HashiCorp.Terraform-Associate-004.v2026-02-24.q128

Exam Code:	Terraform-Associate-004
Exam Name:	HashiCorp Certified: Terraform Associate (004) (HCTA0-004)
Certification Provider:	HashiCorp
Free Question Number:	128
Version:	v2026-02-24
# of views:	108
# of Questions views:	1280
https://www.freepdfdumps.com/HashiCorp.Terraform-Associate-004.v2026-02-24.q128.html	

NEW QUESTION: 1
Which of these are features of HCP Terraform/Terraform Cloud? Pick the 2 correct responses below.
A. Automated infrastructure deployment visualization.
B. A web-based user interface (UI).
C. Automatic backups of configuration and state.
D. Remote state storage.
Answer: A,D (LEAVE A REPLY)
Automated Visualization: HCP Terraform providesvisualization toolsthat map infrastructure configurations, helping users manage complex architectures.
Remote State Storage: Terraform Cloud offersremote state management, essential for teams working collaboratively on shared infrastructure, ensuring consistency and avoiding state conflicts.
For more information, consultTerraform Cloud and HCP Terraform featuresin the official documentation.

NEW QUESTION: 2
Select the command that doesn't cause Terraform to refresh its state.
A. Terraform destroy
B. Terraform apply
C. Terraform plan
D. Terraform state list
Answer: D (LEAVE A REPLY)
This is the command that does not cause Terraform to refresh its state, as it only lists the resources that are currently managed by Terraform in the state file. The other commands will refresh the state file before performing their operations, unless you use the -refresh=false flag.

NEW QUESTION: 3
Which of the following does terraform apply change after you approve the execution plan? (Choose two.)
A. Cloud infrastructure
B. The .terraform directory
C. The execution plan
D. State file
E. Terraform code
Answer: A,D (LEAVE A REPLY)
The terraform apply command changes both the cloud infrastructure and the state file after you approve the execution plan. The command creates, updates, or destroys the infrastructure resources to match theconfiguration. It also updates the state file to reflect the new state of the infrastructure. The .
terraform directory, the execution plan, and the Terraform code are not changed by the terraform apply command. References = Command: apply and Purpose of Terraform State

NEW QUESTION: 4
You much initialize your working directory before running terraform validate.
A. True
B. False
Answer: A (LEAVE A REPLY)
You must initialize your working directory before running terraform validate, as it will ensure that all the required plugins and modules are installed and configured properly. If you skip this step, you may encounter errors or inconsistencies when validating your configuration files.

NEW QUESTION: 5
Terraform installs its providers during which phase?
A. Plan
B. Init
C. Refresh
D. All of the above
Answer: (SHOW ANSWER)
Terraforminstalls providers during the terraform init phase.
Providers aredownloaded and installedwhen running terraform init.
A (terraform plan)is incorrect because terraform plan only calculates changes but does not install providers.
C (terraform refresh)is incorrect because terraform refresh only updates the state but does not install providers.
Official Terraform Documentation Reference:
Provider Installation - HashiCorp Documentation

NEW QUESTION: 6
Variables declared within a module are accessible outside of the module.
A. True
B. False
Answer: B (LEAVE A REPLY)
Variables declared within a module are only accessible within that module, unless they are explicitly exposed as output values1.

NEW QUESTION: 7
When you initialize Terraform, where does it cache modules from the public Terraform Registry?
A. In the /tmp directory.
B. In the .terraform sub-directory.
C. In memory.
D. They are not cached.
Answer: B (LEAVE A REPLY)
Terraformdownloads modules and providersinto the .terraform directory inside the working directory.
A (/tmp/)- Incorrect; modules arenot stored temporarily.
C (In memory)- Incorrect; modules persist on disk.
D (Not cached)- Incorrect; Terraformdoes cache modulesfor efficiency.
Official Terraform Documentation Reference:
Terraform Module Caching

NEW QUESTION: 8
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?
A. Data.vsphere_datacenter.DC.id
B. Vsphere_datacenter.dc.id
C. Data,dc,id
D. Data.vsphere_datacenter,dc
Answer: A (LEAVE A REPLY)
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 is data.
vsphere_datacenter.dc.id. This follows the syntax for accessing data source attributes, which is data.TYPE.
NAME.ATTRIBUTE. In this case, the data source type is vsphere_datacenter, the data source name is dc, and the attribute we want to access is id. The other options are incorrect becausethey either use the wrong syntax, the wrong punctuation, or the wrong case. References = [Data Source: vsphere_datacenter], [Data Source:
vsphere_folder], [Expressions: Data Source References]

NEW QUESTION: 9
Terraform configuration can only import modules from the public registry.
A. True
B. False
Answer: (SHOW ANSWER)
Terraform configuration can import modules from various sources, not only from the public registry. Modules can be sourced from local file paths, Git repositories, HTTP URLs, Mercurial repositories, S3 buckets, and GCS buckets. Terraform supports a number of common conventions and syntaxes for specifying module sources, as documented in the [Module Sources] page. References = [Module Sources]

NEW QUESTION: 10
You can access state stored with the local backend by using terraform_remote_state data source.
A. True
B. False
Answer: (SHOW ANSWER)
You cannot access state stored with the local backend by using the terraform_remote_state data source. The terraform_remote_state data source is used to retrieve the root module output values from some other Terraform configuration using the latest state snapshot from the remote backend. It requires a backend that supports remote state storage, such as S3, Consul, AzureRM, or GCS. The local backend stores the state file locally on the filesystem, which terraform_remote_state cannot access.
References:
Terraform documentation on terraform_remote_state data source: Terraform Remote State Data Source Example usage of remote state: Example Usage (remote Backend)

NEW QUESTION: 11
Which of the following is not a valid Terraform variable type?
A. list
B. array
C. nap
D. string
Answer: B (LEAVE A REPLY)
This is not a valid Terraform variable type. The other options are valid variable types that can store different kinds of values2.

NEW QUESTION: 12
Which of the following should you add in the required_providers block to define a provider version constraint?
A. version ~> 3.1

B. version >= 3.1
C. version = ">= 3.1"
Answer: C (LEAVE A REPLY)
Rationale for Correct Answer (C):
Version constraints must be specified with = and quotes, e.g.:
version = ">= 3.1"
This ensures compatibility with Terraform's syntax.
Analysis of Incorrect Options:
A, B: Incorrect syntax, missing quotes or wrong operator format.
C: Correct Terraform syntax.
Key Concept:
Correct syntax in provider version constraints ensures reproducibility.
Reference:Terraform Exam Objective - Manage Terraform Resources and Providers.

NEW QUESTION: 13
What type of information can be found on the Terraform Registry when using published modules?
A. Required input variables.
B. Outputs.
C. Optional input variables and default values.
D. All of the above.
Answer: D (LEAVE A REPLY)
Rationale for Correct Answer (D):
The Terraform Registry provides documentation for published modules, including:
Required inputs (variables you must supply).
Optional inputs with defaults (so you know what you can override).
Outputs (what values the module returns for use elsewhere).
Therefore, the correct answer is all of the above.
Analysis of Incorrect Options:
A). Required inputs only: Incomplete.
B). Outputs only: Incomplete.
C). Optional inputs only: Incomplete.
D). All of the above: Correct because the Registry provides all relevant documentation.
Key Concept:
Terraform Registry modules include inputs, outputs, and usage details, enabling reusability and modular design.
Reference:Terraform Exam Objective - Interact with Terraform Modules.

NEW QUESTION: 14
You have created a main.tf Terraform configuration consisting of an application server, a database and a load balanced. You ran terraform apply and Terraform created all of the resources successfully.

Now you realize that you do not actually need the load balancer, so you run terraform destroy without any flags. What will happen?

A. Terraform will prompt you to pick which resource you want to destroy

B. Terraform will destroy the application server because it is listed first in the code

C. Terraform will prompt you to confirm that you want to destroy all the infrastructure

D. Terraform will destroy the main, tf file

E. Terraform will immediately destroy all the infrastructure

Answer: (SHOW ANSWER)

This is what will happen if you run terraform destroy without any flags, as it will attempt to delete all the resources that are associated with your current working directory or workspace. You can use the -target flag to specify a particular resource that you want to destroy.

NEW QUESTION: 15

Terraform providers are always installed from the Internet.

A. True

B. False

Answer: B (LEAVE A REPLY)

Terraform providers are not always installed from the Internet. There are other ways to install provider plugins, such as from a local mirror or cache, from a local filesystem directory, or from a network filesystem.

These methods can be useful for offline or air-gapped environments, or for customizing the installation process. You can configure the provider installation methods using the provider_installation block in the CLI configuration file.

NEW QUESTION: 16

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

A. Terraform state show ' provider_type_name

B. Terraform state list

C. Terraform get provider_type_name

D. Terraform state list provider_type_name

Answer: A (LEAVE A REPLY)

This is the command that you would use to access all of the attributes and details of a resource managed by Terraform, by providing the resource address as an argument. For example, terraform state show
'aws_instance.example' will show you all the information about the AWS instance named example.

NEW QUESTION: 17
Infrastructure as Code (IaC) can be stored in a version control system along with application code.
A. True
B. False
Answer: A (LEAVE A REPLY)
Infrastructure as Code (IaC) can indeed be stored in a version control system along with application code.
This practice is a fundamental principle of modern infrastructure management, allowing teams to apply software development practices like versioning, peer review, and CI/CD to infrastructure management.
Storing IaC configurations in version control facilitates collaboration, history tracking, and change management.References = While this concept is a foundational aspect of IaC and is widely accepted in the industry, direct references from the HashiCorp Terraform Associate (003) study materials were not found in the provided files. However, this practice is encouraged in Terraform's best practices and various HashiCorp learning resources.

NEW QUESTION: 18
What is the workflow for deploying new infrastructure with Terraform?
A. Write Terraform configuration, run terraform init to initialize the working directory orworkspace, and run terraform apply
B. Write Terraform configuration, run terraform show to view proposed changes, and terraform apply to create new infrastructure
C. Write Terraform configuration, run terraform apply to create infrastructure, use terraform validate to confirm Terraform deployed resources correctly
D. Write Terraform configuration, run terraform plan to initialize the working directory or workspace, and terraform apply to create the infrastructure
Answer: (SHOW ANSWER)
This is the workflow for deploying new infrastructure with Terraform, as it will create a plan and apply it to the target environment. The other options are either incorrect or incomplete.

