

Install mariadb to the Linux instance and create DB. That DB backup Store in S3 Service (AWS Role).

Step 1: Add Name and Tags

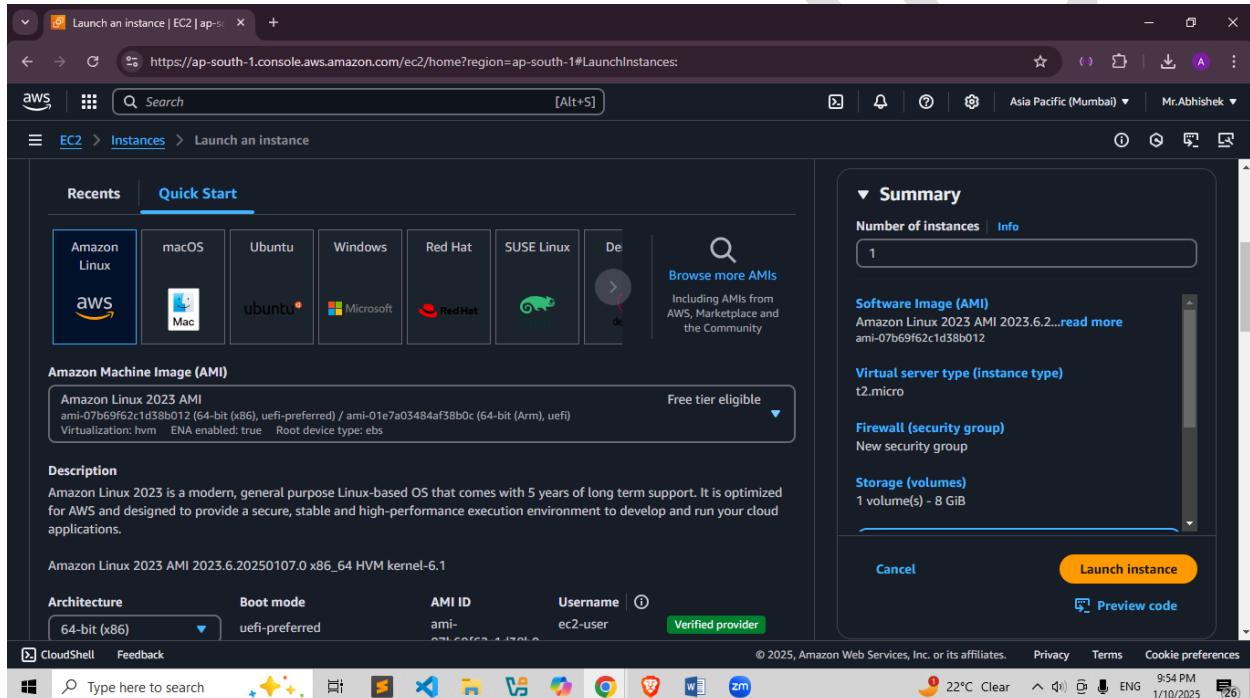
- Add tags to your instance for better management and identification.
 - Example: Key: Name, Value: MyInstance.

The screenshot shows the AWS EC2 Dashboard for the Asia Pacific (Mumbai) Region. The left sidebar includes sections for Dashboard, EC2 Global View, Events, Instances, Images, and Elastic Block Store. The main area displays a summary of resources: 0 instances running, 0 auto scaling groups, 0 capacity reservations, 0 dedicated hosts, 0 elastic IPs, 0 instances, 17 key pairs, 0 load balancers, 0 placement groups, 0 security groups, 0 snapshots, and 0 volumes. Below this is a 'Launch instance' button and a 'Service health' section. On the right, there's a 'EC2 Free Tier' summary with 2 offers in use, and a 'Offer usage (monthly)' section showing Linux EC2 Instances at 0% usage and 746.349167 hours remaining.

The screenshot shows the 'Launch an instance' wizard in progress. It's the first step, titled 'Name and tags'. A text input field contains 'database-S3'. To the right, there's a 'Summary' section showing 1 instance selected. The 'Software Image (AMI)' is set to Amazon Linux 2023 AMI 2023.6.2...read more. The 'Virtual server type (instance type)' is t2.micro. The 'Firewall (security group)' is set to New security group. Under 'Storage (volumes)', it shows 1 volume(s) - 8 GiB. At the bottom are 'Cancel', 'Launch instance', and 'Preview code' buttons.

Step 2: Choose an Amazon Machine Image (AMI)

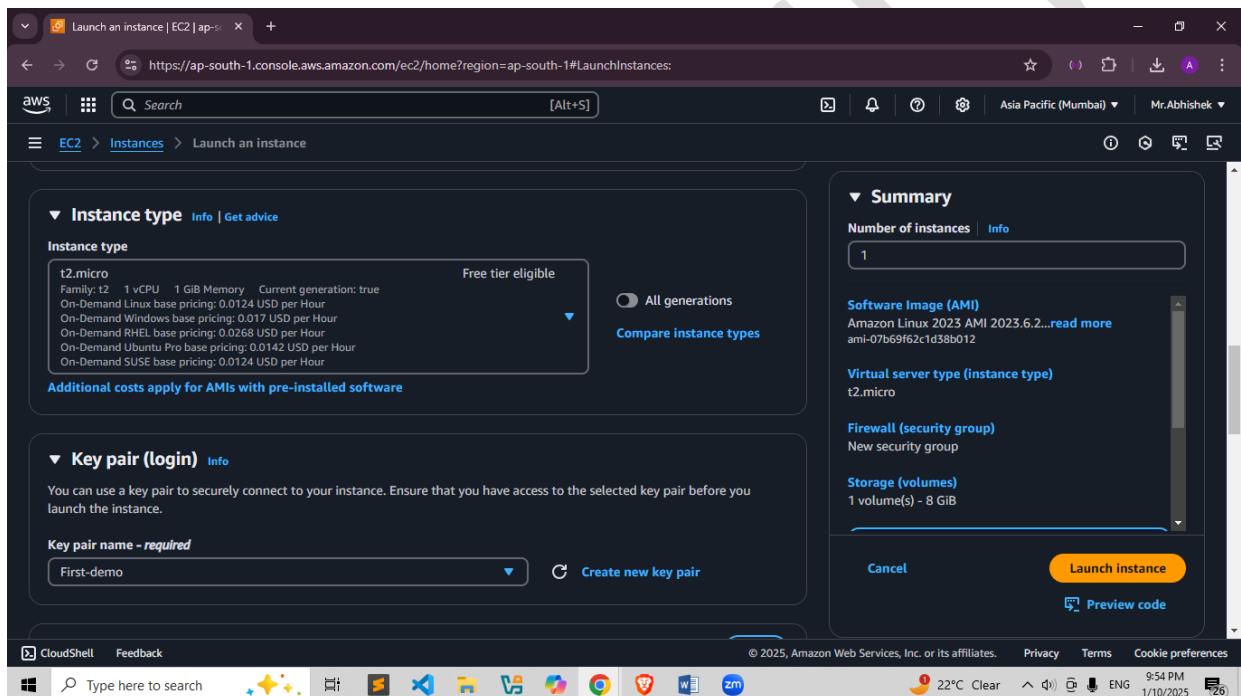
- Select an **AMI**, which is a pre-configured virtual machine template.
 - Examples:
 - **Amazon Linux 2 AMI** (Free-tier eligible).
 - **Ubuntu Server**.
 - **Windows Server** (if you need a Windows environment).
- Choose an AMI that suits your requirements for the operating system and software packages.



