

## Database Security Lab

These 5 experiments cover all **major areas of database security**:

1. Access control
2. Injection attacks
3. Auditing
4. Encryption
5. Role-based access control

### 1. User Management with GRANT and REVOKE

**Objective:** To study database access control using user roles and privileges.

#### a) Create Users

```
CREATE USER 'student1'@'localhost' IDENTIFIED BY 'pass123';
CREATE USER 'student2'@'localhost' IDENTIFIED BY 'pass123';
```

#### b) Grant Privileges

```
GRANT ALL PRIVILEGES ON labdb.* TO 'student1'@'localhost';
GRANT SELECT, INSERT ON labdb.* TO 'student2'@'localhost';
```

#### c) Revoke Privilege

```
REVOKE INSERT ON labdb.* FROM 'student2'@'localhost';
```

#### d) Verify

```
SHOW GRANTS FOR 'student1'@'localhost';
SHOW GRANTS FOR 'student2'@'localhost';
```

#### Output:

a)

```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0279 seconds.)  
CREATE USER 'student1'@'localhost' IDENTIFIED BY 'pass123';  
  
Edit inline      Edit      Create PHP code  
  
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0255 seconds.)  
CREATE USER 'student2'@'localhost' IDENTIFIED BY 'pass123';  
  
Edit inline      Edit      Create PHP code
```

b)

```
✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0334 seconds.)  
GRANT ALL PRIVILEGES ON labdb.* TO 'student1'@'localhost';  
  
Edit inline      Edit      Create PHP code
```

  

```
✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0274 seconds.)  
GRANT SELECT, INSERT ON labdb.* TO 'student2'@'localhost';  
  
Edit inline      Edit      Create PHP code
```

c)

```
✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0394 seconds.)  
REVOKE INSERT ON labdb.* FROM 'student2'@'localhost';  
  
Edit inline      Edit      Create PHP code
```

d)

Your SQL query has been executed successfully.

```
SHOW GRANTS FOR 'student1'@'localhost';  
  
 Profiling [ Edit inline ] [ Edit ] [ Create PH ]  
  
Extra options
```

**Grants for student1@localhost**

```
GRANT USAGE ON ** TO 'student1'@'localhost' IDENT...  
GRANT ALL PRIVILEGES ON `labdb`.* TO 'student1'@'l...  
  
SHOW GRANTS FOR 'student2'@'localhost';  
  
Extra options
```

**Grants for student2@localhost**

```
GRANT USAGE ON ** TO 'student2'@'localhost' IDENT...  
GRANT SELECT ON `labdb`.* TO 'student2'@'localhost...'
```

## 2. Experiment SQL Injection Demonstration

**Objective:** To demonstrate SQL injection and its prevention.

```
CREATE DATABASE labdb;
USE labdb;
```

```
CREATE TABLE users (
id INT AUTO_INCREMENT PRIMARY KEY,
username VARCHAR(50),
password VARCHAR(50)
);
```

```
INSERT INTO users (username, password) VALUES
('admin', 'admin123'),
('student', 'stud123');
```

### Injection Example:

Input:

Username: admin

Password: ' OR '1'='1

Query becomes:

```
SELECT * FROM users WHERE username='admin' AND password=" OR '1'='1';
```

Bypasses authentication.

Prevention: Use prepared statements (parameterized queries)

The screenshot shows a sequence of four successful MySQL queries:

- CREATE DATABASE labdb12;
- USE labdb12;
- CREATE TABLE users ( id INT AUTO\_INCREMENT PRIMARY KEY, username VARCHAR(50), password VARCHAR(50) );
- INSERT INTO users (username, password) VALUES ('admin', 'admin123'), ('student', 'stud123');

Each query is followed by a success message and execution time.

### 3. Experiment Database Auditing

**Objective:** To enable logging and track changes to the database.

#### Procedure:

-- Enable general log (MySQL)

```
SET GLOBAL general_log = 'ON';
SET GLOBAL general_log_output = 'TABLE';
```

-- View logged queries

```
SELECT * FROM mysql.general_log ORDER BY event_time DESC LIMIT 10;
```

