Homework 3: Databases, web scraping, and a basic Shiny app

GUIDO BOZZANO

2023-05-30

Table of Contents

# Money in UK politics

[The Westminster Accounts](https://news.sky.com/story/the-westminster-accounts-12786091), a recent collaboration between Sky News and Tortoise Media, examines the flow of money through UK politics. It does so by combining data from three key sources:

1. [Register of Members’ Financial Interests](https://www.parliament.uk/mps-lords-and-offices/standards-and-financial-interests/parliamentary-commissioner-for-standards/registers-of-interests/register-of-members-financial-interests/),
2. [Electoral Commission records of donations to parties](http://search.electoralcommission.org.uk/English/Search/Donations), and
3. [Register of All-Party Parliamentary Groups](https://www.parliament.uk/mps-lords-and-offices/standards-and-financial-interests/parliamentary-commissioner-for-standards/registers-of-interests/register-of-all-party-party-parliamentary-groups/).

You can [search and explore the results](https://news.sky.com/story/westminster-accounts-search-for-your-mp-or-enter-your-full-postcode-12771627) through the collaboration’s interactive database. Simon Willison [has extracted a database](https://til.simonwillison.net/shot-scraper/scraping-flourish) and this is what we will be working with. If you want to read more about [the project’s methodology](https://www.tortoisemedia.com/2023/01/08/the-westminster-accounts-methodology/).

## Open a connection to the database

The database made available by Simon Willison is an SQLite database

sky\_westminster <- DBI::dbConnect(  
 drv = RSQLite::SQLite(),  
 dbname = here::here("data", "sky-westminster-files.db")  
)

How many tables does the database have?

DBI::dbListTables(sky\_westminster)

## [1] "appg\_donations" "appgs" "member\_appgs" "members"   
## [5] "parties" "party\_donations" "payments"

## Which MP has received the most amount of money?

You need to work with the payments and members tables and for now we just want the total among all years. To insert a new, blank chunk of code where you can write your beautiful code (and comments!), please use the following shortcut: Ctrl + Alt + I (Windows) or cmd + option + I (mac)

#First off I being by loading each table into a different dataset, to do I use the dplr::tbl function to map each of the available tables into 7 separate tables. Also, I make sure to name them differently so that I can easily understand what type of information each table has  
appgdonations\_db <- dplyr::tbl(sky\_westminster, "appg\_donations")  
appgs\_db <- dplyr::tbl(sky\_westminster, "appgs")  
memberappgs\_db <- dplyr::tbl(sky\_westminster, "member\_appgs")  
members\_db <- dplyr::tbl(sky\_westminster, "members")  
parties\_db <- dplyr::tbl(sky\_westminster, "parties")  
partydonations\_db <- dplyr::tbl(sky\_westminster, "party\_donations")  
payments\_db <- dplyr::tbl(sky\_westminster, "payments")  
  
#To find which MP received the most amount of money I choose the members\_db table  
members\_db %>%   
 rename(member\_id = id) %>% #rename the id variable so that it matches the id variable from the payments\_db table  
 left\_join(payments\_db, by="member\_id") %>% #Do a left join with payments\_db by member\_id  
 group\_by(name) %>% #Group by name of each MP  
 summarise(sumvalue = sum(value)) %>% #Sum the total amout of donation value by name  
 arrange(desc(sumvalue)) #Arrange the sumvalue column in a descending order

## Warning: Missing values are always removed in SQL aggregation functions.  
## Use `na.rm = TRUE` to silence this warning  
## This warning is displayed once every 8 hours.

## # Source: SQL [?? x 2]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## # Ordered by: desc(sumvalue)  
## name sumvalue  
## <chr> <dbl>  
## 1 Theresa May 2809765.  
## 2 Sir Geoffrey Cox 2191387.  
## 3 Boris Johnson 1282402   
## 4 Keir Starmer 799936.  
## 5 Andrew Mitchell 769373.  
## 6 Fiona Bruce 712321.  
## 7 John Redwood 692438.  
## 8 Rishi Sunak 546043   
## 9 Liz Truss 538678.  
## 10 Ed Davey 441681.  
## # ℹ more rows

#Answer: Theresa May has received the largest amount of money

## Any entity that accounts for more than 5% of all donations?

Is there any entity whose donations account for more than 5% of the total payments given to MPs over the 2020-2022 interval? Who are they and who did they give money to?

#To answer this question I choose the partydonations\_db table and I rename the party\_id to id so that it matches parties\_db  
  
partydonations\_db %>%   
 rename(id = party\_id) %>%   
 left\_join(parties\_db, by="id") %>%   
 group\_by(entity, name) %>% #Group by entity that donated money and name of MP  
 summarise(sumvalue=sum(value), .groups = "drop") %>% #Sumvalue again droping groups  
 mutate(percentage = sumvalue/sum(sumvalue)\*100) %>% #Calculate the % amount to understand which entity had over 5%  
 filter(percentage > 5) %>% #Filter the information to find those percentages above 5%  
 arrange(desc(percentage)) #Sort percentages in descending order

## # Source: SQL [2 x 4]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## # Ordered by: desc(percentage)  
## entity name sumvalue percentage  
## <chr> <chr> <dbl> <dbl>  
## 1 Unite Labour 8821467. 6.91  
## 2 Lord David Sainsbury Liberal Democrats 8000000 6.27

#Answer: both the Unite and Lord David Sainsbury entities account for more than 5% donations with 6.91% and 6.26% respectively. Also, each has donated to the Labour and Liberal Democrats party respectively.

## Do entity donors give to a single party or not?

* How many distinct entities who paid money to MPS are there?
* How many (as a number and %) donated to MPs belonging to a single party only?

#To find the distinct entities I carried out different distinct entities to different tables  
  
#This first one has 1077 entities that have donated money to MPs  
partydonations\_db %>%   
 filter(value > 0) %>%   
 summarise(distinctent=n\_distinct(entity))

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## distinctent  
## <int>  
## 1 1077

#This first one has 2213 entities that have donated money to MPs  
members\_db %>%   
 rename(member\_id = id) %>%   
 left\_join(payments\_db, by = "member\_id") %>%   
 filter(value > 0) %>%   
 summarise(distinctent=n\_distinct(entity))

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## distinctent  
## <int>  
## 1 2213

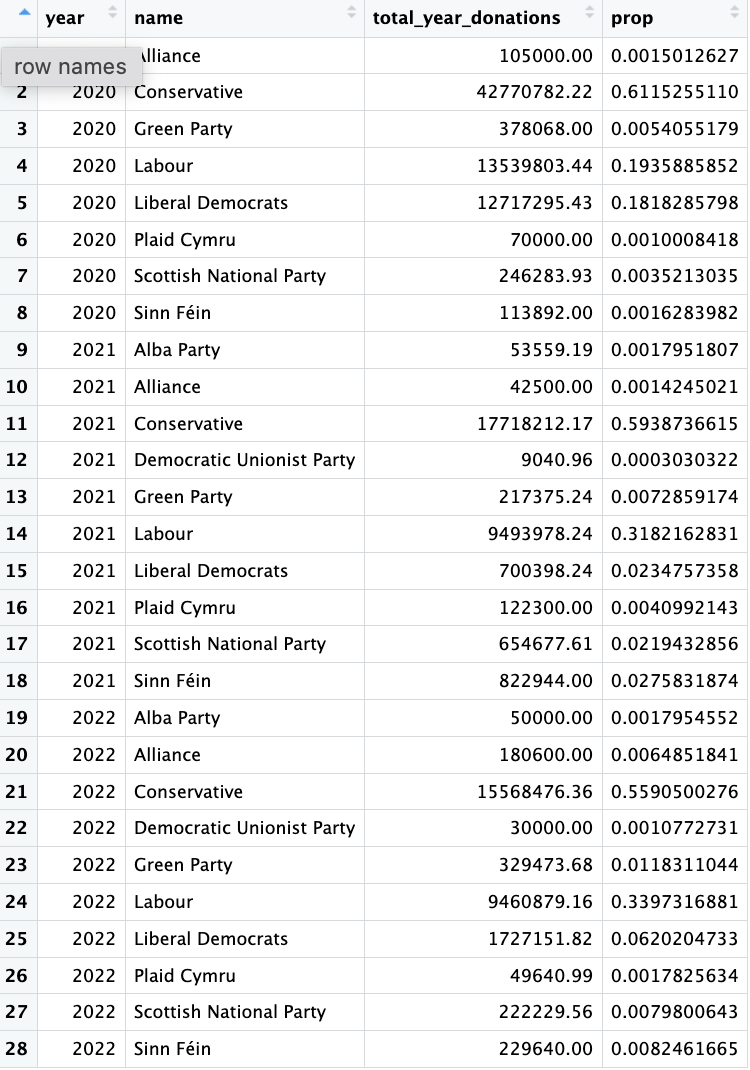
#This first one has 597 entities that have donated to MPs belonging to a single party only  
partydonations\_db %>%   
 rename(id = party\_id) %>%   
 left\_join(parties\_db, by="id") %>%   
 group\_by(entity,name) %>%   
 summarise(count=n(), .groups = "drop") %>%   
 filter(count == 1) %>%   
 summarise(sumvalue=sum(count))

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## sumvalue  
## <int>  
## 1 597

#In this question I have multiple answers but I don't know how to divide between the table information to obtain the percentage. I've given a calculated answer below  
  
#Answer: depending on the table the percentage could be either 597/1077 = 55.4% or 597/2213 = 26.9% that donated to a single party only.

## Which party has raised the greatest amount of money in each of the years 2020-2022?

I would like you to write code that generates the following table.

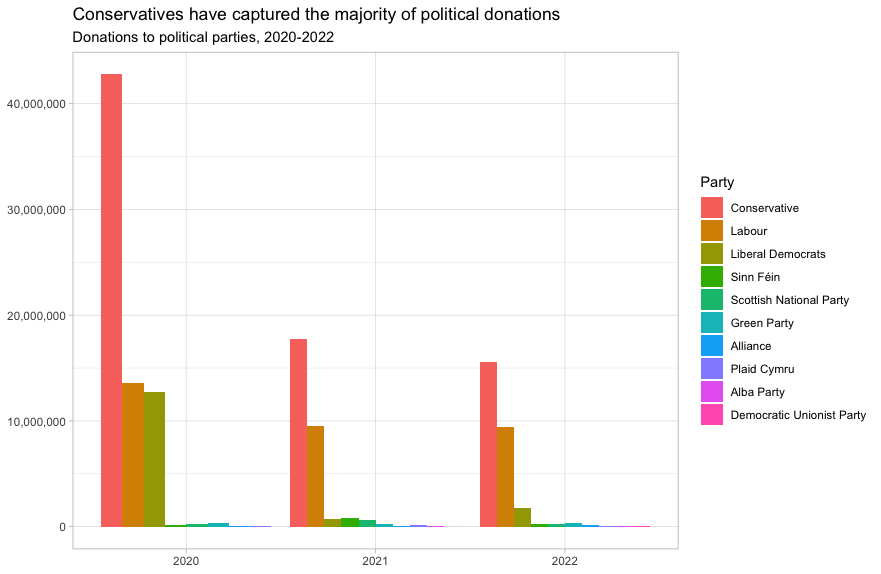


#To replicate this table I choose partydonations\_db  
partydonations\_db %>%   
 rename(id = party\_id) %>% #Rename party\_id to id so that it matches the id variable from parties\_db  
 left\_join(parties\_db, by="id") %>% #Perform a left join by id  
 mutate(year=str\_sub(date,start=1, end=4)) %>% #Mutate a year variable based a substraction from a string that starts on the first character from the date varible up until the 4th  
 group\_by(year, name) %>% #Group the information by year and name of the party  
 summarise(total\_year\_donations=sum(value)) %>% #Calculate the sum value  
 mutate(prop = total\_year\_donations/sum(total\_year\_donations)) %>% #Calculate the percentage using a mutate  
 arrange(year) #Arrange the information by year in ascending order (starting from 2020 to 2022)

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

## # Source: SQL [?? x 4]  
## # Database: sqlite 3.41.2 [/Users/guidobozzano/Desktop/mydsb23hw/data/sky-westminster-files.db]  
## # Groups: year  
## # Ordered by: year  
## year name total\_year\_donations prop  
## <chr> <chr> <dbl> <dbl>  
## 1 2020 Alliance 105000 0.00150  
## 2 2020 Conservative 42770782. 0.612   
## 3 2020 Green Party 378068 0.00541  
## 4 2020 Labour 13539803. 0.194   
## 5 2020 Liberal Democrats 12717295. 0.182   
## 6 2020 Plaid Cymru 70000 0.00100  
## 7 2020 Scottish National Party 246284. 0.00352  
## 8 2020 Sinn Féin 113892 0.00163  
## 9 2021 Alba Party 53559. 0.00180  
## 10 2021 Alliance 42500 0.00142  
## # ℹ more rows

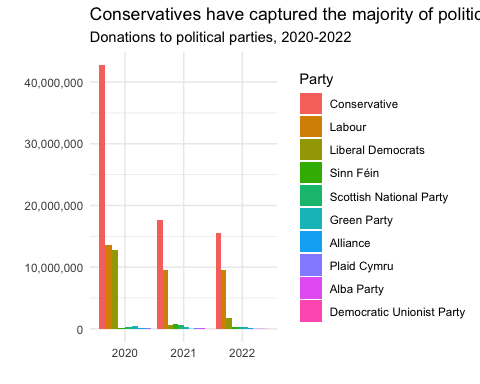
… and then, based on this data, plot the following graph.



