Homework3_1

201982188 통계학과 박현주

2020/9/23

Chap 4. Data transformation

```
library(nycflights13)
library(tidyverse)

#데이터 확인
flights %>% head(5)
```

```
# A tibble: 5 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
# ... with 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>
```

```
flights %>% glimpse
```

```
Observations: 336,776
Variables: 19
$ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013...
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 55...
$ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 60...
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2,...
$ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 8...
$ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 8...
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7,...
$ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6"...
$ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301...
$ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N...
$ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LG...
$ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IA...
$ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149...
$ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 73...
$ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6...
$ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59...
$ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-0...
```

```
flights %>% summary()
```

```
     year          month             day           dep_time    sched_dep_time
Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106
1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906
Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359
Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344
3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729
Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359
                                                NA's   :8255
  dep_delay          arr_time    sched_arr_time   arr_delay
Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000
1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000
Median :  -2.00   Median :1535   Median :1556   Median :  -5.000
Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
NA's   :8255      NA's   :8713                  NA's   :9430
  carrier              flight       tailnum             origin
Length:336776      Min.   :   1   Length:336776      Length:336776
Class :character   1st Qu.: 553   Class :character   Class :character
Mode  :character   Median :1496   Mode  :character   Mode  :character
                   Mean   :1972
                   3rd Qu.:3465
                   Max.   :8500

    dest              air_time        distance         hour
Length:336776      Min.   : 20.0   Min.   :  17   Min.   : 1.00
Class :character   1st Qu.: 82.0   1st Qu.: 502   1st Qu.: 9.00
Mode  :character   Median :129.0   Median : 872   Median :13.00
                   Mean   :150.7   Mean   :1040   Mean   :13.18
                   3rd Qu.:192.0   3rd Qu.:1389   3rd Qu.:17.00
                   Max.   :695.0   Max.   :4983   Max.   :23.00
                   NA's   :9430
    minute        time_hour
Min.   : 0.00   Min.   :2013-01-01 05:00:00
1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00
Median :29.00   Median :2013-07-03 10:00:00
Mean   :26.23   Mean   :2013-07-03 05:22:54
3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00
Max.   :59.00   Max.   :2013-12-31 23:00:00
```

1. filter function

[example]

- month가 1이고 day가 1인 경우

```
flights %>% filter(month == 1, day == 1) %>% head(10)
```

```
# A tibble: 10 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 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>
```

- month가 11과 12인 경우

```
flights %>% filter(month %in% c(11,12)) %>% head(10)
```

```
# A tibble: 10 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    11     1        5           2359         6      352            345
 2  2013    11     1       35           2250       105      123           2356
 3  2013    11     1      455            500        -5      641            651
 4  2013    11     1      539            545        -6      856            827
 5  2013    11     1      542            545        -3      831            855
 6  2013    11     1      549            600       -11      912            923
 7  2013    11     1      550            600       -10      705            659
 8  2013    11     1      554            600        -6      659            701
 9  2013    11     1      554            600        -6      826            827
10  2013    11     1      554            600        -6      749            751
# ... with 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>
```

- [참고]

```
  sqrt(2) ^ 2 == 2
```

```
[1] FALSE
```

```
  near(sqrt(2) ^ 2, 2)
```

```
[1] TRUE
```

여기에 대해 False라고 한다. 왜냐하면 왼쪽은 2에 가까워지는 수이지 2가 아니고 R에서도 무한대 형태의 저장을 못하기 때문이다. 하지만 아래의 결과처럼 비슷하냐고 물으면 그렇다고 출력한다.

[Your turn 01]

1-1. Had an arrival delay of two or more hours

```
flights %>% filter(arr_delay >= 120 ) #arr_delay는 분으로 표시됐으니 주의하자.
```

```
# 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>
```

1-2. Flew to Houston (IAH or HOU)

```
flights %>% filter(dest == 'IAH'| dest =='HOU')
```

```
# 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>
```

1-3. Were operated by United (UA), American (AA), or Delta (DL). Find the data airline to get details.

```
airlines #airlines를 통해 축약어의 풀네임을 볼 수 있다.
```

```
# 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.
```

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

```
# 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>
```

1-4. Departed in summer (July, August, and September)

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

```
# 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>
```

```
flights %>% filter(month %in% 7:9)
```

```
# 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>
```

위와 결과 같다. between을 사용해도 같은 결과를 볼 수 있다.

1-5. Arrived more than two hours late, but didn't leave late

```
flights %>% 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>
```

1-6. Were delayed by at least an hour, but made up over 30 minutes in flight.

```
flights %>% filter(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 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     2015           1915        60     2135           2111
 9  2013     1     3     2257           2000       177       45           2224
10  2013     1     4     1917           1700       137     2135           1950
# ... with 1,834 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>
```

dep_delay가 1시간 이상이지만 비행기에서 30분을 절약했다면 출발 지연에서 도착 지연의 값을 뺐을 때 30분이 단축돼야한다.

1-7.Departed between midnight and 6am (inclusive)

```
flights2 <- flights %>% mutate(new_dep_time = sprintf('%04d',flights$dep_time),
                  dep_hour = as.integer(substr(new_dep_time,1,2)),
                  dep_min = as.integer(substr(new_dep_time,3,4)))

flights2 %>% filter(dep_hour == 24  | dep_time %in% c(1:600))
```

```
# A tibble: 9,373 x 22
    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 9,363 more rows, and 14 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>,
#   new_dep_time <chr>, dep_hour <int>, dep_min <int>
```

```
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 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 9,363 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>
```

첫 번째 방법으로 풀었으나 두번 째 방법이 훨씬 간단하다.

- [참고] modulo operator : %% => returns the remainder of division.

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?

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

```
# 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>
```

위 1-4.번 문제를 between을 이용해 풀면 위와 같다.

