Assignmnet_1

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

Ans: db.restaurant.find().pretty()

```
> show collections
< Golden nest
  Marriot
  restaurants
> db.restaurants.find();
< { _id: ObjectId("635166ca6473e4c38285b606"),
    address:
     { building: '1007',
       coord: [ -73.856077, 40.848447 ],
       street: 'Morris Park Ave',
       zipcode: '10462' },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades:
     [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
```

2. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all.

the documents in the collection restaurant.

Ans: db.restaurant.find({},{restaurant_id:1,name:1,borough:1,cuisine:1})

```
> db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"address.zipcode" :1,"_id":0});
< { address: { zipcode: '10462' },
    borough: 'Bronx',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445' }
  { address: { zipcode: '11225' },
    borough: 'Brooklyn',
    name: 'Wendy\'S',
    restaurant_id: '30112340' }
  { address: { zipcode: '10019' },
    borough: 'Manhattan',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841' }
```

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.

Ans: db.restaurant.find({},{restaurant_id:1,name:1,_id:0,borough:1,cuisine:1})

```
> db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"address.zipcode" :1,"_id":0});
{ address: { zipcode: '10462' },
  borough: 'Bronx',
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445' }
{ address: { zipcode: '11225' },
  borough: 'Brooklyn',
  name: 'Wendy\'S',
  restaurant_id: '30112340' }
{ address: { zipcode: '10019' },
  borough: 'Manhattan',
  name: 'Dj Reynolds Pub And Restaurant',
  restaurant_id: '30191841' }
```

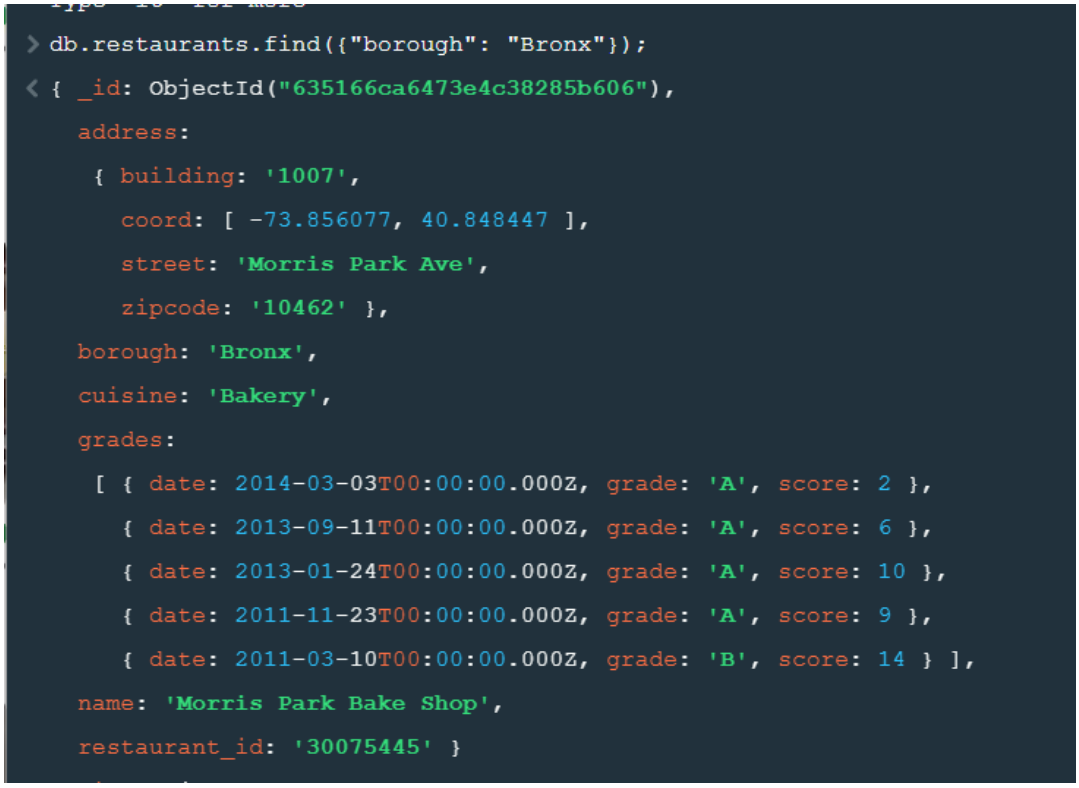
4. Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

Ans: db.restaurant.find({},{address:{zipcode:1},restaurant_id:1,name:1,_id:0})

```
Type "it" for more
> db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"address.zipcode" :1,"_id":0});
{ address: { zipcode: '10462' },
  borough: 'Bronx',
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445' }
{ address: { zipcode: '11225' },
  borough: 'Brooklyn',
  name: 'Wendy\'S',
  restaurant_id: '30112340' }
```

