Como crear un contenedor con Docker-Mysql y persistir la información

platzi.com/tutoriales/1432-docker/3268-como-crear-un-contenedor-con-docker-mysql-y-persistir-la-informacion/

Te han contado maravillas de Docker y no sabes por dónde empezar; en esta oacación vamos a crear un contenedor que contenga Mysql y persistir los datos, para que a la hora de detener el proceso del contenedor nos quedemos con la información generada.

Obtener Docker

No voy a decir que es Docker si hay muchos sitios que lo hacen mejor de lo que yo lo haria, el Get Started de documentación oficial seria un buen punto de partida.

Si aún no lo tienes solo descárgalo desde el sitio oficial Docker y no hay que configurar nada adicional(no en modo noob dev)

Para verificar la instalacion lo poedes hacer desde la terminal

```
$ docker
Docker version18.09.0, build 4d60db4
```

Descargamos una imagen

Lo primero que hay que hacer es descargar el contenedor de Mysql con el siguiente comando.

```
$ docker run -d -p 33060:3306 --name mysql-db -e MYSQL_ROOT_PASSWORD=secret mysql
```

- **-d**: Deatached Mode es la forma en que indicamos que corra en background.
- **-p** : puerto, el contenedor corre en el puerto 3306 pero hacemos un bind para que lo escuchemos en Host el puerto 33061.
- **–name** : para no tener que hacer referencia al hash le asignamos un nombre.
- **-e** : environment le asignamos la contraseña.

Con esta tenemos nuestro contenedor escuchando

```
CONTAINER ID        IMAGE               CREATED             STATUS              PORTS                               NAMES
b62caa4104ed        mysql               5 minutes ago       Up 5 minutes        33060/tcp, 0.0.0.0:33060->3306/tcp  mysql-db
```

Entrar al contenedor

Para entrar al contenedor usamos un modo interactivo para asignar un TTY(terminal) y un STDIN abierto

```
$ docker exec -it mysql-db mysql -p
```

Donde:

- **exec**: indicamos que vamos a pasar un comando.
- **-it** Modo interactivo.
- **mysql -p**: es el comando para entrar a la consola de mysql con el usuario root(si has trabajado con mysql en consola es lo mismo).

Una vez que se ejecuta la línea nos pedirá la contraseña que definimos en MYSQL_ROOT_PASSWORD y estamos dentro del contenedor y podemos lanzar comandos a MYSQl.

En la parte inferior podemos ver un ejemplo de cómo creamos una base de datos.

```
$  docker exec -it mysql-db mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 8.0.13 MySQL Community Server - GPL

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database demo;
Query OK, 1 row affected (0.26 sec)

mysql> show databases;
+--------------------+| Database           |
+--------------------+
| demo               |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

Conectar el Host al contenedor

Cuando levantamos el contenedor creamos una interfaz al puerto 33060:3306 donde el contenedor utiliza el puerto 3306 pero en el host 33060

De esta forma podemos conectar el host mediante el Workbench.

Montar un volumen

Hasta este punto no persistimos los datos que se realicen en nuestro contenedor lo que significa que cuando terminamos con el proceso los cambios se perderán, para eso Docker nos dice que hay que utilizar volúmenes que no es otra cosa que una parte del disco Host se reserve para los datos generados en el contenedor(no el contenedor).

Para eso seguimos los siguientes pasos:

1. Eliminamos el proceso que corre el contenedor creado.

   ```
   $ docker rm -f mysql-db
   ```

2. Eliminamos todos los volúmenes ya que Docker crea volúmenes temporales sin pedirte permiso.

   ```
   $ docker volume prune
   ```

3. Creamos un volumen

   ```
   $ docker volumecreate mysql-db-data
   ```

4. Verificamos que se haya creado el volumen

   ```
   $ docker volume ls
   DRIVER              VOLUME NAME
   local               mysql-db-data
   ```

5. Levantamos nuevamente el Docker y agregamos el volumen con la opcion --mount

   ```
   $  docker run -d -p 33060:3306 --name mysql-db  -e MYSQL_ROOT_PASSWORD=secret --mount
   src=mysql-db-data,dst=/var/lib/mysql mysql
   ```

6. Entramos al contenedor de forma interactiva o desde el Workbench y creamos una base de datos

```
$ docker exec -it mysql-db mysql -p
...
mysql> create database demo;
Query OK, 1 row affected (0.32 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| demo               || information_schema || mysql              || performance_schema || sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

7. Terminamos el proceso tal como en el paso 1

```
$ docker rm -f mysql-db
```

8. Lanzamos nuevamente el proceso como en el paso 5

```
$  docker run -d -p 33060:3306 --name mysql-db  -e MYSQL_ROOT_PASSWORD=secret --mount
src=mysql-db-data,dst=/var/lib/mysql mysql
```

9. Entramos nuevamente al contenedor de forma interactiva y podemos ver que la base de datos que creamos se encuentra

```
$ docker exec -it mysql-db mysql -p
...
mysql> create database demo;
Query OK, 1 row affected (0.32 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| demo               || information_schema || mysql              || performance_schema || sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

Y de esta forma ya estamos trabajando con volúmenes donde persistimos los datos el el Host de forma que si queremos utilizar la base de datos solo hay que montar el volumen.

Conclusión

Y de esta forma ya estamos trabajando con contenedores, como pudieron ver se instaló el Workbench para hacer una conexión vía HTTP y así como lo hacemos al 127.0.0.1:33060 lo podemos hacer a un contenedor remoto.