STOR 320
Homework 2

Amanda Liu (PID:730042603)

1. Class Activity

1. Determine how many flights out of Newark had departure delays of more than 1 hour. (Use filter, please include either a snapshot, or copy-paste of the code you used here.)

10,940 flights out of Newark had departure delays of more than 1 hour.

> filter(flights, dep_delay>60, origin == "EWR")

```
> filter(flights, dep_delay>60, origin == "EWR")
# A tibble: 10,940 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     1      957            733       144     1056
 2  2013     1     1     1120            944        96     1331
 3  2013     1     1     1505           1310       115     1638
 4  2013     1     1     1525           1340       105     1831
 5  2013     1     1     1549           1445        64     1912
 6  2013     1     1     1607           1443        84     1711
 7  2013     1     1     1628           1524        64     1740
 8  2013     1     1     1639           1517        82     1815
 9  2013     1     1     1728           1600        88     2004
10  2013     1     1     1732           1630        62     2028
# ... with 10,930 more rows, and 12 more variables:
#   sched_arr_time <int>, 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>
```

2. Re-order (arrange) the rows first by departure time and then scheduled departure time. Save this new table as By_Arrival_Time.

>By_Arrival_Time <- arrange(flights, dep_time, sched_dep_time)

> By_Arrival_Time

```
> By_Arrival_Time <- arrange(flights, dep_time, sched_dep_time)
> By_Arrival_Time
# A tibble: 336,776 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013     4    10        1           1930       271      106           2101       245      UA
 2  2013     5    22        1           1935       266      154           2140       254      EV
 3  2013     6    24        1           1950       251      105           2130       215      AA
 4  2013     7     1        1           2029       212      236           2359       157      B6
 5  2013     1    31        1           2100       181      124           2225       179      WN
 6  2013     2    11        1           2100       181      111           2225       166      WN
 7  2013     3    18        1           2128       153      247           2355       172      B6
 8  2013     6    25        1           2130       151      249             14       155      B6
 9  2013     2    24        1           2245        76      121           2354        87      B6
10  2013     1    13        1           2249        72      108           2357        71      B6
# ... with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

a. For the new table, considering the first few rows, why are the scheduled departure times so much larger than the departure times?

Because the scheduled departure time is the day before.

b. Order the rows of By_Arrival_time by descending order of arrival delay and note the 10th largest arrival delay. How long was this delay in hours?

> By_Arrival_Time <- arrange(flights, desc(arr_delay))

> By_Arrival_Time

> 875/60.0

[1] 14.58333

This delay was 14.58 hours.

```
> By_Arrival_Time <- arrange(flights, desc(arr_delay))
> By_Arrival_Time
# A tibble: 336,776 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013     1     9      641            900      1301     1242           1530      1272      HA
 2  2013     6    15     1432           1935      1137     1607           2120      1127      MQ
 3  2013     1    10     1121           1635      1126     1239           1810      1109      MQ
 4  2013     9    20     1139           1845      1014     1457           2210      1007      AA
 5  2013     7    22      845           1600      1005     1044           1815       989      MQ
 6  2013     4    10     1100           1900       960     1342           2211       931      DL
 7  2013     3    17     2321            810       911      135           1020       915      DL
 8  2013     7    22     2257            759       898      121           1026       895      DL
 9  2013    12     5      756           1700       896     1058           2020       878      AA
10  2013     5     3     1133           2055       878     1250           2215       875      MQ
# ... with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
> 875/60.0
[1] 14.58333
```

3. Use ‘select’ to make a new table ‘Dep_Delay_Checks’ containing only the following variables: dep_time, sched_dep_time, dep_delay.
    - ~~**a.** In the new tibble, I notice that the order of variables is different than their order in ‘flights’. Why is this?~~
    - **b.** Use ‘mutate’ to make a new variable ‘calc_del’ to be calculated from dep_time and sched_dep_time. (s, you will need this column)
        - **i.** Does **mutate** change anything about the original table ‘Dep_Delay_Checks’? There are good reasons it does not. To preserve your new column, you can save the table under a new name, or more conveniently, save it as the same name. Do that now.

flights <- mutate(flights)

>Dep_Delay_Checks<-mutate(flights, calc_del = dep_time - sched_dep_time)

> Dep_Delay_Checks

