Penguins & NYCflights dataset manipultion and visuaization

Luke Geel

10/19/2020

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(palmerpenguins)
```

Question 1 A. What are the percentages of missing values in bill_length_mm, bill_depth_mm, sex? (recall the percent() function in scales package)

```
d <- nrow(penguins)
a <- nrow(penguins%>%filter(is.na(bill_length_mm)))
b <- nrow(penguins%>%filter(is.na(bill_depth_mm)))
c <- nrow(penguins%>%filter(is.na(sex)))
"% of missing bill length:"
```

```
## [1] "% of missing bill length:"
```

```
a/d *100
```

```
## [1] 0.5813953
```

```
"% of missing bill depth:"
```

```
## [1] "% of missing bill depth:"
```

```
b/d*100
```

```
## [1] 0.5813953
```

```
"% of missing bill length:"
```

```
## [1] "% of missing bill length:"
```

```
c/d*100
```

```
## [1] 3.197674
```

B. What are the means and medians of the body mass for male and female penguins of each species? (Remove the NAs in the sex first)

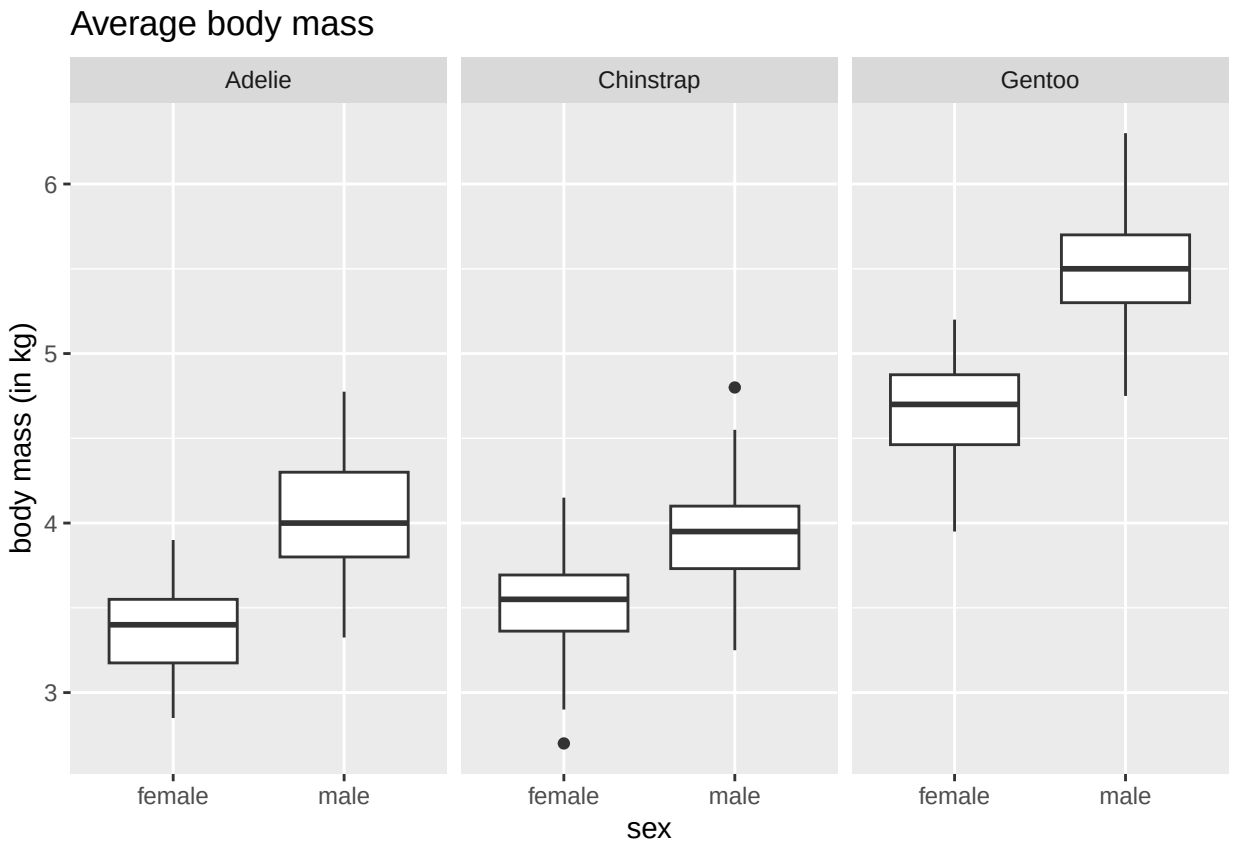
```
penguins_no_NA <- penguins %>% filter(!is.na(sex))
penguins_no_NA %>% group_by(species,sex) %>%summarize(mean_body_mass = mean(body_mass_g), median_body_ma
```