NEW QUESTION: 19

Where can Terraform not load a provider from?

A. Plugins directory

B. Provider plugin chance

C. Official HashCrop Distribution on releases.hashcrop.com

D. Source code

Answer: D (LEAVE A REPLY)

This is where Terraform cannot load a provider from, as it requires a compiled binary file that implements the provider protocol. You can load a provider from a plugins directory, a provider plugin cache, or the official HashiCorp distribution on releases.hashicorp.com.

NEW QUESTION: 20

Which of these ate secure options for storing secrets for connecting to a Terraform remote backend? Choose two correct answers.

A. A variable file

B. Defined in Environment variables

C. Inside the backend block within the Terraform configuration

D. Defined in a connection configuration outside of Terraform

Answer: B,D (LEAVE A REPLY)

Environment variables and connection configurations outside of Terraform are secure options for storing secrets for connecting to a Terraform remote backend. Environment variables can be used to set values for input variables that contain secrets, such as backend access keys or tokens. Terraform will read environment variables that start with TF_VAR_ and match the name of an input variable. For example, if you have an input variable called backend_token, you can set its value with the environment variable TF_VAR_backend_token1. Connection configurations outside of Terraform are files or scripts that provide credentials or other information for Terraform to connect to a remote backend. For example, you can use a credentials file for the S3 backend2, or a shell script for the HTTP backend3. These files or scripts are not part of the Terraform configuration and can be stored securely in a separate location. The other options are not secure for storing secrets. A variable file is a file that contains values for input variables. Variable files are usually stored in the same directory as the Terraform configuration or in a version control system. This exposes the secrets to anyone who can access the files or the repository. You should not store secrets in variable files1. Inside the backend block within the Terraform configuration is where you specify the type and settings of the remote backend. The backend block is part of the Terraform configuration and is usually stored in a version control system. This exposes the secrets to anyone who can access the configuration or the repository. You should not store secrets in the backend block4. References = [Terraform Input Variables]1, [Backend Type: s3]2, [Backend Type: http]3, [Backend Configuration]4

NEW QUESTION: 21

What is a key benefit of the Terraform state file?

A. A state file can schedule recurring infrastructure tasks

B. A state file is a source of truth for resources provisioned with Terraform
C. A state file is a source of truth for resources provisioned with a public cloud console
D. A state file is the desired state expressed by the Terraform code files
Answer: B (LEAVE A REPLY)
This is a key benefit of the Terraform state file, as it stores and tracks the metadata and attributes of the resources that are managed by Terraform, and allows Terraform to compare the current state with the desired state expressed by your configuration files.

NEW QUESTION: 22
A developer launched a VM outside of the Terraform workflow and ended up with two servers with the same name. They are unsure which VM is managed with Terraform, but they do have a list of all active VM IDs.
Which method could you use to determine which instance Terraform manages?
A. Modify the Terraform configuration to add an import block for both of the virtual machines.
B. Run a terraform apply -refresh to identify the virtual machine IDs that are already managed by Terraform.
C. Run terraform state rm on both VMs, then terraform apply to recreate the correct one.
D. Run terraform state list to find the names of all VMs, then run terraform state show for each of them to find which VM ID Terraform manages.
Answer: D (LEAVE A REPLY)
Rationale for Correct Answer (D):
terraform state list shows all resources currently managed by Terraform. terraform state show <resource> provides detailed attributes, including the VM ID. This lets you match the Terraform-managed instance with the actual infrastructure.
Analysis of Incorrect Options:
A: Importing is not needed since one VM is already in state.
B: terraform apply doesn't show which VM ID is managed, it only refreshes attributes.
C: Removing state entries is destructive and may lead to losing state data unnecessarily.
Key Concept:
The state file is Terraform's source of truth for which resources it manages.
Reference:Terraform Exam Objective - Implement and Maintain State.

NEW QUESTION: 23
Which provider authentication method prevents credentials from being stored in the state file?
A. Using environment variables
B. Specifying the login credentials in the provider block
C. Setting credentials as Terraform variables
D. None of the above
Answer: D (LEAVE A REPLY)
None of the above methods prevent credentials from being stored in the state file. Terraform stores the provider configuration in the state file, which may include sensitive information such as

credentials. This is a potential security risk and should be avoided if possible. To prevent credentials from being stored in the state file, you can use one of the following methods:
Use environment variables to pass credentials to the provider. This way, the credentials are not part of the provider configuration and are not stored in the state file. However, this method may not work for some providers that require credentials to be set in the provider block.
Use dynamic credentials to authenticate with your cloud provider. This way, Terraform Cloud or Enterprise will request temporary credentials from your cloud provider for each run and use them to provision your resources. The credentials are not stored in the state file and are revoked after the run is completed. This method is supported for AWS, Google Cloud Platform, Azure, and Vault. References = : [Sensitive Values in State] : Authenticate providers with dynamic credentials

NEW QUESTION: 24
When you run terraform apply, the Terraform CLI will print output values from both the root module and any child modules.
A. True
B. False
Answer: (SHOW ANSWER)
Rationale for Correct Answer (True):
When terraform apply completes successfully, Terraform prints output values. Outputs from both root and child modules are displayed, but child module outputs must be explicitly exposed through the root module outputs to be visible at the CLI.
Analysis of Incorrect Option:
False: Incorrect, because Terraform does display output values, but only if they are exposed from child modules to the root module.
Key Concept:
Outputs help you extract important information (e.g., IP addresses, resource IDs) from your configuration.
Reference:Terraform Exam Objective - Read, Generate, and Modify Configurations.

NEW QUESTION: 25
How does Terraform determine dependencies between resources?
A. Terraform requires resource dependencies to be defined as modules and sourced in order
B. Terraform automatically builds a resource graph based on resources provisioners, special meta- parameters, and the stale file (if present}
C. Terraform requires resources in a configuration to be listed m the order they will be created to determine dependencies
D. Terraform requires all dependencies between resources to be specified using the depends_on parameter
Answer: B (LEAVE A REPLY)
This is how Terraform determines dependencies between resources, by using the references between them in the configuration files and other factors that affect the order of operations.

NEW QUESTION: 26
You add a new resource to an existing Terraform configuration, but do not update the version constraint in the configuration. The existing and new resources use the same provider. The working contains a .terraform.lock, hc1 file.
How will Terraform choose which version of the provider to use?
A. Terraform will use the version recorded in your lock file
B. Terraform will use the latest version of the provider for the new resource and the version recorded in the lock file to manage existing resources
C. Terraform will check your state file to determine the provider version to use
D. Terraform will use the latest version of the provider available at the time you provision your new resource
Answer: A (LEAVE A REPLY)
This is how Terraform chooses which version of the provider to use, when you add a new resource to an existing Terraform configuration, but do not update the version constraint in the configuration. The lock file records the exact version of each provider that was installed in your working directory, and ensures that Terraform will always use the same provider versions until you run terraform init -upgrade to update them.

NEW QUESTION: 27
You are using a networking module in your Terraform configuration with the name label my-network. In your main configuration you have the following code:
When you run terraform validate, you get the following error:
What must you do to successfully retrieve this value from your networking module?
A. Change the reference value to my-network,outputs,vmet_id
B. Define the attribute vmet_id as a variable in the networking modeule
C. Define the attribute vnet_id as an output in the networking module
D. Change the reference value module.my,network,outputs,vnet_id
Answer: (SHOW ANSWER)
This is what you must do to successfully retrieve this value from your networking module, as it will expose the attribute as an output value that can be referenced by other modules or resources. The error message indicates that the networking module does not have an output value named vnet_id, which causes the reference to fail.

NEW QUESTION: 28
You want to use API tokens and other secrets within your team's Terraform workspaces. Where does HashiCorp recommend you store these sensitive values? (Pick 3)
A. In a plaintext document on a shared drive.
B. In a terraform.tfvars file, checked into version control.
C. In a terraform.tfvars file, securely managed and shared with your team.
D. In an HCP Terraform/Terraform Cloud variable, with the sensitive option checked.

E. In HashiCorp Vault.

Answer: (SHOW ANSWER)

Rationale for Correct Answers:

C). terraform.tfvars securely managed: Acceptable if distributed securely outside version control.

D). HCP Terraform/Terraform Cloud sensitive variables: Official best practice for team-based workflows.

E). Vault: HashiCorp Vault is designed for secret management and integrates well with Terraform.

Analysis of Incorrect Options:

A: Storing plaintext secrets on shared drives is insecure.

B: Checking secrets into version control is a major security risk.

Key Concept:

Sensitive values should be managed securely in Vault, HCP Terraform, or securely shared .tfvars files - never in plaintext or version control.

Reference:Terraform Exam Objective - Use Terraform to Manage Infrastructure.

NEW QUESTION: 29

A developer accidentally launched a VM (virtual machine) outside of the Terraform workflow and ended up with two servers with the same name. They don't know which VM Terraform manages but do have a list of all active VM IDs.

Which of the following methods could you use to discover which instance Terraform manages?

A. Run terraform state list to find the names of all VMs, then run terraform state show for each of them to find which VM ID Terraform manages

B. Update the code to include outputs for the ID of all VMs, then run terraform plan to view the outputs

C. Run terraform taint/code on all the VMs to recreate them

D. Use terraform refresh/code to find out which IDs are already part of state

Answer: A (LEAVE A REPLY)

The terraform state list command lists all resources that are managed by Terraform in the current state file1. The terraform state show command shows the attributes of a single resource in the state file2. By using these two commands, you can compare the VM IDs in your list with the ones in the state file and identify which one is managed by Terraform.

NEW QUESTION: 30

A Terraform local value can reference other Terraform local values.

A. True

B. False

Answer: A (LEAVE A REPLY)

* Rationale for Correct answer:Terraform local values are evaluated together and can reference previously defined local values within the same locals block or earlier blocks.

* Analysis of Incorrect Options (Distractors):

* B: Incorrect, locals can reference other locals.

