CSCI 585- Database Systems
Spring 2011
Homework Assignment 2
Due: 4/7/2011 @ 11:59pm
Description

AnimTrack is an online animal tracking system which provides users with different functionalities and features needed for animal tracking. GPS-based animal tracking collars and devices have become more popular due to the fact that they provide continuous tracking capability. For our assignment's purpose we look into the vaccination process of these animals. From Database's point of view, this assignment will make you familiar with the design and implementation of a real-world application (of spatial database), using Oracle10*g*, Oracle Spatial features, and Java (JDBC).

You are required to write two Java programs to 1) store and 2) query your spatial database.

Input Files:

You will be given four files:

1. Image file: **land.jpg**, a 450x300 JPEG file that is a map of part of Los Angeles forests where animals live.
2. **animals.txt** which contains the information of all animals in the forest.
3. **trucks.txt**: which contains the locations of trucks that provide vaccines for animals.
4. **vaccines.txt**: which contains the information of vaccines.

You need to read three input files as follow:

a) Read the files line by line.
b) Each line represents a tuple, containing different columns separated by comma. (You can use StringTokenizer to extract the columns of each tuple.)
c) **(animals.txt)** has 3 columns in each line which are ID, name, and X & Y that represent the animal's location.

d) **(trucks.txt)** has 4 columns in each line which are ID, name, X & Y that represent the truck location, and the ID of the vaccines that are provided by that truck (separated by ;).
e) **(vaccines.txt)** has 3 columns in each line which are ID, name, and the (Name) of the animals that should receive that certain vaccine (separated by ;).

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 get the name of the input files as command line parameters and populate them into your database. It should be executed as: "> java populate animals.txt trucks.txt vaccines.txt".
   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. hw2.java: This program should provide a GUI, similar to the figure 1, to query your database. The GUI should include:
   a) A 450x300 (width x height) panel that shows the given map.
   b) A text field that shows the coordinates (X, Y) of the current mouse location as it moves over the image.
      Please notice that the coordinates given to you in .txt files are based on the origin (0,0) at the upper left corner of the image and (450,300) at its lower right corner.
   c) 2 Check boxes that specify the features we are interested in.
   d) 3 Radio buttons that specify the type of query we want to ask.

e) A text field to show the truck/animal information when clicked on a it in the map
f) A Submit button for submitting the query.
g) A Clear button to clear everything on the map for doing a new query.
h) A text field to show submitted SQL queries to the database.

Description of GUI :

The GUI should show the given map when the application is started up. The title of your main window should display your full name and your student ID.

a) Active Features: multiple features can be checked and those checked features will be retrieved and shown on the map. The following table shows how each point must be displayed.

Feature	Color	Shape
Animals	Red	Circle
Trucks	Yellow	Triangle

b) In order to draw a box as an active region for range query, "Range" radio button should first be selected, then the first click (left mouse button) specifies the lower left corner and the next click specifies the upper right corner of the box. The box must be displayed as follows:

Selected Box	Blue	Show only 2 opposite corners specified by the left mouse clicks with blue crosses and 4 edges connecting the corners with blue lines.

Figure 1 shows an example of a rectangle as an active region.

c) Whenever your program is submitting a query to the database, the SQL statement for that query should be printed in a text field on the GUI (the text field at the bottom of Figure 1).
If you need to send more than one query (e.g., when more than one feature is selected), use an incremental counter for the queries, and print the counter along

with the SQL statement (e.g., “Query 1: select * from restaurants where..;”, “Query 2: select * from photos where …”).

