

Reporte Instalación Docker y PostgreSQL

Integrante: Arturo Gonzáles Bretón

Para llevar a cabo la instalación del sistema operativo *Docker*, el SDBD *PostgreSQL* y la herramienta para administrar bases de datos, *DBeaver*, realicé lo siguiente:

Antes de pasar a la instalación de los componentes, es importante mencionar las especificaciones del equipo en el que realicé la instalación:

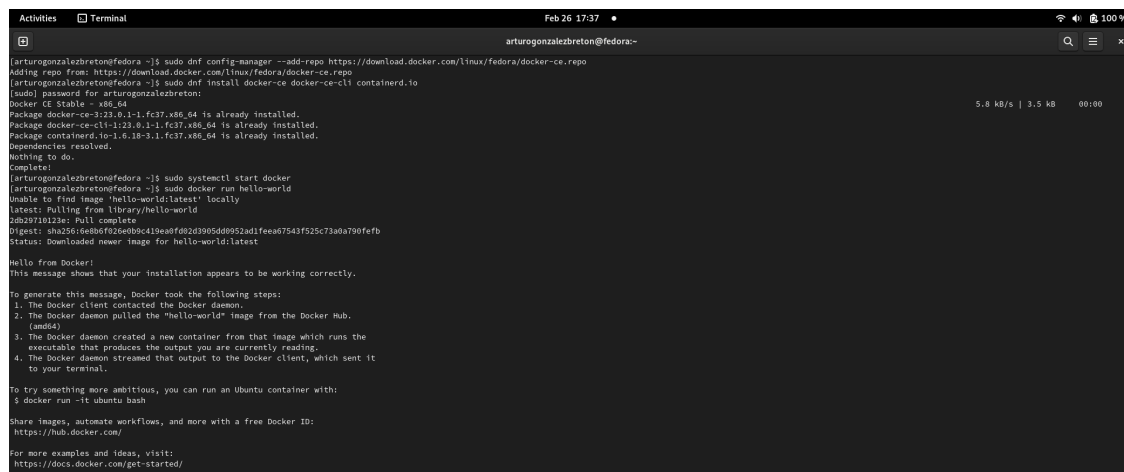
- Sistema operativo: Linux
- Distribución: Fedora
- Versión: 37

```
[arturogonzalezbreton@fedora ~]$ lsb_release -d  
Description:    Fedora release 37 (Thirty Seven)  
[arturogonzalezbreton@fedora ~]$
```

Figura 1: Version del sistema operativo

El primer paso para configurar el equipo, fue instalar *Docker* y verificar que la instalación fuera correcta. Para ello se llevó a cabo lo siguiente:

1. Agregar el repositorio a las fuentes DNF.
2. Instalar los paquetes *docker-ce*, *docker-ce-cli* y *containerd.io*.
3. Iniciar el servicio de *Docker*.
4. Verificar que se pudiera trabajar con las imágenes de *Docker*.



```
Activities Terminal Feb 26 17:37
arturogonzalezbreton@fedora:~$ sudo dnf config-manager --add-repo https://download.docker.com/linux/fedora/docker-ce.repo
Adding repo from: https://download.docker.com/linux/fedora/docker-ce.repo
[arturogonzalezbreton@fedora ~]$ sudo dnf install docker-ce docker-ce-cli containerd.io
[sudo] password for arturogonzalezbreton:
Docker CE Stable - x86_64
Package docker-ce-2:23.0.1-1.fc37.x86_64 is already installed.
Package docker-ce-cli-1:23.0.1-1.fc37.x86_64 is already installed.
Package containerd.io-1.6.18-3.1.fc37.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[arturogonzalezbreton@fedora ~]$ sudo systemctl start docker
[arturogonzalezbreton@fedora ~]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29718123e: Pull complete
Digest: sha256:3b6913c9218b3333e9d934f5e67543f525c73a0a790fe9b
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (cmd)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

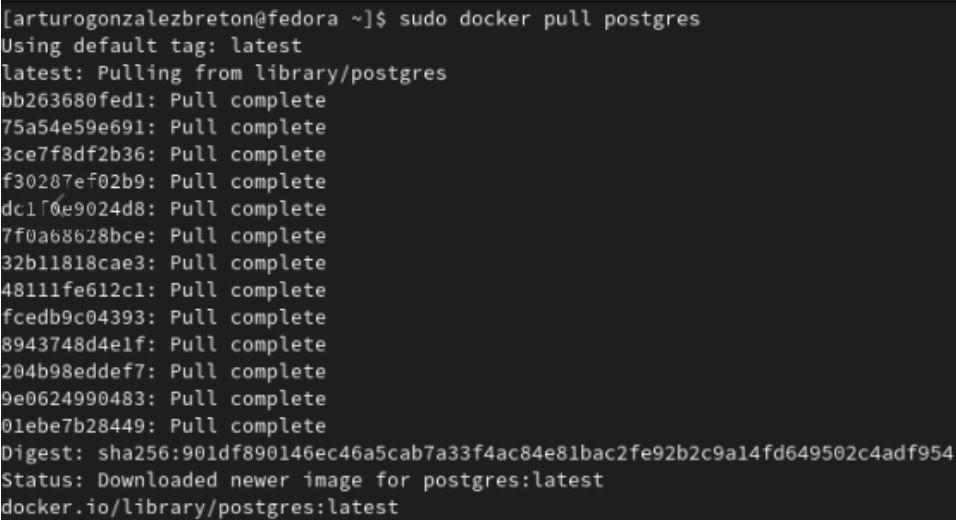
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Figura 2: Instalación de *Docker*

Es importante señalar que la distribución que utilizo ya contaba desde un inicio con el sistema operativo *Docker*, por lo que no hizo falta descargar los paquetes. El segundo paso a realizar, fue descargar la imagen de postgres que utilizará *Docker*:

1. Descargar imagen de postgres.



