

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)

The screenshot shows the OCI Dashboard with the following sections:

- Quickstarts:**
 - APPLICATION DEVELOPMENT: Deploy a low-code app on Autonomous Database using APEX (3-5 mins)
 - APPLICATION DEVELOPMENT: Deploy RStudio in a Container (10-12 mins)
 - APPLICATION DEVELOPMENT: Deploy a baseline landing zone (7-9 mins)
 - DATABASE: Visualize and Analyze Strava Data on Autonomous Database (2-4 mins)
 - APPLICATION DEVELOPMENT: Deploy a WordPress website (6-8 mins)
 - APPLICATION DEVELOPMENT: Deploy Apache Tomcat on Arm-based Ampere A1 (10-12 mins)
- Launch Resources:**
 - COMPUTE: Create a VM instance (2-6 mins)
 - AUTONOMOUS TRANSACTION PROCESSING: Create an ATP database (3-5 mins)
 - AUTONOMOUS DATA WAREHOUSE: Create an ADW database (3-5 mins)
 - NETWORKING: Set up a network with a wizard
 - RESOURCE MANAGER: Create a stack
 - OBJECT STORAGE: Store data
- Right Sidebar:**
 - All systems operational: View health dashboard
 - Cloud Advisor: View your cost savings
 - Account Center: User Management, Billing
 - Oracle Live: Running OCI workloads your way (Learn to run any application faster, Register now)
 - What's New: Financial Services for GPU environment is introduced (Mar 25, 2022), Database Migration is now available in the South Africa Central

Task 1: Create Virtual Cloud Network

1. Click Navigation Menu
Select Networking

Select Virtual Cloud Networks

The screenshot shows the Oracle Cloud interface with the 'Networking' service selected in the sidebar. The 'Virtual Cloud Networks' option is highlighted with a red box. The main content area displays various networking components like Overview, DNS Management, Customer Connectivity, IP Management, and Related Services.

2. Click Start VCN Wizard

The screenshot shows the 'Virtual Cloud Networks' list page. The 'Start VCN Wizard' button is circled in red. The page includes filters for compartment, state, and tag filters, along with columns for Name, State, IPv4 CIDR Block, IPv6 CIDR Block, Default Route Table, DNS Domain Name, and Created.

3. Select 'Create VCN with Internet Connectivity'

Click 'Start VCN Wizard'

Main Name	Created
oraclevcn.com	Sat, Jul 17, 2021, 16:18:54 UTC

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

Create a VCN with Internet Connectivity

Configuration

Important: Before starting:

- Limits:** Ensure your tenancy has not reached its VCN limit. See [Service Limits](#).
- Access:** Ensure you have permission to work in the compartment you select.

Basic Information

VCN NAME: MDS-VCN

COMPARTMENT: MDS-Sandbox

Configure VCN and Subnets

VCN CIDR BLOCK: 10.0.0.0/16

If you plan to peer this VCN with another VCN, the VCNs must not have overlapping CIDRs. [Learn more.](#)

PUBLIC SUBNET CIDR BLOCK: 10.0.0.0/24

The subnet CIDR blocks must not overlap.

Includes:

- VCN
- Public subnet
- Private subnet
- Internet gateway (IG)
- NAT gateway (NAT)
- Service gateway (SG)

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

Terms of Use and Privacy | Cookie Preferences

Copyright © 2021, Oracle and/or its affiliates. All rights reserved.

7. The Virtual Cloud Network creation is completing

Create a VCN with Internet Connectivity

Creating Resources

- Virtual Cloud Network creation complete (Done)
- Create Virtual Cloud Network (1 resolved) (Done)
- Create Subnets (2 resolved) (Done)
- Create Internet Gateway (1 resolved) (Done)
- Create NAT Gateway (1 resolved) (Done)
- Create Service Gateway (1 resolved) (Done)
- Create Route Table for Private Subnet (1 resolved) (Done)
- Create Security List for Private Subnet (1 resolved) (Done)
- Update Route Tables (2 resolved) (Done)

[View Virtual Cloud Network](#)

https://cloud.oracle.com/limits | [Cookie Preferences](#)

Copyright © 2021, Oracle and/or its affiliates. All rights reserved.

