

Configure and Connect to Serverless Aurora - MySQL Database using Cloud9 myself

A guide to building a connection with Cloud9 with a
serverless RDS- Aurora MySQL Database
in the Cloud tendencies.

[Contents]

[click on it](#)

Author: Lennox Thompson

Audience: K-12 & Adult Learners

Amazon Web Services (AWS)
Training & Certification Team

13 December 2018

”Serverless Aurora MySQL cluster”

In this tutorial, you will learn how to configure and connect to Amazon Aurora Serverless. Amazon Aurora is a relational database service with MySQL and PostgreSQL-compatible editions, which offers the performance and availability of enterprise databases at a fraction of the cost. Aurora Serverless is a new on-demand auto-scaling configuration for Aurora which is now generally available for the Aurora MySQL-compatible edition. With Aurora Serverless, your database will automatically start up, shut down, and scale up or down capacity based on your application’s needs – so you’re never paying for what you don’t use, while still benefiting from Aurora’s high availability, scale, and speed.

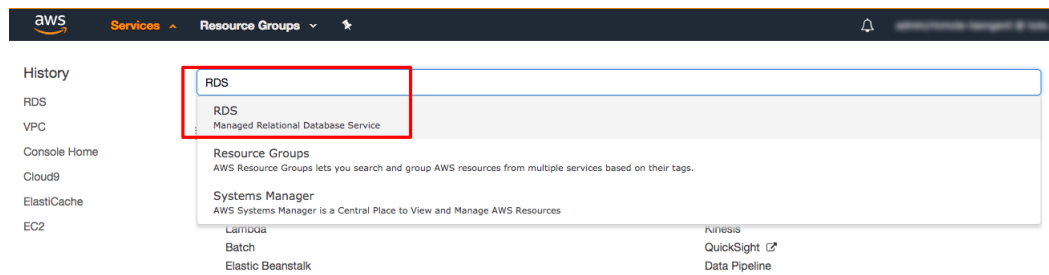
Contents

1	Navigate to the RDS console	3
2	Create an Aurora Serverless DB cluster	4
2.1	In this step, you will use Amazon RDS to create an Aurora Serverless DB cluster.	4
3	Create a Client Environment with Cloud9	9
3.1	database client	9
4	Enable client network access to your Serverless Cluster	13
4.1	Enable network access from your Cloud9 environment to your Serverless DB Cluster.	13
5	Connect to your Aurora Serverless DB Cluster	16
5.1	In this step, from your Cloud9 environment you will access your Aurora Serverless DB cluster.	16

1 Navigate to the RDS console

In this step, you will navigate to the Relational Database Service (RDS) console so you can create an Aurora Serverless DB cluster.

- Open the AWS Management Console, so you can keep this step-by-step guide open. When the screen loads, enter your user name and password to get started. Then begin typing RDS in the search bar and select RDS to open the service console.

