Handling Data with dplyr

Data Analysis and Visualization (Fall 2019)

Instructor: Debopriya Ghosh

Remove flights that were cancelled

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.0     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
not_cancelled = flights %>%
  filter(!is.na(dep_delay), !is.na(arr_delay))
```

Summarize based on avg. departure delay per day

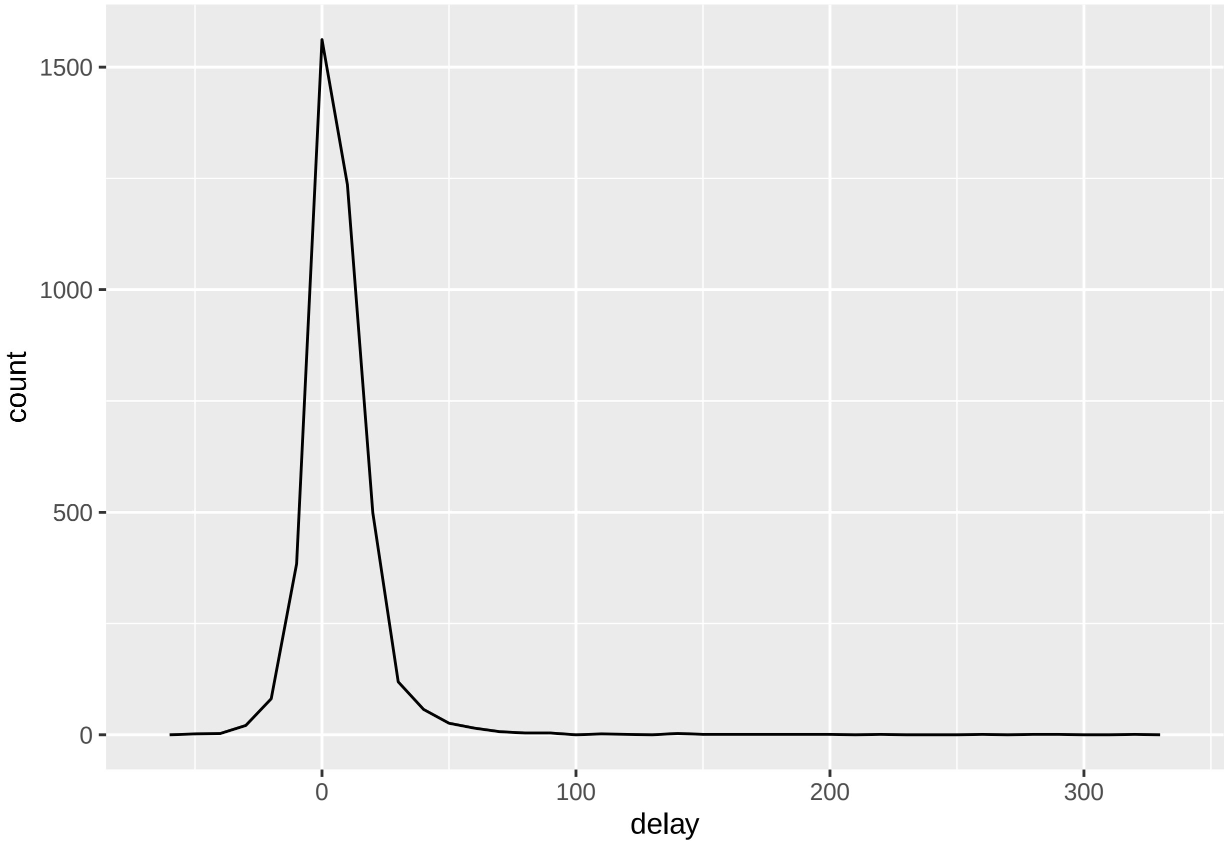
```
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(mean = mean(dep_delay))
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day  mean
