Linux-Foundation

Exam Questions KCNA

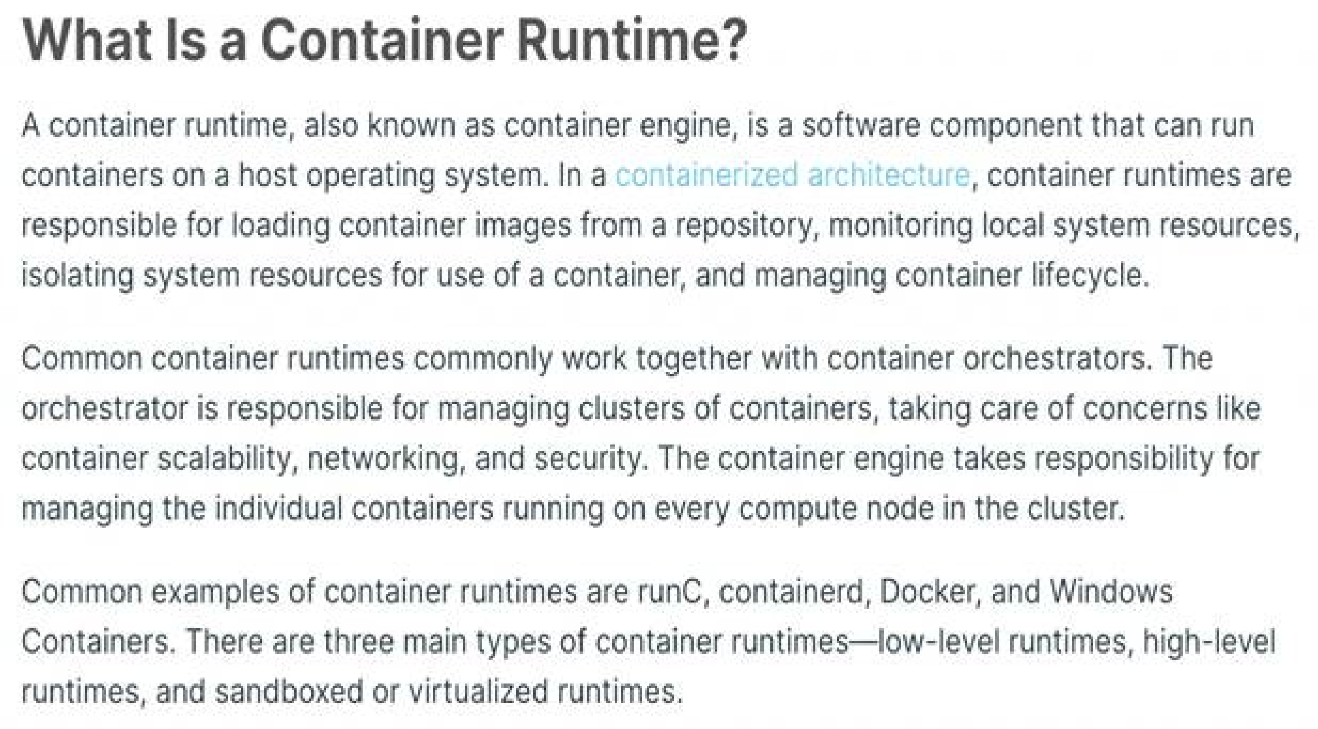
Kubernetes and Cloud Native Associate (KCNA)

NEW QUESTION 1
What is container runtime?

A. The amount of time it takes a container to execute
B. A container image format
C. Another term of kubelet or kubectl
D. Software that runs containers

Answer: D

Explanation:
https://www.aquasec.com/cloud-native-academy/container-security/container-runtime/ Text Description automatically generated



What Is a Container Runtime?

A container runtime, also known as container engine, is a software component that can run containers on a host operating system. In a containerized architecture, container runtimes are responsible for loading container images from a repository, monitoring local system resources, isolating system resources for use of a container, and managing container lifecycle.

Common container runtimes commonly work together with container orchestrators. The orchestrator is responsible for managing clusters of containers, taking care of concerns like container scalability, networking, and security. The container engine takes responsibility for managing the individual containers running on every compute node in the cluster.

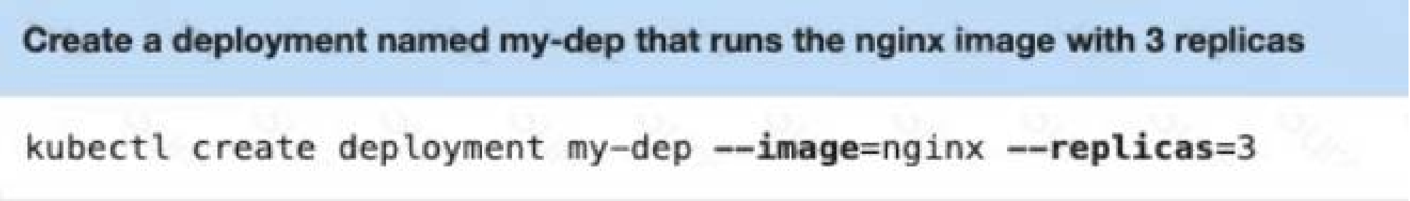
Common examples of container runtimes are runC, containerd, Docker, and Windows Containers. There are three main types of container runtimes—low-level runtimes, high-level runtimes, and sandboxed or virtualized runtimes.

NEW QUESTION 2
How to create deployment name app-dep, image=nginx, and replicas 5 using imperative command?

A. kubectl create app-dep deployment --image=nginx --replicas=5
B. kubectl create deployment app-dep --image=nginx --replicas=5
C. kubectl create app-dep deployment --replicas=5 --image=nginx

Answer: B

Explanation:
https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#-em-deployment-em- Text Description automatically generated with medium confidence



Create a deployment named my-dep that runs the nginx image with 3 replicas

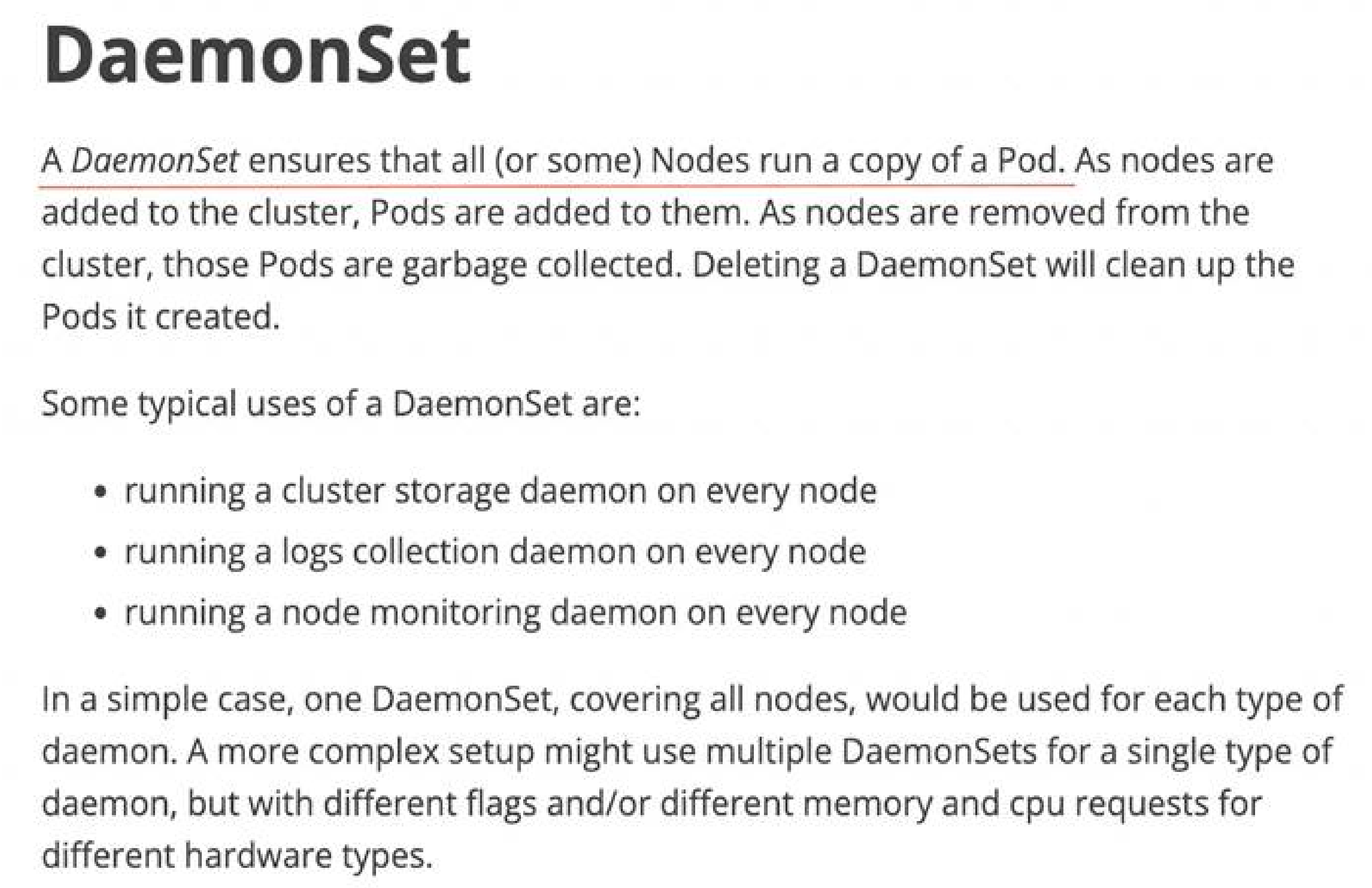
```
kubectl create deployment my-dep --image=nginx --replicas=3
```

NEW QUESTION 3
What Kubernetes resource would allow you to run one Pod on some of your Nodes?

A. DaemonSet
B. ClusterSet
C. Deployment
D. ReplicaSet

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/workloads/controllers/daemonset/ Graphical user interface, text, application Description automatically generated



DaemonSet

A *DaemonSet* ensures that all (or some) Nodes run a copy of a Pod. As nodes are added to the cluster, Pods are added to them. As nodes are removed from the cluster, those Pods are garbage collected. Deleting a DaemonSet will clean up the Pods it created.

Some typical uses of a DaemonSet are:

- running a cluster storage daemon on every node
- running a logs collection daemon on every node
- running a node monitoring daemon on every node

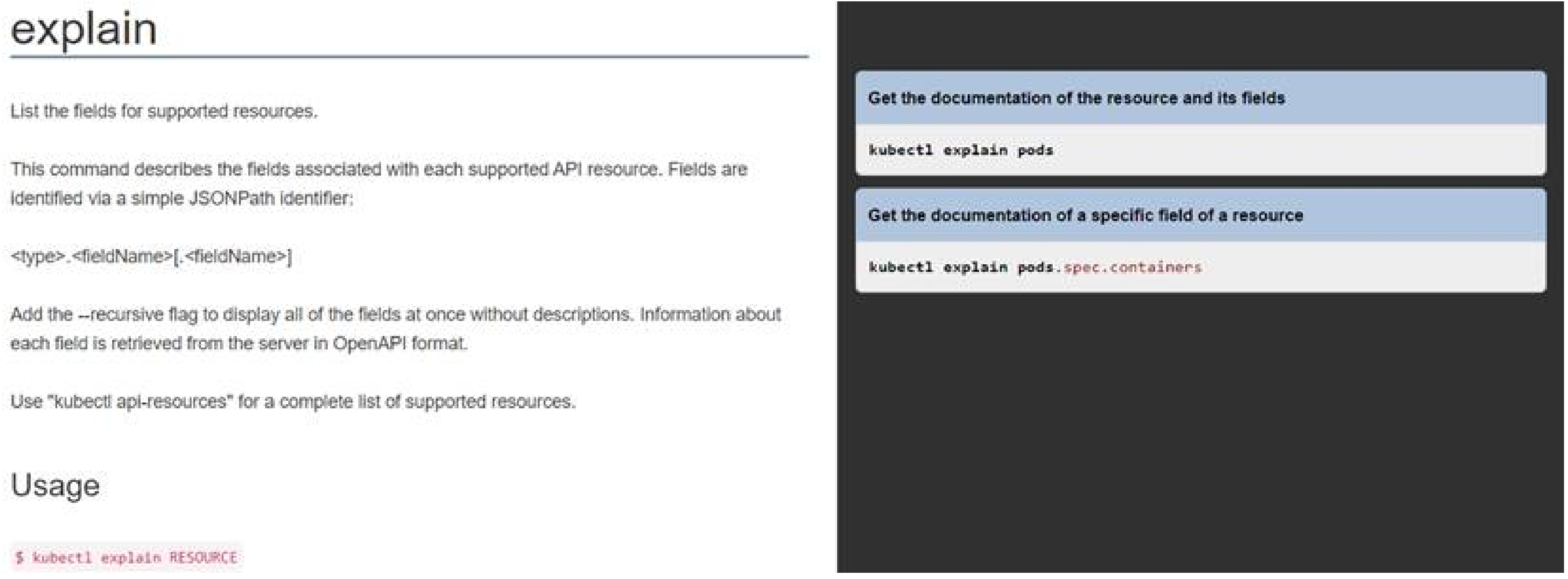
In a simple case, one DaemonSet, covering all nodes, would be used for each type of daemon. A more complex setup might use multiple DaemonSets for a single type of daemon, but with different flags and/or different memory and cpu requests for different hardware types.

NEW QUESTION 4
What command use to get documentation about kubernetes resource type

A. alias k='kubectl' k api-resources
B. alias k='kubectl' k api-list
C. alias k='kubectl' k explain
D. alias k='kubectl' k get resource

Answer: C

Explanation:
https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#explain
Graphical user interface, text, application, email Description automatically generated



explain

List the fields for supported resources.

This command describes the fields associated with each supported API resource. Fields are identified via a simple JSONPath identifier:

<type>.<fieldName>[.<fieldName>]

Add the --recursive flag to display all of the fields at once without descriptions. Information about each field is retrieved from the server in OpenAPI format.

Use "kubectl api-resources" for a complete list of supported resources.

