terraform

terraform plan

The `plan` will check the configuration files (basically all the `*.tf` files in the directory) and will show you the items or changes going to made on target infrastructure or resources. Please note, this command will not actually perform the planned actions.

`terraform plan`

terraform destroy

Warning: `destroy` will delete all resource but with confirmation

Destroying Resources in terraform

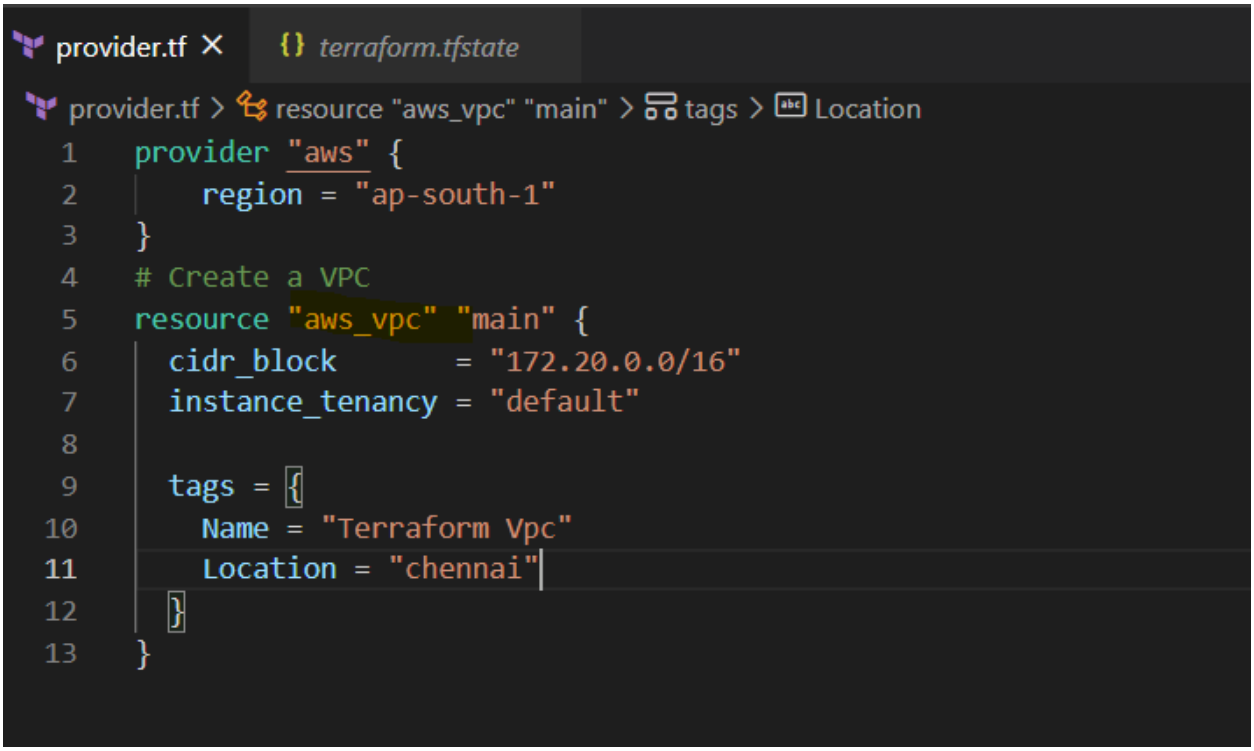
1. To delete all resources

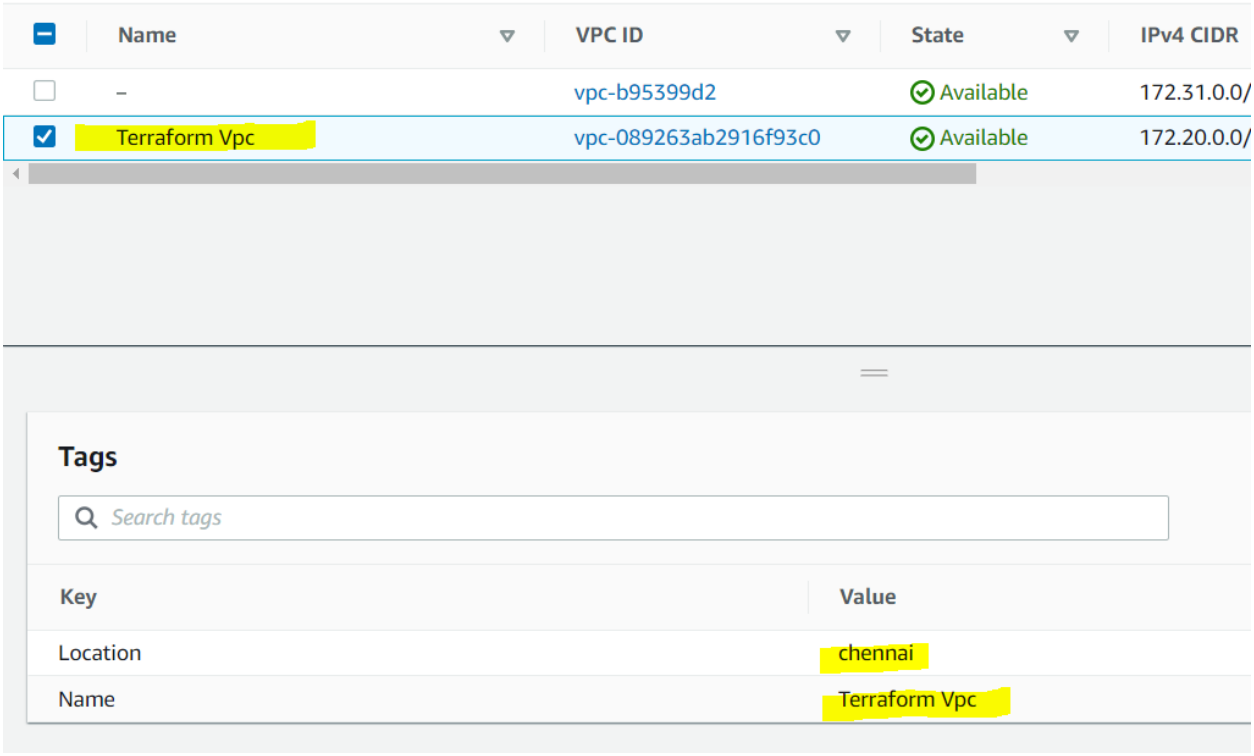
a. terraform destroy

2. To delete a specific resource

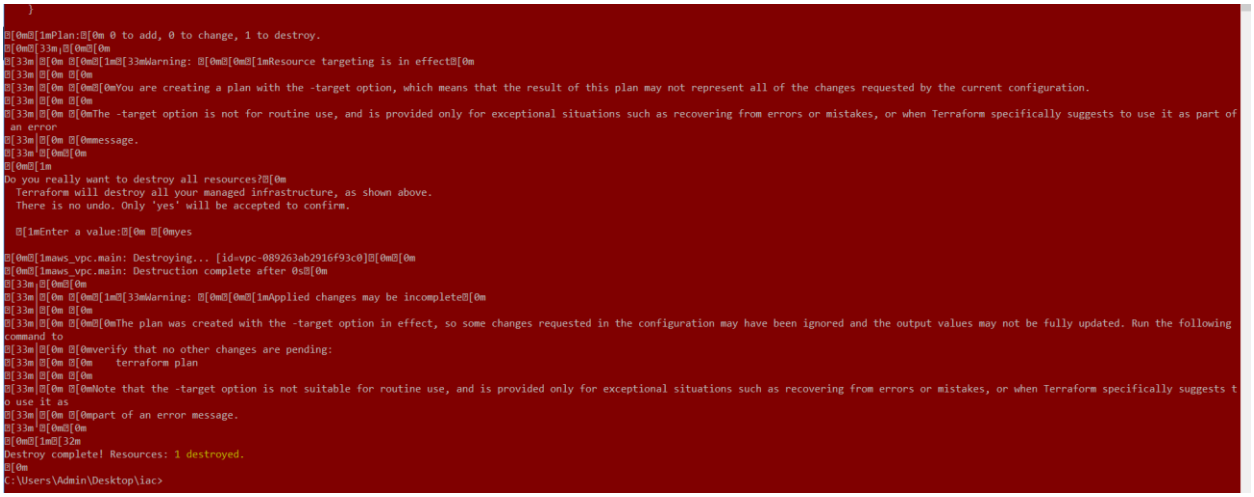
a. Comment that resource code in terraform script. And run terraform apply

