COMANDOS PARA DOCKER (VIRTUAL BOX)

CREAR UN USUARIO NUEVO

--> sudo useradd -m nombre <-- (desde usuario admin)

Siendo m el tag que crea /home/nombre y "nombre" el nombre del nuevo usuario que estamos creando

---

CREAR CONTRASEÑA PARA EL USUARIO

--> sudo passwd nombre <-- (desde usuario admin)

AÑADIR USUARIO A GRUPO DOCKER (desde usuario admin)

--> sudo usermod -aG docker "nombre usuario" <-- || "nombre usuario" se debe sustituir por el nombre del usuario que queremos añadir al grupo

--> newgrp docker <-- || Aplica cambios

AÑADIR USUARIO A GRUPO SUDOERS (desde usuario admin)

--> sudo visudo <-- || entra en un menú donde buscamos "root ALL=(ALL:ALL) ALL" y añadimos a mano nuestro usuario "nuevoUsuario ALL=(ALL:ALL) ALL"

COMPROBAR QUE HA FUNCIONADO (desde nuevo usuario)

--> sudo su <-- || Solicitará la contraseña y cambiará el prompt de $ a #. Lo que indicará que accedemos como superusuario.

BORRAR UN USUARIO (desde cualquier usuario)

--> userdel -r -f "nombre usuario" <--

CREAR UN CONTENEDOR (MOTOR MARIADB) CON VARIABLES DE ENTORNO Y VOLUMEN (desde cualquier usuario)

docker run -d -it -v /home/nuevoUsuario/copiaMariaDB:/var/lib/mysql --name BD_mariadb -p 3336:3306 -e MARIADB_USER=admin -e MARIADB_PASSWORD=admin -e MARIADB_ROOT_PASSWORD=root mariadb

/home/nuevoUsuario/copiaMariaDB: --> dirección nuestra (host) y nombre del archivo que estamos creando

:/var/lib/mysql --> dirección proporcionada por mariadb en Docker Hub (contenedor)

/home/nuevoUsuario/copiaMariaDB:/var/lib/mysql --> Bind Mount directo entre el host y el contenedor

-p 3336:3306 --> puertos para mariadb

-p 8082:80 --> puertos para Apache (http) para ver en el navegador

-e MARIADB_USER=admin --> variable de entorno, crea un usuario llamado "root" (modificable)

-e MARIADB_PASSWORD=admin --> variable de entorno, crea una contraseña para el usuario anterior

-e MARIADB_ROOT_PASSWORD=root --> variable de entorno, crea una contraseña para un usuario normal

mariadb --> nombre de la imagen

PONER EN MARCHA EL CONTENEDOR

--> docker start "nombre contenedor" <--

--> docker ps -a <-- || listado de todos los contenedores. Aquí se puede verificar si está activo (STATUS up) o no (STATUS Exited)

--> docker ps <-- || muestra solo los activos

ENTRAR EN EL CONTENEDOR

--> docker exec -it <nombre contenedor o ID> /bin/bash <-- (para usuario root)

exec --> ejecuta un comando en un contenedor que ya está funcionando

/bin/bash --> bash es el intérprete de comandos (Bash es el Shell, permite interactuar con el entorno del contenedor usando los comandos Bash)

(ES POSIBLE QUE LA IMAGEN NO TENGA BASH, SE PUEDE USAR SH)

ENTRAR EN LA BASE DE DATOS CON VARIABLES DE ENTORNO (debemos estar dentro del contenedor)

--> mariadb -u root-p <-- (para usuario root, la contraseña por defecto es root. Entramos aquí para dar permisos a otro usuario DENTRO de la base de datos)

-->**GRANT ALL PRIVILEGES ON *.* TO 'usuario_env'@'%' WITH GRANT OPTION;** <-- (DA PERMISOS PARA TODO)

--> **exit;** <--

--> mariadb -u "nombre usuario" -p <-- (para usuario que no sea root, lo declaramos en la variable de entorno ”admin”)

-u --> acceso con usuario MARIADB_USER=admin // root

-p --> solicita la contraseña MARIADB_PASSWORD=admin // root

EJECUTAR COMANDOS DENTRO DE LA BASE DE DATOS

--> use “nombre_base_datos” ; <-- || “;” necesario, si no se introduce se queda esperándolo en las siguientes líneas)

--> create database “nombre_bdatos”; <-- || crea una nueva base de datos

--> show databases; <--

--> show tables; <--

--> **select * from “nombre_tabla”;** <-- || muestra todos los registros de la tabla seleccionada

