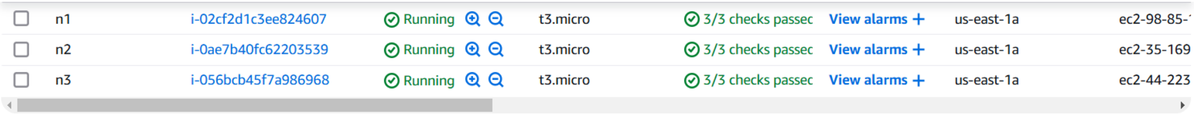
High availability MariaDB clusters set on AWS

Initial setup-

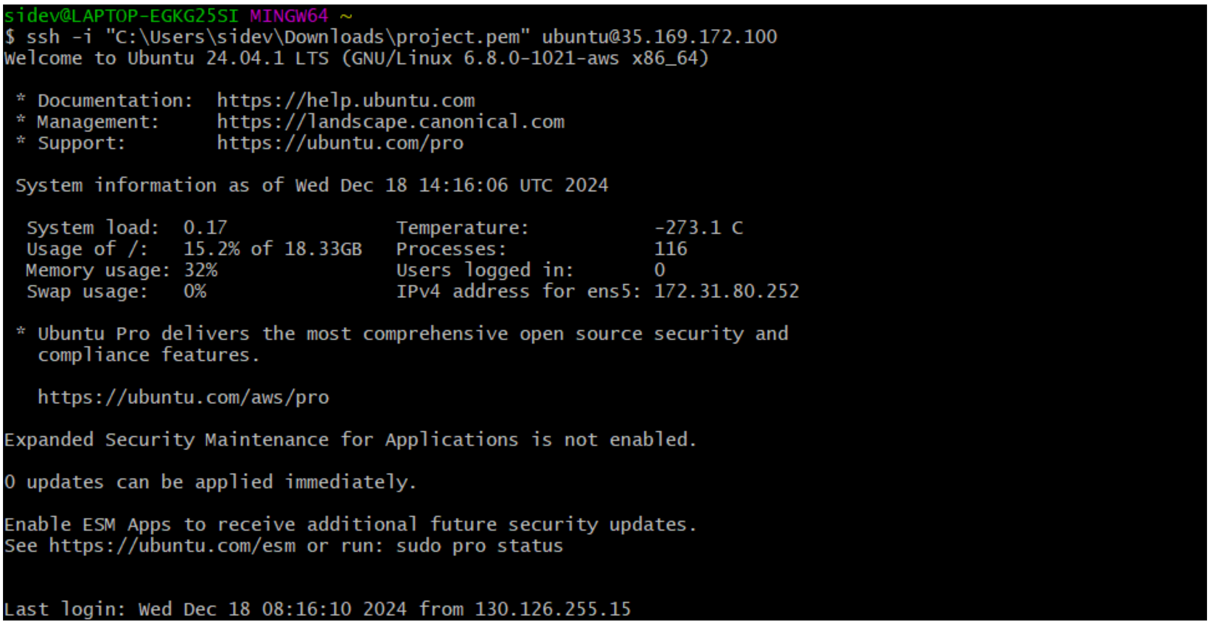
I have to set 3 ec2 t3 micro instances-



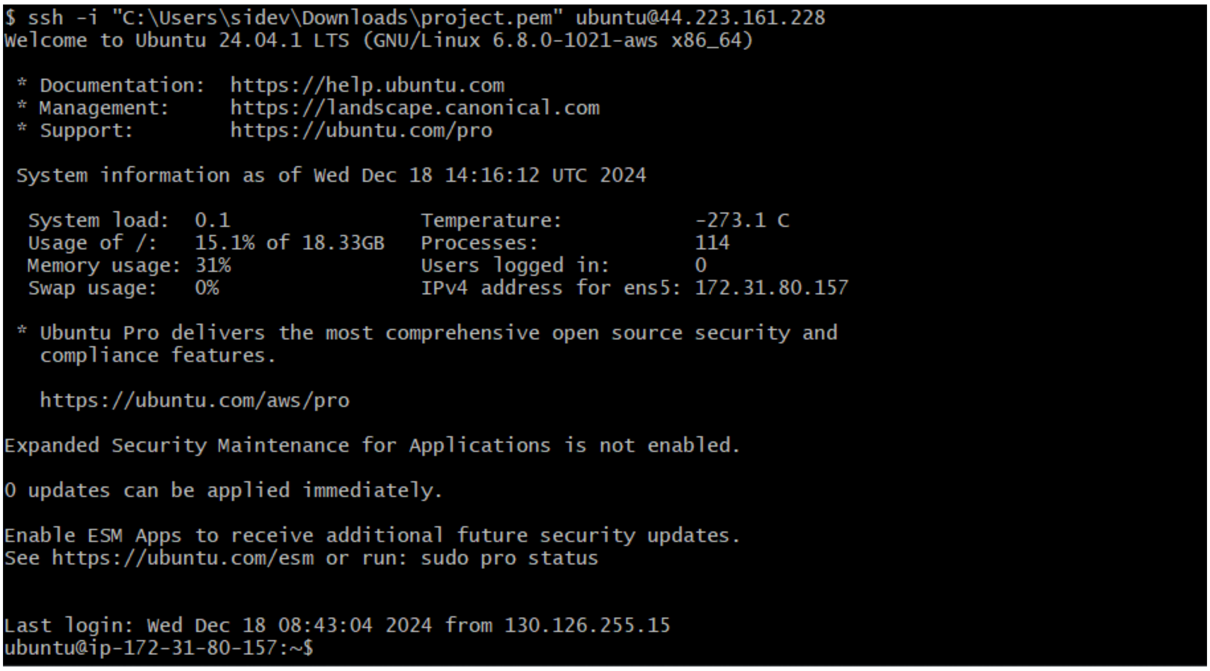
□	n1	i-02cf2d1c3ee824607	✓ Running	t3.micro	✓ 3/3 checks passec	View alarms +	us-east-1a	ec2-98-85-
□	n2	i-0ae7b40fc62203539	✓ Running	t3.micro	✓ 3/3 checks passec	View alarms +	us-east-1a	ec2-35-169
□	n3	i-056bcb45f7a986968	✓ Running	t3.micro	✓ 3/3 checks passec	View alarms +	us-east-1a	ec2-44-223

Here I have kept them in the same region for connectivity

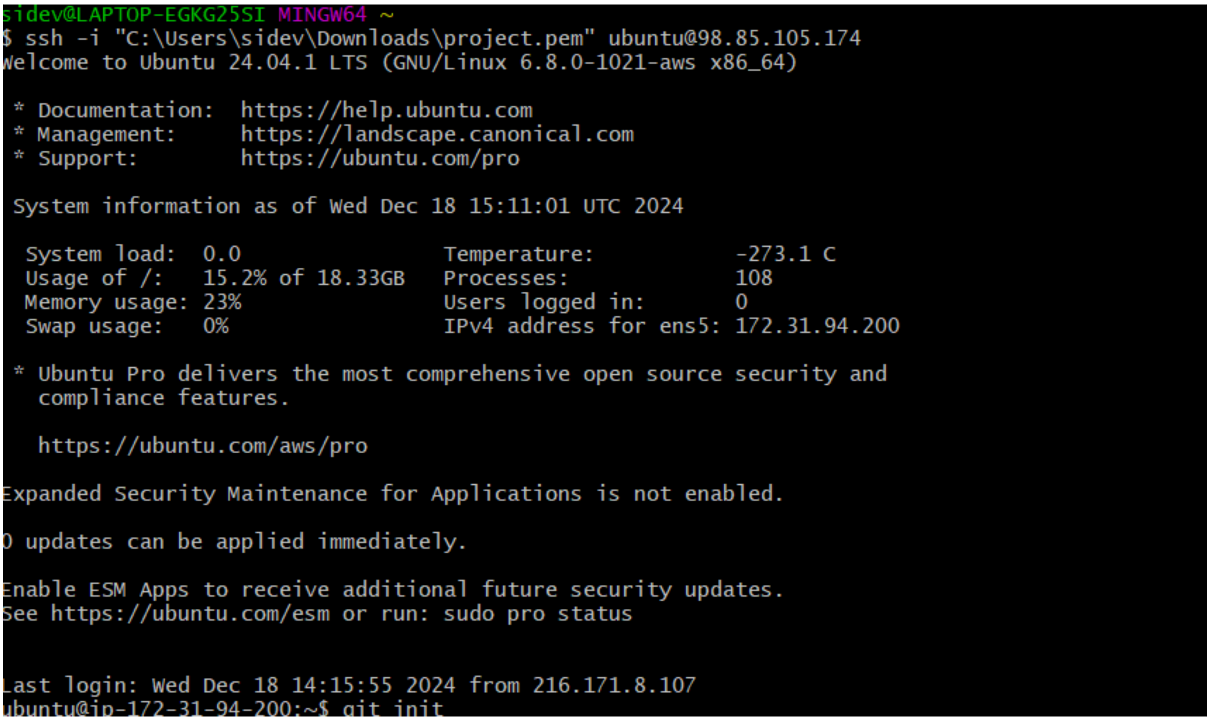
Connection-



```
sidev@LAPTOP-EGKG25SI MINGW64 ~
$ ssh -i "C:\Users\sidev\Downloads\project.pem" ubuntu@35.169.172.100
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Dec 18 14:16:06 UTC 2024

  System load:  0.17              Temperature:           -273.1 C
  Usage of /:   15.2% of 18.33GB   Processes:             116
  Memory usage: 32%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.80.252

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Wed Dec 18 08:16:10 2024 from 130.126.255.15
```



```
$ ssh -i "C:\Users\sidev\Downloads\project.pem" ubuntu@44.223.161.228
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Dec 18 14:16:12 UTC 2024

  System load:  0.1               Temperature:           -273.1 C
  Usage of /:   15.1% of 18.33GB  Processes:             114
  Memory usage: 31%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.80.157

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Wed Dec 18 08:43:04 2024 from 130.126.255.15
ubuntu@ip-172-31-80-157:~$
```

