**MIDTERM TEST**

**(1 tiếng rưỡi)**

# Task 1: Review general knowledge. Please find the correct answer to each question. (3p)

## **Which of the following operators is used to compare with pattern matching data from the database?**

1. LIKE
2. SELECT
3. IN

*Your answer:*

**A**

## **What is SQL execution order?**

## SELECT [TOP] 🡪 FROM 🡪 WHERE 🡪 JOIN 🡪 UNION 🡪 ORDER BY

## FROM 🡪 WHERE 🡪 JOIN 🡪 SELECT 🡪 GROUP BY 🡪 ORDER BY 🡪 LIMIT (TOP N)

## FROM 🡪 JOIN 🡪 WHERE 🡪 GROUP BY 🡪 SELECT 🡪 ORDER BY 🡪 LIMIT (TOP N)

*Your answer:*

**C**

## **Which method can help you to query data from multiple tables?**

1. Subquery
2. JOIN
3. UNION
4. All the above answers

*Your answer:*

**D**

## **Which JOIN query results in the below output table?**

## INNER JOIN

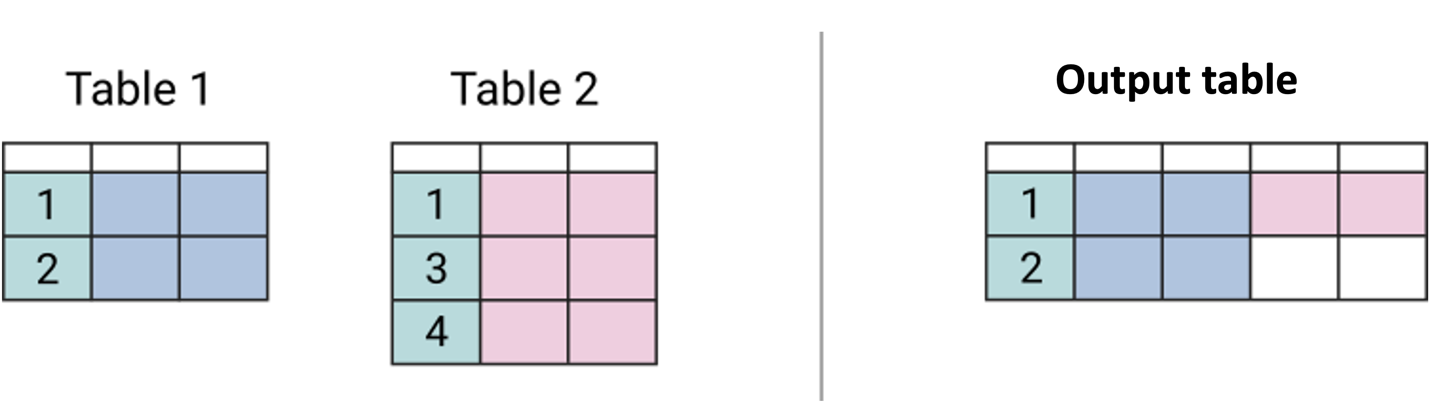
## LEFT JOIN

## RIGHT JOIN

## FULL JOIN

*Your answer:*

**B**



## **1.5. Which JOIN query results in the below output table?**

## INNER JOIN

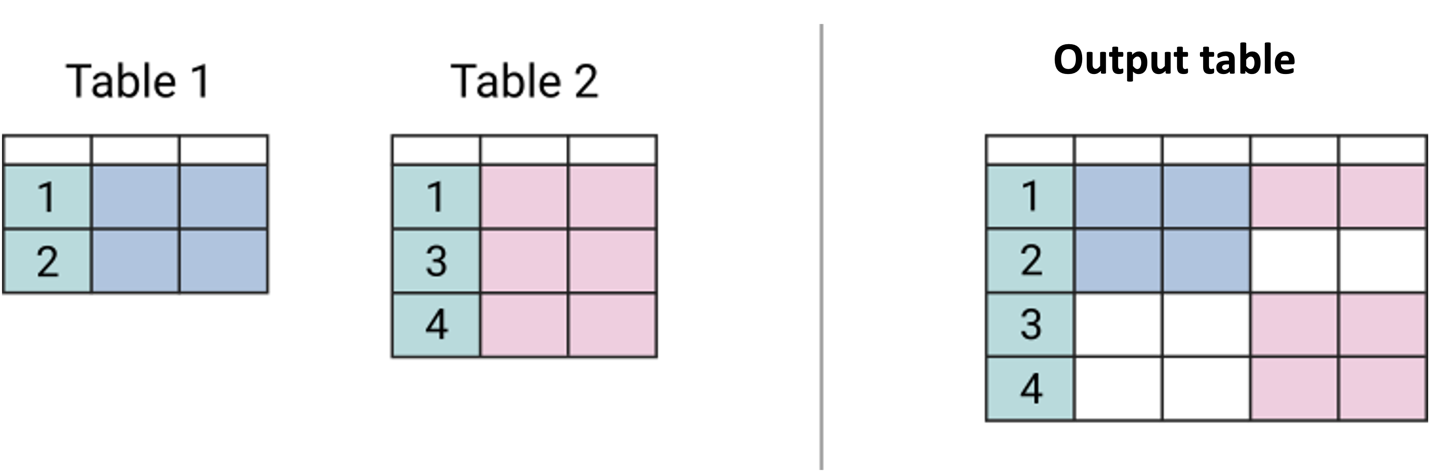
## LEFT JOIN

## RIGHT JOIN

## FULL JOIN

*Your answer:*

**D**

**

# Task 2: Write queries based on following requirements (3p)

## Retrieve a report that includes the following information*: customer\_id, transaction\_id, scenario\_id, transaction\_type, sub\_category, category*. These transactions must meet the following conditions: (1.5p)

# Were created in Feb 2019

# Transaction type is not empty (transaction type IS NOT NULL)

-- 2.1. Retrieve a report that includes the following information: customer\_id, transaction\_id, scenario\_id, transaction\_type, sub\_category, category. These transactions must meet the following conditions: (1.5p)

SELECT customer\_id

    , transaction\_id

    , fact\_19.scenario\_id

