Advanced Explicit Cursor Concepts

- Jayendra Khatod

Objectives

- **Cursors with Parameters**
- **The FOR UPDATE Clause**
- **The WHERE CURRENT OF Clause**
- **Cursors with Sub-queries**
- **REF cursors**

Cursors with Parameters

- **Syntax**

```
CURSOR cursor_name
   [(parameter_name datatype, ...)]
IS
   select_statement;
```

  - **Pass parameter values to a cursor when the cursor is opened and the query is executed.**
  - **Open an explicit cursor several times with a different active set each time.**

Cursors with Parameters

- **Pass the department number and job title to the WHERE clause.**
- **Example**

```
DECLARE
  CURSOR emp_cursor
   (p_deptno NUMBER, p_job VARCHAR2) IS
    SELECTempno, ename
    FROM  emp
    WHERE deptno = p_deptno
     AND  job = p_job;
BEGIN
  OPEN emp_cursor(10, 'CLERK');
...
```

The FOR UPDATE Clause

- **Syntax**

```
SELECT          ...
FROM            ...
FOR UPDATE [OF column_reference][NOWAIT];
```

  - **Explicit locking lets you deny access for the duration of a transaction.**
  - **Lock the rows *before* the update or delete.**

The FOR UPDATE Clause

- Retrieve the employees who work in department 30.
- Example

```
DECLARE
   CURSOR emp_cursor IS
      SELECT empno, ename, sal
      FROM   emp
      WHERE  deptno = 30
      FOR UPDATE OF sal NOWAIT;
```

The WHERE CURRENT OF Clause

- Syntax

```
WHERE CURRENT OF cursor ;
```

  - Use cursors to update or delete the current row.
  - Include the FOR UPDATE clause in the cursor query to lock the rows first.
  - Use the WHERE CURRENT OF clause to reference the current row from an explicit cursor.

The WHERE CURRENT OF Clause

Example

```
DECLARE
  CURSOR sal_cursor IS
    SELECT sal FROM emp
    WHERE deptno = 30
    FOR UPDATE OF sal NOWAIT;
BEGIN
  FOR emp_record IN sal_cursor LOOP
    UPDATEemp
    SET   sal = emp_record.sal * 1.10
    WHERE CURRENT OF sal_cursor;
  END LOOP;
  COMMIT;
END;
```

Cursors with Subqueries

Example

```
DECLARE
  CURSOR my_cursor IS
    SELECT t1.deptno, t1.dname, t2.STAFF
    FROM   dept t1,(SELECT deptno,
                           count(*) STAFF
                     FROM   emp
                     GROUP BY deptno) t2
    WHERE  t1.deptno = t2.deptno
    AND    t2.STAFF >= 5;
```

Defining REF CURSOR Types

Define a REF CURSOR type.

```
Define a REF CURSOR type
TYPE ref_type_name IS REF CURSOR
 [RETURN return_type];
```

Declare a cursor variable of that type.

```
ref_cv ref_type_name;
```

Example:

```
DECLARE
TYPE DeptCurTyp IS REF CURSOR RETURN
departments%ROWTYPE;
dept_cv DeptCurTyp;
```

Using the OPEN-FOR, FETCH, and CLOSE Statements

- **The OPEN-FOR statement associates a cursor variable with a multirow query, executes the query, identifies the result set, and positions the cursor to point to the first row of the result set.**

- **The FETCH statement returns a row from the result set of a multirow query, assigns the values of select-list items to corresponding variables or fields in the INTO clause, increments the count kept by %ROWCOUNT, and advances the cursor to the next row.**

- **The CLOSE statement disables a cursor variable.**

An Example of Fetching

```
DECLARE
    TYPE EmpCurTyp IS REF CURSOR;
    emp_cv EmpCurTyp;
    emp_rec emp%ROWTYPE;
    sql_stmt VARCHAR2(200);
    my_job VARCHAR2(10) := 'SA_REP';
BEGIN
    sql_stmt := 'SELECT * FROM emp
                     WHERE job_id IN (:j,:x)';
  OPEN emp_cv FOR sql_stmt USING my_job, 'AD_VP';
  LOOP
        FETCH emp_cv INTO emp_rec;
        EXIT WHEN emp_cv%NOTFOUND;
        dbms_output.put_line (emp_rec.job_id ||'  '||
        emp_rec.employee_id ||' '||emp_rec.last_name);
        END LOOP;
  CLOSE emp_cv;
END;
/
```

Summary

- You can return different active sets using cursors with parameters.
- You can define cursors with sub-queries and correlated sub-queries.
- You can manipulate explicit cursors with commands:
  - FOR UPDATE Clause
  - WHERE CURRENT OF Clause
  - REF Cursors

Thank You !