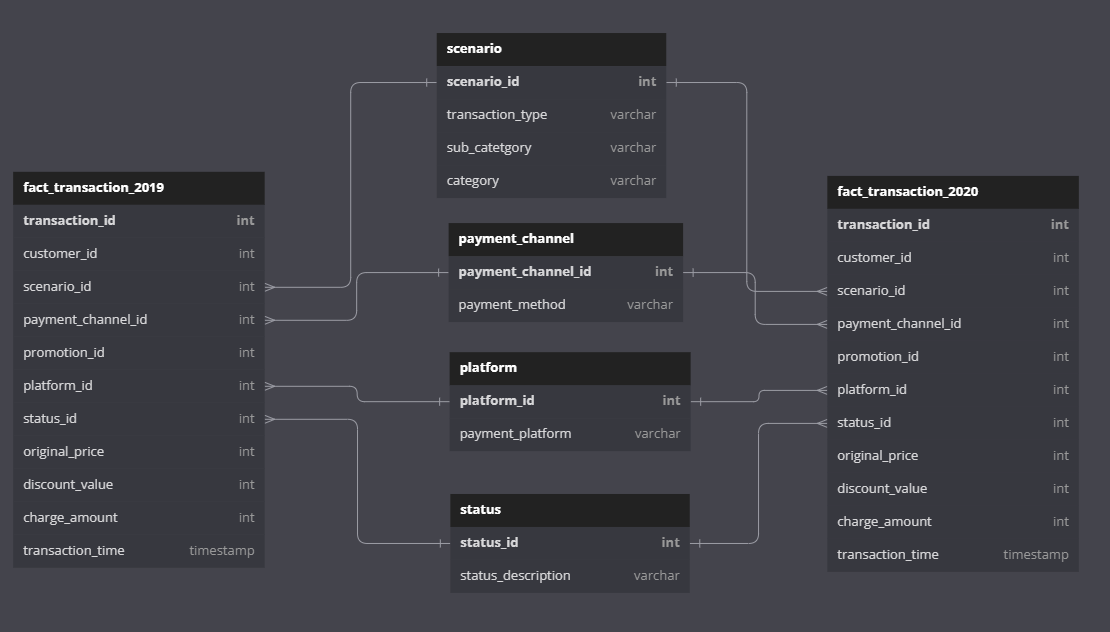
**SQL Project: Paytm**

**I.DataBase Diagram**

****

**II. Overview**

Paytm is an Indian multinational financial technology company.

It specialises in digital payment systems, e-commerce and financial services. Paytm wallet is a secure and RBI (Reserve Bank of India)-approved digital/mobile wallet that provides a myriad of financial features to fulfil every consumer’s payment needs. Paytm wallet can be topped up through UPI (Unified Payments Interface), internet banking, or credit/debit cards. Users can also transfer money from a Paytm wallet to the recipient's bank account or their own Paytm wallet.

Below is a small database of payment transactions from 2019 to 2020 of Paytm Wallet. The database includes 6 tables:

* **fact\_transaction**: Store information of all types of transactions: Payments, Top-up, Transfers, Withdrawals (2019 and 2020)
* **dim\_scenario**: Detailed description of transaction types
* **dim\_payment\_channel**: Detailed description of payment methods
* **dim\_platform**: Detailed description of payment devices
* **dim\_status**: Detailed description of the results of the transaction

**III. SQL Queries**

1. Return the result of **successful transactions** for **each month in 2019**.

**SELECT MONTH (transaction\_time) AS [month]**

**, COUNT (transaction\_id ) AS number\_success\_trans**

**FROM dbo.fact\_transaction\_2019**

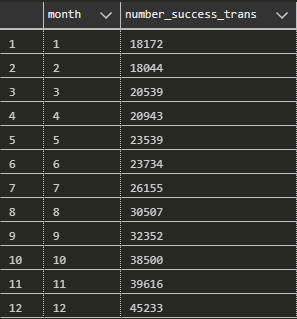
**LEFT JOIN dim\_scenario**

**ON dbo.fact\_transaction\_2019.scenario\_id = dim\_scenario.scenario\_id**

**WHERE status\_id = 1**

**GROUP BY MONTH (transaction\_time)**

**ORDER BY [month]**

**Output:  
**

1. Find the **TOP 3 months** with the **highest** number of **failed transactions** for **each year**.

WITH failed\_month\_table AS (

SELECT YEAR (transaction\_time) AS [year]

