Miniproject in Introduction to Data Science
ITVEST Data Science and Big Data (DSBD)

```
library(tidyverse)
library(lubridate)
library(pander) # for prettier tables
library(scales) # for making prettier axes in plots
library(stringr)
library(sqldf)

theme_set(theme_bw())

panderOptions('big.mark', ',')
```

1 Formalia

Deadline for hand-in: Jan 3, 2018 at 23:55.

Where: Moodle.

What: Rmd file. Possibly also pdf (or html), but Rmd is mandatory.

Groups: Maximum 3 participants, however the project must be handed in individually.

2 Exercises

Here, we focus on the `airlines`, `airports`, `flights`, `planes`, and `weather` datasets:

```
library(nycflights13)
```

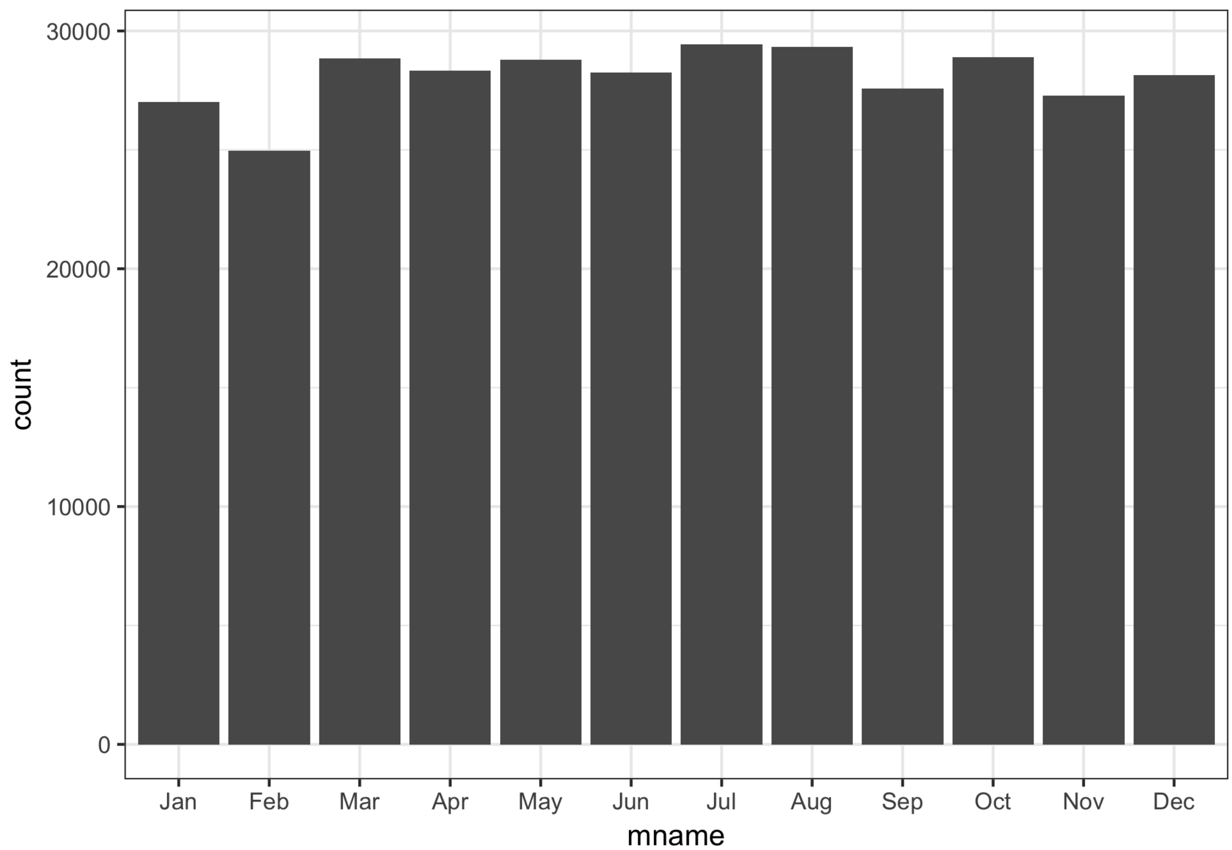
Remember to read about the datasets.

3 Exercises

3.1 Exercise

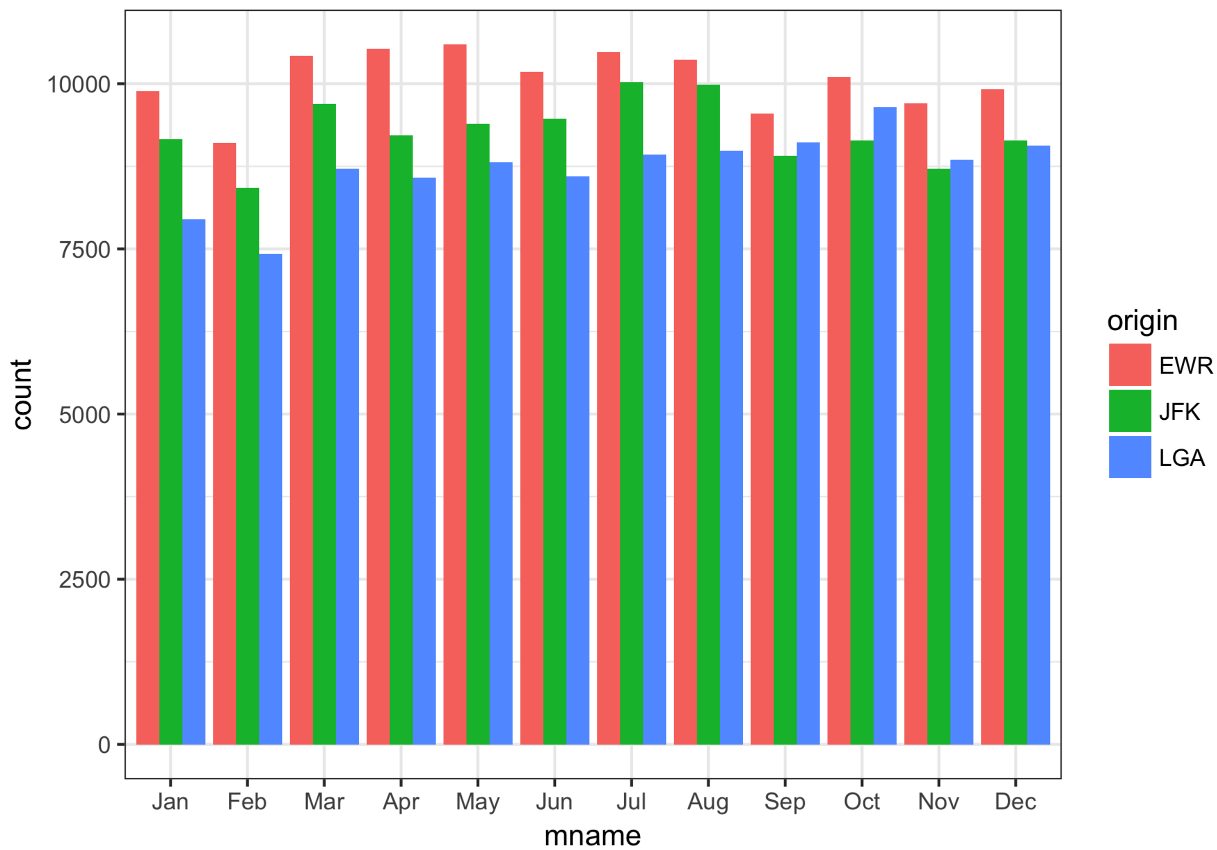
Construct a barplot displaying number of flights per month.
Basic barplot - lets add monthnames as a ordered factor, to have maeningsful labels on the x-axis and ordering the months, according to the calendar

```
library(ggplot2)
flightsWithMontName <- mutate(flights, mname = month.abb[month])
flightsWithMontName$mname <-
  factor(flightsWithMontName$mname,
         levels = c("Jan", "Feb", "Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec" ))

p<-ggplot(data=flightsWithMontName, aes(x=mname)) +
  geom_bar()
p
```



Now, in the barplot (showing number of flights per month), make a separate bar for each origin.

```
p2<-ggplot(data=flightsWithMontName, aes(fill=origin,x=mname)) +
  geom_bar(position="dodge")
p2
```



3.2 Exercise

What are the top-10 destinations and how many flights were made to these?

There is a sqldf package, that allows SQL code to be run against dataframes and tibbles. Generally, throughout this miniproject, I will utillize both SQL and R syntax in selecting, joining etc.

Trying out this SQL technique

```
Toplist <-
  sqldf('select dest, count(*) as numberOfFlights
          from flights
         group by dest
        order by numberOfFlights desc')
```

For illustration purposes, below is shown another technique.
Please note, first use of pipe operator (from magrittr, which is included in tidyverse).

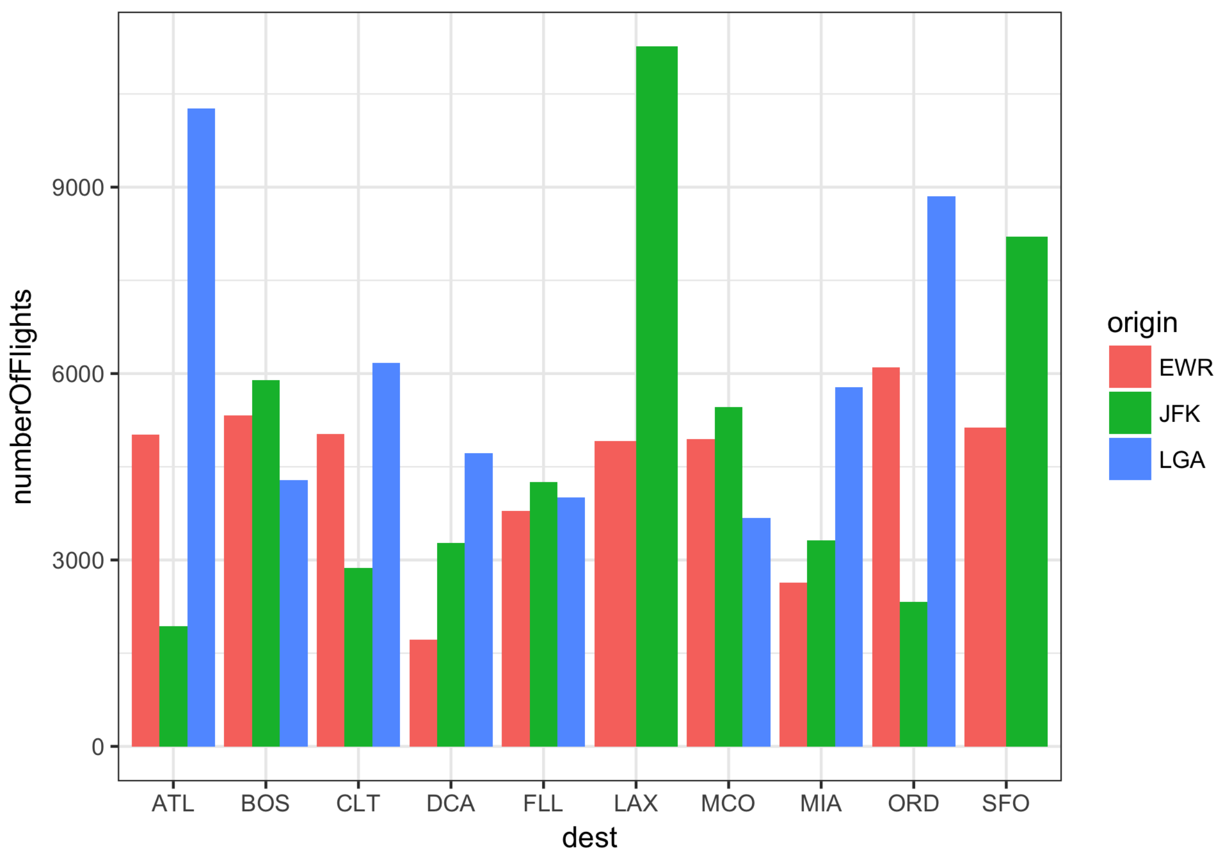
```
Toplist <- flights %>% count(dest) %>%  arrange(desc(n))
Toplist10 <- Toplist %>% top_n(10)
names(Toplist10)[2] = "numberOfFlights"
# Pretty print using pander
pander(Toplist10, big.mark=',', justify = c('left','right'))
```

dest	numberOfFlights
ORD	17,283
ATL	17,215

dest	numberOfFlights
LAX	16,174
BOS	15,508
MCO	14,082
CLT	14,064
SFO	13,331
FLL	12,055
MIA	11,728
DCA	9,705

For these 10 destinations, make a barplot illustrating the number of flights from origin to top-10 destination.

```
Toplist10DestOrigin <-
  sqldf('select dest,
                origin,
                count(*) as numberOfFlights
           from flightsWithMontName
          where dest in (select dest from Toplist10)
          group by dest, origin
          order by numberOfFlights desc');
p4<-ggplot(data=Toplist10DestOrigin, aes(x=dest, y=numberOfFlights, fill=origin)) +
  geom_bar(stat="identity", position="dodge")
p4
```

