
MySQL®
docker

Introducción

Esta sección de la documentación estará enfocada específicamente a la dockerización del R-Studio.

Se mostrará una guía paso a paso de como se ejecutó el contenedor de MySQL para el parcial # 1 de Data Product.

Para lograr esto, nos basamos en la imagen oficial de MySQL encontrada en Docker hub.

Procedimiento

Jalar la imagen oficial

El primer paso es jalar la imagen oficial de MySQL desde Docker hub con el siguiente comando:

Docker pull mysql:latest

```
root@pop-os:/home/ricardo# docker pull mysql:latest
latest: Pulling from library/mysql
Digest: sha256:c358e72e100ab493a0304bda35e6f239db2ec8c9bb836d8a427ac34307d074e
d
Status: Image is up to date for mysql:latest
docker.io/library/mysql:latest
```

En mi caso ya estaba descargada, pero si no fuese así, habría que esperar a que se descarge y tomaría unos minutos.

Probar la imagen

Ahora que la imagen fue descargada correctamente, toca probarla. Para ello se utiliza el siguiente comando:

Docker run –name [nombre_libre] -e MYSQL_ROOT_PASSWORD=[Su_contraseña] -d mysql:latest

```
ricardo@pop-os:~$ sudo docker run --name dbtest -e MYSQL_ROOT_PASSWORD=pass -d
 mysql:latest
[sudo] password for ricardo:
04d9ac3c33a88380fffe8f5390043187af4d00bd6b94e28c4e7b37bb0eaed408
```

Nos devuelve un hash, lo que quiere decir que está corriendo correctamente. De igual forma lo podemos probar de la siguiente forma:

Docker exec -it [Nombre que le pusimos] “bash”

```
root@pop-os:/home/ricardo# docker exec -it dbtest "bash"
root@04d9ac3c33a8:/# ls
bin   docker-entrypoint-initdb.d  home   media  proc  sbin  tmp
boot  entrypoint.sh               lib    mnt    root  srv   usr
dev   etc                         lib64  opt    run   sys   var
root@04d9ac3c33a8:/#
```

Podemos ver que está corriendo correctamente pues, si ponemos “ls”, nos devuelve los directorios del contenedor

Armar nuestra propia imagen

Ahora que ya instalamos y ejecutamos correctamente la imagen oficial de MySQL, nos toca armar nuestra propia imagen por diversas razones:

1. Para indicarle que jale automaticamente los datos .csv que se utilizarán para poblar las tablas.
2. Para poder especificar nuestras credenciales en el dockerfile.
3. Para poder crear una conexión entre ambos contenedores con más facilidad.

Como hacerlo

Lo primero que hay que hacer, es crear una carpeta con un nombre de nuestra preferencia. En mi caso la carpeta se llamará “SQL”.

Dentro de esa carpeta, tenemos que hacer otra carpeta que se llame “data”, y dentro de ella ubicar los tres archivos .csv.

De vuelta en la carpeta “SQL”, tenemos que hacer dos archivos. Se pueden hacer con cualquier editor de texto, pero yo recomiendo Visual Studio Code pued tiene plug ins que nos pueden ayudar en ambos scripts. Esos scripts serán:

1. Dockerfile
2. Load_data.sql

Dockerfile

Este archivo nos sirve para darle instrucciones al contenedor de que acciones ejecutar al ser armado.

```
FROM mysql:latest

# Crea un directorio específico para el parcial
# En el estará la data
RUN mkdir -p /home/parcial/
WORKDIR /home/parcial

# Se definen las credenciales
ENV MYSQL_ROOT_PASSWORD pass
ENV MYSQL_DATABASE parcial1_dp
ENV MYSQL_USER ricardo
ENV MYSQL_PASSWORD pass

# Copia los tres archivos .csv desde "data", al directorio
# que creamos anteriormente
COPY data/ /home/parcial

# Añade nuestro script load_data.sql al entrypoint
```

```
# El entrypoint es lo que se corre de primero al llamar el contenedor
# En el caso de MySQL, este es el directorio default
ADD load_data.sql /docker-entrypoint-initdb.d

# Puerto
EXPOSE 3306
```

Load_data.sql

Este archivo es el que irá dentro del entrypoint del contenedor, es decir, lo que se correrá de primero.

Se pone ahí pues de esta forma las tablas serán pobladas automáticamente, en vez de nosotros tener que poblarlas cada vez que levantamos el contenedor.

```
USE parcial1_dp;

CREATE TABLE video_metadata (
  video_id VARCHAR(20) DEFAULT NULL,
  title VARCHAR(140) DEFAULT NULL,
  link VARCHAR(500) DEFAULT NULL
);

CREATE TABLE video_stats (
  id varchar(20) DEFAULT NULL,
  viewCount int(11) DEFAULT NULL,
  likeCount int(11) DEFAULT NULL,
  dislikeCount int(11) DEFAULT NULL,
  favoriteCount int(11) DEFAULT NULL,
  commentCount int(11) DEFAULT NULL
);

CREATE TABLE videos (
  kind varchar(100) DEFAULT NULL,
  etag varchar(500) DEFAULT NULL,
  id text DEFAULT NULL,
  content_video_id varchar(45) DEFAULT NULL,
  date varchar(45) DEFAULT NULL
);


LOAD DATA INFILE '/home/parcial/academatica_videos_metadata.csv'
INTO TABLE video_metadata
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

```
LOAD DATA INFILE '/home/parcial/academatica_video_stats.csv'
INTO TABLE video_stats
FIELDS TERMINATED BY '\,'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;


LOAD DATA INFILE '/home/parcial/academatica_videos.csv'
INTO TABLE videos
FIELDS TERMINATED BY '\,'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

Este archivo hace varias cosas:

1. Primero crea una base de datos específica para nosotros que se llama parcial1_dp
2. Luego crea tres tablas, una por cada archivo
3. Luego de eso, va al directorio que creamos en el Dockerfile, en el que se encuentran los tres archivos .csv, y va poblando las tablas una por una. Ignora la primera fila por los títulos.

