https://www.linkedin.com/in/ahmed-mohamed-423583151

Day 5 High Availability and Disaster Recovery

PostgreSQL Replication and High Availability Solutions

For postgresql there is two way for high availability and disaster recovery

for high availability best solution is patroni that consist of 3 nodes running postgresql with one node know as leader handle read and write request and other node replica node get transaction from leader written at them , the two replica node only accept read request only
Reason for having three node is to have voting node to avoid split brain in the cluster .
For better management of connection we can use virtual ip the span between the leader node in case failover occur , our haproxy which also help load balance the load by forwarding read request to the replica nodes and r/w request to the leader node , haproxy is also integrated great with patroni so it will be aware if failover happened

For disaster recovery the solution is to use streaming replication , postgresql native way of creating replica , streaming replication work by copying wal log to replica and then the replica store them , the replica will be on read only , and incase of disaster the failover need to be manually done .
In my experience streaming replication fit better for disaster recovery scenario .

High Availability patroni

Here are prequest that need to be follow to setup patroni

- Postgresql installed on three node and then services is disabled
- Etcd need to be installed on three node
- Host file configuration need to be adjusted to point to three node
- Start etcd and each node
- Create patroni ayaml file and then start in first node then follow by rest
  Actually setup patroni is very long process but good news is that there way to automate deployment of patroni using docker container called **autobase**
  autobase simplifies the process. With autobase's console ui, you simply enter your server information and select the version you want, and then autobase takes care of the rest.

Autobase uses ansible templates to perform the tasks, and it requires ssh access to the server, either through a password or ssh key. The user account used to access the server must have root or sudo privileges to ensure the necessary tasks can be executed properly.

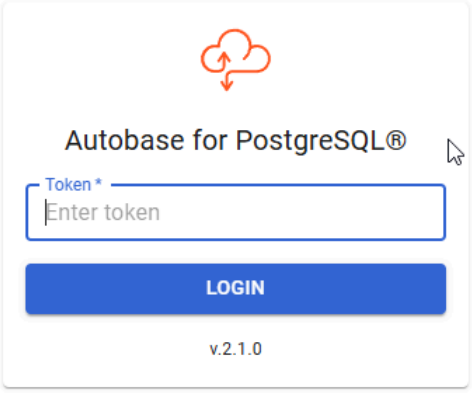
Using autobase to deploy patroni

I have already three node running ubuntu , internet is allowed for downloading
Next steps is to start autobase as docker container , for that you need to install docker you can have dedicated vm running docker , our for portability if you have windows 11 or 10 you can have docker installed by running gust linux system using wls
https://www.windowscentral.com/how-install-wsl2-windows-10

```
docker run -d --name autobase-console \
  --publish 80:80 \
  --publish 8080:8080 \
  --env PG_CONSOLE_API_URL=http://10.10.10.88:8080/api/v1 \
  --env PG_CONSOLE_AUTHORIZATION_TOKEN=secret_token \
  --volume console_postgres:/var/lib/postgresql \
  --volume /var/run/docker.sock:/var/run/docker.sock \
  --volume /tmp/ansible:/tmp/ansible \
  --restart=unless-stopped \
  autobase/console:2.1.0
```

Note: if you are running the console on a dedicated server (rather than on your laptop), replace `localhost` with the server's ip address in the `pg_console_api_url` variable.

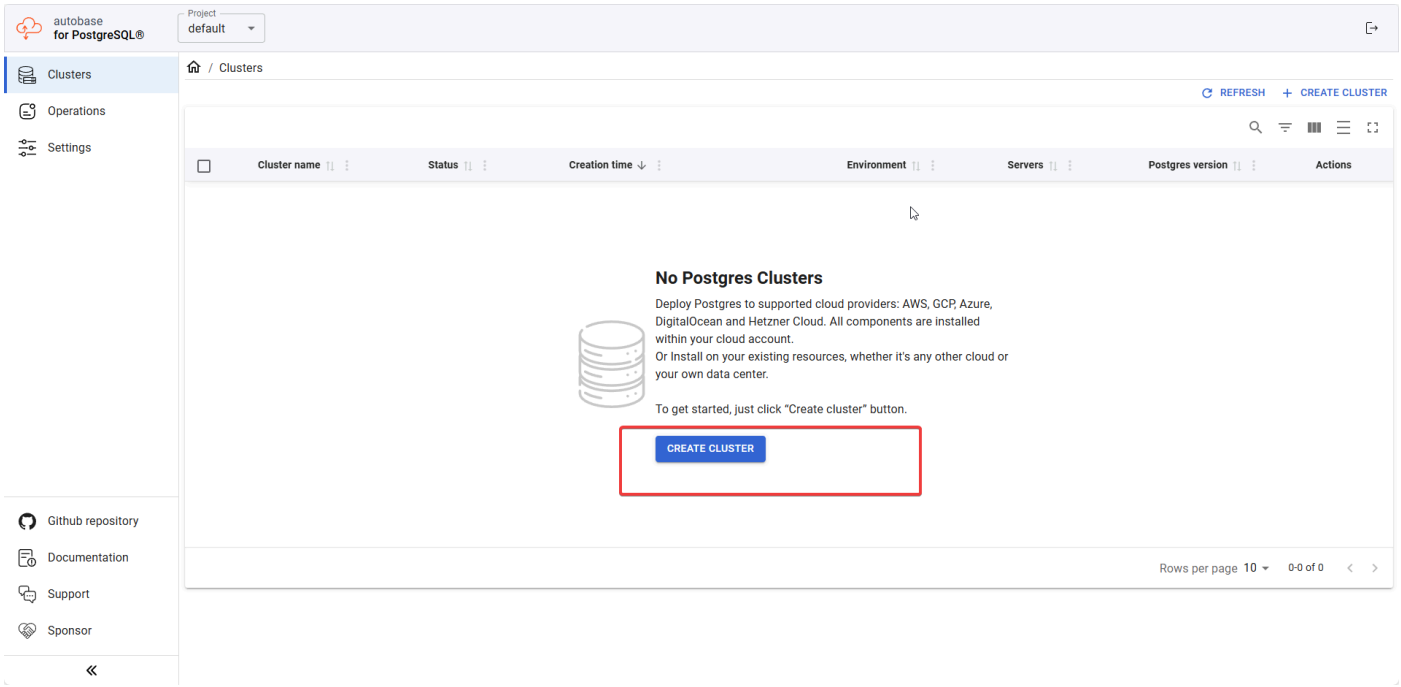
Open the console by browsing to o= http://localhost:80
it will ask for just past `secret_token`



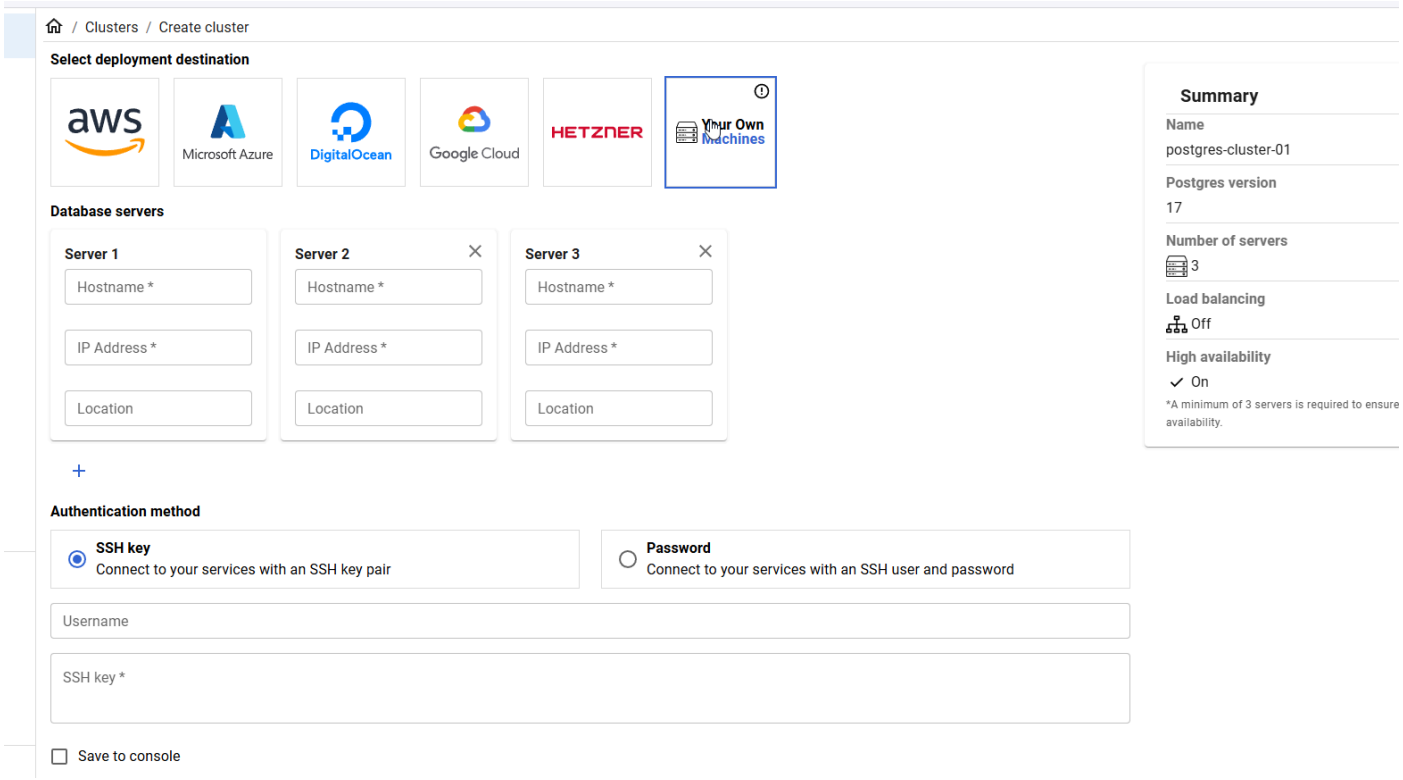
The console is open lets start by deploying patroni
below are details for server

hostname	ip
patroni-node1	10.10.10.86
patroni-node2	10.10.10.87
patroni-node3	10.10.10.85