##    <int> <int> <int> <dbl>
##  1  2013     1     1 11.4
##  2  2013     1     2 13.7
##  3  2013     1     3 10.9
##  4  2013     1     4  8.97
##  5  2013     1     5  5.73
##  6  2013     1     6  7.15
##  7  2013     1     7  5.42
##  8  2013     1     8  2.56
##  9  2013     1     9  2.30
## 10  2013     1    10  2.84
## # ... with 355 more rows
```

Look at planes (identified by their tail number) that have the highest average delays.

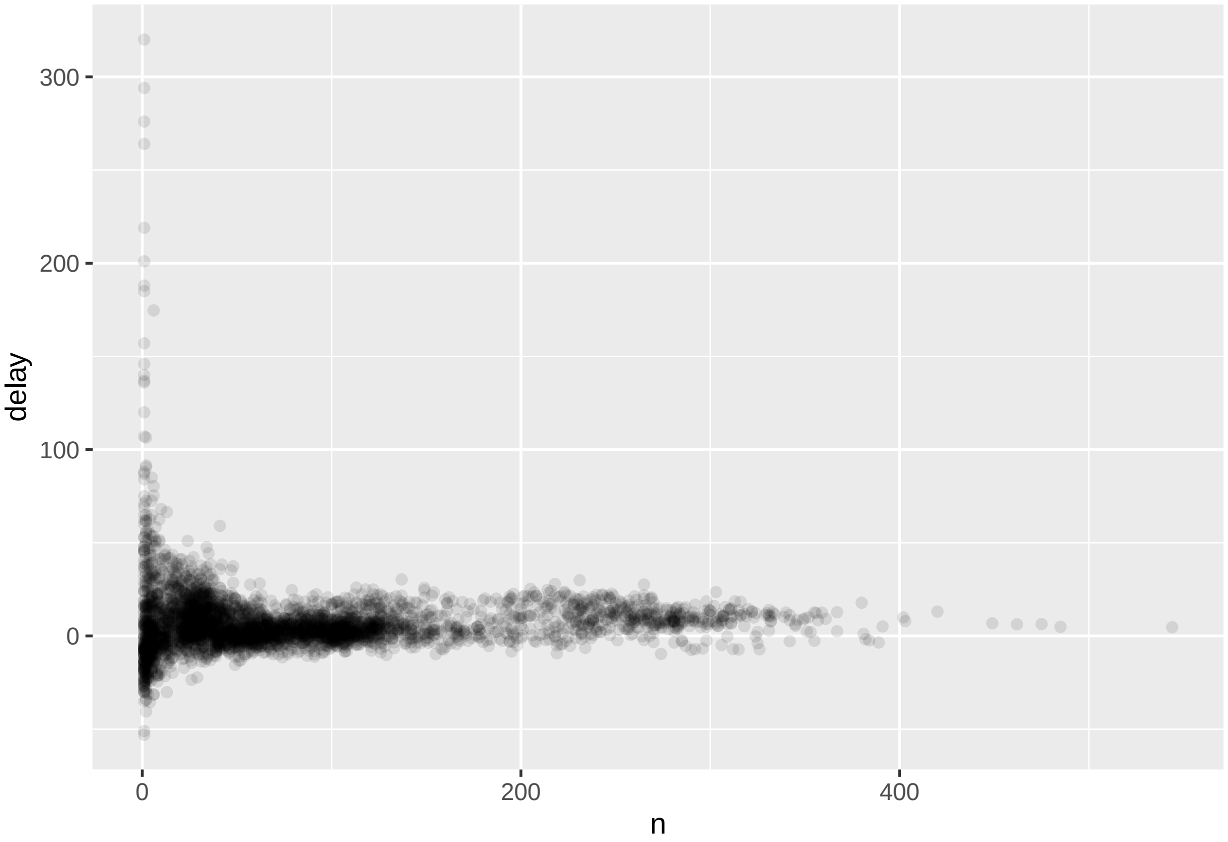
```
delays = not_cancelled %>%
  group_by(tailnum) %>%
  summarize(
    delay = mean(arr_delay)
  )

