



CompTIA

CertyIQ

Premium exam material

Get certification quickly with the CertyIQ Premium exam material.

Everything you need to prepare, learn & pass your certification exam easily. Lifetime free updates

First attempt guaranteed success.

<https://www.CertyIQ.com>

About CertyIQ

We here at CertyIQ eventually got enough of the industry's greedy exam paid for. Our team of IT professionals comes with years of experience in the IT industry Prior to training CertyIQ we worked in test areas where we observed the horrors of the paywall exam preparation system.

The misuse of the preparation system has left our team disillusioned. And for that reason, we decided it was time to make a difference. We had to make In this way, CertyIQ was created to provide quality materials without stealing from everyday people who are trying to make a living.

Doubt Support

We have developed a very scalable solution using which we are able to solve 400+ doubts every single day with an average rating of 4.8 out of 5.

<https://www.certyiq.com>

[Mail us on - certyiqofficial@gmail.com](mailto:certyiqofficial@gmail.com)



Lifetime Free Updates

We provide lifetime free updates to our customers. To make life easier for our valued customers and fulfill their needs



Free Exam PDF

You are sure to pass the exam completely free of charge



Money Back Guarantee

We Provide 100% money back guarantee to our customer in case of any failure

John

October 19, 2022



Thanks you so much for your help. I scored 972 in my exam today. More than 90% were from your PDFs!

Dana

September 04, 2022



Thanks a lot for this updated AZ-900 Q&A. I just passed my exam and got 974, I followed both of your Az-900 videos and the 6 PDF, the PDFs are very much valid, all answers are correct. Could you please create a similar video/PDF for DP900, your content/PDF's is really awesome. The team did a really good job. Thank You 😊.

Ahamed Shibly

2 months ago



Customer support is really fast and helpful, I just finished my exam and this video along with the 6 PDF helped me pass! Definitely recommend getting the PDFs. Thank you!

October 22, 2022



Passed my exam today with 891 marks. Out of 52 questions, 51 were from certyiq PDFs including Contoso case study. Thank You certyiq team!

Henry Rome

2 months ago



These questions are real and 100 % valid. Thank you so much for your efforts, also your 4 PDFs are awesome, I passed the DP900 exam on 1 Sept. With 968 marks. Thanks a lot, buddy!

Esmaria

2 months ago



Simple easy to understand explanations. To anyone out there wanting to write AZ900, I highly recommend 6 PDF's. Thank you so much, appreciate all your hard work in having such great content. Passed my exam Today - 3 September with 942 score.

Hashicorp

(Terraform Associate)

HashiCorp Certified

Total: **283 Questions**

Link: <https://certyiq.com/papers?provider=hashicorp&exam=terraform-associate>

Question: 1

The terraform.tfstate file always matches your currently built infrastructure.

- A. True
- B. False

Answer: B

Explanation:

Reference:

<https://www.terraform.io/docs/language/state/index.html>

" target="_blank" style="word-break: break-all;">

This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment.

Terraform uses this local state to create plans and make changes to your infrastructure. Prior to any operation, Terraform does a **refresh** to update the state with the real infrastructure.

The primary purpose of Terraform state is to store bindings between objects in a remote system and resource instances declared in your configuration. When Terraform creates a remote object in response to a change of configuration, it will record the identity of that remote object against a particular resource instance, and then potentially update or delete that object in response to future configuration changes.

Question: 2

One remote backend configuration always maps to a single remote workspace.

- A. True
- B. False

Answer: B

Explanation:

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 workspaces.

<https://developer.hashicorp.com/terraform/language/settings/backends/remote>

Question: 3

CertyIQ

How is the Terraform remote backend different than other state backends such as S3, Consul, etc.?

- A. It can execute Terraform runs on dedicated infrastructure on premises or in Terraform Cloud
- B. It doesn't show the output of a terraform apply locally
- C. It is only available to paying customers
- D. All of the above

Answer: A

Explanation:

If you and your team are using Terraform to manage meaningful infrastructure, we recommend using the remote backend with Terraform Cloud or Terraform Enterprise.

Reference:

<https://www.terraform.io/docs/language/settings/backends/index.html>

Question: 4

CertyIQ

What is the workflow for deploying new infrastructure with Terraform?

- A. terraform plan to import the current infrastructure to the state file, make code changes, and terraform apply to update the infrastructure.
- B. Write a Terraform configuration, run terraform show to view proposed changes, and terraform apply to create new infrastructure.
- C. terraform import to import the current infrastructure to the state file, make code changes, and terraform apply to update the infrastructure.
- D. Write a Terraform configuration, run terraform init, run terraform plan to view planned infrastructure changes, and terraform apply to create new infrastructure.

Answer: D

Explanation:

Correct answer is D: Write a Terraform configuration, run terraform init, run terraform plan to view planned infrastructure changes, and terraform apply to create new infrastructure.

Question: 5

CertyIQ

A provider configuration block is required in every Terraform configuration.
Example:

```
provider "provider_name" {  
    ...  
}
```

- A. True

B. False

Answer: A

Explanation:

Reference:

<https://github.com/hashicorp/terraform/issues/17928>

Question: 6

CertyIQ

You run a local-exec provisioner in a null resource called null_resource.run_script and realize that you need to rerun the script.

Which of the following commands would you use first?

- A. terraform taint null_resource.run_script
- B. terraform apply -target=null_resource.run_script
- C. terraform validate null_resource.run_script
- D. terraform plan -target=null_resource.run_script

Answer: A

Explanation:

You are all correct that taint has been deprecated and replaced with -replace. But neither D nor any other option here uses the -replace command. Therefore option A is the only valid option given these choices.

