Homerwork 1

SAKURAKO MIZUTANI

5/14/23

# Data Manipulation

## Problem 1: Use logical operators to find flights that:

- Had an arrival delay of two or more hours (\> 120 minutes)  
- Flew to Houston (IAH or HOU)  
- Were operated by United (`UA`), American (`AA`), or Delta (`DL`)  
- Departed in summer (July, August, and September)  
- Arrived more than two hours late, but didn't leave late  
- Were delayed by at least an hour, but made up over 30 minutes in flight

# Had an arrival delay of two or more hours (> 120 minutes)  
flights %>%  
 filter(arr\_delay > 120)

# A tibble: 10,034 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 1 1 811 630 101 1047 830  
 2 2013 1 1 848 1835 853 1001 1950  
 3 2013 1 1 957 733 144 1056 853  
 4 2013 1 1 1114 900 134 1447 1222  
 5 2013 1 1 1505 1310 115 1638 1431  
 6 2013 1 1 1525 1340 105 1831 1626  
 7 2013 1 1 1549 1445 64 1912 1656  
 8 2013 1 1 1558 1359 119 1718 1515  
 9 2013 1 1 1732 1630 62 2028 1825  
10 2013 1 1 1803 1620 103 2008 1750  
# ℹ 10,024 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

# Flew to Houston (IAH or HOU)  
flights %>%  
 filter(dest %in% c("IAH", "HOU"))

# A tibble: 9,313 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 1 1 517 515 2 830 819  
 2 2013 1 1 533 529 4 850 830  
 3 2013 1 1 623 627 -4 933 932  
 4 2013 1 1 728 732 -4 1041 1038  
 5 2013 1 1 739 739 0 1104 1038  
 6 2013 1 1 908 908 0 1228 1219  
 7 2013 1 1 1028 1026 2 1350 1339  
 8 2013 1 1 1044 1045 -1 1352 1351  
 9 2013 1 1 1114 900 134 1447 1222  
10 2013 1 1 1205 1200 5 1503 1505  
# ℹ 9,303 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

# Were operated by United (`UA`), American (`AA`), or Delta (`DL`)  
flights %>%  
 filter(carrier %in% c("UA", "AA", "DL"))

# A tibble: 139,504 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 1 1 517 515 2 830 819  
 2 2013 1 1 533 529 4 850 830  
 3 2013 1 1 542 540 2 923 850  
 4 2013 1 1 554 600 -6 812 837  
 5 2013 1 1 554 558 -4 740 728  
 6 2013 1 1 558 600 -2 753 745  
 7 2013 1 1 558 600 -2 924 917  
 8 2013 1 1 558 600 -2 923 937  
 9 2013 1 1 559 600 -1 941 910  
10 2013 1 1 559 600 -1 854 902  
# ℹ 139,494 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

# Departed in summer (July, August, and September)  
flights %>%  
 filter(month %in% c("7", "8", "9"))

# A tibble: 86,326 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 7 1 1 2029 212 236 2359  
 2 2013 7 1 2 2359 3 344 344  
 3 2013 7 1 29 2245 104 151 1  
 4 2013 7 1 43 2130 193 322 14  
 5 2013 7 1 44 2150 174 300 100  
 6 2013 7 1 46 2051 235 304 2358  
 7 2013 7 1 48 2001 287 308 2305  
 8 2013 7 1 58 2155 183 335 43  
 9 2013 7 1 100 2146 194 327 30  
10 2013 7 1 100 2245 135 337 135  
# ℹ 86,316 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

# Arrived more than two hours late, but didn't leave late  
flights %>%  
 filter(arr\_delay > 120 & dep\_delay <= 0)

# A tibble: 29 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 1 27 1419 1420 -1 1754 1550  
 2 2013 10 7 1350 1350 0 1736 1526  
 3 2013 10 7 1357 1359 -2 1858 1654  
 4 2013 10 16 657 700 -3 1258 1056  
 5 2013 11 1 658 700 -2 1329 1015  
 6 2013 3 18 1844 1847 -3 39 2219  
 7 2013 4 17 1635 1640 -5 2049 1845  
 8 2013 4 18 558 600 -2 1149 850  
 9 2013 4 18 655 700 -5 1213 950  
10 2013 5 22 1827 1830 -3 2217 2010  
# ℹ 19 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

# Were delayed by at least an hour, but made up over 30 minutes in flight  
flights %>%  
 filter(arr\_delay >= 60 & arr\_delay - dep\_delay < 30)

