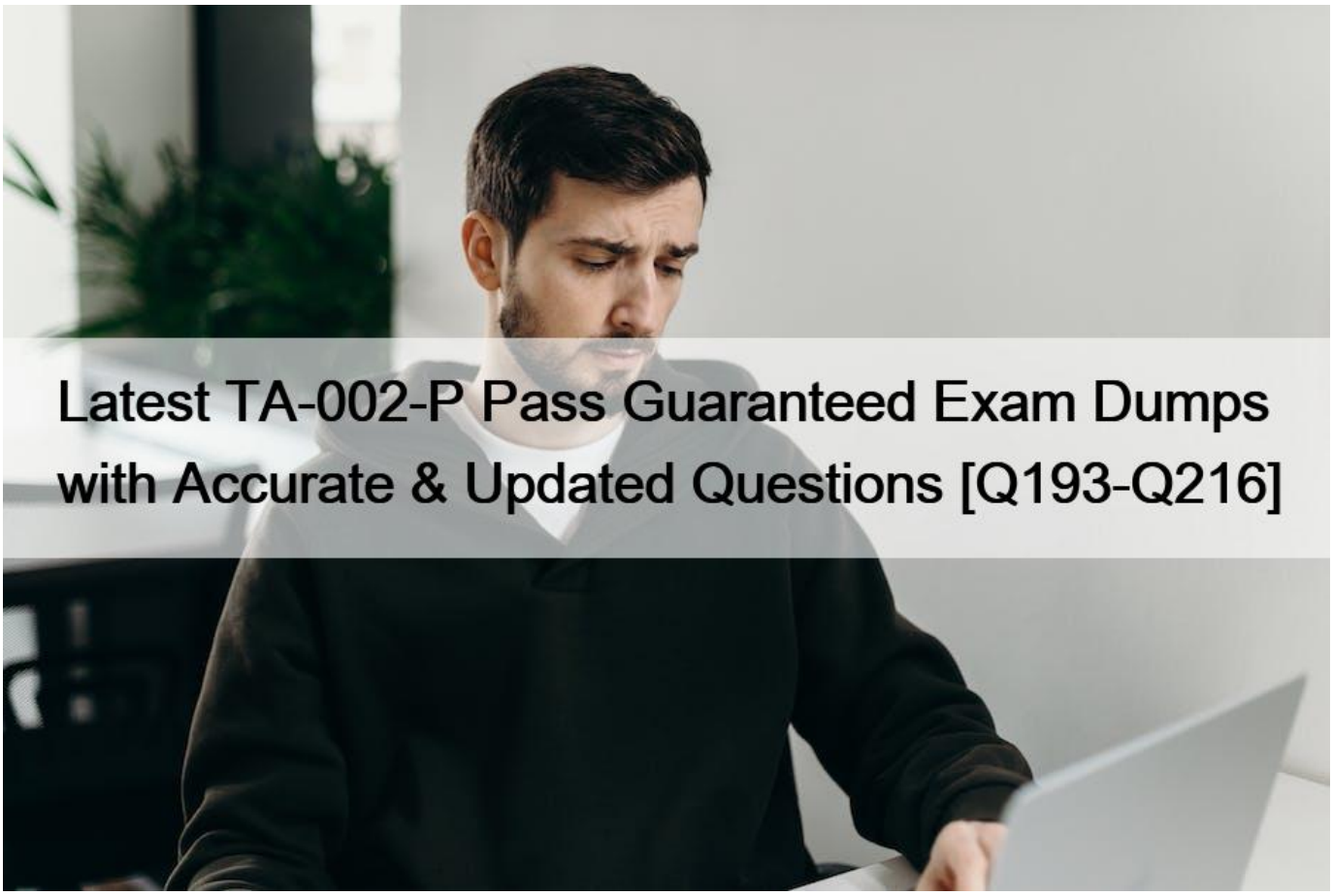
Latest TA-002-P Pass Guaranteed Exam Dumps with Accurate & Updated Questions [Q193-Q216



Latest TA-002-P Pass Guaranteed Exam Dumps with Accurate & Updated Questions
TA-002-P Exam Brain Dumps - Study Notes and Theory

Understanding functional and technical aspects of HashiCorp Certified: Terraform Associate TA-002-P Professional Exam Object Management

The following will be discussed in **HASHICORP TA-002 exam dumps**:

- Utilize resource addressing and resource parameters to connect resources- Exhibit use of variables and outputs- Design and distinguish resource and data configuration- Advantages of Sentinel, registry, and workspaces Differentiate OSS and TFE workspaces- Learn the application of the collection and structural types

Sample Questions

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- It is automatically deleted- Terraform attempts to provision the resource up to three times before exiting with an error- The resource is marked as tainted- The terraform plan is rolled back and all provisioned resources are removed

Explanation

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as ?tainted'. A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not

automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.
True or False? When using the Terraform provider for Vault, the tight integration between these HashiCorp tools provides the ability to mask secrets in the terraform plan and state files.

