

EXPERIMENT 5

Aim: To study and Implement Database as a Service on SQL/NOSQL databases like AWS RDS, AZURE SQL/ MongoDB Lab/ Firebase.

Objective:

- Describe key database concepts.
- Compare and contrast relational and nonrelational databases.
- Identify the AWS database services.
- Discuss the features and concepts of Amazon RDS.
- Describe the benefits and use cases of Amazon RDS.
- Describe how to set up an RDS instance using the console.
- Identify how to connect to the database instance.
- Discuss how to use SQL commands to read and write to the database.

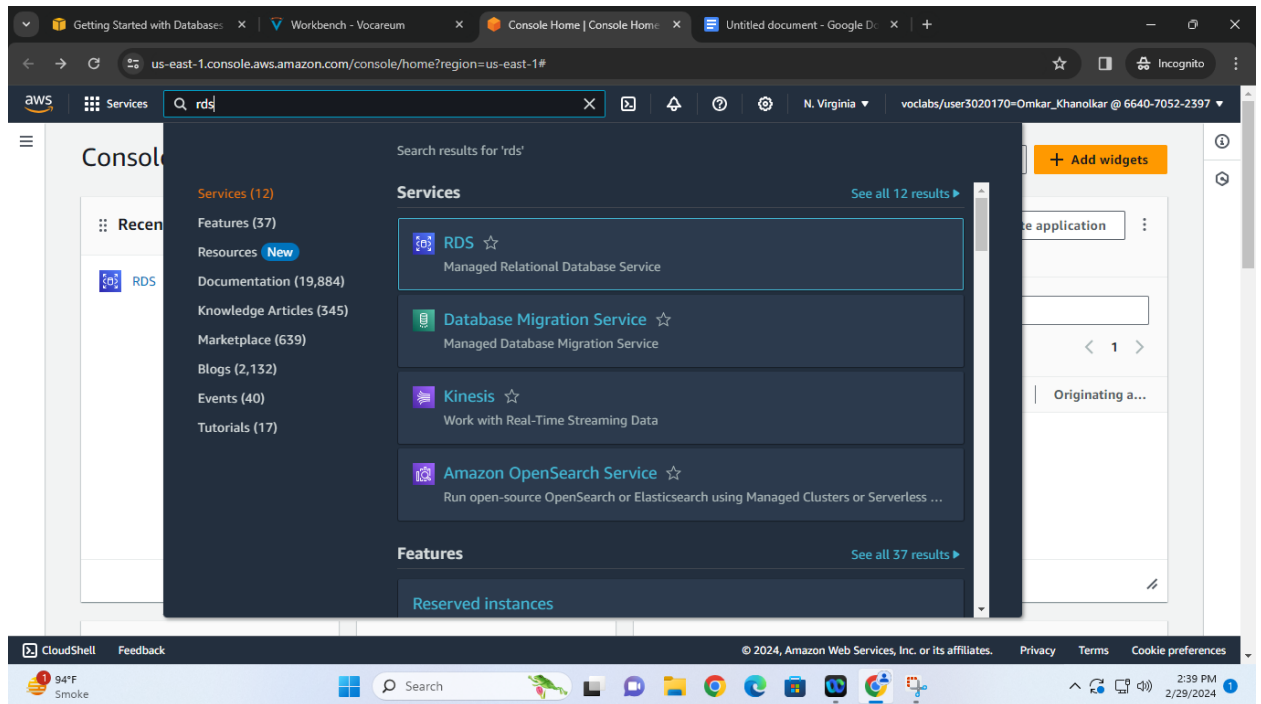
Theory: To embark on a study and implementation of Database as a Service (DBaaS) using SQL/NOSQL databases like AWS RDS, Azure SQL, MongoDB, or Firebase, it's crucial to grasp the foundational concepts and functionalities inherent in these cloud database platforms. DBaaS offers managed database services that eliminate the complexity of database administration tasks, such as provisioning, patching, backups, and scaling, allowing users to focus on application development and data management. The theory behind this endeavor revolves around understanding the core components and features of DBaaS offerings provided by AWS, Azure, MongoDB, and Firebase. These platforms offer a variety of database options, including relational databases like MySQL, PostgreSQL, SQL Server, and non-relational databases like MongoDB, Firestore, and Firebase Realtime Database. Key concepts include selecting the appropriate database service based on data model requirements, performance considerations, scalability needs, and cost constraints. Additionally, understanding database security, data integrity, backup and recovery mechanisms, and data migration strategies are essential aspects of studying and implementing DBaaS. By leveraging services like AWS RDS, Azure SQL Database, MongoDB Atlas, or Firebase, participants gain practical experience in provisioning, configuring, and managing databases in the cloud, ensuring high availability, reliability, and scalability for their applications. This experiment serves as a hands-on exploration of modern database technologies and equips participants with the knowledge and skills to leverage DBaaS effectively for building resilient and scalable data solutions in the cloud.