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

Cookie Preferences'. Copyright notice: 'Copyright © 2021, Oracle and/or its affiliates. All rights reserved.'"/>

MDS-VCN

[Move Resource](#) [Add Tags](#) [Terminate](#)

VCN Information

Compartment: MDS-Sandbox
Created: Wed, May 12, 2021, 19:04:13 UTC
IPv4 CIDR Block: 10.0.0.0/16
IPv6 CIDR Block: No Value
OCID: ...dbrl3a [Show](#) [Copy](#)
DNS Resolver: MDS-VCN
Default Route Table: Default Route Table for MDS-VCN
DNS Domain Name: mdsvcn.oraclevcn.com

Resources

Subnets (2)

Name	State	IPv4 CIDR Block	Subnet Access	Created
Private Subnet-MDS-VCN	Available	10.0.1.0/24	Private (Regional)	Wed, May 12, 2021, 19:04:15 UTC
Public Subnet-MDS-VCN	Available	10.0.0.0/24	Public (Regional)	Wed, May 12, 2021, 19:04:15 UTC

Showing 2 items < 1 of 1 >

Local Peering Gateways (0)

Terms of Use and Privacy | [Cookie Preferences](#)

Copyright © 2021, Oracle and/or its affiliates. All rights reserved.

Task 2: Configure security list to allow MySQL incoming connections

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

Public Subnet-myvcn

Subnet Information

- OCID: ...ivakna [Show](#) [Copy](#)
- IPv4 CIDR Block: 10.0.0.0/24
- Virtual Router Mac Address: 00:00:17:E6:B5:8E
- Subnet Type: Regional

Compartment: DaleDasker-Sandbox

DNS Domain Name: sub04112049310... [Show](#) [Copy](#)

Subnet Access: Public Subnet

DHCP Options: [Default DHCP Options for myvcn](#)

Route Table: [Default Route Table for myvcn](#)

Resources

Security Lists

Add Security List			
Name	State	Compartment	Created
Default Security List for myvcn	Available	DaleDasker-Sandbox	Mon, Apr 11, 2022, 20:49:42 UTC

Showing 1 Item < 1 of 1 >

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

Default Security List for myvcn

Instance traffic is controlled by firewall rules on each Instance in addition to this Security List

Security List Information

- OCID: ...wxz2da [Show](#) [Copy](#)
- Created: Mon, Apr 11, 2022, 20:49:42 UTC

Compartment: DaleDasker-Sandbox

Resources

Ingress Rules

Add Ingress Rules								
	Stateless	Source	IP Protocol	Source Port Range	Destination Port Range	Type and Code	Allows	Description
<input type="checkbox"/>	No	0.0.0.0/0	TCP	All	22			TCP traffic for ports: 22 SSH Remote Login Protocol
<input type="checkbox"/>	No	0.0.0.0/0	ICMP		3, 4			ICMP traffic for: 3, 4 Destination Unreachable: Fragmentation Needed and Don't Fragment was Set
<input type="checkbox"/>	No	10.0.0.0/16	ICMP		3			ICMP traffic for: 3 Destination Unreachable

0 Selected

Showing 3 Items < 1 of 1 >

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

The screenshot shows the Oracle Cloud interface for managing a security list. The main page displays a green hexagonal icon labeled 'SL' with the status 'AVAILABLE'. Below it, there are sections for 'Security List Information' (OCID: ..., Created: Wed, May 12, 2021) and 'Ingress Rules' (3 items listed). A modal window titled 'Add Ingress Rules' is open, showing the configuration for 'Ingress Rule 1'. The rule allows TCP traffic for ports: all, source CIDR: 10.0.0.0/16, IP protocol: TCP, and destination port range: All. An optional description 'MySQL Port Access' is provided. At the bottom of the modal is a blue 'Add Ingress Rules' button.

