



IN

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 15 - SQL I

O QUE IREMOS APRENDER

- 01** CONTEXTUALIZAÇÃO DA AULA DE HOJE
- 02** BANCO DE DADOS
- 03** SQL
- 04** CONSTRAINTS
- 05** TIPOS DE BANCO DE DADOS
- 06** SGBDS
- 07** MYSQL
- 08** CRIANDO UM BANCO DE DADOS
- 09** COMANDOS SQL (CREATE, ALTER, DROP, TRUNCATE)

Contextualização da aula de hoje

Usar um banco de dados é essencial por várias razões em ambientes onde é necessário **armazenar** e **gerenciar** dados de forma eficiente.

Os bancos de dados desempenham um papel crítico em organizações e aplicativos modernos, pois fornecem uma estrutura robusta para armazenar, acessar e proteger os dados. Eles facilitam a tomada de decisões informadas, o compartilhamento de informações, garantem a segurança dos dados e promovem a eficiência operacional.

Banco de Dados

Basicamente, um banco de dados é uma planilha, como a do excel, gerenciada por um software, entretanto, além do software. há outras diferenças:

- Como os dados são armazenados e manipulados
- Quem pode acessar os dados
- Quantos dados podem ser armazenados

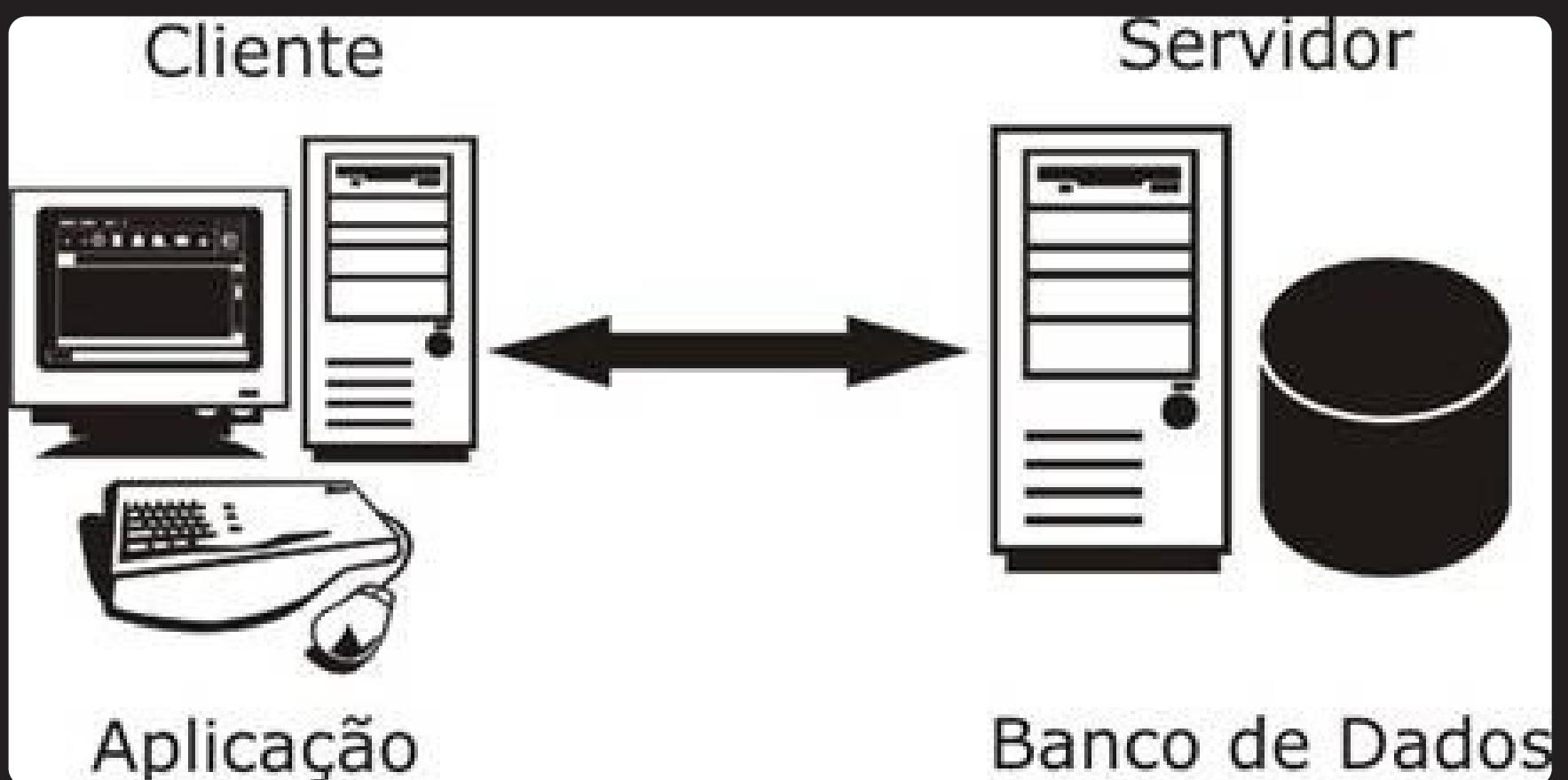


Banco de Dados

Banco de Dados

Um banco de dados é uma coleção organizada de informações, ou dados, estruturados, geralmente armazenados eletronicamente em um sistema de computador.

Normalmente, um banco de dados é controlado por um Sistema de Gerenciamento de Banco de Dados (SGBD).



SQL

O SQL em Python refere-se à integração da linguagem de consulta estruturada (SQL) com a linguagem de programação Python. SQL é uma linguagem utilizada para **gerenciar bancos de dados**, permitindo a criação, modificação e consulta de dados armazenados em um banco de dados relacional. Ao combinar SQL com Python, os desenvolvedores podem aproveitar a capacidade do Python para manipular dados e a flexibilidade do SQL para realizar operações específicas em um banco de dados.

