Report

JDBC

Java Data Base Connectivity.

BY:

- Omar Mohamed Emam   44
- Islam Yousry   14
- Islam Mohamed   12
- Bahaa Khlaf   21

Report Content:

Program Specifications:

- Program creates the driver , connection , statement which are ready to accept your sql query , check it and then transfer it to DBMS to execute it and send the result back (if there is a result) or just edit or create your databases and your tables

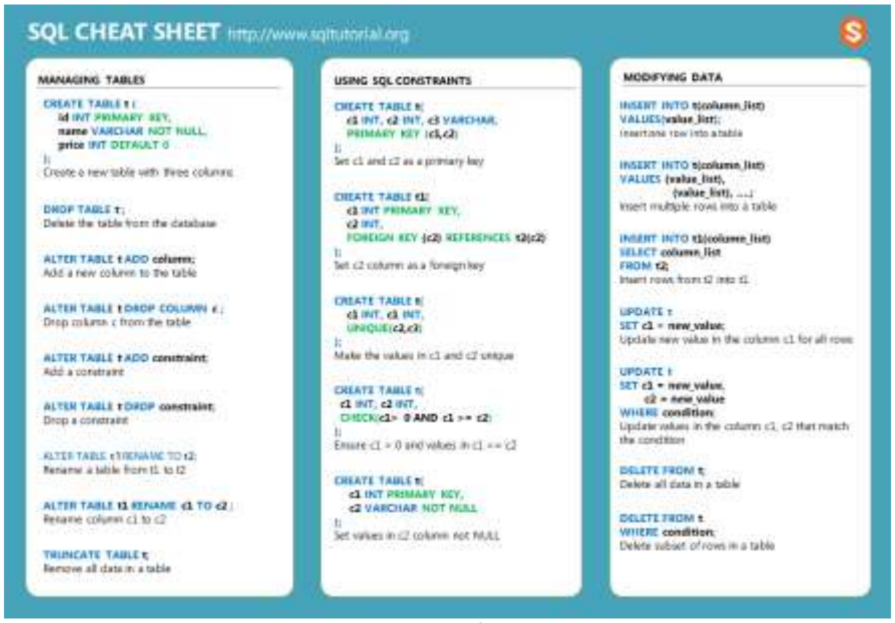
- Program deals with URL as database path in your local machine, so there is no existence for servers and its functions

- Program will keep running even if you entered in wrong syntax, you will be able to retry till you want to stop

- Program runs through self-runnable Cmd app

User Guide:

Run “excucation.bat”, wait creation of connection then enter your SQL query in right syntax and then you can do more queries or exit by typing “end”.



SQL CHEAT SHEET http://www.sqltutorial.org

MANAGING TABLES

```
CREATE TABLE t (
    id INT PRIMARY KEY,
    name VARCHAR NOT NULL,
    price INT DEFAULT 0
);
```
Create a new table with three columns

```
DROP TABLE t;
```
Delete the table from the database

```
ALTER TABLE t ADD column;
```
Add a new column to the table

```
ALTER TABLE t DROP COLUMN c ;
```
Drop column c from the table

```
ALTER TABLE t ADD constraint;
```
Add a constraint

```
ALTER TABLE t DROP constraint;
```
Drop a constraint

```
ALTER TABLE t1 RENAME TO t2;
```
Rename a table from t1 to t2

```
ALTER TABLE t1 RENAME c1 TO c2 ;
```
Rename column c1 to c2

```
TRUNCATE TABLE t;
```
Remove all data in a table

USING SQL CONSTRAINTS

```
CREATE TABLE t(
   c1 INT, c2 INT, c3 VARCHAR,
   PRIMARY KEY (c1,c2)
);
```
Set c1 and c2 as a primary key

```
CREATE TABLE t1(
   c1 INT PRIMARY KEY,
   c2 INT,
   FOREIGN KEY (c2) REFERENCES t2(c2)
);
```
Set c2 column as a foreign key

```
CREATE TABLE t(
   c1 INT, c1 INT,
   UNIQUE(c2,c3)
);
```
Make the values in c1 and c2 unique

```
CREATE TABLE t(
  c1 INT, c2 INT,
  CHECK(c1> 0 AND c1 >= c2)
);
```
Ensure c1 > 0 and values in c1 >= c2

```
CREATE TABLE t(
   c1 INT PRIMARY KEY,
   c2 VARCHAR NOT NULL
);
```
Set values in c2 column not NULL

MODIFYING DATA

```
INSERT INTO t(column_list)
VALUES(value_list);
```
Insert one row into a table

```
INSERT INTO t(column_list)
VALUES (value_list),
       (value_list), ....;
```
Insert multiple rows into a table

```
INSERT INTO t1(column_list)
SELECT column_list
FROM t2;
```
Insert rows from t2 into t1

```
UPDATE t
SET c1 = new_value;
```
Update new value in the column c1 for all rows

