Question 5

Create **cursor** for **Employee** table & **extract** the values from the table. **Declare** the variables, **Open** the cursor & **extract** the values from the cursor. **Close** the cursor.

Employee(E_id, E_name, Age, Salary)

Solution

1. Creating the Employee Table and insert few records

```
CREATE TABLE Employee (
    E_id INT,
    E_name VARCHAR(255),
    Age INT,
    Salary DECIMAL(10, 2)
);

INSERT INTO Employee (E_id, E_name, Age, Salary)
VALUES
    (1, 'Shankar', 30, 50000.00),
    (2, 'Rahul', 25, 45000.00),
    (3, 'Muskaan', 35, 62000.00),
    (4, 'Mathew', 28, 52000.00),
    (5, 'Raj', 32, 58000.00);
```

---

2. Create a Stored Procedure with Cursor

To create a cursor for the Employee table, extract values using the cursor, and then close the cursor in MySQL, you'll need to use stored procedures that support cursor operations.

```
DELIMITER //

CREATE PROCEDURE fetch_employee_data()
BEGIN
    -- Declare variables to store cursor values
    DECLARE emp_id INT;
    DECLARE emp_name VARCHAR(255);
    DECLARE emp_age INT;
    DECLARE emp_salary DECIMAL(10, 2);

    -- Declare a cursor for the Employee table
    DECLARE emp_cursor CURSOR FOR
        SELECT E_id, E_name, Age, Salary
        FROM Employee;

    -- Declare a continue handler for the cursor
    DECLARE CONTINUE HANDLER FOR NOT FOUND
        SET @finished = 1;

    -- Open the cursor
    OPEN emp_cursor;



-- Initialize a variable to control cursor loop
```

```
    SET @finished = 0;

    -- Loop through the cursor results
    cursor_loop: LOOP
        -- Fetch the next row from the cursor into variables
        FETCH emp_cursor INTO emp_id, emp_name, emp_age, emp_salary;

        -- Check if no more rows to fetch
        IF @finished = 1 THEN
            LEAVE cursor_loop;
        END IF;

        -- Output or process each row (for demonstration, print the values)
        SELECT CONCAT('Employee ID: ', emp_id, ', Name: ', emp_name, ', Age: ', emp_age,
',
Salary: ', emp_salary) AS Employee_Info;
    END LOOP;

    -- Close the cursor
    CLOSE emp_cursor;
END//

DELIMITER ;
```

In this stored procedure (fetch_employee_data):

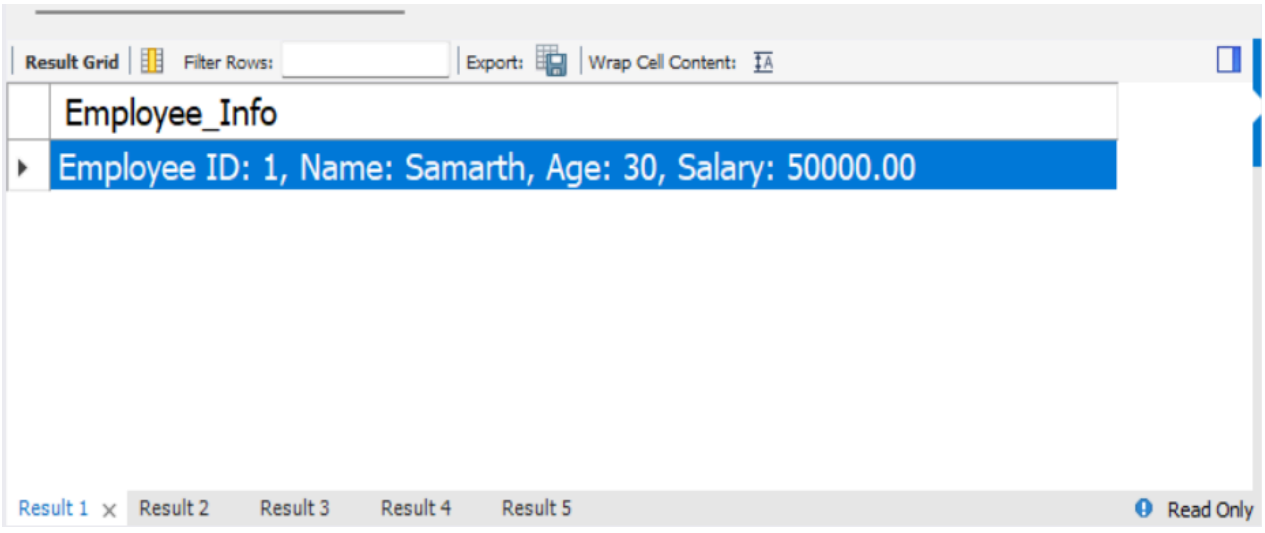
- **We declare variables (emp_id, emp_name, emp_age, emp_salary) to store values retrieved from the cursor.**
- **A cursor (emp_cursor) is declared to select E_id, E_name, Age, and Salary from the Employee table.**
- **We declare a continue handler (CONTINUE HANDLER) for NOT FOUND condition to handle the end of cursor data.**
- **The cursor is opened (OPEN emp_cursor), and a loop (cursor_loop) is used to fetch each row from the cursor.**
- **We fetch values into the variables and process them within the loop (for demonstration, we print the values using a SELECT statement).**
- **The loop continues until all rows are fetched (@finished = 1).**
- **Finally, the cursor is closed (CLOSE emp_cursor).**

---

3. Execute the Stored Procedure

Once the stored procedure fetch_employee_data is created, you can execute it to fetch and process data from the Employee table:

CALL fetch_employee_data();



- **The stored procedure fetch_employee_data declares variables (emp_id, emp_name, emp_age, emp_salary) to store values retrieved from the cursor.**
- **A cursor (emp_cursor) is declared for the Employee table to select E_id, E_name, Age, and Salary.**
- **The cursor is opened (OPEN emp_cursor), and the FETCH statement retrieves the first row from the cursor into the declared variables.**
- **A WHILE loop processes each row fetched by the cursor (SQLSTATE() = '00000' checks for successful fetching).**
- **Within the loop, you can perform operations or output the values of each row.**
- **The CLOSE statement closes the cursor after processing all rows.**