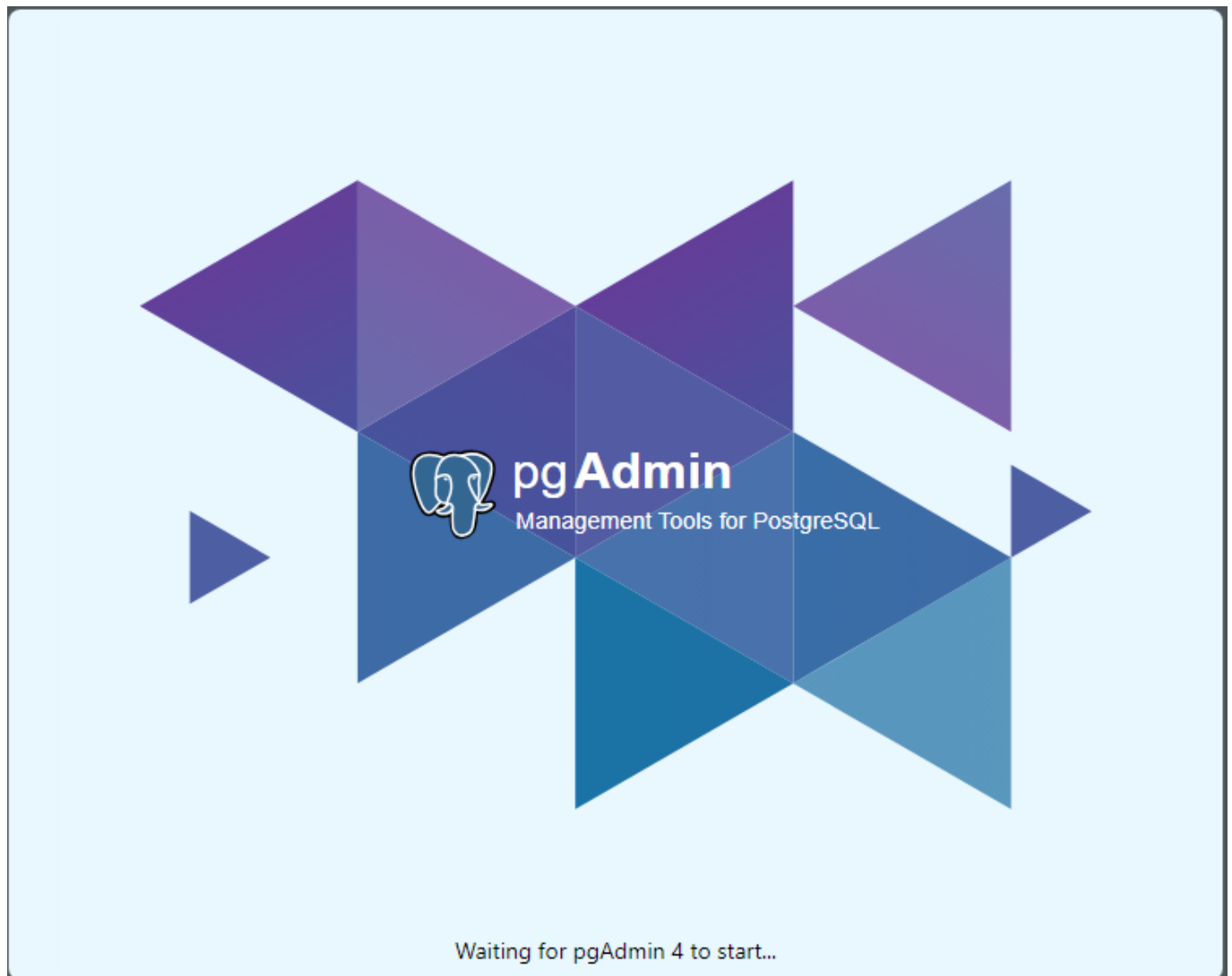


Introdução ao pgAdmin

Dando continuidade à série [Instalando o PostgreSQL com Docker no WSL2](#), chegamos à **Parte 3**, onde vamos explorar o **pgAdmin**, a principal ferramenta gráfica de administração do **PostgreSQL**. O **pgAdmin** é um projeto gratuito e de código aberto, desenvolvido e mantido pela comunidade **PostgreSQL** sob a coordenação da **pgAdmin Development Team**. Ele está em constante evolução há mais de 20 anos (a primeira versão foi lançada em 1998) e atualmente sua versão mais moderna é o **pgAdmin 4**, que substituiu as versões anteriores para oferecer uma interface web mais flexível e compatível com diferentes plataformas.

