

Banco de Dados

MySQL: CREATE e ALTER

Objetivos

- Apresentar os comandos para criação de bancos de dados (esquemas) utilizando o SGBD MySQL.
- Alterar a estrutura de um banco de dados (esquema) já existente.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

Pode-se utilizar tanto a palavra-chave **DATABASE** quanto **SCHEMA**.

Usaremos esta notação { e } para indicar valores/nomes possíveis para um determinado comando, sendo estes separados por uma barra vertical.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

Opcional. Se utilizado, o banco de dados só será criado caso não exista nenhum outro banco com o nome indicado.

Se **IF NOT EXISTS** não for utilizado e já existir um banco de dados com o nome informado, resultará em um erro.

Usaremos a notação [e] para indicar que parte de um comando ou parâmetro é opcional.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

Nome que será dado ao banco de dados.

Usaremos a notação < e > para indicar que o parâmetro para um comando deve ser informado obrigatoriamente.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

- Exemplo:

```
CREATE DATABASE aula_sql;
```

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

- Exemplo:

```
CREATE DATABASE aula_sql;
```

Comando indicando que desejamos criar um banco de dados.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.

- Sintaxe:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] <nome_bd>;
```

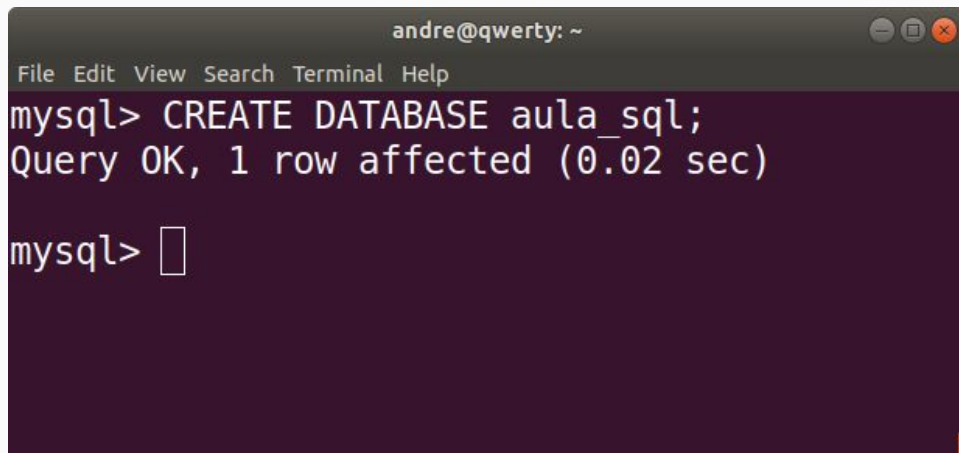
- Exemplo:

```
CREATE DATABASE aula_sql;
```

Nome do banco de dados que deverá ser criado.

- **CREATE DATABASE**

- Cria um banco de dados (um esquema), o qual será utilizado para armazenamento das tabelas de dados.
- Exemplo de uso:

A terminal window titled 'andre@qwerty: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'mysql> CREATE DATABASE aula_sql;' followed by the output 'Query OK, 1 row affected (0.02 sec)'. Below this, the prompt 'mysql>' is followed by a cursor icon.

```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> CREATE DATABASE aula_sql;  
Query OK, 1 row affected (0.02 sec)  
  
mysql> █
```

- **SHOW DATABASES**

- Exibe os bancos de dados existentes no SGBD.

- Sintaxe:

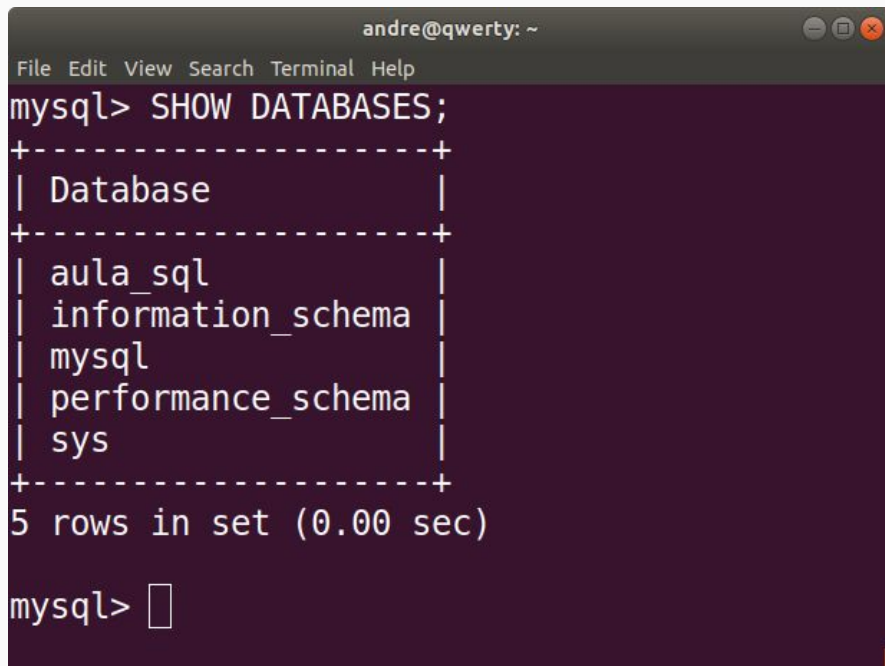
```
SHOW {DATABASES | SCHEMAS};
```

- Exemplo:

