Railway Data Analysis

Malindi

2025-01-02

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.4.2
```

```
## Warning: package 'ggplot2' was built under R version 4.4.2
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(lubridate)

library(naniar)
```

```
## Warning: package 'naniar' was built under R version 4.4.2
```

File Setup

```
data <- read.csv("Data/railway.csv")

str(data)
```

```
## 'data.frame':    31653 obs. of  18 variables:
##  $ Transaction.ID     : chr  "da8a6ba8-b3dc-4677-b176" "b0cdd1b0-f214-4197-be53" "f3ba7a96-f7
13-40d9-9629" "b2471f11-4fe7-4c87-8ab4" ...
##  $ Date.of.Purchase   : chr  "2023-12-08" "2023-12-16" "2023-12-19" "2023-12-20" ...
##  $ Time.of.Purchase   : chr  "12:41:11" "11:23:01" "19:51:27" "23:00:36" ...
##  $ Purchase.Type      : chr  "Online" "Station" "Online" "Station" ...
##  $ Payment.Method     : chr  "Contactless" "Credit Card" "Credit Card" "Credit Card" ...
##  $ Railcard           : chr  "Adult" "Adult" "None" "None" ...
##  $ Ticket.Class       : chr  "Standard" "Standard" "Standard" "Standard" ...
##  $ Ticket.Type        : chr  "Advance" "Advance" "Advance" "Advance" ...
##  $ Price              : int  43 23 3 13 76 35 2 2 37 13 ...
##  $ Departure.Station  : chr  "London Paddington" "London Kings Cross" "Liverpool Lime Street"
"London Paddington" ...
##  $ Arrival.Destination: chr  "Liverpool Lime Street" "York" "Manchester Piccadilly" "Reading"
...
##  $ Date.of.Journey    : chr  "2024-01-01" "2024-01-01" "2024-01-02" "2024-01-01" ...
##  $ Departure.Time     : chr  "11:00:00" "09:45:00" "18:15:00" "21:30:00" ...
##  $ Arrival.Time       : chr  "13:30:00" "11:35:00" "18:45:00" "22:30:00" ...
##  $ Actual.Arrival.Time: chr  "13:30:00" "11:40:00" "18:45:00" "22:30:00" ...
##  $ Journey.Status     : chr  "On Time" "Delayed" "On Time" "On Time" ...
##  $ Reason.for.Delay   : chr  "" "Signal Failure" "" "" ...
##  $ Refund.Request     : chr  "No" "No" "No" "No" ...
```

```
head(data)
```

```
##            Transaction.ID Date.of.Purchase Time.of.Purchase Purchase.Type
## 1 da8a6ba8-b3dc-4677-b176       2023-12-08         12:41:11        Online
## 2 b0cdd1b0-f214-4197-be53       2023-12-16         11:23:01       Station
## 3 f3ba7a96-f713-40d9-9629       2023-12-19         19:51:27        Online
## 4 b2471f11-4fe7-4c87-8ab4       2023-12-20         23:00:36       Station
## 5 2be00b45-0762-485e-a7a3       2023-12-27         18:22:56        Online
## 6 4e1dcd88-3d95-44ef-99fa       2023-12-30         07:56:06        Online
##   Payment.Method Railcard Ticket.Class Ticket.Type Price     Departure.Station
## 1    Contactless    Adult     Standard     Advance    43     London Paddington
## 2    Credit Card    Adult     Standard     Advance    23    London Kings Cross
## 3    Credit Card     None     Standard     Advance     3 Liverpool Lime Street
## 4    Credit Card     None     Standard     Advance    13     London Paddington
## 5    Contactless     None     Standard     Advance    76 Liverpool Lime Street
## 6    Credit Card     None     Standard     Advance    35    London Kings Cross
##     Arrival.Destination Date.of.Journey Departure.Time Arrival.Time
## 1 Liverpool Lime Street      2024-01-01       11:00:00     13:30:00
## 2                 York      2024-01-01       09:45:00     11:35:00
## 3 Manchester Piccadilly      2024-01-02       18:15:00     18:45:00
## 4               Reading      2024-01-01       21:30:00     22:30:00
## 5         London Euston      2024-01-01       16:45:00     19:00:00
## 6                 York      2024-01-01       06:15:00     08:05:00
##   Actual.Arrival.Time Journey.Status Reason.for.Delay Refund.Request
## 1            13:30:00        On Time                            No
## 2            11:40:00        Delayed   Signal Failure           No
## 3            18:45:00        On Time                            No
## 4            22:30:00        On Time                            No
## 5            19:00:00        On Time                            No
## 6            08:05:00        On Time                            No
```

Section 1: Data Preparation and Cleaning

Validate Data Types