Implementation And Output:

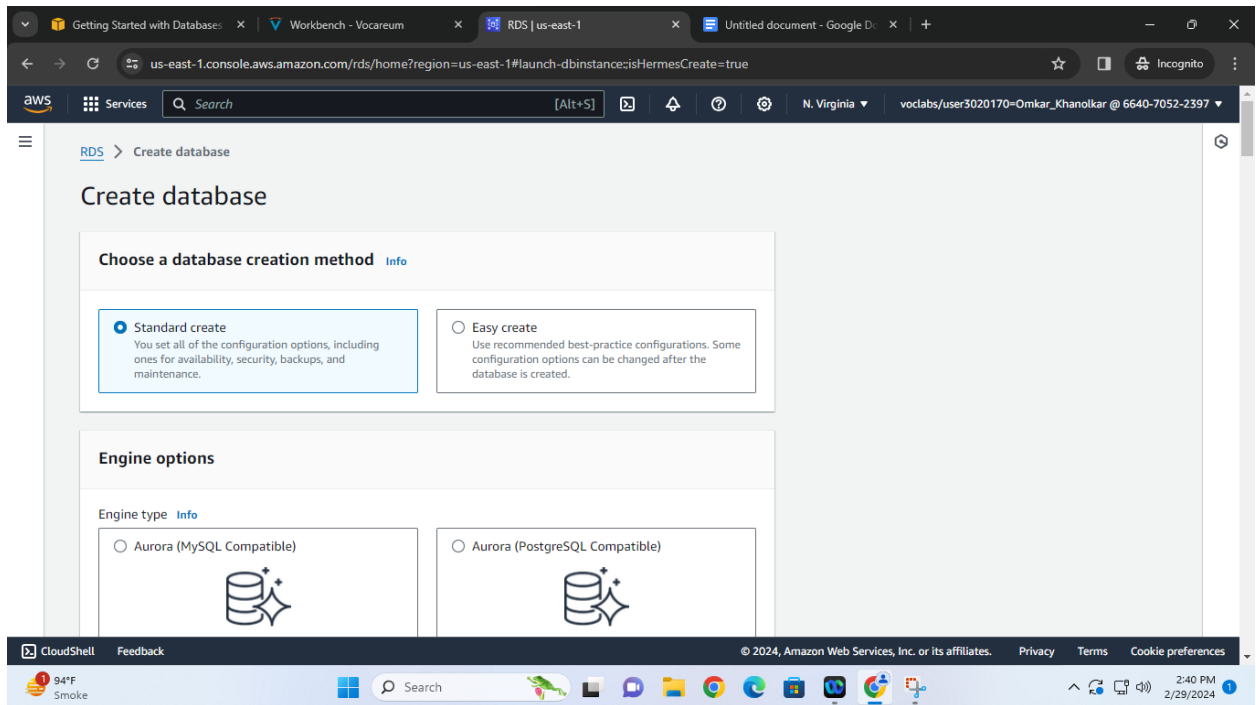
Task 1: Creating an Amazon RDS database

In this task, you create a MySQL database in your virtual private cloud (VPC). MySQL is a popular open-source relational database management system (RDBMS), so there are no software licensing fees.

1. On the **Services** menu, choose **RDS**.



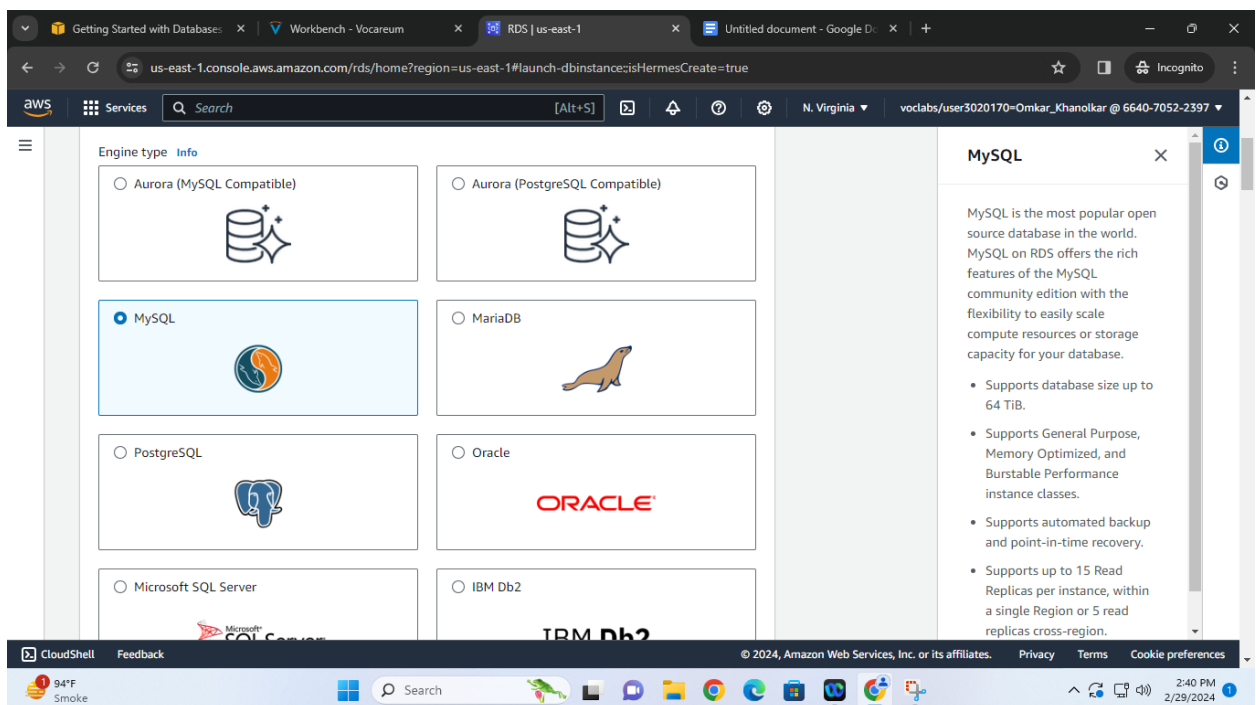
2. Choose **Create database**
For this lab, you will keep the **Choose a database creation method** as **Standard create** to understand the full set of features available.



3. Under **Engine options**, select **MySQL**.

- For Version, keep MySQL 8.0.28.

In the options, you might notice Amazon Aurora. Aurora is a global-scale relational database service built for the cloud with full MySQL and PostgreSQL compatibility. If your company uses large-scale MySQL or PostgreSQL databases, Aurora can provide enhanced performance.



