



SWINBURNE  
UNIVERSITY OF  
TECHNOLOGY

**COS20031**

Computing Technology Design Project

**Week 10-11: Major-specific topics**

**Cybersecurity (Database)**





---

## (A) Grant and Revoke Permission

---

# GRANT PERMISSION



- The **CREATE USER** statement creates one or more user accounts with no privileges. It means that the user accounts can log in to the MySQL Server, but cannot do anything such as selecting a database and querying data from tables.
- To allow user accounts to work with database objects, you need to grant the user accounts privileges. And the **GRANT** statement grants a user account one or more privileges.

# GRANT

Allows a principal to perform actions on a securable



# GRANT EXAMPLE 1

The following illustrates the basic syntax of the GRANT statement:

```
GRANT privilege [,privilege],..  
ON privilege_level  
TO account_name;
```

This example grants the SELECT privilege on the table employees in the sample database to the user account bob@localhost:

```
GRANT SELECT  
ON employees  
TO bob@localhost;
```

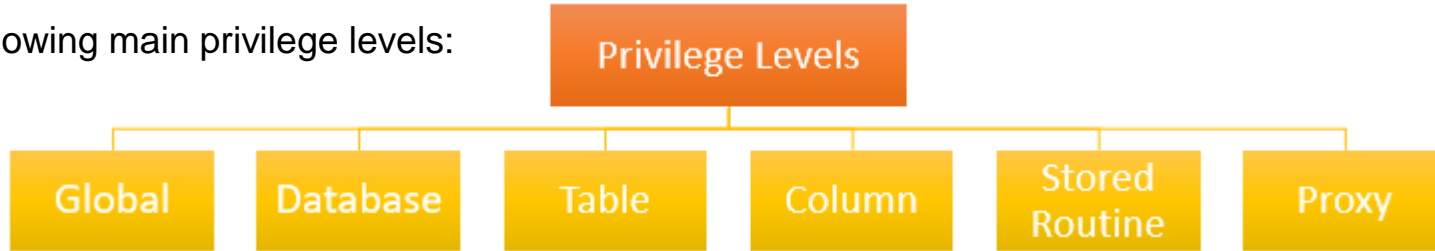


# GRANT EXAMPLE 2

The following example grants UPDATE, DELETE, and INSERT privileges on the table employees to bob@localhost:

```
GRANT INSERT, UPDATE, DELETE  
ON employees  
TO bob@localhost;
```

The following main privilege levels:



Global privileges apply to all databases in a MySQL Server. To assign global privileges, you use the \*.\* syntax, for example:

```
GRANT SELECT  
ON *.*  
TO bob@localhost;
```

# GRANT PRIVILEGES ON FUNCTIONS/PROCEDURES



When dealing with functions and procedures, you can grant users the ability to EXECUTE these functions and procedures in MySQL.

## Syntax

The syntax for granting EXECUTE privileges on a function/procedure in MySQL is:

```
GRANT EXECUTE ON [ PROCEDURE | FUNCTION ] object TO user;
```

## EXECUTE

The ability to execute the function or procedure.

## PROCEDURE

It is used when the privilege is being granted on a procedure in MySQL.

## FUNCTION

It is used when the privilege is being granted on a function in MySQL.

## object

The name of the database object that you are granting privileges for. In the case of granting EXECUTE privileges on a function or procedure, this would be the function name or the procedure name.

## user

The name of the user that will be granted the EXECUTE privileges.



## Grant Example 2: Grant Privileges on Functions/Procedures

For example, if you had a function called *CalcIncome* and you wanted to grant EXECUTE access to the user named *smithj*, you would run the following GRANT statement:

```
GRANT EXECUTE ON FUNCTION CalcIncome TO 'smithj'@'localhost';
```

If you wanted to grant ALL users the ability to EXECUTE this function, you would run the following GRANT statement:

```
GRANT EXECUTE ON FUNCTION CalcIncome TO '*'@'localhost';
```

### Example - Procedure

Let's look at some examples of how to grant EXECUTE privileges on a procedure in MySQL.

