**Schema of the DB restaurant**

{

"\_id" : ObjectId("5c88dbf28fcb620a6a7bf516"),

"address" : {

"building" : "1007",

"coord" : [

-73.856077,

40.848447

],

"street" : "Morris Park Ave",

"zipcode" : "10462"

},

"borough" : "Bronx",

"cuisine" : "Bakery",

"grades" : [

{

"date" : ISODate("2014-03-03T00:00:00Z"),

"grade" : "A",

"score" : 2

},

{

"date" : ISODate("2013-09-11T00:00:00Z"),

"grade" : "A",

"score" : 6

},

{

"date" : ISODate("2013-01-24T00:00:00Z"),

"grade" : "A",

"score" : 10

},

{

"date" : ISODate("2011-11-23T00:00:00Z"),

"grade" : "A",

"score" : 9

},

{

"date" : ISODate("2011-03-10T00:00:00Z"),

"grade" : "B",

"score" : 14

}

],

"name" : "Morris Park Bake Shop",

"restaurant\_id" : "30075445"

}

1. **Write a MongoDB query to display all the documents in the collection restaurants.** db.restaurant.find();
2. **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.** db.restaurant.find({},{“restaurant”:1,”name”:1,”borough”:1,“cuisine”:1});
3. **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.** db.restaurants.find({},{"restaurant\_id":1,"name":1,"borough":1,"cusine":1,"\_id":0});
4. **Write a MongoDB query to display all the restaurant which is in the borough Bronx.**

db.restaurants.find({"borough":"Bronx"}).pretty();

1. **Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx**

db.restaurants.find({"borough":"Bronx"}).pretty().limit(5);

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

db.restaurants.find({"borough":"Bronx"}).pretty().limit(3).skip(5);

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

db.restaurants.find({"grades.score":{$gt:90}}).pretty();  
**OR,**db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});

1. **Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.**

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

1. **Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168**

db.restaurants.find({"address.coord.0":{"$lt":-95.754168}}).pretty();

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.restaurants.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. Note : Do this query without using $and operator.**

db.restaurants.find({

"cuisine":{$ne:"American"},

"grades.score":{$gt:70},

"address.coord":{$lt: -65.75}

});

1. **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.restaurants.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.restaurants.findOne({"name":/^Wil/},{"cuisine":1, "\_id":0, "restaurant\_id":1, "name":1, "borough":1});

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

db.restaurants.findOne({"name":/ces$/},{"cuisine":1, "\_id":0, "restaurant\_id":1, "name":1, "borough":1});

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

> db.restaurants.findOne({"name":/.\*Reg.\*/},{"cuisine":1, "\_id":0, "restaurant\_id":1, "name":1, "borough":1});

1. **Asd**
2. **Asd**
3. **Asd**
4. **Ad**
5. **Asd**
6. **Asd**
7. **Asd**
8. **Asd**
9. **Asd**
10. **Asd**
11. **Asd**
12. **Asd**
13. **Asd**
14. **Asd**
15. **Asd**