Docker & Kubernetes Questions

Beginner-Level Docker & Kubernetes Questions (1-20)

Docker Basics

1. What is Docker, and why is it used?

Docker is a containerization platform that allows developers to package applications along with their dependencies into a single unit called a container.

Why use Docker?
  Ensures consistent environments across different machines.
  Lightweight & faster than virtual machines.
  Easy scaling of applications in microservices architectures.

2. What is the difference between Docker and a Virtual Machine (VM)?

Docker:

Isolation- Uses containers to isolate apps
Performance- Faster, lightweight
Startup Time- Milliseconds
Use Case- Ideal for microservices

Virtual Machine:

Isolation- Uses hypervisor to run separate OS instances
Performance- Slower, resource-intensive
Startup Time- Minutes
Use Case- Best for full OS emulation

3. What is a Docker image?

A Docker image is a read-only template containing everything needed to run an application, including:

Source code
Libraries & dependencies
Configuration files

A container is created from a Docker image using the docker run command.

4. What is a Docker container?

A Docker container is a running instance of a Docker image. It is:
  Lightweight (shares OS kernel)
  Isolated (has its own filesystem, network, and process space)
  Portable (can run on any system with Docker installed)

5. How do you create and run a Docker container?

To run a container from an image:

```
docker run -d --name myapp nginx
```

-d: Run in detached mode (background).
--name myapp: Name the container myapp.
nginx: Use the nginx image.

6. What is the purpose of the Dockerfile?

A Dockerfile is a script that contains instructions to build a Docker image.

Example Dockerfile:

```
FROM node:16
WORKDIR /app
COPY . .
RUN npm install
CMD ["node", "app.js"]
```

FROM: Base image.
WORKDIR: Set working directory.
COPY: Copy files.
RUN: Execute commands (install dependencies).
CMD: Define the default command to run.

7. What are Docker volumes?

Docker volumes store persistent data outside a container’s filesystem.

Types:
Anonymous Volumes: docker run -v /data nginx
Named Volumes: docker volume create mydata
Bind Mounts: docker run -v /host/path:/container/path nginx

8. How do you list running Docker containers?

Use the command:
```
docker ps
```

To list all containers (including stopped ones):
```
docker ps -a
```

9. What is Docker Compose?

Docker Compose is a tool for defining and running multi-container applications.

Example docker-compose.yml:

```
version: "3"
```

```
services:
  web:
    image: nginx
    ports:
      - "80:80"
  db:
    image: mysql
    environment:
      MYSQL_ROOT_PASSWORD: root
```

Start with: `docker-compose up -d`

Stop with: `docker-compose down`

10. What is the difference between CMD and ENTRYPOINT in Docker?

CMD:

Purpose- Default command
Overridable?- Yes
Example- CMD ["python", "app.py"]

ENTRYPOINT:

Purpose- Fixed executable command
Overridable?- No (unless --entrypoint is used)
Example- ENTRYPOINT ["nginx", "-g", "daemon off;"]

Kubernetes Basics

11. What is Kubernetes?

Kubernetes (K8s) is an orchestration platform for managing containerized applications.

Features:
  Automated scaling
  Self-healing (restarts failed containers)
  Load balancing
  Rolling updates

12. What is a Kubernetes Pod?

A Pod is the smallest unit in Kubernetes. It groups one or more containers that share the same network and storage.

13. What is a Kubernetes Deployment?

A Deployment manages Pod creation and updates.

Example YAML:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: app
          image: nginx
```

replicas: 3 → Runs 3 instances.
matchLabels → Ensures the correct Pods are managed.

14. What is a Kubernetes Service?

A Service exposes a set of Pods over a network.

Types:
ClusterIP (default)
NodePort (exposes on a fixed port)
LoadBalancer (uses cloud provider’s load balancer)

Example YAML:

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: my-app
  ports:
    - port: 80
      targetPort: 8080
      nodePort: 30007
```

15. What is the purpose of Kubernetes ConfigMaps and Secrets?

ConfigMaps store non-sensitive configuration data.

Secrets store sensitive data (passwords, API keys).

Example Secret:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
data:
  password: bXlwYXNzd29yZA==
```

16. What is a Kubernetes Namespace?

Namespaces logically separate resources within a cluster.

```
kubectl create namespace dev
kubectl get namespaces
```

17. What is a StatefulSet in Kubernetes?

A StatefulSet is used for stateful applications like databases. Unlike Deployments, it maintains:
  Stable pod identity
  Persistent storage

18. How do you scale a Deployment in Kubernetes?

Manually scale using:

```
kubectl scale deployment my-app --replicas=5
```

19. What is a DaemonSet?

A DaemonSet ensures that one Pod runs on every node (e.g., logging agents, monitoring).

20. How do you update a Kubernetes Deployment?

Update the image and apply changes:

```
kubectl set image deployment/my-app my-container=nginx:latest
```

Intermediate-Level Docker & Kubernetes Questions (21-40)

Docker Intermediate Questions

21. What is the difference between Docker ADD and COPY?

ADD:

Function- Copies files & extracts compressed files
Supports URLs?- Yes

Best Practice- Use for archives (.tar.gz)

COPY:

Function- Copies files only
Supports URLs?- No
Best Practice- Use for simple file copies

Example:

```
COPY config.json /app/config.json
ADD myapp.tar.gz /app/
```

22. How do you optimize Docker images?

Use smaller base images (e.g., alpine instead of ubuntu).

Multi-stage builds to reduce image size:

```
FROM node:16 AS build
WORKDIR /app
COPY . .
RUN npm install && npm run build

FROM nginx:alpine
COPY --from=build /app/dist /usr/share/nginx/html
```

Use .dockerignore to avoid unnecessary files.

23. What is the difference between Docker ENTRYPOINT and CMD?

ENTRYPOINT is not overridden by command-line arguments, while CMD can be.

Best practice: Use ENTRYPOINT for fixed commands.

Example:

```
ENTRYPOINT ["nginx", "-g", "daemon off;"]
CMD ["-p", "80"]
```

24. How do you debug a running Docker container?

Get container logs: `docker logs my-container`
Attach to a running container: `docker exec -it my-container /bin/sh`
Inspect container details: `docker inspect my-container`

25. What is a Docker Multi-Stage Build?

A multi-stage build reduces image size by using multiple FROM statements.

```
FROM golang:1.17 AS builder
WORKDIR /app
COPY . .
RUN go build -o myapp

FROM alpine
COPY --from=builder /app/myapp /myapp
ENTRYPOINT ["/myapp"]
```

The final image only contains the built binary.

26. How does Docker handle networking?

Bridge network (default): Containers communicate via virtual network.
Host network: Container shares the host’s networking stack.
Overlay network: Used in Docker Swarm for multi-host networking.

Example:

```
docker network create mynetwork
docker run --network=mynetwork nginx
```

27. What is the difference between Docker Swarm and Kubernetes?

Docker Swarm:

Orchestration- Lightweight, built into Docker
Scaling- Manual
Service Discovery- Built-in

Kubernetes:

Orchestration- Advanced, feature-rich
Scaling- Auto-scaling
Service Discovery- Needs external setup (DNS, Ingress)

28. How do you remove unused Docker images and containers?

```
docker system prune -a
```

This removes stopped containers, unused networks, and dangling images.

29. What is Docker BuildKit?

Docker BuildKit improves build speed and caching.

Enable it with:

```
DOCKER_BUILDKIT=1 docker build .
```

Benefits:

Faster builds
Parallel execution
Improved caching

30. How do you limit container resource usage?

Use --memory and --cpus:

```
docker run --memory=512m --cpus=1 nginx
```

This limits memory to 512MB and CPU usage to 1 core.

Kubernetes Intermediate Questions

31. How does Kubernetes handle high availability?

Uses multiple master nodes to avoid single points of failure.
Deployments use replica sets to keep applications running.
Load balancing & failover mechanisms ensure availability.

32. What is the role of kubelet in Kubernetes?

Kubelet runs on each node and:
Communicates with the master node
Ensures containers are running
Monitors container health

33. How do you check logs of a running Pod in Kubernetes?

```
kubectl logs my-pod
kubectl logs -f my-pod  # Stream logs in real-time
```

34. What are Kubernetes Labels and Selectors?

Labels identify resources, while selectors filter resources.

Example:

```
metadata:
  labels:
    app: my-app
```

To filter pods by label:

```
kubectl get pods -l app=my-app
```

35. What is a Kubernetes Ingress?

An Ingress manages external access to services.

Example:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
    - host: myapp.com
      http:
        paths:
          - path: /
            backend:
              service:
                name: my-service
                port:
                  number: 80
```

Use Ingress controllers (NGINX, Traefik) to manage Ingress resources.

36. What is the difference between Horizontal Pod Autoscaler (HPA) and Vertical Pod Autoscaler (VPA)?

HPA:

Scaling Type- Adds/removes pods
Use Case- High traffic apps

VPA:

Scaling Type- Adjusts CPU/memory of existing pods
Use Case- Resource optimization

Example of HPA:

```
kubectl autoscale deployment my-app --cpu-percent=50 --min=2 --max=10
```

37. What is a Kubernetes Persistent Volume (PV) and Persistent Volume Claim (PVC)?

A Persistent Volume (PV) is a storage resource, and a Persistent Volume Claim (PVC) requests storage.

Example:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

38. How do you upgrade a running application in Kubernetes?

Modify the image and apply the deployment:

```
kubectl set image deployment/my-app my-container=nginx:1.20
kubectl rollout status deployment my-app
```

39. What is a Kubernetes Job and CronJob?

Job: Runs once and exits.
CronJob: Runs on a schedule (like a Linux cron).

