COMSATS University Islamabad
Department of Computer Science
Lab Project - Fall 2024

Class: BCS-2B **Marks:** 50
Subject: CSC103-Programming Fundamentals **Instructor:** Mr. Qasim Malik

[CLO-5] Build a medium size application in a team environment

Project: Database Management System Implementation

The project involves the development of a Database Management System that allows users to easily manage their data. From the user's point of view, the system will model the data in the form of tables containing rows and columns. However, internally, the system will store the tables in files, either in *comma-separated values* (CSV) or *tab-separated values* (TSV) format. To interact with the system, a *case insensitive* query language has been designed, enabling users to interact in a natural way to manage both tables and the data inside them. Users will be able to perform the following operations:

- Create Statement: To create a new table with the name and columns provided by the user
- Drop Statement: To delete an existing table
- Show Statement: To show all the tables
- Insert Statement: To insert a new row in a given table
- Select Statement: To retrieve rows from a given table
- Update Statement: To update rows in the table
- Delete Statement: To delete rows from a given table
- Help Statement: To display the syntax of each of the above 7 statements
- Exit Statement: To terminate the program

For each of the above statements except the HELP and EXIT statements, two syntaxes have been designed. Given below are these syntaxes along with their examples of use and their expected behavior. The keywords used in the statements are expressed in capital letters to differentiate it from what can be customized.

CREATE Statement:
Syntax 1: CREATE TABLE table_name (column1, column2, column3,…)
Syntax 2: CREATE table_name HAVING column1, column2, column3,…

Example:
CREATE TABLE student (name,regno,address)
CREATE student HAVING name,regno,address

Behavior:
The above statements should create a file named *student.txt* and place the column names (name, regno, address) in the first line, using either a comma or a tab as a separator. The first line of the file will always represent the header of the table. The statement should display a confirmation message. The table name and column names should follow the same rules as Java identifiers.

DROP Statement:
Syntax 1: DROP TABLE table_name
Syntax 2: DROP table_name TABLE

Example:
Syntax 1: DROP TABLE student
Syntax 2: DROP student TABLE

Behavior:
The above statements should delete the *student.txt* file proceeded by a warning message and followed by a confirmation message or error message.

SHOW Statement:
Syntax 1: SHOW ALL
Syntax 2: SHOW TABLES

Example:
Syntax 1: SHOW ALL
Syntax 2: SHOW TABLES

Behavior:
The above statements should display the name of all the text files without their extension.

INSERT Statement:
Syntax 1: INSERT INTO table_name VALUES (value1, value2, value3, ...)
Syntax 2: INSERT IN TABLE table_name VALUES (value1, value2, value3, ...)

Example:
Syntax 1: INSERT INTO student VALUES ("Haq","FA16-BCT-099","Islamabad")
Syntax 2: INSERT IN TABLE student VALUES ("Haq","FA16-BCT-099","Islamabad")

Behavior:
The above statements should open the *student.txt* file and append a new line containing "Haq","FA16-BCT-099","Islamabad". The values for the columns must be provided in double quotes and should also be saved in a similar way.

SELECT Statement:
Syntax 1: SELECT FROM table_name WHERE column = value ORDER BY column
Syntax 2: SELECT FROM TABLE table_name HAVING column = value SORT BY column

In SELECT statement, WHERE and ORDER BY clauses are optional. If WHERE clause is skipped, all the data should be displayed. If ORDER BY clause is missing, the order will be the same as listed in the file.

Example:
Syntax 1: SELECT FROM student WHERE address = "Islamabad" ORDER BY name
Syntax 2: SELECT FROM TABLE student HAVING address = "Islamabad" SORT BY name

Behavior:
The above statements should open *student.txt* file, find out the all the lines that contain "Islamabad" under address column, sort them alphabetically by name, and display all those lines in a well formatted way with the header as shown below:

name	regno	address
------	-------	---------
Haq	FA16-BCT-099	Islamabad

UPDATE Statement:
Syntax1: UPDATE table_name SET column = value WHERE column = value
Syntax2: UPDATE TABLE table_name SET column TO value HAVING column = value

Example:
Syntax1: UPDATE student SET address = "Lahore" WHERE name = "Haq"
Syntax2: UPDATE TABLE student SET address TO "Lahore" HAVING name = "Haq"

Behavior:
The above statements should open the *student.txt* file, create another temporary file, copy all the lines from *student.txt* to the temporary file except for the lines where *name* column value is "Haq" in which case the address value should be first replaced to "Lahore" before copying it to the temporary file. Finally, the temporary file will be saved, *student.txt* file will be deleted, and the temporary file will be renamed to *student.txt*. The statement should display a confirmation message showing how many rows were updated.

DELETE Statement:
Syntax1: DELETE FROM table_name WHERE column = value
Syntax2: DELETE FROM TABLE table_name HAVING column = value

Example:
Syntax1: DELETE FROM student WHERE name = "Haq"
Syntax2: DELETE FROM TABLE student HAVING name = "Haq"

Behavior:
The above statements should open the *student.txt* file, create another temporary file, copy all the lines from *student.txt* to the temporary file except for the lines where *name* column value is "Haq". Finally, the temporary file will be saved, *student.txt* file will be deleted, and the temporary file will be renamed to *student.txt*. The statement should display a confirmation message showing how many rows were updated.

HELP Statement:
This statement should display the syntax for each of the above statements.

EXIT Statement:
This statement should terminate the program.

Project Groups

The class of 48 students has been randomly divided into 24 groups of two students each. Each group will implement the system for the statements syntaxes and the file type assigned to them. An Excel file will be shared, containing the group formation, syntaxes, and the file type assigned to each group.

User Interface

The program should use a command line interface for all its operations. Once your program starts, it should only display the following prompt:

>>

The user input will always be one of the eight statements entered in a single line. In case of an invalid statement, the program should display an error message. The program should also perform error checking within the statements.

What to Submit

- Your java code file via CUI portal
- A well-formatted word file documenting user manual for your developed system. It should also have your java code in its appendix.

Evaluation Criteria

- *Correctness:* Your program should correctly implement all the statements, exhibiting desired behaviors
- *Clarity:* Write clean and modular code
- *Submission Guideline*: Ensure to follow the submission guideline