R Bootcamp

Day 2 Exercises: dplyr

September 14, 2015

Questions in boldface are “bonus” and may require functions or techniques that we haven’t covered in class. If you don’t want to try them, that’s fine; if you do, I encourage you to practice finding the information you need via Google and using it on your problem. This is an invaluable skill.

Fire up a new R Markdown document. Silently load `dplyr`, `ggplot2` and any other packages you will use.

Download the flights.RDS file from the class smartsite. Read the file into R using readRDS. Get a sense for what the size of the data and what it represents using `str`, `summary`, and/or `head`: What spatial and temporal domain do they cover? There is additional information about the data at http://stat-computing.org/dataexpo/2009/the-data.html (http://stat-computing.org/dataexpo/2009/the-data.html). Because the dataset is quite larger, it may be advantagous to build your analyses on a small subset of the data, and once you have them all in or near final form, biuld your document using the full dataset.

```
setwd("~/Dropbox/Teaching/RBootcamp/Day2/")
d = readRDS("flights.RDS")
summary(d)
```

```
##       year          month         day_of_month    day_of_week
##  Min.   :2012   Min.   : 1.000   Min.   : 1.00   Min.   :1.000
##  1st Qu.:2012   1st Qu.: 1.000   1st Qu.: 8.00   1st Qu.:2.000
##  Median :2012   Median :11.000   Median :15.00   Median :4.000
##  Mean   :2012   Mean   : 6.499   Mean   :15.54   Mean   :3.942
##  3rd Qu.:2013   3rd Qu.:12.000   3rd Qu.:23.00   3rd Qu.:6.000
##  Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :7.000
##
##     carrier           fl_num        tail_num           origin
##  WN     :354963   Min.   :   1          :   5489   ATL    : 123915
##  EV     :232481   1st Qu.: 714   N480HA :   1405   ORD    :  94422
##  DL     :227539   Median :1720   N476HA :   1377   DFW    :  90184
##  OO     :201171   Mean   :2351   N478HA :   1375   DEN    :  70970
##  AA     :172342   3rd Qu.:3960   N477HA :   1323   LAX    :  69298
##  UA     :161857   Max.   :8990   N481HA :   1313   IAH    :  58850
##  (Other):611136                  (Other):1949207   (Other):1453850
##  origin_state_abr      dest          dest_state_abr      dep_time
##  CA     :238927   ATL    : 123882   CA     :238924   Min.   :   1
##  TX     :236012   ORD    :  94418   TX     :236006   1st Qu.: 933
##  FL     :149368   DFW    :  90171   FL     :149441   Median :1327
##  GA     :129173   DEN    :  70877   GA     :129141   Mean   :1330
##  IL     :125616   LAX    :  69300   IL     :125611   3rd Qu.:1723
##  NY     : 86916   IAH    :  58855   NY     : 86910   Max.   :2400
##  (Other):995477   (Other):1453986   (Other):995456   NA's   :30721
##    dep_delay          arr_time       arr_delay          air_time
##  Min.   :-111.000   Min.   :   1    Min.   : -92.00   Min.   :  8.0
##  1st Qu.:  -5.000   1st Qu.:1118    1st Qu.: -13.00   1st Qu.: 57.0
##  Median :  -2.000   Median :1517    Median :  -5.00   Median : 88.0
##  Mean   :   8.025   Mean   :1490    Mean   :   3.33   Mean   :107.7
##  3rd Qu.:   6.000   3rd Qu.:1906    3rd Qu.:   7.00   3rd Qu.:137.0
##  Max.   :1633.000   Max.   :2400    Max.   :1627.00   Max.   :691.0
##  NA's   :30721      NA's   :32950   NA's   :35780     NA's   :35780
##     distance        cancelled      cancellation_code carrier_delay
##  Min.   :  24.0   Min.   :0.0000    :1929524         Min.   :   0.0
##  1st Qu.: 337.0   1st Qu.:0.0000   A:   9107         1st Qu.:   0.0
##  Median : 599.0   Median :0.0000   B:  18460         Median :   1.0
##  Mean   : 761.7   Mean   :0.0163   C:   4395         Mean   :  17.5
##  3rd Qu.: 994.0   3rd Qu.:0.0000   D:      3         3rd Qu.:  18.0
##  Max.   :4983.0   Max.   :1.0000                     Max.   :1599.0
##                                                      NA's   :1619153
##  weather_delay       nas_delay       security_delay    late_aircraft_delay
##  Min.   :   0.0    Min.   :   0.0    Min.   :  0.0     Min.   :   0.0
##  1st Qu.:   0.0    1st Qu.:   0.0    1st Qu.:  0.0     1st Qu.:   0.0
##  Median :   0.0    Median :   3.0    Median :  0.0     Median :   4.0
##  Mean   :   2.3    Mean   :  12.6    Mean   :  0.1     Mean   :  22.4
##  3rd Qu.:   0.0    3rd Qu.:  17.0    3rd Qu.:  0.0     3rd Qu.:  28.0
##  Max.   :1615.0    Max.   :1207.0    Max.   :626.0     Max.   :1201.0
##  NA's   :1619153   NA's   :1619153   NA's   :1619153   NA's   :1619153
```

```
# These are all domestic US flights for Nov. 2012 - Feb. 2013.

# While building the analyses, use a small sample (1%) of the data to make calcula
tions faster.
# Once all analyses are done, comment this line to use all the data.
#d = d[sample(1:nrow(d), nrow(d) * .01, replace = FALSE), ]
```

1. How many flights are there in the dataset?

```
nrow(d)
```

```
## [1] 1961489
```

```
# or
summarise(d, n())
```

```
##       n()
## 1 1961489
```

