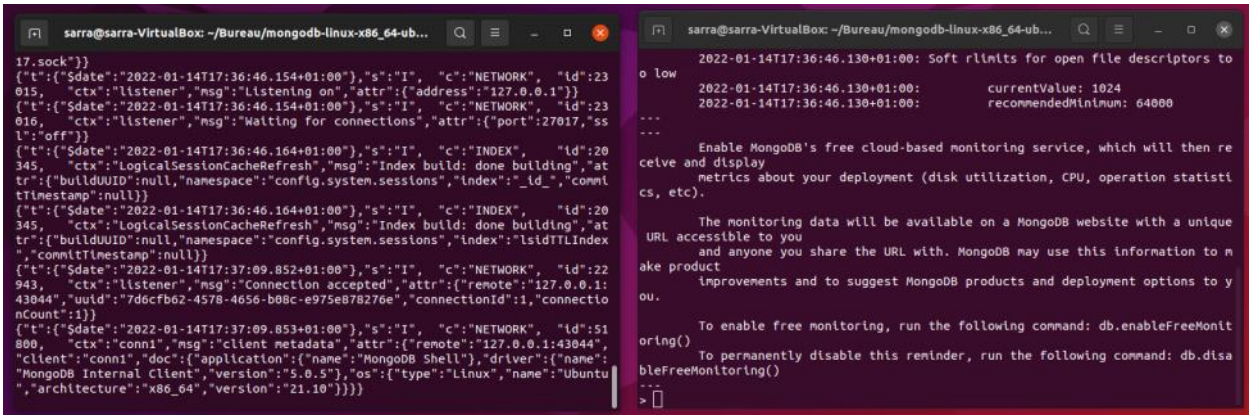
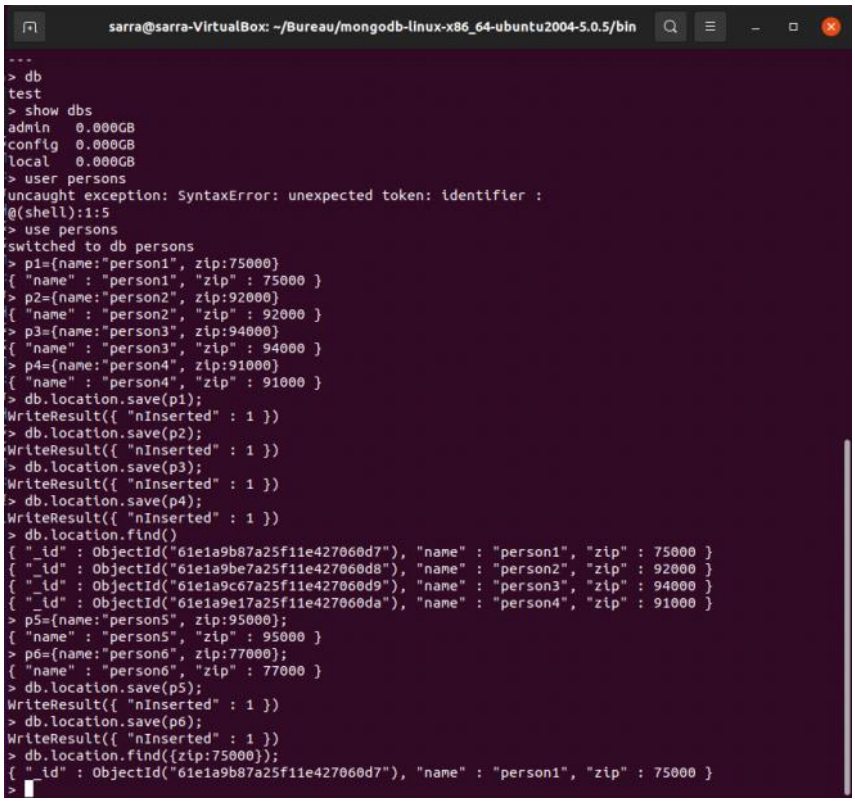
NoSQL - Lab 1.

vendredi 14 janvier 2022 17:29

A - Installing and running the MongoDB Server.

