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

Saagar Hemrajani

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?

westminster\_tbls <- DBI::dbListTables(sky\_westminster)  
  
minister\_db <- setNames(  
 lapply(westminster\_tbls, function(table) {  
 dplyr::tbl(sky\_westminster, table)  
 }),  
 westminster\_tbls  
)

## 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)

max\_donations <- minister\_db$payments %>%   
 left\_join(minister\_db$members, by = c("member\_id" = "id")) %>%   
 group\_by(name) %>%   
 summarize(sum\_donations = sum(value)) %>%   
 arrange(desc(sum\_donations)) %>%   
 slice\_max(sum\_donations)  
  
collect(max\_donations)

## 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.

## 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?

## # A tibble: 1 × 2  
## name sum\_donations  
## <chr> <dbl>  
## 1 Theresa May 2809765.

## 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?

minister\_db$payments

## # Source: table<payments> [?? x 13]  
## # Database: sqlite 3.41.2 [C:\Users\saaga\Dropbox\Saagar@LBS\2023\_Summer\DSForBusiness\mycode\data\sky-westminster-files.db]  
## category category\_name charity date date\_visited description  
## <chr> <chr> <chr> <chr> <chr> <chr>   
## 1 4. Visits outside the UK Gifts and ot… "" Regi… "Dates of v… Internatio…  
## 2 2. (b) Any other suppor… Cash donatio… "" Regi… "" Use of a h…  
## 3 4. Visits outside the UK Gifts and ot… "" Regi… "Dates of v… Flights £1…  
## 4 2. (a) Support linked t… Cash donatio… "" Regi… "" 2000   
## 5 3. Gifts, benefits and … Gifts and ot… "" Regi… "" Two box ti…  
## 6 2. (b) Any other suppor… Cash donatio… "" Regi… "" 1800   
## 7 2. (b) Any other suppor… Cash donatio… "" Regi… "" 10000   
## 8 4. Visits outside the UK Gifts and ot… "" Regi… "Dates of v… Flights an…  
## 9 5. Gifts and benefits f… Gifts and ot… "" Regi… "" Guest at Q…  
## 10 4. Visits outside the UK Gifts and ot… "" Regi… "Dates of v… Flights, a…  
## # ℹ more rows  
## # ℹ 7 more variables: destination\_of\_visit <chr>, entity <chr>, hours <chr>,  
## # id <chr>, member\_id <chr>, purpose\_of\_visit <chr>, value <dbl>

total\_payments\_sum <- minister\_db$payments %>%  
 filter(str\_sub(date, start = -4) %in% c('2020','2021','2022')) %>%   
 summarise(tot = sum(value, na.rm = TRUE)) %>%   
 pull(tot)  
  
minister\_db$payments %>%   
 filter(str\_sub(date, start = -4) %in% c('2020','2021','2022')) %>%   
 group\_by(entity) %>%  
 summarize(pct\_donation = sum(value) / total\_payments\_sum \*100) %>%   
 filter(pct\_donation > 5) %>%   
 arrange(desc(pct\_donation))

## # Source: SQL [1 x 2]  
## # Database: sqlite 3.41.2 [C:\Users\saaga\Dropbox\Saagar@LBS\2023\_Summer\DSForBusiness\mycode\data\sky-westminster-files.db]  
## # Ordered by: desc(pct\_donation)  
## entity pct\_donation  
## <chr> <dbl>  
## 1 Withers LLP 5.34

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

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

minister\_db$payments %>%   
 summarize(distinct\_entities = n\_distinct(entity))

## # Source: SQL [1 x 1]  
## # Database: sqlite 3.41.2 [C:\Users\saaga\Dropbox\Saagar@LBS\2023\_Summer\DSForBusiness\mycode\data\sky-westminster-files.db]  
## distinct\_entities  
## <int>  
## 1 2213

* How many (as a number and %) donated to MPs belonging to a single party only?

donation\_counts <- minister\_db$payments %>%   
 left\_join(minister\_db$members, by = c("member\_id" = "id")) %>%   
 group\_by(entity) %>%   
 summarize(distinct\_parties = n\_distinct(party\_id))  
  
donation\_counts\_df <- collect(donation\_counts)  
  
  
donation\_counts\_df %>%   
 group\_by(distinct\_parties) %>%   
 summarize(count = sum(distinct\_parties), pct = sum(distinct\_parties)/nrow(donation\_counts\_df) \*100)

