Part 1: Write the SQL queries from ZooDatabased.sql to answer questions:

/\*Part1-1\*/

select aname, zookeepid

from handles, animal

where aid=ANIMALID;

/\*Part1-2\*/

select aname, zookeepid

from animal left outer join handles

ON aid=ANIMALID;

/\*Part1-3\*/

select zname, SUM(timetofeed)

from zookeeper, handles, animal

where aid=animalid and zid=zookeepid

group by zname;

/\*Part1-4\*/

select Assigned, zname, aname

from zookeeper, handles, animal

where zid=zookeepid and aid=animalid

order by assigned ASC;

/\*Part1-5\*/

select aname

from animal, handles

where aid=animalid

group by aname

HAVING count(animalid) >=1;

/\*Part1-6\*/

select aname

from animal outer left join handles

ON aid=animalid

group by aname

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.

*#Provided*

**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**:

c.execute(command)

**except** OperationalError **as** msg:

print ("Command skipped: ", msg)

c.close()

conn.commit()

conn.close()

In []:

*#Part2-A: Python script to read queries from Part-1 from SQL file,*

*#Execute each SQL query against SQLite database and print the output of the query*

**import** **sqlite3**

**from** **sqlite3** **import** OperationalError

fd = open ('csc455\_wk4\_assign\_part2a.sql')

content=fd.read()

fd.close()

content=content.strip().split(';')

*#Part2-A: The for lopp to read query against the SQL db to output the result*

**for** i **in** range(len(content)-1):

data=c.execute(content[i]).fetchall()

print('Part2-',i+1)

**for** u **in** range(len(data)):

print(data[u])

print('**\t**')

\*\*\*The following is the result queries against the db based on the provided condition\*\*\*

Out []:

Part2- 1

('Galapagos Penguin', 1)

('Emperor Penguin', 1)

('Alpaca', 1)

('Sri Lankan sloth bear', 2)

('Grizzly bear', 2)

('Giant Panda bear', 2)

('Siberian tiger', 3)

('Bengal tiger', 3)

('South China tiger', 3)

('Alpaca', 3)

Part2- 2

('Galapagos Penguin', 1, 0.5)

('Emperor Penguin', 1, 0.75)

('Sri Lankan sloth bear', 2, 2.5)

('Grizzly bear', 2, 3)

('Giant Panda bear', 2, 1.5)

('Florida black bear', None, 1.75)

('Siberian tiger', 3, 3.5)

('Bengal tiger', 3, 2.75)

('South China tiger', 3, 2.25)

('Alpaca', 1, 0.25)

('Alpaca', 3, 0.25)

('Llama', None, 3.5)

Part2- 3

('Jim Carrey', 1.5)

('Rob Schneider', 8.75)

('Tina Fey', 7.0)

Part2- 4

('01-Jan-2000', 'Jim Carrey', 'Galapagos Penguin')

('01-Jan-2000', 'Jim Carrey', 'Alpaca')

('01-Jan-2000', 'Rob Schneider', 'Siberian tiger')

('02-Jan-2000', 'Jim Carrey', 'Emperor Penguin')

('02-Jan-2000', 'Tina Fey', 'Sri Lankan sloth bear')

('03-Jan-2000', 'Tina Fey', 'Giant Panda bear')

('03-Jan-2000', 'Rob Schneider', 'Bengal tiger')

('04-Jan-2000', 'Tina Fey', 'Grizzly bear')

('04-Jan-2000', 'Rob Schneider', 'Alpaca')

('05-Jan-2000', 'Rob Schneider', 'South China tiger')

Part2- 5

('Alpaca',)

('Bengal tiger',)

('Emperor Penguin',)

('Galapagos Penguin',)

('Giant Panda bear',)

('Grizzly bear',)

('Siberian tiger',)

('South China tiger',)

('Sri Lankan sloth bear',)

Part2- 6

('Bengal tiger',)

('Emperor Penguin',)

('Florida black bear',)

('Galapagos Penguin',)

('Giant Panda bear',)

('Grizzly bear',)

('Llama',)

('Siberian tiger',)

('South China tiger',)

('Sri Lankan sloth bear',)

*#Part 2-B: Create the table and use python to automate loading of the following file into SQLite:*

*#Tools: ipython Notebook*

*#Method: Use pandas library to read and load the table from the URL*

*#Assumption:*

*#Replace the nan by None which is nonetype*

*#Primary Key: License Number (No Schema is formatted since no Functional Dependency provided)*

*#INTEGER Type: License\_Number*

*#DATE Type: Status\_Date, Original Issue\_Date*

*#String Type: Renewed, Status,Driver\_Type, License\_Type, Name, Sex, Chauffeur\_City, Chauffeur\_State, Record\_Number*