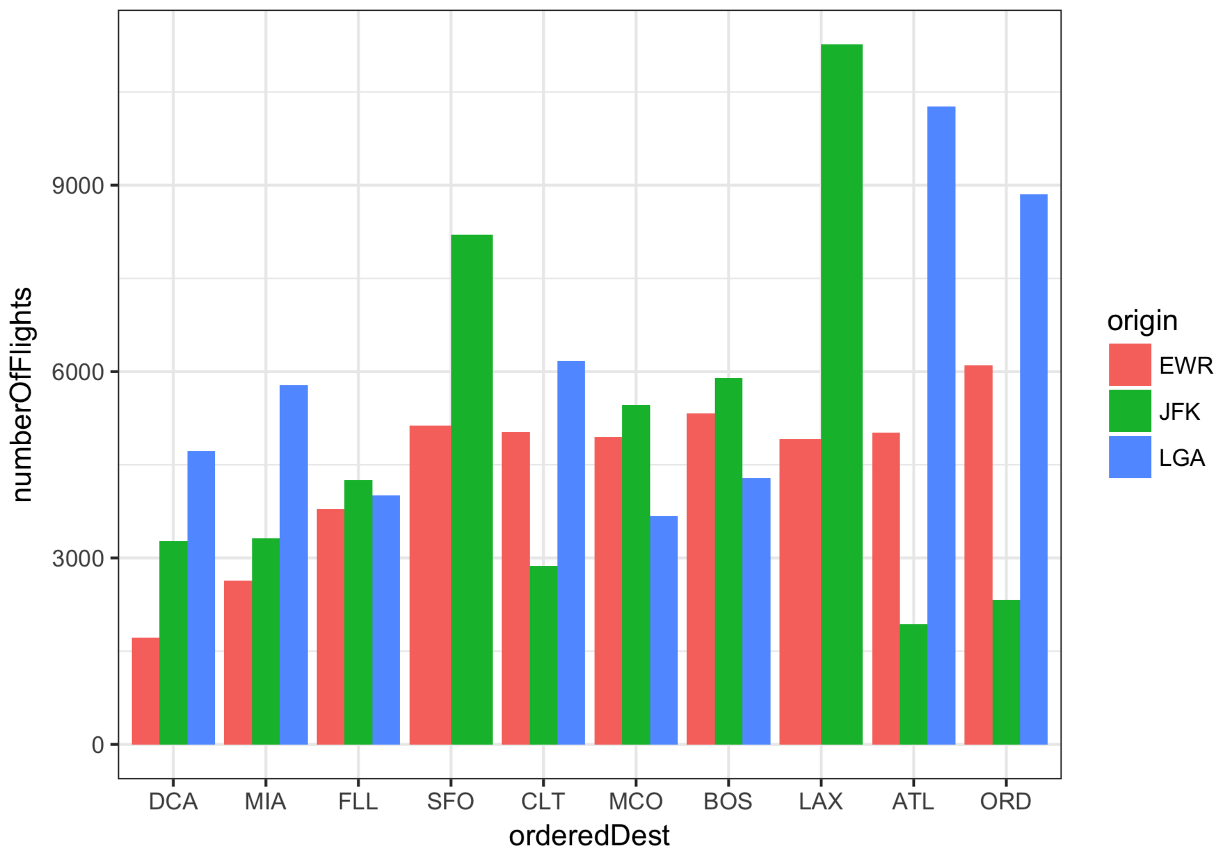


Now, order the bars (destinations) according to the total number of flights to these destina-

tions.

This is done by creating a new Destination variable as a ordered factor, with levels ordered by the number of flight to these destinations.

```
library(dplyr)
# Create a new ordered factor (vector)
orderedDestinationFactor <-
  dplyr::arrange(Toplist10,numberOfFlights) %>%
  select(.,dest) %>%
  unlist(.) %>%
  ordered(.)
# Explanation of the above PIPE:
# arrange step: sort TopList10 by numberOfFligths
# select      : isolate destination variable
# unlist      : select returns list, must be vector
# ordered     : creates ordered factor

# Create new column - orderedDest - from the new factor (vector)
Toplist10DestOrigin$orderedDest <- factor(Toplist10DestOrigin$dest,
                                          levels = orderedDestinationFactor)

# Printing the plot again, now ordering the facet_grid by the new o
# rderedDest variable (ascending)


p5<-ggplot(data=Toplist10DestOrigin, aes(x=orderedDest, y=numberOfFlights, fill=origin)) +
  geom_bar(stat="identity", position="dodge")
p5
```

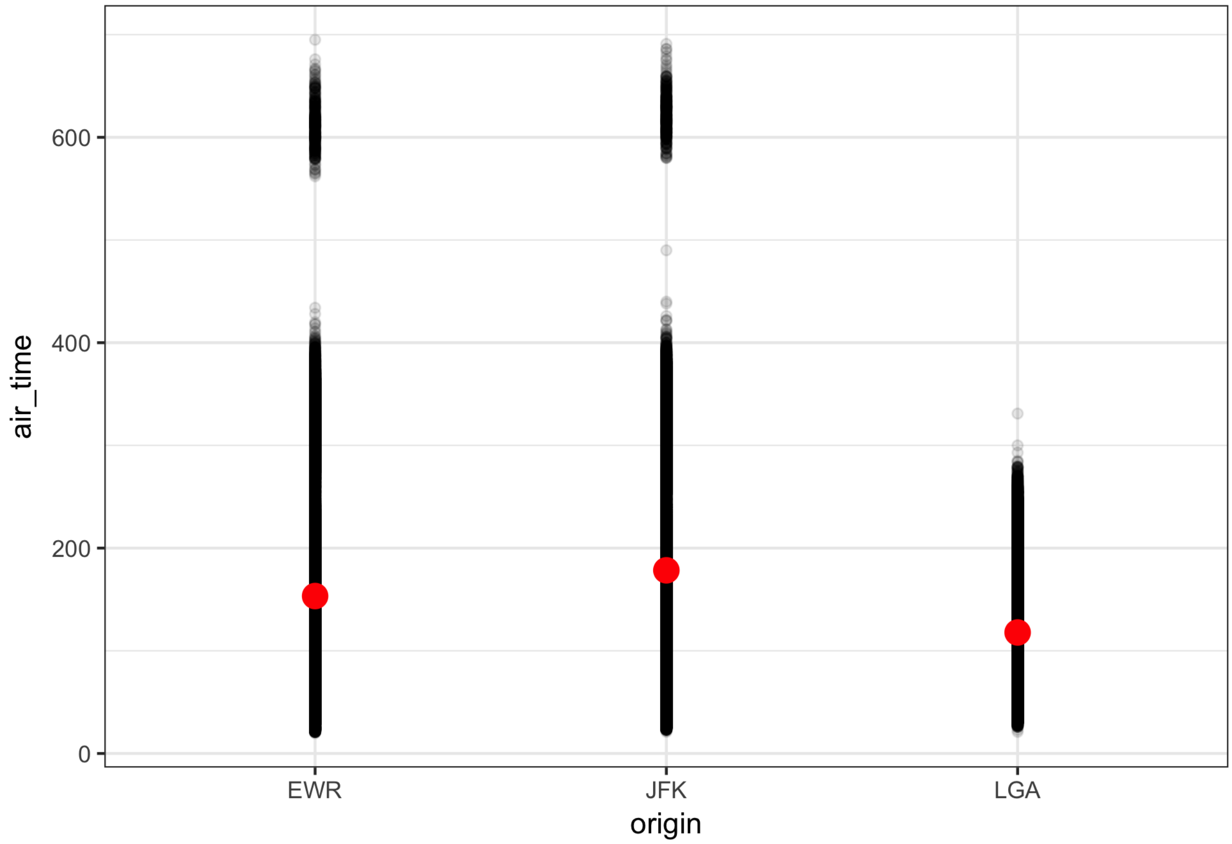


3.3 Exercise

Is the mean air time from each of the three origins different? Further, are these differences statistically significant?

Lets plot the data, to have a first look at the means and individual observations, to get a feel for data
We will create a dataframe that contains the means and plot them together with the individual observations

```
# Please note, that for illustration purposes, two different techniques
# for removing observations with NA values in the relevant column have been used
myGroupedMeans <- group_by(flights, origin) %>%
  summarise(
    air_time = mean(air_time, na.rm = TRUE)
  )

flights %>% filter(!is.na(origin)&!is.na(air_time)) %>%
ggplot(., aes(x = origin, y = air_time)) +
  geom_point(alpha = .1, na.rm=TRUE) +
  geom_point(data=myGroupedMeans, size=4, color="red")
```



In this plot, air_time looks like the means are different but not by a lot, but lets take a closer look:

```
pander(myGroupedMeans, big.mark=',', justify = c('left','right'))
```

origin	air_time
EWR	153.3
JFK	178.3
LGA	117.8

Lets statistically test the following hypotheses:
* NULL Hypothosis: the means aitime are all the same
* Alternative hypothesis: atleast one mean airtime is different from the other means

Let us fit two models:
* a very simple model, predicting air_time from no variables (returns the mean) and
* a model. predicting airtime from the origin variable

ANOVA takes two fitted models and computes analysis of variance

```
model1 = lm(air_time ~ 1, data = flights)
model2 = lm(air_time ~ origin, data = flights)
anova(model1, model2)
```

```
## Analysis of Variance Table
##
## Model 1: air_time ~ 1
## Model 2: air_time ~ origin
```

```
##   Res.Df        RSS Df Sum of Sq     F    Pr(>F)
## 1 327345 2873270224
## 2 327343 2679787595  2 193482628 11817 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(model2)
```

```
##
## Call:
## lm(formula = air_time ~ origin, data = flights)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -157.35  -62.35  -14.30   42.70  541.70
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 153.3000     0.2644  579.86   <2e-16 ***
## originJFK    25.0490     0.3807   65.79   <2e-16 ***
## originLGA   -35.4742     0.3884  -91.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 90.48 on 327343 degrees of freedom
##   (9430 observations deleted due to missingness)
## Multiple R-squared:  0.06734,    Adjusted R-squared:  0.06733
## F-statistic: 1.182e+04 on 2 and 327343 DF,  p-value: < 2.2e-16
```

Well, looking at the P-values, which is VERY small (2.2e-16<0.05), it looks like we shold reject our NULL hypothesis that the means are equal.
This means, that there is actually a significant difference between the means of airtime.
Further analysis using Tukey Method could reviel the actual differences

3.4 Exercise

How many weather observations are there for each origin?

Using SQL syntax and pander for pretty printing

```r
sqldf('select origin,
              count(*) as numberOfObservations
        from weather
       group by origin') %>%
pander(., big.mark=',', justify = c('left','right'))
```

origin	numberOfObservations
EWR	8,708
JFK	8,711
LGA	8,711

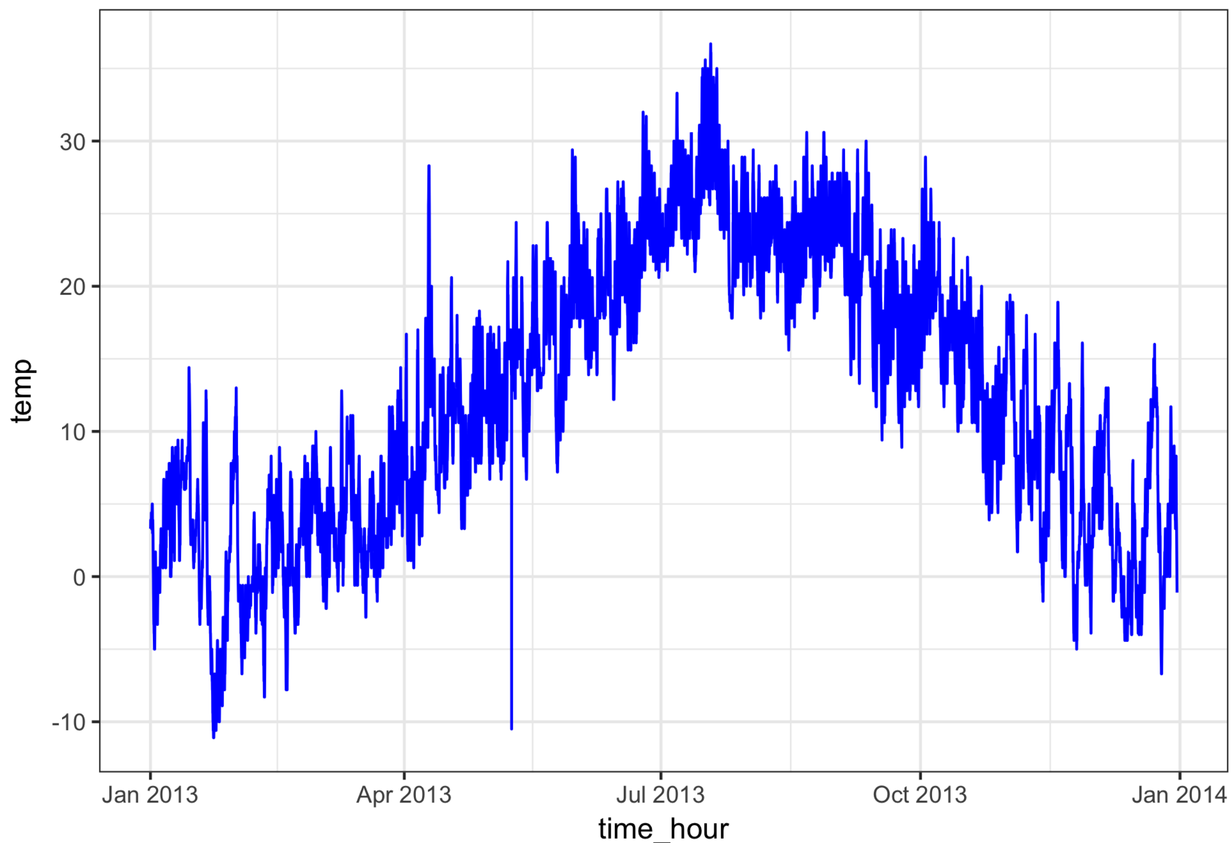
Convert temperature to degrees Celsius. This will be used in the reminder of this miniproject. (We do this for both `temp` and `dewp` using `mutate_at`)

