DUMPSBOSS.

Oracle Cloud Infrastructure 2024 Cloud Operations Professional

Oracle 1z0-1067-24

Version Demo

Total Demo Questions: 9

Total Premium Questions: 94

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, Identity and Access Management	14
Topic 2, Networking	25
Topic 3, Security	6
Topic 4, Governance	27
Topic 5, Observability	6
Topic 6, Business Continuity and Disaster Recovery	7
Topic 7, Mix Questions	9
Total	94

QUESTION NO: 1 - (SIMULATION)

Scenario: 1 (Create a reusable VCN Configuration with Terraform)

Scenario Description: (Hands-On Performance Exam Certification)

You'll launch and destroy a VCN and subnet by creating Terraform automation scripts and issuing commands in Code Editor. Next, you'll download those Terraform scripts and create a stack by uploading them into Oracle Cloud Infrastructure Resource Manager.

You'll then use that service to launch and destroy the same VCN and subnet.

In this scenario, you will:

a. Create a Terraform folder and file in Code Editor.

b. Create and destroy a VCN using Terraform.

c. Create and destroy a VCN using Resource Manager.

ANSWER: See the explanation for the answer

Explanation:

Answer: Create a reusable Terraform configuration containing `vcn.tf`, `variables.tf`, and `terraform.tfvars`; use Terraform CLI to initialize, plan, apply, and destroy it; then upload the same three files as a Resource Manager stack and run a Plan job, an Apply job based on that plan, and finally a Destroy job.

Create a folder named `terraform_vcn` in Code Editor and add the following files.

vcn.tf

```
terraform {
  required_providers {
    oci = {
      source  = "oracle/oci"
      version = ">= 4.67.3"
    }
  }

  required_version = ">= 1.0.0"
}

resource "oci_core_vcn" "example_vcn" {
  compartment_id = var.compartment_id
  display_name   = "VCN-01"
  cidr_blocks    = ["10.0.0.0/16"]
}

resource "oci_core_subnet" "example_subnet" {
  compartment_id = var.compartment_id
  display_name   = "SNT-01"
  vcn_id         = oci_core_vcn.example_vcn.id
  cidr_block     = "10.0.0.0/24"
}
```

variables.tf

```
variable "compartment_id" {
  type = string
}
```

terraform.tfvars — replace the value with the OCID of the target compartment:

```
compartment_id = "ocid1.compartment.oc1..<your_compartment_ocid>"
```

From the Code Editor terminal, change to the configuration directory and run:

```
cd ~/terraform_vcn
terraform init
terraform plan
terraform apply
```

At the confirmation prompt, enter `yes`. Verify that VCN `VCN-01` and subnet `SNT-01` were created. To remove the CLI-created resources, run:

```
terraform destroy
```

Enter `yes` when prompted.

Resource Manager portion:

Download only `vcn.tf`, `variables.tf`, and `terraform.tfvars` to a local `terraform_vcn` folder. Do not upload Terraform working artifacts such as `.terraform` or `.terraform.lock.hcl`.
In **Developer Services > Resource Manager > Stacks**, select **Create stack**.
Select **My configuration**, upload the `terraform_vcn` folder, leave **Use custom Terraform providers** unselected, provide a stack name/description, and select the correct compartment.
On the Variables page, confirm that `compartment_id` is populated from `terraform.tfvars`. Keep **Run apply** deselected and create the stack.
Select **Plan**, name the job `RM-Plan-01`, and submit it. Wait until the job succeeds.
Return to the stack and select **Apply**. Name it `RM-Apply-01`. For **Apply job plan resolution**, select the completed `RM-Plan-01` job rather than **Automatically approve**. Submit the Apply job and wait for success.
Verify that `VCN-01` and `SNT-01` exist in **Networking > Virtual Cloud Networks**.
Return to the stack, select **Destroy**, and confirm the destroy action. Wait for the destroy job to complete successfully, then verify the VCN is terminated.
Use **More actions > Delete stack** and confirm deletion to remove the Resource Manager stack.

The subnet refers to `oci_core_vcn.example_vcn.id`, which creates the Terraform dependency: Terraform/Resource Manager creates the VCN first and uses its resulting OCID when it creates the subnet.

QUESTION NO: 2

You are asked to deploy a new application that has been designed to scale horizontally. The business stakeholders have asked that the application be deployed in us-phoenix-1. Nor-mal usage requires 2 OCPUs. You expect to have few spikes during the week, that will require up to 4 OCPUs, and a major usage uptick at the end of each month that will require 8 OCPUs. What is the most cost-effective approach to implement a highly available and scalable solution? (Choose the best answer.)

