**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! :)

Jesse Paananen

1018596

Last edit on 1.11.2024

**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 | 4.73777889195287 |
| Good\_rating | 3.5880120403088602 |
| Poor\_rating | 2.3481280193236715 |
| Very\_poor\_rating | 1.29082774049217 |

MySQL code that generates the result:

**SELECT** 'Very\_poor\_rating' **AS** Rating\_level, **AVG**(stars) **AS** Average\_star\_rating

**FROM** business

**WHERE** business.`active` = 'true' **AND** stars < 2

**UNION** **ALL**

**SELECT** 'Poor\_rating' **AS** Rating\_level, **AVG**(stars) **AS** Average\_star\_rating

**FROM** business

**WHERE** business.`active` = 'true' **AND** stars >= 2 **AND** stars < 3

**UNION** **ALL**

**SELECT** 'Good\_rating' **AS** Rating\_level, **AVG**(stars) **AS** Average\_star\_rating

**FROM** business

**WHERE** business.`active` = 'true' **AND** stars >= 3 **AND** stars < 4.5

**UNION** **ALL**

**SELECT** 'Excellent\_rating' **AS** Rating\_level, **AVG**(stars) **AS** Average\_star\_rating

**FROM** business

**WHERE** business.`active` = 'true' **AND** stars >= 4.5;

**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** users.`name`, **COUNT**(f.user\_id) **AS** friend\_count

**FROM** users

**JOIN** (

**SELECT** a.user\_id2 **AS** user\_id

**FROM** friends a

**JOIN** friends b **ON** b.user\_id2 = a.user\_id1 **AND** b.user\_id1 = a.user\_id2 **AND** b.user\_id1 > a.user\_id1

**UNION** **ALL**

**SELECT** a.user\_id1 **AS** user\_id

**FROM** friends a

**JOIN** friends b **ON** b.user\_id2 = a.user\_id1 **AND** b.user\_id1 = a.user\_id2 **AND** b.user\_id1 > a.user\_id1

) f **ON** users.user\_id = f.user\_id

**GROUP** **BY** users.user\_id, users.`name`

**ORDER** **BY** friend\_count **DESC**

**LIMIT** 1;

**Question 3 (10 points)**

1. 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**(**78983 reviews)** (5 points)

**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;

1. 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** category, **SUM**(review\_count) **AS** total\_reviews

**FROM** business

**JOIN** business\_categories **ON** business.business\_id = business\_categories.business\_id

**WHERE** business.city = 'Phoenix'

**GROUP** **BY** 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 amount of fans. Also, find this user’s rank in giving useful votes [votes\_funny].

User name: **Connie** (5 points)

Ranking in votes\_funny: **7** (5 points)

MySQL code that generates the result:

**WITH** RankedUsers **AS** (

**SELECT**

users.`name`,

users.`votes\_funny`,

users.`fans`,

**RANK**() **OVER** (**ORDER** **BY** votes\_funny **DESC**) **AS** funny\_rank

**FROM** users

)

**SELECT**

RankedUsers.`name`,

RankedUsers.funny\_rank

**FROM** RankedUsers

**ORDER** **BY** fans **DESC**

**LIMIT** 1;

Question **5 (8 points)**

1. 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: **7946** (4 points)

MySQL code that generates the result:

**SELECT** **COUNT**(**DISTINCT** users.user\_id) **AS** user\_count

**FROM** users

**JOIN** reviews **ON** users.user\_id = reviews.user\_id

**WHERE** ((yelping\_since\_year = 2010 **AND** yelping\_since\_month >= 9) **OR**

(yelping\_since\_year = 2011 **AND** yelping\_since\_month <= 5))

**AND** reviews.votes\_funny = 0

**AND** reviews.votes\_useful = 0

**AND** reviews.votes\_cool = 0;

1. 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** users.`name`, users.review\_count

**FROM** users

**JOIN** reviews **ON** users.user\_id = reviews.user\_id

**JOIN** business **ON** reviews.business\_id = business.business\_id

**WHERE** business.city = 'Phoenix'

**ORDER** **BY** users.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** bag.business\_id = bh.business\_id

**WHERE** bh.day\_of\_week = 'Sunday' **AND** b.`active` = 'true' **AND** b.stars = 5

**AND** bag.subattribute = 'breakfast' **AND** bag.`value` = 'true';

**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]: **Postino** **Arcadia** (4 points)

The ratio of the ratings for the business: **0.5840** (4 points)

MySQL code that generates the result:

**SELECT** b.business\_name, (**COUNT**(r.stars) / b.review\_count) **AS** ratio\_of\_ratings

**FROM** business b

**JOIN** reviews r **ON** b.business\_id = r.business\_id

**WHERE** b.review\_count > 800 **AND** r.stars > 4

**GROUP** **BY** b.business\_id, b.business\_name

**ORDER** **BY** ratio\_of\_ratings **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:

**WITH** top\_3\_hotels **AS** (

**SELECT** b.business\_name, b.stars, (c.time\_18 + c.time\_19 + c.time\_20 + c.time\_21 + c.time\_22 + c.time\_23) **AS** total\_checkins\_18\_to\_23

**FROM** business b

**JOIN** checkins c **ON** b.business\_id = c.business\_id

**JOIN** business\_categories bc **ON** b.business\_id = bc.business\_id

**WHERE** b.**active** = 'true' **AND** bc.category = 'Hotels' **AND** c.day\_of\_week = 'Monday'

**ORDER** **BY** total\_checkins\_18\_to\_23 **DESC** **LIMIT** 3

)