* Key Concept:Local values enable reuse and composition of expressions.
Reference:Terraform Exam Objective - Read, Generate, and Modify Configurations

NEW QUESTION: 31
Module version is required to reference a module on the Terraform Module Registry.
A. True
B. False
Answer: (SHOW ANSWER)
Module version is optional to reference a module on the Terraform Module Registry. If you omit the version constraint, Terraform will automatically use the latest available version of the module

NEW QUESTION: 32
You're writing a Terraform configuration that needs to read input from a local file called id_rsa.pub . Which built-in Terraform function can you use to import the file's contents as a string?
A. file("id_rsa.pub")
B. templaTefil("id_rsa.pub")
C. filebase64("id_rsa.pub")
D. fileset<"id_rsa.pub")
Answer: A (LEAVE A REPLY)
To import the contents of a local file as a string in Terraform, you can use the built-in file function. By specifying file("id_rsa.pub"), Terraform reads the contents of the id_rsa.pub file and uses it as a string within your Terraform configuration. This function is particularly useful for scenarios where you need to include file data directly into your configuration, such as including an SSH public key for provisioning cloud instances.
References = This information is a standard part of Terraform's functionality with built-in functions, as outlined in Terraform's official documentation and commonly used in various Terraform configurations.

NEW QUESTION: 33
Which of these actions are forbidden when the Terraform state file is locked? (Pick the 3 correct responses)

A. terraform apply
B. terraform state list
C. terraform destroy
D. terraform fmt
Answer: A,B,C (LEAVE A REPLY)
When the state file is locked, operations that modify or depend on the state (liketerraform apply,terraform destroy, andterraform state list) are blocked.terraform fmtonly formats the configuration files and does not interact with the state, so it is allowed.
References:
Terraform State Locking

NEW QUESTION: 34
What does terraform import do?
A. Imports existing resources into the state file
B. Imports all infrastructure from a given cloud provider
C. Imports a new Terraform module
D. Imports clean copies of tainted resources
E. None of the above
Answer: A (LEAVE A REPLY)
The terraform import command is used to import existing infrastructure into your Terraform state. This command takes the existing resource and associates it with a resource defined in your Terraform configuration, updating the state file accordingly. It does not generate configuration for the resource, only the state.

NEW QUESTION: 35
You are writing a child Terraform module that provisions an AWS instance. You want to reference the IP address returned by the child module in the root configuration. You name the instance resource "main'.
Which of these is the correct way to define the output value?
A. Option A
B. Option B
C. Option C
D. Option D
Answer: A (LEAVE A REPLY)
According to theofficial Terraform documentationhere:
#Terraform Docs - Declaring an Output Value
In Terraform 0.12 and later, you can directly use expressions in outputs without interpolation syntax:
output "instance_ip_addr" {
value = aws_instance.server.private_ip
}

This aligns perfectly withOption A:
output "instance_ip_addr" {
value = aws_instance.main.private_ip
}
Why not the others?
#Option B: Incorrect syntax. "${main.private_ip}" is referencing main as if it were a global variable or resource, but it isn't defined. Plus, the output name is oddly written as aws_instance.instance_ip_addr, which is not a valid identifier format.
#Option C: Although valid in older versions (<= 0.11), interpolation with "${}" isdeprecatedin Terraform 0.12
+. The docs clearly advise using direct expressions instead.
#Option D: Terraform hasno return statement; this is syntactically invalid in Terraform's HCL.

NEW QUESTION: 36
Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged into stdout.
A. True
B. False
Answer: (SHOW ANSWER)
Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged into stdout, along with other log levels such as TRACE, INFO, WARN, and ERROR. This can be useful for troubleshooting or debugging purposes.

NEW QUESTION: 37
A Terraform provider is NOT responsible for:
A. Exposing resources and data sources based on an APUI
B. Managing actions to take based on resources differences
C. Understanding API interactions with some service
D. Provisioning infrastructure in multiple
Answer: (SHOW ANSWER)
This is not a responsibility of a Terraform provider, as it does not make sense grammatically or logically. A Terraform provider is responsible for exposing resources and data sources based on an API, managing actions to take based on resource differences, and understanding API interactions with some service.

NEW QUESTION: 38
You ate creating a Terraform configuration which needs to make use of multiple providers, one for AWS and one for Datadog. Which of the following provider blocks would allow you to do this?
A)

```
terraform {
 provider "aws" {
   profile = var.aws_profile
   region  = var.aws_region
 }

provider "datadog" {
 api_key = var.datadog_api_key
   app_key = var.datadog_app_key
 }
}
```

B)

C)

D)

A. Option A

B. Option B

C. Option C

D. Option D

Answer: C (LEAVE A REPLY)

Option C is the correct way to configure multiple providers in a Terraform configuration. Each provider block must have a name attribute that specifies which provider it configures2. The other options are either missing the name attribute or using an invalid syntax.

NEW QUESTION: 39

A module can always refer to all variables declared in its parent module.

A. True

B. False

Answer: (SHOW ANSWER)

A module cannot always refer to all variables declared in its parent module, as it needs to explicitly declare input variables and assign values to them from the parent module's arguments. A module cannot access the parent module's variables directly, unless they are passed as input arguments.

NEW QUESTION: 40

What is a Terraform provider not responsible for?

A. Provisioning infrastructure in multiple cloud providers.

B. Managing actions to take based on resource differences.

C. Managing resources and data sources based on an API.
D. Understanding API interactions with a hosted service.
Answer: A (LEAVE A REPLY)
Rationale for Correct Answer (A):
Providers only interact with one specific platform or API. Terraform itself can work with multiple providers in one configuration, but a single provider is not responsible for provisioning across multiple cloud providers.
Analysis of Incorrect Options:
B). Managing actions to take based on resource differences: This is a provider responsibility (through the resource schema).
C). Managing resources and data sources based on an API: Correct - providers define resources/data sources and map them to API calls.
D). Understanding API interactions with a hosted service: Correct - providers encapsulate API logic to allow Terraform to manage resources.
Key Concept:
Providers are API adapters. They handle CRUD operations for resources in their own ecosystem but do not span across multiple cloud platforms.
Reference:Terraform Exam Objective - Manage Terraform Resources and Providers.

NEW QUESTION: 41
What does terraform destroy do?
A. Destroys all infrastructure in the Terraform state file.
B. Destroys all Terraform code files in the current directory, leaving the state file intact.
C. Destroys all infrastructure in the configured Terraform provider.
D. Destroys the Terraform state file, leaving the infrastructure intact.
Answer: A (LEAVE A REPLY)
* Rationale for Correct answer:terraform destroy removes all resources currently tracked in the Terraform state file by issuing destroy requests to the configured providers. It does not delete configuration files or providers themselves.
* Analysis of Incorrect Options (Distractors):
* B: Terraform never deletes configuration files.
* C: Terraform only destroys resources it manages, not all provider infrastructure.
* D: Terraform does not delete the state file by default.
* Key Concept:terraform destroy enforces removal of all managed infrastructure.
Reference:Terraform Exam Objective - Use Terraform to Manage Infrastructure

NEW QUESTION: 42
Module variable assignments are inherited from the parent module and you do not need to explicitly set them.
A. True
B. False

Answer: B (LEAVE A REPLY)
Module variable assignments are not inherited from the parent module and you need to explicitly set them using the source argument. This allows you to customize the behavior of each module instance.

NEW QUESTION: 43
_____backends support state locking.
A. All
B. No
C. Some
D. Only local
Answer: C (LEAVE A REPLY)
Some backends support state locking, which prevents other users from modifying the state file while a Terraform operation is in progress. This prevents conflicts and data loss. Not all backends support this feature, and you can check the documentation for each backend type to see if it does.

NEW QUESTION: 44
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
Answer: A (LEAVE A REPLY)
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.

NEW QUESTION: 45
Which of these commands makes your code more human readable?
A. Terraform validate
B. Terraform output
C. Terraform show
D. Terraform file
Answer: (SHOW ANSWER)
The command that makes your code more human readable is terraform 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 of terraform fmt very important. You can run this command on your configuration files before committing them to source control or as part of your CI/CD pipeline. References = : Command: fmt : Using Terraform fmt Command to Format Your Terraform Code

NEW QUESTION: 46
You decide to move a Terraform state file to Amazon S3 from another location. You write the code below into a file called backend.tf.

```
terraform {
  backend "s3" {
    bucket = "my-tf-bucket"
    region = "us-east-1"
  }
}
```

Which command will migrate your current state file to the new S3 remote backend?
A. terraform state
B. terraform init
C. terraform push
D. terraform refresh
Answer: B (LEAVE A REPLY)
This command will initialize the new backend and prompt you to migrate the existing state file to the new location3. The other commands are not relevant for this task.

NEW QUESTION: 47
A resource block is shown in the Exhibit section of this page. How would you reference the attribute name of this resource in HCL?
A. resource.kubernetes_namespace.example.name
B. kubernetes_namespace.example.name
C. data.kubernetes.namespace.name
D. kubernetes_namespace.test.name
Answer: B (LEAVE A REPLY)
In Terraform, the correct way to reference a resource attribute is:

pgsql
CopyEdit
resource_type.resource_name.attribute
For example, if the resource block is:
hcl
CopyEdit
resource "kubernetes_namespace" "example" {
metadata {
name = "my-namespace"
}
}
To reference the name attribute, use:
pgsql
CopyEdit
kubernetes_namespace.example.name
Explanation of incorrect answers:
A (resource.kubernetes_namespace.example.name)- Incorrect. Terraform does not use the resource. prefix when referencing resources.
C (data.kubernetes.namespace.name)- Incorrect. This syntax is used fordata sources, not resources.
D (kubernetes_namespace.test.name)- Incorrect. The resource name is "example", not "test".
Official Terraform Documentation Reference:
Terraform Resource References

NEW QUESTION: 48
HashiCorp Configuration Language (HCL) supports user-denned functions.
A. True
B. False
Answer: (SHOW ANSWER)
HashiCorp Configuration Language (HCL) does not support user-defined functions. You can only use the built-in functions that are provided by the language. The built-in functions allow you to perform various operations and transformations on values within expressions. The general syntax for function calls is a function name followed by comma-separated arguments in parentheses, such as max(5, 12, 9). You can find the documentation for all of the available built-in functions in the Terraform Registry orthe Packer Documentation, depending on which tool you are using.
References = : Functions - Configuration Language | Terraform : Functions - Configuration Language | Packer

NEW QUESTION: 49
Which of the following is not a way to trigger terraform destroy?
A. Using the destroy command with auto-approve.