Question: 7

CertyIQ

Which provisioner invokes a process on the resource created by Terraform?

- A. remote-exec
- B. null-exec
- C. local-exec
- D. file

Answer: A

Explanation:

The remote-exec provisioner invokes a script on a remote resource after it is created.

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/remote-exec.html>

Question: 8

CertyIQ

Which of the following is not true of Terraform providers?

- A. Providers can be written by individuals
- B. Providers can be maintained by a community of users
- C. Some providers are maintained by HashiCorp

- D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers
- E. None of the above

Answer: E

Explanation:

E. None of the above

All of the statements are true of Terraform providers.

A. Providers can be written by individuals - Any person or organization can develop and distribute a Terraform provider, allowing them to expand Terra form's capabilities to manage resources that it previously could not.

B. Providers can be maintained by a community of users - Many Terraform providers are open source projects, and the development and maintenance of these providers can be collaborative efforts between multiple individuals and organizations.

C. Some providers are maintained by Hashi Corp - Hashi Corp, the creators of Terraform, maintain a number of official providers that cover popular infrastructure providers such as AWS, Google Cloud, and Microsoft Azure.

D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers - Providers can be developed and maintained by cloud vendors, non-cloud vendors, or a combination of both, to expand Terraform 's capabilities and support for different types of infrastructure.

Question: 9

CertyIQ

What command does Terraform require the first time you run it within a configuration directory?

- A. terraform import
- B. terraform init
- C. terraform plan
- D. terraform workspace

Answer: B

Explanation:

terraform init command is used to initialize a working directory containing Terraform configuration files.

Reference:

<https://www.terraform.io/docs/cli/commands/init.html>

Question: 10

CertyIQ

You have deployed a new webapp with a public IP address on a cloud provider. However, you did not create any outputs for your code.

What is the best method to quickly find the IP address of the resource you deployed?

- A. Run terraform output ip_address to view the result
- B. In a new folder, use the terraform_remote_state data source to load in the state file, then write an output for each resource that you find the state file
- C. Run terraform state list to find the name of the resource, then terraform state show to find the attributes

including public IP address

D. Run terraform destroy then terraform apply and look for the IP address in stdout

Answer: C

Explanation:

You can find the info in the state. not A because you don't have to outputs defined.

Question: 11

CertyIQ

Which of the following is not a key principle of infrastructure as code?

- A. Versioned infrastructure
- B. Golden images
- C. Idempotence
- D. Self-describing infrastructure

Answer: B

Explanation:

1. it not possible create golden image from terraform, thats, is an other tools from hashicorp
2. B. Golden images"Golden images" refers to a specific method of deployment where a pre-configured image of an operating system or application is stored and used as a base for all new deployments. This method is not a key principle of Infrastructure as Code, but it is a common method of deployment in traditional IT environments. Infrastructure as Code is based on the following key principles: A. Versioned infrastructure - The infrastructure is treated as code and is version controlled, allowing for auditing, rollback, and collaboration. C. Idempotence - The infrastructure is provisioned in a repeatable and predictable manner, making it possible to run the provisioning scripts multiple times without creating additional resources or causing changes to existing resources. D. Self-describing infrastructure - The code used to provision the infrastructure is human-readable and self-documented, making it easier to understand and maintain over time.

Question: 12

CertyIQ

Terraform variables and outputs that set the "description" argument will store that description in the state file.

- A. True
- B. False

Answer: B

Explanation:

- 1. Answer is B. Descriptions aren't stored in the state file.
- 2. The answer is B. Descriptions aren't stored in the state file.

Question: 13

CertyIQ

What is the provider for this fictitious resource?


```
resource "aws_vpc" "main" {  
    name = "test"  
}
```

- A. vpc
- B. main
- C. aws
- D. test

Answer: C

Explanation:

Reference:

<https://docs.aws.amazon.com/cloudformation-cli/latest/userguide/resource-types.html>

Question: 14

CertyIQ

If you manually destroy infrastructure, what is the best practice reflecting this change in Terraform?

- A. Run terraform refresh
- B. It will happen automatically
- C. Manually update the state file
- D. Run terraform import

Answer: A

Explanation:

A. Run terraform refresh

Yes, in older versions of Terraform (before 0.15), terraform refresh could be used to reconcile the state file with the actual infrastructure. From Terraform 0.15 and later, terraform apply performs this reconciliation.

B. It will happen automatically

No, Terraform does not automatically detect changes made outside of its context. You must run terraform apply (or terraform plan in older versions) to see these changes and apply them to the Terraform state file.

Question: 15

CertyIQ

What is not processed when running a terraform refresh?

- A. State file
- B. Configuration file
- C. Credentials
- D. Cloud provider

Answer: B

Explanation:

1. A. State file - its updated during the refresh B. Configuration file - C. Credentials - required to get onto the cloud D. Cloud provider - required to carry out the refresh of whats in the cloud vs whats in state.
2. The answer is B.

Question: 16**CertyIQ**

What information does the public Terraform Module Registry automatically expose about published modules?

- A. Required input variables
- B. Optional inputs variables and default values
- C. Outputs
- D. All of the above
- E. None of the above

Answer: D**Explanation:**

According to the documentation provided, D is correct. Extract from documentation: "The registry extracts information about the module from the module's source. The module name, provider, documentation, inputs/outputs, and dependencies are all parsed and available via the UI or API, as well as the same information for any submodules or examples in the module's source repository."

definitely D since it's 'public' (all input & output variables should be revealed)

Question: 17**CertyIQ**

If a module uses a local values, you can expose that value with a terraform output.

- A. True
- B. False

Answer: A**Explanation:**

Output values are like function return values.