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

```
flights$dep_time %>% is.na %>% sum #취소된 비행기의 수
```

```
[1] 8255
```

```
flights %>% filter(is.na(dep_time))
```

```
# A tibble: 8,255 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       NA           1630        NA       NA           1815
 2  2013     1     1       NA           1935        NA       NA           2240
 3  2013     1     1       NA           1500        NA       NA           1825
 4  2013     1     1       NA            600        NA       NA            901
 5  2013     1     2       NA           1540        NA       NA           1747
 6  2013     1     2       NA           1620        NA       NA           1746
 7  2013     1     2       NA           1355        NA       NA           1459
 8  2013     1     2       NA           1420        NA       NA           1644
 9  2013     1     2       NA           1321        NA       NA           1536
10  2013     1     2       NA           1545        NA       NA           1910
# ... with 8,245 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>
```

dep_time이 NA라는 것은 취소된 항공이라는 뜻이다. 취소된 8255개의 항공은 위와 같다.

2. arrange function

[example]

```
flights %>% arrange(dep_time) #dep_time을 오름차순으로 보여준다.
```

```
# 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    13        1           2249        72      108           2357
 2  2013     1    31        1           2100       181      124           2225
 3  2013    11    13        1           2359         2      442            440
 4  2013    12    16        1           2359         2      447            437
 5  2013    12    20        1           2359         2      430            440
 6  2013    12    26        1           2359         2      437            440
 7  2013    12    30        1           2359         2      441            437
 8  2013     2    11        1           2100       181      111           2225
 9  2013     2    24        1           2245        76      121           2354
10  2013     3     8        1           2355         6      431            440
# ... 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>
```

```
flights %>% arrange(desc(dep_time)) #내림차순
```

```
# 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    10    30     2400           2359         1      327            337
 2  2013    11    27     2400           2359         1      515            445
 3  2013    12     5     2400           2359         1      427            440
 4  2013    12     9     2400           2359         1      432            440
 5  2013    12     9     2400           2250        70       59           2356
 6  2013    12    13     2400           2359         1      432            440
 7  2013    12    19     2400           2359         1      434            440
 8  2013    12    29     2400           1700       420      302           2025
 9  2013     2     7     2400           2359         1      432            436
10  2013     2     7     2400           2359         1      443            444
# ... 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>
```

[Your turn 02]

1. How could you use arrange() to sort all missing values (in dep_time) to the start?

```
flights %>% mutate(dep_time_na = is.na(flights$dep_time)) %>%
  arrange(desc(dep_time_na),dep_time)
```

```
# 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     1     1       NA           1630        NA       NA           1815
 2  2013     1     1       NA           1935        NA       NA           2240
 3  2013     1     1       NA           1500        NA       NA           1825
 4  2013     1     1       NA            600        NA       NA            901
 5  2013     1     2       NA           1540        NA       NA           1747
 6  2013     1     2       NA           1620        NA       NA           1746
 7  2013     1     2       NA           1355        NA       NA           1459
 8  2013     1     2       NA           1420        NA       NA           1644
 9  2013     1     2       NA           1321        NA       NA           1536
10  2013     1     2       NA           1545        NA       NA           1910
# ... 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>,
#   dep_time_na <lgl>
```

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

```
flights %>% arrange(desc(dep_delay)) %>% select(carrier,origin,dest,dep_delay)
```

```
# A tibble: 336,776 x 4
   carrier origin dest  dep_delay
   <chr>   <chr>  <chr>     <dbl>
 1 HA      JFK    HNL        1301
 2 MQ      JFK    CMH        1137
 3 MQ      EWR    ORD        1126
 4 AA      JFK    SFO        1014
 5 MQ      JFK    CVG        1005
 6 DL      JFK    TPA         960
 7 DL      LGA    MSP         911
 8 DL      JFK    PDX         899
 9 DL      LGA    ATL         898
10 AA      EWR    MIA         896
# ... with 336,766 more rows
```

가장 지연된 비행기는 JFK에서 HNL로 가는 HA이다.

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

```
flights %>% arrange(air_time) %>% select(carrier,origin,dest,air_time)
```

```
# A tibble: 336,776 x 4
   carrier origin dest  air_time
   <chr>   <chr>  <chr>    <dbl>
 1 EV      EWR    BDL         20
 2 EV      EWR    BDL         20
 3 EV      EWR    BDL         21
 4 EV      EWR    PHL         21
 5 EV      EWR    BDL         21
 6 EV      EWR    PHL         21
 7 US      LGA    BOS         21
 8 9E      JFK    PHL         21
 9 EV      EWR    BDL         21
10 EV      EWR    BDL         21
# ... with 336,766 more rows
```

가장 빠른 비행기는 EWR에서 BDL로 가는 EV이다.

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

```
flights %>% arrange(desc(distance))  %>% select(carrier,origin,dest,
                                          distance)
```

```
# A tibble: 336,776 x 4
   carrier origin dest  distance
   <chr>   <chr>  <chr>    <dbl>
 1 HA      JFK    HNL       4983
 2 HA      JFK    HNL       4983
 3 HA      JFK    HNL       4983
 4 HA      JFK    HNL       4983
 5 HA      JFK    HNL       4983
 6 HA      JFK    HNL       4983
 7 HA      JFK    HNL       4983
 8 HA      JFK    HNL       4983
 9 HA      JFK    HNL       4983
10 HA      JFK    HNL       4983
# ... with 336,766 more rows
```

```
flights %>% arrange(distance)  %>% select(carrier,origin,dest,
                                          distance)
```

```
# A tibble: 336,776 x 4
   carrier origin dest  distance
   <chr>   <chr>  <chr>    <dbl>
 1 US      EWR    LGA         17
 2 EV      EWR    PHL         80
 3 EV      EWR    PHL         80
 4 EV      EWR    PHL         80
 5 EV      EWR    PHL         80
 6 EV      EWR    PHL         80
 7 EV      EWR    PHL         80
 8 EV      EWR    PHL         80
 9 EV      EWR    PHL         80
10 EV      EWR    PHL         80
# ... with 336,766 more rows
```

가장 거리가 먼 비행은 JRK에서 HNL로 가는 HA이다. 그 거리는 위와 같다. 가장 짧은 비행은 EWR에서 LGA로 가는 US이다. 그 거리는 위와 같다.

