**DOCKER VOLUMES:**

**>> what is root mode and ditached mode ?**

**ROOT MODE:**

root mode is nothing but directly if you hit a command **$docker run -it image-name**

**Ditached mode:**

Ditached mode is nothing but if you hit a command **$docker run -itd imagename**

>>docket inspect , if you **$docket inspect** command all details will be back

**----------------------------------------------------------------------------------------------------------------**

**DOCKER VOLUMES:**

**Def: Share data between the host operating system to containers.**

**or**

**share the data from host operating system to containers is called docker volume.**

--> Suppose you have host operating system like ubuntu, in that server we are downloading one image and from that image we are creating multiple containers.

Now what i am doing is i am sharing data from host operating system(ubuntu) to containers

--------------------------------------------------------------------------------------------------------------------

**HOW TO SHARE DATA FROM ONE CONTAINER TO ANOTHER CONTAINER :**

>> to share the data from one container to another container the command is ,

**$docker run -itd --volumes-from container-id1 containerid2 imagename(centos)**

**Uses of Docker Volumes:**

* Decoupling containers from storage
* Attach volumes to container
* We can delete the containers but we can’t delete the volumes
* We can share the data from host operating system to containers

**How to create Docker volume:**

* Login into the server like Ubuntu or CentOS and Redhat
* Docker should be installed in the base server

**🡪Create Docker volume**

#docker volume create volume-name(Myvolume)

**🡪List the volumes**

#docker volume ls

**🡪To get the details about the volume**

#docker volume inspect Myvolume(volume-name)

**🡪If you want to remove the volume**

#docker volume rm Myvolume

**If you want to remove all the unused volumes you can run the below command**

$docker volume prune 🡪It will ask you Yes or No

**How can we use Docker Volumes:**

For example I want to start Jenkins container

🡪Now I will pull the Jenkins image using the below command

$docker pull Jenkins ( For more information go to docker hub account and check )

🡪Now I will start the container

**$docker run –p 8080:8080 –p 50000:500000 jenkins**

Now I have to give some name to my container

**$docker run –name Myjenkins –p 8080:8080 –p 50000:500000 jenkins**

Now I want to attach a volume so I will run the command like

**$docker run –name Myjenkins –v Myvolume(volume-name):/var/jenkins\_home –p 8080:8080 –p 50000:500000 jenkins**

Here Myvolume is a volume name and it should get data from /var/jenkins\_home

If I run the above command it will start the Jenkins on a docker container and it will produce the initial admin password

🡪I will just copy the initial admin password for jenkins to login

🡪I will logon to Jenkins and I will create one job after that again I will create new container from the Jenkins image

Now again I will run the same command by renaming container name and changing port number

**$docker run –name Myjenkins1 –v Myvolume(volume-name):/var/jenkins\_home –p 8081:8080 –p 60000:600000 jenkins**

🡪Now this will start another instance of jenkins or different docker container, Once this is started let us see if the volume is shared between the jenkins or not

If I run open jenkins with port number 8081 in the browser I should able to see the job which in the new first container it means data shared between two containers

🡪So the data is being shared between jenkins container which is running on port 8080 and another jenkins which is running on port 8081

Suppose if we delete or we stop these two containers or any one container volumes will be there

* We can delete the containers but not the volumes
* If we delete the containers also volumes will be there

**Bind Mount:**

Bind Mounts is nothing but instead of using docker volumes we will use Bind Mount, We will use docker volume and Bind Mount for same purpose.

The command also same for Bind Mount but instead of giving volume name we will give Bind Mount name (path we can give)

The below command for creating docker volumes

**$docker run –name Myjenkins –v Myvolume(volume-name):/var/jenkins\_home –p 8080:8080 –p 50000:500000 jenkins**

The below command for creating Bind Mounts,

**$docker run –name Myjenkins2 –v /Users/evm2kor/desktop/bindmount:/var/jenkins\_home –p 9090:8080 –p 40000:50000 jenkins**

Now all the data of Jenkins\_home will get stored in bindmount directory on desktop

Here we are changing the name of the containers and port numbers but volume is same

**/Users/evm2kor/desktop/bindmount 🡪 It is directory path**