Data607-Week02 Assignment - Assignment - SQL and R

Vinayak Kamath

2/7/2020

Setup Instruction for the DB creation and Data insertion:

Here are the instructions for downloading the ‘’movies” database. ‘This assumes that you have already installed MySQL, including MySQL workbench’.

1. Download the attached movies.sql [1] file from the GIT repository and copy the files to your local machine.
2. Launch MySQL workbench and Create a new MySQL schema (database) called ‘’movies”
3. Make the ‘’movies” database your default schema and Open the script movies.sql; please modify the LOAD command in the script to point to the correct file location.
4. Verify that records were successfully loaded.

Instruction for the CSV file generation:

Here are the instructions for exporting .CSV file that contains the data for movies[movies.csv], users[users.csv], ratings[ratings.csv] and upcoming_movies[upcoming_movies.csv]. ‘This assumes that you have already installed MySQL, including MySQL workbench’.

1. Download the attached GenerateMoviesRatingsCSV.sql [2] file from the GIT repository and copy the files to your local machine.
2. Launch MySQL workbench and Make the ‘’movies” database your default schema and Open the script GenerateMoviesRatingsCSV.sql; please modify the OUTFILE command in the script to point to the correct file location.
3. Verify that the four CSV files were successfully generated.
4. If the files are not generated locally then pls use the mentioned link to the GIT repository to get the already generated four CSV files. [3]

[1] Git Repository link for movies.sql file: https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/movies.sql
[2] Git Repository link for GenerateMoviesRatingsCSV.sql file: https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/GenerateMoviesRatingsCSV.sql
[3] GIT Repository link for .CSV file that contains the data for movies[movies.csv], users[users.csv], ratings[ratings.csv] and upcoming_movies[upcoming_movies.csv] are https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/movies.csv ; https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/users.csv ; https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/ratings.csv and https://raw.githubusercontent.com/kamathvk1982/Data607-Week02/master/upcoming_movies.csv

---

Loading the Information from .CSV to R Data Frame:

```r
# First, set your working directory to fodler where the RMD and .CSV files are kept; Please modify the f
setwd("~/Vinayak/CUNY/Data607/Assignment/Week02/")

# Import the data files into individual Datfrae and get summary of the Data Frames:
movies.df <- read.csv(file = 'movies.csv', header = F , sep = ',' )
users.df <- read.csv(file = 'users.csv', header = F , sep = ',' )
ratings.df <- read.csv(file = 'ratings.csv', header = F , sep = ',' )
upcoming.movies.df <- read.csv(file = 'upcoming_movies.csv', header = F , sep = ',' )

#Naming the Columns:
names(movies.df) <- c('Id','MovieName','YearOfRelease','Genre')
names(users.df) <- c('Id','UserName','Age')
names(ratings.df) <- c('MovieId','MovieName','UserId','UserName','Ratings')
names(upcoming.movies.df) <- c('Id','MovieName','YearOfRelease','Genre', 'AutoRank')


str(movies.df)
```

```
## 'data.frame':    6 obs. of  4 variables:
##  $ Id           : int  1 2 3 4 5 6
##  $ MovieName    : Factor w/ 6 levels "Avengers: Endgame",..: 2 6 3 4 1 5
##  $ YearOfRelease: int  2020 2019 2019 2019 2019 2019
##  $ Genre        : Factor w/ 3 levels "Action","Comedy",..: 1 3 2 2 1 2
```

```r
str(users.df)
```

```
## 'data.frame':    7 obs. of  3 variables:
##  $ Id      : int  1 2 3 4 5 6 7
##  $ UserName: Factor w/ 7 levels "ABC","LMN","POP",..: 1 5 6 7 3 4 2
##  $ Age     : int  35 38 21 25 45 45 34
```

```r
str(ratings.df)
```

```
## 'data.frame':    42 obs. of  5 variables:
##  $ MovieId  : int  1 1 1 1 1 1 1 2 2 2 ...
##  $ MovieName: Factor w/ 6 levels "Avengers: Endgame",..: 2 2 2 2 2 2 2 6 6 6 ...
##  $ UserId   : int  1 2 3 4 5 6 7 1 2 3 ...
##  $ UserName : Factor w/ 7 levels "ABC","LMN","POP",..: 1 5 6 7 3 4 2 1 5 6 ...
##  $ Ratings  : int  5 4 4 5 3 NA 5 4 4 NA ...
```

```r
str(upcoming.movies.df)
```

```
## 'data.frame':    10 obs. of  5 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10
##  $ MovieName    : Factor w/ 10 levels "Black Widow",..: 7 4 5 8 3 6 1 2 10 9
##  $ YearOfRelease: int  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
##  $ Genre        : Factor w/ 3 levels "Action","Comedy",..: 2 2 2 1 3 1 1 2 1 2
##  $ AutoRank     : int  3 4 5 5 3 3 4 2 4 2
```

Analysis on the Loaded DataSet:

1. Movie With the highest average Rating with ignoring "N/A" ratings among all Users: *Observation*: "Parasite" is on the Top with two users viewing and rating it at 5 each

```r
ratings.df %>%
  filter(Ratings !=  "")  %>%
  group_by(MovieId, MovieName) %>%
  summarize(mean_rank = mean(Ratings)) %>%
  arrange(desc(mean_rank))
```

```
## # A tibble: 6 x 3
## # Groups:   MovieId [6]
##   MovieId MovieName                      mean_rank
##     <int> <fct>                              <dbl>
## 1       6 Parasite                            5
## 2       4 Once Upon a Time in Hollywood       4.43
## 3       1 Birds of Prey                       4.33
## 4       2 The Irishman                        4.33
## 5       5 Avengers: Endgame                   4.33
## 6       3 Knives Out                          4.14
```