2. Which airline has the most flights? The least?

```
sort(table(d$carrier))
```

```
##
##     VX     HA     F9     YV     9E     AS     FL     B6     US     MQ
##  17216  23468  23699  41305  44967  46737  61988  75816 130453 145487
##     UA     AA     OO     DL     EV     WN
## 161857 172342 201171 227539 232481 354963
```

```
# or
d %>%
    group_by(carrier) %>%
    summarise(nFlights = n()) %>%
    arrange(-nFlights)
```

```
## Source: local data frame [16 x 2]
##
##    carrier nFlights
## 1       WN   354963
## 2       EV   232481
## 3       DL   227539
## 4       OO   201171
## 5       AA   172342
## 6       UA   161857
## 7       MQ   145487
## 8       US   130453
## 9       B6    75816
## 10      FL    61988
## 11      AS    46737
## 12      9E    44967
## 13      YV    41305
## 14      F9    23699
## 15      HA    23468
## 16      VX    17216
```

```
# or (and better than above)
d %>%
    count(carrier) %>%
    arrange(-n)
```

```
## Source: local data frame [16 x 2]
##
##    carrier      n
## 1       WN 354963
## 2       EV 232481
## 3       DL 227539
## 4       OO 201171
## 5       AA 172342
## 6       UA 161857
## 7       MQ 145487
## 8       US 130453
## 9       B6  75816
## 10      FL  61988
## 11      AS  46737
## 12      9E  44967
## 13      YV  41305
## 14      F9  23699
## 15      HA  23468
## 16      VX  17216
```

3. How many airlines fly from LAX to SFO?

```
ssCarriers = d[d$origin == "LAX" & d$dest == "SFO", "carrier"]
length(unique(ssCarriers))
```

```
## [1] 6
```

```
# or
d %>%
    filter(origin == "LAX" & dest == "SFO") %>%
    summarise(n_distinct(carrier))
```

```
##   n_distinct(carrier)
## 1                   6
```

4. a. To what airports can you fly from Sacramento (SMF)?
   b. Which of these represents the longest distance flight?
   c. Plot the average time to each airport from SMF. Make sure the order of the destination airports makes the plot easy to read.

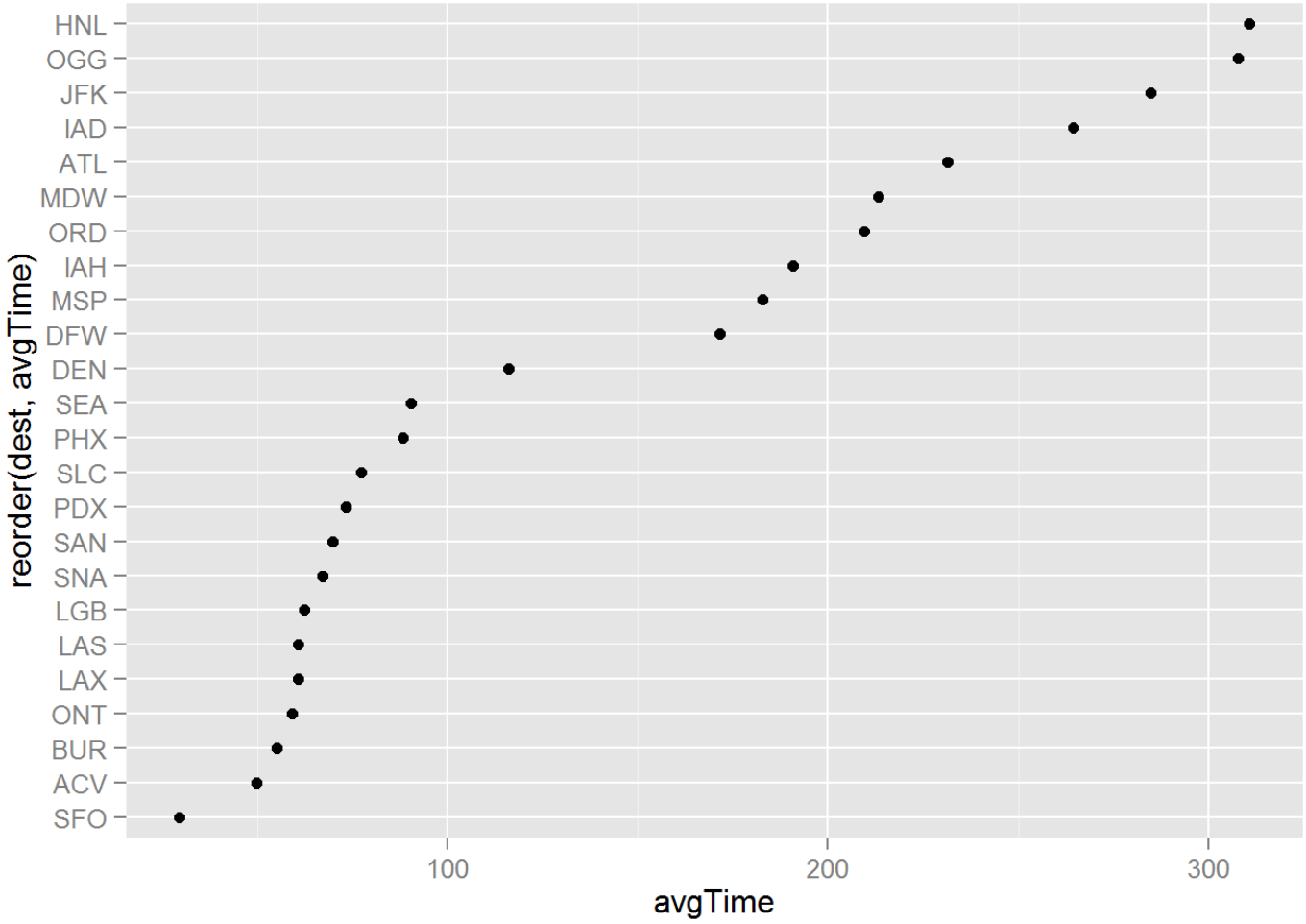
```
# a
d %>%
    filter(origin == "SMF") %>%
    select(dest) %>%
    distinct
```

```
##    dest
## 1   DFW
## 2   OGG
## 3   SEA
## 4   JFK
## 5   LGB
## 6   ATL
## 7   MSP
## 8   SLC
## 9   DEN
## 10  HNL
## 11  LAX
## 12  ACV
## 13  SFO
## 14  IAH
## 15  ORD
## 16  IAD
## 17  PHX
## 18  BUR
## 19  LAS
## 20  MDW
## 21  ONT
## 22  PDX
## 23  SAN
## 24  SNA
```

```
# b
d %>%
    filter(origin == "SMF") %>%
    group_by(dest) %>%
    summarise(dist = unique(distance)) %>%
    top_n(1, dist)
```

```
## Source: local data frame [1 x 2]
##
##   dest dist
## 1  JFK 2521
```

```
# c
d %>%
    filter(origin == "SMF") %>%
    group_by(dest) %>%
    summarise(avgTime = mean(air_time, na.rm = TRUE)) %>%
    ggplot(aes(x = reorder(dest, avgTime), y = avgTime)) +
    geom_point() +
    coord_flip()
```

