```
install.packages("tidyverse")
install.packages("nycflights13")

    Installing package into '/usr/local/lib/R/site-library'
    (as 'lib' is unspecified)

    Installing package into '/usr/local/lib/R/site-library'
    (as 'lib' is unspecified)
```

```
library(tidyverse)
library(nycflights13)
options(repr.plot.width=5, repr.plot.height=4)

    Warning message in system("timedatectl", intern = TRUE):
    "running command 'timedatectl' had status 1"
    ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
    ✔ ggplot2 3.4.0      ✔ purrr   1.0.1
    ✔ tibble  3.1.8      ✔ dplyr   1.0.10
    ✔ tidyr   1.2.1      ✔ stringr 1.4.1
    ✔ readr   2.1.3      ✔ forcats 0.5.2
    ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
    ✖ dplyr::filter() masks stats::filter()
    ✖ dplyr::lag()    masks stats::lag()
```

STATS 306

Homework 2: Using `dplyr`

For each problem, enter the R code in the cell marked "YOUR SOLUTION HERE".

Problem 1: Naming frequency (4 points)

Problem 1 is based on the `babynames` data set. Use help(babynames) to learn more on this dataset

```
install.packages("babynames")
library(babynames)
summary(babynames)

    Installing package into '/usr/local/lib/R/site-library'
    (as 'lib' is unspecified)

          year          sex                name                 n
     Min.   :1880   Length:1924665     Length:1924665     Min.   :    5.0
     1st Qu.:1951   Class :character   Class :character   1st Qu.:    7.0
     Median :1985   Mode  :character   Mode  :character   Median :   12.0
     Mean   :1975                                         Mean   :  180.9
     3rd Qu.:2003                                         3rd Qu.:   32.0
     Max.   :2017                                         Max.   :99686.0
          prop
     Min.   :2.260e-06
     1st Qu.:3.870e-06
     Median :7.300e-06
     Mean   :1.363e-04
     3rd Qu.:2.288e-05
     Max.   :8.155e-02
```

```
help(babynames)
```

(a) What were the top five most popular names for boys and girls in 1925? *1 point*

```
# Your solution here

babynames %>%
  filter(year == 1925) %>%
  arrange(desc(n)) %>%
  group_by(sex) %>%
  top_n(5) %>%
  print
```

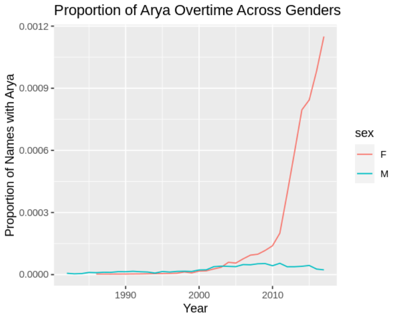
```
Selecting by prop
# A tibble: 10 × 5
# Groups:   sex [2]
    year sex   name          n   prop
   <dbl> <chr> <chr>     <int>  <dbl>
 1  1925 F     Mary      70597 0.0559
 2  1925 M     Robert    60896 0.0529
 3  1925 M     John      57197 0.0497
 4  1925 M     William   53303 0.0463
 5  1925 M     James     52681 0.0458
 6  1925 F     Dorothy   38570 0.0305
 7  1925 F     Betty     32813 0.0260
 8  1925 M     Charles   29581 0.0257
 9  1925 F     Helen     29170 0.0231
10  1925 F     Margaret  24464 0.0194
```

(b) Use `ggplot` to create a plot of the frequency of the name "Arya" over the years among boys and girls, respectively. Does anything noteworthy jump out at you from the plot? Can you explain why this happened? *1 point*

```
# Your solution here
justArya <- babynames %>% filter(name == "Arya")

ggplot(justArya, aes(x=year, y=prop, color =sex)) +
  geom_line()+
  labs(x='Year', y='Proportion of Names with Arya', title = "Proportion of Arya Overtime Across Genders")


#Until around 2005, the name "Arya" remains very unpopular among females and males
#with a proportion close to 0. However, after 2005, there is a huge spike in girls
#with the name Arya which is quite noteworthy. After doing some research online,
#it seems like this spike is contributed to the main character Arya in the
#popular TV show Game of Thorns.
```



(c) Define a name to be "timeless" if it was among the ten most popular names in both 2015 and 1915. How many timeless names are there, and what are they? *2 points*

```
# Your solution here
#FIX SEEMS WRONG
firstyear <- babynames %>%
  filter(year==1915) %>%
  arrange(desc(n))%>%
  head(10)


secondyear <- babynames %>%
  filter(year==2015) %>%
  arrange(desc(n))%>%
  head(10)


intersect(firstyear$name, secondyear$name)
```

```
'William'
```

```
#There is only one timeless name "William" which was among the ten
#most popular names in both 2015 and 1915.
```

Problem 2: Manipulating `flights` (4 points)

This problem contines with the `flights` table that we saw this week in lecture. Recall that we first need to load this database by typing:

```
library(nycflights13)
```

(If you are running on Google Colab, you will also need to install this package each time you start the notebook.)

