

CONCEPTO Y USO DE CURSORES

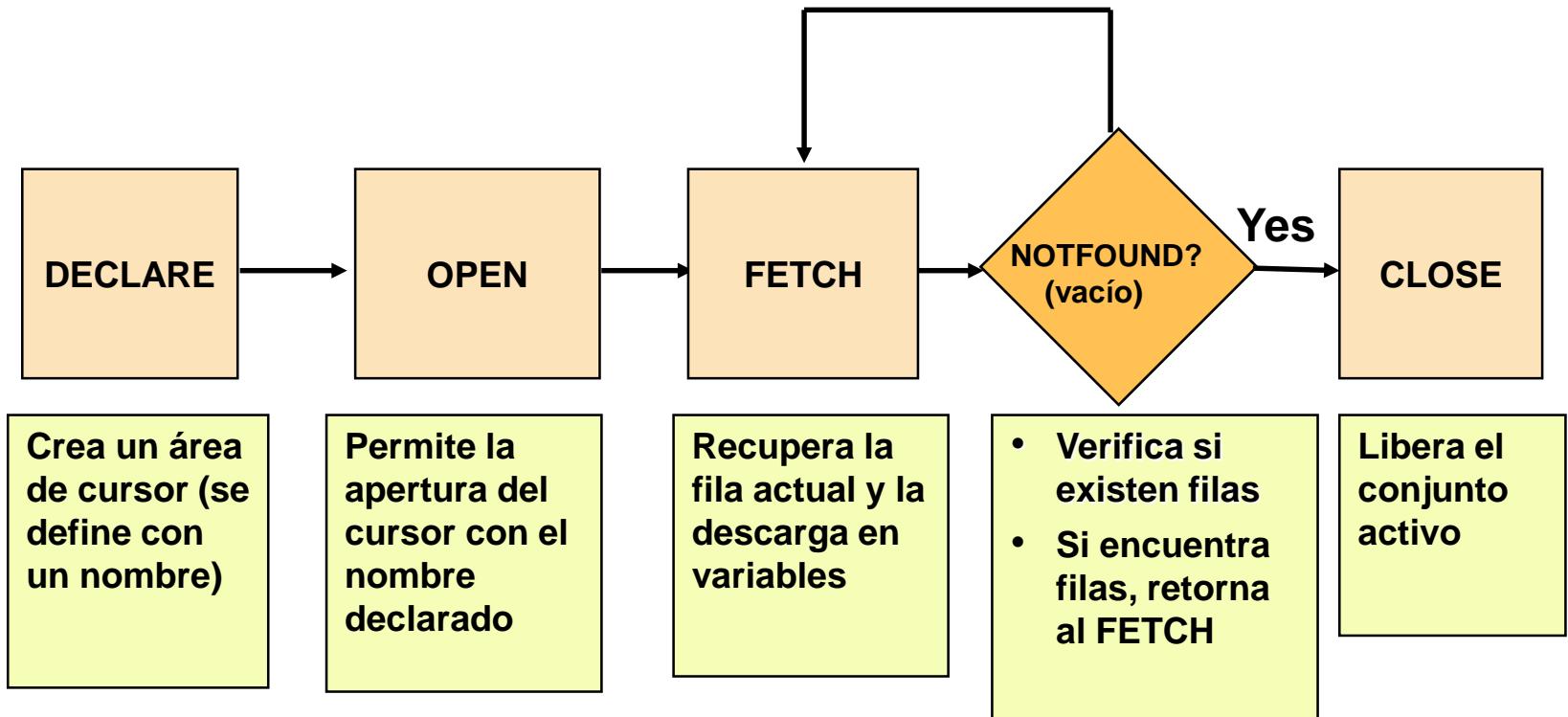
Cursor SQL

- Un CURSOR es un área de trabajo privada para acceder a las filas recuperadas por una sentencia SELECT de SQL
- Hay dos tipos de cursores:
 - **Implícitos:** se declaran automáticamente para las sentencias DML y SELECT.
ORACLE utiliza cursores implícitos para efectuar el parse (análisis) y ejecutar los comandos SQL.
 - **Explícitos:** Declarados y nombrados explícitamente por el programador. Se puede declarar un cursor de este tipo para manejar individualmente las filas devueltas por consultas que devuelven más de una fila.

