
Database Project (DBMS+JDBC) Project Report

Introduction

This project was assigned in Programming-2 (Object Oriented Programming) course by Professor Dr. *Khaled Nagi* in Friday 10th of November, 2017. Due on Friday 1st of November, 2017.

Report is delivered to the Teaching Assistants; Eng. *Ahmed Hamdy* and Eng. *Alaa*.

The project was divided into two main phases which are:

- 1) Database Management System (DBMS) - Phase I.
- 2) Java Database Connectivity (JDBC) - Phase II.

Delivery By:

Mohamed Mashaal, Youssef Ali, Hesham Medhat, Ahmed Osama
IDs : 60 73, 70, 03

Index

Introduction	1
Names and IDs	1
Index	2
Overview	3
Extra Features	3
Team Contribution	4
Design Decisions	5
Design Patterns	6
Assumptions	7
UML Diagrams	7
Sample Runs	x
User Manual/Guide	x

Overview

The two main phases were tested on the online tester (onlinetester.tk). Our team was the:

1. First to finish all test cases of the DBMS in Phase I successfully with 20/20 score.
2. Second to finish all test cases of the JDBC in Phase II successfully with 20/20 score.

Our projects relied on using the design patterns that we have learnt throughout the course to apply the acquired knowledge and put these design patterns to the test and best-fit implementation *if necessary*.

Extra Features

- Graphical User Interface (GUI).
- Logging.
- Multiple selecting.
- Smart query parsing and responding.

Team Contribution

Youssef Ali:

- DBMS
- JDBC
 - Statement Interface.
 - Part of ResultSetMetaData Interface.
 - Testing over online tester.
 - Report.

Mohamed Mashaal:

- DBMS
- JDBC
 - Driver and Connection Interface
 - Testing over online tester
 - Graphical User Interface (GUI) for JDBC.
 - Report.

Hesham Medhat:

- JDBC
 - ResultSet interface implementation.
 - Code and workflow design.
 - Java logging.
 - Basic cleanup.
 - Report.

Ahmed Osama:

- DBMS
 - Graphical User Interface (GUI) for DBMS.
- JDBC

-
- Part of ResultSetMetaData implementation.
 - Code cleaning and refactoring.

Design Decisions

For proper handle of functionalities, and in addition to the use of the design patterns used (listed in the next section), we have decided to make these design decisions for the corresponding purposes for each phase:

Phase I: Database Management System (DBMS):

In order to handle queries by the right objects, we have decided to create the package (DBObjects) that contains the classes; DBContainer, Table, Column, Record.

This is to serve the composition to satisfy *The Principle of Least Knowledge* or *Law of Demeter* to avoid coupling.

Phase II: Java Database Connectivity (JDBC):

To make the iterating and visualisation of the queried data easier, we have decided to make the ResultSet implementation scrollable. This allows the user to go back and forth while iterating over the data of the result of the query.

The user can also jump the head to any required column addressed by its index (corresponding to query order) or the name of the column for easier access.

Although our DBMS returned the queried data through a 2D array 0-indexed, to serve the JDBC implementation correctly, we handled the 0-to-1-index internally in the ResultSet and the ReseltSetMetaData implementations.

Design Patterns

In Phase I: Database Management System (DBMS):

1. MVC (Model-View-Controller) DP

Which is pretty clear from the division of packages.

2. Variation on Composite DP

As explained in the previous section for the Database representation to serve the composition to satisfy *The Principle of Least Knowledge* or *Law of Demeter* to avoid coupling.

3. Attempted Singleton DP

In *DatabasImp* class. However, we couldn't due to the expected instantiation technique in the online tester.

In Phase II: Java Database Connectivity (JDBC):

1. MVC (Model-View-Controller) DP

Naturally to keep the consistency. We had one GUI as the API.

2. Singleton DP

Its implementation indeed worked with the JDBC tests for *DatabasImp*. Also applied Singleton DP for the *DBLogger* in order to have all interfaces use one log rather than several ones.