ggplot(data = delays, mapping = aes(x = delay)) +
  geom_freqpoly(binwidth = 10)
```



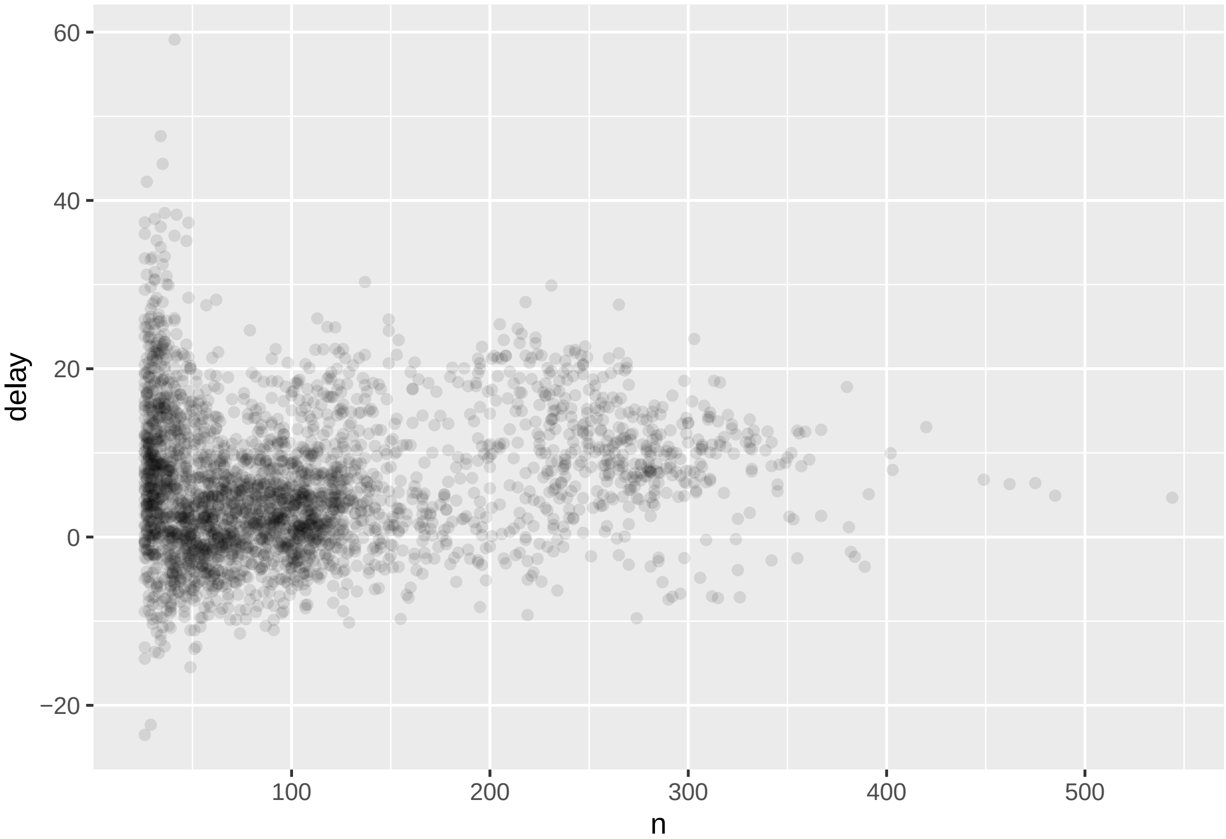
Draw the scatterplot of number of flights versus delay.

```
delays = not_cancelled %>%
  group_by(tailnum) %>%
  summarize(
    delay = mean(arr_delay, na.rm = T),
    n = n()
  )
ggplot(data = delays, mapping = aes(x=n,y= delay)) +
  geom_point(alpha = 1/10)
```



Filter out groupd with small number of observations.