```
> Dep_Delay_Checks
# A tibble: 336,776 x 20
   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin  dest air_time distance  hour minute         time_hour calc_del
  <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>  <int>   <chr>  <chr> <chr>    <dbl>    <dbl> <dbl>  <dbl>            <dttm>    <int>
1  2013     1     1      517            515         2      830            819        11      UA   1545  N14228    EWR   IAH      227     1400     5     15 2013-01-01 05:00:00        2
2  2013     1     1      533            529         4      850            830        20      UA   1714  N24211    LGA   IAH      227     1416     5     29 2013-01-01 05:00:00        4
3  2013     1     1      542            540         2      923            850        33      AA   1141  N619AA    JFK   MIA      160     1089     5     40 2013-01-01 05:00:00        2
4  2013     1     1      544            545        -1     1004           1022       -18      B6    725  N804JB    JFK   BQN      183     1576     5     45 2013-01-01 05:00:00       -1
5  2013     1     1      554            600        -6      812            837       -25      DL    461  N668DN    LGA   ATL      116      762     6      0 2013-01-01 06:00:00      -46
6  2013     1     1      554            558        -4      740            728        12      UA   1696  N39463    EWR   ORD      150      719     5     58 2013-01-01 05:00:00       -4
7  2013     1     1      555            600        -5      913            854        19      B6    507  N516JB    EWR   FLL      158     1065     6      0 2013-01-01 06:00:00      -45
8  2013     1     1      557            600        -3      709            723       -14      EV   5708  N829AS    LGA   IAD       53      229     6      0 2013-01-01 06:00:00      -43
9  2013     1     1      557            600        -3      838            846        -8      B6     79  N593JB    JFK   MCO      140      944     6      0 2013-01-01 06:00:00      -43
10 2013     1     1      558            600        -2      753            745         8      AA    301  N3ALAA    LGA   ORD      138      733     6      0 2013-01-01 06:00:00      -42
 2       533             529         4
 3       542             540         2
 4       544             545        -1
 5       554             600        -6
 6       554             558        -4
 7       555             600        -5
 8       557             600        -3
 9       557             600        -3
10       558             600        -2
# ... with 336,766 more rows
```

    - **c.** Calc_del should match another column. A quick glance at the mutated ‘Dep_Delay_Checks’ reveals that it does not.
        - **i.** What happened?

Calc_del does not match dep_delay. It just calculates the numerical different between dep_time and sched_dep_time, but the time is recorded in different units.

ii. We will find a way to deal with this later. For now, we will have to trust the data providers account of dep_delay and arr_delay.

d. Note that you could have made this table with a single command: Explain.

>Dep_Delay_Checks <- select(flights, contains("dep"))

>Dep_Delay_Checks

This command create a table only contains variables "dep", which are dep_time, sched_dep_time and dep_delay.

```
> Dep_Delay_Checks <- select(flights, contains("dep"))
>
> Dep_Delay_Checks
# A tibble: 336,776 x 3
   dep_time sched_dep_time dep_delay
      <int>          <int>     <dbl>
 1      517            515         2
 2      533            529         4
 3      542            540         2
 4      544            545        -1
 5      554            600        -6
 6      554            558        -4
 7      555            600        -5
```

4. We would like to associate some carriers with their departure delay times.

a. Using dplyr functions, determine the 3 airlines that had the 3 longest departure delays. (arrange, or perhaps select and arrange)

> arr_by_del = arrange(flights, desc(dep_delay))

> car_dep_del <- select(arr_by_del, carrier)

>head(car_dep_del)

HA,MQ,AA

```
> arr_by_del = arrange(flights, desc(dep_delay))
> car_dep_del <- select(arr_by_del, carrier)
> head(car_dep_del)
# A tibble: 6 x 1
  carrier
    <chr>
1      HA
2      MQ
3      MQ
4      AA
5      MQ
6      DL
```

b. Again using dplyr functions, determine the average departure delay and the average arrival delays by carrier (summarise, group_by … look at the book!)

>flights %>% group_by(carrier) %>% summarise(ave_dep_del = mean(dep_delay, na.rm = TRUE), ave_arr_del = mean(arr_delay, na.rm = TRUE))

```
> flights %>% group_by(carrier) %>% summarise(ave_dep_del = mean(dep_delay, na.rm = TRUE), ave_arr_del = mean(arr_delay, n
a.rm = TRUE))
# A tibble: 16 x 3
   carrier ave_dep_del ave_arr_del
     <chr>       <dbl>       <dbl>
 1     9E   16.725769   7.3796692
 2     AA    8.586016   0.3642909
 3     AS    5.804775  -9.9308886
 4     B6   13.022522   9.4579733
 5     DL    9.264505   1.6443409
 6     EV   19.955390  15.7964311
 7     F9   20.215543  21.9207048
 8     FL   18.726075  20.1159055
 9     HA    4.900585  -6.9152047
10     MQ   10.552041  10.7747334
11     OO   12.586207  11.9310345
12     UA   12.106073   3.5580111
13     US    3.782418   2.1295951
14     VX   12.869421   1.7644644
15     WN   17.711744   9.6491199
16     YV   18.996330  15.5569853
```

c. Flying out of Laguardia, which carriers should you avoid if you hate arrival delays? Explain.

>flights %>% filter(origin == "LGA") %>% group_by(carrier) %>% summarise(average_arr_del = mean(arr_delay, na.rm = TRUE)) %>% arrange(desc(average_arr_del))

We should avoid F9 because its arrival delay is longest.