3. Adapter DP

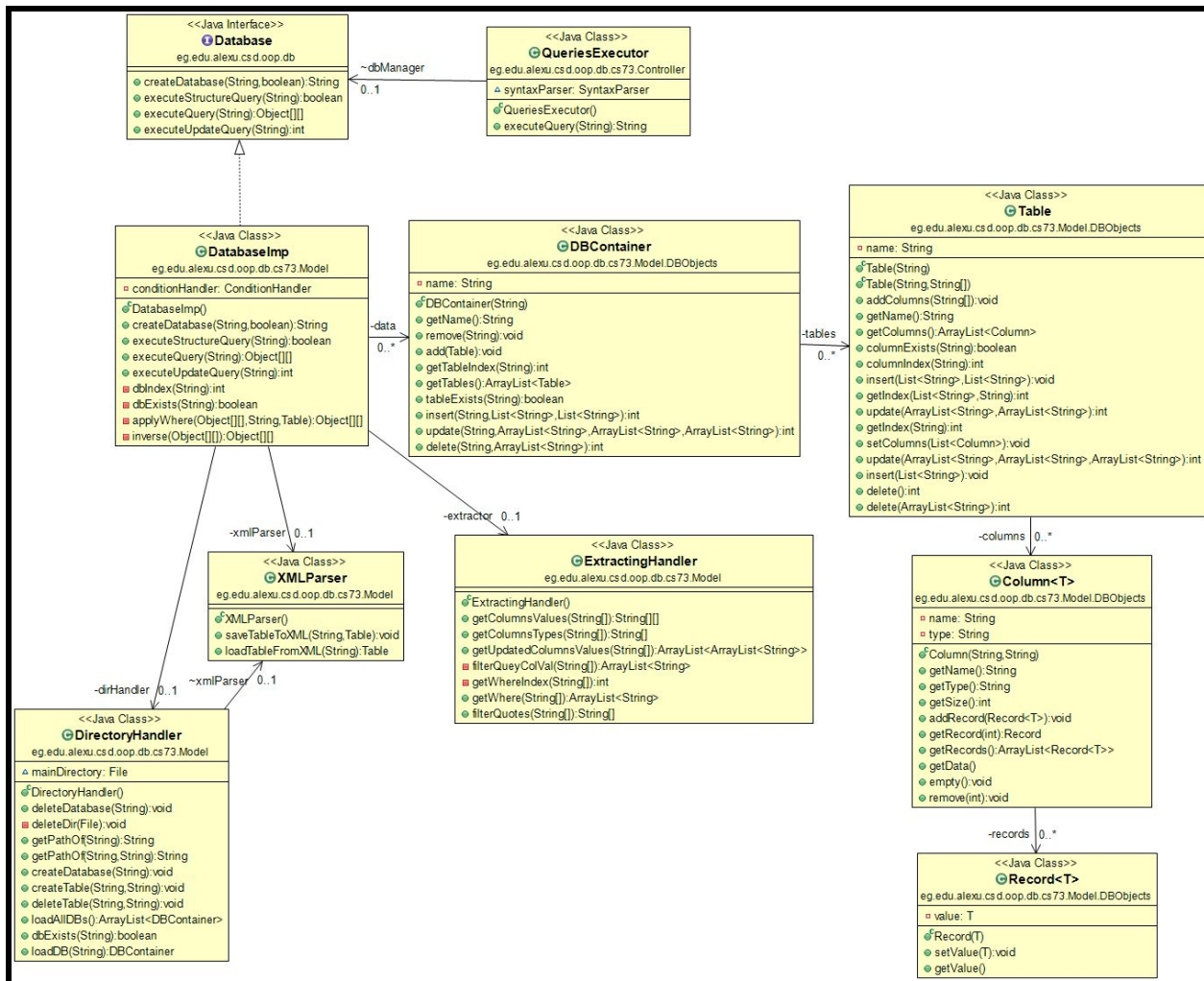
Adapted the *Statement* class to the *DatabasImp* of DBMS.

Assumptions

- The DBMS column names and SQL queries (of course) are case insensitive.
- We have followed the JDBC interfaces' documentation carefully without making any extra assumptions to avoid confusion or misinterpretation for any user or client.