3. select function

[example]

- 해당 변수만 포함

```
select(flights, year, month, day)
```

```
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
 8  2013     1     1
 9  2013     1     1
10  2013     1     1
# ... with 336,766 more rows
```

```
select(flights, year:day)
```

```
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
 8  2013     1     1
 9  2013     1     1
10  2013     1     1
# ... with 336,766 more rows
```

- 해당 변수 제외

```
select(flights, -(year:day)) #제외하고 보겠다.
```

```
# A tibble: 336,776 x 16
   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
      <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
 1      517            515         2      830            819        11 UA
 2      533            529         4      850            830        20 UA
 3      542            540         2      923            850        33 AA
 4      544            545        -1     1004           1022       -18 B6
 5      554            600        -6      812            837       -25 DL
 6      554            558        -4      740            728        12 UA
 7      555            600        -5      913            854        19 B6
 8      557            600        -3      709            723       -14 EV
 9      557            600        -3      838            846        -8 B6
10      558            600        -2      753            745         8 AA
# ... 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>
```

- [참고 1]

1. statr_with : 인자로 시작하는 변수 포함
2. ends_with : 인자로 끝나는 변수 포함

3. contaions : 인자를 포함하는 모든 변수

```
select(flights, starts_with("arr"))
```

```
# A tibble: 336,776 x 2
   arr_time arr_delay
      <int>     <dbl>
 1      830        11
 2      850        20
 3      923        33
 4     1004       -18
 5      812       -25
 6      740        12
 7      913        19
 8      709       -14
 9      838        -8
10      753         8
# ... with 336,766 more rows
```

```
select(flights, ends_with("time"))
```

```
# A tibble: 336,776 x 5
   dep_time sched_dep_time arr_time sched_arr_time air_time
      <int>          <int>    <int>          <int>    <dbl>
 1      517            515      830            819      227
 2      533            529      850            830      227
 3      542            540      923            850      160
 4      544            545     1004           1022      183
 5      554            600      812            837      116
 6      554            558      740            728      150
 7      555            600      913            854      158
 8      557            600      709            723       53
 9      557            600      838            846      140
10      558            600      753            745      138
# ... with 336,766 more rows
```

```
select(flights, contains("dep"))
```

```
# 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
```

- [참고 2]

1. all_of : - 다 보여주지만 원소 중에 해당되지 않은 변수가 있으면 출력을 아무것도 안해준다. - NA가 있으면 출력을 안해준다.
2. any_of : - 해당하는 원소가 데이터에 없다하더라도 출력해준다. - NA가 있어도 출력을 해준다.

[Your turn 03]

---

1. Brainstorm as many ways as possible to select dep_time, dep_delay, arr_time, and arr_delay from flights.

```
select(flights,starts_with('dep'),starts_with('arr'))
```

```
# A tibble: 336,776 x 4
   dep_time dep_delay arr_time arr_delay
      <int>     <dbl>    <int>     <dbl>
 1      517         2      830        11
 2      533         4      850        20
 3      542         2      923        33
 4      544        -1     1004       -18
 5      554        -6      812       -25
 6      554        -4      740        12
 7      555        -5      913        19
 8      557        -3      709       -14
 9      557        -3      838        -8
10      558        -2      753         8
# ... with 336,766 more rows
```

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

```
select(flights,dep_time,dep_time,dep_time)
```

```
# A tibble: 336,776 x 1
   dep_time
      <int>
 1      517
 2      533
 3      542
 4      544
 5      554
 6      554
 7      555
 8      557
 9      557
10      558
# ... with 336,766 more rows
```

그냥 하나만 출력한다.

3. What does the one_of() function do? Why might it be helpful in conjunction with this vector?

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

```
# A tibble: 336,776 x 5
    year month   day dep_delay arr_delay
   <int> <int> <int>     <dbl>     <dbl>
 1  2013     1     1         2        11
 2  2013     1     1         4        20
 3  2013     1     1         2        33
 4  2013     1     1        -1       -18
 5  2013     1     1        -6       -25
 6  2013     1     1        -4        12
 7  2013     1     1        -5        19
 8  2013     1     1        -3       -14
 9  2013     1     1        -3        -8
10  2013     1     1        -2         8
# ... with 336,766 more rows
```

vars안에 있는 것들을 보여준다. vars에 해당하는 어느 하나라도 출력한다.

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"))
```

```
# A tibble: 336,776 x 6
   dep_time sched_dep_time arr_time sched_arr_time air_time time_hour
      <int>          <int>    <int>          <int>    <dbl> <dttm>
 1      517            515      830            819      227 2013-01-01 05:00:00
 2      533            529      850            830      227 2013-01-01 05:00:00
 3      542            540      923            850      160 2013-01-01 05:00:00
 4      544            545     1004           1022      183 2013-01-01 05:00:00
 5      554            600      812            837      116 2013-01-01 06:00:00
 6      554            558      740            728      150 2013-01-01 05:00:00
 7      555            600      913            854      158 2013-01-01 06:00:00
 8      557            600      709            723       53 2013-01-01 06:00:00
 9      557            600      838            846      140 2013-01-01 06:00:00
10      558            600      753            745      138 2013-01-01 06:00:00
# ... with 336,766 more rows
```

contains라는 함수는 default가 ignore.case=T라서 대문자이냐 소문자이냐에 영향을 받지 않는다. 영향을 받게 하려면 F로 바꾸면 된다.

4. mutate function

[example]

```
flights_sml <- select(flights,
                      year:day,
                      ends_with("delay"),
                      distance,
                      air_time
)

flights_sml
```

```
# A tibble: 336,776 x 7
    year month   day dep_delay arr_delay distance air_time
   <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl>
 1  2013     1     1         2        11     1400      227
 2  2013     1     1         4        20     1416      227
 3  2013     1     1         2        33     1089      160
 4  2013     1     1        -1       -18     1576      183
 5  2013     1     1        -6       -25      762      116
 6  2013     1     1        -4        12      719      150
 7  2013     1     1        -5        19     1065      158
 8  2013     1     1        -3       -14      229       53
 9  2013     1     1        -3        -8      944      140
10  2013     1     1        -2         8      733      138
# ... with 336,766 more rows
```

