

Université des Sciences et de la Technologie Houari Boumediene  
Faculté d'Informatique



TP Bio SGBD

---

Rapport TP 4 : TP MongoDB

---

Fait par :

Nom et prénom : ABDELMALEK BENMEZIANE

Matricule : 171731046778

Spécialité : M1 BIOINFO

Section : A

# Contents

<b>1</b>	<b>TP MongoDB</b>	<b>1</b>
1.1	Téléchargement de l'archive restaurant . . . . .	1
1.2	L'extraction de l'archive restaurant . . . . .	1
1.3	La création une base de données "new_york" et une collection "restaurants" . . . . .	2
1.4	Dans une console aller dans le répertoire MONGO/bin . . . . .	3
1.5	L'exécution de la commande . . . . .	3
1.6	Affichage d'un élément au hasard (findOne()) . . . . .	4
1.7	Afficher les noms des restaurants de borough "Brooklyn" . . . . .	5
1.7.1	Donner le nombre de résultat (count) . . . . .	6
1.7.2	L'attribut cuisine : italien . . . . .	6
1.7.3	Ajouter address.street . . . . .	7
1.8	A partir de la requête précédente . . . . .	7
1.9	Les restaurants de Manhattan . . . . .	8
1.10	Afficher les restaurants des Quartiers de NewYork de manière unique (Distinct). . . . .	8
1.11	La liste des grades données par les inspecteurs . . . . .	9
1.12	La même requête en utilisant les agrégats (aggregate : match & project) . . . . .	9
1.12.1	With project . . . . .	9
1.12.2	With match . . . . .	10
1.13	Ajouter un commentaire sur un restaurant . . . . .	10

1.14	supprimer une clé . . . . .	11
1.15	Attribuer un commentaire en fonction des grades obtenus . .	11
1.16	La meilleur note . . . . .	12
1.17	Supprimer les restaurants où le score egale à 0 . . . . .	14
1.18	Exécution du code avec commentaire . . . . .	14
1.18.1	Exécution . . . . .	14
1.18.2	Commentaire . . . . .	14

# List of Figures

1.1	Restaurant . . . . .	1
1.2	Extraction . . . . .	1
1.3	Create db . . . . .	2
1.4	Create collection . . . . .	2
1.5	Cmd . . . . .	3
1.6	Display one item . . . . .	4
1.7	Display name restaurant . . . . .	5
1.8	Count . . . . .	6
1.9	Cuisine italien . . . . .	6
1.10	address.street . . . . .	7
1.11	Grade.score . . . . .	7
1.12	Manhattan . . . . .	8
1.13	New york . . . . .	8
1.14	Grades . . . . .	9
1.15	Project . . . . .	9
1.16	Match . . . . .	10
1.17	Update . . . . .	10
1.18	Delete . . . . .	11
1.19	Update unset . . . . .	11
1.20	Rang . . . . .	12
1.21	Output . . . . .	13
1.22	delete restaurant . . . . .	14
1.23	Execution . . . . .	14

