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

Jacqui Cole

2023-06-05

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"

# The database has 7 tables according to the results of running this DBI request.

## 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)

# Read in the relevant tables as dataframes  
  
payments <- dplyr::tbl(sky\_westminster, "payments")  
members <- dplyr::tbl(sky\_westminster, "members")  
parties <- dplyr::tbl(sky\_westminster, "parties")  
  
#calculate which MP has received the most money  
  
payments %>%   
 group\_by(member\_id) %>%  
 summarise(maxvalue = sum(value)) %>%  
 left\_join(members, by = c("member\_id" = "id")) %>% #joint payments and members with the common MP id to resolve the identity of the MPs.  
 arrange(desc(maxvalue)) # arrange dataframe in descending value.

## 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 8]  
## # Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
## # Ordered by: desc(maxvalue)  
## member\_id maxvalue name gender constituency party\_id short\_name status  
## <chr> <dbl> <chr> <chr> <chr> <chr> <chr> <chr>   
## 1 m8 2809765. Theresa May F Maidenhead p4 Mrs May active  
## 2 m1508 2191387. Sir Geoffr… M Torridge an… p4 Sir Geoff… active  
## 3 m1423 1282402 Boris John… M Uxbridge an… p4 Mr Johnson active  
## 4 m4514 799936. Keir Starm… M Holborn and… p15 Mr Starmer active  
## 5 m1211 769373. Andrew Mit… M Sutton Cold… p4 Mr Mitche… active  
## 6 m3958 712321. Fiona Bruce F Congleton p4 Ms Bruce active  
## 7 m14 692438. John Redwo… M Wokingham p4 Mr Redwood active  
## 8 m4483 546043 Rishi Sunak M Richmond (Y… p4 Mr Sunak active  
## 9 m4097 538678. Liz Truss F South West … p4 Ms Truss active  
## 10 m188 441681. Ed Davey M Kingston an… p17 Mr Davey active  
## # ℹ more rows

# The MP to receive the most money is now in the top row of the resulting table, i.e. it is Teresa May who receives £2809765.42.

## 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?

# Read in the relevant tables as dataframes  
  
payments <- dplyr::tbl(sky\_westminster, "payments")  
members <- dplyr::tbl(sky\_westminster, "members")  
  
#First define and apply the range of year restriction  
  
payments %>%  
   
#capture the last 4 chars of the date variable as these contain the year in xxxx format.   
   
 mutate(year = str\_sub(date, -4)) %>%   
   
#specify the year range 2020-2022 (note that the years only go up to 2022 so this is all that is necessary)   
   
 filter(year >= 2020) %>%   
   
#Now calculate the total value of all payments from entity (donor) to member\_ids (MPs). One needs to group and then ungroup so that this does not restrict the code that follows.  
   
 group\_by(entity,member\_id) %>%   
 summarise(total = sum(value)) %>%  
 ungroup %>%  
  
#Now resolve the identity of the member\_id by joining to the dataframe for members which contains the name of the MPs which we can link to their id which is equivalent to member\_id in the dataframe for payments.  
   
 left\_join(members, by = c("member\_id" = "id")) %>%  
  
#Now arrange the values in descending order and collect to that one can see if any donors have given more than 5 percent of the total payments.  
   
 arrange(desc(total)) %>%  
 collect() %>%  
  
#Remove two rogue rows of data that contain errors.   
  
 filter(!str\_detect(entity, 'Country Foods')) %>%  
 filter(!str\_detect(entity, 'George Watson')) %>%  
  
#Calculate the percentage mentioned above.  
   
 mutate(percent=round(100\*total/sum(total),digits=2)) %>%  
   
#Select the columns of interest and show the results in a table.   
  
 select(entity,name,party\_id,total,percent)

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

## # A tibble: 4,090 × 5  
## entity name party\_id total percent  
## <chr> <chr> <chr> <dbl> <dbl>  
## 1 Withers LLP Sir Geoffrey C… p4 1.81e6 5.26  
## 2 Fiona Bruce and Co LLP Fiona Bruce p4 7.12e5 2.06  
## 3 Charles Stanley John Redwood p4 6.75e5 1.96  
## 4 Cambridge Speaker Series Theresa May p4 4.08e5 1.18  
## 5 Centerview Partners LLP Boris Johnson p4 2.78e5 0.81  
## 6 Council of Insurance Agents & Brokers Boris Johnson p4 2.76e5 0.8   
## 7 Hindustan Times Boris Johnson p4 2.62e5 0.76  
## 8 Unite Rebecca Long-B… p15 2.49e5 0.72  
## 9 Emerging Asset Management Sir Bill Wiggin p4 2.32e5 0.67  
## 10 Hutchison Ports Europe Chris Grayling p4 2.24e5 0.65  
## # ℹ 4,080 more rows

# The results in the table reveal that one entity accounts for more than 5 percent of all donations. The entity in question is WithersLLP who donate to the Conservative MP (party\_id = p4), Sir Geoffrey Cox (name) a total value of 1,812,731.81 which is 5.26 percent of the total amount of donations.

