Cursors

database gestion

1. Create an SP that shows all clients showing their code and name and their representative (only the code), using a cursor.

1. The first line will be dedicated to connect to the desired database, in this case, gestion:

   USE gestion;

2. Next, we are going to create a stored procedure that lists all clients with their names and their representative in our database. For that reason, we write:

```
DELIMITER $$
DROP PROCEDURE IF EXISTS listar_clientes_cursor$$
CREATE PROCEDURE listar_clientes_cursor()
BEGIN
        DECLARE final INTEGER DEFAULT 0;
        DECLARE nc INTEGER;
        DECLARE nr INTEGER;
        DECLARE nom VARCHAR(20);
        DECLARE cur CURSOR FOR SELECT numclie, nombre, repclie FROM clientes;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET final = 1;
        OPEN cur;
        bucle:LOOP
        FETCH cur INTO nc, nom, nr;
        IF final = true THEN
                LEAVE bucle;
        END IF;
        SELECT nc AS num_cliente, nom AS nombre_cliente, nr AS num_representante;
        END LOOP;
    CLOSE cur;
END
$$
DELIMITER ;
```

   You just need to take into account that a delimiter must be set. Then, we add a line in order to delete our SP (that´s important because if we want to change something, we can execute the script again without any problems). Then, we write the SP header with the sentence: CREATE PROCEDURE, its name and its parameters (here we don´t have any). Afterwards, we find between the sentences BEGIN and END, the code of the SP. In this case, we have a

cursor that lists all the information about the clients and their representative. In order to do that, we need to declare some variables. We open the cursor and we say that, as long as we find clients, we want to list them. Finally, we close the cursor and the SP with the delimiter $ $ and we reset the default delimiter.

3. In order to execute it, we must write:

```
USE gestion;
CALL listar_clientes_cursor();
```

2. Now, show the name of the representative too.

```
USE gestion;
DELIMITER $$
DROP PROCEDURE IF EXISTS listar_clientes_representante_cursor$$
CREATE PROCEDURE listar_clientes_representante_cursor()
BEGIN
   DECLARE done INT DEFAULT FALSE;
   DECLARE client_name VARCHAR(50);
   DECLARE representative_name VARCHAR(50);

   -- Declare cursor for selecting client names and their representatives
   DECLARE client_cursor CURSOR FOR
      SELECT c.nombre AS client_name, e.nombre AS representative_name
      FROM clientes c
      INNER JOIN empleados e ON c.repclie = e.numemp;

   -- Declare continue handler to exit loop when no more rows found
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

   -- Open cursor
   OPEN client_cursor;

   -- Start fetching rows
   read_loop: LOOP
      FETCH client_cursor INTO client_name, representative_name;

      -- Exit loop if no more rows
      IF done THEN
         LEAVE read_loop;
      END IF;

      -- Display client and representative information
      SELECT CONCAT('Client: ', client_name, ', Representative: ', representative_name) AS
Client_Representative_Info;
   END LOOP;

   -- Close cursor
   CLOSE client_cursor;
END$$
DELIMITER ;
```

```
USE gestion;
CALL listar_clientes_cursor();
```

3. Create a simple stored procedure that uses cursors in order to retrieve and display information about employees from the empleados table. This stored procedure fetches each employee's ID, name, age, office, title, contract date, manager, quota, and sales. Then, it prints the information for each employee.

```
USE gestion;
DELIMITER $$
DROP PROCEDURE IF EXISTS info_empleados_cursor$$
CREATE PROCEDURE info_empleados_cursor()
BEGIN
   -- Declare variables to store employee data
   DECLARE emp_id INT;
   DECLARE emp_name VARCHAR(20);
   DECLARE emp_age INT;
   DECLARE emp_oficina INT;
   DECLARE emp_title VARCHAR(20);
   DECLARE emp_contract_date DATETIME;
   DECLARE emp_jefe INT;
   DECLARE emp_cuota DECIMAL(19,4);
   DECLARE emp_ventas DECIMAL(19,4);
   DECLARE done INT DEFAULT 0;

   -- Declare cursor to iterate through employee records
   DECLARE emp_cursor CURSOR FOR
      SELECT numemp, nombre, edad, oficina, titulo, contrato, jefe, cuota, ventas
      FROM empleados;

   -- Declare handler for cursor
   DECLARE CONTINUE HANDLER FOR NOT FOUND
      SET done = 1;

   -- Open cursor
   OPEN emp_cursor;

   -- Fetch the first row
   FETCH emp_cursor INTO emp_id, emp_name, emp_age, emp_oficina, emp_title,
emp_contract_date, emp_jefe, emp_cuota, emp_ventas;

   -- Start printing header
   SELECT 'Employee ID', 'Name', 'Age', 'Office', 'Title', 'Contract Date', 'Manager', 'Quota', 'Sales';

   -- Loop through the cursor and print employee information
   WHILE done <> 1 DO
      SELECT emp_id, emp_name, emp_age, emp_oficina, emp_title, emp_contract_date,
emp_jefe, emp_cuota, emp_ventas;
```

```
        FETCH emp_cursor INTO emp_id, emp_name, emp_age, emp_oficina, emp_title,
emp_contract_date, emp_jefe, emp_cuota, emp_ventas;
    END WHILE;

    -- Close cursor
    CLOSE emp_cursor;
END$$

DELIMITER ;


USE gestion;
CALL info_empleados_cursor();
```