*# Set up the Sqlite3*

**import** **sqlite3**

**from** **sqlite3** **import** OperationalError

*#Create hw3\_2b DataBase*

conn = sqlite3.connect('hw3\_2b.db')

c = conn.cursor()

*#Import pandas and numpy library*

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

*#Use pd.read\_csv to read the URL and save to content list*

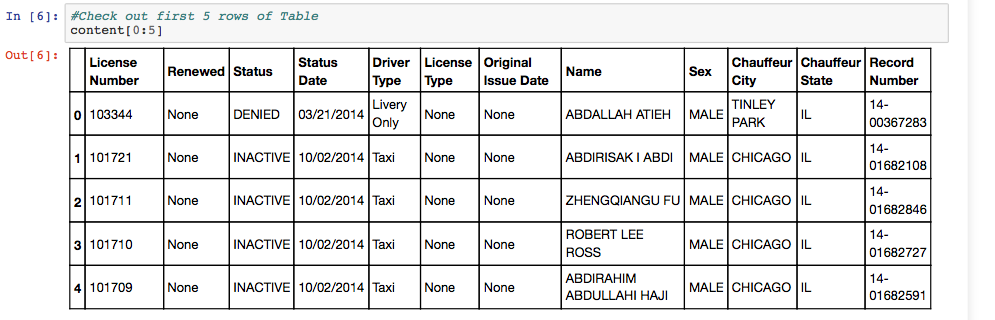
content=pd.read\_csv("http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/Public\_Chauffeurs\_Short\_hw3.csv"

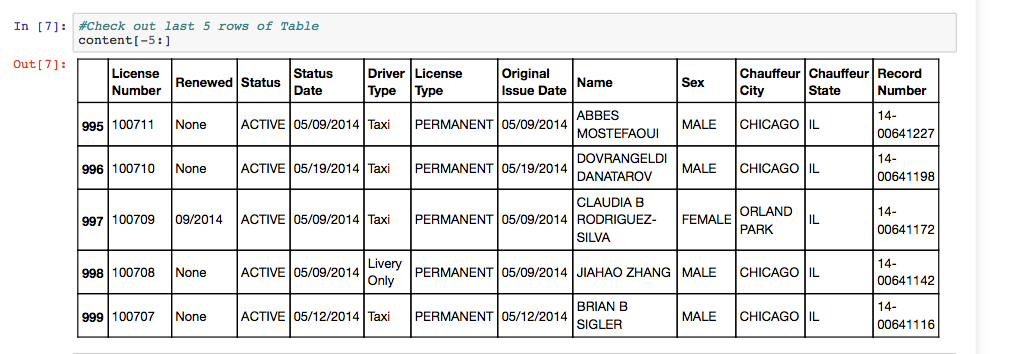
,index\_col=**False**, header=0)

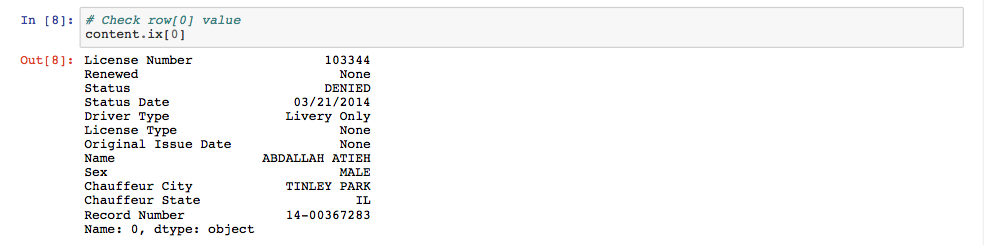
*#Replace all the pd nan by None value in order to insert to sQL*

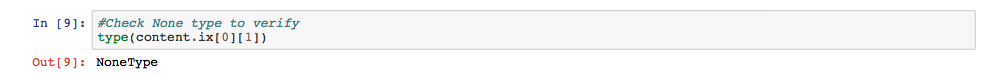
content = content.where((pd.notnull(content)), **None**)

\*\*\* The followings are screenshots checking the table that we read\*\*\*









*#Create Table ‘License’ Script*

Li = '''CREATE TABLE License

(

License\_Number INTEGER NOT NULL,

Renewed varchar(10),

Status varchar(8),

Status\_Date DATE,

Driver\_Type varchar(15),

License\_Type varchar(10),

Original\_Issue\_Date DATE,

Name varchar(25),

Sex varchar(50),

Chauffeur\_City varchar(50),

Chauffeur\_State varchar(50),

Record\_Number varchar(50)

);'''

*#Drop the Table ‘License’*

c.execute("DROP TABLE License")

