04/03/25

Terraform Questions

1) what is Terraform ?

- Terraform is an open-source infrastructure as code (IaC) tool developed by Hashicorp. It allows users to define, Provision and manage infrastructure using a declarative configuration language called HCL (Hashicorp configuration Language).
- Terraform Supports multiple cloud providers, including AWS, Azure and Google cloud, as well as on-prem Solutions.
- It uses a state file to track infrastructure changes and Ensures that the deployed infrastructure matches the desired configoration.

2) what is infrastructure as a code (Iac) ?

Iac is a method of managing & provisioning IT infrastructure using code, rather than manual configuration. It allows teams to automate the setup & management of their infrastructure, making it more Efficient & consistent.

3) What is Terraform provider?

A Terraform provider is a plugin that allows Terraform to interact with cloud platforms, services or APIs. Providers define how Terraform manage resources like servers, databases and networks.

Example:

- Aws provider: Manage Aws resources (EC2, S3, RDS)
- Azure provider: Manage Azure resources (VMs, Storage)
- Google provider: Manages GCP resources (compute Engine, cloud storage)

4) How to define a provider in Terraform?

```
Provider "aws" {
    region = "us-east-1"
}
```

5) Explain the terraform configuration file.?

A Terraform configuration file is a script written in HCL that defines the infrastructure resources terraform will create & manage. The main configuration file usually has a .tf extension, like main.tf

6) what are the key components of a Terraform configration file?

Key components of a Terraform configuration file:

1. Provider Block: Specifies the cloud or service provider.

```
Provider "aws" {
    region = "us-east-1"
}
```

2. Resource Block: Defines the infrastructure components (like servers, databases)

```
resource "aws_instance" "dev" {
    ami = "ami-36xxxx"
    instance_type = "t2.micro"
}
```

3. Variable Block: Allows reusability with input values.

```
Variable "ami" {
    default = "ami-36xxxx"
}
```

4. output Block: Displays values after applying Terraform

```
output "instance_ip" {
    value = aws_instance.example.public_ip
}
```

5. State file: (terraform.tfstate) - stores infrastructure details to track changes.

8) What is immutable infrastructure?

Immutable infrastructure means that once a system (server, VM or container) is deployed, it is never modified. Instead of updating or modifying an existing EC2 instance, Terraform will create a new instance and destroy the old one when changes are applied.

9) What are the benifits of immutable infrastructure?

- Consistency - no config drift
- Reliability - Fewer manual errors
- Easy rollbacks - Switch to previous version easily.
- Scalability - works well with CI/CD pipelines.

10) What are Terraform Modules, and why they are useful?

A Module is a reusable configuration that groups multiple resources. It helps in organizing & reusing code across different Environments.

11) Explain the difference between terraform plan & terraform apply?

terraform plan: Show what changes terraform will make before applying. (it's a dry run)

terraform apply: Actually applies the changes & modifies infrastructure.

12) what is the purpose of terraform init?

It initializes a terraform project by downloading required provider plugins & setting up the backend.

13) what is the Terraform state file (terraform.tfstate)?

It is a file that stores information about the current infrastructure & track changes.

14) How does Terraform use the state file to track infrastructure?

Terraform compares the current state file with the desired configuration to determine what needs to be created, updated or destroyed.

15) what is the difference between local state & the remote state?

Local state: Stored on the local machine.

Remote state: Stored in a shared backend (Ex: AWS S3) for team collaboration.

16) How do you manage terraform state in a team Environment?

Use remote state storage (AWS S3 with DynamoDB for locking) to prevent conflicts when multiple users apply changes.

17) What is State locking, and why it is important?

State locking prevents multiple users from modifying the state file simultaneously, reducing conflicts.

18) How do you handle Terraform state file corruption?

Restore from a backup or use 'terraform state rm' to manually fix issues.

19) What are Terraform variables, and how they are used?

Variables store dynamic values that can be reused across configurations.

20) How do you pass environment-specific values in Terraform?

Use terraform.tfvars files or pass the variables using the command line

terraform apply -var="region"=us-east-1"

21) What is the difference between var. and local. in Terraform?

var. → Refers to input variables.

local. → Defines local values used within the configuration.

22) Explain the use of Terraform output variables?

Output displays useful information after terraform applies the configuration.

