CURSORES EN PL/SQL

Valeria Beratto Ulloa

Qué son?

- PL/SQL utiliza cursores para gestionar las instrucciones **SELECT**.
- Un cursor **es un conjunto de registros devuelto por una instrucción SQL**.
- Técnicamente, los cursores son fragmentos de memoria que reservados para procesar los resultados de una consulta **SELECT**.

Tipos de cursores

- Implícitos => INTO => un valor
- Explícitos => Declarar cursor => n valores

Cursores Implícitos

- SELECT arroja un registro como resultado

Consideraciones:

- *Con cada cursor implícito debe existir la palabra clave **INTO**.*
- *Las variables que reciben los datos devueltos por el cursor tienen que contener el mismo tipo de dato que las columnas de la tabla. Se recomienda usa %TYPE o %ROWTYPE*
- *En caso de que se devuelva más de una fila (o ninguna fila) se producirá una excepción.*

Cursores Implícitos - Excepciones

Excepción	Explicación
NO_DATA_FOUND	Se produce cuando una sentencia SELECT intenta recuperar datos pero ninguna fila satisface sus condiciones. Es decir, cuando "no hay datos"
TOO_MANY_ROWS	Dado que cada cursor implícito sólo es capaz de recuperar una fila , esta excepción detecta la existencia de más de una fila.

Cursores Implícitos - Ejemplo

```
DECLARE
cantidad integer;
BEGIN
SELECT count(1) INTO cantidad
FROM empleados;

DBMS_OUTPUT.PUT_LINE('La cantidad de empleados
es ' || cantidad);

END;
```

Cursores Explícitos

- Acepta una o más filas resultados
- Para trabajar se debe:
  - *Declarar =>**DECLARE***
  - *Abrir =>* ***OPEN nombre_cursor***
  - *Recuperar datos =>**FETCH nombre_cursor INTO variables***
  - *Cerrar =>* ***CLOSE nombre_cursor***

Declarar Cursor

- Se define como otra variable en DECLARE

```
declare
cursor c_emp is
SELECT nombre, direccion
FROM empleados;
begin
/* Sentencias del bloque ...*/
end;
```

- Admite parámetros

```
declare
cursor c_emp (depto in integer)
is
SELECT nombre, direccion
FROM empleados;
WHERE cod_depto= depto;
begin
/* Sentencias del bloque ...*/
end;
```

Cursores Explícitos - Ejemplo

```
DECLARE
  CURSOR c_emp
IS
  SELECT nombre, direccion, cod_dep
  FROM empleados;


 nombre VARCHAR2(20);
 direccion VARCHAR2(50);
 depto INTEGER;

BEGIN
  OPEN c_emp;
  FETCH c_emp INTO nombre,direccion,depto;
  CLOSE c_emp;
END;
```


```
var_emp empleados%ROWTYPE
```

* Para trabajar con los valores del cursor es necesario el uso de ciclos

Cursores Explícitos – Atributos

Atributo	Explicación
%NOTFOUND	TRUE si la recuperación más reciente no devuelve ninguna fila
%FOUND	TRUE si la recuperación más reciente devuelve una fila
%ISOPEN	TRUE si el cursor está abierto
%ROWCOUNT	Proporciona el número total de filas devueltas hasta ese momento

Cursores Explícitos – Atributos Sintaxis

```
IF NOT c_emp%ISOPEN THEN
      OPEN c_emp;
END IF;
```

```
LOOP
        FETCH c_emp into var_emp;
        …
EXIT WHEN c_emp%NOTFOUND
END LOOP;
```

Cursores Explícitos – Atributos Sintaxis

Realice la sintaxis para el ciclo WHILE

```
Recordatorio:

WHILE condición LOOP
Sentencia 1;
…
END LOOP
```

Cursores Explícitos – FOR

Al utilizar un FOR, se realizan implícitamente ejecuta OPEN, FETCH, CLOSE

```
BEGIN
FOR cemp IN (SELECT * FROM
empleados)
LOOP
dbms_output.put_line(cemp.nombre
);
END LOOP;
END
```

CURSORES DE ACTUALIZACIÓN

- Se incorpora en la declaración del cursor el comando FOR UPDATE
- Para actualizar los datos del cursor hay que ejecutar una sentencia **UPDATE** especificando la clausula **WHERE CURRENT OF** <*cursor_name*>.

