# BACKUP .SQL

## Como fazer o backup do banco de dados em produção:

**1. -** Abrir o terminal no seu sistema operacional
**2. -** Criar uma pasta e acessá-la
**3. -** Abrir o terminal dentro da pasta
**4. -** Copiar o link pg_dump dbname=mydb2 -f mydb2dump.sql
**4.1 –** Substitua o dbname=mydb2 pelas seguintes credenciais:

- **a.** --host=endpoint
- **b.** --port=5432
- **c.** --username= nomeUsuario
- **d.** --password
- **e.** --dbname=nomeBancoDeDados

**4.2 –** Renomeie o mydb2dump.sql para o nome do arquivo de backup, não esquecendo de colocar a extensão .sql no final

**Segue abaixo o exemplo:**

pg_dump --host=municipios.cerwdsv23421ds1sa.us-oest-2.rds.amazonaws.com --port=5432 --username=municipios-admin --password –dbname=municipios -f municipios.backup

**5. -** Após todo o processo do tópico anterior, pressionar a tecla enter
**6. -** Digitar a senha do banco de dados
**7. -** Após finalizado o processo de backup, será criado o arquivo dentro da pasta selecionada.

# CRIAR DOCKER COMPOSE QUE SUBA O BANCO LOCALMENTE

→ Criar um docker-compose.yml na sua pasta local e o abrir no VSCode

docker-compose.yml

```yaml
version: '3.8'

services:
  database:
    image: postgres
    container_name: containerDB
    restart: always
    ports:
      - 5431:5432
    environment:
      - POSTGRES_PASSWORD=dgs589
      - POSTGRES_DB=container
    volumes:
      - pgdata:/data/postgres

volumes:
  pgdata:
    driver: local

```

**Observação:** Está sendo utilizado a porta 5431 do local, pois a 5432 já está sendo utilizado pelo postgres local. O 5432 do container permanesse, pois é a porta padrão do postgres

→ Abrir o terminal na pasta local e criar o container com docker compose:
**docker compose -d up**

→ Após, copiar o arquivo .sql para dentro do container
**docker cp nomeBackup.sql nomeContainer:/++**

→ Para entrar no container
**docker exec -it nomeContainer bash**

→ Ao entrar no container, atualizar os pacotes
**apt-get update**

→ Após, instalar o **postgis**
**apt-get install postgis**

→ Abrir o psql:

**psql -U postgres**

→ Criar um banco de dados:

**create database nomeBancoDeDados template template0**

→ Criar a extensão do postgis

**create extension postgis**

→ Sair do psql e voltar para o terminal do container

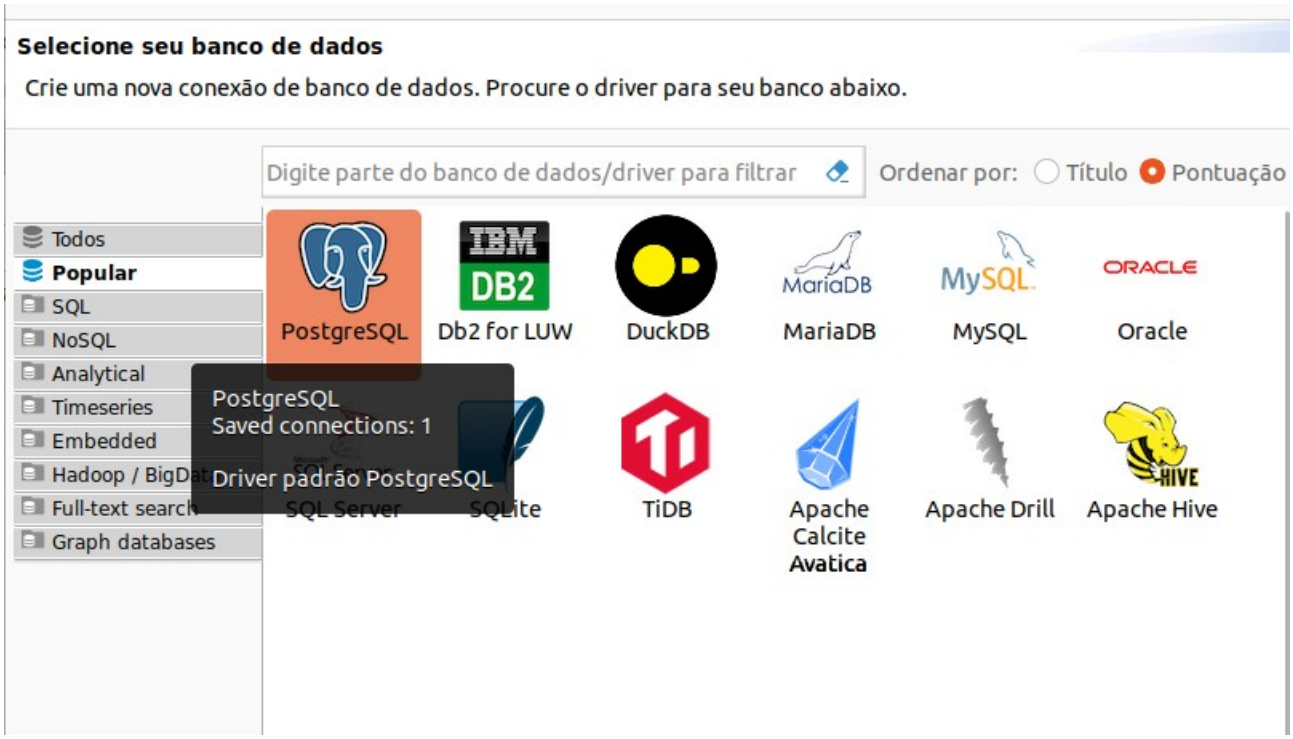
→ Restaurar o banco de dados

**psql -U postgres -W -f nomeBackup.sql nomeBandoDeDados**

## CONECTAR O CONTAINER AO DBEAVER

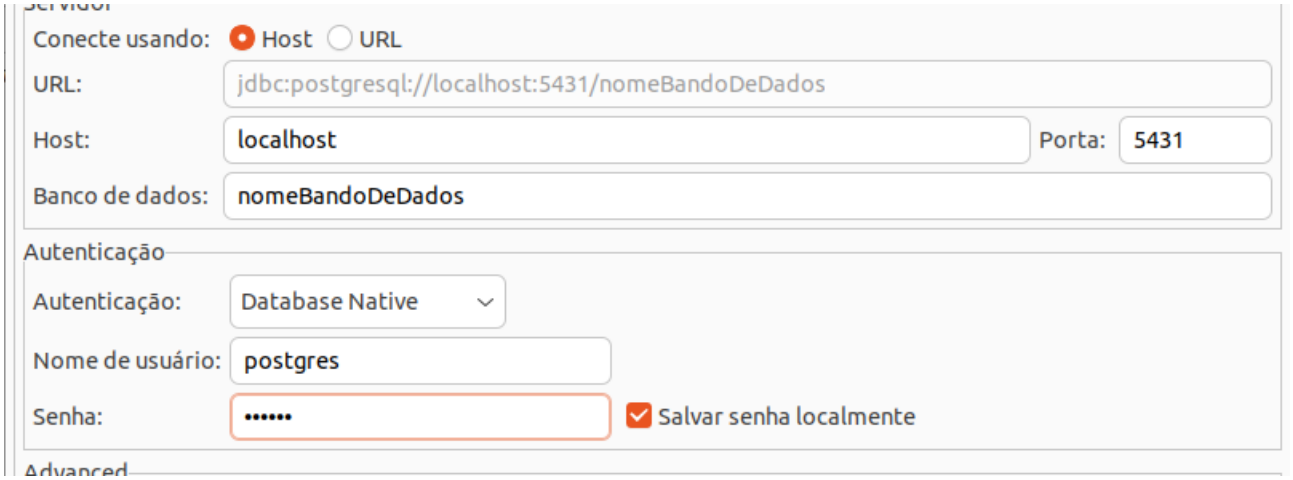
Para conectar o container no DBEAVER, basta:

→ Criar uma nova conexão e escolher o banco **PostgreSQL**

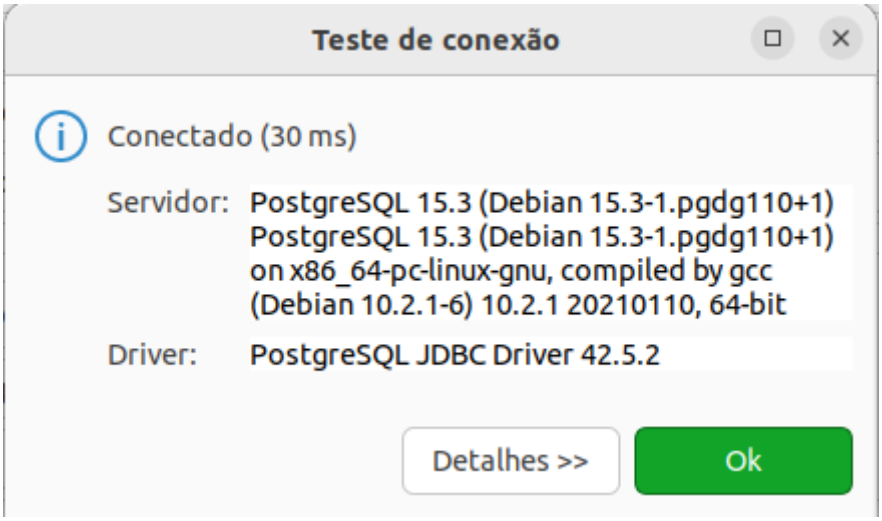



→ Para configurar a conexão, basta:

- Host: **localhost**
- Banco de Dados: **nomeBancoDeDados**
- Porta: **5431** //porta configurada no docker-compose
- Nome de usuário: **postgres**
- Senha: **senhaConfigurada** no docker-compose




→ Após, testar conexão



# RESTAURAÇÃO DO BANCO LOCAL A PARTIR DE UM SQL SEM PRECISAR GERAR UM DOCKER-COMPOSE

Criar um container com a imagem do **postgres**

**docker run --name testeContainer -e POSTGRES_PASSWORD=Dgs589*-+ -d -p 5430:5432 postgres**

Copiar o .sql para dentro do container

**docker cp nomeArquivo.sql nomeContainer:/**

Executar o container

**docker exec -it nomeContainer bash**

No terminal do container, atualizar as dependências

**apt-get update**

Instalar o **postgis**

**apt-get install postgis**

Abrir o psql no terminal do container

**psql -U postgres**

Criar um banco de dados vazio

**create database nomeDB template template0**

Criar a extensão do postgis

**create extension postgis**

sair o psql, ir até o terminal do container e restaurar o banco de dados

**psql -U postgres -W -f nomeArquivo.sql nomeDB**