A. Create an instance with 1 OCPU shape. Use the Resize Instance action to scale up to a larger shape when more resources are needed.

B. Create an instance with 1 OCPU shape. Use a CLI script to clone it when more re-sources are needed.

C. Create an instance pool with a VM.Standard2.2 shape instance configuration. Set up the autoscaling configuration to use 2 availability domains and have a minimum of 2 in-stances, to handle the weekly spikes, and a maximum of 4 instances.

D. Create an instance pool with a VM.Standard2.1 shape instance configuration. Set up the autoscaling configuration to use 2 availability domains and have a minimum of 2 in-stances and a maximum of 8 instances.

ANSWER: D

Explanation:

Create an instance pool with a VM.Standard2.1 shape instance configuration. Set up the autoscaling configuration to use 2 availability domains and have a minimum of 2 in-stances and a maximum of 8 instances. This design directly maps capacity to the stated demand profile because each VM.Standard2.1 instance provides one OCPU. Keeping two instances as the pool minimum supplies the required two OCPUs during ordinary operation while also providing redundancy across two availability domains. Autoscaling can add instances as demand grows, reaching four instances for the expected weekly demand and up to eight instances for the monthly peak. After demand declines, the pool can reduce its size back toward the

configured minimum, avoiding the cost of continuously running peak capacity. Instance pools are designed to manage multiple identically configured compute instances, and autoscaling adjusts pool size in response to defined performance metrics or schedules. Distributing the pool across availability domains improves application availability by reducing exposure to an availability-domain-level failure, while horizontal expansion preserves the application's intended scaling model. Oracle documents instance-pool autoscaling at Autoscaling Instance Pools and describes the available compute shapes at Compute Shapes.

QUESTION NO: 3

Your organization uses an Object Storage lifecycle policy to move infrequently accessed objects to the Archive storage tier. An operator now needs to download one of the archived objects immediately by using the OCI CLI. What must the operator do first? (Choose the best answer.)

A. Restore the archived object, then download it after the restoration is complete.

B. Change the bucket IAM policy to grant `manage objects`, then download the object directly.

C. Enable bucket versioning, then download the latest version of the object directly.

D. Copy the archived object to a new bucket, then download the copied object directly.

ANSWER: A

Explanation:

The operator must restore the archived object before downloading it. Objects in the Archive storage tier are not directly available for retrieval. A restore request makes an archived object available for access for the requested restoration period, after which the object remains in the Archive tier unless it is explicitly moved or managed by another lifecycle action.

Changing the object's IAM policy does not make archived data available, and copying the object cannot bypass the archive retrieval requirement. Enabling bucket versioning affects the handling of object versions and deletions, not the retrieval state of an archived object.

QUESTION NO: 4

Your compute instances are deployed in a private subnet and do not have public IP addresses. They must download software packages from an Oracle Cloud Infrastructure (OCI) Object Storage bucket without sending traffic through the public internet. The VCN has no Internet Gateway. Which solution should you implement? (Choose the best answer.)

A. Create a Service Gateway and route Object Storage service traffic from the private subnet to the Service Gateway.

B. Create a NAT Gateway and route Object Storage service traffic from the private subnet to the NAT Gateway.

C. Create an Internet Gateway and assign public IP addresses to all compute instances.

D. Create a Dynamic Routing Gateway and route Object Storage service traffic to the DRG.

ANSWER: A

Explanation:

Create a Service Gateway and add a route rule for the Object Storage service to the private subnet route table. A Service Gateway provides private access from a VCN to supported Oracle services, including Object Storage, without requiring public IP addresses, a NAT Gateway, or an Internet Gateway. The route rule uses the Object Storage service CIDR label as its destination and the Service Gateway as its target.

A NAT Gateway would allow private instances to reach public endpoints, but it is not required for private access to Object Storage through a Service Gateway. A Dynamic Routing Gateway is used for connectivity to on-premises networks or other VCNs, not for this service-access requirement.

QUESTION NO: 5 - (SIMULATION)

Scenario: 2 (Oracle Cloud-init and AutoScaling: Use cloud-init to Configure Apache on Instances in an Autoscaling Instance Pool)

Scenario Description: (Hands-On Performance Exam Certification)

You're deploying an Apache-based web application on OCI that requires horizontal autoscaling.

