

Amazon Relational Database Service (Amazon RDS) is a fully-managed web service that makes it easier to set up, operate, and scale a relational database in the cloud.

It allows you to create and use MySQL,MariaDB, PostgreSQL,Oracle and MSSQL databases. This means the code, applications, and tools you already use today with your existing databases can be used with RDS.

**What do you mean by fully-managed?**

Things like infrastructure provisioning, software maintenance, and database admin tasks are taken care of AWS.

AWS RDS doesn’t provide root access/shell access to the RDS instance. In short, RDS doesn’t allow the user to access the underlying host operating system. That means, you cannot login to server OS.

Another limitation is that AWS doesn’t provide option to stop RDS instance for a very long time unlike in EC2.An RDS instance for a single AZ can be stopped maximum 7 consecutive days. After 7 days that instance gets restarted.

You can get high availability with a primary instance and a synchronous secondary instance that you can fail over to when problems occur. You can also use MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.

**How to take backup:**

Just like in EBS you can take manual snapshots of RDS or You can have automated backups. If a manual snapshot is not taken before terminating RDS instance, it prompts you to take final snapshot. Once RDS DB instance is deleted, it cannot be recovered.

**Storage for RDS:**

Amazon RDS uses Amazon Elastic Block Store (Amazon EBS) volumes for database and log storage. The exception is Amazon Aurora, which uses AWS proprietary storage system. Depending on the amount of storage requested, Amazon RDS automatically stripes across multiple Amazon EBS volumes to enhance IOPS performance.

Amazon RDS provides three types of storage with a range of storage and performance options:

1. **General Purpose (SSD)**: General Purpose (SSD), also called gp2, volumes offer cost-effective storage that is ideal for a broad range of workloads. These volumes deliver single-digit millisecond latencies and the ability to burst to 3,000 IOPS for extended periods of time.
2. **Provisioned IOPS** : Provisioned IOPS storage is designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency in random access I/O throughput. You specify the amount of storage you want allocated, and then specify the amount of dedicated IOPS you want.
3. **Magnetic** – Amazon RDS also supports magnetic storage for backward compatibility.

**Charges for Amazon RDS:**

User is charged based on:

**1.Instance class :** Type of instance class(micro, Nano etc.)

**2.Running time :** billed by the instance-hour

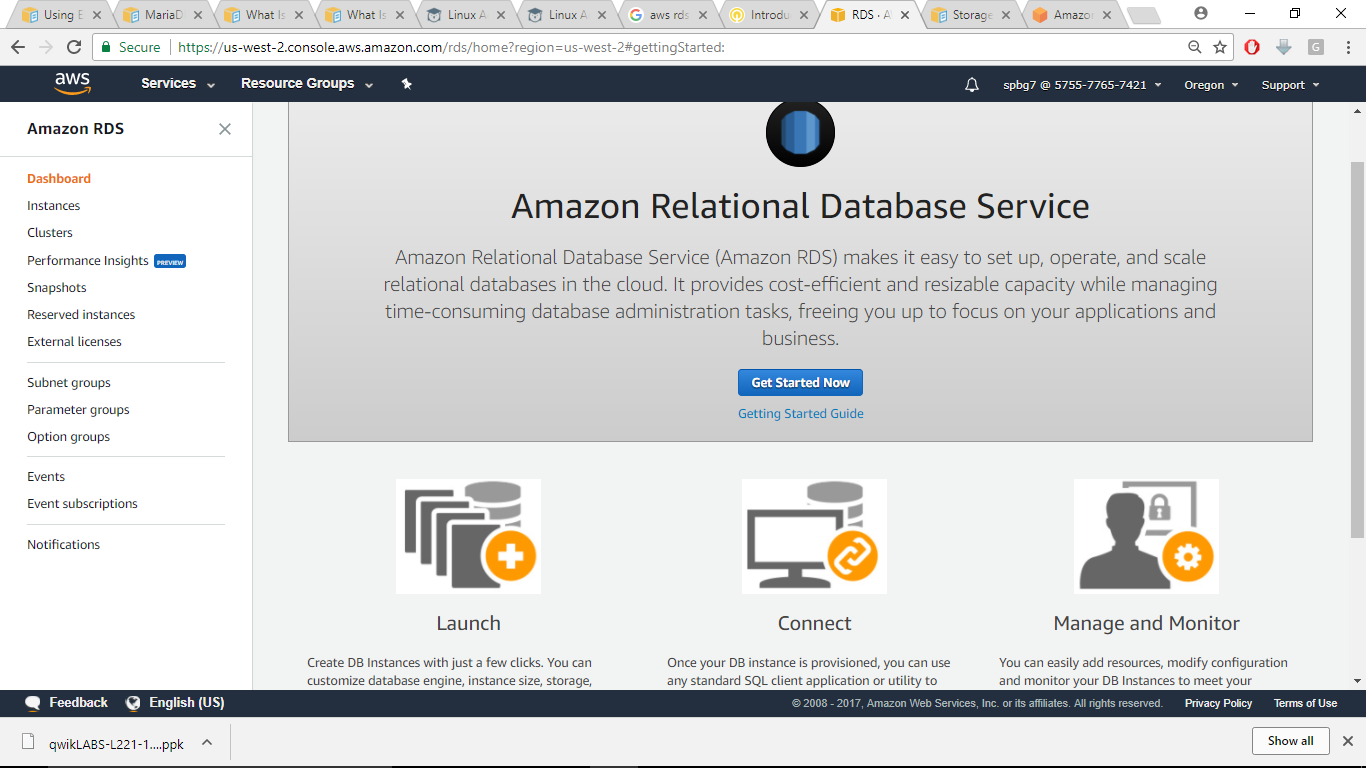
**3.Storage:** The storage capacity that you have provisioned to your DB instance is billed per GB per month.

**4.I/O requests for month**: Total number of storage I/O requests that you have made in a billing cycle.

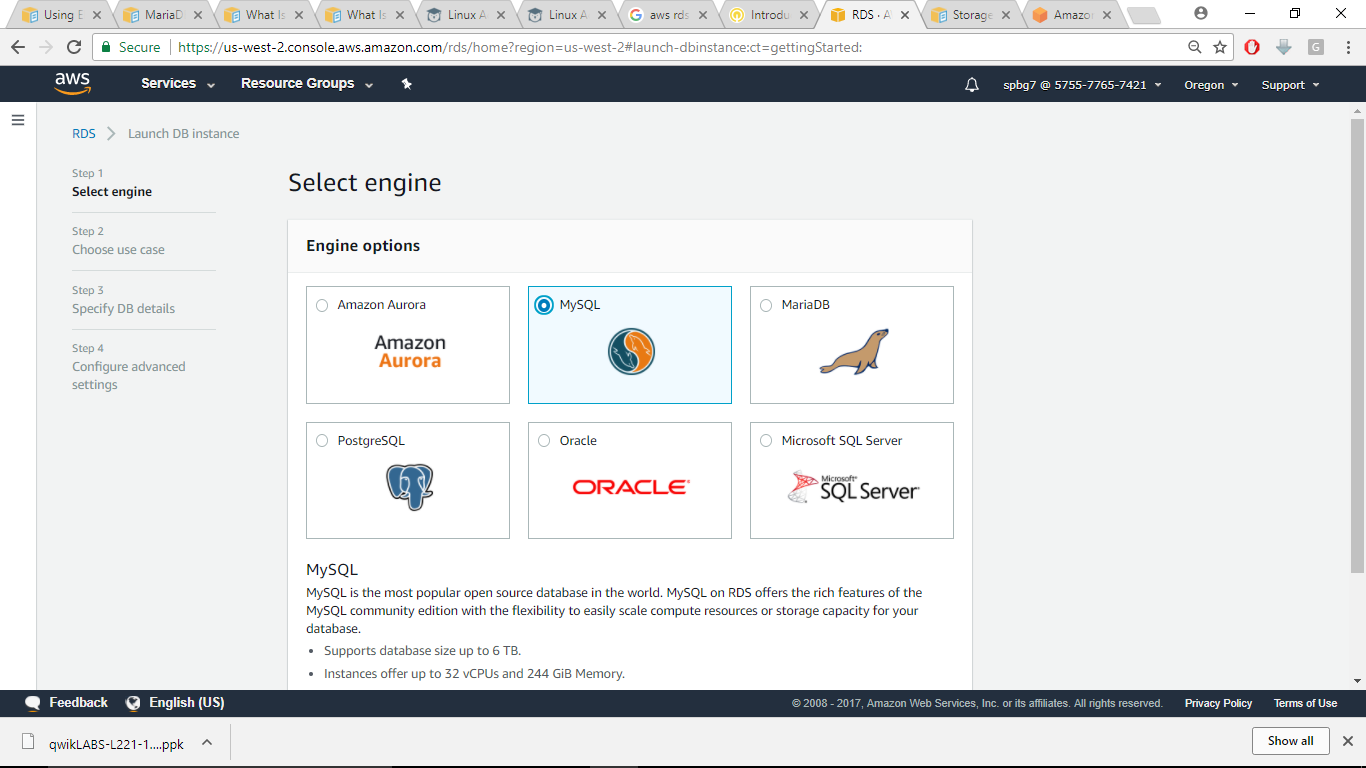
**5.Backup storage:** Backup storage is the storage that is associated with automated database backups and any active database snapshots that you have taken

Lab 1: Creating an RDS instance and connecting to the instance

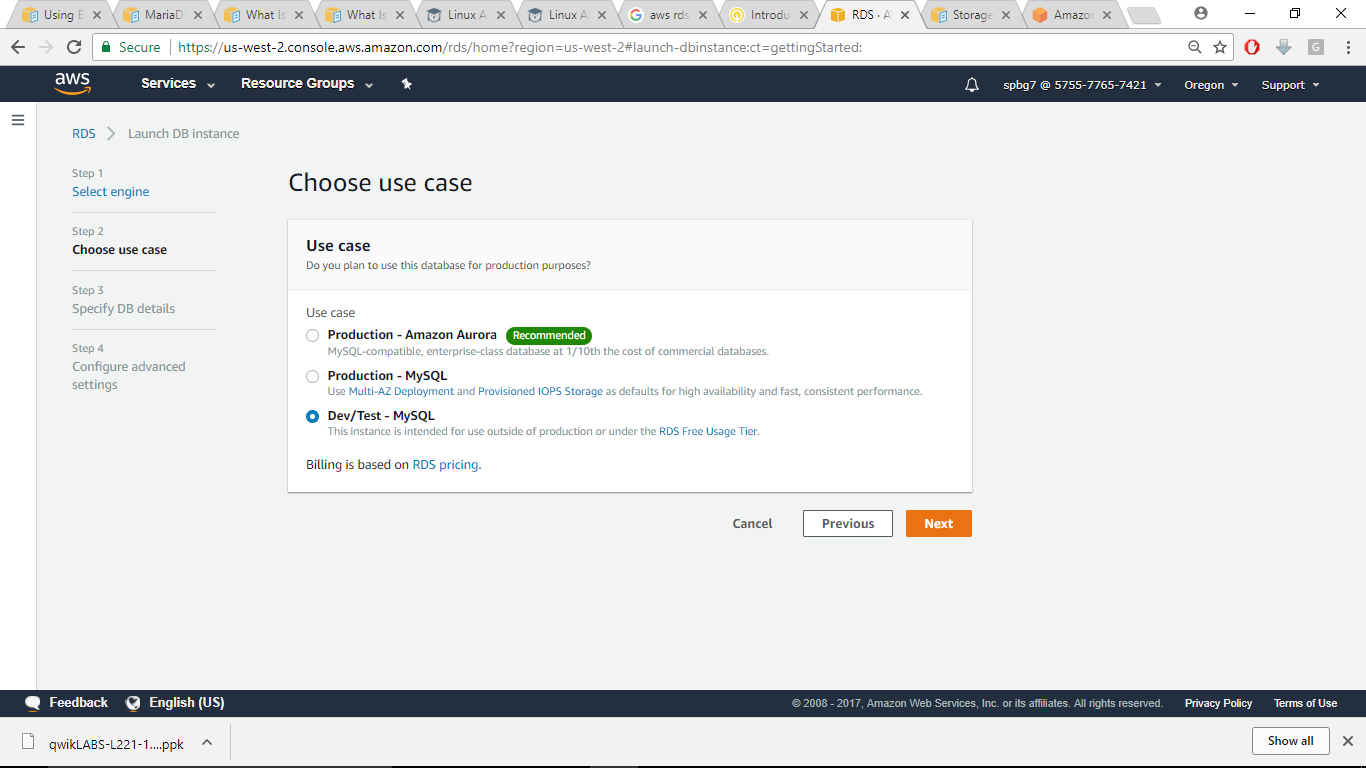
**Step 1: Click RDS on AWS management console and click on get started now**



