week_2_lab

Bingqing Li

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

```
all_data <- list_packages(limit = 500)
head(all_data)
```

```
# A tibble: 6 x 11
  title             id    topics civic_issues publisher excerpt dataset_category
  <chr>             <chr> <chr>  <chr>        <chr>     <chr>   <chr>
1 Committee of Adj~ 260e~ City ~ <NA>         City Pla~ This d~ Table
2 Residential Fron~ 4a65~ Locat~ Mobility,Cl~ Transpor~ Legall~ Table
3 Dinesafe          ea1d~ Publi~ <NA>         Toronto ~ Snapsh~ Table
4 Property Boundar~ 1aca~ Locat~ <NA>         Informat~ This d~ Document
5 Lobbyist Registry 6a87~ City ~ <NA>         Lobbyist~ The Lo~ Document
6 Municipal Licens~ 5da2~ City ~ Affordable ~ Municipa~ This d~ Document
# i 4 more variables: num_resources <int>, formats <chr>, refresh_rate <chr>,
#   last_refreshed <date>
```

```
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)
```

```r
# make the column names nicer to work with
delay_2022 <- clean_names(delay_2022)
# note: I obtained these codes from the 'id' column in the `res` object above
delay_codes <- get_resource("3900e649-f31e-4b79-9f20-4731bbfd94f7")
delay_data_codebook <- get_resource("ca43ac3d-3940-4315-889b-a9375e7b8aa4")

delay_2022 <- delay_2022 |> distinct()


## Removing the observations that have non-standardized lines
delay_2022 <- delay_2022 |> filter(line %in% c("BD", "YU", "SHP", "SRT"))

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

delay_2022
```

```
# A tibble: 19,460 x 11
   date                time  day     station code  min_delay min_gap bound line
   <dttm>              <chr> <chr>   <chr>   <chr>     <dbl>   <dbl> <chr> <chr>
 1 2022-01-01 00:00:00 15:59 Saturd~ LAWREN~ SRDP          0       0 N     SRT
 2 2022-01-01 00:00:00 02:23 Saturd~ SPADIN~ MUIS          0       0 <NA>  BD
 3 2022-01-01 00:00:00 22:00 Saturd~ KENNED~ MRO           0       0 <NA>  SRT
 4 2022-01-01 00:00:00 02:28 Saturd~ VAUGHA~ MUIS          0       0 <NA>  YU
 5 2022-01-01 00:00:00 02:34 Saturd~ EGLINT~ MUATC         0       0 S     YU
 6 2022-01-01 00:00:00 05:40 Saturd~ QUEEN ~ MUNCA         0       0 <NA>  YU
 7 2022-01-01 00:00:00 06:56 Saturd~ DAVISV~ MUNCA         0       0 <NA>  YU
 8 2022-01-01 00:00:00 06:58 Saturd~ ST PAT~ MUNCA         0       0 <NA>  YU
 9 2022-01-01 00:00:00 07:01 Saturd~ PAPE S~ MUNCA         0       0 <NA>  BD
10 2022-01-01 00:00:00 07:43 Saturd~ WILSON~ TUATC        10       0 S     YU
# i 19,450 more rows
# i 2 more variables: vehicle <dbl>, code_desc <chr>
```

```r
delay_2022 <- delay_2022 |>
  mutate(code_srt = ifelse(line=="SRT", code, "NA")) |>
  left_join(delay_codes |> rename(code_srt = `SRT RMENU CODE`, code_desc_srt = `CODE DESCR
  mutate(code = ifelse(code_srt=="NA", code, code_srt),
         code_desc = ifelse(is.na(code_desc_srt), code_desc, code_desc_srt)) |>
  select(-code_srt, -code_desc_srt)
```

