Banco de Dados

Aula 07: Comandos INSERT, UPDATE, DELETE E SELECT

Comandos DML:

- INSERT
- UPDATE
- DELETE
- SELECT

SQL

- A linguagem SQL é basicamente dividida em três tipos de comandos:
- SQL = DDL + DML + DCL
- DDL (definição de dados)
- Comandos: CREATE, DROP, ALTER
- DML (manipulação de dados)
- Comandos: SELECT, INSERT, UPDATE e DELETE
- DCL (controle de dados)
- Comandos: GRANT e REVOKE

Comandos DML

- Os comandos de manipulação de dados (DML) em SQL são representados por:
- INSERT: permite a inclusão de novas linhas nas tabelas
- UPDATE: altera os valores de dados já cadastrados
- DELETE: remove dados já cadastrados
- SELECT: usado para consultar o BD e retornar dados que satisfazem a determinada expressão em um comando

Comando INSERT

- O comando INSERT permite inserir uma linha de dados na tabela e possui a seguinte sintaxe abaixo:

```
INSERT INTO NOME DA TABELA(coluna1,coluna2,coluna3)
VALUES (valor1, valor2, valor3);
```

- **Exemplos:**

```
INSERT INTO Cliente(codigo,nome,sexo)
VALUES("200810", "Regilan Meira", "Masculino");

INSERT INTO Disciplina(codigo,nome,ementa)
VALUES("01", "Banco de Dados", "DER,Modelo
Relacional,SQL");
```

Comando UPDATE

- O comando **`UPDATE`** é usado para mudar valores de linhas de dados que já foram cadastrados anteriormente e que obedecem a determinados critérios, especificados em condições.
- Este comando pode alterar mais de uma linha ao mesmo tempo, caso mais de uma linha obedeça a determinada condição.
- As condições podem também ser representadas utilizando os operadores: **`AND`**, **`OR`** e **`NOT`**

Comando UPDATE

- O comando **UPDATE**, contém a cláusula **WHERE**, de forma a restringir o conjunto dos registros que serão processados pelo comando.
- Se não for colocada a cláusula **WHERE** no comando **UPDATE**, as alterações serão realizadas em todos os registros da tabela.

Comando UPDATE

- Sintaxe:

```
UPDATE NOME DA TABELA
SET coluna1 = valor1, coluna2 = valor2
WHERE condições;
```

- Exemplos:

```
UPDATE Avaliacao SET media = 10;
```

```
UPDATE  Avaliacao SET media = 10
WHERE nome_aluno = "João";
```

```
UPDATE Compras SET preco = 105, forma_pagamento =
"Cartão de Crédito"
WHERE numero_compra = "2008708";
```

Comando UPDATE

- **Situação 01: Aumentar o salário de todos os funcionários em 10%**

- Como se pretende aumentar o salário de todos os elementos da tabela FUNCIONÁRIO, o comando **`UPDATE`** não usará a cláusula **`WHERE`**.

```
UPDATE Funcionario SET salario = salario * 1.1;
```

Comando UPDATE

- **Situação 02: Aumentar o salário do funcionário Regilan Meira e adicionar 1 ano ao tempo de serviço.**

- Nessa situação, estamos restringindo a atualização para o funcionário REGILAN MEIRA, sendo assim faz-se necessário o uso da cláusula WHERE.

```
UPDATE Funcionario SET salario = salario * 1.1, idade =
idade + 1
WHERE nome = "Regilan Meira";
```

Comando UPDATE

- **Situação 03: Adicionar o prefixo 55 ao telefone de todos os hospedes que residem no Brasil**

- Nessa situação, estamos restringindo a atualização para indivíduos **Brasileiros**, sendo assim faz-se necessário o uso da cláusula WHERE.

```
UPDATE Hospedes SET Telefone = "55" + Telefone
WHERE pais = "Brasil";
```

Comando UPDATE

- **Situação 04: Adicionar R$ 150 no salário das mulheres que possuem filhos.**

- Nessa situação, estamos restringindo a atualização dos dados para duas condições. Sendo assim, utilizaremos a cláusula **WHERE** e o operador **AND**.

```
UPDATE Funcionarios SET Salario = Salario + 150
WHERE Sexo = "F" and Filhos > 0;
```

Comando UPDATE

- **Situação 05: Adicionar R$ 150 no salário das mulheres que possuem filhos, ou homens que são casados.**

- Nessa situação, utilizaremos a cláusula **WHERE**, juntamente com o operador **AND e OR**.

```
UPDATE Funcionarios SET Salario = Salario + 150
WHERE (Sexo = "F" and Filhos > 0) OR (Sexo = "M" and
EstadoCivil = "Casado");
```

