**Database Management - Code and Scripts Extracted from Chapter 9**

**1. Installing and Configuring MySQL/MariaDB**

**System Update**

sudo dnf update -y

**Check Available Modules**

sudo dnf module list | grep mysql

**Enable Extra Repositories**

sudo dnf repolist

sudo dnf config-manager --enable appstream

**Regenerate Metadata**

sudo dnf clean all

sudo dnf makecache

**Install MySQL Server**

**From Default Repository**

sudo dnf install mysql-server -y

**From Official MySQL Repository**

sudo dnf install https://dev.mysql.com/get/mysql80-community-release-el8-1.noarch.rpm

sudo dnf config-manager --enable mysql80-community

sudo dnf install mysql-community-server -y

**Verify Installation**

mysql --version

mysql -V

**Start and Enable MySQL Service**

sudo systemctl start mysqld

sudo systemctl enable mysqld

**Basic MySQL Configuration**

**Log in as Root**

mysql -u root -p

**Create Database and User**

CREATE DATABASE example\_db;

CREATE USER 'example\_user'@'localhost' IDENTIFIED BY 'strong\_password';

GRANT ALL PRIVILEGES ON example\_db.\* TO 'example\_user'@'localhost';

FLUSH PRIVILEGES;

**Test Configuration**

mysql -u example\_user -p

SHOW DATABASES;

**2. Managing a Sample Database Schema**

**Create a SQL Script**

sudo nano initialize\_db.sql

**SQL Script Content**

CREATE TABLE employees (

empfullname VARCHAR(50) NOT NULL DEFAULT '',

email VARCHAR(75) NOT NULL DEFAULT '',

PRIMARY KEY (empfullname)

) ENGINE=MyISAM;

INSERT INTO employees VALUES ('admin', 'admin@example.com');

**Run the Script**

mysql -u root -p example\_db < initialize\_db.sql

**3. Database Backup and Restore**

**Backup Entire Database**

mysqldump -u root -p example\_db > example\_db\_backup.sql

**Backup Multiple Databases**

mysqldump -u root -p --databases example\_db my\_database > multiple\_db\_backup.sql

**Restore Database**

mysql -u root -p example\_db < example\_db\_backup.sql

**Automate Backup with Crontab**

crontab -e

**Add the following line:**

0 2 \* \* \* mysqldump -u root -p example\_db > /backup/example\_db\_$(date +\%F).sql

**4. Performance Tuning**

**Optimize MySQL Configuration**

sudo nano /etc/my.cnf

**Add the following lines under [mysqld]:**

innodb\_buffer\_pool\_size = 2G

max\_connections = 300

join\_buffer\_size = 2M

tmp\_table\_size = 128M

max\_heap\_table\_size = 128M

slow\_query\_log = 1

slow\_query\_log\_file = /var/log/mysql/mysql-slow.log

long\_query\_time = 2

**Restart MySQL Service**

sudo systemctl restart mysqld

**Verify Performance Settings**

SHOW VARIABLES LIKE 'innodb\_buffer\_pool\_size';

SHOW VARIABLES LIKE 'max\_connections';

**Analyze and Optimize Queries**

EXPLAIN SELECT \* FROM employees;

OPTIMIZE TABLE employees;

**5. AWS RDS Database Management**

**Create an AWS RDS Instance**

aws rds create-db-instance \

--db-instance-identifier mydbinstance \

--db-instance-class db.t3.micro \

--engine mysql \

--allocated-storage 20 \

--master-username admin \

--master-user-password StrongP@ss123 \

--backup-retention-period 7

**Scale an RDS Instance (Vertical Scaling)**

aws rds modify-db-instance \

--db-instance-identifier mydbinstance \

--db-instance-class db.m6g.large \

--apply-immediately

**Create a Read Replica (Horizontal Scaling)**

aws rds create-db-instance-read-replica \

--db-instance-identifier mydb-replica \

--source-db-instance-identifier mydbinstance \

--db-instance-class db.t3.micro

**Restore from a Snapshot**

aws rds restore-db-instance-from-db-snapshot \

--db-instance-identifier mydbinstance-restore \

--db-snapshot-identifier mydbsnapshot

**Conclusion**

This document extracts the essential code and scripts from Chapter 9, covering MySQL/MariaDB installation, database schema management, backup strategies, performance tuning, and AWS RDS scaling. These scripts provide a structured guide for database administrators working on Rocky Linux and cloud environments.