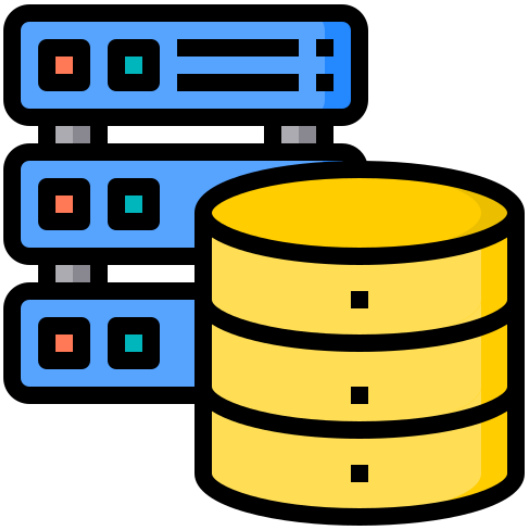

Comprehensive SQL Database Management with MariaDB

Prepared By:
Mahmoud Alsalloum

Table on Contents

Contents

Technology Stack

Ubuntu Server / Client – Lightweight and stable base OS.

MariaDB – Open-source relational database management system for SQL database management.

Bash Scripting – Automating database backups.

Cron – Scheduling automated tasks like backups.

Project Overview

Welcome to my project!

I'm thrilled to share with you the details of my latest work in SQL database management. This project focuses on designing and managing a robust SQL database using MariaDB, hosted on an Ubuntu Server.

The primary goal was to create a reliable and efficient environment for storing and managing user data. To achieve this, I built a user database with a simple yet effective table structure, allowing for seamless data insertion and retrieval. Additionally, I implemented automated backup solutions to ensure data security and reliability, even in the event of unexpected system failures.

This project highlights the importance of efficient database management in maintaining stable and scalable systems. By automating routine tasks like backups and optimizing the database structure, I've laid the groundwork for a system that is not only easy to maintain but also ready for future enhancements.

The entire setup has been thoroughly tested to ensure functionality and reliability, making it a valuable addition to my portfolio of IT projects. I hope you find this project as axcisting and insightful as I did while creating it!

Implementation Steps

The first step in this project was to prepare the server environment for hosting the database. I updated the Ubuntu Server to ensure all packages were up-to-date and secure. I installed MariaDB, a reliable and open-source database management system. See Figure 1.

```
mahmoud@ubuntu-server:~$ sudo apt install mariadb-server -y
```

Figure 1: Install MariaDB

To ensure MariaDB starts automatically after a reboot, I enabled the service. See Figure 2.

```
mahmoud@ubuntu-server:~$ sudo systemctl start mariadb
mahmoud@ubuntu-server:~$ sudo systemctl enable mariadb
Synchronizing state of mariadb.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mariadb
mahmoud@ubuntu-server:~$
```

Figure 2: Start and enable MariaDB

Then, I ran the mysql_secure_installation command to set root password and remove insecure default settings. I created a file named setup_user_database.sql containing the following SQL command. See Figure 3.

```
 GNU nano 7.2                                    setup_user_database.sql *
CREATE DATABASE user_database;


USE user_database;


CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Figure 3: Commands to create SQL database

To verify the fuctionality of the database, I inserted test data into the users table. I created a file named insert_users.sql with the following commands. See Figure 4.

```
 GNU nano 7.2                                    inser_users.sql *
USE user_database;

INSERT INTO users (username, email) VALUES ('MahmoudAlsalloum', 'mahmoud.alsalloum@example.com');
```

Figure 4: Insert new data to the database

I executed the both files. See Figure 5.

```
mahmoud@ubuntu-server:~$ sudo mysql -u root -p < setup_user_database.sql
Enter password:
mahmoud@ubuntu-server:~$ nano inser_users.sql
mahmoud@ubuntu-server:~$ sudo mysql -u root -p < inser_users.sql
Enter password:
mahmoud@ubuntu-server:~$
```

Figure 5: Execute create and insert files

To ensure data security, I automated the backup process. I created a Bash script named backup_user_database.sh. See Figure 6.

```
 GNU nano 7.2                                    backup_user_database.sh *
#!/bin/bash
TIMESTAMP=$(date +"%F")
BACKUP_DIR="/home/backups"
MYSQL_USER="root"
MYSQL_PASSWORD="[redacted]"
DATABASE="user_database"

mkdir -p $BACKUP_DIR
mysqldump -u $MYSQL_USER -p$MYSQL_PASSWORD $DATABASE > $BACKUP_DIR/$DATABASE-$TIMESTAMP.sql
```

Figure 6: Backup script in Bash

I made the script executable by typing this command chmod +x backup_user_database.sh. And then I added a Cron job to run the script daily at 2:00 AM. See Figure 7.

```
 GNU nano 7.2                                    /tmp/crontab.Jk0rEi/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
0 2 * * * /home/backup_user_database.sh
```

Figure 7: Add Cron job for bash script

To ensure the system works as intended, I performed the following tests:

I verified that the database and table were created successfully. See Figure 8.

```
MariaDB [user_database]> SELECT * FROM users;
+----+-----------------+-----------------------------+---------------------+
| id | username        | email                       | created_at          |
+----+-----------------+-----------------------------+---------------------+
|  1 | MahmoudAlsalloum | mahmoud.alsalloum@example.com | 2025-07-03 11:03:01 |
+----+-----------------+-----------------------------+---------------------+
1 row in set (0.001 sec)

MariaDB [user_database]>
```

Figure 8: Verify database

I ran the backup script manually and confirmed the backup file was created. See Figure 9.

```
mahmoud@ubuntu-server:/home/backups$ ls
user_database-2025-07-03.sql
mahmoud@ubuntu-server:/home/backups$
```

Figure 9: Verify backup file