

MongoDB. Home Task 1

Note: if you already have MongoDB installed, please, check that you are running the latest version – 4.0, because it's necessary to complete some of the tasks.

1. Install MongoDB

Follow installation guidelines for your OS at <https://docs.mongodb.com/manual/installation/#mongodb-community-edition>.

2. Import Restaurants Collection

2.1. Save *restaurants.json* on your PC

2.2. Run local instance of MongoDB and port for the instance run *mongod* without any parameters

2.3. Use *mongoimport* (it's in MongoDB installation folder) to import the collection to the database. Run local MongoDB on the default port the following command should create "restaurants" collection in "frontcamp" database *mongoimport --db frontcamp --collection restaurants --file*

2.4. Verify that collection was correctly imported. Verify that the number of the documents in the restaurants collection is 25359.

```
cmd Command Prompt - mongo
C:\Users\Nik_Gapev>mongoimport --db frontcamp --collection restaurants --file "C:\Users\Nik_Gapev\Desktop\restaurants.json"
2018-12-19T13:01:34.656+0300    connected to: localhost
2018-12-19T13:01:35.611+0300    imported 25359 documents

C:\Users\Nik_Gapev>mongo
MongoDB shell version v4.0.4
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("bb4751d9-3037-41ed-a685-a397584bf39e") }
MongoDB server version: 4.0.4
Server has startup warnings:
2018-12-19T11:02:30.012+0300 I CONTROL [initandlisten]
2018-12-19T11:02:30.012+0300 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-12-19T11:02:30.012+0300 I CONTROL [initandlisten] **           Read and write access to data and configuration is unrestricted.
2018-12-19T11:02:30.012+0300 I CONTROL [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use frontcamp
switched to db frontcamp
> db.restaurants.count()
25359
```

3. Querying Restaurants Collection

Answer the following questions:

3.1. How many "Chinese" (cuisine) restaurants are in "Queens" (borough)?

```
> db.restaurants.find({"cuisine": "Chinese", "borough": "Queens"}).count()
728
```

3.2. What is the _id of the restaurant which has the grade with the highest ever score?

```
> db.restaurants.find().sort({"grades.score": -1}).limit(1)
{ "_id": ObjectId("5c1a16ff451f38259a1a71e5"), "address": { "building": "65", "coord": [ -73.9782725, 40.7624822 ], "street": "West 54 Street", "zipcode": "10019" }, "borough": "Manhattan", "cuisine": "American", "grades": [ { "date": ISODate("2014-08-22T00:00:00Z"), "grade": "A", "score": 11 }, { "date": ISODate("2014-03-28T00:00:00Z"), "grade": "C", "score": 131 }, { "date": ISODate("2013-09-25T00:00:00Z"), "grade": "A", "score": 11 }, { "date": ISODate("2013-04-08T00:00:00Z"), "grade": "B", "score": 25 }, { "date": ISODate("2012-10-15T00:00:00Z"), "grade": "A", "score": 11 }, { "date": ISODate("2011-10-19T00:00:00Z"), "grade": "A", "score": 13 } ], "name": "Mumals On 54 / Randolphs'S", "restaurant_id": "48372466" }
```

3.3. Add a grade { grade: "A", score: 7, date: ISODate() } to every restaurant in "Manhattan" (borough).

```
> db.restaurants.updateMany({"borough": "Manhattan"}, {$push: {grades: {"grade": "A", "score": 7, "date": ISODate()}}})
{ "acknowledged": true, "matchedCount": 10259, "modifiedCount": 10259 }
```

3.4. What are the names of the restaurants which have a grade at index 8 with score less then 7? Use projection to include only names without _id.

```
> db.restaurants.distinct("name", {"grades.8.score": {$lt: 7}})
[ "Silver Krust West Indian Restaurant", "Pure Food" ]
```

3.5. What are _id and borough of "Seafood" (cuisine) restaurants which received at least one "B" grade in period from 2014-02-01 to 2014-03-01? Use projection to include only _id and borough.

```
> db.restaurants.find({"cuisine": "Seafood", "grades": {$elemMatch: {"grade": "B", "date": { $gte: ISODate("2014-02-01"), $lt: ISODate("2014-03-01") }}}}, {_id: 1, borough: 1})
{ "_id": ObjectId("5c1a16ff451f38259a1aa5e9"), "borough": "Bronx" }
{ "_id": ObjectId("5c1a16ff451f38259a1aa86a"), "borough": "Manhattan" }
```

