**Assignment No:4**

**Title:** Write a program in PL/SQL to insert data into two tables from one table using an implicit cursor and print a list of students and the name of the departments.

**Problem Statement:** Reads data from a source table (e.g., StudentDetails) using an implicit cursor. Based on certain conditions or transformations, inserts specific record into two separate target tables (e.g., Computer and Civil). After processing the data, prints the list of students along with their department names to verify the correctness of the operation.

**Objective:** To acquire knowledge of database query languages

**Outcome:** Implement different SQL and PLSQL operations using suitable databases

**Tools Required:** Ubuntu OS, Mysql.

**Theory:**

**1. Cursor:**

Oracle creates a memory area, known as the context area, for processing an SQL statement, which contains all the information needed for processing the statement; for example, the number of rows processed, etc. A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the

cursor holds is referred to as the active set.You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors −

* Implicit cursors
* Explicit cursors

**1.1 Implicit Cursors**

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the information in it.

Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted. For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

In PL/SQL, you can refer to the most recent implicit cursor as the SQL cursor, which always has attributes such as %FOUND, %ISOPEN, %NOTFOUND, and %ROWCOUNT. The SQL cursor has additional attributes, %BULK\_ROWCOUNT and %BULK\_EXCEPTIONS, designed for use with the FORALL statement. The following table provides the description of the most used attributes –

|  |  |
| --- | --- |
| **S.No** | **Attribute & Description** |
| 1 | **%FOUND**  Returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows or a SELECT INTO statement returned one or more rows. Otherwise, it returns FALSE. |
| 2 | **%NOTFOUND**  The logical opposite of %FOUND. It returns TRUE if an INSERT, UPDATE, or DELETE statement affected no rows, or a SELECT INTO statement returned no rows. Otherwise, it returns FALSE. |
| 3 | **%ISOPEN**  Always returns FALSE for implicit cursors, because Oracle closes the SQL cursor automatically after executing its associated SQL statement. |
| 4 | **%ROWCOUNT**  Returns the number of rows affected by an INSERT, UPDATE, or DELETE statement, or returned by a SELECT INTO statement. |

The following program will update the table and increase the salary of each customer by 500 and use the SQL%ROWCOUNT attribute to determine the number of rows affected –

|  |
| --- |
| DECLARE  total\_rows number(2);  BEGIN  UPDATE customers  SET salary = salary + 500;  IF sql%notfound THEN  dbms\_output.put\_line('no customers selected');  ELSIF sql%found THEN  total\_rows := sql%rowcount;  dbms\_output.put\_line( total\_rows || ' customers selected ');  END IF;  END;  / |

**1.2 Explicit Cursors**

Explicit cursors are programmer-defined cursors for gaining more control over the context area. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row. Working with an explicit cursor includes the following steps −

* Declaring the cursor for initializing the memory
* Opening the cursor for allocating the memory
* Fetching the cursor for retrieving the data
* Closing the cursor to release the allocated memory

Declaring the cursor defines the cursor with a name and the associated SELECT statement.

|  |
| --- |
| CURSOR c\_customers IS SELECT id, name, address FROM customers; |

Opening the cursor allocates the memory for the cursor and makes it ready for fetching the rows returned by the SQL statement into it.

|  |
| --- |
| OPEN c\_customers; |

Fetching the cursor involves accessing one row at a time.

|  |
| --- |
| FETCH c\_customers INTO c\_id, c\_name, c\_addr; |

Closing the cursor means releasing the allocated memory.

|  |
| --- |
| CLOSE c\_customers; |

Following is a complete example to illustrate the concepts of explicit cursors

|  |
| --- |
| DECLARE  c\_id customers.id%type;  c\_name customers.name%type;  c\_addr customers.address%type;  CURSOR c\_customers is  SELECT id, name, address FROM customers;  BEGIN  OPEN c\_customers;  LOOP  FETCH c\_customers into c\_id, c\_name, c\_addr;  EXIT WHEN c\_customers%notfound;  dbms\_output.put\_line(c\_id || ' ' || c\_name || ' ' || c\_addr);  END LOOP;  CLOSE c\_customers;  END;  / |

PL/SQL block of code using parameterized Cursor, that will merge the data available in the newly created table N\_RollCall with the data available in the table O\_RollCall. If the data in the first table already exist in the second table then that data should be skipped.

|  |
| --- |
| declare  res int;  o\_roll oldstd.oroll%type;  o\_name oldstd.oname%type;  CURSOR old\_cur is SELECT oroll, oname FROM oldstd;  PROCEDURE chk(roll IN int, result OUT int)  AS  v\_count NUMBER;  BEGIN  SELECT COUNT(\*) INTO v\_count FROM newstd WHERE nroll = roll;  IF(v\_count > 0) THEN  result := 1;  ELSE  result := 0;  END IF;  END;  begin  OPEN old\_cur;  LOOP  FETCH old\_cur into o\_roll, o\_name;  EXIT WHEN old\_cur%notfound;  chk(o\_roll,res);  if(res=0) then  insert into newstd values(o\_roll,o\_name);  else  dbms\_output.put\_line('The roll no ' || o\_roll||' already present in new student table');  end if;  END LOOP;  CLOSE old\_cur;  end; |

**Conclusion:**

We have successfully implemented PL/SQL Cursor, to insert data into two tables from one table using an implicit cursor