- True- False

Explanation
Currently, Terraform has no mechanism to redact or protect secrets that are returned via data sources, so secrets read via this provider will be persisted into the Terraform state, into any plan files, and in some cases in the console output produced while planning and applying. These artifacts must, therefore, all be protected accordingly.

NEW QUESTION 193

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?
* Local values
* Dynamic blocks
* Count arguments
* Collection functions

NEW QUESTION 194

Which of the following command can be used to view the specified version constraints for all providers used

in the current configuration.
* terraform providers
* terraform state show
* terraform provider
* terraform plan

Explanation

Use the terraform providers command to view the specified version constraints for all providers used in the

current configuration.

https://www.terraform.io/docs/configuration/providers.html

NEW QUESTION 195

A user creates three workspaces from the command line – prod, dev, and test. Which of the following commands will the user run to switch to the dev workspace?
* terraform workspace dev
* terraform workspace select dev
* terraform workspace -switch dev
* terraform workspace switch dev

Explanation

The terraform workspace select command is used to choose a different workspace to use for further operations.
https://www.terraform.io/docs/commands/workspace/select.html

NEW QUESTION 196

A Terraform provisioner must be nested inside a resource configuration block.
* True
* False
Most provisioners require access to the remote resource via SSH or WinRM, and expect a nested connection block with details about how to connect.

NEW QUESTION 197

Which of the below are paid features of Terraform Cloud?
* Full API Coverage
* Secure variable Storage
* Roles/ Team management
* Cost Estimation
* Private Module Registry
* Sentinel policies
Explanation

https://www.hashicorp.com/products/terraform/pricing/

NEW QUESTION 198

Terraform variables and outputs that set the “description” argument will store that description in the state file.
* True
* False
Explanation/Reference: https://www.terraform.io/docs/language/values/outputs.html

NEW QUESTION 199

Eric needs to make use of module within his terraform code. Should the module always be public and open-source to be able to be used?
* False
* True
Terraform module need not be public and open-source. Module can be placed in –

* Local paths

* Terraform Registry

* GitHub

* Bitbucket

* Generic Git, Mercurial repositories

* HTTP URLs

* S3 buckets

* GCS buckets

https://www.terraform.io/docs/modules/sources.html

NEW QUESTION 200

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?
* terraform apply -var-state-file =”testing.tfvars”
* terraform plan -var-file=”testing.tfvar”
* terraform apply -var-file=”testing.tfvars”
* terraform apply var-file=”testing.tfvars”

https://www.terraform.io/docs/configuration/variables.html

NEW QUESTION 201

What is terraform refresh intended to detect?
* Terraform configuration code changes
* Empty state files
* State file drift
* Corrupt state files

Explanation

“The terraform refresh command reads the current settings from all managed remote objects and updates the

Terraform state to match. Warning: This command is deprecated, because its default behavior is unsafe if you

have misconfigured credentials for any of your providers. See below for more information and recommended

alternatives.” https://www.terraform.io/cli/commands/refresh

NEW QUESTION 202

Which of the below options is the equivalent Terraform 0.12 version of the snippet which is written in Terraform 0.11?

“${var.instance_id}”
* variable.instance_id
* var.instance_ids
* var.instance_id
* None of the above

NEW QUESTION 203

You need to specify a dependency manually. What resource meta-parameter can you use lo make sure Terraform respects thee dependency?

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

NEW QUESTION 204

Using multi-cloud and provider-agnostic tools provides which of the following benefits?
* Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
* Increased risk due to all infrastructure relying on a single tool for management.
* Can be used across major cloud providers and VM hypervisors.
* Slower provisioning speed allows the operations team to catch mistakes before they are applied.
Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

NEW QUESTION 205

While attempting to deploy resources into your cloud provider using Terraform. you begin to see some odd

behavior and experience sluggish responses. In order to troubleshoot you decide to turn on Terraform

debugging. Which environment variables must be configured to make Terraform’s logging more verbose?
* TF_10G_PATM
* TF_LOG
* TF_10G_LEVEL
* TF.LOG.FUE
Explanation

https://www.terraform.io/internals/debugging

NEW QUESTION 206

Which of the following is not a valid Terraform string function?
* replace
* format
* join
* tostring
https://www.terraform.io/docs/configuration/functions/tostring.html

NEW QUESTION 207

True or False: A list(…) contain a number of values of the same type while an object(…) can contain a number of values of different types.
* False
* True
Collection Types

A collection type allows multiple values of one other type to be grouped together as a single value. The type of value within a collection is called its element type. All collection types must have an element type, which is provided as the argument to their constructor.

For example, the type list(string) means “list of strings”, which is a different type than list(number), a list of numbers. All elements of a collection must always be of the same type.

The three kinds of collection type in the Terraform language are:

* list(…): a sequence of values identified by consecutive whole numbers starting with zero.

The keyword list is a shorthand for list(any), which accepts any element type as long as every element is the same type. This is for compatibility with older configurations; for new code, we recommend using the full form.