```
delays %>%
  filter(n > 25) %>%
  ggplot(mapping = aes(x=n,y=delay)) +
  geom_point(alpha = 1/10)
```



Useful Summary Functions

```
# Measures of location
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(
    #average delay
    avg_delay1 = mean(arr_delay),
    #average positive delay
    avg_delay2 = mean(arr_delay[arr_delay > 0])
  )
```

```
## # A tibble: 365 x 5
## # Groups:   year, month [12]
##     year month   day avg_delay1 avg_delay2
##    <int> <int> <int>      <dbl>      <dbl>
##  1  2013     1     1     12.7         32.5
##  2  2013     1     2     12.7         32.0
##  3  2013     1     3      5.73        27.7
##  4  2013     1     4     -1.93        28.3
##  5  2013     1     5     -1.53        22.6
##  6  2013     1     6      4.24        24.4
##  7  2013     1     7     -4.95        27.8
##  8  2013     1     8     -3.23        20.8
##  9  2013     1     9     -0.264       25.6
## 10  2013     1    10     -5.90        27.3
## # ... with 355 more rows
```

Why is the distance to some destinations more variable than to others?

```
# measures of spread
not_cancelled %>%
  group_by(dest) %>%
  summarize(distance_sd = sd(distance)) %>%
  arrange(desc(distance_sd))
```

```
## # A tibble: 104 x 2
##    dest  distance_sd
##    <chr>       <dbl>
##  1 EGE         10.5
##  2 SAN         10.4
##  3 SFO         10.2
##  4 HNL         10.0
##  5 SEA          9.98
##  6 LAS          9.91
##  7 PDX          9.87
##  8 PHX          9.86
##  9 LAX          9.66
## 10 IND          9.46
## # ... with 94 more rows
```

When do first and last flights leave each day?

```
# measures of rank min(x),max(x),quantile(x)
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(
    first = min(dep_time),
    last = max(dep_time)
  )
```

```
## # A tibble: 365 x 5
## # Groups:   year, month [12]
##     year month   day first  last
##    <int> <int> <int> <int> <int>
##  1  2013     1     1   517  2356
##  2  2013     1     2    42  2354
##  3  2013     1     3    32  2349
##  4  2013     1     4    25  2358
##  5  2013     1     5    14  2357
##  6  2013     1     6    16  2355
##  7  2013     1     7    49  2359
##  8  2013     1     8   454  2351
##  9  2013     1     9     2  2252
## 10  2013     1    10     3  2320
## # ... with 355 more rows
```

```
# measures of position first(x), last(x),nth(x,2)
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(
    first = first(dep_time),
    last = last(dep_time)
  )
```

```
## # A tibble: 365 x 5
## # Groups:   year, month [12]
##     year month   day first  last
##    <int> <int> <int> <int> <int>
##  1  2013     1     1   517  2356
##  2  2013     1     2    42  2354
##  3  2013     1     3    32  2349
##  4  2013     1     4    25  2358
##  5  2013     1     5    14  2357
##  6  2013     1     6    16  2355
##  7  2013     1     7    49  2359
##  8  2013     1     8   454  2351
##  9  2013     1     9     2  2252
## 10  2013     1    10     3  2320
## # ... with 355 more rows
```

Which destination have the most carriers?

```
# counts
not_cancelled %>%
  group_by(dest) %>%
  summarize(carriers = n_distinct(carrier)) %>%
  arrange(desc(carriers))
```

```
## # A tibble: 104 x 2
##    dest  carriers
##    <chr>    <int>
##  1 ATL          7
##  2 BOS          7
##  3 CLT          7
##  4 ORD          7
##  5 TPA          7
##  6 AUS          6
##  7 DCA          6
##  8 DTW          6
##  9 IAD          6
## 10 MSP          6
## # ... with 94 more rows
```

Count the total number of miles a plane flew.

```
not_cancelled %>%
  count(tailnum, wt = 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
```

How many flights left before 5am? (these usually indicate delayed flights from previous day)

```
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(n_early = sum(dep_time < 500))
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day n_early
##    <int> <int> <int>   <int>
##  1  2013     1     1       0
##  2  2013     1     2       3
##  3  2013     1     3       4
##  4  2013     1     4       3
##  5  2013     1     5       3
##  6  2013     1     6       2
##  7  2013     1     7       2
##  8  2013     1     8       1
##  9  2013     1     9       3
## 10  2013     1    10       3
## # ... with 355 more rows
```

What proportion of flights are delayed by more than an hour?

