TERRAFORM-ASSOCIATE-003^Q&As^

HashiCorp Certified: Terraform Associate (003)

Pass HashiCorp TERRAFORM-ASSOCIATE-003 Exam with 100% Guarantee

QUESTION 1

In a Terraform Cloud workpace linked to a version control repository speculative plan rum start automatically commit changes to version control.

A. True

B. False

Correct Answer: A

When you use a remote backend that needs authentication, HashiCorp recommends that you:

QUESTION 2

Your DevOps team is currently using the local backend for your Terraform configuration. You would like to move to a remote backend to store the state file in a central location. Which of the following backends would not work?

A. Artifactory

B. Amazon S3

C. Terraform Cloud

D. Git

Correct Answer: D

This is not a valid backend for Terraform, as it does not support locking or versioning of state files4. The other options are valid backends that can store state files in a central location.

QUESTION 3

You add a new provider to your configuration and immediately run terraform apply in the CD using the local backend. Why does the apply fail?

A. The Terraform CD needs you to log into Terraform Cloud first

B. Terraform requires you to manually run terraform plan first

C. Terraform needs to install the necessary plugins first

D. Terraform needs you to format your code according to best practices first

Correct Answer: C

The reason why the apply fails after adding a new provider to the configuration and immediately running terraform apply in the CD using the local backend is because Terraform needs to install the necessary plugins first. Terraform providers are plugins that Terraform uses to interact with various cloud services and other APIs. Each provider has a source address that determines where to download it from. When Terraform encounters a new provider in the configuration, it needs to run terraform init first to install the provider plugins in a local directory. Without the plugins, Terraform cannot communicate with the provider and perform the desired actions. References = [Provider Requirements], [Provider Installation]

QUESTION 4

Which configuration consistency errors does terraform validate report?

A. Terraform module isn\\'t the latest version

B. Differences between local and remote state

C. Declaring a resource identifier more than once

D. A mix of spaces and tabs in configuration files

Correct Answer: C

Terraform validate reports configuration consistency errors, such as declaring a resource identifier more than once. This means that the same resource type and name combination is used for multiple resource blocks, which is not allowed in Terraform. For example, resource "aws_instance" "example" {...} cannot be used more than once in the same configuration. Terraform validate does not report errors related to module versions, state differences, or formatting issues, as these are not relevant for checking the configuration syntax and structure. References = [Validate Configuration], [Resource Syntax]

QUESTION 5

You have provisioned some virtual machines (VMs) on Google Cloud Platform (GCP) using the gcloud command line tool. However, you are standardizing with Terraform and want to manage these VMs using Terraform instead. What are the two things you must do to achieve this? Choose two correct answers.

A. Run the terraform Import-gcp command

B. Write Terraform configuration for the existing VMs

C. Use the terraform import command for the existing VMs

D. Provision new VMs using Terraform with the same VM names

Correct Answer: BC

To import existing resources into Terraform, you need to do two things1:

Write a resource configuration block for each resource, matching the type and name used in your state file.

Run terraform import for each resource, specifying its address and ID. There is no such command as terraform Import-gcp, and provisioning new VMs with the same names will not import them into Terraform.

QUESTION 6

What value does the Terraform Cloud private registry provide over the public Terraform Module Registry?

A. The ability to share modules publicly with any user of Terraform

B. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations

C. The ability to tag modules by version or release

D. The ability to share modules with public Terraform users and members of Terraform Cloud Organizations

Correct Answer: B

The Terraform Cloud private registry provides the ability to restrict modules to members of Terraform Cloud or Enterprise organizations. This allows you to share modules within your organization without exposing them to the public. The private registry also supports importing modules from your private VCS repositories. The public Terraform Module Registry, on the other hand, publishes modules from public Git repositories and makes them available to any user of Terraform. References = : Private Registry - Terraform Cloud : Terraform Registry - Provider Documentation

QUESTION 7

A developer on your team is going lo leaf down an existing deployment managed by Terraform and deploy a new one. However, there is a server resource named aws instant.ubuntu[l] they would like to keep. What command should they use to tell Terraform to stop managing that specific resource?