4. In the **Templates** section, select **Dev/Test**.

You can now select a database configuration, including software version, instance class, storage, and login settings. The *Multi-AZ deployment* option automatically creates a replica of the database in a second Availability Zone for high availability.

Templates

Choose a sample template to meet your use case.

☐ **Production**
Use defaults for high availability and fast, consistent performance.

☒ **Dev/Test**
This instance is intended for development use outside of a production environment.

☐ **Free tier**
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.
[Info](#)

5. In the **Availability and durability** section, choose **Single DB instance**

Availability and durability

Deployment options [Info](#)
The deployment options below are limited to those supported by the engine you selected above.

☐ **Multi-AZ DB Cluster**
Creates a DB cluster with a primary DB instance and two readable standby DB instances, with each DB instance in a different Availability Zone (AZ). Provides high availability, data redundancy and increases capacity to serve read workloads.

☐ **Multi-AZ DB instance**
Creates a primary DB instance and a standby DB instance in a different AZ. Provides high availability and data redundancy, but the standby DB instance doesn't support connections for read workloads.

☒ **Single DB instance**
Creates a single DB instance with no standby DB instances.

6. In the **Settings** section next, configure the following options :

- a. **DB instance identifier:** inventory-db
- **Master username:** admin
 - **Master password:** lab-password
- **Confirm password:** lab-password

Settings

DB instance identifier [Info](#)

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings

Master username [Info](#)

Type a login ID for the master user of your DB instance.

1 to 16 alphanumeric characters. The first character must be a letter.

☐ **Manage master credentials in AWS Secrets Manager**

Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.

Note: Please use these values *verbatim*, do not make any changes.

7. In the **Instance configuration** section, configure the following options for DB instance class:
 - a. Choose **Burstable classes (includes t classes)**.
 - b. Choose **db.t3.micro**.

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

DB instance class [Info](#)

▼ Hide filters

- ☒ Show instance classes that support Amazon RDS Optimized Writes [Info](#)
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.
- ☐ Include previous generation classes
- ☐ Standard classes (includes m classes)
- ☐ Memory optimized classes (includes r and x classes)
- ☒ Burstable classes (includes t classes)

db.t3.micro

2 vCPUs 1 GiB RAM Network: 2,085 Mbps

8. In the **Storage** section next,
 - a. For **Storage type** choose **General Purpose SSD (gp2)** from the Dropdown menu.
 - b. For **Allocated storage** keep 20.
 - c. Clear or Deselect **Enable storage autoscaling**.

Storage type [Info](#)

General Purpose SSD (gp2)
Baseline performance determined by volume size

Allocated storage [Info](#)

20
GiB

The minimum value is 20 GiB and the maximum value is 6,144 GiB

i Provisioning less than 100 GiB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [Learn more](#)

i After you modify the storage for a DB instance, the status of the DB instance will be in storage-optimization. Your instance will remain available as the storage-optimization operation completes. [Learn more](#)

▼ Storage autoscaling

Storage autoscaling [Info](#)

Provides dynamic scaling support for your database's storage based on your application's needs.

☐ Enable storage autoscaling

Enabling this feature will allow the storage to increase after the specified threshold is

9. In the **Connectivity** section, configure the following options:

- For **Compute resource**
 - keep default **Don't connect to an EC2 compute resource**, as you will establish this manually at a later stage.
- For **Virtual private cloud (VPC)** Choose **Lab VPC** from the Dropdown menu.
- For **DB Subnet group**, keep default value **rds-lab-db-subnet-group**
- For **Public access** Keep default value (**No**)
- For **VPC security group (firewall)**
 - Choose the **X** on **default** to remove this security group.
 - Choose **DB-SG** from the dropdown list to add it.
 - For **Availability Zone**, Keep default **No preference**
- For **Database authentication** keep default value **Password authentication**

Connectivity [Info](#)



Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

- ☒ **Don't connect to an EC2 compute resource**
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

- ☐ **Connect to an EC2 compute resource**
Set up a connection to an EC2 compute resource for this database.

Virtual private cloud (VPC) [Info](#)

Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

Lab VPC (vpc-01e575a3a357da9a4)

4 Subnets, 2 Availability Zones



Only VPCs with a corresponding DB subnet group are listed.

- After a database is created, you can't change its VPC.

DB subnet group [Info](#)

Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

rds-lab-db-subnet-group

2 Subnets, 2 Availability Zones



Public access [Info](#)

- ☐ **Yes**
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.
- ☒ **No**
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)

Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

- ☒ **Choose existing**
Choose existing VPC security groups

- ☐ **Create new**
Create new VPC security group

Existing VPC security groups

Choose one or more options



DB-SG X

Database authentication

Database authentication options [Info](#)

☒

Password authentication

Authenticates using database passwords.

☐

Password and IAM database authentication

Authenticates using the database password and user credentials through AWS IAM users and roles.

☐

Password and Kerberos authentication

Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

10. In the **Monitoring** section
- Clear/DeSelect the **Enable Enhanced monitoring** option.

Monitoring

☐ **Enable Enhanced Monitoring**
Enabling Enhanced Monitoring metrics are useful when you want to see how different processes or threads use the CPU.

11. Expand the following **Additional configuration** section by choosing
- Under **Database options**, for **Initial database name**, enter `inventory`

▼ Additional configuration
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name [Info](#)

inventory

If you do not specify a database name, Amazon RDS does not create a database.

