

## Experiment 4

### Docker Installation & Basic Commands of Docker

#### Part A:

##### Steps for Installing Docker:

1. Open the terminal on Ubuntu.
2. Remove any Docker files that are running in the system, using the following command:

```
$ sudo apt-get remove docker docker-engine docker.io
```

After entering the above command, you will need to enter the password of the root and press enter.

3. Check if the system is up-to-date using the following command:

```
$ sudo apt-get update
```

4. Install Docker using the following command:

```
$ sudo apt install docker.io
```

You'll then get a prompt asking you to choose between y/n - choose y

5. Install all the dependency packages using the following command:

```
$ sudo snap install docker
```

*alternate commands to install docker are*

```
$ sudo apt-get install \
  apt-transport-https \
  ca-certificates \
  curl \
```

```
software-properties-common
```

To install packages to allow apt to use a repository over HTTPS

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
```

command to add Docker's official GPG key

```
$ sudo apt-key fingerprint 0EBFCD88
```

Verify that you now have the key with the fingerprint

6. Before testing Docker, check the version installed using the following command:

```
$ docker --version
```

7. Pull an image from the Docker hub using the following command:

```
$ sudo docker run hello-world
```

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent
   it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/engine/userguide/
```

Check if the docker image has been pulled and is present in your system using the

To display all the containers pulled, use the following command:

```
$ sudo docker ps -a
```

## Part B: Docker search, docker Pull and docker run

Use the command `docker search` to search for public images on the Docker hub. It will return information about the image name, description, stars, official and automated.

Now that we know the name of the image, we can pull that from the Docker hub using the command `docker pull`. Here, we are setting the platform option as well.

```
$ sudo docker search mysql
```

*or alternate*

```
$ sudo docker pull --platform linux/x86_64 mysql
```

*or alternate*

```
$ sudo docker pull mysql/mysql-server:tag
```

A terminal window with a dark background and light green text. The prompt is 'debas@debas-Aspire-E1-470P:~/Desktop/backend \$'. The command entered is 'sudo docker pull mysql/mysql-server:5.75.7:'. The output shows the image being pulled from the Docker Hub, with progress bars for each layer and a final digest and status message.

```
debas@debas-Aspire-E1-470P:~/Desktop/backend $ sudo docker pull mysql/mysql-server:5.75.7: Pulling from
mysql/mysql-server
03ba86e1f15c: Pull complete
0f50e23a3dbe: Pull complete
93f9c1d05b5e: Pull complete
5a2d4104e777: Pull complete
Digest: sha256:ab6a10cd7607d8a0dba65e822adfa0cb442d411070e2c82e54e3316912b56531
Status: Downloaded newer image for mysql/mysql-server:5.7
```

Log into MySQL within the docker container using the `docker exec` command:

```
$ sudo docker exec -it mysql bash
```

Now run this command

```
$ sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=anypassword -
d mysql/mysql-server:5.7
```

You can check it by running the following command...The first image as you can see in the snippet is the `mysql-server` image in a new terminal

```
$ sudo docker ps -a
```

Remember, when we created and ran the MySQL container, we provided `MYSQL_ROOT_PASSWORD=anypassword`. Create a database and user, and grant privileges in MySQL (from within the container). Log into MySQL within the docker container using the `docker exec` command, Log into MySQL if you haven't already. After login, the `mysql>` prompt shows up:

```
$ mysql -uroot -panypassword
```

```
$ SHOW DATABASES ;
```

```
$ use database;
```

```
$ exit
```

```
$ exit
```

```
$docker restart
```

Let's restart our stopped container by using the following command. We may want to use this after we reboot our machine.

```
docker restart f8c52bedeccc
```

```
$docker rename
```

Now, let's change the container name from `compassionate_fermi` to `test_db`. We may want to change the name to keep track of our containers more easily.

```
docker rename compassionate_fermi test_db
```

```
$docker exec
```

Access the running container `test_db` by running the following command. It's helpful, if we want to access the MySQL command line and execute MySQL queries.

```
docker exec -it test_db bash
```

```
mysql -uroot -pmy-secret-pw
```

```
SHOW DATABASES;
```

The `-i` and `-t` options are used to access the container in an interactive mode. Then we provide the name of the container we want to access, which in this case `test_db`. Finally, the `bash` command is used to get a bash shell inside the container.

```
$docker logs
```

This command is helpful to debug our Docker containers. It will fetch logs from a specified container.

```
$docker logs test_db
```

If we want to continue to stream new output, use the option `-follow`.

```
docker logs -follow test_db
```

```
$docker rm
```

To remove a container, we can use the following command.

```
docker rm test_db
```

You may encounter an error like

Error response from daemon: You cannot remove a running container .....

