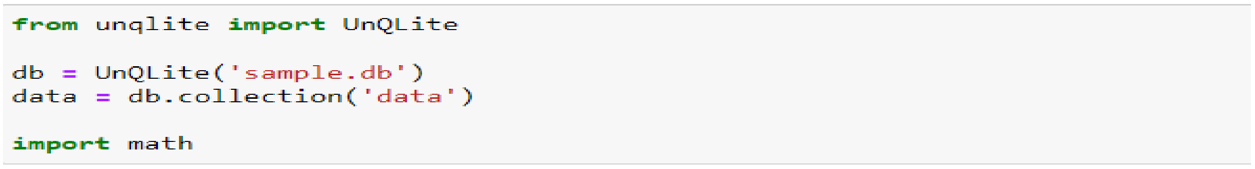
Project 1: No SQL Report

Func 1: FindBusinessBasedOnCity (cityToSearch, saveLocation1, collection)

Reflection:

It is used to search the collection of firms present in the given city and will store into a location.

Here we use a UnQlite library in python to load data from NoSQL database.



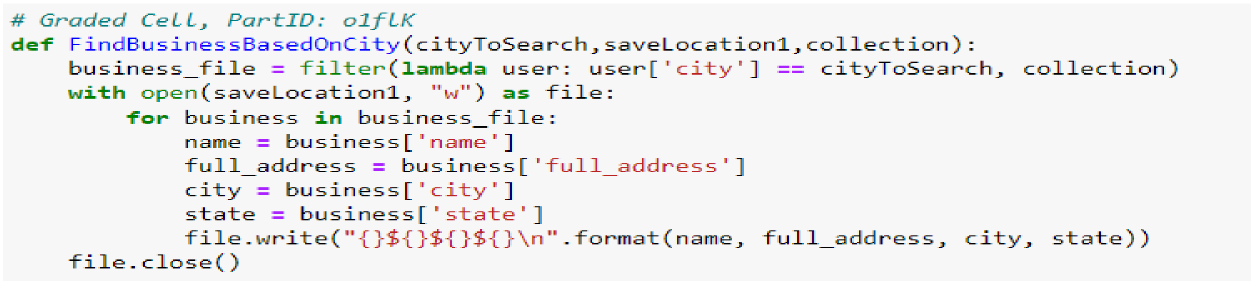
```
from unqlite import UnQLite

db = UnQLite('sample.db')
data = db.collection('data')

import math
```

The first function will take 3 arguments cityToSearch to pass a city name, saveLocation1 to save business in a location and collection of business.

Here in my code first line is to get the city I want to search and then the name, full address, city, state will save to a saveLocation1 which in turn save to a file named “output_city.txt”



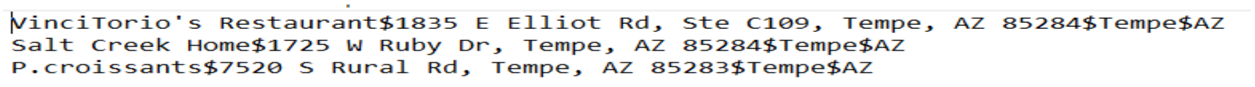
```
# Graded Cell, PartID: o1flK
def FindBusinessBasedOnCity(cityToSearch,saveLocation1,collection):
    business_file = filter(lambda user: user['city'] == cityToSearch, collection)
    with open(saveLocation1, "w") as file:
        for business in business_file:
            name = business['name']
            full_address = business['full_address']
            city = business['city']
            state = business['state']
            file.write("{}${}${}${}\n".format(name, full_address, city, state))
    file.close()
```

Lessons Learned:

I learned that UnQlite library in python that it is used to load data from NoSQL Database and how to filter out the data from a collection of data in NoSQL database using python.

