

SCHOOL OF ENGINEERING AND TECHNOLOGY




COURSEWORK FOR

BSC (HONS) IN COMPUTER SCIENCE

YEAR 1; ACADEMIC SESSION APRIL 2022

SEG1201: DATABASE FUNDAMENTALS

DEADLINE: Week 14 – Monday, 11th July 2022, 8:30 am

No.	Student Full Name	Student ID	Signature
1	Aisha Sofia Binti Najidi	20065231	
2	Emily Teng Jie Qi	20054607	
3	Justin Phang Sheng Xuan	20066502	




Instructions

1. This final assessment contributes 50% to your final grade.
2. This four-member group assignment is primarily for Course Learning Outcome 2 - Implement a database design group project using appropriate tools such as Oracle SQL.
3. Each member of the group is given a specific role to play and is required to present his/her part of the work.
4. Note that if your group has less than four members, each member in the group will need to take up some extra tasks.

IMPORTANT

The University requires students to adhere to submission deadlines for any form of assessment. Penalties are applied in relation to unauthorized late submission of work.

Academic Honesty Acknowledgement

“We Aisha Sofia Binti Najidi, Emily Teng Jie Qi, and Justin Phang Sheng Xuan, verify that this paper contains entirely our own work. We have not consulted with any outside person or materials other than what was specified (an interviewee, for example) in the assignment or the syllabus requirements. Further, we have not copied or inadvertently copied ideas, sentences, or paragraphs from another student. We realize the penalties (refer to page 16, 5.5, Appendix 2, page 44 of the student handbook diploma and undergraduate programme) for any kind of copying or collaboration on any assignment.”,   

(15th June 2022)

Contents

1.0 Construct a case scenario	3
1.1 Business Rules	4
2.0 Design a database.....	5
2.1 FLOWCHART.....	5
2.2 RDM	6
2.3 Attribute Description Table	7
2.4 ERD.....	9
3.0 Implement a database.....	10
3.1 User Views.....	10
3.2 Indexing	11
4.0 Query a database	12
5.0 Teamwork and Presentation.....	15
5.1 Members and Roles.....	15
5.2 Project Plan	15
References.....	16

1.0 Construct a case scenario

Launching in May 2018, Grab (2018) food delivery service, GrabFood, was created to connect people to food businesses. It allows customers to order food online without any minimum order requirements while having a GrabFood delivery partner to pick up the order from the restaurant to bring it to you. However, due to the pandemic in 2020, there was an increase of complaints from the customers. This is because restaurants were taking in too many orders at once, furthermore, delivery drivers were handling too many orders at once. To prevent this problem from exacerbating further, Grab has hired new database designers to control the influx of consumer data while ensuring consumer data remains consistent and up to date.

Grab has a relational database to keep track of orders and deliveries. Grab acts as a delivery service for many stores. Each customer may only have one Grab account. Each customer must register a Grab account with their Identification Card or Passport. Each store may only have one menu, and customers must order from that menu. Employees must be over 18 years old and have a up to date drivers' licence to make deliveries.

The customer shall not be entitled to cancel an order after it is confirmed, if you cancel an order after it has been confirmed, the customer remains liable to pay the applicable fees for the order in full regardless of whether the food has been prepared or not. If there are no employees available for delivery, customers order will be cancelled and fully reimbursed. After each order is made, customers will be notified of an estimated time of arrival for each delivery. To ensure customer receives their orders on time, delivery employees of Grab can only take up to 3 delivery orders at a time. Moreover, the stores can only take up to 20 orders at a time to ensure the food is ready for delivery on time. Customers may place orders from different stores in multiple orders.

Grab will save customer bank details after first time entering unless customer opts out, to ensure easy ordering process moving forward. To ensure that customers receive their orders successfully, payments made will be on-hold until the delivery is made, if delivery is late, customer will be compensated with a voucher for their next order.