```
data$Date.of.Purchase <- ymd(data$Date.of.Purchase)
data$Date.of.Journey <- ymd(data$Date.of.Journey)
data$Purchase.Type <- as.factor(data$Purchase.Type)
data$Payment.Method <- as.factor(data$Payment.Method)
data$Ticket.Type <- as.factor(data$Ticket.Type)
data$Journey.Status <- as.factor(data$Journey.Status)
str(data)
```

```
## 'data.frame':    31653 obs. of  18 variables:
##  $ Transaction.ID     : chr  "da8a6ba8-b3dc-4677-b176" "b0cdd1b0-f214-4197-be53" "f3ba7a96-f7
13-40d9-9629" "b2471f11-4fe7-4c87-8ab4" ...
##  $ Date.of.Purchase   : Date, format: "2023-12-08" "2023-12-16" ...
##  $ Time.of.Purchase   : chr  "12:41:11" "11:23:01" "19:51:27" "23:00:36" ...
##  $ Purchase.Type      : Factor w/ 2 levels "Online","Station": 1 2 1 2 1 1 2 2 2 1 ...
##  $ Payment.Method     : Factor w/ 3 levels "Contactless",..: 1 2 2 2 1 2 2 1 2 2 ...
##  $ Railcard           : chr  "Adult" "Adult" "None" "None" ...
##  $ Ticket.Class       : chr  "Standard" "Standard" "Standard" "Standard" ...
##  $ Ticket.Type        : Factor w/ 3 levels "Advance","Anytime",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Price              : int  43 23 3 13 76 35 2 2 37 13 ...
##  $ Departure.Station  : chr  "London Paddington" "London Kings Cross" "Liverpool Lime Street"
"London Paddington" ...
##  $ Arrival.Destination: chr  "Liverpool Lime Street" "York" "Manchester Piccadilly" "Reading"
...
##  $ Date.of.Journey    : Date, format: "2024-01-01" "2024-01-01" ...
##  $ Departure.Time     : chr  "11:00:00" "09:45:00" "18:15:00" "21:30:00" ...
##  $ Arrival.Time       : chr  "13:30:00" "11:35:00" "18:45:00" "22:30:00" ...
##  $ Actual.Arrival.Time: chr  "13:30:00" "11:40:00" "18:45:00" "22:30:00" ...
##  $ Journey.Status     : Factor w/ 3 levels "Cancelled","Delayed",..: 3 2 3 3 3 3 3 3 2 3 ...
##  $ Reason.for.Delay   : chr  "" "Signal Failure" "" "" ...
##  $ Refund.Request     : chr  "No" "No" "No" "No" ...
```

Check Missing Values

```
missing_summary <- sapply(data, function(x) sum(is.na(x) | x == ""))
print(missing_summary)
```

```
##      Transaction.ID    Date.of.Purchase    Time.of.Purchase       Purchase.Type
##                   0                  NA                   0                   0
##      Payment.Method            Railcard        Ticket.Class         Ticket.Type
##                   0                   0                   0                   0
##               Price   Departure.Station Arrival.Destination     Date.of.Journey
##                   0                   0                   0                  NA
##      Departure.Time        Arrival.Time Actual.Arrival.Time      Journey.Status
##                   0                   0                1880                   0
##    Reason.for.Delay      Refund.Request
##               27481                   0
```

Handling Missing Values

