Uber Data Analysis through Visualisations in R

Data storytelling is an important component of Machine Learning 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. 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.

**Important**: The goal of this project is to learn visualisations in R. I do not claim copyright over any of the content here.

Source: [here](https://data-flair.training/blogs/r-data-science-project-uber-data-analysis/)

Importing necessary libraries

1. **gplot2**: ggplot2 is the most popular data visualization library that is most widely used for creating aesthetic visualization plots.
2. **lubridate**: Use time-frames in the dataset
3. **dplyr**: Data Manipulation
4. **tidyr**: Tidy the data
5. **DT**: Datatables in JS
6. **scales**: With the help of graphical scales, we can automatically map the data to the correct scales with well-placed axes and legends.

In [1]:

library(ggplot2)

library(ggthemes)

library(lubridate)

library(dplyr)

library(tidyr)

library(tidyverse) *# metapackage of all tidyverse packages*

library(DT)

library(scales)

Attaching package: ‘lubridate’

The following objects are masked from ‘package:base’:

date, intersect, setdiff, union

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

filter, lag

The following objects are masked from ‘package:base’:

intersect, setdiff, setequal, union

── **Attaching packages** ─────────────────────────────────────── tidyverse 1.3.0 ──

✔ tibble 3.0.6 ✔ stringr 1.4.0

✔ readr 1.4.0 ✔ forcats 0.5.0

✔ purrr 0.3.4

── **Conflicts** ────────────────────────────────────────── tidyverse\_conflicts() ──

✖ lubridate::as.difftime() masks base::as.difftime()

✖ lubridate::date() masks base::date()

✖ dplyr::filter() masks stats::filter()

✖ lubridate::intersect() masks base::intersect()

✖ dplyr::lag() masks stats::lag()

✖ lubridate::setdiff() masks base::setdiff()

✖ lubridate::union() masks base::union()

Attaching package: ‘scales’

The following object is masked from ‘package:purrr’:

discard

The following object is masked from ‘package:readr’:

col\_factor

Creating vector of colors for the plots

In [2]:

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

colors

1. '#CC1011'
2. '#665555'
3. '#05a399'
4. '#cfcaca'
5. '#f5e840'
6. '#0683c9'
7. '#e075b0'

Read the data from each time-frame

In [3]:

*# Read the data for each month separately*

apr <- read.csv("../input/uberdataset/uber-raw-data-apr14.csv")

may <- read.csv("../input/uberdataset/uber-raw-data-may14.csv")

june <- read.csv("../input/uberdataset/uber-raw-data-jun14.csv")

july <- read.csv("../input/uberdataset/uber-raw-data-jul14.csv")

aug <- read.csv("../input/uberdataset/uber-raw-data-aug14.csv")

sept <- read.csv("../input/uberdataset/uber-raw-data-sep14.csv")

*# Combine the data together*

data <- rbind(apr, may, june, july, aug, sept)

cat("The dimensions of the data are:", dim(data))

The dimensions of the data are: 4534327 4

In [4]:

*# Print the first 6 rows of the data*

head(data)

| A data.frame: 6 × 4 | | | | |
| --- | --- | --- | --- | --- |
|  | Date.Time | Lat | Lon | Base |
|  | <fct> | <dbl> | <dbl> | <fct> |
| 1 | 4/1/2014 0:11:00 | 40.7690 | -73.9549 | B02512 |
| 2 | 4/1/2014 0:17:00 | 40.7267 | -74.0345 | B02512 |
| 3 | 4/1/2014 0:21:00 | 40.7316 | -73.9873 | B02512 |
| 4 | 4/1/2014 0:28:00 | 40.7588 | -73.9776 | B02512 |
| 5 | 4/1/2014 0:33:00 | 40.7594 | -73.9722 | B02512 |
| 6 | 4/1/2014 0:33:00 | 40.7383 | -74.0403 | B02512 |

The data contains the columns Date.Time which is a factor, Latitude and Longitudes which are double and Base which is factor. we will format the datetime into a more readable format using the Date Time conversion function.

