CSC 455: Database Processing for Large-Scale Analytics
Assignment 3

Part 1

In this and the next part we will use an extended version of the schema from Assignment 2. You can find it in a file ZooDatabase.sql posted with this assignment on D2L.

Once again, it is up to you to write the SQL queries to answer the following questions:

1. List the animals (animal names) and the ID of the zoo keeper assigned to them.

Ans:

```
SELECT AName, ZookeepID
FROM Animal,Handles WHERE Animal.AID = Handles.AnimalID;
```

2. Now repeat the previous query and make sure that the animals without a handler also appear in the answer.

Ans:

```
SELECT Aname, ZookeepID
FROM Animal LEFT OUTER JOIN Handles ON Animal.AID = Handles.AnimalID;
```

3. Report, for every zoo keeper name, the total number of hours they spend feeding all animals in their care.

Ans:

```
SELECT ZName, SUM(TimeToFeed) as "Total Time"
FROM ZooKeeper JOIN Handles ON ZooKeeper.ZID = Handles.ZooKeepID
LEFT JOIN Animal ON Handles.AnimalID = Animal.AID
GROUP BY ZooKeeper.ZNAME;
```

4. Report every handling assignment (as a list of assignment date, zoo keeper name and animal name). Sort the result of the query by the assignment date in an ascending order.

Ans:

```
SELECT ASSIGNED, ZName, AName
FROM Handles JOIN ZooKeeper ON ZooKeeper.ZID = Handles.ZooKeepID
LEFT JOIN Animal ON Handles.AnimalID = Animal.AID
ORDER BY ASSIGNED;
```

5. Find the names of animals that have at least 1 zoo keeper assigned to them.

Ans:

```
SELECT AName FROM Animal
LEFT JOIN Handles on Handles.ANIMALID = Animal.AID
GROUP BY AName having count(Handles.ZOOKEEPID) >= 1;
```

6. Find the names of animals that have 0 or 1 (i.e., less than 2) zoo keepers assigned to them.

Ans:

```
SELECT AName FROM Animal
LEFT JOIN Handles on Handles.ANIMALID = Animal.AID
GROUP BY AName having count(Handles.ZOOKEEPID) <= 1;
```

- Optional query:

  List all combination of animals where the difference between feeding time requirement is within 0.25 hours (e.g., Grizzly bear, 3, Bengal tiger, 2.75). Hint: this will require a self-join. Avoid listing identical pairs such as (Grizzly bear, 3, Grizzly bear, 3)

  Ans:

```
SELECT a.AName, b.AName, ABS(a.TimeToFeed - b.TimeToFeed) AS DIFF
FROM Animal a, Animal b
WHERE ABS(a.TimeToFeed - b.TimeToFeed)<=0.25 AND a.AID!=b.AID;
```

Part 2

A. Write a python script that is going to read the queries that you have created in Part-1 from a SQL file, execute each SQL query against SQLite database and print the output of that query. You must read your SQL queries from a file, please do not copy SQL directly into python code. The code that would run commands from the ZooDatabase.sql file is provided (runSQL.py), so all you have to do is to change it so that it reads *your* queries from a SQL file and prints the output of your queries. You can refer to example code from the previous assignment that prints query results using fetchall(). You do not have to format the output in any particular fashion – however, you must print every row individually using a loop.

CODE:

```
import re
import sqlite3
from sqlite3 import OperationalError

conn = sqlite3.connect('csc455_HW3.db')
c = conn.cursor()

# Open and read the file as a single buffer
fd = open('ZooDatabase.sql', 'r')
```

```
# Read as a single document (not individual lines)
sqlFile = fd.read()
fd.close()

# all SQL commands (split on ';' which separates them)
sqlCommands = sqlFile.split(';')

# Execute every command from the input file (separated by ";")
for command in sqlCommands:
# This will skip and report errors
# For example, if the tables do not yet exist, this will skip over
# the DROP TABLE commands
try:
command = re.sub('[\n\t]',"",command) # this removes all the whitespaces
c.execute(command)
except OperationalError, msg:
print "Command skipped: ", msg
for command in sqlCommands:
output = c.execute(command).fetchall()
for row in output:
for value in row:
print value, "\t",
print "\n",
print "\n"
c.close()
conn.commit()
conn.close()
```

B. Repeat the work you did in Part-2 of the previous homework using the following data file: http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/Public_Chauffeurs_Short_hw3.csv
It contains roughly the same data, with two changes: NULL may now be represented by NULL or an empty string (,NULL, or ,,) and some of the names have the following form “Last, First” instead of “First Last”, which is problematic because when you split the string on a comma, you end up with too many values to insert.