```
[arturogonzalezbreton@fedora ~]$ sudo docker pull postgres
Using default tag: latest
latest: Pulling from library/postgres
bb263680fed1: Pull complete
75a54e59e691: Pull complete
3ce7f8df2b36: Pull complete
f30287ef02b9: Pull complete
dc1f0e9024d8: Pull complete
7f0a68628bce: Pull complete
32b11818cae3: Pull complete
48111fe612c1: Pull complete
fcedb9c04393: Pull complete
8943748d4e1f: Pull complete
204b98eddef7: Pull complete
9e0624990483: Pull complete
01ebe7b28449: Pull complete
Digest: sha256:901df890146ec46a5cab7a33f4ac84e81bac2fe92b2c9a14fd649502c4adf954
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest
```

Figura 3: Descarga de la imagen de postgres

Después, se creó el contenedor para la imagen de postgres. En este paso, también se ejecuta el contenedor, sin embargo, en mi equipo el puerto 5432, ya estaba ocupado, por lo que salió un error al crear el contenedor; esto sucedió independientemente de la creación del contenedor:

1. Crear el contenedor para la imagen de postgres.

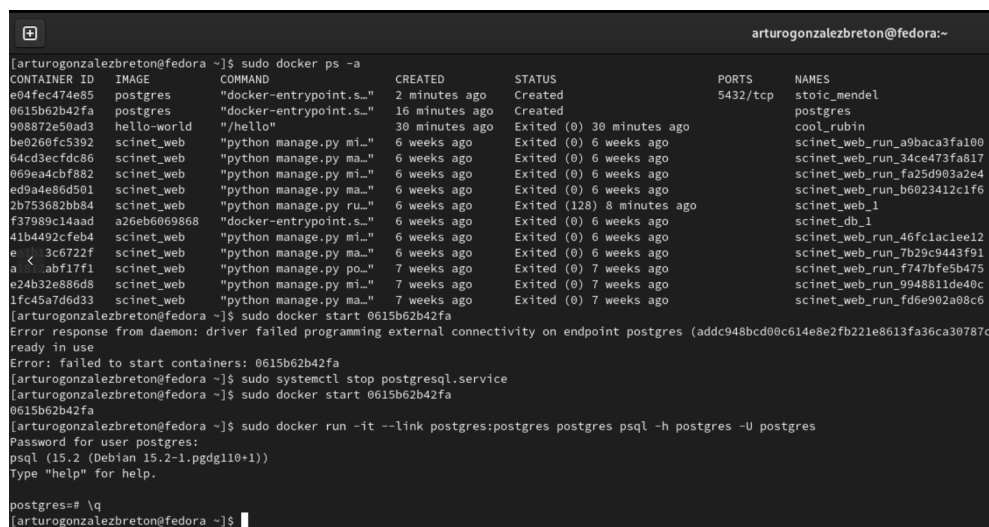
```
[arturogonzalezbreton@fedora ~]$ sudo docker run -d --name postgres -e POSTGRES_PASSWORD=mysecretpassword -p 5432:5432 postgres
[sudo] password for arturogonzalezbreton:
0615b62b42faa22a1d40c15e8f1d3eab8acbfd66e635d72cd880c597472c922a
docker: Error response from daemon: driver failed programming external connectivity on endpoint postgres (dadf04896dff5bfc62c74c
dress already in use.
[arturogonzalezbreton@fedora ~]$ lsof -i :5432
```

Figura 4: Creación del contenedor

Al no encontrar el proceso que mantenía ocupado el puerto 5432, intenté iniciar la ejecución del contenedor nuevamente, lo que resultó en el mismo error. Momentos después, recordé que estaba activo el servicio de postgres en mi equipo y dicho servicio es el que mantenía el puerto ocupado.

Bastó con desactivar este servicio para iniciar la ejecución del contenedor y realizar la conexión a PostgreSQL:

1. Verificar el ID del contenedor.
2. Iniciar ejecución del contenedor.
3. Detener servicio de postgres.
4. Iniciar ejecución del contenedor.
5. Conectar a PostgreSQL utilizando psql.



