1. A well-known claim is that Docker is not suitable for Databases, mainly due to performance considerations. What is your view on this?

Can by using volumes/tmpfs mount this issue can be solved?

What are the key pros and cons of using Docker for Databases?

Also Explain whether dockerizing is necessary for a BigData Environment like Apache Hadoop or Spark?

2. Explain how to keep the containers alive, even when the Docker Daemon is down

3. What are Google's Distroless images? What is the difference between it and Alpine distribution in terms of software's bundled?

4. How to view the logs for the last 1 hour, logs for a particular date, tail the log of a container continuously?

5. Explain how to launch GUI applications from container?For instance, explain how to run a Javafx or Python desktop application on the host machine from the container, so that containers can be used for software distribution.Is it possible to distribute Desktop apps using Docker?

6. How to get the IP address and gateway details of the container? List 5 ways to get it.

7. How to start Containers that will run as user with normal Privileges and not as Root user?

8. What is the key issue addressed by using docker run command with option **–device- write-bps**

9. What is the Command to list all the ports used by a container? How to list all the Mapped ports in a container? List at least two ways to do it

10. Explain in detail about the Dockers file system? What is UFS?Also explain about the snapshotting FS and VFS and their use case scenarios

11. What is meant by topology awareness? List few products that are topology aware for Rack, zone, region for a Docker based environment

12. **Answer the below simple Questions**

a. Why layers are important in Docker?

b. What are the commands that will create layers?

c. How to list the Docker layers of a Container?

d. Where layers can be physically located?

e. What is aufs and how it is related to layers in Docker?

13. **Answer the below simple Questions**

a. What is the difference between *docker ps* and *dockerps-a* commands?

b. How to list the non-running containers?

c. How to list containers that are using redis image?

14. **Answer the below simple Questions**

a. How to list the containers that exit before last 30 minutes?

b. How to list containers that have ports 80 and 443 published?

c. How to list containers with their commands?

15. **Answer the below simple Questions**

a. How to copy Docker images between hosts? What are the different ways to do the same?

b. Like using volumes what are the otherways to handle persistent data in containers?

c. Explain about docker-default profile and the steps to create a new profile with different set of permissions

16.**Explain the simple commands listed below**

docker diff 1bgh67f789c1b

docker container ls -a

docker container ls

docker container ls –h

docker ps—last 5—quiet

docker ps—latest

docker ps—all—format {{.ID}}

docker ps -s

docker ps -n=-1

docker ps -l

docker ps -aq

docker ps -aq—no-trunc

docker container list

docker ps—format ‘table {{.Names}}\t{{.Image}}’

docker container ls -a -q—filter=name=redis docker ps—filter publish=8080

docker ps—filter expose=80–443/tcp

17.**Explain the below simple commands**

docker images purge

docker image prune -a

docker container prune—filter “until=1h”

docker network prune

docker volume prune—filter “label!=cache”

docker system prune—volumes

docker images -f dangling=true

docker ps -a -f status=exited

docker rm $(docker ps -a -f status=exited -q)

docker rm $(docker ps -a -f status=exited -f status=created -q)
docker system prune—all—force—volumes

docker system prune -a -f –volumes

18. **Explain the below simple commands**

docker rm $(docker ps -a | grep “24 hours ago”)

docker rm $(docker ps -a | grep redis | awk ‘{print $1}’)

docker container ls -f ‘status=exited’ -f ‘status=dead’

docker container rm $(docker container ls -q -f ‘exited=0’)

docker container ls—filter volume=remote-volume –format “table \{{.ID}}\t{{.Mounts}}”

docker container ls—filter volume=/appl/data—format “table \{.ID}}\t{{.Mounts}}”

docker container ls—format “{{.ID}}: {{.Command}}”

19. **Answer the below simple interview Questions**

a. How many Containers you had in your Production?

b. Explain about your Production environment and deployment life cycle?

c. How the services are linked in older versions of docker and in Newer versions?

d. Do you used EE or CE edition?Why?

e. Have you used Plugins for volumes and logging?Explain in detail about their performance and the issues faced.

