DATABASE ASSIGNMENT 11

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

```
mongodb> db.restaurent.find()
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685ce'),
    address: {
      building: '469',
      coord: [ -73.961704, 40.662942 ],
      street: 'Flatbush Avenue',
      zipcode: '11225'
    },
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    grades: [
      {
        date: ISODate('2014-12-30T00:00:00.000Z'),
        grade: 'A',
        score: 8
      },
      {
        date: ISODate('2014-07-01T00:00:00.000Z'),
        grade: 'B',
        score: 23
      },
      {
        date: ISODate('2013-04-30T00:00:00.000Z'),
        grade: 'A',
        score: 12
```

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

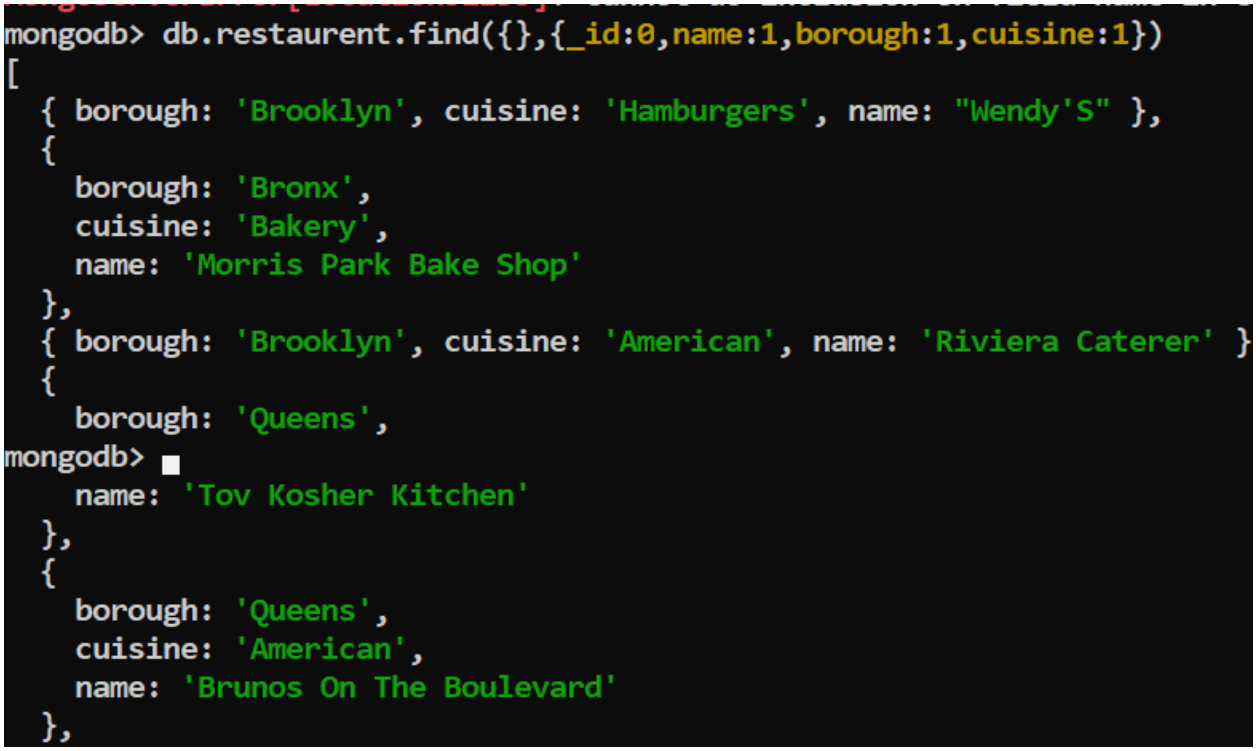
db.restaurent.find({},{name:1,borough:1,cuisine:1})

```
mongodb> db.restaurent.find({},{name:1,borough:1,cuisine:1})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685ce'),
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S"
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a685cf'),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop'
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a685d0'),
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Riviera Caterer'
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a685d1'),
    borough: 'Queens',
    cuisine: 'Jewish/Kosher',
    name: 'Tov Kosher Kitchen'
  },
```

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.restaurent.find({},{_id:0,name:1,borough:1,cuisine:1})



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.restaurent.find({},{restaurent_id:1,name:1,borough:1,zipcode:1,_id:0})

```
mongodb> db.restaurent.find({},{restaurent_id:1,name:1,borough:1,zipcode:1,_id:0})
[
  { borough: 'Brooklyn', name: "Wendy'S" },
  { borough: 'Bronx', name: 'Morris Park Bake Shop' },
  { borough: 'Brooklyn', name: 'Riviera Caterer' },
  { borough: 'Queens', name: 'Tov Kosher Kitchen' },
  { borough: 'Queens', name: 'Brunos On The Boulevard' },
  { borough: 'Staten Island', name: 'Kosher Island' },
  { borough: 'Manhattan', name: 'Dj Reynolds Pub And Restaurant' },
  { borough: 'Brooklyn', name: "Wilken'S Fine Food" },
  { borough: 'Brooklyn', name: 'Regina Caterers' },
  { borough: 'Brooklyn', name: 'May May Kitchen' },
  { borough: 'Manhattan', name: '1 East 66Th Street Kitchen' },
  { borough: 'Bronx', name: 'Wild Asia' },
  { borough: 'Brooklyn', name: 'C & C Catering Service' },
  { borough: 'Brooklyn', name: 'Seuda Foods' },
  { borough: 'Queens', name: 'Carvel Ice Cream' },
  { borough: 'Brooklyn', name: 'Carvel Ice Cream' },
  { borough: 'Brooklyn', name: 'Nordic Delicacies' },
  { borough: 'Brooklyn', name: 'Taste The Tropics Ice Cream' },
  { borough: 'Queens', name: "Sal'S Deli" },
  { borough: 'Manhattan', name: "Bully'S Deli" }
```