Cursos explícitos

- Un cursor explícito requiere una declaración que define la sentencia SELECT en cualquier parte de la sección declarativa de un bloque PL/SQL. Se define con un nombre (los cursos implícitos no se declaran y la manera de referenciarse a ellos es con el prefijo 'SQL')
- Permite un control más preciso en el procesamiento de las filas recuperadas con la sentencia SQL

Control de cursos explícitos

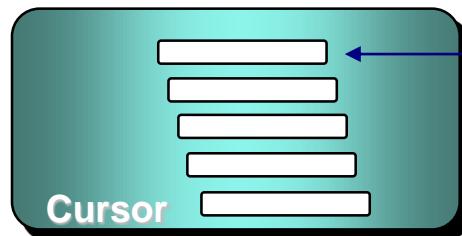


Controlando Cursores Explícitos

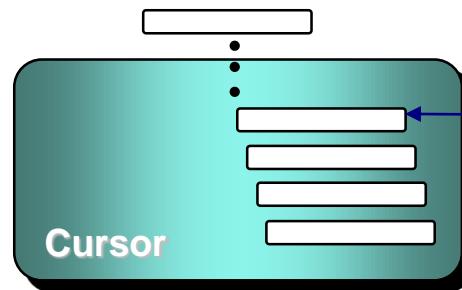
Fetch (recuperación)
de una fila por el
cursor

Continúa hasta que el
archivo esté vacío

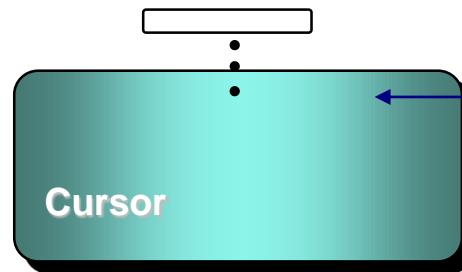
Abre el cursor



Pointer
(Puntero
o Indicador)



Puntero



Puntero

SINTAXIS DE LA DECLARACIÓN

- Declarar el cursor, es darle un nombre y asociarlo a una consulta específica

```
DECLARE  
    CURSOR nombre_cursor IS  
        Sentencia_select;
```

- Se puede incluir variables en las sentencias SELECT que serán sustituidas por valores que permitirán variar el cursor
- No debe incluir la cláusula INTO dentro de la declaración del cursor

Declarando un Cursor: Ejemplo

- Declarando un cursor para obtener la identificación del artículo, nombre del artículo y cantidad a partir del detalle de ventas de un ID de venta dado.

```
DECLARE
    V_ID_VENTA B_DETALLE_VENTAS.Id_Venta%TYPE := 1;
    V_ID        B_ARTICULOS.ID%TYPE;
    V_NOMBRE    B_ARTICULOS.NOMBRE%TYPE;
    V_CANTIDAD  B_DETALLE_VENTAS.CANTIDAD%TYPE;
CURSOR C_VENTAS IS
    SELECT v.id_articulo, a.nombre, v.cantidad
    FROM   b_Detalle_ventas v, b_articulos a
    WHERE  a.id = v.id_articulo AND
           v. id_venta = v_id_venta;
....
```

Apertura del Cursor: Sintaxis

- Al abrir el cursor, se ejecuta la consulta y se identifica el "result set", que consiste en cada fila que cumple con la condición where de la consulta:

```
OPEN nombre_cursor;
```

Si el Query no devuelve ninguna fila, no se levanta una excepción.

Las filas del cursor no son devueltas al ejecutar el OPEN, sino mediante la sentencia FETCH.

```
OPEN C_VENTAS;
```

Recorrido del Cursor: Sintaxis

- FETCH recupera una por una las filas devueltas por la consulta: toma la fila actual y avanza el cursor a la siguiente fila del *result set*.
- Se puede guardar el valor de cada columna por separado, en variables distintas, o almacenar la fila completa en un registro declarado utilizando %ROWTYPE.
- Normalmente se utiliza FETCH dentro de un ciclo LOOP-EXIT WHEN.

Leyendo Datos del Cursor: Fetch - Ejemplo

Recupere los datos del cursor definido anteriormente sobre el detalle de ventas:

```
FETCH C_VENTAS INTO V_ID, V_NOMBRE, V_CANTIDAD;
```

Cerrando el Cursor: Sintaxis

- Cierre el cursor después de completar el procesamiento de las filas
- Si lo requiere, vuelva a abrir el cursor: esto ejecutará nuevamente el query
- No intente realizar operaciones una vez que el cursor ha sido cerrado. Esto levantará la excepción INVALID_CURSOR

```
CLOSE nombre_cursor;
```

Atributos de Cursorres Explícitos

- Los atributos retornan información sobre la ejecución de un *query*. Para utilizarlos, se preceden con el nombre del cursor como prefijo:

Atributo	Tipo	Descripción
%ISOPEN	Boolean	VERDADERO cuando el cursor está abierto
%NOTFOUND	Boolean	VERDADERO cuando el FETCH no retorna fila alguna
%FOUND	Boolean	VERDADERO cuando el FETCH retorna alguna fila
%ROWCOUNT	Number	Devuelve el número total de filas retornadas hasta el momento

Controlando Múltiples Fetch

- Para procesar varias filas del cursor utilice un LOOP
- Lea (Con FETCH) una fila con cada iteración
- Escriba una prueba para determinar si existen más filas que procesar con el atributo %NOTFOUND de la siguiente forma:

IF nombre_cursor%NOTFOUND

- Si no se tienen filas que procesar debe forzar la salida del LOOP

El Atributo % ISOPEN : Ejemplo

- Recupere filas cuando el cursor está abierto.
- Pruebe si el cursor está abierto mediante el atributo %ISOPEN antes de realizar un Fetch:

```
IF C_VENTAS%ISOPEN THEN  
    FETCH C_VENTAS INTO V_ID, V_NOMBRE, V_CANTIDAD;  
ELSE  
    OPEN C_VENTAS;  
END IF;
```

