**EXERCISE QUESTIONS: -**

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

* db.restaurants.find().pretty()

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

* db.restaurants.find( {} ,

{“restaurant\_id” : 1 , “name”: 1 , “borough”: 1,”cuisine”:1}

).pretty()

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.

* db.restaurants.find( {} ,

{“restaurant\_id” : 1 , “name”: 1 , “borough”: 1,”cuisine”:1,”\_id”:0}

).pretty()

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.

* db.restaurants.find( {} ,

{“restaurant\_id” : 1 , “name”: 1 , “borough”: 1,”zip code”:1,”\_id”:0}

). pretty()

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

* db.restaurants.find(

{“borough”: “Bronx”}).limit(5).pretty()

1. Write a MongoDB query to display all the restaurants which is in the borough “Bronx”

* db.restaurants.find(

{“borough”: “Bronx”}).pretty()

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

* db.restaurants.find(

{“borough”: “Bronx”}).skip(5).limit(5).pretty()

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

* db.restaurants.find(

{grade: { $elemMatch: {

“score”:{$gt : 90}

}}}).pretty()

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

* db.restaurants.find(

{grades : {$elemMatch :

{“score”: {$gt: 80 , $lt: 100}

}}}). pretty()

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

* db.restaurants.find(

{“address.coord” : {$lt: -95.754168}

}). pretty()

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

* db.restaurants.find(

{$and: [

{“cuisine”: {$ne: “American”}},

{“grades.score”: {$gt: 70}},

{“address.coord”: {$lt: -65.754168}}

]}).pretty()

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 locate in the longitude less than -65.754168

* db.restaurants.find(

{“cuisine” : {$ne: “American”},

“grades.score” :{gt: 70},

“address.coord”: {lt : -65.754168}

}).pretty()

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.

* db.restaurants.find(

{“cuisine” :{$ne: “American”},

“grades.grade” : “A” ,

“borough” : {$ne: “Brooklyn”}

}).sort({“cuisine”: -1}).pretty()

1. Write a MongoDB query to find the restaurants Id, name, borough and cuisine for those restaurants which contains “Wil” as first three letters for its name.

* db.restaurants.find(

{name: /^Wil/},

{“restaurant\_id”: 1,

“name”: 1,

“borough”:1,

“cuisine”:1 }).pretty()

1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contains ‘ces’ as last three letters for its name.

* db.restaurants.find(

{name: /ces$/},

{“restaurant\_id”: 1,

“name”: 1,

“borough”:1,

“cuisine”:1 }).pretty()

1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contains ‘reg’ as three letters somewhere in its name.

* db.restaurants.find(

{name: /.\*Reg.\* /},

{“restaurant\_id”: 1,

“name”: 1,

“borough”:1,

“cuisine”:1 }).pretty()

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

* db.restaurants.find(

{“borough”: “Bronx”,

$or: [ {“cuisine”: “American”},

{“cuisine”: “Chinease”}]

}).pretty()

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

* db.restaurants.find(

{“borough”: {$in: [“Staten Island”,”Queens”.”Bronx”,”Brooklyn”]}},

{“restaurant\_id”: 1 , “ name”: 1, “borough”: 1, “cuisine”:1}).pretty()

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”

* db.restaurants.find(

{“borough”: {$in: [“Staten Island”,”Queens”.”Bronx”,”Brooklyn”]}},

{“restaurant\_id”: 1 , “ name”: 1, “borough”: 1, “cuisine”:1}).pretty()

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

* db.restaurants.find(

{“grades.score”: {$not: {$gt:10}}},

{“restaurant\_id”: 1 , “name”: 1, “borough”:1 , “cuisinr”:1}).pretty()

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”

* db.restaurants.find(

{$or: [name: /^Wil/},

{“$and”: [{“cuisine”: {$ne: “American”}},

{“cuisine”:{$ne:”Chinees”}}]

}]},

{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1}).pretty()

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...

* db.restaurants.find(

{“grades.date”: ISODate(“2014-08-11T00:00:00Z”),

“grades.grade”: “A”,

“grades.score”: 11},

{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1}).pretty()

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"

* db.restaurants.find(

{“grades.1.date”: ISODate(“2014-08-11T00:00:00Z”),

“grades.1.grade”: “A”,

“grades.1.score”: 9},

{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1}).pretty()

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...

* Db.restaurants.find(

{“address.coord.1”: {$gt: 42 , $lte: 52}},

{“restaurant\_id”:1,”name”:1,”borough”:1,”cuisine”:1}).pretty()

1. 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}).pretty()

1. 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}).pretty()

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.

* db.restaurants.find().sort({“cuisine”:1,“borough”:-1}).pretty()

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

* db.restaurants.find(

{“address.street”: {$exists : true}}).pretty()

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

* db.restaurants.find(

{“address.coord”: {$type: 1}}).pretty()

1. Write a MongoDB query which will select restaurant id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

* db.restaurants.find(

{“grades.score”: {$mod:[7,0]}},

{“restaurant\_id”:1,”name”:1,”grades”:1 }).pretty()

1. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain ‘Mon’ as three letters somewhere in its name

* db.restaurants.find(

{name: {$regex: “mon.\*”, $options: ‘i’}},

{“name”:1,”borough”:1,”address.coord”:1,”cuisine”:1}).pretty()

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

* db.restaurants.find(

{name: {$regex: /^Mad/i,}},

{“name”:1,”borough”:1,”address.coord”:1,”cuisine”:1}).pretty()