```
numeric_cols <- sapply(data, is.numeric)
data[numeric_cols] <- lapply(data[numeric_cols], function(x) ifelse(is.na(x), mean(x, na.rm = TR
UE), x))

#For numeric columns, missing values are replaced with the mean of that column because mean repr
esents the central value of the data.


categorical_cols <- sapply(data, is.character)
data[categorical_cols] <- lapply(data[categorical_cols], function(x) ifelse(x == "" | is.na(x),
"Unknown", x))

#For categorical columns, missing or empty values are replaced with the string "Unknown" because
removing rows with missing categorical values can lead to a loss of valuable information.
```

```
missing_summary_after <- colSums(is.na(data))
print(missing_summary_after)
```

```
##      Transaction.ID    Date.of.Purchase    Time.of.Purchase       Purchase.Type
##                   0                   0                   0                   0
##      Payment.Method            Railcard        Ticket.Class         Ticket.Type
##                   0                   0                   0                   0
##               Price   Departure.Station Arrival.Destination     Date.of.Journey
##                   0                   0                   0                   0
##      Departure.Time        Arrival.Time Actual.Arrival.Time      Journey.Status
##                   0                   0                   0                   0
##    Reason.for.Delay      Refund.Request
##                   0                   0
```

Section 2: Understanding Key Variables

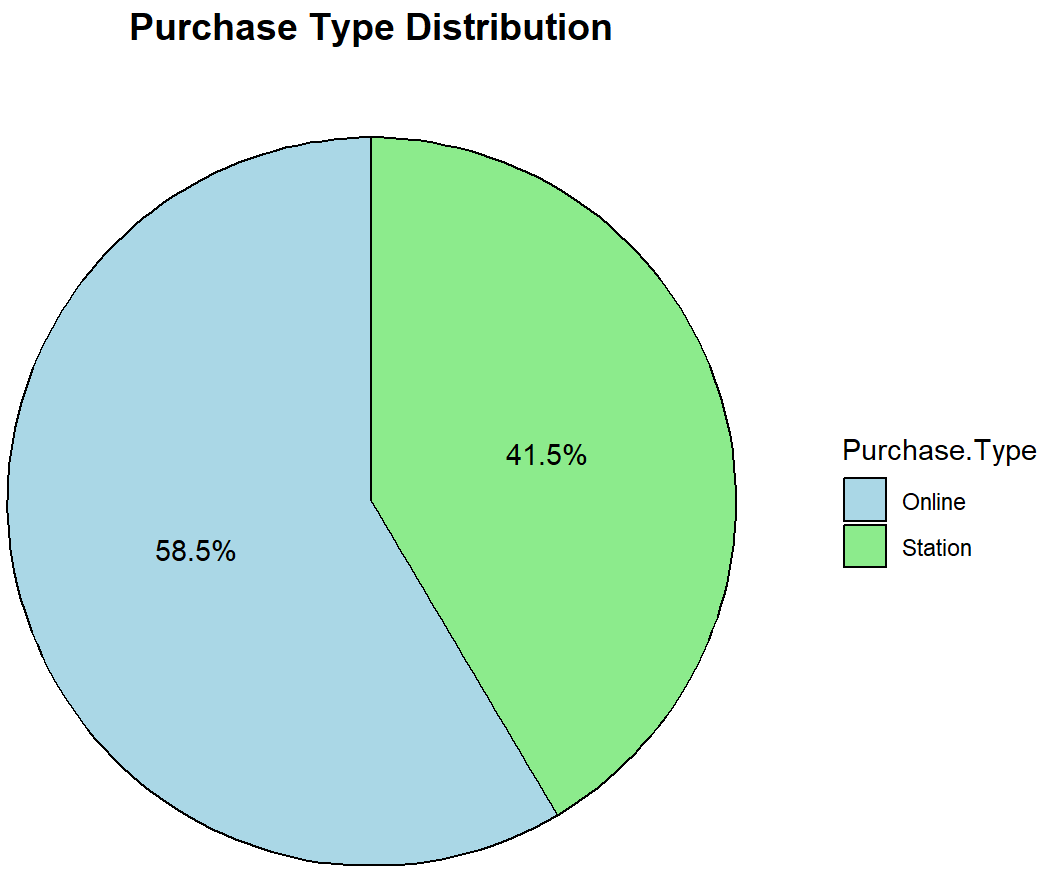
Summary Table for Purchase Type

