Homework 1

Dakota Thompson

2/12/2021

# Complete the following problems from Wickham and Grolemund:

## Section 5.2.4, Exercise 1

### Find all flights that

flights

## # A tibble: 336,776 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 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 336,766 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>

#### Had an arrival delay of two or more hours

Arr\_delay is in minutes, so 60 minutes per hour, then 2 hours would be 120 minutes. From there we have to have all of the delays that are greater than 2 hours and include the 2 hours.

delay = filter(flights, arr\_delay >= 120)  
  
delay

## # A tibble: 10,200 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 1 811 630 101 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  
## # ... with 10,190 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>

#### Flew to Houston (IAH or HOU)

We got the destination airport of IAH or HOU.

dest = filter(flights, dest == 'IAH' | dest == 'HOU')  
  
dest

## # A tibble: 9,313 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 1 517 515 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  
## # ... with 9,303 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>

#### Were operated by United, American, or Delta

We got the carrier airline of UA (United) or AA (American) or DL (Delta). I also googled nycflights13 to see where to find the airline names to codes.

airlines

## # A tibble: 16 x 2  
## carrier name   
## <chr> <chr>   
## 1 9E Endeavor Air Inc.   
## 2 AA American Airlines Inc.   
## 3 AS Alaska Airlines Inc.   
## 4 B6 JetBlue Airways   
## 5 DL Delta Air Lines Inc.   
## 6 EV ExpressJet Airlines Inc.   
## 7 F9 Frontier Airlines Inc.   
## 8 FL AirTran Airways Corporation  
## 9 HA Hawaiian Airlines Inc.   
## 10 MQ Envoy Air   
## 11 OO SkyWest Airlines Inc.   
## 12 UA United Air Lines Inc.   
## 13 US US Airways Inc.   
## 14 VX Virgin America   
## 15 WN Southwest Airlines Co.   
## 16 YV Mesa Airlines Inc.

carrier1 = filter(flights, carrier == 'UA' | carrier == 'AA' | carrier == 'DL')  
  
carrier1

## # A tibble: 139,504 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 1 517 515 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  
## # ... with 139,494 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>

#### Departed in summer (July, August, and September)

We got all the months of summer, which the question says is July-Sept. As such we just need the months between 7 and 9.

summer = filter(flights, between(month, 7, 9))  
  
summer

## # A tibble: 86,326 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 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  
## # ... with 86,316 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>

#### Arrived more than two hours late, but didn’t leave late

We needed all the arr\_delays that were more than 120 minutes late, however we also needed all the dep\_delays to be less than one because we wanted 0 (left on time) and negative numbers (left early) as they didn’t leave late.

deptimearrlate = filter(flights, arr\_delay > 120 & dep\_delay < 1)  
  
deptimearrlate

## # 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>

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

We need the flights to be delayed by more than an hour in leaving, but also the ones who were delayed 30 minutes less than how they were delayed in departure to show they made that time back up.

delaymadeback = filter(flights, arr\_delay >= 120 & arr\_delay <= dep\_delay - 30)  
  
delaymadeback

## # A tibble: 582 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 3 1503 1221 162 1803 1555  
## 3 2013 1 3 1821 1530 171 2131 1910  
## 4 2013 1 3 2257 2000 177 45 2224  
## 5 2013 1 4 2058 1730 208 2 2110  
## 6 2013 1 4 2221 1858 203 116 2240  
## 7 2013 1 6 2051 1820 151 2206 2005  
## 8 2013 1 7 1323 830 293 1604 1154  
## 9 2013 1 11 2136 1735 241 2245 1925  
## 10 2013 1 14 2301 1855 246 212 2245  
## # ... with 572 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>

#### Departed between midnight and 6am (inclusive)

We need the times between midnight and 1 am separate from 1 to 6 because 2400 > 100 and 600. So we have to put all the times from midnight to 1 or all the times from 1 to 6.

midto6 = filter(flights, between(dep\_time, 2400,2459) | between(dep\_time, 100,600))  
  
midto6

## # A tibble: 8,492 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 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 8,482 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>

## Section 5.3.1, Exercise 3

### Sort flights to find the fastest (highest speed) flights.

I added a speed variable of distance over time. Then I arranged the data frame by speed in descending order.

flightsspeed = mutate(flights, speed = distance/air\_time)  
highspeedtop = arrange(flightsspeed, desc(speed))  
  
highspeedtop

## # A tibble: 336,776 x 20  
## 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 5 25 1709 1700 9 1923 1937  
## 2 2013 7 2 1558 1513 45 1745 1719  
## 3 2013 5 13 2040 2025 15 2225 2226  
## 4 2013 3 23 1914 1910 4 2045 2043  
## 5 2013 1 12 1559 1600 -1 1849 1917  
## 6 2013 11 17 650 655 -5 1059 1150  
## 7 2013 2 21 2355 2358 -3 412 438  
## 8 2013 11 17 759 800 -1 1212 1255  
## 9 2013 11 16 2003 1925 38 17 36  
## 10 2013 11 16 2349 2359 -10 402 440  
## # ... with 336,766 more rows, and 12 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>,  
## # speed <dbl>

## Section 5.4.1, Exercise 3

### What does the any\_of() function do?

It searches a data frame for any of the characters put into the function.

