Manual Completo de JOINs no MySQL

Este material é um guia prático para estudar **JOINs no MySQL**, com:

1. **Criação das tabelas**
2. **Inserts**
3. **Manual de JOINs com explicações**
4. **Exemplos práticos (3 por tipo de JOIN)**
5. **Resumo final**

Este material contém, para ajudar na execução dos exemplos:

- Scripts de criação e inserção para montar o banco de dados.
- Explicações e exemplos práticos de todos os tipos de JOIN.
- Casos com diferentes contextos para melhor fixação.
- Resumo final com a comparação entre os JOINs.

PARTE 1 – Criação do Banco de Dados

```
CREATE TABLE clientes (
    id_cliente INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE pedidos (
    id_pedido INT PRIMARY KEY,
    id_cliente INT,
    data DATE,
    FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente)
);

CREATE TABLE produtos (
    id_produto INT PRIMARY KEY,
    nome VARCHAR(100),
    id_fornecedor INT
);

CREATE TABLE fornecedores (
    id_fornecedor INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE alunos (
    id_aluno INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE cursos (
    id_curso INT PRIMARY KEY,
```

```
    nome VARCHAR(100)
);

CREATE TABLE matriculas (
    id_matricula INT PRIMARY KEY,
    id_aluno INT,
    id_curso INT,
    FOREIGN KEY (id_aluno) REFERENCES alunos(id_aluno),
    FOREIGN KEY (id_curso) REFERENCES cursos(id_curso)
);

CREATE TABLE professores (
    id_professor INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE disciplinas (
    id_disciplina INT PRIMARY KEY,
    nome VARCHAR(100),
    id_professor INT,
    FOREIGN KEY (id_professor) REFERENCES professores(id_professor)
);

CREATE TABLE descontos (
    id_desconto INT PRIMARY KEY,
    id_produto INT,
    percentual DECIMAL(5,2),
    FOREIGN KEY (id_produto) REFERENCES produtos(id_produto)
);

CREATE TABLE departamentos (
    id_departamento INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE funcionarios (
    id_funcionario INT PRIMARY KEY,
    nome VARCHAR(100),
    id_departamento INT,
    FOREIGN KEY (id_departamento) REFERENCES departamentos(id_departamento)
);

CREATE TABLE estoque (
    id_produto INT PRIMARY KEY,
    quantidade INT
);

CREATE TABLE certificados (
    id_certificado INT PRIMARY KEY,
    id_aluno INT,
    nome_certificado VARCHAR(100),
    FOREIGN KEY (id_aluno) REFERENCES alunos(id_aluno)
);

CREATE TABLE tamanhos (
```

```
    id_tamanho INT PRIMARY KEY,
    tamanho VARCHAR(10)
);

CREATE TABLE cores (
    id_cor INT PRIMARY KEY,
    cor VARCHAR(20)
);

CREATE TABLE lojas (
    id_loja INT PRIMARY KEY,
    nome VARCHAR(100)
);

CREATE TABLE ingredientes (
    id_ingrediente INT PRIMARY KEY,
    nome VARCHAR(50)
);

CREATE TABLE pratos (
    id_prato INT PRIMARY KEY,
    nome VARCHAR(100)
);
```

PARTE 2 – Inserts de Exemplo

```
-- Clientes
INSERT INTO clientes VALUES (1, 'João');
INSERT INTO clientes VALUES (2, 'Maria');

-- Pedidos
INSERT INTO pedidos VALUES (1, 1, '2025-06-01');
INSERT INTO pedidos VALUES (2, 2, '2025-06-05');

-- Produtos e Fornecedores
INSERT INTO fornecedores VALUES (1, 'Fornecedor A');
INSERT INTO fornecedores VALUES (2, 'Fornecedor B');
INSERT INTO produtos VALUES (1, 'Mouse', 1);
INSERT INTO produtos VALUES (2, 'Teclado', 2);

-- Alunos, Cursos e Matrículas
INSERT INTO alunos VALUES (1, 'Carlos');
INSERT INTO alunos VALUES (2, 'Bruna');
INSERT INTO cursos VALUES (1, 'MySQL');
INSERT INTO cursos VALUES (2, 'HTML');
INSERT INTO matriculas VALUES (1, 1, 1);
INSERT INTO matriculas VALUES (2, 2, 2);

-- Professores e Disciplinas
INSERT INTO professores VALUES (1, 'Prof. Ana');
INSERT INTO professores VALUES (2, 'Prof. Beto');
```

```
INSERT INTO disciplinas VALUES (1, 'Lógica', 1);
-- Prof. Beto ainda não tem disciplina

-- Descontos
INSERT INTO descontos VALUES (1, 1, 10.00); -- Produto 1 com desconto

-- Departamentos e Funcionários
INSERT INTO departamentos VALUES (1, 'TI');
INSERT INTO departamentos VALUES (2, 'RH');
INSERT INTO funcionarios VALUES (1, 'Lucas', 1);

-- Estoque
INSERT INTO estoque VALUES (1, 50);
INSERT INTO estoque VALUES (2, 20);

-- Certificados
INSERT INTO certificados VALUES (1, 1, 'MySQL Expert');

-- Tamanhos e Cores
INSERT INTO tamanhos VALUES (1, 'P');
INSERT INTO tamanhos VALUES (2, 'M');
INSERT INTO cores VALUES (1, 'Azul');
INSERT INTO cores VALUES (2, 'Verde');

-- Lojas
INSERT INTO lojas VALUES (1, 'Loja A');
INSERT INTO lojas VALUES (2, 'Loja B');

-- Ingredientes e Pratos
INSERT INTO ingredientes VALUES (1, 'Tomate');
INSERT INTO ingredientes VALUES (2, 'Queijo');
INSERT INTO pratos VALUES (1, 'Pizza');
INSERT INTO pratos VALUES (2, 'Lasanha');
```

