{

"address": {

"building": "1007",

"coord": [ -73.856077, 40.848447 ],

"street": "Morris Park Ave",

"zipcode": "10462"

},

"borough": "Bronx",

"cuisine": "Bakery",

"grades": [

{ "date": { "$date": 1393804800000 }, "grade": "A", "score": 2 },

{ "date": { "$date": 1378857600000 }, "grade": "A", "score": 6 },

{ "date": { "$date": 1358985600000 }, "grade": "A", "score": 10 },

{ "date": { "$date": 1322006400000 }, "grade": "A", "score": 9 },

{ "date": { "$date": 1299715200000 }, "grade": "B", "score": 14 }

],

"name": "Morris Park Bake Shop",

"restaurant\_id": "30075445"

}

PFA the above sample data in “restaurants.json” attached with this document.

Import it in mongo test database

> mongoimport restaurants.json

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

db.restaurants.find().pretty()

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

db.restaurants.find({},

{restaurant\_id:1,name:1,borough:1,cusisine:1}

).pretty()

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,cusisine:1,\_id:0}

).pretty()

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

db.restaurants.find({},

{restaurant\_id:1,name:1,borough:1,"address.zipcode":1,\_id:0}

).pretty()

5. Write a MongoDB query to display all the restaurant which is in the borough Bronx.

db.restaurants.find({borough:"Bronx"}

).pretty()

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

db.restaurants.aggregate([

{"$match":{"borough":"Bronx"}},

{"$limit":5}

]

).pretty()

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

db.restaurants.aggregate([

{"$match":{"borough":"Bronx"}},

{"$Skip":5}

]

).pretty()

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

db.restaurants.find(

{"grades":{$elemMatch:{score:{$gt:90}}}}

).pretty()

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

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

10. 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()

11. 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":{$gt:70}},{"address.coord.0":{$lt:65.754168}}]}).pretty()

12. 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.coords.1":{lt:-65}}).pretty()

13. 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":{$ne:"American "},"grades.grade":{$eq:"A"},"borough":{$ne:"Brooklyn"}}).pretty()

14. 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.find({'name':{'$regex':"^Wil"}},{restaurant\_1:1,name:1,borough:1,cuisine:1}).pretty()

15. 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.find({'name':{'$regex':"ces$"}},{restaurant\_1:1,name:1,borough:1,cuisine:1}).pretty()

16. 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.find({'name':/Reg/},,{restaurant\_1:1,name:1,borough:1,cuisine:1}).pretty()

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

db.restaurants.find({borough:"Bronx",$or:[{"cuisine":"American"},{"cuisine":"Chinese"}]}).pretty()

18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronx or Brooklyn.

db.restaurants.find({borough:"Bronx",$or:[{"cuisine":"American"},{"cuisine":"Chinese"}]}).pretty()

19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

db.restaurants.find({"borough" :{$nin :["Staten Island","Queens","Bronx","Brooklyn"]}},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.

db.restaurants.find(

{"grades.score" :

{ $not:

{$gt : 10}

}

},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);