Assignment 5
Sarah Cummings

1) A. Create user table in SQLite, and modify Twitter table to include UserID which is a foreign key.

B. Write Python code to read in 7000 tweets, populate the table, and save the bad tweets to a file.

See attached file, assignment5_sarahcummings.py, the contents of which are copied below:

```
import json, re, sqlite3
import urllib.request as urllib

#Connect to database
conn=sqlite3.connect('csc455.db')
#Request a cursor from the database
c=conn.cursor()

#Create the user table
UserTable= '''CREATE TABLE User
(
id NUMBER(25),
name VARCHAT(50),
screen_name VARCHAR(60),
description VARCHAT(50),
friends_count NUMBER(100),
CONSTRAINT User
   Primary Key(id)
);'''

#Create twitter table
Twitter2table= '''CREATE TABLE Twitter2
(
created_at VARCHAR(50),
id_str NUMBER(50),
text VARCHAR(160),
source VARCHAR(100),
in_reply_to_user_id VARCHAR(25),
in_reply_to_screen_name VARCHAR(25),
in_reply_to_status_id VARCHAR(25),
retweet_count NUMBER(5),
contributors  VARCHAR(25),
user_id VARCHAR(20),

CONSTRAINT TwitterPK
Primary Key(id_str),
CONSTRAINT TwitterFK
Foreign Key(user_id)
REFERENCES User(id)
);'''

#drop tables if they exist
c.execute("DROP TABLE IF EXISTS User;")
c.execute("DROP TABLE IF EXISTS Twitter2;")

#make the tables from table strings above
c.execute(UserTable)
c.execute(Twitter2table)

#read in the file from the web
tweetFile = urllib.urlopen("http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/Assignment5.txt")

#create the file for the error tweets
f= open('error_tweets.txt','w')

lines=[]
for i in range(7000):
    #decode the tweets
    decodedTweets = tweetFile.readline().decode("utf8")
    #read just one line
    newLine=tweetFile.readline()
    #add that line to the list
    lines.append(newLine)
    #create twitter dictionary
    try:
        tDict= json.loads(decodedTweets)
        userInsertVals= (tDict['user']['id'],tDict['user']['name'],tDict['user']['screen_name'],tDict['user']['description'], tDict['user']['friends_count'])
        c.execute('INSERT OR IGNORE into User VALUES (?,?,?,?,?)', userInsertVals)
        twitterInsertVals= (tDict['created_at'],tDict['id_str'],tDict['text'],tDict['source'],tDict['in_reply_to_user_id'],tDict['in_reply_to_screen_name'],tDict['in_reply_to_status_id'],tDict['retweet_count'],tDict['contributors'],tDict['user']['id'])
        c.execute('INSERT INTO Twitter2 VALUES (?,?,?,?,?,?,?,?,?,?);', twitterInsertVals)
        # use this for a test: print('For tweet #',i,' the id is : ',tDict['id'], ' and tweet text is: ',tDict['text'])
    except(ValueError):
        string='For tweet #'+str(i)+' the Tweet is corrupted'
        f.write(string)
        f.close

conn.commit()
conn.close()
```

2. A) Write a SQL query to do the following: Find the user (“id” and “name”) with the highest “friend_count” in the database

```
SELECT id, name FROM User
WHERE friends_count = (SELECT MAX(friends_count) FROM User)
```

B) Write python code that is going to perform the same computation

See attached file, assignment5.3_sarahcummings.py , the contents of which are copied below:

```
import json
import urllib.request as urllib

#read in the file from the web
tweetFile = urllib.urlopen("http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/Assignment5.txt")
#create the file for the error tweets
f= open('error_tweets.txt','w')

friendCounts=[]
names=[]
userIDS=[]

lines=[]
for i in range(7000):
    #decode the tweets
    decodedTweets = tweetFile.readline().decode("utf8")
    #read just one line
    newLine=tweetFile.readline()
    #add that line to the list
    lines.append(newLine)
    #create twitter dictionary
    try:
        tDict= json.loads(decodedTweets)
        friendCounts.append(tDict['user']['friends_count'])
        names.append(tDict['user']['name'])
        userIDS.append(tDict['user']['id'])
    except(ValueError):
        pass

maxFriendCount= max(friendCounts)
maxIndex=friendCounts.index(maxFriendCount)
print('The info for the person with the most friends is as follows: \n',
'NAME:',names[maxIndex],'ID:',userIDS[maxIndex],'Friend Count:’,maxFriendCount)
```

3. Word frequency analysis of tweet text in python

See textFrequency.py file, the contents of which are copied below:

```
from operator import itemgetter
import urllib.request as urllib
import json

#read in the file from the web
tweetFile = urllib.urlopen("http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/Assignment5.txt")

#create the file for the error tweets
f= open('error_tweets.txt','w')

text= '.'
lines=[]
for i in range(700):
    #decode the tweets
    decodedTweets = tweetFile.readline().decode("utf8")
    #read just one line
    newLine=tweetFile.readline()
    #add that line to the list
    lines.append(newLine)
    #create twitter dictionary
    try:
        tDict= json.loads(decodedTweets)
        text= text + str(tDict['text'])

    except(ValueError):
        text= text

words = text.split(' ')
dCount = {}

for word in words:
    if word != '':
        if word not in dCount.keys():
            dCount[word] = 0
        dCount[word] = dCount[word]+1

countKeys = dCount.keys()
countVals = dCount.values()
countPairs = zip(countVals, countKeys)
```

```
# Sort the words by descending frequency
sorted_countPairs = sorted(countPairs, key=itemgetter(0), reverse=True)

#print the three most frequent words and their count
print (sorted_countPairs[0:3])
```

4. Extra Credit

A) [1pt] Write a SQL query that finds all animals without a zookeeper assignment using NOT EXISTS with a correlated nested sub-query.

```
SELECT AID, AName FROM Animal
WHERE AID NOT IN (SELECT ANIMALID FROM Handles);
```

B) [2.5pts] Write a trigger using PL/SQL in Oracle that will ensure that TimeToFeed defaults to at least 0.25 (i.e. if TimeToFeed is less than 0.25, reset it to the value of 0.25)

```
CREATE OR REPLACE TRIGGER TimeToFeed
AFTER INSERT ON Animal
FOR EACH ROW
WHEN (TimeToFeed <.025)
BEGIN
        DBMS_OUTPUT.PUT_LINE (‘TimeToFeed replaced with minimum’);
        INSERT INTO Animal.TimeToFeed
        VALUES (0.25);
END;
```

C) [2pts] Write a regular expression for identifying Social Security Numbers in the text.

```
SELECT * FROM Text WHERE REGEXP_LIKE(String, ‘^\d{3}\-\d{2}\-\d{4}\’);
```