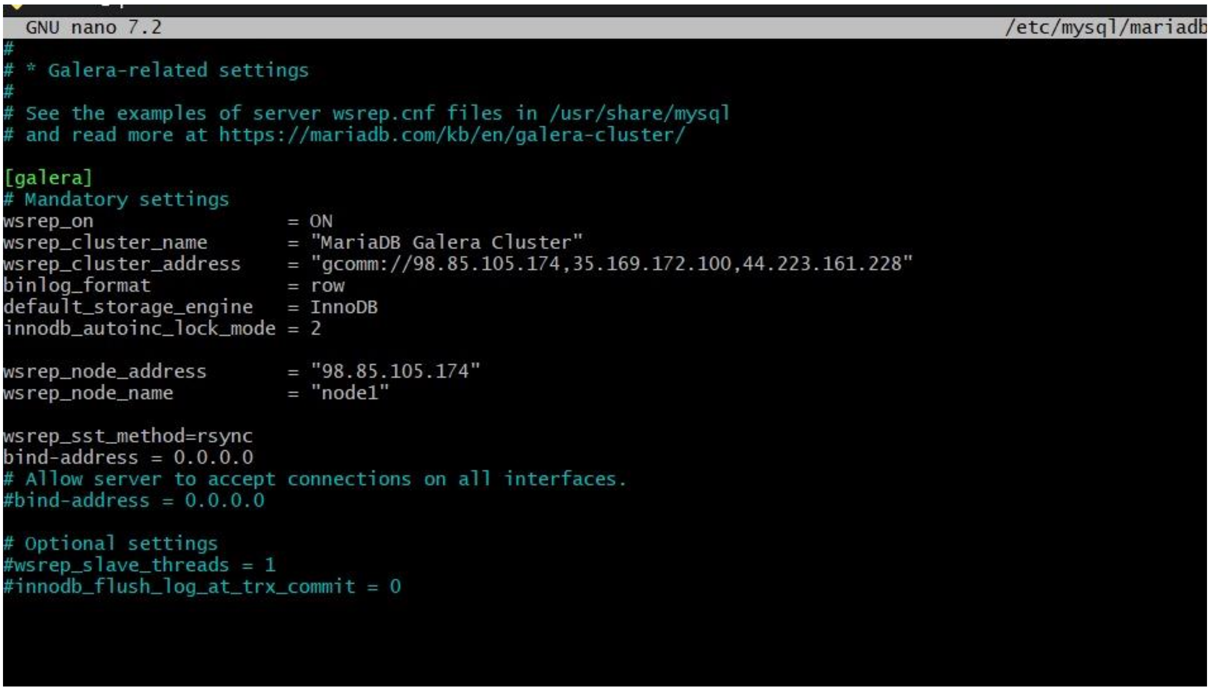


```
sidev@LAPTOP-EGKG25SI MINGW64 ~
$ ssh -i "C:\Users\sidev\Downloads\project.pem" ubuntu@98.85.105.174
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Dec 18 15:11:01 UTC 2024

  System load:  0.0               Temperature:           -273.1 C
  Usage of /:   15.2% of 18.33GB  Processes:             108
  Memory usage: 23%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.94.200

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Wed Dec 18 14:15:55 2024 from 216.171.8.107
ubuntu@ip-172-31-94-200:~$ git init
```

Here, I used an Ubuntu image instead of Linux, during the initial setup the MariaDB was not able to install due to conflicting libraries as I looked deeper into it, I realized the libraries were getting installed in Centos and there were conflicting libraries present already in Linux as well. So after carefully reading the documentation and resources, I found out that Ubuntu supports more MariaDB than Linux

Cluster setup-

In order for clusters to identify other clusters I made changes in the conf file for every MariaDB file here, this was done for every node



```
  GNU nano 7.2                                                                    /etc/mysql/mariadb
#
# * Galera-related settings
#
# See the examples of server wsrep.cnf files in /usr/share/mysql
# and read more at https://mariadb.com/kb/en/galera-cluster/

[galera]
# Mandatory settings
wsrep_on                 = ON
wsrep_cluster_name       = "MariaDB Galera Cluster"
wsrep_cluster_address    = "gcomm://98.85.105.174,35.169.172.100,44.223.161.228"
binlog_format            = row
default_storage_engine   = InnoDB
innodb_autoinc_lock_mode = 2

wsrep_node_address       = "98.85.105.174"
wsrep_node_name          = "node1"

wsrep_sst_method=rsync
bind-address = 0.0.0.0
# Allow server to accept connections on all interfaces.
#bind-address = 0.0.0.0

# Optional settings
#wsrep_slave_threads = 1
#innodb_flush_log_at_trx_commit = 0
```

Now the clusters are setup but still, they are not able to communicate with each other so during experimentation I configured how to do VPC and connect all the instances

The first step to create AMI of the instances-

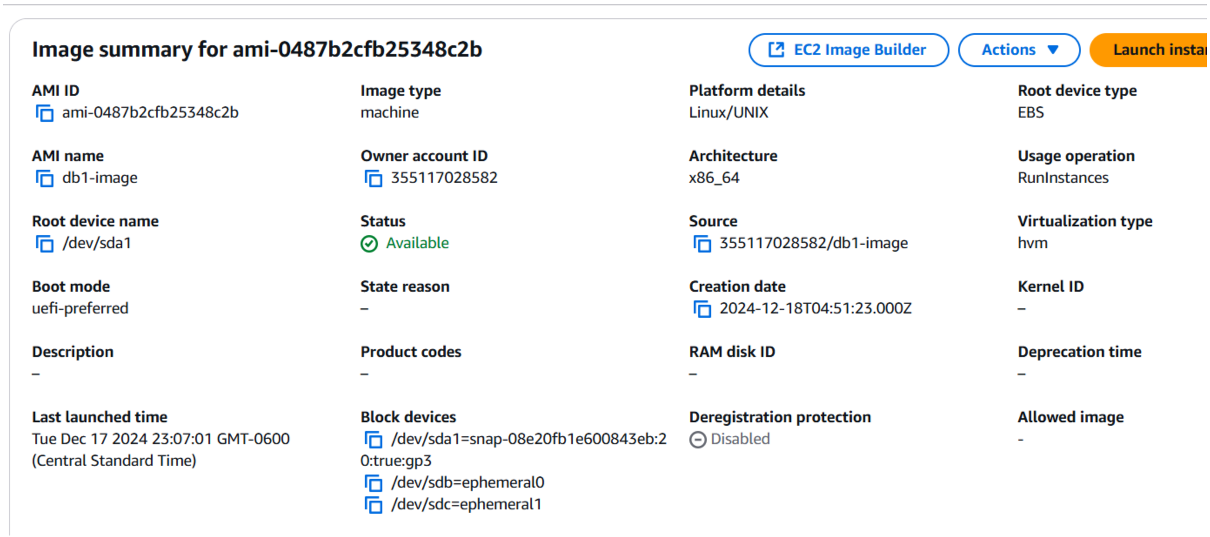
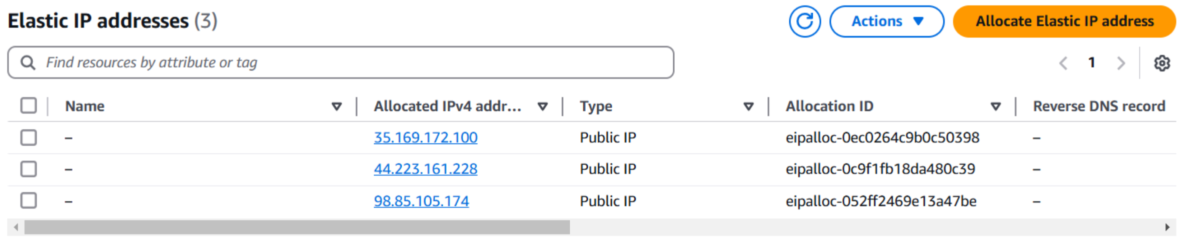


Image summary for ami-0487b2cfb25348c2b

AMI ID ami-0487b2cfb25348c2b	**Image type** machine	**Platform details** Linux/UNIX	**Root device type** EBS
AMI name db1-image	**Owner account ID** 355117028582	**Architecture** x86_64	**Usage operation** RunInstances
Root device name /dev/sda1	**Status** Available	**Source** 355117028582/db1-image	**Virtualization type** hvm
Boot mode uefi-preferred	**State reason** –	**Creation date** 2024-12-18T04:51:23.000Z	**Kernel ID** –
Description –	**Product codes** –	**RAM disk ID** –	**Deprecation time** –
Last launched time Tue Dec 17 2024 23:07:01 GMT-0600 (Central Standard Time)	**Block devices** /dev/sda1=snap-08e20fb1e600843eb:20:true:gp3 /dev/sdb=ephemeral0 /dev/sdc=ephemeral1	**Deregistration protection** Disabled	**Allowed image** -

Then launched new instances and repeated the whole process of connection and initial setup.

