Pass Guaranteed 2026 Linux Foundation High Pass-Rate Exam KCNA Reference

DOWNLOAD the newest ITexamReview KCNA PDF dumps from Cloud Storage for free: https://drive.google.com/open?id=1D8iEUX9nRCMeZwU65Q2vBeWJDcpViBpG

Just choose the right ITexamReview Linux Foundation KCNA exam questions format demo and download it quickly. Download the Linux Foundation KCNA exam questions demo now and check the top features of Linux Foundation KCNA Exam Questions. If you think the Linux Foundation KCNA exam dumps can work for you then take your buying decision. Best of luck in exams and career!!!

The Linux Foundation KCNA exam is designed to test the candidate's knowledge and understanding of Kubernetes and cloud-native technologies. It covers a range of topics such as deploying, managing, and scaling applications in Kubernetes, understanding Kubernetes architecture and components, and configuring Kubernetes networking and security. KCNA Exam also tests the candidate's knowledge of containerization technologies such as Docker and container orchestration tools like Helm.

>> Exam KCNA Reference <<

Download KCNA Demo, Reliable KCNA Exam Guide

Our Linux Foundation KCNA practice materials from our company are invulnerable. And we are consigned as the most responsible company in this area. So many competitors concede our superior position in the market. Besides, we offer some promotional benefits for you. The more times you choose our Linux Foundation KCNA Training Materials, the more benefits you can get, such as free demos of our KCNA exam dumps, three-version options, rights of updates and so on. So customer orientation is the beliefs we honor.

Linux Foundation KCNA Certification is a valuable credential for IT professionals who want to demonstrate their expertise in cloud-native technologies. With a wide range of study resources available and a vendor-neutral approach, the exam is an attractive option for professionals who work with different cloud platforms and want to showcase their skills in a way that is recognized across the industry.

Linux Foundation Kubernetes and Cloud Native Associate Sample Questions (Q133-Q138):

NEW QUESTION # 133
You have a Kubernetes cluster with a deployment named 'my-app' that has a replica count of 3. You need to temporarily scale it down to 1 replica. Which command would you use?

- A. kubectl patch deployments my-app -type=json-patch='{"spec":{"replicas":1}}'
- B. kubectl edit deployment my-app -replicas=1
- C. kubectl apply -f my-app-deployment.yaml --replicas=1
- D. kubectl scale deployments my-app -replicas=1
- E. kubectl set scale deployments my-app -replicas=1

Answer: D,E

Explanation:
Both 'kubectl scale deployments my-app -replicas=l' and 'kubectl set scale deployments my- app --replicas=1' are valid commands to scale down the deployment. Option 'C' also works but it is more verbose. Option 'B' would edit the deployment configuration, which is not the desired outcome in this case. Option 'E' would apply a new configuration, which is also not ideal.

NEW QUESTION # 134
What is a benefits of Kubernetes federation?

- A. Creates highly available clusters in different regions
- B. Low latency
- C. Avoids scalability limits on pods and nodes

Answer: A,B,C

NEW QUESTION # 135
What does "continuous" mean in the context of CI/CD?

- A. Frequent releases, manual processes, repeatable, fast processing
- B. Periodic releases, manual processes, repeatable, automated processing
- C. Periodic releases, automated processes, repeatable, automated processing
- D. Frequent releases, automated processes, repeatable, fast processing

Answer: D

Explanation:
The correct answer is C: in CI/CD, "continuous" implies frequent releases, automation, repeatability, and fast feedback/processing. The intent is to reduce batch size and latency between code change and validation/deployment. Instead of integrating or releasing in large, risky chunks, teams integrate changes continually and rely on automation to validate and deliver them safely.
"Continuous" does not mean "periodic" (which eliminates B and D). It also does not mean "manual processes" (which eliminates A and B). Automation is core: build, test, security checks, and deployment steps are consistently executed by pipeline systems, producing reliable outcomes and auditability.
In practice, CI means every merge triggers automated builds and tests so the main branch stays in a healthy state. CD means those validated artifacts are promoted through environments with minimal manual steps, often including progressive delivery controls (canary, blue/green), automated rollbacks on health signal failures, and policy checks. Kubernetes works well with CI/CD because it is declarative and supports rollout primitives: Deployments, readiness probes, and rollback revision history enable safer continuous delivery when paired with pipeline automation.
Repeatability is a major part of "continuous." The same pipeline should run the same way every time, producing consistent artifacts and deployments. This reduces "works on my machine" issues and shortens incident resolution because changes are traceable and reproducible. Fast processing and frequent releases also mean smaller diffs, easier debugging, and quicker customer value delivery.
So, the combination that accurately reflects "continuous" in CI/CD is frequent + automated + repeatable + fast, which is option C.