*#Populate the Table 'License'*

c.execute(Li)

*#INSERT statement from the Table which is called ‘content’, Total 12 attributes*

**for** i **in** range(len(content)):

c.execute("INSERT INTO License VALUES (?,?,?,?,?,?,?,?,?,?,?,?)",

(int(content.ix[i][0]),

content.ix[i][1],

content.ix[i][2],

content.ix[i][3],

content.ix[i][4],

content.ix[i][5],

content.ix[i][6],

content.ix[i][7],

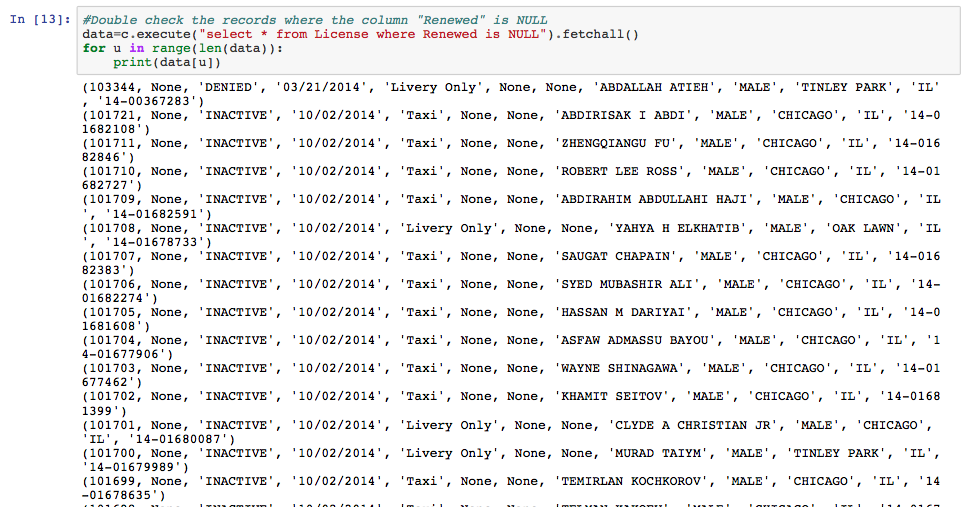
content.ix[i][8],

content.ix[i][9],

content.ix[i][10],

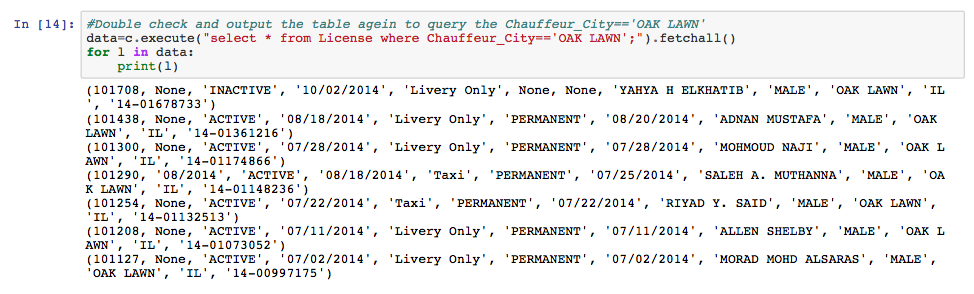
content.ix[i][11]))

\*\*\*The following screenshot is used for checking None (Nonetype) value from the table column 2 (Attribute= Renewed), the screenshot is not showing all the rows because the record is too long\*\*\*



\*\*\* The following screenshot is showing the Table with query against the sql where

*Chauffeur\_City=='OAK LAWN' (Column 10)*



Part 3: Using the company.sql database, write the SQL queries.

/\*Part3-1\*/

select e2.Fname,e2.MINIT, e2.lname, ' superised by ', e1.Fname,e1.MINIT,e1.lname

from employee e2, employee e1

where e2.super\_ssn = e1.ssn and e1.fname='Franklin' and e1.minit='T' and e1.lname='Wong';

/\*Part3-2\*/

select pname, pnumber, sum(hours)

from Project, Works\_on

where pno=pnumber

group by pname,pnumber;

/\*Part3-3\*/

select dname, AVG(salary)

from department, employee

where DNO = Dnumber

group by dnumber,dname

order by dnumber;

/\*Part3-4\*/

select AVG(Salary)

from employee

where SEX='F';

/\*Part3-5\*/

select dname, avg(salary),count(ssn) AS number\_of\_employees

from department, employee

where dnumber=dno

group by dname

having avg(salary)>42000;

/\*Part3-6\*/

select fname, minit,lname,(select max(salary) from employee)-salary AS diff

from employee

where (select max(salary) from employee)-salary<25000;