# Chapter 1

## TP MongoDB

### 1.1 Téléchargement de l'archive restaurant



restaurants.json\_

---

Figure 1.1: Restaurant

### 1.2 L'extraction de l'archive restaurant



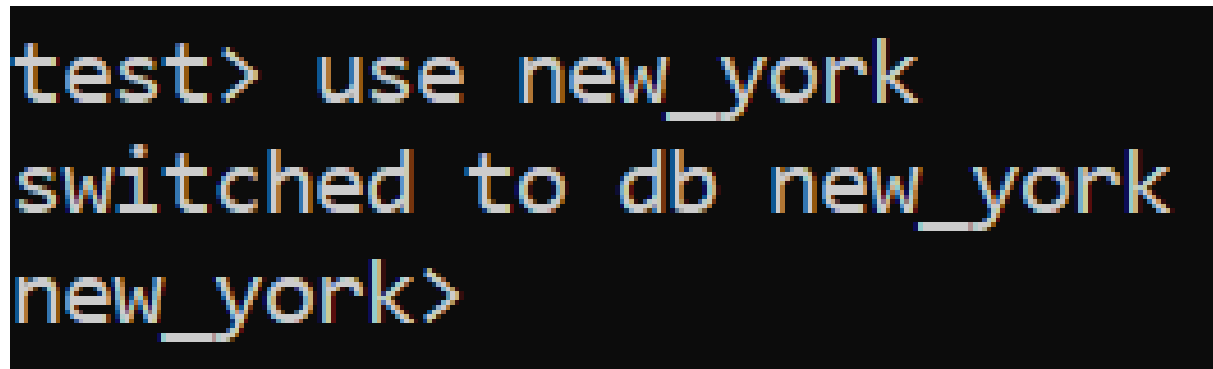
RESTAURANT

Figure 1.2: Extraction

### 1.3 La création une base de données "new\_york" et une collection "restaurants"

On ouvre le mongosh qui est le shell de la base de données mongoDB

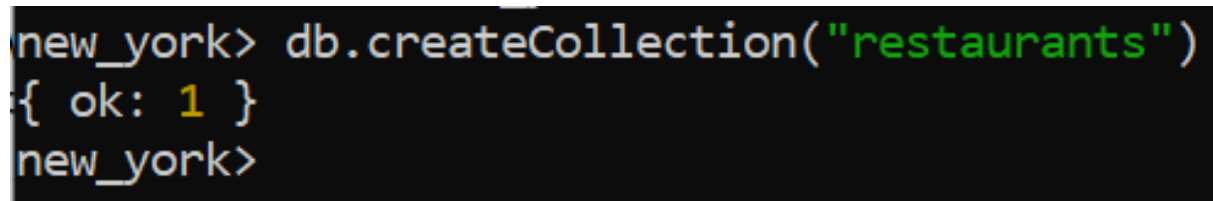
On crée la base de données "new\_york"

A terminal window with a black background and yellow text. The text shows the command 'use new\_york' being entered, followed by the prompt 'switched to db new\_york' and then 'new\_york>'.

```
test> use new_york
switched to db new_york
new_york>
```

**Figure 1.3:** Create db

On crée la collection "restaurants"

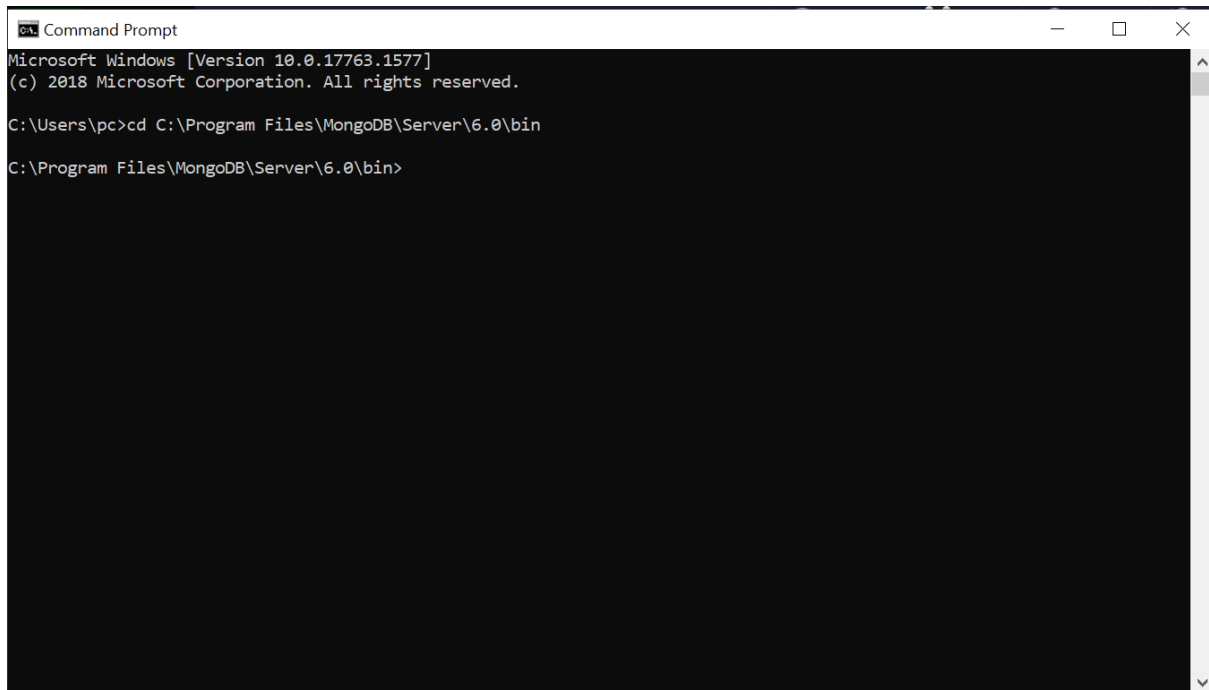
A terminal window with a black background and yellow text. The text shows the command 'db.createCollection("restaurants")' being entered, followed by the response '{ ok: 1 }' and then 'new\_york>'.

```
new_york> db.createCollection("restaurants")
{ ok: 1 }
new_york>
```