Usage

```
$ kubectl explain RESOURCE
```

Get the documentation of the resource and its fields

```
kubectl explain pods
```

Get the documentation of a specific field of a resource

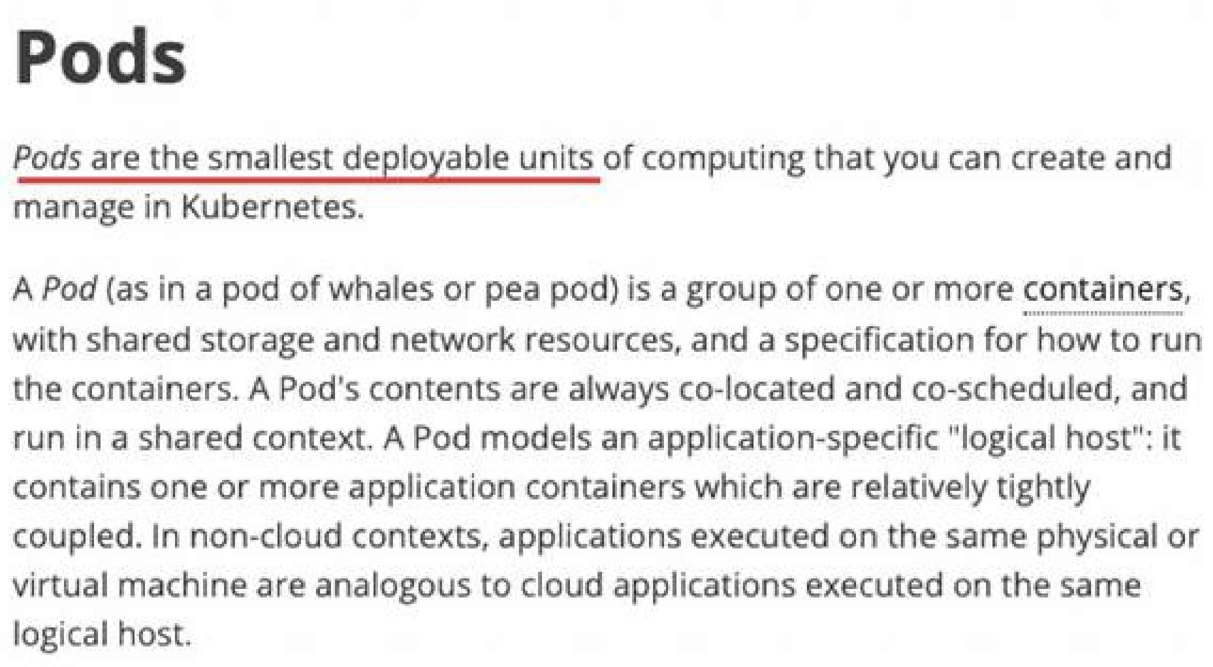
```
kubectl explain pods.spec.containers
```

NEW QUESTION 5
What is the smallest possible unit in Kubernetes to run a container?

A. pod
B. docker
C. service
D. container

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/workloads/pods/
Graphical user interface, text, application Description automatically generated



Pods

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

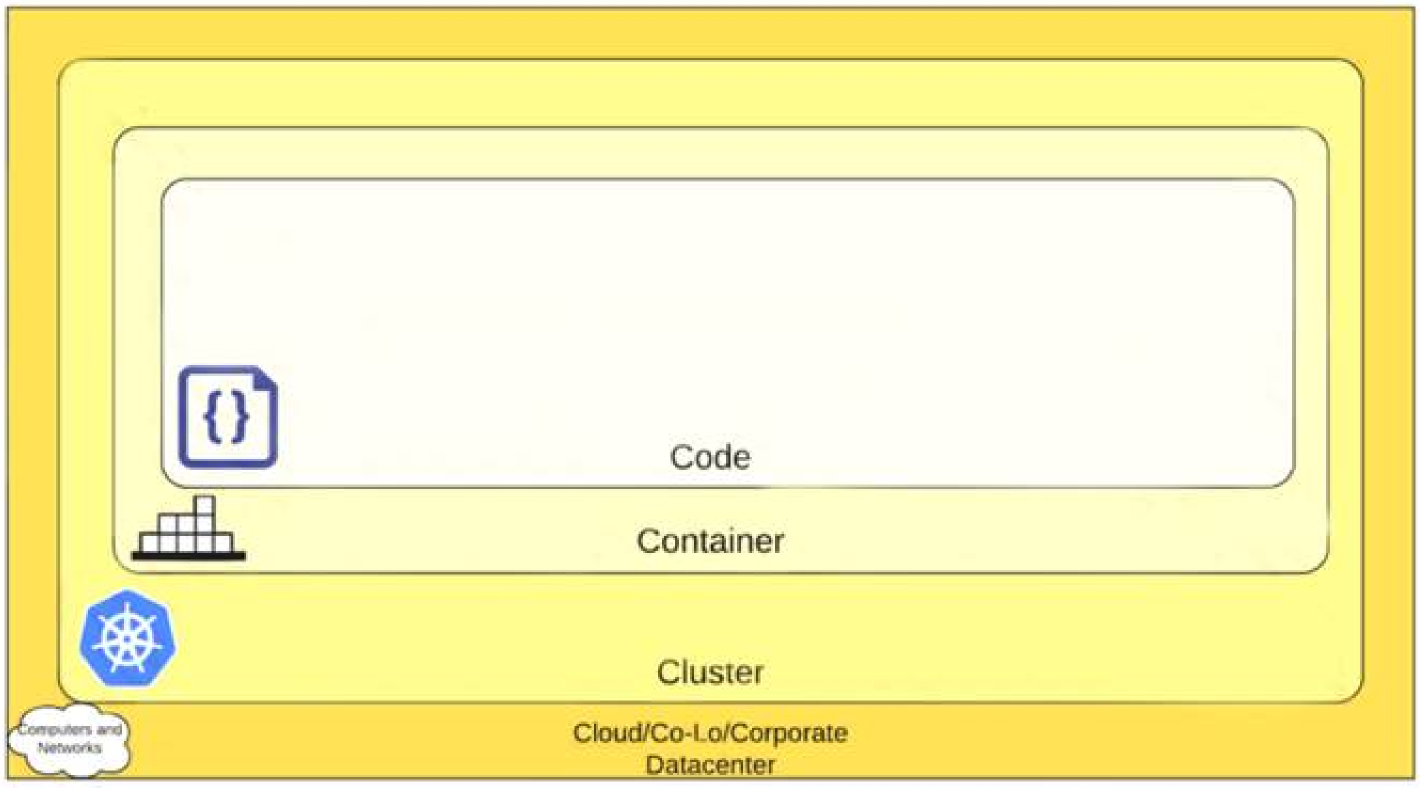
A *Pod* (as in a pod of whales or pea pod) is a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers. A Pod's contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific "logical host": it contains one or more application containers which are relatively tightly coupled. In non-cloud contexts, applications executed on the same physical or virtual machine are analogous to cloud applications executed on the same logical host.

NEW QUESTION 6
There are three Nodes in a cluster, and want to run exactly one replica of a Pod on each Node. Pre-fer to automatically create a replica on any new Nodes when they are added. Which Kubernetes re-source should you use?

A. DaemonSet
B. ReplicaSet
C. NodeSet
D. StatefulSet
E. Deployment

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/workloads/controllers/daemonset/
A DaemonSet runs replicas on all (or just some) Nodes in the cluster.
Table Description automatically generated with medium confidence



NEW QUESTION 7
What standard does kubelet use to communicate with the container runtime?

A. Service Mesh Interface (SMI)
B. CRI-O
C. ContainerD
D. Container Runtime Interface (CRI)

Answer: D

Explanation:
kubelet can communicate with any runtime that supports the CRI standard.

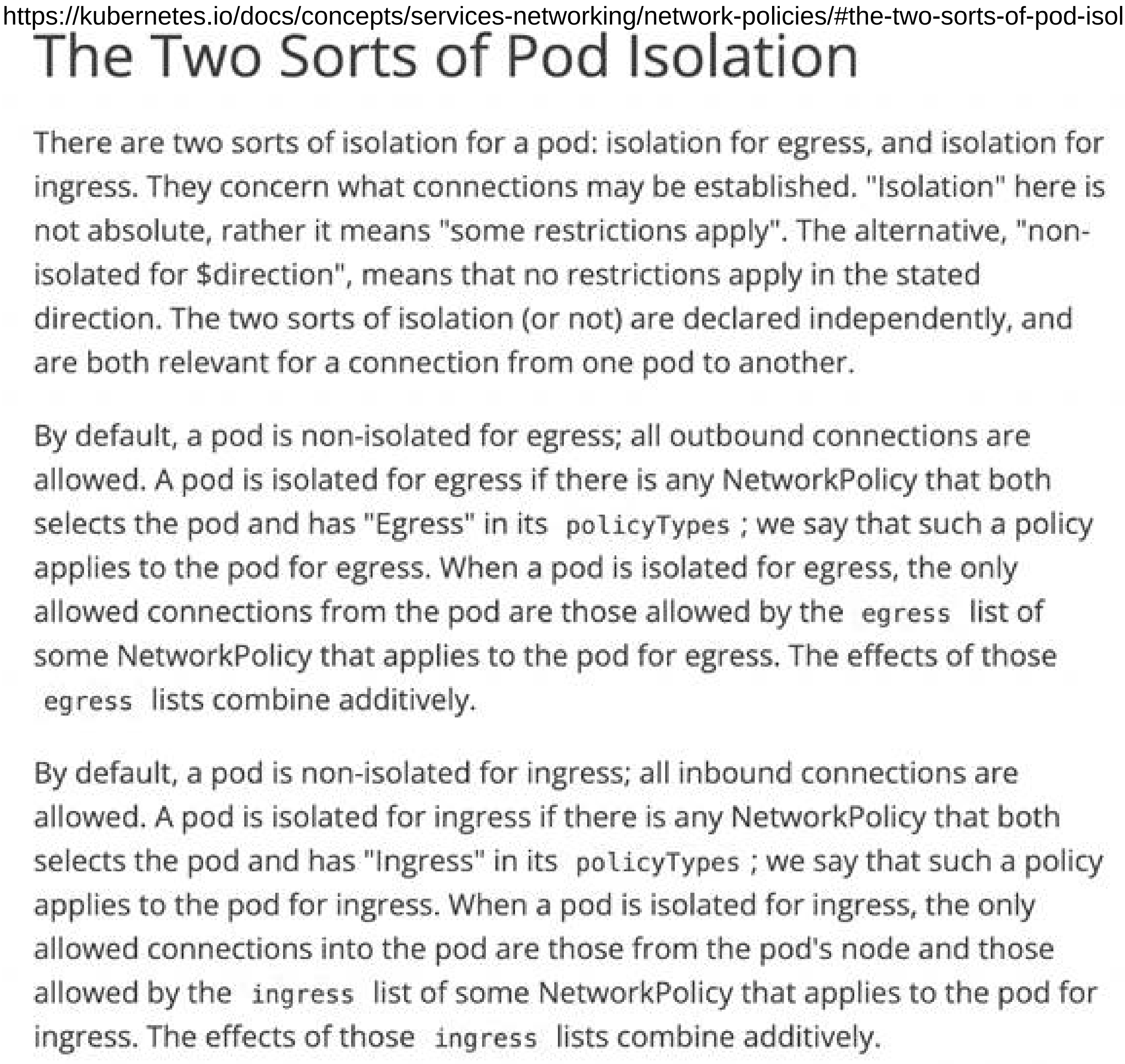
NEW QUESTION 8
Which kubernetes resource type allows defining which pods are isolated when it comes to network-ing?

A. Network policy

B. Domain Name System 'DNS'
C. Role Binding
D. Service

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/services-networking/network-policies/#the-two-sorts-of-pod-isolation Text, letter Description automatically generated



The Two Sorts of Pod Isolation

There are two sorts of isolation for a pod: isolation for egress, and isolation for ingress. They concern what connections may be established. "Isolation" here is not absolute, rather it means "some restrictions apply". The alternative, "non-isolated for $direction", means that no restrictions apply in the stated direction. The two sorts of isolation (or not) are declared independently, and are both relevant for a connection from one pod to another.

By default, a pod is non-isolated for egress; all outbound connections are allowed. A pod is isolated for egress if there is any NetworkPolicy that both selects the pod and has "Egress" in its `policyTypes` ; we say that such a policy applies to the pod for egress. When a pod is isolated for egress, the only allowed connections from the pod are those allowed by the `egress` list of some NetworkPolicy that applies to the pod for egress. The effects of those `egress` lists combine additively.

By default, a pod is non-isolated for ingress; all inbound connections are allowed. A pod is isolated for ingress if there is any NetworkPolicy that both selects the pod and has "Ingress" in its `policyTypes` ; we say that such a policy applies to the pod for ingress. When a pod is isolated for ingress, the only allowed connections into the pod are those from the pod's node and those allowed by the `ingress` list of some NetworkPolicy that applies to the pod for ingress. The effects of those `ingress` lists combine additively.

NEW QUESTION 9
Which of the following is used to request storage in Kubernetes?