PARTE 3 – Manual Prático de JOINs

INNER JOIN

Definição: Retorna registros com correspondência em ambas as tabelas.

Exemplo 1: Clientes e Pedidos (IJ)

```
SELECT clientes.nome, pedidos.data
FROM clientes
INNER JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;
```

Exemplo 2: Produtos e Fornecedores

```sql
SELECT produtos.nome, fornecedores.nome AS fornecedor
FROM produtos
INNER JOIN fornecedores ON produtos.id_fornecedor = fornecedores.id_fornecedor;
```

Exemplo 3: Alunos e Cursos

```sql
SELECT alunos.nome, cursos.nome AS curso
FROM alunos
INNER JOIN matriculas ON alunos.id_aluno = matriculas.id_aluno
INNER JOIN cursos ON matriculas.id_curso = cursos.id_curso;
```

LEFT JOIN

Definição: Retorna todos os registros da tabela à esquerda, com correspondentes da direita (ou NULL se não houver).

Exemplo 1: Clientes e Pedidos (LJ)

```sql
SELECT clientes.nome, pedidos.data
FROM clientes
LEFT JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;
```

Exemplo 2: Professores e Disciplinas

```sql
SELECT professores.nome, disciplinas.nome AS disciplina
FROM professores
LEFT JOIN disciplinas ON professores.id_professor = disciplinas.id_professor;
```

Exemplo 3: Produtos e Descontos

```sql
SELECT produtos.nome, descontos.percentual
FROM produtos
LEFT JOIN descontos ON produtos.id_produto = descontos.id_produto;
```

RIGHT JOIN

Definição: Retorna todos os registros da tabela à direita, com correspondentes da esquerda (ou NULL se não houver).

Exemplo 1: Clientes e Pedidos (RJ)

```
SELECT clientes.nome, pedidos.data
FROM clientes
RIGHT JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;
```

Exemplo 2: Funcionários e Departamentos

```
SELECT funcionarios.nome, departamentos.nome AS departamento
FROM funcionarios
RIGHT JOIN departamentos ON funcionarios.id_departamento =
departamentos.id_departamento;
```

Exemplo 3: Produtos e Estoque

```
SELECT produtos.nome, estoque.quantidade
FROM produtos
RIGHT JOIN estoque ON produtos.id_produto = estoque.id_produto;
```

FULL OUTER JOIN (Simulado)

Definição: Retorna todos os registros de ambas as tabelas, com `NULL` onde não houver correspondência. **Não suportado nativamente no MySQL.**

Exemplo: Alunos e Certificados

```
SELECT alunos.nome, certificados.nome_certificado
FROM alunos
LEFT JOIN certificados ON alunos.id_aluno = certificados.id_aluno

UNION

SELECT alunos.nome, certificados.nome_certificado
FROM certificados
RIGHT JOIN alunos ON alunos.id_aluno = certificados.id_aluno;
```

CROSS JOIN

Definição: Produto cartesiano: cada linha da primeira tabela com todas da segunda.

Exemplo 1: Tamanhos e Cores

```sql
SELECT tamanhos.tamanho, cores.cor
FROM tamanhos
CROSS JOIN cores;
```

Exemplo 2: Produtos e Lojas

```sql
SELECT produtos.nome, lojas.nome AS loja
FROM produtos
CROSS JOIN lojas;
```

Exemplo 3: Ingredientes e Pratos

```sql
SELECT ingredientes.nome, pratos.nome AS prato
FROM ingredientes
CROSS JOIN pratos;
```

PARTE 4 – Resumo dos JOINs

Tipo de JOIN	O que faz	Quando usar
INNER JOIN	Traz só os dados que têm correspondência nas duas tabelas	Comparar dados que se relacionam diretamente
LEFT JOIN	Todos da esquerda, com dados da direita se houver	Ver o que "está faltando" na tabela da direita
RIGHT JOIN	Todos da direita, com dados da esquerda se houver	Ver o que "está faltando" na tabela da esquerda
FULL JOIN	Todos os dados de ambas as tabelas (mesmo sem correspondência)	MySQL: precisa simular com `LEFT JOIN UNION RIGHT JOIN`
CROSS JOIN	Todas as combinações possíveis entre as tabelas	Para gerar combinações, como variações de produto, etc.