**Assignment 3**

* Display all documents in collection restaurant.

> db.addresses.find().pretty()

* To display the fields restaurant\_id, borough, cuisine for all the documents in the collection restaurant.

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

* To display the fields restaurant\_id, borough, cuisine but exclude field\_id for all the documents in the collection restaurant.

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

* To display the fields restaurant\_id, borough, zipcode but exclude field\_id for all the documents in the collection restaurant.

> db.addresses.find({},{ "restaurant\_id":1, "borough":1, "zipcode":1,"\_id":0}).pretty()

* To display first 5 restaurant which is in the borough Bronx.

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

* To display all restaurant which is in the borugh Bronx.

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

* To display next 5 restaurants after skipping first 5 which are in the borough Bronx.

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

* To find the restaurants who achieved score more than 90.

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

* To find the restaurants who achieved score more than 80 but less than 100.

> db.addresses.find({"grades.score":{$gt:80,$lt:100}}).pretty()

* To find the restaurants which locate in latitude value less than -95.754168.

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

* To find the restaurants that do not prepare any cuisine of ‘American” and grade score more than 70 and latitude less than -65.754168.

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

* To find the restaurants which do not prepare any cuisine of ‘American & score is more than 70 and located in longitude less the -65.754168.

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

* To find the restaurant which do not prepare any cuisine of ‘American’ , achieved grade point ‘A’ , does not belongs to Brooklyn. Display according to cuisine in descending order.

> db.addresses.aggregate([{$and:[{cuisine:{$ne:"American"}},{$and:[{"grades.grade":{$eq:"A"}},{borough:{$ne:"Brooklyn"}}]}]},{$sort:{cuisine:-1}}])

* To find restaurant\_id,name , borough, cuisine which contant ‘Wil” as first 3 letters.

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

* To find the restaurant\_id, name, borough,cuisine who contain ‘ces’ as last 3 letters for it’s name

> db.addresses.find({ "name": { "$regex": "ces$"}},

{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

* To find the restaurant\_id, name, borough, cuisine which contain’Reg’ in between for it’s name

> db.addresses.find({name:{$regex:/Reg/}},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

* To find restaurants which belong to borough Bronx and prepared with American / Chinese dish

> db.addresses.find({$and:[{borough:"Bronx"},{$or:[{cuisine:"American"},{cuisine:"Chinese"}]}]})

* To find restaurant\_id , name , borough , cuisine for those restaurants which belong to borough Staten island or Queens or Bronxor Brooklyn

> db.addresses.find({$or:[{borough:"Bronxor Brooklyn"},{borough:"Queens"},{borough:"Staten Island"}]},{ "restaurant\_id":1, "name":1, "borough":1, "cuisine":1})

* To find the restaurant\_id , name , borough and cuisine for those restaurants which are not belonging to the borough Staten Island or queens Bronxor Brooklyn

> db.addresses.find({$or:[{borough:{$ne:"Bronxor Brooklyn"}},{borough:{$ne:"Queens"}},{borough:{$ne:"Staten Island"}}]},{ "restaurant\_id":1, "name":1, "borough":1, "cuisine":1})

* To find the restaurant\_id , name , borough and cuisine for those restaurants where score is not more than 10

> db.addresses.find({"grades.score":{$lt:10}},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

* To find the restaurant\_id , name , borough and cuisine for those restaurants which prepared dish except “American” and “Chinese” or restaurants name begins with ‘Wil’

> db.addresses.find({$or:[{$and:[{cuisine:{$ne:"American"}},{cuisine:{$ne:"Chinese"}}]},{name:{$regex:/^Wil/}}]},{"restaurant\_id":1,"name":1,"cuisine":1,"borough":1}).pretty()

* To find the restaurant id , name and grades for those restaurants which achieved grade of ‘A’ and score 11 on an ISO date “24-08-11T00:00:00Z”

> db.addresses.find({$and:[{"grades.grade":"A"},{$and:[{"grades.score":11},{"grades.date":ISODate("2014-08-11T00:00:00Z")}]}]},{"restaurant\_id":1 , "name":1 , "grades":1})

* To find the restaurant id , name and grades for those restaurants which achieved grade of ‘A’ and score 9 on an ISO date “24-08-11T00:00:00Z”

> db.addresses.find( { "grades.1.date": ISODate("2014-08-11T00:00:00Z"), "grades.1.grade":"A" , "grades.1.score" : 9 }, {"restaurant\_id" : 1,"name":1,"grades":1}).pretty()

* To find the restaurant\_id, name , address , location for those restaurants where 2nd element of coord array contains value more than 42 and upto 52

> db.addresses.find( { "address.coord.1": {$gt : 42, $lte : 52}}, {"restaurant\_id" : 1,"name":1,"address":1,"coord":1}).pretty()

* To arrange the name in ascending order

> db.addresses.aggregate({$sort:{name:1}})

* To arrange the name in descending order

> db.addresses.aggregate({$sort:{name:-1}})

* To arrange the cuisine in ascending and borough in descending

> db.addresses.aggregate({$sort:{cuisine:1,borough:-1}})

* To know whether all the addresses contains the street or not

> db.addresses.find( {"address.street" : { $exists : true } } ).pretty()

* To select all documents in the restaurants collection where the coord fied value is Double

> db.addresses.find( {"address.coord" : {$type : 1} } ).pretty()

* To select restaurant id , name , grades for those restaurants which returns 0 as remainder after diving the score by 7

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

* To find the restaurant id , name borough , longitude , latitude , cuisine for restaurant which contains 3 letters somewhere in its name

> db.addresses.find({name:{$regex:/mon/}},{"coord":1,"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

* To find the restaurant name , borough , longitude and latitude , cuisine for those restaurants which contain ‘Mad’ as first 3 letters of its name

> db.addresses.find({name:{$regex:/^Mad/}},{"coord":1,"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})