SQL Server - Comandos Essenciais

1. Tipos de Comandos SQL

DDL (Data Definition Language)

Comandos para **definir estruturas** do banco:

- `CREATE` - Criar objetos (banco, tabela)
- `ALTER` - Alterar estruturas existentes
- `DROP` - Excluir objetos

DML (Data Manipulation Language)

Comandos para **manipular dados**:

- `INSERT` - Inserir dados
- `UPDATE` - Atualizar dados
- `DELETE` - Excluir dados
- `SELECT` - Consultar dados

2. Criando Banco de Dados e Tabelas

Criar Banco de Dados

```
CREATE DATABASE Loja;
USE Loja;
```

Criar Tabelas

```
CREATE TABLE Clientes (
    id INT PRIMARY KEY IDENTITY(1,1),
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(150),
    data_nascimento DATE,
    ativo BIT DEFAULT 1
);

CREATE TABLE Pedidos (
    id INT PRIMARY KEY IDENTITY(1,1),
    cliente_id INT FOREIGN KEY REFERENCES Clientes(id),
    data_pedido DATETIME DEFAULT GETDATE(),
    valor_total DECIMAL(10,2)
);
```

Principais Tipos de Dados

- **INT**: Números inteiros
- **VARCHAR(n)**: Texto variável (até n caracteres)
- **DATE/DATETIME**: Data e hora
- **DECIMAL(p,s)**: Números decimais (p dígitos, s decimais)
- **BIT**: Booleano (0 ou 1)

3. Manipulando Dados

INSERT - Inserir Dados

```
INSERT INTO Clientes (nome, email, data_nascimento)
VALUES ('João Silva', 'joao@email.com', '1990-05-15');

INSERT INTO Pedidos (cliente_id, valor_total)
VALUES (1, 150.50);
```

UPDATE - Atualizar Dados

```
UPDATE Clientes
SET email = 'joao.silva@email.com'
WHERE id = 1;
```

ALTER - Modificar Estrutura

```
-- Adicionar coluna
ALTER TABLE Clientes ADD telefone VARCHAR(20);

-- Modificar coluna
ALTER TABLE Clientes ALTER COLUMN nome VARCHAR(200);
```

4. JOINS - Unindo Tabelas

INNER JOIN

Retorna apenas registros que existem em **ambas** as tabelas:

```
SELECT c.nome, p.data_pedido, p.valor_total
FROM Clientes c
INNER JOIN Pedidos p ON c.id = p.cliente_id;
```

LEFT JOIN

Retorna **todos** os registros da tabela à esquerda:

```
SELECT c.nome, p.valor_total
FROM Clientes c
LEFT JOIN Pedidos p ON c.id = p.cliente_id;
-- Mostra clientes mesmo sem pedidos
```

5. Exercício Prático

Passo 1: Criar as Tabelas

```
CREATE DATABASE Escola;
USE Escola;

CREATE TABLE Alunos (
    id INT PRIMARY KEY IDENTITY(1,1),
    nome VARCHAR(100) NOT NULL,
    idade INT,
    turma VARCHAR(10)
);

CREATE TABLE Notas (
    id INT PRIMARY KEY IDENTITY(1,1),
    aluno_id INT FOREIGN KEY REFERENCES Alunos(id),
    disciplina VARCHAR(50),
    nota DECIMAL(4,2)
);
```

Passo 2: Inserir Dados

```
-- Inserir alunos
INSERT INTO Alunos (nome, idade, turma) VALUES
('Ana Costa', 16, '3A'),
('Bruno Lima', 17, '3A'),
('Carlos Dias', 16, '3B'),
('Diana Silva', 17, '3B');

-- Inserir notas
INSERT INTO Notas (aluno_id, disciplina, nota) VALUES
(1, 'Matemática', 8.5),
(1, 'Português', 7.0),
(2, 'Matemática', 9.0),
(2, 'Português', 8.5),
(3, 'Matemática', 7.5),
(4, 'Matemática', 6.0);
```

Passo 3: Consultas e Relatórios

Relatório 1: Todos os alunos com suas notas

```
SELECT a.nome, a.turma, n.disciplina, n.nota
FROM Alunos a
LEFT JOIN Notas n ON a.id = n.aluno_id
ORDER BY a.nome, n.disciplina;
```

Relatório 2: Média de notas por aluno

```
SELECT a.nome, AVG(n.nota) as media
FROM Alunos a
INNER JOIN Notas n ON a.id = n.aluno_id
GROUP BY a.nome
HAVING AVG(n.nota) >= 7.0;
```

Relatório 3: Alunos sem notas cadastradas

```
SELECT a.nome, a.turma
FROM Alunos a
LEFT JOIN Notas n ON a.id = n.aluno_id
WHERE n.id IS NULL;
```