B. Passing --destroy at the end of a plan request.
C. Running terraform destroy from the correct directory and then typing yes when prompted in the CLI.
Answer: (SHOW ANSWER)
Destroy Command Options: Theterraform destroycommand is the correct method to destroy resources, and it requires either manual approval or the -auto-approve flag.
Unsupported Triggering Method: The--destroy flagis not a recognized option for plan or apply commands, making B the correct answer as it isnota valid way to initiate resource destruction.
For more on destroying resources, refer to theterraform destroy command documentationin the Terraform CLI reference.

NEW QUESTION: 50
What task does the terraform import command perform?
A. Imports resources from one Terraform state file to another.
B. Imports existing resources into Terraform's state file.
C. Imports a new Terraform module into Terraform's state file.
D. Imports all infrastructure from the configured cloud provider.
E. Imports provider configuration from one state file to another.
Answer: B (LEAVE A REPLY)
Rationale for Correct Answer (B):
terraform import allows Terraform to associate existing resources (created outside of Terraform or by another tool) with a Terraform configuration by writing them into the state file. After import, Terraform can manage those resources.
Analysis of Incorrect Options:
A). From one state file to another: Incorrect, import does not transfer between state files.
C). Importing a module: Incorrect, modules are defined in configuration, not imported.
D). Import all infrastructure: Incorrect, import is per-resource, not bulk.
E). Provider configuration transfer: Incorrect, provider configs are in .tf files, not imported with this command.
Key Concept:
The terraform import command bridges existing resources with Terraform state management.
Reference:Terraform Exam Objective - Implement and Maintain State.

NEW QUESTION: 51
Which two steps are required to provision new infrastructure in the Terraform workflow? Choose two correct answers.
A. Plan
B. Import
C. Alidate
D. Init
E. apply

Answer: D,E (LEAVE A REPLY)
The two steps that are required to provision new infrastructure in the Terraform workflow are init and apply.
The terraform init command initializes a working directory containing Terraform configuration files. It downloads and installs the provider plugins that are needed for the configuration, and prepares the backend for storing the state. The terraform apply command applies the changes required to reach the desired state of the configuration, as described by the resource definitions in the configuration files. It shows a plan of the proposed changes and asks for confirmation before making any changes to the infrastructure. References =
[The Core Terraform Workflow], [Initialize a Terraform working directory with init], [Apply Terraform Configuration with apply]

NEW QUESTION: 52
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
Answer: C (LEAVE A REPLY)
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]

NEW QUESTION: 53
How can a ticket-based system slow down infrastructure provisioning and limit the ability to scale? Choose two correct answers.
A. End-users have to request infrastructure changes
B. Ticket based systems generate a full audit trail of the request and fulfillment process
C. Users can access catalog of approved resources from drop down list in a request form
D. The more resources your organization needs, the more tickets your infrastructure team has to process
Answer: A (LEAVE A REPLY)
These are some of the ways that a ticket-based system can slow down infrastructure provisioning and limit the ability to scale, as they introduce delays, bottlenecks, and manual interventions in the process of creating and modifying infrastructure.

NEW QUESTION: 54

How would you reference the volume IDs associated with the ebs_block_device blocks in this configuration?

```
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
 }
}
```


A. aws_instance.example.ebs_block_device[sda2,sda3).volume_id
B. aws_Instance.example.ebs_block_device.[*].volume_id
C. aws_Instance.example.ebs_block_device.volume_ids
D. aws_instance.example-ebs_block_device.*.volume_id
Answer: D (LEAVE A REPLY)
This is the correct way to reference the volume IDs associated with the ebs_block_device blocks in this configuration, using the splat expression syntax. The other options are either invalid or incomplete.

NEW QUESTION: 55
Which of these are features of Terraform Cloud? Choose two correct answers.
A. A web-based user interface (UI)
B. Automated infrastructure deployment visualization
C. Automatic backups
D. Remote state storage
Answer: A,D (LEAVE A REPLY)
Terraform Cloud includes several features designed to enhance collaboration and infrastructure management.
Two of these features are:

A web-based user interface (UI): This allows users to interact with Terraform Cloud through a browser, providing a centralized interface for managing Terraform configurations, state files, and workspaces.
Remote state storage: This feature enables users to store their Terraform state files remotely in Terraform Cloud, ensuring that state is safely backed up and can be accessed by team members as needed.

NEW QUESTION: 56

A resource block is shown in the Exhibit space of this page. How would you reference the name value of the second instance of this resource?
A. aws_instance.web[2].name
B. aws_instance.web.*.name
C. aws_instance.web[l].name
D. aws_instance.web[l]
E. element(aws_instance.web, 2)
Answer: (SHOW ANSWER)
* Rationale for Correct answer:When using count, Terraform creates resources indexed starting at 0.
With count = 2, the instances are indexed as:
* First instance # aws_instance.web[0]
* Second instance # aws_instance.web[1]
To reference the name attribute of the second instance, the correct syntax is:
aws_instance.web[1].name
* Analysis of Incorrect Options (Distractors):
* A. [2]: Incorrect because indexing starts at 0 and only indices 0 and 1 exist.
* B. *.name: Returns a list of all names, not a single instance value.
* D. [1]: References the whole resource object, not the name attribute.
* E. element(...): Incorrect syntax and wrong index; also unnecessary when direct indexing is available.
* Key Concept:Understanding resource indexing with count and zero-based indexing in Terraform.
Reference:Terraform Exam Objective - Read, Generate, and Modify Configurations

NEW QUESTION: 57
What is the name of the default file where Terraform stores the state?

Type your answer in the field provided. The text field is not case-sensitive and all variations of the correct answer are accepted.

Answer:

Terraform.tfstate

Explanation:

The name of the default file where Terraform stores the state is terraform.tfstate. This file contains a JSON representation of the current state of the infrastructure managed by Terraform. Terraform uses this file to track the metadata and attributes of the resources, and to plan and apply changes. By default, Terraform stores the state file locally in the same directory as the configuration files, but it can also be configured to store the state remotely in a backend.

References = [Terraform State], [State File Format]

NEW QUESTION: 58

Which of the following can you do with terraform plan? (Pick 2 correct responses)

A. Schedule Terraform to run at a planned time in the future.

B. View the execution plan and check if the changes match your expectations.

C. Save a generated execution plan to apply later.

D. Execute a plan in a different workspace.

Answer: B,C (LEAVE A REPLY)

Rationale for Correct Answers:

B). View the execution plan: Correct - terraform plan shows what Terraform intends to do (create, update, or destroy).

C). Save a generated execution plan: Correct - using terraform plan -out=planfile lets you save the plan to apply later with terraform apply planfile.

Analysis of Incorrect Options:

A). Schedule runs in the future: Incorrect, Terraform does not provide scheduling; external tools (like cron, CI

/CD) are required.

D). Execute a plan in a different workspace: Incorrect, execution happens in the current workspace; switching workspaces must be done before running the plan.

Key Concept:

terraform plan is used for previewing and optionally saving an execution plan for controlled and predictable infrastructure changes.

Reference:Terraform Exam Objective - Understand Terraform Basics and CLI.

NEW QUESTION: 59

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

A. True

B. False

Answer: A (LEAVE A REPLY)

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

NEW QUESTION: 60
Which Terraform command checks that your configuration syntax is correct?
A. terraform validate
B. terraform init
C. terraform show
D. terraform fmt
Answer: A (LEAVE A REPLY)
The terraform validate command is used to check that your Terraform configuration files are syntactically valid and internally consistent. It is a useful command for ensuring your Terraform code is error-free before applying any changes to your infrastructure.

NEW QUESTION: 61
What does Terraform use the .terraform.lock.hc1 file for?
A. There is no such file
B. Tracking specific provider dependencies
C. Preventing Terraform runs from occurring
D. Storing references to workspaces which are locked
Answer: B (LEAVE A REPLY)
The .terraform.lock.hcl file is a new feature in Terraform 0.14 that records the exact versions of each provider used in your configuration. This helps ensure consistent and reproducible behavior across different machines and runs.

NEW QUESTION: 62

When you use a remote backend that needs authentication, HashiCorp recommends that you:
A. Write the authentication credentials in the Terraform configuration files
B. Keep the Terraform configuration files in a secret store
C. Push your Terraform configuration to an encrypted git repository
D. Use partial configuration to load the authentication credentials outside of the Terraform code
Answer: D (LEAVE A REPLY)
This is the recommended way to use a remote backend that needs authentication, as it allows you to provide the credentials via environment variables, command-line arguments, or interactive prompts, without storing them in the Terraform configuration files.

NEW QUESTION: 63
Which of the following is not true of Terraform providers?
A. An individual person can write a Terraform Provider
B. A community of users can maintain a provider
C. HashiCorp maintains some providers
D. Cloud providers and infrastructure vendors can write, maintain, or collaborate on Terraform
E. providers
F. None of the above
Answer: (SHOW ANSWER)
All of the statements are true of Terraform providers. Terraform providers are plugins that enable Terraform to interact with various APIs and services1. Anyone can write a Terraform provider, either as an individual or as part of a community2. HashiCorp maintains some providers, such as the AWS, Azure, and Google Cloud providers3. Cloud providers and infrastructure vendors can also write, maintain, or collaborate on Terraform providers, such as the VMware, Oracle, and Alibaba Cloud providers. References =
*1: Providers - Configuration Language | Terraform | HashiCorp Developer
*2: Plugin Development - How Terraform Works With Plugins | Terraform | HashiCorp Developer
*3: Terraform Registry
*: Terraform Registry

NEW QUESTION: 64
When does Terraform create the .terraform.lock.hc1 file?
A. After your first terraform plan
B. After your first terraform apply
C. After your first terraform init
D. When you enable state locking
Answer: C (LEAVE A REPLY)
Terraform creates the .terraform.lock.hcl file after the first terraform init command. This lock file ensures that the dependencies for your project are consistent across different runs by locking the versions of the providers and modules used.