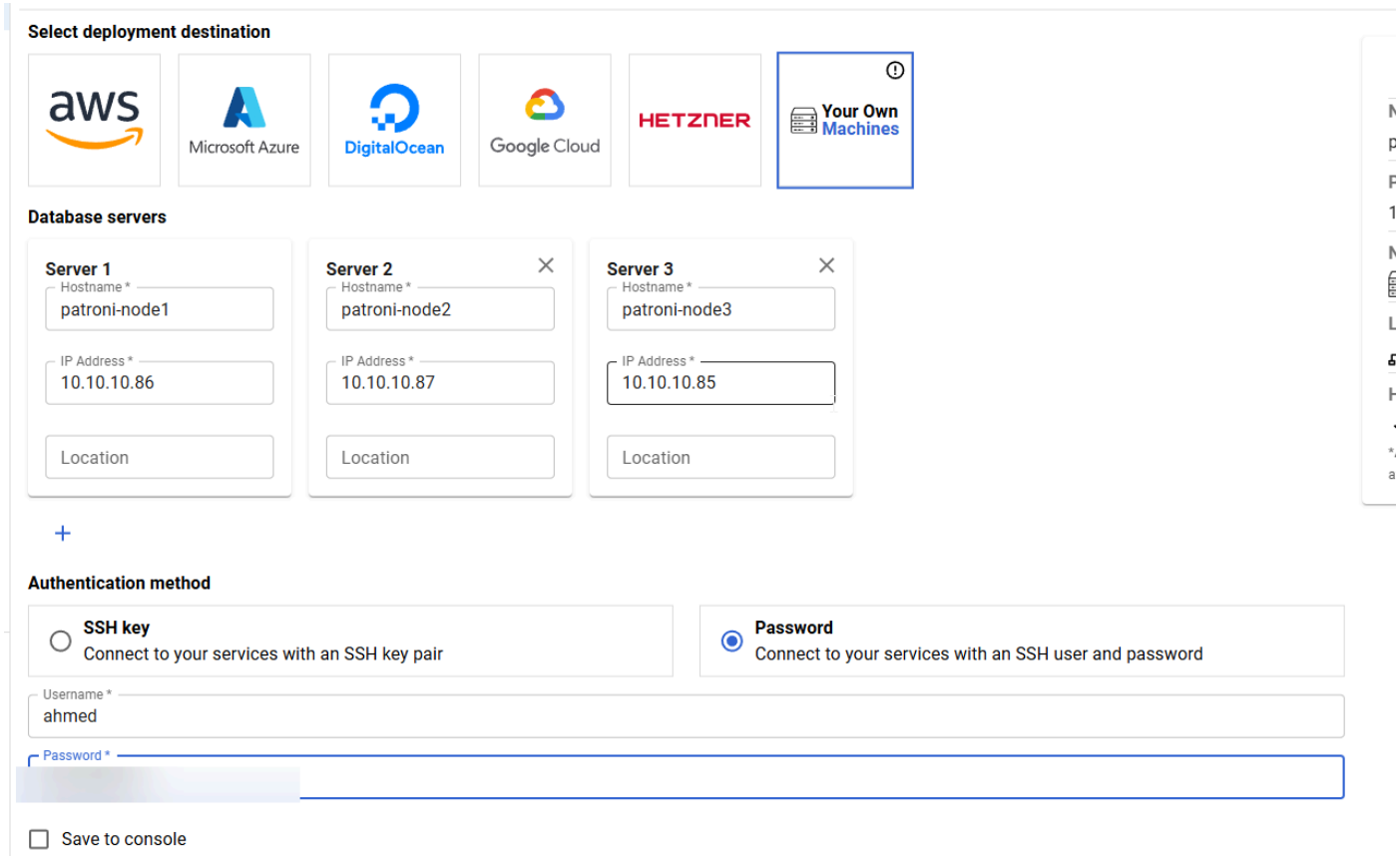
Click on **Create Cluster** to start patron Deployment



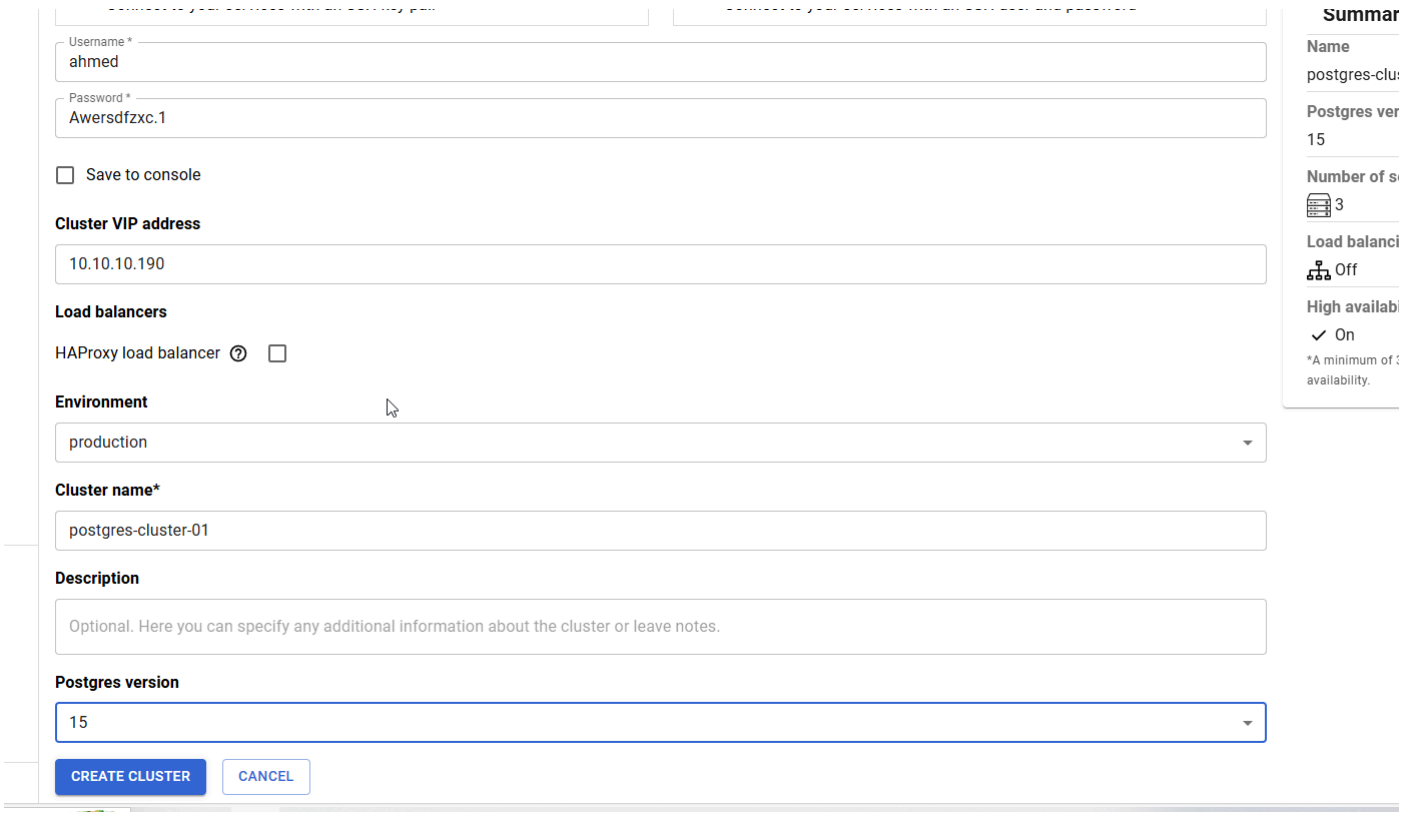
one great thing about autobase it automate deployment of patroni on cloud providers such as aws , gcp ,and azure
For our scenario select **your own machines**



Fill the request information , also make sure the user you proved has sudor privilege's

