Exp-14

Name: RAGUL.A Roll.no:230701252

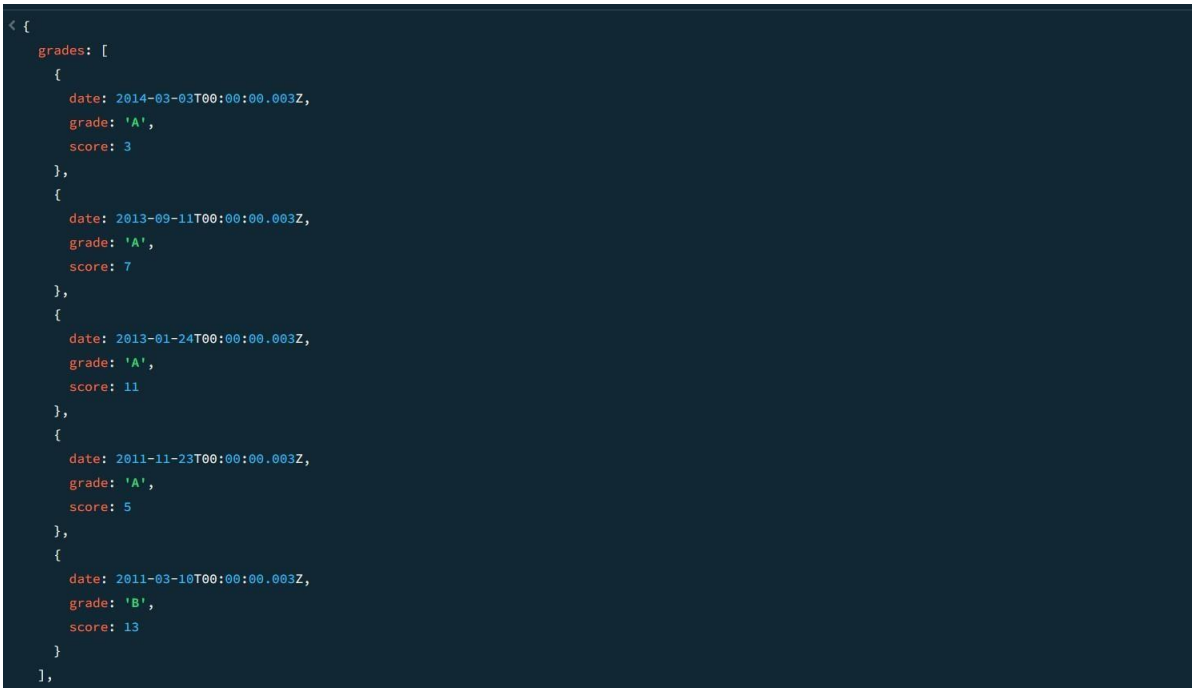
| Ex.No.: 14 | | MONGO DB |
|---|---|---|
| Date: | 26/09/2024 | |

**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.restaurants.find(
  {
    $or: [
      { cuisine: { $nin: ["American", "Chinees"] } },
      { name: { $regex: /^Wil/i } }
    ]
  },
  {
    restaurant_id: 1,
    name: 1,
    borough: 1,
    cuisine: 1,
    _id: 0
  }
);
```

**2. 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
   }
  }
 },
 {
  restaurant_id: 1,
  name: 1,
  grades: 1,
  _id: 0
 }
);
```



**3. 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 9on an ISODate "2014-08-11T00:00:00Z".**

```
db.restaurants.find(
 {
  "grades.1": {
   $elemMatch: {
    grade: "A",
    score: 9
   }
  }
 },
 {
  restaurant_id: 1,
  name: 1,
  grades: 1,
  _id: 0
 }
);
```

**4. 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 whichis more than 42 and upto 52..**

```
db.restaurants.find(
 {
  "address.coord.1": { $gt: 42, $lte: 52 }
 },
 {
  restaurant_id: 1,
  name: 1,
  address: 1,
  _id: 0
 }
);
```

**5. Write a MongoDB query to arrange the name of the restaurants in ascending orderalong with all the columns.**

```
db.restaurants.find().sort({ name: 1 });
```

**SAMPLE OUTPUT:-**


```
{
  _id: ObjectId('671b5e6d56ec9972ca8f5dc4'),
  address: {
    building: 5566,
    coord: [
      -73.867377,
      40.854047
    ],
    street: '28th Avenue',
    zipcode: 10490
  },
  borough: 'Bronx',
  cuisine: 'BBQ',
  grades: [
    {
      date: 2014-03-03T00:00:00.028Z,
      grade: 'A',
      score: 10
    },
    {
      date: 2013-09-11T00:00:00.028Z,
      grade: 'A',
      score: 7
    },
    {
      date: 2013-01-24T00:00:00.028Z,
      grade: 'A',
      score: 11
    },
    {
      date: 2011-11-23T00:00:00.028Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2011-03-10T00:00:00.028Z,
      grade: 'B',
```

```
      score: 15
    }
  ],
  name: 'BBQ Haven',
  restaurant_id: 30075473
}


{
  _id: ObjectId('671b5dab56ec9972ca8f5db0'),
  address: {
    building: 5566,
    coord: [
      -73.859377,
      40.850047
    ],
    street: '8th Avenue',
    zipcode: 10470
  },
  borough: 'Manhattan',
  cuisine: 'French',
  grades: [
    {
      date: 2014-03-03T00:00:00.008Z,
      grade: 'A',
      score: 7
    },
    {
      date: 2013-09-11T00:00:00.008Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2013-01-24T00:00:00.008Z,
      grade: 'A',
      score: 10
    },
    {
      date: 2011-11-23T00:00:00.008Z,
      grade: 'B',
      score: 15
    },
    {
      date: 2011-03-10T00:00:00.008Z,
```

```
    grade: 'A',
    score: 6
  }
 ],
 name: 'Bistro Belle',
 restaurant_id: 30075453
}
```