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

delay_2022 <- delay_2022 |>
  mutate(code_red = case_when(
    str_starts(code_desc, "No") ~ word(code_desc, 1, 2),
    str_starts(code_desc, "Operator") ~ word(code_desc, 1,2),
    TRUE ~ word(code_desc,1))
         )

delay_2022
```

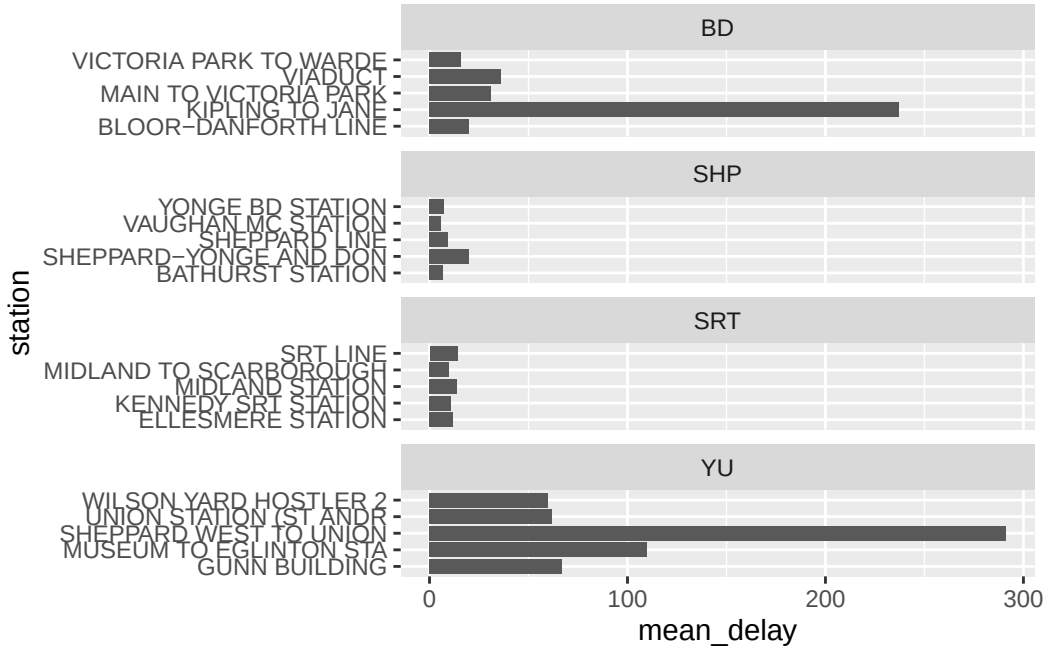
```
# A tibble: 19,460 x 13
   date                time  day     station code  min_delay min_gap bound line
   <dttm>              <chr> <chr>   <chr>   <chr>     <dbl>   <dbl> <chr> <chr>
 1 2022-01-01 00:00:00 15:59 Saturd~ LAWREN~ SRDP          0       0 N     SRT
 2 2022-01-01 00:00:00 02:23 Saturd~ SPADIN~ MUIS          0       0 <NA>  BD
 3 2022-01-01 00:00:00 22:00 Saturd~ KENNED~ MRO           0       0 <NA>  SRT
 4 2022-01-01 00:00:00 02:28 Saturd~ VAUGHA~ MUIS          0       0 <NA>  YU
 5 2022-01-01 00:00:00 02:34 Saturd~ EGLINT~ MUATC         0       0 S     YU
 6 2022-01-01 00:00:00 05:40 Saturd~ QUEEN ~ MUNCA         0       0 <NA>  YU
 7 2022-01-01 00:00:00 06:56 Saturd~ DAVISV~ MUNCA         0       0 <NA>  YU
 8 2022-01-01 00:00:00 06:58 Saturd~ ST PAT~ MUNCA         0       0 <NA>  YU
 9 2022-01-01 00:00:00 07:01 Saturd~ PAPE S~ MUNCA         0       0 <NA>  BD
10 2022-01-01 00:00:00 07:43 Saturd~ WILSON~ TUATC        10       0 S     YU
# i 19,450 more rows
# i 4 more variables: vehicle <dbl>, code_desc <chr>, station_clean <chr>,
#   code_red <chr>
```

Q1

1. Using the `delay_2022` data, plot the five stations with the highest mean delays. Facet the graph by `line`