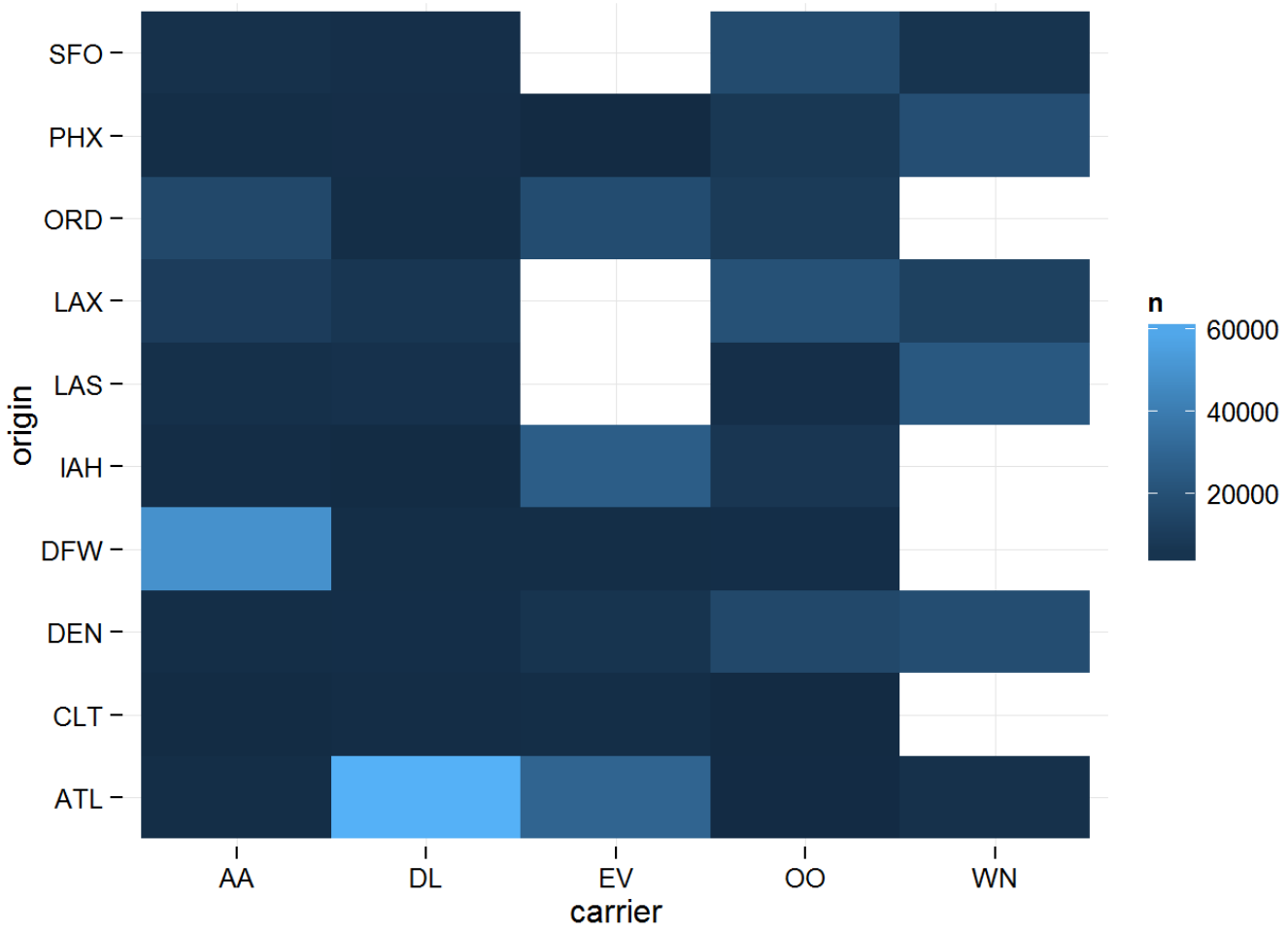


4. a. Identify the ten busiest airports and five highest-volume airlines.
   b. Among those, compute the number of flights each airline had at each origin airport.
   c. Display this information graphically. (**One possibility is a heatmap using `geom_tile` .**)

