**Create an RDS Subnet Group**

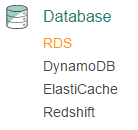
Amazon Relational Database Service (**Amazon RDS**) makes it easy to set up, operate, and scale a relational database in the cloud. Before launching actual RDS instances, you need to configure a **DB Subnet Group**.

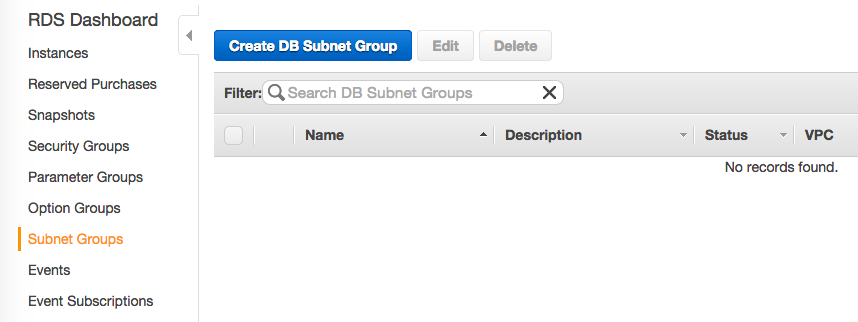
**Subnets** are segments of a VPC's IP address range that allow you to group your resources based on security and operational needs. A **DB Subnet Group** is a collection of subnets (typically private) that you create in a VPC and designate for your DB instances. Each DB subnet group should have subnets in at least two Availability Zones in a given region. Note that SQL Server Mirroring with a SQL Server DB instance requires at least 3 subnets in distinct Availability Zones.

When creating a DB instance in a VPC, you must select a DB subnet group. Amazon RDS uses that DB subnet group and your preferred Availability Zone to select a subnet and an IP address within that subnet to associate with your DB instance. When Amazon RDS creates a DB instance in a VPC, it assigns a network interface to your DB instance by using an IP address selected from your DB Subnet Group. If the primary DB instance of a Multi-AZ deployment fails, Amazon RDS can promote the corresponding standby and subsequently create a new standby using an IP address from an assigned subnet in one of the other Availability Zones.

You can create an RDS Subnet Group using the RDS launch wizard.

Select the RDS service from the Management Console dashboard:

  
  
From the RDS dashboard, click **Subnet Groups** from the left-hand menu.

   
  
Click **Create DB Subnet Group** to open the creation wizard.

You must fill the form using the following data:

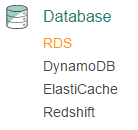
* **Name:** cloudrds
* **Description:** rds lab
* **VPC ID:** select the available one

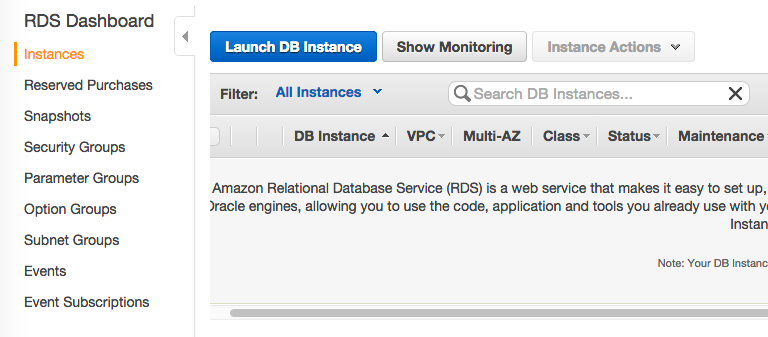
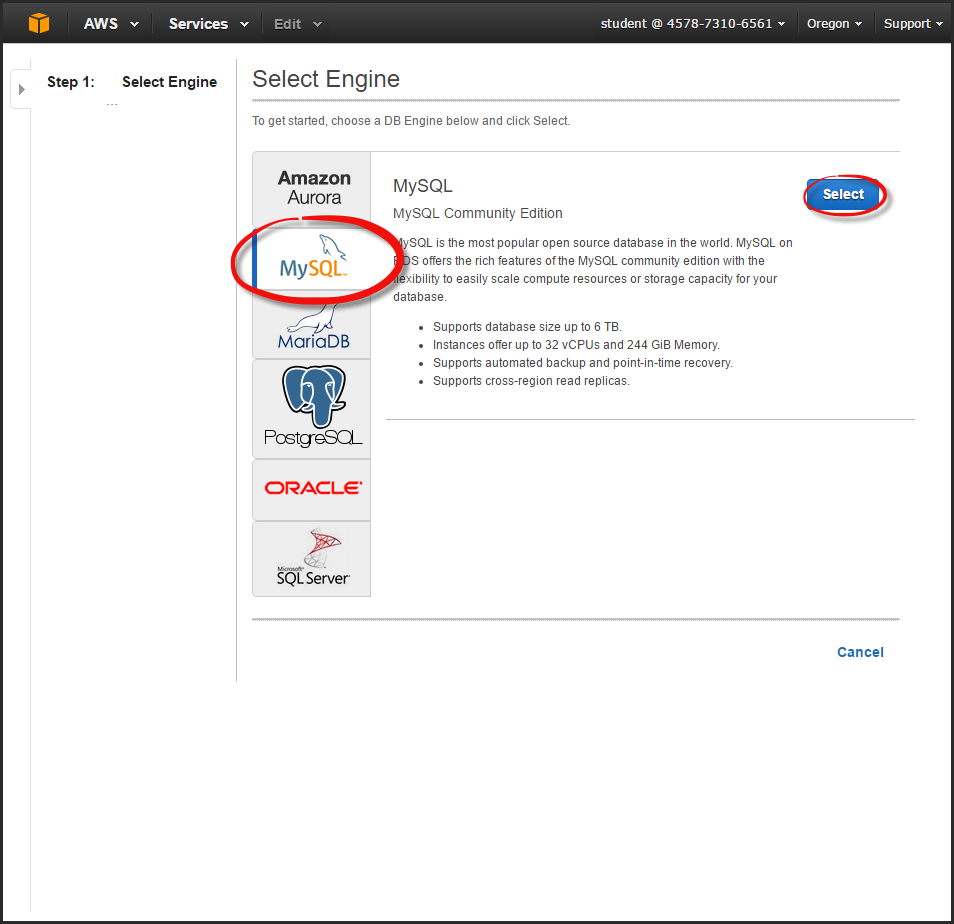
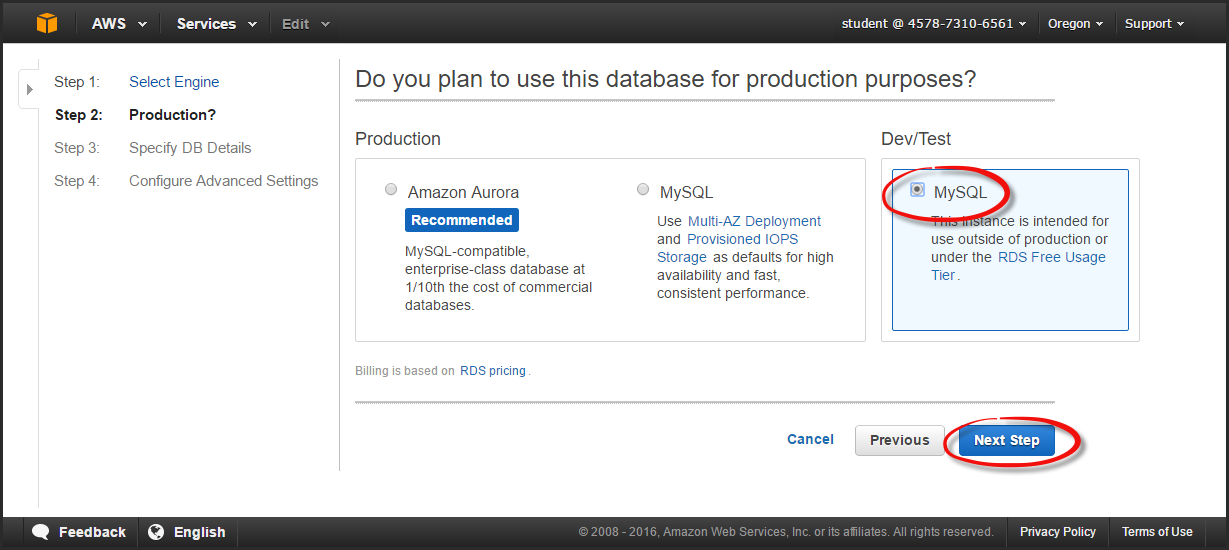
**Create a database cluster using RDS**

Amazon Relational Database Service (**Amazon RDS**) makes it easy to set up, operate, and scale a relational database in the cloud. Amazon RDS is designed for developers or businesses who require the full features and capabilities of a relational database or who wish to migrate existing applications and tools that utilize a relational database. It gives you access to the capabilities of a **MySQL**, **Oracle**, **Microsoft SQL Server**, **MariaDB**, **Amazon Aurora**, or **PostgreSQL** database engine.

The RDS service is fully managed by Amazon. RDS will make sure that the database software stays up-to-date with the latest patches and any faulty compute instance powering your database deployment will be automatically replaced in the event of a hardware failure.You can automatically or manually create database snapshots and easily scale your infrastructure up or down using the AWS Management Console.

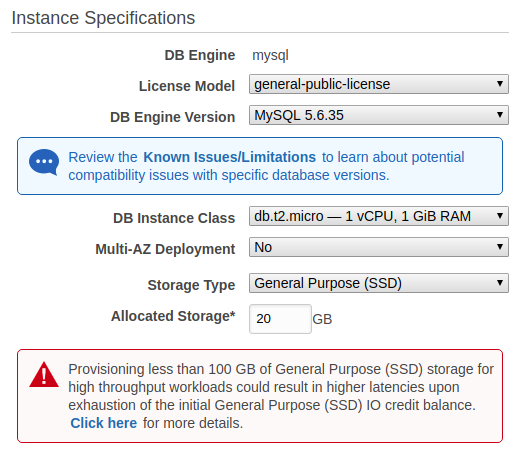
You can launch an RDS instance using the RDS launch wizard. Select the RDS service from the Management Console dashboard:

  
  
Select **Instances** from the left menu and click **Launch DB Instance.**

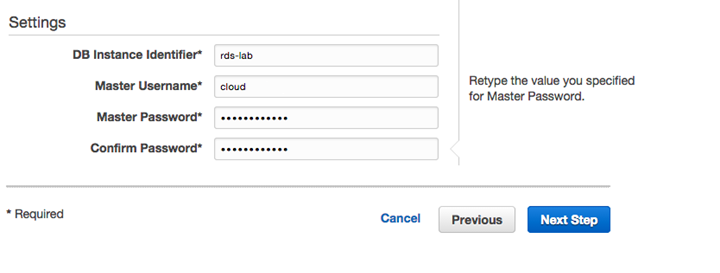
  
  
The **Launch DB Instance Wizard** appears and you must select the database engine. Choose the **MySQL** database engine and click the **Select** button.  
  
  
  
The next step asks if you are planning to use the DB instance you are creating for production purposes. By selecting Yes, the failover option, Multi-AZ and the Provisioned IOPS storage option will be preselected in the following step. For this exercise, please select **Dev/Test MySQL**. Then click **Next Step**.  
  


On the **Specify DB Details** page, you should select the following options:

* **License Model:** general-public-license
* **DB Engine Version:** select the latest one
* **DB Instance Class**: db.t2.micro
* **Multi-AZ Deployment**: No
* **Storage Type:** General Purpose (SSD)
* **Allocated Storage**: 20GB

  
  
Scroll down the page and continue configuring your RDS Instance by filling-in the following fields:

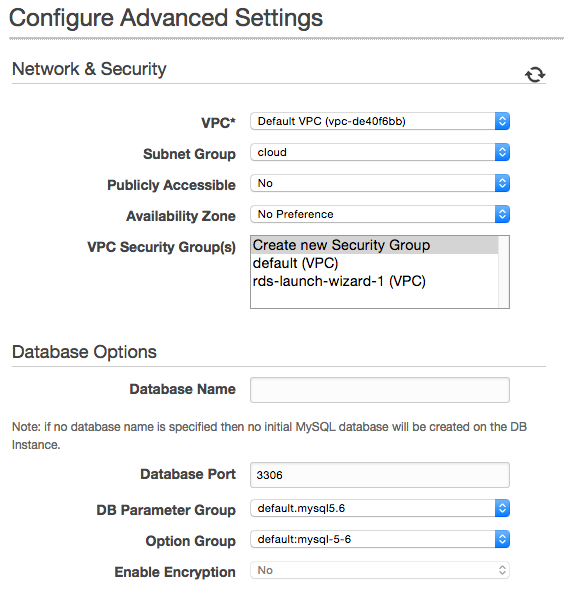
* **DB Instance Identifier**: rds-lab
* **Master Username**: clouduser
* **Master Password**: myStrongRDSpwd!

  
Click **Next Step,** and start configuring the Advanced Settings.

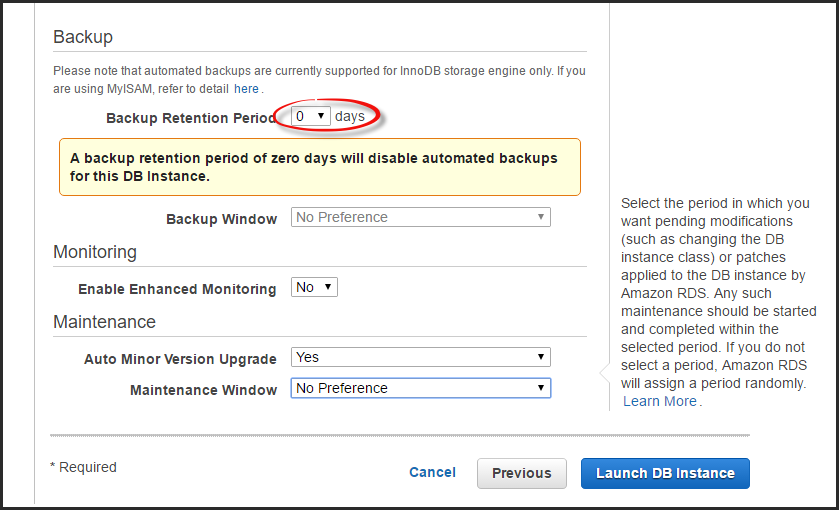
On the **Configure Advanced Settings** page, provide additional information that RDS needs to launch the MySQL DB instance.

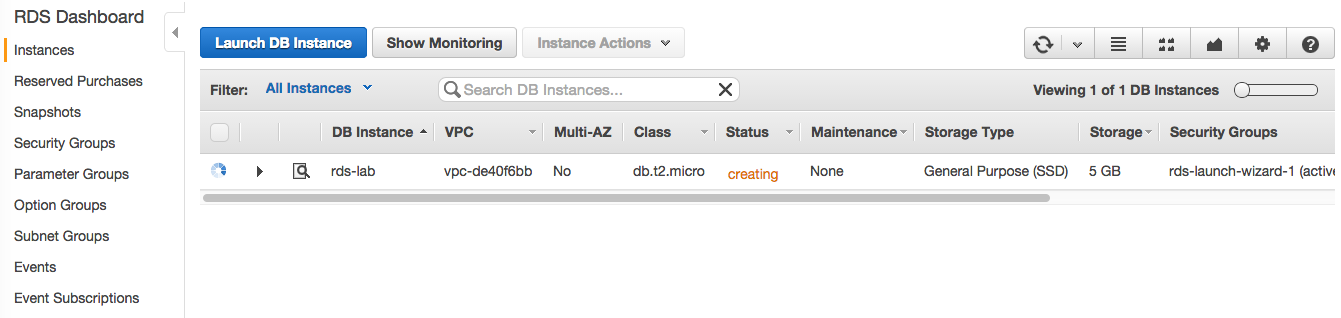
You need to select the following options:

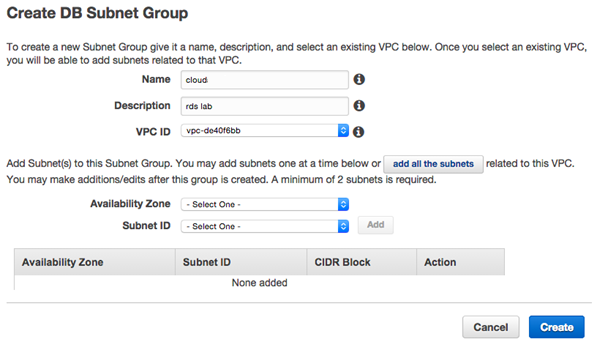
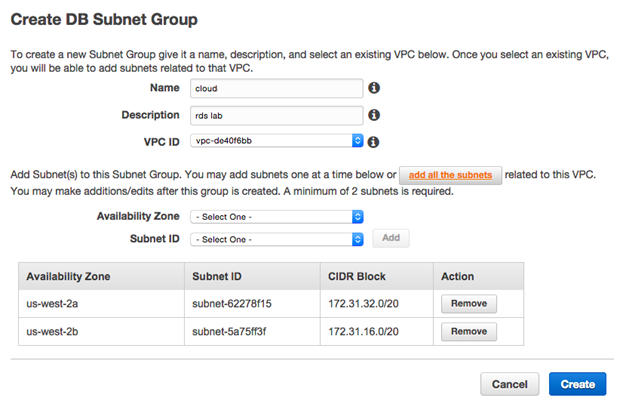
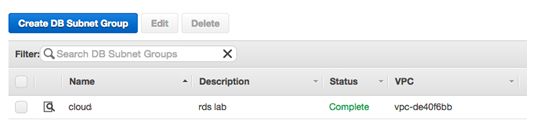
* **VPC**: select the available one
* **Subnet Group**: cloudrds
* **Publicly Accessible**: No
* **Availability Zone**: us-west-2a
* **VPC Security Group(s)**: Create new Security Group
* **Database Name**: rdsappdb
* **Database Port**: 3306
* **DB Parameter Group**: default
* **Option Group**: default
* **Enable Encryption**: No

  
  
Continue specifying the following settings for the **Backup**and **Mainteniance**options:

* **Backup Retention Period**: 0 days
* **Backup Window**: No preference
* **Auto Minor Version Upgrade**: Yes
* **Mainteniance Window**: No preference

  
  
Click **Launch DB Instance,** and then click **View Your DB Instances**.

  
  
**N.B.**, RDS instance creation requires **up to 10 minutes** for completion. Wait until its status becomes *available*.

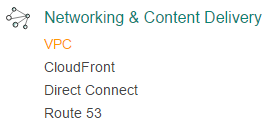
  
  
You must add the available subnets of the selected VPC. Click **add all the subnets**button for filling the subnet list and then click **Create**.  
   
  
After a few seconds, your DB Subnet Group will be available and ready for use.  
 

**Setup security group rules for connecting to the RDS instance**

In order to use the previously created RDS instance inside the VPC, you need to **add an inbound rule to the VPC Security Group** created during the RDS instance creation.

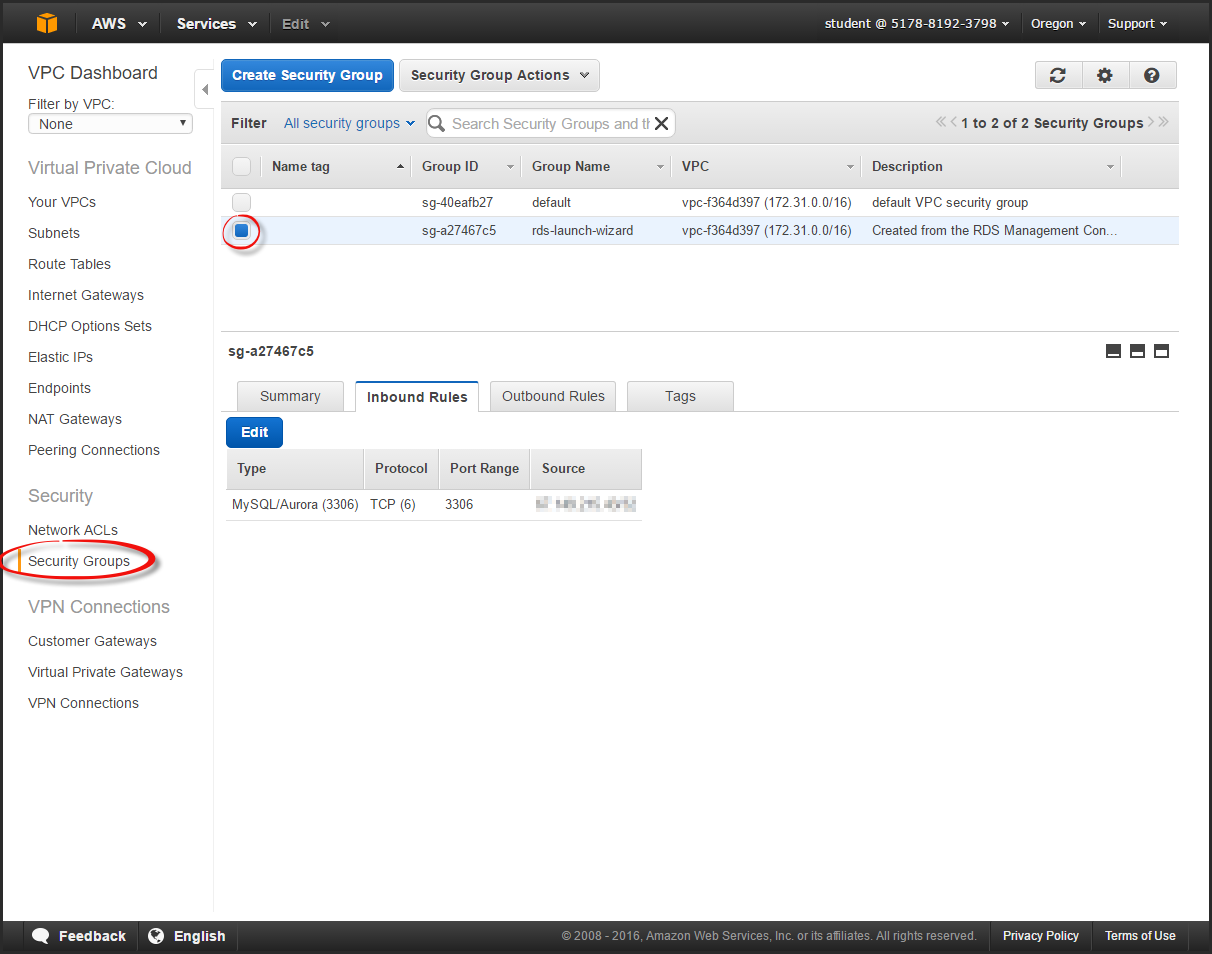
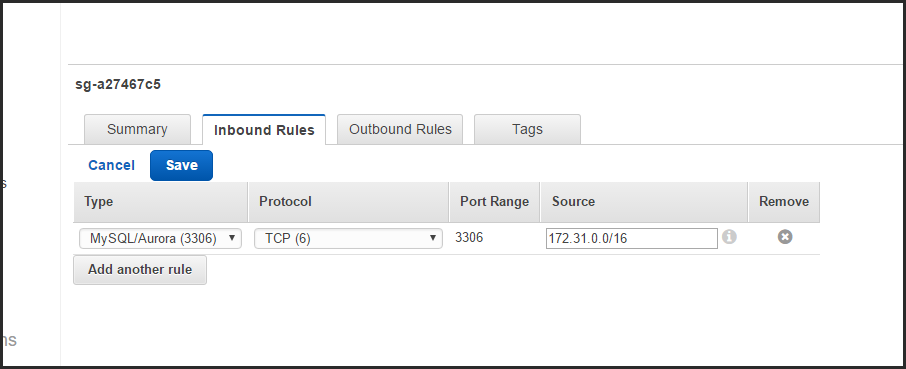
The rules of a **Security Group** control the inbound traffic that's allowed to reach the instances that are associated with the security group and the outbound traffic that's allowed to leave them. By default, security groups allow all outbound traffic and deny all inbound traffic.

You can add new rules to a **VPC Security Group** using the AWS Management Console.

Select the VPC service from the Management Console dashboard:  
  
[](https://assets.cloudacademy.com/bakery/media/uploads/lab-step/blobid2-9bc31829-78e6-4ef9-9b6e-709243aa592d.png)

Note: You've been working in the RDS console. Make sure you go back to the main console and select the **VPC** console.

In the navigation pane, click **Security Groups.** Locate and click the **rds-launch-wizard** security group.

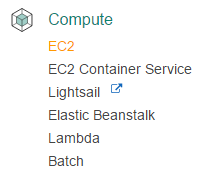
  
  
On the **Inbound Rules** tab, click **Edit**.  
  
  
  
Ensure that the rule is completed using the following information:

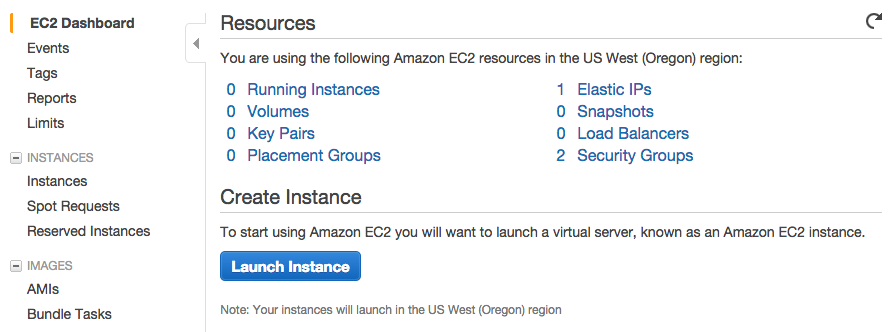
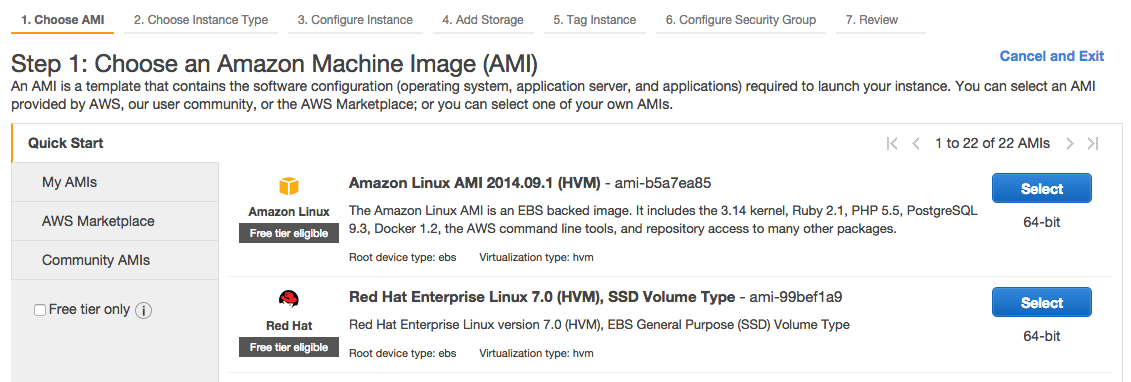
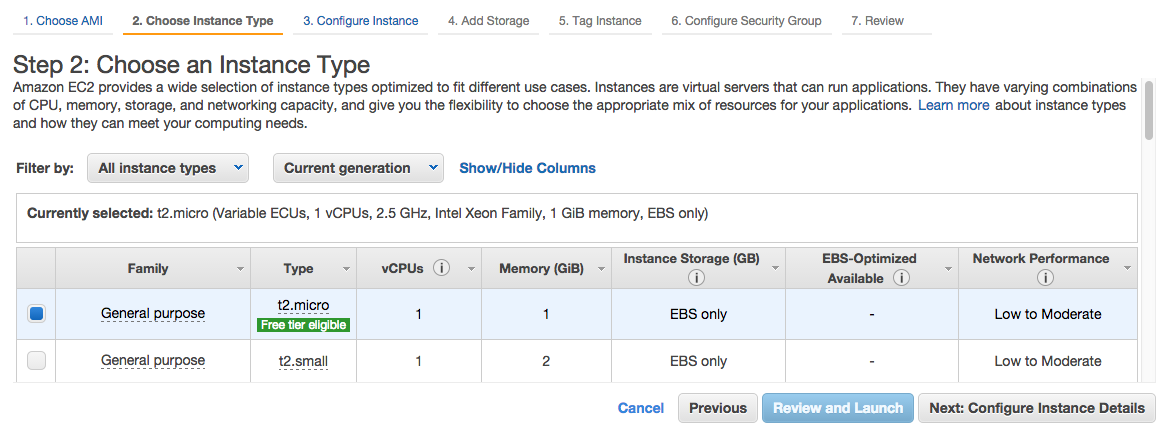
* **Type**: MYSQL
* **Protocol**: TCP
* **Port**: 3306
* **Source**: 172.31.0.0/16

Click **Save**,and you will be ready to connect to your RDS instance inside the VPC.

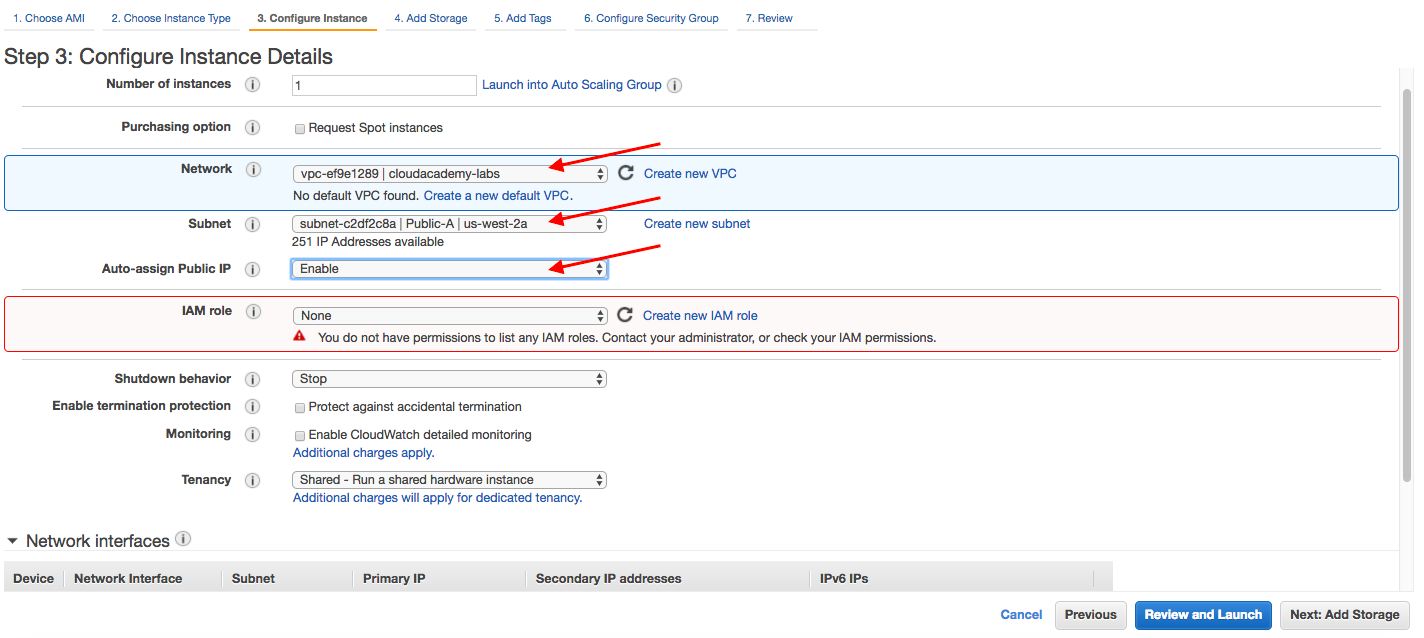
**Create an EC2 instance with a key pair**

You can launch an EC2 instance using the EC2 launch wizard.

Select the EC2 service from the Management Console dashboard:  
  
  
  
From the EC2 dashboard, click **Launch Instance**.

  
  
The **Choose an Amazon Machine Image (AMI)** page displays a list of basic configurations called **Amazon Machine Images (AMIs)** that serve as templates for your instance. Click **Select** for the first listed 64-bit Amazon Linux AMI:  
  
  
  
On the **Choose an Instance Type** page, you should **not** change any options**.** Simply make sure the default **t2.micro** is selected:

Click the **Next: Configure Instance Details** button. On this step of the wizard, make sure the following is configured:

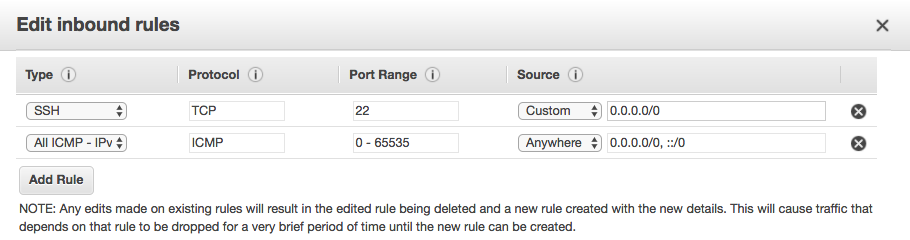


* **Network**: Make sure the labs is selected for the VPC. (It is ok if no default VPC is found.)
* **Subnet**: Select Public-A | US-west-2a
* **Auto-assign Public IP**: Select Enable
* **IAM role**: Notice that you do not have permissions to list IAM roles. That message is expected and ok. The student account has restricted privileges but the Lab will work fine without listing IAM roles.

Click **Next: Add Storage** when ready to proceed.

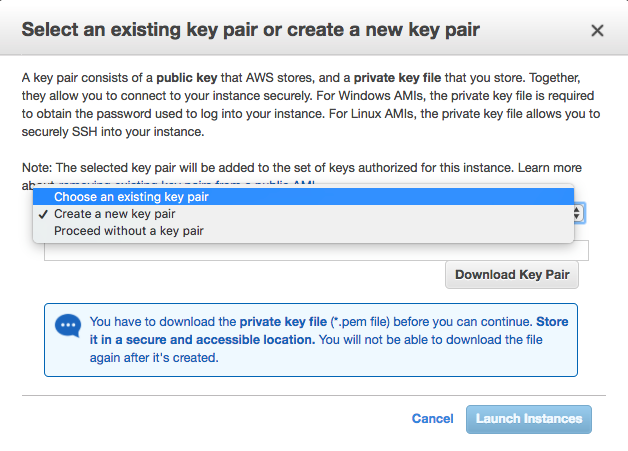
No changes are needed for storage so click **Next: Add Tags** to advance the wizard again.

No changes are needed for tags, so click **Next: Configure Security Group** to advance the wizard again. By default, an SSH rule is added for you. However, you may want to add one rule in order to enable a ping test.  Click **Add Rule** and add the following ICMP rule so you can run a simple ping test later. (*Note*: In many production environments, allowing ping/ICMP communications is not even permitted. However, it is often allowed during initial setup and testing.) Your security group rules should look similar to the following:



Click **Review and Launch**, and then finally **Launch**when ready to proceed.

In the **Select an existing key pair or create a new key pair** dialog box, the drop-down menu has three options:



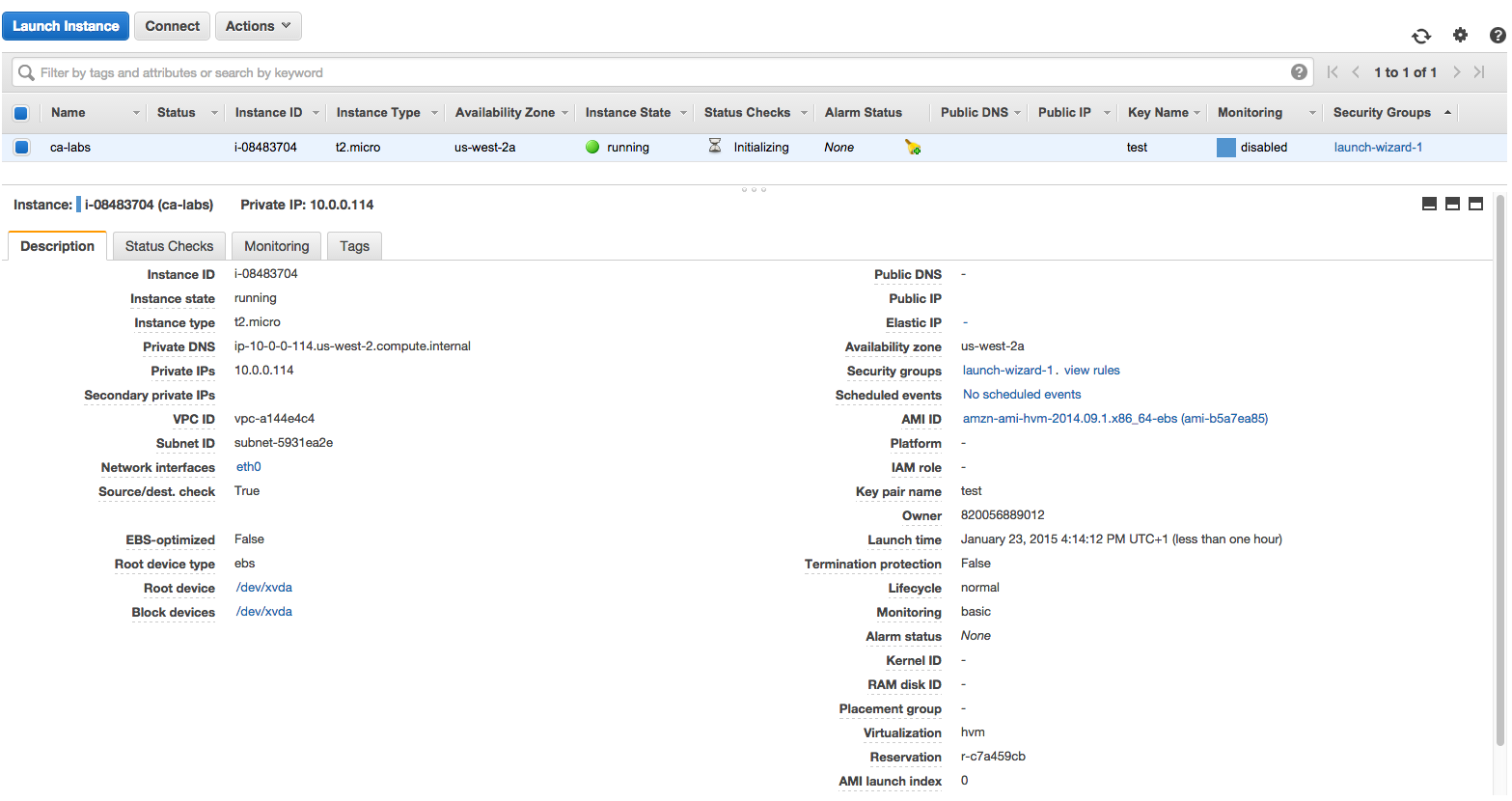
Depending on the Lab, you will choose a different option. The three options are primarily for the following purposes:

1. **Choose an existing key pair**: Works best for Labs that use the PEM/PPK key pairs generated for you) - Instructor
2. **Create a new key pair**: AWS will create a key pair for you to download and use
3. **Proceed without a key pair**: Only use this option when you know you will *not* need to SSH into the instance. This option also requires you to acknowledge you won't be able to connect to the instance.

For this Lab, select **Create a new key pair**. Name it something descriptive and memorable, then download it to your local file system. (In a later Lab Step you will use the keys to SSH into the instance.)

A confirmation page will let you know that your instance is launching. Click **View Instances** to close the confirmation page and return to the console.

On the Instances Screen, you can view the status of your instance. It will take a short time for your instance to be launched. When you launch an instance, its initial state defaults to *pending*. After the instance starts, its Instance State changes to *running*.



Once your instance is running, as a quick communications test from a terminal window, enter the following test:

$ ping *<PublicIPAddress>*

PING 34.214.70.142 (34.214.70.142): 56 data bytes

64 bytes from 34.214.70.142: icmp\_seq=0 ttl=235 time=63.077 ms

64 bytes from 34.214.70.142: icmp\_seq=1 ttl=235 time=47.070 ms

64 bytes from 34.214.70.142: icmp\_seq=2 ttl=235 time=46.933 ms

*Hint*: The public IP address is listed on the **Description** tab of your running instance. Use Control-C or Command-C to stop the ping test.

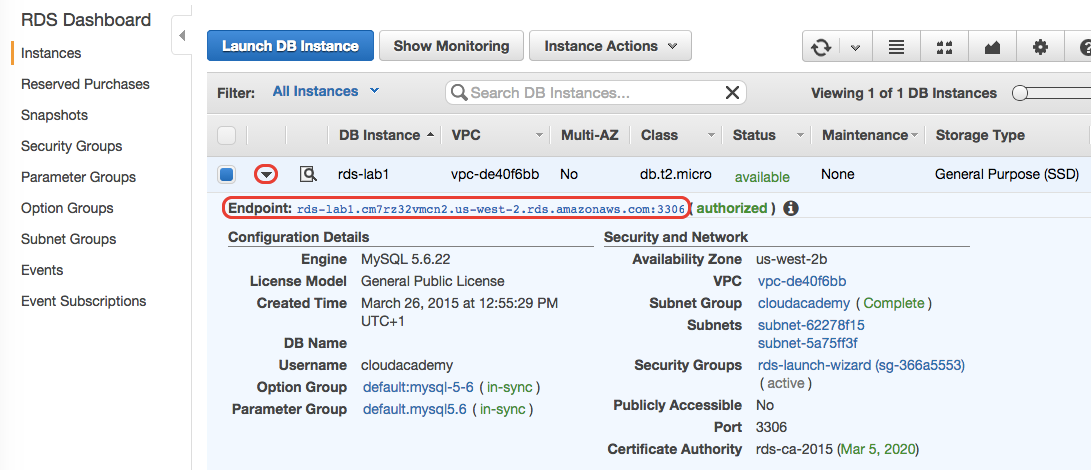
## Connect to RDS and create a database table

Your RDS instance is ready and accessible from any EC2 instance created within the same VPC, so you can connect to the previously created EC2 instance and use the database.

The MySQL client software isn't usually installed in the Linux Amazon AMI, but you can download and install it using the YUM package manager.

Execute the following command: sudo yum -y install mysql

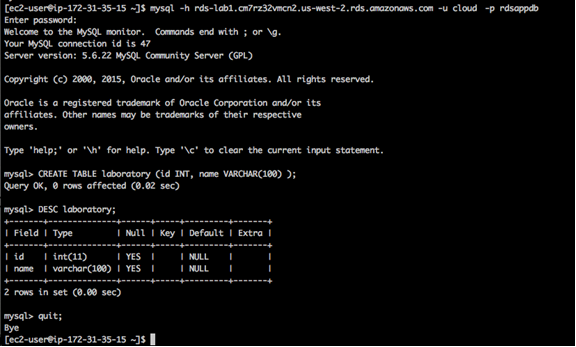
You are ready to connect to your RDS instance using the **RDS Endpoint URL**. It's automatically generated during the RDS instance creation and you can find it in the RDS instance details box.

  
  
In order to complete this lab step, you must connect to your instance, using the RDS Endpoint URL, and create a table named "laboratory" in the database "rdsappdb".

Connect to your RDS instance by using the mysql client and replace your.endpoint.aws.com with your real endpoint URL (w/o including the host port): mysql -h your.endpoint.aws.com -u cloud -p rdsappdb

Insert the DB Master Password (myStrongRDSpwd!) and the MySQL console will be ready to accept SQL queries.

**Note**: remember **NOT** to include the port number of your endpoint URL (e.g. ":3306") or you will not be able to connect to the remote host.

  
  
Create a new table in the "rdsappdb" database writing CREATE TABLE laboratory ( id INT, name VARCHAR(100) ); and then press ENTER.

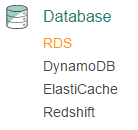
You can check if the table was successfully created using the SQL command: DESC laboratory;

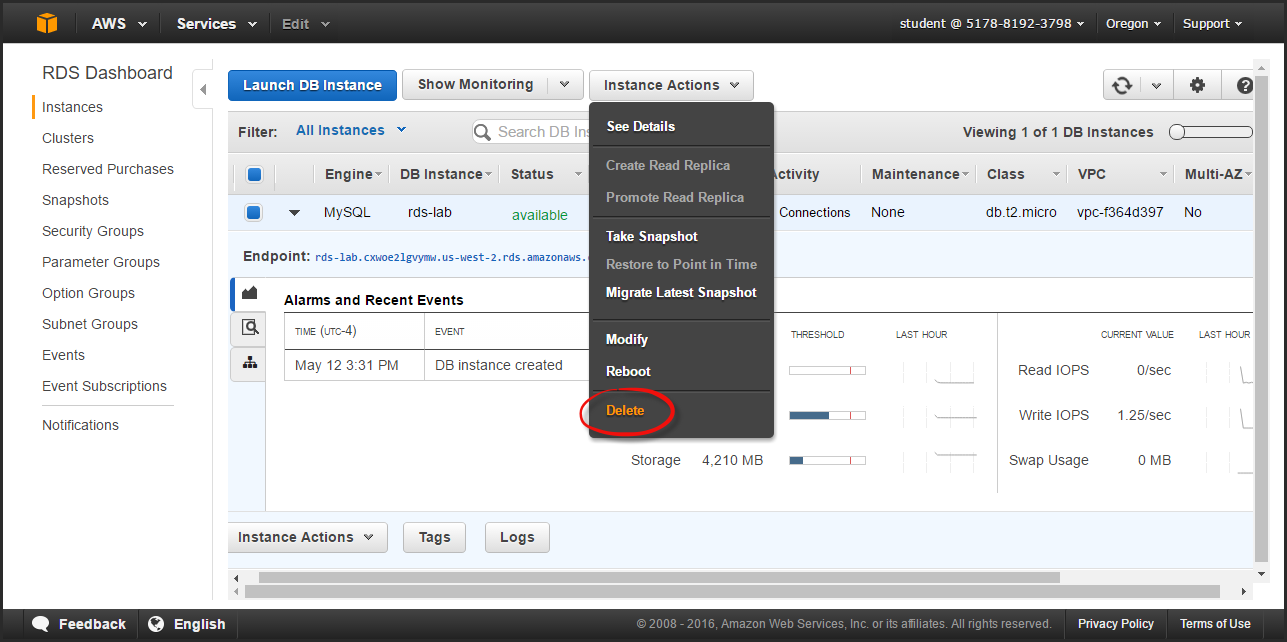
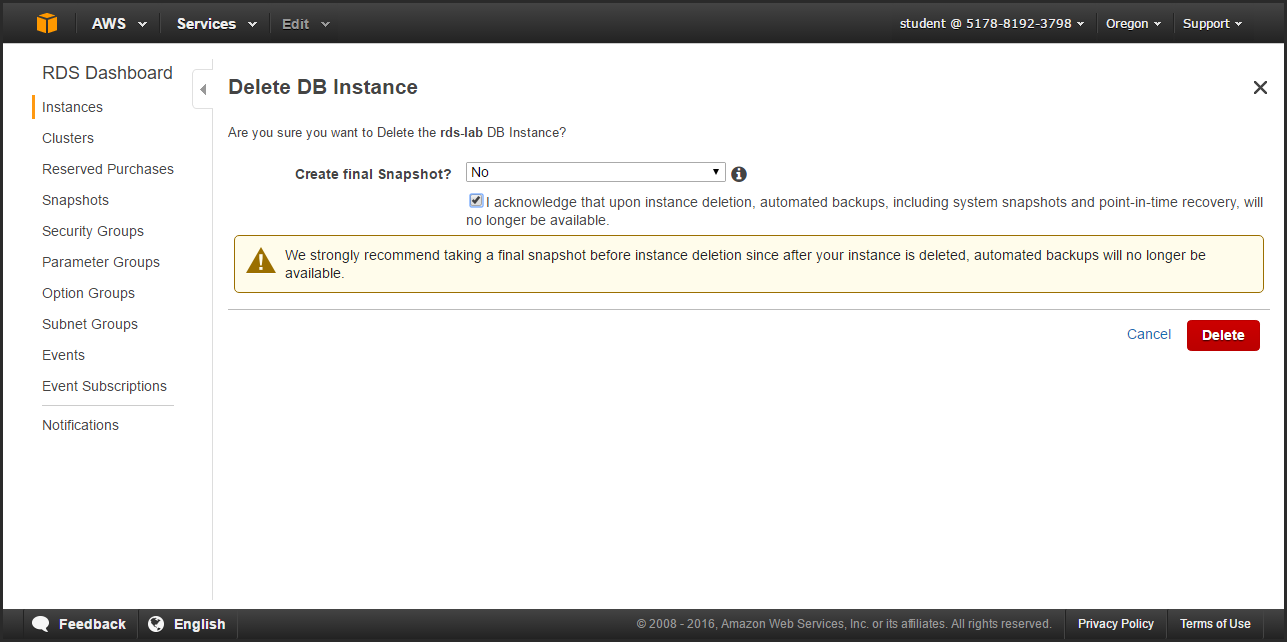
Close your database connection writing the command: quit;

## Destroy an RDS instance

You can delete an RDS Instance using the AWS Management Console.

Select the RDS service from the Management Console dashboard:

  
  
From the RDS dashboard, click **Instances** from the left menu.

Select rds-lab from the RDS Instances list. Click on the **Instance Actions** button, then select **Delete**.  
  
  
  
On the Delete DB Instance page, answer **No** to the Create final Snaphot? query. Check the acknowledgement checkbox, and click **Delete**.  
  
  
  
Your RDS instance is now in the **deleting**status, which will take a few minutes to complete.