Lab 5

Sydney Stitt

Math 241, Week 6

```
# Put all necessary libraries here
library(tidyverse)
library(rnoaa)
library(rvest)
library(httr)
library(dplyr)
library(ggplot2)
library(knitr)
```

Due: Friday, March 1st at 8:30am

Goals of this lab

1. Practice grabbing data from the internet.
2. Learn to navigate new R packages.
3. Grab data from an API (either directly or using an API wrapper).
4. Scrape data from the web.

Potential API Wrapper Packages

Problem 1: Predicting the ~~Un~~predictable: Portland Weather

In this problem let's get comfortable with extracting data from the National Oceanic and Atmospheric Administration's (NOAA) API via the R API wrapper package `rnoaa`.

You can find more information about the datasets and variables here.

```
# Don't forget to install it first!
library(rnoaa)
```

a. First things first, go to this NOAA website to get a key emailed to you. Then insert your key below:

```
options(noaakey = "olfVllAdvgxTnItbErwMcSJwVUCKtoVh")
```

b. From the National Climate Data Center (NCDC) data, use the following code to grab the stations in Multnomah County. How many stations are in Multnomah County?

```
stations <- ncdc_stations(datasetid = "GHCND",
                          locationid = "FIPS:41051")

mult_stations <- stations$data
```

There are 25 stations in Multonomah County

c. January was not so rainy this year, was it? Let's grab the precipitation data for site `GHCND:US1ORMT0006` for this past January.

```
#Grabbing data
ncdc_datatypes(datasetid = "GHCND",
               stationid = "GHCND:US1ORMT0006")
```

```
## $meta
##   offset count limit
## 1      1     5    25
##
## $data
##      mindate    maxdate                                     name datacoverage
## 1 1750-02-01 2024-02-27                            Precipitation            1
## 2 1840-05-01 2024-02-27                                Snowfall            1
## 3 1857-01-18 2024-02-27                               Snow depth            1
## 4 1952-07-01 2024-02-27 Water equivalent of snow on the ground            1
## 5 1998-06-01 2024-02-27           Water equivalent of snowfall            1
##     id
## 1 PRCP
## 2 SNOW
## 3 SNWD
## 4 WESD
## 5 WESF
##
## attr(,"class")
## [1] "ncdc_datatypes"
```

```
precip_se_pdx <- ncdc(datasetid = "GHCND",
                      stationid = "GHCND:US1ORMT0006",
                      datatypeid = "PRCP",
                      startdate = "2023-01-01",
                      enddate = "2023-01-31")
precip_se_pdx
```

```
## $meta
## $meta$totalCount
## [1] 31
##
## $meta$pageCount
## [1] 25
##
## $meta$offset
## [1] 1
##
```

```
##
## $data
## # A tibble: 25 x 8
##    date                datatype station           value fl_m  fl_q  fl_so fl_t
##    <chr>               <chr>    <chr>             <int> <chr> <chr> <chr> <chr>
##  1 2023-01-01T00:00:00 PRCP     GHCND:US1ORMT0006     8 ""    ""    N     0830
##  2 2023-01-02T00:00:00 PRCP     GHCND:US1ORMT0006     0 ""    ""    N     0700
##  3 2023-01-03T00:00:00 PRCP     GHCND:US1ORMT0006    43 ""    ""    N     0826
##  4 2023-01-04T00:00:00 PRCP     GHCND:US1ORMT0006    20 ""    ""    N     0826
##  5 2023-01-05T00:00:00 PRCP     GHCND:US1ORMT0006    46 ""    ""    N     0822
##  6 2023-01-06T00:00:00 PRCP     GHCND:US1ORMT0006    46 ""    ""    N     0822
##  7 2023-01-07T00:00:00 PRCP     GHCND:US1ORMT0006    25 ""    ""    N     0822
##  8 2023-01-08T00:00:00 PRCP     GHCND:US1ORMT0006    99 ""    ""    N     0822
##  9 2023-01-09T00:00:00 PRCP     GHCND:US1ORMT0006   114 ""    ""    N     0806
## 10 2023-01-10T00:00:00 PRCP     GHCND:US1ORMT0006    33 ""    ""    N     0806
## # i 15 more rows
##
## attr(,"class")
## [1] "ncdc_data"
```

d. What is the class of `precip_se_pdx`? Grab the data frame nested in `precip_se_pdx` and call it `precip_se_pdx_data`. Precip_se_pdx is a character list

