Fall 2022 Data Science Intern Challenge

Question 1

Edward

2022-05-10

```
# Import packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# Import data
q1_data_base <- read_csv(file.path("data", "sheet_1.csv"))
```

```
## Rows: 5000 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): payment_method, created_at
## dbl (5): order_id, shop_id, user_id, order_amount, total_items
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Question 1

Let's take a first look:

```
head(q1_data_base)
```

```
## # A tibble: 6 x 7
##   order_id shop_id user_id order_amount total_items payment_method created_at
##      <dbl>   <dbl>   <dbl>        <dbl>       <dbl> <chr>          <chr>
```

```
## 1         1       53      746            224            2 cash            2017-03-13 1~
## 2         2       92      925             90            1 cash            2017-03-03 1~
## 3         3       44      861            144            1 cash            2017-03-14 4~
## 4         4       18      935            156            1 credit_card     2017-03-26 1~
## 5         5       18      883            156            1 credit_card     2017-03-01 4~
## 6         6       58      882            138            1 credit_card     2017-03-14 1~
```

In terms of data types, we can see that 5 variables were read as doubles, while the remaining two are read as character. Some modifications are pertinent: For example, all id variables should be characters because they don't have any statistical meaning:

```
# Transform id variables into characters. Note we are saving these changes into
# a new data frame
q1_data <- q1_data_base %>% mutate_at(vars(contains("id")), as.character)
```

Moreover, *created at* has been imported as a character but it would be more useful as a datetime variable:

```
# Transform created_at into dttm
q1_data <- q1_data %>% mutate(created_at = ymd_hms(created_at))
```

Now let's look at a brief summary of the dataset:

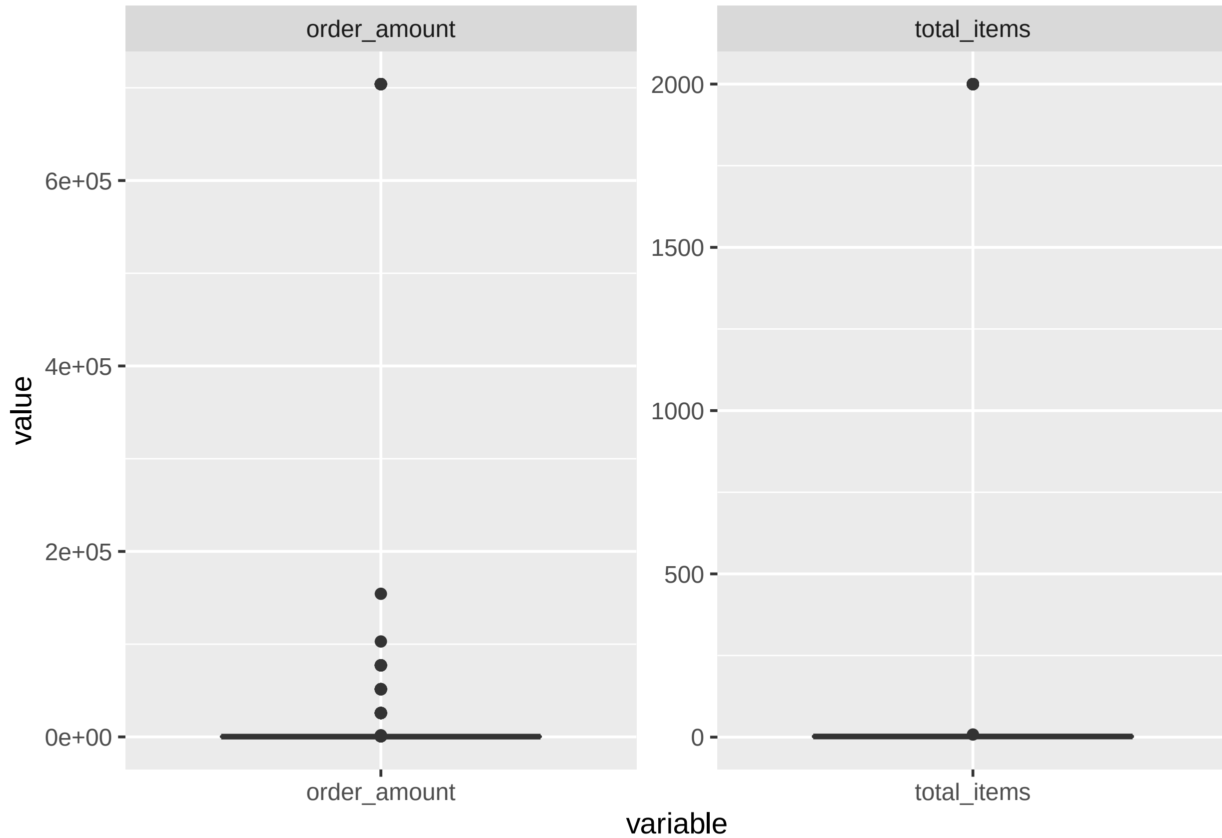
```
# The database's dimension (rows and columns)
dim(q1_data)
```