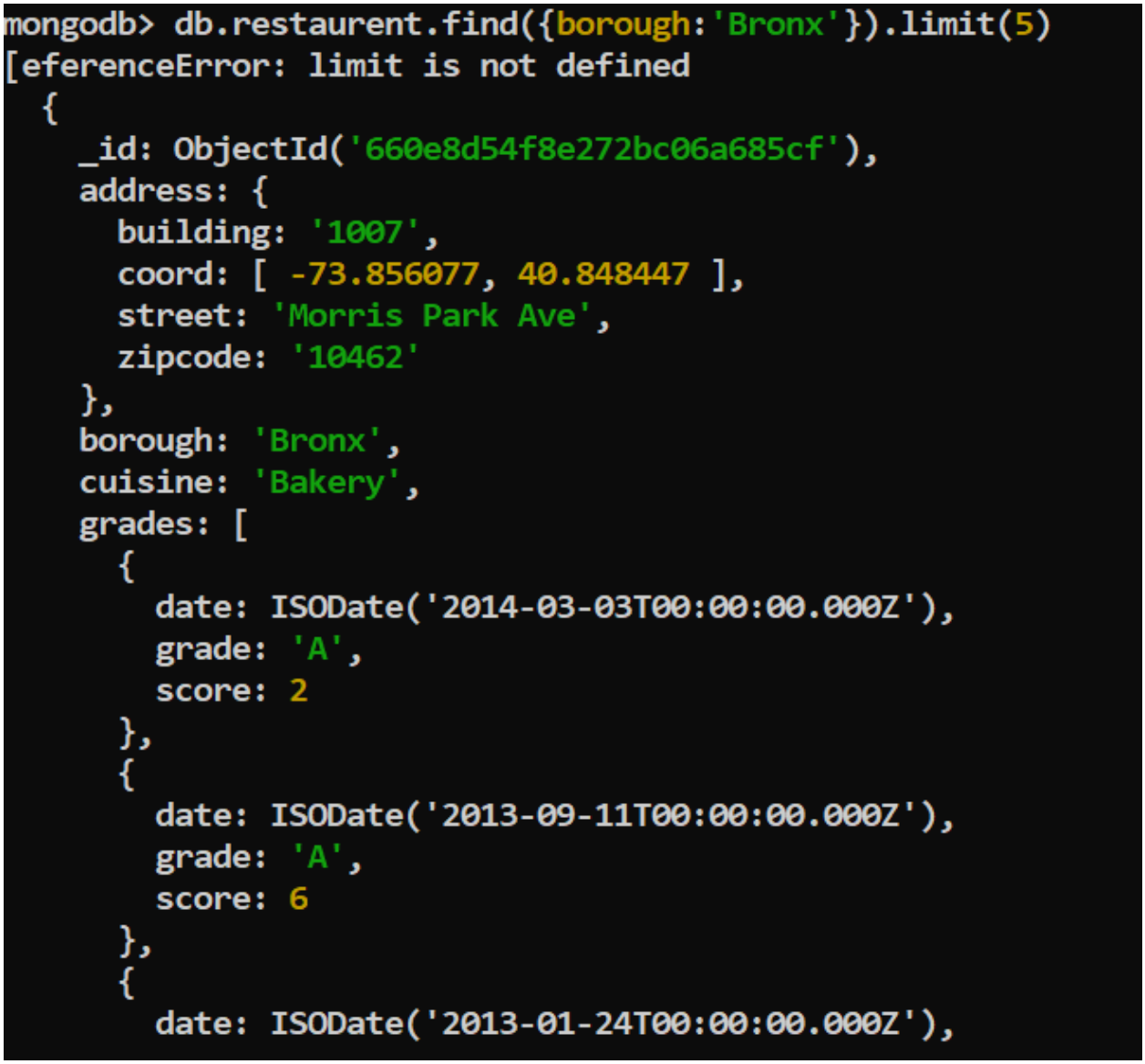
5. Write a MongoDB query to display all the restaurant which is in the borough Bronx

```
db.restaurent.find({borough:'Bronx'},{name:1,borough:1,_id:0})
```

```
mongodb> db.restaurent.find({borough:'Bronx'},{name:1,borough:1,_id:0})
[
  { borough: 'Bronx', name: 'Morris Park Bake Shop' },
  { borough: 'Bronx', name: 'Wild Asia' },
  { borough: 'Bronx', name: 'Carvel Ice Cream' },
  { borough: 'Bronx', name: 'Happy Garden' },
  { borough: 'Bronx', name: 'Happy Garden' },
  { borough: 'Bronx', name: 'Manhem Club' },
  {
    borough: 'Bronx',
    name: 'The New Starling Athletic Club Of The Bronx'
  },
  { borough: 'Bronx', name: 'Yankee Tavern' },
  { borough: 'Bronx', name: 'Mcdwyers Pub' },
  { borough: 'Bronx', name: 'The Punch Bowl' },
  { borough: 'Bronx', name: 'Munchtime' },
  { borough: 'Bronx', name: 'Ihop' },
  { borough: 'Bronx', name: "Lulu'S Coffee Shop" },
  { borough: 'Bronx', name: 'Marina Delray' },
  { borough: 'Bronx', name: "The Lark'S Nest" },
  { borough: 'Bronx', name: 'Terrace Cafe' },
```

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

db.restaurent.find({borough:'Bronx'}).limit(5)



