DUMPSHQ

# Free Questions for Terraform-Associate-003 by dumpshq

## Shared by Short on 10-01-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

# Question 1

**Question Type:** **MultipleChoice**

Which of the following is not a benefit of adopting infrastructure as code?

## Options:

**A-** Versioning

**B-** A Graphical User Interface

**C-** Reusability of code

**D-** Automation

## Answer:

B

## Explanation:

Infrastructure as Code (IaC) provides several benefits, including the ability to version control infrastructure, reuse code, and automate infrastructure management. However, IaC is typically associated with declarative configuration files and does not inherently provide a graphical user interface (GUI).A GUI is a feature that may be provided by specific tools or platforms built on top of IaC principles but is not a direct benefit of IaC itself1.

Reference= The benefits of IaC can be verified from the official HashiCorp documentation on "What is Infrastructure as Code with Terraform?" provided by HashiCorp Developer1.

# Question 2

**Question Type: MultipleChoice**

Which task does terraform init not perform?

## Options:

**A-** Validates all required variables are present

**B-** Sources any modules and copies the configuration locally

**C-** Connects to the backend

**D-** Sources all providers used in the configuration and downloads them

### Answer:

A

### Explanation:

Theterraform initcommand is used to initialize a working directory containing Terraform configuration files. This command performs several different initialization steps to prepare the current working directory for use with Terraform, which includes initializing the backend, installing provider plugins, and copying any modules referenced in the configuration.However, it does not validate whether all required variables are present; that is a task performed byterraform planorterraform apply1.

Reference= This information can be verified from the official Terraform documentation on theterraform initcommand provided by HashiCorp Developer1.

## Question 3

**Question Type: MultipleChoice**

Any user can publish modules to the public Terraform Module Registry.

## Options:

**A-** True

**B-** False

## Answer:

A

## Explanation:

The Terraform Registry allows any user to publish and share modules. Published modules support versioning, automatically generate documentation, allow browsing version histories, show examples and READMEs, and more. Public modules are managed via Git and GitHub, and publishing a module takes only a few minutes.Once a module is published, releasing a new version of a module is as simple as pushing a properly formed Git tag1.

Reference= The information can be verified from the Terraform Registry documentation on Publishing Modules provided by HashiCorp Developer1.

# Question 4

**Question Type:** **MultipleChoice**

Which of these commands makes your code more human readable?

## Options:

**A-** Terraform validate

**B-** Terraform output

**C-** Terraform show

**D-** Terraform fmt

## Answer:

D

## Explanation:

The command that makes your code more human readable isterraform fmt. This command is used to rewrite Terraform configuration files to a canonical format and style, following the Terraform language style conventions and other minor adjustments for readability. The command is optional, opinionated, and has no customization options, but it is recommended to ensure consistency of style across different Terraform codebases. Consistency can help your team understand the code more quickly and easily, making the use ofterraform fmtvery important. You can run this command on your configuration files before committing them to source control or as part of your CI/CD pipeline.Reference= :Command: fmt:Using Terraform fmt Command to Format Your Terraform Code

# Question 5

**Question Type:** **MultipleChoice**

Terraform configuration (including any module references) can contain only one Terraform provider type.

## Options:

**A-** True

**B-** False

## Answer:

B

## Explanation:

Terraform configuration (including any module references) can contain more than one Terraform provider type. Terraform providers are plugins that Terraform uses to interact with various cloud services and other APIs. A Terraform configuration can use multiple providers to manage resources across different platforms and services. For example, a configuration can use the AWS provider to create a virtual machine, the Cloudflare provider to manage DNS records, and the GitHub provider to create a repository. Terraform supports hundreds of providers for different use cases and scenarios.Reference= [Providers], [Provider Requirements], [Provider Configuration]

# Question 6

**Question Type: MultipleChoice**

Which of the following module source paths does not specify a remote module?

## Options:

**A-** Source = ''module/consul''

**B-** Source = ''githhub.comicrop/example''

**C-** Source =''git@github.com:hasicrop/example.git''

**D-** Source = ''hasicrop/consul/aws''

## Answer:

A

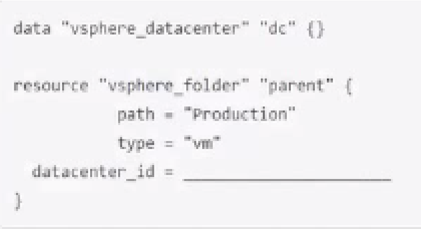
### Explanation:

The module source path that does not specify a remote module issource = 'module/consul'. This specifies a local module, which is a module that is stored in a subdirectory of the current working directory. The other options are all examples of remote modules, which are modules that are stored outside of the current working directory and can be accessed by various protocols, such as Git, HTTP, or the Terraform Registry. Remote modules are useful for sharing and reusing code across different configurations and environments.Reference= [Module Sources], [Local Paths], [Terraform Registry], [Generic Git Repository], [GitHub]

# Question 7

**Question Type:** **MultipleChoice**

How could you reference an attribute from the vsphere_datacenter data source for use with the datacenter_id argument within the vsphere_folder resource in the following configuration?



```
data "vsphere_datacenter" "dc" {}

resource "vsphere_folder" "parent" {
           path = "Production"
           type = "vm"
  datacenter_id = ______________________
}
```

## Options:

**A-** Data.vsphere_datacenter.DC.id

**B-** Vsphere_datacenter.dc.id

**C-** Data,dc,id

**D-** Data.vsphere_datacenter,dc

## Answer:

A

## Explanation:

The correct way to reference an attribute from the vsphere_datacenter data source for use with the datacenter_id argument within the vsphere_folder resource in the following configuration isdata.vsphere_datacenter.dc.id. This follows the syntax for accessing data source attributes, which isdata.TYPE.NAME.ATTRIBUTE. In this case, the data source type isvsphere_datacenter, the data source name isdc, and the attribute we want to access isid. The other options are incorrect because they either use the wrong syntax, the wrong punctuation, or the wrong case.Reference= [Data Source: vsphere_datacenter], [Data Source: vsphere_folder], [Expressions: Data Source Reference]

# Question 8

**Question Type:** **MultipleChoice**

Terraform providers are part of the Terraform core binary.

## Options:

**A-** True

**B-** False

## Answer:

B

## Explanation:

Terraform providers are not part of the Terraform core binary. Providers are distributed separately from Terraform itself and have their own release cadence and version numbers. Providers are plugins that Terraform uses to interact with various APIs, such as cloud providers, SaaS providers, and other services. You can find and install providers from the Terraform Registry, which hosts providers for most major infrastructure platforms. You can also load providers from a local mirror or cache, or develop your own custom providers. To use a provider in your Terraform configuration, you need to declare it in the provider requirements block and optionally configure its settings in the provider block.Reference= :Providers - Configuration Language | Terraform:Terraform Registry - Providers Overview | Terraform