, MONTH (transaction\_time) AS [month]

, COUNT (transaction\_id ) AS number\_failed\_trans

FROM (SELECT transaction\_id, transaction\_time, scenario\_id, status\_id FROM dbo.fact\_transaction\_2019

UNION

SELECT transaction\_id, transaction\_time, scenario\_id, status\_id FROM dbo.fact\_transaction\_2020) AS fact\_table

WHERE status\_id != 1 -- giao dịch lỗi

GROUP BY YEAR (transaction\_time), MONTH (transaction\_time)

-- ORDER BY [year], [month]

)

, rank\_table AS (

SELECT \*

, RANK () OVER ( PARTITION BY [year] ORDER BY number\_failed\_trans DESC ) AS rank\_column

FROM failed\_month\_table

-- ORDER BY [year], [month]

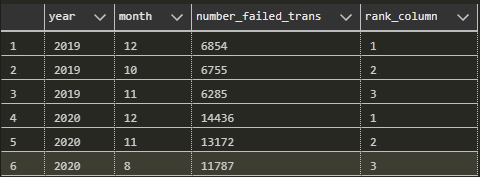
)

SELECT \*

FROM rank\_table

WHERE rank\_column < 4

**Output:**



1. **Calculate the average distance between successful payments for each customer in the Telecom group in 2019.**

**WITH customer\_table AS (**

**SELECT TOP 1000 customer\_id, transaction\_id, transaction\_time**

**, LAG (transaction\_time, 1) OVER ( PARTITION BY customer\_id ORDER BY transaction\_time ASC ) AS previous\_time**

**, DATEDIFF (day, LAG (transaction\_time, 1) OVER ( PARTITION BY customer\_id ORDER BY transaction\_time ASC )**

**, transaction\_time) AS gap\_day**

**FROM (SELECT customer\_id, transaction\_id, transaction\_time, scenario\_id FROM dbo.fact\_transaction\_2019**

**UNION**

**SELECT customer\_id, transaction\_id, transaction\_time, scenario\_id FROM dbo.fact\_transaction\_2020) AS fact\_table**

**LEFT JOIN dbo.dim\_scenario**

**ON fact\_table.scenario\_id = dbo.dim\_scenario.scenario\_id**

**WHERE category = 'Telco'**

**)**

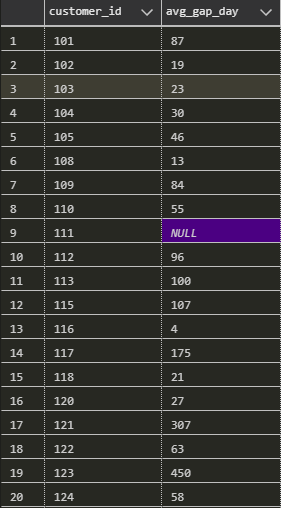
**SELECT customer\_id**

**, AVG (gap\_day) AS avg\_gap\_day**

**FROM customer\_table**

**GROUP BY customer\_id**

**Output:**

****

1. **Time Series Analysis**

**Requirements:** Create a table that shows the number of transactions for the 3 sub-categories: Electricity, Internet, and Water, as well as the total transactions for all 3 sub-categories in each month from 2019-2020. Then calculate the percentage of sales for each sub-category in each month compared to the total transactions.

