UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
–
UTFPR

Projeto Final - text-to-sql

Introdução a Banco de Dados - ICSB30, S01

Gabriel de França Picinato - 2556464
Gustavo Chemin Ribeiro - 2556480

Curitiba, 2025

1 Introdução

Este relatório descreve o projeto final da disciplina de Introdução a Banco de Dados, que consiste em uma aplicação capaz de converter perguntas em linguagem natural para consultas SQL executáveis. A ferramenta utiliza uma interface gráfica desenvolvida em Python e se integra com a API generativa do Google (Gemini) para realizar a tradução da intenção do usuário em uma consulta formal ao banco de dados. O objetivo deste documento é fornecer um guia detalhado sobre como configurar o ambiente de desenvolvimento, instalar as dependências necessárias e executar o programa. O repositório no github pode ser acessado aqui.

2 Pré-requisitos

Antes de iniciar a instalação, é necessário garantir que os seguintes componentes estejam disponíveis:

- Acesso a um terminal (Linux/macOS) ou PowerShell/CMD (Windows).
- Uma **chave de API do Google** válida para uso com o Gemini.
- Um servidor de banco de dados **PostgreSQL** ou **MySQL** instalado, configurado e em execução.

3 Configuração do Ambiente e Instalação

Esta seção detalha os passos para preparar o ambiente de execução da aplicação.

3.1 Passo 1: Instalação do Python 3.10+

O projeto requer a versão 3.10 ou superior do Python. Se esta versão já estiver instalada em seu sistema, prossiga para o passo seguinte.

Caso Específico: Ubuntu 20.04 (ou similar)

Para evitar conflitos com a versão padrão do sistema, recomenda-se instalar o Python 3.10 de forma segura usando o repositório `deadsnakes`. Execute os seguintes comandos no terminal:

```
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository ppa:deadsnakes/ppa
sudo apt install python3.10 python3.10-venv
```

3.2 Passo 2: Ambiente Virtual e Dependências

O uso de um ambiente virtual é crucial para isolar as bibliotecas do projeto.

1. **Crie e ative o ambiente virtual** na pasta raiz do projeto:

```
# Criar o ambiente com a versão correta do Python
python3.10 -m venv venv

# Ativar o ambiente
source venv/bin/activate
```

2. **O arquivo** `requirements.txt` presente na raiz do projeto contém o conteúdo abaixo. Este arquivo lista todas as bibliotecas necessárias.

Conteúdo para o arquivo `requirements.txt`

```
customtkinter==5.2.2
pandas==2.3.0
psycopg2-binary==2.9.10
PyMySQL==1.1.1
google-generativeai==0.8.5
cryptography==45.0.4
python-dotenv==1.1.0
requests>=2.32.0
certifi>=2025.6.15
Pillow
```

3. **Instale as dependências** a partir do arquivo criado:

```
pip install -r requirements.txt
```

4 Configuração do Projeto

4.1 Chave de API

Para que a aplicação possa se comunicar com o serviço do Gemini, a chave de API deve ser configurada corretamente.

1. Na pasta raiz do projeto, crie um arquivo chamado exatamente `.env`.
2. Dentro deste arquivo, adicione a linha a seguir, substituindo `SUA_CHAVE_SECRETA_AQUI` pela sua chave de API real.

Conteúdo para o arquivo `.env`

```
GOOGLE_API_KEY="SUA_CHAVE_SECRETA_AQUI"
```

5 Execução do Programa

Com o ambiente virtual ativado e as configurações finalizadas, a aplicação pode ser iniciada.

1. Certifique-se de que o ambiente virtual está ativo (o prefixo `(venv)` deve aparecer no terminal).

2. Execute o script principal da interface gráfica:

```
python3 gui.py
```

   Após a execução do comando, a interface gráfica do projeto será exibida, pronta para uso.

3. O programa também pode ser executado via terminal, sem o uso da interface gráfica, a partir do seguinte comando:

```
python3 script.py
```

6 A interface

A Figura 1 apresenta a configuração inicial do programa, onde o usuário escolhe qual SGBD irá usar (MySQL ou PostgreSQL), inserindo um usuário e senha.

