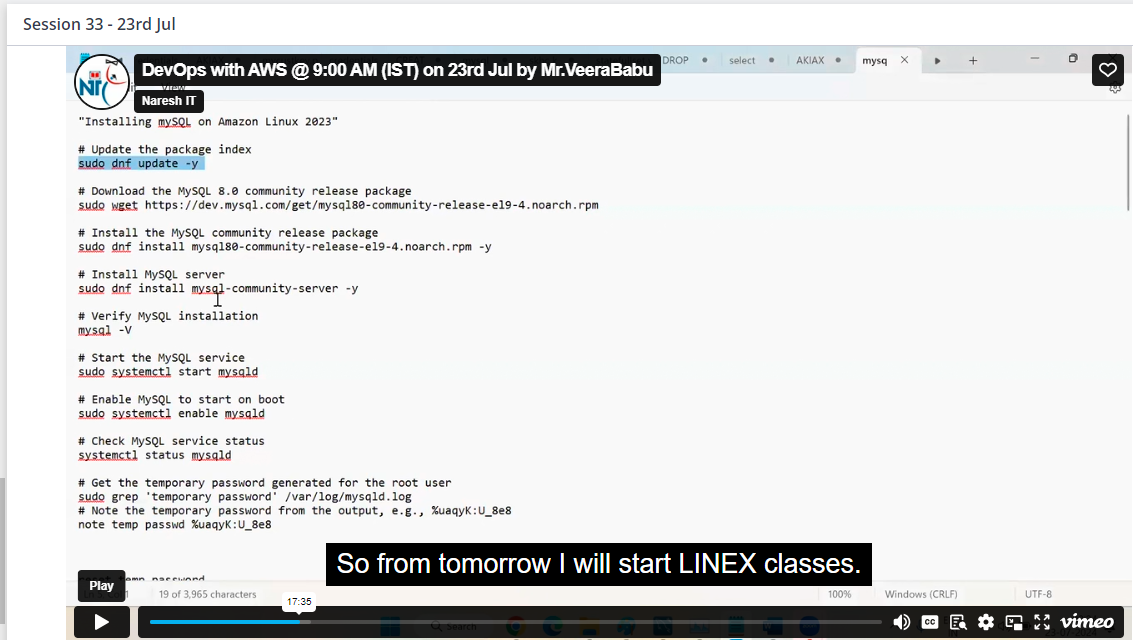
DB on EC2

Install my SQL on ec2 server

Veera(sir) Naresh IT, [7/23/2024 10:44 PM]

mysql -u your\_user -p -h your\_ec2\_public\_ip

Try from ec2------ > MySQL on ec2

"Installing mySQL on Amazon Linux 2023"

# Update the package index

sudo dnf update -y

# Download the MySQL 8.0 community release package

sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-4.noarch.rpm

# Install the MySQL community release package

sudo dnf install mysql80-community-release-el9-4.noarch.rpm -y

# Install MySQL server

sudo dnf install mysql-community-server -y

# Verify MySQL installation

mysql -V

# Start the MySQL service

sudo systemctl start mysqld

# Enable MySQL to start on boot

sudo systemctl enable mysqld

# Check MySQL service status

systemctl status mysqld

# Get the temporary password generated for the root user

sudo grep 'temporary password' /var/log/mysqld.log

# Note the temporary password from the output, e.g., %uaqyK:U\_8e8

note temp passwd %uaqyK:U\_8e8

reset temp password

ALTER USER 'root'@'localhost' IDENTIFIED BY 'Clou&12345';

mysql -u root -p

#create sample data base

CREATE DATABASE IF NOT EXISTS demodb;

SHOW DATABASES;

CREATE TABLE Persons.demodb (

PersonID int,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

);

USE DBName;

give demodb

SHOW TABLES;

#### How to give this db access to external ####

Step 1: Configure MySQL to Allow Remote Connections

Log in to your MySQL server on the primary EC2 instance:

mysql -u root -p

Edit the MySQL configuration file (/etc/my.cnf or /etc/mysql/my.cnf, depending on your installation) to allow connections from any host or a specific IP range. Open the file using a text editor like nano or vim:

sudo nano /etc/my.cnf

Find the bind-address setting and change it to 0.0.0.0 to allow connections from any IP address:

[mysqld]

bind-address = 0.0.0.0

Alternatively, you can specify a particular IP address or range if you want to restrict access:

bind-address = 0.0.0.0

Save the file and restart the MySQL service to apply the changes:

sudo systemctl restart mysqld

Step 2: Grant Remote Access to the MySQL User

Log in to MySQL as the root user:

mysql -u root -p

Grant access to the user for remote connections. Replace demouser, demopassword, and your-remote-ip with the actual username, password, and IP address of the EC2 instance that will be connecting remotely:

GRANT ALL PRIVILEGES ON demodb.\* TO 'demouser'@'your-remote-ip' IDENTIFIED BY 'demopassword';

FLUSH PRIVILEGES;

If you want to allow access from any IP address (less secure), you can use % instead of an IP address:

GRANT ALL PRIVILEGES ON demodb.\* TO 'demouser'@'%' IDENTIFIED BY 'demopassword';

FLUSH PRIVILEGES;

Exit MySQL:

EXIT;

Step 3: Configure Security Groups

Go to the AWS Management Console and navigate to EC2.

Select the security group associated with the EC2 instance running MySQL.

Edit inbound rules to allow traffic on the MySQL port (default is 3306) from the IP address of the other EC2 instance:

Type: Custom TCP Rule

Protocol: TCP

Port Range: 3306

Source: The IP address or security group of the EC2 instance that will be connecting remotely

Save the inbound rule changes.

Step 4: Connect from the Remote EC2 Instance

On the remote EC2 instance, use the mysql command-line client to connect:

mysql -u demouser -p -h primary-ec2-public-dns

Replace primary-ec2-public-dns with the public DNS or IP address of the EC2 instance running MySQL.

Enter the password when prompted.

Summary of Commands

On the MySQL Server (Primary EC2 Instance):

# Edit MySQL configuration file

sudo nano /etc/my.cnf

# Change bind-address to 0.0.0.0

# Restart MySQL service

sudo systemctl restart mysqld

# Log in to MySQL

mysql -u root -p

# Grant access to the user

GRANT ALL PRIVILEGES ON demodb.\* TO 'demouser'@'%' IDENTIFIED BY 'demopassword';

FLUSH PRIVILEGES;

# Exit MySQL

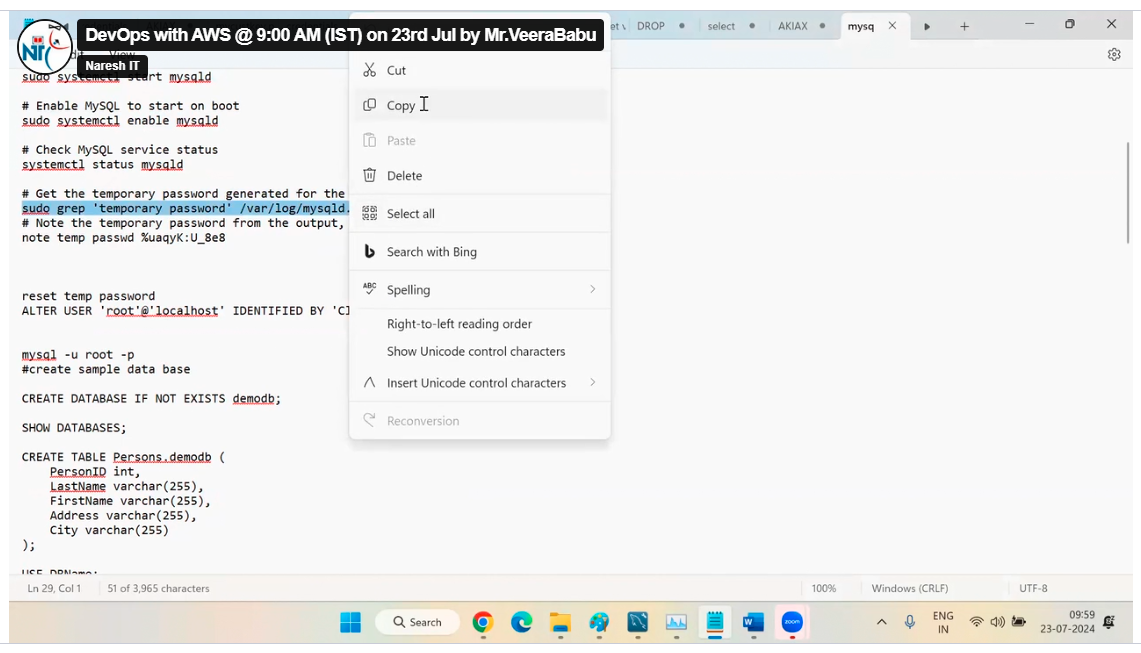
EXIT;

