1. What is Terraform?

Terraform is an open-source Infrastructure as Code (IaC) tool used to automate the provisioning and management of infrastructure resources.

2. Why is Infrastructure as Code important?

Infrastructure as Code allows you to define and manage infrastructure using code, enabling consistency, versioning, and automation.

3. What is the difference between Terraform and other configuration management tools like Ansible or Puppet?

Terraform focuses on provisioning and managing infrastructure resources, while tools like Ansible and Puppet are used for configuration management and automation.

4. What are the key benefits of using Terraform?

Infrastructure as Code, version control, automation, resource provisioning, and easy reproducibility.

5. How does Terraform achieve declarative infrastructure management?

In Terraform, you define the desired state of your infrastructure in configuration files, and Terraform ensures the actual state matches the defined state.

6. What is a Terraform Provider?

A Terraform Provider is a plugin that enables Terraform to interact with a specific cloud provider or service.

7. How do you initialize a Terraform project?

Use the terraform init command to initialize a Terraform project, which downloads necessary providers and modules.

8. What is a Terraform Resource?

A Terraform Resource is a representation of a specific infrastructure component, such as a virtual machine or a storage bucket.

9. How do you create a Terraform Resource in a configuration file?

Define a resource block using the appropriate resource type and configure its attributes.

10. What is a Terraform Module?

- A Terraform Module is a collection of related resources and configurations that can be reused across different projects.

11. How do you apply changes to infrastructure using Terraform?

- Use the terraform apply command to execute the changes defined in the configuration files and provision or modify resources accordingly.

12. What is the purpose of a Terraform State file?

- The Terraform State file keeps track of the actual infrastructure that has been provisioned and helps Terraform manage changes and updates.

13. How can you manage environment-specific configurations in Terraform?

- Use variables and input files to parameterize your Terraform configuration, allowing you to customize configurations for different environments.

14. How do you reference output values from a Terraform configuration?

- You can reference output values defined in a Terraform configuration using interpolation syntax, such as ${module.module_name.output_name}.

15. What is a Terraform Data Source?

- A Terraform Data Source allows you to fetch information from external sources, such as existing resources or external APIs, for use in your configurations.

16. How does Terraform handle changes to existing resources?

- Terraform performs a "plan" to determine the changes required to bring the actual state in line with the desired state, then applies those changes.

17. How can you ensure secure storage of sensitive information, like API keys, in Terraform configurations?

- Use environment variables, or store sensitive data in a separate file (e.g., .tfvars) and use variable interpolation.

18. What is the "terraform.tfvars" file used for?

- The terraform.tfvars file is used to define variable values, allowing you to separate sensitive or environment-specific data from your main configuration.

19. How does Terraform handle dependencies between resources?

- Terraform automatically manages dependencies based on resource configurations and ensures resources are created in the correct order.

20. What is a "Terraform State Backend"?

- A Terraform State Backend is a storage location where Terraform stores the state file, allowing for collaboration and safe management of state.

21. How can you manage multiple environments (e.g., development, production) with Terraform?

- Use workspaces or separate configuration files for different environments, allowing you to maintain separate states and configurations.

22. What is the purpose of "Terraform Remote State"?

- Terraform Remote State allows you to store the state file in a remote backend, enabling collaboration and secure storage.

23. How do you manage shared infrastructure components, such as a VPC, across multiple projects?

- Use Terraform Modules to encapsulate shared infrastructure components, making them reusable across different projects.

24. What is the Terraform "plan" command used for?

- The terraform plan command shows the execution plan, outlining the changes that will be applied to the infrastructure based on the configuration.

25. How do you destroy infrastructure resources created with Terraform?

- Use the terraform destroy command to remove all resources defined in the configuration.

26. What is "Terraform Backends"?

- Terraform Backends define where state files are stored, including local storage, remote storage, and third-party solutions.

27. How do you handle Terraform state locking to prevent concurrent updates?

- State locking is automatically handled by the chosen backend, ensuring that only one user at a time can modify the state.

28. What is the difference between "terraform apply" and "terraform plan"?

- terraform plan shows the execution plan without making any actual changes, while terraform apply executes the changes.

29. How do you organize your Terraform configurations for a complex project?

- Use directories to separate different components or modules of your infrastructure, keeping the configurations modular and maintainable.

30. What is the purpose of Terraform "locals"?

- Locals allow you to define values that can be reused within the configuration, reducing redundancy and improving readability.

31. How can you use conditionals and loops in Terraform configurations?

- Use conditional expressions and "for_each" loops to dynamically configure resources based on certain conditions or variables.

32. What is the "terraform import" command used for?

- The terraform import command allows you to import existing resources into your Terraform state, enabling management through Terraform.

33. How do you version control your Terraform configurations?

- Use a version control system like Git to manage and track changes to your Terraform configuration files.

34. How can you pass variables to Terraform configurations from the command line?

- Use the -var or -var-file options when running the terraform apply or terraform plan commands.

35. What is a "Terraform Sentinel Policy"?

- Sentinel Policies in Terraform allow you to define and enforce governance rules to ensure compliance and security.

36. How can you manage third-party Terraform modules?

- Use the Terraform Registry to search for and use community-contributed modules or create your own private modules.

37. What is a "Terraform Remote State Data Source"?

- The Terraform Remote State Data Source allows you to reference data from a remote state stored in a backend.

38. What is the role of the "Terraform Null Resource"?

- The Terraform Null Resource is used to create additional actions or behaviors in the execution plan that don't directly create infrastructure resources.

39. How does Terraform handle drift in infrastructure resources?

- Drift occurs when the actual state of resources differs from the state recorded in the Terraform state file. Terraform can detect drift and plan to bring the state back in sync.

40. What is the "count" meta-argument used for in Terraform?

- The "count" meta-argument allows you to create multiple instances of a resource using the same configuration block.

41. How can you reference attributes from one resource in another resource's configuration?

- Use the interpolation syntax to reference attributes from other resources, like ${resource_type.resource_name.attribute}.

42. What are "Terraform Variables" and how are they defined?

- Terraform Variables are used to parameterize configurations. They can be defined in configuration files or external variable files (e.g., .tfvars).

43. How do you use "Terraform Outputs" to retrieve information from your infrastructure?

- Terraform Outputs define values that can be extracted after resource creation, making it easier to reference information in later configurations.

44. How does Terraform handle destroying resources that have dependencies?

- Terraform follows dependency relationships and destroys resources in the correct order to avoid issues with dependent resources.

45. What is "Terraform State Locking"?

- Terraform State Locking prevents concurrent modifications to the state file by locking it during operations like apply and destroy.

46. How do you define a variable with a default value in Terraform?

- Define the variable in your configuration file with a default attribute, like variable "example" { default = "default_value" }.

47. How can you import a Terraform module into your configuration?

- Use the module block to reference the module by its source and optionally provide input variables.