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

Ans: db.restaurant.find({borough:'Bronx'})



```
> db.restaurants.find({"borough": "Bronx"});
{ _id: ObjectId("635166ca6473e4c38285b606"),
  address:
   { building: '1007',
     coord: [ -73.856077, 40.848447 ],
     street: 'Morris Park Ave',
     zipcode: '10462' },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades:
   [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
     { date: 2013-09-11T00:00:00.000Z, grade: 'A', score: 6 },
     { date: 2013-01-24T00:00:00.000Z, grade: 'A', score: 10 },
     { date: 2011-11-23T00:00:00.000Z, grade: 'A', score: 9 },
     { date: 2011-03-10T00:00:00.000Z, grade: 'B', score: 14 } ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445' }
```

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

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

```
> db.restaurants.find({"borough": "Bronx"}).limit(5);
< { _id: ObjectId("635166ca6473e4c38285b606"),
    address:
     { building: '1007',
       coord: [ -73.856077, 40.848447 ],
       street: 'Morris Park Ave',
       zipcode: '10462' },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades:
     [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
       { date: 2013-09-11T00:00:00.000Z, grade: 'A', score: 6 },
       { date: 2013-01-24T00:00:00.000Z, grade: 'A', score: 10 },
       { date: 2011-11-23T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2011-03-10T00:00:00.000Z, grade: 'B', score: 14 } ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445' }
```

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

Ans: db.restaurant.find({borough:'Bronx'}).limit(5).skip(5)

```
> db.restaurants.find({"borough": "Bronx"}).skip(5).limit(5);
< { _id: ObjectId("635166ca6473e4c38285b643"),
    address:
     { building: '658',
       coord: [ -73.81363999999999, 40.82941100000001 ],
       street: 'Clarence Ave',
       zipcode: '10465' },
    borough: 'Bronx',
    cuisine: 'American ',
    grades:
     [ { date: 2014-06-21T00:00:00.000Z, grade: 'A', score: 5 },
       { date: 2012-07-11T00:00:00.000Z, grade: 'A', score: 10 } ],
    name: 'Manhem Club',
    restaurant_id: '40364363' }
  { _id: ObjectId("635166ca6473e4c38285b65b"),
    address:
     { building: '2222',
       coord: [ -73.84971759999999, 40.8304811 ],
```

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

Ans: db.restaurant.find({'grades.score':{$gt:90}}).pretty()

```
> db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});
< { _id: ObjectId("635166ca6473e4c38285b764"),
    address:
     { building: '65',
       coord: [ -73.9782725, 40.7624022 ],
       street: 'West   54 Street',
       zipcode: '10019' },
    borough: 'Manhattan',
    cuisine: 'American ',
    grades:
     [ { date: 2014-08-22T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2014-03-28T00:00:00.000Z, grade: 'C', score: 131 },
       { date: 2013-09-25T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2013-04-08T00:00:00.000Z, grade: 'B', score: 25 },
       { date: 2012-10-15T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2011-10-19T00:00:00.000Z, grade: 'A', score: 13 } ],
    name: 'Murals On 54/Randolphs\'S',
    restaurant_id: '40372466' }
```

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

Ans: db.restaurant.find({$and:[{'grades.score':{$gt:90}},{'grades.score':{$lt:100}}]}).pretty()

```
> db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});
{ _id: ObjectId("635166ca6473e4c38285b764"),
    address:
     { building: '65',
       coord: [ -73.9782725, 40.7624022 ],
       street: 'West   54 Street',
       zipcode: '10019' },
    borough: 'Manhattan',
    cuisine: 'American ',
    grades:
     [ { date: 2014-08-22T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2014-03-28T00:00:00.000Z, grade: 'C', score: 131 },
       { date: 2013-09-25T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2013-04-08T00:00:00.000Z, grade: 'B', score: 25 },
       { date: 2012-10-15T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2011-10-19T00:00:00.000Z, grade: 'A', score: 13 } ],
    name: 'Murals On 54/Randolphs\'S',
```

Assignmnet_2

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

Ans: db.restaurant.find({'address.coord.0':{$lt:-95.754168}}).pretty()

```
> db.restaurants.find({"address.coord" : {$lt : -95.754168}});
< { _id: ObjectId("635166ca6473e4c38285bc4e"),
    address:
     { building: '3707',
       coord: [ -101.8945214, 33.5197474 ],
       street: '82 Street',
       zipcode: '11372' },
    borough: 'Queens',
    cuisine: 'American ',
    grades:
     [ { date: 2014-06-04T00:00:00.000Z, grade: 'A', score: 12 },
       { date: 2013-11-07T00:00:00.000Z, grade: 'B', score: 19 },
       { date: 2013-05-17T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2012-08-29T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2012-04-03T00:00:00.000Z, grade: 'A', score: 12 },
       { date: 2011-11-16T00:00:00.000Z, grade: 'A', score: 7 } ],
    name: 'Burger King',
    restaurant_id: '40534067' }
```

2. 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.

