**Mongo DB Assignment – 3**

**Mongo DB Exercise – With the Restaurants Data Set**

1. Write a MongoDb query to display all the documents in the collection restaurants.

db.addresses.find()

1. Write a MongoDb query to display the fields restaurants\_id, name, borough and cuisine for all the documents in the collection restaurant.

db.addresses.find({},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

1. Write a MongoDb query to display the fields restaurants\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.

db.addresses.find({},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1,"\_id":0})

1. Write a MongoDb query to display the fields restaurants\_id, name, borough and zip code, but exclude the field\_id for all the documents in the collection restaurant.

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

1. Write a MongoDb query to display the first 5 restaurants which is in the borough Bronx.

db.addresses.find({"borough":"Bronx"}).limit(5)

1. Write a MongoDb query to display all the restaurant which is in the borough Bronx.

db.addresses.find({"borough":"Bronx"})

1. Write a MongoDb query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

db.addresses.find({"borough":"Bronx"}).skip(5).limit(5)

1. Write a MongoDb query to find the restaurants who achieved a score more than 90.

db.addresses.find({"grades.score":{$gt:90}})

1. Write a MongoDb query to find the restaurants that achieved a score, more than 80 but less than 100.
2. Write a MongoDb query to find the restaurants which locate in latitude value less than -95.754168.

db.addresses.find({"address.coord":{$lt:-95.754168}})

1. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.

db.addresses.find({$and:[{"cuisine":{$ne:'American'}},{"grades.score":{$gt:70}},{"address.coord":{$lt:-65.754168}}]})

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.
2. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.

db.addresses.find({"cuisine":{$ne:"American"},"grades.grade":'A',"borough":{$ne:"Brooklyn"}}).sort({"cuisine":-1})

1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.

db.addresses.find({"name":/^Wil/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

1. db.addresses.find({"name":/ces$/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})
2. db.addresses.find({"name":/.\*Reg.\*/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})
3. db.addresses.find({"borough":"Bronx",$or:[{"cuisine":"American"},{"cuisine":"Chinese"}]})
4. db.addresses.find({"borough":{$in:["Staten Island","Queens","Bronx","Brooklyn"]}},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})
5. db.addresses.find({"borough":{$nin:["Staten Island","Queens","Bronx","Brooklyn"]}},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})
6. db.addresses.find({"grades.score":{$not:{$gt:10}}},{"restaurant\_id":1,"name":1,"cuisine":1,"borough":1})
7. db.addresses.find({$or:[{"name":/^Wil/},{$and:[{"cusine":{$ne:"American"}},{"cusine":{$ne:"Chinese"}}]}]},{"restaurant\_id":1,"name":1,"cuisine":1,"borough":1})
8. db.addresses.find({"grades.grade":'A',"grades.score":11, "grades.date":ISODate("2014-08- 11T00:00:00Z")},{"restaurant\_id":1,"name":1,"grades":1})
9. db.addresses.find({"address.coord.1":{$gt:42,$lt:52}},{"restaurant\_id":1,"name":1,"address":1,"coord":1})
10. db.addresses.find().sort({"name":1})
11. db.addresses.find().sort({"name":-1})
12. db.addresses.find().sort({"cuisine":1,"borough":-1})
13. db.addresses.find({"address.street":{$exists:true}})