# ITERATE & LEAVE Statements in MySQL

Instructor: Sriya Ivaturi

# Introduction

- In MySQL stored programs (procedures, functions), we use loops.
- Two control statements are helpful inside loops:
- LEAVE – exits the loop immediately.
- ITERATE – skips the remaining statements of current loop iteration and starts the next iteration.

# LEAVE Statement

- LEAVE is like “break” in other languages.
- Syntax:

```
LEAVE label_name;
```

- Exits from the loop (or block) with the given label.

# ITERATE Statement

- ITERATE is like “continue” in other languages.
- Syntax:

```
ITERATE label_name;
```

- Exits from the loop (or block) with the given label.

# Example Using LEAVE

```
DELIMITER //
CREATE PROCEDURE leave_example()
BEGIN
 DECLARE i INT DEFAULT 1;
 my_loop: LOOP
   IF i > 5 THEN
     LEAVE my_loop; -- exit loop when i>5
   END IF;
   SELECT i;
   SET i = i + 1;
 END LOOP my_loop;
END//
DELIMITER ;
```

# Example Using ITERATE

```
DELIMITER //
CREATE PROCEDURE iterate_example()
BEGIN
 DECLARE i INT DEFAULT 0;
 my_loop: LOOP
   SET i = i + 1;
   IF i = 3 THEN
     ITERATE my_loop; -- skip printing 3
   END IF;
   SELECT i;
   IF i >= 5 THEN
     LEAVE my_loop; -- exit loop
   END IF;
 END LOOP my_loop;
END//
DELIMITER ;
```

# Key Differences

| Statement | Purpose | Similar to in other languages |
|---|---|---|
| LEAVE | Exit loop entirely | break |
| ITERATE | Skip to next iteration | continue |

# Summary

- Use LEAVE when you want to exit the loop immediately.
- Use ITERATE when you want to skip the rest of the loop’s body and move to the next iteration.
- Both must reference a loop label.