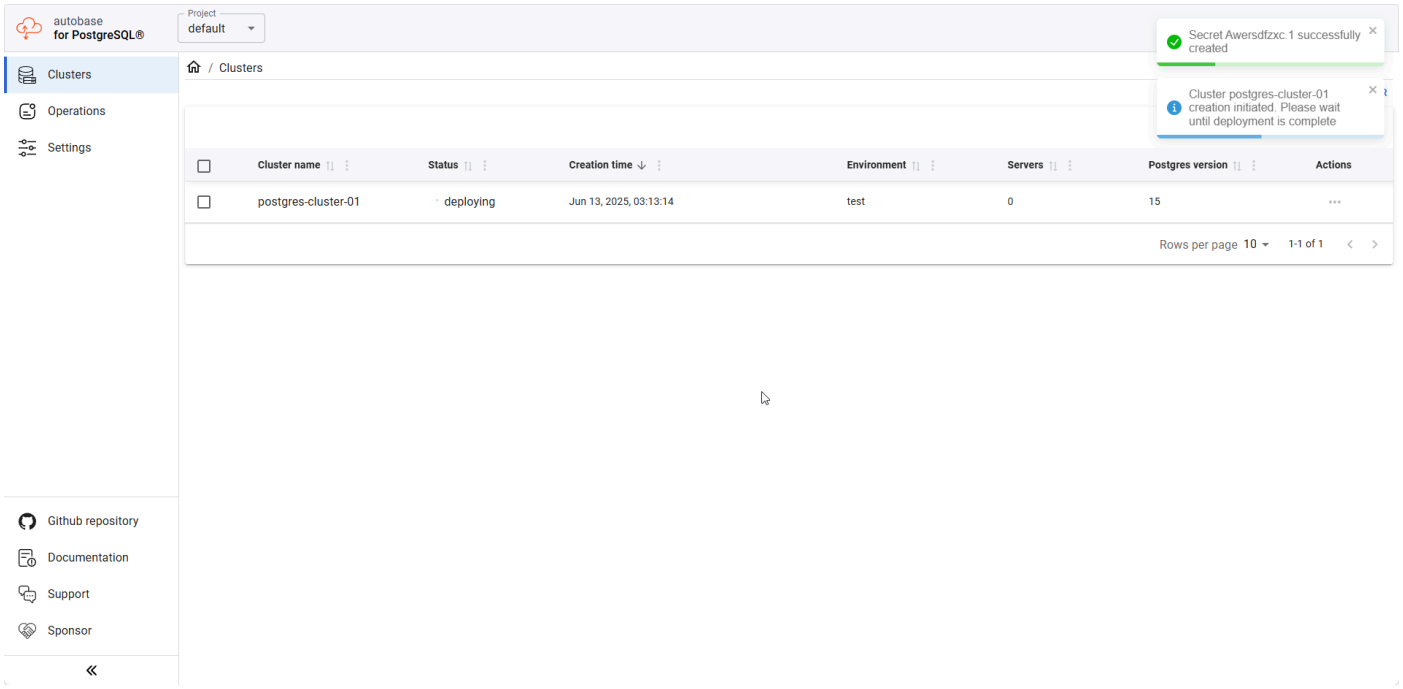


For cluster vip address provide free ip , and also you can opt in to install haproxy for load balance but i will skip installing haproxy



Fill the cluster name its just naming and select postgresqlversion then click **create cluster**

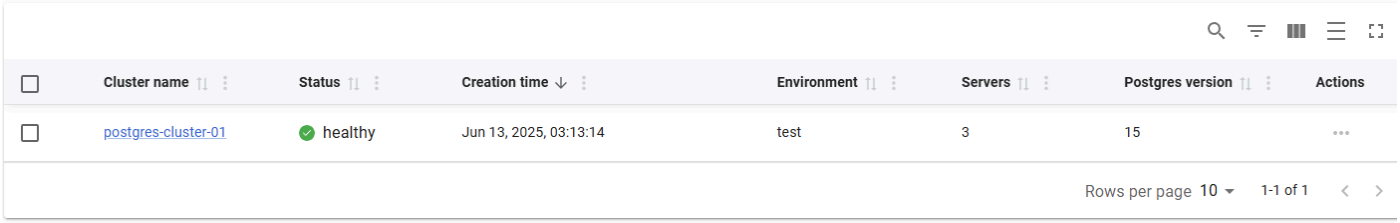
Now the deployment is start it will take time denning on bandwidth and server specs



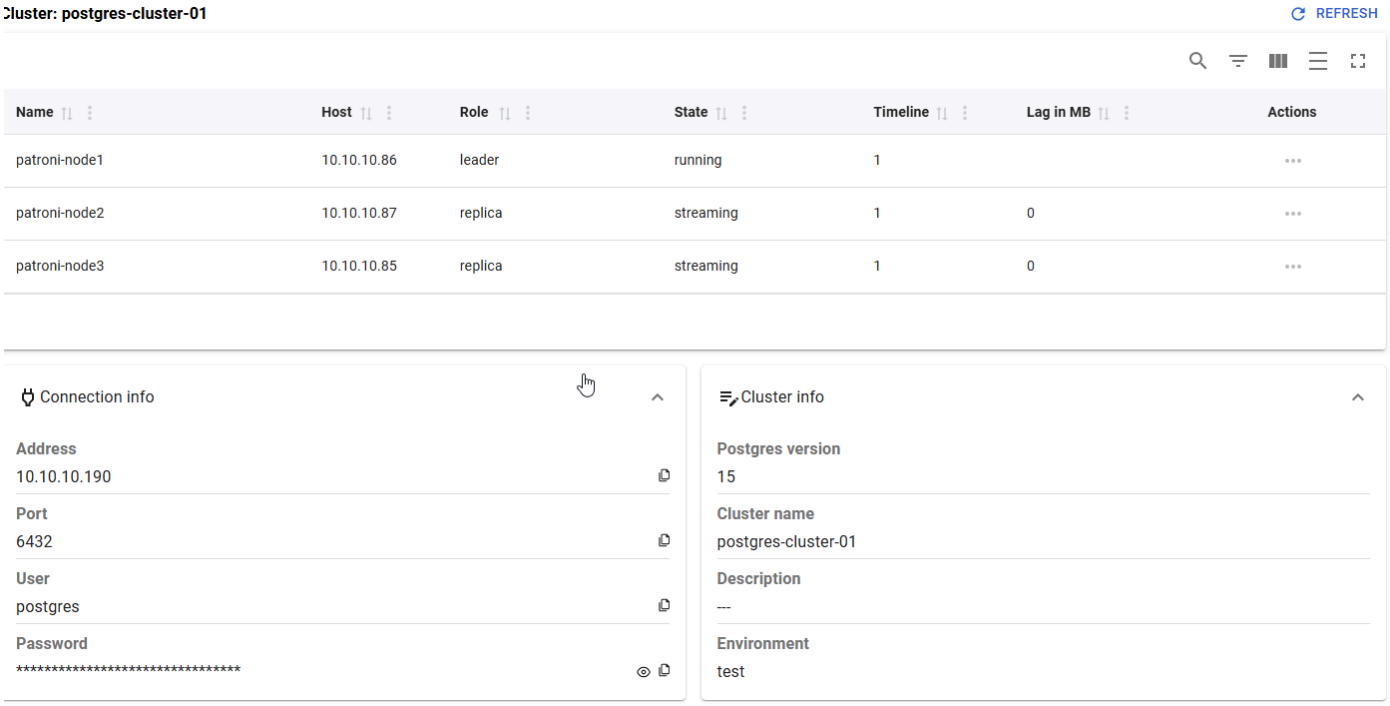
Note that autobase is not only deployment it also do lot thing during deplyment

- Parameter configuration for postgresql
- Install pg_backrest for backup
- Install and configure pgbouncer for connection pooling
- Setting kernel parameters
- Create swap file

- Pgpass : configure a password file
  Once done you will see status of cluster as healthy



Click on it and you will see status of cluster and connection info with password



Login to one of node and you can run some patroni command

Below command will show you teh status pf the cluster

```
patronictl  -c /etc/patroni/patroni.yml list
```

```
root@patroni-node1:/etc/patroni# patronictl  -c /etc/patroni/patroni.yml list
+ Cluster: postgres-cluster-01 (7515222890752999683) ----+-----------+
| Member        | Host         | Role    | State     | TL | Lag in MB |
+---------------+--------------+---------+-----------+----+-----------+
| patroni-node1 | 10.10.10.86 | Leader  | running   |  1 |           |
| patroni-node2 | 10.10.10.87 | Replica | streaming |  1 |         0 |
| patroni-node3 | 10.10.10.85 | Replica | streaming |  1 |         0 |
+---------------+--------------+---------+-----------+----+-----------+
root@patroni-node1:/etc/patroni#
```

If want to preform patroni setup manually i suggest to use percona guide following along to setup patroni
https://docs.percona.com/postgresql/16/solutions/high-availability.html

Streaming replication

Streaming one core native solution for replication and base on it there many tools that help such as patroni remgr
Streaming replication can be used for dr setup it will not have auto failover failover need to be done manually , if you want more easy way to failover you can use `rebmgr` but in my testing `rebmgr` tend to be headcheese for setup and managing

1- Create replication user in the primacy server

Login to `psql` and create replication user as follow

```
`CREATE USER replication REPLICATION LOGIN CONNECTION LIMIT 1 ENCRYPTED PASSWORD '123456789'
```

```
postgres=# CREATE USER replication REPLICATION LOGIN CONNECTION LIMIT 1 ENCRYPTED PASSWORD '123456789'
postgres-# ;
CREATE ROLE
postgres=#
```

2- change configuration in postgresql.conf

Use `nano` or `vi` to update configuration in the file ` `sudo nano /etc/postgresql/15/main/postgresql.conf`

```
listen_addresses = '*'
wal_level = replica
max_wal_senders = 10
wal_keep_segments = 64
```

3- update `pg_hba.conf` file add replica server

Now slave server need authentication for replication. Now append following line to `/etc/postgresql/15/main/pg_hba.conf` file

`sudo nano /etc/postgresql/15/main/pg_hba.conf`

`# Replace 10.10.10.78 with slave server's private ip

host replication replication 10.10.10.78/24 md5

```
#
# Database administrative login by Unix domain socket
local   all             postgres                                peer

# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            scram-sha-256
host    all             all             10.10.10.152/24         trust
host    all             all             10.10.10.151/24         trust
# IPv6 local connections:
host    all             all             ::1/128                 scram-sha-256
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            scram-sha-256
host    replication     all             ::1/128                 scram-sha-256
host    replication     replication     10.10.10.151/24         md5
"/etc/postgresql/15/main/pg_hba.conf" 107L, 5139B
```