Ans: db.restaurant.find({$and:[{cuisine:{$ne:'American '}},{'address.coord.0':{$lt:-65.754168}}]}).pretty()

```
> db.restaurants.find(
                {$and:
                     [
                        {"cuisine" : {$ne :"American "}},
                        {"grades.score" : {$gt : 70}},
                        {"address.coord" : {$lt : -65.754168}}
                     ]
                 }
                     );
< { _id: ObjectId("635166ca6473e4c38285b805"),
    address:
     { building: '345',
       coord: [ -73.9864626, 40.7266739 ],
       street: 'East 6 Street',
       zipcode: '10003' },
```

3. 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 : -65.754168}
                               }
                         );
< { _id: ObjectId("635166ca6473e4c38285b805"),
    address:
     { building: '345',
       coord: [ -73.9864626, 40.7266739 ],
       street: 'East 6 Street',
       zipcode: '10003' },
    borough: 'Manhattan',
```

4. 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.restaurant.find({name:/^Wil/},{restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()

```
> db.restaurants.find(
  {name: /^Wil/},
  {
  "restaurant_id" : 1,
  "name":1,"borough":1,
  "cuisine" :1
  }
  );
< { _id: ObjectId("635166ca6473e4c38285b60d"),
    borough: 'Brooklyn',
    cuisine: 'Delicatessen',
    name: 'Wilken\'S Fine Food',
    restaurant_id: '40356483' }
  { _id: ObjectId("635166ca6473e4c38285b610"),
```

5. 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.

Ans: db.restaurant.find({name:/ces$/},{restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()

```
> db.restaurants.find(
  {name: /ces$/},
  {
  "restaurant_id" : 1,
  "name":1,"borough":1,
  "cuisine" :1
  }
  );
< { _id: ObjectId("635166ca6473e4c38285ba99"),
    borough: 'Manhattan',
    cuisine: 'American ',
    name: 'Pieces',
    restaurant_id: '40399910' }
  { _id: ObjectId("635166ca6473e4c38285bb58"),
    borough: 'Queens',
    cuisine: 'American ',
```

6. 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.

Ans: db.restaurant.find({name:/.*Reg.*/},{restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()

```
> db.restaurants.find(
  {"name": /.*Reg.*/},
  {
  "restaurant_id" : 1,
  "name":1,"borough":1,
  "cuisine" :1
  }
  );
{ _id: ObjectId("635166ca6473e4c38285b60e"),
    borough: 'Brooklyn',
    cuisine: 'American ',
    name: 'Regina Caterers',
    restaurant_id: '40356649' }
```

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

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

```
> db.restaurants.find(
  {
  "borough": "Bronx" ,
  $or : [
  { "cuisine" : "American " },
  { "cuisine" : "Chinese" }
  ]
  }
  );
< { _id: ObjectId("635166ca6473e4c38285b610"),
    address:
     { building: '2300',
       coord: [ -73.8786113, 40.8502883 ],
       street: 'Southern Boulevard',
       zipcode: '10460' },
```

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

Ans: db.restaurant.find().sort({name:1}).pretty()

```
> db.restaurants.find().sort({"name":1});
< { _id: ObjectId("635166cb6473e4c38285c296"),
    address:
     { building: '129',
       coord: [ -73.962943, 40.685007 ],
       street: 'Gates Avenue',
       zipcode: '11238' },
    borough: 'Brooklyn',
    cuisine: 'Italian',
    grades:
     [ { date: 2014-03-06T00:00:00.000Z, grade: 'A', score: 5 },
       { date: 2013-08-29T00:00:00.000Z, grade: 'A', score: 2 },
       { date: 2013-03-08T00:00:00.000Z, grade: 'A', score: 7 },
       { date: 2012-06-27T00:00:00.000Z, grade: 'A', score: 7 },
```

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

the columns.

Ans: db.restaurant.find().sort({name:-1}).pretty()

