DATA 607: Assignment Two

Zachary Safir

02/10/2020

Introduction

For this assignment we were tasked to choose six recent movies or shows. Then to find at least six other people to rate each of the chosen works. I decided to use a data set collected by one of the students in the DATA 607 class. It is an excellent example of raw data collection that requires some data cleaning to be done before it can be used in a SQL database. The following sections will demonstrate the process for that cleaning. From there, the data will be loaded into SQL where I will demonstrate some queries using the loaded data. Then, I will demonstrate how to handle any missing values that are in the data.

Loading the Data

```
library(DBI)
library(RPostgres)
library(readxl)
library(tidyverse)
library(knitr)
library(kableExtra)
Responses <-read_excel("~/CUNY SPS/607/Data/Data 607 - Viewer Reviews (Responses).xlsx")
```

Deafult Columnn Names

Seen below are the default column names of the dataset. In its current state, it is in no shape to be used in a SQL database.

```
kable(colnames(Responses),format = 'latex',col.names = 'Column Names') %>%
    kable_styling(full_width = TRUE)
```

Column Names
Timestamp
Top 10 Most Watched Netflix Shows in 2020 [The Queens Gambit]
Top 10 Most Watched Netflix Shows in 2020 [Emily in Paris]
Top 10 Most Watched Netflix Shows in 2020 [Lucifer]
Top 10 Most Watched Netflix Shows in 2020 [The Umbrella Academy]
Top 10 Most Watched Netflix Shows in 2020 [Money Heist]
Top 10 Most Watched Netflix Shows in 2020 [Dark Desire]
Top 10 Most Watched Netflix Shows in 2020 [Friends]
Top 10 Most Watched Netflix Shows in 2020 [The Crown]
Top 10 Most Watched Netflix Shows in 2020 [Ratched]
Top 10 Most Watched Netflix Shows in 2020 [Dark]
Which TV and/or movie genres do you enjoy watching most?
Which TV and/or movie genres do you enjoy watching least?
On average, how many hours a week do you spend on Netflix each week?
What movie or TV show on Netflix or other streaming services would you highly recommend to adults that wasn't on this list?

Fixing the Columns

The first step in correcting this data will be fixing the columns. Specifically, adding an ID column to each respondent and renaming the current columns. The ID column is extremely important information to help distinguish each observation. It will become even more important in the next section when we reshape the data.

As for renaming, the current columns contain unnecessarily long names with a lot of repeated information. It is also important to note this, in the case of long titles such as we have here with the show names, often times it is a much more prudent choice to replace the names with numeric IDs in order to reduce the complexity of the data stored which will increase the speed at which you can query the data. For this example, due how small the data set is, I will keep the names as they are for simplicity.

```
Responses <- Responses %>%
    mutate(id = row_number()) %>%
    relocate(id) %>%
    rename(
    timestamp = Timestamp,
    `The Queens Gambit` =
    `Top 10 Most Watched Netflix Shows in 2020 [The Queens Gambit]`, `Emily in Paris` =
    `Top 10 Most Watched Netflix Shows in 2020 [Emily in Paris]`, `Lucifer` =
    `Top 10 Most Watched Netflix Shows in 2020 [Lucifer]`, `The Umbrella Academy` =
    `Top 10 Most Watched Netflix Shows in 2020 [The Umbrella Academy]`, `Money Heist` =
    `Top 10 Most Watched Netflix Shows in 2020 [Money Heist]`, `Dark Desire` =
    `Top 10 Most Watched Netflix Shows in 2020 [Dark Desire]`,`Friends` =
    `Top 10 Most Watched Netflix Shows in 2020 [Friends]`,`The Crown` =
    `Top 10 Most Watched Netflix Shows in 2020 [The Crown]`,`Ratched` =
    `Top 10 Most Watched Netflix Shows in 2020 [Ratched]`,`Dark` =
    `Top 10 Most Watched Netflix Shows in 2020 [Dark]`,`fav_genre` =
    `Which TV and/or movie genres do you enjoy watching most?`,
    `least_fav_genre` = `Which TV and/or movie genres do you enjoy watching least?`,
    `avg_time_w` = `On average, how many hours a week do you spend on Netflix each week?`,
    `What movie or TV show on Netflix or other streaming services would you highly recommend to adults
```