**6. Write a MongoDB query to arrange the name of the restaurants in descending alongwith all the columns.**

```
db.restaurants.find().sort({ name: -1 });
```

**SAMPLE OUTPUT**

```
{
 _id: ObjectId('671b5e9456ec9972ca8f5dc8'),
 address: {
  building: 9900,
  coord: [
   -73.868977,
   40.854847
  ],
  street: '32nd Avenue',
  zipcode: 10494
 },
 borough: 'Manhattan',
 cuisine: 'Russian',
 grades: [
  {
   date: 2014-03-03T00:00:00.032Z,
   grade: 'A',
   score: 10
  },
  {
   date: 2013-09-11T00:00:00.032Z,
   grade: 'B',
   score: 5
  },
  {
```

```
    date: 2013-01-24T00:00:00.032Z,
    grade: 'A',
    score: 9
  },
  {
    date: 2011-11-23T00:00:00.032Z,
    grade: 'A',
    score: 8
  },
  {
    date: 2011-03-10T00:00:00.032Z,
    grade: 'A',
    score: 11
  }
 ],
 name: "Tsar's Table",
 restaurant_id: 30075477
}


{
 _id: ObjectId('671b5e6d56ec9972ca8f5dbe'),
 address: {
  building: 9900,
  coord: [
   -73.864977,
   40.852847
  ],
  street: '22nd Avenue',
  zipcode: 10484
 },
 borough: 'Bronx',
 cuisine: 'Italian',
 grades: [
  {
    date: 2014-03-03T00:00:00.022Z,
    grade: 'A',
    score: 8
  },
  {
    date: 2013-09-11T00:00:00.022Z,
    grade: 'B',
    score: 5
  },
```

```
    {
      date: 2013-01-24T00:00:00.022Z,
      grade: 'A',
      score: 12
    },
    {
      date: 2011-11-23T00:00:00.022Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2011-03-10T00:00:00.022Z,
      grade: 'A',
      score: 14
    }
  ],
  name: 'Trattoria Bella',
  restaurant_id: 30075467
}
```

**7. Write a MongoDB query to arrange the name of the cuisine in ascending order and forthat same cuisine borough should be in descending order.**

```
db.restaurants.find().sort({ cuisine: 1, borough: -1 });
```

**SAMPLE OUTPUT:-**

```
{
  _id: ObjectId('671b5d549d3d63480e0a64e9'),
  address: {
    building: 2233,
    coord: [
      -73.858177,
      40.849447
    ],
    street: '5th Avenue',
    zipcode: 10467
  },
  borough: 'Bronx',
  cuisine: 'American',
```

```
  grades: [
    {
      date: 2014-03-03T00:00:00.005Z,
      grade: 'A',
      score: 10
    },
    {
      date: 2013-09-11T00:00:00.005Z,
      grade: 'A',
      score: 6
    },
    {
      date: 2013-01-24T00:00:00.005Z,
      grade: 'B',
      score: 12
    },
    {
      date: 2011-11-23T00:00:00.005Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2011-03-10T00:00:00.005Z,
      grade: 'A',
      score: 14
    }
  ],
  name: 'Burger Bistro',
  restaurant_id: 30075450
}

{
  _id: ObjectId('671b5e6d56ec9972ca8f5dc4'),
  address: {
    building: 5566,
    coord: [
      -73.867377,
      40.854047
    ],
    street: '28th Avenue',
    zipcode: 10490
  },
  borough: 'Bronx',
  cuisine: 'BBQ',
```

```
  grades: [
    {
      date: 2014-03-03T00:00:00.028Z,
      grade: 'A',
      score: 10
    },
    {
      date: 2013-09-11T00:00:00.028Z,
      grade: 'A',
      score: 7
    },
    {
      date: 2013-01-24T00:00:00.028Z,
      grade: 'A',
      score: 11
    },
    {
      date: 2011-11-23T00:00:00.028Z,
      grade: 'A',
      score: 9
    },
    {
      date: 2011-03-10T00:00:00.028Z,
      grade: 'B',
      score: 15
    }
  ],
  name: 'BBQ Haven',
  restaurant_id: 30075473
}
```

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

```
db.restaurants.find(
  {
    "address.street": { $exists: false }
  }
);
```

```
> db.restaurants.find(
    {
      "address.street": { $exists: false }
    }
  );
<
Customers >
```