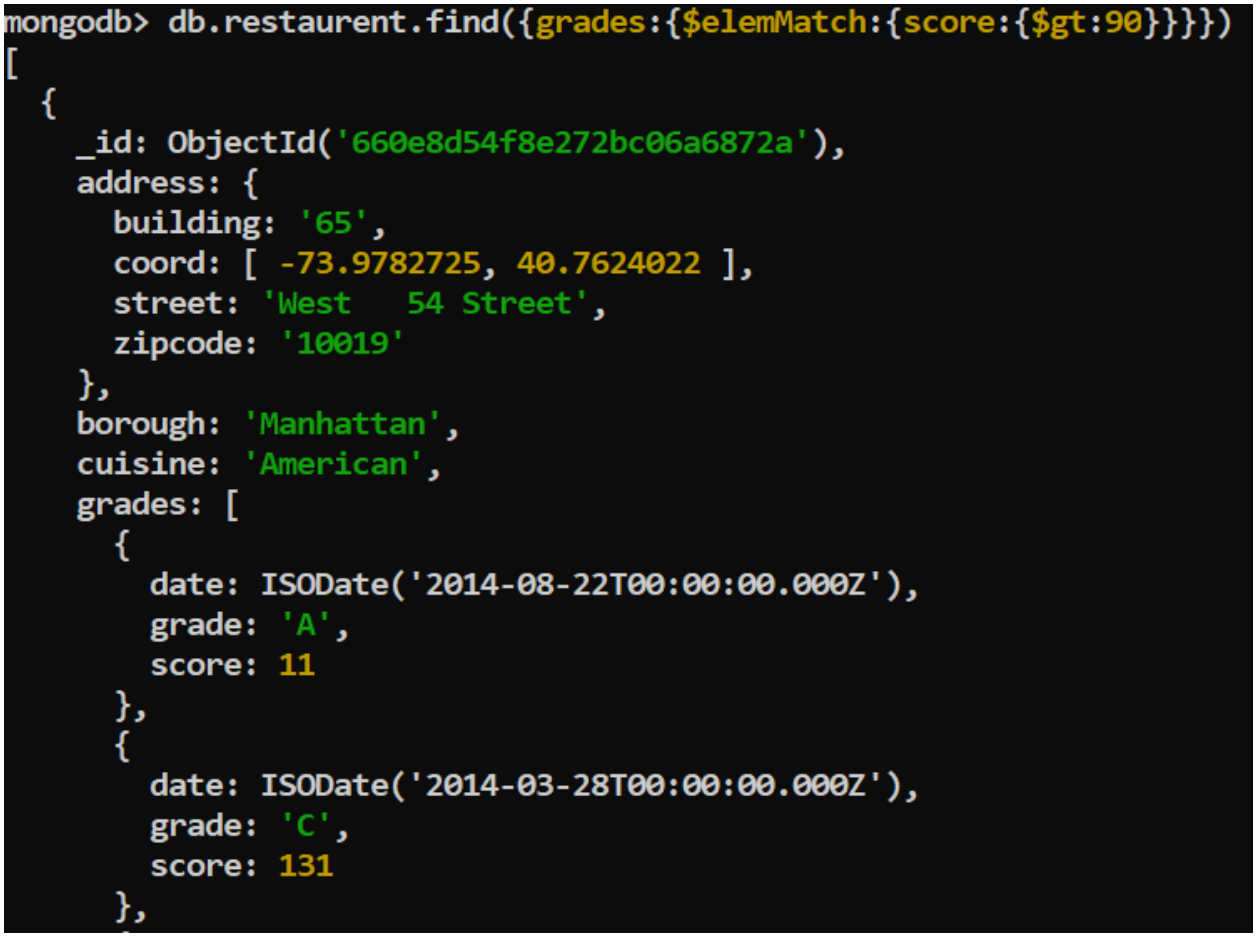
7.Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in

the borough Bronx.

```
db.restaurent.find({borough:'Bronx'}).skip(5)
```

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

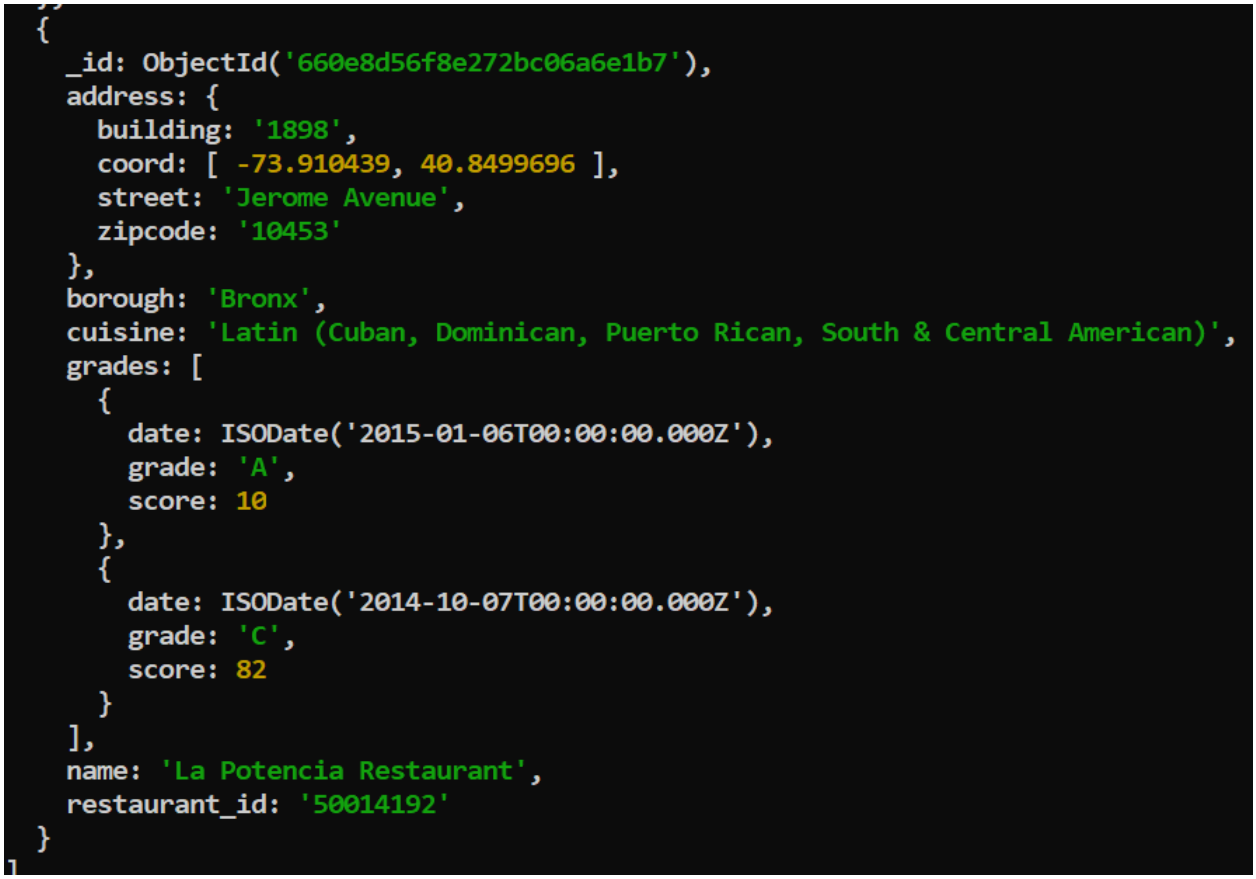
db.restaurent.find({grades:{$elemMatch:{score:{$gt:90}}}})