Stop the container before attempting removal or force remove

As it recommends, we can stop the container first and then remove it or use option `-f` to remove a running container forcefully.

```
$docker stop test_db
$docker rm test_db# or docker rm -f test_db
$docker rmi
```

To free some disk space, we can use the docker rmi command with the image id to remove an image.

```
$docker rmi eb0e825dc3cf
```

These commands come with plenty of helpful options. If you want to know about other available options, run the docker command\_name --help command. For example:

```
docker logs --help
```

```
rvic=registry.docker.io: net/http: TLS handshake timeout
dushyant@dushyant-Predator-PH315-51:~$ ^C
dushyant@dushyant-Predator-PH315-51:~$ sudo docker pull --platform linux/x86_64 mysql/mysql-server:5.7
5.7: Pulling from mysql/mysql-server
d70f3c0cccba: Pull complete
e7dc89aa39f7: Pull complete
76cc4215b650: Pull complete
25b0b53e92: Pull complete
349b52643cc3: Pull complete
62dcf4a4134: Pull complete
c91c597e717d: Pull complete
c7e93886e496: Pull complete
Digest: sha256:1178cdd375f758968cd834ac4057bae41307e64b7c69a9e145896e7b11f48064
Status: Downloaded newer image for mysql/mysql-server:5.7
docker.io/mysql/mysql-server:5.7
dushyant@dushyant-Predator-PH315-51:~$ sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=hello -d mysql/mysql-server:5.7
$: command not found
dushyant@dushyant-Predator-PH315-51:~$ sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=hello -d mysql/mysql-server:5.7
4b142b458c2bcc291d4f0db017af1382e11d5f132216cc938056811e6a66fac3
dushyant@dushyant-Predator-PH315-51:~$ sudo docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED              STATUS              PORTS              NAMES
4b142b458c2b        mysql/mysql-server:5.7   "/entrypoint.sh mysql..." 45 seconds ago       Up 42 seconds (healthy)   33060/tcp, 0.0.0.0:3406->3306/tcp, :::3406->3306/tcp   mysql
d0ab838b67fa        docker/whalesay        "cowsay welcome to D..." 53 minutes ago       Exited (0) 53 minutes ago                               suspicious_kirch
75f0be3ff708        hello-world            "/hello"                  56 minutes ago       Exited (0) 56 minutes ago                               angry_gates
dushyant@dushyant-Predator-PH315-51:~$ sudo docker exec -it mysql bash
bash-4.2#
bash-4.2# ^C
bash-4.2# ^C
bash: '$@003': command not found
bash-4.2#
bash-4.2# ^C
bash-4.2# ^C
bash-4.2# ^C
bash-4.2# exit
dushyant@dushyant-Predator-PH315-51:~$ mysql -uroot -p hello
Command 'mysql' not found, but can be installed with:
sudo apt install mysql-client-core-8.0 # version 8.0.39-0ubuntu0.24.04.2, or
sudo apt install mariadb-client-core # version 1:10.11.8-0ubuntu0.24.04.1
dushyant@dushyant-Predator-PH315-51:~$ mysql -uroot -p hello
Command 'mysql' not found, but can be installed with:
sudo apt install mysql-client-core-8.0 # version 8.0.39-0ubuntu0.24.04.2, or
sudo apt install mariadb-client-core # version 1:10.11.8-0ubuntu0.24.04.1
dushyant@dushyant-Predator-PH315-51:~$ sudo apt install mysql-client-core-8.0
Reading package lists... Done
```

```
2024-10-24T09:03:19.117329Z 0 [Note] Skipping generation of SSL certificates as certificate files are present in data directory.
2024-10-24T09:03:19.117341Z 0 [Warning] A deprecated TLS version TLSv1 is enabled. Please use TLSv1.2 or higher.
2024-10-24T09:03:19.117349Z 0 [Warning] A deprecated TLS version TLSv1.1 is enabled. Please use TLSv1.2 or higher.
2024-10-24T09:03:19.119324Z 0 [Warning] CA certificate ca.pem is self signed.
2024-10-24T09:03:19.119480Z 0 [Note] Skipping generation of RSA key pair as key files are present in data directory.
2024-10-24T09:03:19.120627Z 0 [Note] Server hostname (bind-address): '*'; port: 3306
2024-10-24T09:03:19.120735Z 0 [Note] IPv6 is available.
2024-10-24T09:03:19.120771Z 0 [Note] - '::' resolves to '::';
2024-10-24T09:03:19.120812Z 0 [Note] Server socket created on IP: '::'.
2024-10-24T09:03:19.320575Z 0 [Note] InnoDB: Buffer pool(s) load completed at 241024 9:03:19
2024-10-24T09:03:19.371430Z 0 [Note] Event Scheduler: Loaded 0 events
2024-10-24T09:03:19.371860Z 0 [Note] mysqld: ready for connections.
Version: '5.7.41' socket: '/var/lib/mysql/mysql.sock' port: 3306 MySQL Community Server (GPL)
dushyant@dushyant-Predator-PH315-51:~$ sudo docker stop test_db
test_db
dushyant@dushyant-Predator-PH315-51:~$ sudo docker rm test_db
test_db
dushyant@dushyant-Predator-PH315-51:~$ sudo docker images
$: command not found
dushyant@dushyant-Predator-PH315-51:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
hello-world          latest              d2e94258ddcb       18 months ago      13.3kB
mysql/mysql-server   5.7                a4ad24fe52cd       21 months ago      432MB
docker/whalesay      latest             6b362a9ff73eb      9 years ago        247MB
dushyant@dushyant-Predator-PH315-51:~$ sudo docker logs --help
$: command not found
dushyant@dushyant-Predator-PH315-51:~$ sudo docker logs --help

Usage:  docker logs [OPTIONS] CONTAINER

Fetch the logs of a container

Aliases:
  docker container logs, docker logs

Options:
  --details      Show extra details provided to logs
  -f, --follow   Follow log output
  --since string Show logs since timestamp (e.g. "2013-01-02T13:23:37Z") or relative (e.g. "42m" for 42 minutes)
  -n, --tail string Number of lines to show from the end of the logs (default "all")
  -t, --timestamps Show timestamps
  --until string Show logs before a timestamp (e.g. "2013-01-02T13:23:37Z") or relative (e.g. "42m" for 42 minutes)
dushyant@dushyant-Predator-PH315-51:~$
```