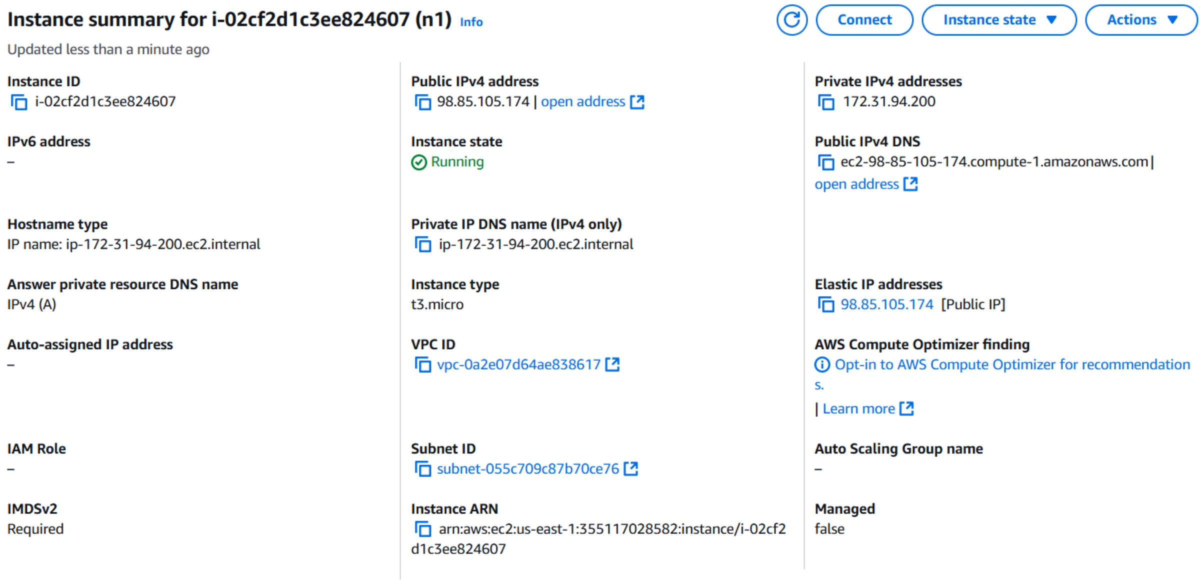
After setup, there is a concept of elastic IP address here the IP address remains constant during the tasks instead of constantly changing.



Elastic IP addresses (3)

Actions ▼
Allocate Elastic IP address
Find resources by attribute or tag
< 1 >

□	Name	Allocated IPv4 addr...	Type	Allocation ID	Reverse DNS record
□	–	35.169.172.100	Public IP	eipalloc-0ec0264c9b0c50398	–
□	–	44.223.161.228	Public IP	eipalloc-0c9f1fb18da480c39	–
□	–	98.85.105.174	Public IP	eipalloc-052ff2469e13a47be	–



Instance summary for i-02cf2d1c3ee824607 (n1) Info

Updated less than a minute ago

Instance ID
i-02cf2d1c3ee824607

IPv6 address
–

Hostname type
IP name: ip-172-31-94-200.ec2.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
–

IAM Role
–

IMDSv2
Required

Public IPv4 address
98.85.105.174 | open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-94-200.ec2.internal

Instance type
t3.micro

VPC ID
vpc-0a2e07d64ae838617

Subnet ID
subnet-055c709c87b70ce76

Instance ARN
arn:aws:ec2:us-east-1:355117028582:instance/i-02cf2d1c3ee824607

Private IPv4 addresses
172.31.94.200

Public IPv4 DNS
ec2-98-85-105-174.compute-1.amazonaws.com | open address

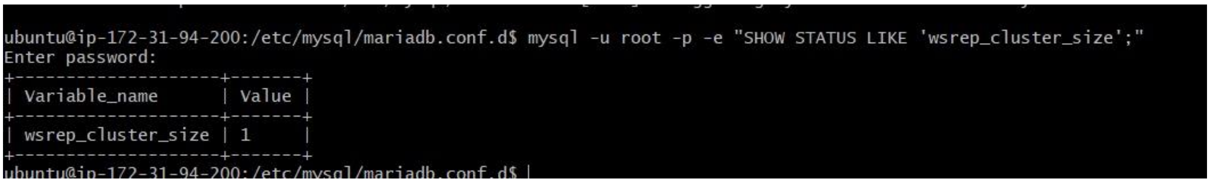
Elastic IP addresses
98.85.105.174 [Public IP]

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
| Learn more

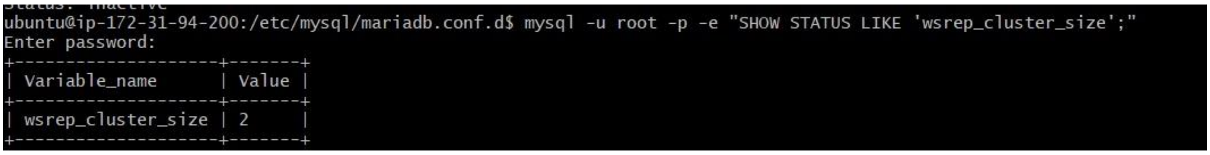
Auto Scaling Group name
–

Managed
false

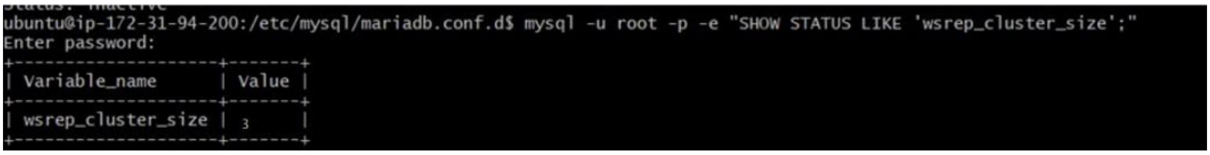
Clusters-



```
ubuntu@ip-172-31-94-200:/etc/mysql/mariadb.conf.d$ mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size';"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 1     |
+--------------------+-------+
ubuntu@ip-172-31-94-200:/etc/mysql/mariadb.conf.d$ |
```

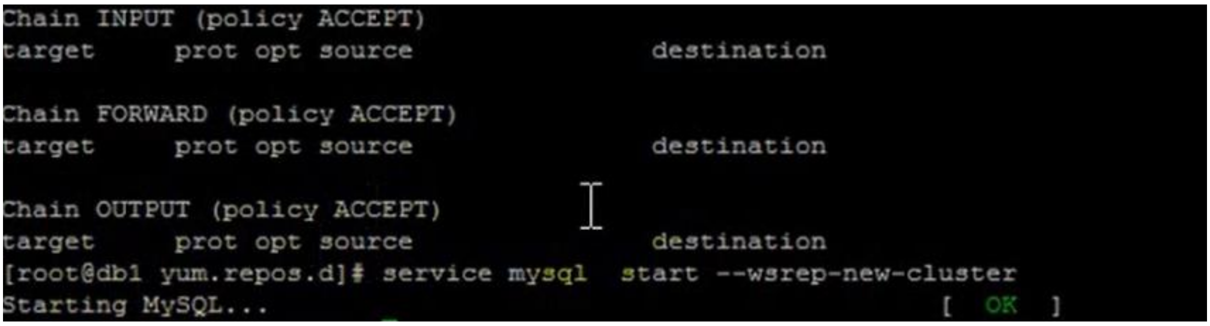


```
Status: inactive
ubuntu@ip-172-31-94-200:/etc/mysql/mariadb.conf.d$ mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size';"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 2     |
+--------------------+-------+
```