A. PersistentVolume 'PV'
B. PersistentVolumeClaim 'PVC'
C. Container Storage Interface 'CSI'
D. StorageClasses

Answer: B

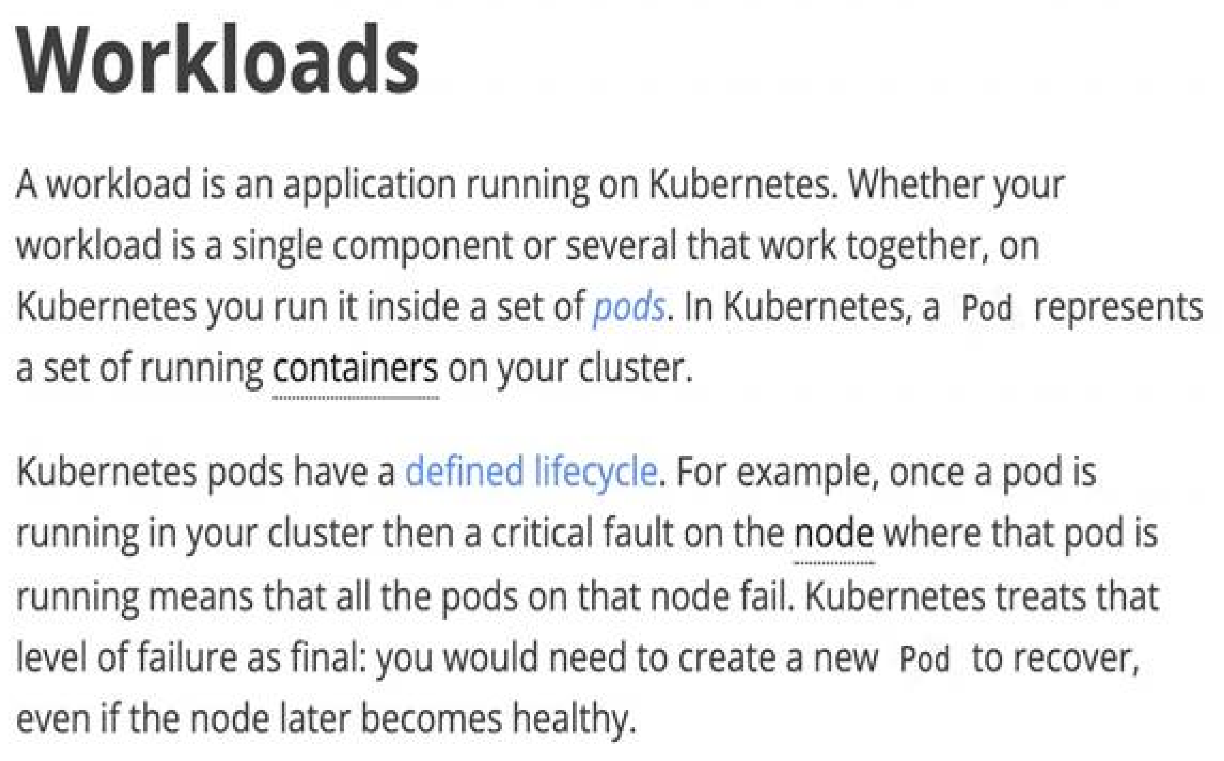
Explanation:
https://kubernetes.io/docs/concepts/storage/persistent-volumes/

NEW QUESTION 10
A ________ is an application running on kubernetes.

A. node
B. pod
C. workload
D. container

Answer: C

Explanation:
https://kubernetes.io/docs/concepts/workloads/ Text Description automatically generated



Workloads

A workload is an application running on Kubernetes. Whether your workload is a single component or several that work together, on Kubernetes you run it inside a set of *pods*. In Kubernetes, a `Pod` represents a set of running containers on your cluster.

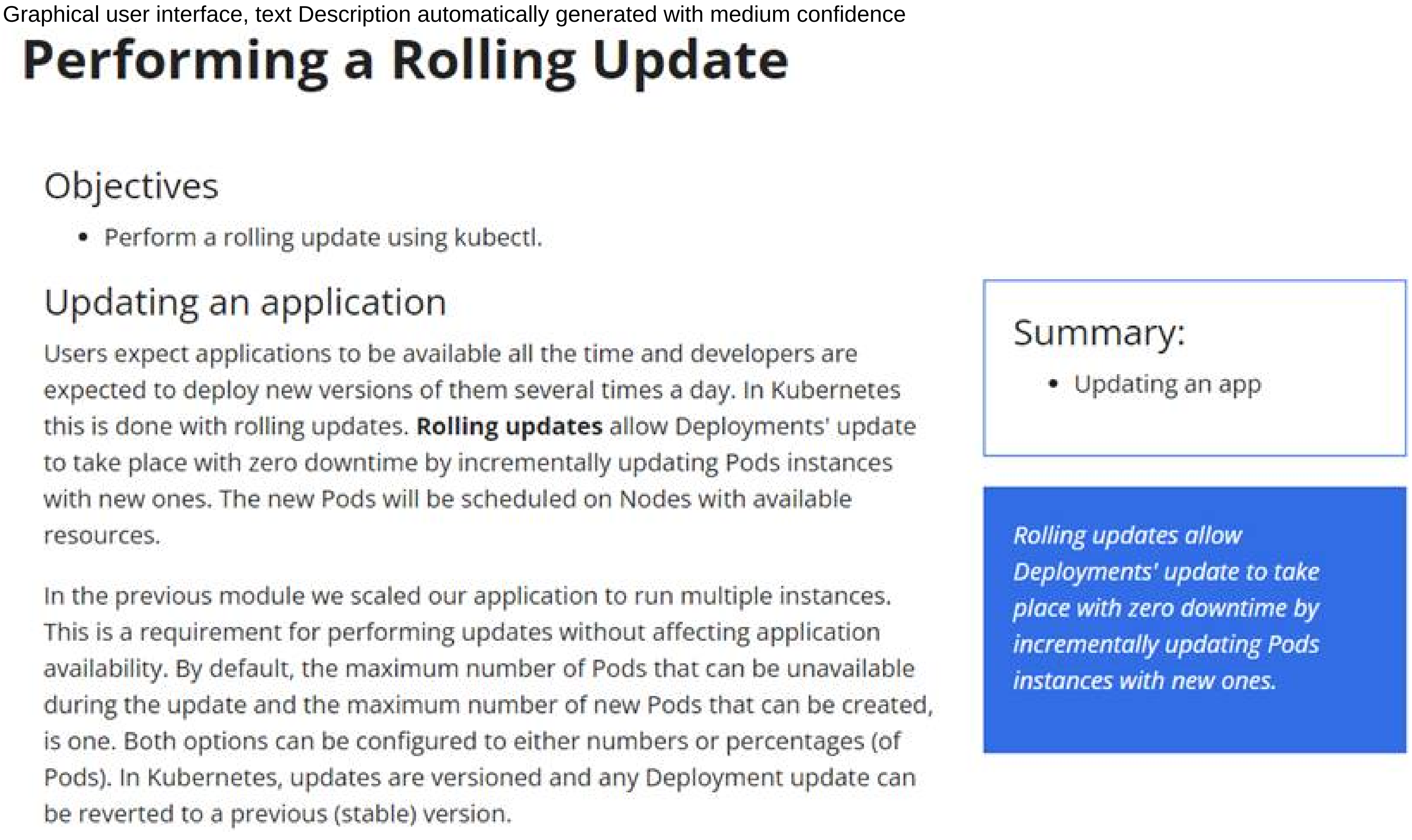
Kubernetes pods have a defined lifecycle. For example, once a pod is running in your cluster then a critical fault on the node where that pod is running means that all the pods on that node fail. Kubernetes treats that level of failure as final: you would need to create a new `Pod` to recover, even if the node later becomes healthy.

NEW QUESTION 10
The Kubernetes rolling update is used for _______ .

A. Updating a service
B. Scaling an application
C. Updating a deployment

Answer: C

Explanation:
https://kubernetes.io/docs/tutorials/kubernetes-basics/update/update-intro/
Graphical user interface, text Description automatically generated with medium confidence



Performing a Rolling Update

Objectives

- Perform a rolling update using kubectl.

Updating an application

Users expect applications to be available all the time and developers are expected to deploy new versions of them several times a day. In Kubernetes this is done with rolling updates. **Rolling updates** allow Deployments' update to take place with zero downtime by incrementally updating Pods instances with new ones. The new Pods will be scheduled on Nodes with available resources.

In the previous module we scaled our application to run multiple instances. This is a requirement for performing updates without affecting application availability. By default, the maximum number of Pods that can be unavailable during the update and the maximum number of new Pods that can be created, is one. Both options can be configured to either numbers or percentages (of Pods). In Kubernetes, updates are versioned and any Deployment update can be reverted to a previous (stable) version.

Summary:

- Updating an app

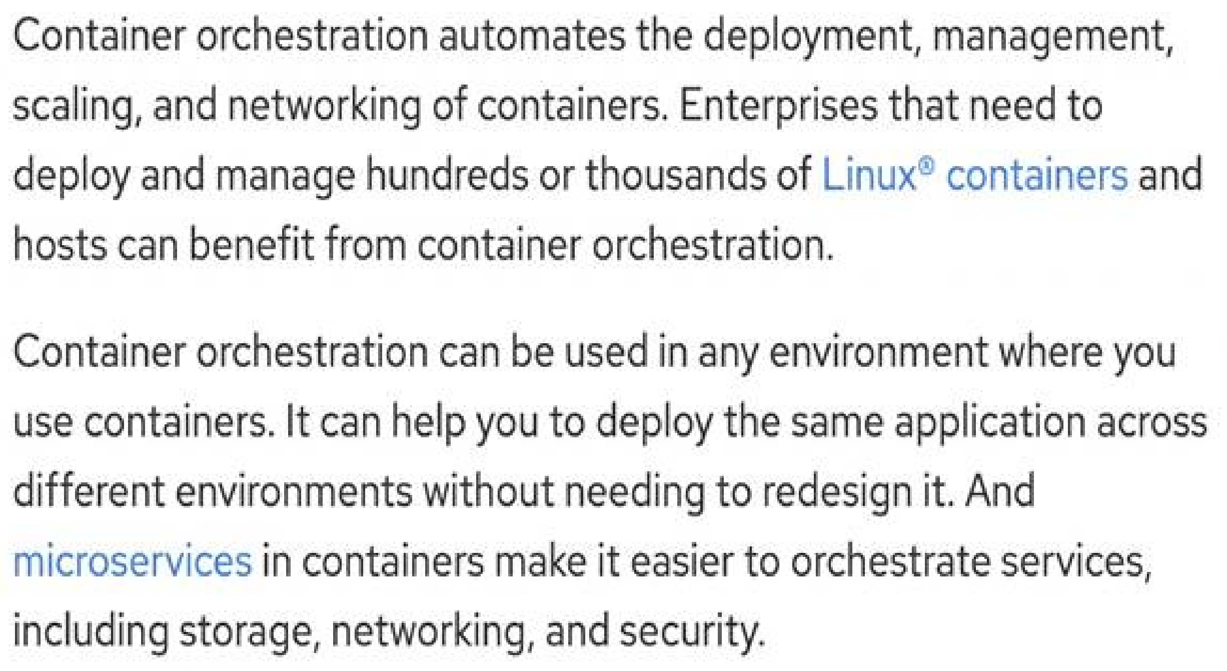
Rolling updates allow Deployments' update to take place with zero downtime by incrementally updating Pods instances with new ones.

NEW QUESTION 11
What is container orchestration?

A. Packaging code and all of its dependencies into a single executable
B. Adding code to a container image so it can run as a container
C. Using automation to manage containers
D. Spinning a new containers to replace old ones

Answer: C

Explanation:
https://www.redhat.com/en/topics/containers/what-is-container-orchestration Text Description automatically generated



Container orchestration automates the deployment, management, scaling, and networking of containers. Enterprises that need to deploy and manage hundreds or thousands of Linux® containers and hosts can benefit from container orchestration.

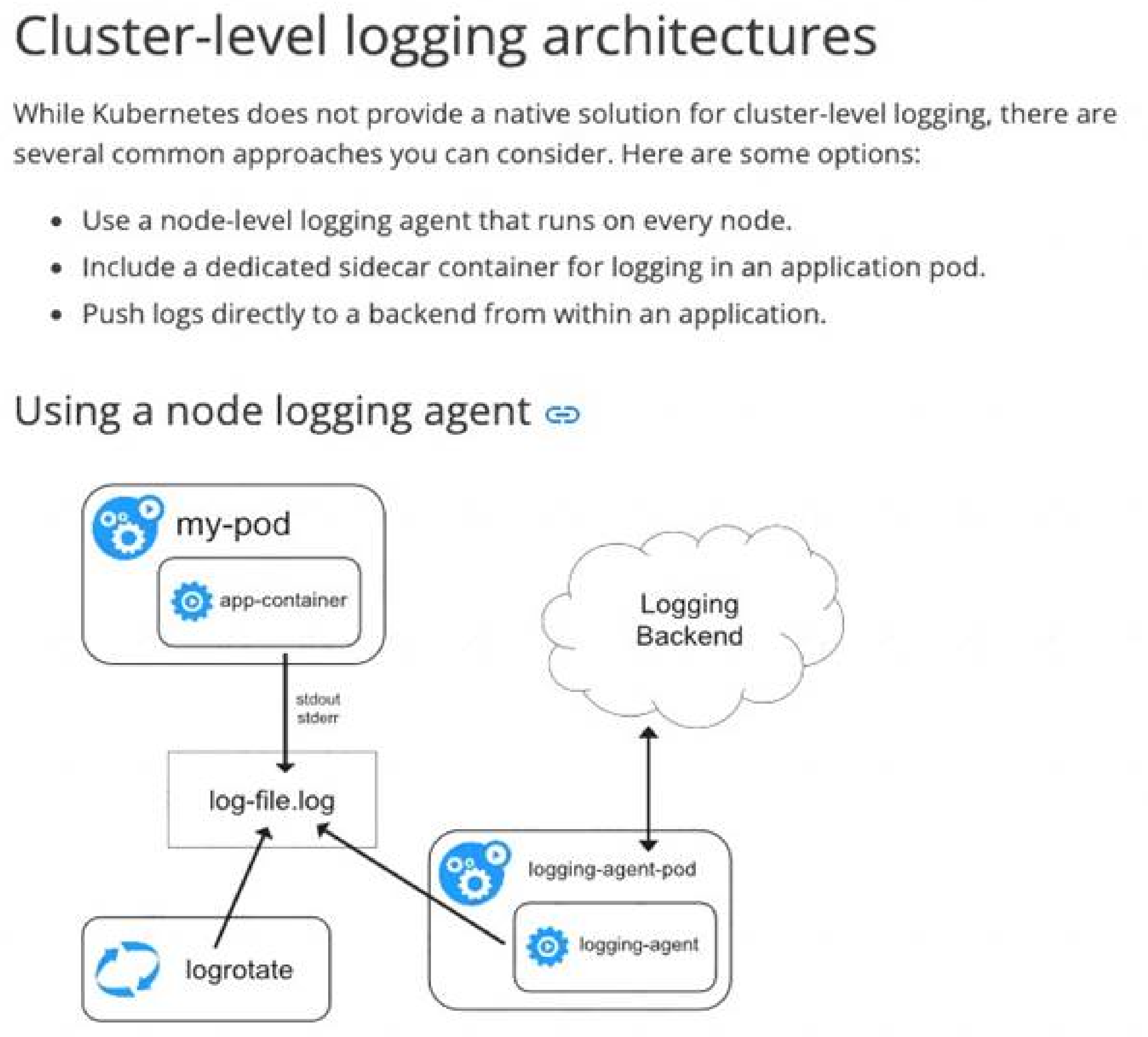
Container orchestration can be used in any environment where you use containers. It can help you to deploy the same application across different environments without needing to redesign it. And microservices in containers make it easier to orchestrate services, including storage, networking, and security.