```
## `summarise()` has grouped output by 'species'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 6 x 4
## # Groups:   species [3]
##   species   sex    mean_body_mass median_body_mass
##   <fct>     <fct>           <dbl>            <dbl>
## 1 Adelie    female          3369.             3400
## 2 Adelie    male            4043.             4000
## 3 Chinstrap female          3527.             3550
## 4 Chinstrap male            3939.             3950
## 5 Gentoo    female          4680.             4700
## 6 Gentoo    male            5485.             5500
```

C. Make the following plot. (x label is removed, and the unit of y axis is changed to kilogram (kg))

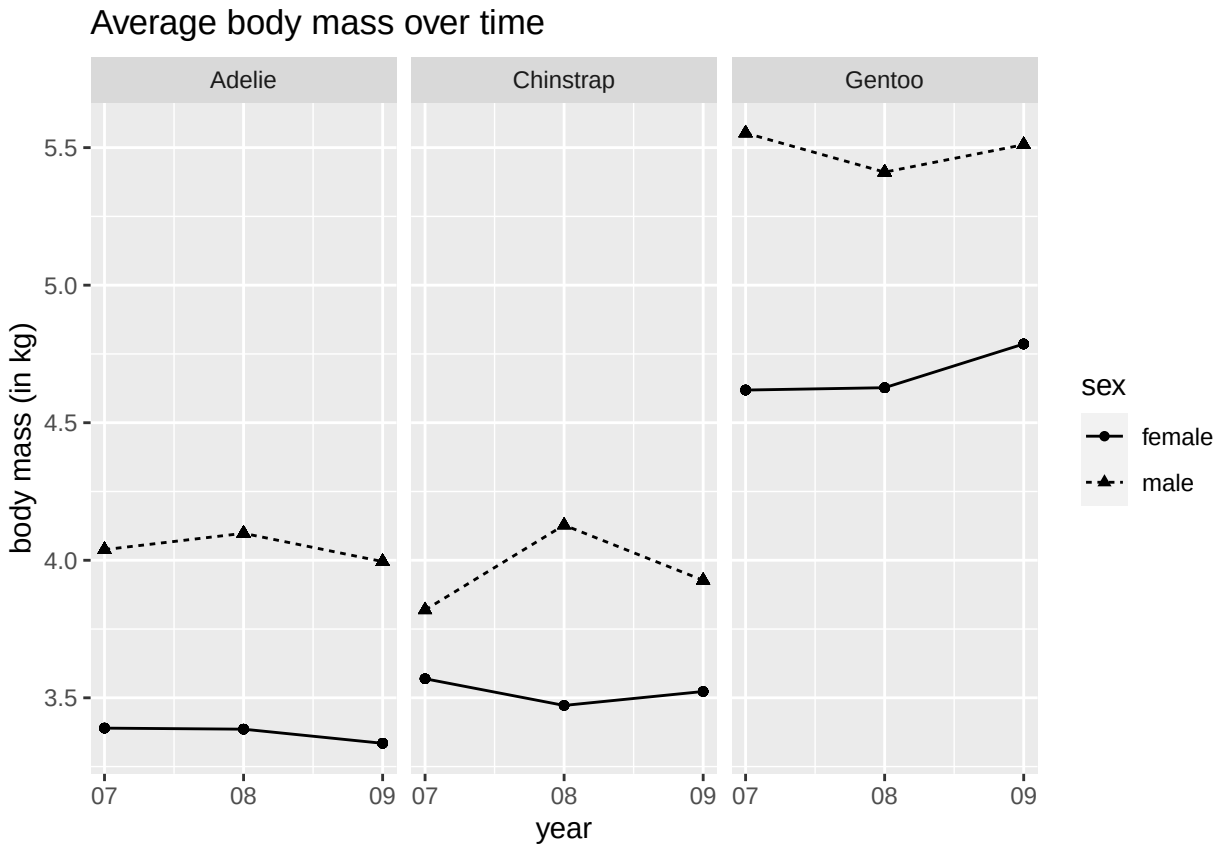
```
penguins_in_kg <- penguins %>%filter(!is.na(sex))%>%mutate(body_mass_kg = (body_mass_g/1000))
ggplot(data = penguins_in_kg)+
  geom_boxplot(mapping = aes(x=sex, y=body_mass_kg))+
  facet_wrap(. ~species, nrow =1)+
  labs(title = "Average body mass", y="body mass (in kg)")
```



D. Make the following plot.

```
penguins%>%filter(!is.na(sex))%>%group_by(sex, year,species)%>%mutate(body_mass_kg = (body_mass_g/1000))
ggplot(aes(x=year, y=mean))+
  geom_line(mapping = aes(linetype = sex))+
  geom_point(aes(shape = sex))+
  facet_wrap(~species)+
  labs(title = "Average body mass over time", y="body mass (in kg)", x="year")+
```