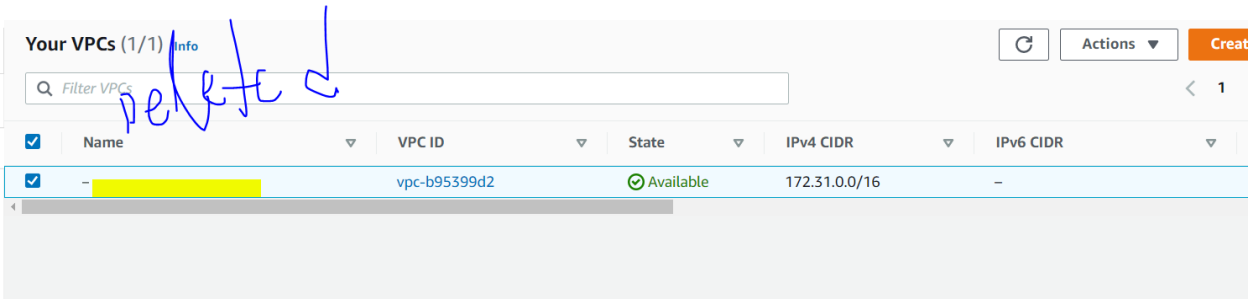
b. terraform destroy -target=resource






```
C:\Users\Admin\Desktop\iac>terraform destroy -target=aws_vpc.main
aws_vpc.main: Refreshing state... [id=vpc-089263ab2916f93c0]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_vpc.main will be destroyed
  - resource "aws_vpc" "main" {
      - arn                              = "arn:aws:ec2:ap-south-1:615086145317:vpc/vpc-089263ab2916f93c0"
      - assign_generated_ipv6_cidr_block = false
      - cidr_block                       = "172.20.0.0/16"
      - default_network_acl_id           = "acl-091212d66b722abf0"
      - default_route_table_id           = "rtb-0dbaa50c6ad356273"
      - default_security_group_id        = "sg-00d8771ac440a5d44"
      - dhcp_options_id                  = "dopt-3e289955"
      - enable_dns_hostnames             = false
      - enable_dns_support               = true
      - id                               = "vpc-089263ab2916f93c0"
      - instance_tenancy                 = "default"
      - main_route_table_id              = "rtb-0dbaa50c6ad356273"
      - owner_id                         = "615086145317"
      - tags                             = {
          - "Location" = "chennai"
          - "Name"     = "Terraform Vpc"
        }
      - tags_all                         = {
          - "Location" = "chennai"
          - "Name"     = "Terraform Vpc"
        }
    }
```





Terraform Auto Approve

- In automation we do not use prompts

- But terraform by default prompts for your confirmations, if you want to auto confirm use following command

