

Week 2 Lab

1. Using the `delay_2022` data, plot the five stations with the highest mean delays. Facet the graph by `line`
2. Using the `opendatatoronto` package, download the data on mayoral campaign contributions for 2014. 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
3. Clean up the data format (fixing the parsing issue and standardizing the column names using `janitor`)
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.
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.
6. List the top five candidates in each of these categories:
 - total contributions
 - mean contribution
 - number of contributions
7. Repeat 5 but without contributions from the candidates themselves.
8. How many contributors gave money to more than one candidate?

```
library(opendatatoronto)
```

```
## Warning: package 'opendatatoronto' was built under R version 4.2.2
```

```
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
## Warning: package 'tibble' was built under R version 4.2.2
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

```
library(stringr)
library(skimr) # EDA
```

```
## Warning: package 'skimr' was built under R version 4.2.2
```

```
library(visdat) # EDA
```

```
## Warning: package 'visdat' was built under R version 4.2.2
```

```
library(janitor)
```

```
## Warning: package 'janitor' was built under R version 4.2.2
```

```
library(lubridate)
library(ggrepel)
```

```
## Warning: package 'ggrepel' was built under R version 4.2.2
```

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

```
## # A tibble: 6 x 11
##   title      id    topics civic~1 publi~2 excerpt datas~3 num_r~4 formats refre~5
##   <chr>    <chr> <chr> <chr> <chr> <chr> <chr> <int> <chr> <chr>
## 1 Building~ f2ca~ Devel~ <NA>  Toront~ Permit~ Table      5 XLS,JS~ Weekly
## 2 Building~ 9425~ Devel~ Climat~ Toront~ Green ~ Table      5 XLS,JS~ Weekly
## 3 Daily Sh~ 21c8~ Commu~ Afford~ Shelte~ Daily ~ Table     12 JSON,C~ Daily
## 4 Traffic ~ a330~ Trans~ <NA>  Transp~ This d~ Map      12 XSD,SH~ As ava~
## 5 BodySafe c405~ City ~ <NA>  Toront~ This d~ Map      9 GPKG,S~ Daily
## 6 Chemical~ ae8e~ Publi~ <NA>  Toront~ This d~ Table      6 XML,JS~ Daily
## # ... with 1 more variable: last_refreshed <date>, and abbreviated variable
## #   names 1: civic_issues, 2: publisher, 3: dataset_category, 4: num_resources,
## #   5: refresh_rate
```

```
res <- list_package_resources("996cfe8d-fb35-40ce-b569-698d51fc683b")
# obtained code from searching data frame above
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)
```

1.

Get the five stations with the highest mean delay. We will only consider stations that appear at least more than once.

```
freq_table = data.frame(table(delay_2022$station))

truth = c()
for (i in 1:nrow(delay_2022)){
  if (freq_table$Freq[which(freq_table$Var1 == delay_2022$station[i])]>1){
    truth = c(truth, TRUE)
  }
}
```

```

else{
  truth = c(truth, FALSE)
}
}

temp_1 = delay_2022
temp_1$truth = truth

temp_2 = temp_1 |>
  filter(temp_1$truth)

temp_3 = temp_2 |>
  group_by(station)

summarize(temp_3, Mean_Delay = mean(min_delay)) |>
  arrange(desc(Mean_Delay))

```

```

## # A tibble: 158 x 2
##   station          Mean_Delay
##   <chr>             <dbl>
## 1 SHEPPARD WEST TO WILSO      20
## 2 VICTORIA PARK TO WARDE     16
## 3 SRT LINE                  14.5
## 4 UNION STATION - ST AND     14
## 5 ELLESMERE STATION          12.1
## 6 GUNN BUILDING              11.2
## 7 KENNEDY SRT STATION        10.9
## 8 SHEPPARD LINE              9.5
## 9 WILSON SOUTH HOSTLER        9
## 10 DUPONT STATION             7.26
## # ... with 148 more rows

```

```

top_5_mean = delay_2022 |>
  filter(station == "SHEPPARD WEST TO WILSO"|
         station == "VICTORIA PARK TO WARDE"|
         station == "SRT LINE"|
         station == "UNION STATION - ST AND "|
         station == "ELLESMERE STATION")
ggplot(data = top_5_mean)+
  geom_histogram(aes(x=min_delay))+
  facet_wrap(~line)+
  labs(title = "Histogram of minute delays in five stations with the highest mean delay")+
  xlab("Minutes Delayed")+
  theme_bw()

