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)

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(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;
```

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 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)**

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: **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;
```

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 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:

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: **356**(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:
```
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;
```