```
> flights %>% filter(origin == "LGA") %>% group_by(carrier) %>% s
rise(average_arr_del = mean(arr_delay, na.rm = TRUE)) %>% arrang
c(average_arr_del))
# A tibble: 13 x 2
   carrier average_arr_del
     <chr>           <dbl>
 1      F9       21.920705
 2      FL       20.115906
 3      YV       15.556985
 4      B6       13.511419
 5      OO        9.434783
 6      MQ        9.334865
 7      EV        9.279878
 8      WN        8.218938
 9      UA        4.642189
10      DL        3.927776
11      US        2.530819
12      9E        1.768546
```

5. Who is speeding?

a. Mutate flights to include a column ‘air_gain’ that shows how much delay time is made up in the air by the airplane. Save this under the same name (flights)

>flights<-mutate(flights,air_gain=flights$dep_delay-flights$arr_delay)

> flights

```
> flights <- mutate(flights, air_gain = flights$dep_delay - flights$a
rr_delay)
> flights
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     1      517            515         2      830
 2  2013     1     1      533            529         4      850
 3  2013     1     1      542            540         2      923
 4  2013     1     1      544            545        -1     1004
 5  2013     1     1      554            600        -6      812
 6  2013     1     1      554            558        -4      740
 7  2013     1     1      555            600        -5      913
 8  2013     1     1      557            600        -3      709
 9  2013     1     1      557            600        -3      838
10  2013     1     1      558            600        -2      753
# ... with 336,766 more rows, and 13 more variables:
#   sched_arr_time <int>, 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>, air_gain <dbl>
```

b. Which two airlines have the highest average air_gain? (Use dplyr functions)

>average_air_gain<-

flights%>%group_by(carrier)%>%summarise(ave_air_gain=

mean(air_gain, na.rm=TRUE)) %>% arrange(desc(ave_air_gain))

> average_air_gain_carrier <- select(average_air_gain, carrier)

> average_air_gain_carrier

AS, HA

```
> average_air_gain <- flights %>% group_by(carrier) %>% summarise(ave
_air_gain = mean(air_gain, na.rm=TRUE)) %>% arrange(desc(ave_air_gain
))
> average_air_gain_carrier <- select(average_air_gain, carrier)
> average_air_gain_carrier
# A tibble: 16 x 1
   carrier
     <chr>
 1      AS
 2      HA
 3      VX
 4      9E
 5      UA
 6      AA
 7      WN
 8      DL
 9      EV
10      B6
11      YV
12      US
```

c. Air_gain is painful to read as minutes, make a new column air_gain_hours.

> flights <- mutate(flights, air_gain_hours = flights$air_gain/60)

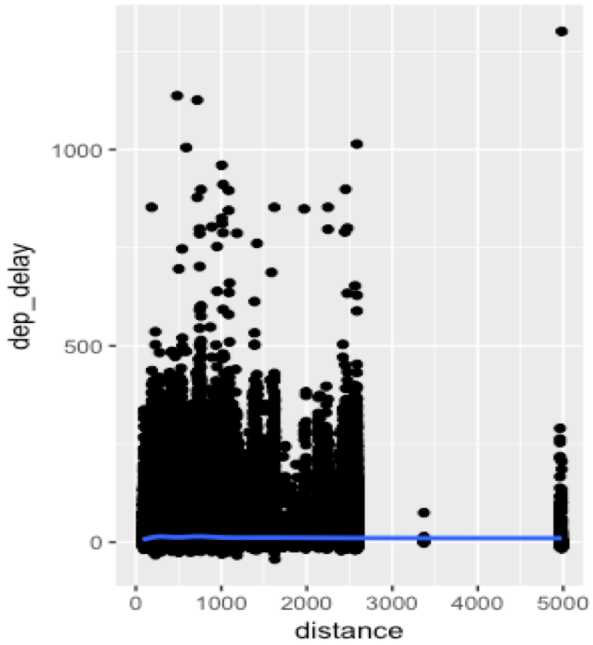
> select(flights, air_gain_hours)