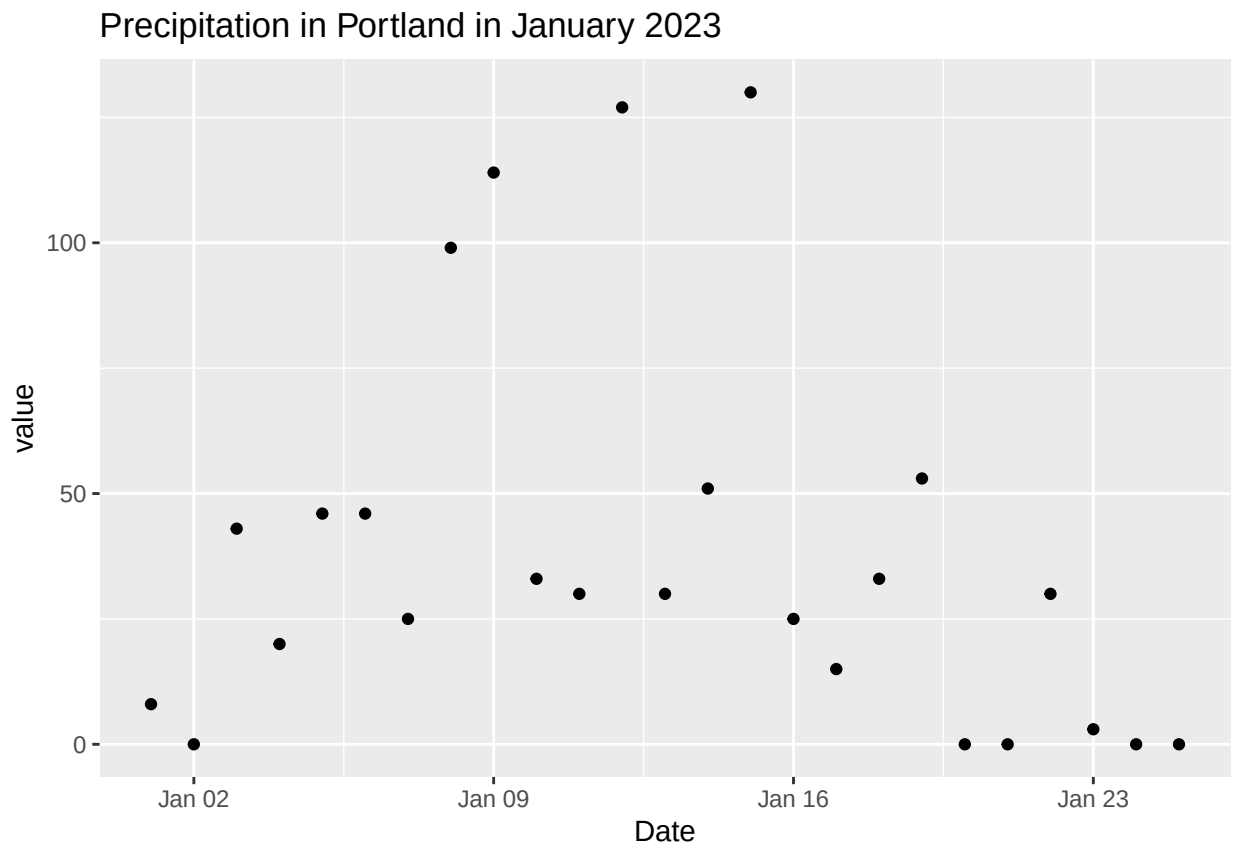
```r
#Grabbing df nested in precip_se_pdx
precip_se_pdx_data <- precip_se_pdx[[2]]
```

e. Use `ymd_hms()` in the package `lubridate` to wrangle the date column into the correct format.

```r
#Fixing the date column with lubridate
precip_se_pdx_data <- precip_se_pdx_data %>%
  mutate(fixeddate = ymd_hms(paste(date)))
```

f. Plot the precipitation data for this site in Portland over time. Rumor has it that we had only one day where it didn't rain. Is that true?

```r
#Graphing precipitation data for Jan 2023
ggplot(precip_se_pdx_data, aes(x = fixeddate, y = value)) +
  geom_point() +
  labs(title = "Precipitation in Portland in January 2023", x = "Date")
```