terraform apply --auto-approve

```
C:\Users\Admin\Desktop\iac>terraform apply --auto-approve

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  [32m+[0m create
[0m
Terraform will perform the following actions:

[1m  # aws_vpc.main[0m will be created[0m[0m
[0m  [32m+[0m[0m resource "aws_vpc" "main" {
      [32m+[0m [0m[1m[0marn[0m[0m                                  = (known after apply)
      [32m+[0m [0m[1m[0massign_generated_ipv6_cidr_block[0m[0m = false
      [32m+[0m [0m[1m[0mcidr_block[0m[0m                           = "172.20.0.0/16"
      [32m+[0m [0m[1m[0mdefault_network_acl_id[0m[0m               = (known after apply)
      [32m+[0m [0m[1m[0mdefault_route_table_id[0m[0m               = (known after apply)
      [32m+[0m [0m[1m[0mdefault_security_group_id[0m[0m            = (known after apply)
      [32m+[0m [0m[1m[0mdhcp_options_id[0m[0m                      = (known after apply)
      [32m+[0m [0m[1m[0menable_classiclink[0m[0m                   = (known after apply)
      [32m+[0m [0m[1m[0menable_classiclink_dns_support[0m[0m       = (known after apply)
      [32m+[0m [0m[1m[0menable_dns_hostnames[0m[0m                 = (known after apply)
      [32m+[0m [0m[1m[0menable_dns_support[0m[0m                   = true
      [32m+[0m [0m[1m[0mid[0m[0m                                  = (known after apply)
      [32m+[0m [0m[1m[0minstance_tenancy[0m[0m                     = "default"
      [32m+[0m [0m[1m[0mipv6_association_id[0m[0m                  = (known after apply)
      [32m+[0m [0m[1m[0mipv6_cidr_block[0m[0m                      = (known after apply)
      [32m+[0m [0m[1m[0mmain_route_table_id[0m[0m                  = (known after apply)
      [32m+[0m [0m[1m[0mowner_id[0m[0m                             = (known after apply)
      [32m+[0m [0m[1m[0mtags[0m[0m                                 = {
          [32m+[0m [0m"Location" = "chennai"
          [32m+[0m [0m"Name"     = "Terraform Vpc"
        }
      [32m+[0m [0m[1m[0mtags_all[0m[0m                             = {
          [32m+[0m [0m"Location" = "chennai"
          [32m+[0m [0m"Name"     = "Terraform Vpc"
        }
    }

[0m[1mPlan:[0m 1 to add, 0 to change, 0 to destroy.
[0m[0m[1maws_vpc.main: Creating...[0m[0m
[0m[1maws_vpc.main: Creation complete after 2s [id=vpc-09d59fe18330e41a0][0m
[0m[1m[32m
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
[0m
C:\Users\Admin\Desktop\iac>
```

Terraform State File

In a multi developer environment we need to keep the state file in a central location where all developers point to the same file.

There are several options to maintain state files in remote locations.

The best and most common option is S3.

```
C:\Users\Admin\Desktop\iac>terraform state list
aws_vpc.main
```

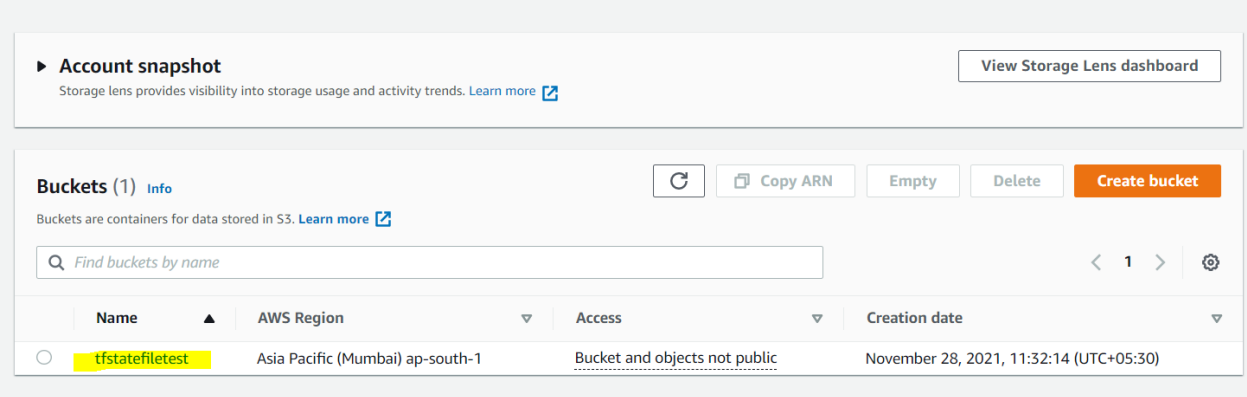
```
Subcommands:
    list                List resources in the state
    mv                  Move an item in the state
    pull                Pull current state and output to stdout
    push                Update remote state from a local state file
    replace-provider    Replace provider in the state
    rm                  Remove instances from the state
    show                Show a resource in the state
```

```
C:\Users\Admin\Desktop\iac>terraform state show  aws_vpc.main
# aws_vpc.main:
resource "aws_vpc" "main" {
    [1m[0marn[0m[0m                                 = "arn:aws:ec2:ap-south-1:615086145317:vpc/vpc-09d59fe18330e41a0"
    [1m[0massign_generated_ipv6_cidr_block[0m[0m = false
    [1m[0mcidr_block[0m[0m                          = "172.20.0.0/16"
    [1m[0mdefault_network_acl_id[0m[0m              = "acl-0b1cd9a2f12128d80"
    [1m[0mdefault_route_table_id[0m[0m              = "rtb-0ef952c5f47fad214"
    [1m[0mdefault_security_group_id[0m[0m           = "sg-0fa580eb3679a48c2"
    [1m[0mdhcp_options_id[0m[0m                     = "dopt-3e289955"
    [1m[0menable_dns_hostnames[0m[0m                = false
    [1m[0menable_dns_support[0m[0m                  = true
    [1m[0mid[0m[0m                                  = "vpc-09d59fe18330e41a0"
    [1m[0minstance_tenancy[0m[0m                    = "default"
    [1m[0mmain_route_table_id[0m[0m                 = "rtb-0ef952c5f47fad214"
    [1m[0mowner_id[0m[0m                            = "615086145317"
    [1m[0mtags[0m[0m                                = {
        "Location" = "chennai"
        "Name"     = "Terraform Vpc"
    }
    [1m[0mtags_all[0m[0m                            = {
        "Location" = "chennai"
        "Name"     = "Terraform Vpc"
    }
}[0m[0m
```