**9. Write a MongoDB query which will select all documents in the restaurants collectionwhere the coord field value is Double.**

```
db.restaurants.find(
  {
    "address.coord": { $type: "double" }
  }
);
```

**SAMPLE OUTPUT:-**

```
{
  _id: ObjectId('671b92d339ec8a9bc8b6588b'),
  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'
}

{
 _id: ObjectId('671b5d549d3d63480e0a64e5'),
 address: {
  building: 1234,
  coord: [
   -73.856577,
   40.848647
  ],
  street: '1st Avenue',
  zipcode: 10463
 },
 borough: 'Bronx',
 cuisine: 'Italian',
 grades: [
  {
   date: 2014-03-03T00:00:00.001Z,
```

```
    grade: 'A',
    score: 5
  },
  {
    date: 2013-09-11T00:00:00.001Z,
    grade: 'A',
    score: 8
  },
  {
    date: 2013-01-24T00:00:00.001Z,
    grade: 'B',
    score: 12
  },
  {
    date: 2011-11-23T00:00:00.001Z,
    grade: 'A',
    score: 7
  },
  {
    date: 2011-03-10T00:00:00.001Z,
    grade: 'A',
    score: 15
  }
 ],
 name: 'Pasta Palace',
 restaurant_id: 30075446
}
```

**10. Write a MongoDB query which will select the restaurant Id, name and grades forthose 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: 0
 }
);
```

**SAMPLE OUTPUT:-**


```
{
  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'
}

{
  grades: [
    {
      date: 2014-03-03T00:00:00.001Z,
      grade: 'A',
      score: 5
    },
    {
```

```
    date: 2013-09-11T00:00:00.001Z,
    grade: 'A',
    score: 8
  },
  {
    date: 2013-01-24T00:00:00.001Z,
    grade: 'B',
    score: 12
  },
  {
    date: 2011-11-23T00:00:00.001Z,
    grade: 'A',
    score: 7
  },
  {
    date: 2011-03-10T00:00:00.001Z,
    grade: 'A',
    score: 15
  }
 ],
 name: 'Pasta Palace',
 restaurant_id: 30075446
}
```

**11. 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 itsname.**

```
db.restaurants.find(
 {
  name: { $regex: /mon/i }
 },
 {
  name: 1,
  borough: 1,
  "address.coord.0": 1, // Longitude
  "address.coord.1": 1, // Latitude
  cuisine: 1,
  _id: 0
 }
);
```

**12. Write a MongoDB query to find the restaurant name, borough, longitude and latitudeand 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.0": 1, // Longitude
    "address.coord.1": 1, // Latitude
    cuisine: 1,
    _id: 0
  }
);
```

**13. Write a MongoDB query to find the restaurants that have at least one grade with ascore of less than 5.**

```
db.restaurants.find(
  {
    "grades.score": { $lt: 5 }
  }
);
```

