



- Conceitos Básicos
- Instalação do MySQL
- Primeiros passos

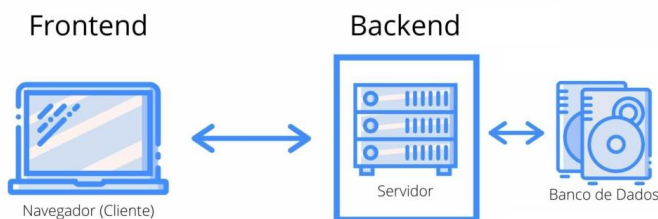
Banco de Dados

Conceitos Básicos

Tópicos Essenciais para Iniciar na Área de Banco de Dados

01 Iniciando

- Antes de iniciar nossa “prosa” sobre banco de dados, vamos entender o que é um banco de dados e onde ele se encaixa em um sistema.
- Como vimos nas folhas anteriores, utilizamos estruturas de dados como listas, tuplas, dicionários ou conjuntos para armazenar os dados. Entretanto, quando paramos o sistema e o executávamos novamente, perdemos todos os dados, concorda? Salvar em arquivos não considero uma opção, pois não entregamos sistemas que salva os dados em arquivos.
- Neste sentido, usamos os bancos de dados, que em uma analogia bem bacana, podem ser pensadas como planilhas do Microsoft Excel, que vai guardar os dados em um determinado formato que, sempre que os dados da aplicação forem manipulados, a planilha é atualizada de modo que, ao reiniciar a aplicação, os dados não são perdidos.
- Mas, em que lugar um banco de dados se encaixa em uma aplicação? A imagem a seguir detalha um pouco do funcionamento.



Fonte: Adaptado de <https://marquesfemendes.com/tecnologia/o-que-e-um-desenvolvedor-backend-e-o-que-de-faz/>

- À esquerda, temos o **frontend**, que é basicamente a interface que o cliente vai ter com o sistema, isto é, a tela, que por sua vez pode ser *web*, *desktop* ou *mobile* (smartphones). Nesta tela, preencheremos formulários, como por exemplo os dados para criação de uma nova conta no AbellaBank.
- Por exemplo, ao clicar no botão “Criar Conta”, os dados preenchidos são remetidos ao **backend**, que por sua vez, os armazena em um banco de dados. É assim que este mundo perfeito funciona!

02 “Leriado” Básico

- Primeiramente, o termo correto é **Sistema de Gerenciamento de Banco de Dados**, conhecido por sua sigla SGBD. Entretanto, quase todo mundo fala equivocadamente e confunde SGBD com **Banco de Dados** (BD).
- SGBD é um conjunto de programas de *software* que permite aos usuários criar, editar, atualizar, armazenar e recuperar dados em tabelas de banco de dados. Por outro lado, BD é uma coleção de dados relacionados.
- Na imagem a seguir creio que podemos tomar as coisas um pouco mais claras.
 - O SGBD é o lugar onde podemos ter vários bancos de dados. Exemplos de SGBD você já pode ter ouvido: Oracle, MySQL, PostgreSQL, Microsoft SQL Server, entre outros.



- Por exemplo, Daniel Abella, utilizo o SGBD MySQL. E, dentro do MySQL, eu crio um banco de dados para o meu sistema de ouvidoria e um banco de dados para cada um das minhas aplicações.
- E, dentro de cada banco de dados, podemos ter várias tabelas, que se parecem em muito com as velhas planilhas do Microsoft Excel. Ficou claro? Um SGBD tem vários BD e cada BD tem várias tabelas.
- Vamos omitir muita teoria e na próxima seção já iniciaremos a parte prática!

