100574229
MySQL for Data Analytics

Big Assignment 2024

- To complete this assignment, upload the "Yelp data.sql" database from Mycourse ["Exercises and database files" section] into HeidiSQL. The database has 16 tables containing information on reviews Yelp users have left for different businesses they have visited.
- Please read each question carefully and consider which information (attributes) you need from each table to arrive at the solution. Provide the answer in a designated space under each question and the MySQL code that generated the result by copying the code from HeidiSQL and pasting it into this document. If asked in the question, also attach screenshots of the results.
- Please check the rows of the data imported for each table (see Yelp Dataset Description.docx) and ensure you have imported the data correctly, not importing the data twice or importing insufficient data.
- If you see an error message like "the server has gone away," please contact the teacher to update the server settings. Aalto IT might update the server settings without informing the teacher, preventing you from importing large files.
- You are free to create additional columns or tables to support your analysis.
- Please submit the assignment as a Word document, not a PDF. A Word document allows us to copy and check your code easily.
- Grading method: If your answer is wrong but the code is correct, you will still receive a substantial minus in points. A wrong answer with the correct code is still considered a failure in real-life business.
- Good luck with the assignment! :)

100574229
MySQL for Data Analytics

Question 1 (4 points, each answer for 1 point)

In the business table, assume that the businesses with a rating below 2 belong to the very poor rating category; the businesses with ratings ["stars" variable] from 2 to below 3 are considered poor ratings; the businesses with ratings from 3 to below 4.5 are considered good ratings, and the businesses that have ratings 4.5 or above are considered excellent ratings. Only the businesses that are still active ("true" in the "active" column) are considered.

Please report the average review count for the businesses in each rating level.

Rating level	Average review count
Excellent_rating	15.4550
Good_rating	31.9416
Poor_rating	8.4493
Very_poor_rating	4.6823

MySQL code that generates the result:

```
SELECT
    CASE
        WHEN b.stars < 2 THEN 'Very Poor Rating'
        WHEN b.stars >= 2 AND b.stars < 3 THEN 'Poor Rating'
        WHEN b.stars >= 3 AND b.stars < 4.5 THEN 'Good Rating'
        WHEN b.stars >= 4.5 THEN 'Excellent Rating'
    END AS rating_level,
    AVG(b.review_count) AS average_review_count
FROM business b
WHERE b.active = 'true'
GROUP BY rating_level
ORDER BY FIELD(rating_level, 'Excellent Rating', 'Good Rating', 'Poor
Rating', 'Very Poor Rating');
```

Question 2 (6 points)

In the table friends, we can see the Yelp users who are friends. Find the user ['name' from table users] with the most friends, where one friendship is considered only once.

Amount of friends: 2032 (3 points)
"Name" [not user_id] of the user: Gabi (3 points)

MySQL code that generates the result:

```
SELECT u.name, f.friend_count AS amount_of_friends
FROM (
    SELECT user_id1 AS user_id, COUNT(DISTINCT user_id2) AS friend_count
    FROM friends
    GROUP BY user_id1
    ORDER BY friend_count DESC
    LIMIT 1
) AS f
JOIN users u ON f.user_id = u.user_id;
```

Question 3 (10 points)

a) In the table "business", there are different attributes for each business reviewed by users in Yelp. Consider the [city] column and find which city received the highest amount of reviews from those cities where the businesses received less than 100,000 reviews in total.

The city with the highest amount of reviews: Scottsdale (5 points)

MySQL code that generates the result:

```
SELECT city, SUM(review_count) AS total_reviews
FROM business
GROUP BY city
HAVING SUM(review_count) < 100000
ORDER BY total_reviews DESC
LIMIT 1;
```

b) In the city "Phoenix", what business category received the highest amount of reviews? Please provide the name of the business category and the number of reviews given to the category.

Business Category: Restaurants (2 points)
Number of reviews received: 100827 (3 points)

MySQL code that generates the result:

```
SELECT bc.category AS business_category, SUM(b.review_count) AS
total_reviews
FROM business b
JOIN business_categories bc ON b.business_id = bc.business_id
WHERE b.city = 'Phoenix'
GROUP BY bc.category
ORDER BY total_reviews DESC
LIMIT 1;
```

Question 4 (10 points)

In the table "users", please find the user's name [name] for that user who ranks as the 1st among all users regarding the number of fans. Also, find this user's rank in giving funny_votes [votes_funny].