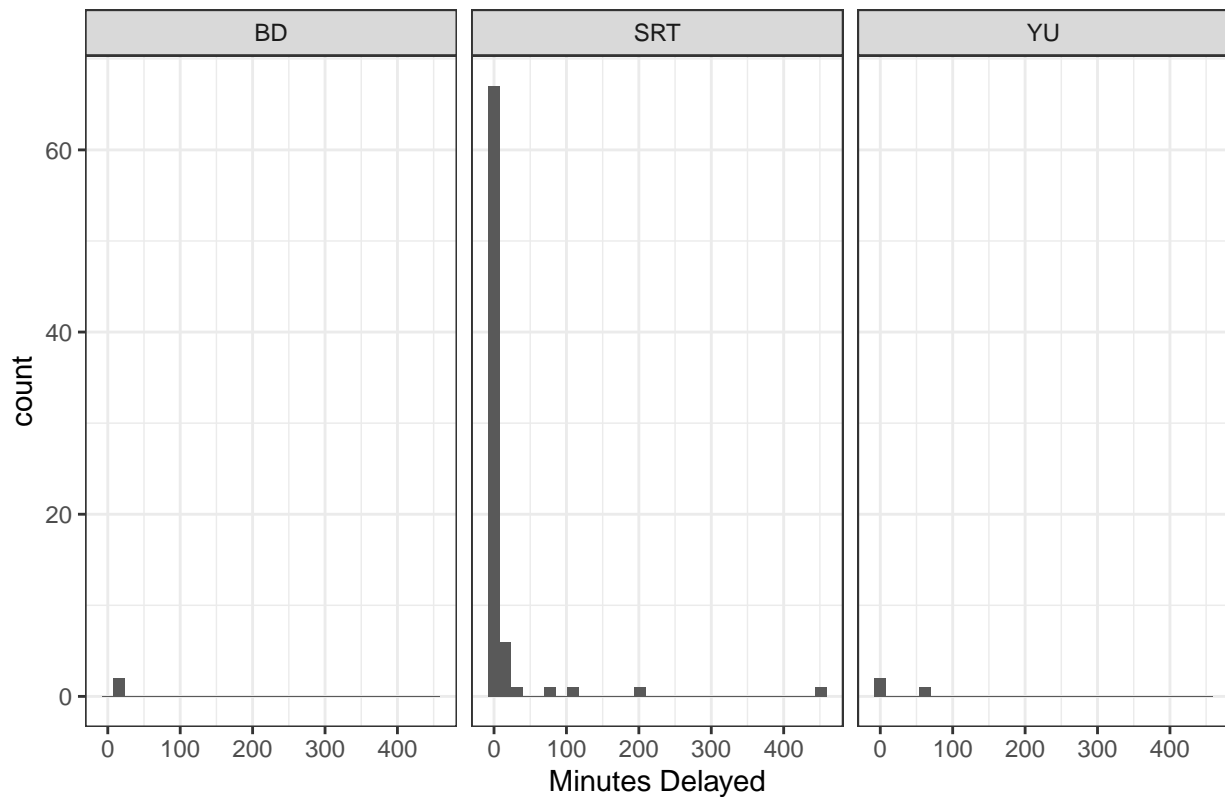
```

```

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

```

Histogram of minute delays in five stations with the highest mean delay



2.

```
library(opendatatoronto)
#search by title
campaign_package = search_packages("campaign contributions 2014")

#get the id
id_2014 = campaign_package$id

temp = list_package_resources(id_2014)

temp
```

```
## # A tibble: 2 x 4
##   name                                id                                format last_mod~1
##   <chr>                                <chr>                                <chr> <date>
## 1 campaign-contributions-2014-data    5b230e92-0a22-4a15-9~ ZIP    2019-07-23
## 2 campaign-contributions-2014-readme-xls aaf736f4-7468-4bda-9~ XLS    2019-07-23
## # ... with abbreviated variable name 1: last_modified
```

```
#5b230e92-0a22-4a15-9572-0b19cc222985 is the id
```

```
mayor_2014 = get_resource("5b230e92-0a22-4a15-9572-0b19cc222985")$`2_Mayor_Contributions_2014_election..`
```

