**Dockers**

**About Docker:**

\* Docker is an application that makes it simple and easy to run application processes in a container .

\* It's a platform for developer and sysadmins.

\* Build once and run anywhere.

\* Docker lets you quickly assemble applications from components and eliminates the friction that can come when shipping code.

\* Dockers are provides better storage than traditional virtualization.

\* In Docker architecture only we have base image, along with base image we have other layers, and these images are put together by the docker using union file system.

**Dockers Installation On ubuntu 14.04**

\* apt-cache search dockers.io

\* apt-cache show dockers.io

\* Download the script from <https://get.docker.com/>

or simply run below command

wget -qO- https://get.docker.com/ | sh

\* usermod -aG docker <username>

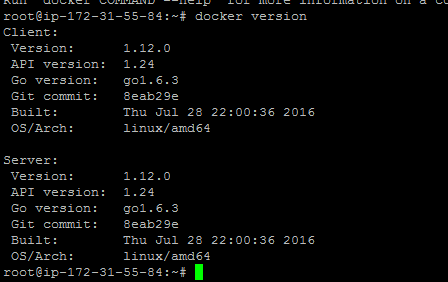
**example**

usermod -aG docker krishna

**Docker Operations :**

**1. Check the docker version**

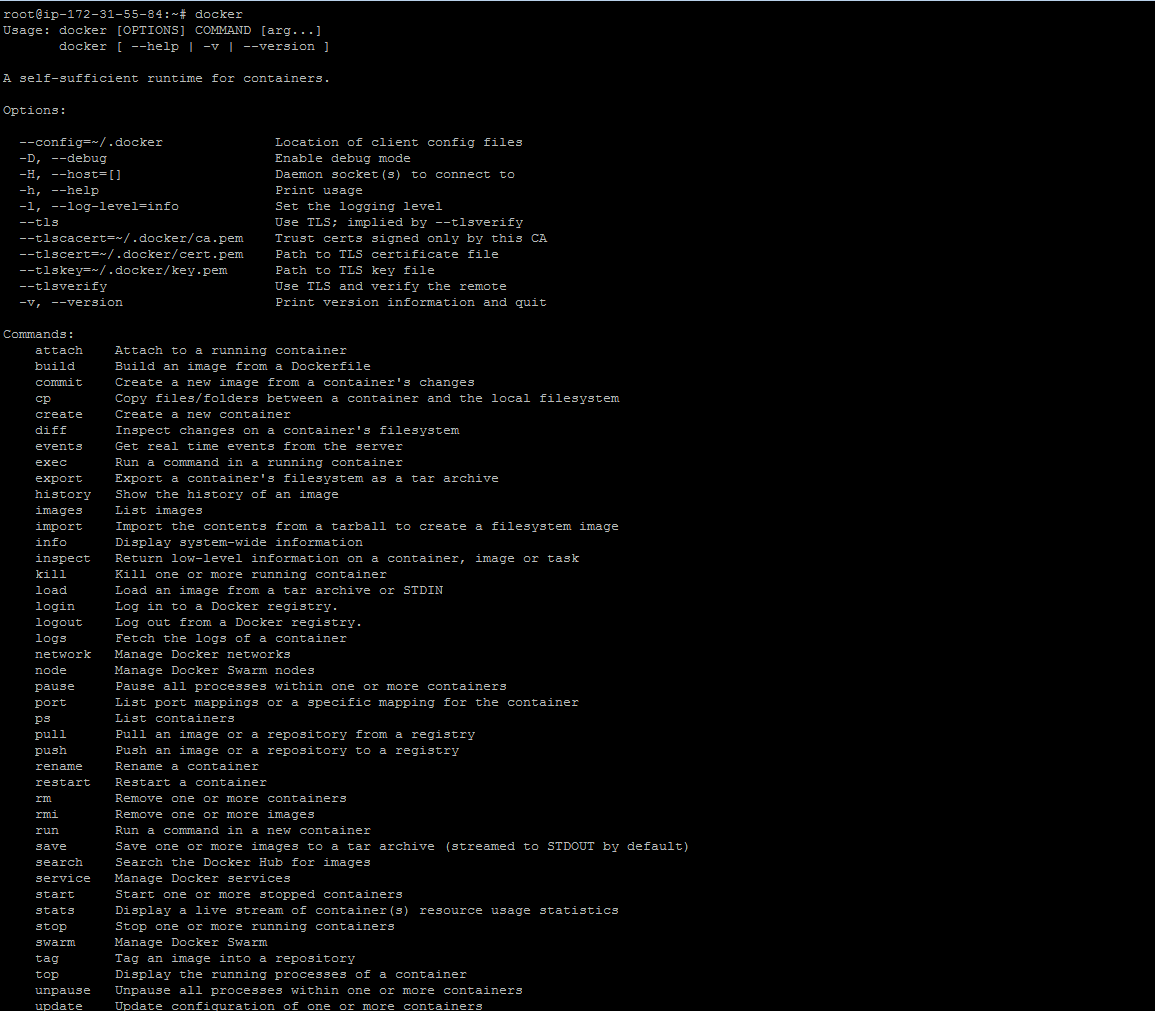
* docker version



**2. Check docker commands and usage**

docker

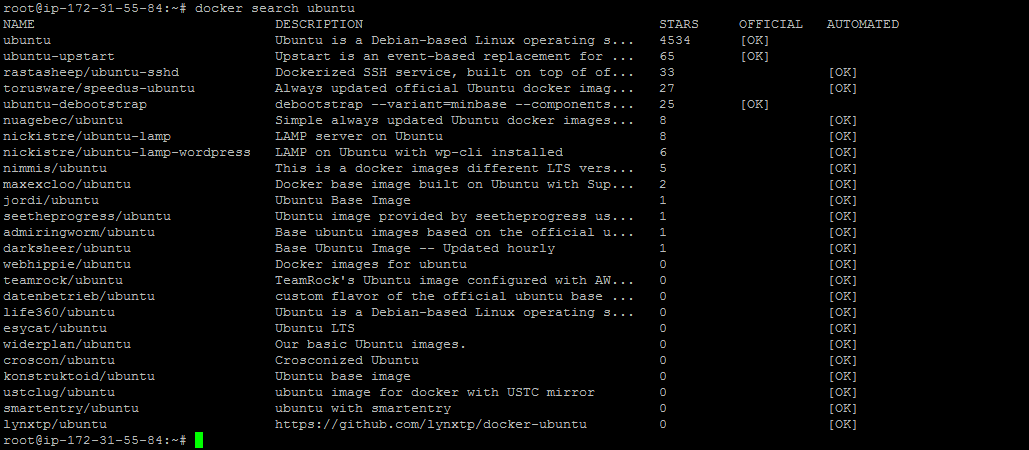
Example :



**3. Search docker images**

docker search ubuntu

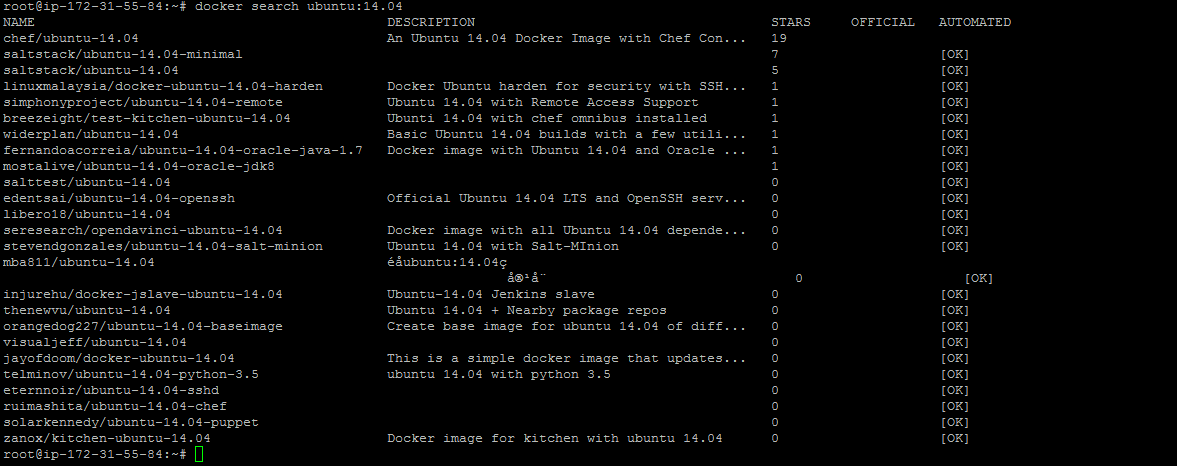
Example :



