

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO  
FACULTAD DE CIENCIAS



FUNDAMENTOS DE BASES DE DATOS

SEMESTRE 2022-2

---

## Práctica 01: Bitácora Padilla

---

*Profesor:*

Gerardo Áviles Rosas

*Ayudantes de Teoría:*

Gerardo Uriel Soto Miranda

Rocío Aylin Huerta González

*Ayudantes de Laboratorio:*

Ricardo Badillo Macías Rodrigo

Alejandro Sánchez Morales

Marzo 7, 2022

## Sistema operativo y versión

OS: Manjaro Linux x86\_64  
Kernel: 5.10.102-1-MANJARO  
DE: KDE Plasma 5.24.2  
CPU: AMD Ryzen 3 3300U @ 2.1GHz  
Memoria: 9932MiB

## Distribución

Distribución: Manjaro Linux 21.4.2 “Qonos”

## Versión de la instalación

PostgreSQL 14.2  
(Debian 14.2-1.pgdg110+1) on x86\_64-pc-linux-gnu,  
compiled by gcc (Debian 10.2.1-6) 10.2.1 20210110,  
64-bit

## Tiempo requerido

Aproximadamente 45 minutos.

## Explicación paso a paso de lo que realizaste

1. Antes de instalar algún paquete actualizamos el sistema utilizando el siguiente comando:

```
1 ~$ sudo pacman -Syu
```

- Esta vez se actualizó el kernel, así que reiniciamos el sistema:

```
1 ~$ sudo reboot now
```

2. Instalación de Docker. Instalamos Docker desde los repositorios de Manjaro ejecutando el siguiente comando:

```
1 ~$ sudo pacman -S docker
```

- 2.1. (Opcional) Dar privilegios de root a Docker. Creamos un nuevo grupo llamado docker, añadimos nuestro usuario y reevaluamos las credenciales. Para ello ocupamos los comandos:

```
1 ~$ sudo groupadd docker  
2 ~$ sudo usermod -aG docker $USER  
3 ~$ newgrp docker
```

Si decidimos saltarnos éste paso, entonces todos los comandos que impliquen a docker deben ejecutarse como superusuario.

2.2. Activamos el servicio de Docker para que se inicialice junto con nuestro sistema:

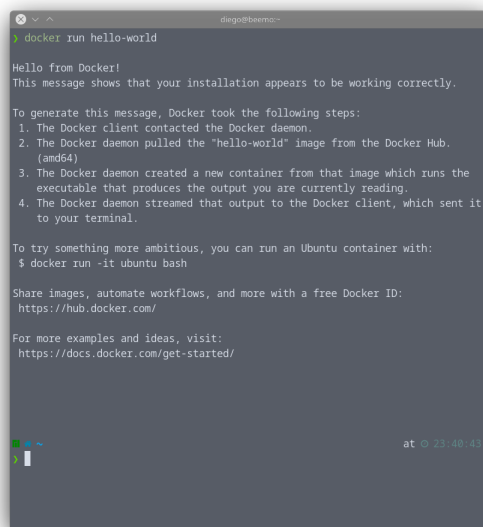
```
1 ~$ sudo systemctl enable docker.service
2 ~$ sudo systemctl enable containerd.service
```

2.3. Reiniciamos el sistema.

```
1 ~$ sudo reboot now
```

2.4. Verificamos la instalación y que el servicio esté habilitado ejecutando el comando:

```
1 ~$ docker run hello-world
```



```
diago@beemo:~$ docker run hello-world

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.
   (amd64)
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/

at 23:49:43
```

3. Instalamos Docker Compose con el propósito de facilitar la instalación de PostgreSQL.

3.1. Descargamos la versión estable más reciente de Docker Compose con los siguientes comandos:

```
1 ~$ DOCKER_CONFIG=${DOCKER_CONFIG:-$HOME/.docker}
2 ~$ mkdir -p $DOCKER_CONFIG/cli-plugins
3 ~$ curl -SL https://github.com/docker/compose/releases/download/
4 v2.2.3/docker-compose-linux-x86_64 -o $DOCKER_CONFIG/cli-plugins/
5 docker-compose
```