#### Output:

| event_time  | user_host                                       | thread_id | server_id | command_type | argument                      |
|---|---|-----------|-----------|--------------|-------------------------------|
| MySQL returned an empty result set (i.e. zero rows). (Query took 0.0001 seconds.)   |   |           |           |              |                               |
| -- Enable logging SET GLOBAL general_log = ON;  |   |           |           |              | -- View logs                  |
| [Edit inline] [Edit] [Create PHP code]  |   |           |           |              |                               |
| 2025-11-14<br>16:02:35.113327   | 11239A023[11239A023] @ localhost<br>[127.0.0.1] | 2709      | 1         | Query        | SELECT *<br>*                 |
| 2025-11-14<br>16:02:35.113189   | 11239A023[11239A023] @ localhost<br>[127.0.0.1] | 2709      | 1         | Init DB      | mysql                         |
| MySQL returned an empty result set (i.e. zero rows). (Query took 0.0000 seconds.)   |   |           |           |              |                               |
| SET GLOBAL log_output = 'TABLE';  |   |           |           |              | -- SELECT<br>DATABASE()<br>() |
| [Edit inline] [Edit] [Create PHP code]  |   |           |           |              |                               |
| ⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available. ⓘ        |   |           |           |              |                               |
| Showing rows 0 - 9 (10 total). Query took 0.1605 seconds. [event_time: 2025-11-14 16:02:35.113327... - 2025-11-14 16:02:35.106134...] |   |           |           |              |                               |
| -- View logs SELECT * FROM mysql.general_log ORDER BY event_time DESC LIMIT 10;   |   |           |           |              | SHOW SESSION VARIABLES        |
| [Edit inline] [Edit] [Create PHP code]  |   |           |           |              |                               |
| 2025-11-14<br>16:02:35.112103   | 11239A023[11239A023] @ localhost<br>[127.0.0.1] | 2709      | 1         | Query        | SESSION                       |
| 2025-11-14<br>16:02:35.111494   | 11239A023[11239A023] @ localhost<br>[127.0.0.1] | 2709      | 1         | Query        | VARIABLES                     |
| SHOW SESSION VARIABLES  |   |           |           |              |                               |

#### 4. Experiment 4: Encryption in Database

**Objective:** To secure data using encryption functions.

a) CREATE TABLE secure\_data (

```
id INT AUTO_INCREMENT PRIMARY KEY,  
secret VARBINARY(255)  
);
```

b) Insert encrypted data (AES)

```
INSERT INTO secure_data(secret) VALUES (AES_ENCRYPT('mysecretpassword', 'key123'));
```

c) Decrypt data

```
SELECT AES_DECRYPT(secret, 'key123') AS decrypted_value FROM secure_data;
```

#### Output:

The screenshot shows the MySQL Workbench interface with three main sections of output:

- Query Results:** Shows the creation of the table "secure\_data\_expt4" and the insertion of a single row with ID 1 and secret value 'mysecretpassword'.
- Logs:** Shows the SQL queries used for each step: creating the table, inserting the data, and selecting it.
- Table Data:** Shows the single row of data in the "secure\_data\_expt4" table, with the decrypted value displayed as "mysecretpassword".

## 5: Role-Based Access Control (RBAC)

**Objective:** To implement RBAC in a database.

**Procedure:**

a) Create a role

```
CREATE ROLE 'manager';
```

b) Assign privileges to role

```
GRANT SELECT, UPDATE ON labdb.* TO 'manager';
```

c) Create user and assign role

```
CREATE USER 'alice'@'localhost' IDENTIFIED BY 'alice123';
GRANT 'manager' TO 'alice'@'localhost';
```

d) Verify

```
SHOW GRANTS FOR 'alice'@'localhost';
```

**Output:**

a)

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0539 seconds.)

```
CREATE ROLE 'manager';
```

Edit inline

Edit

Create PHP code

b)

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0288 seconds.)

```
GRANT SELECT, UPDATE ON labdb.* TO 'manager';
```

Edit inline

Edit

Create PHP code

c)

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0264 seconds.)

```
CREATE USER 'alice'@'localhost' IDENTIFIED BY 'alice123';
```

[Edit inline](#)

[Edit](#)

[Create PHP code](#)

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0394 seconds.)

```
GRANT 'manager' TO 'alice'@'localhost';
```

[Edit inline](#)

[Edit](#)

[Create PHP code](#)

d)

✓ Your SQL query has been executed successfully.

```
SHOW GRANTS FOR 'alice'@'localhost';
```

Profiling

[Edit inline](#)

[Edit](#)

[Create PHP code](#)

[Refresh](#)

[Extra options](#)

#### Grants for alice@localhost

```
GRANT USAGE ON *.* TO `alice`@`localhost`
```

```
GRANT `manager`@`%` TO `alice`@`localhost`
```