For example, if you had a procedure called *MyFirstProc* and you wanted to grant EXECUTE access to the user named *smithj*, you would run the following GRANT statement:

```
GRANT EXECUTE ON PROCEDURE MyFirstProc TO 'smithj'@'localhost';
```



# REVOKE PERMISSION

- The REVOKE statement revokes one or more privileges from a user account.
- The following illustrates the basic syntax of the REVOKE statement that revokes one or more privileges from user accounts:

```
REVOKE  
    privilege [, privilege]..  
ON [object_type] privilege_level  
FROM user1 [, user2] ..;
```

## Revoke

Prevents a principal from performing actions on a securable





# Revoke Privileges on Table

## Syntax

The syntax for revoking privileges on a table in MySQL is:

```
REVOKE privileges ON object FROM user;
```

## privileges

It can be any of the following values:

Privilege	Description
SELECT	Ability to perform SELECT statements on the table.
INSERT	Ability to perform INSERT statements on the table.
UPDATE	Ability to perform UPDATE statements on the table.
DELETE	Ability to perform DELETE statements on the table.
INDEX	Ability to create an index on an existing table.
CREATE	Ability to perform CREATE TABLE statements.
ALTER	Ability to perform ALTER TABLE statements to change the table definition.
DROP	Ability to perform DROP TABLE statements.
GRANT OPTION	Allows you to grant the privileges that you possess to other users.
ALL	Grants all permissions except GRANT OPTION.

Once you have granted privileges, you may need to revoke some or all of these privileges. To do this, you can run a revoke command. You can revoke any combination of SELECT, INSERT, UPDATE, DELETE, REFERENCES, ALTER, or ALL.



# REVOKE EXAMPLE 1

For example, if you wanted to revoke DELETE and UPDATE privileges on a table called *contacts* from a user named *smithj*, you would run the following REVOKE statement:

```
REVOKE DELETE, UPDATE ON contacts FROM 'smithj'@'localhost';
```

If you wanted to revoke all permissions (except GRANT OPTION) on a table for a user named *smithj*, you could use the ALL keyword as follows:

```
REVOKE ALL ON contacts FROM 'smithj'@'localhost';
```

If you had granted SELECT privileges to \* (ie: all users) on the *contacts* table and you wanted to revoke these privileges, you could run the following REVOKE statement:

```
REVOKE SELECT ON contacts FROM '*'@'localhost';
```



# REVOKE PRIVILEGES ON FUNCTIONS/PROCEDURES

Once you have granted EXECUTE privileges on a function or procedure, you may need to REVOKE these privileges from a user in MySQL. To do this, you can execute a REVOKE command.

## Syntax

The syntax for the revoking privileges on a function or procedure in MySQL is:

```
REVOKE EXECUTE ON [ PROCEDURE | FUNCTION ] object FROM user;
```

## EXECUTE

The ability to execute the function or procedure is being revoked.

## PROCEDURE

It is used when the privilege is being revoked on a procedure in MySQL.

## FUNCTION

It is used when the privilege is being revoked on a function in MySQL.

## object

The name of the database object that you are revoking privileges for. In the case of revoking EXECUTE privileges on a function or procedure, this would be the function name or the procedure name.

## user

The name of the user that will be revoked the EXECUTE privileges.



## Revoke Example 2: Revoke Privileges on Functions/Procedures

If you wanted to revoke EXECUTE privileges on a function called *CalcIncome* from a user named *smithj*, you would run the following REVOKE statement:

```
REVOKE EXECUTE ON FUNCTION CalcIncome FROM 'smithj'@'localhost';
```

If you had granted EXECUTE privileges to \* (all users) on the function called *CalcIncome* and you wanted to revoke these EXECUTE privileges, you could run the following REVOKE statement:

```
REVOKE EXECUTE ON FUNCTION CalcIncome FROM '*'@'localhost';
```

### Example - Procedure

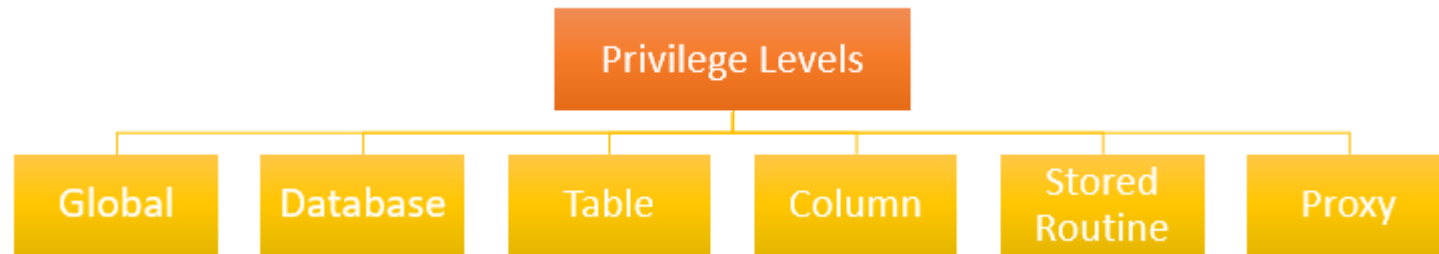
Let's look at some examples of how to revoke EXECUTE privileges on a procedure in MySQL.