Essa ferramenta se tornou o padrão oficial para o **PostgreSQL**, adotado tanto por iniciantes quanto por grandes empresas. Seu objetivo é facilitar a interação com o banco de dados através de uma interface gráfica intuitiva, permitindo que você realize desde tarefas simples, como criar tabelas, até atividades complexas, como gerenciar permissões, realizar backups, monitorar desempenho e analisar consultas.



Principais destaques do pgAdmin

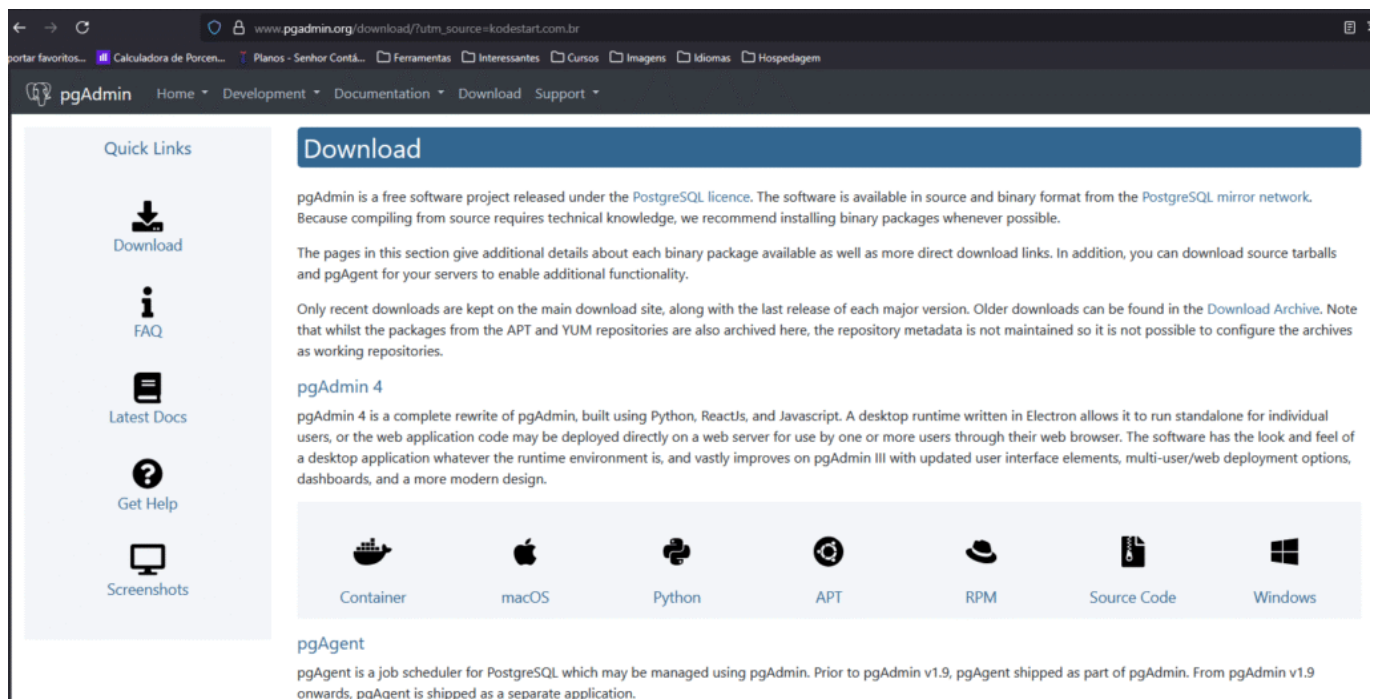
- **Custo:** Totalmente gratuito e open source;
- **Manutenção:** Desenvolvido pela equipe oficial do **PostgreSQL**, com apoio ativo da comunidade global;
- **História:** Projeto consolidado, com mais de duas décadas de evolução contínua;
- **Multiplataforma:** Disponível em Windows, Linux, macOS e também como aplicação web;
- **Funcionalidades:** Editor SQL com autocompletar, gerenciamento de usuários e permissões, comparador de esquemas (Schema Diff), geração de diagramas ERD, ferramentas de backup/restore e muito mais.

Com isso, o **pgAdmin** se apresenta como a porta de entrada perfeita para que deseja aprender

PostgreSQL sem depender apenas do terminal, ao mesmo tempo em que atende usuários avançados com recursos completos de administração.

Instalando o pgAdmin

Agora que você já conhece o **pgAdmin** e entende por que ele é tão importante, vamos colocar a mão na massa! Agora, você vai aprender a instalar o **pgAdmin** no Windows e conectá-lo ao PostgreSQL que configuramos no **Docker** dentro do **WSL2**. Assim, além de ter o banco rodando no ambiente isolado do Docker, você também terá uma interface gráfica prática, organizada e intuitiva para explorar suas tabelas, rodar consultas e administrar tudo com muito mais facilidade.



