

1. Search Aurora and Create MySQL database

The screenshot shows the 'Create database' wizard in the AWS RDS console. At the top, there's a message about a free plan having limited features. Below it, the 'Choose a database creation method' section has two options: 'Full configuration' (which is unselected) and 'Easy create' (which is selected). The 'Easy create' section includes a note that some configuration options can be changed after creation. The 'Configuration' section shows various database engines: Aurora (MySQL Compatible), Aurora (PostgreSQL Compatible), MySQL (selected), PostgreSQL, MariaDB, Oracle (ORACLE logo), Microsoft SQL Server (Microsoft SQL Server logo), and IBM Db2. The MySQL engine is highlighted with a blue border.

2. Select subnet , give public Access and choose VPC security group and create database

The screenshot continues the 'Create database' wizard. It shows the 'DB subnet group' step, where 'Default VPC (vpc-08e1d1e87473288a0)' is selected. A note says that after creation, the VPC cannot be changed. The 'Public access' step follows, with 'Yes' selected. It explains that RDS assigns a public IP address for connectivity from outside the VPC. The 'VPC security group (firewall)' step is shown at the bottom, with 'Choose existing' selected and a note about choosing appropriate security groups.

3. Database is created:

The screenshot shows the AWS RDS (Relational Database Service) console. At the top, there's a blue header bar with the text "Creating database hasrat-db" and a note: "Your database might take a few minutes to launch. You can use settings from hasrat-db to simplify configuration of suggested database add-ons while we finish creating your DB for you." Below this, there's a "View connection details" button and an "X" button. The main area is titled "Databases (1)" with a "Create database" button. A search bar labeled "Filter by databases" is present. The database list table has columns: DB identifier, Status, Role, Engine, Upgrade rollout order, Region ..., and Size. One row is shown: "hasrat-db" (Status: Creating, Role: Instance, Engine: MySQL Co..., Upgrade rollout order: SECOND, Region: -, Size: db.t4g.micro).

✚ Connecting EC2 with Database

4. Create EC2 instance and connect

The screenshot shows the AWS EC2 (Elastic Compute Cloud) instance summary page for an instance named "hasrat-web-forDB". The instance is currently running. The summary table includes the following details:

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0ef02e0f8a3e9aa5b	3.108.53.96 open address	172.31.3.15
IPv6 address	Instance state	Public DNS
-	Running	ec2-3-108-53-96.ap-south-1.compute.amazonaws.com open address
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-172-31-3-15.ap-south-1.compute.internal	ip-172-31-3-15.ap-south-1.compute.internal	-
Answer private resource DNS name	Instance type	AWS Compute Optimizer finding
IPv4 (A)	t3.micro	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address	VPC ID	Auto Scaling Group name
3.108.53.96 [Public IP]	vpc-08e1d1e87473288a0	-
IAM Role	Subnet ID	Managed
-	subnet-0fd08baaf710cf896	false
IMDSv2	Instance ARN	
Required	arn:aws:ec2:ap-south-1:132141656530:instance/i-0ef02e0f8a3e9aa5b	

5. Install Docker in the Instance

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

sc2-user@ip-172-31-3-15 ~]$ sudo yum install docker -y
amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.

=====
Package           Architecture Version      Repository  Size
=====
installing:
docker            x86_64      25.0.13-1.amzn2023.0.2   amazonlinux  46 M
installing dependencies:
container-selinux  noarch     4:2.242.0-1.amzn2023.0.2   amazonlinux  58 k
containerd         x86_64      2.1.5-1.amzn2023.0.1    amazonlinux  23 M
iptables-libs     x86_64      1.8.8-3.amzn2023.0.2    amazonlinux  401 k
iptables-nft      x86_64      1.8.8-3.amzn2023.0.2    amazonlinux  183 k
libcgroup          x86_64      3.0-1.amzn2023.0.1    amazonlinux  75 k
libnftfilter_conntrack x86_64  1.0.8-2.amzn2023.0.2    amazonlinux  58 k
libnftnetlink     x86_64      1.0.1-19.amzn2023.0.2   amazonlinux  30 k
libnftnl          x86_64      1.2.2-2.amzn2023.0.2   amazonlinux  84 k
pigz              x86_64      2.5-1.amzn2023.0.3    amazonlinux  83 k
runc              x86_64      1.3.3-2.amzn2023.0.1   amazonlinux  3.9 M

