# PostgreSQL e PGAdmin 4 em containers Docker

## **O QUE É ESTE TUTORIAL?**

Neste documento você terá o passo a passo para montar e configurar o nosso banco de dados PostgreSQL e pgAdmin 4 em containers Docker.

A função deste material é deixar um ambiente com banco de dados, PostgreSQL e pgAdmin 4, pronto para você utilizar nas dicas do canal xxxx que dará sobre Banco de Dados e linguagem SQL.

Esse material não tem a função de explicar o que é e como funciona o Docker, somente o necessário para o funcionamento deste ambiente.

## **HÁ ALGUM PRÉ-REQUISITO PARA ESTE TUTORIAL FUNCIONAR?**

Sim, pois você já deve o Docker instalado em sua máquina.

Caso você ainda não tenha, veja como instalar no tutorial “Como instalar o docker na minha máquina!” <criar tutorial e inserir o link>

## **VAMOS AO PASSO A PASSO!**

A montagem desse ambiente é rápida e simples!

Serão criados 2 (dois) containers, um com o banco de dados **PostgreSQL** e outro com **pgAdmin 4**, e uma network para a comunicação entre os containers.

Esse ambiente será criado utilizando-se o **Docker Compose**, um serviço do Docker para a criação e execução conjunta dos múltiplos containers de uma solução. Para isso vamos utilizar um arquivo chamado **docker-compose.yml** com os 3 (três) comandos para a criação e configuração dos containers e a network.

### **PASSO 1 – Baixar o arquivo docker-compose.yml**

Baixe o arquivo **docker-compose.yml** para a sua máquina. Mova o arquivo baixado para o seu diretório de trabalho.

Apenas como sugestão, crie seu diretório de trabalho com o nome **curso\_bd\_sql**.

Uma breve explicação do que tem no arquivo **docker-compose.yml**:

version: '3'

services:

  srv-bd-postgresql:

    image: postgres

    environment:

      POSTGRES\_PASSWORD: "curso"

    ports:

      - "15432:5432"

    volumes:

      - /home/luisjesus/Dev/Docker-Compose/PostgreSQL:/var/lib/postgresql/data

    networks:

      - srv-postgres-network

  srv-pgadmin:

    image: dpage/pgadmin4

    environment:

      PGADMIN\_DEFAULT\_EMAIL: "luisjesus.ti@gmail.com"

      PGADMIN\_DEFAULT\_PASSWORD: "curso"

    ports:

      - "16543:80"

    depends\_on:

      - srv-bd-postgresql

    networks:

      - srv-postgres-network

networks:

  srv-postgres-network:

    driver: bridge

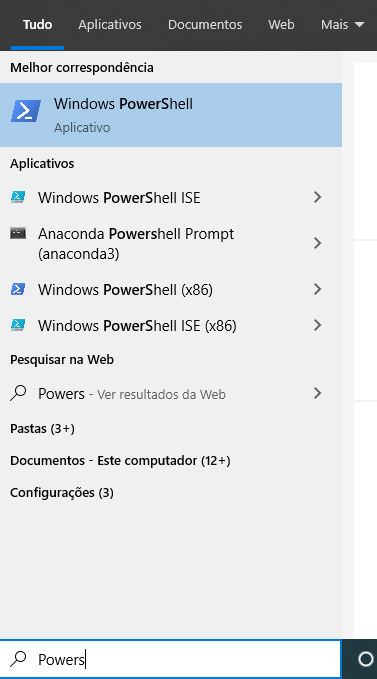
Esse arquivo docker-compose.yml cria:

1. Serviços (Containeres):
   * **srv-bd-postgresql**: com o banco de dados **PostgreSQL** com porta de acesso 15432; e
   * **srv-pgadmin**: com o **PGAdmin 4** com porta de acesso 16543;
2. Rede (Network):
   * **srv-postgres-network**: serviço para comunicação dos containers **srv-bd-postgresql** e **srv-pgadmin**