```
## New names:
```

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

```
head(mayor_2014)
```

```
## # A tibble: 6 x 13
##   2014 Munic~1 ...2 ...3 ...4 ...5 ...6 ...7 ...8 ...9 ...10 ...11 ...12
##   <chr>      <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 Contributor~ Cont~ Cont~ Cont~ Cont~ Good~ Cont~ Rela~ Pres~ Auth~ Cand~ Offi~
## 2 A D'Angelo,~ <NA> M6A ~ 300 Mone~ <NA> Indi~ <NA> <NA> <NA> Ford~ Mayor
## 3 A Strazar, ~ <NA> M2M ~ 300 Mone~ <NA> Indi~ <NA> <NA> <NA> Ford~ Mayor
## 4 A'Court, K ~ <NA> M4M ~ 36 Mone~ <NA> Indi~ <NA> <NA> <NA> Chow~ Mayor
## 5 A'Court, K ~ <NA> M4M ~ 100 Mone~ <NA> Indi~ <NA> <NA> <NA> Chow~ Mayor
## 6 A'Court, K ~ <NA> M4M ~ 100 Mone~ <NA> Indi~ <NA> <NA> <NA> Chow~ Mayor
## # ... with 1 more variable: ...13 <chr>, and abbreviated variable name
## # 1: '2014 Municipal Election - List of Contributors to Mayoralty Candidates'
```

We will clean the data set in 3.

3.

```
library(janitor)
#we can fix the parsing issue
mayor_2014 = row_to_names(mayor_2014, row_number= 1)

head(mayor_2014)
```

```
## # A tibble: 6 x 13
##   Contributor'~1 Contr~2 Contr~3 Contr~4 Contr~5 Goods~6 Contr~7 Relat~8 Presi~9
##   <chr>      <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 A D'Angelo, T~ <NA> M6A 1P5 300 Moneta~ <NA> Indivi~ <NA> <NA>
## 2 A Strazar, Ma~ <NA> M2M 3B8 300 Moneta~ <NA> Indivi~ <NA> <NA>
## 3 A'Court, K Su~ <NA> M4M 2J8 36 Moneta~ <NA> Indivi~ <NA> <NA>
## 4 A'Court, K Su~ <NA> M4M 2J8 100 Moneta~ <NA> Indivi~ <NA> <NA>
## 5 A'Court, K Su~ <NA> M4M 2J8 100 Moneta~ <NA> Indivi~ <NA> <NA>
## 6 Aaron, Robert~ <NA> M6B 1H7 250 Moneta~ <NA> Indivi~ <NA> <NA>
## # ... with 4 more variables: 'Authorized Representative' <chr>,
## # Candidate <chr>, Office <chr>, Ward <chr>, and abbreviated variable names
## # 1: 'Contributor's Name', 2: 'Contributor's Address',
## # 3: 'Contributor's Postal Code', 4: 'Contribution Amount',
## # 5: 'Contribution Type Desc', 6: 'Goods or Service Desc',
## # 7: 'Contributor Type Desc', 8: 'Relationship to Candidate',
## # 9: 'President/ Business Manager'
```

```
#and standardize the column names
mayor_2014 = clean_names(mayor_2014)

head(mayor_2014)
```

```
## # A tibble: 6 x 13
##   contributors~1 contr~2 contr~3 contr~4 contr~5 goods~6 contr~7 relat~8 presi~9
##   <chr>          <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <chr>    <chr>
## 1 A D'Angelo, T~ <NA>    M6A 1P5 300    Moneta~ <NA>    Indivi~ <NA>    <NA>
## 2 A Strazar, Ma~ <NA>    M2M 3B8 300    Moneta~ <NA>    Indivi~ <NA>    <NA>
## 3 A'Court, K Su~ <NA>    M4M 2J8 36     Moneta~ <NA>    Indivi~ <NA>    <NA>
## 4 A'Court, K Su~ <NA>    M4M 2J8 100    Moneta~ <NA>    Indivi~ <NA>    <NA>
## 5 A'Court, K Su~ <NA>    M4M 2J8 100    Moneta~ <NA>    Indivi~ <NA>    <NA>
## 6 Aaron, Robert~ <NA>    M6B 1H7 250    Moneta~ <NA>    Indivi~ <NA>    <NA>
## # ... with 4 more variables: authorized_representative <chr>, candidate <chr>,
## #   office <chr>, ward <chr>, and abbreviated variable names
## #   1: contributors_name, 2: contributors_address, 3: contributors_postal_code,
## #   4: contribution_amount, 5: contribution_type_desc,
## #   6: goods_or_service_desc, 7: contributor_type_desc,
## #   8: relationship_to_candidate, 9: president_business_manager
```

4.

We can first check for missing values using skim package.

