Tenim una col·lecció d'Objectes Restaurant a la ciutat de Nova York, i necessitem algunes consultes... pots ajudar-nos?

1. Escriu una consulta per mostrar tots els documents en la col·lecció Restaurants.

db.restaurants.find({})

1. Escriu una consulta per mostrar el restaurant\_id, name, borough i cuisine per tots els documents en la col·lecció Restaurants.

db.restaurants.find({},{ "restaurant\_id" : true, "name" : true, "borough" : true, "cuisine" : true })

1. Escriu una consulta per mostrar el restaurant\_id, name, borough i cuisine, però exclou el camp \_id per tots els documents en la col·lecció Restaurants.

db.restaurants.find({}, { restaurant\_id : true, name : true, borough : true, cuisine : true, \_id : false })

1. Escriu una consulta per mostrar restaurant\_id, name, borough i zip code, però exclou el camp \_id per tots els documents en la col·lecció Restaurants.

db.restaurants.find({}, { restaurant\_id : true, name : true, borough : true, "address.zipcode" : true, \_id : false })

1. Escriu una consulta per mostrar tots els restaurants que estan en el Bronx.

db.restaurants.find({ borough : "Bronx" })

1. Escriu una consulta per mostrar els primers 5 restaurants que estan en el Bronx.

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

1. Escriu una consulta per mostrar el pròxim 5 restaurants després de saltar els primers 5 del Bronx.

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

Llama a[skip()](https://www.mongodb.com/docs/manual/reference/method/cursor.skip/#mongodb-method-cursor.skip)método en un cursor para controlar dónde MongoDB comienza a devolver resultados. Este enfoque puede ser útil para implementar resultados paginados.

1. Escriu una consulta per trobar els restaurants que tenen un score de més de 90.

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

[$gt](https://www.mongodb.com/docs/manual/reference/operator/query/gt/#mongodb-query-op.-gt)selecciona aquellos documentos donde el valor de fieldes mayor que (es decir >) el especificado value.

1. Escriu una consulta per trobar els restaurants que tenen un score de més de 80 però menys que 100.

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

[$lt](https://www.mongodb.com/docs/manual/reference/operator/query/lt/#mongodb-query-op.-lt)selecciona los documentos donde el valor de fieldes menor que (es decir <) el especificado value.

1. Escriu una consulta per trobar els restaurants que es localitzen en valor de latitud menys de -95.754168.

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

1. Escriu una consulta de MongoDB per a trobar els restaurants que no preparen cap cuisine de 'American' i la seva qualificació és superior a 70 i longitud inferior a -65.754168.

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

[$ne](https://www.mongodb.com/docs/manual/reference/operator/query/ne/#mongodb-query-op.-ne)selecciona los documentos donde el valor de fieldno es igual al especificado value. Esto incluye documentos que no contienen el field.

1. Escriu una consulta per trobar els restaurants que no preparen cap cuisine de 'American' i van aconseguir un marcador més de 70 i localitzat en la longitud menys que -65.754168. **Nota**: Fes aquesta consulta sense utilitzar $and operador.

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

1. Escriu una consulta per trobar els restaurants que no preparen cap cuisine de 'American' i van obtenir un punt de grau 'A' no pertany a Brooklyn. S'ha de mostrar el document segons la cuisine en ordre descendent.

db.restaurants.find({ cuisine : { $ne :"American " }, borough : { $ne : "Brooklyn" }, "grades.grade" : "A" }).sort({ cuisine : -1 })

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que contenen 'Wil' com les tres primeres lletres en el seu nom.

db.restaurants.find({ name : /^Wil/ }, { restaurant\_id : true, name : true, borough : true, cuisine : true })

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que contenen 'ces' com les últimes tres lletres en el seu nom.

db.restaurants.find({ name : /ces$/ }, { restaurant\_id : true, name : true, borough : true, cuisine : true })

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que contenen 'Reg' com tres lletres en algun lloc en el seu nom.

db.restaurants.find({ name : /Reg/ }, { restaurant\_id : true, name : true, borough : true, cuisine : true })

