CURSORS:

CURSOR IS A TEMP.MEMORY / A PRIVATE SQL AREA(PSA) / A WORK SPACE. CURSOR ARE TWO TYPES.THOSE ARE:

I) EXPLICIT CURSOR (USER DEFINE CURSOR)

II) IMPLICIT CURSOR (SYSTEM DEFINE CURSOR)

I) EXPLICIT CURSOR:

THESE CURSOR ARE CREATING BY USER FOR HOLDING MULTIPLE ROWS BUT WE CAN ACCESS ONLY ONE ROW AT TIME. (ONE BY ONE / ROW BY ROW MANNER).

IF WE WANT TO CREATE AN EXPLICIT CURSOR, WE NEED FOLLOW THE FOLLOWING FOUR STEPS.THOSE ARE

1) DECLARING A CURSOR

2) OPEN A CURSOR

3) FETCH ROWS FROM A CURSOR

4) CLOSE A CURSOR

STEPS TO CREATE EXPLICIT CURSOR:

1)DECLARING A CURSOR: IN THIS PROCESS WE DEFINE A CURSOR.

SYNTAX:

DECLARE CURSOR <CURSORNAME> IS < SELECT STATEMENT>;

2)OPENING A CURSOR: WHEN WE OPEN A CURSOR, IT WILL INTERNALLY EXECUTE THE SELECT STATEMENT THAT IS ASSOCIATED WITH THE CURSOR DECLARTION AND LOAD THE DATA INTO CURSOR.

SYNTAX:

OPEN < CURSORNAME>;

<u>3)FETCHING DATA FROM THE CURSOR:</u> IN THIS PROCESS WE ACCESS ROW BY ROW FROM CURSOR.

SYNTAX:

FETCH <CURSORNAME> INTO <VARIABLES>;

<u>4)CLOSING A CURSOR:</u> IN THIS PROCESS, IT RELEASES THE CURRENT RESULT SET OF THE CURSOR LEAVING THE DATASTRUCTURE AVAILABLE FOR REOPENING.

SYNTAX:

CLOSE <CURSORNAME>;

<u>ATTRIBUTES OF EXPLICIT CURSORS:</u>

IT SHOWS STATUS OF THE CURSOR AND IT RETURNS BOOLEAN VALUE.

SYNTAX:

<CURSOR_NAME>%<ATTRIBUTE>;

<u>A. %ISOPEN:</u>

IT RETURNS TRUE, WHEN THE CURSOR OPENS SUCCESSFULLY.

<u>B. %FOUND:</u>

IT RETURNS TRUE, WHEN THE CURSOR CONTAINS DATA.

<u>C. %NOTFOUND:</u>

IT RETURNS TRUE, WHEN THE CURSOR DOESN'T FIND ANY DATA.

<u>D.%ROWCOUNT:</u>

IT RETURS NO. OF FETCH STATEMENTS EXECUTED.RETURN TYPE IS NUMBER.

EX1:

WA CURSOR PROGRAM TO FETCH A SINGLE ROW FROM EMP TABLE?

SOL:

```
DECLARE CURSOR C1 IS SELECT ENAME, SAL FROM EMP;
V_ENAME VARCHAR2(10);
V_SAL NUMBER (10);
BEGIN
OPEN C1;
FETCH C1 INTO V_ENAME, V_SAL;
DBMS_OUTPUT.PUT_LINE(V_ENAME||','||V_SAL);
CLOSE C1;
END;
/
```

OUTPUT:

```
SMITH,800
```

EX2: TO FETCH MULTIPLE ROWS FROM EMP TABLE BY USING LOOPING STATEMENTS?

<u>I) BY USING SIMPLE LOOP:</u>

- IT IS AN INFINITE LOOP.SO THAT WE NEED BREAK A LOOP THEN WE ARE USING "EXIT" STATEMENT.

