

Experiment 15: To understand the concepts of implicit and explicit cursor.

Objective: Students will be able to implement the concept of implicit and explicit cursor.

1. Using implicit cursor update the salary by an increase of 10% for all the records in EMPLOYEES table, and finally display how many records have been updated. If no records exist display the message “No Change”.

```
1 DECLARE
2     v_rows_updated NUMBER;
3 BEGIN
4     UPDATE EMPLOYEES
5     SET SALARY = SALARY * 1.10;
6
7     v_rows_updated := SQL%ROWCOUNT;
8
9     IF v_rows_updated > 0 THEN
10        DBMS_OUTPUT.PUT_LINE(v_rows_updated || ' records updated.');
```

Results	Explain	Describe	Saved SQL	History
4 records updated.				
1 row(s) updated.				
0.01 seconds				

2. Using explicit cursor fetch the employee name, employee_id and salary of all the records from EMPLOYEES table.

INPUT:

```
1 DECLARE
2     CURSOR emp_cursor IS
3         SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY FROM EMPLOYEES;
4
5     v_employee_id EMPLOYEES.EMPLOYEE_ID%TYPE;
6     v_first_name EMPLOYEES.FIRST_NAME%TYPE;
7     v_last_name EMPLOYEES.LAST_NAME%TYPE;
8     v_salary EMPLOYEES.SALARY%TYPE;
9 BEGIN
10    OPEN emp_cursor;
11
12    LOOP
13        FETCH emp_cursor INTO v_employee_id, v_first_name, v_last_name, v_salary;
14
15        EXIT WHEN emp_cursor%NOTFOUND;
16
17        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id ||
18                             ', Name: ' || v_first_name || ' ' || v_last_name ||
19                             ', Salary: ' || v_salary);
20    END LOOP;
21
22    CLOSE emp_cursor;
23 END;
24 /
```

OUTPUT:

Results	Explain	Describe	Saved SQL	History
Employee ID: 2, Name: Jane Smith, Salary: 3300 Employee ID: 3, Name: Sam Brown, Salary: 2750 Employee ID: 1, Name: John Doe, Salary: 2200 Employee ID: 4, Name: Anna Johnson, Salary: 3850 Statement processed. 0.12 seconds				

- Using explicit cursor Insert the records from EMPLOYEES table for the columns employee_id, Last_Name and salary for those records whose salary exceeds 2500 into a new table TEMP_EMP.

Creating TEMP_EMP table:

1	CREATE TABLE TEMP_EMP (
2	EMPLOYEE_ID NUMBER,
3	LAST_NAME VARCHAR2(50),
4	SALARY NUMBER(10, 2)
5);

Results	Explain	Describe	Saved SQL
Table created. 0.03 seconds			

INPUT:

```
1  DECLARE
2      CURSOR emp_cursor IS
3          SELECT EMPLOYEE_ID, LAST_NAME, SALARY
4              FROM EMPLOYEES
5              WHERE SALARY > 2500;
6
7      v_employee_id EMPLOYEES.EMPLOYEE_ID%TYPE;
8      v_last_name EMPLOYEES.LAST_NAME%TYPE;
9      v_salary EMPLOYEES.SALARY%TYPE;
10 BEGIN
11     OPEN emp_cursor;
12
13     LOOP
14         FETCH emp_cursor INTO v_employee_id, v_last_name, v_salary;
15
16         EXIT WHEN emp_cursor%NOTFOUND;
17
18         INSERT INTO TEMP_EMP (EMPLOYEE_ID, LAST_NAME, SALARY)
19             VALUES (v_employee_id, v_last_name, v_salary);
20
21         DBMS_OUTPUT.PUT_LINE('Inserted Record - Employee ID: ' || v_employee_id ||
22                               ', Last Name: ' || v_last_name ||
23                               ', Salary: ' || v_salary);
24     END LOOP;
25
26     CLOSE emp_cursor;
27
28     DBMS_OUTPUT.PUT_LINE('Records inserted into TEMP_EMP successfully.');
```

OUTPUT:

1SELECT * FROM TEMP_EMP;

Results

ExplainDescribeSaved SQLHistory

EMPLOYEE_ID	LAST_NAME	SALARY
2	Smith	3300
3	Brown	2750
4	Johnson	3850

3 rows returned in 0.01 secondsDownload