Reference:

<https://www.terraform.io/docs/language/values/locals.html>

<https://www.terraform.io/docs/language/values/outputs.html>

Question: 18**CertyIQ**

You should store secret data in the same version control repository as your Terraform configuration.

- A. True
- B. False

Answer: B

Explanation:

It is generally considered insecure to store secret data, such as passwords, API keys, and other sensitive information, in the same version control repository as your Terraform configuration. This is because version control repositories are often publicly accessible, and if sensitive information is stored in the repository it can be easily accessed by unauthorized individuals. Additionally, version control repositories typically have a history of all changes made to files, so even if sensitive information is deleted at a later point, it can still be retrieved from the repository history. To properly secure secret data, it is recommended to store it in a secure and encrypted format, such as in a secure vault or by using a tool specifically designed for storing secrets.

Reference:

<https://blog.gruntwork.io/a-comprehensive-guide-to-managing-secrets-in-your-terraform-code-1d586955ace1>

Question: 19**CertyIQ**

Which of the following is not a valid string function in Terraform?

- A. split
- B. join
- C. slice
- D. chomp

Answer: C**Explanation:**

slice is not in string function list.

C - Slice that s used for list, not string, chomp is for used in string

Question: 20**CertyIQ**

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

- A. Provision new VMs using Terraform with the same VM names
- B. Use the terraform import command for the existing VMs
- C. Write Terraform configuration for the existing VMs
- D. Run the terraform import-gcp command

Answer: BC**Explanation:**

1. You should create the equivalent configuration first, and then run import to load it on the state file.
2. To manage the VMs that were provisioned using the gcloud command line tool using Terraform, you must first use the "terraform import" command to import the existing VMs into Terraform's state file. This allows Terraform to recognize the VMs and manage them as part of your infrastructure. After importing the VMs, you must then write Terraform configuration for the existing VMs. This includes defining the resources for the

VMs and specifying the necessary configuration options, such as the instance type and image, as well as any other desired settings. By doing this, you can manage the VMs using Terraform, which provides a single, unified tool for provisioning, configuring, and managing your infrastructure.

Question: 21

CertyIQ

You have recently started a new job at a retailer as an engineer. As part of this new role, you have been tasked with evaluating multiple outages that occurred during peak shopping time during the holiday season. Your investigation found that the team is manually deploying new compute instances and configuring each compute instance manually. This has led to inconsistent configuration between each compute instance. How would you solve this using infrastructure as code?

- A. Implement a ticketing workflow that makes engineers submit a ticket before manually provisioning and configuring a resource
- B. Implement a checklist that engineers can follow when configuring compute instances
- C. Replace the compute instance type with a larger version to reduce the number of required deployments
- D. Implement a provisioning pipeline that deploys infrastructure configurations committed to your version control system following code reviews

Answer: D

Explanation:

Using infrastructure as code (IAC) can help solve the problem of inconsistent configurations by automating the deployment and configuration of compute instances. With IAC, you can define your infrastructure as code, commit it to version control, and then use a provisioning pipeline to automatically deploy and configure the infrastructure based on the code in version control. This ensures that every compute instance is deployed and configured consistently, eliminating the risk of manual configuration errors. Additionally, implementing a provisioning pipeline with code reviews can help catch any potential issues before they are deployed, further reducing the risk of outages during peak shopping time.

Question: 22

CertyIQ

terraform init initializes a sample main.tf file in the current directory.

- A. True
- B. False

Answer: B

Explanation:

It's not a must for the file name to be "main.tf". It checks all .tf files in the current directory and does the needful

Tested. no main/sample file after in it

Question: 23

CertyIQ

Which two steps are required to provision new infrastructure in the Terraform workflow? (Choose two.)

- A. Destroy

- B. Apply
- C. Import
- D. Init
- E. Validate

Answer: BD

Explanation:

Reference:

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

Question: 24

CertyIQ

Why would you use the terraform taint command?

- A. When you want to force Terraform to destroy a resource on the next apply
- B. When you want to force Terraform to destroy and recreate a resource on the next apply
- C. When you want Terraform to ignore a resource on the next apply
- D. When you want Terraform to destroy all the infrastructure in your workspace

Answer: B

Explanation:

The terraform taint command manually marks a Terraform-managed resource as tainted, forcing it to be destroyed and recreated on the next apply.

Reference:

<https://www.terraform.io/docs/cli/commands/taint.html>

Question: 25

CertyIQ

Terraform requires the Go runtime as a prerequisite for installation.

- A. True
- B. False

Answer: B

Explanation:

Reference:

<https://www.terraform.io/docs/extend/guides/v1-upgrade-guide.html>

" target="_blank" style="word-break: break-all;">

As of September 2019, Terraform provider developers importing the Go module `github.com/hashicorp/terraform`, known as Terraform Core, should switch to `github.com/hashicorp/terraform-plugin-sdk`, the Terraform Plugin SDK, instead.

Why a separate module?

While the `helper/*` and other packages in Terraform Core has served us well, in order for provider development to evolve, the SDK needed to break out into its own repository. Terraform Core's versioning has been oriented towards practitioners. With the "unofficial" SDK existing in the core repository, the SDK becomes tied to Core releases and cannot follow semantic versioning. The new standalone SDK github.com/hashicorp/terraform-plugin-sdk follows semantic versioning starting with v1.0.0.

We will use the term "legacy Terraform plugin SDK" when referring to the version of Terraform Core imported and used by providers.

Question: 26

CertyIQ

When should you use the force-unlock command?

