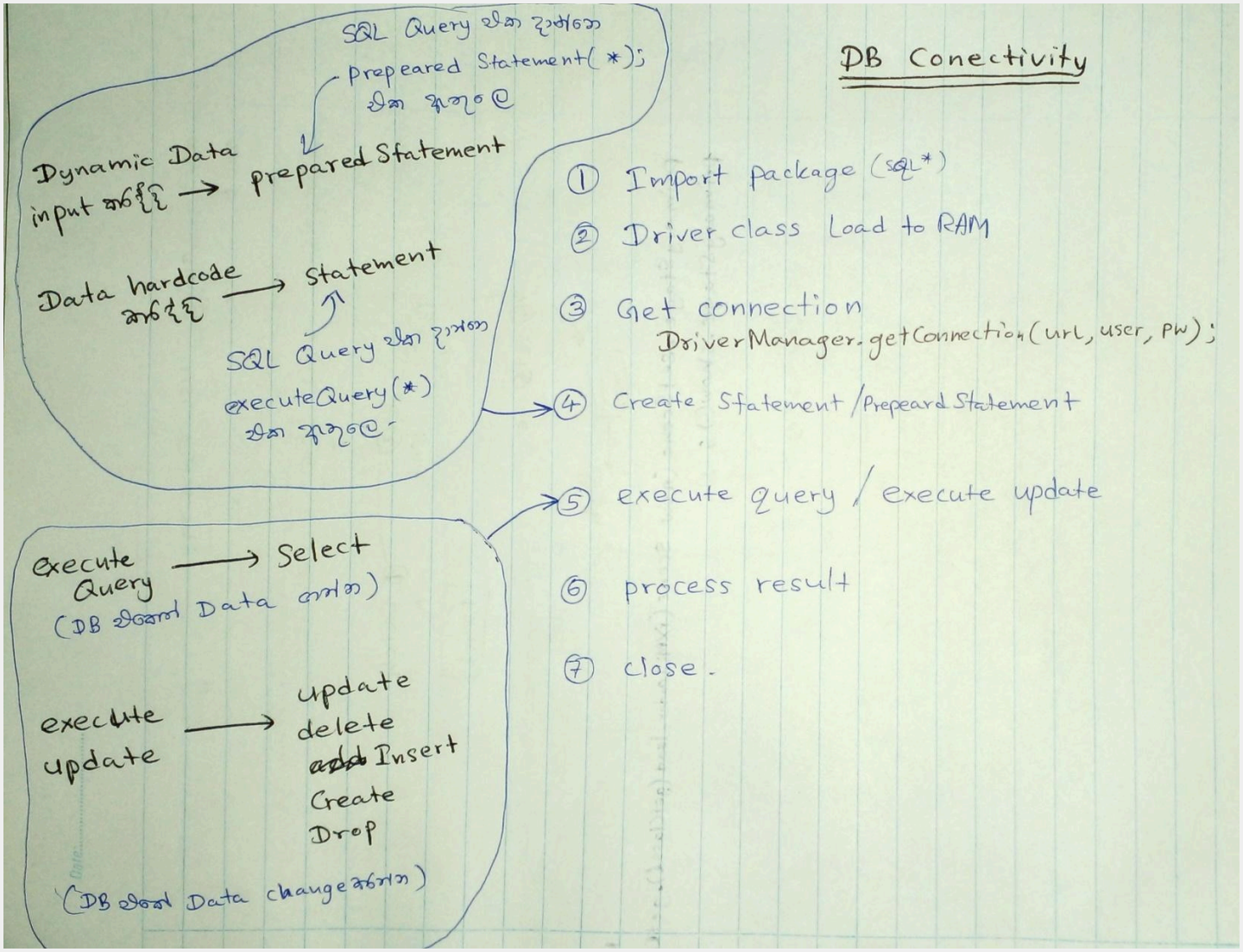
🏠_Home_

🔗mysql-connector-java-8.0.11.jar



Select - hard coded

```
public class Select {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/testDB", "root", "1234");

        Statement st = con.createStatement();
        ResultSet rs = st.executeQuery("select * from Student;");

        while (rs.next()){
            System.out.println(rs.getString(1)+"\t"+rs.getString(2));
        }
    }
}
```