Ejemplo CURSOR ACTUALIZACIÓN

```
DECLARE
CURSOR cemp IS
select nombre, direccion, cod_depto
from empleado where cod_depto=2
FOR UPDATE;
nom empleado.nombre%TYPE;
dir empleado.direccion%TYPE;
Cod_depto empleado.cod_depto%TYPE;


BEGIN
OPEN cemp;
FETCH cemp INTO nom,dir,cod_depto;
WHILE cemp%found
        LOOP
        UPDATE empleado
        SET cod_depto = 1
        WHERE CURRENT OF cemp;
        FETCH cemp INTO nom,dir,cod_depto;
END LOOP;
CLOSE cemp;
COMMIT;

END;
```

Ejercicio

- Considerando el siguiente esquema

Empleado (cod_emp, nombre, salario, cod_dep)

Departamento (cod_depto, nombre_depto)

```
create table Departamento (cod_depto number primary key, nombre_depto varchar2(20));

create table Empleado (cod_emp number primary key, nombre varchar2(20), salario number, cod_dep number, foreign key
(cod_dep) REFERENCES departamento);
```

- Realice un cursor que imprima por pantalla el nombre de los departamento y de los empleados que recibe los salarios más altos.

- Realice un cursor que permita calcular la cantidad de empleados de cada departamento, que reciban menos del sueldo ingresado por teclado, mostrando el resultado por pantalla

Ejercicio 1

- Realice un cursor que imprima por pantalla el nombre de los departamento y de los empleados que recibe los salarios más altos.

```
DECLARE
cursor c_emp is
select nombre_depto, nombre
from Empleado E, Departamento D where E.cod_dep=D.cod_depto and
E.salario = (select max(salario) from empleado);
depto Departamento.nombre_depto%TYPE;
empl  Empleado.nombre%TYPE;

BEGIN
open c_emp;
fetch c_emp into  depto, empl;
while c_emp%found
   LOOP
       dbms_output.put_line(depto || ' ' || empl);
       fetch c_emp into  depto, empl;
   end loop;
close c_emp;
end;
```

Ejercicio 2

- Realice un cursor que permita calcular la cantidad de empleados de cada departamento, que reciban menos del sueldo ingresado por teclado, mostrando el resultado por pantalla

```
DECLARE
cursor cant_emp (sal_con in number) is
select nombre_depto, count(*)
from Empleado E, Departamento D where E.cod_dep=D.cod_depto and
E.salario<=sal_con group by nombre_depto;
depto Departamento.nombre_depto%TYPE;
Cant number;
Sal_con number;

BEGIN
Sal_con:= &Ingrese_valor;
open cant_emp(Sal_con);
fetch cant_emp into  depto, cant;
while cant_emp%found
   LOOP
       dbms_output.put_line(depto || ' ' || cant);
       fetch cant_emp into  depto, cant;
   end loop;
close cant_emp;
end;
```

Ejercicios para trabajo autónomo

- Cree un cursor que permita mostrar por pantalla la cantidad de empleados que tiene cada uno de los departamentos.
- Cree un programa que permita definir el cuartil según la cantidad de empleados según los siguientes cuartil

Cantidad de Empleados	Cuartil
0 – 10	Cuartil 1
11-30	Cuartil 2
30 o 40	Cuartil 3
41 o más	Cuartil 4

- Cree un programa que permita aumentar el salario en un 15% a las personas que ganan menos de 300