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

less than 100.

db.restaurent.find({grades:{$elemMatch:{score:{$gt:80,$lt:100}}}})



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

95.754168.

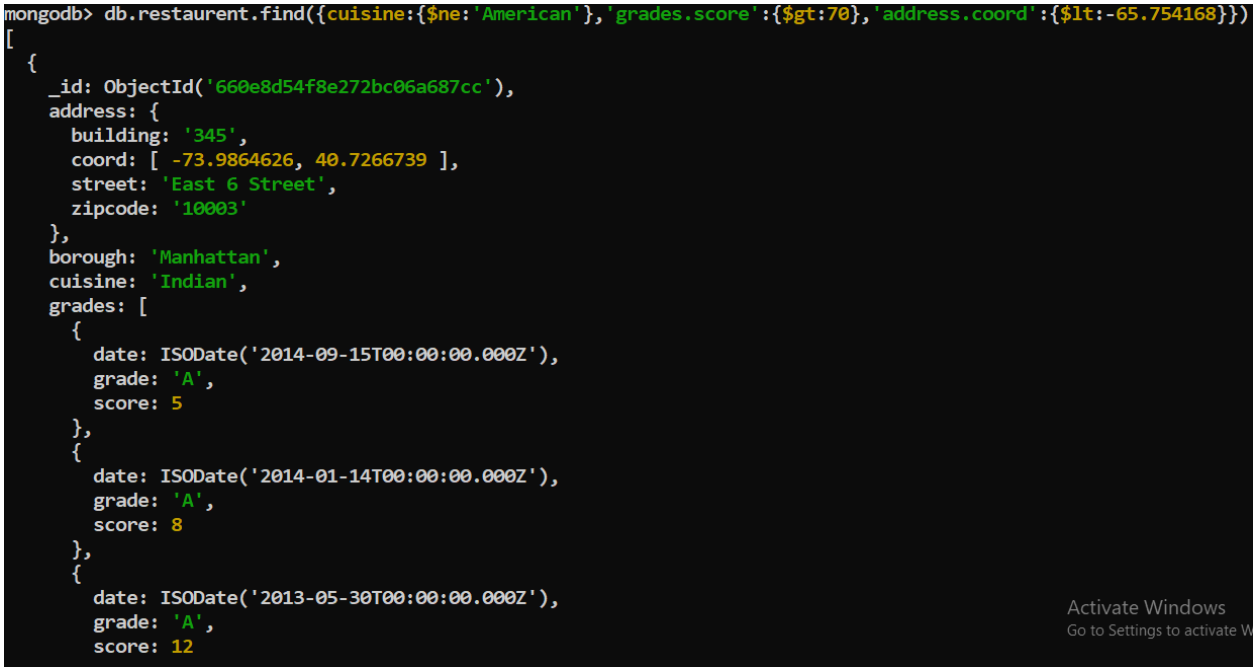
```
db.restaurent.find({'address.coord':{$lt:-95.754168}})
```

```
mongodb> db.restaurent.find({'address.coord':{$lt:-95.754168}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a68c17'),
    address: {
      building: '3707',
      coord: [ -101.8945214, 33.5197474 ],
      street: '82 Street',
      zipcode: '11372'
    },
    borough: 'Queens',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-06-04T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
      {
        date: ISODate('2013-11-07T00:00:00.000Z'),
        grade: 'B',
        score: 19
      },
      {
        date: ISODate('2013-05-17T00:00:00.000Z'),
        grade: 'A',
```

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.restaurent.find({cuisine:{$ne:'American'},'grades.score':{$gt:70},'address.coord':{$lt:-65.754168}})