```
SHOW DATABASES;
```

- **SHOW DATABASES**

- Exibe os bancos de dados existentes no SGBD.
- Exemplo de uso:

A terminal window titled 'andre@qwerty: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'mysql> SHOW DATABASES;' and its output as a table with 5 rows. The output is: +-----+ | Database | +-----+ | aula_sql | | information_schema | | mysql | | performance_schema | | sys | +-----+. Below the table, it says '5 rows in set (0.00 sec)'. The prompt 'mysql>' is followed by a cursor.

```
andre@qwerty: ~
File Edit View Search Terminal Help
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| aula_sql |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> 
```

- **DROP DATABASE**

- Apaga todas as tabelas de um banco de dados e, em seguida, apaga o banco de dados.

- Sintaxe:

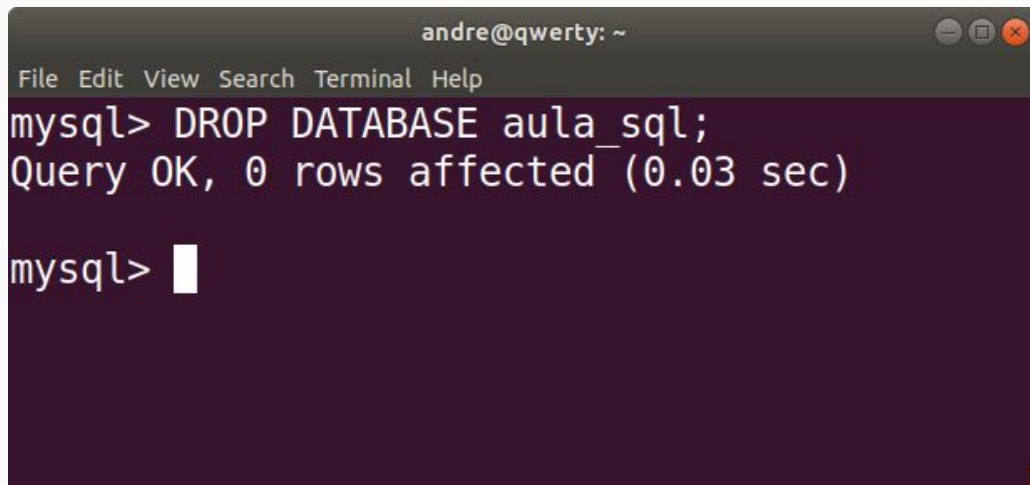
```
DROP {DATABASE | SCHEMA} [IF EXISTS] <nome_bd>;
```

- Exemplo:

```
DROP DATABASE aula_sql;
```

- **DROP DATABASE**

- Apaga todas as tabelas de um banco de dados e, em seguida, apaga o banco de dados.
- Exemplo de uso:

A terminal window titled 'andre@qwerty: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'mysql> DROP DATABASE aula_sql;' being executed, followed by the output 'Query OK, 0 rows affected (0.03 sec)'. The prompt 'mysql>' is shown again with a cursor.

```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> DROP DATABASE aula_sql;  
Query OK, 0 rows affected (0.03 sec)  
  
mysql> █
```

MySQL

Indicação do banco de dados que será utilizado

- Antes de começarmos a criar tabelas em um banco de dados, devemos informar ao MySQL em qual banco de dados desejamos trabalhar.
- Isso é feito com o comando **USE** do MySQL

- Sintaxe:

```
USE <nome_bd>;
```

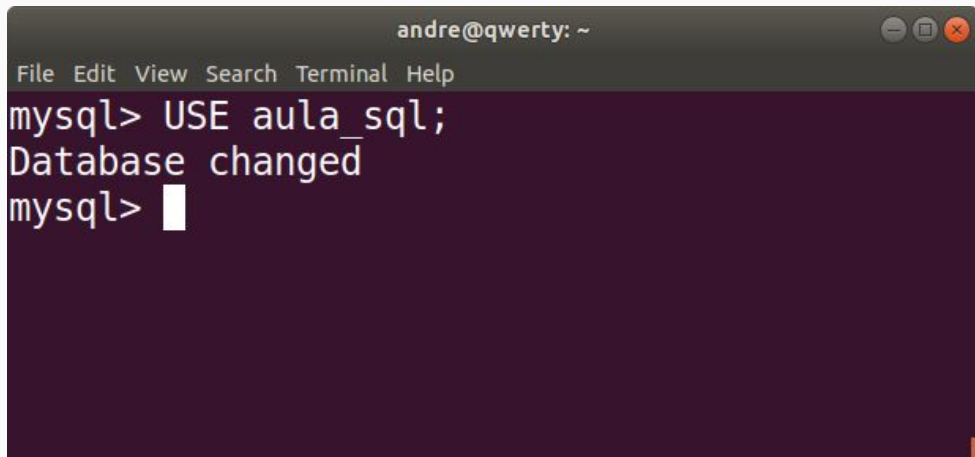
- Exemplo:

```
USE aula_sql
```

MySQL

Indicação do banco de dados que será utilizado

- Antes de começarmos a criar tabelas em um banco de dados, devemos informar ao MySQL em qual banco de dados desejamos trabalhar.
- Isso é feito com o comando **USE** do MySQL.
 - Exemplo de uso:



```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> USE aula_sql;  
Database changed  
mysql> 
```


- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.

- Sintaxe:

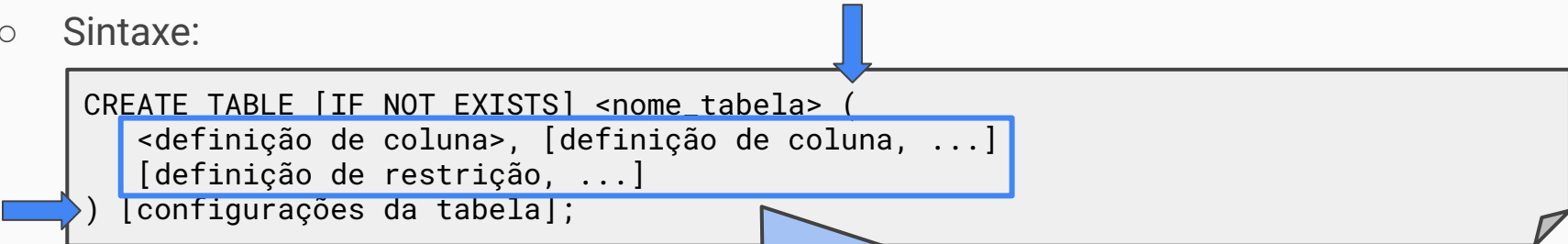
Nome que será dado à tabela.

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.

- Sintaxe:



```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição das colunas e restrições.

Podemos definir várias colunas e restrições, basta separá-los por vírgula.

Note que as definições de colunas e restrições devem estar entre os parênteses.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Configurações adicionais
(opcionais) para a tabela.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
    [PRIMARY KEY]  
    [AUTO_INCREMENT]  
    [UNIQUE]  
    [NOT NULL]  
    [DEFAULT {valor | expressão}]
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
[PRIMARY KEY]  
[AUTO_INCREMENT]  
[DEFAULT {valor | expressão}]
```

Nome da coluna.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
[PRIMARY KEY]  
[AUTO_INCREMENT]  
[UNIQUE]  
[NOT NULL]  
[DEFAULT]
```

Define o tipo de dado da coluna.

Exemplos de tipos:

INTEGER, DECIMAL, FLOAT, DOUBLE,
BOOLEAN, BIT, CHAR, VARCHAR, TEXT,
BLOB, DATE, TIME, DATETIME, ...

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
[PRIMARY KEY]  
[AUTO_INCREMENT]  
[UNIQUE]  
[NOT NULL]  
[DEFAULT {valor | expr}]
```

Determina que a coluna deve ser utilizada como chave primária. Se utilizada, a coluna será automaticamente NOT NULL e UNIQUE, mesmo que não especificado explicitamente.

Caso a chave primária da tabela seja composta por mais de uma coluna, deve-se utilizar a sintaxe de definição de restrições, que será mostrada a diante.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
    [PRIMARY KEY]  
    [AUTO_INCREMENT]  
    [UNIQUE]  
    [NOT NULL]  
    [DEFAULT {valor | expressão}]
```

Utilizado apenas em colunas do tipo inteiro ou de ponto-flutuante.

Quando inserido um registro com valor NULL para uma coluna especificada com AUTO_INCREMENT, o valor é definido automaticamente como o próximo valor da sequência.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
    [PRIMARY KEY]  
    [AUTO_INCREMENT]  
    [UNIQUE]  
    [NOT NULL]  
    [DEFAULT {valor}]
```

Determina que os valores dos registros devem ser únicos, ou seja, não podem se repetir, mesmo que a coluna não seja uma chave.

Ocorrerá um erro caso tente inserir registros com valores repetidos para colunas especificadas com UNIQUE.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
    [PRIMARY KEY]  
    [AUTO_INCREMENT]  
    [UNIQUE]  
    [NOT NULL]  
    [DEFAULT {valor}]
```

Indica que a coluna é obrigatória, ou seja, não podem ser inseridos registros com valor NULL para colunas especificadas com NOT NULL.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de coluna:

```
<nome_coluna> <tipo>  
    [PRIMARY KEY]  
    [AUTO_INCREMENT]  
    [UNIQUE]  
    [NOT NULL]  
    [DEFAULT {valor | expressão}]
```

Determina um valor padrão que deverá ser atribuído à coluna caso ele não seja especificado no momento em que o registro é inserido na tabela.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

Indica que estamos definindo uma restrição de integridade para a tabela que está sendo criada.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

Atribuímos um nome para identificação da restrição criada. Caso não seja especificado, um nome é definido pelo próprio SGBD.

