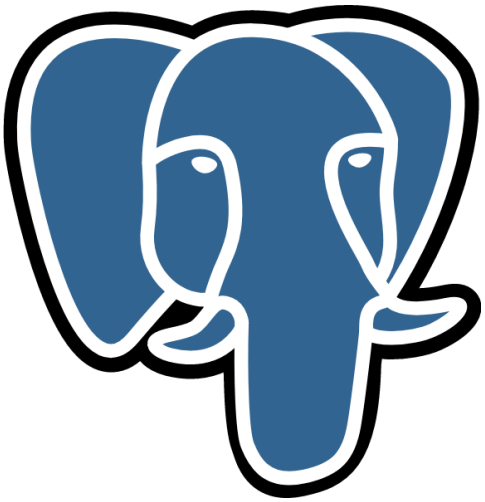


PostgreSQL

Object-relational DataBase Management System

Instal·lació i primers usos:

Per instal·lar el PostgreSQL utilitzarem un contenedor DOCKER, per la qual cosa primer haurem d'instal·lar este últim si encara no el tenim.

Per tant seguirem els següents passos:

1. Si no el tenim[1], instal·lem docker al sistema operatiu seguint les instruccions del següent post (pasos de l'1 al 5).
2. Baixem la darrera **imatge**[2] del PostgreSQL en la següent comanda[3]:

```
profe@profe-ventola ~$ docker pull bitnami/postgresql:latest
latest: Pulling from bitnami/postgresql
4fa569a3cfa6: Pull complete
Digest: sha256:ed9f4476aa1c264f823fea2dcc15d197902acf9cb65a765d452287f702b7099f
Status: Downloaded newer image for bitnami/postgresql:latest
docker.io/bitnami/postgresql:latest
```

[1] Podeu comprovar si teniu el Docker instal·lat en la comanda ***docker --version****, que no us hauria de donar cap error…*

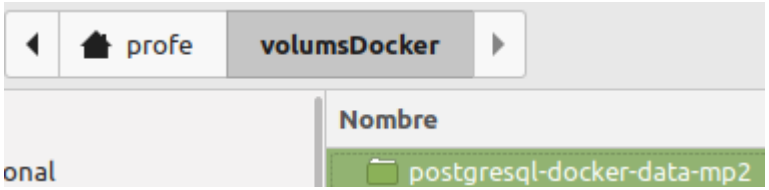
[2] Una imatge del Docker és com una plantilla que mos permetrà crear contenedors a partir d'ella, contenint el software que vulguem utilitzar. A partir d'una sola imatge se poden crear tants contenedors com necessitem

[3] docker pull bitnami/postgresql:latest

3. Podem comprovar que la imatge està descarregada en la següent comanda[4]:

```
profe@profe-ventola ~$ docker images | grep postgresql
bitnami/postgresql                                      latest    d59d7f1760e9   44 hours ago    273MB
```

4. Ara crearem un **contenedor** a partir de la imatge descarregada. El contenedor és com una màquina virtual, però molt més lleugera, on trobarem la nostra base de dades del PostgreSQL.
    a. Si volem que les dades no es perguen al reiniciar el contenedor, haurem de crear un volum corresponent a un directori físic (carpeta) de la nostra màquina, que indicarem a l'ordre de creació (creem la carpeta primer).



Carpeta prèviament creada[5]

    b. Per crear el contenedor executem[6]:

```
profe@profe-ventola ~$ docker run --name=postgresql -d --env=POSTGRESQL_USERNAME
=postgres --env=POSTGRESQL_PASSWORD=postgres --env=POSTGRESQL_DATABASE=test --en
v=TCPIP_SOCKET=TRUE -p 5434:5432 bitnami/postgresql:10
b4fb78e8877ce83b201abe293e6b935083d90f6ec9bb85d3975779713390cc7a
```

*Creo contenedor en nom **postgresql** (--name), el volum correspon al directori **/home/…/postgresql-docker-data-mp2** (-v) i que utilitza la imatge **bitnami/postgresql:10** (-d). Com a paràmetres d'entorn del PostgreSQL li passem l'usuari i password (**postgres**), el port extern:intern pel qual escolta el postgresql (**5434:5432**) i quina serà la base de dades a la que mos connectem (**test**)*

