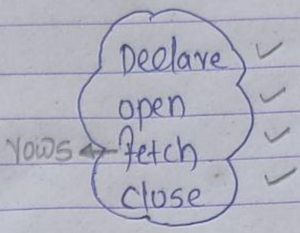
Cursors in PL/SQL ✓ (Important w.r.t interview preparation)

• A cursor holds multiple rows returned by a SQL statement

i) Implicit cursor (generated by oracle)

ii) Explicit cursor

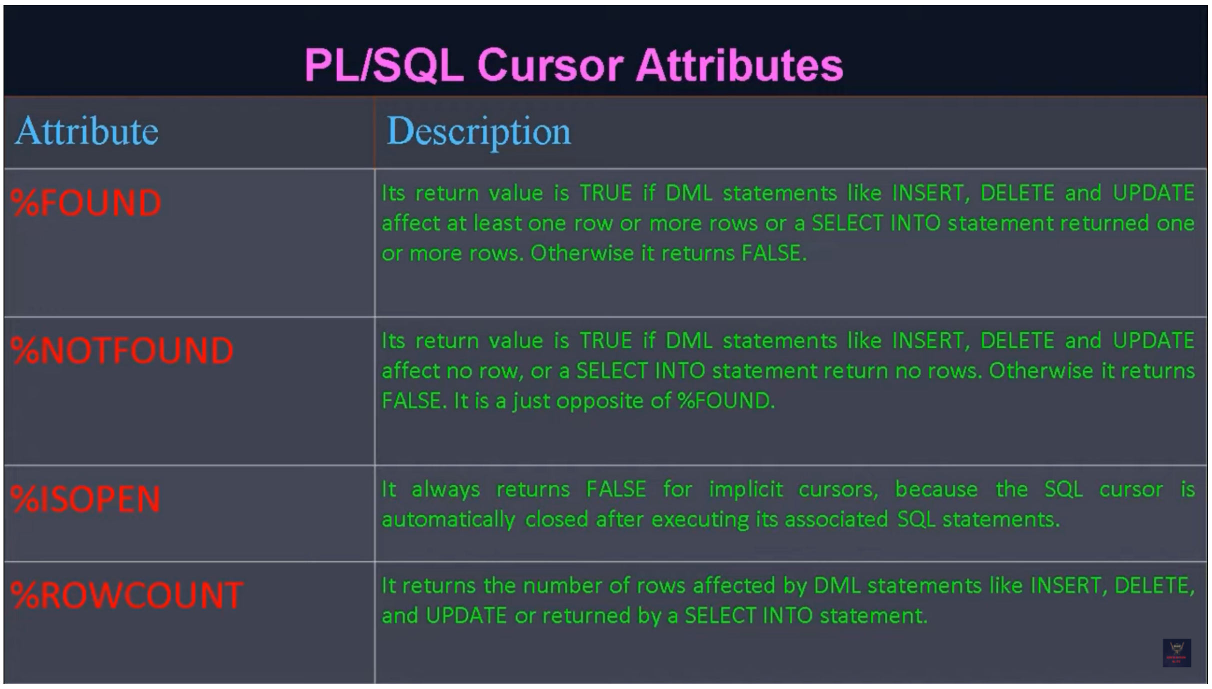
```
CURSOR  C1  IS  select statement
OPEN  C1
FETCH  C1  INTO  -----
CLOSE  C1
```



* Example :

same datatype given here also

```
DECLARE
    C-id      customers.id%type;
    C-name    customers.name%type;
    CURSOR  C1  IS
            SELECT id,name from customers;
BEGIN
   OPEN C1;
   LOOP
      FETCH C1 INTO C-id, C-name;
      EXIT when C1 %.NOTFOUND;
      DBMS-OUTPUT.PUT-LINE(C-id ||'  '||C.name);
   END LOOP;
   CLOSE C1;
END;
```