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

* How many distinct entities who paid money to MPS are there?

#Read in the relevant data table into a dataframe  
  
payments <- dplyr::tbl(sky\_westminster, "payments")  
  
payments %>%  
 summarise(Unique\_Elements = n\_distinct(entity)) # Now summarise with unique elements

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
## Unique\_Elements  
## <int>  
## 1 2213

payments %>%  
 summarise(Unique\_Elements = n\_distinct(member\_id)) # Now summarise with unique elements

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
## Unique\_Elements  
## <int>  
## 1 595

#These results indicate that there are 2213 distinct entity (donors) and only 595 distinct member\_id (MPs).

* How many (as a number and %) donated to MPs belonging to a single party only?
* # First convert the database tables into dataframes  
    
  payments <- dplyr::tbl(sky\_westminster, "payments")  
  parties <- dplyr::tbl(sky\_westminster, "parties")  
  members <- dplyr::tbl(sky\_westminster, "members")  
    
  # Then stitch together the columns of three dataframes where they are sequentially joined by the id of the MP or party name.   
    
  mempay <- payments %>%  
   left\_join(members, by = c("member\_id" = "id"))   
    
  parmempay <- mempay %>%  
   left\_join(parties, by = c("party\_id" = "id"))  
    
  # Checking the result:  
    
  parmempay %>%  
   glimpse
* ## Rows: ??  
  ## Columns: 23  
  ## Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
  ## $ category <chr> "4. Visits outside the UK", "2. (b) Any other sup…  
  ## $ category\_name <chr> "Gifts and other benefits", "Cash donations", "Gi…  
  ## $ charity <chr> "", "", "", "", "", "", "", "", "", "", "", "", "…  
  ## $ date <chr> "Registered in November 2021", "Registered in Jan…  
  ## $ date\_visited <chr> "Dates of visit: 5-12 November 2021", "", "Dates …  
  ## $ description <chr> "International flights £805.07; accommodation £1,…  
  ## $ destination\_of\_visit <chr> "Accra, Ghana", "", "Kingston, Jamaica", "", "", …  
  ## $ entity <chr> "GUBA Foundation", "Mahir Kilic", "People's Natio…  
  ## $ hours <chr> "", "", "", "", "", "", "", "", "", "", "", "", "…  
  ## $ id <chr> "44a5c7f837d9df230b8c1e7f72eea188", "b9f40bd69ac2…  
  ## $ member\_id <chr> "m172", "m172", "m172", "m172", "m172", "m44", "m…  
  ## $ purpose\_of\_visit <chr> "To participate in the GUBA Foundation Yaa Asante…  
  ## $ value <dbl> 2631.51, 2000.00, 2574.57, 2000.00, 500.00, 1800.…  
  ## $ name.x <chr> "Diane Abbott", "Diane Abbott", "Diane Abbott", "…  
  ## $ gender <chr> "F", "F", "F", "F", "F", "M", "M", "M", "M", "M",…  
  ## $ constituency <chr> "Hackney North and Stoke Newington", "Hackney Nor…  
  ## $ party\_id <chr> "p15", "p15", "p15", "p15", "p15", "p4", "p4", "p…  
  ## $ short\_name <chr> "Ms Abbott", "Ms Abbott", "Ms Abbott", "Ms Abbott…  
  ## $ status <chr> "active", "active", "active", "active", "active",…  
  ## $ abbrev <chr> "Lab", "Lab", "Lab", "Lab", "Lab", "Con", "Con", …  
  ## $ background <chr> "ff0000", "ff0000", "ff0000", "ff0000", "ff0000",…  
  ## $ foreground <chr> "ffffff", "ffffff", "ffffff", "ffffff", "ffffff",…  
  ## $ name.y <chr> "Labour", "Labour", "Labour", "Labour", "Labour",…
* # This puts together the three key columns that need to be compared which are entity and abbrev and member\_id. Now we can calculate how many entity donate to MPs belonging to only a single party:  
  #  
  # First, we determine how many party\_id and abbrev are distinct:  
  #  
  parmempay %>%  
   summarise(Unique\_Elements = n\_distinct(party\_id)) # Now
* ## # Source: SQL [1 x 1]  
  ## # Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
  ## Unique\_Elements  
  ## <int>  
  ## 1 13
* # summarise with unique elements   
    
  parmempay %>%  
   summarise(Unique\_Elements = n\_distinct(member\_id)) # Now
* ## # Source: SQL [1 x 1]  
  ## # Database: sqlite 3.41.2 [C:\Users\jcole\dsb2023\data\sky-westminster-files.db]  
  ## Unique\_Elements  
  ## <int>  
  ## 1 595
* # summarise with unique elements   
    
  #These results show that there are only 13 distinct party\_ids and only 13 distinct parties. Since each member\_id has a particular party\_id, and we expect an MP to be a member of only one party, this makes sense. But this isn't proof about the link between member\_id and party\_id uniqueness as a correlation. It's just an indication. So, the following code and analysis was undertaken to prove this:  
    