    , transaction\_type

    , sub\_category

    , category

FROM fact\_transaction\_2019 AS fact\_19

LEFT JOIN dim\_scenario AS scen

ON fact\_19.scenario\_id = scen.scenario\_id

WHERE MONTH(transaction\_time) = 2 -- Were created in Feb 2019

AND transaction\_type IS NOT NULL -- Transaction type not null

## From payment transaction history in February 2019. **Find the top 10% of failed transactions with the highest transaction value *(charged\_amount)***. (1.5p)

*( yêu cầu là tìm ra top 10% giao dịch thất bại của nhóm “Payment” (không thành công) có giá trị charged\_amount cao nhất trong tổng số giao dịch payment trong tháng 2/2019)*

-- 2.2. Find the top 10% of failed transactions with the highest transaction value (charged\_amount).

WITH joined\_table AS(

    SELECT customer\_id

    , transaction\_id

    , fact\_19.scenario\_id

    , transaction\_type

    , sub\_category

    , category

    , charged\_amount

FROM fact\_transaction\_2019 AS fact\_19

LEFT JOIN dim\_scenario AS scen

ON fact\_19.scenario\_id = scen.scenario\_id

LEFT JOIN dim\_status AS stat

ON fact\_19.status\_id = stat.status\_id

WHERE MONTH(transaction\_time) = 2 -- Were created in Feb 2019

    AND transaction\_type IS NOT NULL -- Transaction type not null

    AND NOT fact\_19.status\_id = 1

)

SELECT TOP 10 PERCENT \*

FROM joined\_table

ORDER BY charged\_amount DESC

# Task 3: Retrieve an overview report of customer’s payment behaviors

## Paytm has acquired a lot of customers. Retrieve a report that includes the following information: ***the number of transactions, the number of payment scenarios, the number of payment category and the total of charged amount of each customer***. Some additional conditions need to be met: (2p)