Step 3: Choose an Instance Type

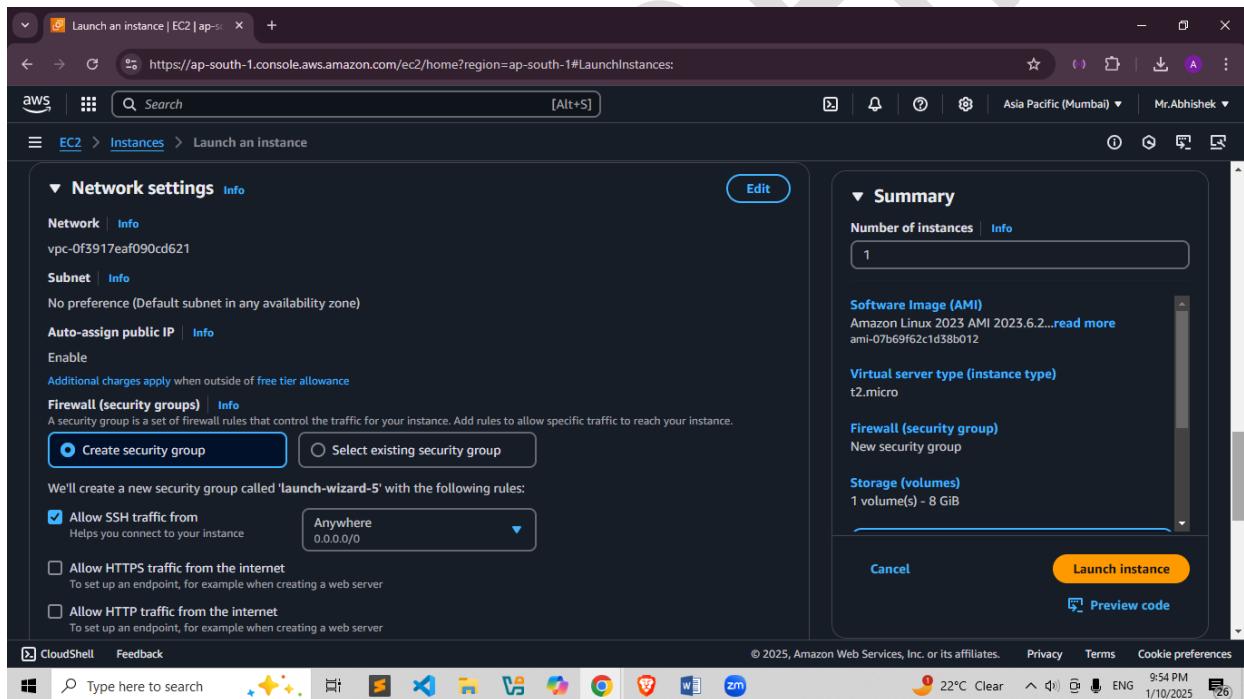
Select the instance type based on your performance needs (CPU, memory, storage, etc.).

- For free-tier eligible users, choose **t2.micro** or **t3.micro**.
- When prompted, create a new key pair or use an existing one for SSH access:
 - Download the private key file (.pem) if creating a new key pair.



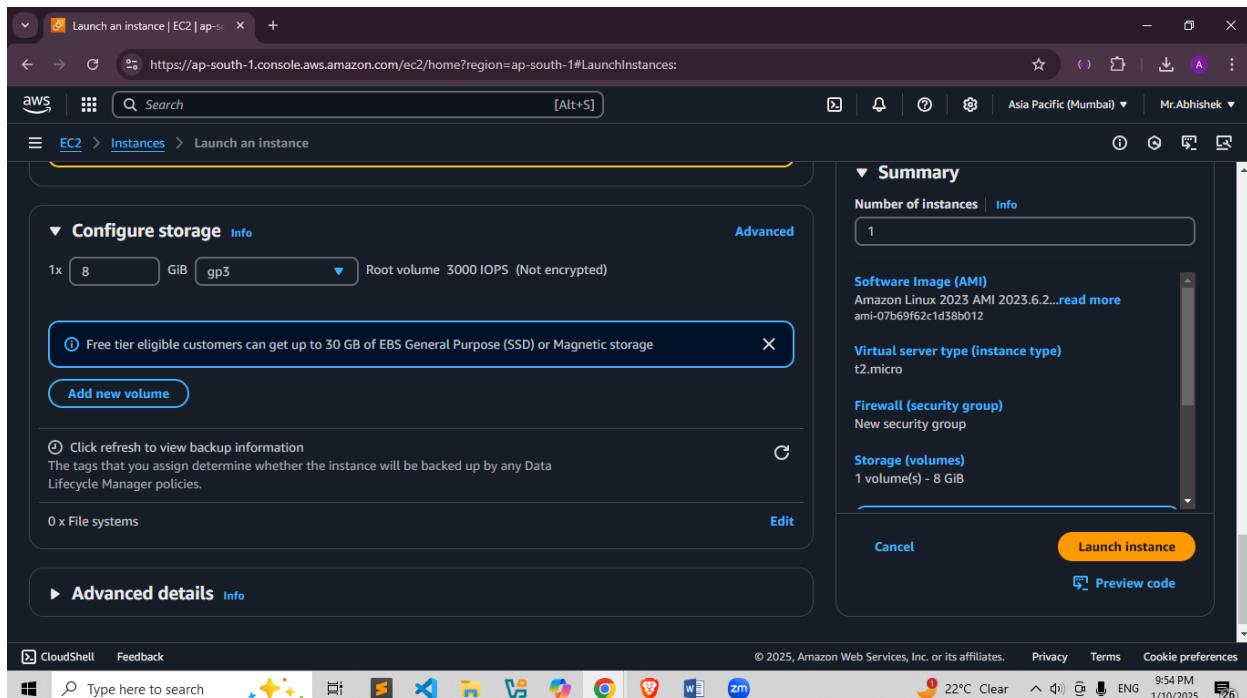
Step 4: Configure Instance Details

- Specify the details for your instance:
 - Number of instances:** Default is 1.
 - Network:** Select a Virtual Private Cloud (VPC).
 - Subnet:** Choose a subnet for your instance.
 - Auto-assign Public IP:** Enable this if you need internet access.
 - IAM Role:** Assign an IAM role if necessary.
 - Advanced options: Configure placement groups, capacity reservations, etc.
- Click **Next** when done.



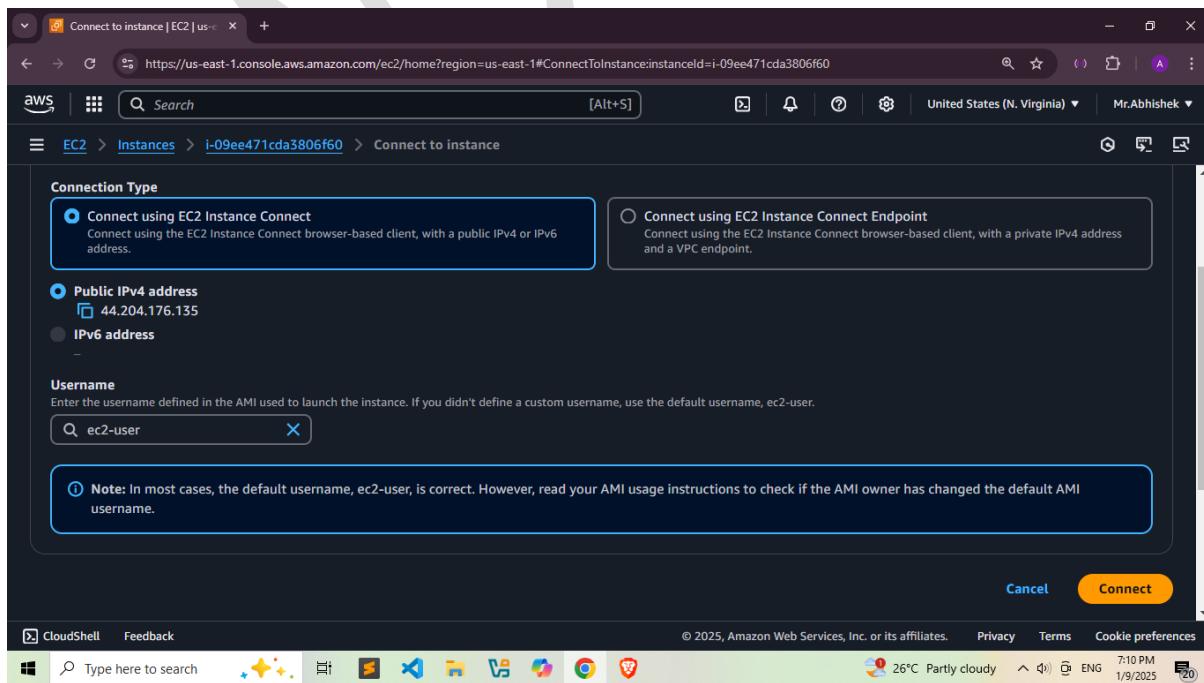
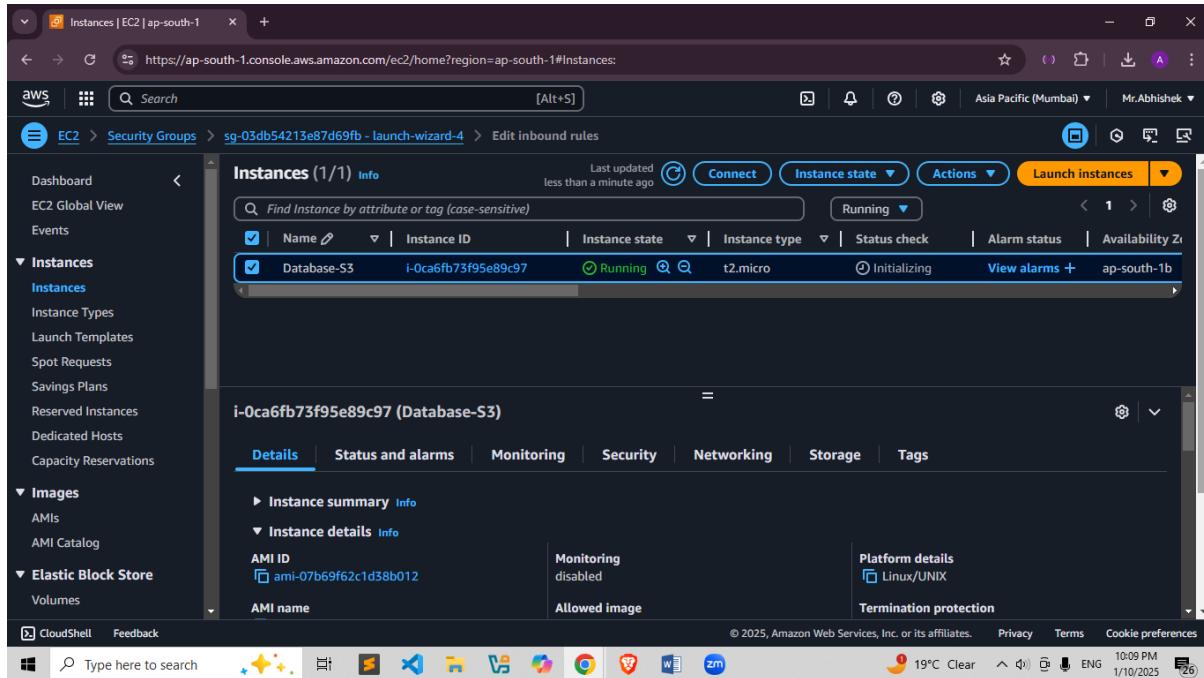
Step 5: Add Storage

