## MongoDB – Complex Queries

## Mongo DB Exercises - With the Restaurants Data Set

## Download the restaurants.zip file

## Unzip the file, you will see restaurants.json file

## Run the mongod server

## Run the following command to import the json file provided. It will load the json file into the mongodb with database name - restaurants, collections name - addresses mongoimport --db restaurants --collection addresses --file restaurants.json

## Run mongo shell command

## show databases

## use restaurants

## db.addresses.find() should print entire json data

## Then start working on the following exercises and submit your queries as the answers to the questions

## Query Reference Links and Cheat sheets

## 1. <https://docs.mongodb.com/manual/crud/>

## Exercise Questions

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

## db.addresses.find()

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

## 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.

## 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.

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

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

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

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

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

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

## 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.

## 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.

## 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.

## 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.

## 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.

## 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.

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

## 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.

## 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.

## 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.

## 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'.

## 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..

## 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"

## 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..

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

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

## 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.

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

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

## 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.

## 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.

## 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.

## 

## Happy Coding!!!

## 

## //1 Collection is addresses and not restaurants

## db.addresses.find()

## //2

## db.addresses.find({},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

## //3

## db.addresses.find({},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1,"\_id":0})

## //4

## db.addresses.find({},{"restaurant\_id":1,"name":1,"borough":1,"address.zipcode":1,"\_id":0})

## //5

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

## //6

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

## //7

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

## //8

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

## //(or)

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

## //9

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

## //(or)

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

## //10

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

## //11

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

## //12

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

## //13

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

## //14

## db.addresses.find({name:/^Wil/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

## //15

## db.addresses.find({name:/ces$/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

## //16

## db.addresses.find({name:/.\*Reg.\*/},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

## //17

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

## //18

## db.addresses.find({$or:[{"borough":"Staten Island"},{"borough":"Quuens"},{"borough":"Bronx"},{"borough":"Brooklyn"}]},{"restaurant\_id":1,"name":1,"cuisine":1})

## //(or)

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

## //19

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

## //20

## db.addresses.find({"grades.score":{$lte:10}},{"restaurant\_id":1,"name":1,"borough":1,"cuisine":1})

## //21

## db.addresses.find( {$or: [{name: /^Wil/},{"$and": [{"cuisine" : {$ne :"American "}},{"cuisine" : {$ne :"Chinees"}}]}]} ,{"restaurant\_id" : 1,"name":1,"borough":1,"cuisine" :1})

## //22

## db.addresses.find({"grades.date":ISODate("2014-08-11T00:00:00Z"),"grades.grade":"A","grades.score":11},{"restaurant\_id":1,"name":1,"grades":1})

## //23

## 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})

## //24

## db.addresses.find({"address.coord.1":{$gt:42,$lte:52}},{"restaurant\_id":1,"name":1,"address":1,"coord":1})

## //25

## db.addresses.find().sort({"name":1})

## //26

## db.addresses.find().sort({"name":-1})

## //27

## db.addresses.find().sort({"cuisine":1,"borough":-1})

## //28

## db.addresses.find({"address.street":{$exists:true}})

## //29

## db.addresses.find({"address.ccord":{$type:1}})

## //30

## db.addresses.find({"grades.score":{$mod:[7,0]}},{"restaurant\_id":1,"name":1,"grades":1})

## //31

## db.addresses.find({name:/.\*mon.\*/},{"name":1,"borough":1,"address.coord":1,"cuisine":1})

## //32

## db.addresses.find({name:/^Mad/},{"name":1,"borough":1,"address.coord":1,"cuisine":1})