**WITH month\_table AS (**

**SELECT**

**YEAR(transaction\_time) AS [year]**

**, MONTH (transaction\_time) AS [month]**

**, sub\_category**

**, COUNT (transaction\_id) AS number\_trans**

**FROM (SELECT transaction\_id, transaction\_time, scenario\_id, status\_id FROM dbo.fact\_transaction\_2019**

**UNION**

**SELECT transaction\_id, transaction\_time, scenario\_id, status\_id FROM dbo.fact\_transaction\_2020)**

**AS fact\_table**

**LEFT JOIN dbo.dim\_scenario**

**ON fact\_table.scenario\_id = dbo.dim\_scenario.scenario\_id**

**LEFT JOIN dbo.dim\_status**

**ON fact\_table.status\_id = dbo.dim\_status.status\_id**

**WHERE category = 'Billing'**

**AND fact\_table.status\_id = 1**

**GROUP BY YEAR(transaction\_time), MONTH (transaction\_time), sub\_category**

**)**

**, pivot\_table AS (**

**SELECT [year], [month]**

**, SUM ( CASE WHEN sub\_category = 'electricity' THEN number\_trans ELSE 0 END ) AS elec\_trans**

**, SUM ( CASE WHEN sub\_category = 'internet' THEN number\_trans ELSE 0 END ) AS internet\_trans**

**, SUM ( CASE WHEN sub\_category = 'water' THEN number\_trans ELSE 0 END ) AS water\_trans**

**FROM month\_table**

**GROUP BY [year], [month]**

**-- ORDER BY [year], [month]**

**)**

**SELECT \***

**, elec\_trans + internet\_trans + water\_trans AS total\_trans\_month**

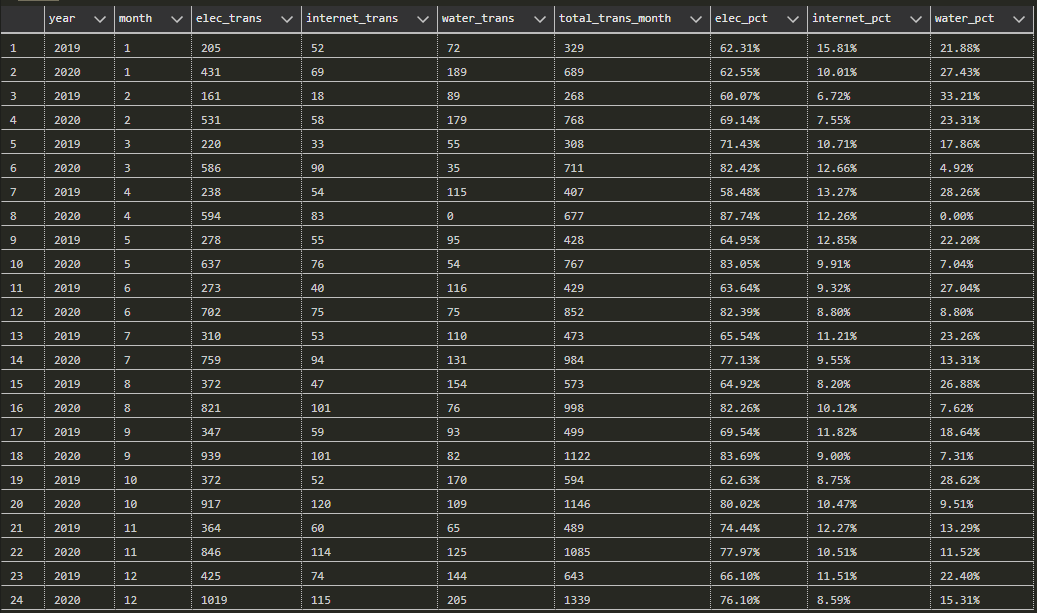
**, FORMAT ( elec\_trans \*1.0 / ( elec\_trans + internet\_trans + water\_trans ), 'p') AS elec\_pct**

**, FORMAT ( internet\_trans \*1.0 / ( elec\_trans + internet\_trans + water\_trans ), 'p') AS internet\_pct**

**, FORMAT ( water\_trans \*1.0 / ( elec\_trans + internet\_trans + water\_trans ), 'p') AS water\_pct**

**FROM pivot\_table**

**Output:**

****

1. **RFM analysis**

Segment all customers in 2019-2020 into 9 distinct groups by percent ranking and count the total customers of each group.

