**MongoDB – Complex Queries**

**Mongo DB Exercises - With the Restaurants Data Set**

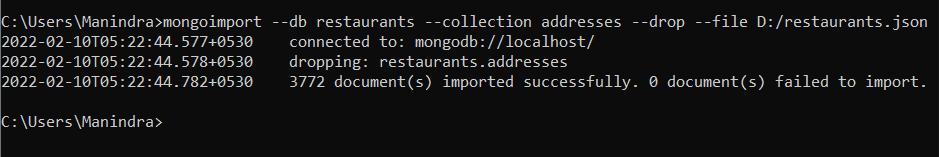
1. Download the restaurants.zip file
2. Unzip the file, you will see restaurants.json file
3. Run the mongod server
4. Run the following command to import the json file provided. It will load the json file into the mongodb with database name - restaurants, collections name - addresses **mongoimport --db restaurants --collection addresses --file restaurants.json**

C:\Users\Manindra>mongoimport --db restaurants --collection addresses --drop --file D:/restaurants.json

2022-02-10T05:22:44.577+0530 connected to: mongodb://localhost/

2022-02-10T05:22:44.578+0530 dropping: restaurants.addresses

2022-02-10T05:22:44.782+0530 3772 document(s) imported successfully. 0 document(s) failed to import.



1. Run mongo shell command
2. show databases
3. use restaurants

C:\Users\Manindra>mongo

> show dbs

admin 0.000GB

config 0.000GB

local 0.000GB

mongo\_practice 0.000GB

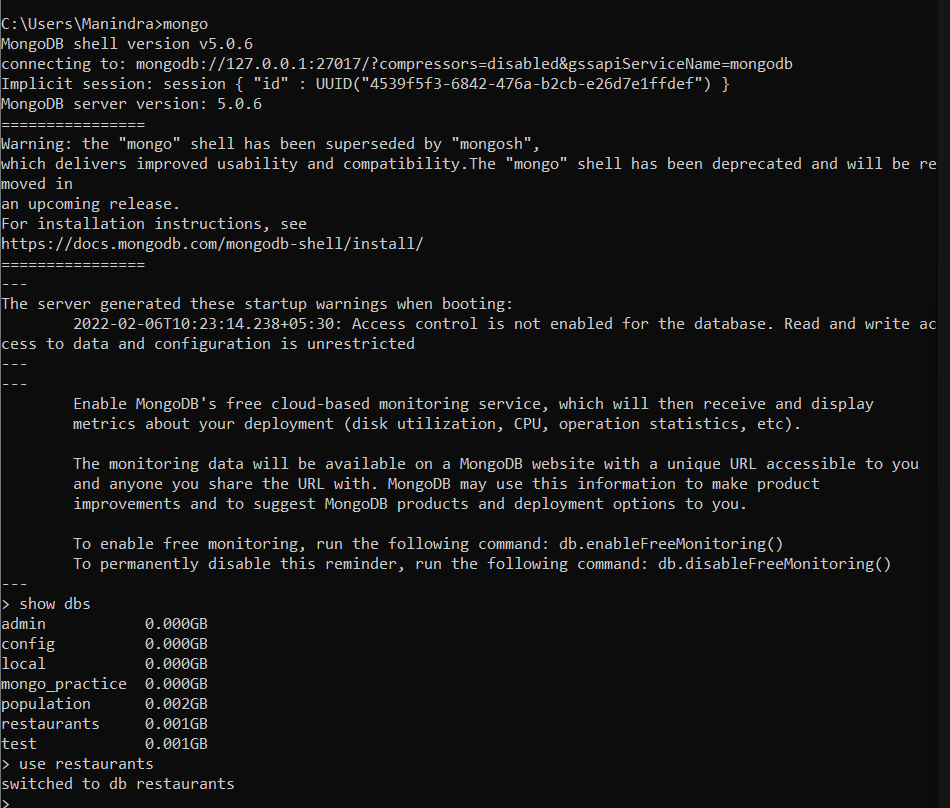
population 0.002GB

restaurants 0.001GB

test 0.001GB

> use restaurants

switched to db restaurants



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

>db.addresses.find()

{ "\_id" : ObjectId("620453cca79d51c831374d67"), "address" : { "building" : "469", "coord" : [ -73.961704, 40.662942 ], "street" : "Flatbush Avenue", "zipcode" : "11225" }, "borough" : "Brooklyn", "cuisine" : "Hamburgers", "grades" : [ { "date" : ISODate("2014-12-30T00:00:00Z"), "grade" : "A", "score" : 8 }, { "date" : ISODate("2014-07-01T00:00:00Z"), "grade" : "B", "score" : 23 }, { "date" : ISODate("2013-04-30T00:00:00Z"), "grade" : "A", "score" : 12 }, { "date" : ISODate("2012-05-08T00:00:00Z"), "grade" : "A", "score" : 12 } ], "name" : "Wendy'S", "restaurant\_id" : "30112340" }



1. 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**

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

> db.addresses.find().pretty()

{

"\_id" : ObjectId("620453cca79d51c831374d67"),

"address" : {

"building" : "469",

"coord" : [

-73.961704,

40.662942

],

"street" : "Flatbush Avenue",

"zipcode" : "11225"

},

"borough" : "Brooklyn",

"cuisine" : "Hamburgers",

"grades" : [

{

"date" : ISODate("2014-12-30T00:00:00Z"),

"grade" : "A",

"score" : 8

},

{

"date" : ISODate("2014-07-01T00:00:00Z"),

"grade" : "B",

"score" : 23

},

{

"date" : ISODate("2013-04-30T00:00:00Z"),

"grade" : "A",

"score" : 12

},

{

"date" : ISODate("2012-05-08T00:00:00Z"),

"grade" : "A",

"score" : 12

}

],

"name" : "Wendy'S",

"restaurant\_id" : "30112340"

}



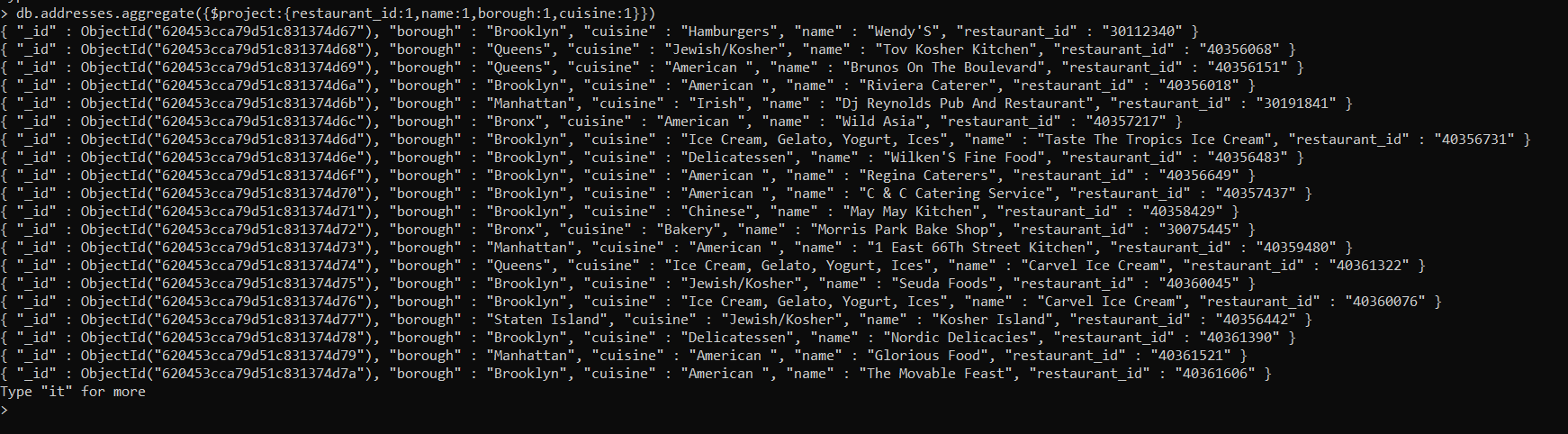
1. Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.

> db.addresses.aggregate({$project:{restaurant\_id:1,name:1,borough:1,cuisine:1}})

{ "\_id" : ObjectId("620453cca79d51c831374d67"), "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant\_id" : "30112340" }

{ "\_id" : ObjectId("620453cca79d51c831374d68"), "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "restaurant\_id" : "40356068" }

{ "\_id" : ObjectId("620453cca79d51c831374d69"), "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard", "restaurant\_id" : "40356151" }



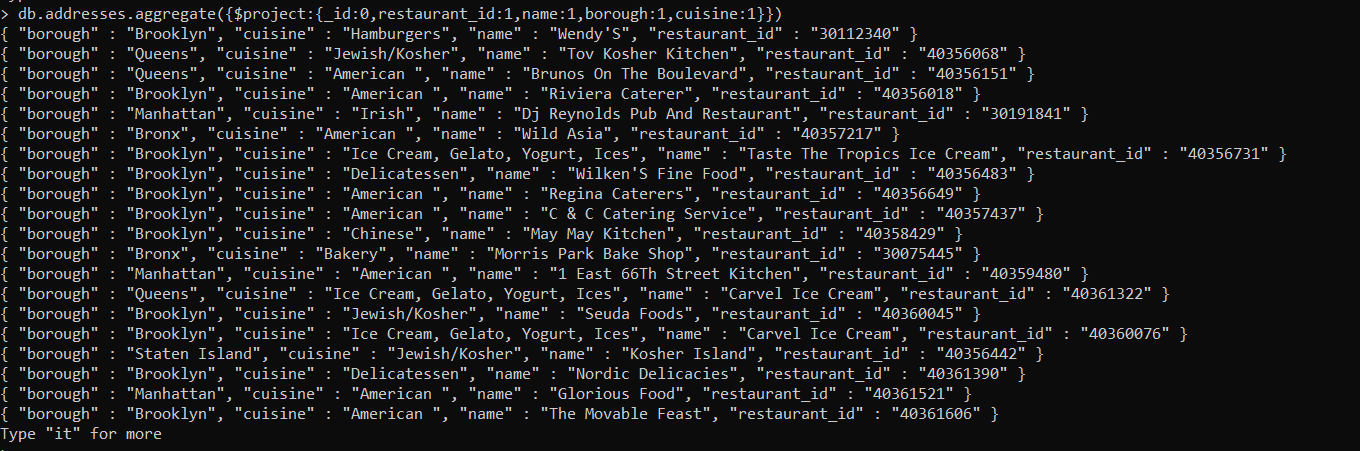
1. Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.

