Using Amazon RDS to create MySQL Database

Amazon Web Services (AWS) provide users with the ability to easily create their own database instance using Amazon Relational Database System (Amazon RDS) service. This guide will illustrate how to set up a MySQL relational database through AWS, and connect to it using Eclipse.

Required Materials

- Amazon Web Services (AWS) Account
- Eclipse Enterprise Edition
- MySQL Database Connector for java .jar file

Suggested Materials

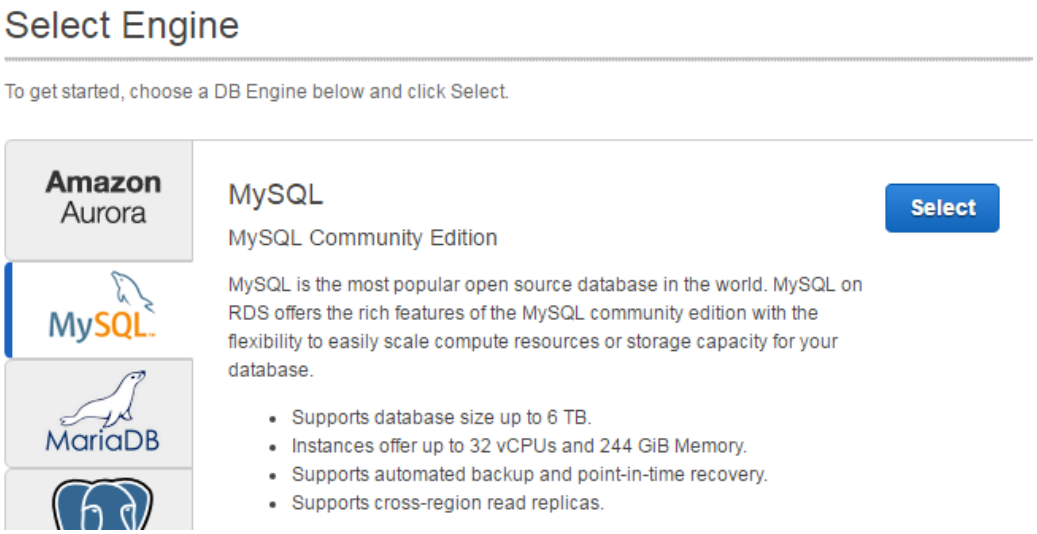
- Cisco VPN for GMU* (installation instructions available here: https://itservices.gmu.edu/services/view-service.cfm?customel_dataPageID_4609=6169)

*The internet connection in some buildings on GMU's campus (such as Innovation Hall) is configured to prevent you from pinging the AWS RDS described in this guide. Using the VPN will bypass this issue, and allow users to ping the database, enter queries, and access the RDS through programs while on the GMU campus.

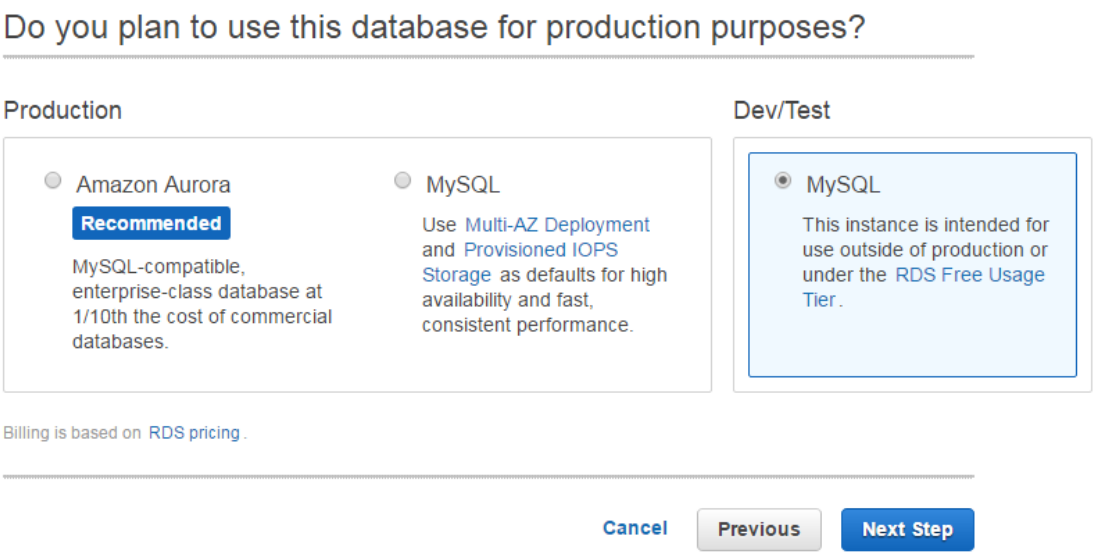
Part I: Setting up the MySQL RDS

1. Log in to the AWS Management Console and select **RDS** from the list of AWS Services
2. Click on the Launch a DB Instance button. 

