FLS 5 Haitie Liu

BBALL STUDY

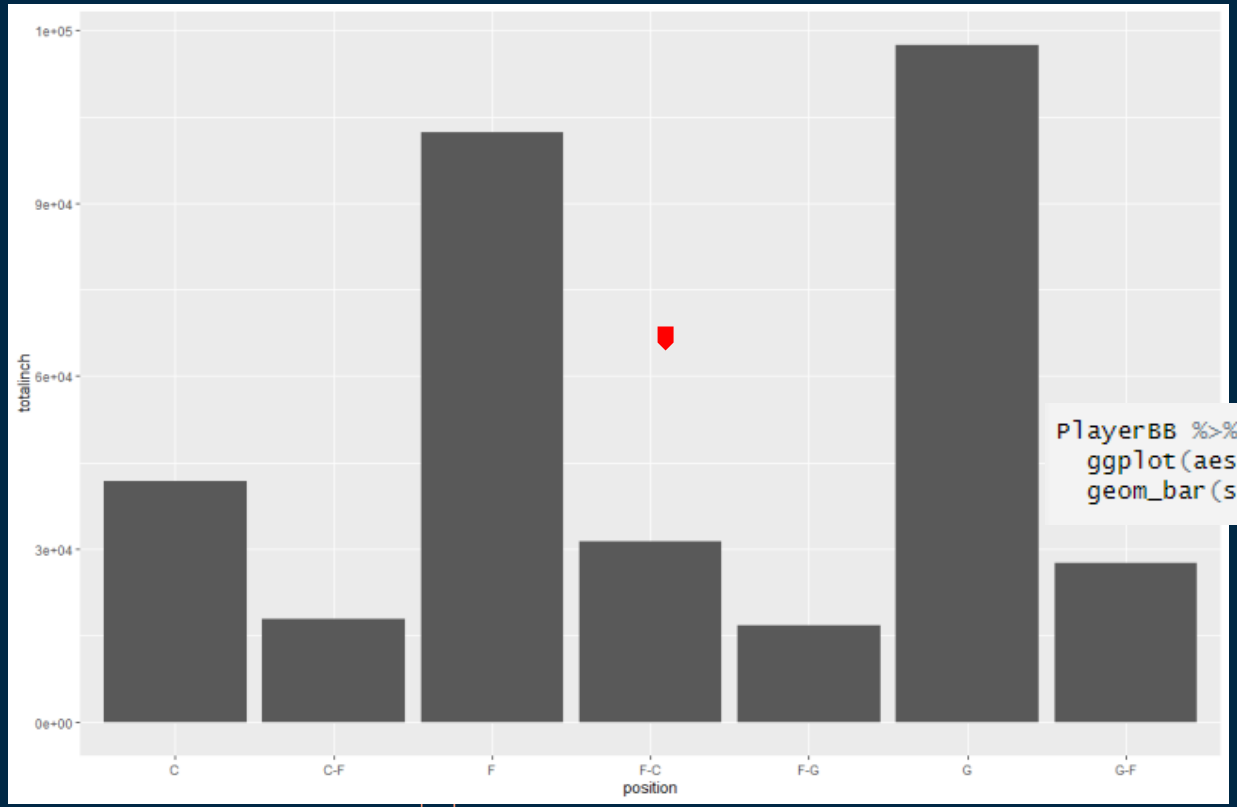
```
PlayerBB=read.csv(file.choose(),header=TRUE)
PlayerBB=separate(PlayerBB,height,into=c("feet","totalinch"),"-")
PlayerBB[2143,5]="6"
PlayerBB[2143,6]="2"
PlayerBB[2143,4]="G"
PlayerBB$totalinch=as.numeric(PlayerBB$feet)*12+as.numeric(PlayerBB$totalinch)
PlayerBB$position=as.factor(PlayerBB$position)
```

Loading data as csv file and "tidy" the dataset, filling in missing value at row 2143.

