Bases de Datos

Unidad 6:

Programación de bases de datos

Sesión 7

8.- Cursores

Los cursores nos van a permitir resolver problemas que simplemente no se pueden resolver con instrucciones sql, o bien, que nos facilitan mucho su solución.

Por ejemplo, pensemos en la siguiente situación.

EL PROBLEMA:

¿Cómo podemos obtener los datos del automóvil más caro de cada marca?

EL PROBLEMA:

¿Cómo podemos obtener los datos de los dos automóviles más caros de cada marca?

¿Podríamos obtener dichos datos con una sola consulta?

Con un procedimiento, te podrías plantear realizarlo con lo que sabes hasta ahora.

¿Cómo debería ser el cuerpo de dicho procedimiento?

EL PROBLEMA:

La solución a problemas como estos está en usar los **CURSORES** dentro de procedimientos o dentro de funciones o triggers.

De momento, piensa que la solución es ir obteniendo cada marca y, por cada marca, obtener los dos automóviles más caros.

LA SOLUCIÓN SIN CURSORES

```
CREATE PROCEDURE sincursor()
BEGIN
declare c int default 0;
declare mar varchar(15);
declare nummarcas int;
-- obtenemos cuantas marcas hay
select count(distinct marca) into nummarcas from automoviles;
-- bucle que se repite tantas veces como marcas haya
-- se obtienen las marcas sin repetir y se saca la que está en la posición c
-- se obtienen los dos coches más caros de la marca extraída
while c <nummarcas do
        select distinct marca into mar from automoviles limit c,1;
        select * from automoviles where marca=mar order by precio desc limit 2;
        set c=c+1;
end while;
END
```

LA SOLUCIÓN USANDO UN CURSOR

```
CREATE PROCEDURE concursor()
BEGIN
declare c int default 0;
declare mar varchar(15);
declare nummarcas int;
declare curMarcas cursor for select distinct marca from automoviles;
select count(distinct marca) into nummarcas from automoviles;
-- declaración del cursor, se asocia a una consulta
-- apertura del cursor. En el cursor quedan todos los datos de la consulta
-- El cursor tiene tantas filas como filas devuelve la consulta asociada
open curMarcas;
while c <nummarcas do
        -- acceso a siguiente fila del cursor y carga de los datos en variables
        fetch curMarcas into mar;
        -- esta select que usa limit podría implementarse con otro cursor
        select * from automoviles where marca=mar order by precio desc limit 2;
        set c=c+1;
end while;
-- cerrar el cursor
close curMarcas;
END
```

LA SOLUCIÓN USANDO UN CURSOR

¿Qué ventajas o inconvenientes aporta el uso de cursores?

LA SOLUCIÓN USANDO UN CURSOR

Ventajas:

1. Ahorra tiempo, porque no necesita esperar el procesamiento y la descarga del conjunto de registros completo.

2. Ahorra memoria, tanto en el servidor como en el cliente porque no tienen que dedicar una gran cantidad de memoria a los conjuntos de resultados.

3. El streaming o ráfaga de datos deja espacio para otras operaciones.

4. Permite operaciones en tablas consultadas (bajo ciertas condiciones) que no afectan directamente a su cursor. Mientras se mantiene un cursor en una fila, otros procesos pueden leer, actualizar e incluso eliminar otras filas. Esto es especialmente útil en tablas con muchos accesos simultáneos.

LA SOLUCIÓN USANDO UN CURSOR

Inconvenientes:

1. Un cursor, no opera en una instantánea consistente de los datos, sino en una fila. Por lo tanto, sus garantías de Atomicidad, Consistencia, Aislamiento y Durabilidad (ACID) se producen únicamente a nivel de fila, no de tabla. Es posible, en caso de necesidad, bloquear la tabla completa sobre la que actúa el cursor, pero esto repercute negativamente en el rendimiento del servidor.

2. La transmisión de cada fila por sí sola puede ser muy ineficiente, ya que cada paquete tiene una sobrecarga de negociación que podría evitar enviando grandes fragmentos de datos.

3. Es difícil crear buenos cursores. Por ejemplo, considera una consulta con un gran conjunto de resultados, que requiera el uso de un cursor, que usa una cláusula GROUP BY con funciones agregadas. El GROUP BY reducirá drásticamente el rendimiento del servidor, porque tiene que generar y almacenar todo el conjunto de resultados a la vez, tal vez incluso manteniendo bloqueos en otras tablas.

LA SOLUCIÓN USANDO UN CURSOR

Regla de oro:

- Si vamos a trabajar en conjuntos de resultados pequeños y creados rápidamente, es preferible evitar el uso de cursores.

