**Docker (Virtualization Platform)**

This is a containerization platform which can be used for creating the development environment, testing environment and production environment..etc. Docker use a concept called containers. This is the next step in virtualization.

**Virtualization**

This is a process where it is posssible to run multiple OS's on one server. This is done thorugh an applcation called hypervisor.

On the server(baremetal) we first install the Host OS and on top of it we install the hypervisor s/w(Vmware esxi,Citrix xen,Microsoft Hyper-v etc).On the hypevisor we can install the Guest os and on the guest os we install the appllications that we require.

The problem with the above architecture is the applications running on the guest os have to pass through multiple layers in order to access the h/w resources.also it is not possible to allocate h/w resources dynamically to the VM's depending on their requirement.

**Docker Containarization**

In docker we have a bare metal on top of which host OS is installed and on the host OS we install a s/w application called "Docker engine".On the docker engine we can run any application as a container.The advantage of this process is these applications have to pass through less number of layers to access the hardware resources

Also docker engine can dynamically allocate h/w resources to individual containers depending on how many process are running on it

Through docker we achive what is called "process isolation" ie we separate the application from its dependency on the underlying OS and we create application running only on docker engine.These applications can then be deployed on any OS where docker is running.

**Docker Images and Containers**

An image is a collection of bin/lib that are necessary for an application to run. All the docker images are present in the cloud site of docker called hub.docker.com

A Container is a running instance of the image. It is the indivdual application or process that docker has created in our user space

**Docker Components**

**Dcoker Host***:* This is the machine where docker is installed and all the docker images are downloaded.

**Docker Client:** This is the terminal of docker where we can fire the docker commands

**Docker Deamon:** This is a background preocess which takes the commands fired by docker client and sends them to the docker images or contianers or docker registry

**Docker Registry***:* This is the cloud site of docker where all the docker images are present. hub.docker.com

We can also create a private registry whcih can be accessed only by our organisation

**Important docker commands**

**Docker Image commands**

1 To downlaod an image

docker pull image\_name

2 To see the list of all images in our docker host

docker images

or

docker image ls

3 To search for an image on hub.docker.com

docker search image\_name

4 To delete an image from our docker host

docker rmi image\_name

5 To upload a docker image

docker push imagename

6 To create an image from a container

docker commit container\_id/container\_name new\_image\_name

7 To create an image from a docker file

docker build -t new\_iamge\_name path\_of\_dockerfile

**Docker Container commands**

8 To see the list of running containers

docker container ls

9 To see the list of all containers(running and stopped)

docker ps -a

10 To find detailed info about a container

docker inspect container\_name/container\_id

11 To see the logs generated by a container

docker logs container\_name/container\_id

12 To start a stopped container

docker start container\_name/container\_id

13 To stop a running container

docker stop container\_name/container\_id

14 To restart a container

docker restart container\_name/container\_id

To restart after 10 seconds

docker restart -t 10 container\_name/container\_id

15 To remove a stopped container

docker rm container\_name/container\_id

16 To remove a running container

docker rm -f container\_name/container\_id

17 To stop all running containers

docker stop $(docker ps -aq)

18 To remove all stopped containers

docker rm $(docker ps -aq)

19 To remove all containers(running and stopped)

docker rm -f $(docker ps -aq)

20 To create a container from an image

docker run image\_name

**Run command options**

-it Used for opening an interactive terminal in the container where we can fire linux commands

-d Runs the container as a deamon ie it runs the container in the background

--name Used for assigning a name to a container

-p Used for port mapping.It will link the container port with the docker host port.Container port is called internal port and docker host port is called external port

Eg: -p 8080:80

Here 8080 will be docker host port and 80 will be container port

-P Used for automatic port mapping ie the internal port of container will be mapped with a port on docker host and this external port will be greater than 30000

-v Used for mounting volumes on a container

--volumes-from Used for creating reusable volumes that can be shared between containers