On the Remote EC2 Instance:

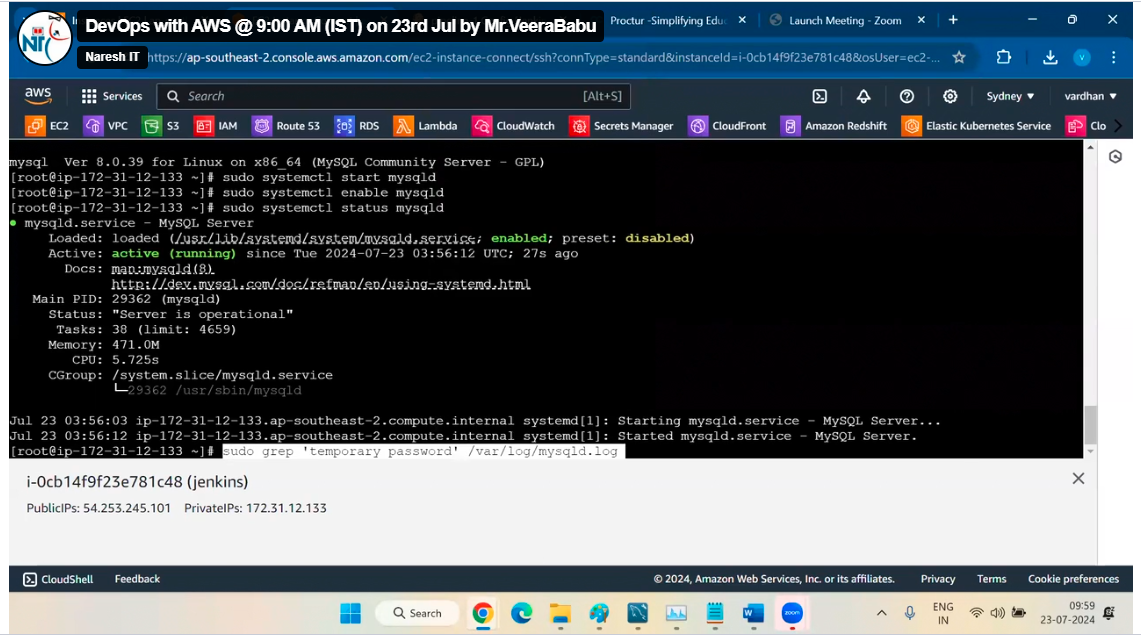
# Connect to the MySQL server

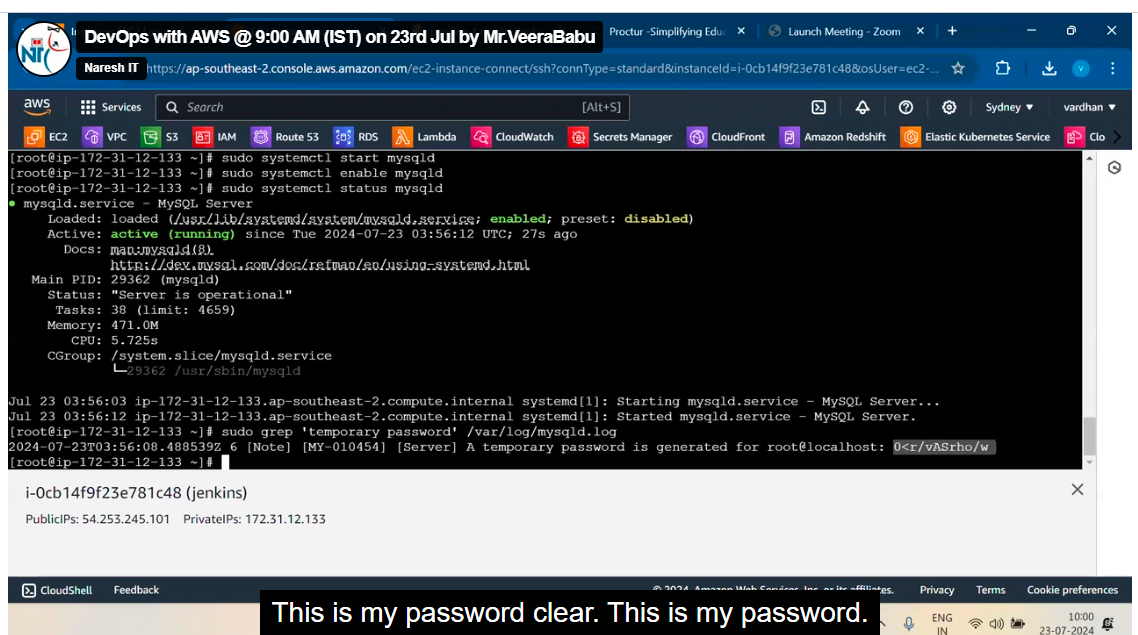
mysql -u demouser -p -h primary-ec2-public-dns