```
> db.restaurants.find().sort(
                            {"name":-1}
                            );
{ _id: ObjectId("635166ca6473e4c38285b6c5"),
  address:
   { building: '6946',
     coord: [ -73.8811834, 40.7017759 ],
     street: 'Myrtle Avenue',
     zipcode: '11385' },
  borough: 'Queens',
  cuisine: 'German',
  grades:
   [ { date: 2014-09-24T00:00:00.000Z, grade: 'A', score: 11 },
     { date: 2014-04-17T00:00:00.000Z, grade: 'A', score: 7 },
```

10. 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.

Ans: db.restaurant.find().sort({name:1,borough:-1}).pretty()

```
> db.restaurants.find().sort(
                            {"cuisine":1,"borough" : -1,}
                           );
< { _id: ObjectId("635166cb6473e4c38285bcf1"),
    address:
     { building: '1345',
       coord: [ -73.959249, 40.768076 ],
       street: '2 Avenue',
       zipcode: '10021' },
    borough: 'Manhattan',
    cuisine: 'Afghan',
    grades:
     [ { date: 2014-10-07T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2013-10-23T00:00:00.000Z, grade: 'A', score: 8 },
```

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

Ans: db.restaurant.find({'address.street':{$exists:true}}).count()

```
> db.restaurants.find(
                        {"address.street" :
                            { $exists : true }
                        }
                      );
< { _id: ObjectId("635166ca6473e4c38285b606"),
    address:
     { building: '1007',
       coord: [ -73.856077, 40.848447 ],
       street: 'Morris Park Ave',
       zipcode: '10462' },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades:
```

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.restaurants.find(
  {"borough" :{$in :["Staten Island","Queens","Bronx","Brooklyn"]}},
  {
  "restaurant_id" : 1,
  "name":1,"borough":1,
  "cuisine" :1
  }
  );
< { _id: ObjectId("635166ca6473e4c38285b606"),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445' }
  { _id: ObjectId("635166ca6473e4c38285b607"),
```

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
  }
  );
< { _id: ObjectId("635166ca6473e4c38285b608"),
    borough: 'Manhattan',
    cuisine: 'Irish',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841' }
  { _id: ObjectId("635166ca6473e4c38285b613"),
```

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
  }
  );
< { _id: ObjectId("635166ca6473e4c38285b611"),
    borough: 'Brooklyn',
    cuisine: 'American ',
```

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

```
> db.restaurants.find(
  {$or: [
    {name: /^Wil/},
    {"$and": [
         {"cuisine" : {$ne :"American "}},
         {"cuisine" : {$ne :"Chinees"}}
     ]}
  ]}
  ,{"restaurant_id" : 1,"name":1,"borough":1,"cuisine" :1}
  );
< { _id: ObjectId("635166ca6473e4c38285b606"),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445' }
```

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.restaurants.find(
                    {
                     "grades.date": ISODate("2014-08-11T00:00:00Z"),
                     "grades.grade":"A" ,
                     "grades.score" : 11
                    },
                    {"restaurant_id" : 1,"name":1,"grades":1}
                );
< { _id: ObjectId("635166ca6473e4c38285b684"),
    grades:
     [ { date: 2014-08-11T00:00:00.000Z, grade: 'A', score: 13 },
       { date: 2013-07-22T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2013-03-14T00:00:00.000Z, grade: 'A', score: 12 },
```

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.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}
                    );
< { _id: ObjectId("635166ca6473e4c38285bc31"),
    grades:
     [ { date: 2015-01-12T00:00:00.000Z, grade: 'A', score: 10 },
       { date: 2014-08-11T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2014-01-14T00:00:00.000Z, grade: 'A', score: 13 },
       { date: 2013-02-07T00:00:00.000Z, grade: 'A', score: 10 },
       { date: 2012-04-30T00:00:00.000Z, grade: 'A', score: 11 } ],
    name: 'Club Macanudo (Cigar Bar)',
    restaurant_id: '40526406' }
```

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

```
> db.restaurants.find(
                        {
                          "address.coord.1": {$gt : 42, $lte : 52}
                        },
                          {"restaurant_id" : 1,"name":1,"address":1,"coord":1}
                    );
< { _id: ObjectId("635166ca6473e4c38285b8a8"),
    address:
     { building: '47',
       coord: [ -78.877224, 42.89546199999999 ],
       street: 'Broadway @ Trinity Pl',
       zipcode: '10006' },
    name: 'T.G.I. Friday\'S',
```

