**AUTOMATIZACIÓN DE TAREAS: CONSTRUCCIÓN DE GUIONES**

**SQL EN RUTINAS: CURSORES**

Hasta ahora la mayoría de los ejemplos contenían instrucciones o expresiones referidas a cálculos matemáticas o de cadenas sin implicar el uso de una base de datos.

Sin embargo, normalmente el uso de procedimientos implica manipular datos de tablas de bases de datos lo que implica usar instrucciones SQL.

En general, podemos usar cualquier instrucción SQL de las que hemos utilizado hasta ahora.

Vamos a ver un ejemplo. Creo el siguiente procedimiento en el esquema prueba:

**CREATE PROCEDURE ejemplo1()**

**BEGIN**

**DECLARE i INT DEFAULT 1;**

**/\*Creación tabla\*/**

**DROP TABLE IF EXISTS tabla1;**

**CREATE TABLE tabla1 (id INT PRIMARY KEY, dato1 VARCHAR(30));**

**WHILE i<=10 DO**

**INSERT INTO tabla1 VALUES (i,CONCAT("registro ",i));**

**SET i=i+1;**

**END WHILE;**

**/\*Actualización tabla\*/**

**SET i=5;**

**UPDATE tabla1 SET dato1=CONCAT ("FILA ",i,” ACTUALIZADA”)**

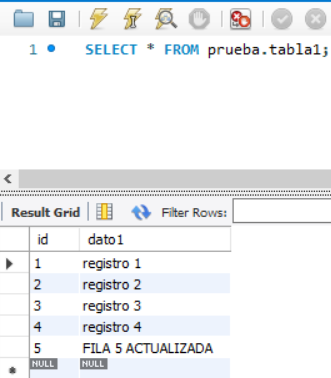
**WHERE id=i;**

**/\*Borrado registros\*/**

**DELETE FROM tabla1 WHERE id>i;**

**END;**

El resultado después de ejecutar este procedimiento será el siguiente:



Otro ejemplo:

**CREATE DEFINER=`root`@`localhost` PROCEDURE `ejemplo2`(dato int)**

**BEGIN**

**DECLARE a int;**

**DECLARE b VARCHAR(25);**

**if dato>6 THEN**

**SELECT ("El género tiene que estar entre 1 y 6") AS MENSAJE;**

**ELSE**

**SELECT idgenero, nombre INTO a,b FROM otrabiblio2019.genero**

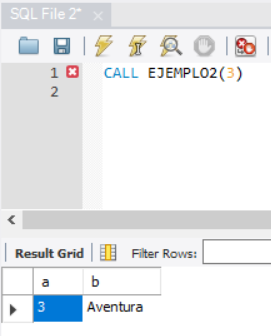
**where idgenero=dato;**

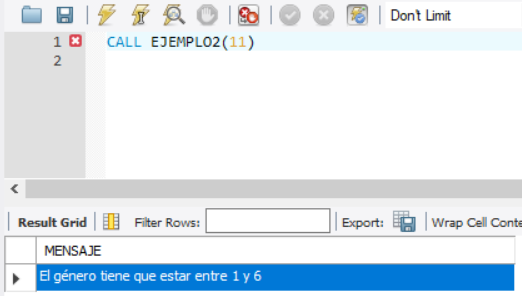
**SELECT a,b;**

**END IF;**

**End**

En este caso, lo que haremos será pedir un dato de entrada y se guardarán en las variables a y b, los valores que tuviera la tabla genero del esquema otrabiblio para el idgenero igual al dato introducido. La sentencia IF informa del rango de valores permitido, puesto que sólo había 6 géneros. El resultado que obtendremos al ejecutar dicho procedimiento será el siguiente:





En el caso anterior, recuperamos sólo los valores de una fila de la tabla genero. Pero ¿qué ocurre cuando queremos recuperar todas las filas de una tabla? En este caso tenemos que utilizar **cursores.**

**CURSORES**

Los cursores me van a permitir recorrer una tabla de principio a fin para recuperar los valores que hay en sus filas. Un cursor va a ser como un puntero que va a comenzar apuntando al primer registro de la tabla y va a ir pasando, fila por fila, hasta el final de esta.

Un cursor se asocia con un conjunto de filas o con una consulta sobre una tabla de la base de datos y siempre se va a definir dentro de un procedimiento.

