

MySQL – CREATE, SHOW, USE, DESCRIBE, DROP, ALTER, CONSTRAINT e INSERT

Ewerton J. Silva, BingChat e Chat GPT

MySQL

MySQL é um sistema de gerenciamento de banco de dados relacional de código aberto. Ele é usado para armazenar e gerenciar dados em um servidor. O MySQL é amplamente utilizado em aplicativos da web e é uma das tecnologias de banco de dados mais populares do mundo.

Os comandos SQL usados para manipular informações de um banco de dados podem ser classificados em comandos relacionados à manipulação (DML), à transação (DTL), à definição (DDL) e ao controle (DCL). Alguns autores incluem seleção ou select nos comandos DML, outros determinam uma nova classificação (DQL).

Classificação dos comandos

- DML é um nome abreviado da Data Manipulation Language que lida com a manipulação de dados e inclui as instruções SQL mais comuns, como SELECT, INSERT, UPDATE, DELETE, etc., e é usado para armazenar, modificar, recuperar, excluir e atualizar dados no banco de dados.
- DTL significa Data Transaction Language e é uma linguagem de transação de dados que lida com transações em um banco de dados.
- DDL significa Data Definition Language e é uma linguagem de definição de dados que lida com esquemas de banco de dados e estruturas. Ele define como os dados são armazenados e acessados no banco de dados. Os comandos DDL mais comuns são: CREATE TABLE, DROP TABLE, ALTER TABLE e TRUNCATE.

Classificação dos comandos

- Os comandos DCL (Data Control Language) são usados para controlar o acesso ao banco de dados. Os comandos mais utilizados são:
GRANT: concede privilégios a um usuário do banco de dados e
REVOKE: remove privilégios de um usuário do banco de dados.
- Os comandos DQL (Data Query Language) são usados para recuperar dados do banco de dados. Um exemplo de comando é SELECT que recupera dados do banco de dados.

Comandos MySQL

Os principais comandos MySQL são aqueles que permitem criar, manipular e consultar dados em um banco de dados relacional:

- CREATE DATABASE: cria um novo banco de dados no servidor MySQL.
- SHOW DATABASES: mostra todos os bancos de dados existentes no servidor MySQL.
- USE: seleciona um banco de dados para usar nos comandos seguintes.
- CREATE TABLE: cria uma nova tabela em um banco de dados especificado.
- SHOW TABLES: mostra todas as tabelas de um banco de dados selecionado.
- DESCRIBE: mostra a estrutura e os tipos de dados de uma tabela.
- INSERT: insere novos registros em uma tabela conforme os argumentos passados.
- SELECT: busca registros em uma tabela de acordo com um critério definido na cláusula WHERE.
- UPDATE: atualiza registros em uma tabela conforme os valores e o critério passados.
- DELETE: remove registros em uma tabela conforme o critério passado.

CREATE DATABASE

O comando CREATE DATABASE em MySQL serve para criar um novo banco de dados no servidor. A sintaxe básica é:

```
CREATE DATABASE nome_do_banco;
```

Onde nome_do_banco é o nome que você quer dar ao seu banco de dados. Por exemplo, se você quiser criar um banco de dados chamado “escola”, você pode usar o seguinte comando:

```
CREATE DATABASE escola;
```

CREATE DATABASE

- Criar um banco de dados chamado “clientes”:

```
CREATE DATABASE clientes;
```

- Criar um banco de dados chamado “supermercado”:

```
CREATE DATABASE supermercado;
```

- Criar um banco de dados chamado “locadora”:

```
CREATE DATABASE locadora;
```

SHOW DATABASES

O comando SHOW DATABASES em MySQL serve para listar os bancos de dados existentes no servidor. A sintaxe básica é:

```
SHOW DATABASES;
```

Esse comando retorna uma tabela com uma coluna chamada Database, que contém os nomes dos bancos de dados.

