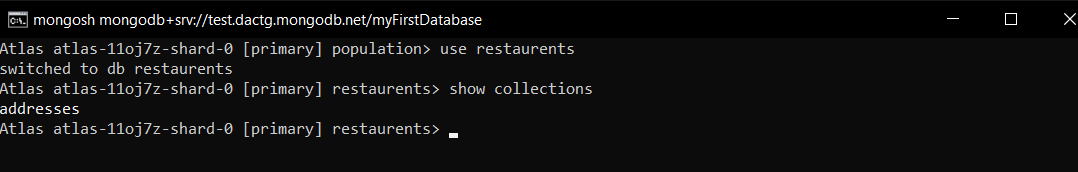
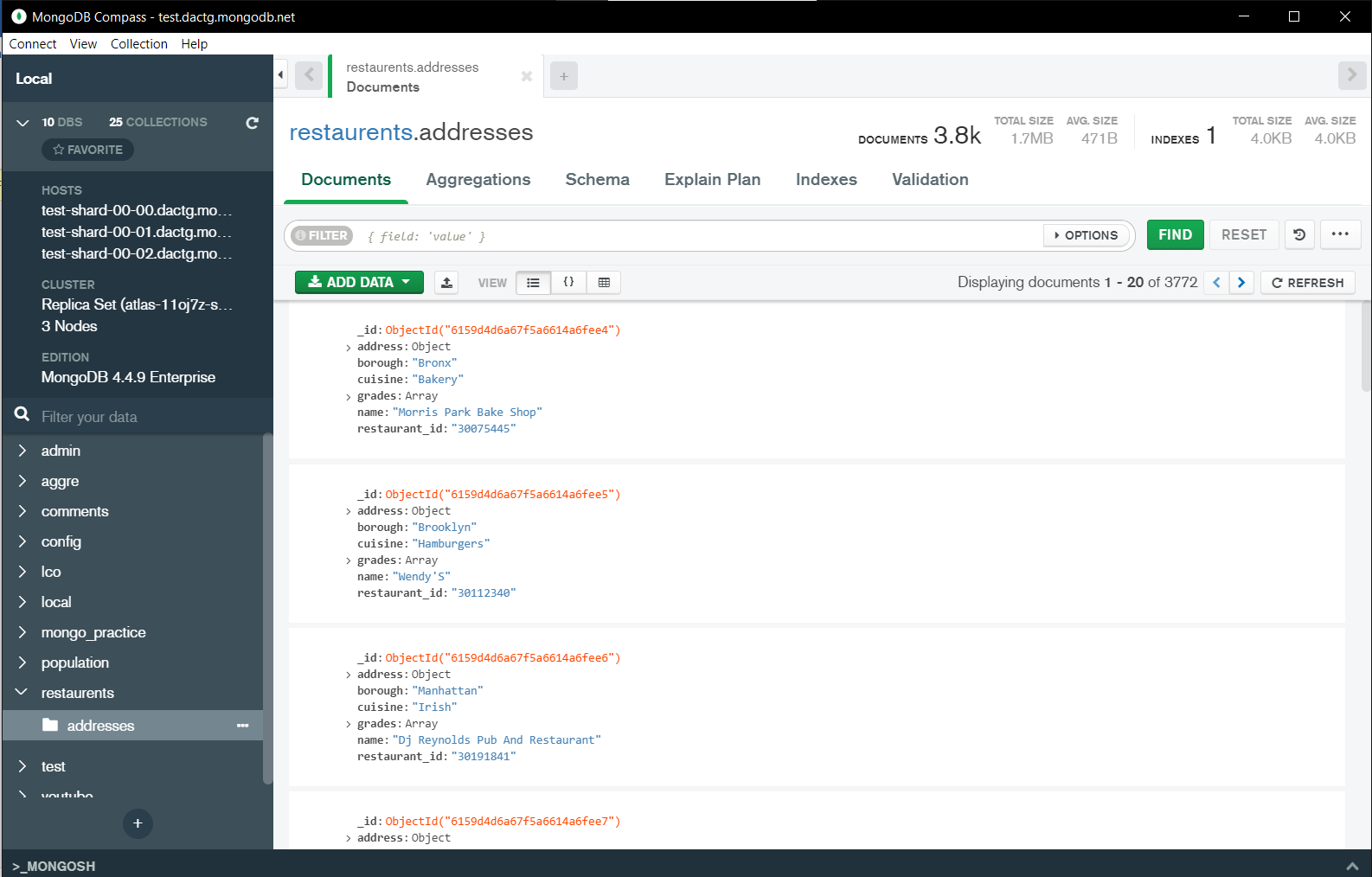
**MongoDB\_Assignment 3**