- Configure the storage for your instance:
 - Root volume size (default is 8 GiB for Amazon Linux).
 - Add additional volumes if required.
 - Choose the storage type (e.g., General Purpose SSD).

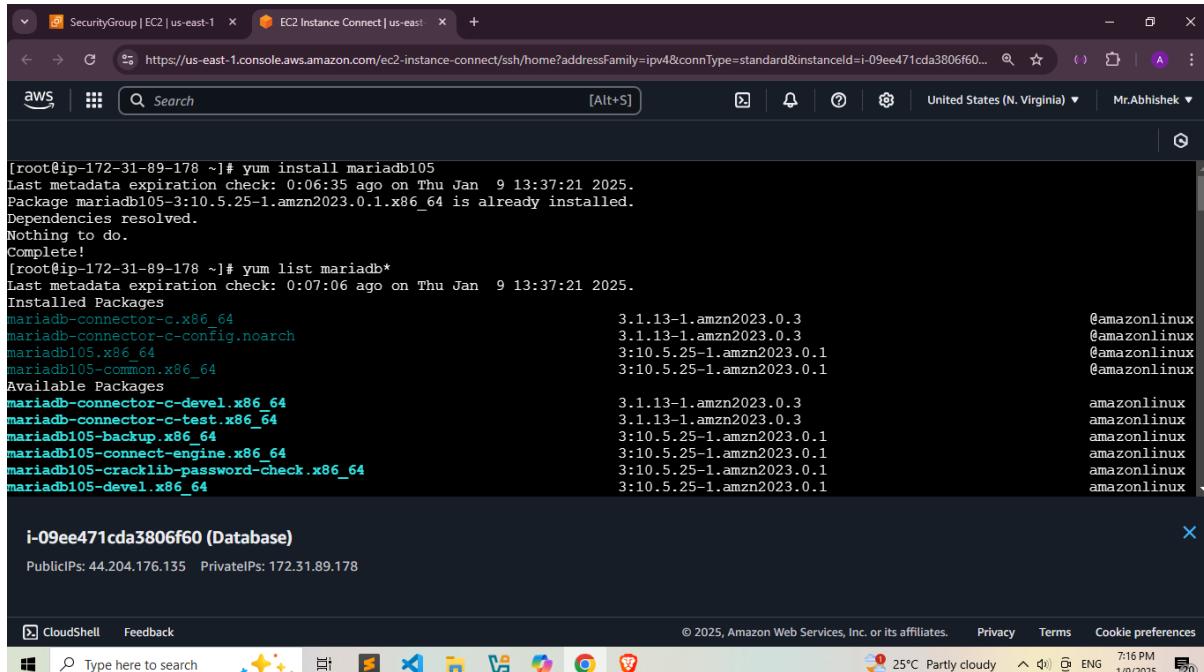


Step 6: Instance to Launch

- You will be redirected to a confirmation page.
- Click **View Instances** to go to the EC2 Dashboard and monitor the status of your instance.



Install MariaDB: Install MariaDB using the package manager. On redhat-based systems like centos, use:



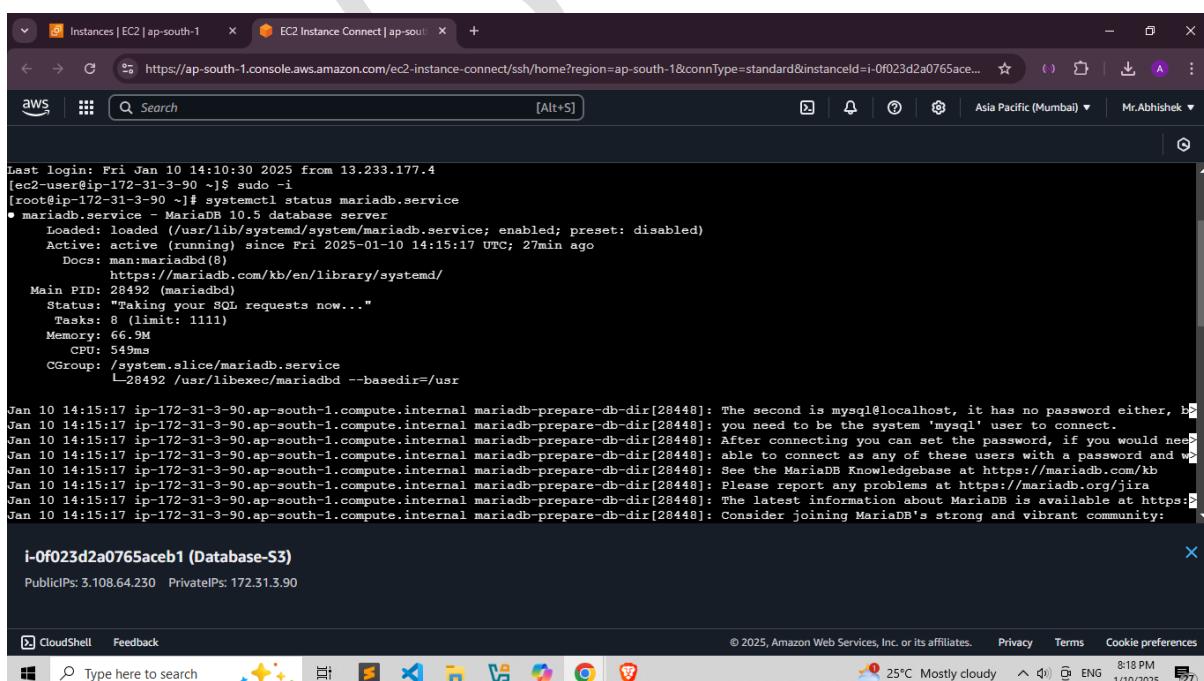
```
[root@ip-172-31-89-178 ~]# yum install mariadb105
Last metadata expiration check: 0:06:35 ago on Thu Jan  9 13:37:21 2025.
Package mariadb105-3:10.5.25-1.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-89-178 ~]# yum list mariadb*
Last metadata expiration check: 0:07:06 ago on Thu Jan  9 13:37:21 2025.
Installed Packages
mariadb-connector-c.x86_64          3.1.13-1.amzn2023.0.3      @amazonlinux
mariadb-connector-c-config.noarch   3.1.13-1.amzn2023.0.3      @amazonlinux
mariadb105.x86_64                  3:10.5.25-1.amzn2023.0.1    @amazonlinux
mariadb105-common.x86_64           3:10.5.25-1.amzn2023.0.1    @amazonlinux
Available Packages
mariadb-connector-c-devel.x86_64    3.1.13-1.amzn2023.0.3      amazonlinux
mariadb-connector-c-test.x86_64     3.1.13-1.amzn2023.0.3      amazonlinux
mariadb105-backup.x86_64          3:10.5.25-1.amzn2023.0.1    amazonlinux
mariadb105-connect-engine.x86_64   3:10.5.25-1.amzn2023.0.1    amazonlinux
mariadb105-cracklib-password-check.x86_64 3:10.5.25-1.amzn2023.0.1    amazonlinux
mariadb105-devel.x86_64          3:10.5.25-1.amzn2023.0.1    amazonlinux
```

