Pedro Sergio Gomes Quinderé

Banco de dados utilizado foi o **MySQL WorkBench.**

Passo 1: Criar o Banco de Dados

Antes de criarmos as tabelas, precisamos de um **banco de dados** para armazená-las. No PostgreSQL, podemos criar um banco de dados com o seguinte comando:

```
-- Criando um novo banco de dados chamado 'empresa_db'
CREATE DATABASE empresa_db;
```

📌 **Explicação:**
✅ O comando CREATE DATABASE cria um novo **banco de dados**.
✅ O nome do banco de dados será **empresa_db**.
✅ Depois de criar, é necessário **selecioná-lo no pgAdmin** antes de criar as tabelas.

Passo 2: Criar a Tabela "departamento"

Agora vamos criar a primeira tabela, chamada **departamento**, que armazenará informações sobre os departamentos da empresa.

```
-- Criando a tabela 'departamento'

CREATE TABLE departamento (

    id_departamento SERIAL PRIMARY KEY, -- Criamos um identificador único para cada departamento

    nome VARCHAR(100) NOT NULL -- Nome do departamento, obrigatório

);
```

📌 **Explicação:**
✅ id_departamento SERIAL PRIMARY KEY:
- SERIAL cria um **número automático** para cada novo departamento.
- PRIMARY KEY significa que essa coluna é a **identificação única** da tabela.

✅ nome VARCHAR(100) NOT NULL:
- VARCHAR(100): Aceita até **100 caracteres** no nome do departamento.
- NOT NULL: O nome **não pode ficar em branco** (obrigatório).

Passo 3: Criar a Tabela "funcionario"

Agora vamos criar a tabela **funcionario**, que armazenará os dados dos funcionários da empresa.

```
-- Criando a tabela 'funcionario'

CREATE TABLE funcionario (

    id_funcionario SERIAL PRIMARY KEY, -- Identificador único para cada funcionário

    nome VARCHAR(150) NOT NULL, -- Nome do funcionário, obrigatório

    cargo VARCHAR(100), -- Cargo do funcionário (opcional)

    salario DECIMAL(10,2) CHECK (salario >= 0), -- Salário do funcionário, não pode ser negativo

    id_departamento INT, -- Chave estrangeira que liga o funcionário ao departamento

    CONSTRAINT fk_departamento FOREIGN KEY (id_departamento) REFERENCES departamento(id_departamento) ON DELETE SET NULL

);
```

📌 **Explicação:**

✅ id_funcionario SERIAL PRIMARY KEY:

- **Cada funcionário** recebe um número único automaticamente.

✅ salario DECIMAL(10,2) CHECK (salario >= 0):

- DECIMAL(10,2): O salário pode ter até **10 dígitos** no total, sendo **2 casas decimais** (exemplo: 4500.50).
- CHECK (salario >= 0): O salário **não pode ser negativo**.

✅ id_departamento INT:

- O campo id_departamento representa **qual departamento** o funcionário pertence.

✅ FOREIGN KEY (id_departamento) REFERENCES departamento(id_departamento):

- Isso cria um **vínculo entre as tabelas**.
- Um funcionário **deve estar associado** a um departamento.

✅ ON DELETE SET NULL:

- Se um departamento for excluído, o campo id_departamento do funcionário **fica vazio (NULL)**, mas o funcionário **não é apagado**.

Passo 4: Criar a Tabela "projeto"

Agora vamos criar a tabela **projeto**, que armazenará informações sobre projetos da empresa.

```
-- Criando a tabela 'projeto'

CREATE TABLE projeto (

    id_projeto SERIAL PRIMARY KEY, -- Identificador único para cada projeto

    nome VARCHAR(150) NOT NULL, -- Nome do projeto, obrigatório

    descricao TEXT, -- Descrição detalhada do projeto (opcional)

    id_departamento INT, -- Qual departamento é responsável pelo projeto

    CONSTRAINT fk_projeto_departamento FOREIGN KEY (id_departamento) REFERENCES departamento(id_departamento) ON DELETE CASCADE

);
```

