Lab Exercise 12– Creating an AWS RDS Instance in Terraform

Objective:

Learn how to use Terraform to create an AWS RDS instance.

Prerequisites:

- Terraform installed on your machine.
- AWS CLI configured with the necessary credentials.

Steps:

1. Create a Terraform Directory:

```
mkdir terraform-rds
cd terraform-rds
```

2. Create Terraform Configuration Files:

Create a file named main.tf:

 # main.tf

```
terraform {
  required_providers {
    aws = {
```

```
    source  = "hashicorp/aws"
    version = "5.68.0"
   }
  }
}

provider "aws" {
  access_key = "ACCESS_KEY_HERE"
  secret_key = "SECRET_KEY_HERE"
  region     = "ap-south-1"
}

# Replace with your actual VPC ID if needed
data "aws_vpc" "default" {
  default = true
}

resource "aws_security_group" "rds_sg" {
  name        = "rds-sg"
  description = "Allow MySQL traffic"
  vpc_id      = data.aws_vpc.default.id

  ingress {
    from_port   = 3306
    to_port     = 3306
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
```

```
 }

 egress {
  from_port   = 0
  to_port     = 0
  protocol    = "-1"
  cidr_blocks = ["0.0.0.0/0"]
 }
}

resource "aws_db_instance" "my_rds" {
 allocated_storage    = 20
 engine               = "mysql"
 engine_version       = "5.7"
 instance_class       = "db.t3.micro"
 db_name              = "upesdb"
 username             = "admin"
 password             = "admin123"
 parameter_group_name = "default.mysql5.7"
 skip_final_snapshot  = true
 publicly_accessible  = true
 vpc_security_group_ids = [aws_security_group.rds_sg.id]
}
```

- Replace "YourPassword123" with a secure password and "your-security-group-id" with your actual security group ID.

- In this configuration, we define an AWS RDS instance with specific settings, such as engine type, instance class, and security group.

3. Initialize, Plan and Apply:

- Run the following Terraform commands to initialize, plan and apply the configuration:

terraform init

```
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.68.0"...
- Installing hashicorp/aws v5.68.0...
- Installed hashicorp/aws v5.68.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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.
```

terraform plan

```
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds> terraform plan
data.aws_vpc.default: Reading...
data.aws_vpc.default: Read complete after 1s [id=vpc-077a2fb10879758de]

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

Terraform will perform the following actions:

  # aws_db_instance.my_rds will be created
  + resource "aws_db_instance" "my_rds" {
      + address                               = (known after apply)
      + allocated_storage                     = 20
      + apply_immediately                     = false
      + arn                                   = (known after apply)
      + auto_minor_version_upgrade            = true
      + availability_zone                     = (known after apply)
      + backup_retention_period               = (known after apply)
      + backup_target                         = (known after apply)
      + backup_window                         = (known after apply)
```

```
      + cidr_blocks      = [
          + "0.0.0.0/0",
        ]
      + from_port        = 3306
      + ipv6_cidr_blocks = []
      + prefix_list_ids  = []
      + protocol         = "tcp"
      + security_groups  = []
      + self             = false
      + to_port          = 3306
        # (1 unchanged attribute hidden)
    },
  ]
  + name                   = "rds-sg"
  + name_prefix            = (known after apply)
  + owner_id               = (known after apply)
  + revoke_rules_on_delete = false
  + tags_all               = (known after apply)
  + vpc_id                 = "vpc-077a2fb10879758de"
}

Plan: 2 to add, 0 to change, 0 to destroy.
```

terraform apply

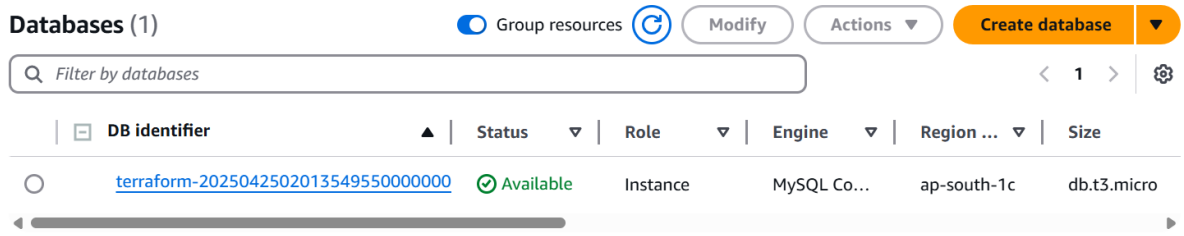
```
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds> terraform apply
data.aws_vpc.default: Reading...
data.aws_vpc.default: Read complete after 1s [id=vpc-077a2fb10879758de]

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

Terraform will perform the following actions:

  # aws_db_instance.my_rds will be created
  + resource "aws_db_instance" "my_rds" {
      + address                               = (known after apply)
      + allocated_storage                     = 20
      + apply_immediately                     = false
      + arn                                   = (known after apply)
      + auto_minor_version_upgrade            = true
      + availability_zone                     = (known after apply)
      + backup_retention_period               = (known after apply)
      + backup_target                         = (known after apply)
      + backup_window                         = (known after apply)
      + ca_cert_identifier                    = (known after apply)
      + character_set_name                    = (known after apply)
      + copy_tags_to_snapshot                 = false
```

