BridgeSync Hub Database Optimization and Deployment

Table of Contents

1. Introduction

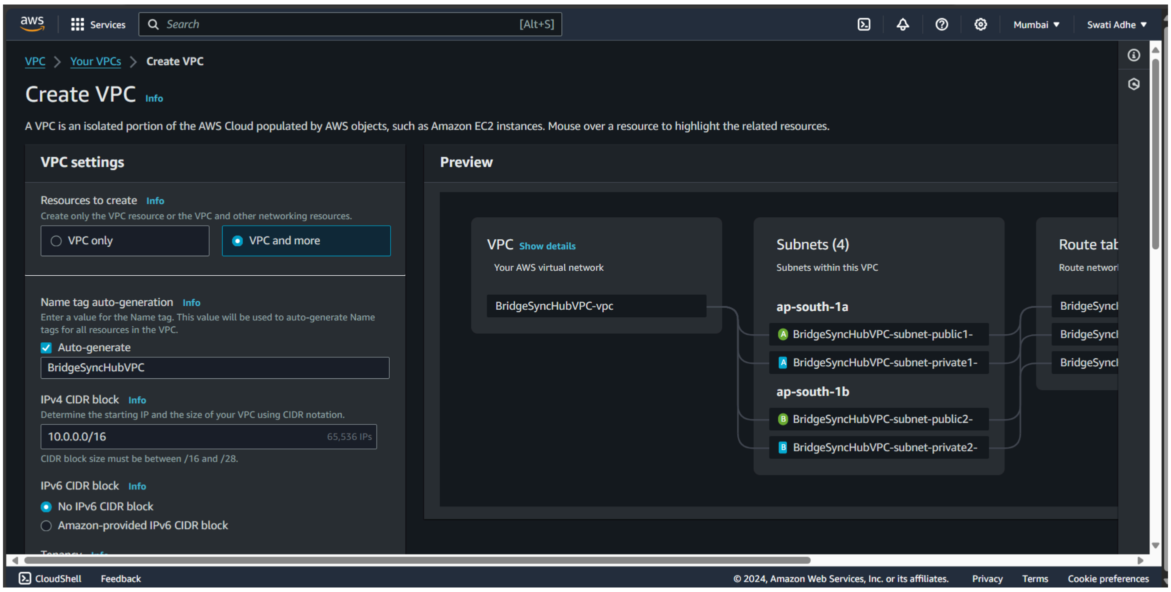
This project **"BridgeSync Hub Database Optimization and Deployment"** addresses the critical challenges faced by BridgeSync Hub in optimizing their database performance and ensuring reliable deployment. With a focus on leveraging AWS services, the project involves setting up a scalable and highly available database environment using Amazon RDS Aurora MySQL and configuring an EC2 instance to integrate with this database. The objective is to enhance the database infrastructure to support the address book application effectively, ensuring improved performance, availability, and scalability.

2. Prerequisites

1. **AWS EC2 Instance**: A running EC2 instance with appropriate SSH key and security group settings.
2. **AWS RDS Aurora MySQL Instance**: Set up within your VPC.
3. **MySQL Client**: Installed on the EC2 instance for database management.

3. Creating a VPC

1. **Open the VPC Console**:
   - Go to the VPC Dashboard.
2. **Create a VPC**:
   - Click on “Create VPC.”
   - Choose “VPC and more” and configure:
     - **Name tag**: `BridgeSyncHubVPC`
     - **IPv4 CIDR block**: `10.0.0.0/16`
     - **Number of Availability Zones (AZs):** 2
     - **Number of public subnets:** 2
     - **Number of private subnets:** 2
     - **NAT gateways:** In 1 Az
   - Click “Create VPC”.



VPC > Your VPCs > Create VPC

Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.

VPC settings

Resources to create Info
Create only the VPC resource or the VPC and other networking resources.

○ VPC only ● VPC and more

Name tag auto-generation Info
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

☑ Auto-generate
BridgeSyncHubVPC

IPv4 CIDR block Info
Determine the starting IP and the size of your VPC using CIDR notation.
10.0.0.0/16 65,536 IPs
CIDR block size must be between /16 and /28.