=> Server Launching : **./mongod**
=> Client Launching : **./mongo**





Pour quitter : **quit()**

Sinon, on peut faire l'installation Linux en ligne de commande : Comment installer MongoDB sur Ubuntu 20.04 | DigitalOcean
Et installer mongosh : Install mongosh — MongoDB Shell

C'est cette dernière méthode que j'ai privilégié, le serveur tourne en processus de fond en lançant "systemctl start mongodb". On lance ensuite un client avec le shell de MongoDB en tapant "mongo" ou "mongosh".

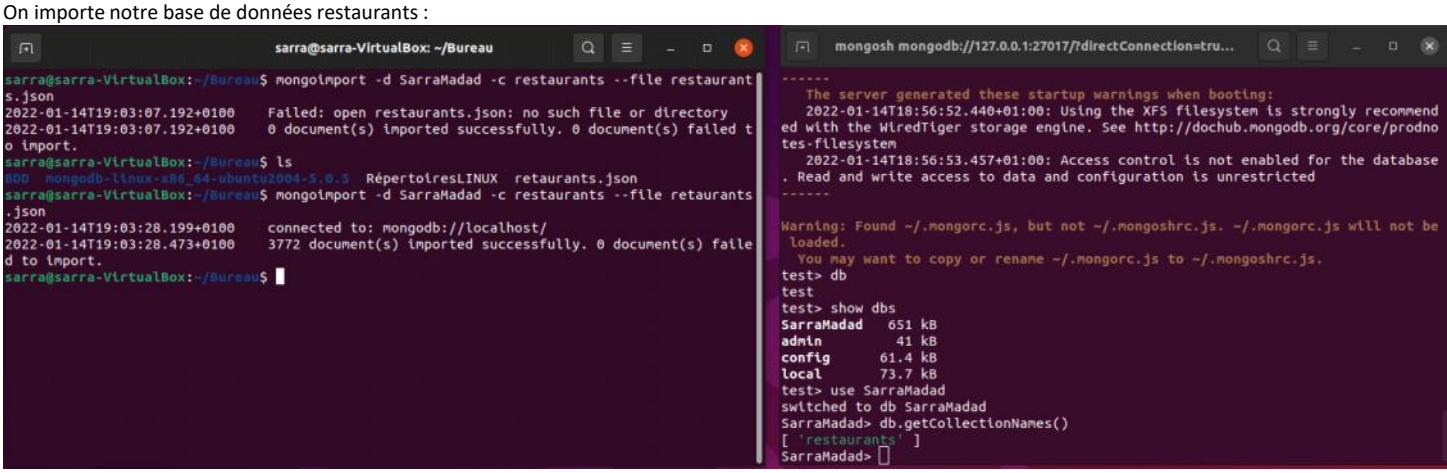
B - Create and fill a database

=> à l'installation de MongoDB, aucune base de données existe. Pour créer une base de données :

- On peut la créer et insérer les enregistrements manuellement.
- On peut la créer et importer les données en format JSON.
- Un serveur MongoDB peut stocker plusieurs bases de données qui peuvent elles-mêmes contenir des collections (= tables)

- 1 ligne/tuple = un document

On importe notre base de données restaurants :



C - Querying a collection

1. **Montrer tous les documents dans la collection "restaurants".**

```
SarraMadad> show collections
restaurants
SarraMadad> db.restaurants.find()
```

2. **Afficher les champs restaurant_id, name, borough et cuisine de tous les documents de la collection "restaurants".**

```
SarraMadad> db.restaurants.find({}, {"restaurant_id":1, "name":1, "borough":1, "cuisine":1});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27138"),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27139"),
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
```

3. **Afficher les champs restaurant_id, name, borough et cuisine et exclure le champ _id de tous les documents de la collection "restaurants".**

```
SarraMadad> db.restaurants.find({}, {"_id":0, "restaurant_id":1, "name":1, "borough":1, "cuisine":1});
[
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Irish',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841'
  },
```

