**CURSOR**

**DEFINATION :**

**A cursor in MySQL is a database object** that allows you to retrieve and manipulate rows returned by a query one at a time. Cursors are mainly used in stored procedures and functions to process individual rows returned by a SELECT statement**.**

**THERE ARE FOUR STEPS IN CURSOR :**

**DECLARE**: A cursor is declared with the DECLARE statement, which associates it with a specific SELECT query.

**DECLARE** cursor\_name **CURSOR FOR** SELECT\_statement;

**OPEN**: The cursor is opened using the OPEN statement, which initializes the cursor and positions it before the first row of the result set.

**OPEN** cursor\_name;

**LOOP START**

**FETCH**: The FETCH statement is used to retrieve the next row in the result set and advance the cursor.

**FETCH** cursor\_name INTO variable1, variable2, ...;

**LOOP END**

**CLOSE**: The cursor is closed using the CLOSE statement when it is no longer needed. This deallocates the resources associated with the cursor.

**CLOSE** cursor\_name;

**CREATE PROCEDURE** process\_records()

BEGIN

**DECLARE** done INT DEFAULT FALSE;

**DECLARE** id INT;

**DECLARE** name VARCHAR(100);

-- Declare the cursor for a SELECT statement

**DECLARE** my\_cursor CURSOR FOR SELECT id, name FROM my\_table;

-- Declare a handler to set 'done' when the cursor reaches the end

**DECLARE** CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Open the cursor

**OPEN** my\_cursor;

-- Fetch rows in a loop

read\_loop: LOOP

**FETCH** ----------------;

IF done THEN

LEAVE read\_loop;

END IF;

**SELECT** ----------------;

END LOOP;

**CLOSE** my\_cursor;

END

**ADVANTAGES :**

* **Row-by-Row Processing:** Ideal for operations that need to handle one row at a time.
* **Sequential Access**: Allows for processing rows in a specific order.
* **Custom Logic**: Enables complex, conditional logic within stored procedures.
* **Flexibility:** Combines SQL with procedural code for advanced data manipulation.

**DISADVANTAGES :**

* **Performance Overhead:** Slower due to processing one row at a time instead of the entire set.
* **Complexity:** Adds complexity to code, making it harder to read and maintain.
* **Resource Usage:** Consumes more memory and CPU, especially with large result sets**.**