# A tibble: 22,881 × 19  
 year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
 <int> <int> <int> <int> <int> <dbl> <int> <int>  
 1 2013 1 1 848 1835 853 1001 1950  
 2 2013 1 1 957 733 144 1056 853  
 3 2013 1 1 1114 900 134 1447 1222  
 4 2013 1 1 1120 944 96 1331 1213  
 5 2013 1 1 1255 1200 55 1451 1330  
 6 2013 1 1 1301 1150 71 1518 1345  
 7 2013 1 1 1337 1220 77 1649 1531  
 8 2013 1 1 1356 1259 57 1538 1438  
 9 2013 1 1 1411 1315 56 1717 1611  
10 2013 1 1 1428 1329 59 1803 1640  
# ℹ 22,871 more rows  
# ℹ 11 more variables: arr\_delay <dbl>, carrier <chr>, flight <int>,  
# tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
# hour <dbl>, minute <dbl>, time\_hour <dttm>

## Problem 2: What months had the highest and lowest proportion of cancelled flights? Interpret any seasonal patterns. To determine if a flight was cancelled use the following code

flights %>%   
 filter(is.na(dep\_time))

# What months had the highest and lowest % of cancelled flights?  
flights %>%  
 #group the data based on the month  
 group\_by(month) %>%  
 summarize(total\_flights = n(),  
# avg\_cancel = cancel/n(),  
 avg\_dep\_delay = mean(dep\_delay, na.rm = TRUE),   
 avg\_arr\_delay = mean(arr\_delay, na.rm = TRUE))

# A tibble: 12 × 4  
 month total\_flights avg\_dep\_delay avg\_arr\_delay  
 <int> <int> <dbl> <dbl>  
 1 1 27004 10.0 6.13   
 2 2 24951 10.8 5.61   
 3 3 28834 13.2 5.81   
 4 4 28330 13.9 11.2   
 5 5 28796 13.0 3.52   
 6 6 28243 20.8 16.5   
 7 7 29425 21.7 16.7   
 8 8 29327 12.6 6.04   
 9 9 27574 6.72 -4.02   
10 10 28889 6.24 -0.167  
11 11 27268 5.44 0.461  
12 12 28135 16.6 14.9

#In summer, delay tends to be larger than in winter

## Problem 3: What plane (specified by the tailnum variable) traveled the most times from New York City airports in 2013? Please left\_join() the resulting table with the table planes (also included in the nycflights13 package).

For the plane with the greatest number of flights and that had more than 50 seats, please create a table where it flew to during 2013.

# Get the tailnum of the plane that traveled the most in 2013  
tailnum\_most\_flights <- flights %>%  
#Group the dataset by flight numbers  
 group\_by(tailnum) %>%  
 summarize(n = n()) %>%  
#Connect the datasets based on flight numbers  
 left\_join(planes, by = "tailnum") %>%  
#For the plane that had more than 50 seats  
 filter(seats > 50) %>%  
#For the plane with the greatest number of flights  
 arrange(desc(n)) %>%  
 head(1) %>%  
 pull(tailnum)   
  
# Get the destinations of the most frequently flown plane in 2013  
tailnum\_table <- flights %>%  
 group\_by(tailnum) %>%  
 summarize(total\_flights = n(),  
 avg\_dep\_delay = mean(dep\_delay, na.rm = TRUE),  
 avg\_arr\_delay = mean(arr\_delay, na.rm = TRUE),  
 dist = sum(distance, na.rm = TRUE)) %>%  
 left\_join(planes, by = "tailnum") %>%  
 filter(tailnum == tailnum\_most\_flights)  
  
tailnum\_table

# A tibble: 1 × 13  
 tailnum total\_flights avg\_dep\_delay avg\_arr\_delay dist year type   
 <chr> <int> <dbl> <dbl> <dbl> <int> <chr>   
1 N328AA 393 7.57 -3.52 939101 1986 Fixed wing mul…  
# ℹ 6 more variables: manufacturer <chr>, model <chr>, engines <int>,  
# seats <int>, speed <int>, engine <chr>

#N328AA

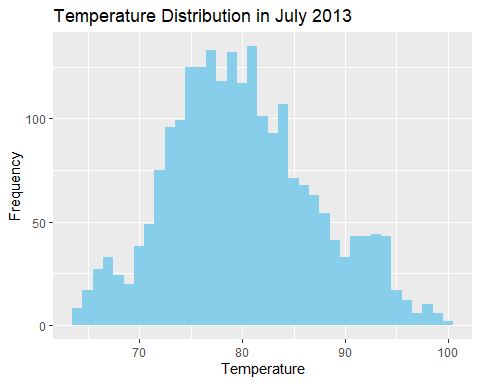
## Problem 4: The nycflights13 package includes a table (weather) that describes the weather during 2013. Use that table to answer the following questions:

- What is the distribution of temperature (`temp`) in July 2013? Identify any important outliers in terms of the `wind\_speed` variable.  
- What is the relationship between `dewp` and `humid`?  
- What is the relationship between `precip` and `visib`?