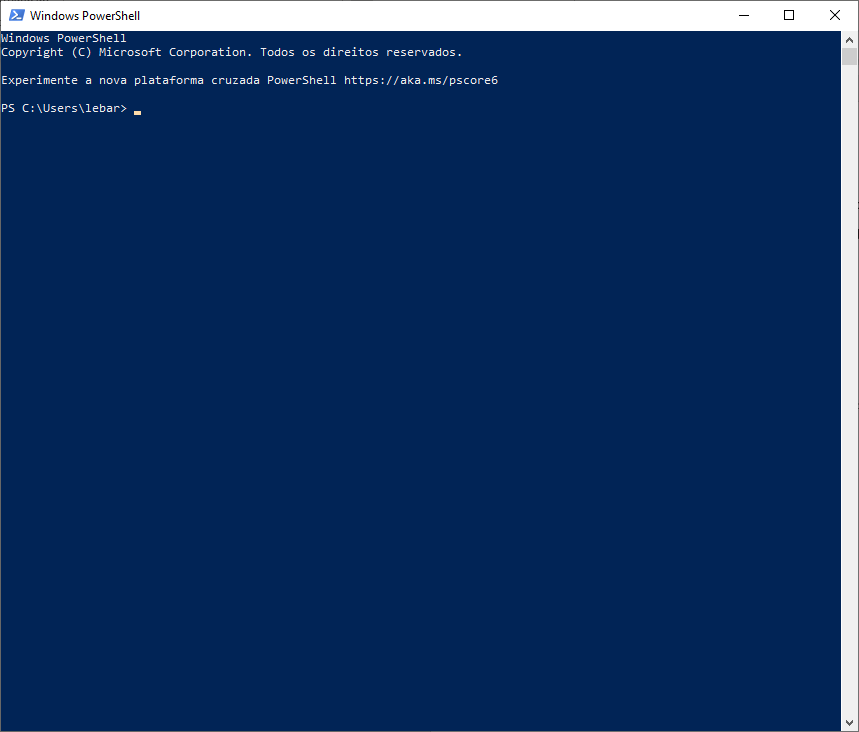
### **PASSO 2 – Colocando para funcionar**

Agora chegou a hora de colocar para funcionar!

Nós vamos trabalhar nesta etapa com o aplicativo Windows PowerShell. Para isso vamos abrir o aplicativo **Windows PowerShell**.



Veja como é o Windows PowerShell na figura abaixo.



Vamos ao diretório onde está o arquivo docker-compose.yml.

No meu caso é o diretório **C:\Users\lebar\Documents\curso-sgbd-sql**.

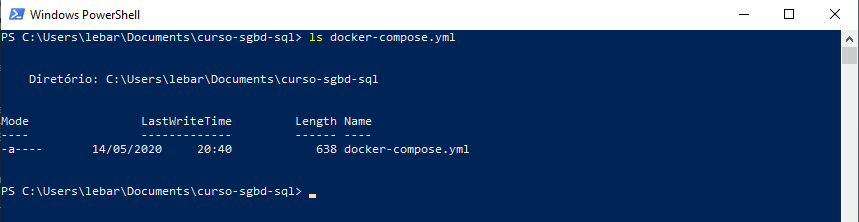
Para mudar de diretório eu usei o comando **cd .\Documents\curso-sgbd-sql**. Também pode ser digitado todo o caminho: **cd C:\Users\lebar\Documents\curso-sgbd-sql**