Mongodb complex-queries

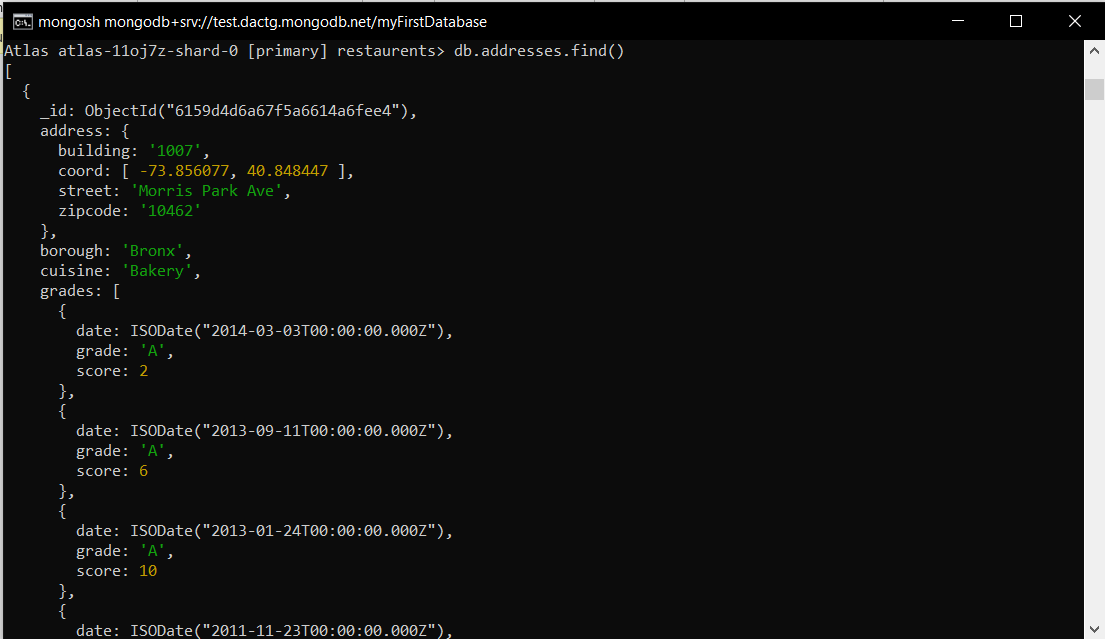
🡪show databases

🡪use restaurants



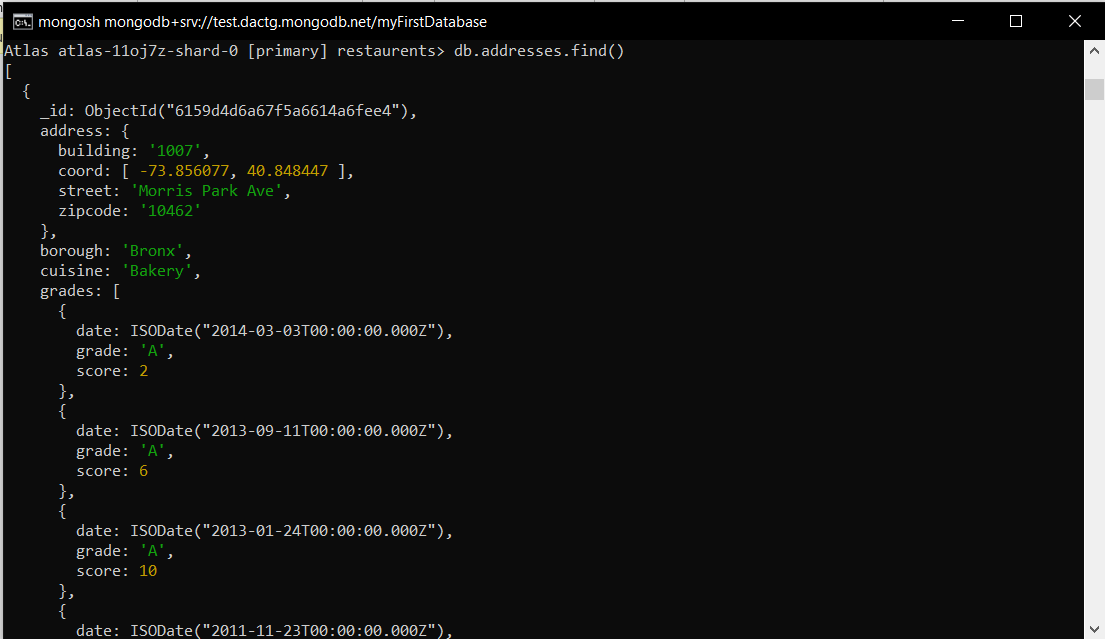


🡪db.addresses.find() should print entire json data

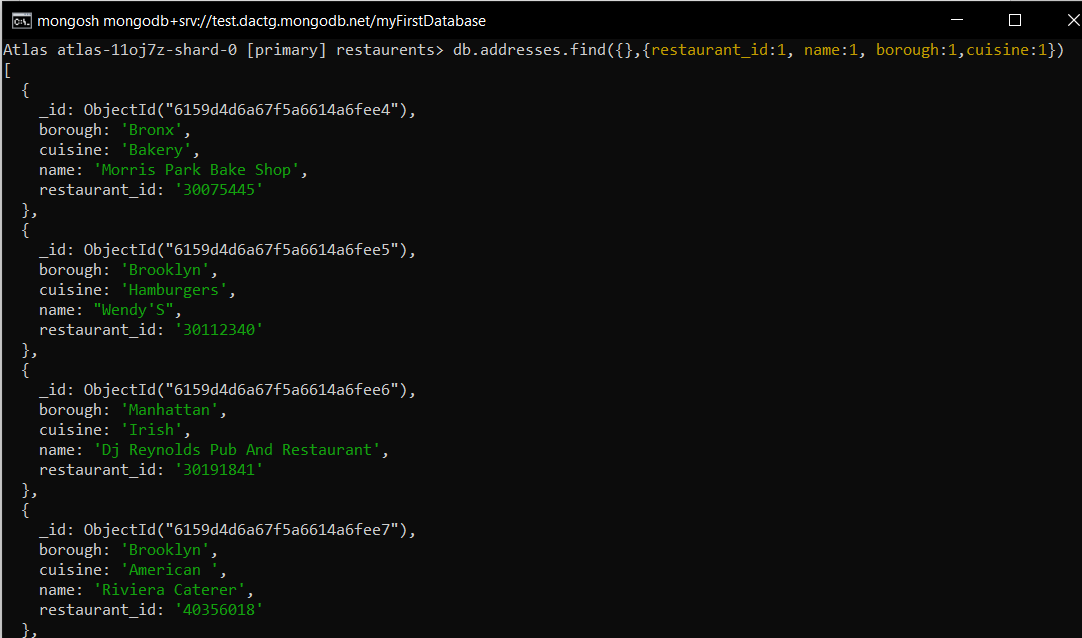


**Exercise Questions**

1. Write a MongoDB query to display all the documents in the collection restaurants
2. db.addresses.find().



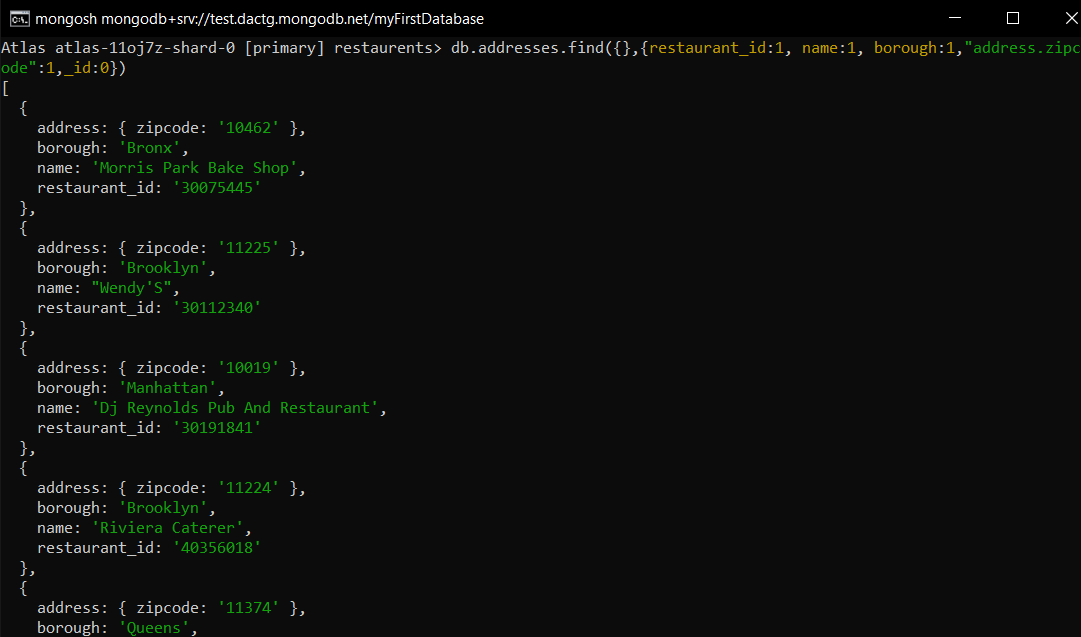
1. Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.
2. db.addresses.find({},{restaurant\_id:1, name:1, borough:1,cuisine:1})



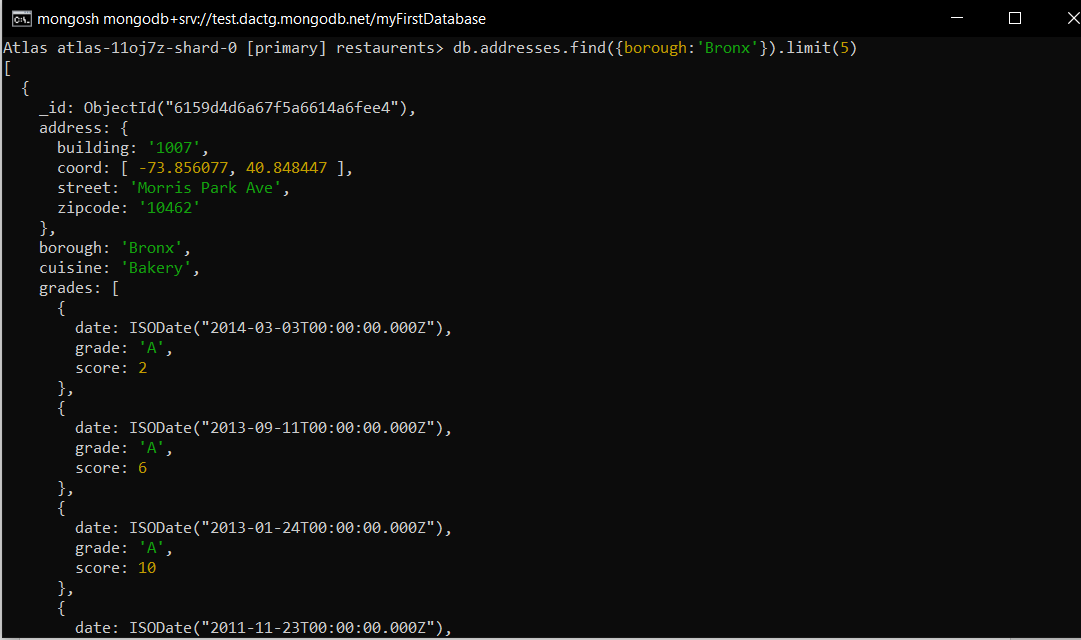
1. Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.
2. db.addresses.find({},{restaurant\_id:1, name:1, borough:1,\_id:0})



