Assignment for the course MySQL for Data Analytics 2023

Student ID: 887799 Name: Nguyen Xuan Binh

- **In business, the result is often more important than the process.** If the final answer is wrong, you will get only a few points even if most of your MySQL codes are correct.
- **Please do not copy codes from other students.** Cheating (such as copying) on assignments will lead to course failure.
- **You can add new columns to a table or create additional tables to support your analysis.**
- **The assignment has a total of 88 points, which will be standardized to 100 points for grading.**

1. In the table orders at the car retailer (**classicmodels**) database, what is the customerNumber of the customer who has the highest frequency of placing orders to the company in 2004 [orderDate in 2004]? You don't need to consider whether the products have been finally shipped or not.

   The customerNumber of the customer: **141** (6 points)

   The highest frequency is: **9** (6 points)

   The MySQL code that generates the result:

```
SELECT customerNumber, COUNT(*) AS frequency FROM orders
WHERE YEAR(orderDate) = 2004
GROUP BY customerNumber
ORDER BY frequency DESC
LIMIT 1;
```

   The result query table of Question 1

customerNumber	frequency
141	9

2. In the **classicmodels** database, customer names and customer numbers can be found [see table "customers"]. Customers make different payments on different dates [See table "payments"]. Please specify the names of two customers who are most **often** (count frequency) to make payments during the weekend [12 points].
   Name of Customer 1: **Mini Gifts Distributors Ltd.**
   Name of Customer 2: **Marseille Mini Autos**
   Note: i) Please provide "customerName", not the contact names from table "customers"

   The MySQL code that generates the result:

```
SELECT c.customerName, COUNT(*) AS weekendPaymentCount
FROM payments AS p
JOIN customers AS c ON p.customerNumber = c.customerNumber
WHERE DAYOFWEEK(p.paymentDate) IN (1, 7)
GROUP BY c.customerName
ORDER BY weekendPaymentCount DESC
LIMIT 2;
```

   The result query table of Question 2

customerName	weekendPaymentCount
Mini Gifts Distributors Ltd.	3
Marseille Mini Autos	3

3. In the classicmodels database, one sales representative is responsible for one or several customers [see "salesRepEmployeeNumber" in table "customers"]. Customers make different payments on different dates [See table "payments"]. In other words, we can say sales representatives help the company get customers to make payments. Now the question is, who is the sales representative that brings the most **revenue** [or **the total amount of the payments from customers in the table payments**] to the company?
   Note: i) Ignoring those customers who are not assigned to any sales representative – they have not made any purchases yet.

   The salesRepEmployeeNumber of the sales representative who brings the most **revenue** is: **1370** [12 points].

   The MySQL code that generates the result:

```
SELECT c.salesRepEmployeeNumber, SUM(p.amount) AS total_revenue
FROM customers AS c
JOIN payments AS p ON c.customerNumber = p.customerNumber
WHERE c.salesRepEmployeeNumber IS NOT NULL
GROUP BY c.salesRepEmployeeNumber
ORDER BY total_revenue DESC
LIMIT 1;
```

   The result query table of Question 3

salesRepEmployeeNumber	total_revenue
1 370	1 112 003.8099999998

4. In the **cfpb_complaints_2500** database, you can find "closed with relief" in the **Company_response** column. Please find the **name of the company** that has the highest **ratio** of cases that is '**closed with relief**'? Only those companies with more than 30 cases [all different kinds of cases, no matter whether they are featured with 'closed with relief' or not] in the database are considered (10 points).
   - **ratio** [or percentage] for an individual company is calculated as:
     The amount of its cases featured with 'closed with relief' / the amount of its total cases

   The name of the company: **Barclays** (5 points)

   The ratio of the cases for the company: **0.4286** (5 points)