This uses the default ggplot colour pallete, as I dont want you to worry about using the [official colours for each party](https://en.wikipedia.org/wiki/Wikipedia:Index_of_United_Kingdom_political_parties_meta_attributes). However, I would like you to ensure the parties are sorted according to total donations and not alphabetically. You may even want to remove some of the smaller parties that hardly register on the graph. Would facetting help you?

# I replicate the table from above and change its name so that I don't overwrite the previous information  
  
partydonations\_db2 <- partydonations\_db %>%   
 rename(id = party\_id) %>%   
 left\_join(parties\_db, by="id") %>%   
 mutate(year=str\_sub(date,start=1, end=4)) %>%   
 group\_by(year, name) %>%   
 summarise(sumvalue=sum(value)) %>%   
 arrange(year, desc(sumvalue))   
  
# Plotting the bar chart  
ggplot(data = partydonations\_db2, aes(x = year, y = sumvalue, fill = reorder(name, -sumvalue))) + #Choose the year as x-axis and sumvalue as y-axis. To breakdown the information by name I perform a reorder by name and sumvalue  
 geom\_bar(stat = "identity", position = "dodge") + #Do a position = "dodge" so that each party name is displayed as independent column  
 labs(x = "", y = "", fill = "Party") + #Set the fill as Party  
 scale\_fill\_discrete(name = "Party") +  
 theme\_minimal() +  
 ggtitle("Conservatives have captured the majority of political donations") + #Set the title  
 labs(subtitle = "Donations to political parties, 2020-2022") + #Set the subtitle  
 scale\_y\_continuous(labels = comma\_format(scale = 1e0)) #Set the y scale to display the millions amount

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



Finally, when you are done working with the databse, make sure you close the connection, or disconnect from the database.

dbDisconnect(sky\_westminster)

# Anonymised Covid patient data from the CDC

We will be using a dataset with [anonymous Covid-19 patient data that the CDC publishes every month](https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data-with-Ge/n8mc-b4w4). The file we will use was released on April 11, 2023, and has data on 98 million of patients, with 19 features. This file cannot be loaded in memory, but luckily we have the data in parquet format and we will use the {arrow} package.

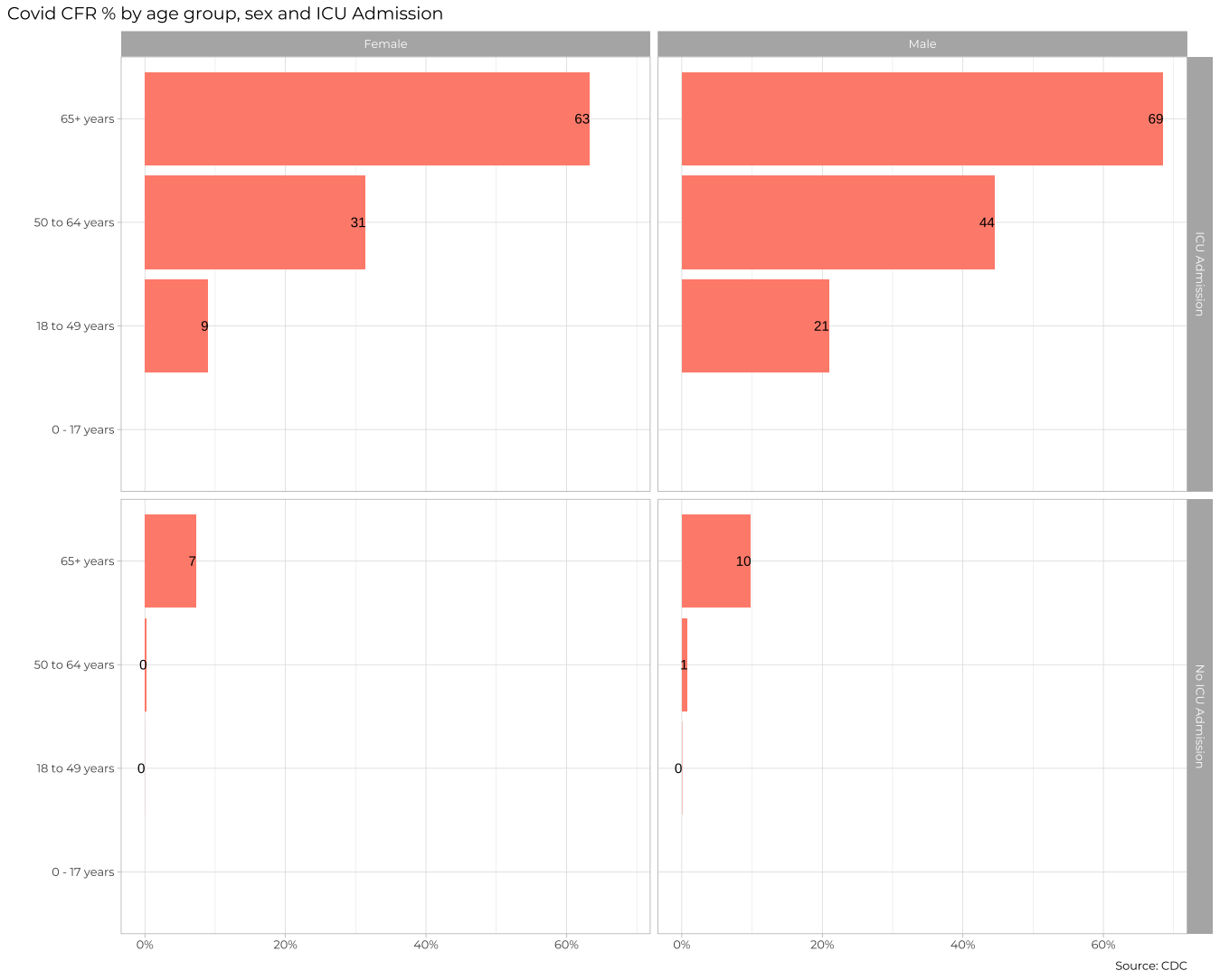
## Obtain the data

The dataset cdc-covid-geography in in parquet format that {arrow}can handle. It is > 600Mb and too large to be hosted on Canvas or Github, so please download it from dropbox <https://www.dropbox.com/sh/q1yk8mmnbbrzavl/AAAxzRtIhag9Nc_hODafGV2ka?dl=0> and save it in your dsb repo, under the data folder

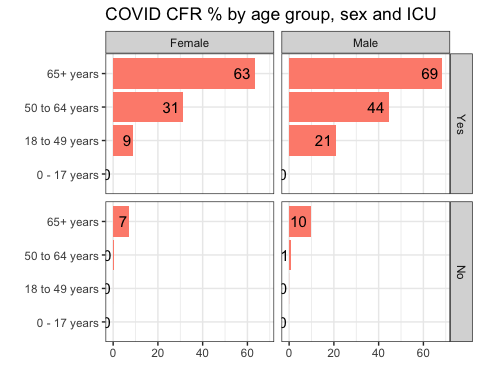
## 0.021 sec elapsed

## FileSystemDataset with 1 Parquet file  
## 97,799,772 rows x 19 columns  
## $ case\_month <string> "2021-09", "2022-09", "2022-01", "2020…  
## $ res\_state <string> "TX", "TX", "TX", "CA", "IL", "CA", "N…  
## $ state\_fips\_code <int32> 48, 48, 48, 6, 17, 6, 36, 36, 36, 53, …  
## $ res\_county <string> "TARRANT", NA, "HARRIS", "SAN BERNARDI…  
## $ county\_fips\_code <int32> 48439, NA, 48201, 6071, 17031, 6085, 3…  
## $ age\_group <string> "18 to 49 years", "18 to 49 years", "1…  
## $ sex <string> "Male", "Male", "Female", "Female", "F…  
## $ race <string> "White", "White", "Unknown", "Asian", …  
## $ ethnicity <string> "Non-Hispanic/Latino", "Non-Hispanic/L…  
## $ case\_positive\_specimen\_interval <int32> NA, NA, NA, NA, 0, NA, 0, 0, 0, 0, 0, …  
## $ case\_onset\_interval <int32> NA, NA, -1, NA, 0, NA, NA, NA, NA, 0, …  
## $ process <string> "Missing", "Missing", "Missing", "Miss…  
## $ exposure\_yn <string> "Missing", "Missing", "Missing", "Miss…  
## $ current\_status <string> "Laboratory-confirmed case", "Probable…  
## $ symptom\_status <string> "Missing", "Missing", "Symptomatic", "…  
## $ hosp\_yn <string> "Missing", "Missing", "No", "No", "No"…  
## $ icu\_yn <string> "Missing", "Missing", "Missing", "Miss…  
## $ death\_yn <string> "Missing", "Missing", "Missing", "Miss…  
## $ underlying\_conditions\_yn <string> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA…

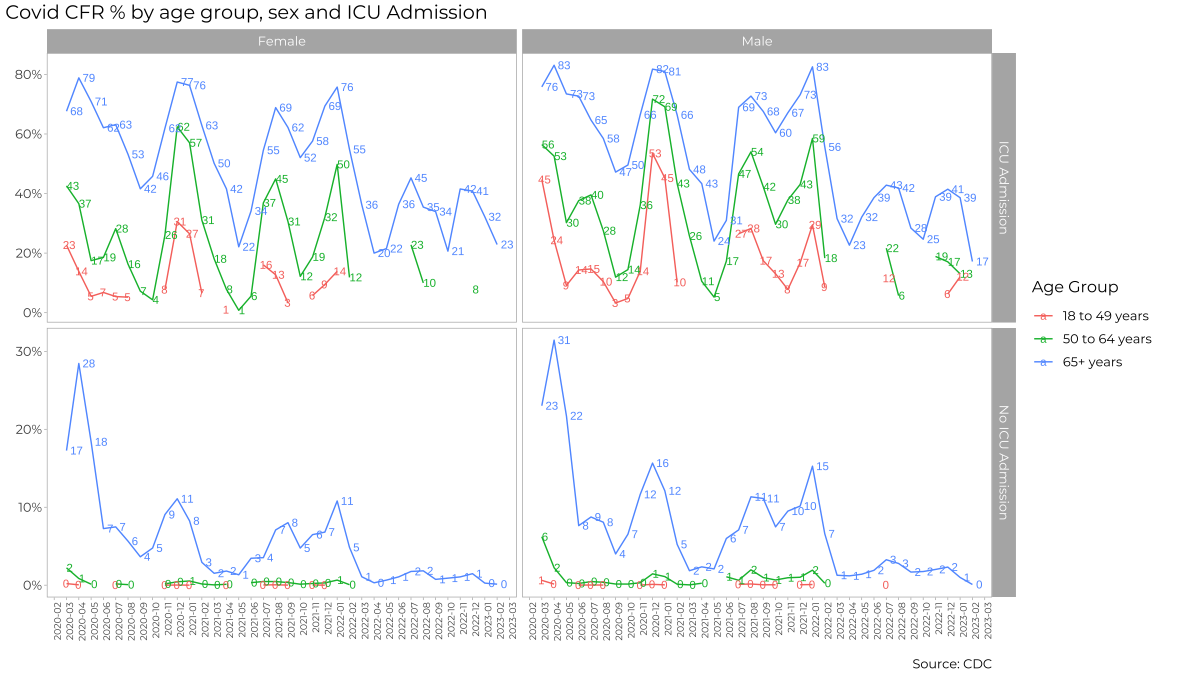
Can you query the database and replicate the following plot?