SQL

O **SQL** em Python refere-se à integração da linguagem de consulta estruturada (SQL) com a linguagem de programação Python. SQL é uma linguagem utilizada para **gerenciar bancos de dados**, permitindo a criação, modificação e consulta de dados armazenados em um banco de dados relacional.

Ao combinar SQL com Python, os desenvolvedores podem aproveitar a capacidade do Python para manipular dados e a flexibilidade do SQL para realizar operações específicas em um banco de dados.

Tipos de Banco de Dados

01

Bancos de dados relacionais

Os itens são organizados como um conjunto de tabelas com colunas e linhas.

02

Bancos de dados orientados a objetos

As informações são representadas na forma de objetos, como na programação orientada a objetos.

03

Bancos de dados distribuídos

Consiste em dois ou mais arquivos localizados em sites diferentes. Pode ser armazenado em vários computadores, localizados no mesmo local físico ou espalhados por diferentes redes.

04

Data warehouses

Um repositório central de dados, um data warehouse é um tipo de banco de dados projetado para consultas e análises rápidas.

05

Bancos de dados autônomos

São baseados em nuvem e usam machine learning para automatizar tarefas de gerenciamento de rotina executadas por administradores de banco de dados.

06

Bancos de dados NoSQL

Banco de dados não relacional permite que dados não estruturados e semiestruturados sejam armazenados e manipulados (em contraste com um banco de dados relacional, que define como todos os dados inseridos no banco de dados devem ser compostos).



Tipos de Banco de Dados

07

Bancos de dados gráficos

Os bancos de dados gráficos armazenam estruturas de dados complexas, que seriam incompatíveis em uma base tradicional. É ideal para lidar com dados altamente interconectados.

09

Banco de dados de documentos/JSON

Projetado para armazenamento, recuperação e gerenciamento de informações orientadas a documentos, os bancos de dados de documentos são uma maneira moderna de armazenar dados no formato JSON, em vez de linhas e colunas.

08

Banco de dados multimodelo

Combinam diferentes tipos de modelos de banco de dados em um back-end único e integrado. Isso significa que eles podem acomodar vários tipos de dados.

10

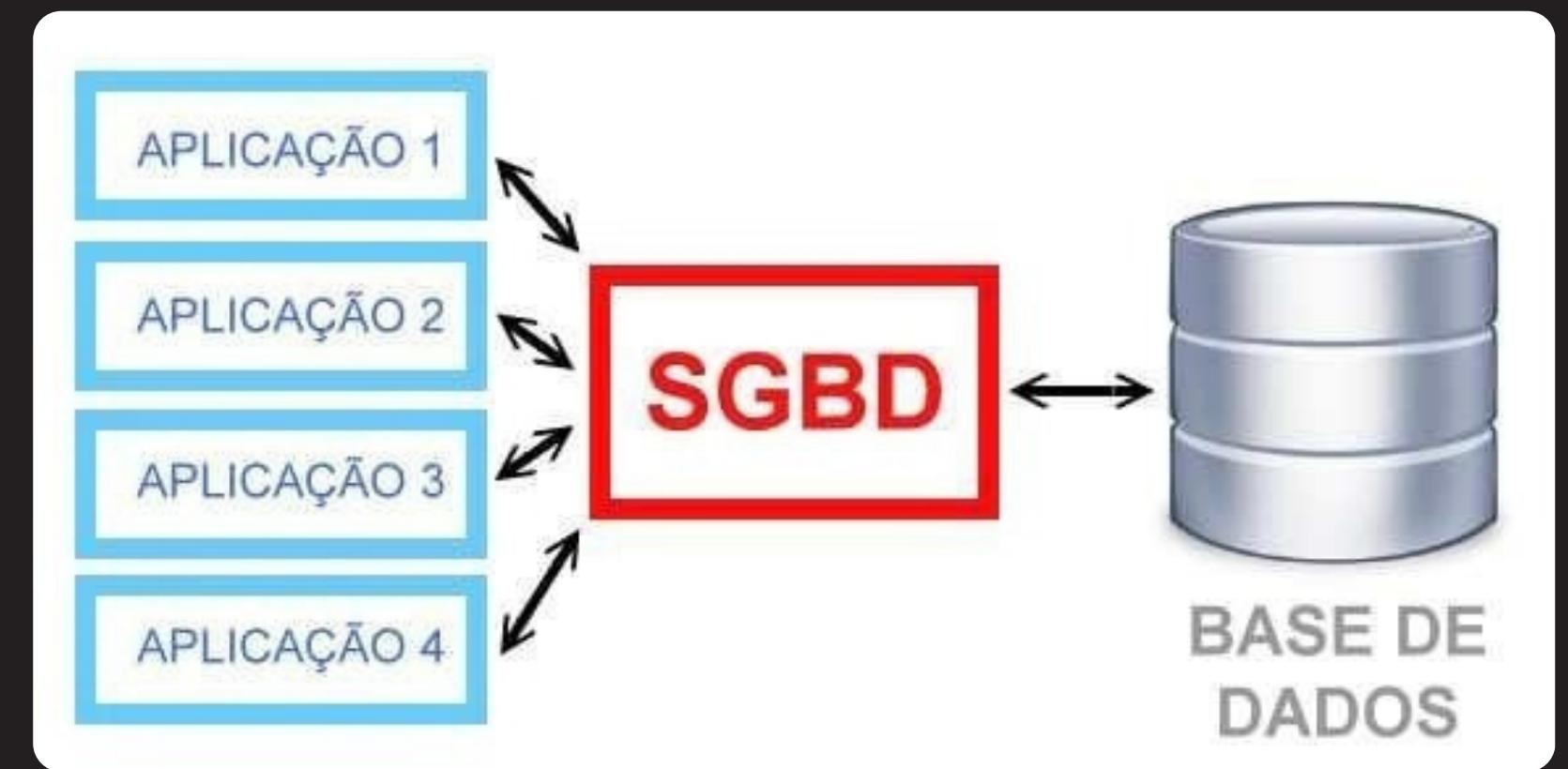
Bancos de dados em nuvem

É uma coleção de dados, estruturados ou não estruturados, que residem em uma plataforma de computação em nuvem privada, pública ou híbrida.



