Alexander Rasho

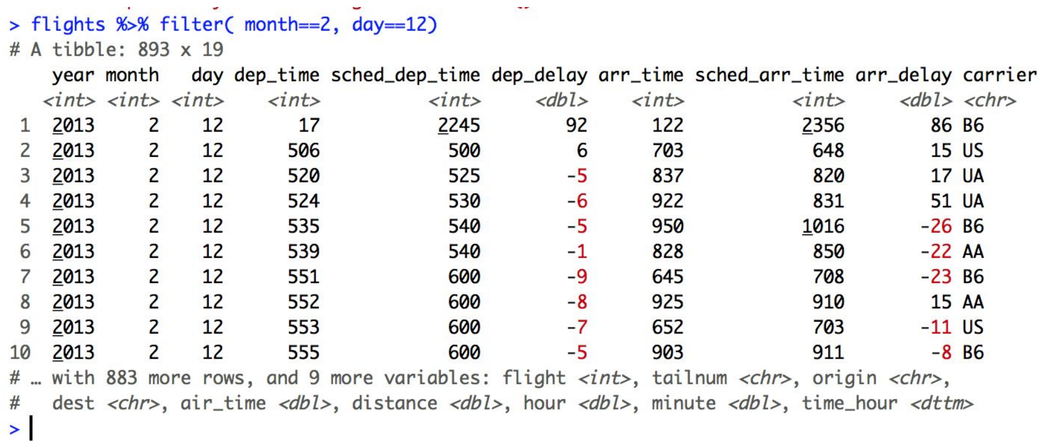
1. Data wrangling using the tidyverse (classwork problems)

Load the nycflights13 library (will have to install the nycflights13 package first) which contains flight arrival and departure data in a table called `flights`. Apply the tidyverse's data wrangling verbs to answer these questions. For each question, **give only the (one line) code**.

a. List data only for flights that departed on February 12,
   Commands:

   > flights %>% filter(month==2, day==12)

   Screen Shot:

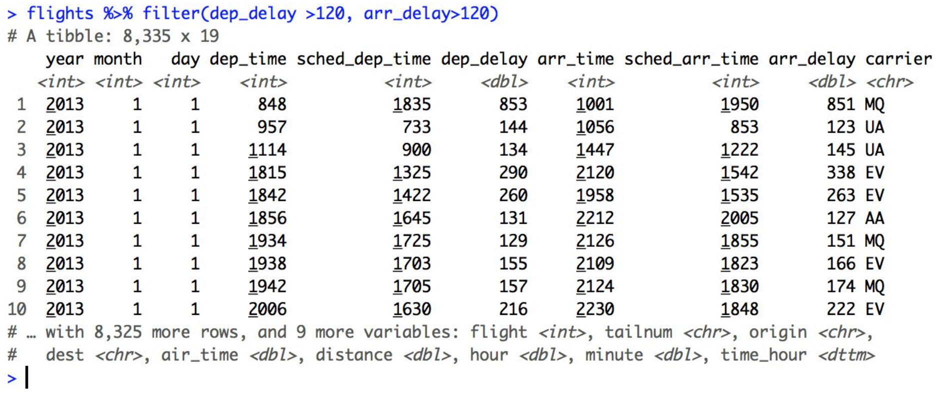


```
> flights %>% filter( month==2, day==12)
# A tibble: 893 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>  
 1  2013     2    12       17           2245        92      122           2356        86 B6     
 2  2013     2    12      506            500         6      703            648        15 US     
 3  2013     2    12      520            525        -5      837            820        17 UA     
 4  2013     2    12      524            530        -6      922            831        51 UA     
 5  2013     2    12      535            540        -5      950           1016       -26 B6     
 6  2013     2    12      539            540        -1      828            850       -22 AA     
 7  2013     2    12      551            600        -9      645            708       -23 B6     
 8  2013     2    12      552            600        -8      925            910        15 AA     
 9  2013     2    12      553            600        -7      652            703       -11 US     
10  2013     2    12      555            600        -5      903            911        -8 B6     
# … with 883 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>
> |
```

b. List data only for flights that were delayed (both arrival and departure) by more than 2 hours.
   Commands:

   > flights %>% filter(dep_delay >120, arr_delay>120)

   Screen Shot:

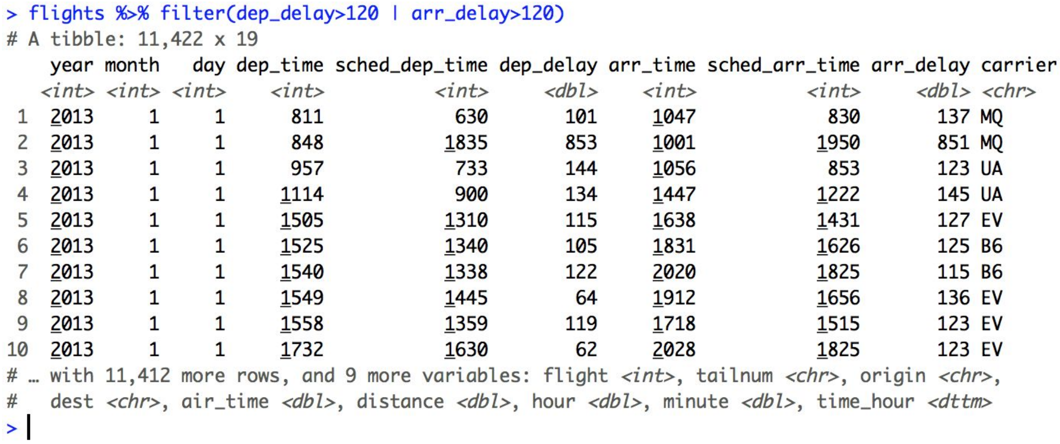


```
> flights %>% filter(dep_delay >120, arr_delay>120)
# A tibble: 8,335 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
 1  2013     1     1      848           1835       853     1001           1950       851 MQ
 2  2013     1     1      957            733       144     1056            853       123 UA
 3  2013     1     1     1114            900       134     1447           1222       145 UA
 4  2013     1     1     1815           1325       290     2120           1542       338 EV
 5  2013     1     1     1842           1422       260     1958           1535       263 EV
 6  2013     1     1     1856           1645       131     2212           2005       127 AA
 7  2013     1     1     1934           1725       129     2126           1855       151 MQ
 8  2013     1     1     1938           1703       155     2109           1823       166 EV
 9  2013     1     1     1942           1705       157     2124           1830       174 MQ
10  2013     1     1     2006           1630       216     2230           1848       222 EV
# … with 8,325 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>
> |
```

c. List data only for flights that were delayed (either arrival or departure) by more than 2 hours.
Commands:

> flights %>% filter(dep_delay>120 | arr_delay>120)

Screen Shot:

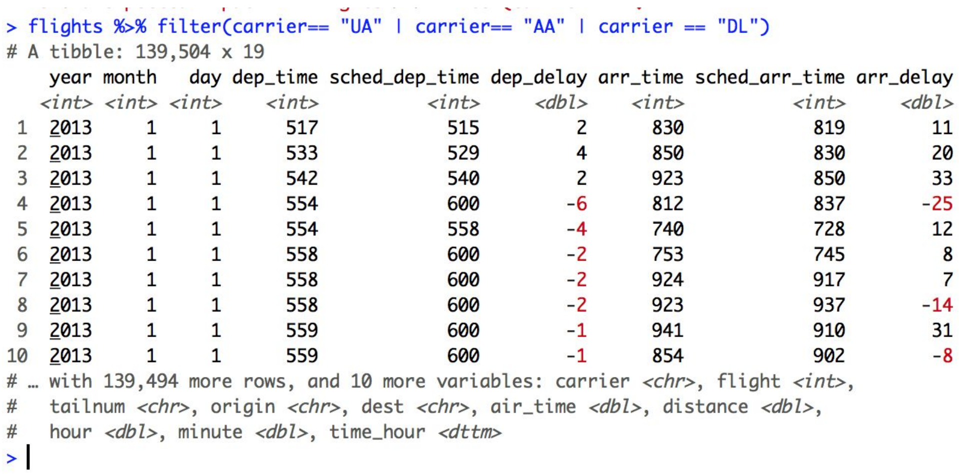


```
> flights %>% filter(dep_delay>120 | arr_delay>120)
# A tibble: 11,422 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
 1  2013     1     1      811            630       101     1047            830       137 MQ
 2  2013     1     1      848           1835       853     1001           1950       851 MQ
 3  2013     1     1      957            733       144     1056            853       123 UA
 4  2013     1     1     1114            900       134     1447           1222       145 UA
 5  2013     1     1     1505           1310       115     1638           1431       127 EV
 6  2013     1     1     1525           1340       105     1831           1626       125 B6
 7  2013     1     1     1540           1338       122     2020           1825       115 B6
 8  2013     1     1     1549           1445        64     1912           1656       136 EV
 9  2013     1     1     1558           1359       119     1718           1515       123 EV
10  2013     1     1     1732           1630        62     2028           1825       123 EV
# … with 11,412 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>
> |
```

d. List data only for flights that were operated by United, American, or Delta.
Commands:

> flights %>% filter(carrier== "UA" | carrier== "AA" | carrier == "DL")
OR
> flights %>% filter(carrier %in% c("UA", "AA", "DL"))

Screen Shot:



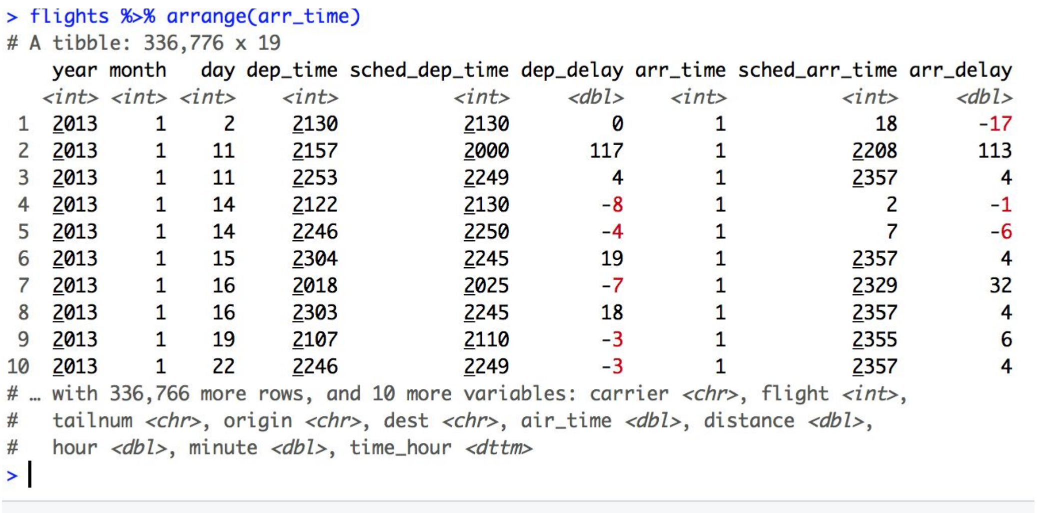
```
> flights %>% filter(carrier== "UA" | carrier== "AA" | carrier == "DL")
# A tibble: 139,504 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
 1  2013     1     1      517            515         2      830            819        11
 2  2013     1     1      533            529         4      850            830        20
 3  2013     1     1      542            540         2      923            850        33
 4  2013     1     1      554            600        -6      812            837       -25
 5  2013     1     1      554            558        -4      740            728        12
 6  2013     1     1      558            600        -2      753            745         8
 7  2013     1     1      558            600        -2      924            917         7
 8  2013     1     1      558            600        -2      923            937       -14
 9  2013     1     1      559            600        -1      941            910        31
10  2013     1     1      559            600        -1      854            902        -8
# … with 139,494 more rows, and 10 more variables: carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
> |
```

e. Sort data in order of fastest flights.

Commands:

>flights %>% arrange(arr_time)

Screen Shot:



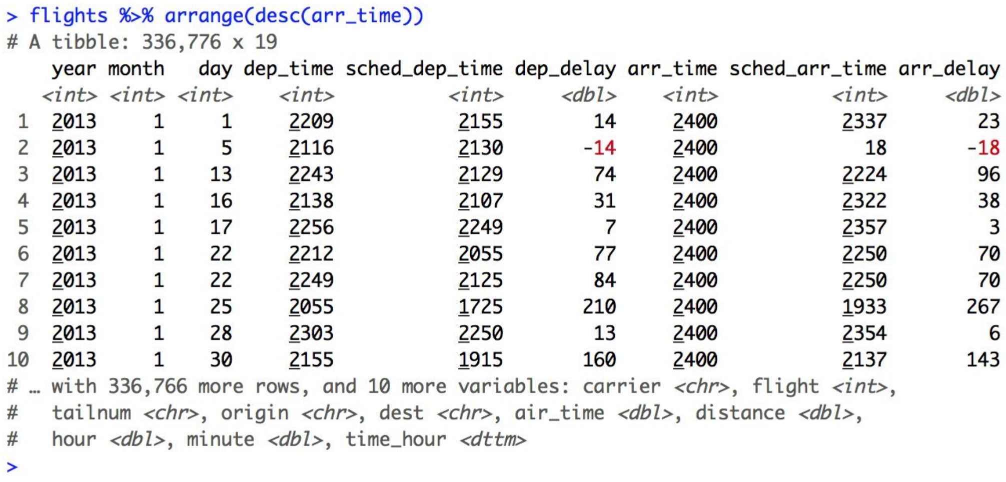
```
> flights %>% arrange(arr_time)
# A tibble: 336,776 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
 1  2013     1     2     2130           2130         0        1             18       -17
 2  2013     1    11     2157           2000       117        1           2208       113
 3  2013     1    11     2253           2249         4        1           2357         4
 4  2013     1    14     2122           2130        -8        1              2        -1
 5  2013     1    14     2246           2250        -4        1              7        -6
 6  2013     1    15     2304           2245        19        1           2357         4
 7  2013     1    16     2018           2025        -7        1           2329        32
 8  2013     1    16     2303           2245        18        1           2357         4
 9  2013     1    19     2107           2110        -3        1           2355         6
10  2013     1    22     2246           2249        -3        1           2357         4
# … with 336,766 more rows, and 10 more variables: carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
> |
```

f. Sort data in order of longest flights.