```
> flights <- mutate(flights, air_gain_hours = flights$air_gain/60)
> select(flights, air_gain_hours)
# A tibble: 336,776 x 1
   air_gain_hours
            <dbl>
 1    -0.15000000
 2    -0.26666667
 3    -0.51666667
 4     0.28333333
 5     0.31666667
 6    -0.26666667
 7    -0.40000000
 8     0.18333333
 9     0.08333333
10    -0.16666667
# ... with 336,766 more rows
```

6. Make a plot to get a view of the relationship between

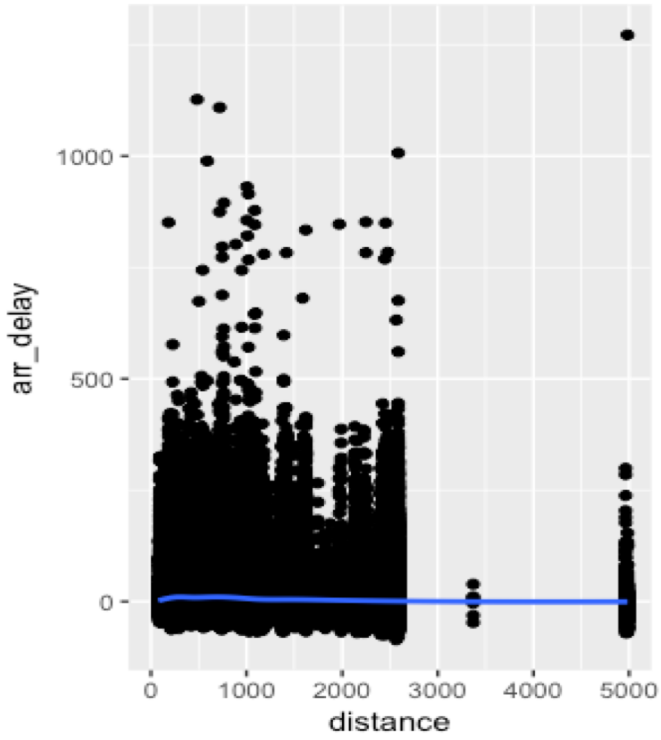
    a. Departure delays and distance

> ggplot(flights)+geom_point(mapping = aes(x = distance, y = dep_delay))+geom_smooth(mapping = aes(x = distance, y = dep_delay))



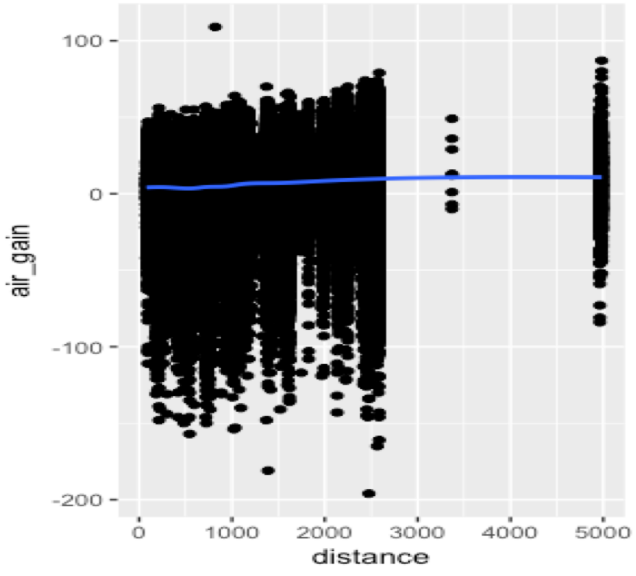
    b. Arrival delays and distance

```
> ggplot(flights)+geom_point(mapping = aes(x = distance, y = arr_delay))+geom_smooth(mapping = aes(x = distance, y = arr_delay))
```



c. Air_gains and distance

```
> ggplot(flights)+geom_point(mapping = aes(x = distance, y = air_gain))+geom_smooth(mapping = aes(x = distance, y = air_gain))
```



7. Arithmetic on vectors in R (new topic). Enter the following vectors into the

- X <- c(1,2,3)
- Y <- c(4, 5, 6)
- Z <- c(10,20)

console

a. Look at X/5, X+Y, X*Y, X/Y. Which of these do NOT reflect matrix arithmetic? Explain your answer.

X*Y and X/Y do not reflect matrix arithmetic. Because they are both 3*1 matrix They cannot be multiplied or divided.

```
> X<-c(1,2,3)
> Y<-c(4,5,6)
> Z<-c(10,20)
> X/5
[1] 0.2 0.4 0.6
> X+Y
[1] 5 7 9
> X*Y
[1]  4 10 18
> X/Y
[1] 0.25 0.40 0.50
```

b. Look at X*Z, X/Z, X+Z. Explain how R treats arithmetic operations on vectors of different lengths.

If the vectors have different lengths, R will automatically extend the shorter vector by repeating itself.

c.

Predict what X+5 should be and check it in the console.

Each of the component in X will be added in 5. Therefore, the results will be (6,7,8).