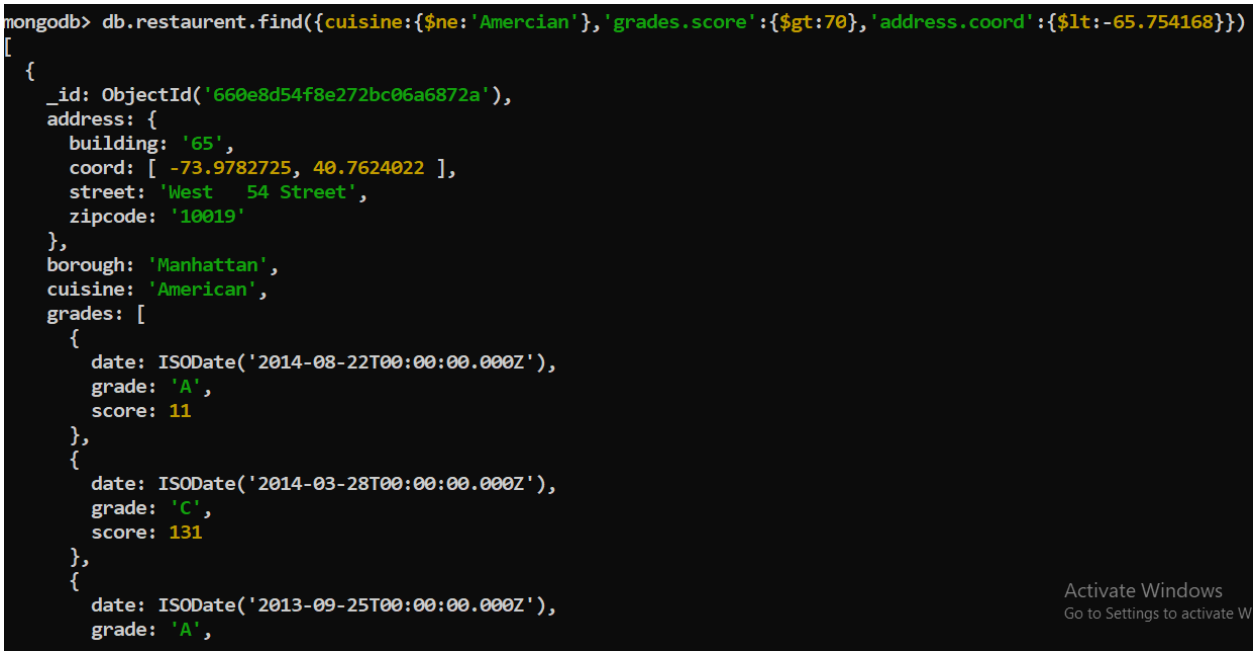


12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of

'American' and achieved a score more than 70 and located in the longitude less than -

65.754168.

db.restaurent.find({cuisine:{$ne:'Amercian'},'grades.score':{$gt:70},'address.coord':{$lt:-65.754168}})

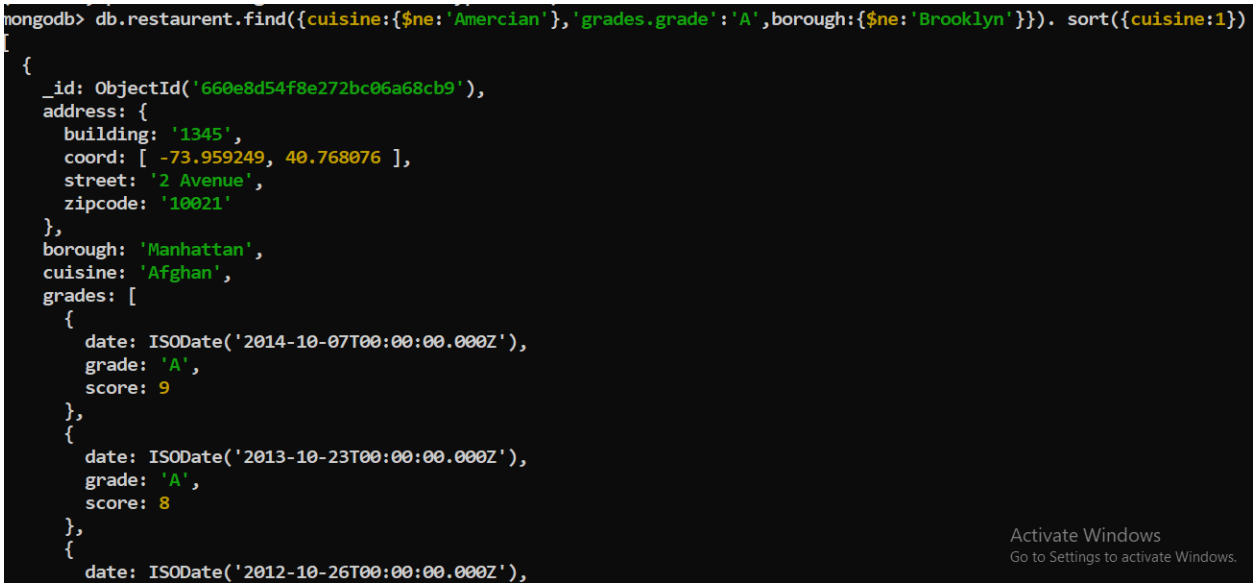


13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of

'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.

db.restaurent.find({cuisine:{$ne:'Amercian'},'grades.grade':'A',borough:{$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.

db.restaurent.find({name:/^Wil/},{restaurent_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.

db.restaurent.find({name:/ces$/},{restaurent_id:1,name:1,borough:1,cuisine:1,_id:0})

```
mongodb> db.restaurent.find({name:/ces$/},{restaurent_id:1,name:1,borough:1,cuisine:1,_id:0})
[
  { borough: 'Manhattan', cuisine: 'American', name: 'Pieces' },
  {
    borough: 'Queens',
    cuisine: 'American',
    name: 'S.M.R Restaurant Services'
  },
  {
    borough: 'Manhattan',
    cuisine: 'American',
    name: 'Good Shepherd Services'
  },
  {
    borough: 'Queens',
    cuisine: 'Ice Cream, Gelato, Yogurt, Ices',
    name: "The Ice Box-Ralph'S Famous Italian Ices"
  },
  { borough: 'Brooklyn', cuisine: 'Jewish/Kosher', name: 'Alices' },
  { borough: 'Manhattan', cuisine: 'American', name: 'Re: Sources' },
  {
    borough: 'Staten Island',
    cuisine: 'Ice Cream, Gelato, Yogurt, Ices',
    name: "Cange'S Italian Ices"
```

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.restaurent.find({name:/reg/},{restaurent_id:1,name:1,borough:1, cuisine:1,_id:0})

```
mongodb> db.restaurent.find({name:/reg/},{restaurent_id:1,name:1,borough:1,cuisine:1,_id:0})
[
  { borough: 'Bronx', cuisine: 'Pizza', name: 'Pregos Pizza' },
  {
    borough: 'Brooklyn',
    cuisine: 'American',
    name: 'Best Western Gregory Hotel'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Café/Coffee/Tea',
    name: "Gregory'S Coffee"
  },
  { borough: 'Bronx', cuisine: 'American', name: 'Zerega Avenue Deli' },
  {
    borough: 'Manhattan',
    cuisine: 'Café/Coffee/Tea',
    name: "Gregory'S Coffee"
  },
```

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and

prepared either American or Chinese dish.

