**Basic cmds:**

$ docker run ansible : to run ansible image

$ docker ps : list containers

$ docker ps -a : running + stopped container list

$ docker stop “container name” : to stop container

$ docker rm “container name” : to remove the container

$ docker images : to see list of images

$ docker rmi “name of image” : remove image

$ docker pull “image name” : to pull image from docker hub

$ docker run ubuntu sleep 5 : when ubuntu starts it goes to sleep for 5 sec.

$ docker exec “name of container” cat /etc/hosts : here exec runs the ‘cat’ cmd on the given container

$ docker run -d “image name” : to run in detach mode

$ docker attach “container name” : to attach with the container

$ docker run -it centos bash : it will run centos, and we will be directly logged in to this container (centos)

$ docker container prune : to delete all stopped containers.

**Run-cmds :**

$ docker run redis:4.0 : to run ‘4.0’ version (use of tag)

$ docker run -it “image name” : to run image in interactive-terminal mode , i.e : if our application ask info from user (like name ,age) …we have run docker image in interactive mode to enter those values.

$ docker run -p 80:5000 “image name” : here 80 = docker host port and 5000 = docker container port (on which our application is running), we run this cmd so we can access this application from our browser (http://192.168.1.5:80)

$ docker run -v /opt/datadir:/var/lib/mysql mysql : mapping a volume, here /opt/datadir= directory outside of container and /var/lib/mysql = container directory ,to retain data even if the container gets deleted

$ docker inspect “container name” : detail info of container

$ docker logs “container name” : to see logs

**Images:**

$ docker build . -f dockerfile -t “name of image” : to builds image from dockerfile and -t is used to give name to the newly created image.

$ docker build -t your-docker-id/your-image:v0.1 . : build image with docker hub id (put dot in the end)

$ docker push ***your-docker-id***/ your-image:v0.1 : to push it to your docker hub repo

$ docker push “name of image” : to push image to docker repo

$ docker history “image-name” : to see history of image

**$ docker exec -u root -t -i container\_id /bin/bash** : to go inside a running container and execute cmds manually

**Environment variables:**

$ docker run -e APP\_COLOR=blue “image name”

**Docker Compose:**

$ docker-compose up : to run the compose.yaml

$ docker run “image1” --link “containername2: hostname2” : to link application 1 with application 2

**Docker Engine:**

$ docker exec “container id” ps -eaf : it will show all the processes running on the container

**Volume:**

$ docker volume create “name of volume” : to create a volume

$ docker run -v /opt/datadir:/var/lib/mysql mysql : mapping a volume, here /opt/datadir= directory outside of container and /var/lib/mysql = container directory ,to retain data even if the container gets deleted

$ docker run -v data\_volume:/var/lib/mysql mysql : it will create **“data\_volume”** volume and will attach it to container

$ docker run \

--mount type=bind,source=/data/mysql,target=/var/lib/mysql mysql : new method to attach volume.

$ docker system df : to see the actual disk usage by images

**Network:**

$ docker network create --driver bridge --subnet 182.18.0.0/16 “n/w-name” : to assign different internal n/w to docker container