   The MySQL code that generates the result:

```
SELECT Company,
    SUM(CASE WHEN Company_response = 'Closed with relief' THEN 1 ELSE 0 END) AS ReliefCases,
    COUNT(*) AS TotalCases,
    (SUM(CASE WHEN Company_response = 'Closed with relief' THEN 1 ELSE 0 END) / COUNT(*)) AS ReliefRatio
FROM cfpb_complaints_2500
GROUP BY Company
HAVING COUNT(*) > 30
ORDER BY ReliefRatio DESC
LIMIT 5;
```

The result query table of Question 4

Company	ClosedWithReliefCount	TotalCases	ReliefRatio
Barclays	15	35	0.4286
Citibank	101	267	0.3783
GE Capital Retail	23	72	0.3194
SunTrust Bank	12	44	0.2727
Capital One	55	211	0.2607

5. In the **cfpb_complaints_2500** database, many complaints are related to 'loan' (those cases where the word 'loan' is included in the column 'Issue'). Please specify **the name of the company** that has the most issues related to '**loan'** on **Wednesday (DATA_received)**. Only complaints with the column '**State'** starting with character "**A**" are considered (8 points).

   The name of the company: **Bank of America** (8 points)

   The MySQL code that generates the result:

```
SELECT Company, COUNT(*) AS loan_issues_Wednesday
FROM cfpb_complaints_2500
WHERE Issue LIKE '%loan%'
AND DAYOFWEEK(Data_received) = 4
AND State LIKE 'A%'
GROUP BY Company
ORDER BY loan_issues_Wednesday DESC
LIMIT 1;
```

   The result query table of Question 5

Company	loan_issues_Wednesday
Bank of America	3

6. In the Chile database, let's assume that an income less than 10,000 is a low income; an income between 10,000 and 100,000 is a middle income; an income higher than 100,000 is a high income. We would like to know whether the income level and the statusquo have a certain relationship for females who voted yes to Pinochet. To answer this question, you need to provide the average statusquo value for the females who voted yes to Pinochet in correspondence to their different income levels. (9 points)

   [Please carefully read the question so that you will not miss any important condition when answering the question; Please provide **three** digits after the decimal point in the results]

Suggestion: you may need to update the table by adding a new column of income_level to answer the question.

Income level	Mean statusquo
High_income	**1.077**
Middle_income	**0.927**
Low_income	**0.937**

The MySQL code that generates the result:

```
# We add a column "income_level" to the "chile" table

ALTER TABLE chile ADD COLUMN income_level VARCHAR(30);

# Then, we populate the "income_level" based on the "income" column

UPDATE chile SET income_level = CASE
   WHEN income < 10000 THEN 'Low_income'
   WHEN income > 100000 THEN 'High_income'
   ELSE 'Middle_income'
END;

# Finally, we calculate the mean statusquo value for the females who voted yes
# After obtaining the table, we order the mean statusquo by high, middle and low income
```

```
SELECT income_level, ROUND(AVG(statusquo), 3) AS "Mean statusquo" FROM chile
WHERE vote = 'Y' AND sex = 'F'
GROUP BY income_level
ORDER BY CASE
    WHEN income_level = 'High_income' THEN 1
    WHEN income_level = 'Middle_income' THEN 2
    WHEN income_level = 'Low_income' THEN 3
END;

# We drop the income_level column from the "chile" table
# so the queries above can be rerun without error

ALTER TABLE chile DROP COLUMN income_level;
```

The result query table of Question 6

income_level	Mean statusquo
High_income	1.077
Middle_income	0.927
Low_income	0.937

7. Based on the use of cfpb consumer complaint database (2500 rows), please count the frequency of the complaints that satisfy the following three conditions at the same time: i) consumer finally disputed with the company (*Consumer_disputed*); and ii) were received on Friday (*Data_received*) and iii) the difference between *Data_received* and *Data_sent_to_company* is more than 5 days. (10 points)
Please specify the name of the company that has the biggest amount of the above-mentioned complaints.

