Task B: The Platform

Chris Dusyk - 200268705

Amazon Aurora

Aurora is AWS' custom relational database engine. It uses the MySQL structure, attached to AWS' new database engine. The database engine makes use of Amazon's distributed cloud services platform as opposed to a traditional database server. The engine itself is sharded across many virtual processors, allowing for very high performance and throughput.

Structure

Aurora uses the MySQL database structure and language on top, making it easy to port an existing MySQL database over to it. Aurora follows all standard MySQL syntax and design, so any new database is handled like a MySQL database.

Underlying Structure

Under-the-hood, Aurora is very different than any current database engine. It's designed around the fact that it's hosted in large datacentres where the database workload can be distributed across many virtual processors. Unfortunately, specific details are hard to come by as this is Amazon's proprietary engine, and is directly competing with other products from Microsoft Azure and Google Cloud.

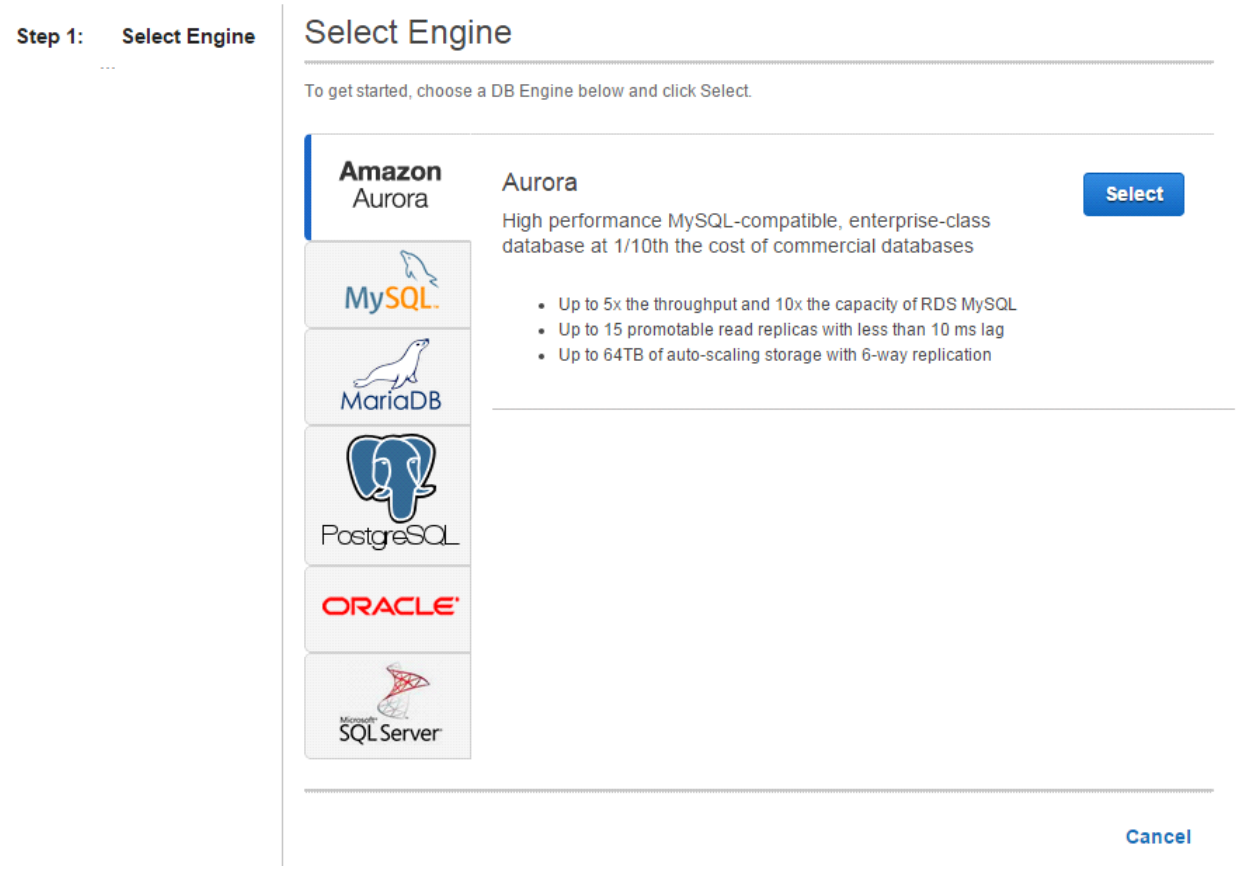
Getting Set Up

Create an AWS Account

AWS uses Amazon accounts, so if you already have an Amazon account, AWS will use that. To create an account, or login, go to http://aws.amazon.com.

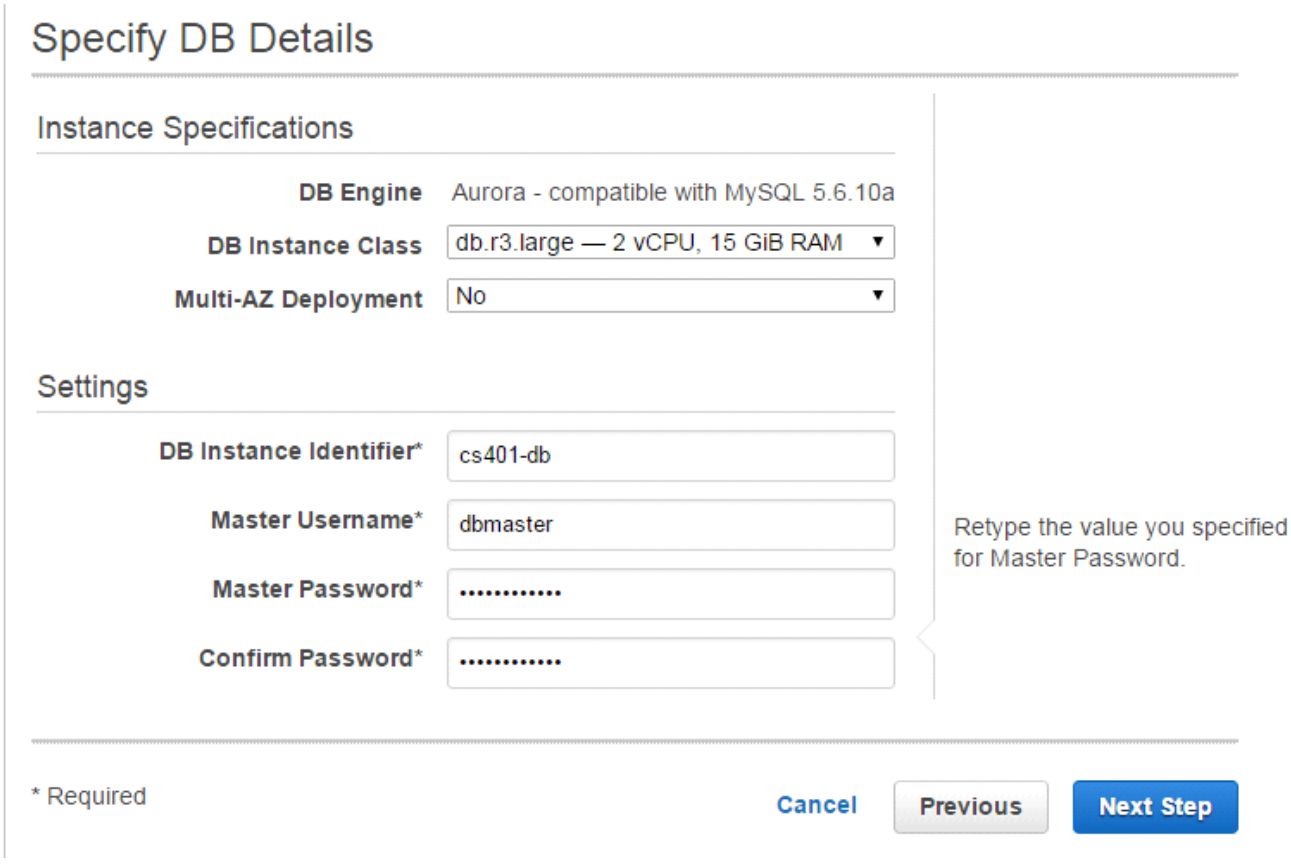
AWS Console

Once logged into the AWS Management Console, go to Services > RDS. This will take you to the RDS Dashboard. From here, you can launch a DB Instance. AWS has instance support for Aurora, MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server. For this assignment, we will use Amazon Aurora:



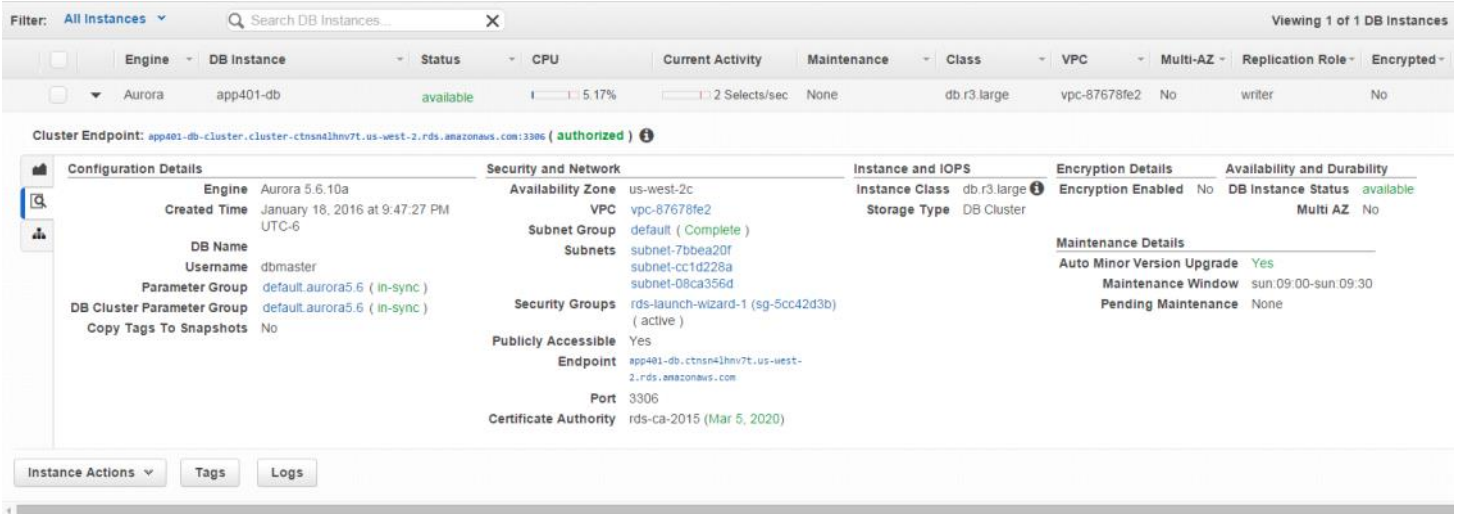
Once Amazon Aurora is selected, you will be taken to the next setup step. Here you will choose the Instance Class, Multi-Availability Zone

Deployment, the instance identifier, and the master user. The Instance Class is the size and power of the instance. For a small database like this example I used db.r3.large (2 vCPU, 15 GB RAM) as that was more than sufficient for the database I needed. Multi-AZ Deployment allows your database to be replicated across 3 availability zones such that if anything happens to the health of your primary instance, there will be immediate failover to one of the replicated instances, so that service is not interrupted. The DB Instance Identifier is the name of your database instance, such as 'cs401-db'. The Master Username and Password are the master credentials for the database:



Once Next Step is clicked, you will be taken to the Advanced Settings page. You can leave these at default and click Launch DB Instance.

Once the instance has finished setting up, you can go to the Instances tab in RDS. Here you will see all of the general information on your DB Instance.

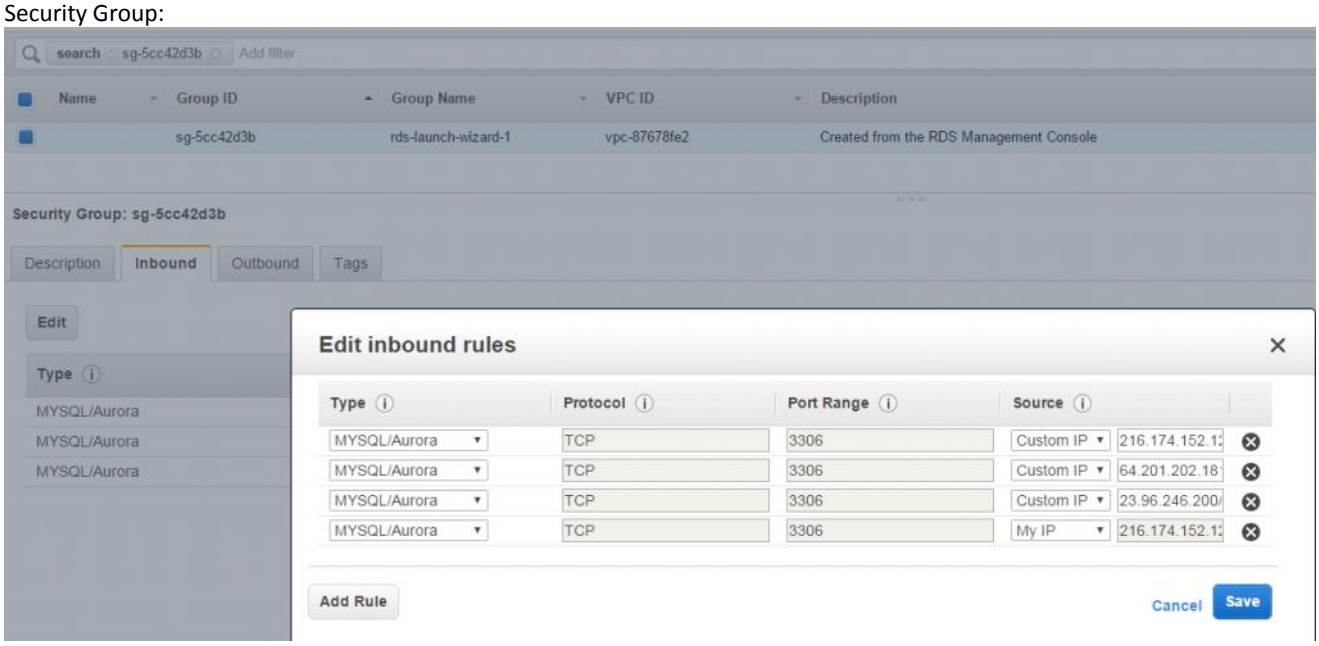


To connect to your database, you will want to use the Endpoint address. In the above example, the endpoint would be "app401-db.ctnsn4lhnv7t.us-west-2.rds.amazonaws.com".

Instance Security

AWS handles access to resources with Security Groups. In the DB Instance dashboard, click on the Security Group link for the instance. In this case, "rds-launch-wizard-1 (sg-5cc42d3b)". This will take you to the Security Group management. Add your IP as an Inbound rule for the

Security Group:



Connecting

Now that all of this is set up, you can connect to the Aurora instance from any management studio that supports MySQL. Simply enter the endpoint, master username, and master password to the connection manager in your tool, and you will be connected to the database. Best practices recommends creating a separate application account in the database, as it makes granular security easier to manage inside the database.

.NET and MySQL

The code in this example uses an ASP.NET web application and the MySQL Connector to connect to Aurora. The MySQL Connector can be obtained by downloading the MySQL Installer from http://dev.mysql.com/downloads/installer/. You will also need to add the DLL's to the ASP.NET project to make use of it in the code:

- MySql.Data
- MySql.Data.Entity.EF6
- MySql.Web

The connection string will need to be added to the web.config file as well. Once this is done, you will be able to connect to the Aurora database with whichever method you prefer.