Para começar, acesse o [site oficial do pgAdmin](https://www.pgadmin.org/download/) e faça o download da versão mais recente para Windows. Após concluir a instalação, abra o programa e vamos direto ao que interessa: registrar o servidor **PostgreSQL** que está rodando no **Docker** dentro do **WSL2**. No **pgAdmin**. Se for uma instalação recente do **pgAdmin** (sem grupos configurados), crie um grupo padrão, clique em menu **Object > Create > Server Group..** e nomeie de **Servers**. Se o **pgAdmin** já estava instalado e o grupo **Servers** já existe, pule essa etapa e clique direto em **Object > Register - Server...**

Register - Server

General

Connection

Parameters

SSH Tunnel

Advanced

Name

PostgreSQL-WSL2

Server group

Servers

Background

Foreground

Connect now?

Comments

Either Host name or Service must be specified.

Close

Reset

Save

Na tela que aparecer *Register -Server*, na aba *General*, no campo *Name*, digite um indetificador claro, por exemplo PostgreSQL-WSL2. Confirme que o *Server Group* está como *Servers* apenas por questões de alinhamento e acompanhamento.

Na aba *Connection*. No campo *Host name/address*, informe o IP do seu **WSL2**. Para descobrir esse IP, abra o **Ubuntu WSL2** no Windows, digite e execute o comando abaixo:

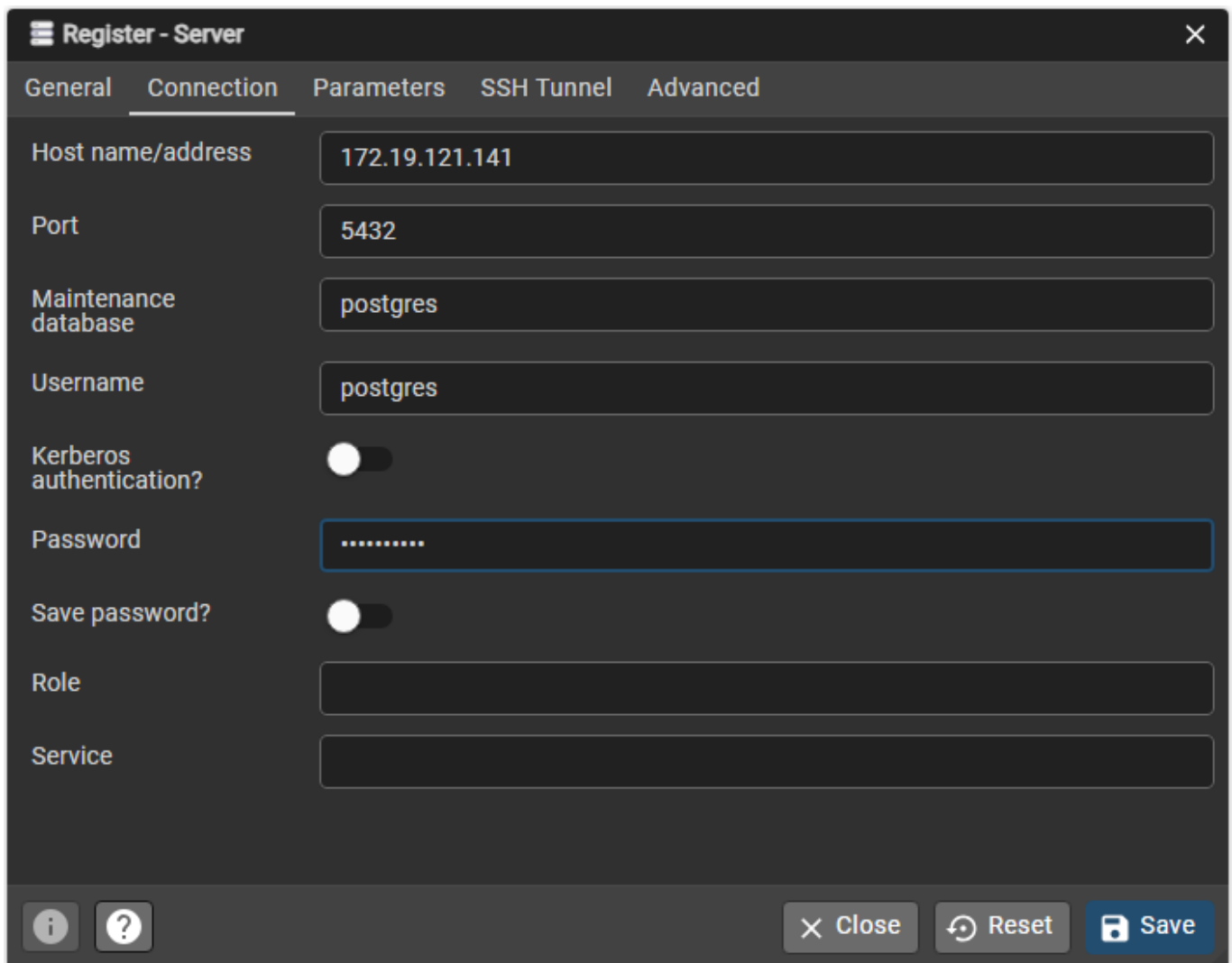


```
hostname -I
```

O comando exibirá um ou mais endereços e você deve usar o primeiro IP da lista (ele costuma coincidir com a linha “IPv4 address for eth0” mostrada no cabeçalho quando o Ubuntu abre).

⚠Atenção: Esse IP pode mudar quando o WSL2 é reiniciado, ou seja, se um dia não conectar, rode o comando novamente e atualize no pgAdmin.

Em *Port*, use 5432 (ou a porta publicada no seu docker - compose), em *Maintenance database*, informe postgres, e preencha *Username* e *Password* com a que definimos na [Parte 2](#) do tutorial anterior. Por fim, clique em *Save* para concluir.



Register - Server

General

Connection

Parameters

SSH Tunnel

Advanced

Host name/address

172.19.121.141

Port

5432

Maintenance database

postgres

Username

postgres

Kerberos authentication?

☐

Password

.....

Save password?

☐

Role

Service

i

?

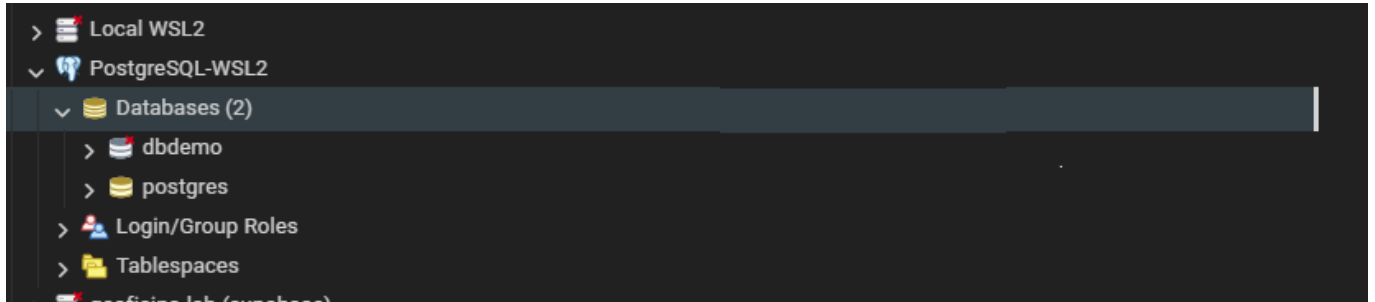
Close

Reset

Save

Após salvar e conectar ao servidor, você verá no painel lateral esquerdo a estrutura do PostgreSQL

organizada por categorias. Ao expandir a opção *Databases*, verá uma lista com os bancos de dados disponíveis naquele servidor. No seu caso, dois bancos deve aparecer, o `postgres` e `dbdemo`.



O banco `postgres` é criado automaticamente pelo **PostgreSQL** e serve como banco administrativo padrão. Já o `dbdemo` foi criado por nós no tutorial anterior ([Parte 2](#)), por meio do script `dbdemo.sql` executado automaticamente quando o container foi iniciado pelo **Docker**. Esse banco foi configurado com **locale pt_BR.UTF-8** e preparado para receber dados com acentuação, ordenações e padrões compatíveis com o português do Brasil.

Você pode clicar sobre `dbdemo`, expandir suas opções e visualizar seções como *Schemas*, *Tables*, *Functions*, entre outras. À medida que avançarmos, é nele que vamos criar tabelas, inserir dados e praticar consultas. Essa confirmação visual mostra que a criação do banco funcionou perfeitamente e que o **pgAdmin** está conectado ao **PostgreSQL** com sucesso.

Conclusão

Se você chegou até aqui, parabéns! Agora você tem o **pgAdmin** instalado, configurado e conectado ao seu servidor **PostgreSQL** rodando dentro do Docker no [WSL2](#). Também confirmou visualmente que o banco `dbdemo`, criado automaticamente no container, está disponível e pronto para ser utilizado. Com isso, finalizamos a configuração da interface gráfica, o que torna seu ambiente de aprendizado muito mais acessível e produtivo.

Na **Parte 4**, vamos dar o próximo passo que é criar as primeiras tabelas dentro do banco `dbdemo`, entender como funciona a estrutura de dados no **PostgreSQL** e praticar comandos SQL de forma visual e prática usando o **pgAdmin**.