MySQL

CONCEITOS E PRÁTICAS DO SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS

O QUE JÁ SABEMOS...

- O que é um Banco de Dados.
- Projeto de Banco de Dados
- Modelo de Dados Relacional
- Diagrama Entidade Relacionamento
- SGBDs
- Normalização
- Etc.

SQL

- **Structured Query Language**, ou **Linguagem de Consulta Estruturada** ou **SQL**;
- Trata-se de uma linguagem específica para a manipulação de tabelas de dados;
- A linguagem padrão universal para manipular bancos de dados relacionais através dos SGBDs.

GRUPOS DE COMANDOS SQL

- Os comandos do SQL **são classificados em três grupos**, de acordo com suas principais funções:
- **DML – Data Manipulation Language**
- **DDL – Data Definition Language**
- **DCL – Data Control Language**

DML

- (Linguagem de Manipulação de Dados);
- É o subconjunto mais utilizado da linguagem **SQL**, pois é através da DML que operamos sobre os dados dos bancos de dados com instruções de inserção, atualização, exclusão e consulta de informações. Comandos como INSERIR, DELETAR, ATUALIZAR, SELECIONAR E ETC.

DDL

- (Linguagem de Definição de Dados) é o subconjunto da **SQL** utilizado para gerenciar a estrutura do banco de dados. Com a DDL podemos criar, alterar e remover objetos (tabelas) no banco de dados.

DCL

- (Linguagem de Controle de Dados) é o subconjunto da **SQL** utilizado para controlar o acesso aos dados, basicamente com dois comandos que permite ou bloqueia o acesso de usuários a dados;

SQL X MYSQL

- Só para constarmos o MySQL não é uma extensão do SQL.
- O MySQL é um Sistema de Gerenciamento de Banco de Dados
- O SQL é a linguagem para manipulação dos dados no SGBD.

SQLX MYSQL

- Para utilizar as características e o funcionamento do SQL é preciso se servir de um Sistema de Gerenciamento de Bancos de Dados (SGBD), isto é, de um ambiente no qual possamos utilizar os comandos desta linguagem para manipular dados.

SQL--REGRAS
REGRAS

- Todas as palavras-chave das instruções **SQL serão escritas em maiúsculo**;
- Sempre no final de cada instrução, deve **ser terminado com um ponto-e- virgula** (;)

INSTALAR MYSQL

- Existem alternativas para conseguir o MySQL em seu computador.
  - Baixar o MySQL no seu site e instala-lo;
  - (ou) Instalar pacotes que venham com o MySQL incluso, caso do XAMPP e WAMP;
  - MySQL Workbench;

PRIMEIRO ENCONTRO

- Abrir o Prompt Comando do Windows.
- Atalho: Win + R
- Executar: **cmd**
- **Go! Go! Go!**

PRIMEIRO ENCONTRO

- Acessar o diretório c:/xampp/mysql/bin pelo prompt

- Usar o comando:

cd xampp/mysql/bin

CONEXÃO COM MYSQL

- Precisamos utilizar um comando para acessar o prompt do MySQL.
- Ao instalarmos o MySQL é obrigatório criar um usuário e senha para o acesso dos Banco de Dados. **Por padrão**, o usuário é **root** e a senha é vazia.
- Estas informações (usuário e senha) são necessários para este passo.

CONEXÃO COM MYSQL

- O comando para acessarmos o MySQL é:

  mysql –u **usuario** –p **senha**

- **Em nosso caso ficando:**

  mysql –u root –p

Conexão realizada!

```
C:\xampp\mysql\bin>mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.1.41 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

SQL

AGORA NÓS VAMOS PRATICAR

CRIAR UM BANCO DE DADOS

- Para criar de um banco de dados o comando é simples.

mysql> **CREATE DATABASE** meu-banco**;**

CREATE DATABASE seguido do nome desejado de banco de dados.

MOSTRAR BANCO DE DADOS

- Podemos verificar rapidamente a existência do BD recém-criado, bem como a de todos os outros criados anteriormente, utilizando a instrução SHOW DATABASES (mostrar bancos de dados);

mysql> **SHOW DATABASES;**

CRIAR BANCO DE DADOS

- SE NÃO EXISTIR...
- Para verificar se exibe um determinado banco de dados antes da criação de um novo. O comando utilizado é:

mysql> **CREATE DATABASE IF NOT EXISTS**
meu-banco;

DELETAR UM BANCO DE DADOS

- Para excluir um banco de dados, usa-se o comando DROP DATABASE, seguido do nome do banco de dados que deseja deletar.

```
mysql> DROP DATABASE meu-banco;
```

CUIDADO AO DELETAR

- É preciso ressaltar que, **ao apagar um banco de dados, todas as suas tabelas e os dados nelas contidos também serão apagados** e, portanto, perdidos de maneira irreversível.

- ENTÃO, CUIDADO! ☺

Alguém pode me dizer?

- Partindo do conceito que vimos que o SQL é divido em três grupos. Estes comandos que utilizamos se enquadram em qual deles?

a) DML

b) DDL

c) DCL

Alguém pode me dizer?

- Partindo do conceito que vimos que o SQL é divido em três grupos. Estes comandos que utilizamos se enquadram em qual deles?

a) DML

b) **DDL**

c) DCL

USAR UM BANCO DE DADOS

- Como vimos, podemos criar vários bancos de dados, porém, podemos manipular apenas um por vez. Assim, antes de começar, é preciso selecionar qual será o banco de dados que queremos alterar.
- Isso é feito utilizando o comando USE (“usar” em inglês), seguido pelo nome do banco de dados em questão.

USAR UM BANCO DE DADOS

```
mysql> USE meu-banco;
```

CRIAR UMA TABELA

- A regra base do comando para criar uma tabela no banco de dados é o comando para criar tabela, seguido do nome da tabela.
- Também é necessário informar os campos da tabela, seu tipo e seu tamanho.

CRIAR UMA TABELA

```
mysql> CREATE TABLE cadastro
(
        nome CHAR (15),
        sobrenome CHAR (20)
);
```

CRIAR UMA TABELA

```
mysql> CREATE TABLE cadastro
(
        nome CHAR (15),
        sobrenome CHAR (20)
);
```

NOME DA TABELA

CAMPO DA TABELA

TIPOS DE DADOS

TAMANHO DO CAMPO

TIPOS DE CAMPOS

- Existem vários tipos possíveis de dados no SQL, embora os mais comuns sejam:
- **INT** ou **INTEGER**: Para inteiros de tamanho normais
- **TIMESTAMP**: Para data e hora e pode ser atribuídos automaticamente;
- **CHAR** e **VARCHAR**: Para caracteres até no max 255 de tamanho;

MOSTRAR TABELA

- Para exibir a lista de tabelas do banco de dados que está usando atualmente, basta utilizar o comando:

mysql> **SHOW TABLES;**

MOSTRAR ESTRUTURA DA TABELA

- Podemos também analisar a estrutura de uma tabela de maneira aprofundada usando o comando DESCRIBE ("descrever", em inglês), seguido pelo nome da tabela.

mysql> **DESCRIBE** minha-tabela**;**

INSERIR VALORES NA TABELA

- O comando de INSERIR é um dos mais utilizados. Para inserir valores em uma determinada tabela, basta seguir a regra:

mysql> **INSERT** nome_da_tabela **VALUES** (‘valor1’, ‘valor2’, ...);

INSERIR VALORES NA TABELA

mysql> **INSERT** tabela (campo1, campo2, campo3, ...) **VALUES** (“valor1”, “valor2”, “valor3”);

SELECIONAR VALORES DA TABELA

- É possível selecionar valores da tabela, utilizando o comando SELECT do SQL. O comando SELECT é, basicamente, a ferramenta principal para consultar informações de um banco de dados, por isso, é comumente chamado de query.

mysql> **SELECT** dados_desejados **FROM** nome_tabela;

SELECIONAR VALORES DA TABELA

- Podemos definir alguns critérios na seleção de dados. Há duas possíveis alternativas para estes critérios, a utilização de um asterisco (*) e da interrogação (?);

ASTERISCO (*)

- Significa tudo, ou seja, todos os dados. Pode ser combinado com um ou mais caracteres para especificar conjuntos de dados com algo em comum, por exemplo, em geral, se digitarmos o critério A* significa que queremos ver todos os registros cujo conteúdo começa com a letra A;

INTERROGAÇÃO (?)

- Representa um caractere desconhecido. Por exemplo, se definirmos como critério o valor ?????, quer dizer que queremos ver somente os registros que, em determinado campo, contenham valores de cinco caracteres.

ALTERAR TABELA

- Para alterar uma tabela, basta utilizar **ALTER TABLE**, o nome da tabela o qual quer alterar e qual operação de alteração quer fazer.

- Operações: Adicionar novo campo, renomear nome da tabela e etc.

  As operações estão em cores destacadas.

RENOMEAR, ADICIONAR E MODIFICAR

