COEN280-Database Systems
Fall 2014
Homework Assignment 3
Due: Tuesday, 11/18/2014

Project description

The goal of this assignment is to design an application that queries a spatial database. In this assignment you will become familiar with spatial data types using Oracle10*g*, Oracle Spatial features, and Java (JDBC). You are required to write two Java programs to:

1) Populate the spatial database.
2) Query the spatial database.

Scenario:

COEN 280 class is given the task to develop a system that helps organizing the geo-tagged photos of SCU main campus. A team of photographers were hired to take geo-tagged photos from SCU main campus. Each photographer is facilitated with a smartphone and a camera with built-in Wi-Fi and GPS. Each photo has EXIF meta information that contains GPS data. The photo along with EXIF meta data will be sent to a centralized server using SCU wireless. Also location of each photographer is tracked by sending GPS data from the smartphone to the server periodically.
Your task is to design a spatial database to keep the photos and locations of photographers. A User Interface should be developed as well to help the system administrator to issue some specific spatial queries to the database.

Input Files:

You will be given three files:

1. Image file: MAP - an 820 × 580 JPEG file that is an image of some area of SCU.
2. Three input files:
   a) building.xy
      Each building is represented by a 2D polygon.
      Col 1: building ID. Col 2: building name. Col3: number of vertices on the polygon.
      The numbers after column 3 are the coordinates of the vertices. They are ordered as x_1, y_1, x_2, y_2, ….., x_n, y_n

For example, a row:

b1, PHA, 4, 100, 120, 150, 130, 120, 200, 120, 220

represents a building with its building ID as "b1" and its name as "PHA". It has 4 vertices whose coordinates are (100, 120), (150, 130), (120, 200) and (120, 220) respectively.

b) photo.xy

Col 1: photo ID, Col 2: Photographer ID, Col 3: x coordinate of the photo. Col 4: y coordinate of the photo.

For simplicity, you can assume the locations of all photos are outside of all building polygons.

c) photographer.xy

Col 1: photographerID. Col2: x coordinate of the photographer's location. Col3: coordinate of the photographer's location.

Required .sql files:

You are required to create two .sql files:
1. createdb.sql: This file should create all required tables. In addition, it should include constraints, indexes, and any other DDL statements you might need for your application.
2. dropdb.sql: This file should drop all tables and the other objects once created by your createdb.sql file.

Required Java Programs:

You are required to implement two Java programs:
1. populate.java: This program should take the names of the input files as command line parameters and populate them into your database. It should be executed as:
"> java populate building.xy photo.xy photographer.xy".

Note that every time you run this program, it should remove the previous data in your tables, otherwise the tables will have redundant data.

2. Hw3.java: This program should provide a GUI, similar to Figure 1, to query your database. The GUI should include:

a) An 820 × 580 panel that shows the map when the application is started up.
b) The tile of the main window should display your full name and your student ID.
c) Two text fields that show the coordinates (x, y) of the current mouse location as it moves over the image. *Please notice that the coordinates*

given in .xy files are based on the origin (0, 0) at the upper left corner of the image and (820, 580) at its lower right corner.

d) 3 Check boxes that specify the feature types that we are currently interested in. Multiple feature types can be checked at the same time. They are called active feature types.
e) 5 Radio buttons that specify the kind of query we are going to do. There are 5 kinds of queries: Whole Region, Range Query, Point Query, Find Photos and Find Photographers. Details are given later. Only one radio button can be checked at any moment.
f) One button to submit the required query.
g) One text field to display the SQL statements for the queries that have been submitted so far. Use incremental counter for the queries, and print the counter along with the SQL statement (e.g., "Query 1: select * from photos;", "Query 2: select * from buildings where …").

