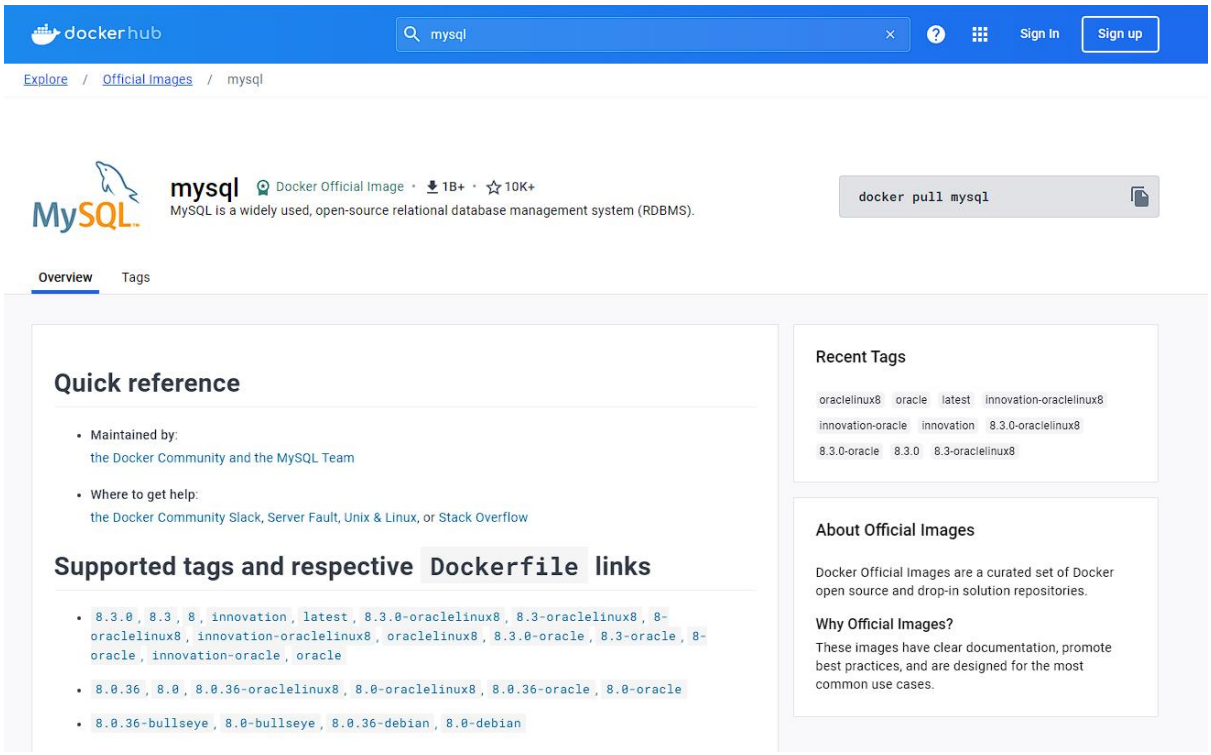
😊 Docker Volumes

1. Now go to docker hub website there you need to search for MySQL official image.
2. Here you can read about the images and you can get to know about them more.

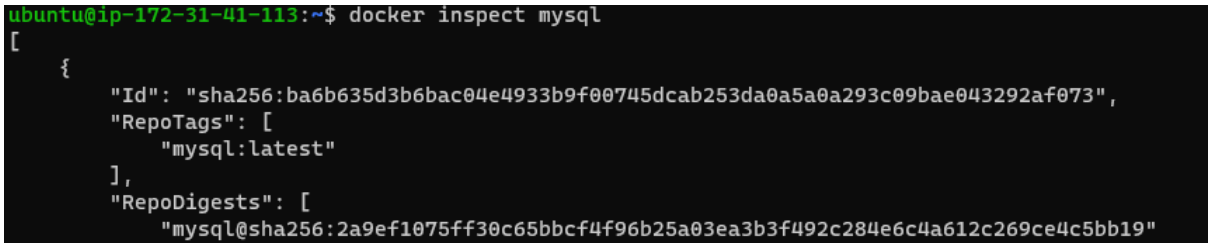


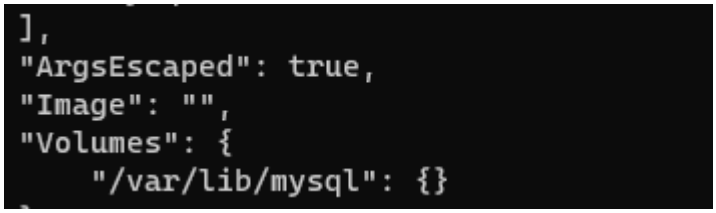
3. First you need to pull MySQL image. Here you can see we have the image.

```
ubuntu@ip-172-31-41-113:~$ docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
b8307a22608d: Pull complete
32f3542f2ba9: Pull complete
795385976c83: Pull complete
03b553142f26: Pull complete
d8ce6174541f: Pull complete
681c64901273: Pull complete
27db2241c611: Pull complete
db8e96eb91f6: Pull complete
28b560af9a2a: Pull complete
d0c9925abfdf: Pull complete
Digest: sha256:2a9ef1075ff30c65bbcf4f96b25a03ea3b3f492c284e6c4a612c269ce4c5bb19
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
ubuntu@ip-172-31-41-113:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-41-113:~$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
mysql        latest    ba6b635d3b6b   3 weeks ago   632MB
```