To configure instances upon provisioning, write a cloud-init script for Oracle Linux 8 that installs and enables Apache (httpd), and opens the firewall for HTTP on TCP port 80. Create an instance configuration and include the cloud-init script in it. Use this instance configuration to create an instance pool and autoscaling configuration.

Pre-Configuration:

To fulfill this requirement, you are provided with the following:

Access to an OCI tenancy, an assigned compartment, and OCI credentials

A VCN Cloud-Init Challenge VCN with an Internet gateway and a public subnet. The security list for the subnet allows ingress via TCP ports 22 and 80 (SSH and HTTP). The route table forwards all egress to the Internet gateway.

Access to the OCI Console

Required IAM policies

An SSH key pair for the compute instance

Public Key https://objectstorage.us-ashburn-1.oraclecloud.com/n/tenancyname/b/PBT_Storage/o/PublicKey.pub

Private Key https://objectstorage.us-ashburn-1.oraclecloud.com/n/tenancyname/b/PBT_Storage/o/PKey.key

Note: Throughout your exam, ensure to use assigned Compartment , User Name , and Region.

Complete the following tasks in the provisioned OCI environment:

Task 1(a): Develop the cloud-init Script:

Task 1(b): Use cloud-init to Configure Apache on Instances in an Autoscaling Instance Pool:

ANSWER: See the explanation for the answer

Explanation:

Use this cloud-init user-data script for both the standalone validation instance and the instance configuration. Cloud-init runs as root, so `sudo` is not required.

```
#!/bin/bash
dnf -y install httpd
systemctl enable httpd
systemctl start httpd
firewall-offline-cmd --add-service=http
systemctl restart firewalld
```

The required service name is `http` (not `https`), which opens TCP port 80 in the Oracle Linux firewall.

1. Create the validation instance

Name: `pbt_cloud_init_vm_01`
Image: Oracle Linux 8
Shape: `VM.Standard.A1.Flex`, 1 OCPU, 6 GB memory
Placement: any available AD
Network: public subnet `Cloud-Init Challenge SNT`, with a public IPv4 assigned
SSH key: paste/upload the provided public key.
Under **Advanced options** / **Cloud-init script**, paste the script above, then create the instance.

2. Create the instance configuration

Name: `pbt_cloud_init_config_01`
Use Oracle Linux 8 and `VM.Standard.A1.Flex` with 1 OCPU and 6 GB.
Use any AD, the `Cloud-Init Challenge SNT` public subnet, and assign a public IPv4.
Provide the supplied SSH public key.
In Advanced Options, add exactly the cloud-init script above.

3. Create the instance pool

Name: `pbt_cloud_init_pool_01`
Instance configuration: `pbt_cloud_init_config_01`
Initial instance count: `1`
Select any available AD; leave fault domain unselected/default.
Use the VCN and public subnet `Cloud-Init Challenge SNT`; do not attach a load balancer.

4. Create and attach the autoscaling configuration

Name: `pbt_cloud_autoscaling_config_01`
Instance pool: `pbt_cloud_init_pool_01`
Autoscaling type: **Metric-based autoscaling**
Cooldown: `300` seconds
Performance metric: **CPU utilization**
Scale-out rule: operator `>`, threshold `75%`, add `1` instance.
Scale-in rule: operator `<`, threshold `25%`, remove `1` instance.
Scaling limits: minimum `1`, maximum `2`, initial `1`.

After creation, each pool instance receives the user-data at first boot, installs and enables Apache, starts it, and permits HTTP through the guest OS firewall. The provided subnet security list already permits inbound TCP/80.

QUESTION NO: 6

Your organization wants to enforce consistent ownership metadata on resources created in project compartments. Which TWO statements are true about OCI defined tags and tag defaults? (Choose two.)

A. Defined tag keys can use validators to restrict allowed tag values.

B. Tag defaults can automatically apply defined tags to supported resources created in a compartment.

C. Free-form tags can be configured with a tag-key validator.

D. Creating a tag default automatically applies the tag to all existing resources in the compartment.

E. Defined tags can be used without first creating a tag namespace.

ANSWER: A B

Explanation:

Defined tag keys can include validators that restrict permitted tag values, such as allowing only values from an approved list or requiring a value to follow a specified pattern. This supports governance by preventing inconsistent values for a tag such as `Operations.CostCenter`.

Tag defaults can automatically apply defined tags to supported resources created in a compartment. This helps ensure that newly created resources receive required metadata without relying on every individual operator to add the tag manually.

Free-form tags are not governed by a tag namespace and tag-key definition, and simply applying tags does not retroactively add them to existing resources. Existing resources must be tagged separately if they require the new metadata.