**SAMPLE OUTPUT:-**

```
{
  _id: ObjectId('671b92d339ec8a9bc8b6588b'),
  address: {
    building: '1007',
```

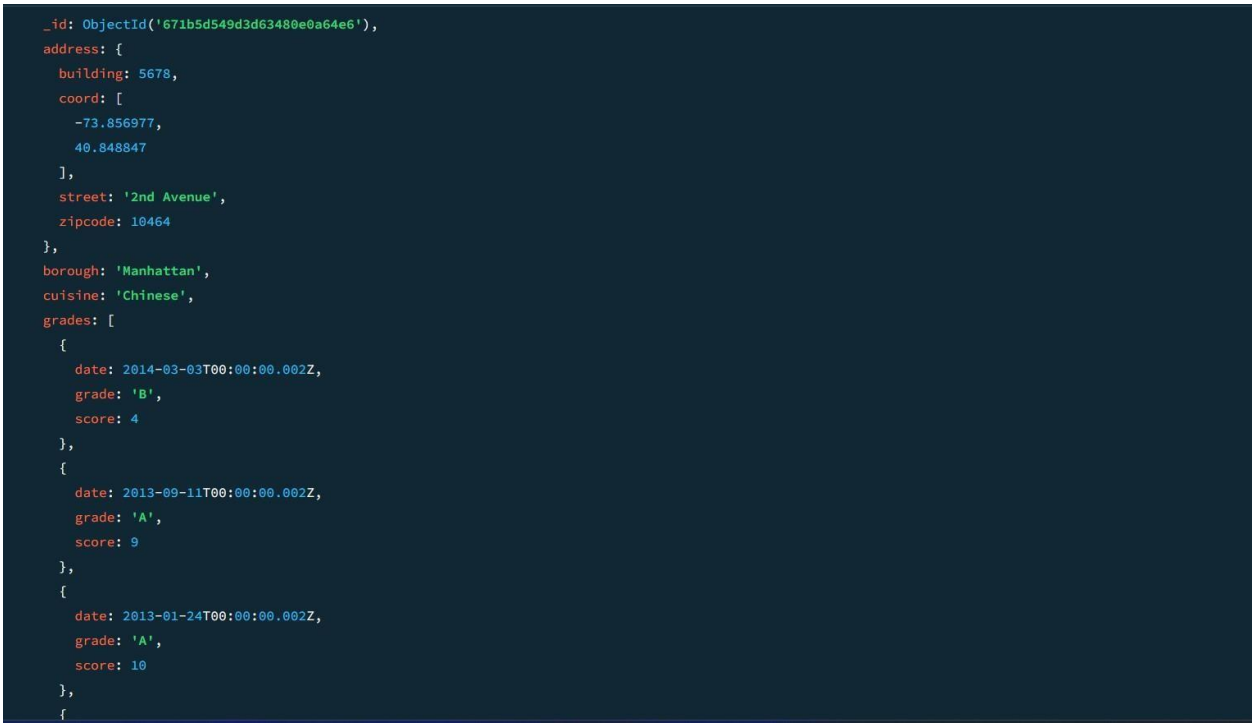
```
    coord: [
      -73.856077,
      40.848447
    ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough:    'Bronx',
  cuisine:    'Bakery',
  grades: [
    {
      date: 2014-03-03T00:00:00.000Z,
      grade: 'A',
      score: 2
    },
    {
      date: 2013-09-11T00:00:00.000Z,
      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'
}


{
  _id: ObjectId('671b5d549d3d63480e0a64e6'),
  address: {
```

```
  building: 5678,
  coord: [
    -73.856977,
    40.848847
  ],
  street: '2nd Avenue',
  zipcode: 10464
 },
 borough: 'Manhattan',
 cuisine: 'Chinese',
 grades: [
  {
   date: 2014-03-03T00:00:00.002Z,
   grade: 'B',
   score: 4
  },
  {
   date: 2013-09-11T00:00:00.002Z,
   grade: 'A',
   score: 9
  },
  {
   date: 2013-01-24T00:00:00.002Z,
   grade: 'A',
   score: 10
  },
  {
   date: 2011-11-23T00:00:00.002Z,
   grade: 'A',
   score: 8
  },
  {
   date: 2011-03-10T00:00:00.002Z,
   grade: 'B',
   score: 16
  }
 ],
 name: 'Dragon Wok',
 restaurant_id: 30075447
}
```

**14. Write a MongoDB query to find the restaurants that have at least one grade with ascore of less than 5 and that are located in the borough of Manhattan.**

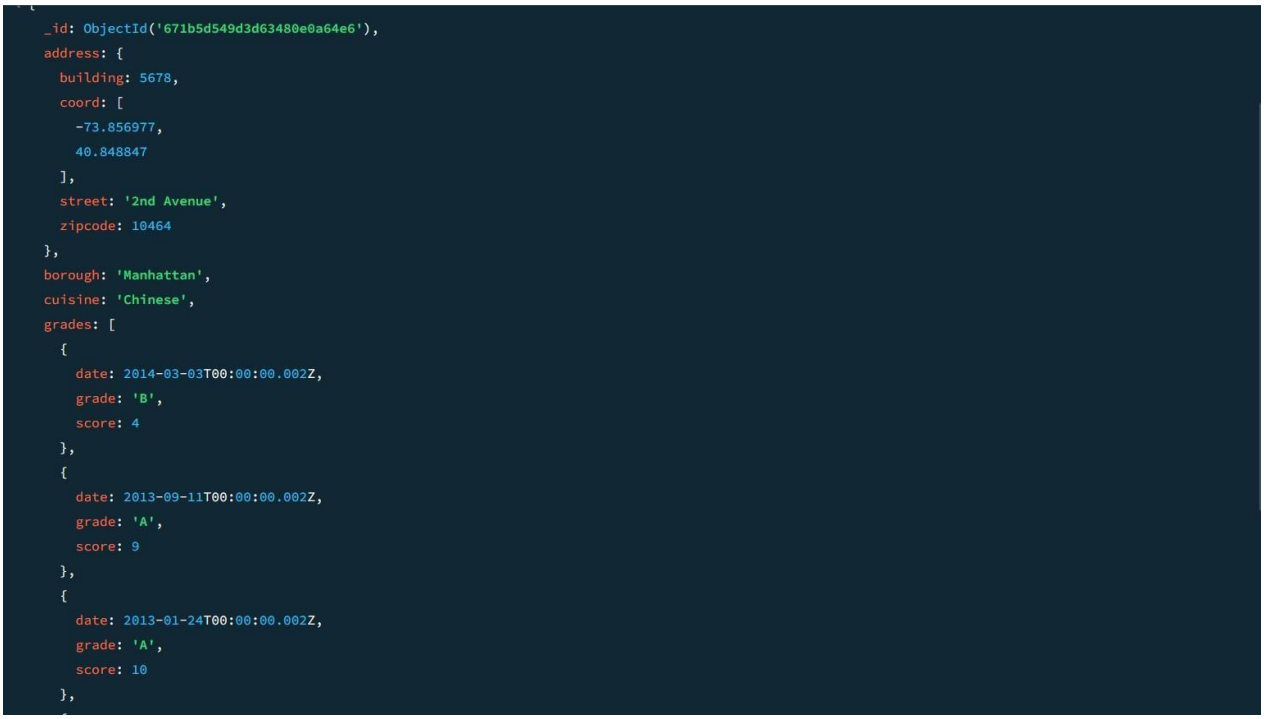
```
db.restaurants.find(
  {
    "grades.score": { $lt: 5 },
    borough: "Manhattan"
  }
);
```



```
_id: ObjectId('671b5d549d3d63480e0a64e6'),
address: {
  building: 5678,
  coord: [
    -73.856977,
    40.848847
  ],
  street: '2nd Avenue',
  zipcode: 10464
},
borough: 'Manhattan',
cuisine: 'Chinese',
grades: [
  {
    date: 2014-03-03T00:00:00.002Z,
    grade: 'B',
    score: 4
  },
  {
    date: 2013-09-11T00:00:00.002Z,
    grade: 'A',
    score: 9
  },
  {
    date: 2013-01-24T00:00:00.002Z,
    grade: 'A',
    score: 10
  },
  {
```