Un cursor se define de la siguiente forma:

**DECLARE nombrecursor CURSOR FOR SELECT….**

Le puedes dar cualquier nombre al cursor, procura que no coincida con palabras reservadas o con nombres de variable, que no tenga espacios en blanco…

Y debe hacerse después de declarar todas las variables necesarias para el procedimiento. **PRIMERO SE DECLARAN LAS VARIABLES Y LUEGO EL CURSOR.**

Los cursores siempre van asociados a una sentencia SELECT. Dicha sentencia indicará que valores va a devolver el cursor. Tendremos que definir tantas variables como valores incluya el SELECT, por supuesto, de su mismo tipo.

**Comandos relacionados con cursores:**

Para manipular los cursores disponemos de los siguientes comandos:

**OPEN:** Abre el cursor e inicializa el conjunto de resultados asociados con el cursor.

OPEN nombrecursor

**FETCH:** Extrae la siguiente fila de valores del conjunto de resultados del cursor, moviendo su puntero interno una posición.

FETCH nombrecursor INTO lista\_de\_variables;

**CLOSE:** Cierra el cursor, liberando la memoria que ocupa y haciendo imposible el acceso a cualquiera de sus datos.

CLOSE nombrecursor;

En el siguiente ejemplo vemos cómo extraer una sola fila de una tabla:

**CREATE DEFINER=`root`@`localhost` PROCEDURE `ejemplocursor`()**

**BEGIN**

**DECLARE a int; /\*variable del mismo tipo que idgenero\*/**

**DECLARE b VARCHAR(25); /\*variable del mismo tipo que nombregenero\*/**

**DECLARE cursor1 CURSOR**

**FOR SELECT idgenero, nombre FROM genero; /\*el cursor va a devolver idgenero y nombre\*/**

**OPEN cursor1; /\*abro el cursor\*/**

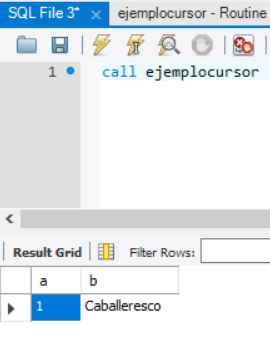
**FETCH cursor1 INTO a,b;/\*guardo los datos del SELECT del cursor en las variables a y b\*/**

**SELECT a,b; /\*muestro a y b\*/**

**CLOSE cursor1; /\*cierro el cursor\*/**

**END**

El resultado será el siguiente:



Para el caso de más de una fila necesitaremos un bucle, que es el que me va a permitir hacer que el puntero se vaya incrementando para ir pasando desde la primera fila hasta la última.

**CURSOR CON WHILE** (Es el tipo de bucle más usado porque, a diferencia de los otros, evalúa la condición antes de leer un registro del cursor, para saber si la tabla está vacía antes de comenzar a recorrerla)

**EJEMPLO DE CURSOR CON WHILE**

Pongo a continuación cómo sería el procedimiento. En amarillo, están marcados los elementos nuevos añadidos, cuya explicación viene a continuación del cuerpo del procedimiento.

**CREATE PROCEDURE `ejcursor4`()**

**BEGIN**

**DECLARE a INT;**

**DECLARE var boolean; /\*variable que me va a permitir saber cuándo he acabado de leer la tabla\*/**

**DECLARE b VARCHAR(25);**

**DECLARE cursor4 CURSOR**

**FOR SELECT idgenero, nombre FROM genero;**

**DECLARE CONTINUE HANDLER FOR NOT FOUND SET var=1;**

**SET var=0;**

**OPEN cursor4;**

**WHILE var=0 DO**

**FETCH cursor4 INTO a,b;**

**if var=0 then**

**select a,b;**

**END IF;**

**END while;**

**CLOSE cursor4;**

**END**

La variable VAR es una variable booleana, es decir, va a contener valor 1 o 0. Siempre la inicializamos a 0 antes de entrar en el cursor.

La sentencia DECLARE CONTINUE HANDLER FOR NOT FOUND SET var=1; hace dos cosas:

* Establecer el valor de la variable booleana a 1 (CUANDO ENCUENTRA FINAL TABLA)
* Permitir al programa continuar la ejecución fuera del bucle.

