
CURSOR HANDLING

By
Parteek Bhatia
Assistant Professor
Dept of Comp Sc & Engg
Thapar University
Patiala

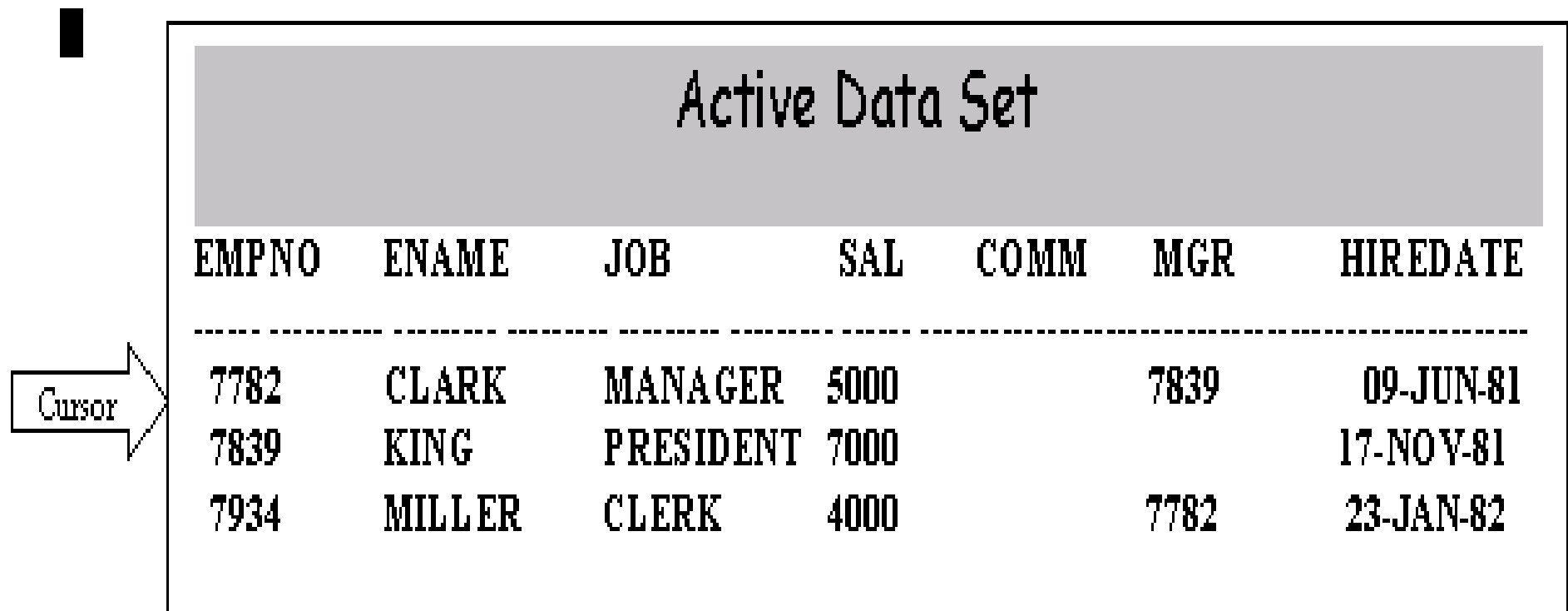
Definition

- Cursor is a memory (work) area that a oracle engine uses for its internal processing for executing and storing the results of SQL statement, and this work area is reserved for SQL's operations also called Oracle's Private area or CURSOR .
- Example: When a user fires a select statement as:
`SELECT empno, ename, job, sal FROM emp
WHERE deptno = 10;`
All the rows returned by the query are stored in the cursor at the Server and will be as displayed at the client end.

