SQL

DDL - Data Definition Language (Linguagem de Definição de dados)

Linguagem de Definição de dados

- DDL ou Data Definition Language (Linguagem de Definição de dados) permite ao usuário definir as novas tabelas e os elementos que serão associados a elas. É responsável pelos comandos de criação e alteração no banco de dados, sendo composto por três comandos: **CREATE**, **ALTER** e **DROP**.

Linguagem de Definição de dados

```
CREATE DATABASE fag;
```

Linguagem de Definição de dados

- O comando **CREATE DATABASE** é responsável pela criação de um novo banco de dados vazio, conforme podemos ver abaixo:
- CREATE DATABASE fag;
- Ao ser executado, estaremos criando o banco de dados chamado “fag”.

Linguagem de Definição de dados

- Já o comando CREATE TABLE irá criar uma nova tabela. Os bancos de dados relacionais guardam seu dados dentro de tabelas que são divididas em colunas. Desta forma, veremos abaixo a criação de uma tabela de usuário. Ao criar, especificaremos as suas colunas e quais tipos de dados elas irão receber (neste caso, um ID e o nome do usuário).

```
CREATE TABLE usuario (id INT, nome VARCHAR (255));
```

SQL Constraints (Restrições) no MySQL

- As restrições são regras aplicadas nas colunas de uma tabela.
- São usadas para limitar os tipos de dados que são inseridos.
- Podem ser especificadas no momento de criação da tabela (CREATE) ou após a tabela ter sido criada (ALTER)

SQL Constraints (Restrições) no MySQL

- As principais constraints são as seguintes:
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - DEFAULT

SQL Constraints (Restrições) no MySQL

NOT NULL

- A constraint NOT NULL impõe a uma coluna a NÃO aceitar valores NULL.
- Ou seja, a constraint NOT NULL obriga um campo a sempre possuir um valor.
- Deste modo, não é possível inserir um registro (ou atualizar) sem entrar com um valor neste campo.

SQL Constraints (Restrições) no MySQL

UNIQUE

- A restrição UNIQUE identifica de forma única cada registro em uma tabela de um banco de dados.
- As constraints UNIQUE e PRIMARY KEY garantem a unicidade em uma coluna ou conjunto de colunas.
- Uma constraint PRIMARY KEY automaticamente possui uma restrição UNIQUE definida, portanto não é necessário especificar essa constraint neste caso.
- É possível termos várias constraints UNIQUE em uma mesma tabela, mas apenas uma Chave Primária por tabela (lembrando que uma PK pode ser composta, ou seja, constituída por mais de uma coluna – mas ainda assim, será uma única chave primária).

SQL Constraints (Restrições) no MySQL

PRIMARY KEY

- A restrição PRIMARY KEY (Chave Primária) identifica de forma única cada registro em uma tabela de banco de dados.
- As Chaves Primárias devem sempre conter valores únicos.
- Uma coluna de chave primária não pode conter valores NULL
- Cada tabela deve ter uma chave primária e apenas uma chave primária.

SQL Constraints (Restrições) no MySQL

FOREIGN KEY

- Uma FOREIGN KEY (Chave Estrangeira) em uma tabela é um campo que aponta para uma chave primária em outra tabela. Desta forma, é usada para criar os relacionamentos entre as tabelas no banco de dados.

SQL Constraints (Restrições) no MySQL

Veja um exemplo de restrição Foreign Key aplicada:

```
CONSTRAINT fk_ID_Autor FOREIGN KEY (ID_Autor)
REFERENCES tbl_autores(ID_Autor)
```

Neste exemplo a chave primária está na tabela tbl_autores e uma chave estrangeira de nome ID_Autor foi criada na tabela atual, usando o nome fk_ID_Autor

SQL Constraints (Restrições) no MySQL

DEFAULT

- A restrição DEFAULT é usada para inserir um valor padrão especificado em uma coluna.
- O valor padrão será adicionado a todos os novos registros caso nenhum outro valor seja especificado na hora de inserir dados.

SQL Constraints (Restrições) no MySQL

DEFAULT

- A restrição DEFAULT é usada para inserir um valor padrão especificado em uma coluna.
- O valor padrão será adicionado a todos os novos registros caso nenhum outro valor seja especificado na hora de inserir dados.