# Check what "weather" contains  
weather %>%  
 head()

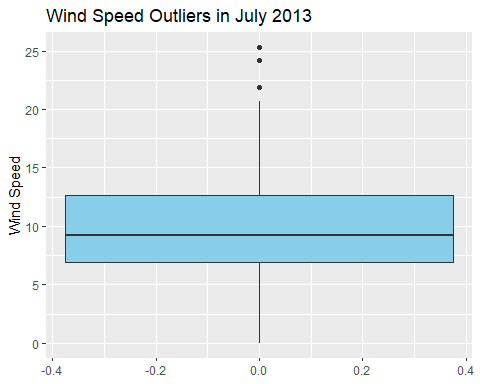
# A tibble: 6 × 15  
 origin year month day hour temp dewp humid wind\_dir wind\_speed wind\_gust  
 <chr> <int> <int> <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 EWR 2013 1 1 1 39.0 26.1 59.4 270 10.4 NA  
2 EWR 2013 1 1 2 39.0 27.0 61.6 250 8.06 NA  
3 EWR 2013 1 1 3 39.0 28.0 64.4 240 11.5 NA  
4 EWR 2013 1 1 4 39.9 28.0 62.2 250 12.7 NA  
5 EWR 2013 1 1 5 39.0 28.0 64.4 260 12.7 NA  
6 EWR 2013 1 1 6 37.9 28.0 67.2 240 11.5 NA  
# ℹ 4 more variables: precip <dbl>, pressure <dbl>, visib <dbl>,  
# time\_hour <dttm>

### Distribution of temperature in July 2013 ###  
# Filter for July 2013  
july\_weather <- weather %>%  
 filter(month == 7, year == 2013)  
  
# Create a histogram for temperature  
ggplot(july\_weather, aes(temp)) +  
 geom\_histogram(binwidth = 1, fill = "skyblue") +  
 labs(x = "Temperature", y = "Frequency", title = "Temperature Distribution in July 2013")



# Create a boxplot for wind speed to identify outliers  
ggplot(july\_weather, aes(y = wind\_speed)) +  
 geom\_boxplot(fill = "skyblue") +  
 labs(y = "Wind Speed", title = "Wind Speed Outliers in July 2013")

Warning: Removed 2 rows containing non-finite values (`stat\_boxplot()`).

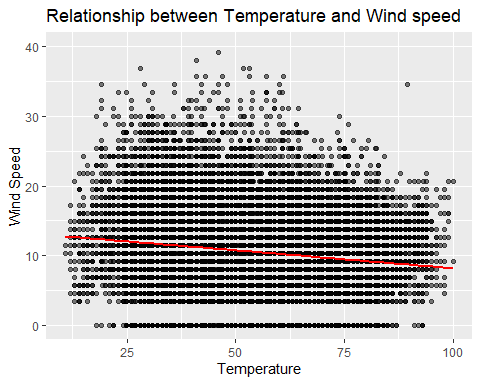


# Scatter plot to see the relationship between temp and wind\_speed  
ggplot(weather, aes(x = temp, y = wind\_speed)) +  
 geom\_point(alpha = 0.5) +  
 geom\_smooth(method = "lm", se = FALSE, color = "red") +  
 scale\_y\_continuous(limits = c(0,40)) +  
 labs(x = "Temperature", y = "Wind Speed", title = "Relationship between Temperature and Wind speed")

`geom\_smooth()` using formula = 'y ~ x'

Warning: Removed 10 rows containing non-finite values (`stat\_smooth()`).

Warning: Removed 10 rows containing missing values (`geom\_point()`).