Importing weathermetrics package, that has conversion functions btw celcius and fahrenheit

```
library(weathermetrics)
celciusWeather <- weather %>% mutate_at(vars(temp,dewp),funs(fahrenheit.to.celsius))
```

Construct a graph displaying the temperature at `JFK`.

```
p6 <- filter(celciusWeather, origin == "JFK") %>% ggplot(., aes(time_hour, temp)) +
  geom_line(color="blue")
p6
```

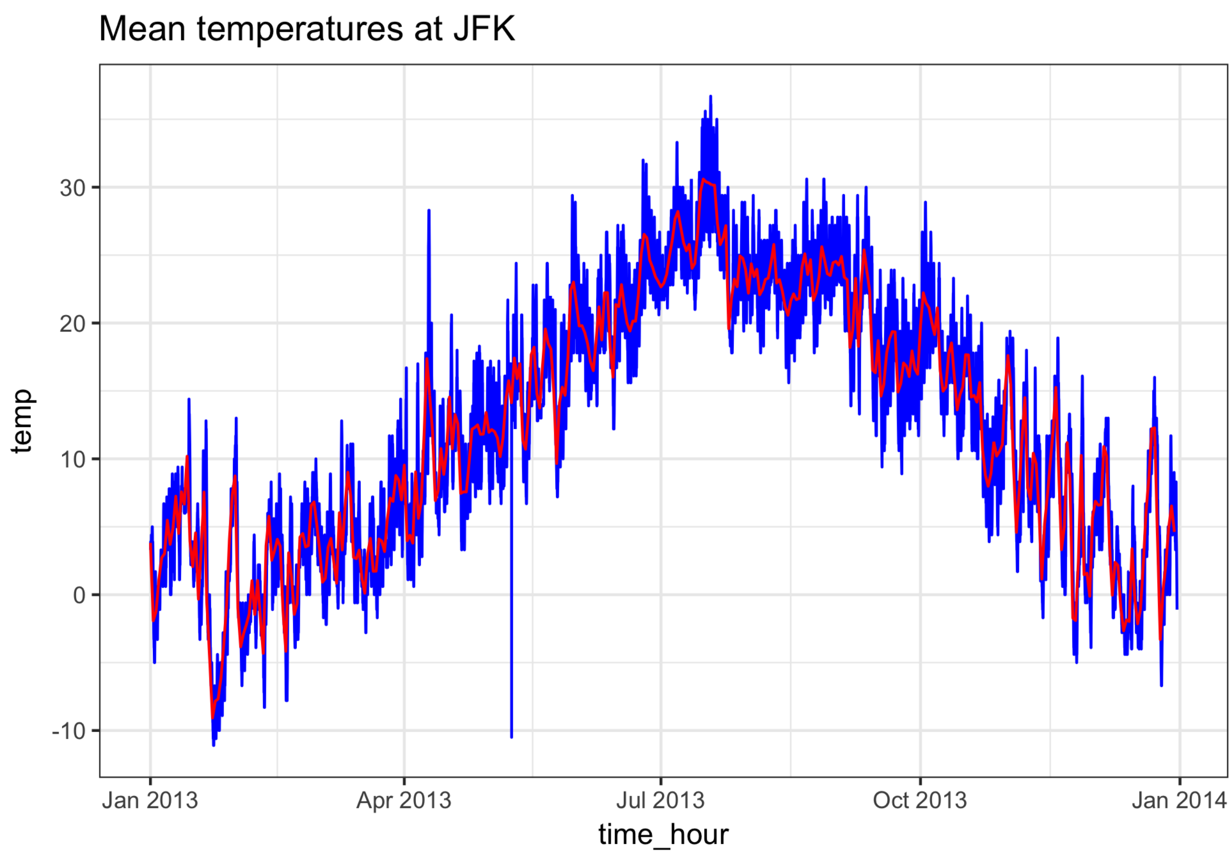


Add a red line showing the mean temperature for each day.

First, lets filter the data to JFK.
Secondly, truncate the timestamp to "day".
Thirdly, lets calculate the mean for each day.

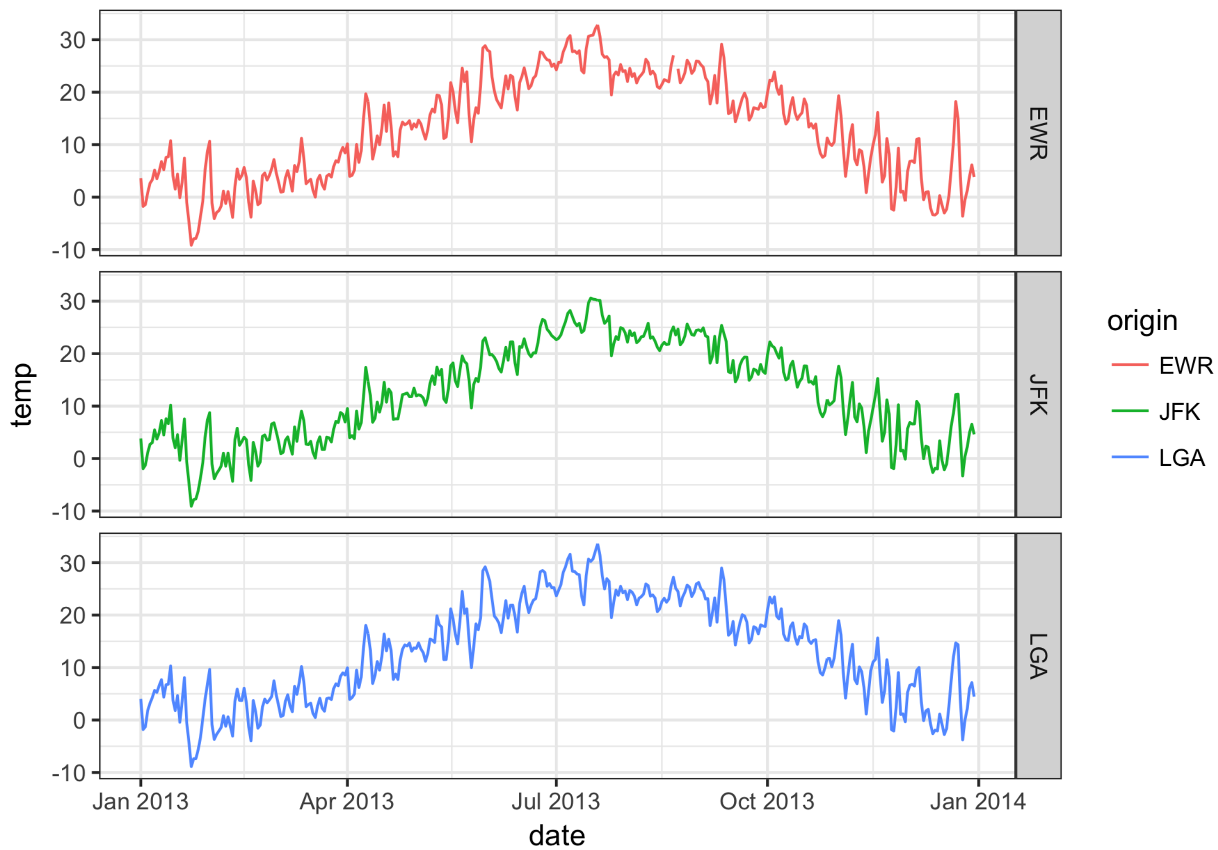
Now, add a line to the previous lot, describing the mean for each day.

```
JFKtemps <- filter(celciusWeather, origin == "JFK")
JFKtemps$date <- floor_date(JFKtemps$time_hour,"day")
JFKmeanTemps <- JFKtemps %>% group_by(date) %>% summarise_each(funs(mean(.)),temp)

p6 + geom_line(data=JFKmeanTemps,aes(date, temp), color="red") +
  ggtitle("Mean temperatures at JFK")
```



Now, visualuse the daily mean temperature for each origin airport.

```
meanTemps <- celciusWeather
## Create a truncated date variable
meanTemps$date <- floor_date(meanTemps$time_hour,"day")
meanTemps <- meanTemps %>% group_by(date, origin) %>% summarise_each(funs(mean(.)),temp)

p7 <- ggplot(meanTemps, aes(date, temp, group = origin, color=origin)) +
  geom_line(na.rm=TRUE) + facet_grid(origin ~ .)
p7
```

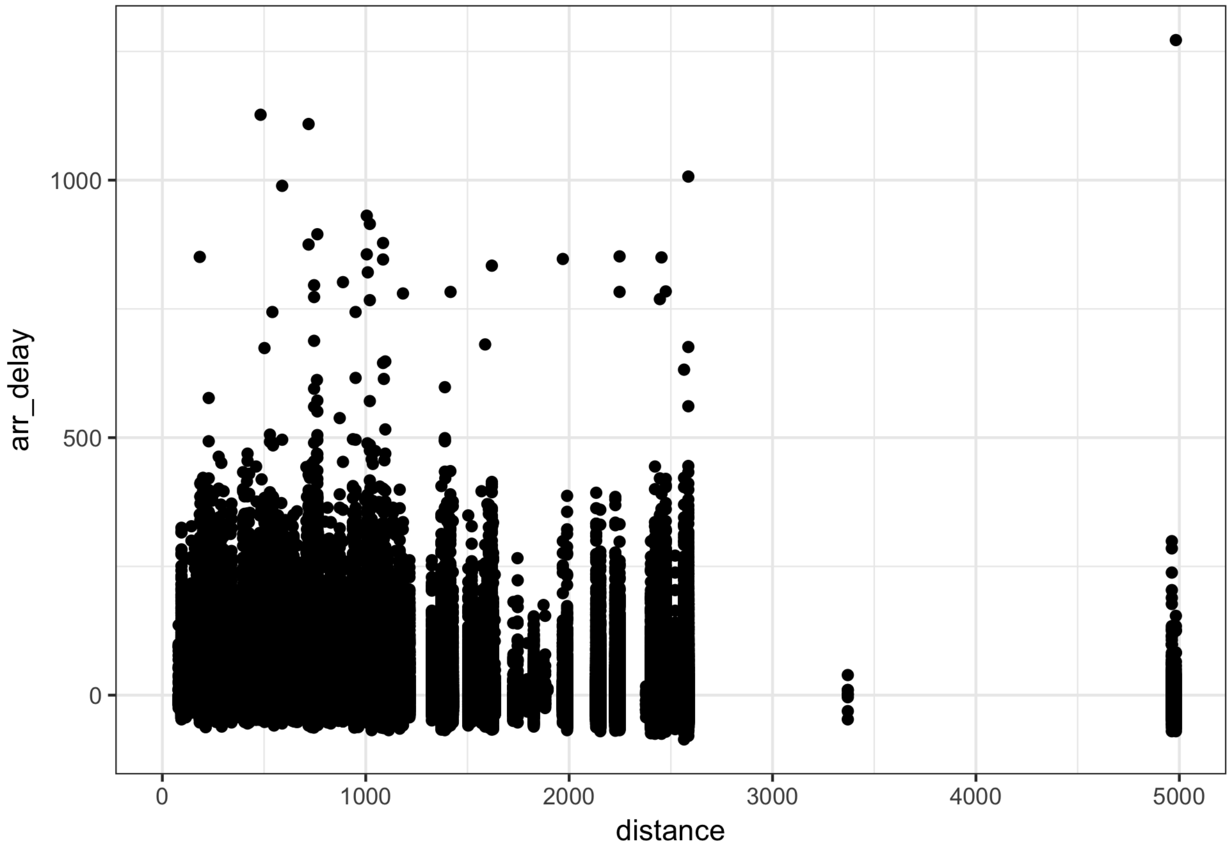


3.5 Exercise

Investigate if arrival delay is associated with the flight distance (and also departure delay).