Each voucher will be given as a code, and each code is valid within one month of redemption. Each customer can only order from stores within a 10km radius of their current location. Each customer will be notified of the tax prices in the payment screen prior to making their payment. Each customer can have different payment methods. Every alcohol related food or beverage items must be made by a customer over 21 years old. In the event an alcohol related food or beverage is ordered and consumed by a customer under 21 years of age, Grab is not liable for any illegal acts.

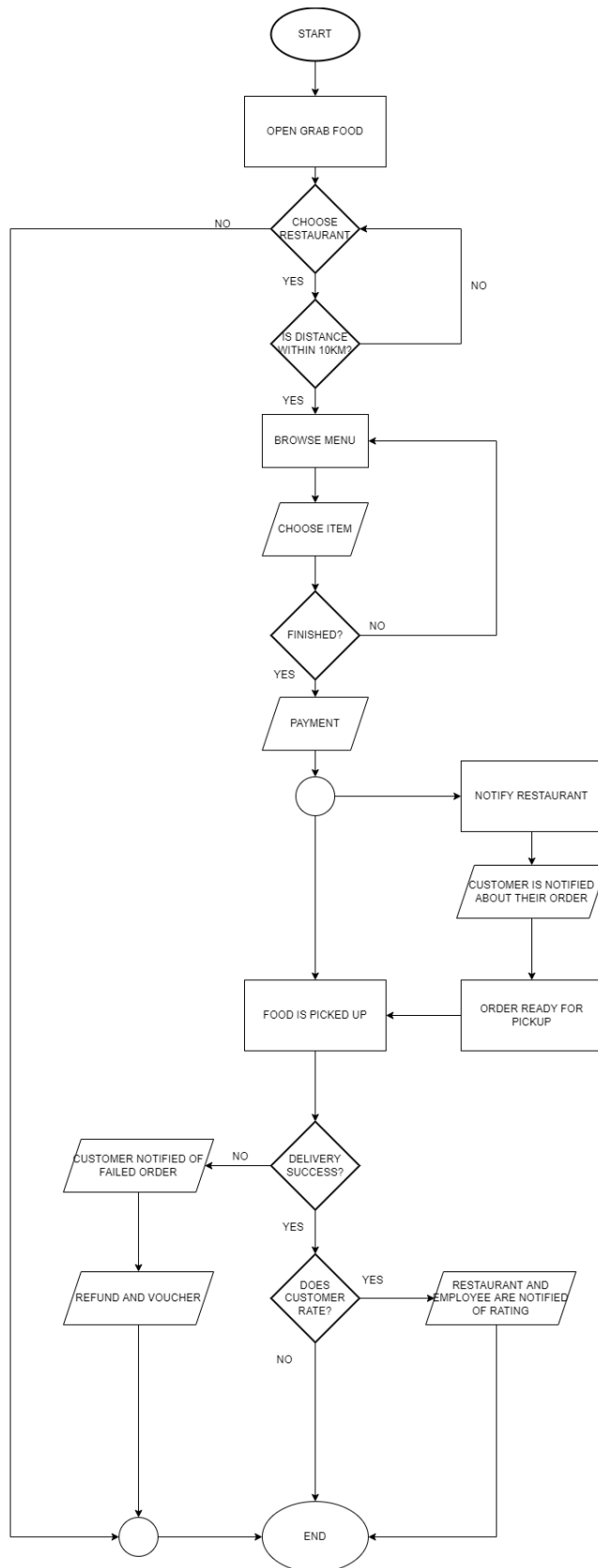
Each customer can rate the store and delivery person via the application after order has been delivered. GrabFood will halt services to stores that have a rating below 2.0 stars. Grab will temporarily halt deliveries made by employees with a rating below 1.5 stars. The tables are as follows Customer, Payment, Order, Order_line, Employee, Delivery, Store, Menu, Rating and Voucher.

1.1 Business Rules

1. Each customer can order multiple times, but a different OrderID will be provided to the system.
2. A unique OrderID will be provided for each order made by a customer.
3. Each store can take multiple orders
4. Each order has one or more order line.
5. Each Delivery is delivered by one employee.
6. One employee can make many deliveries.
7. Each delivery employee can take at most 3 orders at one time.
8. Each restaurant can take 20 orders at a time.
9. Each customer can order from one or more stores.
10. Each store only has one menu.
11. Each customer can only give one rating once per order.
12. Each delivery employee and store can be rated by many customers.
13. Each late delivery can only be compensated with one voucher.
14. Each voucher can only be redeemed once.