```
purchase <- data %>%
  group_by(Purchase.Type) %>%
  summarise(Count = n(),
            Percentage = n() / nrow(data) * 100
            )
print(purchase)
```

```
## # A tibble: 2 × 3
##   Purchase.Type Count Percentage
##   <fct>         <int>      <dbl>
## 1 Online        18521       58.5
## 2 Station       13132       41.5
```

Visualization for Purchase Type based on the summary table

```
ggplot(purchase, aes(x = "", y = Percentage, fill = Purchase.Type)) +
  geom_bar(stat = "identity", width = 1,color = "black") +
  coord_polar(theta = "y") +
  labs(title = "Purchase Type Distribution", x = NULL, y = NULL) +
  theme_void() +
  theme(axis.text.x = element_blank(),
        plot.title = element_text(face = "bold", size = 14, hjust = 0.5)) +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")),
            position = position_stack(vjust = 0.5))+
  scale_fill_manual(values = c("lightblue", "lightgreen"))
```



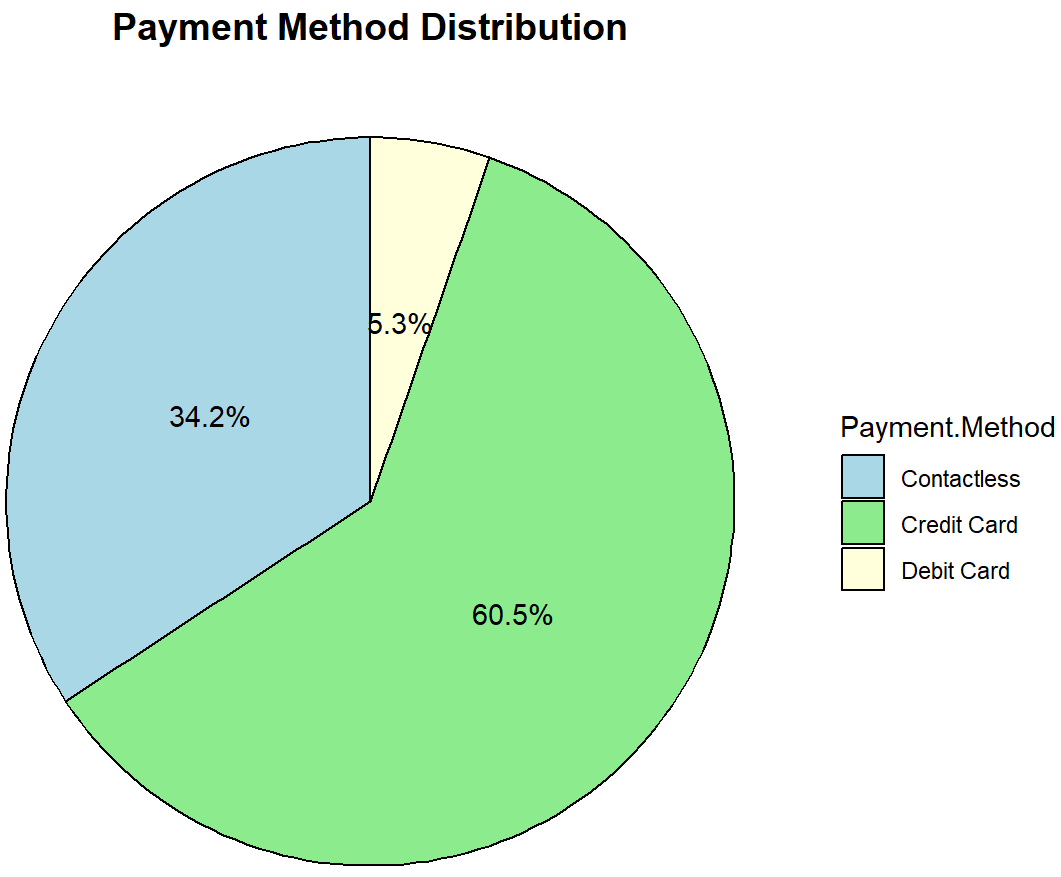
Summary Table for Payment Method