- A. You see a status message that you cannot acquire the lock
- B. You have a high priority change
- C. Automatic unlocking failed
- D. You apply failed due to a state lock

Answer: C

Explanation:

Manually unlock the state for the defined configuration.

Reference:

<https://www.terraform.io/docs/cli/commands/force-unlock.html>

Question: 27

CertyIQ

Terraform can import modules from a number of sources "" which of the following is not a valid source?

- A. FTP server
- B. GitHub repository
- C. Local path
- D. Terraform Module Registry

Answer: A

Explanation:

Correct answer is A:FTP server

Question: 28**CertyIQ**

Which of the following is available only in Terraform Enterprise or Cloud workspaces and not in Terraform CLI?

- A. Secure variable storage
- B. Support for multiple cloud providers
- C. Dry runs with terraform plan
- D. Using the workspace as a data source

Answer: A**Explanation:**

Correct answer is A:Secure variable storage

Question: 29**CertyIQ**

terraform validate validates the syntax of Terraform files.

- A. True
- B. False

Answer: A**Explanation:**

The terraform validate command validates the syntax and arguments of the Terraform configuration files.

Reference:

<https://www.terraform.io/docs/cli/code/index.html>

Question: 30**CertyIQ**

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.)

- A. Run terraform plan -destroy.
- B. This is not possible. You can only show resources that will be created.
- C. Run terraform state rm *.
- D. Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval.

Answer: AD**Explanation:**

A. Run terraform plan -destroy.

D. Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval.

Question: 31

CertyIQ

Which of the following is the correct way to pass the value in the variable num_servers into a module with the input servers?

A. servers = num_servers

B. servers = variable.num_servers

C. servers = var(num_servers)

D. servers = var.num_servers

Answer: D

Explanation:

D to pass a value to a module the syntax is <child module variable name > = <value> Here child module variable name is server reference the value of a variable in main/root module var. <name _of _the variable>

Question: 32

CertyIQ

A Terraform provisioner must be nested inside a resource configuration block.

A. True

B. False

Answer: A

Explanation:

Most provisioners require access to the remote resource via SSH or WinRM, and expect a nested connection block with details about how to connect.

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/connection.html>

Question: 33

CertyIQ

Terraform can run on Windows or Linux, but it requires a Server version of the Windows operating system.

A. True

B. False

Answer: B

Explanation:

B is correct answer : false.

Question: 34

CertyIQ

What does the default "local" Terraform backend store?

- A. tfplan files
- B. Terraform binary
- C. Provider plugins
- D. State file

Answer: D

Explanation:

The local backend stores state on the local filesystem, locks that state using system APIs, and performs operations locally.

Reference:

<https://www.terraform.io/docs/language/settings/backends/local.html>

Question: 35

CertyIQ

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
- 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

Explanation:

Correct answer is A:Run the terraform f m t command during the code linting phase of your CI/CD process

Question: 36

CertyIQ

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

- A. The ability to share modules with public Terraform users and members of Terraform Enterprise Organizations
- B. The ability to tag modules by version or release
- C. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations
- D. The ability to share modules publicly with any user of Terraform

Answer: C

Explanation:

Terraform Cloud's private registry works similarly to the public Terraform Registry and helps you share Terraform providers and Terraform modules across your organization. It includes support for versioning and a searchable list of available providers and modules.

Question: 37

CertyIQ

Which task does terraform init not perform?

- A. Sources all providers present in the configuration and ensures they are downloaded and available locally
- B. Connects to the backend
- C. Sources any modules and copies the configuration locally
- D. Validates all required variables are present

Answer: D

Explanation:

Reference:

<https://www.terraform.io/docs/cli/commands/init.html>

" target="_blank" style="word-break: break-all;">

Usage

Usage: `terraform init [options]`

This command performs several different initialization steps in order to prepare the current working directory for use with Terraform. More details on these are in the sections below, but in most cases it is not necessary to worry about these individual steps.

This command is always safe to run multiple times, to bring the working directory up to date with changes in the configuration. Though subsequent runs may give errors, this command will never delete your existing configuration or state.

Question: 38

CertyIQ

You have declared a variable called `var.list` which is a list of objects that all have an attribute `id`. Which options will produce a list of the IDs? (Choose two.)

- A. `for o in var.list : o => o.id`
- B. `var.list[*].id`
- C. `[var.list[*].id]`
- D. `[for o in var.list : o.id]`

Answer: BD

Explanation:

<https://www.terraform.io/language/expressions/splat>A splat expression provides a more concise way to express a common operation that could otherwise be performed with a for expression.

Question: 39

CertyIQ

Which argument(s) is (are) required when declaring a Terraform variable?

- A. type
- B. default
- C. description
- D. All of the above
- E. None of the above

Answer: E

Explanation:

1. The type argument in a variable block allows you to restrict the type of value that will be accepted as the value for a variable. If no type constraint is set then a value of any type is accepted. Source:

<https://www.terraform.io/language/values/variables>

2. non of the options is mandatory - they are all optional

Question: 40

CertyIQ

When using a module block to reference a module stored on the public Terraform Module Registry such as:

```
module "consul" {  
    source = "hashicorp/consul/aws"  
}
```

How do you specify version 1.0.0?

- A. Modules stored on the public Terraform Module Registry do not support versioning
- B. Append ?ref=v1.0.0 argument to the source path
- C. Add version = "1.0.0" attribute to module block
- D. Nothing " modules stored on the public Terraform Module Registry always default to version 1.0.0

Answer: C

Explanation:

C is correct answer.

The version argument accepts a version constraint string. Terraform will use the newest installed version of the module that meets the constraint; if no acceptable versions are installed, it will download the newest version that meets the constraint.