In [5]:

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

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

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

In [6]:

*# Create individual columns for month day and year*

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

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

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

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

In [7]:

*# Add Time variables as well*

data$second = factor(second(hms(data$Time)))

data$minute = factor(minute(hms(data$Time)))

data$hour = factor(hour(hms(data$Time)))

In [8]:

*# Look at the data*

head(data)

| A data.frame: 6 × 12 | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Date.Time | Lat | Lon | Base | Time | day | month | year | dayofweek | second | minute | hour |
|  | <dttm> | <dbl> | <dbl> | <fct> | <chr> | <fct> | <ord> | <fct> | <ord> | <fct> | <fct> | <fct> |
| 1 | 2014-04-01 00:11:00 | 40.7690 | -73.9549 | B02512 | 00:11:00 | 1 | Apr | 2014 | Tue | 0 | 11 | 0 |
| 2 | 2014-04-01 00:17:00 | 40.7267 | -74.0345 | B02512 | 00:17:00 | 1 | Apr | 2014 | Tue | 0 | 17 | 0 |
| 3 | 2014-04-01 00:21:00 | 40.7316 | -73.9873 | B02512 | 00:21:00 | 1 | Apr | 2014 | Tue | 0 | 21 | 0 |
| 4 | 2014-04-01 00:28:00 | 40.7588 | -73.9776 | B02512 | 00:28:00 | 1 | Apr | 2014 | Tue | 0 | 28 | 0 |
| 5 | 2014-04-01 00:33:00 | 40.7594 | -73.9722 | B02512 | 00:33:00 | 1 | Apr | 2014 | Tue | 0 | 33 | 0 |
| 6 | 2014-04-01 00:33:00 | 40.7383 | -74.0403 | B02512 | 00:33:00 | 1 | Apr | 2014 | Tue | 0 | 33 | 0 |

Data Visualisation

Plotting the trips by hours in a day

In [9]:

hourly\_data <- data %>%

group\_by(hour) %>%

dplyr::summarize(Total = n())

*# Shos data in a searchable js table*

datatable(hourly\_data)

|  | hour | Total |
| --- | --- | --- |

In [10]:

*# Plot the data by hour*

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

geom\_bar(stat="identity",

fill="steelblue",

color="red") +

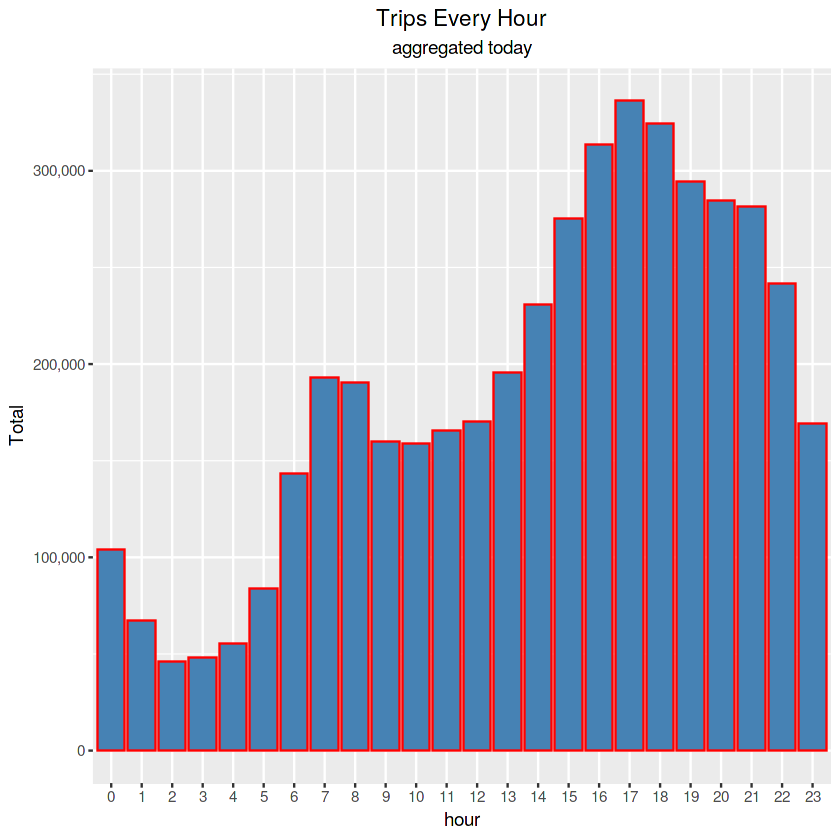
ggtitle("Trips Every Hour", subtitle = "aggregated today") +

theme(legend.position = "none",

plot.title = element\_text(hjust = 0.5),

plot.subtitle = element\_text(hjust = 0.5)) +

scale\_y\_continuous(labels=comma)



Plotting trips by hour and month

In [11]:

*# Aggregate the data by month and hour*

month\_hour\_data <- data %>% group\_by(month, hour) %>% dplyr::summarize(Total = n())