>db.addresses.aggregate({$project:{\_id:0,restaurant\_id:1,name:1,borough:1,cuisine:1}})

{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant\_id" : "30112340" }

{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "restaurant\_id" : "40356068" }

{ "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard", "restaurant\_id" : "40356151" }



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.

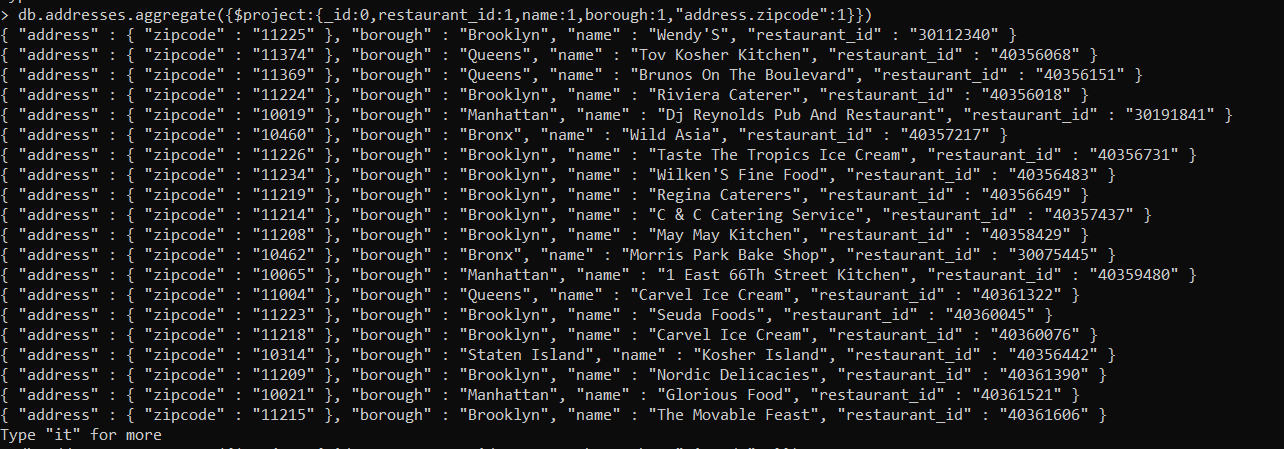
> db.addresses.aggregate({$project:{\_id:0,restaurant\_id:1,name:1,borough:1,"address.zipcode":1}})

{ "address" : { "zipcode" : "11225" }, "borough" : "Brooklyn", "name" : "Wendy'S", "restaurant\_id" : "30112340" }

{ "address" : { "zipcode" : "11374" }, "borough" : "Queens", "name" : "Tov Kosher Kitchen", "restaurant\_id" : "40356068" }

{ "address" : { "zipcode" : "11369" }, "borough" : "Queens", "name" : "Brunos On The Boulevard", "restaurant\_id" : "40356151" }

{ "address" : { "zipcode" : "11224" }, "borough" : "Brooklyn", "name" : "Riviera Caterer", "restaurant\_id" : "40356018" }



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

> db.addresses.aggregate([{$match:{borough:"Bronx"}},{$project:{name:1,\_id:0,borough:1}},{$sort:{name:1}},{$limit:5}])

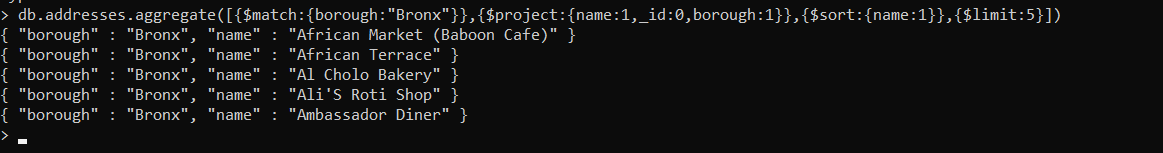
{ "borough" : "Bronx", "name" : "African Market (Baboon Cafe)" }

{ "borough" : "Bronx", "name" : "African Terrace" }

{ "borough" : "Bronx", "name" : "Al Cholo Bakery" }

{ "borough" : "Bronx", "name" : "Ali'S Roti Shop" }

{ "borough" : "Bronx", "name" : "Ambassador Diner" }



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

> db.addresses.aggregate([{$match:{borough:"Bronx"}},{$project:{name:1,\_id:0,borough:1}},{$sort:{name:1}}])

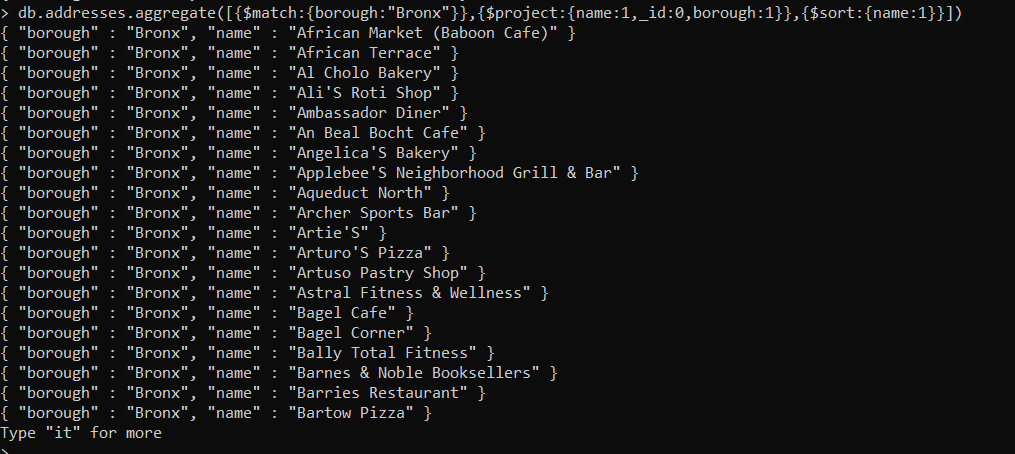
{ "borough" : "Bronx", "name" : "African Market (Baboon Cafe)" }

{ "borough" : "Bronx", "name" : "African Terrace" }

{ "borough" : "Bronx", "name" : "Al Cholo Bakery" }

{ "borough" : "Bronx", "name" : "Ali'S Roti Shop" }

{ "borough" : "Bronx", "name" : "Ambassador Diner" }



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

>db.addresses.aggregate([{$match:{borough:"Bronx"}},{$project:{name:1,\_id:0,borough:1}},{$sort:{name:1}},{$skip:5},{$limit:5}])

{ "borough" : "Bronx", "name" : "An Beal Bocht Cafe" }

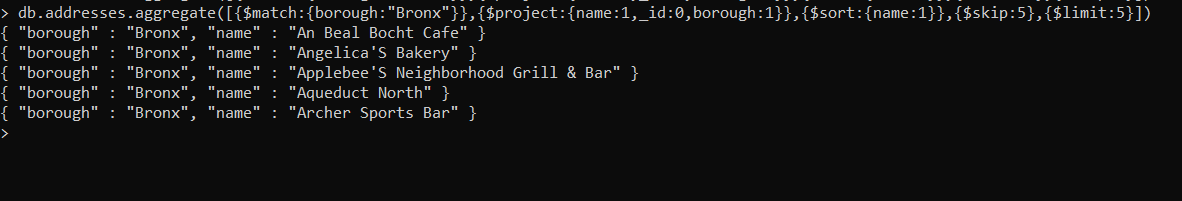
{ "borough" : "Bronx", "name" : "Angelica'S Bakery" }

{ "borough" : "Bronx", "name" : "Applebee'S Neighborhood Grill & Bar" }

{ "borough" : "Bronx", "name" : "Aqueduct North" }

{ "borough" : "Bronx", "name" : "Archer Sports Bar" }

>



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

> db.addresses.aggregate([{$unwind:"$grades"},{$group:{\_id:"$name",totalScore:{$sum:"$grades.score"}}},{$match:{"totalScore":{$gt:90}}}])