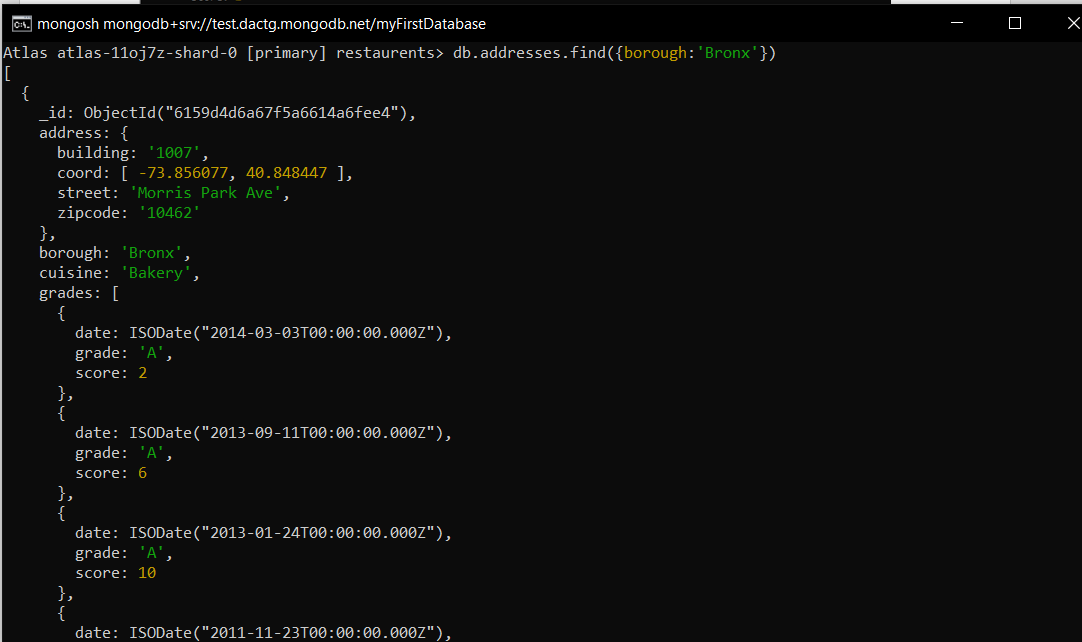
1. Write a MongoDB query to display the fields restaurant\_id, name, borough and zip code, but exclude the field \_id for all the documents in the collection restaurant.
2. db.addresses.find({},{restaurant\_id:1, name:1, borough:1,"address.zipcode":1,\_id:0})



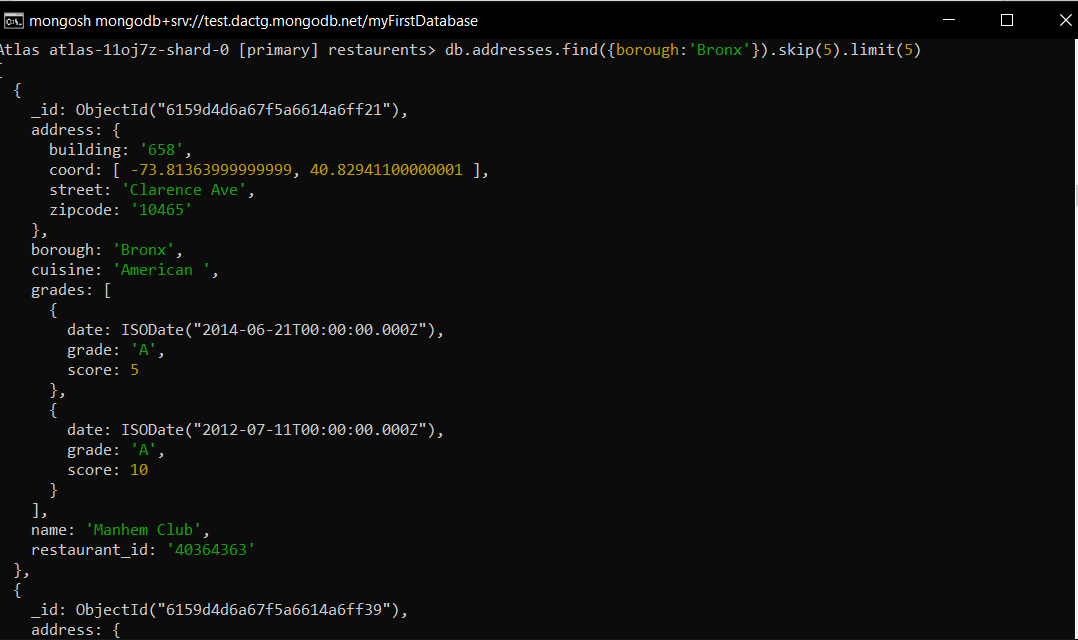
1. Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.
2. db.addresses.find({borough:'Bronx'}).limit(5)



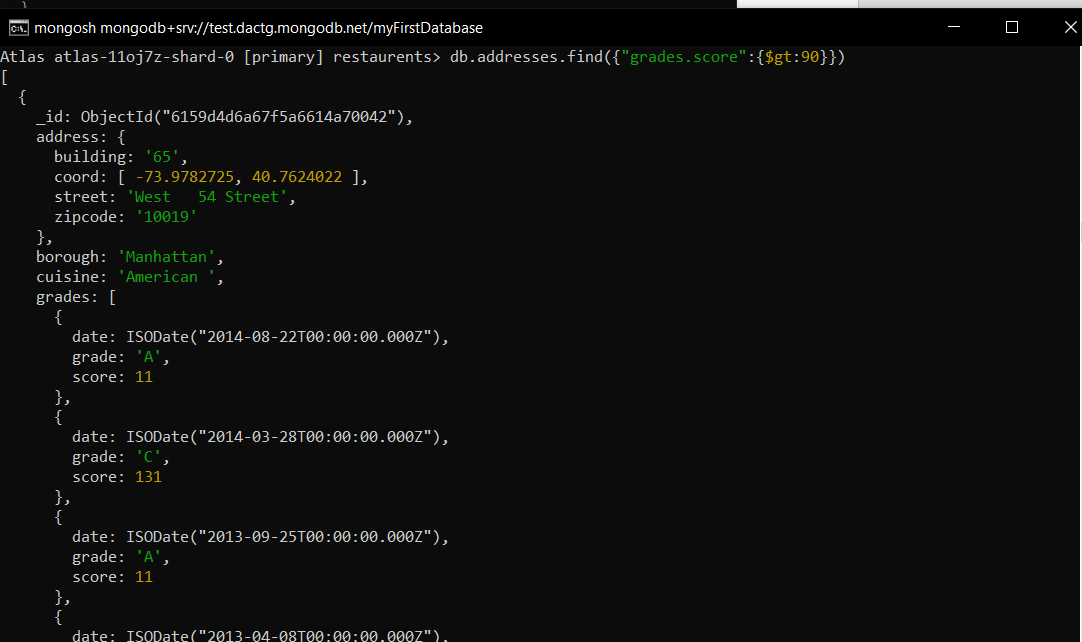
1. Write a MongoDB query to display all the restaurant which is in the borough Bronx.
2. db.addresses.find({borough:'Bronx'})



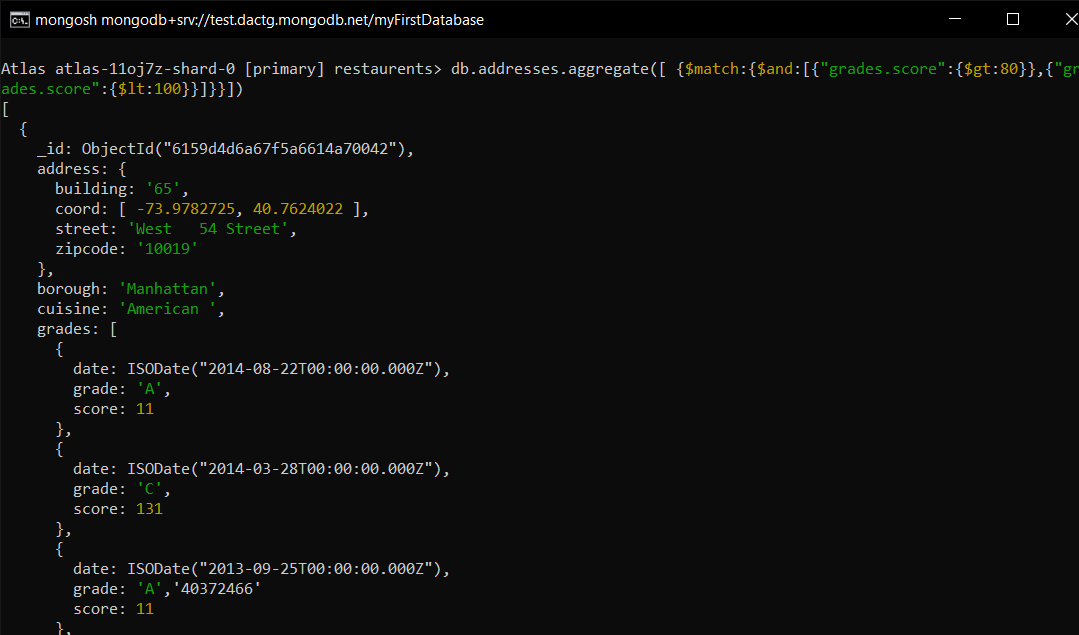
1. Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.
2. db.addresses.find({borough:'Bronx'}).skip(5).limit(5)



1. Write a MongoDB query to find the restaurants who achieved a score more than 90.
2. db.addresses.find({"grades.score":{$gt:90}})