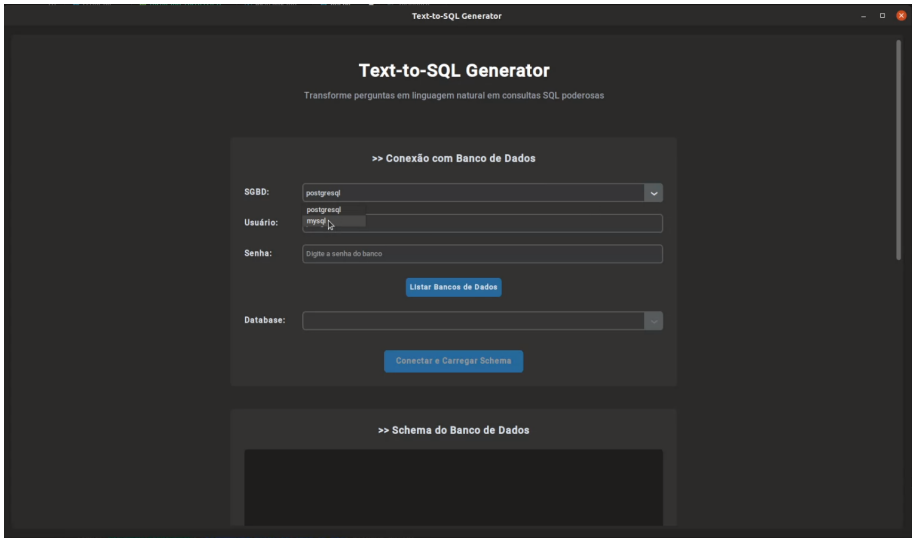

Figura 1. Login no banco de dados

Ao confirmar seu usuário e senha, é disponibilizada uma lista dos bancos de dados disponíveis para aquele usuário, como visto na Figura 2.

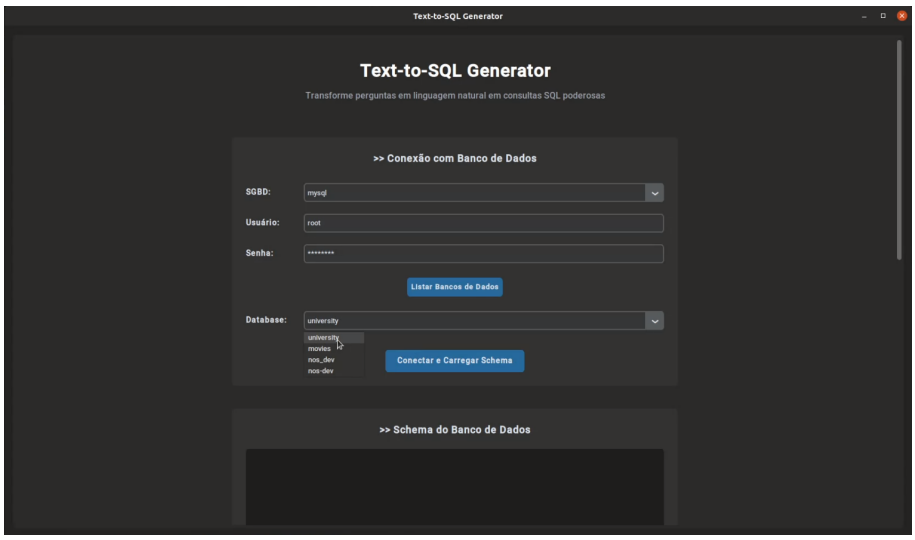

Figura 2. Lista de bancos disponíveis

Ao escolher um banco de dados (e.g. *university*), o *schema* é carregado tanto para o usuário como para o Gemini, permitindo que o usuário faça sua consulta em linguagem natural, como no exemplo da Figura 3.