Version constraints are supported only for modules installed from a module registry, such as the public Terraform Registry or Terraform Cloud's private module registry. Other module sources can provide their own versioning mechanisms within the source string itself, or might not support versions at all. In particular, modules sourced from local file paths do not support version; since they're loaded from the same source repository, they always share the same version as their caller.

Reference:

<https://www.terraform.io/language/modules/syntax#version>

Question: 41

CertyIQ

What features does the hosted service Terraform Cloud provide? (Choose two.)

- A. Automated infrastructure deployment visualization
- B. Automatic backups
- C. Remote state storage
- D. A web-based user interface (UI)

Answer: CD

Explanation:

C and D are correct answers, there is no auto backup in Terraform cloud.

<https://www.terraform.io/enterprise/admin/infrastructure/backup-restore>

Question: 42

CertyIQ

Where does the Terraform local backend store its state?

- A. In the /tmp directory
- B. In the terraform file
- C. In the terraform.tfstate file
- D. In the user's terraform.state file

Answer: C

Explanation:

The local backend stores state on the local filesystem, locks that state using system APIs, and performs operations locally.

Reference:

<https://www.terraform.io/docs/language/settings/backends/local.html>

Question: 43

CertyIQ

Which option can not be used to keep secrets out of Terraform configuration files?

- A. A Terraform provider
- B. Environment variables
- C. A -var flag
- D. secure string

Answer: D

Explanation:

1. It's DWe can use providers to supply variable values (vault for example).We can provide input variable value in parameter for apply command.We can use environment variables.HashiCorp is not mentioning anything about secure strings.Reference:<https://www.terraform.io/language/values/variables>
2. D. secure string is not an option for keeping secrets out of Terraform configuration files.Secure string:
While there is no option for a "secure string" in Terraform, you can use a number of different techniques to encrypt or obfuscate sensitive information in your configuration files. For example, you might use a tool like SOPS to encrypt your Terraform code, or you might use a tool like Vault to store and manage your secrets separately from your code.

Question: 44**CertyIQ**

What is one disadvantage of using dynamic blocks in Terraform?

- A. They cannot be used to loop through a list of values
- B. Dynamic blocks can construct repeatable nested blocks
- C. They make configuration harder to read and understand
- D. Terraform will run more slowly

Answer: C**Explanation:**

1. Looking at documentation "Overuse of dynamic blocks can make configuration hard to read and maintain"
2. correct answer

Question: 45**CertyIQ**

Only the user that generated a plan may apply it.

- A. True
- B. False

Answer: B**Explanation:**

B is correct, a plan can be stored as a file and another person can execute the plan file

Question: 46**CertyIQ**

Examine the following Terraform configuration, which uses the data source for an AWS AMI.
What value should you enter for the ami argument in the AWS instance resource?

```
data "aws_ami" "ubuntu" {  
    ...  
}  
  
resource "aws_instance" "web" {  
    ami = _____  
    instance_type = "t2.micro"  
  
    tags = {  
        Name = "HelloWorld"  
    }  
}
```

- A. aws_ami.ubuntu
- B. data.aws_ami.ubuntu
- C. data.aws_ami.ubuntu.id
- D. aws_ami.ubuntu.id

Answer: C

Explanation:

```
resource "aws_instance" "web"  
ami = data.aws_ami.ubuntu.id
```

Reference:

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance>

Question: 47

CertyIQ

FILL BLANK -

You need to specify a dependency manually.

What resource meta-parameter can you use to make sure Terraform respects the dependency?

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

Answer: depends_on

Question: 48

CertyIQ

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

C. Delete the Terraform state file and execute Terraform apply.

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

Answer: B

Explanation:

Is B for sure. the state file does not contain information about anything else that you might have on the cloud provider.

Question: 49

CertyIQ

What is the name assigned by Terraform to reference this resource?

```
resource "azurerm_resource_group" "dev" {  
  name = "test"  
  location = "westus"  
}
```

A. dev

B. azurerm_resource_group

C. azurerm

D. test

Answer: A

Explanation:

Correct answer is A:dev

Question: 50

CertyIQ

Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged into syslog.

A. True

B. False

Answer: B

Explanation:

1. TF_LOG will enable log and send that to stderr (by default it i screen) not to any syslog server.
TF_LOG_PARH can be set to redirect logs to a specific file

2. B is correct answer. Use TF_LOG_PATH to specify a file path where the log output file should be written. If you do not specify a log path, Terraform writes the specified log output to stderr." Stderr, also known as

standard error, is the default file descriptor where a process can write error messages. In Unix-like operating systems, such as Linux, macOS X, and BSD, stderr is defined by the POSIX standard. Its default file descriptor number is 2. In the terminal, standard error defaults to the user's screen."

<https://www.computerhope.com/jargon/s/stderr.htm#:~:text=Stderr%2C%20also%20known%20as%20standard,defaults%20to%20the%20user's%20screen.>

Question: 51

CertyIQ

Where in your Terraform configuration do you specify a state backend?

- A. The terraform block
- B. The resource block
- C. The provider block
- D. The datasource block

Answer: A

Explanation:

Backends are configured with a nested backend block within the top-level terraform block.

Reference:

<https://www.terraform.io/docs/language/settings/backends/configuration.html>

Question: 52

CertyIQ

In Terraform 0.13 and above, outside of the required_providers block, Terraform configurations always refer to providers by their local names.

- A. True
- B. False

Answer: A

Explanation:

Outside of the required_providers block, Terraform configurations always refer to providers by their local names.

Reference:

<https://www.terraform.io/docs/language/providers/requirements.html>

Question: 53

CertyIQ

What command should you run to display all workspaces for the current configuration?