User name: Connie (5 points)
Ranking in votes_funny: 7 (5 points)

MySQL code that generates the result:

```
-- Find user with the highest number of fans
SELECT u.user_id,
       u.name AS user_name,
       u.fans
FROM users u
ORDER BY u.fans DESC
LIMIT 1;

-- Funny ranking determination
SELECT u.name AS user_name,
       u.fans,
       u.votes_funny,
       RANK() OVER (ORDER BY u.votes_funny DESC) AS funny_votes_rank
FROM users u
WHERE u.user_id = (SELECT user_id
                   FROM users
                   ORDER BY fans DESC
                   LIMIT 1);
```

Question **5 (8 points)**

a) Table "users" contains information on reviews users have left for businesses. From those users who started Yelping between September 2010 and May 2011 [yelping_since_year] [yelping_since_month], what is the number of users who have written reviews but received no votes in funny [votes_funny], useful [votes_useful], and cool [votes_cool] categories in the "**reviews**" table?

Number of users: 4893 (4 points)

MySQL code that generates the result:

```
SELECT COUNT(*)
FROM users
WHERE yelping_since_year = 2010 AND yelping_since_month >= 9
   OR yelping_since_year = 2011 AND yelping_since_month <= 5
   AND votes_funny = 0 AND votes_useful = 0 AND votes_cool = 0;
```

b) From the users that have reviewed businesses located in the city "Phoenix", please identify the user with the highest total amount of reviews written ["review_acount" in the "users" table], and give the name of the user [name] and the total number of reviews written ["review_acount" in the "users" table].

The name of the user: Bruce (2 points)
The number of reviews: 3286 (2 points)

MySQL code that generates the result:

```
SELECT u.name, u.review_count
FROM users u
JOIN (
    SELECT r.user_id
    FROM reviews r
    JOIN business b ON r.business_id = b.business_id
    WHERE b.city = 'Phoenix'
    GROUP BY r.user_id
) AS phoenix_reviewers ON u.user_id = phoenix_reviewers.user_id
ORDER BY u.review_count DESC
LIMIT 1;
```

Question 6 (8 points)

Explore "business", "business_attribute_goodfor" and "business_hours" tables, and answer the following question:

What is the business name [business_name], the opening time [opening_time] on Sundays of a business that is currently **active**, has been reviewed as good for "breakfast" in the table "business_attribute_goodfor" and has received the star rating 5 among other similar businesses?

Note: you can find the data about the "active" and "stars" of a business in the "business" table; "opening_time_hours" at "business_hours" table; "subattribute" and true or false "value" at "business_attributes_goodfor" table.

The name of the business [NOT business_id]: Rayner's Chocolate & Coffee Shop (4 points)
Opening time: 06:00:00 (4 points)

MySQL code that generates the result:

```
SELECT b.business_name, bh.opening_time
FROM business b
JOIN business_attributes_goodfor bag ON b.business_id = bag.business_id
JOIN business_hours bh ON b.business_id = bh.business_id
WHERE b.active = 'true'
  AND b.stars = 5
  AND bag.subattribute = 'breakfast'
  AND bag.value = 'true'
  AND bh.day_of_week = 'Sunday'
LIMIT 1;
```

Question 7 (8 points)

Use the tables "reviews" and "business" to answer this question. Please find the business name [business_name] that has received the highest proportion of star ratings [stars] **above** 4 out of all their reviews. Consider only those businesses which have over 800 reviews in total.

The name of the business [NOT business_id]: Cibo (4 points)
The ratio of the ratings for the business: 0.5174 (4 points)

MySQL code that generates the result:

```
SELECT b.business_name,
       (SUM(CASE WHEN r.stars > 4 THEN 1 ELSE 0 END) / COUNT(r.review_id))
AS high_rating_ratio
FROM reviews r
JOIN business b ON r.business_id = b.business_id
GROUP BY b.business_id, b.business_name
HAVING COUNT(r.review_id) > 800
ORDER BY high_rating_ratio DESC
LIMIT 1;
```

Question 8 (8 points)

The table "checkin" represents a simplified report on the amounts of customers checking in per each hour from Mondays to Sundays. Which are the top 3 busiest hotels in terms of checkins on Mondays that are currently **active**? Consider those businesses that have a business category only in "**Hotels**" [see "business_categories" table, whether a business is of "Hotels & Travel" category or other similar is not relevant here] and the number of checkins only during the evening hours between 18 pm and 23 pm [time_18] – [time_23]. Provide the name of the hotel, star rating, and total amount of customers checking for that business, which had the highest star rating among the three busiest hotels on Monday evenings.