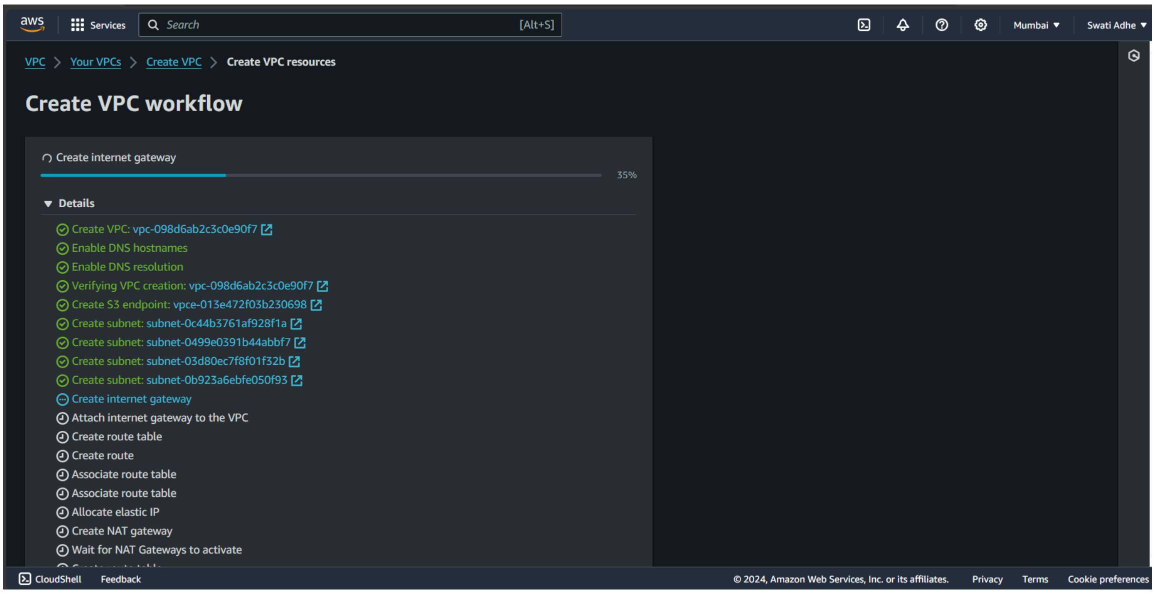
IPv6 CIDR block Info
● No IPv6 CIDR block
○ Amazon-provided IPv6 CIDR block

Preview

VPC Show details
Your AWS virtual network
BridgeSyncHubVPC-vpc

Subnets (4)
Subnets within this VPC
ap-south-1a
BridgeSyncHubVPC-subnet-public1-
BridgeSyncHubVPC-subnet-private1-
ap-south-1b
BridgeSyncHubVPC-subnet-public2-
BridgeSyncHubVPC-subnet-private2-

Route tab
Route networ
BridgeSyncl
BridgeSyncl
BridgeSyncl



VPC > Your VPCs > Create VPC > Create VPC resources

Create VPC workflow

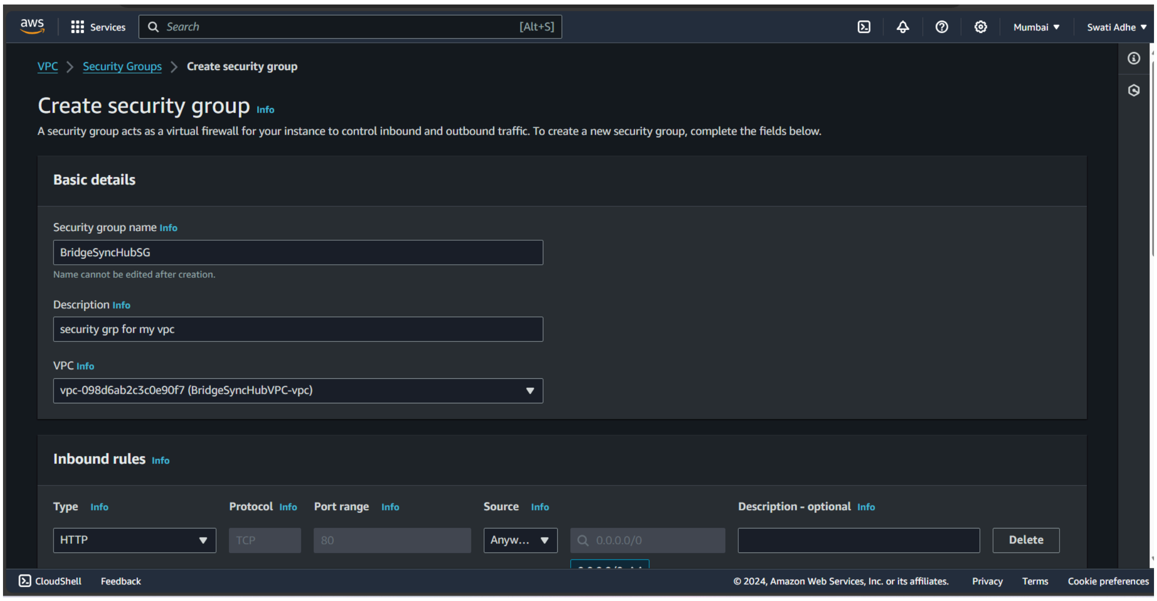
Create internet gateway
35%

▼ Details

- ✓ Create VPC: vpc-098d6ab2c3c0e90f7
- ✓ Enable DNS hostnames
- ✓ Enable DNS resolution
- ✓ Verifying VPC creation: vpc-098d6ab2c3c0e90f7
- ✓ Create S3 endpoint: vpce-013e472f03b230698
- ✓ Create subnet: subnet-0c44b3761af928f1a
- ✓ Create subnet: subnet-0499e0391b44abbf7
- ✓ Create subnet: subnet-03d80ec7f8f01f32b
- ✓ Create subnet: subnet-0b923a6ebfe050f93
- Create internet gateway
- Attach internet gateway to the VPC
- Create route table
- Create route
- Associate route table
- Associate route table
- Allocate elastic IP
- Create NAT gateway
- Wait for NAT Gateways to activate