2.0 Design a database

2.1 FLOWCHART



2.2 RDM

Customer

(**Customer ID**, First_Name, Last_Name, DOB, CLocation, Phone_Number)

Payment

(**Customer ID**, **Order ID**, Total)

Orders

(**Order ID**, Customer_ID, Time_Order_Placed)

Order_line

(**Order ID**, **Store ID**, Product, Product_ID, Quantity, Price)

Employee

(**Employee ID**, EMP_First_Name, EMP_Last_Name, EMP_Phone_Number, Managed_By, Position, Salary)

Delivery

(**Delivery ID**, **Employee ID**, **Order ID**, ETA, Del_Price)

Store

(**Store ID**, Store_Name, Location, Store_Location, AgeRes)

Rating

(**Customer ID**, **Store ID**, **Employee ID**, Store_Rating_Amount, Emp_Rating_Amount)

Voucher

(**Voucher Code**, Customer_ID, Exp_Date, Redeem_Date)

Italic = Foreign Key

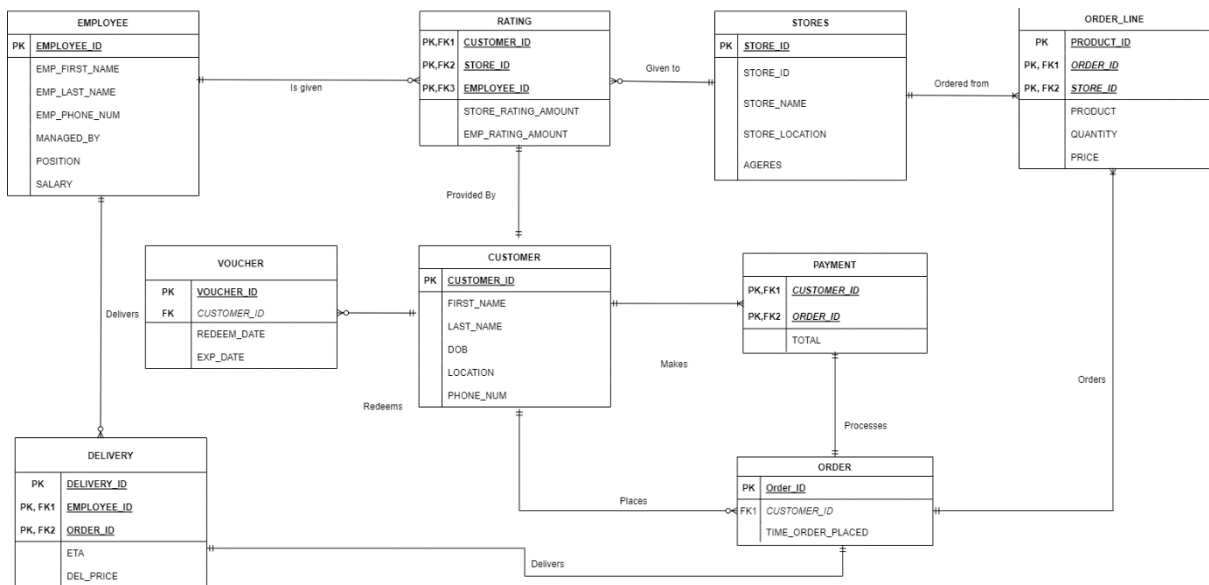
Bold & Underline = Primary Key

2.3 Attribute Description Table

Attributes	Attribute Description	Data type	Justification
Customer_ID	ID of the customer	INT	Customer ID is in integers, so INT is used.
First_Name	First name of the customer	VARCHAR2(30)	Allows for customer name to be 30 letters long.
Last_Name	Last name of the customer	VARCHAR2(30)	Allows for customer name to be 30 letters long.
DOB	Date of Birth	DATE	DOB is in DATE format.
Location	Location of the customer	VARCHAR2(1000)	Allows for both numbers and letters in address, as well as allows long addresses.
Phone_Number	Customer phone number	VARCHAR2(15)	Allows multiple ways to write phone numbers. (E.g.: 60xxxxxxxx, +60xxxxxxxx, 01X-XXX-XXXX, 01XXXXXXXX).
Order_ID	ID of the order	INT	The ORDER_IDs are numeric values (E.g.: 100000).
Total	Total price of the order	NUMBER(10,2)	The total price of the order is in numeric with decimal values.
Time_Order_Placed	The time when the order was placed	TIMESTAMP(2)	TIMESTAMP is used because it not only provides us with date, but it also provides us the time in HH:MM:SS.
Store_ID	ID of store	INT	Store_ID is in integers, so INT is used.
Product	Product information	VARCHAR2(100)	Allows for long name products to be inserted.
