**MongoDB. Home Task 1**

Note: if you already have MongoDB installed, please, check that you are running the latest version – 3.6, because it’s necessary to complete some of the tasks

**1.0 Install MongoDB Follow installation guidelines for your OS at** https://docs.mongodb.com/manual/installation/#mongodb-communityedition

Note: you can skip installing MongoDB as a service; you can install MongoDB Compass 2. Import Restaurants Collection Follow these steps to import restaurants collection to you local data base:

1.1 Save restaurants.json on your PC

1.2 Run local instance of MongoDB  Assuming you want to use default data directory and port for the instance run mongod without any parameters

**mongod**

1.3 Use mongoimport (it’s in MongoDB installation folder) to import the collection to the database  Assuming you run local MongoDB on the default port the following command should create “restaurants” collection in “frontcamp” database

**mongoimport --db frontcamp --collection restaurants --file <path to restaurants.json>**

1.4 Verify that collection was correctly imported  Assuming local MongoDB instance uses the default port, run mongo without any parameters

**mongo**

* Switch to frontcamp database

**use frontcamp**

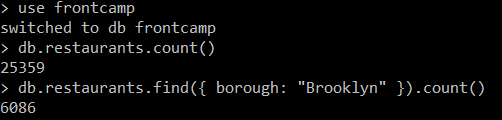
* Verify that the number of the documents in the restaurants collection is 25359

**db.restaurants.count()**

Note: for tasks 3 and 4 create a txt or doc report for your mentor and include both query and its result for every subtask, e.g.:

2.1 Query: db.restaurants.find({ borough: "Brooklyn" }).count()

**Result: 6086**



Note: please perform all tasks in the specified order, because the results may depend on the previous operations 3. Querying Restaurants Collection Note: please use mongo shell for this task

Have a look at a document from the collection to get familiar with the schema:

db.restaurants.findOne()

Answer the following questions and include both query and the result (if applicable) into your report:

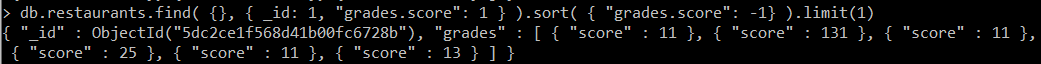
**3.1 How many “Chinese” (cuisine) restaurants are in “Queens” (borough)?**

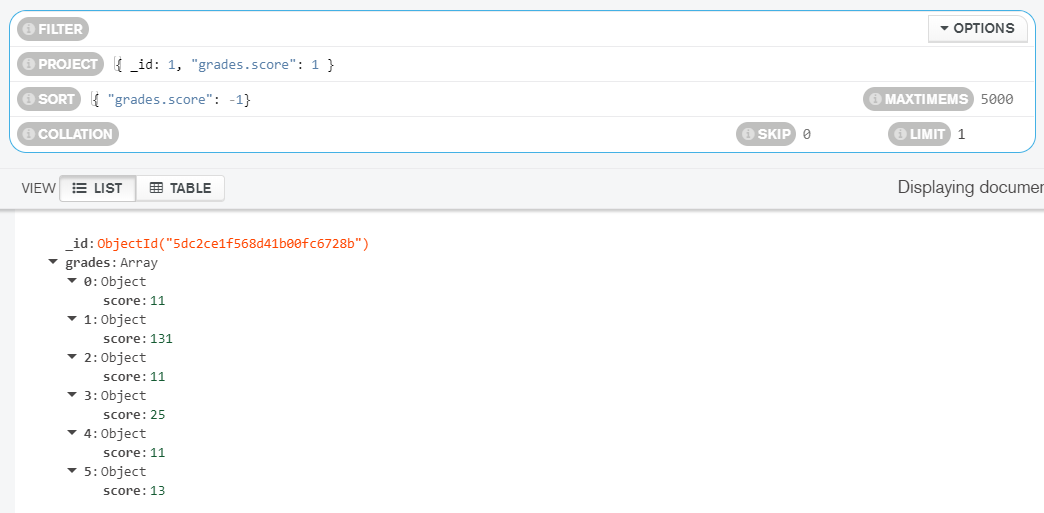
db.restaurants.find( { borough: "Queens", cuisine: "Chinese" }).count()



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

db.restaurants.find( {}, { \_id: 1, "grades.score": 1 } ).sort( { "grades.score": -1} ).limit(1)