4. Creating a Security Group

1. **Open the EC2 Console**:
    - Go to the EC2 Dashboard.
2. **Create a Security Group**:
    - Click on “Security Groups” and then “Create security group.”
    - Configure:
        - **Name**: `BridgeSyncHubSG`
        - **Description**: Security group for BridgeSync Hub.
        - **VPC**: Select the VPC you created.
    - Add inbound rules:
        - **SSH (Port 22)**: Source `0.0.0.0/0` (or restricted IP range for better security).
        - **HTTP (Port 80)**: Source `0.0.0.0/0` (or restricted IP range for better security).
        - **HTTPS (Port 443)**: Source `0.0.0.0/0` (or restricted IP range for better security).
        - **MySQL/Aurora (Port 3306)**: Source `0.0.0.0/0` (or restricted IP range for better security).
3. **Create and Attach Security Group**:
    - Attach the security group to your EC2 instance and RDS instance.



Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name Info: BridgeSyncHubSG

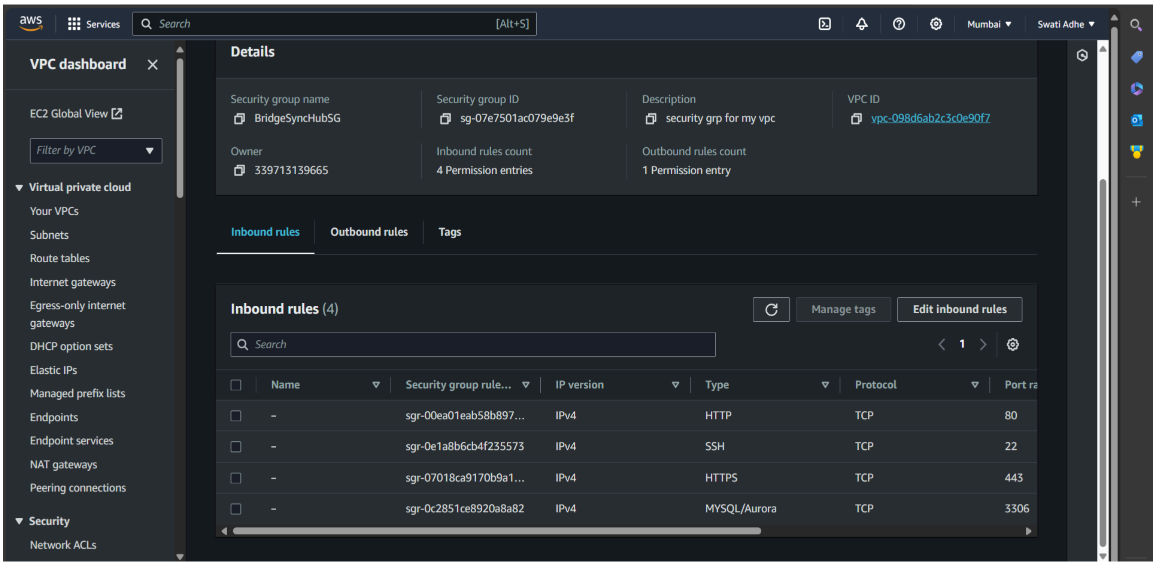
Name cannot be edited after creation.

Description Info: security grp for my vpc

VPC Info: vpc-098d6ab2c3c0e90f7 (BridgeSyncHubVPC-vpc)

Inbound rules Info

Type	Protocol	Port range	Source		Description - optional	
HTTP	TCP	80	Anyw...	0.0.0.0/0		Delete



Details

Security group name	Security group ID	Description	VPC ID
BridgeSyncHubSG	sg-07e7501ac079e9e3f	security grp for my vpc	vpc-098d6ab2c3c0e90f7