QUESTION NO: 7

You are using Oracle Cloud Infrastructure (OCI) console to set up an alarm on a budget to track your OCI spending. Which two are valid targets for creating a budget in OCI? (Choose two.)

A. Select group as the type of target for your budget.

B. Select user as the type of target for your budget.

C. Select Compartment as the type of target for your budget.

D. Select Cost-Tracking Tags as the type of target for your budget.

E. Select Tenancy as the type of target for your budget.

ANSWER: C D

Explanation:

Select Compartment as the type of target for your budget. is correct because OCI Budgets can evaluate spending for resources within a chosen compartment. A compartment budget aggregates the applicable costs for that compartment, including resources in its subcompartments, making it useful for assigning accountability and controlling spend for a project, team, or workload boundary. The root compartment can also be selected when an organization needs a tenancy-wide view of costs.

Select Cost-Tracking Tags as the type of target for your budget. is also correct because OCI cost-tracking tags enable costs to be categorized across organizational boundaries that do not necessarily map to a single compartment. After a tag namespace and tag key are enabled for cost tracking, a budget can target a selected tag value. This supports chargeback or showback scenarios, such as monitoring expenditure for a department, application, or environment whose resources span multiple compartments. In either case, a budget amount and alert rules can be configured so OCI sends notifications as forecasted or actual spending reaches defined thresholds. Oracle documents these budget target choices and tag-based cost tracking in its Cost Management documentation. See Managing Budgets and Using Cost-Tracking Tags.

QUESTION NO: 8

You set up a bastion host in your VCN to only allow your IP address (140.19.2.140) to establish SSH connections to your Compute Instances that are deployed in a private subnet. The Compute Instances have an attached Network Security Group with a Source Type: Network Security Group (NSG), Source NSG: NSG-050504. To secure the bastion host, you added the following ingress rules to its Network Security Group:

However, after checking the bastion host logs, you discovered that there are IP addresses other than your own that can access your bastion host. What is the root cause of this issue? (Choose the best answer.)

A. The port 22 provides unrestricted access to 140.19.2.140 and to other IP address.

B. A netmask of /32 allows all IP address in the 140.19.2.0 network, other than your IP 140.19.2.140

C. All compute instances associated with NSG-050504 are also able to connect to the bastion host.

D. The Security List allows access to all IP address which overrides the Network Security Group ingress rules.

ANSWER: C

Explanation:

All compute instances associated with NSG-050504 are also able to connect to the bastion host. An OCI Network Security Group ingress rule whose source type is *Network Security Group* does not identify one specific public IP address. Instead, it permits matching traffic from the VNICs of every resource that belongs to the referenced source NSG. Therefore, if the bastion host's ingress configuration includes an SSH rule sourced from NSG-050504, any compute instance with a VNIC assigned to that NSG can initiate an SSH connection to the bastion host, subject to the protocol and port specified by the rule. Those connections appear in bastion logs using the instances' private IP addresses, which accounts for addresses other than 140.19.2.140. This behavior is intentional: NSG-to-NSG rules are designed to express application-tier trust relationships and dynamically apply as VNICs are added to or removed from the source NSG. To restrict administrative SSH access exclusively to the stated public address, the applicable SSH ingress rule must use a CIDR source of 140.19.2.140/32 rather than an NSG source. Oracle documents that an NSG can be used as the source or destination of a security rule and that the rule applies to resources' VNICs in that NSG. See Oracle Network Security Groups documentation and Oracle security rules documentation.

QUESTION NO: 9

You create an OCI Notifications topic for production alerts and add several email recipients. Which TWO statements are true about topics and subscriptions? (Choose two.)

A. A topic can have multiple subscriptions.

B. An email subscription must be confirmed before it receives published messages.

C. Creating a topic automatically subscribes all tenancy administrators.

D. Each topic can have only one email subscription.

E. An alarm cannot publish to a topic that has more than one subscription.

ANSWER: A B

Explanation:

An OCI Notifications topic can have multiple subscriptions, allowing the same alarm or event to notify several teams or integrations. Each subscription specifies an endpoint and protocol independently.

An email subscription must be confirmed before it can receive published messages. OCI sends a confirmation message to the specified email address, and the recipient must confirm the subscription. Until confirmation occurs, the subscription remains pending and does not receive normal topic notifications.

Creating a topic does not automatically subscribe all tenancy administrators, and an alarm can publish to a topic regardless of whether the topic uses only one or multiple confirmed subscriptions.