{"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"}

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

Sol: db.restaurants.find()

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

Sol: db.restaurants.find({},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

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

Sol: db.restaurants.find({},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1,”\_id”:0})

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

Sol: db.restaurants.find({“restaurants\_id”:1,”name”:1,”borough”:1,”zipcode”:1,”\_id”=0})

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

Sol: db.restaurants.find({“borough”:”Bronx”})

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

Sol: db.restaurants.find({“borough”:”Bronx”}).limit(5)

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

Sol: db.restaurants.find({“borough”:”Bronx”}).skip(5).limit(5)

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

Sol: db.restaurants.find({“grades”:{$elemMatch:{“score”:{$gt:90}}}})

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

Sol: db.restaurants.find({“grade”:{$elemMatch:{“score”:{$gt:80,$lt:100}}}})

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

Sol: db.restaurants.find({“address.coord”:{$lt : -95.754168}})

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.

Sol:db.restaurants.find({$and:[{“cuisine”:{$ne:”American”}},{“grades.score”:{$gt:70}},{“address.coord”:{$lt:-65754168}}]})

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.

Sol:db.restaurants.find({“cuisine”:”American,”grades.score”:{$gt:70},”address.coord”:{$lt:-65.754168}})

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.

Sol:db.restaurants.find({“cuisine”:{$ne:”American”},”grades.grade”:”A”,”borough”:{$ne:”Brooklyn”}}).sort({“cuisine”:-1})

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.

Sol: db.restaurants.find({name:/^Wil/},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

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.

Sol:db.restaurants.find({name:/ces$/},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

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.

Sol:db.restaurants.find({name:/.\*Reg.\*/},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

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

Sol: db.restaurant.find({“borough”:”Bronx”,$or:[{“cuisine”:”American”},{“cuisine”:”Chinese}]})

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 Bronxor Brooklyn.

Sol: db.restaurants.find({“borough”:{$in:[“”Staten Island”,”Queens”,”Bronx”,”Brooklyn”]}},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

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.

Sol: 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.

Sol:db.restaurants.find({“grades.score”:{$not:{$gt:10}}},{“restaurants\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

21. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

Sol: db.restaurants.find({$or:[{name:/^Wil/},{“$and”:[{“cuisine:{$ne:”American”}},{“cuisine”:{$ne:”Chinees”}}]}]},{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1})

22. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates..

Sol: db.restaurants.find({“grades.date”:ISODATE(“2014-08-11T00:00:00Z),”grades.grade”:”A”,”grades.score”:11},{“restaurant\_id”:1,”name”:1,”grades”:1})

23. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

Sol: db.restaurants.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})

24. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52..

Sol:db.restaurants.find({“address.coord.1”:{$gt:42,$lte:52}},{“restaurant\_id”:1,”name”:1,”address”:1,”coord”:1})

25. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

Sol: db.restaurants.find().sort({“name”:1})

26. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

Sol: db.restaurants.find().sort({“name”: -1})

27. Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

Sol: db.restaurants.find().sort({“cuisine”:1,”borough”: -1})

28. Write a MongoDB query to know whether all the addresses contains the street or not.

Sol: db.restaurants.find({“address.street”:{$exists:true}})

29. Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is sol: db.restaurants.find({“address.coord”:{$type:1}})

30. 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.

Sol: db.restaurants.find({“grades.score”:{$mod:[7,0]}},{“restaurant\_id”:1,”name”:1,”grades”:1})

31. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

Sol: db.restaurants.find({name:{$regex:”mon.\*”,$options:”i”}},{“name”:1,”borough”:1,”address.coord”:1,”cuisine”:1})

32. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.

Sol:db.restaurants.find({name:{$regex:/^Mad/i}},{“name”:1,”borough”:1,”address.coord”:1,”cuisine”:1})