2. Movie With the highest average Rating with treating "N/A" rating as default 2 [Missing Data Handling] ratings among all Users : *Observation*: "Once Upon a Time in Hollywood" leads as the movie was seen by all users and had ratings of 4 and 5 only. "Parasite" landed on the bottom as not many have seen it. Using a Default rating helped us to standardize the average ratings for the movies.

```r
ratings.df %>%
  replace(is.na(.), 1)%>%
  group_by(MovieId, MovieName) %>%
  summarize(mean_rank = mean(Ratings, na.rm= TRUE)) %>%
  arrange(desc(mean_rank))
```

```
## # A tibble: 6 x 3
## # Groups:   MovieId [6]
##   MovieId MovieName                      mean_rank
##     <int> <fct>                              <dbl>
## 1       4 Once Upon a Time in Hollywood       4.43
## 2       3 Knives Out                          4.14
## 3       1 Birds of Prey                       3.86
## 4       5 Avengers: Endgame                   3.86
## 5       2 The Irishman                        2.43
## 6       6 Parasite                            2.14
```

3. Genre based Recommendations for each User: *Observation*: For each user we can take the top ranked movies by them and by taking the top movie Genre we can recommend to them similar rating Genre movies having standardized high rating rounded to the nearer larger integer.

```r
#Step 1: Get the Standardized Rating Data Frame
standardized.ratings.df <- ratings.df %>%
  replace(is.na(.), 1)%>%
  group_by(MovieId, MovieName) %>%
  summarize(mean_rank = ceiling (mean(Ratings, na.rm= TRUE))) %>%
  arrange(desc(mean_rank))

#Step 2: Join with Movies data frame to get the year and genre into the Standardized Rating Data Frame
standardized.ratings.df <- left_join(standardized.ratings.df, movies.df, by = c("MovieId" = "Id", "Movi
standardized.ratings.df
```

```
## # A tibble: 6 x 5
## # Groups:   MovieId [6]
##   MovieId MovieName                     mean_rank YearOfRelease Genre
##     <int> <fct>                             <dbl>         <int> <fct>
## 1       3 Knives Out                            5          2019 Comedy
## 2       4 Once Upon a Time in Hollywood         5          2019 Comedy
## 3       1 Birds of Prey                         4          2020 Action
## 4       5 Avengers: Endgame                     4          2019 Action
## 5       2 The Irishman                          3          2019 Drama
## 6       6 Parasite                              3          2019 Comedy
```

```
#Step 3: Get Users Top rated Genre
users.topgenre.df <- ratings.df %>%
  filter(Ratings !=  "")  %>%
  group_by(UserId, UserName) %>%
  summarize(top_rank = max(Ratings))

users.topgenre.df <- left_join(users.topgenre.df, ratings.df, by = c("UserId" = "UserId", "UserName" ="U
users.topgenre.df <- left_join(users.topgenre.df, movies.df, by = c("MovieId" = "Id", "MovieName" ="Movi
users.topgenre.df <- subset(users.topgenre.df, select = c("UserId","UserName","Genre","top_rank"))

users.topgenre.df <- unique(users.topgenre.df)
users.topgenre.df
```

```
## # A tibble: 12 x 4
## # Groups:   UserId [7]
##    UserId UserName Genre  top_rank
##     <int> <fct>    <fct>     <int>
##  1      1 ABC      Action        5
##  2      1 ABC      Comedy        5
##  3      2 XYZ      Comedy        5
##  4      2 XYZ      Action        5
##  5      3 YYY      Action        4
##  6      3 YYY      Comedy        4
##  7      4 ZZZ      Action        5
##  8      4 ZZZ      Comedy        5
##  9      5 POP      Drama         5
## 10      5 POP      Comedy        5
## 11      6 XXX      Comedy        4
## 12      7 LMN      Action        5
```

3.a. Recommendations for user 3 YYY for Upcoming Movies:

```
user3.recommendations <- subset(users.topgenre.df,  users.topgenre.df$UserId==3,  select = c("UserId","
user3.recommendations <- inner_join(upcoming.movies.df,user3.recommendations, by =c("Genre" ="Genre" ,
user3.recommendations
```

```
##   Id         MovieName YearOfRelease  Genre AutoRank UserId UserName
## 1  2   Inside the Rain          2020 Comedy        4      3      YYY
## 2  7       Black Widow          2020 Action        4      3      YYY
## 3  9 Wonder Woman 1984          2020 Action        4      3      YYY
```

3.b. Recommendations for user 5 POP for Upcoming Movies:

```
user5.recommendations <- subset(users.topgenre.df,  users.topgenre.df$UserId==5,  select = c("UserId","
user5.recommendations <- inner_join(upcoming.movies.df,user5.recommendations, by =c("Genre" ="Genre" ,
user5.recommendations
```

```
##   Id      MovieName YearOfRelease  Genre AutoRank UserId UserName
## 1  3 Military Wives          2020 Comedy        5      5      POP
```