NEW QUESTION 14
What do you call the pattern where you add a second container to the pod to collect logs infor-mation?

A. Sidecar container logging
B. Node level logging
C. Application level logging
D. Cluster level logging

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/cluster-administration/logging/ Diagram Description automatically generated



Cluster-level logging architectures

While Kubernetes does not provide a native solution for cluster-level logging, there are several common approaches you can consider. Here are some options:

- Use a node-level logging agent that runs on every node.
- Include a dedicated sidecar container for logging in an application pod.
- Push logs directly to a backend from within an application.

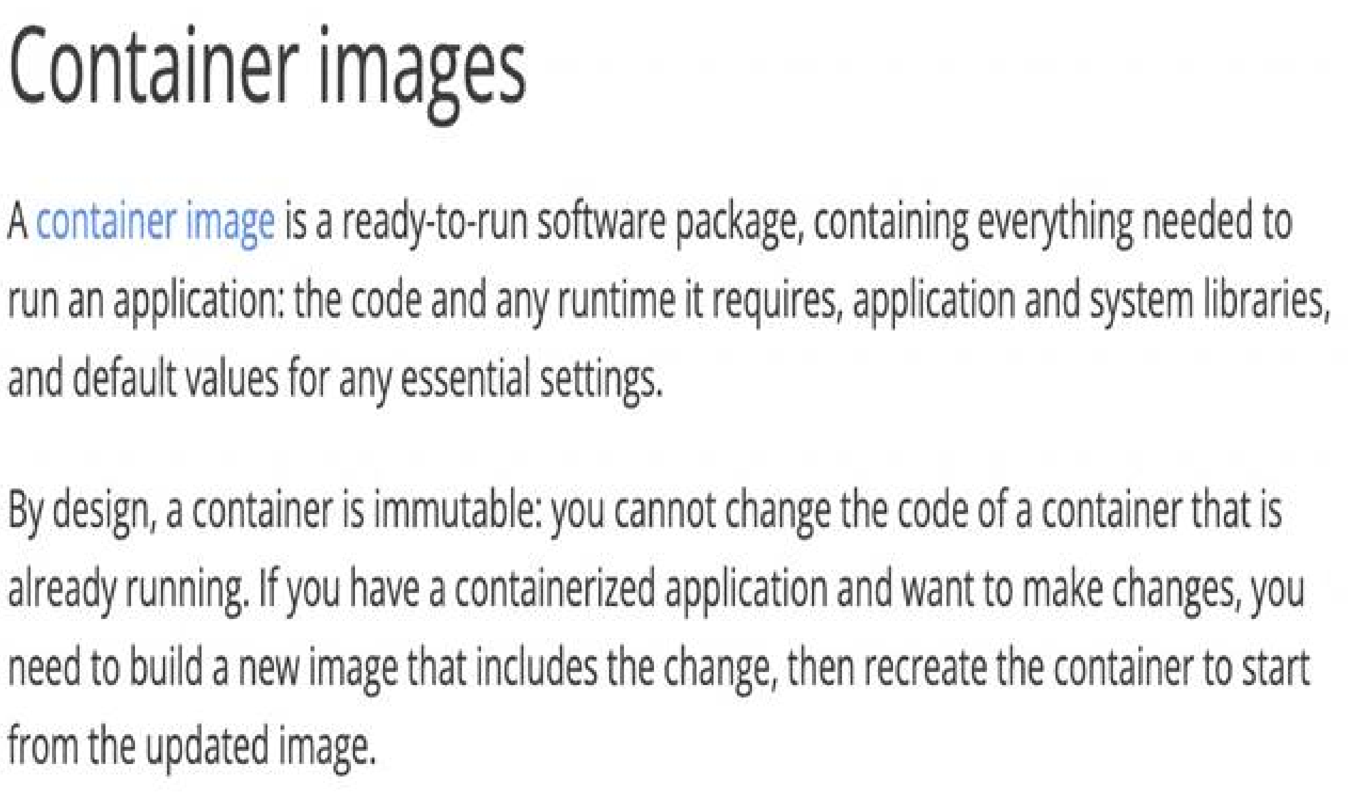
Using a node logging agent

NEW QUESTION 16
A is a ready-to-run software package, containing everything needed to run an application.

A. Container Repository
B. Container Runtime
C. Docker
D. Container Image

Answer: D

Explanation:
https://kubernetes.io/docs/concepts/containers/#container-images Text, letter Description automatically generated



Container images

A container image is a ready-to-run software package, containing everything needed to run an application: the code and any runtime it requires, application and system libraries, and default values for any essential settings.

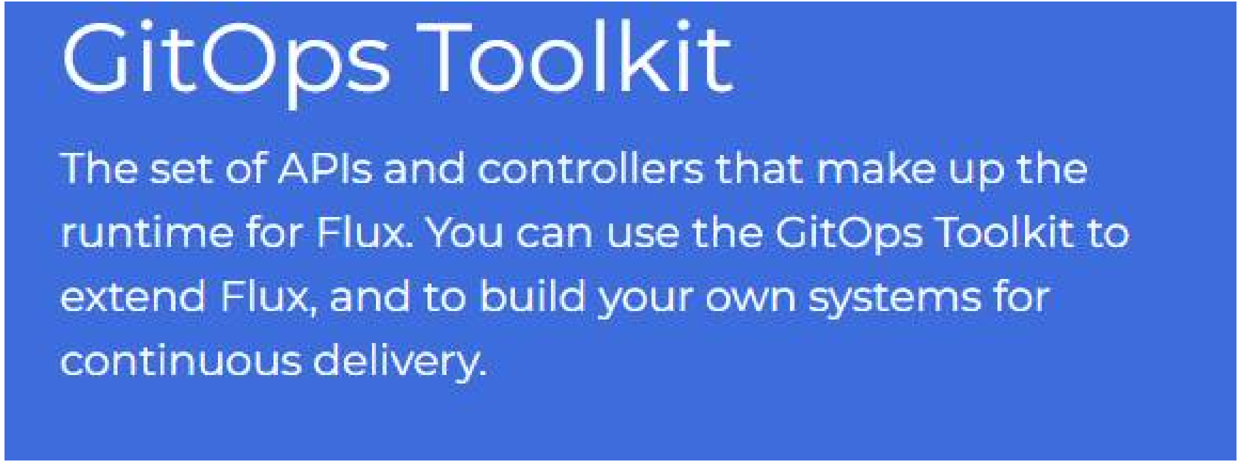
By design, a container is immutable: you cannot change the code of a container that is already running. If you have a containerized application and want to make changes, you need to build a new image that includes the change, then recreate the container to start from the updated image.

NEW QUESTION 21
Which tool is built on the GitOps toolkit?

A. Jenkins-X
B. GitHub Workflow & Actions
C. Flux
D. Jenkins
E. ArgoCD
F. Travis CI

Answer: C

Explanation:
https://fluxcd.io/#gitops-toolkit
Graphical user interface, text, application Description automatically generated



GitOps Toolkit

The set of APIs and controllers that make up the runtime for Flux. You can use the GitOps Toolkit to extend Flux, and to build your own systems for continuous delivery.

Note: Argo CD is a GitOps tool and not using GitOps toolkit

NEW QUESTION 22
Which style of operations are preferred for kubernetes and cloud-native applications?

A. Imperative
B. None of the above
C. Declarative

Answer: C

Explanation:
https://kubernetes.io/docs/tasks/manage-kubernetes-objects/declarative-config/#trade-offs

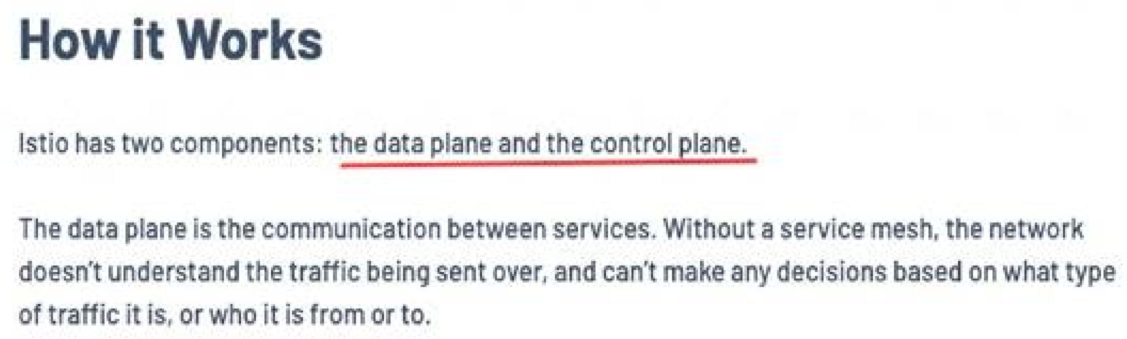
NEW QUESTION 25
What are the two major components of service mesh?

A. Control plane and Data plane
B. Master plane and Data plane
C. None of the options

D. Controller plane and User plane
E. Master plane and User plane

Answer: A

Explanation:
https://istio.io/latest/about/service-mesh/
Graphical user interface, text, application, email Description automatically generated



How it Works

Istio has two components: the data plane and the control plane.

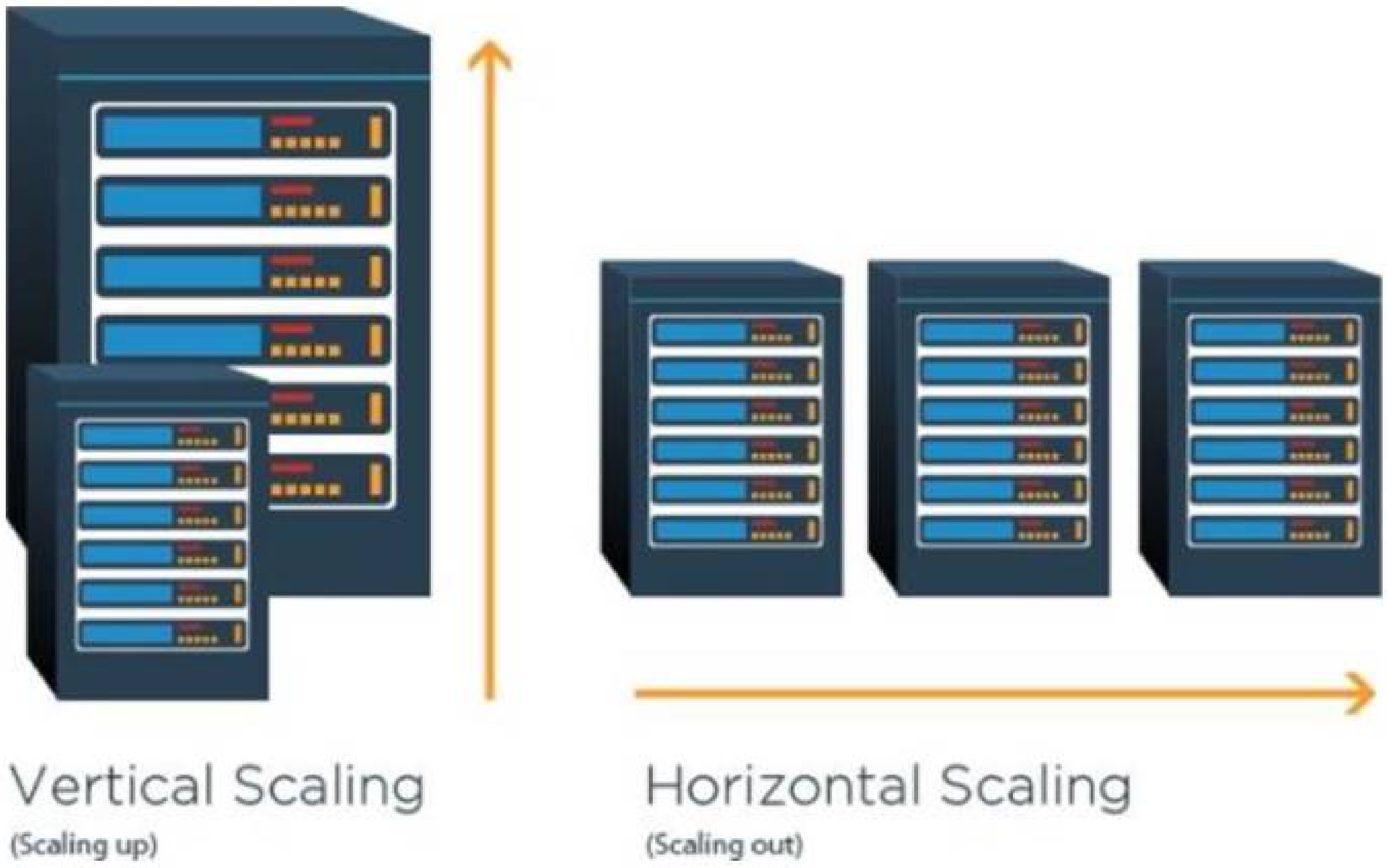
The data plane is the communication between services. Without a service mesh, the network doesn't understand the traffic being sent over, and can't make any decisions based on what type of traffic it is, or who it is from or to.

NEW QUESTION 26
An application that is nearing its usage limit. To increase the amount of users it can handle, you allo-cate additional memory resources to each instance of the application. What type of scaling is this?

