Homework 4

Brady Lamson

2/26/2022

Problem 1

Use filter() with the flights data from the "nycflights13" package to find all the flights that:

a) Arrived more than 2 hours late but didn't leave late.

b) Were delayed by at least an hour, but made up over 30 minutes during flight.

```
flight_data <- nycflights13::flights
```

```
# A

flight_data %>%
    dplyr::filter(
        arr_delay > 120 & dep_delay <= 0
    )
```

```
## # A tibble: 29 x 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
## # ... with 19 more rows, and 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>
```

```r
# B

flight_data %>%
    dplyr::filter(
        dep_delay > 60 & (dep_delay - arr_delay) > 30
    )
```

```
## # A tibble: 1,819 x 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     2205           1720       285       46           2040
##  2  2013     1     1     2326           2130       116      131             18
##  3  2013     1     3     1503           1221       162     1803           1555
##  4  2013     1     3     1839           1700        99     2056           1950
##  5  2013     1     3     1850           1745        65     2148           2120
##  6  2013     1     3     1941           1759       102     2246           2139
##  7  2013     1     3     1950           1845        65     2228           2227
##  8  2013     1     3     2257           2000       177       45           2224
##  9  2013     1     4     1917           1700       137     2135           1950
## 10  2013     1     4     2010           1745       145     2257           2120
## # ... with 1,809 more rows, and 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

Use arrange() to sort the flights data to:

a) Find the fastest flights.

b) Find the longest flights.

```
# A

flight_data %>%
    dplyr::arrange(
        air_time
    ) %>%
    dplyr::select(
        flight, air_time, distance
    ) %>%
    head()
```

```
## # A tibble: 6 x 3
##   flight air_time distance
##    <int>    <dbl>    <dbl>
## 1   4368       20      116
## 2   4631       20      116
## 3   4276       21      116
## 4   4619       21       80
## 5   4368       21      116
## 6   4619       21       80
```

```
# B
# Longest in terms of distance
flight_data %>%
    dplyr::arrange(
        dplyr::desc(distance)
    ) %>%
    dplyr::select(
        flight, air_time, distance, origin, dest
    ) %>%
    head()
```

```
## # A tibble: 6 x 5
##   flight air_time distance origin dest
##    <int>    <dbl>    <dbl> <chr>  <chr>
## 1     51      659     4983 JFK    HNL
## 2     51      638     4983 JFK    HNL
## 3     51      616     4983 JFK    HNL
## 4     51      639     4983 JFK    HNL
## 5     51      635     4983 JFK    HNL
## 6     51      611     4983 JFK    HNL
```

```
# B
# Longest in terms of air_time

flight_data %>%
    dplyr::arrange(
        dplyr::desc(air_time)
    ) %>%
    dplyr::select(
        flight, air_time, distance, origin, dest
    ) %>%
    head()
```

```
## # A tibble: 6 x 5
##   flight air_time distance origin dest
##    <int>    <dbl>    <dbl> <chr>  <chr>
## 1     15      695     4963 EWR    HNL
## 2     51      691     4983 JFK    HNL
## 3     51      686     4983 JFK    HNL
## 4     51      686     4983 JFK    HNL
## 5     51      683     4983 JFK    HNL
## 6     51      679     4983 JFK    HNL
```

Problem 3

```
school <- read.table("nels88.txt", header = TRUE, stringsAsFactors = FALSE) %>%
    as_tibble()
```

a) Use filter() to extract a subset of the rows of the school data set.

```
# Filter out only the students with a bymath score above the mean and
# arrange them in descending order.
school %>%
    dplyr::filter(
        bymath > mean(school$bymath, na.rm = TRUE)
    ) %>%
    dplyr::arrange(
        dplyr::desc(bymath)
    ) %>%
    dplyr::select(
        bymath, f1math, f2math
    )
```

```
## # A tibble: 2,892 x 3
##    bymath f1math f2math
##     <dbl>  <dbl>  <dbl>
##  1   67.2   71.6   76.3
##  2   67.2   70.2   74.8
##  3   67.2   59.2   71.3
##  4   67.2   62.1   76.7
##  5   67.2   71.6   73.7
##  6   67.2   64.6   67.1
##  7   67.2   64.2   66.1
##  8   67.2   71.6   75.0
##  9   67.2   61.9   67.9
## 10   67.2   68.0   75.8
## # ... with 2,882 more rows
```

b) Use summarize to compute a summary statistic for each of at least three variables.