SHOW DATABASES

1 SHOW DATABASES;

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
Database				
▶	information_schema			
	s222_allan12			
	s222_amanda13			
	s222_bruno14			
	s222_camila15			
	s222_eduardo16			
	s222_ewerton87			
	s222_gabriel17			
	s222_gabriel18			
	s222_gabriell19			

SHOW DATABASES

Você também pode usar o comando SHOW DATABASES com o operador LIKE para filtrar os bancos de dados por um padrão. A sintaxe é:

```
SHOW DATABASES LIKE 'padrão';
```

Onde padrão é uma string que pode conter caracteres curinga como % (que representa qualquer sequência de caracteres) e _ (que representa qualquer caractere único). Por exemplo, se você quiser listar apenas os bancos de dados que começam com a letra m, você pode usar o seguinte comando2:

```
SHOW DATABASES LIKE 'm%';
```

USE

O comando USE em MySQL serve para selecionar um banco de dados para trabalhar. A sintaxe básica é:

```
USE nome_do_banco;
```

Onde nome_do_banco é o nome do banco de dados que você quer usar. Por exemplo, se você quiser selecionar o banco de dados “escola”, você pode usar o seguinte comando:

```
USE escola;
```

USE

Depois de selecionar um banco de dados, você pode executar outros comandos como CREATE TABLE, INSERT, SELECT, etc. no banco de dados selecionado. Alguns exemplos de uso do comando USE em MySQL são:

- Selecionar o banco de dados “aniversarios”:

USE aniversarios;

- Selecionar o banco de dados “Supermercado”:

USE Supermercado;

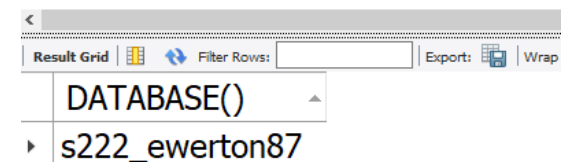
SELECT DATABASE

Para ver qual banco de dados está selecionado em MySQL, você pode usar o seguinte comando:

```
SELECT DATABASE();
```

Esse comando retorna o nome do banco de dados atual ou NULL se nenhum banco de dados estiver selecionado.

```
1  SELECT DATABASE();  
2
```



The screenshot shows a MySQL query result window. At the top, there is a toolbar with icons for 'Result Grid', 'Filter Rows', 'Export', and 'Wrap'. Below the toolbar, the query result is displayed in a table format. The first row is the column header 'DATABASE()', and the second row is the result value 's222_ewerton87'.

DATABASE()
s222_ewerton87

ALTER SCHEMA

Para alterar o nome de um banco de dados em MySQL, você pode usar o comando `ALTER SCHEMA`, para isso você precisa especificar o nome atual e o novo nome do banco de dados. Por exemplo, para renomear um banco de dados chamado `bd_etec` para `bd_novo`, você pode usar o seguinte comando:

```
ALTER SCHEMA bd_etec RENAME TO bd_novo;
```

Depois de executar esse comando, você pode usar o comando `SHOW DATABASES` para verificar se o nome do banco de dados foi alterado.

DROP DATABASE

Para deletar um banco de dados em MySQL, você pode usar o comando `DROP DATABASE`. Esse comando apaga o banco de dados e todas as tabelas que ele contém. Por exemplo, para deletar um banco de dados chamado `bd_etec`, você pode usar o seguinte comando:

```
DROP DATABASE bd_etec;
```

Antes de executar esse comando, certifique-se de que você está conectado ao MySQL com um usuário que tenha permissão para deletar bancos de dados e que você não precisa mais dos dados do banco de dados que vai apagar. Depois de executar o comando, você pode usar o comando `SHOW DATABASES` para verificar se o banco de dados foi realmente excluído.

CREATE TABLE

O comando CREATE TABLE permite adicionar uma tabela no banco de dados. A sintaxe básica do comando é:

```
CREATE TABLE [IF NOT EXISTS] nome_tabela (  
    nome_coluna tipo_dados constraint  
);
```


SINTAXE BÁSICA

- CREATE TABLE cria a tabela no banco de dados.
- IF NOT EXISTS é um comando opcional que verifica se a tabela não existe no banco antes de criá-la.
- nome_tabela é o nome da tabela que você quer criar.
- nome_coluna é o nome de cada coluna da tabela.
- tipo_dados é o tipo de dados que cada coluna pode armazenar (por exemplo, int, varchar, date, etc.).
- constraint é uma restrição opcional que define regras para os valores das colunas (por exemplo, not null, primary key, auto_increment, etc.).