## # A tibble: 7 × 3  
## distinct\_parties count pct  
## <int> <int> <dbl>  
## 1 1 2036 92.0   
## 2 2 256 11.6   
## 3 3 90 4.07   
## 4 4 52 2.35   
## 5 5 15 0.678  
## 6 6 12 0.542  
## 7 8 8 0.362

## 92% of entities donate to only 1 single party

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

donations\_per\_year <- minister\_db$party\_donations %>%   
 mutate(year\_str = str\_sub(date, end = 4)) %>%   
 filter(year\_str %in% c('2020','2021','2022')) %>%   
 group\_by(party\_id, year\_str) %>%   
 summarize(total\_year\_donations = sum(value))  
  
collect(donations\_per\_year) %>%   
 group\_by(year\_str) %>%  
 arrange(desc(total\_year\_donations)) %>%  
 slice(1)

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

## # A tibble: 3 × 3  
## # Groups: year\_str [3]  
## party\_id year\_str total\_year\_donations  
## <chr> <chr> <dbl>  
## 1 p4 2020 42770782.  
## 2 p4 2021 17718212.  
## 3 p4 2022 15568476.

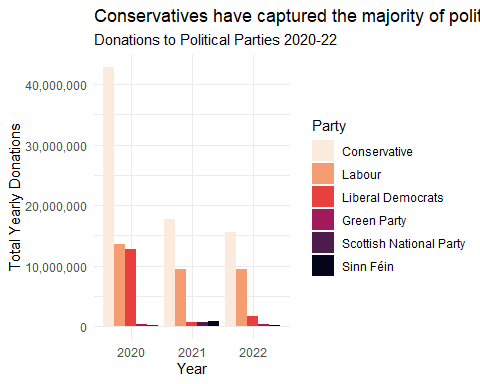
# P4 had the highest donations for each year

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

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

## # A tibble: 28 × 6  
## # Groups: party\_id [10]  
## party\_id year\_str total\_year\_donations all\_party\_donations prop name   
## <chr> <chr> <dbl> <dbl> <dbl> <chr>   
## 1 p1 2020 105000 69941125. 0.00150 Alliance   
## 2 p1 2021 42500 29834986. 0.00142 Alliance   
## 3 p1 2022 180600 27848092. 0.00649 Alliance   
## 4 p1034 2021 53559. 29834986. 0.00180 Alba Party  
## 5 p1034 2022 50000 27848092. 0.00180 Alba Party  
## 6 p15 2020 13539803. 69941125. 0.194 Labour   
## 7 p15 2021 9493978. 29834986. 0.318 Labour   
## 8 p15 2022 9460879. 27848092. 0.340 Labour   
## 9 p17 2020 12717295. 69941125. 0.182 Liberal D…  
## 10 p17 2021 700398. 29834986. 0.0235 Liberal D…  
## # ℹ 18 more rows

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



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.