```
arturogonzalezbreton@fedora:~  
[arturogonzalezbreton@fedora ~]$ sudo docker ps -a  
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS          NAMES  
e04fec474e85   postgres  "docker-entrypoint.s..." 2 minutes ago Created       5432/tcp      stoic_mendel  
0615b62b42fa   postgres  "docker-entrypoint.s..." 16 minutes ago Created                               postgres  
908872e50ad3   hello-world "/hello"                 30 minutes ago Exited (0) 30 minutes ago cool_rubin  
be0260fc5392   scinet_web "python manage.py mi..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_a9baca3fa100  
64cd3ecfcd86   scinet_web "python manage.py ma..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_34ce473fa817  
069ea4cbf882   scinet_web "python manage.py mi..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_fa25d903a2e4  
ed9a4e86d501   scinet_web "python manage.py ma..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_b6023412c1f6  
2b753682bb84   scinet_web "python manage.py ru..." 6 weeks ago   Exited (128) 8 minutes ago scinet_web_1  
f37989c14aad   a26eb069868 "docker-entrypoint.s..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_db_1  
41b4492cfeb4   scinet_web "python manage.py mi..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_46fclac1ee12  
e3c6722f      scinet_web "python manage.py ma..." 6 weeks ago   Exited (0) 6 weeks ago   scinet_web_run_7b29c9443f91  
a3abf17f1     scinet_web "python manage.py po..." 7 weeks ago   Exited (0) 7 weeks ago   scinet_web_run_f747bfe5b475  
e24b32e886d8   scinet_web "python manage.py mi..." 7 weeks ago   Exited (0) 7 weeks ago   scinet_web_run_9948811de40c  
1fc45a7d6d33   scinet_web "python manage.py ma..." 7 weeks ago   Exited (0) 7 weeks ago   scinet_web_run_fd6e902a08c6  
[arturogonzalezbreton@fedora ~]$ sudo docker start 0615b62b42fa  
Error response from daemon: driver failed programming external connectivity on endpoint postgres (addc948bcd08c614e82fb221e8613fa36ca30787c...  
ready in use  
Error: failed to start containers: 0615b62b42fa  
[arturogonzalezbreton@fedora ~]$ sudo systemctl stop postgresql.service  
[arturogonzalezbreton@fedora ~]$ sudo docker start 0615b62b42fa  
0615b62b42fa  
[arturogonzalezbreton@fedora ~]$ sudo docker run -it --link postgres:postgres postgres psql -h postgres -U postgres  
Password for user postgres:  
psql (15.2 (Debian 15.2-1.pgdg110+1))  
Type "help" for help.  
  