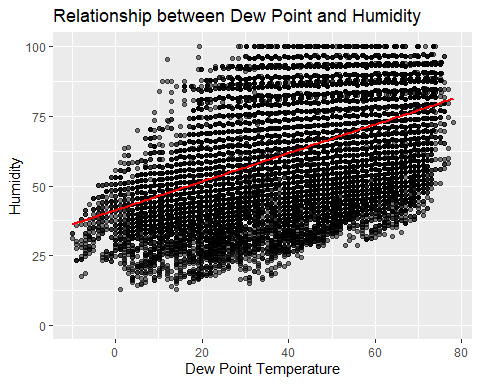


##A. There is a negative correlation between temperature and wind speed  
  
##------------------------------------##  
### Relationship betweeen "dewp" and "humid" ###  
ggplot(weather, aes(x = dewp, y = humid)) +  
 geom\_point(alpha = 0.5) +  
 geom\_smooth(method = "lm", se = FALSE, color = "red") +   
 scale\_y\_continuous(limits = c(0,100)) +  
 labs(x = "Dew Point Temperature", y = "Humidity", title = "Relationship between Dew Point and Humidity")

`geom\_smooth()` using formula = 'y ~ x'

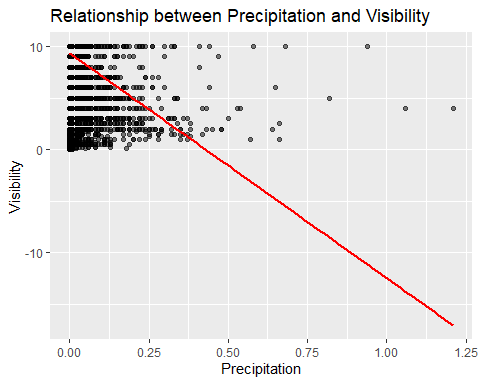
Warning: Removed 1 rows containing non-finite values (`stat\_smooth()`).

Warning: Removed 1 rows containing missing values (`geom\_point()`).



#A. There is a positive correlation between Temperature and Humidity  
  
##------------------------------------##  
###the relationship between `precip` and `visib`###  
ggplot(weather, aes(x = precip, y = visib)) +  
 geom\_point(alpha = 0.5) +  
 geom\_smooth(method = "lm", se = FALSE, color = "red") +  
 labs(x = "Precipitation", y = "Visibility", title = "Relationship between Precipitation and Visibility")

`geom\_smooth()` using formula = 'y ~ x'



#A. There is a negative correlation between precipitation and visibility

## Problem 5: Use the flights and planes tables to answer the following questions:

- How many planes have a missing date of manufacture?  
- What are the five most common manufacturers?  
- Has the distribution of manufacturer changed over time as reflected by the airplanes flying from NYC in 2013? (Hint: you may need to use case\_when() to recode the manufacturer name and collapse rare vendors into a category called Other.)

##planes those who have a missing date of manufacture  
# Count missing manufacture dates  
missing\_manufacture\_dates <- planes %>%  
 filter(is.na(year))  
  
# Print number of planes with missing manufacture dates  
print(paste("Number of planes with missing date of manufacture: ", nrow(missing\_manufacture\_dates)))

[1] "Number of planes with missing date of manufacture: 70"

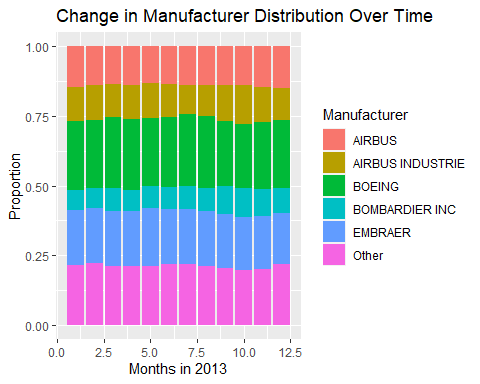
#A. Number of planes with missing date of manufacture: 70  
  
##----------------------------------------------------##  
##the five most common manufacturers  
# Count manufacturers and sort  
top\_manufacturers <- planes %>%  
 count(manufacturer) %>%  
 arrange(desc(n)) %>%  
 head(5)  
  
# Print five most common manufacturers  
print("Five most common manufacturers:")

[1] "Five most common manufacturers:"

print(top\_manufacturers)

# A tibble: 5 × 2  
 manufacturer n  
 <chr> <int>  
1 BOEING 1630  
2 AIRBUS INDUSTRIE 400  
3 BOMBARDIER INC 368  
4 AIRBUS 336  
5 EMBRAER 299

#BOEING 1630  
#AIRBUS INDUSTRIE 400   
#BOMBARDIER INC 368   
#AIRBUS 336   
#EMBRAER 299   
  
##----------------------------------------------------##  
##Distribution of manufacturer over time as reflected by the airplanes flying from NYC in 2013  
  
# Join flights and planes  
flights\_planes <- flights %>%  
 left\_join(planes, by = "tailnum")  
  
# Recode manufacturers  
flights\_planes <- flights\_planes %>%  
 mutate(manufacturer = if\_else(manufacturer %in% top\_manufacturers$manufacturer, manufacturer, "Other"))  
  
# Plot manufacturer distribution by year of manufacture  
flights\_planes %>%  
 ggplot(aes(x = month, fill = manufacturer)) +  
 geom\_bar(position = "fill", na.rm = TRUE) +  
 labs(x = "Months in 2013", y = "Proportion", fill = "Manufacturer", title = "Change in Manufacturer Distribution Over Time")



#There was a small change in manufacturing distribution in 2013, but overall the distributions stayed almost the same.