```
  scale_y_continuous(breaks=seq(0,5.5,0.5))+
  scale_x_continuous(breaks=seq(2007.0,2009.0,1), labels=c("07","08","09"))
```

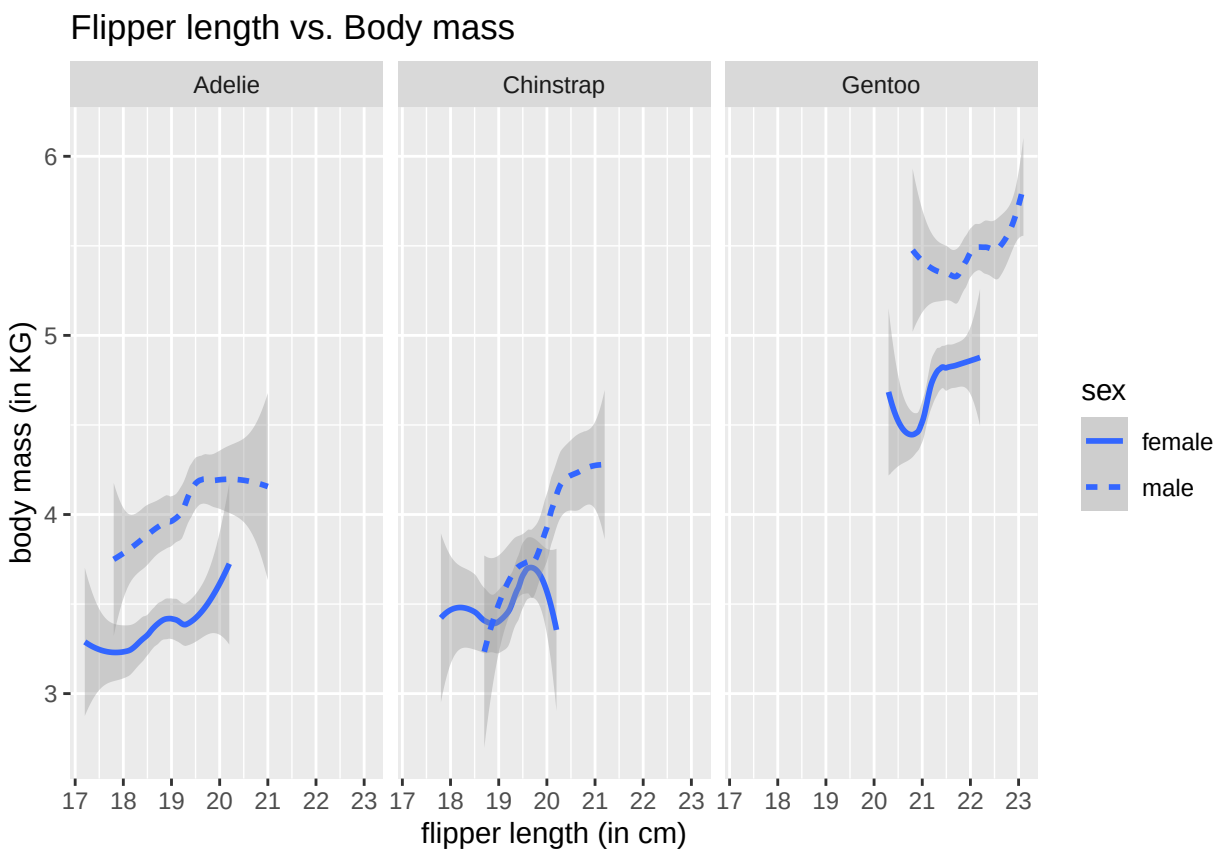


E.

Make the following two plots. (The units of the variables are changed)

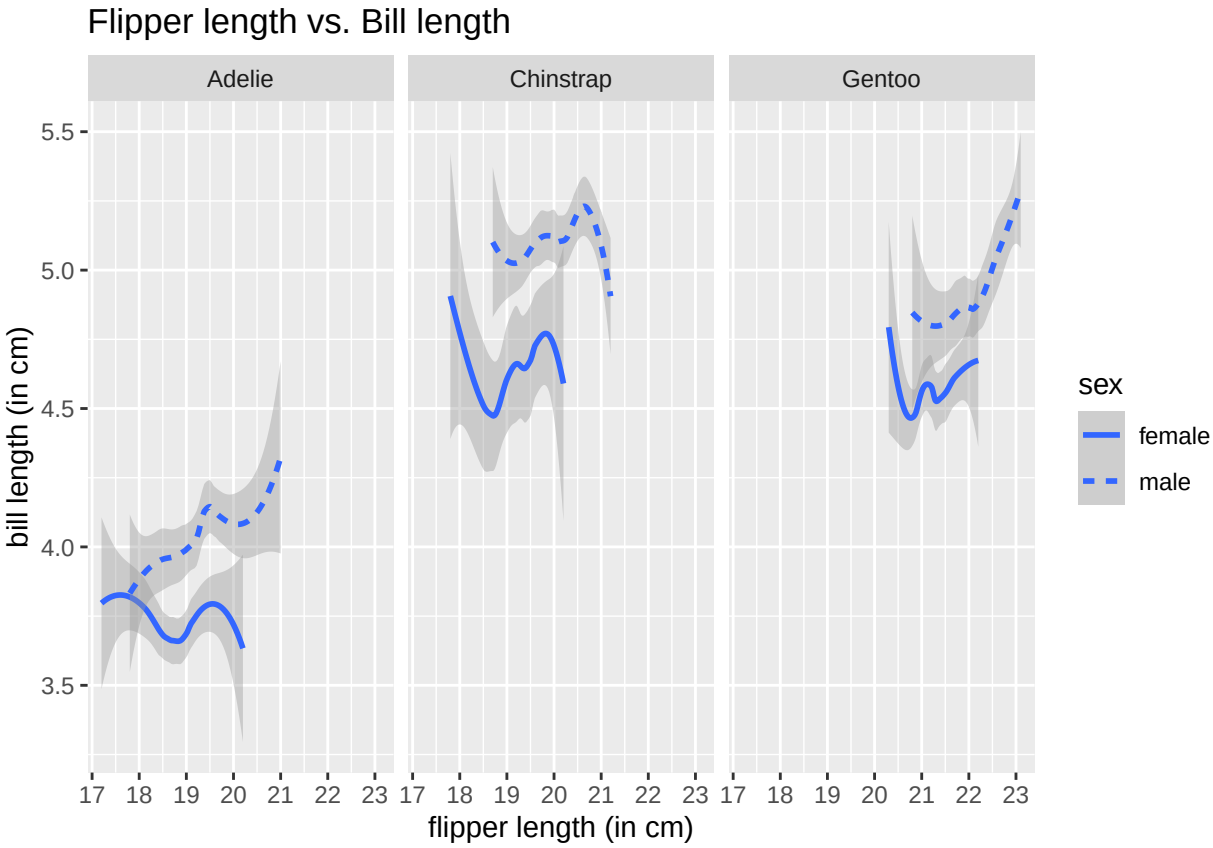
```
penguins_in_kg <- penguins %>%filter(!is.na(sex))%>%mutate(body_mass_kg = (body_mass_g/1000), flipper_l