```
delay_2022 |>
  group_by(line, station) |>
  summarise(mean_delay = mean(min_delay)) |>
  arrange(-mean_delay) |>
  slice(1:5) |>
  ggplot(aes(x = station, y = mean_delay)) +
```

```
  geom_col() +
  facet_wrap(vars(line),
             scales = "free_y",
               nrow = 4) +
  coord_flip()
```



Q2

2. Restrict the `delay_2022` to delays that are greater than 0 and to only have delay reasons that appear in the top 50% of most frequent delay reasons. Perform a regression to study the association between delay minutes, and two covariates: line and delay reason. It's up to you how to specify the model, but make sure it's appropriate to the data types. Comment briefly on the results, including whether results generally agree with the exploratory data analysis above.

```
delay_reasons <- delay_2022 |>
  filter(min_delay > 0) |>
  count(code) |>
  arrange(-n) |>
  mutate(cumulative_sum = cumsum(n)) |>
  filter(cumulative_sum <= sum(n)/2)|>
```

```
  select(code)

delay_reasons
```

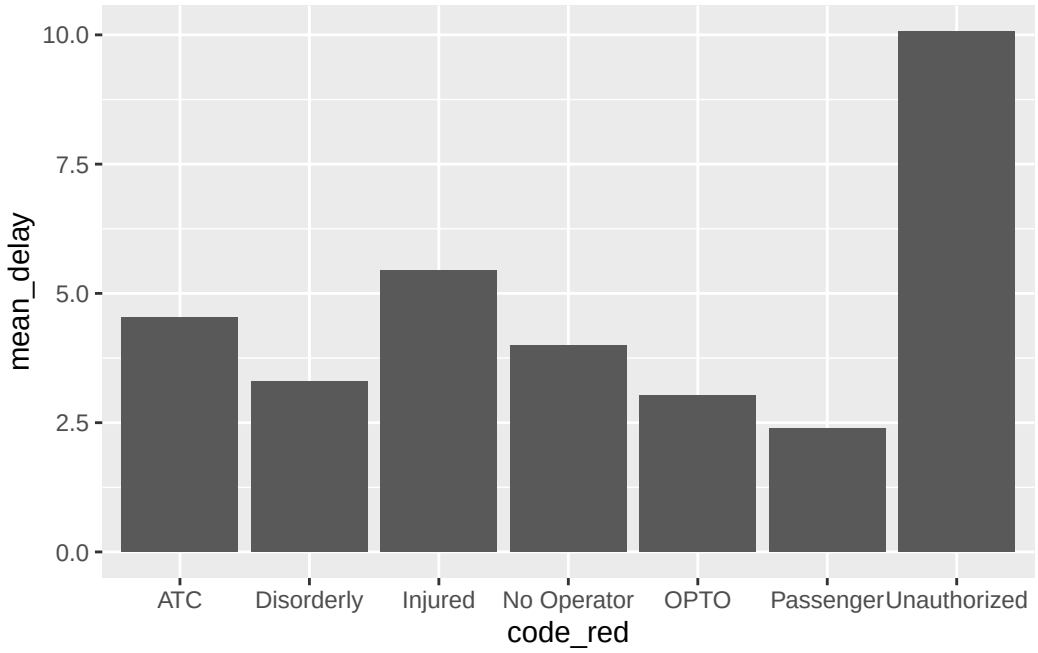
```
# A tibble: 8 x 1
  code
  <chr>
1 SUDP
2 PUOPO
3 MUATC
4 MUPAA
5 SUUT
6 TUNOA
7 SUO
8 MUIR
```