## Problem 6: Use the flights and planes tables to answer the following questions:

- What is the oldest plane (specified by the tailnum variable) that flew from New York City airports in 2013?  
- How many airplanes that flew from New York City are included in the planes table?

##The oldest plane from NYC in 2013  
# Join flights and planes  
flights\_planes <- flights %>%  
 left\_join(planes, by = "tailnum")  
  
# Get the oldest plane based on the tailnum  
oldest\_plane <- flights\_planes %>%  
# filter(origin == "JFK" | origin == "LGA" | origin == "EWR") %>% # Filter flights from NYC  
 filter(!is.na(tailnum)) %>% # Filter out missing data  
 arrange(tailnum) %>% # Sort by tailnum  
 slice(1) # Select the first row  
  
# Print the tail number of the oldest plane  
print(paste("Tail number of the oldest plane: ", oldest\_plane$tailnum))

[1] "Tail number of the oldest plane: D942DN"

#Tail number of the oldest plane: D942DN  
  
##------------------------------------------------------##No. of airplanes from NYC  
#unique\_planes\_in\_flights <- flights %>%  
  
 # Count the number of unique planes in the flights data  
 # summarise(n = n\_distinct(tailnum)) %>%  
  
# Print the results  
 #print(paste("Number of unique airplanes in the flights data: ", unique\_planes\_in\_flights$n))  
  
#A. 4044

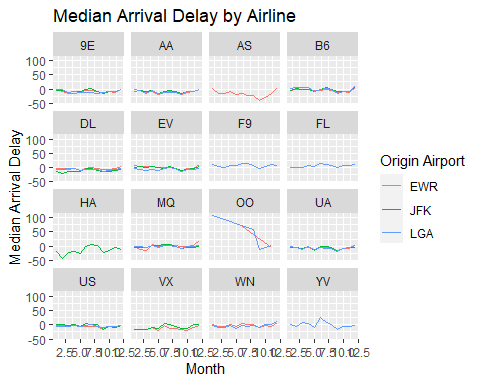
## Problem 7: Use the nycflights13 to answer the following questions:

- What is the median arrival delay on a month-by-month basis in each airport?  
- For each airline, plot the median arrival delay for each month and origin airport.

##the median arrival delay on a month-by-month basis in each airport  
  
# Calculate median arrival delay by month and origin airport  
median\_arrival\_delay <- flights %>%  
 group\_by(month, origin) %>%  
 summarise(median\_delay = median(arr\_delay, na.rm = TRUE), .groups = 'drop')  
  
# Print the median arrival delay  
print(median\_arrival\_delay)

# A tibble: 36 × 3  
 month origin median\_delay  
 <int> <chr> <dbl>  
 1 1 EWR 0  
 2 1 JFK -7  
 3 1 LGA -4  
 4 2 EWR -2  
 5 2 JFK -5  
 6 2 LGA -4  
 7 3 EWR -4  
 8 3 JFK -7  
 9 3 LGA -7  
10 4 EWR -1  
# ℹ 26 more rows

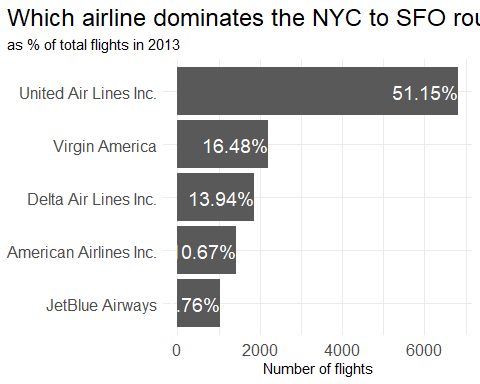
##------------------------------------------##  
##the median arrival delay for each airline for each month and origin  
  
# Calculate median arrival delay by airline, month, and origin airport  
median\_arrival\_delay\_airline <- flights %>%  
 group\_by(carrier, month, origin) %>%  
 summarise(median\_delay = median(arr\_delay, na.rm = TRUE), .groups = 'drop')  
  
# Plot median arrival delay by airline  
ggplot(median\_arrival\_delay\_airline, aes(x = month, y = median\_delay, color = origin)) +  
 geom\_line() +  
 facet\_wrap(~carrier) +  
 labs(x = "Month", y = "Median Arrival Delay", color = "Origin Airport", title = "Median Arrival Delay by Airline")



## Problem 8: Let’s take a closer look at what carriers service the route to San Francisco International (SFO). Join the flights and airlines tables and count which airlines flew the most to SFO. Produce a new dataframe, fly\_into\_sfo that contains three variables: the name of the airline, e.g., United Air Lines Inc. not UA, the count (number) of times it flew to SFO, and the percent of the trips that that particular airline flew to SFO.