Exemplos de uso do comando create table

- Criar uma tabela chamada livros com os campos id, titulo, autor e preco

```
CREATE TABLE livros (  
    id int not null auto_increment,  
    titulo varchar(100) not null,  
    autor varchar(50) not null,  
    preco decimal(10,2) not null,  
    primary key (id)  
);
```

Exemplos de uso do comando create table

- Criar uma tabela chamada clientes com os campos codigo, nome, data_nascimento e telefone

```
CREATE TABLE clientes (  
    codigo int not null auto_increment,  
    nome varchar(60) not null,  
    data_nascimento date,  
    telefone char(8),  
    primary key (codigo)  
);
```

SHOW TABLES

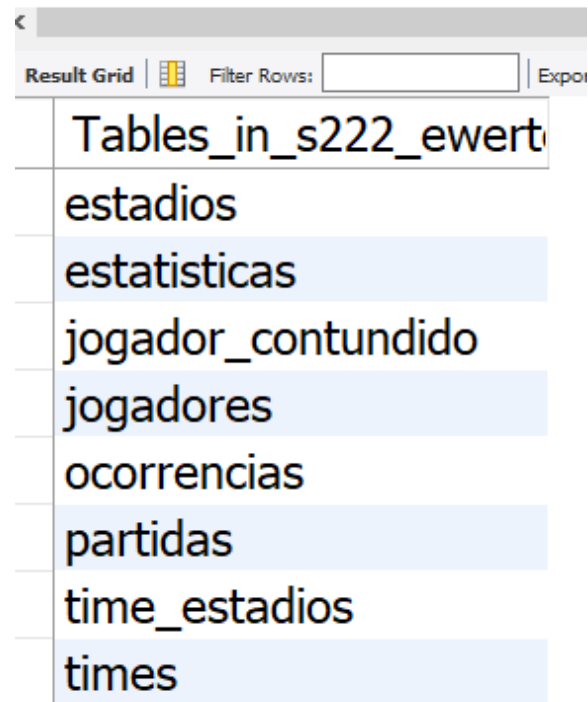
O comando SHOW TABLES em MySQL é usado para listar as tabelas de um banco de dados específico. Para usar esse comando, você precisa primeiro selecionar o banco de dados que deseja usar com o comando USE nome_do_banco_de_dados. Depois disso, você pode digitar SHOW TABLES para ver todas as tabelas do banco de dados selecionado. Por exemplo:

```
USE bd_etec; SHOW TABLES;
```

Isso retornará uma lista das tabelas no banco de dados bd_etec.

SHOW TABLES

```
1  SHOW TABLES;  
2
```



The screenshot shows a database query result interface. At the top, there is a header bar with a back arrow, the text "Result Grid", a grid icon, a "Filter Rows:" input field, and an "Export" button. Below this, a table lists the results of the "SHOW TABLES" query. The table has a single column containing the names of the tables in the database. The tables listed are: "Tables_in_s222_ewert", "estadios", "estadisticas", "jogador_contundido", "jogadores", "ocorrencias", "partidas", "time_estadios", and "times". The rows for "estadisticas", "jogadores", "partidas", and "times" are highlighted with a light blue background.

Tables_in_s222_ewert
estadios
estadisticas
jogador_contundido
jogadores
ocorrencias
partidas
time_estadios
times

RENAME TABLE

Para modificar o nome de uma tabela em mysql, você pode usar o comando RENAME TABLE

A sintaxe básica do comando é:

```
RENAME TABLE nome_tabela_atual TO nome_tabela_novo;
```

- nome_tabela_atual é o nome da tabela que você quer renomear.
- nome_tabela_novo é o novo nome que você quer dar para a tabela.

Exemplo de uso do comando RENAME TABLE

- Renomear a tabela clientes para meus_clientes

```
RENAME TABLE clientes TO meus_clientes;
```

DESCRIBE

O comando DESCRIBE em MySQL é usado para ver a estrutura de uma tabela, incluindo os nomes dos campos, os tipos de dados e outras informações. Para usar esse comando, você precisa primeiro selecionar o banco de dados que contém a tabela que deseja descrever com o comando `USE nome_do_banco_de_dados`. Depois disso, você pode digitar `DESCRIBE nome_da_tabela` para ver os detalhes da tabela. Por exemplo:

```
USE bd_etec; DESCRIBE tb01_alunos;
```