**WITH fact\_table AS -- tạo bảng fact chung**

**(SELECT customer\_id**

**, transaction\_id**

**, transaction\_time**

**, charged\_amount**

**, sub\_category**

**, cast(transaction\_time as *date*) as transaction\_date**

**FROM fact\_transaction\_2019 fact\_2019**

**JOIN dim\_scenario scen ON scen.scenario\_id = fact\_2019.scenario\_id**

**WHERE status\_id = 1**

**AND category = 'Billing'**

**UNION ALL**

**SELECT customer\_id**

**, transaction\_id**

**, transaction\_time**

**, charged\_amount**

**, sub\_category**

**, cast(transaction\_time as *date*) as transaction\_date**

**FROM fact\_transaction\_2020 fact\_2020**

**JOIN dim\_scenario scen ON scen.scenario\_id = fact\_2020.scenario\_id**

**WHERE status\_id = 1**

**AND category = 'Billing')**

**,**

**rfm\_model as (SELECT customer\_id,**

**DATEDIFF(DAY, MAX(transaction\_time), '2020-12-31') AS Recency,**

**COUNT(DISTINCT transaction\_id) AS Frequency,**

**SUM(charged\_amount) AS Monetary**

**FROM fact\_table**

**GROUP BY customer\_id**

**)**

**,**

**rfm\_percent as (**

**SELECT customer\_id**

**, PERCENT\_RANK() OVER ( ORDER BY Recency DESC) AS r\_percent\_rank**

**, PERCENT\_RANK() OVER ( ORDER BY Frequency ASC) AS f\_percent\_rank**

**, PERCENT\_RANK() OVER ( ORDER BY Monetary ASC) AS m\_percent\_rank**

**FROM rfm\_model**

**)**

**, tier\_table as(**

**SELECT customer\_id,**

**CASE WHEN r\_percent\_rank <= 0.25 then 4**

**WHEN r\_percent\_rank <= 0.5 then 3**

**WHEN r\_percent\_rank <= 0.75 then 2**

**ELSE 1 end r\_tier,**

**CASE WHEN f\_percent\_rank <= 0.25 then 4**

**WHEN f\_percent\_rank <= 0.5 then 3**

**WHEN f\_percent\_rank <= 0.75 then 2**

**ELSE 1 end f\_tier,**

**CASE WHEN m\_percent\_rank <= 0.25 then 4**

**WHEN m\_percent\_rank <= 0.5 then 3**

**WHEN m\_percent\_rank <= 0.75 then 2**

**ELSE 1 end m\_tier**

**FROM rfm\_percent**

**)**

**, score\_table as (**

**SELECT customer\_id**

**, CONCAT(r\_tier, f\_tier, m\_tier) as score**

**FROM tier\_table)**

**, label\_table as (**

**SELECT \***

**, (CASE**

**WHEN score = '111' THEN 'Best customers'**

**WHEN score LIKE '[34][34][1-4]' THEN 'Lost Bad Customers'**

**WHEN score LIKE '[34]2[1-4]' THEN 'Lost Customers'**

**WHEN score LIKE '21[1-4]' THEN 'Almost lost'**

**WHEN score LIKE '11[2-4]' THEN 'Loyal customers'**

**WHEN score LIKE '[12][1-3]1' THEN 'Big Spender'**

**WHEN score LIKE '[12]4[1-4]' THEN 'New customers'**

**WHEN score LIKE '[34]1[1-4]' THEN 'Hibernating'**

**WHEN score LIKE '[12][23][2-4]' THEN 'Potential Loyalist'**

**END**

**) as label**

**FROM score\_table)**

**SELECT label**

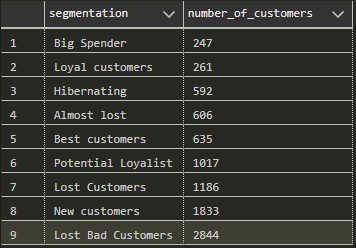
**, COUNT(customer\_id) as number\_of\_customers**

**FROM label\_table**

**GROUP BY label**

**ORDER BY number\_of\_customers**

**Output:**

****

**Note:** The percent\_rank function allows for greater flexibility in filtering and sorting data, while the Ntile function divides data into a fixed number of equally-sized groups, which may not be suitable for all use cases. By using percent\_rank, I can easily filter data based on a particular percentile, such as the top or bottom 50%, which can be useful in various analysis scenarios.