i-09ee471cda3806f60 (Database)

Public IPs: 44.204.176.135 Private IPs: 172.31.89.178

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Partly cloudy 7:16 PM ENG 1/9/2025 (20)



```
Last login: Fri Jan 10 14:10:30 2025 from 13.233.177.4
[ec2-user@ip-172-31-3-90 ~]$ sudo -i
[root@ip-172-31-3-90 ~]# systemctl status mariadb.service
● mariadb.service - MariaDB 10.5 database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: disabled)
  Active: active (running) since Fri 2025-01-10 14:15:17 UTC; 27min ago
    Docs: man:mariadb(8)
          https://mariadb.com/kb/en/library/systemd/
 Main PID: 28492 (mariadb)
 Status: "Taking your SQL requests now..."
   Tasks: 8 (limit: 1111)
  Memory: 66.9M
    CPU: 549ms
   CGroup: /system.slice/mariadb.service
           └─28492 /usr/libexec/mariadb --basedir=/usr

Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: The second is mysql@localhost, it has no password either, b
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: you need to be the system 'mysql' user to connect.
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: After connecting you can set the password, if you would nee
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: able to connect as any of these users with a password and w
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: See the MariaDB Knowledgebase at https://mariadb.com/kb
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: Please report any problems at https://mariadb.org/jira
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: The latest information about MariaDB is available at https:
Jan 10 14:15:17 ip-172-31-3-90.ap-south-1.compute.internal mariadb-prepare-db-dir[28448]: Consider joining MariaDB's strong and vibrant community:
```

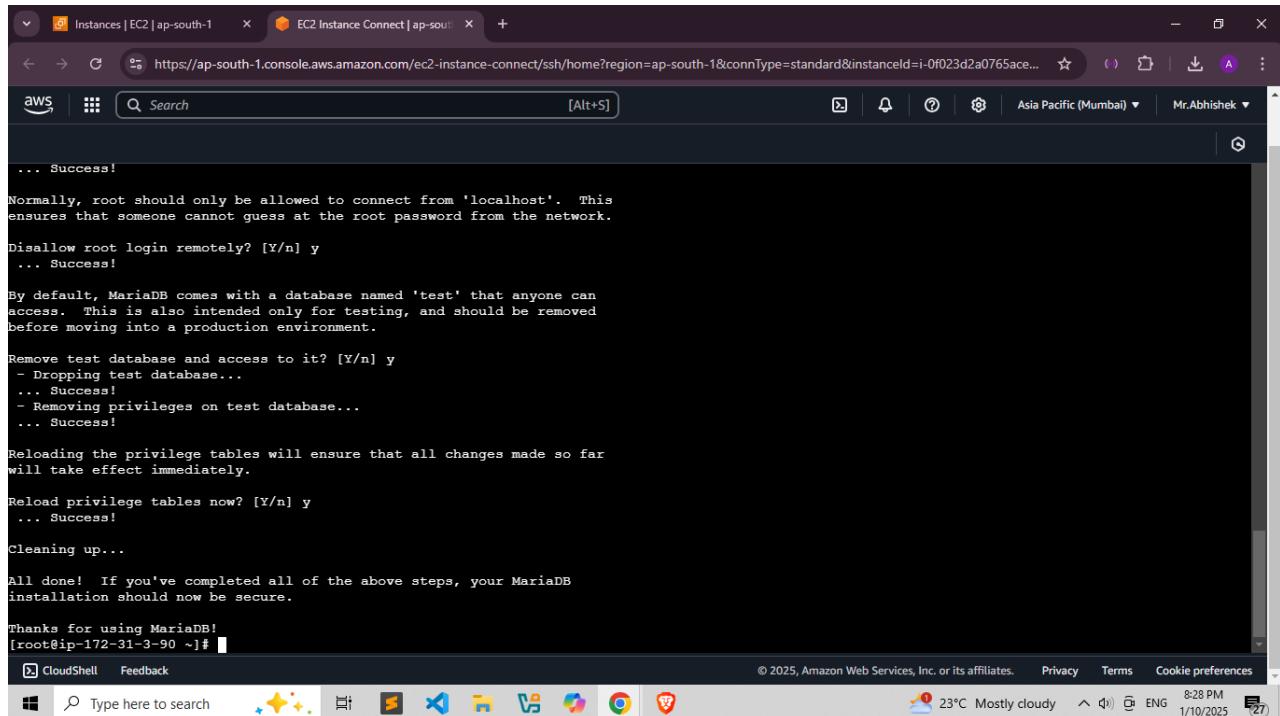
i-0f023d2a0765aceb1 (Database-S3)

Public IPs: 3.108.64.230 Private IPs: 172.31.3.90

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Mostly cloudy 8:18 PM ENG 1/10/2025 (27)

Secure MariaDB Installation: Run the security script to set up security options.



```
... Success!
Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

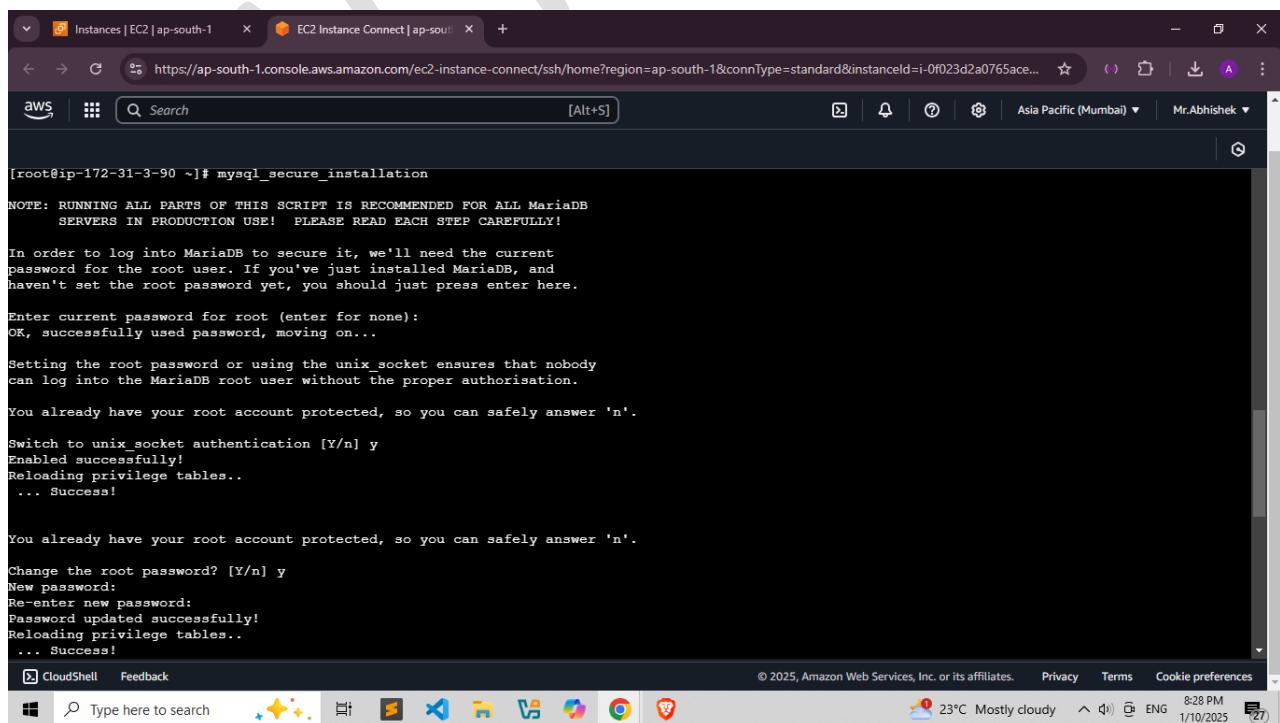
Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.

Thanks for using MariaDB!
[root@ip-172-31-3-90 ~]#
```