```
payment <- data %>%
  group_by(Payment.Method) %>%
  summarise(Count = n(),
            Percentage = n() / nrow(data) * 100,
            )
print(payment)
```

```
## # A tibble: 3 × 3
##   Payment.Method Count Percentage
##   <fct>          <int>      <dbl>
## 1 Contactless    10834      34.2
## 2 Credit Card    19136      60.5
## 3 Debit Card      1683       5.32
```

Visualization for Payment Method

```
ggplot(payment, aes(x = "", y = Percentage, fill = Payment.Method)) +
  geom_bar(stat = "identity", width = 1,color = "black") +
  coord_polar(theta = "y") +
  labs(title = "Payment Method Distribution", x = NULL, y = NULL) +
  theme_void() +
  theme(axis.text.x = element_blank(),
        plot.title = element_text(face = "bold", size = 14, hjust = 0.5)) +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")),
            position = position_stack(vjust = 0.5))+
  scale_fill_manual(values = c("lightblue", "lightgreen","lightyellow"))
```



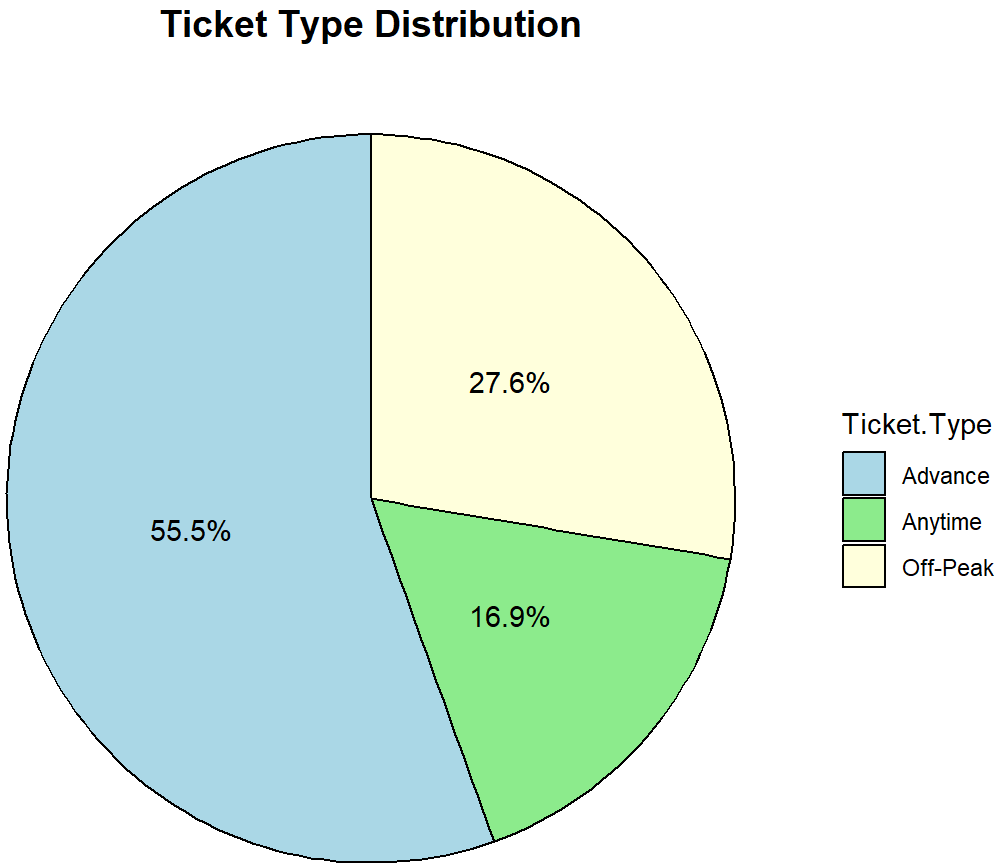
Summary Table for Ticket Type