A. Horizontal Scaling
B. Cluster Autoscaling
C. Recursive Scaling
D. Vertical Scaling

Answer: D

Explanation:
Graphical user interface, diagram Description automatically generated

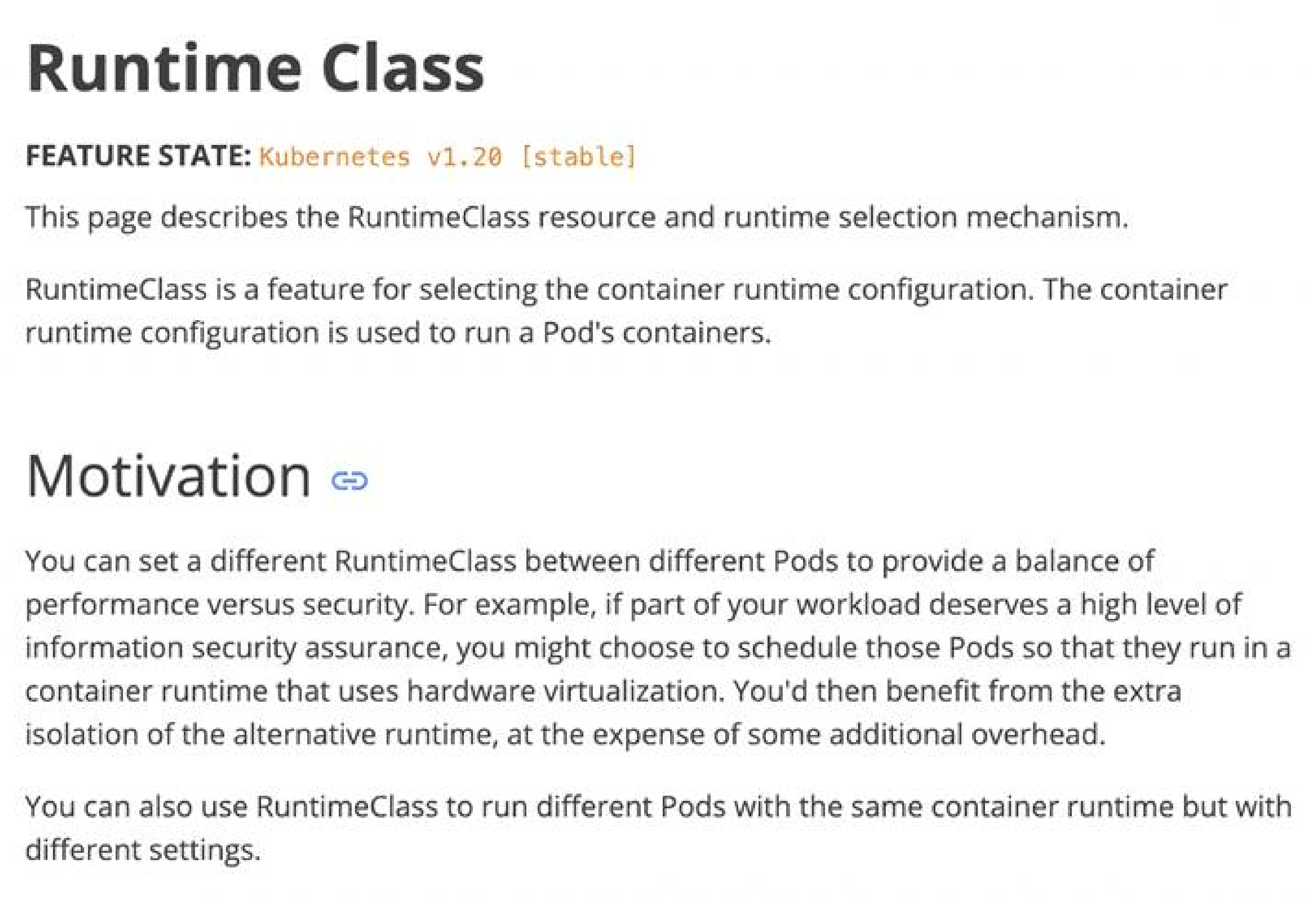


NEW QUESTION 31
What feature is used for selecting the container runtime configuration?

A. RuntimeClass
B. RuntimeContainer
C. Runtime
D. RuntimeConfig

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/containers/runtime-class/
Graphical user interface, text, application, email Description automatically generated



Runtime Class

FEATURE STATE: `Kubernetes v1.20 [stable]`

This page describes the RuntimeClass resource and runtime selection mechanism.

RuntimeClass is a feature for selecting the container runtime configuration. The container runtime configuration is used to run a Pod's containers.

Motivation

You can set a different RuntimeClass between different Pods to provide a balance of performance versus security. For example, if part of your workload deserves a high level of information security assurance, you might choose to schedule those Pods so that they run in a container runtime that uses hardware virtualization. You'd then benefit from the extra isolation of the alternative runtime, at the expense of some additional overhead.

You can also use RuntimeClass to run different Pods with the same container runtime but with different settings.

NEW QUESTION 34
Which part of a Kubernetes cluster is responsible for running container workloads?

A. Worker Node
B. kube-proxy
C. Control plane
D. etcd

Answer: A

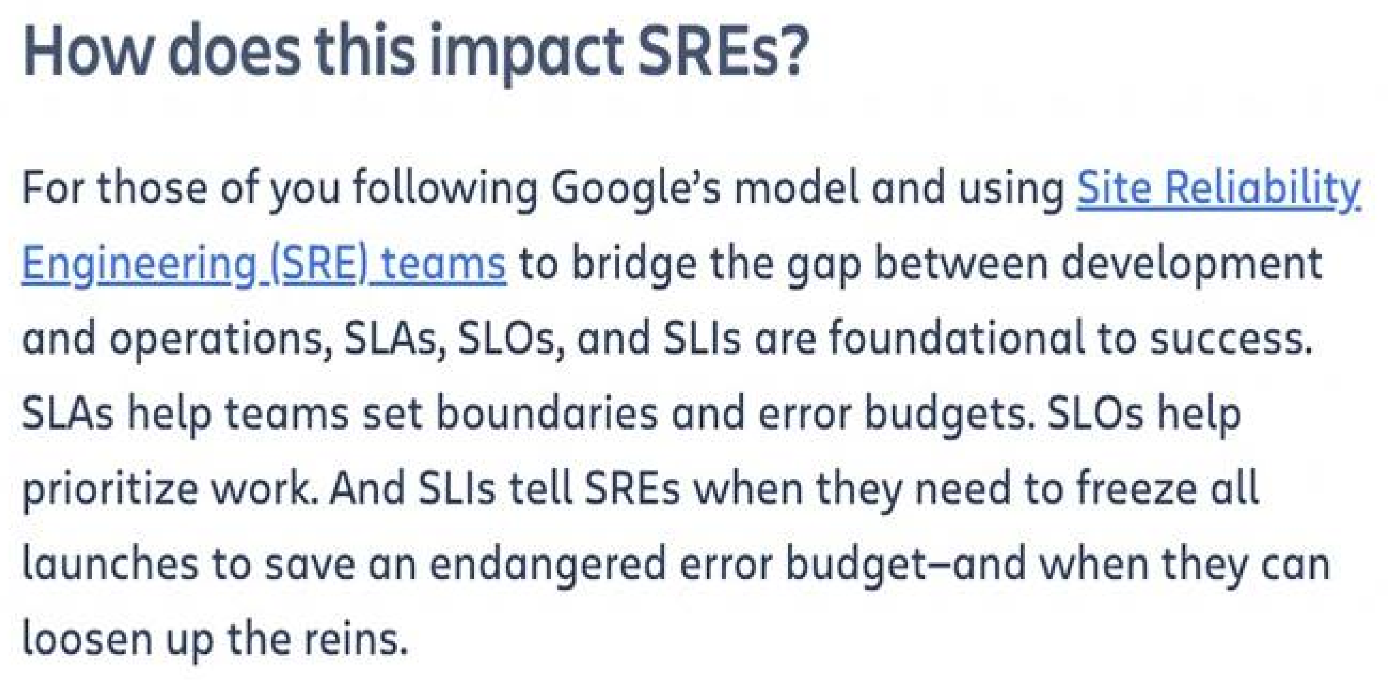
Explanation:
Worker Nodes are responsible for executing containerized workloads.

NEW QUESTION 35
Which role is responsible of creating service level indicator 'SLI', service level objective 'SLO', & Service Level Agreements 'SLA'

A. Site reliability engineer 'SRE'
B. DevOps
C. GitOps
D. Security and compliance engineer
E. Developer

Answer: A

Explanation:
https://www.atlassian.com/incident-management/kpis/sla-vs-slo-vs-sli Text Description automatically generated



How does this impact SREs?

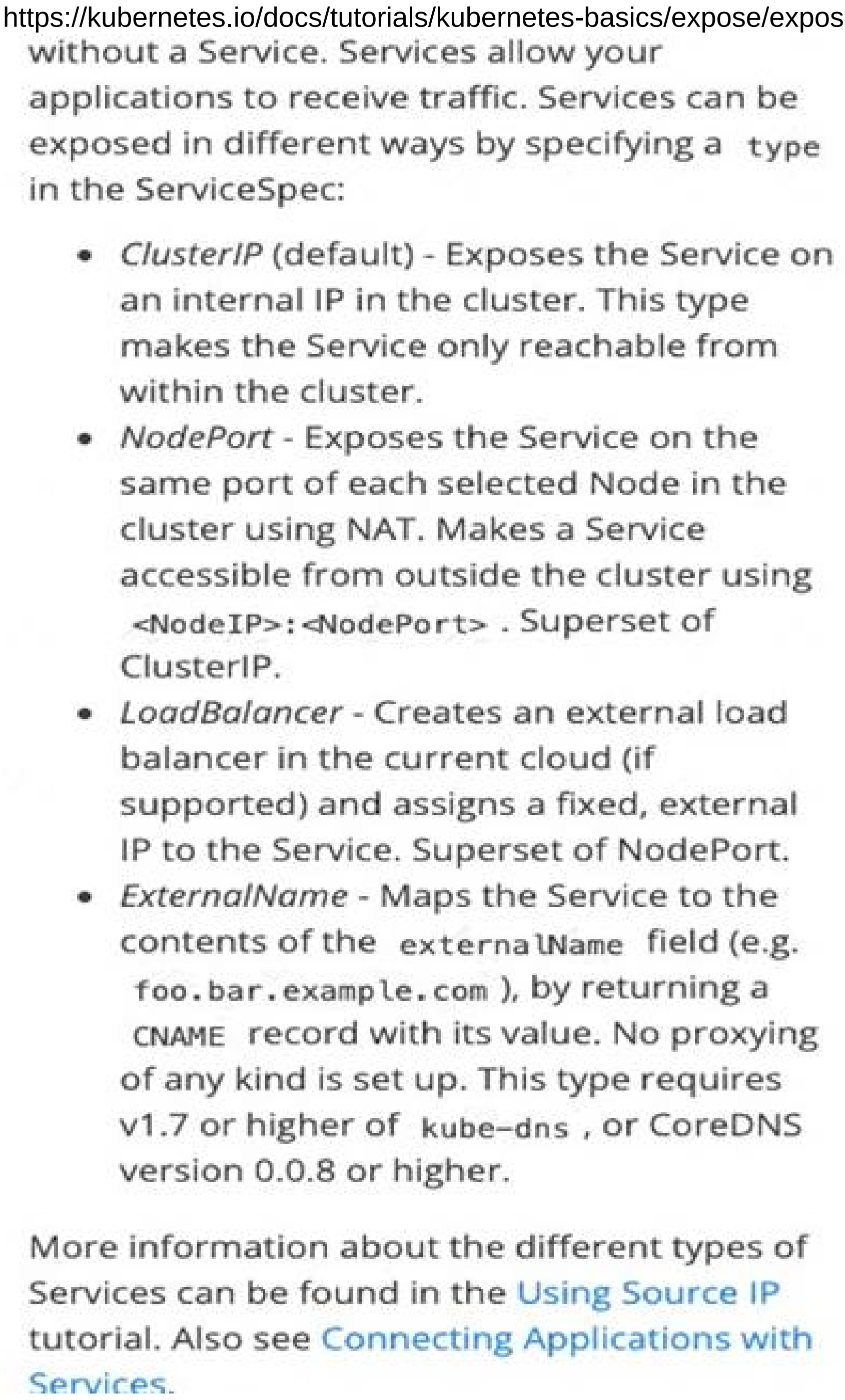
For those of you following Google's model and using Site Reliability Engineering (SRE) teams to bridge the gap between development and operations, SLAs, SLOs, and SLIs are foundational to success. SLAs help teams set boundaries and error budgets. SLOs help prioritize work. And SLIs tell SREs when they need to freeze all launches to save an endangered error budget–and when they can loosen up the reins.

NEW QUESTION 39
Which is not a service type in Kubernetes?

A. ClusterIP
B. NodePort
C. Ingress
D. LoadBalancer
E. ExternalName

Answer: C

Explanation:
https://kubernetes.io/docs/tutorials/kubernetes-basics/expose/expose-intro/



without a Service. Services allow your applications to receive traffic. Services can be exposed in different ways by specifying a `type` in the ServiceSpec:

- *ClusterIP* (default) - Exposes the Service on an internal IP in the cluster. This type makes the Service only reachable from within the cluster.
- *NodePort* - Exposes the Service on the same port of each selected Node in the cluster using NAT. Makes a Service accessible from outside the cluster using `<NodeIP>:<NodePort>` . Superset of ClusterIP.
- *LoadBalancer* - Creates an external load balancer in the current cloud (if supported) and assigns a fixed, external IP to the Service. Superset of NodePort.
- *ExternalName* - Maps the Service to the contents of the `externalName` field (e.g. `foo.bar.example.com`), by returning a `CNAME` record with its value. No proxying of any kind is set up. This type requires v1.7 or higher of `kube-dns` , or CoreDNS version 0.0.8 or higher.

More information about the different types of Services can be found in the Using Source IP tutorial. Also see Connecting Applications with Services.