NEW QUESTION: 65
You have a simple Terraform configuration containing one virtual machine (VM) in a cloud provider. You run terraform apply and the VM is created successfully.
What will happen if you delete the VM using the cloud provider console, then run terraform apply again without changing any Terraform code?
A. Terraform will recreate the VM.
B. Terraform will report an error.
C. Terraform will remove the VM from the state file.
D. Terraform will not make any changes.
Answer: A (LEAVE A REPLY)
* Rationale for Correct answer:Terraform detects drift by refreshing state during terraform apply. Since the VM no longer exists but is still declared in configuration, Terraform will plan and recreate the missing resource.
* Analysis of Incorrect Options (Distractors):
* B. Error: Terraform treats this as drift, not an error.
* C. Remove from state: Terraform does not remove resources automatically unless instructed.
* D. No changes: Incorrect because drift is detected and corrected.
* Key Concept:Terraform enforces the desired state defined in configuration, correcting drift automatically.
Reference:Terraform Exam Objective - Navigate Terraform State and Backends

NEW QUESTION: 66
A module block is shown in the Exhibit space of this page. When you use a module block to reference a module from the Terraform Registry such as the one in the example, how do you specify version 1.0.0 of the module?
A. Append ?ref=v1.0.0 argument to the source path.
B. You cannot. Modules stored on the public Terraform Registry do not support versioning.
C. Add a version = "1.0.0" attribute to the module block.
D. Nothing. Modules stored on the public Terraform module Registry always default to version 1.0.0.
Answer: C (LEAVE A REPLY)
Module Versioning: To specify a version in a module block for modules in theTerraform Registry, you add the version attribute, e.g., version = "1.0.0".
Terraform Registry Support: The public registrysupports versioningby enabling semantic constraints, allowing users to define specific versions compatible with their infrastructure requirements.
Refer to themodule versioning documentationin Terraform's official registry guide.

NEW QUESTION: 67
You created infrastructure outside the Terraform workflow that you now want to manage using Terraform.

Which command brings the infrastructure into Terraform state?

A. terraform get

B. terraform refresh

C. terraform import

D. terraform init

Answer: C (LEAVE A REPLY)

The terraform import command allows Terraform to takeexistinginfrastructure andbring it under Terraform's management.

A (terraform get)is incorrect because it is used to fetch modules.

B (terraform refresh)is incorrect because it only updates Terraform's state to match the infrastructure but does not import resources.

D (terraform init)is incorrect because it only initializes the Terraform working directory.

Official Terraform Documentation Reference:

terraform import - HashiCorp Documentation

NEW QUESTION: 68

Which of the following should you add in the required_providers block to define a provider version constraint?

A. version

B. version = "3.1"

C. version: 3.1

D. version - 3.1

Answer: (SHOW ANSWER)

Rationale for Correct Answer (B):

Provider version constraints in Terraform are specified using the version argument in the required_providers block, e.g.:

```
terraform {
required_providers {
aws = {
source = "hashicorp/aws"
version = "3.1"
}
}
}
```

This ensures Terraform always uses a specific provider version (or constraint expression).

Analysis of Incorrect Options:

A). version: Incomplete, no value specified.

C). version: 3.1: Incorrect syntax, Terraform uses = not :.

D). version - 3.1: Incorrect syntax, - is invalid here.

Key Concept:

Defining provider version constraints ensures consistent provider behavior across environments and avoids breaking changes.
Reference:Terraform Exam Objective - Manage Terraform Resources and Providers.

NEW QUESTION: 69
Which of these are features of HCP Terraform/Terraform Cloud? (Pick the 2 correct responses)
A. Automatic backups of configuration and state.
B. Remote state storage.
C. Automated infrastructure deployment visualization.
D. A web-based user interface (UI).
Answer: B,D (LEAVE A REPLY)
Terraform Cloud provides features like remote state storage and a web-based user interface for managing your Terraform runs. While it offers robust infrastructure as code capabilities, automatic backups of configuration and state are not directly provided by Terraform Cloud; instead, the state is stored remotely and secured.
References:
Terraform Cloud Features

NEW QUESTION: 70
What type of block is used to construct a collection of nested configuration blocks?
A. Dynamic
B. For_each
C. Nesting
D. repeated.
Answer: (SHOW ANSWER)
This is the type of block that is used to construct a collection of nested configuration blocks, by using a for_each argument to iterate over a collection value and generate a nested block for each element. For example, you can use a dynamic block to create multiple ingress rules for a security group resource.

NEW QUESTION: 71
Which of these are benefits of using Sentinel with HCP Terraform/Terraform Cloud? (Pick the 3 correct responses)
A. You can enforce a list of approved AWS AMIs.
B. Sentinel Policies can be written in HashiCorp Configuration Language (HCL).
C. You can check out and check in cloud access keys.
D. Policy-as-code can enforce security best practices.
Answer: A,C,D (LEAVE A REPLY)
Sentinel is a policy-as-code framework that integrates with Terraform Cloud to enforce security, compliance, and governance rules. You can enforce rules such as approved AMIs and ensure security best practices.

Policies are written in the Sentinel language, not HCL.
References:
Sentinel Policies

NEW QUESTION: 72
When do you need to explicitly execute Terraform in refresh-only mode?
A. Before every terraform plan.
B. Before every terraform apply.
C. Before every terraform import.
D. None of the above.
Answer: C (LEAVE A REPLY)
Purpose of Refresh-Only Mode: Running Terraform inrefresh-only modeupdates Terraform's state file with the current state of resources in your infrastructure without making changes to the resources themselves.
Context of Terraform Import: When usingterraform import, you're adding existing resources to the state file, and running Terraform in refresh-only mode before this operation can ensure that any initial configuration syncs precisely with the actual state.
For more on refresh-only mode in relation to terraform import, refer to Terraform's import documentation.

NEW QUESTION: 73
What functionality do providers offer in Terraform?(Pick 3 correct responses)
A. Interact with cloud provider APIs.
B. Provision resources for on-premises infrastructure services.
C. Group a collection of Terraform configuration files that map to a single state file.
D. Provision resources for public cloud infrastructure services.
E. Enforce security and compliance policies.
Answer: (SHOW ANSWER)
A (#Correct)- Providers allow Terraform tointeract with APIsof cloud/on-premises services.
B (#Correct)- Some Terraform providers can provisionon-premises infrastructure, such as VMware, OpenStack, etc.
C (#Incorrect)- This describesTerraform Workspaces, not providers.
D (#Correct)- Terraform providers allow provisioning ofpublic cloud resources(AWS, Azure, GCP, etc.).
E (#Incorrect)- Enforcing security and compliance policies isnot a direct provider function, but it can be done using Sentinel or other policy-as-code tools.
Official Terraform Documentation Reference:
Terraform Providers

NEW QUESTION: 74

Which are forbidden actions when the terraform state file is locked? Choose three correct answers.

A. Terraform state list

B. Terraform destroy

C. Terraform validate

D. Terraform validate

E. Terraform for

F. Terraform apply

Answer: B,C,F (LEAVE A REPLY)

The terraform state file is locked when a Terraform operation that could write state is in progress. This prevents concurrent state operations that could corrupt the state. The forbidden actions when the state file is locked are those that could write state, such as terraform apply, terraform destroy, terraform refresh, terraform taint, terraform untaint, terraform import, and terraform state *. The terraform validate command is also forbidden, because it requires an initialized working directory with the state file. The allowed actions when the state file is locked are those that only read state, such as terraform plan, terraform show, terraform output, and terraform console.

References = [State Locking] and [Command: validate]

NEW QUESTION: 75

Which of these statements about HCP Terraform/Terraform Cloud workspaces is false?

A. They can securely store cloud credentials.

B. They have role-based access controls.

C. Plans and applies can be triggered via version control system integrations.

D. You must use the CLI to switch between workspaces.

Answer: (SHOW ANSWER)

In Terraform Cloud, you can switch between workspaces using both the web UI and CLI. The statement that you "must use the CLI" is false. Workspaces can securely store cloud credentials, offer role-based access control, and integrate with VCS to trigger plan and apply operations.

References:

Terraform Cloud Workspaces

NEW QUESTION: 76

What is the Terraform style convention for indenting a nesting level compared to the one above it?

A. With a tab

B. With two spaces

C. With four spaces

D. With three spaces

Answer: B (LEAVE A REPLY)

This is the Terraform style convention for indenting a nesting level compared to the one above it. The other options are not consistent with the Terraform style guide.

NEW QUESTION: 77
In a HCP Terraform/Terraform Cloud workspace linked to a version control repository, speculative plan runs start automatically when you merge or commit changes to version control.
A. True
B. False
Answer: A (LEAVE A REPLY)
Speculative Plans: Terraform Cloud'sspeculative planfeature runs automatically when changes are detected in a linked VCS repository, enabling users to review potential infrastructure changes without committing them.
Automatic Integration: This feature automates the planning process by triggering when changes are committed, aiding teams in previewing infrastructure changes seamlessly.
For further understanding, see theTerraform Cloud VCS Integrationdocumentation.

NEW QUESTION: 78
What are some benefits of using Sentinel with Terraform Cloud/Terraform Cloud? Choose three correct answers.
A. You can restrict specific resource configurations, such as disallowing the use of CIDR=0.0.0.0/0.
B. You can check out and check in cloud access keys
C. Sentinel Policies can be written in HashiCorp Configuration Language (HCL)
D. Policy-as-code can enforce security best practices
E. You can enforce a list of approved AWS AMIs
Answer: A,D,E (LEAVE A REPLY)
Sentinel is a policy-as-code framework that allows you to define and enforce rules on your Terraform configurations, states, and plans1. Some of the benefits of using Sentinel with Terraform Cloud/Terraform Enterprise are:
*You can restrict specific resource configurations, such as disallowing the use of CIDR=0.0.0.0/0, which would open up your network to the entire internet. This can help you prevent misconfigurations or security vulnerabilities in your infrastructure2.

