

# **BANCO DE DADOS RELACIONAL**

**Introdução ao Banco de Dados Relacional**

# Tópicos da aula



## ❑ 📌 **Objetivos Gerais:**

- Apresentar os conceitos fundamentais de **Banco de Dados Relacional** e o **PostgreSQL**.
- Demonstrar a importância dos bancos de dados no desenvolvimento de software.
- Explicar a estrutura básica do PostgreSQL e seu uso no projeto da ABP.
- Preparar o ambiente de desenvolvimento para uso ao longo do curso.

## ❑ 📌 **Objetivos Específicos:**

- Entender o que é um banco de dados e seus diferentes tipos.
- Diferenciar bancos de dados relacionais e não relacionais.
- Aprender comandos básicos do PostgreSQL.
- Criar um **banco de dados real** no PostgreSQL e realizar operações básicas.

# O que é um Banco de Dados?

## ❑ **Definição:**

- Um **Banco de Dados** é um conjunto de informações organizadas de maneira estruturada.
- Permite armazenamento, recuperação e manipulação eficiente de dados.
- Usado em sistemas de gestão, aplicações web, dispositivos móveis, entre outros.

# O que é um Banco de Dados?

- ❑ **Exemplos do dia a dia:**
  - **Netflix:** Armazena informações sobre filmes, usuários e preferências.
  - **E-commerce:** Guarda dados de produtos, clientes e transações.
  - **Redes Sociais:** Mantém perfis de usuários, postagens e interações.

# Tipos de Bancos de Dados

- ❑ **Banco de Dados Relacional (SQL):**
  - ❑ Estruturado em **tabelas** e **relacionamentos**.
  - ❑ Usa **SQL (Structured Query Language)** para manipulação de dados.
  - ❑ Exemplo: **PostgreSQL, MySQL, SQL Server**.
  
- ❑ **Banco de Dados Não Relacional (NoSQL):**
  - ❑ Armazena dados em **documentos, grafos ou chave-valor**.
  - ❑ Maior flexibilidade para **grandes volumes de dados não estruturados**.
  - ❑ Exemplo: **MongoDB, Firebase, Redis**.
  
- ❑ **Comparação:**
  - ❑ **PostgreSQL (Relacional)** = Melhor para aplicações estruturadas.
  - ❑ **MongoDB (Não Relacional)** = Melhor para aplicações flexíveis e escaláveis.

# Introdução ao PostgreSQL

- ❑ **O que é PostgreSQL?**
  - Um **Sistema Gerenciador de Banco de Dados Relacional (SGBDR)**.
  - Código aberto, altamente seguro e escalável.
  - Suporte a **SQL e PL/pgSQL** (linguagem procedural).
  - Utilizado em aplicações **web, análise de dados e sistemas empresariais**.
  
- ❑ **Vantagens do PostgreSQL:**
  - **Suporte a JSON** – Permite combinar estrutura relacional com dados flexíveis.
  - **Extensibilidade** – Suporte a tipos de dados personalizados.
  - **Desempenho otimizado** para grandes volumes de dados.

# Estrutura do PostgreSQL

## ❑ Principais componentes:

- 1 **Schemas** – Organização lógica dos objetos no banco.
- 2 **Tabelas** – Estrutura principal para armazenar dados.
- 3 **Índices** – Melhoram a performance de buscas.
- 4 **Visões (Views)** – Formas alternativas de consultar dados.
- 5 **Usuários e Permissões** – Controle de acesso ao banco.

# Instalando o PostgreSQL

## ❑ Passos para instalação:

- ✓ Baixe e instale o PostgreSQL via <https://www.postgresql.org/download/>
- ✓ Configure o pgAdmin para gerenciar visualmente os bancos de dados.
- ✓ Acesse o terminal e teste o comando:

```
SELECT version();
```



# O que é SQL?

- ❑ **Structured Query Language (SQL)**
  - ❑ SQL é a linguagem padrão para manipulação de dados em **bancos de dados relacionais**.
  - ❑ Criada nos anos 1970, evoluiu e se tornou essencial em sistemas modernos.
  - ❑ Utilizada para **criar, consultar, modificar e excluir dados** de um banco relacional.
- ❑ **Por que SQL é importante?**
  - ❑ Universal: Suportado por **PostgreSQL, MySQL, SQL Server, Oracle, etc.**
  - ❑ Simples e eficaz para **manipular grandes volumes de dados**.
  - ❑ Indispensável no desenvolvimento de sistemas e aplicações web.

# Principais Comandos SQL

- ❑ **1. Comandos de Definição de Dados (DDL - Data Definition Language)**
  - CREATE TABLE – Cria novas tabelas.
  - ALTER TABLE – Modifica a estrutura de uma tabela.
  - DROP TABLE – Remove tabelas do banco.
  
- ❑ **2. Comandos de Manipulação de Dados (DML - Data Manipulation Language)**
  - SELECT – Recupera dados do banco.
  - INSERT – Insere novos registros.
  - UPDATE – Atualiza dados existentes.
  - DELETE – Remove registros de uma tabela.
  
- ❑ **3. Comandos de Controle de Transações (TCL - Transaction Control Language)**
  - COMMIT – Salva as alterações realizadas.
  - ROLLBACK – Desfaz transações não confirmadas.