Commands:

>flights %>% arrange(desc(arr_time))

Screen Shot:

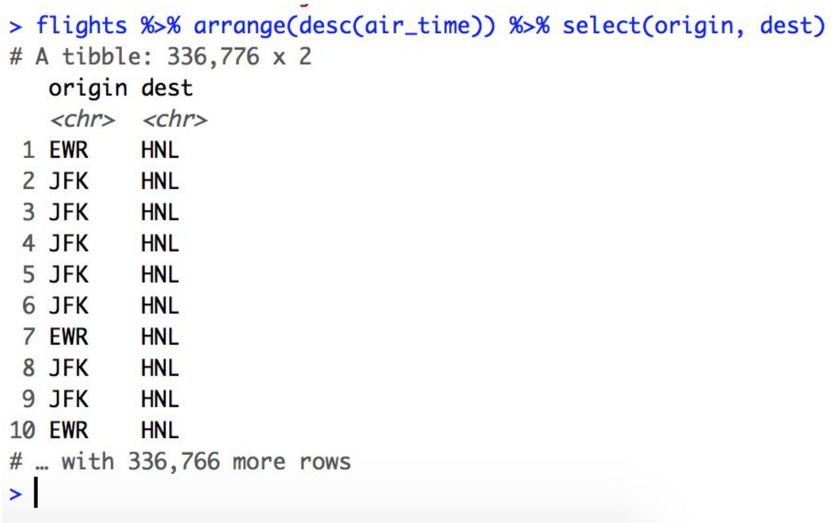


```
> flights %>% arrange(desc(arr_time))
# A tibble: 336,776 x 19
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
 1  2013     1     1     2209           2155        14     2400           2337        23
 2  2013     1     5     2116           2130       -14     2400             18       -18
 3  2013     1    13     2243           2129        74     2400           2224        96
 4  2013     1    16     2138           2107        31     2400           2322        38
 5  2013     1    17     2256           2249         7     2400           2357         3
 6  2013     1    22     2212           2055        77     2400           2250        70
 7  2013     1    22     2249           2125        84     2400           2250        70
 8  2013     1    25     2055           1725       210     2400           1933       267
 9  2013     1    28     2303           2250        13     2400           2354         6
10  2013     1    30     2155           1915       160     2400           2137       143
# … with 336,766 more rows, and 10 more variables: carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>
>
```

g. Show only the origin and destination of flights sorted by longest flights.
Commands:

>flights %>% arrange(desc(air_time)) %>% select(origin, dest)

Screen Shot:

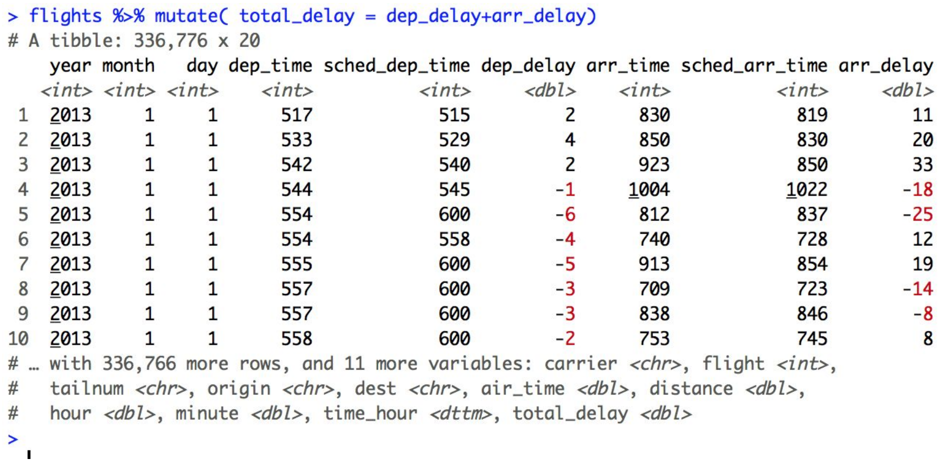


```
> flights %>% arrange(desc(air_time)) %>% select(origin, dest)
# A tibble: 336,776 x 2
   origin dest
   <chr>  <chr>
 1 EWR    HNL
 2 JFK    HNL
 3 JFK    HNL
 4 JFK    HNL
 5 JFK    HNL
 6 JFK    HNL
 7 EWR    HNL
 8 JFK    HNL
 9 JFK    HNL
10 EWR    HNL
# … with 336,766 more rows
>
```

h. Add a new variable that indicates the total delay (both departure and arrival delay).
Commands:

>flights %>% mutate(total_delay = dep_delay+arr_delay)