ggplot(data = penguins_in_kg)+
  geom_smooth(mapping = aes(linetype = sex, x=flipper_length_cm,y=body_mass_kg))+
  facet_wrap(~species, nrow=1)+
  theme()+
  labs(title = "Flipper length vs. Body mass", y="body mass (in KG)", x="flipper length (in cm)")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
ggplot(data = penguins_in_kg)+
  geom_smooth(mapping = aes(linetype = sex, x=flipper_length_cm,y=bill_length_cm))+
  facet_wrap(~species, nrow=1)+
  theme()+
  labs(title = "Flipper length vs. Bill length", y="bill length (in cm)", x="flipper length (in cm)")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Question 2 A. Find out the most frequent destination airports for all domestic flights departing from New York City (in terms of total number of flights).

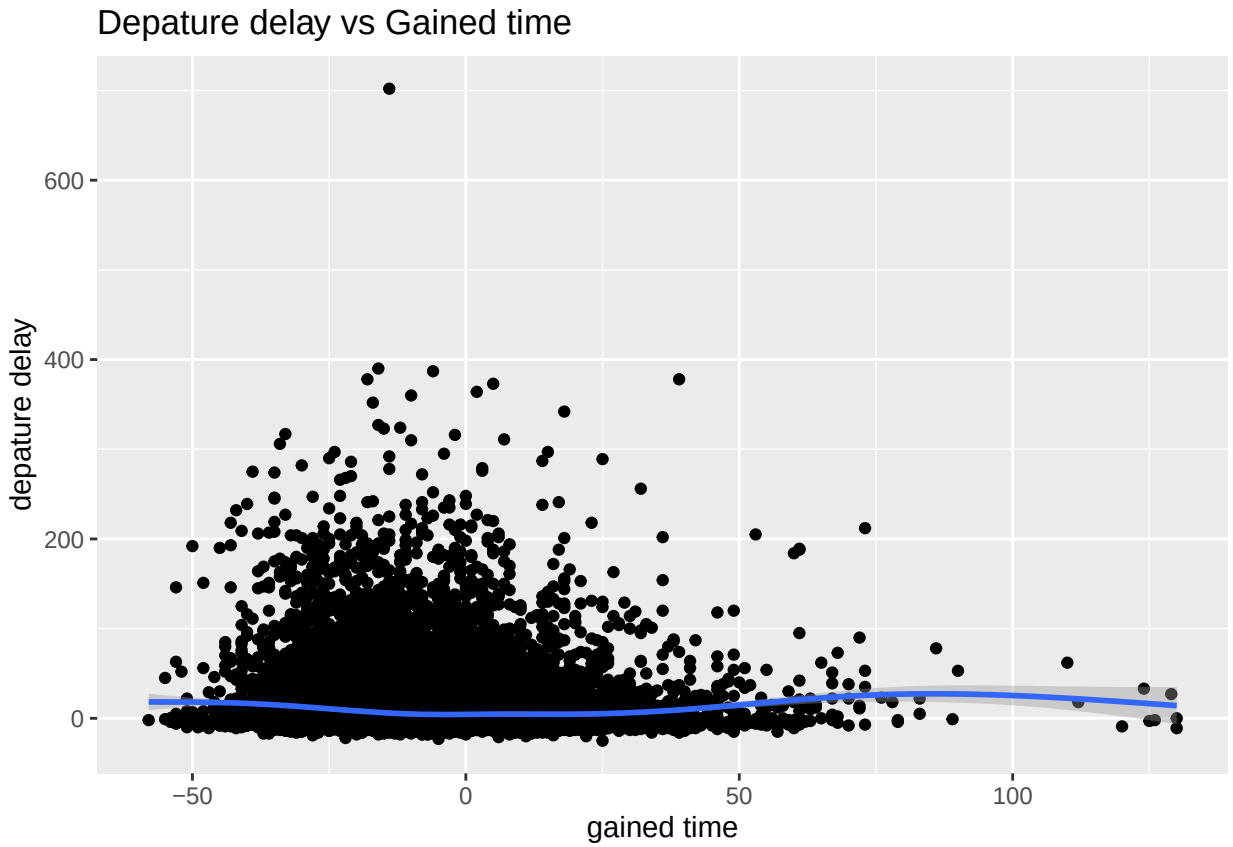
```
library(nycflights13)
flights %>% filter(!is.na(flight)) %>% group_by(dest) %>% summarize(n_flights = n_distinct(flight)) %>%
```

```
## # A tibble: 105 x 2
##    dest  n_flights
##    <chr>     <int>
##  1 IAH         665
##  2 ORD         617
##  3 BOS         469
##  4 CLT         449
##  5 DEN         435
##  6 SFO         432
##  7 LAX         399
##  8 MCO         332
##  9 FLL         313
## 10 CLE         311
## # ... with 95 more rows
```

B. Passengers are often frustrated when their flight departs late, but are not as annoyed if, in the end, pilots can make up some time during the flight. Extract the flights data in October and calculate the "gained time in the air" as the difference between arr_delay and dep_delay. Make a scatterplot of dep_delay vs. the "gained time" and add a smooth line.