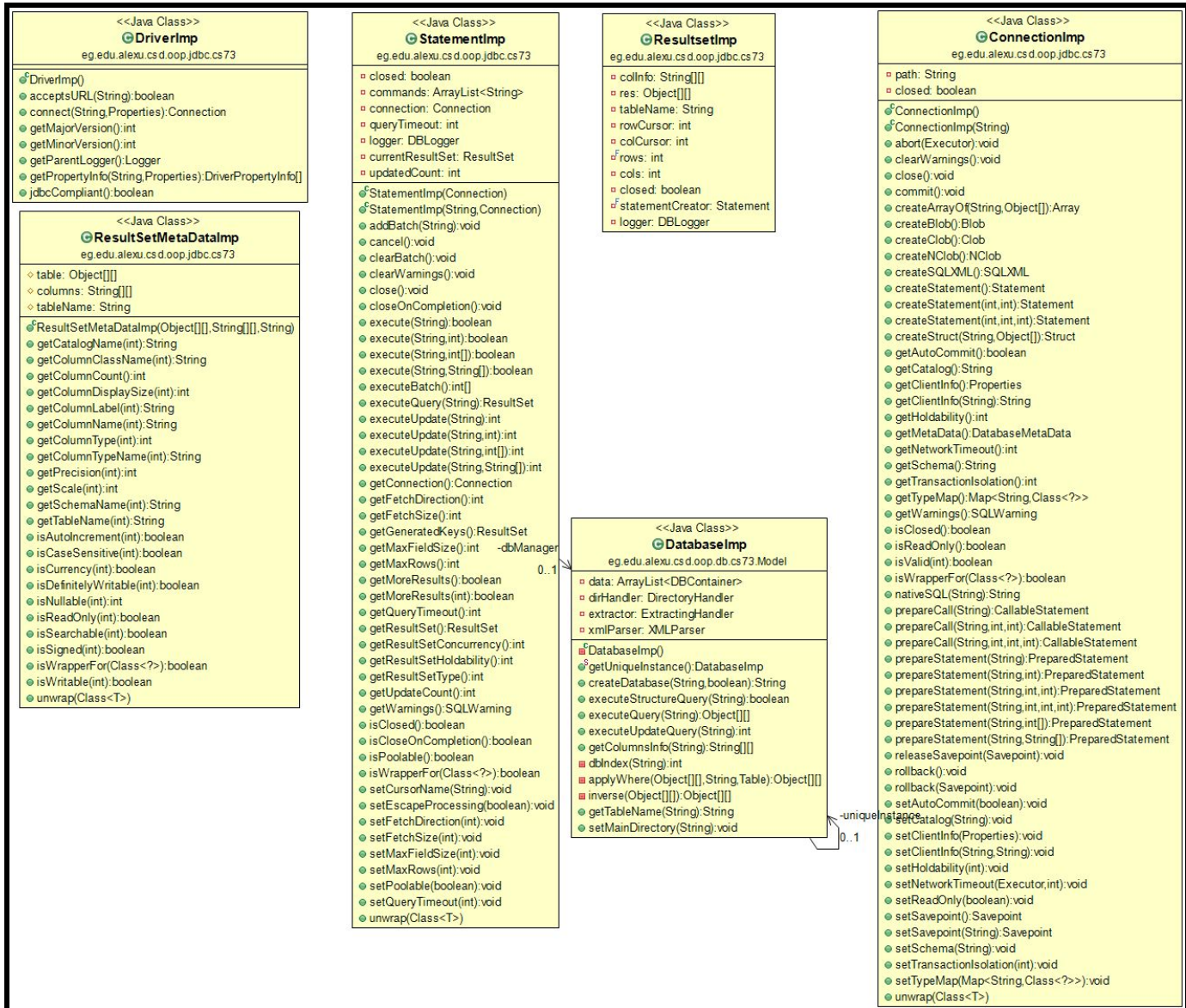
UML Diagrams

DBMS Class Diagram

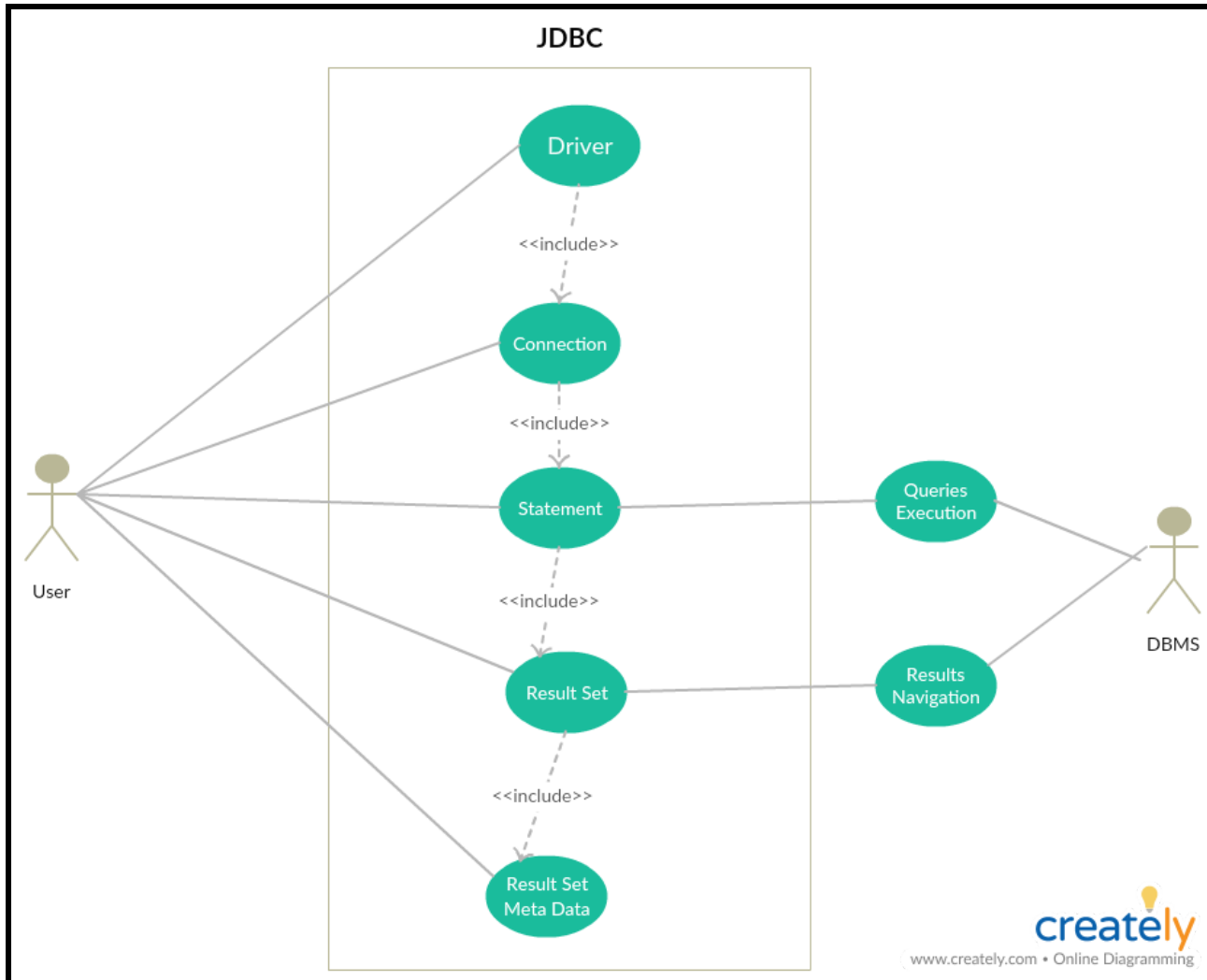