```
skim(mayor_2014)
```

Table 1: Data summary

| | |
|------------------------|------------|
| Name | mayor_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 |

| skim_variable | n_missing | complete_rate | min | max | empty | n_unique | whitespace |
|---------------------------|-----------|---------------|-----|-----|-------|----------|------------|
| 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 |

We first see that the ward variable is useless as all the values are missing. We also remove other variables that have that have 99% of the values missing, except the 8th column which describes the relationship of the contributor to the candidate.

```
mayor_2014 = mayor_2014 |>
  select(-c(2,6,9,10,13))
```

The relationship to candidate column is relevant.

```
mayor_2014$relationship_to_candidate[which(!is.na(mayor_2014$relationship_to_candidate))]
```

```
## [1] "Candidate" "Candidate" "Candidate" "Candidate" "Candidate" "Candidate"
## [7] "Candidate" "Candidate" "Candidate" "Candidate" "Candidate" "Candidate"
## [13] "Candidate" "Candidate" "Candidate" "Candidate" "Spouse" "Spouse"
## [19] "Candidate" "Candidate" "Candidate" "Candidate" "Candidate" "Candidate"
## [25] "Candidate" "Candidate" "Candidate" "Candidate" "Candidate" "Candidate"
## [31] "Candidate" "Candidate" "Spouse"
```

We want to know whether or not candidate made any contribution for 7. Hence, we make a new column where it's 1 if the candidate contributed themselves, and 0 otherwise.

```
#replace NA with FALSE
mayor_2014$relationship_to_candidate[which(is.na(mayor_2014$relationship_to_candidate))]=
  FALSE
#make a column of 1 if the contributor is candidate
#0 otherwise
mayor_2014 = mayor_2014 |>
  mutate(relationship = ifelse(relationship_to_candidate == "Candidate",1,0)) |>
  select(-6)
```

Contribution amount should be in numeric.

```
mayor_2014$contribution_amount=as.numeric(mayor_2014$contribution_amount)
```

The “Office” variable has the same value for each data point, so we may remove it.

```
unique(mayor_2014$office)
```

```
## [1] "Mayor"
```

```
mayor_2014 = mayor_2014 |>
  select(-7)
```

We will first look at summary statistics for the contribution amount.

```
summary(mayor_2014$contribution_amount)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##         1      100     300     608     500 508225
```

The median is smaller than the mean by 308 dollars, so this may suggest some higher-end donations that skew the distribution.

The maximum of 508225 dollars is also notable.

Next, we will look at composition of the contribution type.

```
unique(mayor_2014$contribution_type_desc)
```

```
## [1] "Monetary"      "Goods/Services"
```

```
temp = mayor_2014$contribution_type_desc
#percentage of monetary donation
(length(which(temp == "Monetary"))/length(temp) )*100
```

```
## [1] 99.89215
```

And we look at the composition of the contributor type.