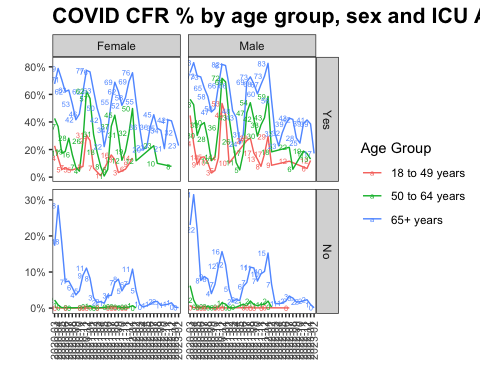
#To replicate this graph I begin by creating table from based off the cdc\_data parquet file  
dbcovid <- cdc\_data %>%   
 filter(sex %in% c("Male", "Female"), #Perform a sex filter that only contains Male and Female  
 !(age\_group %in% c("Unknown", "Missing")), #Exclude the Unkown and Missing values from the age\_group variable  
 icu\_yn %in% c("Yes", "No"), #Filter only the Yes and No from the Icu\_yn varaible  
 death\_yn %in% c("Yes", "No")) %>% #Filter only the Yes and No from the death\_yn variable  
 group\_by(age\_group, icu\_yn, sex) %>% #Group the information by age\_group, icu\_yn and sex as its displayed in the graph  
 summarise(count = n(), #Calculate the number of cases  
 total\_deaths = sum(death\_yn == "Yes"), #Calculate the number of deaths  
 CFR = (total\_deaths / count)\*100) %>% #Calculate the case fatality ratio as total\_deaths / number of cases  
collect() #Collect the information  
  
dbcovid$sex <- factor(dbcovid$sex, levels = c("Female", "Male")) #Set the order as it appears in the graph, first Female then Male  
dbcovid$icu\_yn <- factor(dbcovid$icu\_yn, levels = c("Yes", "No")) #Set the order for ICU admission as Yes and No  
  
ggplot(dbcovid, aes(x = CFR, y = age\_group)) + #create a ggplot with the CFR in the x axis and the age\_group in the y axis  
 geom\_col(position = "dodge", fill = "#FE8E7C") + #set the chart as geom\_col  
 geom\_text(aes(label = round(CFR,0)), #set the labels based on the CFR variable  
 hjust = 1.25,   
 colour = "black",   
 size = 4) +  
 facet\_grid(rows = vars(icu\_yn), #Facet grid the chart by rows with icu\_yn variable and cols with the sex variable  
 cols = vars(sex)) +  
 labs(title = "COVID CFR % by age group, sex and ICU", #Set the title  
 x = "", #Hide the x axis title  
 y = "", #Hide the y axis title  
 ) +  
 theme\_bw()



The previous plot is an aggregate plot for all three years of data. What if we wanted to plot Case Fatality Ratio (CFR) over time? Write code that collects the relevant data from the database and plots the following



#To replicate this chart I being by creating another table named dbcovid2 based of cdc\_data. I also applied the same filsters as before and calculated the same metrics as before  
  
dbcovid2 <- cdc\_data %>%   
 filter(sex %in% c("Male", "Female"),  
 !(age\_group %in% c("Unknown", "Missing","0 - 17 years")),  
 icu\_yn %in% c("Yes", "No"),  
 death\_yn %in% c("Yes", "No")) %>%  
 group\_by(age\_group, icu\_yn, sex, case\_month) %>%  
 summarise(count = n(),  
 total\_deaths = sum(death\_yn == "Yes"),  
 CFR = (total\_deaths / count)) %>%   
collect()  
  
dbcovid2 <- dbcovid2 %>%   
 filter(CFR > 0, CFR < 1) #I included a filter for the CFR variable so that it only includes values between 0 and 1  
  
dbcovid2$sex <- factor(dbcovid2$sex, levels = c("Female", "Male")) #Set the order of the sex variable so its alligned with the chart  
dbcovid2$icu\_yn <- factor(dbcovid2$icu\_yn, levels = c("Yes", "No")) #Set the order of the icu\_yn varible so its alligned with the chart  
  
ggplot(dbcovid2, aes(x = case\_month, y = CFR, group = age\_group, color = age\_group)) + #Plot the information using ggplot, set the x axis as case\_month then y-axis as CFR variable and the group and color as age\_group variables  
 geom\_line() +  
 geom\_text(aes(label = ifelse(CFR>0,round(CFR\*100,0),""), color = age\_group), #define the labels in percetange amount  
 hjust = 1.25,   
 size = 2) +  
 facet\_grid(rows = vars(icu\_yn), #facet grid the chart by setting rows to icu\_yn  
 cols = vars(sex), #facet grid the chart by setting columns as sex  
 scales = "free\_y") + #free the scales for the y axis for each facet grid  
 labs(title = "COVID CFR % by age group, sex and ICU Admission", #set the title  
 x = "", #erase the x axis title  
 y = "", #erase the y axis title  
 ) +  
 scale\_y\_continuous(labels = scales::percent\_format()) + #set y scale to percent format  
 theme\_bw() +  
 theme(  
 plot.title = element\_text(size = 16, face = "bold"), #set the size of the title  
 axis.text.x = element\_text(size = 8, angle = 90, hjust = 1, vjust = 0.5), #display the dates on vertical alignment  
 axis.text.y = element\_text(size = 8),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank()  
 ) +  
 guides(color = guide\_legend(title = "Age Group")) #set the title for the legend to Age Group



For each patient, the dataframe also lists the patient’s states and county [FIPS code](https://en.wikipedia.org/wiki/Federal_Information_Processing_Standard_state_code). The CDC also has information on the [NCHS Urban-Rural classification scheme for counties](https://www.cdc.gov/nchs/data_access/urban_rural.htm)

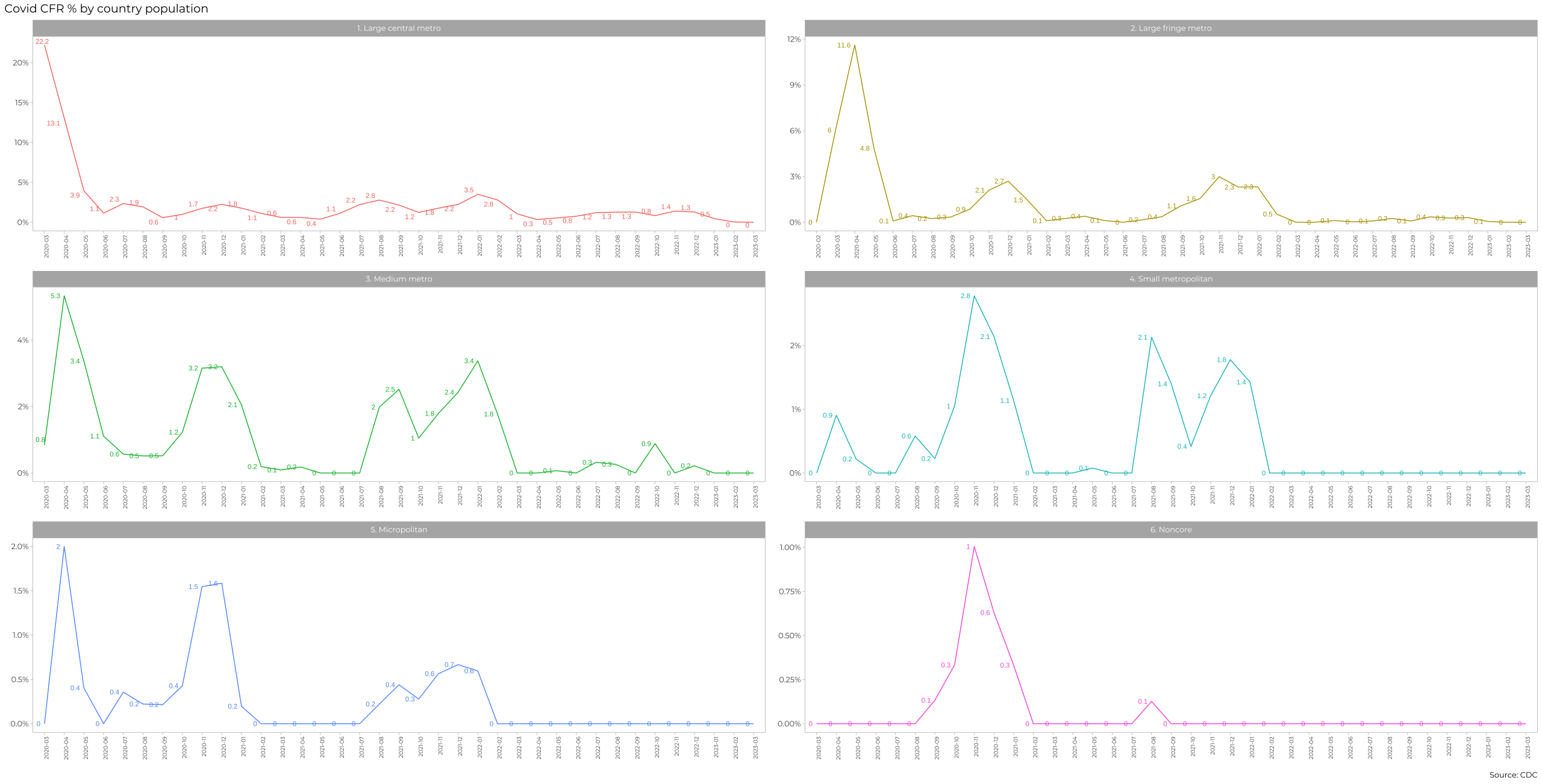
urban\_rural <- read\_xlsx(here::here("data", "NCHSURCodes2013.xlsx")) %>%   
 janitor::clean\_names()   
  
urban\_rural <- urban\_rural %>%   
 mutate(fips\_code = as.integer(fips\_code)) #Transform the fips code into integer as I will be using it afterwards

Each county belongs in six diffent categoreis, with categories 1-4 being urban areas and categories 5-6 being rural, according to the following criteria captured in x2013\_code

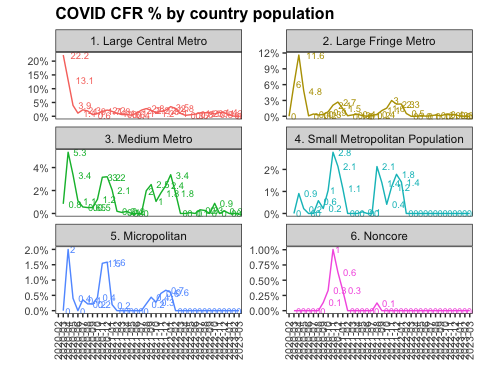
Category name

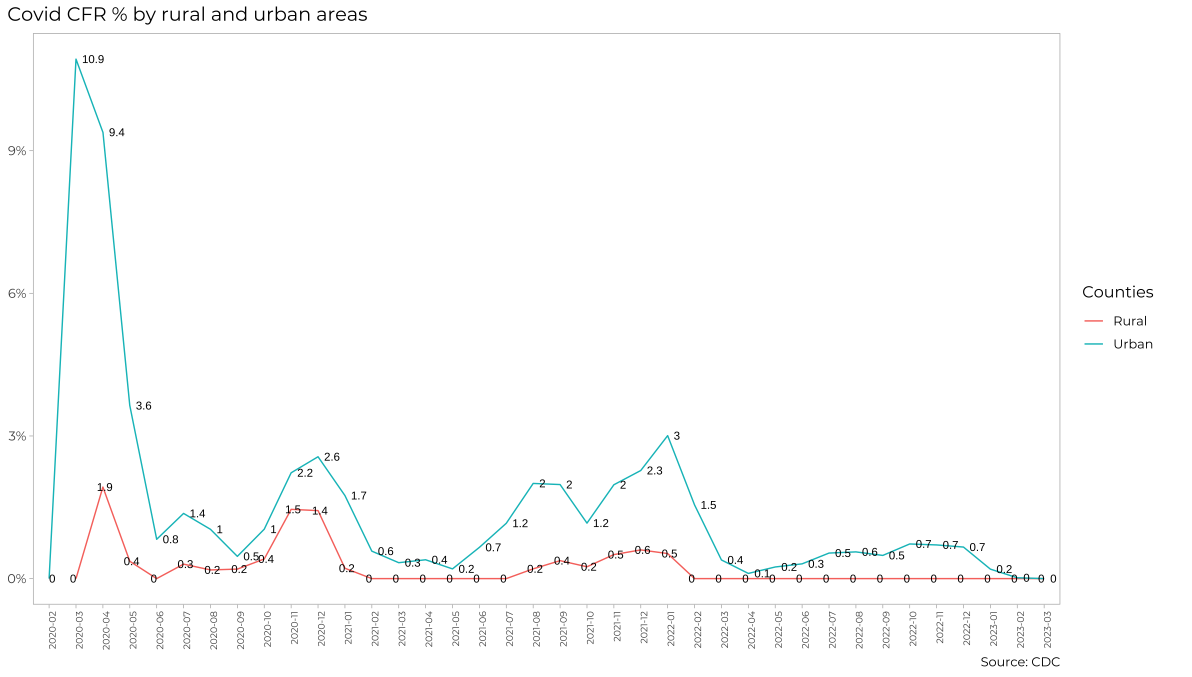
1. Large central metro - 1 million or more population and contains the entire population of the largest principal city
2. large fringe metro - 1 million or more poulation, but does not qualify as 1
3. Medium metro - 250K - 1 million population
4. Small metropolitan population < 250K
5. Micropolitan
6. Noncore

Can you query the database, extract the relevant information, and reproduce the following two graphs that look at the Case Fatality ratio (CFR) in different counties, according to their population?