# Estrutura de um Comando SQL

## ❑ Sintaxe básica:

`SELECT` coluna1, coluna2 `FROM` tabela `WHERE` condição;

## ❑ Exemplo prático:

❑ *Recuperar todos os alunos de um banco de dados:*

`SELECT` \* `FROM` alunos;

❑ *Filtrar alunos com idade maior que 18 anos:*

`SELECT` \* `FROM` alunos `WHERE` idade > 18;

# Criando um Banco de Dados

- ❑ Comando para criar um banco de dados:

```
CREATE DATABASE queimadas_db;
```

- ❑ Criando uma tabela:

```
CREATE TABLE focos_calor (  
    id SERIAL PRIMARY KEY,  
    estado VARCHAR(50),  
    bioma VARCHAR(50),  
    data_ocorrencia DATE  
);
```

# Manipulação de Dados (INSERT, SELECT)

## ❑ Inserindo dados:

```
INSERT INTO focos_calor (estado, bioma, data_ocorrendia)  
VALUES ('São Paulo', 'Mata Atlântica', '2025-02-10');
```

## ❑ Consultando os dados:

```
SELECT * FROM focos_calor;
```

# Atividade Prática

## Passos para a atividade:

- 1 Crie um banco de dados chamado *queimadas\_db*.
- 2 Crie a tabela *focos\_calor* com os campos:
- 3 Insira os seguintes registros:

```
INSERT INTO focos_calor (estado, bioma, data_ocorrendia) VALUES  
( 'Amazonas', 'Amazônia', '2025-02-01'),  
( 'Mato Grosso', 'Cerrado', '2025-02-03'),  
( 'Pará', 'Amazônia', '2025-02-05');
```

- 4 Faça uma consulta para exibir os dados inseridos:

# Atividade Prática

- ❑ Criar 5 tabelas quaisquer
- ❑ Criar tabelas conforme MER (Sistema Bancario):

cliente
<u>NOME_CLIENTE</u>
CIDADE_CLIENTE
ENDERECO_CLIENTE




conta
<u>NUMERO_CONTA</u>
NOME_AGENCIA
SALDO

emprestimo
<u>NUMERO_EMPRESTIMO</u>
NOME_AGENCIA
VALOR

agencia
<u>NOME_AGENCIA</u>
CIDADE_AGENCIA
DEPOSITOS

# Entrega do Requisito (BDR.01)

## O que deve ser entregue?




-  Configuração do ambiente PostgreSQL.
-  Criação do banco de dados e tabelas.
-  Inserção de registros iniciais.

## Requisito atendido: Preparação inicial do projeto.

 **Prazo de entrega:** 15/04 - Sprint 1.



# Referências Bibliográfica da Aula

- ❑  **Livros:**
  - **Elmasri & Navathe (2010).** Sistemas de Banco de Dados.
  - **Silberschatz et al. (2011).** Sistemas de Banco de Dados.
- ❑  **Links úteis:**
  -  [PostgreSQL Documentation](#)

# Bibliografia Básica

- ❑ DATE, C. J. **Introdução a sistemas de bancos de dados**. Rio de Janeiro, Elsevier: Campus, 2004.
- ❑ ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 7 ed. São Paulo: Pearson, 2018.
- ❑ SILBERSCHATZ, A.; SUNDARSHAN, S.; KORTH, H. F. **Sistema de banco de dados**. Rio de Janeiro: Elsevier Brasil, 2016.

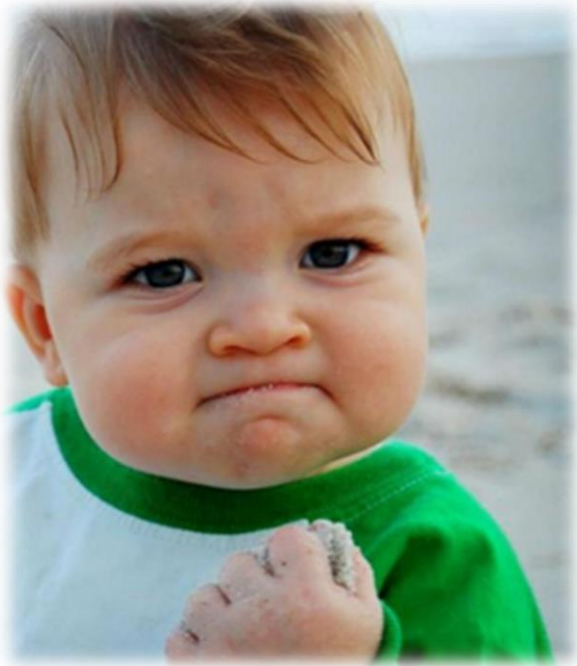
# Bibliografia Complementar

- ❑ BEAULIEU, A. **Aprendendo SQL**. São Paulo: Novatec, 2010.
- ❑ GILLENSON, M. L. **Fundamentos de Sistemas de Gerência de Banco de Dados**. Rio de Janeiro: LTC, 2006.
- ❑ MACHADO, F. N. R. **Banco de Dados: Projeto e Implementação**. São Paulo: Érica, 2005.
- ❑ OTEY, M; OTEY, D. **Microsoft SQL Server 2005: Guia do Desenvolvedor**. Rio de Janeiro: Ciência Moderna, 2007.
- ❑ RAMAKRISHNAN, R.; GEHRKE, J. **Sistemas de Gerenciamento de Bancos de Dados**. 3 ed. Porto Alegre: Bookman, 2008.
- ❑ ROB, P; CORONEL, C. **Sistemas de Banco de Dados: Projeto, Implementação e Gerenciamento**. 8 ed. São Paulo: Cengage Learning, 2011.
- ❑ TEOREY, T; LIGHTSTONE, S; NADEAU, T. **Projeto e Modelagem de Bancos de Dados**. São Paulo: Campus, 2006.

# Dúvidas?



# Considerações Finais



**Professor(a):  
Lucineide Pimenta**

**Bom semestre à todos!**

