CSE 511 Project 1 NoSQL Report

By: Kathy Kexin Tong

Reflection

I approached the project by first understanding what fields are contained in the dataset provided by looking through the output of each data entry in sample.db. I then read the project overview document thoroughly to understand what each function is looking to do, which is to filter the dataset based on criteria for either city or location/category, and output a subset of the fields into a new file in saveLocation.

Next, I followed the instructions provided for each function and coded them up, testing along the way to see if everything works properly by taking it outside of the function and giving example input fields. Here, I opened the saveLocation file and started a loop over each of the 25 items in the collection to search the city name in the full address. If returns true, will append the relevant fields into the saveLocation file. After all the items are looped, the file is closed and we complete the function. The first function based on city, my code is as below,

```
def FindBusinessBasedOnCity(cityToSearch,saveLocation1,collection):
    with open(saveLocation1, "w") as file:
        for line in collection:
            if cityToSearch in line['full_address']:
                file.write(str(line['name']) + "$" + str(line['full_address']) + "$" + str(line['city']) + "$" + str(line['state']))
                file.write('\n')
    file.close()
    pass
```

For the second query based on distance/category, it is more complicated as we need a helper function and have two filters instead of one. I first start the loop the same way of opening saveLocation file and calculate the two filters. If both filters return True, then will add the name to the saveLocation, and close. For the first filter of category, if any category in the item's categories list exist in our search categories list, then return True. For the distance filter, I wrote a helper to calculate distance, and if it is smaller than our maxDistance, then return True. The code is as below,

```
def FindBusinessBasedOnLocation(categoriesToSearch, myLocation, maxDistance, saveLocation2, collection):
    with open(saveLocation2, "w") as file:
        for line in collection:
            dist = DistanceFunction(myLocation[0], myLocation[1], line['latitude'], line['longitude'])
            if any(item in line['categories'] for item in categoriesToSearch) and dist < maxDistance:
                file.write(str(line['name']))
                file.write('\n')
    file.close()
    pass
```

For the helper function to calculate distance, I coded up the given algorithm provided in the overview document in python as below,

```
def DistanceFunction(lat2, lon2, lat1, lon1):
    r = 3959
    l1 = math.radians(lat1)
    l2 = math.radians(lat2)
    d1 = math.radians(abs(lat2-lat1))
    d2 = math.radians(abs(lon2-lon1))
    a = math.sin(d1/2) * math.sin(d1/2) * math.cos(l1) * math.cos(l2) * math.sin(d1/2) * math.sin(d2/2)
    # print(a, 1-a)
```

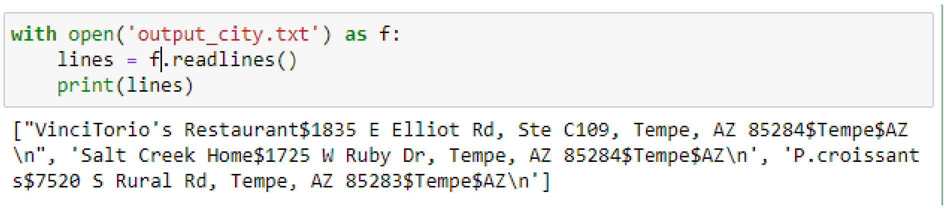
```
c = 2 *math.atan2(math.sqrt(a), math.sqrt(1-a))
d = r * c
return abs(d)
```

Lessons Learned

From this project, one lesson I learned is that working with NoSQL is not as straightforward as writing a standard SQL query. Here, we had to manipulate a lot of lists and dictionaries by looping through each item in the dataset, versus writing simple SELECT/FROM/WHERE clauses through an ODBC connection. I had to figure out how to loop through the items, how to pull in the results file, how to test the function and how to use python as the main interface for data query.