```
> X+5
[1] 6 7 8
```

2. Textbook Homework Problems

5.2.4 #1,2,3 (Part 6 of #1 could be a challenge)

1. Find all flights that
    - Had an arrival delay of two or more hours

> filter(flights, arr_delay>=120)

```
> filter(flights, arr_delay>=120)
# A tibble: 10,200 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>  <int>   <chr>
 1  2013     1     1      811            630       101     1047            830       137      MQ   4576  N531MQ
 2  2013     1     1      848           1835       853     1001           1950       851      MQ   3944  N942MQ
 3  2013     1     1      957            733       144     1056            853       123      UA    856  N534UA
 4  2013     1     1     1114            900       134     1447           1222       145      UA   1086  N76502
 5  2013     1     1     1505           1310       115     1638           1431       127      EV   4497  N17984
 6  2013     1     1     1525           1340       105     1831           1626       125      B6    525  N231JB
 7  2013     1     1     1549           1445        64     1912           1656       136      EV   4181  N21197
 8  2013     1     1     1558           1359       119     1718           1515       123      EV   5712  N826AS
 9  2013     1     1     1732           1630        62     2028           1825       123      EV   4092  N16911
10  2013     1     1     1803           1620       103     2008           1750       138      MQ   4622  N504MQ
# ... with 10,190 more rows, and 7 more variables: origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
```

    - Flew to Houston (IAH or HOU)

> filter(flights, dest %in% c('IAH', 'HOU'))

```
> filter(flights, dest %in% c('IAH', 'HOU'))
# A tibble: 9,313 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>  <int>   <chr>
 1  2013     1     1      517            515         2      830            819        11      UA   1545  N14228
 2  2013     1     1      533            529         4      850            830        20      UA   1714  N24211
 3  2013     1     1      623            627        -4      933            932         1      UA    496  N459UA
 4  2013     1     1      728            732        -4     1041           1038         3      UA    473  N488UA
 5  2013     1     1      739            739         0     1104           1038        26      UA   1479  N37408
 6  2013     1     1      908            908         0     1228           1219         9      UA   1220  N12216
 7  2013     1     1     1028           1026         2     1350           1339        11      UA   1004  N76508
 8  2013     1     1     1044           1045        -1     1352           1351         1      UA    455  N667UA
 9  2013     1     1     1114            900       134     1447           1222       145      UA   1086  N76502
10  2013     1     1     1205           1200         5     1503           1505        -2      UA   1461  N39418
# ... with 9,303 more rows, and 7 more variables: origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
```

- Were operated by United, American, or Delta

> filter(flights, carrier %in% c('UA', 'AA', 'DL'))

```
> filter(flights, carrier %in% c('UA', 'AA', 'DL'))
# A tibble: 139,504 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     1      517            515         2      830
 2  2013     1     1      533            529         4      850
 3  2013     1     1      542            540         2      923
 4  2013     1     1      554            600        -6      812
 5  2013     1     1      554            558        -4      740
 6  2013     1     1      558            600        -2      753
 7  2013     1     1      558            600        -2      924
 8  2013     1     1      558            600        -2      923
 9  2013     1     1      559            600        -1      941
10  2013     1     1      559            600        -1      854
# ... with 139,494 more rows, and 12 more variables: sched_arr_time <int>,
#   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)

> filter(flights, month %in% c(7, 8, 9))

```
> filter(flights, month %in% c(7, 8, 9))
# A tibble: 86,326 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     7     1        1           2029       212      236
 2  2013     7     1        2           2359         3      344
 3  2013     7     1       29           2245       104      151
 4  2013     7     1       43           2130       193      322
 5  2013     7     1       44           2150       174      300
 6  2013     7     1       46           2051       235      304
 7  2013     7     1       48           2001       287      308
 8  2013     7     1       58           2155       183      335
 9  2013     7     1      100           2146       194      327
10  2013     7     1      100           2245       135      337
# ... with 86,316 more rows, and 12 more variables: sched_arr_time <int>,
#   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

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

```
> filter(flights, arr_delay > 120, dep_delay <= 0)
# A tibble: 29 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1    27     1419           1420        -1     1754
 2  2013    10     7     1350           1350         0     1736
 3  2013    10     7     1357           1359        -2     1858
 4  2013    10    16      657            700        -3     1258
 5  2013    11     1      658            700        -2     1329
 6  2013     3    18     1844           1847        -3       39
 7  2013     4    17     1635           1640        -5     2049
 8  2013     4    18      558            600        -2     1149
 9  2013     4    18      655            700        -5     1213
10  2013     5    22     1827           1830        -3     2217
# ... with 19 more rows, and 12 more variables: sched_arr_time <int>,
#   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