This is the logical name of the database that the application will use.
You can review the few other options displayed on the page, but leave them set to

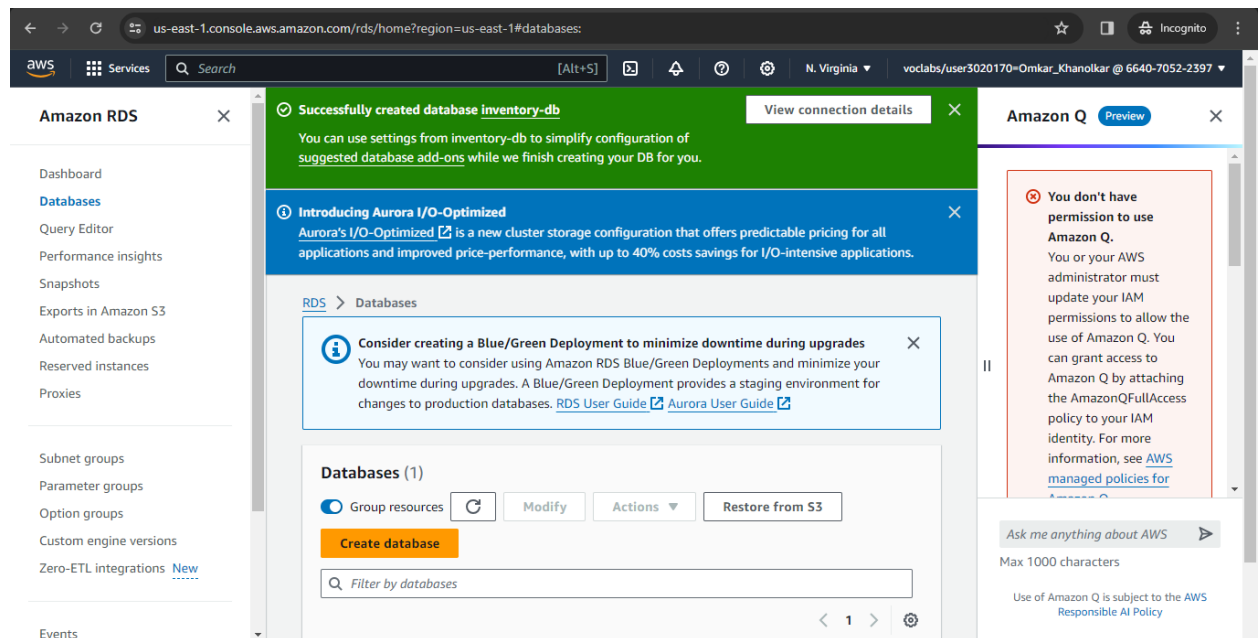
their default values. Options include **automatic backups**, **Log exports**, **Encryption** and **automatic version upgrades**. The ability to activate these features with check boxes demonstrates the power of using a fully managed database solution instead of installing, backing up, and maintaining the database yourself.

12. At the bottom of the page, choose **Create database**

You should receive this message: **Creating database inventory-db**.

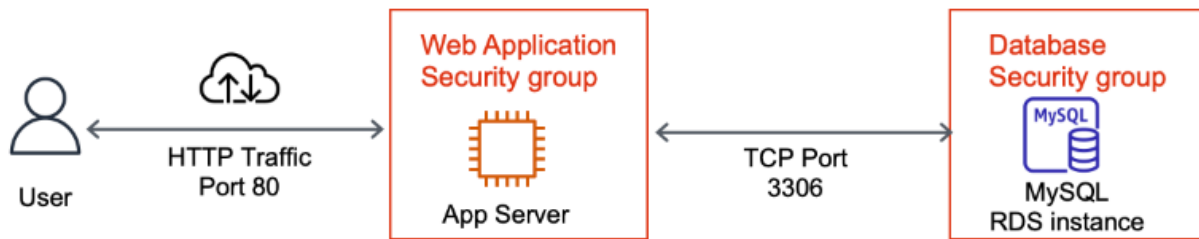
If you receive an error message that mentions **rds-monitoring-role**, confirm that you have cleared the **Enable Enhanced monitoring** option in the previous step, and then try again.

Before you continue to the next task, the database instance status must be **Available**. This process could take several minutes.



Task 2: Configuring web application communication with the database instance

This lab automatically deployed an Amazon Elastic Compute Cloud (Amazon EC2) instance with a running web application. You must use the IP address of the instance to connect to the application. In this task, you will use application to configure connection settings which will be stored in **AWS Secrets Manager** for further use.



13. On the **Services** menu, choose **EC2**.

14. In the left navigation pane, choose **Instances**.

In the center pane, there should be a running instance that is named **App Server**.

Instances (1) Info									
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>					Any state				
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	
<input type="checkbox"/>	App Server	i-078f4f5262930475f	Running	t3.micro	2/2 checks passed	View alarms	us-east-1a	ec2-3-93-198-121.co	

15. Select the check box for the **App Server** instance.

16. In the **Details** tab, copy the **Public IPv4 address** to your clipboard.

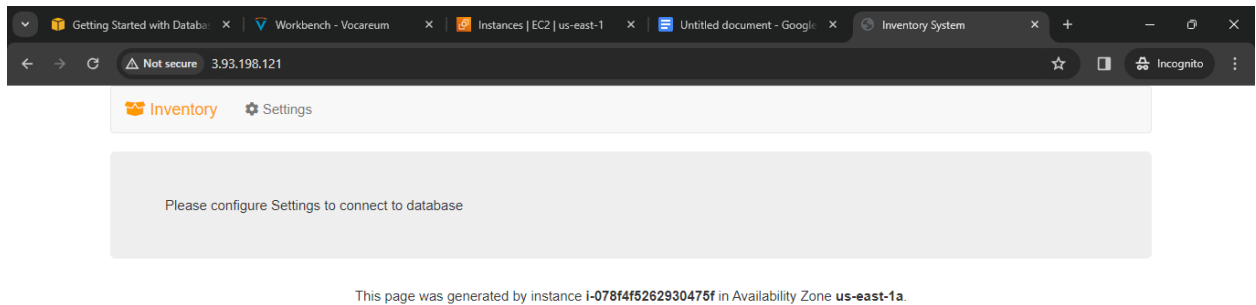
Tip: You can choose **copy** to copy the displayed value the displayed value.