Output

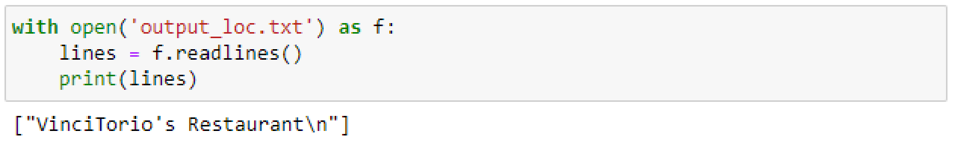
For the example query on the first function based on city and output:
FindBusinessBasedOnCity('Tempe', 'output_city.txt', data)



```
with open('output_city.txt') as f:
    lines = f.readlines()
    print(lines)
```

```
["VinciTorio's Restaurant$1835 E Elliot Rd, Ste C109, Tempe, AZ 85284$Tempe$AZ
\n", 'Salt Creek Home$1725 W Ruby Dr, Tempe, AZ 85284$Tempe$AZ\n', 'P.croissant
s$7520 S Rural Rd, Tempe, AZ 85283$Tempe$AZ\n']
```

For the example query based on categories/location filters and output:
FindBusinessBasedOnLocation(['Buffets'], [33.3482589, -111.9088346], 10, 'output_loc.txt', data)

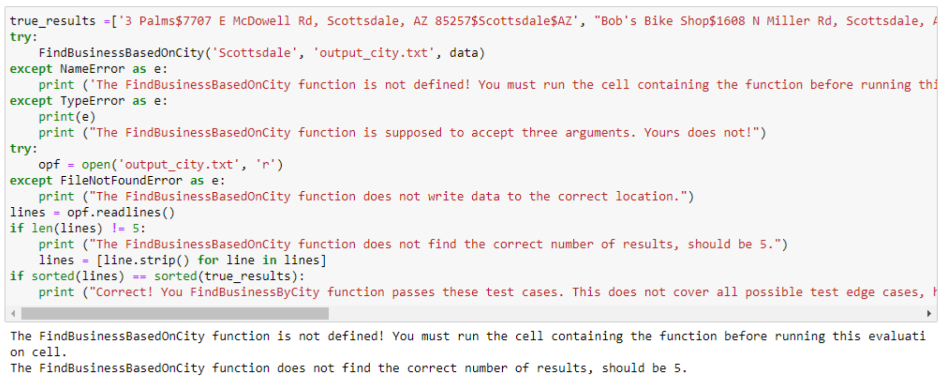


```
with open('output_loc.txt') as f:
    lines = f.readlines()
    print(lines)
```

```
["VinciTorio's Restaurant\n"]
```

Results

Query 1 Test case 1:



```
true_results =['3 Palms$7707 E McDowell Rd, Scottsdale, AZ 85257$Scottsdale$AZ', "Bob's Bike Shop$1608 N Miller Rd, Scottsdale, A
try:
    FindBusinessBasedOnCity('Scottsdale', 'output_city.txt', data)
except NameError as e:
    print ('The FindBusinessBasedOnCity function is not defined! You must run the cell containing the function before running thi
except TypeError as e:
    print(e)
    print ("The FindBusinessBasedOnCity function is supposed to accept three arguments. Yours does not!")
try:
    opf = open('output_city.txt', 'r')
except FileNotFoundError as e:
    print ("The FindBusinessBasedOnCity function does not write data to the correct location.")
lines = opf.readlines()
if len(lines) != 5:
    print ("The FindBusinessBasedOnCity function does not find the correct number of results, should be 5.")
    lines = [line.strip() for line in lines]
if sorted(lines) == sorted(true_results):
    print ("Correct! You FindBusinessByCity function passes these test cases. This does not cover all possible test edge cases, h
```

```
The FindBusinessBasedOnCity function is not defined! You must run the cell containing the function before running this evaluati
on cell.
The FindBusinessBasedOnCity function does not find the correct number of results, should be 5.
```