Ex:

```
output "instance_ip" {
  value = "aws_instance.web.public_ip"
}
```

23) Explain immutable infrastructure & how terraform supports it?

Immutable infrastructure replaces resources instead of updating them. Terraform supports this by destroying & recreating resources when changes are made.

24) What happens if someone manually changes an infrastructure resource managed by Terraform?

Terraform detects drift (differences between the state file & actual resources) and updates or reverts changes accordingly.

25) What is Terraform Backend?

A backend defines where Terraform stores its state file (local or remote).

26) What is Terraform workspace ?

A Terraform workspace allows you to manage multiple Environments (like dev, test and prod) within the same Terraform configuration. Each workspace maintains seperate state files, enabling isolated deployements.

27) Why do we use Terraform workspace ?

Terraform workspaces are useful when :

- Managing multiple Environments with the same configuration.
- Isolating state files - Each workspace has a seperate state file, preventing conflicts.
- Switching between different states easily.

28) Terraform workspaces commands list ?

- terraform workspace show - To check the default workspace.
- terraform workspace new <name> - to create a new workspace.
- terraform workspace list - To check available Environments
- terraform workspace select <name> - To switch/select a workspace.
- terraform workspace delete <name> - To delete a workspace.

29) How does terraform store workspace-related state files ?

Local Backend : Workspaces are stored in a subdirectory of terraform.tfstate.d/

Remote backend (like S3) : workspaces store seperate state files with unique names.

30) what is the difference between default and custom workspaces ?

default workspace : The initial workspace when created when Terraform runs.

custom workspaces : created explicitly to manage seperate infrastructure Environments.

31) can different Terraform workspaces share resources ?

No, each workspace maintains its own state, meaning resources are seperate for each workspace.

32) What happens if you delete a terraform workspace ?

This does not delete infrastructure but only removes the workspace state.

33) What is the difference between Terraform workspaces and Terraform Modules?

Workspaces: Manage multiple Environments with seperate states.

Modules: Reusable components for defining infrastructure.

34) At what stage of work flow state file will create?

The terraform state file (terraform.tfstate) is created during the terraform apply stage of the workflow.

Terraform workflow & State file creation:

1. terraform init: Initializes Terraform but does not create state file.
2. terraform plan: shows what will change but does not modify the state.
3. terraform apply: creates or updates the terraform.tfstate file after applying changes.

Note:. The state file is created when terraform Successfully applies resources for the first time.

- It is then updated whenever changes are made using terraform apply or terraform destroy.

35) When you manually delete an Ec2 instance in aws & run terraform apply command what will happen?

When you manually delete an Ec2 instance in AWS & then run terraform apply, Terraform will detect the missing resource and take action based on the configuration and state file.

What happens?

1. Terraform detects drift
   - Terraform compares the state file (terraform.tfstate) with the actual AWS infrastructure.
   - Since the Ec2 is missing, it sees a drift (difference between the expected and actual state)
2. Terraform recreates the Ec2 instance
   - if the resource is still defined in the .tf file, Terraform will recreate the Ec2 instance with the same configuration.
3. If you don't want terraform to recreate it Run "terraform state rm <resource name>" to remove it from state without recreating.

36) What is Remote Backup in terraform?

A Remote backup in terraform refers to storing the Terraform state file (terraform.tfstate) in a remote backend instead of locally.

37) Why do we use remote backup?

1. Prevent data loss: if the local machine crashes, the state file is still safe.
2. Enables team work: multiple users can work on the same infrastructure.
3. Supports state locking: Avoid conflicts when multiple users apply changes.

Note: S3: stores the state file

Dynamo DB: Enables state locking to prevent conflicts.

38) What is the advantage of remote backup of state file? and why we shouldn't store in local?

Storing the Terraform state file (terraform.tfstate) in a remote backend provides several benifits.

1. collaboration: Multiple team members can access & update
2. State locking: prevent conflicts when multiple users apply changes
3. Security and Backup: Remote storage prevents data loss if the local machine crashes or is lost.
4. consistency: Ensures all team members work with the latest infrastructure state.
5. Automation & CI/CD: Allows integration with pipelines for automatic deployements.

