What is *Terraform*

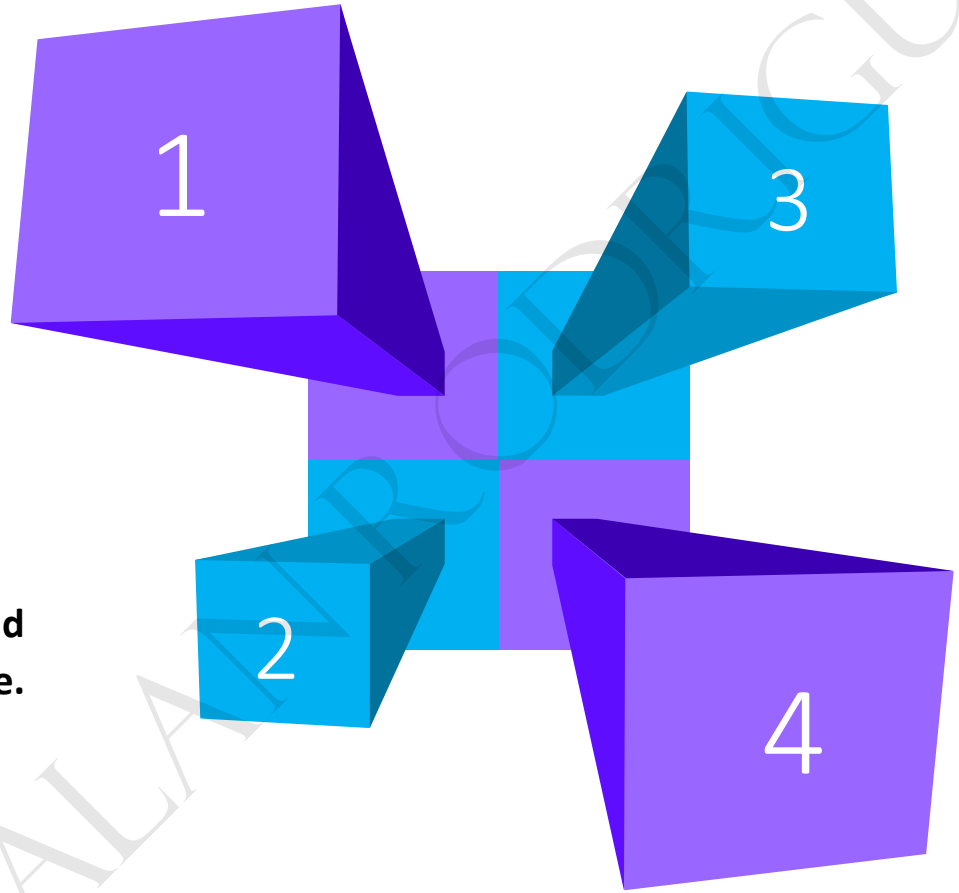
What is Terraform

Tool

Open-source tool developed by HashiCorp.

Language

It uses a declarative language to define the code.



Infrastructure

It helps to automate and manage your infrastructure.

Cloud

You can use it to build infrastructure on multiple cloud environments.

Terraform *Terms*

Workflow

Core Terraform workflow

1. Write – Here you create the code configuration file.

2. Plan – Here you can see the changes that are going to be made by Terraform.

3. Apply – This will apply the changes.

Terms

Terraform configuration – This is the complete document in the Terraform language. This tells terraform how to manage the infrastructure.

Providers – These are plugins for Terraform. This helps Terraform to work with cloud providers.

Terraform *configuration file*

Configuration file

You can write the configuration file either in JSON or HCL.

Argument – This assigns a value to a particular name

Block – This is a container for content.

In a block you will define the type – For example resource.

You then define labels within the block.

Configuration file

The resource block is the most important block.

This defines the infrastructure objects that need to be created.

In the body of the block you define configuration arguments for the resource.

Terraform *plan and apply*

Initialization

terraform init

This is used to initialize a working directory that contains the Terraform configuration files.

This is the first command that needs to run.

This command will also install any plugins required by the providers.

This command will also create a dependency lock file.

Plan

terraform plan -out main.tfplan

This command is going to create an execution plan.

Here you can preview the changes that Terraform is going to carry out.

You can specify to save the generated file to disk.

This command will also create a dependency lock file.

Apply

terraform apply

This command executes the actions within the Terraform plan.

Terraform *state*

Terraform state data

Terraform maintains state data that maps the real-world resources to the configuration.

The state data is stored in a local file named terraform.tfstate.

The state file can also be stored remotely which works best for a team environment.