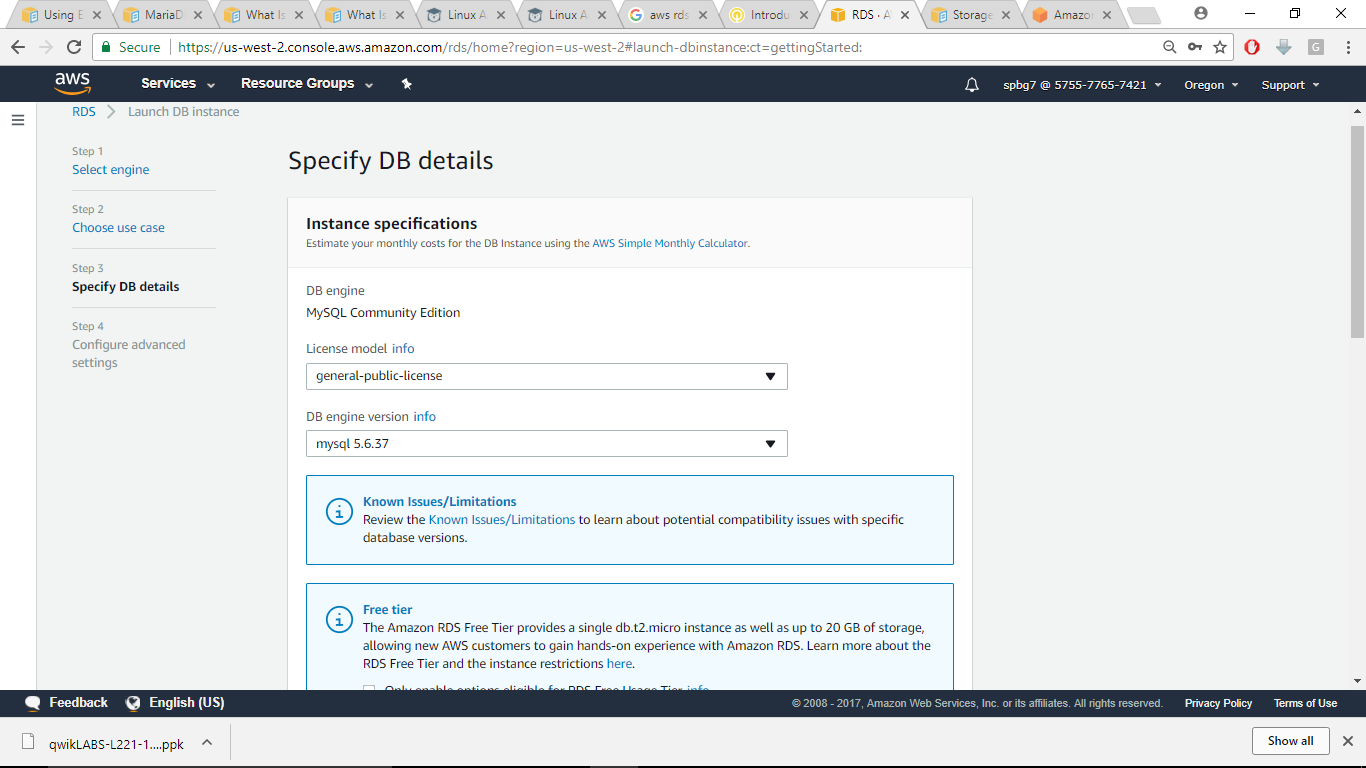
**Step 2: Select the engine as MySQL**



**Step 3: Select dev/test and click next**

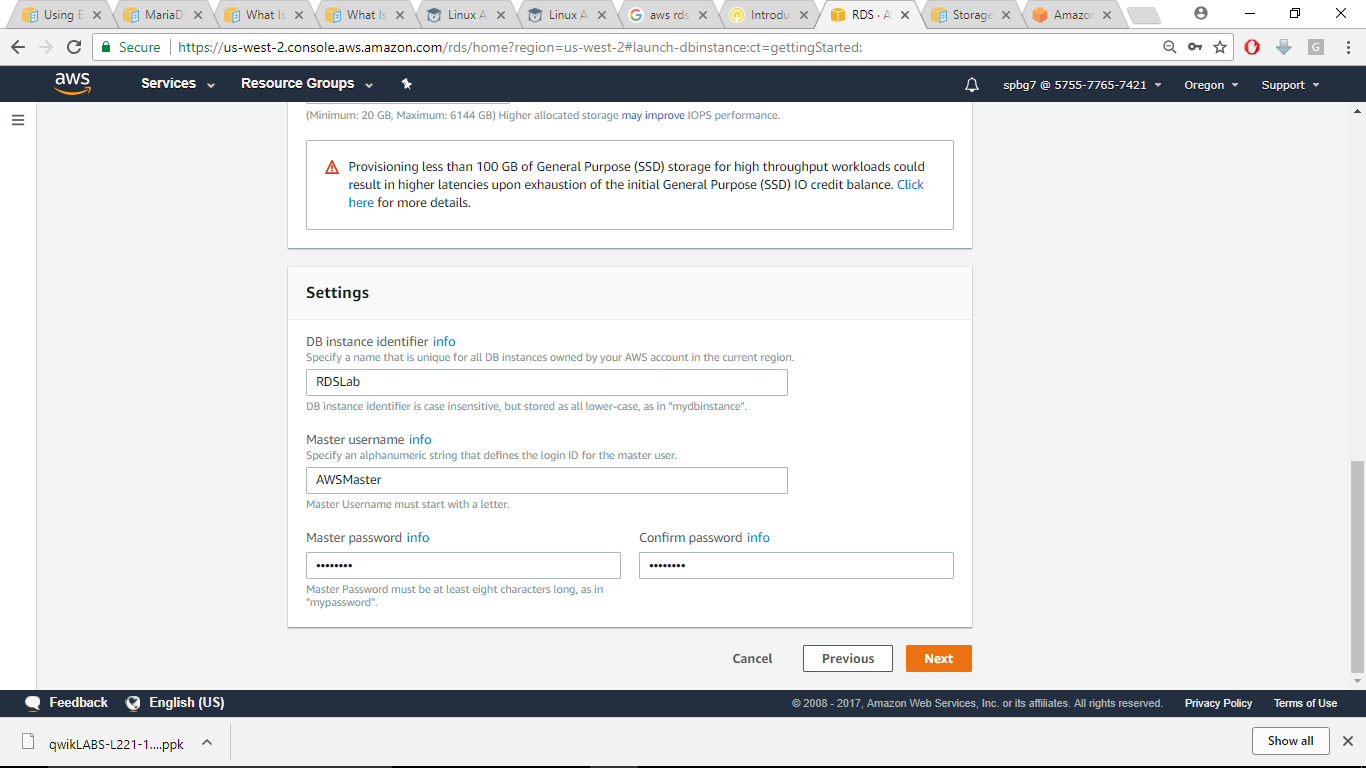


**Step 4: Specify DB instance details as shown:**



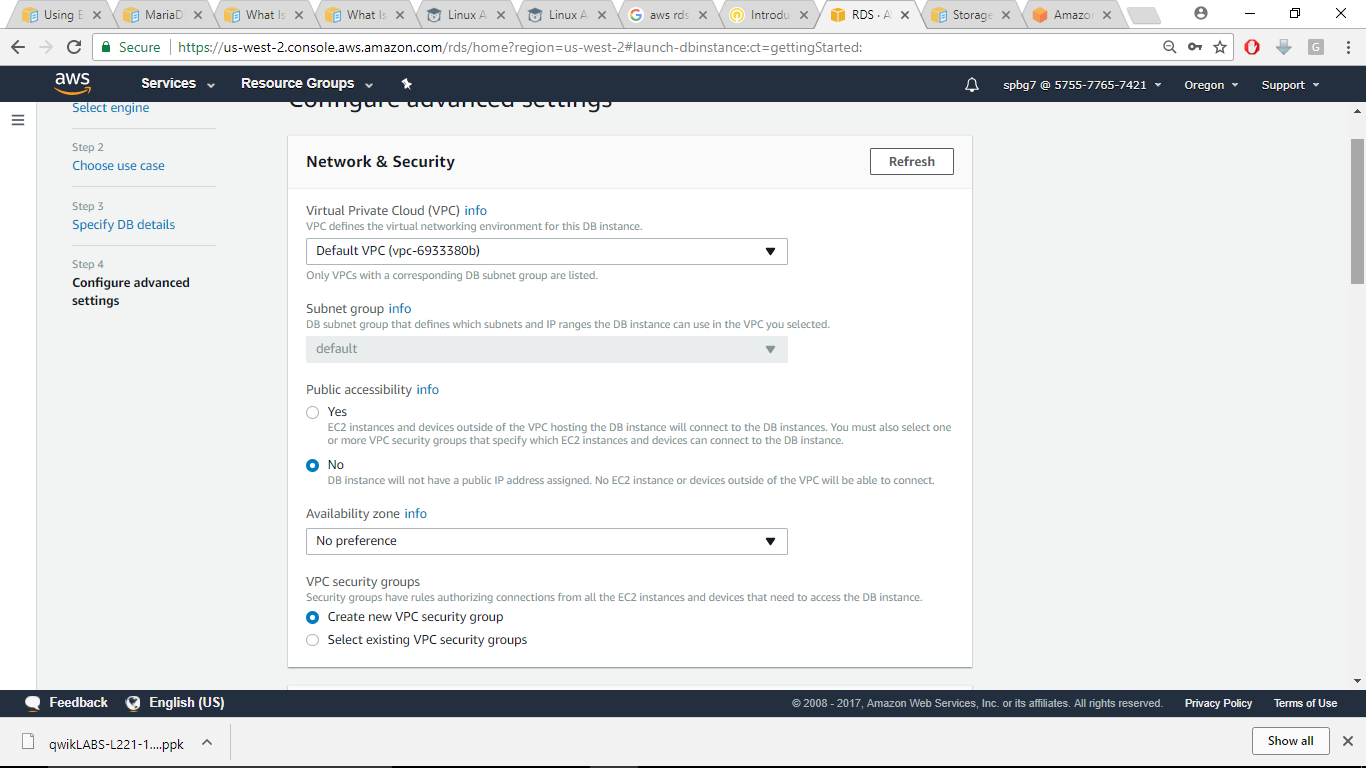


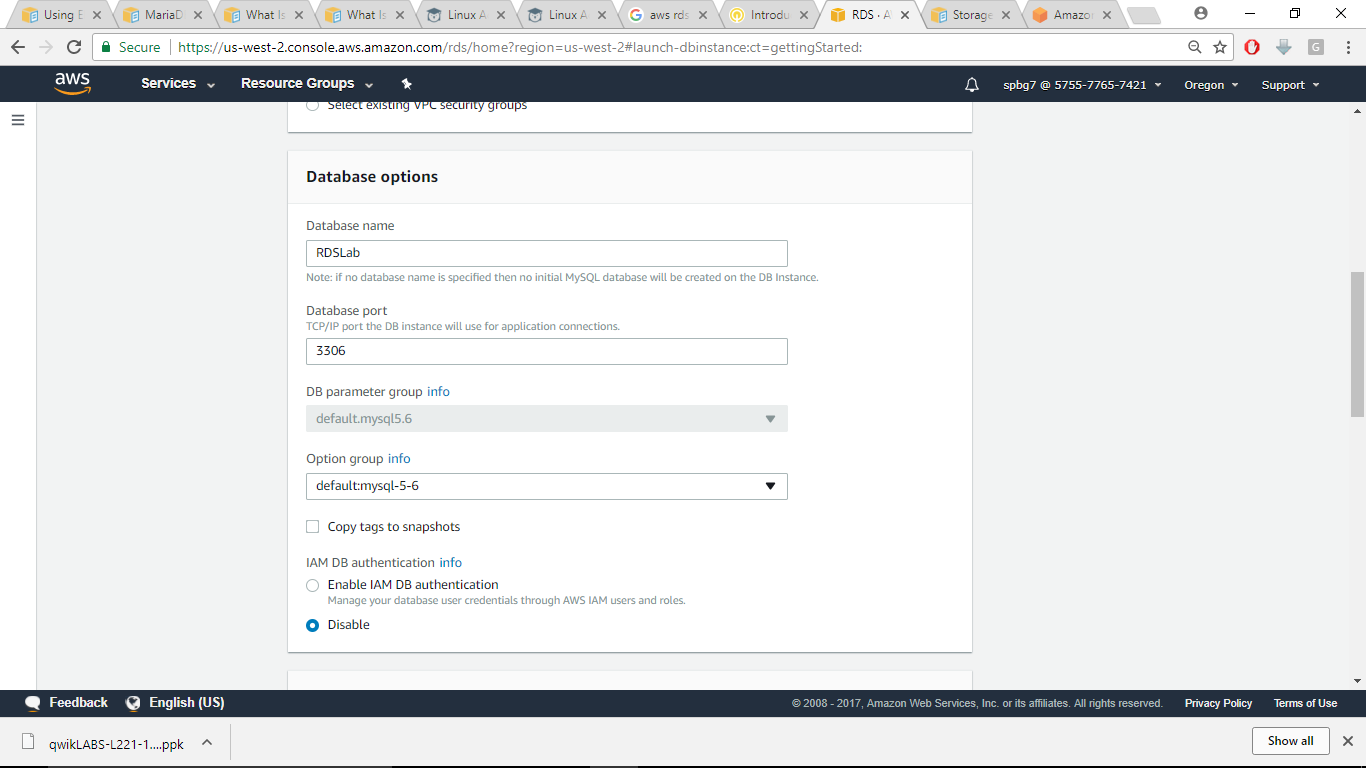
In the Master username and Password provide your own username and password

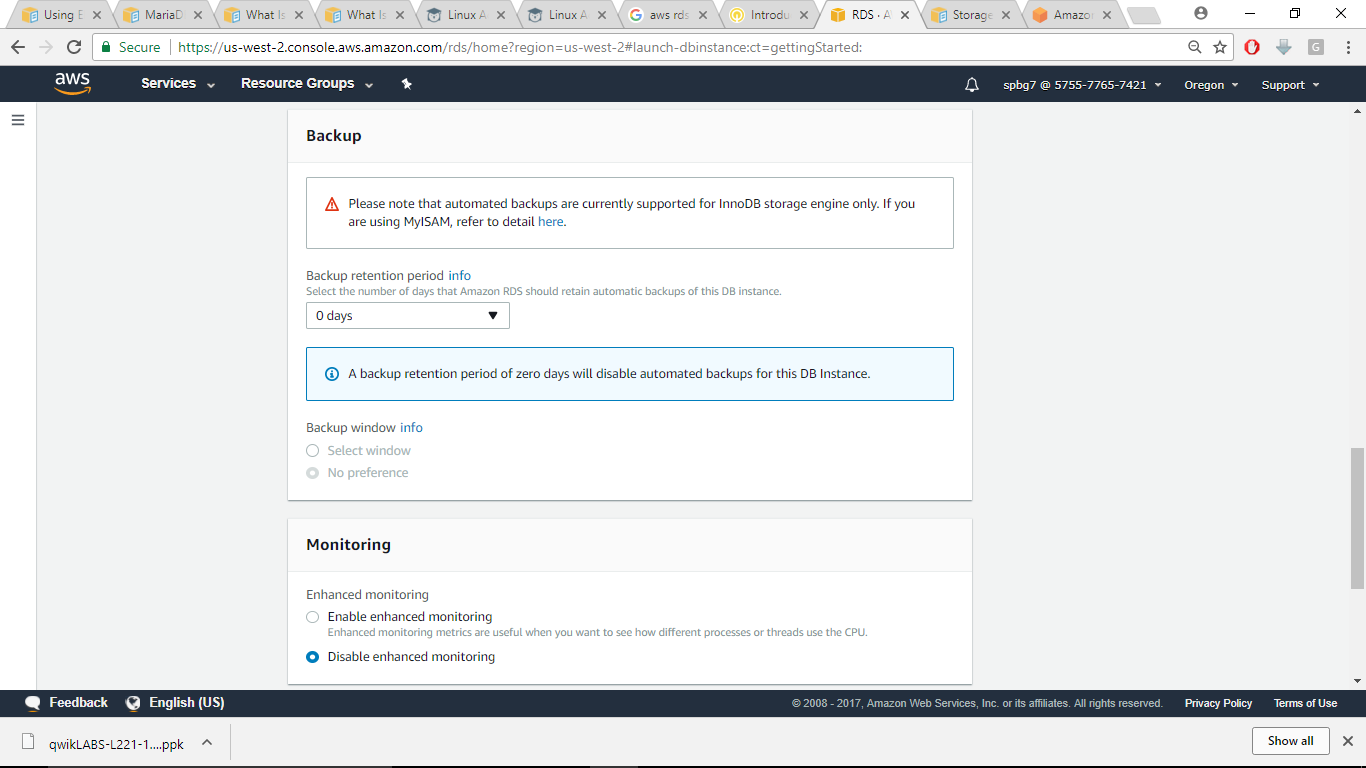


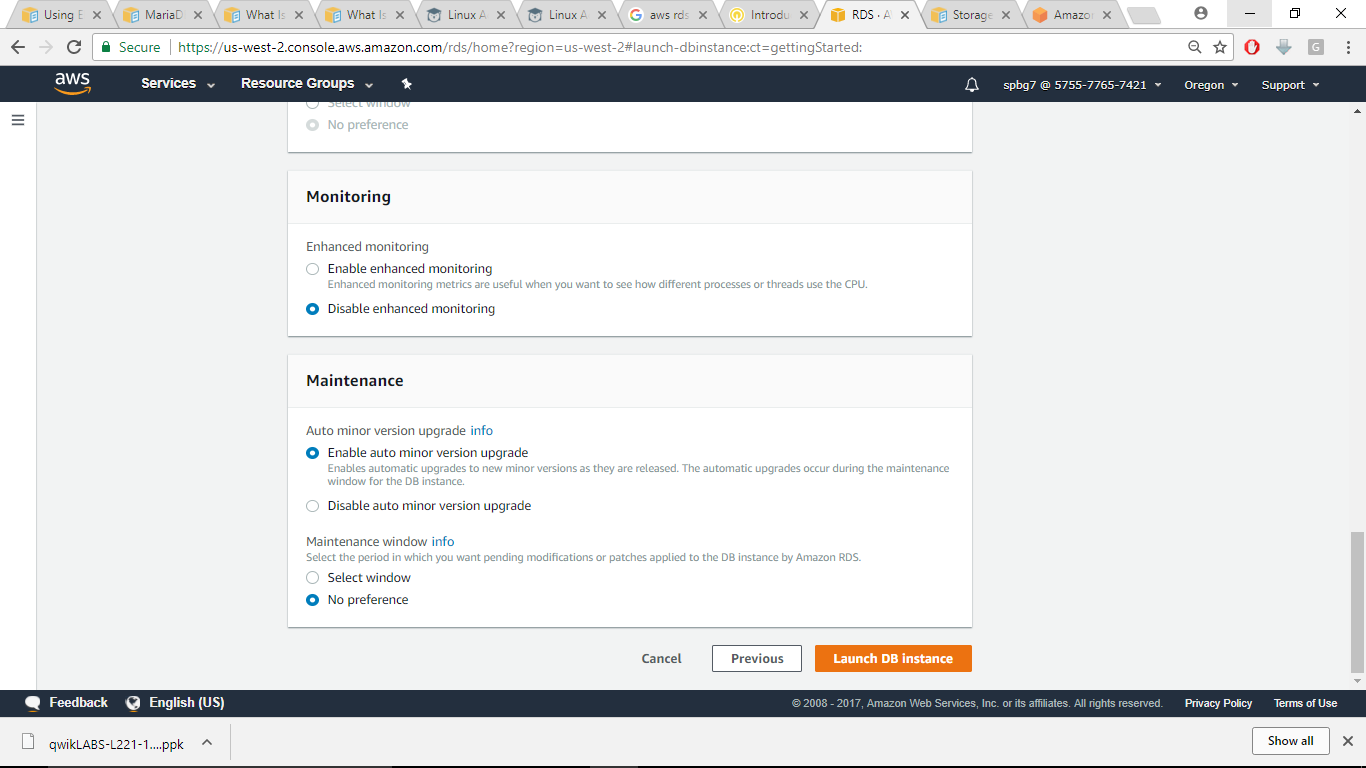
Now click Next.

**Step 5: Configure advanced settings. Set the details as shown in the pics:**





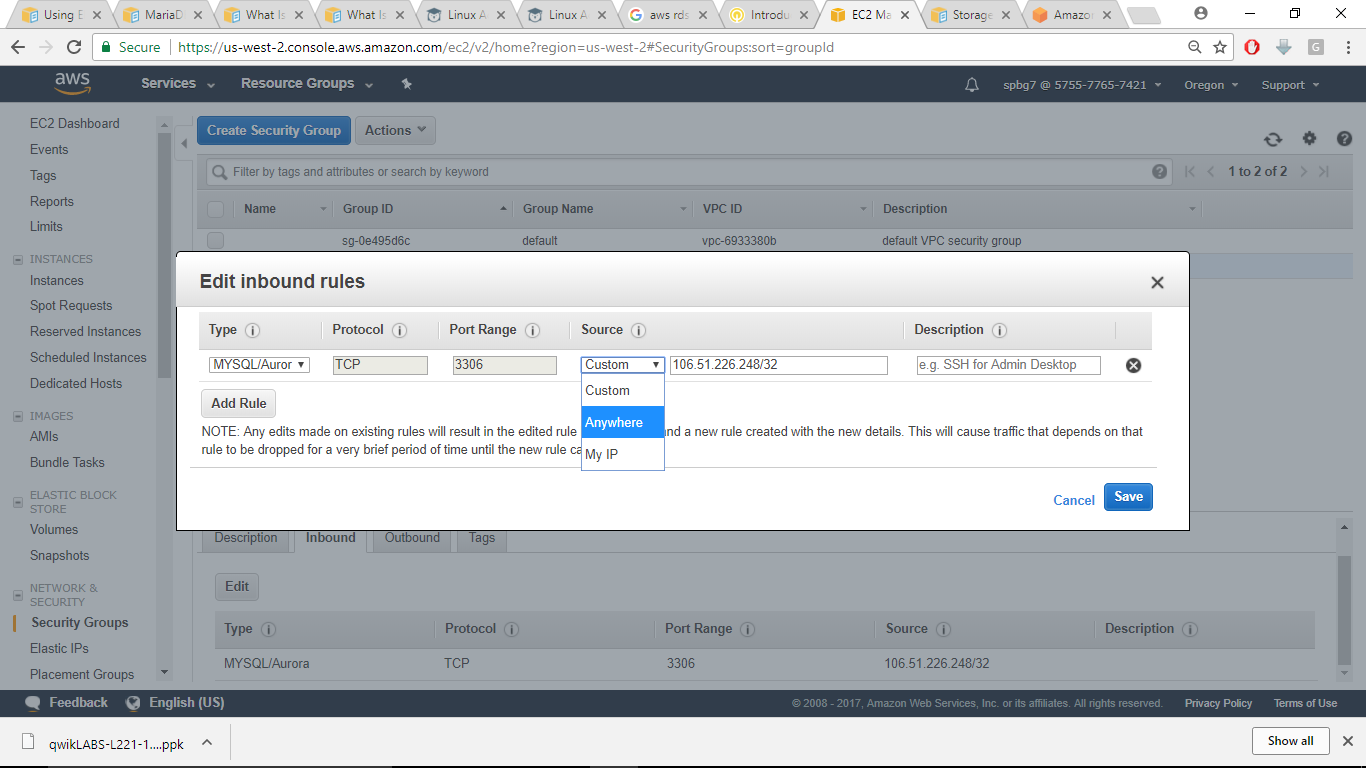




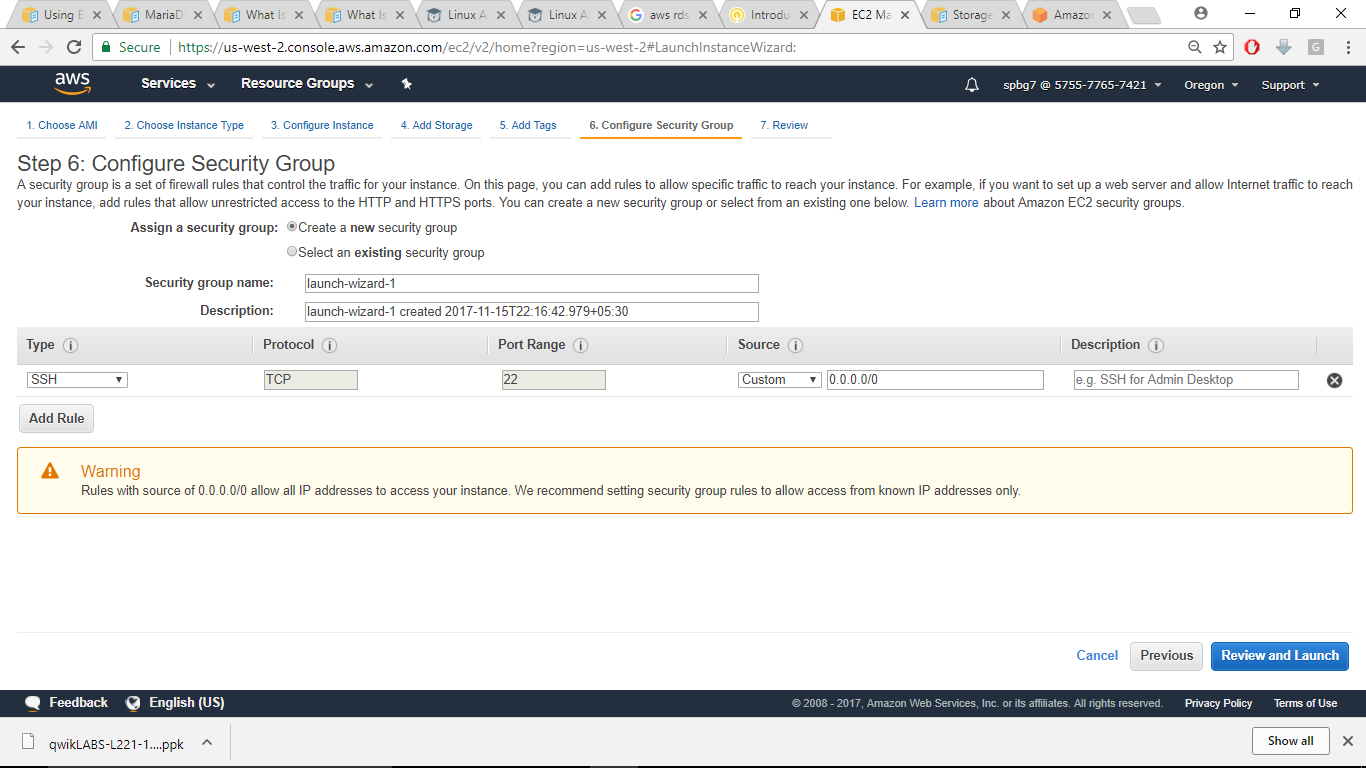
Now click on Launch DB instance

**Step 6: Edit RDS security groups and launch an EC2 instance**

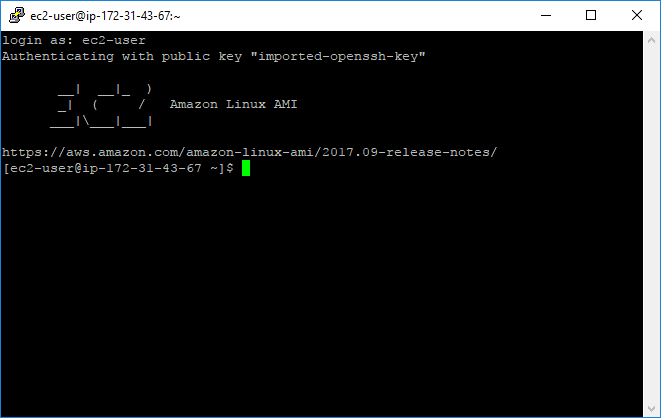
Now go to EC2 dashboard and click on security groups, You’ll notice RDS security group is created. Now edit the inbound rules and change the source to anywhere and click save.