Used separate function to create a new row, totalinch = feet*12+inch

Chaning position from character to factor

BBALL STUDY



Plot position vs. Height in total inch

Fill=position does not plot color, still trying to figure out why

```
PlayerBB %>%
  ggplot(aes(x=position,y=totalinch),fill=position) +
  geom_bar(stat = "identity")
```

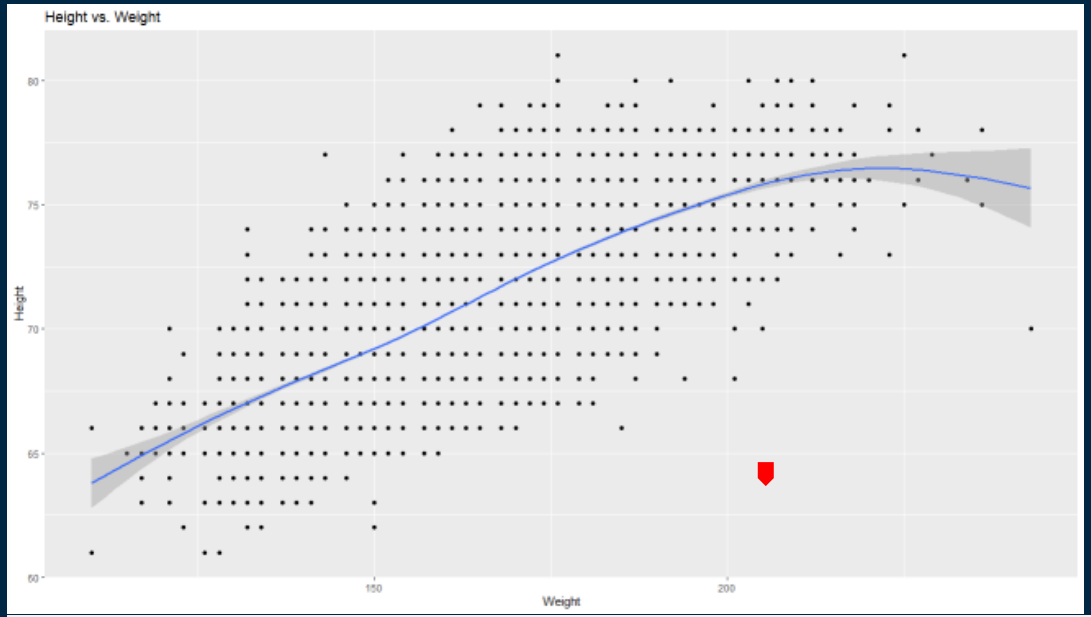
FIFA STUDY . A

```
FIFA=read.csv(file.choose(),header = TRUE)

class(FIFA$Position)
sum(is.na(FIFA$Weight))

FIFA=separate(FIFA,Height,into = c("feet","height"),sep = "'" )
FIFA$feet=as.numeric(FIFA$feet)
FIFA$height=FIFA$feet*12+as.numeric(FIFA$height)
FIFA=drop_na(FIFA)
FIFA$Weight=as.numeric(substr(FIFA$Weight,1,3))
FIFA$Position=as.factor(FIFA$Position)
```

Similarly with previous slide, I used separate() function to manipulate feet and inch. Created FIFA$height column.

For weight, I used substr() function to erase string "lbs" then changing the column as.numeric().