```
unique(mayor_2014$contributor_type_desc)
```

```
## [1] "Individual"  "Corporation"
```

```
temp = mayor_2014$contributor_type_desc
#percentage of individual donation
(length(which(temp == "Individual"))/length(temp) )*100
```

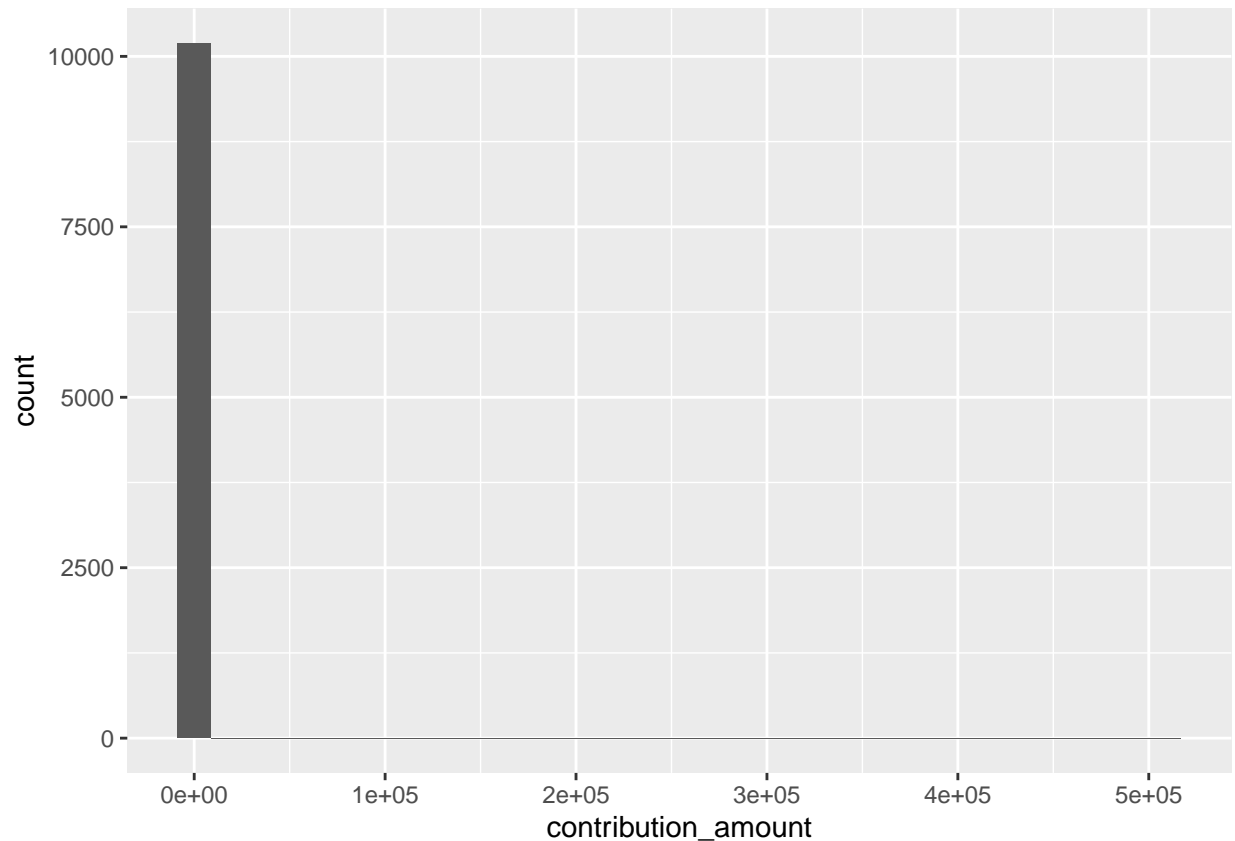
```
## [1] 99.98039
```

5.

We first graph a histogram of the contribution amounts.

```
ggplot(data = mayor_2014, aes(x=contribution_amount))+
  geom_histogram()
```

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

The histogram breaks because of outliers. We will remove contributions of more than 50000 dollars. There are 2 of them.

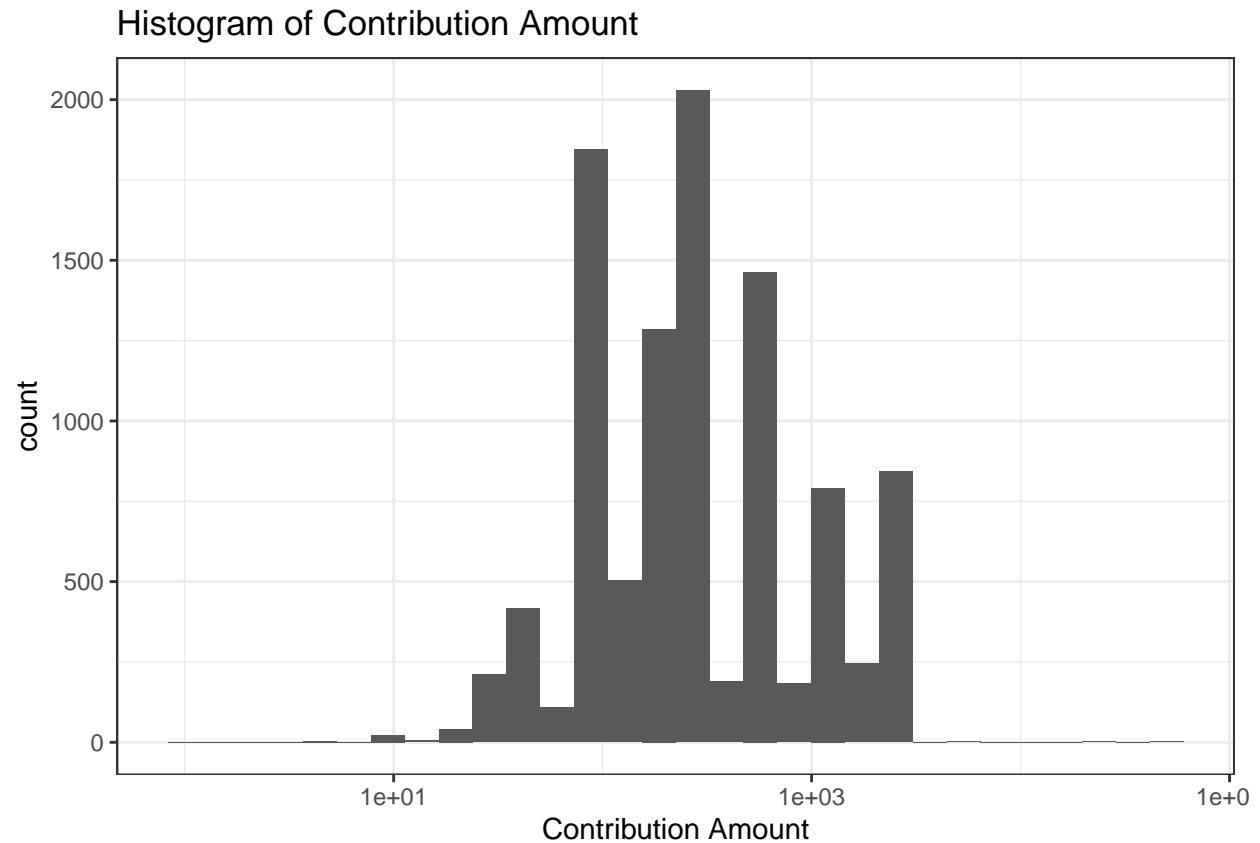
```
length(which(mayor_2014$contribution_amount > 50000))
```