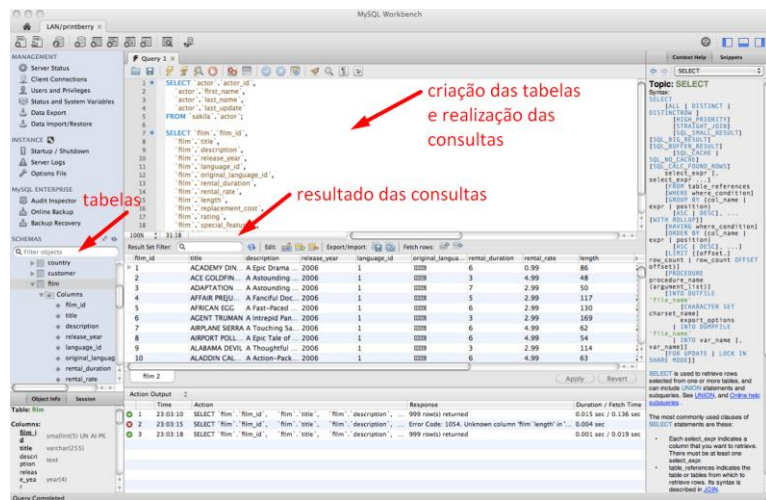
03 Alternativas para usar o MySQL

- Acabooooooooo o leriado! Aqui vamos apresentar duas alternativas de usar o MySQL.

- A primeira delas é utilizando uma ferramenta *online* para os primeiros passos.
- A última alternativa é utilizar as ferramentas apropriadas (e diga-se de passagem, profissionais) para isso.
 - Entenderam que a primeira maneira é só para, digamos, os primeiros dias?
- Cabe destacar que, tudo produzido aqui funciona em qualquer uma das alternativas.

04 Alternativa 1 - BD Fiddle

- **DB Fiddle** é uma ferramenta online que permite você “brincar” com o MySQL, além de outros SGBD como PostgreSQL e SQLite *sem precisar instalar nada*.
 - Lembra que, usamos o Replit para realizar as primeiras “brincadeiras” com Python? Mesma coisa aqui.
- O site é o seguinte <https://www.db-fiddle.com/>
- Uma outra ferramenta online é o **SQL Fiddle**, disponível no endereço <http://sqlfiddle.com/>. Uma boa explicação sobre a ferramenta é o link <https://www.youtube.com/watch?v=ZaOwbRD0qtK> (9 minutos e só).



06 Passo 1 - Criar/Excluir o BD

- Conforme discutimos anteriormente, um SGBD possui vários bancos de dados, que por sua vez, possuem várias tabelas. Explicaremos aqui como criar um banco de dados.



- Para quem estiver ainda usando a ferramenta DB Fiddle, não vai ser necessário realizar este passo, pois ele já cria um automaticamente. Entretanto, para os usuários do Workbench, vai ser necessário.

- Antes de criar um banco de dados, sugere-se verificar os BD que estão criados por meio do comando: **SHOW DATABASES**
- Agora, para criar um BD de nome **ouvidoria** o comando é: **CREATE DATABASE ouvidoria;**
- Caso em algum momento precise excluir o BD criado anteriormente, o comando é **DROP DATABASE ouvidoria;** Lembrando que, uma vez excluído, todas as tabelas que estiverem dentro do BD serão excluídas também.

```
1 SHOW DATABASES
2 CREATE DATABASE ouvidoria;
3 DROP DATABASE ouvidoria;
```

07 Passo 2 - Selecionar o BD Criado

- Na seção anterior criamos um banco de dados, não? Antes de começar a usar, precisamos indicar qual BD vamos utilizar. Para isto, usamos o comando **USE ouvidoria;**