*Policy-as-code can enforce security best practices, such as requiring encryption, authentication, or compliance standards. This can help you protect your data and meet regulatory requirements3.
*You can enforce a list of approved AWS AMIs, which are pre-configured images that contain the operating system and software you need to run your applications. This can help you ensure consistency, reliability, and performance across your infrastructure4.
References =
*1: Terraform and Sentinel | Sentinel | HashiCorp Developer
*2: Terraform Learning Resources: Getting Started with Sentinel in Terraform Cloud
*3: Exploring the Power of HashiCorp Terraform, Sentinel, Terraform Cloud ...
*4: Using New Sentinel Features in Terraform Cloud - Medium

NEW QUESTION: 79
You have multiple team members collaborating on infrastructure as code (IaC) using Terraform, and want to apply formatting standards for readability.
How can you format Terraform HCL (HashiCorp Configuration Language) code according to standard Terraform style convention?
A. Run the terraform fmt command during the code linting phase of your CI/CD process Most Voted
B. Designate one person in each team to review and format everyone's code
C. Manually apply two spaces indentation and align equal sign "=" characters in every Terraform file (*.tf)
D. Write a shell script to transform Terraform files using tools such as AWK, Python, and sed
Answer: A (LEAVE A REPLY)
The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.
This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability. Running this command on your configuration files before committing them to source control can help ensure consistency of style between different Terraform codebases, and can also make diffs easier to read. You can also use the -check and -diff options to check if the files are formatted and display the formatting changes respectively2.
Running the terraform fmt command during the code linting phase of your CI/CD process can help automate this process and enforce the formatting standards for your team. References = [Command: fmt]2

NEW QUESTION: 80
What is the Terraform style convention for indenting a nesting level compared to the one above it?
A. With two spaces.
B. With four spaces.
C. With three spaces.
D. With a tab.

Answer: A (LEAVE A REPLY)
Terraform's Indentation Standards: Terraform's style convention usestwo spaces per nesting levelfor readability, helping to maintain uniform code across teams.
Configuration Files: Consistent indentation is crucial for Terraform's HCL syntax, as it improves readability and avoids parsing issues.
More details are available in theTerraform configuration style guide.

NEW QUESTION: 81
You ate making changes to existing Terraform code to add some new infrastructure. When is the best time to run terraform validate?
A. After you run terraform apply so you can validate your infrastructure
B. Before you run terraform apply so you can validate your provider credentials
C. Before you run terraform plan so you can validate your code syntax
D. After you run terraform plan so you can validate that your state file is consistent with your infrastructure
Answer: C (LEAVE A REPLY)
This is the best time to run terraform validate, as it will check your code for syntax errors, typos, and missing arguments before you attempt to create a plan. The other options are either incorrect or unnecessary.

NEW QUESTION: 82
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
Answer: (SHOW ANSWER)
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]

NEW QUESTION: 83
Running terraform fmt without any flags in a directory with Terraform configuration files will check the formatting of those files, but will never change their contents.
A. True
B. False

Answer: B (LEAVE A REPLY)
* Rationale for Correct answer:By default, terraform fmt modifies files in place to match Terraform's canonical formatting. The -check flag is required to only check formatting without making changes.
* Analysis of Incorrect Options (Distractors):
* A: Incorrect because formatting changes are applied automatically by default.
* Key Concept:terraform fmt enforces consistent formatting by rewriting files unless instructed otherwise.
Reference:Terraform Exam Objective - Understand Terraform Basics and CLI

NEW QUESTION: 84
In a Terraform Cloud workspace linked to a version control repository, speculative plan runs start automatically when you merge or commit changes to version control.
A. True
B. False
Answer: B (LEAVE A REPLY)
In Terraform Cloud, speculative plan runs are not automatically started when changes are merged or committed to the version control repository linked to a workspace. Instead, speculative plans are typically triggered as part of proposed changes in merge requests or pull requests to give an indication of what would happen if the changes were applied, without making any real changes to the infrastructure. Actual plan and apply operations in Terraform Cloud workspaces are usually triggered by specific events or configurations defined within the Terraform Cloud workspace settings.References = This behavior is part of how Terraform Cloud integrates with version control systems and is documented in Terraform Cloud's usage guidelines and best practices, especially in the context of VCS-driven workflows.

NEW QUESTION: 85
What is the workflow for deploying new infrastructure with Terraform?
A. Write Terraform configuration, run terraform plan to initialize the working directory or workspace, and terraform apply to create the infrastructure
B. Write Terraform configuration, run terraform init to initialize the working directory orworkspace, and run terraform apply
C. Write Terraform configuration, run terraform show to view proposed changes, and terraform apply to create new infrastructure
D. Write Terraform configuration, run terraform apply to create infrastructure, use terraform validate to confirm Terraform deployed resources correctly
Answer: (SHOW ANSWER)
This is the workflow for deploying new infrastructure with Terraform, as it will create a plan and apply it to the target environment. The other options are either incorrect or incomplete.

NEW QUESTION: 86

A Terraform backend determines how Terraform loads state and stores updates when you execute which command?

A. apply

B. destroy

C. Both of these are correct.

D. Neither of these are correct.

Answer: (SHOW ANSWER)

A Terraformbackenddetermineswhere and how Terraform state is stored.

Bothterraform applyandterraform destroyinteract with state, so Terraform needs access to the backend for state storage and updates.

D (Neither are correct)is incorrect becauseTerraform state is required for both apply and destroy operations.

Official Terraform Documentation Reference:

Terraform Backends

NEW QUESTION: 87

Which is the best way to specify a tag of v1.0.0 when referencing a module stored in Git (for example.

Git::https://example.com/vpc.git)?

A. Append pref=v1.0.0 argument to the source path

B. Add version = "1.0.0" parameter to module block

C. Nothing modules stored on GitHub always default to version 1.0.0

Answer: A (LEAVE A REPLY)

The best way to specify a tag of v1.0.0 when referencing a module stored in Git is to append ? ref=v1.

0.0 argument to the source path. This tells Terraform to use a specific Git reference, such as a branch, tag, or commit, when fetching the module source code. For example, source = "git::https://example.com/vpc.git?

ref=v1.0.0". This ensures that the module version is consistent and reproducible across different environments. References = [Module Sources], [Module Versions]

NEW QUESTION: 88

Which of the following ate advantages of using infrastructure as code (laC) instead of provisioning with a graphical user interface (GUI)? Choose two correct answers.

A. Lets you version, reuse, and share infrastructure configuration

B. Provisions the same resources at a lower cost

C. Secures your credentials

D. Reduces risk of operator error

E. Prevents manual modifications to your resources

Answer: (SHOW ANSWER)

It lets you version, reuse, and share infrastructure configuration as code files, which can be stored in a source control system and integrated with your CI/CD pipeline.
It reduces risk of operator error by automating repetitive tasks and ensuring consistency across environments.
IaC does not necessarily provision resources at a lower cost, secure your credentials, or prevent manual modifications to your resources - these depend on other factors such as your cloud provider, your security practices, and your access policies.

NEW QUESTION: 89
Which Terraform collection type should you use to store key/value pairs?
A. Set
B. Map
C. Tuple
D. list
Answer: B (LEAVE A REPLY)
The Terraform collection type that should be used to store key/value pairs is map. A map is a collection of values that are accessed by arbitrary labels, called keys. The keys and values can be of any type, but the keys must be unique within a map. For example, var = { key1 = "value1", key2 = "value2" } is a map with two key
/value pairs. Maps are useful for grouping related values together, such as configuration options or metadata. References = [Collection Types], [Map Type Constraints]

NEW QUESTION: 90
Terraform configuration can only call modules from the public registry.
A. True
B. False
Answer: B (LEAVE A REPLY)
Terraform can call modules from various sources including the public Terraform Registry, private registries, local file paths, or version control systems like GitHub.
References:
Terraform Modules

NEW QUESTION: 91
The_____determines how Terraform creates, updates, or delete resources.
A. Terraform configuration
B. Terraform provisioner
C. Terraform provider
D. Terraform core
Answer: C (LEAVE A REPLY)

This is what determines how Terraform creates, updates, or deletes resources, as it is responsible for understanding API interactions with some service and exposing resources and data sources based on that API.

NEW QUESTION: 92
You modified your Terraform configuration to fix a typo in the resource ID by renaming it from photoes to photos. What configuration will you add to update the resource ID in state without destroying the existing resource?
Original configuration:
resource "aws_s3_bucket" "photoes" {
bucket_prefix = "images"
}
Updated configuration:
resource "aws_s3_bucket" "photos" {
bucket_prefix = "images"
}
A. moved {
from = aws_s3_bucket.photoes
to = aws_s3_bucket.photos
}
B. moved {
bucket.photoes = aws_s3_bucket.photos
}
C. moved {
aws_s3_bucket.photoes = aws_s3_bucket.photos
}
D. None. Terraform will automatically update the resource ID.
Answer: (SHOW ANSWER)
Rationale for Correct Answer (A):
Terraform does not automatically update state references when resource identifiers are renamed. Instead, you must use a moved block in your configuration to inform Terraform how to map the

old resource to the new one. This prevents Terraform from destroying and recreating the resource.
Analysis of Incorrect Options:
B & C: Incorrect syntax - moved requires from and to.
D: Incorrect, Terraform won't auto-detect renames and will plan to destroy and recreate the resource unless a moved block is provided.
Key Concept:
The moved block is essential for refactoring resource names without losing resources in state.
Reference:Terraform Exam Objective - Implement and Maintain State.

NEW QUESTION: 93
Terraform providers are part of the Terraform core binary.
A. True
B. False
Answer: B (LEAVE A REPLY)
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. References = : Providers - Configuration Language | Terraform : Terraform Registry - Providers Overview | Terraform

NEW QUESTION: 94
What does the default "local" Terraform backend store?
A. tfplan files
B. State file
C. Provider plugins
D. Terraform binary
Answer: B (LEAVE A REPLY)
The default "local" Terraform backend stores the state file in a local file named terraform.tfstate, which can be used to track and manage the state of your infrastructure3.

NEW QUESTION: 95
What is the purpose of the terraform.lock.hcl file in Terraform?
A. There is no such file.
B. Storing references to workspaces, which are locked.
C. Preventing Terraform runs from occurring.
D. Tracking specific provider dependencies.

Answer: D (LEAVE A REPLY)

Rationale for Correct Answer (D):

The terraform.lock.hcl file is automatically created and maintained by Terraform. It records the specific versions and checksums of providers used in a configuration, ensuring consistent runs across environments and machines.

Analysis of Incorrect Options:

A). There is no such file: Incorrect - the lock file exists and is crucial for dependency management.

B). Storing references to workspaces: Incorrect - workspaces are tracked in the state file, not the lock file.