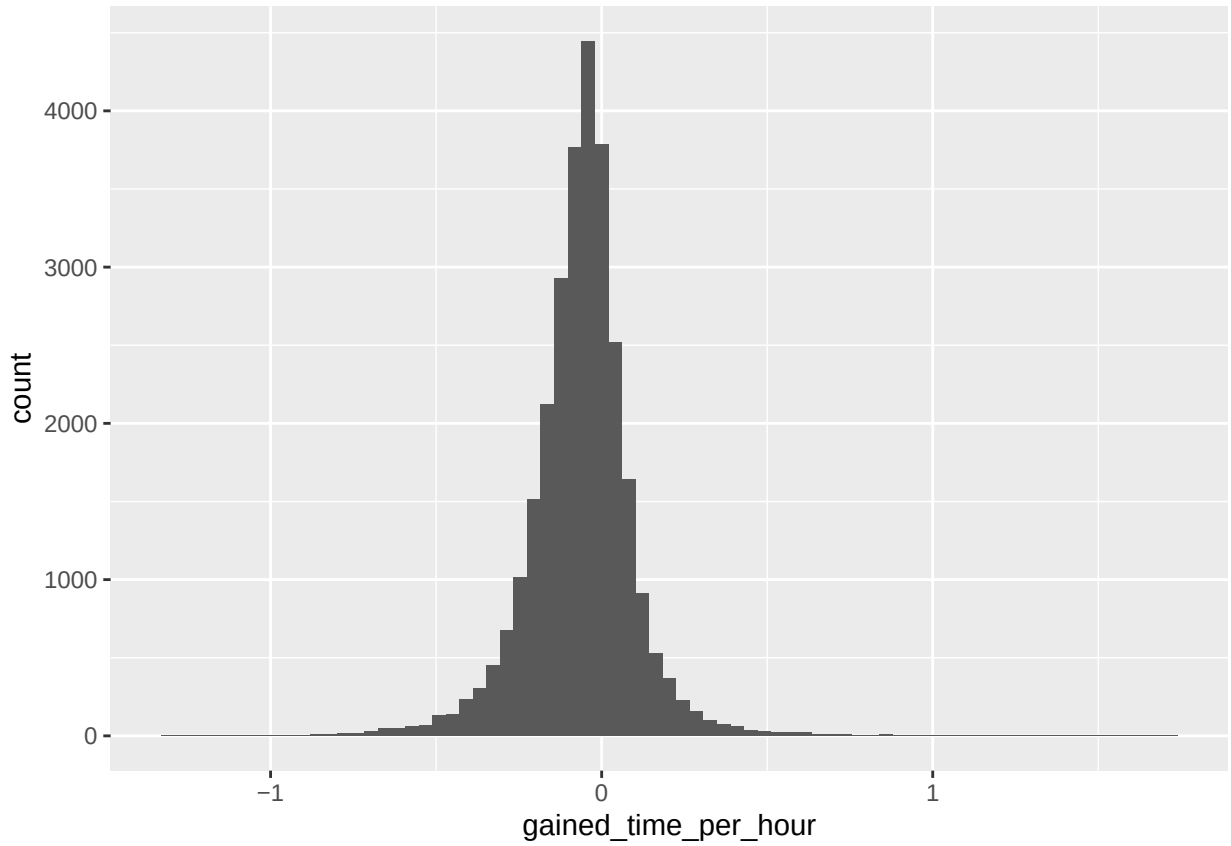
```
flights %>% filter(month == 10)%>%mutate(gained_time_in_air = arr_delay - dep_delay)%>%
  ggplot(aes(y=dep_delay, x=gained_time_in_air))+geom_point()+geom_smooth()+labs(title = "Depature delay
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
## Warning: Removed 271 rows containing non-finite values (`stat_smooth()`).
## Warning: Removed 271 rows containing missing values (`geom_point()`).
```



C. The "gained time" defined above may not be reasonable because long distance flight may be more likely to gain more. Let's define the "gained time per hour" and draw a histogram of this variable. What are the mean and median of this variable?

```
flights%>%filter(month == 10)%>%filter(!is.na(arr_delay) & !is.na(dep_delay))%>%mutate(gained_time_per_h
  ggplot(aes(x=gained_time_per_hour))+geom_histogram(bins = 75)
```




```
flights %>% filter(month == 10 & !is.na(arr_delay) & !is.na(dep_delay)) %>% mutate(gained_time_per_hour
```

```
## # A tibble: 1 x 2
##      mean  median
##     <dbl>   <dbl>
## 1 -0.0634 -0.0552
```

D. Plot the following bar chart on the number of flights took off every 3 hours. You need to 1. remove cancelled flights 2. create a variable which equals 0 if the flight took off between 0 and 3 am, equals 1 if took off between 3 and 6 am... 3. make the bar chart 4. change the labels

```
flights%>%filter(!is.na(dep_time),!is.na(arr_time))
```

