DATA607- Assignment 2

Juan Falck

Introduction

The task ahead is to create a database in SQL which would capture the ratings of different movies by a number of people.

For this assignment I separated the work in two big chunks of work

- The general setup to caputre friend's movie ratings in an easy intuitive manner
- Be able to read from R SQL data to process it, visualize it etc.

General Setup

I decided to create a Linux server in AWS (EC2 instance) and install in it an Apache web server, PHP and MySql.

The general objective is to run in the cloud a webserver which would provide a form where people could use to input their ratings of selected movies.

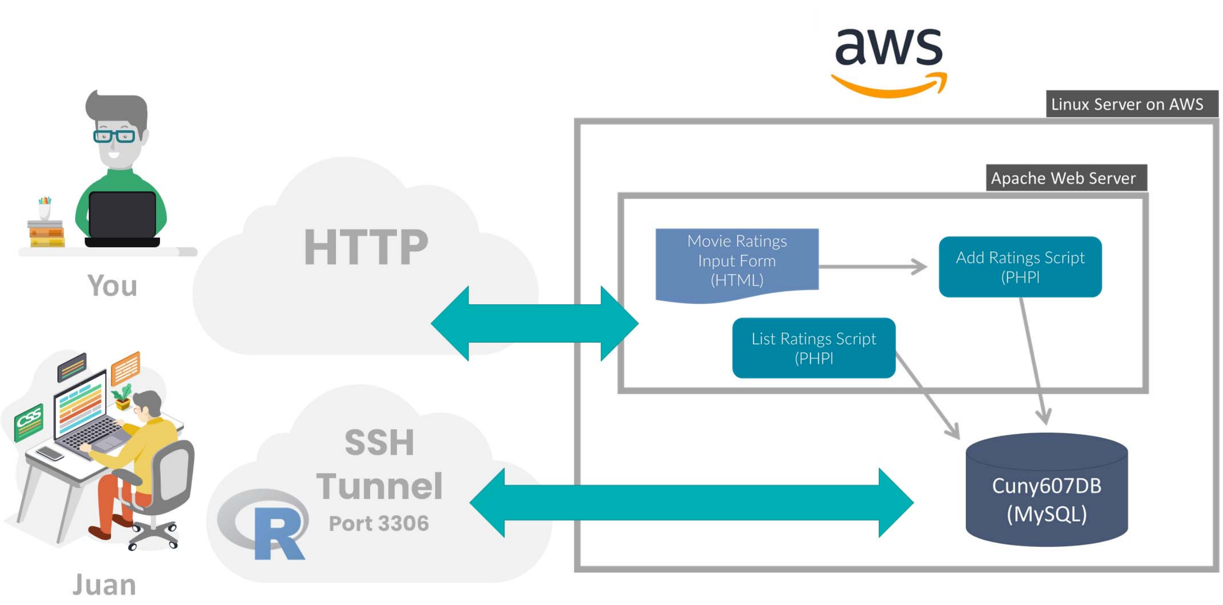
The form will interact through php and insert the responses back into MySQL for further retrieval from within R.

Create Linux server and install MysQL, PHP and Apache

The below diagram shows the general setup of how the whole thing looks: Apache, PHP and MySQL running on a Linux server.

I will from my side setup an SSH Tunnel through port 3306 using my private key in order to run R scripts from my computer at home. For reproducibility I will provide a script with includes a small data set that you can run to run the R code below (see later section on Reproducibility)

Below is a general diagram of the overall structure.