CODE:

```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct 17 02:32:43 2014

@author: Akhilkumar
"""
import sqlite3

# Creating a table called Information.
Information_Table = '''CREATE TABLE Information(
License_Number NUMBER(15) ,
```

```
Renewed DATE,
Status VARCHAR2(15),
Status_Date DATE,
Driver_Type VARCHAR2(20),
License_Type VARCHAR2(20),
Original_Issue_Date DATE,
Name VARCHAR(30),
Sex CHAR(7),
Chauffer_City CHAR(20),
Record_Number VARCHAR2(15) NOT NULL,

CONSTRAINT Rec_NumPK
PRIMARY KEY(Record_Number)
CONSTRAINT Information_FK1
FOREIGN KEY (Chauffer_City)
REFERENCES CLocation(Chauffer_City)
);'''

# Creating a table called Location.
Location_Table = '''CREATE TABLE Location(
Chauffer_City VARCHAR2(30),
Chauffer_State VARCHAR2(15),

CONSTRAINT CLocPK
PRIMARY KEY (Chauffer_City)
);'''


# To open a connection to database
conn = sqlite3.connect("Assignment2.db")

#Request a cursor from the database
cursor = conn.cursor()

# Incase the tables is already present, then drop the tables.
cursor.execute("DROP TABLE IF EXISTS Information")
cursor.execute("DROP TABLE IF EXISTS Location")
# If there is no table with such name, then create Assignment2.db
cursor.execute(Information_Table)
cursor.execute(Location_Table)

# Let's open a file to populate the Information table in python
fd = open('Public_Chauffeurs_Short.csv', 'r');
# As the fisrt row contains the row names, we should drop that row
allLines = fd.readlines()[1:]
fd.close()
```

```
for line in allLines:

values = line.strip().split(',')

# Replacing the NULL values with python NULL
for i in range(len(values)):
if values[i] == 'NULL':
values[i] = None

# The first table should contain the first 10 and the 12 column
Inf = values[:10] + [values[11]]
# The second table should contain City & State column
City = values[9:11]

cursor.execute("INSERT INTO Information VALUES (?,?,?,?,?,?,?,?,?,?,?);", Inf)
cursor.execute("INSERT OR IGNORE INTO Location VALUES (?,?);", City)


# For each table, print the row contents
for table in ['Information', 'Location']:
allRows = cursor.execute("SELECT * FROM %s;" % table).fetchall()

rowCount = 0
# For every row, print the results of the query above, separated by a tab
for eachRow in allRows:
print "ROW#"+str(rowCount),
rowCount = rowCount+1
for value in eachRow:
print value, "\t",
print "\n", # \n is the end of line symbol

# Finalize inserts and close the connection to the database
conn.commit()
conn.close()
```

Part 3

Using the company.sql database (posted in with this assignment), write the following SQL queries.

1. Find the names of all employees who are directly supervised by 'Franklin T Wong'.

Ans:

```
SELECT FName, MINIT, LName
FROM Employee
```

```
WHERE Super_ssn =(SELECT Esn
FROM Employee
WHERE FName = 'Franklin' AND MINIT = 'T' AND LNAME = 'Wong');
```

2. For each project, list the project name, project number, and the total hours per week (by all employees) spent on that project.

Ans:

```
SELECT  Pname,Pnumber, SUM(Hours)
FROM project RIGHT OUTER JOIN works_on ON Works_on.PNo = Project.PNumber
GROUP BY Pname,Pnumber;
```

3. For each department, retrieve the department name and the average salary of all employees working in that department. Order the output by department number in ascending order.

Ans:

```
SELECT Dname,DNUMBER, AVG(salary)
FROM Department JOIN Employee ON Employee.DNO = Department.DNUMBER
GROUP BY Department.DNAME,Department.DNUMBER
ORDER BY Department.DNUMBER ASC;
```

4. Retrieve the average salary of all female employees.

Ans:

```
SELECT AVG(salary) FROM employee WHERE sex = 'F';
```

5. For each department whose average salary is greater than $43,000, retrieve the department name and the number of employees in that department.

Ans:

```
SELECT Dname, COUNT(ssn)
FROM Department JOIN Employee ON Department.Dnumber = Employee.Dno
GROUP BY Department.DNAME, Salary
HAVING AVG(Salary) > 43000;
```

6. Retrieve the names of employees whose salary is within $22,000 of the salary of the employee who is paid the most in the company (e.g., if the highest salary in the company is $82,000, retrieve the names of all employees that make at least $60,000.).

Ans:

```
SELECT Fname,Minit,LName
FROM Employee
WHERE salary >= (SELECT MAX(salary) FROM Employee) - 22000;
```