4. **Afficher les champs restaurant_id, name, borough, zip code et exclure le champs _id de tous les documents de la collection "restaurants".**

```
SarraMadad> db.restaurants.find({}, {"_id":0, "restaurant_id":1, "name":1, "borough":1, "zip code":1});
[
  {
    borough: 'Bronx',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  { borough: 'Brooklyn', name: "Wendy'S", restaurant_id: '30112340' },
  {
    borough: 'Manhattan',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841'
  },
  {
    borough: 'Brooklyn',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  },
```

5. **Afficher tous les restaurants dont borough = Bronx.**

```
SarraMadad> db.restaurants.find({"borough":"Bronx"});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27138"),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate("2014-03-03T00:00:00.000Z"),
        grade: 'A',
        score: 2
      },
```

6. **Afficher les 5 premiers restaurants qui sont dans le Bronx**

```
SarraMadad> db.restaurants.find({"borough":"Bronx"}).limit(5);
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27138"),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
```

7. **Afficher les 5 prochains restaurants après avoir sauté les 5 premiers qui sont dans le Bronx.**

```
SarraMadad> db.restaurants.find({"borough":"Bronx"}).skip(5).limit(5);
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27175"),
    address: {
      building: '658',
      coord: [ -73.81363999999999, 40.82941100000001 ],
      street: 'Clarence Ave',
      zipcode: '10465'
    },
    borough: 'Bronx',
    cuisine: 'American ',
    grades: [
```

8. **Trouver les restaurants qui ont eu un score de plus de 90.**

```
SarraMadad> db.restaurants.find({"grades":{$elemMatch : {"score":{$gt:90}}}}).count()
9
SarraMadad> db.restaurants.find({"grades":{$elemMatch : {"score":{$gt:90}}}});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27296"),
    address: {
      building: '65',
      coord: [ -73.9782725, 40.7624022 ],
      street: 'West   54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American ',
    grades: [
      {
        date: ISODate("2014-08-22T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2014-03-28T00:00:00.000Z"),
        grade: 'C',
        score: 131
```

On aurait aussi pu utiliser la commande :

```
SarraMadad> db.restaurants.find({"grades.score":{$gt:90}}).count();
9
SarraMadad> db.restaurants.find({"grades.score":{$gt:90}});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27296"),
    address: {
      building: '65',
      coord: [ -73.9782725, 40.7624022 ],
      street: 'West   54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American ',
    grades: [
      {
        date: ISODate("2014-08-22T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2014-03-28T00:00:00.000Z"),
        grade: 'C',
        score: 131
```

Si on voulait que CHAQUE score soit supérieur à 90, il aurait fallu mettre {$gt : 90, $not : {$lte : 90}}
Mais cela retourne 0 !

9. **Trouver les restaurants qui ont eu un score de plus de 80 mais de moins de 100.**

```
SarraMadad> db.restaurants.find({"grades":{$elemMatch : {"score":{$gt:80, $lt:100}}}}).count()
9
SarraMadad> db.restaurants.find({"grades":{$elemMatch : {"score":{$gt:80, $lt:100}}}});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27337"),
    address: {
      building: '345',
      coord: [ -73.9864626, 40.7266739 ],
      street: 'East 6 Street',
      zipcode: '10003'
    },
    borough: 'Manhattan',
    cuisine: 'Indian',
    grades: [
      {
```

/!\ elemMatch permet de vérifier élément par élément. Sans elemMatch nous nous serions retrouvés avec des scores à 131 puisque la premières conditions étaient vérifiées !

This happens because both conditions are true. When you don't specify indices for `grades.score` it search through all elements of the array. And it **does** find an item with `score > 80` (which is 131). At the same time there exist other items that are less then 100.

In human language you request "a document that contains score greater then 80 and at the same time contains score less then 100". You don't specify that it should be the same score that satisfy these conditions.