```
mysql> ALTER TABLE pessoas RENAME
TO cadastros;
```

```
mysql> ALTER TABLE pessoas ADD
idade INT(3);
```

```
 mysql> ALTER TABLE pessoas MODIFY
idade INT(5);
```

DELETAR E ORDENAR

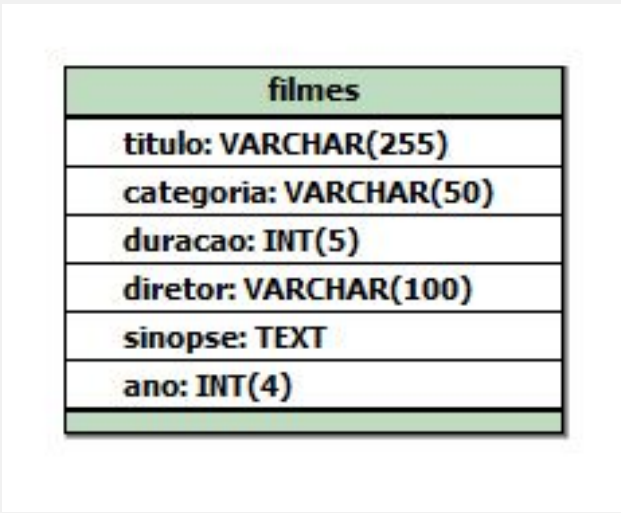
mysql> **ALTER TABLE** pessoas **DROP** cadastros;

mysql> **ALTER TABLE** pessoas **ADD** idade **INT**(3) **AFTER** campo;

mysql> **ALTER TABLE** pessoas **ADD** idade **INT**(3) **FIRST**;

“EXERCÍÇO”

- Crie um banco de dados chamado **cinema**.
- Cria a tabela **filmes**:
- Insira 5 registro;
- Mostre apenas os campos **titulo, duração e ano** dos filmes cadastrados;



cmd

cd /

cd xampp/mysql/bin

(xampp)

cd wamp/bin/mysql/mysql5.5.8/bin

(wamp)

mysql –u root -p

- CRIAR BANCO DE DADOS cadastro;
- CRIAR TABELA pessoas: id, nome, idade;
- CRIAR TABELA times: id, time;

- TODO ID É **PRIMARY KEY NOT NULL AUTO_INCREMENT**

OPÇÕES DOS CAMPOS

- Alguns campos podem ter particularidades. Por exemplo, ser chave primária, não pode ser vazia e etc. Veremos algumas opções.

NOTNULL
NULL

- O campo com a opção NOT NULL, significa que o campo não poderá ser nulo. Para utilizar isso, basta na criação do campo adicionar NOT NULL na frente dele.

mysql> **CREATE TABLE** pessoas (nome **VARCHAR(255) NOT NULL**);

PRIMARY KEY

- Para definirmos que um campo é chave primária, utilizamos a opção PRIMARY KEY, após o nome do campo.

mysql> **CREATE TABLE** pessoas (id **INT(5) PRIMARY KEY**);

AUTO INCREMENT

- Auto incremento, significa que a cada registro de uma tabela, o valor será incrementado (aumentado). Geralmente, utilizamos para campos ID, CODIGO ou CHAVES PRIMARIAS;

AUTO INCREMENT