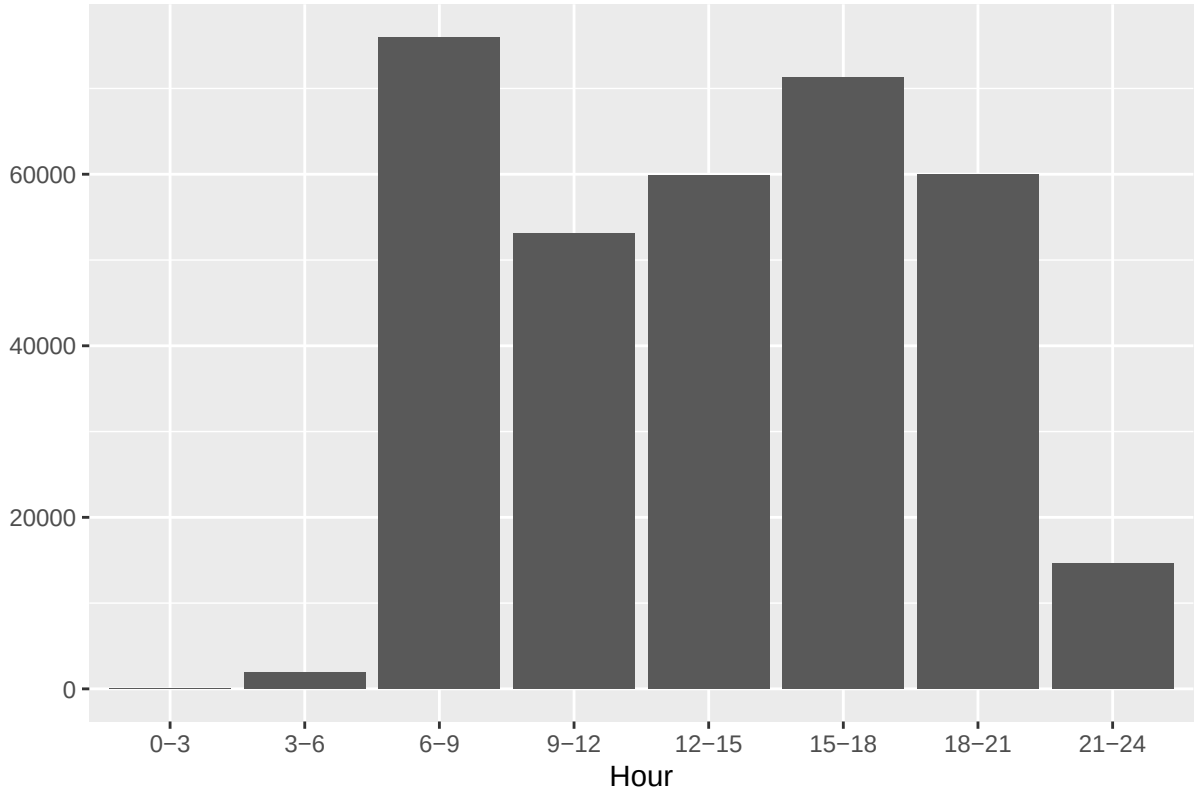
```
## # A tibble: 328,063 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     1      517        515       2     830     819      11 UA
##  2  2013     1     1      533        529       4     850     830      20 UA
##  3  2013     1     1      542        540       2     923     850      33 AA
##  4  2013     1     1      544        545      -1    1004    1022     -18 B6
##  5  2013     1     1      554        600      -6     812     837     -25 DL
##  6  2013     1     1      554        558      -4     740     728      12 UA
##  7  2013     1     1      555        600      -5     913     854      19 B6
##  8  2013     1     1      557        600      -3     709     723     -14 EV
##  9  2013     1     1      557        600      -3     838     846      -8 B6
## 10  2013     1     1      558        600      -2     753     745       8 AA
## # ... with 328,053 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
```

```
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

```
attach(flights)
flights$hourcat[hour >= 0 & hour <3 ] <- "0-3"
```

```
## Warning: Unknown or uninitialised column: `hourcat`.
```

```
flights$hourcat[hour >= 3 & hour <6] <- "3-6"
flights$hourcat[hour >= 6 & hour <9] <- "6-9"
flights$hourcat[hour >= 9 & hour <12] <- "9-12"
flights$hourcat[hour >= 12 & hour <15] <- "12-15"
flights$hourcat[hour >= 15 & hour <18] <- "15-18"
flights$hourcat[hour >= 18 & hour <21] <- "18-21"
flights$hourcat[hour >= 21] <- "21-24"
flights%>%mutate(hourcat = fct_relevel(hourcat, "0-3", "3-6", "6-9", "9-12", "12-15", "15-18", "18-21",
ggplot()+
  geom_bar(aes(x= hourcat))+
  labs(title = "", y="", x="Hour")
```



Question 3 A. Select the babies born in 2010 and used by no less than 10000 babies. (optional) Obtain the names that are used by no less than 10000 boys and 10000 girls.

```
library(babynames)
babynames%>%filter(year == 2010, n >= 10000)
```

```
## # A tibble: 47 x 5
##     year sex   name         n    prop
##    <dbl> <chr> <chr>    <int>   <dbl>
##  1  2010 F     Isabella 22905 0.0117
```

```
##  2  2010 F     Sophia   20639 0.0105
##  3  2010 F     Emma     17338 0.00885
##  4  2010 F     Olivia   17022 0.00869
##  5  2010 F     Ava      15429 0.00788
##  6  2010 F     Emily    14268 0.00729
##  7  2010 F     Abigail  14243 0.00727
##  8  2010 F     Madison  13176 0.00673
##  9  2010 F     Chloe    11750 0.00600
## 10  2010 F     Mia      10637 0.00543
## # ... with 37 more rows
```