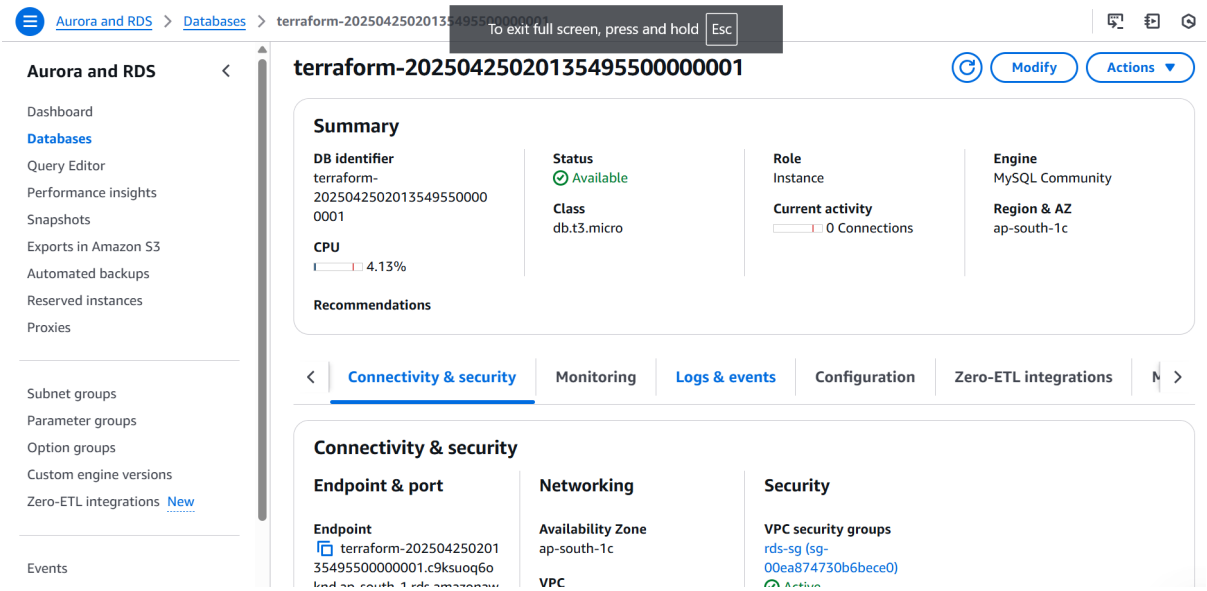
```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_security_group.rds_sg: Creating...
aws_security_group.rds_sg: Creation complete after 2s [id=sg-00ea874730b6bece0]
aws_db_instance.my_rds: Creating...
aws_db_instance.my_rds: Still creating... [10s elapsed]
aws_db_instance.my_rds: Still creating... [20s elapsed]
aws_db_instance.my_rds: Still creating... [30s elapsed]
aws_db_instance.my_rds: Still creating... [40s elapsed]
aws_db_instance.my_rds: Still creating... [50s elapsed]
aws_db_instance.my_rds: Still creating... [1m0s elapsed]
aws_db_instance.my_rds: Still creating... [1m10s elapsed]
aws_db_instance.my_rds: Still creating... [1m21s elapsed]
aws_db_instance.my_rds: Still creating... [1m31s elapsed]
aws_db_instance.my_rds: Still creating... [1m41s elapsed]
aws_db_instance.my_rds: Still creating... [1m51s elapsed]
aws_db_instance.my_rds: Still creating... [2m1s elapsed]
aws_db_instance.my_rds: Still creating... [2m11s elapsed]
aws_db_instance.my_rds: Still creating... [2m21s elapsed]
aws_db_instance.my_rds: Still creating... [2m31s elapsed]
aws_db_instance.my_rds: Still creating... [2m41s elapsed]
aws_db_instance.my_rds: Still creating... [2m51s elapsed]
aws_db_instance.my_rds: Still creating... [3m1s elapsed]
aws_db_instance.my_rds: Still creating... [3m11s elapsed]
aws_db_instance.my_rds: Still creating... [3m21s elapsed]
aws_db_instance.my_rds: Still creating... [3m31s elapsed]
aws_db_instance.my_rds: Still creating... [3m41s elapsed]
aws_db_instance.my_rds: Creation complete after 3m47s [id=db-52PU7AREEQVPLLIFTNKPNLR4VA]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds>
```

- Terraform will prompt you to confirm the creation of the RDS instance. Type yes and press Enter.

