Computing Science 291   
File & Database Management Project 1

Members:  
Caian Benedicto

Kevin De Asis

Professor:  
Yuan,Li Yan

LEC B1

T,R; 11:00AM - 12:20PM

Teaching Assitants:

Prouse,Jarett

LAB H04

R; 5:00PM - 6:20PM

Mariusdottir,Thorey

LAB H01

M; 12:30PM - 1:50PM

The project is an implementation of a database application system for an auto registration system based on the previous work in the first two assignments. This is a simple client-server application developed using Python. The implementation confirms to the auto registration system outlined in Assignment 1 and Assignment 2, as well as the SQL statements that was provided.

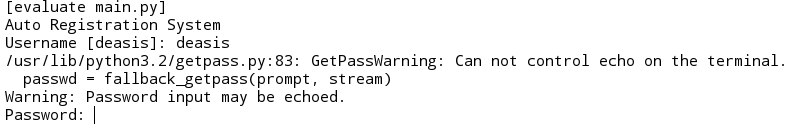
The architecture of the system we developed is a client-server model that has the Oracle database server as the back end and a Python application system as a client end. Python is the client tier that runs the application and the server that handles the database back end. The client establishes a connection with the server and obtains or manipulates data when the user has the right to do so. A graph of the system architecture can be seen below:

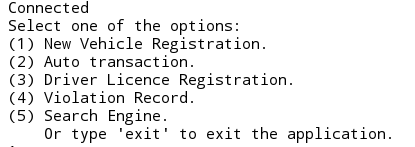
 

The system developed contains five application programs, which are New Vehicle Registration, Auto Transaction, Driver License Registration, Violation Record, and Search Engine. The program starts with a main menu for a user to access the five application programs, which can terminate if the users desires. The whole system uses a module named common.py that assists in data manipulation and user inputs for support in all of its major functions. The program guides the user with displayed information and notifying if any errors in the input, which will be shown in the following paragraphs.

The program begins by prompting the user for an Oracle username and password as shown below:



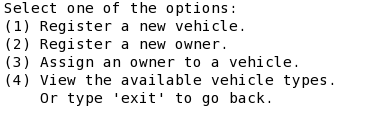
As soon as the user is connected to Oracle, the main menu is displayed in Python Shell that contains the option for the application programs. By entering the string ‘exit’ the client can exit the system. The main menu is shown below:



Note that under the hood process in all of the major function contains a while loop. It keeps repeating until the user enters an empty string that will exit the function or a correct statement that will continue until all conditions are satisfied to complete the function.

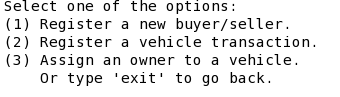
The first option, New Vehicle Registration component registers a new vehicle that is not registered in the database. In our system it is located in a module called vehicle.py, which contains the function new\_vehicle\_registration( ) that is called by the main program. It contains four options namely registration of a new vehicle, register new owner, assign a new owner, and view the available vehicle types. The program prompts the auto registration officer for information in each option in regards to the table associated with each query. The auto registration officer shall enter information about the vehicle and personal information about its new owners, if it is not in the database. It is assumed that vehicle types are stored initially in the database.

When the character “1” is entered in the main menu the auto registration officer has access to the following options in New Vehicle Registration as shown below and as described in the paragraph above:



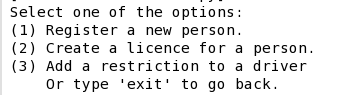
The second option, Auto Transaction completes an auto transaction by allowing the registering officer to enter all the necessary information to complete the task, which are but not limited to buyer and seller information, date, and price. Auto transaction is in a separate module called transaction.py, which contains the function auto\_transaction( ). The function checks if the seller is a current owner and switches the ownership to the buyer. Also, this component removes relevant information of the previous ownership. If a buyer is not in the database the registering officer is prompted to enter information to be registered in respect to the people table.