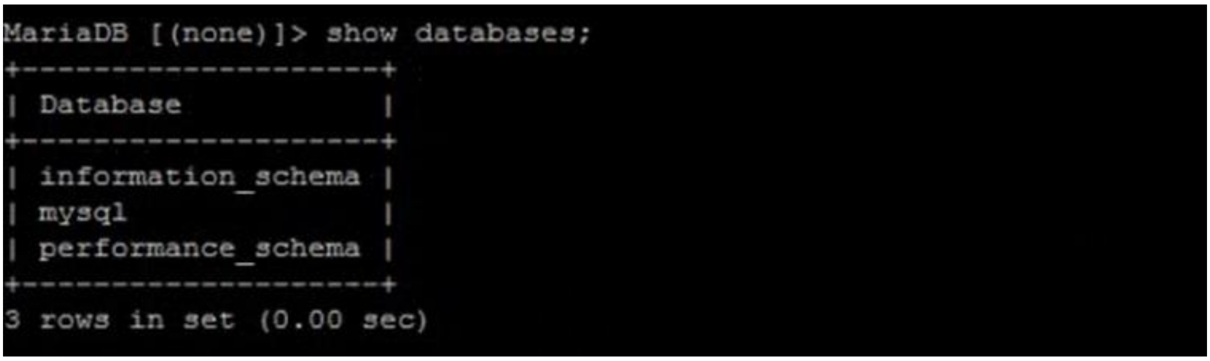
```
Status: inactive
ubuntu@ip-172-31-94-200:/etc/mysql/mariadb.conf.d$ mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size';"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
```

Above you can see cluster size of 3

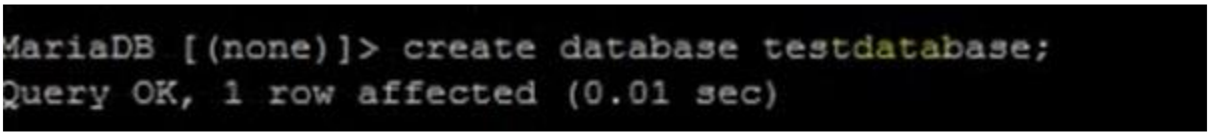


```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@db1 yum.repos.d]# service mysql  start --wsrep-new-cluster
Starting MySQL...                                          [  OK  ]
```

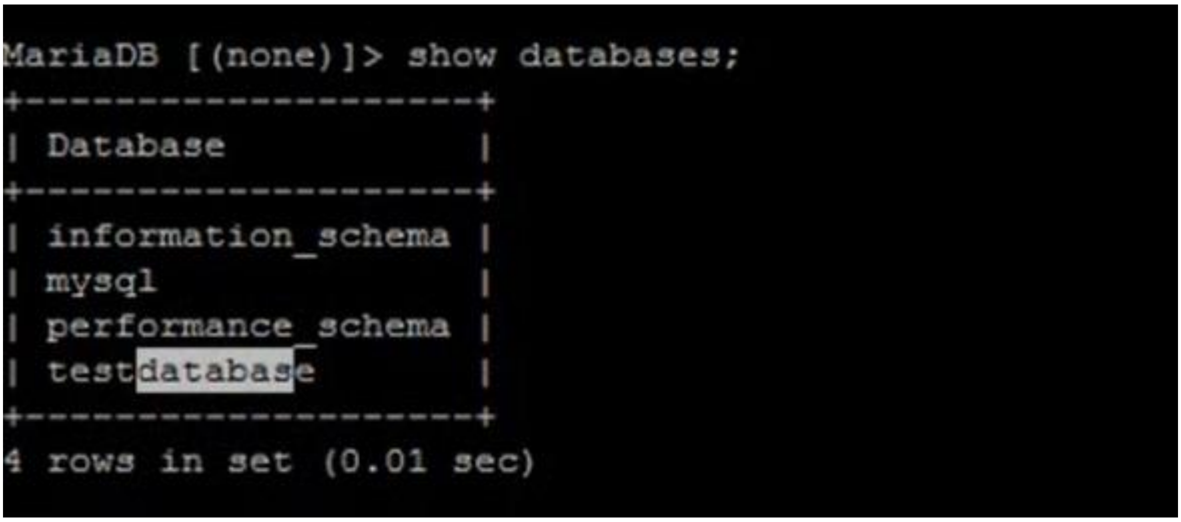
Above you can see all the nodes active, now we can query and see the replication process



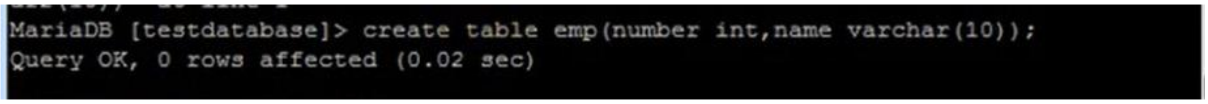
```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)
```



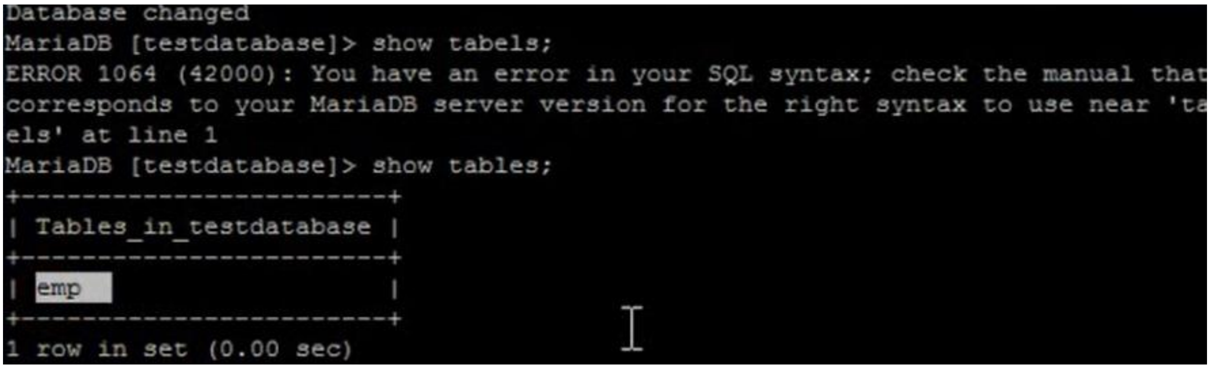
```
MariaDB [(none)]> create database testdatabase;
Query OK, 1 row affected (0.01 sec)
```



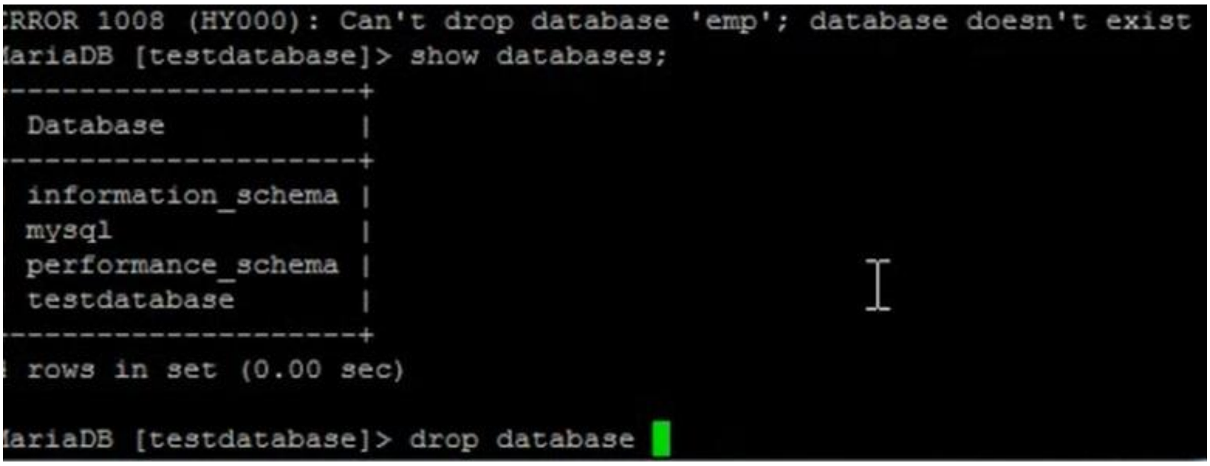
```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| testdatabase       |
+--------------------+
4 rows in set (0.01 sec)
```