  #Read back in my merged dataframe containing all the relevant columns and collect data from the database   
  parmempay %>%  
   collect() %>%  
    
  # Filter out two rows with these two lines of codes. These rows were found to contain errors when running this segment of code without them, and the nature of the code errors uncovered the text entry inconsistencies in these two rows. So, these two lines of code were added to find and remove these two rows:   
    
   filter(!str\_detect(entity, 'Andrew Taylor')) %>%  
   filter(!str\_detect(entity, 'Le Manoir aux Quat')) %>%  
    
  # The three relevant columns were then selected, where their relationships may be deduced counting how many rows they have in common, as this tells us to what extent these variables map to each other with a one-to-one correpondence and are thus distinct, or whether there is multiplicity in intervariable mapping.   
    
   select(entity,member\_id,abbrev) %>%  
   group\_by(entity,member\_id,abbrev) %>%  
    
  # First, we group together all three variables and by just looking at the size of the tibble i.e. y where tibble has dimensions of y x 3, y reveals the number of possible combinations of each variable. Considering all three together gives us a base line (y = 4090). We then have to ungroup this group in order to do the next relationship test and continuing to add code that groups other variables:  
    
   ungroup %>%  
    
  # I now group member\_id and abbrev to check the indication made above that all MPs will be loyal to a single party. We know from the previous question that there are 595 distinct MPs in this dataframe so if the possible combinations (mapping) of MPs and party types is the same (i.e. 595) then they have a one-to-one correspondence, and this proofs the indication above. Let's see - we now group - check the dimension, y, of the tibble that the grouping generates via this R code and then compares it against the number of distinct 595 MPs:  
    
   group\_by(member\_id,abbrev) %>% #proof that MPs stick to one party  
   ungroup %>%  
    
  # The full analysis is given below but this grouping shows all MPs to be loyal to their party, as we had suspected. These variables were then ungrouped so that we could start a new grouping.  
    
  # Entity (donors) and abbrev (party types) are now grouped, to see how many combinations exists between them.  
    
   group\_by(entity,abbrev) %>% #this produces 2467 relationships.  
   ungroup %>% # I then ungroup this group so that I can make another  
    
  # We already saw in the solution to the previous question that there are a total of 2213 distinct entity values (i.e. distinct donors). We lost a few rows in filtering out two cases with data entry inconsistencies. So we just check this value again to be sure:  
    
   group\_by(entity)
* ## # A tibble: 10,537 × 3  
  ## # Groups: entity [2,211]  
  ## entity member\_id abbrev  
  ## <chr> <chr> <chr>   
  ## 1 GUBA Foundation m172 Lab   
  ## 2 Mahir Kilic m172 Lab   
  ## 3 People's National Party (PNP) Women's Movement m172 Lab   
  ## 4 National Union of Rail, Maritime and Transport Workers (RMT) m172 Lab   
  ## 5 The Football Association Premier League m172 Lab   
  ## 6 1912 Club m44 Con   
  ## 7 97 Dining Club m44 Con   
  ## 8 Ministry of Foreign Affairs, Qatar m44 Con   
  ## 9 Qatar Racing and Equestrian Club (Qatar Ministry of Sports … m44 Con   
  ## 10 Catalyst Presents Foundation m104 Con   
  ## # ℹ 10,527 more rows
* #  
  # This all checked out as we expect: y = 2211 i.e. as per the solution to the previous question but two less because I have filtered out two rows in the above analysis.  
  #   
  # The overall data analysis provides these findings and conclusions:  
  #  
  # The results of these groupings and monitoring the associated y values in the y x 3 dimensions of the tibble of variables that have been grouped in different ways, shows that there are:  
  #  
  # 10,537 total rows of entity, member\_id, abbrev  
  # 4090 distinct combinations of entity, member\_id, abbrev  
  # 595 distinct combinations of member\_id and abbrev  
  # 2467 entity distinct combinations of abbrev rows  
  # 2211 rows of entity values (in the previous question, we determined this value to be 2213 which is consistent with this new value when we consider that we removed two rows of data from the dataframe parmempay to overcome a few errors in data entry).  
  #  
  # These all show that all MPs are entirely loyal to one party (there are 595 distinct MPs and 595 combinations of MPs and Party categories, so this proves that there is a one-to-one mapping between them).  
  #  
  # But are donors to MPs as loyal to a party?  
  #  
  # It would seem that they are largely so, but not entirely so, because there are more mappings of donors (entity) to party names (abbrev) than there are distinct donors (entity) to the tune of:  
  #2467-2211 = 256 (regarding the two entries that were filtered out owing to errors, I checked manually the cases of Andrew Taylor and Le Manoir aux Quat and they were found to be loyal to one party).  
  #  
  # The numer of entity that donated to a single party can thus be calculated to be:  
  #   
  # 2213  
  #  
  # which is a percentage:  
  #  
  # (2213/2467)\*100 = 89.7 percent of donors are loyal to a single party in making their donations.