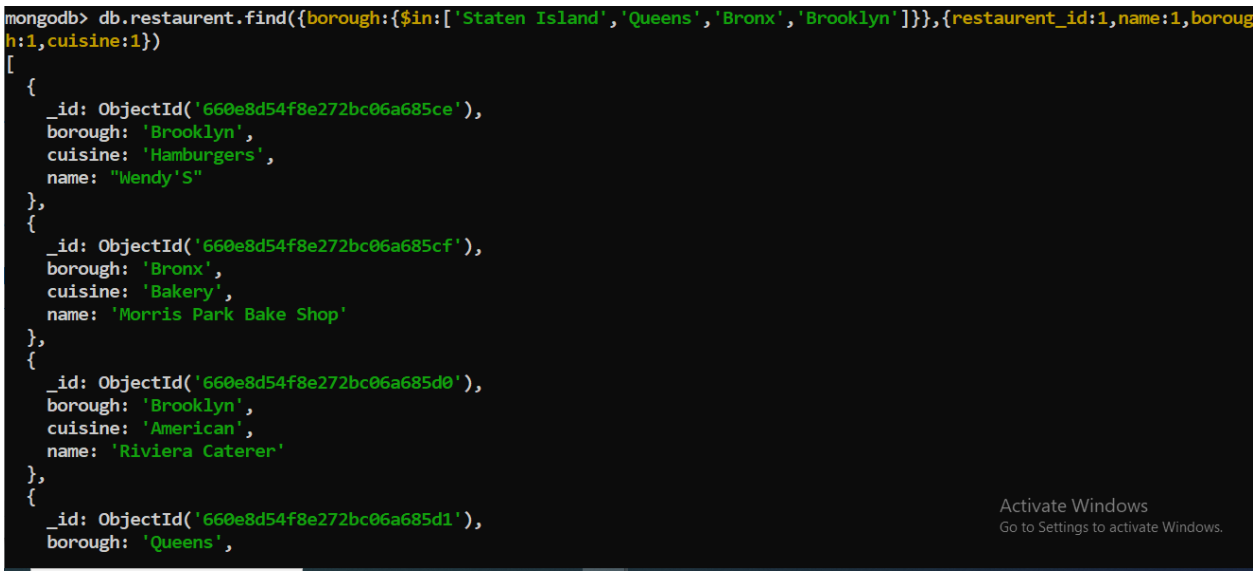
db.restaurent.find({borough:'Bronx',cuisine:{$in:['American','Chinese']}})

```
mongodb> db.restaurent.find({borough:'Bronx',cuisine:{$in:['American','Chinese']}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685d9'),
    address: {
      building: '2300',
      coord: [ -73.8786113, 40.8502883 ],
      street: 'Southern Boulevard',
      zipcode: '10460'
    },
    borough: 'Bronx',
    cuisine: 'American',
    grades: [
      {
        date: ISODate('2014-05-28T00:00:00.000Z'),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate('2013-06-19T00:00:00.000Z'),
        grade: 'A',
        score: 4
      },
      {
        date: ISODate('2012-06-15T00:00:00.000Z'),
        grade: 'A',
```

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.

db.restaurent.find({borough:{$in:['Staten Island','Queens','Bronx','Brooklyn']}},{restaurent_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.

db.restaurent.find({borough:{$nin:['Staten Island','Queens','Bronx','Brooklyn']}},{restaurent_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.restaurent.find({grades:{$elemMatch:{score:{$lt:10}}}})

```
mongodb> db.restaurent.find({grades:{$elemMatch:{score:{$lt:10}}}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685ce'),
    address: {
      building: '469',
      coord: [ -73.961704, 40.662942 ],
      street: 'Flatbush Avenue',
      zipcode: '11225'
    },
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    grades: [
      {
        date: ISODate('2014-12-30T00:00:00.000Z'),
        grade: 'A',
        score: 8
      },
      {
        date: ISODate('2014-07-01T00:00:00.000Z'),
        grade: 'B',
        score: 23
      },
      {
        date: ISODate('2013-04-30T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
```

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'.

db.restaurent.find({$or:[{name:/^wil/i},{cuisine:{$nin:['Amercian','Chinese']}}]},{_id:0,name:1,borough:1,cuisine:1});

```
mongodb> db.restaurent.find({$or:[{name:/^wil/i},{cuisine:{$nin:['Amercian','Chinese']}}]},{_id:0,name:1,borough:1,cuisine
:1});
[
  { borough: 'Brooklyn', cuisine: 'Hamburgers', name: "Wendy'S" },
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop'
  },
  { borough: 'Brooklyn', cuisine: 'American', name: 'Riviera Caterer' },  {
    borough: 'Queens',
    cuisine: 'Jewish/Kosher',
    name: 'Tov Kosher Kitchen'
  },
  {
    borough: 'Queens',
    cuisine: 'American',
    name: 'Brunos On The Boulevard'
  },
  {
mongodb> ■
    cuisine: 'Jewish/Kosher',
    name: 'Kosher Island'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Irish',
```

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

db.restaurent.find({'grades.grade':'A','grades.score':11,'grades.date':ISODate('2014-08-11T00:00:00Z')},{restaurent_id:1,name:1,grade:1})