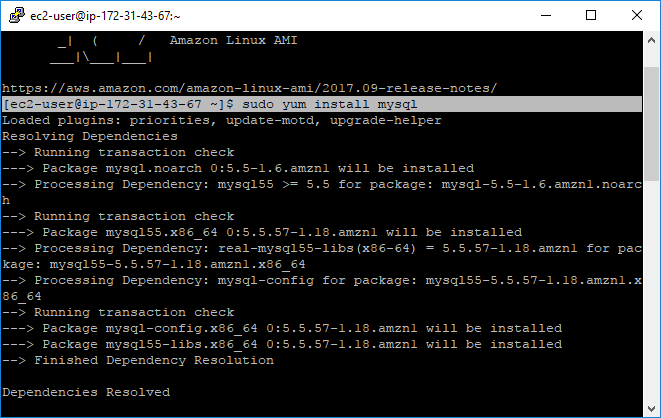
Now create an EC2 t2.micro instance and in the security group just keep this setting only for SSH and then launch the instance.



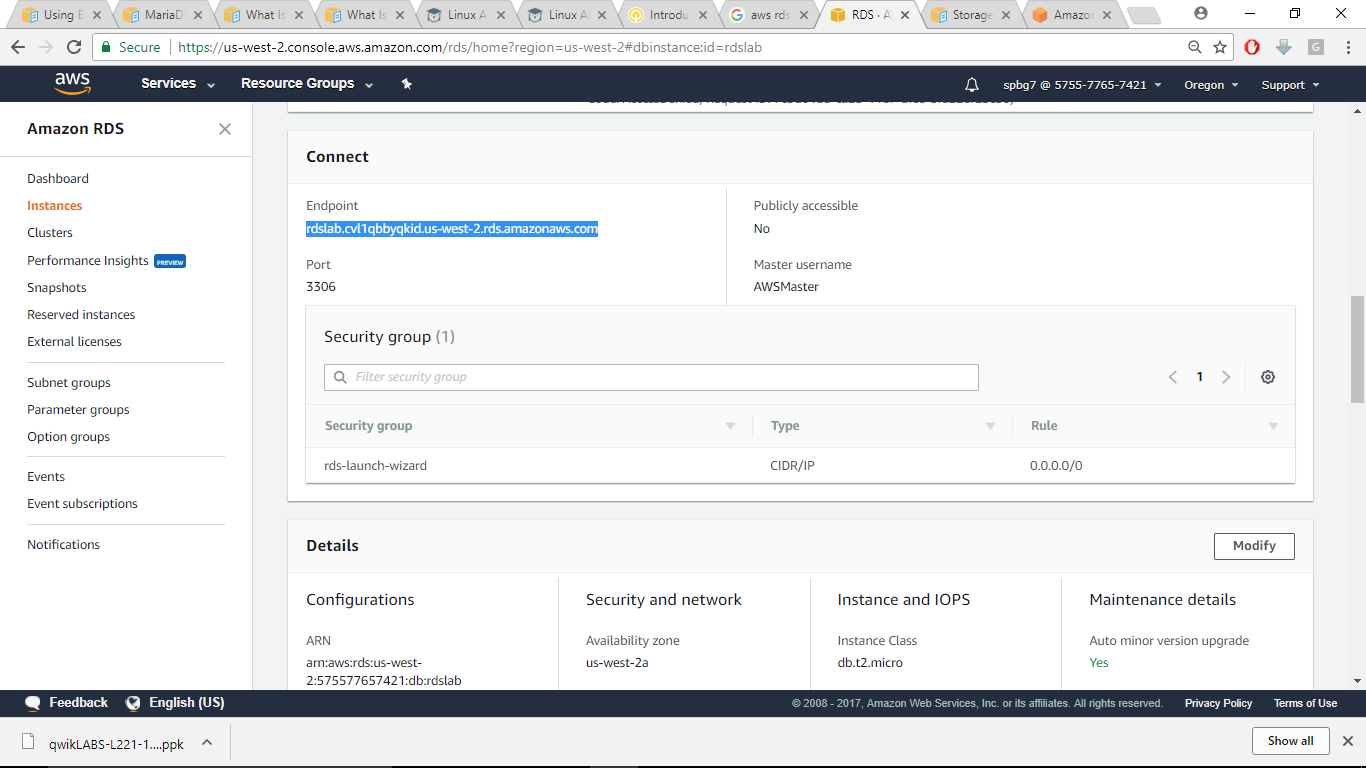
**Step 7: Connect to your instance and then connect to RDS instance**



Now Install mysql on your EC2 instance



Now return to RDS dashboard and select the RDS instance and **copy its endpoint**

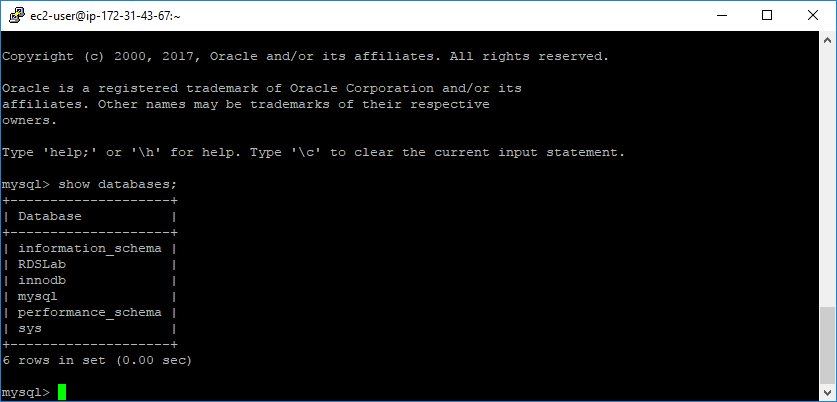


Now switch back to your terminal console and enter the following command

mysql --host endpoint-url --password –user

in the above command replace the endpoint-url with RDS endpoint and click enter. Then, you’re prompted for password and enter it based on what you’ve set.Remove the port number that comes after the endpoint

Now you’re logged into MYSQL console. Execute the following command to verify whether any records return:



**You’ve successfully connected to your database.**

**RDS Components:**

**DB instances:**

A DB instance is an isolated database environment running in the cloud. Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines.

The computation and memory capacity of a DB instance is determined by its DB instance class.(types of DB instances just like EC2 instances:t2,micro,t2.nano etc)

Each RDS engine can create an instance with at least one database in it. Each instance can have multiple user-created databases. Those databases are called DB instance identifier.

You can have up to 40 Amazon RDS DB instances. Of these 40, up to 10 can be Oracle or SQL ServerDB instances under the "License Included" model. All 40 DB instances can be used for MySQL, MariaDB,or PostgreSQL. You can also have 40 DB instances for SQL Server or Oracle under the "BYOL" licensing model

Each DB engine has its own version. The parameters of DB engine is configured using DB Parameter groups. One DB parameter group can be shared among the same instance types of the same DB engine and version.

Amazon RDS creates a master user account for your DB instance as part of the creation process. This master user has permissions to create databases and to perform create, delete, select, update, and insert operations on tables the master user creates. You must set the master user password when you create a DB instance, but you can change it at any time using the Amazon AWS command line tools, Amazon RDS API actions, or the AWS Management Console. You can also change the master user password and manage users using standard SQL commands.

Each DB instance can store data a minimum of 5GB and a maximum of 6TB.The only exception is Microsoft SQL server which supports up to 4 TB.

**Regions and Availability Zones:**

An RDS DB instance can be provisioned in several AZs by selecting Multi-AZ deployment option. When this option is selected, Amazon automatically provisions and maintains a synchronous **standby replica** of the DB instance in a different Availability Zone.

It is advisable to create RDS in multiple AZs for avoiding single points of failures.

Amazon RDS uses several different technologies to provide failover support.

MultiAZdeployments for Oracle, PostgreSQL, MySQL, and MariaDB DB instances use Amazon's **failover technology.**

SQL Server DB instances use **SQL Server Mirroring**.

Amazon Aurora instances stores copies of the data in a DB cluster across multiple Availability Zones in a **single AWS Region**(regardless of whether the instances in the DB cluster span multiple Availability Zones).

RDS synchronized DBs between **primary and secondary instances**. In case a primary instance fails, the load is automatically shifted to a **secondary instance**.In the event of a planned or unplanned outage of your DB instance, Amazon RDS automatically switches to standby replica(secondary instance)**.**

The failover mechanism automatically changes the DNS record of the DB instance to point to the standby DB instance.

**Read Replica:**

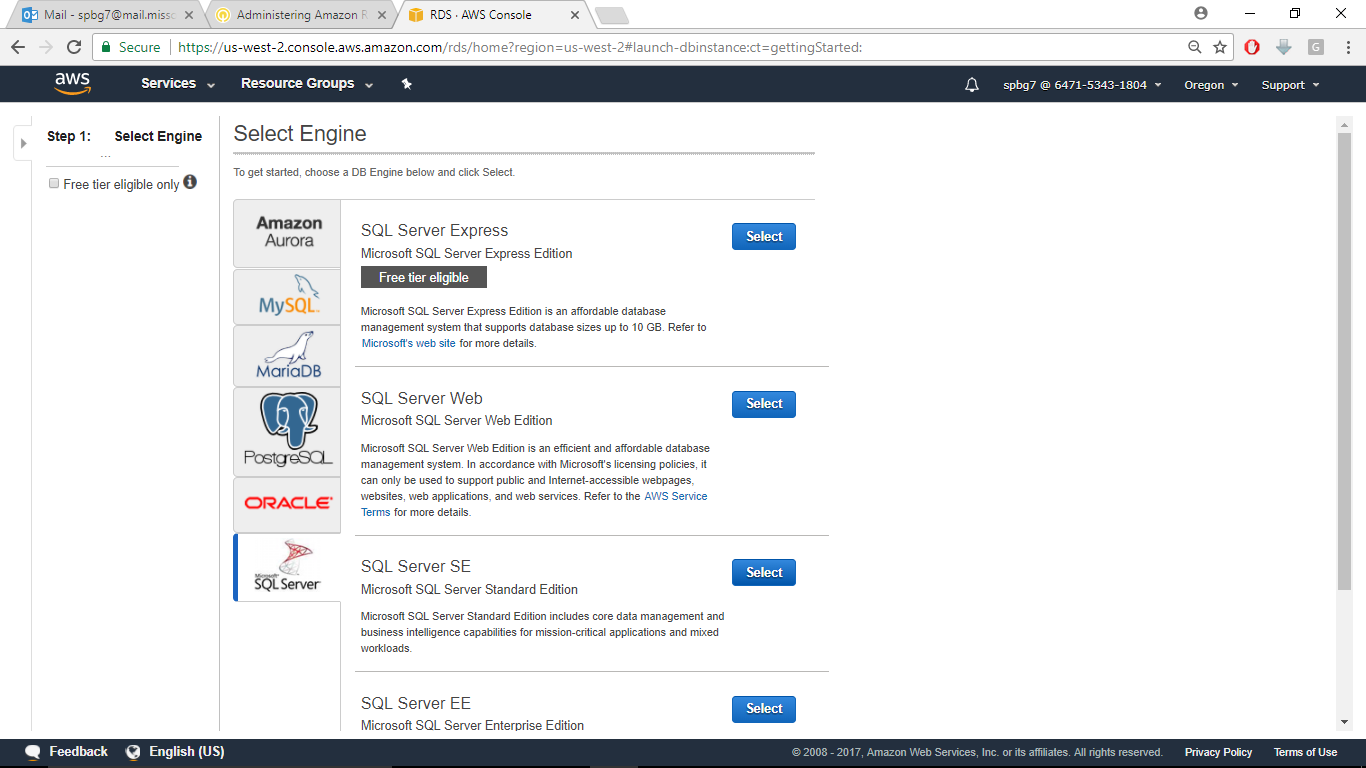
**Working with PostgreSQL, MySQL, and MariaDB Read Replicas**

Amazon RDS uses the MySQL, MariaDB, and PostgreSQL (version 9.3.5 and later) DB engines' built-in replication functionality to create a special type of DB instance called a Read Replica from a source DB instance. Updates made to the source DB instance are asynchronously copied to the Read Replica. You can reduce the load on your source DB instance by routing read queries from your applications to the Read Replica.

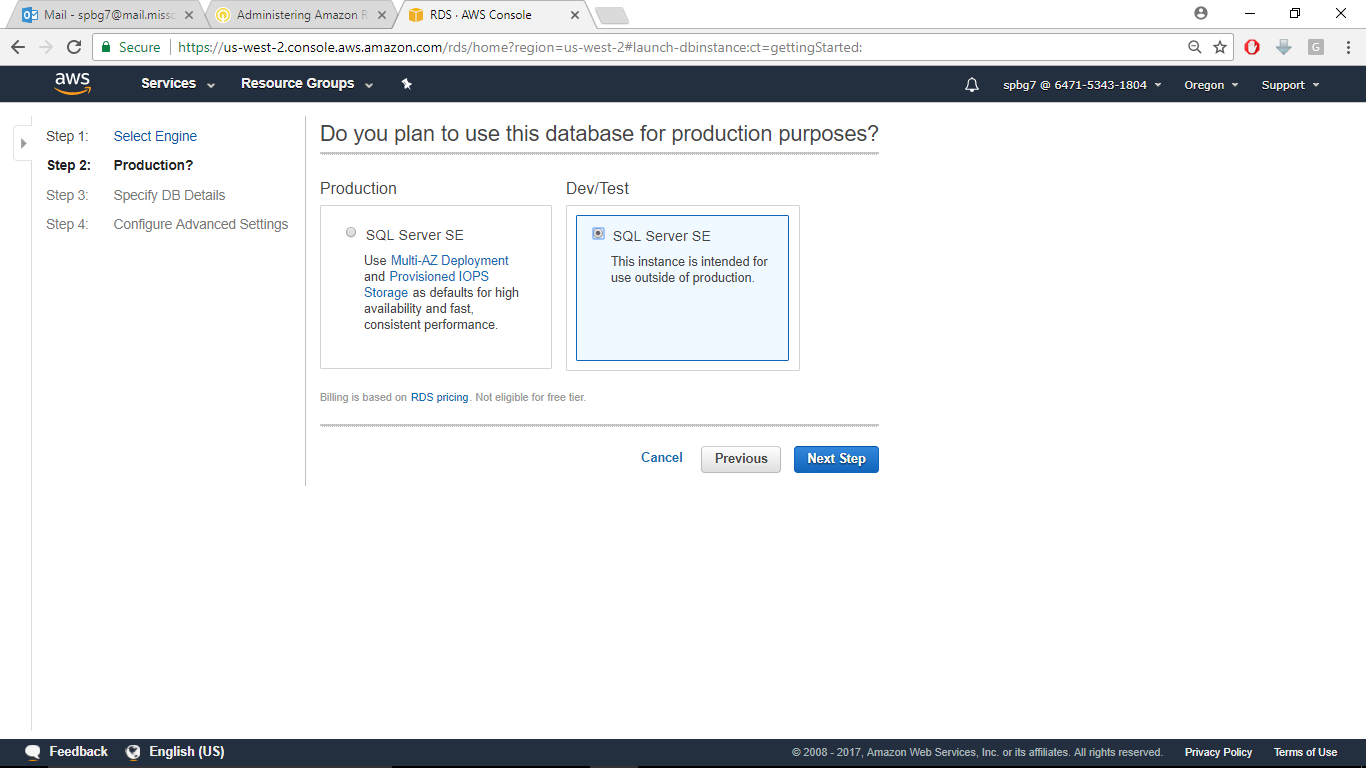
**Security groups:**

**Three types of**

**Lab 2: Migrate a stand-alone database to cloud**

**Step 1: Go to RDS service in AWS console and Select SQL server SE**

**Step 2: Select Dev/Test and click Next step**



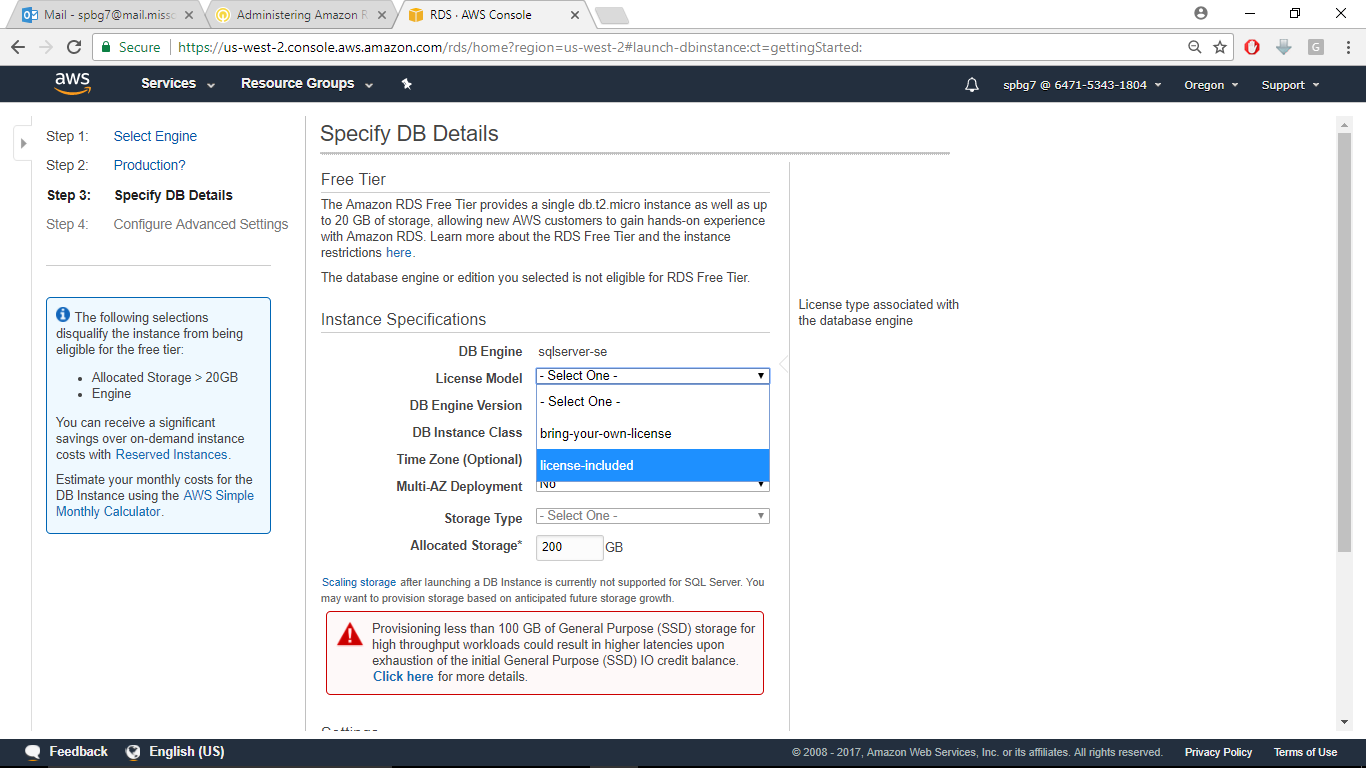
**Step 3: Select**

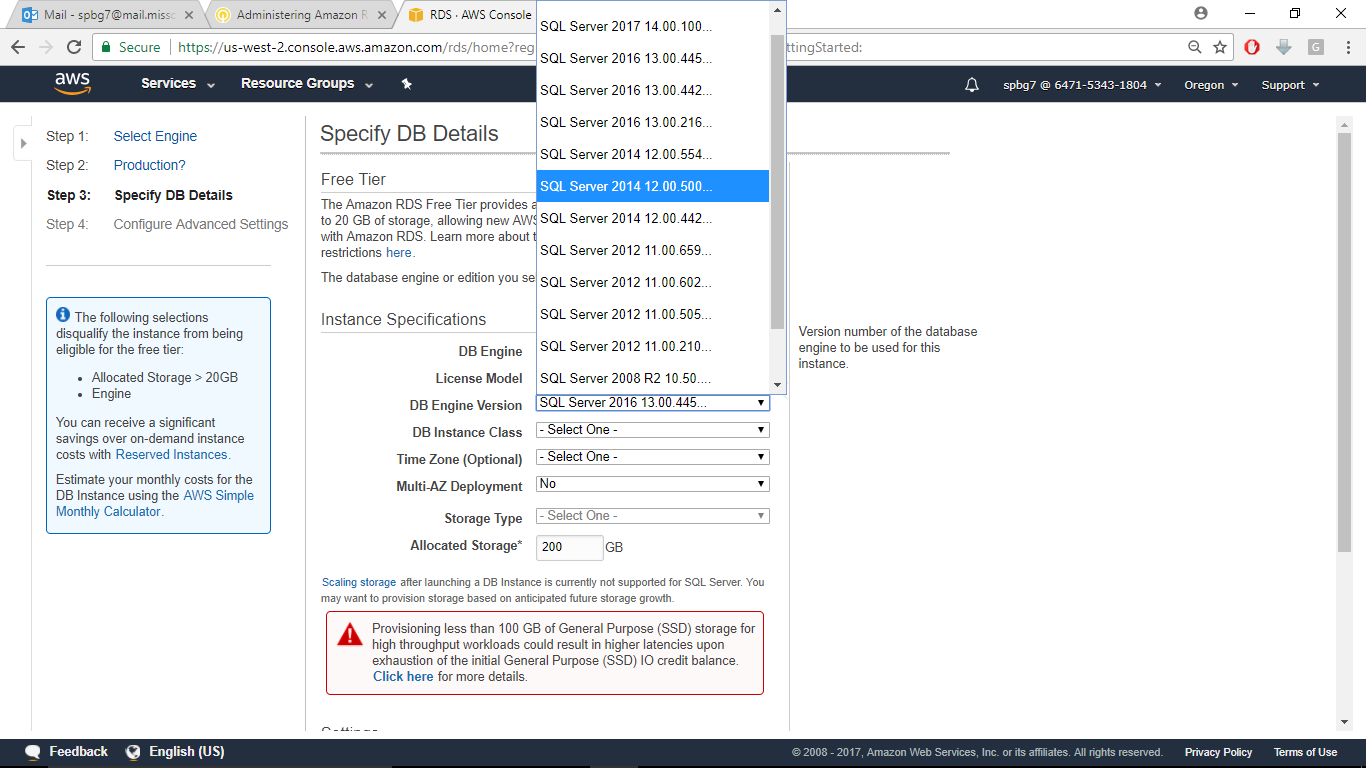
License model :‘License included’

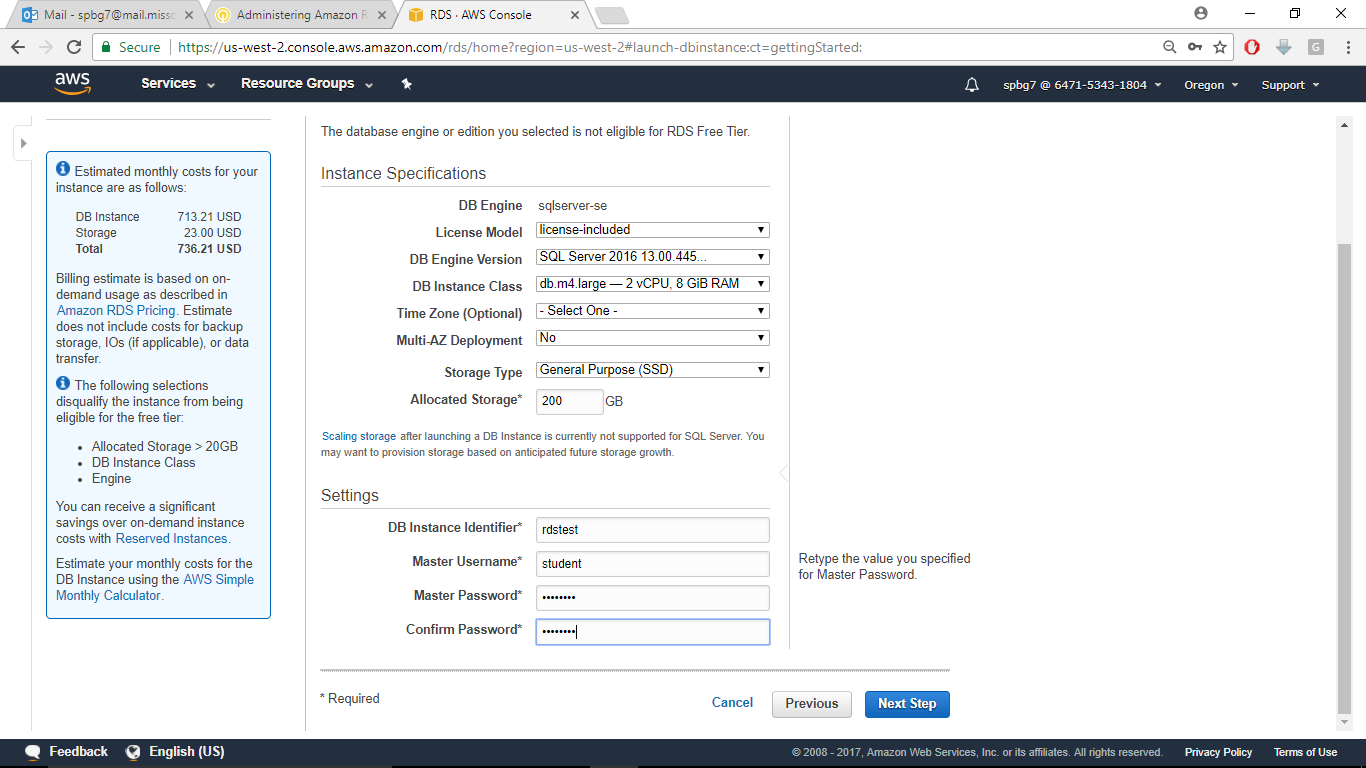
DB engine version: SQL server 2014

DB instance class: m3.large

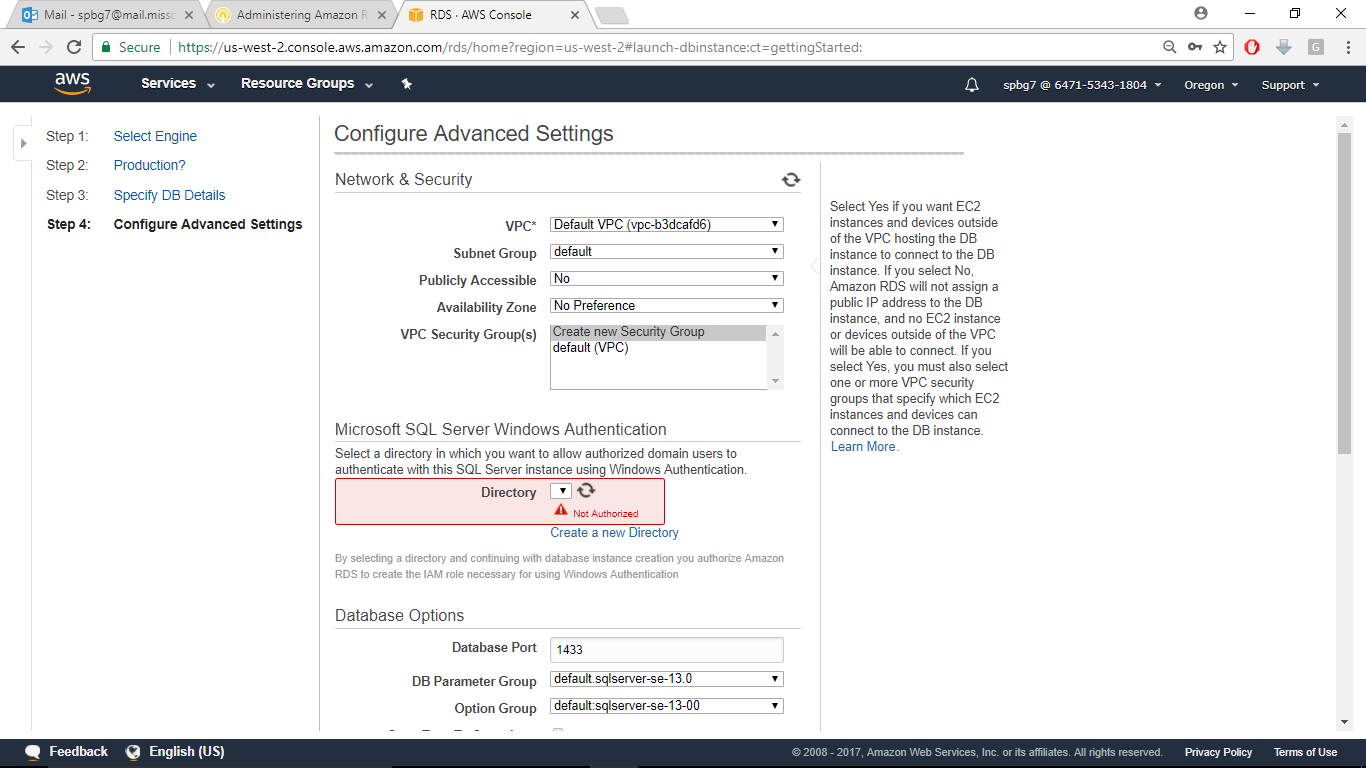
And for settings provide you own username and password

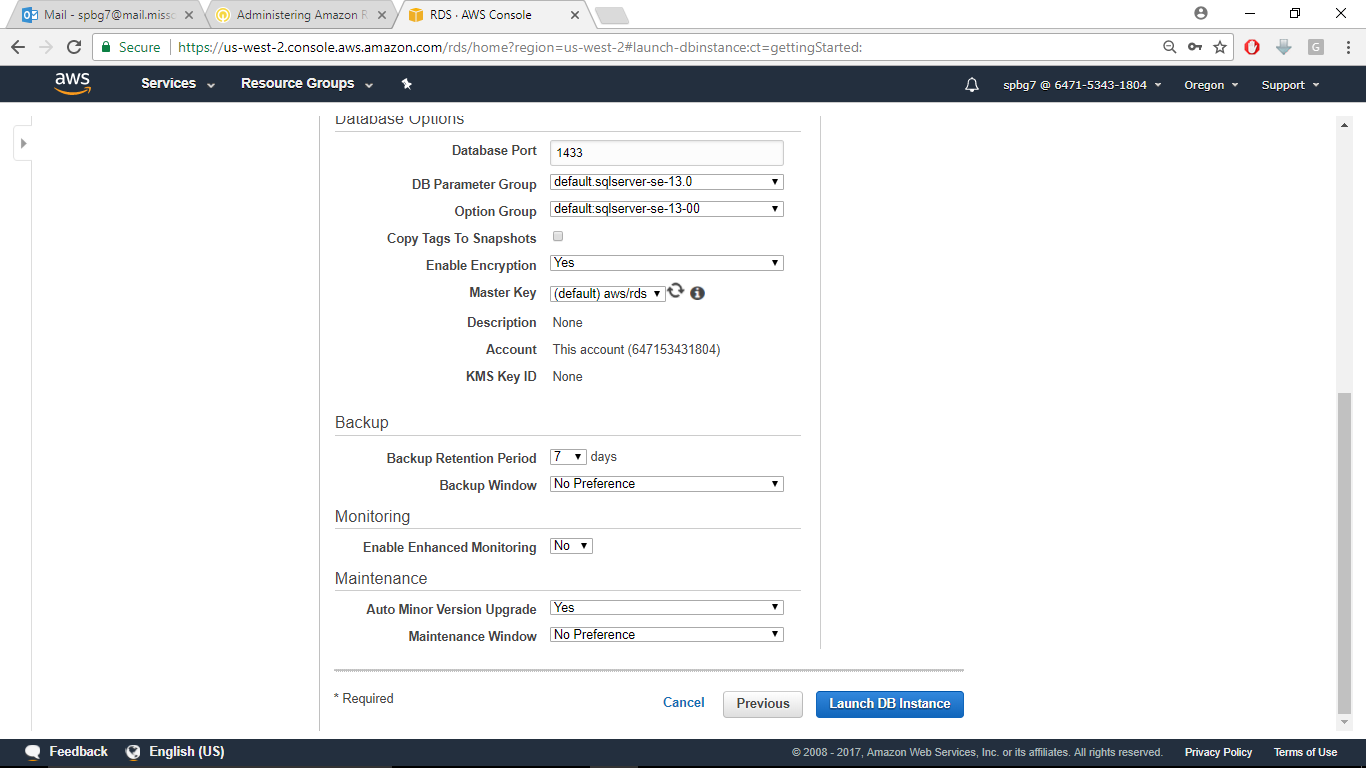




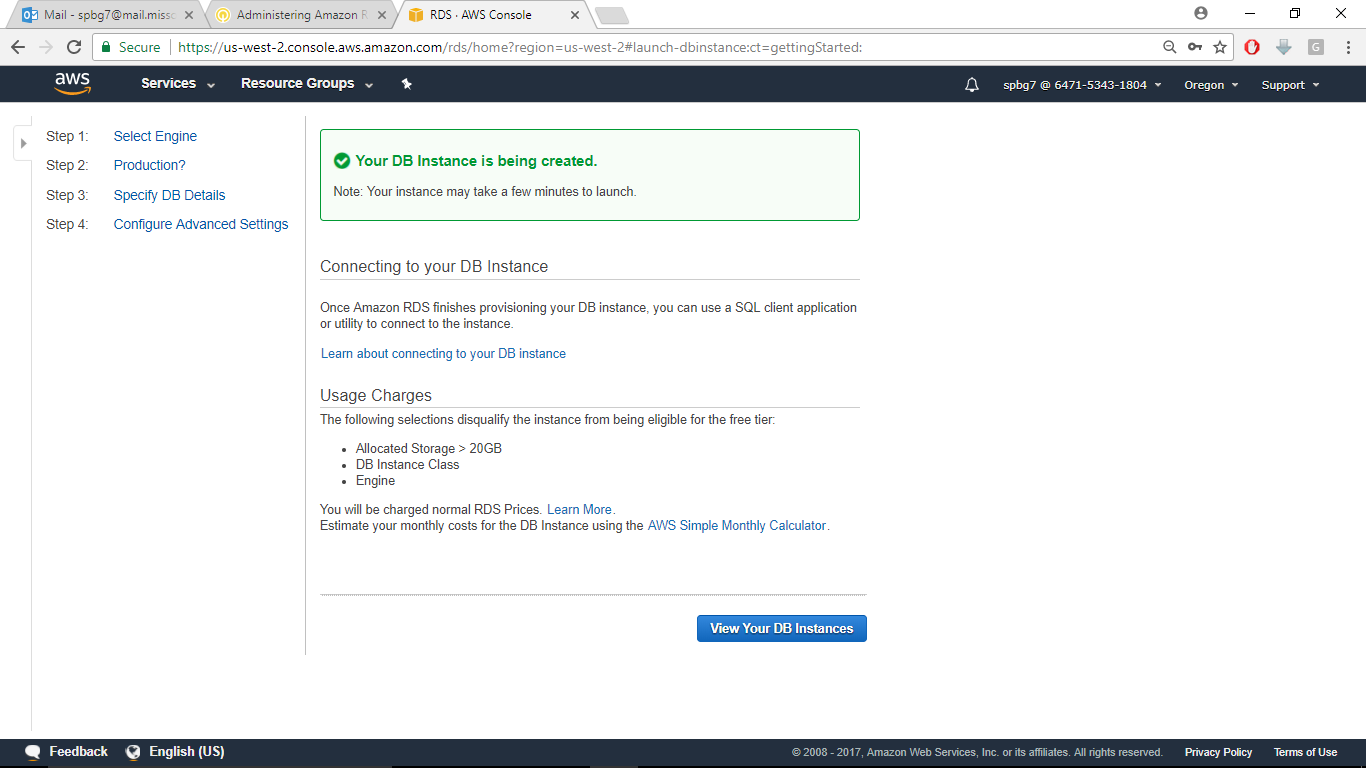


**Step 4: In configure advance details Select default VPC and for security group select ‘Create your own security group’ and keep the remaining options default and click ‘Launch DB instance’**

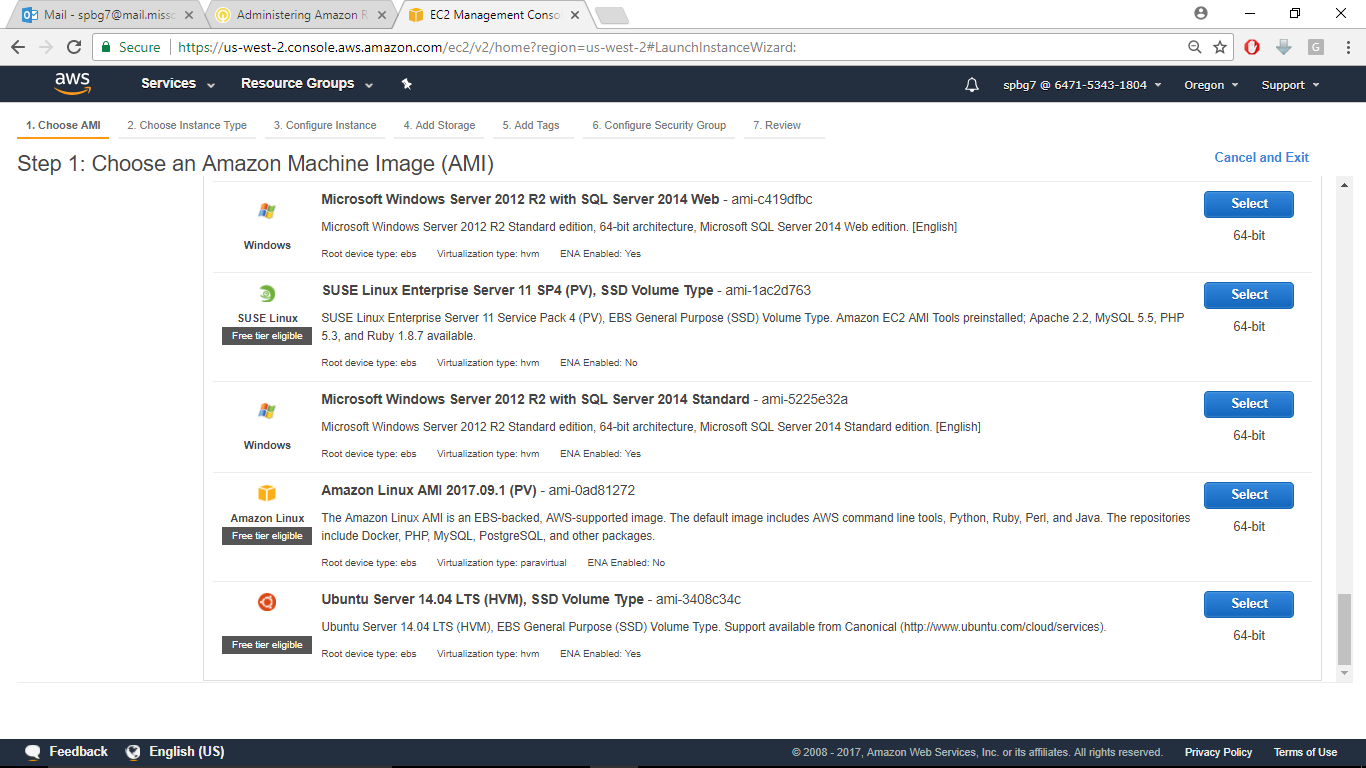




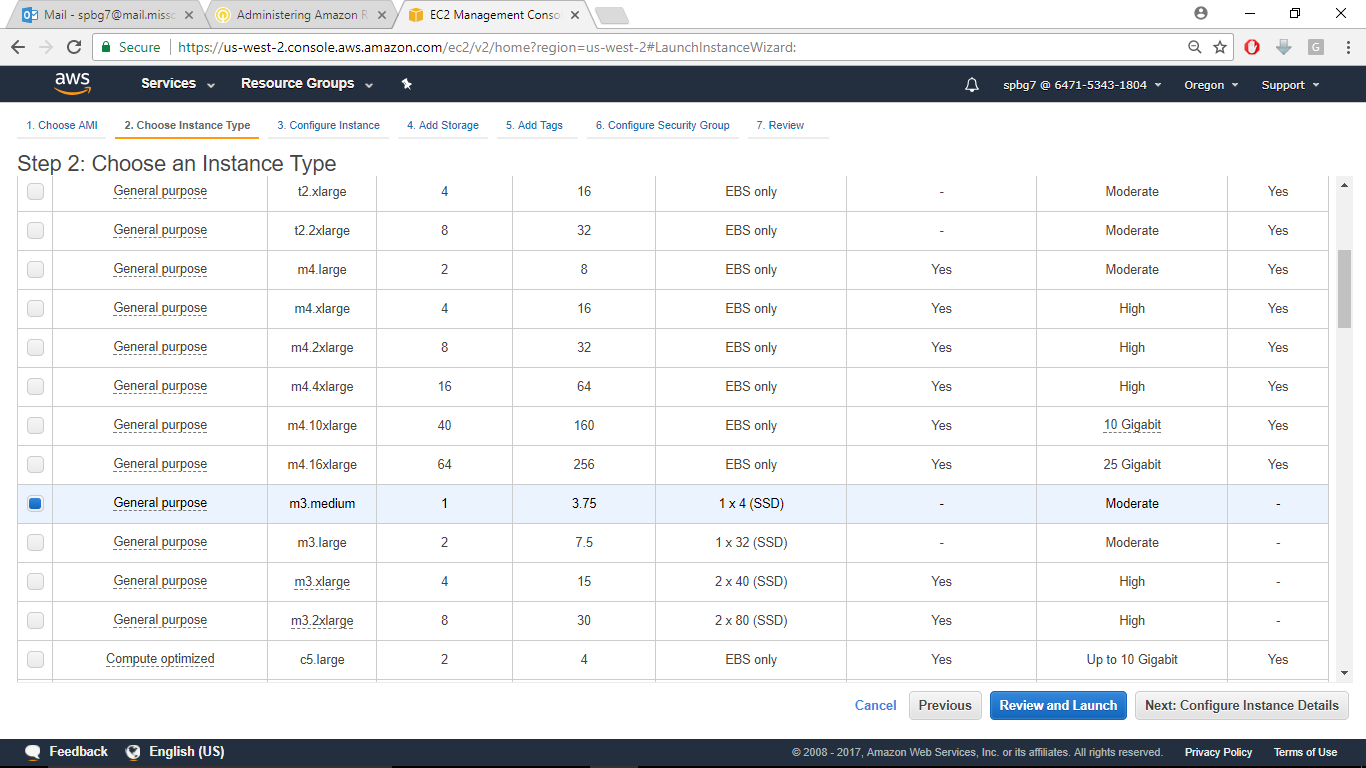
**You can see your DB instance getting created. It will take a while before it is up**

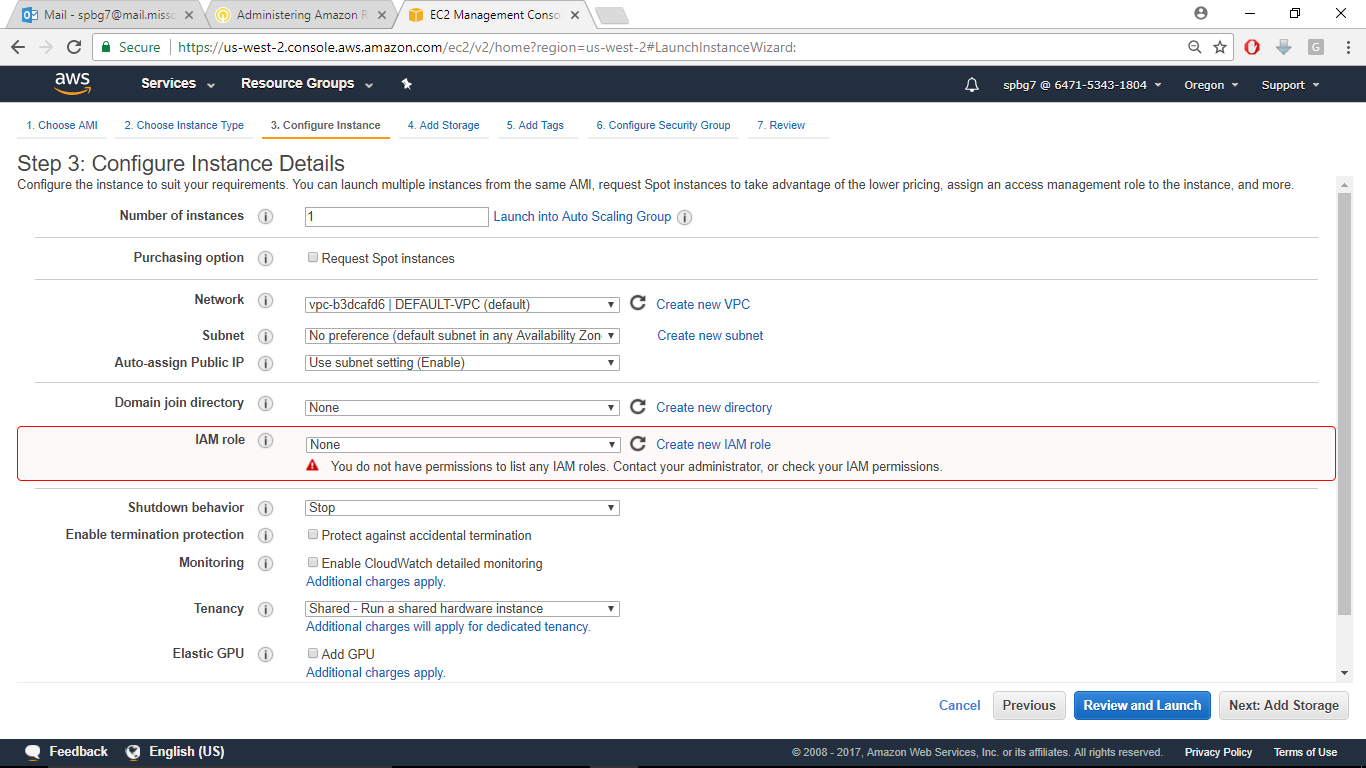


**Step 5: Launch windows server: go to EC2 and click Launch instance and select Microsoft windows server 2012 R2 with SQL server 2014**



**Step 6: Chose the instance type as m3.medium and click Next**

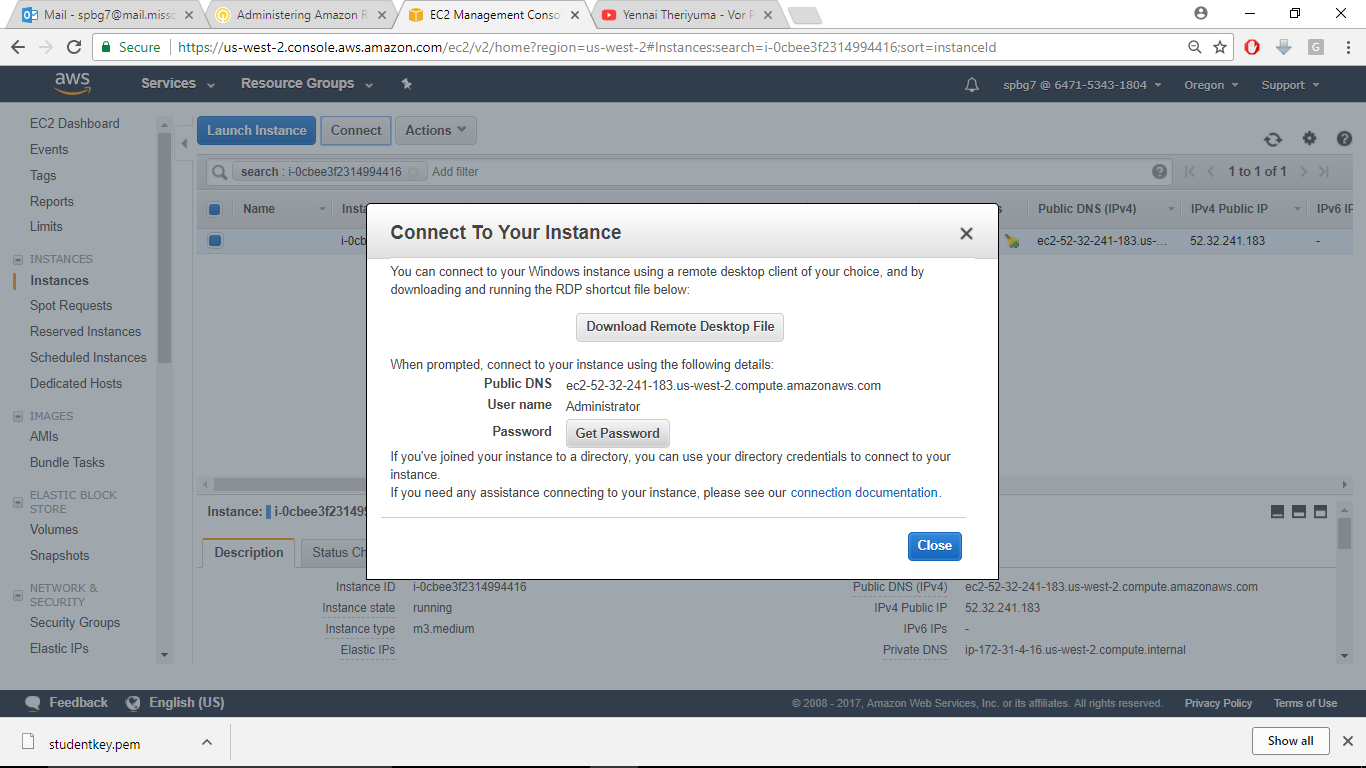


**Now select Default VPC and keep the remaining settings as is and click ‘Review and Launch’** 

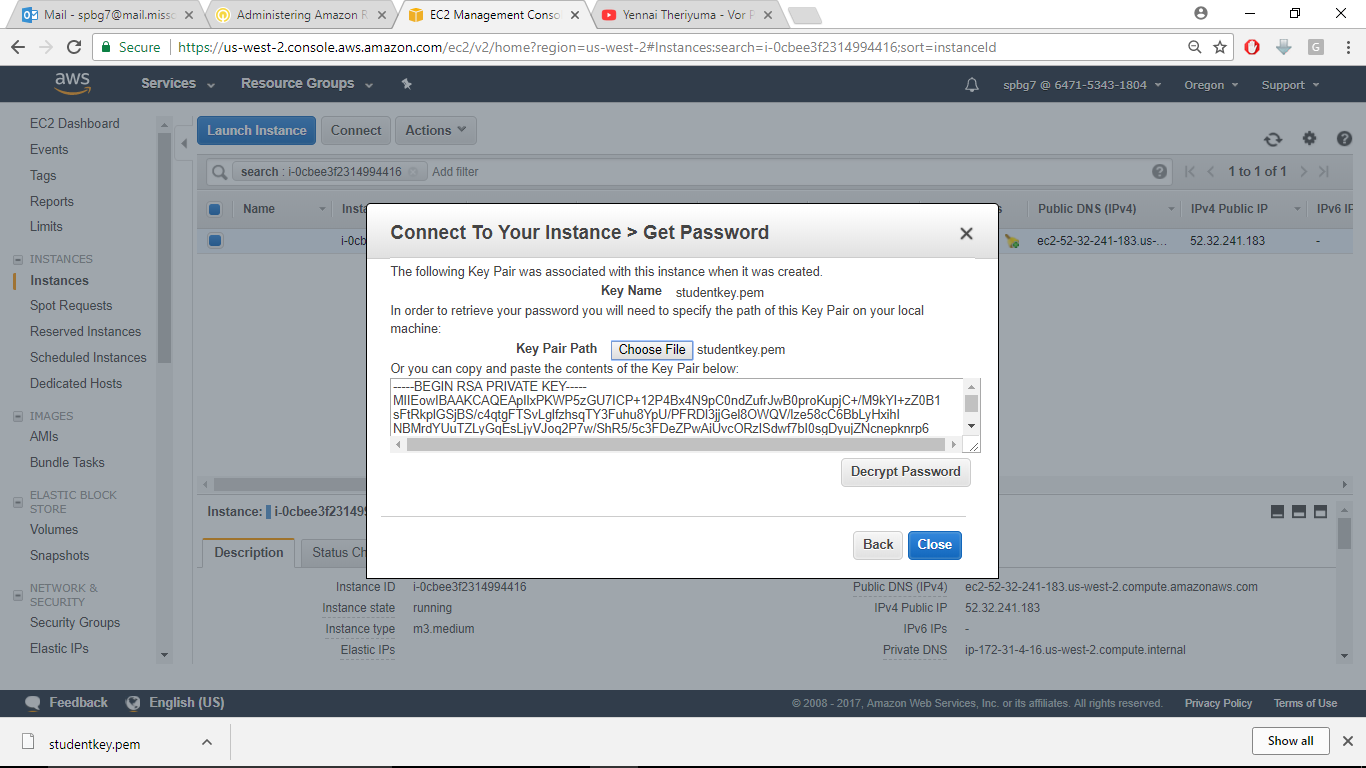
Now it will take you to key pair. For this lab, you create new key pair,then click launch

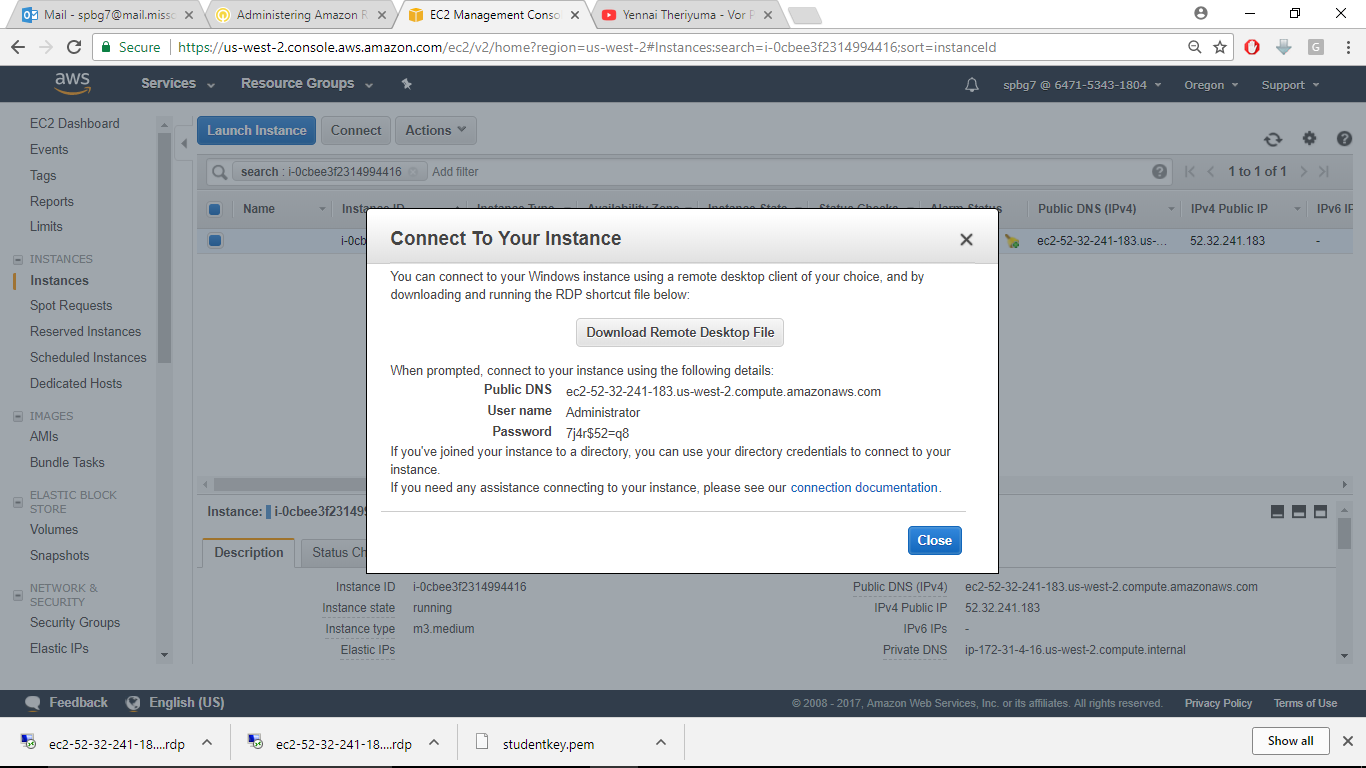
Now we are going to Access RDS instance via Microsoft SQL management studio

**Step 7: Now click connect instance and click get password.**



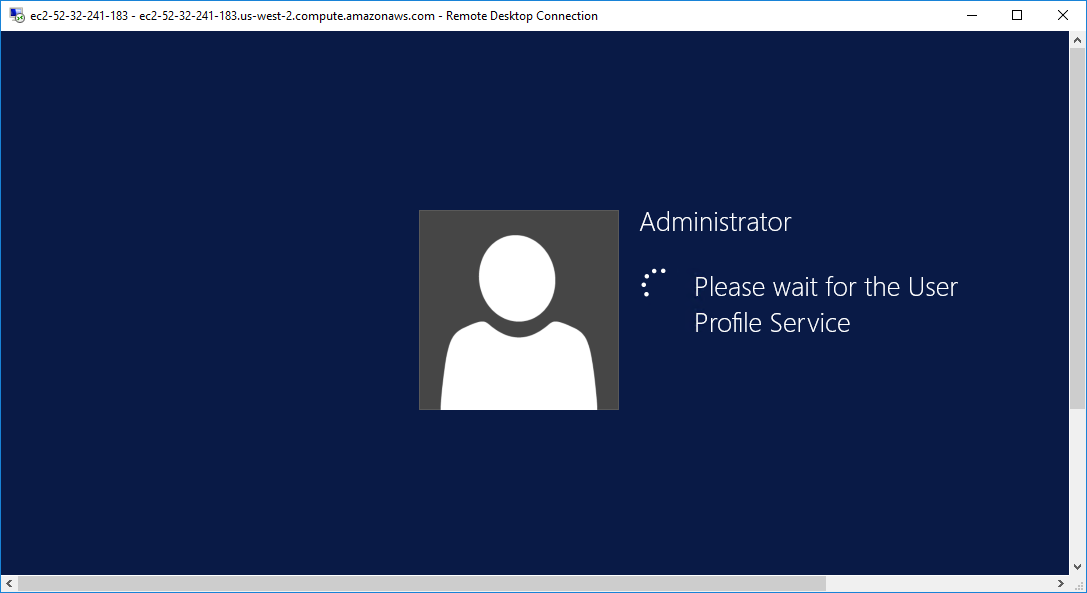
**Now browse to your key file and provide the key path where you have downloaded your key and copy the RSA key and then click decrypt password**



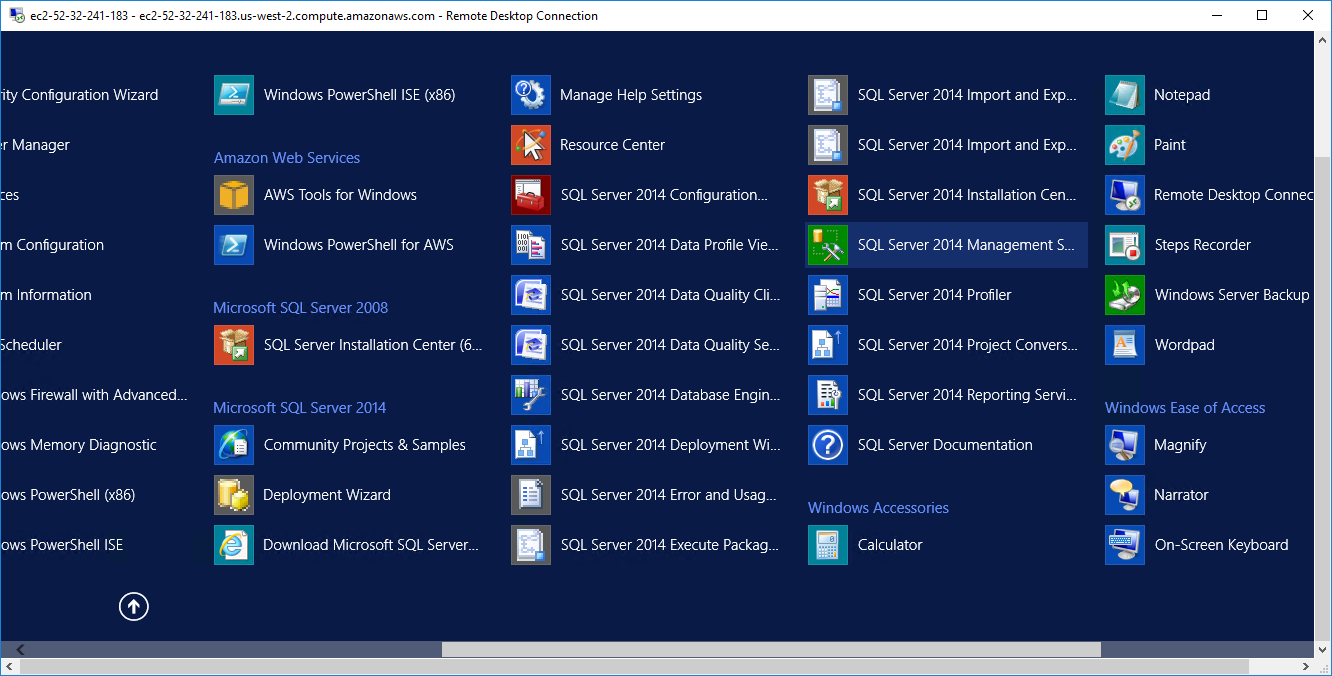


As you notice there’s a Windows Remote Desktop File that is downloaded .You click on it and it asks for the password, just copy the password shown in the above screenshot and click yes for the certificate pop-up.

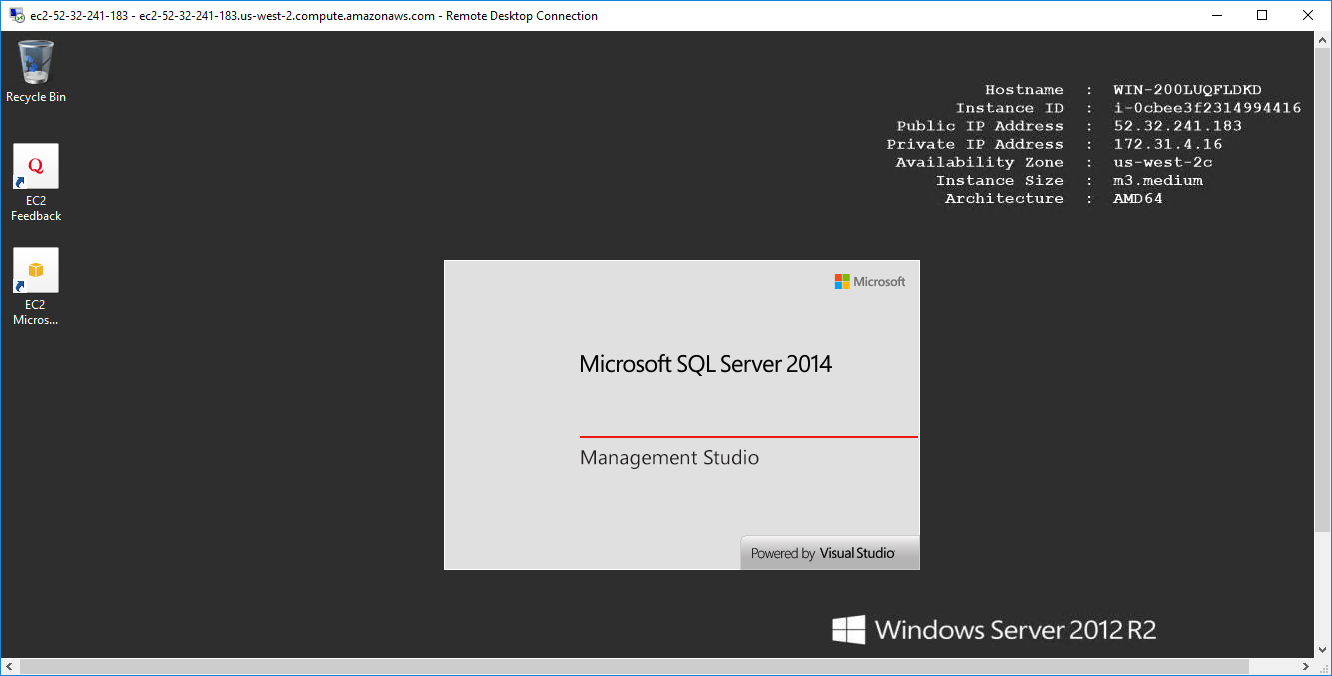
Now you will be logged into your instance in a moment



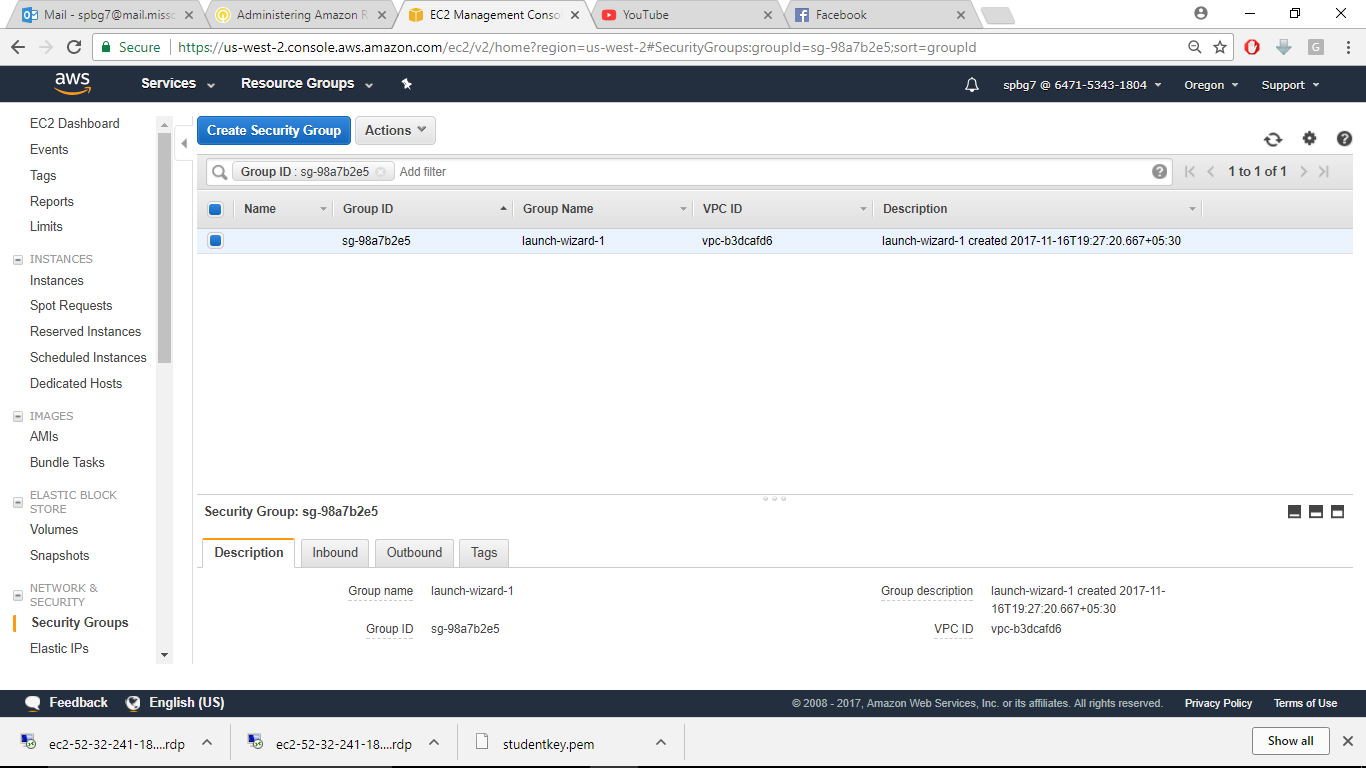
Now you will see the windows desktop. You just click on windows button and click the down arrow mark and find Windows server management studio.

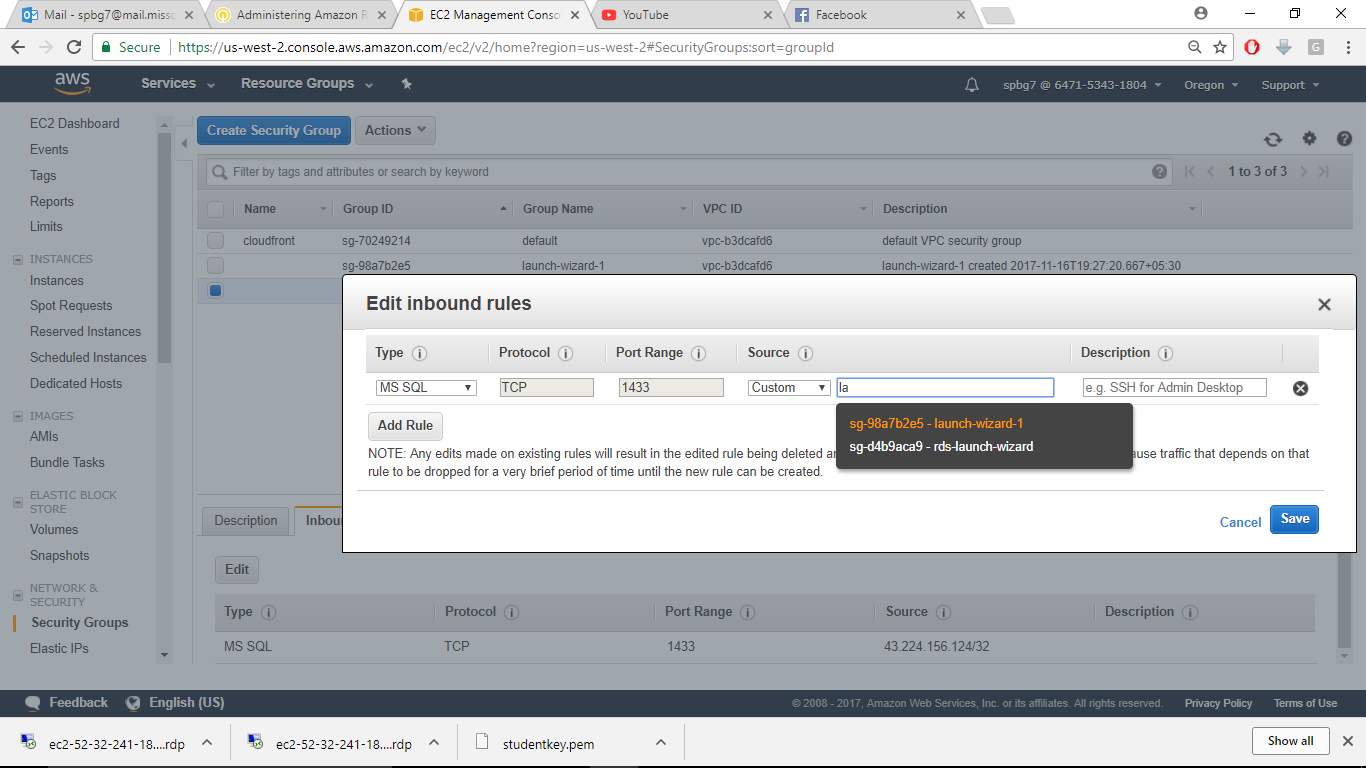


**Once you click on it you’ll see this and it will take 3-5 minutes to start the studio**



**Step 8:Now go to EC2 Dashboard and close the ‘connect to instance’ window and go to security groups and click on RDS security groups which is identified by rds-launch-wizard**

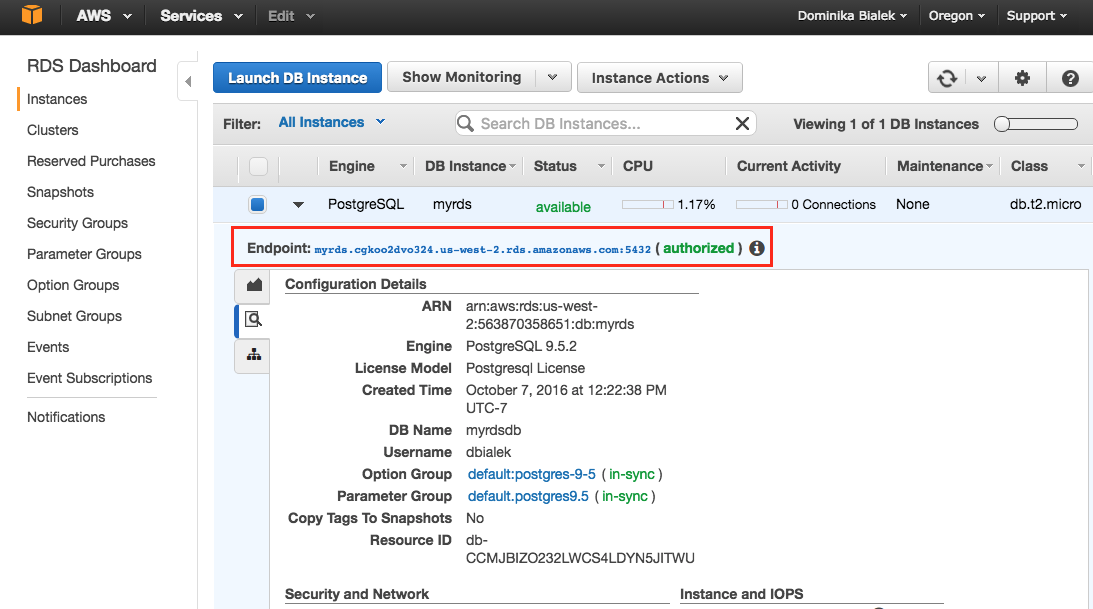


**Now edit the inbound and delete the IP address in custom and replace it with the security group name of EC2 instance and click save.**

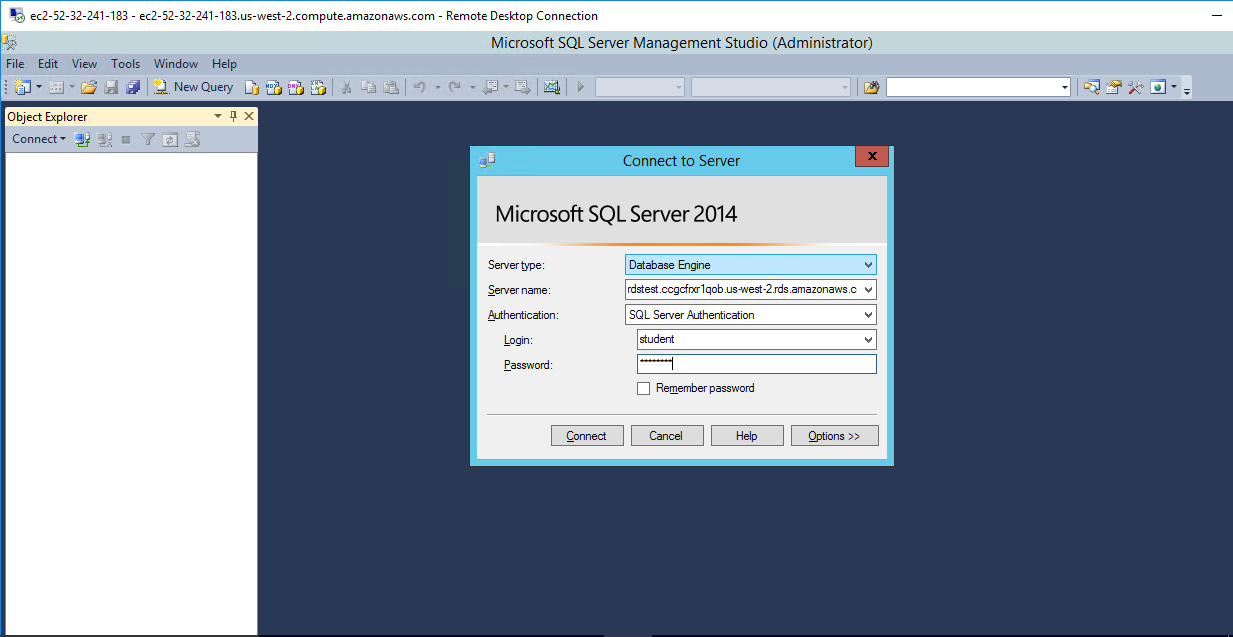
**This ensures that any EC2 instance will be able to connect to RDS instance.**

**Step 9: Connect to server**

Now go to RDS and copy the endpoint(without port number) and clip to your clipboard and then go to your windows RDP client.



Now in your RDP client you’re seeing windows SQL 2014 connection window and in the server name copy the endpoint and in windows authentication:select SQL Server Authentication and in the login password enter the details you provide when you created RDS instance

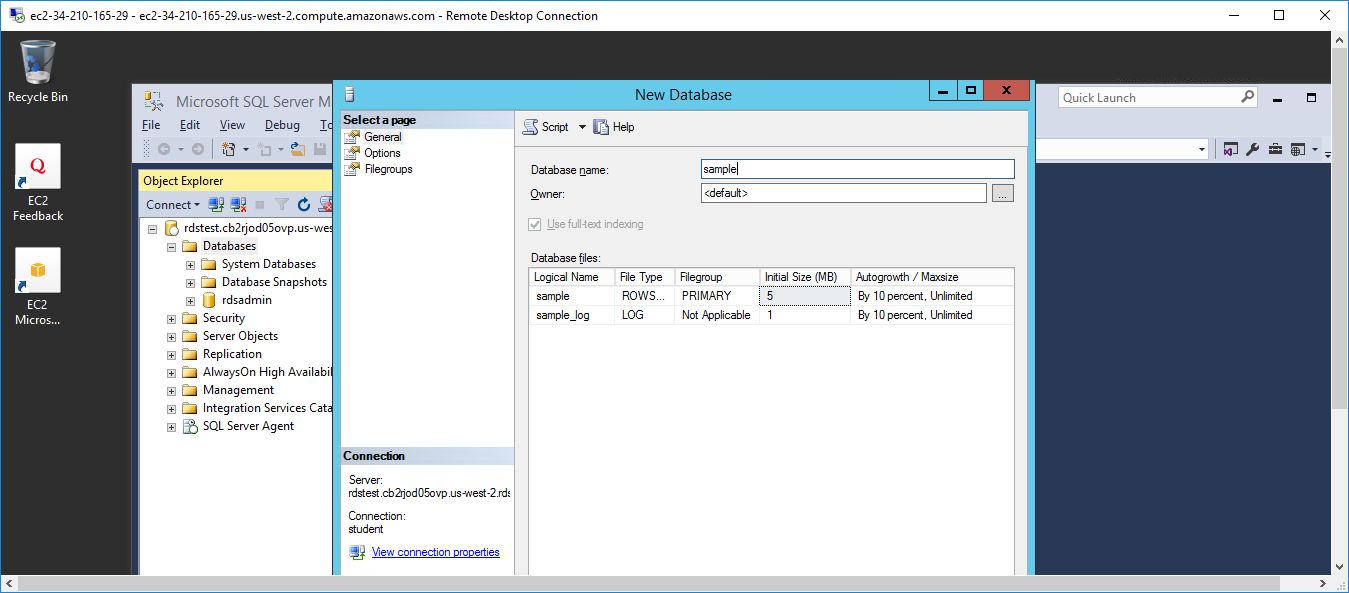


Now you will see Microsoft SQL server management studio opened.

1. **Create a database:**

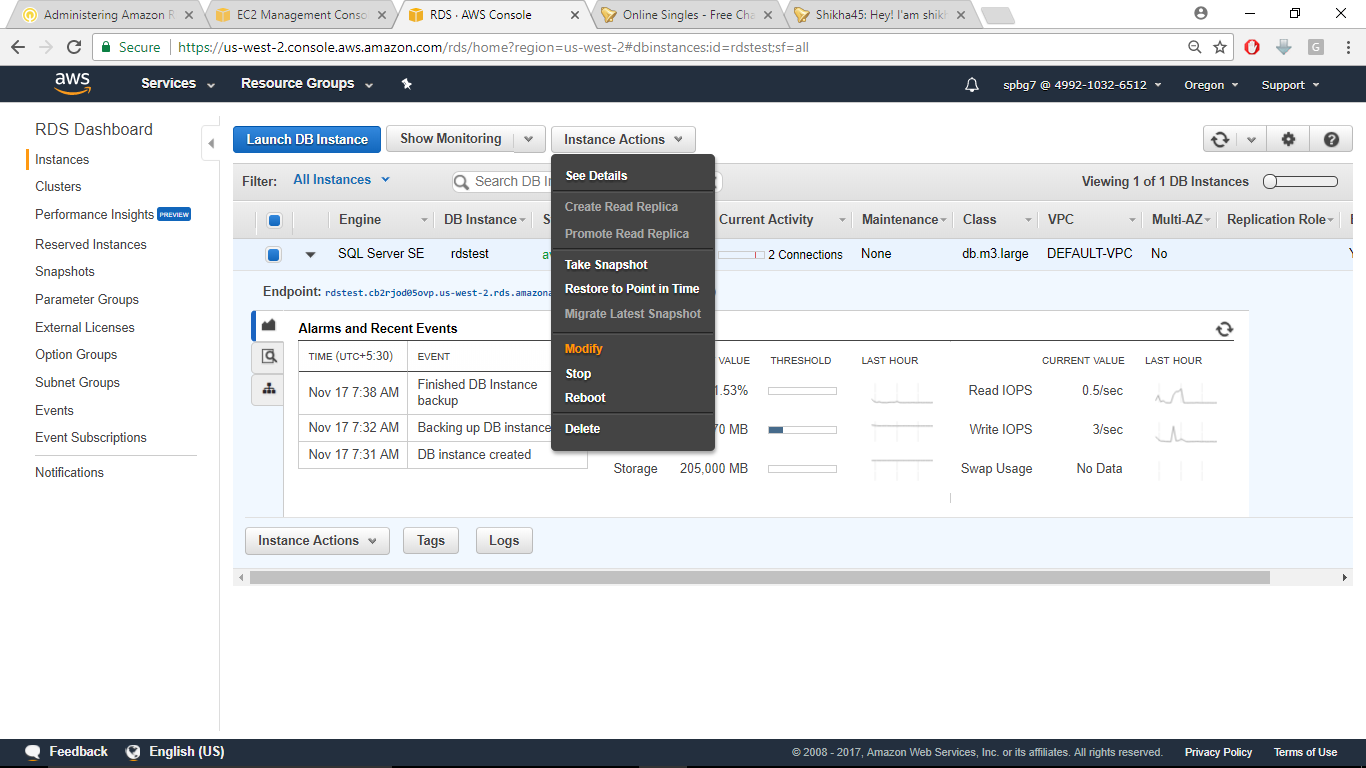
In the object explorer pane click on create database and name it as sample.

So far, what we have done is we used SQL server management studio and created a database and connected to our RDS instance



1. **Database Scaling:** Databases that are running on RDS can be scaled up or down by changing the instance DB type. This adds or remove compute power from DB instance. The process of changing instance type is similar to ec2 instance but here in RDS we don’t need to shut down

Now Return to RDS dashboard click instance instance actions and then modify



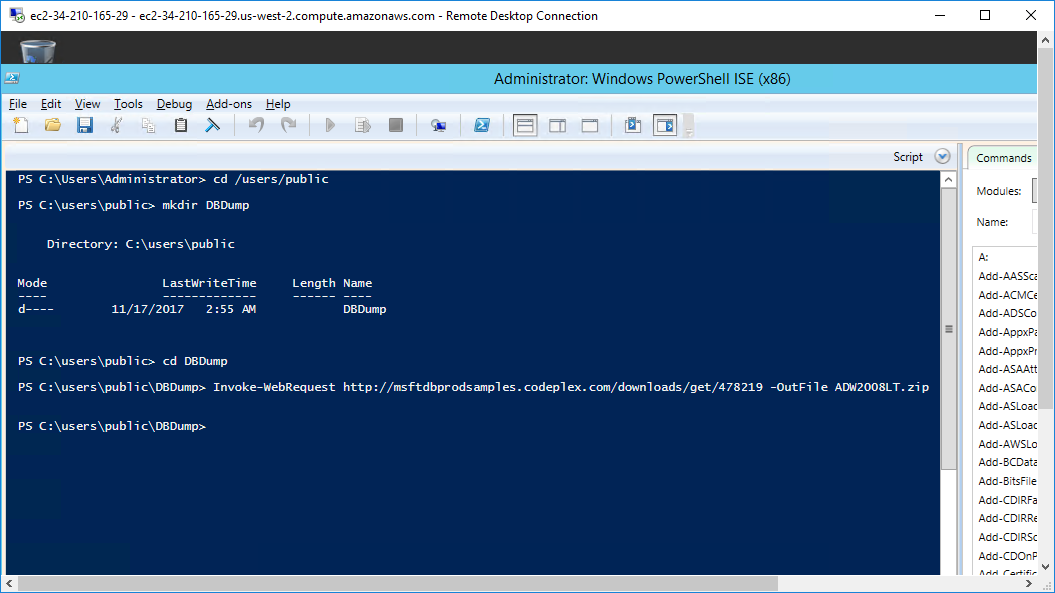
Now change the instance class change it to r3.large and **check apply immediately** and then Click continue and then click modify instance ,This will take several minutes but you proceed with further steps

**Step 10: Migrating a database to RDS:**

Now we shall migrate a simple Database installed on EC2 to RDS .You will use some Windows Powershell commands here

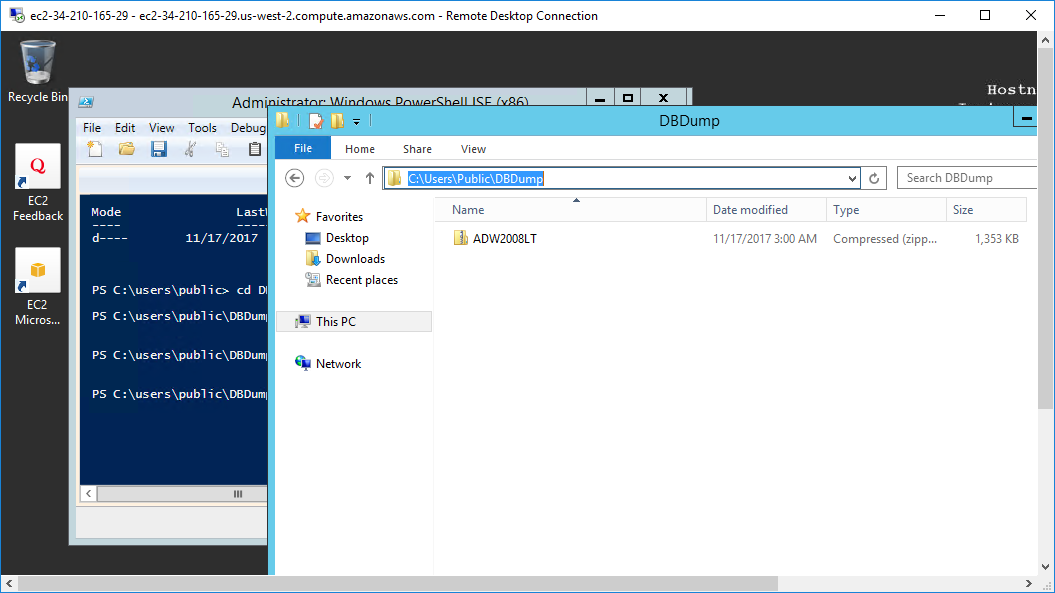
Go to windows RDS and click windows button and click down arrow and find Windows powershell and launch it.

Now press ctrl R to get script pane,and pass on following commands as shown:



This set of commands downloads a sample database ADW2008LT .Now Press F5 to execute the script.After the script gets executed, go to the path where the file is downloaded and chose Extract ALL and Extract the folder

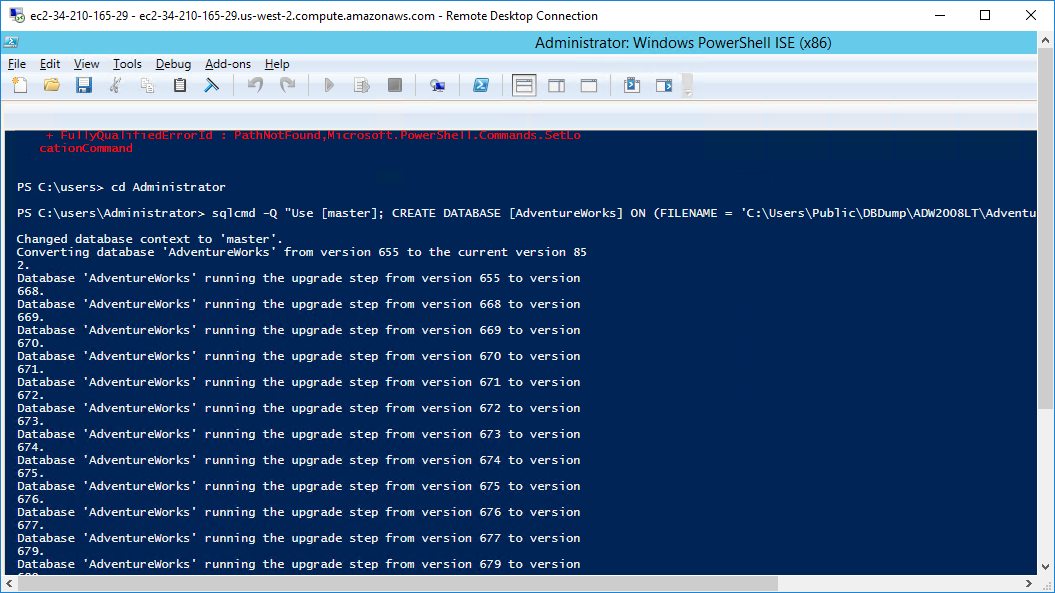
The path is :C:\Users\Public\DBDump



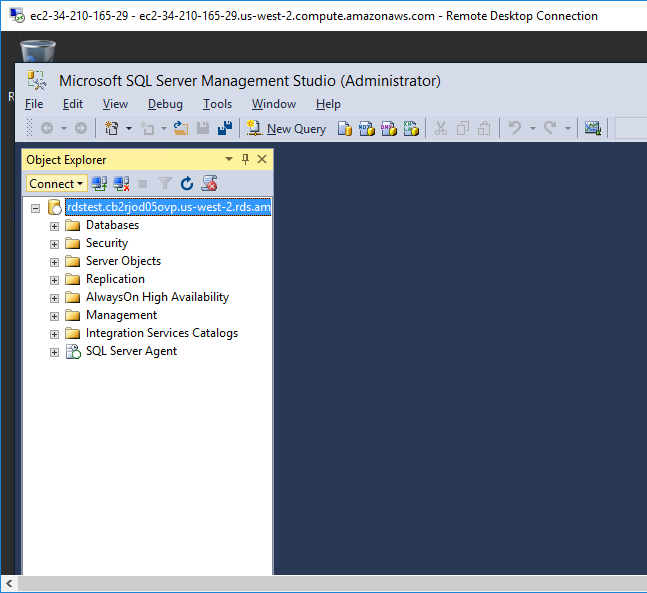
**Now get back to you windows Powershell and use the following command which imports database from EC2 instance**

Change the directory to Administrator. Then you use the command

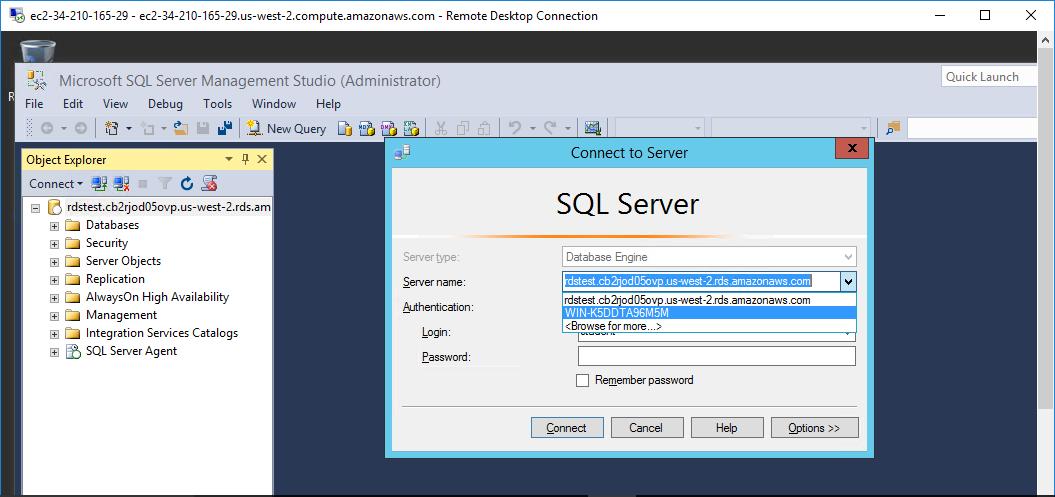
sqlcmd -Q "Use [master]; CREATE DATABASE [AdventureWorks] ON (FILENAME = 'C:\Users\Public\DBDump\ADW2008LT\AdventureWorksLT2008\_Data.mdf'),(FILENAME = 'C:\Users\Public\DBDump\ADW2008LT\AdventureWorksLT2008\_Log.ldf') for ATTACH"



**As you can see the command is getting executed, now minimize PowerShell and maximize SQL Server studio, and in the object explorer click connect**



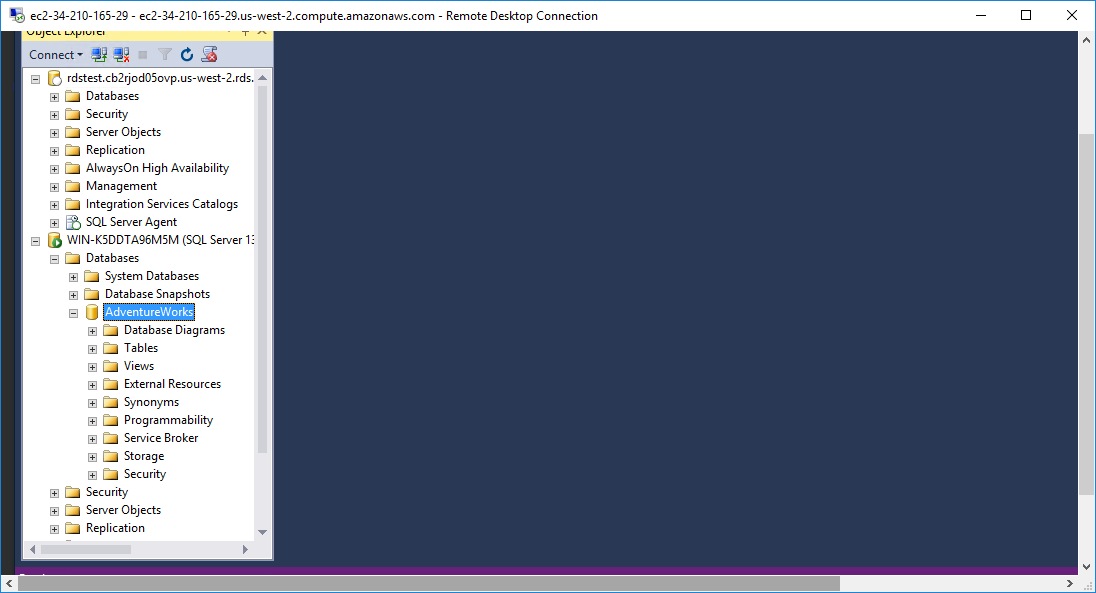
And in the server name select the RDS endpoint and in the Authentication select the server name that starts with WIN-XXX



Now your both Amazon RDS and local SQL server is connected.

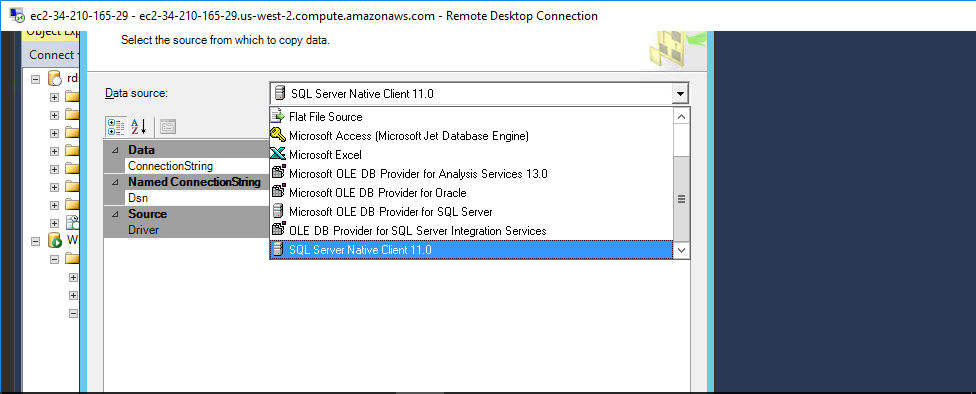
**Migrate database to RDS instance:**

1. **Now click on the object explorer you will find AdventureWorks sample databse that you created using PowerShell commands.**



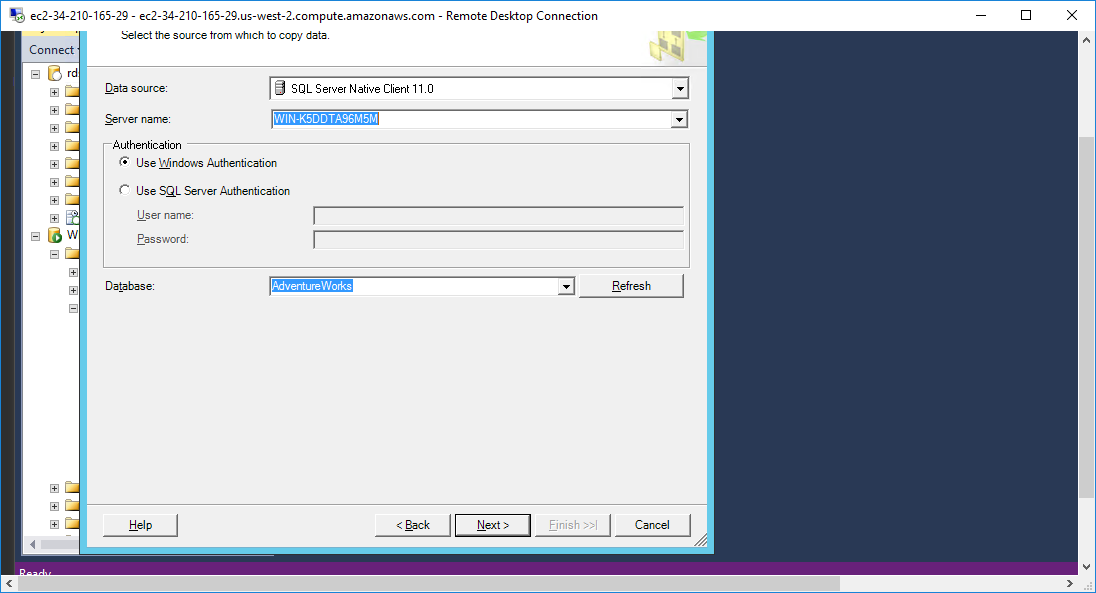
1. **Now right click the database and click on tasks and then ExportData**

You will see this screen:select SQL Server Native Client 11.0

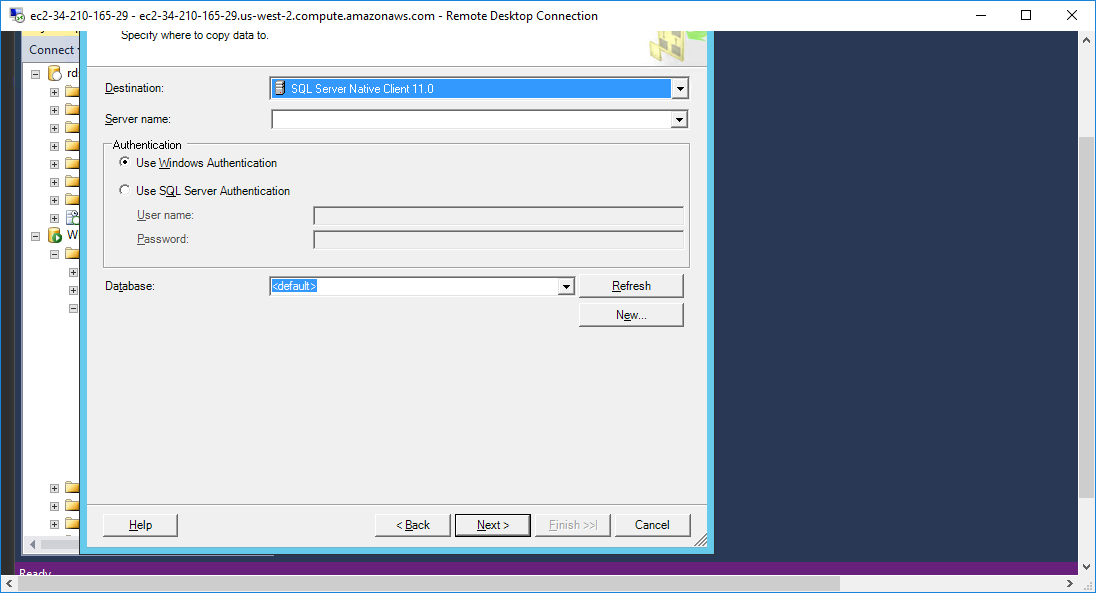


And in the servername drop-down list a name is automatically populated which starts with WIN-

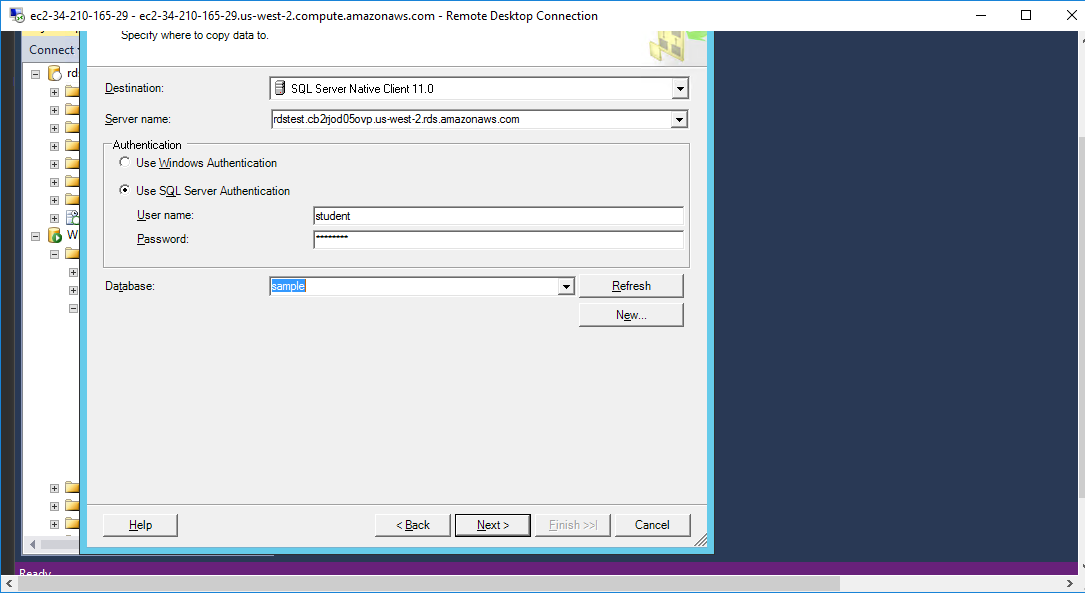
1. **Now click on ‘use Windows Authentication’ and in the database drop-down verify Adventureworks is selected. Now click** Next



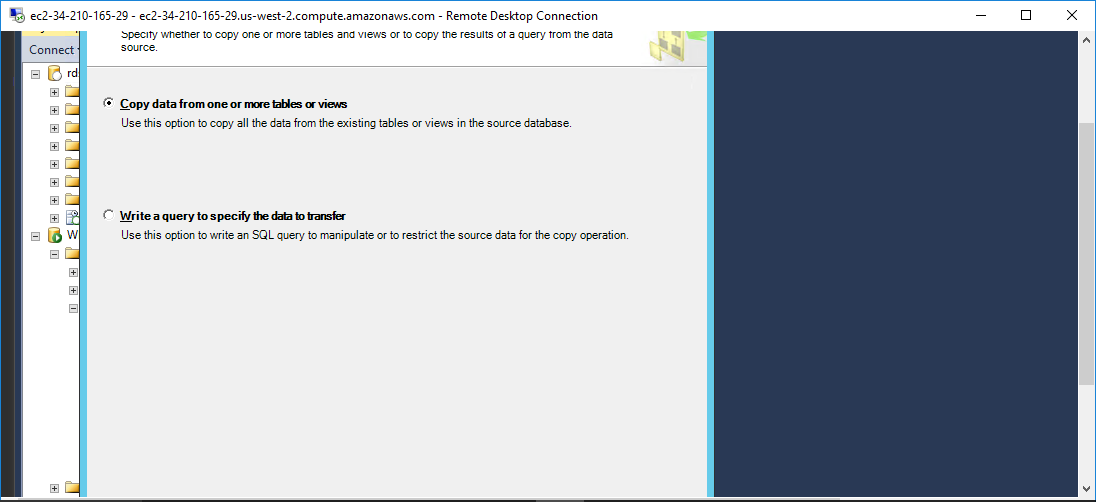
1. **Now on the chose destination page select the SQL Server Native Client 11.0**



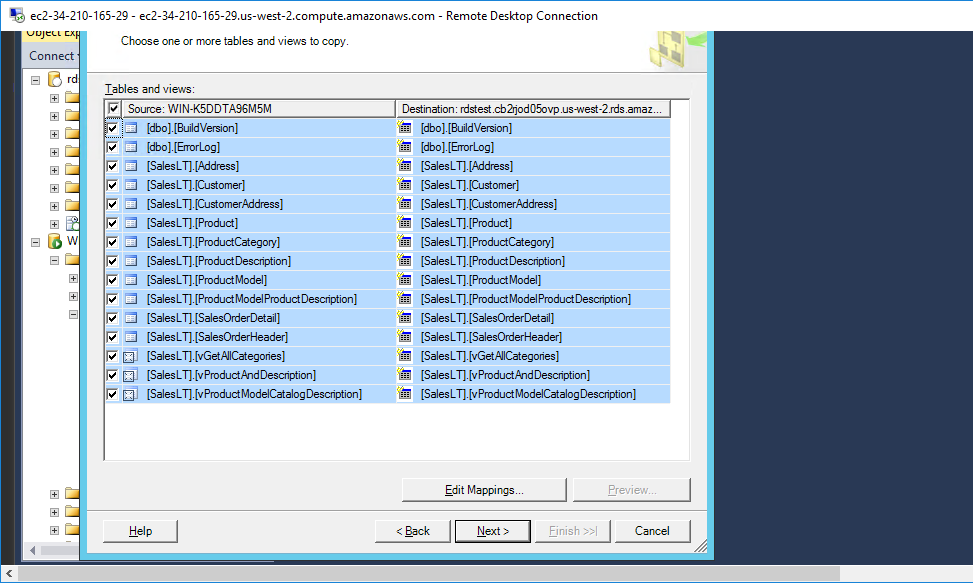
And then chose server name copy and paste the RDS endpoint without port number.Then,select Authentication:Use SQL Server Authentication,and enter the username and password details that you set when you created RDS instance.Then,chose database as Sample and click Next.



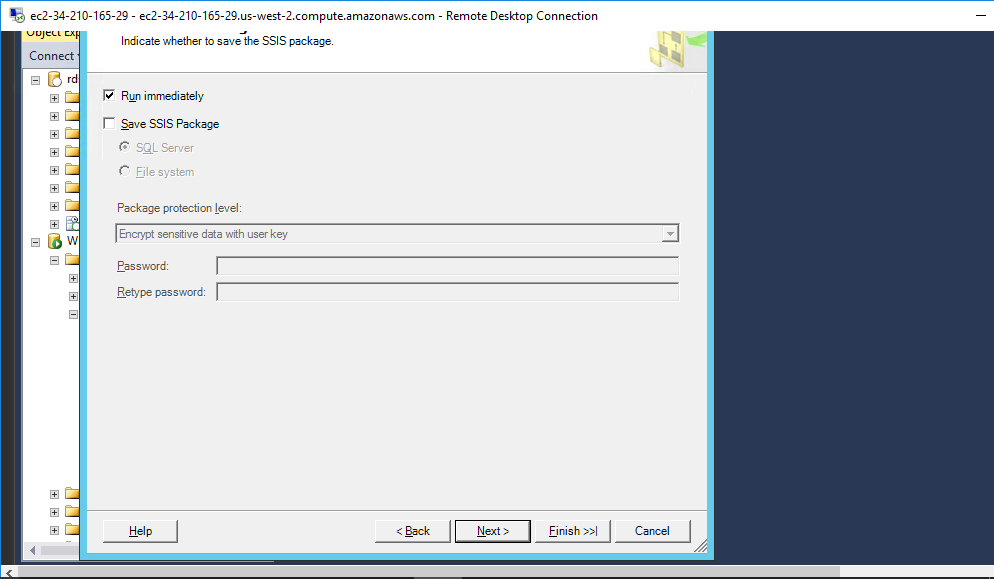
1. **Select copy data from one or more tables or views and click next**



1. **Now Check all sources and click next**



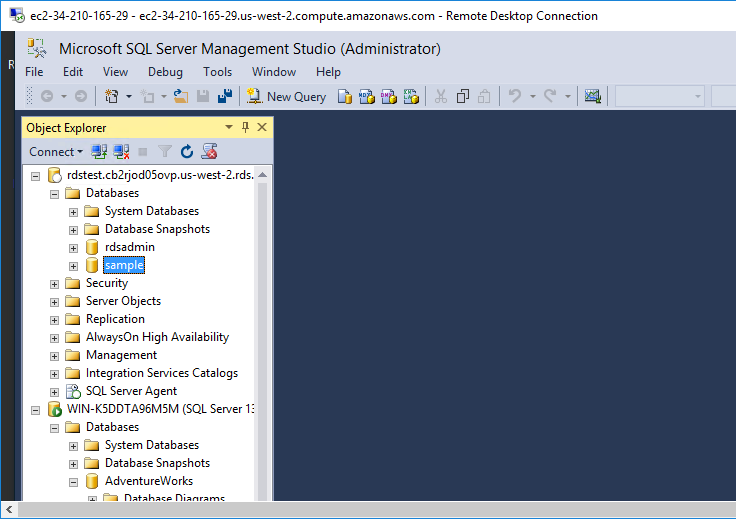
**7.Now verify run immediately is selected and click finish twice**

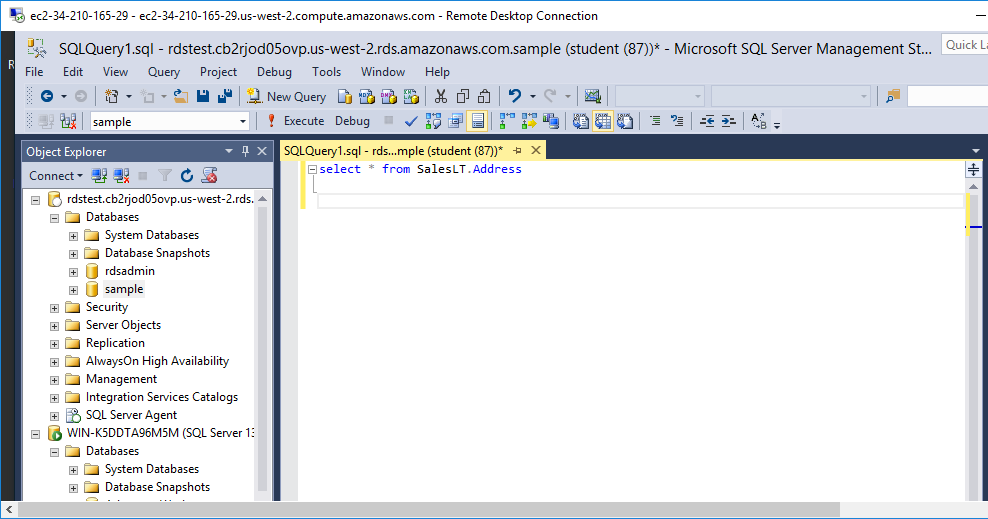
`

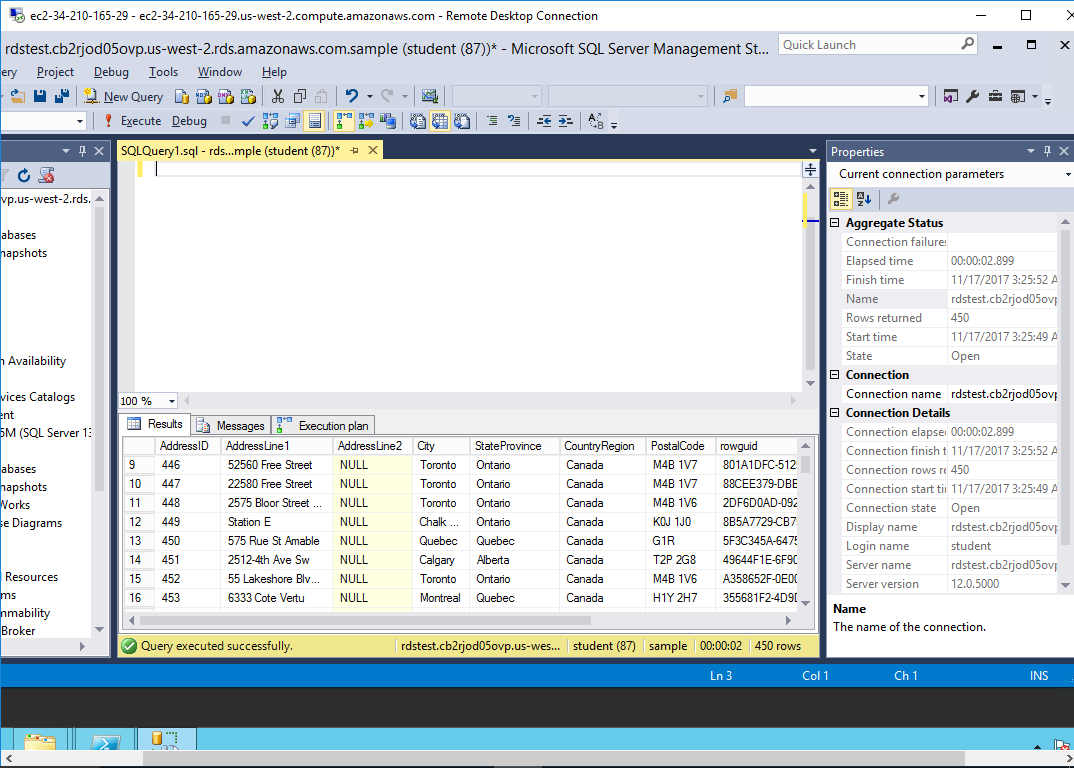
This would take couple of minutes as all the tables and data is being moved to RDS instance.A message will be displayed that operation of copying data to RDS instance is complete.

**8.Now verify the data transfer by running a simple SQL query**

Go to object explorer and expand databases and right click on sample and select new query. An editor will be opened. Now write a simple command as shown



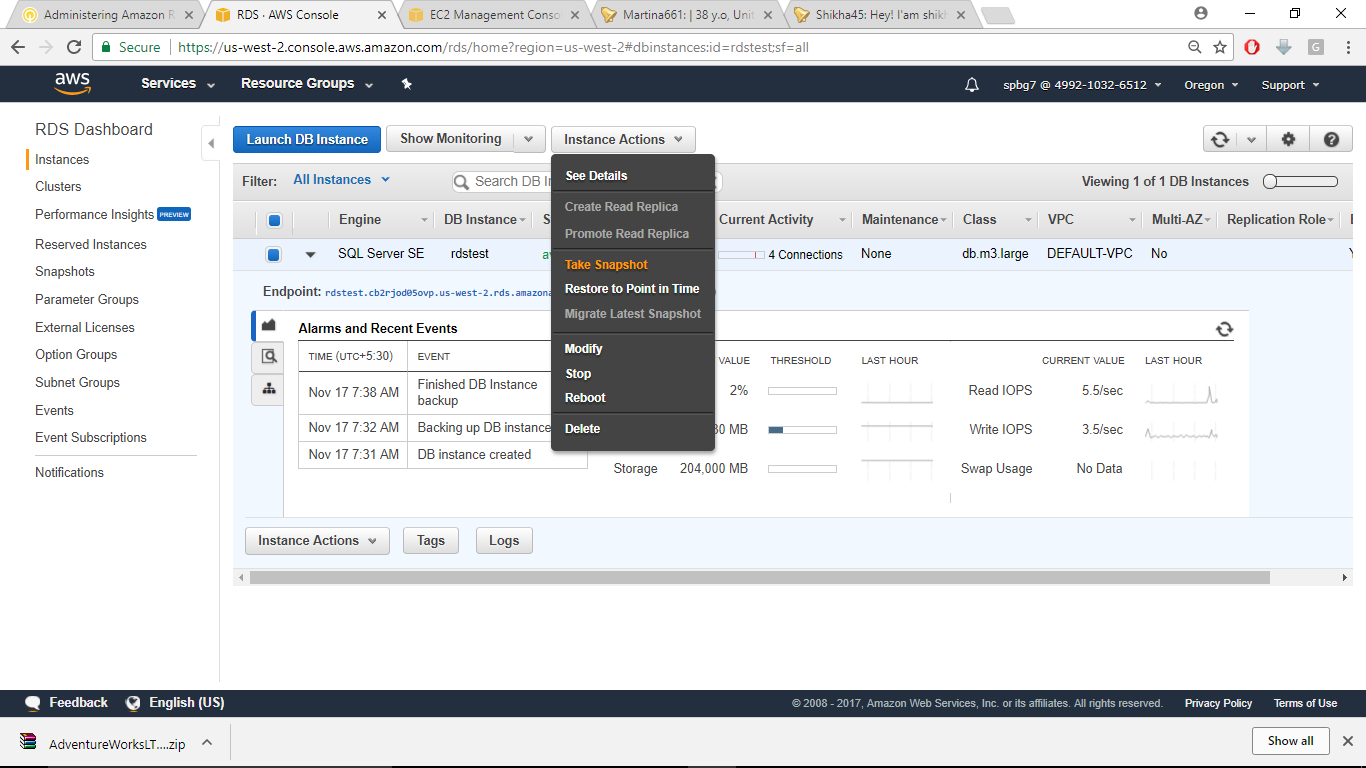




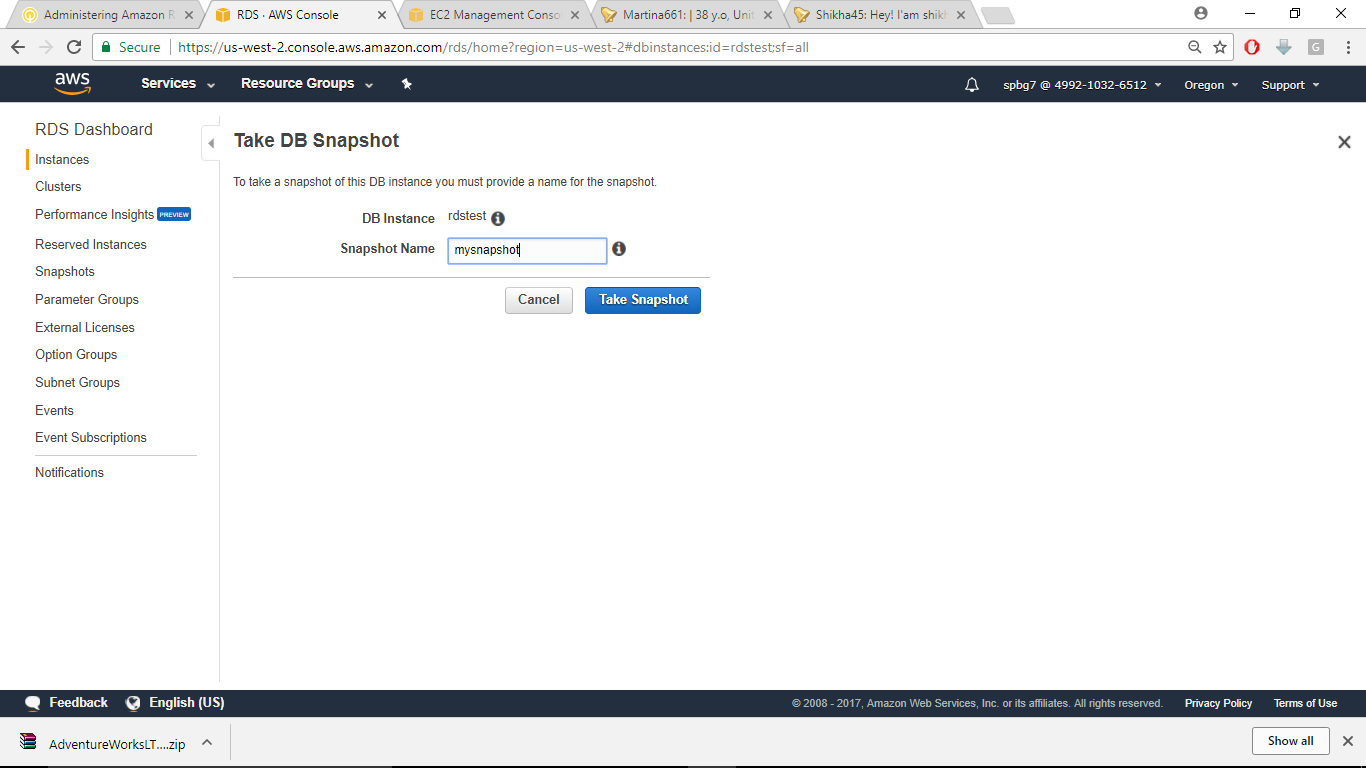
As you can see query is executed successfully.

**Step 11: Perform database backup**

**Now go back to RDS dashboard in AWS and select your DB engine. Then, click on Instance actions and click take snapshot**

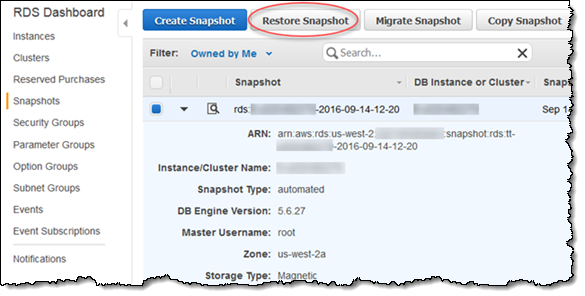


Name the snapshot and click take snapshot



**Step 12: Restore database from snapshot**

After snapshot is created as you can see in Progress column, select your snapshot and click Restore Snapshot.



**Conclusion:**

1.You created an RDS instance with user credentials

2.You launched an EC2 instance in windows

3.You connected Microsoft SQL server studio to rds and created a sample database

4.You downloaded a database using Powershell commands

5.You exported that database to RDS and executed SQL query

6.You took a snapshot and created a database from the snapshot.