C). Preventing Terraform runs: Incorrect - the lock file does not block runs; it enforces consistent provider versions.

Key Concept:

The terraform.lock.hcl file enforces provider version consistency, which is critical for reproducible and reliable Terraform deployments.

Reference:Terraform Exam Objective - Read, Generate, and Modify Configurations.

NEW QUESTION: 96

Which of the following is true about terraform apply?(Pick 2 correct responses)

A. You must pass the output of a terraform plan command to it.

B. By default, it does not refresh your state file to reflect the current infrastructure configuration.

C. Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources.

D. You cannot target specific resources for the operation.

E. It only operates on infrastructure defined in the current working directory or workspace.

Answer: C,E (LEAVE A REPLY)

C (#Correct)- If changes require aresource replacement(e.g., changing an immutable attribute like instance type), Terraform willdestroy and recreate the resource.

E (#Correct)- Terraform only appliesthe configuration in the current directory or workspace.

NEW QUESTION: 97

Why does this backend configuration not follow best practices?

```
terraform {
  backend "s3" {
    bucket     = "terraform-state-prod"
    key        = "network/terraform.tfstate"
    region     = "us-east-1"
    access_key = "AKIAIOSFODNN7EXAMPLE"
    secret_key = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
  }

  required_providers {
       aws = {
         source  = "hashicorp/aws"
         version = "~> 3.38"
       }
     }

     required_version = ">= 0.15"
}
```


A. An alias meta-argument should be included in backend blocks whenever possible
B. You should use the local enhanced storage backend whenever possible
C. You should not store credentials in Terraform configuration
D. The backend configuration should contain multiple credentials so that more than one user can execute terraform plan and terraform apply
Answer: C (LEAVE A REPLY)
This is a bad practice, as it exposes your credentials to anyone who can access your configuration files or state files. You should use environment variables, credential files, or other mechanisms to provide credentials to Terraform.

NEW QUESTION: 98
When do changes invoked by terraform apply take effect?
A. After Terraform has updated the state file
B. Once the resource provider has fulfilled the request
C. Immediately
D. None of the above are correct
Answer: B (LEAVE A REPLY)
Changes invoked by terraform apply take effect once the resource provider has fulfilled the request, not after Terraform has updated the state file or immediately. The state file is only a reflection of the real resources, not a source of truth.

NEW QUESTION: 99
How is terraform import run?
A. As a part of terraform init
B. As a part of terraform plan
C. As a part of terraform refresh

D. By an explicit call
E. All of the above
Answer: (SHOW ANSWER)
The terraform import command is not part of any other Terraform workflow. It must be explicitly invoked by the user with the appropriate arguments, such as the resource address and the ID of the existing infrastructure to import. References = [Importing Infrastructure]

NEW QUESTION: 100
You have a simple Terraform configuration containing one virtual machine (VM) in a cloud provider. You run terraform apply and the VM is created successfully. What will happen if you terraform apply again immediately afterward without changing any Terraform code?
A. Terraform will terminate and recreate the VM.
B. Terraform will create another duplicate VM.
C. Terraform will apply the VM to the state file.
D. Nothing
Answer: (SHOW ANSWER)
Terraform follows adeclarativeapproach, meaning it ensures the infrastructure matches the configuration.
If you run terraform apply againwithout any changes, Terraform willdetect that the infrastructure is already up to dateand willdo nothing.
The command will complete successfully with the message"No changes. Your infrastructure matches the configuration." Official Terraform Documentation Reference:
Terraform Apply - HashiCorp Documentation

NEW QUESTION: 101
When using Terraform to deploy resources into Azure, which scenarios are true regarding state files? (Choose two.)
A. When you change a Terraform-managed resource via the Azure Cloud Console, Terraform updates the state file to reflect the change during the next plan or apply
B. Changing resources via the Azure Cloud Console records the change in the current state file
C. When you change a resource via the Azure Cloud Console, Terraform records the changes in a new state file
D. Changing resources via the Azure Cloud Console does not update current state file
Answer: A,D (LEAVE A REPLY)
Terraform state is a representation of the infrastructure that Terraform manages. Terraform uses state to track the current status of the resources it creates and to plan future changes. However, Terraform state is not aware of any changes made to the resources outside of Terraform, such as through the Azure Cloud Console, the Azure CLI, or the Azure API. Therefore, changing resources via the Azure Cloud Console does not update the current state file, and it may cause inconsistencies or conflicts with Terraform's desired configuration. To avoid this, it is

recommended to manage resources exclusively through Terraform or to use the terraform import command to bring existing resources under Terraform's control.
When you change a Terraform-managed resource via the Azure Cloud Console, Terraform does not immediately update the state file to reflect the change. However, the next time you run terraform plan or terraform apply, Terraform will compare the state file with the actual state of the resources in Azure and detect any drifts or differences. Terraform will then update the state file to match the current state of the resources and show you the proposed changes in the execution plan. Depending on the configuration and the change, Terraform may try to undo the change, modify the resource further, or recreate the resource entirely.
To avoid unexpected or destructive changes, it is recommended to review the execution plan carefully before applying it or to use the terraform refresh command to update the state file without applying any changes.
References = Purpose of Terraform State, Terraform State, Managing State, Importing Infrastructure,
[Command: plan], [Command: apply], [Command: refresh]

NEW QUESTION: 102
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.
Answer: C (LEAVE A REPLY)
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.

NEW QUESTION: 103
If a module declares a variable with a default, that variable must also be defined within the module.
A. True
B. False
Answer: B (LEAVE A REPLY)

A module can declare a variable with a default value without requiring the caller to define it. This allows the module to provide a sensible default behavior that can be customized by the caller if needed. References =
[Module Variables]

NEW QUESTION: 104
One cloud block always maps to a single HCP Terraform/Terraform Cloud workspace.
A. True
B. False
Answer: (SHOW ANSWER)
Rationale for Correct Answer (True):
A cloud block in Terraform configuration specifies a single Terraform Cloud or HCP Terraform workspace.
You cannot use one cloud block for multiple workspaces.
Analysis of Incorrect Option:
False: Incorrect because a cloud block is a one-to-one mapping with a single workspace.
Key Concept:
Cloud blocks manage remote operations and backend configuration tied to one workspace.
Reference:Terraform Exam Objective - Manage Terraform Workspaces and Cloud.

NEW QUESTION: 105
Terraformrequiresthe Go runtime as a prerequisite for installation.
A. True
B. False
Answer: B (LEAVE A REPLY)
Terraformis written in Go, but it isdistributed as a standalone binaryanddoes notrequire the Go runtime.
Users do not need to install Go to run Terraform.
Official Terraform Documentation Reference:
Terraform Install Guide

NEW QUESTION: 106
terraform apply is failing with the following error. What next step should you take to determine the root cause of the problem?
Error:
yaml
CopyEdit
Error loading state: AccessDenied: Access Denied
status code: 403, request id: 288766CE5CCA24A0, host id: web.example.com
A. Run terraform login to reauthenticate with the provider.
B. Set TF_LOG=DEBUG.

C. Review /var/log/terraform.log for error messages.
D. Review syslog for Terraform error messages.
Answer: B (LEAVE A REPLY)
The error messageindicates an authentication issue (403 - Access Denied), which requiresdebugging Terraform logs.
Setting TF_LOG=DEBUG enablesdetailed logs, providing insights into API requests, state loading, and authentication errors.
Terraformdoes not log errors in /var/log/terraform.log or syslog, making optionsC and D incorrect.
A (terraform login)is only relevant forTerraform Cloud, but this error appears to be related toprovider authentication.
Official Terraform Documentation Reference:
Debugging Terraform

NEW QUESTION: 107
A terraform apply can not _____ infrastructure.
A. change
B. destroy
C. provision
D. import
Answer: D (LEAVE A REPLY)
The terraform import command is used to import existing infrastructure into Terraform's state. This allows Terraform to manage and destroy the imported infrastructure as part of the configuration. The terraform import command does not modify the configuration, so the imported resources must be manually added to the configuration after the import. References = [Importing Infrastructure]

NEW QUESTION: 108
A child module can always access variables declared in its parent module.
A. True
B. False
Answer: (SHOW ANSWER)

Child modulesdo not automatically inheritvariables from the parent module.
To pass values from theparent moduleto thechild module, you mustexplicitly define input variablesin the child module and pass them in the parent module.
Example:
hcl
CopyEdit
module "example" {
source = "./child_module"
var1 = "value"
}
Official Terraform Documentation Reference:
Passing Variables to Modules

NEW QUESTION: 109
You have just developed a new Terraform configuration for two virtual machines with a cloud provider. You would like to create the infrastructure for the first time.
Which Terraform command should you runfirst?
A. terraform apply
B. terraform init
C. terraform plan
D. terraform show
Answer: B (LEAVE A REPLY)
B (terraform init)-Must be run firstto initialize the Terraform working directory, download providers, and configure the backend.
A (terraform apply)- Requires initialization first, so itcannotbe run before terraform init.
C (terraform plan)- Also requires terraform init first to generate a plan.
D (terraform show)- Displays the state,not relevantfor first-time deployment.
Official Terraform Documentation Reference:
terraform init - HashiCorp Documentation

NEW QUESTION: 110
One remote backend configuration always maps to a single remote workspace.
A. True
B. False
Answer: A (LEAVE A REPLY)
The remote backend can work with either a single remote Terraform Cloud workspace, or with multiple similarly-named remote workspaces (like networking-dev and networking-prod). The workspaces block of the backend configuration determines which mode it uses. To use a single remote Terraform Cloud workspace, set workspaces.name to the remote workspace's full name (like networking-prod). To use multiple remote workspaces, set workspaces.prefix to a prefix used in all of the desired remote workspace names. For example, set prefix = "networking-" to use

Terraform cloud workspaces with names like networking-dev and networking-prod. This is helpful when mapping multiple Terraform CLI workspaces used in a single Terraform configuration to multiple Terraform Cloud workspaces3. However, one remote backend configuration always maps to a single remote workspace, either by name or by prefix. You cannot use both name and prefix in the same backend configuration, or omit both. Doing so will result in a configuration error3. References = [Backend Type: remote]3

NEW QUESTION: 111
What feature stops multiple users from operating on the Terraform state at the same time?
A. State locking
B. Version control
C. Provider constraints
D. Remote backends
Answer: A (LEAVE A REPLY)
State locking prevents other users from modifying the state file while a Terraform operation is in progress. This prevents conflicts and data loss1.