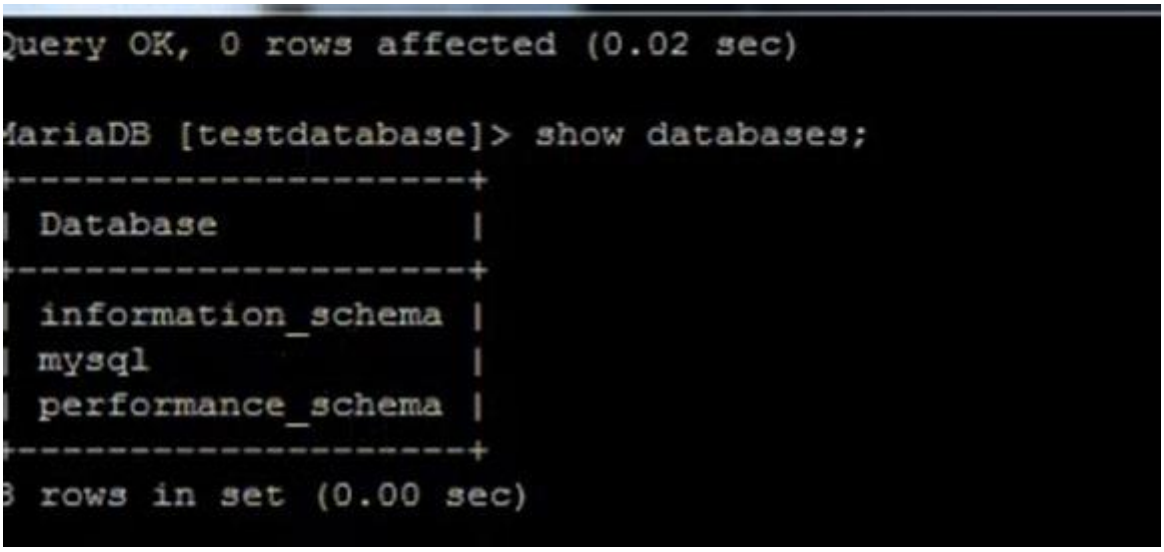
```
MariaDB [testdatabase]> create table emp(number int,name varchar(10));
Query OK, 0 rows affected (0.02 sec)
```



```
Database changed
MariaDB [testdatabase]> show tabels;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MariaDB server version for the right syntax to use near 'ta
els' at line 1
MariaDB [testdatabase]> show tables;
+------------------------+
| Tables_in_testdatabase |
+------------------------+
| emp                    |
+------------------------+
1 row in set (0.00 sec)
```



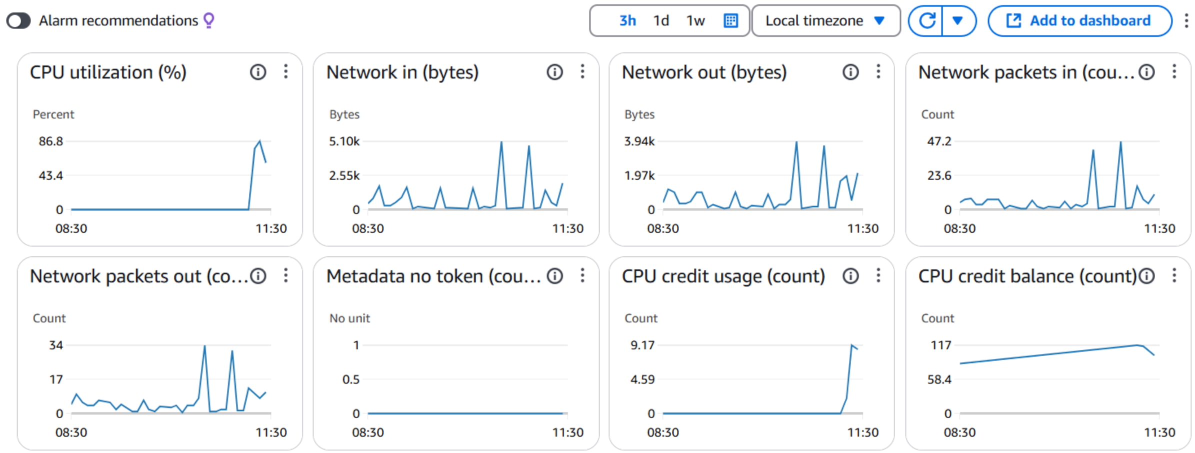
```
RROR 1008 (HY000): Can't drop database 'emp'; database doesn't exist
ariaDB [testdatabase]> show databases;
-------------------+
 Database          |
-------------------+
 information_schema |
 mysql              |
 performance_schema |
 testdatabase       |
-------------------+
 rows in set (0.00 sec)

ariaDB [testdatabase]> drop database
```

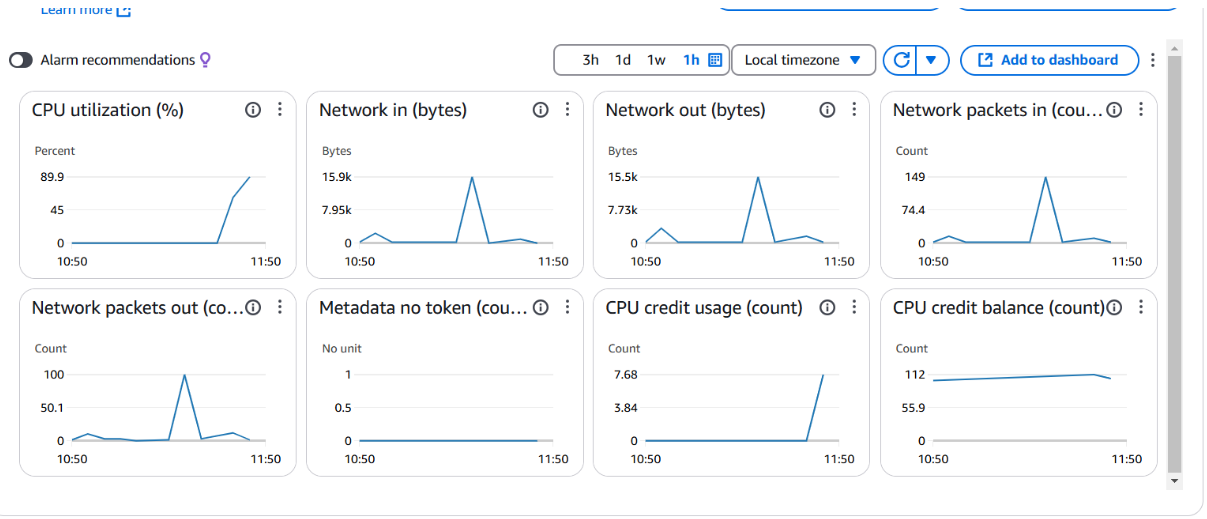


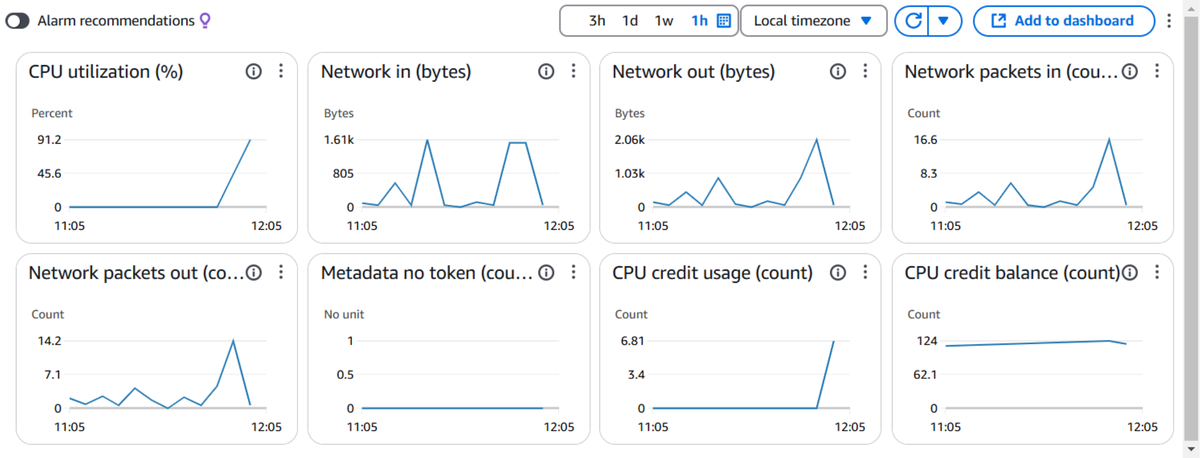
```
Query OK, 0 rows affected (0.02 sec)

MariaDB [testdatabase]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)
```