```
mutate(flights_sml,
       gain = dep_delay - arr_delay,
       speed = distance / air_time - 60
)
```

```
# A tibble: 336,776 x 9
    year month   day dep_delay arr_delay distance air_time  gain speed
   <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl> <dbl> <dbl>
 1  2013     1     1         2        11     1400      227    -9 -53.8
 2  2013     1     1         4        20     1416      227   -16 -53.8
 3  2013     1     1         2        33     1089      160   -31 -53.2
 4  2013     1     1        -1       -18     1576      183    17 -51.4
 5  2013     1     1        -6       -25      762      116    19 -53.4
 6  2013     1     1        -4        12      719      150   -16 -55.2
 7  2013     1     1        -5        19     1065      158   -24 -53.3
 8  2013     1     1        -3       -14      229       53    11 -55.7
 9  2013     1     1        -3        -8      944      140     5 -53.3
10  2013     1     1        -2         8      733      138   -10 -54.7
# ... with 336,766 more rows
```

```
mutate(flights_sml,
       gain = dep_delay - arr_delay,
       hours = air_time / 60,
       gain_per_hour = gain / hours
)
```

```
# A tibble: 336,776 x 10
    year month   day dep_delay arr_delay distance air_time  gain hours
   <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl> <dbl> <dbl>
 1  2013     1     1         2        11     1400      227    -9 3.78
 2  2013     1     1         4        20     1416      227   -16 3.78
 3  2013     1     1         2        33     1089      160   -31 2.67
 4  2013     1     1        -1       -18     1576      183    17 3.05
 5  2013     1     1        -6       -25      762      116    19 1.93
 6  2013     1     1        -4        12      719      150   -16 2.5
 7  2013     1     1        -5        19     1065      158   -24 2.63
 8  2013     1     1        -3       -14      229       53    11 0.883
 9  2013     1     1        -3        -8      944      140     5 2.33
10  2013     1     1        -2         8      733      138   -10 2.3
# ... with 336,766 more rows, and 1 more variable: gain_per_hour <dbl>
```

- [참고] If you only want to keep the new variables, use transmute()

[Your turn 04]

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.

```
chan_time <- function(x){

    new_time <- sprintf("%04d", x)
    new_time <- ifelse(new_time == "2400","0000",new_time)

    hour = as.integer(substr(new_time,1,2))
    minutes = as.integer(substr(new_time,3,4))

    new_time = 60*hour + minutes
    return(new_time)

}

flights %>% mutate(new_dep_time = chan_time(dep_time),
                   new_sched_dep_time = chan_time(sched_dep_time)) %>%
  select(dep_time,new_dep_time,sched_dep_time,new_sched_dep_time) %>%
  arrange(-dep_time)
```

```
# A tibble: 336,776 x 4
   dep_time new_dep_time sched_dep_time new_sched_dep_time
      <int>        <dbl>          <int>              <dbl>
 1     2400            0           2359               1439
 2     2400            0           2359               1439
 3     2400            0           2359               1439
 4     2400            0           2359               1439
 5     2400            0           2250               1370
 6     2400            0           2359               1439
 7     2400            0           2359               1439
 8     2400            0           1700               1020
 9     2400            0           2359               1439
10     2400            0           2359               1439
# ... with 336,766 more rows
```

새로운 시간을 만들어 주기 위해 chan_time 함수를 만들었다. 결과는 위와 같다.

- [다른 방법] : 나머지와 몫을 이용해 시간과 분을 구하고 2400를 변환한 1440의 값은 나눈 나머지 값을 이용해 다시 표현하는 방법.

```
flights_times <- mutate(flights,
                        dep_time_mins = (dep_time %/% 100 * 60 + dep_time %% 100) %% 1440,
                        sched_dep_time_mins = (sched_dep_time %/% 100 * 60 +
                                               sched_dep_time %% 100) %% 1440
)
flights_times %>% select(dep_time, dep_time_mins)
```

```
# A tibble: 336,776 x 2
   dep_time dep_time_mins
      <int>         <dbl>
 1      517           317
 2      533           333
 3      542           342
 4      544           344
 5      554           354
 6      554           354
 7      555           355
 8      557           357
 9      557           357
10      558           358
# ... with 336,766 more rows
```

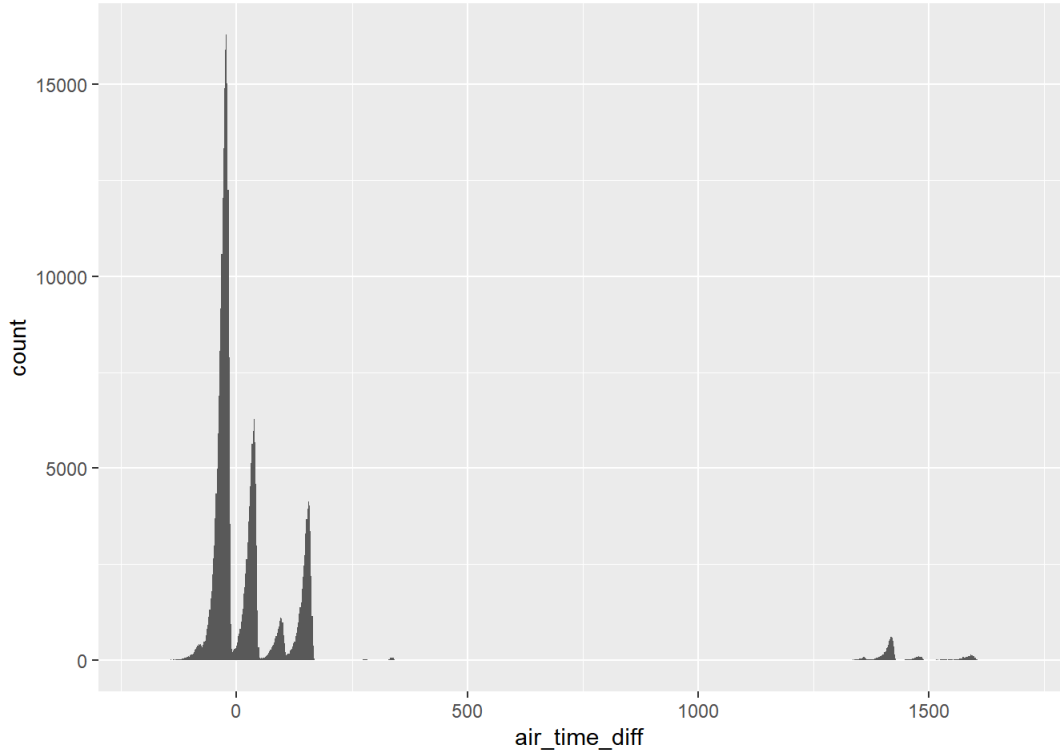
2. Compare air_time with arr_time - dep_time. What do you expect to see? What do you see? What do you need to do to fix it?

