User & Security Management

5.1 MySQL User Accounts and Authentication

User Account Basics

MySQL user accounts consist of two parts:

- **Username**: The login name
- **Host**: The client host that can connect

-- *User format: 'username'@'host'*

'john'@'localhost' -- *Local connections only*

'john'@'192.168.1.100' -- *Specific IP*

'john'@'192.168.1.%' -- *Wildcard (any host in 192.168.1.*)*

'john'@'%' -- *Any host (not recommended for admin users)*

Creating Users

Basic User Creation:

-- *Create local user with password*

CREATE USER 'developer'@'localhost' IDENTIFIED BY 'SecurePass123!';

-- *Create user with specific host*

CREATE USER 'analyst'@'192.168.1.100' IDENTIFIED BY 'AnalystPass123!';

-- *Create user from any host*

CREATE USER 'app_user'@'%' IDENTIFIED BY 'AppPass123!';

-- *Create multiple users at once*

CREATE USER

```
    'user1'@'localhost' IDENTIFIED BY 'Pass1',
    'user2'@'localhost' IDENTIFIED BY 'Pass2',
    'user3'@'localhost' IDENTIFIED BY 'Pass3';

-- Verify creation
SELECT user, host FROM mysql.user WHERE user LIKE 'developer%';
```

User Authentication Methods

Native Password (MySQL 5.x - 8.0.3):

```
CREATE USER 'user1'@'localhost'
    IDENTIFIED WITH mysql_native_password
    BY 'password123';
```

SHA2 Password (MySQL 8.0+ default):

```
CREATE USER 'user2'@'localhost'
    IDENTIFIED WITH caching_sha2_password
    BY 'password123';

-- Requires --get-server-public-key or ~/.my.cnf with public key
```

No Authentication (not recommended):

```
CREATE USER 'guest'@'localhost';
-- User can connect without password
```

5.2 Privilege System

Types of Privileges

Global Privileges (granted on *.*):

-- Server administration

SUPER, PROCESS, RELOAD, SHUTDOWN

-- Database operation

FILE, REPLICATION_SLAVE, REPLICATION_CLIENT

-- All global privileges

GRANT ALL PRIVILEGES ON *.* TO 'user'@'host';

Database Privileges (granted on database.*):

-- Database-level permissions

CREATE, DROP, ALTER, INDEX, LOCK TABLES

-- Data operation

SELECT, INSERT, UPDATE, DELETE

-- Procedures and functions

CREATE ROUTINE, ALTER ROUTINE, EXECUTE

Table Privileges (granted on database.table):

SELECT, INSERT, UPDATE, DELETE

ALTER, INDEX, CREATE, DROP

TRIGGER, REFERENCES

Column Privileges (granted on database.table.column):

-- Limited privileges on specific columns

SELECT (column_list)

UPDATE (column_list)

Granting Privileges

Grant All Database Privileges:

GRANT ALL PRIVILEGES ON database_name.* TO 'user'@'host';

Grant Specific Privileges:

-- SELECT and INSERT only

GRANT SELECT, INSERT ON database_name.* TO 'user'@'host';

-- Global privileges

GRANT RELOAD, REPLICATION SLAVE ON *.* TO 'repl_user'@'host';

-- Table-specific privileges

GRANT SELECT, INSERT, UPDATE

ON database_name.table_name

TO 'user'@'host';

-- Column-specific privileges

GRANT SELECT (id, name, email), UPDATE (email)

ON database_name.users

TO 'user'@'host';

Grant with GRANT OPTION:

```
-- Allows user to grant privileges to others
GRANT ALL PRIVILEGES ON database_name.*
   TO 'admin'@'host'
   WITH GRANT OPTION;
```

Revoking Privileges

```
-- Revoke specific privilege
REVOKE SELECT ON database_name.* FROM 'user'@'host';

-- Revoke all privileges
REVOKE ALL PRIVILEGES ON database_name.* FROM 'user'@'host';

-- Revoke GRANT OPTION
REVOKE GRANT OPTION ON *.* FROM 'user'@'host';
```