Instances (1/1) Info									
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>					Any state				
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
<input checked="" type="checkbox"/>	App Server	i-078f4f5262930475f	Running	t3.micro	2/2 checks passed	View alarms	us-east-1a	ec2-3-93-198-121.com...	3.93.198.121

Instance: i-078f4f5262930475f (App Server)									
<div> Details Status and alarms Monitoring Security Networking Storage Tags </div>									
<div> <div> <div>▼ Instance summary Info</div> <div> <div>Instance ID</div> <div>i-078f4f5262930475f (App Server)</div> </div> <div> <div>IPv6 address</div> <div>–</div> </div> <div> <div>Hostname type</div> <div>IP name: ip-10-0-0-163.ec2.internal</div> </div> <div> <div>Answer private resource DNS name</div> <div>–</div> </div> <div> <div>Auto-assigned IP address</div> <div>–</div> </div> </div> <div> <div>Public IPv4 address</div> <div>3.93.198.121 open address</div> <div>Instance state</div> <div>Running</div> <div>Private IP DNS name (IPv4 only)</div> <div>ip-10-0-0-163.ec2.internal</div> <div>Instance type</div> <div>t3.micro</div> <div>VPC ID</div> <div>–</div> </div> <div> <div>Private IPv4 addresses</div> <div>10.0.0.163</div> <div>Public IPv4 DNS</div> <div>ec2-3-93-198-121.compute-1.amazonaws.com open address</div> <div>Elastic IP addresses</div> <div>–</div> <div>AWS Compute Optimizer finding</div> <div>–</div> </div> </div>									

17. Open a new web browser tab, paste the IP address into the address bar, and then press Enter.

The web application should appear. It does not display much information because the application is not yet connected to the database.



18. Choose **Settings**.

You can now configure the application to use the Amazon RDS database instance that you created earlier. You first retrieve the database endpoint so that the application knows how to connect to a database.

19. Return to the AWS Management Console, but do not close the application tab. (You will return to it soon).

20. On the **Services** menu, choose **RDS**.

21. In the left navigation pane, choose **Databases**.

22. Under **DB identifier**, Choose 'inventory-db'.

23. From the **Connectivity & security** section, copy the **Endpoint** to your clipboard.

It should look similar to this example: **inventory-db.crwxbgqad61a.rds.amazonaws.com**

RDS > Databases > inventory-db

inventory-db

Summary

DB identifier inventory-db	Status Available	Role Instance	Engine MySQL Community	Recommendations
CPU 3.06%	Class db.t3.micro	Current activity 0 Connections	Region & AZ us-east-1b	

[Connectivity & security](#)
[Monitoring](#)
[Logs & events](#)
[Configuration](#)
[Zero-ETL integrations](#)
[Maintenance & backups](#)
[Tags](#)
[Recommendations](#)

Connectivity & security

<h4>Endpoint & port</h4> <p>Endpoint inventory-db.cid6gucjthwr.us-east-1.rds.amazonaws.com</p> <p>Port 3306</p>	<h4>Networking</h4> <p>Availability Zone us-east-1b</p> <p>VPC Lab VPC (vpc-01e575a3a357da9a4)</p> <p>Subnet group rds-lab-db-subnet-group</p> <p>Subnets subnet-0183a568c00b47ad7 subnet-0b2c022f045c79b22</p>	<h4>Security</h4> <p>VPC security groups DB-SG (sg-0d0c8c9cc4d69de80)</p> <p>Active</p> <p>Publicly accessible No</p> <p>Certificate authority rds-ca-rsa2048-g1</p> <p>Certificate authority date May 26, 2061, 05:04 (UTC+05:30)</p>
-----------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

24. Return to the browser tab with the inventory application, and enter the following values:

- For **Endpoint**, paste the endpoint you copied earlier.
- For **Database**, enter `inventory`
- For **Username**, enter `admin`
- For **Password**, enter `lab-password`
- Choose **Save**.

25. The application will now Save this information into **AWS Secrets Manager** and connect to the database, load some initial data, and display information.

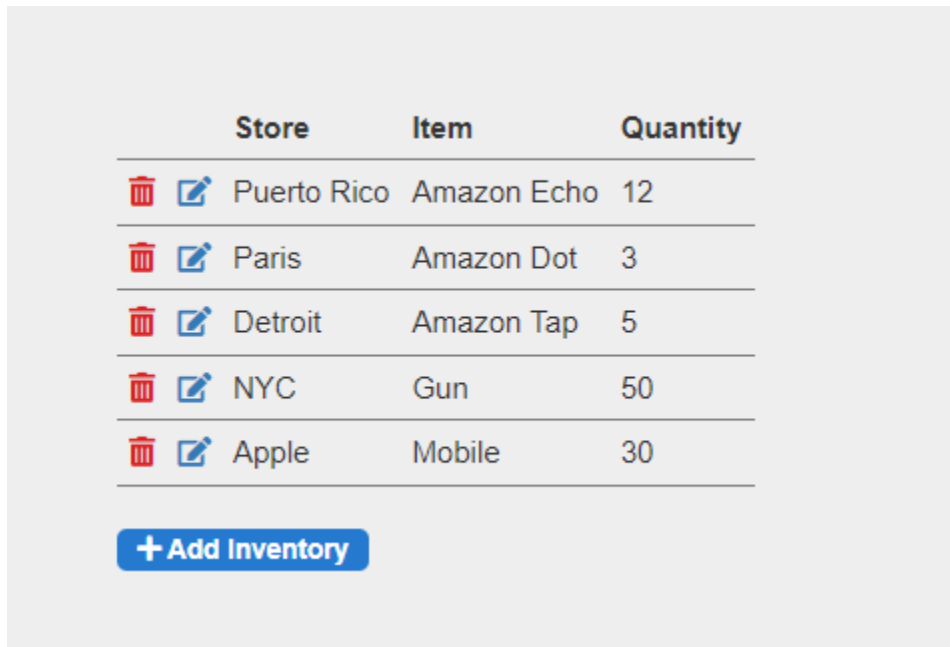
Endpoint	inventory-db.cid6gucjthwr.us-east-1.rds.amazonaws.com
Database	inventory
Username	admin
Password	lab-password
	<input type="button" value="Save"/>