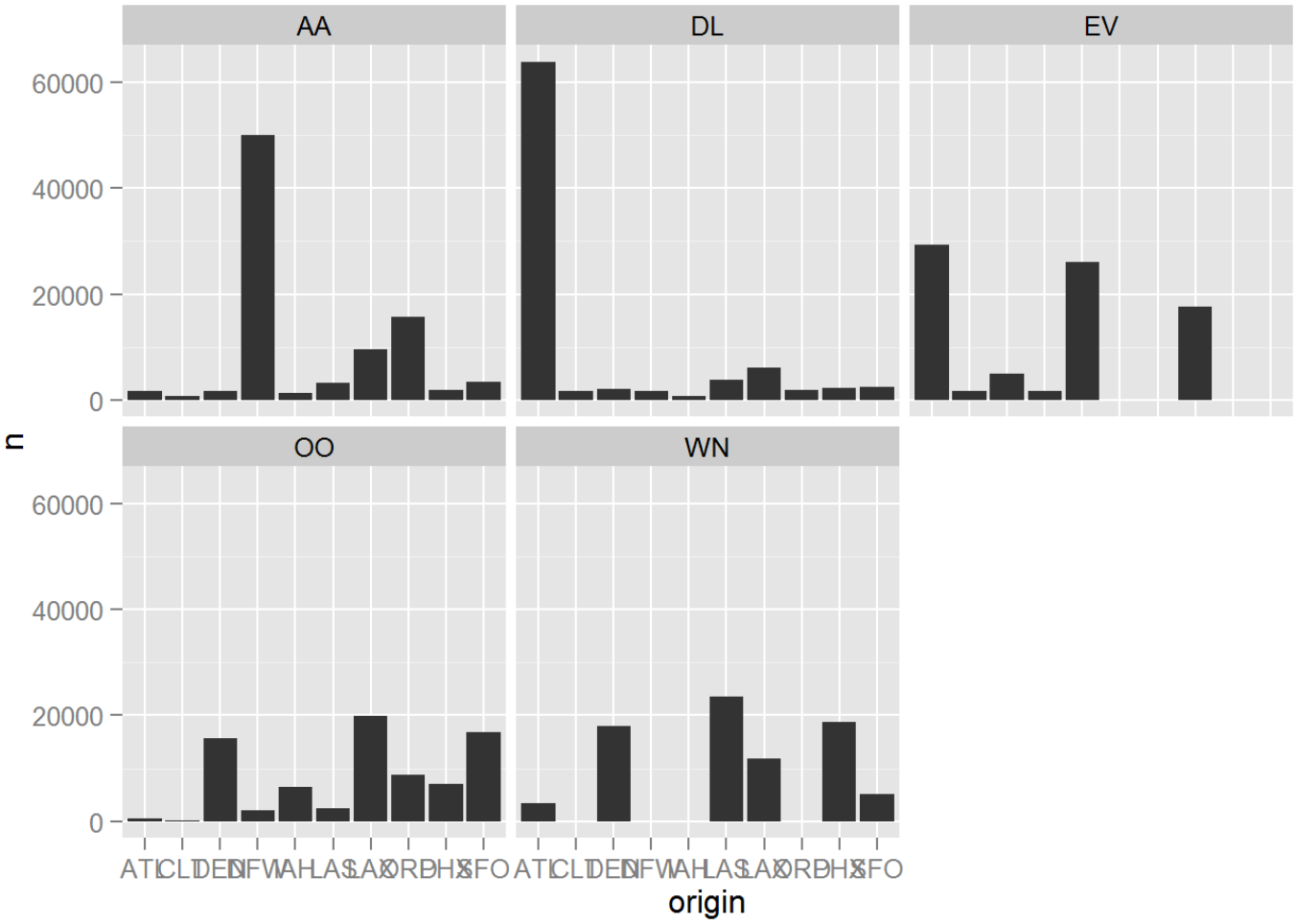
```
# a
o10 = sort(table(d$origin), decreasing = TRUE)[1:10]
c5 = sort(table(d$carrier), decreasing = TRUE)[1:5]
# or
o10 = d %>% count(origin) %>% top_n(10, n) %>% arrange(-n)
c5 = d %>% count(carrier) %>% top_n(5, n) %>% arrange(-n)

# b
cts =
    d %>%
        filter(origin %in% o10$origin & carrier %in% c5$carrier) %>%
        group_by(carrier, origin) %>%
        summarise(n = n())

# c
ggplot(cts, aes(x = carrier, y = origin, fill = n)) +
    geom_tile() +
    theme_minimal()
```

