Exercise 8

Marc Dotson

In RStudio on Posit Cloud, do the following in *two* documents.

1. In a Quarto document with format: docx, wrangle the customer data to include the total amount spent in-store for each customer. Write this data as a CSV file *in the Exercises folder*.
2. In an R Markdown document, modify the dashboard from class by importing the new data, adding a second input function to the sidebar to select from college\_degree, modifying the plot to filter on both input functions, replacing age with total amount spent in-store in the plot, and modifying the call-out boxes to filter on both input functions.
3. Knit the R Markdown document and publish it on shinyapps.io. Include the link at the end of the Quarto document (add a link with [text](URL)), rendered as a Word document, and upload to Canvas.
4. Before Tuesday, download and install the latest version of [R](https://cran.r-project.org/) and [RStudio](https://posit.co/download/rstudio-desktop/) (in that order). (If you have a Mac, note the difference between Intel and Apple silicon!)

**Five points total, one point each for:**

* **Wrangling to get the new crm\_data.**
* **Adding the college\_degree input to the dashboard.**
* **Modifying the plot and value boxes in the dashboard.**
* **Publishing to shinapps.io.**
* **Submitting a rendered Word document with the link included.**

## Import and Wrangle Data

We first need to get the store\_revenue data. I’ve already connected to the database and written it locally, but here are the steps I took.

# Load packages.  
library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ dplyr 1.1.4 ✔ readr 2.1.4  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ ggplot2 3.4.4 ✔ tibble 3.2.1  
✔ lubridate 1.9.3 ✔ tidyr 1.3.0  
✔ purrr 1.0.2   
── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ dplyr::filter() masks stats::filter()  
✖ dplyr::lag() masks stats::lag()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(dplyr)  
library(DBI)

# Connect to the database.  
con <- dbConnect(  
 RPostgreSQL::PostgreSQL(),  
 dbname = "analyticsdb",  
 host = "analyticsdb.ccutuqssh92k.us-west-2.rds.amazonaws.com",  
 port = 55432,  
 user = "quantmktg",  
 password = rstudioapi::askForPassword("Database password")  
)  
  
# Import the data.  
store\_revenue <- tbl(con, "store\_revenue") |>  
 collect()  
  
# Disconnect from the database.  
dbDisconnect(con)  
  
# Write as a CSV.  
write\_csv(store\_revenue, here::here("Data", "store\_revenue.csv"))

Now to import the data that I have locally.

# Import the data.  
customer\_data <- read\_csv(here::here("Data", "customer\_data.csv"))

Rows: 10531 Columns: 13  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
chr (8): gender, married, college\_degree, region, state, review\_time, review...  
dbl (5): customer\_id, birth\_year, income, credit, star\_rating  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

store\_revenue <- read\_csv(here::here("Data", "store\_revenue.csv"))

Rows: 10531 Columns: 169  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
dbl (169): customer\_id, jan\_2005, feb\_2005, mar\_2005, apr\_2005, may\_2005, ju...  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

Now we can tidy the data to calculate the amount spent in-store for each customer.

# Summarize the amount spent overall by each customer.  
total\_spent <- store\_revenue |>   
 pivot\_longer(  
 -customer\_id,  
 names\_to = "month\_year",  
 values\_to = "amount\_spent"  
 ) |>   
 group\_by(customer\_id) |>   
 summarize(total\_spent = sum(amount\_spent))  
  
# Join the new total\_spent column onto the rest of customer\_data.  
crm\_data <- customer\_data |>   
 left\_join(total\_spent, join\_by(customer\_id))  
  
# Write the new crm\_data as a CSV in Exercises.  
write\_csv(crm\_data, here::here("Exercises", "Exercise 08", "crm\_data.csv"))

## Dashboard

You can see my modified Patagonia CRM Dashboard [here](https://marcdotson.shinyapps.io/exercise_08/).