Applied_Stat_2_Lab2

```r
library(opendatatoronto)
library(tidyverse)
library(stringr)
library(skimr) # EDA
library(visdat) # EDA
library(janitor)
library(lubridate)
library(ggrepel)
```

```r
res <- list_package_resources("996cfe8d-fb35-40ce-b569-698d51fc683b") # obtained code from
res <- res |> mutate(year = str_extract(name, "202.?"))
delay_2022_ids <- res |> filter(year==2022) |> select(id) |> pull()

delay_2022 <- get_resource(delay_2022_ids)

# make the column names nicer to work with
delay_2022 <- clean_names(delay_2022)
```

```r
delay_codes <- get_resource("3900e649-f31e-4b79-9f20-4731bbfd94f7")
```

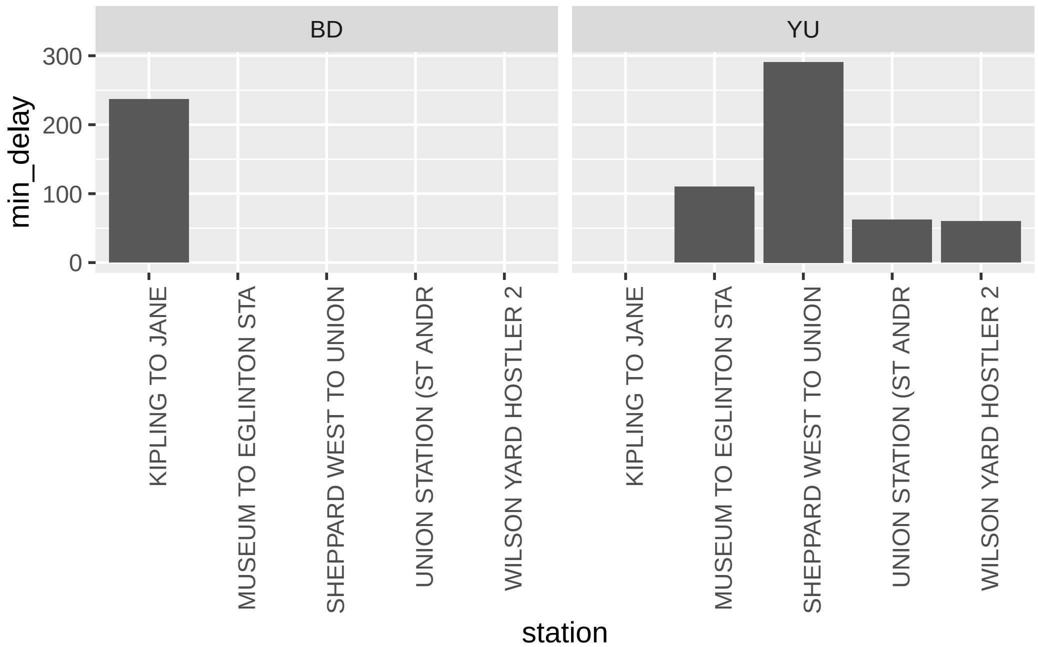
```
New names:
* `` -> `...1`
* `CODE DESCRIPTION` -> `CODE DESCRIPTION...3`
* `` -> `...4`
* `` -> `...5`
* `CODE DESCRIPTION` -> `CODE DESCRIPTION...7`
```

```r
delay_data_codebook <- get_resource("ca43ac3d-3940-4315-889b-a9375e7b8aa4")
```

```
delay_2022 <- delay_2022 |>
  mutate(station_clean = ifelse(str_starts(station, "ST"), word(station, 1,2), word(statio
```

Question 1

```
#Calculate mean delays and sort descending
mean_delays <- delay_2022 |> group_by(station) |> summarize(mean_delay = mean(min_delay),

#Take 5 highest mean delays and add the rest of the data
highest_delays <- head(mean_delays, 5)
delay_2022_top_stations <- delay_2022 %>%
  filter(station %in% highest_delays$station)


ggplot(delay_2022_top_stations, aes(x = station, y = min_delay)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x = element_text(angle = 90, hjust =1)) +
  facet_wrap(~line)
```



