

EXPLICANDO O BANCO DE DADOS

1. Quero saber os registros dos produtos que são perfumes:

```
SELECT * FROM produto WHERE nome LIKE '%Perfume%';
```

Resultado:

	123 produto_id	A-Z nome	123 preco	123 estoque	A-Z genero
1	1	Perfume amadeirado	49,9	100	Masculino
2	8	Perfume Egeo	110	30	Masculino

2. O EXPLAIN mostra como o banco executa a query.

```
EXPLAIN SELECT * FROM produto WHERE nome LIKE '%Perfume%';
```

Resultado:

	A-Z QUERY PLAN
1	Seq Scan on produto (cost=0.00..12.62 rows=1 width=360)
2	Filter: ((nome)::text ~~ '%Perfume% '::text)

3. Cria um índice no campo nome da tabela produto. Isso melhora o desempenho em consultas que buscam pelo nome.

```
CREATE INDEX idx_nome_produto ON produto(nome);
```

4. Alterou a coluna estoque para o tipo varchar(20).

```
ALTER TABLE produto ALTER COLUMN estoque TYPE varchar(20);
```

Resultado:

Nome da coluna:	<input type="text" value="estoque"/>
Tipo de dados:	<input type="text" value="varchar(20)"/>

5. Criou o usuário maria@localhost e concedeu permissão para conectar ao banco, usar o schema public e acesso total às tabelas.

```
CREATE USER "maria@localhost" WITH PASSWORD 'senha123';  
GRANT CONNECT ON DATABASE db_revenda_mariaf TO "maria@localhost";  
GRANT USAGE ON SCHEMA public TO "maria@localhost";  
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO "maria@localhost" WITH GRANT OPTION;
```

6. Criou o usuário isabela@localhost e concedeu permissão limitada.

```
CREATE USER "isabela@localhost" WITH PASSWORD 'senha456';  
GRANT CONNECT ON DATABASE db_revenda_mariaf TO "isabela@localhost";
```

- 7.

```
EXPLAIN SELECT * FROM produto WHERE nome LIKE '%Perfume%';
```

Resultado:

	A-Z QUERY PLAN
1	Seq Scan on produto (cost=0.00..1.12 rows=1 width=414)
2	Filter: ((nome)::text ~~ '%Perfume% '::text)

8. Traz os pedidos com os clientes correspondentes.

```
SELECT p.pedido_id, p.data_pedido, c.nome
FROM pedido p INNER JOIN cliente c ON p.cliente_id = c.cliente_id;
SELECT p.pedido_id, p.data_pedido, c.nome
FROM pedido p LEFT JOIN cliente c ON p.cliente_id = c.cliente_id;
SELECT p.pedido_id, p.data_pedido, c.nome
FROM pedido p RIGHT JOIN cliente c ON p.cliente_id = c.cliente_id;
```

Resultado:

	pedido_id	data_pedido	nome
1	1	2024-08-01	Maria Silva
2	2	2024-08-05	João Santos
3	3	2024-08-07	Ana Pereira
4	4	2024-08-10	Carlos Lima
5	5	2024-08-12	Beatriz Souza
6	6	2024-08-15	Lucas Oliveira
7	7	2024-08-18	Fernanda Rocha
8	8	2024-08-20	Ricardo Alves
9	9	2024-08-22	Juliana Costa
10	10	2024-08-25	Paulo Martins

9. Relaciona pedido_produto (tabela de ligação) com produto e retorna o pedido, produto e quantidade.

```
SELECT pp.pedido_id, pr.nome, pp.quantidade
FROM pedido_produto pp INNER JOIN produto pr ON pp.produto_id = pr.produto_id;
SELECT pp.pedido_id, pr.nome, pp.quantidade
FROM pedido_produto pp LEFT JOIN produto pr ON pp.produto_id = pr.produto_id;
SELECT pp.pedido_id, pr.nome, pp.quantidade
FROM pedido_produto pp RIGHT JOIN produto pr ON pp.produto_id = pr.produto_id;
```

Resultado:

	pedido_id	nome	quantidade
1	1	Perfume amadeirado	2
2	1	Máscara de Skin Care Facial	10
3	3	Espuma barbeadora	3
4	3	Esponja de maquiagem	1
5	4	Espuma barbeadora	1
6	5	Hidratante de lavanda	1
7	7	Esfoliante	1
8	8	Desodorante	1
9	8	Máquina de barbear	1
10	10	Creme antiatrito	1

10. Cada funcionário é relacionado a cada pedido.

```
➤ SELECT f.nome, p.pedido_id
  FROM funcionario f INNER JOIN pedido p ON 1=1 LIMIT 10;
➤ SELECT f.nome, p.pedido_id
  FROM funcionario f LEFT JOIN pedido p ON 1=1 LIMIT 10;
➤ SELECT f.nome, p.pedido_id
  FROM funcionario f RIGHT JOIN pedido p ON 1=1 LIMIT 10;
```

Resultado:

	nome	pedido_id
1	Joana Almeida	1
2	Pedro Gonçalves	1
3	Marcela Ferreira	1
4	Renato Dias	1
5	Luana Ribeiro	1
6	Carlos Henrique	1
7	Ana Beatriz	1
8	Ricardo Lima	1
9	Paula Mendes	1
10	Lucas Souza	1

11. Relaciona pedidos com seus pagamentos e mostra ID do pagamento, ID do pedido e valor.

```
⊖ SELECT pg.pagamento_id, p.pedido_id, pg.valor
  FROM pagamento pg INNER JOIN pedido p ON pg.pedido_id = p.pedido_id;
⊖ SELECT pg.pagamento_id, p.pedido_id, pg.valor
  FROM pagamento pg LEFT JOIN pedido p ON pg.pedido_id = p.pedido_id;
⊖ SELECT pg.pagamento_id, p.pedido_id, pg.valor
  FROM pagamento pg RIGHT JOIN pedido p ON pg.pedido_id = p.pedido_id;
```

Resultado:

	pagamento_id	pedido_id	valor
1	1	1	299,7
2	2	2	0
3	3	3	420
4	4	4	120
5	5	5	79,9
6	6	6	59,9
7	7	7	225
8	8	8	180
9	9	9	0
10	10	10	70

12. Define o campo turno como NULL (sem turno definido) para todos os funcionários com cargo "Caixa".

```
UPDATE funcionario SET turno = NULL WHERE cargo = 'Caixa';
```

Resultado:

```
('Renato Dias', 'Caixa', 1700.00, '2023-02-25', 'Noite'),
```