THE UNIVERSITY OF THE WEST INDIES

Department of Computing

COMP6720 – Advanced Database Management Systems 2021

Lecturer: Dr. Kevin Miller

Problem Definition

NB: For this assignment, you will have the freedom to choose any programming language for your development.

The design and construction of database systems can be an extremely difficult task that takes many things into consideration. This includes thinking about how tables and records gets translated into units of data stored on the actual file system. Important decisions have to made at this level as it relates to file type and organization. Then there is the consideration of performance which includes indexing and hashing and the appropriate structures that should be used to improve searching. In this assignment, you will design and implement a very basic, but your own database system. There will be many assumptions in building this system and only a limited number of features will be developed. In your design, you will build classes to represent a block, record and File. Your system must be able to handle a minimum of 15 records. The blocking factor (bfr) is 5 records per block. One way to think of this, is to use a class to represent a block that contains and array of lists. The size of the array should be 5. The list should have three (3) fields to store and integer and two (2) strings. This is shown in Figure 1.

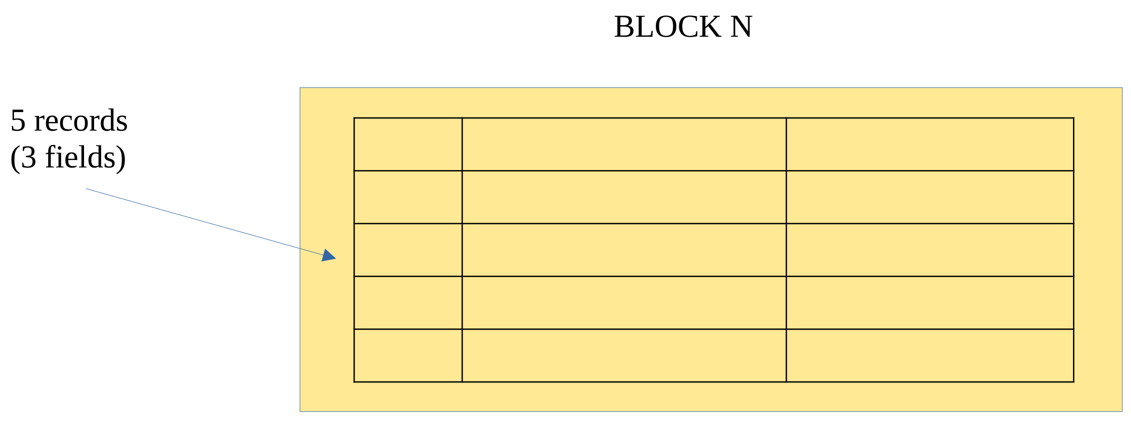

Figure 1: Representation for a block with records

Your implementation will be a client server architecture where the database server will run a continuous process waiting for clients to connect. Your server should be able to accept at least two connections simultaneously. This can be accomplished by using threads to implement the server. This also means that basic transaction management and concurrency control should be observed. At least only one transaction should be able to write/update a shared resource even though both transactions/queries can be accessing the resource at the same time.

This server will take as a command line argument the folder where the databases are supposed to be created. In this folder, sub folders that represent each database will be created and binary files will represent the tables in the sub folders. The client will send SQL commands to the database server to be processed. You are building the DBMS and also a basic database engine.

The following simple queries should be supported:

```
CREATE <database>;
USE <database>;
```

```
CREATE <table name>

(<coldecl> [,<coldecl>*]);

<coldecl> := <col><type>

<type> := int | String
```

eg.
```
eg. CREATE TABLE Employee (
      id int,
      fname String,
      lname String
);
```

```
CREATE INDEX <index> ON <table> (<colname>);
```

```
eg. CREATE INDEX ON Employee(eid)
```

```
SELECT <items>FROM <table> [WHERE <pred>];

<pred> := WHERE <colname> <op> <somevalue>

<op>:= < | > | = | !=
```

```
eg. SELECT eid FROM Employee WHERE eid > 2;
```

```
INSERT INTO <table> VALUES (<items>);
```

eg. INSERT INTO Employee VALUES (1, ‘Pam’,‘Jones’);

```
DELETE FROM <table>WHERE <colname> <op> <somevalue>;
```

eg. DELETE FROM Employee WHERE eid=1;

```
UPDATE <table> SET <attr>=<somevalue> WHERE <colname> <op> <somevalue>;
```

eg. UPDATE Employee SET fname= ‘Pam’ WHERE eid=2;

The interface should take the form:

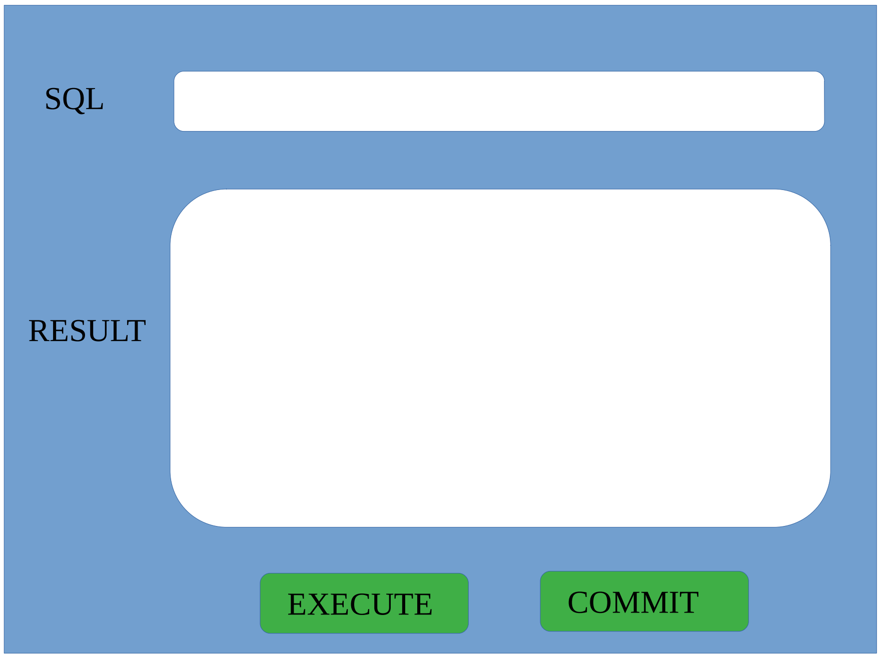

Figure 2: Interface for DBMS

In Figure 2, the button for execute will run the command in the SQL textfield and display the result in the RESULT textarea. The commit button should be used to write all the database content in the current session to permanent storage. That means, if you run an SQL statement that make some form of update to the database objects in memory, then clicking on the commit button should make that change permanent.

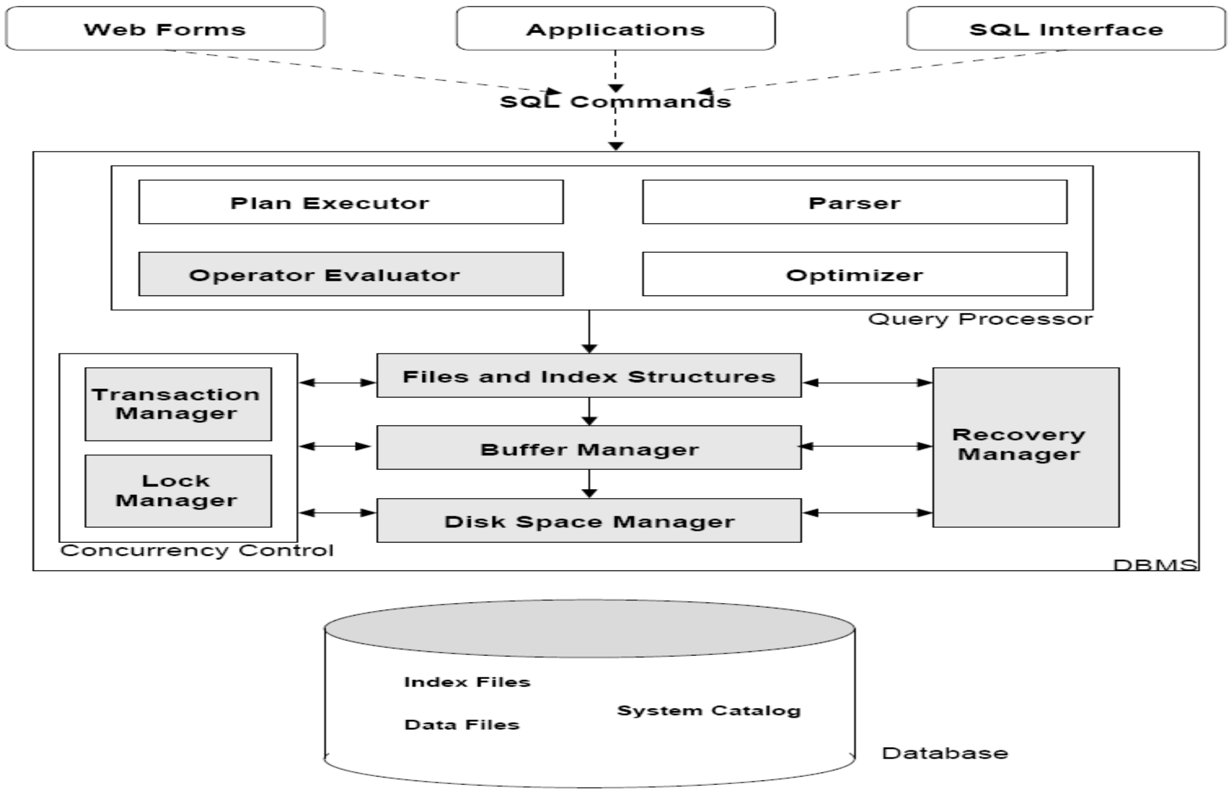

Figure 3: DBMS Architecture

From figure 3, and the project description, the sections that we will focus on in the DBMS architecture should now be obvious. From the top layer, you will be creating an *SQL Interface* which will be the graphical user interface that you will develop. In the query processor, the modules that you will develop, will the be *Parser* and the *Operator Evaluator*. In the Concurrency Control module, you will be developing a basic Lock Manager. The next section that you will develop will be the *Files* and *Index Structure*. Finally, you will implement the *Index Files* and *Data Files*.

NB: The appropriate data structures as it relates to indexing and file organization must be given careful consideration. The type of application that you will build, can either be a web application or a desktop application.

Since this is a group work (groups of 5), you must use the necessary software tools that supports collaborating on documents and source code eg. Google Docs and GitHub.

BONUS MARKS

Bonus marks will be given for additional functionalities. For example, you could create a basic user management system and implement one of the types of access control methods that we discussed in our security lecture. Bonus marks can only be tackled if you have completed all the requirements above.

Submission:

1) All code (well documented).

2) Document the design of the system.

3) In your design document, discuss some of the assumptions you had to make in creating the system.

SUBMISSION DATE: December 1, 2021 @ 11:59PM
DEMO DATE: December 2, 2021 @ 5:30PM

***** THE END *****