Make sure to adjust security settings carefully to maintain security while allowing necessary access.

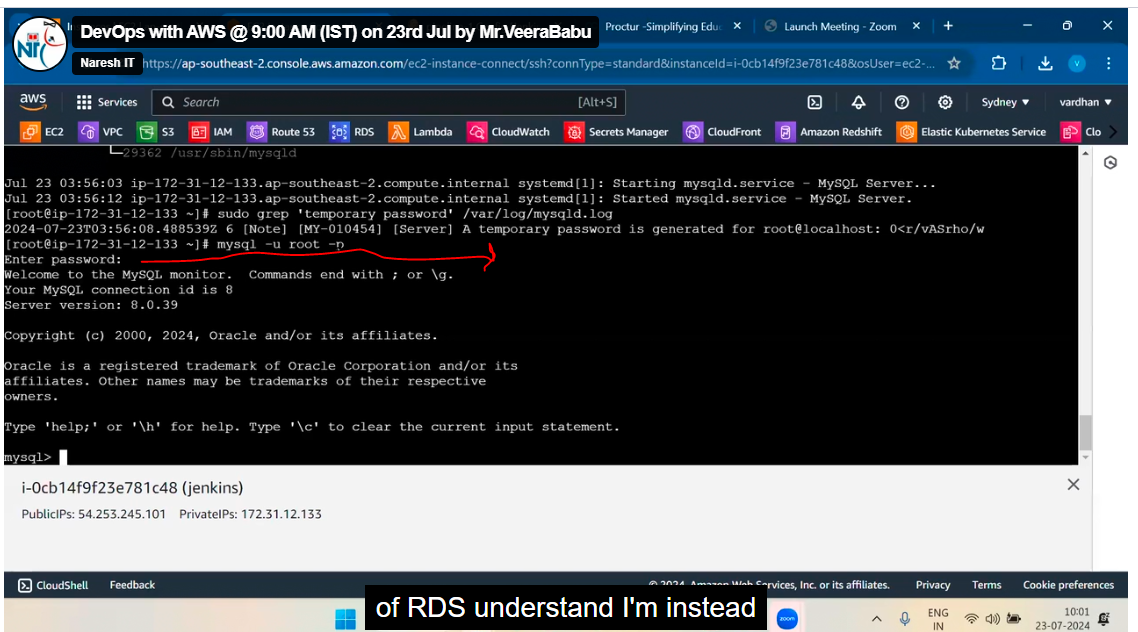


Here we are installed mysql and mysql server





Copy the password

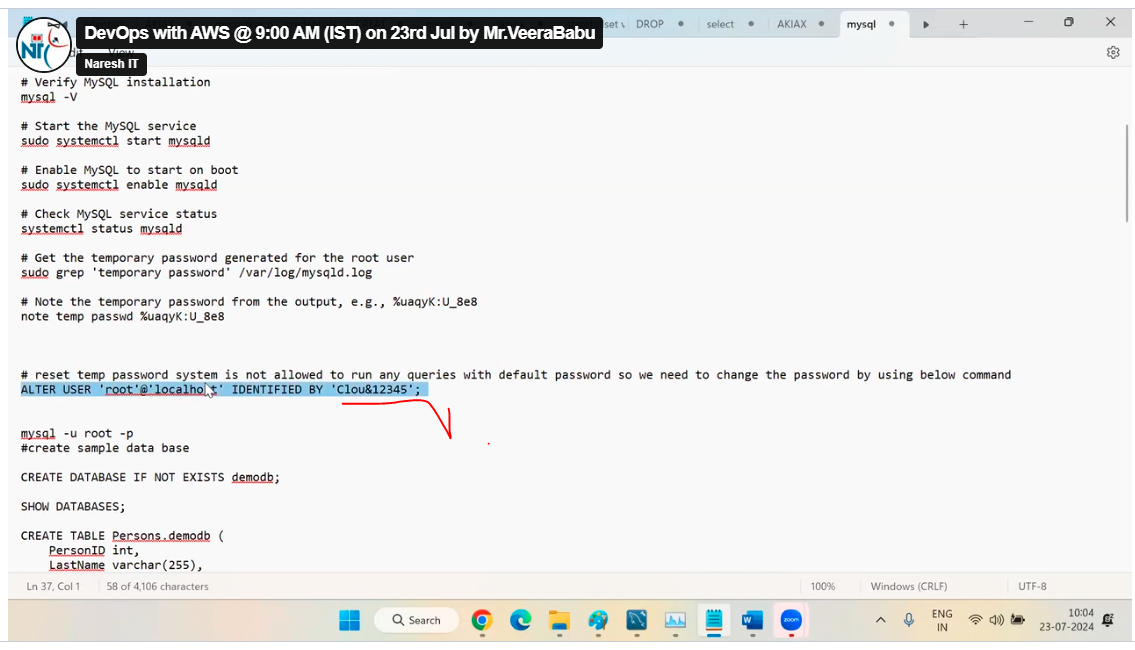