Quantity	Product quantity	NUMBER(3)	The quantity of the order can be a maximum of 999.
Price	Price per product	NUMBER (10,2)	The total price of the order is in numeric with decimal values.
Employee_ID	ID of the employee	NVARCHAR2(5)	To futureproof grab, NVARCHAR2 is used to allow Unicode characters to be used. This allows the ID to be kept short while still being unique.

EMP_First_Name	Employee first name	VARCHAR2(30)	Allows for employee name to be 30 letters long.
EMP_Last_Name	Employee last name	VARCHAR2(30)	Allows for employee name to be 30 letters long.
EMP_Phone_Number	Employee phone number	VARCHAR2(15)	Allows multiple ways to write phone numbers. (Eg: 60xxxxxxxxx, +60xxxxxxxxx,01X-XXX-XXXX, 01XXXXXXXX).
Position	Employee position	VARCHAR2(20)	Allows some positions/titles to have only capitalized alphabets (exp: CEO)
Salary	Employee Salary	NUMBER(10,2)	The total salary of the employee can be in decimal format.
Managed By	Employee Managed by who	NVARCHAR2(5)	It must be in the same data type as the employee ID.
Emp_Rating_Amount	Employee rating amount	NUMBER(2,1)	The employee rating amount can be in decimal (Eg: 5.2).
Delivery_ID	Delivery ID	INT	Delivery_ID is in numeric values only therefore INT is used.
ETA	Estimated time of arrival	TIMESTAMP(2)	TIMESTAMP is used because it not only provides us with date but it also provides us the time in HH:MM:SS.
Delivery_Price	Price of the delivery	NUMBER(10,2)	The total price of the order is in numeric with decimal values.
STORE_Location	Location of the store	VARCHAR2(1000)	The location of the store can be in both alphabetical and numeric values.
Store_Rating_Amount	Store rating amount	NUMBER(2,1)	The store rating amount can be in decimal (Eg: 5.2).
Voucher_Code	Voucher code	CHAR(5)	The voucher codes are in alphabetic and numeric values (Eg: V0001).
Exp_Date	Expiry date	DATE	The Exp_Date attribute is in a date format.
Redeem_Date	Redeem date	DATE	The Redeem_Date attribute is in a date format.