4- Stop PostgreSQL services in replica server

```
`systemctl stop postgresql@15-main.service
systemctl status postgresql@15-main.service
```

5- Edit parameter in postrgesql.conf

use `nano` or `vi` to update configuration in the file ` `sudo nano /etc/postgresql/15/main/postgresql.conf`

```
listen_addresses = ''
wal_level = replica
max_wal_senders = 10
wal_keep_segments = 64
hot_standby = on
```

6- Update `pg_hba.conf` and add the primary server ip

`sudo nano /etc/postgresql/12/main/pg_hba.conf`

```
# Replace 10.10.10.502 with slave master's private IP
host    replication     replication     10.10.10.50/24   md5
```

7- Remove all files in data directory**

for that its better to login as root user and remove all content of data directory

usually it placed by default in `var/lib/postgresql/15/main` if you not sure where data data directory located user the command `pg_lscluster`

```
root@postgresql-replica:~# cd /var/lib/postgresql/15/main/
root@postgresql-replica:/var/lib/postgresql/15/main# rm -rf*
```

```
ahmed@postgresql-replica:~$ sudo -i
root@postgresql-replica:~# cd /var/lib/postgresql/15/main/
root@postgresql-replica:/var/lib/postgresql/15/main# rm -rf*
rm: invalid option -- '*'
Try 'rm --help' for more information.
root@postgresql-replica:/var/lib/postgresql/15/main# rm -rf *
root@postgresql-replica:/var/lib/postgresql/15/main# ls
root@postgresql-replica:/var/lib/postgresql/15/main#
```

8- Use pg_base backup to take backup from primriay server

```
sudo su postgres
cd /var/lib/postgresql/15/main/
pg_basebackup -h 10.10.10.150 -U replication -p 5432 -D /var/lib/postgresql/15/main/  -Fp -Xs -P -R
```

Remember the password set for the replication user we created. In the following step, you will be prompted to enter the password for this replication use

```
postgres@postgresql-replica:~/15/main$ pg_basebackup -h 10.10.10.150 -U replication -p 5432 -D /var/lib/postgresql/12/main/  -Fp -Xs -P -R
Password:
23035/23035 kB (100%), 1/1 tablespace
postgres@postgresql-replica:~/15/main$
```

Once the fetching process is complete, proceed to start the PostgreSQL service

```
`systemct   start postgresql@15-main.service
systemctl status postgresql@15-main.service
```

```
==== AUTHENTICATION COMPLETE ====
ahmed@postgresql-replica:~$ systemctl status postgresql@15-main.service
● postgresql@15-main.service - PostgreSQL Cluster 15-main
     Loaded: loaded (/usr/lib/systemd/system/postgresql@.service; enabled-runtime; preset: enabled)
     Active: active (running) since Sat 2025-06-14 01:55:21 UTC; 27s ago
    Process: 9429 ExecStart=/usr/bin/pg_ctlcluster --skip-systemctl-redirect 15-main start (code=exited, status=0/SUCCESS)
   Main PID: 9434 (postgres)
      Tasks: 5 (limit: 2318)
     Memory: 34.3M (peak: 42.9M)
        CPU: 385ms
     CGroup: /system.slice/system-postgresql.slice/postgresql@15-main.service
             ├─9434 /usr/lib/postgresql/15/bin/postgres -D /var/lib/postgresql/15/main -c config_file=/etc/postgresql/15/main/postgresql.conf
             ├─9435 "postgres: 15/main: checkpointer "
             ├─9436 "postgres: 15/main: background writer "
             ├─9437 "postgres: 15/main: startup recovering 000000010000000000000001A"
             └─9438 "postgres: 15/main: walreceiver streaming 0/1A000060"

Jun 14 01:55:18 postgresql-replica systemd[1]: Starting postgresql@15-main.service - PostgreSQL Cluster 15-main...
Jun 14 01:55:21 postgresql-replica systemd[1]: Started postgresql@15-main.service - PostgreSQL Cluster 15-main.
ahmed@postgresql-replica:~$
```

```
pg_ctl: directory "/var/lib/postgresql/15/main" is not a database cluster directory
2025-06-14 01:55:19.024 UTC [9434] LOG:  starting PostgreSQL 15.13 (Ubuntu 15.13-1.pgdg24.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0, 64-bit
2025-06-14 01:55:19.024 UTC [9434] LOG:  listening on IPv4 address "0.0.0.0", port 5432
2025-06-14 01:55:19.024 UTC [9434] LOG:  listening on IPv6 address "::", port 5432
2025-06-14 01:55:19.028 UTC [9434] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
2025-06-14 01:55:19.039 UTC [9437] LOG:  database system was interrupted; last known up at 2025-06-14 01:54:59 UTC
2025-06-14 01:55:19.156 UTC [9437] LOG:  entering standby mode
2025-06-14 01:55:19.156 UTC [9437] LOG:  starting backup recovery with redo LSN 0/19000028, checkpoint LSN 0/19000060, on timeline ID 1
2025-06-14 01:55:19.162 UTC [9437] LOG:  redo starts at 0/19000028
2025-06-14 01:55:19.165 UTC [9437] LOG:  completed backup recovery with redo LSN 0/19000028 and end LSN 0/19000100
2025-06-14 01:55:19.165 UTC [9437] LOG:  consistent recovery state reached at 0/19000100
2025-06-14 01:55:19.165 UTC [9434] LOG:  database system is ready to accept read-only connections
2025-06-14 01:55:19.208 UTC [9438] LOG:  started streaming WAL from primary at 0/1A000000 on timeline 1
ahmed@postgresql-replica:~$
```

Form the log you can observed that postgresql entered replica mode and its receiving wall log from primary server

To test it create new database in primary server and check if it got replicate top the replica

```
ahmed@postgresql-pri:~$ sudo vi /etc/postgresql/15/main/pg_hba.conf
ahmed@postgresql-pri:~$ sudo -u postgres psql
could not change directory to "/home/ahmed": Permission denied
psql (15.13 (Ubuntu 15.13-1.pgdg24.04+1))
Type "help" for help.

postgres=# create database test1;
CREATE DATABASE
postgres=# \d
Did not find any relations.
postgres=# \l
                                                 List of databases
   Name    |  Owner   | Encoding | Collate |  Ctype  | ICU Locale | Locale Provider |   Access privileges
-----------+----------+----------+---------+---------+------------+-----------------+-----------------------
 postgres  | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            |
 template0 | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            | =c/postgres          +
           |          |          |         |         |            |                 | postgres=CTc/postgres
 template1 | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            | =c/postgres          +
           |          |          |         |         |            |                 | postgres=CTc/postgres
 test1     | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            |
(4 rows)

postgres=#
```

```
ahmed@postgresql-replica:~$ sudo -u postgres psql
could not change directory to "/home/ahmed": Permission denied
psql (15.13 (Ubuntu 15.13-1.pgdg24.04+1))
Type "help" for help.

postgres=# \l
                                                 List of databases
   Name    |  Owner   | Encoding | Collate |  Ctype  | ICU Locale | Locale Provider |   Access privileges
-----------+----------+----------+---------+---------+------------+-----------------+-----------------------
 postgres  | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            |
 template0 | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            | =c/postgres          +
           |          |          |         |         |            |                 | postgres=CTc/postgres
 template1 | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            | =c/postgres          +
           |          |          |         |         |            |                 | postgres=CTc/postgres
 test1     | postgres | UTF8     | C.UTF-8 | C.UTF-8 |            | libc            |
(4 rows)

postgres=#
```

Monitoring replication

We can verify the replication status by using the following command. If the state displays 'streaming', it indicates that everything is functioning correctly

`SELECT * FROM pg_stat_replication;

The command `select * from pg_stat_replication;` is used in postgresql to monitor the status of streaming replication from the **primary (master) server**.

When you run this command on a primary postgresql server, it queries the `pg_stat_replication` view and returns one row for each connected standby (replica) server. This allows you to see real-time information about your replication setup.

In simple terms, it answers questions like:

- Which replica servers are currently connected to me?
- What is the current state of each replication connection?
- How far behind is each replica? (replication lag)
- From which ip address is each replica connecting?

Key Columns in the Output

The output of the command provides several important columns, including:

Column	Description
`application_name`	The name of the standby server, usually set in the replica's configuration.
`client_addr`	The IP address of the connecting standby server.
`state`	The current state of the connection (e.g., `streaming`, `catchup`, `backup`).

Column	Description
`sent_lsn`	The last Write-Ahead Log (WAL) position sent to the replica.
`replay_lsn`	The last WAL position that has been replayed (applied) on the replica.
`write_lag`	The time elapsed between flushing WAL on the primary and writing it on the standby.
`flush_lag`	The time elapsed between flushing WAL on the primary and flushing it on the standby.
`replay_lag`	The time elapsed between flushing WAL on the primary and replaying it on the standby. This is the most common metric for "replication lag".
`sync_state`	The synchronization state of the replica (e.g., `async`, `sync`, `quorum`).

Mysql replication and high availability solutions

For mysql high availability solutions there are mainly two options

- Mysql innodb cluster
- Precona extradb cluster
  For this propose I will demonstrate innodb cluster which consist of three node to avoid split brain and we will use MySQL router for load balance and to track master server and router traffic if failover happened.

