Cursores en SQL

En procedimientos SQL, un cursor permite definir un conjunto de resultados (un conjunto de filas de datos) y realizar una lógica compleja fila por fila.

En el siguiente código se presenta la creación de las tablas para la base de datos llamada “Supermercados_cursores” donde creara las tablas Clientes, Productos y Ventas.

```sql
CREATE DATABASE Supermercado_cursores;

USE Supermercado_cursores;

CREATE TABLE Clientes (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(50) NOT NULL,
  email VARCHAR(50) NOT NULL UNIQUE,
  telefono VARCHAR(15),
  direccion VARCHAR(100)
);

CREATE TABLE Productos (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(50) NOT NULL,
  descripcion VARCHAR(255),
  precio DECIMAL(10,2) NOT NULL,
  cantidad INT NOT NULL
);

CREATE TABLE Ventas (
  id INT PRIMARY KEY AUTO_INCREMENT,
  cliente_id INT NOT NULL,
  producto_id INT NOT NULL,
  fecha DATE NOT NULL,
  cantidad INT NOT NULL,
  FOREIGN KEY (cliente_id) REFERENCES Clientes(id),
  FOREIGN KEY (producto_id) REFERENCES Productos(id)
);
```

Luego encontramos el codigo donde insertamos datos a las tablas ya creadas anteriorment con sus respectivos tipos de datos.

```
31 •  insert into Clientes
32    values (18204322,"miguel","miguel@gmail.com","305573235","carrera 7 #21-10");
33 •  insert into Clientes
34    values (1100214322,"angel","algel@gmail.com","300573235","carrera 8 #22-10");
35
36 •  insert into Productos values (1,"arroz","arroz diana 1 lb",2500,80);
37 •  insert into Productos values (2,"frijol","frijol cargamanto 1 lb",3000,90);
38 •  insert into Productos values (3,"platano","un gajo de platano 1 lb",1800,80);
39 •  insert into Productos values (4,"carne","carne de rez 1 lb",5000,30);
40 •  INSERT INTO Ventas values (1,18204322,1,now(),2);
```

Al finalizar podemos hacer correr el código y nos mostrara la base de datos ya creada con sus respectivas tablas y sus datos insertados en cada tabla.

Luego procedemos a crea un cursor que se pueda adecuar a esta base de datos la cual en esta ocasión se creara un cursor que muestre el id y el nombre del producto y adicionalmente mostrara una columna donde calcule el precio total que hay multiplicando las cantidades de los productos con el precio.

Primero procedemos delimitando el código de nuestro cursor

```
1    Delimiter //
2    //
```

Procedemos a indicar la creación del cursor con la sentencia **create procedure** y le asignamos un nombre a nuestro cursor seguido de los paréntesis, en este case caso el nombre es “prcioTotalProductos()” seguido de la palabra **begin** que nos indicara que empezara a correr el cursor

```
1    Delimiter //
2    create procedure prcioTotalProductos()
3    begin
4    //
```

Agregamos variables que mostraremos en el cursor y le asignamos un nombre y el tipo de dato que será:

```
1    Delimiter //
2    create procedure prcioTotalProductos()
3    begin
4    declare id_producto int;
5    DECLARE nombre_producto varchar(25);
6    declare precio_total int;
7    //
8
```

Declaramos el cursor y le asignamos un nombre y luego designamos la función que realizara el cursor que en este caso como se mencionaba anteriormente, mostrara el id y el nombre del producto junto con otra columna donde muestre el precio total de la multiplicación de la cantidad de productos con el precio de los productos, y todos estos datos los sacaremos de la tabla Productos.

```
Delimiter //
create procedure prcioTotalProductos()
begin
declare id_producto int;
DECLARE nombre_producto varchar(25);
declare precio_total int;

DECLARE cursor_1 cursor for
select id,nombre,precio*cantidad
from productos;
//
```

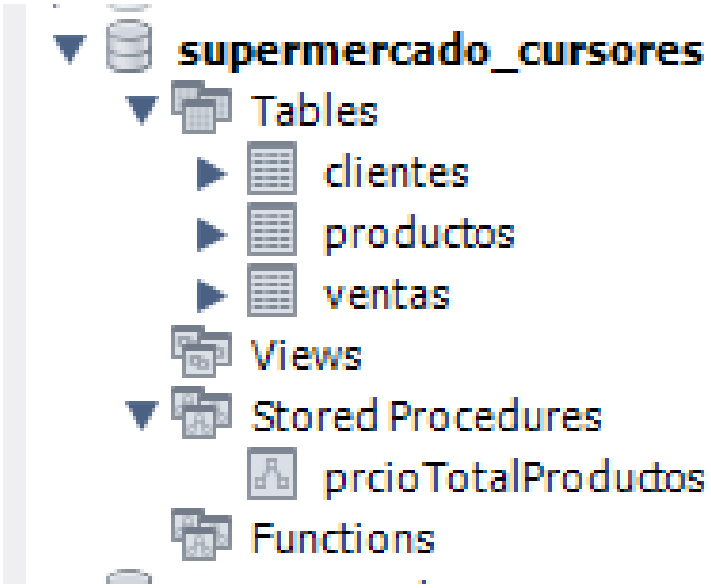
Luego creamos el bucle llamando al cursor que declaramos anteriormente junto con las variables de las columnas que igualmente creamos y las imprimimos con la sentencia Select y finalizamos cerrando el ciclo con la sentencia end loop

```
Delimiter //
create procedure prcioTotalProductos()
begin
declare id_producto int;
DECLARE nombre_producto varchar(25);
declare precio_total int;

DECLARE cursor_1 cursor for
select id,nombre,precio*cantidad
from productos;

open cursor_1;
READ_LOOP: LOOP
fetch cursor_1 INTO id_producto , nombre_producto, precio_total ;
SELECT  id_producto , nombre_producto,  precio_total;
end loop ;
//
```

Y para finalizar el cursor cerramos el cursor anteriormente declarado junto con toda la estructura del código total del cursor.

```
1   Delimiter //
2   create procedure prcioTotalProductos()
3   begin
4   declare id_producto int;
5   DECLARE nombre_producto varchar(25);
6   declare precio_total int;
7
8   DECLARE cursor_1 cursor for
9   select id,nombre,precio*cantidad
10  from productos;
11
12  open cursor_1;
13  READ_LOOP: LOOP
14  fetch cursor_1 INTO id_producto , nombre_producto, precio_total ;
15  SELECT  id_producto , nombre_producto,  precio_total;
16  end loop ;
17
18  close cursor_1;
19  end //
```

Al hacer correr el código quedara guardado en nuestra base de datos



Y para llamar nuestro cursor lo hacemos con la sentencia call junto con el nombre que le asignamos a nustro cursor anteriormente creado

```
1 •  call prcioTotalProductos() ;
```

La base de datos creada con sus tablas, datos y el cursor lo podremos encontrar en el siguiente link de GitHub:

https://github.com/Brayandev8/ProgramacionEnBaseDeDatos.git