Select - dynamic

```java
public class SelectDynamic {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        System.out.print("Enter ID : ");
        String id = new Scanner(System.in).next();

        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/testDB", "root", "1234");

        PreparedStatement pst = con.prepareStatement( "select * from Student where id = ? ");
        pst.setString(1,id);
        ResultSet rs = pst.executeQuery();

        while (rs.next()){
            System.out.println(rs.getString(1)+"  "+rs.getString(2));
        }
    }
}
```

Insert - hard coded *(similar UPDATE, DELETE, CREATE, DROP)*

```java

public class Insert {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/testDB", "root", "1234");

        Statement st = con.createStatement();
        int executeUpdate = st.executeUpdate("insert into Student Values('s01','Amal');");

        if(executeUpdate>0){
            System.out.println("Inserted !");
        }
    }
}
```

Insert - dynamic *(similar UPDATE, DELETE, CREATE, DROP)*

```java
public class InsertDynamic {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        System.out.print("Enter Name : ");
        String name = new Scanner(System.in).next();

        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/testDB", "root", "1234");

        PreparedStatement pst = con.prepareStatement( "insert into Student(Name) Values(?);");
        pst.setString(1,name);
        int executeUpdate = pst.executeUpdate();

        if(executeUpdate>0){
            System.out.println("data added !");
        }
    }
}
```

Singleton Design Pattern

- **Class** එකෙ නමින් **private static object** එකක් හදනෙ (Object එක static නිසා 1යි හැදෙන්නෙ.)
- **Constructor** එක **private** කරනෙ
- Object එකෙ address එක return කරන්න **static getter** එක හදනෙ

```java
class  SingleObject{
    private static SingleObject instance = new SingleObject();
    //create an object of SingleObject (static)

    private SingleObject(){}
    //make the constructor private so that this class cannot be instantiated

    public static SingleObject getInstance(){
        return instance;
    }
}

public class SingletonPatternDemo {
    public static void main(String[] args) {
        //SingleObject object = new SingleObject();
        // cannot create objects because constructor is private

        SingleObject object1 = SingleObject.getInstance();
        System.out.println(object1); //SingleObject@74a14482

        SingleObject object2 = SingleObject.getInstance();
        System.out.println(object2); //SingleObject@74a14482

        // object1 and object2 are same objects
    }
}
```