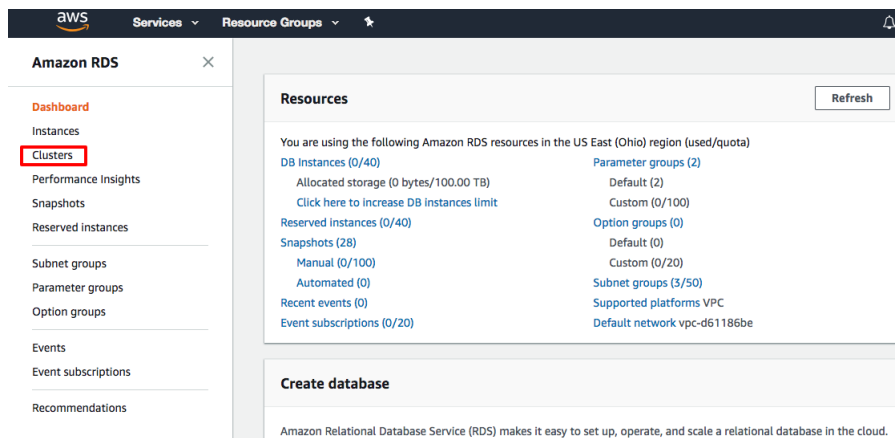


AWS Console: Services - RDS

2 Create an Aurora Serverless DB cluster

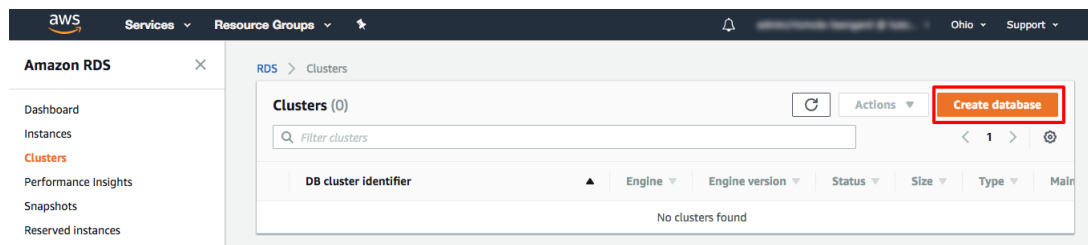
2.1 In this step, you will use Amazon RDS to create an Aurora Serverless DB cluster.

- a. On the Amazon RDS screen, from the navigation bar on the left, select Clusters.



Amazon RDS: Clusters

- b. From the RDS > Clusters screen, select Create database.





Amazon RDS: Clusters


- c. On the Select engine screen, select Amazon Aurora. In the Edition radiobuttons, select MySQL 5.6-compatible. Currently, only the MySQL 5.6 version is available with Aurora Serverless.


Select engine


Engine options


☒ Amazon Aurora



☐ Preview - Parallel Query


☐ MySQL


☐ MariaDB


☐ PostgreSQL


☐ Oracle


☐ Microsoft SQL Server


Amazon Aurora
 Amazon Aurora is a MySQL- and PostgreSQL-compatible enterprise-class database, starting at <\$1/day.

- Up to 5 times the throughput of MySQL and 3 times the throughput of PostgreSQL
- Up to 64TiB of auto-scaling SSD storage
- 6-way replication across three Availability Zones
- Up to 15 Read Replicas with sub-10ms replica lag
- Automatic monitoring and failover in less than 30 seconds

Edition

☒ MySQL 5.6-compatible
Aurora Serverless capacity is only available with this edition.

☐ MySQL 5.7-compatible

☐ PostgreSQL-compatible

☐ Only enable options eligible for RDS Free Usage Tier [Info](#)

Cancel Next

d. On the Specify DB details screen, under Capacity type select the Serverless radio button.

In the Setting pane, in the DB cluster identifier field enter MyClusterName.

Set the Master username and Master password fields with values of your choice and store the username and password for further use later.

Specify DB details

Configuration

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#).

DB engine
Aurora - compatible with MySQL 5.6.10a

Capacity type [Info](#)

☐ Provisioned
You provision and manage the server instance sizes.

☒ **Serverless** [Info](#)
You specify the minimum and maximum of resources for a DB cluster. Aurora scales the capacity based on database load (currently available for Aurora MySQL 5.6).

Settings

DB cluster identifier
Type a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.

MyClusterName

The DB cluster identifier is a case-sensitive, but is stored as all lowercase(as in "mydbcluster"). Constraints: 1 to 60 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Master username [Info](#)
Specify an alphanumeric string that defines the login ID for the master user.

Master Username must start with a letter. Must contain 1 to 16 alphanumeric characters.

Master password [Info](#) Confirm password [Info](#)

Master Password must be at least eight characters long, as in "mypassword". Can be any printable ASCII character except `"/", "\\", "\'", "\"", or "@".`

[Cancel](#)
[Previous](#)
[Next](#)

e. On the Configure advanced settings page, in the Capacity setting pane, you can change the Minimum Aurora capacity unit and the Maximum Aurora capacity unit settings.

Each Aurora capacity unit is equivalent to a specific compute and memory configuration. Aurora Serverless will automatically scale between the minimum and maximum capacity settings based on your cluster's CPU utilization, connections, and available memory.