Configure Remote state file using S3

- Create S3 bucket for storing statefile, this cannot be created using the same terraform project.

- Configure remote state in terraform



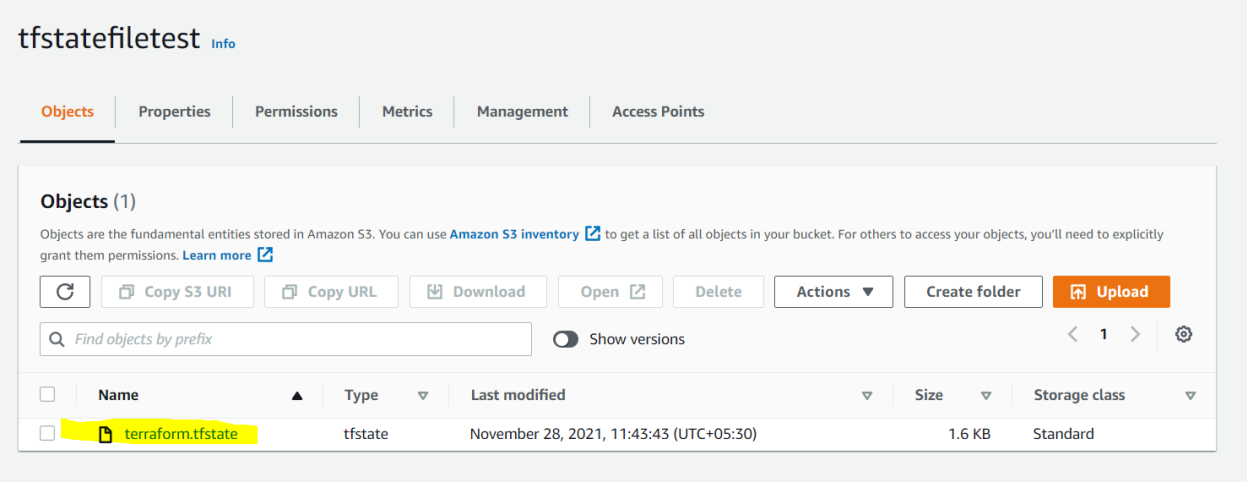
Ref:

https://www.terraform.io/docs/language/settings/backends/s3.html

provider.tf > terraform > backend "s3" > key

```
provider "aws" {
    region = "ap-south-1"
}
# Create a VPC
resource "aws_vpc" "main" {
  cidr_block       = "172.20.0.0/16"
  instance_tenancy = "default"

  tags = {
    Name = "Terraform Vpc"
    Location = "chennai"
  }
}

terraform {
  backend "s3" {
    bucket = "tfstatefiletest"
    key    = "terraform.tfstate"
    region = "ap-south-1"
  }
}
```

```
C:\Users\Admin\Desktop\iac>terraform init

[0m[1mInitializing the backend...[0m
[0m[1mDo you want to copy existing state to the new backend?[0m
  Pre-existing state was found while migrating the previous "local" backend to the
  newly configured "s3" backend. No existing state was found in the newly
  configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.

  [1mEnter a value:[0m [0myes

[0m[32m
Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.[0m

[0m[1mInitializing provider plugins...[0m
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.67.0

[0m[1m[32mTerraform has been successfully initialized![0m[32m[0m
[0m[32m
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.[0m

C:\Users\Admin\Desktop\iac>
```