The state file helps terraform to lay out a plan when it comes to the configuration file.

Terraform state data

Do NOT modify the terraform state file.

Make changes to your configuration file to make changes to your infrastructure.

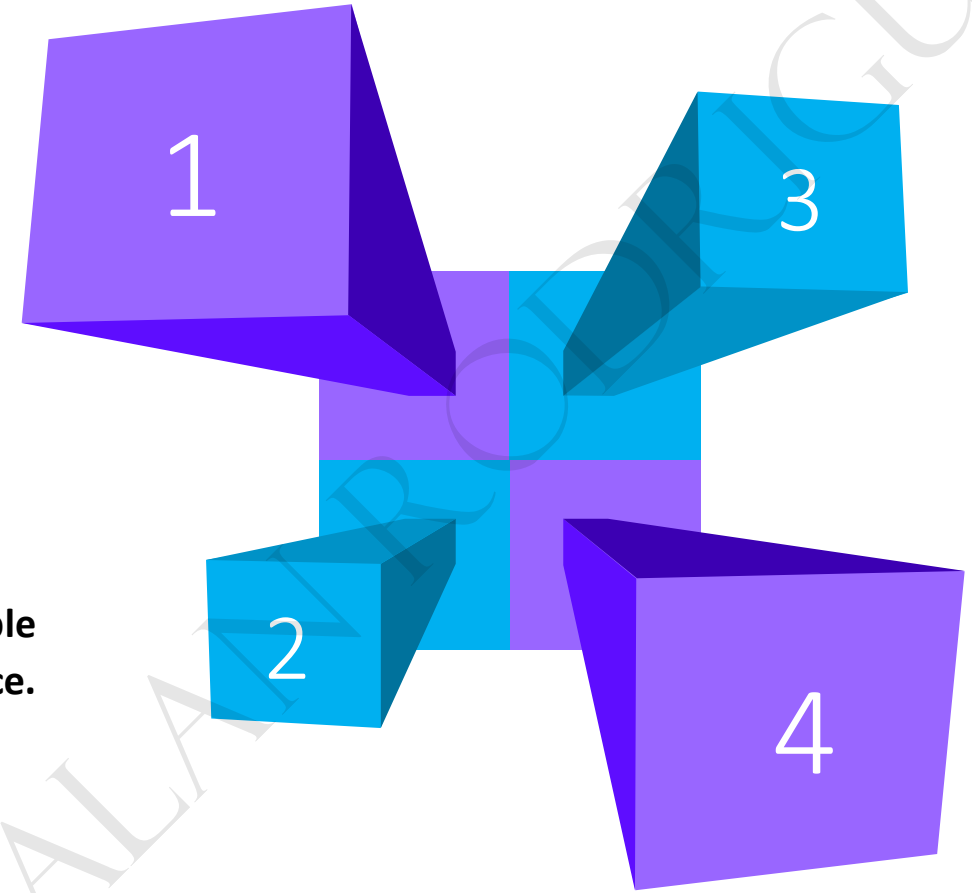
Its also advised that when you maintain your infrastructure with Terraform , don’t make manual changes to the resources.

count *meta-Argument*

The count Meta-Argument

Several objects

This helps to manage several resources in your terraform configuration file.



Index

You can use count.index to get a handle to each unique instance.

Multiple instances

You can create multiple instances of a resource.