Tipos de dados MYSQL

Tipo	Descrição
INT	Inteiros entre -2,147,483,648 e 2,147,483,647
TINYINT	Números inteiros de -128 a 127
SMALLINT	Números inteiros de -32768 a 32767
MEDIUMINT	Números inteiros de -8388608 a 8388607
BIGINT	Números entre -9,223,372,036,854,775,808 e 9,223,372,036,854,775,807
DECIMAL(M,D)	Ponto decimal com M dígitos no total (precisão) e D casas decimais (escala); o padrão é 10,0; M vai até 65 e D até 30.
FLOAT(M,D)	Ponto flutuante com precisão M e escala D; o padrão é 10,2; D vai até 24.
CHAR(M)	String que ocupa tamanho fixo entre 0 e 255 caracteres
BOOL / BOOLEAN	Valores binários 0 / 1; Na verdade, é um alias para o tipo TINYINT(1)

Tipos de dados MYSQL

VARCHAR(M)	String de tamanho variável, com até 65535 M caracteres.
BLOB / MEDIUMBLOB/ TINYBLOB	Campo com tamanho máximo de 65535 caracteres binários; 'Binary Large Objects', são usados para armazenar grandes quantidades de dados, como imagens.
MEDIUMTEXT	Permite armazenar até 16.777.215 caracteres.
LONGTEXT	Permite armazenar até 4.294.967.295 caracteres.
DATE	Uma data de 01/01/1000 a 31/12/9999, no formato YYYY-MM-DD
DATETIME	Uma combinação de data e hora de 01/01/1000 00:00:00 a 31/12/9999 23:59:59, no formato YYYY-MM-DD HH:MM:SS
TIME	Hora apenas, no formato HH:MM:SS
YEAR(M)	Ano nos formatos de 2 ou 4 dígitos; Se forem 2 (YEAR(2)), ano vai de 1970 a 2069; para 4 (YEAR(4)), vai de 1901 a 2155. O padrão é 4.

Exemplo de uso

- CREATE TABLE

```
1  CREATE TABLE IF NOT EXISTS tbl_livro (
2  ID_Livro SMALLINT  AUTO_INCREMENT PRIMARY KEY,
3  Nome_Livro VARCHAR(70) NOT NULL,
4  ISBN13 CHAR(13),
5  ISBN10 CHAR(10),
6  ID_Categoria SMALLINT,
7  ID_Autor SMALLINT NOT NULL,
8  Data_Pub DATE NOT NULL,
9  Preco_Livro DECIMAL(6,2) NOT NULL
10 );
```

Exemplo de uso

- CREATE TABLE

```sql
CREATE TABLE IF NOT EXISTS tasks (
    task_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    start_date DATE,
    due_date DATE,
    status TINYINT NOT NULL,
    priority TINYINT NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
```

Exemplo de uso

- CREATE TABLE

```
1  CREATE TABLE IF NOT EXISTS checklists (
2      todo_id INT AUTO_INCREMENT,
3      task_id INT,
4      todo VARCHAR(255) NOT NULL,
5      is_completed BOOLEAN NOT NULL DEFAULT FALSE,
6      PRIMARY KEY (todo_id , task_id),
7      FOREIGN KEY (task_id)
8          REFERENCES tasks (task_id)
9          ON UPDATE RESTRICT ON DELETE CASCADE
10 );
```

ALTER TABLE

- RENAME
- MODIFY
- CHANGE
- ADD
- DROP
- UNSIGNED
- SIGNED
- PRIMARY KEY
- FORENG KEY
- INDEX
- UNIQUE

ALTER TABLE

```
ALTER TABLE t1 RENAME t2;
```

```
ALTER TABLE t2 MODIFY a TINYINT NOT NULL, CHANGE b c CHAR(20);
```

```
ALTER TABLE t2 ADD d TIMESTAMP;
```

```
ALTER TABLE t2 ADD INDEX (d), ADD UNIQUE (a);
```

```
ALTER TABLE t2 DROP COLUMN c;
```

```
ALTER TABLE t2 ADD c INT UNSIGNED NOT NULL AUTO_INCREMENT,
  ADD PRIMARY KEY (c);
```

```
ALTER TABLE Orders
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

Exemplos de criação de chaves estrangeiras

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

```sql
ALTER TABLE "clientes" ADD CONSTRAINT "fk_cidade" FOREIGN KEY ( "codcidade" ) REFERENCES "cidade" ( "codcidade" ) ;
```