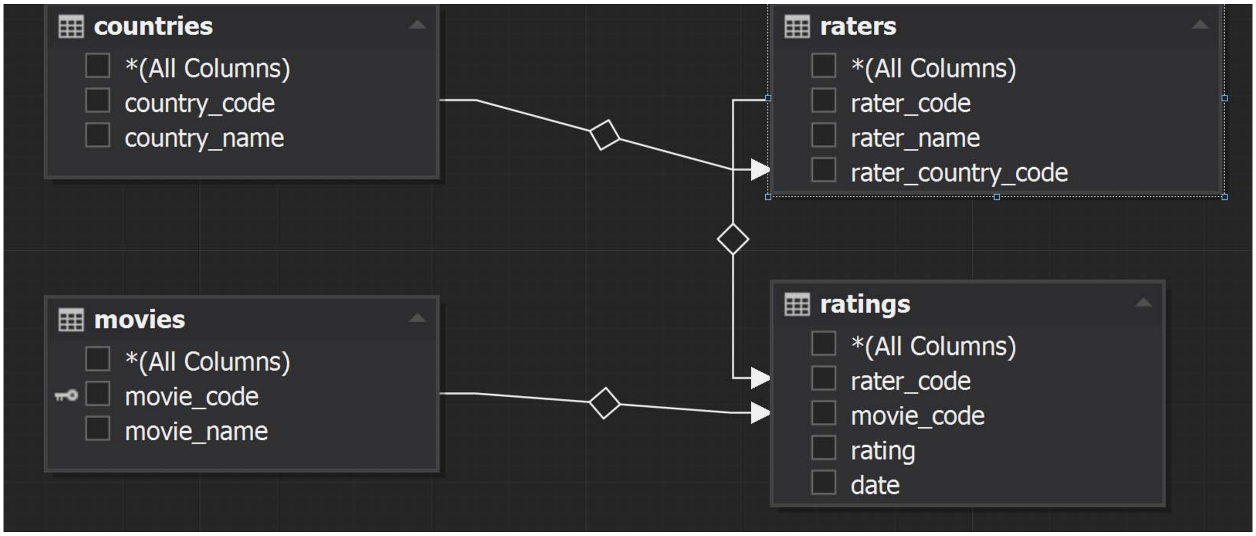


MySQL Database

The general design was to keep 3 reference tables that would be used for each rating submitted. The tables were:

- **movies**, which would deal with the names of the movies to be rated.
- **raters**, which would deal with the names of the people rating moviers.
- **countries**, which would add another dimension to analysis where from each rater we would also capture their country of origin. This would allow to perform things like "average rating of movies by country".
- **ratings**, which would record each movie rated by each person on a scale 1 to 5 from best to worse.

Below is a diagram with the tables and columns used. The "code" field used for raters, is a randomly generate 3-character codes that will be generated automatically by the PHP submission form.



countries
□ *(All Columns)
□ country_code
□ country_name

raters
□ *(All Columns)
□ rater_code
□ rater_name
□ rater_country_code

movies
□ *(All Columns)
□ movie_code
□ movie_name

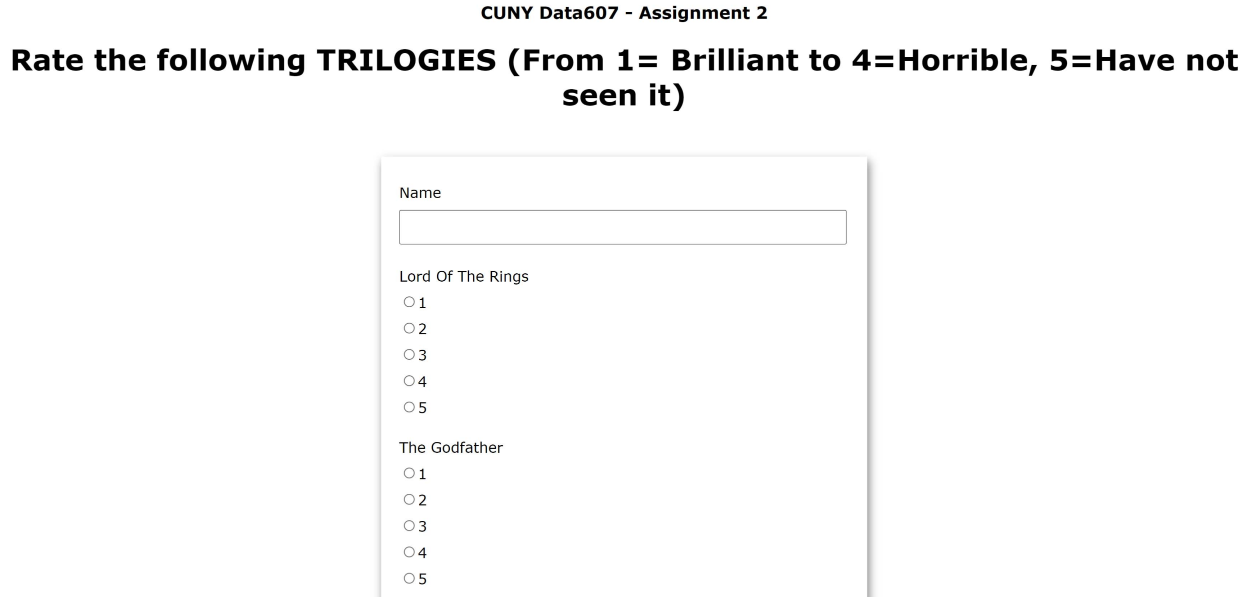
ratings
□ *(All Columns)
□ rater_code
□ movie_code
□ rating
□ date

PHP Form to ask for movie ratings

This is the form that people will use to submit their ratings. you can try it now and submit your own rating. **Please try it!**

http://35.174.11.125/data607c.html

Screenshot of the submission form below.