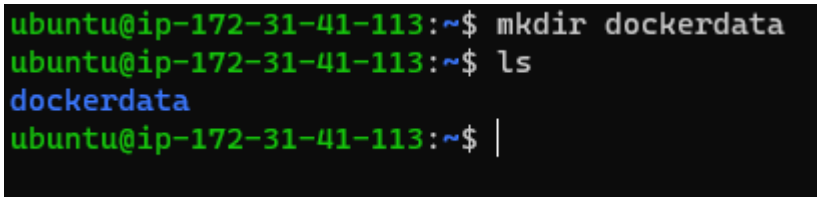
4. Now I want to know what is the volume directory, and also, I want to know what port number the process runs. So, you can run a command docker inspect and you can give the image name.
5. Once you run this command you will get all the information in JSON format. You can read the information yourself.

   docker inspect mysql



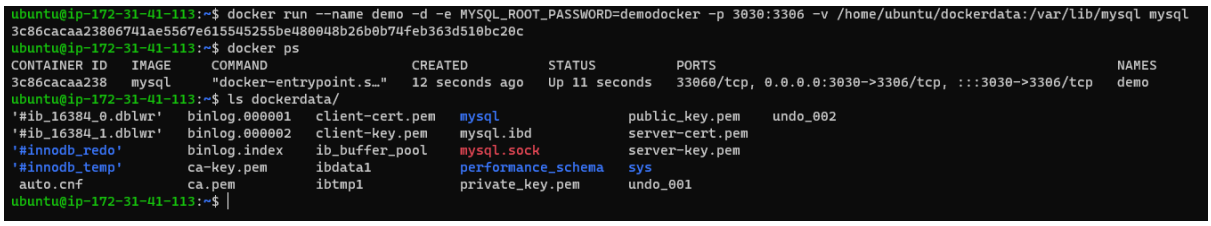


6. Now we have the details so we can run the commands based on it.
7. First you need to create a directory.



8. Now you are going to run this command. Here you can see that the container is running.

   docker run --name demo -d -e MYSQL_ROOT_PASSWORD=demodocker -p 3030:3306 -v /home/ubuntu/dockerdata:/var/lib/mysql mysql
   docker ps
   ls dockerdata/



9. You can also log in to this container. There also you will see exactly the same data.
   docker exec -it demo /bin/bash

```
ubuntu@ip-172-31-41-113:~$ docker exec -it demo /bin/bash
bash-4.4# cd /var/lib/mysql
bash-4.4# ls
'#ib_16384_0.dblwr'   binlog.000001   client-cert.pem   mysql               public_key.pem    undo_002
'#ib_16384_1.dblwr'   binlog.000002   client-key.pem    mysql.ibd           server-cert.pem
'#innodb_redo'        binlog.index    ib_buffer_pool    mysql.sock          server-key.pem
'#innodb_temp'        ca-key.pem      ibdata1           performance_schema  sys
 auto.cnf             ca.pem          ibtmp1            private_key.pem     undo_001
bash-4.4#
```

10. So, this is bind mount. Now if you remove this container, you will see that the date is still there.

```
ubuntu@ip-172-31-41-113:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                                  NAMES
3c86cacaa238   mysql     "docker-entrypoint.s…"   7 minutes ago   Up 7 minutes   33060/tcp, 0.0.0.0:3030->3306/tcp, :::3030->3306/tcp   demo
ubuntu@ip-172-31-41-113:~$ docker stop demo
demo
ubuntu@ip-172-31-41-113:~$ docker rm demo
demo
ubuntu@ip-172-31-41-113:~$ ls
dockerdata
ubuntu@ip-172-31-41-113:~$ ls dockerdata/
'#ib_16384_0.dblwr'   binlog.000001   client-cert.pem   mysql.ibd           server-cert.pem
'#ib_16384_1.dblwr'   binlog.000002   client-key.pem    mysql.sock          server-key.pem
'#innodb_redo'        binlog.index    ib_buffer_pool    performance_schema  sys
'#innodb_temp'        ca-key.pem      ibdata1           private_key.pem     undo_001
 auto.cnf             ca.pem          mysql             public_key.pem      undo_002
ubuntu@ip-172-31-41-113:~$
```