Los Atributos % NOTFOUND y % ROWCOUNT: Ejemplo

- Recupere un número exacto de filas utilizando el atributo %ROWCOUNT
- Determine cuándo terminar el Loop utilizando el atributo %NOTFOUND

```
LOOP  
    FETCH C_VENTAS INTO V_ID, V_NOMBRE, V_CANTIDAD;  
    EXIT WHEN C_VENTAS%ROWCOUNT > 5  
        OR C_VENTAS%NOTFOUND;  
END LOOP;
```

Cursos y Registros: Ejemplo

Procese las filas del conjunto activo convenientemente leyendo valores en un REGISTRO PL/SQL

```
DECLARE
    CURSOR C_VENTAS IS
        SELECT v.id_articulo, a.nombre, v.cantidad
        FROM b_Detalle_ventas v, b_articulos a
        WHERE a.id = v.id_articulo AND
              v.id_venta = 1;
    R_VENTAS C_VENTAS%ROWTYPE;

BEGIN
    OPEN C_VENTAS;
    LOOP
        FETCH C_VENTAS INTO R_VENTAS;
        EXIT WHEN C_VENTAS%ROWCOUNT > 5
            OR C_VENTAS%NOTFOUND;
    END LOOP;
    CLOSE C_VENTAS;
END;
```

Cursor con Parámetros: Sintaxis

- Pase valores de los parámetros a un cursor cuando el cursor se abre y se ejecuta el *query*

```
CURSOR nombre_cursor
    [ (nombre_parametro tipo_dato, ...) ]
IS
sentencia_select;
```

- Esto le permite abrir un cursor explícito varias veces con un conjunto activo diferente cada vez

Cursor con Parámetros: Ejemplo

Pase el id de departamento y cargo a la cláusula WHERE.

```
DECLARE
    CURSOR C_VENTAS(P_ID NUMBER) IS
        SELECT v.id_articulo, a.nombre, v.cantidad
        FROM b_Detalle_ventas v, b_articulos a
        WHERE a.id = v.id_articulo AND
              v.id_venta = P_ID;
    R_VENTAS C_VENTAS%ROWTYPE;

BEGIN
    OPEN C_VENTAS(1);
```

Cursor FOR Loop: Sintaxis

- En los cursores FOR LOOP, las sentencias OPEN, FETCH y CLOSE son implícitas

```
FOR nombre_registro IN nombre_cursor LOOP  
    sentencia1;  
    sentencia2;  
    . . .  
END LOOP;
```

- NO SE DECLARA EL REGISTRO QUE CONTROLA EL LOOP. Su ámbito es solo el del LOOP y por tanto se declara automáticamente

Cursor FOR Loop: Ejemplo

Recupere todos los artículos vendidos en una fecha dada.

```
DECLARE
    CURSOR C_VENTAS IS
        SELECT v.id_articulo, a.nombre, v.cantidad
        FROM b_Detalle_ventas v JOIN b_Ventas c
        ON c.id = v.id_venta JOIN b_articulos a
        ON a.id = v.id_articulo
        WHERE TRUNC(c.fecha) = TO_DATE('02/01/2011', 'DD/MM/YYYY');
BEGIN
    FOR R_VENTAS IN C_VENTAS LOOP -- No hacer OPEN ni FETCH!!
        DBMS_OUTPUT.PUT_LINE (R_VENTAS.NOMBRE || '-'
        || TO_CHAR(R_VENTAS.CANTIDAD, '999G999'));
    END LOOP; --No hacer el CLOSE
END;
```

Cursor FOR LOOP utilizando Subqueries

Recupere artículos para una orden de p_proyecto_suministro, uno por uno, hasta que no existan más artículos:

```
FOR rec_item IN (select id, nombre from b_articulos  
                  where stock_actual < stock_minimo) LOOP  
  
END LOOP; -- CLOSE implícito
```

Cláusula WHERE CURRENT OF

- Permite ACTUALIZAR o BORRAR una fila del propio cursor
- Al incluir la sentencia FOR UPDATE, Oracle lleva el conjunto a ser recuperado

```
SELECT...FROM...FOR UPDATE [OF columna] [NOWAIT]
```

- La fila actual de un cursor explícito es referida usando la cláusula WHERE CURRENT OF

Cláusula WHERE CURRENT OF Ejemplo

```
DECLARE
...
CURSOR C_ARTICULOS IS
    SELECT * FROM B_ARTICULOS FOR UPDATE;
BEGIN
    FOR REC_ART IN C_ARTICULOS LOOP
        UPDATE B_ARTICULOS
        SET PRECIO := ROUND(PRECIO * 1,03)
        WHERE CURRENT OF C_ARTICULOS;
    END LOOP;
    COMMIT;
END;
```