20. How to set the docker daemon to listen to a TCP socket & what is the importance of port 2376? What is the role of /var/run/docker.sock? Can an application in a Docker container can access the /var/run/docker.sock file?

21. Mention 5 Docker Best Practices similar to this one—Single application per container

22. In the output of docker info command, the below is observed. How to fix this? WARNING: No memory limit support

23. Explain in detail about Docker image forgery, reply attack and how to proactively prevent them? Have you used Notary in your projects?How does signer, client, server interact in Notary?What is a root key and the different type of keys involved in a transaction?

24. Demo how the flatten a Docker image to remove history of layers associated with the image? What are the different ways to do it and why it is important for securing information?

How to remove only selected layers?

Have you used Dockviz?

25. What are data containers or Volume only containers?What are the pros and cons of using them?

Is it possible to access containers data from the host? If yes, what are the different ways to do it?What are named volumes?

26. Have you used tmpfs mount? When to use that?

Will it reside on container/Host memory?How to share them between containers? Is it possible?

For Swarm Services which option is recommended—tmpfs or—mount

Explain the below commands, their difference and purpose

docker run -d—read-only -it—tmpfs /appl/tmp voiptempdata
docker run -d -it—name voiptempdata—mount type=tmpfs,destination=/appl/tmp voipasterix

27. When to use volumes and when to use Bind Mounts?

Explain about Docker Managed volumes & Bind mount volumes? What are their pros and cons?

Also Explain the below commands and their difference

docker run -d—name voipadcon -v /app/main:/app/main voip_asterix docker run -d—name voipadcon -volume admin_data:/app/main voip_asterix

docker run -d—name voipadcon -v admin_data:/app/main:ro voip_asterix docker run -d—name= voipadcon—mount source=admin-vol, destination=/app/main,readonly voipserver

28. **Explain the below commands and their purpose**

docker run -ti—name=voip1 -v voipdata:/datav voipserver

docker run -ti—name=voip2—volumes-from voip1 voipserver

29. How will you see Docker events and filter Docker events for the last 1 hour?

30. When to use docker run command with **–memory-swap=0** option

1. What is Docker?

Criteria	**Docker**	**Virtual Machines**
Use of OS	All containers share host OS	Each VM runs in its own OS
Startup time	Very fast	Slow
Isolation	Process level isolation	Full isolation
security	Low	High

You can define Docker as a containerization platform that combines all your applications in a package so that you have all the dependencies to run your applications in any environment. This means your application will run seamlessly on any environment and this makes it easy for having a product ready application. What Docker does is wrap the software that is needed in a file system that has everything for running the code, providing the runtime and all the necessary libraries and system tools. Containerization technology like Docker will share the same operating system kernel with the machine and due to this it is extremely fast. This means that you have to run the Docker only at the beginning and after that since your OS is already running, you will have a smooth and seamless process.

2. What is the benefit of using a Docker over a hypervisor?

Though Docker and Hypervisor might do the same job overall there are many differences between them in terms of how they work. Docker can be thought of as light weight since it uses very less resources and also the host kernel rather than creating it like a Hypervisor.

3. What are the unique features of Docker over other containerization technology?

Here we list some of the most important and unique features of Docker that makes it a top containerization technology unlike any other in the market today

- You can run your Docker container either on your PC or your enterprise IT system
- Along with the Docker Hub which is a repository of all containers you can deploy and download all your applications from a central location
- You can even share your applications with the containers that you create.

4. What is Docker image?

Here we will be explaining what is the Docker image. The Docker image help to create the Docker containers. You can create the Docker image with the build command, due to this it creates a container that starts when it begins to run. All the docker images are stored in the Docker registry like the public docker registry. These have minimal amounts of layers within the image so that there is minimum amount of data on the network.

5. What is Docker container?

Here we will be discussing what is a Docker container. It is a comprehensive set of applications including all its dependencies which share the same OS kernel along with the other containers running in separate processes within the operating system in a user space. The Docker is not tied to any IT infrastructure and thus it can run on any computer system or the cloud. You can create a Docker container using the Docker images and then running it or you can use the images that are already created in the Docker Hub. To simplify things, let us say that the Docker containers are just runtime instances of the Docker image.