```
d2 <- delay_2022 |>
  filter(code %in% delay_reasons$code)

d2
```

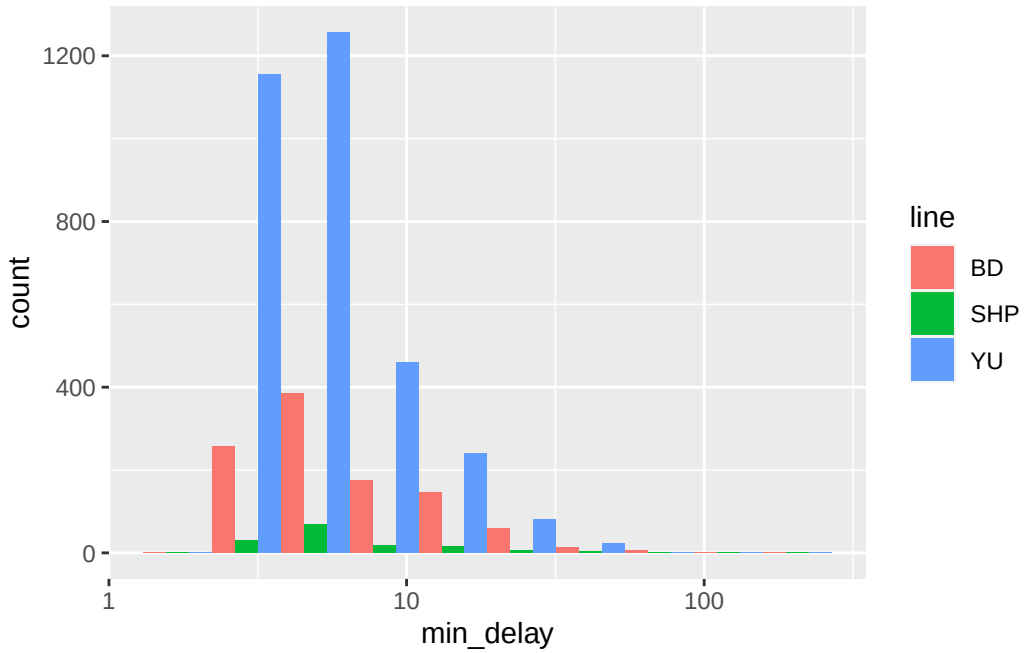
```
# A tibble: 7,678 x 13
   date                time  day     station code  min_delay min_gap bound line
   <dttm>              <chr> <chr>   <chr>   <chr>     <dbl>   <dbl> <chr> <chr>
 1 2022-01-01 00:00:00 02:34 Saturd~ EGLINT~ MUATC         0       0 S     YU
 2 2022-01-01 00:00:00 08:12 Saturd~ FINCH ~ TUNOA         6      12 S     YU
 3 2022-01-01 00:00:00 09:01 Saturd~ WELLES~ SUO           0       0 <NA>  YU
 4 2022-01-01 00:00:00 09:10 Saturd~ SHEPPA~ PUOPO         0       0 S     YU
 5 2022-01-01 00:00:00 09:51 Saturd~ FINCH ~ TUNOA         6      12 S     YU
 6 2022-01-01 00:00:00 12:01 Saturd~ DAVISV~ SUDP          3       8 S     YU
 7 2022-01-01 00:00:00 12:14 Saturd~ RUNNYM~ SUUT         20      25 W     BD
 8 2022-01-01 00:00:00 14:14 Saturd~ ROSEDA~ SUUT          0       0 S     YU
 9 2022-01-01 00:00:00 18:20 Saturd~ EGLINT~ MUATC         3      10 S     YU
10 2022-01-01 00:00:00 18:59 Saturd~ EGLINT~ MUATC         3      10 S     YU
# i 7,668 more rows
# i 4 more variables: vehicle <dbl>, code_desc <chr>, station_clean <chr>,
#   code_red <chr>
```

```r
fig1 <- d2 |>
  group_by(code_red) |>
  summarise(mean_delay = mean(min_delay)) |>
  arrange(-mean_delay) |>
  ggplot(aes(x = code_red,
             y = mean_delay)) +
  geom_col()
```