Numeric

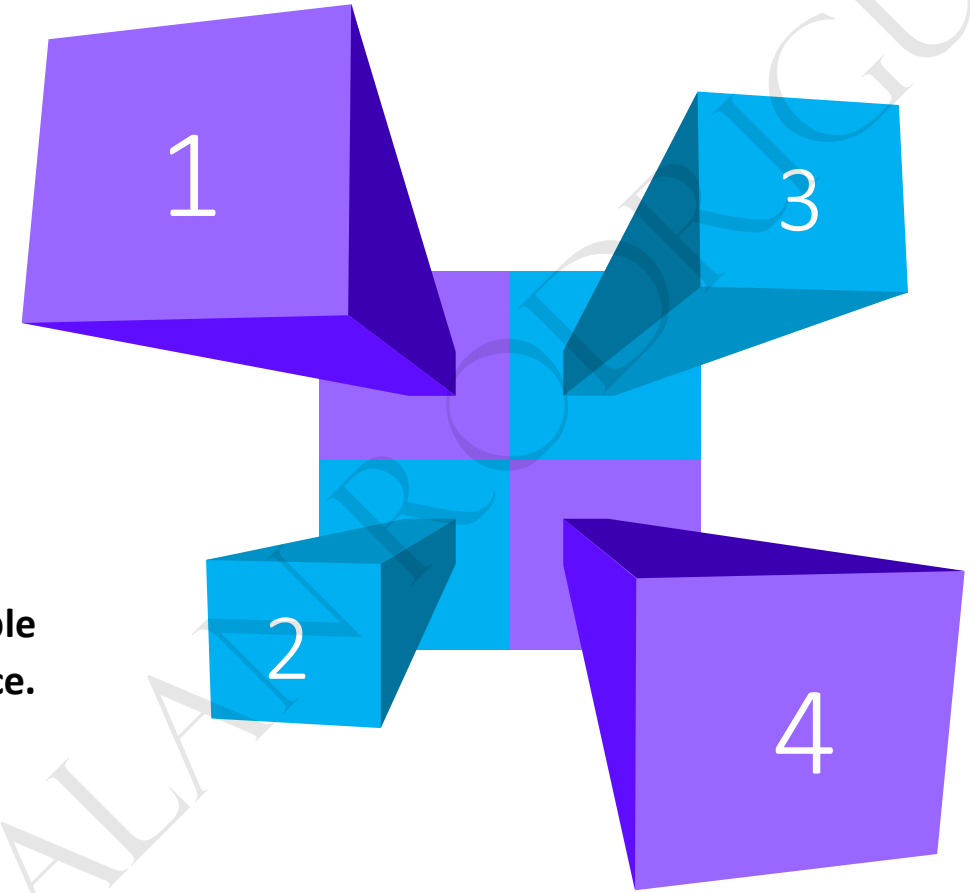
The count meta-argument accepts numeric expressions.

for_each *meta-Argument*

The for_each Meta-Argument

Several objects

This helps to manage several resources in your terraform configuration file.



Map or strings

Here you define a map or set of strings. Terraform will create an instance for each member in that map or set.

Multiple instances

You can create multiple instances of a resource.

each object

The each object helps to modify the configuration of each instance.

Data *Sources*

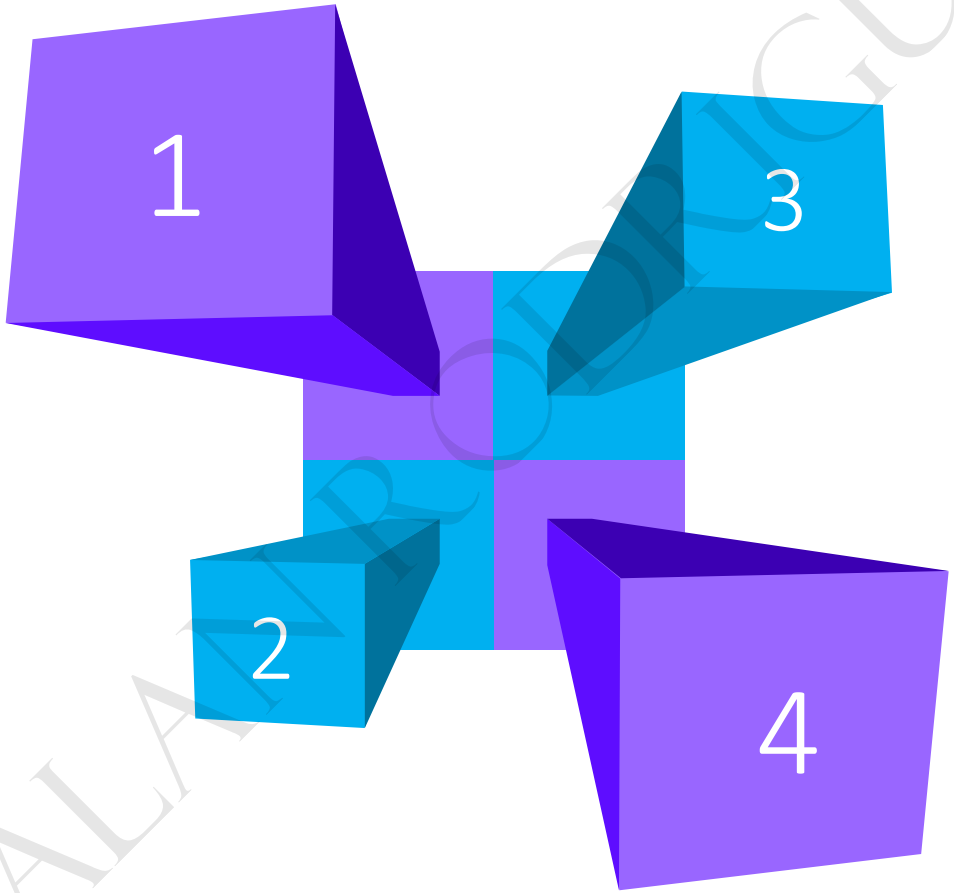
Data Sources

Outside resources

This helps to use information defined outside of Terraform.