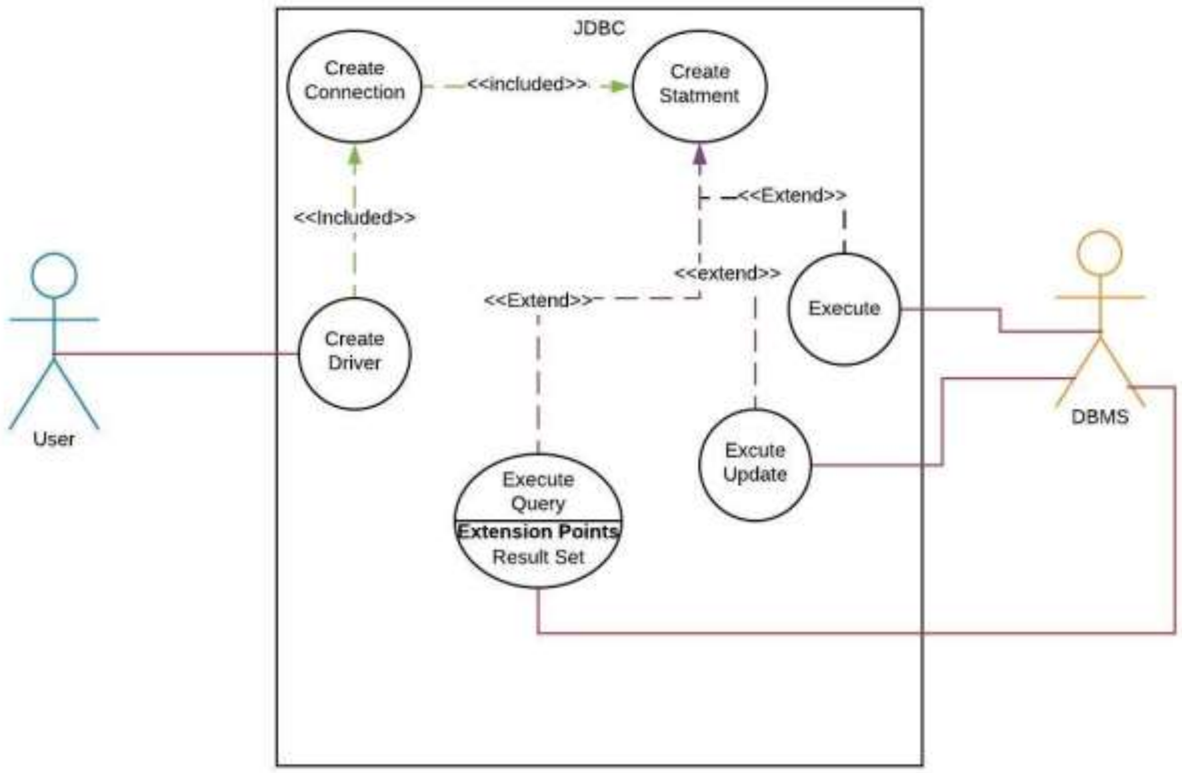
```
UPDATE t
SET c1 = new_value,
    c2 = new_value
WHERE condition;
```
Update values in the column c1, c2 that match the condition

```
DELETE FROM t;
```
Delete all data in a table

```
DELETE FROM t
WHERE condition;
```
Delete subset of rows in a table

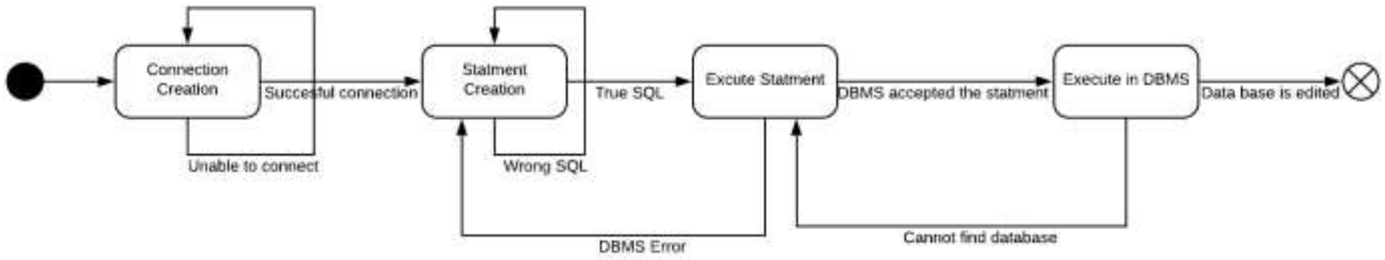
UML Diagrams:

- ### Use Case Diagram :

