Database Management System Project

Subject: Object-Oriented Programming

Members: OM AMAR (BT23CSE106)

ASHISH TUKARAM PAKHLE (BT23CSE107)

DEBASISH MONDAL (BT23CSE108)

TANAY SUNIL UMRE (BT23CSE109)

Instructor: DR MILIND PENURKAR

Project Report

Project Statement : To create an Database Management System using OOPS Concepts and Filehandling.

Concepts Used :

- Basic OOPS
- Multiple Inheritence
- Function Overloading
- Operator Overloading
- Assignment Constructor
- Standard Template Library
- Static Variables and Functions
- Abstract Classes
- Virtual Destructor
- Exception Handling
- File Handling

Classes Used :

1. ParsedQuery (Struct)

It consists of the following members:

- Command : contains the command in string
- Table : contains the name of the table to be modified
- Delete1 : To differentiate between the DELETE from table operation and DELETE table operation
- Columns : Contains the name of the columns to be operated on
- ColumnTypes : Types of the columns in case this is a INSERT command.
- Values : In case this is a INSERT command, these are the values to be inserted.
- conditionColumn : Column on which we are applying the condition

- conditionOperator : The operation through which we are considering the rows to be selected
- conditionValue : The value we are comparing it with in case of a WHERE clause
- other : Name of the second table, in case the command asks to COPY, or check ISEQUAL.

2. <u>Database Object</u>

This is an Abstract Class, and is then overridden by the Classes Table and View. The Database Object consist of a virtual Destructor.

3. <u>Table</u>

It consist of the following members:

- columns : Consists the name of the columns
- columnTypes : Consists of the datatypes of the columns
- rows : consists of the data, i.e. the rows of the table, therefore it is a vector of vector.
- writeTofile() : This function writes the columns and the rows to the csv file so as to keep it saved.
- It has three constructors, one is a copy constructor, which takes another Table as input and then copies it, second constructor takes the column names, data etc. as the input and then creates the Table, whereas the third takes name as the input and creates an empty table.
- loadFromfile() : This loads all the data present in the given filename and saves it in the form of a Table.
- addColumn() : This function is used to add columns.
- addRow() : To add new record to the table.
- updateRows() : This is used to update the Rows using the UPDATE command.
- deleteRows() : This is used to delete the Rows using DELETE command.
- selectRows() : Selects the specified rows and return it in the form of a vector of a vector.
- Operator== : Here we have done operator overloading, it is used to check whether two tables are identical or not.

4. <u>Database</u>

- Tables : It consists of a map of tables, which has Tablename and the corresponding Table Object.
- Views : It consist of a map of view and the corresponding view object.
- Count : This is a static variable, which stores the count of the number of tables created.
- Database() : This is the constructor , which loads all the already existing Tables into the code.
- TableCount() : Returns the count of table stored in the static variable.
- loadExistingTable() : Loads all the existing tables, that is files with the extension '.csv'.
- createTable() : Inserts a newtable into the map.
- getTable() : Returns an existing table.
- deleteTable() : Deletes Table from the map and from the filesystem.

5. **<u>QueryParser</u>**

- db : This is the database that we are operating on.
- Queryparser() : This initializes the Queryparser with the database that we are using and is the constructor.
- parse() : This divides the Query into subparts, for example CREATE TABLE .. is divided into CREATE,TABLE,.. , so as to make it easier to execute the command, which is then stored in the Parsed Query struct.
- Execute() : Once it has been parsed, we execute the command by using the db present in the private members.

<u>Logic :</u>

1. Take the input from the user, then tokenize it, that is divide it into command, tablename, and other parts depending on the command used.
2. After parsing it, execute it, which then uses the database object to call the Table with that tablename and then perform the given operation on it.
3. After performing the given operation, close the database and then reopen it so as to reflect the newly created table as well.

Command List

```
>> CREATE TABLE tablename ( columnname datatype .. )
>> INSERT INTO tablename VALUES data
>> SELECT * FROM tablename
>> SELECT col1 , col2 .... FROM tablename
>> SELECT * FROM tablename WHERE columnname = data
>> UPDATE tablename SET columnmname = newdata WHERE columnname = olddata
>> DELETE FROM tablename WHERE columnname = data
>> COPY FROM oldtable TO newtable
>> DELETE tablename
>> ISEQUAL tablename tablename1
>> SHOW TABLES
>> --Comment
>> EXIT
```