**4. Search the docker images for specific version using tags**

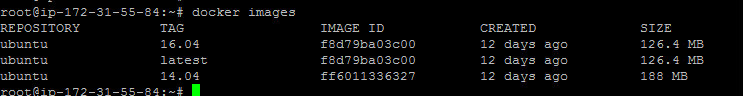
docker search ubuntu:14.04

Example :



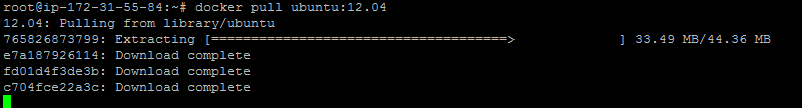
**5 : Check your docker images**

docker images



**6: Pull the docker images**

docker pull ubuntu:14.04



**7: Run the Docker containers**

docker run -it --name=krishna-docker ubuntu:14.04 /bin/bash



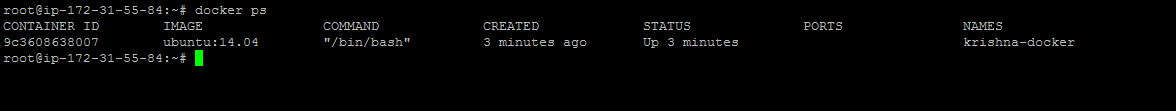
-i :- interactive

t :- attach terminal

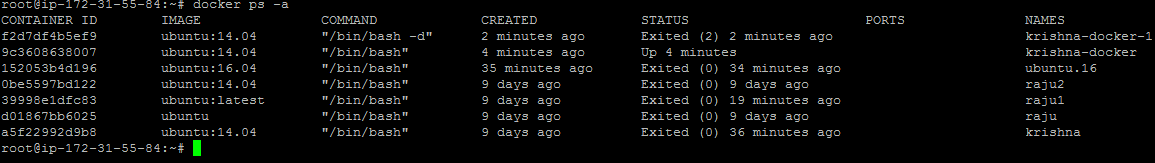
Note : it will login to new docker container once you run above commands

**8: Check the Docker containers status**

docker ps



Note : **docker ps** command will only show the running containers only , you want to see all the containers use **docker ps -a**



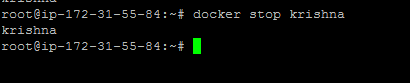
**9: Start the containers**

docker start <container name/id>



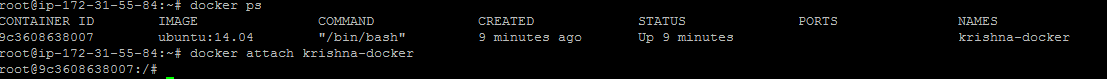
**10: Stop the docker containers**

docker stop <container name/id>



**11: Attach to running container's**

docker attach <container name/id>

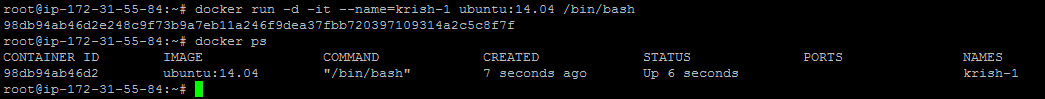


Note : if you want keep docker container running and you want quit from the server , just use

CTRL+P+Q

**12: Create a docker container in detach mode**

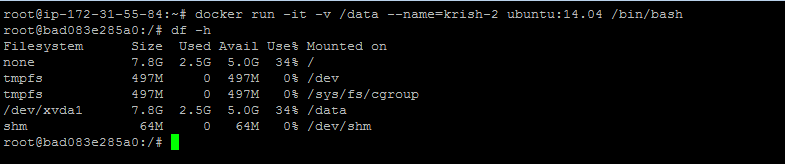
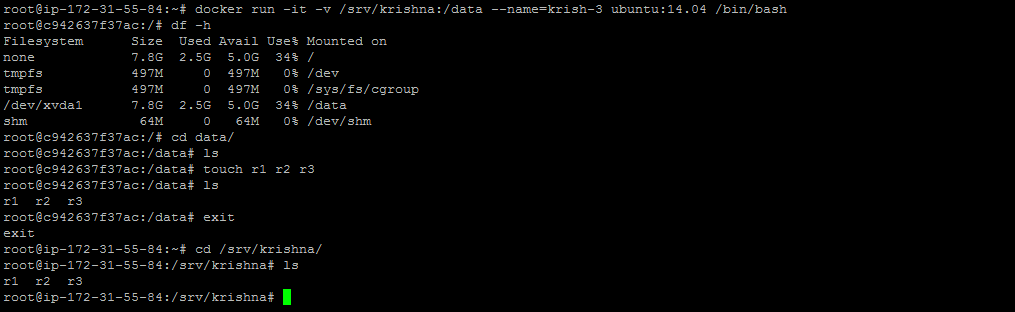
docker run -d -it --name=krish-1 ubuntu:14.04 /bin/bash