- **Change the Root Password:** You will be prompted to change the root password. Follow the prompts to set a new password:



```
[root@ip-172-31-3-90 ~]# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

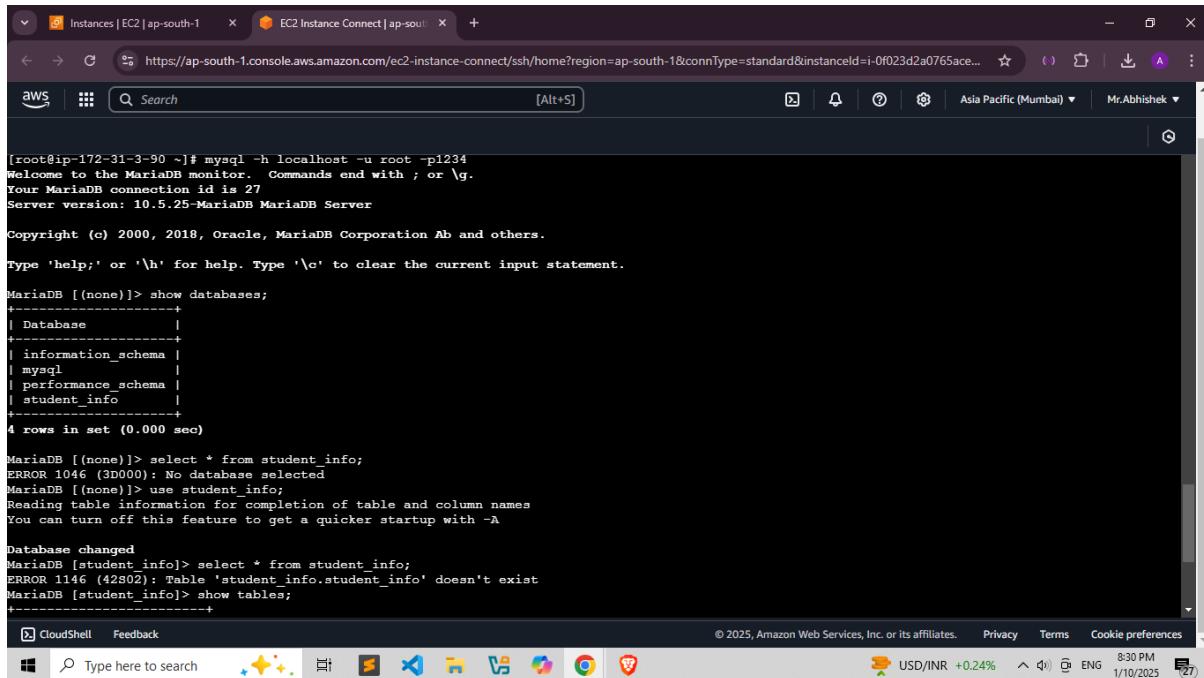
You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] y
Enabled successfully!
Reloading privilege tables..
... Success!

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!
```

- Switch to user **sudo mysql -u root -p1234**.
- Write **show databases;** command.



A screenshot of the AWS CloudShell interface. The terminal window shows a MySQL session connected to the 'student_info' database. The user runs 'show databases;' which lists databases like 'information_schema', 'mysql', 'performance_schema', and 'student_info'. Then, they attempt to select from 'student_info' but get an error because no database is selected. Finally, they run 'use student_info;' and then 'show tables;' to see the available tables ('Courses' and 'student_info'). The AWS logo is visible in the background.

```
[root@ip-172-31-3-90 ~]# mysql -h localhost -u root -p1234
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 27
Server version: 10.5.25-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

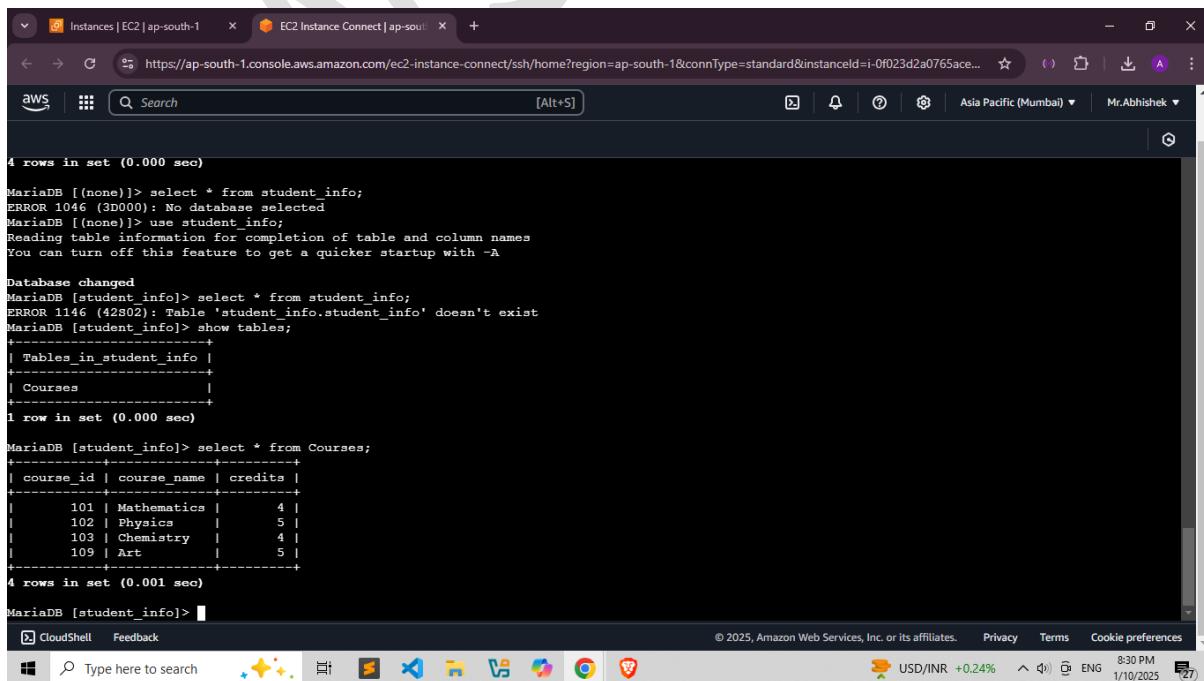
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| student_info |
+-----+
4 rows in set (0.000 sec)

MariaDB [(none)]> select * from student_info;
ERROR 1046 (3D000): No database selected
MariaDB [(none)]> use student_info;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [student_info]> select * from student_info;
ERROR 1146 (42802): Table 'student_info.student_info' doesn't exist
MariaDB [student_info]> show tables;
+-----+
| Tables_in_student_info |
+-----+
| Courses |
+-----+
1 row in set (0.000 sec)

MariaDB [student_info]> 
```

- To switch to a different database in MariaDB, you use the **(USE database_name;)**
- Create Table and Fields also using MySQL commands.



A screenshot of the AWS CloudShell interface. The terminal window shows a MySQL session connected to the 'student_info' database. The user runs 'show databases;' and 'show tables;' to verify the setup. They then create a new table 'Courses' with columns 'course_id', 'course_name', and 'credits'. The table is populated with five rows: (101, Mathematics, 4), (102, Physics, 5), (103, Chemistry, 4), (109, Art, 5). The AWS logo is visible in the background.

```
4 rows in set (0.000 sec)

MariaDB [(none)]> select * from student_info;
ERROR 1046 (3D000): No database selected
MariaDB [(none)]> use student_info;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [student_info]> select * from student_info;
ERROR 1146 (42802): Table 'student_info.student_info' doesn't exist
MariaDB [student_info]> show tables;
+-----+
| Tables_in_student_info |
+-----+
| Courses |
+-----+
1 row in set (0.000 sec)

MariaDB [student_info]> select * from Courses;
+-----+-----+-----+
| course_id | course_name | credits |
+-----+-----+-----+
| 101 | Mathematics | 4 |
| 102 | Physics | 5 |
| 103 | Chemistry | 4 |
| 109 | Art | 5 |
+-----+-----+-----+
4 rows in set (0.001 sec)

MariaDB [student_info]> 
```

Step 1: Create the Role

- Sign in to the AWS Management Console.**
- Navigate to the IAM Console:** In the navigation pane, choose "Roles" and then "Create role."