- A. terraform workspace
- B. terraform workspace show
- C. terraform workspace list
- D. terraform show workspace

Answer: C

Explanation:

terraform workspace list

The command will list all existing workspaces.

Reference:

<https://www.terraform.io/docs/cli/commands/workspace/list.html>

Question: 54

CertyIQ

Terraform providers are always installed from the Internet.

A. True

B. False

Answer: B

Explanation:

Terraform configurations must declare which providers they require, so that Terraform can install and use them.

Reference:

<https://www.terraform.io/docs/language/providers/configuration.html>

Question: 55

CertyIQ

Which of these is the best practice to protect sensitive values in state files?

A. Blockchain

B. Secure Sockets Layer (SSL)

C. Enhanced remote backends

D. Signed Terraform providers

Answer: C

Explanation:

Use of remote backends and especially the availability of Terraform Cloud, there are now a variety of backends that will encrypt state at rest and will not store the state in cleartext on machines running.

Reference:

<https://www.terraform.io/docs/extend/best-practices/sensitive-state.html>

Question: 56

CertyIQ

When does terraform apply reflect changes in the cloud environment?

A. Immediately

B. However long it takes the resource provider to fulfill the request

C. After updating the state file

- D. Based on the value provided to the -refresh command line argument
- E. None of the above

Answer: B

Explanation:

If you are creating a new virtual machine using Terraform, it may take a few minutes for the virtual machine to be created and for it to become available for use. During this time, Terraform will continue to report on the progress of the creation process and will display any errors or issues that may arise. Once the virtual machine has been successfully created, the changes will be reflected in your cloud environment. I go with B

Question: 57

CertyIQ

How would you reference the "name" value of the second instance of this fictitious resource?

```
resource "aws_instance" "web" {  
  count = 2  
  name = "terraform-${count.index}"  
}
```

- A. element(aws_instance.web, 2)
- B. aws_instance.web[1].name
- C. aws_instance.web[1]
- D. aws_instance.web[2].name
- E. aws_instance.web.*.name

Answer: B

Explanation:

B - aws_instance.web[1].name !!!

Index starts at 0 so the second instance would be 1 - the link below confirms this:

<https://www.terraform.io/language/meta-arguments/count#referring-to-instances>

Question: 58

CertyIQ

A Terraform provider is not responsible for:

- A. Understanding API interactions with some service
- B. Provisioning infrastructure in multiple clouds
- C. Exposing resources and data sources based on an API
- D. Managing actions to take based on resource differences

Answer: B

Explanation:

1. The answer should be BA terraform can only provision resource in one Cloud not multiple cloud
2. "A" Terraform provider is responsible for "A" Cloud."Multiple" Terraform providers can be responsible for "Multiple" Clouds.

Question: 59

CertyIQ

Terraform provisioners can be added to any resource block.

- A. True
- B. False

Answer: A

Explanation:

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/syntax.html>

" target="_blank" style="word-break: break-all;">

Terraform includes the concept of provisioners as a measure of pragmatism, knowing that there will always be certain behaviors that can't be directly represented in Terraform's declarative model.

However, they also add a considerable amount of complexity and uncertainty to Terraform usage. Firstly, Terraform cannot model the actions of provisioners as part of a plan because they can in principle take any action. Secondly, successful use of provisioners requires coordinating many more details than Terraform usage usually requires: direct network access to your servers, issuing Terraform credentials to log in, making sure that all of the necessary external software is installed, etc.

The following sections describe some situations which can be solved with provisioners in principle, but where better solutions are also available. We do not recommend using provisioners for any of the use-cases described in the following sections.

Even if your specific use-case is not described in the following sections, we still recommend attempting to solve it using other techniques first, and use provisioners only if there is no other option.

Question: 60

CertyIQ

What is terraform refresh intended to detect?

- A. Terraform configuration code changes
- B. Empty state files
- C. State file drift
- D. Corrupt state files

Answer: C

Explanation:

Reference:

<https://www.hashicorp.com/blog/detecting-and-managing-drift-with-terraform>

" target="_blank" style="word-break: break-all;">

Prior to a plan or apply operation, Terraform does a refresh to update the state file with real-world status. You can also do a refresh any time with `terraform refresh` :

```
$ terraform refresh
aws_instance.example: Refreshing state... (ID: i-011a9893eff09ede1)
```

What Terraform is doing here is reconciling the resources tracked by the state file with the real world. It does this by querying your infrastructure providers to find out what's actually running and the current configuration, and updating the state file with this new information. Terraform is designed to co-exist with other tools as well as manually provisioned resources and so it only refreshes resources under its management.

The output for a refresh is minimal. Terraform lists each resource it is refreshing along with its internal ID. Running `refresh` does not modify infrastructure, but does modify the state file. If the state has drifted from the last time Terraform ran, `refresh` allows that drift to be detected.

Question: 61

CertyIQ

FILL BLANK -

Which flag would you add to terraform plan to save the execution plan to a file?

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

Answer: -out=FILENAME

Explanation:

Reference:

<https://www.terraform.io/docs/cli/commands/plan.html>

" target="_blank" style="word-break: break-all;">

You can use the optional `-out=FILE` option to save the generated plan to a file on disk, which you can later execute by passing the file to `terraform apply` as an extra argument. This two-step workflow is primarily intended for when *running Terraform in automation*.

If you run `terraform plan` without the `-out=FILE` option then it will create a *speculative plan*, which is a description of the effect of the plan but without any intent to actually apply it.

Question: 62

FILL BLANK -

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:

Reference:

<https://www.terraform.io/docs/language/state/index.html>

" target="_blank" style="word-break: break-all;">

State

JUMP TO SECTION ▾

Terraform must store state about your managed infrastructure and configuration. This state is used by Terraform to map real world resources to your configuration, keep track of metadata, and to improve performance for large infrastructures.