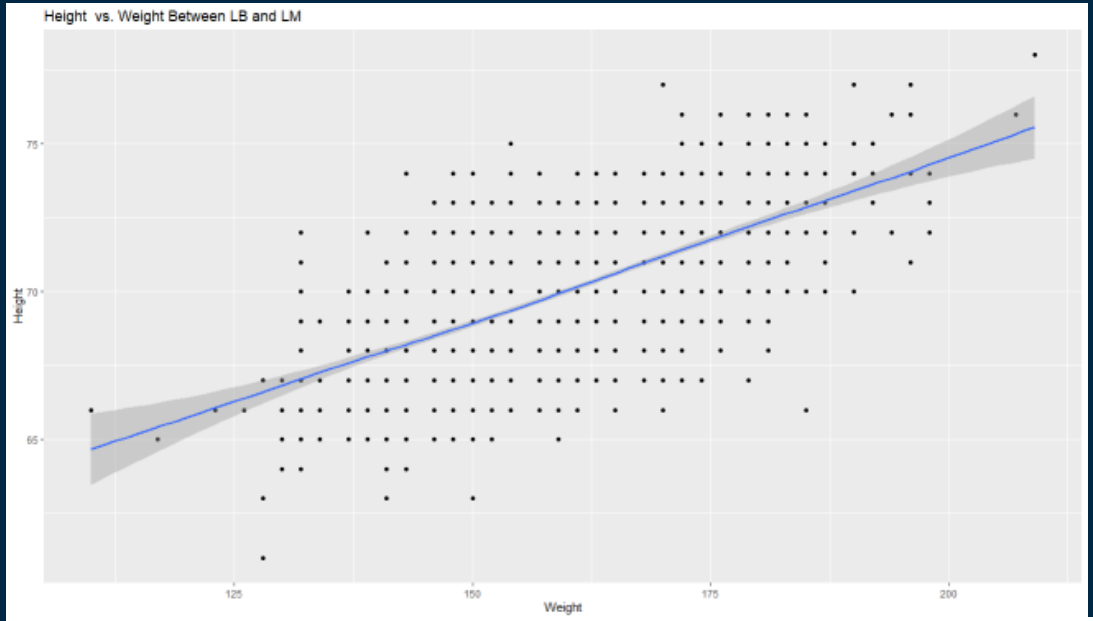
FIFA STUDY . A




```
FIFA %>%
  select (height,Weight,Position) %>%
  ggplot(aes(x=Weight,y=height),color=Position)+geom_point()+geom_smooth()+
  ggtitle("Height vs. Weight") +ylab("Height")
```

Looking at the chart to the left, we can clearly see that there is a positive correlation between height and weight, clearly as weight increase, height increase. Using geom_smooth to plot a line indicating the correlation.

FIFA STUDY . B




```
FIFA %>%
  filter(Position == "LB" | Position == "LM") %>%
  ggplot(aes(x=Weight,y=height),color=Position)+geom_point()+geom_smooth()+
  ggtitle("Height  vs. Weight Between LB and LM") +ylab("Height")
```

Filtering row that contains only LB and LM position. Then plot the line similarly with previous slide.

We can see that:
The linear correlation between height and weight in position LM and LB is even stronger.

STUDY: BABY NAMES Question 1

```
df=read.table(file.choose(),header = FALSE)
summary(df)
structure(df)
df=separate(df,V1,into = c("name","gender","number"),";")

str_view(df$name,"yyy\\b") #looks like its df[212,] Fiona#
y2016=df[-212,]
```

```
> summary(df)
     name              gender             number
 Length:32868       Length:32868       Length:32868
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character
```

Load table df, then creating columns as
Name
Gender
Number

Using str_veiw("yyy\\b") to locate the mis-entered info:
Row 212 : "Fionayyy"

Delete row 212

Save new data as y2016

Baby Names: Question 2

```
y2015=read.table(file.choose(),header = FALSE)
y2015=separate(y2015,V1,into = c("name","gender","number"),",")

tail(y2015,n=10)
#### last 10 row, have all gender male
####total count of 5 and name starting with letter "Z"

final = join(x=y2015,y=df, type= "inner",by = "name")
```

```
> tail(y2015,n=10)
        name gender number
33054   Ziyu      M      5
33055   Zoel      M      5
33056  Zohar      M      5
33057 Zolton      M      5
33058   Zyah      M      5
33059 Zykell      M      5
33060 Zyking      M      5
33061  Zykir      M      5
33062  Zyrus      M      5
33063   Zyus      M      5
```

-Load file y2015,
Create new columns:
Name
Gender
Number

-Print last 10 rows

-Inner join both y2015 and y2016, name the data "final"

Baby Names: Question 3.1

```
colnames(final)[2]="gender1"
colnames(final)[3]="numberx"
colnames(final)[4]="gender2"
colnames(final)[5]="numbery"
final$numberx=as.numeric(final$numberx)
summary(final)

final2=final %>%
  group_by(name)%>%
  filter(gender1==gender2)%>%
  summarise(name,gender1,gender2,totalnumber=numberx+numbery) %>%
  as.data.frame()

final2=arrange(final2,desc(totalnumber))

head(final2,10) #############top 10 popular names
```

```
> head(final2,10) #############top 10 popular names
       name gender1 gender2 totalnumber
1      Emma       F       F       39829
2    Olivia       F       F       38884
3      Noah       M       M       38609
4      Liam       M       M       36468
5    Sophia       F       F       33451
6       Ava       F       F       32577
7     Mason       M       M       31783
8   William       M       M       31531
9     Jacob       M       M       30330
10 Isabella       F       F       30296
```

-Re-Ording the columns of "final", group by name, filter rows which gender from 2015 and gender from 2016 is equal. Therefore, we are only showing the correct name corresponding to the correct gender.

