Analysis of IMDB Movie Database using HIVE

Bhavneet Soni

**Analysis of IMDB Movie Database using HIVE**

We will look at the data collected from IMDB, it was retrieved from kaggle.com (Kaggle.com/datasets, n.d.). The data set consists of a CSV file with 5043 movie records and had the following information about the movie.

1. *Color* – weather the movie is color or Black and White
2. *director\_name* – movies director name
3. *num\_critic\_for\_reviews* – Number of critic reviews
4. *duration –* Length of the movie
5. *director\_facebook\_likes* – Number of facebook likes the director got
6. *actor\_3\_facebook\_likes –* Number of Facebook Likes actor 3 got
7. *actor\_2\_name –* Name of actor 2
8. *actor\_1\_facebook\_likes* – No of Facebook likes actor 1 got
9. *gross –* Gross revenue earned by the movie
10. *genres –* Genres of movies the movie belong to (pipe (|) seperated)
11. *actor\_1\_name* – Actor 1 Name
12. *movie\_title –* Title of the movie
13. *num\_voted\_users* – number of votes from users of IMDB website
14. *cast\_total\_facebook\_likes –* Total number of Facebook likes all the cast members got in total
15. *actor\_3\_name* – Name of actor 3
16. *facenumber\_in\_poster* – Number of faces in the poster of the movie
17. *plot\_keywords* – Plot keywords (pipe (|) seperated)
18. *movie\_imdb\_link* – URL link for the movies official website
19. *num\_user\_for\_reviews –* Number of review movie got from users
20. *language* – Language in which the movie was made
21. *country* – Country of origin for the movie (where it was produced)
22. *content\_rating* – Content Rating
23. *budget –* Budget of the movie
24. *title\_year­* – Year the movie was released
25. *actor\_2\_facebook\_likes –* Number of Facebook likes actor 2 got
26. *imdb\_score* – Total imdb score for the movie
27. *aspect\_ratio* – aspect ratio the movie was released in
28. *movie\_facebook\_likes* – Total number of Facebook likes movie got

The focus of our analysis will be to find out the most liked and the actor who worked the most. For this we will retrieve the column with movie title, title year, actor names (actor 1 2 and 3), number of corresponding Facebook likes from the table. Most liked will be actor with most number of facebook likes. Busiest actor will be the one who is associated with most number of movie titles over all. Since the raw data had way too many coloumns for the scope of this assignment before inserting the data into a Hive table I removed redundant columns from CSV file and used it instead.

Commands used for querying the data are as follows

1. Create a DataBase to hold our tables using CREATE DB movies;
2. use movies; //use the db just created
3. create table with column names etc // defining table schema



1. Loading data from the modified\_movies\_database.csv into the newly created table using

LOAD DATA INPATH ‘users/mapr/modified\_movies\_database.csv INTO TABLE moviesInfo;

1. Once we have the data in to the table we will create a table as select for all the actors and insert data into it from movieInfo table actorNames, facebooklikes, movietitle and title year. Since there are 3 different column for actors, first we will get actor1 names and later on we will append values for actor 2 and actor 3 the table along with the year. Following query was used to achieve this

*CREATE TABLE actors as SELECT actor1name, actor1likes, movie, releaseyear FROM movieInfo*

*INSERT INTO actors SELECT actor2name, actor2likes, movie, releaseyear FROM movieInfo*

*INSERT INTO actors SELECT actor3name, actor3likes, movie, releaseyear FROM movieInfo*

1. There are some fields without data entered in them to make sure we do not have any null values that might skew our results, we will remove all the null values by using the following query

*INSERT OVERWRITE TABLE actors SELECT \* FROM actors WHERE actorName is not null and likes is not null and releaseyear is not null;*

1. To acertain all the data has been inserted successfully we will check the total count of the table rows in the new table by uing the following command

*select count(\*) from actors*

Result from the query was 14575 which seems about right

1. Once all the data has been inserted into actors table we will query it for the most number of facebook likes by using the following command

*SELECT actorname, likes FROM actors JOIN (SELECT max(likes) as maxlikes FROM actors) mostliked ON actors.likes = mostliked.maxlikes*

RESULT for the query was Darcy Donavan with 640000 likes (Never heard of her 😐)

1. For getting the busiest actor we performed the following queries on the actors table

*Select actorname, count(\*) as counter from actors group by actorname sort by counter desc limit 1*

RESULT for the query was Robert De Niro with 53 movies to his name



Effort was made to use the CLI for the maprsandbox to do the hive queries, but ended up using the HUE for creating tables, loading data and doing the queries.

References

*Kaggle.com/datasets*. (n.d.). Retrieved from Kaggle.com: https://www.kaggle.com/deepmatrix/imdb-5000-movie-dataset