# Respostas: Perguntas de Negócios com SQL IV

Códigos SQL de resposta à Prática: Perguntas de Negócios com SQL IV, da 8ª aula, do curso de Data Analytics da Awari.

1. Retorne todos os pagamentos do cliente, com suas datas de aprovação, valor da compra e o valor total que o cliente já gastou em todas as suas compras, mostrando somente os clientes onde o valor da compra é diferente do valor total já gasto;

SELECT

\*

FROM

(

SELECT

O.customer\_id AS cliente,

P.payment\_value AS valor\_compra,

SUM(P.payment\_value) OVER (PARTITION BY O.customer\_id ORDER BY P.payment\_value) AS cum,

SUM(P.payment\_value) OVER (PARTITION BY O.customer\_id) AS valor\_total,

O.order\_approved\_at AS data\_aprovacao

FROM `awari-data-analytics-2023.dataset\_olist.orders` O

INNER JOIN `awari-data-analytics-2023.dataset\_olist.order\_payments` P

ON P.order\_id = O.order\_id

)

WHERE valor\_compra < valor\_total

ORDER BY cliente

2. Retorne as categorias válidas, suas somas totais dos valores de vendas, um ranqueamento de maior valor para menor valor junto com o somatório acumulado dos valores pela mesma regra do ranqueamento;

SELECT

categoria,

total\_vendas,

SUM(total\_vendas) OVER (ORDER BY total\_vendas) AS cum

FROM

(

SELECT

PRO.product\_category\_name AS categoria,

SUM(PAY.payment\_value) AS total\_vendas

FROM `awari-data-analytics-2023.dataset\_olist.order\_payments` PAY

INNER JOIN `awari-data-analytics-2023.dataset\_olist.order\_items` OI

ON PAY.order\_id = OI.order\_id

INNER JOIN `awari-data-analytics-2023.dataset\_olist.products` PRO ON OI.product\_id = PRO.product\_id

WHERE PRO.product\_category\_name IS NOT NULL

GROUP BY 1

)

ORDER BY 2 DESC

3. Crie uma view (SELLER\_STATS) para mostrar por fornecedor, a quantidade de itens enviados, o tempo médio de postagem após a aprovação da compra, a quantidade total de pedidos de cada Fornecedor, note que trabalharemos na mesma query com 2 granularidades diferentes;

--CREATE OR REPLACE VIEW SELLER\_STATS AS

SELECT

S.seller\_id AS VENDEDOR,

COUNT (D.order\_status) AS qtd\_itens,

CASE

WHEN D.order\_status = 'delivered' OR D.order\_status = 'shipped' THEN 'ENVIADO'

ELSE 'NAO\_ENVIADO'

END as STATUS,

SUM (COUNT (D.order\_status)) OVER (PARTITION BY S.seller\_id) AS VENDA\_TOTAL\_VENDEDOR,

avg(date\_diff(D.order\_delivered\_carrier\_date, D.order\_approved\_at, DAY)) AS MEDIA\_POSTAGEM

FROM `awari-data-analytics-2023.dataset\_olist.sellers` S

INNER JOIN `awari-data-analytics-2023.dataset\_olist.order\_items` O

ON O.seller\_id = S.seller\_id

INNER JOIN `awari-data-analytics-2023.dataset\_olist.orders` D

ON D.order\_id = O.order\_id

GROUP BY

STATUS,

VENDEDOR

ORDER BY 1

4. Queremos dar um cupom de 10% do valor da última compra do cliente. Porém os clientes elegíveis a este cupom devem ter feito uma compra anterior a última (a partir da data de aprovação do pedido) que tenha sido maior ou igual o valor da última compra. Crie uma query que retorne os valores dos cupons para cada um dos clientes elegíveis;

SELECT

CLIENTE,

PENULTIMA\_COMPRA,

ULTIMA\_COMPRA,

CASE

WHEN PENULTIMA\_COMPRA >= ULTIMA\_COMPRA THEN ROUND (ULTIMA\_COMPRA \* 0.10,2)

ELSE 0

END CUPOM

FROM

(

select

P.order\_id AS PEDIDO,

C.customer\_unique\_id AS CLIENTE,

D.order\_status AS STATUS,

DATE(D.order\_approved\_at) AS DATA\_APROVACAO,

SUM (P.payment\_value) AS VALOR\_COMPRA,

last\_value (P.payment\_value) over (PARTITION by C.customer\_unique\_id ORDER BY DATE(D.order\_approved\_at)) AS ULTIMA\_COMPRA,

LAG (P.payment\_value) OVER (PARTITION BY C.customer\_unique\_id ORDER BY DATE(order\_approved\_at)) AS PENULTIMA\_COMPRA

from `awari-data-analytics-2023.dataset\_olist.orders` D

INNER JOIN `awari-data-analytics-2023.dataset\_olist.order\_payments` P

ON P.order\_id = D.order\_id

INNER JOIN `awari-data-analytics-2023.dataset\_olist.customers` C

ON C.customer\_id = D.customer\_id

WHERE D.order\_status = 'delivered'

and D.order\_approved\_at IS NOT NULL

GROUP BY 1, 2, 3, 4, P.payment\_value, D.order\_approved\_at

)

WHERE PENULTIMA\_COMPRA IS NOT NULL