PROJECT 1 – NOSQL INDIVIDUAL REPORT

- NAVEEN KUMAR MANOKARAN STUDENT ID -1232482864

The Project was very useful and taught me a lot about how NoSQL Databases are used in restaurant applications like Uber Eats, Zomato, Swiggy etc. It was very interesting to know how the NoSQL databases typically store, retrieve and handle data. I was also very interested to know about the unqlite database and its functionalities.

REFLECTION:

I initially started the project by making a list of the technology stacks that are required for the project and getting a detailed knowledge about the technology stacks. Technology stacks that are used in the project include programing languages like python and NoSQL database like unqlite. I also acquired knowledge about the python math module and its radians, sin, cos, and atlan2 functions. I also studied about error handling in python.

Once I gathered enough knowledge on the technology stacks that are required for the project I started to code the project. I first read the database using the below commands and then printed the collection like given in the below image to see the structure of the collection so that I know the structure when the collection is passed as input to the functions and know how the data can be manipulated.

```
from unqlite import UnQLite

db = UnQLite('sample.db')

data = db.collection('data')
```

```
In [15]: # Ensure the collection exists
         if not data.exists():
             print('The collection does not exist.')
         else:
             # Iterate through the items in the collection and print them
             for item in data.all():
                 print(item)

{'business_id': 'MPyxaNVuWlAQqJ0iKV5rQw', 'type': 'business', 'state': 'AZ', 'latitude': 33.3482589, 'name': "VinciTorio's Rest
aurant", 'full_address': '1835 E Elliot Rd, Ste C109, Tempe, AZ 85284', 'categories': ['Restaurants', 'Buffets', 'Italian'], 'o
pen': True, 'stars': 4, 'city': 'Tempe', 'neighborhoods': [], '__id': 0, 'review_count': 122, 'longitude': -111.9088346}
{'business_id': '6wKacaFIYcgsaYAZXd4Dkw', 'type': 'business', 'state': 'AZ', 'latitude': 33.4466995, 'name': 'Virginia G. Piper
Sports & Fitness Center for Persons with Disab', 'full_address': '5031 E Washington St, Phoenix, AZ 85034', 'categories': ['Act
ive Life', 'Sports Clubs'], 'open': True, 'stars': 5, 'city': 'Phoenix', 'neighborhoods': [], '__id': 1, 'review_count': 3, 'lo
ngitude': -111.9735208}
```

Then, I started by writing the code for **FindBusinessBasedOnCity** function. I first made a rough copy of the function and later optimized the code to obtain the result shown in the image below. I first open the file at save_location parameter provided in write mode. It will overwrite the file if the file is already present else will create a new file.

Next, I iterate over all the collection rows to check if a row contains a city that is same as the city_to_search parameter passed to the function. If a document has the city, then the name, address, city and state of the document that matches the city is stored in the file in this format name$full_address$city$state.

```
In [3]: def FindBusinessBasedOnCity(city_to_search, save_location, collection):
            with open(save_location, 'w') as file:
                for document in collection.all():
                    if document.get('city') == city_to_search:
                        file.write(f"{document['name']}${document['full_address']}${document['city']}${document['state']}\n")
```

Next, I started to code the **FindBusinessBasedOnLocation** and **DistanceFunction.** For the distance function I just used the same logic provided in the question. For the FindBusinessBasedOnLocation function. Like the FindBusinessBasedOnCity function it first opens the save_location file in the write mode to append new data to it or create the file if not present. Next for each document in the collection the function searches if there is an intersection between the set of categories_to_search and the categories in the document. If there is a match, then the distance is calculated between the latitude and longitude of the filtered document and the input latitude and input longitude. Finally, the distance is checked if it's below the max_distance provided as input. If it satisfies the condition, then then the name of the document is written to the save_location.

```
In [4]: FindBusinessBasedOnLocation(categories_to_search, my_location, max_distance, save_location, collection):
        with open(save_location, 'w') as file:
            for document in collection.all():
                if set(categories_to_search).intersection(document.get('categories', [])):
                    business_location = [document.get('latitude'), document.get('longitude')]
                    if business_location[0] is not None and business_location[1] is not None:
                        if distance_function(my_location[0], my_location[1], business_location[0], business_location[1]) <= max_distance:
                            file.write(f"{document['name']}\n")
```

```
In [2]: import math

        def distance_function(lat2, lon2, lat1, lon1):
            R = 3959  # Radius of the Earth in miles
            phi1 = math.radians(lat1)
            phi2 = math.radians(lat2)
            delta_phi = math.radians(lat2 - lat1)
            delta_lambda = math.radians(lon2 - lon1)

            a = math.sin(delta_phi / 2) ** 2 + math.cos(phi1) * math.cos(phi2) * math.sin(delta_lambda / 2) ** 2
            c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
            d = R * c
            return d
```

LESSONS LEARNED:

1. Error handling in python and how errors are typically handled.
2. Data Validation and how to handle “None” type of values and other missing values.
3. How NoSQL databases are handled in a restaurant application. How the restaurant data can be stored and retrieved efficiently to calculate various calculations.
4. Unqlite database and its functionalities.
5. How to efficiently write a function in python by minimized space and time complexity.
6. Various functions of the math module in python and how they typically work.

OUTPUT:

Below are the screenshots of the output_loc and output_city files after running the given test cases.

TEST CASE 1:

Check for the function FindBusinessBasedOnCity

```
In [5]: import math

In [16]: true_results = ["VinciTorio's Restaurant$1835 E Elliot Rd, Ste C109, Tempe, AZ 85284$Tempe$AZ", "P.croissants$7520 S Rural Rd, Te

try:
    FindBusinessBasedOnCity('Tempe', '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 ("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) != 3:
    print ("The FindBusinessBasedOnCity function does not find the correct number of results, should be 3.")

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!
```

```
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
```

TEST CASE 2:

Check for the function FindBusinessBasedOnLocation

In [17]:
```
true_results = ["VinciTorio's Restaurant"]

try:
    FindBusinessBasedOnLocation(['Buffets'], [33.3482589, -111.9088346], 10, '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.")

if lines[0].strip() == true_results[0]:
    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.
```

```
VinciTorio's Restaurant
```

RESULT:

Below shows the results of the code by passing all the testcases provided for the project.

In [5]:
```
import math
```

In [16]:
```
true_results = ["VinciTorio's Restaurant$1835 E Elliot Rd, Ste C109, Tempe, AZ 85284$Tempe$AZ", "P.croissants$7520 S Rural Rd, Te

try:
    FindBusinessBasedOnCity('Tempe', '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 ("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) != 3:
    print ("The FindBusinessBasedOnCity function does not find the correct number of results, should be 3.")

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!
```

In [17]:
```
true_results = ["VinciTorio's Restaurant"]

try:
    FindBusinessBasedOnLocation(['Buffets'], [33.3482589, -111.9088346], 10, '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.")

if lines[0].strip() == true_results[0]:
    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.
```

In [8]:
```
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 3.")

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!
```

In [9]:
```
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 3.")
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!
```

In [10]:

```
true_results =['The Seafood Market']
try:
    FindBusinessBasedOnLocation(['Specialty Food'], [33.3482589, -111.9088346], 10, '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.
```

In [11]:

```
true_results = ['P.croissants']
try:
    FindBusinessBasedOnLocation(['Bakeries'], [33.3482589, -111.9088346], 10, '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.
```

In [12]:

```
true_results = ['The Seafood Market', 'P.croissants']
try:
    FindBusinessBasedOnLocation(['Food', 'Specialty Food'], [33.3482589, -111.9088346], 10, '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) != 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 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.
```