The name of the company: **Bank of America** (5 points)

Frequency of the mentioned complaints of the company: **13** (5 points)

The MySQL code that generates the result:

```
SELECT Company, COUNT(*) AS mentionedComplaintsCount
FROM cfpb_complaints_2500
# i) consumer finally disputed with the company (Consumer_disputed)
WHERE Consumer_disputed = 'Yes'
# ii) were received on Friday (Data_received)
AND DAYOFWEEK(Data_received) = 6
# iii) the difference between Data_received and Data_sent_to_company is more than 5 days
AND DATEDIFF(Data_sent_to_company, Data_received) > 5
GROUP BY Company
ORDER BY mentionedComplaintsCount DESC
LIMIT 5;
```

The result query table of Question 7

Company	mentionedComplaintsCount
Bank of America	13
Wells Fargo	6
JPMorgan Chase	4
Capital One	4
GE Capital Retail	2

8. In the data "tripadvisor_review_sample_without_reviewtext", based on the review titles that have **at least two words** [a total of 5 points]:
   1) Regarding the **first** word used in the review title, what is the most popular word [case-insensitive]?

   The word is: **great** [1 points] The frequency of the word is: **2144** [1 points]

   2) Regarding the **second** word used in the review title, what is the most popular word [case-insensitive]?

   The word is: **hotel** [1 points] The frequency of the word is: **1420** [1 points]

   3) For those titles that **start with** the words "bad" OR "terrible", what are the most popular **second** word [case-insensitive]:
   The word is: **experience** [1 points]

   Note: Before answering question 8, please clean the title based on the following two requirements.
   i) Please remember to first **remove** six types of punctuation marks from the title, including:

“	”	"	-	,	!

In order to obtain consistent results, please do not remove more punctuation marks than we specified above and also do not replace the punctuation with an empty space.
   ii) Please remove empty spaces from both sides of the title.

The MySQL code that generates the result:

```
# Create a new column of clean_title

ALTER TABLE tripadvisor_review_sample_without_reviewtext
ADD COLUMN cleaned_title VARCHAR(200);

#  i) First, we remove six types of punctuation marks from the title, including:
#    “   ”    "   -   ,   !
# We do not replace the punctuation with an empty space

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(title, '“', '');

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(cleaned_title, '”', '');

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(cleaned_title, '"', '');
```

```
UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(cleaned_title, '-', '');

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(cleaned_title, ',', '');

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = REPLACE(cleaned_title, '!', '');

# ii) Please remove empty spaces from both sides of the title.
# TRIM() function removes leading and trailing spaces from a string.

UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = TRIM(cleaned_title);

# Set all titles to lowercase, since the queries ask for case-insensitive
UPDATE tripadvisor_review_sample_without_reviewtext
SET cleaned_title = LOWER(cleaned_title);

# 1. Most Popular First Word

SELECT SUBSTRING_INDEX(cleaned_title, ' ', 1) AS firstWord, COUNT(*) AS
firstWordCount
FROM tripadvisor_review_sample_without_reviewtext
# Title has at least 2 words if it contains at least 1 white space
WHERE LENGTH(cleaned_title) - LENGTH(REPLACE(cleaned_title, ' ', '')) >= 1
GROUP BY firstWord
ORDER BY firstWordCount DESC
LIMIT 1;


# 2. Most Popular Second Word

SELECT SUBSTRING_INDEX(SUBSTRING_INDEX(cleaned_title, ' ', 2), ' ', -1) AS
secondWord, COUNT(*) AS secondWordCount
FROM tripadvisor_review_sample_without_reviewtext
# Title has at least 2 words if it contains at least 1 white space
WHERE LENGTH(cleaned_title) - LENGTH(REPLACE(cleaned_title, ' ', '')) >= 1
GROUP BY secondWord
ORDER BY secondWordCount DESC
LIMIT 1;
```