```
school %>%
    dplyr::summarise(
        bymath_mean = mean(bymath, na.rm = TRUE),
        f1math_max = max(f1math, na.rm = TRUE),
        f2math_min = min(f2math, na.rm = TRUE),
        proportion_hispanic = sum(hispanic, na.rm = TRUE) / nrow(school)
    )
```

```
## # A tibble: 1 x 4
##   bymath_mean f1math_max f2math_min proportion_hispanic
##         <dbl>      <dbl>      <dbl>               <dbl>
## 1        44.5       72.9       28.0               0.142
```

c) Use mutate() or transmute() to compute at least one new variable for the data set.

```
# Compute an average score for each student,
# then organize in descending order by that average.

school %>%
    dplyr::rowwise() %>%
    dplyr::mutate(
        avg_score = mean(c(bymath, f1math, f2math), na.rm = TRUE)
    ) %>%
    dplyr::select(
        bymath, f1math, f2math, avg_score
    ) %>%
    dplyr::arrange(
        dplyr::desc(avg_score)
    )
```

```
## # A tibble: 6,170 x 4
## # Rowwise:
##    bymath f1math f2math avg_score
##     <dbl>  <dbl>  <dbl>     <dbl>
##  1   NA     70.8   72.8      71.8
##  2   67.2   71.6   76.3      71.7
##  3   67.2   71.6   75.7      71.5
##  4   63.8   71.6   79.0      71.5
##  5   64.1   71.6   78.5      71.4
##  6   67.2   68.8   77.9      71.3
##  7   67.2   66.8   79.9      71.3
##  8   67.2   71.6   75.0      71.3
##  9   64.2   71.6   77.7      71.2
## 10   63.2   70.2   80.0      71.1
## # ... with 6,160 more rows
```

Problem 6 (Book)

Each of these tasks can be performed using a single data verb. Say what that verb is.

a) Find the average of one of the variables.

- summarize

b) Add a new column that is the ratio between two variables.

- mutate

c) Sort the cases in descending order of a variable.

- arrange

d) Create a new data table that includes only those cases that meet a criterion.

- filter

e) From a data table with three categorical variables A, B, and C, and a quantitative variable X, produce a data frame that has the same cases but only the variables A and X.

- select

Problem 9 (Book)

In the flights data set hat month had the highest proportion of cancelled flights? What month had the lowest?

```
# We can make the assumption that an NA value for arrival time
# represents a cancelled flight. Using this we can narrow things down.

cancel_percentage <- flight_data %>%
    dplyr::group_by(
        month
    ) %>%
    dplyr::summarise(
        cancel_percent = (sum(is.na(arr_time)) / n() * 100) %>% round(digits = 3)
    ) %>%
    dplyr::mutate(
        month = lubridate::month(month, label = TRUE)
    )

most_cancels <- cancel_percentage %>%
    dplyr::slice(which.max(cancel_percent))

least_cancels <- cancel_percentage %>%
    dplyr::slice(which.min(cancel_percent))

dplyr::full_join(
    most_cancels, least_cancels
)
```

```
## Joining, by = c("month", "cancel_percent")
```

```
## # A tibble: 2 x 2
##   month cancel_percent
##   <ord>          <dbl>
## 1 Feb            5.17
## 2 Oct            0.855
```

Problem 14 (Book)

Use the nycflights13 package and the flights data frame to answer the following question:

What plane (specified by the tailnum variable) traveled the most times from New York City airports in 2013? Plot the number of trips per week over the year.

```
flight_data %>%
    dplyr::group_by(tailnum) %>%
    dplyr::summarise(
        freq = n()
    ) %>%
    stats::na.omit() %>%
    dplyr::slice(
        which.max(freq)
    )
```

```
## # A tibble: 1 x 2
##   tailnum  freq
##   <chr>   <int>
## 1 N725MQ    575
```

From this we know that tailnum N725MQ is the highest frequency plane on this data set. Since this data set only accounts for flights with an origin of NYC airports (others are considered NA) and it already only accounts for flights in 2013, we can simply check the frequency of each tail number to get us where we need to go. From here we need to pull out relevant information related to this tail number so we can plot its flight patterns.

```
weekly_frequency <- flight_data %>%
    dplyr::filter(tailnum == "N725MQ") %>%
    dplyr::group_by(week = lubridate::week(time_hour)) %>%
    dplyr::summarise(
        freq = n()
    )

ggplot2::ggplot(weekly_frequency, ggplot2::aes(x = week, y = freq)) +
    ggplot2::geom_line()
```