And here is some bonus ggplot code to plot your dataframe

# Join flights and airlines  
flights\_airlines <- flights %>%  
 left\_join(airlines, by = "carrier")  
  
# Filter for flights to SFO and count flights by airline  
fly\_into\_sfo <- flights\_airlines %>%  
 filter(dest == "SFO") %>%  
 count(name) %>%  
 rename(count = n) %>%  
 mutate(percent = count / sum(count) \* 100)  
  
# plot the data  
fly\_into\_sfo %>%   
 mutate(name = fct\_reorder(name, count)) %>%   
 ggplot() +  
 aes(x = count, y = name) +  
  
 # a simple bar/column plot  
 geom\_col() +  
  
 # add labels, so each bar shows the % of total flights   
 geom\_text(aes(label = sprintf("%.2f%%", percent)),   
 hjust = 1,   
 colour = "white",   
 size = 5) +  
  
 # add labels to help our audience   
 labs(title="Which airline dominates the NYC to SFO route?",   
 subtitle = "as % of total flights in 2013",  
 x= "Number of flights",  
 y= NULL) +  
 theme\_minimal() +   
  
# change the theme   
 theme(  
  
# so title is left-aligned  
 plot.title.position = "plot",  
  
# text in axes appears larger   
 axis.text = element\_text(size=12),  
  
# title text is bigger  
 plot.title = element\_text(size=18)  
 ) +  
  
# add one final layer of NULL, so if you comment out any lines  
 NULL



#A. UA 51.15%, VA 16.48%, DA 13.94%, ...

## Problem 9: Let’s take a look at cancellations of flights to SFO. We create a new dataframe cancellations as follows

cancellations <- flights %>%   
   
 # just filter for destination == 'SFO'  
 filter(dest == 'SFO') %>%   
   
 # a cancelled flight is one with no `dep\_time`   
 filter(is.na(dep\_time))

I want you to think how we would organise our data manipulation to create the following plot. No need to write the code, just explain in words how you would go about it.



1. **Filtering the Data:** Start by filtering the **cancellations** dataframe for flights originating from EWR and JFK. This can be done using the **filter()** function in the **dplyr** package. In this case, you will need to use the **origin** variable.
2. **Grouping the Data:** Next, group the data by month, carrier, and origin airport using the **group\_by()** function. This will allow you to calculate the number of cancellations for each month, carrier, and origin airport.
3. **Summarizing the Data:** After grouping, summarize the data to count the number of cancellations for each group. This can be done using the **summarise()** function. You would create a new variable, e.g., **num\_cancellations**, that represents the number of cancellations.
4. **Creating the Plot:** With the summarized data, you’re now ready to create the histogram. You would use the **ggplot()** function from the **ggplot2** package to initialize the plot, and **geom\_histogram()** or **geom\_bar()** to add the histogram bars. The **aes()** function will be used to map the variables to the plot aesthetics. You would map the **num\_cancellations** variable to the x-axis, and the **origin** variable to the fill aesthetic to create a stacked histogram that compares EWR and JFK.
5. **Faceting the Plot:** To compare the cancellations by carrier and month, you would use the **facet\_grid()** or **facet\_wrap()** function to create a grid of histograms. You would set the facets to **carrier ~ month**, which would create a separate histogram for each combination of carrier and month.
6. **Customizing the Plot:** Finally, you might want to customize the plot to improve its appearance and readability. You could use the **labs()** function to add labels to the x-axis, y-axis, and the plot title. You could also use the **theme()** function to customize the plot theme. For example, you could change the text size, color, and position, and modify the legend and the plot background.

## Problem 10: On your own – Hollywood Age Gap

The website https://hollywoodagegap.com is a record of *THE AGE DIFFERENCE IN YEARS BETWEEN MOVIE LOVE INTERESTS*. This is an informational site showing the age gap between movie love interests and the data follows certain rules:

* The two (or more) actors play actual love interests (not just friends, coworkers, or some other non-romantic type of relationship)
* The youngest of the two actors is at least 17 years old
* No animated characters

The age gaps dataset includes “gender” columns, which always contain the values “man” or “woman”. These values appear to indicate how the characters in each film identify and some of these values do not match how the actor identifies. We apologize if any characters are misgendered in the data!