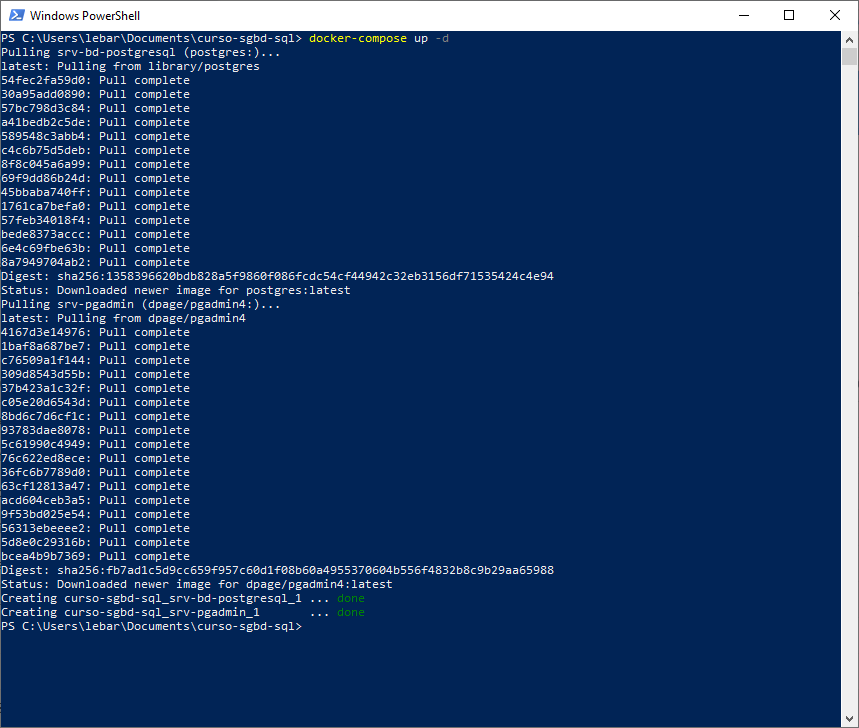
Para verificar se o arquivo está no diretório, digite o comando **ls** (letras L e S) e o nome do arquivo que está procurando.

O comando ls lista o conteúdo do diretório (arquivos e diretórios). O comando seguido de um nome, **ls docker-compose.yml**, vai listar somente o arquivo procurado (o docker-compose.yml) se existir no diretório.

A figura abaixo mostra o comando e o resultado da busca, o nome do arquivo na lista.



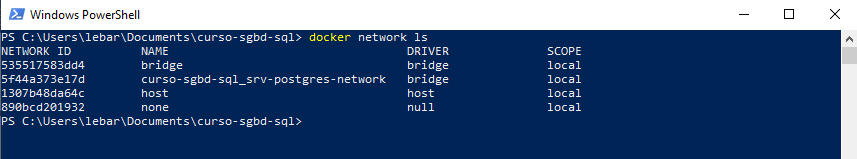
Vamos executar o comando **docker-compose up -d** que vai criar os containers. Caso as imagens do PostgreSQL e do PGAdmin ainda não exista na máquina, esse comando vai realizar o download (é o que ocorreu neste caso).



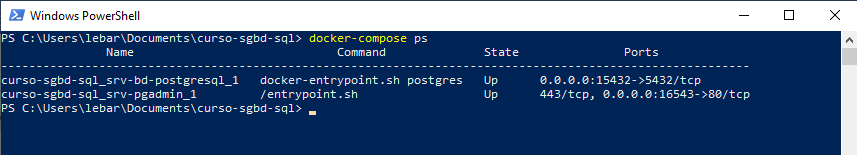
### **PASSO 3 – Verificando se funcionou**

Ao final, vamos executar alguns comandos Docker para verificar o que foi criado pelo docker-compose.

Primeiro vamos verificar a criação da rede (network) srv-postgres-network com o comando **docker network ls**. Veja que tem o **curso-sgbd-sql\_postgres-network**, que é a concatenação do nome do diretório, **curso-sgbd-sql**, com o nome da network, **srv-postgres-network**.

