BD UD6 Cursores

# Introducción

Un cursor es un recurso de programación que nos ofrecen casi todos los gestores de bases de datos relacional y consiste en la posibilidad de recorrer **fila a fila** un conjunto de resultados provenientes de una sentencia SELECT.

La ventaja que tiene el uso de cursores es que vamos a poder guardar en variables locales cada fila de resultados y manejar esa información como queramos (para realizar otras consultas, operaciones de INSERT, UPDATE o DELETE).

Recordad cuando vimos procedimientos y funciones que en algunos ejercicios teníamos que limitar el resultado del SELECT a una única fila, ya que el INTO no podría hacerse. Con un cursor, la consulta va a poder devolver muchas filas, ya que nos permite obtener los datos de cada fila e ir pasando de una fila a la siguiente.

Sólo se debe hacer uso de este recurso si no es posible obtener el mismo resultado con una única operación SQL.

En nuestra base de datos de ejemplo, supongamos que queremos comprobar si las ganancias totales de cada atracción coinciden con la suma de las ganancias de los días en los que se celebró.

Esta operación la podríamos hacer

* Empleando un cursor, de tal forma que iríamos recorriendo fila a fila cada una de las atracciones, y por cada atracción obtendríamos su nombre y ganancia total, que guardaríamos en variables locales, haciendo una consulta de la suma de las ganancias por cada día de la atracción guardada en la variable, comprobando si la suma coincide con la ganancia total.
* Realizar una consulta SQL de la forma:

1 **SELECT** \*

2 **FROM** ATRACCIONES

3 **WHERE** ganancias <> (**SELECT** **SUM**(ganancias)

4 **FROM** ATRACCION\_DIA

5 **WHERE** ATRACCION\_DIA.nombre\_atraccion = ATRACCIONES.nombre)

Siempre es mejor la segunda opción ya que es mucho más rápida. Con el cursor tenemos que ir procesando fila a fila.

Pero ahora supongamos que queremos obtener una cadena con el formato: atraccion1:gananciatotal:gananciasumada, atraccion2:gananciatotal:gananciasumada

Con las atracciones que no cumplen que la suma sea igual...

En este caso ya no podremos hacer uso de la sentencia SELECT anterior y tendremos que hacer uso de un CURSOR.

Un cursor tiene las siguientes características:

* **Son de sólo lectura:** Sólo sirven para leer datos. Es decir, sólo podremos declarar un cursor para leer los datos que provienen de una consulta SELECT y nunca podremos modificar los datos de la tabla a través del cursor.
* **De acceso secuencial:** La información que va a procesar el cursor (el resultado de un SELECT) es secuencial. Vamos a recorrer fila a fila desde la primera a la última de forma secuencial, una detrás de otra y no podremos saltar a una fila cualquiera de forma directa, tendremos que pasar por todas las anteriores.
* Puede crearse dentro de un procedimiento, función o trigger.

# Manejo de cursores

Necesitamos realizar las siguientes operaciones:

* [DECLARACIÓN DEL CURSOR](https://dev.mysql.com/doc/refman/8.0/en/declare-cursor.html): Donde definimos el nombre del cursor y la consulta SQL SELECT que va a devolver el conjunto de resultados.
* [APERTURA DEL CURSOR](https://dev.mysql.com/doc/refman/8.0/en/open.html): Para poder procesar la información que viene en el SELECT.
* [LECTURA DEL CURSOR](https://dev.mysql.com/doc/refman/8.0/en/fetch.html): Esto se realiza con la orden FETCH que permite guardar la información de una fila en variables locales y pasa a la siguiente fila del conjunto de resultados. La lectura va en un bucle que se ejecuta hasta que se terminan de leer todas las filas.
* [CIERRE DEL CURSOR](https://dev.mysql.com/doc/refman/8.0/en/close.html): Para liberar de la memoria el cursor creado.

Veamos en que sentencias SQL se traduce cada uno de los pasos anteriores:

* **DECLARACIÓN DEL CURSOR**: Donde definimos el nombre del cursor y la consulta SQL SELECT que va a devolver el conjunto de resultados.

1 **DECLARE** nome\_cursor **CURSOR** **FOR**

2 sentencia\_SELECT;

Notas:

* Indicar que para salir del bucle vamos a 'capturar' una excepción de 'no hay más datos'. La definición de dicha captura tienen que ir después de DECLARE.
* La declaración del cursor tiene que ir después de la declaración de las variables y [condiciones](https://dev.mysql.com/doc/refman/8.0/en/declare-condition.html).
* En la sentencia SELECT seleccionaremos las columnas que queremos guardar 'por cada fila' al recorrer el cursor.
* Dentro de un procedimiento, trigger o función podemos tener varios cursores, pero todos ellos deben de tener un nombre diferente.

