INSTALL MYSQL AND ALLOW REMOTE ACCESS

Update OS: sudo apt update

Upgrade if necessary: sudo apt upgrade

Installing MYSQL (for Kali user) : sudo apt install mariadb-server mariadb-client (This is because kali is for penetration testing and not for DB) Skip this if you are using Ubuntu, Oracle Linux, Debian and CentOS

(Kali users just type mariadb to start your database)

Debian, Ubuntu, Oracle Linux : sudo apt install mysql-server

Check status of MYSQL: sudo systemctl status mysql -- Check if active --

Securing MYSQL: installing MYSQL (for Kali user) : mysql_secure_installation

Securing MYSQL: installing MYSQL (fDebian & Others) : apt install mysql_secure_installation

This will ask you for MYSQL root password that you set during installation

NOTE

Config file	:	/etc/mysql/mysql.conf.d/mysqld.cnf
Data DIR	:	/var/lib/mysql
Log	:	/var/log/mysql/error.log

ALLOW REMOTE ACCESS TO MYSQL

1.) You must create a user that can connect using any Ip address

2.) you must give all necessary privileges to the user

3.) Allow open Ip address to the MYSQL

4.) Open firewall port for user to connect to MYSQL Only for CENTOS

5.) Open Firewall Ip address & and port for user to connect to MYSQL ONLY FOR DEBIAN,UBUNTU & Oracle Linux

6.) To connect MYSQL to DBEAVER you must download mysql connector incase if it doesnt have it

7.) After putting the right credentials in DBEAVER you must go to SSH to put the server root user and password for this to connect

8.) After putting the MYSQL connector @ drivers/libraries @find class ensure it is "com.mysql.jdbc.Driver"

9.) Connect it now

1.) Create a user that can connect to any Ip address: CREATE USER 'promise'@'%' IDENTIFIED BY 'Koko1234$';

2.) Grant superuser privileges to the user: GRANT SUPER ON *.* TO 'promise'@'%';

a.) Grant all rights (SELECT, INSERT, UPDATE, DELETE) on all tables in the 'radius' database

GRANT SELECT, INSERT, UPDATE, DELETE ON radius.* TO 'promise'@'%';

or Grant it all rights to all DB

GRANT SELECT, CREATE, ALTER, DROP, REFERENCES, RELOAD, INSERT, UPDATE, DELETE ON *.* TO 'promise'@'%';

b.) Grant replication rights:

GRANT REPLICATION SLAVE ON *.* TO 'promise'@'%';

c.) Grant all connections

GRANT ALL PRIVILEGES ON *.* TO 'promise'@'localhost' WITH GRANT OPTION; OR (GRANT ALL PRIVILEGES ON *.* TO 'promise'@'%' WITH GRANT OPTION;)

d.) Reload privileges to apply the changes and to free up memory that the server cached as a result of creating user and grant statement

FLUSH PRIVILEGES;

-- Exit the MySQL shell

3.) Allow open Ip address to the MYSQL (Debian (etc/mysql/mysql.conf.d/)) (CENTOS(ONLY): /etc/my.cnf)

nano /etc/mysql/mysql.conf.d/mysqld.cnf change 127.0.0.1 to 0.0.0.0 You can Change port to 1700

NOTE: incase you didnt see it put bind-address = 0.0.0.0

Restart mysql services: systemctl restart mysql.service (CENTOS(ONLY) service restart mysqld.service)

4.) Open firewall port for user to connect to MYSQL Only for CENTOS

nano /etc/iptables

Above pot 22 put this : -A INPUT -p tcp -m state --state NEW,ESTABLISHED -m tcp --dport 3306 -j ACCEPT

5.) Open Firewall Ipadd & and port for user to connect to MYSQL ONLY FOR DEBIAN,UBUNTU & Oracle Linux

Here Firewall can be opened using any command such as FIREWALLD OR UFW

```
            for firewalld: firewall-cmd --zone=public --add-port=1700/tcp --permanent
                           firewall-cmd --add-source=remote_ip_add --permanent
                           firewall-cmd --reload
                           firewall-cmd --list-port          to see all open ports
                           firewall-cmd --list-sources       to see all open ip
```

for ufw: sudo ufw allow from 102.67.30.178 to any port 1700 This allow this Ip to connect using 1700

sudo ufw allow from 102.67.30.178 to any This allows this IP to connect using any port

sudo ufw allow from 192.168.0.0/16 sudo ufw allow 3306/tcp sudo ufw status ufw reload

6.) To connect MYSQL to DBEAVER you must download mysql connector incase if it doesnt have it

7.) After putting the right credentials in DBEAVER you must go to SSH to put the server root user and password for this to connect

8.) After putting the MYSQL connector @ drivers/libraries @find class ensure it is "com.mysql.jdbc.Driver"

9.) Connect it now

BACKUP

Go to the path you wish your backup to be: mysqldump -u username -p database_name > backup_file.sql

RESTORE

Create a fresh database: create database radius

log out and go to the path where the backup is saved: mysql -u username -p new_database_name < backup_file.sql

Same can be used to update existing database: mysql -u username -p new_database_name < backup_file.sql

RESTORE WITH PERCENTAGE VIEWER

To show percentage during restore install the Pipe Viewer and initiate the restore

Install it: sudo apt-get install pv

Unzip gz db displaying percentage: gunzip --stdout Jan-3.sql.gz | pv -ptreb -i 2 > radius_Jan3.sql