The output below displays the new column names. We have now made the data far easier to read and use.

```
kable(colnames(Responses),format = 'latex',col.names = 'Column Names') %>%
    kable_styling(full_width = F)
```

Column Names
id
timestamp
The Queens Gambit
Emily in Paris
Lucifer
The Umbrella Academy
Money Heist
Dark Desire
Friends
The Crown
Ratched
Dark
fav_genre
least_fav_genre
avg_time_w
recommend

Subsetting and Reshaping the Data

The raw data had each show displayed in separate columns. While this may make it easy to view for a person, it makes it very challenging to query from. The data shape we need is one where all the shows reside in a singular column. This is where the id column becomes especially useful for keeping track of which response belongs to who.

However, we cannot simply reshape the data as it is. Given that the data set contains more specific questions, such as what the respondents favorite genre is, if we were to include that information into the reshaped data, it would get repeated over and over and become unnecessary bloat. Fixing this is simple, we need to create separate data tables where we can look up this information.

In addition, in order to properly quantity the respondents responses, we need to create a numeric column that represents each response. Doing so with dpylr's mutate and case_when functions make this very easy. We can change each response to the appropriate number, and make the non responses NA values.

```
Date <- Responses %>%
        select(id,timestamp)

Recommendations <- Responses %>%
                    select(id,fav_genre,least_fav_genre,avg_time_w,recommend)

ShowResponses <- Responses %>%
                    select(id,`The Queens Gambit`,`Emily in Paris`, Lucifer,
                           `The Umbrella Academy`,`Money Heist`,`Dark Desire`
                           ,Friends,`The Crown`,Ratched,Dark) %>%
                    gather(key = 'show',value = 'response',c(`The Queens Gambit`,
                                                             `Emily in Paris`, Lucifer,
                                                             `The Umbrella Academy`,
                                                             `Money Heist`,`Dark Desire`,
                                                             Friends,`The Crown`,Ratched,
                                                             Dark)) %>%
                   mutate(score = case_when(
                        response == "Poor" ~  1,
                        response ==   "Fair" ~  2,
                        response ==  "Average" ~  3,
                        response ==   "Good" ~  4,
                        response ==   "Excellent" ~  5,
                          TRUE ~ NA_real_))
```

Cleaned Data Tables

Shown below is a look at the three newly created tables.

```
kable(head(Date),format = 'latex') %>%
    kable_styling(full_width = F)
```

id	timestamp
1	2021-02-08 15:47:13
2	2021-02-08 20:26:56
3	2021-02-08 21:07:26
4	2021-02-08 21:11:43
5	2021-02-08 21:13:00
6	2021-02-09 08:16:13

```
kable(head(Recommendations),format = 'latex') %>%
    kable_styling(full_width = T)
```

id	fav_genre	least_fav_genre	avg_time_w	recommend
1	Comedy, Drama, Action and Adventure	Reality	20	The Great Pretender
2	Drama	Reality, Horror & Sci-fi	2	Fleabag, on Amazon
3	Comedy, Action and Adventure, Horror and Sci-Fi	Drama, Children & Family, Reality	1	Death To 2020
4	Drama, Action and Adventure, Horror and Sci-Fi	Comedy, Documentary, Reality	8	the 100
5	Comedy, Action and Adventure, Documentary	Reality, Horror & Sci-fi	5	NA
6	Comedy, Documentary	Action and Adventure, Children & Family, Reality, Horror & Sci-fi	0	Wondershowzen

```
kable(head(ShowResponses),format = 'latex' )%>%
    kable_styling(full_width = F)
```

id	show	response	score
1	The Queens Gambit	Excellent	5
2	The Queens Gambit	Excellent	5
3	The Queens Gambit	Excellent	5
4	The Queens Gambit	Good	4
5	The Queens Gambit	No opinion - I haven't seen it	NA
6	The Queens Gambit	Poor	1

Creating an SQL Connection and Loading the Data

Now that the data is in proper format, we will load it into a SQL database. The process for making the database itself is fairly simple once you have your chosen SQL software installed. In this assignment PostgreSQL is used. This link explains the process for creating a database within the software https://www.enterprisedb.com/postgres-tutorials/how-create-postgresql-database-and-users-using-psql-and-pgadmin.

