ExamPro HashiCorp Terraform Review

Study online at https://quizlet.com/_eyv681

The Terraform registry can search based on the following search terms:	name and provider, Every page on the registry has a search field for finding modules. Enter any type of module you're looking for (examples: "vault", "vpc", "database"), and the resulting modules will be listed. The search query will look at module name, provider, and description to match your search terms. On the results page, filters can be used to further refine search results.
Which Terraform Workflow (Write -> Plan -> Apply) does this describe?	Correct. In the Team Workflow, infrastructure as code (IaC) is managed collaboratively using a version control system, where team members propose changes through pull requests. This allows for code review and collaborative improvements before changes are applied. The use of Terraform Cloud for running terraform apply when a pull request is approved highlights a centralized and automated approach to applying changes, ensuring consistency and traceability. This setup is indicative of a team environment where multiple contributors are working together on the same infrastructure codebase, which is a key characteristic of the Team Workflow. The project resides in a repo, and the backend is configured to use Terraform Cloud Pull requests are submitted to the repo with new changes When the Pull Request is approved, Terraform Cloud runs terraform apply
When we want the most verbose information from terraform logging what severity should we set?	Trace - Only when I would be "tracing" the code and trying to find one part of a function specifically. Debug - Information that is diagnostically helpful to people more than just developers (IT, sysadmins, etc.).
How does Terraform Cloud backup states?	Terraform Cloud saves a history of state files every time you perform a run https://www.terraform.io/docs/language/state/index.html
Is this a valid configuration for remote-exec? resource "aws_instance" "web" { # ... provisioner "remote-exec" { inline = ["puppet apply", "consul join ${aws_instance.web.private_ip}",] interpreter = ["bash", "-e"] } }	interpreter is not a valid argument for remote-exec. local-exec does have a valid argument called interpreter https://www.terraform.io/docs/language/resources/provisioners/remote-exec.html#argument-reference
The Terraform Registry contains both public and private providers and modules	The Terraform Registry only contains public providers and modules. https://www.terraform.io/docs/registry/private.html
A DevOps Engineer needs to reference an existing AMI (machine image) for an AWS Virtual machine called example. resource "aws_instance" "example" { ami = "ami-abc123" instance_type = "t2.micro" ebs_block_device { device_name = "sda2" volume_size = 16 } ebs_block_device { device_name = "sda3" volume_size = 20 } }	resource "aws_instance" "example2" { ami = aws_instance.example.ami This is correct because we are referencing the resource block
What type of backend is this Terraform configuration file using? terraform { } provider "aws" { region = "us-east-1" profile = "sandbox" } resource "aws_instance" "my_example_server" { ami = "ami-0c2b8ca1dad447f8a" instance_type = "t2.nano" tags = { Name = "MyExampleServer" } }	It is using the local backend. This is true, by default this will be used. local
How can you quickly start using Sentinel with Terraform Cloud?	Enable Sentinel in Terraform Cloud's organization settings and write Sentinel policies in the Terraform Cloud UI This is the correct answer. Sentinel can be enabled at the organization level in Terraform Cloud, allowing you to define and manage policies directly within the Terraform Cloud UI. You can write Sentinel policies using the Sentinel language, which is specifically designed for policy enforcement in Terraform configurations.
	Each Terraform module must declare which providers it requires, so that Terraform can install and use them. Provider requirements are declared in a required_providers block.