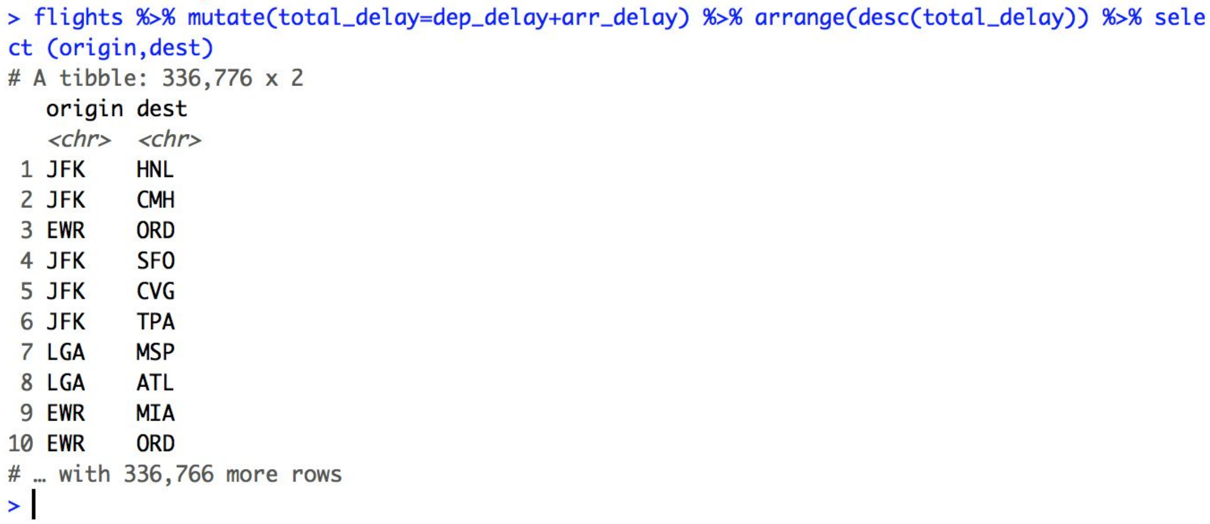
Screen Shot:



```
> flights %>% mutate( total_delay = dep_delay+arr_delay)
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
 1  2013     1     1      517            515         2      830            819        11
 2  2013     1     1      533            529         4      850            830        20
 3  2013     1     1      542            540         2      923            850        33
 4  2013     1     1      544            545        -1     1004           1022       -18
 5  2013     1     1      554            600        -6      812            837       -25
 6  2013     1     1      554            558        -4      740            728        12
 7  2013     1     1      555            600        -5      913            854        19
 8  2013     1     1      557            600        -3      709            723       -14
 9  2013     1     1      557            600        -3      838            846        -8
10  2013     1     1      558            600        -2      753            745         8
# … with 336,766 more rows, and 11 more variables: carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dttm>, total_delay <dbl>
>
```

i. Show only the origin and destination of flights sorted by descending order of total delay.
   Commands:
   >flights %>% mutate(total_delay=dep_delay+arr_delay) %>% arrange(desc(total_delay)) %>% select (origin,dest)

   Screen Shot:



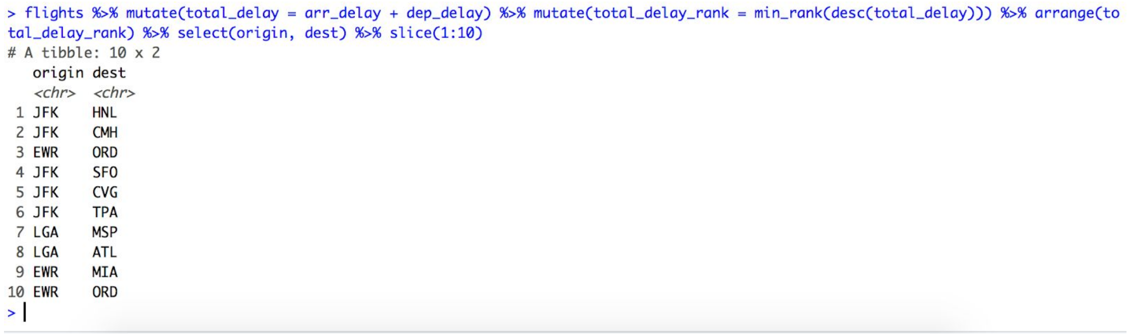
```
> flights %>% mutate(total_delay=dep_delay+arr_delay) %>% arrange(desc(total_delay)) %>% sele
ct (origin,dest)
# A tibble: 336,776 x 2
   origin dest
   <chr>  <chr>
 1 JFK    HNL
 2 JFK    CMH
 3 EWR    ORD
 4 JFK    SFO
 5 JFK    CVG
 6 JFK    TPA
 7 LGA    MSP
 8 LGA    ATL
 9 EWR    MIA
10 EWR    ORD
# … with 336,766 more rows
>
```

j. Show only the origin and destination of 10 most delayed flights (Hint: use the min_rank() function which assigns ranks 1, 2, 3, …).