What you're looking for I guess is a query for simultaneously checking range for each element.

```
{ "grades" : { $elemMatch: { score: { $gt:80, $lt:100 } } } }
```

10. **Trouver les restaurants qui ont une valeur de latitude inférieure à -95.754168.**

La latitude est le premier élément du tableau de coordonnées, la longitude le second élément.
On pourrait aussi récupérer le premier élément avec address.coord.0.

```
SarraMadad> db.restaurants.find({"address.coord":{$lt:-95.754168}}).count();
9
SarraMadad> db.restaurants.find({"address.coord":{$lt:-95.754168}});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27780"),
    address: {
      building: '3707',
      coord: [ -101.8945214, 33.5197474 ],
      street: '82 Street',
      zipcode: '11372'
    },
    borough: 'Queens',
    cuisine: 'American ',
    grades: [
```

```
SarraMadad> db.restaurants.find({"address.coord.0":{$lt:-95.754168}}).count();
9
SarraMadad> db.restaurants.find({"address.coord.0":{$lt:-95.754168}});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27780"),
    address: {
      building: '3707',
      coord: [ -101.8945214, 33.5197474 ],
      street: '82 Street',
      zipcode: '11372'
    },
    borough: 'Queens',
    cuisine: 'American ',
    grades: [
      {
        date: ISODate("2014-06-04T00:00:00.000Z"),
        grade: 'A',
        score: 12
      },
      {
        date: ISODate("2013-11-07T00:00:00.000Z"),
        grade: 'B',
        score: 19
```

11. **Trouver les restaurants qui ne préparent pas de cuisine "américaine", ont un score de plus de 70 et sont situés sur une latitude inférieure à -65,754168.**

/!\ pour le champ "American " il y a un espace entre le "n" et le dernier ".

```
SarraMadad> db.restaurants.find({$and:[{"cuisine":{$ne:"American "}}, {"grades.score":{$gt:70}}, {"address.coord.0"
:{$lt:-65.754168}}]}).count();
15
SarraMadad> db.restaurants.find({$and:[{"cuisine":{$ne:"American "}}, {"grades.score":{$gt:70}}, {"address.coord.0"
:{$lt:-65.754168}}]});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27337"),
    address: {
      building: '345',
      coord: [ -73.9864626, 40.7266739 ],
      street: 'East 6 Street',
      zipcode: '10003'
    },
    borough: 'Manhattan',
    cuisine: 'Indian',
    grades: [
```

12. **Trouver les restaurants qui ne préparent pas de cuisine "américaine", ont un score de plus de 70 et sont situés sur une longitude inférieure à -65,754168 (sans utiliser l'opérateur $and).**

```
SarraMadad> db.restaurants.find({"cuisine":{$ne:"American "}, "grades.score":{$gt:70}, "address.coord.1":{$lt:-65.7
54168}});
```

13. **Trouver les restaurants qui ne préparent pas de cuisine "américaine", ont un score A et n'appartiennent pas au quartier de Brooklyn. Le document doit être affiché avec la cuisine triée par ordre décroissant.**

```
SarraMadad> db.restaurants.find({$and:[{"cuisine":{$ne:"American "}}, {"grades.grade":"A"}, {"borough":{$ne:"
Brooklyn"}}]}).sort({cuisine:-1});
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27844"),
    address: {
      building: '89',
      coord: [ -73.9995899, 40.7168015 ],
      street: 'Baxter Street',
      zipcode: '10013'
    },
    borough: 'Manhattan',
    cuisine: 'Vietnamese/Cambodian/Malaysia',
    grades: [
      {
        date: ISODate("2014-08-21T00:00:00.000Z"),
        grade: 'A',
        score: 13
```

14. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui contiennent "Wil" comme premières lettres de leur nom.**

```
SarraMadad> db.restaurants.find({"name":/^Wil/i}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1}).cou
nt();
9
SarraMadad> db.restaurants.find({"name":/^Wil/i}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Brooklyn',
    cuisine: 'Delicatessen',
    name: "Wilken'S Fine Food",
    restaurant_id: '40356483'
  },
  {
    borough: 'Bronx',
    cuisine: 'American ',
    name: 'Wild Asia',
    restaurant_id: '40357217'
  },
```

15. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui contiennent "ces" comme dernières lettres de leur nom.**

```
SarraMadad> db.restaurants.find({"name":/ces$/i}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1}).cou
nt();
18
SarraMadad> db.restaurants.find({"name":/ces$/i}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Manhattan',
    cuisine: 'American ',
    name: 'Pieces',
    restaurant_id: '40399910'
  },
  {
    borough: 'Queens',
    cuisine: 'American ',
    name: 'S.M.R Restaurant Services',
    restaurant_id: '40403857'
  },
```

16. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui contiennent "Reg" comme lettres quelque part dans leur nom.**

```
SarraMadad> db.restaurants.find({"name":/Reg/}, {_id:0, restaurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Brooklyn',
    cuisine: 'American ',
    name: 'Regina Caterers',
    restaurant_id: '40356649'
  },
  {
    borough: 'Manhattan',
    cuisine: 'Café/Coffee/Tea',
    name: 'Caffe Reggio',
    restaurant_id: '40369418'
  },
```

17. **Trouver les restaurants qui se situent dans le Bronx et qui préparent de la cuisine américaine, chinoise.**

```
SarraMadad> db.restaurants.find({borough:"Bronx", cuisine:{$in: ["American", "Chinese"]}}, {_id:0, restaurant
_id:1, name:1, borough:1, cuisine:1}).count();
48
SarraMadad> db.restaurants.find({borough:"Bronx", cuisine:{$in: ["American", "Chinese"]}}, {_id:0, restaurant
_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Bronx',
    cuisine: 'Chinese',
    name: 'Happy Garden',
    restaurant_id: '40363289'
  },
  {
    borough: 'Bronx',
    cuisine: 'Chinese',
    name: 'Happy Garden',
    restaurant_id: '40364296'
  },
```

18. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui sont dans le quartier Staten Island, Queens ou Bronxor Brooklyn.**

```
SarraMadad> db.restaurants.find({"borough":{$in: ["Staten Island", "Queens","Bronx", "Brooklyn"]}}, {_id:0, r
estaurant_id:1, name:1, borough:1, cuisine:1}).count();
5667
SarraMadad> db.restaurants.find({borough:{$in: ["Staten Island", "Queens","Bronx", "Brooklyn"]}}, {_id:0, res
taurant_id:1, name:1, borough:1, cuisine:1}).count();
5667
SarraMadad> db.restaurants.find({"borough":{$in: ["Staten Island", "Queens","Bronx", "Brooklyn"]}}, {_id:0, r
estaurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
```

19. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui ne sont pas dans le quartier Staten Island, Queens ou Bronxor Brooklyn.**

```
SarraMadad> db.restaurants.find({borough:{$nin: ["Staten Island", "Queens","Bronx", "Brooklyn"]}}, {_id:0, re
staurant_id:1, name:1, borough:1, cuisine:1}).count();
5649
SarraMadad> db.restaurants.find({borough:{$nin: ["Staten Island", "Queens","Bronx", "Brooklyn"]}}, {_id:0, re
staurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    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'
  },
```

20. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui ont eu un score qui ne dépassent pas 10.**

```
SarraMadad> db.restaurants.find({"grades.score":{$lte:10}}, {_id:0, restaurant_id:1, name:1, borough:1, cuisi
ne:1});
[
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
```

21. **Trouver l'id, le nom, le quartier et la cuisine des restaurants qui préparent de la cuisine qui n'est pas américaine ou chinoise OU les restaurants dont le nom commence par "Wil".**

```
SarraMadad> db.restaurants.find({$or: [{"cuisine":{$nin: ["American ", "Chinese"]}}, {"name":/^Wil/}]}, {_id:
0, restaurant_id:1, name:1, borough:1, cuisine:1}).count();
7209
SarraMadad> db.restaurants.find({$or: [{"cuisine":{$nin: ["American ", "Chinese"]}}, {"name":/^Wil/}]}, {_id:
0, restaurant_id:1, name:1, borough:1, cuisine:1});
[
  {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  },
```

