Relacionamentos entre Tabelas com JOINs no SQL

Aula prática sobre INNER JOIN, LEFT JOIN, RIGHT JOIN e FULL JOIN

Professor: (Nome do Professor)

Disciplina: Administração de Banco de Dados

Data: (Inserir Data)

Objetivos da Aula

- Compreender os relacionamentos entre tabelas
- Conhecer os tipos de JOIN existentes
- Aplicar consultas SQL com JOINs
- Identificar registros faltantes ou inconsistentes

O que é um JOIN?

- JOIN permite combinar dados de duas ou mais tabelas com base em colunas relacionadas

- Utiliza-se principalmente para análise e cruzamento de dados

- A condição de junção geralmente é feita por campos chave (ex: id_titular)

Tipos de JOIN

- INNER JOIN – Apenas registros com correspondência nas duas tabelas

- LEFT JOIN – Todos os registros da tabela da esquerda + correspondentes da direita

- RIGHT JOIN – Todos os registros da tabela da direita + correspondentes da esquerda

- FULL JOIN – Todos os registros, com ou sem correspondência (simulado no MySQL)

Estrutura de Tabelas de Exemplo

- titulares: id_titular, nome
- dependentes: id_dependente, nome, id_titular
- parentesco: id_dependente, grau
- Observações:
  - Dados intencionalmente inconsistentes
  - Sem chaves estrangeiras declaradas

INNER JOIN

- Retorna apenas registros com correspondência nas duas tabelas

```
SELECT d.nome AS dependente, t.nome AS titular
FROM dependentes d
INNER JOIN titulares t ON d.id_titular = t.id_titular;
```

- Exibe apenas dependentes com titulares válidos

LEFT JOIN

- Retorna todos os dependentes, com ou sem titular

```
SELECT d.nome AS dependente, t.nome AS titular
FROM dependentes d
LEFT JOIN titulares t ON d.id_titular = t.id_titular;
```

- Dependentes sem titular aparecem com NULL

RIGHT JOIN

- Retorna todos os titulares, com ou sem dependente

```
SELECT d.nome AS dependente, t.nome AS titular
FROM dependentes d
RIGHT JOIN titulares t ON d.id_titular = t.id_titular;
```

- Titulares sem dependentes aparecem com NULL

FULL JOIN (Simulado)

- Retorna todos os registros das duas tabelas, com ou sem correspondência

MySQL não possui suporte nativo; utiliza-se UNION

```
SELECT d.nome AS dependente, t.nome AS titular
FROM dependentes d
LEFT JOIN titulares t ON d.id_titular = t.id_titular
UNION
SELECT d.nome AS dependente, t.nome AS titular
FROM dependentes d
RIGHT JOIN titulares t ON d.id_titular = t.id_titular;
```

Exercício 1 – INNER JOIN

- Objetivo: Listar apenas dependentes com titular existente

- Sugestão: Use INNER JOIN

Exercício 2 – LEFT JOIN

- Objetivo: Listar todos os dependentes, mesmo sem titular

- Sugestão: Use LEFT JOIN

Exercício 3 – RIGHT JOIN

- Objetivo: Listar todos os titulares, mesmo sem dependente

- Sugestão: Use RIGHT JOIN

Exercício 4 – FULL JOIN

- Objetivo: Unir todos os registros com ou sem relacionamento
- Sugestão: Simular FULL JOIN usando LEFT JOIN + RIGHT JOIN + UNION

Exemplos Reais de Uso de JOIN

- Clientes e pedidos
- Produtos e categorias
- Funcionários e dependentes
- Alunos e matrículas

Possíveis Problemas com JOINs

- Campos nulos ou ausentes
- Tipos de dados diferentes
- JOINs mal definidos causando repetições
- Performance baixa em joins com muitas tabelas

Boas Práticas com JOIN

- Verificar os dados antes de aplicar INNER JOIN
- Usar LEFT JOIN para descobrir registros órfãos
- Formatar NULLs com COALESCE()
- Utilizar apelidos (AS) para clareza no código

Conclusão da Aula

- JOINs são essenciais para análise relacional
- Dominar os tipos de JOIN aumenta a produtividade
- Teste com dados inconsistentes para melhor aprendizado

Próxima Aula

- Tópico: Subqueries e Views no MySQL

- Tarefa: Trazer exercícios resolvidos e dúvidas sobre JOINs