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'

This screenshot shows the 'Add Ingress Rules' dialog box with the following configurations for 'Ingress Rule 1':

- Allow:** TCP traffic 3306,33060
- SOURCE TYPE:** CIDR (0.0.0.0/0)
- IP PROTOCOL:** TCP
- SOURCE PORT RANGE:** All
- DESTINATION PORT RANGE:** 3306,33060
- DESCRIPTION:** MySQL Port Access

The 'Add Ingress Rules' button at the bottom of the dialog is highlighted in blue.

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

Default Security List for myvcn

Instance traffic is controlled by firewall rules on each instance in addition to this Security List

[Move Resource](#) [Add Tags](#) [Terminate](#)

[Security List Information](#) [Tags](#)

OCID: ...wxz2da [Show](#) [Copy](#) **Compartment:** DaleDasker-Sandbox

Created: Mon, Apr 11, 2022, 20:49:42 UTC

Ingress Rules

<input type="checkbox"/>	Stateless	Source	IP Protocol	Source Port Range	Destination Port Range	Type and Code	Allows	Description
<input type="checkbox"/>	No	0.0.0.0/0	TCP	All	22			TCP traffic for ports: 22 SSH Remote Login Protocol
<input type="checkbox"/>	No	0.0.0.0/0	ICMP			3, 4		ICMP traffic for: 3, 4 Destination Unreachable: Fragmentation Needed and Don't Fragment was Set
<input type="checkbox"/>	No	10.0.0.0/16	ICMP			3		ICMP traffic for: 3 Destination Unreachable
<input type="checkbox"/>	No	0.0.0.0/0	TCP	All	3306			TCP traffic for ports: 3306 MySQL Port Access
<input type="checkbox"/>	No	0.0.0.0/0	TCP	All	33060			TCP traffic for ports: 33060 MySQL Port Access

0 Selected Showing 5 Items < 1 of 1 >

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

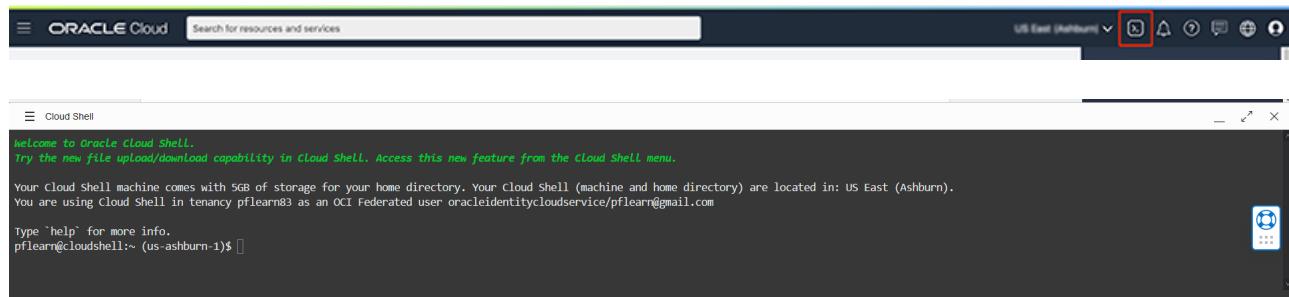
Prerequisites

- 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.

☰ Cloud Shell

```
fdescamp@cloudshell:~ (us-ashburn-1)$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/fdescamp/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/fdescamp/.ssh/id_rsa.
Your public key has been saved in /home/fdescamp/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:E7UaGFjjmA5TcT+0o5z6QoF [REDACTED] fdescamp@98[REDACTED]
The key's randomart image is:
+---[RSA 2048]---+
|+o+o++= . .
|+o.+.= * o .
|o.=.o o B .
|o+.+.. o *
|+ E.. + S
|.= o... .
|+ .o
|.+ ...
|o.o.....
+---[SHA256]---
```

