# **MySQL — Week 2: Advanced Notes & Examples**

## **Table of Contents**

1. Subqueries  
   * Nested Subqueries
   * Correlated Subqueries
2. UNION and UNION ALL
3. Stored Procedures  
   * IN, OUT, INOUT Parameters
   * Examples
4. Triggers
5. Functions
6. Views

## **1. Subqueries**

A **subquery** is a query inside another query. It is useful for breaking down complex problems.

### **Nested Subquery**

Executed once and passed to the outer query.

SELECT name

FROM students

WHERE id IN (

SELECT student\_id

FROM marks

WHERE score > 80

);

### **Correlated Subquery**

Depends on the outer query and executes for each row.

SELECT s.name, s.id

FROM students s

WHERE score > (

SELECT AVG(score)

FROM marks m

WHERE m.student\_id = s.id

);

## **2. UNION and UNION ALL**

### **UNION**

* Combines results of two queries.
* Removes duplicates.

SELECT city FROM customers

UNION

SELECT city FROM suppliers;

### **UNION ALL**

* Combines results but keeps duplicates.

SELECT city FROM customers

UNION ALL

SELECT city FROM suppliers;

**Key Notes:**

* Both queries must have the same number of columns.
* Data types must be compatible.

## **3. Stored Procedures**

A stored procedure is a reusable set of SQL statements stored in the database.

### **Syntax**

DELIMITER $$

CREATE PROCEDURE procedure\_name(parameter\_list)

BEGIN

-- SQL statements

END $$

DELIMITER ;

### **Example**

DELIMITER $$

CREATE PROCEDURE get\_students()

BEGIN

SELECT \* FROM students;

END $$

DELIMITER ;

### **Parameters**

* **IN** → Input value.
* **OUT** → Return value.
* **INOUT** → Both input and output.

DELIMITER $$

CREATE PROCEDURE get\_marks(IN student\_id INT, OUT avg\_marks FLOAT)

BEGIN

SELECT AVG(score) INTO avg\_marks

FROM marks

WHERE marks.student\_id = student\_id;

END $$

DELIMITER ;

## **4. Triggers**

A trigger is executed automatically when an event occurs (INSERT, UPDATE, DELETE).

### **Syntax**

DELIMITER $$

CREATE TRIGGER trigger\_name

{BEFORE | AFTER} {INSERT | UPDATE | DELETE} ON table\_name

FOR EACH ROW

BEGIN

-- actions

END $$

DELIMITER ;

### **Example**

DELIMITER $$

CREATE TRIGGER before\_insert\_students

BEFORE INSERT ON students

FOR EACH ROW

BEGIN

SET NEW.created\_at = NOW();

END $$

DELIMITER ;

* **NEW** → Refers to new row values.
* **OLD** → Refers to existing row values.

## **5. Functions**

Functions return a single value and can be used in SQL queries.

### **Syntax**

DELIMITER $$

CREATE FUNCTION function\_name(parameters)

RETURNS datatype

DETERMINISTIC

BEGIN

-- SQL statements

RETURN value;

END $$

DELIMITER ;

### **Example**

DELIMITER $$

CREATE FUNCTION get\_total\_marks(s\_id INT) RETURNS INT

DETERMINISTIC

BEGIN

DECLARE total INT;

SELECT SUM(score) INTO total FROM marks WHERE student\_id = s\_id;

RETURN total;

END $$

DELIMITER ;

Usage:

SELECT name, get\_total\_marks(id) AS total\_score

FROM students;

## **6. Views**

A **view** is a virtual table based on a query result.

### **Creating a View**

CREATE VIEW high\_scorers AS

SELECT s.name, m.score

FROM students s

JOIN marks m ON s.id = m.student\_id

WHERE m.score > 80;

### **Using a View**

SELECT \* FROM high\_scorers;

### **Updating Through Views**

UPDATE high\_scorers SET score = 95 WHERE name = 'John';

### **Dropping a View**

DROP VIEW high\_scorers;

# **✅ Summary**

* **Subqueries**: Inner queries (Nested, Correlated).
* **UNION / UNION ALL**: Combine results, with or without duplicates.
* **Stored Procedures**: Reusable SQL blocks with parameters.
* **Triggers**: Automatic actions on table events.
* **Functions**: Return single values, usable in queries.
* **Views**: Virtual tables for simplicity and readability.