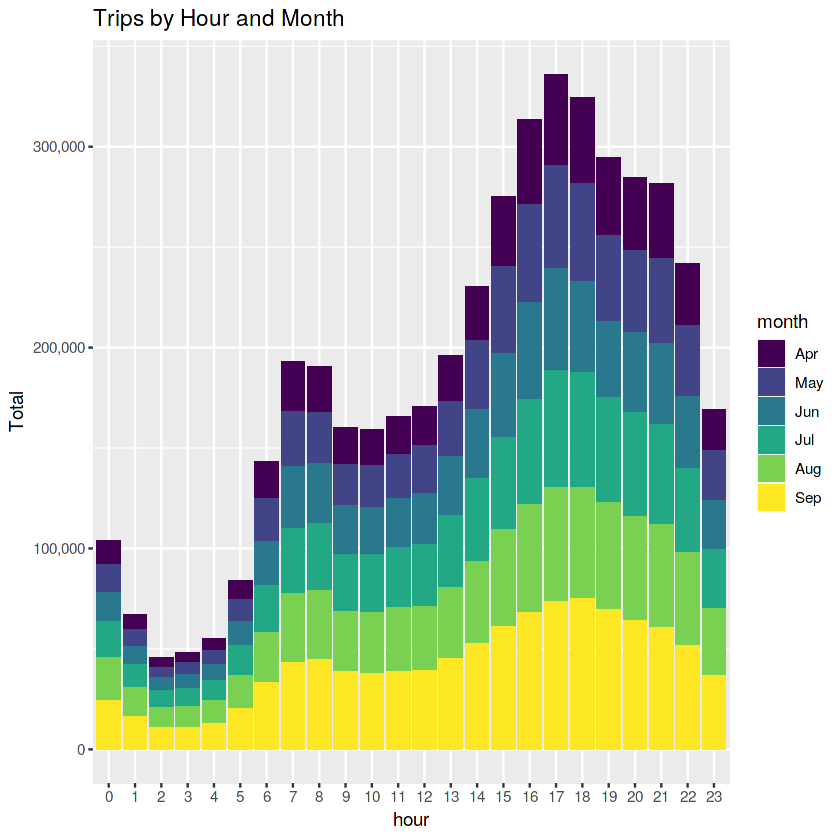
ggplot(month\_hour\_data, aes(hour, Total, fill=month)) +

geom\_bar(stat = "identity") +

ggtitle("Trips by Hour and Month") +

scale\_y\_continuous(labels = comma)

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



The highest

Plotting data by trips during every day of the month

In [12]:

*# Aggregate data by day of the month*

day\_data <- data %>% group\_by(day) %>% dplyr::summarize(Trips = n())

day\_data

| A tibble: 31 × 2 | | |
| --- | --- | --- |
|  | day | Trips |
|  | <fct> | <int> |
| 1 | 1 | 127430 |
| 2 | 2 | 143201 |
| 3 | 3 | 142983 |
| 4 | 4 | 140923 |
| 5 | 5 | 147054 |
| 6 | 6 | 139886 |
| 7 | 7 | 143503 |
| 8 | 8 | 145984 |
| 9 | 9 | 155135 |
| 10 | 10 | 152500 |
| 11 | 11 | 148860 |
| 12 | 12 | 160606 |
| 13 | 13 | 156892 |
| 14 | 14 | 140148 |
| 15 | 15 | 153726 |
| 16 | 16 | 158921 |
| 17 | 17 | 152524 |
| 18 | 18 | 151319 |
| 19 | 19 | 153088 |
| 20 | 20 | 144179 |
| 21 | 21 | 141112 |
| 22 | 22 | 146952 |
| 23 | 23 | 156032 |
| 24 | 24 | 144169 |
| 25 | 25 | 152667 |
| 26 | 26 | 153405 |
| 27 | 27 | 145652 |
| 28 | 28 | 141157 |
| 29 | 29 | 149086 |
| 30 | 30 | 167160 |
| 31 | 31 | 78073 |

In [13]:

*# Plot the data for the day*

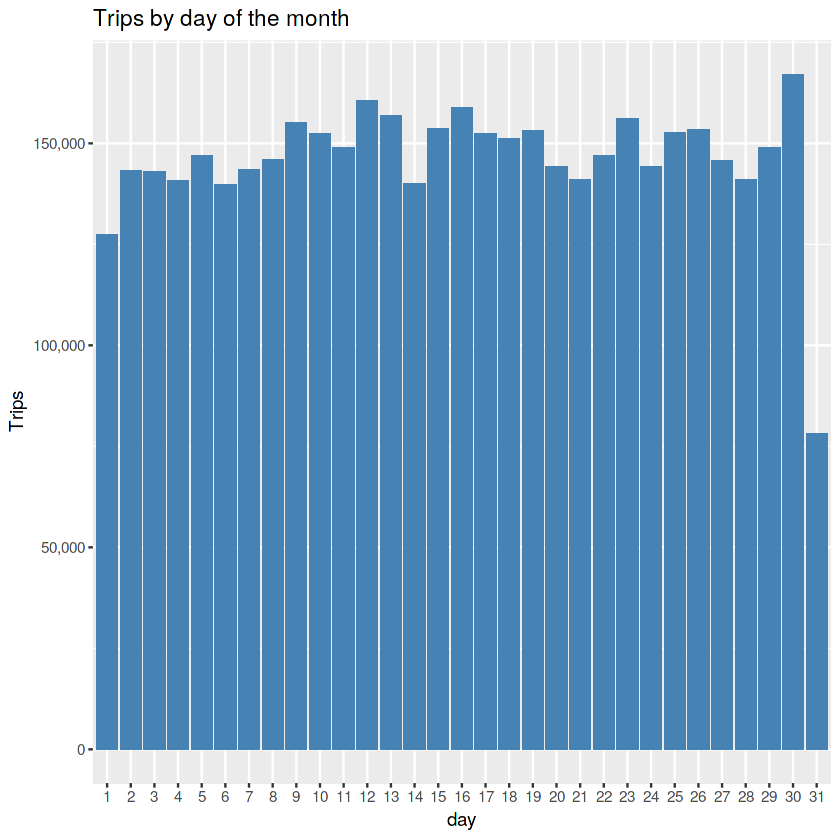
ggplot(day\_data, aes(day, Trips)) +

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

ggtitle("Trips by day of the month") +

theme(legend.position = "none") +

scale\_y\_continuous(labels = comma)



In [14]:

*# Collect data by day of the week and month*

day\_month\_data <- data %>% group\_by(dayofweek, month) %>% dplyr::summarize(Trips = n())

day\_month\_data

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

| A grouped\_df: 42 × 3 | | |
| --- | --- | --- |
| dayofweek | month | Trips |
| <ord> | <ord> | <int> |
| Sun | Apr | 51251 |
| Sun | May | 56168 |
| Sun | Jun | 79656 |
| Sun | Jul | 76327 |
| Sun | Aug | 110246 |
| Sun | Sep | 116532 |
| Mon | Apr | 60861 |
| Mon | May | 63846 |
| Mon | Jun | 94655 |
| Mon | Jul | 93189 |
| Mon | Aug | 91633 |
| Mon | Sep | 137288 |
| Tue | Apr | 91185 |
| Tue | May | 76662 |
| Tue | Jun | 88134 |
| Tue | Jul | 137454 |
| Tue | Aug | 107124 |
| Tue | Sep | 163230 |
| Wed | Apr | 108631 |
| Wed | May | 89857 |
| Wed | Jun | 99654 |
| Wed | Jul | 147717 |
| Wed | Aug | 115256 |
| Wed | Sep | 135373 |
| Thu | Apr | 85067 |
| Thu | May | 128921 |
| Thu | Jun | 115325 |
| Thu | Jul | 148439 |
| Thu | Aug | 124117 |
| Thu | Sep | 153276 |
| Fri | Apr | 90303 |
| Fri | May | 133991 |
| Fri | Jun | 105056 |
| Fri | Jul | 102735 |
| Fri | Aug | 148674 |
| Fri | Sep | 160380 |
| Sat | Apr | 77218 |
| Sat | May | 102990 |
| Sat | Jun | 81364 |
| Sat | Jul | 90260 |
| Sat | Aug | 132225 |
| Sat | Sep | 162057 |

In [15]:

*# Plot the above data*

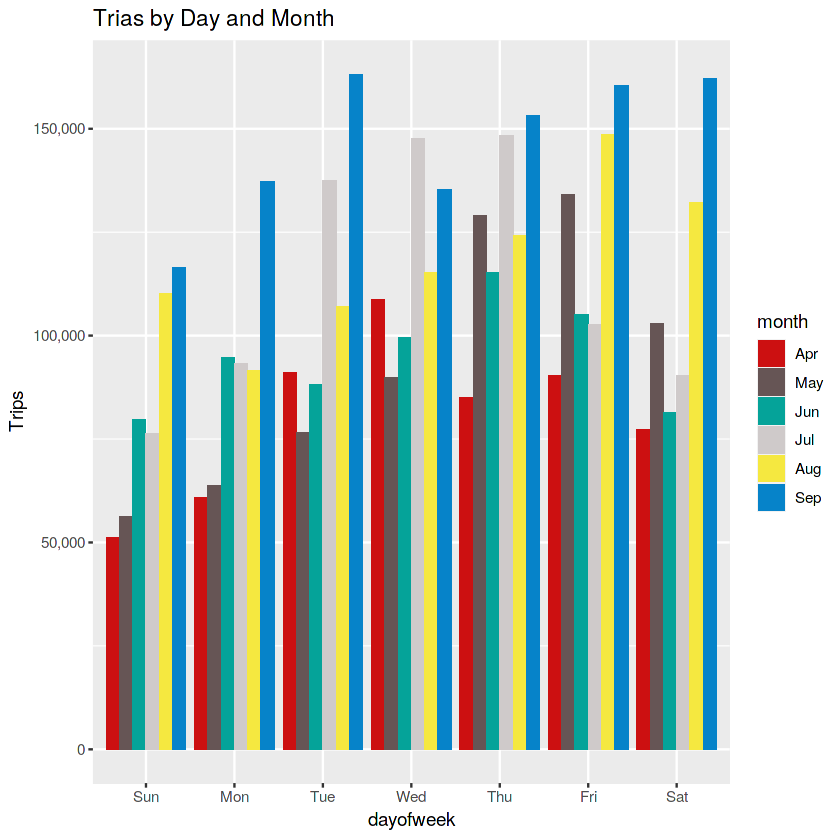
ggplot(day\_month\_data, aes(dayofweek, Trips, fill = month)) +

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

ggtitle("Trias by Day and Month") +

scale\_y\_continuous(labels = comma) +

scale\_fill\_manual(values = colors)



Number of Trips place during months in a year

In [16]:

month\_data <- data %>% group\_by(month) %>% dplyr::summarize(Total = n())

month\_data

| A tibble: 6 × 2 | | |
| --- | --- | --- |
|  | month | Total |
|  | <ord> | <int> |
| 1 | Apr | 564516 |
| 2 | May | 652435 |
| 3 | Jun | 663844 |
| 4 | Jul | 796121 |
| 5 | Aug | 829275 |
| 6 | Sep | 1028136 |

In [17]:

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

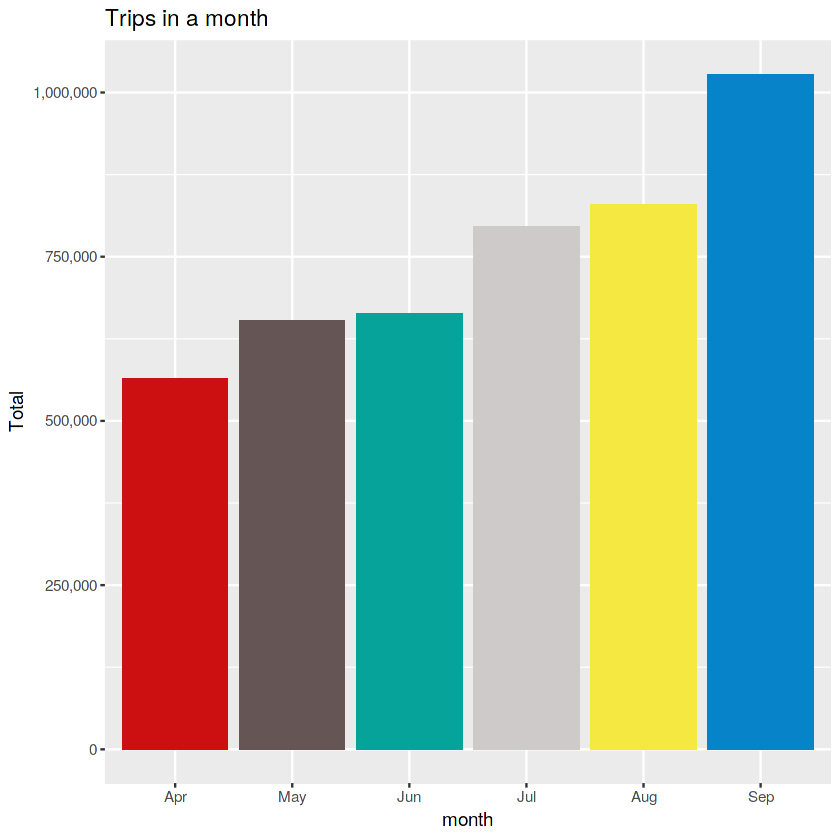
geom\_bar(stat = "Identity") +

ggtitle("Trips in a month") +

theme(legend.position = "none") +

scale\_y\_continuous(labels = comma) +

scale\_fill\_manual(values = colors)