Data block

The data source is defined with the use of the data block.



Single resource

Each data source is associated with a single resource.

Provider

The data source depends on the provider.

Provisioners

Provisioners

This can be used to implement certain behaviors that cannot be implemented via Terraform's declarative model.

local-exec – This can be used to invoke a local executable after a resource is created.

This will invoke a process on the machine that is running Terraform.

Provisioners

Connection Block

This describes how to access a remote resource.

Connection blocks can be nested directly within a resource block. This would affect the resource's provisioners.

It could also come within a provisioner block. This would affect the provisioner.

Provisioners

file provisioner

This can be used to copy files or directories from the machine running Terraform to the newly created resource.

This provisioner supports both ssh and winrm type connections.

Azure Web App Logging

Azure Web App Logging

You get a set of logging features that are available for the Azure Web App.

The different types of logging that are available are

Application Logging – This captures log messages that are generated by your application code.

Web server logging – This records raw HTTP request data.

Azure Web App Logging

Detailed Error Messages – This stores copies of the .htm error pages that would have been sent to the client browser.

Deployment logging – These are logs when you publish content to an application.

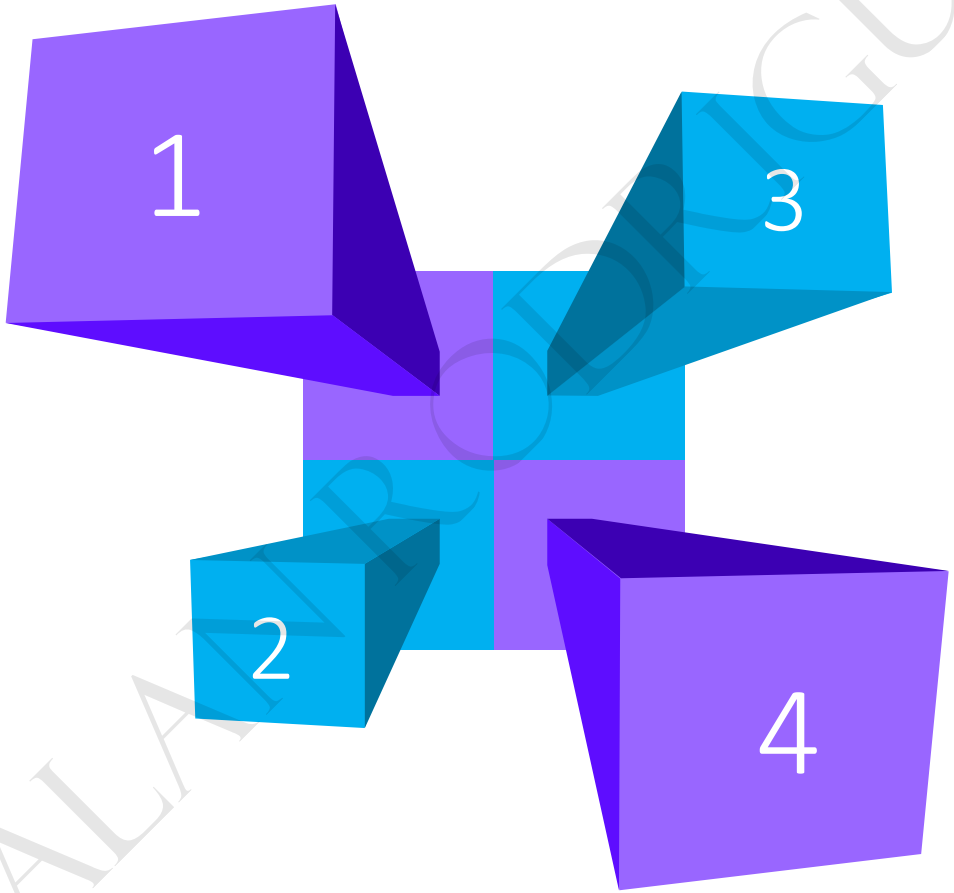
You can also stream logs in real time.

Terraform Cloud

Terraform Cloud

SaaS

This is a software as a service offering wherein a lot of aspects are managed on the cloud.



Backup

Terraform cloud also maintains backups of the state file.

Local CLI

You can still run your local CLI to work with Terraform cloud, but the state file can be managed on the cloud itself.

Users

You can have multiple users work with Terraform Cloud.