The name of the business [NOT business_id]: Hotel Valley Ho (2 points)
Star rating: 4 (3 points)
The total number of customers checking in (18 pm-23 pm): 53 (3 points)

MySQL code that generates the result:

```
SELECT b.business_name,
       b.stars AS star_rating,
```

```
        SUM(c.time_18 + c.time_19 + c.time_20 + c.time_21 + c.time_22 +
c.time_23) AS total_checkins
FROM checkins c
JOIN business b ON c.business_id = b.business_id
JOIN business_categories bc ON b.business_id = bc.business_id
WHERE c.day_of_week = 'Monday'
  AND b.active = 'true'
  AND bc.category = 'Hotels'
GROUP BY b.business_id, b.business_name, b.stars
ORDER BY total_checkins DESC, b.stars DESC
LIMIT 3;
```

Question 9 (8 points, each answer for 2 points)

Assuming that when a hair salon specializes only in certain areas rather than providing all different kinds of services, the quality tends to be better. Your task is to determine if the star ratings for businesses in the table "business_attribute_hairtypesspecializedin" tend to be better or worse when their areas of specialization increase or decrease.

To answer, please create four specialization categories based on the amount of true values in the subattribute categories each business_id in the "business_attribute_hairtypesspecializedin" has. For the categorization, the businesses that specialize in 8 or 7 categories belong to the full_specialities category; the businesses with 6 or 5 categories belong to the multiple_specialities category; the businesses with 4 or 3 categories belong to some_specialities category, and the businesses with 2 or 1 categories belong to few_specialities category. You can obtain "stars" values from the "business" table.

Please report the average stars for the businesses in each specialty category.

Speciality category	Average stars
Full_specialities (8 – 7)	4.0417
Multiple_specialities (6 – 5)	4.65
Some_specialities (4 – 3)	4.5
Few_specialities (2 – 1)	4.875

MySQL code that generates the result:

```
SELECT
    CASE
        WHEN specialization_count >= 7 THEN 'Full_specialities'
        WHEN specialization_count = 5 OR specialization_count = 6 THEN
'Multiple_specialities'
        WHEN specialization_count = 3 OR specialization_count = 4 THEN
'Some_specialities'
        WHEN specialization_count = 1 OR specialization_count = 2 THEN
'Few_specialities'
    END AS speciality_category,
    AVG(b.stars) AS average_stars
FROM (
    SELECT bah.business_id, COUNT(*) AS specialization_count
    FROM business_attributes_hairtypesspecializedin bah
    WHERE bah.value = 'true'
    GROUP BY bah.business_id
) AS specialization_counts
JOIN business b ON specialization_counts.business_id = b.business_id
GROUP BY speciality_category;
ORDER BY FIELD(speciality_category, 'full_specialities',
'multiple_specialities', 'some_specialities', 'few_specialities');
```

Question 10 (10 points)

In the “tip” table in the Yelp database, based on the column “tip_text” that have at least two words:

a) Was the frequency of the word “food” as the first or second word more popular? Please ignore whether the letter in the word is an upper or lower case. Please give frequencies:

The word “Food” as 1st word: 346 (2 points)
The word “Food” as 2nd word: 1358 (3 points)

b) What is the most popular word when food was the second word?

The most popular first word with the word “food” as second word: great (2 points)
Frequency of the word: 477 (3 points)

Please remember to drop seven punctuation marks from the title, including:

“	”	"	-	,	!	`

We do not consider removing more punctuation marks to obtain consistent results. Also, please remove empty spaces from both sides of the title.

MySQL code that generates the result:

```
WITH CleanedTips AS (
    SELECT LOWER(

REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(tip_text, '"', ''),
'“', ''), '”', ''), '-', ''), ',', ''), '!', ''), '`', '')
           ) AS cleaned_text
    FROM tips
    WHERE LENGTH(TRIM(tip_text)) - LENGTH(REPLACE(TRIM(tip_text), ' ', ''))
>= 1  -- Ensures at least two words
),
FirstSecondWordCounts AS (
    SELECT
        COUNT(CASE WHEN SUBSTRING_INDEX(cleaned_text, ' ', 1) = 'food' THEN
1 END) AS food_as_first,
        COUNT(CASE WHEN SUBSTRING_INDEX(SUBSTRING_INDEX(cleaned_text, ' ',
2), ' ', -1) = 'food'
                   AND SUBSTRING_INDEX(cleaned_text, ' ', 1) != 'food' THEN
1 END) AS food_as_second
    FROM CleanedTips
),
MostCommonFirstWord AS (
    SELECT SUBSTRING_INDEX(cleaned_text, ' ', 1) AS first_word,
           COUNT(*) AS frequency
    FROM CleanedTips
    WHERE SUBSTRING_INDEX(SUBSTRING_INDEX(cleaned_text, ' ', 2), ' ', -1) =
'food'
    GROUP BY first_word
    ORDER BY frequency DESC
    LIMIT 1
)
-- Output
```

```
SELECT
    fsc.food_as_first AS "The word 'Food' as 1st word",
    fsc.food_as_second AS "The word 'Food' as 2nd word",
    mcfw.first_word AS "Most popular first word with 'food' as second
word",
    mcfw.frequency AS "Frequency of the word"
FROM FirstSecondWordCounts fsc, MostCommonFirstWord mcfw;
```