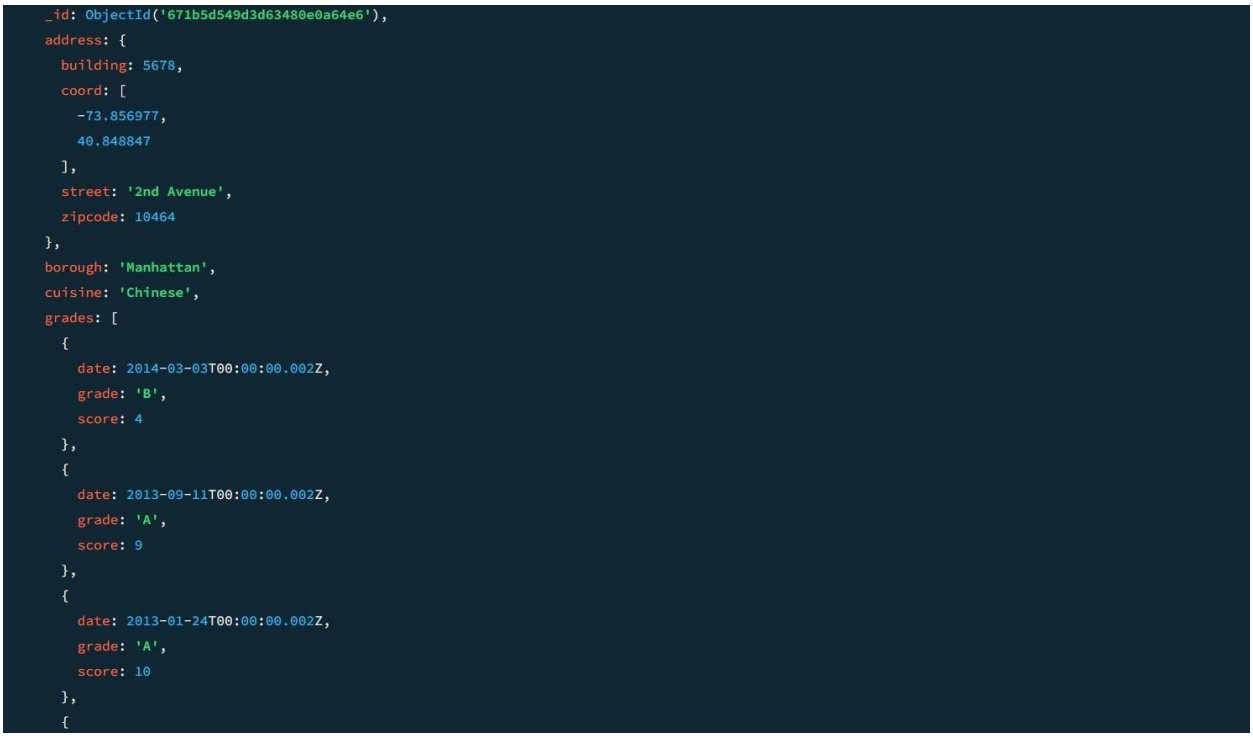
**15. Write a MongoDB query to find the restaurants that have at least one grade with ascore of less than 5 and that are located in the borough of Manhattan or Brooklyn.**

```
db.restaurants.find(
  {
    "grades.score": { $lt: 5 },
    borough: { $in: ["Manhattan", "Brooklyn"] }
  }
);
```




**16. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, andtheir cuisine is not American.**

```
db.restaurants.find(
  {
    "grades.score": { $lt: 5 },
    borough: { $in: ["Manhattan", "Brooklyn"] },
    cuisine: { $ne: "American" }
  }
);
```




**17. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, andtheir cuisine is not American or Chinese.**

```
db.restaurants.find(
  {
    "grades.score": { $lt: 5 },
    borough: { $in: ["Manhattan", "Brooklyn"] },
    cuisine: { $nin: ["American", "Chinese"] }
  }
);
```

**18. Write a MongoDB query to find the restaurants that have a grade with a score of 2 anda grade with a score of 6.**

```
db.restaurants.find(
  {
    grades: {
      $all: [
        { $elemMatch: { score: 2 } },
```

```
      { $elemMatch: { score: 6 } }
    ]
  }
 }
);
```