If you wanted to revoke EXECUTE privileges on a procedure called *MyFirstProc* from a user named *smithj*, you would run the following REVOKE statement:

```
REVOKE EXECUTE ON PROCEDURE MyFirstProc FROM 'smithj'@'localhost';
```



## (B) Define your database users and their access privileges





# Database users and their access privileges

MySQL Workbench

new connection x

File Edit View Query Database Server Tools Scripting Help

Navigator: Administration - Users and Privileges

**MANAGEMENT**

- Server Status
- Client Connections
- Users and Privileges**
- Status and System Variables
- Data Export
- Data Import/Restore

**INSTANCE**

- Startup / Shutdown
- Server Logs
- Options File

**PERFORMANCE**

- Dashboard
- Performance Reports
- Performance Schema Setup

**Administration** Schemas

Information

**Users and Privileges**

new connection

User Accounts

User	From Host
mysql.infoschema	localhost
mysql.session	localhost
mysql.sys	localhost
root	localhost
sam	%

**Details for account sam@%**

Login Account Limits **Administrative Roles** Schema Privileges

Role	Description
<input checked="" type="checkbox"/> DBA	grants the rights to perform all tasks
<input checked="" type="checkbox"/> MaintenanceAdmin	grants rights needed to maintain server
<input checked="" type="checkbox"/> ProcessAdmin	rights needed to assess, monitor, and kill a
<input checked="" type="checkbox"/> UserAdmin	grants rights to create users logins and res
<input checked="" type="checkbox"/> SecurityAdmin	rights to manage logins and grant and reve
<input checked="" type="checkbox"/> MonitorAdmin	minimum set of rights needed to monitor s
<input checked="" type="checkbox"/> DBManager	grants full rights on all databases
<input checked="" type="checkbox"/> DBDesigner	rights to create and reverse engineer any d
<input checked="" type="checkbox"/> ReplicationAdmin	rights needed to setup and manage replica
<input checked="" type="checkbox"/> BackupAdmin	minimal rights needed to backup any data

Global Privileges

- ☒ ALTER
- ☒ ALTER ROUTINE
- ☒ CREATE
- ☒ CREATE ROUTINE
- ☒ CREATE TABLESPACE
- ☒ CREATE TEMPORARY TABLES
- ☒ CREATE USER
- ☒ CREATE VIEW
- ☒ DELETE
- ☒ DROP
- ☒ EVENT
- ☒ EXECUTE
- ☒ FILE
- ☒ GRANT OPTION
- ☒ INDEX
- ☒ INSERT

Revoke All Privileges

Add Account Delete Refresh

Revert Apply



---

## (C) Understand SQL Injection

“Code injection is the top security threat to web applications.”

— Open Web Application  
Security Project (OWASP)

---



# What is the SQL Injection?

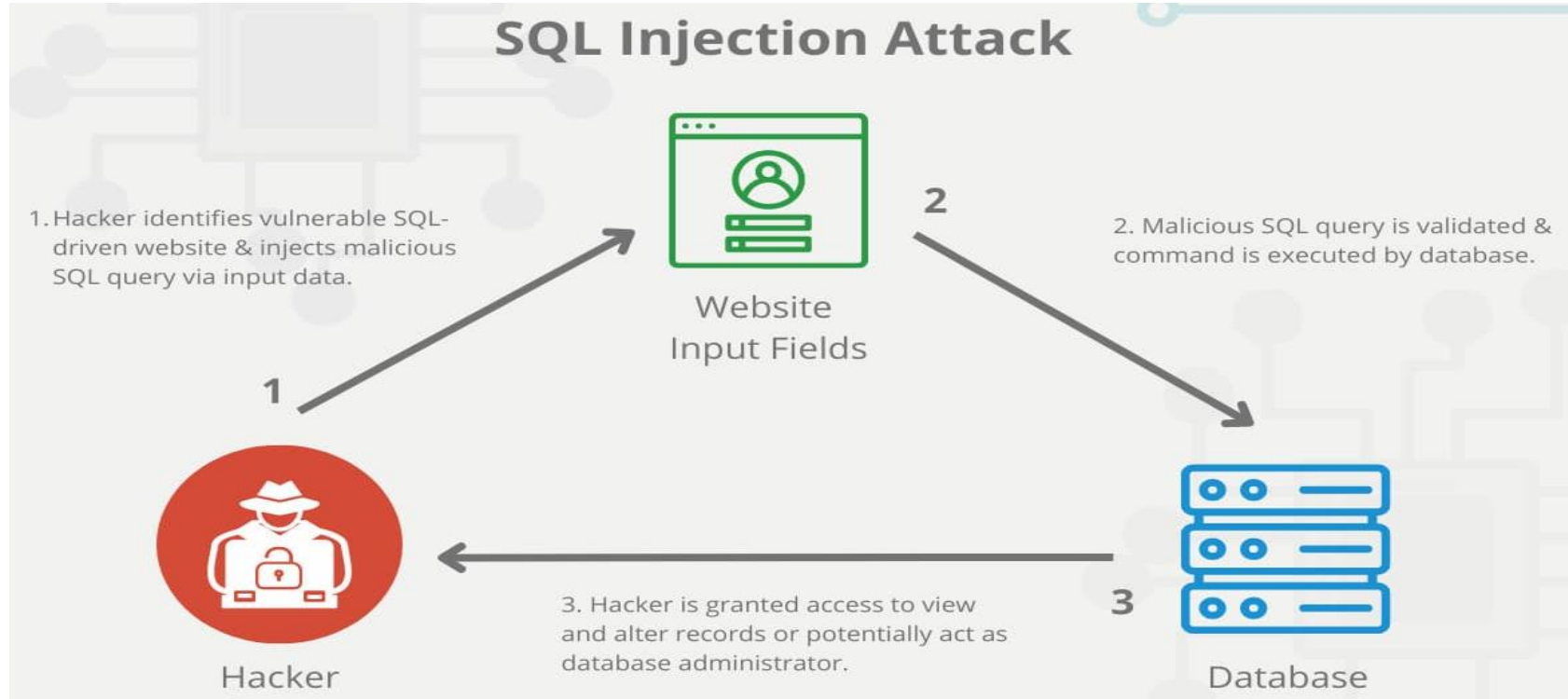
SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. This can allow an attacker to view data that they are not normally able to retrieve.