22. **Trouver l'id, le nom et les notes des restaurants qui ont eu un A et un score de 11 à une ISODate "2014-08-11T00 :00 :00Z" parmi plusieurs dates d'enquêtes.**

```
SarraMadad> db.restaurants.find({"grades":{$elemMatch: {"grade":"A", "score":11, "date":ISODate(
2014-08-11T00:00:00Z")}}}, {_id:0, restaurant_id:1, name:1, grades:1}).count();
3
SarraMadad> db.restaurants.find({"grades":{$elemMatch: {"grade":"A", "score":11, "date":ISODate("
2014-08-11T00:00:00Z")}}}, {_id:0, restaurant_id:1, name:1, grades:1});
[
  {
    grades: [
      {
        date: ISODate("2014-08-11T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
```

23. **Trouver l'id, le nom et les notes des restaurants dont le second élément du tableau de notes contient un A et un score de 9 à une ISODate "2014-08-11T00 :00 :00Z".**

```
SarraMadad> db.restaurants.find({$and: [{"grades.1.grade":"A"}, {"grades.1.score":9}, {"grades.1.
date":ISODate("2014-08-11T00:00:00Z")}]}, {_id:0, restaurant_id:1, name:1, grades:1}).count();
3
SarraMadad> db.restaurants.find({$and: [{"grades.1.grade":"A"}, {"grades.1.score":9}, {"grades.1.
date":ISODate("2014-08-11T00:00:00Z")}]}, {_id:0, restaurant_id:1, name:1, grades:1});
[
  {
    grades: [
      {
        date: ISODate("2015-01-12T00:00:00.000Z"),
        grade: 'A',
        score: 10
      },
Aide  {
        date: ISODate("2014-08-11T00:00:00.000Z"),
        grade: 'A',
        score: 9
      },
```

24. **Trouver l'id, le nom, l'adresses et la localisation géographique des restaurants dont le second élément du tableau de coordonnées contient une valeur qui peut aller de 42 à 52.**

```
SarraMadad> db.restaurants.find({$and: [{"address.coord.1":{$gt:42}}, {"address.coord.1":{$lte:52
}}]}, {_id:0, restaurant_id:1, name:1, address:1}).count();
21
SarraMadad> db.restaurants.find({$and: [{"address.coord.1":{$gt:42}}, {"address.coord.1":{$lte:52
}}]}, {_id:0, restaurant_id:1, name:1, address:1});
[
  {
    address: {
      building: '47',
      coord: [ -78.877224, 42.89546199999999 ],
      street: 'Broadway @ Trinity Pl',
      zipcode: '10006'
    },
    name: "T.G.I. Friday'S",
    restaurant_id: '40387990'
  },
```

25. **Trier les noms des restaurants par ordre croissant avec toutes les colonnes.**

```
SarraMadad> db.restaurants.find().sort({name:1});
```

```
SarraMadad> db.restaurants.find({}, {_id:0, name:1}).sort({name:1});
[
  { name: '(Lewis Drug Store) Locanda Vini E Olii' },
  { name: '(Lewis Drug Store) Locanda Vini E Olii' },
  { name: '(Lewis Drug Store) Locanda Vini E Olii' },
  { name: '1 East 66Th Street Kitchen' },
  { name: '1 East 66Th Street Kitchen' },
  { name: '1 East 66Th Street Kitchen' },
  { name: '101 Deli' },
  { name: '101 Deli' },
  { name: '101 Deli' },
  { name: '101 Restaurant And Bar' },
  { name: '101 Restaurant And Bar' },
  { name: '101 Restaurant And Bar' },
  { name: '1020 Bar' },
  { name: '1020 Bar' },
  { name: '1020 Bar' },
  { name: '104-01 Foster Avenue Coffee Shop(Ups)' },
  { name: '104-01 Foster Avenue Coffee Shop(Ups)' },
  { name: '104-01 Foster Avenue Coffee Shop(Ups)' },
  { name: '10Th Avenue Pizza & Cafe' },
  { name: '10Th Avenue Pizza & Cafe' }
```