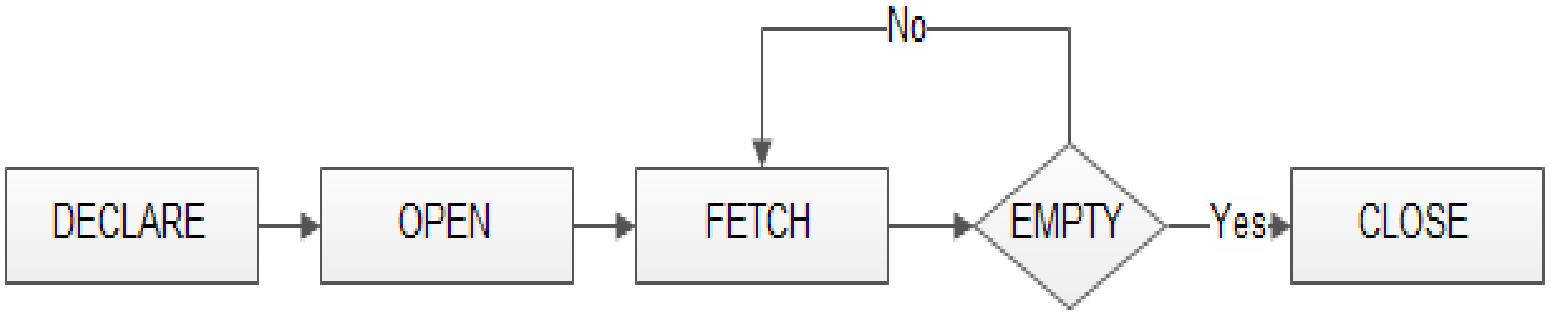
- Los cursores destacan en **consultas** complejas, de naturaleza secuencial **con grandes conjuntos de resultados** y **bajos requisitos de consistencia**.

QUÉ ES UN CURSOR

Un cursor es un objeto que permite tener cargados los datos de una hoja de resultados de una consulta y a los que se puede acceder fila a fila de la hoja de resultados o, lo que es lo mismo, del cursor.

Un cursor puede usarse en procedimientos, en funciones y en triggers.

MANEJO DE UN CURSOR



DECLARACIÓN DE UN CURSOR

Se realiza con la instrucción **DECLARE**. En la declaración, se especifica el nombre del cursor y la consulta que va a cargar los resultados en el cursor. La declaración de los cursores debe ir a continuación de las declaraciones de variables, pero antes de la declaración de los manejadores.

SINTAXIS:

DECLARE nombreCursor CURSOR FOR instrucción_select;

EJEMPLOS:

DECLARE curMarcas CURSOR FOR SELECT DISTINCT marca FROM automoviles;

DECLARE curPartidos CURSOR FOR SELECT eqloc, eqvis, golesloc, golesvis FROM partidos WHERE numjornada = n;

ABRIR UN CURSOR

Al abrir un cursor, los resultados de la consulta asociada al cursor se cargan en el cursor.

Para abrir un cursor, se usa la sintaxis: **OPEN *nombreCursor;***

EJEMPLO:

```
declare curMarcas cursor for select distinct marca from automoviles;
…….....
OPEN curMarcas;
```

curMarcas

marca
Audi
BMW
Citroen
Ford
Mercedes
Opel
Renault
Seat

ACCEDER A LOS DATOS DE UNA FILA DE UN CURSOR

Para acceder a los datos de una fila de los resultados cargados en el cursor, se usa la instrucción FETCH.

La primera vez que se ejecuta FETCH, una vez abierto el cursor, se tiene acceso a la primera fila de la hoja de resultados. La segunda vez que se ejecuta, se avanza a la fila siguiente, por lo que se tiene acceso a la los datos de la segunda fila de la hoja de resultados.

SINTAXIS:

FETCH nombreCursor INTO listaVariables;

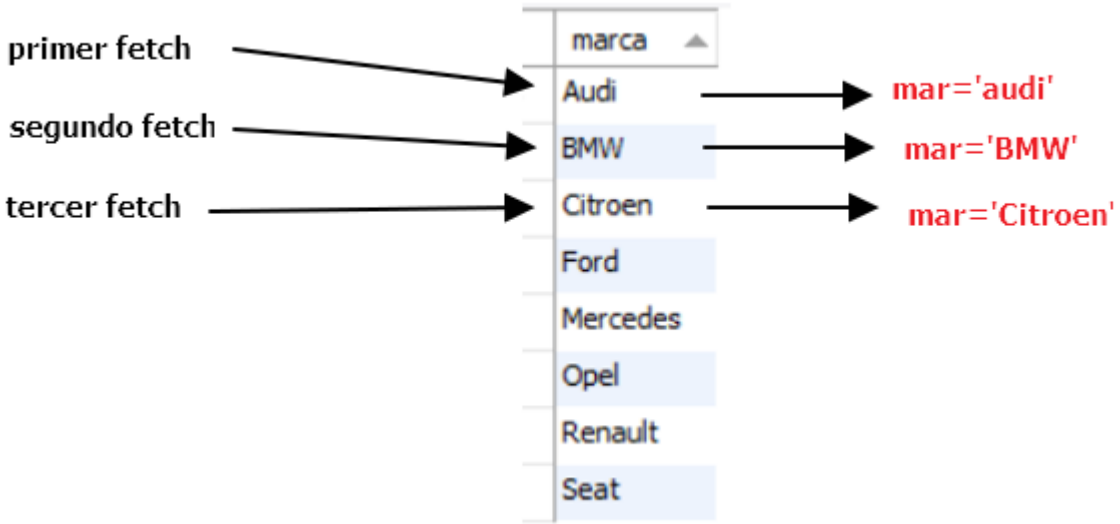
Si el cursor está asociado a una consulta que devuelve una sola columna de resultados, en listaVariables sólo habrá una variable.