SGBDS

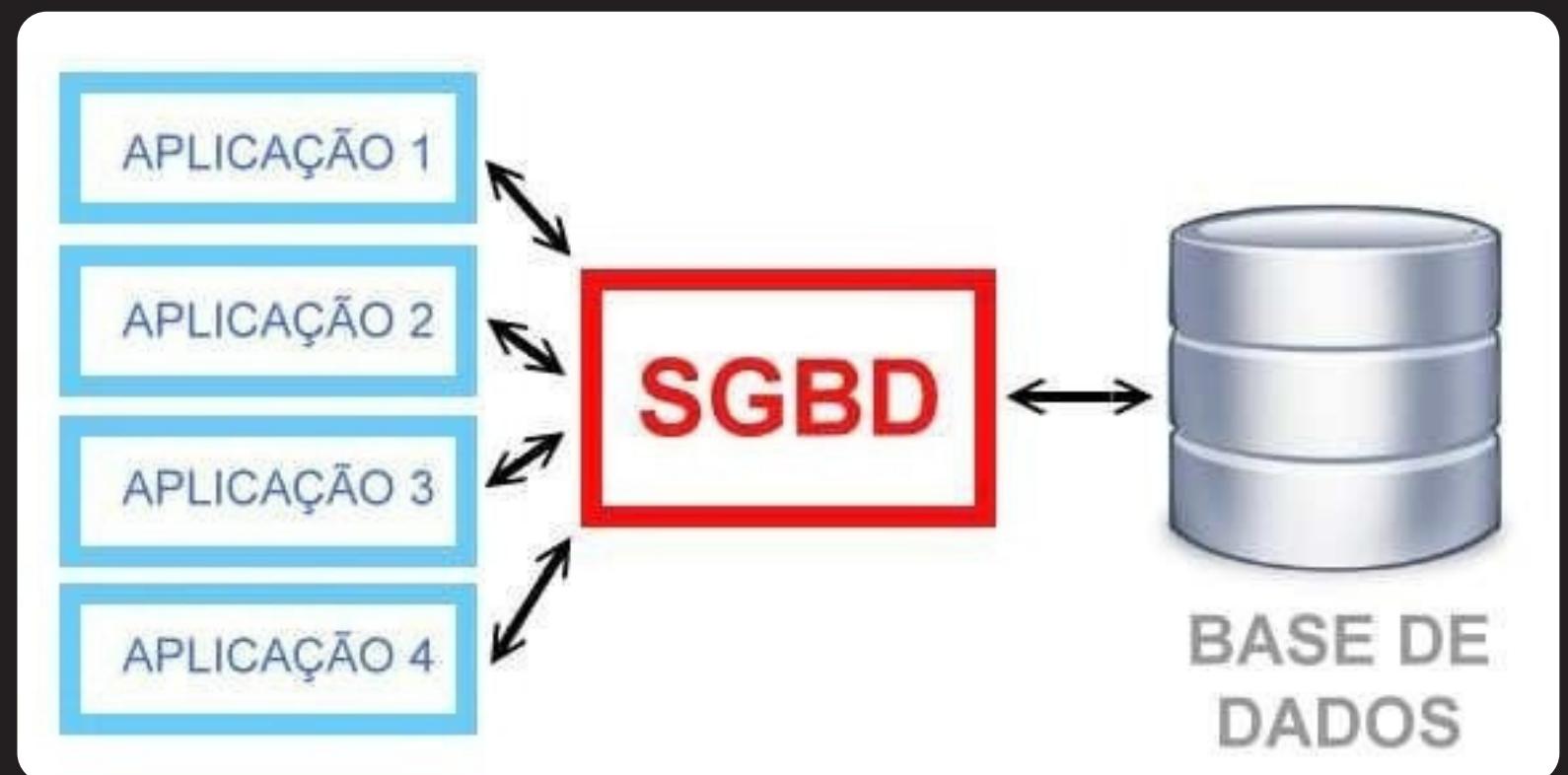
Um banco de dados normalmente requer um programa abrangente de banco de dados, conhecido como sistema de gerenciamento de banco de dados (SGBD). Um SGBD serve como uma **interface entre o banco de dados e seus usuários finais ou programas**, permitindo que os usuários recuperem, atualizem e gerenciem como as informações são organizadas e otimizadas.



SGBDS

Um SGBD também facilita a supervisão e o controle de bancos de dados, permitindo uma variedade de operações administrativas, como monitoramento de desempenho, ajuste e backup e recuperação.

Alguns exemplos de softwares de bancos de dados populares ou SGBD incluem MySQL, Microsoft Access, Microsoft SQL Server, FileMaker Pro, Oracle Database e dBASE.