{ "\_id" : "Fresco", "totalScore" : 97 }

{ "\_id" : "Grand Hyatt New York Hotel", "totalScore" : 96 }

{ "\_id" : "Pho Bang Restaurant", "totalScore" : 167 }

{ "\_id" : "Cilantro", "totalScore" : 125 }

{ "\_id" : "North Shore Diner", "totalScore" : 98 }

{ "\_id" : "Triple Crown Diner", "totalScore" : 95 }

{ "\_id" : "Arturo'S", "totalScore" : 136 }

{ "\_id" : "Market Restaurant (C-D Block)", "totalScore" : 105 }

{ "\_id" : "Antojitos Mexicanos", "totalScore" : 93 }

{ "\_id" : "Grini'S Grill & Restaurant", "totalScore" : 103 }

{ "\_id" : "Moonstruck East", "totalScore" : 121 }

{ "\_id" : "Cha Cha'S Backyard Garden Bar & Cafe", "totalScore" : 154 }

{ "\_id" : "New Roma Pizza", "totalScore" : 122 }

{ "\_id" : "Rosa'S Pizza", "totalScore" : 127 }

{ "\_id" : "Park Plaza Restaurant", "totalScore" : 120 }

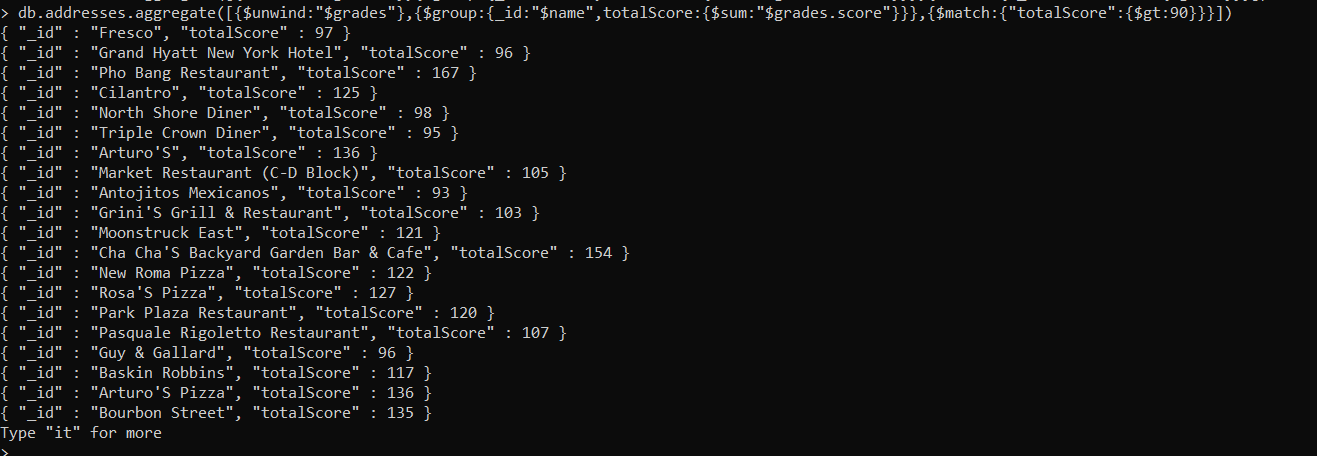
{ "\_id" : "Pasquale Rigoletto Restaurant", "totalScore" : 107 }

{ "\_id" : "Guy & Gallard", "totalScore" : 96 }

{ "\_id" : "Baskin Robbins", "totalScore" : 117 }

{ "\_id" : "Arturo'S Pizza", "totalScore" : 136 }

{ "\_id" : "Bourbon Street", "totalScore" : 135 }



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

> db.addresses.aggregate([{$unwind:"$grades"},{$group:{\_id:"$name",totalScore:{$sum:"$grades.score"}}},{$match:{"totalScore":{$gt:80,$lt:100}}}])

{ "\_id" : "Country Donuts", "totalScore" : 89 }

{ "\_id" : "Los Arrieros Restaurant", "totalScore" : 88 }

{ "\_id" : "New Leaf Cafe", "totalScore" : 98 }

{ "\_id" : "Cebu", "totalScore" : 84 }

{ "\_id" : "Abir Halal Restaurant", "totalScore" : 82 }

{ "\_id" : "Sunrise Coffee Shop", "totalScore" : 85 }

{ "\_id" : "Glatt Kosher Family Chinese Restaurant", "totalScore" : 91 }

{ "\_id" : "Ariyoshi Japanese Restaurant", "totalScore" : 97 }

{ "\_id" : "Johns Cafe & Resturant", "totalScore" : 92 }

{ "\_id" : "Jimmy Ryan'S", "totalScore" : 89 }

{ "\_id" : "Pizza Park", "totalScore" : 94 }

{ "\_id" : "Speedy Deli", "totalScore" : 91 }

{ "\_id" : "The Ranch Restaurant & Bar Of Guyana", "totalScore" : 94 }

{ "\_id" : "Schick'S Manor", "totalScore" : 96 }

{ "\_id" : "Yankee Clipper", "totalScore" : 82 }

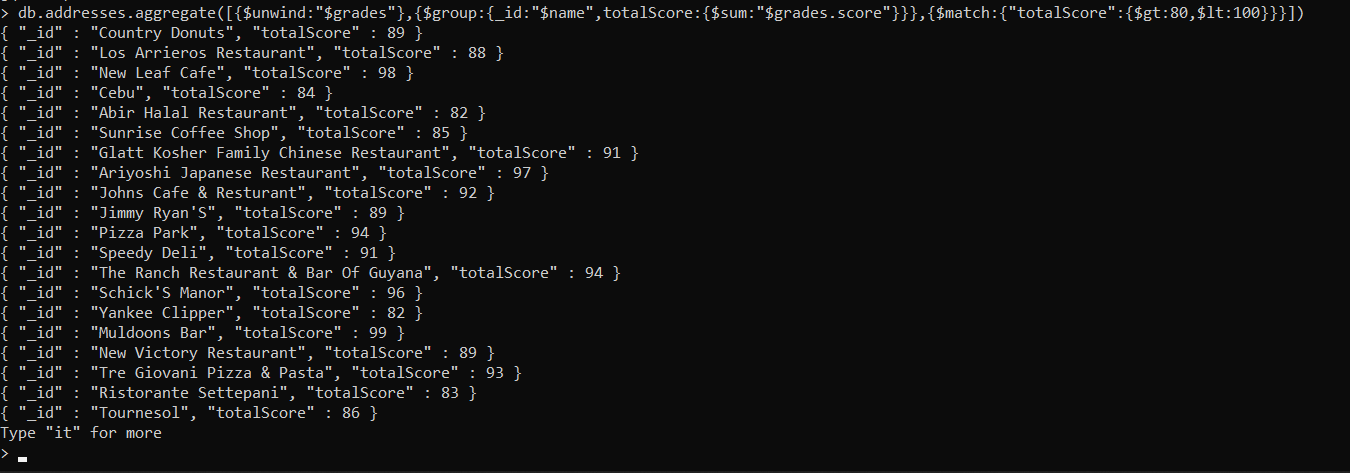
{ "\_id" : "Muldoons Bar", "totalScore" : 99 }

{ "\_id" : "New Victory Restaurant", "totalScore" : 89 }

{ "\_id" : "Tre Giovani Pizza & Pasta", "totalScore" : 93 }

{ "\_id" : "Ristorante Settepani", "totalScore" : 83 }

{ "\_id" : "Tournesol", "totalScore" : 86 }



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

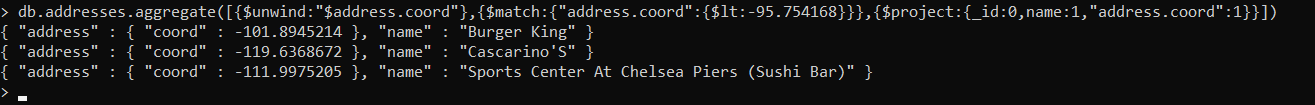
> db.addresses.aggregate([{$unwind:"$address.coord"},{$match:{"address.coord":{$lt:-95.754168}}},{$project:{\_id:0,name:1,"address.coord":1}}])

{ "address" : { "coord" : -101.8945214 }, "name" : "Burger King" }

{ "address" : { "coord" : -119.6368672 }, "name" : "Cascarino'S" }

{ "address" : { "coord" : -111.9975205 }, "name" : "Sports Center At Chelsea Piers (Sushi Bar)" }

>



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.addresses.aggregate([ {$unwind:"$grades"}, {$match:{cuisine:{$ne:"American "}}}, {$match:{"address.coord.0":{$lt:-65.754168}}}, {$group:{\_id:{name:"$name",cuisine:"$cuisine",latitude:"$address.coord",totalScore:{$sum:"$grades.score"}}}},{$match:{"\_id.totalScore":{$gt:70}}}, {$project:{"\_id.name":1,"\_id.cuisine":1,"\_id.totalScore":1,"\_id.latitude":1}} ])