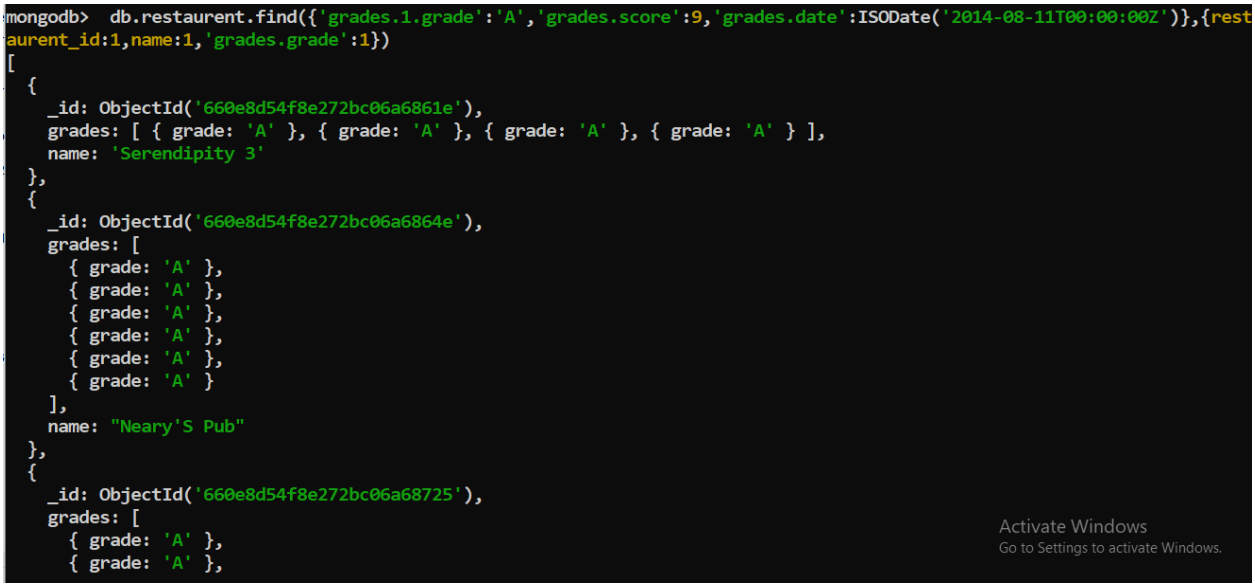
```
mongodb> db.restaurent.find({'grades.grade':'A','grades.score':11,'grades.date':ISODate('2014-08-11T00:00:00Z')},{restaur
restaurent_id:1,name:1,grade:1})
[
  { _id: ObjectId('660e8d54f8e272bc06a6864e'), name: "Neary'S Pub" },
  {
    _id: ObjectId('660e8d54f8e272bc06a68725'),
    name: 'Don Filippo Restaurant'
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a689ec'),
    name: "Mustang Sally'S Restaurant"
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a68bf9'),
    name: 'Club Macanudo (Cigar Bar)'
  },
  {
    _id: ObjectId('660e8d54f8e272bc06a68cec'),
    name: "Marino'S Pizza & Restaurant"
  },
```

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".

db.restaurent.find({'grades.1.grade':'A','grades.score':9,'grades.date': ISODate('2014-08-11T00:00:00Z')},{restaurent_id:1,name:1,'grades.grade':1})

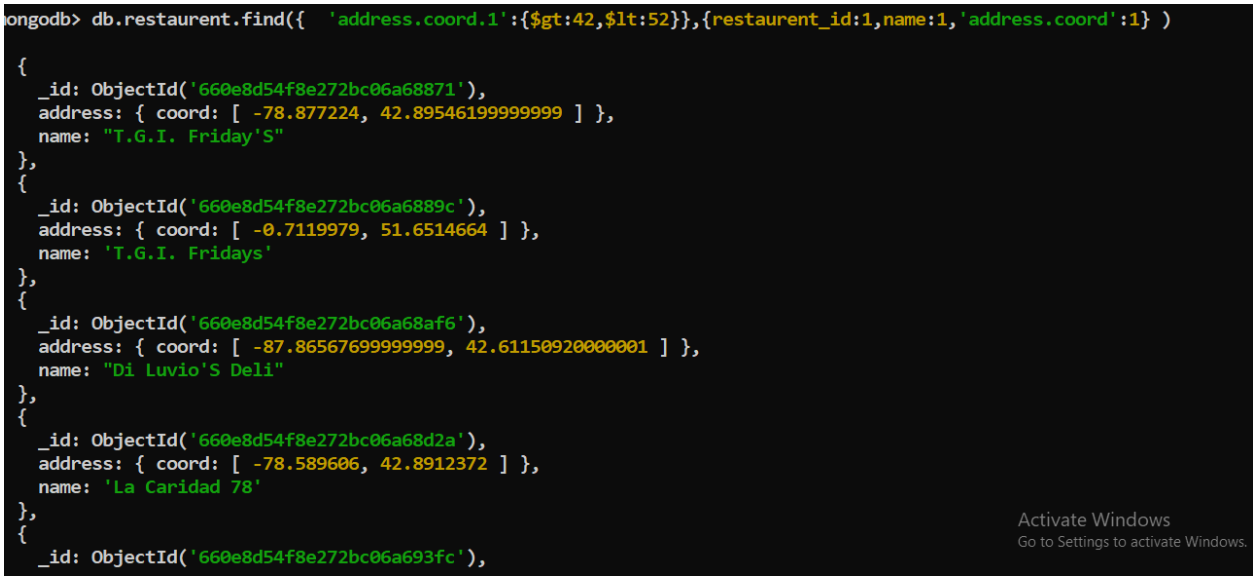


24. Write a MongoDB query to find the restaurant Id, name, address and geographical

location for those restaurants where 2nd element of coord array contains a value which is

more than 42 and upto 52

db.restaurent.find({ 'address.1.coord':{$gt:42,$lt:52},{restaurent_id:1,name:1,'address.coord':1} })



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

with all the columns.

db.restaurent.find().sort({name:1})

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

all the columns.

db.restaurent.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.