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>
```

```
pflearn@cloudshell:~ (us-ashburn-1)$ cd .ssh
pflearn@cloudshell:~ (us-ashburn-1)$ ls
id_rsa id_rsa.pub
pflearn@cloudshell:~ (us-ashburn-1)$ ]
```

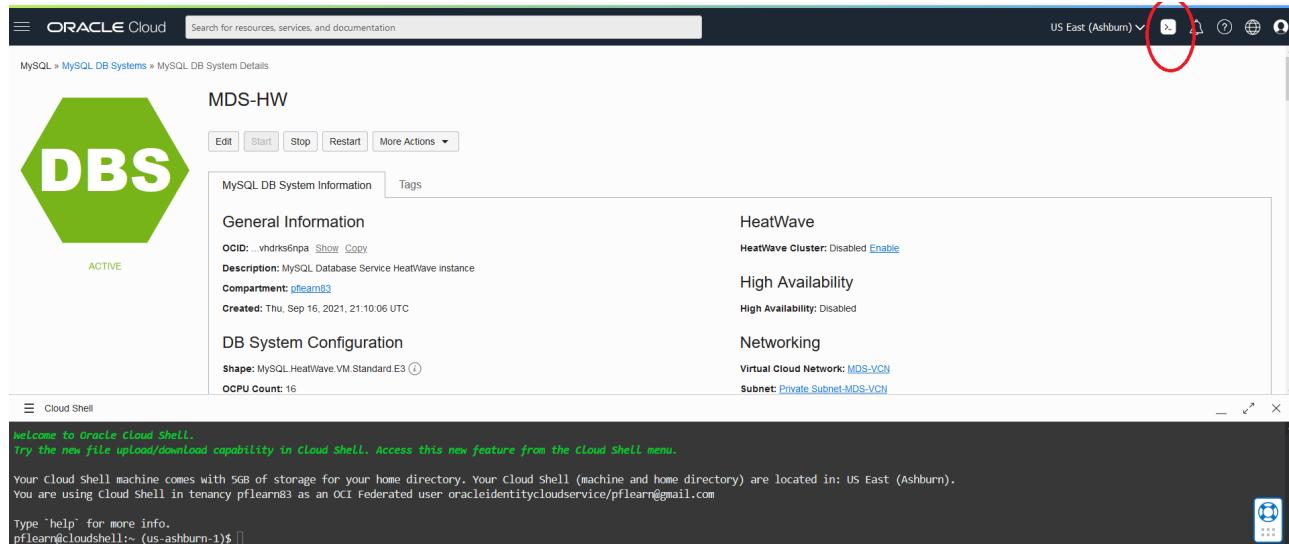
Note: in the output there are two files, a *private key*: `id_rsa` and a *public key*: `id_rsa.pub`. Keep the private key safe and don't share its content with anyone. The public key will be needed for various activities and can be uploaded to certain systems as well as copied and pasted to facilitate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

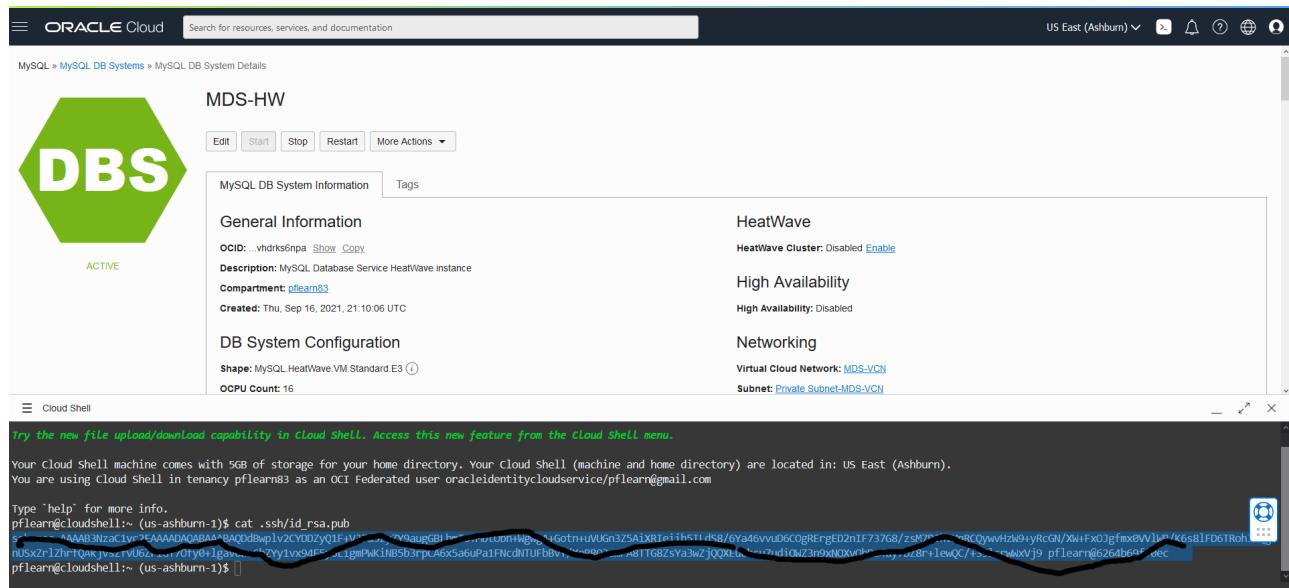
Open the Cloud shell



The screenshot shows the Oracle Cloud MySQL DB System Details page for a database named "MDS-HW". The "General Information" section includes the OCID: vhdrrks6npa, Description: MySQL Database Service HeatWave instance, Compartment: pfearn83, and Created: Thu, Sep 16, 2021, 21:10:06 UTC. The "DB System Configuration" section shows the Shape: MySQL HeatWave VM Standard E3 and OCPU Count: 16. The "HeatWave" section indicates it is disabled. The "High Availability" section shows it is also disabled. The "Networking" section lists the Virtual Cloud Network: MDS-VCN and Subnet: Private Subnet-MDS-VCN. At the bottom, there is a "Cloud Shell" section with a welcome message and a command prompt pflearn@cloudshell:~ (us-ashburn-1)\$.

Enter the following command

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



The screenshot shows the Oracle Cloud MySQL DB System Details page for a database named "MDS-HW". The "General Information" section includes the OCID: vhdrrks6npa, Description: MySQL Database Service HeatWave instance, Compartment: pfearn83, and Created: Thu, Sep 16, 2021, 21:10:06 UTC. The "DB System Configuration" section shows the Shape: MySQL HeatWave VM Standard E3 and OCPU Count: 16. The "HeatWave" section indicates it is disabled. The "High Availability" section shows it is also disabled. The "Networking" section lists the Virtual Cloud Network: MDS-VCN and Subnet: Private Subnet-MDS-VCN. At the bottom, there is a "Cloud Shell" section with a welcome message and a command prompt pflearn@cloudshell:~ (us-ashburn-1)\$ followed by the output of the cat command, which is a long string of characters.

3. Copy the id_rsa.pub content the notepad

Your notepad should look like this

id-rsa.pub