PL/SQL Cursor Attributes

Attribute	Description
%FOUND	Its return value is TRUE if DML statements like INSERT, DELETE and UPDATE affect at least one row or more rows or a SELECT INTO statement returned one or more rows. Otherwise it returns FALSE.
%NOTFOUND	Its return value is TRUE if DML statements like INSERT, DELETE and UPDATE affect no row, or a SELECT INTO statement return no rows. Otherwise it returns FALSE. It is a just opposite of %FOUND.
%ISOPEN	It always returns FALSE for implicit cursors, because the SQL cursor is automatically closed after executing its associated SQL statements.
%ROWCOUNT	It returns the number of rows affected by DML statements like INSERT, DELETE, and UPDATE or returned by a SELECT INTO statement.

```
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

EXAMPLE-1:-(Implicit Cursor)

```
mysql> create database Dhiraj;
Query OK, 1 row affected (0.01 sec)

mysql> use Dhiraj;
Database changed
mysql> CREATE TABLE tutorials (
    ->    ID INT PRIMARY KEY,
    ->    TITLE VARCHAR(100),
    ->    AUTHOR VARCHAR(40),
    ->    DATE VARCHAR(40)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> insert into tutorials values(5, 'Cassandra', 'Pruthvi', '2019-04-06');
```

```
Query OK, 1 row affected (0.01 sec)

mysql> insert into tutorials values(1, 'Java', 'Krishna', '2019-09-01');
Query OK, 1 row affected (0.00 sec)

mysql> insert into tutorials values(2, 'JFreeCharts', 'Satish', '2019-05-01');
Query OK, 1 row affected (0.00 sec)

mysql> insert into tutorials values(3, 'JavaSprings', 'Amit', '2019-05-01');
Query OK, 1 row affected (0.00 sec)

mysql> insert into tutorials values(4, 'Android', 'Ram', '2019-03-01');
Query OK, 1 row affected (0.00 sec)

mysql> CREATE TABLE backup (
   ->    ID INT,
   ->    TITLE VARCHAR(100),
   ->    AUTHOR VARCHAR(40),
   ->    DATE VARCHAR(40)
   -> );
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER //
mysql> CREATE PROCEDURE ExampleProc()
   ->    BEGIN
   ->       DECLARE done INT DEFAULT 0;
   ->       DECLARE tutorialID INTEGER;
   ->       DECLARE tutorialTitle, tutorialAuthor, tutorialDate VARCHAR(20);
   ->       DECLARE cur CURSOR FOR SELECT * FROM tutorials;
   ->       DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
   ->       OPEN cur;
   ->       label: LOOP
   ->       FETCH cur INTO tutorialID, tutorialTitle, tutorialAuthor, tutorialDate;
   ->       INSERT INTO backup VALUES(tutorialID, tutorialTitle, tutorialAuthor, tutorialDate);
   ->       IF done = 1 THEN LEAVE label;
   ->       END IF;
   ->       END LOOP;
   ->       CLOSE cur;
   ->    END//
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> CALL ExampleProc;
Query OK, 1 row affected (0.02 sec)

mysql> select * from backup;
+------+-------------+---------+------------+
| ID   | TITLE       | AUTHOR  | DATE       |
+------+-------------+---------+------------+
|    1 | Java        | Krishna | 2019-09-01 |
|    2 | JFreeCharts | Satish  | 2019-05-01 |
|    3 | JavaSprings | Amit    | 2019-05-01 |
|    4 | Android     | Ram     | 2019-03-01 |
|    5 | Cassandra   | Pruthvi | 2019-04-06 |
|    5 | Cassandra   | Pruthvi | 2019-04-06 |
+------+-------------+---------+------------+
6 rows in set (0.00 sec)