=====
268 kB/s | 29 kB  00:00
```

6. Starting docker and checking status

```
Complete!
[ec2-user@ip-172-31-3-15 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-3-15 ~]$ sudo systemctl docker status
Unknown command verb docker.
[ec2-user@ip-172-31-3-15 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
  Active: active (running) since Thu 2025-12-18 16:06:04 UTC; 33s ago
    TriggeredBy: • docker.socket
    Docs: https://docs.docker.com
   Process: 27286 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
   Process: 27287 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
   Main PID: 27288 (dockerd)
     Tasks: 9
       Memory: 30.3M
          CPU: 343ms
        CGroup: /system.slice/docker.service
                  └─dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Dec 18 16:06:03 ip-172-31-3-15.ap-south-1.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
Dec 18 16:06:03 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:03.835222920Z" level=info msg="Starting up"
Dec 18 16:06:03 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:03.898887461Z" level=info msg="Loading containers: start."
Dec 18 16:06:04 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:04.387985060Z" level=info msg="Loading containers: done."
Dec 18 16:06:04 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:04.416964656Z" level=info msg="Docker daemon" commit=165516e containerd=8
Dec 18 16:06:04 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:04.417145066Z" level=info msg="Daemon has completed initialization"
Dec 18 16:06:04 ip-172-31-3-15.ap-south-1.compute.internal dockerd[27288]: time="2025-12-18T16:06:04.465804775Z" level=info msg="API listen on /run/docker.sock"
Dec 18 16:06:04 ip-172-31-3-15.ap-south-1.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-22/22 (END)
```

7. Pulling Docker Image named : philippaul/node-mysql-app:02 (simple web app for database)

```
[ec2-user@ip-172-31-3-15 ~]$ sudo docker pull philippaul/node-mysql-app:02
02: Pulling from philippaul/node-mysql-app
2ff1d7c41c74: Pull complete
o253aeafeaa7: Pull complete
3d2201bd995c: Pull complete
1de76e268b10: Pull complete
d9a8df589451: Pull complete
6f51ee005dea: Pull complete
5f32ed3c3f27: Pull complete
0c8cc2f24a4d: Pull complete
0d27a8e86132: Pull complete
o35ca9a95db0: Pull complete
46a182df3db1: Pull complete
f5b1a7ebbe97: Pull complete
ff7978b844b1: Pull complete
Digest: sha256:f7c1cffb42a2f4a40b626b0d03f8b83bbc8ef3f88d0682cd43f395bf9e42966b
Status: Downloaded newer image for philippaul/node-mysql-app:02
docker.io/philippaul/node-mysql-app:02
[ec2-user@ip-172-31-3-15 ~]$ sudo docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
philippaul/node-mysql-app  02      4b941beb4207  13 months ago  923MB
[ec2-user@ip-172-31-3-15 ~]$
```

- Copy endpoint of Database we created in AWS for using as DB_HOST for connecting docker image with database

Aurora and RDS > Databases > hasrat-db

hasrat-db

Summary

DB identifier: hasrat-db

Status: Available

Role: Instance

Engine: MySQL Community

Region & AZ: ap-south-1a

Connectivity & security

Endpoint: hasrat-db.cpkauo000ugi.ap-south-1.rds.amazonaws.com

Port: 3306

Networking

Availability Zone: ap-south-1a

VPC: vpc-08e1d1e87473288a0

Subnet group: default-vpc-08e1d1e87473288a0

Subnets:

Security

VPC security groups: default (sg-00cbafab4c9630f66)

Publicly accessible: Yes

Certificate authority: rds-ca-rsa2048-g1

- Connect the docker image and Database with the following command

```
[ec2-user@ip-172-31-3-15 ~]$ sudo docker run --rm -p 80:3000 -e DB_HOST="hasrat-db.cpkauo000ugi.ap-south-1.rds.amazonaws.com" -e DB_USER="hasrat-db" -e DB_PASSWORD="hasrat123" -d philippaul/node-mysql-app:02
[ec2-user@ip-172-31-3-15 ~]$
```

- Checking status if the web server and database both are running properly

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Thu Dec 18 17:55:57 2025 from 13.233.177.4
[ec2-user@ip-172-31-10-235 ~]$ sudo docker run --rm -p 80:3000 -e DB_HOST="hasrat-db.cpkauo000ugi.ap-south-1.rds.amazonaws.com" -e DB_USER="admin" -e DB_PASSWORD="hasrat123" -d philippaul/node-mysql-app:02
[ec2-user@ip-172-31-10-235 ~]$ sudo docker logs -f interesting_almeida
Server is running on http://localhost:3000
Database "my_app_db" is ready.
Using database "my_app_db"
Table "contacts" is ready.
[ec2-user@ip-172-31-10-235 ~]$
```

11. Now copy instance public IP address and paste it on Browser URL to check if the Web app is running

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'EC2' selected, followed by 'Dashboard', 'AWS Global View', 'Events', 'Instances' (selected), 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', 'Capacity Manager', 'Images' (AMIs, AMI Catalog), and 'Elastic Block Store' (Volumes, Snapshots). The main area is titled 'Instance summary for i-06ccc0de09bb8d86f' (Updated less than a minute ago). It displays the following details:

- Instance ID:** i-06ccc0de09bb8d86f
- IPv6 address:** -
- Hostname type:** IP name: ip-172-31-10-235.ap-south-1.compute.internal
- Answer private resource DNS name:** IPv4 (A)
- Auto-assigned IP address:** 13.203.209.183 [Public IP]
- IAM Role:** -
- IMDSv2:** Required
- Instance state:** Running
- VPC ID:** vpc-08e1d1e87473288a0
- Subnet ID:** subnet-0fd08baaf710cf896
- Instance ARN:** arn:aws:ec2:ap-south-1:132141656530:instance/i-06ccc0de09bb8d86f
- Private IP DNS name (IPv4 only):** ip-172-31-10-235.ap-south-1.compute.internal
- Instance type:** t3.micro
- Elastic IP addresses:** -
- Public DNS:** ec2-13-203-209-183.ap-south-1.compute.amazonaws.com
- AWS Compute Optimizer finding:** Opt-In to AWS Compute Optimizer for recommendations. | Learn more
- Auto Scaling Group name:** -
- Managed:** false

At the top right, there are buttons for 'Connect', 'Instance state', and 'Actions'. A tooltip 'Public IPv4 address copied' points to the Public IP address field.