```
ssh-rsa AAAAAB3NzaC1yc2EAAAQABAAQDdBwpIv2CYDDZyQ1F+V3Fu5zyZY9augGBLbmT6+Mbt0bn+WwgA+Gotn
+uVUGn3Z5AiXRIeiih5ILDs8/6Ya46vvuD6COgRErgED2nIF737G8/zsM7P6hVMnRCQywvHzW9+yRcGN/XW
+Fx0Jgfmx0V1WB/K6s81FD6TRohIhQgnUSxZr1ZhrgQAkjvsZfvU6ZrIGT70fy0+1gav0wu6bZYy1vx94E5y6LigmPKiNB5b3rpCA6x
5a6uPa1FNcdNTUFbBvY/XoBRQ3amPA8TTG8zsYa3wZjQQXEu6bsu7udiQWZ3n9xNOXv0bDa4ay7DZ8r+1ewQC/+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**

Name	State	Public IP	Private IP	Shape	OCPU count	Memory (GB)	Availability domain	Fault domain	Created
No items found.									

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

Create an instance to deploy and run applications, or save as a reusable Terraform stack for creating an instance with Resource Manager.

Name: MDS-Client

Create in compartment: Test_1

priscilagalvao40 (root)/Test_1

Placement

Availability domain: AD-3 *Always Free-eligible*

Fault domain: Let Oracle choose the best fault domain

Image and shape

Image: Oracle Linux 8

Image build: 2022.01.24-0

Shape: VM.Standard.E2.1.Micro *Always Free-eligible*

OCPUs: 1

Memory (GB): 1

Network bandwidth (Gbps): 0.48

Networking

Virtual cloud network: MDS-VCN

Subnet: Public Subnet-MDS-VCN

Launch options: -

Use network security groups to control traffic: No

Assign a public IPv4 address: Yes

DNS record: Yes

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

Create compute instance

Create an instance to deploy and run applications, or save as a reusable Terraform stack for creating an instance with Resource Manager.

Name: MDS-Client

Create in compartment: Test_1

priscilagalvao40 (root)/Test_1

Placement

Availability domain: AD-3 *Always Free-eligible*

Fault domain: Let Oracle choose the best fault domain

Image and shape

A **shape** is a template that determines the number of CPUs, amount of memory, and other resources allocated to a newly created instance.

Image: Oracle Linux 8

Shape: VM.Standard.E2.1.Micro *Always Free-eligible*

Show advanced options

Networking

Virtual cloud network: MDS-VCN

Subnet: Public Subnet-MDS-VCN

Launch options: -

Browse all shapes

Don't see the shape you want? To access all shapes, [upgrade](#). You'll pay only for what you use, no minimum terms and no prepayments.

Upgrade

Instance type

Virtual machine *Always Free-eligible*

A virtual machine is an independent computing environment that runs on top of physical bare metal hardware.

Bare metal machine

A bare metal compute instance gives you dedicated physical server access for highest performance and strong isolation.

Shape series

AMD	Intel	Ampere	Specialty and previous generation
AMD processors.	Intel processors.	Ampere processor.	Earlier generation AMD and Intel standard shapes, Always Free, Dense I/O, GPU, and HPC shapes.

Image: Oracle Linux 8

Shape name	OCPUs	Memory (GB)	Network bandwidth (Gbps)	Max. total VNICs
VM.Standard.E2.1.Micro <i>Always Free-eligible</i>	1	1	0.48	1
VM.Standard.E2.2	2	16	1.4	2
VM.Standard.E2.4	4	32	2.8	4
VM.Standard.E2.8	8	64	5.6	8
VM.Standard1.1	1	7	0.6	2
VM.Standard1.2	2	14	1.2	2
VM.Standard1.4	4	28	1.2	4

Select shape | **Cancel**

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

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

Networking

Virtual cloud network: MDS-VCN

Subnet: Public Subnet-MDS-VCN

Launch Options: -

Use network security groups to control traffic: No

Assign a public IPv4 address: Yes

DNS record: Yes

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

Create compute instance

Public IP address

Assign a public IPv4 address Do not assign a public IPv4 address

Note: Assigning a public IP address makes this instance accessible from the internet. If you're not sure whether you need a public IP address, you can always assign one later.

[Show advanced options](#)

Add SSH keys

Generate an [SSH key pair](#) to connect to the instance using a Secure Shell (SSH) connection, or upload a public key that you already have.

Generate a key pair for me Upload public key files (.pub) Paste public keys No SSH keys

SSH keys

pwO13b4OVasnriieuUrk/gMnjGT1fczWMXvU4W1hzfo8bzlqJUJtDuepE+3Ky1GTLi9024MBY3+9BR3OBh/HlsyUD+uhfzmeVn3nP2tpHPGIHUJIT8p7qJF57 perside_fo@1073eea1d41

+ Another key

Example: ssh-rsa AAAAB3Nza...NWap6Prb ssh-key-2021-01-27 [See all supported key types](#)

Boot volume

A [boot volume](#) is a detachable device that contains the image used to boot the compute instance.

