1ero instale docker desde digital ocean, la version community

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker version
Client: Docker Engine - Community
 Version:           20.10.17
 API version:       1.41
 Go version:        go1.17.11
 Git commit:        100c701
 Built:             Mon Jun  6 23:02:57 2022
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true

Server: Docker Engine - Community
 Engine:
  Version:          20.10.17
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.17.11
  Git commit:       a89b842
  Built:            Mon Jun  6 23:01:03 2022
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.7
  GitCommit:        0197261a30bf81f1ee8e6a4dd2dea0ef95d67ccb
 runc:
  Version:          1.1.3
  GitCommit:        v1.1.3-0-g6724737
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

2 Registro en docker hub

3-Obtener la imagen de busy box: hago un pull de la imagen de docker

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
50783e0dfb64: Pull complete
Digest: sha256:ef320ff10026a50cf5f0213d35537ce0041ac1d96e9b7800bafd8bc9eff6c693
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$
```

- Verificar qué versión y tamaño tiene la imagen bajada, obtener una lista de imágenes locales:
- Explicar porque no se obtuvo ningún resultado: Porque no le estoy corriendo ningún comando, el docker run crea una capa de contenedor que se graba en la imagen especificada.
- Especificamos algún comando a correr dentro del contendor, ejecutar por ejemplo: el echo es como un print

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker run busybox echo "Hola Mundo"
Hola Mundo
```

- Ver los contendores ejecutados utilizando el comando ps:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

este comando sirve para enumerar los contenedores en ejecucion de forma predeterminada, pero se pueden usar diferentes parametros para obtener una lista de otros contenedores.

el ps significa Estado de proceso, el comando es docer ps [OPTIONS]

- Vemos que no existe nada en ejecución, correr entonces:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND               CREATED         STATUS                     PORTS     N
AMES
706185783c55   busybox   "echo 'Hola Mundo'"   3 minutes ago   Exited (0) 3 minutes ago             p
eaceful_grothendieck
05a8903f3a7f   busybox   "sh"                  9 minutes ago   Exited (0) 8 minutes ago             f
riendly_robinson
```

- Mostrar el resultado y explicar que se obtuvo como salida del comando anterior.

si corremos docker ps –help me tira todos las flags o parametros, y el -a me tira todos los contenedores que estan corriendo

5- Ejecutando en modo interactivo

- Ejecutar el siguiente comando

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker run -it busybox sh
/ #
```

- Para cada uno de los siguientes comandos dentro de contenedor, mostrar los resultados:

  ps

```
PID   USER     TIME  COMMAND
    1 root      0:00 sh
    7 root      0:00 ps
```

```
/ # uptime
 17:50:50 up 18:23,  0 users,  load average: 2.59, 2.01, 1.34
/ #
```

```
/ # free
              total        used        free      shared  buff/cache   available
Mem:       16151344    11642868      187292     1112360     4321184     3054852
Swap:       2097148      132328     1964820
/ #
```

```
/ # ls -l /
total 36
drwxr-xr-x    2 root     root         12288 Jul 29 01:32 bin
drwxr-xr-x    5 root     root           360 Aug 18 17:49 dev
drwxr-xr-x    1 root     root          4096 Aug 18 17:49 etc
drwxr-xr-x    2 nobody   nobody        4096 Jul 29 01:32 home
dr-xr-xr-x  582 root     root             0 Aug 18 17:49 proc
drwx------    1 root     root          4096 Aug 18 17:50 root
dr-xr-xr-x   13 root     root             0 Aug 18 17:49 sys
drwxrwxrwt    2 root     root          4096 Jul 29 01:32 tmp
drwxr-xr-x    3 root     root          4096 Jul 29 01:32 usr
drwxr-xr-x    4 root     root          4096 Jul 29 01:32 var
```

- Salimos del contendor con:

6- Borrando contendores terminados

- Obtener la lista de contendores

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND               CREATED          STATUS                      PORTS
 NAMES
2ca63625bee4   busybox   "sh"                  3 minutes ago    Exited (0) 38 seconds ago
 cranky_faraday
706185783c55   busybox   "echo 'Hola Mundo'"   8 minutes ago    Exited (0) 8 minutes ago
 peaceful_grothendieck
05a8903f3a7f   busybox   "sh"                  14 minutes ago   Exited (0) 14 minutes ago
 friendly_robinson
```

- Para borrar podemos utilizar el id o el nombre (autogenerado si no se especifica) de contendor que se desee, por ejemplo:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker rm friendly_robinson
friendly_robinson
```

rm= remove

- Para borrar todos los contendores que no estén corriendo, ejecutar cualquiera de los siguientes comandos:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker rm $(docker ps -a -q -f status=exited)
2ca63625bee4
706185783c55
```

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] Y
Total reclaimed space: 0B
```