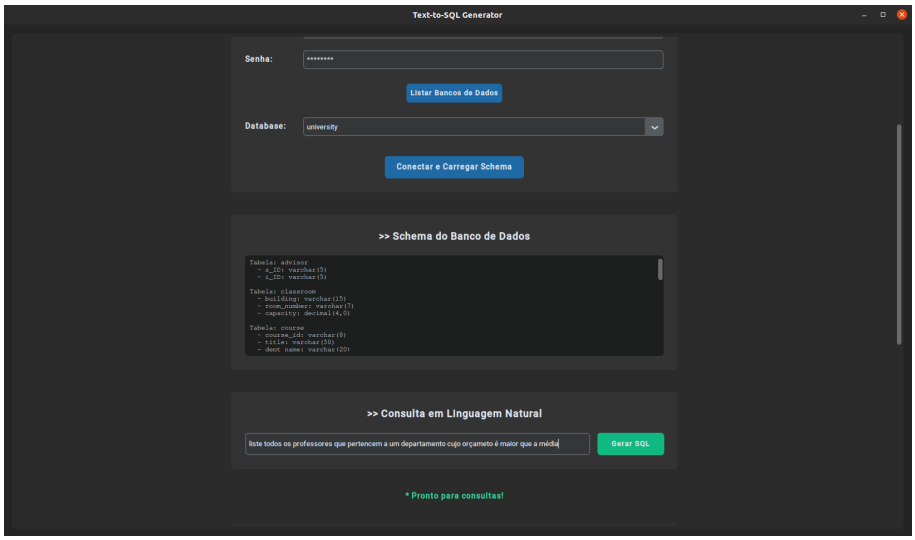

Figura 3. Consulta em linguagem natural

Ao pressionar em "Gerar SQL", o modelo processa o texto em linguagem natural retornando uma consulta SQL válida e mostrando seu resultado (Figura 4).

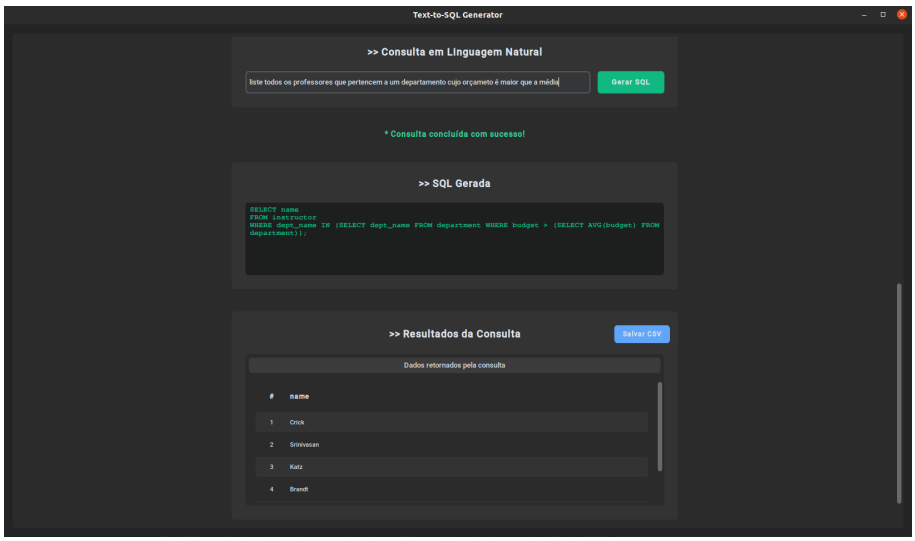

Figura 4. Query SQL e resultado da consulta

Por fim, a interface dá a opção de salvar um arquivo .csv do resultado da consulta, permitindo abrir o resultado em uma planilha para trabalhar com os dados posteriormente, como é possível ver nas Figuras 5 e 6.