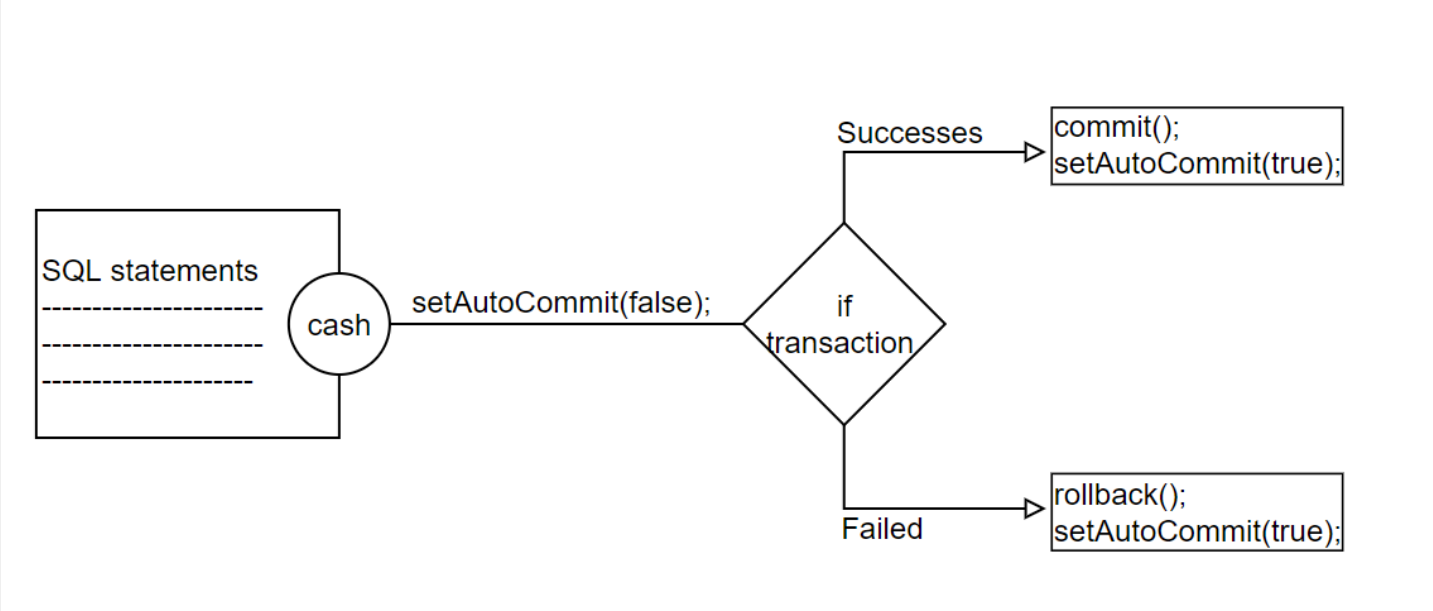
Singleton implementation for DB Connectivity

```java
import java.sql.*;

public class Select {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {

        Connection con = DBConnection.getDBConnection().getConnection();

        Statement st = con.createStatement();
        ResultSet rs = st.executeQuery("select * from Student;");

        while (rs.next()){
            System.out.println(rs.getString(1)+"\t"+rs.getString(2));
        }
    }
}


public class DBConnection{
    private static DBConnection dbConnection;
    private Connection connection;

    private DBConnection() throws ClassNotFoundException, SQLException {
```

```
        Class.forName("com.mysql.cj.jdbc.Driver");
        connection = DriverManager.getConnection("jdbc:mysql://localhost/testDB?useSSL=false", "root", "1234");
    }

    public Connection getConnection() {
        return connection;
    }

    public static DBConnection getDBConnection(){
      if(dbConnection==null){
          dbConnection = new DBConnection();
      }
      return dbConnection;
    }

}
```

DB Transaction

table 2ක් හෝ් කිහිපයක data එක පාර change කිරීම



- setAutoCommit(false); - Query එකක් read කල පසු auto commit වීම වලක්වයි.
- commit(); - Transaction එක සිදුකරයි.
- rollback(); - Transaction එක cancel කරයි.

```
public class DBTransactionDemo {
    public static void main(String[] args) throws SQLException {
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/thogakade?useSSL=false", "root", "1234");

        con.setAutoCommit(false);

        // SQL statements
        Statement st = con.createStatement();
        st.executeUpdate("INSERT INTO customer VALUES ('c78','Smith','USA','30000')");
        st.executeUpdate("INSERT INTO customer VALUES ('c79','Smith','USA','30000')");
        st.executeUpdate("INSERT INTO customer VALUES ('c80','Smith','USA','30000')");
```

```
        /*st.executeUpdate("delete from customer where id = 'c78';");
        st.executeUpdate("delete from customer where id = 'c79';");
        st.executeUpdate("delete from customer where id = 'c80';");*/
        ResultSet rs = st.executeQuery("select * from customer");
        while (rs.next()){
            System.out.println(rs.getString(1)+"  "+rs.getString(2)+"  "+rs.getString(3)+"  "+rs.getString(4));
        }

        System.out.print( "Save results ?" );
        String ans = new Scanner(System.in).next();

        if(ans.equals("yes")){
            con.commit();
            System.out.println("committed");
        }else{
            con.rollback();
            System.out.println("roll backed");
        }
        con.setAutoCommit(true);
    }
}
```