

EXERCISE 18

Structure of 'restaurants' collection:

```
{  
    "address": {  
        "building": "1007",  
        "coord": [ -73.856077, 40.848447 ],  
        "street": "Morris Park Ave",  
        "zipcode": "10462"  
    },  
    "borough": "Bronx",  
    "cuisine": "Bakery",  
    "grades": [  
        { "date": { "$date": 1393804800000 }, "grade": "A", "score": 2 },  
        { "date": { "$date": 1378857600000 }, "grade": "A", "score": 6 },  
        { "date": { "$date": 1358985600000 }, "grade": "A", "score": 10 },  
        { "date": { "$date": 1322006400000 }, "grade": "A", "score": 9 },  
        { "date": { "$date": 1299715200000 }, "grade": "B", "score": 14 }  
    ],  
    "name": "Morris Park Bake Shop",  
    "restaurant_id": "30075445"  
}
```

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'.

\$or : I
[cuisine: { \$min: ["American", "chinese"] }] ,

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..

"grades": {
 { elementMatch: { "grade": "A", "score": 11, "date": ISODate("2014-08-11T00:00:00Z") } } } ,

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

{ "grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") } ,

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

{ "address.coord.1": { \$gt: 42, \$lt: 52 } },
{ restaurant_id: 1, name: 1, address: 1, -1 }] }

which is more than 42 and upto 52..

5. 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});

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

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

7. 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});

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

{ "address.street": { \$exists: false } }

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

{ "address.coord": { \$type: "double" } }

10. 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.

{ "grade.score": { \$mod: [7, 0] } },
{ restaurant_id: 1, name: 1, grade: 1, _id: 0 }

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 its name.

{ name: /^{mon}\w{2}/ },
{ name: 1, borough: 1, "address.coord": 1, cuisine: 1, _id: 0 }

12. 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.

{ name: /^{Mad}\w{2}/ },
{ name: 1, borough: 1, "address.coord": 1, cuisine: 1, _id: 0 }

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

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

14. 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.

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

15. 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.

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, and their cuisine is not American.

{ "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, and their cuisine is not American or Chinese.

{ "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 and a grade with a score of 6.

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

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

{ "grades.score": { \$all: [2, 6] },
borough: { \$in: ["Manhattan"] } }

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

db.restaurants.find({ "grades.score": { \$all: [2, 6] },
borough: { \$in: ["Manhattan", "Brooklyn"] } })

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

{ "grade.score": { \$all: [2,6] },
borough: { \$in: ["Manhattan", "Brooklyn"] },
cuisine: { \$ne: "American" } }

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

{ "grade.score": { \$all: [2,6] },
borough: { \$in: ["Manhattan",
"Brooklyn"] },
cuisine: { \$ne: "Chinese" } }

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

db.restaurants.find(
{ "grade.score": { \$in: [2,6] } })
;

Sample document of 'movies' collection

```
{  
    _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/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcG  
deQXVyNzQzNzQxNzl@._V1_SY1000_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. Several scenes have color included - all hand tinted."}
```

```
languages: [ 'English' ],  
released: ISODate("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: ISODate("2015-08-08T19:16:10.000Z")  
}
```

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' collection that have a runtime greater than 120 minutes.

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

3. Find all movies with full information from the 'movies' collection that have "Short" genre.

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

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

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

5. Retrieve all movies from the 'movies' collection that were released in the USA and include complete information for each movie.

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

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

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

7. 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}}});

8. 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}}});

9. Retrieve all movies from the 'movies' collection that have complete information and have a viewer rating higher than 4 on Tomatoes.

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

10. Retrieve all movies from the 'movies' collection that have received an award.

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

11. 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.

*{ "awards.nomination": {\$gt: 0},
{ title: 1, language: 1, released: 1, director: 1, awards: 1, year: 1, genre: 1,
runtime: 1, cast: 1, countries: 1, -id: 0 } }*

12. Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB with cast

db.movies.find(

{ cast: "Charles Kaysen"},

*{ title: 1, language: 1, released: 1, director: 1, awards: 1, year: 1,
genre: 1, runtime: 1, cast: 1, countries: 1, -id: 0 }*

);

including "Charles Kayser".

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

{
 released: ISODate("1893-05-09T00:00:00Z"),
 { title: 1, language: 1, released: 1, directors: 1, writers: 1, -id: 0 }
}

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

```
db.movies.find(  
  { title: / scene/i }  
  { title: 1, language: 1, released: 1, directors: 1, writers: 1,  
    -id: 0 }  
,
```