Por ejemplo, vamos a declarar un cursor en el que seleccionaremos los artistas que no tengan jefe:

1 **DECLARE** v\_noHayMasDatos INT **DEFAULT** **FALSE**;

2

3 **DECLARE** c\_artistas **CURSOR** **FOR**

4 **SELECT** apellidos,nombre **FROM** ARTISTAS **WHERE** nif\_jefe **IS** **NULL**;

5

6 **DECLARE** **CONTINUE** **HANDLER** **FOR** **NOT** **FOUND** **SET** v\_noHayMasDatos = **TRUE**;

Notas:

* **Linea 1:** Declaramos una variable de tipo INT que vamos a utilizar para saber cuando acabamos de recorrer las filas del cursor. Recordad que la definición de variables tiene que ir antes de la definición del cursor.
* **Linea 3:** Declaramos el cursor.
* **Linea 6:** Capturamos la excepción que se produce cuando el cursor no encuentra más datos. Lo que indica dicha línea es, "cuando se produce la excepción, asigna a la variable v\_noHayMasDatos el valor TRUE.
* **APERTURA DEL CURSOR**: Para poder procesar la información que viene en el SELECT.

**OPEN** nombre\_cursor;

Siguiendo nuestro ejemplo:

**DECLARE** v\_noHayMasDatos INT **DEFAULT** **FALSE**;

**DECLARE** c\_artistas **CURSOR** **FOR**

**SELECT** apellidos,nombre **FROM** ARTISTAS **WHERE** nif\_jefe **IS** **NULL**;

**DECLARE** **CONTINUE** **HANDLER** **FOR** **NOT** **FOUND** **SET** v\_noHayMasDatos = **TRUE**;

**OPEN** c\_artistas;

* **LECTURA DEL CURSOR**: Esto se realiza con la orden FETCH que permite guardar la información de una fila en variables locales y pasa a la siguiente fila del conjunto de resultados. La lectura va en un bucle que se ejecuta hasta que se terminan de leer todas las filas.

**FETCH** nombre\_cursor **INTO** variable1[,variable2,...];

* En la parte INTO deben ir tantas variables como columnas tengamos en la sentencia SELECT.
* Cada variable guardará el dato de la columna que venga en el SELECT, de tal forma que la primera variable guardará el dato de la primera columna, la segunda variable, el dato de la segunda columna y así sucesivamente.
* Cada variable debe estar definida con el mismo tipo de dato que la columna a la que está asociada.

Esta es la parte que va a necesitar de un bucle, en el que vamos 'fila a fila' recorriendo el resultado de ejecutar la sentencia SELECT que define el cursor.

El paso de una fila a la siguiente se realiza con la sentencia FETCH. Veamos en que consiste:

**Recorriendo el cursor**[Tabla

Descripción generada automáticamente con confianza media](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_1.jpg) Después del OPEN, el cursor se sitúa justo antes de la primera fila de resultados del SELECT[Diagrama, Gráfico de cajas y bigotes

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_2.jpg)[Gráfico, Diagrama, Gráfico de cajas y bigotes

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_3.jpg)

Al realizar el siguiente FETCH podemos recuperar los datos de la siguiente fila.[Diagrama

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_4.jpg)[Diagrama, Gráfico de cajas y bigotes

Descripción generada automáticamente con confianza media](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_5.jpg)

[Gráfico, Diagrama

Descripción generada automáticamente con confianza media](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_6.jpg)

[Tabla

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_7.jpg)

Al realizar el último FETCH se lanza la excepción que vamos a capturar con la orden DECLARE HANDLER.

**Nota:** En SQLServer los conceptos son los mismos que estamos viendo. La diferencia es la forma en cómo SQLServer testea el fin de datos en el cursor, ya que chequea la [función @@FETCH\_STATUS](https://docs.microsoft.com/ca-es/sql/t-sql/functions/fetch-status-transact-sql?view=sql-server-2017) como condición para salir del bucle.