Create **Save as stack** **Cancel**

Cloud Shell

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

MDS-Client [Always Free](#)

Start **Stop** **Reboot** **Edit** **More Actions ▾**

Instance Information Oracle Cloud Agent Tags

General Information

- Availability Domain: AD-2
- Fault Domain: FD-2
- Region: iad
- OCID: ...bcsgga [Show](#) [Copy](#)
- Launched: Mon, Jul 26, 2021, 18:50:25 UTC
- Compartment: persideforster91 (root)
- Capacity Type: On-demand

Instance Details

- Virtual Cloud Network: [mds_vcn](#)
- Maintenance Reboot: -
- Image: [Oracle-Linux-2021.06.20.0](#)
- Launch Mode: PARAVIRTUALIZED
- Instance Metadata Service: Versions 1 and 2 [Edit](#)
- Live Migration: Use recommended default
- Maintenance Recovery Action: Restore Instance

Shape Configuration

Instance Access

The instance must be running before you can connect to it.

- Public IP Address: 10.0.0.3 [COPY](#)
- Username: opc

Primary VNIC

- Private IP Address: 10.0.0.3
- Network Security Groups: None [Edit](#)
- Subnet: [Public Subnet-mds_vcn](#)
- Private DNS record: Enable
- Hostname: mds-client
- Internal FQDN: mds-client [Show](#) [Copy](#)

Launch Options

- NIC Attachment Type: PARAVIRTUALIZED
- Remote Data Volume: PARAVIRTUALIZED
- Firmware: UEF:64
- Boot Volume Type: PARAVIRTUALIZED

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

MDS-Client

Instance information

- Shielded instance
- Oracle Cloud Agent
- Tags

General information

- Availability domain: AD-3
- Fault domain: FD-3
- Region:iad
- OCID: ...qys4va [Show](#) [Copy](#)
- Launched: Wed, Feb 16, 2022, 19:16:20 UTC
- Compartment: priscilagalvao40 (root)/Test_1
- Capacity type: On-demand

Instance details

- Virtual cloud network: [MDS-VCN](#)
- Maintenance reboot: -
- Image: [Oracle-Linux-8.5-2022.01-24-0](#)
- Launch mode: PARAVIRTUALIZED
- Instance metadata service: Versions 1 and 2 [Edit](#)
- Live migration: Use recommended default
- Maintenance recovery action: Restore instance

Shape configuration

- Shape: VM.Standard.E2.2
- OCPUs: 2
- Network bandwidth (Gbps): 1.4
- Memory (GB): 16
- Local disk: Block storage only

Instance access

You connect to a running Linux instance using a Secure Shell (SSH) connection. You'll need the private key from the SSH key pair that was used to create the instance.

- Public IP address: 150.230.173.204 [Copy](#)
- Username: opc

Primary VNIC

- Private IP address: 10.0.0.4
- Network security groups: None [Edit](#)
- Subnet: [Public Subnet-MDS-VCN](#)
- Private DNS record: Enable
- Hostname: mds-client
- Internal FQDN: mds-client... [Show](#) [Copy](#)

Launch options

- NIC attachment type: PARAVIRTUALIZED
- Remote data volume: PARAVIRTUALIZED
- Firmware: UEFI_64
- Boot volume type: PARAVIRTUALIZED
- In-transit encryption: Disabled
- Secure Boot: Disabled
- Measured Boot: Disabled
- Trusted Platform Module: Disabled

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

MySQL

DB Systems in pfelearn83 (root) Compartment

Create MySQL DB System Actions ▾						
<input type="checkbox"/>	Name	DB System State	High Availability	HeatWave Cluster	HeatWave State	Created
<input type="checkbox"/>	MDS-HW	Active	Disabled	Disabled	-	Thu, Sep 16, 2021, 21:10:06 UTC

Showing 1 item < 1 of 1 >

b. Click the **myclient** Compute Instance link

The screenshot shows the Oracle Cloud Instance Details page for an instance named 'MDS-Client'. The instance status is 'RUNNING'. In the 'Instance Access' section, the 'Public IP Address' field is highlighted with a red circle, showing the value '150.196.123.33'. Other fields in this section include 'Username: opc' and 'Private IP Address: 10.0.0.3'. The page also displays sections for 'General Information', 'Instance Details', and 'Shape Configuration'.