postgres=# \q  
[arturogonzalezbreton@fedora ~]$
```

Figura 5: Conexión a PostgreSQL

Una vez instalada esta herramienta, procedí a abrirla. Inmediatamente después de hacer esto, se mostró en pantalla un menú para seleccionar el motor de base de datos. Seleccioné PostgreSQL:

1. Seleccionar motor de base de datos.

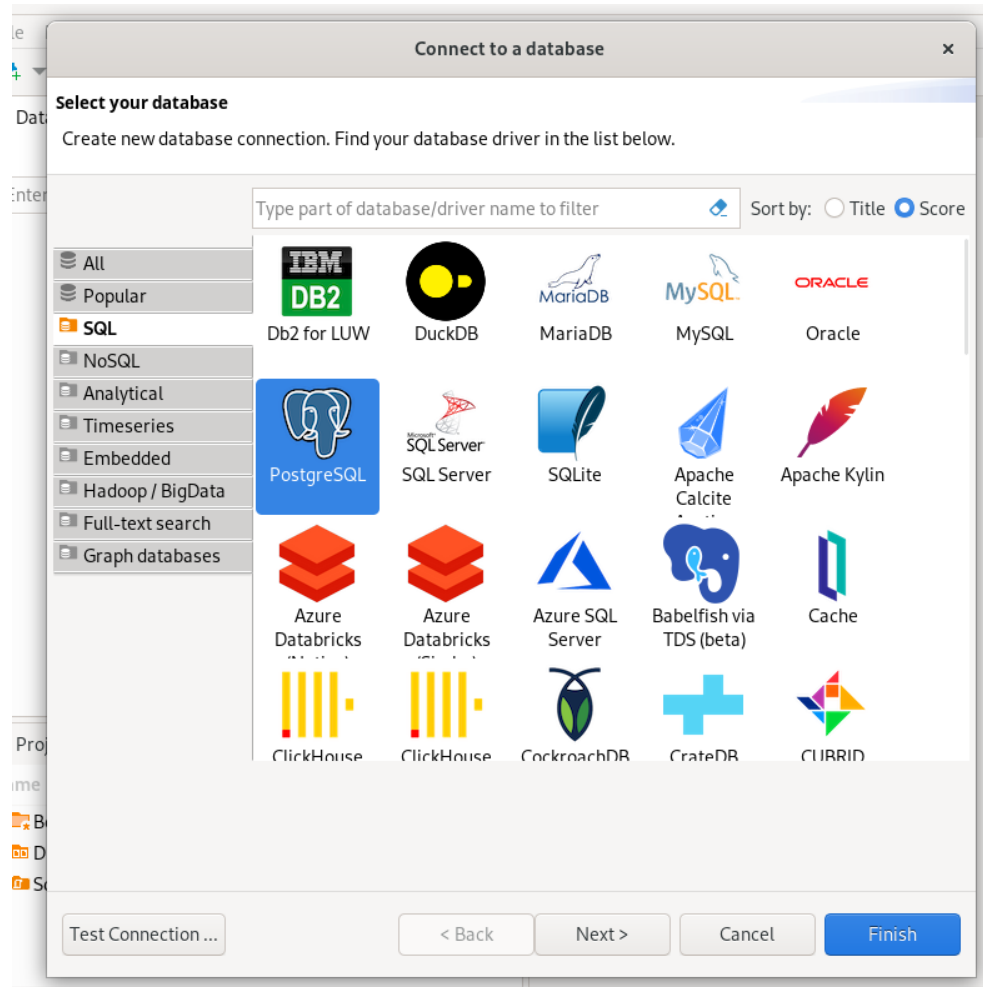


Figura 7: Seleccionar motor de base de datos

Posteriormente, conecté la base de datos con DBeaver:

1. Conectar base de datos.

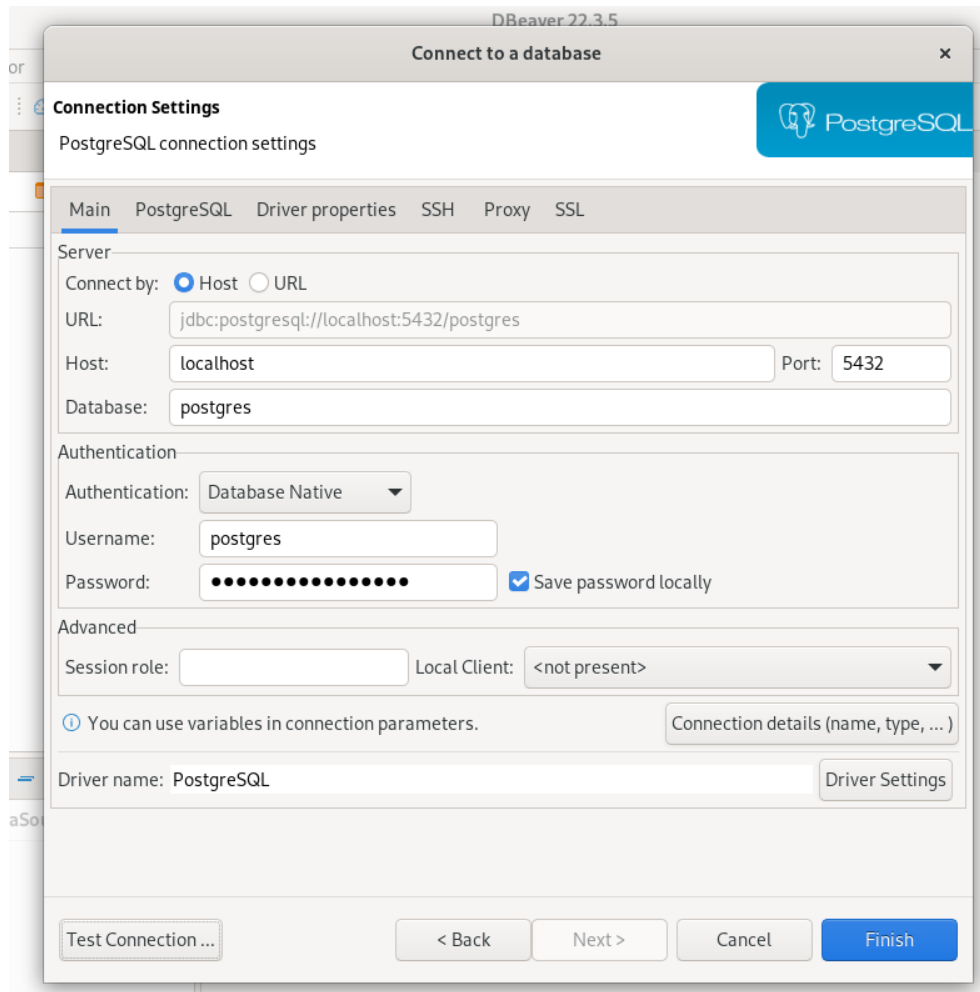
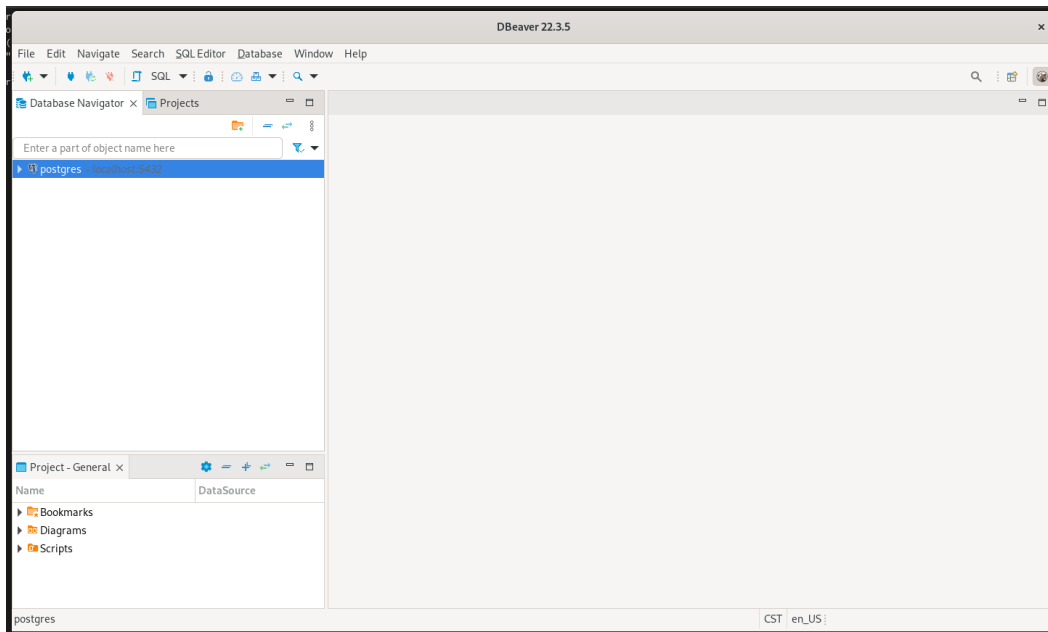


Figura 8: Creación de la conexión

Una vez creada la conexión, la herramienta DBeaver quedó lista para utilizarse con el contenedor:

1. DBeaver listo para utilizarse con PostgreSQL.



La instalación y configuración me tomó aproximadamente 20 minutos, pues, al contar ya con los paquetes de *Docker*, no tuve que descargar nada además de DBeaver y la imagen de postgres. El procedimiento fue muy sencillo y el único problema al que me enfrenté, fue que el puerto en el que intentaba ejecutar el contenedor, ya estaba ocupado; sin embargo, este problema se resolvió de manera muy sencilla, por lo que casi no entorpeció el procedimiento.