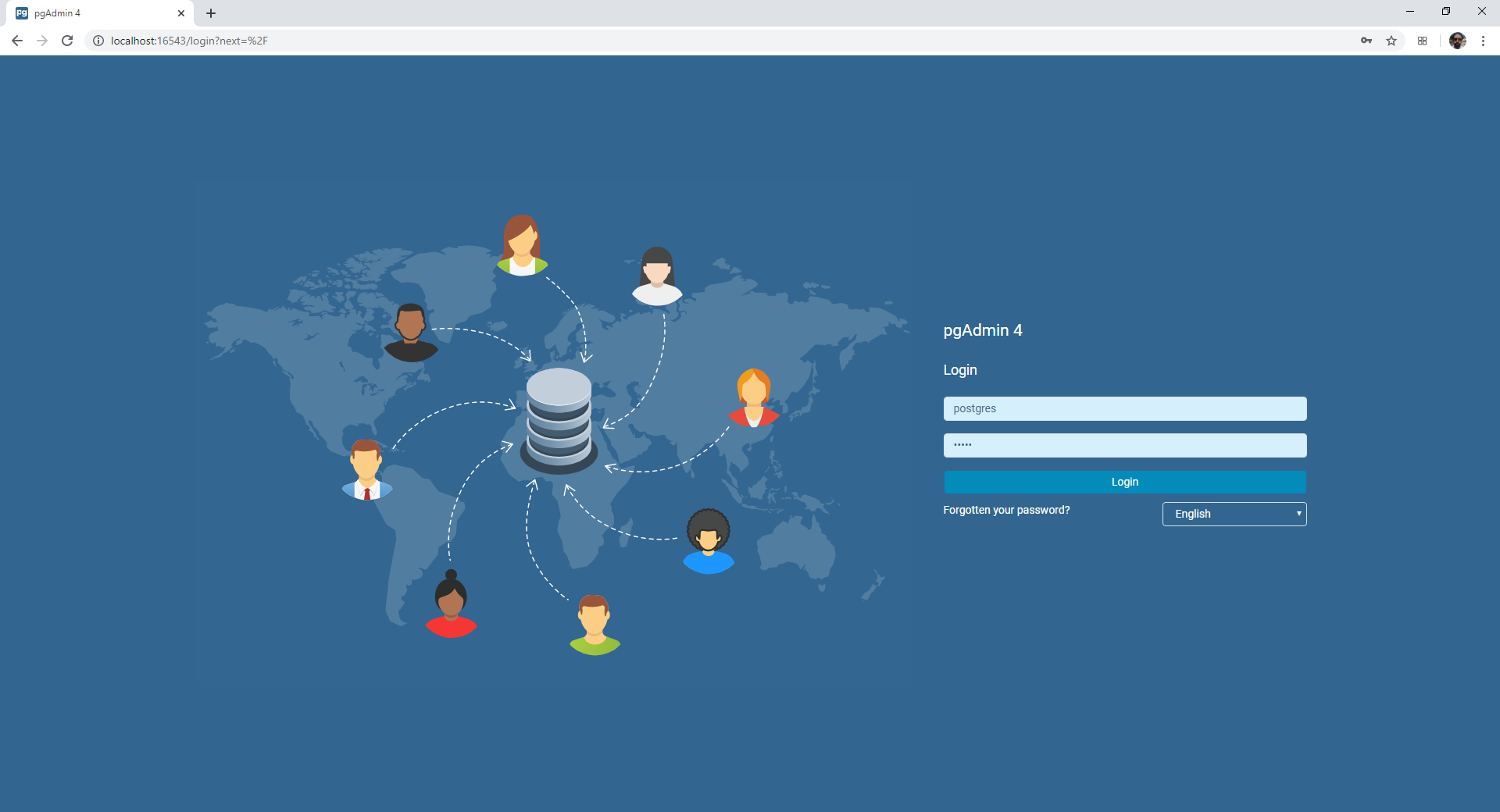


Agora vamos verificar a criação do banco de dados PostgreSQL e do PGAdmin com o comando **docker-compose ps**. Podemos ver o banco de dados PostgreSQL na porta 15432 e o PGAdmin 4 na porta 16543.



### **PASSO 4 – Testando e configurando o ambiente para usar o banco de dados**

Em um browser (neste exemplo eu uso o Google Chrome), acessem o endereço: <http://localhost:16543>. Aparecerá a tela abaixo:



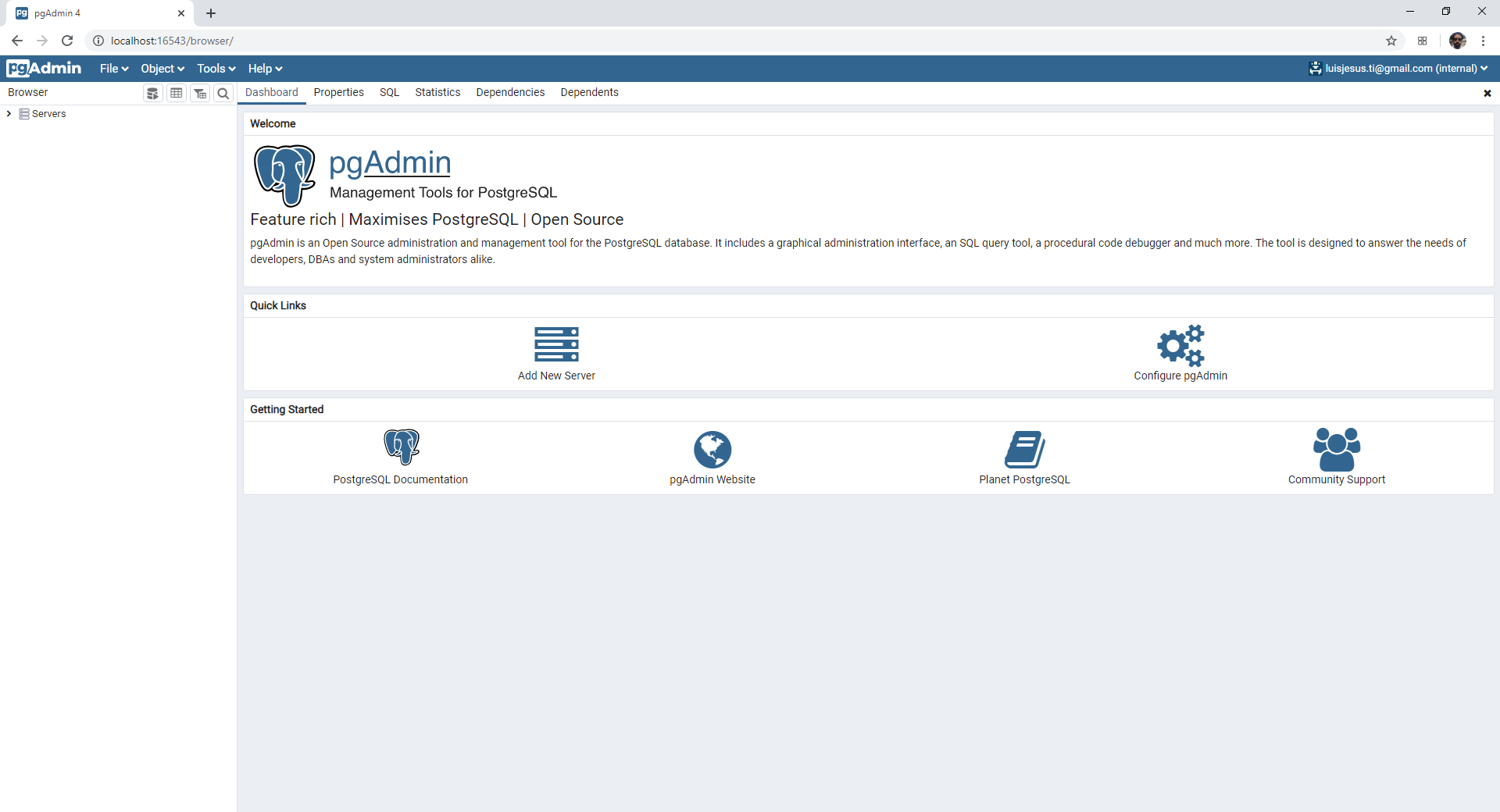
Essa é a tela de acesso ao PGAdmin 4.

Vamos fornecer as credenciais definidas no nosso arquivos docker-compose.yml, quais sejam:

Usuário: luisjesus.ti@gmail.com

Senha: curso

Pronto! Já estamos no painel de gerenciamento PGAdmin 4 (veja a tela abaixo)