Commands:

>flights %>% mutate(total_delay = arr_delay + dep_delay) %>% mutate(total_delay_rank = min_rank(desc(total_delay))) %>% arrange(total_delay_rank) %>% select(origin, dest) %>% slice(1:10) %>% view()

Screen Shot:

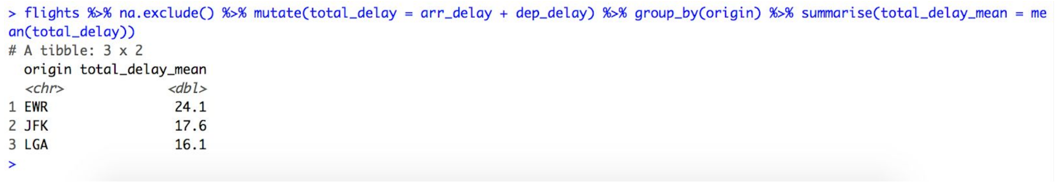


```
> flights %>% mutate(total_delay = arr_delay + dep_delay) %>% mutate(total_delay_rank = min_rank(desc(total_delay))) %>% arrange(to
tal_delay_rank) %>% select(origin, dest) %>% slice(1:10)
# A tibble: 10 x 2
   origin dest 
   <chr>  <chr>
 1 JFK    HNL  
 2 JFK    CMH  
 3 EWR    ORD  
 4 JFK    SFO  
 5 JFK    CVG  
 6 JFK    TPA  
 7 LGA    MSP  
 8 LGA    ATL  
 9 EWR    MIA  
10 EWR    ORD  
> |
```

k. Show the average total delay (excluding NA values) for every origin city.
Commands:

>flights %>% na.exclude() %>% mutate(total_delay = arr_delay + dep_delay) %>% group_by(origin) %>% summarise(total_delay_mean = mean(total_delay))

Screen Shot:

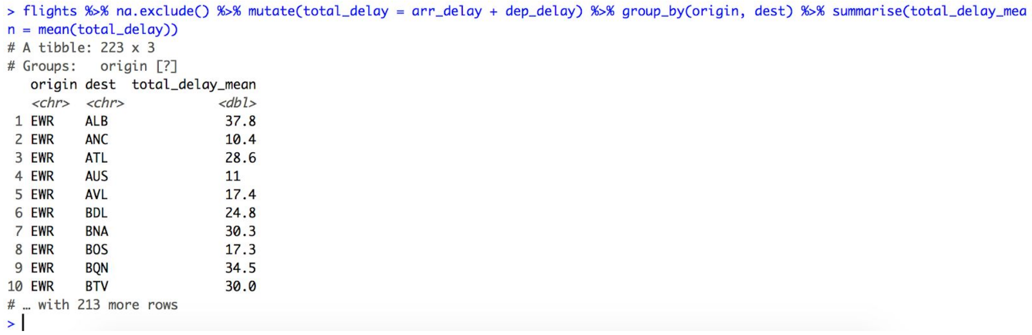


```
> flights %>% na.exclude() %>% mutate(total_delay = arr_delay + dep_delay) %>% group_by(origin) %>% summarise(total_delay_mean = me
an(total_delay))
# A tibble: 3 x 2
  origin total_delay_mean
  <chr>             <dbl>
1 EWR                24.1
2 JFK                17.6
3 LGA                16.1
> 
```

l. Show the average total delay (excluding NA values) for every origin-destination city pair.
Commands:

>flights %>% na.exclude() %>% mutate(total_delay = arr_delay + dep_delay) %>% group_by(origin, dest) %>% summarise(total_delay_mean = mean(total_delay))

Screen Shot:



```
> flights %>% na.exclude() %>% mutate(total_delay = arr_delay + dep_delay) %>% group_by(origin, dest) %>% summarise(total_delay_mea
n = mean(total_delay))
# A tibble: 223 x 3
# Groups:   origin [?]
   origin dest  total_delay_mean
   <chr>  <chr>            <dbl>
 1 EWR    ALB               37.8
 2 EWR    ANC               10.4
 3 EWR    ATL               28.6
 4 EWR    AUS               11
 5 EWR    AVL               17.4
 6 EWR    BDL               24.8
 7 EWR    BNA               30.3
 8 EWR    BOS               17.3
 9 EWR    BQN               34.5
10 EWR    BTV               30.0
# … with 213 more rows
> |
```