```
## [1] 5000    7
```

```
# Summary of the numerical and date sample
summary(q1_data %>% select(order_amount, total_items, created_at))
```

```
##   order_amount     total_items         created_at
##  Min.   :    90   Min.   :   1.000   Min.   :2017-03-01 00:08:09.00
##  1st Qu.:   163   1st Qu.:   1.000   1st Qu.:2017-03-08 07:08:03.75
##  Median :   284   Median :   2.000   Median :2017-03-16 00:21:20.50
##  Mean   :  3145   Mean   :   8.787   Mean   :2017-03-15 22:20:37.07
##  3rd Qu.:   390   3rd Qu.:   3.000   3rd Qu.:2017-03-23 10:39:58.25
##  Max.   :704000   Max.   :2000.000   Max.   :2017-03-30 23:55:35.00
```

```
# Histograms and boxplots of order_amount and total_items

q1_data %>%
  select(order_amount, total_items) %>%
  pivot_longer(cols = c(order_amount, total_items),
                       names_to = "variable") %>%
  ggplot(aes(x = variable, y = value)) +
  geom_boxplot() +
  facet_wrap(~ variable, scales = "free")
```



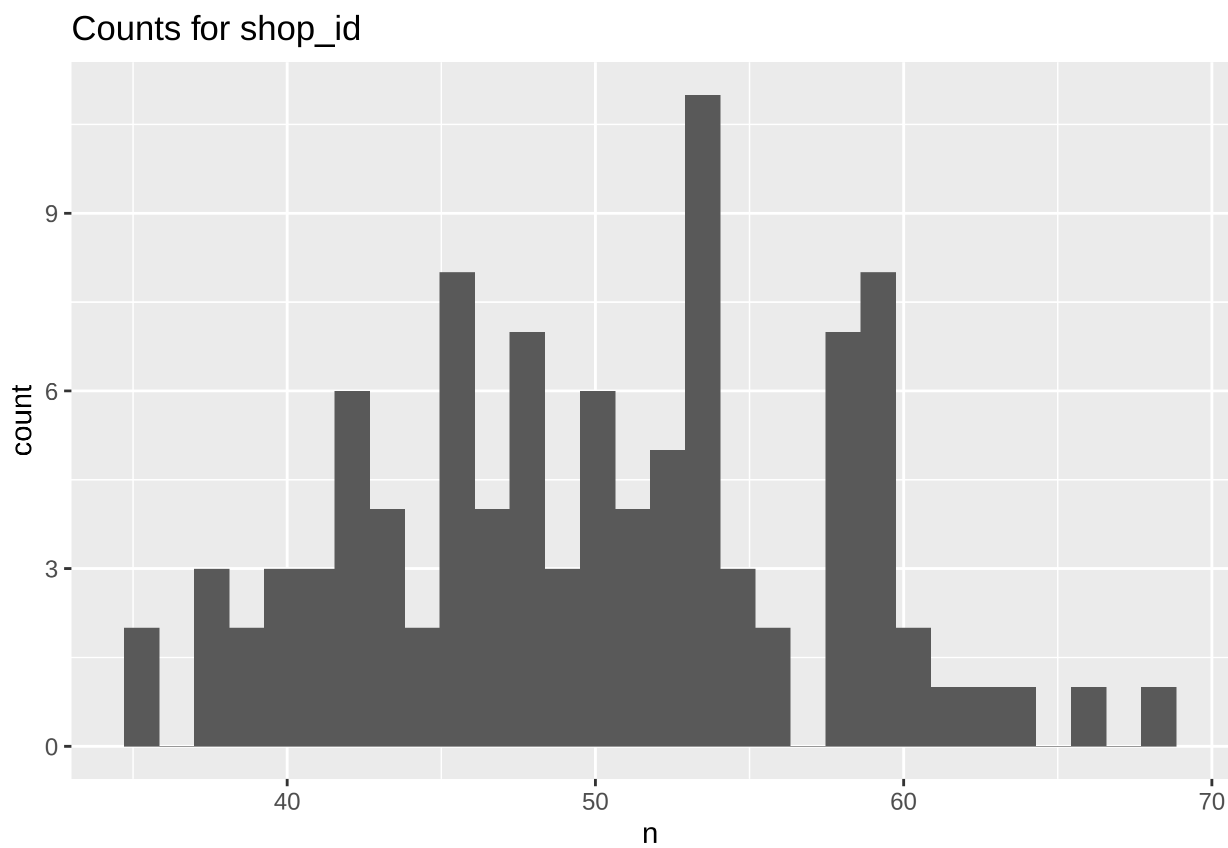

