

- 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

```
rails@rails-System: ~  
> db.restaurant.find({}, {'name':1, 'restaurant_id':1, 'borough':1, 'cuisine':1, '_id':0}).pretty()  
{  
  "borough" : "Bronx",  
  "cuisine" : "Bakery",  
  "name" : "Morris Park Bake Shop",  
  "restaurant_id" : "30075445"  
}  
{  
  "borough" : "Manhattan",  
  "cuisine" : "Irish",  
  "name" : "Dj Reynolds Pub And Restaurant",  
  "restaurant_id" : "30191841"  
}  
{  
  "borough" : "Brooklyn",  
  "cuisine" : "American",  
  "name" : "Riviera Caterer",  
  "restaurant_id" : "40356018"  
}  
{  
  "borough" : "Queens",  
  "cuisine" : "Jewish/Kosher",  
  "name" : "Tov Kosher Kitchen",  
  "restaurant_id" : "40356068"  
}  
{  
  "borough" : "Queens",  
  "cuisine" : "American",  
  "name" : "Brunos On The Boulevard",  
  "restaurant_id" : "40356151"  
}  
{  
  "borough" : "Staten Island",  
  "cuisine" : "Jewish/Kosher",  
  "name" : "Kosher Island",  
  "restaurant_id" : "40356442"  
}  
{  
  "borough" : "Brooklyn",  
  "cuisine" : "Delicatessen",  
  "name" : "Wilken'S Fine Food",  
  "restaurant_id" : "40356483"  
}  
{  
  "borough" : "Brooklyn",  
  "cuisine" : "American",  
  "name" : "Regina Caterers",  
  "restaurant_id" : "40356649"  
}  
{  
  "borough" : "Bronx",
```

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

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

⇒ db.restaurant.find({'borough':'Bronx'}, {'name':1, '_id':0}).pretty()

```

rails@rails-System: ~
    "_id" : ObjectId("5947bc1fb60c37c179375cea"),
    "name" : "Lulu'S Coffee Shop"
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375cf0"), "name" : "Marina Delray" },
    "_id" : ObjectId("5947bc1fb60c37c179375d15"), "name" : "The Lark'S Nest" },
    "_id" : ObjectId("5947bc1fb60c37c179375d16"), "name" : "Terrace Cafe" },
    "_id" : ObjectId("5947bc1fb60c37c179375d17"), "name" : "African Terrace" },
    "_id" : ObjectId("5947bc1fb60c37c179375d18"), "name" : "Cool Zone" },
    "_id" : ObjectId("5947bc1fb60c37c179375d19"), "name" : "Beaver Pond" }
  ],
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d1a"),
    "name" : "African Market (Baboon Cafe)"
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d3a"),
    "name" : "Blue Bay Restaurant"
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d42"),
    "name" : "Seashore Restaurant"
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d4c"), "name" : "Bronx Grill" }
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d4f"),
    "name" : "Johnny'S Reef Restaurant"
  },
  {
    "_id" : ObjectId("5947bc1fb60c37c179375d51"), "name" : "White Castle" }
]
Type "it" for more
> db.restaurant.find({'borough':'Bronx'},{'name':1, '_id':0}).pretty()
{
  "name" : "Morris Park Bake Shop" }
{
  "name" : "Wild Asia" }
{
  "name" : "Happy Garden" }
{
  "name" : "Yankee Tavern" }
{
  "name" : "Mcdwyers Pub" }
{
  "name" : "Munchtime" }
{
  "name" : "Ithop" }
{
  "name" : "Lulu'S Coffee Shop" }
{
  "name" : "Marina Delray" }
{
  "name" : "The Lark'S Nest" }
{
  "name" : "Terrace Cafe" }
{
  "name" : "African Terrace" }
{
  "name" : "Cool Zone" }
{
  "name" : "Beaver Pond" }
{
  "name" : "African Market (Baboon Cafe)" }
{
  "name" : "Blue Bay Restaurant" }
{
  "name" : "Seashore Restaurant" }
{
  "name" : "Bronx Grill" }
{
  "name" : "Johnny'S Reef Restaurant" }
{
  "name" : "White Castle" }
]
Type "it" for more
> db.restaurant.find({'borough':'Bronx'},{'name':1, '_id':0}).pretty()