-Save the new data as "final2"

-There are 26,550 names recorded, and the top 10 is shown to the left.

Baby Names: Question 3.2

```
girlfinal=final2 %>%
  filter(gender1 == "F" & gender2 == "F")

head(girlfinal,10) #######Top 10 girl name

final3=girlfinal[1:10,]
write.csv(final3,"D:\\R")
```

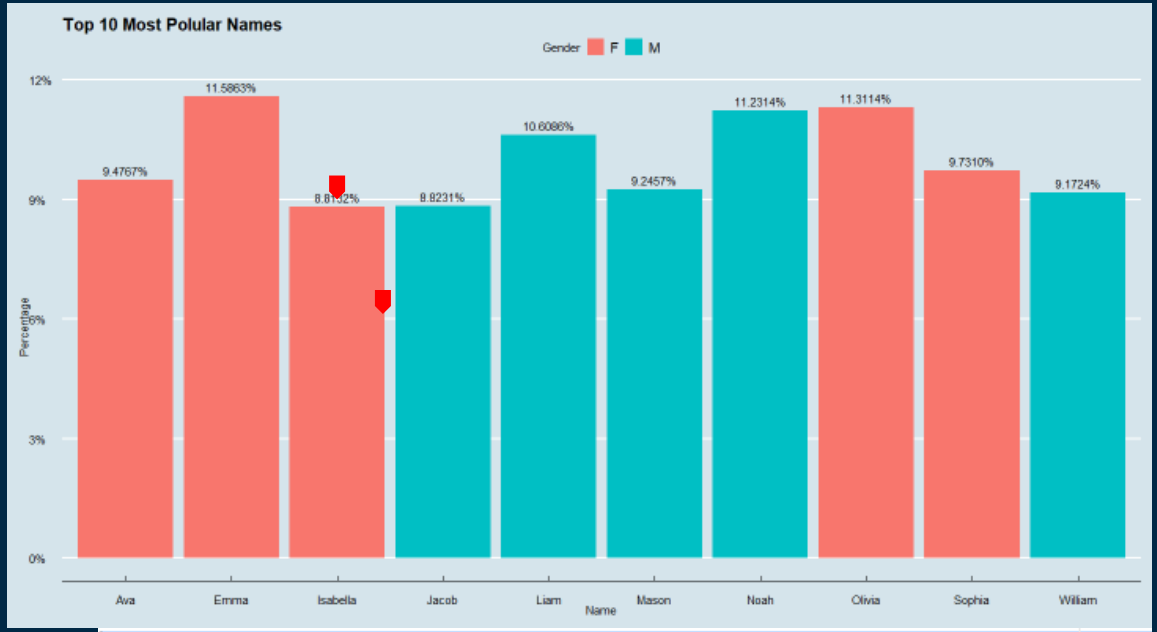
```
> head(girlfinal,10) #######Top 10 girl name
        name gender1 gender2 totalnumber
1       Emma       F       F        39829
2     Olivia       F       F        38884
3     Sophia       F       F        33451
4        Ava       F       F        32577
5   Isabella       F       F        30296
6        Mia       F       F        29237
7  Charlotte       F       F        24411
8    Abigail       F       F        24070
9      Emily       F       F        22692
10    Harper       F       F        21016
```

-Using filter () to show only girls name, gender == F

-Save the new data

-Write out to csv file using "girlfinal [1:10,]"