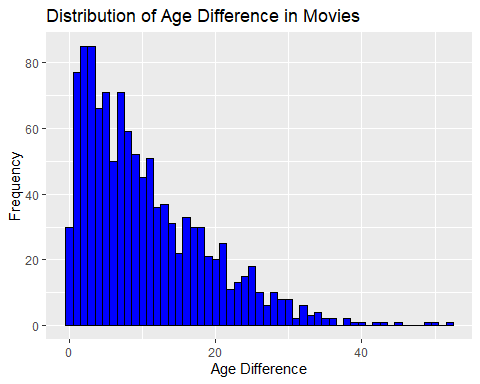
The following is a data dictionary of the variables used

| variable | class | description |
| --- | --- | --- |
| movie\_name | character | Name of the film |
| release\_year | integer | Release year |
| director | character | Director of the film |
| age\_difference | integer | Age difference between the characters in whole years |
| couple\_number | integer | An identifier for the couple in case multiple couples are listed for this film |
| actor\_1\_name | character | The name of the older actor in this couple |
| actor\_2\_name | character | The name of the younger actor in this couple |
| character\_1\_gender | character | The gender of the older character, as identified by the person who submitted the data for this couple |
| character\_2\_gender | character | The gender of the younger character, as identified by the person who submitted the data for this couple |
| actor\_1\_birthdate | date | The birthdate of the older member of the couple |
| actor\_2\_birthdate | date | The birthdate of the younger member of the couple |
| actor\_1\_age | integer | The age of the older actor when the film was released |
| actor\_2\_age | integer | The age of the younger actor when the film was released |

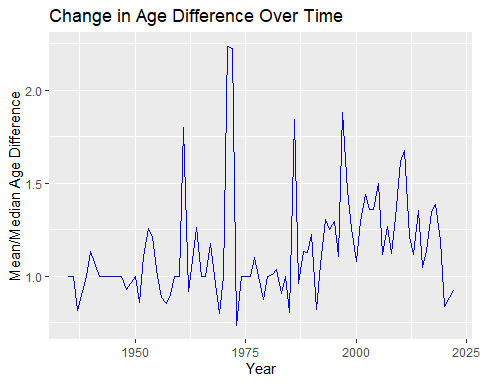
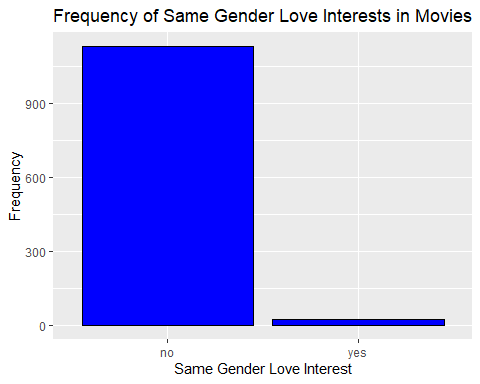
age\_gaps <- readr::read\_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2023/2023-02-14/age\_gaps.csv')

Rows: 1155 Columns: 13  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
chr (6): movie\_name, director, actor\_1\_name, actor\_2\_name, character\_1\_gend...  
dbl (5): release\_year, age\_difference, couple\_number, actor\_1\_age, actor\_2\_age  
date (2): actor\_1\_birthdate, actor\_2\_birthdate  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

How would you explore this data set? Here are some ideas of tables/ graphs to help you with your analysis

* How is age\_difference distributed? What’s the ‘typical’ age\_difference in movies?
* #Distribution of "age\_difference"  
  ggplot(age\_gaps, aes(x = age\_difference)) +  
   geom\_histogram(binwidth = 1, fill = "blue", color = "black") +  
   labs(x = "Age Difference", y = "Frequency",   
   title = "Distribution of Age Difference in Movies")
* 
* #-> left skewed with the mean of 10.4  
    
  #'typical' age\_difference   
  mean\_age\_diff <- mean(age\_gaps$age\_difference, na.rm = TRUE) #10.42  
  median\_age\_diff <- median(age\_gaps$age\_difference, na.rm = TRUE) # 8
* The half plus seven\ rule. Large age disparities in relationships carry certain stigmas. One popular rule of thumb is the [half-your-age-plus-seven](https://en.wikipedia.org/wiki/Age_disparity_in_sexual_relationships#The_.22half-your-age-plus-seven.22_rule) rule. This rule states you should never date anyone under half your age plus seven, establishing a minimum boundary on whom one can date. In order for a dating relationship to be acceptable under this rule, your partner’s age must be:

How frequently does this rule apply in this dataset?