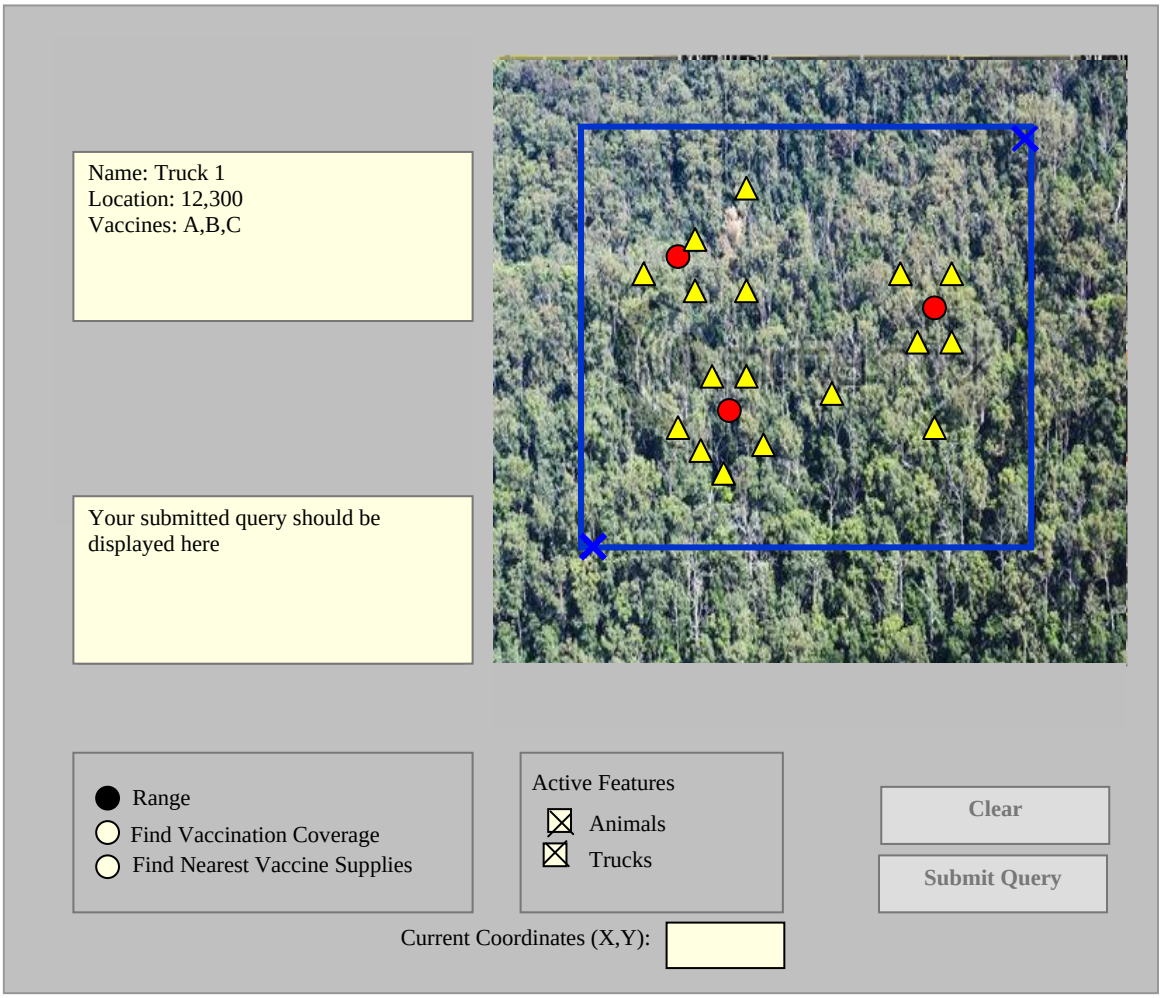

Figure 1 – Graphical User Interface of your application

Queries:

We will examine your homework as following:

1. ***For Range query:***

Assume we are interested in finding all the animals / trucks in a range. By selecting this radio button, the region selector gets activated. We then specify the corners of the box by clicking left mouse button on the map twice (your program should show a cross for each clicked corner), at this point your program should draw the box, then we select one or both of the features, and then click on the "Submit Query" button. Your program should submit separate queries for each feature. Each query returns all objects that are inside (or intersect with) the selected box. Your program should display these objects on the map. Once the objects are retrieved, clicking on each object should show its detailed information on the left side of the GUI. For trucks, name, xy location, and the name of the vaccines that the truck carries will be displayed in the left text box. For animals, the name and xy location should be displayed in the left text box.
Figure 1. Shows an example of range query asking for trucks, and animals inside the box.

2. Find Vaccination Coverage

Suppose we want to find which animals can be covered by the vaccine supply in a certain truck. Just as in the previous part, we select a range that contains at least one truck and in that range we choose a truck and we select the "Find vaccination coverage" radio button. Then, we press the "submit query" button. By submitting the query, all the animals in that region that can potentially receive any of the vaccines carried by this truck are displayed in a new color. Once the animals are retrieved, clicking on each animal should show its detailed information on the left side of the GUI.

3. ***Find Nearest Vaccine Supplies***

Suppose we want to vaccine one of the animals, we need a set of vaccine for this certain animal. For each of the vaccines, we want to find the closest truck that carries it. To do so, just as in the previous part, we select a range that contains at least one animal and in that range we choose an animal and we select the "Find Nearest Vaccine Supplies" radio button. Finally, we click on the "Submit Query" button. By submitting the query, all the closest trucks that carries the animal's vaccines are displayed in a new color. Once the trucks are retrieved, clicking on each truck should show its detailed information on the left side of the GUI.

Example: Suppose an animal needs vaccines A and B, where truck1 is the closest truck to the animal containing vaccine A, and truck2 is the closest truck containing vaccine B. Your program should show both trucks as the result. Note that if the closest truck to the animal happens to carry all the vaccines that the animal needs, your result should only contain that truck.

Submission Guidelines

1. The links to the document for Oracle Spatial Reference is provided in the website. The input files are provided in the blackboard website.

2. Oracle JDBC Driver and Spatial Java APIs:

Oracle Spatial Java Library (sdoapi.jar), is located in `$ORACLE_HOME/db/md/lib/` folder. It is required to manipulate spatial objects of Oracle10g in Java program. Oracle Spatial Java API document is also available on the class web page. You can compile your source as:

$ javac –classpath .:$ORACLE_HOME/db/jdbc/lib/classes12.jar:$ORACLE_HOME/db/md/lib/sdoapi.jar hw2.java

You can run your application as:

$ java –classpath .:$ORACLE_HOME/db/jdbc/lib/classes12.jar:$ORACLE_HOME/md/lib/sdoapi.jar hw2

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., hw2.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 (e.g., hw2.tar) using the following command:

> tar cvf hw2.tar *.java createdb.sql dropdb.sql readme.txt

You can do this in your Aludra account. 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 electronically using the EXACT following command FROM YOUR ALUDRA ACCOUNT. Do NOT try to submit any other file than hw2.tar.

Aludra.usc.edu> submit -user csci585 -tag hw2 hw2.tar

7. The submit command will immediately respond with a SUCCEEDED if your submission of file "hw2.tar" is successful. That will be your means to know that your homework has reached the right place. Your submissions will be time stamped, so we will know the exact time when you made the submission. Submit by the deadline.

8. You need to submit your assignment no later than **11:59 PM on 4/7/2011.**

9. Start working on your assignment early.

Distribution of points:

Points	Contents	Comments
10	Creating/Dropping database tables.	If either of them does not work properly, you will probably lose points for other parts associated as well.
15	Populating database.	
15	GUI containing all of the requirements mentioned for user interfaces.	Hint: implement DB connectivity & queries first. If time left, move to GUI construction.
20	Range Query	Database connectivity code will be counted here as well.
20	Find nearest vaccines supplies	
20	Find vaccination coverage	
100		