A. Terraform plan rm:aws_instance.ubuntu[1]

B. Terraform state rm:aws_instance.ubuntu[1]

C. Terraform apply rm:aws_instance.ubuntu[1]

D. Terraform destory rm:aws_instance.ubuntu[1]

Correct Answer: B

To tell Terraform to stop managing a specific resource without destroying it, you can use the terraform state rm command. This command will remove the resource from the Terraform state, which means that Terraform will no longer track or update the corresponding remote object. However, the object will still exist in the remote system and you can later use terraform import to start managing it again in a different configuration or workspace. The syntax for this command is terraform state rm , where is the resource address that identifies the resource instance to remove. For example, terraform state rm aws_instance.ubuntu[1] will remove the second instance of the aws_instance resource named ubuntu from the state. References = : Command: state rm : Moving Resources

QUESTION 8

You are working on some new application features and you want to spin up a copy of your production deployment to perform some quick tests. In order to avoid having to configure a new state backend, what open source Terraform feature would allow you create multiple states but still be associated with your current code?

A. Terraform data sources

B. Terraform local values

C. Terraform modules

D. Terraform workspaces

E. None of the above

Correct Answer: D

Terraform workspaces allow you to create multiple states but still be associated with your current code. Workspaces are like "environments" (e.g. staging, production) for the same configuration. You can use workspaces to spin up a copy of your production deployment for testing purposes without having to configure a new state backend. Terraform data sources, local values, and modules are not features that allow you to create multiple states. References = Workspaces and How to Use Terraform Workspaces

QUESTION 9

Terraform variable names are saved in the state file.

A. True

B. False

Correct Answer: B

Terraform variable names are not saved in the state file, only their values are. The state file only stores the attributes of the resources and data sources that are managed by Terraform, not the variables that are used to configure them.

QUESTION 10

Which of the following commands would you use to access all of the attributes and details of a resource managed by Terraform?

A. terraform state list `provider_type.name\\'

B. terraform state show `provider_type.name\\'

C. terraform get `provider_type.name\\'

D. terraform state list

Correct Answer: B

The terraform state show command allows you to access all of the attributes and details of a resource managed by Terraform. You can use the resource address (e.g. provider_type.name) as an argument to show the information about a

specific resource. The terraform state list command only shows the list of resources in the state, not their attributes. The terraform get command downloads and installs modules needed for the configuration. It does not show any information

about resources. References = [Command:

state show] and [Command: state list]

QUESTION 11

What is one disadvantage of using dynamic blocks in Terraform?

A. Dynamic blocks can construct repeatable nested blocks

B. Terraform will run more slowly

C. They cannot be used to loop through a list of values

D. They make configuration harder to read and understand

Correct Answer: D

This is one disadvantage of using dynamic blocks in Terraform, as they can introduce complexity and reduce readability of the configuration. The other options are either advantages or incorrect statements.

QUESTION 12

When using multiple configuration of the same Terraform provider, what meta-argument must you include in any non-default provider configurations?

A. Alias

B. Id

C. Depends_on

D. name

Correct Answer: A

This is the meta-argument that you must include in any non-default provider configurations, as it allows you to give a friendly name to the configuration and reference it in other parts of your code. The other options are either invalid or irrelevant for this purpose.

QUESTION 13

You have a Terraform configuration that defines a single virtual machine with no references to it, You have run terraform apply to create the resource, and then removed the resource definition from your Terraform configuration file. What will happen you run terraform apply in the working directory again?

A. Terraform will remove the virtual machine from the state file, but the resource will still exist

B. Nothing

C. Terraform will error

D. Terraform will destroy the virtual machine

Correct Answer: D

This is what will happen if you run terraform apply in the working directory again, after removing the resource definition from your Terraform configuration file. Terraform will detect that there is a resource in the state file that is not present in the configuration file, and will assume that you want to delete it.

QUESTION 14

You have never used Terraform before and would like to test it out using a shared team account for a cloud provider. The shared team account already contains 15 virtual machines (VM). You develop a Terraform configuration containing one VM. perform terraform apply, and see that your VM was created successfully. What should you do to delete the newly-created VM with Terraform?

A. The Terraform state file contains all 16 VMs in the team account. Execute terraform destroy and select the newly-created VM.

B. Delete the Terraform state file and execute terraform apply.

C. The Terraform state file only contains the one new VM. Execute terraform destroy.

D. Delete the VM using the cloud provider console and terraform apply to apply the changes to the Terraform state file.

Correct Answer: C

This is the best way to delete the newly-created VM with Terraform, as it will only affect the resource that was created by your configuration and state file. The other options are either incorrect or inefficient.

QUESTION 15

It is best practice to store secret data in the same version control repository as your Terraform configuration.

A. True

B. False

Correct Answer: B

It is not a best practice to store secret data in the same version control repository as your Terraform configuration, as it could expose your sensitive information to unauthorized parties or compromise your security. You should use environment variables, vaults, or other mechanisms to store and provide secret data to Terraform.