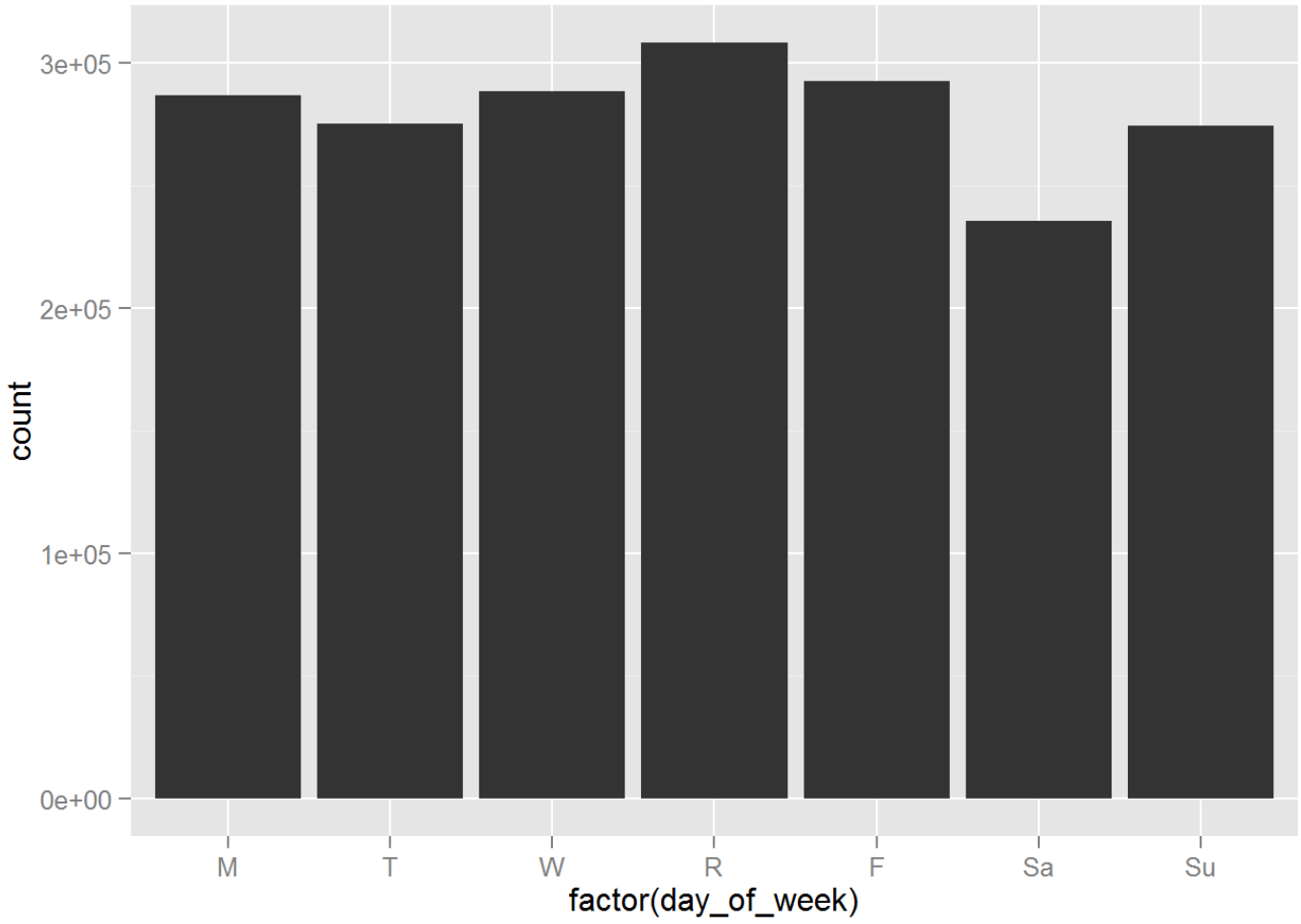



```
#or, could do a facetted-histogram:
ggplot(cts, aes(x = origin, y = n)) +
  geom_bar(stat = 'identity', position = 'dodge') +
  facet_wrap(~carrier)
```



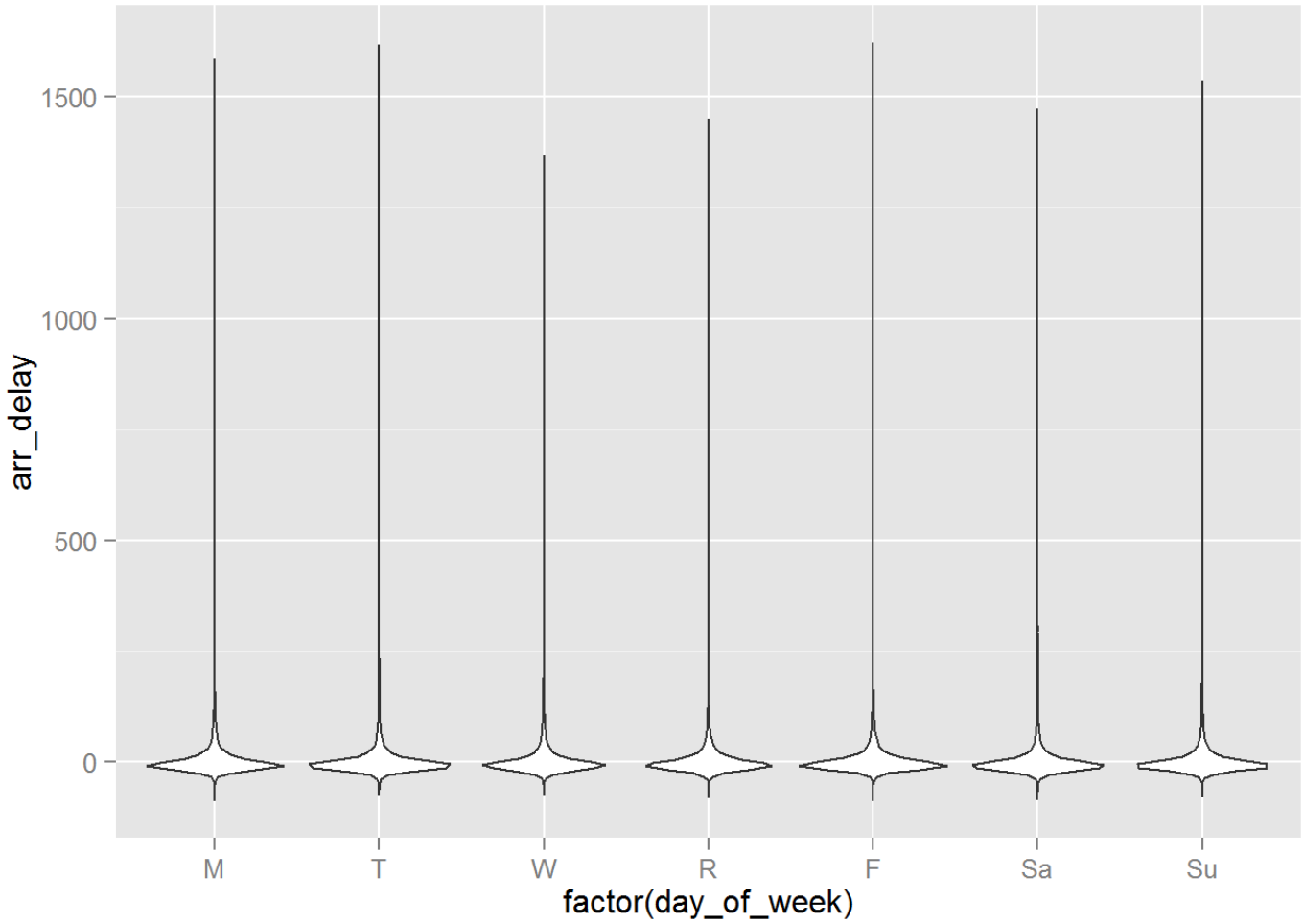
5. Answer all of the following questions with a plot.
    a. How does flight volume change over the days of the week?
    b. How do flight delays change over the days of the week?
    c. What is the relationship between departure and arrival delays?
    d. Calculate the difference between arrival delay and departure delay. What is the relationship of that value and flight distance? How do you explain that relationship?

```
# a
ggplot(d, aes(x = factor(day_of_week))) +
    geom_histogram() +
    scale_x_discrete(labels = c("M", "T", "W", "R", "F", "Sa", "Su"))
```