Question 2

```r
delay_2022 <- delay_2022 |>
  left_join(delay_codes |> rename(code = `SUB RMENU CODE`, code_desc = `CODE DESCRIPTION..
```

```
Joining with `by = join_by(code)`
```

```r
#Filter data by top 50% of delays
delay_2022_top_0.5 <- delay_2022 |>
                      filter(min_delay>0)|>
                      group_by(code)|>
                      summarise(no_rows = length(code))|>
                      arrange(-no_rows)|>
                      mutate(cumulative_sum = cumsum(no_rows))|>
                      mutate(half_sum = sum(no_rows)/2)|>
                      filter(cumulative_sum<=half_sum)

frequent_delay_codes <- delay_2022_top_0.5$code


lm_table_delay_code <- delay_2022 |>
                      filter(min_delay>0 & (code %in% frequent_delay_codes))
#Linear model with line and code as covariates
delay_model <- lm(min_delay ~ line + code, data = lm_table_delay_code)
summary(delay_model)
```

```

Call:
lm(formula = min_delay ~ line + code, data = lm_table_delay_code)

Residuals:
    Min      1Q  Median      3Q     Max 
-10.475  -2.450  -1.072   0.890 227.525 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   5.7698     0.3485  16.554  < 2e-16 ***
lineSHP       1.3899     0.5828   2.385 0.017132 *  
lineYU       -0.3203     0.2521  -1.270 0.204022    
codeMUIR      1.5470     0.4432   3.491 0.000486 ***
```

```
codeMUPAA    -1.6602     0.3741  -4.438  9.3e-06 ***
codePUOPO    -0.9396     0.3405  -2.759 0.005814 **
codeSUDP      0.9928     0.3344   2.969 0.003003 **
codeSUO       5.1117     0.4381  11.667  < 2e-16 ***
codeSUUT      7.7057     0.4069  18.938  < 2e-16 ***
codeTUNOA    -1.3775     0.3954  -3.484 0.000499 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.38 on 4396 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.1668,    Adjusted R-squared:  0.1651
F-statistic:  97.8 on 9 and 4396 DF,  p-value: < 2.2e-16
```

Our model suggests that the line "SHP" contributes significantly to delay minutes, which was not seen at all in the Explanatory Data Analysis in Question 1. This is because in Question 1 we found the five stations with highest mean delay. Naturally, the highest delay times were produced by outlier accidents that caused significantly longer delays than the total average is. Hence the stations or lines with the most frequent albeit shorter delays were not captured in the EDA. This is also explained by the fact that the most frequent delays do not contribute to the highest delays, which is also suggested by the relatively low beta coefficient estimates in our model.

Question 3

```r
#Data preprocessing
all_data <- search_packages("campaign")
campaign_id <- all_data$id
resource <- list_package_resources(campaign_id[1])
campaign_data <- get_resource('8b42906f-c894-4e93-a98e-acac200f34a4')
```

```
New names:
New names:
New names:
New names:
New names:
New names:
New names:
* `` -> `...2`
* `` -> `...3`
```

```r
campaign_data_mayoral <- campaign_data[[2]]
colnames(campaign_data_mayoral) <- as.character(campaign_data_mayoral[1,])
campaign_data_mayoral <- campaign_data_mayoral[-1,]
rownames(campaign_data_mayoral) <- NULL
campaign_data_mayoral <- clean_names(campaign_data_mayoral)
```

Question 4

We skim through the data using the skim function.

```r
skim(campaign_data_mayoral)
```

Table 1: Data summary

Name	campaign_data_mayoral
Number of rows	10199
Number of columns	13
Column type frequency:	
character	13
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
contributors_name	0	1	4	31	0	7545	0
contributors_address	10197	0	24	26	0	2	0
contributors_postal_code	0	1	7	7	0	5284	0
contribution_amount	0	1	1	18	0	209	0
contribution_type_desc	0	1	8	14	0	2	0
goods_or_service_desc	10188	0	11	40	0	9	0
contributor_type_desc	0	1	10	11	0	2	0
relationship_to_candidate	10166	0	6	9	0	2	0
president_business_manager	10197	0	13	16	0	2	0
authorized_representative	10197	0	13	16	0	2	0
candidate	0	1	9	18	0	27	0
office	0	1	5	5	0	1	0
ward	10199	0	NA	NA	0	0	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace

There are many blank columns or columns with missing values such as 'contributors_address','authorized_representative','president_business_manager' and so on. Furthermore, there are a couple of variables that should be factors and some that should be numerical.

```
not_all_na <- function(x) all(!is.na(x))
campaign_data_mayoral <- campaign_data_mayoral|>
                          select(where(not_all_na))
campaign_data_mayoral
```

```
# A tibble: 10,199 x 7
   contributors_name  contributors_postal_code contribution_amount
   <chr>              <chr>                    <chr>
 1 A D'Angelo, Tullio M6A 1P5                  300
 2 A Strazar, Martin  M2M 3B8                  300
 3 A'Court, K Susan   M4M 2J8                  36
 4 A'Court, K Susan   M4M 2J8                  100
 5 A'Court, K Susan   M4M 2J8                  100
 6 Aaron, Robert B    M6B 1H7                  250
 7 Abadi, Babak       M5S 2W7                  500
 8 Abadi, Babak       M5S 2W7                  500
 9 Abadi, David       M5S 2W7                  300
10 Abate, Frank       L4H 2K7                  150
# i 10,189 more rows
# i 4 more variables: contribution_type_desc <chr>,
#   contributor_type_desc <chr>, candidate <chr>, office <chr>
```

```
campaign_data_mayoral$contributor_type_desc <- as.factor(campaign_data_mayoral$contributor
campaign_data_mayoral$contribution_type_desc <- as.factor(campaign_data_mayoral$contributi
campaign_data_mayoral$contribution_amount <- as.numeric(campaign_data_mayoral$contribution
campaign_data_mayoral
```

```
# A tibble: 10,199 x 7
   contributors_name  contributors_postal_code contribution_amount
   <chr>              <chr>                                  <dbl>
 1 A D'Angelo, Tullio M6A 1P5                                  300
 2 A Strazar, Martin  M2M 3B8                                  300
```

```
 3 A'Court, K Susan   M4M 2J8                               36
 4 A'Court, K Susan   M4M 2J8                              100
 5 A'Court, K Susan   M4M 2J8                              100
 6 Aaron, Robert B    M6B 1H7                              250
 7 Abadi, Babak       M5S 2W7                              500
 8 Abadi, Babak       M5S 2W7                              500
 9 Abadi, David       M5S 2W7                              300
10 Abate, Frank       L4H 2K7                              150
# i 10,189 more rows
# i 4 more variables: contribution_type_desc <fct>,
#   contributor_type_desc <fct>, candidate <chr>, office <chr>
```

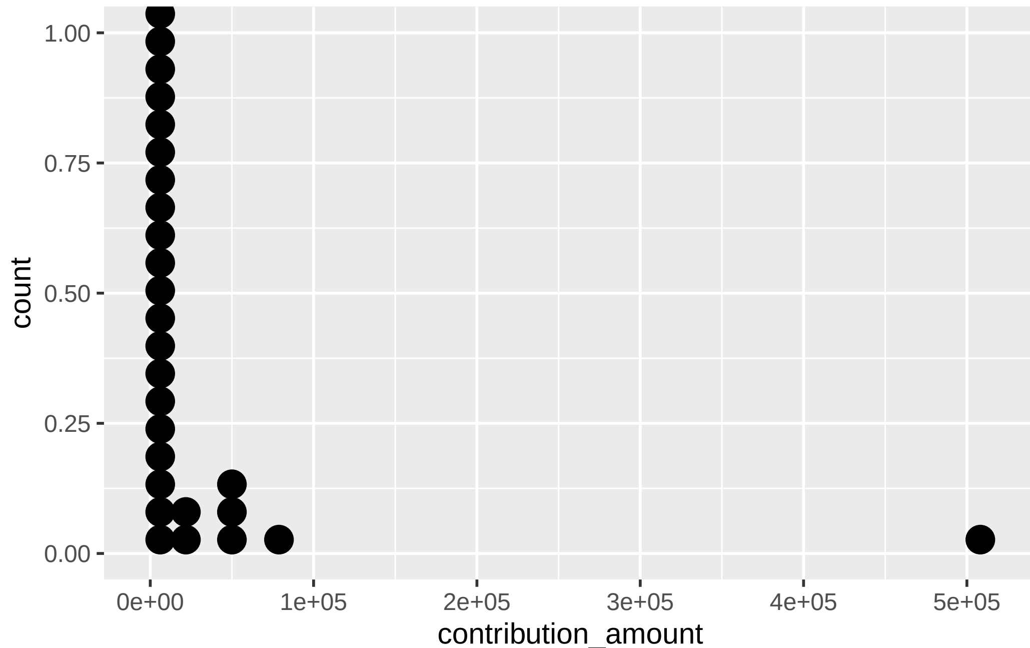
Question 5

```r
campaign_data_mayoral |> arrange(-contribution_amount)
```

```
# A tibble: 10,199 x 7
   contributors_name contributors_postal_code contribution_amount
   <chr>             <chr>                                  <dbl>
 1 Ford, Doug        M9A 2C3                              508225.
 2 Ford, Rob         M9A 3G9                               78805.
 3 Ford, Doug        M9A 2C3                               50000
 4 Ford, Rob         M9A 3G9                               50000
 5 Ford, Rob         M9A 3G9                               50000
 6 Goldkind, Ari     M5P 1P5                               23624.
 7 Ford, Rob         M9A 3G9                               20000
 8 Ford, Rob         M9A 3G9                               12210
 9 Di Paola, Rocco   M3H 2T1                                6000
10 Thomson, Sarah    M4W 2X6                                4426.
# i 10,189 more rows
# i 4 more variables: contribution_type_desc <fct>,
#   contributor_type_desc <fct>, candidate <chr>, office <chr>
```