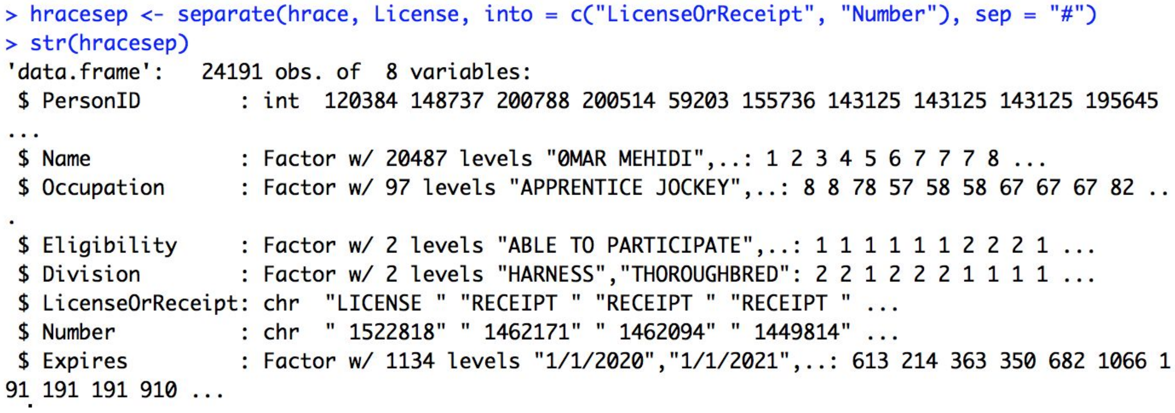
2. Data reshaping using the tidyverse

a. Consider the attached .csv file “horse_racing.csv” which contains data related to horse racing licensing in New York[1]. The `License` column has two types of values: license numbers and receipt numbers. Load the dataset and transform it such that this column is split into two:

   i. `LicenseOrReceipt`: a factor with two levels “License” and “Receipt”

   ii. `Number`: numeric column with the license/receipt number

Show (1) your code, and (2) copy & paste the output of the function **`str()`** on your final table.



```
> hracesep <- separate(hrace, License, into = c("LicenseOrReceipt", "Number"), sep = "#")
> str(hracesep)
'data.frame':	24191 obs. of  8 variables:
 $ PersonID        : int  120384 148737 200788 200514 59203 155736 143125 143125 143125 195645
...
 $ Name            : Factor w/ 20487 levels "0MAR MEHIDI",..: 1 2 3 4 5 6 7 7 7 8 ...
 $ Occupation      : Factor w/ 97 levels "APPRENTICE JOCKEY",..: 8 8 78 57 58 58 67 67 67 82 ..
.
 $ Eligibility     : Factor w/ 2 levels "ABLE TO PARTICIPATE",..: 1 1 1 1 1 1 2 2 2 1 ...
 $ Division        : Factor w/ 2 levels "HARNESS","THOROUGHBRED": 2 2 1 2 2 2 1 1 1 1 ...
 $ LicenseOrReceipt: chr  "LICENSE " "RECEIPT " "RECEIPT " "RECEIPT " ...
 $ Number          : chr  " 1522818" " 1462171" " 1462094" " 1449814" ...
 $ Expires         : Factor w/ 1134 levels "1/1/2020","1/1/2021",..: 613 214 363 350 682 1066 1
91 191 191 910 ...
```

b. Consider the attached .csv file, “language_diversity.csv,” which contains data on the diversity of languages in different countries and other parameters[2].

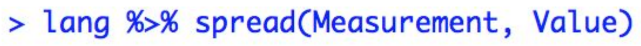
   a. Is the data “tidy”? Explain your answer in 2-3 sentences.

[1] Original dataset: https://data.ny.gov/Government-Finance/Horse-Racing-Licensing/cz9u-yj7m/data

[2] Dataset from: https://github.com/jvcasillas/untidydata#language_diversity

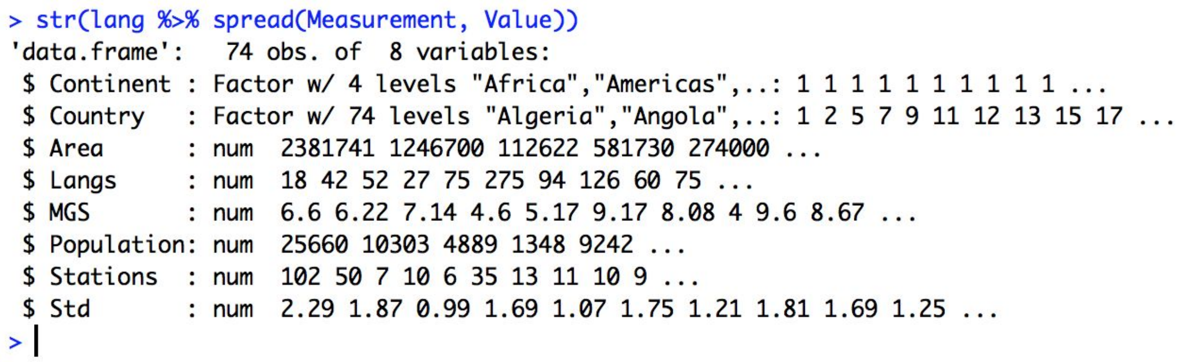
It’s not tidy because the variables MGS, Population, Stations, Std, langs and area should be made into columns instead of rows. Observations are unique values of MGS, Population, Stations, Std, langs and area in a given country and continent.

b. Convert the data to tidy data. Show (1) your code, and (2) copy & paste the output of the function `str` on your final table.
Code:



```
> lang %>% spread(Measurement, Value)
```

Output:



```
> str(lang %>% spread(Measurement, Value))
'data.frame':	74 obs. of  8 variables:
 $ Continent : Factor w/ 4 levels "Africa","Americas",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Country   : Factor w/ 74 levels "Algeria","Angola",..: 1 2 5 7 9 11 12 13 15 17 ...
 $ Area      : num  2381741 1246700 112622 581730 274000 ...
 $ Langs     : num  18 42 52 27 75 275 94 126 60 75 ...
 $ MGS       : num  6.6 6.22 7.14 4.6 5.17 9.17 8.08 4 9.6 8.67 ...
 $ Population: num  25660 10303 4889 1348 9242 ...
 $ Stations  : num  102 50 7 10 6 35 13 11 10 9 ...
 $ Std       : num  2.29 1.87 0.99 1.69 1.07 1.75 1.21 1.81 1.69 1.25 ...
> |
```

c. Consider the attached .csv file, “diseases.csv,” which contains data from Australia on hospitalizations[3].