### Why might it be helpful in conjunction with this vector?

#### vars <- c(“year”, “month”, “day”, “dep\_delay”, “arr\_delay”)

It could make it easier to traverse a huge data frame for the flights that aren’t performing as well (being on time) and when certain flights took place. It could also be used as a way to search flights within a period of time that were late, early, or on time.

## Section 5.5.2, Exercise 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?

flightsarrdep = mutate(flights, arrmindep = arr\_time-dep\_time)  
flightsarrdep = select(flightsarrdep, arrmindep, air\_time)  
  
flightsarrdep

## # A tibble: 336,776 x 2  
## arrmindep air\_time  
## <int> <dbl>  
## 1 313 227  
## 2 317 227  
## 3 381 160  
## 4 460 183  
## 5 258 116  
## 6 186 150  
## 7 358 158  
## 8 152 53  
## 9 281 140  
## 10 195 138  
## # ... with 336,766 more rows

I would expect to see them to be the same because the departure and arrival times would be the beginning and end of a flight. The air time and arrival - departure are different however. To fix this I would need to account for from where they left because of time zones.

## Section 5.6.7, Exercises 4 and 5

### Look at the number of cancelled flights per day. Is there a pattern?

# This takes all the flights that don't have arrivals delays or departure delays and assumes then that they were cancelled. It then groups by day to show how many were cancelled out of the total on that day.  
  
cancelledbyday = flights %>%  
 mutate(cancelled = (is.na(arr\_delay) | is.na(dep\_delay))) %>%  
 group\_by(year, month, day) %>%  
 summarise(numbercancelled = sum(cancelled),  
 totalflights = n())

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

cancelledbyday

## # A tibble: 365 x 5  
## # Groups: year, month [12]  
## year month day numbercancelled totalflights  
## <int> <int> <int> <int> <int>  
## 1 2013 1 1 11 842  
## 2 2013 1 2 15 943  
## 3 2013 1 3 14 914  
## 4 2013 1 4 7 915  
## 5 2013 1 5 3 720  
## 6 2013 1 6 3 832  
## 7 2013 1 7 3 933  
## 8 2013 1 8 7 899  
## 9 2013 1 9 9 902  
## 10 2013 1 10 3 932  
## # ... with 355 more rows

There seems to be more cancelled flights in the winter. There even seemed to be a major snow storm in early February.

### Is the proportion of cancelled flights related to the average delay?

# This takes the average cancelled per day (the same way we got the total above) to compare to the mean of the delayed flights.   
  
cancelledvdelay = flights %>%  
 mutate(cancelled = (is.na(arr\_delay) | is.na(dep\_delay))) %>%  
 group\_by(year, month, day) %>%  
 summarise(  
 cancelledmean = mean(cancelled),   
 departuremean = mean(dep\_delay, na.rm = TRUE),  
 arrivalmean = mean(arr\_delay, na.rm = TRUE)) %>%  
 ungroup()

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

cancelledvdelay

## # A tibble: 365 x 6  
## year month day cancelledmean departuremean arrivalmean  
## <int> <int> <int> <dbl> <dbl> <dbl>  
## 1 2013 1 1 0.0131 11.5 12.7   
## 2 2013 1 2 0.0159 13.9 12.7   
## 3 2013 1 3 0.0153 11.0 5.73   
## 4 2013 1 4 0.00765 8.95 -1.93   
## 5 2013 1 5 0.00417 5.73 -1.53   
## 6 2013 1 6 0.00361 7.15 4.24   
## 7 2013 1 7 0.00322 5.42 -4.95   
## 8 2013 1 8 0.00779 2.55 -3.23   
## 9 2013 1 9 0.00998 2.28 -0.264  
## 10 2013 1 10 0.00322 2.84 -5.90   
## # ... with 355 more rows

There seems to be higher average cancellations on days where there are higher average delays, for both departure and arrival.