O nome que identifica uma restrição é útil quando precisamos remover/alterar as restrições de integridade de uma tabela.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

Indicamos o tipo da restrição que está sendo definida. Neste caso, estamos definindo uma restrição de chave primária.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

Com essa sintaxe podemos definir uma chave primária composta por múltiplas colunas.

Ao definir a restrição de chave primária na definição de uma coluna, estamos limitados a chaves primárias simples.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave primária:

```
CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...])
```

Especificação das colunas que compõem a chave primária da tabela. Observe que as colunas devem ser especificadas entre os parênteses

Caso a chave primária seja composta, as colunas devem estar separadas por vírgula

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
    REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...])
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...])
```

Indicamos o tipo da restrição que está sendo definida. Neste caso, estamos definindo uma restrição de integridade referencial, ou seja, uma chave estrangeira.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...])
```

Especificação das colunas que são chave estrangeira, ou seja, as da tabela que está sendo criada que devem referenciar colunas de outra tabela.

Observe que as colunas devem ser especificadas entre parênteses e, caso seja mais de uma colunas, estas devem ser separadas por vírgula.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...])
```

Nome da tabela referenciada.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...])
```

Colunas referenciadas.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de chave estrangeira:

```
CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna> [coluna] ...)  
REFERENCES <tabela_ref> (<coluna_ref> [coluna_ref] ...)
```



A especificação das colunas referenciadas deve seguir a mesma sequência das colunas que as referenciam.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de coluna(s) única(s):

```
CONSTRAINT [nome_restrição] UNIQUE (<coluna>, [coluna, ...])
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de coluna(s) única(s):

```
CONSTRAINT [nome_restrição] UNIQUE (<coluna>, [coluna, ...])
```

Indicamos o tipo da restrição que está sendo definida. Neste caso, estamos definindo que uma coluna (ou conjunto de colunas) deve ser único para cada instância (linha) na tabela.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Definição de restrição de coluna(s) única(s):

```
CONSTRAINT [nome_restrição] UNIQUE (<coluna>, [coluna, ...])
```

Especificação da coluna (ou colunas)
que compõem a restrição.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Configurações da tabela:

```
[ENGINE = <valor>]  
[CHARACTER SET = <valor>]
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Configurações da tabela:

```
[ENGINE = <valor>]  
[CHARACTER SET = <valor>]
```

Especifica o formato de armazenamento de tabelas. O valor padrão é **InnoDB**.

Adiante veremos os valores possíveis.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Sintaxe:

```
CREATE TABLE [IF NOT EXISTS] <nome_tabela> (  
    <definição de coluna>, [definição de coluna, ...]  
    [definição de restrição, ...]  
) [configurações da tabela];
```

Configurações da tabela:

```
[ENGINE = <valor>]  
[CHARACTER SET = <valor>]
```

Define o conjunto de caracteres utilizados para armazenamento dos dados na tabela.

Alguns valores possíveis são:

utf8 : UTF-8 Unicode
utf16 : UTF-16 Unicode
latin1 : ISO-8859-1

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Exemplo:

```
CREATE TABLE Funcionario (  
    id INTEGER PRIMARY KEY AUTO_INCREMENT,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL  
);  
  
CREATE TABLE Dependente (  
    id INTEGER NOT NULL,  
    id_funcionario INTEGER NOT NULL,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
    CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
        REFERENCES Funcionario (id)  
);
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.

- Exemplo:

```
CREATE TABLE Funcionario (  
    id INTEGER PRIMARY KEY AUTO_INCREMENT,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL  
);
```

```
CREATE TABLE Dependente (  
    id INTEGER NOT NULL,  
    id_funcionario INTEGER NOT NULL,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
    CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
        REFERENCES Funcionario (id)  
);
```

Definição da tabela **Funcionário**, formada pelas colunas **id**, **nome** e **cpf**.

A coluna **id** forma a chave-primária da tabela.

A coluna **nome** não pode ser nula.

A coluna **cpf** não pode repetir para os diferentes registros (linhas) da tabela.

MySQL

Criação de tabelas

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Exemplo:

```
CREATE TABLE Funcionario (  
    id INTEGER PRIMARY KEY AUTO_INCREMENT,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL  
);
```

```
CREATE TABLE Dependente (  
    id INTEGER NOT NULL,  
    id_funcionario INTEGER NOT NULL,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
    CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
        REFERENCES Funcionario (id)  
);
```

Definição da tabela **Dependente**, formada pelas colunas **id**, **id_funcionario**, **nome** e **cpf**.

A coluna **id** forma a chave-primária da tabela.

A coluna nome não pode ser nula.

A coluna **cpf** não pode repetir para os diferentes registros (linhas) da tabela.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Exemplo:

```
CREATE TABLE Funcionario (  
    id INTEGER PRIMARY KEY AUTO_INCREMENT,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL  
);
```

