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

Eugene Nesterenko, github YevheniiN281

2023-05-31

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")  
)  
  
class(sky\_westminster)

## [1] "SQLiteConnection"  
## attr(,"package")  
## [1] "RSQLite"

How many tables does the database have?

The table has 7 tables:

* appg\_donatons
* appgs
* member\_appgs
* members
* parties
* party\_donations
* payments

DBI::dbListTables(sky\_westminster)

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

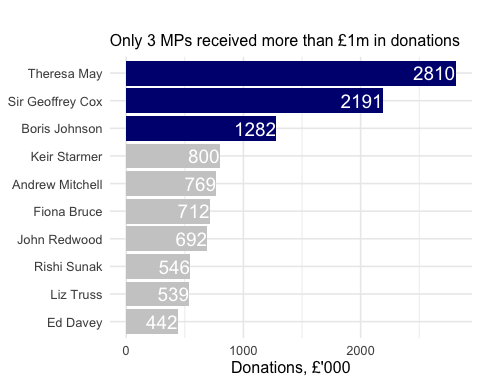
## Which MP has received the most amount of money?

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)

# Reference the tables from the database  
payments <- dplyr::tbl(sky\_westminster, "payments")  
members <- dplyr::tbl(sky\_westminster, "members")  
  
payments\_members <- left\_join(x = payments, y = members, by = c("member\_id"="id"))  
  
  
  
library(forcats)  
  
# Creating a dataframe with top-10 members by size of total donaions  
top\_members <- left\_join(x = payments, y = members, by = c("member\_id"="id")) %>%   
 group\_by(name) %>%   
 summarize(total\_donations\_000 = round(sum(value) / 1000,0)) %>%   
 select(name, total\_donations\_000) %>%   
 slice\_max(order\_by = total\_donations\_000, n=10)  
  
  
# converting to dataframe   
# (for some reason R did not treat "top\_members" as dataframe, which resulted in problems when applying fct\_reorder function)  
df\_top\_members <-as.data.frame(top\_members)

## 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.

my\_colours\_2 <- c("grey80", "navy")   
  
# Building a bar chart for the top-10 members by size of donation in a descending order  
df\_top\_members %>%   
   
 mutate(  
 million\_donations = ifelse(total\_donations\_000 >= 1000, TRUE, FALSE),  
 name = forcats::fct\_reorder(name, total\_donations\_000)  
 ) %>%  
   
 ggplot(aes(y = name , x = total\_donations\_000, fill = million\_donations)) +  
 geom\_col(show.legend = FALSE) +  
 scale\_fill\_manual(values = my\_colours\_2) +  
 geom\_text(  
 aes(label = total\_donations\_000, x = total\_donations\_000 - .25),  
 colour = "white",  
 size = 5,  
 hjust = 1) +  
   
 # Adding theme and axis titles  
 theme\_minimal(base\_size = 12)+  
 labs(  
 title = "",  
 subtitle = "Only 3 MPs received more than £1m in donations",  
 x = "Donations, £'000",  
 y = NULL  
 ) +  
 NULL



## 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?

# Function to return last n characters  
library(stringr)  
  
# Create a variable for the year of donation  
payments <- payments %>%   
 mutate(year = as.integer(str\_sub(date,-4)))  
  
# Summarize percentage of total donations by entity  
entity\_donations <- payments %>%   
   
 # Include only years 2020-2022  
 filter(year %in% c(2020:2022)) %>%  
   
 # Group by entity  
 group\_by(entity) %>%  
   
 # Summarize absolute value and calculate percentage value of the donations by entity  
 summarize(total\_donations = sum(value)) %>%   
 mutate(percent\_donations = round((total\_donations / sum(total\_donations)) \*100,1)) %>%  
   
 # Select only entity and percentage of donations, arrange from largest to smallest percentage  
 select(entity, percent\_donations) %>%   
   
 slice\_max(order\_by = percent\_donations, n=10)  
  
df\_entity\_donations <- as.data.frame(entity\_donations)  
  
#define colours to use: grey for everything, tomato for major donors (>5%)  
my\_colours <- c("grey80", "navy")   
  
df\_entity\_donations %>%   
   
 # Reordering donors by % of donations (descending) and marking the donors who contributed more than 5%  
 mutate(  
 major\_donor = ifelse(percent\_donations > 5,TRUE,FALSE),  
 entity = forcats::fct\_reorder(entity, percent\_donations)  
 ) %>%  
  