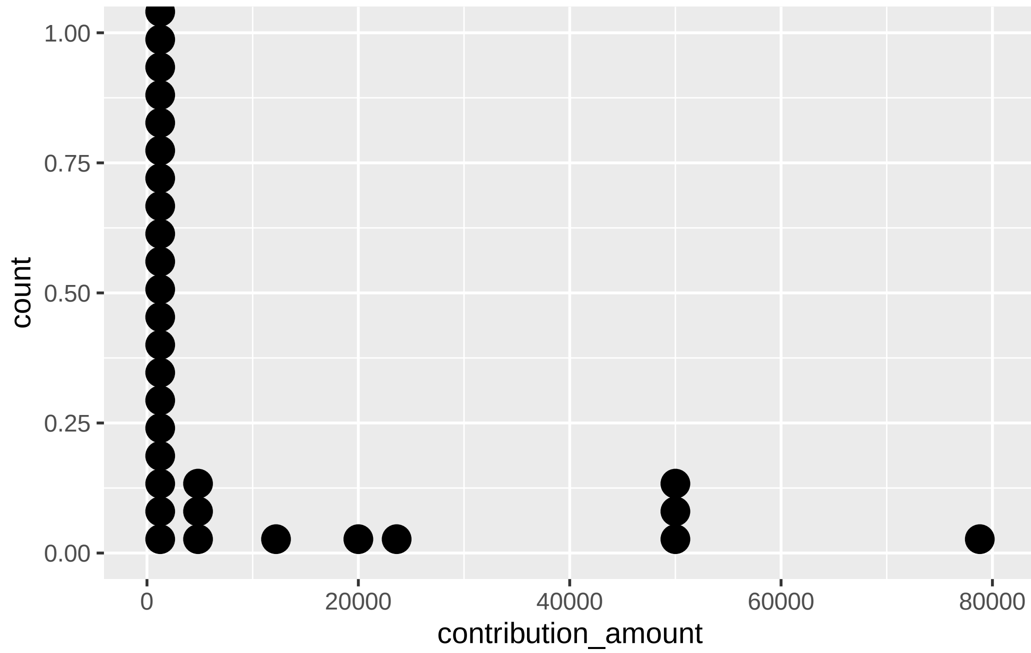
```r
ggplot(data = campaign_data_mayoral,aes(x=contribution_amount))+
  geom_dotplot()
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with
`binwidth`.
```



