Riley Payung

CDS 302-DL1

05/10/2020

Assignment 8

Imports

In [1]:

```
import sqlite3 as sql
import numpy as np
import pandas as pd
import random
```

In [2]:

```
# connect to the database
conn = sql.connect('socialNetwork.db');
```

In [3]:

```
# Set the table creation queries
querys = ['''
            CREATE TABLE IF NOT EXISTS User (
              u_id int NOT NULL,
              username varchar(25),
              birthyear int,
              primary key (u_id));
          ''',
          '''
            CREATE TABLE IF NOT EXISTS Follows (
              u_id INTEGER,
              follower_id INTEGER,
              foreign key (u_id) references User(u_id),
              foreign key (follower_id) references User(u_id));
          ''',
          '''
            CREATE TABLE IF NOT EXISTS Event (
              e_id INTEGER NOT NULL,
              topic varchar,
              day INTEGER,
              month INTEGER,
              primary key (e_id));
          ''',
          '''
            CREATE TABLE IF NOT EXISTS Participates (
              u_id INTEGER,
              e_id INTEGER,
              foreign key (u_id) references User(u_id),
              foreign key (e_id) references Event(e_id));
          ''']
```

In [4]:

```
# Execute Queries
for i in querys:
    conn.execute(i);
```

In [5]:

```
# Create usernames and birthyears for the users table
usernames = ['Jeoffery','Benjamin','Langston','Peters','Callin']
birthyear = [1990,2000,2002,1994,1989]

# Add users to the user table
for i in range(0,len(usernames)):
    query = 'INSERT INTO User VALUES (\''+ str(i+1) + '\',\'' + usernames[i] + '\',\'' + s
tr(birthyear[i]) + '\');';
    conn.execute(query);
```

In [6]:

```
# View the User table
pd.read_sql_query("SELECT * FROM User",conn)
```

Out[6]:

	u_id	username	birthyear
0	1	Jeoffery	1990
1	2	Benjamin	2000
2	3	Langston	2002
3	4	Peters	1994
4	5	Callin	1989

In [7]:

```
# Create topics to add to the Events table
topics = ['Game Show 2021','PAX East Gaming Festival','MagFest','Games 4 Change Festival',
'GameJam']
# Add Events to the events table.
for i in range(0,len(topics)):
    query = 'INSERT INTO Event VALUES (\''+ str(i+1) + '\',\'' + topics[i] + '\',\'' + str
(random.randint(1,31)) + '\',\'' + str(random.randint(1,13)) + '\');';
    conn.execute(query);
```

In [8]:

```
# View the Event table
pd.read_sql_query("SELECT * FROM Event",conn)
```

Out[8]:

	e_id	topic	day	month
0	1	Game Show 2021	8	9
1	2	PAX East Gaming Festival	25	6
2	3	MagFest	1	6
3	4	Games 4 Change Festival	10	9
4	5	GameJam	21	7

In [9]:

```
# Create 15 random event and user participations
for i in range(0,15):
    u = random.randint(1,5);
    e = random.randint(1,5);
    query = 'INSERT INTO Participates VALUES (\'' + str(u) + '\',\'' + str(e) + '\');';
    conn.execute(query);
```

In [10]:

```
# View the Participates table
pd.read_sql_query("SELECT * FROM Participates ORDER BY e_id",conn)
```

Out[10]:

	u_id	e_id
0	3	1
1	2	1
2	5	1
3	1	2
4	5	2
5	4	2
6	2	2
7	3	3
8	3	3
9	4	3
10	4	4
11	3	4
12	3	5
13	4	5
14	1	5

In [11]:

```
# Set the queries to change the users table.
querys = ['''
    ALTER TABLE User
    ADD uLat decimal;
    ''',
    '''
    ALTER TABLE User
    ADD uLon decimal;
    ''']
```

In [12]:

```
# Execute Queries
for i in querys:
    conn.execute(i);
```

In [13]:

```
# View the users table
pd.read_sql_query("Select * from User",conn)
```