Output:

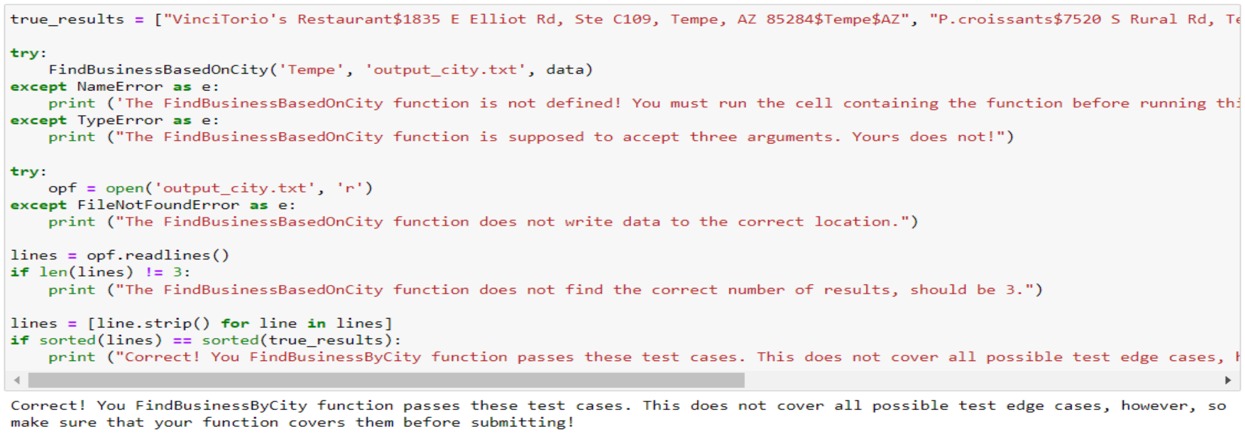
All the records in the output_city.txt will have name, full address, city and state and each new record will store in new line.



```
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
```

Result:

All the test cases have been passed without any errors



```
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!
```

Func 2: FindBusinessBasedOnLocation (categoriesToSearch, myLocation, maxDistance, saveLocation2, collection) –

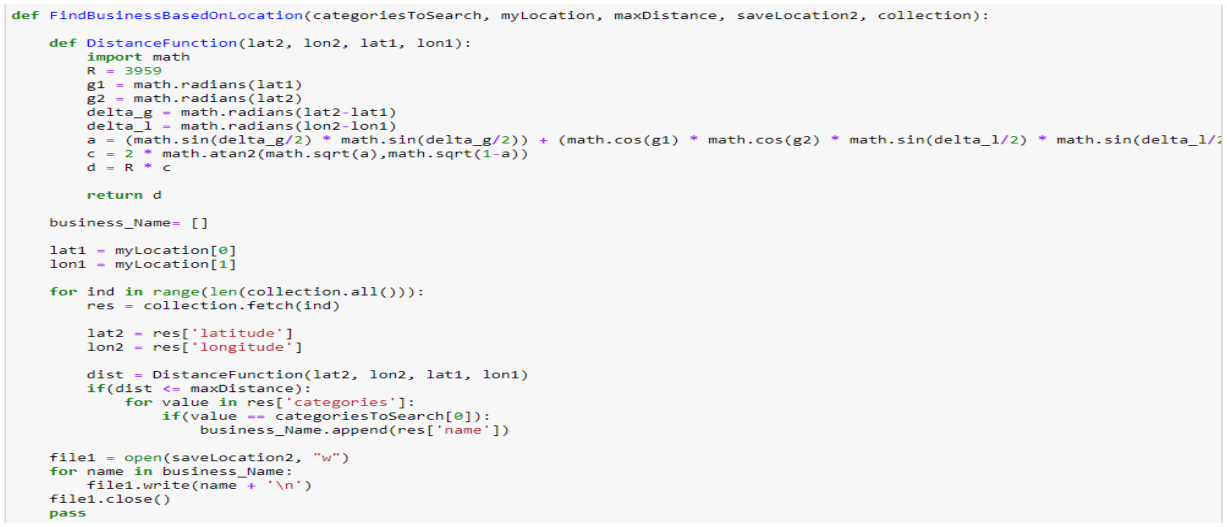
Reflection:

It is used get the name of the business that is present the maximum distance from the given location and in the categories list.

Here we need to import the math library as math functions are used inside the function.

