

ATIVIDADE DA AULA 2 - Banco de Dados - Relacional (2º Semestre)

ALUNA: Nadia Fernandes Ferreira
RA: 2581392613046

CÓDIGO:

```
CREATE DATABASE queimadas_db;
```

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

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

```
SELECT * FROM focos_calor;
```

PROCESSO PARA CRIAÇÃO DO BANCO DE DADOS queimadas_db:

Esta é a primeira atividade da disciplina de Banco de Dados Relacional (2º semestre). A proposta consiste na execução de um código SQL no pgAdmin, com o objetivo de criar um banco de dados denominado queimadas_db e, a partir dele, gerar uma tabela. A seguir, apresentarei o passo a passo do processo, bem como o resultado obtido por mim e o resultado esperado.

Bom, com o PostgreSQL devidamente instalado no computador e seguindo as instruções apresentadas em aula, acessei o pgAdmin e criei um novo servidor.

Em seguida, dentro do database padrão "postgres", abri o Query Tool e executei a primeira parte do código SQL, responsável pela criação do banco de dados chamado "queimadas_db".

Esse procedimento foi realizado separadamente, pois no PostgreSQL o comando CREATE DATABASE não pode ser executado dentro de um bloco de transação.

The screenshot shows the pgAdmin Query Tool interface. The title bar says 'Dependencies X Dependents X Processes X queimadas_db/postgres X postgres/postgres@PostgreSQL Local*'. The toolbar has various icons, with the 'Execute query' icon highlighted by a red arrow. The main area shows a single query: 'CREATE DATABASE queimadas_db;'. Below the query, a red box highlights the text 'Criação do database queimadas_db'. In the bottom right corner of the main window, another red box highlights the message 'Banco já havia sido criado (Bom sinal!)'. The status bar at the bottom shows 'Data Output Messages Notifications'.

```
Dependencies X Dependents X Processes X queimadas_db/postgres X postgres/postgres@PostgreSQL Local*
postgres/postgres@PostgreSQL Local
No limit
Query History
1 CREATE DATABASE queimadas_db;
Criação do database queimadas_db
Data Output Messages Notifications
ERROR: database "queimadas_db" already exists
ERROR: database "queimadas_db" already exists
SQL state: 42P04
Banco já havia sido criado (Bom sinal!)
```

Porém, quando executei o comando CREATE DATABASE queimadas_db;, apareceu a mensagem:

ERROR: database "queimadas_db" already exists

Isso aconteceu porque o banco já tinha sido criado anteriormente por mim. Mesmo assim, foi importante testar o comando para entender como o PostgreSQL funciona e confirmar que o banco estava criado corretamente.

Seguindo o passo a passo, acessei o database queimadas_db e comecei a escrever cada parte do código SQL responsável pela criação da tabela.

Foram utilizados comandos como CREATE TABLE, INSERT INTO e VALUES, além das variáveis/dados: estado, bioma e data_ocorrencia.

Cada parte do código foi executada separadamente, uma por vez, com o objetivo de garantir o funcionamento correto e obter o resultado esperado.

The screenshot shows a pgAdmin interface with a query editor window. The code consists of three distinct parts:

- Line 1: `CREATE TABLE focos_calor(` (1º Parte executada)
- Line 9: `INSERT INTO focos_calor(estado, bioma, data_ocorrencia) VALUES` (2º Parte executada)
- Line 13: `SELECT * FROM focos_calor;` (3º Parte executada)

A red arrow points from the top of the interface to the title bar, with the text "Cada um é executado um query". Another red arrow points to the database name in the title bar, with the text "Essa parte é executada no database queimadas_db".

```
1 CREATE TABLE focos_calor(
2     id SERIAL PRIMARY KEY,
3     estado VARCHAR(50),
4     bioma VARCHAR(50),
5     data_ocorrencia DATE
6 );
7
8     INSERT INTO focos_calor(estado, bioma, data_ocorrencia) VALUES
9         ('Amazonas', 'Amazonia', '2025-02-01'),
10        ('Mato Grosso', 'Cerrado', '2025-02-03'),
11        ('Pará', 'Amazônia', '2025-02-05');
12
13     SELECT * FROM focos_calor;
```

Vale ressaltar que também foi utilizado o comando id SERIAL PRIMARY KEY.

Ele permite que seja criada uma identificação automática para cada registro inserido na tabela, tornando os dados mais organizados e facilitando sua identificação.

Por fim, a imagem abaixo mostra a tabela resultante, que corresponde ao resultado esperado após a execução do código.

The screenshot shows the pgAdmin Data Output tab displaying the results of the query. The table has four columns: id, estado, bioma, and data_ocorrencia. The data is as follows:

	<code>id [PK] integer</code>	<code>estado character varying (50)</code>	<code>bioma character varying (50)</code>	<code>data_ocorrencia date</code>
1	1	Amazonas	Amazonia	2025-02-01
2	2	Mato Grosso	Cerrado	2025-02-03
3	3	Pará	Amazônia	2025-02-05

A red arrow points to the table data with the text "Aqui temos a tabela obtida com os dados!"

Esse foi o processo da atividade 1, realizada por mim. Obrigada pela atenção!