Data Audit Markdown

Hive Data Audit Analysis Results (Arseniy Dolgin)

I have decided to preform my analysis in R and submit an RMarkdown HTML report as my answer, as well as the whole R project folder with all the inputs and code.

I noticed that the table given in the prompt was an .xlsx table. I could have opened it in Excel and saved as a CSV, and then used the default "read_csv()" function to load the data into the environment. I decided not to alter the original file manually (in case if I would be expected to preform the same format of analysis in the future), and instead used "readxl" package in R:

```
knitr::opts_chunk$set(echo = TRUE)

library("readxl")
# some additional libraries that I might end up using:
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.4     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```
original_table <-  read_excel("Data/Hive Annotation Job Results.xlsx")

## I run the following command to verify that each column was loaded into the R tibble with a correct d
str(original_table)
```

```
## tibble [5,000 x 7] (S3: tbl_df/tbl/data.frame)
##  $ file           : chr [1:5000] "1650255-9" "1650490-9" "1650594-3" "1649219-5" ...
##  $ object_id      : num [1:5000] 21650 18975 813 2639 362 ...
##  $ tabular        : logi [1:5000] TRUE FALSE TRUE TRUE TRUE TRUE ...
##  $ semantic       : logi [1:5000] TRUE FALSE TRUE TRUE TRUE TRUE ...
##  $ definition_list: logi [1:5000] FALSE TRUE FALSE FALSE TRUE FALSE ...
##  $ header_row     : logi [1:5000] TRUE FALSE TRUE TRUE FALSE TRUE ...
##  $ header_column  : logi [1:5000] TRUE FALSE FALSE FALSE FALSE FALSE ...
```

```
## I have decided to analyse this table by following the rules described in the
# background facts; that is, I formulated the rules as logical statements and my
# plan is to add at least 5 more columns to the table:
#   rule_1_violation; (1 if violated, 0 if not),
#   rule_2_violation; (1 if violated, 0 if not),
#   rule_3_violation; (1 if violated, 0 if not),
#   rule_4_violation; (1 if violated, 0 if not),
#   sum_of_violations; (adds the outputs of the 4 previous columms).
# duplicate the table and do the work there in order not to mess up the original by accident...

table_in_work <- original_table

## Adding column that tests for rule 1:
## if definition list, should be tabular and semantic:
table_in_work$rule_1_violation <- ifelse(table_in_work$definition_list == TRUE & table_in_work$tabular =

## Adding column that tests for rule 2:
## if semantic, should be tabular:
table_in_work$rule_2_violation <- ifelse(table_in_work$semantic == TRUE & table_in_work$tabular == TRUE

## Adding column that tests for rule 3:
## if not tabular, should neither be semantic nor definition list:
table_in_work$rule_3_violation <- ifelse(table_in_work$tabular == FALSE & table_in_work$definition_list

## Adding column that tests for rule 4:
## if tabular and either has a header row or a header column, should be semantic:
## I could not remember the syntax for chaining OR (|) with AND (&) in an if statement,
## so I made a helper column that returns TRUE if either header_row or header_column is TRUE, and retur

table_in_work$helper_column <- ifelse(table_in_work$header_row == TRUE | table_in_work$header_column ==

table_in_work$rule_4_violation <- ifelse(table_in_work$tabular == TRUE & table_in_work$helper_column ==

## remove the helper column, since unlikely to be needed:

table_in_work <- table_in_work[-c(11)]

## create a column that sums up all of the rule violations:
```

```
table_in_work$violations_sum <- table_in_work$rule_1_violation + table_in_work$rule_2_violation + table_
```

At this point the data is ready for analysis visualizations.