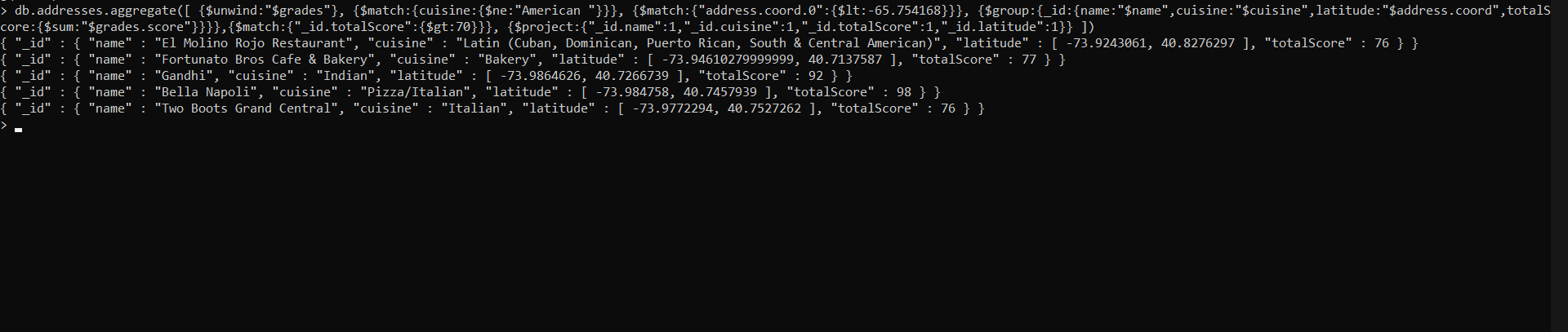
{ "\_id" : { "name" : "El Molino Rojo Restaurant", "cuisine" : "Latin (Cuban, Dominican, Puerto Rican, South & Central American)", "latitude" : [ -73.9243061, 40.8276297 ], "totalScore" : 76 } }

{ "\_id" : { "name" : "Fortunato Bros Cafe & Bakery", "cuisine" : "Bakery", "latitude" : [ -73.94610279999999, 40.7137587 ], "totalScore" : 77 } }

{ "\_id" : { "name" : "Gandhi", "cuisine" : "Indian", "latitude" : [ -73.9864626, 40.7266739 ], "totalScore" : 92 } }

{ "\_id" : { "name" : "Bella Napoli", "cuisine" : "Pizza/Italian", "latitude" : [ -73.984758, 40.7457939 ], "totalScore" : 98 } }

{ "\_id" : { "name" : "Two Boots Grand Central", "cuisine" : "Italian", "latitude" : [ -73.9772294, 40.7527262 ], "totalScore" : 76 } }



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.

> db.addresses.aggregate([ {$unwind:"$grades"}, {$match:{cuisine:{$ne:"American "}}}, {$match:{"address.coord.1":{$lt:65.754168}}}, {$group:{\_id:{name:"$name",cuisine:"$cuisine",latitude:"$address.coord",totalScore:{$sum:"$grades.score"}}}},{$match:{"\_id.totalScore":{$gt:70}}}, {$project:{"\_id.name":1,"\_id.cuisine":1,"\_id.totalScore":1,"\_id.latitude":1}} ])

{ "\_id" : { "name" : "Two Boots Grand Central", "cuisine" : "Italian", "latitude" : [ -73.9772294, 40.7527262 ], "totalScore" : 76 } }

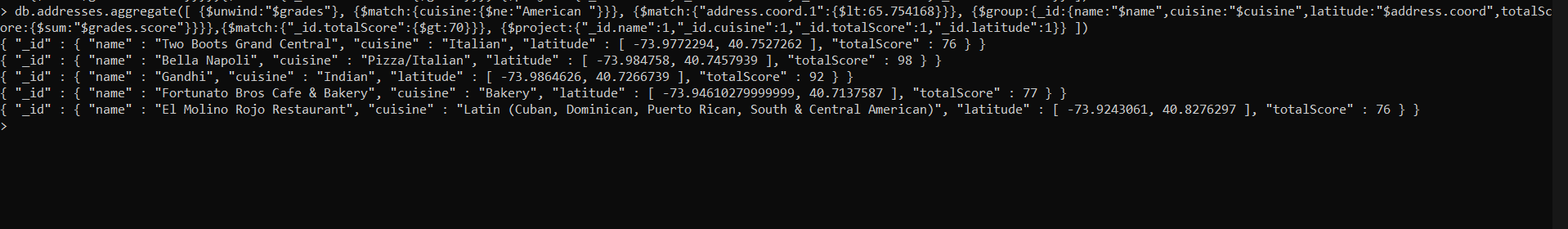
{ "\_id" : { "name" : "Bella Napoli", "cuisine" : "Pizza/Italian", "latitude" : [ -73.984758, 40.7457939 ], "totalScore" : 98 } }

{ "\_id" : { "name" : "Gandhi", "cuisine" : "Indian", "latitude" : [ -73.9864626, 40.7266739 ], "totalScore" : 92 } }

{ "\_id" : { "name" : "Fortunato Bros Cafe & Bakery", "cuisine" : "Bakery", "latitude" : [ -73.94610279999999, 40.7137587 ], "totalScore" : 77 } }

{ "\_id" : { "name" : "El Molino Rojo Restaurant", "cuisine" : "Latin (Cuban, Dominican, Puerto Rican, South & Central American)", "latitude" : [ -73.9243061, 40.8276297 ], "totalScore" : 76 } }

>



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.addresses.aggregate([ {$unwind:"$grades"}, {$match:{cuisine:{$ne:"American "}}}, {$match:{"grades.grade":"A"}}, {$match:{borough:{$ne:"Brooklyn"}}}, {$sort:{cuisine:-1}},{$project:{name:1,cuisine:1,\_id:0,"grades.grade":1,borough:1}} ])

{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Thai Son" }

{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Thai Son" }

{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Thai Son" }

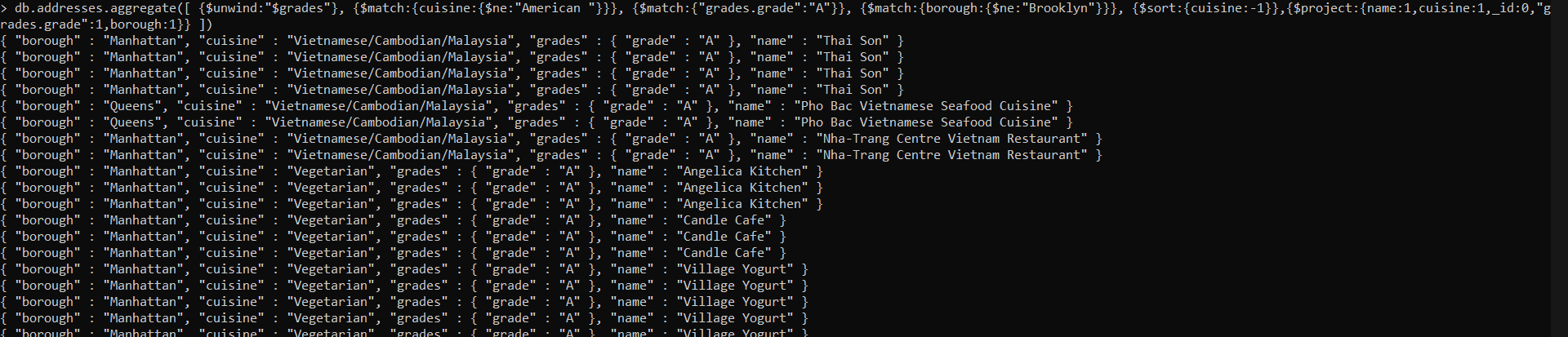
{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Thai Son" }

{ "borough" : "Queens", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Pho Bac Vietnamese Seafood Cuisine" }

{ "borough" : "Queens", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Pho Bac Vietnamese Seafood Cuisine" }

{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Nha-Trang Centre Vietnam Restaurant" }

{ "borough" : "Manhattan", "cuisine" : "Vietnamese/Cambodian/Malaysia", "grades" : { "grade" : "A" }, "name" : "Nha-Trang Centre Vietnam Restaurant" }



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.

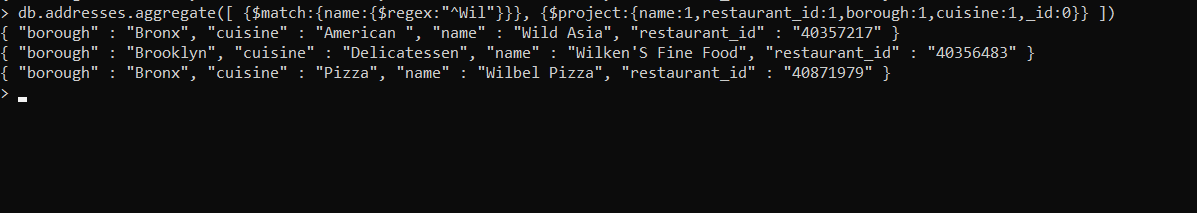
> db.addresses.aggregate([ {$match:{name:{$regex:"^Wil"}}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Wild Asia", "restaurant\_id" : "40357217" }

{ "borough" : "Brooklyn", "cuisine" : "Delicatessen", "name" : "Wilken'S Fine Food", "restaurant\_id" : "40356483" }

{ "borough" : "Bronx", "cuisine" : "Pizza", "name" : "Wilbel Pizza", "restaurant\_id" : "40871979" }

>



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.