Si el cursor está asociado a una consulta que devuelve N columnas de resultados, en listaVariables habrá N variables.

ACCEDER A LOS DATOS DE UN CURSOR

EJEMPLO:

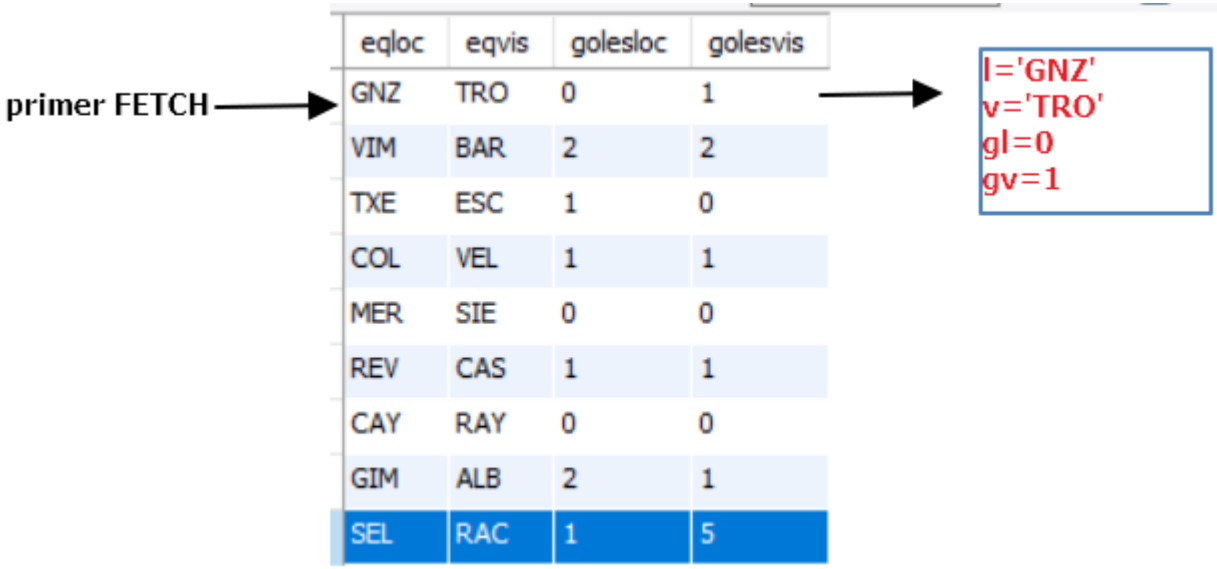
```
declare curMarcas cursor for select distinct marca from automoviles;
open curMarcas;
set c=0;
while c <3 do
        fetch curMarcas into mar;
        ……
        set c=c+1;
end while;
```



ACCEDER A LOS DATOS DE UN CURSOR

EJEMPLO:

```
declare curPartidos cursor for select eqloc, eqvis, golesloc, golesvis from partidos where numjornada = 1;
open curPartidos;
set c=0;
while c <10 do
        fetch curPartidos into l, v, gl, gv;
        ……
        set c=c+1;
end while;
```



CERRAR UN CURSOR

Todo cursor que haya sido abierto en una rutina, debe ser cerrado.

Si no se cierra, estará consumiendo recursos (memoria) ya que los datos cargados en el cursor seguirán almacenados en memoria.

Tradicionalmente, aunque se hubiera terminado la rutina o procedimiento que contenía un cursor, este permanecía en memoria. Con las últimas versiones de MySQL, todo cursor no cerrado explícitamente se cierra automáticamente al alcanzarse el END, del bloque BEGIN…END donde fue creado. No obstante, es recomendable, y una buena práctica, cerrar un cursor cuando ya no es necesario.

SINTAXIS de CIERRE de un CURSOR:

CLOSE nombreCursor;