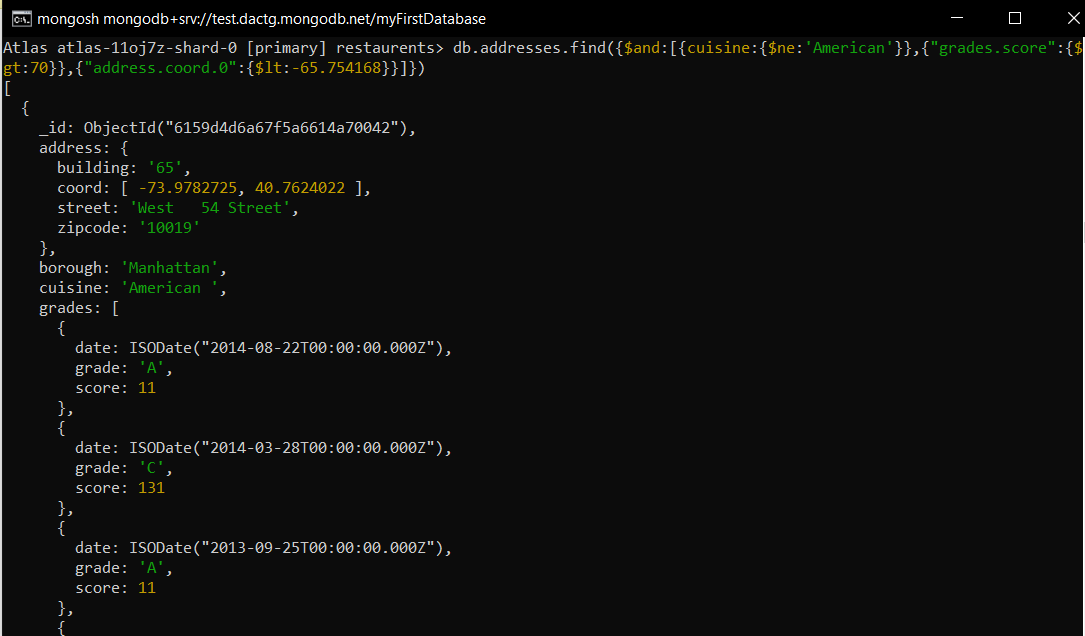
1. Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.
2. db.addresses.aggregate([{$match:{$and:[{"grades.score":{$gt:80}},{"grades.score":{$lt:100}}]}}])



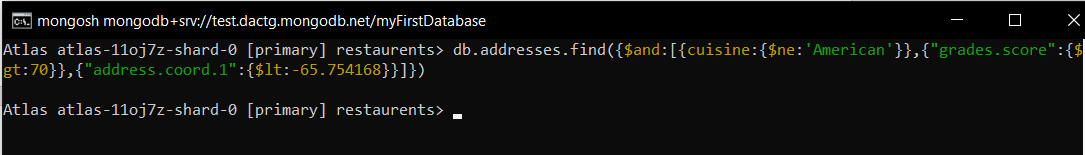
1. Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.
2. db.addresses.find({"address.coord.0":{$lt: -95.754168}})



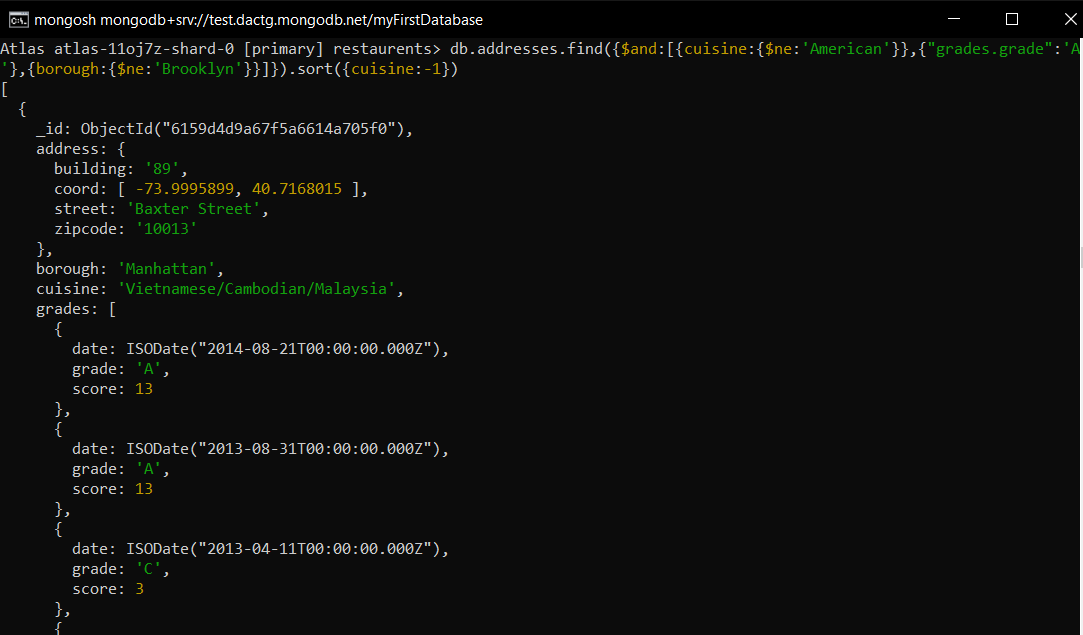
1. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.
2. db.addresses.find({$and:[{cuisine:{$ne:'American'}},{"grades.score":{$gt:70}},{"address.coord.0":{$lt:-65.754168}}]})



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. db.addresses.find({$and:[{cuisine:{$ne:'American'}},{"grades.score":{$gt:70}},{"address.coord.1":{$lt:-65.754168}}]})



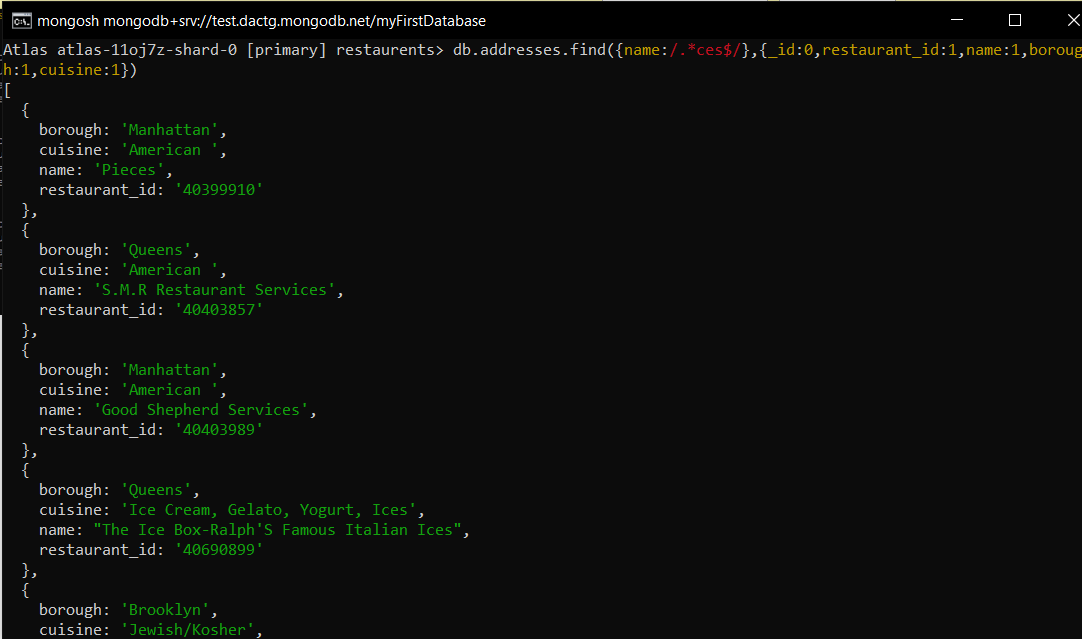
1. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.
2. db.addresses.find({$and:[{cuisine:{$ne:'American'}},{"grades.grade":'A'},{borough:{$ne:'Brooklyn'}}]}).sort({cuisine:-1})



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.
2. db.addresses.find({name:/^Wil/},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})