This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment.

Question: 63

A Terraform local value can reference other Terraform local values.

A. True

B. False

Answer: A**Explanation:**

Reference:

<https://www.terraform.io/docs/configuration-0-11/locals.html>

" target="_blank" style="word-break: break-all;">

The `locals` block defines one or more local variables within a module. Each `locals` block can have as many locals as needed, and there can be any number of `locals` blocks within a module.

The names given for the items in the `locals` block must be unique throughout a module. The given value can be any expression that is valid within the current module.

The expression of a local value can refer to other locals, but as usual reference cycles are not allowed. That is, a local cannot refer to itself or to a variable that refers (directly or indirectly) back to it.

It's recommended to group together logically-related local values into a single block, particularly if they depend on each other. This will help the reader understand the relationships between variables. Conversely, prefer to define *unrelated* local values in *separate* blocks, and consider annotating each block with a comment describing any context common to all of the enclosed locals.

Question: 64

CertyIQ

Which of the following is not a valid Terraform collection type?

- A. list
- B. map
- C. tree
- D. set

Answer: C

Explanation:

Reference:

<https://www.terraform.io/docs/language/expressions/type-constraints.html>
" target="_blank" style="word-break: break-all;">

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.

Maps can be made with braces (`{}`) and colons (`:`) or equals signs (`=`): `{ "foo": "bar", "bar": "baz" }` OR `{ foo = "bar", bar = "baz" }`. Quotes may be omitted on keys, unless the key starts with a number, in which case quotes are required. Commas are required between key/value pairs for single line maps. A newline between key/value pairs is sufficient in multi-line maps.

Note: although colons are valid delimiters between keys and values, they are currently ignored by `terraform fmt` (whereas `terraform fmt` will attempt vertically align equals signs).

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

Question: 65

CertyIQ

When running the command `terraform taint` against a managed resource you want to force recreation upon, Terraform will immediately destroy and recreate the resource.

- A. True
- B. False

Answer: B

Explanation:

its not immediately

The `terraform taint` command informs Terraform that a particular object has become degraded or damaged. Terraform represents this by marking the object as "tainted" in the Terraform state, and Terraform will propose to replace it in the next plan you create.

Reference:

<https://www.devopsschool.com/blog/terraform-taint-and-untaint-explained-with-example-programs-and-tutorials/>

Question: 66

CertyIQ

All standard backend types support state storage, locking, and remote operations like plan, apply and destroy.

- A. True
- B. False

Answer: B

Explanation:

1. not all of them so False

2. B is correct answer : False. "By default, Terraform uses a backend called local, which stores state as a local file on disk. You can also configure one of the built-in backends listed in the documentation sidebar. Some of these backends act like plain remote disks for state files, while others support locking the state while operations are being performed. This helps prevent conflicts and inconsistencies. The built-in backends listed are the only backends. You cannot load additional backends as plugins.

"<https://www.terraform.io/language/settings/backends/configuration#available-backends>

Question: 67

CertyIQ

How can terraform plan aid in the development process?

- A. Validates your expectations against the execution plan without permanently modifying state
- B. Initializes your working directory containing your Terraform configuration files
- C. Formats your Terraform configuration files
- D. Reconciles Terraform's state against deployed resources and permanently modifies state using the current status of deployed resources

Answer: A

Explanation:

A. Validates your expectations against the execution plan without permanently modifying state

Terraform's plan command is used to create an execution plan that outlines the steps that Terraform will take to reach your desired infrastructure state. It allows you to preview and validate the changes that will be made to your infrastructure before actually making those changes. This can be helpful in the development process because it allows you to see exactly what will be changed and ensure that it aligns with your expectations before you apply those changes.

Reference:

<https://github.com/hashicorp/terraform/issues/19235>

Question: 68

CertyIQ

You would like to reuse the same Terraform configuration for your development and production environments with a different state file for each.

Which command would you use?

- A. terraform import
- B. terraform workspace
- C. terraform state
- D. terraform init

Answer: B

Explanation:

B is correct answer : Workspaces.

<https://www.terraform.io/language/state/workspaces#when-to-use-multiple-workspaces>

Question: 69**CertyIQ**

What is the name assigned by Terraform to reference this resource?

```
mainresource "google_compute_instance" "main" {  
    name = "test"  
}
```

- A. compute_instance
- B. main
- C. google
- D. teat

Answer: B**Explanation:**

Main

is the name of the resources

Question: 70**CertyIQ**

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. Pass variables to Terraform with a "var flag"
- B. Copy the sensitive variables into your Terraform code
- C. Store the sensitive variables in a secure_vars.tf file
- D. Store the sensitive variables as plain text in a source code repository

Answer: A**Explanation:**

Reference:

<https://developer.hashicorp.com/terraform/language/values/variables>

Question: 71**CertyIQ**

Your security team scanned some Terraform workspaces and found secrets stored in a plaintext in state files. How can you protect sensitive data stored in Terraform state files?

- A. Delete the state file every time you run Terraform
- B. Store the state in an encrypted backend
- C. Edit your state file to scrub out the sensitive data
- D. Always store your secrets in a secrets.tfvars file.

Answer: B

Explanation:

Reference:

<https://www.terraform.io/docs/language/state/sensitive-data.html>

" target="_blank" style="word-break: break-all;">

Storing state remotely can provide better security. As of Terraform 0.9, Terraform does not persist state to the local disk when remote state is in use, and some backends can be configured to encrypt the state data at rest.