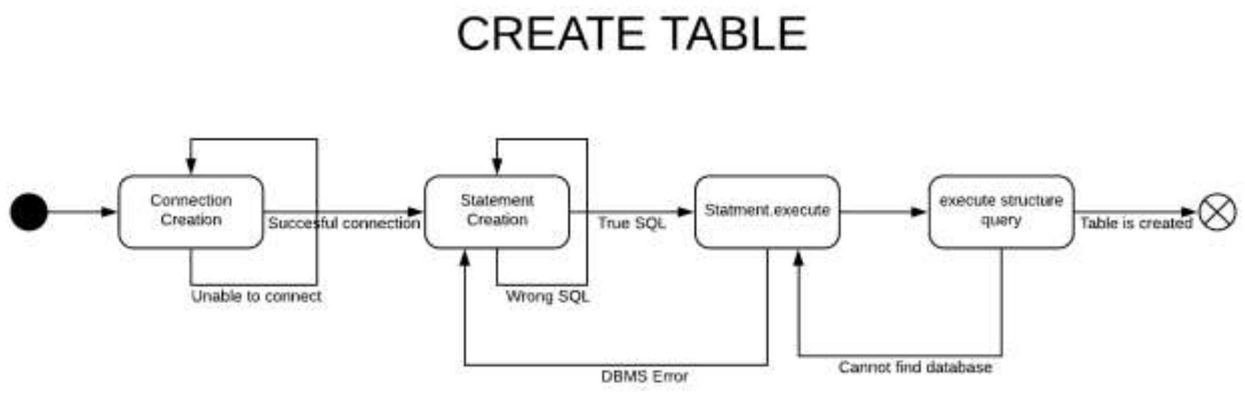


- ### State Diagram For 3 Scenarios :

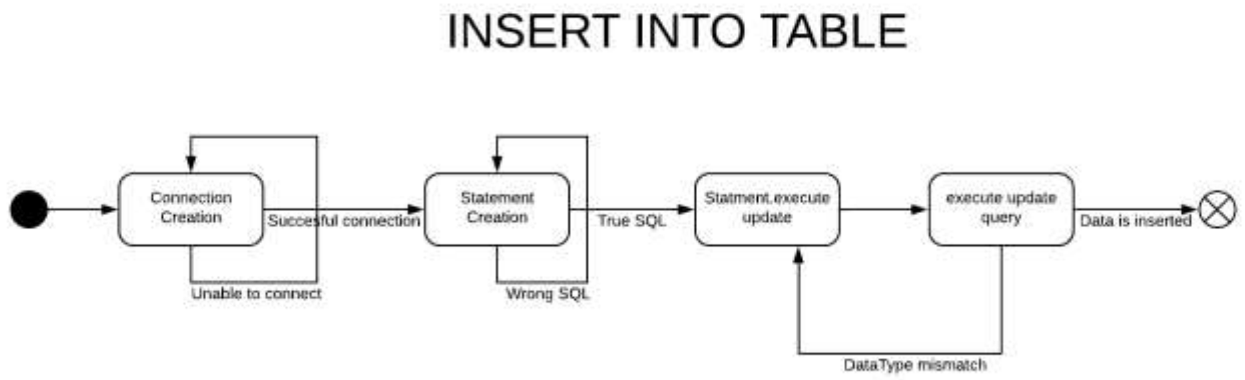
** General Scenario:



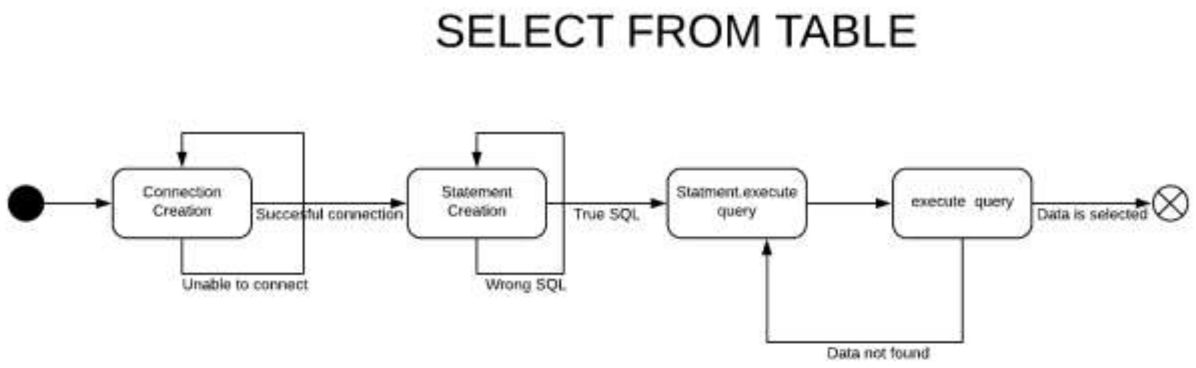
** Scenario #1 (create table):