**SAMPLE OUTPUT:-**

```
{
 _id: ObjectId('671b92d339ec8a9bc8b6588b'),
 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'
}


{
 _id: ObjectId('671b5c5f9d3d63480e0a64e4'),
 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
}
```

**19. Write a MongoDB query to find the restaurants that have a grade with a score of 2 anda grade with a score of 6 and are located in the borough of Manhattan.**

```
db.restaurants.find(
  {
    borough: "Manhattan",
    grades: {
      $all: [
        { $elemMatch: { score: 2 } },
        { $elemMatch: { score: 6 } }
      ]
    }
  }
);
```

**20. Write a MongoDB query to find the restaurants that have a grade with a score of 2 anda grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.**

```
db.restaurants.find(
  {
    borough: { $in: ["Manhattan", "Brooklyn"] },
    grades: {
      $all: [
        { $elemMatch: { score: 2 } },
```

```
      { $elemMatch: { score: 6 } }
    ]
  }
 }
);
```

**21. Write a MongoDB query to find the restaurants that have a grade with a score of 2 anda grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.**

```
db.restaurants.find(
 {
  borough: { $in: ["Manhattan", "Brooklyn"] },
  grades: {
   $all: [
    { $elemMatch: { score: 2 } },
    { $elemMatch: { score: 6 } }
   ]
  },
  cuisine: { $ne: "American" }
 }
);
```

**22. Write a MongoDB query to find the restaurants that have a grade with a score of 2 anda grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.**

```
db.restaurants.find(
 {
  borough: { $in: ["Manhattan", "Brooklyn"] },
  grades: {
   $all: [
    { $elemMatch: { score: 2 } },
    { $elemMatch: { score: 6 } }
   ]
  },
  cuisine: { $nin: ["American", "Chinese"] }
 }
);
```

**23. Write a MongoDB query to find the restaurants that have a grade with a score of 2 ora grade with a score of 6.**

```
db.restaurants.find(
  {
    $or: [
      { "grades.score": 2 },
      { "grades.score": 6 }
    ]
  }
);
```

**SAMPLE OUTPUT:-**


```
{
  _id: ObjectId('671b5d549d3d63480e0a64e9'),
  address: {
    building: 2233,
    coord: [
      -73.858177,
      40.849447
    ],
    street: '5th Avenue',
    zipcode: 10467
  },
  borough: 'Bronx',
  cuisine: 'American',
  grades: [
    {
      date: 2014-03-03T00:00:00.005Z,
      grade: 'A',
      score: 10
    },
    {
      date: 2013-09-11T00:00:00.005Z,
      grade: 'A',
      score: 6
    },
    {
      date: 2013-01-24T00:00:00.005Z,
```

```
    grade: 'B',
    score: 12
  },
  {
    date: 2011-11-23T00:00:00.005Z,
    grade: 'A',
    score: 9
  },
  {
    date: 2011-03-10T00:00:00.005Z,
    grade: 'A',
    score: 14
  }
],
name: 'Burger Bistro',
restaurant_id: 30075450
}

{
_id: ObjectId('671b5dab56ec9972ca8f5daf'),
address: {
  building: 4455,
  coord: [
    -73.858977,
    40.849847
  ],
  street: '7th Avenue',
  zipcode: 10469
},
borough: 'Bronx',
cuisine: 'Thai',
grades: [
  {
    date: 2014-03-03T00:00:00.007Z,
    grade: 'A',
    score: 9
  },
  {
    date: 2013-09-11T00:00:00.007Z,
    grade: 'B',
    score: 6
  },
  {
    date: 2013-01-24T00:00:00.007Z,
```

```
    grade: 'A',
    score: 12
  },
  {
    date: 2011-11-23T00:00:00.007Z,
    grade: 'A',
    score: 8
  },
  {
    date: 2011-03-10T00:00:00.007Z,
    grade: 'B',
    score: 14
  }
],
name: 'Thai Delight',
restaurant_id: 30075452
}
```

# MOVIES COLLECTION

**1. Find all movies with full information from the 'movies' collection that released in the year 1893.**

```
db.movies.find({ year: 1893 });
```

**2. Find all movies with full information from the 'movies' collectionthat have a runtime greater than 120 minutes.**

```
db.movies.find({ runtime: { $gt: 120 } });
```

**SAMPLE OUTPUT:-**


```
{
  _id: ObjectId('573a1390f29313caabcd42ec'),
  plot: 'An astronaut stranded on Mars must survive alone.',genres:
  [
    'Sci-Fi',
    'Drama'
  ],
  runtime: 135,
  cast: [
    'Matt Damon',
    'Jessica Chastain'
  ],
  poster: 'https://m.media-amazon.com/images/poster4.jpg',title:
  'Mars Alone',
  fullplot: 'An astronaut, left alone on Mars, struggles to survive with limited resources while awaiting rescue.',
  languages: [
```

```
  'English'
],
released: 2015-10-02T00:00:00.000Z,
directors: [
  'Ridley Scott'
],
rated: 'PG-13',
awards: { wins:
8,
  nominations: 6,
  text: '8 wins & 6 nominations.'
},
lastupdated: '2021-08-09 17:22:30.000000000',
year: 2015,
imdb: {
rating: 8,
  votes: 25650,
  id: 443
},
countries: [
  'USA'
],
type: 'movie',
tomatoes: {
viewer: {
    rating: 4.5,
    numReviews: 2201,
    meter: 93
  },
  fresh: 18,
  critic: {
  rating: 8.5,
    numReviews: 25,
    meter: 96
  },
```

```
    rotten: 1,
    lastUpdated: 2021-07-19T21:20:55.000Z
  }
}
```