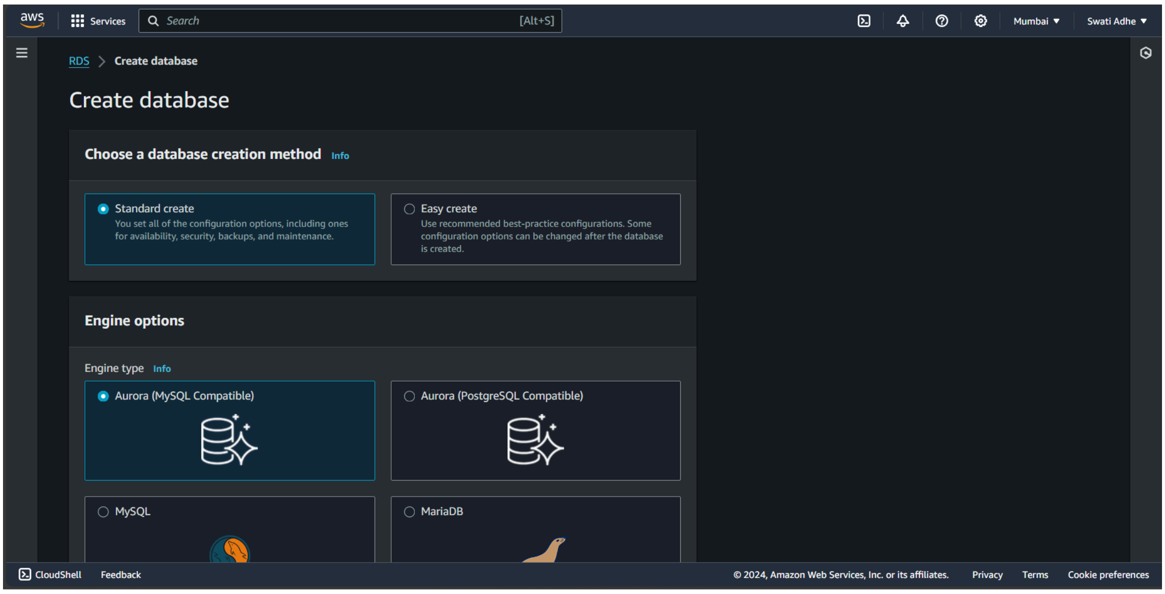
Owner	Inbound rules count	Outbound rules count
339713139665	4 Permission entries	1 Permission entry

Inbound rules (4)

□	Name	Security group rule...	IP version	Type	Protocol	Port ra
□	–	sgr-00ea01eab58b897...	IPv4	HTTP	TCP	80
□	–	sgr-0e1a8b6cb4f235573	IPv4	SSH	TCP	22
□	–	sgr-07018ca9170b9a1...	IPv4	HTTPS	TCP	443
□	–	sgr-0c2851ce8920a8a82	IPv4	MYSQL/Aurora	TCP	3306

5. Creating an RDS Aurora MySQL Instance

1. **Open the RDS Console**:
   - Go to the RDS Dashboard.
2. **Create a New Aurora MySQL Cluster**:
   - Click “Create database.”
   - Choose “Aurora (MySQL Compatible)”.
   - Select the database template: “Production”.
3. **Configure Database Settings**:
   - **DB Cluster Identifier**: `bridgesynchubcluster`
   - **Master Username**: `admin`
   - **Master Password**: `admin123`
4. **Choose DB Instance Class**:
   - Select an appropriate instance class (e.g., `db.r6g.2xlarge`).
5. **Configure Availability & Durability**:
   - **Multi-AZ Deployment**: Enable for high availability.
   - **Aurora Replicas**: Optionally configure.
6. **Configure Connectivity**:
   - **VPC**: Select the VPC you created.
   - **Subnet Group**: Select the subnet group associated with your VPC.
   - **Public Accessibility**: Choose “No” if you don”t need public access.
   - **VPC Security Groups**: Attach the security group you created.
7. **Review and Create**:
   - Review your settings and click “Create database.”



Create database

Choose a database creation method Info

● Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

○ Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

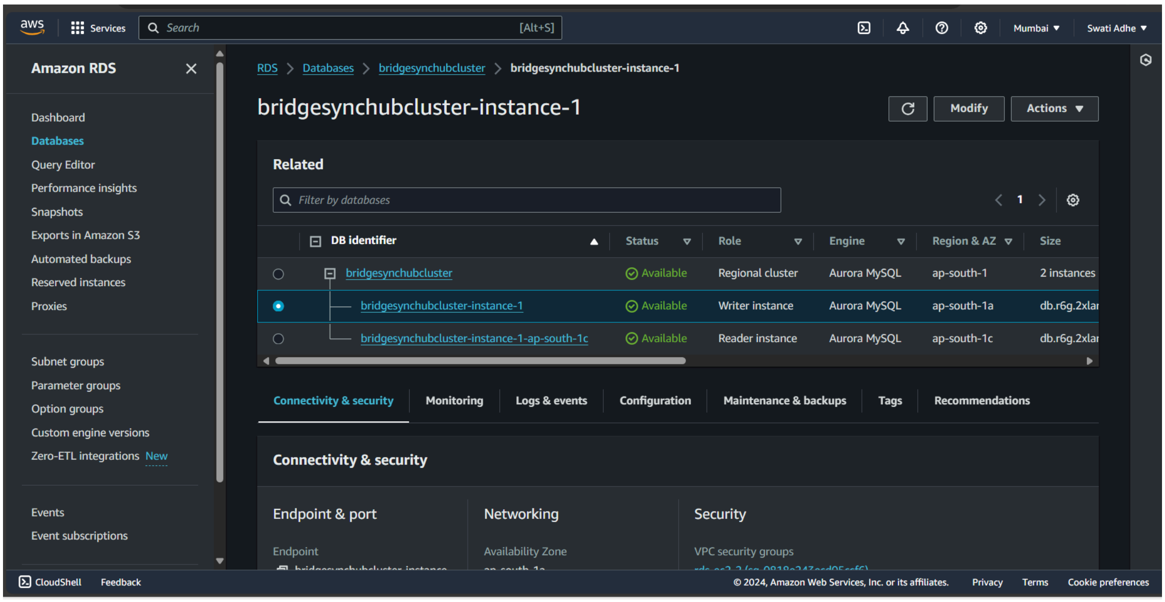
Engine type Info

● Aurora (MySQL Compatible)