MYSQL

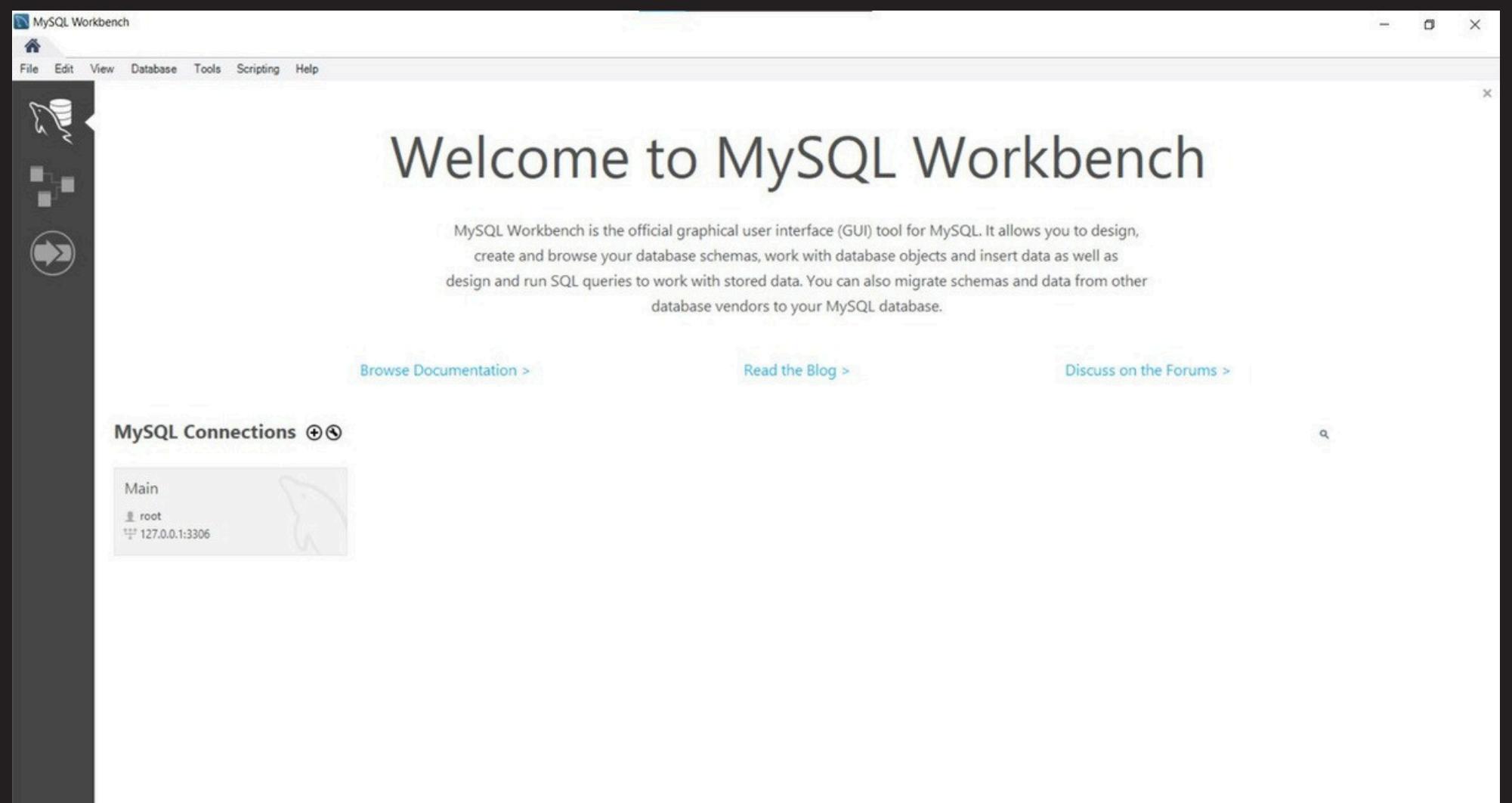
O MySQL é o SGBD por trás de alguns dos principais sites e aplicativos baseados na web do mundo, incluindo Uber, LinkedIn, Facebook, Twitter e YouTube.

O MySQL Workbench é uma **ferramenta visual de design de banco de dados** que integra desenvolvimento, administração, design, criação e manutenção de banco de dados SQL em um único ambiente de desenvolvimento integrado para o sistema de banco de dados MySQL.



MYSQL

O MySQL Workbench é uma ferramenta gráfica para desenvolvedores e administradores de bancos de dados MySQL. Ele permite criar, gerenciar e consultar bancos de dados, além de oferecer recursos para modelagem de dados, administração de usuários e migração de dados. Em resumo, é uma ferramenta abrangente para trabalhar com MySQL.