* Only show Top 10 highest customers by the total of charged amount
* Transactions created from 01/01/2020 to 31/03/2020
* Status description is successful
* Transaction type is payment

WITH joined\_table AS(

    SELECT customer\_id

    , count(transaction\_id) AS number\_trans

    , count(distinct(fact\_20.scenario\_id)) AS number\_scenarios

    , count(distinct(category)) AS number\_categories

    , sum(charged\_amount) AS total\_amount

FROM fact\_transaction\_2020 AS fact\_20

LEFT JOIN dim\_scenario AS scen

ON fact\_20.scenario\_id = scen.scenario\_id

WHERE MONTH(transaction\_time) < 4

AND status\_id = 1

AND transaction\_type = 'Payment'

GROUP BY customer\_id

)

SELECT TOP 10 \*

FROM joined\_table

ORDER BY total\_amount DESC

## Seeing the customer’s payment behaviors *(from 3.1 with all customers bỏ điều kiện TOP 10, lấy những điều kiện còn lại),* we want to segment customers into 2 groups: (1) whose total amount is greater than the average value of all customers, and (2) whose total is below the average value.

### Find the average of the total amount of all customers, then label who is **“greater\_than\_avg’**, who is **“lower\_than\_avg”.** (1p)

WITH joined\_table AS(

    SELECT customer\_id

    , count(transaction\_id) AS number\_trans

    , count(distinct(fact\_20.scenario\_id)) AS number\_scenarios

    , count(distinct(category)) AS number\_categories

    , CAST(sum(charged\_amount) AS FLOAT) AS total\_amount

FROM fact\_transaction\_2020 AS fact\_20

LEFT JOIN dim\_scenario AS scen

ON fact\_20.scenario\_id = scen.scenario\_id

WHERE MONTH(transaction\_time) < 4

AND status\_id = 1

AND transaction\_type = 'Payment'

GROUP BY customer\_id

)

, total\_table AS (

SELECT customer\_id

    , total\_amount

    , (SELECT avg(total\_amount) FROM joined\_table) AS avg\_amount

FROM joined\_table

)

SELECT \*

    , CASE WHEN total\_amount < avg\_amount THEN 'lower\_than\_avg'

    ELSE 'greater\_than\_avg'

    END group\_customers

FROM total\_table

### Calculate how many customers **spend more than the average value and how much they account for in total**? (1p)

WITH joined\_table AS(

    SELECT customer\_id

    , count(transaction\_id) AS number\_trans

    , count(distinct(fact\_20.scenario\_id)) AS number\_scenarios

    , count(distinct(category)) AS number\_categories

    , CAST(sum(charged\_amount) AS FLOAT) AS total\_amount

FROM fact\_transaction\_2020 AS fact\_20

LEFT JOIN dim\_scenario AS scen

ON fact\_20.scenario\_id = scen.scenario\_id

WHERE MONTH(transaction\_time) < 4

AND status\_id = 1

AND transaction\_type = 'Payment'

GROUP BY customer\_id

)

, total\_table AS (

SELECT customer\_id

    , total\_amount

    , (SELECT avg(total\_amount) FROM joined\_table) AS avg\_amount

FROM joined\_table

)

, avg\_table AS(

    SELECT \*

    , CASE WHEN total\_amount < avg\_amount THEN 'lower\_than\_avg'

    ELSE 'greater\_than\_avg'

    END group\_customers

FROM total\_table

)

SELECT group\_customers

    , count(group\_customers) AS number\_customers

    , (SELECT COUNT(group\_customers) FROM avg\_table) AS total\_customers

    , FORMAT ((count(group\_customers)\*1.0)/(SELECT COUNT(group\_customers) FROM avg\_table),'p') AS pct

FROM avg\_table

WHERE group\_customers = 'greater\_than\_avg'

GROUP BY group\_customers