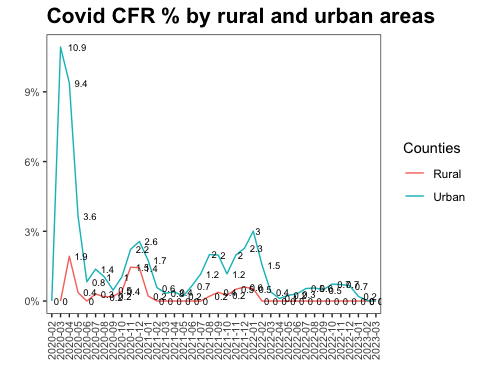


#To replicate this chart I shared my code with Kostis and he shared his. Since I couldn't get to the same result I used his code to replicate the chart  
  
dbcovid3 <- cdc\_data %>%   
 select(sex, age\_group, death\_yn, icu\_yn, case\_month, county\_fips\_code) %>%   
 filter(sex %in% c("Male","Female"),   
 !is.na(age\_group),   
 !(age\_group %in% c("Unknown", "Missing")),  
 death\_yn %in% c("Yes","No"),   
 icu\_yn %in% c("Yes","No")) %>% # select and clean the data  
 group\_by(sex, age\_group, death\_yn, icu\_yn, case\_month, county\_fips\_code) %>%   
 summarise(count = n()) %>%  
 collect()  
  
dbcovid4 <- dbcovid3 %>%  
 pivot\_wider(names\_from = death\_yn,  
 values\_from = count) %>%   
 clean\_names() %>%   
 drop\_na(no) %>%   
 mutate(  
 yes = ifelse(is.na(yes),0,yes),  
 death\_rate = yes/(no+yes)) %>%   
 left\_join(urban\_rural, by=c("county\_fips\_code" = "fips\_code")) %>%   
 drop\_na(county\_fips\_code) %>%   
 mutate(category = case\_when(x2013\_code == 1 ~ "1. Large Central Metro",  
 x2013\_code == 2 ~ "2. Large Fringe Metro",  
 x2013\_code == 3 ~ "3. Medium Metro",  
 x2013\_code == 4 ~ "4. Small Metropolitan Population",  
 x2013\_code == 5 ~ "5. Micropolitan",  
 x2013\_code == 6 ~ "6. Noncore",  
 TRUE ~ "Other"))  
  
summary\_data <- dbcovid4 %>%  
 group\_by(case\_month, category) %>%  
 summarise(CFR = sum(yes)/(sum(no)+sum(yes)), .groups = "drop")  
  
ggplot(summary\_data, aes(x = case\_month, y = CFR, group = category, color = category)) +  
 geom\_line() +  
 geom\_text(aes(label = ifelse(CFR>0,round(CFR\*100,1),0)),  
 hjust = -0.4,   
 size = 2.5) +  
 facet\_wrap(~ category, nrow = 3, scales = "free\_y") +  
 labs(title = "COVID CFR % by country population",  
 x = "",  
 y = "") +  
 theme\_bw() +  
 theme(  
 plot.title = element\_text(size = 12, face = "bold"),  
 axis.text.x = element\_text(size = 8, angle = 90, hjust = 1, vjust = 0.5),  
 axis.text.y = element\_text(size = 8),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 legend.position = "none") +  
 scale\_y\_continuous(labels = scales::percent)





#To replicate this chart I shared my code with Kostis and he shared his. Since I couldn't get to the same result I used his code to replicate the chart  
  
dbcovid6 <- cdc\_data %>%   
 select(sex, age\_group, death\_yn, icu\_yn, case\_month, county\_fips\_code) %>%   
 filter(sex %in% c("Male","Female"),   
 !is.na(age\_group),   
 !(age\_group %in% c("Unknown", "Missing")),  
 death\_yn %in% c("Yes","No"),   
 icu\_yn %in% c("Yes","No")) %>% # select and clean the data  
 group\_by(sex, age\_group, death\_yn, icu\_yn, case\_month, county\_fips\_code) %>%   
 summarise(count = n()) %>%  
 collect()  
  
dbcovid7 <- dbcovid6 %>%  
 pivot\_wider(names\_from = death\_yn,  
 values\_from = count) %>%   
 clean\_names() %>%   
 drop\_na(no) %>%   
 mutate(  
 yes = ifelse(is.na(yes),0,yes),  
 death\_rate = yes/(no+yes)) %>%   
 left\_join(urban\_rural, by=c("county\_fips\_code" = "fips\_code")) %>%   
 drop\_na(county\_fips\_code) %>%   
 mutate(urban14\_rural56 = case\_when(x2013\_code == 5 | x2013\_code == 6 ~ "Rural",  
 TRUE ~ "Urban"))  
  
summary\_data <- dbcovid7 %>%  
 group\_by(case\_month, urban14\_rural56) %>%  
 summarise(CFR = sum(yes)/(sum(no)+sum(yes)), .groups = "drop")  
  
ggplot(summary\_data, aes(x = case\_month, y = CFR, group = urban14\_rural56 , color = urban14\_rural56)) +  
 geom\_line() +  
 geom\_text(aes(label = ifelse(CFR>0,round(CFR\*100,1),0)),  
 hjust = -0.4,   
 size = 2.5,  
 color = "black") +  
 labs(title = "Covid CFR % by rural and urban areas",  
 x = "",  
 y = "",  
 color = "Counties") +  
 theme\_bw() +  
 theme(  
 plot.title = element\_text(size = 16, face = "bold"),  
 axis.text.x = element\_text(size = 8, angle = 90, hjust = 1, vjust = 0.5),  
 axis.text.y = element\_text(size = 8),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank()  
 ) +  
 scale\_y\_continuous(labels = scales::percent)



# Money in US politics

In the United States, [*“only American citizens (and immigrants with green cards) can contribute to federal politics, but the American divisions of foreign companies can form political action committees (PACs) and collect contributions from their American employees.”*](https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs)

We will scrape and work with data foreign connected PACs that donate to US political campaigns. The data for foreign connected PAC contributions in the 2022 election cycle can be found at <https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022>. Then, we will use a similar approach to get data such contributions from previous years so that we can examine trends over time.

All data come from [OpenSecrets.org](https://www.opensecrets.org), a *“website tracking the influence of money on U.S. politics, and how that money affects policy and citizens’ lives”*.

library(robotstxt)  
paths\_allowed("https://www.opensecrets.org")

## [1] TRUE

base\_url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022"  
  
contributions\_tables <- base\_url %>%  
 read\_html()

* First, make sure you can scrape the data for 2022. Use janitor::clean\_names() to rename variables scraped using snake\_case naming.

base\_url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022"  
  
contributions\_tables <- base\_url %>%  
 read\_html()   
  
table\_data <- contributions\_tables %>% #Keep the first table from contribution\_tables and store into table\_data  
 html\_nodes("table") %>%  
 .[[1]] %>%   
 html\_table(fill = TRUE)  
  
df <- as.data.frame(table\_data) #Convert the extracted table into a data frame  
  
  
df <- df %>% clean\_names(case = "snake") #Clean the variable names using janitor::clean\_names()

* Clean the data:
  + Write a function that converts contribution amounts in total, dems, and repubs from character strings to numeric values.
  + Separate the country\_of\_origin\_parent\_company into two such that country and parent company appear in different columns for country-level analysis.

# write a function to parse\_currency  
parse\_currency <- function(x){  
 x %>%  
   
 # remove dollar signs  
 str\_remove("\\$") %>%  
   
 # remove all occurrences of commas  
 str\_remove\_all(",") %>%  
   
 # convert to numeric  
 as.numeric()  
}  
  
# clean country/parent co and contributions   
contributions <- df %>%  
 separate(country\_of\_origin\_parent\_company,   
 into = c("country", "parent"),   
 sep = "/",   
 extra = "merge") %>%  
 mutate(  
 total = parse\_currency(total),  
 dems = parse\_currency(dems),  
 repubs = parse\_currency(repubs)  
 )

* Write a function called scrape\_pac() that scrapes information from the Open Secrets webpage for foreign-connected PAC contributions in a given year. This function should
  + have one input: the URL of the webpage and should return a data frame.
  + add a new column to the data frame for year. We will want this information when we ultimately have data from all years, so this is a good time to keep track of it. Our function doesn’t take a year argument, but the year is embedded in the URL, so we can extract it out of there, and add it as a new column. Use the str\_sub() function to extract the last 4 characters from the URL. You will probably want to look at the help for this function to figure out how to specify “last 4 characters”.

scrape\_pac <- function(url) {  
 # Read the HTML content from the website  
 page <- read\_html(url)  
   
 # Extract the table containing the data  
 table\_data <- page %>% html\_nodes("table") %>% .[[1]] %>% html\_table(fill = TRUE)  
   
 # Convert the extracted table into a data frame  
 df <- as.data.frame(table\_data)  
   
 # Clean the variable names using janitor::clean\_names()  
 df <- df %>% clean\_names(case = "snake")  
   
 # Extract the year from the URL and add it as a new column  
 df$year <- str\_sub(url, -4)  
   
 # Return the data frame  
 return(df)  
}  
  
url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022"  
scraped\_data <- scrape\_pac(url)  
print(scraped\_data)