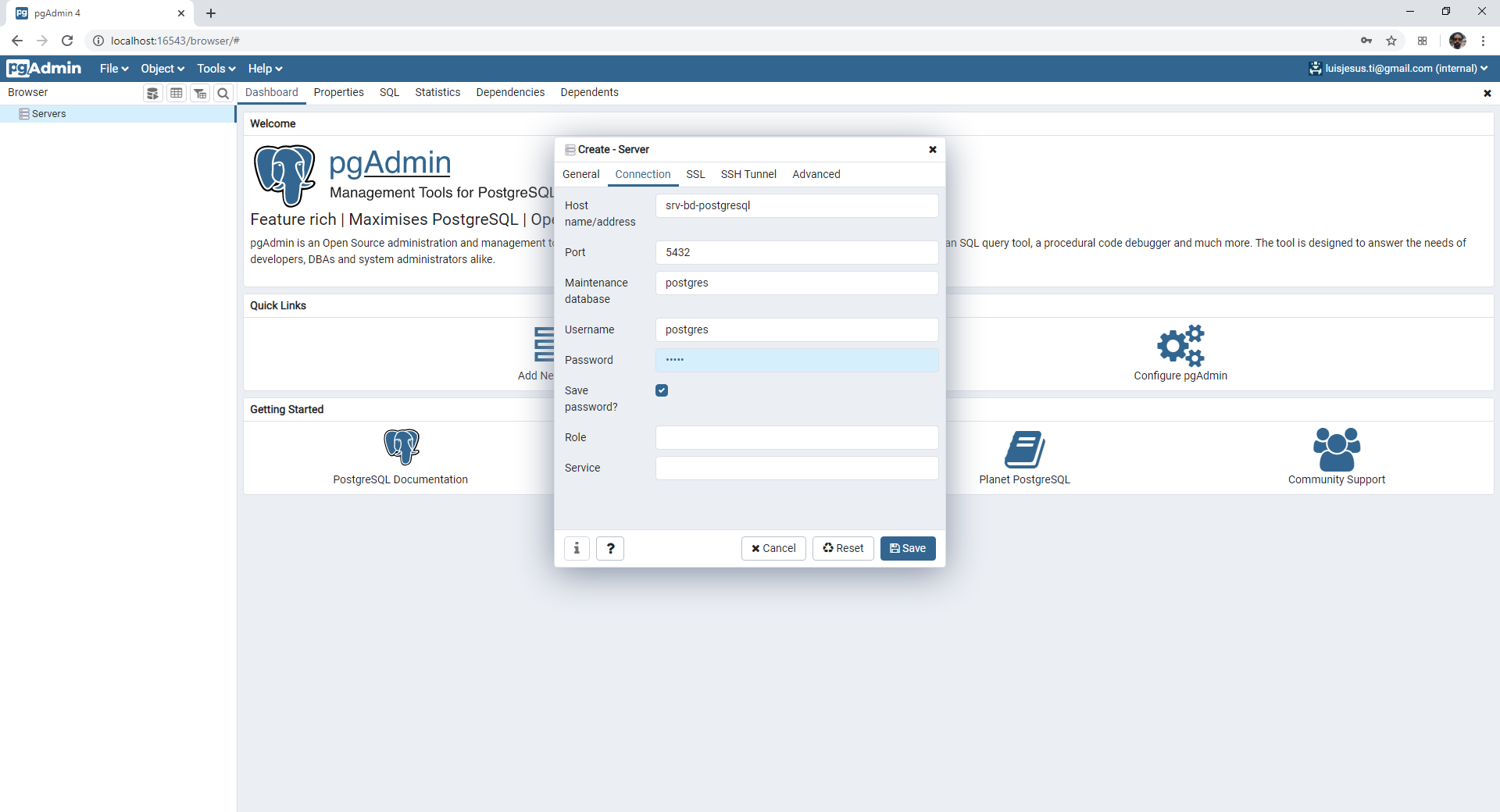


-----

O próximo passo é criar a conexão com a instância de banco de dados PostgreSQL.

Para isso vamos:

- Adicionar um novo Servidor (*Add New Server*). Aparecerá a janela abaixo.



Para criar a conexão para acesso à instância do PostgreSQL, vamos informar:

**Host name/address**: srv-bd-postgresql

- Esse é o nome do serviço PostgreSQL que definimos no nosso docker-compose.yml.

**Port**: 5432

- Essa é a porta padrão de acesso ao container. Porta disponível na rede srv-postgres-network, e definido no nosso docker-compose.yml.