When user fires the SQL statement like

```
SQL > SELECT empno,ename,job,sal,comm,mgr,hiredate  
FROM scott.emp WHERE deptno =10 ;
```

When oracle engine executes the above query as a result some memory area is reserved and is fed with the result of the SQL statement which is shown below: -



The diagram illustrates the execution of an SQL query. A box labeled "Cursor" with an arrow points to a table titled "Active Data Set". The table contains the results of the query, with columns EMPNO, ENAME, JOB, SAL, COMM, MGR, and HIREDATE. The data is as follows:

Active Data Set						
EMPNO	ENAME	JOB	SAL	COMM	MGR	HIREDATE
7782	CLARK	MANAGER	5000		7839	09-JUN-81
7839	KING	PRESIDENT	7000			17-NOV-81
7934	MILLER	CLERK	4000		7782	23-JAN-82

Types of Cursor

- Implicit Cursors
- Explicit Cursors

Implicit Cursor

- Implicit Cursors are declared by PL/SQL implicitly for all SQL statements. They are opened and managed by Oracle engine internally. So there is no need to open and manage by the users, these operations are performed automatically.
- Implicit Cursors named “SQL” are declared by PL/SQL implicitly for all SQL statements. Implicit cursor attributes can be used to access information about the status of last insert, update, delete or single-row select statements. This can be done by preceding the implicit cursor attribute with the cursor name (i.e. SQL).

General Cursor Attributes

Attributes	Description
%ISOPEN	It returns TRUE if cursor is open, FALSE otherwise.
%FOUND	It returns TRUE if record was fetched successfully from the opened cursor, and FALSE otherwise.
%NOTFOUND	It returns TRUE if record was not fetched successfully and FALSE otherwise.
%ROWCOUNT	It returns number of records processed from cursor.

Implicit Cursor Attributes

Attributes	Description
SQL%ISOPEN	It is always FALSE because Oracle automatically closes an Implicit cursor after executing its SQL statement.
SQL%FOUND	It is TRUE if the most recently executed DML statement was successful.
SQL%NOTFOUND	It is TRUE if the most recently executed DML statement was not successful.
SQL%ROWCOUNT	It returns the number of rows affected by an INSERT, UPDATE, DELETE, or single row SELECT statement.

Use of SQL%NOTFOUND Attribute

Consider a PL/SQL code to display a message to check whether the record is deleted or not.

```
BEGIN
```

```
DELETE EMP WHERE EMPNO=&EMPNO;
```

```
IF SQL%NOTFOUND THEN
```

```
DBMS_OUTPUT.PUT_LINE('RECORD NOT DELETED');
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('RECORD DELETED');
```

```
END IF;
```

```
END
```

Use of SQL%ROWCOUNT Attribute

A PL/SQL code to display a message to give the number of records deleted by the delete statement issued in a PL/SQL block.

```
DECLARE
    N NUMBER;
BEGIN
    DELETE EMP WHERE DEPTNO=&DEPTNO;
    N:=SQL%ROWCOUNT;
    DBMS_OUTPUT.PUT_LINE('TOTAL NUMBER OF RECORD
    DELETED' || N);
END;
```

Explicit Cursor Handling

Steps:

- Declare the cursor

CURSOR <cursor-name> IS <select statement>;

- Open the cursor

OPEN <cursor-name>;

- Fetch data from the cursor one row at a time into memory variables.

FETCH <cursor-name> INTO <variables>;

- Process the data held in the memory variables as required using a loop.
- Exit from the loop after processing is complete.
- Close the cursor

Close <cursor-name>;

Need

- Select sal into s from emp where deptno=10;
- Select ename into e from emp where deptno=10;

Consider a PL/SQL code to display the empno, ename, job of employees of department number 10.

```
DECLARE
    CURSOR C1 IS SELECT EMPNO, ENAME, JOB FROM EMP WHERE
    DEPTNO=10;
    REC C1%ROWTYPE; /*rec is a row type variable for cursor c1 record,
                    containing empno, ename and job*/
BEGIN
    OPEN C1;
    LOOP
BEGIN
    OPEN C1;
    LOOP
        FETCH C1 INTO REC;
        EXIT WHEN C1%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('EMPNO '||REC.EMPNO);
        DBMS_OUTPUT.PUT_LINE('ENAME '||REC.ENAME);
        DBMS_OUTPUT.PUT_LINE('JOB '||REC.JOB);
    END LOOP;
    CLOSE C1;
END;
```