mysql> select * from tutorials;
+----+-------------+---------+------------+
| ID | TITLE       | AUTHOR  | DATE       |
+----+-------------+---------+------------+
|  1 | Java        | Krishna | 2019-09-01 |
|  2 | JFreeCharts | Satish  | 2019-05-01 |
|  3 | JavaSprings | Amit    | 2019-05-01 |
|  4 | Android     | Ram     | 2019-03-01 |
|  5 | Cassandra   | Pruthvi | 2019-04-06 |
+----+-------------+---------+------------+
5 rows in set (0.00 sec)
```

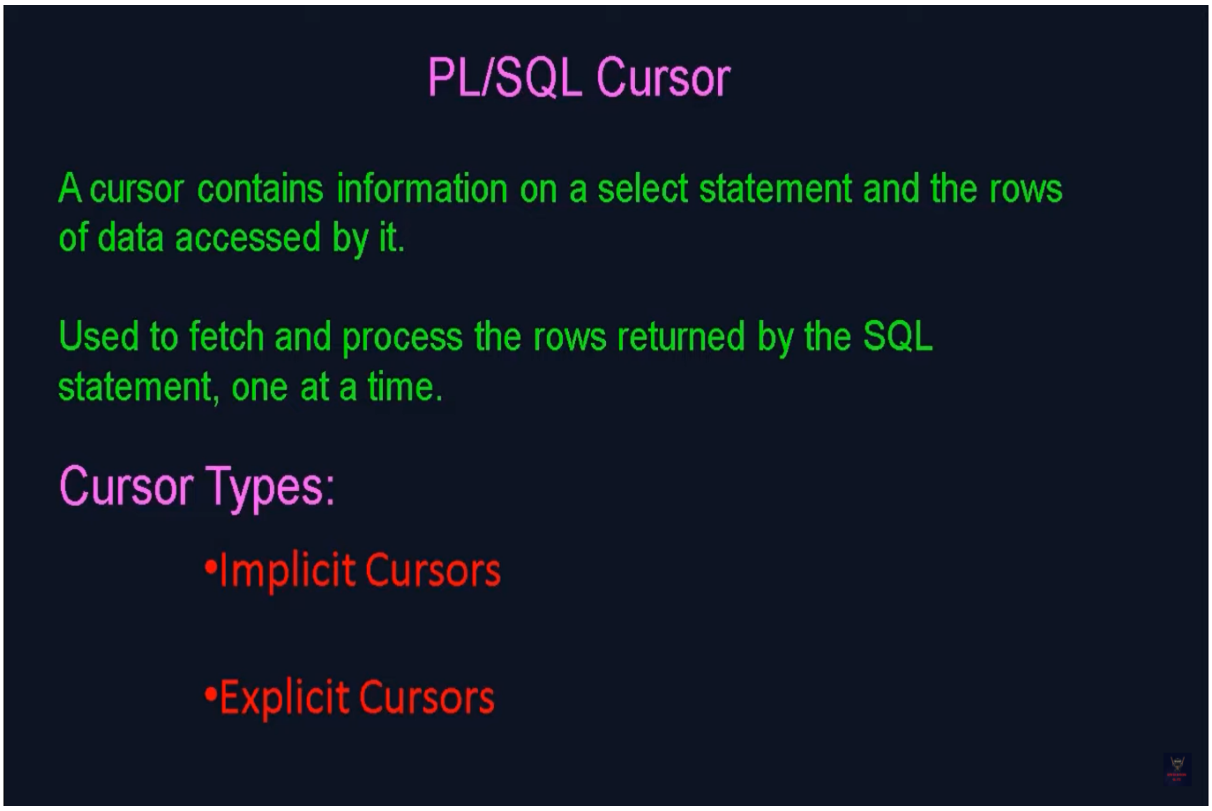
EXAMPLE-2:-(Implicit Cursor)

```
Select * from customers;

+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
+----+----------+-----+-----------+----------+
```

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;
/
```

```
Select * from customers;

+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2500.00 |
|  2 | Khilan   |  25 | Delhi     |  2000.00 |
|  3 | kaushik  |  23 | Kota      |  2500.00 |
|  4 | Chaitali |  25 | Mumbai    |  7000.00 |
|  5 | Hardik   |  27 | Bhopal    |  9000.00 |
|  6 | Komal    |  22 | MP        |  5000.00 |
+----+----------+-----+-----------+----------+
```