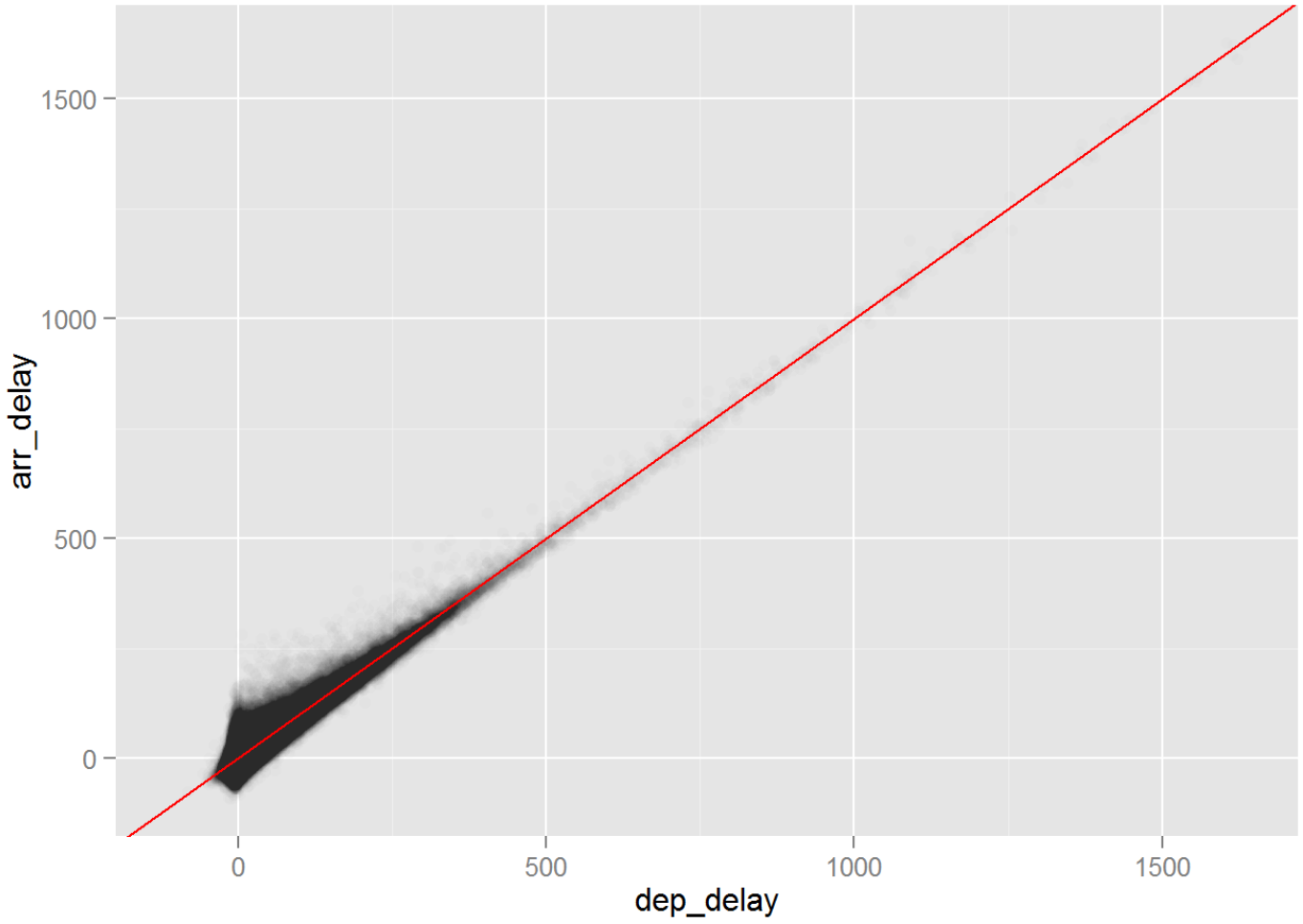

```
# b
d %>%
  ggplot(aes(x = factor(day_of_week), y = arr_delay)) +
    geom_violin() +
    scale_x_discrete(labels = c("M", "T", "W", "R", "F", "Sa", "Su"))
```

```
## Warning in loop_apply(n, do.ply): Removed 35780 rows containing non-finite
## values (stat_ydensity).
```




```
# c
ggplot(d, aes(x = dep_delay, y = arr_delay)) +
    geom_point(alpha = .01) +
    geom_abline(intercept = 0, slope = 1, color = 'red')
```