3.2. Concedemos permisos de ejecución al binario que acabamos de descargar:

```
1 ~$ chmod +x $DOCKER_CONFIG/cli-plugins/docker-compose
```

3.3. Y verificamos la instalación:

```
1 ~$ docker compose version
```

A terminal window titled 'diego@beemo:~' showing the command 'docker compose version' and its output 'Docker Compose version v2.2.3'. The terminal has a dark background with light green text. A timestamp 'at 00:07:18' is visible in the bottom right corner.

```
diego@beemo:~  
> docker compose version  
Docker Compose version v2.2.3  
at 00:07:18
```

4. Instalación de PostgreSQL. Utilizando la herramienta Docker Compose, seguimos los siguientes pasos para crear un contenedor e instalar PostgreSQL.

4.1. Nos posicionamos en el directorio \$HOME y creamos los directorios docker y postgres:

```
1 ~$ cd ~  
2 ~$ mkdir -p docker/postgres
```

4.2. Ahora nos movemos al directorio postgres:

```
1 ~$ cd docker/postgres
```

4.3. Creamos un directorio para la persistencia de datos:

```
1 ~/d/p$ mkdir -p volumes/postgres_data
```

4.4. Escribimos un archivo llamado docker-compose.yml, y escribimos lo siguiente:

```
1 ##### docker-compose.yml #####  
2 version: '3'  
3 services:  
4   postgres:  
5     image: 'postgres:latest'  
6     restart: always  
7     env_file:  
8       - database.env  
9     volumes:  
10      - ./volumes/postgres_data:/var/lib/postgresql/data  
11     ports:  
12      - '5432:5432'  
13  
14 volumes:  
15   postgres_data:
```

```

1: docker-compose.yml  3: database.env  buffers  tab 1/1
13 version: '3'
12 services:
11   postgres:
10     image: 'postgres:latest'
9     restart: always
8     env_file:
7       - database.env
6     volumes:
5       - ./volumes/postgres_data:/var/lib/postgresql/data
4     ports:
3       - '5432:5432'
2
1 volumes:
14   postgres_data:

```

NORMAL docker-compose.yml yam... 100% 14:17  
"docker-compose.yml" 14L, 250C escritos

4.5. Guardamos, y dentro del mismo directorio creamos el archivo database.env. Donde vamos a escribir nuestras credenciales y a darle un nombre a la base de datos:

```

1                                     ##### databse.env #####
2   POSTGRES_USER = docker
3   POSTGRES_PASSWORD = secret
4   POSTGRES_DB = postgres

```

4.6. Guardamos el archivo y sin cambiar de directorio ejecutamos el siguiente comando:

```

1   ~/d/p$ docker compose up -d

```

```

$ docker compose up -d
[*] Running 0/14
  postgres Pulling
    77c22623b5a6 Downloading 1.944MB/31.37MB 7.0s
    77c22623b5a6 Downloading 1.813MB/4.414MB 7.0s
    0f6a6a85d014 Download complete 7.0s
    60127288256 Downloading 288.54B/1.418MB 7.0s
    18c92d43e721 Waiting 7.0s
    598625d1c828 Waiting 7.0s
    ec548f0dc89b Waiting 7.0s
    10c57d7e8b40 Waiting 7.0s
    6b54026a980b Waiting 7.0s
    5f04de1f64de Waiting 7.0s
    19248ca3a367 Waiting 7.0s
    1fbbf10b9826 Waiting 7.0s
    745f153e9d8d Waiting 7.0s

