|  |  |
| --- | --- |
| 1 | Lab Program 2 |
| 1 | Write a MongoDB query to find the restaurants who achieved a score more than 90. |
|  | db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}}); |
| 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 |
|  | db.restaurants.find(  {$and:  [  {"cuisine" : {$ne :"American "}},  {"grades.score" : {$gt : 70}},  {"address.coord" : {$lt : -65.754168}}  ]  }  ); |
| 3 | 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.restaurants.find(  {name: /^Wil/},  {  "restaurant\_id" : 1,  "name":1,"borough":1,  "cuisine" :1  }  ); |
| 4 | Find the restaurant Id, name, borough and cuisine for those restaurants which contain ces as last three letters for its name |
|  | db.restaurants.find(  {name: /ces$/},  {  "restaurant\_id" : 1,  "name":1,"borough":1,  "cuisine" :1  }  ); |
| 5 | Write a query to update the name of the restaurant with ID your name. |
|  | db.restaurants.insertOne({  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  },  {  date: ISODate('2011-11-23T00:00:00.000Z'),  grade: 'A',  score: 9  },  {  date: ISODate('2011-03-10T00:00:00.000Z'),  grade: 'B',  score: 14  }  ],  name: 'Dayananda Sagar',  restaurant\_id: '30075445'  }) |
|  | db.restaurants.updateOne( { name: 'Dayananda Sagar' }, { $set: { name: 'Ullas' } } ) |
|  | db.restaurants.find({ name: 'Ullas' }) |
| 2 | Lab Program 3 |
| 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.restaurants.find(  {  grades: {  $elemMatch: {  grade: 'A',  score: 11,  date: ISODate("2014-08-11T00:00:00Z")  }  }  },  {  restaurant\_id: 1,  name: 1,  grades: 1  }  ) |
| 2 | 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.restaurants.find(  {  "address.coord.1": { $gt: 42, $lte: 52 }  },  {  restaurant\_id: 1,  name: 1,  address: 1  }  ) |
| 3 | Write a MongoDB query to find the average score for each restaurant. |
|  | db.restaurants.aggregate([  {  $unwind: "$grades"  },  {  $group: {  \_id: "$restaurant\_id",  name: { $first: "$name" },  averageScore: { $avg: "$grades.score" }  }  },  {  $project: {  \_id: 0,  restaurant\_id: "$\_id",  name: 1,  averageScore: 1  }  }  ]) |
|  | This query performs the following steps:   1. **$unwind**: Deconstructs the grades array field from the input documents to output a document for each element. 2. **$group**: Groups the documents by restaurant\_id, and calculates the average score using the $avg operator. It also retrieves the name of each restaurant. 3. **$project**: Reshapes each document in the stream, including only the restaurant\_id, name, and averageScore fields in the output and excluding the \_id field. |
| 4 | Write a MongoDB query to find the count of restaurants for each cuisine and borough |
|  | db.restaurants.aggregate([  {  $group: {  \_id: { cuisine: "$cuisine", borough: "$borough" },  count: { $sum: 1 }  }  },  {  $project: {  \_id: 0,  cuisine: "$\_id.cuisine",  borough: "$\_id.borough",  count: 1  }  },  {  $sort: {  borough: 1,  cuisine: 1  }  }  ]) |
| 5 | Create a query to update all grade scores of the restaurant with given ID to increase them by 5 points. |
|  | db.restaurants.updateMany(  { restaurant\_id: "30075445" },  { $inc: { "grades.$[].score": 5 } }  ) |