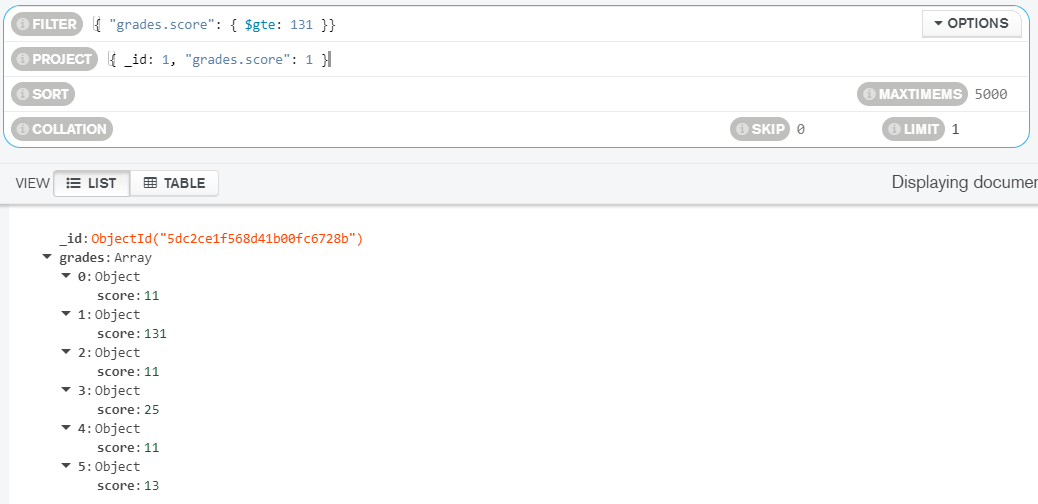




\***Check**\*

db.restaurants.find( { "grades.score": { $gte: 131 } }, { \_id: 1, "grades.score": 1 })





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

db.restaurants.updateMany( { borough: "Manhattan" }, { $push: { grades: { "date": ISODate(), "grade": "A", "score": 7 }}} )



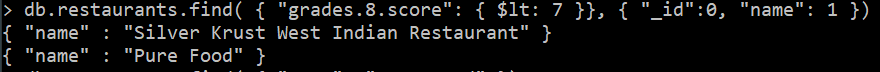
\***Check**\*

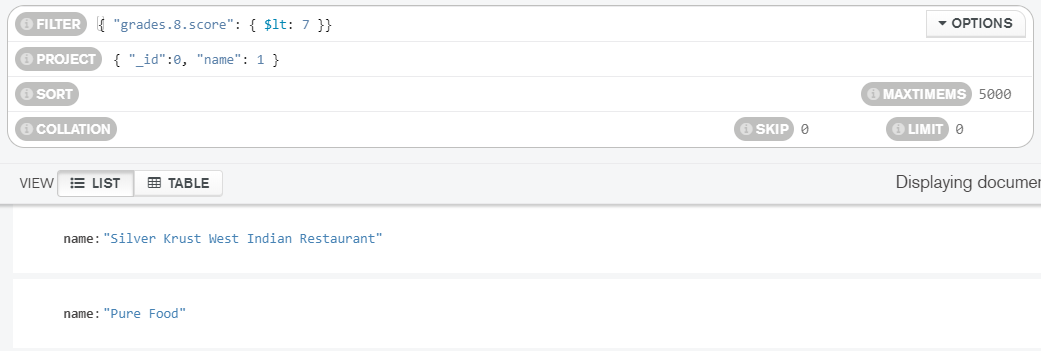
db.restaurants.find( { borough: "Manhattan" } ).count()



**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.**

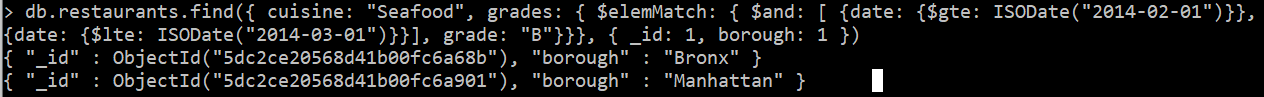
db.restaurants.find( { "grades.8.score": { $lt: 7 }}, { "\_id":0, "name": 1 })

