



CEUB

EDUCAÇÃO SUPERIOR

ceub.br



BANCO DE DADOS II

SQL - CONSULTAS AVANÇADAS

Prof. Leonardo R. de Deus

1. CONSULTAS SQL BÁSICO

Padrão de uma consulta básica utilizando SQL:

SELECT	o que queremos selecionar (atributos da tabela)
FROM	de onde vamos selecionar (de qual tabela)
WHERE	qual condição deve ser atendida

```
SELECT nome_cliente, estado  
FROM loja.tb01_cliente  
WHERE estado = 'DF'
```

1. CONSULTAS SQL BÁSICO

Padrão de uma consulta básica utilizando SQL:

SELECT FROM WHERE	o que queremos selecionar (atributos da tabela)
	de onde vamos selecionar (de qual tabela)
	qual condição deve ser atendida

OPERADORES DE COMPARAÇÃO

Operador de Comparação	Descrição	Exemplo	Descrição do Exemplo
A = B	A é igual a B.	preco_unitario=200	O preço unitário é 200 G.
A > B	A é maior que B.	preco_unitario>200	O preço unitário é maior que 200 G.
A >= B	A é maior ou igual a B.	preco_unitario>=200	O preço unitário é maior ou igual a 200 G.
A < B	A é menor do que B.	preco_unitario<200	O preço unitário é menor que 200 G.
A <= B	A é menor ou igual a B.	preco_unitario<=200	O preço unitário é menor ou igual a 200 G.
A <> B	A é diferente de B.	preco_unitario<>200	O preço unitário não pode ser 200 G.

OPERADORES LÓGICOS

Operador Lógico	Descrição	Exemplo	Descrição do Exemplo
AND	A e B	codigo_produto >= 200 AND preco_unitario = 100	O código do produto é maior ou igual a 200 e o preço unitário é 100 G.
OR	A ou B	codigo_produto >= 200 OR preco_unitario = 100	O código do produto é maior ou igual a 200 ou o preço unitário é 100 G.
NOT	Não A	NOT preco_unitario = 100	O preço unitário não pode ser 100 G.

2. FUNÇÕES DE AGREGAÇÃO

Avançando um pouco

Utilizadas para realizar cálculos e resumir dados em uma consulta.

FUNÇÃO	DESCRIÇÃO
COUNT	Retorna a quantidade de registros
SUM	Efetua a soma dos valores de um campo numérico
AVG	Calcula a média dos valores de um campo numérico
MIN	Retorna o menor valor existente em um campo da tabela
MAX	Retorna o maior valor existente em um campo da tabela

Qual o preço médio dos produtos por categoria?

Mão na Massa!

**anexar tarefa
diária**

3. UTILIZAÇÃO DE JOINS

E como consultar informações que não estão na mesma tabela?

JOIN

CLIENTES			PEDIDOS		
ID_Cliente	Nome	CPF	ID_Pedido	Data	Valor
Nome		180.00	Nome	Data	Valor
101, João Silva		180.00	101, João Silva		150.00
Jidc Silva			drc		
123-465-779-00		150.00	2023-10-26-26		150.00
P-001		180.00	P-002		150.00
123-466-779-00		160.00	P-001 101		150.00
Bran		200.00	101		150.00
26th		260.00	102		250.00
2002		250.00	2003 102		250.00
PUMin, Short		200.00	PURm MAdcre		250.00
Bam UI		250.00	60bL		250.00

3. UTILIZAÇÃO DE JOINS

INNER JOIN (simplesmente Join)

Mostre o nome do cliente, o número e data do pedido

SELECT

c.nome_cliente,
p.numero_pedido,
p.data_pedido

FROM

loja.tb01_cliente **AS** c

INNER JOIN loja.tb04_pedido **AS** p **ON** c.id_cliente = p.id_cliente;

nome_cliente character varying (100)	numero_pedido character varying (50)	data_pedido date
Ana Carolina Souza	PED-2025-00001	2025-01-15
Ana Carolina Souza	PED-2025-00010	2025-03-22
Ana Carolina Souza	PED-2025-00025	2025-08-10
Bruno Alves	PED-2025-00002	2025-01-18
Bruno Alves	PED-2025-00011	2025-04-01
Carla Dias	PED-2025-00003	2025-02-01
Daniel Costa	PED-2025-00004	2025-02-05