### Which carrier has the worst delays?

# This takes all the flights by carrier to find the average arrival delay (as that matters more for next departure and could've made time back up over a flight) and then sort the average delay to have the worst on top.  
  
worstdelay = flights %>%  
 group\_by(carrier) %>%  
 summarise(arr\_delay = mean(arr\_delay, na.rm = TRUE)) %>%  
 arrange(desc(arr\_delay))  
  
worstdelay

## # A tibble: 16 x 2  
## carrier arr\_delay  
## <chr> <dbl>  
## 1 F9 21.9   
## 2 FL 20.1   
## 3 EV 15.8   
## 4 YV 15.6   
## 5 OO 11.9   
## 6 MQ 10.8   
## 7 WN 9.65   
## 8 B6 9.46   
## 9 9E 7.38   
## 10 UA 3.56   
## 11 US 2.13   
## 12 VX 1.76   
## 13 DL 1.64   
## 14 AA 0.364  
## 15 HA -6.92   
## 16 AS -9.93

It looks like F9, which is Frontier, has the worst delays.

### Challenge: can you disentangle the effects of bad airports vs. bad carriers? Why/why not? (Hint: think about flights %>% group\_by(carrier, dest) %>% summarise(n()))

You could compare all of the carrier delays to all the airport delays/routes.

# This shows how much arrival delay there is, by carrier, on each route (between airports). We took the arrival delay and the total flights for each carrier on each route, then the arrival delay just on those routes and the total amount of flights.  
  
challengeq = flights %>%  
 filter(!is.na(arr\_delay)) %>%  
 group\_by(origin, dest, carrier) %>%  
 summarise(arr\_delay = sum(arr\_delay),  
 flights = n()) %>%  
 group\_by(origin, dest) %>%  
 mutate(totalarrdelay = sum(arr\_delay),  
 totalflights = sum(flights)) %>%  
 ungroup()

## `summarise()` has grouped output by 'origin', 'dest'. You can override using the `.groups` argument.

challengeq

## # A tibble: 437 x 7  
## origin dest carrier arr\_delay flights totalarrdelay totalflights  
## <chr> <chr> <chr> <dbl> <int> <dbl> <int>  
## 1 EWR ALB EV 6018 418 6018 418  
## 2 EWR ANC UA -20 8 -20 8  
## 3 EWR ATL 9E -25 4 64525 4876  
## 4 EWR ATL DL 31149 3116 64525 4876  
## 5 EWR ATL EV 32330 1654 64525 4876  
## 6 EWR ATL UA 1071 102 64525 4876  
## 7 EWR AUS UA 2840 664 -454 957  
## 8 EWR AUS WN -3294 293 -454 957  
## 9 EWR AVL EV 2210 251 2210 251  
## 10 EWR BDL EV 2746 405 2904 412  
## # ... with 427 more rows

# This takes the total we made above to find the difference between the route arrival delay and the carrier arrival delay (on a route). It then takes the mean arrival delay of the carriers (arr\_delay over the amount of flights) and then gets the difference between that and the difference between the route arrival delay and the carrier arrival delay (on a route). I googled how to get rid of the NaNs. Then we can isolate the actual average of the arrival delays to compare.   
  
challengeq2 = challengeq %>%  
 mutate(otherarrdelay = (totalarrdelay - arr\_delay)/(totalflights - flights),  
 meanarrdelay = arr\_delay / flights,  
 diffarrdelay = meanarrdelay - otherarrdelay) %>%  
 filter(!is.nan(diffarrdelay)) %>%  
 group\_by(carrier) %>%  
 summarise(diffarrdelay = mean(diffarrdelay)) %>%  
 arrange(desc(diffarrdelay))  
  
challengeq2

## # A tibble: 15 x 2  
## carrier diffarrdelay  
## <chr> <dbl>  
## 1 OO 27.3   
## 2 F9 17.3   
## 3 EV 11.0   
## 4 B6 6.41   
## 5 FL 2.57   
## 6 VX -0.202  
## 7 AA -0.970  
## 8 WN -1.27   
## 9 UA -1.86   
## 10 MQ -2.48   
## 11 YV -2.81   
## 12 9E -3.54   
## 13 US -4.14   
## 14 DL -10.2   
## 15 AS -15.8

## Section 5.7.1, Exercise 3

### What time of day should you fly if you want to avoid delays as much as possible?

timeofday = flights %>%  
 group\_by(hour) %>%  
 summarise(arr\_delay = mean(arr\_delay, na.rm = TRUE)) %>%  
 arrange(arr\_delay)  
  
timeofday

## # A tibble: 20 x 2  
## hour arr\_delay  
## <dbl> <dbl>  
## 1 7 -5.30   
## 2 5 -4.80   
## 3 6 -3.38   
## 4 9 -1.45   
## 5 8 -1.11   
## 6 10 0.954  
## 7 11 1.48   
## 8 12 3.49   
## 9 13 6.54   
## 10 14 9.20   
## 11 23 11.8   
## 12 15 12.3   
## 13 16 12.6   
## 14 18 14.8   
## 15 22 16.0   
## 16 17 16.0   
## 17 19 16.7   
## 18 20 16.7   
## 19 21 18.4   
## 20 1 NaN

You should fly in the early morning!