MySQL 安装与配置指南

> 本文档介绍在 WSL Ubuntu 环境中安装、配置和使用 MySQL 的完整流程

目录

安装 MySQL

1. 更新软件源

```
sudo apt update
sudo apt upgrade -y
```

2. 安装 MySQL Server

```
sudo apt install mysql-server -y
```

3. 验证安装

```
mysql --version
```

示例输出：

```
mysql  Ver 8.0.35-0ubuntu0.22.04.1 for Linux on x86_64 ((Ubuntu))
```

初始配置

1. 启动 MySQL 服务

```
sudo service mysql start
```

或使用 systemctl：

```
sudo systemctl start mysql
```

2. 查看 MySQL 运行状态

```
sudo service mysql status
```

3. 设置开机自启

```
sudo systemctl enable mysql
```

4. 运行安全配置脚本

```
sudo mysql_secure_installation
```

这个命令会引导你完成：

- 设置 root 密码
- 删除匿名用户
- 禁止 root 远程登录
- 删除测试数据库
- 重新加载权限表

用户管理

1. 首次登录 MySQL

```
sudo mysql
```

或者使用密码登录：

```
mysql -u root -p
```

2. 修改 root 用户认证方式

默认情况下，root 用户使用 `auth_socket` 插件，需要修改为密码认证：

```
-- 修改 localhost 访问的 root 用户
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '你的密码';

-- 如果需要远程访问，修改任意主机访问
ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY '你的密码';

-- 刷新权限
FLUSH PRIVILEGES;
```

3. 创建新用户

```
-- 创建本地用户
CREATE USER 'myuser'@'localhost' IDENTIFIED BY 'password123';

-- 创建可远程访问的用户
CREATE USER 'myuser'@'%' IDENTIFIED BY 'password123';

-- 授予所有权限
GRANT ALL PRIVILEGES ON *.* TO 'myuser'@'localhost' WITH GRANT OPTION;

-- 授予特定数据库权限
GRANT ALL PRIVILEGES ON mydb.* TO 'myuser'@'localhost';

-- 刷新权限
FLUSH PRIVILEGES;
```

4. 查看用户列表

```
SELECT User, Host FROM mysql.user;
```

5. 删除用户

```
DROP USER 'myuser'@'localhost';
```

数据库操作

基本命令

```sql
-- 查看所有数据库
SHOW DATABASES;

-- 创建数据库
CREATE DATABASE mydb CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

-- 使用数据库
USE mydb;

-- 查看当前数据库
SELECT DATABASE();

-- 删除数据库
DROP DATABASE mydb;
```

表操作

```sql
-- 查看所有表
SHOW TABLES;

-- 创建表
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE,
    age INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- 查看表结构
DESCRIBE users;
SHOW CREATE TABLE users;

-- 修改表结构
ALTER TABLE users ADD COLUMN phone VARCHAR(20);
ALTER TABLE users MODIFY COLUMN age SMALLINT;
ALTER TABLE users DROP COLUMN phone;

-- 删除表
DROP TABLE users;
```

数据操作 (CRUD)

```sql
-- 插入数据
INSERT INTO users (name, email, age) VALUES ('张三', 'zhangsan@example.com', 25);

INSERT INTO users (name, email, age) VALUES
    ('李四', 'lisi@example.com', 30),
    ('王五', 'wangwu@example.com', 28);

-- 查询数据
SELECT * FROM users;
SELECT name, email FROM users WHERE age > 25;
SELECT * FROM users ORDER BY age DESC LIMIT 10;

-- 更新数据
UPDATE users SET age = 26 WHERE name = '张三';
UPDATE users SET age = age + 1 WHERE age < 30;

-- 删除数据
DELETE FROM users WHERE name = '张三';
DELETE FROM users WHERE age < 20;
```

Python 集成

安装 MySQL 驱动

```bash
# 使用 pip
pip install mysql-connector-python
# 或者
pip install pymysql

# 使用 uv
uv add mysql-connector-python
# 或者
uv add pymysql
```

使用 mysql-connector-python

```python
import mysql.connector
from mysql.connector import Error

def create_connection():
    """创建数据库连接"""
    try:
        connection = mysql.connector.connect(
            host='localhost',
            database='mydb',
            user='root',
            password='your_password'
        )
        if connection.is_connected():
            print("成功连接到 MySQL 数据库")
            return connection
    except Error as e:
        print(f"连接错误: {e}")
        return None

def insert_user(connection, name, email, age):
    """插入用户数据"""
    try:
        cursor = connection.cursor()
        query = "INSERT INTO users (name, email, age) VALUES (%s, %s, %s)"
        cursor.execute(query, (name, email, age))
        connection.commit()
        print(f"成功插入用户: {name}")
    except Error as e:
        print(f"插入错误: {e}")
    finally:
        cursor.close()

def query_users(connection):
    """查询所有用户"""
    try:
        cursor = connection.cursor(dictionary=True)
        cursor.execute("SELECT * FROM users")
        users = cursor.fetchall()
        for user in users:
            print(user)
        return users
```

```
    except Error as e:
        print(f"查询错误: {e}")
    finally:
        cursor.close()

# 使用示例
if __name__ == "__main__":
    conn = create_connection()
    if conn:
        insert_user(conn, "张三", "zhangsan@example.com", 25)
        query_users(conn)
        conn.close()
```