```
flights_times %>% mutate(
                        arr_time_mins = chan_time(arr_time),
                        dep_time_mins = chan_time(dep_time),
                        air_time_diff = air_time - (arr_time_mins - dep_time_mins)) %>%
  filter(air_time_diff != 0) %>% nrow()
```

```
[1] 327150
```

air_time_diff와 air_time의 값이 같아야 하는 다른 게 327150개나 있다.

```
flights_times %>% mutate(
                        arr_time_mins = chan_time(arr_time),
                        dep_time_mins = chan_time(dep_time),
                        air_time_diff = air_time - (arr_time_mins - dep_time_mins)) %>%
  ggplot(aes(x=air_time_diff))+
  geom_histogram(binwidth = 2)
```



비행을 하는 도중에 자정이 지

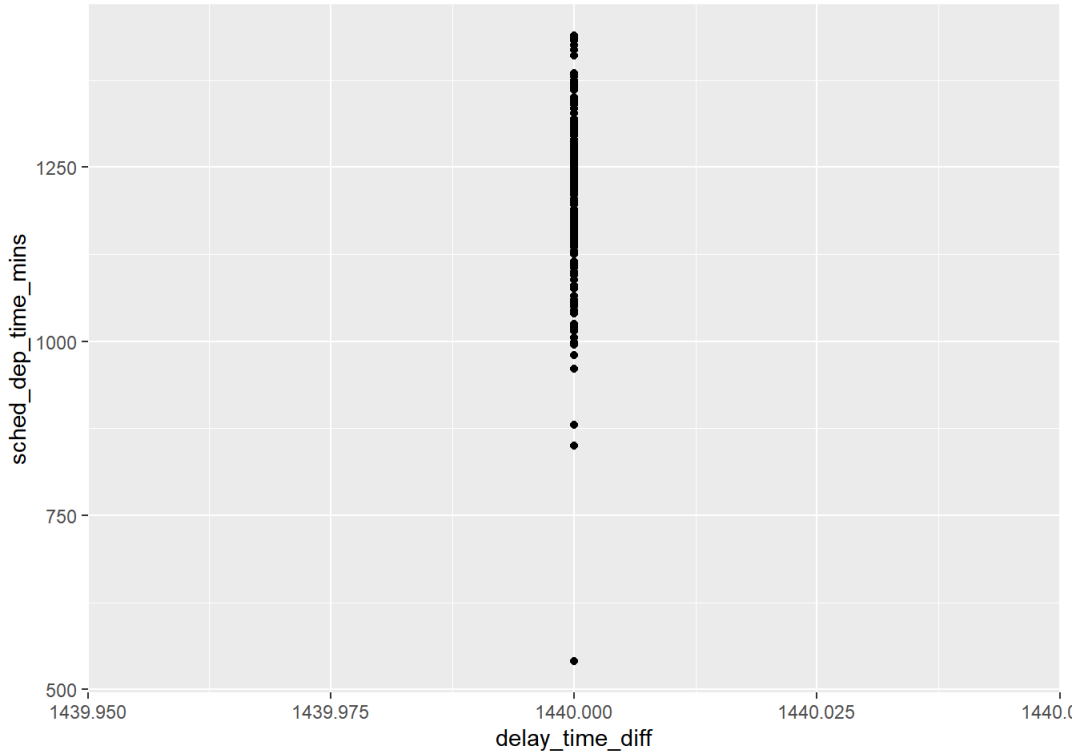
나게 되면 시간의 차이가 발생한다. (정확하게 이유를 모르겠다.)

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

```
flights_times %>% mutate(
  dep_time_mins = chan_time(dep_time),
  sched_dep_time_mins = chan_time(sched_dep_time),
  delay_time_diff = dep_delay - (dep_time_mins - sched_dep_time_mins)) %>%
  filter(delay_time_diff != 0) %>% nrow()
```

```
[1] 1236
```

```
flights_times %>% mutate(
  dep_time_mins = chan_time(dep_time),
  sched_dep_time_mins = chan_time(sched_dep_time),
  delay_time_diff = dep_delay - (dep_time_mins - sched_dep_time_mins)) %>%
  filter(delay_time_diff != 0 & !is.na(delay_time_diff)) %>%
  ggplot(aes(x=delay_time_diff,y = sched_dep_time_mins))+
  geom_point()
```



delay_time과 delay_time_diff

도 차이가 나는 게 1236개가 있다. delay_time_diff는 1440의 값을 가지는데 자정을 지났을 때 발생하기 때문이다. (정확하게 이유를 모르겠다.)

5. summaries function

[exmple]

- dep_delay의 평균값

```
summarise(flights, delay = mean(dep_delay, na.rm = TRUE))
```

```
# A tibble: 1 x 1
  delay
  <dbl>
1  12.6
```

- 365일 즉 day에 대한 평균 delay

```
by_day <- group_by(flights, year, month, day)
summarise(by_day, delay = mean(dep_delay, na.rm = TRUE))
```