3. UTILIZAÇÃO DE JOINS

LEFT JOIN

Mostre o
número do
pedido feito por
cada cliente

```
SELECT
    c.nome_cliente,
    p.numero_pedido
FROM
    loja.tb01_cliente AS c
LEFT JOIN
    loja.tb04_pedido AS p ON c.id_cliente = p.id_cliente
ORDER BY c. nome_cliente
```

23	Larissa Mendes	PED-2025-00016
24	Marcos Andrade	PED-2025-00017
25	Marta Vilela	[null]
26	Natália Rocha	PED-2025-00020
27	Otávio Nunes	PED-2025-00021
28	Patrícia Azevedo	PED-2025-00022
29	Raul Prado	[null]

3. UTILIZAÇÃO DE JOINS

RIGHT JOIN

Mostre o
número e o
status dos
pedidos

SELECT

p.numero_pedido,
s.descricao

FROM

loja.tb04_pedido **AS** p

RIGHT JOIN

loja.tb03_status_pedido **AS** s **ON** p.id_status_pedido = s.id_status_pedido

23	PED-2025-00020	Entregue
24	PED-2025-00021	Pagamento Aprovado
25	PED-2025-00022	Processando
26	[null]	Contestado
27	[null]	Em análise

3. UTILIZAÇÃO DE JOINS

FULL JOIN

Mostre o
nome do
cliente,
número e
status dos
pedidos

SELECT

c.nome_cliente,
p.numero_pedido,
s.descricao

FROM

loja.tb01_cliente **AS** c

FULL JOIN loja.tb04_pedido **AS** p **ON** c.id_cliente = p.id_cliente

FULL JOIN loja.tb03_status_pedido as s **ON** p.id_status_pedido =
s.id_status_pedido

23	Natália Rocha	PED-2025-00020	Entregue
24	Otávio Nunes	PED-2025-00021	Pagamento Aprovado
25	Patrícia Azevedo	PED-2025-00022	Processando
26	Marta Vilela	[null]	[null]
27	Ana Carolina Souza	[null]	[null]
28	Raul Prado	[null]	[null]
29	Augusto Dias	[null]	[null]
30	[null]	[null]	Contestado
31	[null]	[null]	Em análise

Qual o valor total gasto por cada cliente?

Mão na Massa!

**anexar tarefa
diária**

4. SUBCONSULTAS

É um **SELECT** dentro de outro comando SQL, normalmente um **WHERE** ou **FROM**.

São usadas para responder perguntas que dependem do resultado de outra pergunta.

Quais produtos custam mais que a média de preço de todos os produtos da loja?

SELECT nome_produto, preco

FROM loja.tb02_produto

WHERE preco > (**SELECT** **AVG**(preco) **FROM** loja.tb02_produto);

4. SUBCONSULTAS

Outro exemplo:

Quais clientes já fizeram pelo menos um pedido?

```
SELECT
    nome_cliente,
    email
FROM loja.tb01_cliente
WHERE id_cliente IN (SELECT DISTINCT id_cliente FROM loja.tb04_pedido);
```

4. SUBCONSULTAS

Subconsulta na cláusula **SELECT**

Para cada produto, mostre seu nome, seu preço e também o preço médio de todos os produtos da mesma categoria

SELECT

```
nome_produto, categoria,  
preco,  
SELECT ROUND(AVG(preco), 2)  
FROM loja.tb02_produto AS p2  
WHERE p2.categoria = p1.categoria ) AS media_preco_categoria
```

```
FROM loja.tb02_produto AS p1
```

```
ORDER BY categoria, nome_produto;
```


4. SUBCONSULTAS

Subconsulta na cláusula FROM

Selecione o nome do cliente e a quantidade total de itens que ele já comprou

SELECT

c.nome_cliente,
itens_por_pedido.total_itens

FROM loja.tb01_cliente **AS** c

JOIN loja.tb04_pedido **AS** p **ON** c.id_cliente = p.id_cliente

JOIN (**SELECT** id_pedido, **SUM**(quantidade) **AS** total_itens

FROM loja.tb05_item_pedido

GROUP BY id_pedido) **AS** itens_por_pedido **ON** p.id_pedido = itens_por_pedido.id_pedido

ORDER BY c.nome_cliente;

5. COMMONS TABLE EXPRESSIONS

CTE é uma tabela temporária nomeada, que armazena o resultado de um SELECT, que pode então ser usado em outra consulta principal.