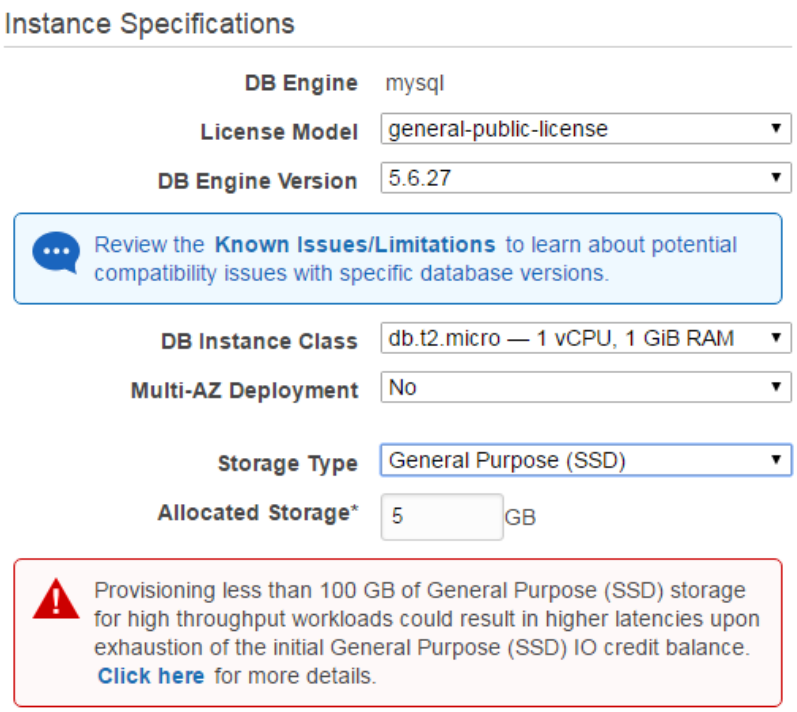
3. On the Select Engine page, choose **MySQL** and click the Select Button.



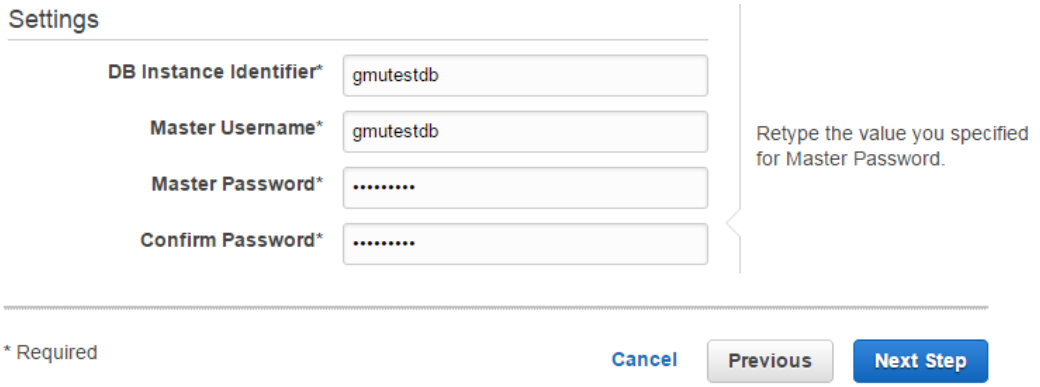
4. On the next page, select the option under **Dev/Test** for use with the AWS Free Usage Tier, and click **Next Step**.



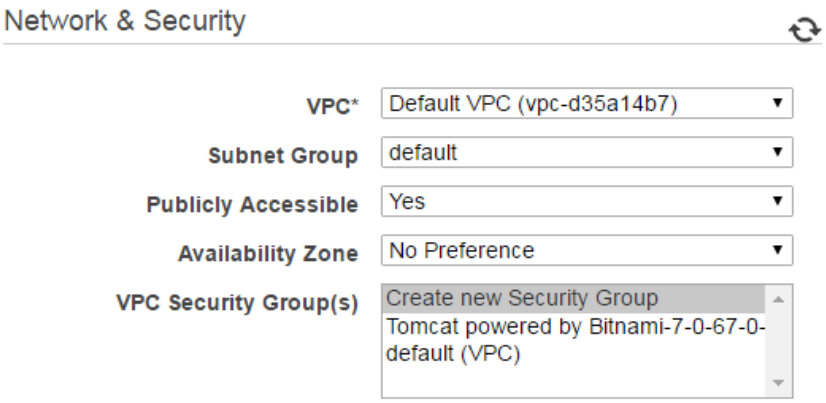
5. You will now specify the details of your database. To use Free Usage Tier options, you will select **db.t2.micro – 1 vCPU, 1GiB RAM** for the Instance class. This is the least powerful database available. Similarly you will select **No** for the Multi-AZ Deployment field. Ignore the warning message about provisioning 100 GB of storage space since it does not apply to you.