 # 3. Most Popular Second Word for Titles Starting with "bad" or "terrible"

```
SELECT SUBSTRING_INDEX(SUBSTRING_INDEX(cleaned_title, ' ', 2), ' ', -1) AS
secondWord, COUNT(*) AS secondWordCount
FROM tripadvisor_review_sample_without_reviewtext
# Titles with first word as bad or terrible
WHERE (cleaned_title LIKE 'bad %' OR cleaned_title LIKE 'terrible %')
GROUP BY secondWord
ORDER BY secondWordCount DESC
LIMIT 1;


# Drop the cleaned_title column from the table
# so queries above can be rerun without error
ALTER TABLE tripadvisor_review_sample_without_reviewtext DROP COLUMN
cleaned_title;
```

The result query table of Question 8.1

firstWord	firstWordCount
great	2 144

The result query table of Question 8.2

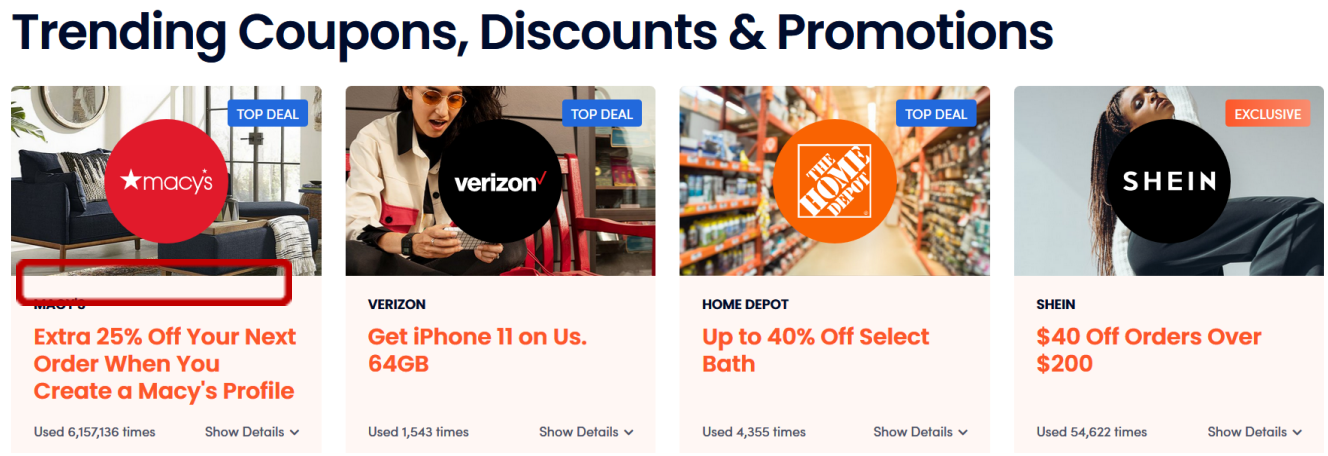
secondWord	secondWordCount
hotel	1 420

The result query table of Question 8.3

secondWord	secondWordCount
experience	7

9. The following screenshot shows the sale volume of a coupon site (https://www.savings.com/). Assuming that you have collected the sale volume information of each coupon on this website **on a daily basis**. For instance, the **total sale volume** for the Coupon "Macy's" is 6,157,136 at the data collection date.

   You want to calculate the daily sale volume for each coupon of this website (maybe because this website is the competitor of your company) for further analysis. How can you do that task using MySQL? **(5 points)**



   Based on the data of the **total sale volume** for each coupon (like **Table A)**. Please calculate **the daily sale volume**, as shown in **Table B**.

   Explanation: For instance, the **daily** sale volume for CouponA on '2021-09-05' is 25, because the **total** sale volume for CouponA is 37 on '2021-09-**05'** and 12 on '2021-09-**04'.** The difference in total sale volumes between the two dates is the daily sale volume, which is 37 – 12 = 25.