```r
fig1
```




```r
ggplot(data = d2) +
  geom_histogram(aes(x = min_delay, fill = line), position = 'dodge', bins = 10) +
  scale_x_log10()
```




```
d2 |>
  group_by(line)|>
  summarise(mean_delay = mean(min_delay))
```

```
# A tibble: 3 x 2
  line  mean_delay
  <chr>      <dbl>
1 BD          3.85
2 SHP         4.93
3 YU          3.68
```

```
model <- glm(min_delay ~ line +  code_red, data = d2, family = poisson)

summary(model)
```

```
Call:
glm(formula = min_delay ~ line + code_red, family = poisson,
    data = d2)
```

```
Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)           1.52700    0.02149  71.065  < 2e-16 ***
lineSHP               0.35350    0.03227  10.955  < 2e-16 ***
lineYU               -0.01546    0.01417  -1.091    0.275
code_redDisorderly   -0.33997    0.02143 -15.863  < 2e-16 ***
code_redInjured       0.16474    0.02707   6.086 1.16e-09 ***
code_redNo Operator  -0.13433    0.02905  -4.624 3.76e-06 ***
code_redOPTO         -0.43794    0.02378 -18.418  < 2e-16 ***
code_redPassenger    -0.65841    0.02166 -30.396  < 2e-16 ***
code_redUnauthorized  0.78030    0.02185  35.712  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 48948  on 7677  degrees of freedom
Residual deviance: 42660  on 7669  degrees of freedom
AIC: 58205

Number of Fisher Scoring iterations: 6
```

The reference line is line BD with delay reason of ATC. The signs for lineSHP is positive and lineYU is negative, meaning that compared to the reference line, SHP has longer delay time and YU has shorter delay time. This results are same with mean delay time: BD 3.848210, SHP 4.925764, YU 3.677986. Beside, the signs of the coefficients for code_red of injured and Unauthorized are positive, meaning that these reasons for delays are associated with longer delays compared to the reference category. The exploratory data analysis in the provided image suggested that these reasons may cause longer . Thus, we still can conclude that regression results can generally agree with the exploratory data analysis above.

Q3

3. Using the `opendatatoronto` package, download the data on mayoral campaign contributions for 2014 and clean it up. Hints:
    - find the ID code you need for the package you need by searching for 'campaign' in the `all_data` tibble above
    - you will then need to `list_package_resources` to get ID for the data file
    - note: the 2014 file you will get from `get_resource` has a bunch of different campaign contributions, so just keep the data that relates to the Mayor election

- clean up the data format (fixing the parsing issue and standardizing the column names using `janitor`)

```
id <- all_data[all_data$title =='Elections - Campaign Contributions',]$id
res <- list_package_resources(id) # obtained code from searching data frame above
res <- res |> mutate(year = str_extract(name, "201.?")) |> filter(str_like(name,'%Data'))
camp_2014_ids <- res |> filter(year==2014) |> select(id) |> pull()

camp_2014 <- get_resource(camp_2014_ids)

# make the column names nicer to work with
camp_2014 <- clean_names(camp_2014)
camp_2014 <- camp_2014$x2_mayor_contributions_2014_election_xls
colnames(camp_2014) <- camp_2014[1,]
camp_2014 <- camp_2014[-1,]
camp_2014 <- camp_2014 |> clean_names()
camp_2014
```