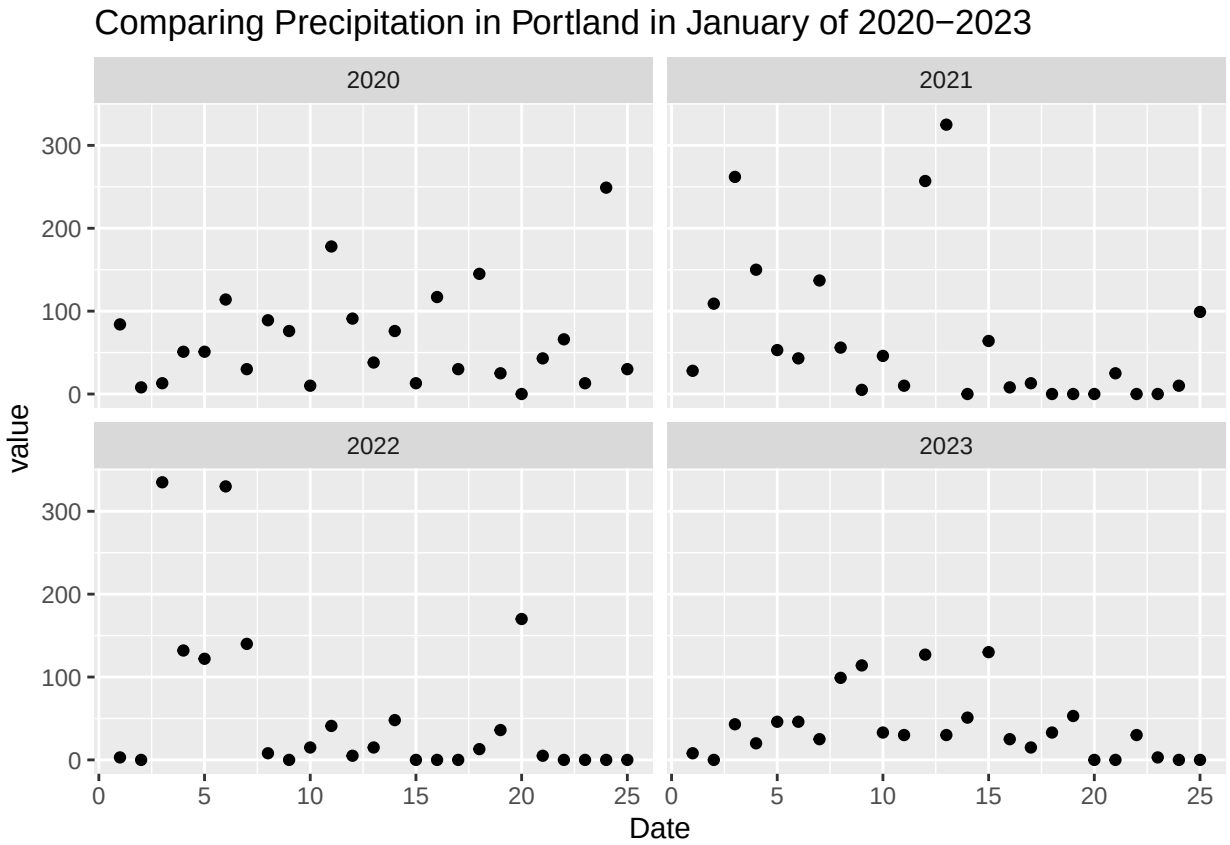
That is not true – there were five days that there was no rain in Portland.