> db.addresses.aggregate([ {$match:{name:{$regex:"ces$"}}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Pieces", "restaurant\_id" : "40399910" }

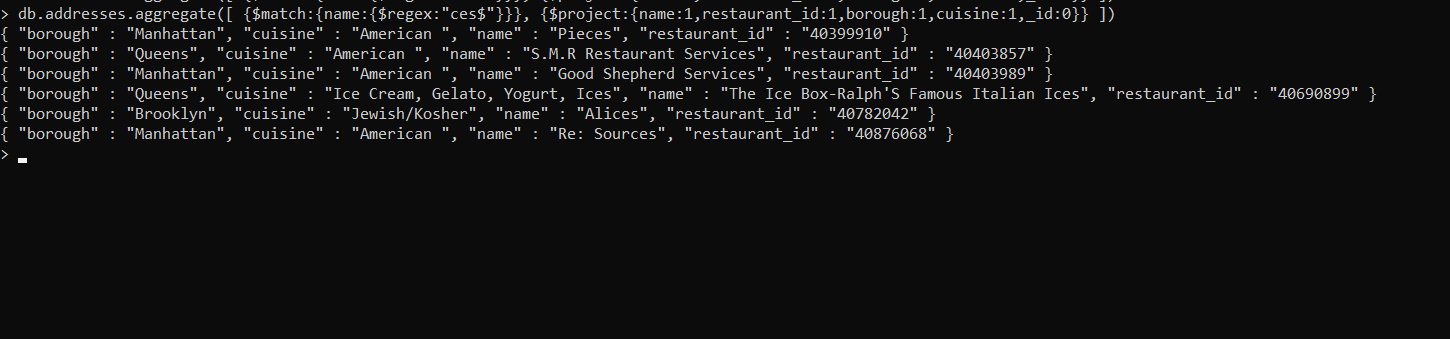
{ "borough" : "Queens", "cuisine" : "American ", "name" : "S.M.R Restaurant Services", "restaurant\_id" : "40403857" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Good Shepherd Services", "restaurant\_id" : "40403989" }

{ "borough" : "Queens", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "The Ice Box-Ralph'S Famous Italian Ices", "restaurant\_id" : "40690899" }

{ "borough" : "Brooklyn", "cuisine" : "Jewish/Kosher", "name" : "Alices", "restaurant\_id" : "40782042" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Re: Sources", "restaurant\_id" : "40876068" }



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.

> db.addresses.aggregate([ {$match:{name:/Reg/g}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers", "restaurant\_id" : "40356649" }

{ "borough" : "Manhattan", "cuisine" : "Café/Coffee/Tea", "name" : "Caffe Reggio", "restaurant\_id" : "40369418" }

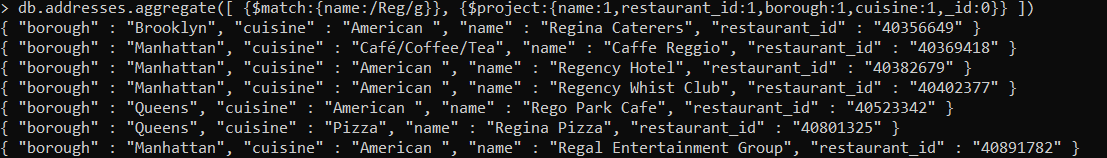
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regency Hotel", "restaurant\_id" : "40382679" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regency Whist Club", "restaurant\_id" : "40402377" }

{ "borough" : "Queens", "cuisine" : "American ", "name" : "Rego Park Cafe", "restaurant\_id" : "40523342" }

{ "borough" : "Queens", "cuisine" : "Pizza", "name" : "Regina Pizza", "restaurant\_id" : "40801325" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regal Entertainment Group", "restaurant\_id" : "40891782" }



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

> db.addresses.aggregate([ {$match:{borough:"Bronx"}}, {$match:{$or:[{cuisine:"American "},{cuisine:"Chinese"}]}},{$project:{name:1,\_id:0,cuisine:1,borough:1}} ])

{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Wild Asia" }

{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Happy Garden" }

{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Happy Garden" }

{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Manhem Club" }

{ "borough" : "Bronx", "cuisine" : "American ", "name" : "The New Starling Athletic Club Of The Bronx" }



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.

> db.addresses.aggregate([ {$match:{$or:[{borough:"Bronx"},{borough:"Brooklyn"},{borough:"Queens"},{borough:"Staten Island"}]}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant\_id" : "30112340" }

{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "restaurant\_id" : "40356068" }

{ "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard", "restaurant\_id" : "40356151" }

{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Riviera Caterer", "restaurant\_id" : "40356018" }

{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Wild Asia", "restaurant\_id" : "40357217" }

{ "borough" : "Brooklyn", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Taste The Tropics Ice Cream", "restaurant\_id" : "40356731" }

{ "borough" : "Brooklyn", "cuisine" : "Delicatessen", "name" : "Wilken'S Fine Food", "restaurant\_id" : "40356483" }

{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers", "restaurant\_id" : "40356649" }

{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "C & C Catering Service", "restaurant\_id" : "40357437" }

{ "borough" : "Brooklyn", "cuisine" : "Chinese", "name" : "May May Kitchen", "restaurant\_id" : "40358429" }

{ "borough" : "Bronx", "cuisine" : "Bakery", "name" : "Morris Park Bake Shop", "restaurant\_id" : "30075445" }



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.addresses.aggregate([ {$match:{borough:{$nin:["Bronx","Brooklyn","Queens","Staten Island"]}}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

{ "borough" : "Manhattan", "cuisine" : "Irish", "name" : "Dj Reynolds Pub And Restaurant", "restaurant\_id" : "30191841" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "1 East 66Th Street Kitchen", "restaurant\_id" : "40359480" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Glorious Food", "restaurant\_id" : "40361521" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "P & S Deli Grocery", "restaurant\_id" : "40362264" }

{ "borough" : "Manhattan", "cuisine" : "Chicken", "name" : "Harriet'S Kitchen", "restaurant\_id" : "40362098" }

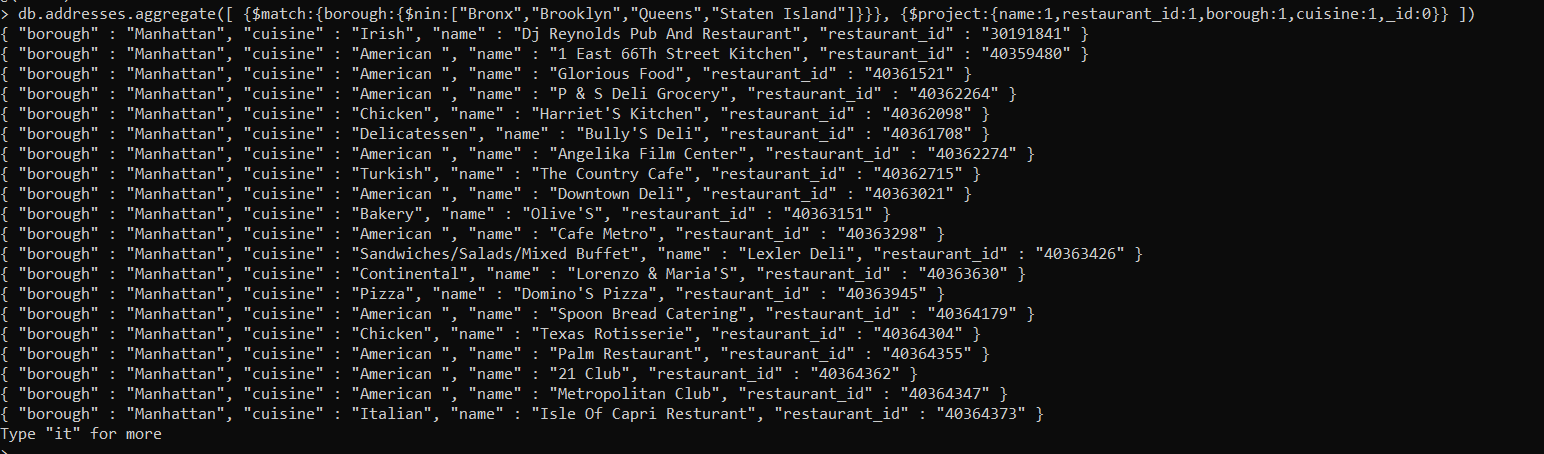
{ "borough" : "Manhattan", "cuisine" : "Delicatessen", "name" : "Bully'S Deli", "restaurant\_id" : "40361708" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Angelika Film Center", "restaurant\_id" : "40362274" }

{ "borough" : "Manhattan", "cuisine" : "Turkish", "name" : "The Country Cafe", "restaurant\_id" : "40362715" }

{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Downtown Deli", "restaurant\_id" : "40363021" }

{ "borough" : "Manhattan", "cuisine" : "Bakery", "name" : "Olive'S", "restaurant\_id" : "40363151" }



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.