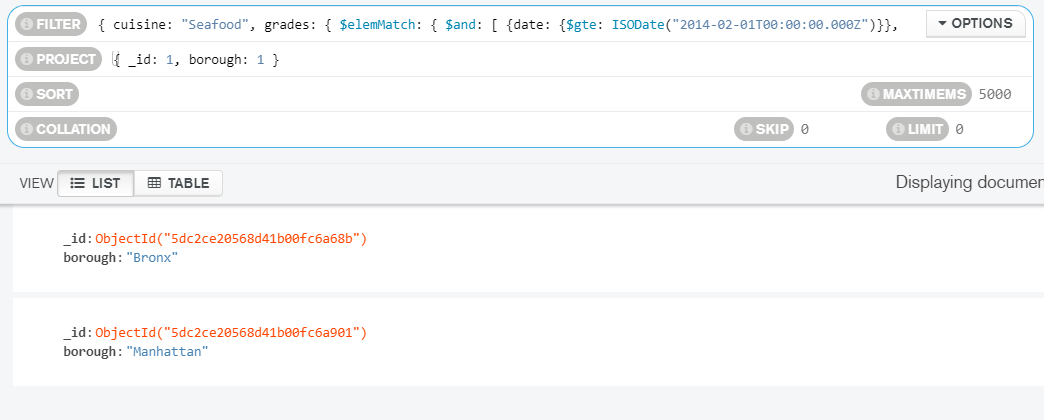




**3.5 What are \_id and borough of “Seafood” (cuisine) restaurants which received at least one “B” grade in period from 2014-02-01 to 2014-03-01? Use projection to include only \_id and borough.**

db.restaurants.find({ cuisine: "Seafood", grades: { $elemMatch: { $and: [ {date: {$gte: ISODate("2014-02-01")}}, {date: {$lte: ISODate("2014-03-01")}}], grade: "B"}}}, { \_id: 1, borough: 1 })



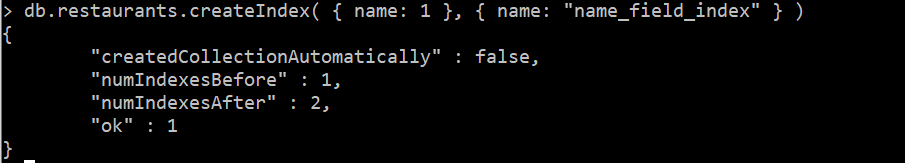


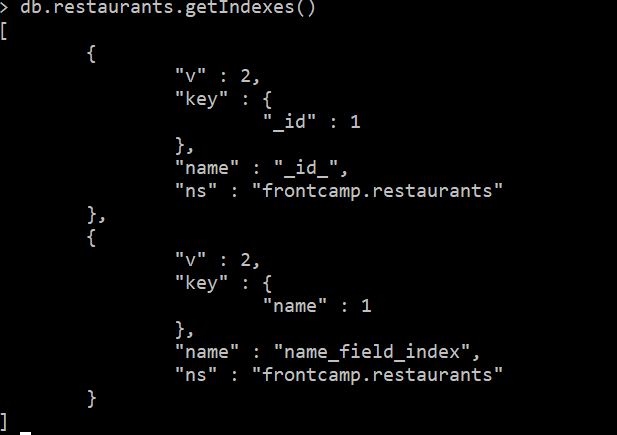
4.0 Indexing Restaurants Collection Note: you may use MongoDB Compass for this task if you want to

Create the following indexes:

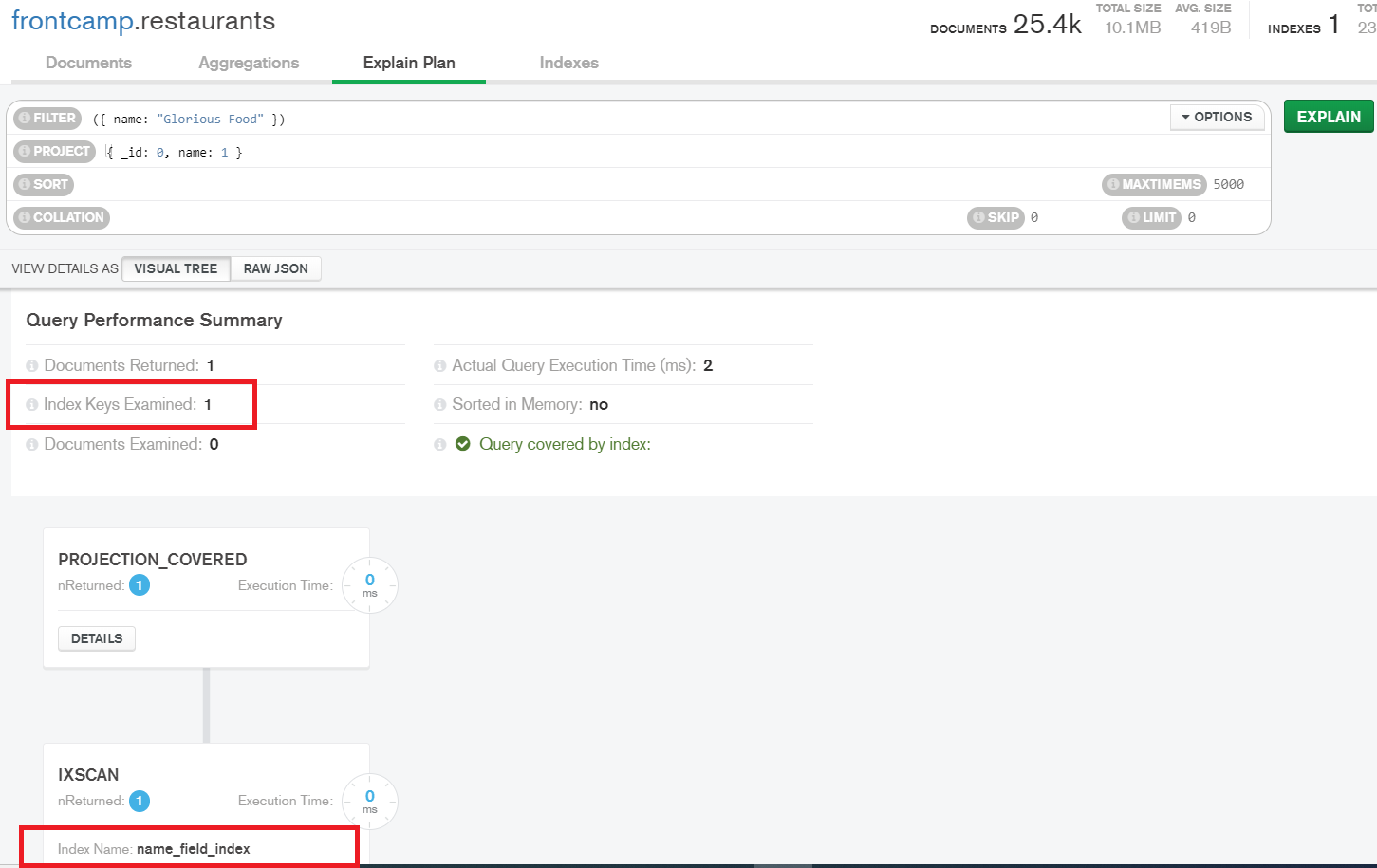
**4.1 Create an index which will be used by this query and provide proof (from explain() or Compass UI) that the index is indeed used by the winning plan:**

db.restaurants.createIndex( { name: 1 }, { name: "name\_field\_index" } )



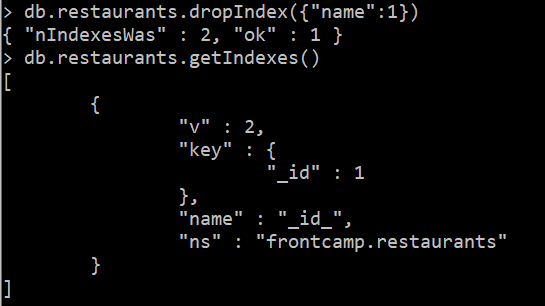


db.restaurants.find({ name: "Glorious Food" })



