MongoDB – Complex Queries Assignment

Exercise Questions

1. Write a MongoDB query to display all the documents in the collection restaurants.

- **db.addresses.find().pretty()**

2. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.

- **db.addresses.aggregate([{$project: {_id:1,name:1,borough:1,cuisine:1}}]).pretty()**

3. 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.

- **db.addresses.aggregate([{$project: {restaurant_id:1,name:1,borough:1,cuisine:1,_id:0}}]).pretty()**

4. Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

- **db.addresses.aggregate([{$project:{_id:0,restaurant_id:1,name:1,"address.zipcode":1}}]).pretty()**

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

- **db.addresses.aggregate([{$match:{borough:"Bronx"}}, {$limit:5}]).pretty()**

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

- **db.addresses.aggregate([{$match:{borough:"Bronx"}}]).pretty()**

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

- **db.addresses.aggregate([{$match:{borough:"Bronx"}},{$skip:5}, {$limit:5}]).pretty()**

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

- **db.addresses.aggregate([{$unwind:"$grades"},{$group:{_id: {name:"$name"},sum:{$sum:"$grades.score"}}},{$match:{sum: {$gt:90}}}]).pretty()**

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

- **db.addresses.aggregate([{$unwind:"$grades"},{$group:{_id: {name:"$name"},sum:{$sum:"$grades.score"}}},{$match:{sum: {$gt:80,$lt:100}}}]).pretty()**

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

- **db.addresses.find({"address.coord.0":{$lt: -90}}).pretty()**

11. 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.addresses.find({$and:[{"cuisine":{$ne:"American"}},{"grades.score": {$gt:70}},{"address.coord.0":{$lt:-65.754168}}]}).pretty()**

12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168.

- **db.addresses.find({$and:[{"cuisine":{$ne:"American"}},{"grades.score": {$gt:70}},{"address.coord.0":{$lt:-65.754168}}]}).pretty()**

13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.

- **db.addresses.find({$and:[{"cuisine":{$ne:"American"}}, {"grades.grade":"A"},{"borough": {$ne:"Brroklyn"}}]}).sort({"cuisine":-1}).pretty()**

14. 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.addresses.find({name:{$regex:/Wil/i}},**
  {"restaurant_id":1,"name":1,"borough":1,"cuisine":1}).pretty()

15. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.

- **db.addresses.find({name:{$regex:/ces$/}},**
  {"restaurant_id":1,"name":1,"borough":1,"cuisine":1}).pretty()

16. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

- **db.addresses.find({name:{$regex:/Reg/i}},**
  {restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

- **db.addresses.find({"borough":"Bronx",$or:[{"cuisine":"American"},**
  {"cuisine":"Chinese"}]}).pretty()

18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.

- **db.addresses.find({"borough":{$in:**
  ["StatenIsland","Queens","Bronx","Brooklyn"]}},
  {"restaurant_id":1,"name":1,"borough":1,"cuisine":1}).pretty()

19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

- **db.addresses.find({"borough":{$nin:**
  ["StatenIsland","Queens","Bronx","Brooklyn"]}},
  {"restaurant_id":1,"name":1,"borough":1,"cuisine":1}).pretty()

20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.

- **db.addresses.find({"grades.score":{$lt:10}}, {"restaurant_id":1,"name":1,"borough":1,"cuisine":1}).pretty ()**

21. 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.addresses.find({$or:[{name:{$regex:/^Wil/i}},{$and:[{"cuisine": {$ne:"American"}},{"cuisine":{$ne:"Chinese"}}]}], {restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()**

22. 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.addresses.find({"grades.date":ISODate("2014-08-11T00:00:00Z"),"grades.grade":"A","grades.score":11},{"restaurant_id":1,"name":1,"grades":1}).pretty()**

23. 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"

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

24. 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.addresses.find({ "address.coord.1": {$gt : 42, $lte : 52}}, {"restaurant_id" :1,"name":1,"address":1,"coord":1});**

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

- **db.addresses.aggregate({$sort:{name:1}}).pretty()**

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

- **db.addresses.aggregate({$sort:{name:-1}}).pretty()**

27. 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.addresses.find().sort({"cuisine":1,"borough" : -1,});**

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

- **db.addresses.find({"address.street" : { $exists : true }});**

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

- **db.addresses.find({"address.coord" : {$type : 1} });**

30. 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.

- **db.addresses.find({"grades.score" :{$mod : [7,0]}}, {restaurant_id : 1, name : 1, grades : 1});**

31. 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.

- **db.addresses.find({ name :{ $regex : "mon.*", $options: "i" } }, { "name":1, "borough":1,"address.coord":1, "cuisine" :1 });**

32. 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.

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

Happy Coding!!!