```
CREATE TABLE Dependente (  
    id INTEGER NOT NULL,  
    id_funcionario INTEGER NOT NULL,  
    nome VARCHAR(50) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
    CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
        REFERENCES Funcionario (id)  
);
```

Definimos que a chave-primária da tabela **Dependente** será composta pelas colunas **id** e **id_funcionário**.

- **CREATE TABLE**

- Cria uma tabela no banco de dados.

- Exemplo:

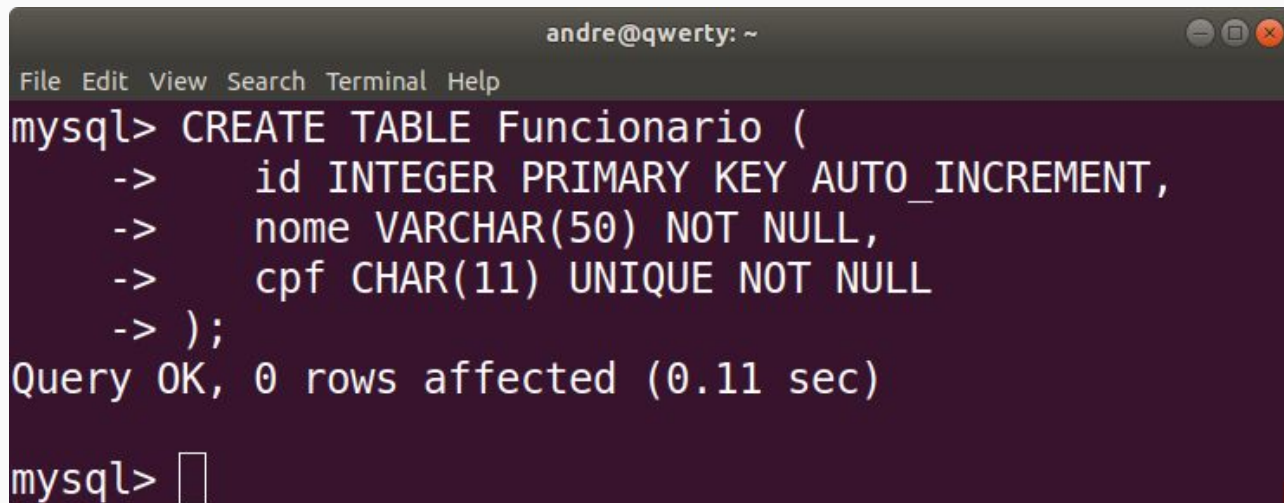
```
CREATE TABLE Funcionario (  
  id INTEGER PRIMARY KEY AUTO_INCREMENT,  
  nome VARCHAR(50) NOT NULL,  
  cpf CHAR(11) UNIQUE NOT NULL  
);
```

```
CREATE TABLE Dependente (  
  id INTEGER NOT NULL,  
  id_funcionario INTEGER NOT NULL,  
  nome VARCHAR(50) NOT NULL,  
  cpf CHAR(11) UNIQUE NOT NULL,  
  CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
  CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
    REFERENCES Funcionario (id)  
);
```

Aqui definimos que a coluna **id_funcionario** é uma chave-estrangeira que referencia a coluna **id** da tabela **Funcionário**.

- **CREATE TABLE**

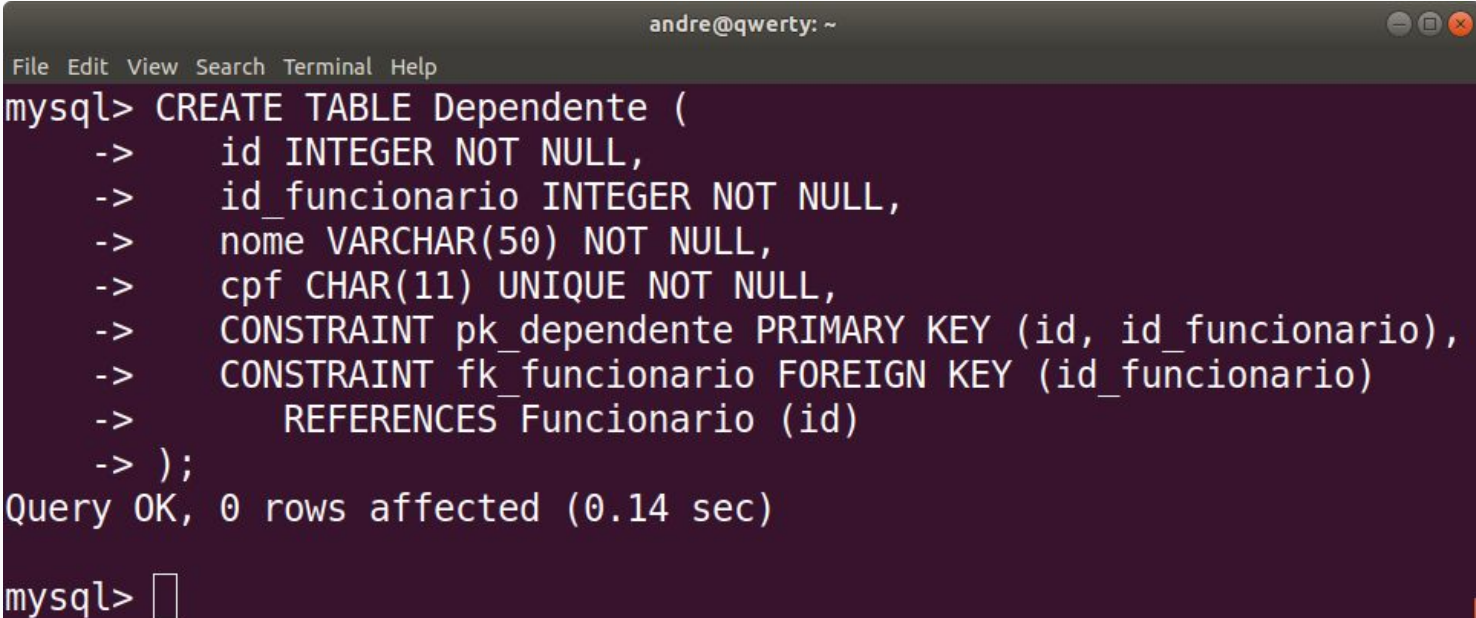
- Cria uma tabela no banco de dados.
- Exemplo de uso:

A terminal window titled 'andre@qwerty: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows a MySQL prompt 'mysql>' followed by the command 'CREATE TABLE Funcionario ('. The command is entered in four lines: 'id INTEGER PRIMARY KEY AUTO_INCREMENT,', 'nome VARCHAR(50) NOT NULL,', 'cpf CHAR(11) UNIQUE NOT NULL', and ');'. The response 'Query OK, 0 rows affected (0.11 sec)' is shown. The prompt 'mysql>' is followed by a cursor.