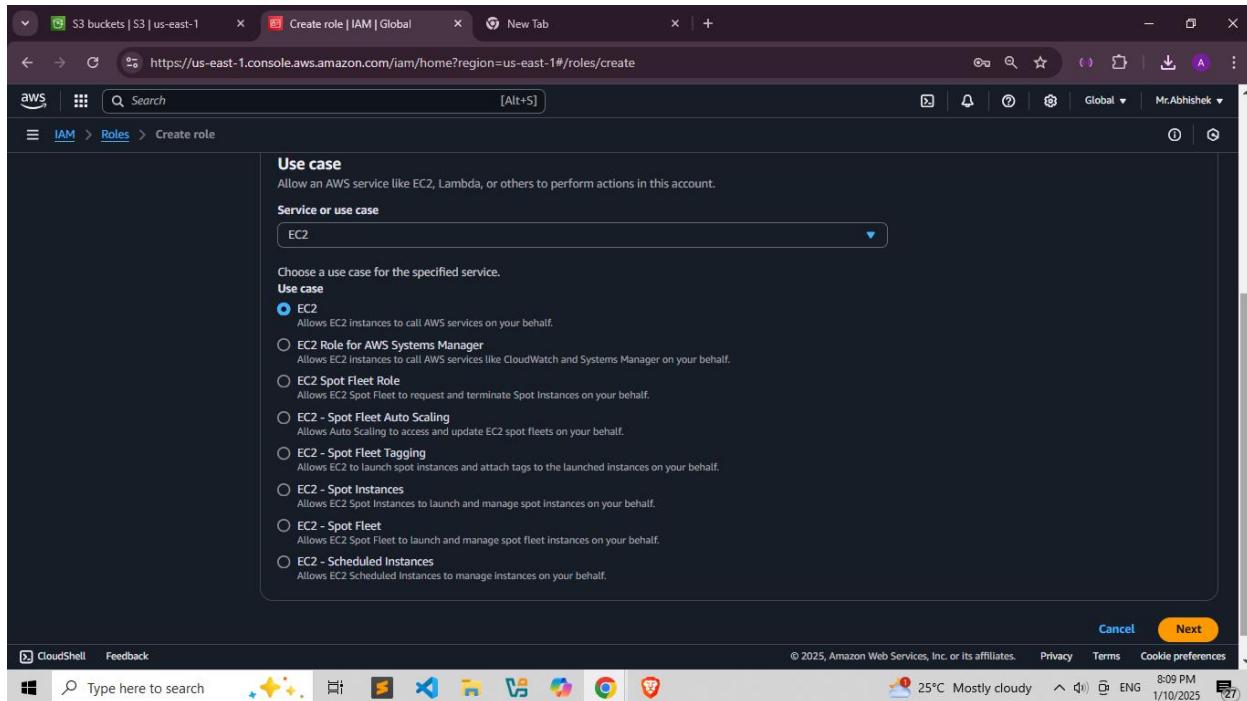
The screenshot shows the AWS IAM Roles page. On the left, there's a sidebar with "Identity and Access Management (IAM)" and a search bar. Under "Access management", "Roles" is selected. The main area displays a table titled "Roles (10) Info" with columns for "Role name", "Trusted entities", and "Last updated". The roles listed are: AWSServiceRoleForAutoScaling, AWSServiceRoleForCloudWatchEvents, AWSServiceRoleForElasticLoadBalancing, AWSServiceRoleForOrganizations, AWSServiceRoleForRDS, AWSServiceRoleForRoute53Resolver, and AWSServiceRoleForSupport. Each role has a link to its details and a "Delete" button.

Select the Trusted Entity:

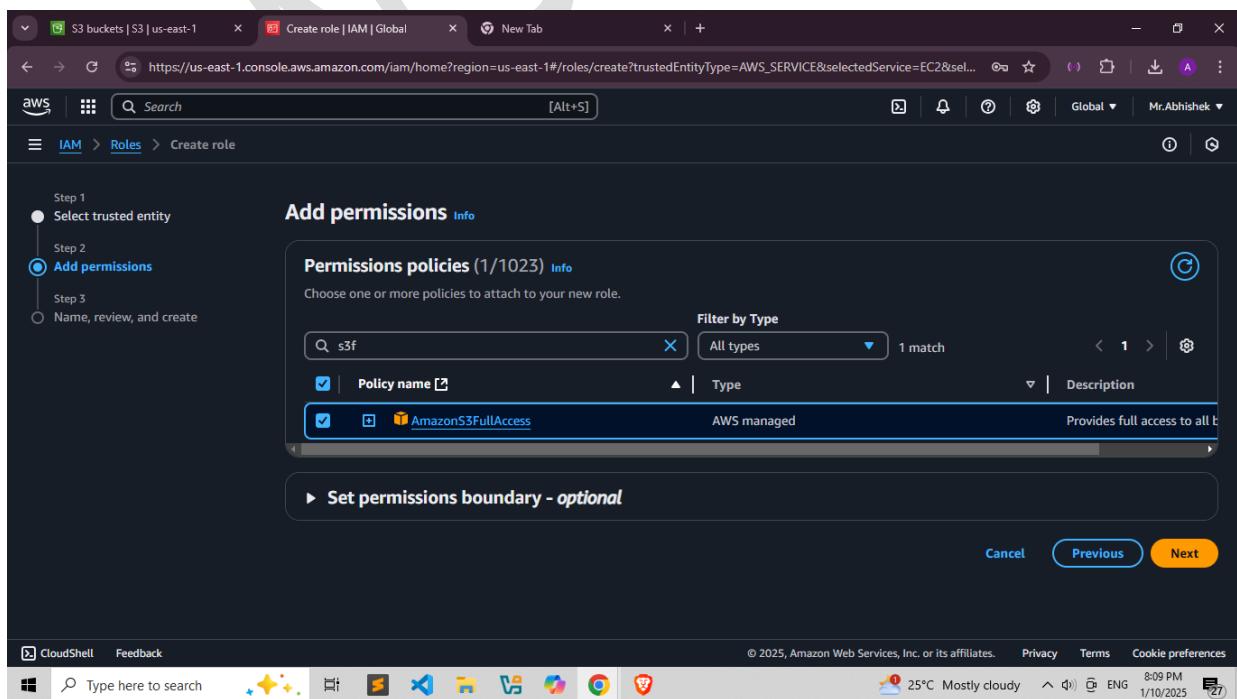
- Choose "AWS service."

The screenshot shows the "Select trusted entity" step in the IAM Role creation wizard. On the left, a sidebar lists "Step 1 Select trusted entity", "Step 2 Add permissions", and "Step 3 Name, review, and create". The main area is titled "Select trusted entity Info" and contains a "Trusted entity type" section. It shows five options: "AWS service" (selected), "AWS account", "Web identity", "SAML 2.0 federation", and "Custom trust policy". Each option has a brief description. The "AWS service" option is highlighted with a blue border.

- Select "EC2" as the service that will use this role.



- Click "Next: Permissions."



Add Tags (Optional):

- You can add tags to your role to help you organize and manage your resources.
- Click "Next: Review."
- Click On Create Role Button.

Add Tags (Optional):

The screenshot shows the 'Create role' wizard in the AWS IAM console, specifically Step 3: Add tags. The 'Add tags - optional' section is displayed, showing a note that says 'No tags associated with the resource.' and a button to 'Add new tag'. Navigation buttons 'Cancel', 'Previous', and 'Create role' are at the bottom.