> db.addresses.aggregate([ {$group:{\_id:{id:"$restaurant\_id",name:"$name",borough:"$borough",cuisine:"$cuisine",totalScore:{$sum:"$grades.score"}}}},{$match:{"\_id.totalScore":{$lte:10}}}, {$project:{"\_id.totalScore":1,"\_id.name":1,"\_id.id":1,"\_id.borough":1,"\_id.cuisine":1}} ])

{ "\_id" : { "id" : "40729118", "name" : "Seven Stars Bakery", "borough" : "Brooklyn", "cuisine" : "Bakery", "totalScore" : 8 } }

{ "\_id" : { "id" : "40385580", "name" : "Lattanzi", "borough" : "Manhattan", "cuisine" : "Italian", "totalScore" : 10 } }

{ "\_id" : { "id" : "40883581", "name" : "Sheas Emerald", "borough" : "Bronx", "cuisine" : "Irish", "totalScore" : 9 } }

{ "\_id" : { "id" : "40403946", "name" : "Ambassador Diner", "borough" : "Bronx", "cuisine" : "American ", "totalScore" : 4 } }

{ "\_id" : { "id" : "40483729", "name" : "Cafe Madison", "borough" : "Manhattan", "cuisine" : "Café/Coffee/Tea", "totalScore" : 8 } }

{ "\_id" : { "id" : "40839236", "name" : "Concession # 120/ The Bullpen", "borough" : "Brooklyn", "cuisine" : "American ", "totalScore" : 9 } }

{ "\_id" : { "id" : "40761262", "name" : "Circle In The Square Theatre", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 2 } }

{ "\_id" : { "id" : "40719738", "name" : "Juniors", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 7 } }

{ "\_id" : { "id" : "40610155", "name" : "Citibank Executive Conference Center", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 5 } }

{ "\_id" : { "id" : "40524878", "name" : "Mcdonald'S", "borough" : "Brooklyn", "cuisine" : "Hamburgers", "totalScore" : 7 } }

{ "\_id" : { "id" : "40839319", "name" : "Stand # 210", "borough" : "Brooklyn", "cuisine" : "American ", "totalScore" : 3 } }

{ "\_id" : { "id" : "40393912", "name" : "Julliard & S.A.B. Cafeteria", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 8 } }

{ "\_id" : { "id" : "40592920", "name" : "Forest Park Golf Course", "borough" : "Queens", "cuisine" : "American ", "totalScore" : 7 } }

{ "\_id" : { "id" : "40397486", "name" : "Credit Agricole-Cbi Executive Room", "borough" : "Manhattan", "cuisine" : "French", "totalScore" : 9 } }

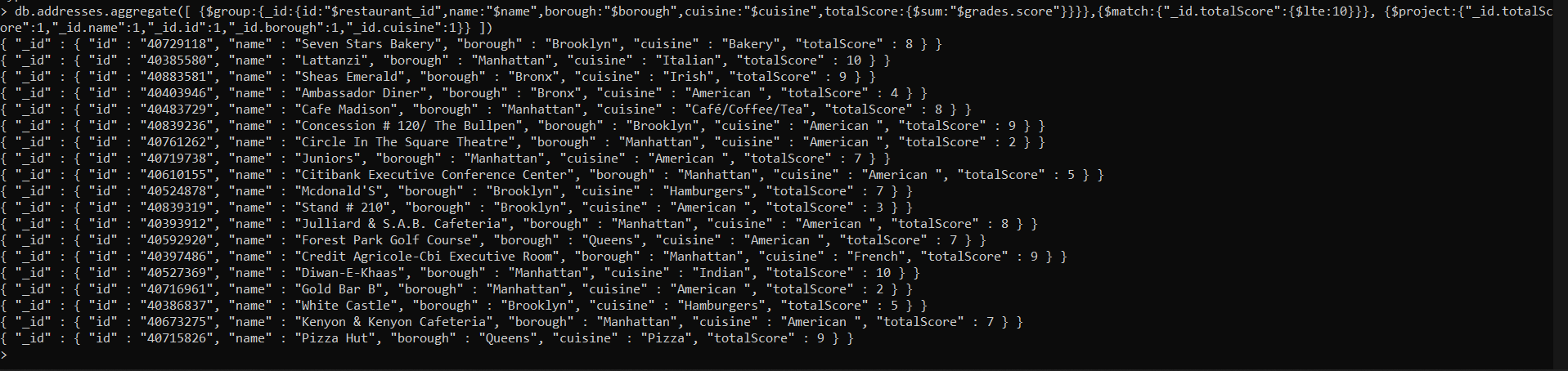
{ "\_id" : { "id" : "40527369", "name" : "Diwan-E-Khaas", "borough" : "Manhattan", "cuisine" : "Indian", "totalScore" : 10 } }

{ "\_id" : { "id" : "40716961", "name" : "Gold Bar B", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 2 } }

{ "\_id" : { "id" : "40386837", "name" : "White Castle", "borough" : "Brooklyn", "cuisine" : "Hamburgers", "totalScore" : 5 } }

{ "\_id" : { "id" : "40673275", "name" : "Kenyon & Kenyon Cafeteria", "borough" : "Manhattan", "cuisine" : "American ", "totalScore" : 7 } }

{ "\_id" : { "id" : "40715826", "name" : "Pizza Hut", "borough" : "Queens", "cuisine" : "Pizza", "totalScore" : 9 } }



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.addresses.aggregate([ {$match:{cuisine:{$nin:["American ","Chinese"]},name:{$not:{$regex:"^Wil"}}}}, {$project:{name:1,restaurant\_id:1,borough:1,cuisine:1,\_id:0}} ])

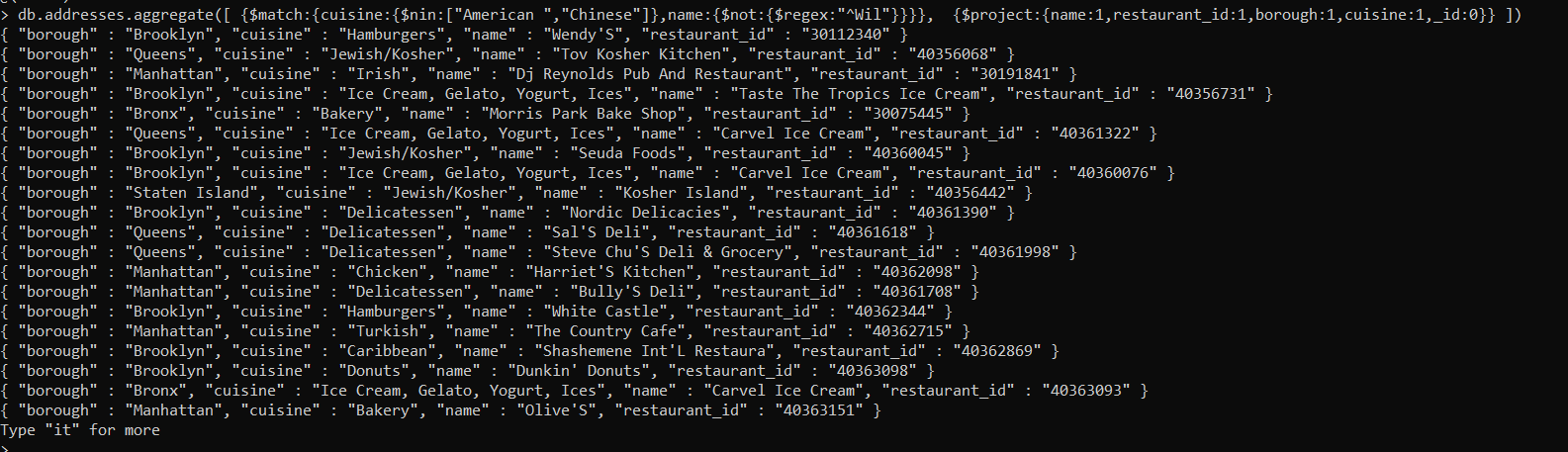
{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant\_id" : "30112340" }

{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "restaurant\_id" : "40356068" }

{ "borough" : "Manhattan", "cuisine" : "Irish", "name" : "Dj Reynolds Pub And Restaurant", "restaurant\_id" : "30191841" }

{ "borough" : "Brooklyn", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Taste The Tropics Ice Cream", "restaurant\_id" : "40356731" }

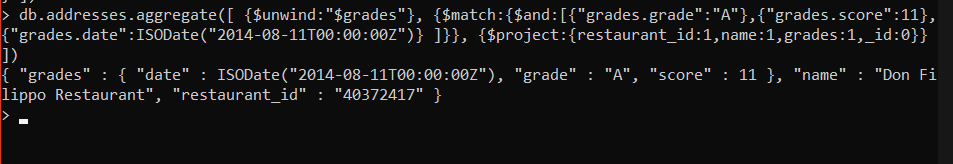
{ "borough" : "Bronx", "cuisine" : "Bakery", "name" : "Morris Park Bake Shop", "restaurant\_id" : "30075445" }



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.addresses.aggregate([ {$unwind:"$grades"}, {$match:{$and:[{"grades.grade":"A"},{"grades.score":11},{"grades.date":ISODate("2014-08-11T00:00:00Z")} ]}}, {$project:{restaurant\_id:1,name:1,grades:1,\_id:0}} ])

{ "grades" : { "date" : ISODate("2014-08-11T00:00:00Z"), "grade" : "A", "score" : 11 }, "name" : "Don Filippo Restaurant", "restaurant\_id" : "40372417" }



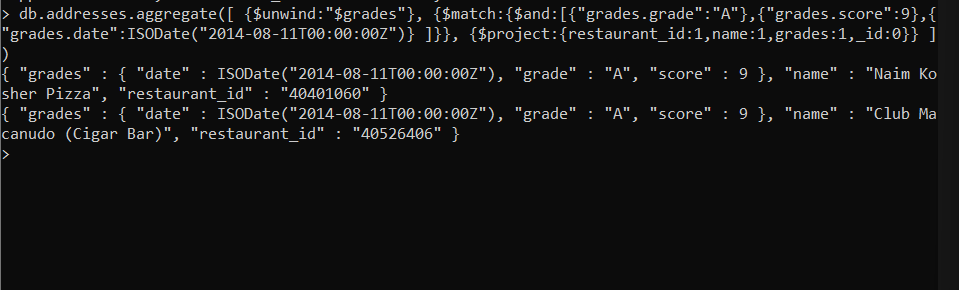
1. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z"