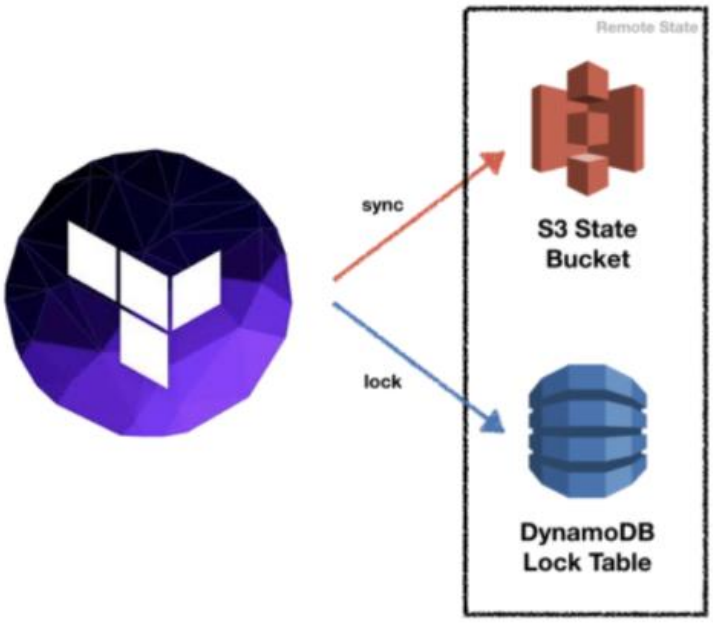


How do you lock a state?

DynamoDB State Locking

The following configuration is optional:

- `dynamodb_endpoint` - (Optional) Custom endpoint for the AWS DynamoDB API. This can also be sourced from the `AWS_DYNAMODB_ENDPOINT` environment variable.
- `dynamodb_table` - (Optional) Name of DynamoDB Table to use for state locking and consistency. The table must have a primary key named `LockID` with type of `string`. If not configured, state locking will be disabled.



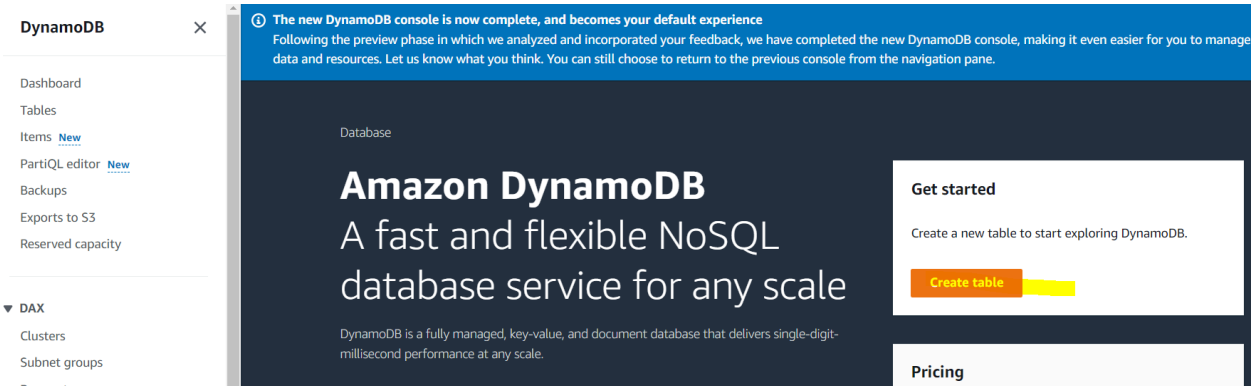


Table name

This will be used to identify your table.