```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> CREATE TABLE Funcionario (  
->     id INTEGER PRIMARY KEY AUTO_INCREMENT,  
->     nome VARCHAR(50) NOT NULL,  
->     cpf CHAR(11) UNIQUE NOT NULL  
-> );  
Query OK, 0 rows affected (0.11 sec)  
  
mysql> █
```

- **CREATE TABLE**

- Cria uma tabela no banco de dados.
- Exemplo de uso:

A terminal window titled 'andre@qwerty: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows a MySQL prompt 'mysql>' followed by a 'CREATE TABLE' command for a table named 'Dependente'. The command specifies columns: 'id' (INTEGER NOT NULL), 'id_funcionario' (INTEGER NOT NULL), 'nome' (VARCHAR(50) NOT NULL), and 'cpf' (CHAR(11) UNIQUE NOT NULL). It also includes a primary key constraint 'pk_dependente' on 'id' and 'id_funcionario', and a foreign key constraint 'fk_funcionario' on 'id_funcionario' that references the 'id' column of a table named 'Funcionario'. The command ends with a semicolon. The output shows 'Query OK, 0 rows affected (0.14 sec)' and the prompt returns to 'mysql>' with a cursor.

```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> CREATE TABLE Dependente (  
-> id INTEGER NOT NULL,  
-> id_funcionario INTEGER NOT NULL,  
-> nome VARCHAR(50) NOT NULL,  
-> cpf CHAR(11) UNIQUE NOT NULL,  
-> CONSTRAINT pk_dependente PRIMARY KEY (id, id_funcionario),  
-> CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
-> REFERENCES Funcionario (id)  
-> );  
Query OK, 0 rows affected (0.14 sec)  
mysql> █
```

MySQL

Storage Engines

- Formatos de armazenamento de tabelas:

Storage Engine	Descrição
InnoDB	Tabelas com suporte a chaves-estrangeiras e transação com bloqueio de linhas. É o <i>engine</i> padrão utilizado pelo MySQL.
MyISAM	Um formato binário portátil de armazenamento de tabelas. Não suporta transações nem chaves-estrangeiras.
MEMORY	Armazenamento dos dados apenas em memória principal.
CSV	Tabelas são armazenadas em arquivos no formato CSV (<i>Comma-Separated Values</i>)

- Existem outros *engines* além dos apresentados acima.

- **SHOW TABLES**

- Exibe as tabelas existentes no banco de dados atualmente em uso.

- Sintaxe:

```
SHOW TABLES;
```

- Exemplo de uso:



The screenshot shows a terminal window with the title 'andre@qwerty: ~'. The terminal contains the following text:

```
File Edit View Search Terminal Help
mysql> SHOW TABLES;
+-----+
| Tables_in_aula_sql |
+-----+
| Dependente         |
| Funcionario        |
+-----+
2 rows in set (0.00 sec)