First I wanted to see a summary of the last column I created (violations_sum):

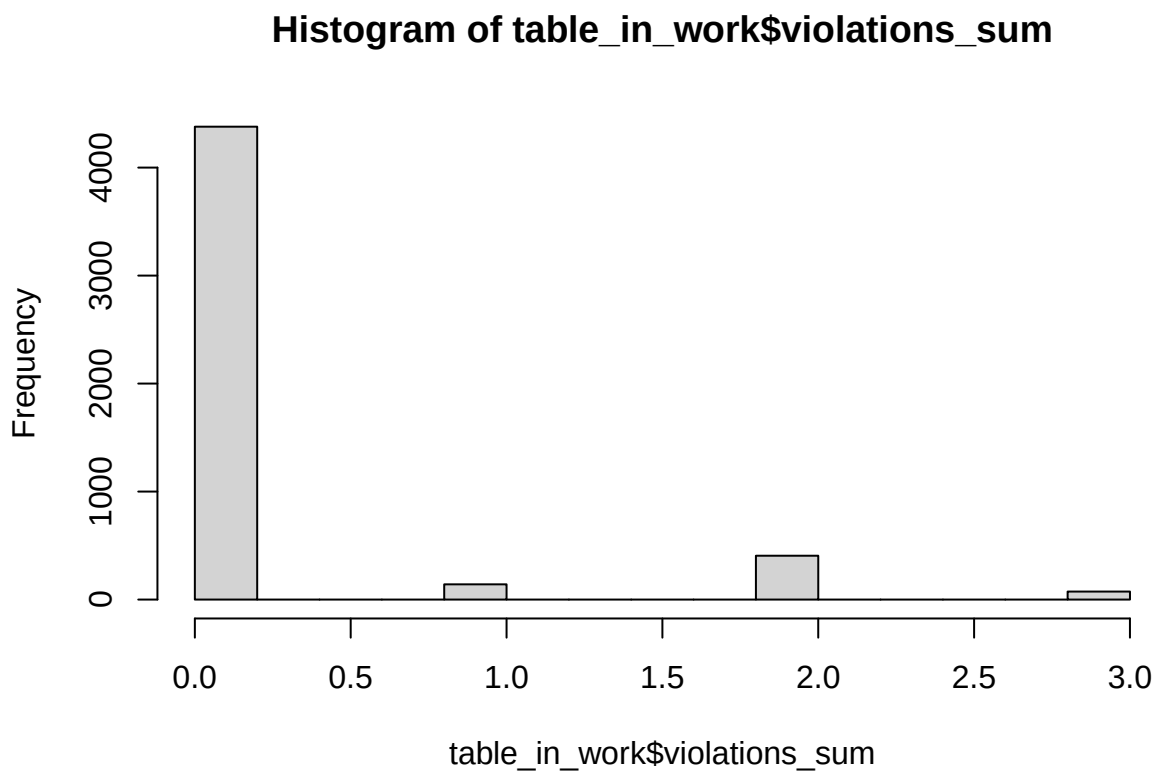
```
summary(table_in_work$violations_sum)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   0.235   0.000   3.000
```

From the printout above we can see that the maximum number of rules a single row has violated is 3.

Next I wanted to see a histogram of how frequently a certain sum of violations appears:

```
hist (table_in_work$violations_sum)
```

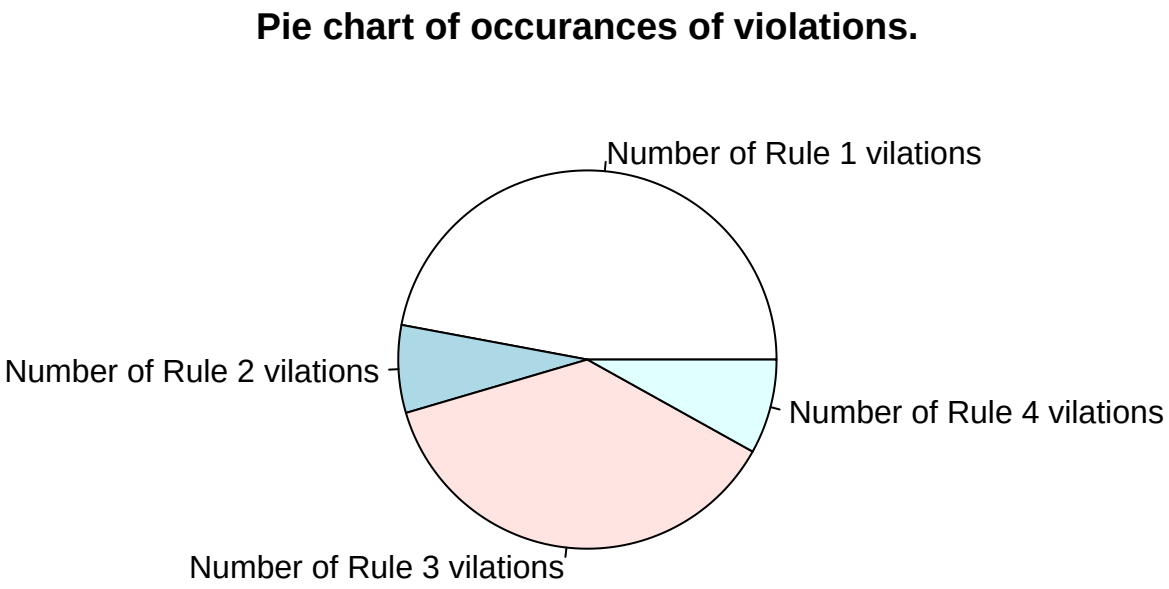



```
table_in_work$violations_sum <- as.numeric(table_in_work$violations_sum)

summary_table <- table(table_in_work$violations_sum)
```