The auto transaction as stated in the previous paragraph can be accessed from the main menu by entering the character “2”, which shows the options to the registering officer similar to the image below:



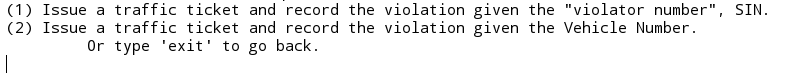
The third option, Drive License Registration component obtains the information needed to issue a driver’s license from the driver’s registering officer, which includes a picture from the driver. All files are assumed to be stored in a local disk system.Driver’s license registration is in the module named license.py that contains the function named driver\_licence\_registration(). It allows registration of a new person into the system by prompting the user. The user is to enter information with respect to the drive\_licence table to issue a driver’s license, which must have an SIN that exists in the database.

To access the third option the character “3” is entered in the main menu and the driver registering officer is directed to the drive license registration option and prompted with the following choices as shown below, which follows the objectives in the preceding paragraph:



The fourth component of the system, Violation Record component is used by a police officer to issue a traffic ticket and record the violation. The system uses the violation.py module to access the violation\_record() function. The information about ticket\_type is assumed to have been loaded in the initial database and the violation record component prompts the police officer with information regarding ticket table.

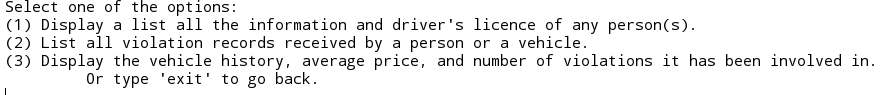
A police officer is able to access the violation record component by entering the character ‘4’ in the main menu. The violation record component can be seen below:



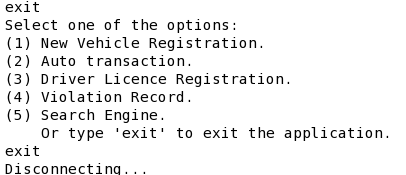
The final component is the search engine that is in a module named search.py that contains the function search\_engine(). It is divided into three parts.

The first part returns name, license number, address, driving class, driving conditions, and expiration date of a driver if an officer enters either the license number or a name. The second part returns all the violation records received by a person if either the driver’s license or SIN of a person is entered. Finally, a vehicle history is printed including the number of times a vehicle has been changed hand, the average price, and the number of violation it has been involved in by entering the vehicle’s serial number.

From the main menu the search component is accessed by entering ‘5’, which directs the officer to the options specified above and an image of the search component can be seen below:

****

The user is able to disconnect from Oracle by entering “exit” from the main menu, which disconnects from the server and exits the database application system, which can be seen below:



**Testing**

To check the proper function of the application the following test sequence was applied to cover the maximum number of test cases:

**Vehicle Registration:**

1. Insert the following vehicles (Serial number, Maker, Model, Year, Color, Type):
   1. a1, Maker 1, model a, 1999, Black, 0
   2. a2, maker 2, Model b, 2000, Red, 2
   3. a3, maker 1, Model a, 1999, Yellow, 0
   4. a4, Maker 3, Model A, 2005, Red, 1
   5. a1 – Error

As expected, the attempt to insert an existent serial number produced a warning to the user.

1. Add the following people to the database (SIM, Name, Height, Weight, Eye Color, Hair Color, Address, Gender, Birthday):
   1. 001, Carl Sagan, 1.78, 60, Brown, Black, Addr 1, m, 05/10/1976
   2. 002, Sarah, 1.67, 53, Green, blonde, Addr 2, f, 04/01/1987
   3. 003, Mc'kellen, 2, 90, Brown, Brown, Addr 3, m, 10/10/2000
   4. 004, saraH, 1.58, 55, Blue, Red, Addr 4, f, 05/20/1998
   5. 003 - Error
   6. 005, Test, 0, 0, 0, 0, 0, x - Error
   7. 005, Test, 0, 0, 0, 0, 0, m, 50/10/2000 - Error
   8. 005, Test, 0, 0, 0, 0, 0, m, 10/50/2000 - Error
   9. 005, Carl NotSagan, 1.98, 78.2222, Brown, Brown, Addr 5, m, 01/30/1998
   10. 006, saraH, 1.79, 60, Black, Black, Addr 6, f, 02/28/1999

As expected, the test cases e, f, g and h were identified as invalid and a warning was issued to the user.