For disaster recovery the best option is to use the old school master slave replication , similar to PostgreSQL streaming replication the failover has to be done manually , master slave replicate bin-log , and require to be concisely monitored in case replication failing happened In this section i will demonstrate master slave for innodb cluster to keep the guide short i advice to search on online and setup it up by your self

MySQL master slave

the guide show how to setup MySQL master slave on MySQL enterprise
you can obtain the enterprise iso from oracle website

installing MySQL Enterprise db1 & db2

we already downloaded the MySQL enterprise iso and added it on vm

we have added three spread disk on the vm that we use to create separate directory for the following

- MySQL data dir
- MySQL bin-log
- MySQL backup
  also make sure to disable selinuix

https://www.linkedin.com/in/ahmed-mohamed-423583151

```
NAME        FSTYPE      LABEL      UUID                                   MOUNTPOINT
sda
├─sda1      xfs                    4f9847a8-fcf4-4cd5-8112-839b4ce105fc   /boot
└─sda2      LVM2_member            Puc9HS-HSQD-BxFG-aVb6-Vmpg-h4qF-9a9zDg
  ├─rl-root xfs                    8f4becc1-2a8e-4454-b960-3f1bbb07bfc8   /
  └─rl-swap swap                   2dd6962b-67d2-48ce-a3a6-ecec205956c1   [SWAP]
sdb
└─sdb1      xfs                    8ad799ef-01f7-4531-b60a-e75c02f46cfb   /mysqldata
sdc
└─sdc1      xfs                    90204968-e050-47ee-8ca1-9e0779360b5e   /mysqlbinlog
sdd
└─sdd1      xfs                    e463b348-389e-4f61-9f24-6f1c0f92e884   /mysqlbackup
sr0         iso9660     06_13_2024 2024-06-13-22-23-36-00
[dba@mysql-enterprise-db01 ~]$ df -h
Filesystem           Size  Used Avail Use% Mounted on
devtmpfs             1.8G     0  1.8G   0% /dev
tmpfs                1.8G     0  1.8G   0% /dev/shm
tmpfs                1.8G  8.5M  1.8G   1% /run
tmpfs                1.8G     0  1.8G   0% /sys/fs/cgroup
/dev/mapper/rl-root   46G  3.8G   42G   9% /
/dev/sdd1             32G  261M   32G   1% /mysqlbackup
/dev/sdc1             32G  261M   32G   1% /mysqlbinlog
/dev/sdb1             32G  261M   32G   1% /mysqldata
/dev/sda1           1014M  231M  784M  23% /boot
tmpfs                357M     0  357M   0% /run/user/1000
[dba@mysql-enterprise-db01 ~]$
```

once we have setup the separate mount point for the three directory we will now proceed with the installation

the installation file itself will be compressed , we can use `tar -xf` to extract the file

```
[root@mysql-enterprise-db01 dba]# ls
mysql-commercial-backup-8.4.0-1.javascript.1.el8.x86_64.rpm          mysql-connector-j-commercial-8.4.0-1.1.el8.noarch.rpm
mysql-commercial-client-8.4.0-1.javascript.1.el8.x86_64.rpm          mysql-connector-odbc-commercial-8.4.0-1.1.el8.x86_64.rpm
mysql-commercial-client-plugins-8.4.0-1.javascript.1.el8.x86_64.rpm  mysql-connector-odbc-commercial-setup-8.4.0-1.1.el8.x86_64.rpm
mysql-commercial-common-8.4.0-1.javascript.1.el8.x86_64.rpm          mysql-connector-python3-commercial-8.4.0-1.1.el8.x86_64.rpm
mysql-commercial-icu-data-files-8.4.0-1.javascript.1.el8.x86_64.rpm  mysql-enterprise-8.4.0-javascript_el8_x86_64_bundle.tar
mysql-commercial-libs-8.4.0-1.javascript.1.el8.x86_64.rpm            mysql-router-commercial-8.4.0-1.javascript.1.el8.x86_64.rpm
mysql-commercial-server-8.4.0-1.javascript.1.el8.x86_64.rpm          mysql-shell-commercial-8.4.0-1.1.el8.x86_64.rpm
mysql-commercial-test-8.4.0-1.javascript.1.el8.x86_64.rpm            mysqlx-connector-python3-commercial-8.4.0-1.1.el8.x86_64.rpm
mysql-connector-c++-commercial-8.4.0-1.1.el8.x86_64.rpm              README.txt
mysql-connector-c++-commercial-jdbc-8.4.0-1.1.el8.x86_64.rpm         repodata
[root@mysql-enterprise-db01 dba]#
```

we will start by installing `mysql-commercial-backup-8.4.0-1.javascript.1.el8.x86_64.rpm` using `yum localinstall`

```
yum localinstall mysql-commercial-backup-8.4.0-1.javascript.1.el8.x86_64.rpm
```

now snice we found everything is working fine and we are able to install rpm packages on system, lets continue installing the reset of the rpm package

note: you need to install the rpm in the order as showing in below command because some rpm packages relay on the other for dependency

```
yum localinstall mysql-commercial-client-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-commercial-client-plugins-8.4.0-
1.javascript.1.el8.x86_64.rpm mysql-
commercial-common-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-commercial-icu-data-files-8.4.0-
1.javascript.1.el8.x86_64.rpm mysql-commercial-libs-8.4.0-1.javascript.1.el8.x86_64.rpm mysql
-commercial-server-8.4.0-1.javascript.1.el8.x86_64.rpm  mysql-commercial-test-8.4.0-1.javascript.1.el8.x86_64.rpm
```

do the same on db2

```
[root@mysql-enterprise-db01 dba]# yum localinstall mysql-commercial-client-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-commercial-client-plugins-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-
commercial-common-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-commercial-icu-data-files-8.4.0-1.javascript.1.el8.x86_64.rpm mysql-commercial-libs-8.4.0-1.javascript.1.el8.x86_64.rpm mysql
-commercial-server-8.4.0-1.javascript.1.el8.x86_64.rpm  mysql-commercial-test-8.4.0-1.javascript.1.el8.x86_64.rpm
Last metadata expiration check: 0:08:08 ago on Fri 14 Jun 2024 10:32:59 PM +03.
Dependencies resolved.
================================================================================================================================================================================================
 Package                                                Architecture                Version                                                           Repository                         Size
================================================================================================================================================================================================
Installing:
 mysql-commercial-client                                x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                       13 M
 mysql-commercial-client-plugins                        x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      3.8 M
 mysql-commercial-common                                x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      724 k
 mysql-commercial-icu-data-files                        x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      2.3 M
 mysql-commercial-libs                                  x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      1.5 M
 mysql-commercial-server                                x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      112 M
 mysql-commercial-test                                  x86_64                      8.4.0-1.javascript.1.el8                                          @commandline                      334 M
Installing dependencies:
 perl-Carp                                              noarch                      1.42-396.el8                                                      baseos                             29 k
 perl-Data-Dumper                                       x86_64                      2.167-399.el8                                                     baseos                             57 k
 perl-Digest                                            noarch                      1.17-395.el8                                                      appstream                          26 k
 perl-Digest-MD5                                        x86_64                      2.55-396.el8                                                      appstream                          36 k
 perl-Encode                                            x86_64                      4:2.97-3.el8                                                      baseos                            1.5 M
 perl-Errno                                             x86_64                      1.28-422.el8                                                      baseos                             75 k
 perl-Exporter                                          noarch                      5.72-396.el8                                                      baseos                             33 k
 perl-File-Path                                         noarch                      2.15-2.el8                                                        baseos                             37 k
```

editing my.cnf file db01

before we start the services for MySQL we need to update config for MySQL to point binlog and MySQL data to new mount point we created

use any prefeed note editing tool i will be using `vi`

`vi /etc/my.cnf`

update `datadir` and and add `log-bin` and update it with binlog diretcory

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.4/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock
log-bin=/mysqlbinlog/mysql-bin.log
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

updating the owner of mysql directory's db1&db2

you need to update owner of the directory we have setup for MySQL with MySQL user

`chown -R mysql:mysql mysqldata/ chown -R mysql:mysql mysqlbinlog/ chown -R mysql:mysql mysqlbackup/`

```
[root@mysql-enterprise-db01 /]# chown -R mysql:mysql mysqldata/
[root@mysql-enterprise-db01 /]# chown -R mysql:mysql mysqlbinlog/
[root@mysql-enterprise-db01 /]# chown -R mysql:mysql mysqlbackup/
[root@mysql-enterprise-db01 /]# ll
total 20
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 bin -> usr/bin
dr-xr-xr-x.   5 root  root  4096 Jun 14 22:15 boot
drwxr-xr-x.  21 root  root  3240 Jun 14 22:24 dev
drwxr-xr-x. 105 root  root  8192 Jun 14 23:07 etc
drwxr-xr-x.   3 root  root    17 Jun 14 22:12 home
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 lib -> usr/lib
lrwxrwxrwx.   1 root  root     9 Oct 11  2021 lib64 -> usr/lib64
drwxr-xr-x.   2 root  root     6 Oct 11  2021 media
drwxr-xr-x.   2 root  root     6 Oct 11  2021 mnt
drwxr-xr-x.   2 mysql mysql    6 Jun 14 22:22 mysqlbackup
drwxr-xr-x.   2 mysql mysql    6 Jun 14 22:21 mysqlbinlog
drwxr-xr-x.   2 mysql mysql    6 Jun 14 22:20 mysqldata
drwxr-xr-x.   2 root  root     6 Jun 14 22:17 mysqliso
drwxr-xr-x.   2 root  root     6 Oct 11  2021 opt
dr-xr-xr-x. 144 root  root     0 Jun 14 22:24 proc
dr-xr-x---.   2 root  root   135 Jun 14 22:24 root
drwxr-xr-x.  37 root  root  1040 Jun 14 22:42 run
lrwxrwxrwx.   1 root  root     8 Oct 11  2021 sbin -> usr/sbin
drwxr-xr-x.   2 root  root     6 Oct 11  2021 srv
dr-xr-xr-x.  13 root  root     0 Jun 14 22:24 sys
drwxrwxrwt.   7 root  root   161 Jun 14 23:09 tmp
drwxr-xr-x.  12 root  root   144 Jun 14 22:02 usr
drwxr-xr-x.  21 root  root  4096 Jun 14 22:14 var
[root@mysql-enterprise-db01 /]#
```

update host file db01& db02

using any note editing tool and update the host file we hostname and ip of both DB