São usadas para quebrar um problema complexo em passos lógicos e sequenciais, tornando o código mais legível e organizado.

Quais produtos custam mais que a média de preço de todos os produtos da loja?

```
WITH media_geral AS (  
    SELECT AVG(preco) AS preco_medio FROM loja.tb02_produto  
)  
SELECT nome_produto, preco, preco_medio  
FROM loja.tb02_produto, media_geral  
WHERE preco > preco_medio;
```

Mesmo problema, soluções diferentes

Mostre o valor total gasto por cada cliente na loja.

- 1. Selecionar o valor total gasto por cada cliente;**
- 2. Buscar o nome de cada cliente para mostrar o resultado final.**

Mesmo problema, soluções diferentes

Mostre o valor total gasto por cada cliente na loja.

CTE

```
WITH gasto_por_cliente AS (  
  SELECT  
    p.id_cliente,  
    SUM(ip.quantidade * ip.valor_unitario) AS valor_total_gasto  
  FROM  
    loja.tb04_pedido AS p  
  JOIN  
    loja.tb05_item_pedido AS ip ON p.id_pedido = ip.id_pedido  
  GROUP BY  
    p.id_cliente  
)  
  
SELECT  
  c.nome_cliente,  
  gpc.valor_total_gasto  
FROM  
  loja.tb01_cliente AS c  
JOIN  
  gasto_por_cliente AS gpc ON c.id_cliente = gpc.id_cliente  
ORDER BY  
  gpc.valor_total_gasto DESC;
```

Mesmo problema, soluções diferentes

Mostre o valor total gasto por cada cliente na loja.

Subconsulta

```
SELECT
    c.nome_cliente,
    sub.valor_total_gasto
FROM
    loja.tb01_cliente AS c
JOIN
    (
        SELECT
            p.id_cliente,
            SUM(ip.quantidade * ip.valor_unitario) AS valor_total_gasto
        FROM
            loja.tb04_pedido AS p
        JOIN
            loja.tb05_item_pedido AS ip ON p.id_pedido = ip.id_pedido
        GROUP BY
            p.id_cliente
    ) AS sub ON c.id_cliente = sub.id_cliente
ORDER BY
    sub.valor_total_gasto DESC;
```

Quando optar por uma ou outra?

Subconsulta	CTE
A lógica é simples, curta e direta	A lógica tem múltiplos passos ou é complexa
O resultado é necessário apenas uma vez	Você precisa reutilizar o mesmo resultado intermediário na consulta
A consulta inteira é pequena e a legibilidade não é um problema	A legibilidade do código e a manutenção do código são importantes

Qual é a categoria de produto mais vendida em termos de quantidade de itens? Mostre o ranking.

Mão na Massa!

**anexar tarefa
diária**

Mais além ...

Utilizando a cláusula **HAVING**

É usada para filtrar grupos inteiros de linhas depois que eles já foram criados pela cláusula **GROUP BY**, e aplicar condições em funções de agregação (count, sum, avg, etc).

Mais além ...

Utilizando a cláusula HAVING

Característica	WHERE	HAVING
Propósito	Filtrar linhas individuais	Filtrar grupos de linhas
Quando Atua?	Antes do GROUP BY	Depois do GROUP BY
Uso com Agregações	Não pode conter funções de agregação	É feito para conter funções de agregação
Ordem na consulta	Vem antes do GROUP BY	Vem depois do GROUP BY

Mais além ...

Utilizando a cláusula HAVING

Mostre o nome e quantidade de pedidos, de todos os clientes que fizeram mais de um pedido.

```
SELECT
    c.nome_cliente,
    COUNT(p.id_pedido) AS total_pedidos
FROM
    loja.tb01_cliente AS c
JOIN
    loja.tb04_pedido AS p ON c.id_cliente = p.id_cliente
GROUP BY
    c.nome_cliente
HAVING
    COUNT(p.id_pedido) > 1;
```

**OBRIGADO
A TODOS!**