○ Aurora (PostgreSQL Compatible)

○ MySQL

○ MariaDB



bridgesynchubcluster-instance-1

Related

	DB identifier	Status	Role	Engine	Region & AZ	Size
○	bridgesynchubcluster	✓ Available	Regional cluster	Aurora MySQL	ap-south-1	2 instances
●	bridgesynchubcluster-instance-1	✓ Available	Writer instance	Aurora MySQL	ap-south-1a	db.r6g.2xlar
○	bridgesynchubcluster-instance-1-ap-south-1c	✓ Available	Reader instance	Aurora MySQL	ap-south-1c	db.r6g.2xlar

Connectivity & security

Endpoint & port

Endpoint

Networking

Availability Zone

Security

VPC security groups

6. Creating and Connecting an EC2 Instance

6.1. Creating an EC2 Instance

1. **Open the EC2 Console**:
   - Go to the EC2 Dashboard.
2. **Launch a New EC2 Instance**:
   - Click “Launch Instance.”
   - Select an Amazon Machine Image (AMI) of your choice (e.g., Amazon Linux 2).
   - Choose an instance type (e.g., `t2.micro` for testing).
   - Configure instance details:
     - **Network**: Select the VPC you created.
     - **Subnet**: Select one of the subnets.
   - Add storage if needed.
   - Configure security group:
     - Choose the security group you created.
   - **User Data**: Paste the following script into the “User data” field for automatic setup:

```
#!/bin/sh

# Install a LAMP stack
dnf install -y httpd wget php-fpm php-mysqli php-json php php-devel
dnf install -y mariadb105-server
dnf install -y httpd php-mbstring

# Start the web server
chkconfig httpd on
systemctl start httpd

# Install the web pages for our lab
if [ ! -f /var/www/html/immersion-day-app-php7.zip ]; then
   cd /var/www/html
   wget -O 'immersion-day-app-php7.zip'
```

```
'https://static.us-east-1.prod.workshops.aws/public/8228a7bb-0533-4d
04-9ae9-f532c745e359/assets/immersion-day-app-php7.zip'
   unzip immersion-day-app-php7.zip
fi

# Install the AWS SDK for PHP
if [ ! -f /var/www/html/aws.zip ]; then
   cd /var/www/html
   mkdir vendor
   cd vendor
   wget
https://docs.aws.amazon.com/aws-sdk-php/v3/download/aws.zip
   unzip aws.zip
fi

# Update existing packages
dnf update -y
```

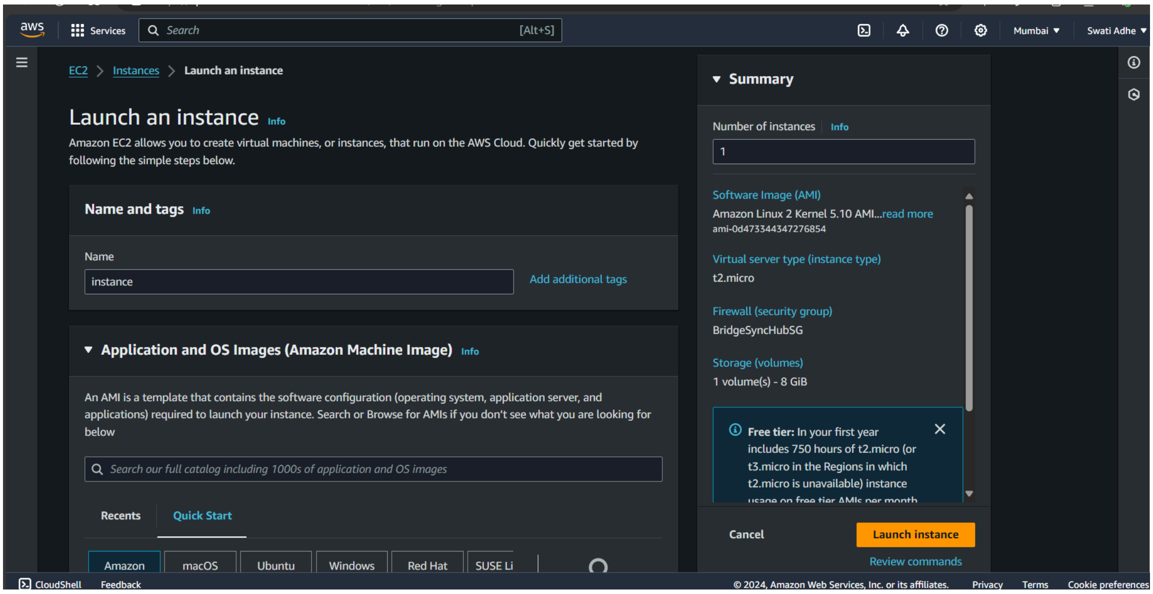
6.2. Connecting an EC2 Instance

You can connect to your EC2 instance using one of the following methods:

1. **EC2 Instance Connect:**
    1. Choose the "EC2 Instance Connect" option.
    2. Click "Connect" to open a browser-based terminal.