JDBC UML Diagram

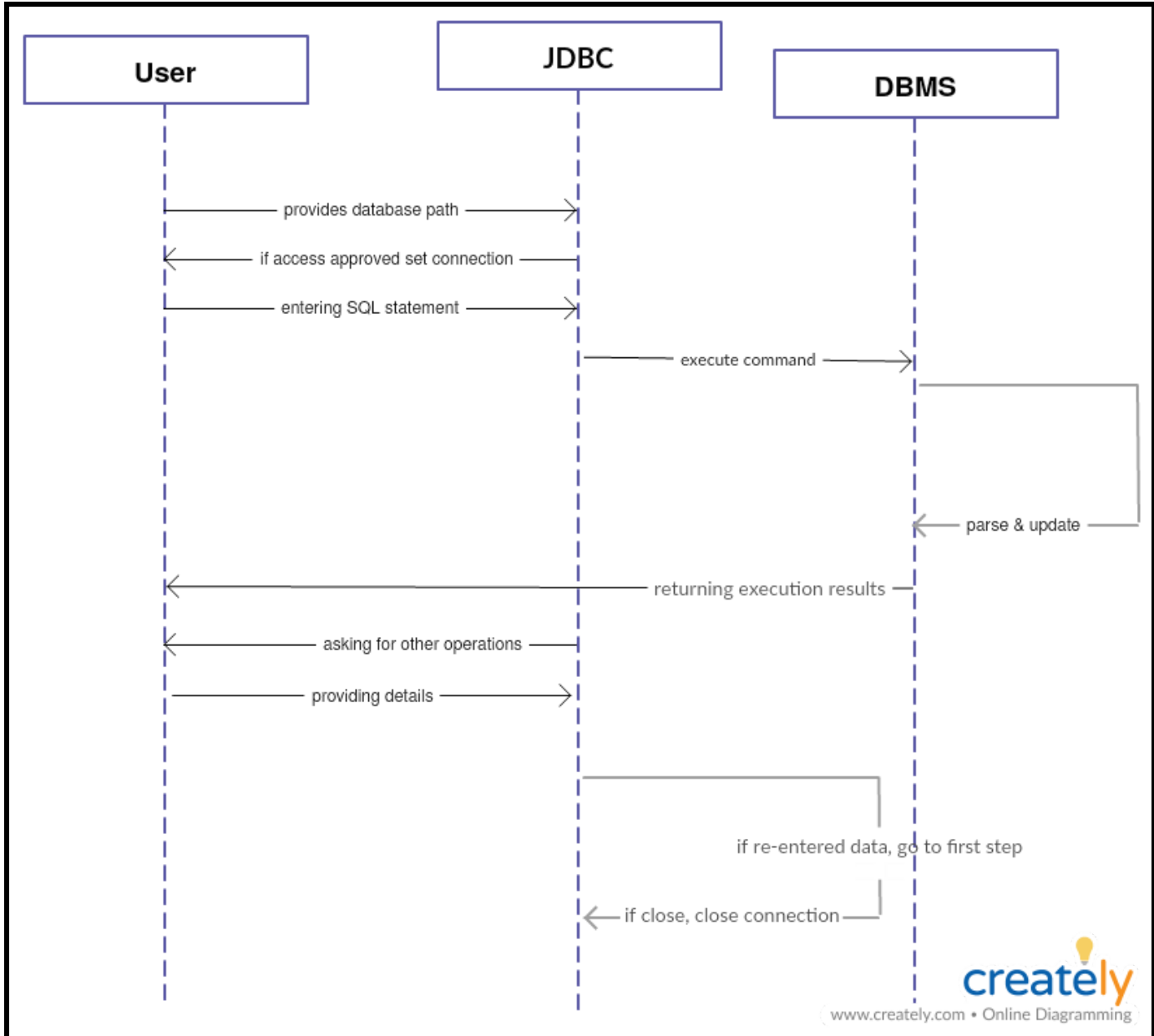
Class Diagram



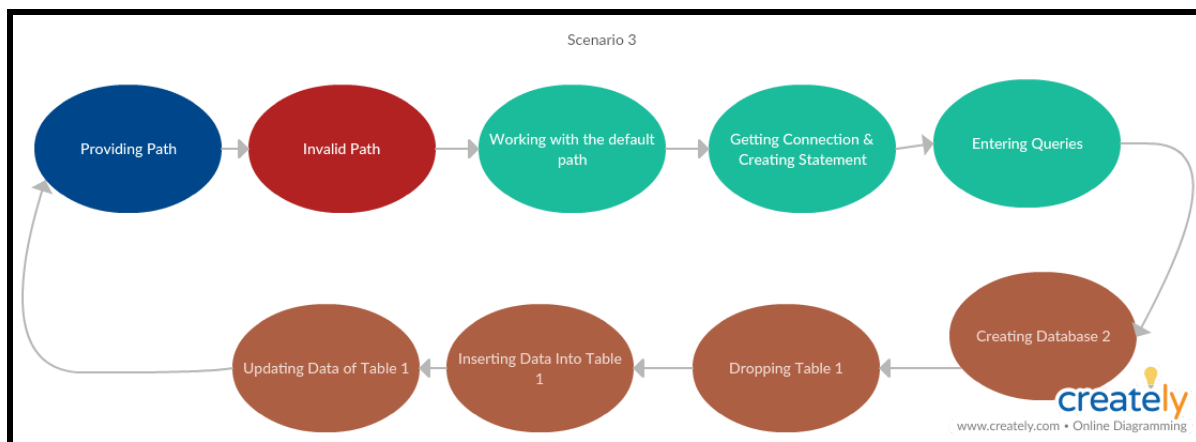
