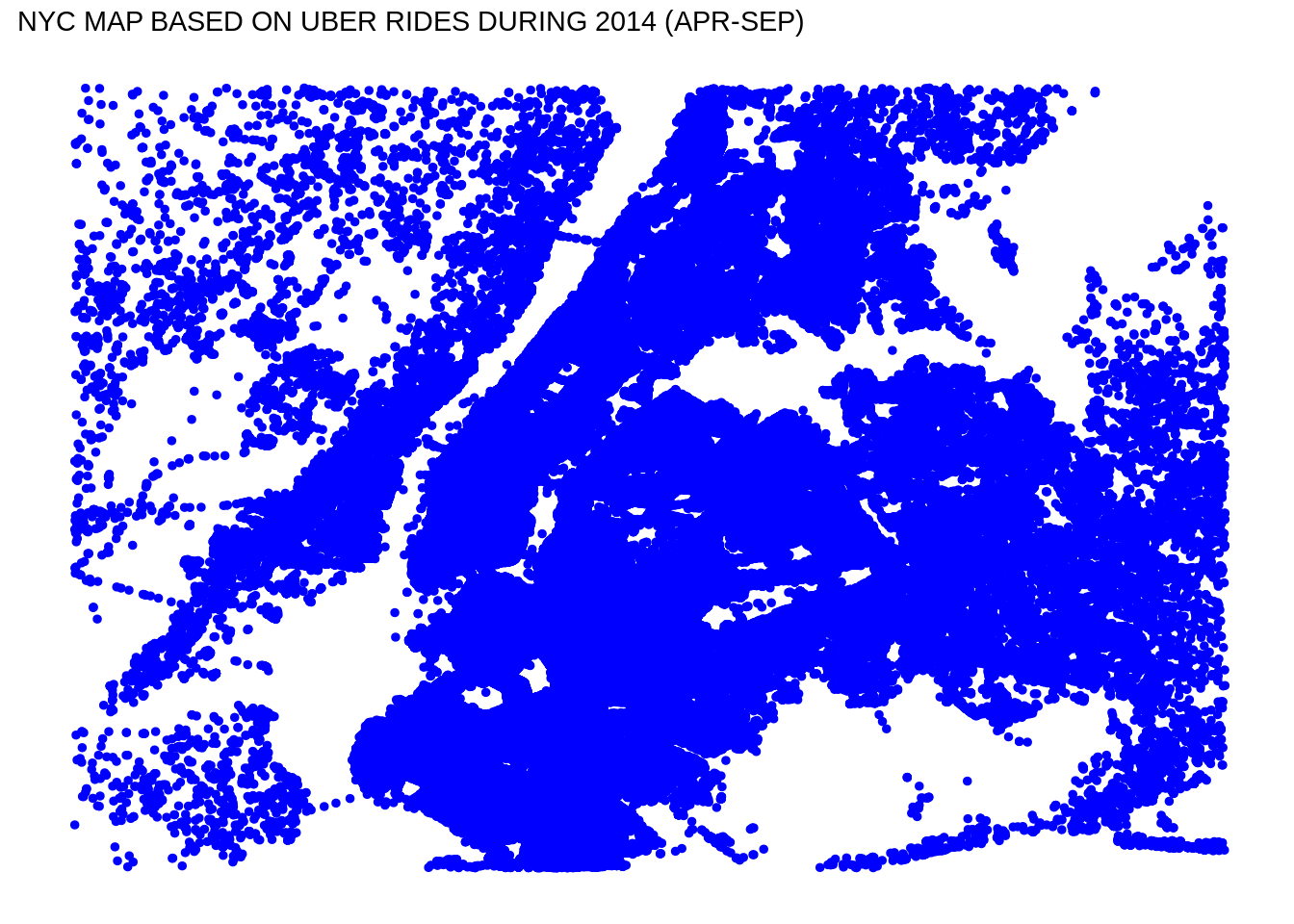
scale\_y\_continuous(limits=c(min\_lat, max\_lat)) +

theme\_map() +

ggtitle("NYC MAP BASED ON UBER RIDES DURING 2014 (APR-SEP) by BASE")