---

EJECUTAR COMANDOS DENTRO DEL CONTENEDOR (DESDE LINUX)

--> docker exec <nombre contenedor o ID> mkdir usr/nombre <-- || se crea un directorio que se ubica en la raíz del sistema dentro de docker

--> docker exec <nombre contenedor o ID> touch usr/nombre <--

--> docker exec <nombre contenedor o ID> ls /usr/local/apache2/htdocs <--

--> docker exec -it <nombre contenedor o ID> sh -c "echo 'hola' > /usr/local/apache2/htdocs/index.html" <--

mkdir --> crea directorios/carpetas

touch --> crea archivos

ls --> lista el contenido del directorio pasado en la ruta (LINUX)

/usr/local/apache2/htdocs --> directorio raíz donde se almacenan archivos HTML

sh -c "echo 'hola' > /usr/local/apache2/htdocs/index.html" --> escribe la cadena 'hola' en un archivo llamado index.html en el directorio htdocs

---

CONCEDER PRIVILEGIOS DENTRO DE LA BASE DE DATOS

--> GRANT ALL PRIVILEGES ON *.* TO 'nombre_usuario' <-- || Da todos los permisos a todos los objetos

---

CONOCER TODOS LOS PUERTOS ACTIVOS

--> docker exec -it "nombre_del_contenedor" netstat -a <-- || muestra todos los puertos que tienen conexión y escucha

--> docker exec -it "nombre_del_contenedor" netstat -at <-- || muestra solo conexiones TCP

--> docker exec -it "nombre_del_contenedor" netstat -l <-- || muestra solo los puertos que están en escucha

--> docker exec -it "nombre_del_contenedor" netstat -an <-- || muestra información detallada con direcciones IP y numeros de puerto en lugar de nombres

--> docker exec -it "nombre_del_contenedor" netstat -ap <-- || muestra todos los programas y puertos asociados LINUX

INSTALAR NETSTAT/REPOSITORIOS (DEBIAN/UBUNTU)

--> docker exec -it "nombre_del_contenedor" apt-get update <--

--> docker exec -it "nombre_del_contenedor" apt-get install -y net-tools <--

apt-get update --> Actualiza los repositorios del sistema

net-tools --> Instala el paquete que incluye netstat y otros repositorios

---

MANEJO REMOTO (servidor SSH)

--> docker exec -it "nombre_del_contenedor" apt-get update <--

--> docker exec -it "nombre_del_contenedor" apt install -y openssh-server <--

Puerto por defecto --> 22

openssh-server --> instala el paquete openssh-server, proporciona el servidor SSH

-y --> responde automaticamente SI a cualquier pregunta que pueda surgir

DETENER SERVICIO SSH

--> sudo service ssh stop <-- || Cierra las conexiones SSH existentes

(detener el servicio SSH significa apagar el demonio SSH que maneja las conexiones seguras por shell)

CREAR UNA NUEVA IMAGEN A PARTIR DE UN CONTENEDOR

--> docker commit "nombre_del_contenedor" "nombre_de_la_nueva_imagen" <--

--> docker run "nombre_de_la_nueva_imagen" <-- || Crea un contenedor con la nueva imagen

commit --> crea una nueva imagen a partir de un contenedor conservando la configuración realizada en su interior. La nueva imagen se crea en local, generalmente en "/var/lib/docker", dentro de la carpeta "image" (PUEDE VARIAR)

MOVERSE DENTRO DE LINUX

cd ruta -> abrir carpeta que se pasa como parámetro (cd Escritorio)

cd .. -> va a la carpeta anterior a la que nos encontramos

cd ../carpeta -> abre una carpeta que está dentro de la carpeta padre (la anterior a la que nos encontramos)

pwd -> muestra donde nos encontramos

ls -> lista el contenido de la carpeta en la que estamos

ls ruta -> muestra el contenido de la carpeta que se pasa como parámetro (ls Escritorio)

ls -la ruta -> muestra TODO el contenido de la carpeta en forma de lista detallada. (ruta opcional, si no se pasa como parámetro toma la ruta actual)

cp archivo_origen archivo_destino -> copia archivo origen en el destino y si no existe se crea.

nano “nombre_archivo” -> abre el editor de texto de la terminal, si no existe lo crea.

cat “nombre archivo” -> muestra las primeras líneas del contenido del archivo.