All good data analysis starts with a visualization

```
ggplot(flights, aes(x=distance, y=arr_delay)) + geom_point(na.rm=TRUE)
```



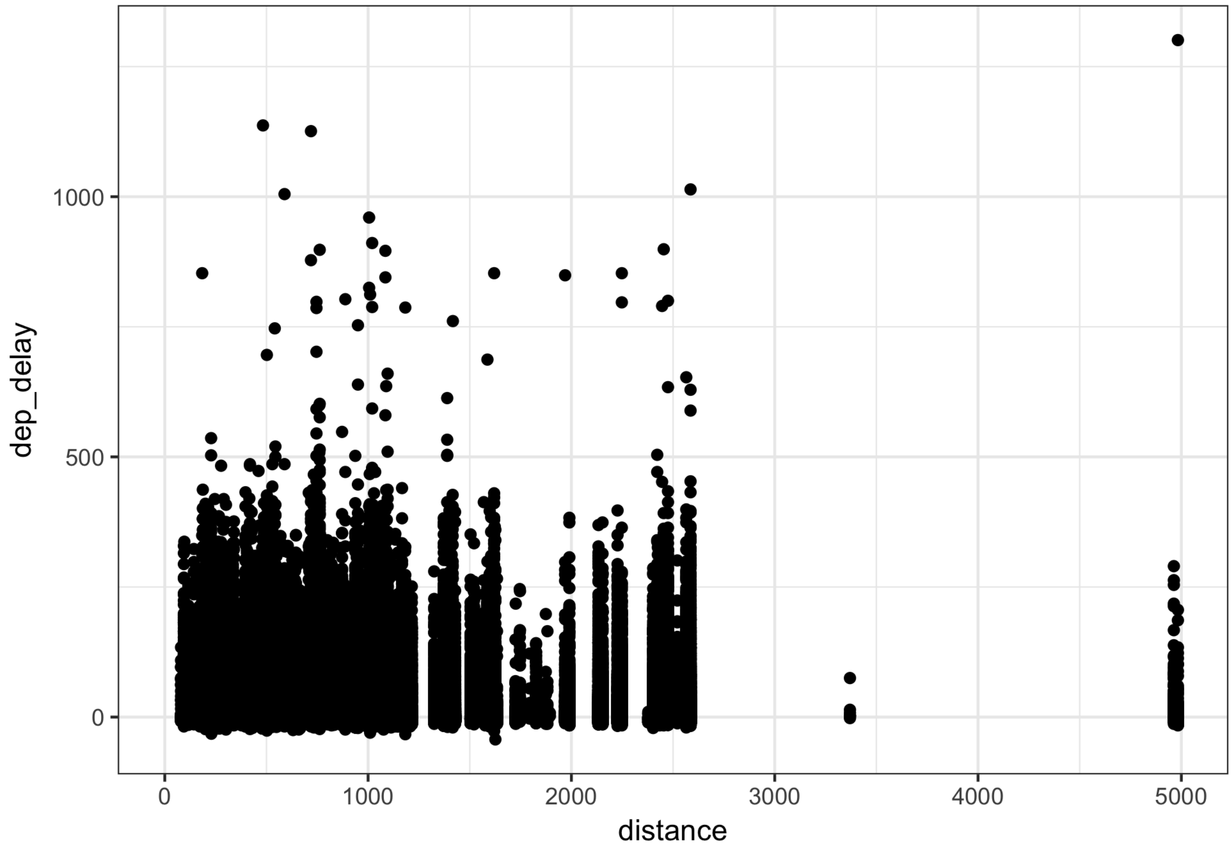
It does not visually seen like delay is associated with flight distance.
But, to be more scientific about it, lets do a correlation test (Pearsons):
This test fits a linear model and returns a measure of how good the points fit the line.
This measure is 0 for no correlation, and -1 or 1 for complete correlation (positive or negative slope of line).

```
cor.test(flights$arr_delay, flights$distance)
```

```
##
##  Pearson's product-moment correlation
##
## data:  flights$arr_delay and flights$distance
## t = -35.465, df = 327340, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.06527959 -0.05845448
## sample estimates:
##         cor
## -0.06186776
```

This shows very weak correlation, as expected.
The correlation coefficiant is almost 0 (-0.06186776), which indicate no relationship.
Lets try departure_delay:

```
ggplot(flights, aes(x=distance, y=dep_delay)) + geom_point(na.rm=TRUE)
```




```
cor.test(flights$distance, flights$dep_delay)
```

```
##
##  Pearson's product-moment correlation
##
## data:  flights$distance and flights$dep_delay
## t = -12.424, df = 328520, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.02508846 -0.01825261
## sample estimates:
##         cor
## -0.02167079
```

Nope, even less correlation !
-0.02167079

3.6 Exercise

Investigate if departure delay is associated with weather conditions at the origin airport. This includes descriptives (mean departure delay), plotting, regression modelling, considering missing values etc.

Lets first invastigate the mean departure delays across weather conditions.
Calculating means requires, that we first investigave missing values. Calculating MEAN in NA will return NA.

```r
summary(flights$dep_delay)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  -43.00   -5.00   -2.00   12.64   11.00 1301.00    8255
```

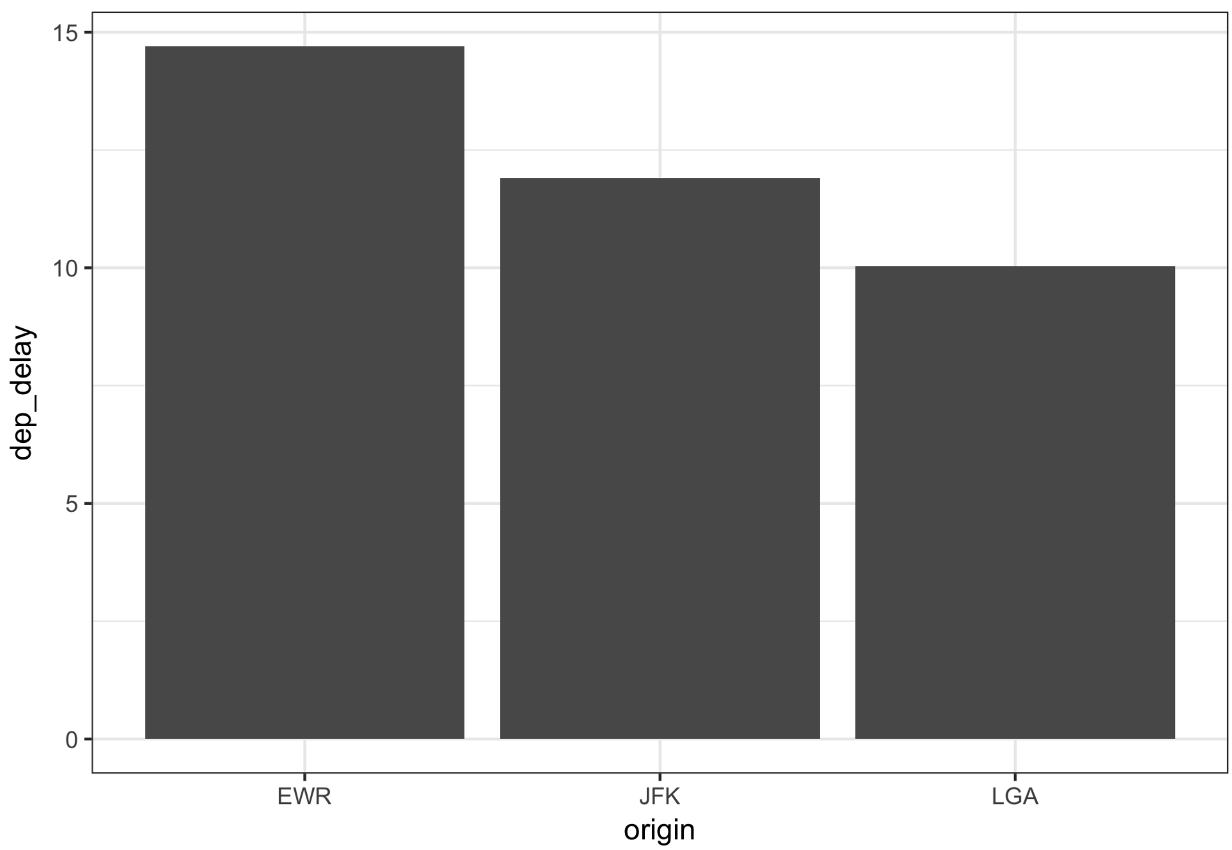
Looks like there are 8255 missing.
Lets consider this: missing departure delays must mean, that there is no delay, eg. the flight left on time. This is my own interpretation, as I have no contact with domain knowledge to ask and the documentation does not contain an answer. Thus, lets set the dep_delay to 0 if it is missing, before calculating means. Since we will be using this filter again and again, lets persist it in a new dataset, although it could also generally be handled with na.action functions.

```r
myFlights <- flights %>% mutate(dep_delay = ifelse(is.na(dep_delay), 0, dep_delay))
aggregate(myFlights$dep_delay, by=list(flights$origin), FUN=mean) %>%
  pander(., big.mark=',', justify = c('left','right'))
```

Group.1	x
EWR	14.7
JFK	11.91
LGA	10.04

Looks like there is some difference between the means.
Lets take a closer look, using a barplot.

```r
ggplot(data = myFlights, mapping = aes(x = origin, y=dep_delay)) +
  geom_bar(stat='summary')
```



So, lets see, if these delays are associated with weather conditions.
Here, weather condisions are described by the variables in the weather dataset, so we first need to join flight data and weather data, eg. answering then question "How was the weather, when the flight was supposed to leave".

Weatherinformation are point-in-time information and scheduled departure time is aswell. However, the time_hour column in flights dataset describes which weather measurement describes the departure, so no need to mutate and doing a between join.

Turning to SQL syntax to do the join - deliberately using a left join to see, if anything is missing in the weather data:

```
myFlightWeather <- sqldf('select a.origin, a.dep_delay, a.time_hour as flightsTime,
                                 b.time_hour as weatherTime, b.temp, b.dewp, b.humid,
                                 b.wind_dir,b.wind_speed,wind_gust, b.precip, b.pressure,
                                 b.visib
                            from myFlights a left outer join
                                 weather b on a.origin = b.origin
                             and a.time_hour = b.time_hour')
summary(myFlightWeather)
```

```
##     origin            dep_delay        flightsTime
##  Length:336776      Min.   : -43.00   Min.   :1.357e+09
##  Class :character   1st Qu.:  -5.00   1st Qu.:1.365e+09
##  Mode  :character   Median :  -1.00   Median :1.373e+09
##                     Mean   :  12.33   Mean   :1.373e+09
##                     3rd Qu.:  10.00   3rd Qu.:1.381e+09
##                     Max.   :1301.00   Max.   :1.389e+09
##
##   weatherTime             temp             dewp           humid
##  Min.   :1.357e+09   Min.   : 10.94   Min.   :-9.94   Min.   : 12.74
##  1st Qu.:1.365e+09   1st Qu.: 39.92   1st Qu.:26.60   1st Qu.: 45.79
##  Median :1.373e+09   Median : 55.94   Median :42.80   Median : 60.54
##  Mean   :1.373e+09   Mean   : 55.94   Mean   :41.57   Mean   : 61.41
##  3rd Qu.:1.381e+09   3rd Qu.: 71.06   3rd Qu.:57.92   3rd Qu.: 77.92
##  Max.   :1.388e+09   Max.   :100.04   Max.   :78.08   Max.   :100.00
##  NA's   :1199        NA's   :1222     NA's   :1222    NA's   :1222
##     wind_dir       wind_speed         wind_gust            precip
##  Min.   :  0.0   Min.   :   0.000   Min.   :   0.000   Min.   :0.0000
##  1st Qu.:100.0   1st Qu.:   6.905   1st Qu.:   7.946   1st Qu.:0.0000
##  Median :220.0   Median :  10.357   Median :  11.919   Median :0.0000
##  Mean   :196.8   Mean   :  10.625   Mean   :  12.227   Mean   :0.0028
##  3rd Qu.:290.0   3rd Qu.:  13.809   3rd Qu.:  15.892   3rd Qu.:0.0000
##  Max.   :360.0   Max.   :1048.361   Max.   :1206.432   Max.   :1.1800
##  NA's   :8170    NA's   :1248       NA's   :1248       NA's   :1199
##     pressure          visib
##  Min.   : 983.8   Min.   : 0.000
##  1st Qu.:1012.7   1st Qu.:10.000
##  Median :1017.5   Median :10.000
##  Mean   :1017.9   Mean   : 9.172
##  3rd Qu.:1022.9   3rd Qu.:10.000
##  Max.   :1042.1   Max.   :10.000
##  NA's   :37618    NA's   :1199
```

By looking at weatherTime column, it looks like, there are 1199 flights that cannot be paired with weather data because of missing weather data.