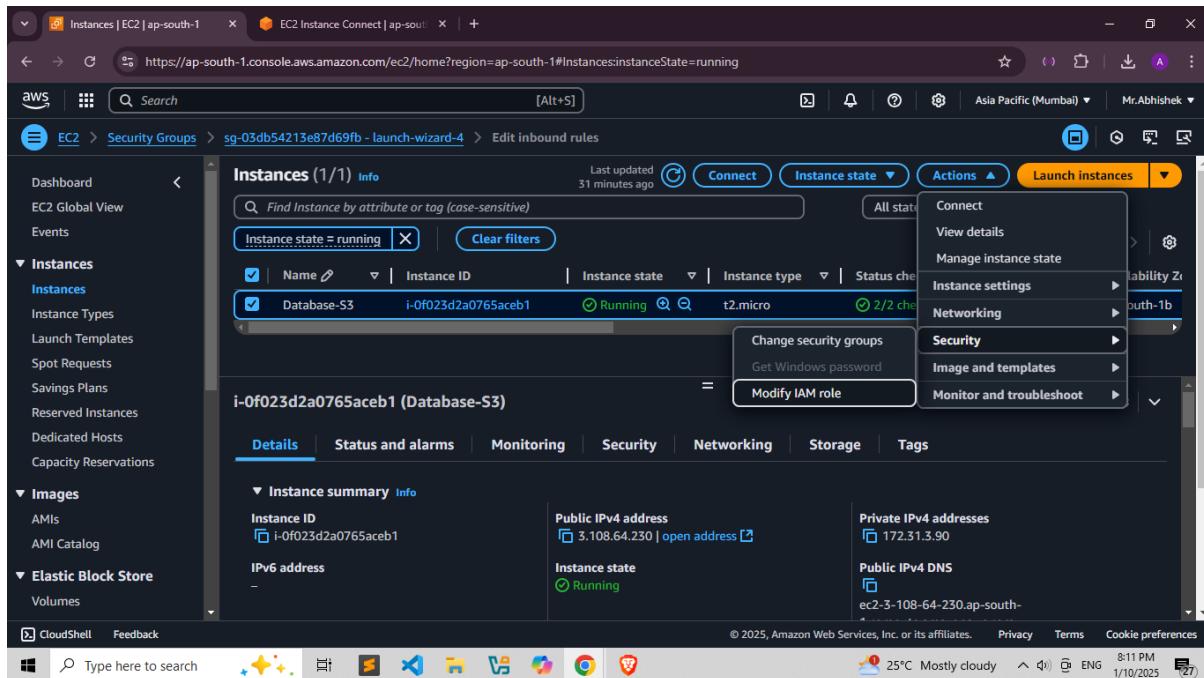
The screenshot shows the AWS IAM console with the URL <https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/RDS-role?section=permissions>. The left sidebar is titled "Identity and Access Management (IAM)" and includes sections for Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings, Root access management), Access reports (Access Analyzer, External access, Unused access, Analyzer settings, Credential report), CloudShell, and Feedback. The main content area is titled "RDS-role Info" and describes the role as allowing EC2 instances to call AWS services on your behalf. It shows the "Summary" tab with details like Creation date (January 10, 2025, 20:10 UTC+05:30), Last activity (None), ARN (arn:aws:iam::637423624381:role/RDS-role), and Maximum session duration (1 hour). An "Instance profile ARN" is also listed. Below the summary is a "Permissions" tab, which displays one attached policy: "AmazonS3FullAccess" (AWS managed). The policy is listed in a table with columns for Policy name, Type, and Attached entities.

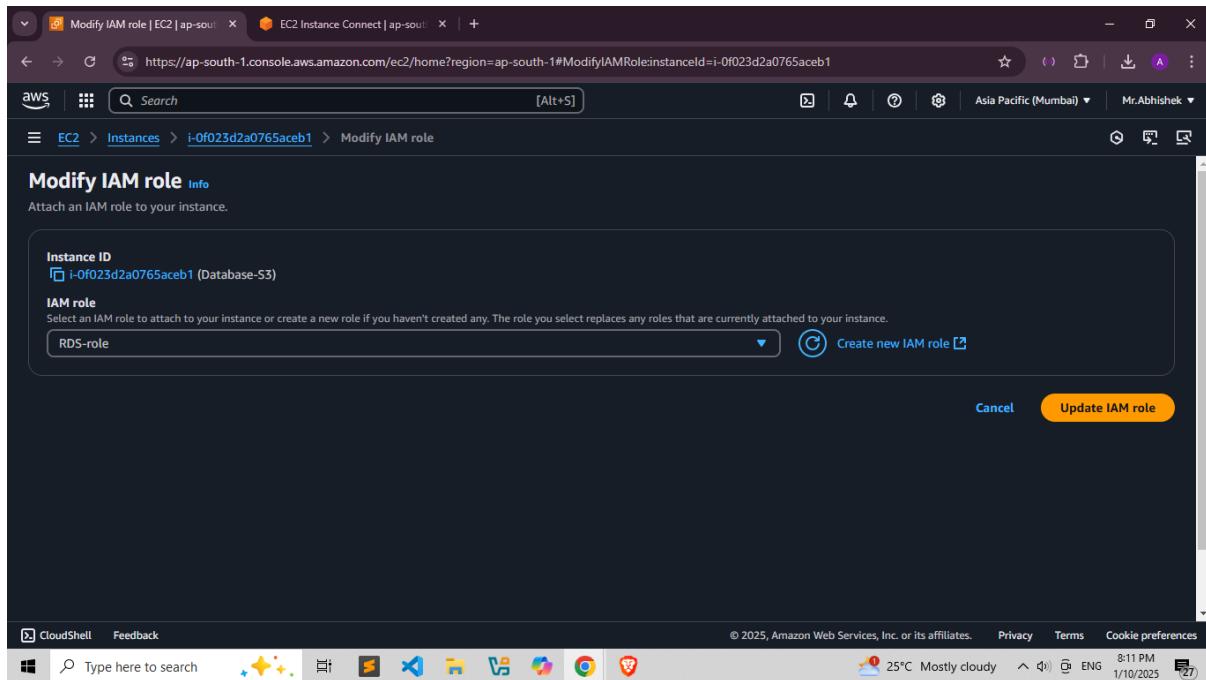
Policy name	Type	Attached entities
AmazonS3FullAccess	AWS managed	2

Step 3: Attach the Role to an EC2 Instance

Attach IAM Role:

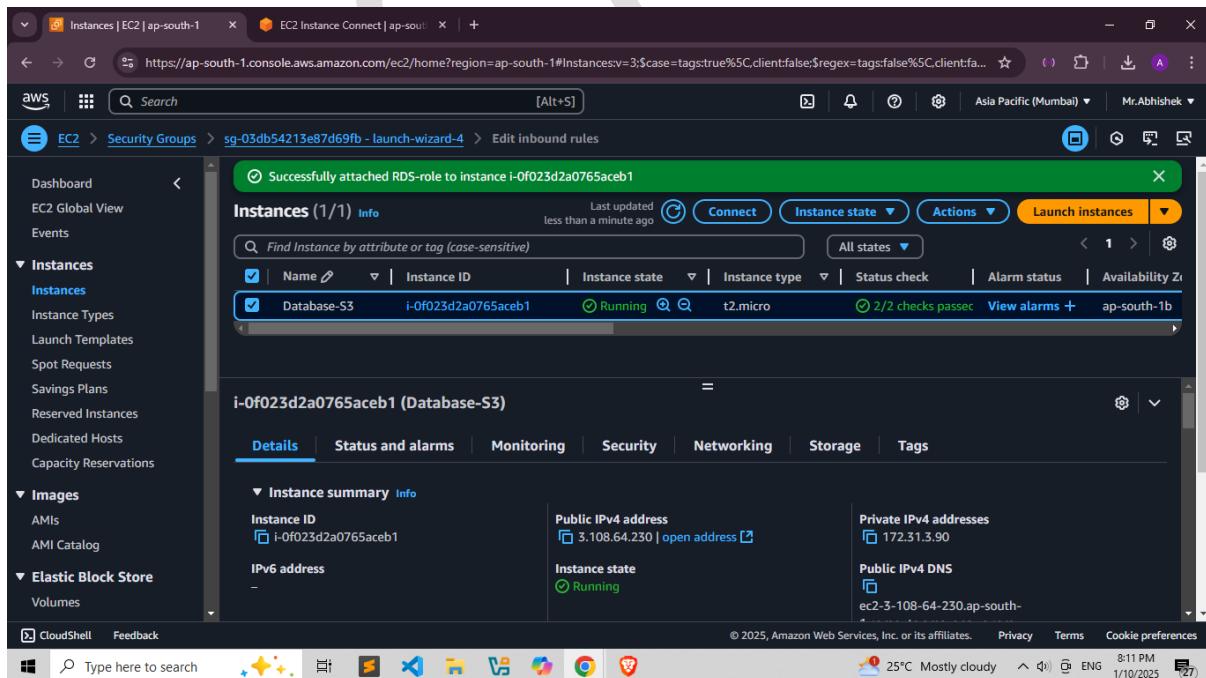
1. Choose "Actions" > "Security" > "Modify IAM Role."
2. In the "IAM role" drop-down menu, select the role you created (EC2_S3_FullAccess_Role).
3. Click "Update IAM role."