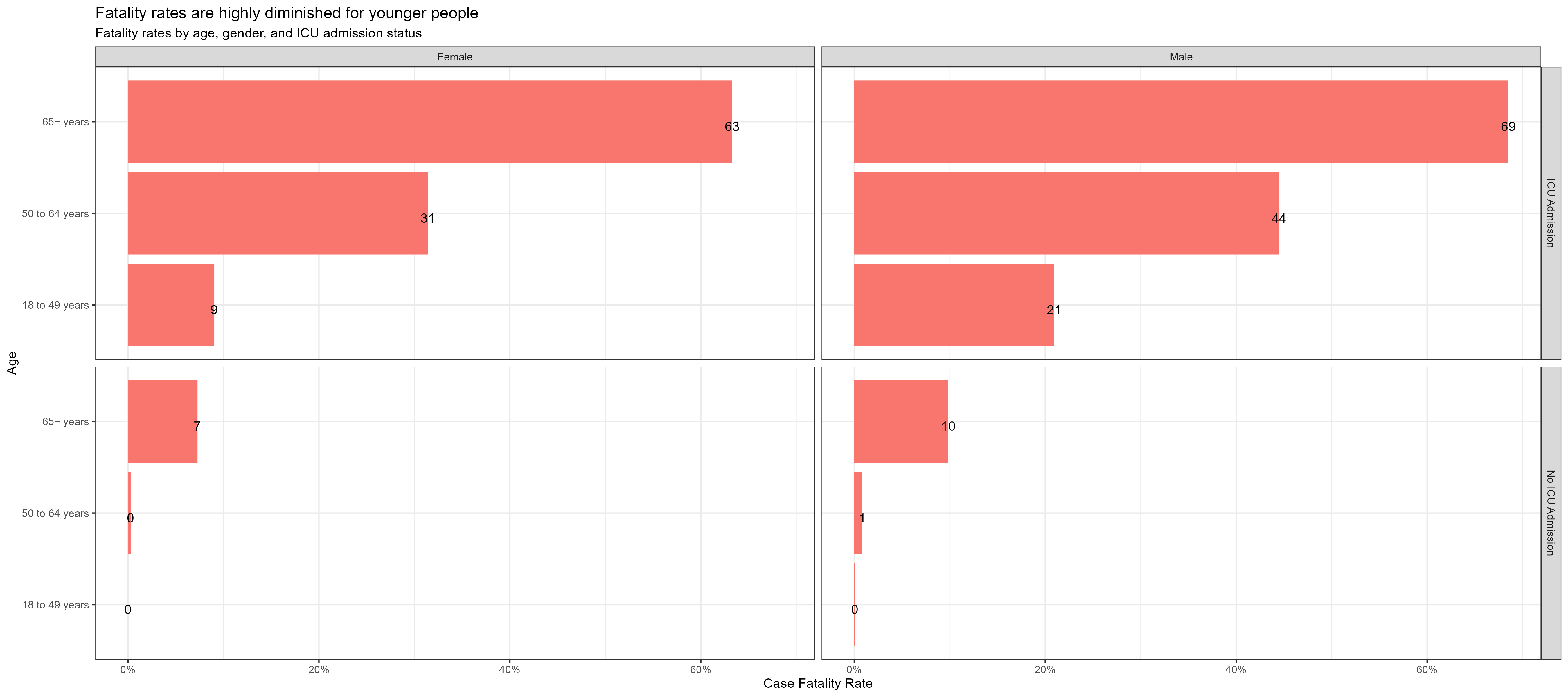
## Obtain the data

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

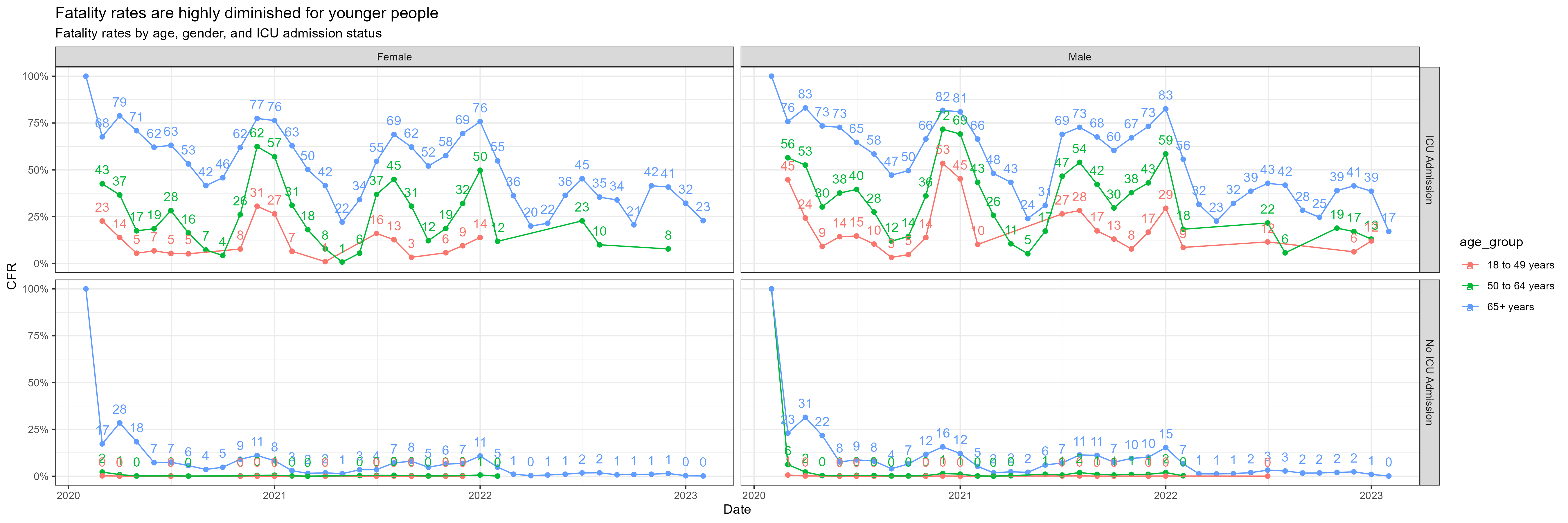
## 0.06 sec elapsed

## 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?



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)