4. Verify RDS Instance in AWS Console:

- Log in to the AWS Management Console and navigate to the RDS service.
- Verify that the specified RDS instance with the specified settings has been created.





5. Update RDS Configuration:

- If you want to modify the RDS instance configuration, update the main.tf file with the desired changes.
- Rerun the **terraform apply** command to apply the changes.

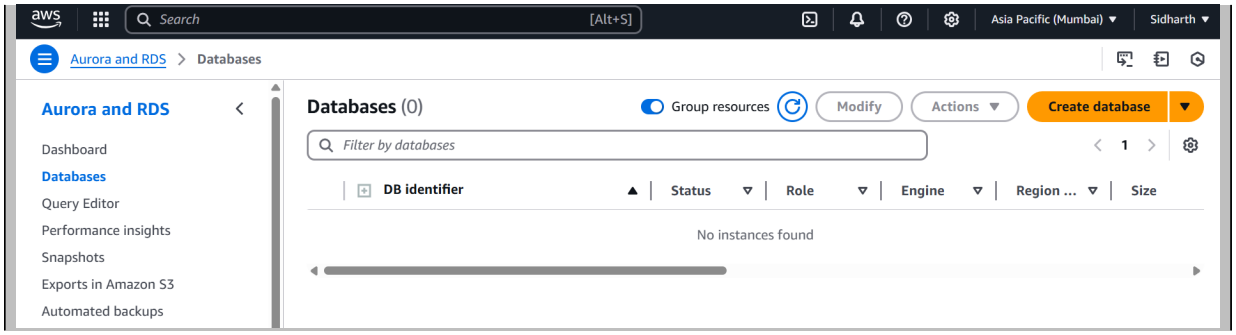
6. Clean Up

After testing, you can clean up the RDS instance:

terraform destroy

```
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds> terraform destroy
data.aws_vpc.default: Reading...
data.aws_vpc.default: Read complete after 1s [id=vpc-077a2fb10879758de]
aws_security_group.rds_sg: Refreshing state... [id=sg-00ea874730b6bece0]
aws_db_instance.my_rds: Refreshing state... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA]

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_db_instance.my_rds will be destroyed
  - resource "aws_db_instance" "my_rds" {
      - address                               = "terraform-20250425020135495500000001.c9ksuoq6oknd.ap-south-1.rds.amazonaws.com" -> n
ull
      - allocated_storage                     = 20 -> null
      - apply_immediately                     = false -> null
      - arn                                   = "arn:aws:rds:ap-south-1:976193261889:db:terraform-20250425020135495500000001" -> null
      - auto_minor_version_upgrade            = true -> null
      - availability_zone                     = "ap-south-1c" -> null
      - backup_retention_period               = 0 -> null
      - backup_target                         = "region" -> null
      - backup_window                         = "23:20-23:50" -> null
      - ca_cert_identifier                    = "rds-ca-rsa2048-g1" -> null
      - copy_tags_to_snapshot                 = false -> null
      - customer_owned_ip_enabled             = false -> null
      - db_name                               = "upesdb" -> null
```

```
Plan: 0 to add, 0 to change, 2 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_db_instance.my_rds: Destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 10s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 20s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 30s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 40s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 50s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m0s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m10s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m20s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m30s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m40s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 1m50s elapsed]
aws_db_instance.my_rds: Still destroying... [id=db-52PU7AREEQVPLLIFTNKPNLR4VA, 2m0s elapsed]
aws_db_instance.my_rds: Destruction complete after 2m2s
aws_security_group.rds_sg: Destroying... [id=sg-00ea874730b6bece0]
aws_security_group.rds_sg: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
PS G:\New Volume E\6th sem\System Provisioning Lab\terraform-rds>
```



Confirm the destruction by typing yes.

7. Conclusion:

This lab exercise demonstrates how to use Terraform to create an AWS RDS instance. You learned how to define RDS settings, initialize and apply the Terraform configuration, and verify the creation of the RDS instance in the AWS Management Console. Experiment with different RDS settings in the main.tf file to observe how