Comandos SQL em MySQL: DDL, DML, DQL, DCL e TCL

O SQL (Structured Query Language) é dividido em várias sublinguagens, cada uma com um propósito específico no gerenciamento de bancos de dados. Abaixo, detalhamos cada uma delas com exemplos práticos usando MySQL.

DDL (Data Definition Language)

DDL é usada para **definir, modificar e excluir a estrutura** do banco de dados, incluindo esquemas, tabelas, índices e outras estruturas.

Criação de Estruturas

- **Criar um Banco de Dados:** Inicia um novo espaço para armazenar dados.

```
CREATE DATABASE minha_loja;
```

- **Usar um Banco de Dados:** Seleciona o banco de dados ativo para as próximas operações.

```
USE minha_loja;
```

- **Criar uma Tabela:** Define as colunas e tipos de dados que uma tabela irá conter.

```
CREATE TABLE produtos (
    id INT PRIMARY KEY AUTO_INCREMENT, -- Chave primária que incrementa
automaticamente
    nome VARCHAR(100) NOT NULL,        -- Nome do produto, não pode ser nulo
    preco DECIMAL(10, 2),              -- Preço com 2 casas decimais
    estoque INT DEFAULT 0              -- Quantidade em estoque, padrão 0
);
```

Modificação de Estruturas

- **Modificar uma Tabela (Adicionar uma Coluna):** Inclui uma nova coluna na tabela existente.

```
ALTER TABLE produtos
ADD COLUMN descricao TEXT; -- Adiciona uma coluna 'descricao' do tipo TEXT
```

- **Modificar uma Tabela (Modificar Tipo de Coluna):** Altera o tipo de dado ou outras propriedades de uma coluna existente.

```
ALTER TABLE produtos
MODIFY COLUMN preco DECIMAL(12, 2); -- Altera o 'preco' para ter 12 dígitos
no total, 2 após a vírgula
```

- **Modificar uma Tabela (Renomear uma Coluna):** Muda o nome de uma coluna existente.

```
ALTER TABLE produtos
CHANGE COLUMN nome nome_produto VARCHAR(100); -- Renomeia 'nome' para
'nome_produto'
```

Exclusão de Estruturas

- **Remover uma Coluna:** Exclui uma coluna específica de uma tabela.

```
ALTER TABLE produtos
DROP COLUMN descricao; -- Remove a coluna 'descricao'
```

- **Remover uma Tabela:** Deleta a tabela inteira e todos os seus dados.

```
DROP TABLE produtos;
```

- **Remover um Banco de Dados:** Exclui o banco de dados completo e todas as suas tabelas e dados.

```
DROP DATABASE minha_loja;
```

DML (Data Manipulation Language)

DML é utilizada para **inserir, atualizar e excluir dados** nas tabelas. Ela lida com o conteúdo das estruturas de dados.

Inserção de Dados

- **Inserir Dados em uma Tabela:** Adiciona novas linhas à tabela.

```
INSERT INTO produtos (nome, preco, estoque)
VALUES ('Camiseta', 29.99, 150);
```

É possível inserir múltiplas linhas em uma única declaração:

```
INSERT INTO produtos (nome, preco, estoque)
VALUES ('Calça Jeans', 89.90, 80),
       ('Tênis', 150.00, 50);
```

Atualização de Dados

- **Atualizar Dados Existentes:** Modifica os valores de linhas já existentes na tabela.

```
UPDATE produtos
SET preco = 35.00          -- Define o novo preço
WHERE nome = 'Camiseta';   -- Para produtos com nome 'Camiseta'

UPDATE produtos
SET estoque = estoque - 10 -- Reduz o estoque em 10 unidades
WHERE id = 2;              -- Para o produto com ID 2
```

Exclusão de Dados

- **Excluir Dados de uma Tabela:** Remove linhas da tabela.

```
DELETE FROM produtos
WHERE estoque = 0; -- Remove todos os produtos com estoque igual a zero

DELETE FROM produtos
WHERE id = 3;      -- Remove o produto com ID 3
```

DQL (Data Query Language)

DQL é usada exclusivamente para **consultar e recuperar dados** de tabelas. Embora frequentemente agrupada com DML, sua função de apenas leitura a distingue.