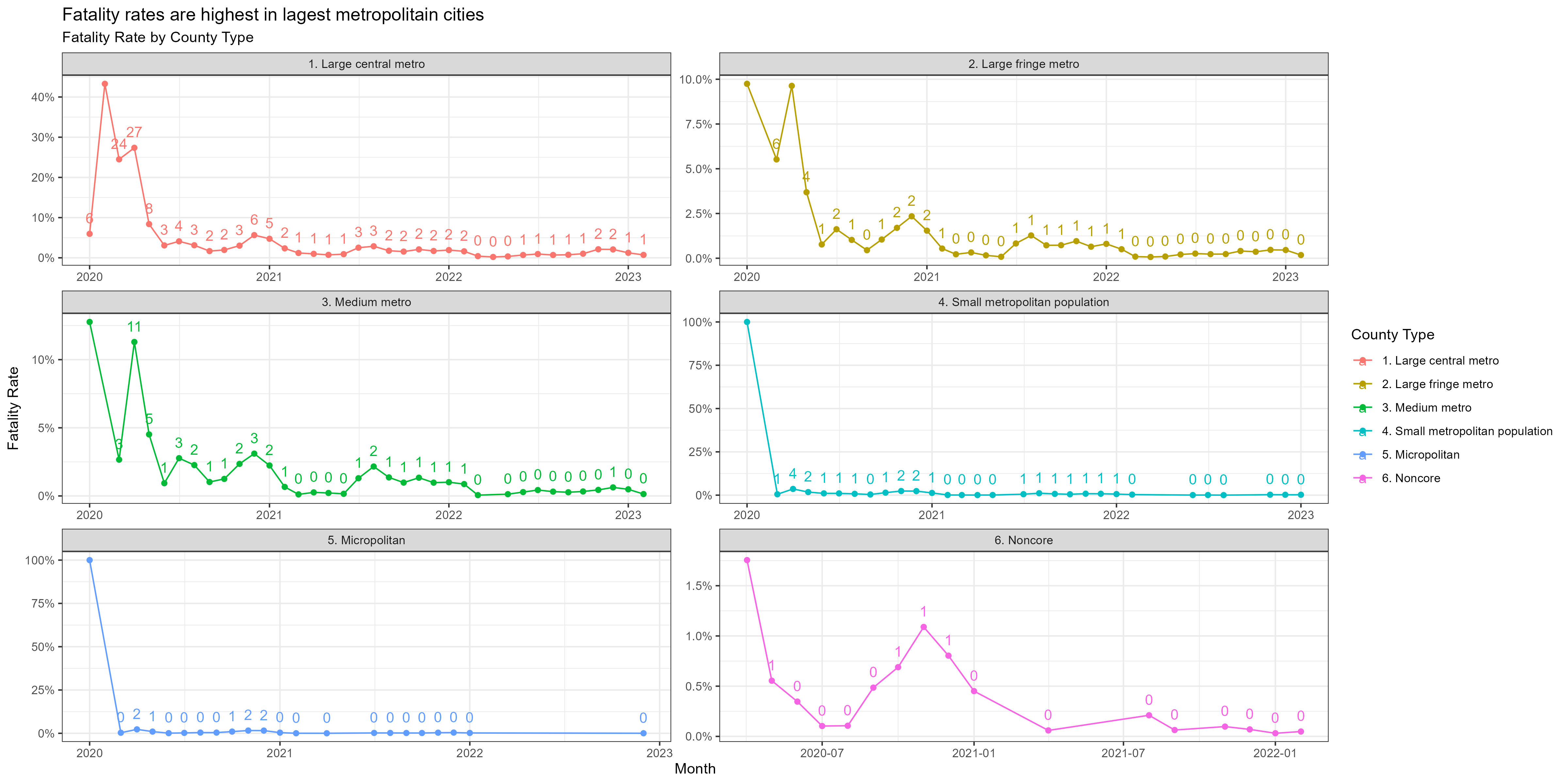
urban\_rural <- read\_xlsx(here::here("data", "NCHSURCodes2013.xlsx")) %>%   
 janitor::clean\_names()