* Which movie has the greatest number of love interests?
* library(dplyr)  
  movies\_love\_interests <- age\_gaps %>%  
   group\_by(movie\_name) %>%  
   summarise(num\_love\_interests = n\_distinct(couple\_number)) %>%  
   arrange(desc(num\_love\_interests))  
    
  top\_movie <- movies\_love\_interests[1, ]  
  print(top\_movie)
* # A tibble: 1 × 2  
   movie\_name num\_love\_interests  
   <chr> <int>  
  1 Love Actually 7
* #A. Love Actually
* Which actors/ actresses have the greatest number of love interests in this dataset?
* actor\_love\_interests <- age\_gaps %>%  
   mutate(actor = coalesce(actor\_1\_name, actor\_2\_name)) %>%  
   group\_by(actor) %>%  
   summarise(num\_love\_interests = n\_distinct(couple\_number)) %>%  
   arrange(desc(num\_love\_interests))  
    
  top\_actor <- actor\_love\_interests[1, ]  
  print(top\_actor)
* # A tibble: 1 × 2  
   actor num\_love\_interests  
   <chr> <int>  
  1 Pierce Brosnan 4
* #A. Pierce Brosnan
* Is the mean/median age difference staying constant over the years (1935 - 2022)?
* age\_diff\_over\_time <- age\_gaps %>%  
   group\_by(release\_year) %>%  
   summarise(mean\_age\_diff = mean(age\_difference, na.rm = TRUE),  
   median\_age\_diff = median(age\_difference, na.rm = TRUE))  
    
  ggplot(age\_diff\_over\_time, aes(x = release\_year)) +  
   geom\_line(aes(y = mean\_age\_diff/median\_age\_diff), color = "blue") +  
   labs(x = "Year", y = "Mean/Median Age Difference",   
   title = "Change in Age Difference Over Time")
* 
* #It has been changed over time, from around 7.5 to 2.25
* How frequently does Hollywood depict same-gender love interests?
* age\_gaps <- age\_gaps %>%  
   mutate(same\_gender = if\_else(character\_1\_gender == character\_2\_gender, "yes", "no"))  
    
  prop\_same\_gender <- mean(age\_gaps$same\_gender == "yes", na.rm = TRUE)  
    
  ggplot(age\_gaps, aes(x = same\_gender)) +  
   geom\_bar(fill = "blue", color = "black") +  
   labs(x = "Same Gender Love Interest", y = "Frequency",   
   title = "Frequency of Same Gender Love Interests in Movies")
* 
* same\_gender\_percentage <- prop\_same\_gender \* 100  
  print(same\_gender\_percentage)
* [1] 1.991342
* #A. 1.99%

# Deliverables

There is a lot of explanatory text, comments, etc. You do not need these, so delete them and produce a stand-alone document that you could share with someone. Render the edited and completed Quarto Markdown (qmd) file as a Word document (use the “Render” button at the top of the script editor window) and upload it to Canvas. You must be commiting and pushing tour changes to your own Github repo as you go along.

# Details

* Who did you collaborate with: Google
* Approximately how much time did you spend on this problem set: 15 hours
* What, if anything, gave you the most trouble: Understand how to merge datasets

**Please seek out help when you need it,** and remember the [15-minute rule](https://mam2022.netlify.app/syllabus/#the-15-minute-rule). You know enough R (and have enough examples of code from class and your readings) to be able to do this. If you get stuck, ask for help from others, post a question on Slack– and remember that I am here to help too!

As a true test to yourself, do you understand the code you submitted and are you able to explain it to someone else?

# Rubric

13/13: Problem set is 100% completed. Every question was attempted and answered, and most answers are correct. Code is well-documented (both self-documented and with additional comments as necessary). Used tidyverse, instead of base R. Graphs and tables are properly labelled. Analysis is clear and easy to follow, either because graphs are labeled clearly or you’ve written additional text to describe how you interpret the output. Multiple Github commits. Work is exceptional. I will not assign these often.

8/13: Problem set is 60–80% complete and most answers are correct. This is the expected level of performance. Solid effort. Hits all the elements. No clear mistakes. Easy to follow (both the code and the output). A few Github commits.

5/13: Problem set is less than 60% complete and/or most answers are incorrect. This indicates that you need to improve next time. I will hopefully not assign these often. Displays minimal effort. Doesn’t complete all components. Code is poorly written and not documented. Uses the same type of plot for each graph, or doesn’t use plots appropriate for the variables being analyzed. No Github commits.