`vi /etc/hosts`

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
mysql-enterprise-db02   10.217.10.8
mysql-enterprise-db01   10.217.10.7
~
~
~
~
```

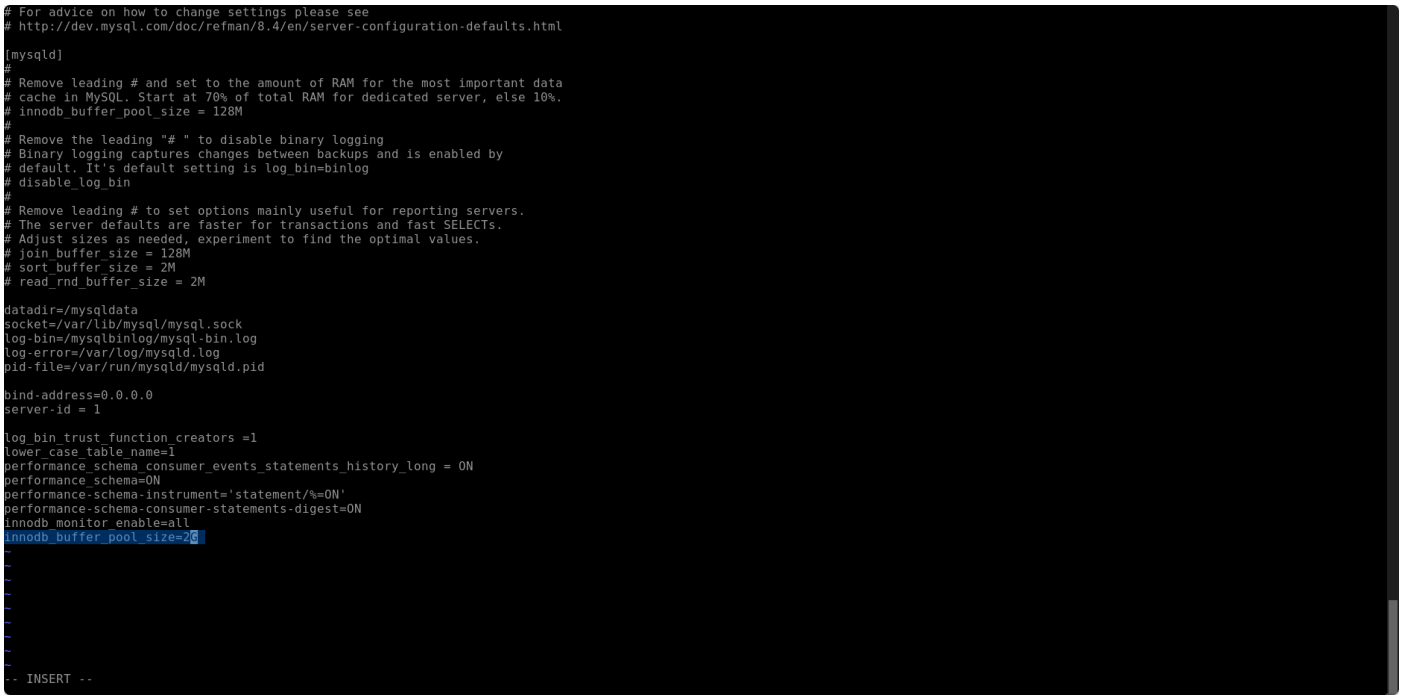
configure master -slave DB01

before we start the up MySQL

we will update my.cnf file with some important variables

- `bin-address` make it equal to `0.0.0.0`
- `server-id` give unique server id for both master and slave

- `log_bin_trust_function_creators =1`
- `lower_case_table_name=1`
- `performance_schema_consumer_events_statements_history_long = ON`
- `performance_schema=ON`
- `performance-schema-instrument='statement/%=ON'`
- `performance-schema-consumer-statements-digest=ON`
- `innodb_monitor_enable=all`
- `innodb_buffer_pool_size=2G` or 70 to 80 % out of the memeory in the OS



below is the full my.cnf file after edititing

```
[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock
log-bin=/mysqlbinlog/mysql-bin.log
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

bind-address=0.0.0.0
server-id = 1

log_bin_trust_function_creators =1
lower_case_table_names=1
performance_schema_consumer_events_statements_history_long = ON
performance_schema=ON
performance-schema-instrument='statement/%=ON'
performance-schema-consumer-statements-digest=ON
```

```
innodb_monitor_enable=all
innodb_buffer_pool_size=2G
~
```

start mysql DB01

to start mysql using the below command

```
mysqld --initialize-insecure --user=mysql --lower_case_table_names=1
```

after that start mysql services

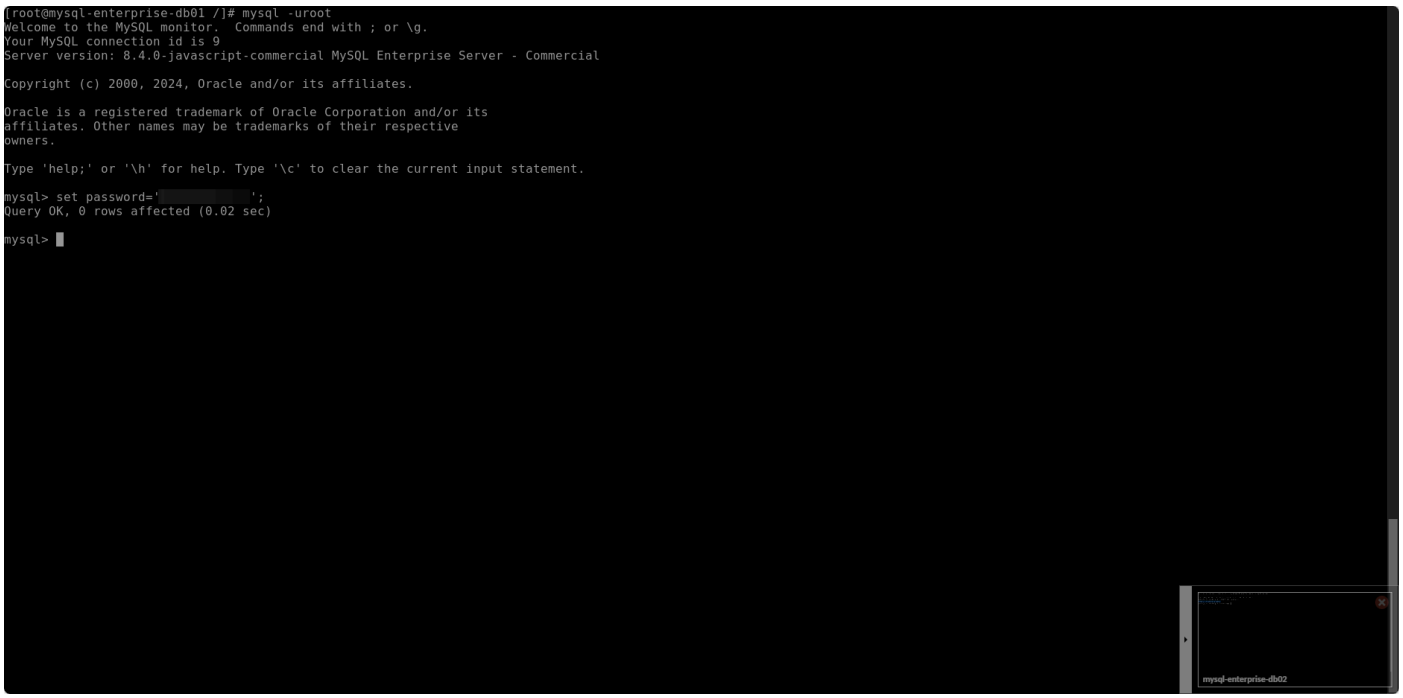
`systemctl start mysqld systemctl enable mysqld`

```
[root@mysql-enterprise-db01 /]# systemctl start mysqld
[root@mysql-enterprise-db01 /]#
```

now you can login to mysql `mysql -uroot`

we will set password for the root user

```
set password='password';
```



`exit` and login again with root using the password you setup

```
mysql> exit
Bye
[root@mysql-enterprise-db01 /]# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.4.0-javascript-commercial MySQL Enterprise Server - Commercial

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

creating replication user DB01

we will create user for replication on db1

```
create user 'repl' identified by 'repl123';
```

```
mysql> create user 'repl' identified by 'repl123';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

give the necessary grants for `repl` user for replication prepose

```
grant replication slave,backup_admin,clone_admin on *.* to 'repl'@'%';
grant select on performance_schema.* to 'repl'@'%';
```