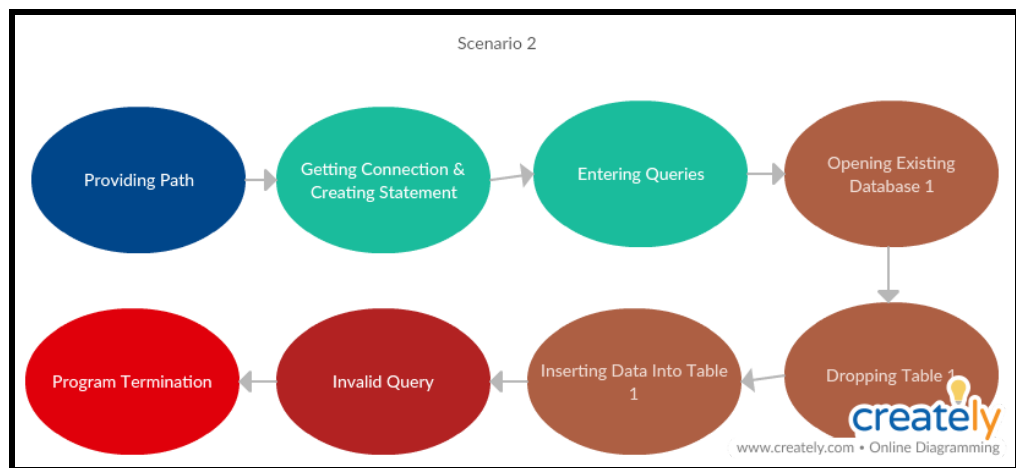
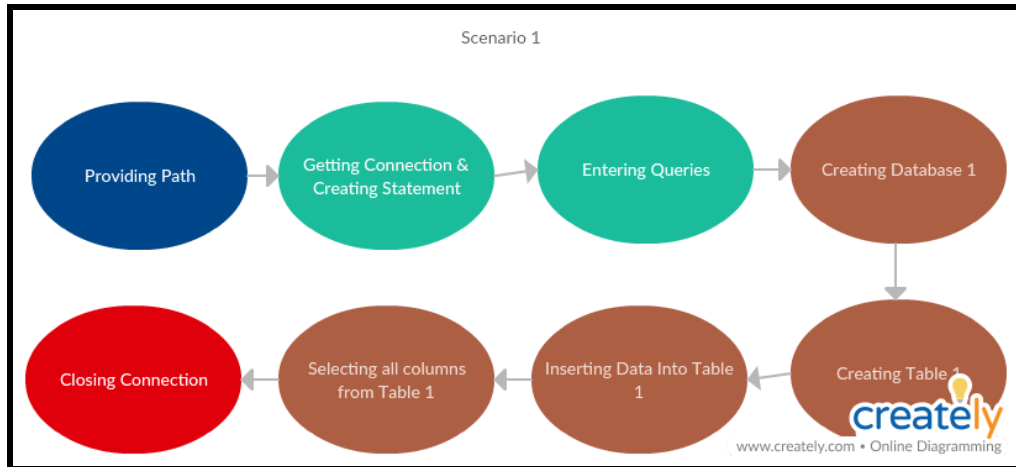
Use Case Diagram