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

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

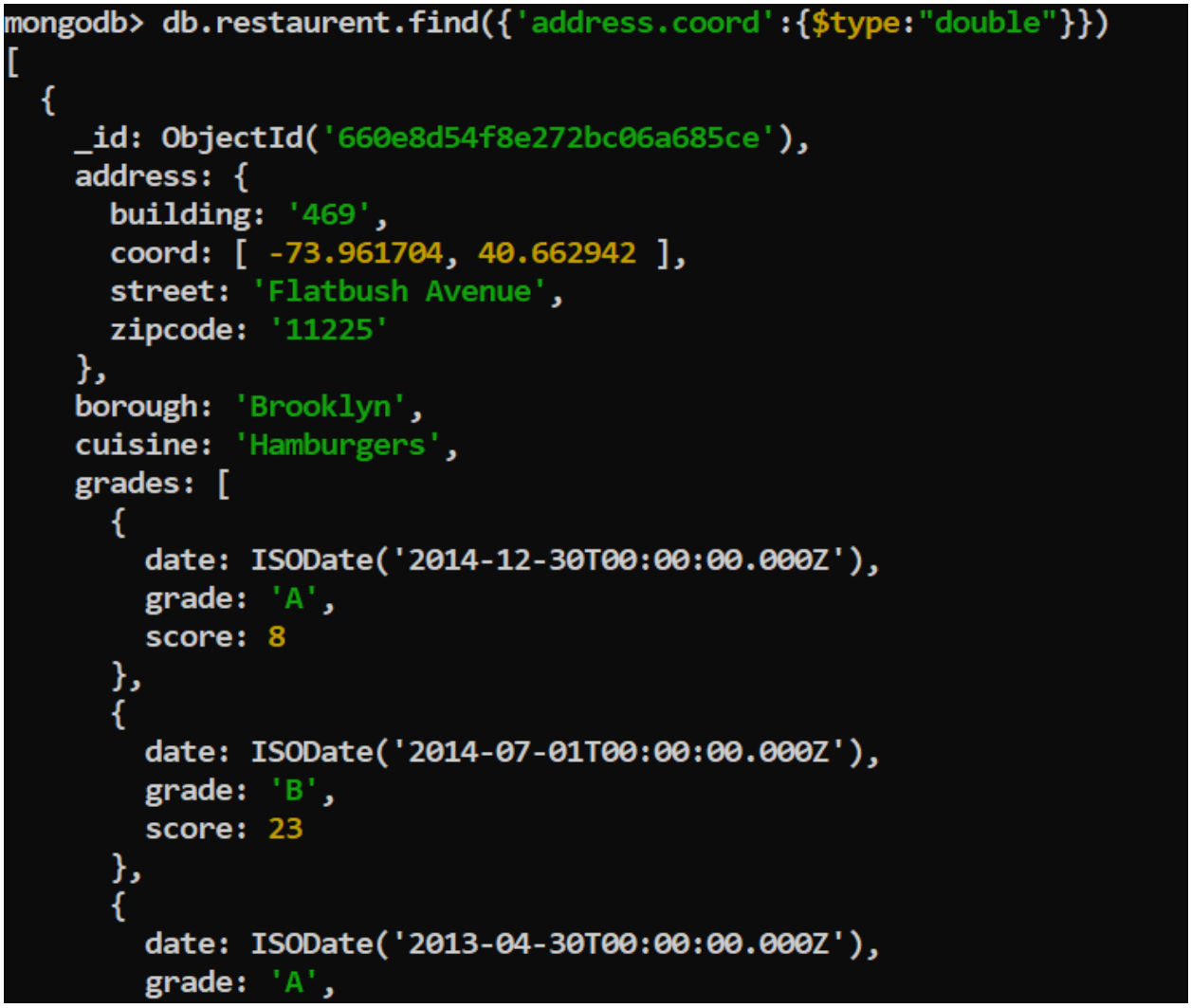
db.restaurent.find({'address.street':{$exist:'true'}})

```
mongodb>  db.restaurent.find({'address.street':{$exists:'true'}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685ce'),
    address: {
      building: '469',
      coord: [ -73.961704, 40.662942 ],
      street: 'Flatbush Avenue',
      zipcode: '11225'
    },
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    grades: [
      {
        date: ISODate('2014-12-30T00:00:00.000Z'),
        grade: 'A',
        score: 8
      },
      {
        date: ISODate('2014-07-01T00:00:00.000Z'),
        grade: 'B',
        score: 23
      },
      {
        date: ISODate('2013-04-30T00:00:00.000Z'),
        grade: 'A',
```

29. Write a MongoDB query which will select all documents in the restaurants collection

where the coord field value is Double.

db.restaurent.find({'address.coord':{$type:"Double"}})



```
mongodb> db.restaurent.find({'address.coord':{$type:"double"}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685ce'),
    address: {
      building: '469',
      coord: [ -73.961704, 40.662942 ],
      street: 'Flatbush Avenue',
      zipcode: '11225'
    },
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    grades: [
      {
        date: ISODate('2014-12-30T00:00:00.000Z'),
        grade: 'A',
        score: 8
      },
      {
        date: ISODate('2014-07-01T00:00:00.000Z'),
        grade: 'B',
        score: 23
      },
      {
        date: ISODate('2013-04-30T00:00:00.000Z'),
        grade: 'A',
```

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.

db.restaurent.find({'grades.score':{$mod:[7,0]}})

```
mongodb> db.restaurent.find({'grades.score':{$mod:[7,0]}})
[
  {
    _id: ObjectId('660e8d54f8e272bc06a685cf'),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
```

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.

db.restaurent.find({name:/mon/},{restaurent_id:1,name:1,borough:1,cuisine:1,'address.coord':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.

db.restaurent.find({name:/^Mad/},{restaurent_id:1,name:1,borough:1,cuisine:1,'address.coord':1})

```
mongodb>  db.restaurent.find({name:/^Mad/},{restaurent_id:1,name:1,borough:1,cuisine:1,'address.coord':1})
[
  {
    _id: ObjectId('660e8d55f8e272bc06a6a565'),
    address: { coord: [ -74.0103118, 40.7042077 ] },
    borough: 'Manhattan',
    cuisine: 'Mexican',
    name: 'Mad Dog & Beans'
  },
  {
    _id: ObjectId('660e8d56f8e272bc06a6d7f9'),
    address: { coord: [ -73.99346899999999, 40.682505 ] },
    borough: 'Brooklyn',
    cuisine: 'Mexican',
    name: 'Mad Dog & Beans'
  },
  {
    _id: ObjectId('660fb5e56806c1ca84ff58bb'),
    address: { coord: [ -74.0103118, 40.7042077 ] },
    borough: 'Manhattan',
    cuisine: 'Mexican',
    name: 'Mad Dog & Beans'
  },
  {
    _id: ObjectId('660fb5e66806c1ca84ff8b50'),
    address: { coord: [ -73.99346899999999, 40.682505 ] },
    borough: 'Brooklyn',
    cuisine: 'Mexican',
```