* + docker images -> will list all the images which are present locally in the docker engine
  + docker ps -a -> means process status it will give you the status of all running containers as well as stopped containers
  + docker ps -> It will only give you the list of running containers
  + docker info -> will provide info for your docker
  + docker -v -> will give you the version of docker installed on your system
  + docker run -it --name <container\_Name> <Image\_name> /bin/bash
  + Ex -> docker run -it --name sanchit\_container ubuntu /bin/bash
  + docker rm <container\_name> ex docker rm sanchit\_container
  + docker start <container\_name> ex docker start sanchit\_container
  + docker stop <container\_name>. Ex docker stop sanchit\_container
  + docker attach <container\_name> ex docker attach sanchit\_container
  + Exit -> to exit out of the conatiner
  + docker pull ubuntu -> will pull the image from docker hub
  + docker images -> will list down all the images
  + docker commit <container\_name> <Image\_name> ex docker commit sanchit\_ubuntu\_container newUbuntu
  + docker diff <container\_name> ex docker diff sanchit\_ubuntu\_container
* This will list the changes that you have done on an image   In case if you forgot to provide name to your new image for docker commit command the you can use the below command to give a tag name to the container   docker tag <Image Id> <new\_image\_name> :<Tag name> for ex   docker tag c7a88698ebda newest\_ubuntu:latest
  + - docker system prune -a
* Will delete everything, like docker images, docker containers all the things would be deleted form your machine.
  + Build an image from a Dockerfile
* docker build -t myimg .
  + Docker file for creating a volume
* Dockerfile FROM ubuntu Volume ["/myVolume"]   Create an image from the docker file by building it docker build -t myimage . Docker run -it --name container1 myimage /bin/bash   After running access the myVolume folder and create 2 or 3 file  cd myVolume touch myfile1, myfile2   Create another container from a diff image Docker run -it --name=container2 --privileged=true --volumes-from container1 openjdk /bin/bash   Example docker run -it --name=container2 --volumes-from mycontainer --privileged=true openjdk /bin/bash      If you want to do the same thing from commands    docker run -it --name container3  -v /volume2 ubuntu /bin/bash docker run -it --name container4 --privileged=true --volumes-from=container3 centos /bin/bash     Example    docker run -it --name container3 -v /volume2 centos /bin/bash   docker run -it --name container4 --privileged=true --volumes-from=container3 ubuntu /bin/bash         Creating volume and Host to container volume mapping     Host to Container Mapping  Example docker run -it --name hostcont -v /Users/I546178/documents/perosnal\_docker:/volume4 --privileged=true centos /bin/bash
  + docker volume ls  -> To display all the volumes
  + docker port <conatinerName> -> used to display all the mapped ports from a container
  + Docker exec -it <container\_name> /bin/bash -> to go inside container (Diff between the docker attach and docker exec is that with docker exec a new process gets created where as docker attach does not create a new process rather it just goes into the main process)
  + Docker run -itd -> d stands for daemon and it will not go inside the container
  + Docker expose and publish (-p) difference is that expose is used for inter container communication whereas P stands for port or publish, port or publish command has higher precedence than expose as -p will overwrite the expose command, if you use -p then it will implicitly have an expose.
* docker run -itd --name <container\_name> -p 8080:8080 jenkins Example->
  + docker run -itd --name techserver -p 8080:8080 jenkins
* To login to your account via terminal docker login -u sanchit007

For docker push command refer the below url:   <https://stackoverflow.com/questions/41984399/denied-requested-access-to-the-resource-is-denied-docker>