terraform-lock

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

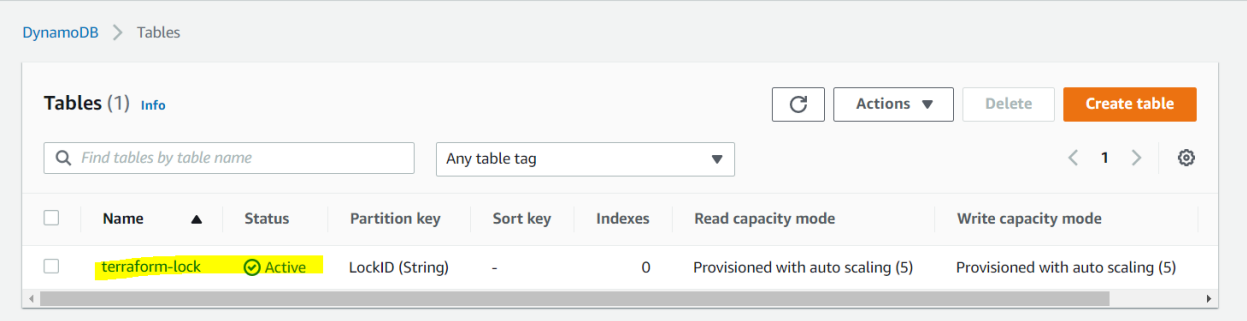
Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table hosts for scalability and availability.

LockID

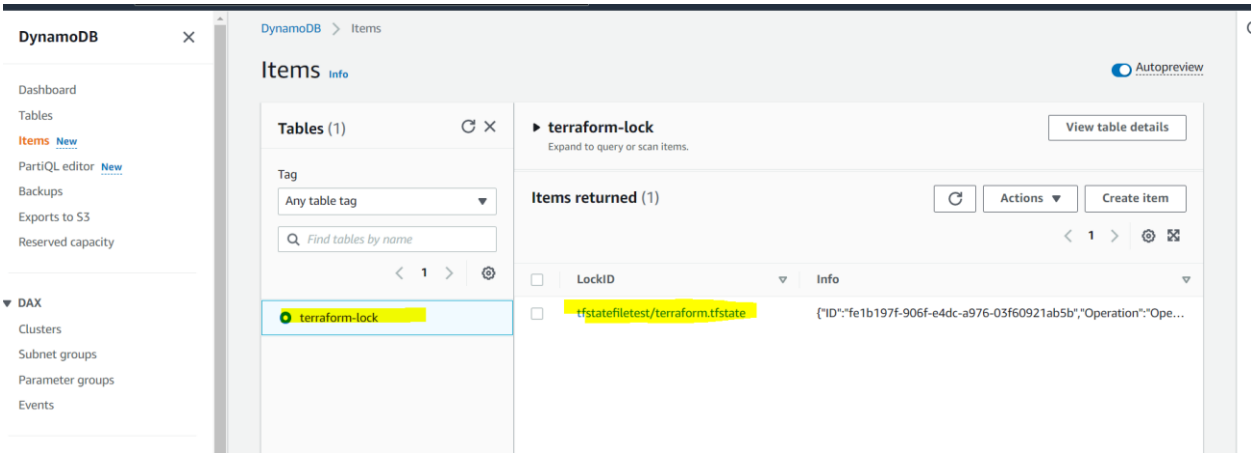
String

1 to 255 characters and case sensitive.




```
C:\Users\Admin\Desktop\iac>terraform init

[0m[1mInitializing the backend...[0m

[0m[1mInitializing provider plugins...[0m
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.67.0

[0m[1m[32mTerraform has been successfully initialized![0m[32m[0m
```

```
C:\Users\Admin\Desktop\iac>terraform apply --auto-approve
Acquiring state lock. This may take a few moments...
[0m[1maws_vpc.main: Refreshing state... [id=vpc-09d59fe18330e41a0][0m

[0m[1m[32mNo changes.[0m[1m Your infrastructure matches the configuration.[0m

[0mTerraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
Releasing state lock. This may take a few moments...
[0m[1m[32m
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
[0m
C:\Users\Admin\Desktop\iac>
```



Ref:

https://quileswest.medium.com/how-to-lock-terraform-state-with-s3-bucket-in-dynamodb-3ba7c4e637