```
ticket <- data %>%
  group_by(Ticket.Type) %>%
  summarise(Count = n(),
            Percentage = n() / nrow(data) * 100,
            )
print(ticket)
```

```
## # A tibble: 3 × 3
##   Ticket.Type Count Percentage
##   <fct>       <int>      <dbl>
## 1 Advance     17561       55.5
## 2 Anytime      5340       16.9
## 3 Off-Peak     8752       27.6
```

Visualization for Ticket Type Summery Table

```
ggplot(ticket, aes(x = "", y = Percentage, fill = Ticket.Type)) +
  geom_bar(stat = "identity", width = 1,color = "black") +
  coord_polar(theta = "y") +
  labs(title = "Ticket Type Distribution", x = NULL, y = NULL) +
  theme_void() +
  theme(axis.text.x = element_blank(),
        plot.title = element_text(face = "bold", size = 14, hjust = 0.5)) +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")),
            position = position_stack(vjust = 0.5))+
  scale_fill_manual(values = c("lightblue", "lightgreen", "lightyellow"))
```



Section 3: Delayed Journeys Analysis

```
delayed_data <- data %>%
  filter(Journey.Status == "Delayed")
```

Create new column for weekdays

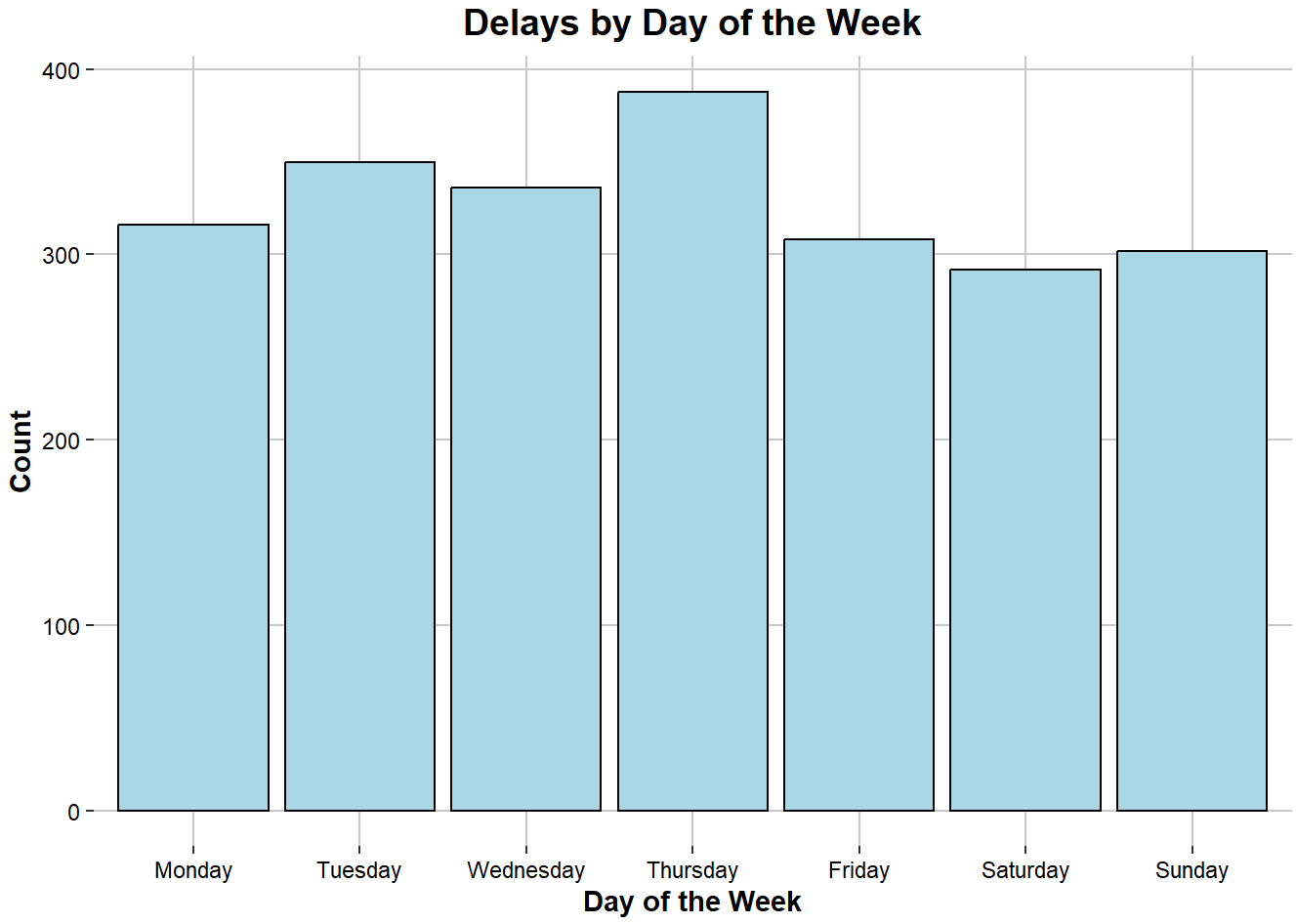
```
delayed_data <- delayed_data %>%
  mutate(Weekday = weekdays(Date.of.Journey))