```
# A tibble: 365 x 4
# Groups:   year, month [12]
    year month   day delay
   <int> <int> <int> <dbl>
 1  2013     1     1 11.5
 2  2013     1     2 13.9
 3  2013     1     3 11.0
 4  2013     1     4  8.95
 5  2013     1     5  5.73
 6  2013     1     6  7.15
 7  2013     1     7  5.42
 8  2013     1     8  2.55
 9  2013     1     9  2.28
10  2013     1    10  2.84
# ... with 355 more rows
```

```
by_day
```

```
# A tibble: 336,776 x 19
# Groups:   year, month, day [365]
    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>
```

- dest를 그룹화한 정보

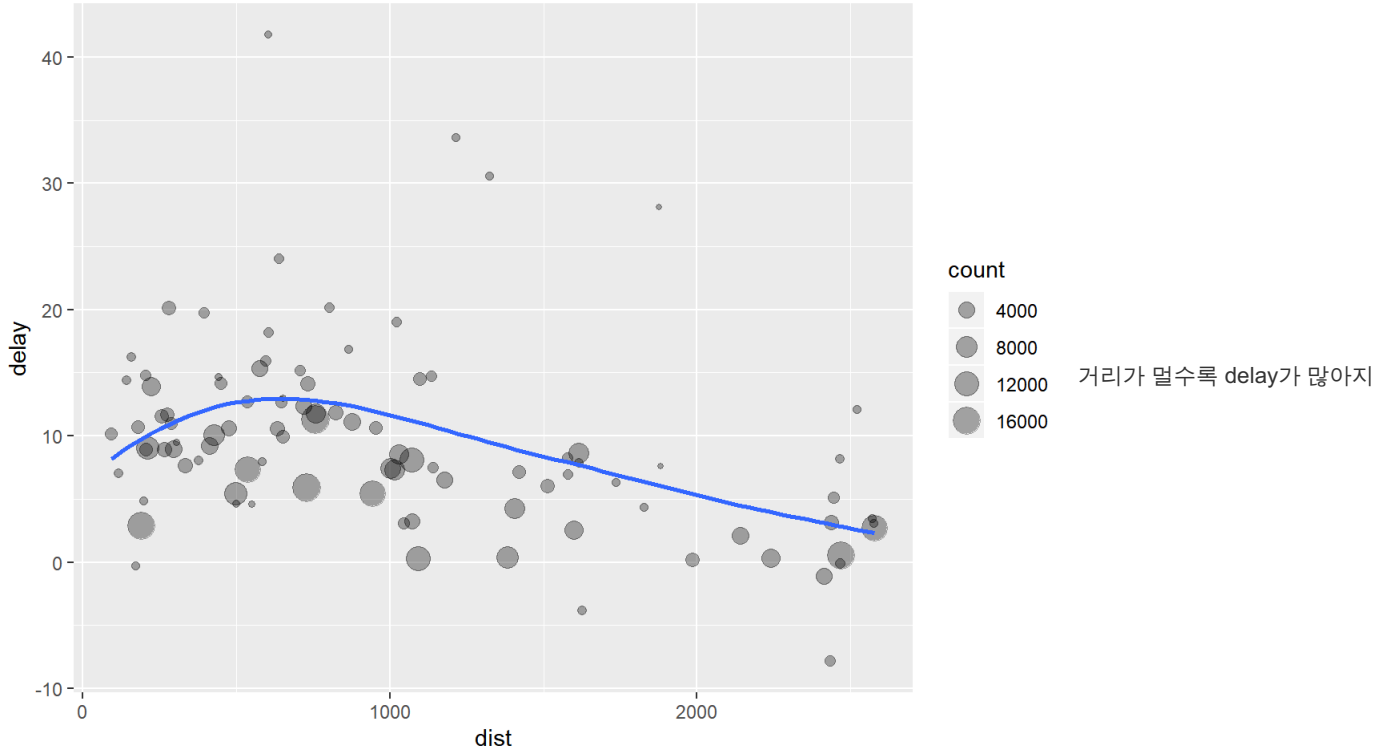
```
by_dest <- group_by(flights, dest)#105개에 대해서 그룹을 지었다.

delay <- summarise(by_dest,
                   count = n(), #그룹에 대해 몇개의 데이터가 있는지 갯수를 저장해준다.
                   dist = mean(distance, na.rm = TRUE), #거리의 mean 변수
                   delay = mean(arr_delay, na.rm = TRUE) #arr_delay의 mean변수
)
delay
```

```
# A tibble: 105 x 4
   dest  count  dist delay
   <chr> <int> <dbl> <dbl>
 1 ABQ     254 1826   4.38
 2 ACK     265  199   4.85
 3 ALB     439  143  14.4
 4 ANC       8 3370  -2.5
 5 ATL   17215  757. 11.3
 6 AUS    2439 1514.  6.02
 7 AVL     275  584.  8.00
 8 BDL     443  116   7.05
 9 BGR     375  378   8.03
10 BHM     297  866. 16.9
# ... with 95 more rows
```

dest에 따라 비행기의 수, 거리의 평균, arr_delay의 평균을 보여준다. NA값이 있을 수도 있으니 항상 'na.rm=TRUE'를 해줘야 한다. NA를 포함하게 되면 오류를 출력하는 경우가 많다.

- 멀리 날아가면 비행기도 딜레이가 심한가 알아보기 위한 그림

```
flights %>%
  group_by(dest) %>%
  summarise(
    count = n(),
    dist = mean(distance, na.rm = TRUE),
    delay = mean(arr_delay, na.rm = TRUE)
  ) %>%
  filter(count > 20, dest != "HNL") %>%
  ggplot(mapping = aes(x = dist, y = delay)) +
  geom_point(aes(size = count), alpha = 1/3) + #size는 count에 따라 주겠다.
  geom_smooth(se = FALSE)
```



는 가를 보았는데 딱히 그런 것은 아니다. 오히려 감소하는 추세이다.

[Your turn 05]

1. Come up with another approach that will give you the same output as not_cancelled %>% count(dest) and not_cancelled %>% count(tailnum, wt = distance) (without using count()).

dep_delay와 arr_delay가 NA값이 아닌 정보. 즉, 취소되지 않은 항공을 의미한다.