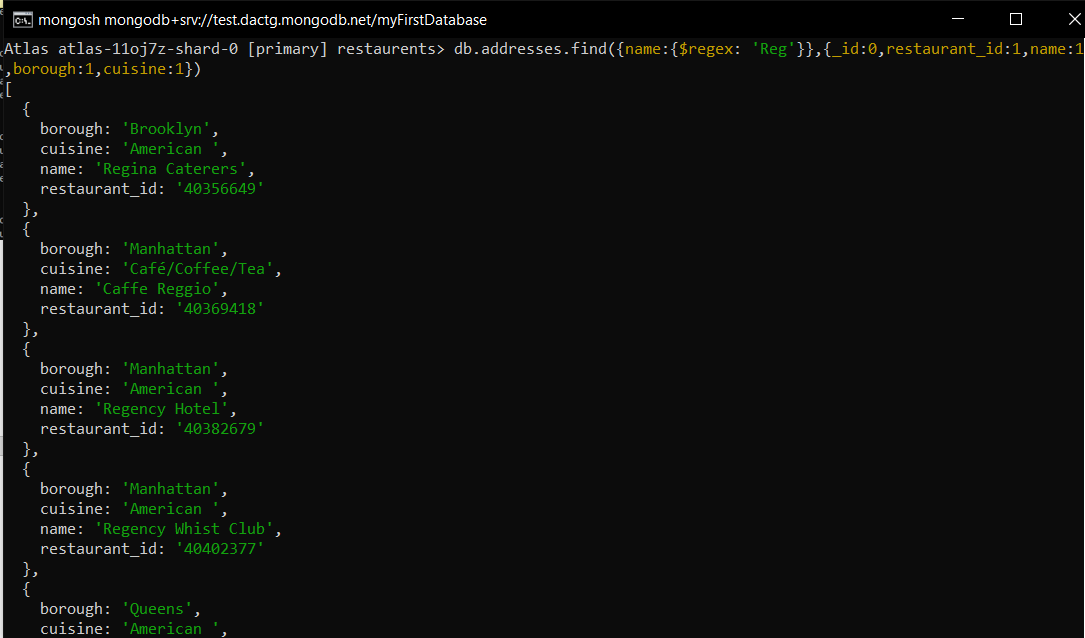


1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.
2. db.addresses.find({name:/.\*ces$/},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})



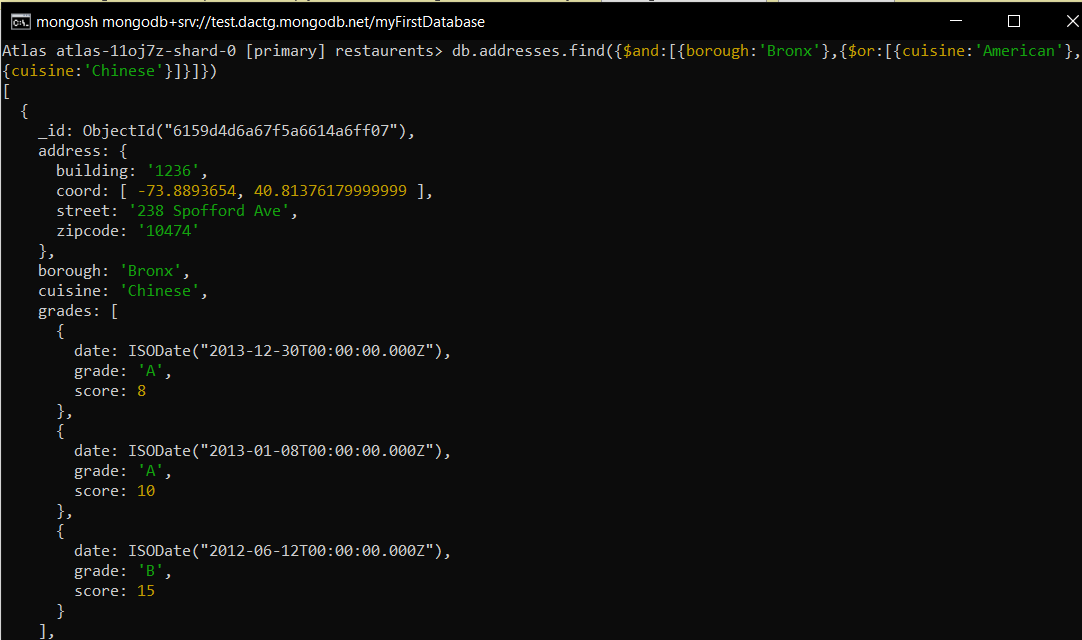
1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

* db.addresses.find({name:{$regex: 'Reg'}},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})



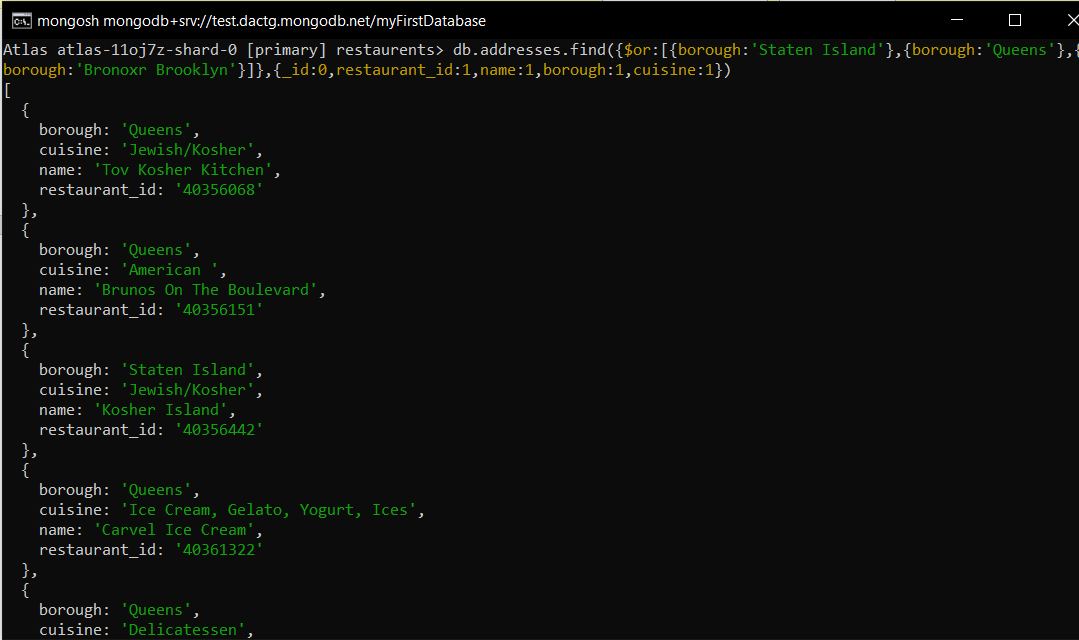
1. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

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



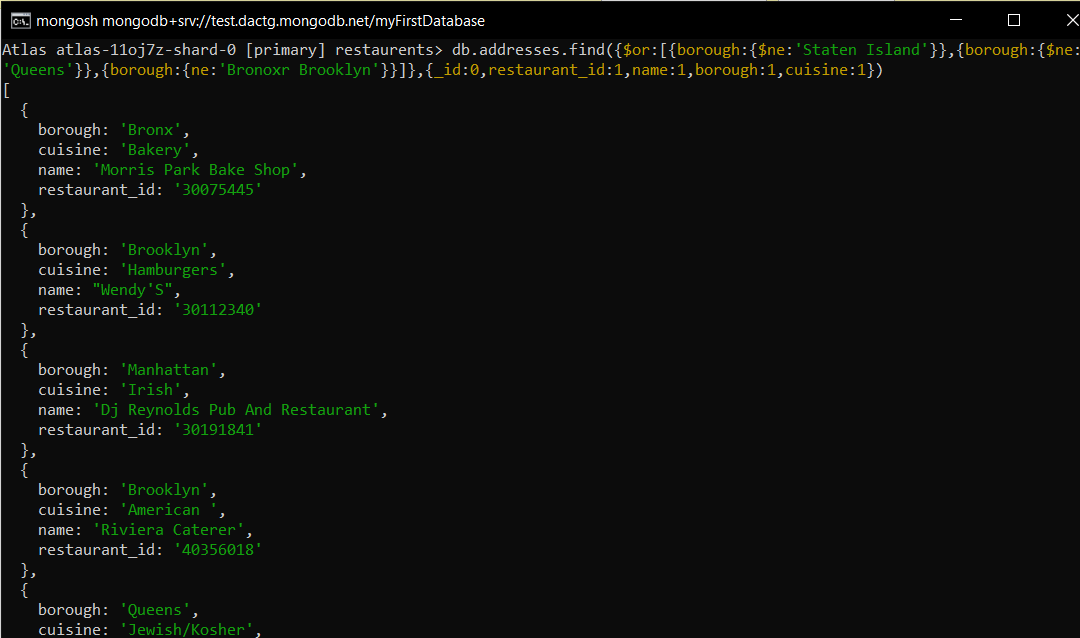
1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.