* map(…): a collection of values where each is identified by a string label.

The keyword map is a shorthand for map(any), which accepts any element type as long as every element is the same type. This is for compatibility with older configurations; for new code, we recommend using the full form.

* set(…): a collection of unique values that do not have any secondary identifiers or ordering.

https://www.terraform.io/docs/configuration/types.html

Structural Types

A structural type allows multiple values of several distinct types to be grouped together as a single value. Structural types require a schema as an argument, to specify which types are allowed for which elements.

The two kinds of structural type in the Terraform language are:

* object(…): a collection of named attributes that each have their own type.

The schema for object types is { <KEY> = <TYPE>, <KEY> = <TYPE>, … } – a pair of curly braces containing a comma-separated series of <KEY> = <TYPE> pairs. Values that match the object type must contain all of the specified keys, and the value for each key must match its specified type. (Values with additional keys can still match an object type, but the extra attributes are discarded during type conversion.)

* tuple(…): a sequence of elements identified by consecutive whole numbers starting with zero, where each element has its own type.

The schema for tuple types is [<TYPE>, <TYPE>, …] – a pair of square brackets containing a comma-separated series of types. Values that match the tuple type must have exactly the same number of elements (no more and no fewer), and the value in each position must match the specified type for that position.

For example: an object type of object({ name=string, age=number }) would match a value like the following:

{

name = “John”

age = 52

}

Also, an object type of object({ id=string, cidr_block=string }) would match the object produced by a reference to an aws_vpc resource, like aws_vpc.example_vpc; although the resource has additional attributes, they would be discarded during type conversion.

Finally, a tuple type of tuple([string, number, bool]) would match a value like the following:

[“a”, 15, true]

https://www.terraform.io/docs/configuration/types.html

NEW QUESTION 208

What kind of configuration block will create an infrastructure object with settings specified in the block?
* state
* provider
* resource
* data

NEW QUESTION 209

What is the standard workflow that a developer follows while working with terraform open source version?
* Run terraform refresh to update the terraform state , then write the terraform code , and finally run

terraform apply.
* Run terraform destroy first since you need to start from fresh every time , before running terraform

apply.
* Write terraform code , and run terraform push , to update the terraform state to the remote repo , which

in turn will take care of the next steps.
* Write the terraform code on the developer machine , run terraform plan to check the changes , and run

terraform apply to provision the infra.
Explanation

You do not need to run terraform refresh as terraform plan implicitly will run terraform refresh.

https://www.terraform.io/guides/core-workflow.html

NEW QUESTION 210

When using providers that require the retrieval of data, such as the HashiCorp Vault provider, in what phase does Terraform actually retrieve the data required?
* terraform delete
* terraform plan
* terraform init
* terraform apply

NEW QUESTION 211

You have used Terraform to create an ephemeral development environment in the cloud and are now ready to destroy all the infrastructure described by your Terraform configuration. To be safe, you would like to first see all the infrastructure that will be deleted by Terraform.

Which command should you use to show all of the resources that will be deleted? (Choose two.)
* Run terraform plan -destroy.
* This is not possible. You can only show resources that will be created.
* Run terraform state rm *.
* Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval.

NEW QUESTION 212

When using providers that require the retrieval of data, such as the HashiCorp Vault provider, in what phase

does Terraform actually retrieve the data required?
* terraform delete
* terraform plan
* terraform init
* terraform apply

NEW QUESTION 213

When do you need to explicitly execute terraform refresh?
* Before every terraform plan
* Before every terraform apply
* Before every terraform import
* None of the above

Explanation

Wherever possible, avoid using terraform refresh explicitly and instead rely on Terraform’s behavior of

automatically refreshing existing objects as part of creating a normal plan. Source:

https://www.terraform.io/cli/commands/refresh

NEW QUESTION 214

Which of the following is not a valid string function in Terraform?
* split
* join
* slice
* chomp

NEW QUESTION 215

Which of the following is the right substitute for static values that can make Terraform configuration file more dynamic and reusable?
* Output value
* Input parameters
* Functions
* Modules

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module’s own source code, and allowing modules to be shared between different configurations.

NEW QUESTION 216

terraform init retrieves the source code tot all referenced modules
* True
* False
Explanation

Terraform installs providers, initialises source code & modules etc at this stage

Understanding functional and technical aspects of HashiCorp Certified: Terraform Associate TA-002-P Professional Exam Object Management

The following will be discussed in **HASHICORP TA-002 exam dumps**:

- Perform modifications to infrastructure with Terraform- Describe default local backend- Express backend block in arrangement and best practices for partial arrangements- Contrast module source alternatives- Destroy Terraform controlled infrastructure- Define variable scope inside modules- Create and review a performance plan for Terraform- Determining module version- Learn concealed management in state files- Specify remote state storage mechanisms and supported regular backends. **Pass HashiCorp TA-002-P Test Practice Test Questions Exam Dumps:**

https://www.examcollectionpass.com/HashiCorp/TA-002-P-practice-exam-dumps.html]