DESCRIBE

```
1  DESCRIBE times;
```

```
2
```

Result Grid Filter Rows: Export: Wrap Cell Content:						
	Field	Type	Null	Key	Default	Extra
	tme_id	smallint(6)	NO	PRI	NULL	auto_increment
	tme_nome	varchar(30)	NO		NULL	
	tme_sigla	char(3)	YES		NULL	
▶	tme_brasao	varchar(128)	NO		NULL	

Você pode ver que cada campo tem um nome, um tipo de dado, uma restrição de nulidade, uma chave primária ou estrangeira (se houver), um valor padrão (se houver) e informações extras (se houver).

DROP TABLE

O comando DROP TABLE permite excluir uma tabela existente e todos os seus dados. A sintaxe básica do comando é:

```
DROP TABLE [IF EXISTS] nome_tabela;
```

- IF EXISTS é um comando opcional que verifica se a tabela existe antes de excluí-la.
- nome_tabela é o nome da tabela que você quer excluir.

Exemplo de uso do comando DROP TABLE

- Excluir a tabela clientes se ela existir

`DROP TABLE IF EXISTS clientes;`

ALTER TABLE

O comando ALTER TABLE permite modificar a estrutura de uma tabela existente, como adicionar, remover ou alterar colunas, chaves ou restrições. A sintaxe básica do comando é:

```
ALTER TABLE nome_tabela ação;
```

- nome_tabela é o nome da tabela que você quer modificar.
- ação é o que você quer fazer com a tabela, como ADD (adicionar), DROP (remover), MODIFY (modificar), etc.

Onde ação pode ser:

- ADD COLUMN nome_da_coluna tipo_de_dado para adicionar uma nova coluna à tabela;
- DROP COLUMN nome_da_coluna para excluir uma coluna da tabela;
- MODIFY COLUMN nome_da_coluna tipo_de_dado para alterar o tipo de dado de uma coluna;
- RENAME TO novo_nome_da_tabela para renomear a tabela;
- ADD CONSTRAINT nome_da_restricao tipo_de_restricao (nome_da_coluna) para adicionar uma restrição à tabela, como chave primária, chave estrangeira ou única;
- DROP CONSTRAINT nome_da_restricao para excluir uma restrição da tabela.

Exemplos de uso do comando ALTER TABLE

- Adicionar uma coluna chamada editora na tabela livros

`ALTER TABLE livros ADD editora varchar(50);`

- Remover a coluna telefone da tabela clientes

`ALTER TABLE clientes DROP telefone;`

- Modificar o tipo de dados da coluna preco na tabela livros para int

`ALTER TABLE livros MODIFY preco int;`

ALTER TABLE

- Por exemplo, se você quiser adicionar uma coluna chamada tb01_email_aluno do tipo varchar(50) à sua tabela tb01_alunos, você pode usar o seguinte comando:

```
ALTER TABLE tb01_alunos ADD COLUMN tb01_email_aluno varchar(50);
```

- Se você quiser excluir essa coluna, você pode usar o seguinte comando:

```
ALTER TABLE tb01_alunos DROP COLUMN tb01_email_aluno;
```

- Se você quiser alterar o nome da coluna tme_sg para tme_sigla, você pode usar o seguinte comando:

```
ALTER TABLE times CHANGE tme_sg tme_sigla CHAR(3);
```

ALTER TABLE

- Se você quiser renomear a sua tabela para alunos, você pode usar o seguinte comando:

```
ALTER TABLE tb01_alunos RENAME TO alunos;
```

- Se você quiser adicionar uma restrição de chave única à coluna tb01_rg_aluno, você pode usar o seguinte comando:

```
ALTER TABLE tb01_alunos ADD CONSTRAINT rg_unico UNIQUE (tb01_rg_aluno);
```

- Se você quiser excluir essa restrição, você pode usar o seguinte comando:

```
ALTER TABLE tb01_alunos DROP CONSTRAINT rg_unico;
```

Resetar auto incremento após apagar todos os registros de uma tabela.

```
ALTER TABLE nomeTabela AUTO_INCREMENT = 1;
```


CONSTRAINT

É recomendado utilizar o comando `CONSTRAINT` para definir a chave estrangeira no relacionamento entre tabelas no mysql. O comando `CONSTRAINT` permite especificar o nome da restrição de chave estrangeira e as ações referenciais que devem ocorrer quando uma linha da tabela pai é excluída ou atualizada.