Sequence Diagram

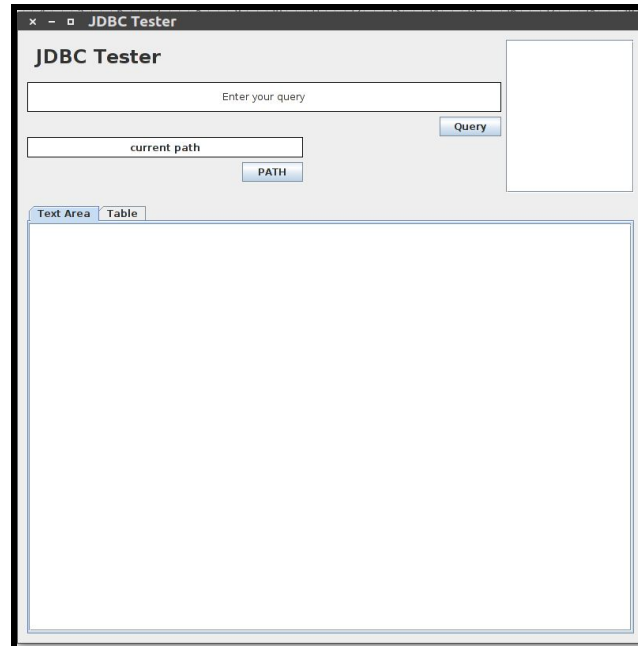


State Diagrams

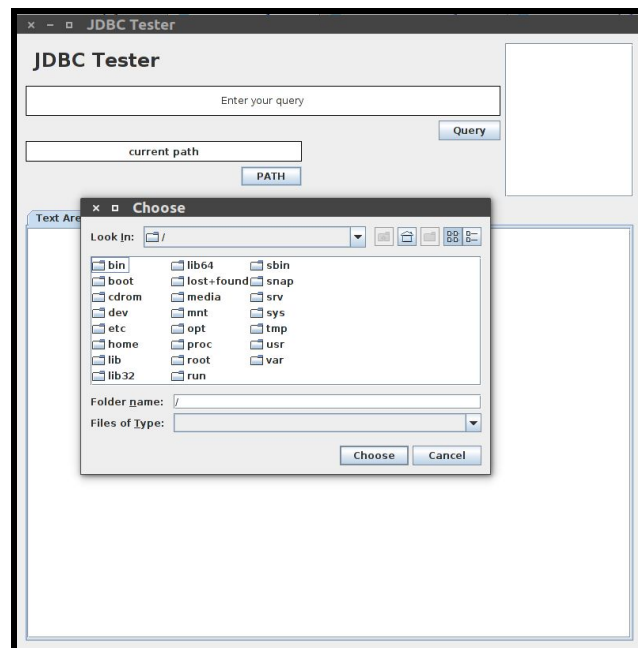


Snapshots

Main window



Changing the main path



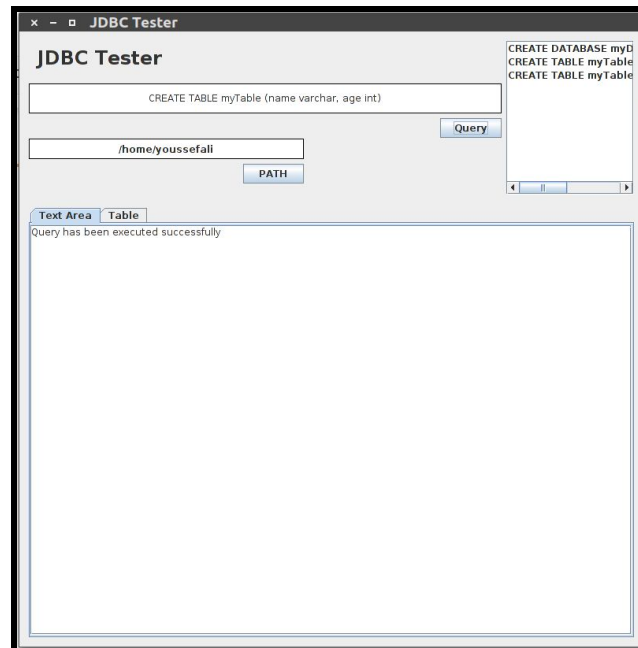
Database creation



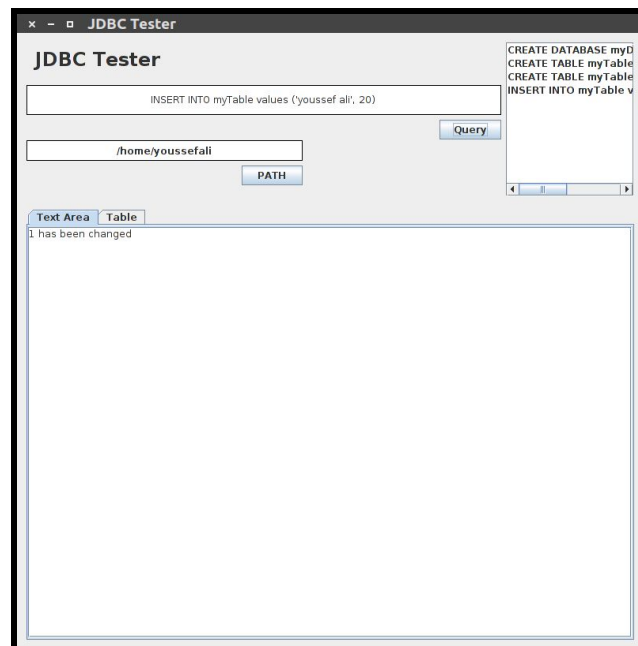
Entering wrong query



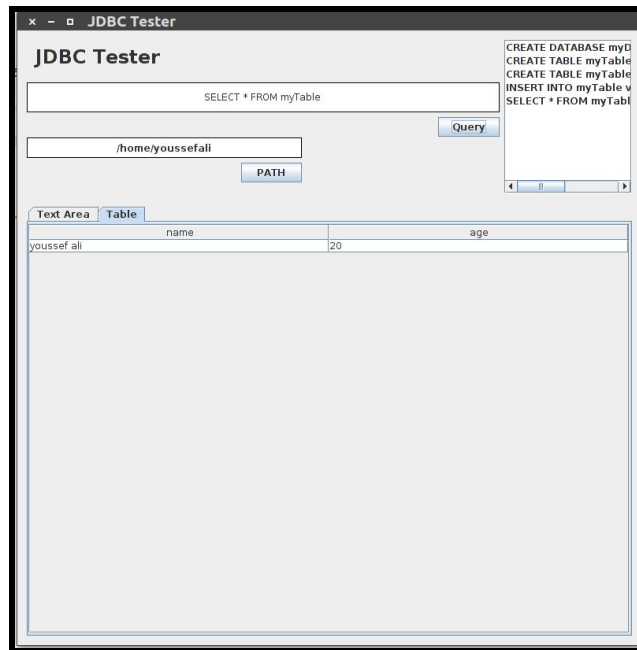
Table creation



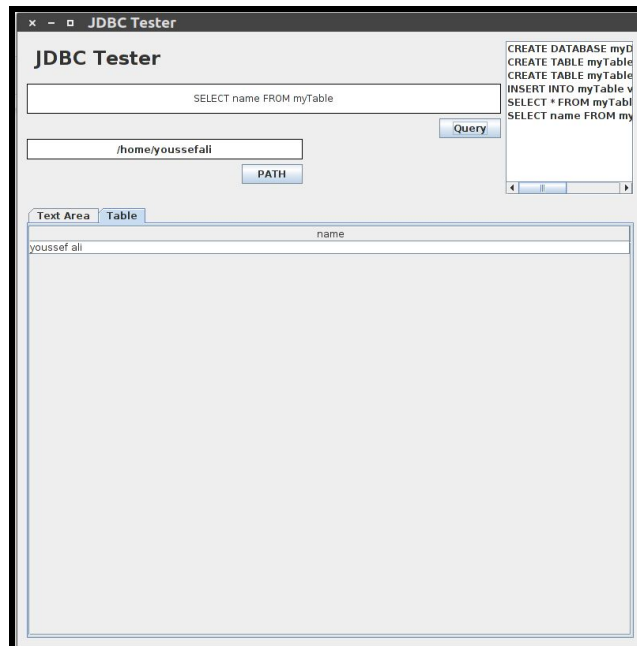
Insertion into table



All columns selection



One column selection



Multi column selection

The screenshot shows the JDBC Tester application window. At the top, the title bar reads "x - JDBC Tester". The main window has a title "JDBC Tester". Below the title, there is a text area containing the SQL query "SELECT name,age FROM myTable". To the right of this text area is a "Query" button. Below the query text area is a text field containing the path "/home/youssefali" and a "PATH" button. To the right of the path text field is a small window containing the following SQL code: "CREATE DATABASE myD", "CREATE TABLE myTable", "CREATE TABLE myTable", "INSERT INTO myTable v", "SELECT * FROM myTable", "SELECT name FROM my", "SELECT name,age FRO". Below the path text field is a tabbed interface with two tabs: "Text Area" and "Table". The "Table" tab is selected, and it displays a table with two columns: "name" and "age". The table contains one row of data: "youssef ali" and "20".