```
## [1] 2
```

```
temp = mayor_2014 |>
  filter(contribution_amount <=50000)

ggplot(data = temp)+
  geom_histogram(aes(x=contribution_amount))+
  xlab("Contribution Amount")+
  scale_x_log10()+
  labs(title = "Histogram of Contribution Amount")+
  theme_bw()
```

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

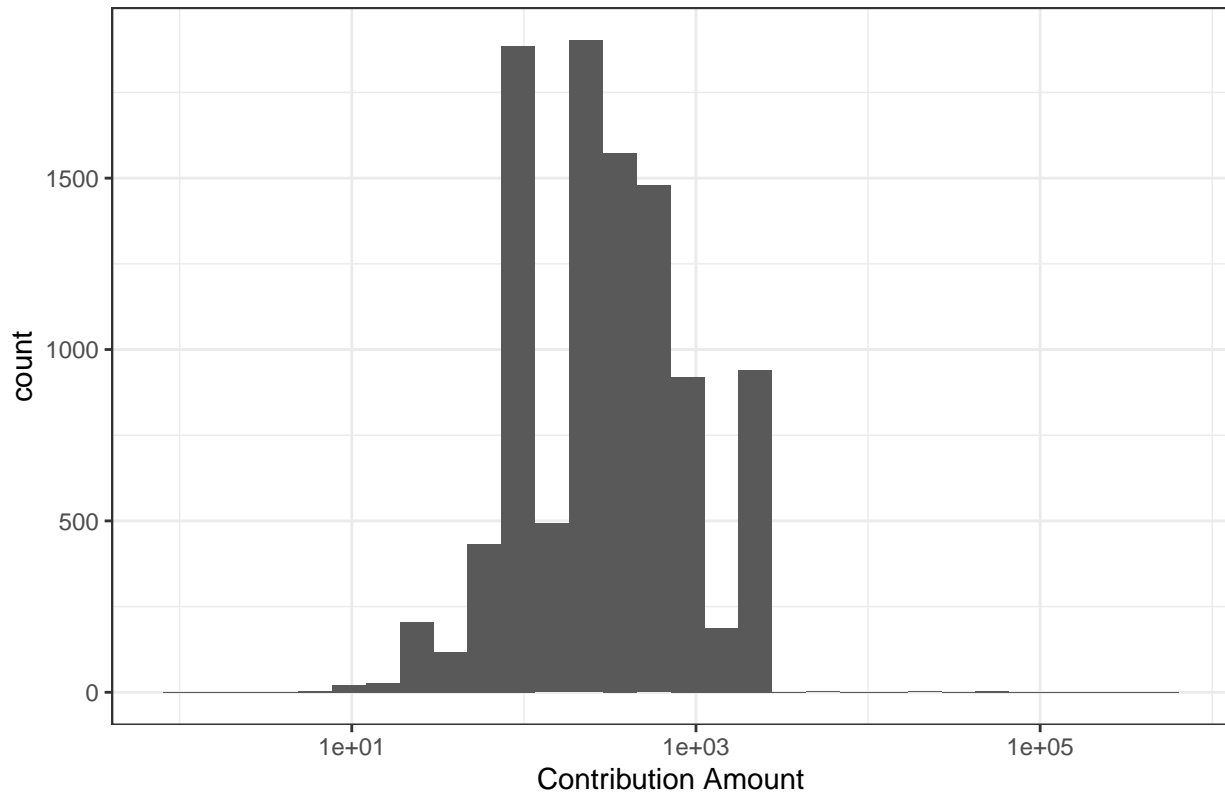


*#We can also plot it on the log-scale
#in which case we do not need to remove the outlier*