```
mysql> grant replication slave,backup_admin,clone_admin on *.* to 'repl'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> grant select on performance_schema.* to 'repl'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

sysbench oltp_read_write --db-driver=pgsql --pgsql-host=10.10.10.4 --pgsql-user=postgres -tables=10 --table-size=100000 \ --pgsql-db=prodc
ution

sysbench oltp_read_write --db-driver=pgsql --pgsql-host=10.10.10.4 --pgsql-user=postgres --tables=10 --ta
ble-size=100000 --pgsql-db=prodcution prepare

copy my.cnf file from db01 to db02

to save time we will copy the my.cnf file from db01 to db02 using `scp`

`scp -rp /etc/my.cnf root@10.217.10.8:/etc`

```
[root@mysql-enterprise-db01 /]# systemctl enable mysqld
[root@mysql-enterprise-db01 /]# scp -rp /etc/my.cnf root@10.217.10.8:/etc
The authenticity of host '10.217.10.8 (10.217.10.8)' can't be established.
ECDSA key fingerprint is SHA256:EqA9QFtbTpWAlt0uR0S1fb8JZZZhwEXGmJkHxUBfdxo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.217.10.8' (ECDSA) to the list of known hosts.
root@10.217.10.8's password:
my.cnf
[root@mysql-enterprise-db01 /]#
```

```
[root@mysql-enterprise-db02 dba]# cat /etc/my.cnf
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.4/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock
log-bin=/mysqlbinlog/mysql-bin.log
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

bind-address=0.0.0.0
server-id = 1

log_bin_trust_function_creators =1
lower_case_table_names=1
performance_schema_consumer_events_statements_history_long = ON
performance_schema=ON
performance-schema-instrument='statement/%=ON'
performance-schema-consumer-statements-digest=ON
innodb_monitor_enable=all
innodb_buffer_pool_size=2G
[root@mysql-enterprise-db02 dba]#
```

next we will just update `server-id` in my.cnf file to unique one than one setup in db01

`vi /etc/my.cnf`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.4/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock
log-bin=/mysqlbinlog/mysql-bin.log
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

bind-address=0.0.0.0
server-id = 2

log_bin_trust_function_creators =1
lower_case_table_names=1
performance_schema_consumer_events_statements_history_long = ON
performance_schema=ON
performance-schema-instrument='statement/%=ON'
performance-schema-consumer-statements-digest=ON
innodb_monitor_enable=all
innodb_buffer_pool_size=2G
~
~
~
~
~
~
~
~
:wq
```

start mysql DB02

to start mysql using the below command

```
mysqld --initialize-insecure --user=mysql --lower_case_table_names=1
```

after that start mysql services

`systemctl start mysqld systemctl enable mysqld`

```
[root@mysql-enterprise-db02 dba]# mysqld --initialize-insecure --user=mysql --lower_case_table_names=1
[root@mysql-enterprise-db02 dba]# systemctl start mysqld
[root@mysql-enterprise-db02 dba]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2024-06-14 23:44:29 +03; 1min 47s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 1092 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 1577 (mysqld)
   Status: "Server is operational"
    Tasks: 34 (limit: 22592)
   Memory: 717.6M
   CGroup: /system.slice/mysqld.service
           └─1577 /usr/sbin/mysqld

Jun 14 23:44:17 mysql-enterprise-db02 systemd[1]: Starting MySQL Server...
Jun 14 23:44:29 mysql-enterprise-db02 systemd[1]: Started MySQL Server.
[root@mysql-enterprise-db02 dba]#
```

now you can login to mysql `mysql -uroot`

we will set password for the root user

```
set password='password';
```

```
[root@mysql-enterprise-db02 /]# mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.4.0-javascript-commercial MySQL Enterprise Server - Commercial

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> set password='Awersdfzxc.1';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

```
[root@mysql-enterprise-db02 /]# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.4.0-javascript-commercial MySQL Enterprise Server - Commercial

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

install plugins on db02 for cloning

we will start by installing plugin for cloning instance of db01 to db02

```
install plugin clone soname  "mysql_clone.so";
install plugin group_replication soname 'group_replication.so';

set global clone_valid_donor_list='mysql-enterprise-db01:3306';

set global log_error_verbosity=3;
```

```
mysql> install plugin clone soname  "mysql_clone.so";
Query OK, 0 rows affected (0.06 sec)

mysql> install plugin group_replication soname 'group_replication.so';
Query OK, 0 rows affected (0.10 sec)

mysql> set global clone_valid_donor_list='mysql-enterprise-db01:3306';
Query OK, 0 rows affected (0.00 sec)

mysql> set global log_error_verbosity=3;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

on db1

also install the below plugin

```
install plugin clone soname "mysql_clone.so";
```

```
[root@mysql-enterprise-db01 dba]# mys l -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.4.0-javascript-commercial MySQL Enterprise Server - Com

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> install plugin clone soname  "mysql_clone.so";
Query OK, 0 rows affected (0.06 sec)

mysql>
```

now clone the instance using the below command inside mysql console

```
clone instance from 'repl'@'mysql-enterprise-db01':3306 identified by 'repl123';
```

now stop mysqld on db02
and go to the data dir
and remove the `auto.cnf` file

```
[root@mysql-enterprise-db02 /]# systemctl stop mysqld
[root@mysql-enterprise-db02 /]# cd mysqldata/
[root@mysql-enterprise-db02 mysqldata]# ls
 ahmed        ca.pem           '#clone'             ib_buffer_pool  '#innodb_temp'   mysql_upgrade_history   public_key.pem    sys
 auto.cnf     client-cert.pem  '#ib_16384_0.dblwr'   ibdata1         mysql            performance_schema      server-cert.pem   undo_001
 ca-key.pem   client-key.pem   '#ib_16384_1.dblwr'  '#innodb_redo'    mysql.ibd        private_key.pem         server-key.pem    undo_002
[root@mysql-enterprise-db02 mysqldata]# rm -rf auto.cnf
[root@mysql-enterprise-db02 mysqldata]# ls
 ahmed        client-cert.pem  '#ib_16384_0.dblwr'   ibdata1         mysql                 performance_schema   server-cert.pem   undo_001
 ca-key.pem   client-key.pem   '#ib_16384_1.dblwr'  '#innodb_redo'   mysql.ibd             private_key.pem      server-key.pem    undo_002
 ca.pem      '#clone'           ib_buffer_pool      '#innodb_temp'   mysql_upgrade_history  public_key.pem      sys
[root@mysql-enterprise-db02 mysqldata]#
```

then start mysqld on db02

changer master on db02

now we will change the master to db01 on db02

before that we need to get log position on db01
using the below command
`SHOW BINARY LOG STATUS ;`

```
mysql> SHOW BINARY LOG STATUS
    -> ;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000007 |      573 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)

mysql>
```

so the log file is 000007 and postion is 573 this information is important when we start the replica

```
CHANGE MASTER  TO   MASTER_HOST='10.217.10.7', MASTER_USER='repl', MASTER_password='repl123',
MASTER_log_file='mysql-bin.000002', MASTER_log_pos=1903 ;
```

```
mysql> CHANGE REPLICATION SOURCE TO SOURCE_HOST='10.217.10.7', SOURCE_user='repl', SOURCE_password='repl123',  SOURCE_log_file='mysql-bin.000007', SOURCE_log_pos=573 ;
Query OK, 0 rows affected, 2 warnings (0.05 sec)

mysql>
```

now start replica by using the below command

`START REPLICA;`

```
mysql> START REPLICA
    -> ;
Query OK, 0 rows affected (0.02 sec)

mysql>
```

now to verify use the below command

`show replica status\G ;`

```
              Source_Port: 3306
             Connect_Retry: 60
           Source_Log_File: mysql-bin.000007
       Read_Source_Log_Pos: 573
            Relay_Log_File: mysql-enterprise-db02-relay-bin.000002
             Relay_Log_Pos: 328
     Relay_Source_Log_File: mysql-bin.000007
        Replica_IO_Running: Yes
       Replica_SQL_Running: Yes
           Replicate_Do_DB:
       Replicate_Ignore_DB:
        Replicate_Do_Table:
    Replicate_Ignore_Table:
   Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:
                Last_Errno: 0
                Last_Error:
              Skip_Counter: 0
       Exec_Source_Log_Pos: 573
           Relay_Log_Space: 555
           Until_Condition: None
            Until_Log_File:
             Until_Log_Pos: 0
        Source_SSL_Allowed: No
        Source_SSL_CA_File:
        Source_SSL_CA_Path:
           Source_SSL_Cert:
         Source_SSL_Cipher:
            Source_SSL_Key:
      Seconds_Behind_Source: 0
Source_SSL_Verify_Server_Cert: No
              Last_IO_Errno: 0
              Last_IO_Error:
             Last_SQL_Errno: 0
             Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
           Source_Server_Id: 1
                Source_UUID: 9c3b97a8-2a8b-11ef-9bce-fafd554e038e
           Source_Info_File: mysql.slave_master_info
                  SQL_Delay: 0
        SQL_Remaining_Delay: NULL
  Replica_SQL_Running_State: Replica has read all relay log; waiting for more updates
         Source_Retry_Count: 10
                Source_Bind:
    Last_IO_Error_Timestamp:
   Last_SQL_Error_Timestamp:
           Source_SSL_Crl:
       Source_SSL_Crlpath:
```

failover

we will demonstrated how to failover to slave node

SET PRIMARY TO READ-ONLY

```
SET GLOBAL read_only = TRUE; SET GLOBAL event_scheduler = 'OFF'; FLUSH TABLES WITH READ LOCK; SHOW MASTER STATUS;
```