26. You can use the web application to Add inventory, edit, and delete inventory information.











The inventory information is stored in the Amazon RDS MySQL database that you created earlier in the lab. This means that any failure in the application server will not


lead to loss of any data. It also means that multiple application servers can access the same data.

27. Insert new records into the table. Ensure that the table has **5** or more inventory records.



The screenshot shows a web application interface for an inventory system. It features a table with three columns: 'Store', 'Item', and 'Quantity'. There are five rows of data, each with a red trash icon and a blue edit icon to its left. Below the table is a blue button with a white plus sign and the text '+ Add Inventory'.

	Store	Item	Quantity
 	Puerto Rico	Amazon Echo	12
 	Paris	Amazon Dot	3
 	Detroit	Amazon Tap	5
 	NYC	Gun	50
 	Apple	Mobile	30

 Add Inventory

You have now successfully launched the application and connected it to the database.

Optional: To access the saved parameters, go to the AWS Management console. On the **Services** menu, choose **Secrets Manager** , choose **Secrets**.

Task 3: Monitoring the Database Instance

Monitoring is an important part of maintaining the reliability, availability, and performance of any database. Amazon RDS service provides many useful metrics to monitor the health of your database instance. In this task you will explore few useful metrics for the database instance you created.

28. Return to the AWS Management Console.

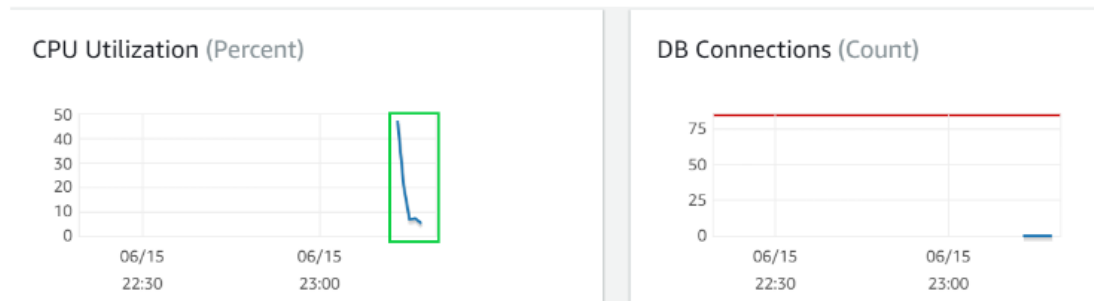
29. On the **Services** menu , choose **RDS**.

30. Choose **Databases**.

31. Choose 'inventory-db'.

32. In the pane below, choose **Monitoring** tab.

33. Observe the CloudWatch metrics indicating respective database Instance parameters as shown in example below.



34. Perform various operations on the web application like add, update or remove records from inventory database and observe the changes in the values mentioned above.
35. Scroll down further to observe other available metrics.

RDS > Databases > inventory-db

inventory-db

[Refresh](#) [Modify](#) [Actions](#)

Summary

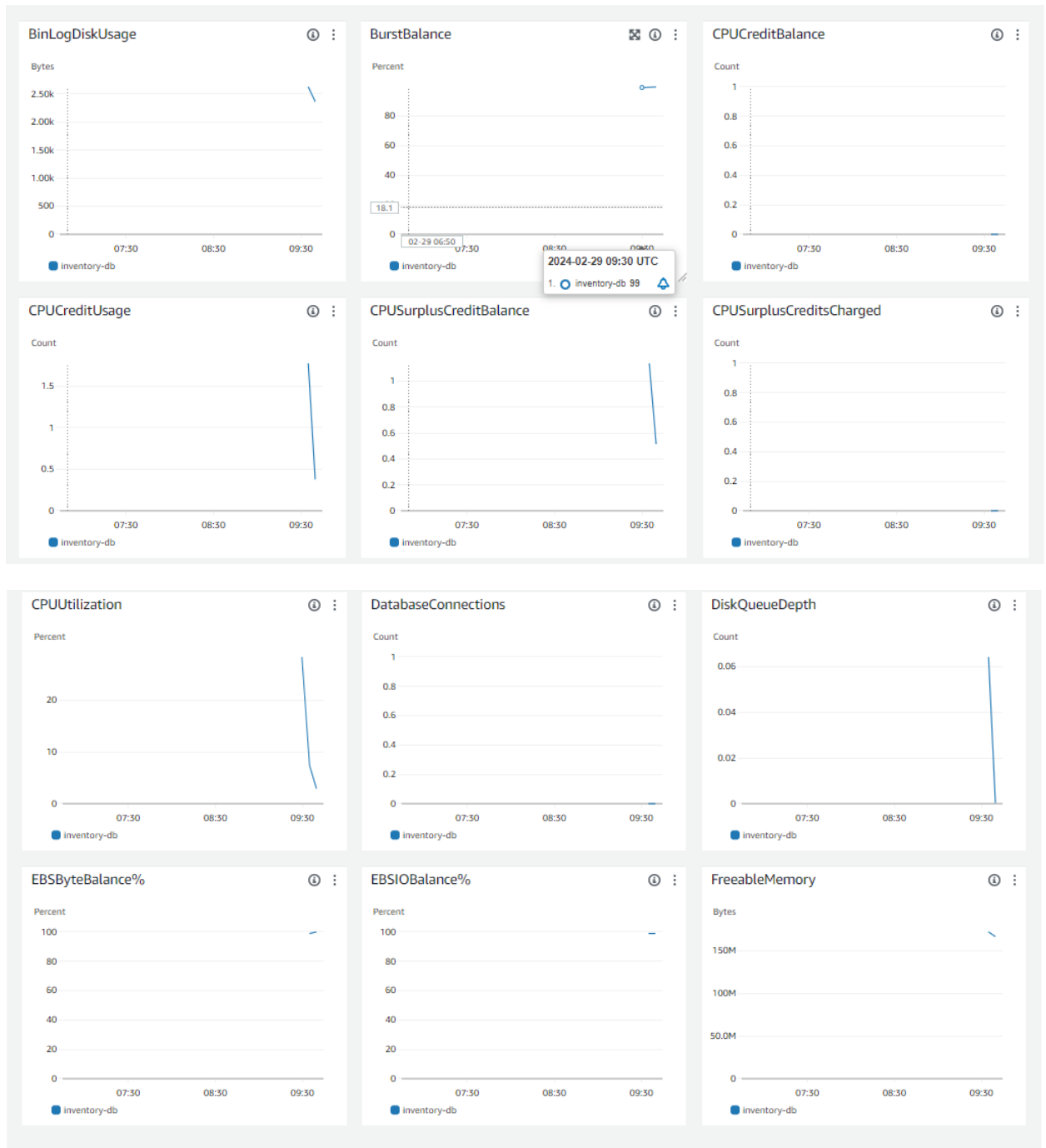
DB identifier inventory-db	Status Available	Role Instance	Engine MySQL Community	Recommendations
CPU 3.06%	Class db.t3.micro	Current activity 0 Connections	Region & AZ us-east-1b	