Comando UPDATE

- **Situação 06: Conceder desconto de 5% nos preços dos veículos que possuírem cor diferente de preto e branco.**

- Nessa situação, utilizaremos a cláusula **WHERE**, juntamente com o operador **AND**.

```
UPDATE Veiculos SET Preco = Preco – Preço * 0.05
WHERE Cor <> “Branco” AND Cor <> “Preto”;
```

Comando UPDATE

- **Situação 07: Conceder desconto de 5% nos preços dos produtos à base de leite.**
- Nessa situação, utilizaremos o operador LIKE.
- O operador LIKE permite fazer comparações de partes da string. Para isso utilizaremos dois curingas (“ % “)

```
UPDATE Produto SET Preco = Preco – Preço * 0.05
WHERE Nome Like “Leite%”;
```

Comando DELETE

- O comando **DELETE** é usado para remover linhas de uma tabela. Este comando pode remover mais de uma linha ao mesmo tempo, caso mais de uma linha obedeça a uma certa condição.

- As condições podem ser representadas utilizando os operadores **AND**, **OR** e **NOT**

Comando DELETE

- O comando **DELETE**, contém a cláusula **WHERE**, de forma a restringir o conjunto dos registros que serão processados pelo comando.
- Se não for colocada a cláusula **WHERE** no comando **DELETE**, serão apagados todos os registros de uma tabela.
- Assim como no comando **UPDATE**, podemos utilizar os operadores relacionais (>, >=, <,<=, =, <>, like) e os operadores lógicos(**AND**, **OR**) para especificar as condições de exclusão de dados.

Comando DELETE

- Sintaxe:

```
DELETE FROM NOME DA TABELA
WHERE <condições>;
```

- Exemplos:

```
DELETE FROM ESCOLA;

DELETE FROM ESCOLA
WHERE ALUNO = "TIAGO PEREIRA";

DELETE FROM PRODUTOS
WHERE NOME Like "LEITE%";

DELETE FROM CLIENTES
WHERE QuantidadeCompras <= 3;
```

Comando SELECT

- O comando SELECT é usado para consultar o banco de dados e retornar dados recuperados que satisfazem a determinada condição expressa no comando.

- **Sua sintaxe é representada da seguinte forma:**

```
SELECT <lista de atributos>
FROM NOME DA TABELA
WHERE <condições>;
```

Comando SELECT

- Exemplos:

```
SELECT codigo, aluno, media FROM NOTAS
WHERE aluno = "Tiago";

SELECT matricula, nome,responsavel,data_nascimento,cpf,rg,
endereco,codigo_curso,observacoes FROM ALUNOS
WHERE nome = "Camila";

SELECT * FROM ALUNOS
WHERE nome = "Camila";
```

- Obs: o símbolo * na clausula SELECT indica que deverá ser selecionado todos os campos de uma tabela.

Comando SELECT

- Situação 01: Escrever o comando SQL que permite obter o RG, Nome e o Código Postal de todos os clientes registrados no banco de dados.

```
SELECT Rg, Nome, CodigoPostal
FROM Cliente;
```

Comando SELECT

- **Situação 02: Selecionar todos os dados de todos os pacientes cadastrados no Hospital**

- Nesta situação usaremos o curinga (*) , ao invés de escrever todos os campos da tabela Paciente no comando SQL.

```
SELECT * FROM Pacientes;
```

Comando SELECT

- **Situação 03: Selecionar o ID, Nome, Idade e o Salário de todos os Funcionários com Idade entre 30 e 40 anos**

- Nesta situação usaremos o operador WHERE, juntamente com operadores lógicos e relacionais.

```
SELECT Id,Nome,Idade,Salario
FROM Funcionario
Where Idade >= 30 AND Idade <= 40;
```

Comando SELECT

- **Situação 04: Selecionar o ID, Nome, Idade e o Salário de todos os Funcionários cuja a idade não está entre 30 e 40 anos.**

- Nesta situação usaremos o operador WHERE, juntamente com operadores lógicos (AND e NOT) e relacionais.

```
SELECT Id,Nome,Idade,Salario
FROM Funcionario
Where Not(Idade >= 30 AND Idade <= 40);
```

Comando SELECT

- **Situação 05: Selecionar todos os indivíduos que possuem sobrenome "Silva"**

- Nesta situação usaremos o operador Like e o coringa "%"

```
SELECT * FROM Pessoa
Where Nome Like "%Silva%";
```

Comando SELECT

- Situação 06: Selecionar a quantidade de votos dos Partidos: PP, PSDB, PMDB e PSB nas eleições municipais de 2016.

```
SELECT QuantidadeVotos
FROM Votos
Where (Partido = “PP” OR Partido = “PSDB” OR Partido =
“PMDB” OR Partido = “PSB”) AND AnoEleicao = 2020;
```