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

# Convert relevant data tables into dataframes  
  
party\_donations <- dplyr::tbl(sky\_westminster, "party\_donations")  
parties <- dplyr::tbl(sky\_westminster, "parties")  
  
#Determine the total number of donations and place in descending order  
  
total\_party\_donations <- party\_donations %>%   
group\_by(date, party\_id) %>%   
summarise(total\_donations = sum(value, na.rm = TRUE)) %>%   
ungroup() %>%  
arrange(desc(total\_donations)) %>%   
   
# join parties to party\_donations in order to resolve identities of the parties with the donors, and then collect the data   
   
left\_join(parties, by = c("party\_id"="id")) %>%   
collect() %>%   
   
# Resolve the value of the year from the date variable that is given  
   
mutate(date = lubridate::ymd(date), #lubridate doesn't work on DB directly-- need to collect first   
year = year(date)) %>%   
   
# Determine the proportion of the total number of donations per year with respect to the total value of the donations and arrange the results in descending order  
   
group\_by(year, name) %>%   
summarise(total\_year\_donations = sum(total\_donations)) %>%   
mutate(prop = total\_year\_donations / sum(total\_year\_donations)) %>%   
ungroup() %>%  
arrange(desc(prop))

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

## Warning: ORDER BY is ignored in subqueries without LIMIT  
## ℹ Do you need to move arrange() later in the pipeline or use window\_order() instead?

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

# Display the resulting dataframe  
  
total\_party\_donations

## # A tibble: 28 × 4  
## year name total\_year\_donations prop  
## <dbl> <chr> <dbl> <dbl>  
## 1 2020 Conservative 42770782. 0.612   
## 2 2021 Conservative 17718212. 0.594   
## 3 2022 Conservative 15568476. 0.559   
## 4 2022 Labour 9460879. 0.340   
## 5 2021 Labour 9493978. 0.318   
## 6 2020 Labour 13539803. 0.194   
## 7 2020 Liberal Democrats 12717295. 0.182   
## 8 2022 Liberal Democrats 1727152. 0.0620  
## 9 2021 Sinn Féin 822944 0.0276  
## 10 2021 Liberal Democrats 700398. 0.0235  
## # ℹ 18 more rows

#This table shows that the conservative (C) party obtains the dominant portion of the donations, with labour (Lab) second and, depending on the year, the liberal democrats (LD), Sinn Fein (SF) , the Scottish National Party (SNP) are third with the combination of other parties (O) making up the rest. By year, this summarises as:  
  
#2020: C (61%), Lab (19%), LD (18%), O (2%)  
#2021: C (59%), Lab (32%), SF (3%), LD (2%), SNP (2%), O (2%)  
#2022: C (56%), Lab (34%), LD (6%), O (4%).  
  
#Thus, the conservative party progressively lost their donations from 2020-2022 while the labour party gained in this market share, while the liberal democrats lost nearly all support, only recovering slightly in 2022.

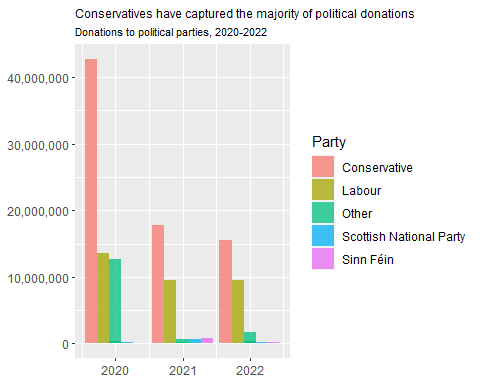
I would like you to write code that generates the following table.

#Now reordering the table in the manner requested by the question below to match that of total\_donations\_table.png in the images folder:  
  
total\_party\_donations[order(total\_party\_donations$year,total\_party\_donations$name), ]

## # A tibble: 28 × 4  
## year name total\_year\_donations prop  
## <dbl> <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 18 more rows

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

library(scales) # a library for the graphing options  
total\_party\_donations %>% #read in the dataframe  
 arrange(year) %>% #arrange data such that year is in ascending order  
  
# This classifies the main parties and groups the minor ones into other  
  
 mutate(recode\_name = case\_when(  
 name %in% c("Conservative") ~ "Conservative",  
 name %in% c("Labour") ~ "Labour",  
 name %in% c("Liberal Demoncrats") ~ "Liberal Democrats",  
 name %in% c("Sinn Féin") ~ "Sinn Féin",  
 name %in% c("Scottish National Party") ~ "Scottish National Party",  
 TRUE ~ "Other"  
 )) %>%   
  
# I now plot the data as side-by-side bar chart whose magnitude is in decreasing order and with titles and labels to aid visualisation:  
   
 ggplot(mapping = aes(year, total\_year\_donations, fill = recode\_name)) + geom\_bar(stat = "identity", position = position\_dodge(), alpha = 0.75) + labs(x = NULL, y = NULL, title = "Conservatives have captured the majority of political donations", subtitle = "Donations to political parties, 2020-2022", fill = "Party") + theme(plot.title = element\_text(size=9),plot.subtitle = element\_text(size=8)) + scale\_y\_continuous(labels = comma)



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?