1. Escriu una consulta per trobar els restaurants que pertanyen al Bronx i van preparar qualsevol plat americà o xinès.

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

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que pertanyen a Staten Island o Queens o Bronx o Brooklyn.

db.restaurants.find({ $or : [{ borough:"Queens" }, { borough:"Brooklyn" }, { borough:"Staten Island" }, { borough:"Bronx" }]}, { "borough" : true, "restaurant\_id" : true, "name" : true, "cuisine" : true })

[$nor](https://www.mongodb.com/docs/manual/reference/operator/query/nor/#mongodb-query-op.-nor)realiza una operación lógica NORen una matriz de una o más expresiones de consulta y selecciona los documentos que **fallan** en todas las expresiones de consulta de la matriz.

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que no pertanyen a Staten Island o Queens o Bronx o Brooklyn.

db.restaurants.find({ $nor : [{ borough:"Queens" }, { borough:"Brooklyn" }, { borough:"Staten Island" }, { borough:"Bronx" }]}, { "borough" : true, "restaurant\_id" : true, "name" : true, "cuisine" : true })

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que aconsegueixin un marcador que no és més de 10.

db.restaurants.find({ "grades.score" : { $lt : 10 }}, { "borough" : true, "restaurant\_id" : true, "name" : true, "cuisine" : true })

1. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que preparen peix excepte 'American' i 'Chinees' o el name del restaurant comença amb lletres 'Wil'.

db.restaurants.find( { $or : [{ name : /^Wil/ }, { $and : [{ cuisine : { $ne : "Chinese"}}, { cuisine : { $ne : "American " }} ] } ] }, { restaurant\_id : true, name : true, borough : true, cuisine : true })

1. Escriu una consulta per trobar el restaurant\_id, name, i grades per a aquells restaurants que aconsegueixin un grau "A" i un score 11 en dades d'estudi ISODate "2014-08-11T00:00:00Z".

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

1. Escriu una consulta per trobar el restaurant\_id, name i grades per a aquells restaurants on el 2n element de varietat de graus conté un grau de "A" i marcador 9 sobre un ISODate "2014-08-11T00:00:00Z".

db.restaurants.find({ "grades.1.score" : 9, "grades.1.date" : ISODate("2014-08-11T00:00:00Z"), "grades.1.grade" : "A"}, { name : true, restaurant\_id : true, grades : true })

1. Escriu una consulta per trobar el restaurant\_id, name, adreça i ubicació geogràfica per a aquells restaurants on el segon element del array coord conté un valor que és més de 42 i fins a 52.

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

[$lte](https://www.mongodb.com/docs/manual/reference/operator/query/lte/#mongodb-query-op.-lte)selecciona los documentos donde el valor de fieldes menor o igual a (es decir <=) el especificado value.

1. Escriu una consulta per organitzar el nom dels restaurants en ordre ascendent juntament amb totes les columnes.

db.restaurants.find({}).sort({ name : 1 })

1. Escriu una consulta per organitzar el nom dels restaurants en ordre descendent juntament amb totes les columnes.

db.restaurants.find({}).sort({ name : -1 })

1. Escriu una consulta per organitzar el nom de la cuisine en ordre ascendent i pel mateix barri de cuisine. Ordre descendent.

db.restaurants.find({}).sort({ cuisine : 1, borough : -1 })

1. Escriu una consulta per saber totes les direccions que no contenen el carrer.

db.restaurants.find({}, { "address.street" : null, name : true })

1. Escriu una consulta que seleccionarà tots els documents en la col·lecció de restaurants on el valor del camp coord és Double.

db.restaurants.find({}, { "address.coord" : { $type : 1 }, name : true })

1. Escriu una consulta que seleccionarà el restaurant\_id, name i grade per a aquells restaurants que retornin 0 com a resta després de dividir el marcador per 7.

db.restaurants.find({ "grades.score" : { $mod : [7, 0] }}, { "grades.grade" : true, name : true, restaurant\_id : true })

los[$mod](https://www.mongodb.com/docs/manual/reference/operator/query/mod/#mongodb-query-op.-mod)El operador devuelve un error si la [ divisor, remainder ]matriz contiene menos o más de dos elementos.