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>
```

The screenshot shows a terminal window titled 'Cloud Shell'. The user is attempting to log in via SSH with the command 'ssh -i ~/.ssh/id_rsa opc@150.196.123.33'. The terminal displays the following output:

```
perside_fo@cloudshell:~ (uk-london-1)$ ssh opc@150.196.123.33
The authenticity of host '150.196.123.33' (150.196.123.33) can't be established.
ECDSA key fingerprint is
ECDSA key fingerprint is
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '150.196.123.33' (ECDSA) to the list of known hosts.
Last login: Sat Sep 25 21:10:08 2021 from 129.213.201
[opc@mds-client ~]$
```

** 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:

The screenshot shows the Oracle Cloud Instance details page for 'instance-20220404-1110-Test8'. The instance status is 'RUNNING'. Key configuration details are highlighted with red boxes:

- General information:** Availability domain: AD-1, Fault domain: FD-3, Region: iad, OCID: ...ls4ia, Launched: Mon, Apr 4, 2022, 18:11:18 UTC, Compartment: mysqlse (root)/Sandbox/DaleDasker-Sandbox, Capacity type: On-demand.
- Instance access:** Public IP address: 129.80.237.221 (highlighted), Username: opc.
- Primary VNIC:** Private IP address: 10.0.0.95 (highlighted), Network security groups: None, Subnet: subnet-20220401-1345, Private DNS record: Enable, Hostname: instance-20220404-1110-test8 (highlighted), Internal FQDN: instance-20220404-1110-test8... (highlighted).
- Launch options:** Launch mode: PARAVIRTUALIZED.

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

- Green** shell> the command must be executed in the Operating System shell
- Blue** mysql> the command must be executed in a client like MySQL, MySQL Workbench
- Orange** 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

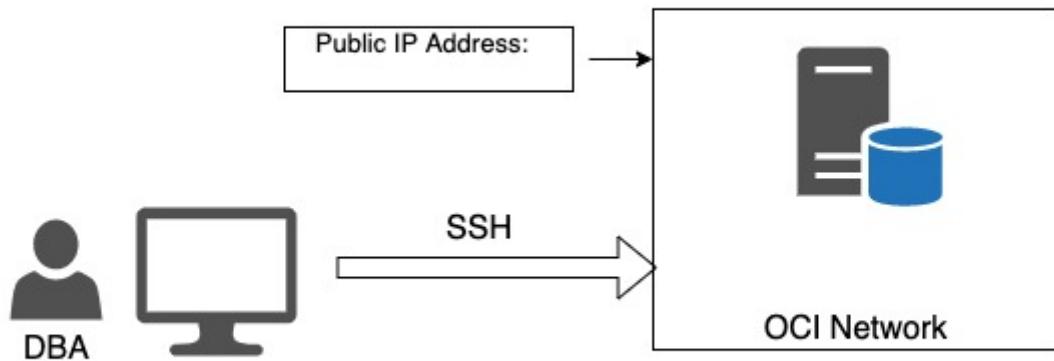
5. Linux **opc** user has limited privileges. To work with administrative privileges, use "sudo" like **Green** shell> sudo su - root

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

- o a. SSH connection
- o b. SSH key file named "id_rsa" or "
- o c. username "opc"
- o d. no password
- o 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 Enterprise Edition Install
- Install MySQL Shell and Connect to MySQL Enterprise

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

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>
```

```
perside_fo@cloudshell:~ (uk-london-1)$ ssh opc@150.230.116
The authenticity of host '150.230.116 (150.230.116,)' can't be established.
ECDSA key fingerprint is [REDACTED]
ECDSA key fingerprint is [REDACTED]
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '150.230.116 (ECDSA)' to the list of known hosts.
Last login: Sat Sep 25 21:10:08 2021 from 129.213.201 [REDACTED]
[opc@mds-client ~]$
```

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 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