Connectivity & security | **Monitoring** | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Tags | Recommendations

New monitoring view is available
RDS now supports a new monitoring view which includes Performance Insights and CloudWatch metrics. To access the new monitoring view, select **Modify** to modify your database and turn on Performance Insights. [Modify](#)

CloudWatch (25) Period: 5 minutes Add instance to compare Monitoring

< 1 2 3 >



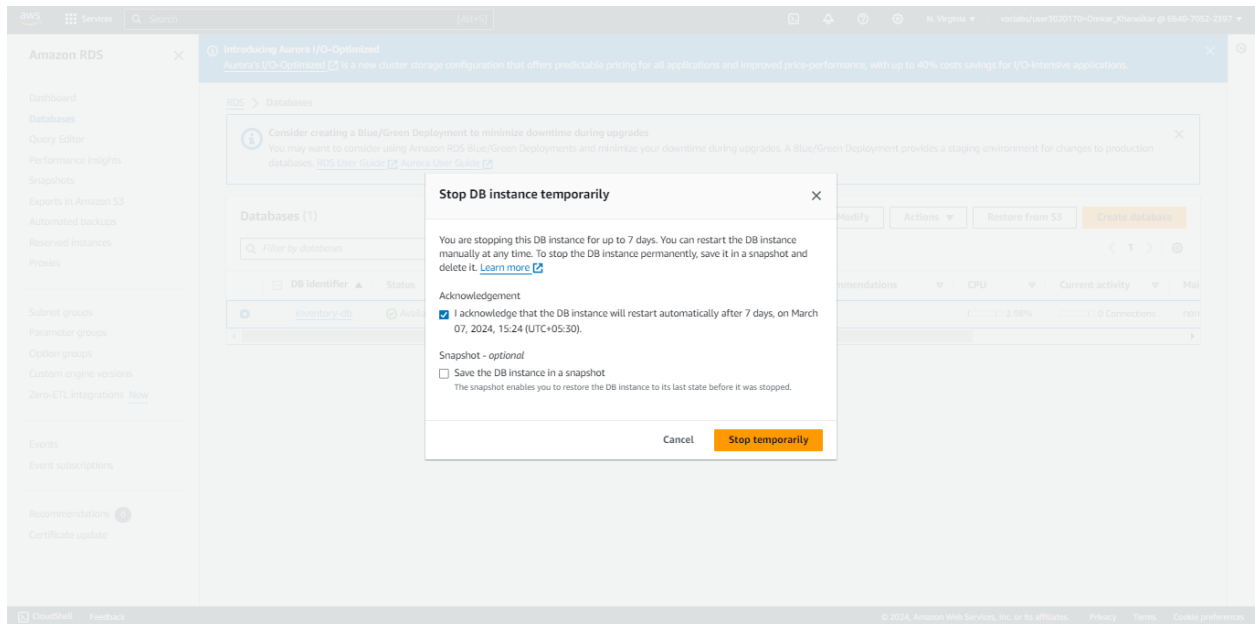
Task 4: Performing operations on Database.

In this task, you learn about a few administrative tasks that can be performed on the database in Amazon RDS.