From the histogram we can see that most of the rows had no violations, and 2 violations per row occured more often than either 1 or 3 violations per row.

For the next part I have decided to improve the visual representation of my analysis and build several pie charts that I gradually improved to be more informative.

```
# count the number of times each violation occurs:

sum_rule_1_viol <- sum(table_in_work$rule_1_violation)
sum_rule_2_viol <- sum(table_in_work$rule_2_violation)
sum_rule_3_viol <- sum(table_in_work$rule_3_violation)
sum_rule_4_viol <- sum(table_in_work$rule_4_violation)

# lets build a pie chart of violation sums:

slices <- c(sum_rule_1_viol, sum_rule_2_viol, sum_rule_3_viol, sum_rule_4_viol)
lbls <- c("Number of Rule 1 vilations", "Number of Rule 2 vilations", "Number of Rule 3 vilations", "Nu
pie(slices, labels = lbls, main="Pie chart of occurances of violations.")
```

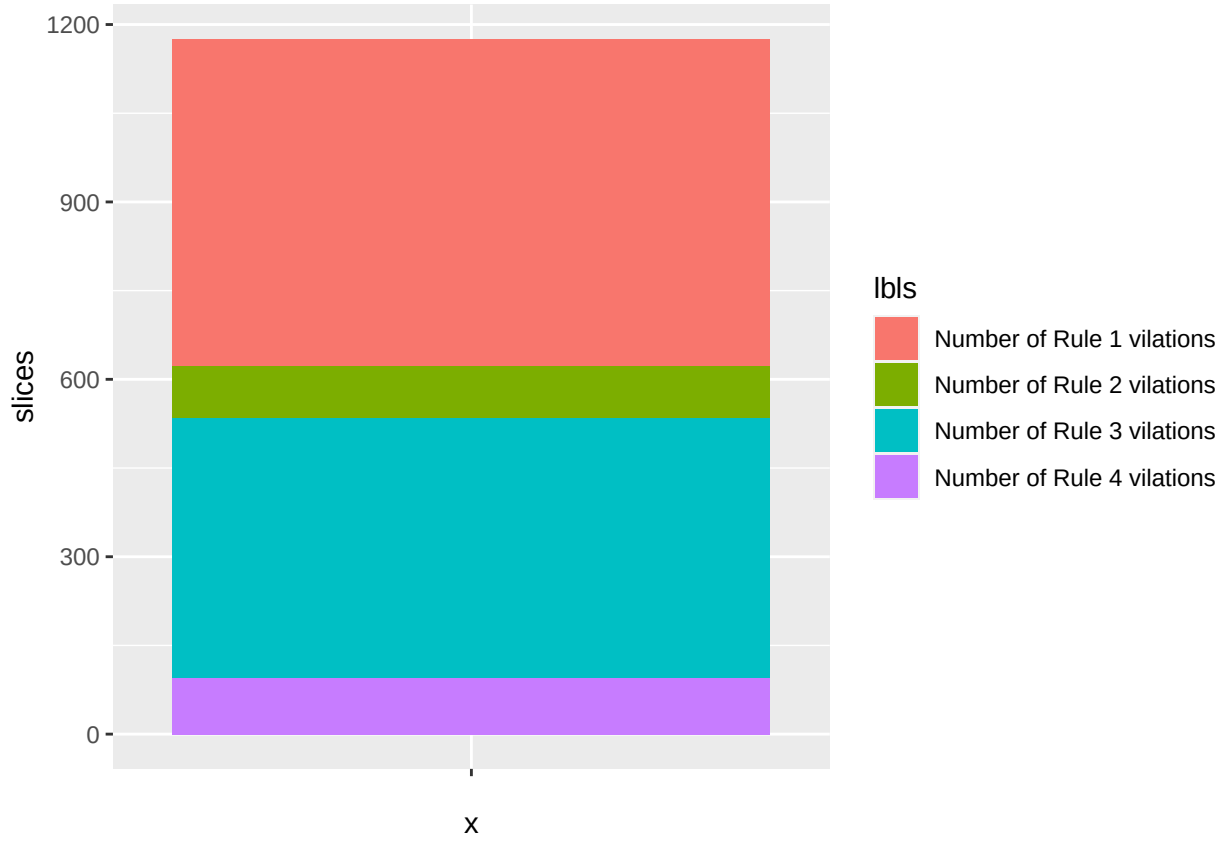



```
# this basic pie chart does not include the count labels inside each slice, a ggplot2 pie chart is need

table_for_pie <- data.frame(lbls, slices)
table_for_pie
```

```
##                          lbls slices
```