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

dbDisconnect(sky\_westminster) #disconnecting from the database

# 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

## 0.19 sec elapsed

Can you query the database and replicate the following plot?

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

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()

Each county belongs in seix 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?

# 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) # checking the the website allows us to scrape data  
paths\_allowed("https://www.opensecrets.org")

## [1] TRUE

# Reading in the website URL and creating an associated dataframe:  
  
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"  
    
   tables <- base\_url %>%# get tables that exist on url   
   read\_html() %>%   
   html\_nodes(css="table") %>% # this will isolate all tables on page   
   html\_table() # Parse an html table into a dataframe table  
    
   ## Use `tables[[1]]` to parse the first table (for 2022)  
  contributions <- tables[[1]] %>%   
  janitor::clean\_names() #%>% #default option is snake\_case  
  #data.frame(years = years(1:215))  
  #mutate(newCol(1:215) = year)  
   #  
  #This converts the table headers into snake\_case e.g. pac\_name\_affiliate  
  #  
  #Add a year column (for later)  
  #  
  #mutate(year=year)
* 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 <- contributions %>%  
 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”.
* 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?
* Construct a vector called urls that contains the URLs for each webpage that contains information on foreign-connected PAC contributions for a given year.
* Map the scrape\_pac() function over urls in a way that will result in a data frame called contributions\_all.
* Write the data frame to a csv file called contributions-all.csv in the data folder.
* # I did this question after the consultancyjobs question and so  
  # I did my initial code developments that built up this type of   
  # problem therein; see my working in that question that led to the  
  # ultimate function. I have now done this question in a similar  
  # style, so I have not repeated the description of the initial   
  # development work, although those developments and testing are  
  # implicit to the final code below. Thereby:  
  #  
  # turn the code into a function  
  scrape\_pac <- function(year) {  
    
   base\_url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/"  
   url <- str\_c(base\_url,year)  
  # listings\_html <- read\_html(url)   
    
  tables <- url %>%# get tables that exist on url   
  read\_html() %>%   
  html\_nodes(css="table") %>% # this will isolate all tables on page   
  html\_table() # Parse an html table into a dataframe table  
    
  contributions <- tables[[1]] %>%   
  janitor::clean\_names() %>% #default option is snake\_case  
  mutate(year=year)  
    
  # clean country/parent co and contributions   
  contributions <- contributions %>%  
   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)  
   )  
    
    
  return(contributions)  
  }  
    
  #Now that I have this function, I define year elements and map this function to the desired output  
    
  year <- seq(from=2000, to=2022, by=2)   
  contributions\_all <- map\_df(year,scrape\_pac)  
    
  write.csv(contributions\_all, "contributions\_all.csv", row.names = FALSE)

# 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?

## www.consultancy.uk

## [1] TRUE

base\_url <- "https://www.consultancy.uk/jobs/page/1"  
  
listings\_html <- base\_url %>%  
 read\_html()

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?

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.

#  
# NB: A practical point: I progressively work through the development of the above questions throughout this code block.   
#  
#The resulting code that defines the final function and so forth is provided at the end of this code block.  
#  
#  
#  
# Initial test work to develop the underpinning code ideas towards the function development:- working through the first two questions:  
  
#This function will 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?  
  
# I tackle the above two questions in one set of code, whereby I input all 5 pages of the website tabular information sequentially:  
  
base\_url1 <- "https://www.consultancy.uk/jobs/page/1"  
base\_url2 <- "https://www.consultancy.uk/jobs/page/2"  
base\_url3 <- "https://www.consultancy.uk/jobs/page/3"  
base\_url4 <- "https://www.consultancy.uk/jobs/page/4"  
base\_url5 <- "https://www.consultancy.uk/jobs/page/5"  
  
  
# I then isolate all (in this case 1) table on each of the five pages and parse each table into a dataframe  
  
table1 <- base\_url1 %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe  
  
table2 <- base\_url2 %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe  
  
table3 <- base\_url3 %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe  
  
table4 <- base\_url4 %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe  
  
table5 <- base\_url5 %>%  
 read\_html() %>%  
 html\_nodes(css="table") %>% # this will isolate all tables on page  
 html\_table() # Parse an html table into a dataframe  
  