```
# A tibble: 10,199 x 13
   contributors_name  contributors_address contributors_postal_code
   <chr>              <chr>                <chr>
 1 A D'Angelo, Tullio <NA>                 M6A 1P5
 2 A Strazar, Martin  <NA>                 M2M 3B8
 3 A'Court, K Susan   <NA>                 M4M 2J8
 4 A'Court, K Susan   <NA>                 M4M 2J8
 5 A'Court, K Susan   <NA>                 M4M 2J8
 6 Aaron, Robert B    <NA>                 M6B 1H7
 7 Abadi, Babak       <NA>                 M5S 2W7
 8 Abadi, Babak       <NA>                 M5S 2W7
 9 Abadi, David       <NA>                 M5S 2W7
10 Abate, Frank       <NA>                 L4H 2K7
# i 10,189 more rows
# i 10 more variables: contribution_amount <chr>, contribution_type_desc <chr>,
#   goods_or_service_desc <chr>, contributor_type_desc <chr>,
#   relationship_to_candidate <chr>, president_business_manager <chr>,
#   authorized_representative <chr>, candidate <chr>, office <chr>, ward <chr>
```

Q4

4. Summarize the variables in the dataset. Are there missing values, and if so, should we be worried about them? Is every variable in the format it should be? If not, create new

variable(s) that are in the right format.

```
skim(camp_2014)
```

Table 1: Data summary

Name	camp_2014
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

There are missing values in Contributor's Address, Goods or Service Desc, Relationship to Candidate, President/Business Manager, Authorized Representative and Ward. Given that our dataset contains 10,199 samples, and these columns have missing values for most or all rows, we should consider deleting these columns when analyzing the data. Therefore, we should not be concerned about these missing values.

For contribution_amount, it is character. The correct format is numeric number.

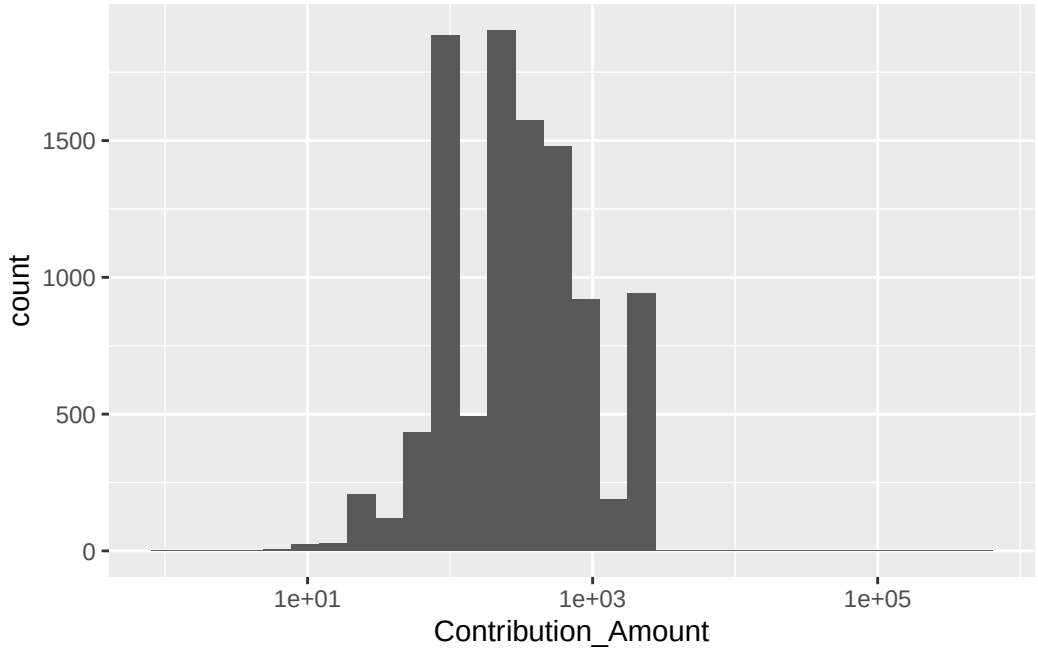
```r
camp_2014 <- camp_2014 |>
  mutate(Contribution_Amount = as.integer(contribution_amount))