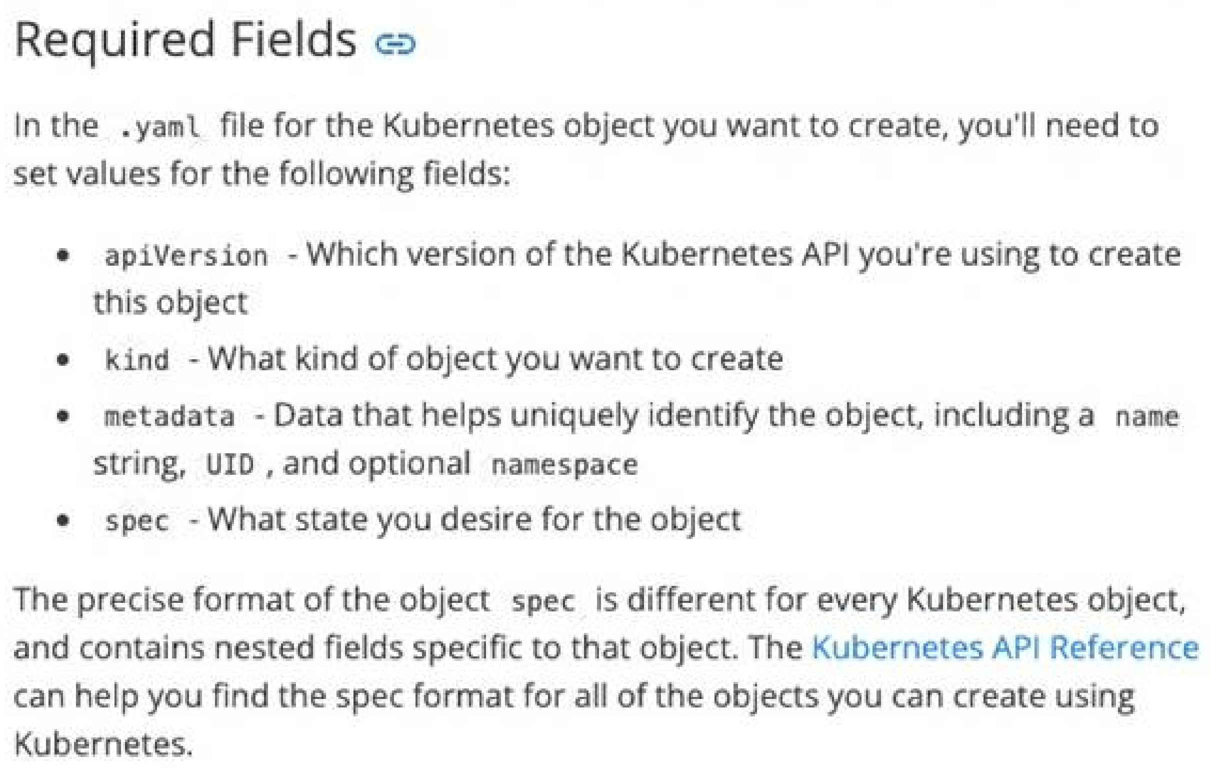
Text Description automatically generated

NEW QUESTION 41
Which of the following is not the required field to describe Kubernetes objects?

A. metadata
B. apiVersion
C. Kind
D. Container
E. spec

Answer: D

Explanation:
https://kubernetes.io/docs/concepts/overview/working-with-objects/kubernetes-objects/ Graphical user interface, text, application Description automatically generated



Required Fields

In the `.yaml` file for the Kubernetes object you want to create, you'll need to set values for the following fields:

- `apiVersion` - Which version of the Kubernetes API you're using to create this object
- `kind` - What kind of object you want to create
- `metadata` - Data that helps uniquely identify the object, including a `name` string, `UID`, and optional `namespace`
- `spec` - What state you desire for the object

The precise format of the object `spec` is different for every Kubernetes object, and contains nested fields specific to that object. The Kubernetes API Reference can help you find the spec format for all of the objects you can create using Kubernetes.

NEW QUESTION 44
Open Container Initiative set container standards for

A. Code, Build, Distribute, Deploy containers
B. Run, build, and image
C. Code, Build, Distribute containers
D. Run, Build, Distribute containers

Answer: D

NEW QUESTION 45
What is not semantic versioning?

A. 1.0.0
B. 2022-05-04
C. 1.0.0-alpha
D. 1.0.0-beta.2

Answer: B

Explanation:
https://semver.org/
RegEx SemVer at https://regex101.com/r/vkijKf/1/

NEW QUESTION 48
Observability and monitoring are not the same?

A. True
B. False

Answer: A

NEW QUESTION 49
What is autoscaling?

A. Automatically measuring resource usage
B. Automatically assigning workloads to nodes in a cluster
C. Automatically repairing broken application instances
D. Automatically adding or removing compute resources as needed

Answer: D

Explanation:
https://kubernetes.io/blog/2016/07/autoscaling-in-kubernetes/
Autoscaling means automatically scaling up or down in response to real-time usage data.

NEW QUESTION 52
Continuous delivery is .

A. Manually deploying the code
B. Coding, Building and Testing the code
C. Automatically deploying code to [container or server] environment

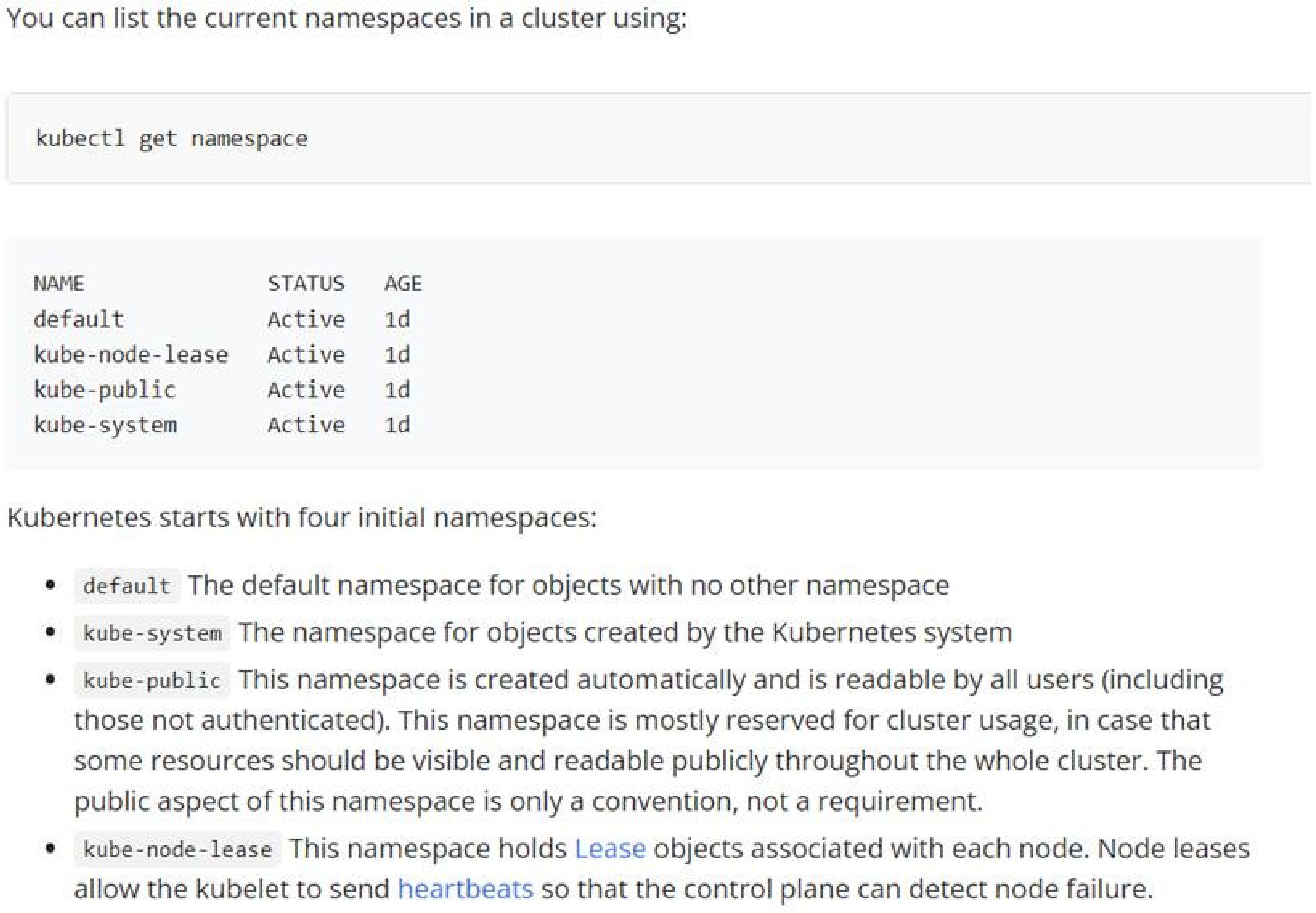
Answer: C

NEW QUESTION 55
What are default kubernetes namespaces?

A. default, kube-public, kube-system, kube-node-lease
B. kube-default, kube-public, kube-system, kube-node-lease
C. default, kube-public, kube-systems, kube-node-lease
D. default, kube-public, kube-system, kube-node-leases

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/
Graphical user interface, text Description automatically generated with medium confidence



You can list the current namespaces in a cluster using:

```
kubectl get namespace
```

```
NAME              STATUS   AGE
default           Active   1d
kube-node-lease   Active   1d
kube-public       Active   1d
kube-system       Active   1d
```

Kubernetes starts with four initial namespaces:

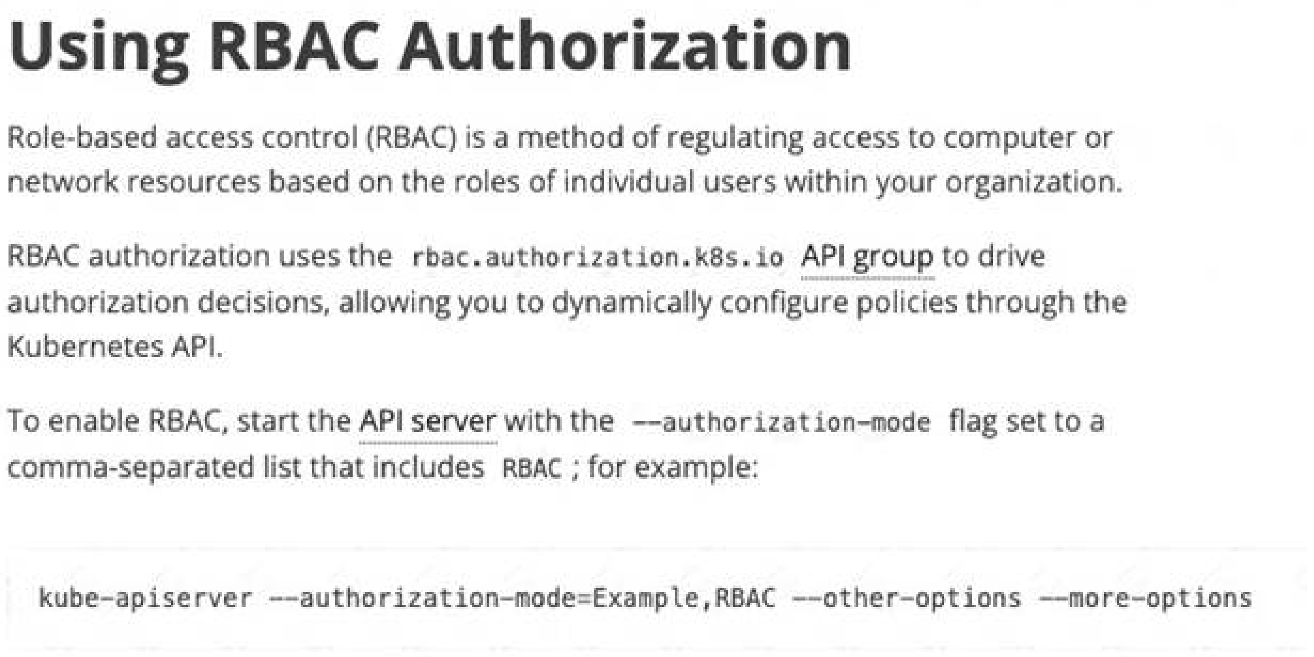
- `default` The default namespace for objects with no other namespace
- `kube-system` The namespace for objects created by the Kubernetes system
- `kube-public` This namespace is created automatically and is readable by all users (including those not authenticated). This namespace is mostly reserved for cluster usage, in case that some resources should be visible and readable publicly throughout the whole cluster. The public aspect of this namespace is only a convention, not a requirement.
- `kube-node-lease` This namespace holds Lease objects associated with each node. Node leases allow the kubelet to send heartbeats so that the control plane can detect node failure.

NEW QUESTION 60
Which of the following best describes the way kubernetes Role-based access control (RBAC) works?

A. Kubernetes does not do RBAC
B. Kubernetes RBAC states which users can perform which actions against which re-source
C. Kubernetes RBAC lists which operations on which resources are denied to users
D. Kubernetes RBAC is responsible for authenticating subjects such as users and groups

Answer: B

Explanation:
https://kubernetes.io/docs/reference/access-authn-authz/rbac/
Graphical user interface, text, application, email Description automatically generated



Using RBAC Authorization

Role-based access control (RBAC) is a method of regulating access to computer or network resources based on the roles of individual users within your organization.

RBAC authorization uses the `rbac.authorization.k8s.io` API group to drive authorization decisions, allowing you to dynamically configure policies through the Kubernetes API.