#This produces 5 dataframes of the five tables from the 5 pages:  
  
table1

## [[1]]  
## # A tibble: 45 × 4  
## Job Firm `Functional area` Type   
## <chr> <chr> <chr> <chr>  
## 1 "Intermediate Quantity Surveyor\nPanoptic Cons… Pano… "Project Managem… Job   
## 2 "Management Consultants\nCollinson Grant" Coll… "Process Managem… Job   
## 3 "Data Engineer\nValcon" Valc… "Data Science\n+… Job   
## 4 "Senior Strategist\nThe Upside" The … "Strategy\n+2\nM… Job   
## 5 "Director Client Services - Life Sciences\nGen… Geni… "Strategy\n+2\nM… Job   
## 6 "Independent Consultant \ndss+" dss+ "Sustainability" Job   
## 7 "Experienced Hire\nFairgrove Partners" Fair… "Strategy\n+1\nM… Job   
## 8 "Senior 3D/Motion Designer\nYonder Consulting" Yond… "Marketing\n+1\n… Job   
## 9 "Consultant Roles (at all levels) – IT Advisor… Maso… "Digital\n+4\nIT… Job   
## 10 "Senior Analyst\nCIL Management Consultants" CIL … "Strategy\n+1\nD… Job   
## # ℹ 35 more rows

table2

## [[1]]  
## # A tibble: 45 × 4  
## Job Firm `Functional area` Type   
## <chr> <chr> <chr> <chr>  
## 1 "Marketing Executive/Manager\nPanoptic Consult… Pano… "Marketing" Job   
## 2 "Internship Business Development UK\nCOMATCH" COMA… "Sales" Inte…  
## 3 "Principal consultant\nChange Management Group" Chan… "Strategy\n+2\nP… Job   
## 4 "Independent Consultants\nFairgrove Partners" Fair… "Strategy\n+1\nM… Job   
## 5 "Director, Business Intelligence | Forensic & … FTI … "Forensic & Liti… Job   
## 6 "Technology and Architecture Roles \nCoeus Con… Coeu… "Digital\n+4\nIT… Job   
## 7 "Customer Experience Specialist\nDigital Power" Digi… "CRM" Job   
## 8 "Supply Chain Operations Analyst\nBearingPoint" Bear… "Supply Chain" Job   
## 9 "Consultant Data Governance & Data Quality\nVa… Valc… "Data Science" Job   
## 10 "Manager and Senior Manager\nProcura Consultin… Proc… "Project Managem… Job   
## # ℹ 35 more rows

table3

## [[1]]  
## # A tibble: 45 × 4  
## Job Firm `Functional area` Type   
## <chr> <chr> <chr> <chr>  
## 1 "Deals | Forensic Accounting - Senior Manager\… PwC "Forensic & Liti… Job   
## 2 "Programme Director - Energy & Utilities\nCapg… Capg… "Management\n+5\… Job   
## 3 "Cyber Security Consultant\nBearingPoint" Bear… "Cyber Security" Job   
## 4 "Technical Web Analyst\nDigital Power" Digi… "Digital" Job   
## 5 "Managing Consultant - Networks/Telecoms\nPA C… PA C… "Unknown" Job   
## 6 "Senior Consultant - Cyber Security\nThreeTwoF… Thre… "Cyber Security" Job   
## 7 "Innovation Consultants: All levels\nAyming" Aymi… "Finance\n+2\nDa… Job   
## 8 "PH-4762; Senior Project Manager - Sales Expan… B2E … "Sales" Job   
## 9 "Senior Consultant, Mergers & Acquisitions\nWe… West… "Mergers & Acqui… Job   
## 10 "Transactions Tax, Senior Consultant \nFTI Con… FTI … "Management\n+5\… Job   
## # ℹ 35 more rows

table4

## [[1]]  
## # A tibble: 45 × 4  
## Job Firm `Functional area` Type   
## <chr> <chr> <chr> <chr>  
## 1 "Resource Co-ordinator\nPA Consulting" PA C… "Unknown" Job   
## 2 "MM-4817; Business Analysts, Fixed Voice IMS &… B2E … "Mobile & Apps" Job   
## 3 "Consultant / Senior Consultant - DevOps Archi… Capg… "IT Strategy\n+1… Job   
## 4 "Physicist - Defence & Security\nPA Consulting" PA C… "Cyber Security" Job   
## 5 "Senior Director | Analytics | Strategic Commu… FTI … "Management\n+6\… Job   
## 6 "CyberArk Architect - Manager\nPwC" PwC "Unknown" Job   
## 7 "Full Stack Engineer\nPA Consulting" PA C… "Unknown" Job   
## 8 "Cyber | Azure IAM architect - Senior Manager\… PwC "Cyber Security" Job   
## 9 "Senior Consultant, Managed Document Review | … FTI … "IT Strategy" Job   
## 10 "Cloud Economics - Managing Consultant / Senio… Capg… "Cloud" Job   
## # ℹ 35 more rows