CUNY Data607 - Assignment 2

Rate the following TRILOGIES (From 1= Brilliant to 4=Horrible, 5=Have not seen it)

Name

Lord Of The Rings

○1
○2
○3
○4
○5

The Godfather

○1
○2
○3
○4
○5

Some considerations written in the PHP code

- Each person submitting a rating has a 3-character code assigned. This done by the PHP script randomizing characters and providing one to each rater.
- For this exercise we hard-coded "US" as country. In real case scenario we would use the person's IP address to determine country.
- The PHP scripts generates the time when the rating was submitted.

Connectivity

In order to be able to run R scripts from my home computer I will need to establish an SSH connection to the create Linux server in AWS. This Linux server requires a private key for authentication, so establishing a tunnel into the Linux server would not be reproducible.

For reproducibility

Since tunneling and providing the private key for the Linux server is not part of the design, **I am providing a full SQL script with sample data that you can run in your own machine and the run all the R scripts provided below.**

Before running script please make sure you create a database im MySQL callned "cuny607". After that you can run the script below.

https://github.com/jrfalck/CUNY-Data607/blob/main/Assignment_2/cuny607_202202121543.sql

R scripts for data access and manipulation

Required packages

We will use RMariaDB for connectivity from R to the remote AWS Linux server's MySQL. tidyverse for general data wrangling. reshape2 will be used for a few data transformations.

```
library(RMariaDB)
library(tidyverse)
library(reshape2)
```

Connect R to the remote MySQL server

If you are trying to reproduce results, set here you db password and change host to "host_local". Also set here the address of your MySQL server.

```
mypwd <- "data607password"
host_local = '192.168.1.151'
host_aws_tunnel = '127.0.0.1'

ratingsDB <- dbConnect(RMariaDB::MariaDB(),
                   user= 'data607usr',
                   password= mypwd,
                   dbname= 'cuny607',
                   host=host_aws_tunnel,
                   port = 3306)
```

Run the following command to verify the connection is ok. You should see a list of tables.

```
dbListTables(ratingsDB)
```

```
## [1] "basic_ratings" "countries"     "movies"        "raters"
## [5] "ratings"       "ref_movies"
```

Our first query from R to MySQL

Let's run a simple query of the ratings submitted. We will list only the first 6 records.

```
query<-'select * from ratings'
result <- dbSendQuery(ratingsDB,query)
ratingsdf <- dbFetch(result,6)
ratingsdf
```

```
##   rater_code movie_code rating       date
## 1        001        001      1 2022-02-10
## 2        001        002      1 2022-02-10
## 3        001        003      2 2022-02-10
## 4        001        004      1 2022-02-10
## 5        001        005      1 2022-02-10
## 6        001        006      5 2022-02-10
```

Display a formatted list of all Movie Ratings

This query makes use of joins for the display all ratings, including their name and country of origin.

```
query<-'SELECT
  raters.rater_name,
  countries.country_name,
  movies.movie_name,
  ratings.rating,
  ratings.date
FROM movies
  INNER JOIN ratings
    ON movies.movie_code = ratings.movie_code
  CROSS JOIN countries
  INNER JOIN raters
    ON countries.country_code = raters.rater_country_code
    AND raters.rater_code = ratings.rater_code'

result <- dbSendQuery(ratingsDB,query)
ratingsdf2 <- dbFetch(result)
dbClearResult(result)
dbDisconnect(ratingsDB)
ratingsdf2
```

```
##           rater_name   country_name         movie_name rating       date
## 1         Juan Falck      Singapore  Lord Of The Rings      1 2022-02-10
## 2         Juan Falck      Singapore      The Godfather      1 2022-02-10
## 3         Juan Falck      Singapore          Star Wars      2 2022-02-10
## 4         Juan Falck      Singapore Back To The Future      1 2022-02-10
## 5         Juan Falck      Singapore         The Matrix      1 2022-02-10
## 6         Juan Falck      Singapore       Mighty Ducks      5 2022-02-10
## 7       Donald Trump  United States  Lord Of The Rings      1 2022-02-10
## 8       Donald Trump  United States          Star Wars      3 2022-02-10
## 9       Donald Trump  United States         The Matrix      2 2022-02-10
## 10 Elisabeth Windsor United Kingdom      The Godfather      1 2022-02-10
## 11 Elisabeth Windsor United Kingdom          Star Wars      2 2022-02-10
## 12 Elisabeth Windsor United Kingdom Back To The Future      3 2022-02-10
## 13 Elisabeth Windsor United Kingdom         The Matrix      5 2022-02-10
## 14              AMLO         Mexico         The Matrix      3 2022-02-10
## 15         Joe Biden  United States      The Godfather      2 2022-02-10
## 16         Joe Biden  United States          Star Wars      2 2022-02-10
## 17         Joe Biden  United States Back To The Future      1 2022-02-10
```