**13 : Assing persisten storage to docker containers**

docker run -it -v /data --name=krish-2 ubuntu:14.04 /bin/bash

docker run -it -v /srv/krishna:/data --name=krish-3 ubuntu:14.04 /bin/bash



**14 Docker Port mapping**

By default Docker containers can make connections to the outside world, but the outside world cannot connect to containers.

So we need to define port mapping here

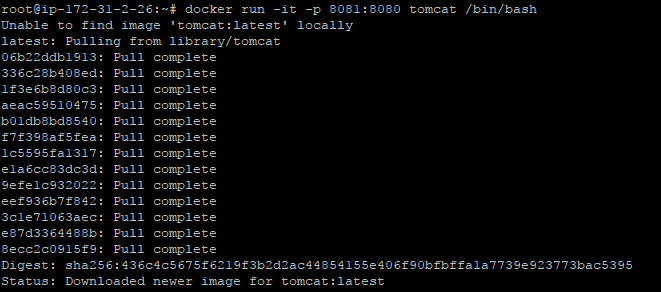
Hostport: - port which we can access outside the world

Container port: Port which actually application running.

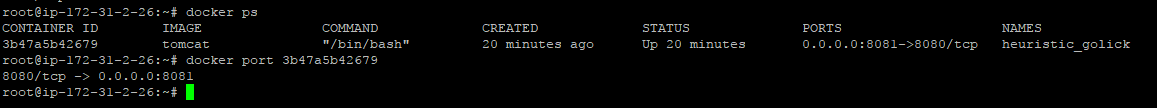
We will define with the option –p <host-port>:<container-port>

**Example**:

docker run -it -p 8081:8080 tomcat /bin/bash



**Verification:**



Open the browser and access with port 8081

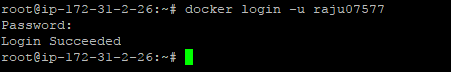
http://<ip-addr>:8081

**15. Docker login**

Create a docker account on <https://cloud.docker.com/>

Then Login from your command line

**docker login –u <docker-user-id>**



**16. Docker tag,commit,push**

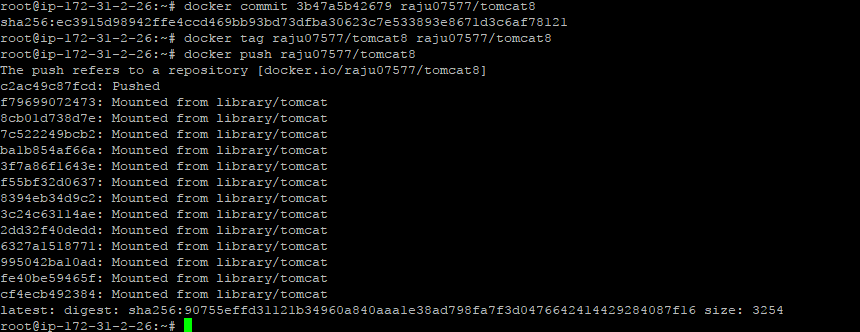
Tag the container to the name that will use when we push to repo.

**For example**, you have cloned Ubuntu image and then run container with that image , then installed some dependency package , then installed required application and configured based on your required way , now your container is running with your application with your custom required way. So you need commmit this container as a image with tag name , for that you need to tag it and then push using tag name.