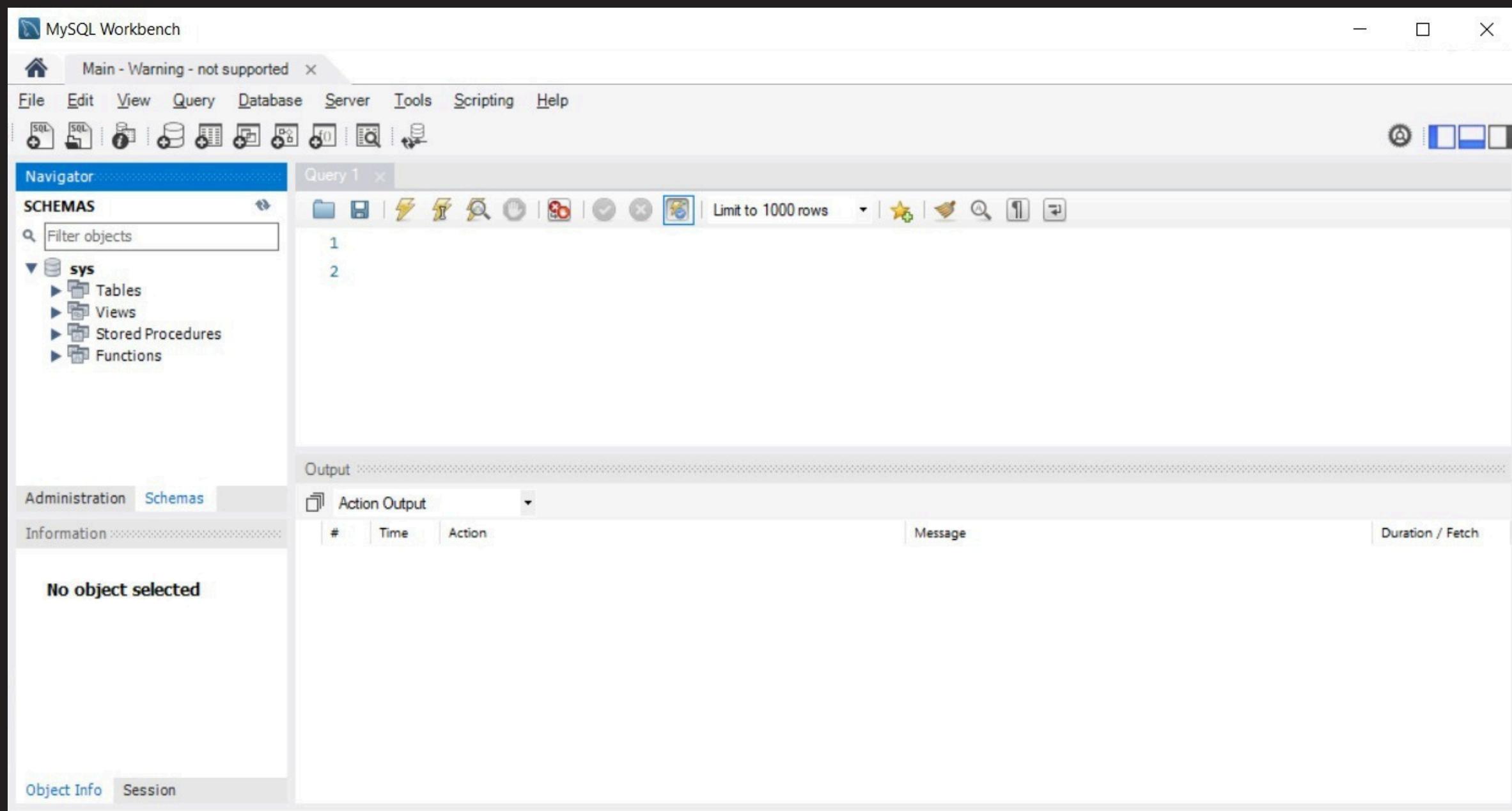
MYSQL

O workbench configura um servidor na nossa máquina onde podemos criar bancos de dados, executar códigos SQL, entre outras funcionalidades. Essa conexão é chamada de "Local Instance".

O usuário por padrão é o usuário root.

Esse servidor roda no nosso localhost, na porta 3306 por padrão.

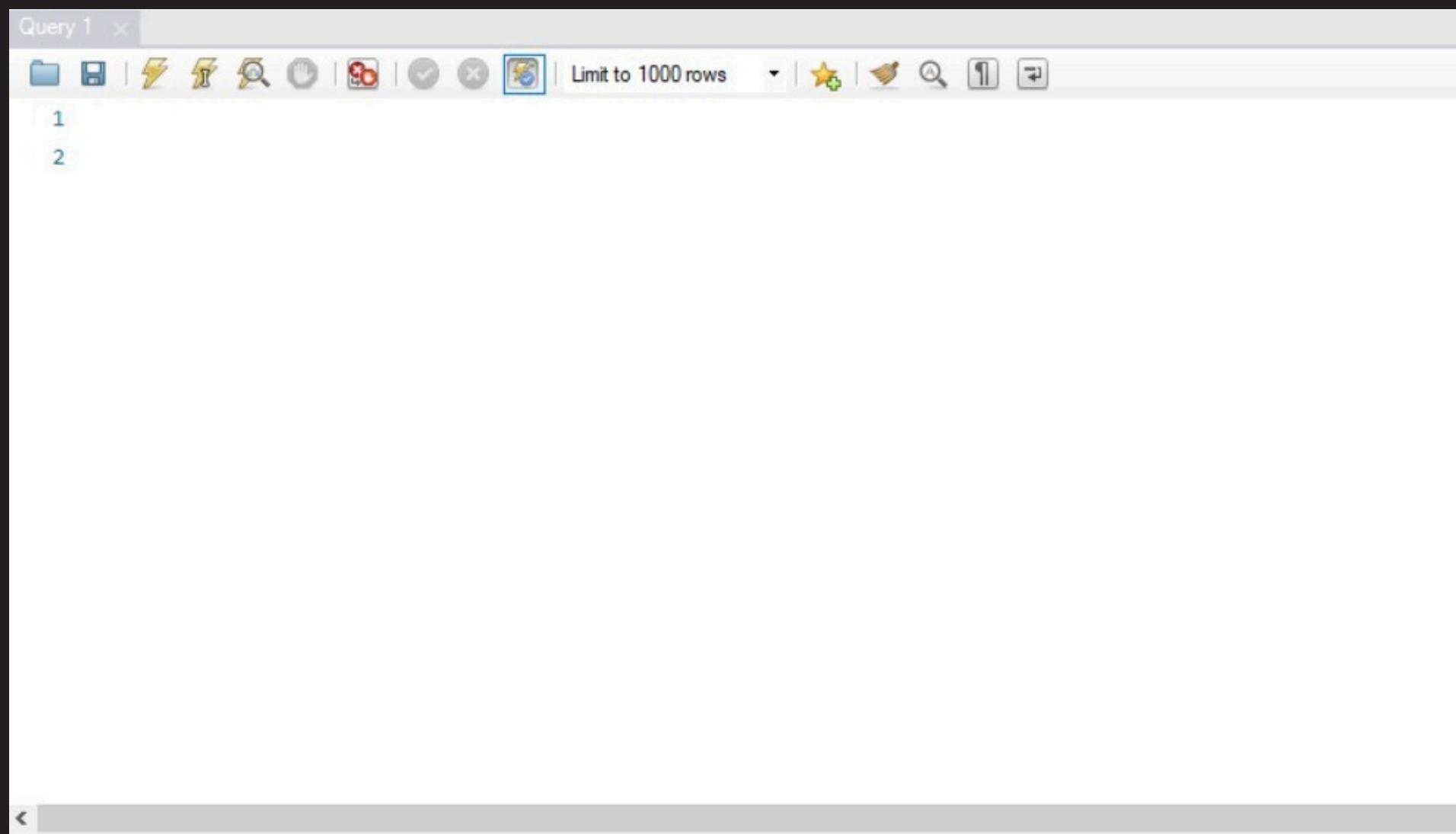
MYSQL



Ao clicar nessa conexão e logar pelo usuário root, vemos essa tela.