To enable RBAC, start the API server with the `--authorization-mode` flag set to a comma-separated list that includes `RBAC`; for example:

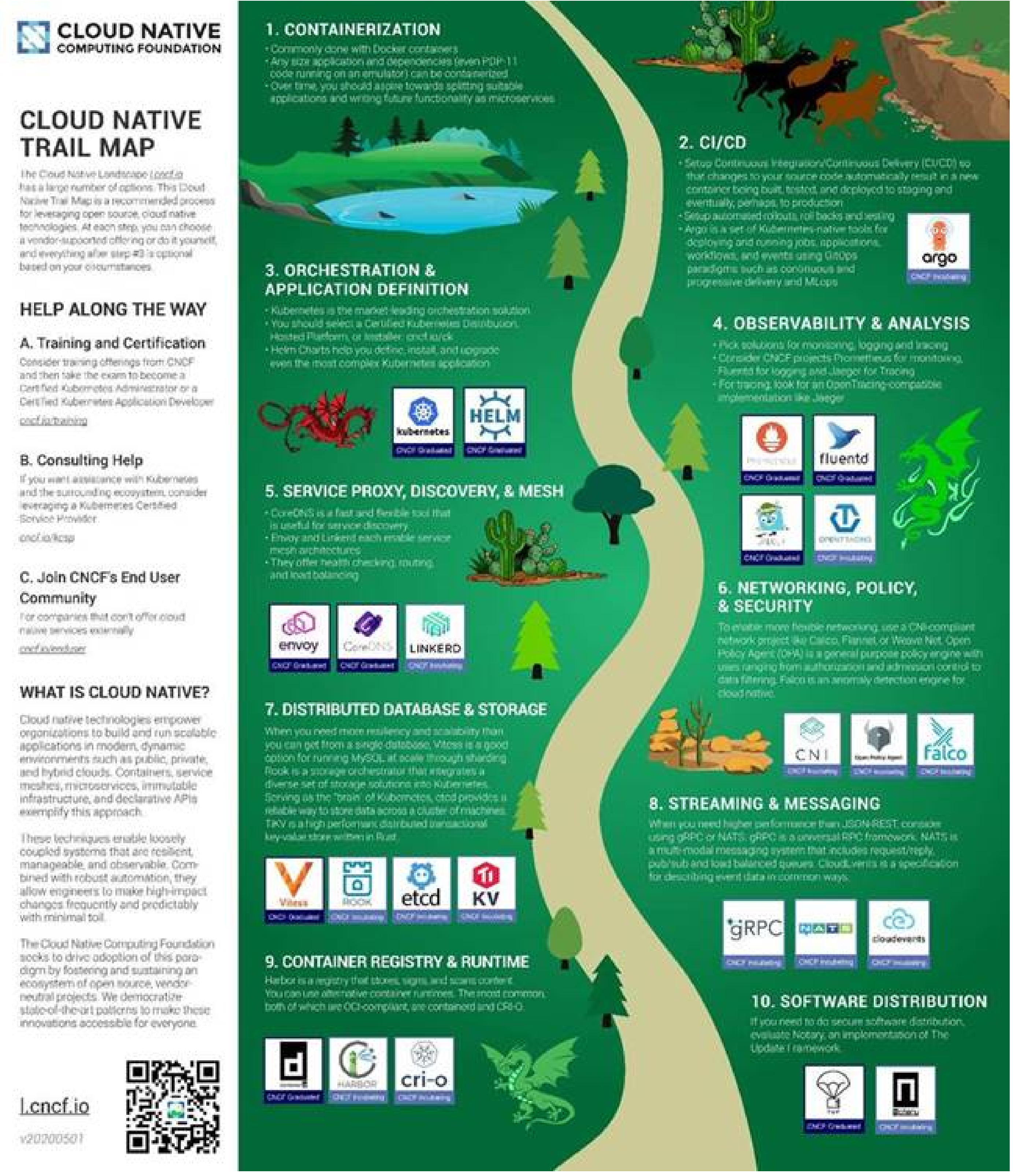
```
kube-apiserver --authorization-mode=Example,RBAC --other-options --more-options
```

NEW QUESTION 65
Fluentd is the leading project in the CNCF space for logging?

A. TRUE
B. FALSE

Answer: A

Explanation:
https://github.com/cncf/landscape#trail-map



CLOUD NATIVE COMPUTING FOUNDATION

CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape l.cncf.io has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification
Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer
cncf.io/training

B. Consulting Help
If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider
cncf.io/kcsp

C. Join CNCF's End User Community
For companies that don't offer cloud native services externally
cncf.io/enduser

WHAT IS CLOUD NATIVE?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

l.cncf.io

v20200501

1. CONTAINERIZATION
- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

2. CI/CD
- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing
- Argo is a set of Kubernetes-native tools for deploying and running jobs, applications, workflows, and events using GitOps paradigms such as continuous and progressive delivery and MLops

3. ORCHESTRATION & APPLICATION DEFINITION
- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/ck
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes application

4. OBSERVABILITY & ANALYSIS
- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger

5. SERVICE PROXY, DISCOVERY, & MESH
- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing

6. NETWORKING, POLICY, & SECURITY
To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net. Open Policy Agent (OPA) is a general purpose policy engine with uses ranging from authorization and admission control to data filtering. Falco is an anomaly detection engine for cloud native.

7. DISTRIBUTED DATABASE & STORAGE
When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes. Serving as the "brain" of Kubernetes, etcd provides a reliable way to store data across a cluster of machines. TiKV is a high performant distributed transactional key-value store written in Rust.

8. STREAMING & MESSAGING
When you need higher performance than JSON-REST, consider using gRPC or NATS. gRPC is a universal RPC framework. NATS is a multi-modal messaging system that includes request/reply, pub/sub and load balanced queues. CloudEvents is a specification for describing event data in common ways.

9. CONTAINER REGISTRY & RUNTIME
Harbor is a registry that stores, signs, and scans content. You can use alternative container runtimes. The most common, both of which are OCI-compliant, are containerd and CRI-O.

10. SOFTWARE DISTRIBUTION
If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.

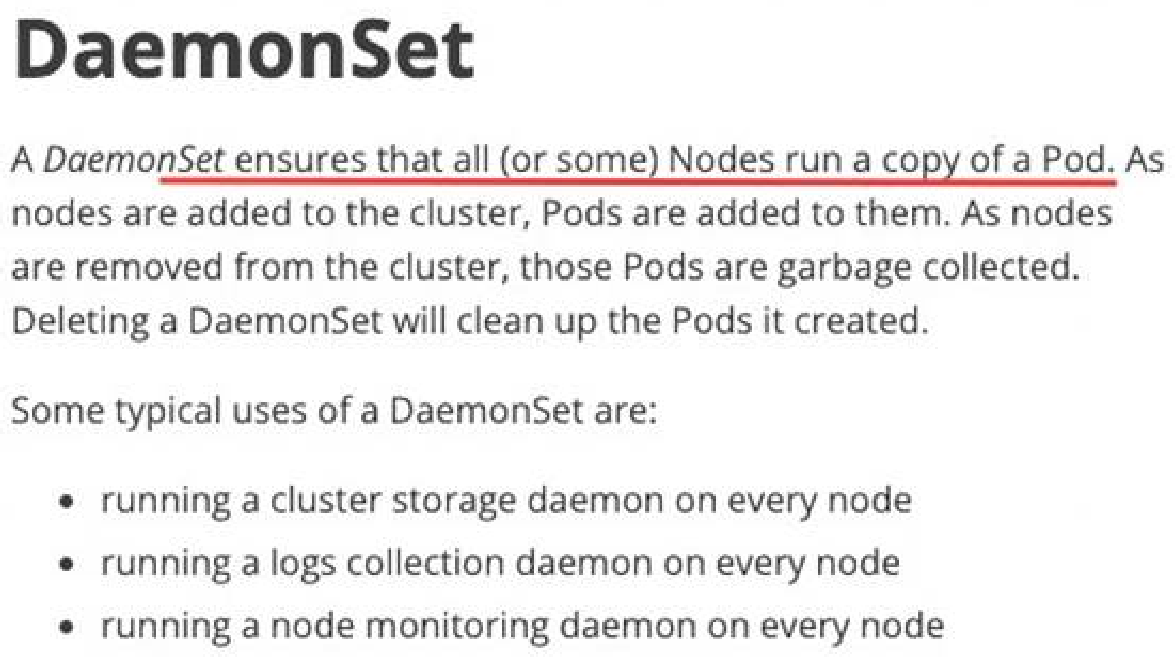
NEW QUESTION 69

What is the functionality of the daemon set?

A. To run a copy of the pod in all the nodes of the cluster
B. To initialize the pod before starting the main pod
C. To run a copy of the pod in a single node of the cluster

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/workloads/controllers/daemonset/
Graphical user interface, text, application Description automatically generated with medium confidence



DaemonSet

A *DaemonSet* ensures that all (or some) Nodes run a copy of a Pod. As nodes are added to the cluster, Pods are added to them. As nodes are removed from the cluster, those Pods are garbage collected. Deleting a DaemonSet will clean up the Pods it created.

Some typical uses of a DaemonSet are:

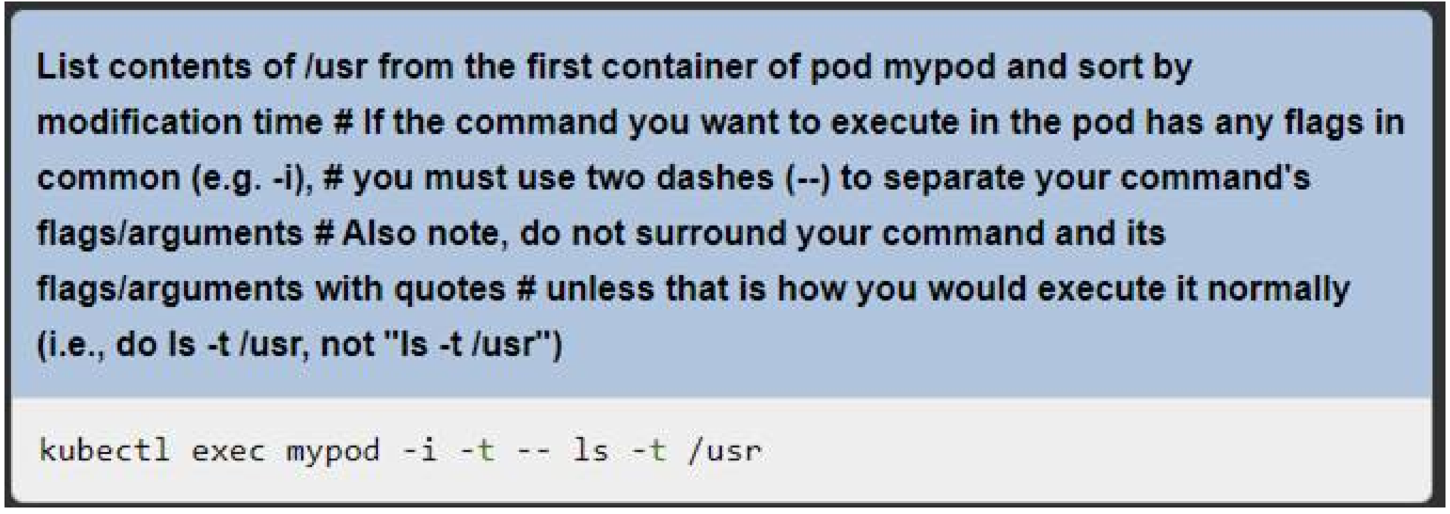
- running a cluster storage daemon on every node
- running a logs collection daemon on every node
- running a node monitoring daemon on every node

NEW QUESTION 73
What is the command used to login to the pod?

A. kubectl login
B. kubectl list
C. kubectl exec
D. kubectl get

Answer: C

Explanation:
https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#exec



List contents of /usr from the first container of pod mypod and sort by modification time # If the command you want to execute in the pod has any flags in common (e.g. -i), # you must use two dashes (--) to separate your command's flags/arguments # Also note, do not surround your command and its flags/arguments with quotes # unless that is how you would execute it normally (i.e., do ls -t /usr, not "ls -t /usr")

```
kubectl exec mypod -i -t -- ls -t /usr
```

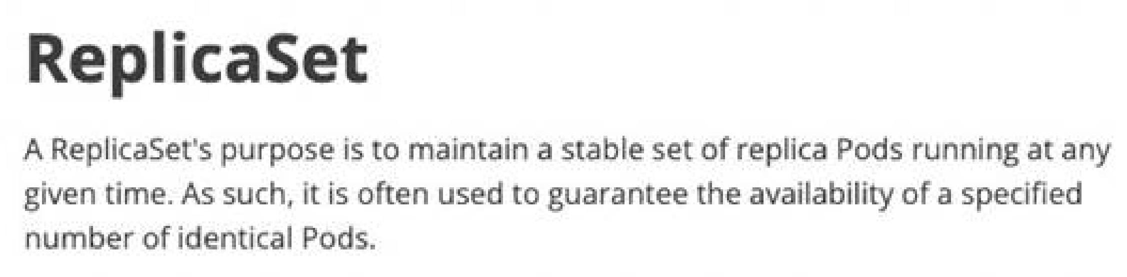
Text Description automatically generated

NEW QUESTION 76
Which kubernetes object do deployments use behind the scenes when they need to scale pods?