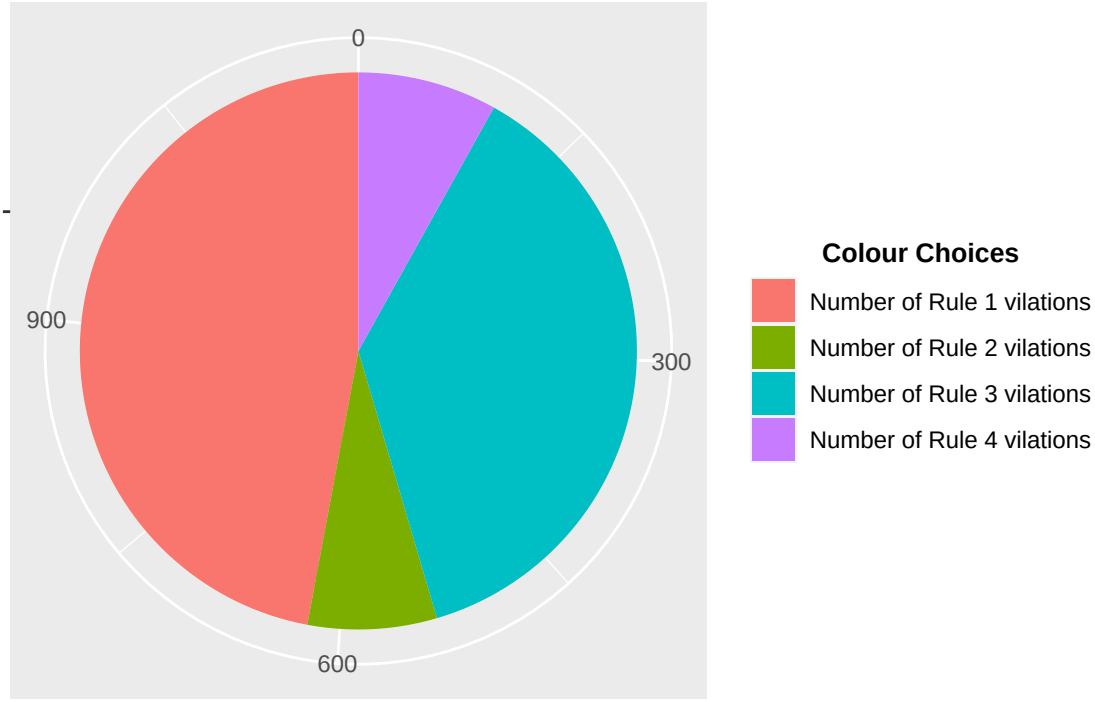
```
## 1 Number of Rule 1 vilations    553
## 2 Number of Rule 2 vilations     88
## 3 Number of Rule 3 vilations    439
## 4 Number of Rule 4 vilations     95
```

```r
# pie chart flows from a bar chart:

ggplot(table_for_pie, aes(x = "", y = slices, fill = lbls)) +
  geom_bar(width = 1, stat = "identity")
```




```r
# first ggplot2 pie chart:

ggplot(table_for_pie, aes(x = "", y = slices, fill = lbls)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar(theta = "y", start = 0) +
  labs(x = "", y = "", title = "Number of each Rule Violation Occurances: \n",
       fill = "Colour Choices") +
  theme(plot.title = element_text(hjust = 0.5),
        legend.title = element_text(hjust = 0.5, face="bold", size = 10))
```