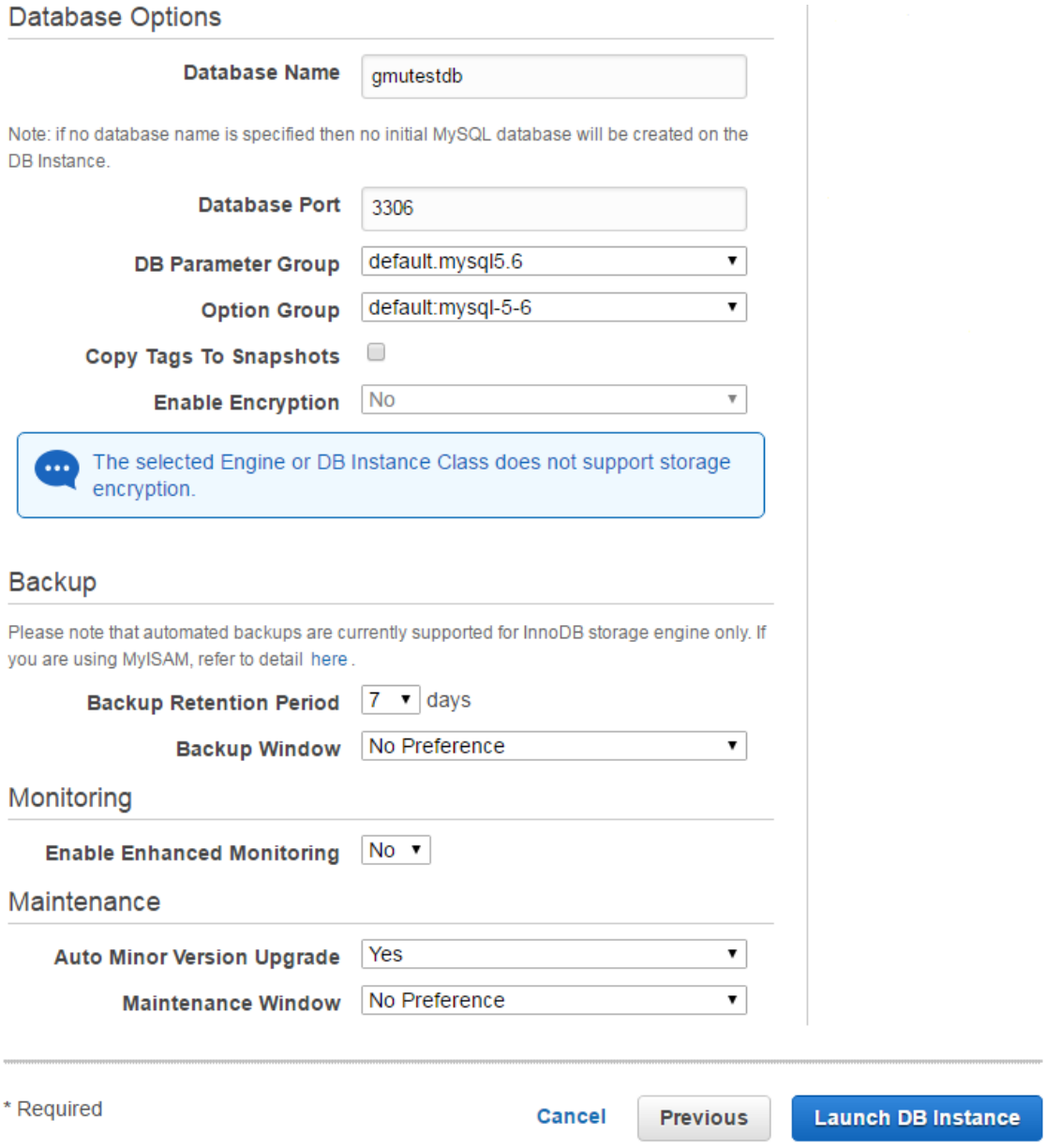
6. On the same page, you will need to declare a name to identify your database, along with a username and password for accessing it. In this guide we will use **"gmutest"** for all three purposes as shown below. After you fill out this information in Settings, click **Next Step**.



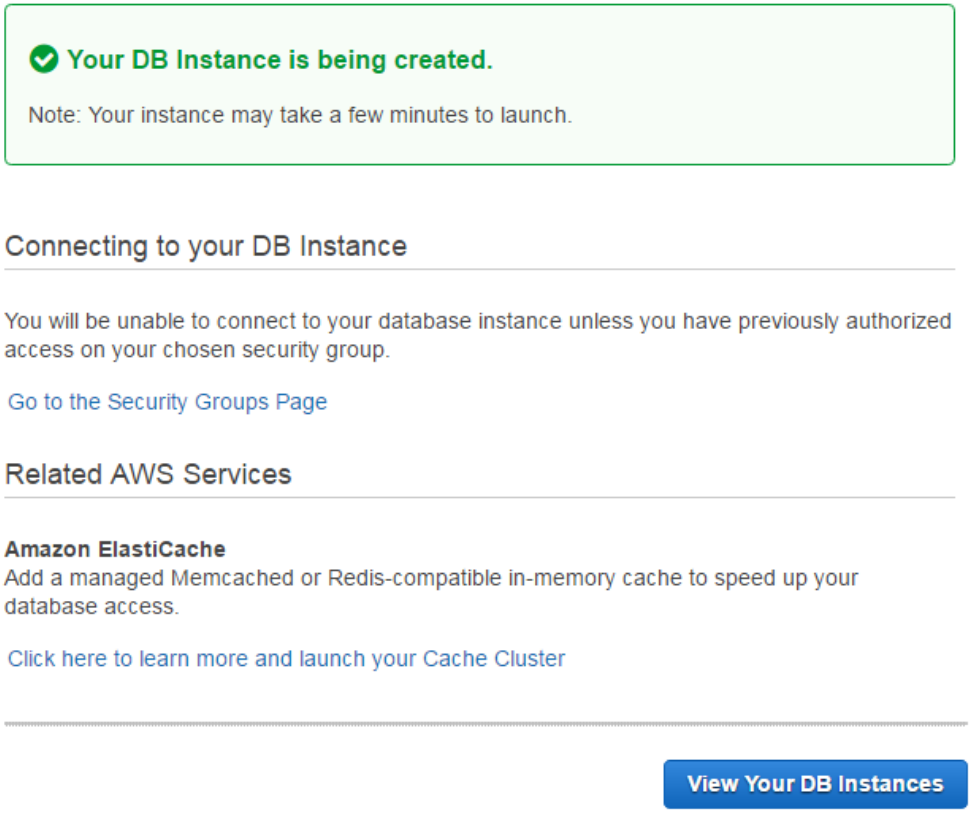
7. On the Advanced settings page the VPC Security Group(s) field will allow you to configure the security of the RDS using Amazon's security groups. We will be selecting **Create new Security Group** instead of relying on any previous security groups that may or may not be set up.