A. POD
B. Deployment
C. Horizontal pod autoscaler
D. Api Scheduler
E. Replicasets

Answer: E

Explanation:
https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/
Graphical user interface, text, application, email Description automatically generated



ReplicaSet

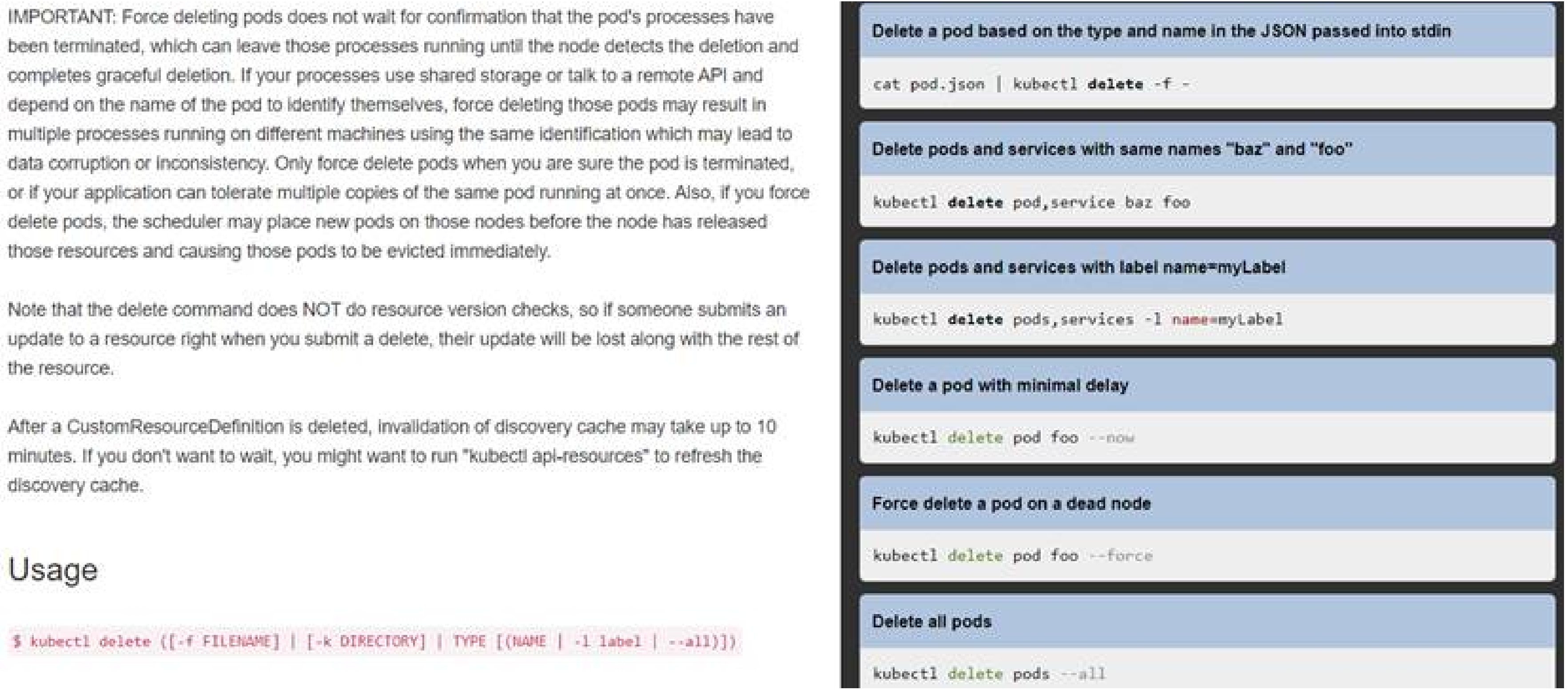
A ReplicaSet's purpose is to maintain a stable set of replica Pods running at any given time. As such, it is often used to guarantee the availability of a specified number of identical Pods.

NEW QUESTION 78
'kubectl delete -n my-ns po,svc --all' will delete pods and services including uninitialized ones in the namespace 'my-ns'

A. FALSE
B. TRUE

Answer: B

Explanation:
https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#delete Graphical user interface, text, application, email Description automatically generated



IMPORTANT: Force deleting pods does not wait for confirmation that the pod's processes have been terminated, which can leave those processes running until the node detects the deletion and completes graceful deletion. If your processes use shared storage or talk to a remote API and depend on the name of the pod to identify themselves, force deleting those pods may result in multiple processes running on different machines using the same identification which may lead to data corruption or inconsistency. Only force delete pods when you are sure the pod is terminated, or if your application can tolerate multiple copies of the same pod running at once. Also, if you force delete pods, the scheduler may place new pods on those nodes before the node has released those resources and causing those pods to be evicted immediately.

Note that the delete command does NOT do resource version checks, so if someone submits an update to a resource right when you submit a delete, their update will be lost along with the rest of the resource.

After a CustomResourceDefinition is deleted, invalidation of discovery cache may take up to 10 minutes. If you don't want to wait, you might want to run "kubectl api-resources" to refresh the discovery cache.

Usage

```
$ kubectl delete ([-f FILENAME] | [-k DIRECTORY] | TYPE [(NAME | -l label | --all)])
```

Delete a pod based on the type and name in the JSON passed into stdin

```
cat pod.json | kubectl delete -f -
```

Delete pods and services with same names "baz" and "foo"

```
kubectl delete pod,service baz foo
```

Delete pods and services with label name=myLabel

```
kubectl delete pods,services -l name=myLabel
```

Delete a pod with minimal delay

```
kubectl delete pod foo --now
```

Force delete a pod on a dead node

```
kubectl delete pod foo --force
```

Delete all pods

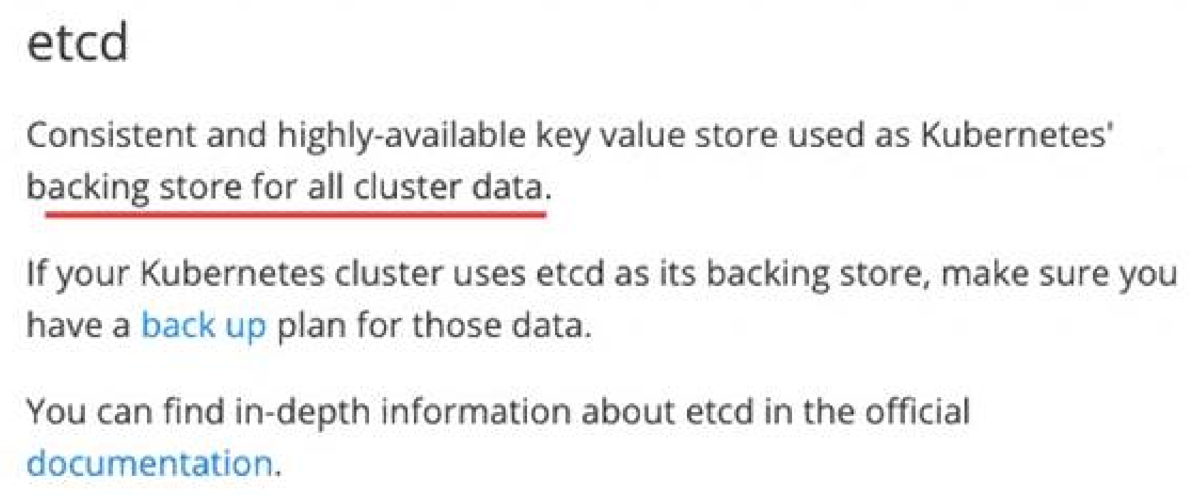
```
kubectl delete pods --all
```

NEW QUESTION 79
In Kubernetes, what is considered the primary cluster data source?

A. etcd (pronounce: esty-d)
B. api server
C. kubelet
D. scheduler

Answer: A

Explanation:
https://kubernetes.io/docs/concepts/overview/components/#etcd
Graphical user interface, text, application, email Description automatically generated



etcd

Consistent and highly-available key value store used as Kubernetes' backing store for all cluster data.

If your Kubernetes cluster uses etcd as its backing store, make sure you have a back up plan for those data.

You can find in-depth information about etcd in the official documentation.

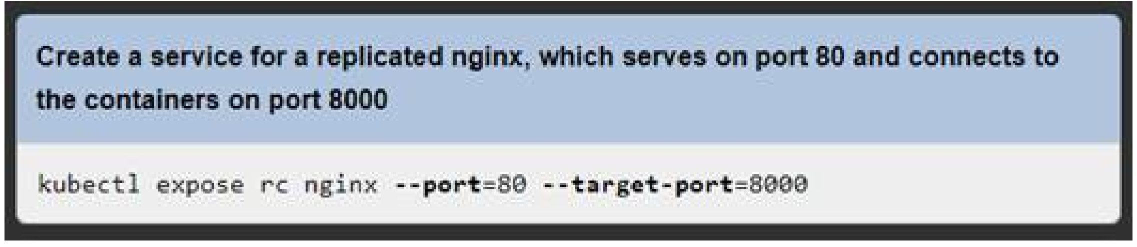
NEW QUESTION 82
Which command is used to expose Kubernetes service

A. kubectl expose
B. kubectl create

C. kubectl run

Answer: A

Explanation:
https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#expose Text Description automatically generated



Create a service for a replicated nginx, which serves on port 80 and connects to the containers on port 8000

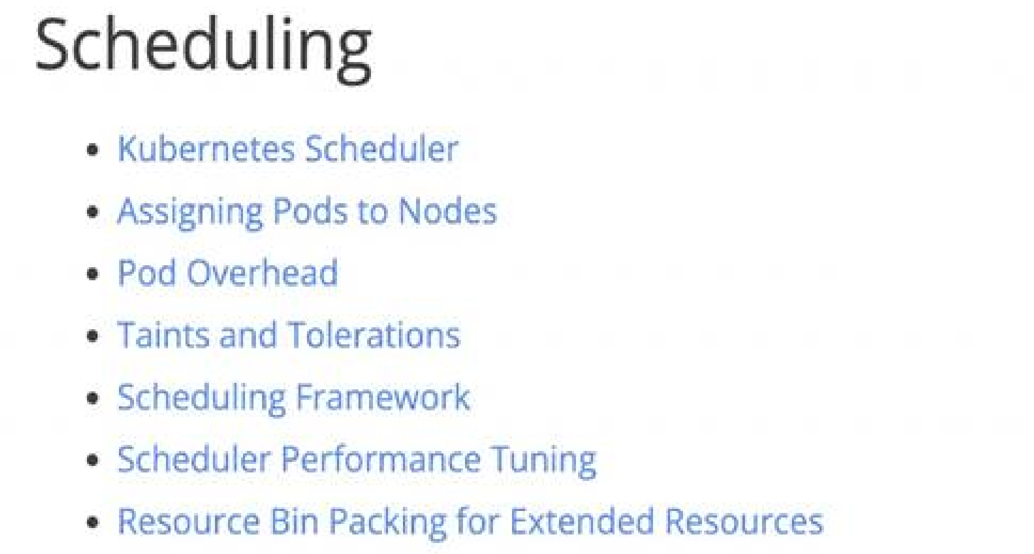
```
kubectl expose rc nginx --port=80 --target-port=8000
```

NEW QUESTION 84
What is scheduling in Kubernetes

A. Determining when to execute a cron-job
B. Assigning pods to nodes
C. Joining a new nodes to the clusters
D. Setting a time for automated tasks

Answer: B

Explanation:
https://kubernetes.io/docs/concepts/scheduling-eviction/
Graphical user interface, application Description automatically generated



Scheduling

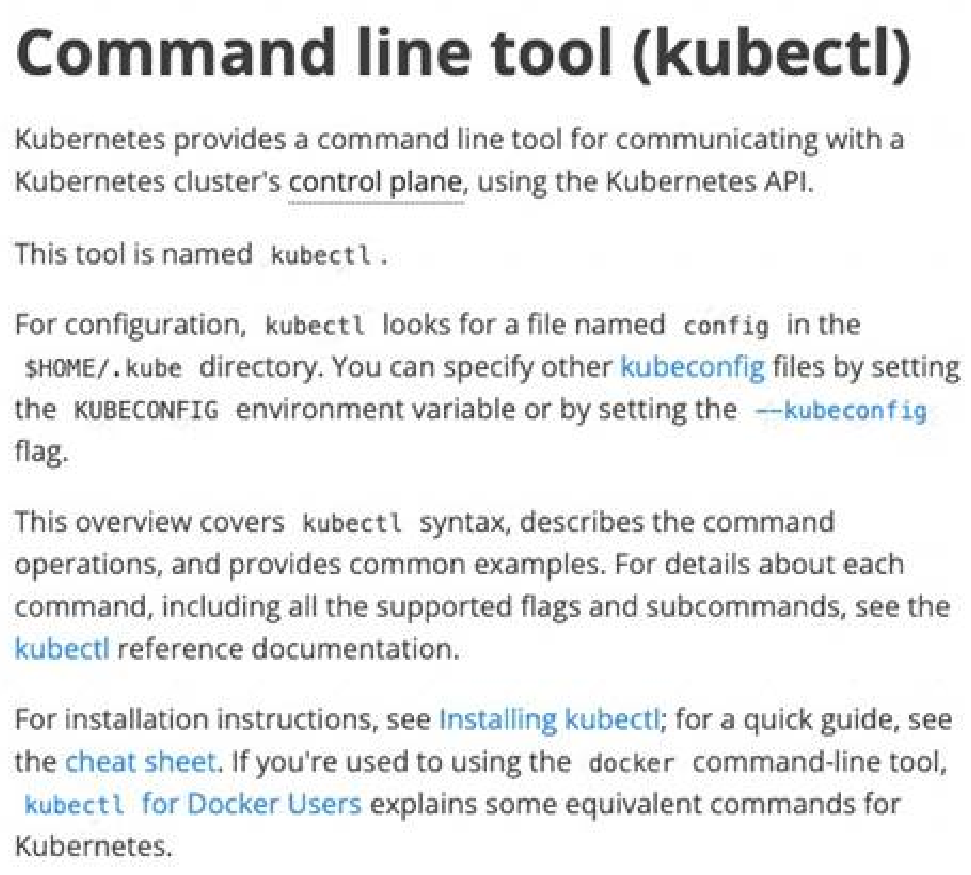
- Kubernetes Scheduler
- Assigning Pods to Nodes
- Pod Overhead
- Taints and Tolerations
- Scheduling Framework
- Scheduler Performance Tuning
- Resource Bin Packing for Extended Resources

NEW QUESTION 86
Which command-line tool is used to interact with the Kubernetes cluster?

A. kube-api
B. kubectl
C. kube-scheduler

Answer: B

Explanation:
https://kubernetes.io/docs/reference/kubectl/
Graphical user interface, text, application, email Description automatically generated



Command line tool (kubectl)

Kubernetes provides a command line tool for communicating with a Kubernetes cluster's control plane, using the Kubernetes API.

This tool is named `kubectl` .

For configuration, `kubectl` looks for a file named `config` in the `$HOME/.kube` directory. You can specify other kubeconfig files by setting the `KUBECONFIG` environment variable or by setting the `--kubeconfig` flag.

This overview covers `kubectl` syntax, describes the command operations, and provides common examples. For details about each command, including all the supported flags and subcommands, see the kubectl reference documentation.

For installation instructions, see Installing kubectl; for a quick guide, see the cheat sheet. If you're used to using the `docker` command-line tool, `kubectl` for Docker Users explains some equivalent commands for Kubernetes.

NEW QUESTION 91

......