Now we are inside of data base

A screenshot of a computer

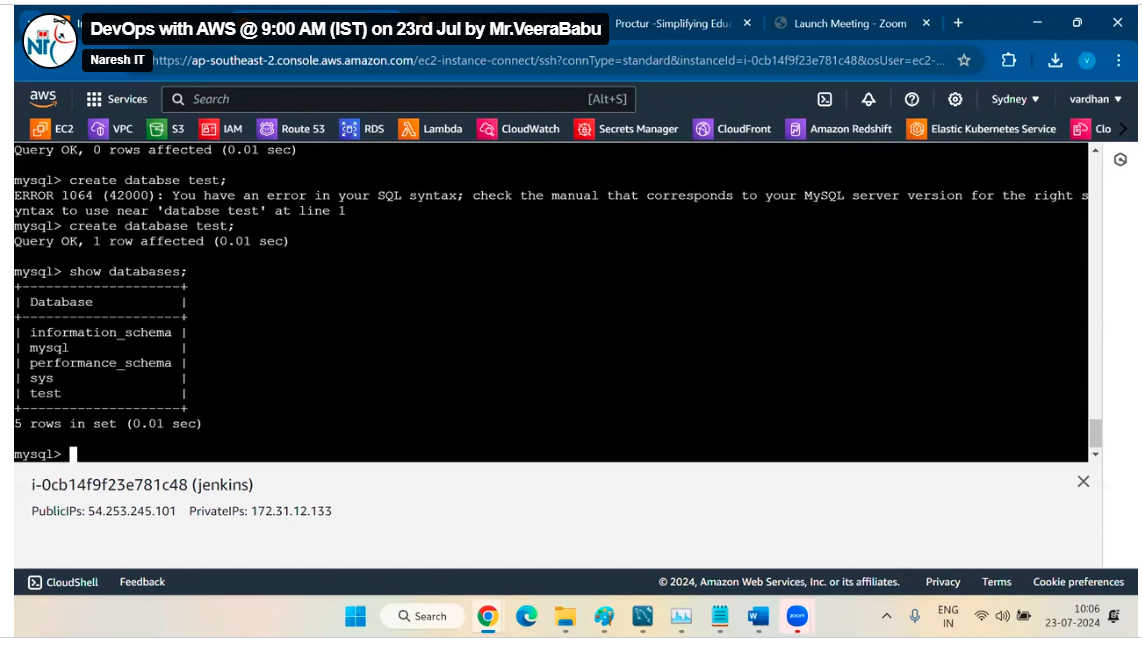
Description automatically generated

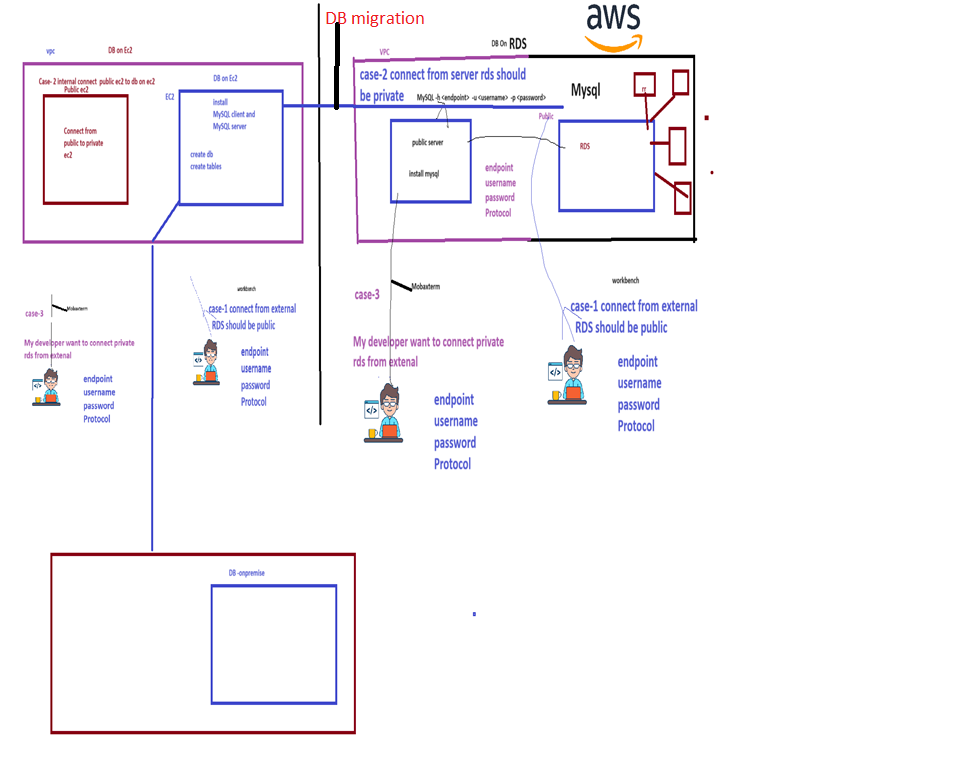
This error because we are not change password, without changing the password we can enter any data in DB



