Aurora MySQL Hands-on Lab Manual – Version 2.1

This hands-on lab manual will cover following tasks and procedure:

- ✓ **Create Aurora MySQL DB** Cluster
- ✓ M**odify security group** to allow access to the Aurora MySQL DB instance from your computer
- ✓ **Load data from S3** into Aurora MySQL database
- ✓ **Create read replica** instance & access table
- ✓ **Database copy** using Aurora "**Clone**" feature
- ✓ Perform DML query on primary DB and validate data on primary and cloned DB copy

Note:

AWS Region: Use AWS region **(us-west-2)** to perform all activities in this Lab.

Create an Amazon Aurora MySQL Database Cluster using db.t2.small instance type

Create an IAM policy to grant access to the lab S3 bucket resources

1. Open the IAM Management Console.
2. In the navigation pane, choose **Policies**.
3. Choose **Create policy**.
4. In the create policy screen, select JSON tab.
5. Copy and paste the following replacing the existing contents.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket",
            "s3:GetBucketPolicy",
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::aurora-lab-loft/*"
         ]
      }
   ]
}
```

6. Choose **Review policy**.
7. In the Review Policy Screen, Set the name of the IAM policy to *aurora-lab-s3-policy*
8. Choose **Create policy**.

Create an IAM role to allow Amazon Aurora to access AWS S3

1. Open the IAM console.
2. In the navigation pane, choose **Roles**.
3. Choose **Create role**.
4. Under **AWS service**, choose **RDS**.
5. Under **Select your use case**, choose **RDS – CloudHSM and Directory Service**.
6. Choose **Next: Permissions**.
7. Choose **Next: Review**.
8. Set **Role Name** to *aurora-lab-s3-role*
9. Choose **Create Role**.
10. In the navigation pane, choose **Roles**.
11. In the **Search** field, enter *aurora-lab-s3-role*, and click the role when it appears in the list.
12. On the **Permissions** tab, detach the following default roles from the policy:

```
AmazonRDSDirectoryServiceAccess

RDSCloudHsmAuthorizationRole
```

    To detach a role, click the **X** associated with the role on the right, and then click **Detach**.

13. On the **Permissions** tab, choose **Attach policy**.
14. On the **Attach policy** page, enter *aurora-lab-s3-policy* in the **Search** field.
15. When it appears in the list, select the policy
16. Choose **Attach policy**.
17. Click *aurora-lab-s3-role* again.
18. Copy the Role ARN and save it to your notepad. You are going to need this at a later step.

Create Cluster parameter group

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose **Parameter groups**.
4. Choose **Create parameter group**.
   The **Create parameter group** window appears.
5. Select the following values.

Parameter group family	aurora-mysql5.7
Type	DB Cluster Parameter Group
Group name	aurora-lab-cluster-pg
Description	aurora-lab-cluster-pg

6. Choose **Create**.

Modify Cluster parameter group

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose **Parameter groups**.
4. In the list, select *aurora-lab-cluster-pg* parameter group.
5. Choose **Parameter group actions**, and then choose **Edit**.
6. Enter *aurora_load_from_s3_role* in the "Filter parameters" field
7. Paste the IAM role ARN value that you had copied in the previous step in the Values field for the parameter.
8. Choose **Save changes**.

Create Amazon Aurora MySQL database cluster

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose **Instances**.
4. Choose **Create database** to start the Create database wizard. The wizard opens on the **Select engine** page.
5. On the **Select Engine** page, choose Amazon Aurora under Engine Options and then choose MySQL 5.7-compatible under edition.
6. Choose Next
7. Choose the following DB cluster specifications:

Instance Size:	t2.small
Multi-AZ deployment:	No
DB Instance Identifier:	aurora-lab-instance
Master username:	rdsmaster
Master password:	Welcome123

8. Choose Next
9. Choose the following DB cluster specifications in Configure advanced settings screen.

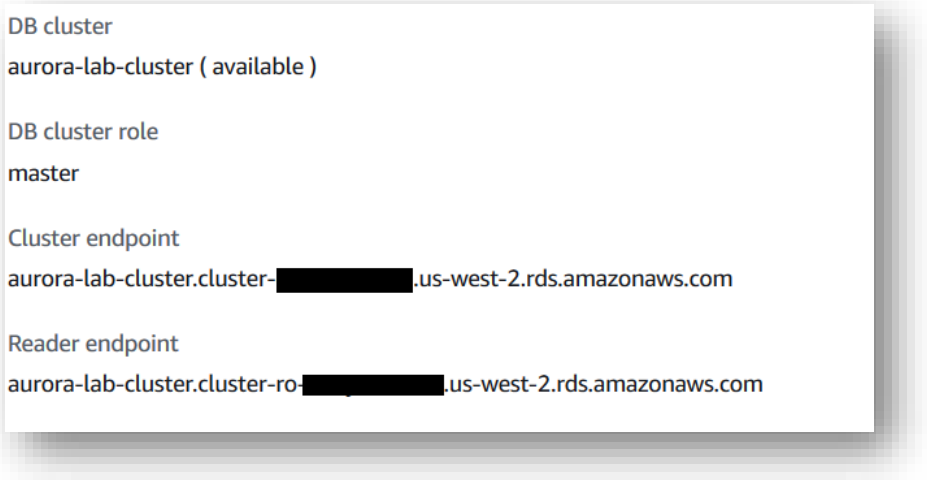
VPC:	Use the default VPC
Subnet group:	default
Public Accessibility:	Yes (should be OK for this lab)
Availability Zone:	No preference
VPC Security Group:	Create new VPC security group
DB Cluster Identifier:	aurora-lab-cluster
Database Name:	auroralabdb
Database Port:	3306
DB parameter group:	default.aurora-mysql5.7
DB cluster parameter group:	*aurora-lab-cluster-pg*
Encryption:	Disable encryption
Failover Priority	No preference
Backup	1 day

Enhanced Monitoring	Disable enhanced monitoring
Log exports	Leave it at defaults
Maintenance	Leave it at defaults
Enable deletion protection	Uncheck the box

10. Choose Create database

Find the Aurora Cluster endpoints

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose Clusters
4. Click the name of your cluster which is *aurora-lab-cluster*
5. Copy/Paste the Cluster and Reader endpoints in your notepad. You will need them later.



Explainer Notes

By default, Aurora Reader endpoint will point to the writer DB instance when you create an Aurora cluster with the Multi AZ deployment option disabled. When you add a read replica later, the reader endpoint will automatically use read replica for all read only queries.

Grant database access from your computer

Validating the DB connection

Verify you can access Aurora MySQL cluster from your computer using the master username

$ mysql -h <aurora cluster endpoint> -u rdsmaster -p

Enter password: <Enter the password you supplied during set up of the Aurora Cluster>

mysql>

To exit from the “mysql>” prompt, use CTRL-D

If the above works, you can skip the following step.

Modify VPC Security group to allow access from your computer

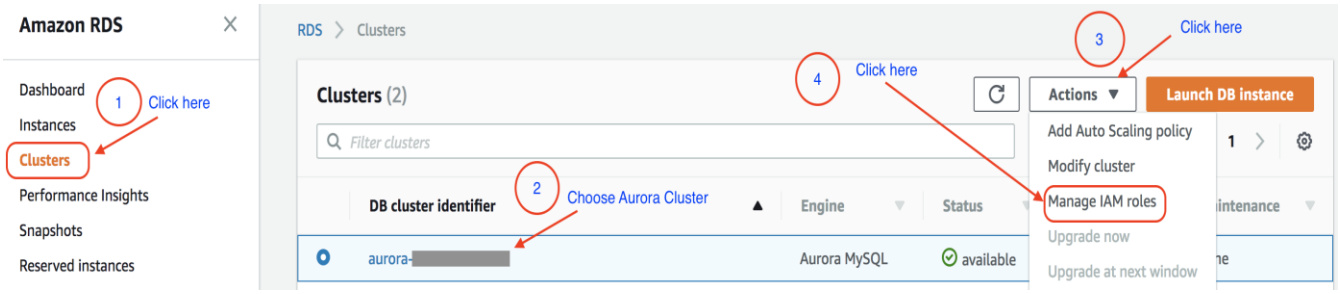
1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose Clusters
4. Click the name of your Aurora Cluster which is *aurora-lab-cluster*
5. Under “DB Cluster Members”, click on your aurora writer instance.
6. Under “Security group rules”, click on the name of the security group
7. Choose Inbound Rules.
8. Choose **Edit**.
9. Select Type=> MYSQL/Aurora, Source=> Anywhere. (This should okay for this lab)
10. Choose Save

Now revisit the previous step to verify connection to the Aurora DB cluster is working from your computer.

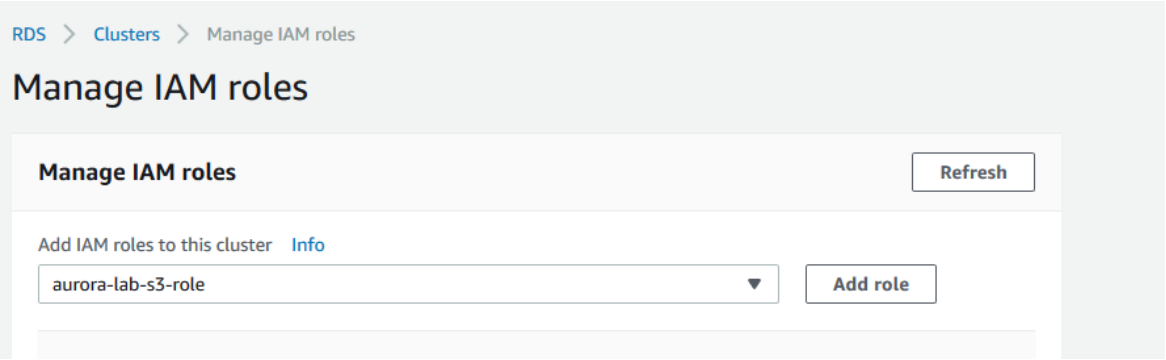
Load data from S3 into Aurora MySQL database

Associating an IAM Role with an Amazon Aurora MySQL DB Cluster

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose Clusters
4. Select your aurora cluster, which is *aurora-lab-cluster*, and click Actions drop down.
5. Choose Manage IAM roles.



6. In **Manage IAM roles**, choose *aurora-lab-s3-role* to associate with your DB cluster from **Add IAM roles to this cluster**.
7. Choose Add role.
8. Choose Done.



Data load into Aurora MySQL database

Connect to Aurora cluster end-point:

```
$ mysql -h <aurora cluster endpoint> -u rdsmaster -p
Enter password: <Enter the password you supplied during set up of the Aurora Cluster>
```

Create a landsat database:

```
mysql> CREATE DATABASE landsat;
mysql> USE landsat;
```

Create the scene_list table:

```
CREATE TABLE `scene_list` (
  `entityId` varchar(64) DEFAULT NULL,
  `acquisitionDate` datetime DEFAULT NULL,
  `cloudCover` decimal(5,2) DEFAULT NULL,
  `processingLevel` varchar(8) DEFAULT NULL,
  `path` int(11) DEFAULT NULL,
  `row` int(11) DEFAULT NULL,
  `min_lat` decimal(8,5) DEFAULT NULL,
  `min_lon` decimal(8,5) DEFAULT NULL,
  `max_lat` decimal(8,5) DEFAULT NULL,
  `max_lon` decimal(8,5) DEFAULT NULL,
  `download_url` varchar(128) DEFAULT NULL);
```

Validate IAM S3 role settings parameter (output should look similar to the following)

```
mysql> show variables like 'aurora_load_from_s3_role';
+---------------------------+-----------------------------------------------------+
| Variable_name             | Value                                               |
+---------------------------+-----------------------------------------------------+
| aurora_load_from_s3_role | arn:aws:iam::<AWS Account #>:role/aurora-lab-s3-role |
+---------------------------+-----------------------------------------------------+
1 row in set (0.01 sec)
```

Load landsat data into scene_list table:

```
mysql> LOAD DATA FROM S3 's3://aurora-lab-loft/scene_list' INTO TABLE
scene_list FIELDS TERMINATED BY ',';
```

Above command takes about a minute to load

Run SQL query against scene_list table:

```
mysql> select count(*) from scene_list;

mysql> select * from scene_list limit 5;
```

Create a read replica

Create a new Read replica called "*aurora-lab-instance-**rr***" within the same region. Use same security group as used in your writer Aurora DB instance.

To create new read replica:

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose **Instances**.
4. Choose *aurora-lab-instance* MySQL DB instance and choose **Create Aurora read replica** from **Instance actions**.
5. Choose the following DB cluster specifications:

Availability Zone	No preference
Publicly accessible	Yes
Encryption	Disable encryption
DB instance class	db.t2.small
Aurora replica source	leave it at default
DB instance identifier	aurora-lab-instance-**rr**
Failover	No preference
DB parameter group	default.aurora-mysql5.7
Monitoring	Disable enhanced monitoring
Maintenance	Yes

6. Choose Create Aurora replica

Wait for the Status of the replica to be available.

Validating read replica

Use Aurora Reader endpoint to run a SELECT query:

$ mysql –h <Aurora reader endpoint> –u rdsmaster -p

mysql> use landsat;

```
mysql> select count(*) from scene_list;
```

Create a database copy by using Clone feature

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the top-right corner of the AWS Management Console, select US West (Oregon) region.
3. In the navigation pane, choose **Instances**.
4. Choose *aurora-lab-instance* MySQL DB instance and choose **Create clone** from **Instance actions**.
5. On the **Create Clone** page, type ***aurora-lab-instance-clone*** for the primary instance of the clone DB cluster as the **DB instance identifier**.
6. Leave all the other options at default.
7. Choose Create Clone.
8. In the navigation pane, choose **Clusters**.
9. Click the cloned aurora DB cluster which is *aurora-lab-instance-**clone-cluster***
10. Copy/Paste the Cluster endpoints in your notepad. You will need it the next section.
11. In the navigation pane, choose **Instances**
12. Wait for *aurora-lab-instance-**clone*** instance status to be available.

Perform DML query on primary DB and validate data on primary and cloned DB copy

Connect to primary MySQL database using Aurora cluster end-point & run the following delete query.

```
$ mysql -h <aurora cluster endpoint> -u <username> -p

mysql> use landsat;

mysql> select count(*) from scene_list;
+----------+
| count(*) |
+----------+
|  1059757 |
+----------+

mysql> delete from scene_list limit 25;
Query OK, 25 rows affected (0.01 sec)

mysql> select count(*) from scene_list;
+----------+
| count(*) |
+----------+
|  1059732 |
+----------+
```

Connect to cloned MySQL database & validate number of table rows:

```
$ mysql -h <cloned aurora cluster endpoint> -u<username> -p
```

```
mysql> use landsat;

mysql> select count(*) from scene_list;
+----------+
| count(*) |
+----------+
|  1059757 |
+----------+
```