**(Name:Aniket)**

**Exercise Questions**

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

Ans: db.getCollection('addresses').find({})

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

Ans: db.getCollection('addresses').aggregate([

{$group:{\_id:{restId:"$\_id",name:"$name",borough:"$borough",cuisine:"$cuisine"}}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$project:{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$project:{\_id:0,restaurant\_id:1,name:1,borough:1,"address.zipcode":1}}])

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

Ans: db.getCollection('addresses').aggregate([

{$match:{borough:"Bronx"}},

{$limit:5}])

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

Ans:. db.getCollection('addresses').aggregate([

{$match:{borough:"Bronx"}}])

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

Ans: db.getCollection('addresses').aggregate([

{$match:{borough:"Bronx"}},

{$skip:5},

{$limit:5}])

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

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{"grades.score":{$gt:90}}}])

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

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{"grades.score":{$gt:80,$lt:100}}}])

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

Ans: db.getCollection('addresses').aggregate([

{$match:{"address.coord.0":{$lt:-95.754168}}},

/\*{$sort:{"address.coord":1}}\*/])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{$and:[{cuisine:{$ne:"American "},

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

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{$and:[{cuisine:{$ne:"American "},"grades.score":{$gt:70},"address.coord.1":{$lt:-65.754168}}]}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{$and:[{cuisine:{$ne:"American "},"grades.grade":"A",borough:{$ne:"Brooklyn"}}]}},

{$sort:{cuisine:-1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{name:{$regex:/^Wil.\*/}}},

{$project:{restaurant\_id:1,name:1,borough:1,cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{name:{$regex:/.\*ces$/}}},

{$project:{restaurant\_id:1,name:1,borough:1,cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{name:{$regex:/Reg/}}},

{$project:{restaurant\_id:1,name:1,borough:1,cuisine:1}}])

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

Ans: db.getCollection('addresses').aggregate([

{$match:{borough:"Bronx",cuisine:{$in:["American ","Chinese"]}}},

{$project:{name:1,cuisine:1,borough:1}},

{$sort:{cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{$or:[{borough:'Staten Island'}, {"borough": "Bronxor Brooklyn"}, {"borough": "Queens"}]}},

{$project:{name:1,cuisine:1,borough:1,restaurant\_id:1}},

{$sort:{cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{borough:{$nin:["Staten Island","Queens","Bronxor Brooklyn"]}}},

{$project:{name:1,cuisine:1,borough:1,restaurant\_id:1}},

{$sort:{cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{"grades.score":{$lt:10}}},

{$project:{name:1,cuisine:1,borough:1,restaurant\_id:1,'grades.score':1}},

{$sort:{cuisine:1}}])

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'.

Ans: db.getCollection('addresses').aggregate([

{$match:{$or:[{cuisine:{$nin:["American ","Chinese"]}},{name:{$regex:/^Wil/}}]}},

{$project:{name:1,cuisine:1,borough:1,restaurant\_id:1}},

{$sort:{cuisine:1}}])

1. 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..

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

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

{$project:{name:1,grades:1,restaurant\_id:1}},

{$sort:{cuisine:1}}])

1. 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"

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

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

{$project:{name:1,grades:1,restaurant\_id:1}},

{$sort:{cuisine:1}}])

1. 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..

Ans: db.getCollection('addresses').aggregate([

{$match:{$and:[{"address.coord.1":{$gt:42}},{"address.coord.1":{$lt:52}}]}},

{$project:{name:1,address:1,restaurant\_id:1}}])

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

Ans: db.getCollection('addresses').aggregate([

{$sort:{name:1}},

{$project:{name:1,\_id=0}}])

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

Ans: db.getCollection('addresses').aggregate([

{$sort:{name:-1}},

{$project:{name:1,\_id=0}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$sort:{cuisine:1,borough:-1}},

{$project:{cuisine:1,borough:1}}])

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

Ans: db.getCollection('addresses').aggregate([

{$match:{"address.street":{$regex:/Street/}}},

{$project:{"address.street":1}}])

Without street

db.getCollection('addresses').aggregate([

{$match:{"address.street":{$ne:{$regex:/Street/}}}},

{$project:{"address.street":1}}])

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

Ans; db.getCollection('addresses').aggregate([

{$match:{"address.coord":{$type:"double"}}},

{$project:{"address.coord":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 b

Ans: db.getCollection('addresses').aggregate([

{$unwind:"$grades"},

{$match:{"grades.score":{$mod:[7,0]}}},

{$project:{restaurant\_id:1,name:1,"grades":1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{name:{$regex:/mon/}}},

{$project:{restaurant\_id:1,name:1,borough:1,"address.coord":1,cuisine:1}}])

1. 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.

Ans: db.getCollection('addresses').aggregate([

{$match:{name:{$regex:/^Mad.\*/}}},

{$project:{restaurant\_id:1,name:1,borough:1,"address.coord":1,cuisine:1}}])