> db.addresses.aggregate([ {$unwind:"$grades"}, {$match:{$and:[{"grades.grade":"A"},{"grades.score":9},{"grades.date":ISODate("2014-08-11T00:00:00Z")} ]}}, {$project:{restaurant\_id:1,name:1,grades:1,\_id:0}} ])

{ "grades" : { "date" : ISODate("2014-08-11T00:00:00Z"), "grade" : "A", "score" : 9 }, "name" : "Naim Kosher Pizza", "restaurant\_id" : "40401060" }

{ "grades" : { "date" : ISODate("2014-08-11T00:00:00Z"), "grade" : "A", "score" : 9 }, "name" : "Club Macanudo (Cigar Bar)", "restaurant\_id" : "40526406" }

>



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.addresses.aggregate([{$match:{"address.coord.1":{"$gt":42,"$lt":52}}}, {$project:{restaurant\_id:1,name:1,"address.coord":1,\_id:0}} ])

{ "address" : { "coord" : [ -78.877224, 42.89546199999999 ] }, "name" : "T.G.I. Friday'S", "restaurant\_id" : "40387990" }

{ "address" : { "coord" : [ -0.7119979, 51.6514664 ] }, "name" : "T.G.I. Fridays", "restaurant\_id" : "40388936" }

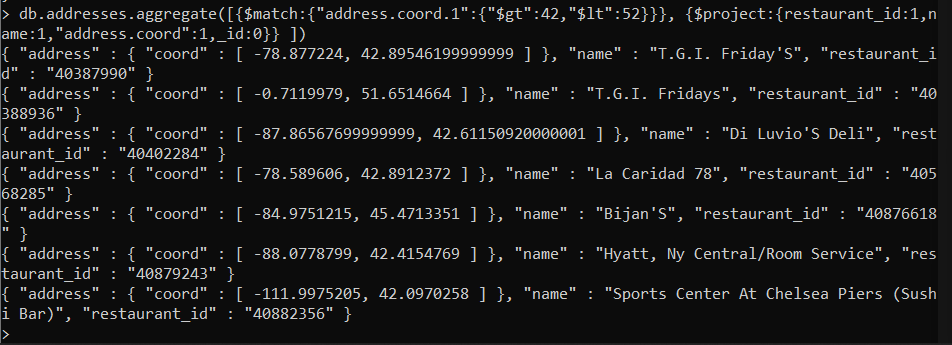
{ "address" : { "coord" : [ -87.86567699999999, 42.61150920000001 ] }, "name" : "Di Luvio'S Deli", "restaurant\_id" : "40402284" }

{ "address" : { "coord" : [ -78.589606, 42.8912372 ] }, "name" : "La Caridad 78", "restaurant\_id" : "40568285" }

{ "address" : { "coord" : [ -84.9751215, 45.4713351 ] }, "name" : "Bijan'S", "restaurant\_id" : "40876618" }

{ "address" : { "coord" : [ -88.0778799, 42.4154769 ] }, "name" : "Hyatt, Ny Central/Room Service", "restaurant\_id" : "40879243" }

{ "address" : { "coord" : [ -111.9975205, 42.0970258 ] }, "name" : "Sports Center At Chelsea Piers (Sushi Bar)", "restaurant\_id" : "40882356" }



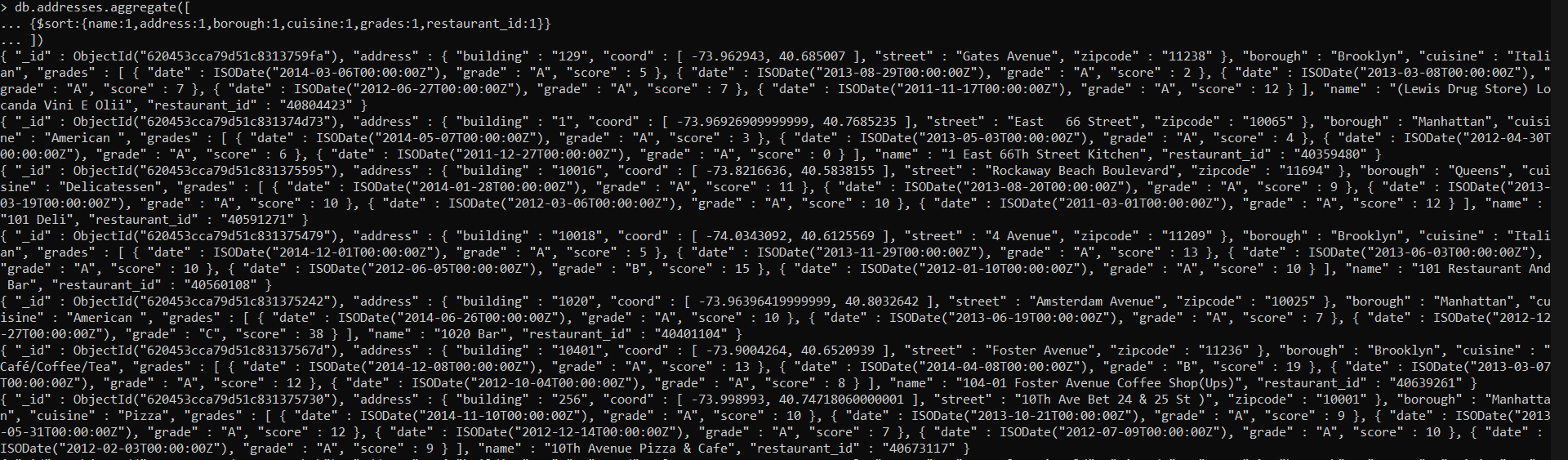
1. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

> db.addresses.aggregate([

... {$sort:{name:1,address:1,borough:1,cuisine:1,grades:1,restaurant\_id:1}}

... ])

{ "\_id" : ObjectId("620453cca79d51c8313759fa"), "address" : { "building" : "129", "coord" : [ -73.962943, 40.685007 ], "street" : "Gates Avenue", "zipcode" : "11238" }, "borough" : "Brooklyn", "cuisine" : "Italian", "grades" : [ { "date" : ISODate("2014-03-06T00:00:00Z"), "grade" : "A", "score" : 5 }, { "date" : ISODate("2013-08-29T00:00:00Z"), "grade" : "A", "score" : 2 }, { "date" : ISODate("2013-03-08T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2012-06-27T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2011-11-17T00:00:00Z"), "grade" : "A", "score" : 12 } ], "name" : "(Lewis Drug Store) Locanda Vini E Olii", "restaurant\_id" : "40804423" }



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

> db.addresses.aggregate([ {$sort:{name:-1,address:-1,borough:-1,cuisine:-1,grades:-1,restaurant\_id:-1}} ])

{ "\_id" : ObjectId("620453cca79d51c831374e24"), "address" : { "building" : "6946", "coord" : [ -73.8811834, 40.7017759 ], "street" : "Myrtle Avenue", "zipcode" : "11385" }, "borough" : "Queens", "cuisine" : "German", "grades" : [ { "date" : ISODate("2014-09-24T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2014-04-17T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2013-03-12T00:00:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2012-10-02T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2012-05-09T00:00:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2011-12-28T00:00:00Z"), "grade" : "B", "score" : 24 } ], "name" : "Zum Stammtisch", "restaurant\_id" : "40367377" }



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.addresses.aggregate([ {$sort:{cuisine:1,borough:-1}} ])

{ "\_id" : ObjectId("620453cca79d51c83137544e"), "address" : { "building" : "1345", "coord" : [ -73.959249, 40.768076 ], "street" : "2 Avenue", "zipcode" : "10021" }, "borough" : "Manhattan", "cuisine" : "Afghan", "grades" : [ { "date" : ISODate("2014-10-07T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2013-10-23T00:00:00Z"), "grade" : "A", "score" : 8 }, { "date" : ISODate("2012-10-26T00:00:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2012-04-26T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2012-01-12T00:00:00Z"), "grade" : "P", "score" : 10 } ], "name" : "Afghan Kebab House", "restaurant\_id" : "40552806" }



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

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

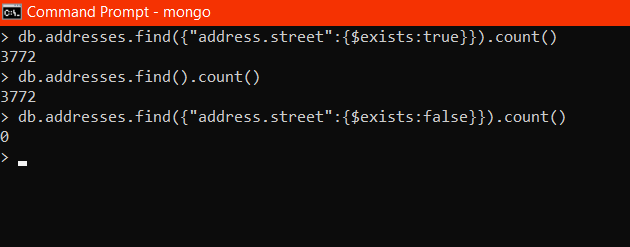
3772

> db.addresses.find().count()

3772

> db.addresses.find({"address.street":{$exists:false}}).count()

0



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

> db.addresses.aggregate([ {$match:{$and:[{"address.coord.0":{$exists:true}},{"address.coord.1":{$exists:true}}]}}])

{ "\_id" : ObjectId("620453cca79d51c831374d67"), "address" : { "building" : "469", "coord" : [ -73.961704, 40.662942 ], "street" : "Flatbush Avenue", "zipcode" : "11225" }, "borough" : "Brooklyn", "cuisine" : "Hamburgers", "grades" : [ { "date" : ISODate("2014-12-30T00:00:00Z"), "grade" : "A", "score" : 8 }, { "date" : ISODate("2014-07-01T00:00:00Z"), "grade" : "B", "score" : 23 }, { "date" : ISODate("2013-04-30T00:00:00Z"), "grade" : "A", "score" : 12 }, { "date" : ISODate("2012-05-08T00:00:00Z"), "grade" : "A", "score" : 12 } ], "name" : "Wendy'S", "restaurant\_id" : "30112340" }

