DUMPSARENA

Oracle Cloud Infrastructure 2024 DevOps Professional

Oracle 1z0-1109-24

Version Demo

Total Demo Questions: 5

Total Premium Questions: 53

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, DevOps Culture and Practices	3
Topic 2, Continuous Integration	3
Topic 3, Continuous Delivery and Deployment	27
Topic 4, DevSecOps	7
Topic 5, Infrastructure as Code	8
Topic 6, Observability	4
Topic 7, Mix Questions	1
Total	53

QUESTION NO: 1

You have been asked to provision a new production environment on Oracle Cloud Infrastructure (OCI). After working with the solution architect, you decide that you are going to automate this process.

Which OCI service can help automate the provisioning of this new environment?

A. OCI Streaming Service

B. Oracle Functions

C. Oracle Container Engine for Kubernetes

D. OCI Resource Manager

ANSWER: D

Explanation:

OCI Resource Manager is the appropriate service because it provides OCI-managed infrastructure as code capabilities based on Terraform. A team can define the production environment—such as compartments, networking, compute instances, load balancers, identity policies, and other OCI resources—in Terraform configuration files, then use Resource Manager to create and manage those resources consistently. Resource Manager organizes Terraform configurations into stacks and executes them through jobs, including plan, apply, and destroy operations. This supports repeatable provisioning workflows and helps ensure that the production environment is deployed according to the architecture agreed with the solution architect. It also provides state management and job logs within OCI, reducing the operational burden of operating Terraform tooling separately. Configuration source can be stored in services such as OCI Code Repositories or external source-control repositories, supporting controlled and versioned infrastructure changes. For a production environment, this infrastructure-as-code approach enables reviewable, predictable deployments and makes it practical to recreate or update the environment over time. Oracle documents Resource Manager as an OCI service for automating deployment and operations of OCI resources using Terraform. See the OCI Resource Manager overview and the Terraform with Resource Manager documentation.

QUESTION NO: 2

As an engineer building and deploying applications using an OCI DevOps project, which two capabilities can help ensure the security and reliability of the code in the build and deployment pipelines? (Choose two.)

A. Using third-party tools like Ansible, Terraform, or OverOps to analyze code for security defects or bugs in code quality

B. Using Application Dependency Management (ADM) to identify security weaknesses in software applications by checking their dependencies

C. Using JIRA to track user stories and bug fixes in the development process

D. Using version control tools like Git or SVN to track and manage changes in the codebase

E. Using third-party tools like Sonatype, SonarQube, or OverOps to analyze code for security defects or bugs in code quality

ANSWER: B E

Explanation:

Using Application Dependency Management (ADM) to identify security weaknesses in software applications by checking their dependencies is correct because software commonly incorporates open-source and third-party libraries whose known vulnerabilities can affect the delivered application. OCI ADM analyzes application dependency inventories, identifies vulnerable components through vulnerability knowledge sources, and provides remediation-oriented information. Incorporating this type of dependency analysis into a pipeline helps teams detect risk before an artifact is promoted to later environments.

Using third-party tools like Sonatype, SonarQube, or OverOps to analyze code for security defects or bugs in code quality is also correct. OCI DevOps build pipelines support build stages that execute commands from a build specification, allowing teams to run external security, quality, and analysis tooling as part of automated continuous integration. These tools can

produce actionable findings on vulnerabilities, insecure coding patterns, defects, and maintainability concerns. Making such checks repeatable in the pipeline establishes consistent quality gates for each change and improves confidence in code that reaches deployment. OCI's DevOps guidance describes build pipelines as the mechanism for continuously building, testing, and delivering software, while ADM documentation details dependency vulnerability analysis. See Oracle Cloud Infrastructure DevOps Build Pipelines and Oracle Cloud Infrastructure Application Dependency Management Overview.

QUESTION NO: 3

Your team manages a production environment through an OCI Resource Manager stack. An administrator suspects that networking resources were modified manually outside Terraform and wants to assess the difference without changing any OCI resources.

Which two statements are true about using a Drift Detection job? (Choose two.)

A. It compares the stack state with the current state of the corresponding OCI resources.

B. It reports differences without modifying the OCI resources.

C. It automatically updates the Terraform configuration to match manually changed resources.

D. It automatically applies the changes required to restore the declared configuration.

E. It automatically imports manually created resources into the stack state.

ANSWER: A B

Explanation:

A Drift Detection job compares the resources represented in the stack state with the actual resources in OCI and reports detected differences. It is intended to identify configuration drift caused by changes made outside the Terraform workflow.

Drift detection is an assessment operation and does not create, modify, or delete resources. An Apply job is required to make Terraform-managed infrastructure changes. Resources created manually are not automatically imported into the Terraform state merely because drift detection is run.

QUESTION NO: 4

You are using the Oracle Cloud Infrastructure (OCI) DevOps service and you have successfully built and tested your software applications in your Build Pipeline. The resulting output needs to be stored in a container repository.

Which stage should you add next to your Build Pipeline?

A. Trigger deployment

B. Managed build

C. Deliver artifacts

D. Export packages

ANSWER: C

Explanation:

Deliver artifacts is the correct stage because it takes artifacts produced by an earlier managed build stage and publishes them to a configured artifact destination. In this scenario, the destination is an OCI Container Registry repository. A managed build can build and test a container image, but the image must then be delivered so that it is available outside the build environment for later consumption by deployment pipelines or other processes.

When configuring this stage, the pipeline identifies the build output artifact and defines its destination details, including the target container repository and image tag. OCI DevOps supports delivering build outputs to OCI services such as Container

Registry, Artifact Registry, and Object Storage. For containerized applications, delivering the image to Container Registry establishes the versioned image location that deployment workflows can reference. This separates artifact publication from deployment and provides a durable, centrally managed repository for the image produced by the pipeline.

Oracle documents the build-pipeline artifact-delivery capability and its supported destinations in the OCI DevOps Build Pipelines documentation. Container image repositories and image management are described in the OCI Container Registry overview.

QUESTION NO: 5

As a DevOps engineer working on managing clusters on the OCI platform for your organization, which statement is true about managing cluster add-ons in OCI OKE Cluster?

A. When creating a new cluster, essential cluster add-ons cannot be disabled.

B. When enabling a cluster add-on, you cannot configure the add-on by specifying one or more key/value pairs to pass as arguments to the cluster add-on.

C. When creating a new cluster, essential cluster add-ons are set to manually update.

D. When you disable a cluster add-on using the console, the add-on is completely removed from the cluster.

ANSWER: A

Explanation:

When creating a new cluster, essential cluster add-ons cannot be disabled. In Oracle Kubernetes Engine, cluster add-ons provide functionality that is installed and managed as part of the cluster environment. Oracle designates certain add-ons as essential because the cluster depends on their services for fundamental operation. Consequently, these add-ons are enabled as part of cluster creation and are not optional at that stage. This protection ensures that a newly provisioned OKE cluster has the core platform capabilities it requires before workloads are deployed, rather than allowing an administrator to create a cluster in an unsupported or nonfunctional baseline state.

For DevOps teams, this means add-on planning should distinguish required platform services from optional capabilities. Essential add-ons are part of the managed cluster foundation, while the add-on management workflow can be used to control eligible nonessential add-ons and their supported configuration settings. Oracle documents the add-on lifecycle, including enabling, disabling, configuring, and updating add-ons, in its OKE cluster add-ons documentation. See the Oracle Kubernetes Engine documentation for managing cluster add-ons and the Oracle Kubernetes Engine overview.