Baby Names: Question 4 Visualization



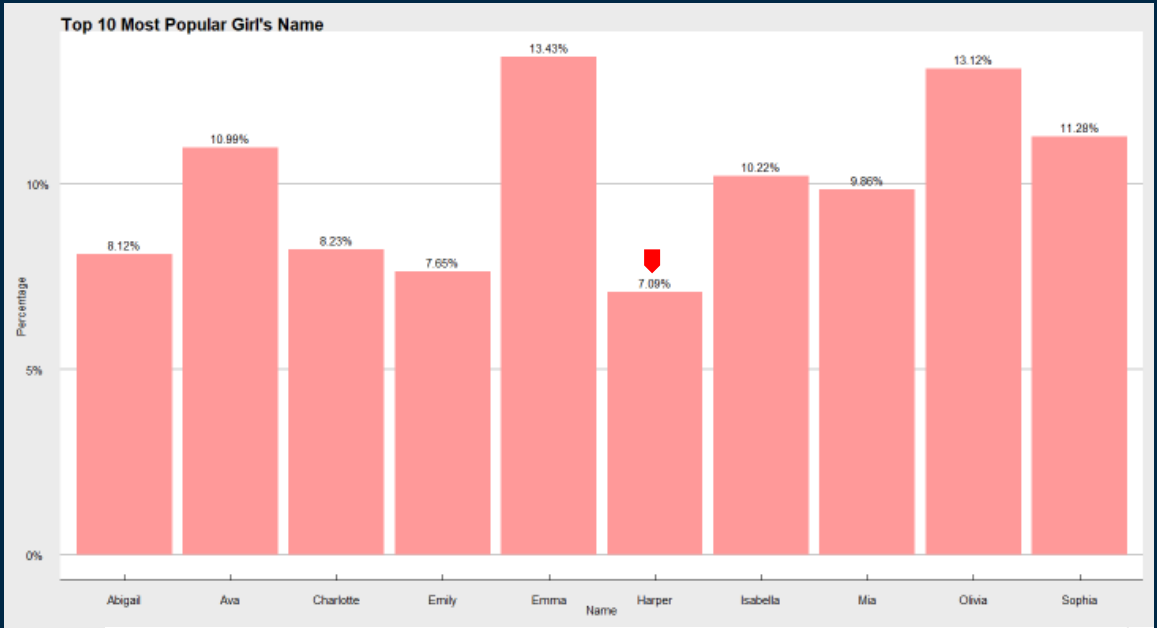

```
finalall%>%
  ggplot(aes(x= name,  group=gender1)) +
  geom_bar(aes(y = totalnumber/sum(totalnumber), fill = gender1), stat="identity") +
  scale_y_continuous(labels = scales::percent,)+
  geom_text(aes( label = scales::percent(totalnumber/sum(totalnumber)),
                 y= totalnumber/sum(totalnumber) ), stat= "identity", vjust = -.5)+
  ggtitle("Top 10 Most Polular Names") + xlab("Name")+ ylab("Percentage") +
  scale_fill_discrete(name="Gender")+
  ggthemes::theme_economist()
```

-Plot total data for both girls and boys, listing top 10 here to the left

-Show y axis as percentage, we can see that there are 5 girls name and 5 boys name made to the top 10

-"Emma","Noah","Olivia" all made to the top 3, and they are all over 11 percent out the top 10.