table5

## [[1]]  
## # A tibble: 45 × 4  
## Job Firm `Functional area` Type   
## <chr> <chr> <chr> <chr>  
## 1 "DevOps Engineer Consultant (London/Manchester… Capg… "Software" Job   
## 2 "Cloud Operating Model - Consultant / Senior C… Capg… "Cloud" Job   
## 3 "Financial Services Advisory | Manager | Banki… PwC "Unknown" Job   
## 4 "Marketing Executive | Forensic & Litigation C… FTI … "Marketing\n+1\n… Job   
## 5 "PR Manager\nPA Consulting" PA C… "Unknown" Job   
## 6 "Cloud Strategy - Consultant / Senior Consulta… Capg… "Strategy\n+1\nC… Job   
## 7 "Executive Assistant, Business Transformation,… FTI … "Unknown" Job   
## 8 "Financial Services Advisory | Manager | Banki… PwC "Unknown" Job   
## 9 "Lead Software Engineer\nPA Consulting" PA C… "Software" Job   
## 10 "Cloud Economics - Consultant / Senior Consult… Capg… "Cloud" Job   
## # ℹ 35 more rows

# Now I develop the more sophisticated approach with the next two questions:  
  
#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?  
  
#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  
  
base\_url <- "https://www.consultancy.uk/jobs/page/"  
page = c(1,2,3,4,5)  
url <- str\_c(base\_url, page)  
url

## [1] "https://www.consultancy.uk/jobs/page/1"  
## [2] "https://www.consultancy.uk/jobs/page/2"  
## [3] "https://www.consultancy.uk/jobs/page/3"  
## [4] "https://www.consultancy.uk/jobs/page/4"  
## [5] "https://www.consultancy.uk/jobs/page/5"

job <- listings\_html %>%   
html\_nodes(css = "span.title") %>%   
html\_text2()   
  
firm <- listings\_html %>%   
html\_nodes(css = ".hide-phone .row-link") %>%   
html\_text2()  
  
functionality <- listings\_html %>%   
html\_nodes(css = ".hide-tablet-and-less .row-link") %>%   
html\_text2()  
  
jobtype <- listings\_html %>%   
html\_nodes(css = ".hide-tablet-landscape .row-link") %>%   
html\_text2()  
  
job

## [1] "Intermediate Quantity Surveyor"   
## [2] "Management Consultants"   
## [3] "Data Engineer"   
## [4] "Senior Strategist"   
## [5] "Director Client Services - Life Sciences"   
## [6] "Independent Consultant"   
## [7] "Experienced Hire"   
## [8] "Senior 3D/Motion Designer"   
## [9] "Consultant Roles (at all levels) – IT Advisory"   
## [10] "Senior Analyst"   
## [11] "Internship / Work Placement"   
## [12] "PH-4804; Test Automation Manager, Python / Azure"   
## [13] "Senior Infrastructure & Cloud Services Advisor"   
## [14] "Analyst, satellite and space markets"   
## [15] "Analyst"   
## [16] "Internships"   
## [17] "Consultant Treasury Technology"   
## [18] "PMO Lead"   
## [19] "HR Manager"   
## [20] "Data Scientist"   
## [21] "Business Analyst"   
## [22] "Analyst"   
## [23] "Sourcing & Commercial Role"   
## [24] "Senior Consultant - Local Government Strategy"   
## [25] "Senior Business Development Manager"   
## [26] "M&A Managing Partner UK"   
## [27] "Manager - Technology"   
## [28] "Associate Consultant"   
## [29] ""   
## [30] "AWS Principal Architect"   
## [31] "Business Development Manager UK"   
## [32] "Principal Consultants"   
## [33] "Consultants and Senior Consultants"   
## [34] "Healthcare consultant"   
## [35] "Associate"   
## [36] "Senior Consultant | Energy & Natural Resources | Strategic Communications"  
## [37] "Strategy& - Strategy Senior Associate"   
## [38] "Director - Supply Chain Strategy & Transformation"   
## [39] "GCP Cloud Engineer"   
## [40] "Manufacturing and construction consultant"   
## [41] "Marketing Executive"   
## [42] "Marketing Assistant"   
## [43] "Business Development Manager – R&D Incentives"   
## [44] "Manager"   
## [45] "Consulting Roles"

firm

## [1] "Panoptic Consultancy Group" "Collinson Grant"   
## [3] "Valcon" "The Upside"   
## [5] "Genioo" "dss+"   
## [7] "Fairgrove Partners" "Yonder Consulting"   
## [9] "Mason Advisory" "CIL Management Consultants"  
## [11] "Skarbek Associates" "B2E Consulting"   
## [13] "West Monroe" "Analysys Mason"   
## [15] "Enfuse Group" "Simon-Kucher"   
## [17] "Zanders" "ThreeTwoFour"   
## [19] "BearingPoint" "Digital Power"   
## [21] "Humatica" "Change Management Group"   
## [23] "Coeus Consulting" "Campbell Tickell"   
## [25] "Ayming" "Marktlink"   
## [27] "First Consulting" "Bain & Company"   
## [29] "Alvarez & Marsal" "PA Consulting"   
## [31] "COMATCH" "Q5"   
## [33] "Procura Consulting" "Develop Consulting"   
## [35] "McKinsey & Company" "FTI Consulting"   
## [37] "PwC" "Capgemini Invent"   
## [39] "PA Consulting" "Develop Consulting"   
## [41] "Genioo" "ThreeTwoFour"   
## [43] "Ayming" "CIL Management Consultants"  
## [45] "Q5"

