Handling Date values for Database operations
====================================
=> Sometimes as the part of programing requirement,we have to insert and retrieve Date like
   DOB,DOJ,DOM,DOP...wrt database.
=> It is not recommended to maintain date values in the form of String,b'z comparisons will become
     difficult.

In Java we have two Date classes
 1. java.util.Date
 2. java.sql.Date

=> java.sql.Date is the child class of java.util.Date.

=> java.sql.Date is specially designed class for handling Date values wrt database.
     Otherthan database operations,if we want to represent Date in our java program then we should
     go for java.util.Date.
=> java.util.Date can represent both Date and Time where as java.sql.Date represents only Date but
     not time.

```
1) class Test
2) {
3)    public static void main(String[] args)
4)    {
5)          java.util.Date udate=new java.util.Date();
6)          System.out.println("util Date:"+udate);
7)          long l =udate.getTime();
8)          java.sql.Date sdate= new java.sql.Date(l);
9)          System.out.println("sql Date:"+sdate);
10)    }
11) }
```

```
util Date:Mon Mar 20 19:07:29 IST 2017
sql Date:2017-03-20
```

Differences between java.util.Date and java.sql.Date
    java.util.Date
1) It is general Utility Class to handle Dates in our Java Program.
2) It represents both Data and Time.

   java.sql.Date
1) It is specially designed Class to handle Datesw.r.t DB Operations.
2) It represents only Date but not Time.

Note: In sql package Time class is availble to represent Time values and TimeStamp class is
                available to represent both Date and Time.

-> Inserting Date Values into Database:
        Various databases follow various styles to represent Date.
   Eg:
    Oracle: dd-MMM-yy eg: 28-May-90
    MySQL : yyyy-mm-dd eg: 1990-05-28

=> If we use simple Statement object to insert Date values then we should provide Date value in   the database supported format,which is difficult to the

programmer.
=> If we use PreparedStatement,then we are not required to worry about database supported form,
   just we have to call
       pst.setDate (2, java.sql.Date);
This method internally converts date value into the database supported format.
Hence it is highly recommended to use PreparedStatement to insert Date values into database.

Steps to insert Date value into Database:
 => DB: create table users(name varchar2(10),dob date);
1. Read Date from the end user(in String form)
       System.out.println("Enter DOP(dd-mm-yyyy):");
       String dop=sc.next();

2. Convert date from String form to java.util.Date form by using SimpleDateFormat object.
       SDF sdf= new SDF("dd-MM-yyyy");
       java.util.Date udate=sdf.parse(dop);

3. convert date from java.util.Date to java.sql.Date
       long l = udate.getTime();
       java.sql.Date sdate=new java.sql.Date(l);

4. set sdate to query
       pst.setDate(2,sdate);

5. int rowAffected= pst.executeUpdate();//Execute the query.

***Note:
If end user provides Date in the form of "yyyy-MM-dd" then we can convert directly that String into
 java.sql.Date form as follows...
 eg:: String s = "1980-05-27";
         java.sql.Date sdate=java.sql.Date.valueOf(s);

Retrieving Date value from the database
=======================================
=> For this we can use either simple Statement or PreparedStatement.
=> The retrieved Date values are Stored in ResultSet in the form of "java.sql.Date" and we can get
      this value by using getDate() method.
=> Once we got java.sql.Date object,we can format into our required form by using
      SimpleDateFormat object.

Sequence
========
1. Database
       (java.sql.Date)sqldate = rs.getDate(2);
2. Our required String Form
       String s = sdf.format(sqldate);
3. String s holds the date.

Need of DTO in projects
=======================
 DTO -> It stands for Data Transfer Object.
 This object is used for transferring the data from one layer to another layer in realtime applications.