4. Indexing Restaurants Collection

4.1. Create an index which will be used by this query and provide proof (from explain() or Compass UI) that the index is indeed used by the winning plan: `db.restaurants.find({ name: "Glorious Food" })`

```
    "numIndexesAfter" : 2,
    "ok" : 1
}
> db.restaurants.find({name: "Glorious Food"}).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "name" : {
        "$eq" : "Glorious Food"
      }
    },
    "winningPlan" : {
      "stage" : "FETCH",
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "name" : 1
        },
        "indexName" : "name_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "name" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : false,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "name" : [
            ["Glorious Food\","Glorious Food\"]
          ]
        }
      },
      "rejectedPlans" : [ ]
    },
    "serverInfo" : {
      "host" : "EPBYMINW5062",
      "port" : 27017,
      "version" : "4.0.4",
      "gitVersion" : "f288a3bdf201007f3693c58e140056adf8b04839"
    },
    "ok" : 1
  }
}
```

4.2. Drop index from task 4.1

```
> db.restaurants.dropIndex("name_1")
{
  "ok" : 0,
  "errmsg" : "index not found with name [name_1]",
  "code" : 27,
  "codeName" : "IndexNotFound"
}
> db.restaurants.dropIndex("name_1")
{ "nIndexesWas" : 2, "ok" : 1 }
> db.restaurants.find({name: "Glorious Food"}).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "name" : {
        "$eq" : "Glorious Food"
      }
    },
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "name" : {
          "$eq" : "Glorious Food"
        }
      },
      "direction" : "forward"
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW5062",
    "port" : 27017,
    "version" : "4.0.4",
    "gitVersion" : "f288a3bdf201007f3693c58e140056adf8b04839"
  },
  "ok" : 1
}
```

4.3. Create an index to make this query covered and provide proof (from explain() or Compass UI) that it is indeed covered: `db.restaurants.find({ restaurant_id: "41098650" }, { _id: 0, borough: 1 })`

```
> db.restaurants.createIndex({restaurant_id: 1})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
> db.restaurants.find({restaurant_id: "41098650"}, {_id: 0, borough: 1}).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "restaurant_id" : {
        "$eq" : "41098650"
      }
    },
    "winningPlan" : {
      "stage" : "PROJECTION",
      "transformBy" : {
        "_id" : 0,
        "borough" : 1
      },
      "inputStage" : {
        "stage" : "FETCH",
        "inputStage" : {
          "stage" : "IXSCAN",
          "keyPattern" : {
            "restaurant_id" : 1
          },
          "indexName" : "restaurant_id_1",
          "isMultiKey" : false,
          "multiKeyPaths" : {
            "restaurant_id" : [ ]
          },
          "isUnique" : false,
          "isSparse" : false,
          "isPartial" : false,
          "indexVersion" : 2,
          "direction" : "forward",
          "indexBounds" : {
            "restaurant_id" : [
              "[\\"41098650\\", \\"41098650\\"]"
            ]
          }
        }
      }
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW5062",
    "port" : 27017,
    "version" : "4.0.4",
    "gitVersion" : "f288a3bdf201007f3693c58e140056adf8b04839"
  },
  "ok" : 1
}
```

4.4. Create a partial index on cuisine field which will be used only when filtering on borough equal to "Staten Island": `db.restaurants.find({ borough: "Staten Island", cuisine: "American" })` – uses index `db.restaurants.find({ borough: "Staten Island", name: "Bagel Land" })` – does not use index `db.restaurants.find({ borough: "Queens", cuisine: "Pizza" })` – does not use index

```
> db.restaurants.createIndex({cuisine: 1}, {partialFilterExpression: {borough: "Staten Island"}})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
>
```

4.5. Create an index to make query from task 3.4 covered and provide proof (from explain() or Compass UI) that it is indeed covered 5.

```
> db.restaurants.find({borough: "Staten Island", cuisine: "American"}).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Staten Island"
          }
        },
        {
          "cuisine" : {
            "$eq" : "American"
          }
        }
      ]
    },
    "winningPlan" : {
      "stage" : "FETCH",
      "filter" : {
        "borough" : {
          "$eq" : "Staten Island"
        }
      },
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "cuisine" : 1
        },
        "indexName" : "cuisine_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "cuisine" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : true,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "cuisine" : [
            ["American", "American"]
          ]
        }
      }
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW5062",
    "port" : 27017,
    "version" : "4.0.4",
    "gitVersion" : "f288a3bdf201007f3693c58e140056adf8b04839"
  },
  "ok" : 1
}
```