2.4 ERD



3.0 Implement a database

3.1 User Views

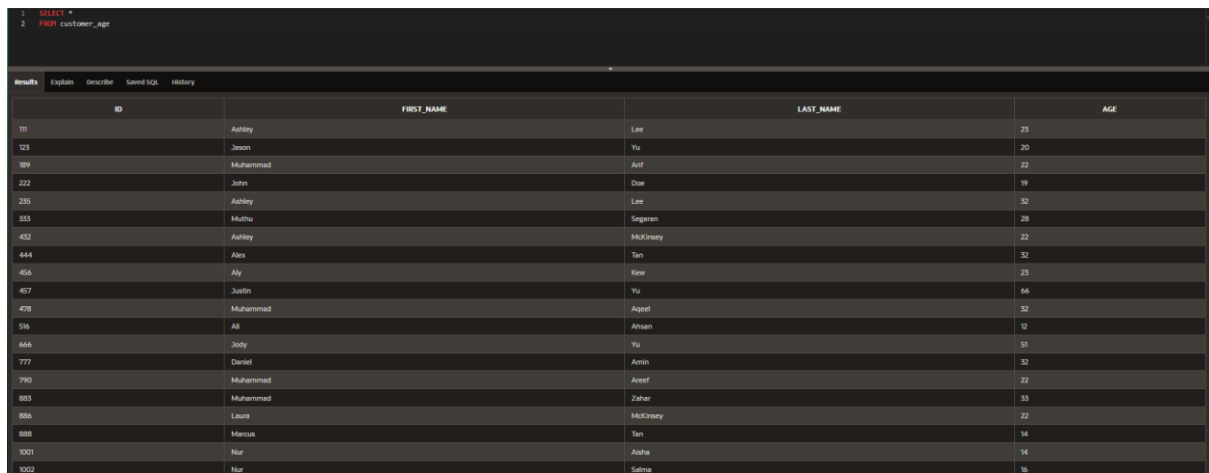
Create View For Customer Age

CREATE VIEW customer_age AS

SELECT CUSTOMER_ID AS ID, FIRST_NAME, LAST_NAME, ((EXTRACT (YEAR
FROM CURRENT_DATE)) - (EXTRACT (YEAR FROM DOB))) AS Age

FROM Customer

Results:



ID	FIRST_NAME	LAST_NAME	AGE
111	Ashley	Lee	23
125	Jason	Yu	20
189	Muhammad	Araf	22
222	John	Doe	19
235	Ashley	Lee	32
333	Muthu	Segaran	28
432	Ashley	McKinney	22
444	Alex	Tan	32
456	Aly	Kew	23
497	Justin	Yu	46
478	Muhammad	Aggel	32
136	Ab	Ahsan	12
466	Judy	Yu	31
777	Daniel	Amin	32
790	Muhammad	Arief	22
883	Muhammad	Zahar	33
886	Laura	McKinney	22
888	Marcus	Tan	14
1001	Nur	Aisha	14
1002	Nur	Salma	16

Create View For Employee Average Rating

CREATE VIEW employee_average_ratings AS


SELECT Employee.EMPLOYEE_ID AS Employee,
ROUND((AVG(RATING.EMP_RATING_AMOUNT)),1) AS Rating_Amount

FROM Employee, Rating

WHERE EMPLOYEE.EMPLOYEE_ID = RATING.EMPLOYEE_ID

GROUP BY EMPLOYEE.EMPLOYEE_ID

Results:



EMPLOYEE	RATING_AMOUNT
A48	4.5
A10	4.5
A17	3.3
A49	5
A01	3.7
A02	3
A12	3.5

7 rows returned in 0.02 seconds [Download](#)

Create View For Store Average Rating

```
CREATE VIEW store_average_ratings AS

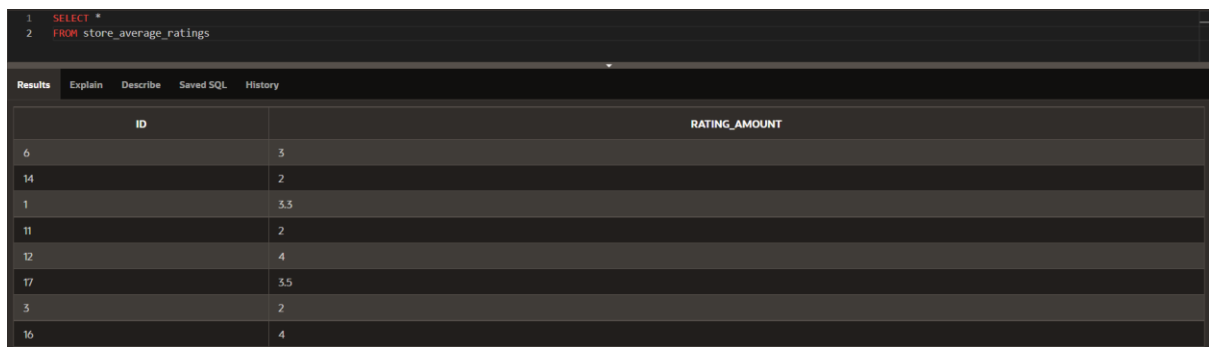
SELECT STORES.STORE_ID AS ID, ROUND((AVG
(RATING.STORE_RATING_AMOUNT)),1) AS Rating_Amount

FROM Stores, Rating

WHERE Stores.Store_ID = Rating.Store_ID

GROUP BY Stores.Store_ID
```