* db.addresses.find({$or:[{borough:'Staten Island'},{borough:'Queens'},{borough:'Bronoxr Brooklyn'}]},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})



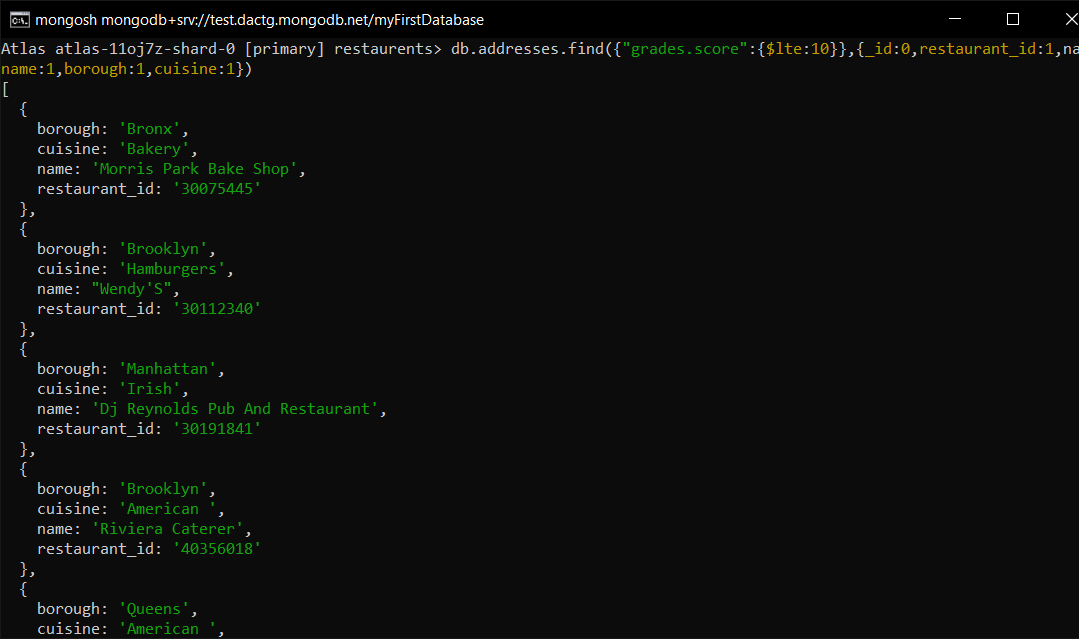
1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

* db.addresses.find({$or:[{borough:{$ne:'Staten Island'}},{borough:{$ne:'Queens'}},{borough:{ne:'Bronor Brooklyn'}}]},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:})



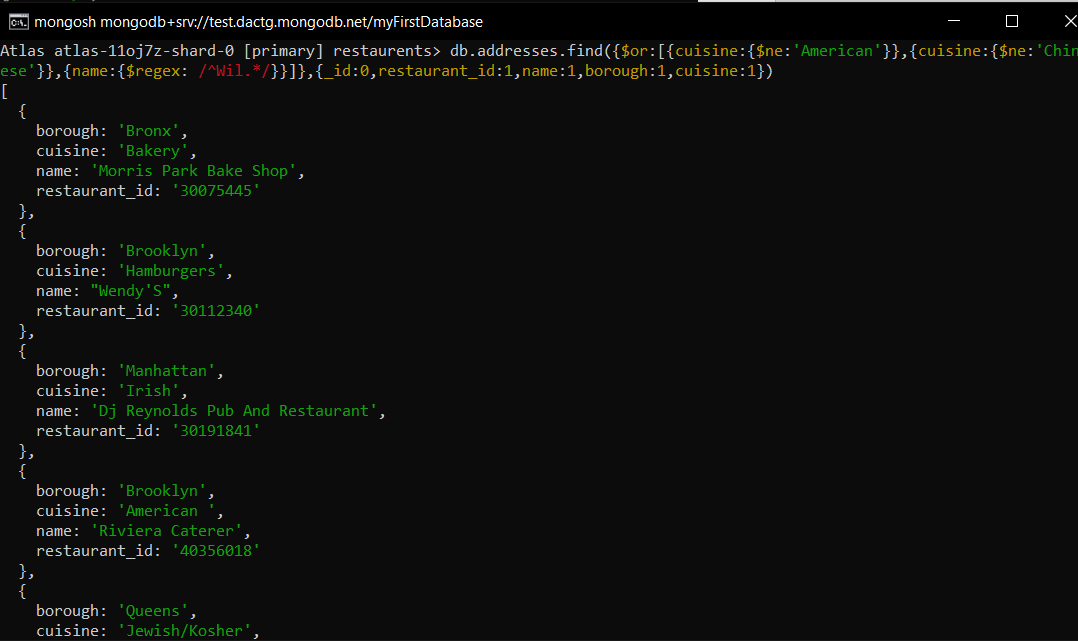
1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.

* db.addresses.find({"grades.score":{$lte:10}},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})



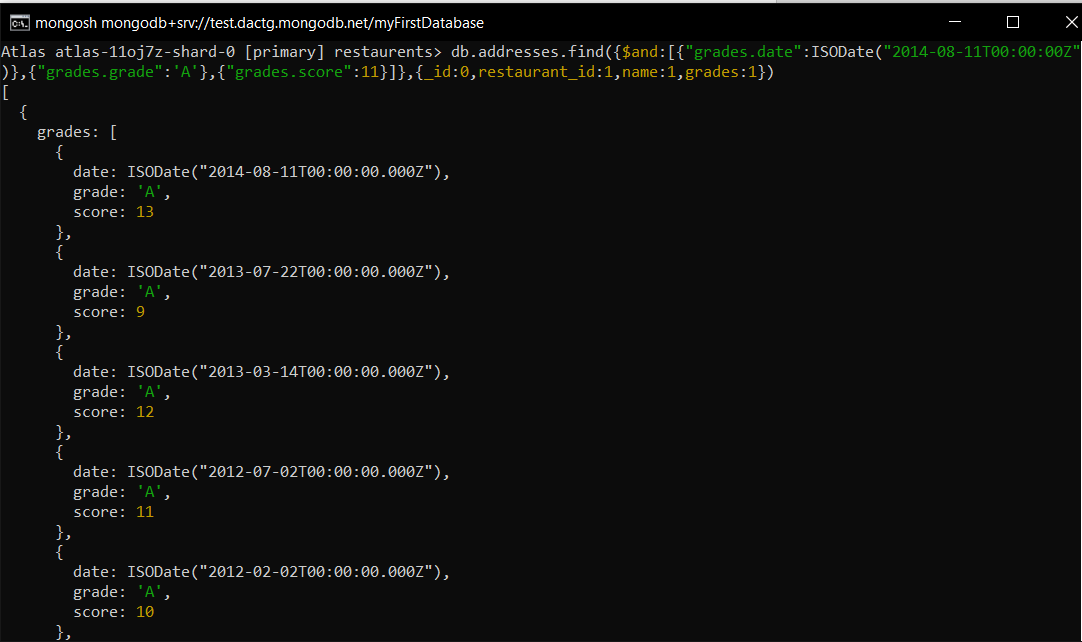
1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

* db.addresses.find({$or:[{cuisine:{$ne:'American'}},{cuisine:{$ne:'Chinese'}},{name:/^Wil.\*/}]},{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1})



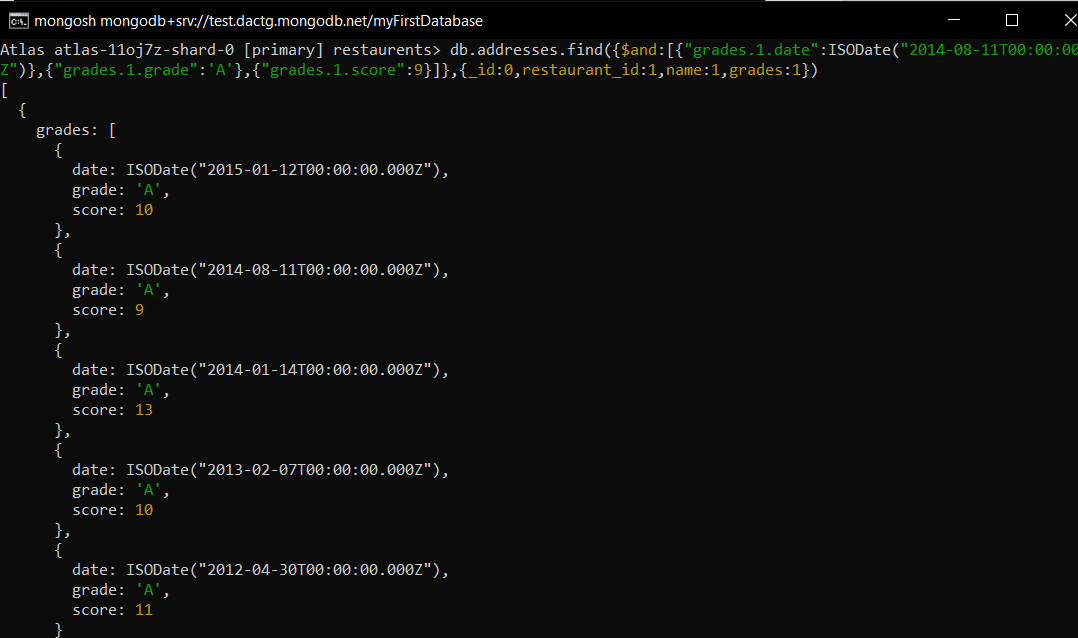
1. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates..