g. (Bonus) Adapt the code to create a visualization that compares the precipitation data for January over the the last four years. Do you notice any trend over time?

```
#Compiling data for precip for Januarys from 2020 to 2023
precip_se_pdx_2020 <- ncdc(datasetid = "GHCND",
                      stationid = "GHCND:US1ORMT0006",
                      datatypeid = "PRCP",
                      startdate = "2020-01-01",
                      enddate = "2020-01-31")
precip_se_pdx_2020_data <- precip_se_pdx_2020[[2]]
precip_se_pdx_2020_data <- precip_se_pdx_2020_data %>%
  mutate(fixeddate = ymd_hms(paste(date)))
precip_se_pdx_2021 <- ncdc(datasetid = "GHCND",
                      stationid = "GHCND:US1ORMT0006",
                      datatypeid = "PRCP",
                      startdate = "2021-01-01",
                      enddate = "2021-01-31")
precip_se_pdx_2021_data <- precip_se_pdx_2021[[2]]
precip_se_pdx_2021_data <- precip_se_pdx_2021_data %>%
  mutate(fixeddate = ymd_hms(paste(date)))
precip_se_pdx_2022 <- ncdc(datasetid = "GHCND",
                      stationid = "GHCND:US1ORMT0006",
                      datatypeid = "PRCP",
                      startdate = "2022-01-01",
```

```
                      enddate = "2022-01-31")
precip_se_pdx_2022_data <- precip_se_pdx_2022[[2]]
precip_se_pdx_2022_data <- precip_se_pdx_2022_data %>%
  mutate(fixeddate = ymd_hms(paste(date)))
precip_se_pdx_20_23 <- full_join(precip_se_pdx_2020_data, precip_se_pdx_2021_data)
precip_se_pdx_20_23 <- full_join(precip_se_pdx_20_23, precip_se_pdx_2022_data)
precip_se_pdx_20_23 <- full_join(precip_se_pdx_20_23, precip_se_pdx_data)
precip_se_pdx_20_23 <- precip_se_pdx_20_23 %>%
  mutate(year = year(paste(fixeddate))) %>%
  mutate(mday = mday(paste(fixeddate)))

#Graphing Januarys precip data
ggplot(precip_se_pdx_20_23, aes(x = mday, y = value)) +
  geom_point() +
  facet_wrap(~year) +
  labs(title = "Comparing Precipitation in Portland in January of 2020-2023", x = "Date")
```



Precipitation trends in January have become less variable and more consistent in the past four years.

Problem 2: From API to R

For this problem I want you to grab web data by either talking to an API directly with `httr` or using an API wrapper. It must be an API that we have NOT used in class or in Problem 1.

Once you have grabbed the data, do any necessary wrangling to graph it and/or produce some summary statistics. Draw some conclusions from your graph and summary statistics.

API Wrapper Suggestions for Problem 2

Here are some potential API wrapper packages. Feel free to use one not included in this list for Problem 2.

- `gtrendsR`: "An interface for retrieving and displaying the information returned online by Google Trends is provided. Trends (number of hits) over the time as well as geographic representation of the results can be displayed."
- `rfishbase`: For the fish lovers
- `darksky`: For global historical and current weather conditions

```
#Setting Up spotifyr API wrapper package
install.packages("spotifyr", repos = "http://cran.us.r-project.org")
```