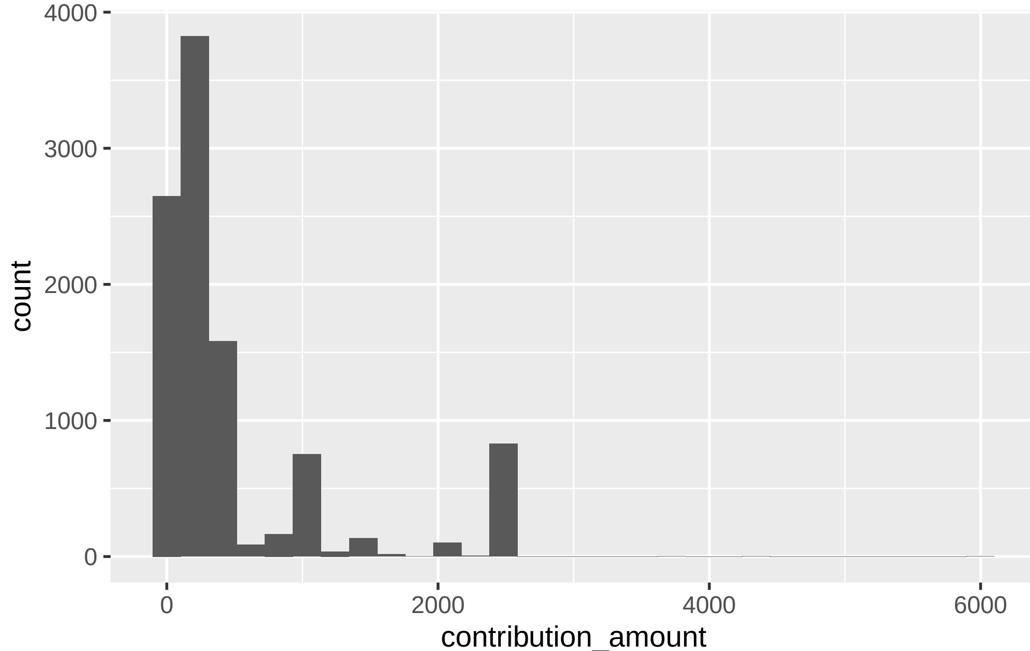

```
campaign_data_mayoral_contribution_distribution <-
  campaign_data_mayoral |> filter(contribution_amount < 100000)

ggplot(data=campaign_data_mayoral_contribution_distribution, aes(x=contribution_amount))+
  geom_dotplot()
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with
`binwidth`.
```



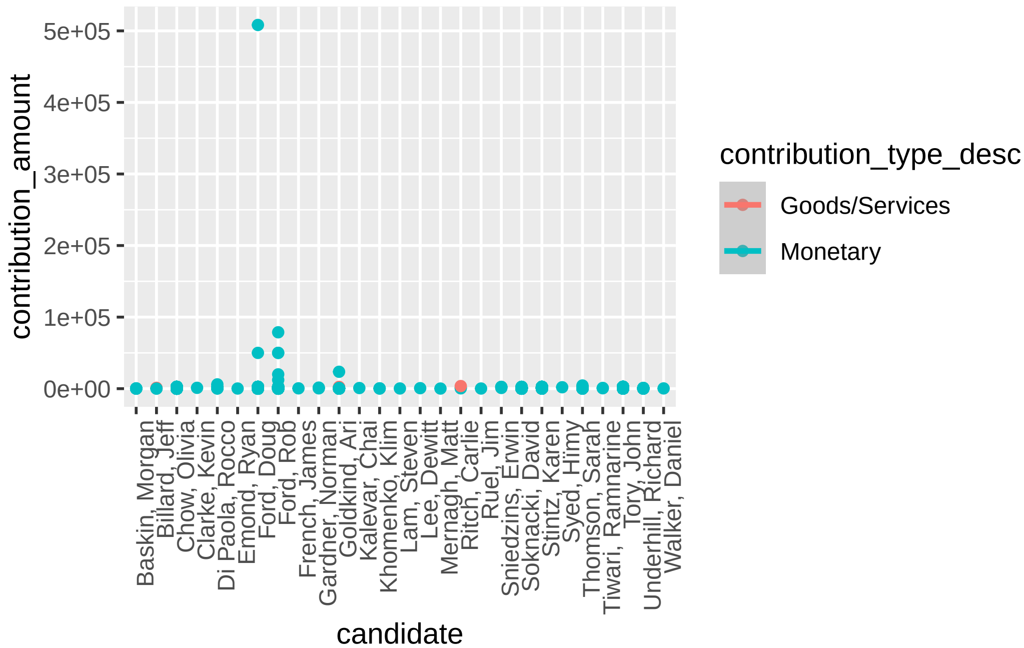

```
campaign_data_mayoral_contribution_distribution_2 <-
  campaign_data_mayoral |> filter(contribution_amount < 10000)

ggplot(data=campaign_data_mayoral_contribution_distribution_2, aes(x=contribution_amount))
  geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = campaign_data_mayoral, aes(x = candidate,
                                         y = contribution_amount,
                                         color = contribution_type_desc ))+
  geom_point()+
  geom_smooth()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



Looking at the plots above, the following outliers can be found: The biggest outlier is the donation of 500,000$ by Doug Ford. Then there are also a few other donations close to 100,000$ done by candidates Ryan Emond and Rob Ford. After filtering for donations less than 100,000$ we see a few outlier donations of above 20,000$. Thus we filter once more for donations less than 10,000$ to get a better sense of the majority of the data.

Question 6