Cloud watch monitoring results-

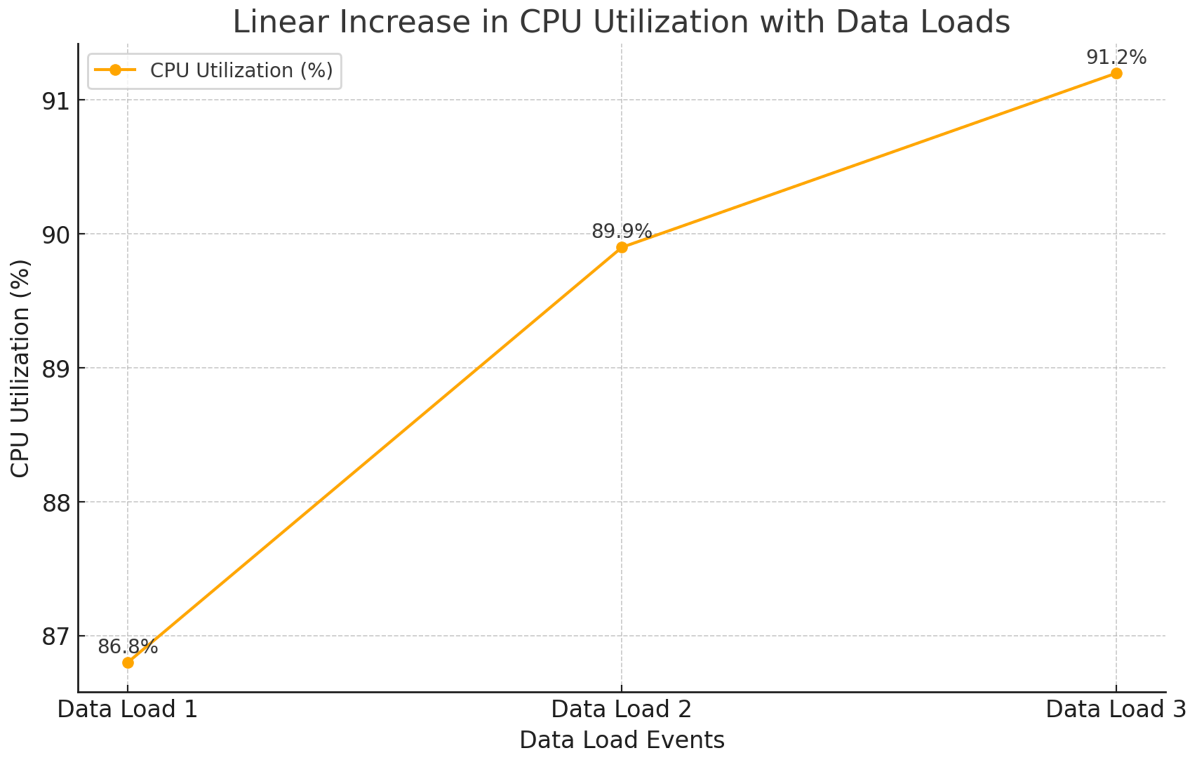
Vm2-







Analysis of the results-



Conclusion-

Galera Cluster operates on a **multi-primary architecture**, where all nodes can act as primary. This eliminates the risk of a single point of failure, as any node can handle operations seamlessly during failures.

A potential loophole arises during a **network partition**. If nodes cannot communicate, both nodes may fail due to a split-brain scenario. To mitigate this, an **arbiter node** can be introduced to participate in the quorum process, ensuring election and preventing simultaneous failures.

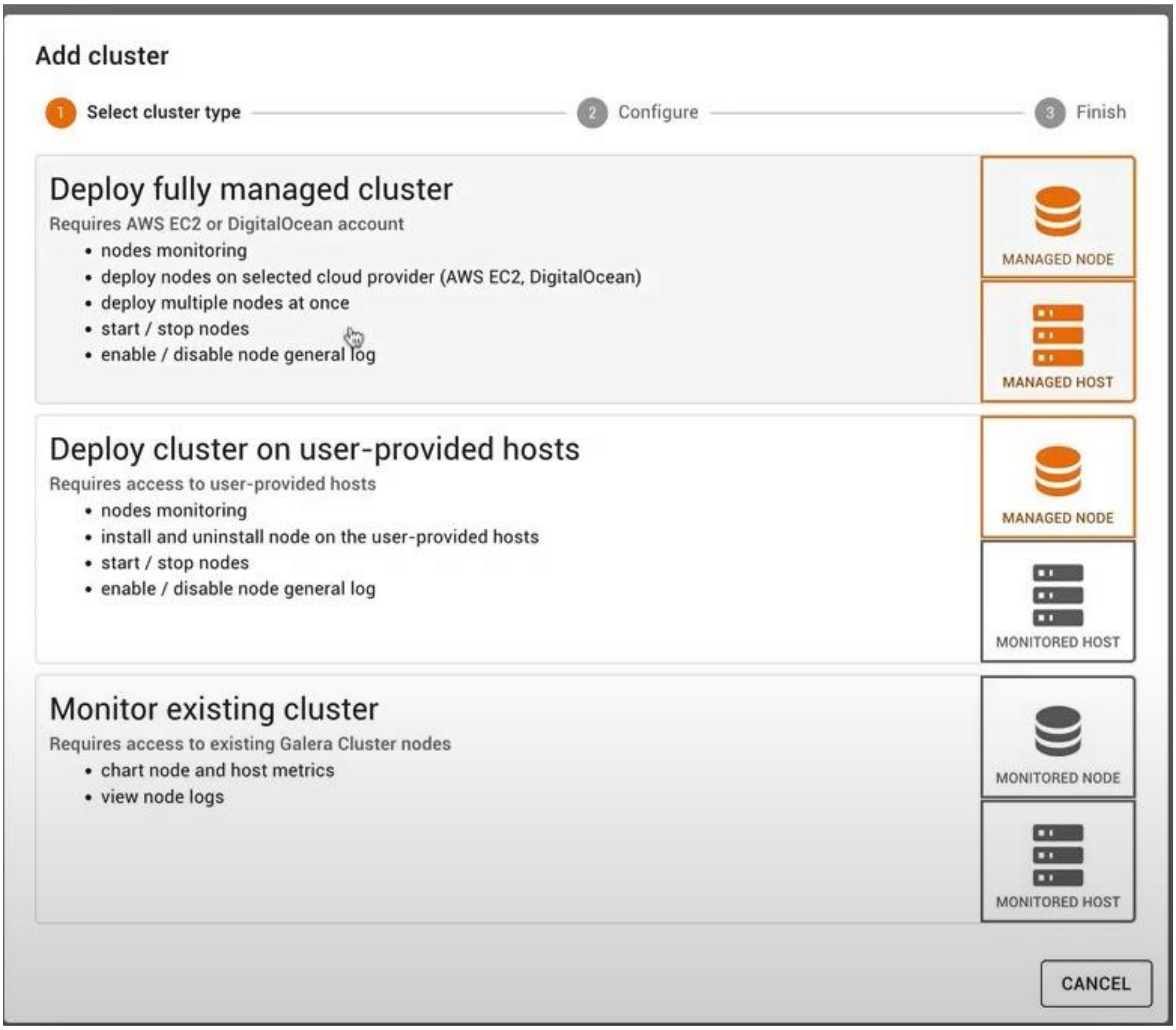
Replication lag is **minimal** in Galera Cluster because updates are applied synchronously across all nodes. This ensures consistency and near real-time replication, even during heavy data loads.

From the monitoring results, CPU utilization showed a **linear increase** with each data load (86.8%, 89.9%, 91.2%). Network activity spikes confirm efficient data transfers with no significant lag, aligning with Galera's synchronous replication capabilities.

In summary, Galera Cluster ensures high availability and low replication lag. However, an **arbiter node** implementation is recommended to handle network failures and prevent split-brain issues effectively.

Using Galera Manager-

Galera gives these options where if you do just select nodes can be implemented



Add cluster

1 Select cluster type — 2 Configure — 3 Finish

Deploy fully managed cluster

Requires AWS EC2 or DigitalOcean account

- nodes monitoring
- deploy nodes on selected cloud provider (AWS EC2, DigitalOcean)
- deploy multiple nodes at once
- start / stop nodes
- enable / disable node general log

Deploy cluster on user-provided hosts

Requires access to user-provided hosts

- nodes monitoring
- install and uninstall node on the user-provided hosts
- start / stop nodes
- enable / disable node general log

