Accelerate Database Development and Testing with Amazon Aurora – Version 2.2

Lab Guide

PART 1. Setup Environment

Please log into the AWS Management Console and select AWS region as noted below.

You will be using the **Singapore (ap-southeast-1)** region.

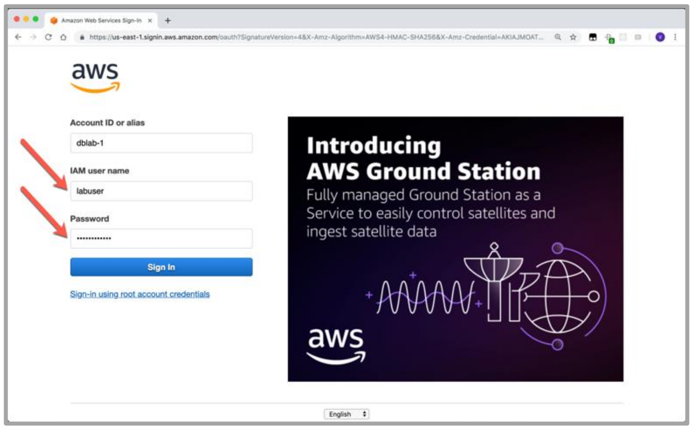
In this part of the lab you will leverage AWS CloudFormation to provision an Aurora MySQL 5.6 compatible database cluster, along with a Linux EC2 instance to be used as a workstation. You will connect to the workstation using SSH.

The environment deployed using CloudFormation includes several components, as listed below. Please download the CloudFormation template (instructions below) and review it for more details.

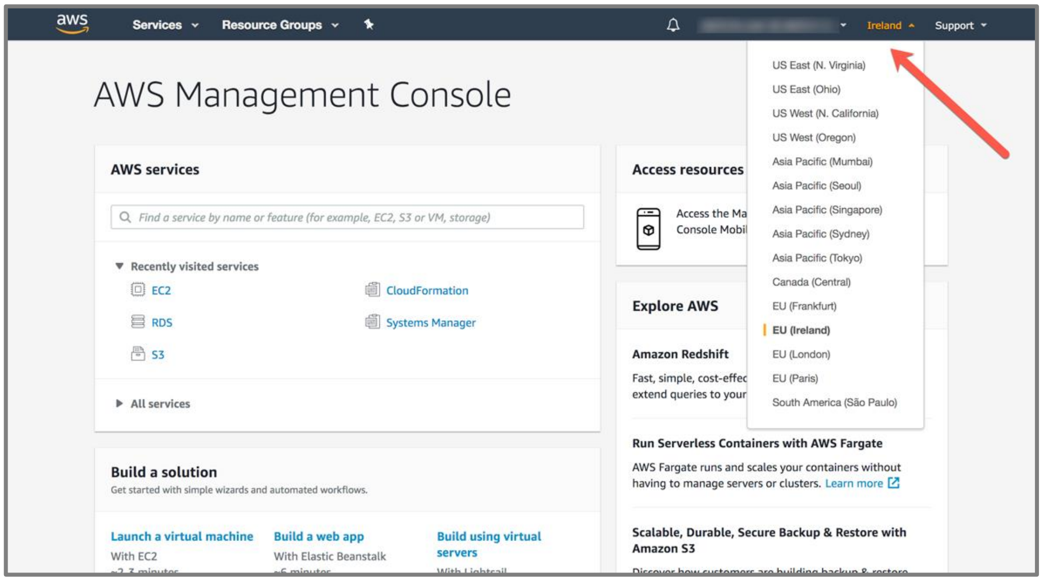
1. Amazon VPC network configuration with public and private subnets
2. Database subnet group and relevant security groups for the cluster and workstation
3. Amazon EC2 instance configured with the software components needed for the lab
4. Roles with access permissions for the workstation and cluster permissions for enhanced monitoring, S3 access and logging
5. Custom cluster and DB instance parameter groups for the Amazon Aurora cluster, enabling logging and performance schema
6. Amazon Aurora DB cluster with 2 nodes: a writer and read replica
7. Read replica auto scaling configuration
8. AWS Systems Manager command document to execute a load test

Task 1.1 – Sign in to Management Console, Select Region and Create Key Pair

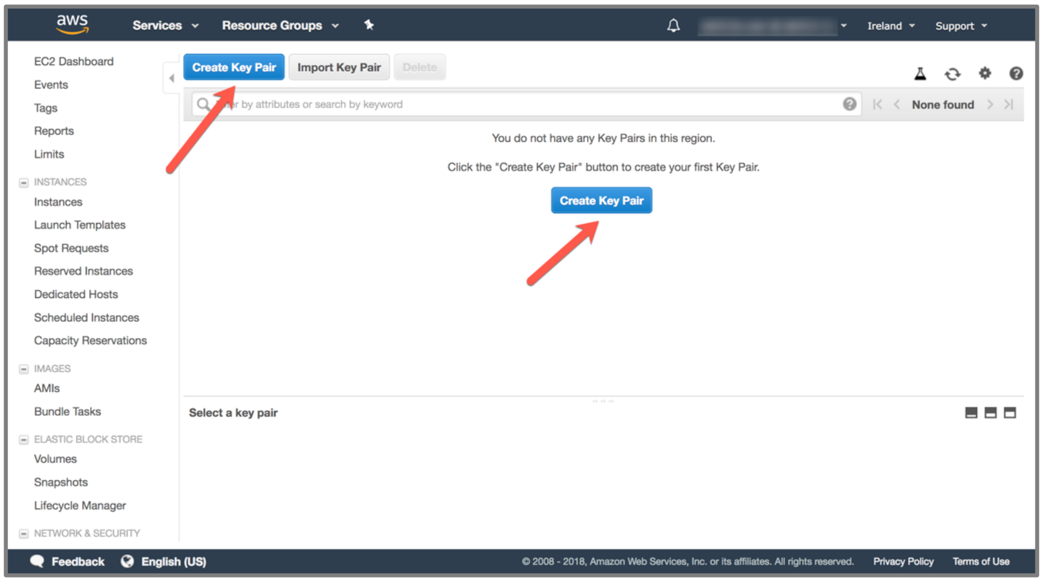
1. **Sign In** to AWS Console.


Account ID or alias
dblab-1
IAM user name
labuser
Password
Sign In
Sign-in using root account credentials
Introducing AWS Ground Station
Fully managed Ground Station as a Service to easily control satellites and ingest satellite data
English

2. Ensure the **Singapore (ap-southeast-1)** region is selected in the top right corner, if not use that dropdown to choose the correct region



3. Open the **Key Pairs** section of the EC2 service console, using this short link: https://amzn.to/2IzP387.
4. Ensure you are still in the correct region, and click **Create Key Pair**.

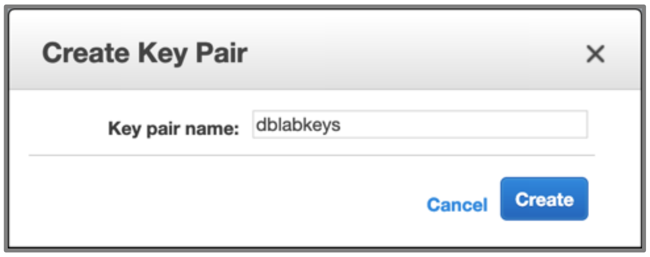


You do not have any Key Pairs in this region.

Click the "Create Key Pair" button to create your first Key Pair.

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

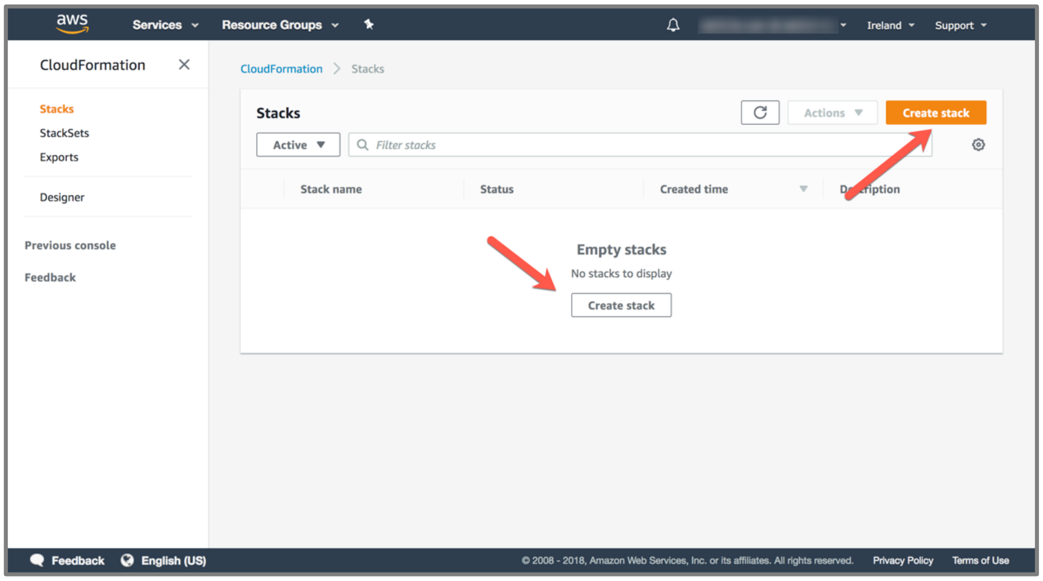
5. Name the key pair "`dblabkeys`" and then click **Create** and download the file named **dblabkeys.pem** to your computer, save it in a memorable location like your desktop. You will need this file later in the lab.



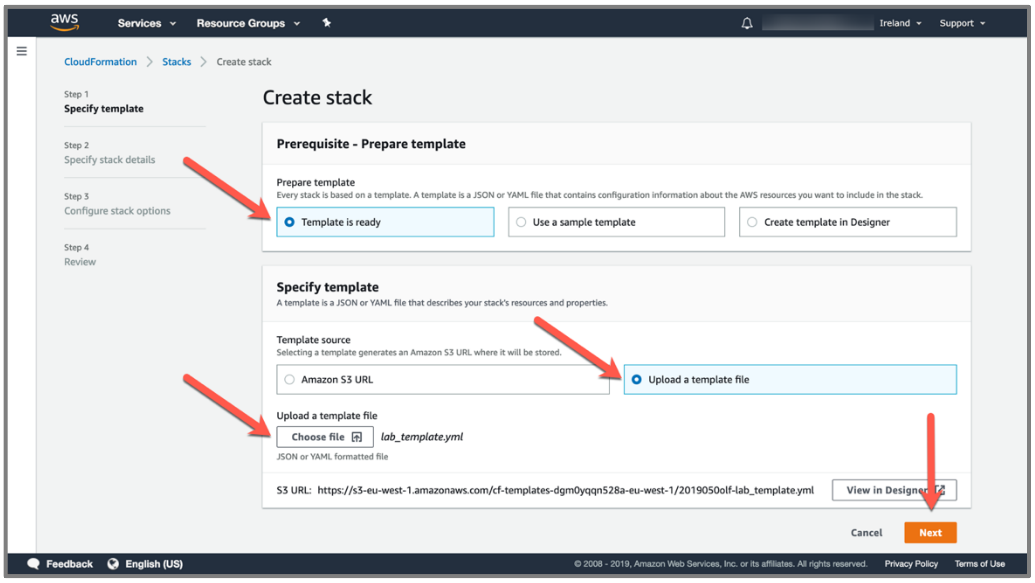
Task 1.2 - Creating a Stack using CloudFormation

1. Download the CloudFormation template named **lab_template.yml** from http://bit.ly/aurora-dblab-template. Save it in a memorable location such as your desktop, you will need to reference it.
2. Open the **CloudFormation** service console located at: https://amzn.to/2BIn3Jh.
3. Click **Create Stack.**

Notice: The CloudFormation console has been upgraded recently. Depending on your previous usage of the CloudFormation console UI, you may see the old design or the new design, you may also be presented with a prompt to toggle between them. In this lab we are using the **new design** for reference, although the steps will work similarly in the old console design as well, if you are more familiar with it.



4. Select the radio button named **Upload a template**, then **Choose file** and select the template file you downloaded previously named **lab_template.yml** and then click **Next**.


aws
Services
Resource Groups
Ireland
Support
CloudFormation > Stacks > Create stack
Step 1 Specify template
Step 2 Specify stack details
Step 3 Configure stack options
Step 4 Review
Cancel
Next
Feedback
English (US)
© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

[✓ Template is ready] Use a sample template Create template in Designer

Specify template
A template is a JSON or YAML file that describes your stack's resources and properties.

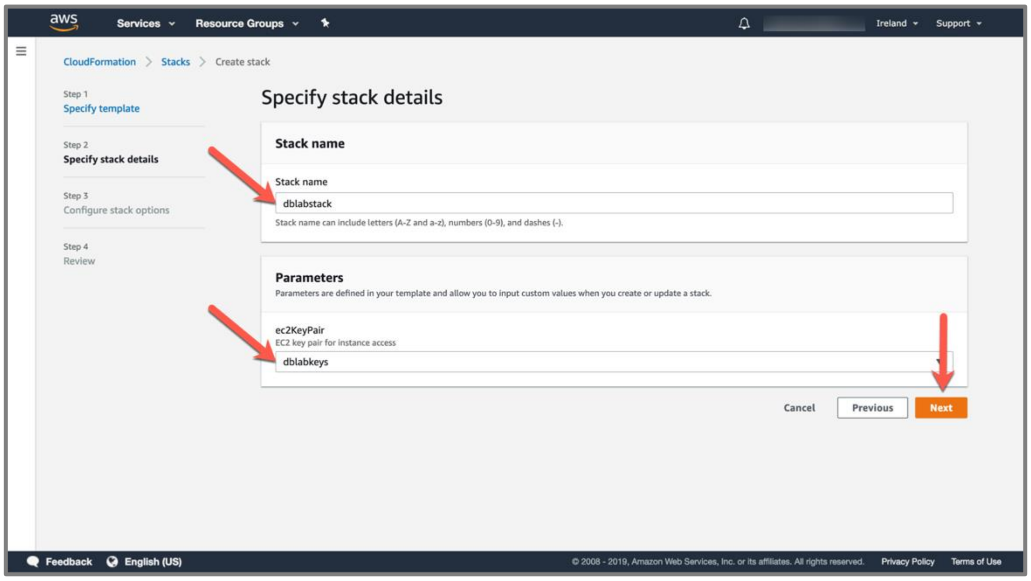
Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

Amazon S3 URL [✓ Upload a template file]

Upload a template file
Choose file *lab_template.yml*
JSON or YAML formatted file

S3 URL: https://s3-eu-west-1.amazonaws.com/cf-templates-dgm0yqqn528a-eu-west-1/2019050olf-lab_template.yml View in Designer

5. In the field named **Stack Name**, enter the value "`dblabstack`", select the **ec2KeyPair** value as "`dblabkeys`" (the key pair you have created previously) and then click **Next**.


aws
Services
Resource Groups
Ireland
Support
CloudFormation > Stacks > Create stack
Step 1 Specify template
Step 2 Specify stack details
Step 3 Configure stack options
Step 4 Review
Cancel
Previous
Next
Feedback
English (US)
© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Specify stack details

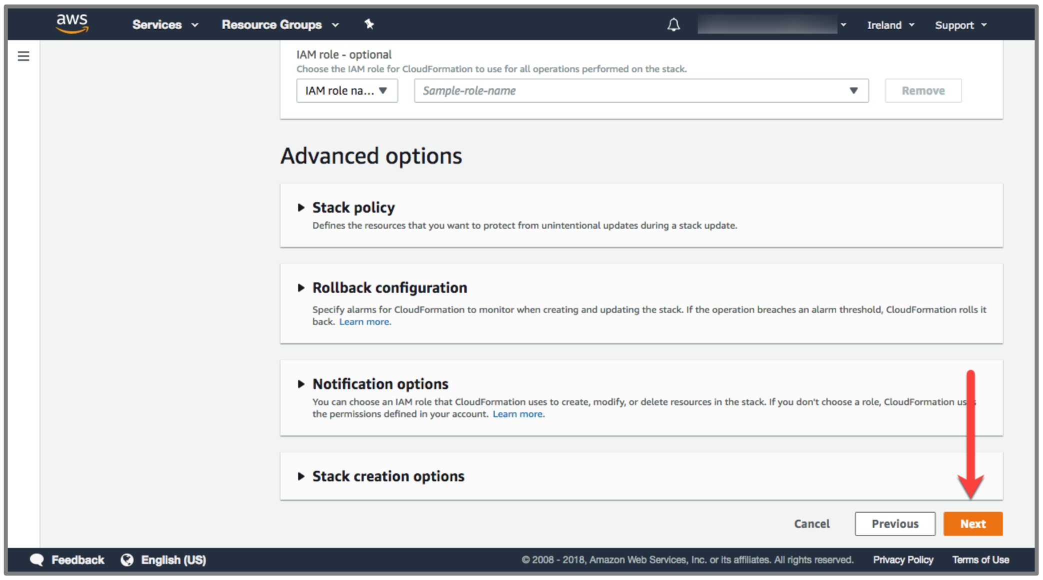
Stack name

Stack name: dblabstack
Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters
Parameters are defined in your template and allow you to input custom values when you create or update a stack.

ec2KeyPair
EC2 key pair for instance access
dblabkeys

6. On the **Configure stack options** page, leave the defaults as they are, scroll to the bottom and click **Next**.



IAM role - optional

Choose the IAM role for CloudFormation to use for all operations performed on the stack.

IAM role na... Sample-role-name Remove

Advanced options

Stack policy

Defines the resources that you want to protect from unintentional updates during a stack update.

Rollback configuration

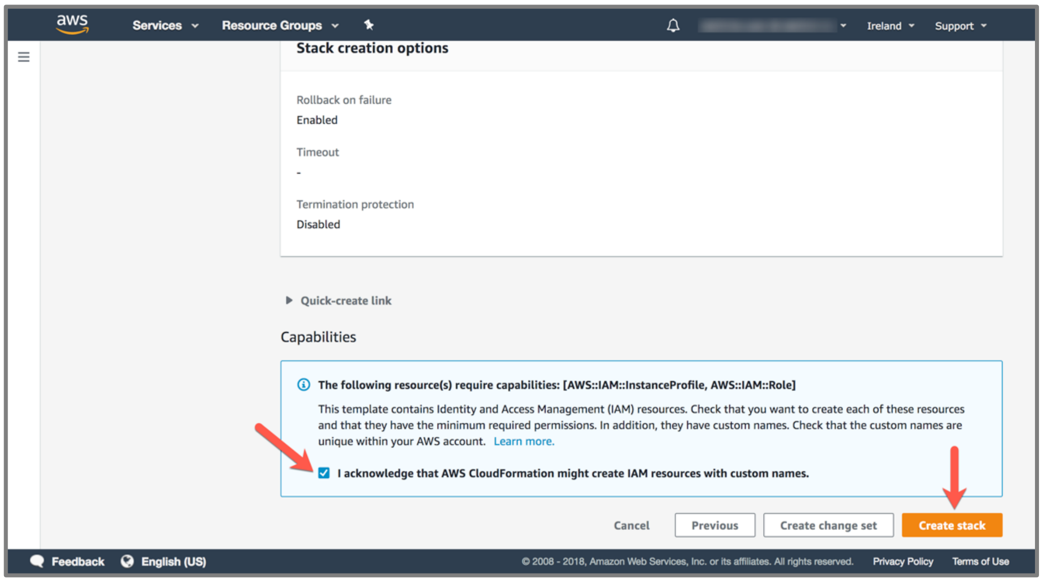
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back. Learn more.

Notification options

You can choose an IAM role that CloudFormation uses to create, modify, or delete resources in the stack. If you don't choose a role, CloudFormation uses the permissions defined in your account. Learn more.

Stack creation options

7. On the **Review dblabstack** page, scroll to the bottom, check the box that reads: **I acknowledge that AWS CloudFormation might create IAM resources with custom names** and then click **Create**.



Stack creation options

Rollback on failure
Enabled

Timeout
-

Termination protection
Disabled

Quick-create link

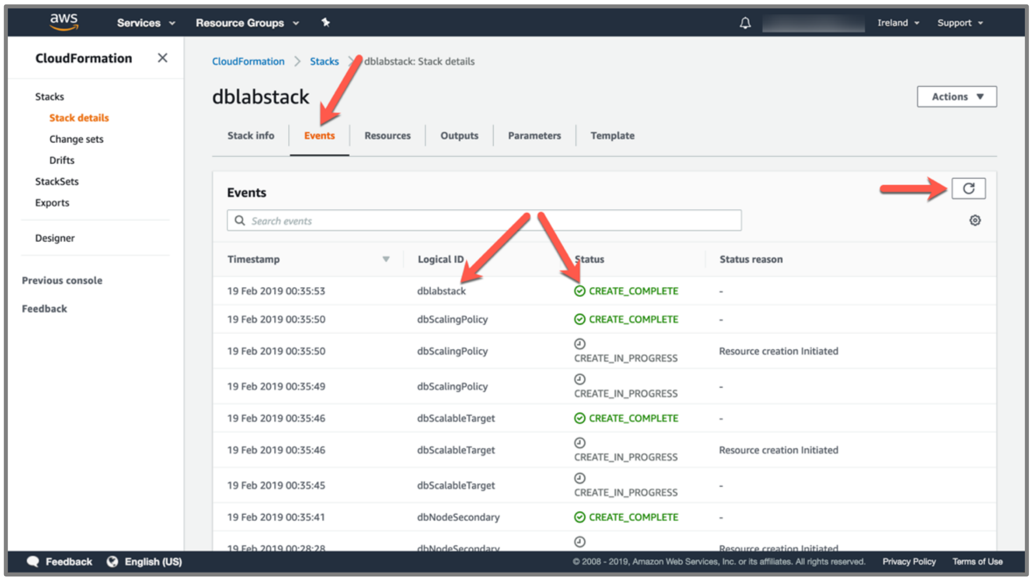
Capabilities

The following resource(s) require capabilities: [AWS::IAM::InstanceProfile, AWS::IAM::Role]

This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. Learn more.

☑ I acknowledge that AWS CloudFormation might create IAM resources with custom names.

8. The stack will take approximatively 20 minutes to provision, you can monitor the status on the **Stack detail** page. You can monitor the progress of the stack creation process by refreshing the **Events** tab. The latest event in the list will indicate CREATE_COMPLETE for the **dblabstack** resource.



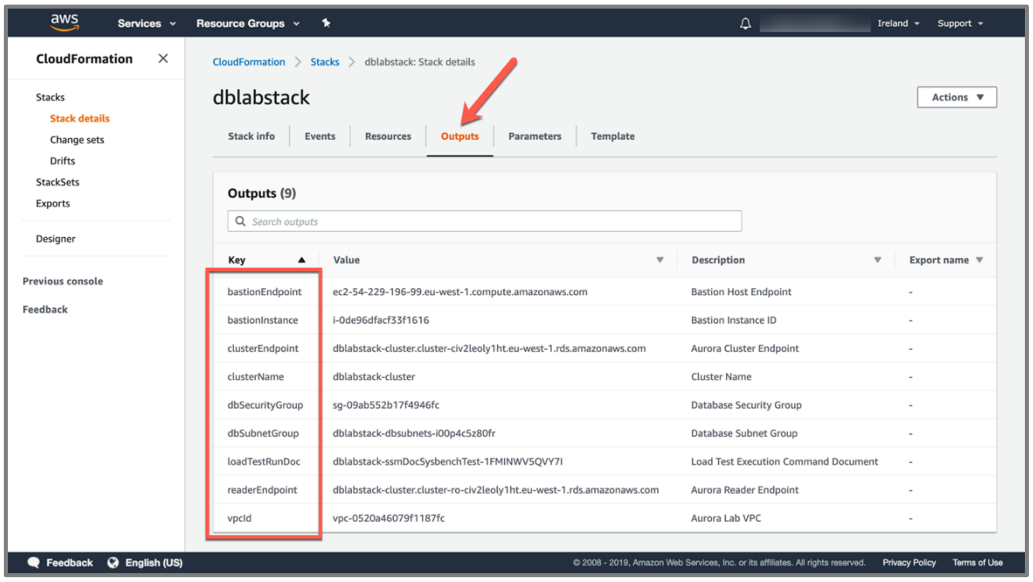
CloudFormation > Stacks > dblabstack: Stack details

dblabstack

Events

Timestamp	Logical ID	Status	Status reason
19 Feb 2019 00:35:53	dblabstack	CREATE_COMPLETE	-
19 Feb 2019 00:35:50	dbScalingPolicy	CREATE_COMPLETE	-
19 Feb 2019 00:35:50	dbScalingPolicy	CREATE_IN_PROGRESS	Resource creation Initiated
19 Feb 2019 00:35:49	dbScalingPolicy	CREATE_IN_PROGRESS	-
19 Feb 2019 00:35:46	dbScalableTarget	CREATE_COMPLETE	-
19 Feb 2019 00:35:46	dbScalableTarget	CREATE_IN_PROGRESS	Resource creation Initiated
19 Feb 2019 00:35:45	dbScalableTarget	CREATE_IN_PROGRESS	-
19 Feb 2019 00:35:41	dbNodeSecondary	CREATE_COMPLETE	-
19 Feb 2019 00:28:28	dbNodeSecondary		Resource creation Initiated

9. Once the status of the stack is CREATE_COMPLETE, click on the **Outputs** tab. The values here will be critical to the completion of the remainder of the lab. Please take a moment to save these values somewhere that you will have easy access to them during the remainder of the lab. The names that appear in the **Key** column are referenced directly in the instructions in subsequent steps, using the parameter format: **[outputKey]**



CloudFormation > Stacks > dblabstack: Stack details

dblabstack

Outputs (9)

Key	Value	Description	Export name
bastionEndpoint	ec2-54-229-196-99.eu-west-1.compute.amazonaws.com	Bastion Host Endpoint	-
bastionInstance	i-0de96dfacf33f1616	Bastion Instance ID	-
clusterEndpoint	dblabstack-cluster.cluster-civ2leoly1ht.eu-west-1.rds.amazonaws.com	Aurora Cluster Endpoint	-
clusterName	dblabstack-cluster	Cluster Name	-
dbSecurityGroup	sg-09ab552b17f4946fc	Database Security Group	-
dbSubnetGroup	dblabstack-dbsubnets-i00p4c5z80fr	Database Subnet Group	-
loadTestRunDoc	dblabstack-ssmDocSysbenchTest-1FMINWV5QVY7I	Load Test Execution Command Document	-
readerEndpoint	dblabstack-cluster.cluster-ro-civ2leoly1ht.eu-west-1.rds.amazonaws.com	Aurora Reader Endpoint	-
vpcId	vpc-0520a46079f1187fc	Aurora Lab VPC	-

Task 1.3 - Connecting to the workstation EC2 instance

For Windows users: We will use PuTTY and PuTTY Key Generator to connect to the workstation using SSH. If you do not have these applications already installed please use the steps in **Appendix 1 - Setting up PuTTY and connecting via SSH** below.

For macOS or Linux users: You can connect using the following command from a terminal, however you will need to change the permissions of the certificate file first:

```
chmod 0600 [path to downloaded .pem file]
```

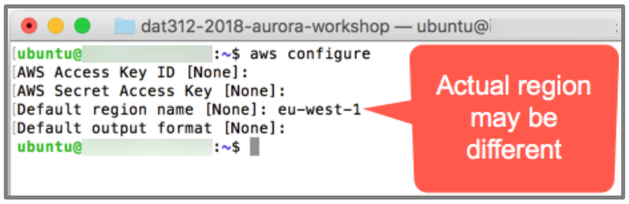
```
ssh -i [path to downloaded .pem file] ubuntu@[bastionEndpoint]
```

Task 1.4 – Set up the AWS CLI and seed the DB Cluster

1. Enter the following command in the SSH console to configure the AWS CLI:

   ```
   aws configure
   ```

   Then select the defaults for everything except the default region name. For the default region name, enter "`ap-southeast-1`".

   ```
   ubuntu@                    :~$ aws configure
   AWS Access Key ID [None]:
   AWS Secret Access Key [None]:
   Default region name [None]: eu-west-1
   Default output format [None]:
   ubuntu@                    :~$
   ```

   dat312-2018-aurora-workshop — ubuntu@
Actual region may be different

2. Connect to the Aurora database using the following command:

   ```
   mysql -h [clusterEndpoint] -u masteruser -p mylab
   ```

   Unless otherwise specified the cluster master username is **masteruser** and the password is **Password1**

3. Run the following queries on the database server, they will create a table, and load data from S3 into it:

   ```
   DROP TABLE IF EXISTS `sbtest1`;
   ```

   ```
   CREATE TABLE `sbtest1` (
    `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
    `k` int(10) unsigned NOT NULL DEFAULT '0',
    `c` char(120) NOT NULL DEFAULT '',
    `pad` char(60) NOT NULL DEFAULT '',
   PRIMARY KEY (`id`),
   KEY `k_1` (`k`)
   ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
   ```

```
LOAD DATA FROM S3 MANIFEST
's3-us-west-2://auroraworkshopassets/data/sbtable/sample.manifest'
REPLACE
INTO TABLE sbtest1
CHARACTER SET 'latin1'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\r\n';
```

PART 2. Cluster Endpoints and Auto Scaling

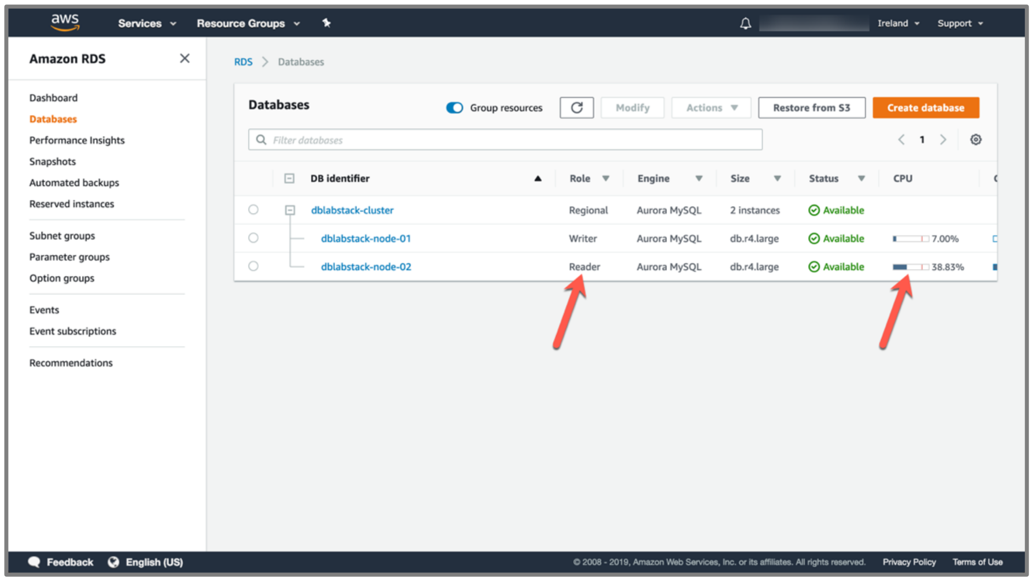
In this part we will explore the cluster endpoints and how auto scaling of read replicas operates.

Task 2.1 – Running a read-only workload

1. On the bastion host, execute the following statement:

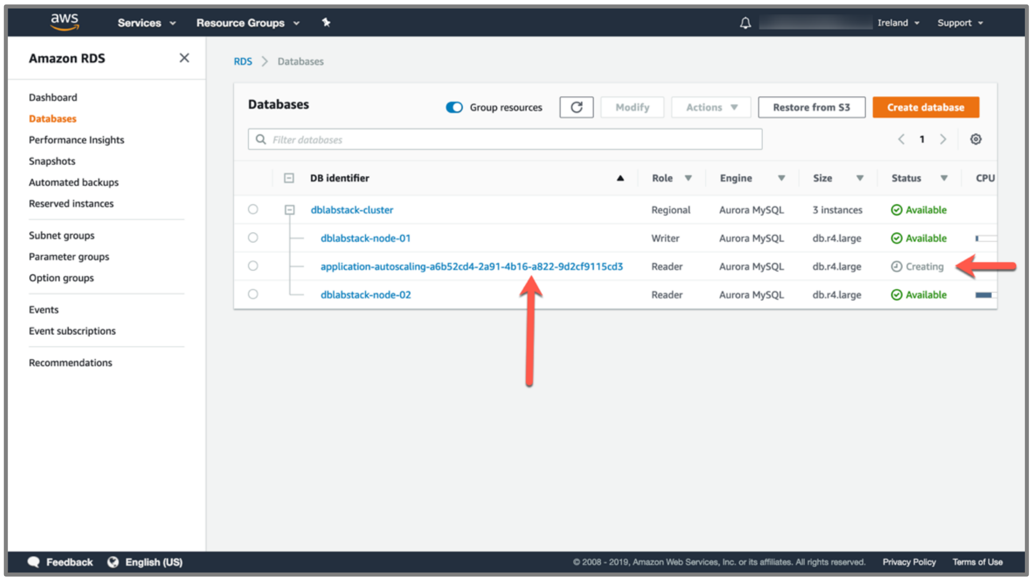
   ```
   python loadtest.py [readerEndpoint]
   ```

2. Open the **Amazon RDS** service console located at: https://amzn.to/2GNcMPk.
3. Take note that the reader node is currently receiving load. It may take a minute or more for the metrics to fully reflect the incoming load.



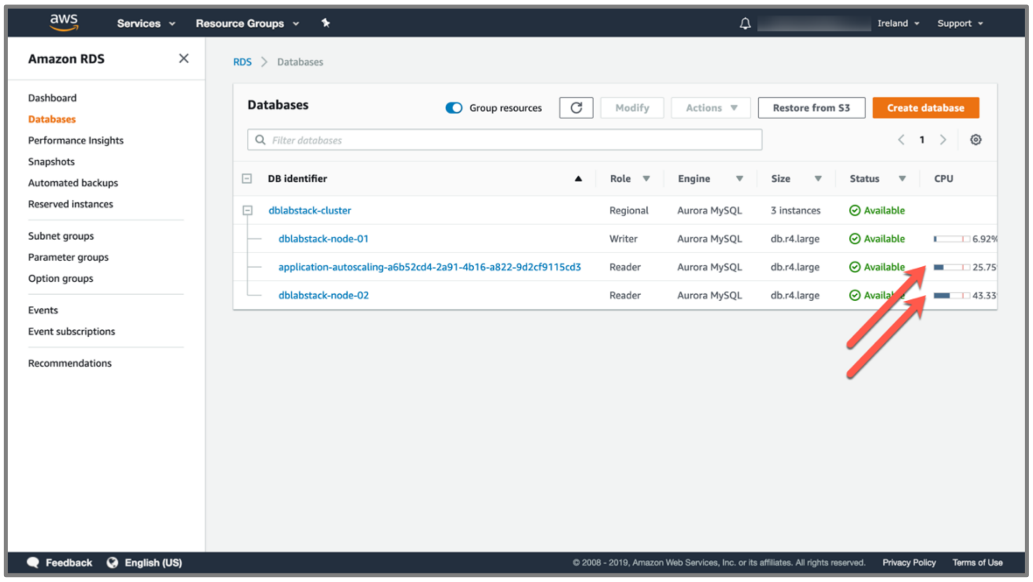
DB identifier	Role	Engine	Size	Status	CPU
dblabstack-cluster	Regional	Aurora MySQL	2 instances	Available	
dblabstack-node-01	Writer	Aurora MySQL	db.r4.large	Available	7.00%
dblabstack-node-02	Reader	Aurora MySQL	db.r4.large	Available	38.83%

4. After a few minutes return to the list of instances and notice that a new reader is being provisioned to your cluster.



DB identifier	Role	Engine	Size	Status	CPU
dblabstack-cluster	Regional	Aurora MySQL	3 instances	Available	
dblabstack-node-01	Writer	Aurora MySQL	db.r4.large	Available	
application-autoscaling-a6b52cd4-2a91-4b16-a822-9d2cf9115cd3	Reader	Aurora MySQL	db.r4.large	Creating	
dblabstack-node-02	Reader	Aurora MySQL	db.r4.large	Available	

5. Once the replicas are added, note that they are starting to receive load.



DB identifier	Role	Engine	Size	Status	CPU
dblabstack-cluster	Regional	Aurora MySQL	3 instances	Available	
dblabstack-node-01	Writer	Aurora MySQL	db.r4.large	Available	6.92%
application-autoscaling-a6b52cd4-2a91-4b16-a822-9d2cf9115cd3	Reader	Aurora MySQL	db.r4.large	Available	25.75
dblabstack-node-02	Reader	Aurora MySQL	db.r4.large	Available	43.33

6. You can now type `CTRL+C` on the bastion host to end the read load, if you wish to. After a while the additional readers will be removed automatically.

PART 3. Cloning and backtracking databases

Task 3.1 - Creating a Clone

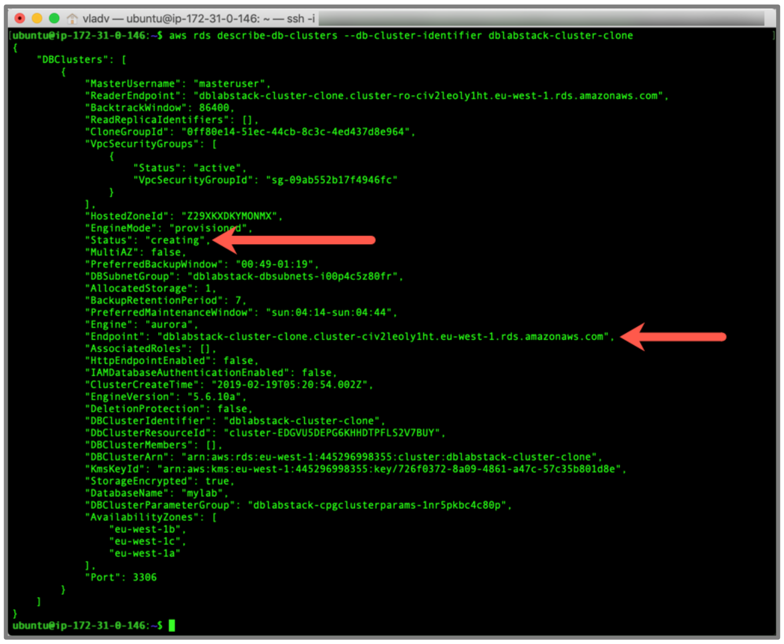
1. On the bastion host, enter:

   ```
   aws rds restore-db-cluster-to-point-in-time --restore-type copy-on-write -
   -use-latest-restorable-time --source-db-cluster-identifier [clusterName] -
   -db-cluster-identifier [clusterName]-clone --vpc-security-group-ids
   [dbSecurityGroup] --db-subnet-group-name [dbSubnetGroup] --backtrack-
   window 86400
   ```

2. Next, to check the status of the creation of your clone, enter the following command on the bastion host. The cloning process can take several minutes to complete. See the example output below.

   Note: This step will create the Aurora DB cluster itself, but without any compute nodes. You will will add a computer node in the next step.

   ```
   aws rds describe-db-clusters --db-cluster-identifier [clusterName]-clone
   ```



```
ubuntu@ip-172-31-0-146:~$ aws rds describe-db-clusters --db-cluster-identifier dblabstack-cluster-clone
{
    "DBClusters": [
        {
            "MasterUsername": "masteruser",
            "ReaderEndpoint": "dblabstack-cluster-clone.cluster-ro-civ2leoly1ht.eu-west-1.rds.amazonaws.com",
            "BacktrackWindow": 86400,
            "ReadReplicaIdentifiers": [],
            "CloneGroupId": "0ff80e14-51ec-44cb-8c3c-4ed437d8e964",
            "VpcSecurityGroups": [
                {
                    "Status": "active",
                    "VpcSecurityGroupId": "sg-09ab552b17f4946fc"
                }
            ],
            "HostedZoneId": "Z29XKXDKYMONMX",
            "EngineMode": "provisioned",
            "Status": "creating",
            "MultiAZ": false,
            "PreferredBackupWindow": "00:49-01:19",
            "DBSubnetGroup": "dblabstack-dbsubnets-i00p4c5z80fr",
            "AllocatedStorage": 1,
            "BackupRetentionPeriod": 7,
            "PreferredMaintenanceWindow": "sun:04:14-sun:04:44",
            "Engine": "aurora",
            "Endpoint": "dblabstack-cluster-clone.cluster-civ2leoly1ht.eu-west-1.rds.amazonaws.com",
            "AssociatedRoles": [],
            "HttpEndpointEnabled": false,
            "IAMDatabaseAuthenticationEnabled": false,
            "ClusterCreateTime": "2019-02-19T05:20:54.002Z",
            "EngineVersion": "5.6.10a",
            "DeletionProtection": false,
            "DBClusterIdentifier": "dblabstack-cluster-clone",
            "DbClusterResourceId": "cluster-EDGVU5DEPG6KHHDTPFLS2V7BUY",
            "DBClusterMembers": [],
            "DBClusterArn": "arn:aws:rds:eu-west-1:445296998355:cluster:dblabstack-cluster-clone",
            "KmsKeyId": "arn:aws:kms:eu-west-1:445296998355:key/726f0372-8a09-4861-a47c-57c35b801d8e",
            "StorageEncrypted": true,
            "DatabaseName": "mylab",
            "DBClusterParameterGroup": "dblabstack-cpgclusterparams-1nr5pkbc4c80p",
            "AvailabilityZones": [
                "eu-west-1b",
                "eu-west-1c",
                "eu-west-1a"
            ],
            "Port": 3306
        }
    ]
}
ubuntu@ip-172-31-0-146:~$
```

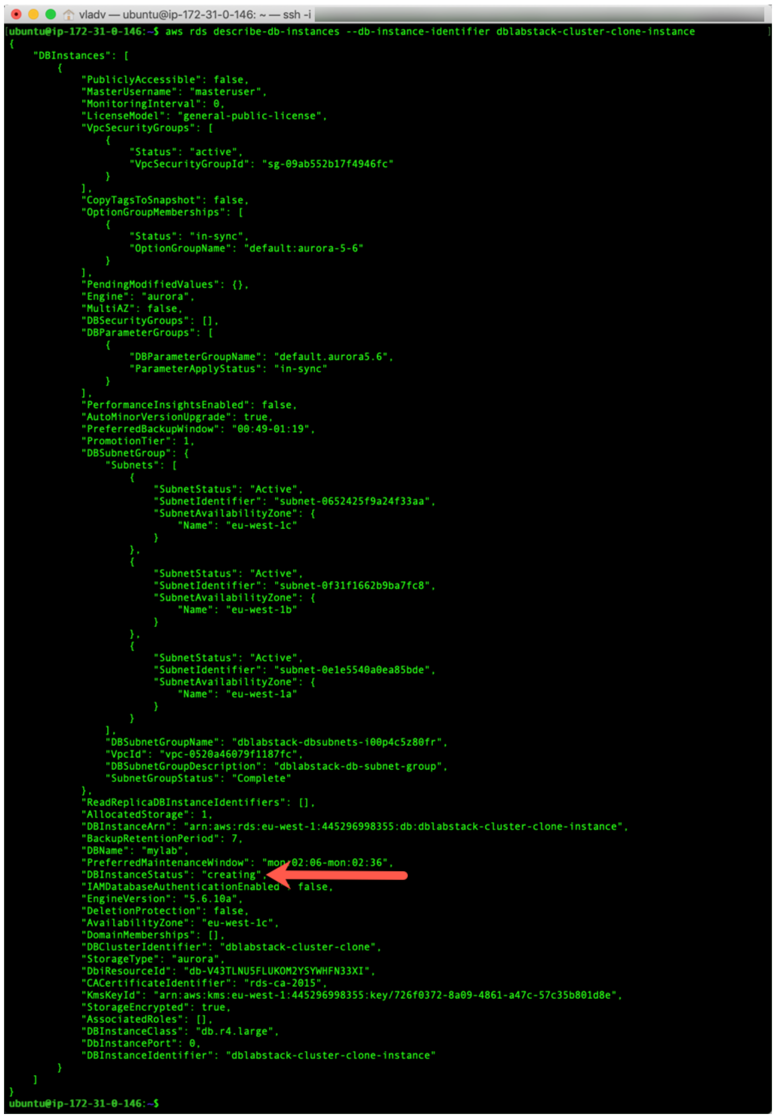
3. Take note of both the **"Status"** and the **"Endpoint."** Once the **Status** becomes **available**, you can add an instance to the cluster and once the instance is added, you will want to connect to the cluster via the **Endpoint** value. To add an instance to the cluster once the status becomes **available**, enter the following:

   ```
   aws rds create-db-instance --db-instance-class db.r4.large --engine aurora
   --db-cluster-identifier [clusterName]-clone --db-instance-identifier
   [clusterName]-clone-instance
   ```

4. To check the creation of the instance, enter the following at the command line:

```
aws rds describe-db-instances --db-instance-identifier [clusterName]-
clone-instance
```

5. Once the **DBInstanceStatus** changes from **creating** to **available**, you have a functioning clone. Creating a node in a cluster also takes several minutes.



```
ubuntu@ip-172-31-0-146:~$ aws rds describe-db-instances --db-instance-identifier dblabstack-cluster-clone-instance
{
    "DBInstances": [
        {
            "PubliclyAccessible": false,
            "MasterUsername": "masteruser",
            "MonitoringInterval": 0,
            "LicenseModel": "general-public-license",
            "VpcSecurityGroups": [
                {
                    "Status": "active",
                    "VpcSecurityGroupId": "sg-09ab552b17f4946fc"
                }
            ],
            "CopyTagsToSnapshot": false,
            "OptionGroupMemberships": [
                {
                    "Status": "in-sync",
                    "OptionGroupName": "default:aurora-5-6"
                }
            ],
            "PendingModifiedValues": {},
            "Engine": "aurora",
            "MultiAZ": false,
            "DBSecurityGroups": [],
            "DBParameterGroups": [
                {
                    "DBParameterGroupName": "default.aurora5.6",
                    "ParameterApplyStatus": "in-sync"
                }
            ],
            "PerformanceInsightsEnabled": false,
            "AutoMinorVersionUpgrade": true,
            "PreferredBackupWindow": "00:49-01:19",
            "PromotionTier": 1,
            "DBSubnetGroup": {
                "Subnets": [
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-0652425f9a24f33aa",
                        "SubnetAvailabilityZone": {
                            "Name": "eu-west-1c"
                        }
                    },
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-0f31f1662b9ba7fc8",
                        "SubnetAvailabilityZone": {
                            "Name": "eu-west-1b"
                        }
                    },
                    {
                        "SubnetStatus": "Active",
                        "SubnetIdentifier": "subnet-0e1e5540a0ea85bde",
                        "SubnetAvailabilityZone": {
                            "Name": "eu-west-1a"
                        }
                    }
                ],
                "DBSubnetGroupName": "dblabstack-dbsubnets-i00p4c5z80fr",
                "VpcId": "vpc-0520a46079f1187fc",
                "DBSubnetGroupDescription": "dblabstack-db-subnet-group",
                "SubnetGroupStatus": "Complete"
            },
            "ReadReplicaDBInstanceIdentifiers": [],
            "AllocatedStorage": 1,
            "DBInstanceArn": "arn:aws:rds:eu-west-1:445296998355:db:dblabstack-cluster-clone-instance",
            "BackupRetentionPeriod": 7,
            "DBName": "mylab",
            "PreferredMaintenanceWindow": "mon:02:06-mon:02:36",
            "DBInstanceStatus": "creating",
            "IAMDatabaseAuthenticationEnabled": false,
            "EngineVersion": "5.6.10a",
            "DeletionProtection": false,
            "AvailabilityZone": "eu-west-1c",
            "DomainMemberships": [],
            "DBClusterIdentifier": "dblabstack-cluster-clone",
            "StorageType": "aurora",
            "DbiResourceId": "db-V43TLNU5FLUKOM2YSYWHFN33XI",
            "CACertificateIdentifier": "rds-ca-2015",
            "KmsKeyId": "arn:aws:kms:eu-west-1:445296998355:key/726f0372-8a09-4861-a47c-57c35b801d8e",
            "StorageEncrypted": true,
            "AssociatedRoles": [],
            "DBInstanceClass": "db.r4.large",
            "DbInstancePort": 0,
            "DBInstanceIdentifier": "dblabstack-cluster-clone-instance"
        }
    ]
}
ubuntu@ip-172-31-0-146:~$
```

6. Once your instance is created, connect to the instance using the following command:

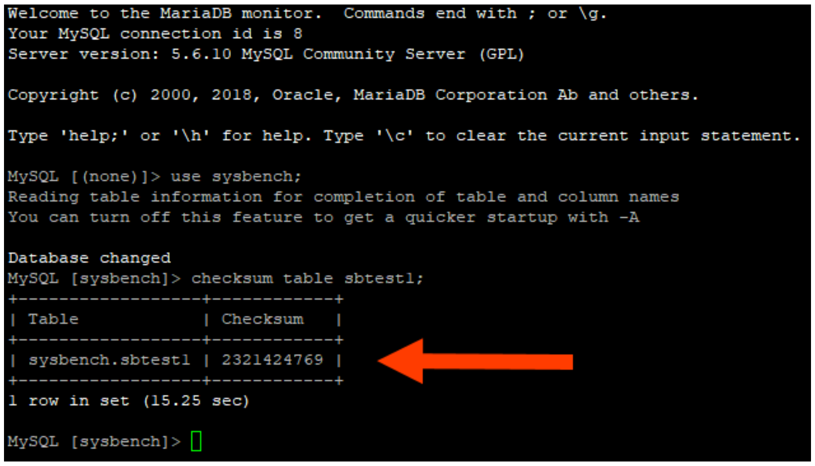
```
mysql -h [cluster endpoint of clone cluster] -u masteruser -p mylab
```

Note: the master user account credentials will be the same as with the source of the cloned cluster. If you customized the CloudFormation template and changed the values, use the customized username and password.

7. In order to verify that the clone is identical to the source, we will perform a checksum of the sbtest1 table using the following:

```
checksum table sbtest1;
```

8. The output of your commands should look similar to the example below:



```
Welcome to the MariaDB 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, MariaDB Corporation Ab and others.

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

MySQL [(none)]> use sysbench;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [sysbench]> checksum table sbtest1;
+------------------+------------+
| Table            | Checksum   |
+------------------+------------+
| sysbench.sbtest1 | 2321424769 |
+------------------+------------+
1 row in set (15.25 sec)

MySQL [sysbench]>
```

9. Please take note of the value for your specific clone cluster.

10. Next, we will disconnect from the clone and connect to the original cluster with the following:

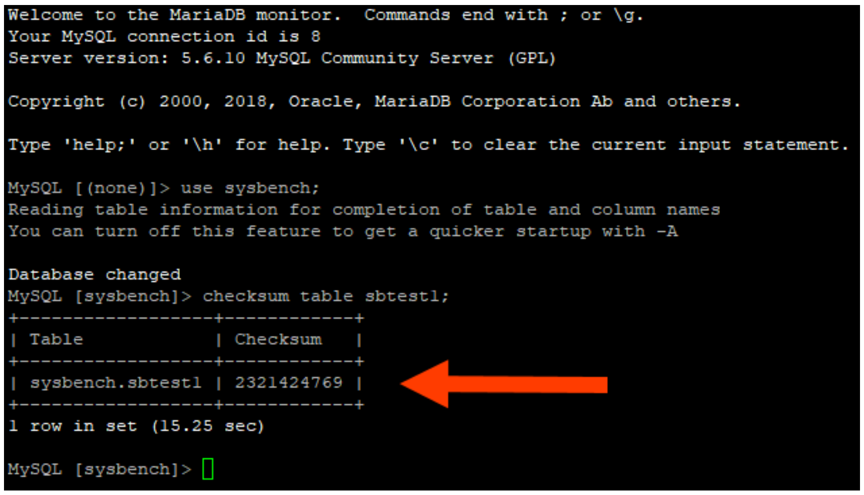
    quit;

    mysql -h [clusterEndpoint] -u masteruser -p mylab

11. Next, we will execute the same commands that we executed on the clone:

    checksum table sbtest1;

12. Please take note of the value for your specific source cluster. The checksum should be identical.



```
Welcome to the MariaDB 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, MariaDB Corporation Ab and others.

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

MySQL [(none)]> use sysbench;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [sysbench]> checksum table sbtest1;
+------------------+------------+
| Table            | Checksum   |
+------------------+------------+
| sysbench.sbtest1 | 2321424769 |
+------------------+------------+
1 row in set (15.25 sec)

MySQL [sysbench]>
```

Task 3.2 - Backtracking the Database

1. Reconnect to the cloned cluster using:

```
quit;

mysql -h [cluster endpoint of clone cluster] -u masteruser -p mylab
```

2. Drop the **sbtest1** table:

   Note: Consider executing the commands below one at a time, waiting a few seconds between each one. This will make it easier to determine a good point in time for testing backtrack.

```
select current_timestamp();

drop table sbtest1;

select current_timestamp();

quit;
```

3. Remember or save the time markers displayed above, you will use them as references later.
4. Run the following command to replace the dropped table using the sysbench command:

```
sysbench oltp_write_only --threads=1 --mysql-host=[cluster endpoint of
clone cluster] --mysql-user=masteruser --mysql-password=Password1 --mysql-
port=3306 --tables=1 --mysql-db=mylab --table-size=1000000 prepare
```

5. Reconnect to the cloned cluster, and checksum the table again, the checksum value should be different than both the original clone value and source cluster:

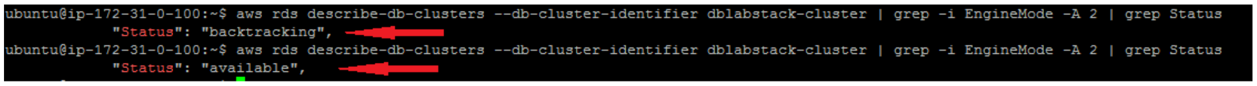
```
mysql -h [cluster endpoint of clone cluster] -u masteruser -p mylab

checksum table sbtest1;

quit;
```

6. Backtrack the database to a time slightly after the second time marker. (Right after dropping the table).
   For e.g. If the second time marker is "2019-06-03 19:19:40", specify *backtrack-to* parameter value as "2019-06-03T19:19:40Z" in the command below.

```
aws rds backtrack-db-cluster --db-cluster-identifier [clusterName]-clone -
-backtrack-to "yyyy-mm-ddThh:mm:ssZ"
```

7. Run the below command to track the progress of the backtracking operation. The operation should complete in a few minutes. Status will change from "backtracking" to "available" when backtrack is complete.

```
aws rds describe-db-clusters --db-cluster-identifier [clusterName]-clone |
grep -i EngineMode -A 2 | grep Status
```



```
ubuntu@ip-172-31-0-100:~$ aws rds describe-db-clusters --db-cluster-identifier dblabstack-cluster | grep -i EngineMode -A 2 | grep Status
            "Status": "backtracking",
ubuntu@ip-172-31-0-100:~$ aws rds describe-db-clusters --db-cluster-identifier dblabstack-cluster | grep -i EngineMode -A 2 | grep Status
            "Status": "available",
```

8. Wait for backtracking to complete and then connect back to the database. The **sbtest1** table should be missing from the database.

```
mysql -h [cluster endpoint of clone cluster] -u masteruser -p mylab

show tables;

quit;
```

9. Now backtrack again to a time slightly before the first time marker above. (Right before dropping the table).

```
aws rds backtrack-db-cluster --db-cluster-identifier [clusterName]-clone -
-backtrack-to "yyyy-mm-ddThh:mm:ssZ"
```

10. Run the below command to track the progress of the backtracking operation. The operation should complete in a few minutes.

```
aws rds describe-db-clusters --db-cluster-identifier [clusterName]-clone |
grep -i EngineMode -A 2 | grep Status
```

11. Connect back to the database. The **sbtest1** table should now be available in the database again, but contain the original data set.

```
mysql -h [cluster endpoint of clone cluster] -u masteruser -p mylab

show tables;

quit;
```

PART 4. Performance Insights

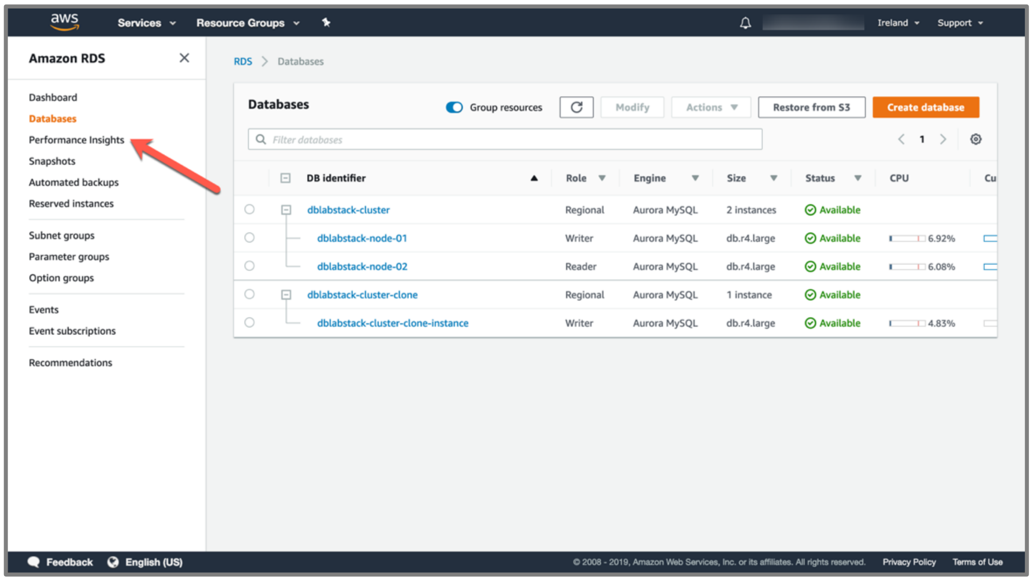
Task 4.1 – Generate Load on Your Database Cluster

1. You will use Percona's TPCC-like benchmark script based on sysbench to generate load. For simplicity we have packaged the correct set of commands in an AWS Systems Manager Command Document. You will use AWS Systems Manager Run Command to execute the test.
2. On the workstation host, execute the following statement:

```
aws ssm send-command --document-name [loadTestRunDoc] --instance-ids
[bastionInstance]
```

3. The command will be sent to the workstation EC2 instance which will prepare the test data set and run the load test. It may take up to a minute for CloudWatch to reflect the additional load in the metrics.

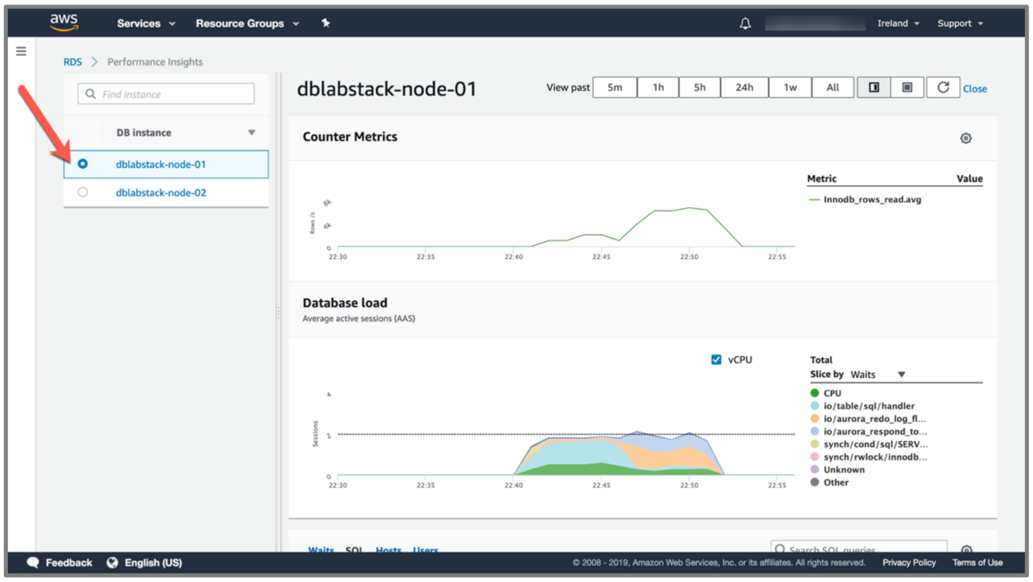
4. Navigate to the RDS service console (https://amzn.to/2GNcMPk) and click on **Performance Insights** in the left side navigation bar.



Databases

DB identifier	Role	Engine	Size	Status	CPU
dblabstack-cluster	Regional	Aurora MySQL	2 instances	Available	
dblabstack-node-01	Writer	Aurora MySQL	db.r4.large	Available	6.92%
dblabstack-node-02	Reader	Aurora MySQL	db.r4.large	Available	6.08%
dblabstack-cluster-clone	Regional	Aurora MySQL	1 instance	Available	
dblabstack-cluster-clone-instance	Writer	Aurora MySQL	db.r4.large	Available	4.83%

5. Examine the performance of your DB instance **demostack-node-01** using Performance Insights. What conclusions can you reach?



Part 5. Delete AWS resources

Follow these steps to clean up AWS resources after you are done with all the lab activities.

Task 5.1 – Delete Cloned Aurora Cluster

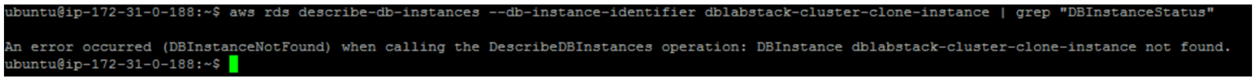
1. Delete the Aurora instance on the cloned cluster by entering the following on the bastion host.

```
aws rds delete-db-instance --db-instance-identifier [clusterName]-
clone-instance --skip-final-snapshot
```

2. Verify the Aurora instance is deleted, by entering the following on the command line.

   ```
   aws rds describe-db-instances --db-instance-identifier [clusterName]-
   clone-instance | grep "DBInstanceStatus"
   ```

   Once the instance is deleted, you will see the following error message while running the command.



```
ubuntu@ip-172-31-0-188:~$ aws rds describe-db-instances --db-instance-identifier dblabstack-cluster-clone-instance | grep "DBInstanceStatus"

An error occurred (DBInstanceNotFound) when calling the DescribeDBInstances operation: DBInstance dblabstack-cluster-clone-instance not found.
ubuntu@ip-172-31-0-188:~$
```

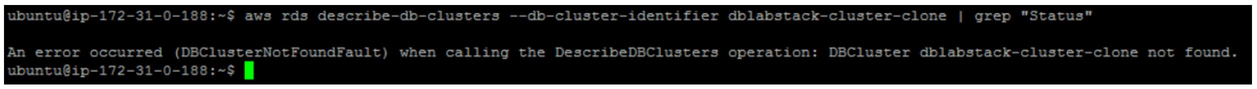
3. Delete the cloned Aurora cluster by entering the following on the command line.

   ```
   aws rds delete-db-cluster --db-cluster-identifier [clusterName]-clone -
   -skip-final-snapshot
   ```

4. Verify the Cloned Aurora Cluster is delete, by entering the following on the command line.

   ```
   aws rds describe-db-clusters --db-cluster-identifier [clusterName]-
   clone | grep "Status"
   ```

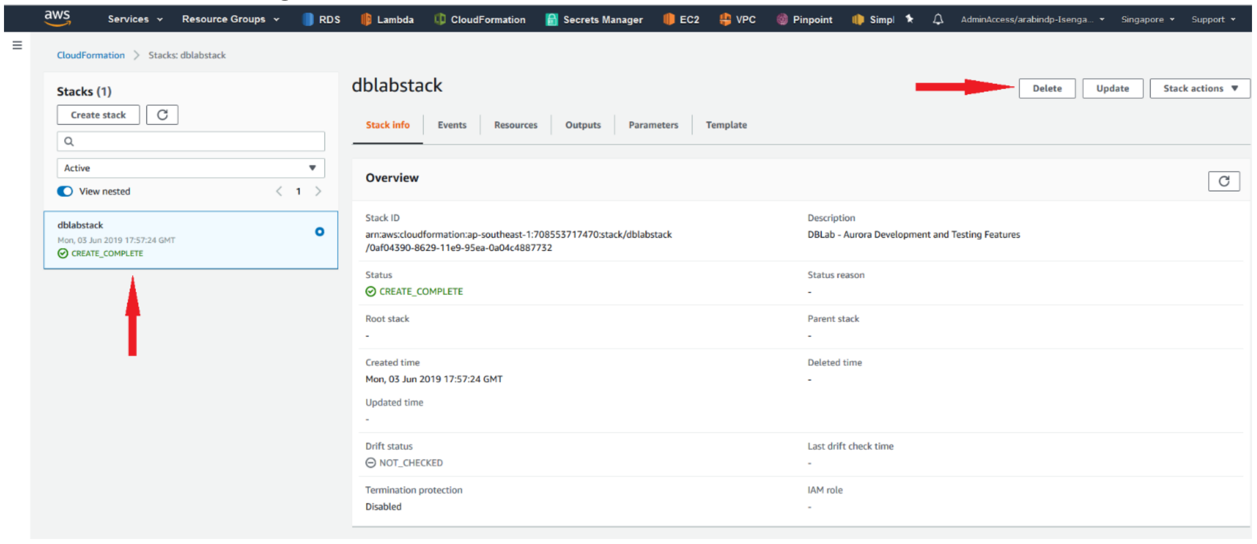
   When the cluster is deleted, you will see the following error message while running the command.



```
ubuntu@ip-172-31-0-188:~$ aws rds describe-db-clusters --db-cluster-identifier dblabstack-cluster-clone | grep "Status"

An error occurred (DBClusterNotFoundFault) when calling the DescribeDBClusters operation: DBCluster dblabstack-cluster-clone not found.
ubuntu@ip-172-31-0-188:~$
```

Task 5.2 – Delete CloudFormation stack

1. Open the **CloudFormation** service console located at: https://amzn.to/2BIn3Jh.
2. Select the CloudFormation stack you created as part of this lab in the left panel and select Delete button on the right side of the screen.



dblabstack
Mon, 03 Jun 2019 17:57:24 GMT
CREATE_COMPLETE

dblabstack

Stack ID
arn:aws:cloudformation:ap-southeast-1:708553717470:stack/dblabstack
/0af04390-8629-11e9-95ea-0a04c4887732

Description
DBLab - Aurora Development and Testing Features

Status
CREATE_COMPLETE

Status reason
-

Root stack
-

Parent stack
-

Created time
Mon, 03 Jun 2019 17:57:24 GMT

Deleted time
-

Updated time
-

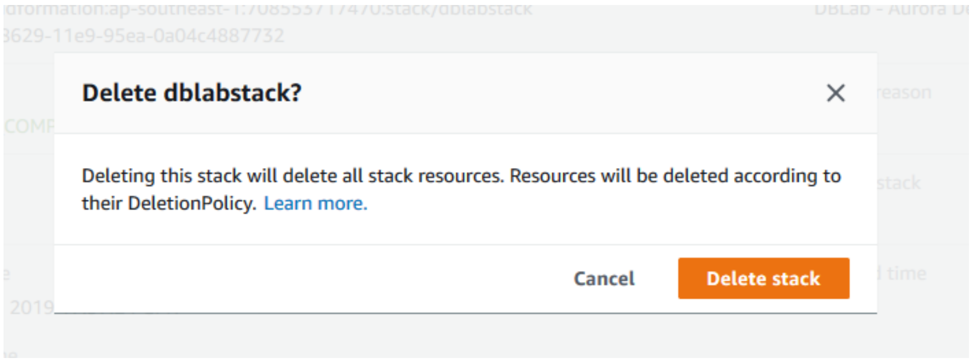
Drift status
NOT_CHECKED

Last drift check time
-

Termination protection
Disabled

IAM role
-

3. Select Delete stack when prompted. It will take 15 minutes to delete the stack.



Delete dblabstack?

Deleting this stack will delete all stack resources. Resources will be deleted according to their DeletionPolicy. Learn more.

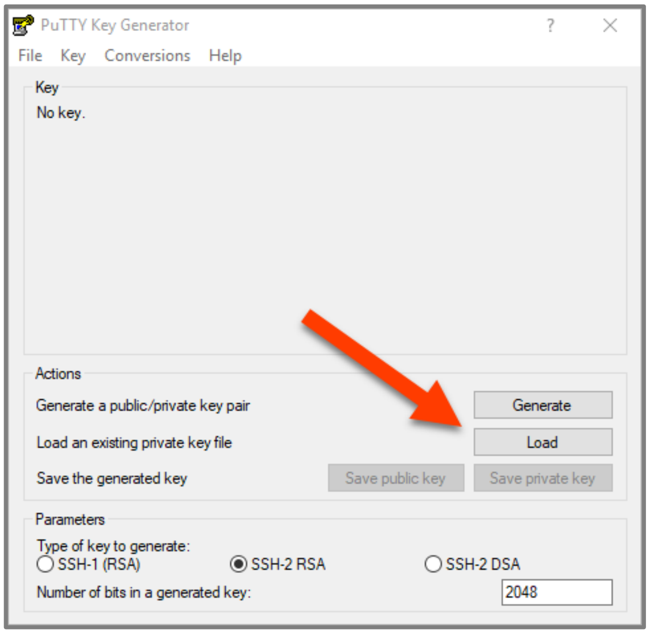
APPENDIX

Appendix 1 – Setting up PuTTY and connecting via SSH

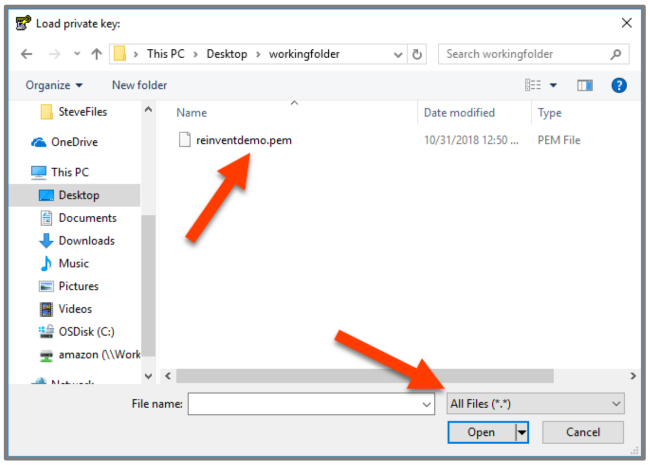
For Windows users, please download **PuTTY** (putty) and the **PuTTY Key Generator** (puttygen) from the following links:

https://bit.ly/2ETxptZ
https://bit.ly/2AEoOHp

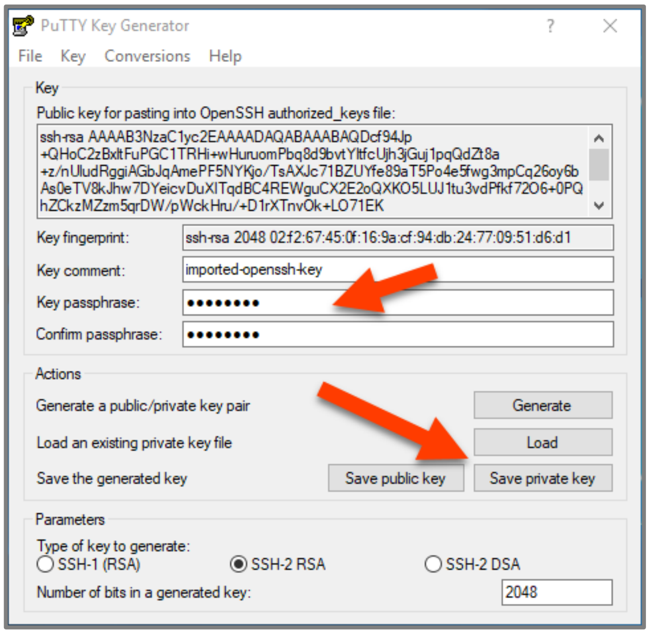
1. Once you have downloaded putty and puttygen, open **puttygen** and click on **Load**.



2. Please make sure that the file filter is set to "All Files (*.*) and then select **dblabkeys.pem**.



3. Fill in the **Key passphrase** and **Confirm passphrase** fields with a password of your choice that will be used to encrypt your private key and then click **Save private key**. Please use "`dblabkeys.ppk`" as your new key name.



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDcf94Jp
+QHoC2zBxltFuPGC1TRHi+wHuruomPbq8d9bvtYltfcUjh3jGuj1pqQdZt8a
+z/nUludRggiAGbJqAmePF5NYKjo/TsAXJc71BZUYfe89aT5Po4e5fwg3mpCq26oy6b
As0eTV8kJhw7DYeicvDuXITqdBC4REWguCX2E2oQXKO5LUJ1tu3vdPfkf72O6+0PQ
hZCkzMZzm5qrDW/pWckHru/+D1rXTnvOk+LO71EK
```

Key fingerprint: ssh-rsa 2048 02:f2:67:45:0f:16:9a:cf:94:db:24:77:09:51:d6:d1

Key comment: imported-openssh-key

Key passphrase: ••••••••

Confirm passphrase: ••••••••

Generate a public/private key pair

Load an existing private key file

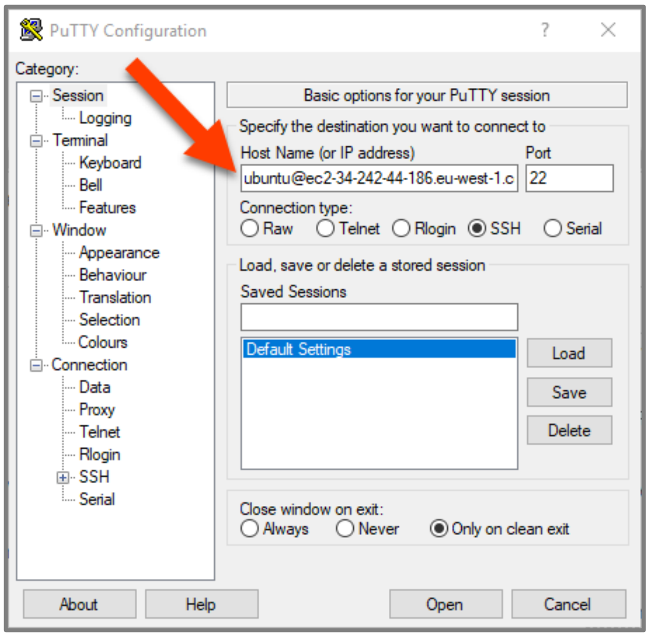
Save the generated key

Type of key to generate: SSH-1 (RSA) [✓ SSH-2 RSA] SSH-2 DSA

Number of bits in a generated key: 2048

4. Next, open putty and enter into the **Host Name (or IP address)** field the following value:

`ubuntu@[bastionEndpoint]`



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): ubuntu@ec2-34-242-44-186.eu-west-1.c

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

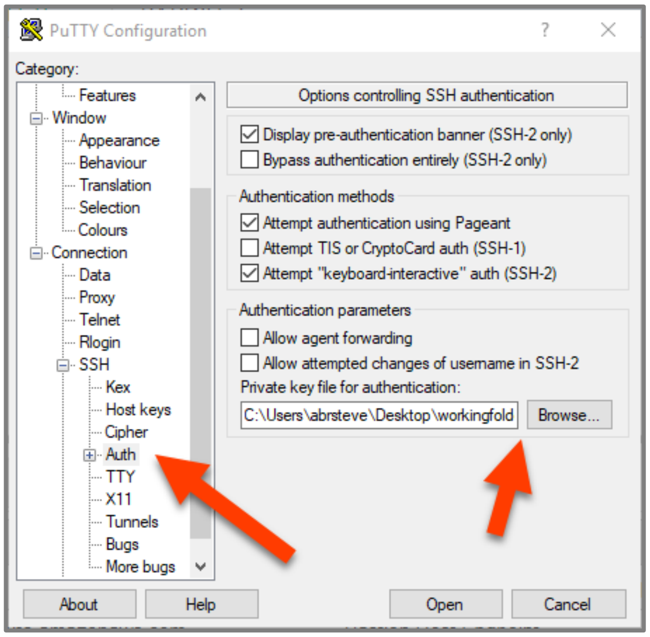
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

5. Next, navigate within PuTTY to **Connection** → **SSH** → **Auth** and browse to the **reinventdemo.ppk** file that you created with the PuTTY Key Generator previously, and then click **Open**.


PuTTY Configuration

Category:

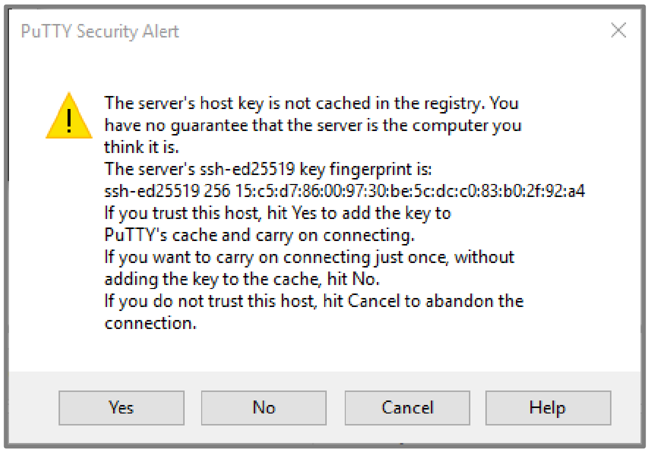
Options controlling SSH authentication

☑ Display pre-authentication banner (SSH-2 only)
☐ Bypass authentication entirely (SSH-2 only)

Authentication methods
☑ Attempt authentication using Pageant
☐ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

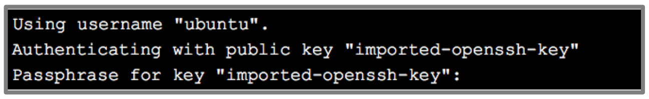
Authentication parameters
☐ Allow agent forwarding
☐ Allow attempted changes of username in SSH-2
Private key file for authentication:
C:\Users\abrsteve\Desktop\workingfold

6. When prompted by the PuTTY Security Alert, click **Yes**.


PuTTY Security Alert

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's ssh-ed25519 key fingerprint is:
ssh-ed25519 256 15:c5:d7:86:00:97:30:be:5c:dc:c0:83:b0:2f:92:a4
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

7. Next, enter the password that you configured when you created the **dblabkeys.ppk** private file previously.


```
Using username "ubuntu".
Authenticating with public key "imported-openssh-key"
Passphrase for key "imported-openssh-key":
```