(a) Use `filter()` to find all the flights that had an arrival delay of more than two hours. How many of these flights were there? *1/2 point*

```
# Your solution here
flights %>%
  filter(arr_delay > 120) %>%
  nrow()

#There are 10,034 flights that arrived more than 2 hours delayed.
```

```
10034
```

(b) Was there a flight scheduled on every day of 2013? If so, write code that verifies this. If not, write code that shows which days had no scheduled flights. *1/2 point*

```
# Your solution here

flights %>%
  filter(year == 2013) %>%
  group_by(year, month, day) %>%
  summarise(
    n = n()
  ) %>%
  nrow()

#Yes, since there are 365 groups, there was a flight for each day of 2013
#as each year has 365 days,
```

```
`summarise()` has grouped output by 'year', 'month'. You can override using the
`.groups` argument.
365
```

(c) Say you want to maximize your chance of taking a flight that leaves on time (or early). Which airport and carrier should you choose? (For example, "UA departing out of EWR"). Support your reasoning with code. *1 point*

```
# Your solution here

flights %>%
  filter_at(vars(dep_delay), all_vars(!is.na(.))) %>%
  mutate(
    onTime= case_when(dep_delay<=0 ~ 1, dep_delay>0 ~ 0)
  ) %>%
  group_by(origin) %>%
  mutate(
    onTimeOriginPROP = mean(onTime)
  )  %>%
  ungroup() %>%
  group_by(carrier) %>%
  mutate(
    onTimecarrierPROP = mean(onTime)
  ) %>%
  ungroup() %>%
  group_by(origin, carrier) %>%
  arrange(desc(onTimeOriginPROP), desc(onTimecarrierPROP)) %>%
  select(origin, carrier, onTimeOriginPROP, onTimecarrierPROP) %>%
  print

#To maximize your chance of taking a flight that leaves on time,
#you should take a flight out of LGA with US as LGA and US have
#the highest on time proportion from their respective airport
#and carrier.
```

```
# A tibble: 328,521 × 4
# Groups:   origin, carrier [35]
   origin carrier onTimeOriginPROP onTimecarrierPROP
   <chr>  <chr>              <dbl>             <dbl>
```

```
 1 LGA    US                   0.668             0.760
 2 LGA    US                   0.668             0.760
 3 LGA    US                   0.668             0.760
 4 LGA    US                   0.668             0.760
 5 LGA    US                   0.668             0.760
 6 LGA    US                   0.668             0.760
 7 LGA    US                   0.668             0.760
 8 LGA    US                   0.668             0.760
 9 LGA    US                   0.668             0.760
10 LGA    US                   0.668             0.760
# … with 328,511 more rows
```

(d) What time of day should you fly if you want to avoid delays as much as possible? *2 points* (This question is intentionally open-ended. There is no one correct answer. Use the data and the commands we have learned to argue your case.)

```
help(flights)
```

```
# Your solution here
flights %>%
  group_by(sched_dep_time) %>%
  summarise(
    mean_delay = mean(dep_delay, na.rm=T)
  )  %>%
  arrange(mean_delay) %>%
  head(5) %>%
  print
```

```
# A tibble: 5 × 2
  sched_dep_time mean_delay
           <int>      <dbl>
1           2201      -7
2            516      -5
3           2158      -4
4            913      -3.83
5            931      -3.58
```

```
flights %>%
  group_by(sched_arr_time) %>%
  summarise(
    mean_delay = mean(arr_delay, na.rm=T)
  )  %>%
  arrange(mean_delay) %>%
  head(5) %>%
  print
```

```
# A tibble: 5 × 2
  sched_arr_time mean_delay
           <int>      <dbl>
1            139      -39
2            103      -35
3            125      -28
4            136      -28
5            733      -22.3
```

```
#To avoid delays as much as possible, you should book a flight
#that departs around 22:01 (10:01 PM) and lands around 1:39 AM.
```

Problem 3: Challenge problem (3 points)

Define a flight to be *spooky* if it was in transit at 13:13h (i.e. 1:13pm) on Friday the 13th of any month. You should assume that a flight is in transit between its `dep_time` and its `arr_time`. How many spooky flights are there in the dataset?

```
library(lubridate)
```

```
# Your solution here
#see if 1:13 is between dep_time and arr_time. arr_time - 1:13 positive. dep_time -1:13 negative

flights %>%
  mutate(
    dayofweek = wday(time_hour, week_start=1)
  ) %>%
  filter(day ==13 & dayofweek ==5) %>%
  mutate(
  mutate(
    spooky = case_when(arr_time-1313>=0 & dep_time-1313<=0 ~ T)
  ) %>%
  filter(spooky == T) %>%
  nrow()



  #There are 236 spooky flights in the dataset.
```

```
236
```