**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



1. 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



1. 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



1. 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**

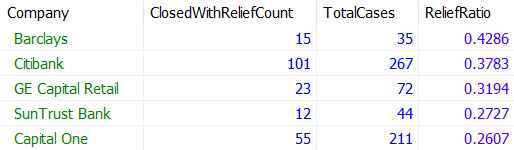
**GROUP BY Company**

**HAVING COUNT(\*) > 30**

**ORDER BY ReliefRatio DESC**

**LIMIT 5;**

The result query table of Question 4



1. 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



1. 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**

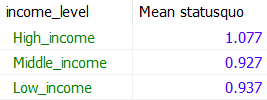
**END;**

**# 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



1. 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**

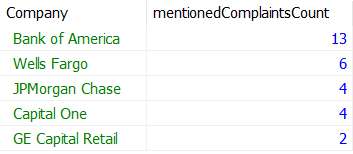
**AND DATEDIFF(Data\_sent\_to\_company, Data\_received) > 5**

**GROUP BY Company**

**ORDER BY mentionedComplaintsCount DESC**

**LIMIT 5;**

The result query table of Question 7



1. In the data “tripadvisor\_review\_sample\_without\_reviewtext”, based on the review titles that have **at least two words** [a total of 5 points]:
2. 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]

1. 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]

1. 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



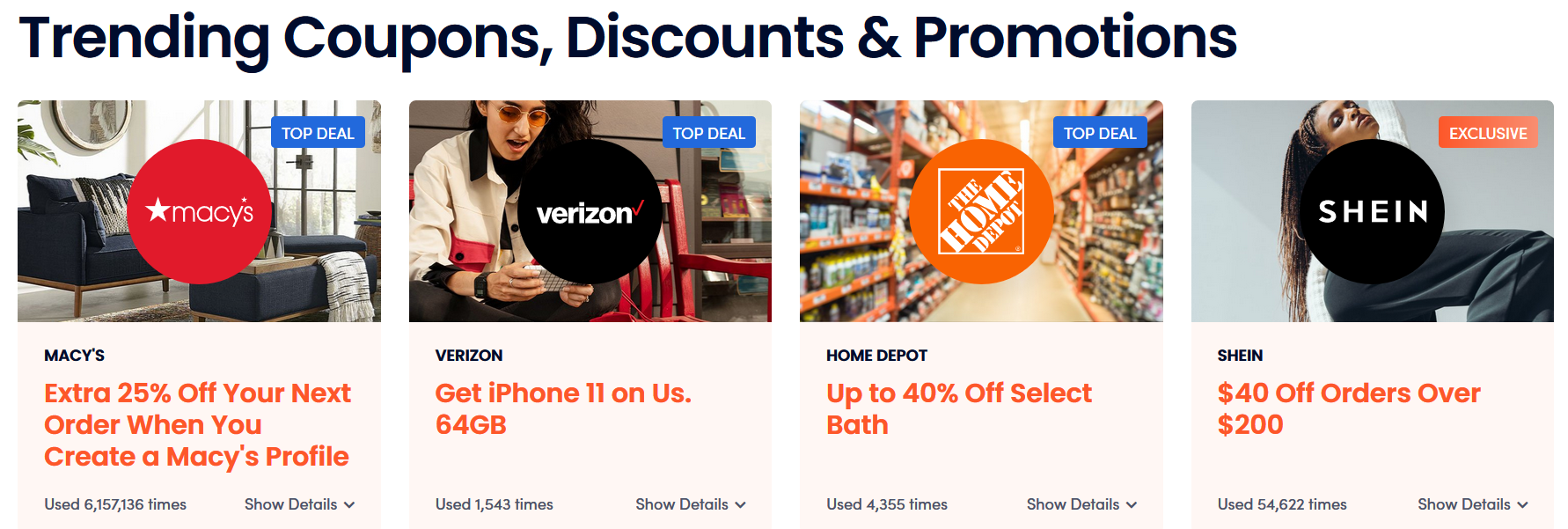
The result query table of Question 8.2



The result query table of Question 8.3



1. 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.

|  |  |
| --- | --- |
| **Row Data: Table A** | **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**

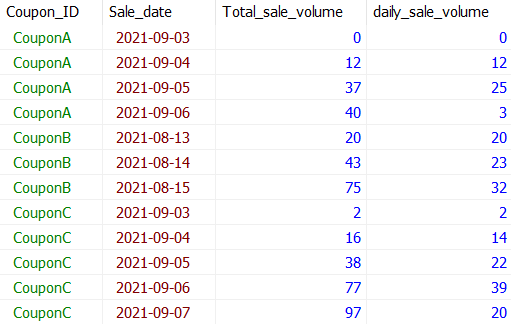
**AND table\_B.Sale\_date = DATE\_ADD(table\_A.Sale\_date, INTERVAL -1 DAY)**

**ORDER BY**

**table\_A.Coupon\_ID,**

**table\_A.Sale\_date;**

The result query table of Question 9

****

1. 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, **for those tourists whose** **first visit** 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?

1. **For those tourists whose** **first visit** 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 in the future** (2 points).
2. **For those tourists whose** **first visit** 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 **in the future** (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. “**In the future**” 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



The result query table of Question 10. ii

****