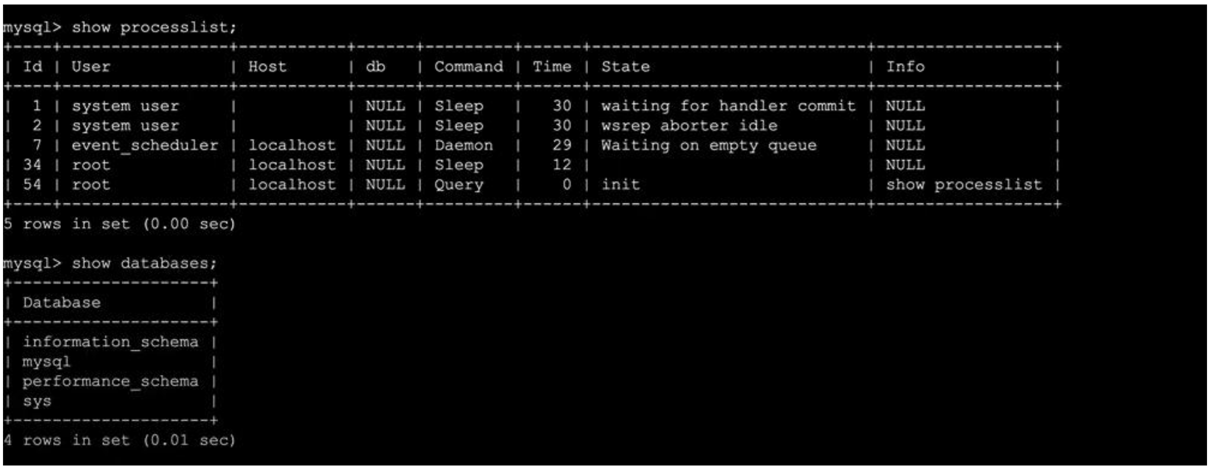
Monitor existing cluster

Requires access to existing Galera Cluster nodes

- chart node and host metrics
- view node logs

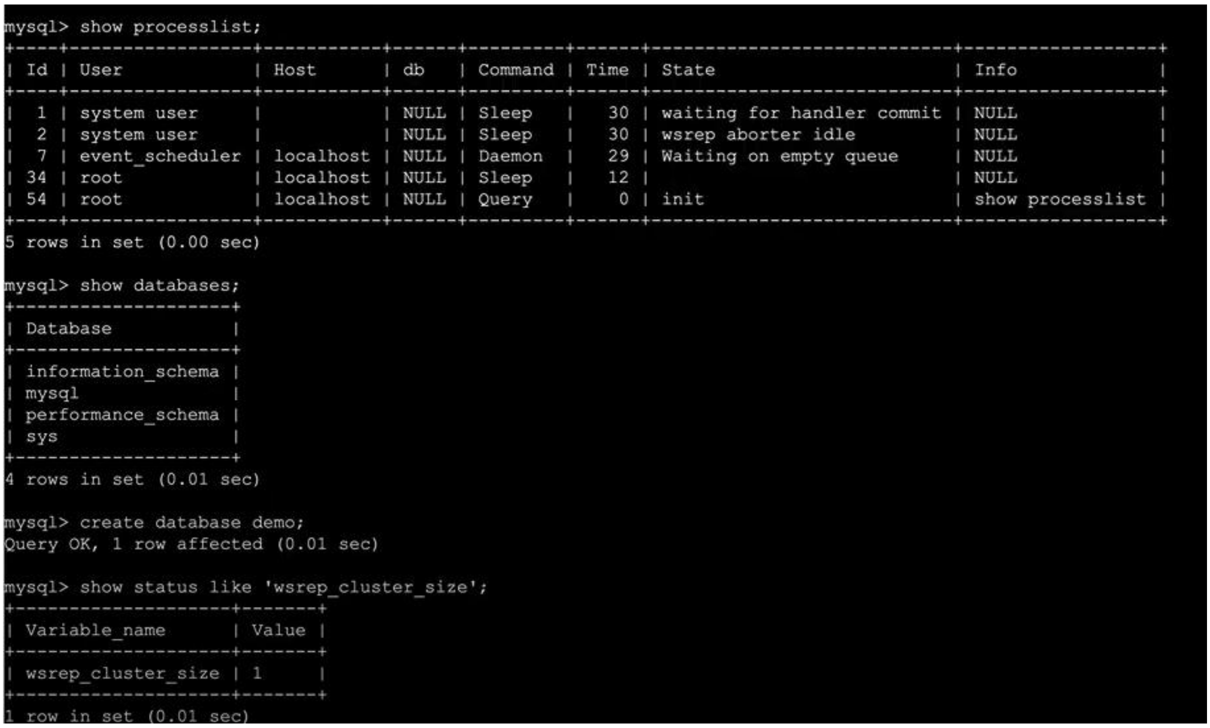
Basic configuration-

Here for configuration, we enter the SSH path while logging in here, also here Galera manager provides terminal commands where we can implement queries

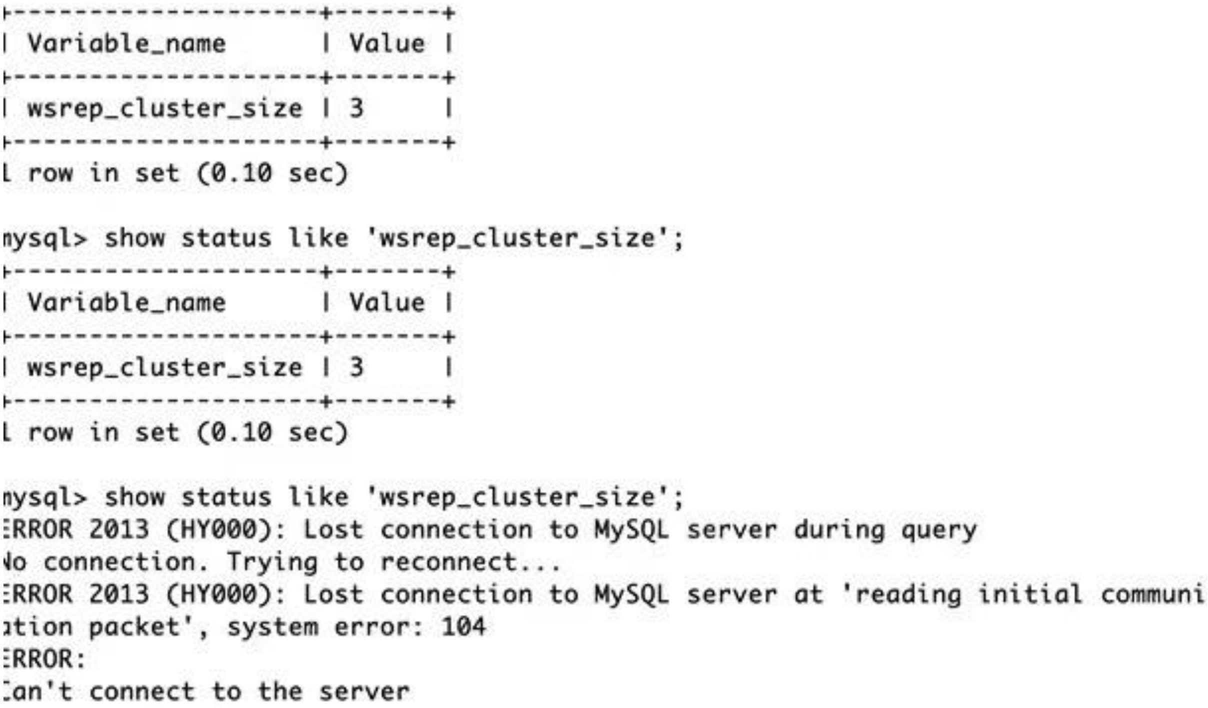


```
mysql> show processlist;
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
| Id | User            | Host      | db   | Command | Time | State                      | Info             |
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
|  1 | system user     |           | NULL | Sleep   |   30 | waiting for handler commit | NULL             |
|  2 | system user     |           | NULL | Sleep   |   30 | wsrep aborter idle         | NULL             |
|  7 | event_scheduler | localhost | NULL | Daemon  |   29 | Waiting on empty queue     | NULL             |
| 34 | root            | localhost | NULL | Sleep   |   12 |                            | NULL             |
| 54 | root            | localhost | NULL | Query   |    0 | init                       | show processlist |
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
5 rows in set (0.00 sec)

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

So here we do some querying to check our database

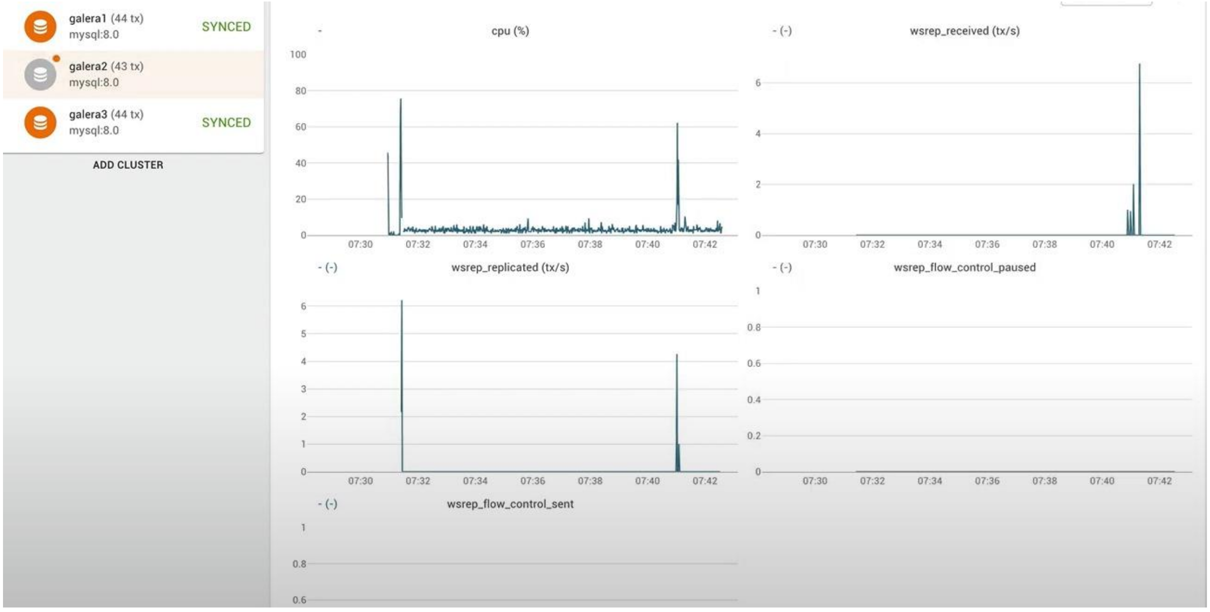


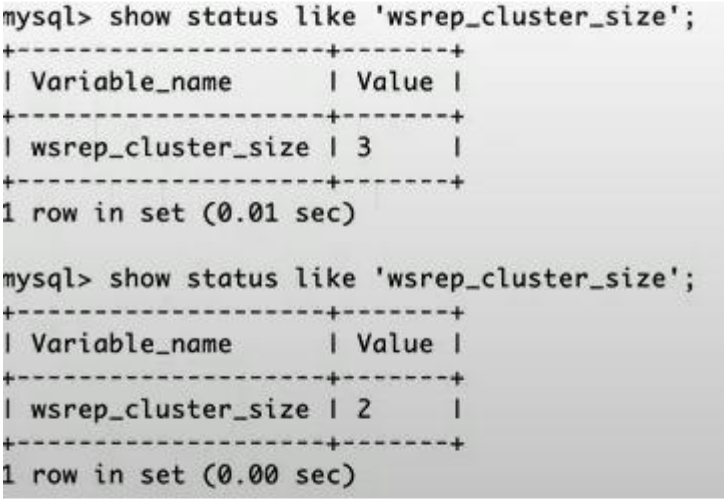
```
mysql> show processlist;
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
| Id | User            | Host      | db   | Command | Time | State                      | Info             |
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
|  1 | system user     |           | NULL | Sleep   |   30 | waiting for handler commit | NULL             |
|  2 | system user     |           | NULL | Sleep   |   30 | wsrep aborter idle         | NULL             |
|  7 | event_scheduler | localhost | NULL | Daemon  |   29 | Waiting on empty queue     | NULL             |
| 34 | root            | localhost | NULL | Sleep   |   12 |                            | NULL             |
| 54 | root            | localhost | NULL | Query   |    0 | init                       | show processlist |
+----+-----------------+-----------+------+---------+------+----------------------------+------------------+
5 rows in set (0.00 sec)