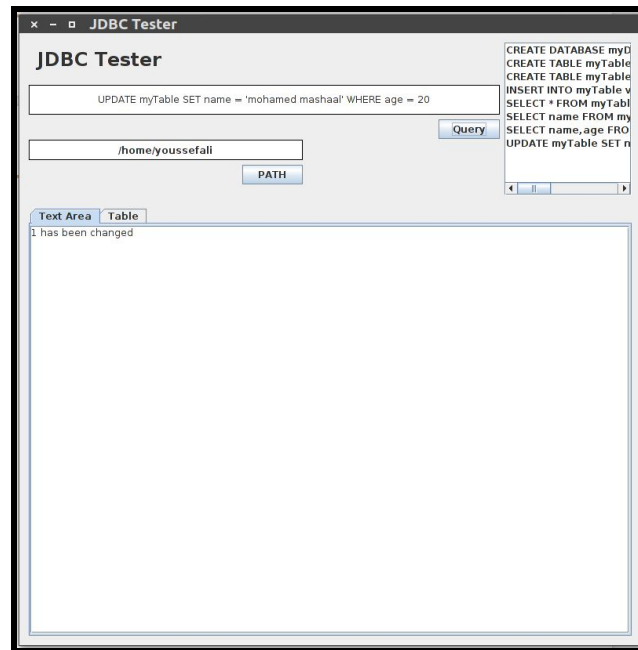
name	age
youssef ali	20

Multi column selection

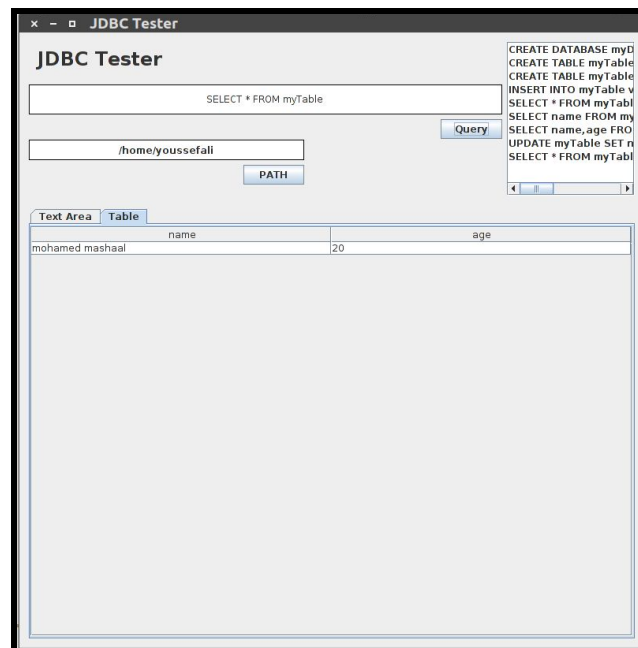
The screenshot shows the JDBC Tester application window. At the top, the title bar reads "x - JDBC Tester". The main window has a title "JDBC Tester". Below the title, there is a text area containing the SQL query "SELECT name,age FROM myTable". To the right of this text area is a "Query" button. Below the query text area is a text field containing the path "/home/youssefali" and a "PATH" button. To the right of the path text field is a small window containing the following SQL code: "CREATE DATABASE myD", "CREATE TABLE myTable", "CREATE TABLE myTable", "INSERT INTO myTable v", "SELECT * FROM myTable", "SELECT name FROM my", "SELECT name,age FRO". Below the path text field is a tabbed interface with two tabs: "Text Area" and "Table". The "Table" tab is selected, and it displays a table with two columns: "name" and "age". The table contains one row of data: "youssef ali" and "20".

name	age
youssef ali	20

Updating columns



Re-selecting the data



DBMS CLI Snapshot

```
Input command :  
CREATE DATABASE DB2  
DATABASE CREATED SUCCESSFULLY  
Input command :  
CREATE Table table____1(column1 int , column2 varchar)  
TABLE CREATED SUCCESSFULLY  
Input command :  
Insert into table____1 values (3 , 'whatever')  
1 ROWS HAS BEEN UPDATED  
Input command :  
select * from table____1  
B whatever  
  
Input command :
```

User Manual/Guide

- Choosing Main Directory for databases :
 - Press Path Button.
 - Choose desired directory, New Connection would be established and current path would be shown in the label.
- Executing Commands :
 - Enter a valid Sql Command in the text field.
 - Press Query Button.
 - In case of Select query , Result would be shown in the Table tab.
 - In case of Create, drop, delete, update, insert Indicating statements would shown in the text area below.
- History :

Latest Commands executed would be shown in the History list.