INFSCI 2710 Database Management, Fall 2022

Homework 2: SQL

100 pts

Due Date: 10/13. Please submit a pdf to the Canvas assignment.

Note: is no actual data provided in this assignment, you can assume there are plenty row of data been inserted to the database, **you just need to provide MYSQL statements**

Consider the following relational database from Yelp

Business

- Id
- Name
- Address
- City
- State
- Category: business category
- Stars: average stars from all reviewed users
- Review_count: total number of reviews
- Is_open: a Boolean indicate if it's still open or not

User

- Id
- Name
- Review_count number of reviews this user wrote
- Yelping_since: the date that user registered

- Fans: total number of fans

Review

- Id
- Stars
- Date
- Text
- Useful: total number of ‘useful’ rated by other users
- Funny: total number of ‘funny’ rated by other users
- Cool: total number of ‘cool’ rated by other users
- Business_id
- User_id

Tip

- User_id
- Business_id
- Text
- Date
- amount

Q1 [5 pt] write a SQL query to find name of all the business in the ‘fast food’ category

```
SELECT Name
FROM Business
WHERE Category = ‘fast food’
```

Q2 [10 pt] write a SQL query to find the top 3 users based on total number of fans

```
SELECT id, name, Fans
FROM user
ORDER BY Fans DESC
LIMIT 3
```

Q3 [10 pt] write a SQL query to find the number of users registered after 2020

```
SELECT COUNT (DISTINCT (ID))
FROM user
WHERE Yelping_since >= 2020-01-01;
```

Q4 [10 pt] write a SQL query to list the cities with the most reviews in descending order

```
SELECT city,
SUM (review_count) AS reviews
FROM business
GROUP BY city
ORDER BY reviews DESC
```

Q5 [10 pt] write SQL to update stars and review_counts for business named " yami sushi"

Seems like many of you misunderstand my intention, I should add 're-calculate the values based on the review table' to the prompt

```
UPDATE Business AS B
  INNER JOIN (
    SELECT
      AVG(R.Stars) avg_stars,
      COUNT(R.Id) count_id,
      R.Business_id
    FROM
      Review R
    GROUP BY
      R.Business_id
  ) AS R2 ON B.Id = R2.Business_id
SET
  B.Stars = R2.avg_stars,
  B.Review_count = R2.count_id
WHERE
  B.Name = "yami sushi";
```

Q6 [10 pt] write a SQL query to find the distribution of star ratings to the business in the Pittsburgh (return stars and corresponding counts)

```
SELECT stars, SUM (review_count) AS count
FROM business
WHERE city == 'Pittsburgh'
GROUP BY stars
```

Q7 [10 pt] for the restaurant named "yami sushi" write a SQL query to find if there are more reviews with the word 'love' or with the word 'hate'

```
SELECT C1. love_count , C2.hate_count,
FROM
  (
    SELECT
      COUNT (*) love_count
    FROM
      Business B
```

```
        INNER JOIN Review R ON B.Id = R.Business_id
    WHERE
        B.Name = "yami sushi"
        AND R.text LIKE '%love%'
) C1,
(
    SELECT
        COUNT (*) hate_count
    FROM
        Business B
        INNER JOIN Review R ON B.Id = R.Business_id
    WHERE
        B.Name = "yami sushi"
        AND R.text LIKE '%hate%'
) C2;
```

Q8 [10 pt] for each user, calculate average star, average number of "useful", "funny" and "cool" for all the reviews this user wrote

```
SELECT U.Name,
        AVG(R.Stars) avg_stars,
         AVG(R.Useful) avg_useful,
        AVG(R.Funny) avg_funny,
        AVG(R.Cool) avg_cool
FROM Review R
INNER JOIN User U ON R.User_id = U.Id
GROUP BY R.User_id;
```

Q9 [10 pt] Find the user who has the largest number of fans, return the name and amount for the top 5 businesses he tipped

```
SELECT B.Name, T.amount
FROM Tip T
INNER JOIN Business B ON T.Business_id = B.Id
WHERE User_id =
(
        SELECT User_id
        FROM User
```

```
        ORDER BY Fans DESC
        LIMIT 1
)
ORDER BY T.amount DESC
LIMIT 5;
```

Q10 [15 pt] Group business based on the one that are open and the ones that are closed, what differences you can find between those two groups (you can use different SQL to answer different sub questions)

1. Are there more reviews written for the business that are still open
2. Are the average star rating higher for business that are open than business that are closed
3. Are there more tips for business that are open than business that are closed

1.

```
SELECT Is_open, SUM(Review_count) AS review_counts
FROM Business
GROUP BY Is_open;
```

2.

```
SELECT Is_open, AVG(Stars)
FROM Business
GROUP BY Is_open;
```

3.

```
SELECT B.Is_open, SUM(T.amount)
FROM Business B
INNER JOIN Tip T ON B.Id = T.Business_id
GROUP BY B.Is_open;
```