ExamPro HashiCorp Terraform Review

Study online at https://quizlet.com/_eyv681

The following is a valid configuration for a provider? terraform { providers{ aws = { source = "hashicorp/aws" version = "3.58.0" } } } provider "aws" { # Configuration options }	terraform { required_providers { aws = { source = "hashicorp/aws" version = "3.58.0" } } } provider "aws" { # Configuration options } https://www.terraform.io/docs/language/providers/requirements.html#requiring-providers
When you publish modules to the Terraform Registry what is their visibility?	All modules will be publicly available on the Internet without signing up or logging in. There is no option to make them private. Terraform Registry allows you to publish modules by connecting a GitHub public repository. You aren't required to signup, you can connect your GitHub account for the sole purpose of publishing a public repo. If you need private modules you can use Terraform Cloud which has a private registry.
Which of the following is NOT a built-in string function?	Slice is a built-in Collection function slice extracts some consecutive elements from within a list. > slice(["a", "b", "c", "d"], 1, 3) ["b", "c",]
A DevOps engineer needs to provision a resource that is not directly associated with a specific resource. How can they define their resource within their terraform configuration script?	resource "terraform_data" "cluster" { # ... } If you need to run provisioners that aren't directly associated with a specific resource, you can associate them with a terraform_data. https://www.terraform.io/docs/language/resources/provisioners/null_resource.html
A module needs to have the latest patches applied but not update the major or minor version. Which of the following will achieve this requirement?	~>: Allows only the rightmost version component to increment. For example, to allow new patch releases within a specific minor release, use the full version number: ~> 1.0.4 will allow installation of 1.0.5 and 1.0.10 but not 1.1.0. This is usually called the pessimistic constraint operator. https://www.terraform.io/docs/language/expressions/version-constraints.html
A DevOps Engineer is defining a security group within their Terraform Configuration. They need to define many ingress rules to allow port access any many different IP ranges and protocols. They want to dry up their code and use one of Terraform's meta-arguments to produce many nested blocks for ingress rules. Which type of block should they use?	A dynamic block acts much like a for expression, but produces nested blocks instead of a complex typed value. It iterates over a given complex value, and generates a nested block for each element of that complex value. https://www.terraform.io/docs/language/expressions/dynamic-blocks.html
When you run terraform plan it will also run terraform validate?	When you run terraform plan it includes a validation check. Its more lightweight to run terraform validate. https://www.terraform.io/docs/cli/commands/validate.html
What is the general order of a terraform lifecycle?	init > validate > fmt > plan > apply > destroy Validating the Terraform configuration before formatting is considered a better practice because it helps ensure that the configuration is syntactically correct and functionally accurate before applying any stylistic changes to the code.
When you want to remove a record tracking a remote object in your state file but have the remote object (eg. Azure Virtual Machine) to still exist, which command do you use?	terraform state rm Usage: terraform state rm [options] ADDRESS... Terraform will search the state for any instances matching the given resource address, and remove the record of each one so that Terraform will no longer be tracking the corresponding remote objects https://www.terraform.io/docs/cli/commands/state/rm.html

When troubleshooting terraform, when is it advised to set verbose logging? (Select 2)	In the HashiCorp Learn tutorial on debugging, its recommended for Core and Provider errors to turn on verbose logging and submit the issue to Github for the respected open-source project. https://learn.hashicorp.com/tutorials/terraform/troubleshooting-workflow?utm_source=WEBSITE&utm_medium=WEB_IO&utm_offer=ARTICLE_PAGE&utm_content=DOCS Language errors: The primary interface for Terraform is the HashiCorp Configuration Language (HCL), a declarative configuration language. The Terraform core application interprets the configuration language. When Terraform encounters a syntax error in your configuration, it prints out the line numbers and an explanation of the error. State errors: The Terraform state file stores information on provisioned resources. It maps resources to your configuration and tracks all associated metadata. If state is out of sync, Terraform may destroy or change your existing resources. After you rule out configuration errors, review your state. Ensure your configuration is in sync by refreshing, importing, or replacing resources. Core errors: The Terraform core application contains all the logic for operations. It interprets your configuration, manages your state file, constructs the resource dependency graph, and communicates with provider plugins. Errors produced at this level may be a bug. Later in this tutorial, you will learn best practices for opening a GitHub issue for the core development team. Provider errors: The provider plugins handle authentication, API calls, and mapping resources to services. Later in this tutorial, you will learn best practices for opening a GitHub issue for the provider development team.
When defining a data source block, how can we narrow down the resource we want to select from a remote provider?	The filter block allows a data source to select resources from a provider. data "aws_ami" "web" { filter { name = "state" values = ["available"] } filter { name = "tag:Component" values = ["web"] } } https://registry.terraform.io/providers/hashicorp/aws/latest/docs#argument-reference
When running terraform init, it will do the following:	1) Download plugin dependencies (https://www.terraform.io/docs/cli/commands/init.html), 2) Create a .terraform directory
In order to authenticate to Terraform Cloud what is recommended for local development?	Terraform Login command can be used to automatically obtain and save an API token for Terraform Cloud, Terraform Enterprise, or any other host that offers Terraform services. https://www.terraform.io/docs/cli/commands/login.html This is the recommended way to connect to terraform
How do Terraform backups work when using a local backend?	Terraform takes the current state and stores it in a file called terrraform.tfstate.backup Its not easy to find documentation for this feature, but if you test in practice you will see that this is how it works locally.
Does the terraform state file store sensitive values?	Yes, it can store sensitive state files, regardless of attributes applied Yes, when you use input variables the values may be stored within your state file depending on how your file is configured. You should always treat your state file as containing sensitive data. Its recommended to use a remote backend or Terraform Cloud to manage state.