```
not_cancelled %>%
  group_by(year, month, day) %>%
  summarize(hour_perc = mean(arr_delay > 60))
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day hour_perc
##    <int> <int> <int>     <dbl>
##  1  2013     1     1    0.0722
##  2  2013     1     2    0.0851
##  3  2013     1     3    0.0567
##  4  2013     1     4    0.0396
##  5  2013     1     5    0.0349
##  6  2013     1     6    0.0470
##  7  2013     1     7    0.0333
##  8  2013     1     8    0.0213
##  9  2013     1     9    0.0202
## 10  2013     1    10    0.0183
## # ... with 355 more rows
```

Grouping by multiple variables

When you group by multiple variables, each summary peels off one level of grouping. This makes it easy to progressively roll up a dataset.

```
daily = group_by(flights, year, month, day)
(per_day = summarize(daily, flights = n()))
```

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

```
(per_month = summarize(per_day, flights = sum(flights)))
```

```
## # A tibble: 12 x 3
## # Groups:   year [1]
##     year month flights
##    <int> <int>   <int>
##  1  2013     1   27004
##  2  2013     2   24951
##  3  2013     3   28834
##  4  2013     4   28330
##  5  2013     5   28796
##  6  2013     6   28243
##  7  2013     7   29425
##  8  2013     8   29327
##  9  2013     9   27574
## 10  2013    10   28889
## 11  2013    11   27268
## 12  2013    12   28135
```

```
(per_year = summarize(per_month, flights = sum(flights)))
```

```
## # A tibble: 1 x 2
##    year flights
##   <int>   <int>
## 1  2013  336776
```

Grouped Mutates (and Filters) Find the worst members of each group

```
flights_sml = select(flights,
                     year:day,
                     ends_with("delay"),
                     distance,
                     air_time)
flights_sml %>%
  group_by(year,month,day) %>%
  filter(rank(desc(arr_delay))< 10)
```

```
## # A tibble: 3,306 x 7
## # Groups:   year, month, day [365]
##     year month   day dep_delay arr_delay distance air_time
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl>
##  1  2013     1     1       853       851      184       41
##  2  2013     1     1       290       338     1134      213
##  3  2013     1     1       260       263      266       46
##  4  2013     1     1       157       174      213       60
##  5  2013     1     1      216      222      708      121
##  6  2013     1     1      255      250      589      115
##  7  2013     1     1      285      246     1085      146
##  8  2013     1     1      192      191      199       44
##  9  2013     1     1      379      456     1092      222
## 10  2013     1     2      224      207      550       94
## # ... with 3,296 more rows
```

Find all groups bigger than a threshold

```
popular_dest = flights %>%
  group_by(dest) %>%
  filter(n() > 365)
popular_dest
```

