As funções de janela são uma aplicação avançada do uso do SQL para manipulação de dados.

É uma grande ferramenta para possuir, que pode te ajudar a chegar mais rápido em análises estatísticas avançadas!

Para solidificar seus conhecimentos, responda às perguntas abaixo.

Submeta suas respostas em um arquivo com extensão .sql, e use comentários (com os caracteres -- ou /\* \*/) para escrever suas respostas por extenso e delimitar as questões.

Dica: utilize subconsultas para facilitar seu trabalho!

/\*

1. Escreva uma consulta SQL que, com base no total em pagamentos realizados (coluna amount da tabela payment), divida todos os consumidores da locadora em 4 grupos.

Além disso, com ajuda da função CASE WHEN, classifique consumidores nesses grupos, do de maiores pagadores ao de menores, respectivamente em:

o "Especial"

o "Fiel"

o "Ocasional"

o "Iniciante"

Sua consulta final deve conter as colunas: email, classificacao, total\_pago

\*/

SELECT c.email,

CASE WHEN NTILE(4) OVER (ORDER BY SUM(p.amount) DESC) = 1 THEN 'Especial'

WHEN NTILE(4) OVER (ORDER BY SUM(p.amount) DESC) = 2 THEN 'Fiel'

WHEN NTILE(4) OVER (ORDER BY SUM(p.amount) DESC) = 3 THEN 'Ocasional'

ELSE 'Iniciante' END AS classificacao,

SUM(p.amount) AS total\_pago

FROM customer AS c

INNER JOIN payment AS p

ON c.customer\_id = p.customer\_id

GROUP BY c.email

ORDER BY total\_pago DESC;

/\*

2. Escreva uma consulta SQL que responda: qual foi a primeira loja da rede a atingir um total de $10.000 no mês de julho/2005?

\*/

WITH sales AS (

SELECT DATE\_FORMAT(p.payment\_date, '%Y-%m-%d') data,

c.store\_id,

SUM(p.amount) AS total\_vendas

FROM customer AS C

INNER JOIN payment AS p

ON c.customer\_id = p.customer\_id

WHERE p.payment\_date BETWEEN '2005-07-01' AND '2005-07-31'

GROUP BY data, c.store\_id

ORDER BY store\_id, data),

cumulative\_sum AS (

SELECT \*,

SUM(total\_vendas) OVER(PARTITION BY store\_id ORDER BY data) acumulado\_vendas

FROM sales)

SELECT data,

store\_id,

acumulado\_vendas

FROM cumulative\_sum

WHERE acumulado\_vendas >= 10000

ORDER BY data ASC

LIMIT 1;