Baby Names: Question 4 Visualization

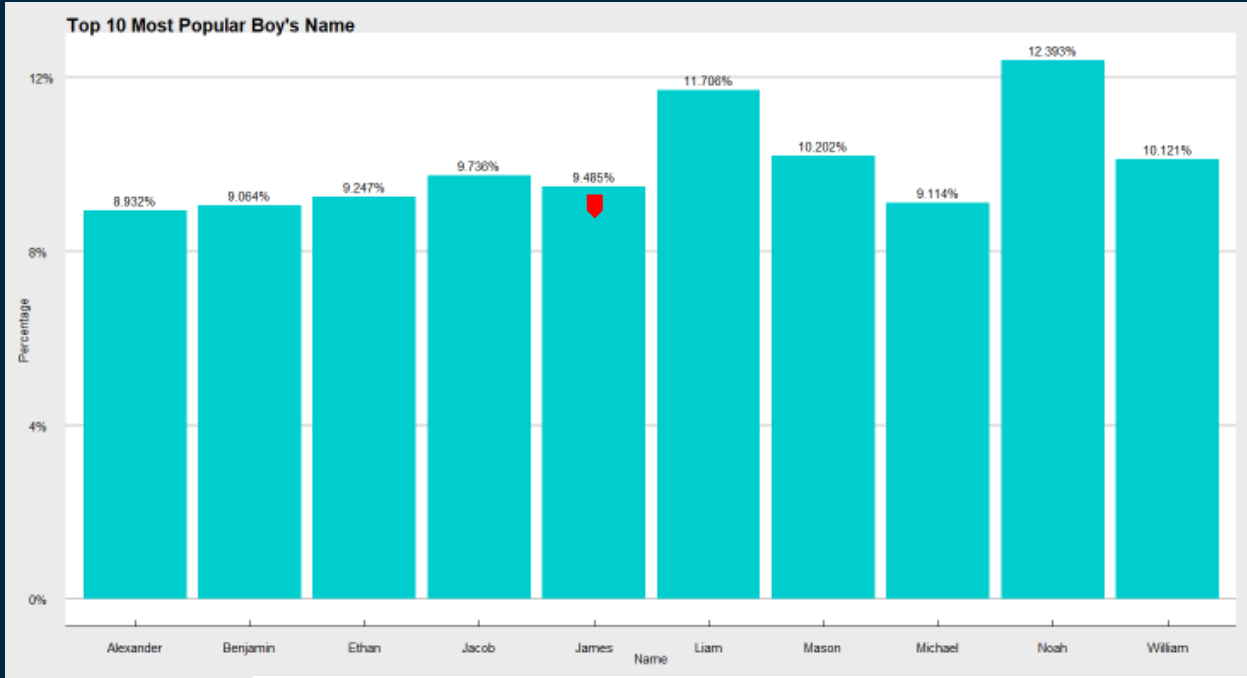



```
plotgirl=final3 %>%
  ggplot(aes(x=name))+
  geom_bar(aes(y=totalnumber/sum(totalnumber)),fill="#FF9999",stat = "identity")+
  scale_y_continuous(labels = scales::percent)+
  geom_text(aes(label=scales::percent(totalnumber/sum(totalnumber)),
                y=totalnumber/sum(totalnumber)),stat = "identity", vjust= -.5)+
  ggtitle("Top 10 Most Popular Girl's Name")+ xlab("Name") + ylab("Percentage")+
  scale_fill_discrete(name="Gender")+
  ggthemes::theme_economist_white()
```

-This is the data for top 10 girls' name.

-Similarly, we can tell that "Emma" and "Olivia" are the most popular both over 13% of the top 10.

Baby Names: Question 4 Visualization




```
plotboy=finalboys%>%
  ggplot(aes(x=name))+
  geom_bar(aes(y=totalnumber/sum(totalnumber)),fill="cyan3",stat = "identity")+
  scale_y_continuous(labels = scales::percent)+
  geom_text(aes(label=scales::percent(totalnumber/sum(totalnumber)),
                y=totalnumber/sum(totalnumber)),stat = "identity", vjust= -.5)+
  ggtitle("Top 10 Most Popular Boy's Name")+ xlab("Name") + ylab("Percentage")+
  scale_fill_discrete(name="Gender")+
  ggthemes::theme_economist_white()
```

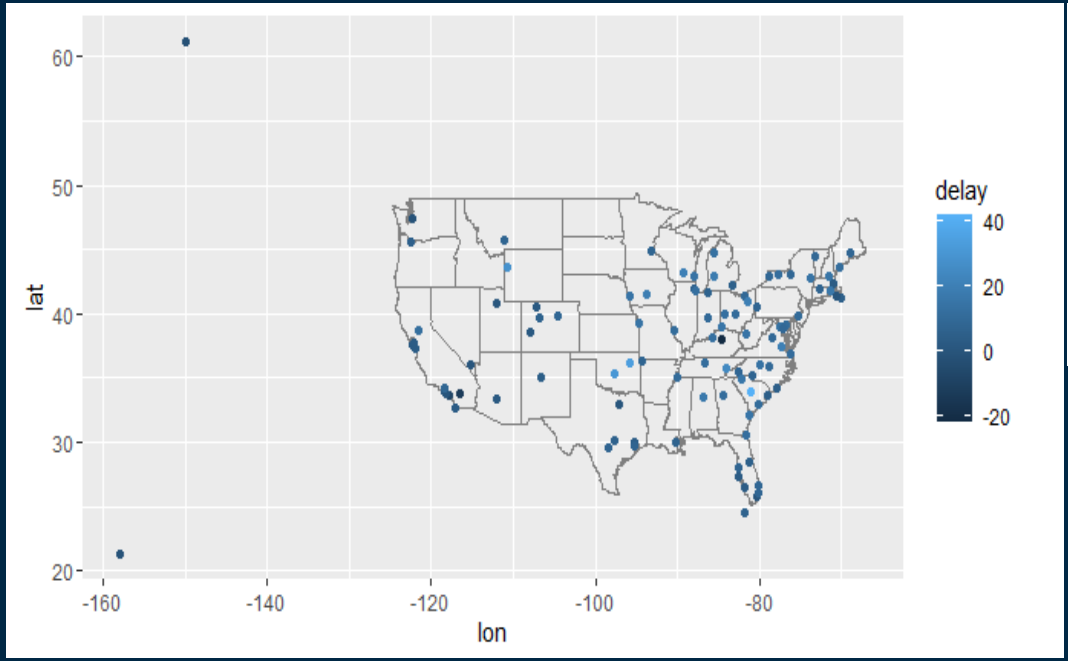
-This is the data for top 10 boys' name.