Lets see how many gaps there are in the weather data - it is supposed to have one measurement every hour

```
myExtendedWeather <- mutate(weather, validFrom = time_hour,
                            validTo = lead(time_hour)-1, gap=(lead(time_hour)-time_hour))
sqldf('select gap||" - hours" as gaps,
              count(*) as antal
        from myExtendedWeather
        group by gap') %>% pander(., big.mark=',', justify = c('left','right'))
```

gaps	antal
NA	1
-8735.0 - hours	2
1.0 - hours	26,082
2.0 - hours	30
3.0 - hours	9
6.0 - hours	6

Looks like, if we accept up to 2 hours old weatherdata, we can include 30 more flights, but since we do not know wether this is acceptable, we wont. So, we will turn to inner-joining instead.

```
myFlightWeather <- sqldf('select a.origin, a.dep_delay, a.time_hour as flightsTime,
                               b.time_hour as weatherTime, b.temp, b.dewp, b.humid,
                               b.wind_dir, b.wind_speed, wind_gust, b.precip, b.pressure,
                               b.visib
                          from myFlights a inner join
                               myExtendedWeather b on a.origin = b.origin
                           and a.time_hour = b.time_hour')
summary(myFlightWeather)
```

```
##     origin            dep_delay        flightsTime
##   Length:335577      Min.   : -43.00   Min.   :1.357e+09
##   Class :character   1st Qu.:  -5.00   1st Qu.:1.365e+09
##   Mode  :character   Median :  -1.00   Median :1.373e+09
##                      Mean   :  12.34   Mean   :1.373e+09
##                      3rd Qu.:  10.00   3rd Qu.:1.381e+09
##                      Max.   :1301.00   Max.   :1.388e+09
##
##    weatherTime             temp             dewp          humid
##   Min.   :1.357e+09   Min.   : 10.94   Min.   :-9.94   Min.   : 12.74
##   1st Qu.:1.365e+09   1st Qu.: 39.92   1st Qu.:26.60   1st Qu.: 45.79
##   Median :1.373e+09   Median : 55.94   Median :42.80   Median : 60.54
##   Mean   :1.373e+09   Mean   : 55.94   Mean   :41.57   Mean   : 61.41
##   3rd Qu.:1.381e+09   3rd Qu.: 71.06   3rd Qu.:57.92   3rd Qu.: 77.92
##   Max.   :1.388e+09   Max.   :100.04   Max.   :78.08   Max.   :100.00
##                       NA's   :23       NA's   :23      NA's   :23
##     wind_dir       wind_speed         wind_gust            precip
##   Min.   :  0.0   Min.   :   0.000   Min.   :   0.000   Min.   :0.000000
##   1st Qu.:100.0   1st Qu.:   6.905   1st Qu.:   7.946   1st Qu.:0.000000
##   Median :220.0   Median :  10.357   Median :  11.919   Median :0.000000
##   Mean   :196.8   Mean   :  10.625   Mean   :  12.227   Mean   :0.002813
##   3rd Qu.:290.0   3rd Qu.:  13.809   3rd Qu.:  15.892   3rd Qu.:0.000000
##   Max.   :360.0   Max.   :1048.361   Max.   :1206.432   Max.   :1.180000
##   NA's   :6971    NA's   :49         NA's   :49
##     pressure          visib
```

```
##  Min.   : 983.8   Min.   : 0.000
##  1st Qu.:1012.7   1st Qu.:10.000
##  Median :1017.5   Median :10.000
##  Mean   :1017.9   Mean   : 9.172
##  3rd Qu.:1022.9   3rd Qu.:10.000
##  Max.   :1042.1   Max.   :10.000
##  NA's   :36419
```

So, now that we paired departure delays with weatherdata, lets take a look at the data:
We discover, that we still have columns with missing data in the dataset. For the rest of the columns, lets replace the missing values with the mean on the actual existing observations by creating a vector referencing the rows with missing values for temperature and using it to isolate these rows:

```r
myFlightWeather$temp[which(is.na(myFlightWeather$temp))] <-
  mean(myFlightWeather$temp, na.rm=TRUE)
myFlightWeather$dewp[which(is.na(myFlightWeather$dewp))] <-
  mean(myFlightWeather$dewp, na.rm=TRUE)
myFlightWeather$humid[which(is.na(myFlightWeather$humid ))] <-
  mean(myFlightWeather$humid, na.rm=TRUE)
myFlightWeather$wind_dir[which(is.na(myFlightWeather$wind_dir))] <-
  mean(myFlightWeather$wind_dir, na.rm=TRUE)
myFlightWeather$wind_speed[which(is.na(myFlightWeather$wind_speed))] <-
  mean(myFlightWeather$wind_speed, na.rm=TRUE)
myFlightWeather$wind_gust[which(is.na(myFlightWeather$wind_gust))] <-
  mean(myFlightWeather$wind_gust, na.rm=TRUE)
myFlightWeather$pressure[which(is.na(myFlightWeather$pressure))] <-
  mean(myFlightWeather$pressure, na.rm=TRUE)
summary(myFlightWeather)
```

```
##     origin            dep_delay        flightsTime
##  Length:335577      Min.   : -43.00   Min.   :1.357e+09
##  Class :character   1st Qu.:  -5.00   1st Qu.:1.365e+09
##  Mode  :character   Median :  -1.00   Median :1.373e+09
##                     Mean   :  12.34   Mean   :1.373e+09
##                     3rd Qu.:  10.00   3rd Qu.:1.381e+09
##                     Max.   :1301.00   Max.   :1.388e+09
##   weatherTime              temp             dewp           humid
##  Min.   :1.357e+09   Min.   : 10.94   Min.   :-9.94   Min.   : 12.74
##  1st Qu.:1.365e+09   1st Qu.: 39.92   1st Qu.:26.60   1st Qu.: 45.79
##  Median :1.373e+09   Median : 55.94   Median :42.80   Median : 60.54
##  Mean   :1.373e+09   Mean   : 55.94   Mean   :41.57   Mean   : 61.41
##  3rd Qu.:1.381e+09   3rd Qu.: 71.06   3rd Qu.:57.92   3rd Qu.: 77.92
##  Max.   :1.388e+09   Max.   :100.04   Max.   :78.08   Max.   :100.00
##     wind_dir       wind_speed         wind_gust            precip
##  Min.   :  0.0   Min.   :   0.000   Min.   :   0.000   Min.   :0.000000
##  1st Qu.:110.0   1st Qu.:   6.905   1st Qu.:   7.946   1st Qu.:0.000000
##  Median :220.0   Median :  10.357   Median :  11.919   Median :0.000000
##  Mean   :196.8   Mean   :  10.625   Mean   :  12.227   Mean   :0.002813
##  3rd Qu.:290.0   3rd Qu.:  13.809   3rd Qu.:  15.892   3rd Qu.:0.000000
##  Max.   :360.0   Max.   :1048.361   Max.   :1206.432   Max.   :1.180000
##     pressure          visib
##  Min.   : 983.8   Min.   : 0.000
##  1st Qu.:1013.5   1st Qu.:10.000
##  Median :1017.8   Median :10.000
##  Mean   :1017.8   Mean   : 9.172
```

```
##   3rd Qu.:1022.1   3rd Qu.:10.000
##   Max.   :1042.1   Max.   :10.000
```

We now have a "clean" dataset to work on, so lets see if departure deplay is associated with weather conditions: Fitting a linear model, trying to predict the dep_delay from weather variables.

```
myModel <- lm(dep_delay ~ temp+dewp+humid+wind_dir+wind_speed+wind_gust+precip+pressure+
                visib,myFlightWeather)
summary(myModel)
```

```
##
## Call:
## lm(formula = dep_delay ~ temp + dewp + humid + wind_dir + wind_speed +
##     wind_gust + precip + pressure + visib, data = myFlightWeather)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -148.50  -17.36  -11.68   -1.09 1298.71
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.788e+02  1.109e+01  34.147  < 2e-16 ***
## temp        -2.804e-01  4.171e-02  -6.722 1.79e-11 ***
## dewp         4.950e-01  4.471e-02  11.071  < 2e-16 ***
## humid       -2.742e-01  2.180e-02 -12.581  < 2e-16 ***
## wind_dir    -7.040e-03  6.941e-04 -10.143  < 2e-16 ***
## wind_speed   1.234e-01  5.934e-03  20.790  < 2e-16 ***
## wind_gust           NA         NA      NA       NA
## precip       4.704e+01  3.524e+00  13.349  < 2e-16 ***
## pressure    -3.282e-01  1.050e-02 -31.248  < 2e-16 ***
## visib       -2.239e+00  4.237e-02 -52.845  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.26 on 335568 degrees of freedom
## Multiple R-squared:  0.02648,    Adjusted R-squared:  0.02646
## F-statistic:  1141 on 8 and 335568 DF,  p-value: < 2.2e-16
```

Funny enough, the model does not calculate coefficients for wind_gust (they are NA in the above summary) why is this ? Well, linear regression expects the explaining variables to be independent, which in this case, they are not.

Check it out below, wind_gust actually correlates to wind_speed very much :

```
cor.test(~ wind_gust + wind_speed, myFlightWeather)
```

```
##
##  Pearson's product-moment correlation
##
## data:  wind_gust and wind_speed
## t = 1.3745e+10, df = 335580, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  1 1
## sample estimates:
## cor
##   1
```

Back to the model: we have an Adjusted R-squared, which tells us the amount of variation in dep_delay variable that is explained by variation in the explaining variables, here 0.02039 eg about 2%. which is very little. That means, that we cannot say, that dep_delays are associated with weather conditions.

3.7 Exercise

Is the age of the plane associated to delay?

```
## Note another technique to replace NA values
myPlaneDelays <- sqldf('select a.year,
                               case when dep_delay is null then 0
                                    else dep_delay end
                                 as dep_delay
                          from planes a inner join
                               flights b on a.tailnum = b.tailnum')
cor.test(~ dep_delay + year, myPlaneDelays)
```

```
##
##  Pearson's product-moment correlation
##
## data:  dep_delay and year
## t = 8.2698, df = 278860, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.01194765 0.01936887
## sample estimates:
##        cor
## 0.01565847
```

Peaesons correlations coefficient falls in the intervakl btwn -1 and 1. The closer to -1 or 1 it is, the stringer the correlation is, eg 0 means no correlation.

This test shows a Pearsons test value of 0.016, which means, that there is hardly any correlation. So, NO, age and departure delay are not associated.

3.8 Exercise

It seems like the plane manufacturer could use a cleaning. After that, how many manufacturers have more than 200 planes?
And how many flights are each manufacturer with more than 200 planes responsible for?

Lets take a look at the Manufacturer column:

```
sqldf('select manufacturer,
              count(*)
         from planes
        group by manufacturer ')
```

```
##                     manufacturer count(*)
## 1                     AGUSTA SPA        1
## 2                         AIRBUS      336
## 3               AIRBUS INDUSTRIE      400
## 4          AMERICAN AIRCRAFT INC        2
## 5             AVIAT AIRCRAFT INC        1
## 6         AVIONS MARCEL DASSAULT        1
```

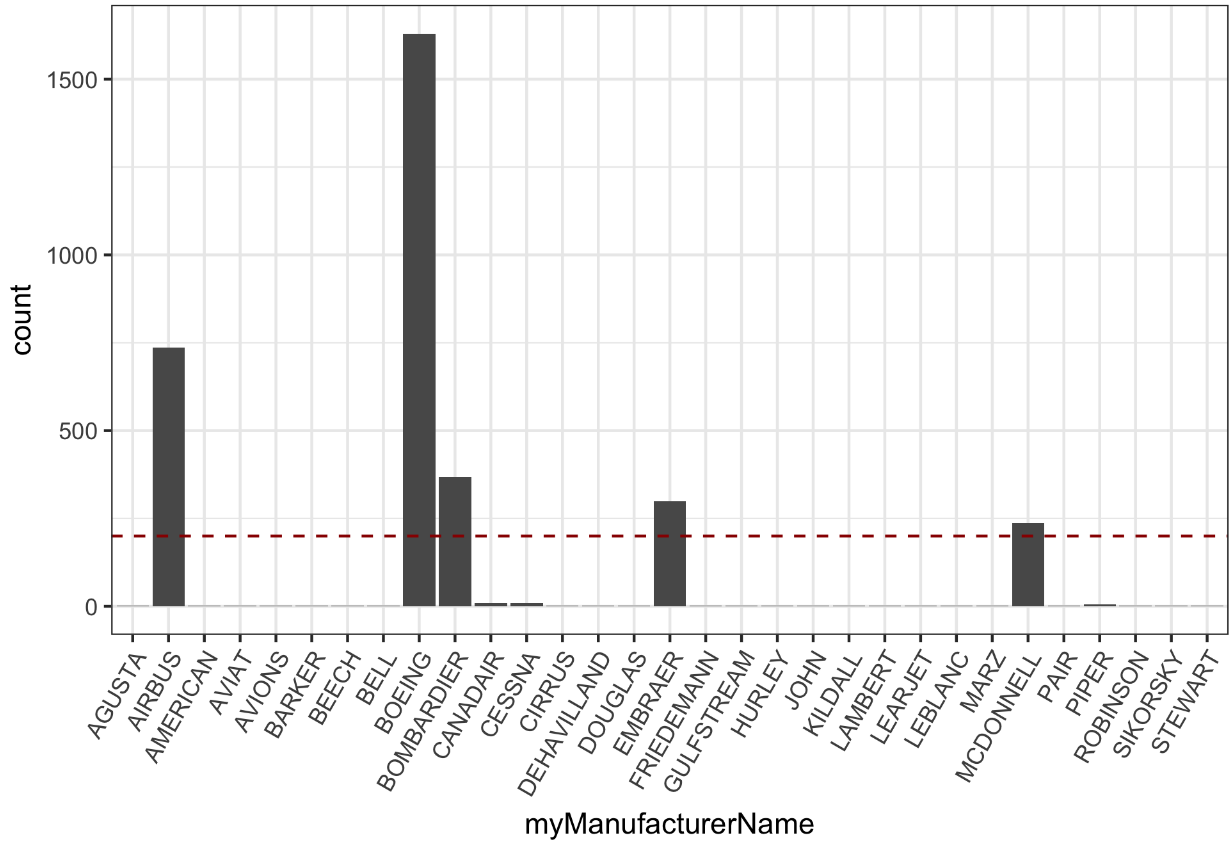
```
## 7                   BARKER JACK L       1
## 8                           BEECH       2
## 9                            BELL       2
## 10                         BOEING    1630
## 11                 BOMBARDIER INC     368
## 12                       CANADAIR       9
## 13                   CANADAIR LTD       1
## 14                         CESSNA       9
## 15             CIRRUS DESIGN CORP       1
## 16                    DEHAVILLAND       1
## 17                        DOUGLAS       1
## 18                        EMBRAER     299
## 19                 FRIEDEMANN JON       1
## 20           GULFSTREAM AEROSPACE       2
## 21             HURLEY JAMES LARRY       1
## 22                    JOHN G HESS       1
## 23                   KILDALL GARY       1
## 24                LAMBERT RICHARD       1
## 25                    LEARJET INC       1
## 26                LEBLANC GLENN T       1
## 27                     MARZ BARRY       1
## 28              MCDONNELL DOUGLAS     120
## 29 MCDONNELL DOUGLAS AIRCRAFT CO     103
## 30 MCDONNELL DOUGLAS CORPORATION      14
## 31                    PAIR MIKE E       1
## 32                          PIPER       5
## 33         ROBINSON HELICOPTER CO       1
## 34                       SIKORSKY       1
## 35                   STEWART MACO       2
```