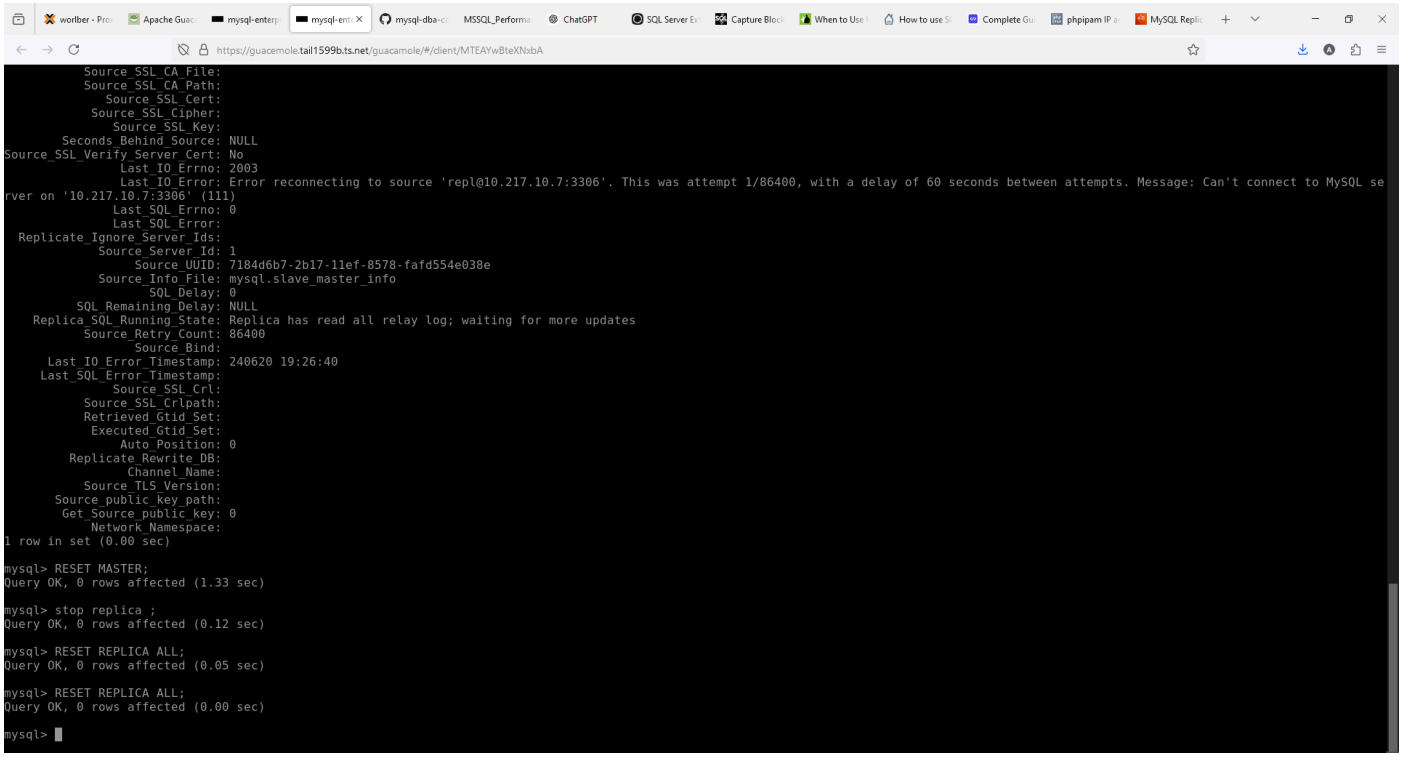
Stop service on old primary

```
systemctl disable mysqld.service systemctl stop mysqld.service
```

```
[root@mysql-enterprise-db01 ~]# systemctl stop  mysqld.service
[root@mysql-enterprise-db01 ~]# systemctl disable  mysqld.service
Removed /etc/systemd/system/multi-user.target.wants/mysqld.service.
[root@mysql-enterprise-db01 ~]#
```

RESET REPLICA ON NEW-PRIMARY

```
SHOW REPLICA STATUS\G
RESET MASTER;
stop replica ;
RESET REPLICA ALL;
SHOW REPLICA STATUS;
```



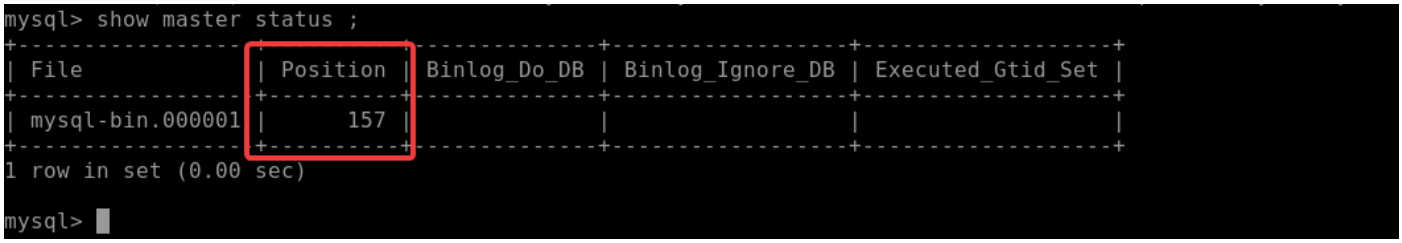
START MYSQL ON OLD-PRIMARY

```
systemctl enable mysqld.service
systemctl start mysqld.service
```

CONFIGURE OLD-PRIMARY TO BECOME REPLICA

before that obtain the log position from the new primary node (old slave)

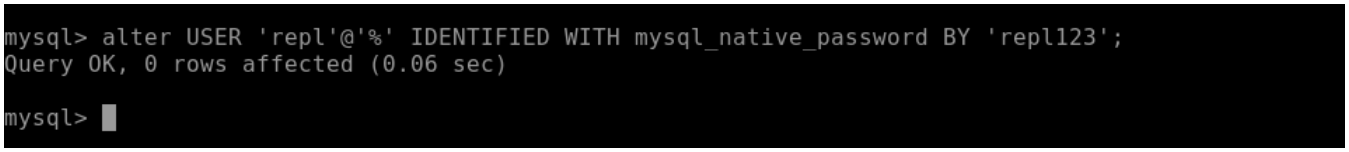
```
show master status ;
```




```
RESET REPLICA ALL;
CHANGE master   TO master_HOST='10.217.10.8', master_USER='repl', master_PASSWORD='repl123', MASTER_log_pos=157 ;
START REPLICA;
SHOW REPLICA STATUS\G;
```

you may get the below erro related to our repl user usingf string*sha2_password*
so we will have to alter user to be saved in mysql+naitave_sql authecating methoid

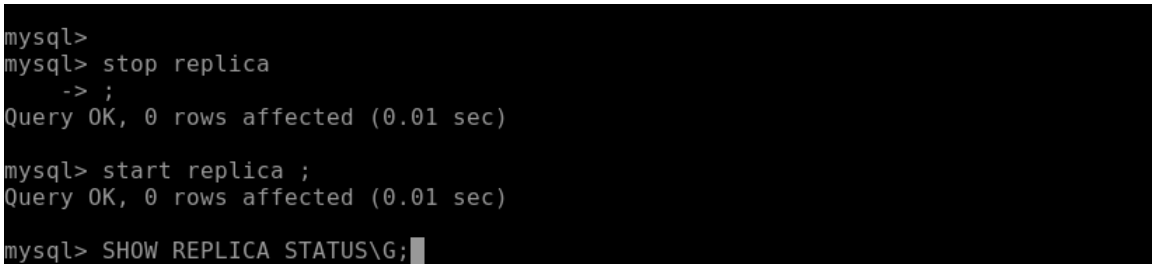

```
alter USER 'repl'@'%' IDENTIFIED WITH mysql_native_password BY 'repl123';
```



stop and start the replica in old primary

```
stop replica

start replica ;
```



```
  Replicate_Wild_Ignore_Table:
                  Last_Errno: 0
                  Last_Error:
                Skip_Counter: 0
         Exec_Source_Log_Pos: 439
             Relay_Log_Space: 834
             Until_Condition: None
              Until_Log_File:
               Until_Log_Pos: 0
          Source_SSL_Allowed: No
          Source_SSL_CA_File:
          Source_SSL_CA_Path:
             Source_SSL_Cert:
           Source_SSL_Cipher:
              Source_SSL_Key:
       Seconds_Behind_Source: 0
Source_SSL_Verify_Server_Cert: No
               Last_IO_Errno: 0
               Last_IO_Error:
              Last_SQL_Errno: 0
              Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Source_Server_Id: 2
                  Source_UUID: 1c0236df-2b19-11ef-993d-b6374bf31f1a
             Source_Info_File: mysql.slave_master_info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
    Replica_SQL_Running_State: Replica has read all relay log; waiting for more updates
          Source_Retry_Count: 86400
                 Source_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
             Source_SSL_Crl:
         Source_SSL_Crlpath:
         Retrieved_Gtid_Set:
          Executed_Gtid_Set:
              Auto_Position: 0
        Replicate_Rewrite_DB:
                Channel_Name:
          Source_TLS_Version:
       Source_public_key_path:
        Get_Source_public_key: 0
            Network_Namespace:
1 row in set (0.00 sec)

ERROR:
No query specified
```

Error is clear you may start testing the replication .

https://www.linkedin.com/in/ahmed-mohamed-423583151