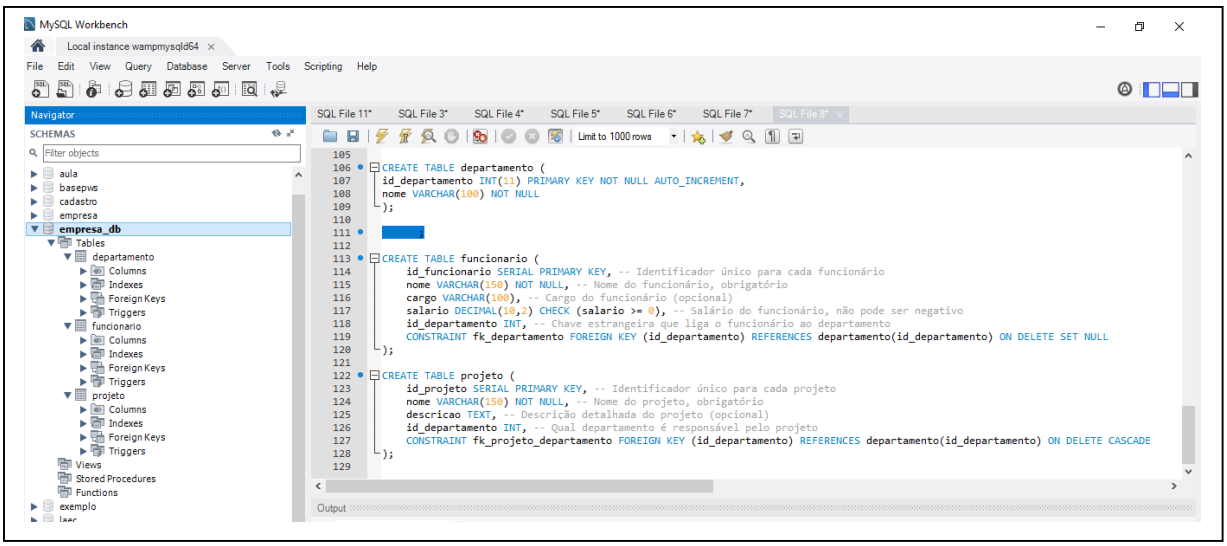
📌 **Explicação:**
✅ descricao TEXT:

- O tipo TEXT permite armazenar **textos longos**, úteis para descrever os projetos.

✅ FOREIGN KEY (id_departamento) REFERENCES departamento(id_departamento) ON DELETE CASCADE:

- Se o departamento for excluído, **todos os projetos dele serão apagados automaticamente**.

Banco de dados e tabelas criadas:



Passo 5: Inserindo Dados nas Tabelas

Agora que criamos as tabelas, vamos adicionar **alguns dados** para testar.

```
-- Inserindo departamentos

INSERT INTO departamento (nome) VALUES ('Recursos Humanos');

INSERT INTO departamento (nome) VALUES ('TI');

INSERT INTO departamento (nome) VALUES ('Marketing');
```

📌 **Explicação:**

- INSERT INTO adiciona novos registros na tabela.
- Estamos criando **três departamentos**: Recursos Humanos, TI e Marketing.

Agora, vamos inserir alguns funcionários:

```
-- Inserindo funcionários

INSERT INTO funcionario (nome, cargo, salario, id_departamento) VALUES ('Ana Souza', 'Analista', 4500.00, 1);

INSERT INTO funcionario (nome, cargo, salario, id_departamento) VALUES ('Carlos Silva', 'Desenvolvedor', 7000.00, 2);
```

```
INSERT INTO funcionario (nome, cargo, salario, id_departamento) VALUES ('Mariana Costa', 'Designer', 4000.00, 3);
```

📌 **Explicação:**

- VALUES (...) insere os dados na ordem das colunas.
- id_departamento = 1 significa que **Ana trabalha no departamento 1 (Recursos Humanos)**.

Agora, vamos inserir alguns projetos:

```
-- Inserindo projetos

INSERT INTO projeto (nome, descricao, id_departamento) VALUES ('Website Corporativo', 'Criação do site institucional', 2);

INSERT INTO projeto (nome, descricao, id_departamento) VALUES ('Campanha Publicitária', 'Marketing digital nas redes sociais', 3);
```

📌 **Explicação:**

- Cada projeto pertence a um departamento **(id_departamento)**.