It looks like the manuafacturer name can be standardized by isolating the first part of the string (until first occurance of space character).
Lets utillize some simple regex.

```r
myPlanes <- mutate(planes, myManufacturerName = str_extract(manufacturer,"^\\w*")) %>%
  select(myManufacturerName, tailnum)
```

Below, the number of planes pr. manufacturer is plottet. It is clear, that there are 5 manufacturers with more than 200 planes

```r
g <- ggplot(myPlanes, aes(myManufacturerName))
g + geom_bar() + theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
  geom_hline(aes(yintercept=200), colour="#990000", linetype="dashed")
```



Also, the actual count is listed.

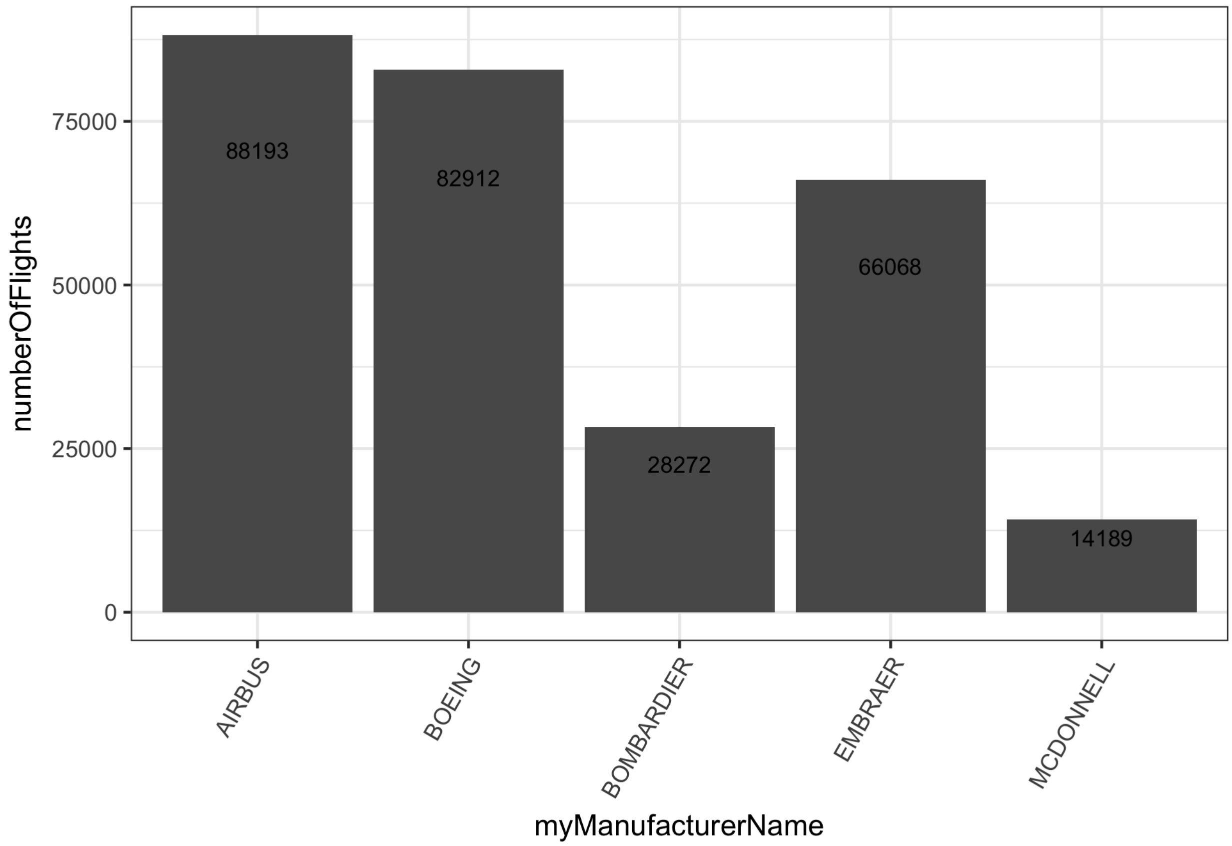
```
topManufacturers <- sqldf('select myManufacturerName, count(*)
                            from myPlanes
                           group by myManufacturerName
                          having count(*) > 200')
pander(topManufacturers, big.mark=',', justify = c('left','right'))
```

myManufacturerName	count(*)
AIRBUS	736
BOEING	1,630
BOMBARDIER	368
EMBRAER	299
MCDONNELL	237

Top manufacturers (>200 planes) are responsible for the following number of flights.

```
manufacturerFlights <-
sqldf('select a.myManufacturerName,
              count(*) as numberOfFlights
         from topManufacturers a
        inner join
              myPlanes b on a.myManufacturerName = b.myManufacturerName
        inner join
              flights c  on b.tailnum = c.tailnum
        group by a.myManufacturerName
```

```
        order by count(*) desc')

g <- ggplot(manufacturerFlights,
            aes(x=myManufacturerName, y=numberOfFlights, label=numberOfFlights))
g + geom_bar(stat='identity') + theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
  geom_text(size = 3, position = position_stack(vjust = 0.8))
```



3.9 Exercise

It turns out that Airbus has several main models, e.g. several A320 (A320-211, A320-212 etc.) and so on.
Create a frequency table of the main models for Airbus and how many planes there are in each.

Assuming that Airbus main models are all defined by the first 4 characters in the name:

```
airbusFreqs <- sqldf('select substr(model,1,4) as model, count(*) as frequency
                        from planes
                      where lower(manufacturer) like "%airbus%"
                      group by substr(model,1,4)
                      order by count(*) desc')
pander(airbusFreqs, big.mark=',', justify = c('left','right'))
```

model	frequency
A320	415

model	frequency
A319	208
A321	94
A330	18
A340	1

3.10 Exercise

Are larger planes (measured by number of seats) more or less delayed than smaller planes?

This question implies a correlation between the number of seats and delay - lets test this

```
seatDelays <- sqldf('select a.seats,
                            case when arr_delay is null then 0
                                 else arr_delay end
                              as arr_delay
                         from planes a inner join
                              flights b on a.tailnum = b.tailnum')
cor.test(~ arr_delay + seats, seatDelays)
```

```
##
##  Pearson's product-moment correlation
##
## data:  arr_delay and seats
## t = -37.382, df = 284170, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.07361066 -0.06629322
## sample estimates:
##         cor
## -0.06995288
```

There does not seem to be a correlation between number of seats and arr_delays (Pearsons -0.07). We can flit this model aswell

```
l <- lm(arr_delay ~ seats, seatDelays)
summary(l)
```

```
##
## Call:
## lm(formula = arr_delay ~ seats, data = seatDelays)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -90.96  -23.73  -11.18    6.82 1275.51
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12.856385   0.179292   71.71   <2e-16 ***
## seats       -0.043401   0.001161  -37.38   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44.44 on 284168 degrees of freedom
```

```
## Multiple R-squared:  0.004893,   Adjusted R-squared:  0.00489
## F-statistic:  1397 on 1 and 284168 DF,  p-value: < 2.2e-16
```

However the adjusted R-squared allso tells us, that very little of the variation in arrival delay is explained by the number of seats.
How about departure delay ?

```
seatDelays <- sqldf('select a.seats,
                            case when dep_delay is null then 0
                                 else dep_delay end
                              as dep_delay
                       from planes a inner join
                            flights b on a.tailnum = b.tailnum')
cor.test(~ dep_delay + seats, seatDelays)
```

```
##
##  Pearson's product-moment correlation
##
## data:  dep_delay and seats
## t = -27.02, df = 284170, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.05428841 -0.04695383
## sample estimates:
##        cor
## -0.0506218
```

There does not seem to be a correlation between number of seats and arr_delays (Pearsons -0.07).
We can fit this model aswell.

```
l <- lm(dep_delay ~ seats, seatDelays)
summary(l)
```

```
##
## Call:
## lm(formula = dep_delay ~ seats, data = seatDelays)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -54.19  -17.78  -13.75   -1.64 1294.84
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.870555   0.162426  103.87   <2e-16 ***
## seats       -0.028420   0.001052  -27.02   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40.26 on 284168 degrees of freedom
## Multiple R-squared:  0.002563,   Adjusted R-squared:  0.002559
## F-statistic: 730.1 on 1 and 284168 DF,  p-value: < 2.2e-16
```

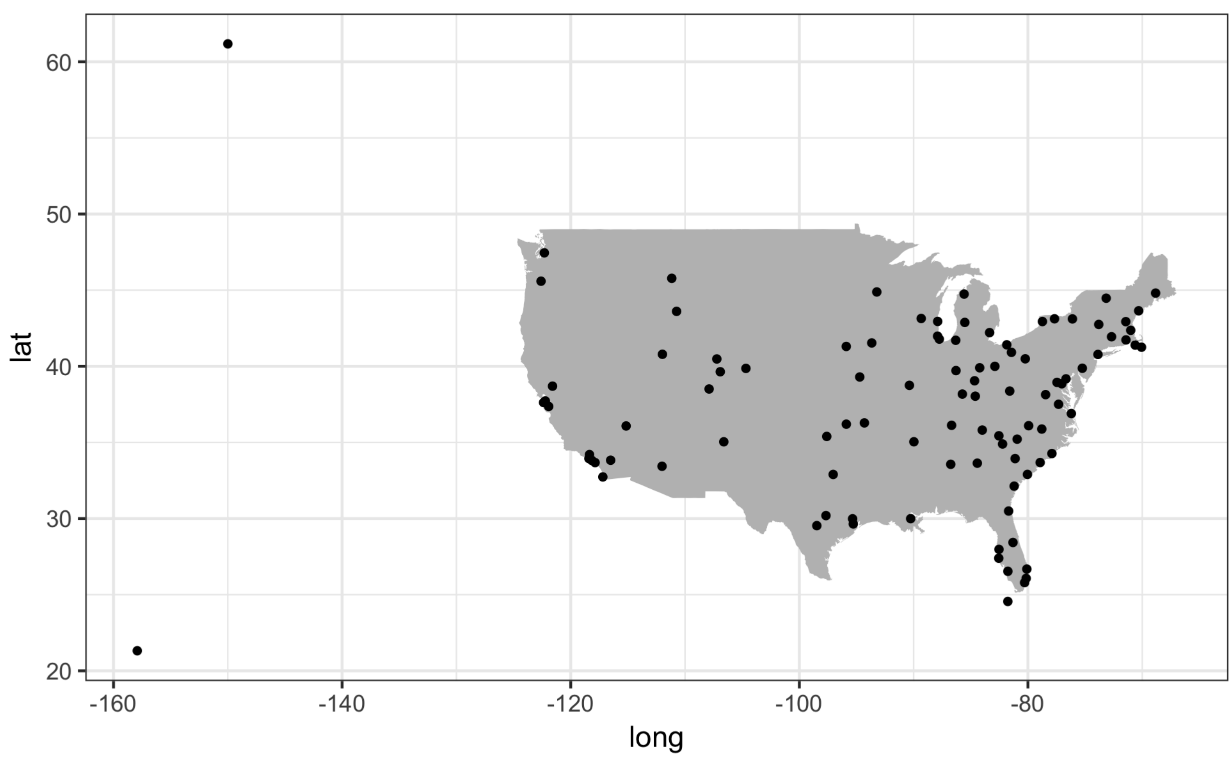
It looks like the same result - we cannot say, that there is a relationship between the number of seats and plane delay.