-"Noah" , "Liam" , "William", "Mason" are the top 4, have percentage over 10% against all top 10

TAKE AWAY

-Running into problems when try to make data tidy. Sometimes a really small problem could take hours to solve. Making data tidy is a rigorous and essential job for data scientist, many times I overlook the "small things"

Exercises 1 (pages 186-187) from the Wickham text



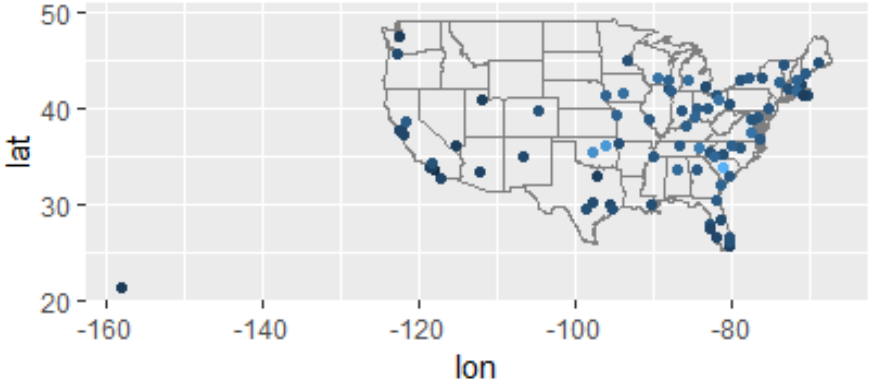

```
library(tidyverse)
library(nycflights13)

avg_dest_delays <-
  flights %>%
  group_by(dest) %>%
  transmute(delay = mean(arr_delay,na.rm = TRUE)) %>%
  unique()%>%
  inner_join(airports,by=c(dest="faa") )


avg_dest_delays %>%
  ggplot(aes(lon, lat,color=delay)) + borders("state") +
  geom_point() + coord_quickmap()
```

Group by dest
Then create a different column

Inner join with airports

Plot + coord_quickmap

Exercises 2-5 (pages 186-187) from the Wickham text




```
############question number 2
testflight=flights
colnames(avg_dest_delays)[1]="avgdelay"

testflight=avg_dest_delays %>%
  select(avgdelay,delay,lat,lon) %>%
  right_join(testflight,by=c(avgdelay="dest"))

###########question 3
###Is there a relationship between the age of a plane and its delays?
###I think there is, but I have no data to prove it

##########question 4
#what weather conditions make it more likely to see a delay?
#anwser: rain

##########question 5
testflight=separate(testflight,time_hour,into=c("date","hour"),sep = " ")
str(testflight$date)

testflight%>%
  filter(date=="2013-06-13")%>%
  ggplot(aes(lon,lat,color=delay)) +borders("state")+
  geom_point()+coord_quickmap()
```