2. SSH Client:

```
ssh -i "key.pem"
ec2-user@13.201.5.98
```



Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: instance

▾ Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images
Recents
Quick Start
Amazon
macOS
Ubuntu
Windows
Red Hat
SUSE Li

▾ Summary

Number of instances: 1

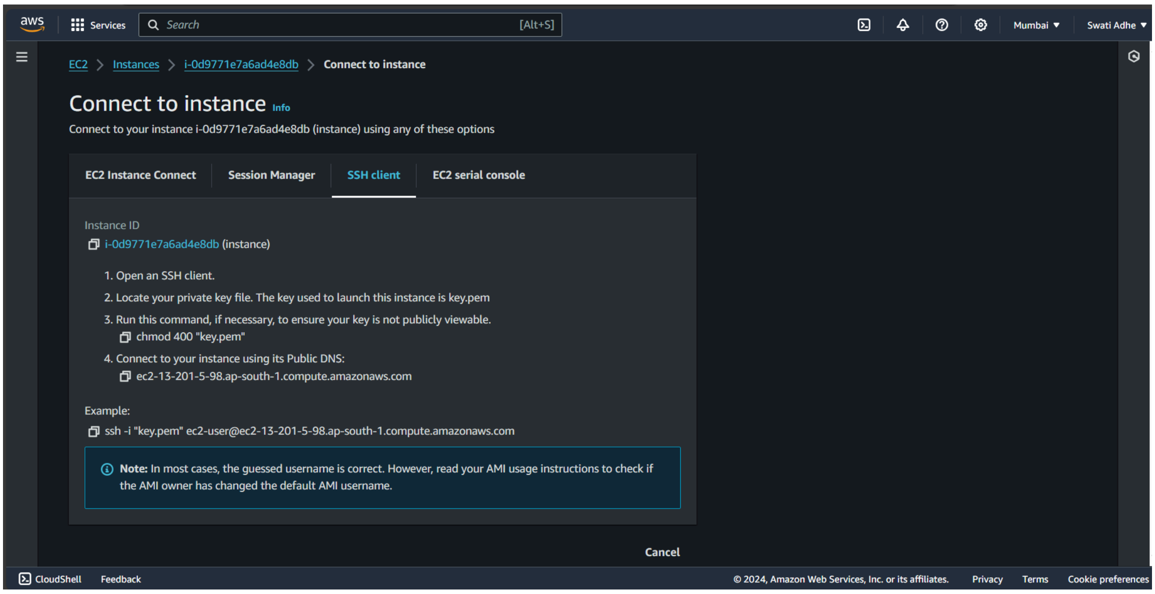
Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-0d473344347276854

Virtual server type (instance type)
t2.micro

Firewall (security group)
BridgeSyncHubSG

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month



Connect to instance

Connect to your instance i-0d9771e7a6ad4e8db (instance) using any of these options

Instance ID
i-0d9771e7a6ad4e8db (instance)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   ```
   chmod 400 "key.pem"
   ```
4. Connect to your instance using its Public DNS:
   ```
   ec2-13-201-5-98.ap-south-1.compute.amazonaws.com
   ```

Example:

```
ssh -i "key.pem" ec2-user@ec2-13-201-5-98.ap-south-1.compute.amazonaws.com
```

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

7. Installing MySQL Client

1. Update Package Lists:

```
sudo yum update -y
```

2. Install MySQL Client:

```
sudo yum install -y mysql
```

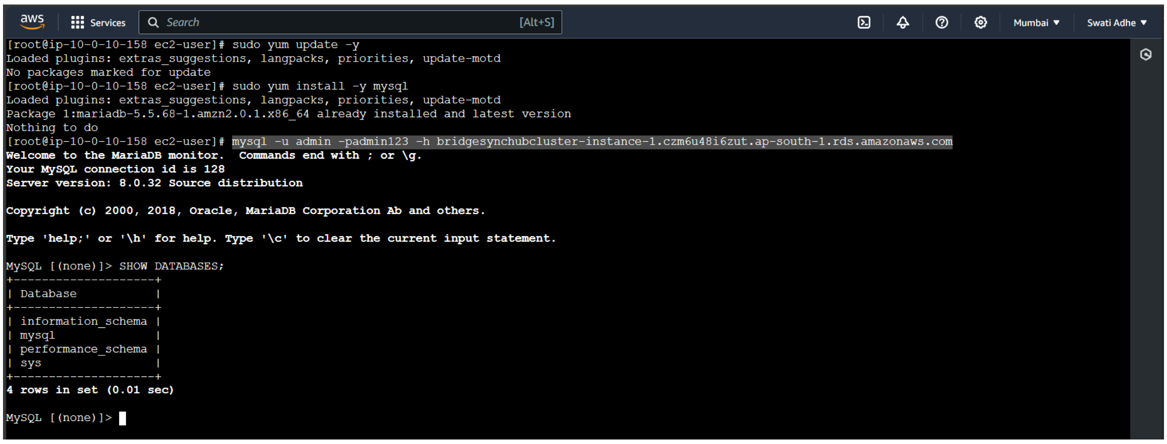
8. Connecting to RDS Aurora MySQL

1. Switch to the root user:

```
sudo su
```

2. Use the MySQL client to connect to your RDS Aurora MySQL instance:

```
mysql -u <master-username> -p<master-password> -h <aurora-writer-endpoint>
```

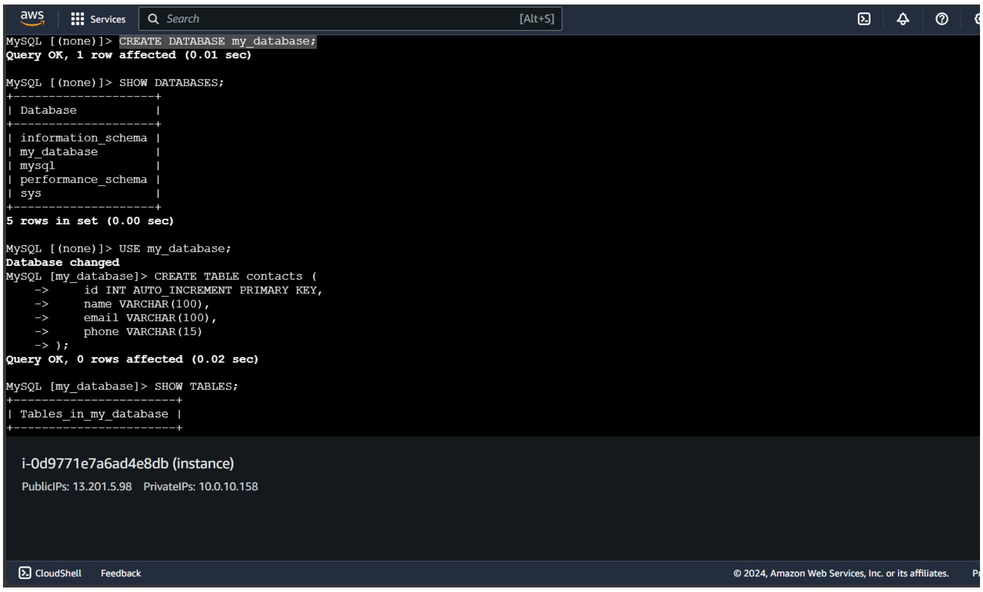


```
[root@ip-10-0-10-158 ec2-user]# sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[root@ip-10-0-10-158 ec2-user]# sudo yum install -y mysql
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package 1:mariadb-5.5.68-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
[root@ip-10-0-10-158 ec2-user]# mysql -u admin -padmin123 -h bridgesynchubcluster-instance-1.czm6u48i6zut.ap-south-1.rds.amazonaws.com
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 128
Server version: 8.0.32 Source distribution

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

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

MySQL [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.01 sec)

MySQL [(none)]> 
```

9. Managing Databases and Tables

- **Creating a New Database**

```
CREATE DATABASE my_database;
```



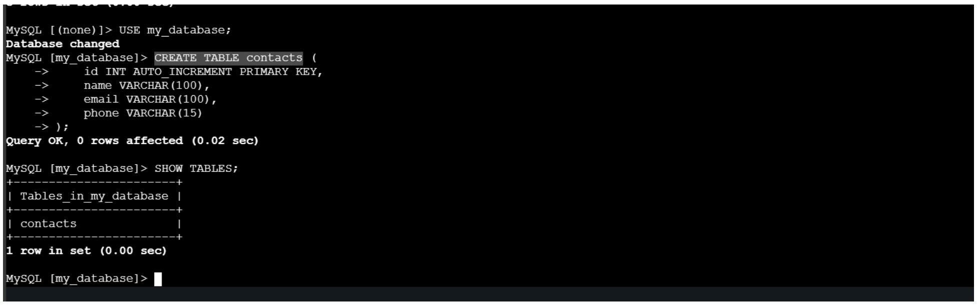
```
MySQL [(none)]> CREATE DATABASE my_database;
Query OK, 1 row affected (0.01 sec)

MySQL [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| my_database        |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

MySQL [(none)]> USE my_database;
Database changed
MySQL [my_database]> CREATE TABLE contacts (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.02 sec)

MySQL [my_database]> SHOW TABLES;
+-----------------------+
| Tables_in_my_database |
+-----------------------+
```

i-0d9771e7a6ad4e8db (instance)

PublicIPs: 13.201.5.98   PrivateIPs: 10.0.10.158

- **Creating Tables**

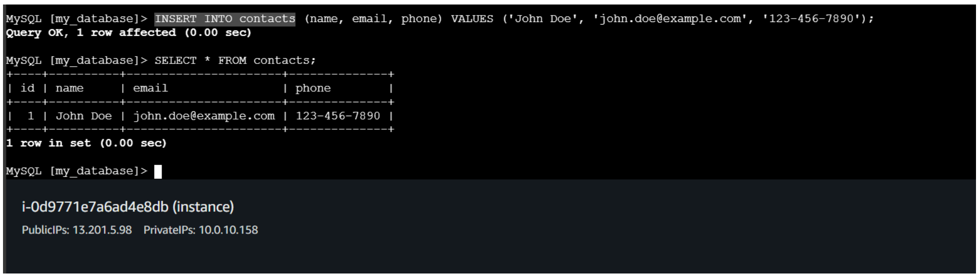
```
USE my_database;