* db.addresses.find({$and:[{"grades.date":ISODate("2014-0811T00:00:00Z")},{"grades.grade":'A'},{"grades.score":11}]},{\_id:0,restaurant\_id:1,name:1,grades:1})



1. 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"

* db.addresses.find({$and:[{"grades.1.date":ISODate("2014-08-11T00:00:00Z")},{"grades.1.grade":'A'},{"grades.1.score":9}]},{\_id:0,restaurant\_id:1,name:1,grades:1})



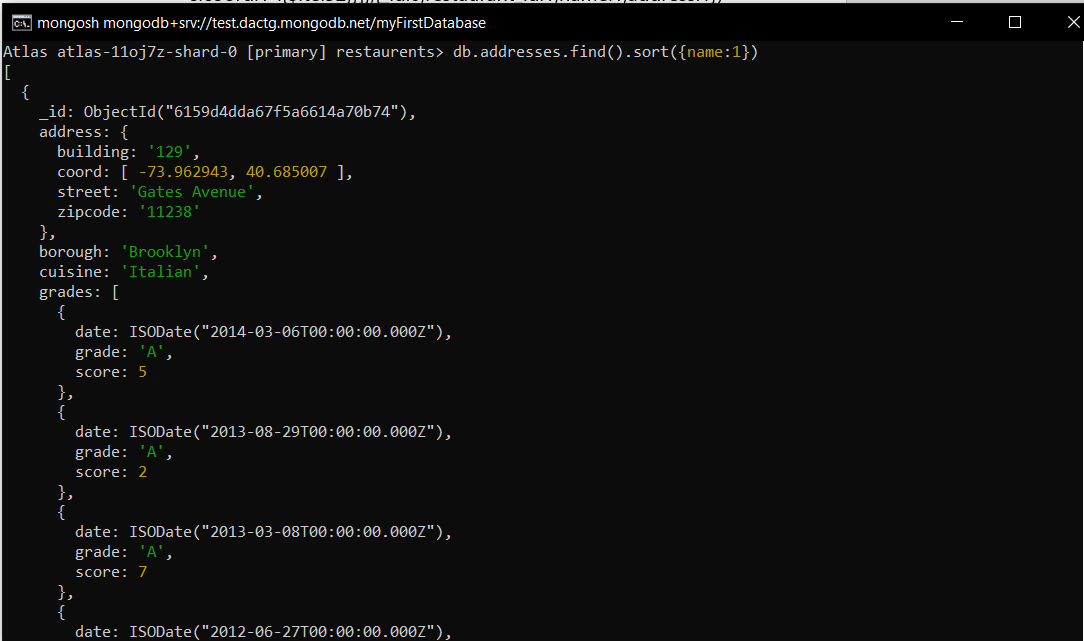
1. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52.

* db.addresses.find({$and:[{"address.coord.1":{$gt:42}},{"address.coord.1":{$lte:52}}]},{\_id:0,restaurant\_id:1,name:1,address:1})



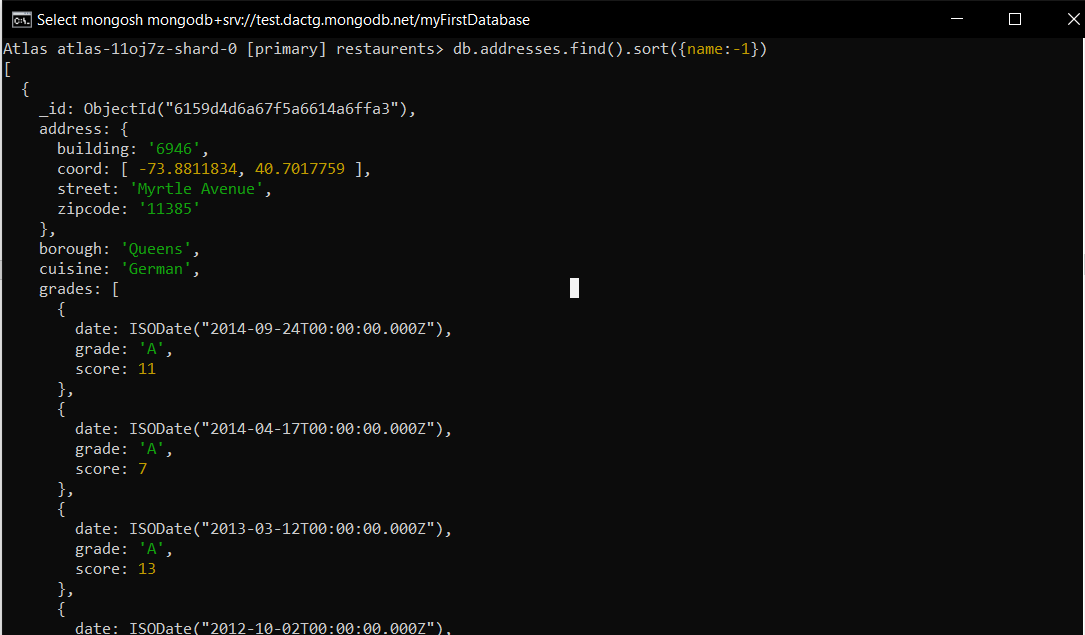
1. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

* db.addresses.find().sort({name:1})



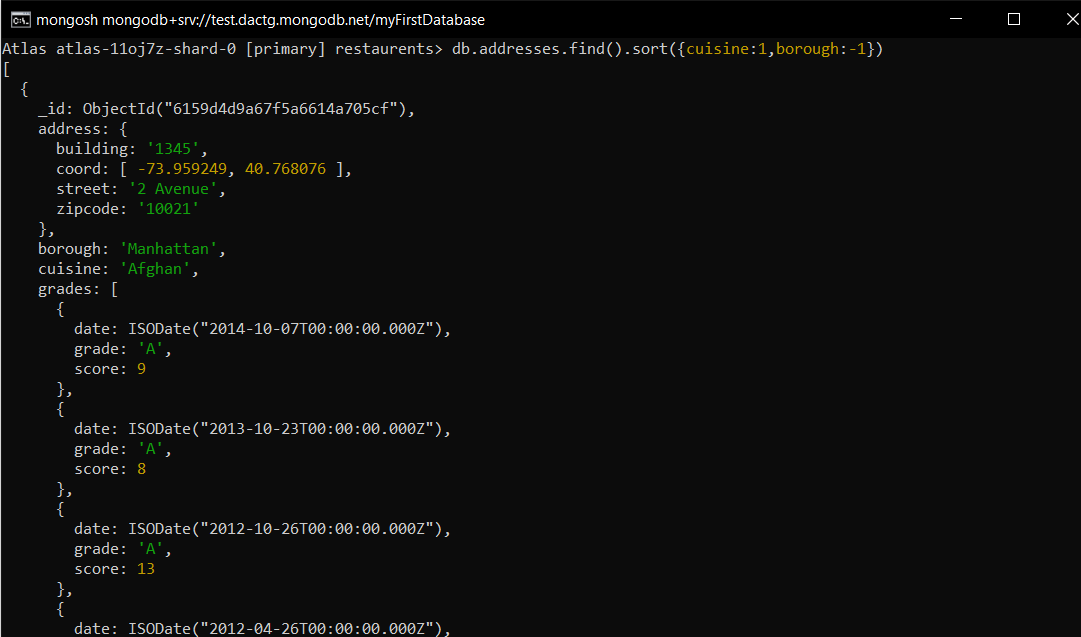
1. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

* db.addresses.find().sort({name:-1})