Coupon_ID	Sale_date	Total_sale_volume
CouponA	2021-09-03	0
CouponA	2021-09-04	12
CouponA	2021-09-05	37
CouponA	2021-09-06	40
CouponB	2021-08-14	43
CouponB	2021-08-15	75
CouponB	2021-08-13	20
CouponC	2021-09-04	16
CouponC	2021-09-05	38
CouponC	2021-09-06	77
CouponC	2021-09-07	97
CouponC	2021-09-03	2

Row Data: Table A

Coupon_ID	Sale_date	Total_sale_volume	daily_sale_volume
CouponA	2021-09-03	0	0
CouponA	2021-09-04	12	12
CouponA	2021-09-05	37	25
CouponA	2021-09-06	40	3
CouponB	2021-08-13	20	20
CouponB	2021-08-14	43	23
CouponB	2021-08-15	75	32
CouponC	2021-09-03	2	2
CouponC	2021-09-04	16	14
CouponC	2021-09-05	38	22
CouponC	2021-09-06	77	39
CouponC	2021-09-07	97	20

Result: Table B

Requirement:

1. Please write code to generate the results shown in Table B – the code will be the answer.
2. You can download Table A from Mycourse (coupon_sale_volume.sql)

The MySQL code that generates the result **(5 points)**:

```
SELECT
    table_A.Coupon_ID,
    table_A.Sale_date,
    table_A.Total_sale_volume,
    table_A.Total_sale_volume - IFNULL(table_B.Total_sale_volume, 0) AS
daily_sale_volume
FROM coupon_sale_volume AS table_A
LEFT JOIN
    coupon_sale_volume AS table_B ON table_A.Coupon_ID = table_B.Coupon_ID
    AND table_B.Sale_date = DATE_ADD(table_A.Sale_date, INTERVAL -1 DAY)
ORDER BY
    table_A.Coupon_ID,
    table_A.Sale_date;
```

The result query table of Question 9

Coupon_ID	Sale_date	Total_sale_volume	daily_sale_volume
CouponA	2021-09-03	0	0
CouponA	2021-09-04	12	12
CouponA	2021-09-05	37	25
CouponA	2021-09-06	40	3
CouponB	2021-08-13	20	20
CouponB	2021-08-14	43	23
CouponB	2021-08-15	75	32
CouponC	2021-09-03	2	2
CouponC	2021-09-04	16	14
CouponC	2021-09-05	38	22
CouponC	2021-09-06	77	39
CouponC	2021-09-07	97	20

10. Assuming you are now a business analyst offering consultant service to the tourism minister of Finland **(a total of 5 points)**. The minister wants to know how tourists travel within Finland between different cities.

    - Specifically, **<u>for those tourists whose first visit</u>** to Finland is Helsinki city [i.e., the first review in the database (in terms of review_date) is about a hotel in **Helsinki_Uusimaa**], which city would most likely be visited by those tourists in the future?

        i) **<u>For those tourists whose first visit</u>** to Finland is to Helsinki city [i.e. first review in the database (in terms of review_date) is about a hotel in **Helsinki_Uusimaa]**, they also visited "**Rovaniemi_Lapland"** for **184 times <u>in the future</u>** (2 points).

        ii) **<u>For those tourists whose first visit</u>** to Finland is to Helsinki city [i.e. first review in the database (in terms of review_date) is about a hotel in **Helsinki_Uusimaa]**, **12** of **them** also visited **both** "**Saariselka_Lapland"** and "**Rovaniemi_Lapland**" cities **<u>in the future</u>** (3 points).