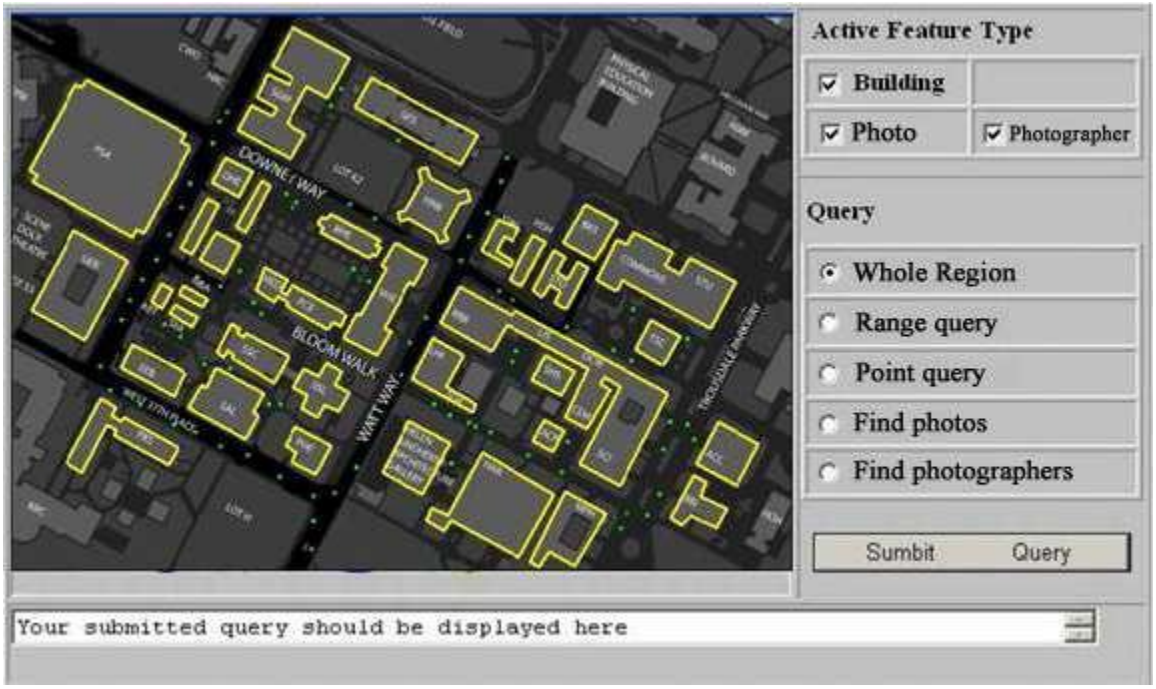

Figure 1

Queries:

1. Whole Region
   This is to display all the features of the active feature types in the whole map. They should be displayed according to the following standard:

Feature	Color	Shape
Buildings	Yellow	Polygon (outline, not solid region)
Photos	Blue	Circle (radius 3 pixels)
Photographers	Green	Square (5x5 pixels)

Table 1

2. Range Query
   When this radio button is checked, the user can draw a polygon in the map. After pushing the Submit Query button, only the features of the active feature types that are inside (or intersect with) the polygon area displayed. These features should also be displayed according to Table 1. The user draws the polygon by clicking the left mouse button to select its vertices sequentially and then clicking the right mouse button to close the polygon. The vertices should be connected by red line segments on the screen as they are being selected. When the Range Query radio button is unchecked, the selected polygon should disappear.

3. Point Query
   When this radio button is checked, the user can select a point in the map. This point is displayed as a red square (5×5 pixels). You should also display a red circle centered at this point whose radius is 100 pixels. After pushing the Submit Query button, only the active features that are inside (or intersect with) this circle are displayed. Their shapes are specified in Table 1. Their colors are as follows: for each active feature type, the feature that is nearest to the selected point among all the features of this feature type and within the red circle is displayed in yellow color. All the other features are displayed in green color. When the Point Query radio button is unchecked, the selected point and the associated red circle should disappear.

4. Find Photos
   When this radio button is checked, all the features of all feature types in the map should be displayed (according to Table 1). The user can select a photographer by clicking the left mouse button to select a point in the map. The photographer that is nearest to this point is selected and is displayed as a red square (5×5 pixels). Next, the user draw a polygon like the polygon of range query and after pushing the Submit Query button, all the photos inside the polygon which are

taken by selected photographer are shown. These photos should be displayed in red color.

5. Find Photographers
   When this radio button is checked, all the features of all feature types in the map should be displayed (according to Table 1). The user can select a building by clicking the left mouse button to select a point inside its polygon. The color of this building should be changed into red. After pushing the Submit Query button, all the photos that are taken close to this building should be displayed in red color. A photo is regarded to be taken close to a building if and only if the distance between the photo location and the building polygon is within 80 pixels. In addition, the photographers that can see the selected building should be displayed in red color. A building can be seen by a photograph if and only if the line connecting them does not intersect with any building.

Submission Guidelines

1. The links to the document for Oracle Spatial Reference, 3 input files, the campus image are provided on the course website.

2. Oracle JDBC Driver and Spatial Java APIs:
   Oracle Spatial Java Library (sdoapi.zip), is located in camino. It is required to manipulate spatial objects of Oracle10g in Java program. Oracle Spatial Java API document is also available on the course website.

   You can compile your source as:

   ```
   $ javac –classpath
   .:$ORACLE_HOME/jdbc/lib/classes12.zip:$ORACLE_HOME/md/lib/sdoapi.zip
   Hw3.java
   ```

   You run your application as:

   ```
   $ java –classpath
   .:$ORACLE_HOME/jdbc/lib/classes12.zip:$ORACLE_HOME/md/lib/sdoapi.zip
   Hw3
   ```

3. You need to have a readme.txt file that should include your name, student id, the list of the submitted files, and how to compile/run them. There is **25 points penalty** if this file or some of the required information is missing from your submission.

4. For the second Java program (i.e., Hw3.java), you may develop your assignment using more than one Java program. It is recommended (but not required) to separate the GUI codes and database related codes into different files.

5. You must make a .tar file to include all of your files in one file (<your_username>_hw3.tar)
Do NOT include the .class files, input files, or sdoapi.zip in your .tar file. We will compile your .java files.

6. You need to submit the assignment through camino. Do NOT try to submit any other file than hw3.tar.

7. You need to develop your databases on your design center oracle account but you can write your Java programs on any machine you wish. You can use any Java Visual Programs (e.g., JBuilder, Netbeans, Eclipse) you wish to design your GUI.

8. Start working on your assignment early.

11. Grading guideline:

Points		
5	Creating/Dropping database tables.	
10	Populating database	
20	GUI containing all of the requirements mentioned for user interfaces.	Hint: implement DB connectivity & queries first. If time left, move to GUI construction.

5	Whole Region
10	Range Query
10	Point Query
15	Find Photos
25	Find Photographers