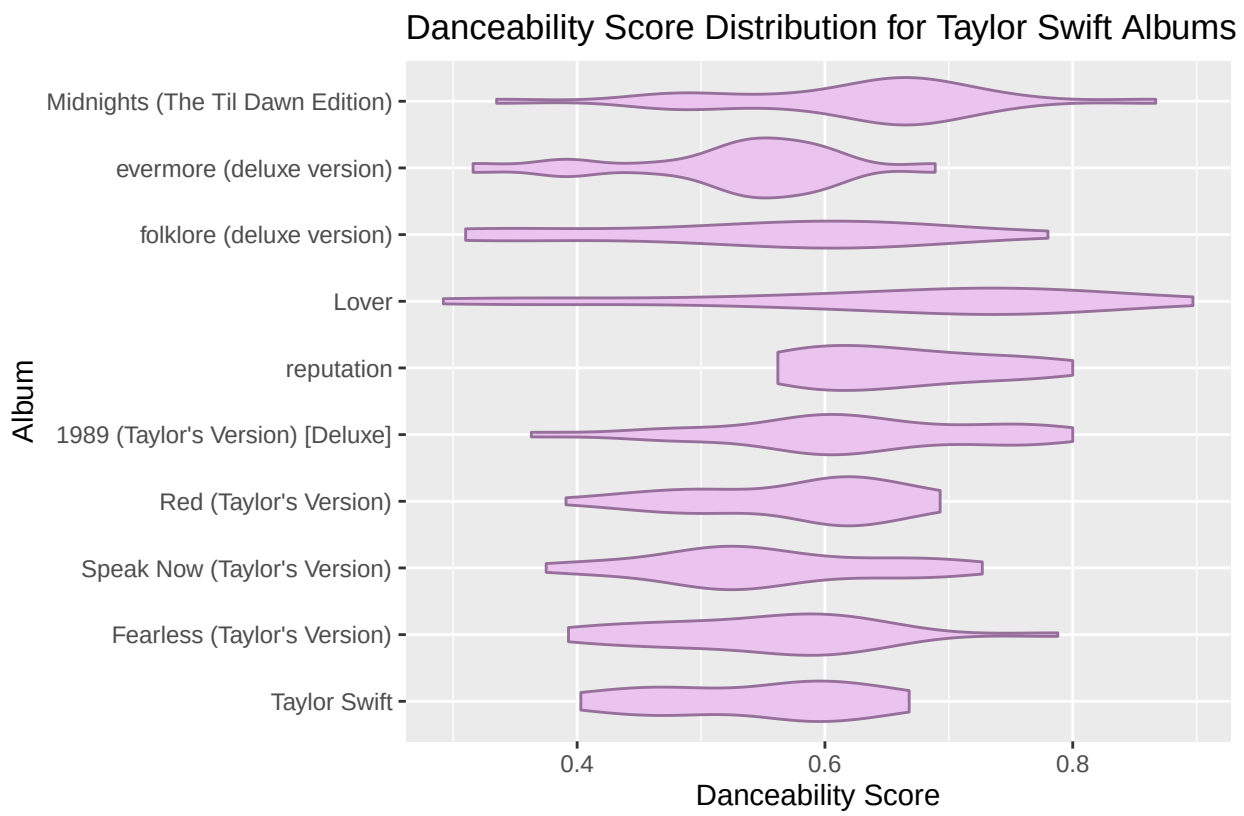
```
##
## The downloaded binary packages are in
##  /var/folders/2c/4fq7fr5x2092jngyt_mx_4nh0000gn/T//RtmpvQv1Um/downloaded_packages
```

```
library(spotifyr)
id <- "8f3663c9914e495c8bf3a9260fb29a49"
secret <- "1919190aa1c74e6cbb09245c1d19741a"
Sys.setenv(SPOTIFY_CLIENT_ID = id)
Sys.setenv(SPOTIFY_CLIENT_SECRET = secret)
access_token <- get_spotify_access_token()
```

```
#Wrangling data for Taylor Swift's discography
tswift <- get_artist_audio_features("taylor swift")
tswift_wrangled <- tswift %>%
  filter(album_name == "Speak Now (Taylor's Version)" |
         album_name == "1989 (Taylor's Version) [Deluxe]" |
         album_name == "Midnights (The Til Dawn Edition)" |
         album_name == "Red (Taylor's Version)" |
         album_name == "Fearless (Taylor's Version)" |
         album_name == "evermore (deluxe version)" |
         album_name == "folklore (deluxe version)" |
         album_name == "Lover" |
         album_name == "reputation" |
         album_name == "Taylor Swift") %>%
  group_by(album_name) %>%
  mutate(avg_dance = mean(danceability))
```

```
#Graphing Taylor Swift's data
ggplot(tswift_wrangled, aes(x = danceability, y = album_name)) +
  geom_violin(fill = "#EAC3EE",
              color = "#96709A") +
  scale_y_discrete(limits=c("Taylor Swift",
                            "Fearless (Taylor's Version)",
                            "Speak Now (Taylor's Version)",
                            "Red (Taylor's Version)",
                            "1989 (Taylor's Version) [Deluxe]",
                            "reputation",
                            "Lover",
                            "folklore (deluxe version)",
```

```
                            "evermore (deluxe version)",
                            "Midnights (The Til Dawn Edition)")) +
  labs(x = "Danceability Score",
       y = "Album",
       title = "Danceability Score Distribution for Taylor Swift Albums",
       caption = "Danceability score index developed by Spotify. Data retrieved using spotifyr API wrapp
```



Taylor Swift's most danceable songs, according to Spotify's danceability score index, are on her albums "Lover" and "Midnights". Both albums also have the largest range of danceability socre. The most consistently danceable album is "reputation". In general, Taylor Swift's albums have a great range of danceability, and are not very homogenous. Her newer albums ("Lover", "folklore", "evermore", "Midnights") tend to contain songs that are very low in danceability score, showing that her music writing is becoming more variable, including less danceable songs, over time.