Expand the Additional scaling configuration section. You can disable cluster pausing by unchecking the Pause compute capacity after consecutive minutes of inactivity checkbox. Using the hours, minutes, and seconds drop down list boxes, you can change the length of inactivity time until the cluster pauses. By default your cluster will pause after 5 consecutive minutes of inactivity.

For this tutorial, you should leave these default values.

Capacity settings

Billing estimate is based on published prices. [Learn more](#)

Minimum Aurora capacity unit [Info](#)

2
4Gb RAM

Maximum Aurora capacity unit [Info](#)

64
122Gb RAM

▼ Additional scaling configuration

☒ Pause compute capacity after consecutive minutes of inactivity [Info](#)

You are only charged for database storage while the compute capacity is paused

00

 hours

05

 minutes

00

 seconds

Max: 24 hours

f. On the Network Security pane, in the Virtual Private Cloud (VPC) list, select Create new VPC.

In the Subnet group list, select Create new DB Subnet Group.

In VPC security groups list, select Create new VPC security group. You will modify this new security group to allow network traffic from your database client to access your new Aurora Serverless cluster in a later step.

Select Create database.

Network & Security

Virtual Private Cloud (VPC) [Info](#)

VPC defines the virtual networking environment for this DB instance.

Create new VPC

↕

Only VPCs with a corresponding DB subnet group are listed.

Subnet group [Info](#)

DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

Create new DB Subnet Group

↕

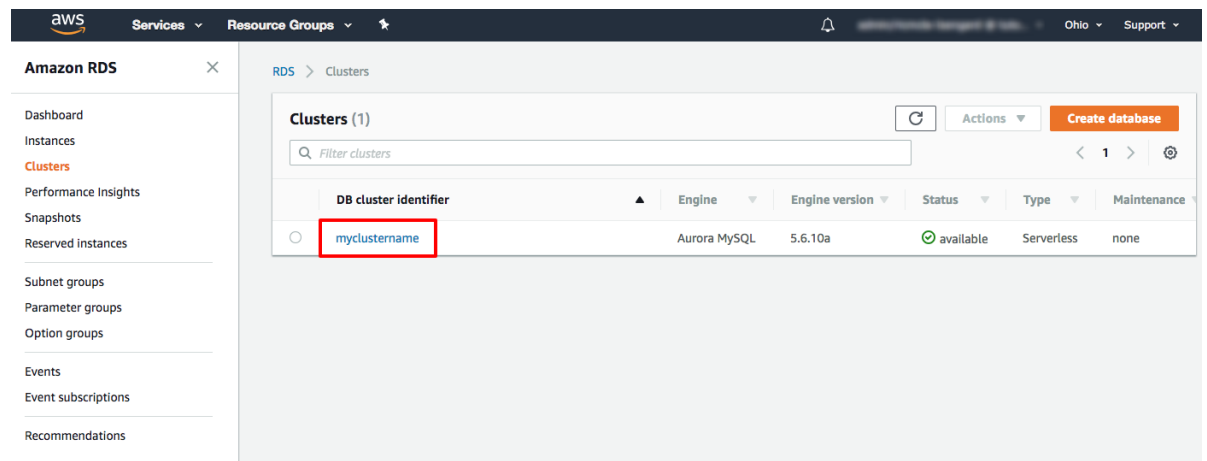
VPC security groups

Security groups have rules authorizing connections from all the EC2 instances and devices that need to access the DB instance.