docker commit <container-id> <name-of the –app>

docker tag < name-of the –app> <tag-name-of-app>

docker push <tag-name>



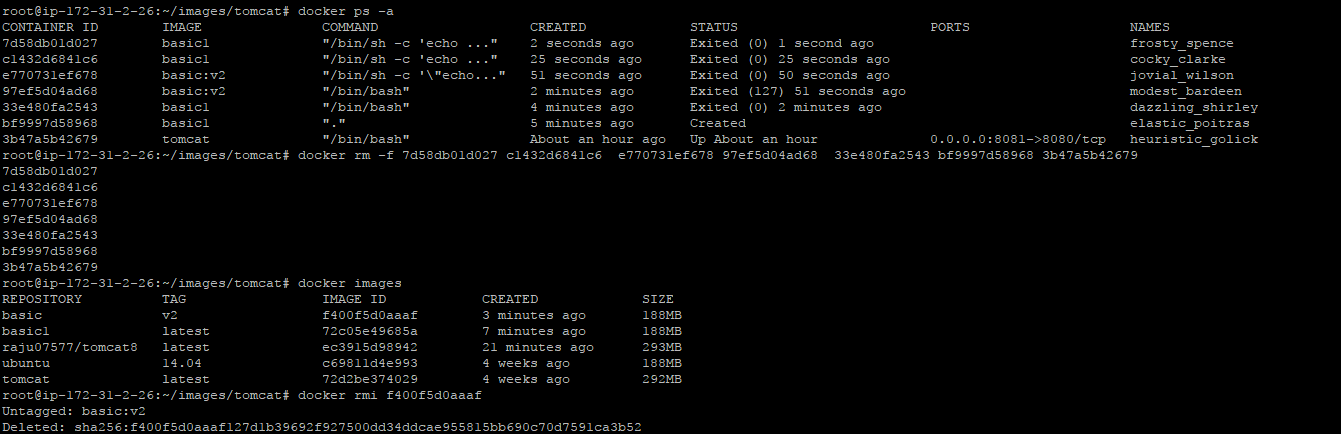
17. Delete the Docker images and container

**To delete the container**

Docker rm –f <container id>

**To delete the image**

Docker rmi <image-id>



**Docker file creation:**

A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

**CMD:** The command that runs when the container starts

vim Dockerfile

FROM ubuntu:14.04

MAINTAINER KRISHNA RAJU.RUDRARAJU01@GMAIL.COM

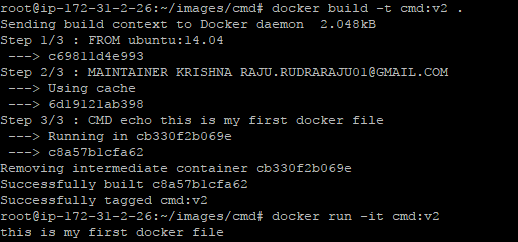
CMD echo this is my first docker file

**Create image using build**

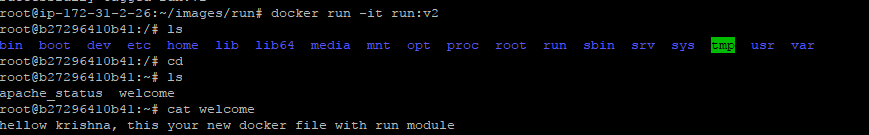
Docker build –t <tag-name > <location of docker file>

**Then run the created image**

Docker run –it <image-name>



|  |  |
| --- | --- |
| **RUN:** | Executes a command and save the result as a new layer |
|  | **Vim Dockerfile**  from ubuntu:14.04  maintainer krishna raju.rudraraju01@gmail.com  RUN echo "hellow krishna, this your new docker file with run module" > /root/welcome  RUN apt-get update  RUN apt-get install apache2 -y  RUN echo "apache is installaed" > /root/apache\_status  **Build image**:    **Run the container using new image** |



**ADD**: Copies a file from the host system onto the container

**Vim Dockerfile**

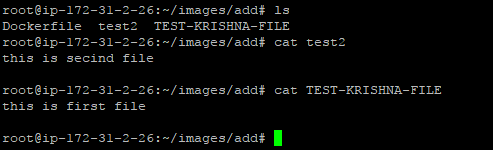
from ubuntu:14.04

maintainer krishna raju.rudraraju01@gmail.com

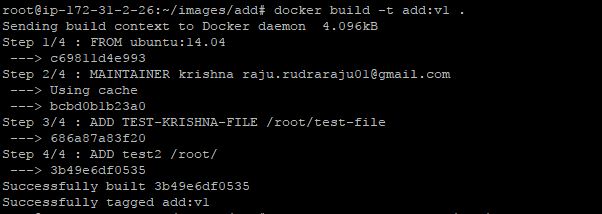
ADD TEST-KRISHNA-FILE /root/test-file

ADD test2 /root/

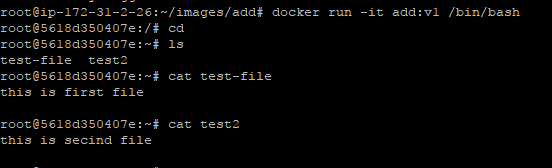
Create file called TEST-KRISHNA-FILE and test2 in your current dockerfile location