```

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

=> db.restaurant.find({'grades': {'\$elemMatch: {'score': {\$gt: 90}}}}).pretty()

```

rails@rails-System: ~
> db.restaurant.find({'grades': {'$elemMatch: {'score': {$gt: 90}}}}).pretty()
{
  "_id" : ObjectId("5947bc1fb60c37c179375e3e"),
  "address" : {
    "building" : "345",
    "coord" : [
      -73.9864626,
      40.7266739
    ],
    "street" : "East 6 Street",
    "zipcode" : "10003"
  },
  "borough" : "Manhattan",
  "cuisine" : "Indian",
  "grades" : [
    {
      "date" : ISODate("2014-09-15T00:00:00Z"),
      "grade" : "A",
      "score" : 5
    },
    {
      "date" : ISODate("2014-01-14T00:00:00Z"),
      "grade" : "A",
      "score" : 8
    },
    {
      "date" : ISODate("2013-05-30T00:00:00Z"),
      "grade" : "A",
      "score" : 12
    },
    {
      "date" : ISODate("2013-04-24T00:00:00Z"),
      "grade" : "P",
      "score" : 2
    },
    {
      "date" : ISODate("2012-10-01T00:00:00Z"),
      "grade" : "A",
      "score" : 9
    },
    {
      "date" : ISODate("2012-04-06T00:00:00Z"),
      "grade" : "C",
      "score" : 92
    },
    {
      "date" : ISODate("2011-11-03T00:00:00Z"),
      "grade" : "C",
      "score" : 41
    }
  ]
}

```

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

=> `db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5)`

```
rails@rails-System: ~  
},  
  "borough" : "Manhattan",  
  "cuisine" : "American",  
  "grades" : [  
    {  
      "date" : ISODate("2014-08-22T00:00:00Z"),  
      "grade" : "A",  
      "score" : 11  
    },  
    {  
      "date" : ISODate("2014-03-28T00:00:00Z"),  
      "grade" : "C",  
      "score" : 131  
    },  
    {  
      "date" : ISODate("2013-09-25T00:00:00Z"),  
      "grade" : "A",  
      "score" : 11  
    },  
    {  
      "date" : ISODate("2013-04-08T00:00:00Z"),  
      "grade" : "B",  
      "score" : 25  
    },  
    {  
      "date" : ISODate("2012-10-15T00:00:00Z"),  
      "grade" : "A",  
      "score" : 11  
    },  
    {  
      "date" : ISODate("2011-10-19T00:00:00Z"),  
      "grade" : "A",  
      "score" : 13  
    }  
  ],  
  "name" : "Murals On 54/Randolphs'S",  
  "restaurant_id" : "40372466"  
}  
> db.restaurant.find({'borough':'Bronx'},{'name': 1}).limit(5)  
{ "_id" : ObjectId("5947bc1fb60c37c179375c3f"), "name" : "Morris Park Bake Shop" }  
{ "_id" : ObjectId("5947bc1fb60c37c179375c49"), "name" : "Wild Asia" }  
{ "_id" : ObjectId("5947bc1fb60c37c179375c74"), "name" : "Happy Garden" }  
{ "_id" : ObjectId("5947bc1fb60c37c179375cac"), "name" : "Yankee Tavern" }  
{ "_id" : ObjectId("5947bc1fb60c37c179375cbf"), "name" : "Mcdwyers Pub" }  
> db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5)  
{ "name" : "Morris Park Bake Shop" }  
{ "name" : "Wild Asia" }  
{ "name" : "Happy Garden" }  
{ "name" : "Yankee Tavern" }  
{ "name" : "Mcdwyers Pub" }  
>
```

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

=> `db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5).skip(5)`

```

rails@rails-System: ~
"grade" : "A",
"score" : 11
},
{
  "date" : ISODate("2014-03-28T00:00:00Z"),
  "grade" : "C",
  "score" : 131
},
{
  "date" : ISODate("2013-09-25T00:00:00Z"),
  "grade" : "A",
  "score" : 11
},
{
  "date" : ISODate("2013-04-08T00:00:00Z"),
  "grade" : "B",
  "score" : 25
},
{
  "date" : ISODate("2012-10-15T00:00:00Z"),
  "grade" : "A",
  "score" : 11
},
{
  "date" : ISODate("2011-10-19T00:00:00Z"),
  "grade" : "A",
  "score" : 13
}
},
{
  "name" : "Murals On 54/Randolphs'S",
  "restaurant_id" : "40372466"
}
}
> db.restaurant.find({'borough':'Bronx'},{'name': 1}).limit(5)
{ "_id" : ObjectId("5947bc1fb60c37c179375c3f"), "name" : "Morris Park Bake Shop" }
{ "_id" : ObjectId("5947bc1fb60c37c179375c49"), "name" : "Wild Asia" }
{ "_id" : ObjectId("5947bc1fb60c37c179375c74"), "name" : "Happy Garden" }
{ "_id" : ObjectId("5947bc1fb60c37c179375cac"), "name" : "Yankee Tavern" }
{ "_id" : ObjectId("5947bc1fb60c37c179375cbf"), "name" : "Mcdwyers Pub" }
> db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5)
{ "name" : "Morris Park Bake Shop" }
{ "name" : "Wild Asia" }
{ "name" : "Happy Garden" }
{ "name" : "Yankee Tavern" }
{ "name" : "Mcdwyers Pub" }
> db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5).skip(5)
{ "name" : "Munchtime" }
{ "name" : "Ihop" }
{ "name" : "Lulu'S Coffee Shop" }
{ "name" : "Marina Delray" }
{ "name" : "The Lark'S Nest" }
>