Out[13]:

	u_id	username	birthyear	uLat	uLon
0	1	Jeoffery	1990	None	None
1	2	Benjamin	2000	None	None
2	3	Langston	2002	None	None
3	4	Peters	1994	None	None
4	5	Callin	1989	None	None

In [14]:

```
# Set the queries to change the events table
querys = ['''
    ALTER TABLE Event
    ADD eLat decimal;
    ''',
    '''
    ALTER TABLE Event
    ADD eLon decimal;
    ''']
```

In [15]:

```
# Execute Queries
for i in querys:
    conn.execute(i);
```

In [16]:

```
# View the Events Table
pd.read_sql_query("Select * from Event",conn)
```

Out[16]:

	e_id	topic	day	month	eLat	eLon
0	1	Game Show 2021	8	9	None	None
1	2	PAX East Gaming Festival	25	6	None	None
2	3	MagFest	1	6	None	None
3	4	Games 4 Change Festival	10	9	None	None
4	5	GameJam	21	7	None	None

In [17]:

```
# Update the Events Table to include the appropriate Lat and Lon
for i in range(0,5):
    eLat = random.uniform(20.0,50.0);
    eLon = random.uniform(-125.0,-70.0);
    query = '''
        UPDATE Event
        SET eLat = \''''+ str(eLat) + '''\'
        WHERE e_id = \'''' + str(i+1) + '\';';
    conn.execute(query);
    query = '''
        UPDATE Event
        SET eLon = \''''+ str(eLon) + '''\'
        WHERE e_id = \'''' + str(i+1) + '\';';
    conn.execute(query);
```

In [18]:

```
# View the Events Table
pd.read_sql_query("SELECT * FROM Event",conn);
```

In [19]:

```
# update the user table to include the appropriate lat and lon.
for i in range(0,5):
    uLat = random.uniform(20.0,50.0);
    uLon = random.uniform(-125.0,-70.0);
    query = '''
        UPDATE User
        SET uLat = \''''+ str(uLat) + '''\'
        WHERE u_id = \'''' + str(i+1) + '\';';
    conn.execute(query);
    query = '''
        UPDATE User
        SET uLon = \''''+ str(uLon) + '''\'
        WHERE u_id = \'''' + str(i+1) + '\';';
    conn.execute(query);
```

In [20]:

```
# View the users table
pd.read_sql_query("SELECT * FROM User",conn)
```

Out[20]:

	u_id	username	birthyear	uLat	uLon
0	1	Jeoffery	1990	23.010282	-91.777039
1	2	Benjamin	2000	26.805392	-70.891343
2	3	Langston	2002	47.393556	-115.116318
3	4	Peters	1994	46.399989	-120.934283
4	5	Callin	1989	24.808276	-82.608977

In [21]:

```
# Commit the changes up to this point
conn.commit();
```

In [22]:

```
# Create DF1:

df1 = pd.read_sql_query(
'''
    SELECT User.username as 'USER', User.uLat as 'USER Lat', User.uLon as 'USER Lon', Even
t.e_id as 'EVENT ID', Event.eLat as 'EVENT Lat', Event.eLon as 'EVENT Lon'
    FROM User, Event, Participates
    WHERE User.u_id = Participates.u_id AND Event.e_id = Participates.e_id
    ORDER BY Event.e_id

''',conn);
df1
```

Out[22]:

	USER	USER Lat	USER Lon	EVENT ID	EVENT Lat	EVENT Lon
0	Langston	47.393556	-115.116318	1	30.120631	-79.389768
1	Benjamin	26.805392	-70.891343	1	30.120631	-79.389768
2	Callin	24.808276	-82.608977	1	30.120631	-79.389768
3	Jeoffery	23.010282	-91.777039	2	34.241616	-107.254684
4	Callin	24.808276	-82.608977	2	34.241616	-107.254684
5	Peters	46.399989	-120.934283	2	34.241616	-107.254684
6	Benjamin	26.805392	-70.891343	2	34.241616	-107.254684
7	Langston	47.393556	-115.116318	3	41.116502	-80.517448
8	Langston	47.393556	-115.116318	3	41.116502	-80.517448
9	Peters	46.399989	-120.934283	3	41.116502	-80.517448
10	Peters	46.399989	-120.934283	4	38.572654	-96.740716
11	Langston	47.393556	-115.116318	4	38.572654	-96.740716
12	Langston	47.393556	-115.116318	5	25.542626	-84.609951
13	Peters	46.399989	-120.934283	5	25.542626	-84.609951
14	Jeoffery	23.010282	-91.777039	5	25.542626	-84.609951