For example:

- **Terraform Cloud** always encrypts state at rest and protects it with TLS in transit. Terraform Cloud also knows the identity of the user requesting state and maintains a history of state changes. This can be used to control access and track activity. **Terraform Enterprise** also supports detailed audit logging.
- The S3 backend supports encryption at rest when the `encrypt` option is enabled. IAM policies and logging can be used to identify any invalid access. Requests for the state go over a TLS connection.

Question: 72

CertyIQ

In contrast to Terraform Open Source, when working with Terraform Enterprise and Cloud Workspaces, conceptually you could think about them as completely separate working directories.

- A. True
- B. False

Answer: A

Explanation:

Answer is A

From the doc: "Terraform Cloud manages infrastructure collections with workspaces instead of directories. A workspace contains everything Terraform needs to manage a given collection of infrastructure, and separate workspaces function like completely separate working directories."

<https://www.terraform.io/cloud-docs/workspaces>

Question: 73

CertyIQ

You want to know from which paths Terraform is loading providers referenced in your Terraform configuration (*.tf files). You need to enable debug messages to find this out. Which of the following would achieve this?

- A. Set the environment variable `TF_LOG=TRACE`
- B. Set verbose logging for each provider in your Terraform configuration
- C. Set the environment variable `TF_VAR_log=TRACE`
- D. Set the environment variable `TF_LOG_PATH`

Answer: A

Explanation:

Reference:

<https://www.terraform.io/docs/cli/config/environment-variables.html>


" target="_blank" style="word-break: break-all;">

Terraform refers to a number of environment variables to customize various aspects of its behavior. None of these environment variables are required when using Terraform, but they can be used to change some of Terraform's default behaviors in unusual situations, or to increase output verbosity for debugging.

TF_LOG

Enables detailed logs to appear on stderr which is useful for debugging. For example:

```
export TF_LOG=trace
```

Copy 

Question: 74

CertyIQ

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: D

Explanation:

Correct answer is D:By an explicit call

Question: 75

CertyIQ

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, and run terraform apply again without changing any Terraform code?

- A. Terraform will remove the VM from state file
- B. Terraform will report an error
- C. Terraform will not make any changes
- D. Terraform will recreate the VM

Answer: D

Explanation:

Correct answer is D. Terraform will recreate the VM.

In Terraform, the state file is used to store the current state of your infrastructure. When you run terraform apply, Terraform compares the state of your infrastructure as defined in the configuration files with the state recorded in the state file, and then makes any necessary changes to bring the infrastructure into compliance with the configuration.

Question: 76

CertyIQ

Which of these options is the most secure place to store secrets for connecting to a Terraform remote backend?

- A. Defined in Environment variables
- B. Inside the backend block within the Terraform configuration
- C. Defined in a connection configuration outside of Terraform
- D. None of above

Answer: A

Explanation:

We recommend using environment variables to supply credentials and other sensitive data. If you use - backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

Reference:

<https://www.terraform.io/language/settings/backends/configuration#credentials-and-sensitive-data>

Question: 77

CertyIQ

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

- A. Amazon S3
- B. Artifactory
- C. Git
- D. Terraform Cloud

Answer: C

Explanation:

Reference:

<https://www.terraform.io/cdktf/concepts/remote-backends>

<https://developer.hashicorp.com/terraform/cdktf/concepts/remote-backends>

Question: 78

CertyIQ

Which backend does the Terraform CLI use by default?

- A. Terraform Cloud
- B. Consul
- C. Remote
- D. Local

Answer: D

Explanation:

D is correct answer : local

"By default, Terraform implicitly uses a backend called local to store state as a local file on disk. Every other backend stores state in a remote service of some kind, which allows multiple people to access it. Accessing state in a remote service generally requires some kind of access credentials, since state data contains extremely sensitive information."

Reference:

<https://www.terraform.io/docs/language/settings/backends/configuration.html>

Default Backend

If a configuration includes no backend block, Terraform defaults to using the `local` backend, which stores state as a plain file in the current working directory.

Question: 79

When you initialize Terraform, where does it cache modules from the public Terraform Module Registry?

- A. On disk in the /tmp directory
- B. In memory
- C. On disk in the .terraform sub-directory
- D. They are not cached

Answer: C

Explanation:

C is correct answer : hidden terraform directory

"A hidden .terraform directory, which Terraform uses to manage cached provider plugins and modules, record which workspace is currently active, and record the last known backend configuration in case it needs to migrate state on the next run. This directory is automatically managed by Terraform, and is created during initialization."

Reference:

<https://www.terraform.io/docs/language/modules/sources.html>

Question: 80

You write a new Terraform configuration and immediately run terraform apply in the CLI using the local backend. Why will the apply fail?

- A. Terraform needs you to format your code according to best practices first
- B. Terraform needs to install the necessary plugins first
- C. The Terraform CLI needs you to log into Terraform cloud first
- D. Terraform requires you to manually run terraform plan first

Answer: B

Explanation:

1. Need to run Terraform Init first to install the plugins
2. B is the correct, terraform init to download required plugins

Thank you

Thank you for being so interested in the premium exam material.
I'm glad to hear that you found it informative and helpful.

But Wait

I wanted to let you know that there is more content available in the full version. The full paper contains additional sections and information that you may find helpful, and I encourage you to download it to get a more comprehensive and detailed view of all the subject matter.

[Download Full Version Now](#)



Future is Secured
100% Pass Guarantee



24/7 Customer Support
Mail us - certyiqofficial@gmail.com



Free Updates
Lifetime Free Updates!

Total: **283 Questions**

Link: <https://certyiq.com/papers?provider=hashicorp&exam=terraform-associate>