**3. Find all movies with full information from the 'movies' collectionthat have "Short" genre.**

```
db.movies.find({ genres: "Short" });
```

**SAMPLE OUTPUT:-**

```
{
  _id: ObjectId('573a1390f29313caabcd42e8'),
  plot: 'A group of bandits stage a brazen train hold-up, only to find a
determined posse hot on their heels.',
  genres: [
    'Short',
    'Western'
  ],
  runtime: 11,
  cast: [
    'A.C. Abadie',
    "Gilbert M. 'Broncho Billy' Anderson",
    'George Barnes',
    'Justus D. Barnes'
  ],
  poster: 'https://m.media-
amazon.com/images/M/MV5BMTU3NjE5NzYtYTYyNS00MDVmLWIwYjg
tMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SY1
000_SX677_AL_.jpg',
  title: 'The Great Train Robbery',
  fullplot: "Among the earliest existing films in American cinema - notable as
the first film that presented a narrative story to tell - it depicts a group of
cowboy outlaws who hold up a train and rob the
```

passengers. They are then pursued by a Sheriff's posse. Severalscenes have color included - all hand tinted.",
languages: [
'English'
],
released: 1903-12-01T00:00:00.000Z,
directors: [ 'Edwin
S. Porter'
],
rated: 'TV-G',
awards: {
wins: 1,
nominations: 0,
text: '1 win.'
},
lastupdated: '2015-08-13 00:27:59.177000000',
year: 1903,
imdb: {
rating: 7.4,
votes: 9847,
id: 439
},
countries: [
'USA'
],
type: 'movie',
tomatoes: {
viewer: {
rating: 3.7,
numReviews: 2559,
meter: 75
},
fresh: 6,
critic: {
rating: 7.6,

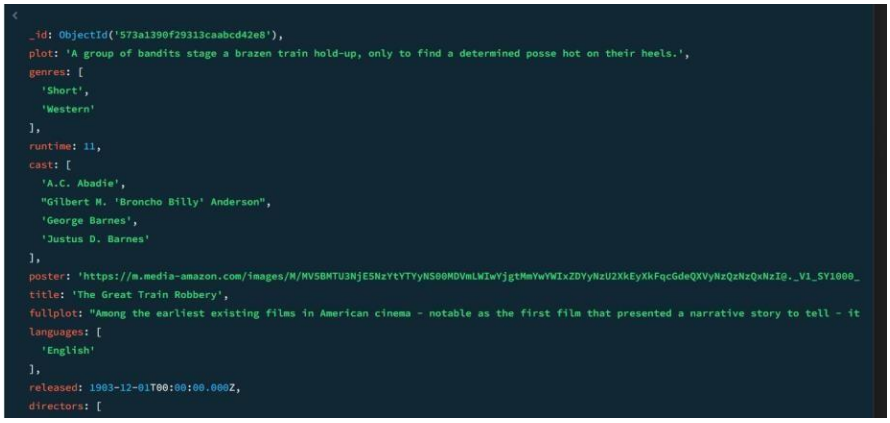
```
    numReviews: 6,
    meter: 100
   },
   rotten: 0,
   lastUpdated: 2015-08-08T19:16:10.000Z
  }
}
```

**4. Retrieve all movies from the 'movies' collection that were directed by "William K.L. Dickson" and include complete information for eachmovie.**

```
db.movies.find({ directors: "William K.L. Dickson" });
```

**6. Retrieve all movies from the 'movies' collection that were releasedin the USA and include complete information for each movie.**

```
db.movies.find({ countries: "USA" });
```




**7. Retrieve all movies from the 'movies' collection that have complete information and are rated as "UNRATED".**

```
db.movies.find({ rated: "UNRATED" });
```

**8. Retrieve all movies from the 'movies' collection that have complete information and have received more than 1000 votes on IMDb.**