## pac\_name\_affiliate  
## 1 Accenture (Accenture)  
## 2 Acreage Holdings  
## 3 Air Liquide America  
## 4 Airbus Group  
## 5 Alexion Pharmaceuticals (AstraZeneca PLC)  
## 6 Alkermes Inc  
## 7 Allianz of America (Allianz)  
## 8 AMG Vanadium  
## 9 Anheuser-Busch (Anheuser-Busch InBev)  
## 10 AON Corp (AON plc)  
## 11 APL Maritime (CMA CGM)  
## 12 Aptiv PLC  
## 13 Arcadis US (Arcadis NV)  
## 14 Arch Capital Group (US)  
## 15 Arkema Inc  
## 16 Ashton Woods Homes  
## 17 Assured Guaranty Municipal Corp (Assured Guaranty Corp)  
## 18 Astellas US  
## 19 AstraZeneca Pharmaceuticals (AstraZeneca PLC)  
## 20 Atkins North America  
## 21 Austal USA  
## 22 Avangrid Inc (Iberdrola SA)  
## 23 Bacardi USA  
## 24 BAE Systems (BAE Systems)  
## 25 Barclays Group US  
## 26 Barrick Goldstrike Mines  
## 27 BASF Corp  
## 28 Bayer Corp (Bayer AG)  
## 29 Beam Suntory (Suntory Holdings)  
## 30 BMO Financial Corp (Bank of Montreal)  
## 31 BMO Financial Corp (Bank of Montreal)  
## 32 Boehringer Ingelheim Corp  
## 33 Bombardier Aerospace (Bombardier Inc)  
## 34 BP North America  
## 35 Bridgestone Americas  
## 36 Bumble Bee Foods (FCF Co)  
## 37 CAE USA  
## 38 CalPortland Co  
## 39 Carmeuse Lime  
## 40 Case New Holland  
## 41 CEMEX Inc  
## 42 CGI Technologies & Solutions  
## 43 Chubb Group of Insurance Companies  
## 44 CIBC Bancorp  
## 45 Cirrus Aircraft Corp  
## 46 Continental Automotive Systems (Continental AG)  
## 47 Covestro LLC (Covestro AG)  
## 48 Credit Suisse Securities  
## 49 CRH Americas (CRH PLC)  
## 50 Cronos USA  
## 51 CSL Behring  
## 52 Delhaize America  
## 53 Deutsche Bank Securities  
## 54 Diageo North America (Diageo PLC)  
## 55 Eaton Corp (Eaton Corp)  
## 56 EDF Renewables (EDF Group)  
## 57 EDP Renewables NA  
## 58 Eisai Inc  
## 59 Elbit Systems of America  
## 60 Electrolux Home Products  
## 61 Elekta Inc (Elekta AB)  
## 62 Embraer Aircraft Holdings  
## 63 EMD Serono Inc  
## 64 Empire District Electric (Algonquin Power & Utilities)  
## 65 Empower Retirement (Power Financial Corp)  
## 66 Enbridge Inc  
## 67 Endo Pharmaceuticals  
## 68 ENGIE North America  
## 69 Ericsson Inc  
## 70 Essent US Holdings  
## 71 Experian  
## 72 Farmers Group (Zurich Insurance Group)  
## 73 Fincantieri Marinette Marine  
## 74 Finsbury Glover Hering (WPP Group)  
## 75 Florida East Coast Industries (SoftBank Corp)  
## 76 Florida East Coast Railway (Grupo Mexico)  
## 77 Framatome (EDF Group)  
## 78 Fresenius Medical Care North America  
## 79 G4S Secure Solutions  
## 80 Garmin International (Garmin Ltd)  
## 81 GE Appliances  
## 82 Genentech Inc (Roche Holdings)  
## 83 General Cigar Co  
## 84 GenesisCare USA  
## 85 Gerdau Ameristeel Corp  
## 86 Glanbia Foods (Glanbia PLC)  
## 87 Grand Trunk Western-Illinois Central RR  
## 88 Greyhound Lines  
## 89 GSK plc  
## 90 Haleon plc  
## 91 Hannover Life Reassurance Co  
## 92 Hatch LTK (Hatch Ltd)  
## 93 Headwaters Inc  
## 94 Heineken USA  
## 95 Herbalife Nutrition  
## 96 Holcim US (Holcim Group)  
## 97 Horizon Therapeutics  
## 98 HSBC North America (HSBC Holdings)  
## 99 HSBC North America (HSBC Holdings)  
## 100 IBI Group Engineering Services  
## 101 IDEMIA Identity & Security (IDEMIA)  
## 102 IGT Global Solutions  
## 103 Infineon Technologies  
## 104 Intelsat US  
## 105 Intergraph Corp  
## 106 Ipsen Biopharmaceuticals  
## 107 ITG Brands (Imperial Brands)  
## 108 Jackson National Life Insurance (Prudential PLC)  
## 109 Jackson National Life Insurance (Prudential PLC)  
## 110 JBS USA  
## 111 John Hancock Life Insurance  
## 112 Johnson Controls (Johnson Controls International)  
## 113 Komatsu Mining  
## 114 Lanxess Corp (Lanxess AG)  
## 115 Lehigh Hanson  
## 116 Leonardo DRS  
## 117 Liberty Utilities (Algonquin Power & Utilities)  
## 118 Lincare Holdings (Linde plc)  
## 119 Linde plc (Linde plc)  
## 120 Livanova USA  
## 121 Louis Dreyfus Co  
## 122 LSEG US  
## 123 Lundbeck Inc  
## 124 LyondellBasell Industries  
## 125 Maersk Inc (AP Moller-Maersk)  
## 126 Mallinckrodt Pharmaceuticals  
## 127 Maxar Technologies (Maxar Technologies)  
## 128 MBDA Inc  
## 129 Medtronic Inc  
## 130 Mercedes-Benz USA (Daimler AG)  
## 131 Messer North America  
## 132 Mitsubishi Power  
## 133 Molson Coors Beverage Co (Molson Coors Brewing)  
## 134 MUFG Americas  
## 135 Munich American Reassurance  
## 136 Nammo Inc (Nammo AS)  
## 137 National Grid USA (National Grid plc)  
## 138 Nestle Purina PetCare (Nestle SA)  
## 139 Nomura Holding America  
## 140 NOVA Chemicals  
## 141 Novartis Corp (Novartis AG)  
## 142 Novo Nordisk  
## 143 Novocure Inc  
## 144 Nutrien Ag Solutions  
## 145 Otsuka America (Otsuka Pharmaceutical)  
## 146 Permobil Inc (Investor AB)  
## 147 Pernod Ricard USA  
## 148 Pharmavite LLC (Otsuka Pharmaceutical)  
## 149 Philips Electronics North America (Philips)  
## 150 Prime Policy Group (WPP Group)  
## 151 Protective Life Corp (Dai-Ichi Life)  
## 152 Purpose Financial (Grupo Salinas)  
## 153 Putnam Investments (Power Financial Corp)  
## 154 RBC Bank (Royal Bank of Canada)  
## 155 Recurrent Energy  
## 156 RELX Inc  
## 157 Resolute FP US  
## 158 Reynolds American (British American Tobacco)  
## 159 Rio Tinto America  
## 160 Roche Diagnostics (Roche Holdings)  
## 161 Rolls-Royce North America (Rolls-Royce PLC)  
## 162 Saab North America  
## 163 Sabic Innovative Plastics  
## 164 Safelite Group  
## 165 Samsung Electronics America  
## 166 Sanofi US (Sanofi)  
## 167 Santander Bank (Banco Santander)  
## 168 SAP America  
## 169 Schwan's Co  
## 170 Securitas Security Services USA  
## 171 Serco Inc  
## 172 Shell USA  
## 173 Siemens Corp (Siemens AG)  
## 174 Smith & Nephew  
## 175 Smithfield Foods  
## 176 Smiths Group Services Corp (Smiths Group)  
## 177 Smiths Group Services Corp (Smiths Group)  
## 178 Sodexo Inc  
## 179 Solvay America  
## 180 Sony Music Entertainment (Sony Corp)  
## 181 Sony Pictures Entertainment (Sony Corp)  
## 182 Spark Therapeutics (Roche Holdings)  
## 183 SSAB Americas  
## 184 Standard Insurance Co  
## 185 Steris Corp  
## 186 SUEZ Water  
## 187 Sun Life Financial (Sun Life Financial)  
## 188 Sunovion Pharmaceuticals (Sumitomo Chemical)  
## 189 Sunpower Corp  
## 190 Swedish Match North America (Swedish Match AB)  
## 191 Swiss Re America  
## 192 Syngenta Corp  
## 193 Tabacalera USA (Imperial Brands)  
## 194 Takeda Pharmaceuticals USA (Takeda Pharmaceutical Co)  
## 195 TD Bank USA  
## 196 TE Connectivity  
## 197 TECO Energy  
## 198 Teva Pharmaceuticals USA  
## 199 Toyota Motor North America  
## 200 Trane Technologies  
## 201 TransAmerica  
## 202 Transcanada USA Services (TC Energy)  
## 203 UBS Americas  
## 204 UCB Inc (UCB SA)  
## 205 Ultra Electronics USA  
## 206 Universal Music Group (Universal Music Group)  
## 207 Valent USA (Sumitomo Chemical)  
## 208 Varian Medical Systems (Siemens AG)  
## 209 VNA Holding (AB Volvo)  
## 210 VT Halter Marine  
## 211 Washington Gas Light Co (AltaGas Ltd)  
## 212 Westinghouse Electric  
## 213 WSP USA  
## 214 ZF Automotive US (ZF Friedrichshafen AG)  
## 215 Zurich Insurance (Zurich Insurance Group)  
## country\_of\_origin\_parent\_company total  
## 1 Ireland/Accenture plc $3,000  
## 2 Canada/Acreage Holdings $0  
## 3 France/L'Air Liquide SA $17,300  
## 4 Netherlands/Airbus Group $193,500  
## 5 UK/AstraZeneca PLC $186,250  
## 6 Ireland/Alkermes Plc $84,500  
## 7 Germany/Allianz AG Holding $31,500  
## 8 Netherlands/AMG Advanced Metallurgical Group $2,525  
## 9 Belgium/Anheuser-Busch InBev $457,500  
## 10 UK/AON PLC $98,500  
## 11 France/CMA CGM SA $19,000  
## 12 Ireland/Aptiv PLC $10,350  
## 13 Netherlands/Arcadis NV $1,000  
## 14 UK/Arch Capital Group $8,500  
## 15 France/Arkema $51,500  
## 16 Canada/Great Gulf Group $2,500  
## 17 UK/Assured Guaranty Ltd $112,100  
## 18 Japan/Astellas Pharma $145,500  
## 19 UK/AstraZeneca PLC $310,750  
## 20 UK/SNC-Lavalin $18,000  
## 21 Australia/Austal Ltd $50,200  
## 22 Spain/Iberdrola SA $155,000  
## 23 UK/Bacardi Ltd $46,200  
## 24 UK/BAE Systems $739,500  
## 25 UK/Barclays plc $123,000  
## 26 Canada/Barrick Gold Corp $14,500  
## 27 Germany/BASF SE $412,000  
## 28 Germany/Bayer AG $209,500  
## 29 Japan/Suntory Holdings $46,000  
## 30 Canada/Bank of Montreal $0  
## 31 Canada/Bank of Montreal $193,000  
## 32 Germany/CH Boehringer Sohn $182,900  
## 33 Canada/Bombardier Inc $36,500  
## 34 UK/BP PLC -$2,000  
## 35 Japan/Bridgestone Corp $29,500  
## 36 Taiwan/FCF Co $6,000  
## 37 Canada/CAE Inc $11,000  
## 38 Japan/Taiheiyo Cement Co $214,000  
## 39 Netherlands/Carmeuse SA $1,000  
## 40 Netherlands/CNH Industrial $101,000  
## 41 Mexico/CEMEX SAB de CV $42,500  
## 42 Canada/CGI Group $48,000  
## 43 Switzerland/Chubb Ltd $98,000  
## 44 Canada/Canadian Imperial Bank of Commerce $2,000  
## 45 China/Aviation Industry Corp of China $0  
## 46 Germany/Continental AG $59,000  
## 47 Germany/Covestro AG $14,000  
## 48 Switzerland/Credit Suisse Group $55,000  
## 49 Ireland/CRH PLC $549,600  
## 50 Canada/Cronos Group $33,000  
## 51 Australia/CSL Ltd $55,100  
## 52 Belgium/Ahold Delhaize $43,500  
## 53 Germany/Deutsche Bank AG $0  
## 54 UK/Diageo PLC $17,575  
## 55 Ireland/Eaton Plc $52,353  
## 56 France/EDF Group $38,000  
## 57 Portugal/EDP - Energias de Portugal $22,000  
## 58 Japan/Eisai Co Ltd $23,100  
## 59 Israel/Elbit Systems Ltd $63,000  
## 60 Sweden/AB Electrolux $4,500  
## 61 Sweden/Elekta AB $88,100  
## 62 Brazil/Embraer-Empresa Brasileira de Aeronautic $61,000  
## 63 Germany/Merck KGaA $149,500  
## 64 Canada/Algonquin Power & Utilities -$1,000  
## 65 Canada/Power Financial Corp $45,500  
## 66 Canada/Enbridge Inc $84,500  
## 67 Ireland/Endo International $0  
## 68 France/ENGIE $21,500  
## 69 Sweden/Telefonaktiebolaget LM Ericsson $117,500  
## 70 UK/Essent Group $10,500  
## 71 UK/Experian $554,500  
## 72 Switzerland/Zurich Financial Services AG $262,000  
## 73 Italy/Fincantieri-Cantieri Navali Italiani SpA $19,500  
## 74 UK/WPP Group $124,250  
## 75 Japan/SoftBank Corp $62,000  
## 76 Mexico/Grupo Mexico $2,150  
## 77 France/EDF Group $32,000  
## 78 Germany/Fresenius Medical Care $405,500  
## 79 UK/G4S plc $2,900  
## 80 Switzerland/Garmin Ltd $117,000  
## 81 China/Haier Group $23,000  
## 82 Switzerland/Roche Holdings $295,000  
## 83 Sweden/Scandinavian Tobacco Group $500  
## 84 Australia/GenesisCare $17,000  
## 85 Brazil/Gerdau Inc $14,000  
## 86 Ireland/Glanbia PLC $0  
## 87 Canada/Canadian National Railway $22,000  
## 88 UK/FirstGroup PLC $10,500  
## 89 UK/GlaxoSmithKline $205,000  
## 90 UK/Haleon plc $2,500  
## 91 Germany/Hannover Re $17,500  
## 92 Canada/Hatch Ltd $5,800  
## 93 Australia/Boral Ltd $10,500  
## 94 Netherlands/L'Arche Green NV $26,000  
## 