NEW QUESTION: 112
In Terraform HCL, an object type of object({name=string, age-number}) would match this value.

A.
```
{
  name = "John"
  age = fifty two
}
```

B.
```
{
  name = "John"
  age = 52
}
```

C.
```
{
  name =  John
  age = fifty two
}
```

D.
```
{
  name =  John
  age = 52
}
```


A. Option A

B. Option B

C. Option C

D. Option D

Answer: B (LEAVE A REPLY)

From the official Terraform Type Constraints Documentation:

The object({ name = string, age = number }) type expects:
name to be aquoted string, e.g. "John"
age to be anumber, e.g. 52
Option Bsatisfies both:
{
name = "John" ##Valid string
age = 52 ##Valid number
}
Let's analyze the incorrect ones:
#Option A: "fifty two" is not a valid number.
#Option C: John is unquoted (invalid string), and "fifty two" is not a number.
#Option D: name = John is not quoted, making it an invalid string in HCL.
Terraform is strict about type and formatting. Stringsmust be quoted, and numbersmust be numeric literals.

NEW QUESTION: 113
terraform validate uses provider APIs to verify your infrastructure settings.
A. True
B. False
Answer: (SHOW ANSWER)
terraform validateonlychecks the configuration's syntax and internal consistency-itdoes notinteract with provider APIs or check if the infrastructure settings are correct.
It ensures that the Terraform code is syntactically correct and follows proper HCL structure.
However, it doesnotverify if resources are valid according to the provider API or if the credentials are correct.
To verify actual infrastructure settings, use terraform plan, which interacts with the provider APIs.
Official Terraform Documentation Reference:
terraform validate - HashiCorp Documentation

NEW QUESTION: 114
Exhibit.

Exhibit

You need to deploy resources into two different regions in the same Terraform configuration. To do this, you declare multiple provider configurations as shown in the Exhibit space on this page. What meta-argument do you need to configure in a resource block to deploy the resource to the us-west-2 AWS region?

A. provider = aws.west

B. alias = aws.west

C. provider = west

D. alias = west

Answer: (SHOW ANSWER)

* Rationale for Correct answer:When multiple provider configurations are defined using an alias, Terraform requires the provider meta-argument inside the resource block to explicitly reference the aliased provider.In this case, the provider is defined as aws with alias = "west", so resources targeting us-west-2 must specify:

provider = aws.west

This tells Terraform exactly which provider configuration to use.

* Analysis of Incorrect Options (Distractors):

* B. alias = aws.west: Incorrect - alias is only used inside the provider block, not in resources.

* C. provider = west: Incorrect - Terraform requires the full provider name (aws.west), not just the alias.

* D. alias = west: Incorrect - resources do not support an alias meta-argument.

* Key Concept:Using aliased provider configurations and the provider meta-argument to deploy resources across multiple regions or accounts.

Reference:Terraform Exam Objective - Manage Terraform Resources and Providers

NEW QUESTION: 115

Which of the following is not an action performed by terraform init?

A. Initialize a configured backend.

B. Create template configuration files.

C. Load required provider plugins.
D. Retrieve the source code for all referenced modules.
Answer: (SHOW ANSWER)
* Rationale for Correct answer:terraform init initializes backends, downloads providers, and retrieves modules - but it does not generate or create configuration files.
* Analysis of Incorrect Options (Distractors):
* A: Correct action performed by terraform init.
* C: Providers are downloaded during initialization.
* D: Module source code is fetched during initialization.
* Key Concept:terraform init prepares the working directory but does not create configuration.
Reference:Terraform Exam Objective - Understand Terraform Basics and CLI

NEW QUESTION: 116
Which command should you run to check if all code in a Terraform configuration that references multiple modules is properly formatted without making changes?
A. terraform fmt -write-false
B. terraform fmt -list -recursive
C. terraform fmt -check -recursive
D. terraform fmt -check
Answer: (SHOW ANSWER)
This command will check if all code in a Terraform configuration that references multiple modules is properly formatted without making changes, and will return a non-zero exit code if any files need formatting. The other commands will either make changes, list the files that need formatting, or not check the modules.

NEW QUESTION: 117
The Terraform binary version and provider versions must match each other in a single configuration.
A. True
B. False
Answer: (SHOW ANSWER)
The Terraform binary version and provider versions do not have to match each other in a single configuration.
Terraform allows you to specify provider version constraints in the configuration's terraform block, which can be different from the Terraform binary version1. Terraform will use the newest version of the provider that meets the configuration's version constraints2. You can also use the dependency lock file to ensure Terraform is using the correct provider version3. References =
*1: Providers - Configuration Language | Terraform | HashiCorp Developer
*2: Multiple provider versions with Terraform - Stack Overflow
*3: Lock and upgrade provider versions | Terraform - HashiCorp Developer

NEW QUESTION: 118
You're building a CI/CD (continuous integration/continuous delivery) pipeline and need to inject sensitive variables into your Terraform run. How can you do this safely?
A. Copy the sensitive variables into your Terraform code
B. Store the sensitive variables in a secure_varS.tf file
C. Store the sensitive variables as plain text in a source code repository
D. Pass variables to Terraform with a -var flag
Answer: D (LEAVE A REPLY)
This is a secure way to inject sensitive variables into your Terraform run, as they will not be stored in any file or source code repository. You can also use environment variables or variable files with encryption to pass sensitive variables to Terraform.

NEW QUESTION: 119
Which of the following locations can Terraform use as aprivate sourcefor modules?(Pick 2 correct responses)
A. Public repository on GitHub.
B. Public Terraform Registry.
C. Internally hosted VCS (Version Control System) platform.
D. Private repository on GitHub.
Answer: C,D (LEAVE A REPLY)
Terraform allows usingprivate module sourceshosted in:
C (Internally hosted VCS platforms, e.g., GitLab, Bitbucket, GitHub Enterprise)# D (Private GitHub repositories)# A (Public GitHub repository)#-GitHubis supported, but apublic repo is not "private".
B (Public Terraform Registry)#-The public registryis not a private source.
Official Terraform Documentation Reference:
Terraform Module Sources

NEW QUESTION: 120
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
Answer: C (LEAVE A REPLY)
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 downloadit 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]

NEW QUESTION: 121
Using the terraform state rm command against a resource will destroy it.
A. True
B. False
Answer: B (LEAVE A REPLY)
The terraform state rm commandremovesa resource from Terraform's state file butdoes not destroythe resource in the actual infrastructure. It only removes Terraform's knowledge of the resource, meaning Terraform will no longer manage it.
If you run terraform state rm on a resource, Terraform will forget that the resource exists. However, the resource will still exist in the cloud or infrastructure provider.
If you later run terraform apply, Terraform may try torecreatethe resource because it is no longer present in its state file.
Official Terraform Documentation Reference:
terraform state rm - HashiCorp Documentation

NEW QUESTION: 122
If a DevOps team adopts AWS CloudFormation as their standardized method for provisioning public cloud resoruces, which of the following scenarios poses a challenge for this team?
A. The team is asked to manage a new application stack built on AWS-native services
B. The organization decides to expand into Azure wishes to deploy new infrastructure
C. The team is asked to build a reusable code based that can deploy resources into any AWS region
D. The DevOps team is tasked with automating a manual, web console-based provisioning.
Answer: B (LEAVE A REPLY)
This is the scenario that poses a challenge for this team, if they adopt AWS CloudFormation as their standardized method for provisioning public cloud resources, as CloudFormation only

supports AWS services and resources, and cannot be used to provision infrastructure on other cloud platforms such as Azure.

NEW QUESTION: 123
Which of the following arguments are required when declaring a Terraform output?
A. value
B. description
C. default
D. sensitive
Answer: A (LEAVE A REPLY)
When declaring a Terraform output, the value argument is required. Outputs are a way to extract information from Terraform-managed infrastructure, and the value argument specifies what data will be outputted. While other arguments like description and sensitive can provide additional context or security around the output, value is the only mandatory argument needed to define an output.References = The requirement of the value argument for outputs is specified in Terraform's official documentation, which provides guidelines on defining and using outputs in Terraform configurations.

NEW QUESTION: 124
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
Answer: D (LEAVE A REPLY)
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

NEW QUESTION: 125
You should run terraform fnt to rewrite all Terraform configurations within the current working directory to conform to Terraform-style conventions.
A. True

B. False

Answer: A (LEAVE A REPLY)

You should run terraform fmt to rewrite all Terraform configurations within the current working directory to conform to Terraform-style conventions. This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability. It is recommended to use this command to ensure consistency of style across different Terraform codebases. The command is optional, opinionated, and has no customization options, but it can help you and your team understand the code more quickly and easily. References = : Command: fmt : Using Terraform fmt Command to Format Your Terraform Code

NEW QUESTION: 126

Which backend does the Terraform CLI use by default?

A. Depends on the cloud provider configured

B. HTTP

C. Remote

D. Terraform Cloud

E. Local

Answer: (SHOW ANSWER)

This is the backend that the Terraform CLI uses by default, unless you specify a different backend in your configuration. The local backend stores the state file in a local file named terraform.tfstate, which can be used to track and manage the state of your infrastructure.

NEW QUESTION: 127

Which command must you first run before performing further Terraform operations in a working directory?

A. terraform import

B. terraform workspace

C. terraform plan

D. terraform init

Answer: (SHOW ANSWER)

terraform init is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It initializes a working directory containing Terraform configuration files and downloads any required providers and modules. The other commands are used for different purposes, such as importing existing resources, switching between workspaces, generating execution plans, etc.

NEW QUESTION: 128

You want to define multiple data disks as nested blocks inside the resource block for a virtual machine. What Terraform feature would help you define the blocks using the values in a variable?

A. Local values

B. Count arguments

C. Collection functions
D. Dynamic blocks
Answer: D (LEAVE A REPLY)
Dynamic blocks in Terraform allow you to define multiple nested blocks within a resource based on the values of a variable. This feature is particularly useful for scenarios where the number of nested blocks is not fixed and can change based on variable input.