Number of each Rule Violation Occurances:



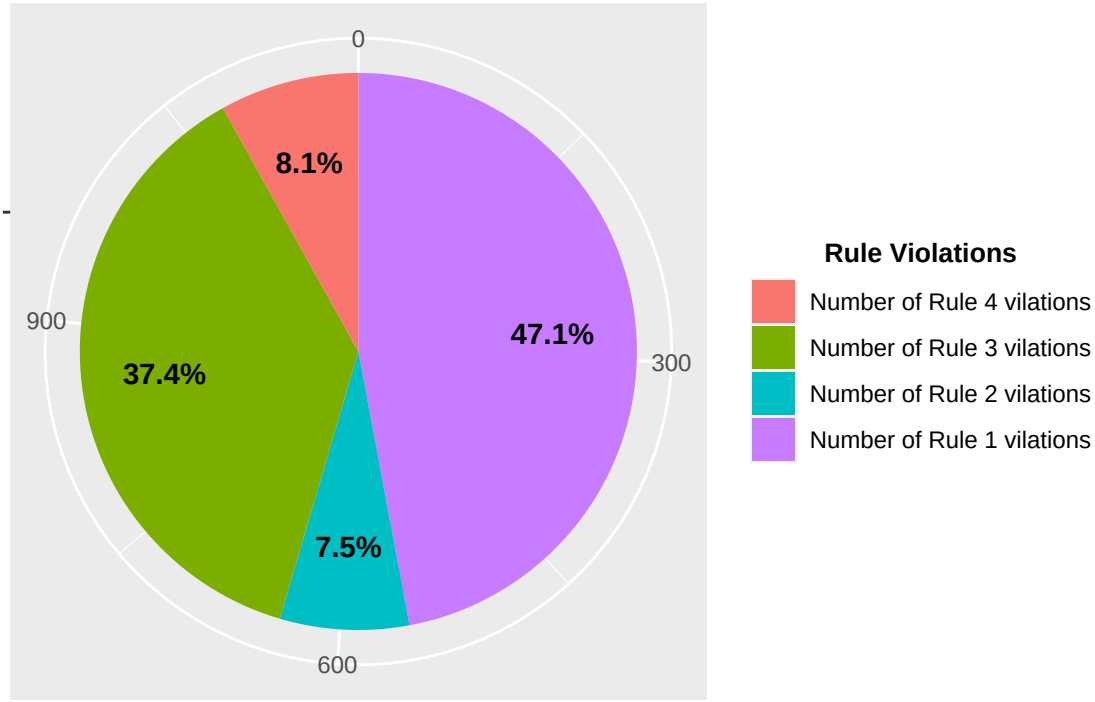

```
# adding more labels:

table_for_pie_percent <- table_for_pie %>%
  mutate(lbls = factor(lbls,
                       levels = lbls[length(lbls):1]),
         cumulative = cumsum(slices),
         midpoint = cumulative - slices / 2,
         labels = paste0(round((slices/ sum(slices)) * 100, 1), "%"))

table_for_pie_percent
```

```
##                         lbls slices cumulative midpoint labels
## 1 Number of Rule 1 vilations    553        553    276.5  47.1%
## 2 Number of Rule 2 vilations     88        641    597.0   7.5%
## 3 Number of Rule 3 vilations    439       1080    860.5  37.4%
## 4 Number of Rule 4 vilations     95       1175   1127.5   8.1%
```

```
# ggplot2 Pie Chart with percentage labels

ggplot(table_for_pie_percent, aes(x = "", y = slices, fill = lbls)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar(theta = "y", start = 0) +
  labs(x = "", y = "", title = "Percent of each Rule Violation Occurances: \n",
       fill = "Rule Violations") +
  geom_text(aes(x = 1.2, y = midpoint , label = labels), color="black",
```