Restore: pv radius202310302131.sql | mysql -u promise -p radius

MYSQL REPLICATION

AT Master

Go to Conf : /etc/mysql/mysql.conf.d/mysqld.cnf

1) Edit the mysqld.cnf file to allow all remote connection
2) Uncomment the #server-id   = 1
3) Uncomment the #log_bin   = /var/log/mysql/mysql-bin.log to enable binary login.
4) Uncomment #binlog_do_db   = include_database_name   If you want slave to read a particular DB
5) Restart the database: systemctl restart mysql.serve or service mysql restart
6) Login the DB to creater user for it
   i. CREATE USER ‘rep’@’%’ IDENTIFY BY ‘PASSWORD’
   ii. GRANT REPLICATION SLAVE ON*.* TO ‘rep’@’%’ ;
   iii. Flush tables with read lock;
   iv. SHOW MASTER STATUS \G Note all these below you will paste exactly in slave

      File | Position | Binlog_Do_DB
   v. FLUSH PRIVILEGES;
   vi. Unlock tables;   Not necessary

   Note all these can be written in small letters

ON SLAVE

1. Edit the mysqld.cnf file to allow all remote connection
2. Uncomment the #server-id   = 1
3. Uncomment the #log_bin   = /var/log/mysql/mysql-bin.log to enable binary login.
4. Restart the DB
5. Login to DB

```
CHANGE MASTER TO
  MASTER_HOST = 'MASTER_IP',
  MASTER_PORT = 1700,  -- Specify your desired port number
  MASTER_LOG_FILE = 'mysql-bin.000005',
  MASTER_LOG_POS = 1545,
  MASTER_USER = 'promise',
  MASTER_PASSWORD = 'password',
  MASTER_SSL = 1;
```

Start Replication: start slave

Show Replication status: show slave status \g

Possible Error: If you have already existed Database recreate it on the slave and automatically it will create the tables inside it

MYSQL COMMANDS

Drop User: DROP USER 'kolatoye'@'localhost';

Change root password: alter USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'password';

Create the user to local IP address: CREATE USER 'promise'@'localhost' IDENTIFIED BY 'password';

Create user for any remote IP access: CREATE USER 'promise'@'%' IDENTIFIED BY 'password';

Create the user to connect with DB server IP: CREATE USER 'promise'@'l92.168.11.105' identified by 'password';

Grant superuser privileges: GRANT SUPER ON *.* TO 'promise'@'localhost';

Grant all rights (SELECT, INSERT, UPDATE, DELETE) on all tables in the 'radius' database

GRANT SELECT, INSERT, UPDATE, DELETE ON radius.* TO 'promise'@'localhost';

or

This grant it all rights to all DB: GRANT SELECT, CREATE, ALTER, DROP, REFERENCES, RELOAD, INSERT, UPDATE, DELETE ON *.* TO 'promise'@'%';

Grant replication rights: GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'promise'@'localhost';

Grant all connections: GRANT ALL PRIVILEGES ON *.* TO 'promise'@'localhost' WITH GRANT OPTION;

-- Reload privileges to apply the changes and to free up memory that the server cached as a result of creating user and grant statement: FLUSH PRIVILEGES;

-- Exit the MySQL shell: EXIT;

Display all created users: SELECT user, host FROM mysql.user;

Revoke privileges from the user 'kunle': REVOKE DELETE, DROP, UPDATE ON *.* FROM 'kunle'@'%';

DELET DROP & TRUNCATE TABLE

Delete is to remove a column from the table in the DB. E.g. delete from table2 where id =23

Drop Table: Never run this command unless with signed document with approval: drop table_name

Truncate: This is a form of dropping the table information without dropping the index: truncate table_name

INDEXES

Explain shows all the record and how many it has searched: explain select * from fob;

Show how many indexes has been applied to a particular table. This helps to know if the table needs to be reindexed

Show index from fob. If it shows zero then you can reindex it but if not just know that reindex is not your best option

Apply index: alter table fob add index pf (school) pc means index name which can mean anything

Transfer from Server to Server: scp bakup.sql.gz promise@172.17.2.18:/path/filename.sql.gz

Extract BACKUP FILE:

For (gz file)

Extract without percentage Viewer: gunzip db_full_2023-Oct-30.sql.gz

Extract with percentage Viewer: gunzip -c radius.sql.gz | pv -ptreb -i 2 > radius.20231101.sql

Extract with percentage Viewer: gunzip --stdout radius.20.sql.gz | pv -ptreb -i 2 > radius.20.sq

For (zip)

unzip -p ERP_LIVE202312220925.zip | pv -b -n -i 1 -s $(unzip -l ERP_LIVE202312220925.zip | awk 'END {print $1}') > odoo_erp.sql OR

unzip -p ERP_LIVE202312220925.zip | pv -b -i 1 -s $(unzip -l ERP_LIVE202312220925.zip | awk 'END {print $1}') > odoo_erp.sql

OR my best

unzip -p ERP_LIVE202401090725.zip | pv -pter -i 1 -s $(unzip -l ERP_LIVE202401090725.zip | awk 'END {print $1}') > odoo_erp.sql

MariaDB ON CENT OS

Allow Remote Access:

Log in and create user and grant it all necessary privilege

Go to main conf locate at : nano /usr/share/mysql/my-huge.cnf

Add this line anywhere, preferably below the socket: bind-address = 0.0.0.0