Problem 3: Scraping Reedie Data

Let's see what lovely data we can pull from Reed's own website.

a. Go to https://www.reed.edu/ir/success.html and scrape the two tables.

```
#Grabbing data from Reed's website
reed_url <- "https://www.reed.edu/ir/success.html"
tables <- reed_url %>%
  read_html() %>%
  html_nodes(css = "table")
```

b. Grab and print out the table that is entitled "GRADUATE SCHOOLS MOST FREQUENTLY ATTENDED BY REED ALUMNI". Why is this data frame not in a tidy format?

```
#Grabbing the table from Reed's website data
grad_school_table <- html_table(tables[[2]], fill = TRUE)
print(grad_school_table)
```

```
## # A tibble: 11 x 4
##    MBAs                JDs                          PhDs                      MDs
##    <chr>               <chr>                        <chr>                     <chr>
##  1 U. of Chicago       Lewis & Clark  Law School   U.C., Berkeley            Oregon~
##  2 Portland State U.   U.C., Berkeley              U. of Washington          U. of ~
##  3 Harvard U.          U. of Oregon                 U. of Chicago             Washin~
##  4 U. of Washington    U. of Washington             Stanford U.               UC., S~
##  5 Columbia U.         New York U.                  U. of Oregon              Stanfo~
##  6 U of Pennsylvania.  U. of Chicago                Harvard U.                Harvar~
##  7 Stanford U.         Yale U.                      Cornell U.                Case W~
##  8 Yale U.             Harvard U.                   Columbia U.               Cornel~
##  9 U.C., Berkeley      U.C. Hastings Law School     U.C., Los Angeles         Johns ~
## 10 U. of Oregon        Cornell U.                   Yale U.                   U. of ~
## 11 UC., Los Angeles.   Georgetown U.                U. of Wisconsin, Madison U. of ~
```

This dataframe is not tidy because of the titles of the columns should be a variable itself (type of degree obtained)

c. Wrangle the data into a tidy format. Glimpse the resulting data frame.

```
#Tidying up Reed's table data
grad_school_table <- grad_school_table %>%
  pivot_longer(c("MBAs", "JDs", "PhDs", "MDs"),
               names_to = "Types of Degree",
               values_to = "Universities")

glimpse(grad_school_table)
```

```
## Rows: 44
## Columns: 2
## $ `Types of Degree` <chr> "MBAs", "JDs", "PhDs", "MDs", "MBAs", "JDs", "PhDs",~
## $ Universities      <chr> "U. of Chicago", "Lewis & Clark  Law School", "U.C.,~
```

d. Now grab the "OCCUPATIONAL DISTRIBUTION OF ALUMNI" table and turn it into an appropriate graph. What conclusions can we draw from the graph?

```
#Tidying up occupations table data
occupations_table <- html_table(tables[[1]], fill = TRUE)
print(occupations_table)
```