3.11 Exercise

On a map (`map_data("usa")`), plot the airports that has flights to them.

Creating a dataset with destination airport coordinates (long/lat).
Below is two different approaches for illustration purposes, SQL-syntax and R-syntax, utillizing pipe operator from dplyr. Lets continue with the R-version, not that it matters much - both contain the same data (101 observations, 3 variables)

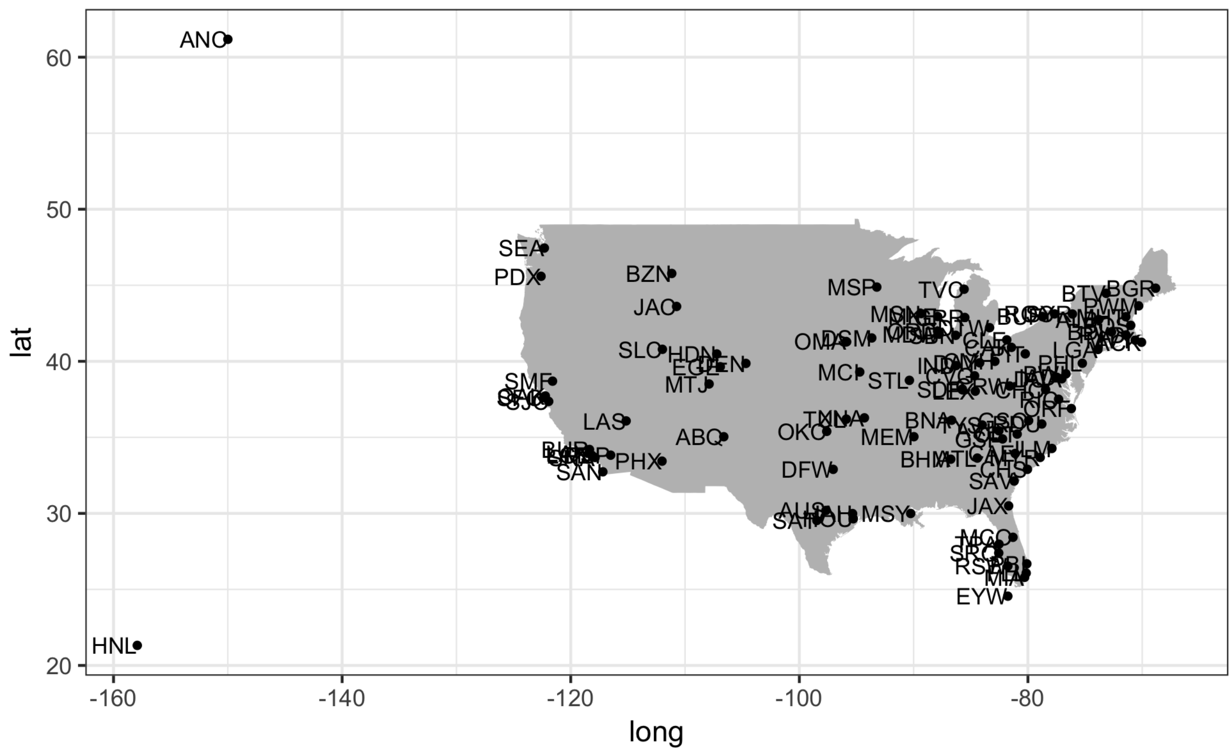
```
mySqlAirports <- sqldf('select distinct a.faa,
                               a.lat, a.lon
                          from airports a inner join
                               flights b on (a.faa = b.dest)
                         order by a.faa')
myAirports <- airports %>% mutate(dest = faa) %>% inner_join(flights) %>%
  select(faa, lat, lon) %>% distinct %>% arrange(faa)

## Adding the myAirports dataset as points on the map
usa <- map_data("usa")
myMap <- ggplot() +
  geom_polygon(data = usa, aes(x = long, y = lat, group = group), fill = "grey") +
    geom_point(data = myAirports, aes(x = lon, y = lat), color = "black", size = 1) +
  coord_quickmap()
myMap
```



Oops, two destination airports are outside the map.
Lets just add a label to see, that its ANC (Anchorage) and HNL (Honolulu), which is in Alaska and Hawaii.

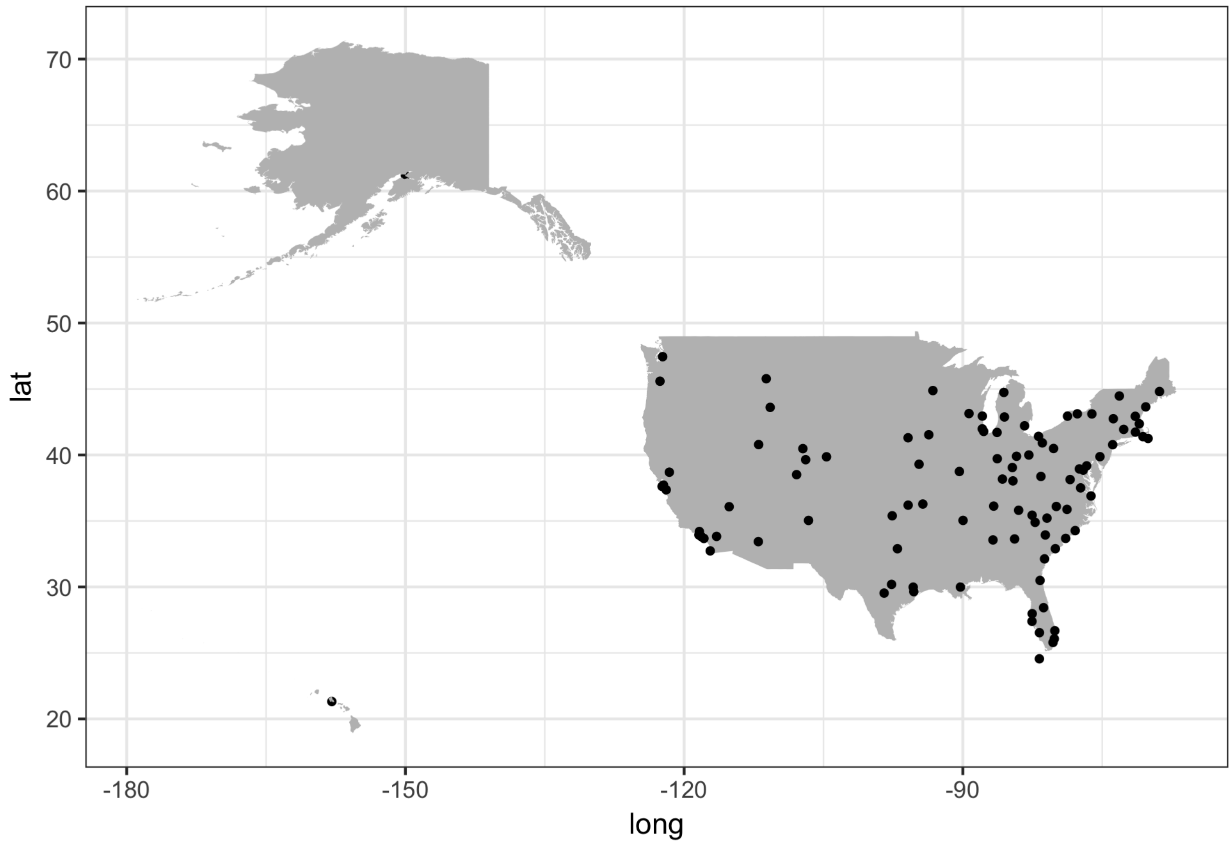
So, apparently, the USA map only covers mainland USA

```
myMap + geom_text(data = myAirports,
                  aes(x = lon, y = lat, label = faa), color = "black", hjust=1,size=3)
```



We have to borrow Canada and Hawaii from the world map.
Alaska includes far western Aleutian Islands, but we dont need them here and they distort the map because they are so far away, so lets drop everything further west than 180 degrees and add Alaska and Hawaii to the map.

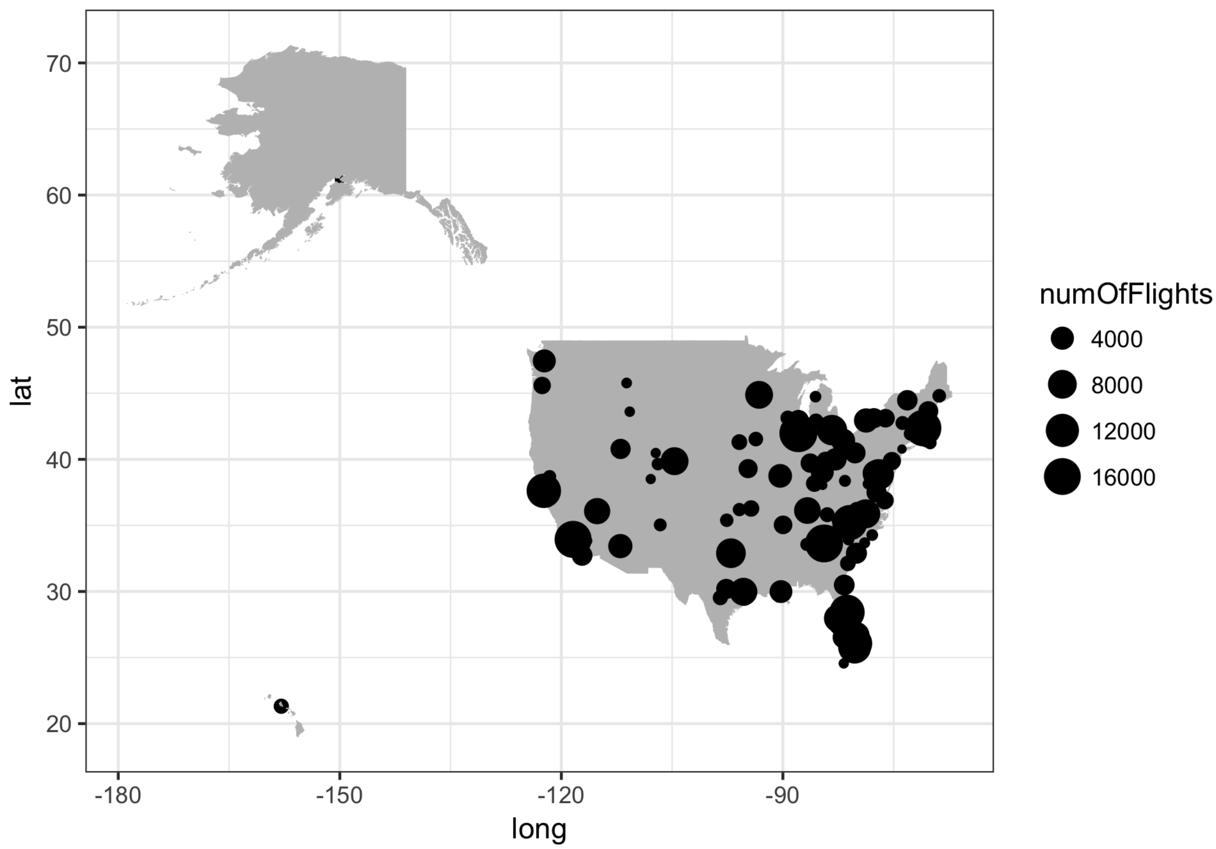
```
library(mapdata)
ak<-map_data('worldHires','USA:Alaska')
ak<-subset(ak,long<0)
hw<-map_data('worldHires','Hawaii')

myMapOfNorthAm <- myMap +
  geom_polygon(data=ak,     aes(x = long, y = lat, group = group), fill="grey") +
  geom_polygon(data=hw,     aes(x = long, y = lat, group = group), fill="grey")
myMapOfNorthAm
```



Make a similar plot, but now points must have size relative to the number of flights each airport is destination for.