8. Under Database Options on the same page, we will copy the DB Instance Identifier, "**gmutestdb**", for the Database Name. No changes will be made to the backup settings, so click **Launch DB Instance** after making the change.

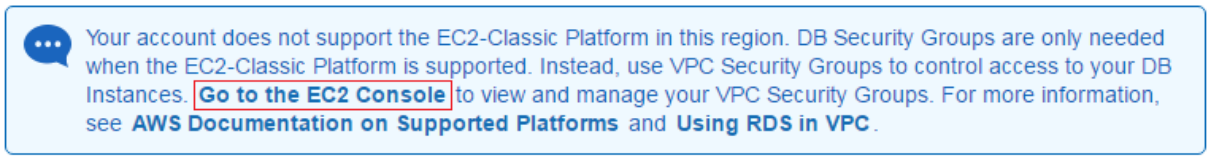


9. After clicking the launch button, you will be taken to a page indicating that your DB Instance is being created. Because we need to configure access to the RDS, click on **View Your DB Instances** at the bottom of the page.

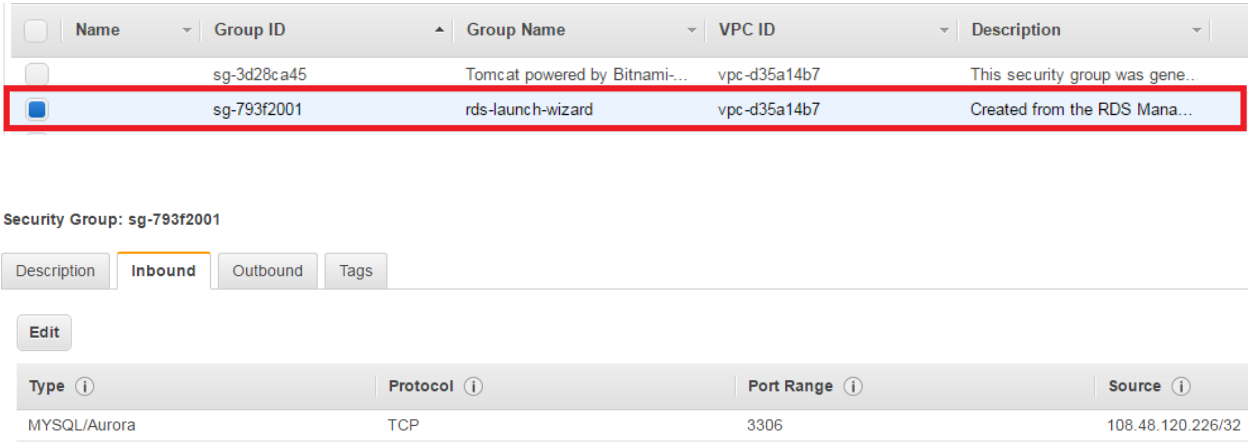


10. You will see that the status of your database says "creating" at first. The status will eventually change to "backing-up", and then to "available" when it is ready to use. You will also see an **Endpoint**, which is part of the internet address needed to make the database connection. The Endpoint will be discussed in Part II, Step 9 when we connect to the RDS with Eclipse EE.

11. Now we need to configure the security group so that our database can be accessed by Eclipse EE from different IP addresses. Click on **Security Groups** on the left hand side of the screen. You should get a message stating that you need to go to the EC2 Console to configure your security groups. Click on the hyperlink in that message.

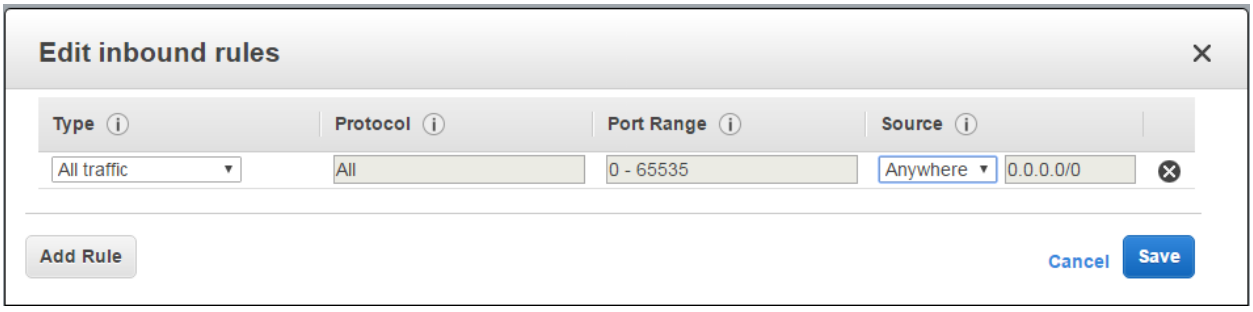


12. You will be taken to a screen that lists all of your current security groups. Click on the security group that was just created from the previous steps. It will have a Group Name similar to **rds-launch-wizard**. The bottom of the screen will describe the security rules in place for the RDS, and inbound traffic will originally be limited to just one IP address. You will need to make the inbound rules less restrictive.