**Figure 1.4:** Create collection

On utilisant l'interface graphique Compass, on importe le fichier : restaurants.json

## 1.4 Dans une console aller dans le répertoire MONGO/bin



```
Command Prompt
Microsoft Windows [Version 10.0.17763.1577]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\pc>cd C:\Program Files\MongoDB\Server\6.0\bin
C:\Program Files\MongoDB\Server\6.0\bin>
```

Figure 1.5: Cmd

## 1.5 L'exécution de la commande

Cette commande ne marche pas avec la version 6.0 de mongoDB

## 1.6 Affichage d'un élément au hasard (findOne())

```
new_york> db.restaurants.findOne()
{
  _id: ObjectId("64377064f87cc54ccacfe202"),
  address: {
    building: '1007',
    coord: { type: 'Point', coordinates: [ -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 },
    { date: ISODate("2013-09-11T00:00:00.000Z"), grade: 'A', score: 6 },
    {
      date: ISODate("2013-01-24T00:00:00.000Z"),
      grade: 'A',
      score: 10
    },
    { date: ISODate("2011-11-23T00:00:00.000Z"), grade: 'A', score: 9 },
    {
      date: ISODate("2011-03-10T00:00:00.000Z"),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
new_york>
```

Figure 1.6: Display one item



## 1.7 Afficher les noms des restaurants de borough "Brooklyn"

```
new_york> db.restaurants.find({"borough" : "Brooklyn"},{name:1})
[
  { _id: ObjectId("64377064f87cc54ccacfe203"), name: "Wendy'S" },
  {
    _id: ObjectId("64377064f87cc54ccacfe205"),
    name: 'Riviera Caterer'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe209"),
    name: "Wilken'S Fine Food"
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe20a"),
    name: 'Regina Caterers'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe20b"),
    name: 'Taste The Tropics Ice Cream'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe20d"),
    name: 'C & C Catering Service'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe20e"),
    name: 'May May Kitchen'
  },
  { _id: ObjectId("64377064f87cc54ccacfe210"), name: 'Seuda Foods' },
  {
    _id: ObjectId("64377064f87cc54ccacfe211"),
    name: 'Carvel Ice Cream'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe213"),
    name: 'Nordic Delicacies'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe215"),
    name: 'The Movable Feast'
  },
]
```

Figure 1.7: Display name restaurant

### 1.7.1 Donner le nombre de résultat (count)

```
new_york> db.restaurants.find({"borough" : "Brooklyn"},{_id:0, name:1}).count()  
84
```

Figure 1.8: Count

### 1.7.2 L'attribut cuisine : italien

```
new_york> db.restaurants.find({$and: [{"borough" : "Brooklyn"}, {"cuisine" : "Italian"}]}, {name:1})  
[  
  {  
    _id: ObjectId("64377064f87cc54ccacfe23a"),  
    name: 'Philadelphia Grille Express'  
  },  
  { _id: ObjectId("64377064f87cc54ccacfe267"), name: 'New Corner' },  
  {  
    _id: ObjectId("64377064f87cc54ccacfe27c"),  
    name: "Gargiulo'S Restaurant"  
  },  
  {  
    _id: ObjectId("64377064f87cc54ccacfe28c"),  
    name: "Michael'S Restaurant"  
  },  
  {  
    _id: ObjectId("6437817af87cc54ccacfe303"),  
    name: 'Philadelphia Grille Express'  
  },  
  { _id: ObjectId("6437817af87cc54ccacfe330"), name: 'New Corner' },  
  {  
    _id: ObjectId("6437817af87cc54ccacfe345"),  
    name: "Gargiulo'S Restaurant"  
  },  
  {  
    _id: ObjectId("6437817af87cc54ccacfe355"),  
    name: "Michael'S Restaurant"  
  }  
]  
new_york>
```