- **Selecionar Todos os Dados de uma Tabela:** Recupera todas as colunas de todas as linhas.

```
SELECT *
FROM produtos;
```

- **Selecionar Colunas Específicas:** Recupera apenas as colunas desejadas.

```
SELECT nome, preco
FROM produtos;
```

- **Selecionar Dados com Condição (Filtragem):** Recupera dados que satisfazem uma condição específica.

```
SELECT nome, estoque
FROM produtos
WHERE preco > 50.00; -- Produtos com preço maior que 50
```

- **Selecionar e Ordenar Dados:** Organiza os resultados da consulta.

```
SELECT nome, preco
FROM produtos
ORDER BY preco DESC; -- Ordena por preço em ordem decrescente
```

- **Contar Registros:** Retorna o número de linhas que correspondem a uma condição.

```
SELECT COUNT(*) AS total_produtos
FROM produtos; -- Conta o total de produtos
```

- **Agrupar Dados:** Agrega dados com base em uma ou mais colunas.

```
SELECT preco, COUNT(id) AS quantidade_por_preco
FROM produtos
GROUP BY preco; -- Conta produtos por cada preço
```

DCL (Data Control Language)

DCL é utilizada para **controlar acessos e permissões** dentro do banco de dados, garantindo a segurança.

- **Criar um Usuário:** Define uma nova conta de usuário para o MySQL.

```
CREATE USER 'novo_usuario'@'localhost' IDENTIFIED BY 'senha_forte_123';
```

`'localhost'` indica que o usuário só pode se conectar a partir do próprio servidor.

- **Conceder Permissões a um Usuário:** Atribui privilégios específicos a um usuário.

```
GRANT SELECT, INSERT ON minha_loja.produtos TO 'novo_usuario'@'localhost';
```

Isso permite que 'novo_usuario' apenas selecione e insira dados na tabela `produtos` do banco de dados `minha_loja`.

- **Revogar Permissões de um Usuário:** Remove privilégios concedidos anteriormente.

```
REVOKE INSERT ON minha_loja.produtos FROM 'novo_usuario'@'localhost';
```

Isso remove a permissão de inserção de dados da tabela `produtos` para 'novo_usuario'.

- **Remover um Usuário:** Exclui uma conta de usuário do MySQL.

```
DROP USER 'novo_usuario'@'localhost';
```

TCL (Transaction Control Language)

TCL é usada para **gerenciar transações**, garantindo a atomicidade, consistência, isolamento e durabilidade (ACID) das operações no banco de dados.

- **Iniciar uma Transação:** Marca o início de uma nova transação.

```
START TRANSACTION;
-- Ou: BEGIN;
-- Ou: BEGIN WORK;
```

- **Confirmar uma Transação:** Torna permanentes todas as mudanças feitas desde o `START TRANSACTION`.

```
START TRANSACTION;

-- Operação 1: Reduz o estoque de um produto
UPDATE produtos SET estoque = estoque - 1 WHERE id = 1;

-- Operação 2: Registra a venda
INSERT INTO vendas (produto_id, quantidade) VALUES (1, 1);

COMMIT; -- Confirma ambas as operações juntas
```

- **Reverter uma Transação:** Desfaz todas as mudanças feitas desde o `START TRANSACTION`, retornando o banco de dados ao estado anterior.

```
START TRANSACTION;

-- Tenta reduzir o estoque por um valor muito alto (ex: 100)
UPDATE produtos SET estoque = estoque - 100 WHERE id = 1;
```

```
-- Se ocorrer um erro ou a lógica indicar que a operação é inválida
ROLLBACK; -- Desfaz a atualização de estoque
```

- **Definir Savepoints (Pontos de Retorno):** Cria pontos dentro de uma transação para os quais você pode reverter, sem desfazer toda a transação.

```
START TRANSACTION;

INSERT INTO produtos (nome, preco, estoque) VALUES ('Fone de Ouvido', 79.99,
200);

SAVEPOINT depois_fone; -- Cria um ponto de retorno aqui

-- Tenta uma atualização que pode não ser desejada
UPDATE produtos SET preco = 85.00 WHERE nome = 'Fone de Ouvido';

-- Se decidir que a atualização não é boa, mas a inserção sim
ROLLBACK TO SAVEPOINT depois_fone; -- Reverte apenas para o ponto
'depois_fone'

COMMIT; -- Confirma a inserção do 'Fone de Ouvido' (mas não a alteração de
preço)
```