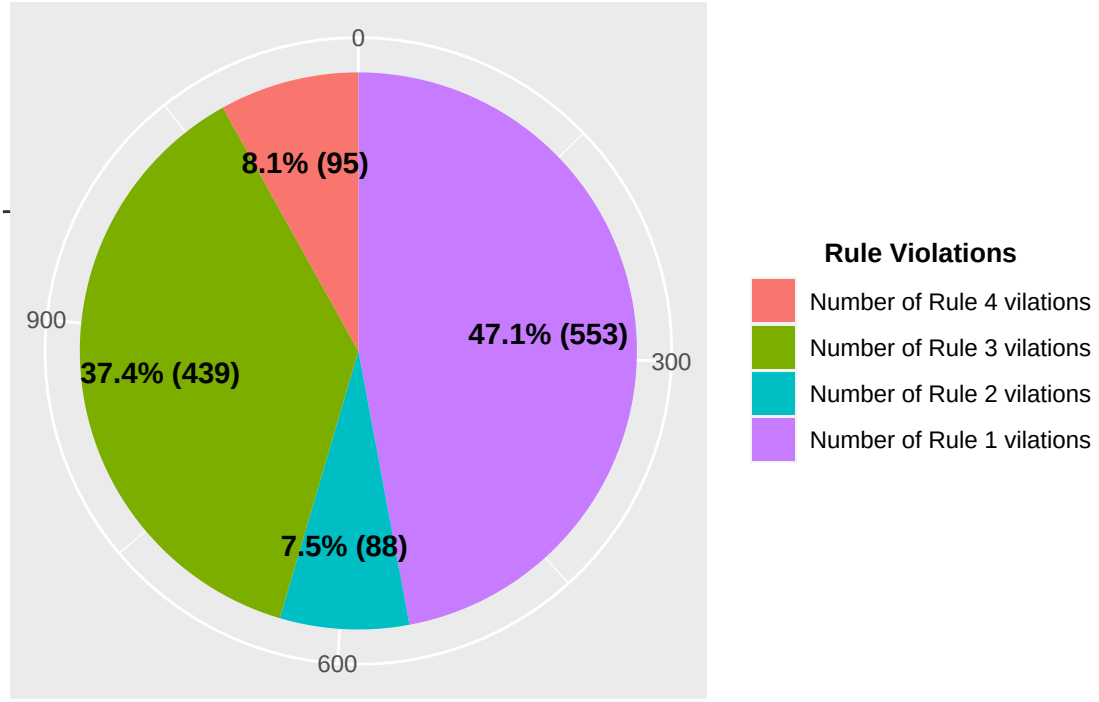
```
            fontface = "bold") +
  theme(plot.title = element_text(hjust = 0.5),
        legend.title = element_text(hjust = 0.5, face="bold", size = 10))
```

Percent of each Rule Violation Occurances:




```
# adding counts to the pie chart:

table_for_pie_labels <- table_for_pie_percent %>%
  mutate(lbls = factor(lbls, levels = lbls[length(lbls):1]),
         cumulative = cumsum(slices),
         midpoint = cumulative - slices / 2,
         labels = paste0(round((slices/ sum(slices)) * 100, 1), "%", " (", slices, ") "))

# pie chart with counts of each violation:

ggplot(table_for_pie_labels, aes(x = "", y = slices, fill = lbls)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar(theta = "y", start = 0) +
  labs(x = "", y = "", title = "Number of each Rule Violation Occurances: \n",
       fill = "Rule Violations") +
  geom_text(aes(x = 1.2, y = midpoint , label = labels), color="black",
            fontface = "bold") +
  theme(plot.title = element_text(hjust = 0.5),
        legend.title = element_text(hjust = 0.5, face="bold", size = 10))
```

Number of each Rule Violation Occurances:




```
# write to a CSV file the table_in_work that includes evaluation columns:

write.csv(table_in_work, "C:/Users/Uebi Nubov/Desktop/Hive Table Analysis/Hive Data Audit Prompt/Output/
```

While the pie charts are fun, in this particular case they might be a bit misleading.

If one wanted to find out how many files were labeled incorrectly, he or she could be tempted to simply sum up the counts from the pie charts above. Yet, this would be misleading since as we have seen throughout the analysis, a single row might have more than 1 type of violations.