This function will take 5 arguments categoriesToSearch, myLocation, maxDistance, saveLocation2, collection.

First, we need to compute the distance amid the given myLocation and the other locations using "**DistanceFunction"** and then we must check whether the calculated distance is less than the maximal distance and the specific group is in the list of DB Categories then the name of the business will be saved to saveLocation2 which in turn will write to a file "output_loc.txt"



```
def FindBusinessBasedOnLocation(categoriesToSearch, myLocation, maxDistance, saveLocation2, collection):

    def DistanceFunction(lat2, lon2, lat1, lon1):
        import math
        R = 3959
        g1 = math.radians(lat1)
        g2 = math.radians(lat2)
        delta_g = math.radians(lat2-lat1)
        delta_l = math.radians(lon2-lon1)
        a = (math.sin(delta_g/2) * math.sin(delta_g/2)) + (math.cos(g1) * math.cos(g2) * math.sin(delta_l/2) * math.sin(delta_l/
        c = 2 * math.atan2(math.sqrt(a),math.sqrt(1-a))
        d = R * c

        return d

    business_Name= []

    lat1 = myLocation[0]
    lon1 = myLocation[1]

    for ind in range(len(collection.all())):
        res = collection.fetch(ind)

        lat2 = res['latitude']
        lon2 = res['longitude']

        dist = DistanceFunction(lat2, lon2, lat1, lon1)
        if(dist <= maxDistance):
            for value in res['categories']:
                if(value == categoriesToSearch[0]):
                    business_Name.append(res['name'])

    file1 = open(saveLocation2, "w")
    for name in business_Name:
        file1.write(name + '\n')
    file1.close()
    pass
```

Lessons Learned:

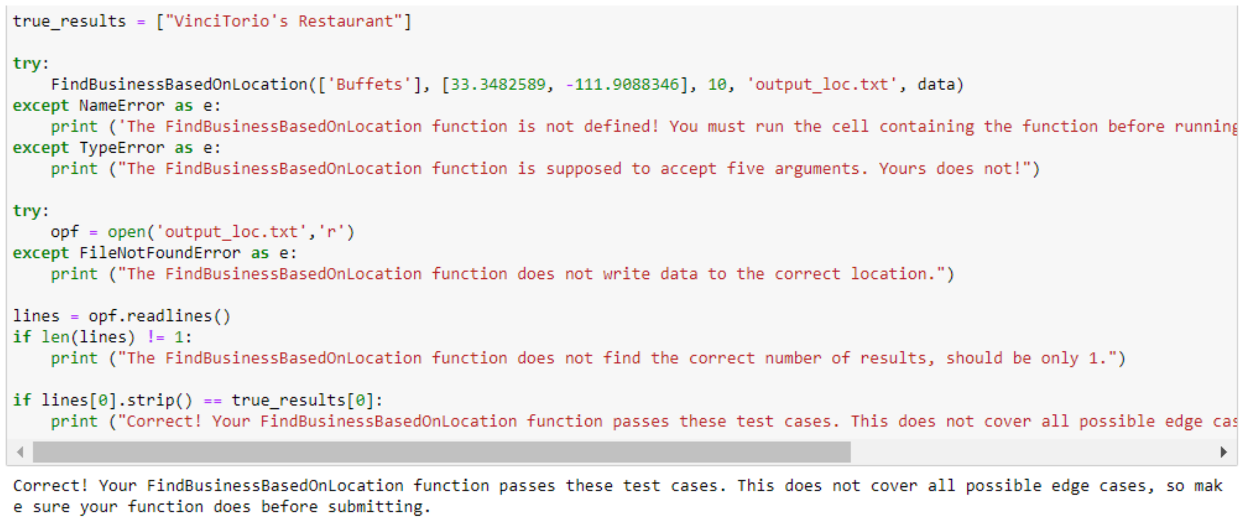
I learned using the math functions to calculate the distance and it took me so much time as auto grader is giving errors without “import math” in the function and how to iterate through the list in python.

Output:

```
VinciTorio's Restaurant
```

Result:

All the test cases have been passed



```
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.
```