STORED PROCEDURES AND CURSORS

LAB-2

What is a Stored Procedure?

- A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.
- So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.
- You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.
- SQL server creates an execution plan, and stores it in the cache. It improves the performance and execution speed.

In this session, We use mySQL Workbench to practice the scripts

Stored Procedures Syntax

```
DELIMITER $$
CREATE PROCEDURE procedure_name(parameters)
BEGIN
      SQL statements;
END $$
DELIMITER ;
```

To execute the stored procedure –

```
CALL procedure_name;
```

To delete a stored procedure –

```
DROP PROCEDURE procedure_name;
```

The most important part is parameters. Parameters are used to pass values to the Procedure. There are 3 different types of parameters, they are as follows:

1. **IN:**
   This is the Default Parameter for the procedure. It always receives the values from calling program.
2. **OUT:**
   This parameter always sends the values to the calling program.
3. **INOUT:**
   This parameter performs both the operations. It Receives value from as well as sends the values to the calling program.

Example

```
DELIMITER //

CREATE PROCEDURE SelectAllCustomers()
BEGIN
    SELECT * FROM Customers;
END //

DELIMITER ;

-- Call the procedure
CALL SelectAllCustomers();
```

Parameterised Stored Procedures:

- Stored Procedures With Multiple Parameters

```
DELIMITER //

CREATE PROCEDURE SelectAllCustomers (
    IN input VARCHAR(30),
    IN code INT
)
BEGIN
    SELECT *
    FROM Customers
    WHERE City = input
      AND PostalCode = code;
END //

DELIMITER ;

-- Example call
CALL SelectAllCustomers('London', 12345);
```

Nested stored procedures

- Whenever you write a statement that calls a procedure X in the body of another stored procedure Y, you’re nesting X in Y.

- Nested stored procedures help to break up large amounts of SQL statements into smaller reusable pieces, hence enhancing modularity.

Example

```
DELIMITER //

CREATE PROCEDURE say_hello(IN name VARCHAR(50))
BEGIN
    SELECT CONCAT('Hello, ', name, '!') AS greeting;
END //

CREATE PROCEDURE greet_team()
BEGIN
    CALL say_hello('Alice');
    CALL say_hello('Bob');
END //

DELIMITER ;

CALL greet_team();
```

What is Cursor in SQL ?

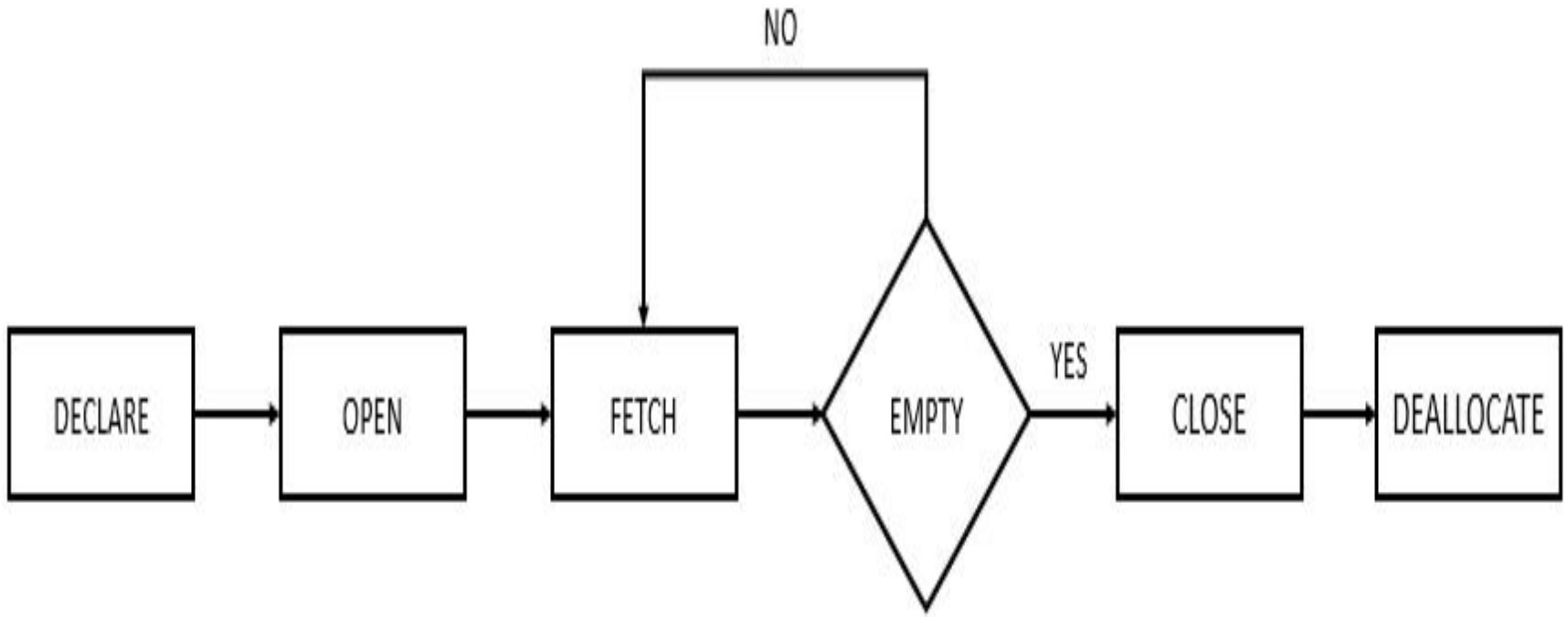
Cursor is a Temporary Memory or Temporary Work Station. It is Allocated by Database Server at the Time of Performing DML(Data Manipulation Language) operations on Table by User. Cursors are used to store Database Tables. There are 2 types of Cursors: Implicit Cursors, and Explicit Cursors. These are explained as following below.

1. **Implicit Cursors:**
   Implicit Cursors are also known as Default Cursors of SQL SERVER. These Cursors are allocated by SQL SERVER when the user performs DML operations.
2. **Explicit Cursors :**
   Explicit Cursors are Created by Users whenever the user requires them. Explicit Cursors are used for Fetching data from Table in Row-By-Row Manner.

Should we use cursors?

- Relational Databases are very good at handling data in sets.
- But, if we need to process data on row-by-row basis, then cursors are useful.
- Cursors can easily be replaced by Joins

Life cycle of a cursor:



Syntax

- DECLARE variables; records;
- create a cursor;
- BEGIN OPEN cursor;
- FETCH cursor;
- process the records;
- CLOSE cursor;
- END;

Example

```
DECLARE product_name varchar(30), list_price decimal;

DECLARE cursor_product CURSOR FOR
    SELECT product_name, list_price FROM production.products;

Open cursor_product;
Start LOOP
        FETCH cursor_product into product_name, list_price;
        IF NOT FOUND, LEAVE LOOP;
        END IF;
END LOOP

CLOSE cursor_product;
DEALLOCATE cursor_product;
```

References

- https://www.geeksforgeeks.org/sql-difference-between-functions-and-stored-procedures-in-pl-sql/
- https://docs.oracle.com/cd/A97630_01/appdev.920/a96624/01_oview.htm#740
- https://www.mysqltutorial.org/stored-procedures-parameters.aspx
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/declare-cursor-transact-sql?view=sql-server-ver16
- https://coderpad.io/blog/development/advanced-stored-procedures-in-mysql/