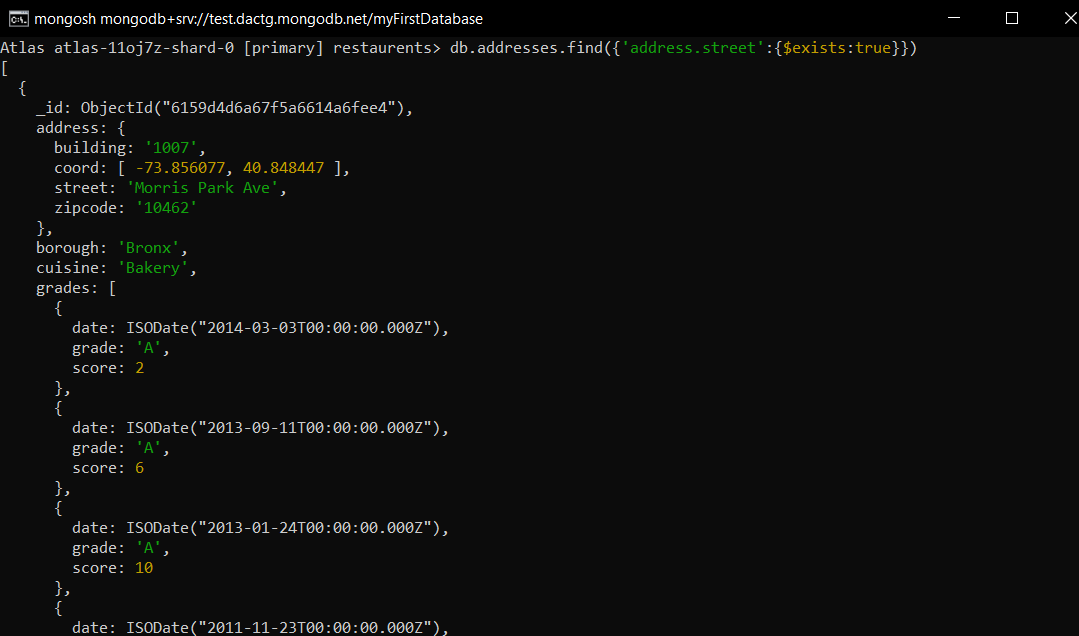
1. Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

* db.addresses.find().sort({cuisine:1,borough:-1})



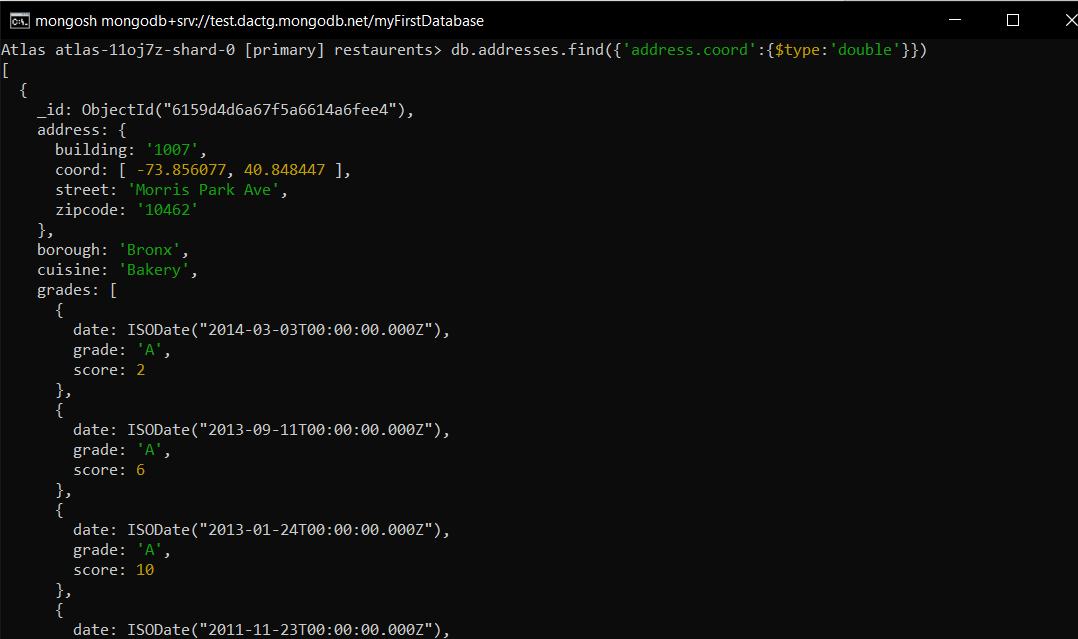
1. Write a MongoDB query to know whether all the addresses contains the street or not.

* db.addresses.find({'address.street':{$exists:true}})



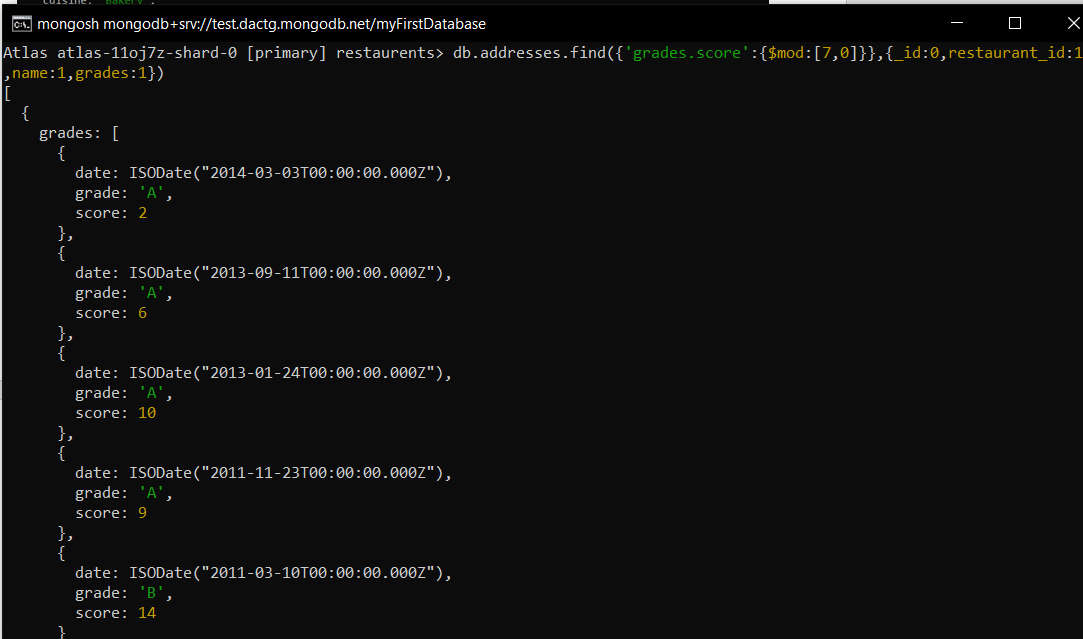
1. Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.

* db.addresses.find({'address.coord':{$type:'double'}})



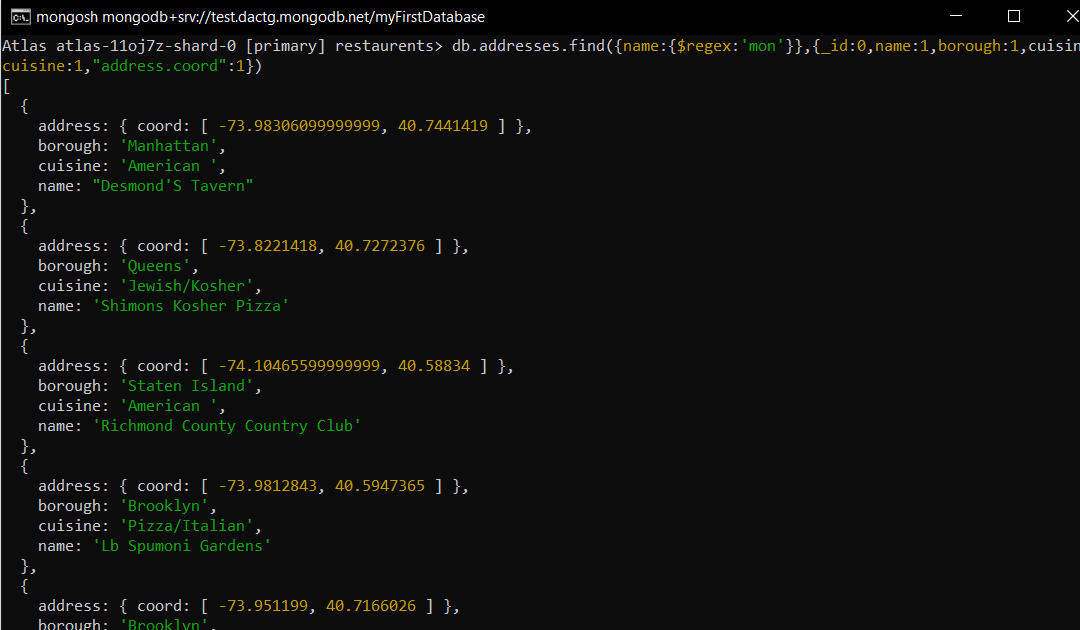
1. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

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



1. 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.

* db.addresses.find({name:{$regex:'mon'}},{\_id:0,name:1,borough:1,cuisine:1,"address.coord":1})



1. 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

* db.addresses.find({name:{$regex:/^Mad.\*/}},{\_id:0,name:1,borough:1,cuisine:1,"address.coord":1})