In [23]:

```
# Create DF2:

df2 = pd.read_sql_query(
'''
    SELECT Event.e_id, COUNT(User.u_id) as 'Users Attended'
    FROM User NATURAL JOIN Event, Participates
    WHERE User.u_id = Participates.u_id AND Event.e_id = Participates.e_id
    GROUP BY Event.e_id

''',conn);
df2
```

Out[23]:

	e_id	Users Attended
0	1	3
1	2	4
2	3	3
3	4	2
4	5	3

In [24]:

```
# Create the harvesine distance function.
from math import sin, cos, sqrt, atan2, radians
def harvesine_dist(lat1, lon1, lat2, lon2):

    lat1 = radians(lat1);
    lat2 = radians(lat2);
    lon1 = radians(lon1);
    lon2 = radians(lon2);

    R = 6373.0 # approximate radius of earth in km

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

In [25]:

```
# calculate the distances and add a new data column to df1 (Distance (KM))
distance = [];
print(df1.dtypes)
for i in df1.index:
    distance.append(harvesine_dist(float(df1['EVENT Lat'][i]),float(df1['EVENT Lon'][i]),float(df1['USER Lat'][i]),float(df1['USER Lon'][i])));
df1['distance (KM)'] = distance;
df1
```

```
USER           object
USER Lat      float64
USER Lon      float64
EVENT ID        int64
EVENT Lat     float64
EVENT Lon     float64
dtype: object
```

Out[25]:

	USER	USER Lat	USER Lon	EVENT ID	EVENT Lat	EVENT Lon	distance (KM)
0	Langston	47.393556	-115.116318	1	30.120631	-79.389768	3599.318513
1	Benjamin	26.805392	-70.891343	1	30.120631	-79.389768	908.800871
2	Callin	24.808276	-82.608977	1	30.120631	-79.389768	670.791154
3	Jeoffery	23.010282	-91.777039	2	34.241616	-107.254684	1956.297896
4	Callin	24.808276	-82.608977	2	34.241616	-107.254684	2596.912631
5	Peters	46.399989	-120.934283	2	34.241616	-107.254684	1776.530823
6	Benjamin	26.805392	-70.891343	2	34.241616	-107.254684	3560.987973
7	Langston	47.393556	-115.116318	3	41.116502	-80.517448	2818.183473
8	Langston	47.393556	-115.116318	3	41.116502	-80.517448	2818.183473
9	Peters	46.399989	-120.934283	3	41.116502	-80.517448	3263.038992
10	Peters	46.399989	-120.934283	4	38.572654	-96.740716	2155.861295
11	Langston	47.393556	-115.116318	4	38.572654	-96.740716	1781.431636
12	Langston	47.393556	-115.116318	5	25.542626	-84.609951	3612.441235
13	Peters	46.399989	-120.934283	5	25.542626	-84.609951	3954.773612
14	Jeoffery	23.010282	-91.777039	5	25.542626	-84.609951	779.215705

In [26]:

```
# close the connection to the database
conn.close();
```

Question 1

```
The data is in main memory since the data has not been saved to the disk through th
e use of a csv or other file. It is also not in the actual database, just in a pand
as dataframe.
```

Question 2

```
No, you would not see it anymore since the terminal would be terminated for python,
and the memory would be flushed. If you were to save the data to the actual databas
e, you would be able to see it with a simple sql query.
```