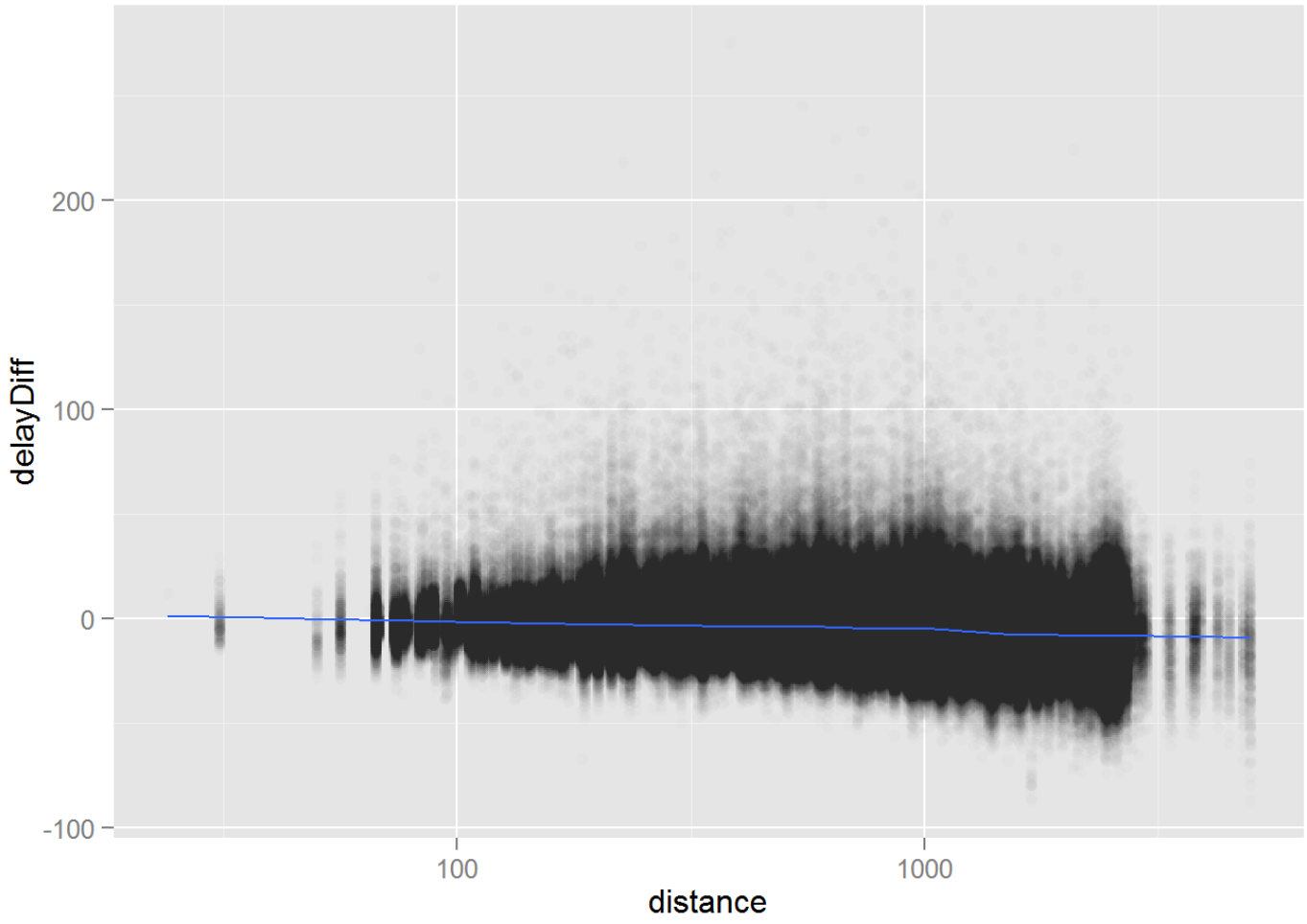
```
## Warning in loop_apply(n, do.ply): Removed 35780 rows containing missing
## values (geom_point).
```




```
# d
d %>%
    mutate(delayDiff = arr_delay - dep_delay) %>%
    ggplot(aes(x = distance, y = delayDiff)) +
    geom_point(alpha = .01) +
    scale_x_log10() +
    geom_smooth()
```

```
## Warning in loop_apply(n, do.ply): Removed 35780 rows containing missing
## values (stat_smooth).
```

```
## Warning in loop_apply(n, do.ply): Removed 35780 rows containing missing
## values (geom_point).
```




```
# Flights go faster when they depart later!
```

6. For this question, be sure you're handling missing values appropriately.
    a. What proportion of flights departed late?
    b. What proportion of flights arrived late?
    c. **What proportion of flights that departed late arrived late**?

```
# a
sum(d$dep_delay > 0, na.rm = TRUE) / sum(!is.na(d$dep_delay))
```

```
## [1] 0.3716506
```

```
# b
sum(d$arr_delay > 0, na.rm = TRUE) / sum(!is.na(d$arr_delay))
```

```
## [1] 0.365536
```