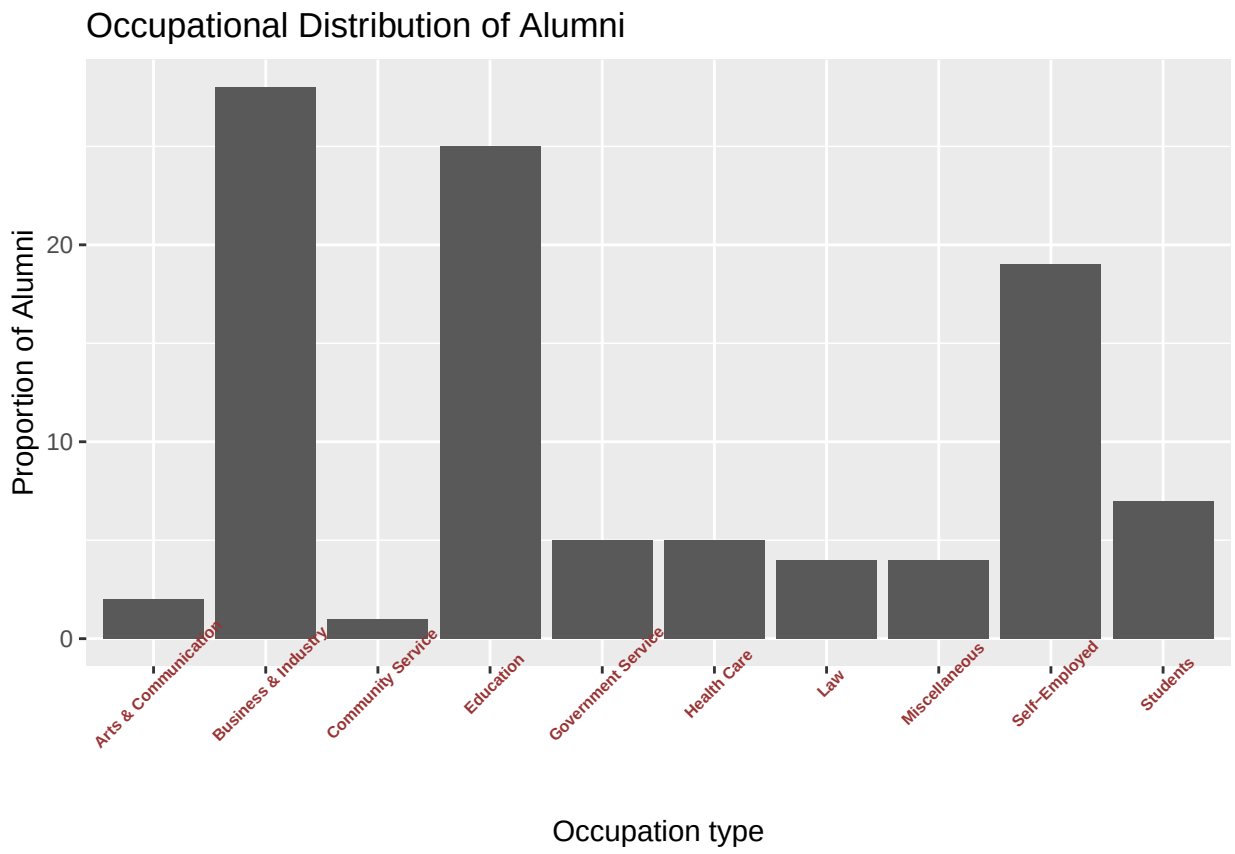
```
## # A tibble: 10 x 2
##    X1                  X2
##    <chr>               <chr>
##  1 Business & Industry  28%
```

```
##  2 Education            25%
##  3 Self-Employed        19%
##  4 Students             7%
##  5 Government Service   5%
##  6 Health Care          5%
##  7 Law                  4%
##  8 Miscellaneous        4%
##  9 Arts & Communication 2%
## 10 Community Service    1%
```

```
occupations_table <- occupations_table %>%
  mutate(X2 = parse_number(X2))
print(occupations_table)
```

```
## # A tibble: 10 x 2
##    X1                     X2
##    <chr>               <dbl>
##  1 Business & Industry    28
##  2 Education              25
##  3 Self-Employed          19
##  4 Students                7
##  5 Government Service      5
##  6 Health Care             5
##  7 Law                     4
##  8 Miscellaneous           4
##  9 Arts & Communication    2
## 10 Community Service       1
```

```
#Graphing occupations table data
ggplot(occupations_table, aes(x = X1, y = X2)) +
  geom_col() +
  theme(axis.text.x = element_text(face = "bold", color = "#993333",
                           size = 6, angle = 45)) +
  labs(x = "Occupation type", y = "Proportion of Alumni", title = "Occupational Distribution of Alumni"
```



We can see that the majority of students go on to work in the business/industry and education work fields. We can also conclude that a large proportion of students are also self-employed.

e. Let's now grab the Reed graduation rates over time. Grab the data from here.

Do the following to clean up the data:

- Rename the column names.

```
#Tidying up Reed grad rate table
reed_grad_url <- "https://www.reed.edu/ir/gradrateshist.html"
grad_rate_data <- reed_grad_url %>%
  read_html() %>%
  html_nodes(css = "table")
grad_rate_table <- html_table(grad_rate_data[[1]], fill = TRUE)
colnames(grad_rate_table) <- c("Entering_Class_Year", "Cohort_Size", "4", "5", "6")
```

- Remove any extraneous rows.

```
#Tidying up Reed grad rate table
grad_rate_table <- grad_rate_table %>%
  filter(!row_number() %in% c(1))
```