7- Montando volúmenes

Hasta este punto los contenedores ejecutados no tenían contacto con el exterior, ellos corrían en su propio entorno hasta que terminaran su ejecución. Ahora veremos cómo montar un volumen dentro del contenedor para visualizar por ejemplo archivos del sistema huésped:

- Ejecutar el siguiente comando, cambiar myusuario por el usuario que corresponda. En linux/Mac puede utilizarse /home/miusuario):

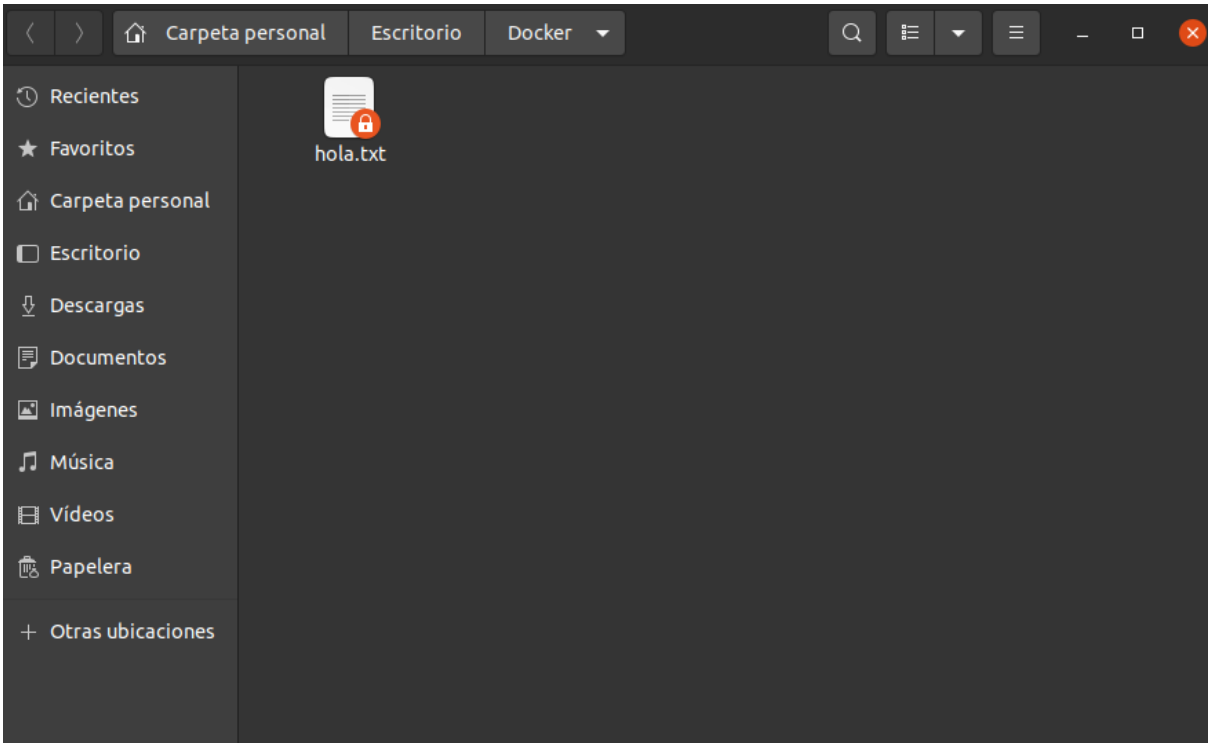
```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker run -it -v ~/Escritorio/Docker/:/var/escritorio
busybox /bin/sh
/ #
```

```
ls: /Escritorio: No such file or directory
/ # ls -l /var/escritorio
total 0
```

Creo el archivo en la ruta establecida arriba

```
# touch /var/escritorio/hola.txt
```

- Verificar que el Archivo se ha creado en el escritorio o en el directorio home según corresponda.



8- Publicando puertos

En el caso de aplicaciones web o base de datos donde se interactúa con estas aplicaciones a través de un puerto al cual hay que acceder, estos puertos están visibles solo dentro del contenedor. Si queremos acceder desde el exterior deberemos exponerlos.