```
# Distribution of the categorical variables (id and payment method)

## Number of distinct categories
q1_data %>% select_if(is.character) %>% map_df(n_distinct)
```

```
## # A tibble: 1 x 4
##   order_id shop_id user_id payment_method
##      <int>   <int>   <int>          <int>
## 1     5000     100     301              3
```

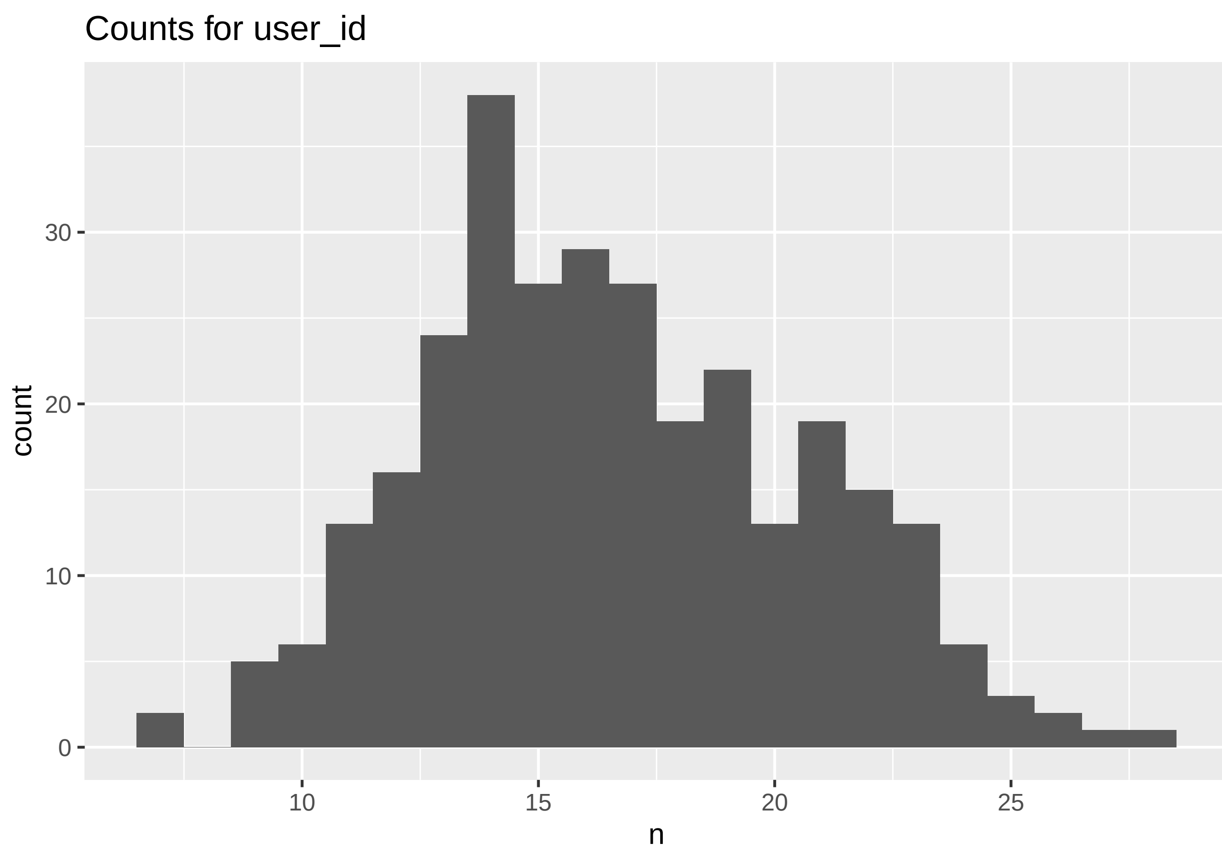
```
## Counts of orders for every category but order_id