1. Assign the following owners to vehicles (SIN, Vehicle Serial Number, Is Primary Owner):
   1. 003, a1, yes
   2. 001, a2, yes
   3. 004, a3, yes
   4. 004, a4, yes
   5. 005, a1, no
   6. 005, a1, yes - Error
   7. 005, a2, yes

Situation f was unsuccessful because the current person is already an owner of the vehicle.

**Auto Transaction:**

1. Register Transaction (Vehicle Serial Number, Seller, Buyer, Date, Price):
   1. a10 - Error
   2. a1, 004 - Error
   3. a1, 003, 003 - Error
   4. a1, 003, 002, 03/17/2014, 2000

The first three cases failed, respectively, because of an invalid serial number, for the seller SIN not belonging to an owner of that vehicle and for the seller and buyer being the same person.

1. Add Owner (SIN, Vehicle Serial Number, Is Primary Owner):
   1. 003, a1, yes
2. Register Transaction (Vehicle Serial Number, Seller, Buyer, Date, Price):
   1. a1, 003, 002, 03/18/2014, 1990
3. Add Owner (SIN, Vehicle Serial Number, Is Primary Owner):
   1. 003, a1, no
4. Register Transaction (Vehicle Serial Number, Seller, Buyer, Date, Price):
   1. a1, 002, 003, 03/18/2014, 1980

From the analysis of the database contents, the application managed to update the owners successfully at each transaction while keeping the database consistent in the case of operation failures.

**License Registration:**

1. License Creation (License Number, SIN, License Class, Picture File, Issuing Date, Expiration Date):
   1. l001, 007 - Error
   2. l001, 003, 1, ./images/ - Error
   3. l001, 003, 1, ./images/a.png, 05/20/1999, 05/20/2000
   4. l001, 003, 1, ./images/a.png, 05/20/1999, 05/20/2000
   5. l002, 005, 2, ./images/b.jpg, 05/22/1999, 05/22/2001
   6. l002 - Error
   7. l003, 005 - Error
   8. l003, 004, 3, ./images/c.png, 03/15/1998, 03/15/1999
   9. l004, 006, 2, /fulldirectory/images/c.png, 03/15/1998, 03/15/1999

Testing case ‘a’ fails due to inexistent SIN, case ‘b’ fails due to inexistent image file, case ‘f’ fails due to already existent license number and case ‘g’ fails due to provided SIN already been associated with a license. The database was verified to confirm that the images were uploaded successfully.

1. Adding Restriction (License Number, Restriction Description):
   1. l005 - Error
   2. l001, Restriction 1
   3. l001, Restriction 2
   4. l001, Restriction 3
   5. l002, Restriction 4

The only case where a warning is issued is case ‘a’ due to inexistent license number.

**Violation Records:**

1. Add Violation For SIN (Violator SIN, Officer SIN, Vehicle Serial Number, Ticket Type, Issuing Date, Details, Address):
   1. 009 - Error
   2. 001, 009 - Error
   3. 001, 009, a10 - Error
   4. 001, 009, a10, x - Error
   5. 001, 002, a1, parking , 03/19/2014, None, Addr 1
   6. 001, 002, a1, redlight, 03/19/2014, None, Addr 2

Test cases ‘a’ to ‘f’ failed due to, respectively, inexistent violator SIN, officer SIN, vehicle serial number and violation type.

1. Add Violation For Primary Owner (Vehicle Serial Number, Officer SIN, Ticket Type, Issuing Date, Details, Address):
   1. a1, 002, speed, 03/19/2014, None, Addr 2

The application successfully retrieved the SIN of the vehicle’s primary owner.

**Search Engine:**

1. Search for driver:
   1. l007 - No records
   2. noname - No records
   3. l001 - 1 record
   4. l002 - 1 record
   5. sarah - 2 records
2. Search violations:
   1. 001 - 2 records
   2. 003 - 1 record
   3. l001 - 1 record
   4. l009 - No records
3. Search vehicle:
   1. a1 - 3 violations / 3 transactions / avg. price 1990
   2. a2 - 0 violations / 0 transactions / avg. price N/A
   3. a3 - 0 violations / 0 transactions / avg. price N/A
   4. a10 - No records