☒ Create new VPC security group

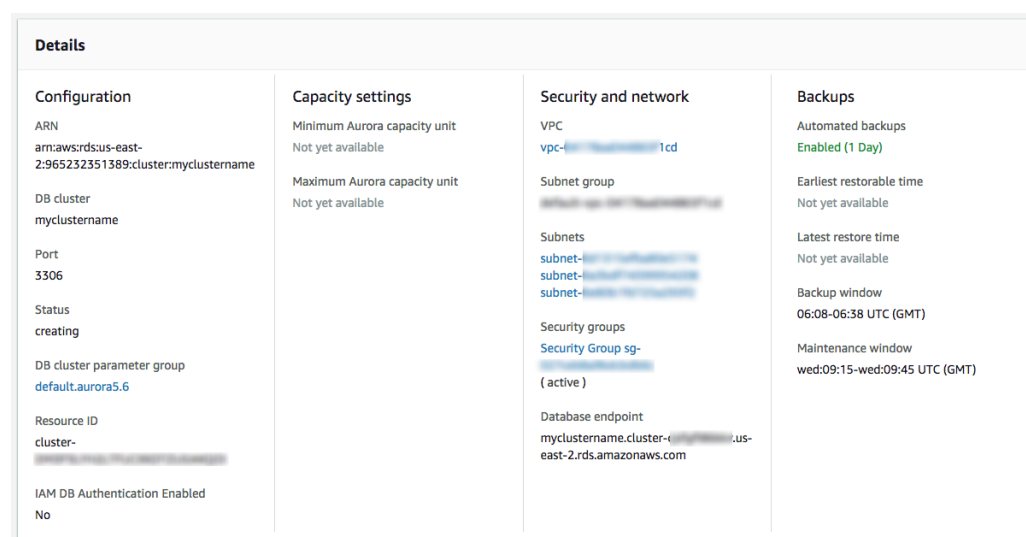
☐ Choose existing VPC security groups

g. The RDS > Clusters screen will load and your MyClusterName cluster will appear as in the creating status. Click on MyClusterName in the cluster list to access detailed information about your cluster.



h. The MyClusterName detail screen will load. This screen contains monitoring information including the Serverless Database Capacity graph that shows the number of Aurora Capacity Units in use over time and Recent Events pane which details scaling and pausing/resume events.

Scroll to the to the Details pane. Record the VPC value and Database end-point values for use later in the tutorial.

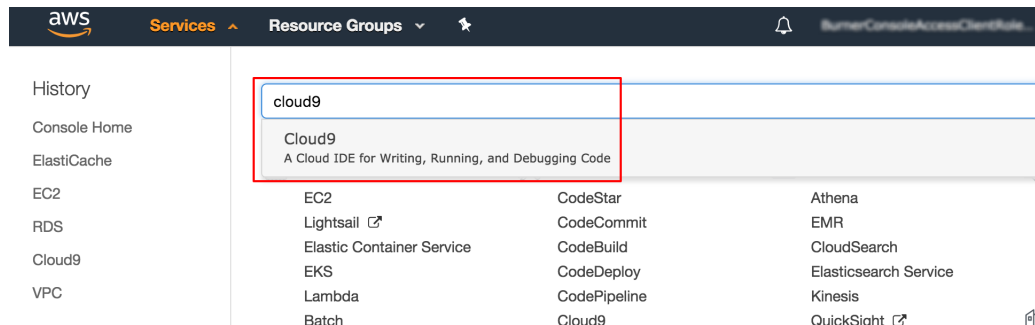


3 Create a Client Environment with Cloud9

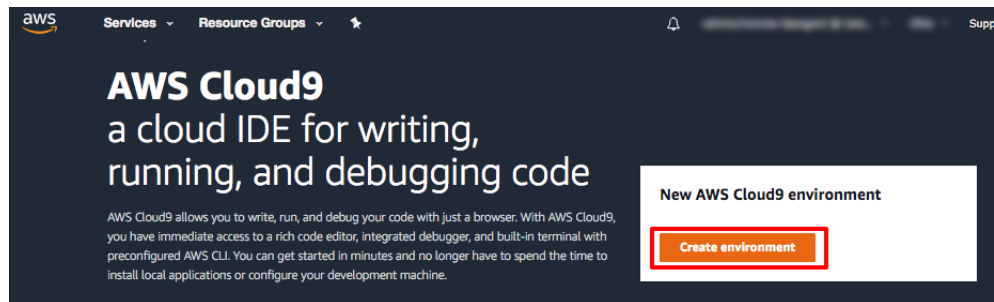
3.1 database client

After creating the MyClusterName cluster, your next task is to create a database client inside the same VPC. To complete this task you will create a Cloud9 environment to use as your database client.

- a. From the top AWS Web Console menu, select Services. In the search bar, begin typing Cloud9 and select Cloud9 to open the service console.



- b. On the AWS Cloud9 screen, select Create environment.



- c. On the Name environment screen, in the Name field type MyCloud9Env and select Next step.

ricmcla@amazon.com - 965232351388 / admin (Not Production Account)

aws Services Resource Groups

AWS Cloud9 > Environments > Create environment

Step 1
Name environment

Step 2
Configure settings

Step 3
Review

Name environment

Environment name and description

Name
The name needs to be unique per user. You can update it at any time in your environment settings.

MyCloud9Env

Limit: 60 characters

Description - Optional
This will appear on your environment's card in your dashboard. You can update it at any time in your environment settings.

Write a short description for your environment

Limit: 200 characters

Cancel Next step

d. On the Configure setting screen, leave the environment type as Create a new instance for environment (EC2) and the Instance type as t2.micro.

Configure settings

Environment settings

Environment type [Info](#)
Choose between creating a new EC2 instance for your new environment or connecting directly to your server over SSH.

☒ **Create a new instance for environment (EC2)**
Launch a new instance in this region to run your new environment.

☐ Connect and run in remote server (SSH)
Display instructions to connect remotely over SSH and run your new environment.

Instance type

☒ **t2.micro (1 GiB RAM + 1 vCPU)**
Free-tier eligible. Ideal for educational users and exploration.

☐ t2.small (2 GiB RAM + 1 vCPU)
Recommended for small-sized web projects.

☐ m4.large (8 GiB RAM + 2 vCPU)
Recommended for production and general-purpose development.

☐ Other instance type
Select an instance type.

t2.nano ▼

Cost-saving setting
Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation settings of half an hour of no activity to maximize savings.

After 30 minutes (default) ▼

IAM role
AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)

AWSServiceRoleForAWSCloud9

e. Because Aurora Serverless DB clusters do not have publically accessible endpoints, your MyClusterName can only be accessed from within the same VPC.

To place MyCloud9Env in the same VPC as MyClusterName, scroll down the Configure setting screen and expand the Network settings (advanced) section. From the Network (VPC) drop down, select MyClusterName's VPC which you recorded step 2h.

Select Next step.

On the Review page, select Create environment.

After your new Cloud9 environment has been created, proceed to the next step.

IAM role

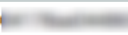
AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)

AWSServiceRoleForAWSCloud9

▼ Network settings (advanced)

Network (VPC)

Launch your EC2 instance into an existing Amazon Private Cloud (VPC) or create a new one.


vpc-1cd



[Create new VPC](#)

Subnet

Select a range of IP addresses in your VPC to isolate EC2 resources from each other.

subnet- | Non-default in us-east-2a

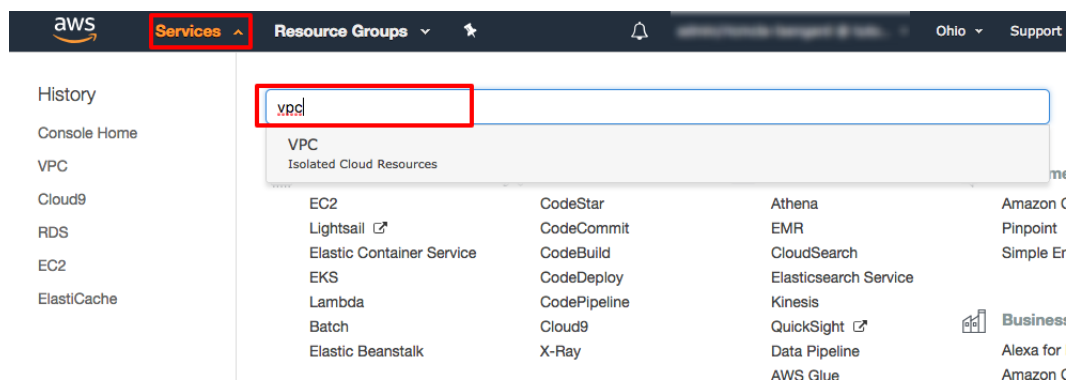


[Create new subnet](#)

4 Enable client network access to your Serverless Cluster

4.1 Enable network access from your Cloud9 environment to your Serverless DB Cluster.

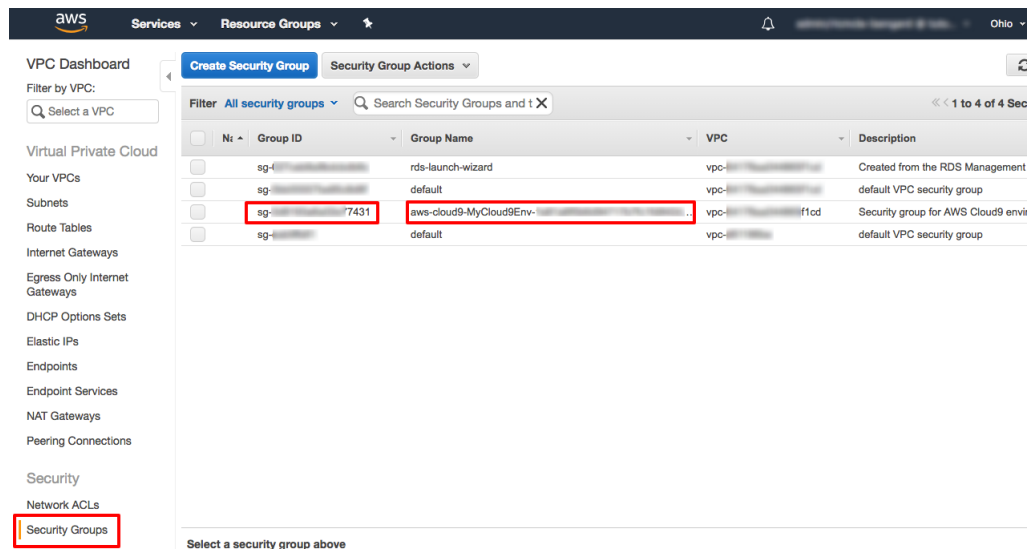
a. To make navigation easier, load the AWS Web Console in another browser tab by clicking [here](#). In the new browser window, on the top menu bar, select Services then type VPC in the search bar and select VPC from the list.



b. On the VPC Dashboard page, in the left navigation select Security groups.

In the Group Name column, find the security group that begins with aws-cloud9-MyCloud9Env. Note the Group ID of this security group.

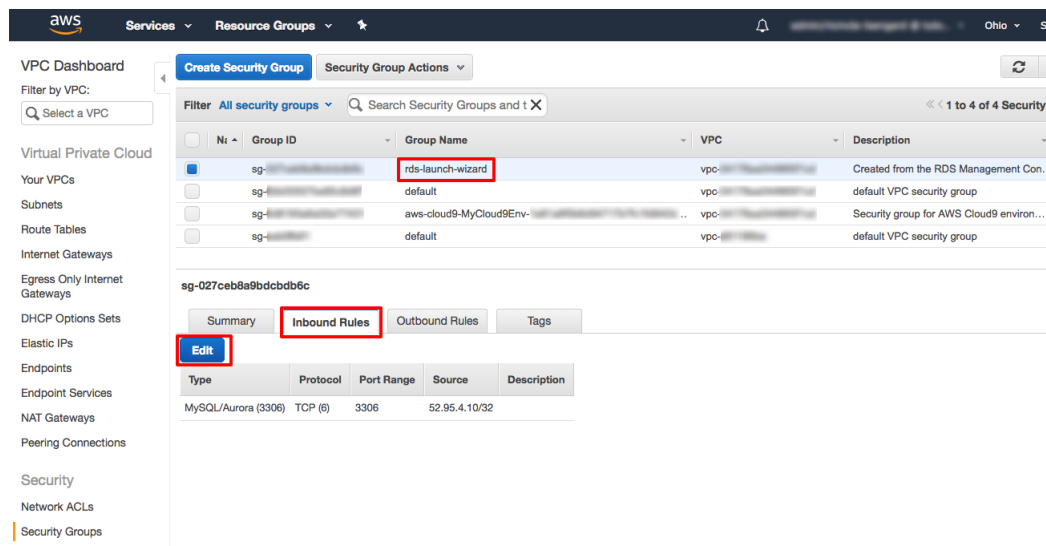
As an example, in the screen shot to the right, you would note the security group that ends with 7431. Your Group ID will be different than is pictured in this screen shot.



c. In the list of security groups, select the security group that begins with RDS-Launch-Wizard.

Then select the **Inbound Rules** tab.

Then select **Edit**.



d. In the Inbound Rules tab, select Add another rule.

In the Type column, select MySQL/Aurora (3306) from the drop down list.

Then click into the Source column field and a drop down list will appear. Select the security Group ID that you noted in step 4b. Then select Save.

As an example, in the screen shot to the right, from the list you would select the security group that ends 7431. Your specific Group ID will be different than is pictured in this screen shot.

The screenshot shows the AWS VPC console interface. On the left is a navigation sidebar with categories like Virtual Private Cloud, Security, and Network ACLs. The main area displays the 'Security Group Actions' for 'sg-027ceb8a9bdcdb6c'. The 'Inbound Rules' tab is active, showing a table of rules. The second rule is highlighted, with its 'Source' dropdown menu open. The dropdown list shows several security groups, and the one with ID '7431' is selected. The 'Save' button is highlighted with a red box.

Type	Protocol	Port Range	Source	Description	Remove
MySQL/Aurora (3306)	TCP (6)	3306	sg-027ceb8a9bdcdb6c		
MySQL/Aurora (3306)	TCP (6)	3306	sg-027ceb8a9bdcdb6c		

5 Connect to your Aurora Serverless DB Cluster

5.1 In this step, from your Cloud9 environment you will access your Aurora Serverless DB cluster.

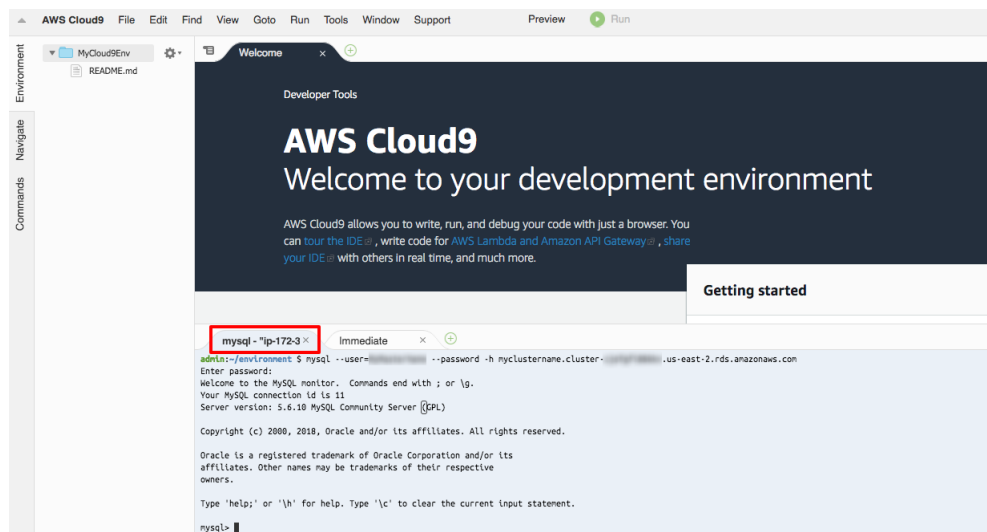
- a. Switch back to your MyCloud9Env browser window.

