

transforme ■ se



JAVA

AULA 8 - Parte 2 - Manipulando Dados



Manipulando Dados



DML e DQL

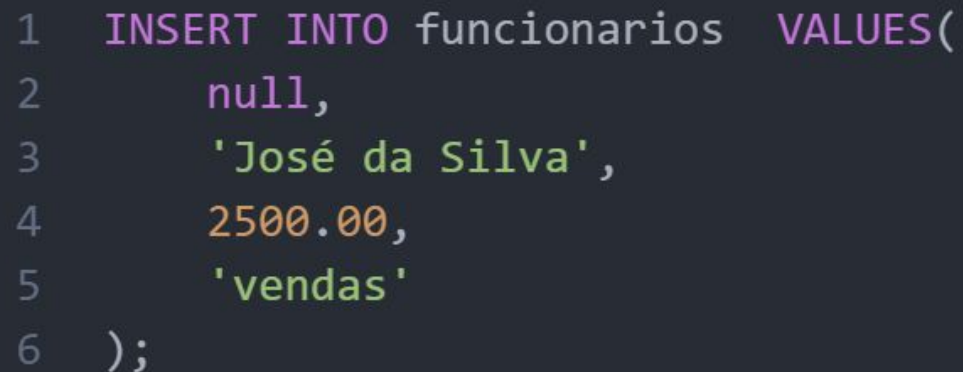
- Linguagem de Manipulação de Dados
 - INSERT: Insere dados
 - UPDATE: Altera dados
 - DELETE: Exclui dados
- Linguagem de Consulta de Dados
 - SELECT: Retorna dados
 - Ordenação de dados
 - Agrupamento de dados
 - Filtros de seleção
 - Funções aritméticas

Gerenciando e manipulando dados

INSERT INTO <Tabela> VALUES (Valores).

Em tradução poderíamos dizer: Insira dentro da tabela os valores (valores). Assim, passamos na ordem dos campos os valores para cada campo. Temos apenas de lembrar de passar o valor da chave primária como “null”, uma vez que ela será auto-incrementada.

No nosso exemplo da tabela funcionários poderíamos fazer dessa forma:



```
1  INSERT INTO funcionarios  VALUES(  
2      null,  
3      'José da Silva',  
4      2500.00,  
5      'vendas'  
6  );
```

Gerenciando e manipulando dados

Também poderíamos fazer também a inserção de valores em um ou mais campos específicos, para isso, podemos passar o nome dos campos antes dos valores, a notação seria assim:

INSERT INTO TABELA (CAMPOS) VALUES (VALORES)

No nosso exemplo da tabela funcionários:



```
1  INSERT INTO funcionarios (nome, salario, departamento)  VALUES(  
2      'João da Silva',  
3      4500.00,  
4      'diretoria'  
5  );
```

Gerenciando e manipulando dados

SELECT Campos FROM <Tabela>; (Selecione os campos x, y e z da tabela e retorne os dados)

SELECT * FROM <Tabela> (Selecione TODOS os campos da tabela e retorne)

O comando SELECT é utilizado para fazer consultas e retornar dados do banco. Podemos definir quais campos queremos retornar ou, utilizar o “ * ” que diz mysql que queremos retornar todos os campos de uma tabela.

Vamos fazer uma consulta e pedir para retornar apenas os valores dos campos nome e salário:



```
1 SELECT nome, salario FROM funcionarios;
```

	nome	salario
▶	José da Silva	2500
	José da Silva	2500
	José da Silva	2500
	João da Silva	4500

Gerenciando e manipulando dados

UPDATE <Tabela> SET Campo = Valor

O comando UPDATE 'seta' um valor para um campo específico da tabela. Poderíamos ler desta forma: "Atualize na tabela X o campo Y com o valor Z."

Um ponto de atenção aqui é que sem um filtro, uma alteração como a feita abaixo altera TODOS os valores de um campo numa tabela.



```
1 UPDATE funcionarios SET nome = 'Maria Souza';
```

	nome	salario
▶	Maria Souza	2500
	Maria Souza	2500
	Maria Souza	2500
	Maria Souza	4500

Gerenciando e manipulando dados

- DELETE FROM <Tabela>

O comando DELETE é utilizado para excluir um ou todos os registros de uma tabela. Aqui também temos que lembrar que sem um filtro, tudo será excluído. Veremos os filtros na sequência.



```
1  DELETE FROM funcionarios;
```

Filtros de Seleção



Filtros de Seleção

Registros selecionados (WHERE)

Para usarmos de maneira eficaz os comandos que vimos anteriormente precisamos do filtro Where. Esse filtro vai funcionar como se fosse um if do java, fazendo testes com operadores relacionais e caso a condição ou condições passadas sejam verdadeiras, os registros que atenderem à condição serão retornados, alterados ou deletados.