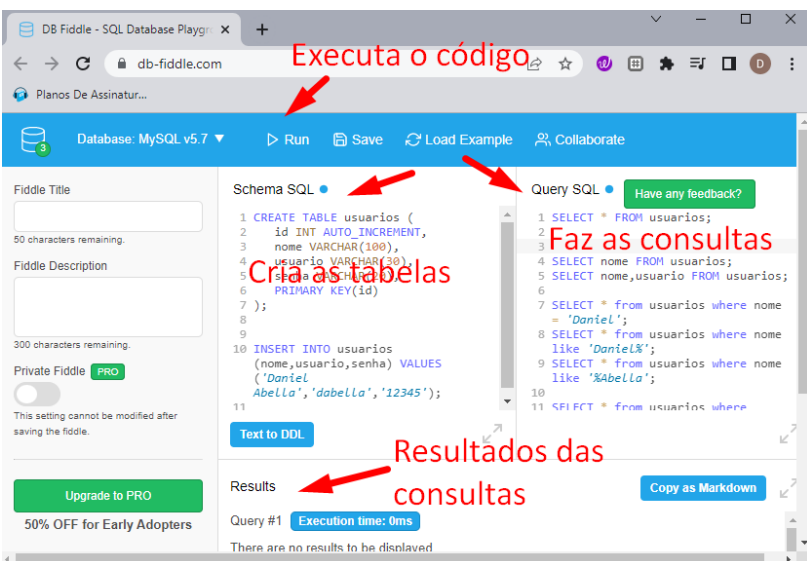
```
1 USE ouvidoria;
```

08 Tipos de Dados em MySQL

- Imagine que vamos precisar criar uma tabela para armazenar todos os dados dos usuários do sistema. Caso não conhecêssemos BD e usássemos o Excel, fariamos algo como o da página a seguir. Note que, temos 4 colunas, sendo elas: ID (identificação), Nome, Usuário e Senha.

05 Alternativa 2 - MySQL e Workbench

- Como dito anteriormente, sugere-se esta segunda alternativa em que o MySQL é instalado na sua máquina. Para isto, siga todos os passos descritos no site <https://www.alura.com.br/artigos/mysql-do-download-e-instalacao-ate-sua-primeira-tabela>
- Além do MySQL, você vai precisar instalar o **Workbench**, que é uma ferramenta para gerenciar os BD do MySQL. Ou seja, do Workbench que vamos criar as tabelas e fazer os comandos necessário para inserir, alterar, excluir ou listar os dados armazenados nas tabelas de um BD.
- Pelo amor de Deus, ao instalar não esqueça de anotar o usuário e senha que você usou para criar o banco de dados!



- E em nosso “banco de dados” temos 3 usuários, sendo o primeiro deles Daniel (nome) com ID possuindo valor 1, usuário dabella e senha 12345.

	A	B	C	D
1	ID	Nome	Usuário	Senha
2		1 Daniel	dabella	12345
3		2 Arthur	aabella	223344
4		3 Nathaly	nabella	334455



Utilizei a coluna ID para que, cada vez que um novo usuário seja inserido na tabela, incremente a coluna ID. Por exemplo, Daniel, como primeiro usuário na tabela, recebeu o ID 1, enquanto que, Arthur, como segundo usuário, teve o ID 2 e assim sucessivamente..

- Em Python não tínhamos os tipos de dados str, int, float, bool, etc? Em BD também temos!
- Apresento a seguir os principais tipos dentre os diversos existentes.

Tipo	Descrição	Exemplos
Int	Int do Python	1, 2
Float	Float do Python	1.1, 1.0
Varchar(N)	Str, onde N é o tamanho máximo da Str	“Oi”, “Daniel Abella”
Boolean	Bool do Python	true, false, TRUE, FALSE, True, False

- Agora que sabemos os tipos de dados, na seção a seguir vamos explicar como tirar do “Excel” para uma tabela do BD.

09 Passo 3 - Criar/Excluir uma Tabela

- Nos passos anteriores, criamos um BD chamado ouvidoria e o selecionamos para uso. Agora, vamos criar a nossa primeira tabela.
- Para criar a tabela, precisamos de duas coisas, o nome da tabela e o nome dos campos (e respectivos tipos).
- Baseando-se no Excel anterior, sugiro que a tabela se chame USUARIOS e que possua 4 campos, sendo eles ID, nome, usuário e senha.
- Para criar esta tabela, o comando a seguir deve ser utilizado, que possui os seguintes detalhes:

- Linha 1:** O nome da tabela é **usuarios**,
 - Entre os parênteses **()**; temos a relação dos campos, neste exemplo, os 4 supracitados
 - Entre os campos, colocamos vírgulas

```
1 CREATE TABLE usuarios (
2   id INT AUTO_INCREMENT,
3   nome VARCHAR(100),
4   usuario VARCHAR(30),
5   senha VARCHAR(20),
6   PRIMARY KEY(id)
7 );
```

- Linha 2:** Criamos um campo chamado **ID** que é um inteiro.
 - Este campo é incrementável, ou seja, a cada registro novo, vai aumentando de 1 em 1
 - Para isso, usamos o termo **AUTO_INCREMENT**
- Linha 3:** Criamos um campo chamado **nome** que é uma String de até 100 caracteres
- Linha 4:** Criamos um campo chamado **usuario** que é uma String de até 30 caracteres
- Linha 5:** Criamos um campo chamado **senha** que é uma String de até 20 caracteres
- Linha 6:** Precisamos definir que coluna não pode repetir valores, ou seja, que é a chave primária (PRIMARY KEY - PK)
 - Neste exemplo, usamos o campo **id** como PK
- Caso queira excluir a tabela acima, basta um **DROP TABLE ouvidoria;**

```
1 DROP
2 TABLE ouvidoria;
```

10 Passo 4 - Inserir Linhas na Tabela

- Uma vez criada a tabela **USUARIOS**, a tabela está vazia, isto é, sem linhas. Para adicionar novos usuários, o comando é o seguinte: **INSERT INTO usuarios (nome,usuario,senha) VALUES ('Daniel Abella','dabella','12345');**

```
1 INSERT INTO usuarios (nome, usuario, senha)
2 VALUES
3 (
4   'Daniel Abella', 'dabella', '12345'
5 );
```

- Note que:
- #1 Informamos o nome da tabela após o comando INSERT INTO
- #2 Após o nome da tabela, entre os parênteses temos o nome das colunas que serão preenchidas
- Se temos 4 colunas, porque não informamos a coluna ID
 - Porque colocamos esta coluna para incrementar (aumentar de 1 em 1) automaticamente
 - Desta maneira, não precisamos especificar este campo, pois “ele se vira sozinho”
 - Caso não tivesse o comando AUTO_INCREMENT, teríamos que especificar

- #3 Na sequência, temos a cláusula VALUES seguida por ();
 - Se em #2 especificamos 3 colunas, dentro do VALUES, colocaremos na mesma ordem os valores correspondentes

- Te desafio a inserir também Arthur e Nathaly para ficar igual ao Excel.

	A	B	C	D
1	ID	Nome	Usuário	Senha
2	1	Daniel	dabella	12345
3	2	Arthur	aabella	223344
4	3	Nathaly	nabella	334455

11 Passo 5 - Listas as Linhas da Tabela

- Vamos neste passo realizar um SELECT, para listar todos os registros inseridos na tabela USUARIOS.
- Na sequência apresentamos um exemplo de alguns cenários comuns no uso de SELECT:

```

1 SELECT * FROM usuarios;
2 SELECT nome FROM usuarios;
3 SELECT nome,usuario FROM usuarios;
4
5 SELECT * from usuarios where nome = 'Daniel';
6 SELECT * from usuarios where nome like 'Daniel%';
7 SELECT * from usuarios where nome like '%Abella';
8
9 SELECT * from usuarios where usuario = 'dabella'
10 and nome = 'Daniel Abella';

```

- **Linha 1:** Apresentamos todos (por isso o *) campos de todos os usuários
- **Linha 2:** Apresentamos apenas o campo nome de todos os usuários
- **Linha 3:** Apresentamos os campos nome e usuário de todos os usuários
- **Linha 5:** Apresentamos todos os campos dos usuários que o campo nome é igual a Daniel
- **Linha 6:** Apresentamos todos os campos dos usuários que o campo nome inicia com Daniel
 - Para isso, usamos o like e, dentro da String, usamos o % para dizer que vem coisa depois
 - Se fosse os que terminassem com Abella, observe a linha 7
- **Linhas 9 e 10:** Apresentamos todos os campos dos usuários que o campo usuário seja dabella e o nome seja Daniel Abella
 - Para isso, usamos o and, como em Python
 - Ainda temos or e muitos outros operadores

