**Assignment 5**

**MongoDB Restaurant Analysis**

1. Create database – restaurant, create collection – rescollection. Insert the documents into collections.

from pymongo import MongoClient

import json

if \_\_name\_\_ == "\_\_main\_\_":

    print("welcome to pymongo")

    client=MongoClient("mongodb://localhost:27017")

    db=client['restaurant']

    collection=db['rescollection']

    with open ('D:/python trianings/resorants\_air/restaurants-dataset.json',"r",encoding="utf-8") as file:

        record=file.read()

        record=record.replace('\n','')

        record=record.replace('}{','},{')

        record="["+record+"]"

        file\_data=json.loads(record)

    if isinstance(file\_data,list):

        collection.insert\_many(file\_data)

    else:

        collection.insert\_one(file\_data)

1. Display all the documents in the collection restaurants.

> db.rescollection.find()

1. Display the fields restaurant\_id, name, borough, and zip code, but exclude the field \_id for all the documents in the collection restaurant.

>db.rescollection.find({},{\_id:0,restaurant\_id:1,name:1,borough:1,"address.zipcode":1})

1. Find the restaurants who achieved a score more than 90.

>db.rescollection.aggregate([{$match:{"grades.score":{$gt:90}}},{$group:{\_id:'$grades.score',total:{$sum:"$grades.score"}}}])

1. Show the restaurants that achieved a score, more than 80 but less than 100.

>db.rescollection.aggregate([{$match:{"grades.score":{$gt:80,$lt:100}}},{$group:{\_id:'$grades.score',total:{$sum:"$grades.score"}}}])

1. Write Query to show the restaurants that do not prepare any cuisine of american & their grade score > 70.

>db.rescollection.find({$and:[{cuisine:{$ne:'American'}},{"grades.score":{$gt:70}}]})

1. Write a MongoDB query to arrange the name of the cuisine in an ascending order and for that same borough arranged in descending order.

db.rescollection.find().sort({"cuisine":1,"borough":-1})

1. Write a MongoDB query to arrange the name of the cuisine in descending order.

db.rescollection.find().sort({"cuisine":-1})

1. Show the restaurant Id, name, borough and cuisines for those restaurants which prepared dish except 'American' and 'Chinese' or restaurant's name begins with letter 'Bil'.

>db.rescollection.find({$or:[{name:/^Bil/},{"$and":[{"cuisine":{$ne:"American"}},

{"cuisine":{$ne:"Chinese"}}]}]},{resraurant\_id:1,name:1,borough:1,cuisine:1})

1. Show the restaurant Id, name, borough and cuisines and score for restaurant's name begins with letter 'Bil'.

>db. rescollection.find({$or:[{name: /^Bil/},

{“restaurant\_id" : 1,"name":1,"borough":1,"cuisine" :1, "grades.score":1});

1. Show the restaurant Id, name, borough and cuisines and score for restaurant serving “Indian” as cuisines.

db.rescollection.aggregate([{$match:{cuisine:"Indian"}}],{restaurant\_id:1,name:1,borough:1,cuisine:1,'grades.score':1})

1. Write a query to show all the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' anywhere in its name.

db.rescollection.find({"name":{$regex:/.\*li/}},{restaurant\_id:1,name:1,borough:1,cuisine:1,'g rades.score':1})

1. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

db.rescollection.find({"grades.score":{$mod:[7,0]}},{restaurant\_id:1,name:1,grades:1})

1. Show document/record counts that has street and not street in addresses.

db.rescollection.find({"address.street":{$exists:true}})

1. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168