Figure 1.9: Cuisine italien

### 1.7.3 Ajouter address.street

```
new_york> db.restaurants.find({$and : [{"borough" : "Brooklyn"}, {"cuisine" : "Italian"}]}, {name : 1, "address.street" : 1})
[
  {
    _id: ObjectId("64377064f87cc54ccacfe23a"),
    address: { street: '4 Avenue' },
    name: 'Philadelphia Grille Express'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe267"),
    address: { street: '8 Avenue' },
    name: 'New Corner'
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe27c"),
    address: { street: 'West 15 Street' },
    name: "Gargiulo'S Restaurant"
  },
  {
    _id: ObjectId("64377064f87cc54ccacfe28c"),
    address: { street: 'Avenue R' },
    name: "Michael'S Restaurant"
  },
  {
    _id: ObjectId("6437817af87cc54ccacfe303"),
    address: { street: '4 Avenue' },
    name: 'Philadelphia Grille Express'
  },
  {
    _id: ObjectId("6437817af87cc54ccacfe330"),
    address: { street: '8 Avenue' },
    name: 'New Corner'
  },
  {
    _id: ObjectId("6437817af87cc54ccacfe345"),
    address: { street: 'West 15 Street' },
    name: "Gargiulo'S Restaurant"
  },
  {
    _id: ObjectId("6437817af87cc54ccacfe355"),
    address: { street: 'Avenue R' },
    name: "Michael'S Restaurant"
  }
]
```

Figure 1.10: address.street

## 1.8 A partir de la requête précédente

```
new_york> db.restaurants.find({$and : [{"borough" : "Brooklyn"}, {"cuisine" : "Italian"}]}, {_id : 0, name : 1, "grade.score" : 1})
[
  { name: 'Philadelphia Grille Express' },
  { name: 'New Corner' },
  { name: "Gargiulo'S Restaurant" },
  { name: "Michael'S Restaurant" },
  { name: 'Philadelphia Grille Express' },
  { name: 'New Corner' },
  { name: "Gargiulo'S Restaurant" },
  { name: "Michael'S Restaurant" }
]
new_york>
```

Figure 1.11: Grade.score

## 1.9 Les restaurants de Manhattan

```
new_york> db.restaurants.find({"borough" : "Manhattan","grades.score" :{$lt : 10}},{name: 1,"grades.score" : 1,_id : 0})
[
  {
    grades: [ { score: 2 }, { score: 11 }, { score: 12 }, { score: 12 } ],
    name: 'Dj Reynolds Pub And Restaurant'
  },
  {
    grades: [ { score: 3 }, { score: 4 }, { score: 6 }, { score: 8 } ],
    name: '1 East 66Th Street Kitchen'
  },
  {
    grades: [
      { score: 12 },
      { score: 16 },
      { score: 9 },
      { score: 13 },
      { score: 11 }
    ],
    name: 'Glorious Food'
  },
  {
    grades: [ { score: 12 }, { score: 11 }, { score: 6 }, { score: 8 } ],
    name: "Bully'S Deli"
  },
  {
    grades: [
      { score: 10 },
      { score: 13 },
      { score: 13 },
      { score: 11 },
      { score: 10 },
      { score: 7 }
    ],
    name: "Harriet'S Kitchen"
  },
  {
    grades: [ { score: 26 }, { score: 9 }, { score: 20 }, { score: 12 } ],
    name: 'P & S Deli Grocery'
  },
]
```

Figure 1.12: Manhattan

## 1.10 Afficher les restaurants des Quartiers de NewYork de manière unique (Distinct).

```
new_york> db.restaurants.distinct("name",{borough : "NewYork"})
[]
new_york>
```

Figure 1.13: New york

## 1.11 La liste des grades données par les inspecteurs

```
new_york> db.restaurants.distinct("grades.grade")
[ 'A', 'B', 'C', 'Not Yet Graded', 'P', 'Z' ]
new_york>
```