26. **Trier les noms des restaurants par ordre décroissant avec toutes les colonnes.**

```
SarraMadad> db.restaurants.find().sort({name:-1});
```

```
SarraMadad> db.restaurants.find({}, {_id:0, name:1}).sort({name:-1});
[
  { name: 'Zum Stammtisch' },
  { name: 'Zum Stammtisch' },
  { name: 'Zum Stammtisch' },
  { name: 'Zum Schneider' },
  { name: 'Zum Schneider' },
  { name: 'Zum Schneider' },
  { name: "Zorba'S" },
  { name: "Zorba'S" },
  { name: "Zorba'S" },
  { name: 'Zebu Grill' },
  { name: 'Zebu Grill' },
  { name: 'Zebu Grill' },
```

27. **Trier les noms des cuisines par ordre croissant et pour chaque type de cuisine, trier les quartiers par ordre décroissant.**

```
SarraMadad> db.restaurants.find({}, {_id:0, cuisine:1, borough:1}).sort({cuisine:1, borough:-1});

[
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Manhattan', cuisine: 'Afghan' },
  { borough: 'Queens', cuisine: 'African' },
  { borough: 'Queens', cuisine: 'African' },
  { borough: 'Queens', cuisine: 'African' },
  { borough: 'Brooklyn', cuisine: 'African' },
  { borough: 'Brooklyn', cuisine: 'African' },
  { borough: 'Brooklyn', cuisine: 'African' },
  { borough: 'Bronx', cuisine: 'African' },
  { borough: 'Bronx', cuisine: 'African' }
```

28. **Savoir si toute les adresses contiennent une rue ou non.**

```
SarraMadad> db.restaurants.find({"address.street":{$exists:true}}).count()
11316
SarraMadad> db.restaurants.find({"address.street":{$exists:false}}).count()
0
```

29. **Sélectionner tous les documents dans la collection "restaurants" dont le champ coord est double.**

```
SarraMadad> db.restaurants.find({"address.coord":{$type:"double"}}).pretty()
[
  {
    _id: ObjectId("61e1baf09a0fb0e6a9a27138"),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
```

30. **Sélectionner l'id, le nom et les dates des restaurants qui retournent un reste à 0 suite à la division du score par 7.**

```
SarraMadad> db.restaurants.find({"grades.score":{$mod: [7,0]}}, {_id:0, restaurant_id:1, name:1,
grades:1}).count();
4755
SarraMadad> db.restaurants.find({"grades.score":{$mod: [7,0]}}, {_id:0, restaurant_id:1, name:1,
grades:1});
```

31. **Trouver le nom, le quartier, la longitude, la latitude et la cuisine des restaurants dont le nom**

a "mon" quelque part dans son contenu.

```
SarraMadad> db.restaurants.find({"name":/mon/}, {_id:0, name:1, "address.coord":1, borough:1, cui
sine:1}).count()
63
SarraMadad> db.restaurants.find({"name":/mon/}, {_id:0, name:1, "address.coord":1, borough:1, cui
sine:1});
[
  {
    address: { coord: [ -73.98306099999999, 40.7441419 ] },
    borough: 'Manhattan',
    cuisine: 'American ',
    name: "Desmond'S Tavern"
  },
```

32. **Trouver le nom, le quartier, la longitude, la latitude et la cuisine des restaurant dont le nom contient "Mad" comme premières lettres.**

```
SarraMadad> db.restaurants.find({"name":/^Mad/i}, {_id:0, name:1, "address.coord":1, borough:1, c
uisine:1}).count();
24
SarraMadad> db.restaurants.find({"name":/^Mad/i}, {_id:0, name:1, "address.coord":1, borough:1, c
uisine:1});
[
  {
    address: { coord: [ -73.9860597, 40.7431194 ] },
    borough: 'Manhattan',
    cuisine: 'American ',
    name: 'Madison Square'
  },
  {
    address: { coord: [ -73.98302199999999, 40.742313 ] },
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Madras Mahal'
```