**SELECT** business\_name, stars, total\_checkins\_18\_to\_23

**FROM** top\_3\_hotels

**ORDER** **BY** stars **DESC** **LIMIT** 1;

**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) | 4.041666666666667 |
| Multiple\_specialities (6 – 5) | 4.65 |
| Some\_specialities (4 – 3) | 4.5 |
| Few\_specialities (2 – 1) | 4.875 |

MySQL code that generates the result:

**WITH** hairtype\_counts **AS** (

**SELECT** business\_id, **COUNT**(**CASE** **WHEN** bah.`value` = 'true' **THEN** 1 **END**) **AS** true\_count

**FROM** business\_attributes\_hairtypesspecializedin bah

**GROUP** **BY** business\_id),

business\_category **AS**(

**SELECT** business\_id, true\_count,

**CASE**

**WHEN** true\_count >= 7 **THEN** 'Full specialities (8-7)'

**WHEN** true\_count >= 5 **THEN** 'Multiple specialities (6-5)'

**WHEN** true\_count >= 3 **THEN** 'Some specialities (4-3)'

**ELSE** 'Few specialities (2-1)'

**END** **AS** speciality\_category

**FROM** hairtype\_counts),

business\_stars **AS**(

**SELECT** b.stars, bc.business\_id, bc.speciality\_category

**FROM** business\_category bc

**JOIN** business b **ON** bc.business\_id = b.business\_id

)

**SELECT** speciality\_category, **AVG**(stars) **AS** average\_stars

**FROM** business\_stars

**GROUP** **BY** speciality\_category;

**Question 10 (10 points)**

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

1. 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: **356**(2 points)

The word “Food” as 2nd word: **1358**(3 points)

1. 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:

**CREATE** **TEMPORARY** **TABLE** cleaned\_tips **AS**

**SELECT**

**LOWER**(**TRIM**(**REPLACE**(**REPLACE**(**REPLACE**(**REPLACE**(**REPLACE**(**REPLACE**(**REPLACE**(tip\_text, '"', ''), '“', ''), '”', ''), '-', ''), ',', ''), '!', ''), '`', ''))) **AS** cleaned\_text

**FROM** tips;

**CREATE** **TEMPORARY** **TABLE** word\_positions **AS**

**SELECT**

cleaned\_text,

**SUBSTRING\_INDEX**(cleaned\_text, ' ', 1) **AS** first\_word,

**CASE** **WHEN** **LENGTH**(cleaned\_text) - **LENGTH**(**REPLACE**(cleaned\_text, ' ', '')) >= 1

**THEN** **SUBSTRING\_INDEX**(**SUBSTRING\_INDEX**(cleaned\_text, ' ', 2), ' ', -1)

**ELSE** **NULL**

**END** **AS** second\_word

**FROM** cleaned\_tips;

**SELECT**

**SUM**(**CASE** **WHEN** first\_word = 'food' **THEN** 1 **ELSE** 0 **END**) **AS** food\_as\_first\_word,

**SUM**(**CASE** **WHEN** second\_word = 'food' **THEN** 1 **ELSE** 0 **END**) **AS** food\_as\_second\_word

**FROM** word\_positions;

**SELECT**

first\_word,

**COUNT**(\*) **AS** frequency

**FROM** word\_positions

**WHERE** second\_word = 'food'

**GROUP** **BY** first\_word

**ORDER** **BY** frequency **DESC**

**LIMIT** 1;

**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]: **Nimbus American Bistro N' Brewery**(3 points)

The decrease in reviews from 2012 to 2013: **78**(7 points)

My SQL code that generates the result:

**SELECT** business.business\_name,

(**COUNT**(**CASE** **WHEN** **YEAR**(review\_date) = 2012 **THEN** 1 **END**) -

**COUNT**(**CASE** **WHEN** **YEAR**(review\_date) = 2013 **THEN** 1 **END**)) **AS** review\_decline

**FROM** business

**JOIN** reviews **ON** reviews.business\_id = business.business\_id

**GROUP** **BY** business.business\_name

**ORDER** **BY** review\_decline **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: **282**

MySQL code that generates the result:

**WITH** FilteredReviews **AS** (

**SELECT** \*

**FROM** reviews

**WHERE** **YEAR**(review\_date) = 2012

),

NonDuplicateReviews **AS** (

**SELECT** \*

**FROM** FilteredReviews r

**WHERE** **NOT** **EXISTS** (

**SELECT** 1

**FROM** FilteredReviews sub

**WHERE** r.business\_id = sub.business\_id

**AND** r.user\_id = sub.user\_id

**AND** r.review\_date = sub.review\_date

**GROUP** **BY** sub.business\_id, sub.user\_id, sub.review\_date

**HAVING** **COUNT**(\*) > 1

)

),

RankedReviews **AS** (

**SELECT**

business\_id,

user\_id,

stars,

review\_date,

**ROW\_NUMBER**() **OVER** (**PARTITION** **BY** business\_id, user\_id **ORDER** **BY** review\_date) **AS** visit\_number

**FROM** NonDuplicateReviews

),

FirstAndLaterVisits **AS** (

**SELECT**

f.business\_id,

f.user\_id,

f.stars **AS** first\_visit\_stars,

l.stars **AS** later\_visit\_stars

**FROM** RankedReviews f

**JOIN** RankedReviews l

**ON** f.business\_id = l.business\_id

**AND** f.user\_id = l.user\_id

**AND** f.visit\_number = 1

**AND** l.visit\_number = 2

)

**SELECT** **COUNT**(\*) **AS** increased\_satisfaction\_count

**FROM** FirstAndLaterVisits

**WHERE** later\_visit\_stars > first\_visit\_stars;