INF 551 – Spring 2019

Homework #3:

Due: March 29, Friday (end of day)
100 points

In this homework, we consider the “Google Play Store Apps” data set at https://www.kaggle.com/lava18/google-play-store-apps. We have adapted this dataset into MySQL format and you are provided with a SQL script for creating and populating tables. You need to import the data into “inf551” database which should have “inf551” as user name and “inf551” (all without quotes) as password. Refer to usage on beers-tables.sql on how to run the script. After execution, your inf551 database should have additional two tables:

googleplaystore(App varchar(700) primary key, Category varchar(1000), Rating float, Reviews int, Size varchar(20), Installs varchar(20), Type varchar(80), Price float, Content_Rating varchar(20), Genres varchar(80), Last_Updated date, Current_Ver varchar(100), Android_Ver varchar(100));

googleplaystore_user_reviews(App varchar(700), Review varchar(767), Sentiment varchar(100), Sentiment_Polarity float, Sentiment_Subjectivity float, primary key (App, Review), foreign key (App) references googleplaystore(App));

Note that googleplaystore_user_reviews also has a fulltext index on its Review column, created by the following command:

create fulltext index review_idx on googleplaystore_user_reviews(Review);

1. [70 points] Write an SQL query for each of the following questions. For each query, also show its output.
   a. Find the top-10 (by the review counts) apps which have at least 100 reviews. Output the names of apps and the numbers of reviews . (Only use table googleplaystore. Result columns name: appName, reviewNum)
   b. Find out how many apps do not have reviews. (Consider apps which do not appear in googleplaystore_user_reviews table. Result column name: countApp)
      Write two versions of SQL queries for this question: one using “not in”; the other outer join.
   c. Find the names of apps which have at least two different reviews. (Only use table googleplaystore_user_reviews. Result column name: appName)
      (Requirement: do not use subquery.)
   d. Find the names of apps which have only one review. (Only use table googleplaystore_user_reviews. Result column name: appName)
      Write two versions of SQL queries for this question: one with aggregation; the other without.
   e. Find the most expensive app. Output the app name and its price. (Result column name: appName, price)

f. Find top-10 categories (ranked by the average price of apps in the category) of apps. Output categories and average prices, by the descending order of average prices. (Result column name: category, avgPrice)

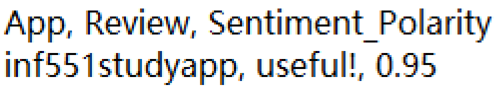
g. Find the number of free apps (by type) each of which has at least 10 positive (by its sentiment) reviews. (Result column name: countApp)

h. Find the number of apps each of which has at least one positive and at least one negative review. (Result column name: countApp)

2. [30 points] Write a python script "search.py" that takes a list of keywords (separated by spaces) and outputs the top-10 apps, along with their reviews and polarity scores (in descending order of average polarity score) for apps whose review contains one or more keywords in the list and whose polarity and subjectivity scores are both greater than .8. You need to output your result to a <first_name>_<last_name>_q2.csv file
(Install Python mysql-connector for this question)

Output Format:

The output file is a csv file, containing all results you have found. The header is: "App, Review, Sentiment_Polarity". There is no requirement for the number of decimals for the polarity value. Please refer to the format in the following figure.



```
App, Review, Sentiment_Polarity
inf551studyapp, useful!, 0.95
```

Example execution:

```
python search.py "perfect good"
```

You need to use "match … against" in MySQL whose details can be found here:

https://dev.mysql.com/doc/refman/5.5/en/fulltext-search.html

Submissions

1. Name your sql scripts and python as below and submit to Blackboard by the due time. ~~DO NOT place them in a folder or zip file.~~ All lowercase for your <FirstName> and <LastName>. Your submission must be a zip file with the naming convention: <FirstName>_<LastName>_hw3.zip. Directly pack your code in the zip file and don't contain any folder in it. You don't need to include any result.
   - <FirstName>_<LastName>_q1_a.sql
   - <FirstName>_<LastName>_q1_b_notin.sql (using "not in" solution)
   - <FirstName>_<LastName>_q1_b_join.sql (using "outer join" solution)
   - <FirstName>_<LastName>_q1_c.sql
   - <FirstName>_<LastName>_q1_d_agg.sql (using aggregation solution)
   - <FirstName>_<LastName>_q1_d_withoutagg.sql (not using aggregation solution)
   - <FirstName>_<LastName>_q1_e.sql
   - <FirstName>_<LastName>_q1_f.sql

- <FirstName>_<LastName>_q1_g.sql
- <FirstName>_<LastName>_q1_h.sql
- <FirstName>_<LastName>_search.py

Grading Policy:

1. Homework assignments are due at 11:59pm on the due date and should be submitted on Blackboard. Late homework will be deducted by 10% of its points for every 24 hours that it is late. No credit will be given after 72 hours of its due time.
2. If your python code cannot be run with the commands as above on EC2, there will be a 20 points penalty
3. If your SQL script cannot be run on EC2, there will be a 20 points penalty.
4. Please use Python 2.7 (installed by default on EC2) and MySQL (5.6) for the coursework, there will be a 20 points penalty if you use a different version.
5. If you don't follow the restriction of a question, there will be no point for this question.
6. If our grading program cannot find a specified tag/header/column name, there will be 20% penalty for this question.
7. If the filename of your program is wrong, there will be 20% penalty for this question.
8. There will be no point if the total execution time for all questions exceeds 2 minutes.
9. You can only use standard Python libraries and mysql-connector (i.e. external libraries like Numpy or Pandas are not allowed.)
10. If the content of your question 1 files is not a valid sql code, there will be 50% penalty for the question.