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

```
> filter(flights, dep_delay >= 60, dep_delay-arr_delay > 30)
# A tibble: 1,844 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     1     2205           1720       285       46
 2  2013     1     1     2326           2130       116      131
 3  2013     1     3     1503           1221       162     1803
 4  2013     1     3     1839           1700        99     2056
 5  2013     1     3     1850           1745        65     2148
 6  2013     1     3     1941           1759       102     2246
 7  2013     1     3     1950           1845        65     2228
 8  2013     1     3     2015           1915        60     2135
 9  2013     1     3     2257           2000       177       45
10  2013     1     4     1917           1700       137     2135
# ... with 1,834 more rows, and 12 more variables: sched_arr_time <int>,
#   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)

> filter(flights, dep_time <=600 | dep_time == 2400)

```
> filter(flights, dep_time <=600 | dep_time == 2400)
# A tibble: 9,373 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     1      517            515         2      830
 2  2013     1     1      533            529         4      850
 3  2013     1     1      542            540         2      923
 4  2013     1     1      544            545        -1     1004
 5  2013     1     1      554            600        -6      812
 6  2013     1     1      554            558        -4      740
 7  2013     1     1      555            600        -5      913
 8  2013     1     1      557            600        -3      709
 9  2013     1     1      557            600        -3      838
10  2013     1     1      558            600        -2      753
# ... with 9,363 more rows, and 12 more variables: sched_arr_time <int>,
#   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>
```

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?

Between is a more convenient way of testing two inequalities at once. It is an easier way to find the results between two values. It tests whether its first is greater or equal to its second, and less or equal to its third. For example, we can simplify the flights in summer by:

> filter(flights, between(month, 7, 9))

```
> filter(flights, between(month, 7, 9))
# A tibble: 86,326 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     7     1        1           2029       212      236
 2  2013     7     1        2           2359         3      344
 3  2013     7     1       29           2245       104      151
 4  2013     7     1       43           2130       193      322
 5  2013     7     1       44           2150       174      300
 6  2013     7     1       46           2051       235      304
 7  2013     7     1       48           2001       287      308
 8  2013     7     1       58           2155       183      335
 9  2013     7     1      100           2146       194      327
10  2013     7     1      100           2245       135      337
# ... with 86,316 more rows, and 12 more variables:
#   sched_arr_time <int>, 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>
```

3. How many flights have a missing dep_time? What other variables are missing? What might these rows represent?

> sum(is.na(flights$dep_time))

>summary(flights)

8255 flights have a missing dep_time.

arr_time(8713), dep_delay(8255), arr_delay(9430) and air_time(9430)

It represents that scheduled flights were cancelled or didn't arrive.

```
> sum(is.na(flights$dep_time))
[1] 8255
```

```
> summary(flights)
      year          month             day           dep_time    sched_dep_time   dep_delay
 Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106   Min.   : -43.00
 1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906   1st Qu.:  -5.00
 Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359   Median :  -2.00
 Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344   Mean   :  12.64
 3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729   3rd Qu.:  11.00
 Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359   Max.   :1301.00
                                                 NA's   :8255                  NA's   :8255
    arr_time    sched_arr_time   arr_delay          carrier              flight
 Min.   :   1   Min.   :   1   Min.   : -86.000   Length:336776      Min.   :   1
 1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000   Class :character   1st Qu.: 553
 Median :1535   Median :1556   Median :  -5.000   Mode  :character   Median :1496
 Mean   :1502   Mean   :1536   Mean   :   6.895                      Mean   :1972
 3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000                      3rd Qu.:3465
 Max.   :2400   Max.   :2359   Max.   :1272.000                      Max.   :8500
 NA's   :8713                  NA's   :9430
   tailnum             origin              dest              air_time        distance
 Length:336776      Length:336776      Length:336776      Min.   : 20.0   Min.   :  17
 Class :character   Class :character   Class :character   1st Qu.: 82.0   1st Qu.: 502
 Mode  :character   Mode  :character   Mode  :character   Median :129.0   Median : 872
                                                          Mean   :150.7   Mean   :1040
                                                          3rd Qu.:192.0   3rd Qu.:1389
                                                          Max.   :695.0   Max.   :4983
                                                          NA's   :9430
      hour           minute        time_hour                      air_gain       air_gain_hours
 Min.   : 1.00   Min.   : 0.00   Min.   :2013-01-01 05:00:00   Min.   :-196.00   Min.   :-3.267
 1st Qu.: 9.00   1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00   1st Qu.:  -3.00   1st Qu.:-0.050
 Median :13.00   Median :29.00   Median :2013-07-03 10:00:00   Median :   7.00   Median : 0.117
 Mean   :13.18   Mean   :26.23   Mean   :2013-07-03 05:02:36   Mean   :   5.66   Mean   : 0.094
 3rd Qu.:17.00   3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00   3rd Qu.:  17.00   3rd Qu.: 0.283
 Max.   :23.00   Max.   :59.00   Max.   :2013-12-31 23:00:00   Max.   : 109.00   Max.   : 1.817
                                                               NA's   :9430      NA's   :9430
```

5.3.1 #2,3,4

2. Sort flights to find the most delayed flights. Find the flights that left earliest.

> arrange(flights, desc(dep_delay))

