CSE 511: Project 1 Report

Anurag Banerjee

Reflection:

In this assignment, we had to implement two functions to fetch data from a given NoSQL database collection based on provided criteria. The two functions were as follows:

1. First, we had to implement the **FindBusinessBasedOnCity** function in which we had been given a city name, a write location, and the list of all businesses as a collection from a NoSQL database. We had to find all the businesses in our collection that were in the provided city and write them to the provided write location in the given format of **Name$FullAddress$City$State**. I approached this by first fetching all the businesses from the given collection that had the city as the one that had been provided as input and storing them in a list named businessesByCity. Then, I traversed the list and for each business in this list, I wrote the details to the provided file name in the format Name$FullAddress$City$State. My function was as follows:

```
def FindBusinessBasedOnCity(cityToSearch, saveLocation1, collection):
    businessesByCity = filter(lambda business: business['city'] == cityToSearch, collection)

    with open(saveLocation1, 'w') as file:
        for business in businessesByCity:
            line = business['name'] + '$' + business['full_address'] + '$' + business['city'] + '$' + business['state']
            file.write(line)
            file.write('\n')
```

2. Next, we had to implement the **FindBusinessBasedOnLocation** function in which we were provided a list of categories to search, a location's co-ordinates, a max distance value, a write location, and the list of all businesses as a collection from a NoSQL database. We were also provided with a formula to calculate the distance between two points whose coordinates were known to us. We had to find all the businesses within the given maximum distance value and which belonged to any of the search categories that were provided as input, and write the name of the business to the provided location. I approached this by traversing through the list of businesses and in turn traversing through each of their categories. If any category of a business was present in the categories to search that was provided to us as input, I appended this business to a list named eligibleBusinesses and moved on to the next business. Then, I traversed through all the businesses present in the eligibleBusinesses and checked whether their distance was less than or equal to the maximum distance value that was provided to us as input. I checked the distance using the distance formula provided. I used a helper function named **DistanceFunction**. If the distance met the above cirteria I wrote the name of the business in the provided location. My function was as follows:

```
def FindBusinessBasedOnLocation(categoriesToSearch, myLocation, maxDistance, saveLocation2, collection):
    eligibleBusinesses = []
    for business in collection:
        businessCategories = business['categories']
        for category in categoriesToSearch:
            if category in businessCategories:
                eligibleBusinesses.append(business)
                break

    with open(saveLocation2, 'w') as file:
        for business in eligibleBusinesses:
            distance = DistanceFunction(myLocation[0], myLocation[1], business['latitude'], business['longitude'])
            if distance <= maxDistance:
                file.write(business['name'])
                file.write('\n')
```

For the helper function to calculate distance, I made use of the haversine formula (which was provided) to calculate the distance between two points on earth. The formula that I used is as follows:

Haversine formula:

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km);
note that angles need to be in radians to pass to trig functions!

I used this formula in my helper function named **DistanceFunction** to calculate the distance between two points as follows:

```
def DistanceFunction(lat2, lon2, lat1, lon1):
    R = 3959
    phi1 = radians(lat1)
    phi2 = radians(lat2)
    delta_phi = radians(lat2 - lat1)
    delta_lambda = radians(lon2 - lon1)
    a = (sin(delta_phi/2) * sin(delta_phi/2)) + (cos(phi1) * cos(phi2) * sin(delta_lambda/2) * sin(delta_lambda/2))
    c = 2 * atan2(sqrt(a), sqrt(1-a))
    d = R * c

    return d
```

Lessons Learned:

Some of the things that I learned while doing this project are as follows:

1. I learned how data is stored in NoSQL databases in the form of collections that have documents that are in the form of JSON. I got to know that this format is used when there is no strict need for structure/schema in our database.
2. I learned how to traverse through a dataset that is given to us in the form of a collection from a NoSQL database.
3. I learned different data manipulation techniques to filter the data according to our needs from the given collection. The criteria could be like city names or categories to which a business may belong to.
4. I learned how to write data to files in python.
5. I also got to know about the "**haversine**" formula which is used to calculate the distance between two points in a sphere. I learned how this formula could be used to calculate the distance between two points on the earth's surface whose latitude and longitude were given to us.

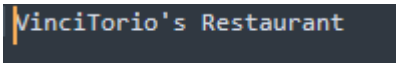
Output:

The two output files were as follows:

1. **output_city.txt:** This file stored the result of running the **FindBusinessBasedOnCity** function.

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

2. **Output_loc.txt:** This file stored the result of running the **FindBusinessBasedOnLocation** function.



Result:

The result of running the function for the given test cases were as follows:

1. **FindBusinessBasedOnCity** function result

```
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!
```

2. **FindBusinessBasedOnLocation** function result

```
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.
```