```
babynames%>% group_by(sex)%>%
  filter(n>10000)%>%
  arrange(sex)
```

```
## # A tibble: 5,844 x 5
## # Groups:   sex [2]
##     year sex   name      n   prop
##    <dbl> <chr> <chr> <int>  <dbl>
##  1  1888 F     Mary  11754 0.0620
##  2  1889 F     Mary  11648 0.0616
##  3  1890 F     Mary  12078 0.0599
##  4  1891 F     Mary  11703 0.0595
##  5  1892 F     Mary  13172 0.0586
##  6  1893 F     Mary  12784 0.0568
##  7  1894 F     Mary  13151 0.0557
##  8  1895 F     Mary  13446 0.0544
##  9  1896 F     Mary  13811 0.0548
## 10  1897 F     Mary  13413 0.0540
## # ... with 5,834 more rows
```

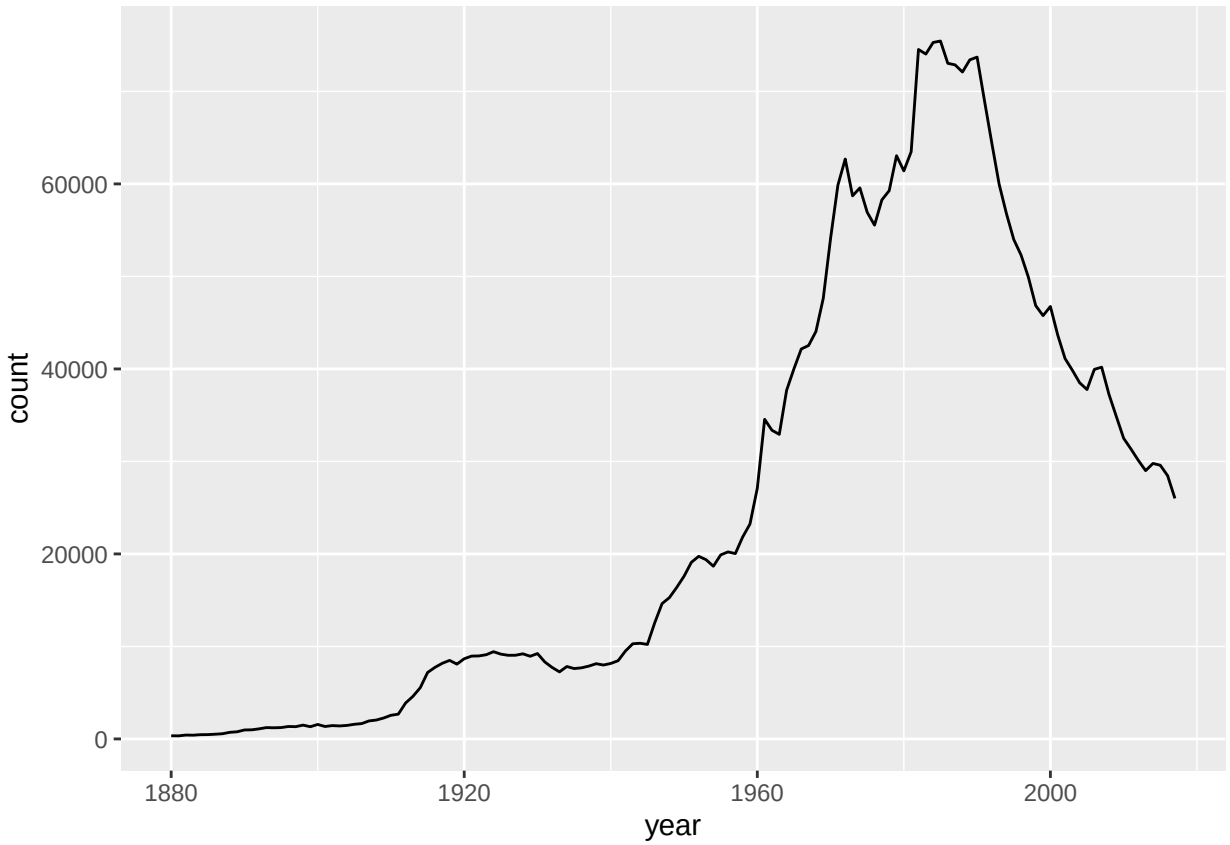
B. Find out the longest names (in terms of the number of letters) in 2010. (str_length() calculates the number of letters of a string).

```
babynames%>%filter(year == 2010)%>%mutate(letters_in_name = str_length(name))%>%arrange(desc(letters_in
```

```
## # A tibble: 34,067 x 6
##     year sex   name                n       prop letters_in_name
##    <dbl> <chr> <chr>           <int>      <dbl>           <int>
##  1  2010 M     Christianjoseph     7 0.00000341              15
##  2  2010 M     Christopherjame     5 0.00000244              15
##  3  2010 M     Jaydenalexander     5 0.00000244              15
##  4  2010 F     Mariaguadalupe     16 0.00000817              14
##  5  2010 F     Mariadelcarmen     10 0.00000511              14
##  6  2010 F     Sarahelizabeth      9 0.0000046               14
##  7  2010 M     Michaelanthony     15 0.00000731              14
##  8  2010 M     Alexanderjames      8 0.0000039               14
##  9  2010 M     Christianjames      7 0.00000341              14
## 10  2010 M     Oluwatimilehin      7 0.00000341              14
## # ... with 34,057 more rows
```

C. Make a time series plot (line plot) of the number of babies having a long name (>=10 letters). Below is an example.