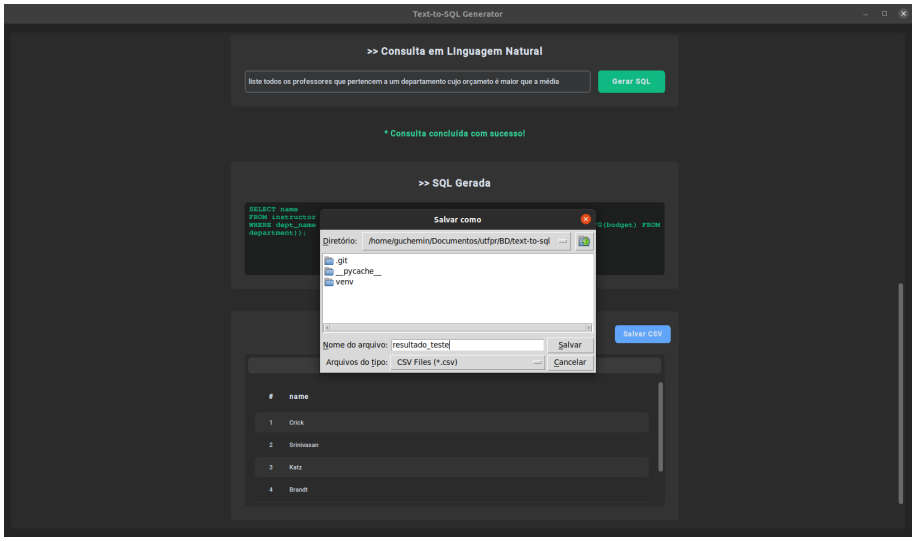

Figura 5. Salvando resultado em .csv

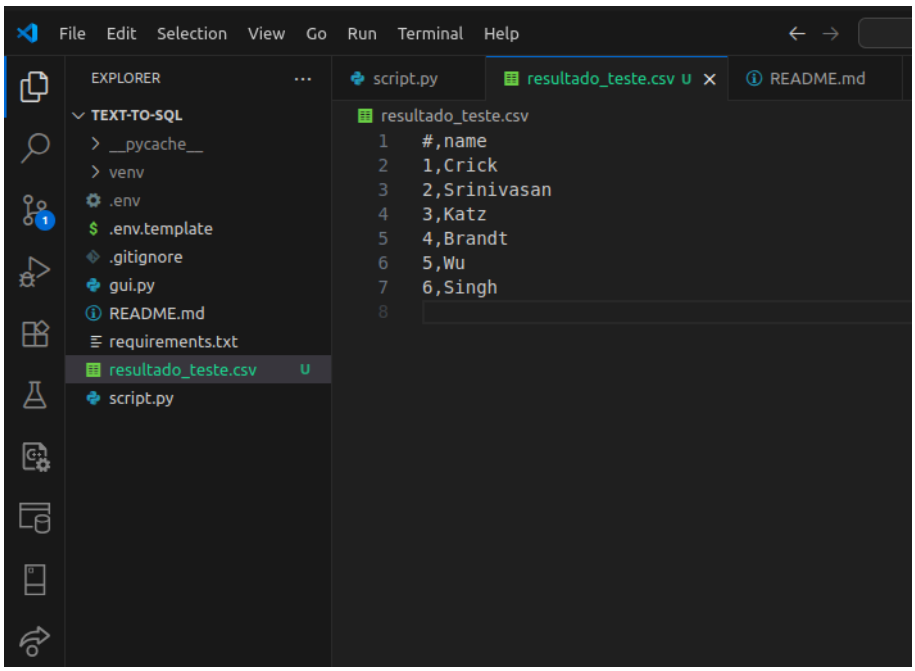

Figura 6. Arquivo .csv salvo

Para melhor visualização do funcionamento do programa, foi gravado um vídeo demonstrando as funcionalidades, que pode ser conferido pelo seguinte link: vídeo.

7 Conclusão

O projeto do banco de dados apresentado neste relatório cumpre os objetivos propostos ao criar uma ponte funcional entre a linguagem natural do usuário e a linguagem SQL do banco de dados. Seguindo os passos de configuração e execução detalhados, é possível replicar o ambiente e operar a aplicação com sucesso, demonstrando a viabilidade da integração entre interfaces de usuário, APIs de inteligência artificial e sistemas de gerenciamento de bancos de dados para tornar o acesso à informação mais intuitivo e acessível.