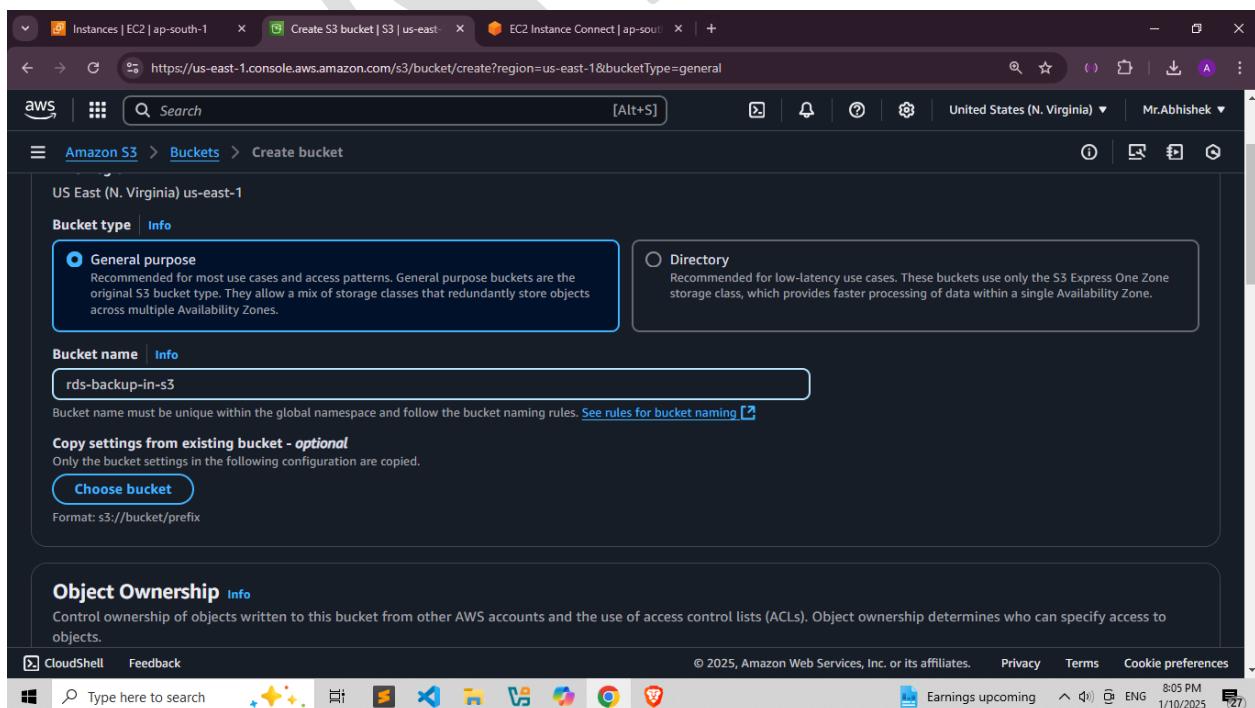
Step 4: Verify Role Attachment

- Verify:** Make sure that the EC2 instance now has the appropriate role attached by checking the "IAM Role" field in the instance details.



Step 1: Create a Bucket

1. Click on "Create bucket".
2. Configure the Bucket:
 - o **Bucket name:** Enter a unique name for your bucket. Bucket names must be globally unique and must comply with certain naming conventions.
 - o **Region:** Choose the AWS region where you want the bucket to be created.



Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudShell Feedback Type here to search 8:04 PM ENG 1/10/2025

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Disable

Enable

Tags - optional (0)

You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

No tags associated with this bucket.

[Add tag](#)

Default encryption Info

Server-side encryption is automatically applied to new objects stored in this bucket.

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudShell Feedback Type here to search 8:04 PM ENG 1/10/2025

Encryption type [Info](#)

- Server-side encryption with Amazon S3 managed keys (SSE-S3)
- Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the Storage tab of the [Amazon S3 pricing page](#).

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

- Disable
- Enable

Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)

Account snapshot - updated every 24 hours [All AWS Regions](#) [View Storage Lens dashboard](#)

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

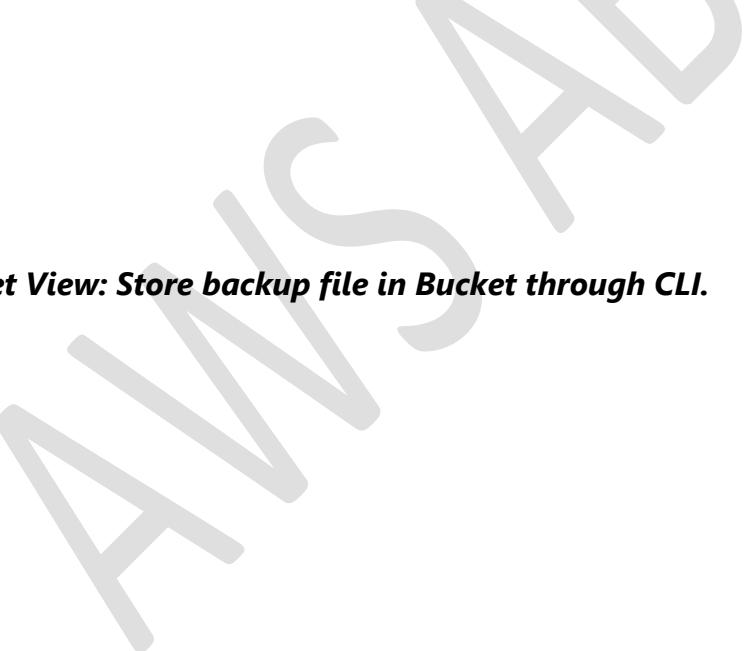
General purpose buckets [Directory buckets](#)

Name	AWS Region	IAM Access Analyzer	Creation date
rds-backup-in-s3	US East (N. Virginia) us-east-1	View analyzer for us-east-1	January 10, 2025, 20:05:12 (UTC+05:30)

[Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

Copy this backup file to the S3 Bucket using:

➤ **aws s3 cp filename.sql s3://bucket_name**



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| student_info |
+-----+
4 rows in set (0.000 sec)

MariaDB [(none)]> exit
Bye
[ec2-user@ip-172-31-3-90 ~]$ mysqldump -u root -p1234 student_info > backupdata.sql
[ec2-user@ip-172-31-3-90 ~]$ ls
backupdata.sql
[ec2-user@ip-172-31-3-90 ~]$ aws s3 ls
2025-01-10 14:35:13 rds-backup-in-s3
[ec2-user@ip-172-31-3-90 ~]$ aws s3 cp backupdata.sql s3://rds-backup-in-s3
upload: ./backupdata.sql to s3://rds-backup-in-s3/backupdata.sql
[ec2-user@ip-172-31-3-90 ~]$ aws s3 ls
[ec2-user@ip-172-31-3-90 ~]$ aws s3 ls

i-0f023d2a0765aceb1 (Database-S3)
PublicIPs: 3.108.64.230 PrivateIPs: 172.31.3.90

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Type here to search 25°C Mostly cloudy 8:18 PM 1/10/2025 [27]
```

Bucket View: Store backup file in Bucket through CLI.

The screenshot shows the AWS S3 console interface. The URL in the address bar is <https://us-east-1.console.aws.amazon.com/s3/buckets/rds-backup-in-s3?region=us-east-1&bucketType=general&tab=objects>. The page title is "rds-backup-in-s3 Info". The "Objects" tab is selected. There is one object listed:

Name	Type	Last modified	Size	Storage class
backupdata.sql	sql	January 10, 2025, 20:18:09 (UTC+05:30)	2.0 KB	Standard

Below the table, there is a note: "Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)".