Results:



ID	RATING_AMOUNT
6	3
14	2
1	3.3
11	2
12	4
17	3.5
3	2
16	4

3.2 Indexing

CREATE INDEX CUSTOMER_AGE on CUSTOMER (CUSTOMER_ID, FIRST_NAME, LAST_NAME, DOB);

Index is required for this as customer details, especially date of birth, to calculate the age of the customer to determine the age restriction. Furthermore, customer id is frequently used throughout all the tables, such as orders and delivery table.

CREATE INDEX STORE_RATING on STORES (STORE_ID, STORE_NAME, AGERES);

Index is required to ensure that stores must have an average rating above 1.5 to be applicable for orders

CREATE INDEX EMPLOYEE_RATING on EMPLOYEE (EMPLOYEE_ID, EMP_FIRST_NAME, EMP_LAST_NAME);

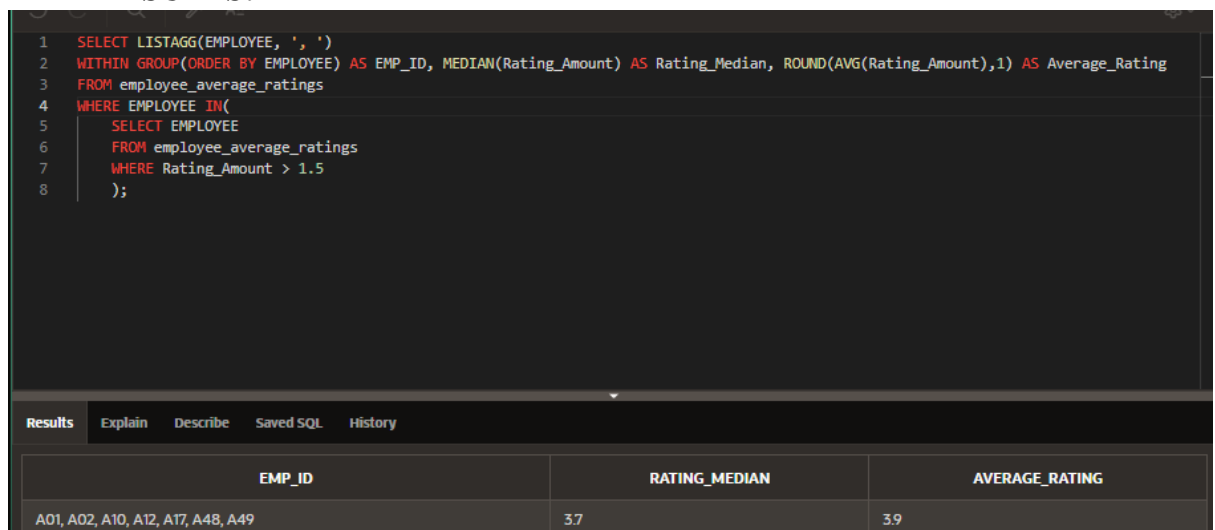
Index is required to monitor employee ratings.