```

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

=> db.restaurant.find({'grades': {'\$elemMatch': {'score': {'\$gt: 80, \$lt: 100}}}}).pretty()

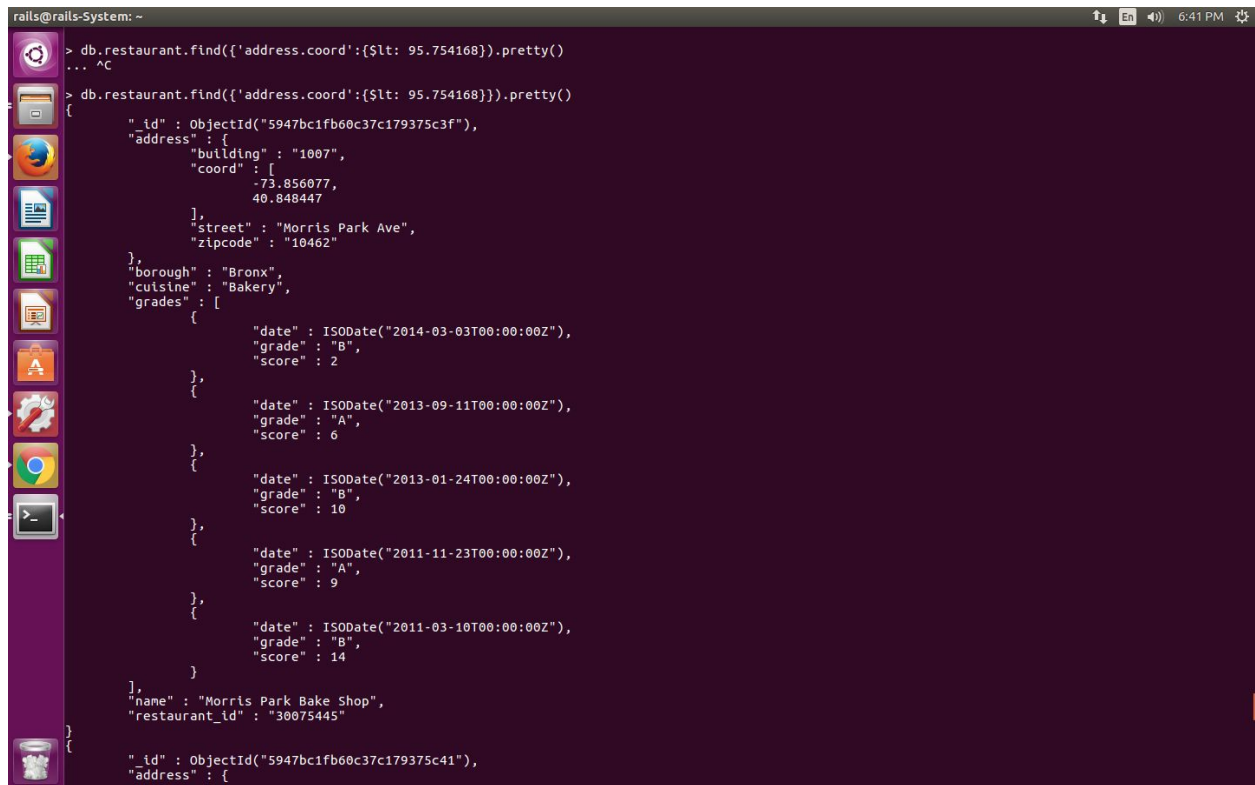
```

rails@rails-System: ~
{ "name" : "Munchtime" }
{ "name" : "Ihop" }
{ "name" : "Lulu'S Coffee Shop" }
{ "name" : "Marina Delray" }
{ "name" : "The Lark'S Nest" }
> db.restaurant.find({'grades': {'$elemMatch': {'score': {'$gt: 80, $lt: 100}}}}).pretty()
{
  "_id" : ObjectId("5947bc1fb60c37c179375e3e"),
  "address" : {
    "building" : "345",
    "coord" : [
      -73.9864626,
      40.7266739
    ],
    "street" : "East 6 Street",
    "zipcode" : "10003"
  },
  "borough" : "Manhattan",
  "cuisine" : "Indian",
  "grades" : [
    {
      "date" : ISODate("2014-09-15T00:00:00Z"),
      "grade" : "A",
      "score" : 5
    },
    {
      "date" : ISODate("2014-01-14T00:00:00Z"),
      "grade" : "A",
      "score" : 8
    },
    {
      "date" : ISODate("2013-05-30T00:00:00Z"),
      "grade" : "A",
      "score" : 12
    },
    {
      "date" : ISODate("2013-04-24T00:00:00Z"),
      "grade" : "B",
      "score" : 2
    },
    {
      "date" : ISODate("2012-10-01T00:00:00Z"),
      "grade" : "A",
      "score" : 9
    },
    {
      "date" : ISODate("2012-04-06T00:00:00Z"),
      "grade" : "C",
      "score" : 92
    }
  ],
  "name" : "The Lark'S Nest"
}
>