Siguiendo nuestro ejemplo:

1 **DECLARE** v\_noHayMasDatos INT **DEFAULT** **FALSE**;

2 **DECLARE** v\_apellidos varchar(100);

3 **DECLARE** v\_nombre varchar(45);

4

5 **DECLARE** c\_artistas **CURSOR** **FOR**

6 **SELECT** apellidos,nombre **FROM** ARTISTAS **WHERE** nif\_jefe **IS** **NULL**;

7

8 **DECLARE** **CONTINUE** **HANDLER** **FOR** **NOT** **FOUND** **SET** v\_noHayMasDatos = **TRUE**;

9

10 **OPEN** c\_artistas;

11

12 read\_loop: LOOP

13 **FETCH** c\_artistas **INTO** v\_apellidos,v\_nombre

14

15 **IF** v\_noHayMasDatos **THEN**

16 LEAVE read\_loop;

17 **END** **IF**;

18

19 *-- Por cada fila disponemos del apellido y nombre. Ahora podemos utilizar estos datos para lo que queramos*

20 **END** LOOP;

**Líneas 2,3:** Definimos las variables que van a guardar los datos de cada fila. Los tipos de datos se corresponden con los tipos definidos en la tabla ARTISTAS para las columnas apellidos y nombre.

**Líneas 12-20:** Bucle con la orden FETCH y la condición de salida del bucle al terminar de recorrer las filas.

**NOTA IMPORTANTE:** Al estar dentro de un bucle debemos de asegurarnos que la condición de salida siempre se alcanza ya que si no estaríamos en un bucle infinito.

Debemos asegurarnos que el FETCH está dentro del bucle ya que es la orden que va a provocar la excepción y va a permitir salir del bucle.

Ahora dentro del bucle podemos hacer lo que queramos con los datos que se encuentran en las variables v\_apellidos y v\_nombre, por ejemplo

* CIERRE DEL CURSOR: Para liberar de la memoria el cursor creado.

1 **CLOSE** nombre\_cursor;

Siguiendo nuestro ejemplo:

1 **DECLARE** v\_noHayMasDatos INT **DEFAULT** **FALSE**;

2 **DECLARE** v\_apellidos varchar(100);

3 **DECLARE** v\_nombre varchar(45);

4

5 **DECLARE** c\_artistas **CURSOR** **FOR**

6 **SELECT** apellidos,nombre **FROM** ARTISTAS **WHERE** nif\_jefe **IS** **NULL**;

7

8 **DECLARE** **CONTINUE** **HANDLER** **FOR** **NOT** **FOUND** **SET** v\_noHayMasDatos = **TRUE**;

9

10 **OPEN** c\_artistas;

11

12 read\_loop: LOOP

13 **FETCH** c\_artistas **INTO** v\_apellidos,v\_nombre

14

15 **IF** v\_noHayMasDatos **THEN**

16 LEAVE read\_loop;

17 **END** **IF**;

18

19 *-- Por cada fila disponemos del apellido y nombre. Ahora podemos utilizar estos datos para lo que queramos*

20 **END** LOOP;

21

22 **CLOSE** c\_artistas;

No es obligatorio el uso de LOOP y podremos hacer uso de REPEAT o WHILE, pero teniendo en cuenta que al realizar el último FETCH capturamos la excepción y debemos de adaptar el comportamiento a cada tipo de bucle.

El proceso para implementar un cursor siempre es el mismo [Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Cursor_8.jpg)

**Ejercicios propuestos Cursores**

**Ejercicio 1** Crea un procedimiento de nombre atracciones\_checkGanancias en el que queremos comprobar si las ganancias totales de cada atracción coinciden con la suma de las ganancias de los días en los que se celebró. El procedimiento debe devolver una cadena con el formato: atraccion1:gananciatotal:gananciasumada, atraccion2:gananciatotal:gananciasumada con las atracciones que no cumplen que la suma sea igual...

**Ejercicio 2** Crea un procedimiento de nombre artistas\_addSuplementoPorCuidados, que compruebe a cuantos animales cuida cada uno de los artistas. Aquellos artistas que cuidan más de un número de animales indicados por un parámetro se les dará un plus a su nómina igual al número de animales que cuida multiplicado por 100 euros. Muestra el nombre y complemento de cada artista así como la suma de todos los complementos.

El resultado debe aparecer como una única consulta (no valen varios SELECT).

Para ello haz uso de una tabla temporal que vaya guardando los datos (el nombre completo y el suplemento de la ganancia) y posteriormente haz un SELECT de dicha tabla. Para crear una tabla temporal haz uso [de la sentencia CREATE TEMPORARY TABLE](https://dev.mysql.com/doc/refman/8.0/en/create-temporary-table.html). Dicha orden debe ir después de la orden DECLARE CONTINUE HANDLER del cursor.

Recuerda borrar la tabla temporal al salir del procedimiento con la orden DROP TEMPORARY TABLE.