Query 1 Test case 2:

```python
true_results =['Arizona Exterminating Co.$521 E Broadway Rd, Mesa, AZ 85204$Mesa$AZ', 'Bikram Yoga$1940 W 8th St, Ste 111, Mesa,
try:
    FindBusinessBasedOnCity('Mesa', 'output_city.txt', data)
except NameError as e:
    print ('The FindBusinessBasedOnCity function is not defined! You must run the cell containing the function before running thi
except TypeError as e:
    print(e)
    print ("The FindBusinessBasedOnCity function is supposed to accept three arguments. Yours does not!")

try:
    opf = open('output_city.txt', 'r')
except FileNotFoundError as e:
    print ("The FindBusinessBasedOnCity function does not write data to the correct location.")

lines = opf.readlines()
if len(lines) != 7:
    print ("The FindBusinessBasedOnCity function does not find the correct number of results, should be 7.")
lines = [line.strip() for line in lines]
if sorted(lines) == sorted(true_results):
    print ("Correct! You FindBusinessByCity function passes these test cases. This does not cover all possible test edge cases, h
```

```
Correct! You FindBusinessByCity function passes these test cases. This does not cover all possible test edge cases, however, so
make sure that your function covers them before submitting!
```

Query 2 Test case 1:

```python
true_results =['Turf Direct']
try:
    FindBusinessBasedOnLocation(['Gardeners'], [33.3482589, -111.9088346], 20, 'output_loc.txt', data)
except NameError as e:
    print ('The FindBusinessBasedOnLocation function is not defined! You must run the cell containing the function before running
except TypeError as e:
    print ("The FindBusinessBasedOnLocation function is supposed to accept five arguments. Yours does not!")
try:
    opf = open('output_loc.txt','r')
except FileNotFoundError as e:
    print ("The FindBusinessBasedOnLocation function does not write data to the correct location.")

lines = opf.readlines()
if len(lines) != 1:
    print ("The FindBusinessBasedOnLocation function does not find the correct number of results, should be only 1.")
lines = [line.strip() for line in lines]

if sorted(lines) == sorted(true_results):
    print ("Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge cas
```

```
Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge cases, so mak
e sure your function does before submitting.
```

Query 2 Test case 2:

```python
true_results = ['Nothing Bundt Cakes', 'P.croissants']
try:
    FindBusinessBasedOnLocation(['Bakeries'], [33.3482589, -111.9088346], 15, 'output_loc.txt', data)
except NameError as e:
    print ('The FindBusinessBasedOnLocation function is not defined! You must run the cell containing the function before runni
except TypeError as e:
    print ("The FindBusinessBasedOnLocation function is supposed to accept five arguments. Yours does not!")

try:
    opf = open('output_loc.txt','r')
except FileNotFoundError as e:
    print ("The FindBusinessBasedOnLocation function does not write data to the correct location.")

lines = opf.readlines()
if len(lines) != 2:
    print ("The FindBusinessBasedOnLocation function does not find the correct number of results, should be only 2.")
lines = [line.strip() for line in lines]
if sorted(lines) == sorted(true_results):
    print ("Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge c
```

```
Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge cases, so mak
e sure your function does before submitting.
```

Query 2 Test Case 3:

```python
true_results = ['Nothing Bundt Cakes', 'Olive Creations', 'P.croissants', 'The Seafood Market']
try:
    FindBusinessBasedOnLocation(['Food', 'Specialty Food'], [33.3482589, -111.9088346], 30, 'output_loc.txt', data)
except NameError as e:
    print ('The FindBusinessBasedOnLocation function is not defined! You must run the cell containing the function before running
except TypeError as e:
    print ("The FindBusinessBasedOnLocation function is supposed to accept five arguments. Yours does not!")
try:
    opf = open('output_loc.txt','r')
except FileNotFoundError as e:
    print ("The FindBusinessBasedOnLocation function does not write data to the correct location.")

lines = opf.readlines()
if len(lines) != 4:
    print ("The FindBusinessBasedOnLocation function does not find the correct number of results, should be only 4.")
lines = [line.strip() for line in lines]
if sorted(lines) == sorted(true_results):
    print ("Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge cas
```

```
Correct! Your FindBusinessBasedOnLocation function passes these test cases. This does not cover all possible edge cases, so mak
e sure your function does before submitting.
```