**1. Criação do Banco de Dados**

SQL

CREATE DATABASE sistema\_comercial;

USE sistema\_comercial;

**2. Criação das Tabelas**

**Tabela clientes:**

SQL

CREATE TABLE clientes (

id\_cliente INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR(255) NOT NULL,

sobrenome VARCHAR(255) NOT NULL,

fone VARCHAR(20),

email VARCHAR(255),

data\_cadastro DATE

);

**Tabela produtos:**

SQL

CREATE TABLE produtos (

id\_produto INT AUTO\_INCREMENT PRIMARY KEY,

nome\_produto VARCHAR(255),

preco DECIMAL(10, 2),

data\_inclusao DATE

);

**Tabela vendas:**

SQL

CREATE TABLE vendas (

id\_venda INT AUTO\_INCREMENT PRIMARY KEY,

id\_cliente INT,

id\_produto INT,

quantidade INT,

valor\_total DECIMAL(10, 2),

data\_venda DATETIME DEFAULT NOW(),

FOREIGN KEY (id\_cliente) REFERENCES clientes(id\_cliente),

FOREIGN KEY (id\_produto) REFERENCES produtos(id\_produto)

);

**3. Inserção de Dados Exemplo**

**Inserir alguns clientes:**

SQL

INSERT INTO clientes (nome, sobrenome, fone, email, data\_cadastro) VALUES

('Carlos', 'Santos', '(11) 98765-4321', 'carlos.santos@gmail.com', CURDATE()),

('Ana', 'Pereira', '(21) 92345-6789', 'ana.pereira@hotmail.com', CURDATE()),

('Roberto', 'Costa', '(31) 93456-7890', 'roberto.costa@yahoo.com', CURDATE()),

('Fernanda', 'Moura', '(51) 99876-5432', 'fernanda.moura@gmail.com', CURDATE()),

('Patrícia', 'Oliveira', '(41) 91234-5678', 'patricia.oliveira@outlook.com', CURDATE()),

('Marcelo', 'Alves', '(71) 93456-1234', 'marcelo.alves@hotmail.com', CURDATE()),

('Juliana', 'Carvalho', '(61) 98765-6789', 'juliana.carvalho@gmail.com', CURDATE()),

('Pedro', 'Lima', '(81) 92345-9876', 'pedro.lima@yahoo.com', CURDATE()),

('Lucas', 'Silveira', '(85) 93456-7654', 'lucas.silveira@outlook.com', CURDATE()),

('Sofia', 'Andrade', '(95) 99876-2345', 'sofia.andrade@gmail.com', CURDATE());

**Inserir alguns produtos:**

SQL

INSERT INTO produtos (nome\_produto, preco) VALUES

('Impressora', 700.00),

('Monitor LED 24"', 1200.00),

('Mouse Gamer', 150.00),

('Teclado Mecânico', 350.00),

('Headset Bluetooth', 450.00),

('HD Externo 1TB', 380.00),

('Cadeira Gamer', 890.00),

('Smartwatch', 950.00),

('Roteador Wi-Fi', 230.00),

('SSD 512GB', 600.00);

**Inserir vendas:**

SQL

INSERT INTO vendas (id\_cliente, id\_produto, quantidade, valor\_total) VALUES

(1, 4, 1, 700.00),

(2, 5, 2, 2400.00),

(3, 6, 3, 450.00),

(4, 7, 1, 350.00),

(5, 8, 1, 450.00),

(1, 9, 2, 760.00),

(6, 10, 1, 890.00),

(2, 11, 1, 950.00),

(3, 12, 2, 460.00),

(4, 13, 1, 600.00);

**4. Consultas Básicas e Funções Agregadas**

**Função SUM() - Soma total de vendas:**

SQL

SELECT SUM(valor\_total) AS Total\_Vendas FROM vendas;

**Função AVG() - Média do valor das vendas:**

SQL

SELECT AVG(valor\_total) AS Media\_Vendas FROM vendas;

**Função MAX() - Valor máximo de uma venda:**

SQL

SELECT MAX(valor\_total) AS Venda\_Maxima FROM vendas;