    Note:
    - Please read the assignment questions carefully!
    - Please download the "assignment_tourist_Finland.sql" dataset from MyCourse.
        - It would be good to remove previous review-related tables before you import the database file so that you won't mix the current assignment tables with previous review-related tables.
        - The sql file contains two tables of "hotel" [431 records] and "review2" [56,709 records]. Column 'Id' of the table 'hotel' is connected to the column 'hotel_id' of the table 'review2'
    - It may happen that a traveler wrote multiple reviews about hotels in different cities as his/her **first** reviews (on the same but earliest review date). If one of these 'first' reviews includes Helsinki city, the traveler should be counted as a tourist whose first visit to Finland is Helsinki city. If one of these 'first' reviews includes 'Rovaniemi_Lapland' city, that trip is **NOT** considered visiting 'Rovaniemi_Lapland' city. "**<u>In the future</u>**" means future trips after these first reviews that were written on the same but earliest review day - *I know this does not sound so logical, but it is good to increase the difficulty of the assignment question for training your mind).*
    - Assume that different values in the column "city" of the "hotel" table represent different cities. For instance, "**Saariselka**" and "**Rovaniemi**" are two different cities.

    The MySQL code that generates the results **(2+3 points)***:*

 # Part (i): Count the number of times tourists who first visited Helsinki also visited Rovaniemi_Lapland in the future

```
SELECT COUNT(*) AS visitsToRovaniemi
FROM (
  # Subquery for find the user IDs where their first review_date is in Helsinki_Uusimaa
  SELECT r2.user_id
  FROM review2 r2
  JOIN hotel h ON r2.hotel_id = h.id
  WHERE h.city = 'Helsinki_Uusimaa'
  GROUP BY r2.user_id
  HAVING MIN(r2.review_date) = ANY (
    # This subsubquery means if a user's earliest review date in Helsinki is the same as their earliest review date in general, then their first recorded stay (according to the reviews) was in Helsinki
    SELECT MIN(r2_sub.review_date)
    FROM review2 r2_sub
    WHERE r2_sub.user_id = r2.user_id
  )
) AS firstVisitorsHelsinki
JOIN review2 r2_future ON r2_future.user_id = firstVisitorsHelsinki.user_id
JOIN hotel h_future ON r2_future.hotel_id = h_future.id
WHERE h_future.city = 'Rovaniemi_Lapland'
AND r2_future.review_date > (
  # Visits to Rovaniemi on the first review date in general are not counted
  SELECT MIN(r2_sub.review_date)
  FROM review2 r2_sub
  WHERE r2_sub.user_id = firstVisitorsHelsinki.user_id
);
```

```
# Part (ii): Count the number of tourists who first visited Helsinki also
# visited Rovaniemi_Lapland or Saariselka_Lapland in the future

SELECT COUNT(DISTINCT user_id) AS NumberOfTouristsVisitBothCities
FROM (
  SELECT r2.user_id
  FROM review2 r2
  JOIN hotel h ON r2.hotel_id = h.id
  WHERE h.city = 'Helsinki_Uusimaa'
  GROUP BY r2.user_id
  HAVING MIN(r2.review_date) = ANY (
    SELECT MIN(r2_sub.review_date)
    FROM review2 r2_sub
    WHERE r2_sub.user_id = r2.user_id
  )
) AS firstVisitorsHelsinki
WHERE EXISTS (
  SELECT 1
  FROM review2 r2
  JOIN hotel h ON r2.hotel_id = h.id
  WHERE r2.user_id = firstVisitorsHelsinki.user_id
  AND h.city = 'Saariselka_Lapland'
  AND r2.review_date > (
    SELECT MIN(r2_sub.review_date)
    FROM review2 r2_sub
    WHERE r2_sub.user_id = firstVisitorsHelsinki.user_id
  )
)
```

```
AND EXISTS (
  SELECT 1
  FROM review2 r2
  JOIN hotel h ON r2.hotel_id = h.id
  WHERE r2.user_id = firstVisitorsHelsinki.user_id
  AND h.city = 'Rovaniemi_Lapland'
  AND r2.review_date > (
    SELECT MIN(r2_sub.review_date)
    FROM review2 r2_sub
    WHERE r2_sub.user_id = firstVisitorsHelsinki.user_id
  )
);
```

The result query table of Question 10. i

visitsToRovaniemi
184

The result query table of Question 10. ii

NumberOfTouristsVisitBothCities
12