95 UK/Herbalife Ltd $85,500  
## 96 Switzerland/Holcim Group $37,000  
## 97 Ireland/Horizon Pharma PLC $149,000  
## 98 UK/HSBC Holdings $34,500  
## 99 UK/HSBC Holdings $61,000  
## 100 Canada/IBI Group $150  
## 101 France/IDEMIA $29,900  
## 102 Italy/B&D Holding Di Marco Drago e C Sapa $42,900  
## 103 Germany/Infineon Technologies AG $20,000  
## 104 Luxembourg/Intelsat Holdings $49,000  
## 105 Sweden/Hexagon AB $2,900  
## 106 France/Ipsen SA $40,000  
## 107 UK/Imperial Brands $14,000  
## 108 UK/Prudential PLC $169,198  
## 109 UK/Prudential PLC $87,000  
## 110 Brazil/JBS SA $18,500  
## 111 Canada/Manulife Financial $94,500  
## 112 Ireland/Johnson Controls International $211,500  
## 113 Japan/Komatsu Ltd $20,599  
## 114 Germany/Lanxess AG $0  
## 115 Germany/HeidelbergCement AG $102,000  
## 116 Italy/Leonardo SpA $204,000  
## 117 Canada/Algonquin Power & Utilities $0  
## 118 UK/Linde plc $23,500  
## 119 UK/Linde Plc $11,000  
## 120 UK/Livanova Plc $6,000  
## 121 Netherlands/Louis Dreyfus Group $2,000  
## 122 UK/London Stock Exchange Group $2,000  
## 123 Denmark/H Lundbeck A/S $127,400  
## 124 Netherlands/LyondellBasell Industries $95,000  
## 125 Denmark/AP Moller-Maersk $60,500  
## 126 UK/Mallinckrodt Plc $500  
## 127 Canada/Maxar Technologies $173,500  
## 128 UK/MBDA $1,000  
## 129 Ireland/Medtronic Plc $225,000  
## 130 Germany/Daimler AG $3,500  
## 131 Germany/Messer Group $28,500  
## 132 Japan/Mitsubishi Power $1,000  
## 133 Canada/Molson Coors Brewing $77,500  
## 134 Japan/Mitsubishi UFJ Financial Group $0  
## 135 Germany/Munich Re Group $12,500  
## 136 Norway/Nammo AS $9,000  
## 137 UK/National Grid plc $58,500  
## 138 Switzerland/Nestle SA $0  
## 139 Japan/Nomura Holdings $49,500  
## 140 United Arab Emirates/International Petroleum Investment Co $0  
## 141 Switzerland/Novartis AG $281,500  
## 142 Denmark/Novo Nordisk A/S $288,250  
## 143 UK/Novocure $94,500  
## 144 Canada/Nutrien Ltd $46,500  
## 145 Japan/Otsuka Pharmaceutical $125,500  
## 146 Sweden/Investor AB $1,000  
## 147 France/Pernod Ricard SA $27,000  
## 148 Japan/Otsuka Pharmaceutical $94,500  
## 149 Netherlands/Philips $119,750  
## 150 UK/WPP Group $42,750  
## 151 Japan/Dai-Ichi Life $177,000  
## 152 Mexico/Grupo Salinas $46,500  
## 153 Canada/Power Financial Corp $11,500  
## 154 Canada/Royal Bank of Canada $91,500  
## 155 Canada/Canadian Solar $13,900  
## 156 UK/RELX Group $94,500  
## 157 Canada/Resolute Forest Products $75,500  
## 158 UK/British American Tobacco plc $242,100  
## 159 UK/Rio Tinto Group $33,500  
## 160 Switzerland/Roche Holdings $103,000  
## 161 UK/Rolls-Royce PLC $234,000  
## 162 Sweden/Saab AB $4,500  
## 163 Saudi Arabia/SABIC $7,000  
## 164 Belgium/D'Ieteren SA $3,500  
## 165 South Korea/Samsung Group $394,000  
## 166 France/Sanofi $233,000  
## 167 Spain/Banco Santander $86,500  
## 168 Germany/SAP SE $88,500  
## 169 South Korea/CJ CheilJedang $162,500  
## 170 Sweden/Securitas AB $0  
## 171 UK/Serco Group $117,500  
## 172 UK/Shell plc $5,000  
## 173 Germany/Siemens AG $184,000  
## 174 UK/Smith & Nephew Plc $96,000  
## 175 China/WH Group $41,000  
## 176 UK/Smiths Group PLC $33,500  
## 177 UK/Smiths Group PLC $29,948  
## 178 France/Sodexo $2,900  
## 179 Belgium/Solvay SA $65,000  
## 180 Japan/Sony Corp $32,500  
## 181 Japan/Sony Corp $42,500  
## 182 Switzerland/Roche Holdings $8,500  
## 183 Sweden/SSAB AB $31,500  
## 184 Japan/Meiji Yasuda Life Insurance $4,500  
## 185 Ireland/Steris PLC $65,000  
## 186 France/SUEZ Environnement $20,500  
## 187 Canada/Sun Life Financial $132,000  
## 188 Japan/Sumitomo Chemical $107,200  
## 189 France/TotalEnergies $32,000  
## 190 Sweden/Swedish Match AB $5,000  
## 191 Switzerland/Swiss Reinsurance $0  
## 192 China/ChemChina $207,000  
## 193 UK/Imperial Brands $4,321  
## 194 Japan/Takeda Pharmaceutical Co $207,500  
## 195 Canada/Toronto-Dominion Bank $259,500  
## 196 Switzerland/TE Connectivity $92,000  
## 197 Canada/Emera Inc $37,000  
## 198 Israel/Teva Pharmaceutical Industries $72,400  
## 199 Japan/Toyota Motor Corp $1,053,977  
## 200 Ireland/Trane Technologies $26,300  
## 201 Netherlands/Aegon NV $418,500  
## 202 Canada/TC Energy $138,500  
## 203 Switzerland/UBS AG $1,470,000  
## 204 Belgium/UCB SA $52,500  
## 205 UK/Ultra Electronics $80,000  
## 206 Netherlands/Universal Music Group NV $131,000  
## 207 Japan/Sumitomo Chemical $8,000  
## 208 Germany/Siemens AG $65,800  
## 209 Sweden/AB Volvo $67,500  
## 210 Singapore/ST Engineering $18,000  
## 211 Canada/AltaGas Ltd $23,400  
## 212 Canada/Brookfield Business Partners $41,000  
## 213 Canada/WSP Global $64,750  
## 214 Germany/ZF Friedrichshafen AG -$2,000  
## 215 Switzerland/Zurich Insurance Group $308,000  
## dems repubs year  
## 1 $0 $3,000 2022  
## 2 $0 $0 2022  
## 3 $14,800 $2,500 2022  
## 4 $82,500 $111,000 2022  
## 5 $104,000 $82,250 2022  
## 6 $34,500 $50,000 2022  
## 7 $20,500 $11,000 2022  
## 8 $0 $2,525 2022  
## 9 $218,000 $239,500 2022  
## 10 $52,000 $46,500 2022  
## 11 $8,000 $11,000 2022  
## 12 $5,350 $5,000 2022  
## 13 $0 $1,000 2022  
## 14 $5,500 $3,000 2022  
## 15 $27,000 $24,500 2022  
## 16 $0 $2,500 2022  
## 17 $55,500 $56,600 2022  
## 18 $63,500 $82,000 2022  
## 19 $128,500 $182,250 2022  
## 20 $5,000 $13,000 2022  
## 21 $5,000 $45,200 2022  
## 22 $103,000 $52,000 2022  
## 23 $19,500 $25,700 2022  
## 24 $320,000 $419,500 2022  
## 25 $49,500 $73,500 2022  
## 26 $11,000 $3,500 2022  
## 27 $222,000 $190,000 2022  
## 28 $117,500 $92,000 2022  
## 29 $26,000 $20,000 2022  
## 30 $0 $0 2022  
## 31 $101,000 $92,000 2022  
## 32 $86,900 $96,000 2022  
## 33 $16,500 $20,000 2022  
## 34 $0 -$2,000 2022  
## 35 $15,000 $14,500 2022  
## 36 $6,000 $0 2022  
## 37 $6,000 $5,000 2022  
## 38 $13,000 $201,000 2022  
## 39 $0 $1,000 2022  
## 40 $5,000 $96,000 2022  
## 41 $16,500 $26,000 2022  
## 42 $40,000 $8,000 2022  
## 43 $51,500 $46,500 2022  
## 44 $2,000 $0 2022  
## 45 $0 $0 2022  
## 46 $31,000 $28,000 2022  
## 47 $9,500 $4,500 2022  
## 48 $25,000 $30,000 2022  
## 49 $208,500 $341,100 2022  
## 50 $21,000 $12,000 2022  
## 51 $32,700 $22,400 2022  
## 52 $19,000 $24,500 2022  
## 53 $0 $0 2022  
## 54 $12,575 $5,000 2022  
## 55 $24,500 $27,853 2022  
## 56 $32,500 $5,500 2022  
## 57 $20,500 $1,500 2022  
## 58 $11,300 $11,800 2022  
## 59 $16,500 $46,500 2022  
## 60 $1,500 $3,000 2022  
## 61 $41,500 $46,600 2022  
## 62 $18,500 $42,500 2022  
## 63 $80,000 $69,500 2022  
## 64 -$1,000 $0 2022  
## 65 $18,500 $27,000 2022  
## 66 $23,500 $61,000 2022  
## 67 $0 $0 2022  
## 68 $17,500 $4,000 2022  
## 69 $59,500 $58,000 2022  
## 70 $2,500 $8,000 2022  
## 71 $256,500 $298,000 2022  
## 72 $139,000 $123,000 2022  
## 73 $11,000 $8,500 2022  
## 74 $71,500 $52,750 2022  
## 75 $20,500 $41,500 2022  
## 76 $1,850 $300 2022  
## 77 $15,000 $17,000 2022  
## 78 $180,000 $225,500 2022  
## 79 $2,900 $0 2022  
## 80 $39,500 $77,500 2022  
## 81 $5,000 $18,000 2022  
## 82 $154,500 $140,500 2022  
## 83 $0 $500 2022  
## 84 $9,500 $7,500 2022  
## 85 $6,000 $8,000 2022  
## 86 $0 $0 2022  
## 87 $7,000 $15,000 2022  
## 88 $6,000 $4,500 2022  
## 89 $54,000 $151,000 2022  
## 90 $0 $2,500 2022  
## 91 $10,000 $7,500 2022  
## 92 $5,800 $0 2022  
## 93 $1,000 $9,500 2022  
## 94 $14,000 $12,000 2022  
## 95 $35,500 $50,000 2022  
## 96 $22,000 $15,000 2022  
## 97 $89,000 $60,000 2022  
## 98 $11,000 $23,500 2022  
## 99 $36,500 $24,500 2022  
## 100 $0 $150 2022  
## 101 $6,000 $23,900 2022  
## 102 $32,900 $10,000 2022  
## 103 $12,500 $7,500 2022  
## 104 $25,500 $23,500 2022  
## 105 $0 $2,900 2022  
## 106 $22,500 $17,500 2022  
## 107 $5,500 $8,500 2022  
## 108 $50,998 $118,200 2022  
## 109 $50,000 $37,000 2022  
## 110 $7,000 $11,500 2022  
## 111 $58,500 $36,000 2022  
## 112 $117,500 $94,000 2022  
## 113 $9,500 $11,099 2022  
## 114 $0 $0 2022  
## 115 $22,500 $79,500 2022  
## 116 $76,500 $127,500 2022  
## 117 $0 $0 2022  
## 118 $6,000 $17,500 2022  
## 119 $11,000 $0 2022  
## 120 $4,000 $2,000 2022  
## 121 $0 $2,000 2022  
## 122 $2,000 $0 2022  
## 123 $72,400 $55,000 2022  
## 124 $41,500 $53,500 2022  
## 125 $29,500 $31,000 2022  
## 126 $4,500 -$4,000 2022  
## 127 $93,000 $78,000 2022  
## 128 $0 $1,000 2022  
## 129 $119,000 $106,000 2022  
## 130 $0 $3,500 2022  
## 131 $12,000 $16,500 2022  
## 132 $0 $1,000 2022  
## 133 $23,000 $54,500 2022  
## 134 $0 $0 2022  
## 135 $5,000 $7,500 2022  
## 136 $2,000 $7,000 2022  
## 137 $49,500 $9,000 2022  
## 138 $0 $0 2022  
## 139 $19,500 $30,000 2022  
## 140 $0 $0 2022  
## 141 $159,500 $122,000 2022  
## 142 $147,250 $141,000 2022  
## 143 $46,000 $48,500 2022  
## 144 $15,000 $31,500 2022  
## 145 $71,500 $54,000 2022  
## 146 $1,000 $0 2022  
## 147 $13,500 $13,500 2022  
## 148 $50,000 $44,500 2022  
## 149 $54,250 $65,500 2022  
## 150 $24,500 $18,250 2022  
## 151 $46,500 $130,500 2022  
## 152 $18,500 $28,000 2022  
## 153 $5,500 $6,000 2022  
## 154 $45,000 $46,500 2022  
## 155 $9,900 $4,000 2022  
## 156 $58,500 $36,000 2022  
## 157 $5,500 $70,000 2022  
## 158 $63,600 $178,500 2022  
## 159 $14,500 $19,000 2022  
## 160 $50,000 $53,000 2022  
## 161 $87,000 $147,000 2022  
## 162 $2,500 $2,000 2022  
## 163 $6,000 $1,000 2022  
## 164 $3,500 $0 2022  
## 165 $214,500 $179,500 2022  
## 166 $122,500 $110,500 2022  
## 167 $54,000 $32,500 2022  
## 168 $43,500 $45,000 2022  
## 169 $49,000 $113,500 2022  
## 170 $0 $0 2022  
## 171 $69,000 $48,500 2022  
## 172 $0 $5,000 2022  
## 173 $101,500 $82,500 2022  
## 174 $56,500 $39,500 2022  
## 175 $19,000 $22,000 2022  
## 176 $16,500 $17,000 2022  
## 177 $13,500 $16,448 2022  
## 178 $2,900 $0 2022  
## 179 $35,500 $29,500 2022  
## 180 $19,500 $13,000 2022  
## 181 $26,000 $16,500 2022  
## 182 $6,500 $2,000 2022  
## 183 $9,000 $22,500 2022  
## 184 $2,000 $2,500 2022  
## 185 $27,500 $37,500 2022  
## 186 $18,500 $2,000 2022  
## 187 $75,000 $57,000 2022  
## 188 $51,900 $55,300 2022  
## 189 $28,500 $3,500 2022  
## 190 $0 $5,000 2022  
## 191 $0 $0 2022  
## 192 $56,000 $151,000 2022  
## 193 $2,300 $2,021 2022  
## 194 $119,000 $88,500 2022  
## 195 $144,500 $114,000 2022  
## 196 $39,500 $52,500 2022  
## 197 $4,500 $32,500 2022  
## 198 $34,500 $37,900 2022  
## 199 $538,097 $515,880 2022  
## 200 $3,300 $23,000 2022  
## 201 $211,000 $207,500 2022  
## 202 $30,500 $108,000 2022  
## 203 $775,000 $695,000 2022  
## 204 $31,000 $21,500 2022  
## 205 $13,500 $66,500 2022  
## 206 $76,000 $55,000 2022  
## 207 $4,500 $3,500 2022  
## 208 $39,300 $26,500 2022  
## 209 $40,000 $27,500 2022  
## 210 $1,000 $17,000 2022  
## 211 $14,400 $9,000 2022  
## 212 $15,000 $26,000 2022  
## 213 $44,250 $20,500 2022  
## 214 $0 -$2,000 2022  
## 215 $144,000 $164,000 2022