4.0 Query a database

- a. Write a user query with one sub-query, and two new aggregate functions which have not been taught in this subject. Please explore the functions from the ORACLE website
- Grab wishes to analyse employee's rating during the pandemic for their year-end report, so the statistics team from grab food wishes to see the delivery employees and the discrepancy between the median rating, and average rating.
 - CODE:**

```
SELECT LISTAGG(EMPLOYEE, ', ')
  WITHIN GROUP(ORDER BY EMPLOYEE) AS EMP_ID,
  MEDIAN(Rating_Amount) AS Rating_Median,
  ROUND(AVG(Rating_Amount),1) AS Average_Rating
FROM employee_average_ratings
WHERE EMPLOYEE IN(
  SELECT EMPLOYEE
  FROM employee_average_ratings
  WHERE Rating_Amount > 1.5
);
```

- RESULTS:**



The screenshot shows an SQL IDE with a query editor and a results pane. The query editor contains the following SQL code:

```
1 SELECT LISTAGG(EMPLOYEE, ', ')
2   WITHIN GROUP(ORDER BY EMPLOYEE) AS EMP_ID, MEDIAN(Rating_Amount) AS Rating_Median, ROUND(AVG(Rating_Amount),1) AS Average_Rating
3 FROM employee_average_ratings
4 WHERE EMPLOYEE IN(
5   SELECT EMPLOYEE
6   FROM employee_average_ratings
7   WHERE Rating_Amount > 1.5
8 );
```

The results pane shows the following table:

EMP_ID	RATING_MEDIAN	AVERAGE_RATING
A01, A02, A10, A12, A17, A48, A49	3.7	3.9

- b. Write a user query with a 4-table join, 2 user conditions (note: and user conditions are not the same) and a GROUP BY clause and HAVING subclause. Ensure that your scenario reflects the reason for such join, and not join those tables for the sake of joining.

- Grab wishes to find the most active employee in the Selangor area to give them a reward. As such, GrabFood has given the database designers a task to find a list of the most active employees in terms of delivery in Selangor.
- **CODE:**

```
SELECT Employee.EMPLOYEE_ID,
Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME,
Customer.CLOCATION,
COUNT(DELIVERY.EMPLOYEE_ID) AS
NUMBER_OF_COUNT
FROM Employee
JOIN Delivery ON Employee.EMPLOYEE_ID =
Delivery.EMPLOYEE_ID
JOIN Orders ON Delivery.ORDER_ID = Orders.ORDER_ID
JOIN Customer ON Orders.Customer_ID =
Customer.CUSTOMER_ID
WHERE Customer.CLOCATION LIKE '%Selangor%'
GROUP BY Employee.EMPLOYEE_ID,
Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME,
Customer.CLOCATION
HAVING COUNT(Orders.ORDER_ID) > 2
```

- **RESULTS:**



The screenshot shows a SQL query execution interface. The query is displayed in a text area, and the results are shown in a table below. The table has five columns: EMPLOYEE_ID, EMP_FIRST_NAME, EMP_LAST_NAME, CLOCATION, and NUMBER_OF_COUNT. The results show one row for employee AD1, with first name Ali, last name Darish, location 34, Persiaran Subang, Taman Indah Subang Lrt, 47619 Subang Jaya, Selangor, Malaysia, and a count of 3.

EMPLOYEE_ID	EMP_FIRST_NAME	EMP_LAST_NAME	CLOCATION	NUMBER_OF_COUNT
AD1	Ali	Darish	34, Persiaran Subang, Taman Indah Subang Lrt, 47619 Subang Jaya, Selangor, Malaysia	3

- c. Write a user query with a right outer join and 3 user conditions. One of the conditions uses the LIKE keyword.

- Grab wishes to find employees that have not done a delivery, or any employees that have recently (In the past 3 months) not done any deliveries in order to take disciplinary actions towards them.
- **CODE:**

```
SELECT Delivery.DELIVERY_ID, Employee.EMP_LAST_NAME,
Employee.EMP_FIRST_NAME FROM Delivery RIGHT JOIN Employee
ON Delivery.EMPLOYEE_ID = Employee.EMPLOYEE_ID WHERE
Employee.POSITION LIKE 'Delivery' AND DELIVERY.DELIVERY_ID
IS NULL OR DELIVERY.ETA < '01-APR-2021' ORDER BY
Delivery.DELIVERY_ID;
```