13. Click on the **Edit** button at the bottom of the page under the Inbound tab (as shown above). You will be presented with a dialog window to edit the rules for accessing the database. We will not concern ourselves with security since this is purely for illustrative purposes. As such, we will be opening up our RDS to all types of traffic, on all protocols, on all ports, from any IP address. Use

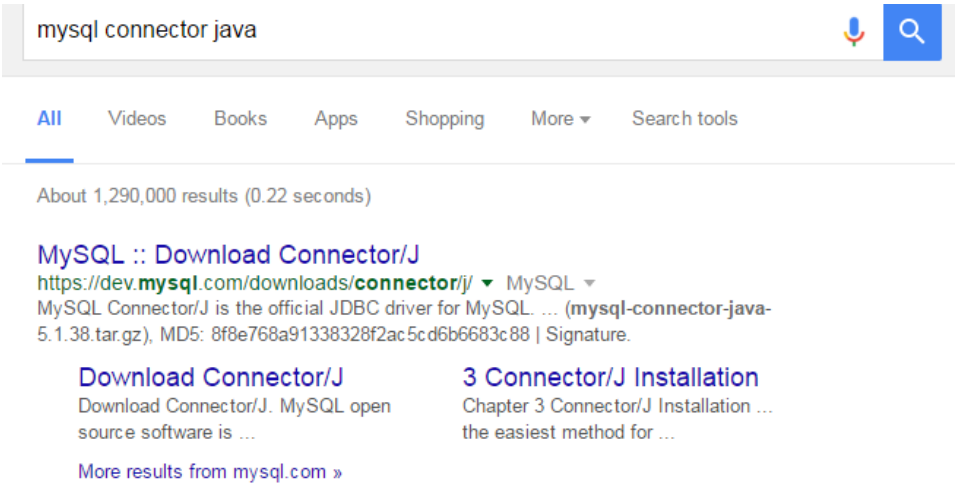
the drop down menus to accomplish this. The modified access rules should look like the following picture when entered. Click **Save** when you are done.



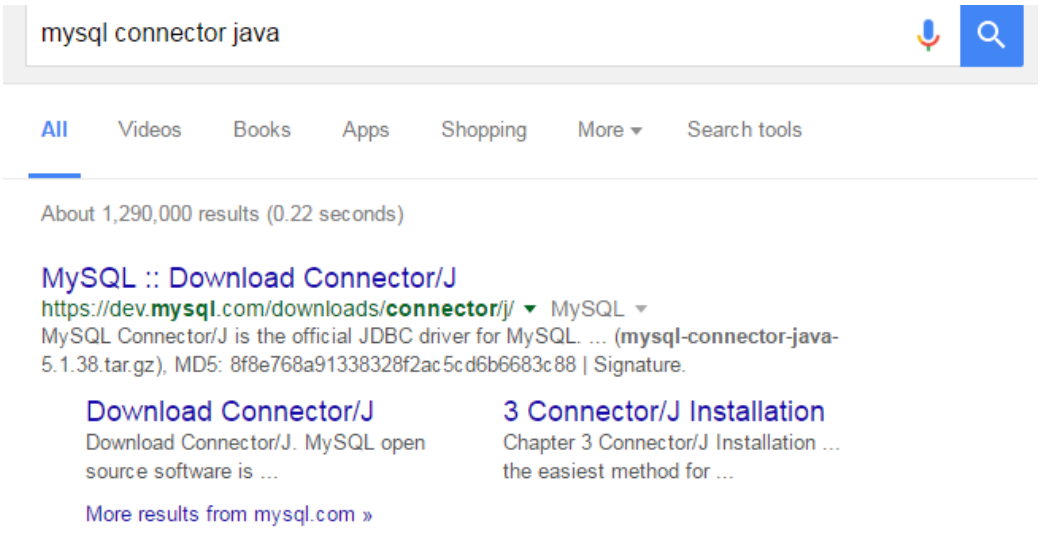
14. At this point you have successfully set up a MySQL RDS for remote use through AWS. The next section will involve accessing the RDS through Eclipse EE.