* Define the URLs for 2022, 2020, and 2000 contributions. Then, test your function using these URLs as inputs. Does the function seem to do what you expected it to do?

url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022"  
scraped\_data22 <- scrape\_pac(url)  
#print(scraped\_data22) #I commented this print because the export file became too big to export with redundant information  
   
url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2020"  
scraped\_data20 <- scrape\_pac(url)  
#print(scraped\_data20) #I commented this print because the export file became too big to export with redundant information  
  
url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2000"  
scraped\_data00 <- scrape\_pac(url)  
#print(scraped\_data00) #I commented this print because the export file became too big to export with redundant information  
  
#Answer: Yes, the function is working as expected, it scrapes the url and displays the correct data by year

* Construct a vector called urls that contains the URLs for each webpage that contains information on foreign-connected PAC contributions for a given year.

#Define the range of years  
start\_year <- 2018  
end\_year <- 2022  
  
#Create an empty vector to store the URLs  
urls <- c()  
  
#Run a for over each year and construct the URL  
for (year in start\_year:end\_year) {  
 url <- paste0("https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/", year)  
 urls <- c(urls, url)  
}  
  
urls #Display the vector of URLs

## [1] "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2018"  
## [2] "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2019"  
## [3] "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2020"  
## [4] "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2021"  
## [5] "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/2022"

* Map the scrape\_pac() function over urls in a way that will result in a data frame called contributions\_all.

# Map the scrape\_pac() function over urls and combine the results into a data frame  
contributions\_all <- map\_dfr(urls, ~ scrape\_pac(.x))  
  
# clean country/parent co and contributions   
contributions\_all <- contributions\_all %>%  
 separate(country\_of\_origin\_parent\_company,   
 into = c("country", "parent"),   
 sep = "/",   
 extra = "merge") %>%  
 mutate(  
 total = parse\_currency(total),  
 dems = parse\_currency(dems),  
 repubs = parse\_currency(repubs)  
 )

* Write the data frame to a csv file called contributions-all.csv in the data folder.

# Specify the file path for the CSV file  
file\_path <- "data/contributions-all.csv"  
  
# Write the data frame to the CSV file  
write.csv(contributions\_all, file = file\_path, row.names = FALSE)  
  
# Print a message to confirm the file write  
cat("Data frame has been written to", file\_path, "\n")

## Data frame has been written to data/contributions-all.csv

# Scraping consulting jobs

The website [https://www.consultancy.uk/jobs/](https://www.consultancy.uk/jobs) lists job openings for consulting jobs.

library(robotstxt)  
paths\_allowed("https://www.consultancy.uk") #is it ok to scrape?  
  
base\_url <- "https://www.consultancy.uk/jobs/page/1"  
  
listings\_html <- base\_url %>%  
 read\_html()  
  
tables <- base\_url %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe

Identify the CSS selectors in order to extract the relevant information from this page, namely

1. job
2. firm
3. functional area
4. type

Can you get all pages of ads, and not just the first one, https://www.consultancy.uk/jobs/page/1 into a dataframe?

* Write a function called scrape\_jobs() that scrapes information from the webpage for consulting positions. This function should
  + have one input: the URL of the webpage and should return a data frame with four columns (variables): job, firm, functional area, and type
  + Test your function works with other pages too, e.g., <https://www.consultancy.uk/jobs/page/2>. Does the function seem to do what you expected it to do?
  + Given that you have to scrape ...jobs/page/1, ...jobs/page/2, etc., define your URL so you can join multiple stings into one string, using str\_c(). For instnace, if page is 5, what do you expect the following code to produce?

#Based on the question I wrote the function straight away with the required information. The only need the function needs is the URL and I've set the for to go from pages 1 to 5. Thus, If the page is 5 the code should produce only the data table including information from page 5 but if it is from pages 1 to 5 it will scrap the information from pages 1 to 5 inclusive  
  
scrape\_jobs <- function(url) {  
 job\_list <- character()  
 firm\_list <- character()  
 area\_list <- character()  
 type\_list <- character()  
  
 # Loop through pages  
 for (page\_num in 1:5) { # Change the range as needed  
 page\_url <- paste0(url, "/page/", page\_num)  
 page <- read\_html(page\_url)  
  
 # Scrape job details using CSS selectors  
 jobs <- page %>% html\_nodes('tr.active td:first-child span.title') %>% html\_text()  
 firms <- page %>% html\_nodes('tr.active td.hide-phone a.row-link') %>% html\_text()  
 areas <- page %>% html\_nodes('tr.active td.hide-tablet-and-less a.row-link.initial') %>% html\_text()  
 types <- page %>% html\_nodes('tr.active td.hide-tablet-landscape a.row-link') %>% html\_text()  
  
 # Append scraped data to lists  
 job\_list <- c(job\_list, jobs)  
 firm\_list <- c(firm\_list, firms)  
 area\_list <- c(area\_list, areas)  
 type\_list <- c(type\_list, types)  
 }  
  
 # Process functional area list to extract specific categories and remove newline characters  
 area\_list <- gsub("\n", "", area\_list)  
 area\_list <- sapply(strsplit(area\_list, "\n"), function(x) trimws(x[1]))  
   
 # Create a dataframe  
 df <- data.frame(  
 job = job\_list,  
 firm = firm\_list,  
 "functional area" = area\_list,  
 type = type\_list,  
 stringsAsFactors = FALSE  
 )  
  
 return(df)  
}  
  
# Usage example  
url <- "https://www.consultancy.uk/jobs"  
data\_frame <- scrape\_jobs(url)  
glimpse(data\_frame) #I perform a glimpse here because the data displayed is too large for the word file

## Rows: 225  
## Columns: 4  
## $ job <chr> "Data Scientist", "Healthcare consultant", "Internship…  
## $ firm <chr> "Digital Power", "Develop Consulting", "Simon-Kucher",…  
## $ functional.area <chr> "Data Science", "Lean & SixSigma", "Pricing", "Strateg…  
## $ type <chr> "Job", "Job", "Internship", "Job", "Job", "Job", "Job"…

base\_url <- "https://www.consultancy.uk/jobs/page/1"  
url <- str\_c(base\_url, page)

* Construct a vector called pages that contains the numbers for each page available
* Map the scrape\_jobs() function over pages in a way that will result in a data frame called all\_consulting\_jobs.
* Write the data frame to a csv file called all\_consulting\_jobs.csv in the data folder.

scrape\_jobs <- function(url) {  
 job\_list <- character()  
 firm\_list <- character()  
 area\_list <- character()  
 type\_list <- character()  
  
 page\_url <- url  
  
 # Scrape the number of pages  
 num\_pages <- read\_html(page\_url) %>%   
 html\_nodes('.pagination a.page-link') %>%   
 html\_text() %>%   
 as.numeric() %>%   
 max(na.rm = TRUE)  
  
 # Loop through pages  
 for (page\_num in 1:8) {  
 page\_url <- paste0(url, "/page/", page\_num)  
 page <- read\_html(page\_url)  
  
 # Scrape job details using CSS selectors  
 jobs <- page %>% html\_nodes('tr.active td:first-child span.title') %>% html\_text()  
 firms <- page %>% html\_nodes('tr.active td.hide-phone a.row-link') %>% html\_text()  
 areas <- page %>% html\_nodes('tr.active td.hide-tablet-and-less a.row-link.initial') %>% html\_text()  
 types <- page %>% html\_nodes('tr.active td.hide-tablet-landscape a.row-link') %>% html\_text()  
  
 # Append scraped data to lists  
 job\_list <- c(job\_list, jobs)  
 firm\_list <- c(firm\_list, firms)  
 area\_list <- c(area\_list, areas)  
 type\_list <- c(type\_list, types)  
 }  
  