IN

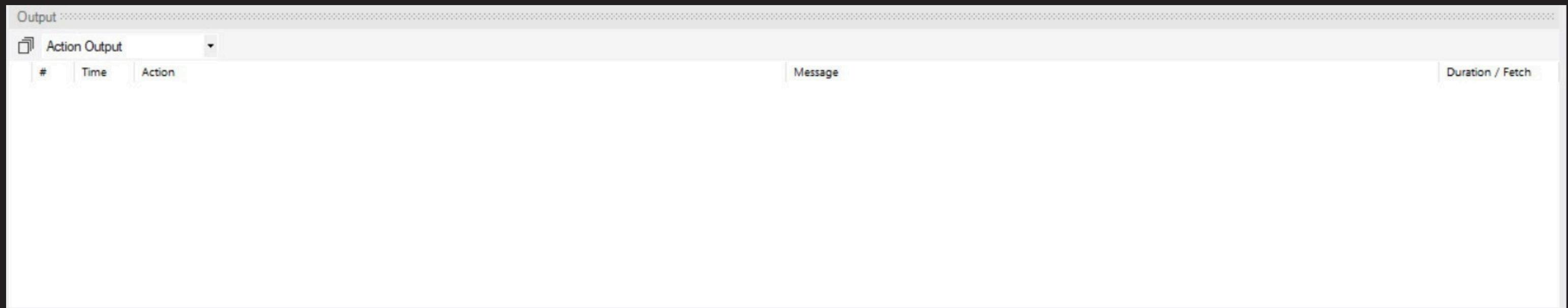
MYSQL



Na parte central da tela temos o ambiente onde escreveremos os código SQL, salvaremos e executaremos.

MYSQL

Na parte inferior temos as saídas, podemos visualizar o resultado da execução dos códigos SQL por exemplo.



MYSQL



No lado esquerdo, temos um menu de navegação com algumas opções importantes para o gerenciamento, performance e configuração. Na parte inferior termos duas abas, "Administration" e "Schemas".

MYSQL

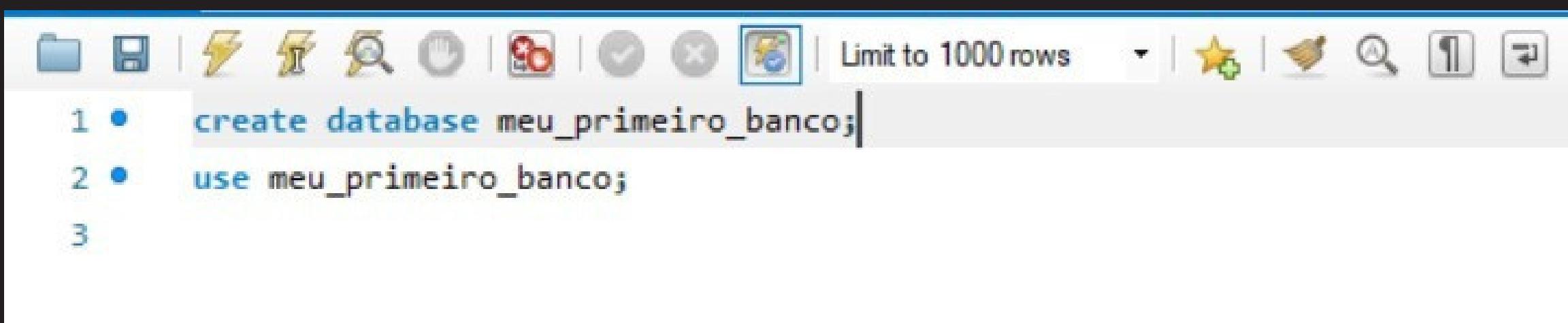


Na aba "Schemas" temos os bancos de dados criados. Dentro de cada Schema conseguimos ver as tabelas, as views, as stored Procedures e as Functions.

Criando um Banco de Dados

Primeiro comando SQL

Para criar nosso primeiro banco de dado (schema), utilizamos o comando:
create database nome_do_banco.