Therefore, next I create a simple summary of how many rows are labeled entirely correctly vs how many rows have at lest 1 mistake:

```
# Create a new column called violation_present, which will take a value of TRUE if column violations_su

table_in_work$violation_present <- ifelse(table_in_work$violations_sum > 0, TRUE, FALSE)

summary(table_in_work$violation_present)
```

```
##    Mode   FALSE    TRUE
## logical    4379     621
```

```
# From this table we can see that 4379 rows (files) were labeled entirely correctly, while 621 rows wer

# the following code outputs 2 CSV files - one that contains all rows that were labeled correctly, and

mislabeled_rows <- table_in_work[table_in_work$violation_present,]
# Show mislabeled_rows:
mislabeled_rows
```

```
## # A tibble: 621 x 13
##    file      object_id tabular semantic definition_list header_row header_column
##    <chr>         <dbl> <lgl>   <lgl>    <lgl>           <lgl>      <lgl>
##  1 1650490-9     18975 FALSE   FALSE    TRUE            FALSE      FALSE
##  2 1604753-~      4708 FALSE   FALSE    TRUE            FALSE      FALSE
##  3 1605331-~     39514 FALSE   FALSE    TRUE            FALSE      FALSE
##  4 1606581-~      2918 FALSE   FALSE    TRUE            FALSE      FALSE
##  5 1616665-1       555 FALSE   TRUE     TRUE            FALSE      FALSE
##  6 1649097-~      5254 FALSE   FALSE    TRUE            FALSE      FALSE
##  7 1637773-3       988 TRUE    FALSE    TRUE            FALSE      FALSE
##  8 1648589-0       163 FALSE   TRUE     FALSE           FALSE      FALSE
##  9 1636115-3      9913 FALSE   TRUE     TRUE            FALSE      FALSE
## 10 1637232-2      4035 TRUE    FALSE    FALSE           TRUE       FALSE
## # ... with 611 more rows, and 6 more variables: rule_1_violation <dbl>,
## #   rule_2_violation <dbl>, rule_3_violation <dbl>, rule_4_violation <dbl>,
## #   violations_sum <dbl>, violation_present <lgl>
```

```
# write results to file:
write.csv(mislabeled_rows, "C:/Users/Uebi Nubov/Desktop/Hive Table Analysis/Hive Data Audit Prompt/Outpu

correctly_labeled_rows <- table_in_work[!table_in_work$violation_present,]
# Show correctly_labeled_rows:
correctly_labeled_rows
```

```
## # A tibble: 4,379 x 13
##    file      object_id tabular semantic definition_list header_row header_column
##    <chr>         <dbl> <lgl>   <lgl>    <lgl>           <lgl>      <lgl>
##  1 1650255-9     21650 TRUE    TRUE     FALSE           TRUE       TRUE
##  2 1650594-3       813 TRUE    TRUE     FALSE           TRUE       FALSE
##  3 1649219-5      2639 TRUE    TRUE     FALSE           TRUE       FALSE
##  4 1650596-1       362 TRUE    TRUE     TRUE            FALSE      FALSE
##  5 1650643-1       326 TRUE    TRUE     FALSE           TRUE       FALSE
##  6 1650654-~     67869 TRUE    TRUE     FALSE           TRUE       FALSE
##  7 1650626-~     16079 TRUE    TRUE     TRUE            FALSE      TRUE
##  8 1616723-5      1466 TRUE    TRUE     TRUE            FALSE      FALSE
##  9 1616773-4      1482 TRUE    TRUE     TRUE            TRUE       FALSE
## 10 1616723-3       751 TRUE    TRUE     TRUE            FALSE      TRUE
## # ... with 4,369 more rows, and 6 more variables: rule_1_violation <dbl>,
## #   rule_2_violation <dbl>, rule_3_violation <dbl>, rule_4_violation <dbl>,
## #   violations_sum <dbl>, violation_present <lgl>
```

```
# write results to file:
write.csv(correctly_labeled_rows, "C:/Users/Uebi Nubov/Desktop/Hive Table Analysis/Hive Data Audit Prom
```

My recommendation is to either try and create rules for fixing minor mislabeling mistakes (less than 2 violations), and try to correct for those automatically, or to simply say that even a single violation is a mistake and return the mislabeled rows for review, while keeping the correctly labeled rows.

This concludes my analysis, thank you for your time! :)