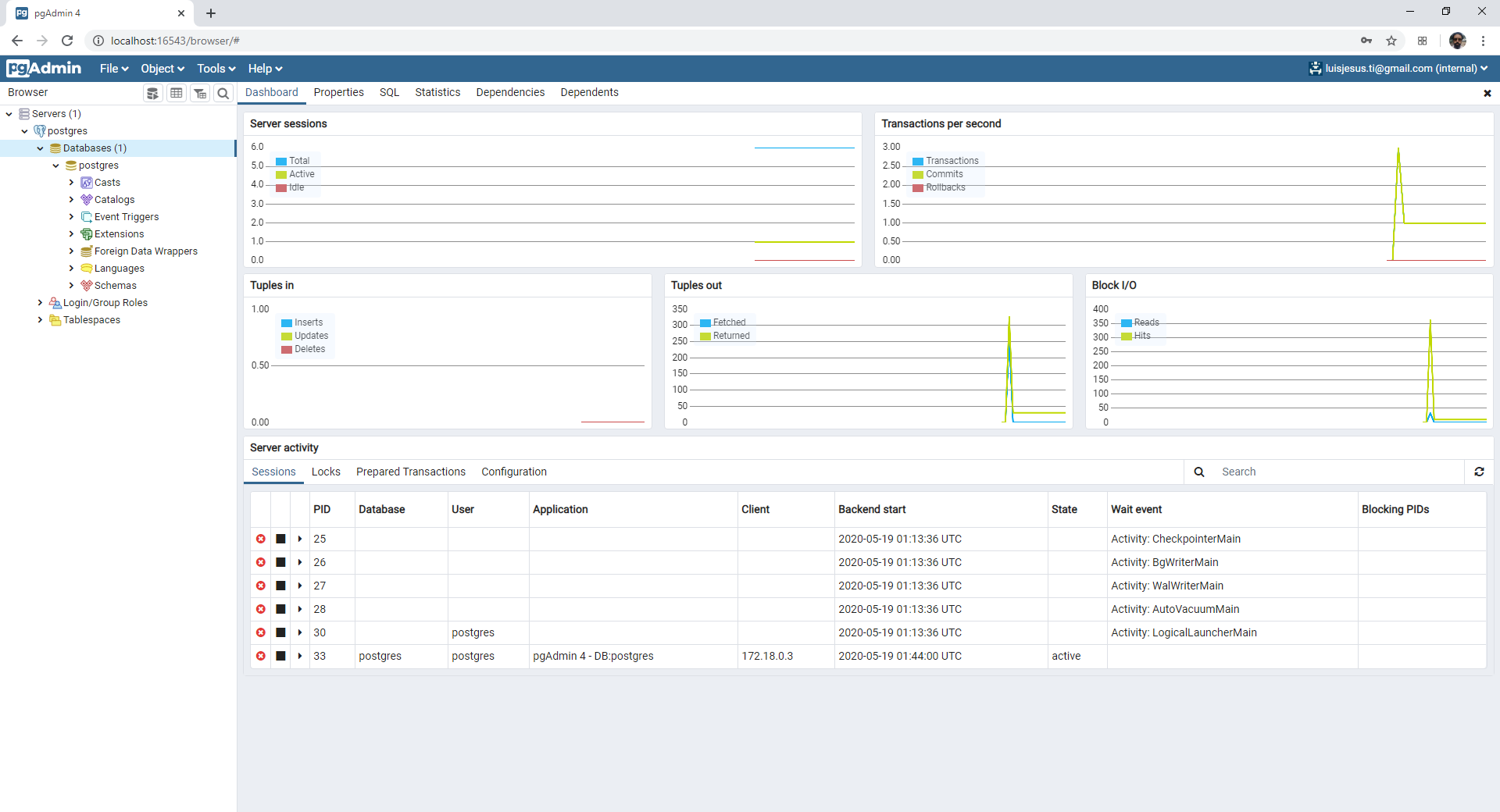
**Username**: postgres

- Valor padrão definido no nosso docker-compose.yml.

**Password**: curso

- Valor padrão definido no nosso docker-compose.yml.

Feito isso, já podemos ver, na figura abaixo, que o PGAdmin 4 acessa o banco de dados PostgreSQL.



[PostgreSQL - Docker Hub](https://hub.docker.com/_/postgres/)

[pgAdmin 4 - Docker Hub](https://hub.docker.com/r/dpage/pgadmin4/)