camp_2014
```

```
# A tibble: 10,199 x 14
   contributors_name  contributors_address contributors_postal_code
   <chr>              <chr>                <chr>
 1 A D'Angelo, Tullio <NA>                 M6A 1P5
 2 A Strazar, Martin  <NA>                 M2M 3B8
 3 A'Court, K Susan   <NA>                 M4M 2J8
 4 A'Court, K Susan   <NA>                 M4M 2J8
 5 A'Court, K Susan   <NA>                 M4M 2J8
 6 Aaron, Robert B    <NA>                 M6B 1H7
 7 Abadi, Babak       <NA>                 M5S 2W7
 8 Abadi, Babak       <NA>                 M5S 2W7
 9 Abadi, David       <NA>                 M5S 2W7
10 Abate, Frank       <NA>                 L4H 2K7
# i 10,189 more rows
# i 11 more variables: contribution_amount <chr>, contribution_type_desc <chr>,
#   goods_or_service_desc <chr>, contributor_type_desc <chr>,
#   relationship_to_candidate <chr>, president_business_manager <chr>,
#   authorized_representative <chr>, candidate <chr>, office <chr>, ward <chr>,
#   Contribution_Amount <int>
```

Q5

5. Visually explore the distribution of values of the contributions. What contributions are notable outliers? Do they share a similar characteristic(s)? It may be useful to plot the distribution of contributions without these outliers to get a better sense of the majority of the data.

```r
camp_2014 |>
  ggplot() +
  geom_histogram(aes(x = Contribution_Amount)) +
  scale_x_log10()
```



Our initial EDA hinted at some outlying contribution amounts.

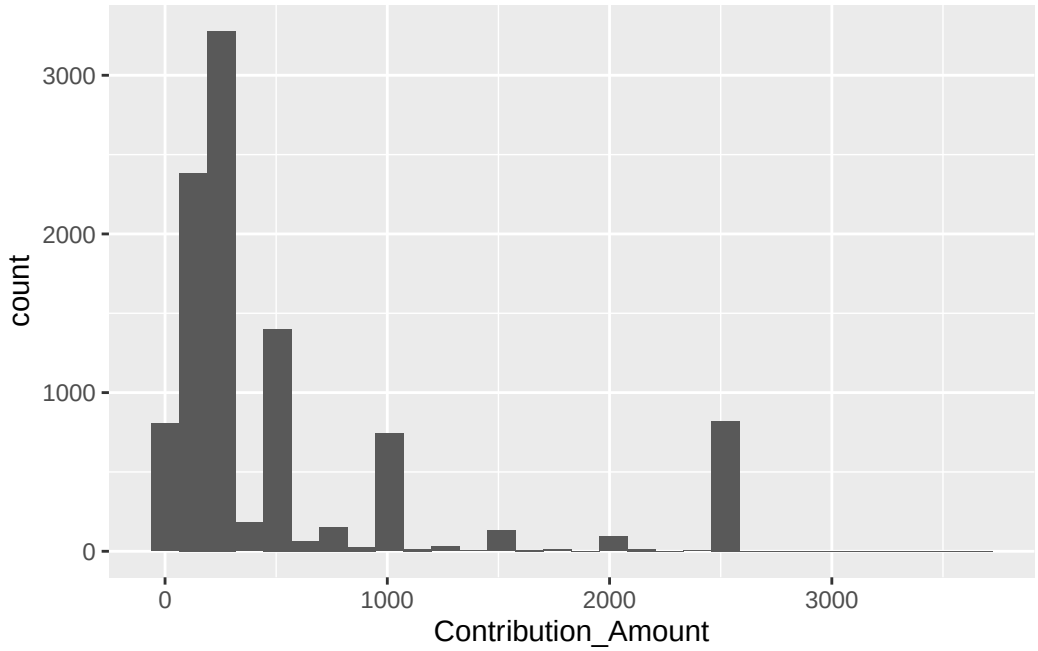
```
camp_2014 |>
  arrange(-Contribution_Amount) |>
  select(contributors_name, Contribution_Amount, relationship_to_candidate)
```