11. Bind mount is mostly used to inject data from host machine to the container, like code developers are writing. So, while the container is running, they can do all the code changes in the host machine directory and that will be reflected in the container. But for preserving data purpose, the better option is volumes, docker volumes.
12. Now run docker volume you will see how to create volume.
13. After that create a volume

    docker volume create dbdata

```
ubuntu@ip-172-31-41-113:~$ docker volume

Usage:  docker volume COMMAND

Manage volumes

Commands:
  create      Create a volume
  inspect     Display detailed information on one or more volumes
  ls          List volumes
  prune       Remove unused local volumes
  rm          Remove one or more volumes

Run 'docker volume COMMAND --help' for more information on a command.
ubuntu@ip-172-31-41-113:~$ docker volume create dbdata
dbdata
ubuntu@ip-172-31-41-113:~$ docker volume ls
DRIVER    VOLUME NAME
local     4cc6ddab75efd2238385c2db9ef19c13751ca4eb91c3faa8b06a638cab06a17d
local     dbdata
local     f8e37b12f807d83f8ba8ec7a948e0bb9a0d02e0a8e25290e0c486d12ec8295ca
ubuntu@ip-172-31-41-113:~$ docker run --name demo -d -e MYSQL_ROOT_PASSWORD=demodocker -p 3030:3306 -v dbdata:/var/lib/mysql mysql
ef4409947340abbd9a0c46ded27b71f5544b7eceafeb88c89d8b50566d4eaa69
ubuntu@ip-172-31-41-113:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS         PORTS                                                  NAMES
ef440994734c   mysql     "docker-entrypoint.s…"   10 seconds ago   Up 9 seconds   33060/tcp, 0.0.0.0:3030->3306/tcp, :::3030->3306/tcp   demo
ubuntu@ip-172-31-41-113:~$
```

14. As you might have deleted the container so you have to create the container again but this time instead of giving full name of the directory you can give the name of volume.

    docker run --name demo -d -e MYSQL_ROOT_PASSWORD=demodocker -p 3030:3306 -v dbdata:/var/lib/mysql mysql

15. And you can see the container is working.
16. Now if you want to see the logs you can run the logs commands.
17. After that run the inspect command and at the bottom you can find the IP address which it provides you. If you ping it the IP will respond.

```
ubuntu@ip-172-31-41-113:~$ ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.104 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.045 ms
64 bytes from 172.17.0.2: icmp_seq=4 ttl=64 time=0.052 ms
64 bytes from 172.17.0.2: icmp_seq=5 ttl=64 time=0.046 ms
^C
--- 172.17.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4079ms
rtt min/avg/max/mdev = 0.045/0.058/0.104/0.022 ms
ubuntu@ip-172-31-41-113:~$
```

18. Now if you want access MySQL you can also do that but first you need to install MySQL for that you can use the command it gives you copy that and paste it.

    mysql -h 172.17.0.2 -u root -pdemodocker

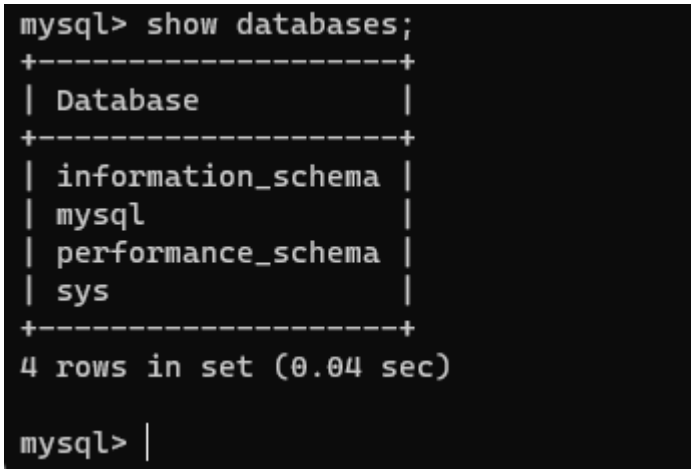
```
ubuntu@ip-172-31-41-113:~$ mysql -h 172.17.0.2 -u root -pdemodocker
Command 'mysql' not found, but can be installed with:
sudo apt install mysql-client-core-8.0     # version 8.0.35-0ubuntu0.22.04.1, or
sudo apt install mariadb-client-core-10.6  # version 1:10.6.12-0ubuntu0.22.04.1
```

19. After installing it if you run the command again you will see you are in MySQL.

```
ubuntu@ip-172-31-41-113:~$ mysql -h 172.17.0.2 -u root -pdemodocker
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.3.0 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

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.
```

20. Now if you run command to see MySQL database you will see it.

    show databases;


mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| performance_schema |
| sys |
+--------------------+
4 rows in set (0.04 sec)
mysql>