Step 3: Set Up MySQL Database on Lightsail

2026-04-08

Table of contents

1 Overview

Setting up MySQL database on AWS Lightsail for Spring Boot production deployment

- Database installation and security

- User and database creation
- Performance optimization
- Backup configuration
- Spring Boot integration

This lesson covers complete MySQL setup on Lightsail instance, from installation through production configuration.

---

2 Database Options on Lightsail

MySQL on Lightsail Instance - Full control over configuration - No additional charges - Simplified networking - Cost-effective for smaller apps

When to Use RDS - Thousands of concurrent users - High availability requirements - Automated scaling needs - Enterprise-level backup requirements

Lightsail MySQL is perfect for small to medium applications. RDS becomes valuable at scale.

---

3 Installing MySQL

Connect to your Lightsail instance and install MySQL:

```
# Update system packages
sudo apt update && sudo apt upgrade -y

# Install MySQL server and client
sudo apt install mysql-server mysql-client -y
```

Installation takes 2-3 minutes and automatically starts MySQL service

Always start with system updates for security. MySQL installation includes both server and client components.

---

4 Securing MySQL Installation

Run the security script to remove insecure defaults:

```
sudo mysql_secure_installation
```

Configure these settings: {.incremental}

- VALIDATE PASSWORD COMPONENT: **Y**
- Password validation policy: **2 (STRONG)**
- Set strong root password
- Remove anonymous users: **Y**
- Disallow root remote login: **Y**
- Remove test database: **Y**

The security script is crucial for production deployments. Save the root password securely - you'll need it for administration.

---

5 Creating Application Database

Access MySQL and create dedicated resources:

```
-- Create database with proper charset
CREATE DATABASE myapp_prod CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;

-- Create application user
CREATE USER 'springapp'@'localhost'
IDENTIFIED BY 'your-secure-app-password';

-- Grant privileges
GRANT ALL PRIVILEGES ON myapp_prod.* TO 'springapp'@'localhost';
FLUSH PRIVILEGES;
```

Use utf8mb4 charset for full Unicode support. Create dedicated user with minimal required privileges for security.

---

6 Testing Database Setup

Verify your configuration works:

```
-- Switch to new database
USE myapp_prod;

-- Check current user and database
SELECT USER(), DATABASE();

-- Test table operations
CREATE TABLE test_connection (
    id INT AUTO_INCREMENT PRIMARY KEY,
    message VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Always test database operations before proceeding. This ensures connectivity and permissions are correct.

---

7 MySQL Performance Configuration

Edit MySQL configuration file:

```
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

Add performance settings:

```
# Performance tuning
innodb_buffer_pool_size = 256M
innodb_log_file_size = 64M
max_connections = 100

# Character set
character-set-server = utf8mb4
collation-server = utf8mb4_unicode_ci
```

These settings optimize MySQL for small to medium applications on Lightsail instances.

---

8 Apply Configuration Changes

Restart MySQL to apply new settings:

```
# Restart MySQL service
sudo systemctl restart mysql

# Verify service status
sudo systemctl status mysql
```

Status should show “active (running)” in green

Always verify service status after configuration changes. Green “active (running)” indicates successful restart.

---

9 Setting Up Automated Backups

Create backup directory:

```
sudo mkdir -p /backup/mysql
sudo chown mysql:mysql /backup/mysql
```

Create backup script:

```
sudo nano /usr/local/bin/mysql-backup.sh
```

Script includes: - Daily database dumps - 7-day retention policy - Automated cleanup

Automated backups are essential for production databases. This simple script provides basic backup functionality.

---

10 Spring Boot Database Configuration

Update production properties:

```
# Production MySQL configuration
spring.datasource.url=jdbc:mysql://localhost:3306/myapp_prod
spring.datasource.username=springapp
spring.datasource.password=your-secure-app-password
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Production JPA settings
spring.jpa.hibernate.ddl-auto=validate
spring.jpa.show-sql=false
```

Production configuration should use validate mode for ddl-auto and disable SQL logging for performance.

---

11 Security Best Practices

{.incremental}

- Use strong, unique passwords
- Create dedicated application users
- Limit user privileges to required databases
- Keep database local to instance
- Regular security updates
- Implement backup verification

Security should be built in from the start. These practices prevent common security vulnerabilities.

---

12 Summary

Completed Tasks:

- Installed and secured MySQL server
- Created production database and user
- Configured performance optimizations
- Set up automated backup system

- Prepared Spring Boot configuration

Next: Deploy Spring Boot application and connect to MySQL database

Your Lightsail instance now has a production-ready MySQL database configured for optimal performance and security.