## For shop_id
q1_data %>%
  count(shop_id) %>%
  ggplot(aes(x = n)) +
  geom_histogram(binwdith = 1) +
  labs(title = "Counts for shop_id")
```

```
## Warning: Ignoring unknown parameters: binwdith
```

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




```
#3 For user_id
q1_data %>%
  count(user_id) %>%
  ggplot(aes(x = n)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Counts for user_id")
```




```r
# For payment_method
q1_data %>%
  count(payment_method) %>%
  arrange(desc(n))
```

```
## # A tibble: 3 x 2
##   payment_method     n
##   <chr>          <int>
## 1 credit_card     1735
## 2 debit           1671
## 3 cash            1594
```

In the proposed problem, it is said that the AOV (Average Order value) is 3145.128, which is unusually high for just a pair of sneakers. What can be the cause of this? Looking at the numerical summary, we can see that there are huge outliers in both the order_amount (which is the price of an order) and the total_items (which is, presumably, the total number of bought sneaker pairs), which easily distorts the calculated histograms and boxplots. For example, values from order_amount range from 90 to 704000, while total items range between 1 and 2000. These findings suggest a heavily right-skewed distributions for both variables, which influences the average in a significant way (see how the average is higher than the median in both variables). Because of this, the median is more appropiate for the AOV calculation, because it is robust to outliers, so if instead we focus on the median, we have a value of 284€ for the order amount and 2 for total_items.

The previous result cand lead us into another question: Is the median representative enough of the price of a pair of sneakers? Note that some orders in this dataset include more than just one item. For instance, the outliers orders that amounted to 704000 dollars come from orders with a total of 2000 items, as we now show:

```r
q1_data %>%
  filter(order_amount == 704000) %>%
```

```
  arrange(created_at)
```

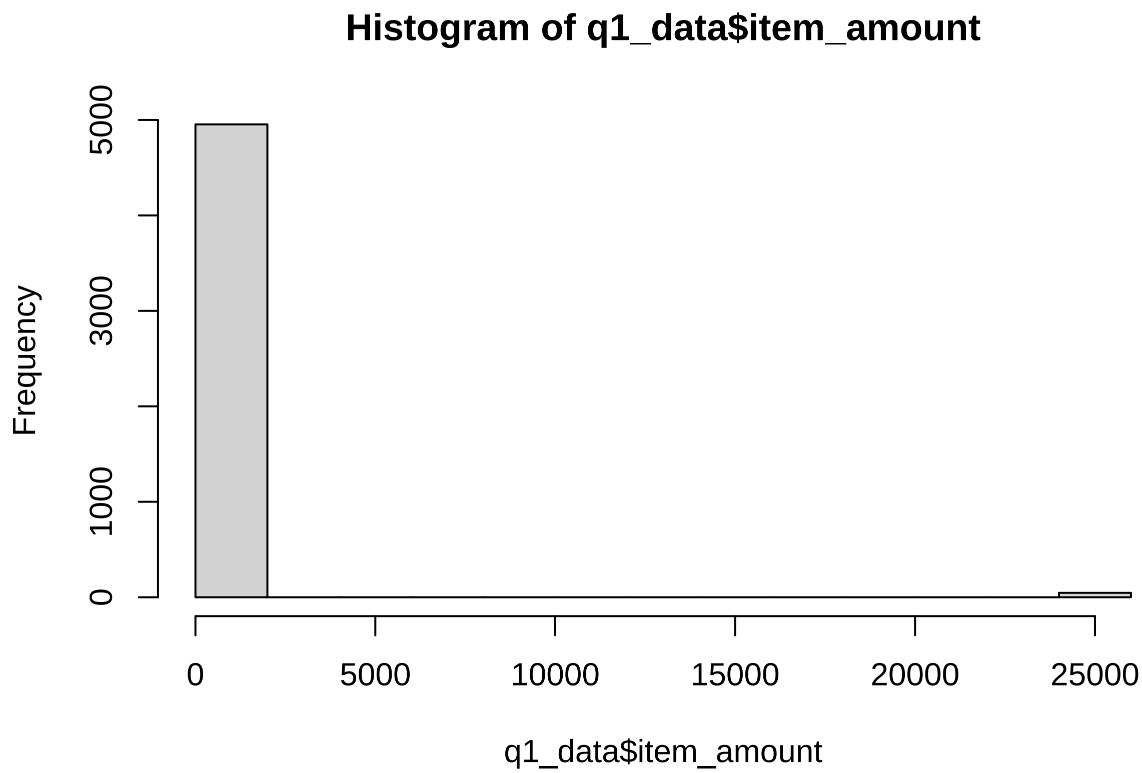
```
## # A tibble: 17 x 7
##    order_id shop_id user_id order_amount total_items payment_method
##    <chr>    <chr>   <chr>          <dbl>       <dbl> <chr>
##  1 521      42      607           704000        2000 credit_card
##  2 4647     42      607           704000        2000 credit_card
##  3 61       42      607           704000        2000 credit_card
##  4 16       42      607           704000        2000 credit_card
##  5 2298     42      607           704000        2000 credit_card
##  6 1437     42      607           704000        2000 credit_card
##  7 2154     42      607           704000        2000 credit_card
##  8 1363     42      607           704000        2000 credit_card
##  9 1603     42      607           704000        2000 credit_card
## 10 1563     42      607           704000        2000 credit_card
## 11 4869     42      607           704000        2000 credit_card
## 12 1105     42      607           704000        2000 credit_card
## 13 3333     42      607           704000        2000 credit_card
## 14 4883     42      607           704000        2000 credit_card
## 15 2836     42      607           704000        2000 credit_card
## 16 2970     42      607           704000        2000 credit_card
## 17 4057     42      607           704000        2000 credit_card
## # ... with 1 more variable: created_at <dttm>
```

If we want an average price for a pair of sneakers (recall that all shops in this dataset sell the same model of sneakers), we would need to divide order_amount by total_items. We will do this now into a new variable called item_amount:

```
# Create new feature that calculates mean amount per item
q1_data <- q1_data %>% mutate(item_amount = order_amount / total_items)

# Distribution of the newly created variable
summary(q1_data$item_amount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    90.0   133.0   153.0   387.7   169.0 25725.0
```

```
hist(q1_data$item_amount)
```




```
q1_data %>% filter(item_amount == max(item_amount))
```

```
## # A tibble: 46 x 8
##    order_id shop_id user_id order_amount total_items payment_method
##    <chr>    <chr>   <chr>          <dbl>       <dbl> <chr>
##  1 161      78      990            25725           1 credit_card
##  2 491      78      936            51450           2 debit
##  3 494      78      983            51450           2 cash
##  4 512      78      967            51450           2 cash
##  5 618      78      760            51450           2 cash
##  6 692      78      878           154350           6 debit
##  7 1057     78      800            25725           1 debit
##  8 1194     78      944            25725           1 debit
##  9 1205     78      970            25725           1 credit_card
## 10 1260     78      775            77175           3 credit_card
## # ... with 36 more rows, and 2 more variables: created_at <dttm>,
## #   item_amount <dbl>
```

In the distribution of the new variable, we can see that shop 78 sells pairs of sneakers that cost 25725 (!!) dollars, which is suspicious. If we treated this as an outlier and removed it from the sample, the median of 284 dollars across orders would still be the same. Additionally, it seems that the median value of the pairs of sneakers, according to item_amount, is 153 dollars, which looks relatively affordable.