使用 SQLAlchemy (ORM)

```
uv add sqlalchemy pymysql
```

```python
from sqlalchemy import create_engine, Column, Integer, String, DateTime
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker
from datetime import datetime

# 创建数据库引擎
engine = create_engine('mysql+pymysql://root:password@localhost/mydb')
Base = declarative_base()

# 定义模型
class User(Base):
    __tablename__ = 'users'

    id = Column(Integer, primary_key=True, autoincrement=True)
    name = Column(String(100), nullable=False)
    email = Column(String(100), unique=True)
    age = Column(Integer)
    created_at = Column(DateTime, default=datetime.now)

# 创建表
Base.metadata.create_all(engine)

# 创建会话
Session = sessionmaker(bind=engine)
session = Session()

# 插入数据
new_user = User(name="张三", email="zhangsan@example.com", age=25)
session.add(new_user)
session.commit()

# 查询数据
users = session.query(User).filter(User.age > 20).all()
for user in users:
    print(f"{user.name} - {user.email}")

session.close()
```

常用命令

Shell 命令

命令	说明
`sudo service mysql start`	启动 MySQL
`sudo service mysql stop`	停止 MySQL
`sudo service mysql restart`	重启 MySQL
`sudo service mysql status`	查看状态
`mysql -u root -p`	登录 MySQL
`mysqldump -u root -p mydb > backup.sql`	备份数据库
`mysql -u root -p mydb < backup.sql`	恢复数据库

MySQL 命令

命令	说明
`SHOW DATABASES;`	显示所有数据库
`USE database_name;`	切换数据库
`SHOW TABLES;`	显示所有表
`DESCRIBE table_name;`	显示表结构
`SHOW PROCESSLIST;`	显示进程列表
`EXIT;` 或 `\q`	退出 MySQL

故障排查

问题 1：无法启动 MySQL

```
# 查看错误日志
sudo tail -f /var/log/mysql/error.log

# 检查端口占用
sudo lsof -i :3306

# 重启服务
sudo systemctl restart mysql
```

问题 2：root 用户无法登录

```
# 使用 sudo 登录
sudo mysql

# 然后修改认证方式
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '新密码';
FLUSH PRIVILEGES;
```

问题 3：远程连接失败

```
# 编辑配置文件
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

# 修改 bind-address
bind-address = 0.0.0.0

# 重启 MySQL
sudo systemctl restart mysql

# 确保用户允许远程访问
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

问题 4：忘记 root 密码

```
# 1. 停止 MySQL
sudo systemctl stop mysql

# 2. 跳过权限检查启动
sudo mysqld_safe --skip-grant-tables &

# 3. 登录 MySQL
mysql -u root

# 4. 重置密码
FLUSH PRIVILEGES;
ALTER USER 'root'@'localhost' IDENTIFIED BY '新密码';

# 5. 重启 MySQL
sudo systemctl restart mysql
```

性能优化

1. 创建索引

```
-- 创建单列索引
CREATE INDEX idx_email ON users(email);

-- 创建复合索引
CREATE INDEX idx_name_age ON users(name, age);

-- 创建唯一索引
CREATE UNIQUE INDEX idx_unique_email ON users(email);

-- 查看索引
SHOW INDEX FROM users;

-- 删除索引
DROP INDEX idx_email ON users;
```

2. 查询优化

```sql
-- 查看执行计划
EXPLAIN SELECT * FROM users WHERE email = 'test@example.com';

-- 分析表
ANALYZE TABLE users;

-- 优化表
OPTIMIZE TABLE users;
```

最佳实践

1. **定期备份**：使用 `mysqldump` 或自动化脚本备份数据
2. **使用事务**：确保数据一致性
3. **合理索引**：针对常查询字段创建索引
4. **连接池**：在应用中使用数据库连接池
5. **安全配置**：修改默认端口、启用防火墙、使用强密码

相关资源

- MySQL 官方文档
- MySQL Tutorial
- SQLAlchemy 文档
- MySQL Connector Python

最后更新: 2025-10-16