```
ggplot(data = mayor_2014)+  
  geom_histogram(aes(x=contribution_amount))+  
  xlab("Contribution Amount")+  
  scale_x_log10()+  
  labs(title = "Histogram of Contribution Amount, log-scale")+  
  theme_bw()
```

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

Histogram of Contribution Amount, log-scale



Notably, there's an outlier with the contribution of 508224.73.

```
mayor_2014 |>
  filter(contribution_amount == 508224.73)

## # A tibble: 1 x 7
##   contributors_name contributors_postal_code contr-2 contr-3 contr-4 candi-5 relat-6
##   <chr>           <chr>                  <dbl> <chr>   <chr>   <chr>   <dbl>
## 1 Ford, Doug      M9A 2C3                      508225. Moneta~ Indivi~ Ford, ~     1
## # ... with abbreviated variable names 1: contributors_postal_code,
## #   2: contribution_amount, 3: contribution_type_desc,
## #   4: contributor_type_desc, 5: candidate, 6: relationship
```

This contribution turns out to be Doug Ford self-financing his own campaign.

A frequency table can also be interesting. We print top 15 contribution amount with respect to frequency.

```
sort(table(mayor_2014$contribution_amount), decreasing = TRUE)[1:15]
```

```
##
## 100  500  300  200 2500 1000  250   50  150   25  750 1500  400 2000   60
## 1767 1392 1367 1245  825  735  605  323  243  181  133  130  118   93   92
```

Another thing of interest is the number of “small” contribution. OpenSecrets, a non-profit organization that tracks campaign financing, defines small contributions as contribution of less than 200 dollars. So we will use this definition.

```
small = length(which(mayor_2014$contribution_amount < 200))

#the percentage of small contribution

small/nrow(mayor_2014) * 100
```

```
## [1] 31.28738
```

So less than half of the donations were “small” contributions.

6.

```
temp = mayor_2014 |>
  group_by(candidate)

total_contribution = temp|>
  summarize(Total = sum(contribution_amount))|>
  arrange(desc(Total))

total_contribution
```

```
## # A tibble: 27 x 2
##   candidate      Total
##   <chr>         <dbl>
## 1 Tory, John    2767869.
## 2 Chow, Olivia  1638266.
## 3 Ford, Doug    889897.
## 4 Ford, Rob     387648.
## 5 Stintz, Karen 242805
## 6 Soknacki, David 132431
## 7 Goldkind, Ari  41125.
## 8 Thomson, Sarah 34628.
## 9 Di Paola, Rocco 21126
## 10 Underhill, Richard 15660
## # ... with 17 more rows
```

So the top five candidates in total contribution is Tory, Chow, Ford, Doug, Ford Rob, and Stintz.

```
temp = mayor_2014 |>
  group_by(candidate)

mean_contribution = temp|>
  summarize(Mean = mean(contribution_amount))|>
  arrange(desc(Mean))

mean_contribution
```

```
## # A tibble: 27 x 2
##   candidate      Mean
##   <chr>         <dbl>
## 1 Sniedzins, Erwin 2025
```

```
## 2 Syed, Himy      2018
## 3 Ritch, Carlie   1887.
## 4 Ford, Doug      1456.
## 5 Clarke, Kevin   1200
## 6 Di Paola, Rocco 1174.
## 7 Tory, John      1064.
## 8 Gardner, Norman 1000
## 9 Stintz, Karen    995.
## 10 Kalevar, Chai   900
## # ... with 17 more rows
```

So the top five candidates in mean contribution is Sniedzin, Syed, Ritch, Ford, Doug and Clarke.