**Then Build the image:**



**Then create a container using this image**



**WORKDIR:** Set the default working directory for the container

**How to find free space in docker container.**

Ans-#Docker stats container-id

$ docker stats

CONTAINER CPU % MEM USAGE / LIMIT MEM % NET I/O BLOCK I/O

1285939c1fd3 0.07% 796 KiB / 64 MiB 1.21% 788 B / 648 B 3.568 MB / 512 KB

9c76f7834ae2 0.07% 2.746 MiB / 64 MiB 4.29% 1.266 KB / 648 B 12.4 MB / 0 B

d1ea048f04e4 0.03% 4.583 MiB / 64 MiB 6.30% 2.854 KB / 648 B 27.7 MB / 0 B

1. What is the default **network in docker**?

Ans: Bridge.

1. What is the default **network in Docker Swam**.

Ans: overlay network

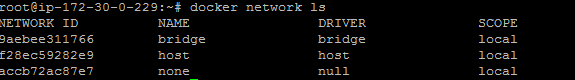
1. How to create a **network in** **docker**.

Ans: docker network create -d bridge <network name>

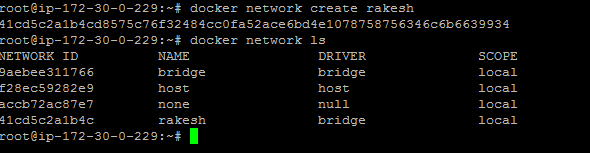
4) How to create a network in **docker swam**

Ans: docker network create -d overlay <network name>

**How to check docker network.**



**How to Create docker network**



**How to create network in docker swam**

**Ans:** docker network create -d overlay rakesh

**How to add running swam container**

docker network create -d overlay --attachable rakesh

[**https://docs.docker.com/engine/reference/commandline/network\_create/#examples**](https://docs.docker.com/engine/reference/commandline/network_create/#examples)

**How to attach an network to the container.**

docker run -itd --network=mynet busybox

**How to create an single subnet in bridge network**

docker network create --driver=bridge --subnet=192.168.0.0/16 br0

**How to create multi subnets in overlay networks**

docker network create -d overlay \

--subnet=192.168.0.0/16 \

--subnet=192.170.0.0/16 \

--gateway=192.168.0.100 \

--gateway=192.170.0.100 \

--ip-range=192.168.1.0/24 \

->Difference between **docker top** and **docker stats.**

Ans:- Docker top is used to see single container details. Docker stats is to see all containers details.

-> difference between **docker copy** and **docker add**

Ans:- docker Copy is used to copy data from server to docker container.

Docker add is used to copy data from out side of the server.

Difference between **RUN** and **CMD.**

**Run:-** command triggers while we build the docker image.

**CMD**:- command triggers while we launch the created docker image.

Entrypoint:- this is used after container launched it will do tasks.

Example:- restarting services like apache service etc..

->what is **--rm** in docker.

# Ans:- docker --rm does not delete container after exit

Example: docker run -it --rm ubuntu

->How to check **docker logs.**

Ans:- docker logs <container name/container id>

->How to follow docker logs.

Ans:- docker logs -f <container name>

->How to follow docker logs in last 10 lines

Ans:- docker logs -f --tail 10 <container name>

->How to create mysql image in docker.

Ans:- docker run --name databasename -e MYSQL\_ROOT\_PASSWORD=password -d mysql:latest.

->How to link two containers.

Ans:- docker run --link databasename:db -p 8080:8080 adminer

How to stop all the running containers

Ans:- docker stop $(docker ps -q)

How to remove all the containers

Ans:- docker rm $(docker ps -a -q)

How to filter docker containers.

Ans:- docker ps --filter status=created/exited/stop/running

How to know details of container

Ans:- docker inspect <container ID>

How to remove all exiting containers.

Ans:- docker rm $(docker ps --filter status=exited -q)

How to clean all the networks at a time.

Ans:- docker network prune

How to list networks

Ans:- docker network ls

How to remove all the volumes/containers.

Ans:- docker volume prune

How to build docker-compose file.

Ans:- docker-compose up -d

How to rebuild docker-compose file

Ans:- docker-compose up -d --build

How to stop, restart,pause docker-compose file

Ans:- docker-compose restart/pause etc…