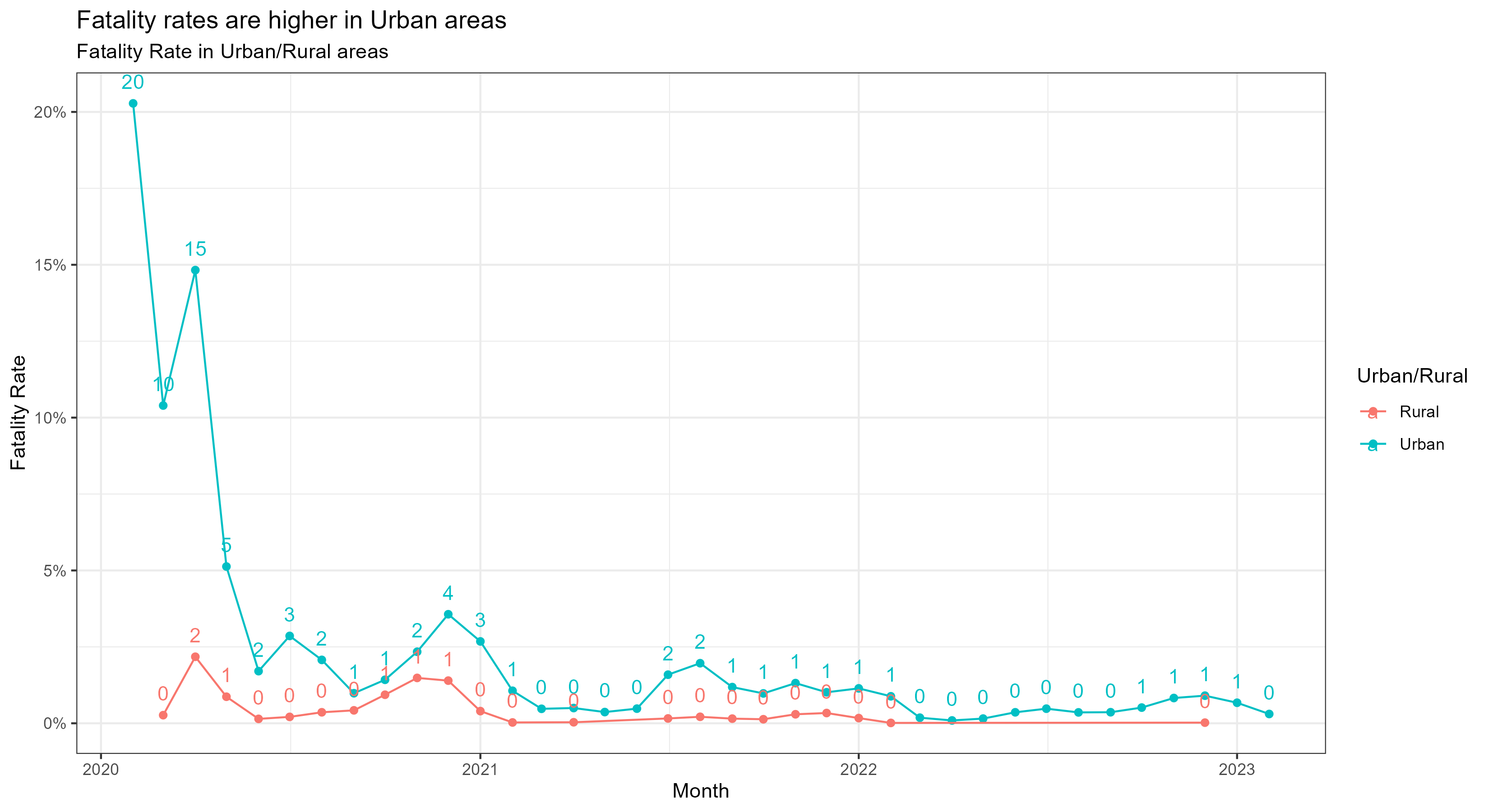
Each county belongs in six different categories, with categories 1-4 being urban areas and categories 5-6 being rural, according to the following criteria captured in x2013\_code

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)  
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()   
  
contributions <- contributions\_tables %>%   
 html\_element(".DataTable-Partial") %>% ## select table element  
 html\_table()

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

* 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”.