Connecting to the database only requires one line of code. For this assignment, I used a generic password for the connection. In a proper database, we would protect the password by putting it in a file that we could then use the path to its location for verification.

```
con <- dbConnect(RPostgres::Postgres(), user='postgres', password='zach', dbname='Zach')
```

The code below demonstrates how easy it is to send data loaded into R to your SQL database. The only issue I ran into was using capital letters for the table names. SQL puts everything to lowercase by default, by using capitals in the names it throws SQL for a loop. The tables created were empty and could not be deleted using the default drop table command. A similar issue occurred with any column name that had a capital in it too. It will act as if those columns do not exist. For that reason, I made all columns and table names lowercase.

By default, the command dbWriteTable will number each row, but this is easy to change by setting row.names to false. Setting overwrite to true overwrites any existing table with the same name. I used it here since I knit this file many times over as I edit, and it was throwing an error that the tables already existed in the database.

```
dbWriteTable(con,'date',Date,row.names=FALSE,overwrite=TRUE)
dbWriteTable(con,'recommendations',Recommendations,row.names=FALSE,overwrite=TRUE)
dbWriteTable(con,'show_responses',ShowResponses,row.names=FALSE,overwrite=TRUE)
```

Using SQL within R

The output below is generated using an SQL chunk directly within R. As we can see, all three tables were successfully loaded into my database. The second output verifies that the data was loaded in without any issues. From there, we can use any SQL command to query the data.

```
SELECT table_name
FROM information_schema.tables
WHERE table_schema = 'public'
ORDER BY table_name;
```

table_name
date
recommendations
show_responses

```
SELECT * FROM show_responses;
```

id	show	response	score
1	The Queens Gambit	Excellent	5
2	The Queens Gambit	Excellent	5
3	The Queens Gambit	Excellent	5
4	The Queens Gambit	Good	4
5	The Queens Gambit	No opinion - I haven't seen it	NA
6	The Queens Gambit	Poor	1
7	The Queens Gambit	No opinion - I haven't seen it	NA
8	The Queens Gambit	No opinion - I haven't seen it	NA
9	The Queens Gambit	No opinion - I haven't seen it	NA
10	The Queens Gambit	No opinion - I haven't seen it	NA