```

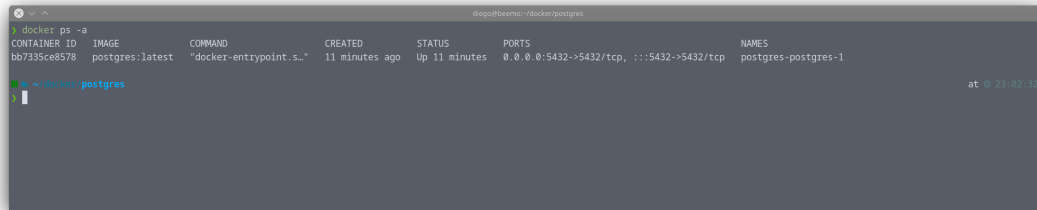
```

$ docker compose up -d
[*] Running 14/14
  postgres Pulled
    77c22623b5a6 Pull complete 91.3s
    77c22623b5a6 Pull complete 91.9s
    0f6a6a85d014 Pull complete 92.0s
    60127288256 Pull complete 92.1s
    18c92d43e721 Pull complete 92.7s
    598625d1c828 Pull complete 92.8s
    ec548f0dc89b Pull complete 92.9s
    10c57d7e8b40 Pull complete 93.0s
    6b54026a980b Pull complete 147.0s
    5f04de1f64de Pull complete 147.1s
    19248ca3a367 Pull complete 147.2s
    1fbbf10b9826 Pull complete 147.2s
    745f153e9d8d Pull complete 147.3s
[*] Running 2/2
  Network postgres_default Created 0.1s
  Container postgres-postgres-1 Star... 0.4s
  Container postgres
    took 2m 33s at 22:58:58

```

4.7. Y revisamos que el contenedor se haya creado correctamente:

```
1 ~$ docker ps -a
```

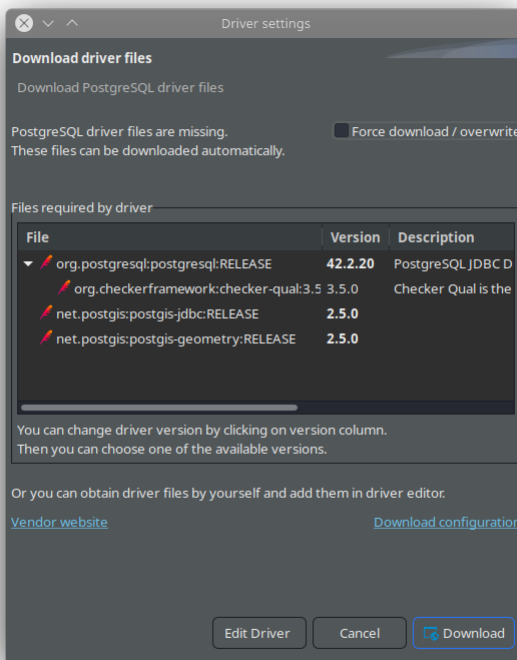


5. Instalación DBeaver. Regresamos al directorio \$HOME e instalamos el cliente SQL DBeaver con los comandos:

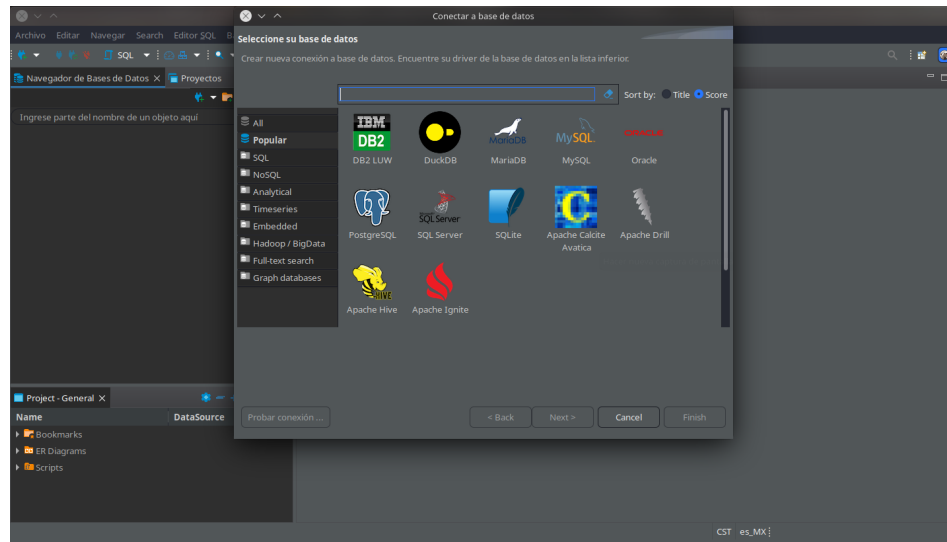
```
1 ~/d/p$ cd ~
2 ~$ sudo pacman -S dbeaver
```