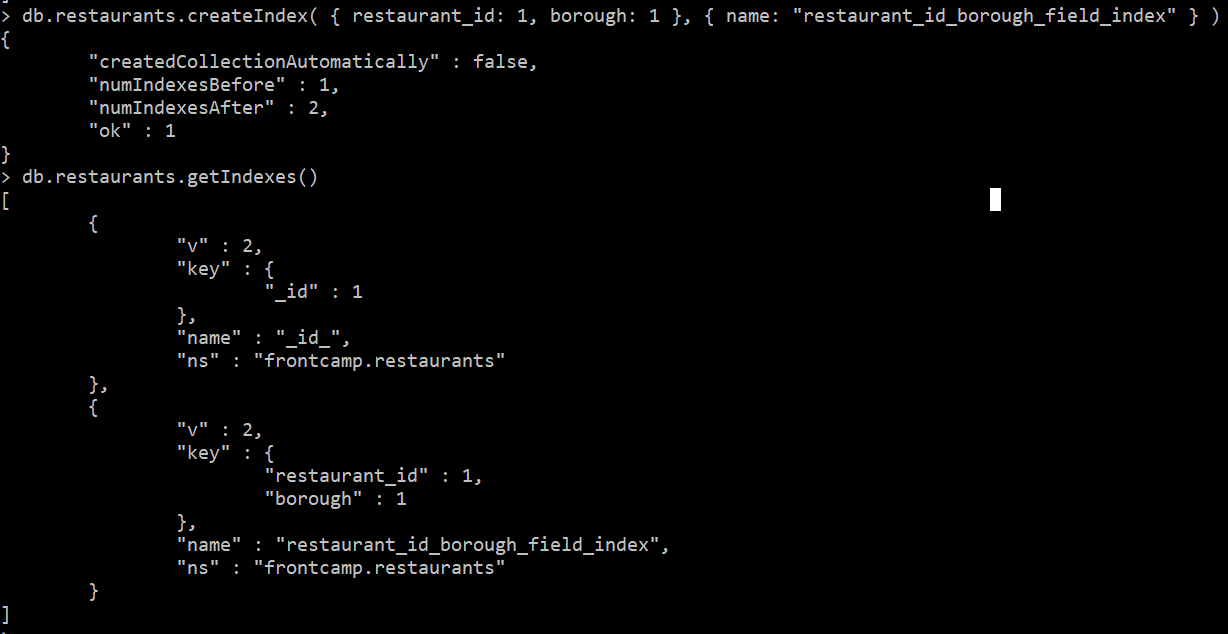
**4.2 Drop index from task**

db.restaurants.dropIndex({"name":1})