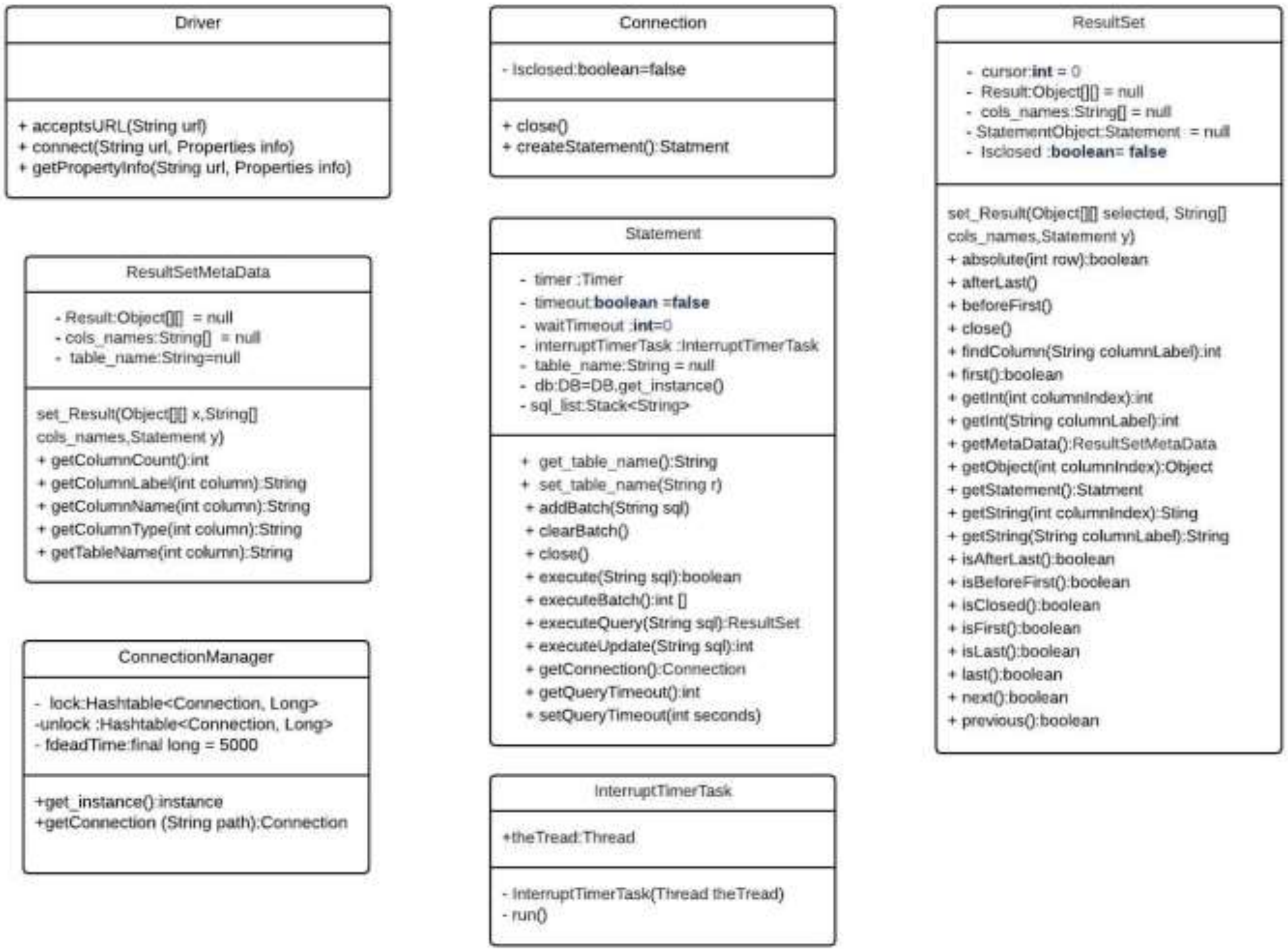
** Scenario #2 (insert into table):



