Cursors in MySQL

Row-by-row processing using cursors inside stored procedures

Instructor: Sriya Ivaturi

When to Use Cursors

- - When a single SQL query is not sufficient
- - Complex row-wise calculations
- - Conditional logic on each row
- - Dynamic updates or inserts

Steps to Use a Cursor

- 1. DECLARE the cursor
- 2. OPEN the cursor
- 3. FETCH rows into variables
- 4. LOOP through rows
- 5. CLOSE the cursor

Syntax of Cursor in MySQL

- DECLARE cursor_name CURSOR FOR SELECT_statement;
- OPEN cursor_name;
- FETCH cursor_name INTO var1, var2, ...;
- CLOSE cursor_name;

Cursor with Loop Structure

- DECLARE done INT DEFAULT FALSE;
- DECLARE name_var VARCHAR(50);
- DECLARE emp_cursor CURSOR FOR SELECT name FROM employees;
- DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

- OPEN emp_cursor;
- read_loop: LOOP
- FETCH emp_cursor INTO name_var;
- IF done THEN LEAVE read_loop; END IF;
- SELECT CONCAT('Employee: ', name_var);
- END LOOP;
- CLOSE emp_cursor;

Explanation of Important Parts

- - DECLARE CURSOR: defines result set
- - CONTINUE HANDLER: handles end of data
- - LOOP: processes each row
- - LEAVE: exits loop after last row

Simple Cursor Example

```
DELIMITER $$
CREATE PROCEDURE ShowEmployees()
BEGIN
  DECLARE done INT DEFAULT FALSE;
  DECLARE emp_name VARCHAR(50);
  DECLARE emp_cursor CURSOR FOR SELECT name FROM employees;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
  OPEN emp_cursor;
  read_loop: LOOP
    FETCH emp_cursor INTO emp_name;
    IF done THEN LEAVE read_loop; END IF;
    SELECT CONCAT('Employee: ', emp_name);
  END LOOP;
  CLOSE emp_cursor;
END$$
DELIMITER ;
```

Questions

- Q1: What happens if you forget to declare a CONTINUE HANDLER?
- Q2: Can cursors be used outside stored procedures in MySQL?
- Q3: What is the difference between LEAVE and ITERATE?

Answers

- A1: Cursor throws error on FETCH after last row
- A2: No, cursors must be used inside stored procedures
- A3:
- - LEAVE: exits the loop
- - ITERATE: skips to next iteration

Summary

- - Cursors allow row-wise logic in stored procedures
- - CONTINUE HANDLER prevents runtime errors
- - LOOP + FETCH for row processing
- - Use LEAVE to exit loop safely