**Função MIN() - Valor mínimo de uma venda:**

SQL

SELECT MIN(valor\_total) AS Venda\_Minima FROM vendas;

**5. Funções de Manipulação de Texto**

**Função UCASE() ou UPPER() - Converter texto para maiúsculas:**

SQL

SELECT UCASE(nome) AS Nome\_Maiusculo FROM clientes;

**Função LCASE() ou LOWER() - Converter texto para minúsculas:**

SQL

SELECT LCASE(nome) AS Nome\_Minusculo FROM clientes;

**6. Funções de Data e Hora**

**Função NOW() - Data e hora atual:**

SQL

SELECT NOW() AS Data\_Hora\_Atual;

**Função CURDATE() - Data atual:**

SQL

SELECT CURDATE() AS Data\_Atual;

**Função CURTIME() - Hora atual:**

SQL

SELECT CURTIME() AS Hora\_Atual;

**Função CURRENT\_USER() - Nome do usuário corrente:**

SQL

SELECT CURRENT\_USER() AS Usuario\_Atual;

**Função DATE\_FORMAT() - Formatar data:**

SQL

SELECT DATE\_FORMAT(data\_venda, '%d/%m/%Y') AS Data\_Formatada FROM vendas;

**Função TIME\_FORMAT() - Formatar hora:**

SQL

SELECT TIME\_FORMAT(NOW(), '%H:%i:%s') AS Hora\_Formatada;

**7. Funções para Extrair Componentes de Data e Hora**

**Função YEAR() - Extrair o ano de uma data:**

SQL

SELECT YEAR(data\_venda) AS Ano\_Venda FROM vendas;

**Função MONTH() - Extrair o mês de uma data:**

SQL

SELECT MONTH(data\_venda) AS Mes\_Venda FROM vendas;

**Função DAY() - Extrair o dia de uma data:**

SQL

SELECT DAY(data\_venda) AS Dia\_Venda FROM vendas;

**Função HOUR() - Extrair a hora de uma hora:**

SQL

SELECT HOUR(data\_venda) AS Hora\_Venda FROM vendas;

**Função MINUTE() - Extrair os minutos de uma hora:**

SQL

SELECT MINUTE(data\_venda) AS Minuto\_Venda FROM vendas;

**Função SECOND() - Extrair os segundos de uma hora:**

SQL

SELECT SECOND(data\_venda) AS Segundo\_Venda FROM vendas;

**8. Consultas Avançadas com LIKE**

**Função LIKE - Buscar clientes por parte do nome ou email:**

SQL

-- Clientes cujo nome contenha "Silva":

SELECT \* FROM clientes WHERE email LIKE '%Silva%';

-- Clientes cujo email termine com "gmail.com":

SELECT \* FROM clientes WHERE email LIKE '%gmail.com';

--------------------------------------------------------------------------------------------------------

A função **DISTINCT** no SQL é usada para remover valores duplicados em uma consulta, retornando apenas valores únicos em uma ou mais colunas. Vou apresentar exemplos de como usá-la em diferentes cenários no contexto do seu banco de dados.

**9. Exemplo Básico com a Tabela clientes**

Se você quiser obter uma lista dos diferentes sobrenomes registrados, eliminando duplicatas:

SQL

SELECT DISTINCT sobrenome FROM clientes;

* Este comando retorna **apenas sobrenomes únicos**, removendo repetições.

**10. Exemplo com a Tabela produtos**

Se houver registros duplicados no nome dos produtos e você quiser listar apenas os produtos diferentes:

SQL

SELECT DISTINCT nome\_produto FROM produtos;

* Este comando retornará uma lista de produtos, sem repetição de nomes.

**11. Usando DISTINCT em Combinação com Várias Colunas**

Se você quiser garantir que tanto o **nome** do cliente quanto o **sobrenome** sejam únicos na sua consulta:

SQL

SELECT DISTINCT nome, sobrenome FROM clientes;

* Aqui, o DISTINCT vai considerar **nome e sobrenome juntos**, retornando combinações únicas de ambos.