4. Create an SP that, using a cursor, shows all employees that work on a certain office. Employees with a NULL value on the office field will not be displayed.

```
USE gestion;
DELIMITER $$
DROP PROCEDURE IF EXISTS listar_empl_oficina_cursor$$
CREATE PROCEDURE listar_empl_oficina_cursor()
BEGIN
	DECLARE final INTEGER DEFAULT 0;
	DECLARE ofi INTEGER;
	DECLARE gofi INTEGER; -- para controlar cuando cambia de oficina
	DECLARE num INTEGER;
	DECLARE ciu VARCHAR(20);
	DECLARE nom VARCHAR(20);
	DECLARE cur CURSOR FOR
		SELECT O.oficina, O.ciudad, E.numemp, E.nombre
		FROM Oficinas O INNER JOIN Empleados E ON O.oficina=E.oficina
		ORDER BY O.oficina, E.numemp;

	DECLARE CONTINUE HANDLER FOR NOT FOUND SET final = 1;
	OPEN cur;
	FETCH cur INTO ofi, ciu, num, nom;
	SET gofi = ofi; -- nos guardamos la oficina

	bucle: WHILE final = 0 DO
		SELECT ofi AS oficina, ciu AS ciudad;

		WHILE gofi = ofi DO
			SELECT num AS numemp, nom AS nomemp; -- visualizamos el empleado
			FETCH NEXT FROM cur INTO ofi, ciu, num, nom; -- leemos el siguiente
			IF final = 1 THEN
				LEAVE bucle;
			END IF;
		END WHILE;

		SET gofi = ofi;

	   END WHILE bucle;

     CLOSE cur;
END
$$
DELIMITER ;
```

```
USE gestion;

CALL listar_empl_oficina_cursor();
```

5. We want to obtain from all employees in office 12 their number (numemp), name (of the employee) and how many orders he/she has placed, but using the stored procedure pedidos_rep that we created in a previous exercise.

```
USE gestion;
DELIMITER $$
DROP PROCEDURE IF EXISTS listar_empl_oficina_cursor_12$$
CREATE PROCEDURE listar_empl_oficina_cursor_12()
BEGIN
	DECLARE final INTEGER DEFAULT 0;
	DECLARE num INTEGER;
	DECLARE cur CURSOR FOR SELECT numemp FROM empleados WHERE oficina = 12;
	DECLARE CONTINUE HANDLER FOR NOT FOUND SET final = 1;

	OPEN cur;
	FETCH cur INTO num; -- Leemos la primera fila (el primer numemp)
	WHILE final = 0 DO
		CALL pedidos_rep(num);
		 FETCH cur INTO num; -- leemos el siguiente
	END WHILE;
	CLOSE cur;
END
$$
DELIMITER ;
```

Note that the procedure returns two rows with data and two rows without data, corresponding to employees who have no orders.

```
USE gestion;
CALL listar_empl_oficina_cursor_12();
```

6. Create a simple stored procedure using cursors that retrieves the total sales (ventas) for each employee (numemp) from the empleados table and prints out the results.

```
USE gestion;
DELIMITER //
DROP PROCEDURE IF EXISTS ventastotales//
CREATE PROCEDURE ventastotales()
BEGIN
   DECLARE done INT DEFAULT FALSE;
   DECLARE emp_id INT;
   DECLARE emp_name VARCHAR(20);
   DECLARE total_sales DECIMAL(19, 4);

   -- Declare cursor for selecting employee IDs and names
   DECLARE cur_emp CURSOR FOR SELECT numemp, nombre FROM empleados;

   -- Declare continue handler to exit loop
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

   OPEN cur_emp; -- Open the cursor

   -- Loop through the cursor
   read_loop: LOOP
      -- Fetch data from cursor into variables
      FETCH cur_emp INTO emp_id, emp_name;

      -- Check if there are no more rows to fetch
      IF done THEN
         LEAVE read_loop;
      END IF;

      -- Calculate total sales for the employee
      SELECT SUM(importe) INTO total_sales FROM pedidos WHERE rep = emp_id;

      -- Print employee's name and total sales
      SELECT CONCAT('Employee: ', emp_name, ', Total Sales: ', total_sales) AS Result;
   END LOOP;

   CLOSE cur_emp; -- Close the cursor
END //


DELIMITER ;
```

```
USE gestion;
CALL ventastotales();
```

7. Create a stored procedure using cursors that retrieves the names of all employees working in a specified office.

```
USE gestion;
DELIMITER //

CREATE PROCEDURE GetEmployeesInOffice(IN officeID INT)
BEGIN
   DECLARE empID INT;
   DECLARE empName VARCHAR(20);
   DECLARE done INT DEFAULT FALSE;

   -- Declare cursor to iterate through employees in the specified office
   DECLARE empCursor CURSOR FOR
      SELECT numemp, nombre
      FROM empleados
      WHERE oficina = officeID;

   -- Declare handler for cursor
   DECLARE CONTINUE HANDLER FOR NOT FOUND
      SET done = TRUE;

   -- Open the cursor
   OPEN empCursor;

   -- Initialize done variable
   SET done = FALSE;

   -- Start fetching rows from the cursor
   emp_loop: LOOP
      -- Fetch employee ID and name from cursor into variables
      FETCH empCursor INTO empID, empName;

      -- Check if no more rows to fetch
      IF done THEN
         LEAVE emp_loop;
      END IF;

      -- Output employee ID and name
      SELECT empID, empName;
   END LOOP emp_loop;

   -- Close the cursor
   CLOSE empCursor;
END//
DELIMITER ;

USE gestion;
CALL GetEmployeesInOffice(12);
```