mysql> 
```

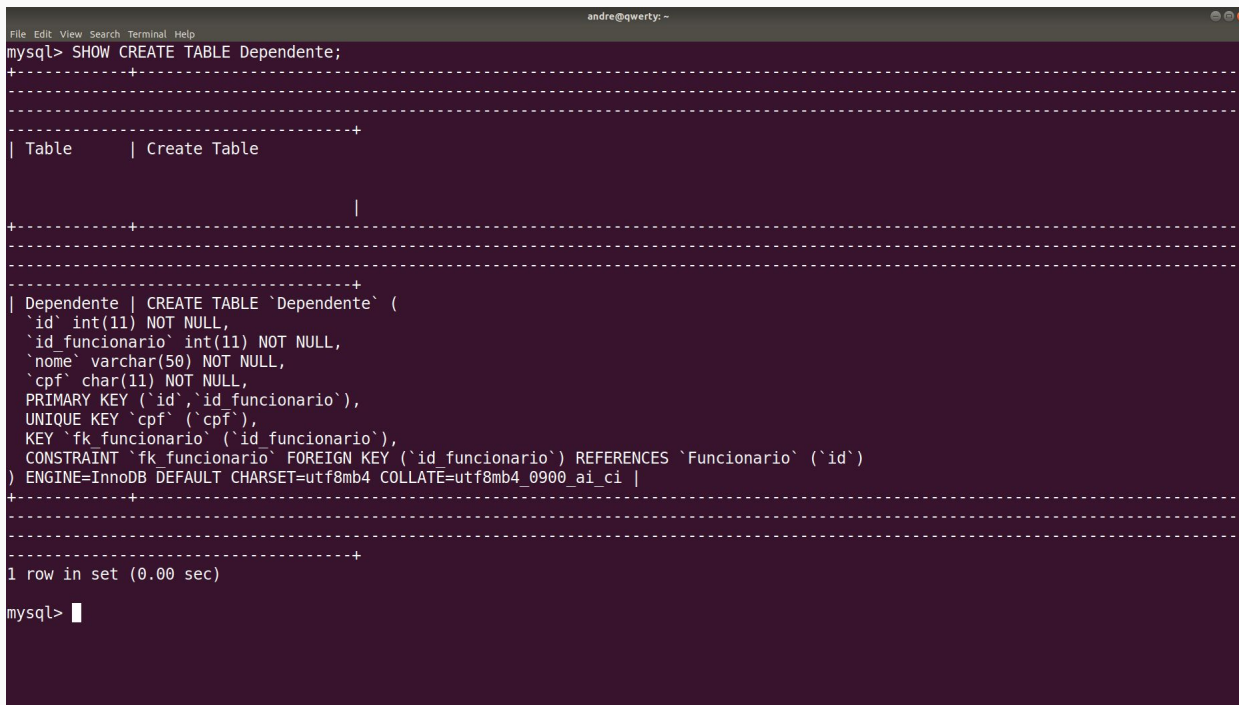
- **SHOW CREATE TABLE**

- Exibe comando completo utilizado para a criação de uma tabela
- Sintaxe:

```
SHOW CREATE TABLE Dependente;
```


- **SHOW CREATE TABLE**

- Exibe comando completo utilizado para a criação de uma tabela
- Exemplo de uso:



```
File Edit View Search Terminal Help
andre@qwerty: ~
mysql> SHOW CREATE TABLE Dependente;
+-----+-----+
| Table | Create Table |
+-----+-----+
| Dependente | CREATE TABLE `Dependente` (
  `id` int(11) NOT NULL,
  `id_funcionario` int(11) NOT NULL,
  `nome` varchar(50) NOT NULL,
  `cpf` char(11) NOT NULL,
  PRIMARY KEY (`id`,`id_funcionario`),
  UNIQUE KEY `cpf` (`cpf`),
  KEY `fk_funcionario` (`id_funcionario`),
  CONSTRAINT `fk_funcionario` FOREIGN KEY (`id_funcionario`) REFERENCES `Funcionario` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-----+-----+
1 row in set (0.00 sec)

mysql> █
```

- **DROP TABLE**

- Apaga uma tabela de um banco de dados.
- Sintaxe:

```
DROP TABLE [IF EXISTS] <nome_da_tabela>;
```

- Exemplo:

```
DROP TABLE Dependente;
```

- **DROP TABLE**

- Apaga uma tabela de um banco de dados.
- Exemplo de uso:



```
andre@qwerty: ~  
File Edit View Search Terminal Help  
mysql> DROP TABLE Dependente;  
Query OK, 0 rows affected (0.07 sec)  
  
mysql> █
```

- **ALTER TABLE**

- Altera a estrutura de uma tabela do banco de dados.
- Sintaxe:

```
ALTER TABLE <tabela> <alterações>;
```

- O que pode ser alterado na estrutura de uma tabela:
 - Adicionar e remover uma colunas
 - Redefinir, renomear e reordenar colunas
 - Definir e remover chave primária
 - Adicionar e remover chaves estrangeiras
 - Adicionar e remover outras restrições/chaves/índices
 - Renomear tabela
 - Renomear chaves e índices
 - Alterar configurações da tabela
 - ...

- **ALTER TABLE**

- Adicionar coluna

- Sintaxe:

```
ALTER TABLE <tabela>  
  ADD [COLUMN] <nova_coluna> <especificações>  
  [{FIRST | AFTER} <coluna_existente>];
```

- Exemplos:

```
ALTER TABLE Funcionario  
  ADD COLUMN email VARCHAR(50) UNIQUE;
```

```
ALTER TABLE Funcionario  
  ADD COLUMN telefone VARCHAR(15)  
  AFTER nome;
```

- **ALTER TABLE**

- Remover coluna

- Sintaxe:

```
ALTER TABLE <tabela>  
    DROP [COLUMN] <nome_coluna>;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    DROP COLUMN telefone;
```

- **ALTER TABLE**

- Renomear coluna

- Sintaxe:

```
ALTER TABLE <tabela>  
    RENAME COLUMN <nome_coluna> TO <novo_nome>;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    RENAME COLUMN cpf TO documento;
```

- **ALTER TABLE**

- Alterar especificações de uma coluna

- Sintaxe:

```
ALTER TABLE <tabela>  
    MODIFY [COLUMN] <nome_coluna> <especificações>  
    [{FIRST | AFTER} <coluna_existente>];
```

- Exemplos:

```
ALTER TABLE Funcionario  
    MODIFY COLUMN nome VARCHAR(30) NOT NULL;
```


- **ALTER TABLE**

- Renomear e alterar especificações de uma coluna

- Sintaxe:

```
ALTER TABLE <tabela>  
  CHANGE [COLUMN] <nome_coluna> <novo_nome> <especificações>  
  [{FIRST | AFTER} <coluna_existente>];
```

- Exemplos:

```
ALTER TABLE Funcionario  
  CHANGE COLUMN documento cpf VARCHAR(14) NOT NULL;
```

Como a coluna **documento** já tinha uma restrição do tipo **UNIQUE** associada a ela anteriormente, não precisamos incluir novamente. Se incluído, resultará em duas restrições **UNIQUE** associadas à coluna.

Restrições do tipo chave primária, chave estrangeira, não repetição (**UNIQUE**) são modificados separadamente.

- **ALTER TABLE**

- Remover chave primária

- Sintaxe:

```
ALTER TABLE <tabela>  
    DROP PRIMARY KEY;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    MODIFY COLUMN id INTEGER NOT NULL;
```

```
ALTER TABLE Funcionario  
    DROP PRIMARY KEY;
```

- **ALTER TABLE**

- Remover chave primária

- Sintaxe:

```
ALTER TABLE <tabela>  
    DROP PRIMARY KEY;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    MODIFY COLUMN id INTEGER NOT NULL;
```

```
ALTER TABLE Funcionario  
    DROP PRIMARY KEY;
```

Caso alguma coluna que compõe a chave-primária seja **AUTO_INCREMENT**, devemos modificá-la antes da remoção da chave-primária de forma que ela não seja mais uma coluna **AUTO_INCREMENT**.

- **ALTER TABLE**

- Definir chave primária

- Sintaxe:

```
ALTER TABLE <tabela>  
    ADD CONSTRAINT [nome_restrição] PRIMARY KEY (<coluna>, [coluna, ...]);
```

- Exemplos:

```
ALTER TABLE Funcionario  
    ADD CONSTRAINT PRIMARY KEY (id);
```

- **ALTER TABLE**

- Remover chave estrangeira

- Sintaxe:

```
ALTER TABLE <tabela>  
    DROP FOREIGN KEY <nome_restrição>;
```

- Exemplos:

```
ALTER TABLE Dependente  
    DROP FOREIGN KEY fk_funcionario;
```

- **ALTER TABLE**

- Definir chave estrangeira

- Sintaxe:

```
ALTER TABLE <tabela>  
  ADD CONSTRAINT [nome_restrição] FOREIGN KEY (<coluna>, [coluna, ...])  
    REFERENCES <tabela_ref> (<coluna_ref>, [coluna_ref, ...]);
```

- Exemplos:

```
ALTER TABLE Dependente  
  ADD CONSTRAINT fk_funcionario FOREIGN KEY (id_funcionario)  
    REFERENCES Funcionario (id);
```

- **ALTER TABLE**

- Remover restrição UNIQUE

- Sintaxe:

```
ALTER TABLE <tabela>  
    DROP {INDEX | KEY} <nome_restricao>;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    DROP INDEX cpf;
```

- **ALTER TABLE**

- Adicionar restrição UNIQUE

- Sintaxe:

```
ALTER TABLE <tabela>  
  ADD CONSTRAINT [nome_restrição] UNIQUE [{INDEX | KEY}]  
    (<coluna>, [coluna, ...]);
```

- Exemplos:

```
ALTER TABLE Funcionario  
  ADD CONSTRAINT unique_cpf UNIQUE (cpf);
```


- **ALTER TABLE**

- Renomear tabela

- Sintaxe:

```
ALTER TABLE <tabela>  
    RENAME [{TO | AS}] <novo_nome_tabela>;
```

- Exemplos:

```
ALTER TABLE Funcionario  
    RENAME Empregado;
```

- **ALTER TABLE**

- Alterar configurações de uma tabela
- Sintaxe:

```
ALTER TABLE <tabela>  
    ENGINE = <valor>;
```

```
ALTER TABLE <tabela>  
    AUTO_INCREMENT = <valor>;
```

```
ALTER TABLE <tabela>  
    CHARACTER SET = <valor>;
```

```
ALTER TABLE <tabela>  
    CONVERT TO CHARACTER SET <valor>;
```