[4] docker images | grep postgresql
[5] No ho he provat, però suposo que s'ha de crear abans d'executar la comanda
[6] docker run --name=postgresql -v ~/volumsDocker/postgresql-docker-data-mp2/raw-data-db:/data/db -d --env=POSTGRESQL_USERNAME=postgres --env=POSTGRESQL_PASSWORD=postgres --env=POSTGRESQL_DATABASE=test --env=TCPIP_SOCKET=TRUE -p 5434:5432 bitnami/postgresql:10

5. Per iniciar i parar el contenedor usarem, respectivament, les següents ordres[7] [8]:

```
profe@profe-ventola ~$ docker start postgresql
postgresql
profe@profe-ventola ~$ docker stop postgresql
postgresql
```

6. Podem comprovar quins ports estan utilitzant els diferents contenedors del Docker, en la següent comanda[9]:

```
profe@profe-ventola ~$ docker container ls --format "table {{.ID}}\t{{.Names}}\t{{.Ports}}" -a
CONTAINER ID   NAMES                PORTS
cf2e7e391ccb   postgresql           5432/tcp
```

Podem vore que el contenedor escolta pel port 5432, que és el port per defecte

7. Per poder connectar a la BD des del terminal, iniciarem una connexió al contenedor mitjançant el ***bash***, de la següent forma[10]:

```
profe@profe-ventola ~$ docker exec -it  postgresql bash
I have no name!@cf2e7e391ccb:/$
```

8. A tots els efectes estem a disposició d'accedir a la base de dades des del client del PostgreSQL, d'igual forma a com ho faríem si haguéssem instal·lat el servidor a la nostra màquina[11]:

```
I have no name!@cf2e7e391ccb:/$ psql -d test -U postgres -h localhost
Password for user postgres:
psql (15.2)
Type "help" for help.

test=#
```

[7] docker start postgresql
docker stop santis-mysql
[8] Per defecte, cada vegada que vulguem usar el MySQL haurem d'arrancar el contenedor
[9] docker container ls --format "table {{.ID}}\t{{.Names}}\t{{.Ports}}" -a
[10] docker exec -it postgresql bash
[11] psql -d test -U postgres -h localhost

GUI d'administració

Per versions del PostgreSQL a partir de la 10 podem usar la versió 4 del pgAdmin, i per anteriors la versió 3.

Jo he instal·lat un contenedor en la modalitat web del pgAdmin 4, usant la següent comanda [12]:

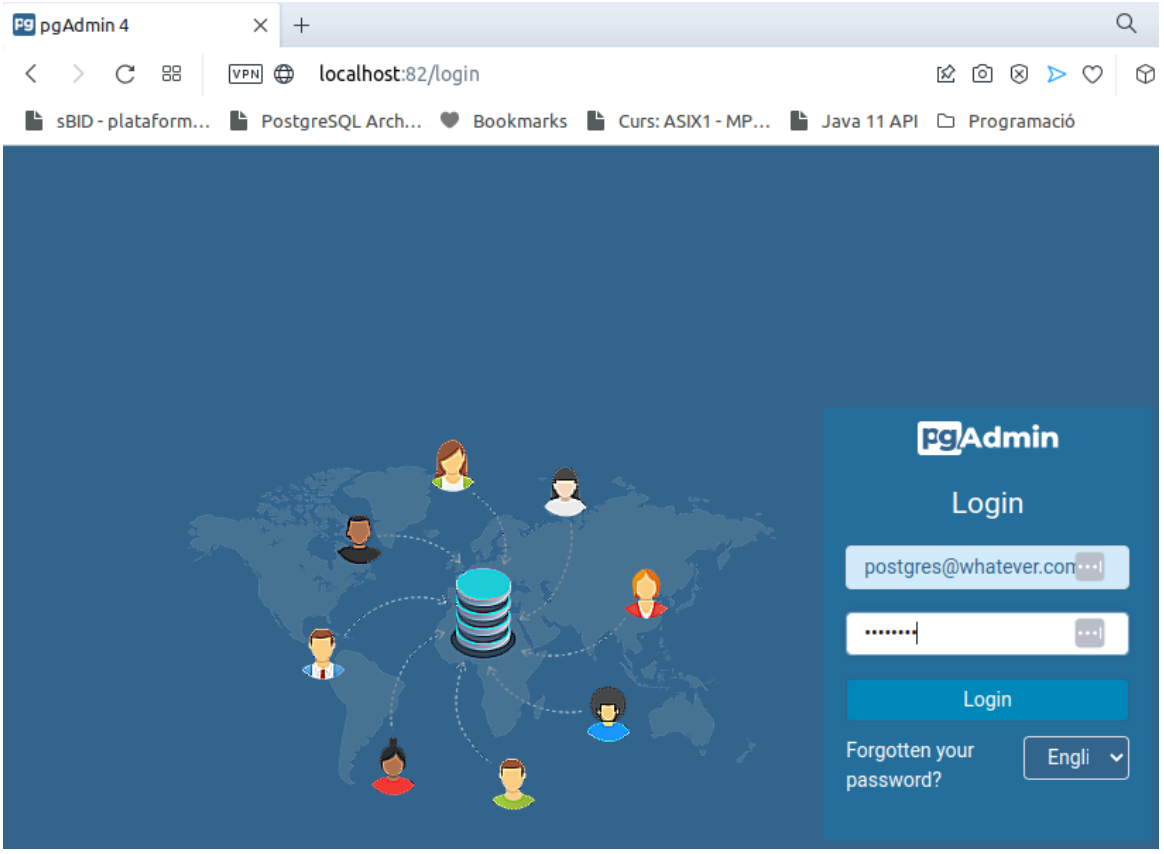
```
profe@profe-ventola ~$ docker run --name my-pgadmin -p 82:80 -e 'PGADMIN_DEFAULT_EMAIL=postgres@whatev
er.com' -e 'PGADMIN_DEFAULT_PASSWORD=postgres' -d dpage/pgadmin4
```

Hem creat un contenedor anomenat ***my-pgadmin*** que escolta pel port ***82***. Les credencials usuari/contrasenya per validar-se són ***postgres@whatever.com/postgres***. Com sempre, iniciem el contenedor[13]:

```
profe@profe-ventola ~$ docker start my-pgadmin

my-pgadmin
```

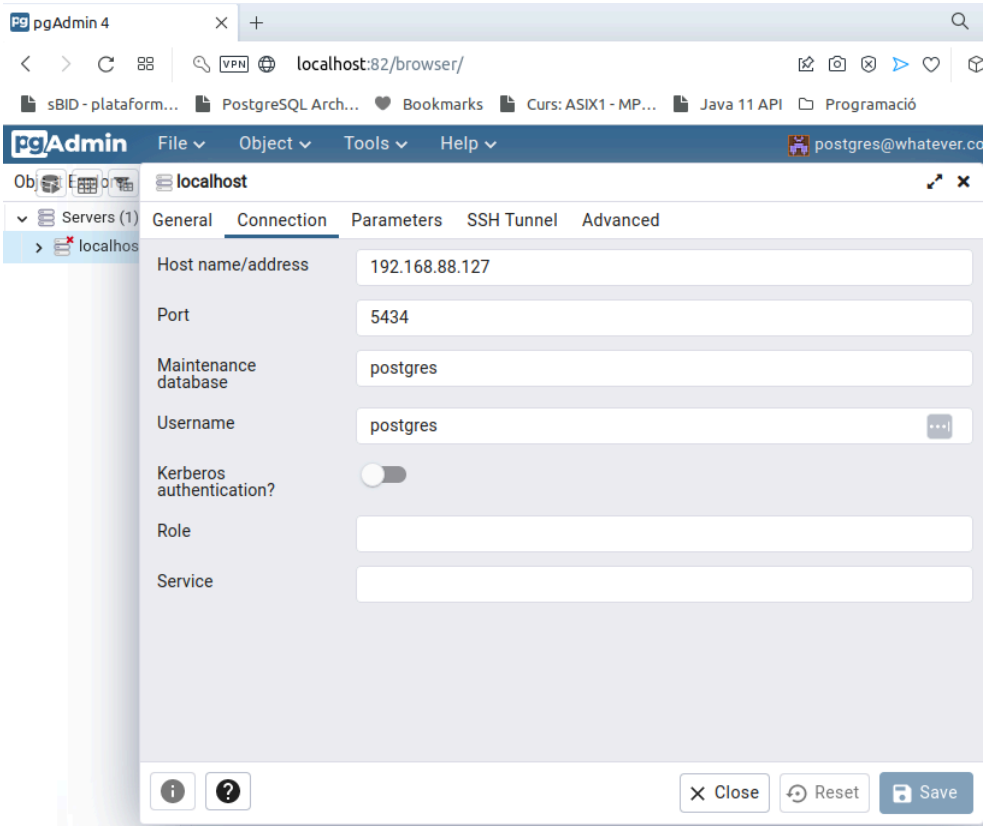
Si accedim des d'un navegador a l'adreça ***localhost:82*** i posem les credencials entrarem a l'entorn web d'administració:



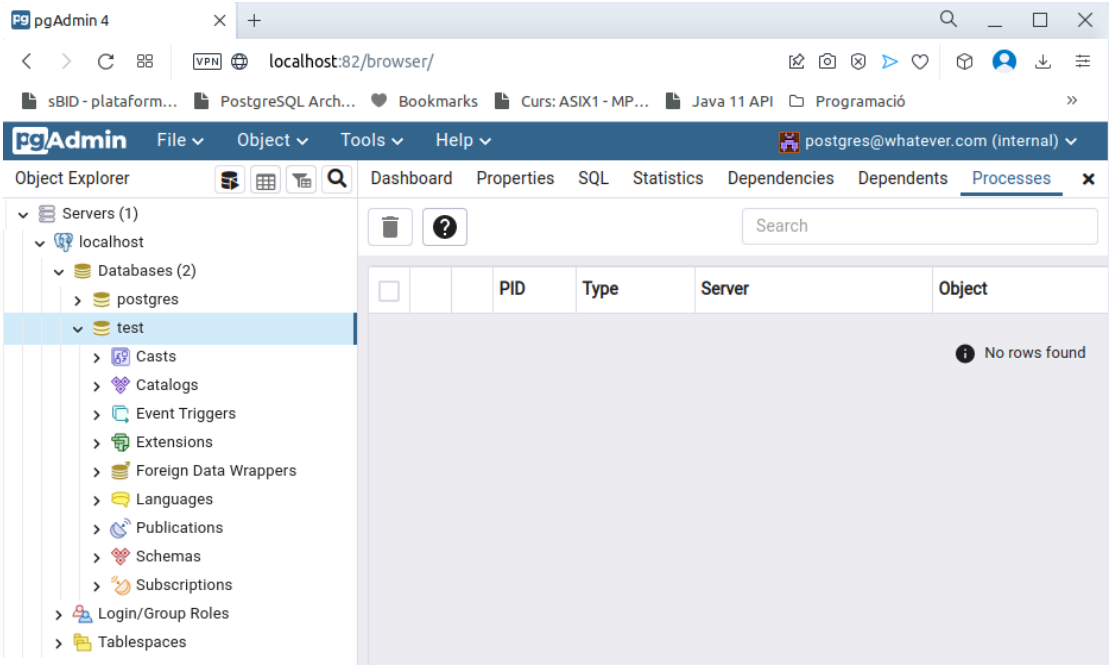
[12] docker run --name my-pgadmin -p 82:80 -e 'PGADMIN_DEFAULT_EMAIL=postgres@whatever.com' -e 'PGADMIN_DEFAULT_PASSWORD=postgres' -d dpage/pgadmin4

[13] És important arrancar el contenedor del pgadmin després que el del PostgreSQL i no abans, sinó pot donar problemes a l'hora de validar-mos

Per connectar en el servidor PostgreSQL del contenedor, a les propietats del localhost, pestanya **Connection** haurem de canviar el port que ve per defecte (***5432***), pel que hem definit inicialment al crear el contenedor (***5434***), i posar la IP privada del nostre ordenador (***192.168.88.127*** a l'exemple):



Apretem el botó **Save**, posem la contrasenya ***postgres*** si la demana, i entrem dins l'entorn d'administració:



Webs consultades:

https://hub.docker.com/r/dpage/pgadmin4/

https://migueldoctor.medium.com/how-to-run-postgresql-pgadmin-in-3-steps-using-docker-d6fe06e47ca1