Heatmap visualization of day, hour and month

Heatmap by Hour and Day

In [18]:

day\_hour\_data <- data %>% group\_by(day, hour) %>% dplyr::summarize(Total = n())

datatable(day\_hour\_data)

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

|  | day | hour | Total |
| --- | --- | --- | --- |

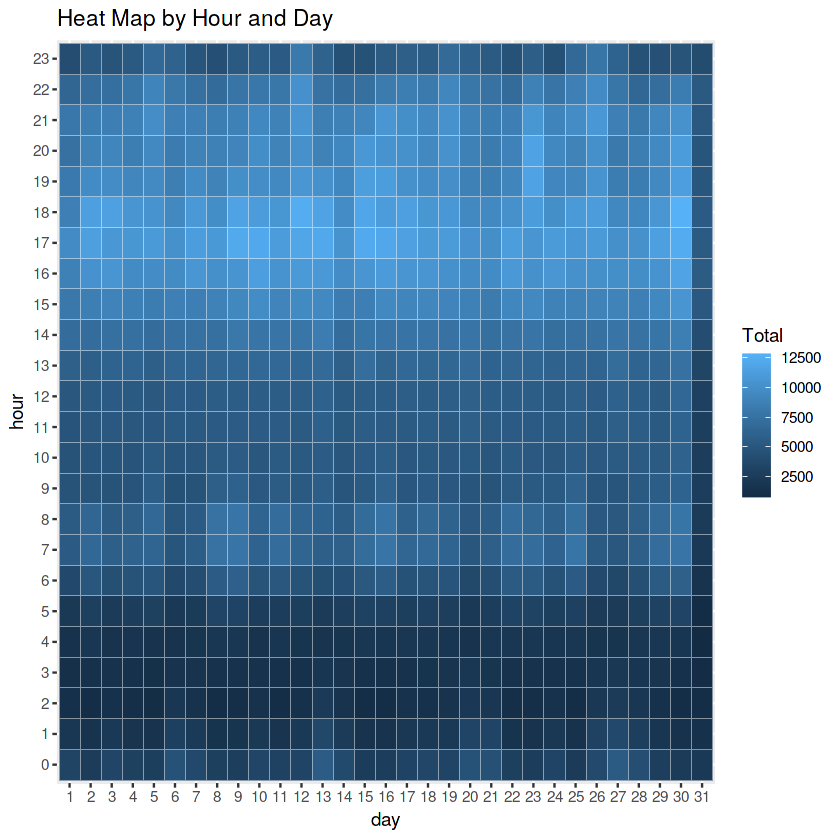
In [19]:

*# Plot a heatmap*

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

geom\_tile(color = "white") +

ggtitle("Heat Map by Hour and Day")



Plot Heatmap by day and month

In [20]:

*# Collect data by month and day*

month\_day\_data <- data %>% group\_by(month, day) %>% dplyr::summarize(Trips = n())