Displaying all ratings by Person

This R chunk will display all raters (people) as rows, all movies rated as columns and their respetive rating in each cell.

```
head(dcast(ratingsdf2,rater_name~movie_name, value.var='rating', sum))
```

```
##          rater_name Back To The Future Lord Of The Rings Mighty Ducks Star Wars
## 1              AMLO                  0                 0            0         0
## 2      Donald Trump                  0                 1            0         3
## 3 Elisabeth Windsor                  3                 0            0         2
## 4         Joe Biden                  1                 0            0         2
## 5        Juan Falck                  1                 1            5         2
##   The Godfather The Matrix
## 1             0          3
## 2             0          2
## 3             1          5
## 4             2          0
## 5             1          1
```

The code above works but has a flaw where the NA's are displayed as 0. We may not want this behavior.

```
df_rater_na <- dcast(ratingsdf2,rater_name~movie_name, value.var='rating',fun.aggregate = NULL)
df_rater_na
```

```
##          rater_name Back To The Future Lord Of The Rings Mighty Ducks Star Wars
## 1              AMLO                 NA                NA           NA        NA
## 2      Donald Trump                 NA                 1           NA         3
## 3 Elisabeth Windsor                  3                NA           NA         2
## 4         Joe Biden                  1                NA           NA         2
## 5        Juan Falck                  1                 1            5         2
##   The Godfather The Matrix
## 1            NA          3
## 2            NA          2
## 3             1          5
## 4             2         NA
## 5             1          1
```

The code above should show the places (movies) where the rater didn't submit a rating as NA.

Displaying average ratings by Country

This will be another view of our data, now making use of the Country feature, it will average all ratings for all movies by movie and by country.

```
dcast(ratingsdf2,country_name~movie_name, value.var='rating', mean)
```

```
##     country_name Back To The Future Lord Of The Rings Mighty Ducks Star Wars
## 1         Mexico                NaN               NaN          NaN       NaN
## 2      Singapore                  1                 1            5       2.0
## 3 United Kingdom                  3               NaN          NaN       2.0
## 4  United States                  1                 1          NaN       2.5
##   The Godfather The Matrix
## 1           NaN          3
## 2             1          1
## 3             1          5
## 4             2          2
```

Replace NA's with average rating of column

In case were tryng to replace the NA's (for example trying to figure out if person would like a movie) We will replace all NA's of a movie with the average rating of that same movie.

```
# we will make a copy of the table with NA's and use this one instead

df_rater_mean <- df_rater_na

for(i in 1:ncol(df_rater_mean)){
  df_rater_mean[is.na(df_rater_mean[,i]), i] <- mean(df_rater_mean[,i], na.rm = TRUE)
}

df_rater_mean
```

```
##          rater_name Back To The Future Lord Of The Rings Mighty Ducks Star Wars
## 1              AMLO           1.666667                 1            5      2.25
## 2      Donald Trump           1.666667                 1            5      3.00
## 3 Elisabeth Windsor           3.000000                 1            5      2.00
## 4         Joe Biden           1.000000                 1            5      2.00
## 5        Juan Falck           1.000000                 1            5      2.00
##   The Godfather The Matrix
## 1      1.333333       3.00
## 2      1.333333       2.00
## 3      1.000000       5.00
## 4      2.000000       2.75
## 5      1.000000       1.00
```

Conclusion

SQL in my opinion is a great platform to gather and organize massive amounts of data. Technically we could put this form in production and ask many many people to submit their ratings. While people submit ratings, we could use R sporadically to compile reports and statistics of the data being collected from the comfort of our home.