```
## # A tibble: 332,577 x 19
## # Groups:   dest [77]
##     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 332,567 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>
```

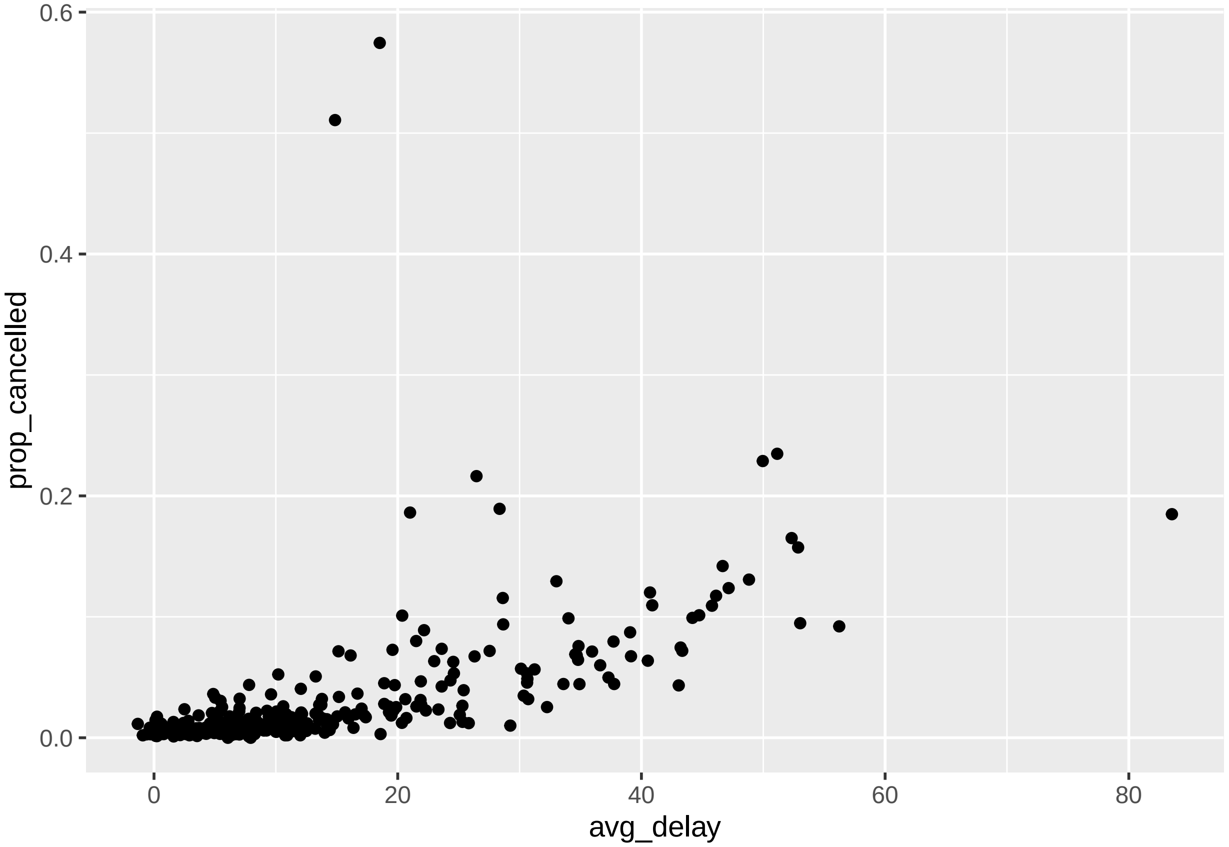
Standardize to compute per group metrics

```
popular_dest %>%
  filter(arr_delay > 0) %>%
  mutate(prop_delay = arr_delay / sum(arr_delay)) %>%
  select(year:day, dest, arr_delay, prop_delay)
```

```
## # A tibble: 131,106 x 6
## # Groups:   dest [77]
##     year month   day dest  arr_delay prop_delay
##    <int> <int> <int> <chr>     <dbl>      <dbl>
##  1  2013     1     1 IAH          11  0.000111
##  2  2013     1     1 IAH          20  0.000201
##  3  2013     1     1 MIA          33  0.000235
##  4  2013     1     1 ORD          12  0.0000424
##  5  2013     1     1 FLL          19  0.0000938
##  6  2013     1     1 ORD           8  0.0000283
##  7  2013     1     1 LAX           7  0.0000344
##  8  2013     1     1 DFW          31  0.000282
##  9  2013     1     1 ATL          12  0.0000400
## 10  2013     1     1 DTW          16  0.000116
## # ... with 131,096 more rows
```

```
flights_new = flights %>%
  group_by(year, month, day) %>%
  summarize(avg_delay = mean(dep_delay,na.rm = T),
            cancelled = sum(is.na(arr_delay) | is.na(dep_delay)),
            flightsTotal = n(),
            prop_cancelled = sum(is.na(arr_delay) | is.na(dep_delay))/n())

ggplot(data = flights_new) +
  geom_point(mapping = aes(x= avg_delay, y = prop_cancelled))
```


prop_cancelled
avg_delay