MONGO DB – ASSIGNMENT - 3

Connect with Compass app -> create a database -> enter db name(restaurants) and collection name(addresses) ->create -> import data ->select JSON and browse the zip file(restaurants.json) -> import

```
C:\Users\ABC>mongosh "mongodb+srv://cluster0.b3io6.mongodb.net/Cluster0" --username Mrunal -password Mrunal1222
```

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.find({}, {restaurant_id:1, name:1,borough:1, cuisine:1 });
```

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.find({}, {restaurant_id:1, _id:0, name:1,borough:1, cuisine:1 });
```

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.find({}, {"restaurant_id":1, _id:0, "name":1,"borough":1, "address.zipcode":1 });
```

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

```
db.addresses.find({"borough":"Bronx"}).limit(5);
```

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

db.addresses.find({"borough":"Bronx"});

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

db.addresses.find({"borough":"Bronx"}).limit(5).skip(5);

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

db.addresses.find({"grades.score":{$gt:90}});

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

db.addresses.find({"grades.score":{$gt:80, $lt:100}});

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

db.addresses.find({"address.coord":{$lt : -95.754168}});

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 "}}, {"address.coord.0" : {$lt : -65.754168}}, {"grades.score" : {$gt : 70}}]});

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({"cuisine" : {$ne : "American "}, "grades.score" :{$gt: 70},"address.coord" : {$lt : -65.754168}});

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 : "Brooklyn "}}]}).sort({cuisine : -1});

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.*/}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

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" : { $regex: /.*ces$/}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

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/}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

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", cuisine: {$in: ["American ","Chinese"]}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

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 Bronx or Brooklyn.

db.addresses.find({$or: [{"borough": "Staten Island"}, {"borough": "Bronx or Brooklyn"}, {"borough": "Queens"}]}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

db.addresses.find({borough: {$in: ["Staten Island","Queens","Bronx","Brooklyn"]}} , {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

<u>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.</u>

db.addresses.find({borough: {$nin: ["Staten Island","Queens","Bronx","Brooklyn"]}} , {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

<u>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.</u>

db.addresses.find({"grades.score": {$lte: 10}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

<u>21. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinese' or restaurant's name begins with letter 'Wil'.</u>

db.addresses.find({$nor: [{cuisine: {$in: ["American ","Chinese"]}},{name: /^Wil.*/}]},{_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});

<u>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..</u>

db.addresses.find({"grades" : {$elemMatch: {"date": ISODate("2014-08-11T00:00:00Z"), "grade":"A", "score":11}}}, {_id:0, restaurant_id:1, name:1, grades:1});

<u>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"</u>

db.addresses.find({$and: [{"grades.1.grade":"A"}, {"grades.1.score": 9}, {"grades.1.date": ISODate("2014-08-11T00:00:00Z")}]},{_id:0, 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({$and : [{"address.coord.1": {$gt : 42}},{"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1});

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

db.addresses.find({},{_id:0, name:1}).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.find({},{_id:0, name:1}).sort({name: -1});

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({}, {_id:0, cuisine:1, borough:1}).sort({borough:-1, cuisine: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: "double"}}, {_id:0, address:1});

<u>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.</u>

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

<u>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.</u>

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

<u>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.</u>

```
db.addresses.find({name: {$regex: /^Mad.*/}},{_id:0, name:1, borough:1, "address.coord":1, cuisine:1}).pretty();
```