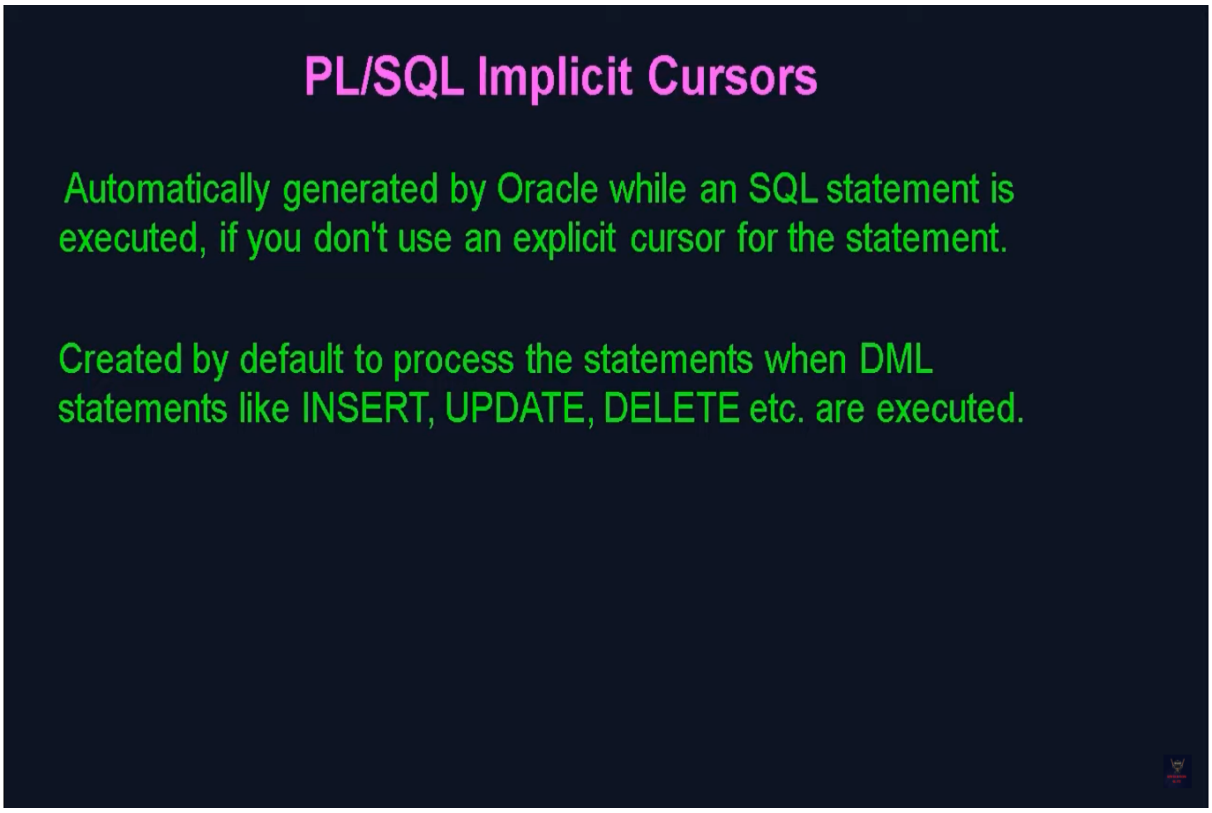
PL/SQL Cursor

A cursor contains information on a select statement and the rows of data accessed by it.

Used to fetch and process the rows returned by the SQL statement, one at a time.

Cursor Types:

- Implicit Cursors
- Explicit Cursors



PL/SQL Implicit Cursors

Automatically generated by Oracle while an SQL statement is executed, if you don't use an explicit cursor for the statement.

Created by default to process the statements when DML statements like INSERT, UPDATE, DELETE etc. are executed.