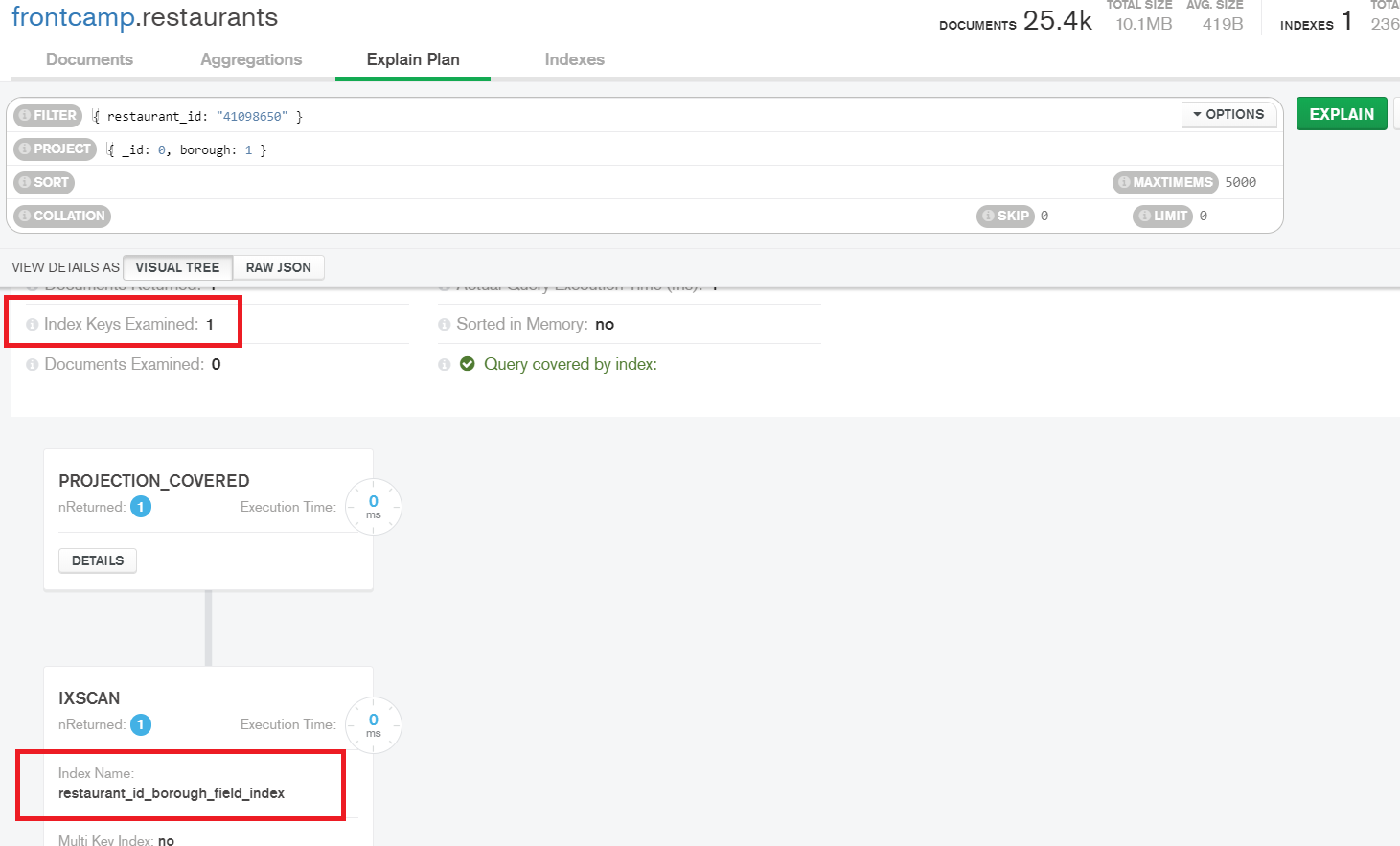


**4.3 Create an index to make this query covered and provide proof (from explain() or Compass UI) that it is indeed covered:**

db.restaurants.createIndex( { restaurant\_id: 1, borough: 1 }, { name: "restaurant\_id\_borough\_field\_index" } )



db.restaurants.find({ restaurant\_id: "41098650" }, { \_id: 0, borough: 1 })