** Scenario #3 (select from table):

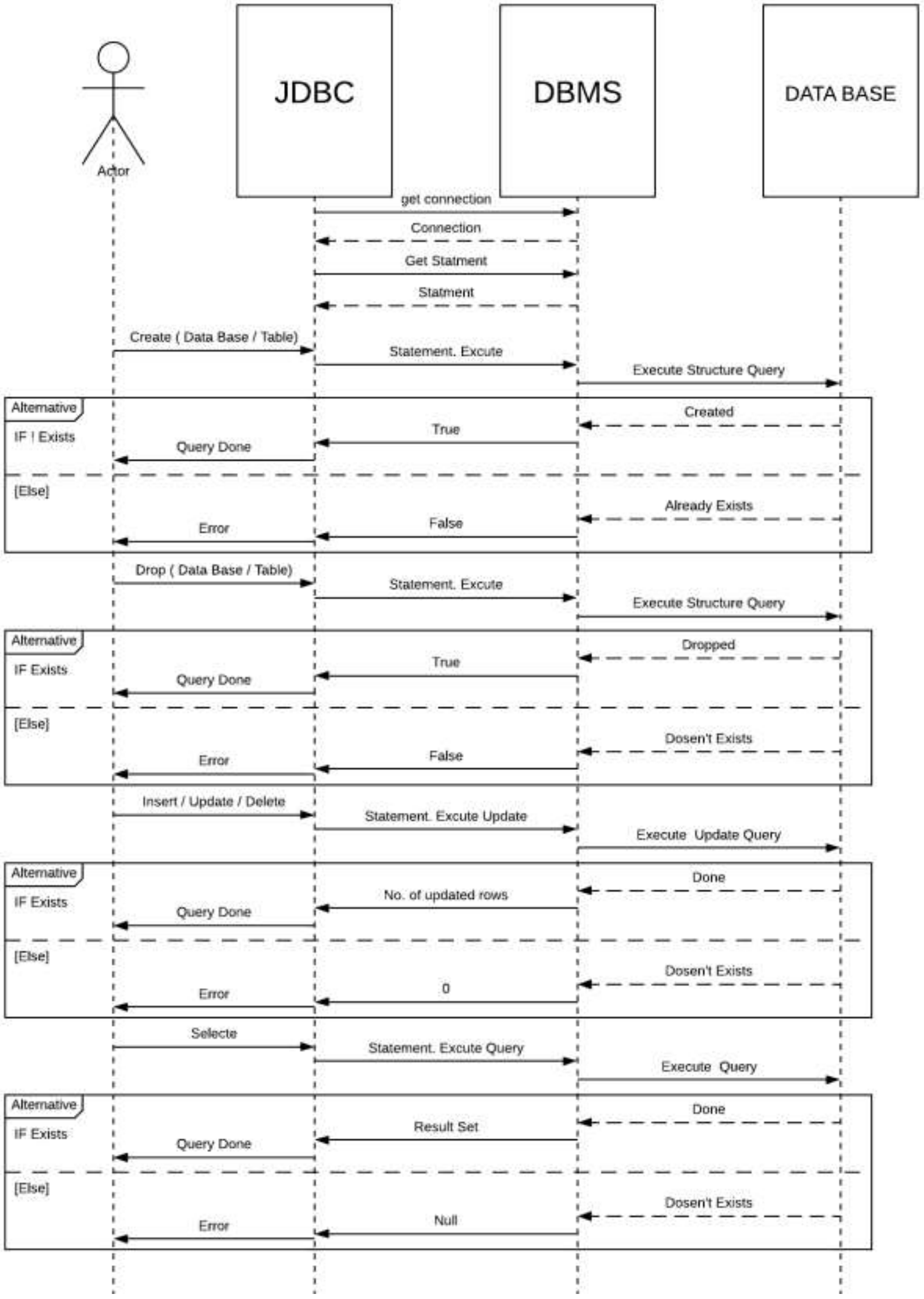


• Class Diagram :



Driver
+ acceptsURL(String url)
+ connect(String url, Properties info)
+ getPropertyInfo(String url, Properties info)
Connection
- Isclosed:boolean=false
+ close()
+ createStatement():Statment
ResultSet
- cursor:int = 0
- Result:Object[][] = null
- cols_names:String[] = null
- StatementObject:Statement = null
- Isclosed :boolean= false
set_Result(Object[][] selected, String[] cols_names,Statement y)
+ absolute(int row):boolean
+ afterLast()
+ beforeFirst()
+ close()
+ findColumn(String columnLabel):int
+ first():boolean
+ getInt(int columnIndex):int
+ getInt(String columnLabel):int
+ getMetaData():ResultSetMetaData
+ getObject(int columnIndex):Object
+ getStatement():Statment
+ getString(int columnIndex):Sting
+ getString(String columnLabel):String
+ isAfterLast():boolean
+ isBeforeFirst():boolean
+ isClosed():boolean
+ isFirst():boolean
+ isLast():boolean
+ last():boolean
+ next():boolean
+ previous():boolean
ResultSetMetaData
- Result:Object[][] = null
- cols_names:String[] = null
- table_name:String=null
set_Result(Object[][] x,String[] cols_names,Statement y)
+ getColumnCount():int
+ getColumnLabel(int column):String
+ getColumnName(int column):String
+ getColumnType(int column):String
+ getTableName(int column):String
Statement
- timer :Timer
- timeout:boolean =false
- waitTimeout :int=0
- interruptTimerTask :InterruptTimerTask
- table_name:String = null
- db:DB=DB.get_instance()
- sql_list:Stack<String>
+ get_table_name():String
+ set_table_name(String r)
+ addBatch(String sql)
+ clearBatch()
+ close()
+ execute(String sql):boolean
+ executeBatch():int []
+ executeQuery(String sql):ResultSet
+ executeUpdate(String sql):int
+ getConnection():Connection
+ getQueryTimeout():int
+ setQueryTimeout(int seconds)
ConnectionManager
- lock:Hashtable<Connection, Long>
-unlock :Hashtable<Connection, Long>
- fdeadTime:final long = 5000
+get_instance():instance
+getConnection (String path):Connection
InterruptTimerTask
+theTread:Thread
- InterruptTimerTask(Thread theTread)
- run()

- Sequence Diagram :



Program Description:

We implemented some function in JDBC interfaces as following:

In `java.sql.Driver:`

- Accept URL: takes the url which must be path of data base and check if it's correct or not
- Connect: Create a new connection between the driver and the database
- Get property Info: Gets information about the possible properties for this driver.