```

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

=>db.restaurant.find({'address.coord':{'\$lt: 95.754168}}).pretty()

A terminal window titled 'rails@rails-System: ~' shows a MongoDB query being executed. The query is 'db.restaurant.find({'address.coord':{'\$lt: 95.754168}}).pretty()'. The output displays two restaurant records. The first record is for 'Morris Park Bake Shop' with a 'restaurant_id' of '30075445'. It is located in the Bronx at 'Morris Park Ave' with coordinates [-73.856077, 40.848447]. The second record is partially visible at the bottom. The terminal also shows a previous command and its output, which was truncated with '... ^C'.

```
rails@rails-System: ~  
> db.restaurant.find({'address.coord':{'$lt: 95.754168}}).pretty()  
... ^C  
> db.restaurant.find({'address.coord':{'$lt: 95.754168}}).pretty()  
{  
  "_id" : ObjectId("5947bc1fb60c37c179375c3f"),  
  "address" : {  
    "building" : "1007",  
    "coord" : [  
      -73.856077,  
      40.848447  
    ],  
    "street" : "Morris Park Ave",  
    "zipcode" : "10462"  
  },  
  "borough" : "Bronx",  
  "cuisine" : "Bakery",  
  "grades" : [  
    {  
      "date" : ISODate("2014-03-03T00:00:00Z"),  
      "grade" : "B",  
      "score" : 2  
    },  
    {  
      "date" : ISODate("2013-09-11T00:00:00Z"),  
      "grade" : "A",  
      "score" : 6  
    },  
    {  
      "date" : ISODate("2013-01-24T00:00:00Z"),  
      "grade" : "B",  
      "score" : 10  
    },  
    {  
      "date" : ISODate("2011-11-23T00:00:00Z"),  
      "grade" : "A",  
      "score" : 9  
    },  
    {  
      "date" : ISODate("2011-03-10T00:00:00Z"),  
      "grade" : "B",  
      "score" : 14  
    }  
  ],  
  "name" : "Morris Park Bake Shop",  
  "restaurant_id" : "30075445"  
},  
  "_id" : ObjectId("5947bc1fb60c37c179375c41"),  
  "address" : {
```

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

```
rails@rails-System: ~
}
  },
  "name" : "Olive's",
  "restaurant_id" : "40363151"
}
Type "it" for more
> db.restaurant.find({'cuisine': {'$ne': 'American '}, 'grades':{'$elemMatch': {'score': {'$gt': 70}}}, 'address.coord': {'$lt': -65.754168}}).pretty()
{
  "_id" : ObjectId("5947bc1fb60c37c179375e3e"),
  "address" : {
    "building" : "345",
    "coord" : [
      -73.9864626,
      40.7266739
    ],
    "street" : "East 6 Street",
    "zipcode" : "10003"
  },
  "borough" : "Manhattan",
  "cuisine" : "Indian",
  "grades" : [
    {
      "date" : ISODate("2014-09-15T00:00:00Z"),
      "grade" : "A",
      "score" : 5
    },
    {
      "date" : ISODate("2014-01-14T00:00:00Z"),
      "grade" : "A",
      "score" : 8
    },
    {
      "date" : ISODate("2013-05-30T00:00:00Z"),
      "grade" : "A",
      "score" : 12
    },
    {
      "date" : ISODate("2013-04-24T00:00:00Z"),
      "grade" : "P",
      "score" : 2
    },
    {
      "date" : ISODate("2012-10-01T00:00:00Z"),
      "grade" : "A",
      "score" : 9
    },
    {
      "date" : ISODate("2012-04-06T00:00:00Z"),
      "grade" : "C",
      "score" : 92
    }
  ],
}
```

- Write a MongoDB query to update the restaurant's grade to 'B' whose score is more than or equal to 10(update only the first grade).