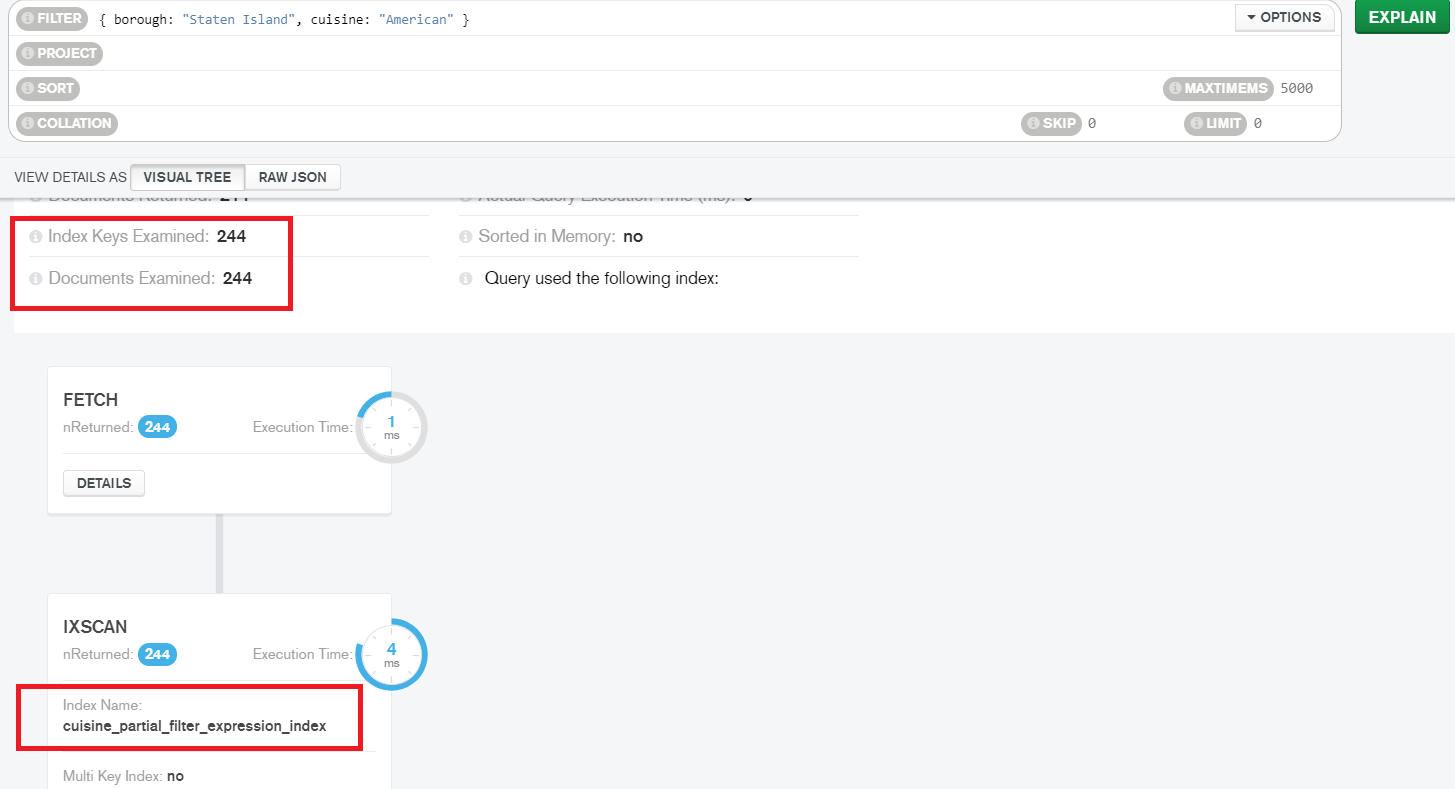


4.4 Create a partial index on cuisine field which will be used only when filtering on borough equal to “Staten Island”:

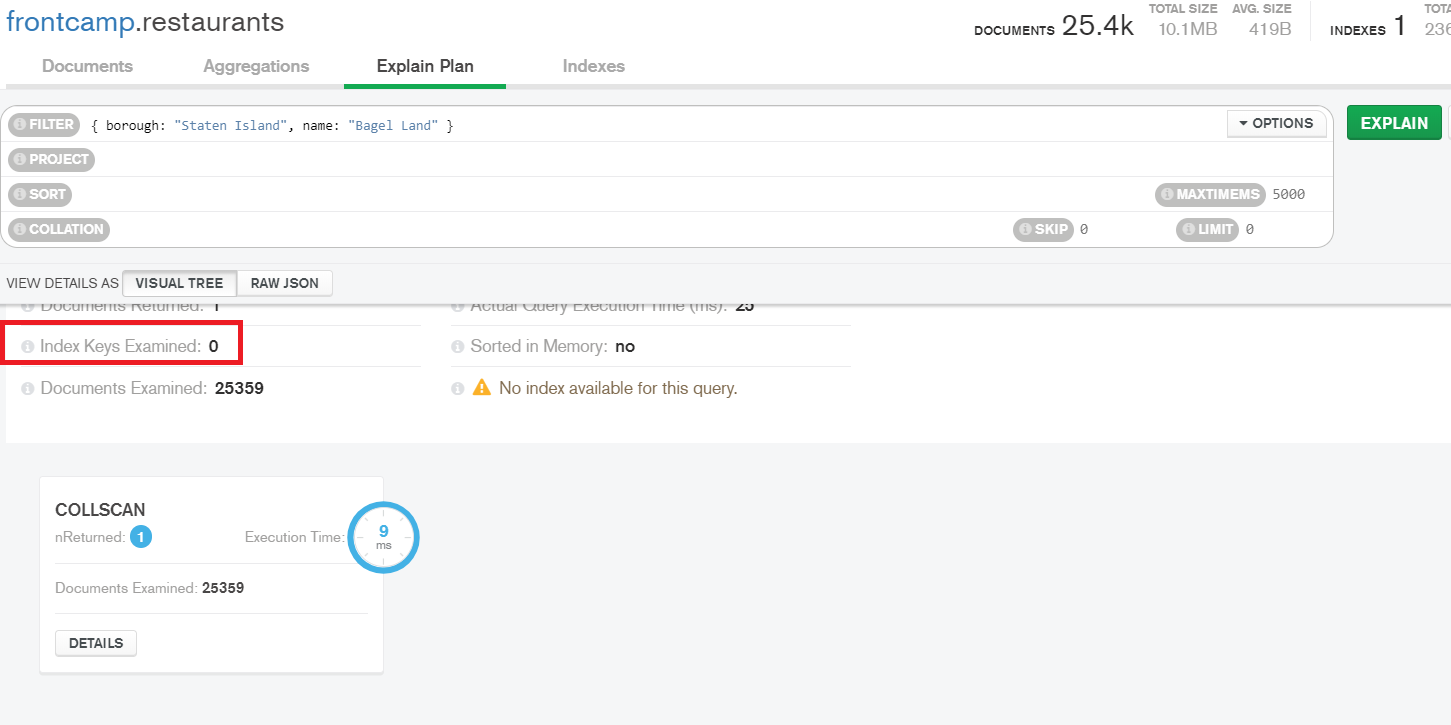
db.restaurants.createIndex({ cuisine: 1 }, { partialFilterExpression:{ "borough":"Staten Island" }, name: "cuisine\_partial\_filter\_expression\_index"} )