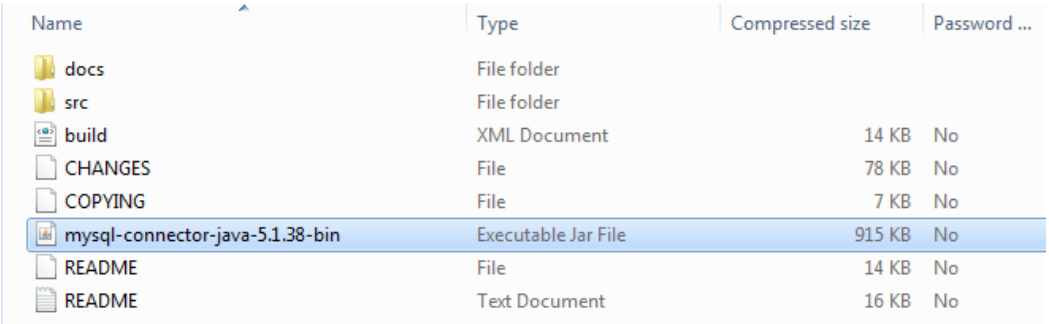
Part II: Connecting Eclipse EE to your MySQL RDS

1. If you have not already done so, download the MySQL Connector for java. A Google search for **mysql connector java** will yield a useful download link as the first option.

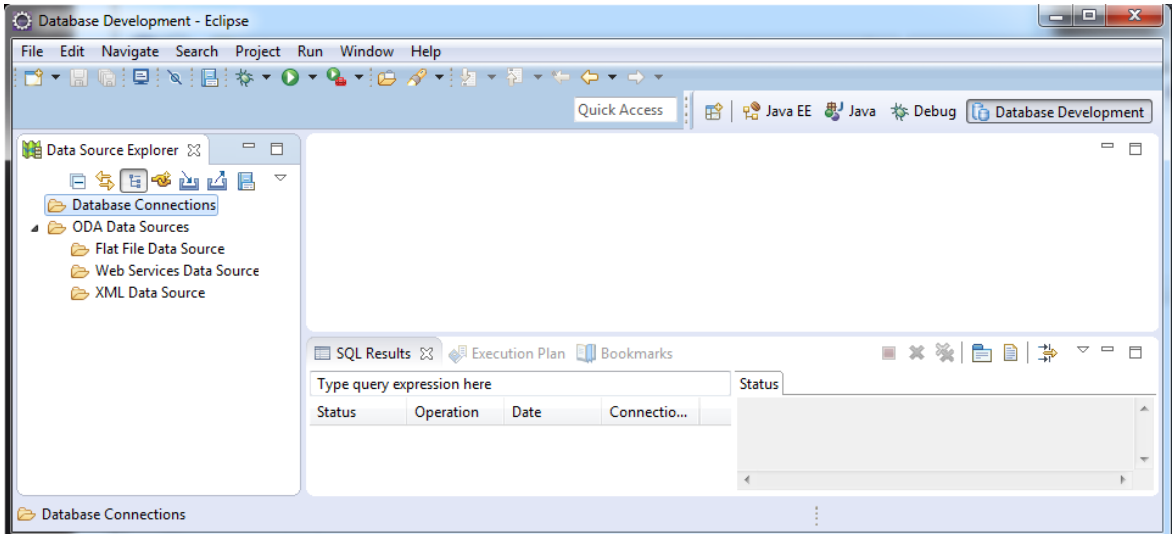


2. Click on the search result and download the zip file at the following page. Unzip the file, and place the .jar file in a place you will remember.