Viewing Privileges

```
-- View current user privileges
SHOW GRANTS FOR CURRENT_USER();

-- View specific user privileges
SHOW GRANTS FOR 'user'@'host';

-- Query privilege tables directly
SELECT * FROM mysql.user
   WHERE user = 'username'\G
```

```
SELECT * FROM mysql.db
   WHERE user = 'username'\G

SELECT * FROM mysql.tables_priv
   WHERE user = 'username'\G
```

Privilege Hierarchy

Global (*.*)

↓

Database (database.*)

↓

Table (database.table)

↓

Column (database.table.column)

If privilege granted at higher level, not needed at lower level.

5.3 User Management

Renaming Users

-- Rename user (MySQL 5.7.7+)

```
RENAME USER 'old_name'@'host' TO 'new_name'@'host';
```

-- Change host

```
RENAME USER 'user'@'localhost' TO 'user'@'192.168.1.100';
```

-- Verify

```
SELECT user, host FROM mysql.user WHERE user = 'new_name';
```

Changing Passwords

-- Alter user password

```
ALTER USER 'user'@'host' IDENTIFIED BY 'NewPassword123!';
```

-- Self-changing password

```
ALTER USER 'current_user'@'localhost' IDENTIFIED BY 'NewPass123!';
```

-- Using SET PASSWORD (older method, still works)

```
SET PASSWORD FOR 'user'@'host' = 'NewPassword123!';
```

Dropping Users

-- Drop single user

```
DROP USER 'user'@'host';
```

-- Drop multiple users

```
DROP USER 'user1'@'localhost', 'user2'@'host';
```

```
-- Drop if exists (MySQL 5.7.4+)
DROP USER IF EXISTS 'user'@'host';

-- Verify
SELECT user, host FROM mysql.user WHERE user = 'user_name';
```

Resource Limits

Control resource usage per user:

```
-- Create user with resource limits
CREATE USER 'limited_user'@'localhost'
    IDENTIFIED BY 'LimitPass123!'
    WITH
        MAX_QUERIES_PER_HOUR 1000
        MAX_UPDATES_PER_HOUR 500
        MAX_CONNECTIONS_PER_HOUR 10
        MAX_USER_CONNECTIONS 2;

-- Modify limits for existing user
ALTER USER 'user'@'host'
    WITH
        MAX_QUERIES_PER_HOUR 2000
        MAX_CONNECTIONS_PER_HOUR 20;

-- Remove limits
ALTER USER 'user'@'host'
    WITH UNLIMITED;
```

```
-- Check current resource usage
SHOW STATUS LIKE 'Queries';
SHOW STATUS LIKE 'Max_used_connections';
```

Monitoring User Connections

```
-- View active connections
SHOW PROCESSLIST;

-- More details
SELECT
   ID,
   USER,
   HOST,
   DB,
   COMMAND,
   TIME,
   ROWS_SENT,
   ROWS_EXAMINED
FROM INFORMATION_SCHEMA.PROCESSLIST;

-- Kill specific connection
KILL CONNECTION process_id;

-- Kill specific query
KILL QUERY process_id;
```

5.4 SSL-Based Connections

Checking SSL Support

-- Verify SSL support

SHOW VARIABLES LIKE 'have_ssl'; *-- Should be YES*

-- View SSL settings

SHOW VARIABLES LIKE '%ssl%';

Setting Up SSL

Generate SSL Certificates:

 # Automatic (MySQL 5.7.6+)

sudo mysql_ssl_rsa_setup --uid=mysql

 # Manual using OpenSSL

openssl genrsa 2048 > ca-key.pem

openssl req -new -x509 -nodes -days 365000 -key ca-key.pem > ca.pem

openssl req -newkey rsa:2048 -days 365000 -nodes -keyout server-key.pem > server-req.pem

openssl x509 -req -in server-req.pem -days 365000 -CA ca.pem -CAkey ca-key.pem > server-cert.pem

Configure MySQL:

[mysqld]

ssl_ca = /var/lib/mysql/ca.pem

ssl_cert = /var/lib/mysql/server-cert.pem

ssl_key = /var/lib/mysql/server-key.pem

Requiring SSL for Users

```
-- User must use SSL
CREATE USER 'secure_user'@'%' IDENTIFIED BY 'SecurePass123!'
    REQUIRE SSL;

-- Require specific cipher
CREATE USER 'cipher_user'@'%' IDENTIFIED BY 'Pass123!'
    REQUIRE CIPHER 'DHE-RSA-AES128-GCM-SHA256';

-- Require X.509 certificate
CREATE USER 'cert_user'@'%' IDENTIFIED BY 'Pass123!'
    REQUIRE X509;

-- Require specific certificate and key
CREATE USER 'cert_user2'@'%' IDENTIFIED BY 'Pass123!'
    REQUIRE SUBJECT '/C=US/ST=California/O=MyCompany/CN=user'
    ISSUER '/C=US/ST=California/O=MyCompany/CN=ca'
    CIPHER 'DHE-RSA-AES128-GCM-SHA256';
```

Connecting with SSL

```
# Basic SSL connection
mysql -u secure_user -p \
    -h mysql.example.com \
    --ssl-mode=REQUIRED
```

```
# With certificate verification
mysql -u secure_user -p \
    -h mysql.example.com \
    --ssl-ca=/path/to/ca.pem \
    --ssl-mode=VERIFY_IDENTITY

# SSL modes: DISABLED, PREFERRED, REQUIRED, VERIFY_CA, VERIFY_IDENTITY
```

Verifying SSL Connection

```
-- Check current connection SSL status
SHOW STATUS LIKE 'ssl_version';
SHOW STATUS LIKE 'ssl_cipher';

-- In mysql client
\s  -- Displays connection info including SSL details
```

5.5 Password Policies

Validate Password Plugin

-- Install plugin

INSTALL PLUGIN validate_password SONAME 'validate_password.so';

-- Verify installation

SHOW PLUGINS LIKE 'validate_password';

-- Check policy settings

SHOW VARIABLES LIKE 'validate_password%';

Password Policy Configuration

[mysqld]

validate-password.policy = 2

 # 0 = LOW (length >= 8)

 # 1 = MEDIUM (length >= 8, mixed case, digits, special chars)

 # 2 = STRONG (length >= 8, mixed case, digits, special chars, no dictionary)

validate-password.length = 12

validate-password.mixed_case_count = 1

validate-password.number_count = 1

validate-password.special_char_count = 1

Setting Passwords with Policy

-- This will fail - too simple

CREATE USER 'weak_user'@'localhost' IDENTIFIED BY 'simple';

-- Error: Password does not satisfy the current policy requirements

-- This will succeed - meets policy

```
CREATE USER 'strong_user'@'localhost' IDENTIFIED BY 'StrongPass123!@#';
```

5.6 Best Practices for User Security

1. **Use Strong Passwords**: Follow policy requirements
2. **Principle of Least Privilege**: Grant only necessary privileges
3. **Host Restrictions**: Limit access to specific hosts
4. **SSL Connections**: Use for remote connections
5. **Remove Default Users**: Delete 'root'@'%' and anonymous users
6. **No 'root' for Applications**: Create specific application users
7. **Monitor Connections**: Track user activity
8. **Regular Audits**: Review user privileges periodically

```
-- Security audit queries

-- Find users with no password

SELECT user, host FROM mysql.user WHERE authentication_string = '';


-- Find users with overly broad privileges

SELECT user, host FROM mysql.user WHERE Super_priv = 'Y';


-- Find users without specific host restrictions

SELECT user, host FROM mysql.user WHERE host = '%';
```

5.7 Summary: Day 5 Key Takeaways

1. **User Format**: 'username'@'host' determines connection source
2. **Authentication**: Native password, SHA2 password, or no authentication
3. **Privileges**: Global, database, table, and column levels
4. **Resource Limits**: Control per-user resource consumption
5. **SSL Security**: Encrypt client-server communication
6. **Password Policies**: Enforce strong password requirements
7. **Best Practices**: Least privilege, strong passwords, host restrictions