Why shouldn't we store the state file locally:

- Risk of data loss: If the local machine is lost/crash, the state file is lost.
- No collaboration: only one person can modify the infrastructure at a time.
- No locking Mechanism: Multiple users may run terraform at the same time, causing conflicts.
- Security Risks: The local state file may contain Sensitive data.

39) What is terraform validate?

The terraform validate command is used to check the Syntax & Structure of terraform configuration files (.tf files)

What it Does:

- Detects Syntax Errors in .tf files
- Ensures correct argument types are used.
- Validate Variable references.

40) What is Data block in Terraform?

A data block in terraform is used to fetch existing information from cloud providers or external systems without creating new resources.

- Read only - Retrieve data but not modify resources
- Avoids hardcoding
- Fetching VPCs, AMIs, SGs.

41) What is Dynamic block in terraform?

A dynamic block in terraform is used to generate multiple nested blocks within a resource based on a variable or loop. It helps when you need to create repeating configurations without writing duplicate code.

Why dynamic Blocks:

- Avoids duplication - Generates multiple blocks dynamically
- Simplifies code - Reduces manual Effort
- uses loops - Iterates over a list

Ex:

```
dynamic "ingress" {
  for_each = [22, 80, 443]
  content {
    from_port = ingress.value
    to_port = ingress.value
    Protocol = "tcp"
  } cidr blocks = ["0.0.0.0/0"]
}
```

- creates 3 ingress rules for ports 22, 80, 443 dynamically.

42) what are terraform provisioners?

Provisioners in terraform are used to Execute Scripts or commands on a local or remote machine after resource creation or destruction.

They help with bootstrapping, configuration management, or custom scripts.

43) What are the types of provisioners in terraform?

Terraform supports 3 types of provisioners:

1. local provisioners: Runs commands on the machine where terraform is Executed. (local-exec)
2. Remote provisioner: Runs commands inside the created resource (an Ec2 instance)
3. File provisioners: is used to copy files from the local machine to a remote instance.

44) What is the difference between local-exec & remote-exec comm provisioners?

Provisioner	Description	Example use case
local-exec	Runs commands on the local machine where terraform runs.	Sending notifications, running scripts locally.
remote-exec	Runs commands inside the created resource (VM, Ec2, etc.)	Installing Software, configuring instances.

45) What is the use of self inside a provisioner?

Self allows access to attribute of the resource it's attached to.

Ex:

```
Provisioner "remote-exec" {
  inline = ["echo ${self.public_ip}"]
```

Here, self.public_ip refers to the public ip of the created instance.

46) What is terraform import?

terraform import is a command used to bring Existing resources into Terraform state without modifying them. It allows terraform to manage resources that were created outside of terraform.

Ex: Importing an AWS ec2 instance into terraform state

```
terraform import aws_instance.example i-1234567890abcdef0
```

47) Why do we need terraform import?

- To bring manually created resources under terraform control.
- To migrate Existing infrastructure to terraform
- To avoid recreating Existing resources.

48) Does terraform import update the .tf file?

No, terraform import only updates the state file (terraform.tfstate), not the terraform configuration (.tf files)

After importing, we must manually write the corresponding resource block in the .tf file.

49) What happens if we don't write the .tf file after importing?

Terraform will track the resource in the state file but won't have a matching .tf configuration. Running terraform plan will show a plan to destroy the resource because terraform assume's it's no longer needed.

50) Can we import multiple resources at once?

No. Terraform only allows one resource at a time using terraform import. you need to run the command seperately for each resource.

51) What are the limitations of terraform import?

1. Does not generate .tf files → you must manually write the configuration.
2. only imports one resource at a time → Bulk imports require multiple commands.
3. Mismatched configurations can cause drift → The .tf file must match the actual resource.

52) How do you verify an imported resource?

After importing, run terraform plan. This ensures the .tf file matches the imported state.

53) Can you import a resource without knowing its ID?

No, you must provide the exact resource ID to import it. you can find the ID from the cloud providers console.

54) How do you remove an imported resource from Terraform state?

terraform state rm <resource_type>.name

Ex:

terraform state rm aws_instance.example.

This removes the resource from terraform state but does not delete the actual cloud resource.

55) What is a terraform refresh?

Terraform refresh is a command that updates the terraform state file to match the current state of your actual infrastructure.

56) What happens if you manually delete an Ec2 instance & run terraform apply?

Terraform detects drift & recreates the instance.