What is a database cursor? I guess, is the question which majority of us have faced at least once in our life either during our college studies, or job interview or while doing oracle certification. Database cursor is an important topic from oracle certification as well as from job interview perspective. Thus I am writing this blog by taking both the perspectives in mind so that you will get good marks in your exams as well as ace your Job interview.

**What is Database Cursor?**

Cursor is a pointer to a memory area called context area. This context area is a memory region inside the Process Global Area or PGA assigned to hold the information about the processing of a SELECT statement or DML Statement such as INSERT, DELETE, UPDATE or MERGE.

***A quick tip:***  
*Refrain from saying that cursor is a pointer to the data stored in the database. Saying this in your interview will definitely put you in an indefinite queue of candidates who never receive a call back.  Because cursor is the pointer to the memory area called context area not to the data of the database.*

**What is the Context Area?**

Let’s dig a little deeper and see what the context area is?

The context area is a special memory region inside the Process Global Area or PGA which helps oracle server in processing an SQL statement by holding the important information about that statement.

This information includes:

* Rows returned by a query.
* Number of rows processed by a query.
* A pointer to the parsed query in the shared pool.

Using cursor you can control the context area as it is a pointer to the same.

***A Quick Tip****:*   
*Parsing an SQL statement is the term used for the process that includes the transferring of information to the server, whereby the SQL statement is evaluated as being valid.*

**Advantages of Cursors**

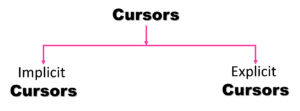
There are two main advantages of a cursor:

1. Cursor is names thus you can reference it in your program whenever you want.
2. Cursor allows you to fetch and process rows returned by a SELECT statement by a row at a time.

**Types of cursors in oracle database:**

There are two types of cursors in oracle database:

1. Implicit cursor
2. Explicit cursor



**Implicit Cursors in Oracle Database**

As the name suggests these are the cursors which are automatically created by the oracle server every time an SQL DML statement is executed. User cannot control the behavior of these cursors. Oracle server creates an implicit cursor in the background for any PL/SQL block which executes an SQL statement as long as an explicit cursor does not exist for that SQL statement.

Oracle server associates a cursor with every DML statement. Each of the Update & Delete statements has cursors which are responsible to identify those set of rows that are affected by the operation. Also the implicit cursor fulfills the need of a place for an Insert statement to receive the data that is to be inserted into the database.

***Info Byte:***   
***The Most recently opened cursor is called SQL Cursor.***

**Explicit Cursor in oracle database**

In contrast to implicit cursors, we have explicit cursors. Explicit cursors are user defined cursors which means user has to create these cursors for any statement which returns more than one row of data. Unlike implicit cursor user has full control of explicit cursor. An explicit cursor can be generated only by naming the cursor in the declaration section of the PL/SQL block.

**Advantages of Explicit Cursor:**

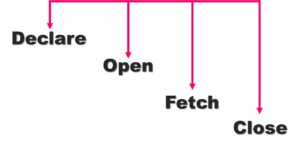
1. Since Explicit cursors are user defined hence they give you more programmatic control on your program.
2. Explicit cursors are more efficient as compared to implicit cursors as in latters case it is hard to track data errors.

**Steps for creating an Explicit Cursor**

To create an explicit cursor you need to follow 4 steps. These 4 steps are:

* Declare
* Open
* Fetch
* Close

In case of implicit cursors oracle server performs all these steps automatically for you.



**Info Byte:   
Unless the complete cycle of declaring, opening, fetching and closing has been performed, you can’t use a cursor.**

**Declare: How To Declare a Database Cursor?**

Declaring a cursor means initializing a cursor into memory. You define explicit cursor in declaration section of your PL/SQL block and associate it with the SELECT statement.

**Syntax**

CURSOR cursor\_name IS select\_statement;

**Open: How to Open a Database Cursor?**

Whenever oracle server comes across an ‘Open Cursor’ Statement the following four steps take place automatically.

1. All the variables including bind variables in the WHERE clause of a SELECT statement are examined.
2. Based on the values of the variables, the active set is determined, and the PL/SQL engine executes the query for that cursor. Variables are examined at cursor open time.
3. The PL/SQL engine identifies the active set of data.
4. The active set pointer sets to the first row.

***Active set: Rows from all the involved tables that meet the WHERE clause criteria.***

**Syntax**

OPEN cursor\_name;

**Fetch: How to retrieve data from cursor?**

The process of retrieving the data from the cursor is called fetching. Once the cursor is declared and opened then you can retrieve the data from it. Let’s see how.

**Syntax**

FETCH cursor\_name INTO PL/SQL variable;

Or

FETCH cursor\_name INTO PL/SQL record;

**What happens when you execute fetch command of a cursor?**

1. The use of a FETCH command is to retrieve one row at a time from the active set. This is usually done inside a loop. The values of each row in the active set can then be stored in the corresponding variables or PL/SQL record one at a time, performing operations on each one successively.
2. After completion of each FETCH, the active set pointer is moved forward to the next row. Therefore, each FETCH returns successive rows of the active set, until the entire set is returned. The last FETCH does not assign values to the output variables as they still contain their previous values.

**Close: How To Close a Database Cursor?**

Once you are done working with your cursor it’s advisable to close it. As soon as the server comes across the closing statement of a cursor it will relinquish all the resources associated with it.

**Syntax**

CLOSE cursor\_name;

**Info Byte:**   
**You can no longer fetch from a cursor once it’s closed. Similarly it is impossible to close a cursor once it is already closed. Either of these actions will result in an Oracle error.**

**Basic Programming Structure of the Cursor**

Here is the basic programming structure of the cursor in oracle database.

DECLARE

CURSOR cursor\_name IS select\_statement;

BEGIN

OPEN cursor\_name;

FETCH cursor\_name INTO PL/SQL variable [PL/SQL record];

CLOSE cursor\_name;

END;

/

**Cursor Attributes**

Oracle provides four attributes which work in correlation with cursors. These attributes are:

* %FOUND
* %NOTFOUND
* %ISOPEN
* %ROWCOUNT

First three attributes ‘Found’, ‘NotFound’ and ‘IsOpen’ are Boolean attributes whereas the last one ‘RowCount’ is a numeric attribute.

Let’s quickly take a look at all these attributes.

**%Found**

Cursor attribute ‘Found’ is a Boolean attribute which returns TRUE if the previous FETCH command returned a row otherwise it returns FALSE.

**%NotFound**

‘Not Found’ cursor attribute is also a Boolean attribute which returns TRUE only when previous FETCH command of the cursor did not return a row otherwise this attribute will return FALSE.

**%IsOpen**

Again ‘Is Open’ Cursor attribute is a Boolean attribute which you can use to check whether your cursor is open or not. It returns TRUE if the cursor is open otherwise it returns FALSE.

**%RowCount**

Row count cursor attribute is a numeric attribute which means it returns a numeric value as a result and that value will be the number of records fetched from a cursor at that point in time.

So, this is the detailed blog on cursors in oracle database. Hope it will help you with your exams as well as with your job interview. Do make sure to share this article on your social media or you can share this blog on your twitter.

Thanks for reading & have a great day!