CREATE TABLE

1. O que é uma TABELA em um banco de dados?

Uma **tabela** é como uma planilha dentro do banco de dados. Ela armazena informações organizadas em **colunas** (como se fossem as colunas de uma planilha) e em **linhas** (cada linha é um registro de dados). Por exemplo, numa tabela de clientes, cada linha representaria um cliente diferente, e cada coluna armazenaria uma informação sobre esse cliente, como nome, e-mail ou telefone.

2. O que é o comando `CREATE TABLE`?

O comando **CREATE TABLE** é usado para **criar uma nova tabela** no banco de dados. Ele define:

- O **nome da tabela**.
- As **colunas** que vão existir nessa tabela.
- O tipo de dado que vai ser armazenado em cada coluna.

3. Exemplo com a tabela `Clientes`:

Vamos imaginar que você está criando uma tabela para armazenar informações dos **clientes** de uma empresa, como o nome, e-mail e telefone.

```
CREATE TABLE Clientes (
        cliente_id INT PRIMARY KEY,
        nome VARCHAR(100),
        email VARCHAR(100),
        telefone VARCHAR(20)
);
```

O que cada parte do comando faz:

- **CREATE TABLE Clientes**: Esse comando cria uma nova tabela chamada **Clientes**.
- **cliente_id INT PRIMARY KEY**: A coluna `cliente_id` será um número inteiro que identifica cada cliente de forma única (chave primária).
- **nome VARCHAR(100)**: A coluna `nome` vai armazenar o nome do cliente, com até 100 caracteres.
- **email VARCHAR(100)**: A coluna `email` vai guardar o e-mail do cliente, também com até 100 caracteres.
- **telefone VARCHAR(20)**: A coluna `telefone` vai armazenar o número de telefone do cliente, com até 20 caracteres.

4. Explicando os tipos de dados:

- **INT**: Um número inteiro. Usamos esse tipo de dado para o `cliente_id`, que é a identificação única do cliente.
- **VARCHAR(n)**: Texto com um limite de até `n` caracteres. Usamos esse tipo de dado para armazenar o nome, e-mail e telefone dos clientes. Por exemplo, `VARCHAR(100)` significa que o texto pode ter até 100 caracteres

5. Importância da Chave Primária (PRIMARY KEY):

A **chave primária** (`PRIMARY KEY`) é um campo que garante que cada registro da tabela será **único**. No nosso exemplo, usamos a coluna `cliente_id` como chave primária. Isso significa que dois clientes diferentes **não podem ter o mesmo `cliente_id`**.

6. Exemplo mais completo com a tabela `Pedidos`:

Vamos criar uma tabela chamada **Pedidos**, que armazena os pedidos feitos pelos clientes. Essa tabela vai ter uma **chave estrangeira** para indicar qual cliente fez o pedido:

```
CREATE TABLE Pedidos (
        pedido_id INT PRIMARY KEY,
        data_pedido DATE,
        valor_total DECIMAL(10, 2),
        cliente_id INT,
        FOREIGN KEY (cliente_id) REFERENCES Clientes(cliente_id)
);
```

Explicação:

- **pedido_id INT PRIMARY KEY**: O `pedido_id` é a chave primária da tabela **Pedidos**, identificando cada pedido de forma única.
- **data_pedido DATE**: Armazena a data em que o pedido foi feito.
- **valor_total DECIMAL(10, 2)**: Guarda o valor total do pedido, com até 10 dígitos, incluindo 2 casas decimais.
- **cliente_id INT**: Armazena o ID do cliente que fez o pedido. Esse ID vem da tabela **Clientes**.
- **FOREIGN KEY (cliente_id) REFERENCES Clientes(cliente_id)**: Cria uma relação entre a tabela **Pedidos** e a tabela **Clientes**, indicando que o `cliente_id` na tabela **Pedidos** deve existir na tabela **Clientes**.

7. Atividade Prática:

Agora, que tal criar uma tabela de **Produtos**? Use a estrutura abaixo como base:

- A tabela se chamará **Produtos**.
- Ela vai armazenar o `produto_id` (número inteiro, chave primária), `nome` do produto (texto com até 100 caracteres), e o `preco` (decimal com 10 dígitos, sendo 2 decimais).