Any password

After that create any file as test

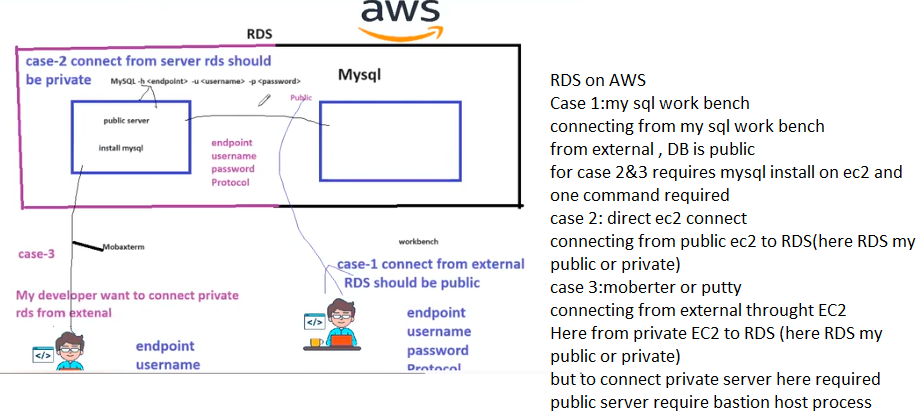




DB migration (here DBS tool required)

In this process we are moving DB from ec2 server to RDS

Here DB on RDS



Amazon Redshift (AWS service)