```
mysql> CREATE TABLE animals (id
INT(5) NOT NULL PRIMARY KEY
AUTO_INCREMENT, name
VARCHAR(50) NOT NULL);
```

CLAUSULA WHERE

- Usando a cláusula **WHERE**, podemos especificar um critério de seleção para selecionar os registros necessários de uma tabela.

CLAUSULA WHERE

- O **WHERE** funciona como uma condição em qualquer linguagem de programação. Esta cláusula é usada para comparar determinado valor com o valor do campo disponível na tabela MySQL. Exemplo:

  SELECIONE campo_x DA tabela_y ONDE campo_x seja igual ao valor

CLAUSULA WHERE

```
mysql> SELECT * FROM pessoas WHERE
id=1;
```

CLAUSULA WHERE

Operador	Descrição	Exemplo
=	Verifica se os valores dos dois operadores são iguais ou não, se sim, então condição torna-se verdade.	(A = B) não é verdade.
! =	Verifica se os valores de dois operandos são iguais ou não, se os valores não são iguais então a condição torna-se verdade.	(A! = B) é verdadeiro.
>	Verifica se o valor do operando esquerdo é maior que o valor do operando da direita, se sim, então a condição se torna verdadeira.	(A> B) não é verdade.
<	Verifica se o valor do operando esquerdo é menor que o valor do operando da direita, se sim, então condição torna-se verdade.	(A <B) é verdadeiro.
> =	Verifica se o valor do operando esquerdo é maior ou igual ao valor do operando da direita, se sim, então a condição se torna verdadeira.	(A> = B) não é verdade.
<=	Verifica se o valor do operando esquerdo é menor ou igual ao valor do operando da direita, se sim, então condição torna-se verdade.	(A <= B) é verdadeiro.

LIMITAR

- Pode-se limitar a quantidades de registros. Se não queremos uma lista extensa e só precisamos das 5 primeiras, coloca-se o LIMIT de 5.

```
mysql> SELECT * FROM pessoas LIMIT 5;
```

ORDENAR

- Quando for necessário ordenar a limita a lista de registros em ordem crescente (ASC) ou decrescente (DESC).

- Para utilizar a ordenação, precisa escolher por qual campo será feita a ordenação.

ORDERNAR

ORDEM DECRESCENTE

mysql> **SELECT * FROM** pessoas **ORDER BY** idade **DESC**.

ORDEM CRESCENTE

mysql> **SELECT * FROM** pessoas **ORDER BY** idade **ASC**.

LIKE

- O LIKE é usado para fazer buscas por partes de conteúdos. Por exemplos, precisamos capturar todas as pessoas com que tem Ana no nome, utilizamos do seguinte código:

LIKE

```
mysql> SELECT * FROM pessoas
WHERE nome LIKE '%ana%' LIMIT 2;
```

- O LIKE é utilizado da seguinte forma:

  LIKE %conteudo%

UNIR TABELA TABELA

- Pode-se unir duas tabelas ou mais, juntas. Para isso utiliza-se o INNER JOIN.

```
mysql> SELECT * FROM pessoas INNER JOIN times;
```

UNIR TABELAS

mysql>

```
SELECT pessoas.nome, times.time
FROM pessoas
INNER JOIN times ON pessoas.time_id
= times.id LIMIT 5;
```

UNIR TABELAS

```
mysql> SELECT tabela1.campo1,
tabela1.campo2, tabela2.campo1
FROM tabela1
INNER JOIN tabela2 ON tabela1.campo  1
= tabela2.campo1;
```