Figure 1.14: Grades

## 1.12 La même requête en utilisant les agrégats (aggregate : match & project)

### 1.12.1 With project

```
new_york> db.restaurants.aggregate([
...   { $project: { _id: 0, grade: "$grades.grade" } },
...   { $unwind: "$grade" },
...   { $group: { _id: "$grade" } }
... ])
[
  { _id: 'Z' },
  { _id: 'P' },
  { _id: 'Not Yet Graded' },
  { _id: 'A' },
  { _id: 'B' },
  { _id: 'C' }
]
new_york>
```

Figure 1.15: Project

### 1.12.2 With match

```
new_york> db.restaurants.aggregate([
...   { $match: {} },
...   { $project: { _id: 0, grade: "$grades.grade" } },
...   { $unwind: "$grade" },
...   { $group: { _id: "$grade" } }
... ])
[
  { _id: 'C' },
  { _id: 'A' },
  { _id: 'Z' },
  { _id: 'P' },
  { _id: 'Not Yet Graded' },
  { _id: 'B' }
]
```

Figure 1.16: Match

## 1.13 Ajouter un commentaire sur un restaurant

```
new_york> db.restaurants.update (
...   {"_id" : ObjectId("594b9172c96c61e672dcd689")},
...   {$set : {"comment" : "My new comment"}}
... )
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
new_york>
```

Figure 1.17: Update

## 1.14 supprimer une clé

```
new_york> db.restaurants.update (
... {"_id" : ObjectId("594b9172c96c61e672dcd689")},
... {$unset : {"comment" : 1}}
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
new_york>
```

Figure 1.18: Delete

## 1.15 Attribuer un commentaire en fonction des grades obtenus

```
new_york> db.restaurants.update( { "grades.grade": { $not: { $eq: "C" } } }, { $set: { "comment": "acceptable" } });
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
new_york> db.restaurants.find({"comment": "acceptable","grades.grade" : 'C'}).count()
0
new_york> db.restaurants.find({"comment": "acceptable","grades.grade" : 'A'}).count()
1
new_york>
```

Figure 1.19: Update unset

## 1.16 La meilleur note

```
new_york> db.restaurants.aggregate([
...   {
...     $project: {
...       _id: 1,
...       name: 1,
...       grades: 1,
...       score: {
...         $sum: {
...           $switch: {
...             branches: [
...               { case: { $eq: ["$grades.grade", "A"] }, then: 3 },
...               { case: { $eq: ["$grades.grade", "B"] }, then: 1 },
...               { case: { $eq: ["$grades.grade", "C"] }, then: -1 }
...             ],
...             default: 0
...           }
...         }
...       }
...     },
...     {
...       $sort: { score: -1 }
...     },
...     {
...       $limit: 1
...     }
...   ])
```

Figure 1.20: Rang



```

[
  {
    _id: ObjectId("64377064f87cc54ccacfe202"),
    grades: [
      {
        date: ISODate("2014-03-03T00:00:00.000Z"),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate("2013-09-11T00:00:00.000Z"),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate("2013-01-24T00:00:00.000Z"),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate("2011-11-23T00:00:00.000Z"),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate("2011-03-10T00:00:00.000Z"),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    score: 0
  }
]

```

Figure 1.21: Output

## 1.17 Supprimer les restaurants où le score egale à 0

```
new_york> db.restaurants.deleteMany({ "grades.score": 0 })
{ acknowledged: true, deletedCount: 1258 }
new_york>
```

Figure 1.22: delete restaurant

## 1.18 Exécution du code avec commentaire

### 1.18.1 Exécution

```
new_york> mapFunction = function () { emit(this.borough, 1); }; reduceFunction = function (key, values) { return Array.sum(values); }; queryParam = {"query": {}, "out": {"inline": true}}; db.paris.mapReduce(mapFunction, reduceFunction, queryParam);
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.
See https://docs.mongodb.com/manual/core/map-reduce for details.
MongoServerError: 'inline' takes only numeric '1'
new_york>
```

Figure 1.23: Execution

### 1.18.2 Commentaire

On remarque que mongodb a affiché une erreur dans un paramètre inline {"inline": true}, alors que ce paramètre prend une valeur numérique au lieu d'une chaîne de caractères. Output : DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.

See <https://docs.mongodb.com/manual/core/map-reduce> for details.

**MongoServerError** : 'inline' takes only numeric '1'