- Execute arbitrary SQL requests
- Craft string to manipulate SQL syntax and inject code
- Any query which uses dynamic-data is vulnerable
- URL parameters, form data, cookies, database data
- SQLi
- Single biggest security concern
- Ranked number one security threat by the Open Web Application Security Project (OWASP)





# What is the SQL Injection Attack?



# Example #1: SQL Injection for Insert Command



```
$sql = "INSERT INTO subjects ";  
$sql .= "(menu_name, position, visible) ";  
$sql .= "VALUES (";  
$sql .= "" . $subject['menu_name'] . ", ";  
$sql .= "" . $subject['position'] . ", ";  
$sql .= "" . $subject['visible'] . "";  
$sql .= ")";
```

```
$subject['menu_name'] = "David's Story";
```

```
"INSERT INTO subjects (menu_name, position, visible)  
VALUES ('David's Story', '1', TRUE)";
```

## Example #2: SQL Injection for Select/Insert Command



```
$sql = "SELECT * subjects WHERE id=" . $id . "";
```

```
$id = "5";
```

```
"SELECT * FROM subjects WHERE id='5'"
```

```
$id = "; INSERT INTO admins (username, password)  
VALUES ('hacker', 'abcd1234'); SELECT * FROM  
subjects WHERE id='5';
```

```
"SELECT * FROM subjects WHERE id="; INSERT INTO  
admins (username, password) VALUES ('hacker',  
'abcd1234'); SELECT * FROM subjects WHERE id='5'"
```



---

## (D) Sanitize data for SQL

---



# Mitigating SQL Injection Attacks: Input Sanitization

One way SQL injections can be mitigated is through input sanitization. Sanitization is the process of removing dangerous characters from user input.

This is important because they allow attackers to extend SQL queries to gain more information from a database.

Careful, this method is not the perfect defense against SQL injections. Removing characters may have no effect in some queries and, if an attacker finds a way to bypass the sanitization process, they can easily inject data into your system.

## Sanitize Data for MySQL

- Convert characters with meaning in SQL syntax to data
- Add a backslash before all single quotes
- `$subject['menu_name'] = "David\'s Story";`

Dangerous characters might include:

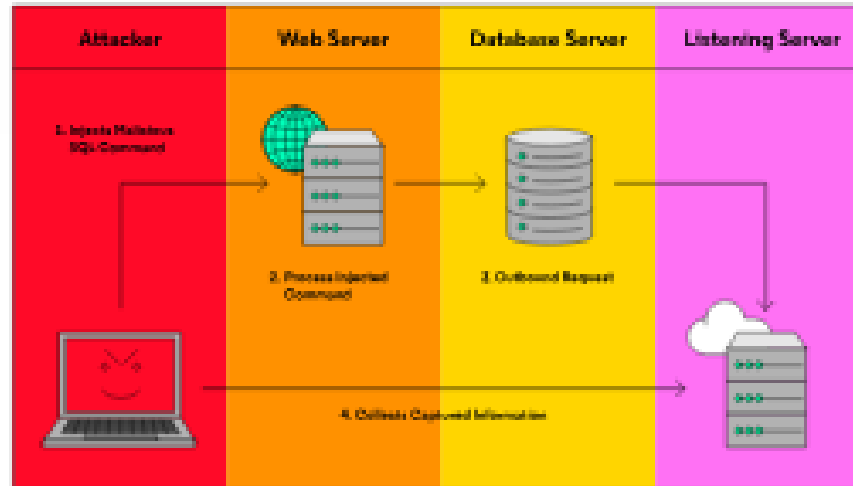
- `'`
- `;`
- `--`

# Example #1: Sanitize data for SQL

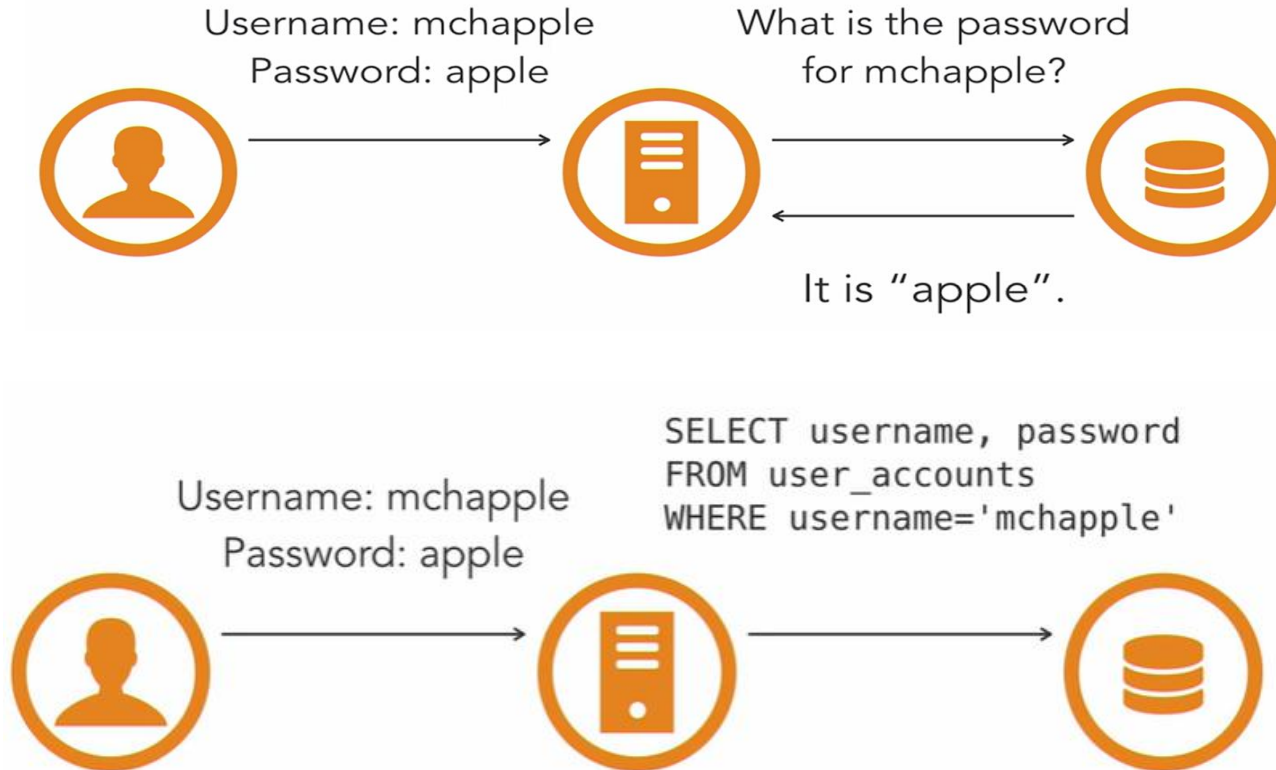
```
$sql = "SELECT * subjects ";  
$sql .= "WHERE id=" . db_escape($db, $id) . "";
```

```
$id = "; DROP TABLE payments; --";
```

```
"SELECT * FROM subjects WHERE id=' DROP TABLE  
payments; --'"
```



# Preventing SQL injection





## Example: Preventing SQL injection

```
SELECT username, password  
FROM user_accounts  
WHERE username = 'mchapple';
```

```
UPDATE user_accounts  
SET password='hacked'  
WHERE username='mchapple';
```

```
-- '
```





See Canvas.