Os Operadores relacionais são:

- Igual (=), Diferente (!=)
- Maior (>), Maior ou igual (>=)
- Menor (<), Menor ou igual (<=)
- Nulo (IS NULL), ou não-nulo (IS NOT NULL)
- Entre intervalo (BETWEEN)
- Valor parcial (LIKE)

Os operadores lógicos são:

- AND
- OR
- NOT

Consultando dados com filtros

Agora podemos aplicar filtros ao nosso SELECT. Primeiro, na imagem à esquerda podemos ver um SELECT sem filtro algum, que vai trazer todos os registros. Na imagem à direita, vamos aplicar um filtro, vamos trazer apenas o registro que tem no campo nome o valor “Maria Souza”, a sintaxe genérica para o uso dos filtros podemos ver abaixo.

SELECT Campos FROM <Tabela> WHERE Condição

	id	nome	salario	departamento
	1	Madalena	2500	vendas
	2	Ana	1500	vendas
	3	Marta	3000	vendas
▶	4	Maria Souza	4500	diretoria

	id	nome	salario	departamento
▶	4	Maria Souza	4500	diretoria
	NULL	NULL	NULL	NULL

O comando usado:



```
1 SELECT * FROM funcionarios WHERE nome = "Maria Souza";
```

Consultando dados com filtros

Vamos atualizar o salário dos funcionários que ganham menos de R\$3000,00. O comando Update vai ser combinado com o WHERE para isso.

UPDATE Tabela SET Campo = Valor WHERE Condição

Antes de aplicar o UPDATE:

	id	nome	salario	departamento
	1	Madalena	2500	vendas
	2	Ana	1500	vendas
▶	3	Marta	3000	vendas
	4	Maria Souza	4500	diretoria

Após aplicar o UPDATE:

	id	nome	salario	departamento
▶	1	Madalena	2750	vendas
	2	Ana	1650	vendas
	3	Marta	3000	vendas
	4	Maria Souza	4500	diretoria

O comando usado:

Aplicamos uma atualização no valor do salário com um aumento de 10% para quem tem salário inferior a R\$3000,00



```
1 UPDATE funcionarios SET salario = salario + (salario * (10/100)) WHERE salario < 3000;
```

Consultando dados com filtros

Vamos agora deletar um funcionário da tabela.

DELETE FROM Tabela WHERE Condição

Antes de aplicar o DELETE:

	id	nome	salario	departamento
	1	Madalena	2500	vendas
	2	Ana	1500	vendas
▶	3	Marta	3000	vendas
	4	Maria Souza	4500	diretoria

Após aplicar o UPDADELETETE:

	id	nome	salario	departamento
▶	1	Madalena	2750	vendas
	2	Ana	1650	vendas
	4	Maria Souza	4500	diretoria

O comando usado:

Deletamos o funcionário cujo id era igual a 3.



```
1 DELETE FROM funcionarios WHERE id = 3;
```

Atributos especiais



Atributos especiais

Podemos utilizar apelidos para nossas tabelas e campos de modo a facilitar na hora de escrever nosso sql. Para apelidar uma tabela fazemos assim:

SELECT Campos FROM Tabela Apelido ...



```
1  SELECT f.nome FROM funcionarios f WHERE f.nome = 'Madalena';
```

Apelidamos nossa tabela funcionarios de 'f' apenas. Isso pode ajudar muito quando precisamos escrever uma consulta muito grande. Quando usamos apelidos, precisamos, ao referenciar algum campo chamar o apelido.nome_do_campo, como no exemplo: f.nome.

	nome
▶	Madalena

Atributos especiais

Podemos utilizar apelidos para nossas tabelas e campos de modo a facilitar na hora de escrever nosso sql. Para apelidar um campo fazemos assim:

SELECT Campo AS Apelido FROM ...



```
1  SELECT nome as primeiro_nome FROM funcionarios;
```

Apelidamos o campo nome como primeiro_nome. O retorno do SELECT vai trazer esse apelido ao invés do nome normal do campo. Também é bastante útil quando estamos trabalhando com consultas envolvendo tabelas que contenham nomes de campos iguais.

	primeiro_nome
▶	Madalena
	Ana
	Maria Souza

transforme ■ se

O conhecimento é o poder
de transformar o seu futuro.