SOL:

```
 DECLARE CURSOR C1 IS SELECT ENAME, SAL FROM EMP;
 V_ENAME VARCHAR2(10);
 V_SAL NUMBER (10);
 BEGIN
 OPEN C1;
 LOOP
 FETCH C1 INTO V_ENAME, V_SAL;
 EXIT WHEN C1%NOTFOUND;
 DBMS_OUTPUT.PUT_LINE(V_ENAME||','||V_SAL);
 END LOOP;
 CLOSE C1;
 END;
 /
```

```
OUTPUT:
SMITH,800
ALLEN,1600
WARD,1250
.................
..................
```

<u>II) BY USING WHILE LOOP:</u>

```
DECLARE CURSOR C1 IS SELECT ENAME, SAL FROM EMP;
 V_ENAME VARCHAR2(10);
 V_SAL NUMBER (10);
 BEGIN
 OPEN C1;
 FETCH C1 INTO V_ENAME, V_SAL; ---LOOP START FROM 1ST ROW
 WHILE(C1%FOUND)
 LOOP
 DBMS_OUTPUT.PUT_LINE(V_ENAME||','||V_SAL);
 FETCH C1 INTO V_ENAME, V_SAL; ---LOOP CONTINUE UPTO LAST
ROW
 END LOOP;
 CLOSE C1;
 END;
 /
```

<u>III) BY USING FOR LOOP:</u>

```
DECLARE CURSOR C1 IS SELECT ENAME, SAL FROM EMP;
BEGIN
FOR I IN C1
LOOP
DBMS_OUTPUT.PUT_LINE (I. ENAME||','||I.SAL);
END LOOP;
END;
/
```

<u>NOTE:</u> WHENEVER WE ARE USING "FOR LOOP" STATEMENT IN CURSOR FOR FETCHING ROWS FROM A CURSOR MEMORY THEN THERE NO NEED TO OPEN CURSOR, FETCH ROW FROM CURSOR AND CLOSE CURSOR BY EXPLICITLY BECAUSE INTERNALLY ORACLE SERVER WILL OPEN, FETCH AND CLOSE CURSOR BY IMPLICITLY.

HERE, FOR LOOP EXECUTE NO. OF TIMES DEPENDS ON NO. OF ROWS IN CURSOR(C1). EVERY TIME FOR LOOP IS EXECUTE AND FETCH A ROW FROM C1 AND

ASSIGNED / STORED IN LOOP VARIABLE (I) AND LATER I LOOP VARIABLE VALUES ARE PRINTED.

EX3:

WA CURSOR PROGRAM TO FETCH TOP FIVE HIGHEST SALARIES EMPLOYEE ROWS FROM EMP TABLE?

SOL:

```
DECLARE CURSOR C1 IS SELECT ENAME, SAL FROM EMP ORDER BY
SAL DESC;
V_ENAME VARCHAR2(10);
V_SAL NUMBER (10);
BEGIN
OPEN C1;
LOOP
FETCH C1 INTO V_ENAME, V_SAL;
EXIT WHEN C1%ROWCOUNT>5;
```

```
DBMS_OUTPUT.PUT_LINE(V_ENAME||','||V_SAL);
END LOOP;
CLOSE C1;
END;
/
```

<u>OUTPUT:</u>

```
KING,5000
FORD,3000
SCOTT,3000
JONES,2975
BLAKE,2850
```

EX4:

WA CURSOR PROGRM TO FETCH EVEN POSITION ROWS FROM EMP TABLE?

SOL:

```
DECLARE CURSOR C1 IS SELECT EMPNO, ENAME FROM EMP;
V_EMPNO NUMBER (10);
V_ENAME VARCHAR2(10);
BEGIN
OPEN C1;
LOOP
FETCH C1 INTO V_EMPNO, V_ENAME;
EXIT WHEN C1%NOTFOUND;
IF MOD(C1%ROWCOUNT,2) =0 THEN
DBMS_OUTPUT.PUT_LINE(V_EMPNO||','||V_ENAME);
END IF;
END LOOP;
CLOSE C1;
END;
/
```

OUTPUT:

```
7499, ALLEN
7566, JONES
7698, BLAKE
7788, SCOTT
7844, TURNER
```

EX5: WA CURSOR PROGRAM TO FETCH 9TH POSITION ROW FROM EMP TABLE?

SOL:

```
DECLARE CURSOR C1 IS SELECT ENAME FROM EMP;
V_ENAME VARCHAR2(10);
BEGIN
OPEN C1;
LOOP
FETCH C1 INTO V_ENAME;
EXIT WHEN C1%NOTFOUND;
IF C1%ROWCOUNT=9 THEN
DBMS_OUTPUT.PUT_LINE(V_ENAME);
END IF;
END LOOP;
CLOSE C1;
END;
/
```

OUTPUT:

```
KING
```

PARAMETERIZED CURSORS:

- WHENEVER WE ARE PASSING PARAMETERS TO THE CURSOR AT THE TIME DECLARATION IS CALLED AS PARAMETERIZED CURSOR. THESE PARAMETERIZED CURSOR WANT TO DECLARE THEN WE FOLLOW THE FOLLOWING TWO STEPS ARE