Creación del contenedor

Ahora que ya tenemos los dos scripts, podemos armar nuestro propio contenedor basándonos en ellos.

Nos dirigimos a la carpeta “SQL” en donde están los dos scripts y nuestra carpeta que contiene los tres .csv y ejecutamos el siguiente comando:

Docker build -t [Nombre libre] .

```
root@pop-os:/home/ricardo/Downloads/SQL# docker build -t mysql-parcial .
Sending build context to Docker daemon  219.6kB
Step 1/10 : FROM mysql:latest
 ---> 0d64f46acfd1
Step 2/10 : RUN mkdir -p /home/parcial/
 ---> Running in 022d7d807101
Removing intermediate container 022d7d807101
 ---> f62f35525b38
Step 3/10 : WORKDIR /home/parcial
 ---> Running in 7a74d14ffa9e
Removing intermediate container 7a74d14ffa9e
 ---> 37e5350de9d8
Step 4/10 : ENV MYSQL_ROOT_PASSWORD pass
 ---> Running in 79904d072a5b
Removing intermediate container 79904d072a5b
 ---> c62eaa9791a4
Step 5/10 : ENV MYSQL_DATABASE parcial1_dp
 ---> Running in e2c790d29f29
Removing intermediate container e2c790d29f29
 ---> c0b2ea2118cd
Step 6/10 : ENV MYSQL_USER ricardo
 ---> Running in 949f5428c533
Removing intermediate container 949f5428c533
 ---> 0bfb03fdae54
Step 7/10 : ENV MYSQL_PASSWORD pass
 ---> Running in c864f94239c5
Removing intermediate container c864f94239c5
 ---> ec8ccb3871b4
Step 8/10 : COPY data/ /home/parcial
 ---> 781a8fb3c844
Step 9/10 : ADD load_data.sql /docker-entrypoint-initdb.d
 ---> 070a1421f1ae
Step 10/10 : EXPOSE 3306
 ---> Running in 40989bac859b
Removing intermediate container 40989bac859b
 ---> 60d7c128d778
Successfully built 60d7c128d778
Successfully tagged mysql-parcial:latest
```

Podemos ver que se ejecutaron todos los pasos con éxito. Ahora solo nos queda correrlo, para ello utilizamos el siguiente comando:

docker run --name db -p 3306:3306 -d [Nombre que pusimos] --secure-file-priv=/home/parcial

Nota: Es muy importante especificar el parámetro de --secure-file-priv=/home/parcial, pues si no el contenedor no tendrá acceso a ese directorio, y no podrá poblar las tablas.

```
root@pop-os:/home/ricardo/Downloads# docker run --name dbparcial -p 3306:3306
-d mysql-parcial --secure-file-priv=/home/parcial
f2839accef16ae5bc1a1eb770cab6191be91eae38a5d2f060a8e24a3cbc4ba20
```

Efectivamente nos devuelve un hash, lo que quiere decir que está corriendo correctamente. Lo podemos probar de la siguiente forma:

Docker exec -it [Nombre que le pusimos] “bash”

```
root@pop-os:/home/ricardo/Downloads# docker exec -it dbparcial "bash"
root@f2839accef16:/home/parcial# ls
academatica_video_stats.csv  academatica_videos_metadata.csv
academatica_videos.csv
```

Efectivamente corre, y podemos ver que el directorio default es /home/parcial, justo como lo especificamos en el Dockerfile. Adicionalmente, podemos ver que los tres archivos csv se encuentran dentro del contenedor.
Pero vayamos mas allá, chequeemos que efectivamente la base de datos exista y las tablas estén pobladas:

Para ello, podemos ejecutar el siguiente comando siempre dentro del bash:

Mysql -uricardo -ppass

El -u indica el usuario, en este caso es “Ricardo” pues ese especificamos dentro del Dockerfile.

El -p es la contraseña, e igualmente en este caso es pass pues ese especificamos en el Dockerfile.

```
root@f2839accef16:/home/parcial# mysql -uricardo -ppass
mysql: [Warning] Using a password on the command line interface can be insecur
e.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.21 MySQL Community Server - GPL

Copyright (c) 2000, 2020, 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.
```

Si logramos acceder correctamente, este debería de ser el output.

Para ver si esta la base de datos, colocamos el siguiente comando:

Show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| parcial1_dp        |
+--------------------+
```

Efectivamente podemos ver que la base de datos “parcial1_dp” existe.

Para utilizarla ponemos:

Use parcial1_dp;

```
mysql> use parcial1_dp;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Y para listar las tablas dentro de la base de datos, ponemos:

Show tables;

```
mysql> show tables;
+-----------------------+
| Tables_in_parcial1_dp |
+-----------------------+
| video_metadata        |
| video_stats           |
| videos                |
+-----------------------+
3 rows in set (0.00 sec)
```

Efectivamente están las tres tablas. Podemos ver los datos de una si utilizamos un select convencional de SQL:

```
mysql> SELECT COUNT(*) FROM videos;
+----------+
| COUNT(*) |
+----------+
|      861 |
+----------+
1 row in set (0.02 sec)
```

Por ejemplo, la tabla de videos. Ahora que ya obtuvimos ese output, estamos listos para hacer la conexión entre ambos contenedores.