=>db.restaurant.update({'grades': {'\$elemMatch':{'score': {'\$gte': 10}}}}, { \$set: {'grades':{'\$elemMatch: {'grade':'B'}}}})


```
rails@rails-System: ~$ cat data.json
{
  "grade": "A",
  "score": 6
},
{
  "date": ISODate("2012-06-19T00:00:00Z"),
  "grade": "A",
  "score": 13
},
{
  "name": "West 79Th Street Boat Basin Cafe",
  "restaurant_id": "40756344"
}
}

> db.find.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades': }}
[1]+  Stopped                  mongo
rails@rails-System:~$ mongo
MongoDB shell version: 3.2.16
connecting to: test
Server has startup warnings:
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'
> db.find.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.grade': 'A' }})
2017-09-05T18:51:18.668+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:63
> db.find.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.0.grade': 'A' }})
2017-09-05T18:51:38.643+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:63
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.grade': 'A' }})
2017-09-05T18:51:59.371+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:69
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.0.grade': 'B' }})
2017-09-05T18:52:14.075+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:69
> db.restaurant.update({'grades.score': {'$gte': 10}}, {'$set': {'grades.0.grade': 'B' }})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.0.grade': 'B' }})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.0.grade': 'B' }})
2017-09-05T18:54:18.489+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:80
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades.0.grade': 'B' }})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update({'grades': {'$elemMatch': {'score': {'$gte': 10}}}}, {'$set': {'grades': {'$elemMatch': {'grade': 'B'}} }})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

- Write a MongoDB query to delete the restaurants in borough Bronx.
=>db.restaurant.remove({'borough':'Bronx'})

```
rails@rails-System: ~
"restaurant_id" : "40756344"
}
> db.find.update({'grades': {'$elemMatch: {'score': {$gte: 10}}}}, {'$set: {'grades': }}
[1]+  Stopped                  mongo
rails@rails-System:~$ mongo
MongoDB shell version: 3.2.16
connecting to: test
Server has startup warnings:
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'
2017-09-05T15:50:00.636+0530 I CONTROL [initandlisten]
> db.find.update({'grades': {'$elemMatch: {'score': {$gte: 10}}}}, {'$set: {'grades.grade': 'A' }} })
2017-09-05T18:51:18.668+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:63
> db.find.update({'grades': {'$elemMatch: {'score': {$gte: 10}}}}, {'$set: {'grades.0.grade': 'A' }} })
2017-09-05T18:51:38.643+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:63
> db.restaurant.update({'grades': {'$elemMatch: {'score': {$gte: 10}}}}, {'$set: {'grades.grade': 'A' }} })
2017-09-05T18:51:59.371+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:69
> db.restaurant.update({'grades': {'$elemMatch: {'score': {$gte: 10}}}}, {'$set: {'grades.0.grade': 'B' }} })
2017-09-05T18:52:14.075+0530 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:69
> db.restaurant.update( {'grades.score':{$gte: 10}}, { $set: {'grades.0.grade':'B' } } )
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update( {'grades': {'$elemMatch :{$gte: 10}}}, { $set: {'grades.0.grade':'B' } } )
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update( {'grades': {'$elemMatch :{'score': {$gte: 10}}}}, { $set: {'grades.0.grade':'B' } } )
2017-09-05T18:54:18.489+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:80
> db.restaurant.update( {'grades': {'$elemMatch :{'score': {$gte: 10}}}}, { $set: {'grades.0.grade':'B' } } )
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.update( {'grades': {'$elemMatch :{'score': {$gte: 10}}}}, { $set: {'grades':{$elemMatch: {'grade':'B'}}} } )
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.restaurant.delete({'borough':'Bronx'})
2017-09-05T18:57:34.471+0530 E QUERY [thread1] TypeError: db.restaurant.delete is not a function :
@(shell):1:1
> db.restaurant.remove({'borough':'Bronx'})
WriteResult({ "nRemoved" : 0 })
> db.restaurant.find({'borough':'Bronx'},{'name': 1, '_id': 0}).limit(5)
> db.restaurant.find({'borough':'Bronx'})
> db.restaurant.remove({'borough':'Manhattan'})
WriteResult({ "nRemoved" : 0 })
>
```