```
not_cancelled <- flights %>%
  filter(!is.na(dep_delay), !is.na(arr_delay))

not_cancelled %>% count(dest)
```

```
# A tibble: 104 x 2
   dest      n
   <chr> <int>
 1 ABQ     254
 2 ACK     264
 3 ALB     418
 4 ANC       8
 5 ATL   16837
 6 AUS    2411
 7 AVL     261
 8 BDL     412
 9 BGR     358
10 BHM     269
# ... with 94 more rows
```

```
not_cancelled %>% group_by(dest)%>% summarise(n = n())
```

```
# A tibble: 104 x 2
   dest      n
   <chr> <int>
 1 ABQ     254
 2 ACK     264
 3 ALB     418
 4 ANC       8
 5 ATL   16837
 6 AUS    2411
 7 AVL     261
 8 BDL     412
 9 BGR     358
10 BHM     269
# ... with 94 more rows
```

```
not_cancelled %>%
  count(tailnum, wt = distance) #wt : 가중치
```

```
# A tibble: 4,037 x 2
   tailnum      n
   <chr>    <dbl>
 1 D942DN    3418
 2 N0EGMQ  239143
 3 N10156  109664
 4 N102UW   25722
 5 N103US   24619
 6 N104UW   24616
 7 N10575  139903
 8 N105UW   23618
 9 N107US   21677
10 N108UW   32070
# ... with 4,027 more rows
```

```
not_cancelled %>% group_by(tailnum) %>%
  summarise(n = sum(distance))
```

```
# A tibble: 4,037 x 2
   tailnum      n
   <chr>    <dbl>
 1 D942DN    3418
 2 N0EGMQ  239143
 3 N10156  109664
 4 N102UW   25722
 5 N103US   24619
 6 N104UW   24616
 7 N10575  139903
 8 N105UW   23618
 9 N107US   21677
10 N108UW   32070
# ... with 4,027 more rows
```

wt는 해당 변수에 대한 가중치를 준다. count를 사용하지 않을 때는 sum()을 이용하면 같은 결과가 나온다.

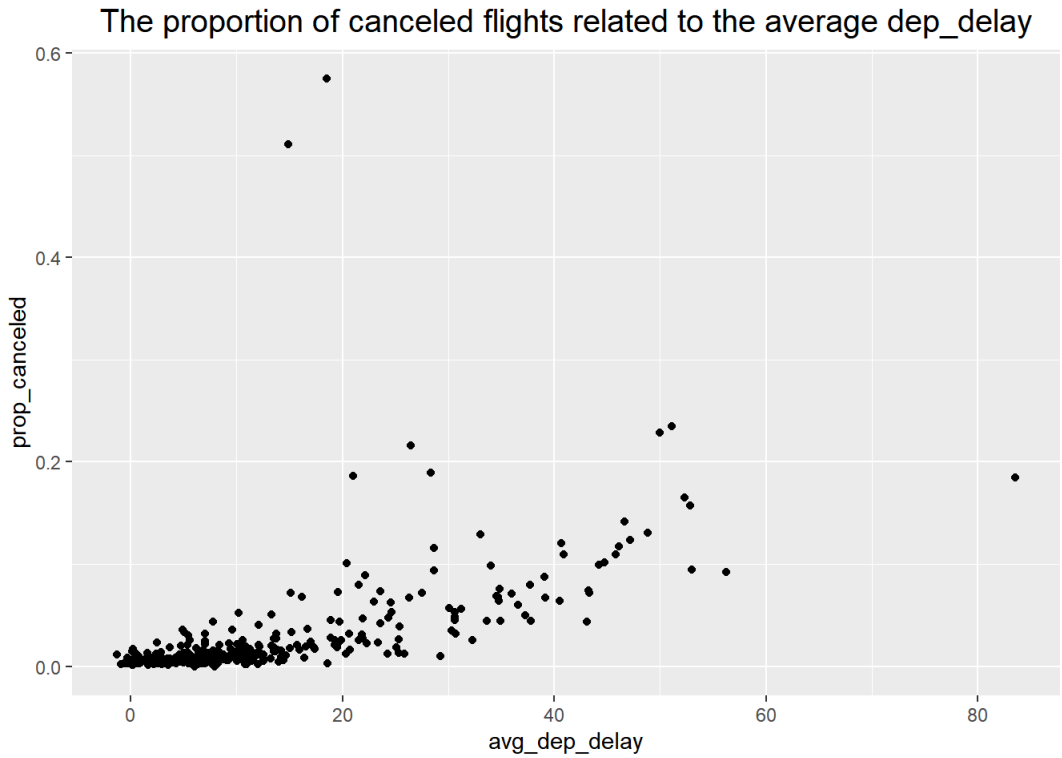
2. Look at the number of canceled flights per day. Is there a pattern? Is the proportion of canceled flights related to the average delay?

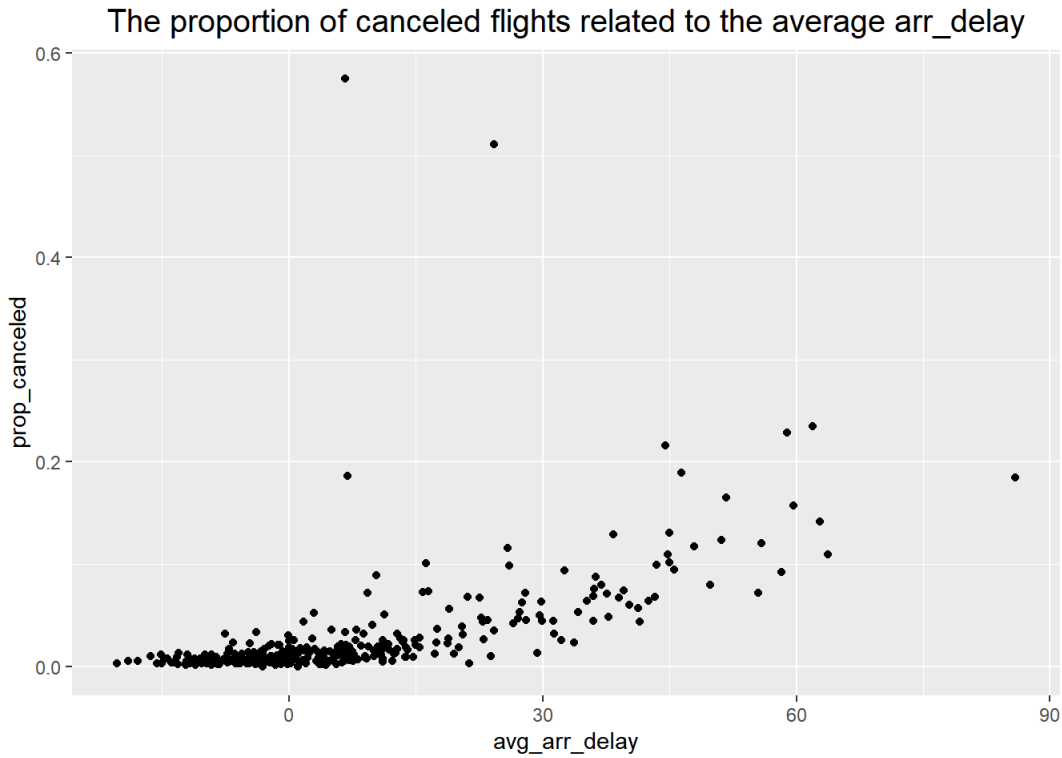
```
flights %>%
  mutate( canceled = is.na(dep_delay)|is.na(arr_delay)) %>%
  group_by(year, month, day) %>%
  summarise(
            num_flights = sum(flight),
            canceled_flights = n(),
            avg_dep_delay = mean(dep_delay,na.rm=TRUE),
            avg_arr_delay = mean(arr_delay,na.rm=TRUE),
            prop_canceled = mean(canceled))  -> new_flights
new_flights
```