- **RESULTS:**

```

1 SELECT Delivery.DELIVERY_ID, Employee.EMP_LAST_NAME, Employee.EMP_FIRST_NAME
2 FROM Delivery
3 RIGHT JOIN Employee
4 ON Delivery.EMPLOYEE_ID = Employee.EMPLOYEE_ID
5 WHERE Employee.POSITION LIKE 'Delivery' AND Delivery.DELIVERY_ID IS NULL OR Delivery.ETA < '01-APR-2021' ORDER BY Delivery.DELIVERY_ID;
6

```

DELIVERY_ID	EMP_LAST_NAME	EMP_FIRST_NAME
27	Ashraf	Alysa
32	Darsh	Elie
35	Raju	Muthu
38	Kim	Ahmad
-	Tari	Peter
-	Kim	Ala
-	Wong	Alysa
-	Kim	Larry

d. Write a user query using a unary join. Ensure that your scenario reflects the reason for such requirement. Display only the relevant records and fields.

- Grab wishes to find the managers of each employee, to easily find out which manager to contact in the situation where disciplinary action needs to be taken.

- **CODE:**

```

SELECT MANAGER.EMP_FIRST_NAME AS MANAGER,
LISTAGG(EMP.EMPLOYEE_ID, ', ' ) WITHIN GROUP(ORDER BY
EMP.EMPLOYEE_ID) AS Employees
FROM EMPLOYEE EMP, EMPLOYEE MANAGER
WHERE EMP.MANAGED_BY = MANAGER.EMPLOYEE_ID
GROUP BY MANAGER.EMP_FIRST_NAME

```

- **RESULTS:**

```



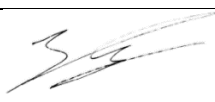
1 SELECT MANAGER.EMP_FIRST_NAME AS MANAGER, LISTAGG(EMP.EMPLOYEE_ID, ', ' ) WITHIN GROUP(ORDER BY EMP.EMPLOYEE_ID) AS Employees
2 FROM EMPLOYEE EMP, EMPLOYEE MANAGER
3 WHERE EMP.MANAGED_BY = MANAGER.EMPLOYEE_ID
4 GROUP BY MANAGER.EMP_FIRST_NAME
5

```

MANAGER	EMPLOYEES
Alysa	A02, A08, A10, A19, A26, A34, A39, A45, A50
Emily	A04, A06, A16, A22, A28, A35, A42, A46
Justin	A05, A18, A23, A29, A38, A43, A47

5.0 Teamwork and Presentation

5.1 Members and Roles

No.	Student Full Name	Student ID	Roles	Signature
1	Aisha Sofia Binti Najidi	20065231	PROJECT MANAGER, CEO	
2	Emily Teng Jie Qi	20054607	DATABASE DESIGNER	
3	Justin Phang Sheng Xuan	20066502	DATABASE TEAM LEADER	

5.2 Project Plan

TASK	ASSIGNMENT	START	END	PROGRESS
<ul style="list-style-type: none"> Constructing case scenario List Business requirements Create Business rules Record presentation 	All Members All Members Emily, Justin Aisha	10/06/22	15/06/22	100%
<ul style="list-style-type: none"> Flow Chart Diagram ERD RDM Progress Report 	Justin Justin Emily Aisha	18/06/22	22/06/22	100%
<ul style="list-style-type: none"> Create Script Create Views and Index Create Check constraints Populate database Progress Report 	Aisha Justin Emily Justin Aisha	25/06/22	29/06/22	100%
<ul style="list-style-type: none"> User queries SQL statements and results Project progress report 	Aisha Emily, Justin Aisha	02/07/22	06/07/22	100%
<ul style="list-style-type: none"> Presentation 	All Members	12/07/2022	12/07/2022	-

References

Grab. (2021, August 15). *Grab launches grabfood beta and unveils vision to be an everyday app*. Grab SG. Retrieved June 15, 2022, from <https://www.grab.com/sg/press/business/grab-launches-grabfood-beta-and-unveils-vision-to-be-an-everyday-app/>

Grab. (2019, October 25). *Terms of Service: Transport, Delivery and Logistics*. Grab MY. <https://www.grab.com/my/terms-policies/transport-delivery-logistics/>