weekday_delay <- delayed_data %>%
  group_by(Weekday) %>%
  summarise(Count = n())
```

Visualization for Delays by Weekday

```
weekday_delay$Weekday <- factor(weekday_delay$Weekday, levels = c("Monday", "Tuesday", "Wednesda
y", "Thursday", "Friday", "Saturday", "Sunday"))

ggplot(weekday_delay, aes(x = Weekday, y = Count)) +
  geom_bar(stat = "identity", fill = "lightblue", color = "black") +
  labs(title = "Delays by Day of the Week", x = "Day of the Week", y = "Count") +
  theme(
    plot.title = element_text(face = "bold", size = 14, hjust = 0.5),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    panel.grid.major = element_line(color = "gray80"),
    panel.grid.minor = element_blank(),
    panel.background = element_rect(fill = "white")
  )
```

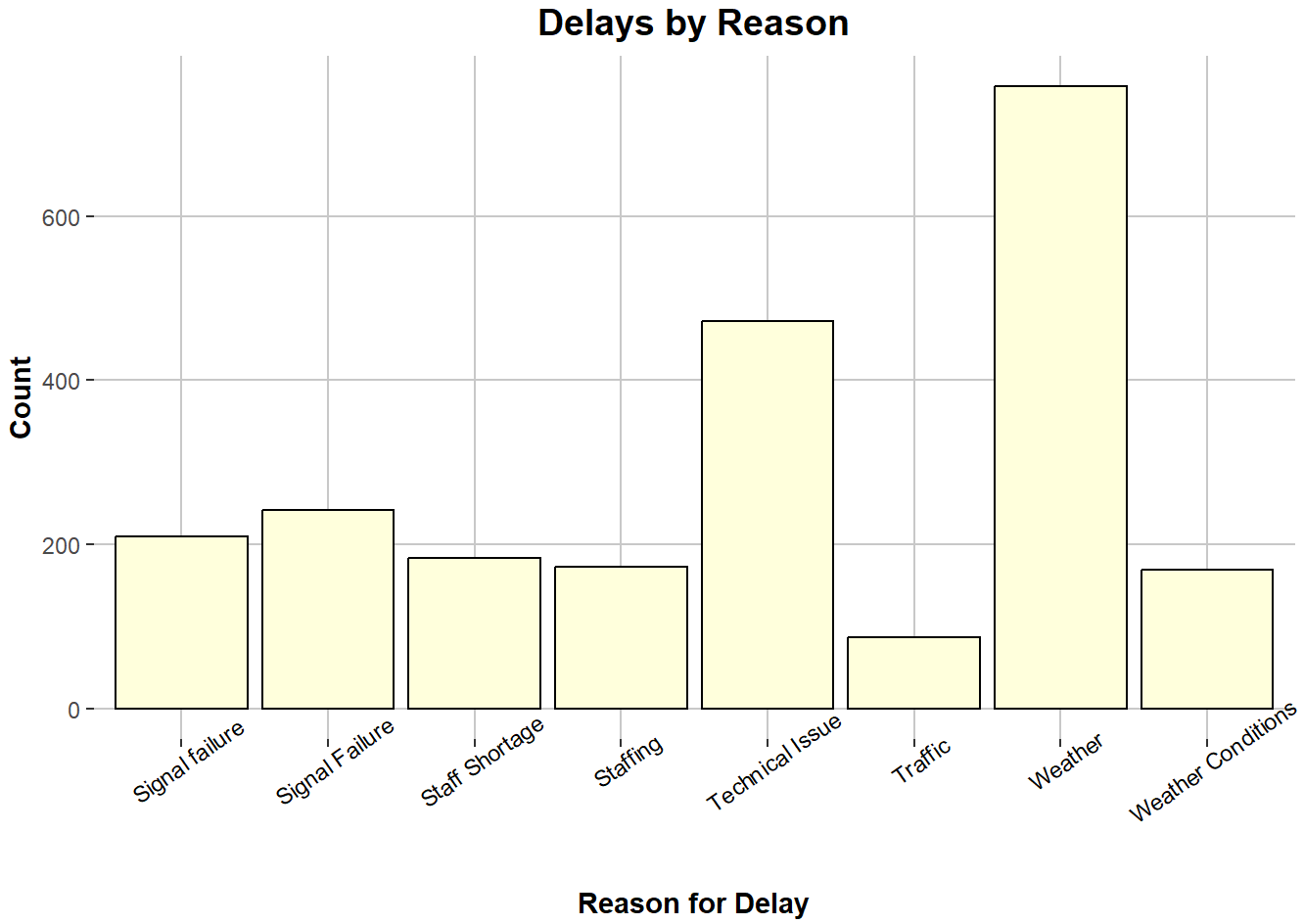


Summery table for delay reason

```
delay_reason <- delayed_data %>%
  group_by(Reason.for.Delay) %>%
  summarise(Count = n())
```

Visualization for Delays by Reason

```
ggplot(delayed_data, aes(x = Reason.for.Delay)) +
  geom_bar(fill = "lightyellow",color = "black") +
  labs(title = "Delays by Reason", x = "Reason for Delay", y = "Count")+
  theme(
    plot.title = element_text(face = "bold", size = 14, hjust = 0.5),
    axis.title = element_text(face = "bold"),
    axis.text.x = element_text(color = "black",angle = 35),
    panel.grid.major = element_line(color = "gray80"),
    panel.grid.minor = element_blank(),
    panel.background = element_rect(fill = "white")
  )
```




```
heatmap_data <- delayed_data %>%
  group_by(Weekday, Reason.for.Delay) %>%
  summarise(Count = n(), .groups = "drop")
```

Visualization for Reason for Delay by Weekday

```
heatmap_data$Weekday <- factor(heatmap_data$Weekday,
                               levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Frida
y", "Saturday", "Sunday"))

ggplot(heatmap_data, aes(x = Weekday, y = Reason.for.Delay, fill = Count)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "lightblue", high = "red") +
  labs(title = "Delays by Weekday and Reason",
       x = "Weekday",
       y = "Reason for Delay",
       fill = "Count") +
  theme(
    plot.title = element_text(face = "bold", size = 14, hjust = 0.5),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    panel.background = element_rect(fill = "white")
  )
```