```
candidate_contri <- campaign_data_mayoral|>
                    group_by(candidate) |>
                    summarise(
                    total_contri = sum(contribution_amount, na.rm = TRUE),
                    mean_contri = mean(contribution_amount, na.rm = TRUE),
                    contri_count = n()
                    )

top_total_contri <- candidate_contri |>
                    arrange(-total_contri)|>
                    select(candidate,total_contri)|>
                    head(5)
top_mean_contri <- candidate_contri |>
```

```r
                     arrange(-mean_contri)|>
                     select(candidate,mean_contri)|>
                     head(5)
top_contri_count <- candidate_contri  |>
                      arrange(-contri_count)|>
                      select(candidate,contri_count)|>
                      head(5)

top_total_contri
```

```
# A tibble: 5 x 2
  candidate     total_contri
  <chr>                <dbl>
1 Tory, John        2767869.
2 Chow, Olivia      1638266.
3 Ford, Doug         889897.
4 Ford, Rob          387648.
5 Stintz, Karen      242805
```

```r
top_mean_contri
```

```
# A tibble: 5 x 2
  candidate        mean_contri
  <chr>                  <dbl>
1 Sniedzins, Erwin       2025
2 Syed, Himy             2018
3 Ritch, Carlie          1887.
4 Ford, Doug             1456.
5 Clarke, Kevin          1200
```

```r
top_contri_count
```

```
# A tibble: 5 x 2
  candidate       contri_count
  <chr>                  <int>
1 Chow, Olivia            5708
2 Tory, John              2602
3 Ford, Doug               611
4 Ford, Rob                538
5 Soknacki, David          314
```

Question 7

```
non_candidate_contri <- campaign_data_mayoral |>
                        filter(contributors_name != candidate)

non_candidate_contri <- non_candidate_contri|>
                        group_by(candidate) |>
                        summarise(
                        total_contri_popular = sum(contribution_amount, na.rm = TRUE),
                        mean_contri_popular = mean(contribution_amount, na.rm = TRUE),
                        contri_count_popular = n()
                    )

top_total_contri_popular <- non_candidate_contri |>
                            arrange(-total_contri_popular)|>
                            select(candidate,total_contri_popular)|>
                            head(5)
top_mean_contri_popular <- non_candidate_contri |>
                           arrange(-mean_contri_popular)|>
                           select(candidate,mean_contri_popular)|>
                           head(5)
top_contri_count_popular <- non_candidate_contri |>
                            arrange(-contri_count_popular)|>
                            select(candidate,contri_count_popular)|>
                            head(5)
top_total_contri_popular
```

```
# A tibble: 5 x 2
  candidate     total_contri_popular
  <chr>                        <dbl>
1 Tory, John                2765369.
2 Chow, Olivia              1634766.
3 Ford, Doug                 331173.
4 Stintz, Karen              242805
5 Ford, Rob                  174510.
```

```
top_mean_contri_popular
```

```
# A tibble: 5 x 2
  candidate         mean_contri_popular
```

```
  <chr>                           <dbl>
1 Ritch, Carlie                   1887.
2 Sniedzins, Erwin                1867.
3 Tory, John                      1063.
4 Gardner, Norman                 1000
5 Tiwari, Ramnarine               1000
```

```
top_contri_count_popular
```

```
# A tibble: 5 x 2
  candidate        contri_count_popular
  <chr>                           <int>
1 Chow, Olivia                     5706
2 Tory, John                       2601
3 Ford, Doug                        608
4 Ford, Rob                         531
5 Soknacki, David                   314
```

Question 8

```
multiple_contri <- campaign_data_mayoral |>
  group_by(contributors_name) |>
  summarise(unique_candidates = n_distinct(candidate))

multiple_contri_count <- sum(multiple_contri$unique_candidates > 1)
multiple_contri_count
```

```
[1] 184
```