```
> arrange(flights, desc(dep_delay))
# A tibble: 336,776 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time
   <int> <int> <int>    <int>          <int>     <dbl>    <int>
 1  2013     1     9      641            900      1301     1242
 2  2013     6    15     1432           1935      1137     1607
 3  2013     1    10     1121           1635      1126     1239
 4  2013     9    20     1139           1845      1014     1457
 5  2013     7    22      845           1600      1005     1044
 6  2013     4    10     1100           1900       960     1342
 7  2013     3    17     2321            810       911      135
 8  2013     6    27      959           1900       899     1236
 9  2013     7    22     2257            759       898      121
10  2013    12     5      756           1700       896     1058
# ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
#   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>
```

>arrange(flights, dep_delay)

```
> arrange(flights, dep_delay)
# A tibble: 336,776 x 21
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013    12     7     2040           2123       -43       40           2352        48      B6
 2  2013     2     3     2022           2055       -33     2240           2338       -58      DL
 3  2013    11    10     1408           1440       -32     1549           1559       -10      EV
 4  2013     1    11     1900           1930       -30     2233           2243       -10      DL
 5  2013     1    29     1703           1730       -27     1947           1957       -10      F9
 6  2013     8     9      729            755       -26     1002            955         7      MQ
 7  2013    10    23     1907           1932       -25     2143           2143         0      EV
 8  2013     3    30     2030           2055       -25     2213           2250       -37      MQ
 9  2013     3     2     1431           1455       -24     1601           1631       -30      9E
10  2013     5     5      934            958       -24     1225           1309       -44      B6
# ... with 336,766 more rows, and 11 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
#   air_gain <dbl>, air_gain_hours <dbl>
```

3. Sort flights to find the fastest flights.

>arrange(flights,desc(distance/air_time))

```
> arrange(flights,desc(distance/air_time))
# A tibble: 336,776 x 21
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013     5    25     1709           1700         9     1923           1937       -14      DL
 2  2013     7     2     1558           1513        45     1745           1719        26      EV
 3  2013     5    13     2040           2025        15     2225           2226        -1      EV
 4  2013     3    23     1914           1910         4     2045           2043         2      EV
 5  2013     1    12     1559           1600        -1     1849           1917       -28      DL
 6  2013    11    17      650            655        -5     1059           1150       -51      DL
 7  2013     2    21     2355           2358        -3      412            438       -26      B6
 8  2013    11    17      759            800        -1     1212           1255       -43      AA
 9  2013    11    16     2003           1925        38       17             36       -19      DL
10  2013    11    16     2349           2359       -10      402            440       -38      B6
# ... with 336,766 more rows, and 11 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
#   air_gain <dbl>, air_gain_hours <dbl>
```

4. Which flights travelled the longest? Which travelled the shortest?

Longest: > **arrange(flights,desc(distance))**

Shortest:>**arrange(flights,distance)**

```
> arrange(flights,desc(distance))
# A tibble: 336,776 x 21
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013     1     1      857            900        -3     1516           1530       -14      HA
 2  2013     1     2      909            900         9     1525           1530        -5      HA
 3  2013     1     3      914            900        14     1504           1530       -26      HA
 4  2013     1     4      900            900         0     1516           1530       -14      HA
 5  2013     1     5      858            900        -2     1519           1530       -11      HA
 6  2013     1     6     1019            900        79     1558           1530        28      HA
 7  2013     1     7     1042            900       102     1620           1530        50      HA
 8  2013     1     8      901            900         1     1504           1530       -26      HA
 9  2013     1     9      641            900      1301     1242           1530      1272      HA
10  2013     1    10      859            900        -1     1449           1530       -41      HA
# ... with 336,766 more rows, and 11 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
#   air_gain <dbl>, air_gain_hours <dbl>
> arrange(flights,distance)
# A tibble: 336,776 x 21
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>   <chr>
 1  2013     7    27       NA            106        NA       NA            245        NA      US
 2  2013     1     3     2127           2129        -2     2222           2224        -2      EV
 3  2013     1     4     1240           1200        40     1333           1306        27      EV
 4  2013     1     4     1829           1615       134     1937           1721       136      EV
 5  2013     1     4     2128           2129        -1     2218           2224        -6      EV
 6  2013     1     5     1155           1200        -5     1241           1306       -25      EV
 7  2013     1     6     2125           2129        -4     2224           2224         0      EV
 8  2013     1     7     2124           2129        -5     2212           2224       -12      EV
 9  2013     1     8     2127           2130        -3     2304           2225        39      EV
10  2013     1     9     2126           2129        -3     2217           2224        -7      EV
# ... with 336,766 more rows, and 11 more variables: flight <int>, tailnum <chr>, origin <chr>,
#   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
#   air_gain <dbl>, air_gain_hours <dbl>
```

5.4.1 #2

2. What happens if you include the name of a variable multiple times in a select() call?

> select(flights, sched_dep_time,sched_dep_time,sched_dep_time)

Nothing. It still turns out once.