**12. Exemplo com a Tabela vendas**

Para listar os clientes que fizeram compras, garantindo que o mesmo cliente não apareça várias vezes, você pode usar:

SQL

SELECT DISTINCT id\_cliente FROM vendas;

* Isso retornará uma lista de **clientes únicos** que realizaram pelo menos uma compra.

**13. Combinação de DISTINCT com Outras Funções**

Você também pode usar o DISTINCT com funções agregadas, como COUNT(). Por exemplo, para contar quantos produtos únicos foram vendidos:

SQL

SELECT COUNT(DISTINCT id\_produto) AS Produtos\_Unicos\_Vendidos FROM vendas;

* Aqui, **somente os IDs de produtos únicos** serão contados, mesmo que o produto tenha sido vendido várias vezes.

**14. Funções de Manipulação de Strings**

Funções para manipulação de textos.

**14.1. CONCAT() - Concatenar Strings**

Concatena duas ou mais strings em uma única.

SQL

SELECT CONCAT(nome, ' ', sobrenome) AS Nome\_Completo FROM clientes;

* Retorna o nome completo dos clientes concatenando o nome e sobrenome.

**2.2. SUBSTRING() - Extrair parte de uma string**

Extrai uma parte específica de uma string.

SQL

SELECT SUBSTRING(nome, 1, 3) AS Inicio\_Nome FROM clientes;

* Extrai os três primeiros caracteres do nome do cliente.

**2.3. REPLACE() - Substituir parte de uma string**

Substitui parte de uma string por outra.

SQL

SELECT REPLACE(email, 'gmail.com', 'example.com') AS Novo\_Email FROM clientes;

* Substitui "gmail.com" por "example.com" nos e-mails dos clientes.

**2.4. LENGTH() - Comprimento de uma string**

Retorna o número de caracteres de uma string.

SQL

SELECT LENGTH(nome) AS Comprimento\_Nome FROM clientes;

* Retorna o número de caracteres no nome de cada cliente.

**3. DATEDIFF() - Diferença entre duas Datas**

Calcula a diferença em dias entre duas datas.

SQL

SELECT DATEDIFF(NOW(), data\_cadastro) AS Dias\_Desde\_Cadastro FROM clientes;

**4. Funções Matemáticas**

Funções matemáticas para realizar cálculos numéricos.

**4.1. ABS() - Valor Absoluto**

Retorna o valor absoluto de um número (remove o sinal negativo).

sql

SELECT ABS(-15) AS Valor\_Absoluto;

* Retorna 15.

**4.2. CEIL() e FLOOR() - Arredondar para Cima ou Baixo**

Arredonda um número para o inteiro superior ou inferior.

SQL

SELECT CEIL(4.2) AS Arredonda\_Cima,

FLOOR(4.8) AS Arredonda\_Baixo;

* CEIL(4.2) retorna 5 (arredondando para cima), e FLOOR(4.8) retorna 4 (arredondando para baixo).

**4.3. ROUND() - Arredondar um Número**

Arredonda um número para um número específico de casas decimais.

SQL

SELECT ROUND(123.4567, 2) AS Valor\_Arredondado;

* Retorna 123.46, arredondando para duas casas decimais.

**4.4. MOD() - Resto da Divisão**

Retorna o resto de uma divisão entre dois números.

Sql

SELECT MOD(10, 3) AS Resto;

* Retorna 1, que é o resto da divisão de 10 por 3.

**5. Funções Condicionais**

**5.1. IF() - Avaliar Condições**

Retorna um valor se a condição for verdadeira, outro se for falsa.

SQL

SELECT nome, IF(sobrenome = 'Silva', 'Sim', 'Não') AS Sobrenome\_Silva

FROM clientes;

* Retorna "Sim" se o sobrenome for "Silva", e "Não" caso contrário.

**5.2. CASE - Estrutura Condicional**

Permite definir condições múltiplas.

SQL

SELECT c.nome,

CASE WHEN v.quantidade >= 10 THEN 'Grande Pedido'

WHEN v.quantidade BETWEEN 5 AND 9 THEN 'Pedido Médio'

ELSE 'Pequeno Pedido'

END AS Tipo\_Pedido

FROM vendas v

JOIN clientes c ON v.id\_cliente = c.id\_cliente

LIMIT 0, 25;