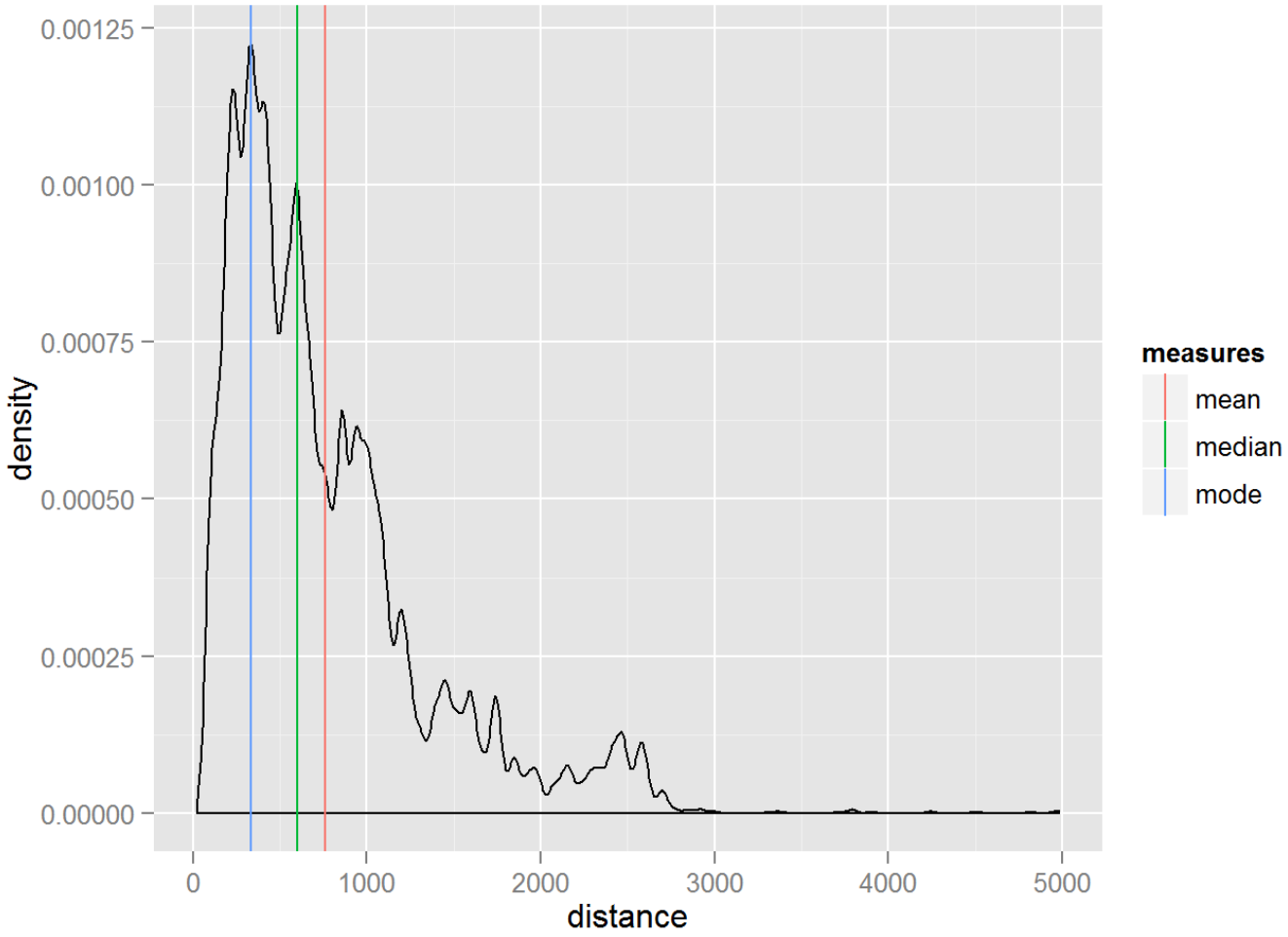
```
# c
arrDelayOfDelayedDepartures = complete.cases(d$arr_delay[d$dep_delay > 0])
sum(arrDelayOfDelayedDepartures > 0) / length(arrDelayOfDelayedDepartures)
```

```
## [1] 0.9554292
```

7. a. Calculate the mean, median, and modal flight distance.
   b. Plot the distribution of flight distances and add lines for each of the three summary statistics. **Do this programmatically and ensure that the statistic represeneted by each line is clear to the viewer.**
   c. How many pairs of airports share the modal flight distance (don't worry about origin-dest vs. dest-origin)? **Display them, with their number of count, in a nicely formatted table. The `xtable` package may be helpful here.**

```
### a
# Go ahead and put the summary statistics in a data.frame for use in b
summaryStats =
    data.frame(
        measures = c("mean", "median", "mode"),
        distance = c(mean(d$distance), median(d$distance),
                     as.numeric(names(sort(table(d$distance), decreasing = TRUE)
[1])))
    )
# Could also find modal distance using dplyr:
modalDist =
    d %>%
    count(distance) %>%
    top_n(1, n) %>%
    .$distance

### b
ggplot() +
    geom_density(data = d, aes(x = distance)) +
    geom_vline(data = summaryStats, aes(xintercept = distance, color = measures),
               show_guide = TRUE)
```




```
### c
pairs =
    d %>%
        filter(distance == summaryStats[3, 2]) %>%
        count(origin, dest)
nrow(pairs)
```

```
## [1] 8
```

```
htmlTable = xtable(pairs)
```

```
# This code chunk has the folowing option: "results = 'asis" to render the html re
turned by the function.
print(htmlTable, type = 'html')
```

	origin	dest	n
1	BWI	CMH	485
2	CMH	BWI	484
3	LAX	OAK	1834
4	LAX	SFO	5306

5	OAK	LAX	1836
6	PHL	RDU	800
7	RDU	PHL	798
8	SFO	LAX	5292

```
# You could of course make that prettier by adjusting the arguments to `xtable` in the previous code chunk.
```