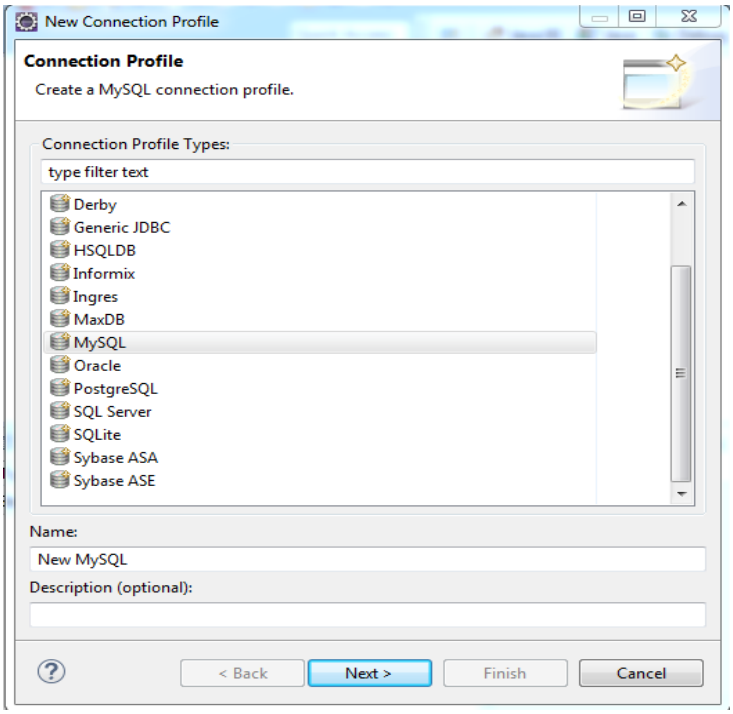




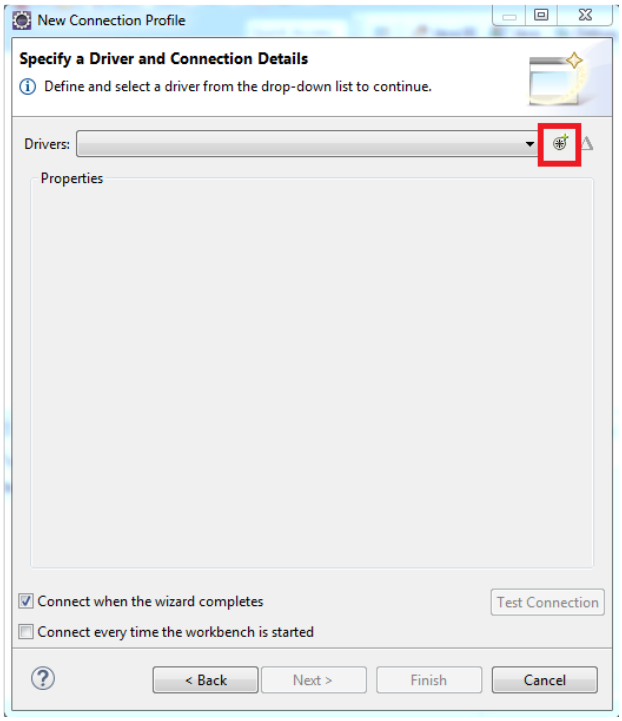
3. Open Eclipse EE. In the top menu, select **Window → Perspective → Other**. Select **Database Development** from the list of available perspectives. Your Eclipse screen should look like the following screenshot.



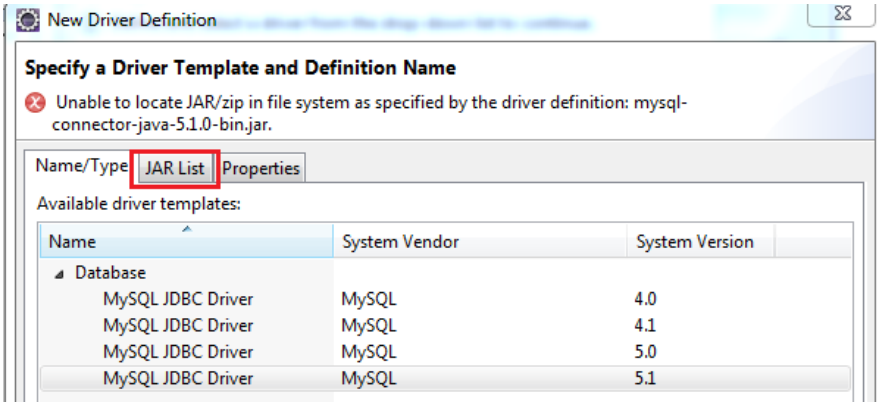
4. Inside the Data Source Explorer on the left hand side of the screen, right click on Database Connections and select **New**. This will present you with a Window for creating a new Connection profile. We are working with a MySQL database, so select **MySQL** in the list, then click **Next**.



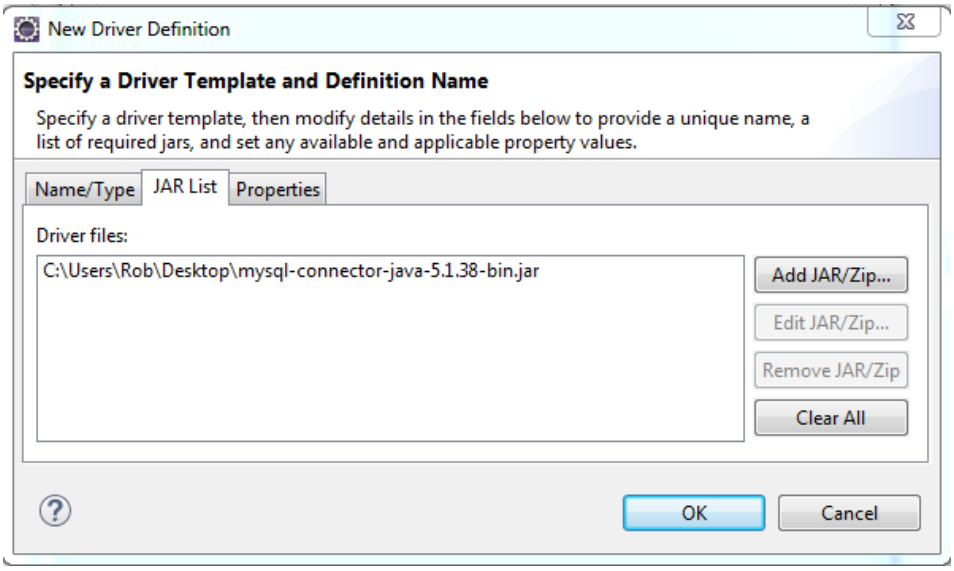
5. On the next screen you will see a prompt to specify the driver and connection details. This process will be used to integrate the previously downloaded MySQL Connector .jar file into Eclipse. Click on the **small circular icon** next to the drivers list to begin the process.