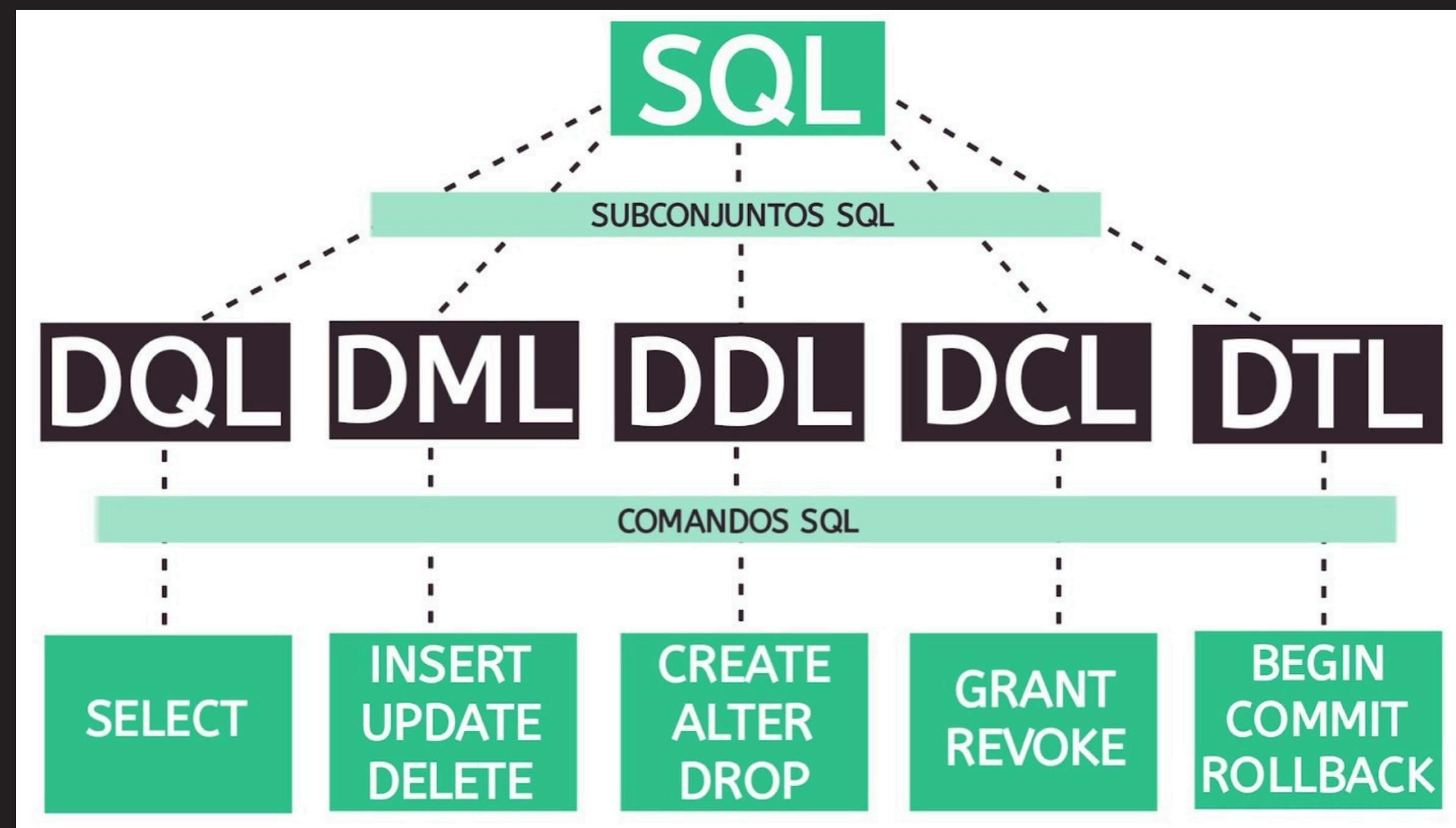
A screenshot of the MySQL Workbench interface. The toolbar at the top has various icons for database management. Below the toolbar, there is a query editor window. The first line of code in the editor is '1 • create database meu_primeiro_banco;'. The second line is '2 • use meu_primeiro_banco;'. The third line is '3'. The 'create database' command is highlighted in blue, indicating it is the currently selected or active command.

Lembrando que esse comando irá apenas criar o banco de dados, para utilizarmos um banco de dados criado, utilizamos: **use nome_do_banco**

Comandos SQL

o SQL é uma linguagem para se trabalhar com banco de dados relacionais. Com o SQL podemos gerenciar os nossos dados em um banco de dados através das querys, ou seja, através das requisições. Vamos entender que temos diferentes grupos de comandos para fazer diferentes requisições ao SGBD. Cada grupo de comando também será responsável por uma letra do **CRUD**

Comandos SQL



IN

Tipos de Comandos

01

CREATE

Criar ou adicionar novas entradas

02

READ (RETRIEVE)

Ler, recuperar ou ver entradas existentes

03

UPDATE

Atualizar ou editar entradas existentes

04

Delete (Destroy)

Remover entradas existentes

IN

Comandos SQL

Os comandos DDL (Data Definition Language) são estruturais. Eles servem para definir aspectos da estrutura do banco de dados, como criar tabelas, excluir tabelas, alterar tabelas e mais. São mais focados em um aspecto amplo, e menos nos dados necessariamente.

Ou seja, esses comandos servirão para a estrutura do banco de dados e não para os dados.

Comandos SQL

Os comandos DDL normalmente não são usados por um usuário geral, pois, nesse cenário, um usuário seria capaz de mexer na estrutura do banco de dados. Fazem parte dos comandos DDL: **CREATE, DROP, ALTER, TRUNCATE, COMMENT, RENAME**

Comandos SQL - CREATE



```
1 CREATE DATABASE IF NOT EXISTS name_db
2
3 -- Sintaxe Básica
4
5 -- O banco será criado se não existir um
6 -- banco de dados com o mesmo nome
```

Existem duas instruções CREATE disponíveis no SQL:

- CREATE DATABASE
- CREATE TABLE

O comando CREATE DATABASE irá criar um novo banco de dados

Comandos SQL - CREATE

Com o banco de dados criado, precisamos criar as tabelas, para armazenar os dados.

O comando `CREATE TABLE` é usado para criar uma tabela. Sabemos que uma tabela é composta por linhas e colunas, portanto, ao criar tabelas, precisamos fornecer todas as informações ao SQL sobre os nomes das colunas, tipos de dados a serem armazenados nas colunas, tamanho dos dados, etc.



```
1 CREATE TABLE nome_tabela(  
2     coluna_01 tipo_dado(tamanho)  
3     coluna_02 tipo_dado(tamanho)  
4     coluna_03 tipo_dado(tamanho)  
5 )  
6 -- Sintaxe Básica  
7  
8 CREATE TABLE Usuario(  
9     id int,  
10    nome varchar(40) -- Tamanho máximo  
11    esta_ativo boolean  
12 )  
13  
14 -- Exemplo
```

Comandos SQL - ALTER TABLE

Usado para modificar a estrutura de uma tabela existente, como adicionar, modificar ou excluir colunas.



```
1 ALTER TABLE alunos  
2 CHANGE COLUMN curso curso_atual varchar(50);
```

Este comando renomeia a coluna "curso" para "curso_atual" e altera o tipo de dados para VARCHAR(50).



```
1 ALTER TABLE alunos  
2 ADD COLUMN data_matricula DATE;
```

Este comando adiciona uma nova coluna chamada "data_matricula" à tabela "alunos", que armazenará datas de matrícula.