Question 2:
Right_join, avgdelay,delay,lat,lon col into flight

Question 5:
Above is the delay map for airports on the day of 2013-06-13

Sentences question from stringr

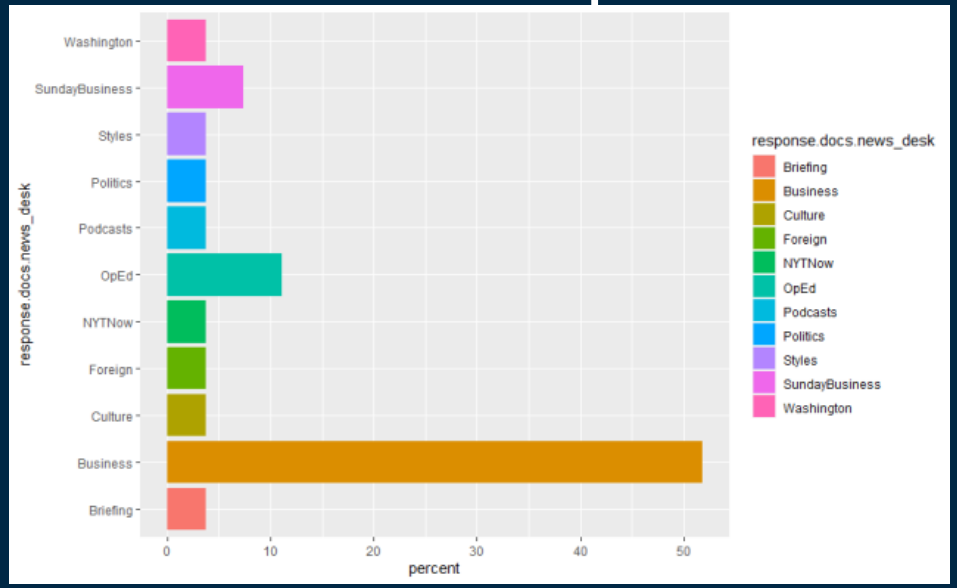
```
sentences
str_view_all(sentences,"'")
sentences1= str_replace_all(sentences,"'", "      ")
grep("      ",sentences1)
head(sentences1)
```

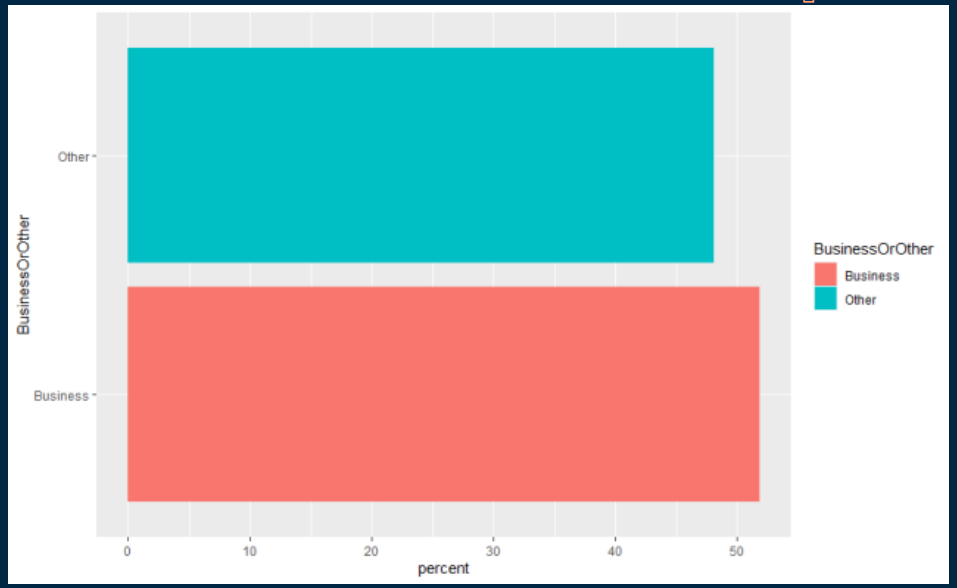
Using str_replace to separate out the " ' "

Using grep function to find where the " ' " are.

Wasn't sure what the question was asking, by plotting the frequency the after the contraction.

NYT API Duplication with Business or Other





Duplicated the design and looking into Business news or other

Key word: Elon Musk from 1-20 to 1-30

Still working on it but running out of time

TAKE AWAY

-For the second question(stringr), it would be helpful to see the solution.

-There are still many things unclear for me on the NYT example, need more time to study it. For example:
NYT API already classified reponse.docs.type_of_material for us, as news or Op-Ed, why are we creating another classifier to categorizing news or other. What is the purpose behind it.