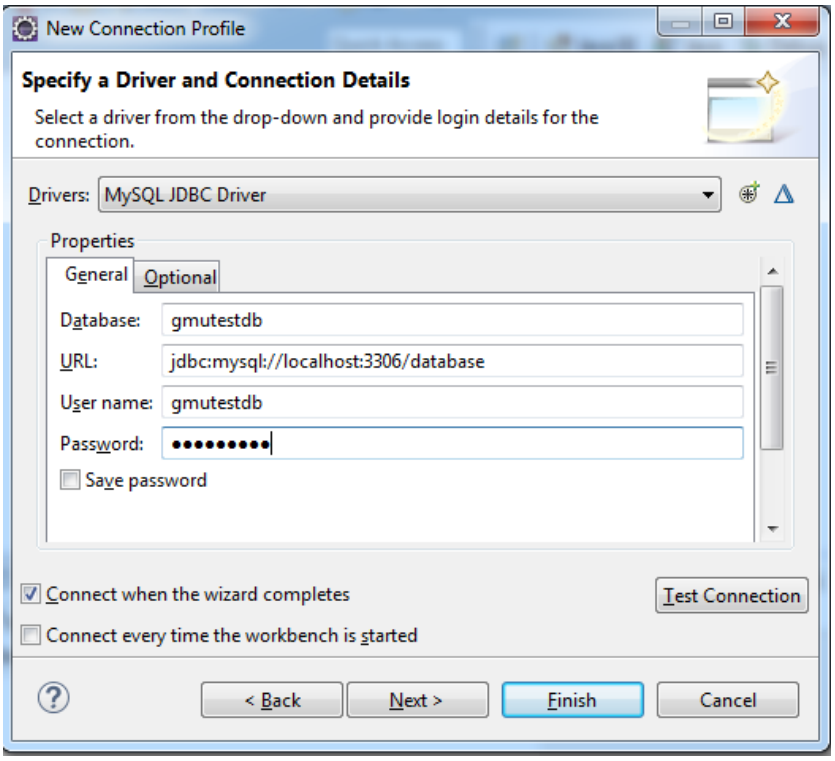
6. Select the latest version of the MySQL JDBC Driver from the list. There will be an error message saying that the required jar cannot be found. To resolve this issue click on the **JAR List** tab at the top of the screen.



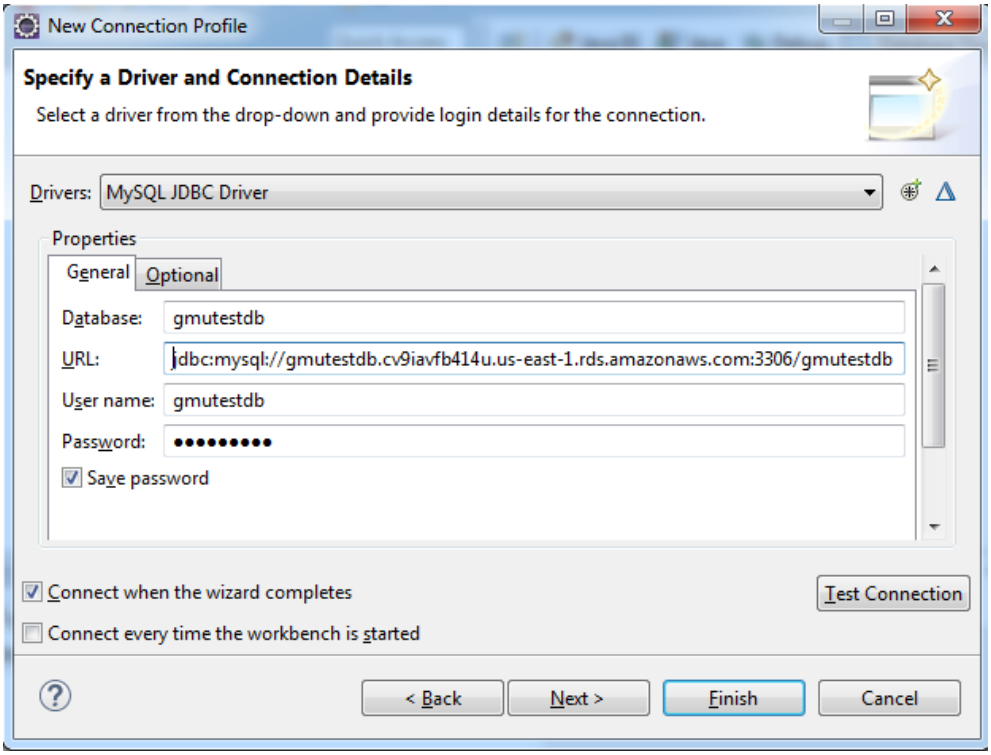
7. Click on the **Add JAR/Zip** button, and navigate to the MySQL Connector .jar file that you downloaded. You can remove the other, unhelpful driver file from the list since it is not being used. When the results look like the next screenshot, click **OK**.



8. The following screen will be used to ping the database and test for a successful connection. It is filled with default values that will need to be replaced. For the Database field, use the name of the database that was chosen in Part I, Step 8. In our case the name “gmutestdb” was also used as the user name and password, so those fields should be filled in as well. We are still not done because the URL needs to be modified, but the current result should look like the following screenshot.



9. The URL will take the following format: [**jdbc:mysql://ENDPOINT/DATABASE_NAME**]. Essentially, you need to replace two things with the default URL above. Replace "localhost:3306" with the Endpoint from your Amazon RDS. It was highlighted blue in Part I, Step 10, and can be seen when looking at the details of your RDS instance in the AWS management console. Additionally, you need to replace "database" with the actual name of the database from Part I, Step 8. You can also go ahead and **save the password** for future convenience. The resulting setup should look like the following screenshot.



10. With all of the setup completed, you can click on the **Test Connection** to ping the database. If everything is successful, you will get a message like the one below. If you are on the GMU campus, turning on your VPN might be required to successfully test the connection.