It is data wear house service, it will support multiple DB engines

It will store the entire data of databases into redshift

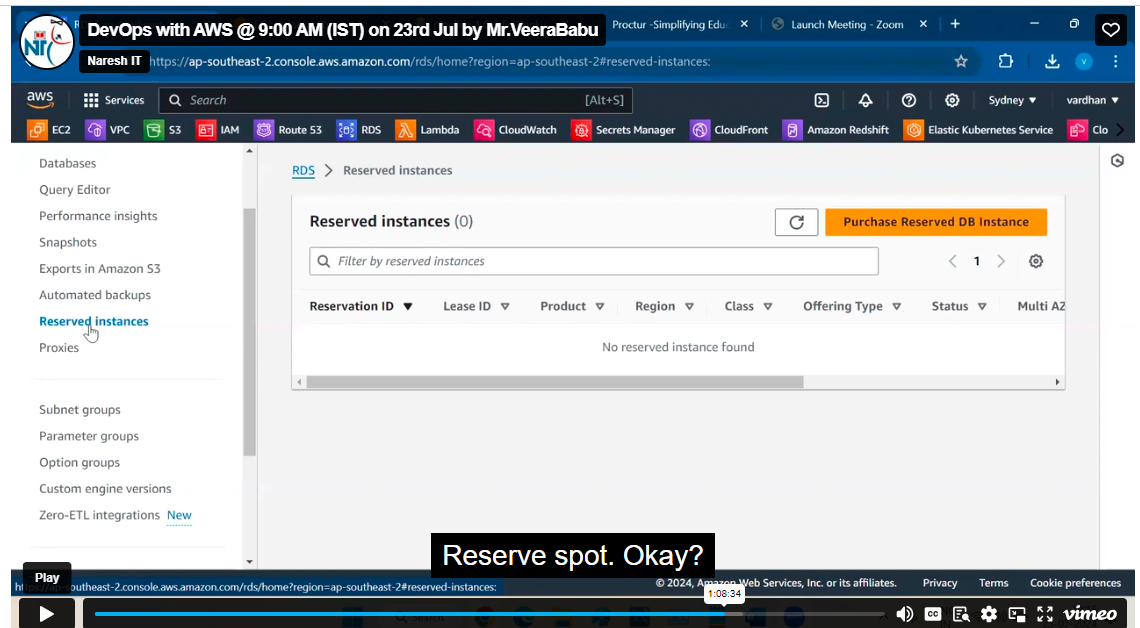
Redshift will support petabyte scale

Public cannot interact directly with red shift, public can interact to RDS, RDS will diverts to redshift

Multiple DB can be created in data base

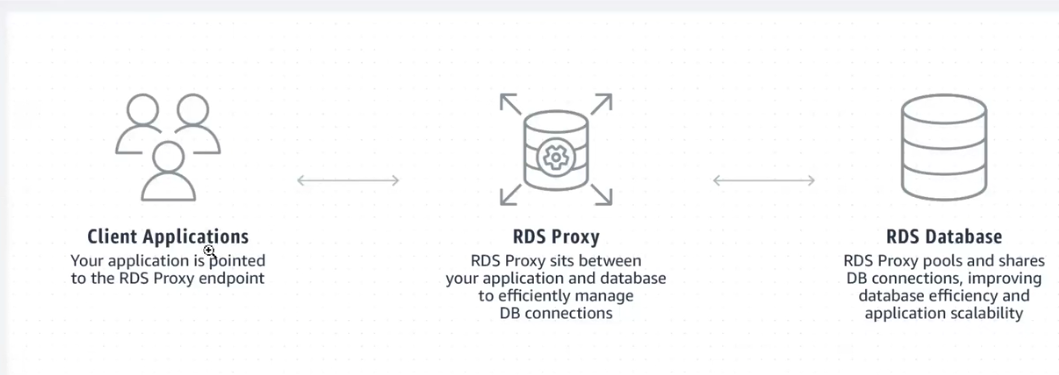
Example we have D-mart which supply to stores and public will go to stores to buy, here Amazon Redshift is like D-Mart, RDS is like stores

Reserved Instances



This is nothing but we are paying money for DB instance for reservation for my be 3 years etc as you required.

Proxices



If connect RDS from external by mySQL work bench the request will go to RDS and in this time the RDS requires some resources(like ram, cpu to execute this query) to connect for example to test credentials like username and password it is taking some resources

To avoid loss of resources we have proxy

RDS proxy will validate your requests

If your credentials are correct then only the poxy will forward your request to RDS other wise no