```
# A tibble: 10,199 x 3
   contributors_name Contribution_Amount relationship_to_candidate
   <chr>                           <int> <chr>
 1 Ford, Doug                     508224 Candidate
 2 Ford, Rob                       78804 Candidate
 3 Ford, Doug                      50000 Candidate
 4 Ford, Rob                       50000 Candidate
 5 Ford, Rob                       50000 Candidate
 6 Goldkind, Ari                   23623 Candidate
 7 Ford, Rob                       20000 Candidate
 8 Ford, Rob                       12210 Candidate
 9 Di Paola, Rocco                  6000 Candidate
10 Thomson, Sarah                   4425 Candidate
# i 10,189 more rows
```

The persons who contributed a lot are candidate. Let's plot the distribution without candidates

themselves contributions.

```
camp_2014 |>
  filter(relationship_to_candidate != 'Candidate'
         | is.na(relationship_to_candidate) ) |>
  ggplot() +
  geom_histogram(aes(Contribution_Amount))
```



Q6

6. List the top five candidates in each of these categories:
    - total contributions
    - mean contribution
    - number of contributions

total contributions

```
camp_2014 |>
  group_by(candidate) |>
  summarise(tot_con = sum(Contribution_Amount, na.rm = T)) |>
  arrange(-tot_con)|>
  select(candidate, tot_con) |>
  head(5)
```

```
# A tibble: 5 x 2
  candidate     tot_con
  <chr>           <int>
1 Tory, John    2767865
2 Chow, Olivia  1638261
3 Ford, Doug     889895
4 Ford, Rob      387645
5 Stintz, Karen  242805
```

mean contribution

```
camp_2014 |>
  group_by(candidate) |>
  summarise(mean_con = mean(Contribution_Amount, na.rm = TRUE)) |>
  arrange(-mean_con)|>
  select(candidate, mean_con) |>
  head(5)
```

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

number of contributions

```
camp_2014 |>
  group_by(candidate) |>
  summarise(num_con = n()) |>
  arrange(-num_con)|>
  select(candidate, num_con) |>
  head(5)
```

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

Q7

7. Repeat 6 but without contributions from the candidates themselves.

```
d7 <- camp_2014 |>
  filter(relationship_to_candidate != 'Candidate' | is.na(relationship_to_candidate))
```

total contributions

```
d7 |>
  group_by(candidate) |>
  summarise(tot_con = sum(Contribution_Amount, na.rm = TRUE)) |>
  arrange(-tot_con)|>
  select(candidate, tot_con) |>
  head(5)
```

```
# A tibble: 5 x 2
  candidate     tot_con
  <chr>           <int>
1 Tory, John    2765365
2 Chow, Olivia  1635761
```

```
3 Ford, Doug     331171
4 Stintz, Karen  242805
5 Ford, Rob      174508
```

mean contribution

```
d7 |>
  group_by(candidate) |>
  summarise(mean_con = mean(Contribution_Amount)) |>
  arrange(-mean_con)|>
  select(candidate, mean_con) |>
  head(5)
```

```
# A tibble: 5 x 2
  candidate         mean_con
  <chr>                <dbl>
1 Ritch, Carlie        1887.
2 Sniedzins, Erwin     1867.
3 Tory, John           1063.
4 Gardner, Norman      1000
5 Tiwari, Ramnarine    1000
```

number of contributions

```
d7 |>
  group_by(candidate) |>
  summarise(num_con = n()) |>
  arrange(-num_con)|>
  select(candidate, num_con) |>
  head(5)
```

```
# A tibble: 5 x 2
  candidate       num_con
  <chr>             <int>
1 Chow, Olivia       5707
2 Tory, John         2601
3 Ford, Doug          608
4 Ford, Rob           531
5 Soknacki, David     314
```

Q8

8. How many contributors gave money to more than one candidate?

```
camp_2014 |>
  select(contributors_name, candidate)|>
  distinct()|>
  group_by(contributors_name) |>
  summarise(num_con = n()) |>
  filter(num_con>1) |>
  dim()
```

```
[1] 184   2
```

184 contributors gave money to more than one candidate