- Ejecutar la siguiente imagen, en este caso utilizamos la bandera -d (detach) para que nos devuelva el control de la consola:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker run -d daviey/nyan-cat-web
Unable to find image 'daviey/nyan-cat-web:latest' locally
latest: Pulling from daviey/nyan-cat-web
b7f33cc0b48e: Pull complete
5f9b58fd6dd4: Pull complete
1adeef8edfca: Pull complete
cc8a2986b124: Pull complete
7220539c61d6: Pull complete
Digest: sha256:57ac8fd383ada137e22a2894e92f74287f4566be0ae21ca97828b34a93a646c6
Status: Downloaded newer image for daviey/nyan-cat-web:latest
a7cac1287adcb911b9d7ca0b345f478da3834c6d76e024d74be3c30587bc8d1d
```

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$  docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED          STATUS          PORTS
 NAMES
a7cac1287adc   daviey/nyan-cat-web   "nginx -g 'daemon of…"   20 seconds ago   Up 20 seconds   80/tcp, 443/tcp
 hardcore_proskuriakova
```

- Vemos que el contendor expone 2 puertos el 80 y el 443, pero si intentamos en un navegador acceder a http://localhost no sucede nada.
- Procedemos entonces a parar y remover este contenedor:

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker kill hardcore_proskuriakova
hardcore_proskuriakova
```

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker rm hardcore_proskuriakova
hardcore_proskuriakova
```

- Vamos a volver a correrlo otra vez, pero publicando uno de los puertos solamente, el este caso el 80

Como ya esta en uso el puerto 80 debo usar otro

```
docker: Error response from daemon: driver failed programming external connectivity on endpoint sharp_clarke (85
3e4e2e751cf1738166f4f6469baf70e4b219a9f7d95619e6b561bbb16140e7): Error starting userland proxy: listen tcp4 0.0.
0.0:80: bind: address already in use.
```

lo cambio por el 81

```
uan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ docker run -d -p 81:80 daviey/nyan-cat-web
05d48296ad3d52b8ee5e952af13d5c90e09d4e20d93d1aaa8183226c954b12a
```

9- Utilizando una base de datos

- Levantar una base de datos PostgreSQL

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ sudo docker run --name my-postgres -e POSTGRES_PASSWORD=mysecretpa
ssword -v $HOME/.postgres:/var/lib/postgresql/data -p 5432:5432 -d postgres:9.4
[sudo] contraseña para juan-pablo:
Unable to find image 'postgres:9.4' locally
9.4: Pulling from library/postgres
619014d83c02: Pull complete
7ec0fe6664f6: Pull complete
9ca7ba8f7764: Pull complete
9e1155d037e2: Pull complete
febcfb7f8870: Pull complete
8c78c79412b5: Pull complete
5a35744405c5: Pull complete
27717922e067: Pull complete
36f0c5255550: Pull complete
dbf0a396f422: Pull complete
ec4c06ea33e5: Pull complete
e8dd33eba6d1: Pull complete
51c81b3b2c20: Pull complete
2a03dd76f5d7: Pull complete
Digest: sha256:42a7a6a647a602efa9592edd1f56359800d079b93fa52c5d92244c58ac4a2ab9
Status: Downloaded newer image for postgres:9.4
46c7501855e82a171e22704edc51b587cf6d810d36174e5c43bfd0e415e7e8ae
```

- Ejecutar sentencias utilizando esta instancia

```
juan-pablo@juanpablo-OMEN-Laptop-15-ek0xxx:~$ sudo docker exec -it my-postgres /bin/bash
root@46c7501855e8:/# psql -h localhost -U postgres
psql (9.4.26)
Type "help" for help.

postgres=#
```

```
postgres=# \l
                                List of databases
   Name    |  Owner   | Encoding |  Collate   |   Ctype    |   Access privileges
-----------+----------+----------+------------+------------+-----------------------
 postgres  | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
 template0 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
(3 rows)