Passo 6: Consultas para Testar

Agora, podemos **consultar os dados inseridos**.

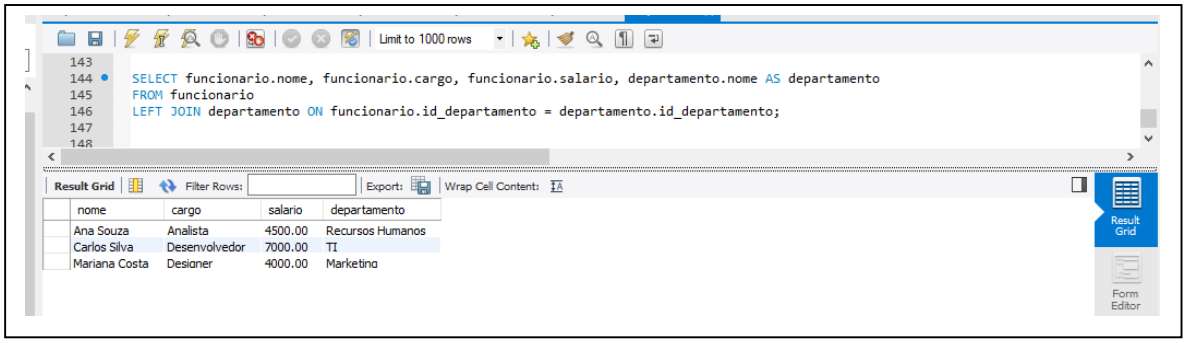
```
-- Exibir todos os funcionários e seus departamentos

SELECT funcionario.nome, funcionario.cargo, funcionario.salario, departamento.nome AS departamento

FROM funcionario

LEFT JOIN departamento ON funcionario.id_departamento = departamento.id_departamento;
```

Primeira consulta efetuada:



Explicação:

- SELECT seleciona **quais informações** queremos ver.
- LEFT JOIN combina a tabela **funcionario** com a tabela **departamento**, mostrando o nome do **departamento** junto.

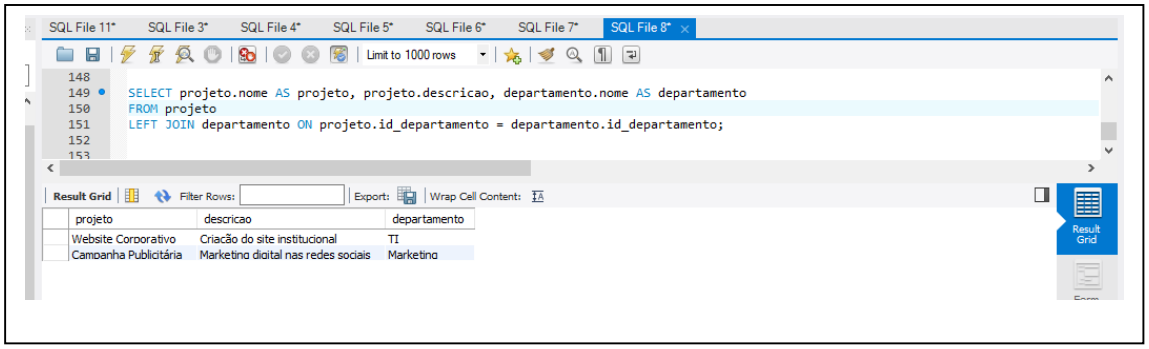
```
-- Exibir todos os projetos e seus departamentos

SELECT projeto.nome AS projeto, projeto.descricao, departamento.nome AS departamento

FROM projeto

LEFT JOIN departamento ON projeto.id_departamento = departamento.id_departamento;
```

Segunda consulta efetuada:



📌 **Explicação:**

- Aqui estamos listando os **projetos**, suas **descrições** e qual **departamento** é responsável por cada um.

Resumo

✅ Criamos um **banco de dados**.
✅ Criamos três **tabelas** (departamento, funcionario, projeto).
✅ Implementamos **chaves primárias e estrangeiras**.
✅ Inserimos **dados de exemplo**.
✅ Realizamos **consultas para visualizar as informações**.