STEP1: DECLARE PARAMETERIZED CURSOR:

SYNTAX:

DECLARE CURSOR <CURSOR NAME> (<PARAMETER NAME> <DATATYPE>, ……..) IS SELECT * FROM <TN> WHERE <CONDITION>;

STEP2: OPEN PARAMETERIZED CURSOR:

SYNTAX:

OPEN <CURSOR NAME> (<PARAMETER NAME> / <VALUE>);

EX1:

WA CURSOR PROGRAM TO ACCEPT DEPTNO AS A PARAMETER AND DISPLAY THE NO. OF EMPLOYEE WORKING IN THE GIVEN DEPTNO FROM EMP TABLE?

SOL:

```
DECLARE CURSOR C1(P_DEPTNO NUMBER) IS SELECT ENAME,
DEPTNO FROM EMP

WHERE DEPTNO=P_DEPTNO;

V_ENAME VARCHAR2(10);

V_DEPTNO NUMBER (10);

BEGIN

OPEN C1(&P_DEPTNO);

LOOP

FETCH C1 INTO V_ENAME, V_DEPTNO;

EXIT WHEN C1%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(V_ENAME||','||V_DEPTNO);
```

```
END LOOP;
CLOSE C1;
END;
/
```

OUTPUT:

```
ENTER VALUE FOR P_DEPTNO: 10
MILLER,10
CLARK,10
KING,10
```

EX2: WA CURSOR PROGRAM TO ACCEPT EMPNO AS A PARAMETER AND CHECK THAT EMPLOYEE IS EXISTS OR NOT EXISTS IN EMP TABLE?

SOL:

```
DECLARE CURSOR C1(P_EMPNO NUMBER) IS SELECT ENAME FROM EMP
WHERE EMPNO=P_EMPNO;
V_ENAME VARCHAR2(10);
BEGIN
OPEN C1(&P_EMPNO);
FETCH C1 INTO V_ENAME;
IF C1%FOUND THEN
DBMS_OUTPUT.PUT_LINE ('EMPLOYEE EXISTS, NAME IS: -'||V_ENAME);
ELSE
DBMS_OUTPUT.PUT_LINE ('EMPLOYEE NOT EXISTS');
END IF;
CLOSE C1;
END;
/
```

OUTPUT:

```
ENTER VALUE FOR P_EMPNO: 7788
EMPLOYEE EXISTS, NAME IS: -SCOTT
```

IMPLICIT CURSOR:

- THESE CURSOR ARE DECLARING BY ORACLE SERVER BY DEFAULT.ORACLE DECLARE THESE CURSOR AFTER EXECUTION OF DML COMMAND (INSERT / UPDATE / DELETE).

- IMPLICIT CURSOR TELLING US THE STATUS OF LAST DML COMMAND WHETHER SUCCESSFULL OR NOT.

ATTRIBUTES OF IMPLICIT CURSORS:

1. %ISOPEN:

- IT RETURNS TRUE THEN CURSOR SUCCESSFUL OPEN OTHERWISE RETURNS FALSE.

2. %NOTFOUND:

- IT RETURNS TRUE THEN LAST DML COMMAND IS FAIL OTHERWISE RETURNS FALSE.

3. %FOUND:

- IT RETRUNS TRUE THEN LAST DML COMMAND IS SUCCESSFULLY EXECUTED

OTHERWISE RETURNS FALSE.

4.%ROWCOUNT:

- IT RETURNS NO. OF ROWS AFFECTED BY LAST DML COMMAND.

EX:

```
DECLARE
V_EMPNO NUMBER (10);
BEGIN
V_EMPNO: =&V_EMPNO;
DELETE FROM EMP WHERE EMPNO=V_EMPNO;
IF SQL%FOUND THEN
DBMS_OUTPUT.PUT_LINE ('RECORD IS DELETED');
ELSE
DBMS_OUTPUT.PUT_LINE ('RECORD IS NOT EXISTS');
END IF;
END;
/
```

OUTPUT:

```
ENTER VALUE FOR V_EMPNO: 7788
RECORD IS DELETED
```

REF. CURSORS:

- WHEN WE ASSIGN "SELECT STATEMENT" AT THE TIME OF OPENING CURSOR IS CALLED AS "REF.CURSOR".