12. Add username in the web app

The screenshots show a web application titled 'Contact App' running in a browser. The URL in the address bar is '13.203.209.183'. The application interface is simple, featuring a central box with the title 'Contact App'.

Screenshot 1 (Top): The input field contains 'hasrathr123@gmail.com'. Below it are two orange buttons: 'Add Username' and 'Show All Contacts'. The 'Add Username' button is highlighted with a pink background.

Screenshot 2 (Bottom): The input field still contains 'hasrathr123@gmail.com'. Below the buttons, a new contact entry is displayed:
hasrathr123@gmail.com
Delete

13. Open mysql to check if the data (User name) is stored in the RDS:

```
[ec2-user@ip-172-31-10-235 ~]$ sudo docker run -it --rm mysql:8.0 mysql -h hasrat-db.cpkauo000ugi.ap-south-1.rds.amazonaws.com -u admin -p
0.0.0: Pulling from library/mysql
...
Digest: sha256:0275a35e79c60cace68fac520602d9f6897feb9b0941a1471196b1a01760e581
Status: Downloaded newer image for mysql:8.0
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 65
Server version: 8.0.43 Source distribution

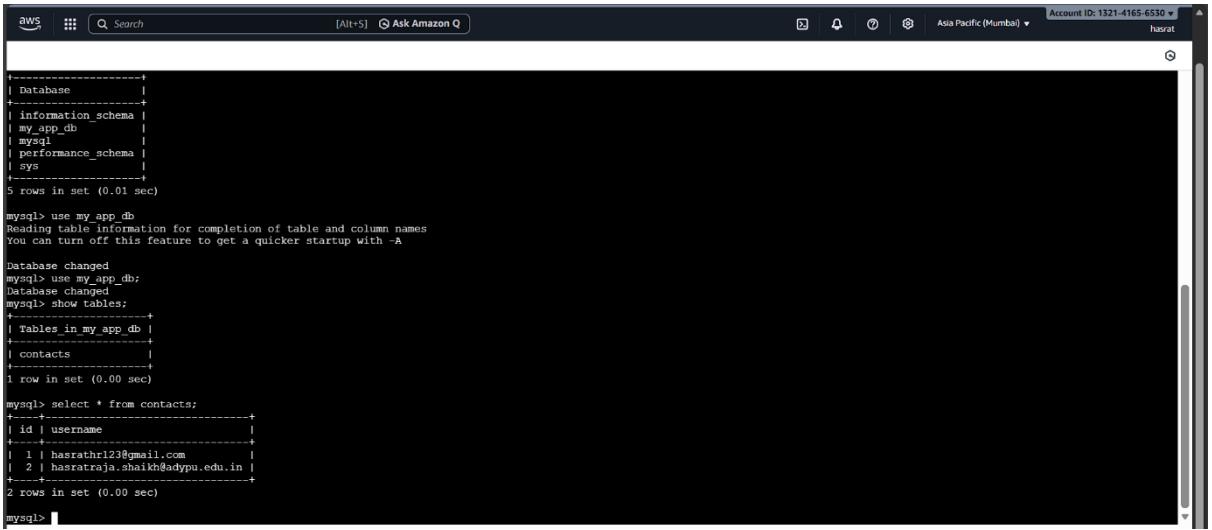
Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> [REDACTED]
```

14. We see the Data is stored successfully



The screenshot shows a Lambda function's code editor with a terminal window. The terminal displays the MySQL command-line interface. The user runs the command `show databases;`, which lists the databases: database, information_schema, my_app_db, mysql, performance_schema, and sys. Then, the user switches to the `my_app_db` database with `use my_app_db;`. After running `show tables;`, it shows two tables: contacts and users. Finally, the user runs `select * from contacts;`, which returns two rows of data: (1, hasrathr123@gmail.com) and (2, hasratraja.shaiikh@adypu.edu.in).

```
aws | Search [Alt+S] Ask Amazon Q Account ID: 1321-4165-6530 ▾ hasrat
-----
| database          |
+-----+
| information_schema |
| my_app_db         |
| mysql             |
| performance_schema |
| sys               |
+-----+
5 rows in set (0.01 sec)

mysql> use my_app_db
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
mysql> show tables;
+-----+
| Tables_in_my_app_db |
+-----+
| contacts           |
| users              |
+-----+
1 row in set (0.00 sec)

mysql> select * from contacts;
+----+-----+
| id | username        |
+----+-----+
| 1  | hasrathr123@gmail.com |
| 2  | hasratraja.shaiikh@adypu.edu.in |
+----+
2 rows in set (0.00 sec)

mysql> [REDACTED]
```