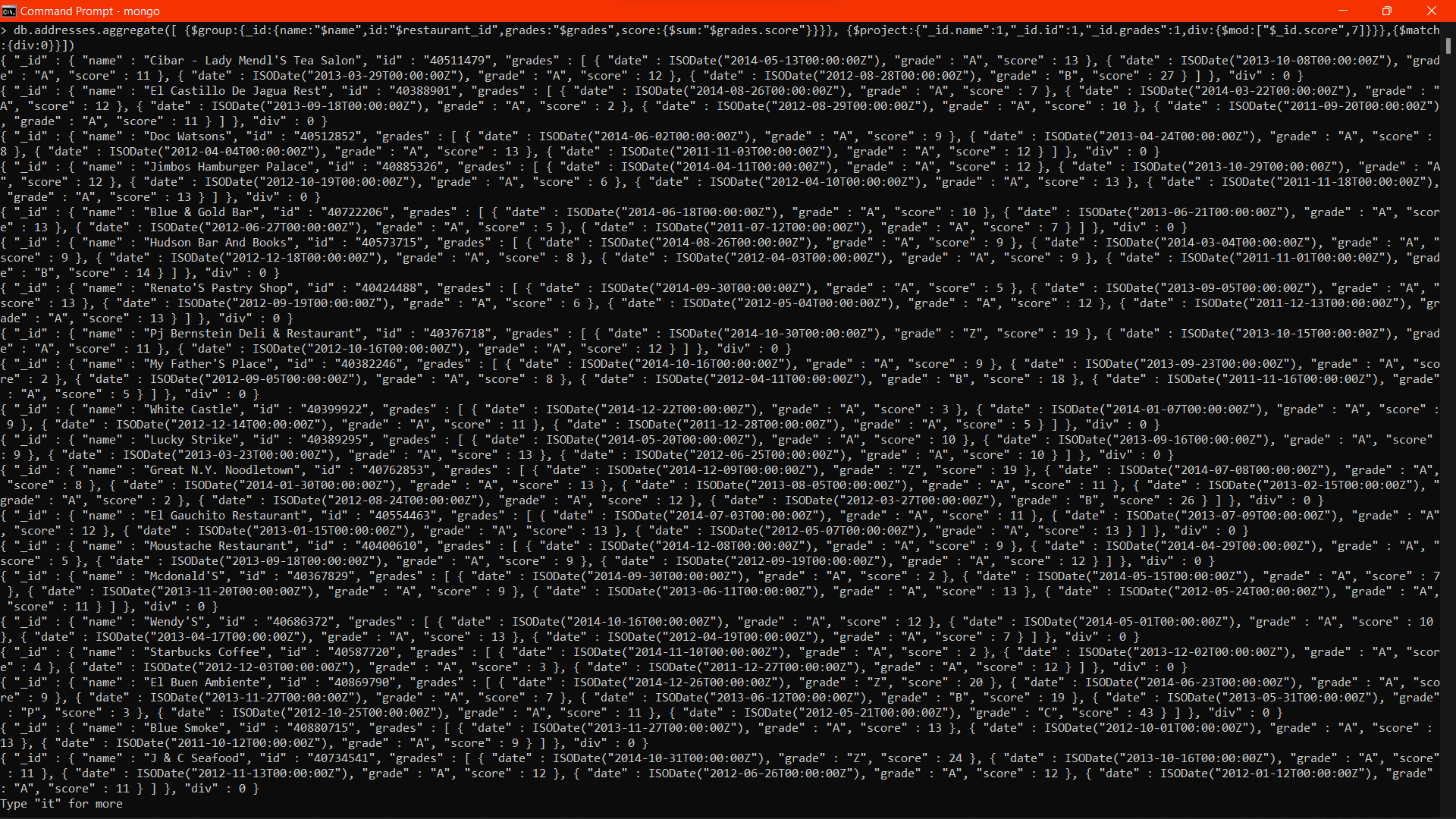


1. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

db.addresses.aggregate([ {$group:{\_id:{name:"$name",id:"$restaurant\_id",grades:"$grades",score:{$sum:"$grades.score"}}}}, {$project:{"\_id.name":1,"\_id.id":1,"\_id.grades":1,div:{$mod:["$\_id.score",7]}}},{$match:{div:0}}])

{ "\_id" : { "name" : "Cibar - Lady Mendl'S Tea Salon", "id" : "40511479", "grades" : [ { "date" : ISODate("2014-05-13T00:00:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2013-10-08T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2013-03-29T00:00:00Z"), "grade" : "A", "score" : 12 }, { "date" : ISODate("2012-08-28T00:00:00Z"), "grade" : "B", "score" : 27 } ] }, "div" : 0 }

{ "\_id" : { "name" : "El Castillo De Jagua Rest", "id" : "40388901", "grades" : [ { "date" : ISODate("2014-08-26T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2014-03-22T00:00:00Z"), "grade" : "A", "score" : 12 }, { "date" : ISODate("2013-09-18T00:00:00Z"), "grade" : "A", "score" : 2 }, { "date" : ISODate("2012-08-29T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2011-09-20T00:00:00Z"), "grade" : "A", "score" : 11 } ] }, "div" : 0 }



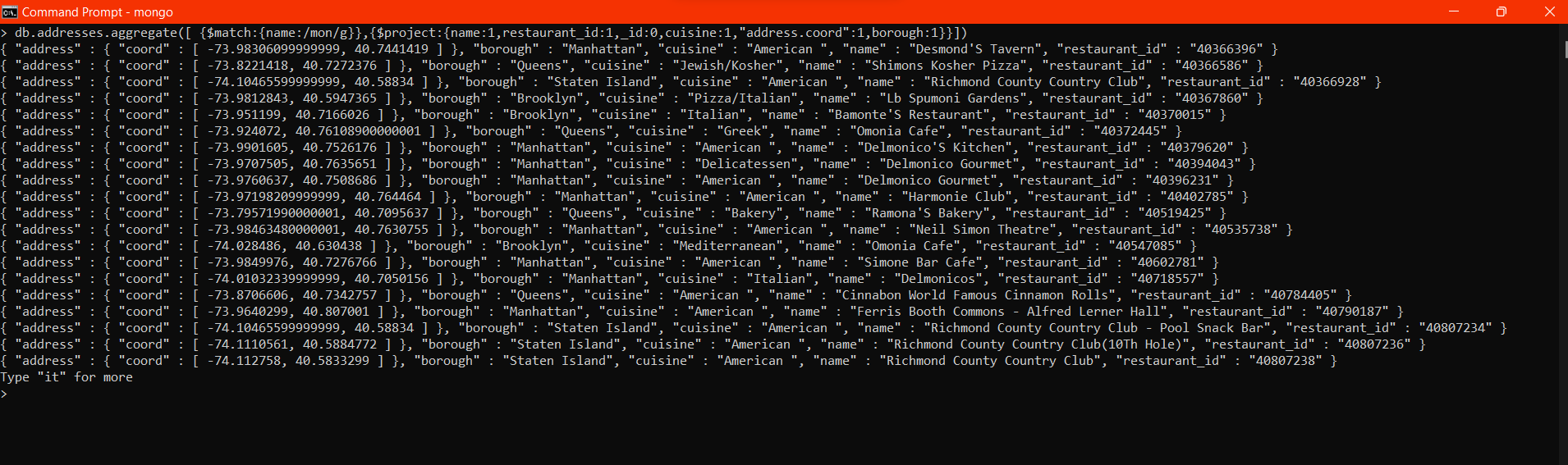
1. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

> db.addresses.aggregate([ {$match:{name:/mon/g}},{$project:{name:1,restaurant\_id:1,\_id:0,cuisine:1,"address.coord":1,borough:1}}])

{ "address" : { "coord" : [ -73.98306099999999, 40.7441419 ] }, "borough" : "Manhattan", "cuisine" : "American ", "name" : "Desmond'S Tavern", "restaurant\_id" : "40366396" }

{ "address" : { "coord" : [ -73.8221418, 40.7272376 ] }, "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Shimons Kosher Pizza", "restaurant\_id" : "40366586" }

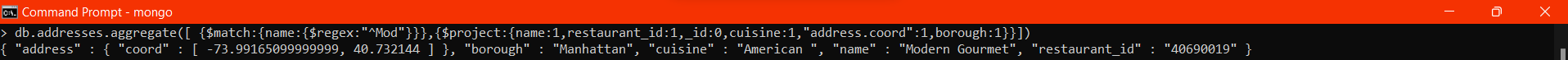
{ "address" : { "coord" : [ -74.10465599999999, 40.58834 ] }, "borough" : "Staten Island", "cuisine" : "American ", "name" : "Richmond County Country Club", "restaurant\_id" : "40366928" }



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.addresses.aggregate([ {$match:{name:{$regex:"^Mod"}}},{$project:{name:1,restaurant\_id:1,\_id:0,cuisine:1,"address.coord":1,borough:1}}])

{ "address" : { "coord" : [ -73.99165099999999, 40.732144 ] }, "borough" : "Manhattan", "cuisine" : "American ", "name" : "Modern Gourmet", "restaurant\_id" : "40690019" }



Happy Coding!!!