NEW QUESTION # 136
Which of these is a valid container restart policy?

- A. On login
- B. On start
- C. On failure
- D. On update

Answer: C

Explanation:
The correct answer is D: On failure. In Kubernetes, restart behavior is controlled by the Pod-level field spec.restartPolicy, with valid values Always, OnFailure, and Never. The option presented here ("On failure") maps to Kubernetes' OnFailure policy. This setting determines what the kubelet should do when containers exit:
Always: restart containers whenever they exit (typical for long-running services) OnFailure: restart containers only if they exit with a

non-zero status (common for batch workloads) Never: do not restart containers (fail and leave it terminated) So "On failure" is a valid restart policy concept and the only one in the list that matches Kubernetes semantics.
The other options are not Kubernetes restart policies. "On login," "On update," and "On start" are not recognized values and don't align with how Kubernetes models container lifecycle. Kubernetes is declarative and event-driven: it reacts to container exit codes and controller intent, not user "logins." Operationally, choosing the right restart policy is important. For example, Jobs typically use restartPolicy: OnFailure or Never because the goal is completion, not continuous uptime. Deployments usually imply "Always" because the workload should keep serving traffic, and a crashed container should be restarted. Also note that controllers interact with restarts: a Deployment may recreate Pods if they fail readiness, while a Job counts completions and failures based on Pod termination behavior.
Therefore, among the options, the only valid (Kubernetes-aligned) restart policy is D.

NEW QUESTION # 137
Which item is a Kubernetes node component?

- A. etcd
- B. kube-proxy
- C. kube-scheduler
- D. kubectl

Answer: B

Explanation:
A Kubernetes node component is a component that runs on worker nodes to support Pods and node-level networking/operations. Among the options, kube-proxy is a node component, so C is correct.
kube-proxy runs on each node and implements parts of the Kubernetes Service networking model. It watches the API server for Service and endpoint updates and then programs node networking rules (iptables/IPVS, or equivalent) so traffic sent to a Service IP/port is forwarded to one of the backend Pod endpoints. This is essential for stable virtual IPs and load distribution across Pods.
Why the other options are not node components:
kube-scheduler is a control plane component; it assigns Pods to nodes but does not run on every node as part of node functionality.
kubectl is a client CLI tool used by humans/automation; it is not a cluster component.
etcd is the control plane datastore; it stores cluster state and is not a per-node workload component.
Operationally, kube-proxy can be replaced by some modern CNI/eBPF dataplanes, but in classic Kubernetes architecture it remains the canonical node-level component for Service rule programming. Understanding which components are node vs control plane is key for troubleshooting: node issues involve kubelet/runtime/kube-proxy/CNI; control plane issues involve API server/scheduler/controller-manager/etcd.
So, the verified node component in this list is kube-proxy (C).

NEW QUESTION # 138
.....

www.vce4dumps.com □□□ and search for [KCNA] to obtain a free download □KCNA Valid Test Book

- KCNA Latest Test Preparation □ KCNA Latest Test Simulator □ KCNA Valid Test Book □ Download “ KCNA ” for free by simply entering ☀ www.pdfvce.com □☀□ website □Valid KCNA Mock Test
- KCNA PDF Questions □ KCNA New Dumps Sheet □ Exam KCNA Details □ Search for 【 KCNA 】 and easily obtain a free download on “ www.prepawayete.com ” □Vce KCNA Torrent
- www.stes.tyc.edu.tw, mpgimer.edu.in, www.stes.tyc.edu.tw, www.zazzle.com, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, lms.ait.edu.za, www.stes.tyc.edu.tw, lms.ait.edu.za, www.stes.tyc.edu.tw, Disposable vapes

P.S. Free 2026 Linux Foundation KCNA dumps are available on Google Drive shared by ITexamReview: https://drive.google.com/open?id=1D8iEUX9nRCMeZwU65Q2vBeWJDcpViBpG