Question 11 (10 points)

Based on the "review" table, please compare the total amounts of reviews for each business for 2013 and 2012, respectively, and report the name of the business [business_name] and the difference in reviews for the business that had the highest decrease in number of reviews received from 2012 to 2013.

Name of the business [NOT business_id]: Matt's Big Breakfast (3 points)
The decrease in reviews from 2012 to 2013: 107 (7 points)

My SQL code that generates the result:

```
WITH reviews_by_year AS (
    SELECT
        business_id,
        YEAR(review_date) AS review_year,
        COUNT(review_id) AS total_reviews
    FROM reviews
    WHERE YEAR(review_date) IN (2012, 2013)
    GROUP BY business_id, review_year
),
review_diff AS (
    SELECT
        r2012.business_id,
        b.business_name,
        r2012.total_reviews AS reviews_2012,
        r2013.total_reviews AS reviews_2013,
        (r2012.total_reviews - COALESCE(r2013.total_reviews, 0)) AS
review_difference
    FROM reviews_by_year r2012
    LEFT JOIN reviews_by_year r2013 ON r2012.business_id =
r2013.business_id AND r2013.review_year = 2013
    JOIN business b ON r2012.business_id = b.business_id
    WHERE r2012.review_year = 2012
)
SELECT
    business_name,
    review_difference
FROM review_diff
ORDER BY review_difference DESC
LIMIT 1;
```

Question 12 (10 points)

Analyze the revisiting customer's satisfaction on their next visit to a business branch. You will only need the table "reviews" for this task.

Users[user_id] can leave a lower star rating [stars] on their first visit [review_date] to a business [business_id], but give a higher star rating on their next visit to the same business, indicating that the customer revisited the business and is more satisfied on their next visit. Please count the number of users who rated a business higher on their second [or third, et.] visit than their first visit.

Please consider only the reviews from **2012** and thus it is recommended to filter only rows with reviews written in 2012 first to reduce the run time of the SQL commands.

Note1: If one user visits different businesses, all the visits count, and the same user could be counted several times in the analysis.

Note 2: If a user wrote two or more reviews for a business within the same day, this user's reviews for the specific business are considered untrustworthy. Thus, we drop this user's reviews for this specific business. However, the user's reviews on other businesses will be retained if no more untrustworthy reviews are found.

Amount of users who rated a business higher on their second visit: 251

MySQL code that generates the result:

```
-- Create a temporary table to select reviews from 2012
WITH reviews_2012 AS (
    SELECT
        r.review_id,
        r.user_id,
        r.business_id,
        r.stars,
        r.review_date
    FROM reviews r
    WHERE YEAR(r.review_date) = 2012
),

-- Filter valid reviews where users did not submit multiple reviews for the
same business on the same day
valid_reviews AS (
    SELECT
        rwc.user_id,
        rwc.business_id,
        rwc.review_date,
        rwc.stars
    FROM (
        SELECT
            r2012.user_id,
            r2012.business_id,
            r2012.review_date,
            r2012.stars,
            COUNT(r2012.review_id) OVER (PARTITION BY r2012.user_id,
r2012.business_id, r2012.review_date) AS review_count
        FROM reviews_2012 r2012
    ) AS rwc
    WHERE rwc.review_count = 1
),

-- Rank user visits to the same business by review date
ranked_visits AS (
    SELECT
        vr.user_id,
        vr.business_id,
        vr.stars,
```

```
        vr.review_date,
        RANK() OVER (PARTITION BY vr.user_id, vr.business_id ORDER BY
vr.review_date) AS visit_rank
    FROM valid_reviews vr
)

-- Count users who rated a business higher on a subsequent visit
SELECT
    COUNT(DISTINCT rv1.user_id) AS user_count
FROM ranked_visits rv1
JOIN ranked_visits rv2
    ON rv1.user_id = rv2.user_id
    AND rv1.business_id = rv2.business_id
WHERE rv1.visit_rank = 1
  AND rv2.visit_rank > 1
  AND rv2.stars > rv1.stars;
```