- REF.CURSORS ARE TWO TYPES.THOSE ARE,

1. **WEAK REF.CURSOR**
2. **STRONG REF.CURSOR**

1. WEAK REF.CURSOR:

WHEN WE DECLARE REF.CURSOR WITHOUT RETURN TYPES IS CALLED AS WEAK REF.CURSOR.

SYNTAX:

<CURSOR VARIABLE NAME>   SYS_REFCURSOR; ------> (IT IS PRE-DEFINE TYPE)

EX:

```
DECLARE
C1 SYS_REFCURSOR;
I EMP%ROWTYPE;
BEGIN
OPEN C1 FOR SELECT * FROM EMP WHERE DEPTNO=10;
LOOP
FETCH C1 INTO I;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (I.
EMPNO||','||I.ENAME||','||I.SAL||','||I.DEPTNO);
END LOOP;
CLOSE C1;
END;
/
```

<u>2)STRONG REF.CURSOR:</u>

- WHEN WE DECLARE A REF. CURSOR ALONG WITH RETURN TYPE IS CALLED AS STRONG REF.CURSOR.

<u>CREATE A USER DEFINE STRONG REF.CURSOR DATATYPE:</u>

SYNTAX:

TYPE <TYPE NAME> IS REF CURSOR RETURN <TYPE>;---->(IT IS USER-DEFINE TYPE)

EX:

```
DECLARE
TYPE UD_REFCURSOR IS REF CURSOR RETURN EMP%ROWTYPE;
C1 UD_REFCURSOR;
I EMP%ROWTYPE;
BEGIN
OPEN C1 FOR SELECT * FROM EMP WHERE DEPTNO=10;
LOOP
```

```
FETCH C1 INTO I;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (I.
EMPNO||','||I.ENAME||','||I.SAL||','||I.DEPTNO);
END LOOP;
CLOSE C1;
END;
/
```

EX. OF WEAK CURSOR ON MULTIPLE TABLES:

```
DECLARE
C1 SYS_REFCURSOR;
I EMP%ROWTYPE;
J DEPT%ROWTYPE;
V_DEPTNO NUMBER (10): =&V_DEPTNO;
BEGIN
IF V_DEPTNO = 10 THEN
OPEN C1 FOR SELECT * FROM EMP WHERE DEPTNO=10;
LOOP
FETCH C1 INTO I;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (I.
EMPNO||','||I.ENAME||','||I.SAL||','||I.DEPTNO);
END LOOP;
ELSIF V_DEPTNO = 20 THEN
OPEN C1 FOR SELECT * FROM DEPT WHERE DEPTNO=20;
LOOP
FETCH C1 INTO J;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (J. DEPTNO||','||J.DNAME||','||J.LOC);
END LOOP;
```

```
CLOSE C1;
END IF;
END;
/
```

DIFFERENCES B/W WEAK AND STRONG REF.CURSOR:

WEAK REF CURSOR	STRONG REF CURSOR
1. IT IS NOT DECLARE WITH "RETURN" TYPE.	1. DECLARING WITH "RETURN" TYPE.
2. PRE-DEFINE TYPE IS AVAILABLE	2. PRE-DEFINE TYPE IS NOT AVAILABLE THATS WHY WE ARE CREATING "USED DEFINE TYPE".
3. IT CAN ACCESS ROWS OF ANY TYPE OF TABLE (MORE THAN ONE TABLE)	3. IT CAN ACCESS ROWS OF A SPECIFIC TABLE ONLY.

EX. OF STRONG CURSOR ON MULTIPLE TABLES:(NOT SUPPORTING)

```
DECLARE
TYPE UD_REFCURSOR IS REF CURSOR RETURN EMP%ROWTYPE;
C1 UD_REFCURSOR;
I EMP%ROWTYPE;
J DEPT%ROWTYPE;
V_DEPTNO NUMBER (10): =&V_DEPTNO;
BEGIN
IF V_DEPTNO = 10 THEN
OPEN C1 FOR SELECT * FROM EMP WHERE DEPTNO=10;
LOOP
```

```
FETCH C1 INTO I;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (I.
EMPNO||','||I.ENAME||','||I.SAL||','||I.DEPTNO);
END LOOP;
ELSIF V_DEPTNO = 20 THEN
OPEN C1 FOR SELECT * FROM DEPT WHERE DEPTNO=20;
LOOP
FETCH C1 INTO J;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE (J. DEPTNO||','||J.DNAME||','||J.LOC);
END LOOP;
CLOSE C1;
END IF;
END;
/
```