```
temp = mayor_2014 |>
  group_by(candidate)

num_contribution = temp|>
  summarize(num = length(contribution_amount))|>
  arrange(desc(num))

num_contribution
```

```
## # A tibble: 27 x 2
##   candidate      num
##   <chr>         <int>
## 1 Chow, Olivia   5708
## 2 Tory, John     2602
## 3 Ford, Doug      611
## 4 Ford, Rob       538
## 5 Soknacki, David 314
## 6 Stintz, Karen   244
## 7 Goldkind, Ari   47
## 8 Underhill, Richard 41
## 9 Thomson, Sarah  40
## 10 Di Paola, Rocco 18
## # ... with 17 more rows
```

So the top five candidates in number of contribution is Chow, Tory, Ford, Doug, Ford, Rob and Soknacki.

7.

We remove donation made by the candidates themselves

```
remove_candidate = mayor_2014 |>
  filter(relationship == 0)

#the maximum donation is now 3660.

max(remove_candidate$contribution_amount)
```

```
## [1] 3660
```

We repeat Q6.

```
temp = remove_candidate |>
  group_by(candidate)

total_contribution = temp|>
  summarize(Total = sum(contribution_amount))|>
  arrange(desc(Total))

total_contribution
```

```
## # A tibble: 17 x 2
##   candidate      Total
##   <chr>         <dbl>
## 1 Tory, John    2765369.
## 2 Chow, Olivia  1635766.
## 3 Ford, Doug    331173.
## 4 Stintz, Karen 242805
## 5 Ford, Rob     174510.
## 6 Soknacki, David 132431
## 7 Thomson, Sarah 27702.
## 8 Goldkind, Ari  17501
## 9 Underhill, Richard 15660
## 10 Di Paola, Rocco 15126
## 11 Ritch, Carlie  5660
## 12 Sniedzins, Erwin 5600
## 13 Gardner, Norman 3000
## 14 Baskin, Morgan 1550
## 15 Billard, Jeff  1486.
## 16 Tiwari, Ramnarine 1000
## 17 Lam, Steven    300
```

So the top five candidates in total contribution is Tory, Chow, Ford, Doug, Stintz, Rob, Ford.

```
temp = remove_candidate|>
  group_by(candidate)

mean_contribution = temp|>
  summarize(Mean = mean(contribution_amount))|>
  arrange(desc(Mean))

mean_contribution
```

```
## # A tibble: 17 x 2
##   candidate      Mean
##   <chr>         <dbl>
## 1 Ritch, Carlie  1887.
## 2 Sniedzins, Erwin 1867.
## 3 Tory, John     1063.
## 4 Gardner, Norman 1000
## 5 Tiwari, Ramnarine 1000
## 6 Stintz, Karen   995.
## 7 Di Paola, Rocco  890.
```

```
## 8 Thomson, Sarah      729.
## 9 Ford, Doug          545.
## 10 Billard, Jeff       496.
## 11 Soknacki, David     422.
## 12 Underhill, Richard 382.
## 13 Goldkind, Ari       380.
## 14 Ford, Rob           329.
## 15 Lam, Steven         300.
## 16 Chow, Olivia       287.
## 17 Baskin, Morgan     194.
```

So the top five candidates in mean contribution is Ritch, Sniedzins, Tory, Gardner, and Tiwari.

```
temp = remove_candidate |>
  group_by(candidate)

num_contribution = temp|>
  summarize(num = length(contribution_amount))|>
  arrange(desc(num))

num_contribution
```

```
## # A tibble: 17 x 2
##   candidate      num
##   <chr>         <int>
## 1 Chow, Olivia   5707
## 2 Tory, John    2601
## 3 Ford, Doug     608
## 4 Ford, Rob     531
## 5 Soknacki, David 314
## 6 Stintz, Karen  244
## 7 Goldkind, Ari   46
## 8 Underhill, Richard 41
## 9 Thomson, Sarah  38
## 10 Di Paola, Rocco 17
## 11 Baskin, Morgan   8
## 12 Billard, Jeff    3
## 13 Gardner, Norman  3
## 14 Ritch, Charlie  3
## 15 Sniedzins, Erwin 3
## 16 Lam, Steven     1
## 17 Tiwari, Ramnarine 1
```

So the top five candidates in number of contribution is Chow, Tory, Ford, Doug, Ford, Rob and Soknacki.

8.

Since it's possible for two people to have the same name, we first filter by the address. We will assume that people with the same name and the same address is a one person, even though it's possible that two people with the same name are living in the same address.

```
#choosing contributor address that appears twice.
temp = mayor_2014|>
  group_by(contributors_postal_code)|>
  filter(length(contributors_postal_code) > 1)

#We then group by name, choosing people that appear twice.

temp = temp|>
  group_by(contributors_name)|>
  filter(length(contributors_name)>1)

#the number of people who made more than one contribution is

length(unique(temp$contributors_name))

## [1] 1849
```