# write a function to parse\_currency  
scrape\_pac <- function(url){  
 year <- substr(url, start = nchar(url) - 4 + 1, stop = nchar(url))  
 contributions\_tables <- url %>%  
 read\_html()   
   
 contributions <- contributions\_tables %>%   
 html\_element(".DataTable-Partial") %>% ## select table element  
 html\_table()  
   
 # clean country/parent co and contributions   
 contributions <- contributions %>%  
 janitor::clean\_names() %>%   
 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),  
 year = as.numeric(year)  
   
 )  
   
 contributions  
}

* 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?

stub\_url <- "https://www.opensecrets.org/political-action-committees-pacs/foreign-connected-pacs/"  
  
url\_lst <- lapply(c("2000","2020","2022"), function(a) paste0(stub\_url,a))  
  
output\_list <- lapply(url\_lst, scrape\_pac)  
  
print(output\_list)  
# Ouput Does seem as expected

* 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.

years <- seq(2000, 2022, by = 2)  
  
urls <- lapply(years, function(a) paste0(stub\_url,a))  
  
contributions\_all <- bind\_rows(lapply(urls, scrape\_pac))  
  
print(contributions\_all %>% head(5))  
  
# Dump to CSV  
write.csv(contributions\_all, file = here::here("data", "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?  
  
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 #dataTable > tbody > tr:nth-child(2) > td:nth-child(1) > a > span.title
2. firm #dataTable > tbody > tr:nth-child(2) > td.hide-phone > a
3. functional area #dataTable > tbody > tr:nth-child(2) > td.hide-tablet-and-less
4. type #dataTable > tbody > tr:nth-child(2) > td.hide-tablet-landscape

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

library(robotstxt)  
paths\_allowed("https://www.consultancy.uk") #is it ok to scrape?  
  
  
listings\_tbl <- listings\_html %>%   
 html\_element(".dataTable") %>% ## select table element  
 html\_table()  
  
  
cons\_url <- "https://www.consultancy.uk/jobs/page/"  
htmls <- list()  
i <- 1 # Initial value  
error <- FALSE  
  
while (!error) {  
 # Check for an error condition  
 print(i)  
 tryCatch({  
 # Code to be executed if no error condition  
 url <- paste0(cons\_url,i)  
 print(url)  
 listings\_html <-   
 paste0(cons\_url,i) %>%  
 read\_html()  
   
 htmls <<- c(htmls, listings\_html)  
  
 }, error = function(err) {  
 print("Error")   
 error <<- TRUE  
 })  
   
 # Update the loop variable  
 i <- i + 1  
}

* 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?
* 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.

# write a function to parse\_currency  
  
scrape\_job <- function(url){  
 print(url)  
 listings\_html <-   
 url %>%  
 read\_html()  
   
 listings\_tbl <- listings\_html %>%   
 html\_element(".dataTable") %>% ## select table element  
 html\_table()  
  
 # clean country/parent co and contributions   
 listings\_tbl <- listings\_tbl %>%  
 janitor::clean\_names()  
 listings\_tbl  
}  
  
scrape\_jobs <- function(base\_url){  
 error <- FALSE  
 df\_list <- list()  
 i <- 1 # Initial value  
  
 while (!error) {  
 # Check for an error condition  
 print(i)  
 tryCatch({  
 # Code to be executed if no error condition  
 cons\_url <- paste0(base\_url,i)  
 listings\_tbl <- scrape\_job(cons\_url)  
   
 df\_list <- append(df\_list, list(listings\_tbl))  
  
  
 }, error = function(err) {  
 message("Error:", conditionMessage(err))  
 print("Error")   
 error <<- TRUE  
 })  
 # Update the loop variable  
 i <- i + 1  
 }  
  
 final\_df <- bind\_rows(df\_list)  
 print(final\_df)  
}  
   
all\_jobs <- scrape\_jobs("https://www.consultancy.uk/jobs/page/")  
   
print(all\_jobs %>% head(5))  
  
# Dump to CSV  
write.csv(all\_jobs, file = here::here("data", "all\_consulting\_jobs.csv"), row.names = FALSE)

# Details

* Who did you collaborate with:
* What, if anything, gave you the most trouble: Long loading time for cdc data, scraping errors, getting charts to look right