```
> select(flights, sched_dep_time,sched_dep_time,sched_dep_time)
# A tibble: 336,776 x 1
   sched_dep_time
            <int>
 1            515
 2            529
 3            540
 4            545
 5            600
 6            558
 7            600
 8            600
 9            600
10            600
# ... with 336,766 more rows
```

5.5.2 #1,2,3,5

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.

> transmute(flights, sched_dep_time = (sched_dep_time%%100)*60+sched_dep_time%%100,dep_time=(dep_time %%100)*60+dep_time%%100)

```
> transmute(flights, sched_dep_time = (sched_dep_time%%100)*60+sched_dep_time%%100,dep_time
60+dep_time%%100)
# A tibble: 336,776 x 2
   sched_dep_time dep_time
            <dbl>    <dbl>
 1            915     1037
 2           1769     2013
 3           2440     2562
 4           2745     2684
 5              0     3294
 6           3538     3294
```

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?

arr_time is in clock format, but dep_time is calculated by minutes-after-midnight. So we need to convert arr_time to minutes-after-midnight.

arr_time - dep_time vary significantly from air_time.

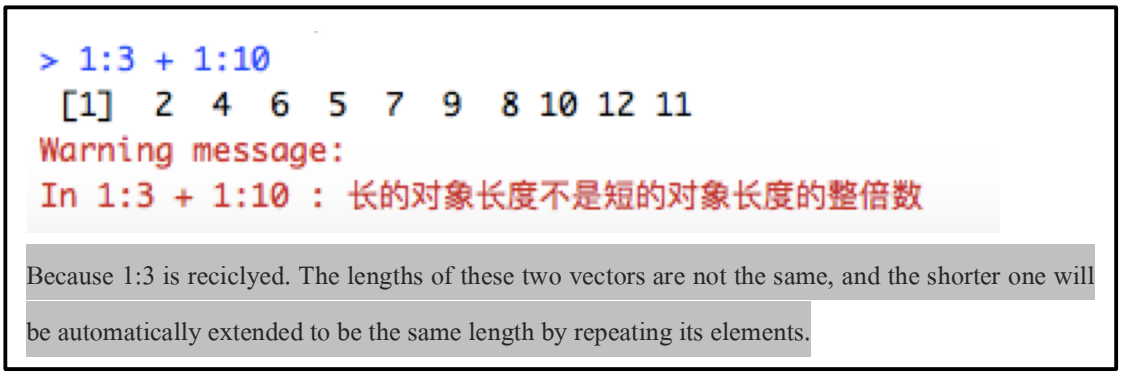
```
> flights_new <- select(flights, air_time, arr_time, dep_time)
> mutate(flights_new, air_time_new = arr_time - dep_time)
# A tibble: 336,776 x 4
   air_time arr_time dep_time air_time_new
      <dbl>    <int>    <int>        <int>
 1      227      830      517          313
 2      227      850      533          317
 3      160      923      542          381
 4      183     1004      544          460
 5      116      812      554          258
 6      150      740      554          186
 7      158      913      555          358
 8       53      709      557          152
 9      140      838      557          281
10      138      753      558          195
# ... with 336,766 more rows
```

3. Compare dep_time, sched_dep_time, and dep_delay. How would you expect those three numbers to be related?

We would expect that sched_dep_time + dep_delay == dep_time.

```
> select(flights,dep_time,sched_dep_time,dep_delay)
# A tibble: 336,776 x 3
   dep_time sched_dep_time dep_delay
      <int>          <int>     <dbl>
 1      517            515         2
 2      533            529         4
 3      542            540         2
 4      544            545        -1
 5      554            600        -6
 6      554            558        -4
 7      555            600        -5
 8      557            600        -3
 9      557            600        -3
10      558            600        -2
# ... with 336,766 more rows
```

5. What does 1:3 + 1:10 return? Why?



Because 1:3 is reciclyed. The lengths of these two vectors are not the same, and the shorter one will be automatically extended to be the same length by repeating its elements.