Reference:

1. <https://www.simplilearn.com/tutorials/docker-tutorial/how-to-install-docker-on->

ubuntu

2. <https://towardsdatascience.com/12-essential-docker-commands-you-should-know-c2d5a7751bb5>

3. <https://docs.docker.com/engine/reference/commandline/container/>

4. <https://towardsdatascience.com/15-docker-commands-you-should-know-970ea5203421>

```
190 sudo apt-get remove docker docker-engine docker.io
191 sudo apt-get update
192 sudo apt install docker.io
193 sudo snap install docker
194 docker --version
195 sudo docker run hello-world
196 docker run docker/whalesay cowsay boo
197 sudo docker run docker/whalesay cowsay welcome to Docker!
198 sudo docker images
199 sudo docker ps -l
200 sudo docker ps -a
201 sudo docker ps
202 sudo docker search mysql
203 sudo docker exec -it mysql bash
204 mysql -v
205 sudo apt install mysql
206 sudo apt install sql
207 mysql --root
208 sudo docker pull mysql/mysql-server:5.75.7:pullingfrom mysql/mysql-server
209 \cd
210 sudo docker -v
211 sudo docker exec -it mysql bash
212 $ sudo docker search mysql
213 sudo docker search mysql
214 sudo docker pull --platform linux/x86_64 mysql/mysql-server:5.7
215 $ sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=hello -d mysql/mysql-server:5.7
216 sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=hello -d mysql/mysql-server:5.7
217 sudo docker ps -a
218 sudo docker exec -it mysql bash
219 mysql -uroot -p hello
220 mysql -uroot -phello
221 sudo apt install mysql-client-core-8.0
222 mysql -uroot -phello
223 mysql -uroot -p hello
224 sudo apt install mysql-server
225 mysql -uroot -p hello
226 sudo systemctl status mysql
227 sudo systemctl start mysql
228 mysql -uroot -p
229 sudo mysql
230 mysql -uroot -p
231 $ sudo docker restart mysql
232 sudo docker restart mysql
233 sudo docker rename compassionate_fermi test_db
234 sudo docker ps -a
235 sudo docker rename mysql test_db
236 sudo docker exec -it test_db bash
237 mysql -uroot -p
238 sudo docker logs test_db
239 sudo docker stop test_db
240 sudo docker rm test_db
241 $ sudo docker images
242 sudo docker images
243 $ sudo docker logs --help
244 sudo docker logs --help
245 sudo docker image
246 sudo docker images
247 sudo docker kill mysql
248 sudo docker ps
249 sudo docker run docker/whalesay cowsay welcome to Docker!
250 sudo docker ps
```