 # Process functional area list to extract specific categories and remove newline characters  
 area\_list <- gsub("\n", "", area\_list)  
 area\_list <- sapply(strsplit(area\_list, "\n"), function(x) trimws(x[1]))  
   
 # Create a dataframe  
 df <- data.frame(  
 job = job\_list,  
 firm = firm\_list,  
 "functional area" = area\_list,  
 type = type\_list,  
 stringsAsFactors = FALSE  
 )  
  
 return(df)  
}  
  
# Specify the base URL  
base\_url <- "https://www.consultancy.uk/jobs"  
  
# Scrape all consulting jobs  
all\_consulting\_jobs <- map\_df(base\_url, scrape\_jobs)

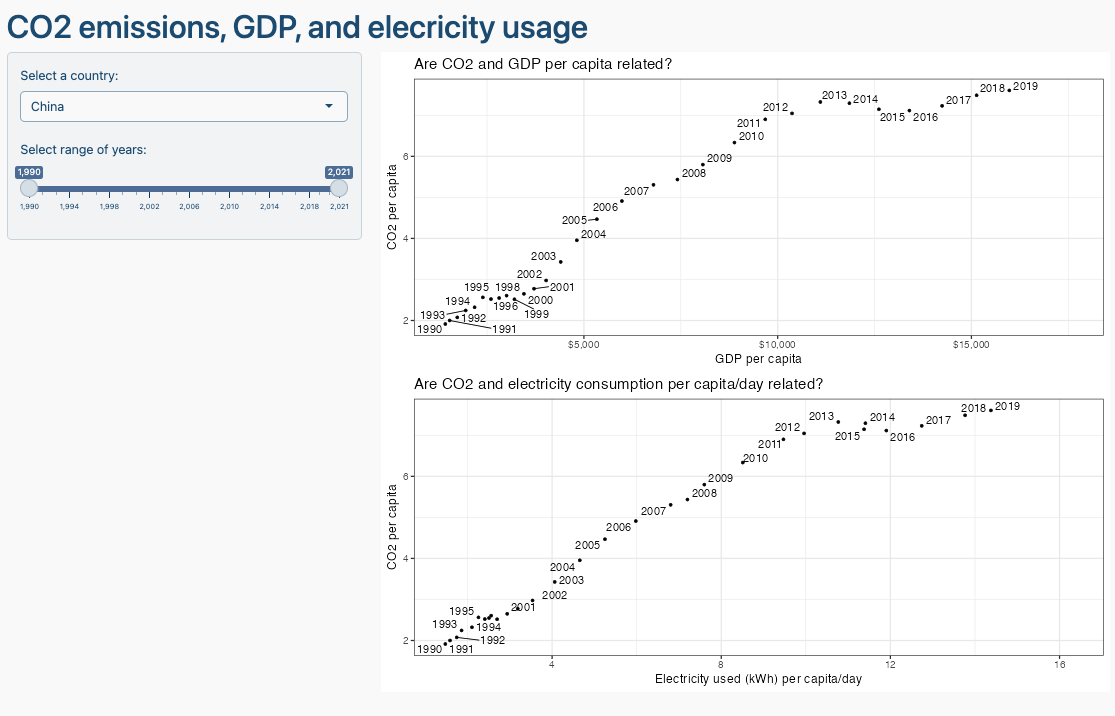
## Warning in max(., na.rm = TRUE): no non-missing arguments to max; returning  
## -Inf

# Specify the file path to save the CSV file  
file\_path <- "data/all\_consulting\_jobs.csv"  
  
# Save the dataframe to a CSV file  
write.csv(all\_consulting\_jobs, file = file\_path, row.names = FALSE)  
  
# Print the message  
cat("Data has been scraped and saved to:", file\_path, "\n")

## Data has been scraped and saved to: data/all\_consulting\_jobs.csv

# Create a shiny app - OPTIONAL

We have already worked with the data on electricity production and usage, GDP/capita and CO2/capita since 1990. You have to create a simple Shiny app, where a user chooses a country from a drop down list and a time interval between 1990 and 2020 and shiny outputs the following



You can use chatGPT to get the basic layout of Shiny app, but you need to adjust the code it gives you. Ask chatGPT to create the Shiny app using the gapminder data and make up similar requests for the inputs/outpus you are thinking of deploying.

#Over here I used the energy information from Homework 2 to create the shiny app. Therefore, I took this code below from my previous homework. I didn't use the gapminder dataset  
  
# Download electricity data  
url <- "https://nyc3.digitaloceanspaces.com/owid-public/data/energy/owid-energy-data.csv"  
  
energy <- read\_csv(url) %>%   
 filter(year >= 1990) %>%   
 drop\_na(iso\_code) %>%   
 select(1:3,  
 biofuel = biofuel\_electricity,  
 coal = coal\_electricity,  
 gas = gas\_electricity,  
 hydro = hydro\_electricity,  
 nuclear = nuclear\_electricity,  
 oil = oil\_electricity,  
 other\_renewable = other\_renewable\_exc\_biofuel\_electricity,  
 solar = solar\_electricity,  
 wind = wind\_electricity,   
 electricity\_demand,  
 electricity\_generation,  
 net\_elec\_imports, # Net electricity imports, measured in terawatt-hours  
 energy\_per\_capita, # Primary energy consumption per capita, measured in kilowatt-hours Calculated by Our World in Data based on BP Statistical Review of World Energy and EIA International Energy Data  
 energy\_per\_gdp, # Energy consumption per unit of GDP. This is measured in kilowatt-hours per 2011 international-$.  
 per\_capita\_electricity, # Electricity generation per capita, measured in kilowatt-hours  
 )

## Rows: 21890 Columns: 129  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): country, iso\_code  
## dbl (127): year, population, gdp, biofuel\_cons\_change\_pct, biofuel\_cons\_chan...  
##   
## 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.

# Download data for CO2 emissions per capita  
co2\_percap <- wb\_data(country = "countries\_only",   
 indicator = "EN.ATM.CO2E.PC",   
 start\_date = 1990,   
 end\_date = 2022,  
 return\_wide=FALSE) %>%   
 filter(!is.na(value)) %>%   
 #drop unwanted variables  
 select(-c(unit, obs\_status, footnote, last\_updated)) %>%   
 rename(year = date,  
 co2percap = value)  
  
# Download data for GDP per capita  
gdp\_percap <- wb\_data(country = "countries\_only",   
 indicator = "NY.GDP.PCAP.PP.KD",   
 start\_date = 1990,   
 end\_date = 2022,  
 return\_wide=FALSE) %>%   
 filter(!is.na(value)) %>%   
 #drop unwanted variables  
 select(-c(unit, obs\_status, footnote, last\_updated)) %>%   
 rename(year = date,  
 GDPpercap = value)  
  
energy\_tidy <- energy %>%  
 pivot\_longer(cols = starts\_with(c("biofuel", "coal", "gas", "hydro", "nuclear", "oil", "other\_renewable", "solar", "wind")), #Pivoting the data set to make it long tidy format  
 names\_to = "source",  
 values\_to = "electricity") %>%   
 rename(iso3c = iso\_code) #renaming the iso\_code column to iso3c to merge it later  
  
  
merged\_data <- left\_join(gdp\_percap, co2\_percap, by = c("iso2c","iso3c","country","year")) %>% #Carry out a merge based on the by variables  
 left\_join(energy\_tidy, by = c("iso3c", "year")) #Merge again with the energy\_tidy table based on iso3c and year  
  
country\_data <- merged\_data %>%   
 filter(!is.na(GDPpercap), !is.na(co2percap)) %>%   
 select(year, country.x, GDPpercap,co2percap, energy\_per\_capita)  
  
country\_data <- unique(country\_data, by = c("year","country.x","GDPpercap","co2percap","energy\_per\_capita"))  
  
  
# Define UI  
ui <- fluidPage(  
 titlePanel("CO2 emissions, GDP, and electricity usage"),  
 sidebarLayout(  
 sidebarPanel(  
 selectInput("country", "Choose Country:",  
 choices = unique(country\_data$country.x),  
 selected = "China"),  
 sliderInput("year\_range", "Select Year Range:",  
 min = 1990,  
 max = 2021,  
 value = c(1990, 2021),  
 step = 1)  
 ),  
 mainPanel(  
 plotOutput("scatter\_plot\_gdp"),  
 plotOutput("scatter\_plot\_electricity")  
 )  
 )  
)  
  
# Define server logic  
server <- function(input, output) {  
   
 # Prepare data based on user input  
 data <- reactive({  
 country\_data %>%  
 filter(country.x == input$country,  
 year >= input$year\_range[1],  
 year <= input$year\_range[2]) %>%  
 filter(!is.na(GDPpercap) & !is.na(co2percap) & !is.na(energy\_per\_capita)) %>%  
 select(year, GDPpercap, co2percap, energy\_per\_capita)  
 })  
   
 # Create scatter plot for GDP per capita vs CO2 per capita  
 output$scatter\_plot\_gdp <- renderPlot({  
 country\_data <- data()  
 country\_name <- input$country  
   
 scatter\_plot <- ggplot(data = country\_data, aes(x = GDPpercap, y = co2percap)) +  
 geom\_point() +  
 geom\_text(aes(label = year), vjust = -0.5, hjust = 0.5, size = 3) +  
 labs(  
 title = paste("CO2 per Capita vs GDP per Capita in", country\_name),  
 x = "GDP per Capita",  
 y = "CO2 per Capita"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(size = 8))  
   
 scatter\_plot  
 })  
   
 # Create scatter plot for Energy per capita vs CO2  
 output$scatter\_plot\_electricity <- renderPlot({  
 country\_data <- data()  
 country\_name <- input$country  
   
 scatter\_plot2 <- ggplot(data = country\_data, aes(x = energy\_per\_capita, y = co2percap)) +  
 geom\_point() +  
 geom\_text(aes(label = year), vjust = -0.5, hjust = 0.5, size = 3) +  
 labs(  
 title = paste("Electricity Usage per Capita vs CO2 per Capita in", country\_name),  
 x = "Electricity Usage (kWh) per Capita/Day",  
 y = "CO2 per Capita"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(size = 8))  
   
 scatter\_plot2  
 })  
}  
  
# Run the application  
shinyApp(ui = ui, server = server)

# Deliverables

There is a lot of explanatory text, comments, etc. You do not need these, so delete them and produce a stand-alone document that you could share with someone. Knit the edited and completed R Markdown (Rmd) file as a Word or HTML document (use the “Knit” button at the top of the script editor window) and upload it to Canvas. You must be commiting and pushing your changes to your own Github repo as you go along.

# Details

* Who did you collaborate with: IGNACIO GAING
* Approximately how much time did you spend on this problem set: 20 hours
* What, if anything, gave you the most trouble: Replicating the graphs was the most troublesome. For two charts I asked Kostis to review my code and he shared his answer with me because even though I could write something it didn’t give me the same values as the image. Therefore, after trying for a couple of hours I wrote Kostis in Slack

**Please seek out help when you need it,** and remember the [15-minute rule](https://dsb2023.netlify.app/syllabus/#the-15-minute-rule). You know enough R (and have enough examples of code from class and your readings) to be able to do this. If you get stuck, ask for help from others, post a question on Slack– and remember that I am here to help too!

As a true test to yourself, do you understand the code you submitted and are you able to explain it to someone else?

# Rubric

13/13: Problem set is 100% completed. Every question was attempted and answered, and most answers are correct. Code is well-documented (both self-documented and with additional comments as necessary). Used tidyverse, instead of base R. Graphs and tables are properly labelled. Analysis is clear and easy to follow, either because graphs are labeled clearly or you’ve written additional text to describe how you interpret the output. Multiple Github commits. Work is exceptional. I will not assign these often.

8/13: Problem set is 60–80% complete and most answers are correct. This is the expected level of performance. Solid effort. Hits all the elements. No clear mistakes. Easy to follow (both the code and the output). A few Github commits.

5/13: Problem set is less than 60% complete and/or most answers are incorrect. This indicates that you need to improve next time. I will hopefully not assign these often. Displays minimal effort. Doesn’t complete all components. Code is poorly written and not documented. Uses the same type of plot for each graph, or doesn’t use plots appropriate for the variables being analyzed. No Github commits.