  
 # Building a horizontal bar chart for the percentage of donations  
 ggplot(aes(y = entity , x = percent\_donations, fill = major\_donor)) +  
 geom\_col(show.legend = FALSE) +  
 scale\_fill\_manual(values = my\_colours) +  
 geom\_text(  
 aes(label = percent\_donations, x = percent\_donations - .05),  
 colour = "white",  
 size = 5,  
 hjust = 1) +  
   
 # Adding theme and axis titles  
 theme\_minimal(base\_size = 12)+  
 labs(  
 title = "",  
 subtitle = "Withers LLP contributed more than 5% of total donations",  
 x = "Percent of donations",  
 y = NULL  
 ) +  
 NULL

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

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

# Select distinct entities from payments table  
unique\_entities <- payments %>%   
 distinct(entity)  
  
# Convert the result to dataframe  
unique\_entities <- as.data.frame(unique\_entities)  
  
# Print the number of distinct entities  
print(paste("Number of distinct entities =", nrow(unique\_entities)))

## [1] "Number of distinct entities = 2213"

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

# Reference the parties table from the database  
parties <- dplyr::tbl(sky\_westminster, "parties")  
  
# Create a joined table for MPs and their parties  
member\_party <- left\_join(x = members, y = parties, by = c("party\_id"="id")) %>%   
 mutate(  
 member\_name = name.x,  
 party\_name = name.y  
 ) %>%   
 select(member\_name, id, party\_name, party\_id)  
  
# Join the payments table with MP/parties dataframe  
payments\_members\_party <- left\_join(x = payments, y = member\_party, by = c("member\_id"="id"))  
  
entity\_party\_donations <- payments\_members\_party %>%   
 group\_by(entity, party\_name) %>%   
 summarize(total\_donations\_000 = round(sum(value) / 1000,0), na.rm=FALSE) %>%  
 select(entity, party\_name, total\_donations\_000)  
  
entity\_party\_donations\_classifier <- entity\_party\_donations %>%   
 group\_by(entity) %>%   
 summarize(recepients\_number = n(), na.rm=FALSE) %>%   
 mutate(single\_party\_donor = ifelse(recepients\_number == 1, TRUE, FALSE))  
  
entity\_party\_donations <- left\_join(x = entity\_party\_donations, y = entity\_party\_donations\_classifier, by = "entity") %>%   
 mutate(proved\_single\_party\_donation = single\_party\_donor \* total\_donations\_000)  
  
entity\_party\_donations %>%  
 ungroup() %>%   
 summarize(single\_party\_donors = as.double(sum(single\_party\_donor)),  
 total\_donors = as.double(n()),  
 single\_party\_donations = sum(proved\_single\_party\_donation),  
 total\_donations = sum(total\_donations\_000),  
 ) %>%   
 mutate(  
 percent\_single\_donors = round((single\_party\_donors/total\_donors)\*100,0),  
 percent\_single\_donations = round((single\_party\_donations/total\_donations)\*100,0)  
 )

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

## # Source: SQL [1 x 6]  
## # Database: sqlite 3.39.3 [/Users/user/Desktop/Code/DAM\_R/ENesterenko-DSB23/data/sky-westminster-files.db]  
## single\_party\_donors total\_donors single\_party\_donations total\_donations  
## <dbl> <dbl> <dbl> <dbl>  
## 1 2036 2469 29867 34593  
## # ℹ 2 more variables: percent\_single\_donors <dbl>,  
## # percent\_single\_donations <dbl>

**2036 entities (82% of total)** donate only to members of one party

In terms of total donation value, **£29.9m (86%)** comes from entities that donate only to members of one party.

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

***I can not understand the question, as next to blocks of code do not return any result. I would expect the .png file to open somewhere (or be shown in the Viewer) - however, I only have the .db file for the whole database.***

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

… 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?

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

dbDisconnect(sky\_westminster)

# 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: TYPE NAMES HERE
* Approximately how much time did you spend on this problem set: ANSWER HERE
* What, if anything, gave you the most trouble: ANSWER HERE

**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.