Example:

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: my-cronjob
spec:
  schedule: "0 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: hello
              image: busybox
              command: ["echo", "Hello from Kubernetes"]
          restartPolicy: OnFailure
```

40. How do you debug Kubernetes pods that are stuck in "CrashLoopBackOff"?

Check pod logs:
```
kubectl logs my-pod
```

Describe the pod for errors:
```
kubectl describe pod my-pod
```

Exec into the container:
```
kubectl exec -it my-pod -- /bin/sh
```

Advanced-Level Docker & Kubernetes Questions (41-60)

Docker Advanced Questions

41. What are Docker namespaces and cgroups? How do they contribute to containerization?

Namespaces isolate resources (PID, network, mount points, etc.) for each container.

Cgroups (Control Groups) limit CPU, memory, and disk usage.

Together, they ensure process isolation and resource allocation.

Example:

```
cat /proc/self/cgroup
```

42. What is the difference between Docker Volumes, Bind Mounts, and tmpfs?

Volumes:

Type- Volumes
Persistent?- Yes
Use Case- Best for data persistence

Bind Mounts:

Type- Bind Mounts
Persistent?- Yes
Use Case- Direct host file access

tmpfs:

Type- tmpfs
Persistent?- No
Use Case- In-memory storage for performance

Example (Volume):

```
docker run -v myvolume:/data nginx
```

43. What are Docker BuildKit advantages?

Parallel execution speeds up builds.
Efficient caching reduces rebuild time.
Security improvements via secret mounts.

Enable BuildKit:

```
DOCKER_BUILDKIT=1 docker build .
```

44. How do you secure a Docker container?

Use minimal base images (e.g., alpine).
Run as non-root user.
Limit container capabilities (--cap-drop=ALL).
Use read-only filesystems (--read-only).

Example:

```
docker run --user 1001 --read-only nginx
```

45. How do multi-stage builds improve security in Docker?

Keeps sensitive files out of the final image.
Reduces attack surface by discarding unnecessary dependencies.

Example:

```
FROM golang AS build
COPY . .
RUN go build -o myapp

FROM alpine
COPY --from=build /myapp /myapp
ENTRYPOINT ["/myapp"]
```

46. What are immutable infrastructure principles, and how do they apply to Docker?

Containers should be replaced, not modified.
Use image versioning instead of patching live containers.

Example: Deploy new image versions instead of updating running containers.

47. How does Docker Content Trust (DCT) improve security?

Ensures image integrity with digital signatures.

Enable DCT:

```
export DOCKER_CONTENT_TRUST=1
```

48. How do you troubleshoot a Docker daemon issue?

Check logs: journalctl -u docker.service
Restart service: systemctl restart docker
Debug mode: dockerd --debug

49. What is the difference between Docker stack and Docker compose?

Docker Compose is for single-host deployments.

Docker Stack is for multi-node Swarm clusters.

50. How do you handle container networking in a multi-host Docker Swarm?

Overlay networks span multiple hosts.

Example:

```
docker network create -d overlay mynetwork
```

Kubernetes Advanced Questions

51. How does Kubernetes handle stateful applications?

Uses StatefulSets instead of Deployments.
Provides stable network identities and persistent storage.

Example:

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: "mysql"
  replicas: 3
```

52. What are PodDisruptionBudgets (PDBs)?

Ensures minimum availability during voluntary disruptions.

Example:

```
apiVersion: policy/v1
kind: PodDisruptionBudget
metadata:
  name: my-pdb
spec:
  minAvailable: 2
  selector:
    matchLabels:
      app: my-app
```

53. How do you secure Kubernetes Secrets?

Use encryption at rest.

Store secrets in external vaults (e.g., HashiCorp Vault).

Example:

```
kubectl create secret generic db-secret --from-literal=password=mysecurepassword
```

54. What are Kubernetes Admission Controllers?

They intercept API requests before they reach the cluster.

Example: PodSecurityPolicies, ValidatingWebhookConfiguration.

55. How does Kubernetes handle node failures?

Kubelet marks node as NotReady.
Pods are rescheduled onto healthy nodes.
Node auto-repair triggers in cloud-managed clusters.

56. What is a Kubernetes Mutating Webhook?

Modifies requests dynamically before they reach the cluster.

Example: Injecting sidecars into Pods.

57. How do you debug networking issues in Kubernetes?

Check Pod-to-Pod connectivity:
```
kubectl exec -it pod1 -- ping pod2
```

Inspect network policies:
```
kubectl get networkpolicy
```

Validate DNS resolution:
```
kubectl exec -it pod -- nslookup my-service
```

58. How does Kubernetes Horizontal Pod Autoscaler (HPA) work internally?

Uses metrics API (CPU/memory usage).

Adjusts replica count dynamically.

Example:

```
kubectl autoscale deployment my-app --cpu-percent=50 --min=2 --max=10
```

59. How do you implement multi-tenancy in Kubernetes?

Use Namespaces to isolate workloads.

Apply RBAC (Role-Based Access Control).

Example:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: team-a
  name: team-a-role
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
```

60. What is Kubernetes Cluster Federation?

Manages multiple clusters as a single entity.

Benefits: Cross-region high availability, policy consistency.

Example tool: kubefed