functionality

## [1] "Project Management" "Process Management" "Data Science"   
## [4] "Strategy" "Strategy" "Sustainability"   
## [7] "Strategy" "Marketing" "Digital"   
## [10] "Strategy" "Marketing" "Unknown"   
## [13] "Cloud" "Strategy" "Digital"   
## [16] "Pricing" "Corporate Finance" "Project Management"  
## [19] "Human Resources" "Data Science" "Data Science"   
## [22] "Process Management" "Project Management" "Strategy"   
## [25] "Sales" "Unknown" "Mobile & Apps"   
## [28] "Strategy" "Management" "Unknown"   
## [31] "Sales" "Strategy" "Project Management"  
## [34] "Lean & SixSigma" "Strategy" "Unknown"   
## [37] "Strategy" "Strategy" "Cloud"   
## [40] "Lean & SixSigma" "Marketing" "Marketing"   
## [43] "Sales" "Strategy" "Strategy"

jobtype

## [1] "Job" "Job" "Job" "Job" "Job"   
## [6] "Job" "Job" "Job" "Job" "Job"   
## [11] "Internship" "Job" "Job" "Job" "Job"   
## [16] "Internship" "Job" "Job" "Job" "Job"   
## [21] "Job" "Job" "Job" "Job" "Job"   
## [26] "Job" "Job" "Job" "Job" "Job"   
## [31] "Job" "Job" "Job" "Job" "Job"   
## [36] "Job" "Job" "Job" "Job" "Job"   
## [41] "Job" "Job" "Job" "Job" "Job"

# This has provided me a way to collect the five pages into one 'url'  
  
# Now I try to work on the creation of the actual function cf the question:  
  
#Map the scrape\_jobs() function over pages in a way that will result in a data frame called all\_consulting\_jobs.  
  
  
#############################################################  
  
# turn our code into a function  
get\_listings <- function(page) {  
   
 base\_url <- "https://www.consultancy.uk/jobs/page/"  
 url <- str\_c(base\_url, page)  
 listings\_html <- read\_html(url)   
#get\_jobs <- base\_url %>%  
# read\_html() %>%  
# html\_nodes(css="table") %>% # this will isolate all tables on page  
# html\_table() # Parse an html table into a dataframe  
  
job <- listings\_html %>%   
html\_nodes(css = "span.title") %>%   
html\_text2()   
  
firm <- listings\_html %>%   
html\_nodes(css = ".hide-phone .row-link") %>%   
html\_text2()  
  
functionality <- listings\_html %>%   
html\_nodes(css = ".hide-tablet-and-less .row-link") %>%   
html\_text2()  
  
jobtype <- listings\_html %>%   
html\_nodes(css = ".hide-tablet-landscape .row-link") %>%   
html\_text2()  
   
jobs\_df <- tibble( # the first 'job' is the variable name, the second job after the equal sign is what you got from the CSS selector  
job = job,   
firm = firm,   
functionality = functionality,   
 jobtype = jobtype)   
  
# we now have the tibble, so this is what the function should return  
return(jobs\_df)  
  
}  
  
# iterate across pages 1-5  
page <- 1:5  
  
# use purrr::map\_df(), as we want the output to be a dataframe  
all\_consulting\_jobs <- map\_df(page, get\_listings)  
  
# This works!! And produces a dataframe called all\_consulting\_jobs  
#  
# Now I just need to write the output in a csv format cf: the question:  
  
#Write the data frame to a csv file called all\_consulting\_jobs.csv in the data folder.  
  
write.csv(all\_consulting\_jobs, "all\_consulting\_jobs.csv", row.names = FALSE)

# 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.

# As noted above this question is optional. I have thus not included a solution in this homework document. I do plan to create a shiny app and present this in another form. I found the following tutorial on line that may be helpful to learn what is required of this shiny app creation:  
# https://mastering-shiny.org/basic-app.html  
#

# 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: No-one although I asked the course professor a handful of questions once I felt that I was close with the final answer but was seemingly missing a small trick or there was a minor issue; or in one case, where I was totally stuck and needed freeing with a bit of code to understand the issue and catalyse me to propel forward.
* Approximately how much time did you spend on this problem set: 2.5 days
* What, if anything, gave you the most trouble: I found it hard to work out when to collect and, additionally, where specifically in the code one should place a collect(). I couldn’t open the covid data on my laptop even when using arrow. Thus, I could not answer this question. I found it difficult to understand what exactly was being asked by the skywestminster questions; once I could comprehend the actual question, I could manage the coding, albeit I initially struggled with the group and ungroup tricks which are important in the earlier questions about this database. That said, I now feel that I understand the grouping and ungrouping concepts within a stream of code and so I have learnt well from this exercise on this account.

**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.