6. What is Docker hub?

You can think of Docker Hub as a cloud registry that lets you link the code repositories, create the images and test them. You can also store your pushed images, or you can link to the Docker Cloud, so that the images can be deployed to the host. You have a centralized container image discovery resource which can be used for collaboration of your teams, automating the workflow, distribution and change management by creating the development pipeline.

7. What is Docker Swarm?

You can think of Docker Swarm as the way of orchestrating the Docker containers. You will be able to implement the Dockers in a cluster. You can convert your Docker pools into a single Docker Swarm for easy management and monitoring.

8. What is the use of Dockerfile?

The Dockerfile can be thought of as a set of instructions that you need to pass on to the Docker so that the images can be built from the specified instructions in the Dockerfile. You can think of the Dockerfile as a text document which has all the commands that are needed for creating a Docker image. You can create an automated build that lets you execute multiple command-lines one after the other.

9. Is it possible to use JSON instead of YAML for Docker compose?

You can use JSON instead of YAML for Docker compose file. So when you are using the JSON file for composing then you have to specify the filename with the following command:

docker-compose -f docker-compose.json up

10. Tell us how you have used Docker in your past position?

This is a question that you could bring upon your whole experience with Docker and if you have used any other Container technologies before Docker. You could also explain the ease that this technology has brought in the automation of the development to production lifecycle management. You can also discuss about any other integrations that you might have worked along with Docker such as Puppet, Chef or even the most popular of all technologies – Jenkins. If you do not have any experience with Docker itself but similar tools from this space, you could convey the same and also show in your interest towards learning this leading containerization technology.

11. What is the process for creating a Docker container?

You can use any of the specific Docker image for creating a Docker container using the below command.

docker run -t -i command name

This command not only creates the container but also will start it for you. If you want to check if the Docker container has been created or not then you need to have the following command which will list all the Docker containers along with the host on which the Docker container runs.

docker ps -a

12. What is the process for stopping and restarting a Docker container?

If you want to stop a Docker container then you need to use the following command:

CONTAINER_ID:

docker stop CONTAINER_ID

If you want to restart a Docker container then you need to use the following command

CONTAINER_ID:

docker restart CONTAINER_ID

13. How do you scale your Docker containers?

The Docker containers can be scaled to any level starting from a few hundreds to even thousands or millions of containers. The only condition is that the containers need the memory and the OS at all times and there should not be a constraint on these when the Docker is getting scaled.

Q2. What is Docker?

I will suggest you to start with a small definition of Docker.

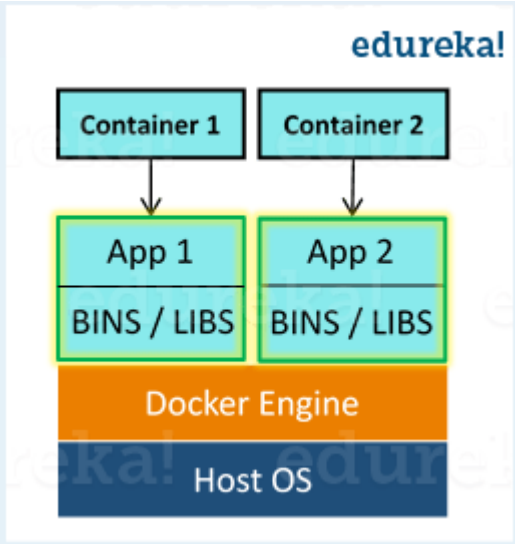
- Docker is a containerization platform which packages your application and all its dependencies together in the form of containers so as to ensure that your

application works seamlessly in any environment be it development or test or production.

- Docker containers, wrap a piece of software in a complete filesystem that contains everything needed to run: code, runtime, system tools, system libraries etc. anything that can be installed on a server.
- This guarantees that the software will always run the same, regardless of its environment.

You can refer the diagram shown below, as you can see that containers run on a single machine share the same operating system kernel, they start instantly as only apps need to start as the kernel is already running and uses less RAM.