In `java.sql.Connection:`

- Crete Statement: Creates a Statement object that will generate ResultSet objects with the given type and concurrency.
- Close: Releases this Connection object's database and JDBC resources immediately instead of waiting for them to be automatically released.

In `java.sql.Statement:`

- Add Batch: check if sql query is right in syntax or not and add it to stack
- Clear Batch: Remove all queries from the stack
- Execute: Send Sql query to execute structure query function in DBMS
- Execute Update: Send Sql query to execute update query function in DBMS
- Execute Query: Send Sql query to execute query function in DBMS
- Set timeout: makes sure that execute function won't pass the required run time
- Get timeout: get execution time for execute functions

In `java.sql.ResultSet:`

- Absolute : Moves the cursor to the given row number in this ResultSet object
- After Last: Moves the cursor to the end of this ResultSet object, just after the last row.
- Before First: Moves the cursor to the front of this ResultSet object, just before the first row.
- Close: Releases this ResultSet object's database and JDBC resources immediately instead of waiting for this to happen when it is automatically closed.
- Find Column: Maps the given ResultSet column label to its ResultSet column index.
- First: Moves the cursor to the first row in this ResultSet object.
- Get Int: Retrieves the value of the designated column in the current row of this ResultSet object as an Int in the Java programming language.
- Get Meta Data: Retrieves the number, types and properties of this ResultSet object's columns.
- Get Object: Gets the value of the designated column in the current row of this ResultSet object as an Object in the Java programming language
- Get Statement: Retrieves the Statement object that produced this ResultSet object.
- Get String(Int column Index) : Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java programming language.
- Get String (String column Label): Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java programming language.
- Is After Last: Retrieves whether the cursor is after the last row in this ResultSet object.
- Is Before First: Retrieves whether the cursor is before the first row in this ResultSet object.
- Is Closed: Retrieves whether this ResultSet object has been closed.
- Is First: Retrieves whether the cursor is on the first row of this ResultSet object.
- Is Last: Retrieves whether the cursor is on the last row of this ResultSet object.
- Last: Moves the cursor to the last row in this ResultSet object.
- Next: Moves the cursor forward one row from its current position.
- Previous: Moves the cursor to the previous row in this ResultSet object.

Design Patterns:

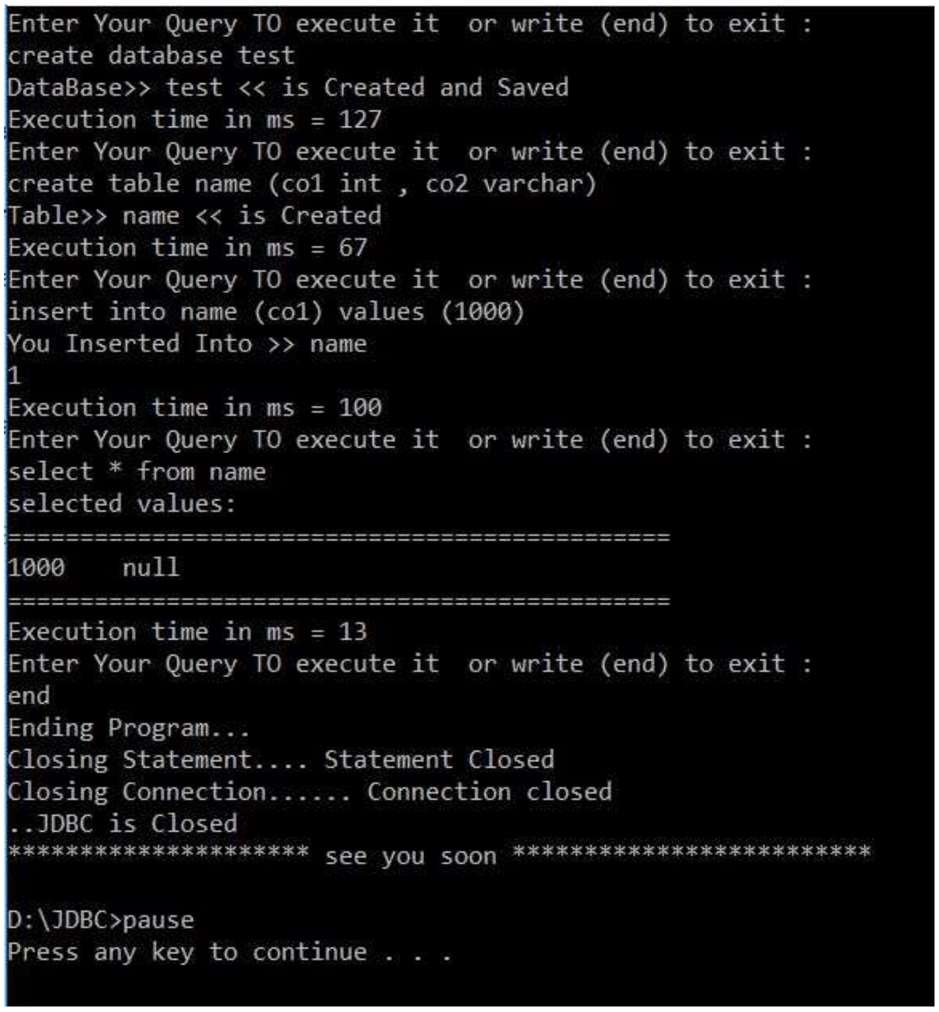
Object Pool design pattern:

Object pool pattern is a software creational design pattern which is used in situations where the cost of initializing a class instance is very high.
Basically, an Object pool is a container which contains some amount of objects. So, when an object is taken from the pool, it is not available in the pool until it is put back.

Singleton design pattern:

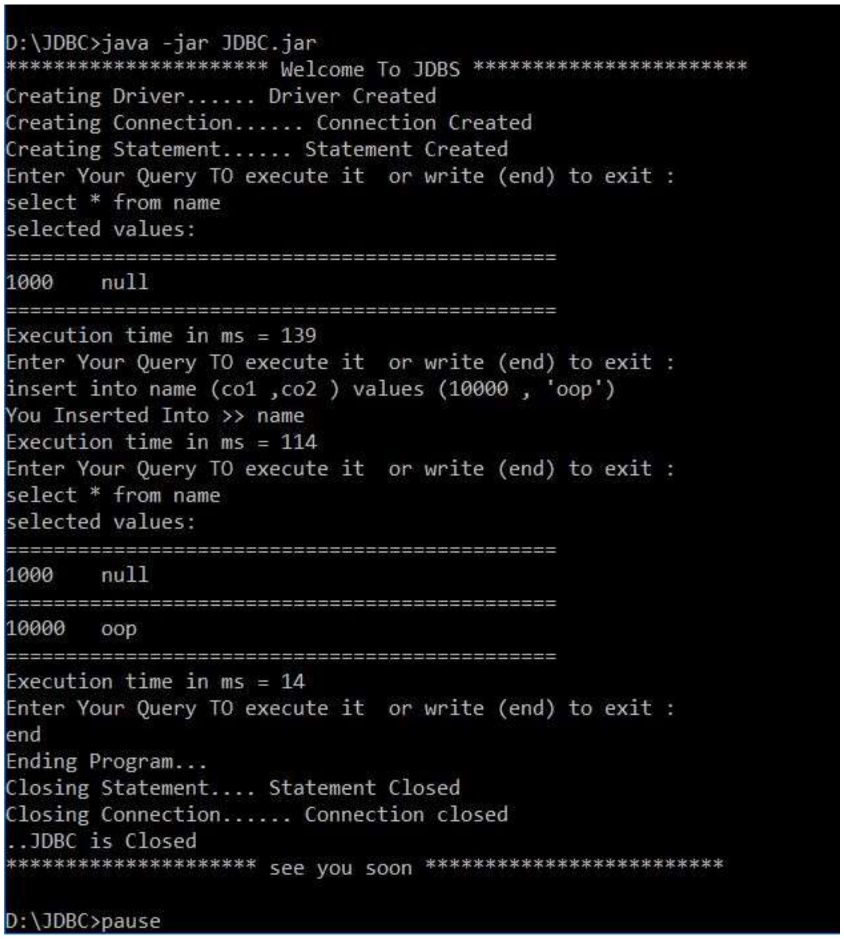
The singleton pattern is a design pattern that restricts the instantiation of a class to one object.

Sample Runs:



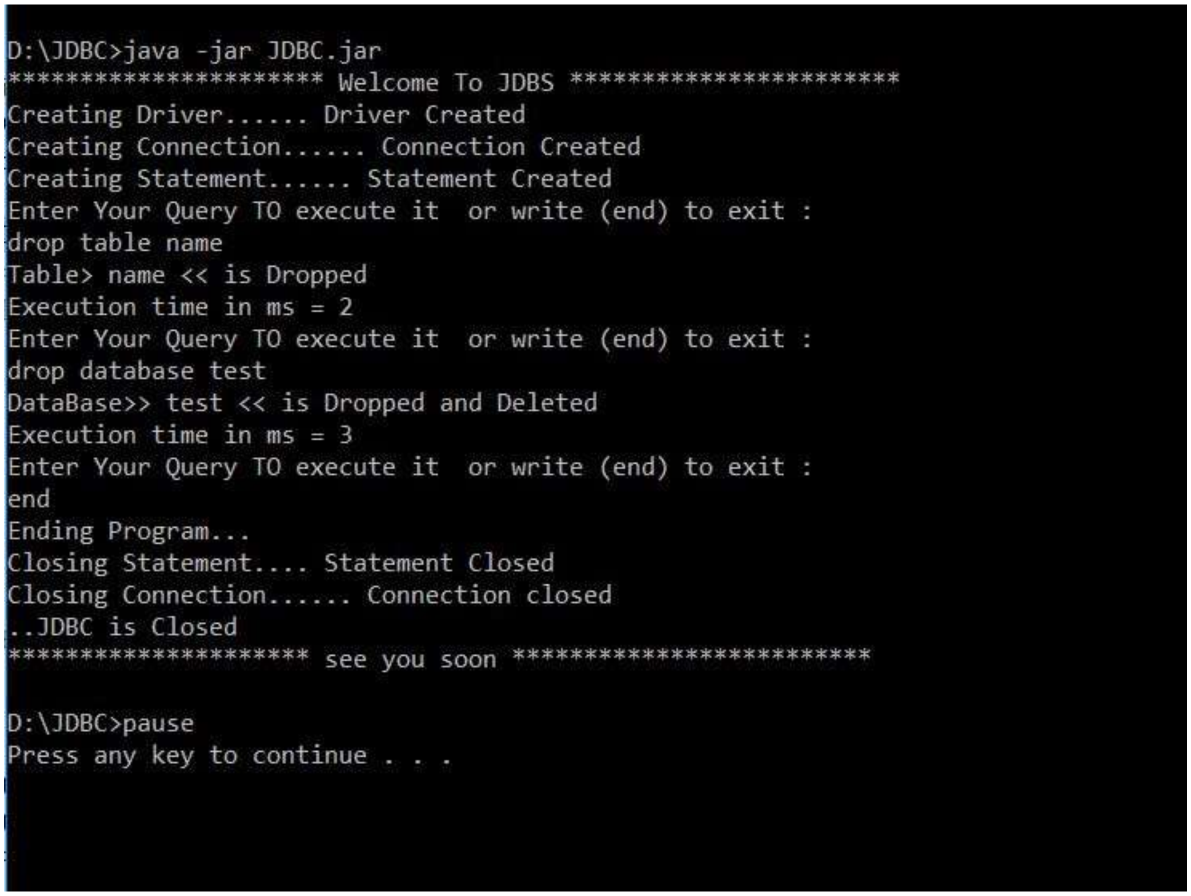
```
Enter Your Query TO execute it  or write (end) to exit :
create database test
DataBase>> test << is Created and Saved
Execution time in ms = 127
Enter Your Query TO execute it  or write (end) to exit :
create table name (co1 int , co2 varchar)
Table>> name << is Created
Execution time in ms = 67
Enter Your Query TO execute it  or write (end) to exit :
insert into name (co1) values (1000)
You Inserted Into >> name
1
Execution time in ms = 100
Enter Your Query TO execute it  or write (end) to exit :
select * from name
selected values:
=============================================
1000    null
=============================================
Execution time in ms = 13
Enter Your Query TO execute it  or write (end) to exit :
end
Ending Program...
Closing Statement.... Statement Closed
Closing Connection...... Connection closed
..JDBC is Closed
******************** see you soon ************************

D:\JDBC>pause
Press any key to continue . . .
```



```
D:\JDBC>java -jar JDBC.jar
********************** Welcome To JDBS ***********************
Creating Driver...... Driver Created
Creating Connection...... Connection Created
Creating Statement...... Statement Created
Enter Your Query TO execute it  or write (end) to exit :
select * from name
selected values:
==============================================
1000    null
==============================================
Execution time in ms = 139
Enter Your Query TO execute it  or write (end) to exit :
insert into name (co1 ,co2 ) values (10000 , 'oop')
You Inserted Into >> name
Execution time in ms = 114
Enter Your Query TO execute it  or write (end) to exit :
select * from name
selected values:
==============================================
1000    null
==============================================
10000   oop
==============================================
Execution time in ms = 14
Enter Your Query TO execute it  or write (end) to exit :
end
Ending Program...
Closing Statement.... Statement Closed
Closing Connection...... Connection closed
..JDBC is Closed
********************* see you soon ************************

D:\JDBC>pause
```



```
D:\JDBC>java -jar JDBC.jar
********************** Welcome To JDBS ***********************
Creating Driver...... Driver Created
Creating Connection...... Connection Created
Creating Statement...... Statement Created
Enter Your Query TO execute it  or write (end) to exit :
drop table name
Table> name << is Dropped
Execution time in ms = 2
Enter Your Query TO execute it  or write (end) to exit :
drop database test
DataBase>> test << is Dropped and Deleted
Execution time in ms = 3
Enter Your Query TO execute it  or write (end) to exit :
end
Ending Program...
Closing Statement.... Statement Closed
Closing Connection...... Connection closed
..JDBC is Closed
********************* see you soon ************************

D:\JDBC>pause
Press any key to continue . . .
```