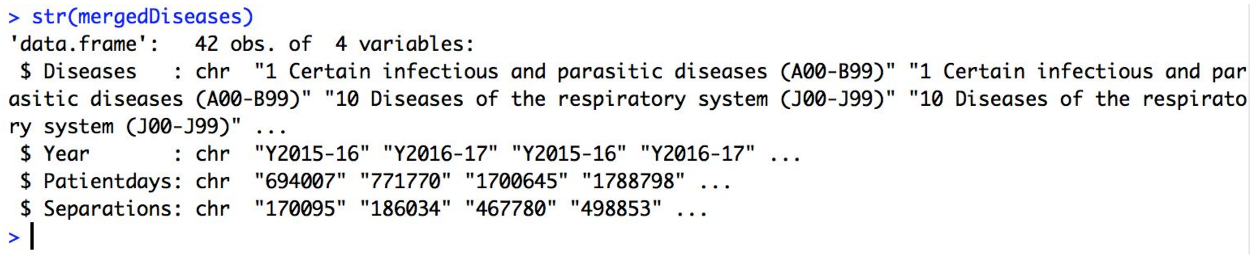
Diseases	**Patientdays_Y2015-16**	**Separations_Y2015-16**	**Patientdays_Y2016-17**	**Separations_Y2016-17**
1 Certain infectious and parasitic diseases (A00-B99)	694,007	170,095	771,770	186,034
2 Neoplasms (C00-D48)	2,223,563	666,594	2,235,045	684,075
3 Diseases of the blood and blood−forming organs and certain disorders involving the immune mechanism (D50-D89)	317,085	175,590	335,699	190,568

The first few rows are shown above. Load this file and convert the table to the tidy format shown below. Note the new column names. Show (1) your code, and (2) copy & paste the output of the function `str` on your final table. (*Hint: this will require multiple transforms from gather/separate/select. Read the file with **read_csv**, not read.csv*)

[3] Dataset from: https://www.aihw.gov.au/reports/hospitals/principal-diagnosis-data-cubes/contents/data-cubes

Diseases	Year	Patientdays	Separations
1 Certain infectious and parasitic diseases (A00-B99)	Y2015-16	694,007	170,095
1 Certain infectious and parasitic diseases (A00-B99)	Y2016-17	771,770	186,034
2 Neoplasms (C00-D48)	Y2015-16	2,223,563	666,594
2 Neoplasms (C00-D48)	Y2016-17	2,235,045	684,075

```
> diseasestemp <- select(diseases, c(1,2,4))
> diseasestemp2 <- select(diseases, c(1,3,5))
> fixdiseases1 <- separate(diseasestemp, "Patientdays_Y2015-16",
into=c("Y2015-16","Patientdays"), sep="_")
> fixdiseases1 <- separate(fixdiseases1, "Patientdays_Y2016-17",
into=c("Y2016-17","Patientdays"), sep="_")
> fixdiseases1 <- gather(fixdiseases1, Year, Patientdays, c(2:4))
> fixdiseases1 <- fixdiseases1[-c(43:63),]  #deletes extra rows that were created
> fixdiseases2 <- separate(diseasestemp2, "Separations_Y2015-16",
into=c("Y2015-16","Separations"), sep="_")
> fixdiseases2 <- gather(fixdiseases2, Year, Separations, c(2:4))
> fixdiseases2 <- fixdiseases2[-c(43:63),] #deletes extra rows that were created
> mergedDiseases <- merge(fixdiseases1, fixdiseases2, by=c("Diseases", "Year")) #merge the 2
seperate datasets into the complete dataset
```



```
> str(mergedDiseases)
'data.frame':   42 obs. of  4 variables:
 $ Diseases   : chr  "1 Certain infectious and parasitic diseases (A00-B99)" "1 Certain infectious and par
asitic diseases (A00-B99)" "10 Diseases of the respiratory system (J00-J99)" "10 Diseases of the respirato
ry system (J00-J99)" ...
 $ Year       : chr  "Y2015-16" "Y2016-17" "Y2015-16" "Y2016-17" ...
 $ Patientdays: chr  "694007" "771770" "1700645" "1788798" ...
 $ Separations: chr  "170095" "186034" "467780" "498853" ...
> 
```