--network Used for assigning a network to a container

-e Used for assigning environment variables

-rm Used for removing the container on exit

--mem Used for allocating memory to containers

--cpu Used for allocating fixed amount of cpu to containers

21 To see the ports used by a container

docker port container\_name/container\_id

**Docker Networking commands**

22 To see the list of docker networks

docker network ls

23 To create a new network

docker network create networkname

24 To find detailed info about a network

docker network inspect network\_name/network\_id

25 To delete a network

docker network rm network\_name/network\_id

26 To attach a network to a running container

docker network connect network\_name/network\_id container\_name/container\_id

27 To disconnect a network from a running container

docker network disconnect network\_name/network\_id container\_name/container\_id

28 To run a particular command in a running container

docker exec -it container\_name/container\_id command\_to\_run

Eg: To open bash terminal in an already running contianer

docker exec -it container\_name/container\_id bash

29 To comeout of a container without exit

ctrl+p,ctrl+q

30 To go into a container which is already runing in background

docker attach container\_name/container\_id

**Creating and using a containers**

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another

Creating containers by using fallow command:

Docker run <options> baseimage

Options:

--name, -it, -d, -p, -P, -e….etc

**UseCase1:**

Start an nginx container in detached mode and name it webserver. Map the external port of docker host 8080 with internal port 80 of container

1 Start nginx as a container

dcoker run -d -p 8080:80 --name webserver nginx

2 To access the nginx

Launch any browser

ipaddress-of-dockerhost:8080

**UseCase2:**

Start a tomcat container and publish the port numbers,and give it a name mytomcat

1 Start tomcat

docker run --name mytomcat -d -P tomcat

2 To find the ports of tomcat

docker port mytomcat

3 To see the home page of tomcat

Launch any browser

ipaddrsss-of-dockerhost:32768 (external port displayed in the above command)

**UseCase 3:**

Start an ubuntu container and go into its interactive terminal. Fire some linux commands in it

1 Start ubuntu as a container

docker run --name myubuntu -it ubuntu

2 In the container we can run linux commands

3 To come out of the container

exit

**UseCase4:**

Start mysql as a container. Go into its interactive terminal and login into the database,Create some sql tables

1 Start mysql as a container

docker run -d --name rr-mysql -e MYSQL\_ROOT\_PASSWORD=root123 mysql

2 To open interactive terminal in the already runnign container

docker exec -it rr-mysql bash

In the container to login into the db

mysql -u root -p

Enter password

3 To see the list of default availabel databases

show databases;

4 To switch into any of the above databases

use dbname;

5 To create some tables in this db

Open

https://justinsomnia.org/2009/04/the-emp-and-dept- tables-for-mysql/

Copy the code for creating emp and depttables

Paste in the mysql container

6 select \* from emp;

select \* from dept;

**Customising Docker images:**

We can create our own docker images and save them in hub.docker.com or in our private docker registry

This can be done in two ways

1 Create a container customised according to our requirment and save it as a image(snapshot)

2 Use dockerfile

**UseCase5:**

Create a ubuntu container and install git in it

Save that container as an image(snpashot)

Start a container frm this new image and we should find git

installed on it

1 Start ubuntu as a contianer

docker run -it --name myubuntu ubuntu

2 In the container

apt-get update

apt-get install -y git

exit

3 Save the container as an image

docker commit containerid/containername new-imagename

Eg. docker commit myubuntu ubuntu1

4 Start a container from the above image and we will find

git preinstalled on it

docker run -it ubuntu1

In the container

git --version

**Docker Volumes**

Since docker containers are for temporary usage, once the container is removed all the data stored in the container will be lost.To preserve the data we can use docker volumes

Docker volumes are of two types

1 Simple docker volumes

2 Docker volume containers

**Simple Docker Volumes:**

These volumes can be used by only one docker container

**UseCase6:**

Create a empty folder /rr\_tutorial.

Start an ubuntu container and mount /rr\_tutorial folder on it

Create some files in /rr\_tutorial in the contianer

Stop and remove the container,we should still be able to access the files

1 Create a directory

mkdir /rr\_tutorial

2 Create a ubuntu container and mount /rr\_tutorial on it

docker run --name myubuntu -it -v /rr\_tutorial ubuntu

In the container

cd rr\_tutorial

touch file1 file2 file3

exit

3 To find the location where the mounted data is stored

docker inspect myubuntu

In the JSON output that is generated search for "Mounts"

Copy the "Source" path

4 Stop and remove the container

docker stop myubuntu

docker rm myubuntu

5 To access the files created in the container

cd path-of-source-folder