Elas podem ser especificadas no momento de criação da tabela (`CREATE`) ou após a tabela ter sido criada (`ALTER`).

CONSTRAINT

As principais constraints são as seguintes:

- NOT NULL (uma coluna a NÃO aceitar valores NULL)
- UNIQUE (identifica de forma única cada registro em uma tabela)
- PRIMARY KEY (identifica de forma única cada registro em uma tabela e automaticamente possui uma restrição UNIQUE)
- FOREIGN KEY (é usada para criar os relacionamentos entre as tabelas, sendo um campo que aponta para uma chave primária em outra tabela)
- DEFAULT (usada para inserir um valor padrão especificado em uma coluna)

A sintaxe básica do comando CONSTRAINT é:

```
ALTER TABLE nome_tabela_filha  
ADD CONSTRAINT nome_constraint  
FOREIGN KEY (nome_coluna_filha)  
REFERENCES nome_tabela_pai (nome_coluna_pai)  
[ON DELETE ação_referencial]  
[ON UPDATE ação_referencial];
```

Sintaxe básica

- nome_tabela_filha é o nome da tabela que contém a chave estrangeira.
- nome_constraint é o nome da restrição de chave estrangeira.
- nome_coluna_filha é o nome da coluna que contém a chave estrangeira na tabela filha.
- nome_tabela_pai é o nome da tabela que contém a chave primária referenciada pela chave estrangeira.
- nome_coluna_pai é o nome da coluna que contém a chave primária na tabela pai.
- ON DELETE *ação referencial* é a ação que deve ocorrer quando uma linha da tabela pai é excluída. As opções são: CASCADE, SET NULL, NO ACTION ou RESTRICT.
- ON UPDATE *ação referencial* é a ação que deve ocorrer quando uma linha da tabela pai é atualizada. As opções são as mesmas do ON DELETE

Exemplo de uso do comando CONSTRAINT

- Criar uma tabela de produtos com uma coluna id como chave primária

```
CREATE TABLE produtos (  
    id INT PRIMARY KEY,  
    descricao VARCHAR(100),  
    preco DECIMAL(10,2)  
);
```

Exemplo de uso do comando CONSTRAINT

- Criar uma tabela de pessoas com uma coluna id como chave primária

```
CREATE TABLE pessoas (  
  id INT PRIMARY KEY,  
  nome VARCHAR(50),  
  idade INT  
);
```

- Criar uma tabela de carros com uma coluna id_pessoa como chave estrangeira

```
CREATE TABLE carros (  
  placa VARCHAR(10) PRIMARY KEY,  
  modelo VARCHAR(50),  
  cor VARCHAR(20),  
  id_pessoa INT,  
  CONSTRAINT fk_carros_pessoas FOREIGN KEY (id_pessoa) REFERENCES pessoas (id)  
);
```

Exemplo de uso do comando CONSTRAINT

- Adicionar uma restrição de chave estrangeira na coluna produto_id

```
ALTER TABLE pedidos
```

```
ADD CONSTRAINT fk_pedidos_produtos
```

```
FOREIGN KEY (produto_id)
```

```
REFERENCES produtos (id)
```

```
ON DELETE CASCADE -- se um produto for excluído, todos os pedidos  
relacionados também serão excluídos
```

```
ON UPDATE NO ACTION; -- se um produto for atualizado, nada acontecerá  
nos pedidos relacionados
```

Exemplo de uso do comando CONSTRAINT

- Criar uma tabela de clientes com uma coluna cidade com valor padrão São Paulo

```
CREATE TABLE clientes (  
    id INT PRIMARY KEY,  
    nome VARCHAR(50),  
    telefone VARCHAR(15),  
    cidade VARCHAR(20) DEFAULT 'São Paulo'  
);
```


Exemplo de uso do comando CONSTRAINT

- Criar uma tabela de produtos com uma coluna codigo com restrição única

```
CREATE TABLE produtos (  
    id INT PRIMARY KEY,  
    codigo VARCHAR(10) UNIQUE,  
    nome VARCHAR(50),  
    preco DECIMAL(10,2)  
);
```