- Reshape the data so that there are columns for

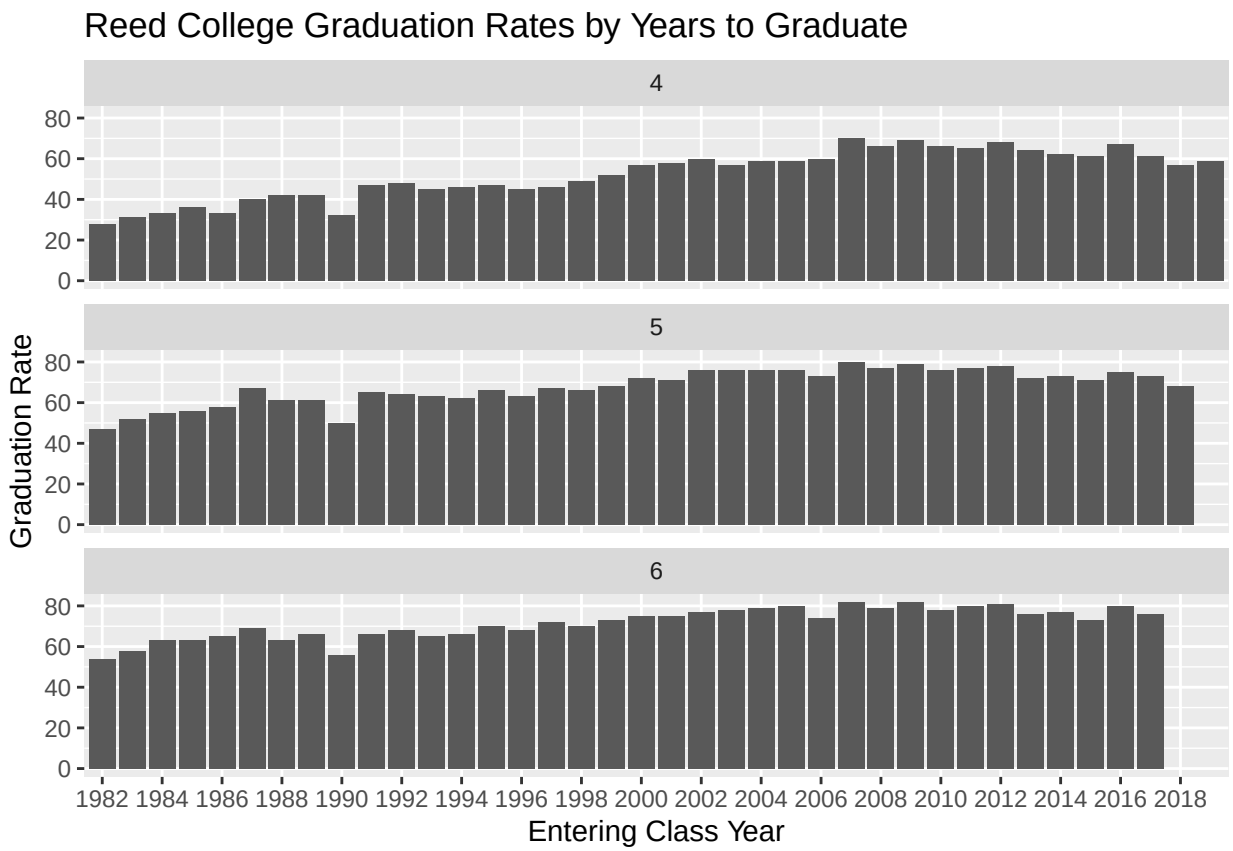
  - Entering class year
  - Cohort size
  - Years to graduation
  - Graduation rate

```
#Pivoting data in Reed grad rate table
grad_rate_table <- grad_rate_table %>%
  pivot_longer(c("4", "5", "6"),
               names_to = "Years_to_Graduation",
               values_to = "Graduation_Rate") %>%
  mutate(Graduation_Rate = parse_number(Graduation_Rate)) %>%
  filter(!is.na(Graduation_Rate))
```

- Make sure each column has the correct class.

f. Create a graph comparing the graduation rates over time and draw some conclusions.

```
#Graphing Reed grad rates over time
ggplot(grad_rate_table, aes(x = Entering_Class_Year, y = Graduation_Rate)) +
  geom_col() +
  facet_wrap(~Years_to_Graduation, ncol = 1) +
  scale_x_discrete(breaks=c(1982, 1984, 1986, 1988, 1990, 1992, 1994, 1996, 1998, 2000, 2002, 2004, 2006
  labs(title = "Reed College Graduation Rates by Years to Graduate",
       x = "Entering Class Year",
       y = "Graduation Rate")
```



Graduation rates for students graduating in 4, 5 and 6 years have generally increased over time since 1982. In general, the graduation rate for students graduating in 5 and 6 years is higher than those graduating in 4 years. The graduation rate for students graduating in 4 years has seen the most significant increase. Thus, there may be factors that contribute to the increased graduation rates like increased student support services.