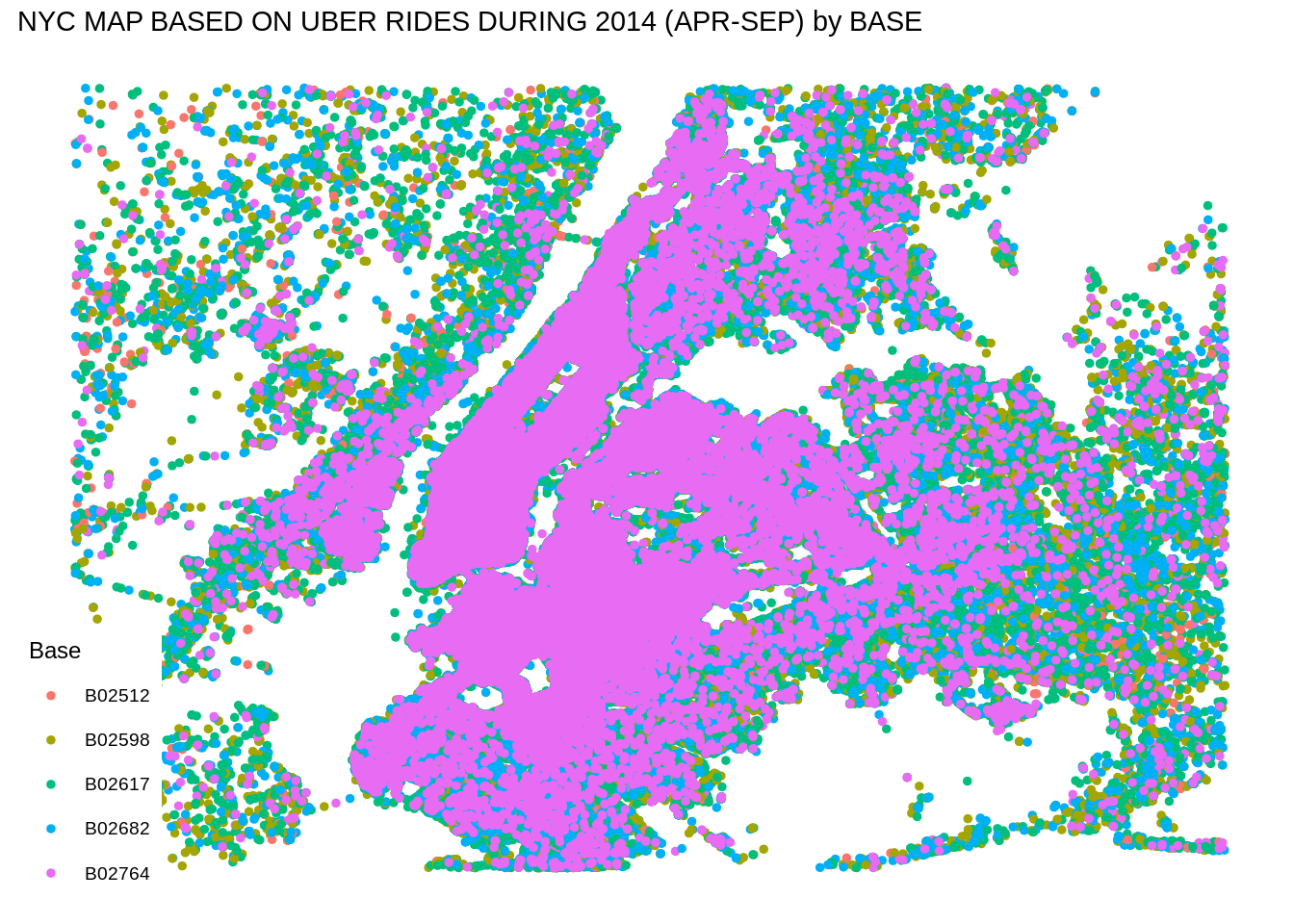
**Input Screenshot 22:**

  
**Output:**



Uber data analysis using R

**Output:**



## Summary

At the end of the Uber data analysis R project, we observed how to create data visualizations. We made use of packages like ggplot2 that allowed us to plot various types of visualizations that pertained to several time-frames of the year. With this, we could conclude how time affected customer trips. Finally, we made a geo plot of New York that provided us with the details of how various users made trips from different bases.