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

```
> db.restaurants.find().sort({"name":1});
< { _id: ObjectId("635166cb6473e4c38285c296"),
    address:
     { building: '129',
       coord: [ -73.962943, 40.685007 ],
       street: 'Gates Avenue',
       zipcode: '11238' },
    borough: 'Brooklyn',
    cuisine: 'Italian',
    grades:
     [ { date: 2014-03-06T00:00:00.000Z, grade: 'A', score: 5 },
       { date: 2013-08-29T00:00:00.000Z, grade: 'A', score: 2 },
```

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

```
> db.restaurants.find().sort({"name":-1});
< { _id: ObjectId("635166ca6473e4c38285b6c5"),
    address:
     { building: '6946',
       coord: [ -73.8811834, 40.7017759 ],
       street: 'Myrtle Avenue',
       zipcode: '11385' },
    borough: 'Queens',
    cuisine: 'German',
    grades:
     [ { date: 2014-09-24T00:00:00.000Z, grade: 'A', score: 11 },
       { date: 2014-04-17T00:00:00.000Z, grade: 'A', score: 7 },
       { date: 2013-03-12T00:00:00.000Z, grade: 'A', score: 13 },
       { date: 2012-10-02T00:00:00.000Z, grade: 'A', score: 9 },
```

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.restaurants.find().sort({"cuisine":1,"borough" : -1,});
< { _id: ObjectId("635166cb6473e4c38285bcf1"),
    address:
     { building: '1345',
       coord: [ -73.959249, 40.768076 ],
       street: '2 Avenue',
       zipcode: '10021' },
    borough: 'Manhattan',
    cuisine: 'Afghan',
    grades:
     [ { date: 2014-10-07T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2013-10-23T00:00:00.000Z, grade: 'A', score: 8 },
       { date: 2012-10-26T00:00:00.000Z, grade: 'A', score: 13 },
```

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

```
> db.restaurants.find({"address.street" :{ $exists : true }});
< { _id: ObjectId("635166ca6473e4c38285b606"),
    address:
     { building: '1007',
       coord: [ -73.856077, 40.848447 ],
       street: 'Morris Park Ave',
       zipcode: '10462' },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades:
     [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
       { date: 2013-09-11T00:00:00.000Z, grade: 'A', score: 6 },
```

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

```
> db.restaurants.find({"address.coord" :{$type : 1}});
{ _id: ObjectId("635166ca6473e4c38285b606"),
  address:
   { building: '1007',
     coord: [ -73.856077, 40.848447 ],
     street: 'Morris Park Ave',
     zipcode: '10462' },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades:
   [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
     { date: 2013-09-11T00:00:00.000Z, grade: 'A', score: 6 },
     { date: 2013-01-24T00:00:00.000Z, grade: 'A', score: 10 },
     { date: 2011-11-23T00:00:00.000Z, grade: 'A', score: 9 },
```

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.restaurants.find(
                        {"grades.score" :
                           {$mod : [7,0]}
                        },
                           {"restaurant_id" : 1,"name":1,"grades":1}
                      );
< { _id: ObjectId("635166ca6473e4c38285b606"),
    grades:
     [ { date: 2014-03-03T00:00:00.000Z, grade: 'A', score: 2 },
       { date: 2013-09-11T00:00:00.000Z, grade: 'A', score: 6 },
       { date: 2013-01-24T00:00:00.000Z, grade: 'A', score: 10 },
       { date: 2011-11-23T00:00:00.000Z, grade: 'A', score: 9 },
       { date: 2011-03-10T00:00:00.000Z, grade: 'B', score: 14 } ],
```

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

```
> db.restaurants.find({ name :{ $regex : "mon.*", $options: "i" }},
                        {
                          "name":1,
                          "borough":1,
                          "address.coord":1,
                          "cuisine" :1
                         });
< { _id: ObjectId("635166ca6473e4c38285b69a"),
    address: { coord: [ -73.98306099999999, 40.7441419 ] },
    borough: 'Manhattan',
    cuisine: 'American ',
    name: 'Desmond\'S Tavern' }
  { _id: ObjectId("635166ca6473e4c38285b6a3"),
    address: { coord: [ -73.8221418, 40.7272376 ] },
    borough: 'Queens',
```

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.restaurants.find({ name :{ $regex : /^Mad/i, }},
                        {
                          "name":1,
                          "borough":1,
                          "address.coord":1,
                          "cuisine" :1
                         });
< { _id: ObjectId("635166ca6473e4c38285bb42"),
    address: { coord: [ -73.9860597, 40.7431194 ] },
    borough: 'Manhattan',
    cuisine: 'American ',
    name: 'Madison Square' }
  { _id: ObjectId("635166ca6473e4c38285bc10"),
    address: { coord: [ -73.98302199999999, 40.742313 ] },
    borough: 'Manhattan',
```