CIS 4301 // HW #8 // Spring 2020

Date Assigned: March 27, 2020
Date Due: April 3, 2020

Submission Format

You will submit a soft copy of your solution using e-Learning (http://elearning.ufl.edu) by the end of the day (23:59 / 11:59 PM) on the assigned date (March 22). Save your solution as a **java** file and name file **hw8** (hw8.java). Note, Canvas will be set to only receive an Java file. Test your solutions compiling and executing your Java code. Do not declare any of your classes to be *public* [note this will allow us to compile all classes submitted in the hw8.java file].

Assignment

At the top of every solution file you submit this semester include: your name, section number, the assignment number, and the date due. Complete the four exercises below, using the following relations.

TravelAgencyDB Schema
Booking(agent: String, traveler_ssn: integer, trip_id: integer)
FormedBy(id: integer, location: String, start_date: Date, end_date: Date)
GoesOn(ssn: integer, id: integer)
Leg(location: String, start_date: Date, end_date: Date)
Owns(ssn: integer, passport_number: integer, country: String)
Passport(number: integer, country: String, expiration_date: Date, owner: String)
TravelAgent(name: String, years_experience: integer, phone: String)
Traveller(name: String, ssn: integer, dob: Date)
Trip(id: integer, start_location: String, end_location: String, start_date: Date, end_date: Date, real: price)

Exercises

All operations will be performed on the TravelAgencyDB database.

1. Below is a stub for the class OurTravelAgency. Default and initialization constructors are provided for you. The method *menu* will be the entry point into the functionality you provide. You are allowed to implement other methods and classes as they are useful to your solution. The class TravelAgencyDriver is also given to provide an example of how your system will be tested.

2. After each of the query operations listed here, display [print] relevant results. When executing the user's currently built SQL statement, display all the tuples returned.

3. Provide the user with a menu system that facilitates the following operations. Note, operations / menu selection will continue until the user chooses to stop. Here is the menu you will implement, followed by the functionality included within that implementation.

Menu: provide a numeric menu for selection options. The user will input the letter associated with the desired option and this cycle will continue until they choose to end the program (c).

(a) Execute a complete SQL command
(b) View Data
(c) Logout / End Program

(a) Execute a complete SQL command: will receive a text string from the user with the desired SQL command and then execute it. Note, this would only be useful to users who know how to write SQL statements. Inexperienced users would leverage the other options.

(b) View Data: this command will provide the user with a list of all the tables in the database in a list fashion. The user will then be able to select one of the numeric options, and view contents from that table. Before displaying the contents of the selected table, display the set of the tables attributes, again allowing the user to enter a comma separated list of all the attributes desired [one or more, request by attribute name]. Finally, display all of the data in the table for the attributes selected.

(c) Close all connections and end the program.

OurTravelAgency and TravelAgencyDriver Stubs

```
class TravelAgencyDriver {

  public static void main( String args[] ) {
    OurTravelAgency ota = new OurTravelAgency();
    ota.menu();
  }
}

class OurTravelAgency {
  private String database;
  private String username;
  private String password;

  public OurTravelAgency() {
    database = "TravelAgencyDB";
    username = "root";
    password = "root";
  }
  public OurTravelAgency( String database, String username, String
password ) {
    this.database = database;
    this.username = username;
    this.password = password;
  }
  public void menu() {
    // implement your menu
  }

  // implement other methods as needed
}

// implement other classes as needed
```