6. Conectar PostgreSQL con DBeaver.

6.1. Abrimos Beaver y descargamos los drivers que nos pide:

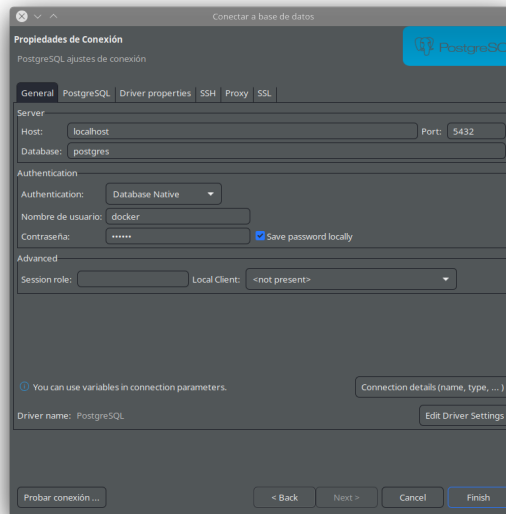


6.2. En el menú que despliega seleccionamos PostgreSQL y hacemos click en siguiente:

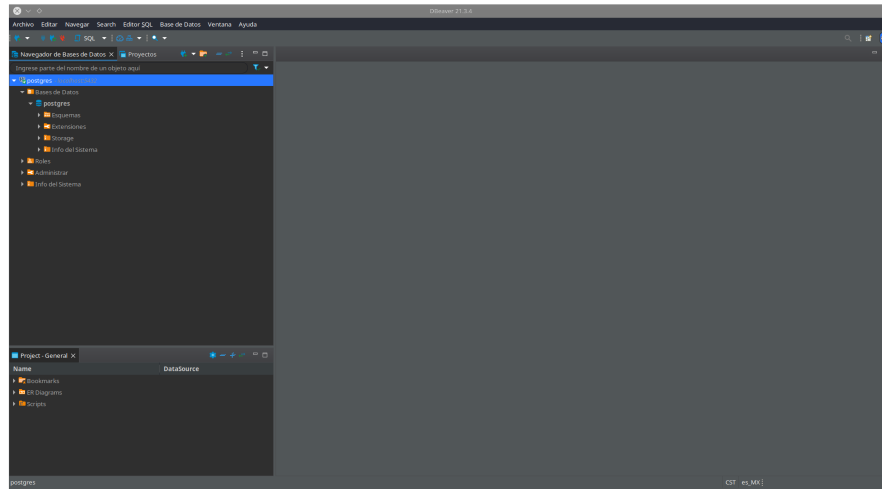


6.3. Escribimos nuestras credenciales en los cuadros correspondientes según lo que especificamos en el archivo database.env:

- Host: localhost
- Port: 5432
- Database: POSTGRES\_DB
- Nombre de usuario: POSTGRES\_USER
- Contraseña: POSTGRES\_PASSWORD



6.4. Hacemos click en finalizar y hemos terminado:



## Comentarios y problemas a los que te enfrentaste durante la instalación

En general no tuve problemas al instalar PostgreSQL con Docker, la parte más tardada fue conectar con DBeaver pues la configuración que escribí al principio para Docker Compose estaba generando errores al momento de crear la base de datos con las credenciales especificadas. Una vez que noté el error y cambié el archivo de configuración no volví a experimentar problemas.