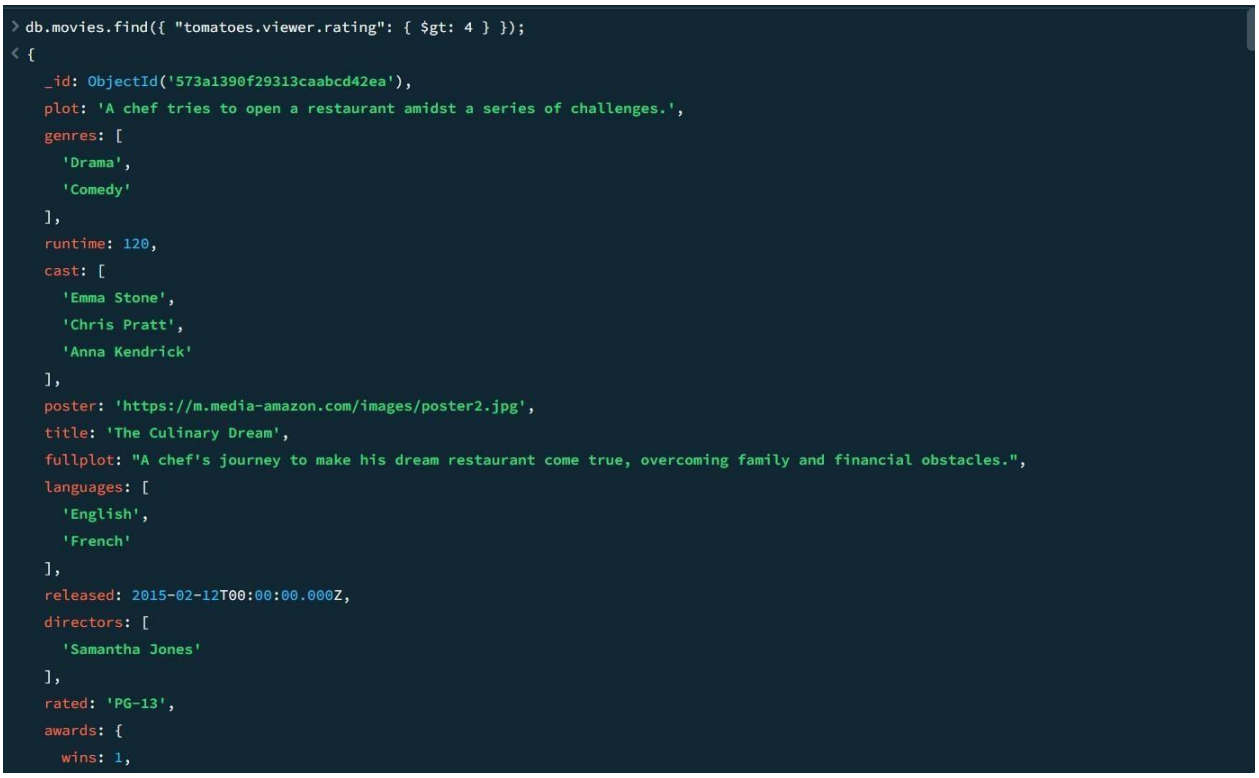
```
db.movies.find({ "imdb.votes": { $gt: 1000 } });
```


**9. Retrieve all movies from the 'movies' collection that have complete information and have an IMDb rating higher than 7.**

```
db.movies.find({ "imdb.rating": { $gt: 7 } });
```


**10. Retrieve all movies from the 'movies' collection that have complete information and have a viewer rating higher than 4 onTomatoes.**

**db.movies.find({ "tomatoes.viewer.rating": { $gt: 4 } });**




**11. Retrieve all movies from the 'movies' collection that have receivedan award.**

**db.movies.find({ "awards.wins": { $gt: 0 } });**


**12. Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB that have at least one nomination.**

```
db.movies.find(
   { "awards.nominations": { $gt: 0 } },
   {
      title: 1,
      languages: 1,
      released: 1,
      directors: 1,
      writers: 1,
      awards: 1,
      year: 1,
      genres: 1,
      runtime: 1,
      cast: 1,
      countries: 1
```

```
    }
);
```

```
>_MONGOSH
),
< {
    _id: ObjectId('573a1390f29313caabcd42e9'),
    genres: [
      'Adventure',
      'Fantasy'
    ],
    runtime: 95,
    cast: [
      'Ethan Hawke',
      'Jane Doe',
      'Mark Strong'
    ],
    title: 'The Amulet Quest',
    languages: [
      'English'
    ],
    released: 2008-07-15T00:00:00.000Z,
    directors: [
      'John Smith'
    ],
    awards: {
      wins: 2,
      nominations: 1,
      text: '2 wins & 1 nomination.'
    },
    year: 2008,
    countries: [
      'USA'
```

**13. Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB with cast including "Charles Kayser".**

```
db.movies.find(
    { cast: "Charles Kayser" },
    {
        title: 1,
        languages: 1,
        released: 1,
        directors: 1,
        writers: 1,
        awards: 1,
        year: 1,
```

```
        genres: 1,
        runtime: 1,
        cast: 1,
        countries: 1
    }
);
```

**14. Retrieve all movies with title, languages, released, directors, writers, countries from the 'movies' collection in MongoDB that released on May 9, 1893.**

```
db.movies.find(
    { released: ISODate("1893-05-09T00:00:00Z") },
    {
        title: 1,
        languages: 1,
        released: 1,
        directors: 1,
        writers: 1,
        countries: 1
    }
);
```

**14. Retrieve all movies with title, languages, released, directors, writers, countries from the 'movies' collection in MongoDB that have aword "scene" in the title.**

```
db.movies.find(
    { title: { $regex: /scene/i } },
    {
        title: 1,
        languages: 1,
```

```
        released: 1,
        directors: 1,
        writers: 1,
        countries: 1
    }
);
```