12 Passo 6 - Atualizar as Linhas da Tabela

- Na seção anterior, aprendemos a inserir uma linha com registros em uma tabela. E, por alguma situação, precisássemos alterar o valor de uma linha? Para isto, utilizaremos o comando UPDATE, cujo exemplos e comentários estão a seguir.

```

1 UPDATE usuarios
2   SET nome = 'Daniel Abella Souza' WHERE id = 1;
3 UPDATE usuarios
4   SET nome = 'Daniel' WHERE id = 1 or usuario='dsouza';
5 UPDATE usuarios
6   SET nome = 'Dan' and usuario = 'ddan'
7   WHERE id = 1 and usuario='dsouza';

```

- **Linha 1:** Alteramos o campo **nome** para o usuário com **ID 1**
- **Linha 2:** Alteramos o campo **nome** para o usuário com **ID 1** ou **usuário dsouza**
- **Linha 3:** Alteramos os campos **nome** e **usuario** para o **usuário** com **ID 1** e **usuário dsouza**

13 Passo 7 - Excluir as Linhas na Tabela

- Para excluir as linhas da tabela **USUARIOS**, usamos o comando **DELETE**. Abaixo apresentamos os exemplos e as considerações:

```

1 DELETE FROM usuarios;
2 DELETE FROM usuarios WHERE id = 1;
3 DELETE FROM usuarios WHERE id = 1 or usuario='dsouza';
4 DELETE FROM usuarios WHERE id = 1 and usuario='dsouza';

```

- **Linha 1:** Excluimos todos os usuários (veja que não temos *, pois não excluimos uma dada coluna)
- **Linha 2:** Excluimos todos os usuários cujo ID seja 1
- **Linha 3:** Excluimos todos os usuários cujo ID seja 1 ou usuário dsouza
- **Linha 4:** Excluimos todos os usuários cujo ID seja 1 e usuário dsouza

14 E agora?

- Seguramente omitimos uma série de detalhes que serão aprofundados posteriormente. Entretanto, o conhecimento suficiente para iniciar nesta área de banco de dados está relacionado aqui. Na folha seguinte, apresentaremos como fazer essas operações (inserção, listagem, atualização e exclusão) a partir de um código em Python.

15 Resumo dos Comandos

Comando	Função	Exemplos
SELECT	Lista as linhas	<pre> 1 SELECT * FROM usuarios; 2 SELECT nome FROM usuarios; 3 SELECT nome,usuario FROM usuarios; 4 SELECT * from usuarios where nome = 'Daniel'; 5 SELECT * from usuarios where nome like 'Daniel%'; 6 SELECT * from usuarios where nome like '%Abella'; 7 SELECT * from usuarios where usuario = 'dabella' 8 and nome = 'Daniel Abella'; </pre>
UPDATE	Atualiza linhas	<pre> 1 UPDATE usuarios 2 SET nome = 'Daniel Abella Souza' WHERE id = 1; 3 UPDATE usuarios 4 SET nome = 'Daniel' WHERE id = 1 or usuario='dsouza'; 5 UPDATE usuarios 6 SET nome = 'Dan' and usuario = 'ddan' 7 WHERE id = 1 and usuario='dsouza'; </pre>
DELETE	Exclui as linhas	<pre> 1 DELETE FROM usuarios; 2 DELETE FROM usuarios WHERE id = 1; 3 DELETE FROM usuarios WHERE id = 1 or usuario='dsouza'; 4 DELETE FROM usuarios WHERE id = 1 and usuario='dsouza'; </pre>
SHOW DATABASES		
CREATE DATABASE db1	Cria o DB db1	
USE db1	Usa o db1	