```
# A tibble: 365 x 8
# Groups:   year, month [12]
    year month   day num_flights canceled_flights avg_dep_delay avg_arr_delay
   <int> <int> <int>       <int>            <int>         <dbl>         <dbl>
 1  2013     1     1     1533700              842         11.5          12.7
 2  2013     1     2     1808422              943         13.9          12.7
 3  2013     1     3     1748643              914         11.0           5.73
 4  2013     1     4     1766085              915          8.95         -1.93
 5  2013     1     5     1252814              720          5.73         -1.53
 6  2013     1     6     1583123              832          7.15          4.24
 7  2013     1     7     1859993              933          5.42         -4.95
 8  2013     1     8     1805441              899          2.55         -3.23
 9  2013     1     9     1819691              902          2.28         -0.264
10  2013     1    10     1863369              932          2.84         -5.90
# ... with 355 more rows, and 1 more variable: prop_canceled <dbl>
```

day에 따라 취소한 항공의 개수는 위와 같다. 지연된 시간(dep and arr)의 평균값이 취소된 항공의 비율과 관련을 알아보기 위해 아래의 그림을 그렸다.

```
#avg_dep_delay vs prop_canceled & avg_arr_delay vs prop_canceled
new_flights %>% ggplot(aes(x=avg_dep_delay, y = prop_canceled)) +
  geom_point()+
  ggtitle("The proportion of canceled flights related to the average dep_delay")+
  theme(plot.title = element_text(size=15,hjust = 0.5)) -> p1

new_flights %>% ggplot(aes(x=avg_arr_delay, y = prop_canceled)) +
  geom_point()+
  ggtitle("The proportion of canceled flights related to the average arr_delay")+
  theme(plot.title = element_text(size=15,hjust = 0.5)) -> p2

#library("gridExtra")
#grid.arrange(p1, p2, ncol = 1, nrow = 2)
p1;p2
```





지연된 시간의 평균값이 클수록 취소될 확률이 높아지는 것을 보여준다. 특히 지연 시간이 길수록 취소될 확률은 더욱 높아진다.

3. Which carrier has the worst delays? Challenge: can you disentangle the effects of bad airports vs. bad carriers? Why/why not?

```
#carrier
flights %>% group_by(carrier) %>%
  summarise(num_flights = n(),arr_delay_mean = mean(arr_delay,na.rm = TRUE),dep_delay_mean = mean(dep_delay,
na.rm = TRUE)) %>%
  arrange(desc(arr_delay_mean,dep_delay_mean))
```

```
# A tibble: 16 x 4
   carrier num_flights arr_delay_mean dep_delay_mean
   <chr>         <int>          <dbl>          <dbl>
 1 F9              685         21.9            20.2
 2 FL             3260         20.1            18.7
 3 EV            54173         15.8            20.0
 4 YV              601         15.6            19.0
 5 OO               32         11.9            12.6
 6 MQ            26397         10.8            10.6
 7 WN            12275          9.65           17.7
 8 B6            54635          9.46           13.0
 9 9E            18460          7.38           16.7
10 UA            58665          3.56           12.1
11 US            20536          2.13            3.78
12 VX             5162          1.76           12.9
13 DL            48110          1.64            9.26
14 AA            32729          0.364           8.59
15 HA              342         -6.92            4.90
16 AS              714         -9.93            5.80
```

지연이 가장 심한 항공은 F9이다.

4. Counts the number of flights to a destination and sorts them from highest to lowest.

```
flights %>%
  count(dest, sort=TRUE)
```

```
# A tibble: 105 x 2
   dest      n
   <chr> <int>
 1 ORD   17283
 2 ATL   17215
 3 LAX   16174
 4 BOS   15508
 5 MCO   14082
 6 CLT   14064
 7 SFO   13331
 8 FLL   12055
 9 MIA   11728
10 DCA    9705
# ... with 95 more rows
```

```
flights %>% count(dest) %>% arrange(desc(n))
```

```
# A tibble: 105 x 2
   dest      n
   <chr> <int>
 1 ORD   17283
 2 ATL   17215
 3 LAX   16174
 4 BOS   15508
 5 MCO   14082
 6 CLT   14064
 7 SFO   13331
 8 FLL   12055
 9 MIA   11728
10 DCA    9705
# ... with 95 more rows
```