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

mysql> create database demo;
Query OK, 1 row affected (0.01 sec)

mysql> show status like 'wsrep_cluster_size';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 1     |
+--------------------+-------+
1 row in set (0.01 sec)
```



```
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
l row in set (0.10 sec)

nysql> show status like 'wsrep_cluster_size';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
l row in set (0.10 sec)

nysql> show status like 'wsrep_cluster_size';
:RROR 2013 (HY000): Lost connection to MySQL server during query
lo connection. Trying to reconnect...
:RROR 2013 (HY000): Lost connection to MySQL server at 'reading initial communi
ation packet', system error: 104
:RROR:
:an't connect to the server
```

So now we check for node failure





```
mysql> show status like 'wsrep_cluster_size';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
1 row in set (0.01 sec)

mysql> show status like 'wsrep_cluster_size';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 2     |
+--------------------+-------+
1 row in set (0.00 sec)
```

Conclusion-

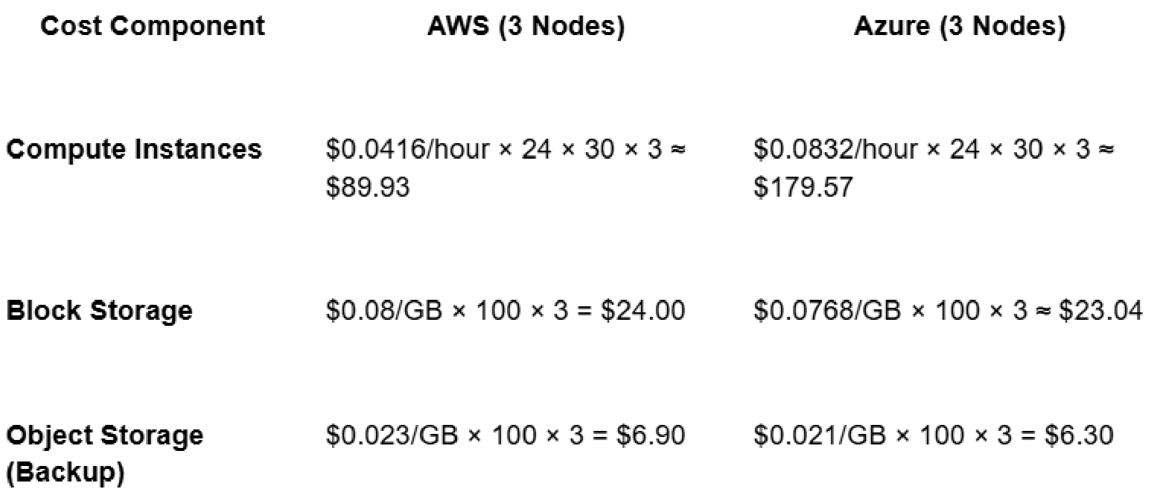
During **Node 1 and Node 2 deployment**, CPU usage and transaction spikes were significantly higher, indicating active workload and synchronization between the nodes.

When **one node was shut down**, the overall activity (CPU usage, wsrep_received, and wsrep_replicated) decreased as fewer nodes were participating in the replication process.

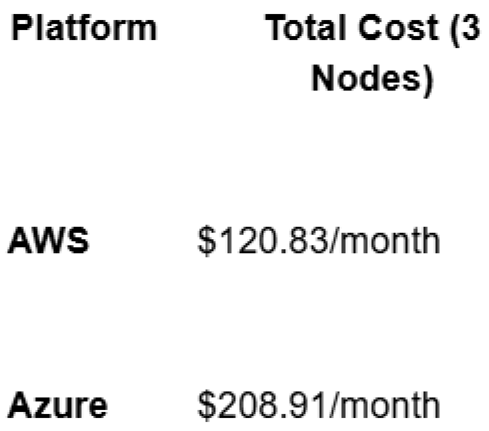
Conclusion:
The spike patterns reflect deployment and shutdown activities, with

reduced load and synchronization when a node was offline. The cluster adjusted as expected, maintaining stability during these transitions.

Cost analysis-



Cost Component	AWS (3 Nodes)	Azure (3 Nodes)
Compute Instances	$0.0416/hour × 24 × 30 × 3 ≈ $89.93	$0.0832/hour × 24 × 30 × 3 ≈ $179.57
Block Storage	$0.08/GB × 100 × 3 = $24.00	$0.0768/GB × 100 × 3 ≈ $23.04
Object Storage (Backup)	$0.023/GB × 100 × 3 = $6.90	$0.021/GB × 100 × 3 = $6.30



Platform	Total Cost (3 Nodes)
AWS	$120.83/month
Azure	$208.91/month

The Azure cost is an estimated cost as Azure has multiple free tier restrictions as multiple B2ms instances cannot be deployed. Also, another option was trying another region but that will lead to failure in connecting with other nodes, as vpc needs the same region to configure it. Hence estimating cost and providing suggestions was my best bet.

Conclusive Summary-

This project demonstrates the behavior and performance of a **Galera Cluster** setup, showcasing its multi-primary architecture, minimal replication lag, and robust synchronization across nodes.

Key findings include:

- CPU and transaction spikes occur during node deployment and synchronization, reflecting active workloads.
- Brief flow control pauses were observed, ensuring consistent replication and stability during high-load operations.
- The system effectively managed node failures, with reduced activity observed when one node was shut down.

Galera Manager served as a user-friendly **UI for cluster management**. However, it primarily facilitates monitoring and basic operations. For actual **node deployment**, further licensing beyond the free trial is required. Additionally, node deployment within Galera Manager remains unclear, as the instances were directly managed on AWS infrastructure, and no documentation for deploying it. After further research galera manager cannot launch additional nodes it just can manage them was discovered

Overall, the project highlights Galera Cluster's efficiency in high-availability scenarios while identifying areas for further clarity regarding deployment processes.

References-

- **Dataset-**

  https://github.com/kite1988/nus-sms-corpus/blob/master/smsCorpus_en_sql_2015.03.09_all.zip

  https://www.kaggle.com/datasets/thedevastator/unlock-profits-with-e-commerce-sales-data

  https://www.kaggle.com/datasets/manjeetsingh/retaildataset/data

- **https://galeracluster.com/library/documentation/galera-manager.html**

- **https://dev.mysql.com/doc/refman/8.4/en/mysqldump.html**

- **https://galeracluster.com/library/documentation/install-mariadb.html**