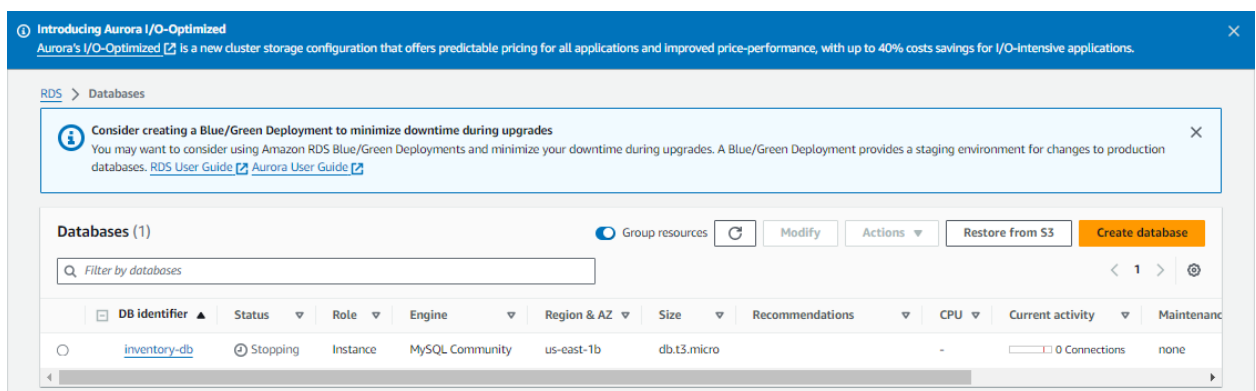
36. Return to the RDS Management Console (if you have navigated out)
37. Choose **Databases**.

38. Choose 'inventory-db'.
39. Under **Actions** menu, there are various operations to perform e.g. **Stop temporarily**, **Reboot**, etc.
40. Choose **Stop temporarily**, to stop the instance temporarily. (Database automatically restarts after 7 days)
 - a. Select checkbox under 'Acknowledgement'
 - b. Choose **Stop temporarily**

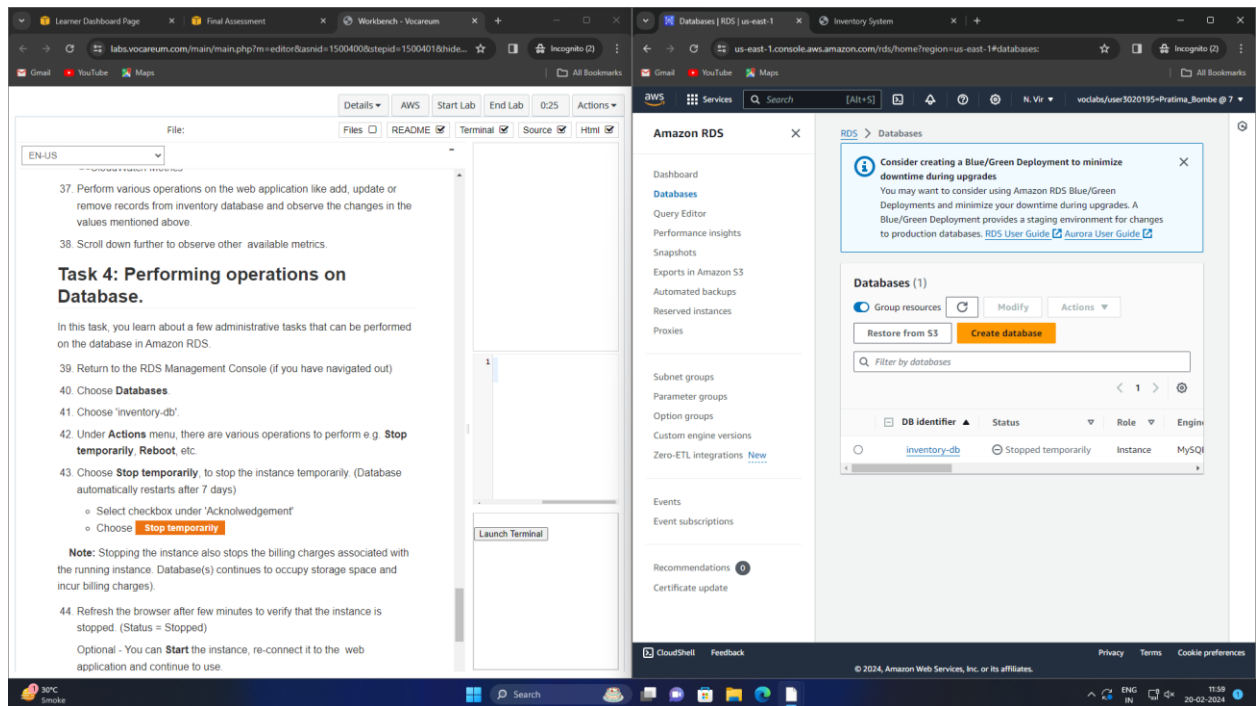
Note: Stopping the instance also stops the billing charges associated with the running instance. Database(s) continues to occupy storage space and incur billing charges).



41. Refresh the browser after few minutes to verify that the instance is stopped. (Status = Stopped)
- Optional - You can **Start** the instance, re-connect it to the web application and continue to use.



In this lab you launched MYSQL RDS database instance, configured an existing web application to interact with the database instance. Then you performed basic tasks such as querying, updating records and monitored the various metrics to gain insights into the health of the database and finally performed basic database administrative operations.



Conclusion: In conclusion, the endeavor to study and implement Database as a Service (DBaaS) on SQL/NOSQL databases such as AWS RDS, Azure SQL, MongoDB, and Firebase has provided valuable insights into the capabilities and advantages of cloud-native database solutions. Throughout this exploration, we have delved into the foundational principles of DBaaS offerings, understanding their role in simplifying database management tasks while providing scalability, reliability, and security. By leveraging platforms like AWS RDS, Azure SQL, MongoDB Atlas, and Firebase, we have gained practical experience in provisioning, configuring, and managing databases in the cloud, catering to both relational and non-relational data models. Furthermore, the experiment has highlighted the importance of selecting the appropriate database service based on data requirements, performance considerations, and scalability needs. Additionally, understanding database security, data integrity, backup, and recovery mechanisms have been essential aspects of this study. Overall, the study and implementation of DBaaS have equipped us with the knowledge and skills to build resilient, scalable, and cost-effective data solutions, accelerating innovation and driving business growth in the cloud era.