CREATE TABLE contacts (
   id INT AUTO_INCREMENT
PRIMARY KEY,
   name VARCHAR(100),
   email VARCHAR(100),
    phone VARCHAR(15)
);
```



```
MySQL [(none)]> USE my_database;
Database changed
MySQL [my_database]> CREATE TABLE contacts (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.02 sec)

MySQL [my_database]> SHOW TABLES;
+-----------------------+
| Tables_in_my_database |
+-----------------------+
| contacts              |
+-----------------------+
1 row in set (0.00 sec)

MySQL [my_database]> 
```

- **Inserting Data**

```
INSERT INTO contacts (name, email, phone)
VALUES ('John Doe', 'john.doe@example.com',
'123-456-7890');
```



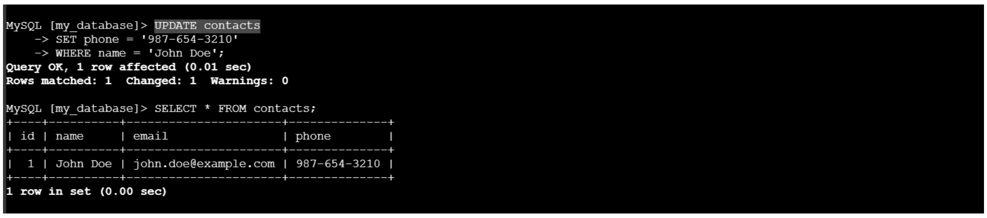
```
MySQL [my_database]> INSERT INTO contacts (name, email, phone) VALUES ('John Doe', 'john.doe@example.com', '123-456-7890');
Query OK, 1 row affected (0.00 sec)

MySQL [my_database]> SELECT * FROM contacts;
+----+----------+----------------------+--------------+
| id | name     | email                | phone        |
+----+----------+----------------------+--------------+
|  1 | John Doe | john.doe@example.com | 123-456-7890 |
+----+----------+----------------------+--------------+
1 row in set (0.00 sec)

MySQL [my_database]> 
```

i-0d9771e7a6ad4e8db (instance)

PublicIPs: 13.201.5.98   PrivateIPs: 10.0.10.158

- **Modifying Data**

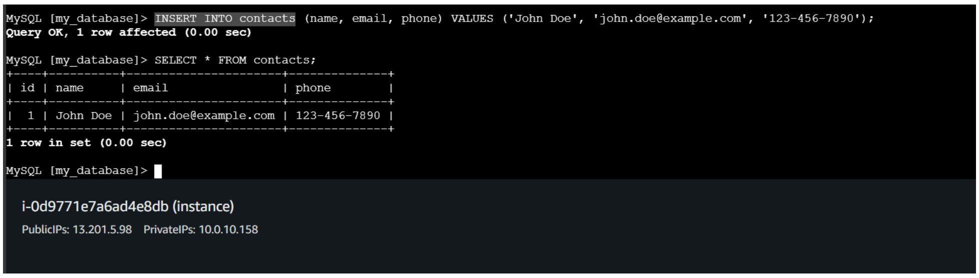
```
UPDATE contacts
SET phone =
'987-654-3210'
WHERE name = 'John
Doe';
```



```
MySQL [my_database]> UPDATE contacts
    -> SET phone = '987-654-3210'
    -> WHERE name = 'John Doe';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MySQL [my_database]> SELECT * FROM contacts;
+----+----------+----------------------+--------------+
| id | name     | email                | phone        |
+----+----------+----------------------+--------------+
|  1 | John Doe | john.doe@example.com | 987-654-3210 |
+----+----------+----------------------+--------------+
1 row in set (0.00 sec)
```

- **Deleting Data**

```
DELETE FROM contacts
WHERE name = 'John
Doe';
```



```
MySQL [my_database]> INSERT INTO contacts (name, email, phone) VALUES ('John Doe', 'john.doe@example.com', '123-456-7890');
Query OK, 1 row affected (0.00 sec)

MySQL [my_database]> SELECT * FROM contacts;
+----+----------+----------------------+--------------+
| id | name     | email                | phone        |
+----+----------+----------------------+--------------+
|  1 | John Doe | john.doe@example.com | 123-456-7890 |
+----+----------+----------------------+--------------+
1 row in set (0.00 sec)

MySQL [my_database]>
```

i-0d9771e7a6ad4e8db (instance)

PublicIPs: 13.201.5.98 PrivateIPs: 10.0.10.158

10. Conclusion

The project **"BridgeSync Hub Database Optimization and Deployment"** successfully deployed Amazon RDS Aurora MySQL and configured an EC2 instance. This setup delivers a robust and reliable database solution, enhancing the performance and stability of the BridgeSync Hub address book application.