Problem Set 8

name?!

date?!

Due date: Monday, November 2nd

From Problem Set #7, Part 2

If you previously completed this part please copy/paste your code below.

Question 7 / Challenge

Use the readxl library and load two data sets from the "two_data_sheets" file. There's a parameter that you can specify which sheet to load with read_excel(). In this case, we have data about rat reaction time in sheet 1 and home visits in sheet 2.

```r
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(readxl)
df_rats <- read_excel("../data/two_data_sheets.xlsx", 1)
df_home <- read_excel("../data/two_data_sheets.xlsx", 2)
```

Question 8

For the rats data, pivot the data frame from wide to long format. We want the 1, 2, 3 columns, which represent the amount of cheese placed in a maze, to transform into a column called "cheese". The values in the cheese column will be the time, which represents the amount of time the rat took to complete the maze. Please use the head() function to print your data frame.

```r
# convert from wide to long
df_rats$subject <- factor(df_rats$subject)
df_rats_long <- df_rats %>%
  pivot_longer(c(`1`, `2`, `3`), names_to = "cheese", values_to = "time")

head(df_rats_long)
```

```
## # A tibble: 6 x 3
##   subject cheese  time
##   <fct>   <chr>  <dbl>
## 1 rat_101 1      14.4
## 2 rat_101 2       9.01
## 3 rat_101 3       8.20
## 4 rat_102 1      11.7
## 5 rat_102 2       8.59
## 6 rat_102 3       8.49
```

Question 9

Compute the mean and standard deviation of the maze time.

```r
df_rats_long %>%
  # organize by amount of cheese
  group_by(cheese) %>%
  # summarize
  summarize(mean = mean(time), # mean function
            sd = sd(time))     # standard deviation function
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 3
##   cheese  mean    sd
##   <chr>  <dbl> <dbl>
## 1 1      12.8  1.43
## 2 2       9.88 0.904
## 3 3       8.51 0.279
```

Question 10

Last one (we promise). With the home visits data, pivot the data frame from long to wide. We want the names from the action column to become unique columns and the values to represent the counts. Please use the head() function to print your data frame.

```r
# pivot from long to wide
df_home_wide <- df_home %>%
  pivot_wider(names_from = action, values_from = count)

head(df_home_wide)
```

```
## # A tibble: 6 x 5
##   location       year interview `home visit` questionnaire
##   <chr>         <dbl>     <dbl>        <dbl>         <dbl>
## 1 Washington DC  2015       103           76           200
## 2 Washington DC  2016        71           43           168
## 3 Washington DC  2017        45           60            90
## 4 St Louis       2015        90           86           210
## 5 St Louis       2016        95           82           175
## 6 St Louis       2017        78           71           106
```

Problem Set 8

For this problem set we will work with four tables that are relational to each other. The following keys link the tables to each other:

- patient_id: patients, schedule
- visit_id: schedule, visits
- doctor_id: visits, doctors

Question 1

You've been tasked to collect information on patients within the patients data set. To start this task, you need to join the patient data to the schedule data. We only want to keep the observations that are both present in the patient data AND the schedule data.

Which kind of join do you use?

inner join

How many observations do you see? Note: Some patients have multiple visits.

124 observations

```r
# inner join by patient_id
inner.join.patient <- patients %>%
  inner_join(schedule, by = "patient_id")

#str(inner.join.patient)
```

Question 2

In the visits data, we have a variable called "follow_up" where Y means a follow-up is needed and N means a follow-up is not needed. How many of these patients require a follow-up? You will want to first make a join and then subset.

Which join did you use?

left join

How many patients need a follow-up?

27

```r
left.follow.up <- inner.join.patient %>%
  left_join(visits, by = "visit_id")

# two ways we can filter:
follow.up <- left.follow.up %>% filter(follow_up == "Y")
follow.up <- left.follow.up[which(left.follow.up$follow_up == "Y"), ]
```

Question 3

Which doctors do these patients need follow-up with? Print out the doctors. You can use unique(). Call this data frame **doctors.contact**.

Which join did you use?

left join

```r
doctors.contact <- follow.up %>%
  left_join(doctors, by = "doctor_id")

unique(doctors.contact$doctor)
```

```
##  [1] "Ariadne Anthony"   "Millie Albert"     "Ellesha Castaneda"
##  [4] "Bea Frame"         "Vera Irwin"        "Cade Gale"
##  [7] "Estelle Landry"    "Wiktoria Travis"   "Huzaifa Chung"
## [10] "Jamie-Lee Wilder"  "Jeremy Camacho"    "Daanyaal Griffin"
## [13] "Ammar Phelps"      "Rabia Browning"    "Amritpal Goodman"
## [16] "Merlin Jacobs"     "Tudor Moran"
```

Question 4 / Challenge

Find out which patients that are in schedule but not in doctors.contact. Hint: anti_join()

Create a unique list of these patients with their demographic information from the patients data.

```
not.in.contact <- schedule %>%
  anti_join(doctors.contact, by = "patient_id") %>% # drop observations in schedule
  left_join(patients, by = "patient_id") %>% # join demographic information
  select(-c(visit_id, date)) %>%  # drop variables that prevent us from using distinct() properly
  distinct() # grab distinct rows
```