1. Escriu una consulta per trobar el name de restaurant, borough, longitud i altitud i cuisine per a aquells restaurants que contenen 'mon' com tres lletres en algun lloc del seu nom.

db.restaurants.find({ name : /mon/ }, { name : true, restaurant\_id : true, borough : true, cuisine : true, "address.coord" : true })

1. Escriu una consulta per trobar el name de restaurant, borough, longitud i latitud i cuisine per a aquells restaurants que contenen 'Mad' com primeres tres lletres del seu nom.

db.restaurants.find({name : /^Mad/}, {name : true, restaurant\_id : true, borough : true, cuisine : true, "address.coord" : true})

los[$elemMatch](https://www.mongodb.com/docs/manual/reference/operator/query/elemMatch/#mongodb-query-op.-elemMatch)El operador coincide con documentos que contienen un campo de matriz con al menos un elemento que coincide con todos los criterios de consulta especificados.

filtro LIKE de ciudades que empiecen por M.

A la hora de utilizar las expresiones regulares, los índices solo se utilizarán de forma eficiente cuando utilizemos la forma /^cadena/

Otros podrían ser ciudades que contengan una «r»:

cursor = db.ciudades.find({ciudad:/r/});

O ciudades que acaben en d:

cursor = db.ciudades.find({ciudad:/d$/});

Podemos realizar las siguientes ordenaciones en [MongoDB](http://www.manualweb.net/tutorial-mongodb/" \o "Manual MongoDB). Si queremos ordenar las ciudades de forma ascendente escribiremos:

cursor = db.ciudades.find().sort({ciudad:1});

Por el contrario, si queremos ordenar las ciudades de forma descendente:

cursor = db.ciudades.find().sort({ciudad:-1});

los[$or](https://www.mongodb.com/docs/manual/reference/operator/query/or/#mongodb-query-op.-or)El operador realiza una ORoperación lógica en una matriz de uno o más <expressions> y selecciona los documentos que cumplen al menos uno de los <expressions>.

los[$type](https://www.mongodb.com/docs/manual/reference/operator/query/type/#mongodb-query-op.-type)El operador acepta alias de cadena para los tipos BSON además de los números correspondientes a los tipos BSON.

| **Type** | **Number** | **Alias** | **Notes** |
| --- | --- | --- | --- |
| Double | 1 | "double" |  |
| String | 2 | "string" |  |
| Object | 3 | "object" |  |
| Array | 4 | "array" |  |
| Binary data | 5 | "binData" |  |
| Undefined | 6 | "undefined" | Deprecated. |
| ObjectId | 7 | "objectId" |  |
| Boolean | 8 | "bool" |  |
| Date | 9 | "date" |  |
| Null | 10 | "null" |  |
| Regular Expression | 11 | "regex" |  |
| DBPointer | 12 | "dbPointer" | Deprecated. |
| JavaScript | 13 | "javascript" |  |
| Symbol | 14 | "symbol" | Deprecated. |
| JavaScript code with scope | 15 | "javascriptWithScope" | Deprecated in MongoDB 4.4. |
| 32-bit integer | 16 | "int" |  |
| Timestamp | 17 | "timestamp" |  |
| 64-bit integer | 18 | "long" |  |
| Decimal128 | 19 | "decimal" |  |
| Min key | -1 | "minKey" |  |
| Max key | 127 | "maxKey" |  |

[$type](https://www.mongodb.com/docs/manual/reference/operator/query/type/#mongodb-query-op.-type) supports the number alias, which will match against the following [BSON](https://www.mongodb.com/docs/manual/reference/glossary/#std-term-BSON) types:

[double](https://www.mongodb.com/docs/manual/reference/mongodb-extended-json/#mongodb-bsontype-Double)

[32-bit integer](https://www.mongodb.com/docs/manual/reference/mongodb-extended-json/#mongodb-bsontype-Int32)

[64-bit integer](https://www.mongodb.com/docs/manual/reference/mongodb-extended-json/#mongodb-bsontype-Int64)

[decimal](https://www.mongodb.com/docs/manual/reference/mongodb-extended-json/#mongodb-bsontype-Decimal128)