Consider a PL/SQL code to display the employee number and name of top 5 highest paid employees.

```
DECLARE
    EMPNAME EMP.ENAME%TYPE;
    EMPSAL EMP.SAL%TYPE;
    CURSOR TEMP1 IS SELECT ENAME, SAL FROM EMP ORDER BY
        SAL DESC;
BEGIN
    OPEN TEMP1;
    LOOP
        FETCH TEMP1 INTO EMPNAME, EMPSAL;
        EXIT WHEN TEMP1%ROWCOUNT>5 OR TEMP1%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(EMPNAME || EMPSAL);
    END LOOP;
    CLOSE TEMP1;
END;
```

Cursor FOR Loop

- The Cursor FOR Loop implicitly declares its loop index as a record of type %ROWTYPE, opens a cursor, repeatedly fetches rows of the values from the active set into fields in the record, then closes the cursor when all rows have been processed or when the EXIT command is encountered.

- Syntax

FOR <variable name> IN <cursor_name> LOOP
<statements>;
END LOOP;

Cursor FOR Loop

- A cursor for loop automatically does the following:
- Implicitly declares its loop index or `variable_name` as a `%rowtype` record.
- Opens a cursor.
- Fetches a row from the cursor for each loop iteration.
- Closes the cursor when all rows have been processed

A PL/SQL code to display the empno, ename, job of employees of department number 10 with CURSOR FOR Loop statement.

DECLARE

CURSOR C1 IS SELECT EMPNO, ENAME, JOB FROM EMP
WHERE DEPTNO=10;

BEGIN

FOR REC IN C1 LOOP

DBMS_OUTPUT.PUT_LINE('EMPNO '||REC.EMPNO);

DBMS_OUTPUT.PUT_LINE('ENAME '||REC.ENAME);

DBMS_OUTPUT.PUT_LINE('JOB '||REC.JOB);

END LOOP;

END;

A PL/SQL code to display the employee number and name of top 5 highest paid employees with CURSOR FOR LOOP statement.

DECLARE

CURSOR TEMP1 IS SELECT ENAME, SAL FROM EMP
ORDER BY SAL DESC;

BEGIN

FOR REC IN TEMP1 LOOP

EXIT WHEN TEMP1%ROWCOUNT>5;

DBMS_OUTPUT.PUT_LINE(REC.ENAME || REC.SAL);

END LOOP;

END;

Cursors with Parameters

Syntax:

To declare

```
CURSOR cursor_name (variable_name datatype) IS  
<SELECT statement...>
```

To Open

```
OPEN cursor_name (value/variable/expression);
```

Consider a PL/SQL code to display the empno, ename, job of employees of a particular department number whose value is passed as a parameter.

```
DECLARE
    CURSOR C1 (d number) IS SELECT EMPNO, ENAME, JOB FROM EMP WHERE
        DEPTNO=d;
    REC C1%ROWTYPE; /*rec is a row type variable for cursor c1 record,
                        containing empno, ename and job*/
BEGIN
    OPEN C1(&d);
    LOOP
BEGIN
        OPEN C1;
        LOOP
            FETCH C1 INTO REC;
            EXIT WHEN C1%NOTFOUND;
            DBMS_OUTPUT.PUT_LINE('EMPNO '||REC.EMPNO);
            DBMS_OUTPUT.PUT_LINE('ENAME '||REC.ENAME);
            DBMS_OUTPUT.PUT_LINE('JOB '||REC.JOB);
        END LOOP;
    CLOSE C1;
END;
```

Consider a PL/SQL code to display the empno, ename, job of employees of a particular department number whose value is passed as a parameter.

With For Loop

```
DECLARE
```

```
    CURSOR C1(d number) IS SELECT EMPNO, ENAME, JOB  
    FROM EMP WHERE DEPTNO=d;
```

```
BEGIN
```

```
FOR REC IN C1(&d) LOOP
```

```
    DBMS_OUTPUT.PUT_LINE('EMPNO '||REC.EMPNO);
```

```
    DBMS_OUTPUT.PUT_LINE('ENAME '||REC.ENAME);
```

```
    DBMS_OUTPUT.PUT_LINE('JOB '||REC.JOB);
```

```
END LOOP;
```

```
END;
```