Note: Unlike Virtual Machines which has its own OS Docker containers uses the host OS



Next set of Docker interview questions will focus on various components of Docker.

Q3. What is Docker image?

Docker image is the source of Docker container. In other words, Docker images are used to create containers. Images are created with the build command, and they'll produce a container when started with run. Images are stored in a Docker registry such as registry.hub.docker.com because they can become quite large, images are designed to be composed of layers of other images, allowing a minimal amount of data to be sent when transferring images over the network.

Q4. What is Docker container?

Docker containers include the application and all of its dependencies, but share the kernel with other containers, running as isolated processes in user space on the host

operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.
Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.

Docker containers are basically runtime instances of Docker images.

Q5 What is Docker hub?

Docker hub is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images, and links to Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

Q6. How is Docker different from other container technologies?

Docker containers are easy to deploy in a cloud. It can get more applications running on the same hardware than other technologies, it makes it easy for developers to quickly create, ready-to-run containerized applications and it makes managing and deploying applications much easier. You can even share containers with your applications.
If you have some more points to add you can do that but make sure the above the above explanation is there in your answer.

Q7. What is Docker Swarm?

Docker Swarm is native clustering for Docker. It turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.

I will also suggest you to include some supported tools:

- Dokku
- Docker Compose
- Docker Machine
- Jenkins

Q8. What is Dockerfile used for?

This answer, according to me should begin by explaining the use of Dockerfile.

Docker can build images automatically by reading the instructions from a Dockerfile.

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.

Q9. Can I use json instead of yaml for my compose file in Docker?

You can use json instead of yaml for your compose file, to use json file with compose, specify the filename to use for

eg:
docker-compose -f docker-compose.json up

Q10. Tell us how you have used Docker in your past position?

Explain how you have used Docker to help rapid deployment. Explain how you have scripted Docker and used Docker with other tools like Puppet, Chef or Jenkins.

If you have no past practical experience in Docker and have past experience with other tools in a similar space, be honest and explain the same. In this case, it makes sense if you can compare other tools to Docker in terms of functionality.

Q11. How to create Docker container?

I will suggest you to give a direct answer to this.

We can use Docker image to create Docker container by using the below command:

```
1                    docker run -t -i command name
```

This command will create and start a container.

You should also add, If you want to check the list of all running container with the status on a host use the below command:

```
1                    docker ps -a
```

Q12. How to stop and restart the Docker container?

In order to stop the Docker container you can use the below command:

```
1                    docker stop container ID
```

Now to restart the Docker container you can use:

```
1                    docker restart container ID
```

Q13 How far do Docker containers scale?

Large web deployments like Google and Twitter, and platform providers such as Heroku and dotCloud all run on container technology, at a scale of hundreds of thousands or even millions of containers running in parallel.

Q14. What platforms does Docker run on?

I will start this answer by saying Docker runs on only Linux and Cloud platforms and then I will mention the below vendors of Linux:

- Ubuntu 12.04, 13.04 et al
- Fedora 19/20+
- RHEL 6.5+
- CentOS 6+
- Gentoo
- ArchLinux
- openSUSE 12.3+
- CRUX 3.0+

Cloud:

- Amazon EC2
- Google Compute Engine
- Microsoft Azure
- Rackspace

Note that Docker does not run on Windows or Mac.

Q15. Do I lose my data when the Docker container exits?

You can answer this by saying, no I won't lose my data when Docker container exits, any data that your application writes to disk gets preserved in its container until you explicitly delete the container. The file system for the container persists even after the container halts.

Q16. Mention some commonly used Docker command?

Below are some commonly used Docker commands:

Command	Description
dockerd	Launch the Docker daemon
info	Display system-wide information
inspect	Return low-level information on a contair
version	Show the Docker version information
build	Build an image from a Dockerfile
commit	Create a new image from a container's ch
history	Show the history of an image
load	Load an image from a tar archive or STDI
attach	Attach to a running container
create	Create a new container
diff	Inspect changes on a container's filesyste
kill	Kill a running container