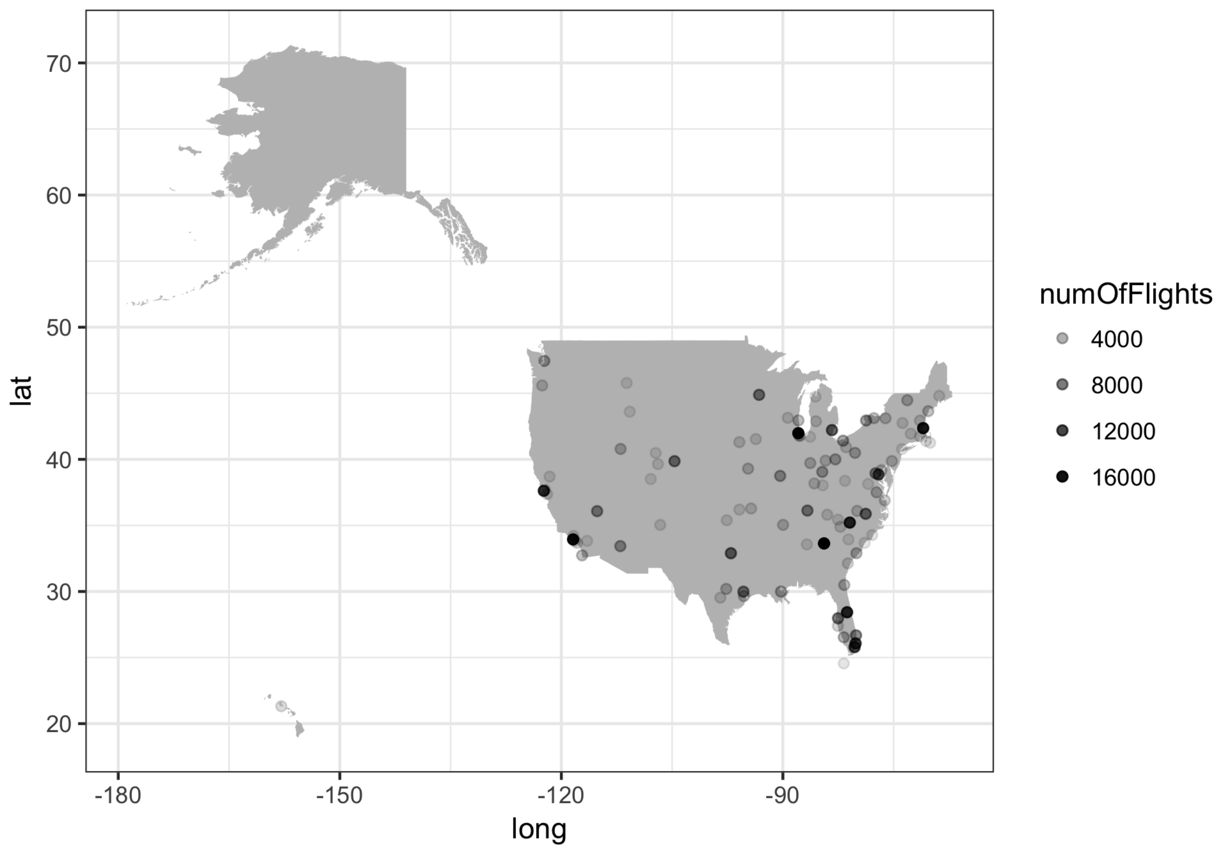
Lets make a new variable that describe this number of flights, and use this variable for the "size" parameter on the map

```
myAirportsAndflights <- sqldf('select a.faa, a.lat, a.lon, count(*) as numOfFlights
                                 from airports a inner join
                                      flights b on (a.faa = b.dest)
                                group by a.faa, a.lat, a.lon
                                order by a.faa')

ggplot() +
  geom_polygon(data = usa, aes(x = long, y = lat, group = group), fill = "grey") +
    geom_point(data = myAirportsAndflights,
               aes(x = lon, y = lat, size=numOfFlights), color = "black") +
    geom_polygon(data=ak,    aes(x = long, y = lat, group = group), fill="grey") +
    geom_polygon(data=hw,    aes(x = long, y = lat, group = group), fill="grey")
```



That makes airports blur together. Lets try alpha instead.
That makes it easier to see the top destinations, but it also erases the lowest ranking so it depends on what we are trying th achieve.

```
ggplot() +
  geom_polygon(data = usa, aes(x = long, y = lat, group = group), fill = "grey") +
    geom_point(data = myAirportsAndflights,
               aes(x = lon, y = lat, alpha=numOfFlights), color = "black") +
    geom_polygon(data=ak,    aes(x = long, y = lat, group = group), fill="grey") +
    geom_polygon(data=hw,    aes(x = long, y = lat, group = group), fill="grey")
```



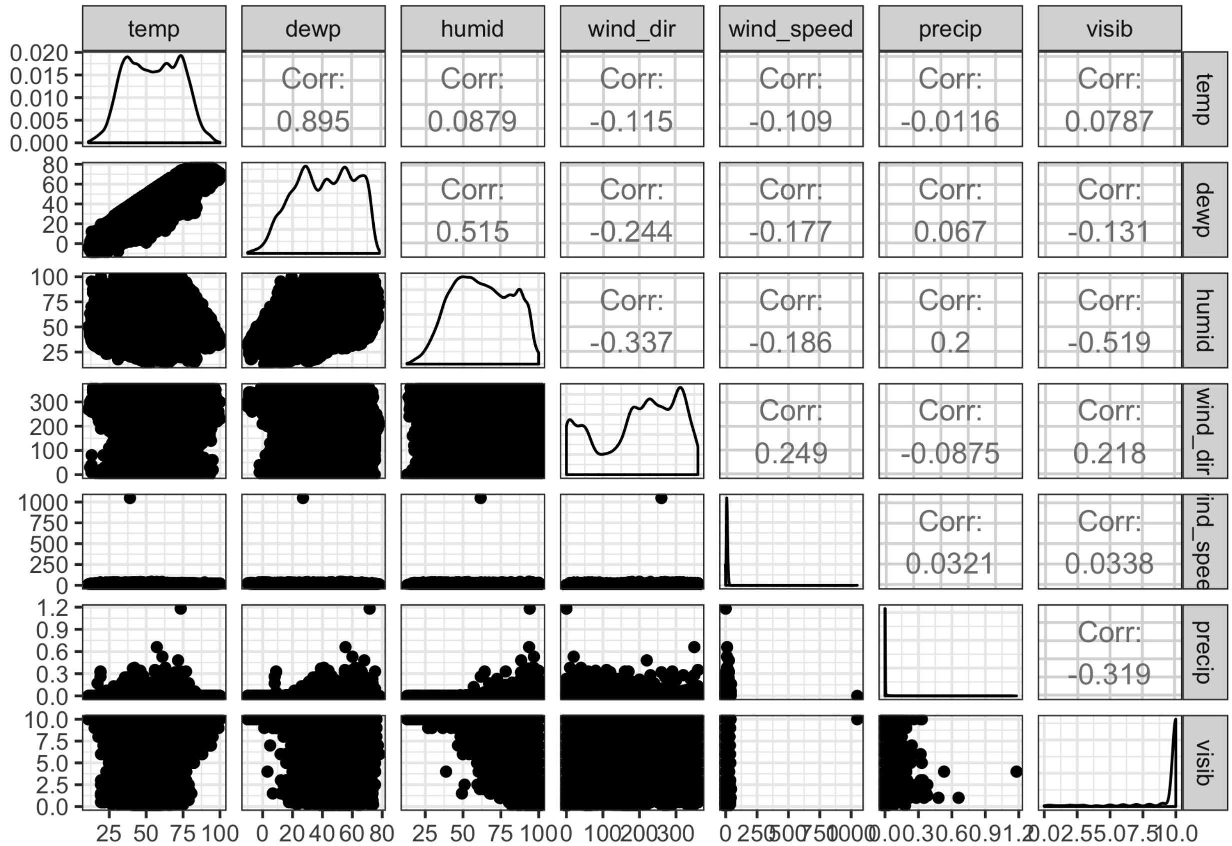
3.12 Exercise

Do a principal component analysis of the weather at JFK using the following columns: `temp, dewp, humid, wind_dir, wind_speed, precip, visib` (only on `complete.cases()`).

** How many principal components should be used to capture the variability in the weather data?**

Lets start by ploting each variable againt eachother, to get a visual sense of data.

```
myPcaWeather <- weather %>% select(temp, dewp, humid, wind_dir, wind_speed, precip, visib)
myCompleteWeather <- myPcaWeather[complete.cases(myPcaWeather), ]
library(GGally)
GGally::ggpairs(myCompleteWeather)
```



First of all, we see that variables are on completely different scales, so we need to normalize the variables when doing the PCA. Secondly, we see, that temperature (temp) and dewpoint(dewp) seem correlated, which could indicate, that there is some redundancy in the variables (inter-correlations). Just for fun, lets test the correlation between temp and dewp using the cor.test to show, that it is essentially the same measure as in the plot above.:

```
cor.test(weather$temp, weather$dewp)
```

```
##
##  Pearson's product-moment correlation
##
## data:  weather$temp and weather$dewp
## t = 322.8, df = 26127, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8917070 0.8965691
## sample estimates:
##       cor
## 0.8941644
```

A persons correlation of 0.89 reviels a almost perfect correlation. Without being a meteorologist, this still doesnt come as a surprise.

So, lets do the PCA to see, if we can reduce the number of variables.
We will do the PCA and take a look at the portion of variance explained by each PC.

To determine how many principle components to be used to determine a "reasonable" amount of the total variance, different methods could be applied. First off, the "eigenvalue-one" criterion (or the "Kaiser criterion") could be applied- which means choosing PCs with an eigenvalue greater than 1.

This is equivalent to choosing any PC that is responsible for a greater part of the variance than any one variable. prcomp stores the standard-deviations of each PC which is just the square root of the variance (eg. the eigenvalue).

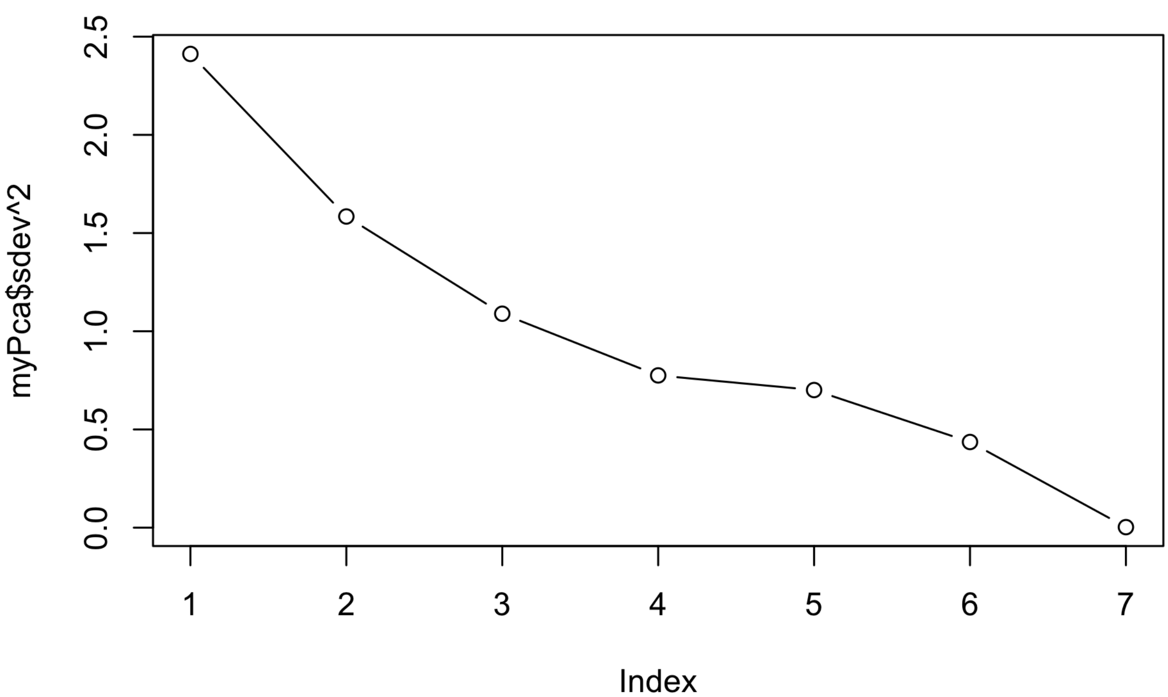
Ths suggests using the first three PCs.

```
myPca <- prcomp(myCompleteWeather, scale. = T)
myPca$sdev^2
```

```
## [1] 2.411839223 1.584349107 1.089191947 0.775082278 0.700694300 0.436111338
## [7] 0.002731807
```

Another criterion would be the "Scree test", which requires us to plot the eigenvalues.
In this approach, we look for a sudden descend in the variance accounted for. In this case, there is no obvious break, except maybe after the fifth PC. This break is not that significant, but it does suggest using the first 5 PCs

```
plot(myPca$sdev^2, type="b")
```



A third approach could be the proportion of variance accounted for.

We could decide, that we want at least 70% of the variance in the data explained and that we wanted to include all PCs that explain atleast 10% of the variance. These measures is printed below, and this indicates that the first 5 PC??s should be used, since PC5 accounts for almost excactly 10% and this would mean including about 93% of the total variance.

So, since two of the approaches suggest using 5 PCs, that is what I will choose.

Another reason for choosing 5 PCs is that in this case we can afford it, with regards to the size of the dataset and processing time, if we were to proceed building a model on these new variables (PCs). In other scenarios it could be, that the amount of data would be a reason to minimize the number of PCs chosen for further processing.

```
summary(myPca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6     PC7
## Standard deviation     1.5530 1.2587 1.0436 0.8804 0.8371 0.6604 0.05227
## Proportion of Variance 0.3446 0.2263 0.1556 0.1107 0.1001 0.0623 0.00039
## Cumulative Proportion  0.3446 0.5709 0.7265 0.8372 0.9373 0.9996 1.00000
```