EJERCICIOS

Julian Gonzalez

2023-09-02

```r
library("nycflights13")
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
allflights <- nycflights13::flights
```

```
 Exercise 5.2.4 - Item 1: Find all flights that:
1. Had an arrival delay of two or more hours:

```r
filter1 <- filter(flights, arr_delay >= 120)
```

2. Flew to Houston (IAH or HOU):

```r
filter2 <- filter(flights, dest %in% c("IAH", "HOU"))
```

3. Were operated by United, American, or Delta:

```r
airlines <- airlines
filter3 <- filter(flights, carrier %in% c("AA", "DL", "UA"))
```

4. Departed in summer (July, August, and September):

```r
filter4 <- filter(flights, month >= 7, month <= 9)
```

5. Arrived more than two hours late, but didn't leave late:
```

```
filter5 <- filter(flights, arr_delay > 120, dep_delay <= 0)
```

6. Were delayed by at least an hour, but made up over 30 minutes in flight:

```
filter6 <- filter(flights, dep_delay >= 60, dep_delay - arr_delay > 30)
```

7. Departed between midnight and 6am (inclusive):

```
filter7 <- filter(flights, dep_time <= 600 | dep_time == 2400)
```

Exercise 5.2.4 - Item 2: Another useful dplyr filtering helper is between(). What does it do? Can you use it to simplify the code needed to answer the previous challenges?:# Answer: Yes, because the expression between(x, left, right) is equivalent to x >= left & x <= right. So, of the answers in the previous exercise, we could simplify the statement of departed in summer (month >= 7 & month <= 9) using the between() function.

```
filter8 <- filter(flights, between(month, 7, 9))
```

Exercise 5.3.1 - Item 1: How could you use arrange() to sort all missing values to the start? (Hint: use is.na()):

```
arrange1 <- arrange(flights, desc(is.na(dep_time)), dep_time)
```

Exercise 5.3.1 - Item 2: Sort flights to find the most delayed flights. Find the flights that left earliest:

```
arrange2 <- arrange(flights, desc(dep_delay))
arrange3 <- arrange(flights, dep_delay)
```

Exercise 5.3.1 - Item 3: Sort flights to find the fastest (highest speed) flights: "Fastest flight" can be interpreted in two ways. The first as "the flight with the shortest flight time" (arrange4) and the second as "the flight with the highest average forward speed" (arrange5).

```
arrange4 <- head(arrange(flights, air_time))
arrange5 <- head(arrange(flights, desc(distance / air_time)))
```

Exercise 5.3.1 - Item 4: Which flights travelled the farthest? Which travelled the shortest?:

```
arrange6 <- arrange(flights, desc(air_time))
arrange7 <- arrange(flights, air_time)
```

Exercise 5.4.1 - Item 2: What happens if you include the name of a variable multiple times in a select() call?: Answer: The select() call ignores the duplication. So, any duplicated variables are only included once, in the first location they appear.

```
select1 <- select(flights, year, month, day, year, year)
```

Exercise 5.4.1 - Item 3: What does the any_of() function do? Why might it be helpful in conjunction with this vector? vars <- c("year", "month", "day", "dep_delay", "arr_delay"):

```
vars <- c("year", "month", "day", "dep_delay", "arr_delay")
```

Answer: The any_of() function selects variables with a character vector rather than unquoted variable name arguments. This function is useful because it is easier to programmatically generate character vectors with variable names than to generate unquoted variable names, which are easier to type.

```
select2 <- select(flights, any_of(vars))
```

Exercise 5.4.1 - Item 4: Does the result of running the following code surprise you? How do the select helpers deal with case by default? How can you change that default? select(flights, contains("TIME")):

```
item4 <- select(flights, contains("TIME"))
```

Answer: It's surprising since, the default behavior for contains() is to ignore case. This is important because users searching for variable names probably have a better sense of the letters in the variable than their capitalization. On the other hand, some of database engines have case insensitive column names, so making functions that match variable names case insensitive by default will make the behavior of select() consistent regardless of whether the table is stored as an R data frame or in a database. Finally, to change the behavior you need to add the argument ignore.case = FALSE.

```
select3 <- select(flights, contains("TIME", ignore.case = FALSE))
```

Exercise 5.5.2 - Item 1: Currently dep_time and sched_dep_time are convenient to look at, but hard to compute with because they're not really continuous numbers. Convert them to a more convenient representation of number of minutes since midnight:

```
time2mins <- function(x) {
  (x %/% 100 * 60 + x %% 100) %% 1440
}
mutate1 <- mutate(flights, dep_time_mins = time2mins(dep_time), sched_dep_time_mins = time2mins(sched_de
select4 <- select(mutate1, dep_time, dep_time_mins, sched_dep_time, sched_dep_time_mins)
```

Exercise 5.5.2 - Item 2: Compare air_time with arr_time - dep_time. What do you expect to see? What do you see? What do you need to do to fix it?: Answer: I expect that air_time is the difference between the arrival (arr_time) and departure times (dep_time). In other words, air_time = arr_time - dep_time. What I can see is that air_time is not equal to arr_time - dep_time, because: 1. The flight passes midnight, so arr_time < dep_time. In these cases, the difference in airtime should be by 24 hours (1,440 minutes) and 2. The flight crosses time zones, and the total air time will be off by hours (multiples of 60). All flights in flights departed from New York City and are domestic flights in the US. This means that flights will all be to the same or more westerly time zones. Given the time-zones in the US, the differences due to time-zone should be 60 minutes (Central) 120 minutes (Mountain), 180 minutes (Pacific), 240 minutes (Alaska), or 300 minutes (Hawaii). Finally, To fix these time-zone issues, I would want to convert all the times to a date-time to handle overnight flights, and from local time to a common time zone, most likely UTC, to handle flights crossing time-zones, leaning on that, the relationship between air_time, arr_time, and dep_time is air_time <= arr_time - dep_time, supposing that the time zones of arr_time and dep_time are in the same time zone.

Exercise 5.6.7 - Item 1: Brainstorm at least 5 different ways to assess the typical delay characteristics of a group of flights. Consider the following scenarios: - A flight is 15 minutes early 50% of the time, and 15 minutes late 50% of the time. - A flight is always 10 minutes late. - A flight is 30 minutes early 50% of the time, and 30 minutes late 50% of the time. - 99% of the time a flight is on time. 1% of the time it's 2 hours late. Which is more important: arrival delay or departure delay?: Answer: 1. What this question gets at is a fundamental question of data analysis: the cost function. As analysts, the reason we are interested in flight delay because it is costly to passengers. But it is worth thinking carefully about how it is costly and use that information in ranking and measuring these scenarios. 2. In many scenarios, arrival delay is more important. In most cases, being arriving late is more costly to the passenger since it could disrupt the next stages of their travel, such as connecting flights or scheduled meetings. 3. If a departure is delayed without affecting the arrival time, this delay will not have those affects plans nor does it affect the total time spent traveling.# 4. This delay could be beneficial, if less time is spent in the cramped confines of the airplane itself, or a negative, if that delayed time is still spent in the cramped confines of the airplane on the runway. 5. Variation in arrival time is worse than consistency. If a flight is always 30 minutes late and that delay is known, then it is as if the arrival time is that delayed time. The traveler could easily plan for this. But higher variation in flight times makes it harder to plan.