month\_day\_data

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

| A grouped\_df: 183 × 3 | | |
| --- | --- | --- |
| month | day | Trips |
| <ord> | <fct> | <int> |
| Apr | 1 | 14546 |
| Apr | 2 | 17474 |
| Apr | 3 | 20701 |
| Apr | 4 | 26714 |
| Apr | 5 | 19521 |
| Apr | 6 | 13445 |
| Apr | 7 | 19550 |
| Apr | 8 | 16188 |
| Apr | 9 | 16843 |
| Apr | 10 | 20041 |
| Apr | 11 | 20420 |
| Apr | 12 | 18170 |
| Apr | 13 | 12112 |
| Apr | 14 | 12674 |
| Apr | 15 | 20641 |
| Apr | 16 | 17717 |
| Apr | 17 | 20973 |
| Apr | 18 | 18074 |
| Apr | 19 | 14602 |
| Apr | 20 | 11017 |
| Apr | 21 | 13162 |
| Apr | 22 | 16975 |
| Apr | 23 | 20346 |
| Apr | 24 | 23352 |
| Apr | 25 | 25095 |
| Apr | 26 | 24925 |
| Apr | 27 | 14677 |
| Apr | 28 | 15475 |
| Apr | 29 | 22835 |
| Apr | 30 | 36251 |
| ⋮ | ⋮ | ⋮ |
| Sep | 1 | 19961 |
| Sep | 2 | 28831 |
| Sep | 3 | 32631 |
| Sep | 4 | 38360 |
| Sep | 5 | 42319 |
| Sep | 6 | 40520 |
| Sep | 7 | 30134 |
| Sep | 8 | 30360 |
| Sep | 9 | 34560 |
| Sep | 10 | 35910 |
| Sep | 11 | 36439 |
| Sep | 12 | 39540 |
| Sep | 13 | 43205 |
| Sep | 14 | 28122 |
| Sep | 15 | 29454 |
| Sep | 16 | 36092 |
| Sep | 17 | 35531 |
| Sep | 18 | 40274 |
| Sep | 19 | 41017 |
| Sep | 20 | 38864 |
| Sep | 21 | 28620 |
| Sep | 22 | 28312 |
| Sep | 23 | 30316 |
| Sep | 24 | 31301 |
| Sep | 25 | 38203 |
| Sep | 26 | 37504 |
| Sep | 27 | 39468 |
| Sep | 28 | 29656 |
| Sep | 29 | 29201 |
| Sep | 30 | 33431 |

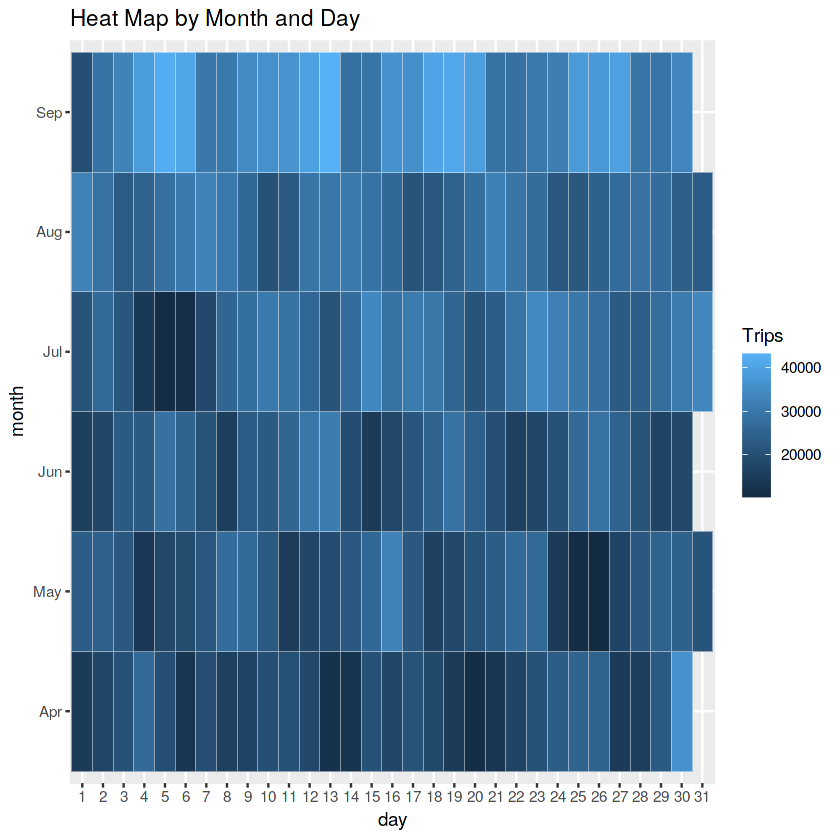
In [21]:

*# Plot a heatmap*

ggplot(month\_day\_data, aes(day, month, fill = Trips)) +

geom\_tile(color = "white") +

ggtitle("Heat Map by Month and Day")