Comandos SQL - DROP

O comando DROP é utilizado para excluir um banco de dados inteiro ou apenas uma tabela.

A instrução DROP destrói os objetos como um banco de dados, tabela, índice ou visão.



- 1 `DROP TABLE nome_tabela`
- 2 `DROP DATABASE nome_bd`

Comandos SQL - TRUNCATE

A instrução TRUNCATE é usada para remover os dados de uma tabela para fins de deslocação (vazia para reutilização).

O resultado dessa operação remove rapidamente todos os dados de uma tabela, geralmente ignorando vários mecanismos de imposição de integridade.

A instrução TRUNCATE TABLE é logicamente equivalente à instrução DELETE FROM (sem a cláusula WHERE).



```
1 TRUNCATE TABLE nome_tabela
2
3 -- Ao executar a comando acima,
4 -- a tabela será truncada ou seja,
5 -- os dados serão excluídos, mas a
6 -- estrutura permanecerá na memória
7 -- para operações posteriores
```

Comandos SQL - TRUNCATE X DROP

Truncar preserva a estrutura da tabela para uso futuro, ao contrário de DROP TABLE onde a tabela é excluída com sua estrutura completa. A exclusão da tabela ou banco de dados usando a instrução DROP não pode ser revertida, portanto, deve ser usada com cautela.

Comandos SQL - CONSTRAINTS

Você pode aplicar restrições, como chaves primárias, chaves estrangeiras, restrições únicas e verificação, para garantir a integridade dos dados.



Comandos SQL - CONSTRAINTS

Chave Primária (Primary Key): Uma constraint de chave primária é usada para identificar de forma exclusiva cada registro em uma tabela. Ela garante que os valores em uma coluna (ou um conjunto de colunas) sejam únicos e não nulos.



```
1 CREATE TABLE alunos(  
2     id_aluno INT PRIMARY KEY  
3     nome VARCHAR(50)  
4 )
```

Comandos SQL - CONSTRAINTS

A chave estrangeira (Foreign Key) estabelece uma relação entre duas tabelas, garantindo que os valores em uma coluna de uma tabela correspondam aos valores de uma coluna em outra tabela, geralmente a chave primária. Isso assegura a integridade referencial, permitindo que os dados relacionados sejam consistentes e organizados.

```
1 CREATE TABLE Turmas (
2     Turma_ID INT PRIMARY KEY,
3     Nome_Turma VARCHAR(50) NOT NULL
4 );
5
6 CREATE TABLE Alunos (
7     Aluno_ID INT PRIMARY KEY,
8     Nome_Aluno VARCHAR(50) NOT NULL,
9     Turma_ID INT,
10    FOREIGN KEY (Turma_ID) REFERENCES Turmas(Turma_ID)
11 );
```

Comandos SQL - CONSTRAINTS

Restrição Única (Unique Constraint):

Uma restrição única garante que os valores em uma coluna sejam únicos, mas não necessariamente que sejam não nulos. Ela pode ser usada para impedir a inserção de duplicatas.



```
1 CREATE TABLE produtos (
2     codigo_produto INT UNIQUE,
3     nome_produto VARCHAR(100)
4 );
```

Comandos SQL - CONSTRAINTS

Restrição de Verificação (Check Constraint): Uma restrição de verificação permite especificar uma condição que os valores em uma coluna devem atender. Isso pode ser usado para garantir que os dados estejam dentro de um intervalo específico.



```
1 CREATE TABLE funcionarios (
2     id_funcionario INT,
3     salario DECIMAL(10, 2),
4     CHECK (salario >= 0)
5 );
```

Comandos SQL - CONSTRAINTS

Restrição de Não Nulo (Not Null Constraint):
Uma restrição de não nulo garante que uma coluna não pode conter valores nulos, ou seja, todos os registros devem ter valores para essa coluna.



```
1 CREATE TABLE clientes (
2     id_cliente INT,
3     nome VARCHAR(50) NOT NULL
4 );
```

ATIVIDADE PRÁTICA 1

Crie um banco de dados chamado "escola" e as seguintes tabelas:

Tabela "alunos" com colunas: id_aluno, nome, idade.

Tabela "cursos" com colunas: id_curso, nome_curso, carga_horaria. Tabela "matriculas" com colunas: id_matricula , id_aluno id_curso, data_matricula.

ATIVIDADE PRÁTICA 2

Usando o comando TRUNCATE TABLE, exclua todos os dados da tabela "matriculas" sem excluir a estrutura da tabela.

ATIVIDADE PRÁTICA 3

Usando o comando DROP DATABASE, exclua o banco de dados "escola". Certifique-se de que você tenha feito um backup dos dados, pois essa ação apagará todo o banco de dados.

DESAFIO PRÁTICO

Sistema de uma escola Crie um banco de dados para um sistema de uma escola, esse banco de dados ficará responsável por persistir os dados sobre alunos, professores, turmas e disciplinas. Para os alunos é importante que contenha um número de matrícula, o nome, a idade, e o endereço.

DESAFIO PRÁTICO

Sistema de uma escola Para os professores, deverá conter um número de matrícula, nome, especialidade e endereço. Para a turma deverá conter um identificador, horário de início e dia de semana.

Para disciplina é importante que contenha um identificador, nome e quantidade de aulas.

SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 16 DE PYTHON:
SQL II



INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

Aula 16 - SQL II

MANIPULAÇÃO DE DADOS(INSERT - UPDATE - DELETE - SELECT)

MÃOS NO CÓDIGO



IN

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 15 - SQL I