Q1) db.restaurants.find({ $or : [{ cuisine : {$nin : ["American", "Chinese"]}},
{name : { $regex : /^Wil/}}]},
{restaurant_id : 1, name : 1, borough : 1, cuisine : 1});

Q2) db.restaurants.find({ "grades" : {$elemMatch : {grade : "A", score : 11,
date : ISODate("2014-08-11T00:00:00Z")}}},
{restaurant_id : 1, name : 1, grades : 1});

Q3) db.restaurants.find({ "grades.1" : {grade : "A", score : 9, date : ISODate
("2014-08-11T00:00:00Z")}}, {restaurant_id : 1, name : 1, grades : 1});

Q4) db.restaurants.find({ "address.coord.1" : { $gt : 42, $lte : 52 }},
{restaurant_id : 1, name : 1, address : 1, "address.coord" : 1});

Q5) db.restaurants.find().sort({ name : 1 });

Q6) db.restaurants.find().sort({ name : -1 });

Q7) db.restaurants.find().sort({ cuisine : 1, borough : -1 });

Q8) db.restaurants.find({ "address.street" : {$exists : true }});

Q9) db.restaurants.find({ "address.coord" : { $type : "double" }});

Q10) db.restaurants.find({ "grades" : {$elemMatch : {score : {$mod : [7, 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 9 on an 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 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.

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

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.

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

Q11) db.restaurant.find ({ name: { $regex: /mon/i } },
{ name: 1, borough: 1, "address.coord": 1, cuisine: 1 });

Q12) db.restaurant.find ({ name: { $regex: /^Mad/i } },
{ name: 1, borough: 1, "address.coord": 1, cuisine: 1 });

Q13) db.restaurant.find ({ "grades.score": { $lt: 5 } });

Q14) db.restaurant.find ({ "grades.score": { $lte: 5 } });

Q15) db.restaurant.find ({ "grades.score": { $lte: 5 }, borough: "Manhattan" });

Q16) db.restaurant.find ({ "grades.score": { $lt: 5 },
borough: { $in: ["Manhattan", "Brooklyn"] },
cuisine: { $ne: "American" } });

Q17) db.restaurant.find ({ "grades.score": { $lt: 5 },
borough: { $nin: ["Manhattan", "Brooklyn"] },
cuisine: { $in: ["American", "Chinese"] } });

Q18) db.restaurants.find ({ grades: { $elemMatch: { $and: [{ score: { $lte: 5 } },
{ $elemMatch: { score: 2 } }] }, borough: "Manhattan" });

Q19) db.restaurants.find ({ grades: { $all: [{ $elemMatch: { score: 6 } }],
borough: "Manhattan" } },
{ score: 2 } }, { $elemMatch: { score: 6 } }],
"Manhattan" });

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

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.

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.

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

Q20) db.restaurant.find({ { grades: { $all: [{ $elemMatch :
{score: 2} }, { $elemMatch : {score : 6}}}},
borough: { $in: ["Manhattan", "Brooklyn"]},
cuisine: { $ne: "American"} });

Q21) db.restaurant.find({ { grades: { $all: [{ $elemMatch : {score : 2}},
{$elemMatch: {score : 6}}}, borough : {$in : ["Manhattan",
"Brooklyn"]}, cuisine: {$ne : "American"}});

Q22) db.restaurant.find({ grades: {$all: [{ $elemMatch : {score:2}
}, { $elemMatch : {score:6} }},
borough : { $in : ["Manhattan", "Brooklyn"]},
cuisine : { $in : ["American", "Chinese"]}});

Q23) db.restaurants.find({ $or: [{"grades.score": 2},
{"grades.score": 6}]});

---

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.

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.

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.

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.

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.

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.

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.

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.

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.

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'
    ],
}
```

Q1) db.movies.find ({? year : 1893})

Q2) db.movies.find ({runtime : {$gt : 120} })

Q3) db.movies.find ({ genres : "Short" })

Q4) db.movies.find ({ directors : "William K. L. Dickson" })

Q5) db.movies.find ({ countries : "USA" })

Q6) db.movies.find ({ "imdb.votes" : {$gt : 1000 } })

Q7) db.movies.find ({ rated : "UNRATED" })

Q8) db.movies.find ({ "imdb.rating" : {$gt : 7.3} })

Q9) db.movies.find ({ "tomatoes.viewer.rating" : {$gte : 4 } })

Q10) db.movies.find ({ "award.wins" : {$gt : 0} })

Q11) db.movies.find ({ "award.nominations" : {$gte : 1 } },
{title : 1, language : 1, released : 1, directors : 1, writers : 1,
awards : 1, year : 1, genres : 1, runtime : 1, cast : 1,
countries : 1 })

Q12) db.movies. find ({ cast : "Charles Kayser" }).
{ title : 1, language : 1, released : 1, director : 1,
writers : 1, awards : 1, year : 1, genres : 1, runtime : 1,
cast : 1, countries : 1 })

Q13) db. movies. find ({ { released : ISODate (" 1893-05-01 T00:00:00.
000Z") } },
{ title : 1, language : 1, released : 1, director : 1, writers : 1,
countries : 1 })

Q14) db.movies. find ({ title : { $regex : "scene", $options: { $ "" } } },
{ title : 1, language : 1, released : 1, director : 1, writers : 1,
countries : 1 }).

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

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

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

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

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

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

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

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

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.

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".

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

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.

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	5
Program/Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	