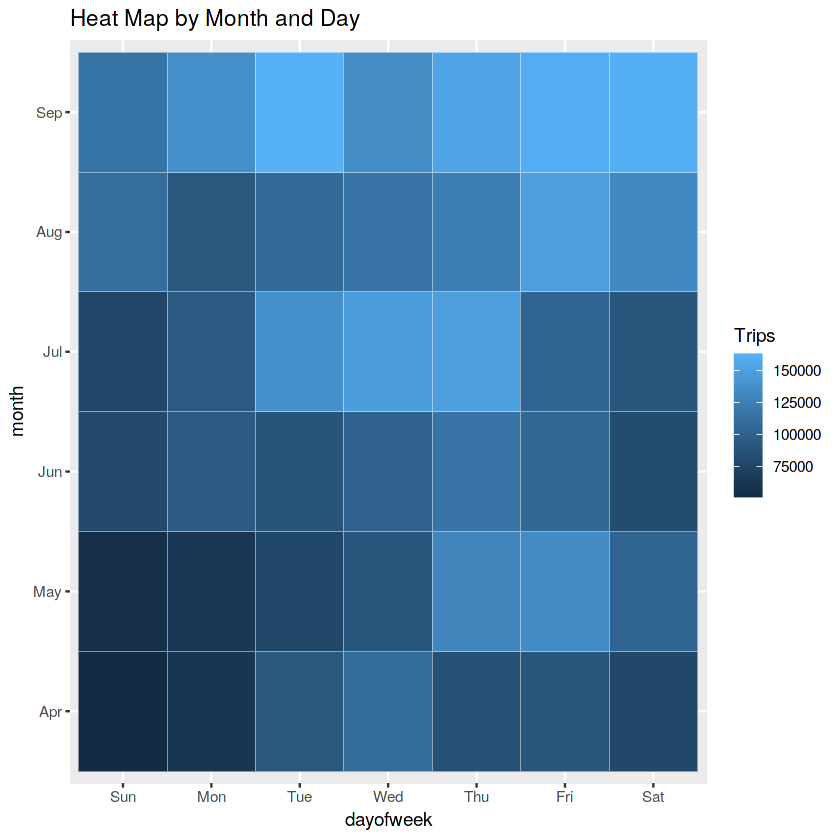
In [22]:

*# Plot a heatmap by day of the week and month*

ggplot(day\_month\_data, aes(dayofweek, month, fill = Trips)) +

geom\_tile(color = "white") +

ggtitle("Heat Map by Month and Day")



Creating a map visualization of rides in NYC

In [23]:

*# Set Map Constants*

min\_lat <- 40

max\_lat <- 40.91

min\_long <- -74.15

max\_long <- -73.7004

In [24]:

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

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

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

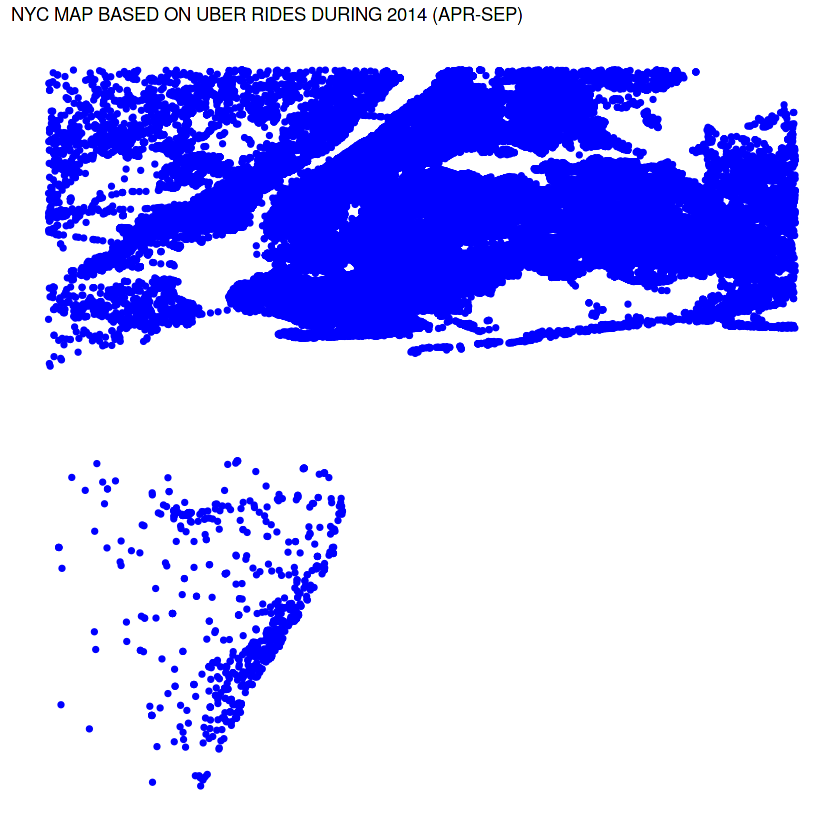
scale\_y\_continuous(limits=c(min\_lat, max\_lat)) +

theme\_map() +

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

Warning message:

“Removed 70180 rows containing missing values (geom\_point).”



In [ ]: