Introduction

About this Workshop

Securing the data stored in your MySQL Server against failures is key towards achieving Regulatory Compliance. This workshop covers the installation, configuration and testing of MySQL High Availability architectures. We will go through how to setup and run different Highly Availability Solutions.

Estimated Workshop Time: 1 hours 30 minutes (This estimate is for the entire workshop - it is the sum of the estimates provided for each of the labs included in the workshop.)

Your Free Tier server should be accessible for a couple of days after this workshop so what you do not finish when following instruction will be able to be covered later at your own pace.

Prerequisite

For working on this workshop you will need a clean computer to run on. We should have provided you with a link on setting up a free account on Oracle Cloud Infrastructure (OCI).

If you have not set it up, then please go through the following steps:

- OCI Free Tier Setup

Objectives

In this workshop, you will learn how to work with

- MySQL Enterprise Edition
- MySQL Shell
- MySQL High Availability
- MySQL Router

Learn More

- MySQL Home Page
- MySQL Replication
- MySQL Group Replication
- MySQL Shell
- MySQL ReplicaSet
- MySQL InnoDB Cluster
- MySQL InnoDB ClusterSet

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - Dale Dasker, August 2022

Create your Virtual Cloud Network and Related Components

Introduction

Create your VCN and subnets

The first step when deploying a Compute in Oracle Cloud Infrastructure (OCI) is to setup a Virtual Cloud Network. The VCN will be used to connect your linux instance to the internet. You will configure all the components needed to create your virtual network.

Estimated Time: 10 minutes

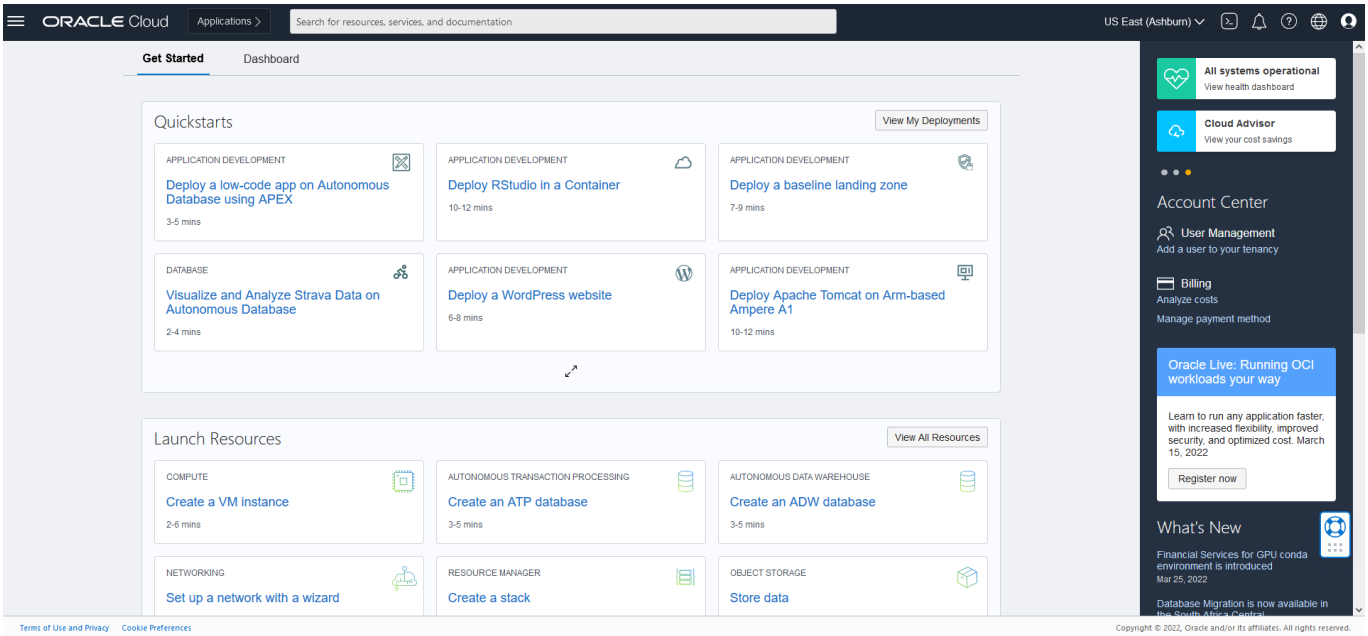
Objectives

In this lab, you will be guided through the following tasks:

- Create Virtual Cloud Network
- Configure security list to allow MySQL incoming connections

Prerequisites

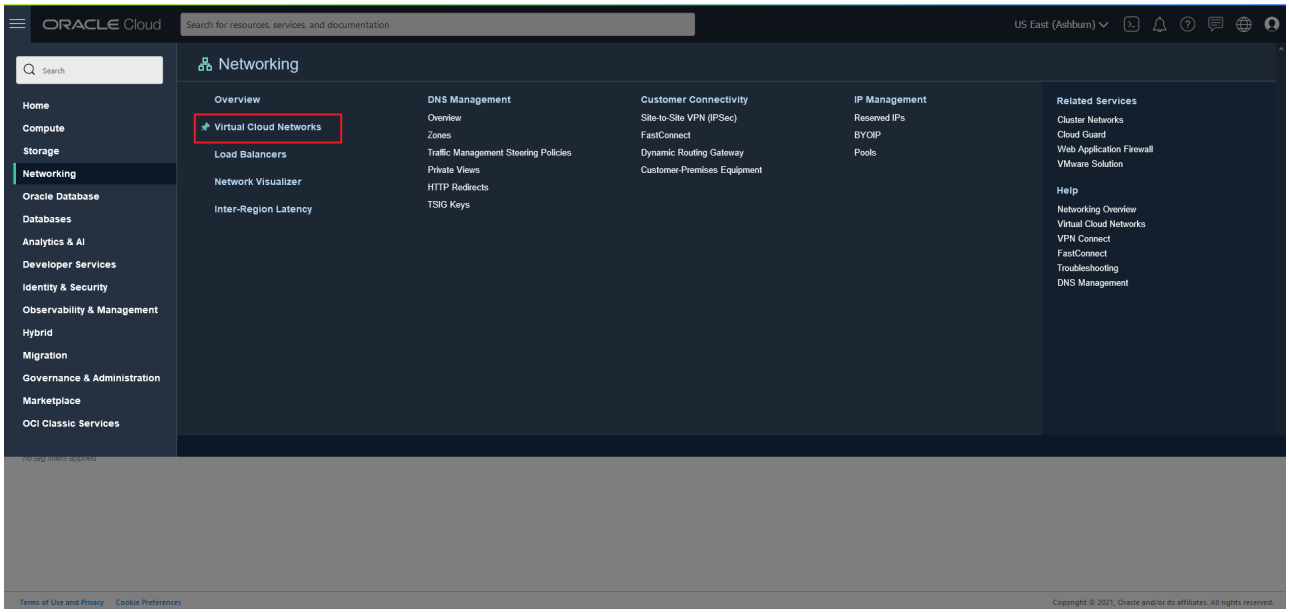
- An Oracle Free Tier or Paid Cloud Account
- A web browser
- Login to OCI to land on OCI Dashboard (This image shows a trial account)



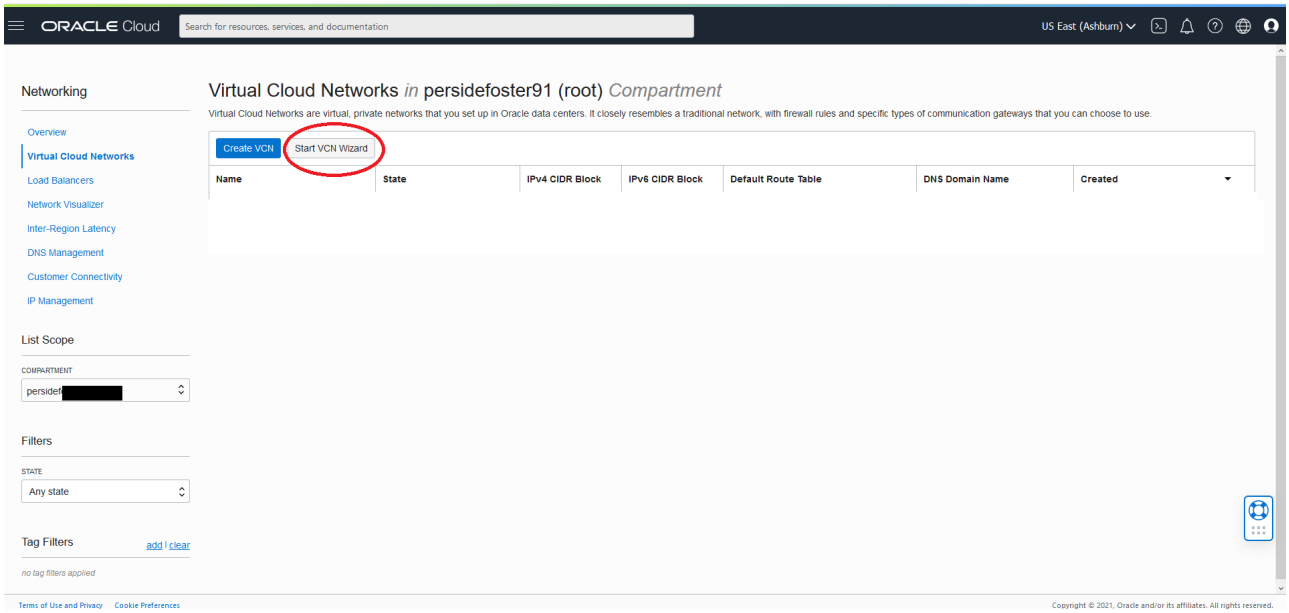
Task 1: Create Virtual Cloud Network

1. Click Navigation Menu
   Select Networking

Select Virtual Cloud Networks

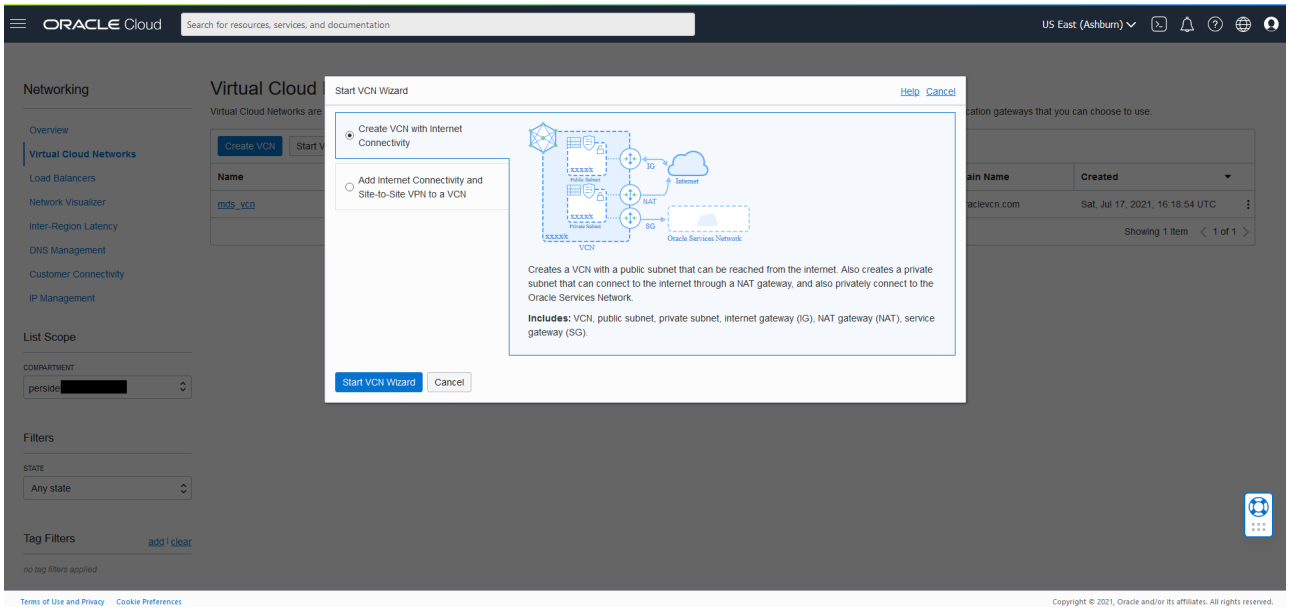


2. Click **Start VCN Wizard**



3. Select 'Create VCN with Internet Connectivity'

Click 'Start VCN Wizard'

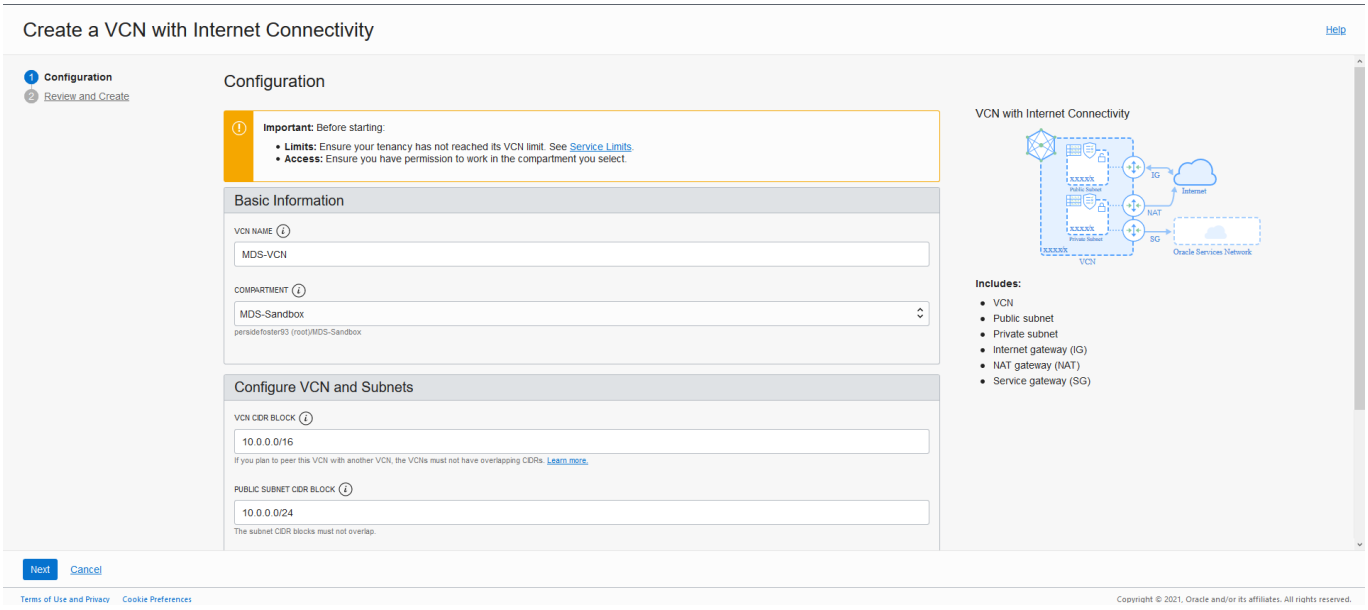


4. Create a VCN with Internet Connectivity

   On Basic Information, complete the following fields:

VCN Name: `<copy>myvcn</copy>` Compartment: Select **(root)**

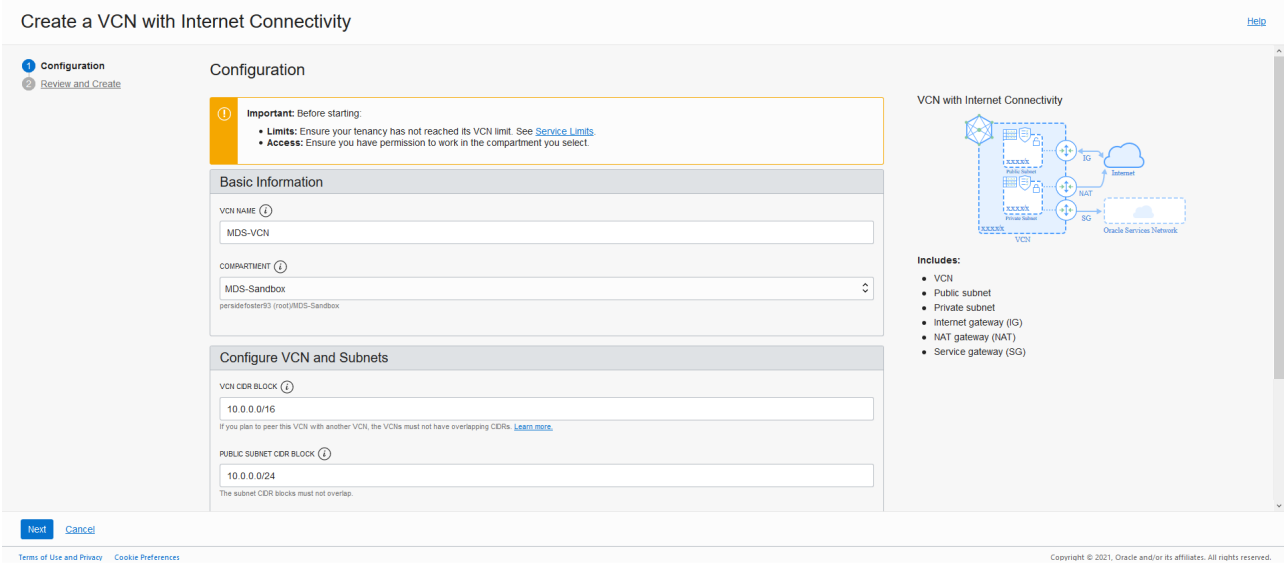
Your screen should look similar to the following



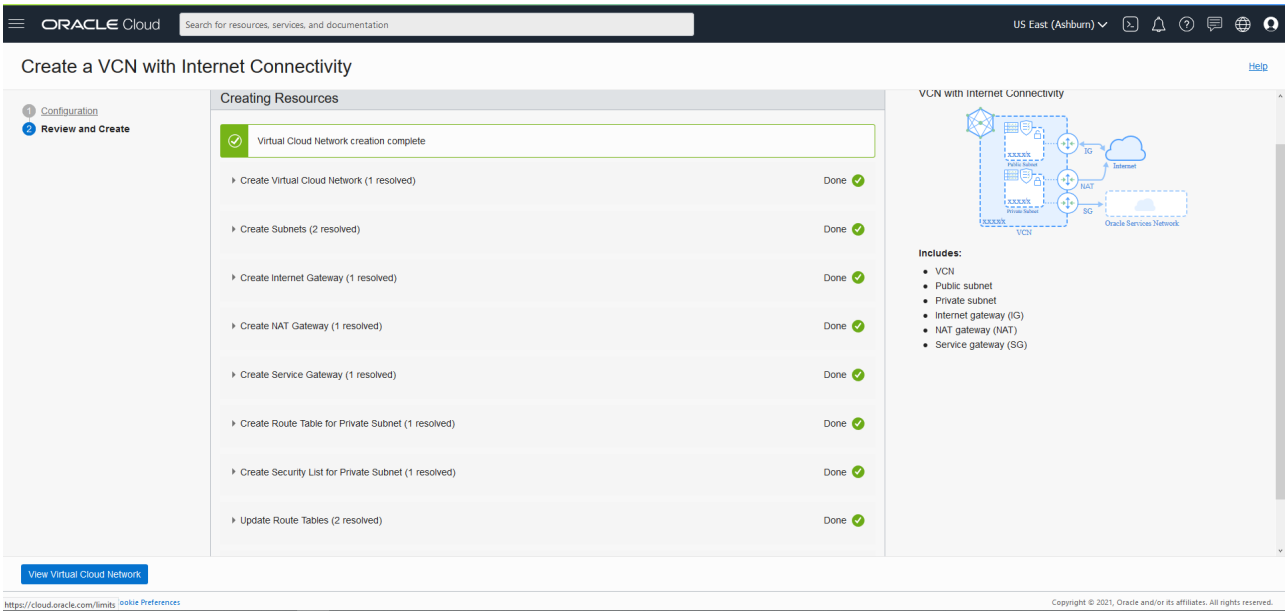
5. Click 'Next' at the bottom of the screen

6. Review Oracle Virtual Cloud Network (VCN), Subnets, and Gateways

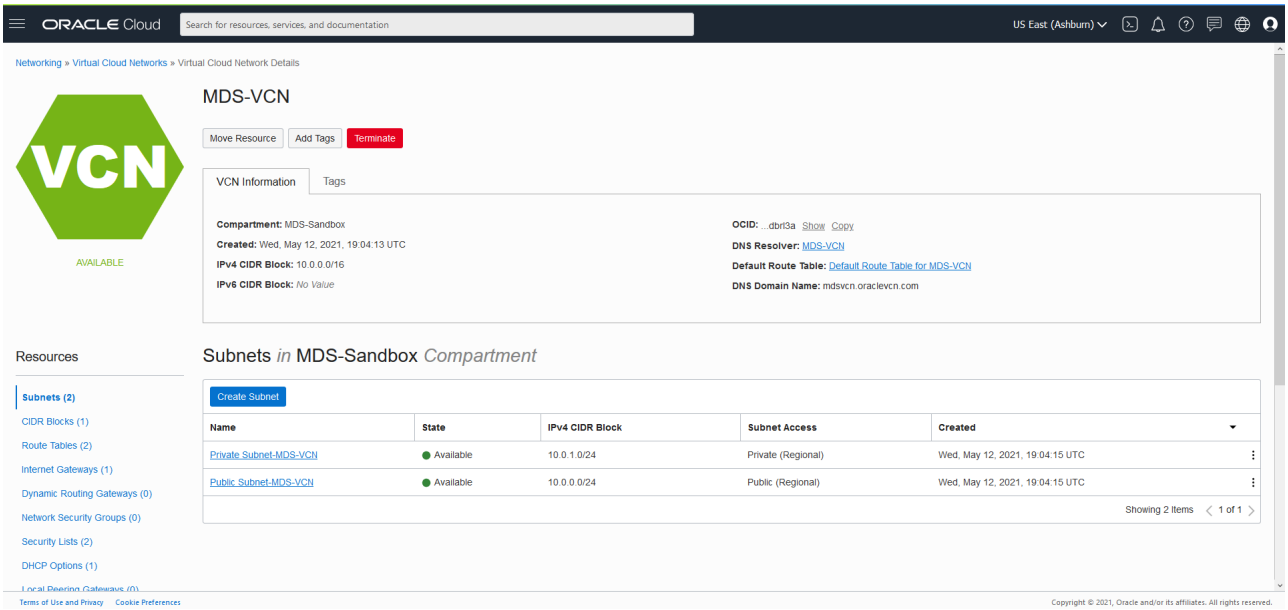
Click 'Create' to create the VCN



7. The Virtual Cloud Network creation is completing

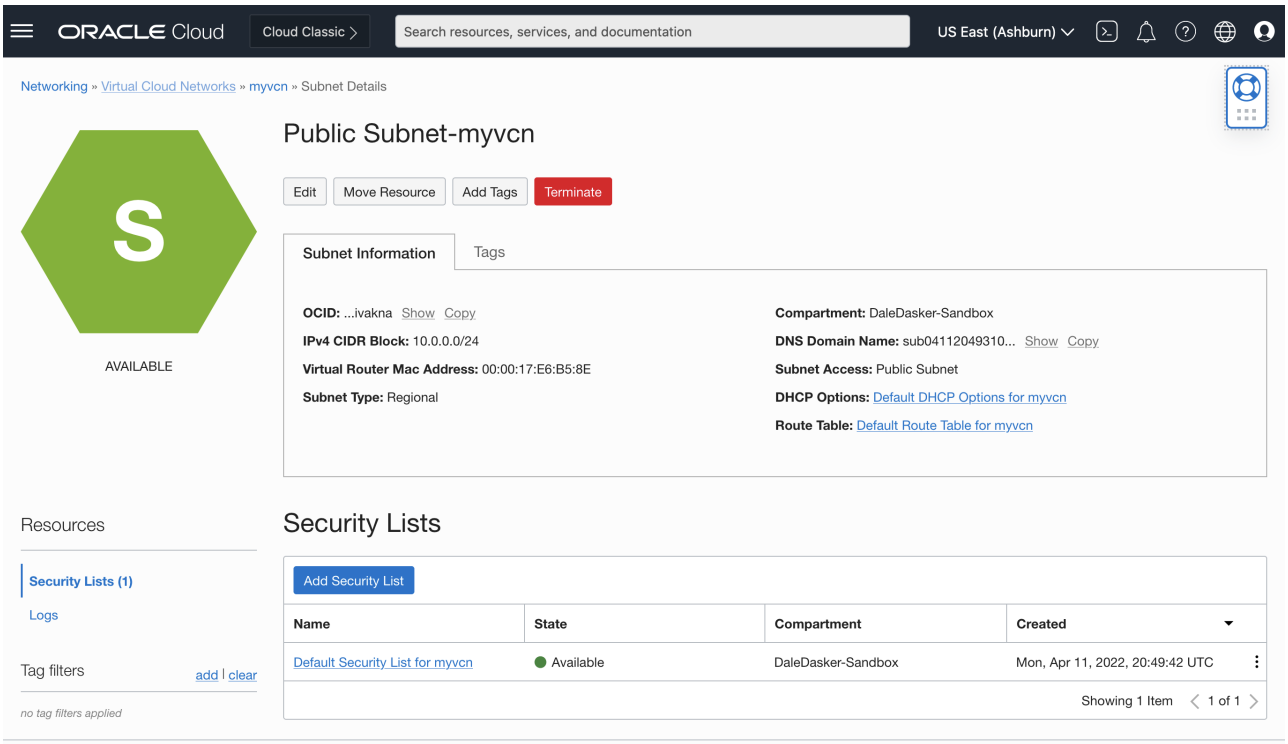


8. Click 'View Virtual Cloud Network' to display the created VCN

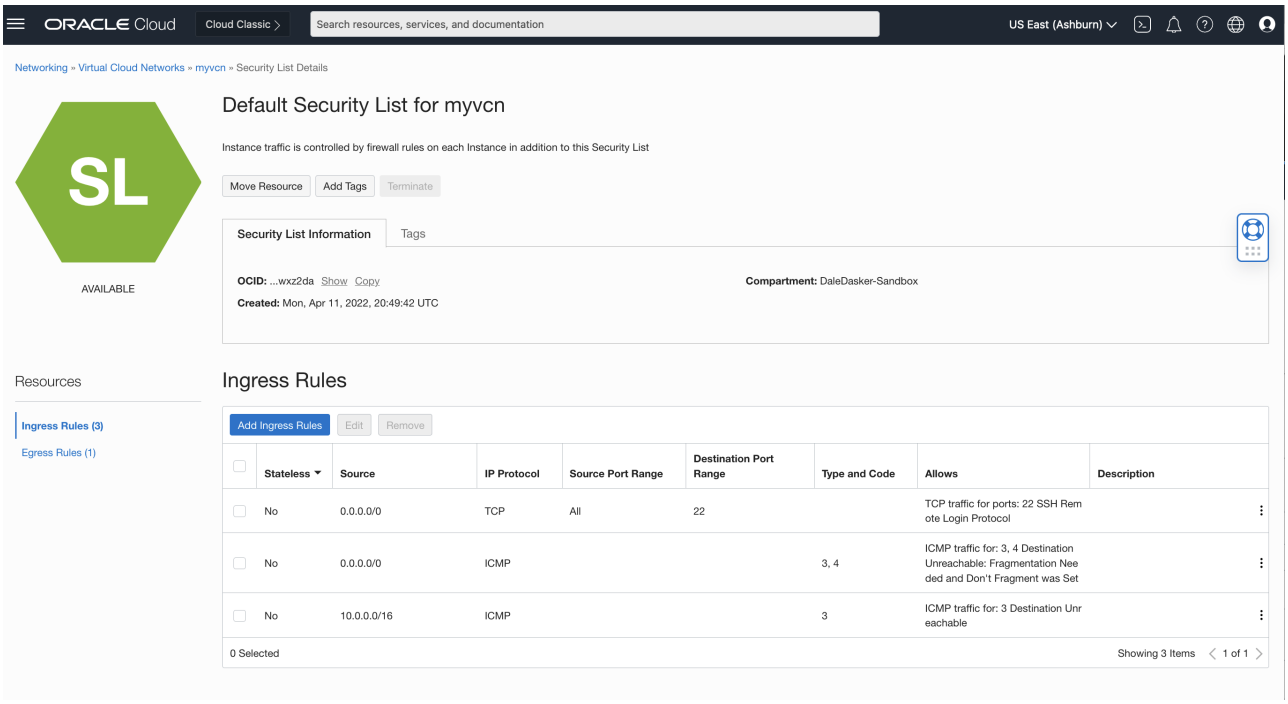


Task 2: Configure security list to allow MySQL incoming connections

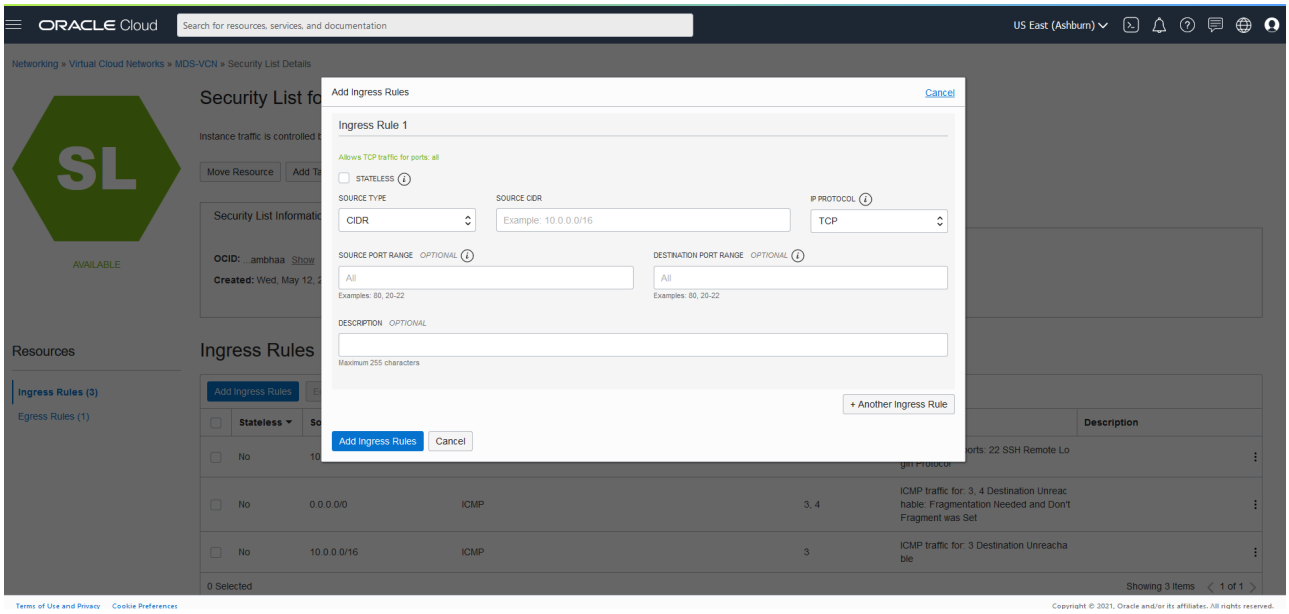
1. On myvcn page under 'Subnets in (root) Compartment', click '**Public Subnet-myvcn**'



2. On Public Subnet-myvcn page under 'Security Lists', click '**Security List for Public Subnet-myvcn**'

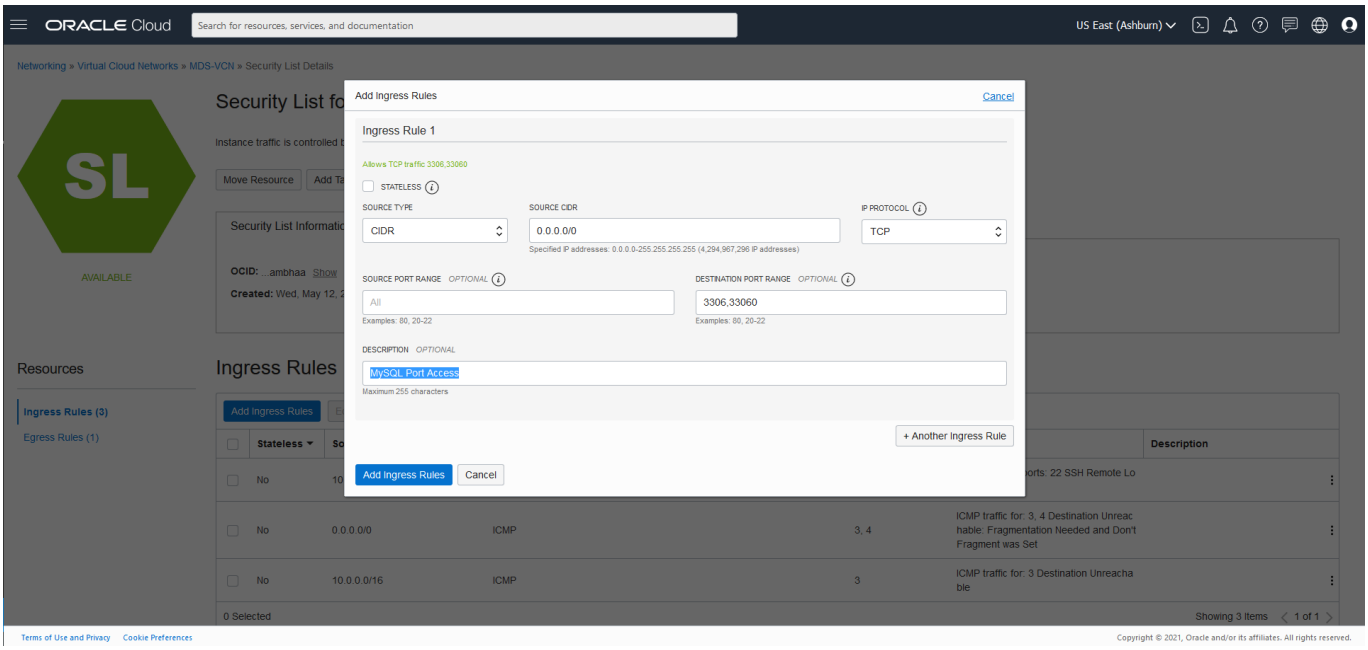


3. On Security List for Public Subnet-myvcn page under 'Ingress Rules', click '**Add Ingress Rules**'

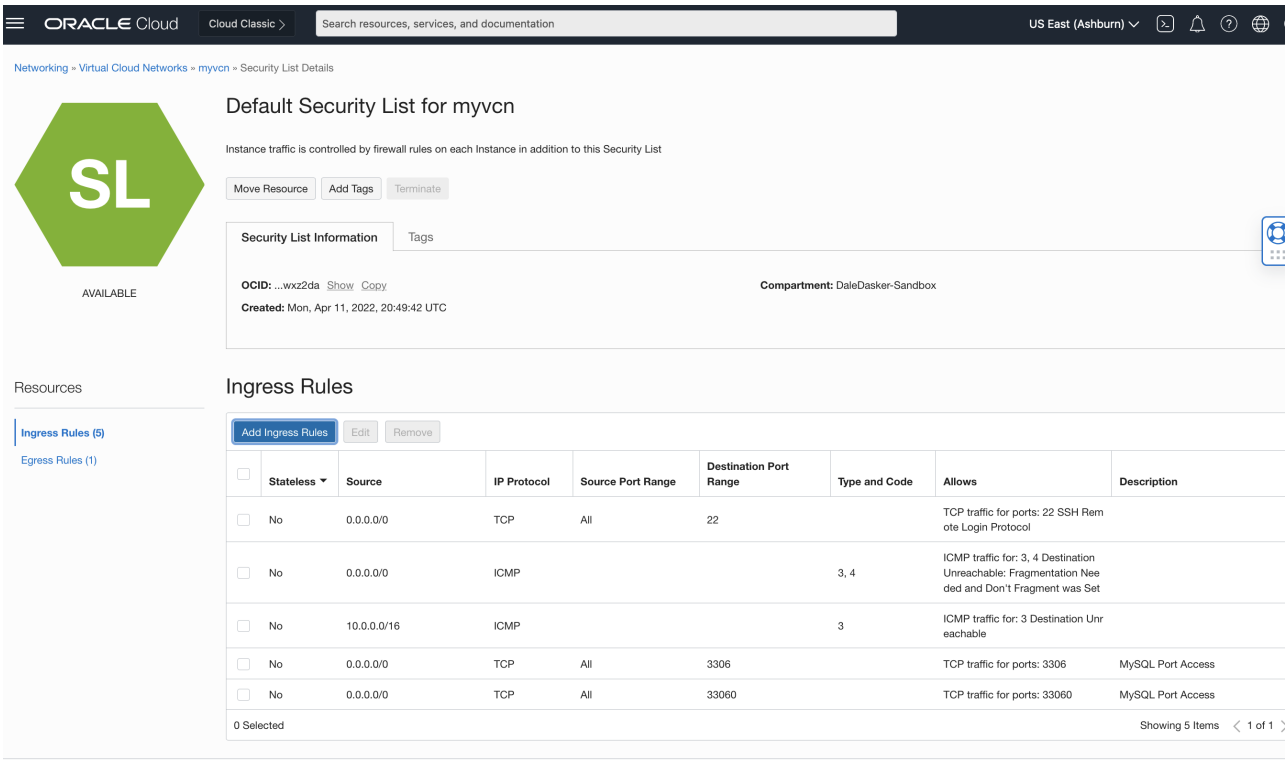


4. On Add Ingress Rules page under Ingress Rule 1

Add an Ingress Rule with Source CIDR `<copy>0.0.0.0/0</copy>` Destination Port Range `<copy>3306,33060</copy>` Description `<copy>MySQL Port Access</copy>` Click 'Add Ingress Rule'



5. On Security List for Public Subnet-myvcn page, the new Ingress Rules will be shown under the Ingress Rules List



You may now proceed to the next lab

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, April 2023

Create Linux Compute Instance

Introduction

Oracle Cloud Infrastructure Compute lets you provision and manage compute hosts, known as instances . You can create instances as needed to meet your compute and application requirements. After you create an instance, you can access it securely from your computer or cloud shell.

Create Linux Compute Instance

In this lab, you use Oracle Cloud Infrastructure to create an Oracle Linux instance.

Estimated Time: 20 minutes

Objectives

In this lab, you will be guided through the following tasks:

- Create SSH Key on OCI Cloud
- Create Compute Instance

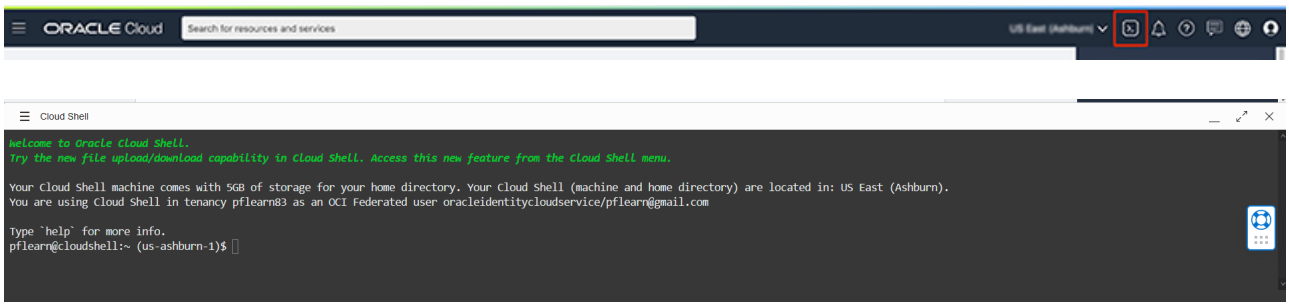
Prerequisites

- An Oracle Free Tier or Paid Cloud Account
- A web browser
- Should have completed Lab 1

Task 1: Create SSH Key on OCI Cloud Shell

The Cloud Shell machine is a small virtual machine running a Bash shell which you access through the Oracle Cloud Console (Homepage). You will start the Cloud Shell and generate a SSH Key to use for the Bastion session.

1. To start the Oracle Cloud shell, go to your Cloud console and click the cloud shell icon at the top right of the page. This will open the Cloud Shell in the browser, the first time it takes some time to generate it.



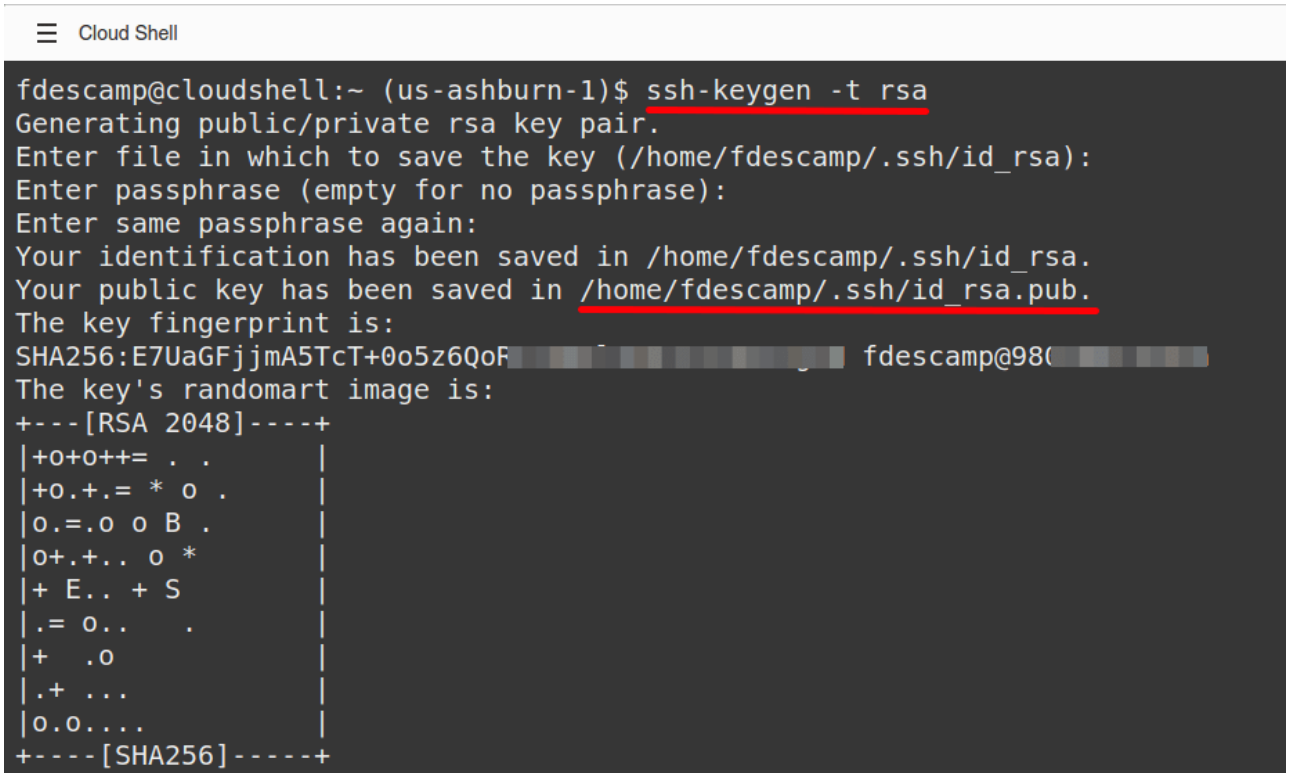
Note: You can use the icons in the upper right corner of the Cloud Shell window to minimize, maximize, restart, and close your Cloud Shell session.

2. Once the cloud shell has started, create the SSH Key using the following command:

```
<copy>ssh-keygen -t rsa</copy>
```

Press enter for each question.

Here is what it should look like.



3. The public and private SSH keys are stored in ~/.ssh/id_rsa.pub.

4. Examine the two files that you just created.

```
<copy>cd .ssh</copy>
```

```
<copy>ls</copy>
```

Note: in the output there are two files, a *private key:* `id_rsa` and a *public key:* `id_rsa.pub`. Keep the private key safe and don't share its content with anyone. The public key will be needed for various activities and can be uploaded to certain systems as well as copied and pasted to facilitate secure communications in the cloud.

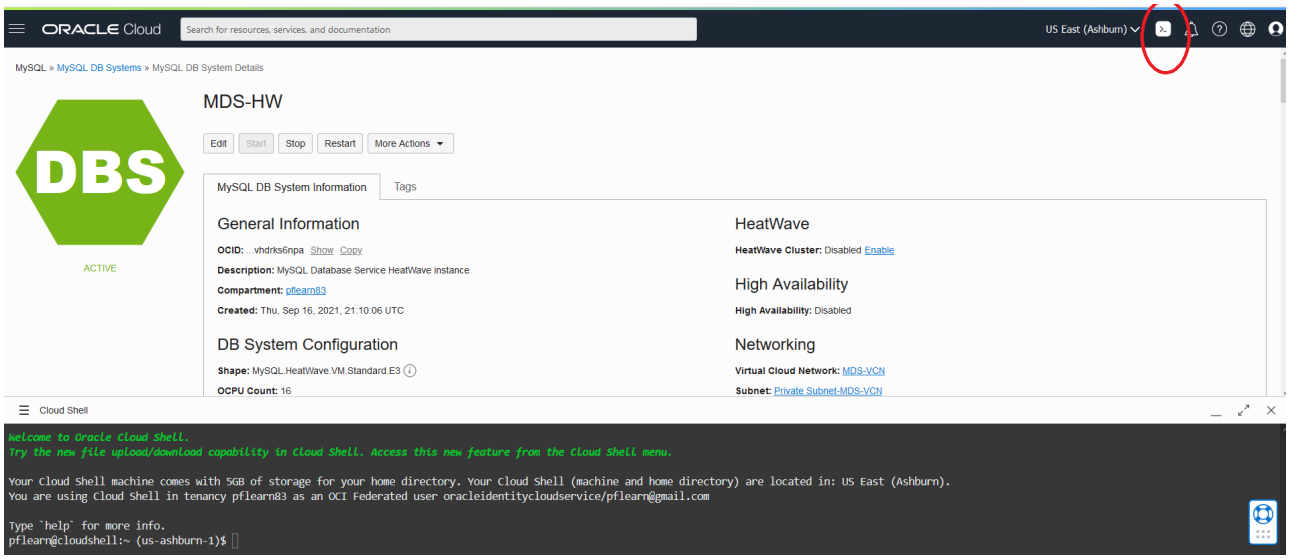
Task 2: Create Compute instance

You will need a compute Instance to connect to your brand new MySQL database.

1. Before creating the Compute instance open a notepad

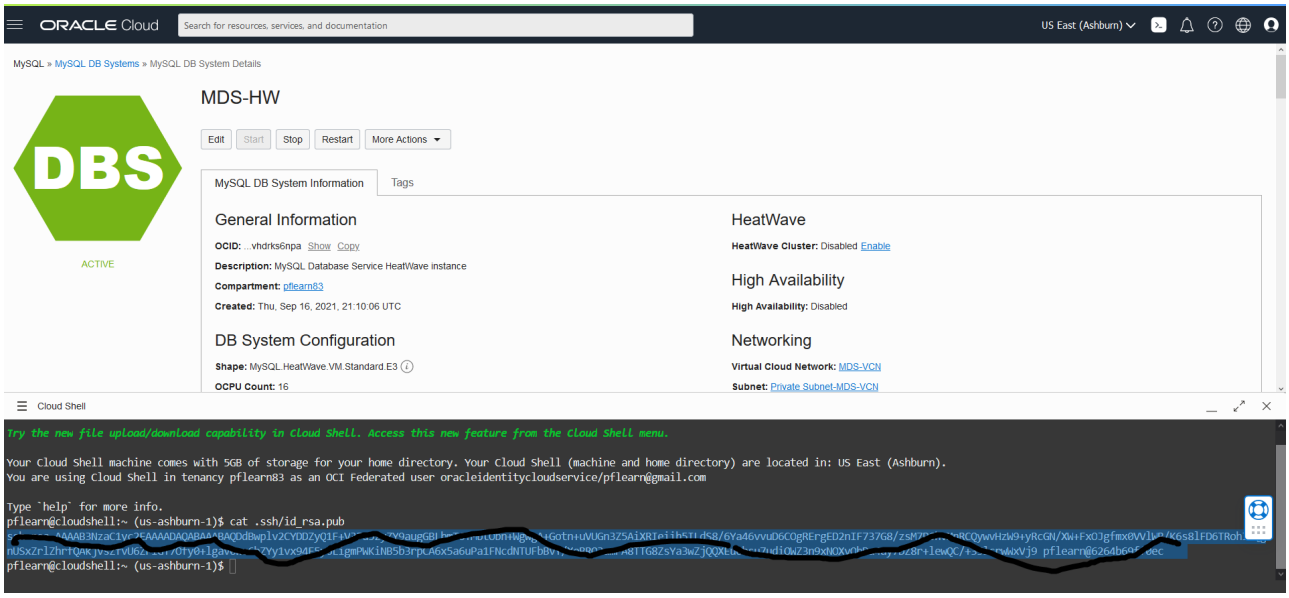
2. Do the followings steps to copy the public SSH key to the notepad

   Open the Cloud shell



   Enter the following command

```
<copy>cat ~/.ssh/id_rsa.pub</copy>
```



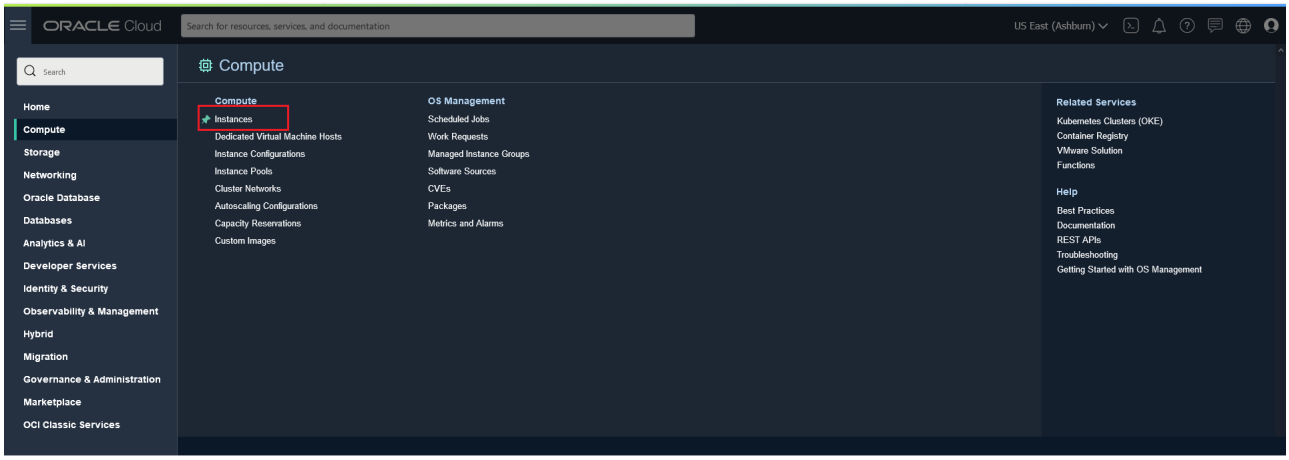
3. Copy the id_rsa.pub content the notepad

Your notepad should look like this

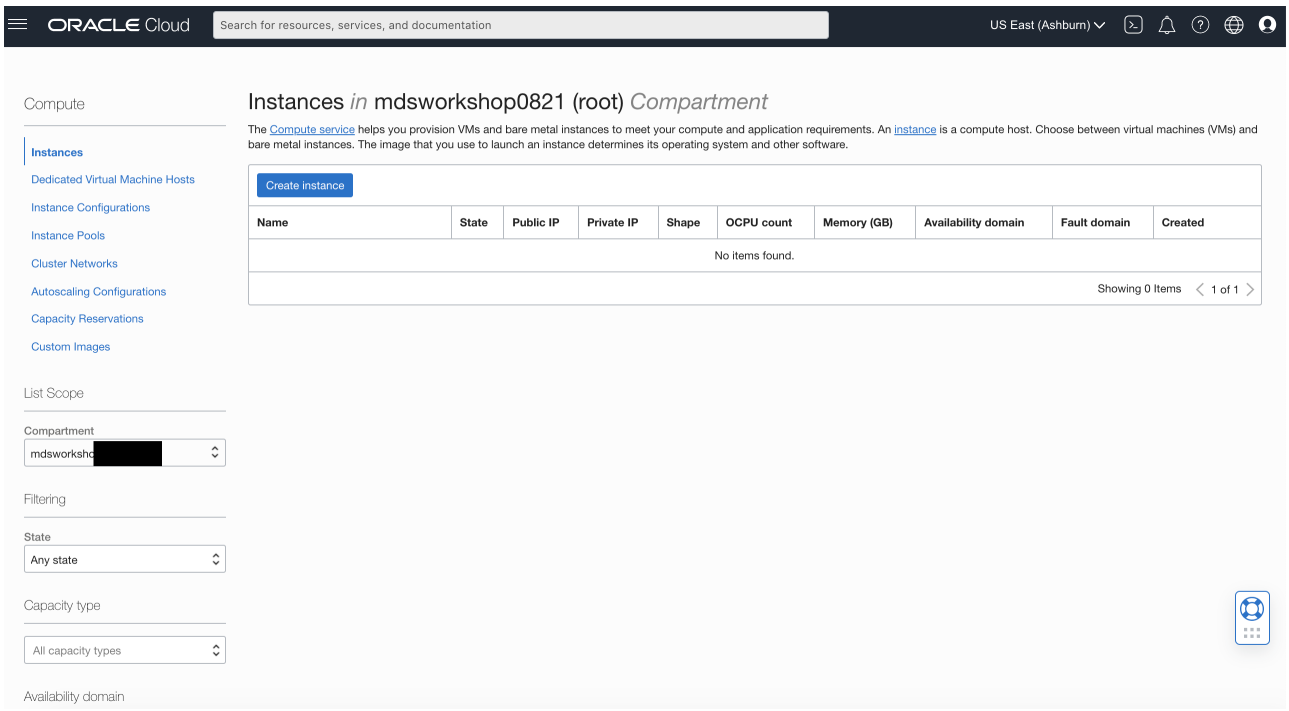
id-rsa.pub

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDdBwp1v2CYDDZyQ1F+V3Fu5zyZY9augGBLbmT6+MbtObn+WgwgA+Gotn
+uVUGn3Z5AiXRIeiih5ILdS8/6Ya46vvuD6COgRErgED2nIF737G8/zsM7P6hVMnRCQywvHzW9+yRcGN/XW
+FxOJgfmx0VVlWB/K6s81FD6TRohIhQgnUSxZrlZhrfQAkjvsZfvU6ZrIGT7Ofy0+lgav0wu6bZYy1vx94E5y6LigmPWKiNB5b3rpCA6x
5a6uPa1FNcdNTUFbBvY/XoBRQ3amPA8TTG8ZsYa3wZjQQXEu6bsu7udiQWZ3n9xNOXvObDa4ay7DZ8r+lewQC/+351crwWxVj9
pflearn@6264b69ff0ec
```

4. To launch a Linux Compute instance, go to Navigation Menu Compute Instances



5. On Instances in **(root)** Compartment, click **Create Instance**



6. On Create Compute Instance

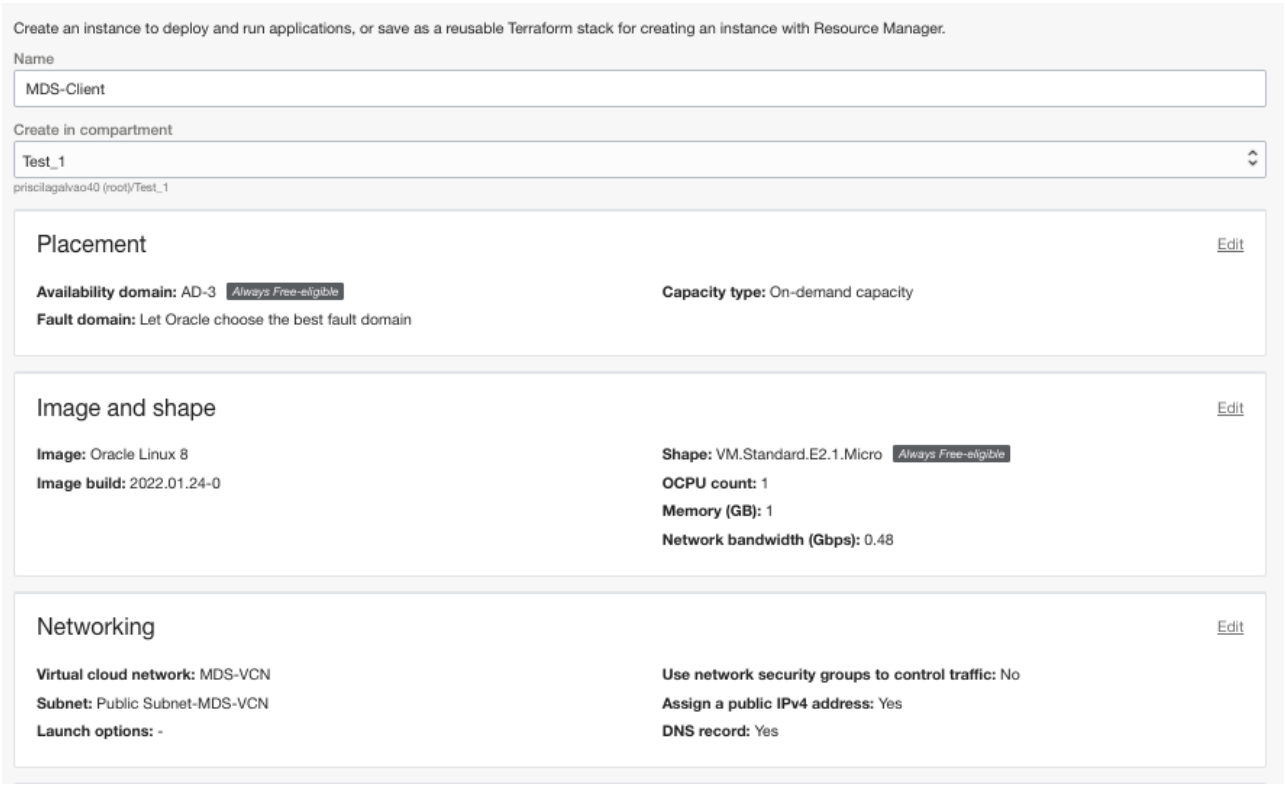
Enter Name `<copy>myclient</copy>`

7. Make sure **(root)** compartment is selected

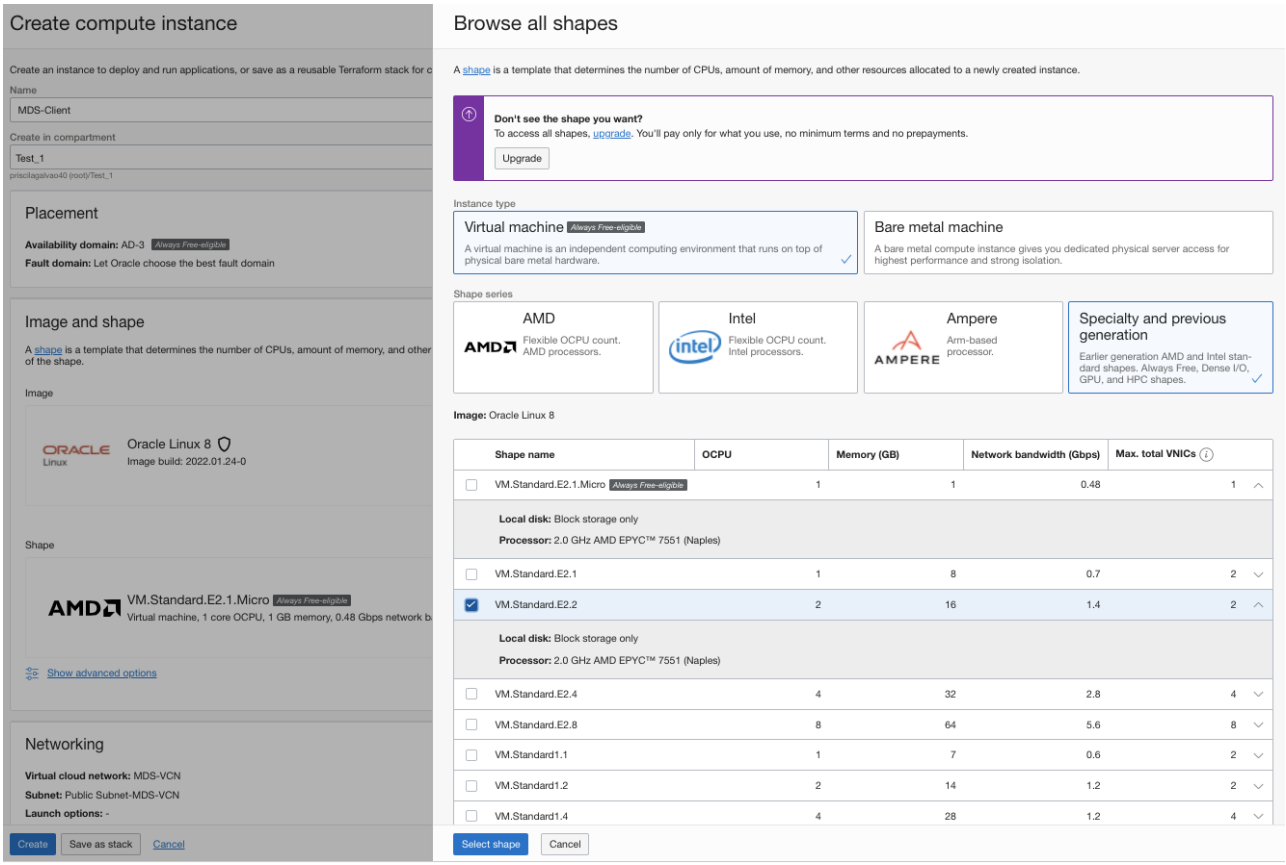
8. On Placement, keep the selected Availability Domain

9. On Image and Shape click the **Edit** link

    - On Image: Keep the selected Image, Oracle Linux 8

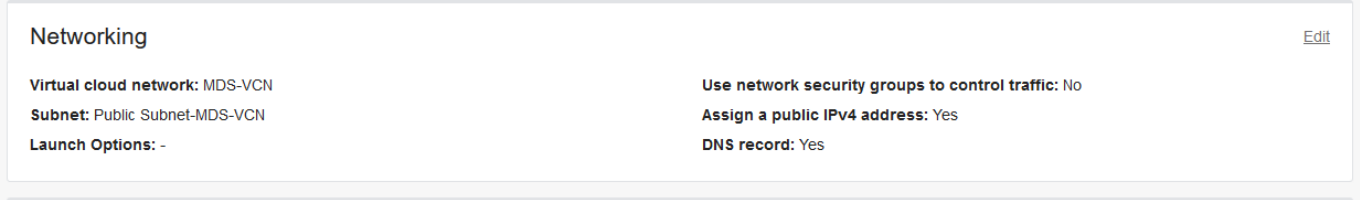


    - On Shape - Click the **change shape** button
    - Select Instance Shape: VM.Standard.E2.2

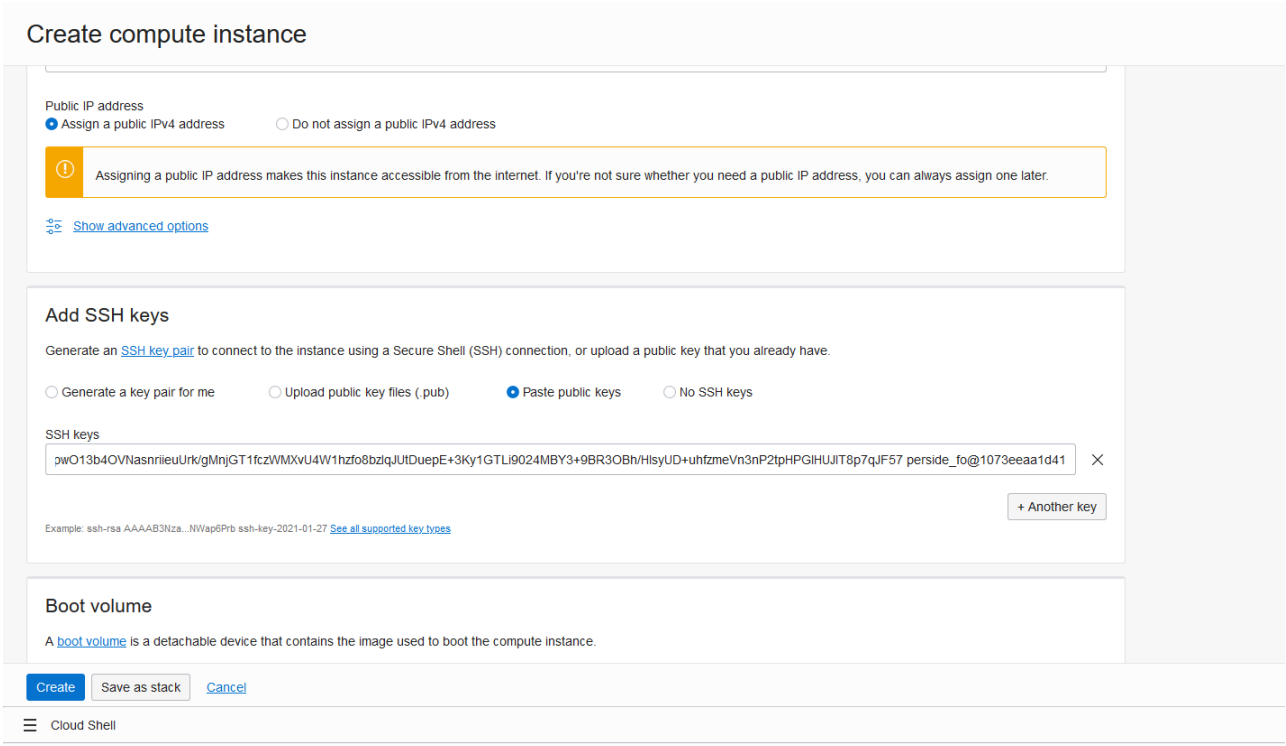


10. On Networking, make sure '**myvcn**' is selected

'Assign a public IP address' should be set to Yes

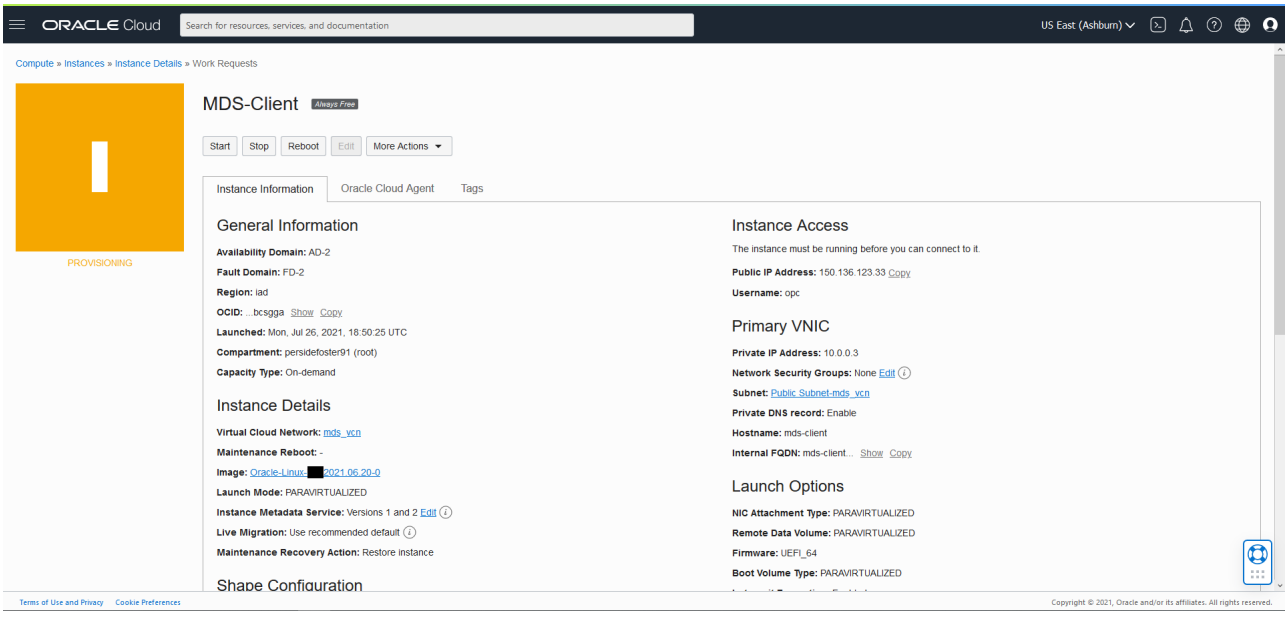


11. On Add SSH keys, paste the public key from the notepad.

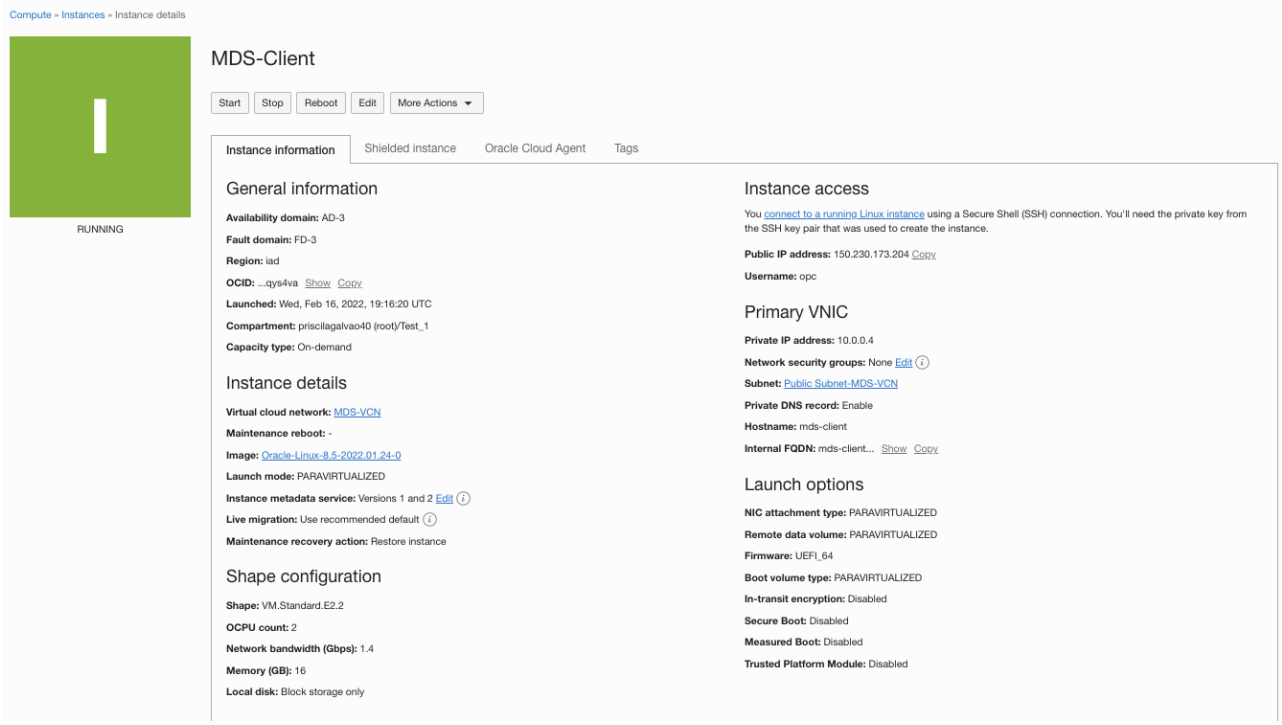


12. Click '**Create**' to finish creating your Compute Instance.

13. The New Virtual Machine will be ready to use after a few minutes. The state will be shown as 'Provisioning' during the creation



14. The state 'Running' indicates that the Virtual Machine is ready to use.

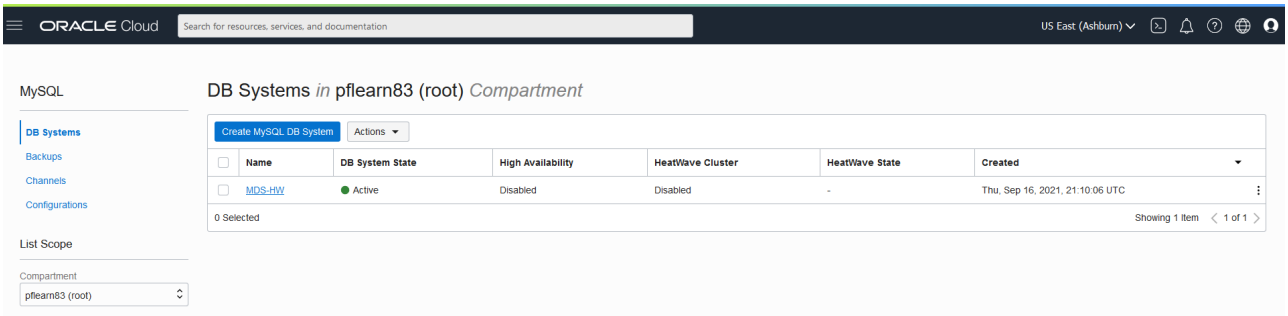


Task 3:Connect to Compute Instance with SSH Key

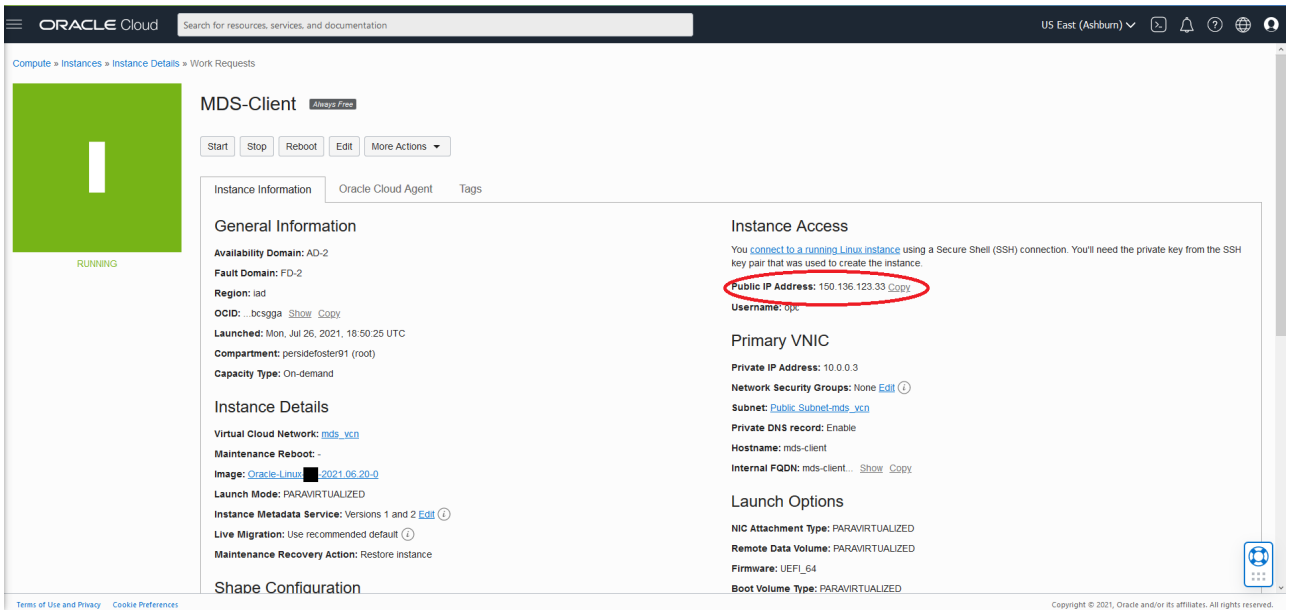
To connect to **myclient** you will need to properly setup your SSH command. Do the following steps:

1. Copy the public IP address of the active Compute Instance to a notepad

   a. Go to Navigation Menu Compute Instances



   b. Click the `myclient` Compute Instance link



c. Copy `myclient` plus the `Public IP Address` to the notepad

2. Indicate the location of the private key you created earlier with **myclient**.

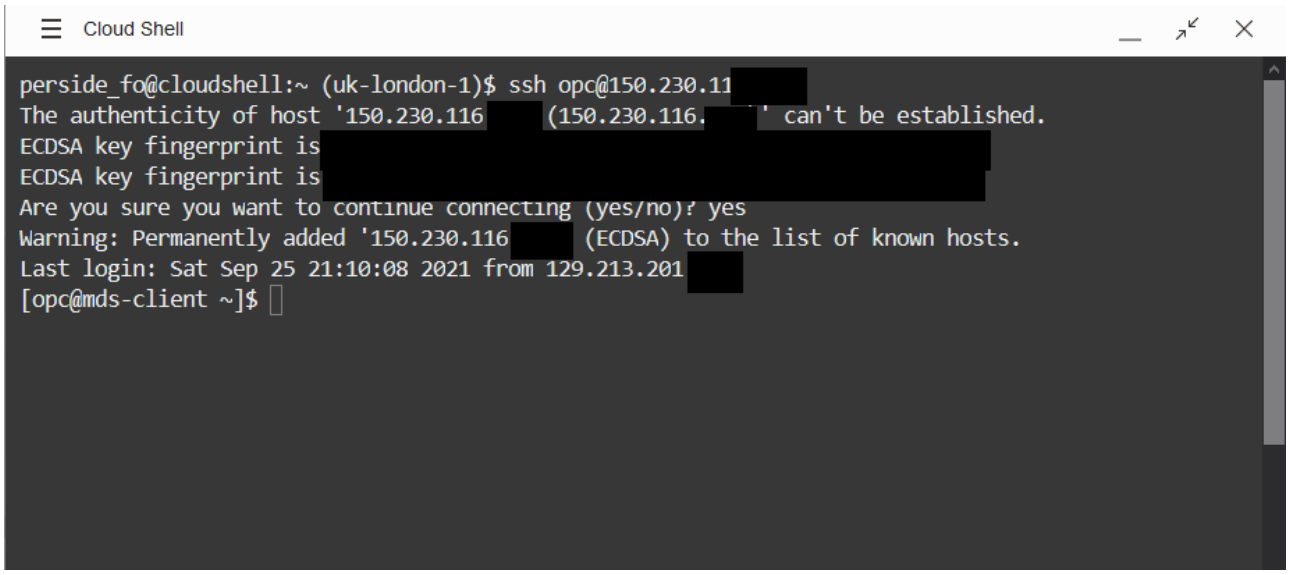
   Enter the username **opc** and the Public **IP Address**.

   Note: The **myclient** instance shows the Public IP Address as mentioned on TASK 5: #11

   (Your SSH login command should look like this:

   ssh -i ~/.ssh/id_rsa opc@132.145.170...)

```
<copy>ssh -i ~/.ssh/id_rsa opc@<your_compute_instance_ip></copy>
```



** You are ready to install MySQL on the Compute Instance**

You may now proceed to the next lab

Acknowledgements

SETUP

Environment Setup

Objective: Connect Personal Computer to the Oracle Network and the Oracle Cloud Infrastructure (OCI)

In this lab you will Download lab materials, plus connect your Personal Computer to the Oracle Network and the Oracle Cloud Infrastructure (OCI)

Estimated Lab Time: -- 10 minutes

Objectives

In this lab, you will:

- Download lab materials
- Setup SSH client
- Record Server information

Prerequisites

In compliance with Oracle security policies, I acknowledge I will not load actual confidential customer data or Personally Identifiable Information (PII) into my demo environment

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed

Task 1: Download Lab Material and SSH client

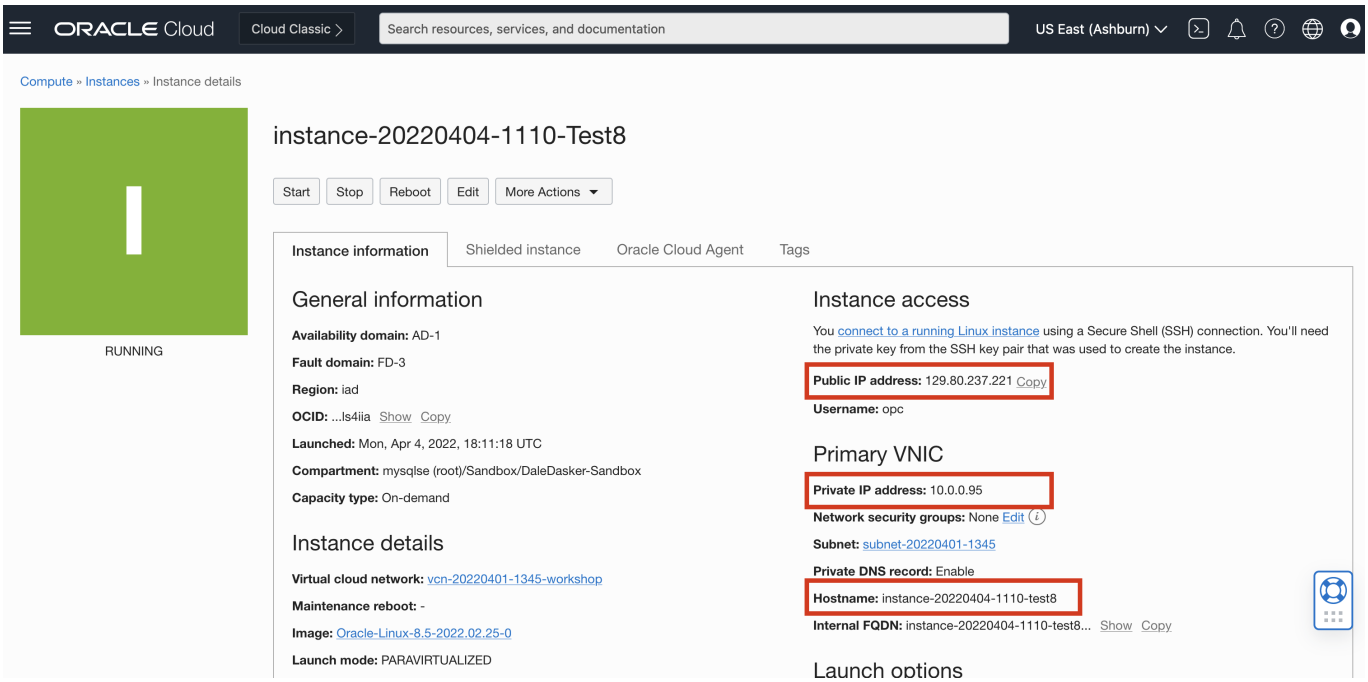
1. presentation
2. SSH keys to connect labs (it's the same key in two different formats). These keys should have been created when you were creating your Compute Instance.
    - id_rsa in native openssl format. Use it with Workbench
    - id_rsa.ppk in putty format for windows. Use it only with putty
3. If you have not yet installed an SSH client on your laptop, please install one e.g. (windows) https://www.putty.org/

Task 2: Record Lab Server info on Notepad

student###-Server:

- Hostname:
- Hostname FQDN:
- Public IP: (e.g. 130.61.56.195)
- Private IP: (e.g. 10.0.11.18)

Example:



Task 3: Review Misc Lab Information

1. Document standard

   - When in the manual you read **shell>** the command must be executed in the Operating System shell.
   - When in the manual you read **mysql>** the command must be executed in a client like MySQL, MySQL Shell, MySQL Workbench, etc. We recommend students to use MySQL Shell to practice with it.
   - When in the manual you read MySQL **mysqlsh>** the command must be executed in MySQL Shell.

2. Lab standard

   - shell> the command must be executed in the Operating System shell
   - mysql> the command must be executed in a client like MySQL, MySQL Workbench
   - mysqlsh> the command must be executed in MySQL shell

3. The software used for the labs is located on a local /workshop folder within each server.

4. Tip: set the keep alive for SSH connection to 60 seconds, to keep session open during lectures

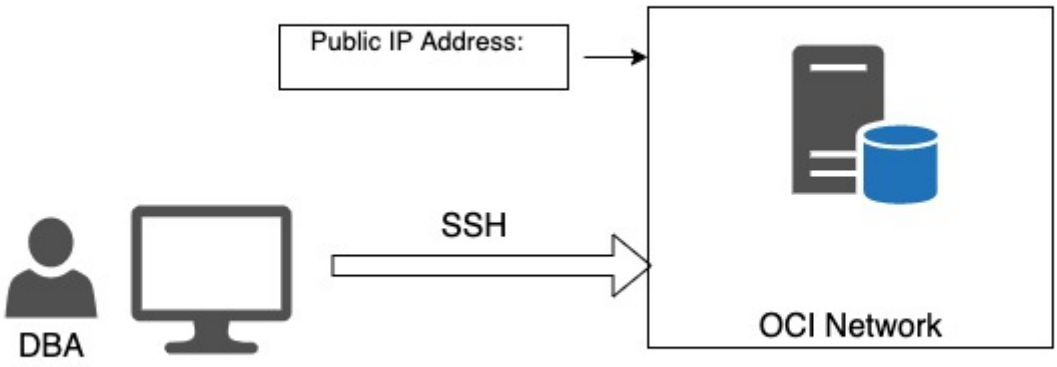
5. Linux ***opc*** user has limited privileges. To work with administrative privileges, use "sudo" like shell> sudo su - root

Task 4: Setup Lab Server and Connection

1. Server description **ServerA** will be used to run the full Workshop on. You will:

   - Install MySQL Enterprise Edition 8.0.
   - Install a MySQL Shell as a command line interface for MySQL Enterprise Edition.
   - Install the Sample Employees Database

2. Sever Connections example:



3. Test the connection to your Linux machines from your laptop using these parameters
    - a. SSH connection
    - b. SSH key file named “id_rsa” or "
    - c. username “opc”
    - d. no password
    - e. Public IP address of your assigned Linux VM (serverA, serverB)
4. Examples of connections:

Linux: use “id_rsa” key file

shell> `<copy>ssh –i id_rsa opc@public_ip </copy>`

Task 5: Setup workshop directory on Server

1. SSH to Server

    shell>

    ```
    <copy>ssh –i id_rsa opc@public_ip </copy>
    ```

2. Make /workshop Directory

    shell>

    ```
    <copy>mkdir workshop
    cd workshop</copy>
    ```

3. Wget workshop files

For the Ashburn Region shell>

```
<copy>wget  https://objectstorage.us-ashburn-
1.oraclecloud.com/p/WTu4TLn77_zTRNpHh2471dSS_AYojuhpKydcRsSE6S75EojezJ
_oLT6H0X7tB1AQ/n/idazzjlcjqzj/b/bucket-20220901-1608-
workshop_RPM/o/workshop32.zip;</copy>
```

4. Extract workshop files

shell>

```
<copy>unzip  workshop32.zip </copy>
```

Learn More

- Creating SSH Keys
- Compute SSH Connections

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, April 2023

INSTALL - MYSQL ENTERPRISE EDITION

Introduction

Detailed Installation of MySQL Enterprise Edition 8.0 and MySQL Shell on Linux

Objective: RPM Installation of MySQL 8 Enterprise on Linux

RPM Installation of MySQL Enterprise 8 on Linux

Estimated Time: 15 minutes

Objectives

In this lab, you will:

- Install MySQL Enterprise Edition
- Start and test MySQL Enterpriese Edition Install
- Install MySQL Shell and Connect to MySQL Enterprise

Prerequisites

Test code This lab assumes you have:

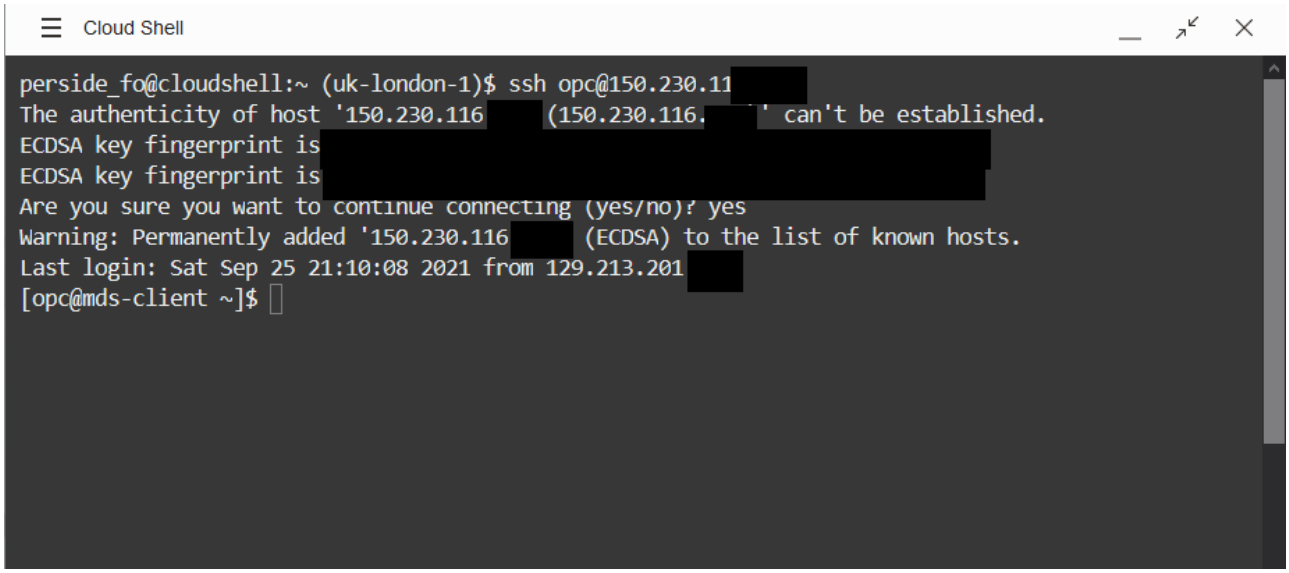
- An Oracle account
- All previous labs successfully completed
- Lab standard
  - shell> the command must be executed in the Operating System shell
  - mysql> the command must be executed in a client like MySQL, MySQL Workbench
  - mysqlsh> the command must be executed in MySQL shell

Task 1: Install MySQL Enterprise Edition using Linux RPM's

Note: If not already connected with SSH

- connect to **myclient** instance using Cloud Shell (**Example:** ssh -i ~/.ssh/id_rsa opc@132.145.17....)

```
<copy>ssh -i ~/.ssh/id_rsa opc@<your_compute_instance_ip></copy>
```



1. Install the RPM's

shell> <copy>cd ~/workshop</copy>

shell> <copy>sudo yum -y install *.rpm</copy>

Task 2: Start and test MySQL Enterprise Edition Install

1. Start your new mysql instance

shell> <copy>sudo systemctl start mysqld</copy>

2. Verify that process is running

shell> <copy>ps -ef | grep mysqld</copy>

shell> <copy>netstat -an | grep 3306</copy>

3. Another way is searching the message “ready for connections” in error log as one of the last

shell> <copy>sudo grep -i ready /var/log/mysqld.log</copy>

4. Retrieve root password for first login:

shell> <copy>sudo grep -i 'temporary password' /var/log/mysqld.log</copy>

5. Login to the the mysql-enterprise installation and check the status (you will be asked to change password)

shell>

```
<copy>mysqlsh --uri root@localhost:3306 --sql -p </copy>
```

6. Create New Password for MySQL Root

mysqlsh> <copy>ALTER USER 'root'@'localhost' IDENTIFIED BY 'Welcome1!';</copy>

mysqlsh> `<copy>\status</copy>`

7. Create a new administrative user called 'admin' with remote access and full privileges

mysqlsh> `<copy>CREATE USER 'admin'@'%' IDENTIFIED BY 'Welcome1!'; GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%' WITH GRANT OPTION;</copy>`

mysqlsh> `<copy>\quit</copy>`

Learn More

- MySQL Linux Installation
- MySQL Shell Installation

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering
- **Last Updated By/Date** - <Dale Dasker, April 2023

DEPLOYING - InnoDB ReplicaSets

Introduction

InnoDB ReplicaSets Objective: deploying MySQL sandboxes and then creating an InnoDB ReplicaSet

*This lab walks you through creating MySQL Sandboxes, deploying InnoDB ReplicaSets, bootstrapping MySQL Router and testing failovers

Estimated Time: 15 minutes

Objectives

In this lab, you will do the followings:

- Connect to MySQL Shell
- Create MySQL Sandboxes
- Create InnoDB ReplicaSet

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed

Lab standard

- shell> the command must be executed in the Operating System shell
- mysql> the command must be executed in a client like MySQL, MySQL Workbench
- mysqlsh> the command must be executed in MySQL shell

Notes:

- Open a notepad file and your linux Private IP on student###-serverA
- serverA PRIVATE ip: (client_ip)

Task 1: Connect to mysql-enterprise on Server

1. Connect to your MySQL Shell

   shell>

   ```
   <copy>mysqlsh</copy>
   ```

   mysqlsh>

```
<copy>\option --persist history.autoSave 1</copy>
```

2. Create 3 MySQL Sandboxes

   a. **mysqlsh>**

```
<copy>dba.deploySandboxInstance(3310, {password: "password"})</copy>
```

   b. **mysqlsh>**

```
<copy>dba.deploySandboxInstance(3320, {password: "password"})</copy>
```

   c. **mysqlsh>**

```
<copy>dba.deploySandboxInstance(3330, {password: "password"})</copy>
```

Task 2: Create ReplicaSet

1. Using the MySQL Shell Connection, connect the Shell to Sandbox on Port 3310 and create ReplicaSet

   a. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

   b. **mysqlsh>**

```
<copy>var rs = dba.createReplicaSet("example")</copy>
```

   c. **mysqlsh>**

```
<copy>rs.status()</copy>
```

2. Add 2 instances to ReplicaSet

   a. **mysqlsh>**

```
<copy>rs.addInstance('root@localhost:3320')</copy>
```

b. **mysqlsh>**

```
<copy>rs.addInstance('root@localhost:3330')</copy>
```

c. **mysqlsh>**

```
<copy>rs.status()</copy>
```

Task 3: Test failovers

1. Test changing the Primary. This is good for instances where you want to safely failover to a new Replica

   a. Failover to 3320 instance

   mysqlsh>

   ```
   <copy>rs.setPrimaryInstance('root@localhost:3320')</copy>
   ```

   b. Check status

   mysqlsh>

   ```
   <copy>rs.status()</copy>
   ```

   c. Failover back to 3310 instance

   mysqlsh>

   ```
   <copy>rs.setPrimaryInstance('root@localhost:3310')</copy>
   ```

   d. Check status (**Note** You can see extended details by passing the {extended: [1|2} })

   mysqlsh>

   ```
   <copy>rs.status()</copy>
   ```

Task 4: Deploy MySQL Router

1. Create a new SSH Shell window to your Compute Instance and create a directory for MySQL Router configuration and data

shell>

```
<copy>mkdir ~/mysqlrouter
cd ~/mysqlrouter</copy>
```

2. Bootstrap MySQL Router and Deploy Router against 3310 Instance (Which is now the Source)

shell>

```
<copy>mysqlrouter --bootstrap root@localhost:3310 -d
/home/opc/mysqlrouter</copy>
```

shell>

```
<copy>./start.sh &</copy>
```

shell>

```
<copy>ps -ef | grep mysqlrouter</copy>
```

shell>

```
<copy>mysql -P6446 --protocol=tcp -uroot -ppassword</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

3. Failover the Source and check if the Router follows

mysqlsh>

```
<copy>rs.setPrimaryInstance('root@localhost:3320')</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

4. Kill the Source and force failover

mysqlsh>

```
<copy>dba.stopSandboxInstance(3320, {password: "password"})</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

mysqlsh>

```
<copy>shell.connect('root@localhost:3310')</copy>
```

mysqlsh>

```
<copy>rs = dba.getReplicaSet()</copy>
```

mysqlsh>

```
<copy>rs.forcePrimaryInstance()</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

Task 5: Clean up environment

1. Using the MySQL Shell interface, remove the Sandboxes

mysqlsh>

```
<copy>dba.stopSandboxInstance(3310, {password: "password"})</copy>
```

mysqlsh>

```
<copy>dba.stopSandboxInstance(3330, {password: "password"})</copy>
```

mysqlsh>

```
<copy>dba.deleteSandboxInstance(3310)</copy>
```

mysqlsh>

```
<copy>dba.deleteSandboxInstance(3320)</copy>
```

mysqlsh>

```
<copy>dba.deleteSandboxInstance(3330)</copy>
```

2. Stop MySQL Router and remove the files

shell>

```
<copy>./stop.sh</copy>
```

shell>

```
<copy>rm -Rdf ./*</copy>
```

Learn More

- CREATE USER
- MySQL Access Control Lists

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering

DEPLOYING - InnoDB Cluster

Introduction

InnoDB Cluster: Objective: deploying MySQL sandboxes and then creating an InnoDB Cluster

*This lab walks you through creating MySQL Sandboxes, deploying InnoDB Cluster, bootstrapping MySQL Router and testing failovers

Estimated Time: 15 minutes

Objectives

In this lab, you will do the followings:

- Connect to MySQL Shell
- Create MySQL Sandboxes
- Create InnoDB Cluster

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed

MySQL Instances ports

- "Portland": 3310, 3320, 3330

Lab standard

- shell> the command must be executed in the Operating System shell
- mysql> the command must be executed in a client like MySQL, MySQL Workbench
- mysqlsh> the command must be executed in MySQL shell

Notes:

- Open a notepad file and your linux Private IP on student###-serverA
- serverA PRIVATE ip: (client_ip)

Task 1: Connect to mysql-enterprise on Server

1. Connect to your MySQL Shell

   shell>

   ```
   <copy>mysqlsh</copy>
   ```

2. Create 3 MySQL Sandboxes

a. **mysqlsh>**

```
<copy>dba.deploySandboxInstance(3310, {password: "password"})
dba.deploySandboxInstance(3320, {password: "password"})
dba.deploySandboxInstance(3330, {password: "password"})</copy>
```

b. **mysqlsh>**

```
<copy>\quit</copy>
```

e. Load some sample data

shell>

```
<copy>mysql -P3310 --protocol=tcp -uroot -ppassword -e"CREATE DATABASE
world"</copy>
```

shell>

```
<copy>mysql -P3310 --protocol=tcp -uroot -ppassword world <
world_innodb.sql</copy>
```

Task 2: Create InnoDB Cluster

1. Connect to your MySQL Shell

shell>

```
<copy>mysqlsh</copy>
```

2. Using the MySQL Shell Connection, connect the Shell to Sandbox on Port 3310 and create InnoDB Cluster

a. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

b. **mysqlsh>**

```
<copy>var PortlandCluster = dba.createCluster("PortlandCluster")
</copy>
```

c. **mysqlsh>**

```
<copy>PortlandCluster.status()</copy>
```

3. Add 2 instances to InnoDB Cluster

a. **mysqlsh>**

```
<copy>PortlandCluster.addInstance('root@localhost:3320')</copy>
```

b. **mysqlsh>**

```
<copy>PortlandCluster.addInstance('root@localhost:3330')</copy>
```

c. **mysqlsh>**

```
<copy>PortlandCluster.status()</copy>
```

d. **mysqlsh>**

```
<copy>\connect root@localhost:3320</copy>
```

e. **mysqlsh>**

```
<copy>\sql</copy>
```

f. **mysqlsh>**

```
<copy>SHOW DATABASES;</copy>
```

g. **mysqlsh>**

```
<copy>USE world;</copy>
```

h. **mysqlsh>**

```
<copy>SHOW TABLES;</copy>
```

i. **mysqlsh>**

```
<copy>\js</copy>
```

j. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

Task 3: Test failovers

1. Test changing the Primary. This is good for situations where you want to safely failover to a new Replica

   a. Failover to 3320 instance

   mysqlsh>

   ```
   <copy>PortlandCluster.setPrimaryInstance("root@localhost:3320")</copy>
   ```

   b. Check status

   mysqlsh>

   ```
   <copy>PortlandCluster.status()</copy>
   ```

   c. Failover back to 3310 instance

   mysqlsh>

   ```
   <copy>PortlandCluster.setPrimaryInstance("root@localhost:3310")</copy>
   ```

   d. Check status (**Note** You can see extended details by passing the {extended: [1|2} })

   mysqlsh>

```
<copy>PortlandCluster.status()</copy>
```

Task 4: Deploy MySQL Router

1. Create a new SSH Shell window to your Compute Instance and create a directory for MySQL Router configuration and data

shell>

```
<copy>cd ~/mysqlrouter</copy>
```

2. Bootstrap MySQL Router and Deploy Router against 3310 Instance (Which is now the Source)

shell>

```
<copy>mysqlrouter --bootstrap root@localhost:3310 -d
/home/opc/mysqlrouter</copy>
```

shell>

```
<copy>./start.sh &</copy>
```

shell>

```
<copy>ps -ef | grep mysqlrouter</copy>
```

shell>

```
<copy>mysql -P6446 --protocol=tcp -uroot -ppassword</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

3. Failover the Source and check if the Router follows

mysqlsh>

```
<copy>PortlandCluster.setPrimaryInstance('root@localhost:3320')</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

4. Kill the Source and force failover

mysqlsh>

```
<copy>dba.stopSandboxInstance(3320, {password: "password"})</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

5. Restart the Secondary (3320)

mysqlsh>

```
<copy>dba.startSandboxInstance(3320)</copy>
```

mysqlsh>

```
<copy>PortlandCluster.status()</copy>
```

Task 5: Clean up environment

1. Stop MySQL Router and remove the files

shell>

```
<copy>./stop.sh</copy>
```

shell>

```
<copy>rm -Rdf ./*</copy>
```

Learn More

- CREATE USER
- MySQL Access Control Lists

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering

DEPLOYING - InnoDB ClusterSets

Introduction

InnoDB ClusterSets: Objective: deploying MySQL sandboxes and then creating an InnoDB ClusterSets

*This lab walks you through creating MySQL Sandboxes, deploying InnoDB ClusterSets, bootstrapping MySQL Router and testing failovers

Estimated Time: 15 minutes

Objectives

In this lab, you will do the followings:

- Connect to MySQL Shell
- Create MySQL Sandboxes
- Create InnoDB ClusterSets

Prerequisites

This lab assumes you have:

- An Oracle account
- All previous labs successfully completed

MySQL Instances ports

- "Portland": 3310, 3320, 3330
- "Seattle": 3410, 3420, 3430
- Lab standard
  - shell> the command must be executed in the Operating System shell
  - mysql> the command must be executed in a client like MySQL, MySQL Workbench
  - mysqlsh> the command must be executed in MySQL shell

Notes:

- Open a notepad file and your linux Private IP on student###-serverA
- serverA PRIVATE ip: (client_ip)

Task 1: Connect to mysql-enterprise on Server

1. Connect to your MySQL Shell

   shell>

```
<copy>mysqlsh</copy>
```

2. Create 3 Additional MySQL Sandboxes

a. **mysqlsh>**

```
<copy>dba.deploySandboxInstance(3410, {password: "password"})
dba.deploySandboxInstance(3420, {password: "password"})
dba.deploySandboxInstance(3430, {password: "password"})</copy>
```

Task 2: Create an InnoDB ClusterSet

1. Using the MySQL Shell Connection, connect the Shell to Sandbox on Port 3310 and create ClusterSet starting with 3410 Instance

a. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

b. **mysqlsh>**

```
<copy>var PortlandCluster = dba.getCluster()</copy>
```

c. **mysqlsh>**

```
<copy>PortlandCluster.status()</copy>
```

d. **mysqlsh>**

```
<copy>NWClusterSet = PortlandCluster.createClusterSet("NWCluster")
</copy>
```

e. **mysqlsh>**

```
<copy>NWClusterSet.status()</copy>
```

f. **mysqlsh>**

```
<copy>SeattleCluster =
NWClusterSet.createReplicaCluster("127.0.0.1:3410","SeattleCluster")
</copy>
```

g. **mysqlsh>**

```
<copy>SeattleCluster.status()</copy>
```

h. **mysqlsh>**

```
<copy>NWClusterSet.status()</copy>
```

2. Add 2 instances to Secondary (Replica) InnoDB Cluster

a. **mysqlsh>**

```
<copy>SeattleCluster.addInstance('root@localhost:3420')</copy>
```

b. **mysqlsh>**

```
<copy>SeattleCluster.addInstance('root@localhost:3430')</copy>
```

c. **mysqlsh>**

```
<copy>SeattleCluster.status()</copy>
```

d. **mysqlsh>**

```
<copy>\connect root@localhost:3410</copy>
```

e. **mysqlsh>**

```
<copy>\sql</copy>
```

f. **mysqlsh>**

```
<copy>SHOW DATABASES;</copy>
```

g. **mysqlsh>**

```
<copy>USE world;</copy>
```

h. **mysqlsh>**

```
<copy>SHOW TABLES;</copy>
```

i. **mysqlsh>**

```
<copy>\js</copy>
```

j. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

Task 3: Test failovers

1. Test changing the Primary. This is good for instances where you want to safely failover to a new Replica

   a. Failover to 3320 instance

mysqlsh>

```
<copy>PortlandCluster.setPrimaryInstance("root@localhost:3320")</copy>
```

b. **mysqlsh>**

```
<copy>\connect root@localhost:3320</copy>
```

c. Check status

mysqlsh>

```
<copy>PortlandCluster.status()</copy>
```

d. **mysqlsh>**

```
<copy>NWClusterSet = dba.getClusterSet()</copy>
```

e. **mysqlsh>**

```
<copy>NWClusterSet.status()</copy>
```

f. Failover back to 3310 instance

mysqlsh>

```
<copy>PortlandCluster.setPrimaryInstance("root@localhost:3310")</copy>
```

g. **mysqlsh>**

```
<copy>\connect root@localhost:3310</copy>
```

h. Check status (**Note** You can see extended details by passing the {extended: [1|2} })

mysqlsh>

```
<copy>PortlandCluster.status()</copy>
```

i. **mysqlsh>**

```
<copy>NWClusterSet = dba.getClusterSet()</copy>
```

j. **mysqlsh>**

```
<copy>NWClusterSet.status()</copy>
```

Task 4: Deploy MySQL Router

1. Create a new SSH Shell window to your Compute Instance and create a directory for MySQL Router configuration and data

shell>

```
<copy>cd ~/mysqlrouter</copy>
```

2. Bootstrap MySQL Router and Deploy Router against 3310 Instance (Which is now the Source)

shell>

```
<copy>mysqlrouter --bootstrap root@localhost:3310 -d
/home/opc/mysqlrouter --name='Portland'</copy>
```

shell>

```
<copy>./start.sh &</copy>
```

shell>

```
<copy>ps -ef | grep mysqlrouter</copy>
```

shell>

```
<copy>mysql -P6446 --protocol=tcp -uroot -ppassword</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

mysqlsh>

```
<copy>NWClusterSet.listRouters()</copy>
```

mysqlsh>

```
<copy>NWClusterSet.routingOptions()</copy>
```

mysqlsh>

```
<copy>NWClusterSet.describe()</copy>
```

3. Failover the Source and check if the Router follows

mysqlsh>

```
<copy>PortlandCluster.setPrimaryInstance('root@localhost:3320')</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

mysqlsh>

```
<copy>PortlandCluster.setPrimaryInstance('root@localhost:3310')</copy>
```

4. Failover to the Replica Cluster

mysqlsh>

```
<copy>NWClusterSet.setPrimaryCluster('SeattleCluster')</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

mysqlsh>

```
<copy>\connect root@localhost:3410</copy>
```

mysqlsh>

```
<copy>NWClusterSet.status()</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

5. Fail back to Portland Cluster

mysqlsh>

```
<copy>NWClusterSet.setPrimaryCluster('PortlandCluster')</copy>
```

mysqlsh>

```
<copy>\connect root@localhost:3310</copy>
```

mysqlsh>

```
<copy>NWClusterSet.status()</copy>
```

mysql>

```
<copy>SELECT @@port;</copy>
```

Learn More

- CREATE USER
- MySQL Access Control Lists

Acknowledgements

- **Author** - Dale Dasker, MySQL Solution Engineering