postgres=#
```

```
postgres=# create database test;
CREATE DATABASE
```

```
postgres=# \connect test
You are now connected to database "test" as user "postgres".
test=#
```

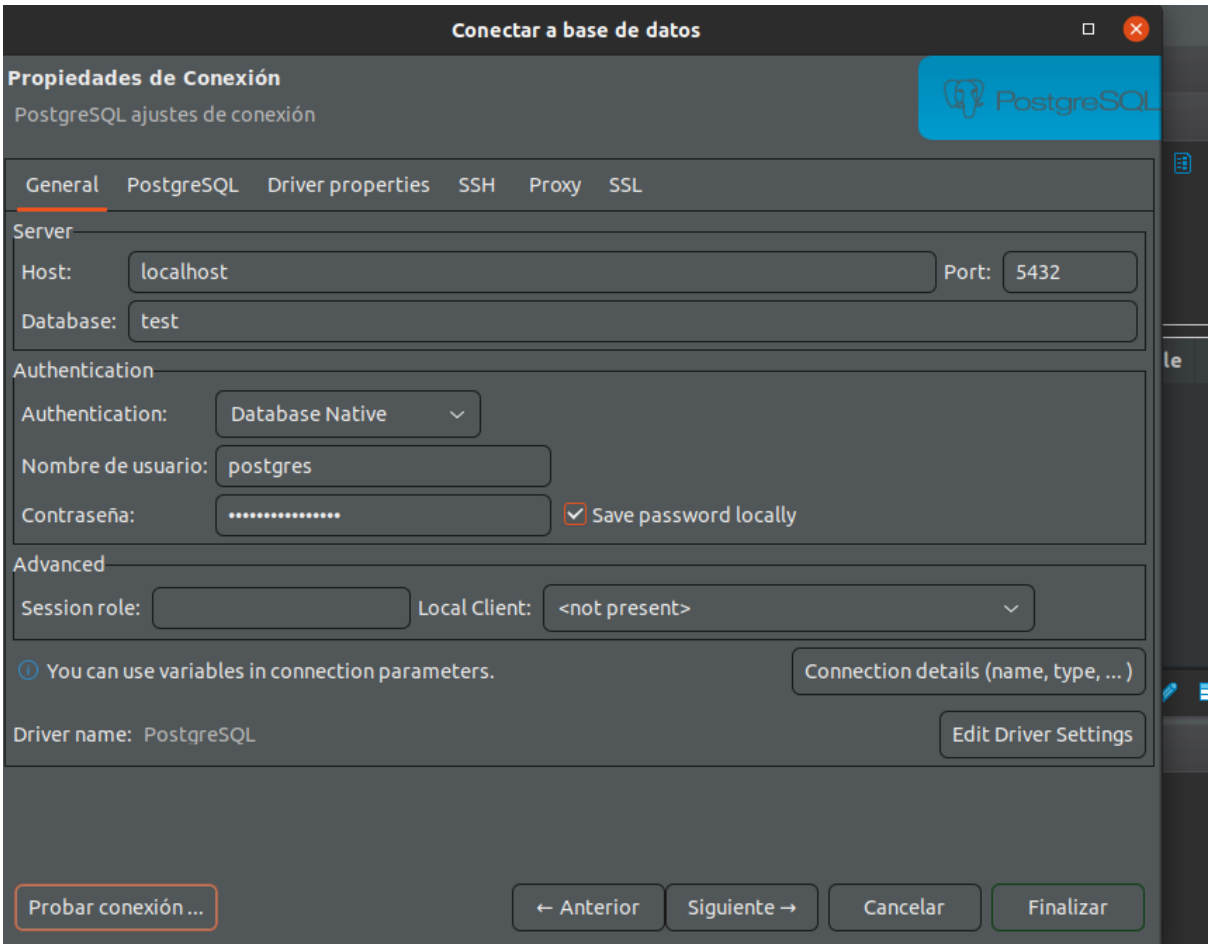
```
test=# create table tabla_a (mensaje varchar(50));
CREATE TABLE
```

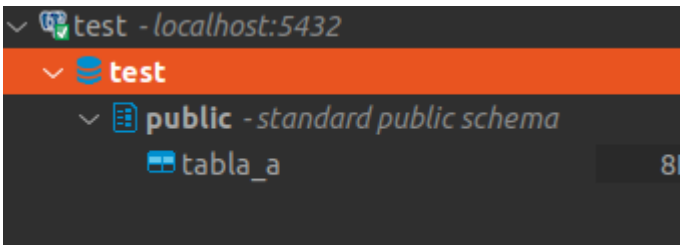
```
test=# insert into tabla_a (mensaje) values('Hola mundo!');
INSERT 0 1
```

```
test=# select * from tabla_a;
   mensaje
-------------
 Hola mundo!
(1 row)
```

```
test=# \q
root@46c7501855e8:/#
```

- Conectarse a la base utilizando alguna IDE (Dbeaver - https://dbeaver.io/, eclipse, IntelliJ, etc...). Interactuar con los objectos objectos creados.





- Explicar que se logro con el comando `docker run` y `docker exec` ejecutados en este ejercicio.

Docker run: con este comando lo que hace es generar una instancia de una imagen, que una vez creada, se puede parar o volver a iniciar, esa instancia queda guardada en nuestro sistema, sirve entonces para inicializar un contenedor de docker usando una imagen

Docker exec: sirve para un contenedor que ya esta corriendo, ejecutar comandos