```
babynames%>%mutate(letters_in_name = str_length(name))%>%group_by(year)%>%filter(letters_in_name >= 10)%
ggplot()+geom_line(aes(x=year,y=total_count))+
  labs(x="year",y="count")
```

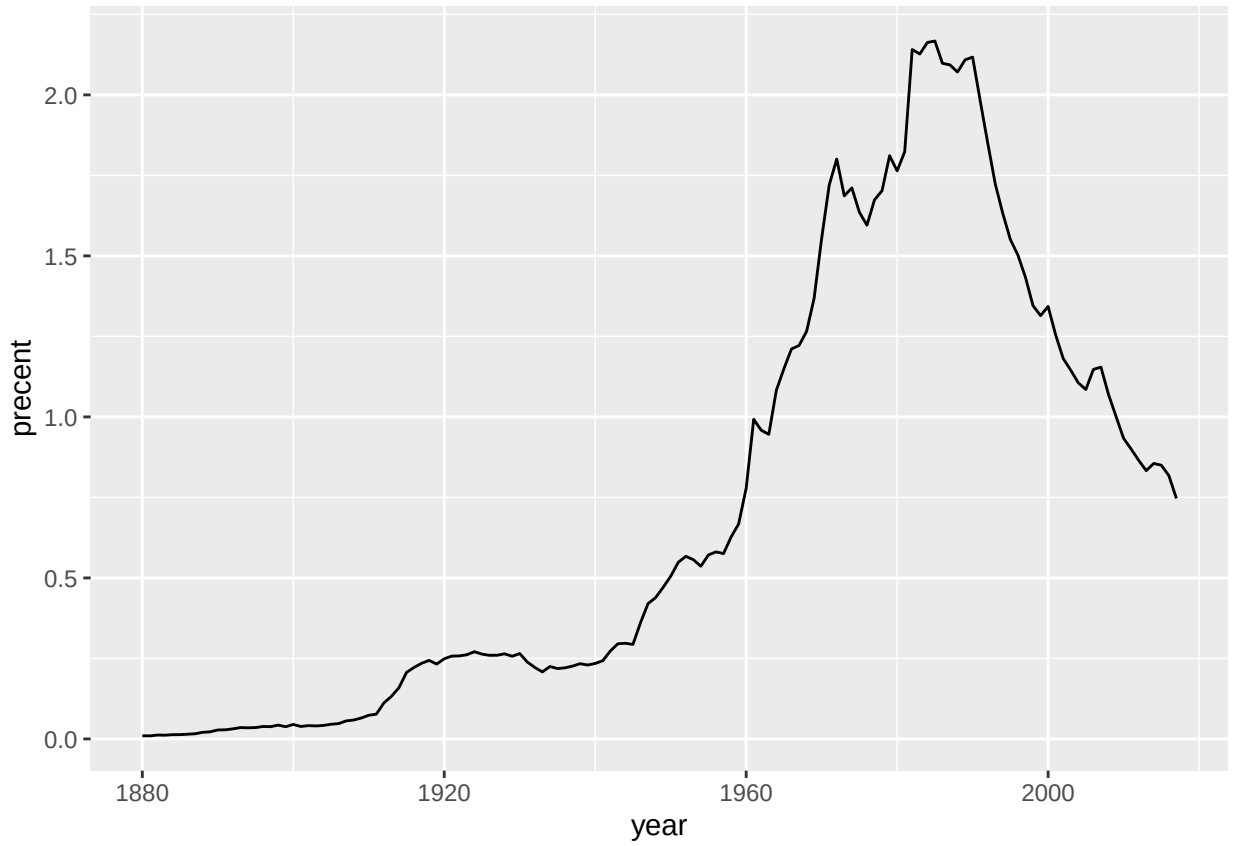


D. Use proportions (number of long name babies/ total number of babies in that year) instead of counts for the previous question.

```
babynames%>%
  summarize(n=sum(n))
```

```
## # A tibble: 1 x 1
##           n
##       <int>
## 1 348120517
```

```
babynames%>%
  mutate(num_lett_name = str_length(name))%>%
  filter(num_lett_name>=10)%>%
  group_by(year)%>%
  summarize(num_of_prop = (sum(n)/348120517)*100)%>%
  ggplot(aes(x=year,y=num_of_prop*100))+
  geom_line()+
  labs(x="year",y="precent")
```



Question 4

```
toy <- tibble(
  x = c(4, 2, 2, 3, 3, 1, 3, 4, 5),
  group = c("c", "b", "a", "c", "b", "a", "a", "b", "c")
)
```

A. Suppose we would like to sort x in an increasing order within each group like this Does the following code work? toy %>% group_by(group) %>% arrange(x) No, this code does not work as it sorts by the group first, rather than sort by x. This code now works: toy %>% group_by(group) %>% arrange(group,x)

```
toy %>%
  group_by(group) %>%
  arrange(group,x)
```

```
## # A tibble: 9 x 2
## # Groups:   group [3]
##       x group
##   <dbl> <chr>
## 1     1 a
## 2     2 a
## 3     3 a
## 4     2 b
## 5     3 b
## 6     4 b
## 7     3 c
## 8     4 c
## 9     5 c
```

B. Suppose we would like to subtract the within group mean from each x. For example, the first observation in toy is in group c, with x equal to 4. Then we want to have 4 minus the mean of the three values in group c, (4+3+5)/3. Does the following code work? Make it work using group_by(). toy %>% mutate(x_new = x - mean(x))

```
toy %>%
  group_by(group)%>%
  mutate(x_new = x - mean(x))%>%
  arrange(group,x)
```

```
## # A tibble: 9 x 3
## # Groups:   group [3]
##       x group x_new
##   <dbl> <chr> <dbl>
## 1     1 a        -1
## 2     2 a         0
## 3     3 a         1
## 4     2 b        -1
## 5     3 b         0
## 6     4 b         1
## 7     3 c        -1
## 8     4 c         0
## 9     5 c         1
```