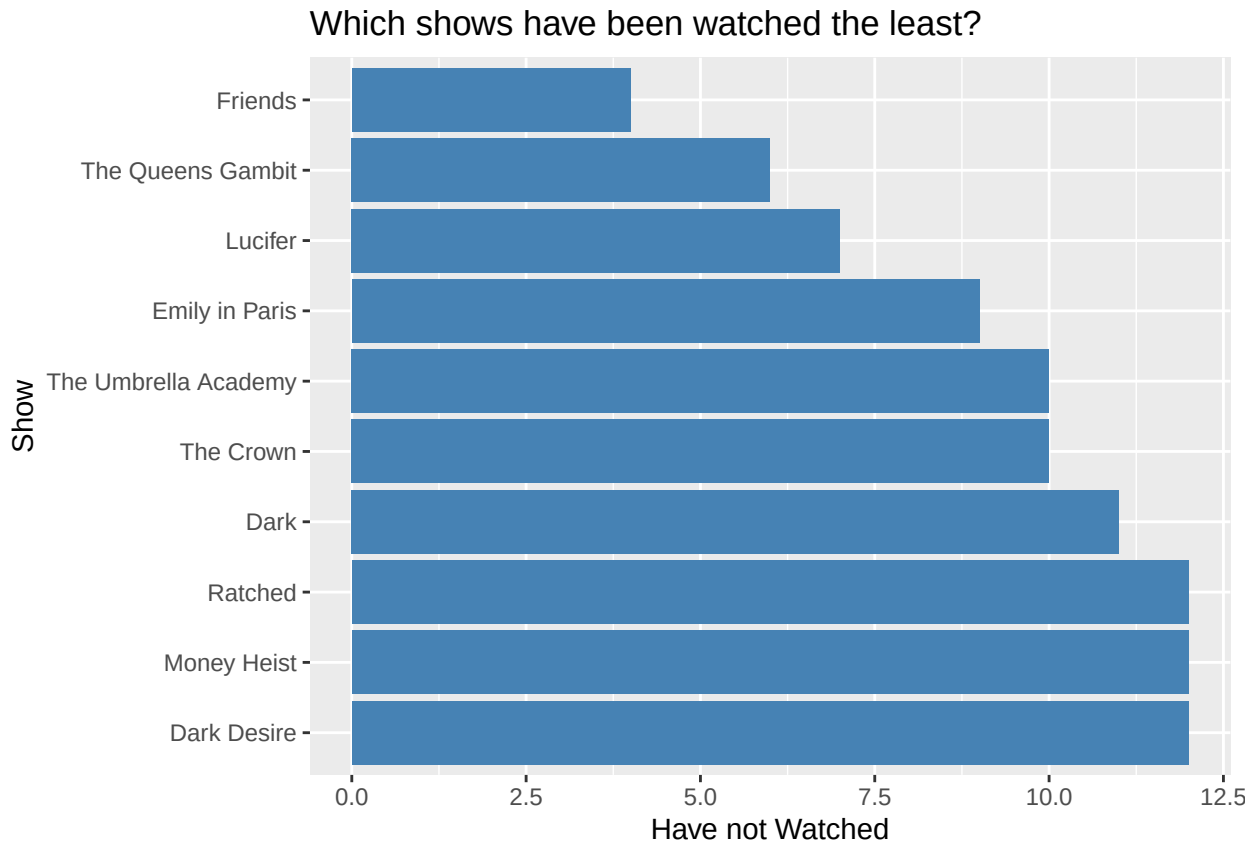
```
SELECT show
FROM show_responses
WHERE score = 5;
```

show
The Queens Gambit
The Queens Gambit
The Queens Gambit
The Queens Gambit
Lucifer
Friends
Friends
The Crown
Dark

Analysis and Missing Data Handling

The missing data is valuable information. In the case of this data, it will let us know which shows are not being watched by our 607 classmates. Trying to input values for the missing data would create false results. We do not have enough information here to make inferences about each show. Instead, we will graph the missing values below and discover the most watched show, and the least watched show.

```
ShowResponses %>%
  group_by(show) %>%
  tally((is.na(score))) %>%
  ggplot(aes(x=reorder(show,-n),y=n)) +
  geom_bar(stat = 'identity',fill="steelblue") +
  coord_flip() +
  labs(title = 'Which shows have been watched the least?', y = 'Have not Watched',
       x= "Show")
```



As we can see from the graphic above, Friends is the most watched and the shows Ratched, Money Heist, and Dark Desire are the least watched shows within the responses. If we had more data available, we could possibly make inferences as to why those shows are not being watched as much. This is an important note to make. The missing values do have important uses, and blindly disregarding or transforming them can skew any analysis done.

Conclusion

This assignment had us collect survey results and load that information into a SQL database. As my work on this assignment showed, raw data collected in the wild can be seriously messy. Proper data handling is an important skill to have, otherwise you will not be able to use the data you have taken the time to collect. Whenever you are working with a new dataset, it is extremely important that you ensure the format is correct and that there is nothing that will create problems later on. In my case, I learned the hard way that SQL does not like capital letters. I had go over and correct all my variable names to make sure they were all lowercase. It is also, once again, very important that you do not blindly remove or replace missing data. There are so many insights you can gain from using it properly. Hopefully this assignment has demonstrated all of this clearly. Thank you very much for taking the time to read through my assignment.