In the bash terminal tab in MyCloud9Env, type in the following command. Substitute your Master username and database endpoint for the values in the command and press Enter.

```
mysql --user=[your Master username] --password -h [your database endpoint]
```

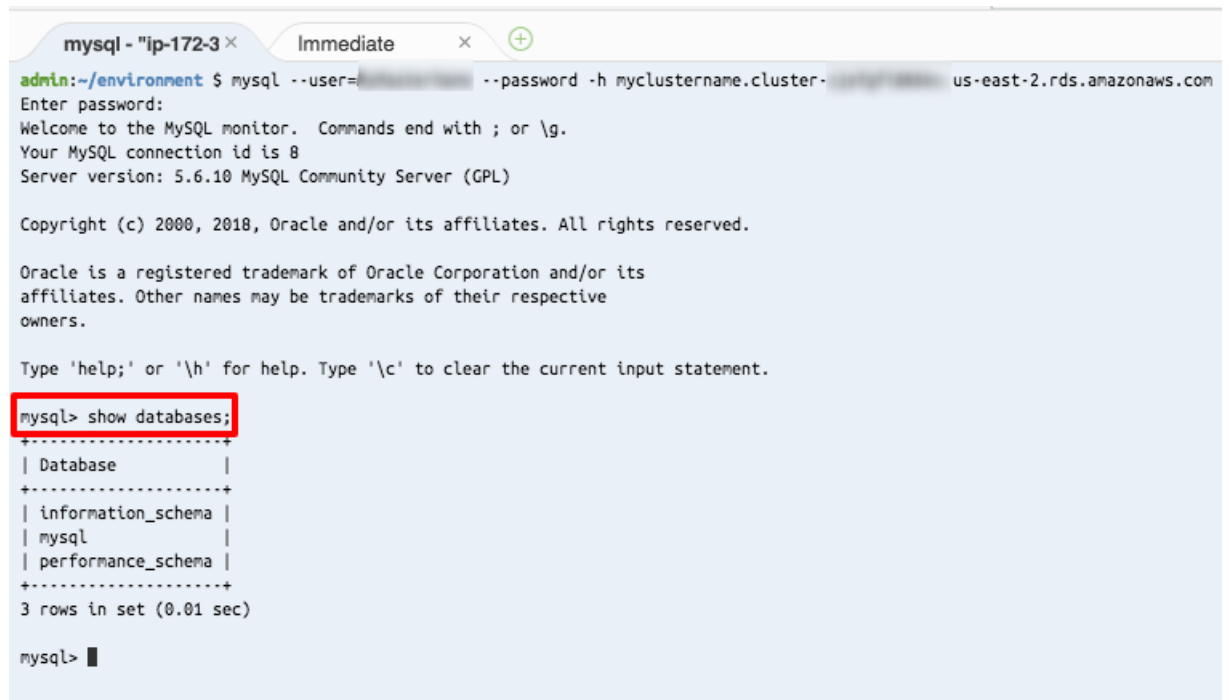
When prompted, enter your Master password and press Enter.

You should now be connected to the MyClusterName Aurora Serverless DB cluster!



- b. You can issues commands to the Aurora Serverless DB cluster using the connection you have established. For example, you can show the databases on the server by pasting the following command into the MyCloud9Env bash tab:

```
show databases;
```


A terminal window titled 'mysql - "ip-172-3 x' with a sub-tab 'Immediate'. The prompt is 'admin:~/environment \$'. The command executed is 'mysql --user=[redacted] --password [redacted] -h myclustername.cluster-[redacted].us-east-2.rds.amazonaws.com'. The output shows the MySQL monitor welcome message, connection ID 8, and server version 5.6.10. The user enters 'show databases;' which is highlighted with a red box. The output shows a table with 3 rows: information_schema, mysql, and performance_schema.

```
admin:~/environment $ mysql --user=[redacted] --password [redacted] -h myclustername.cluster-[redacted].us-east-2.rds.amazonaws.com
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.6.10 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database                |
+-----+
| information_schema       |
| mysql                   |
| performance_schema       |
+-----+
3 rows in set (0.01 sec)

mysql> █
```

Congratulations!

You have created, connected to an Aurora Serverless DB cluster. To experience the real benefits of Aurora Serverless, connect it to your applications with variable or infrequently use, dev/test environments, multi-tenant applications and other apps that can benefit from on-demand auto-scaling.