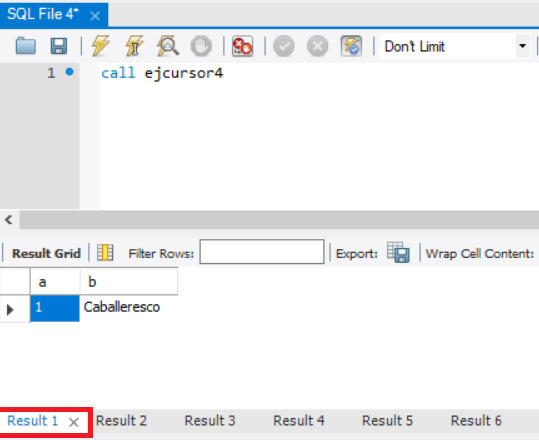
Si traducimos esa sentencia, obtendremos algo parecido a “Declara un manejador continuo o puntero que se va a mover hasta que encuentre el final de la tabla, en cuyo caso pondrá la variable booleana var a 1”. Esto indicará al cursor que ya ha leído todos los valores de la tabla.

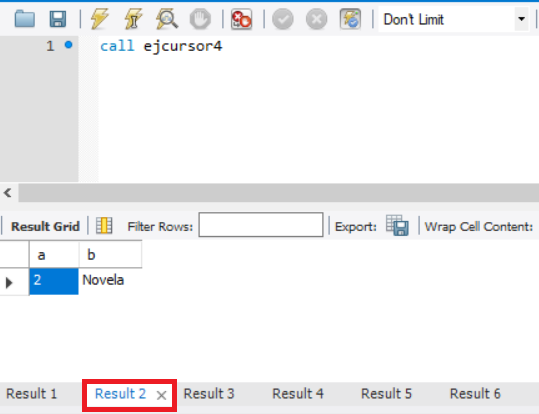
Es necesario ponerlo puesto que al llegar a la última fila de la tabla no hay más datos que obtener, por lo que habrá que detectar cuando se produce esta situación.

Vemos que, después del FETCH, el cursor muestra los datos mientras var=0, puesto que cuando sea 1 no va a tener datos que mostrar.

Los cursores no siempre tienen que usarse para mostrar datos. Podemos usar procedimientos con cursores también para guardar valores, que después se pasarán a otros programas o aplicaciones para que trabajen con ellas.

El resultado que obtendremos al ejecutarlo será el siguiente:





En cada una de las pestañas, saldrá un género diferente, hasta los 6 que tenemos.

**CURSOR CON REPEAT UNTIL**

Es similar al anterior, pero, en este caso, la condición se evalúa al final en lugar de al principio.

**EJEMPLO DE CURSOR CON REPEAT**

**CREATE PROCEDURE `ejcursor3`()**

**BEGIN**

**DECLARE a INT(11);**

**DECLARE var boolean;**

**DECLARE b VARCHAR(25);**

**DECLARE cursor3 CURSOR FOR SELECT idgenero, nombre FROM otrabibli02019.genero;**

**DECLARE CONTINUE HANDLER FOR NOT FOUND SET var=1;**

**SET VAR=0;**

**OPEN cursor3;**

**REPEAT**

**FETCH cursor3 INTO a,b;**

**IF VAR=0 THEN**

**select a,b;**

**endif;**

**UNTIL VAR=1**

**END REPEAT;**

**CLOSE cursor3;**

**END**

La estructura de los cursores es siempre igual. Lo único que tenemos que cambiar es la sentencia SELECT que va dentro de la definición del cursor (DECLARE CURSOR). Esa sentencia SELECT tendrá que realizar la operación que nos indique el enunciado que queremos hacer. Además, tendremos que definir tantas variables como campos aparezcan en el SELECT y, por supuesto, del mismo tipo. La sentencia SELECT podrá ser sencilla, como esta que hemos visto, o más complicada, puesto que puede contener cualquiera de los comandos e instrucciones que hemos visto a lo largo de la segunda evaluación.

**EJERCICIOS PARA PRACTICAR**

1. Crea un procedimiento en el que usarás un cursor que visualice el nombre y la localidad de todos los departamentos del esquema empresa.
2. Crea un cursor, en el esquema empresa, para visualizar el apellido, oficio y comisión de aquellos empleados cuya comisión supere un valor que introduzcamos como parámetro.