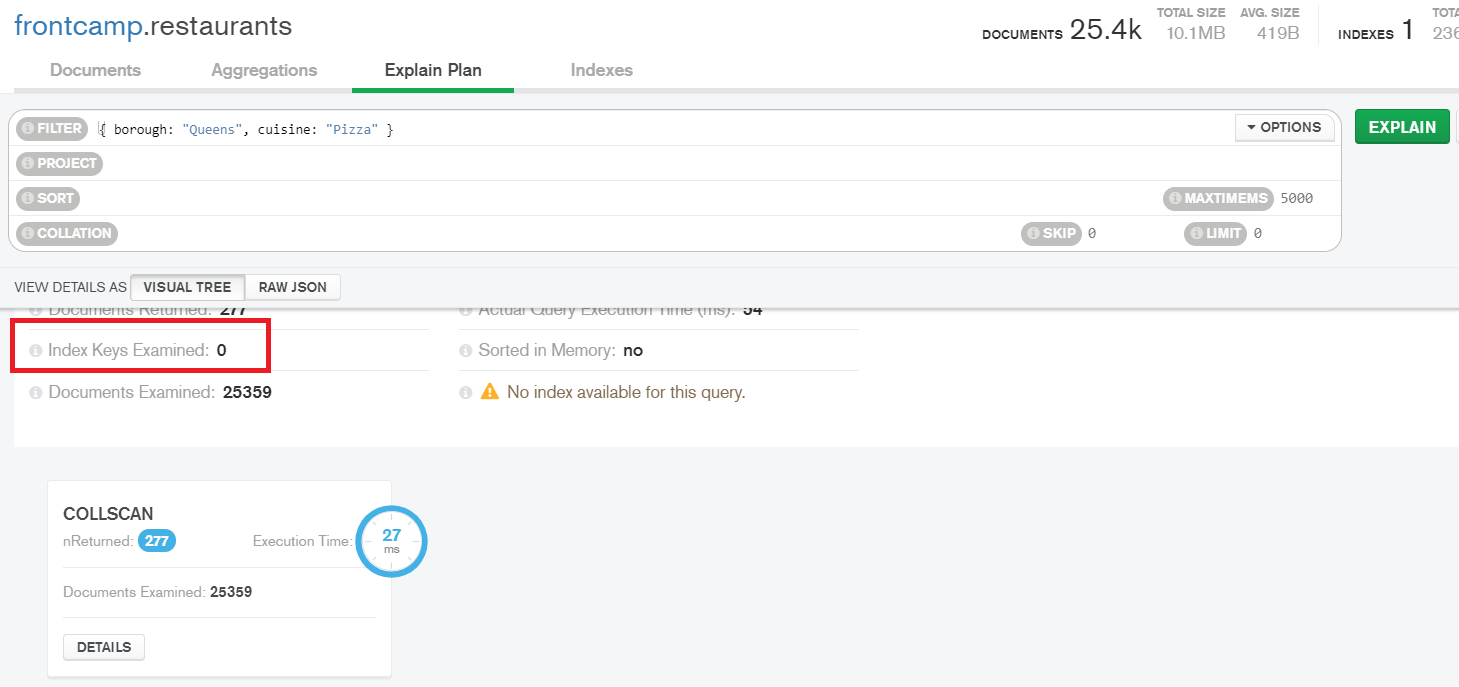
db.restaurants.find({ borough: "Staten Island", cuisine: "American" }) – uses index



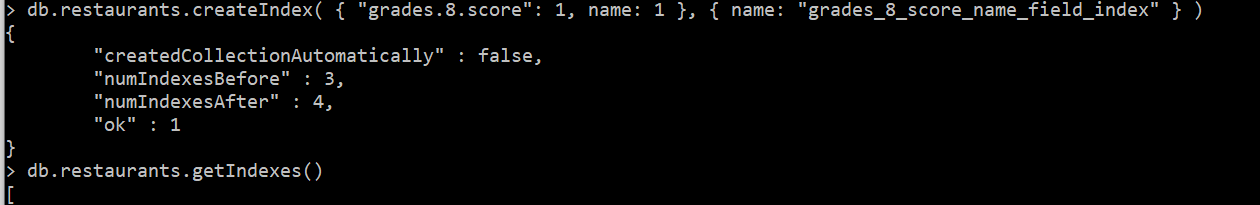
db.restaurants.find({ borough: "Staten Island", name: "Bagel Land" }) – does not use index

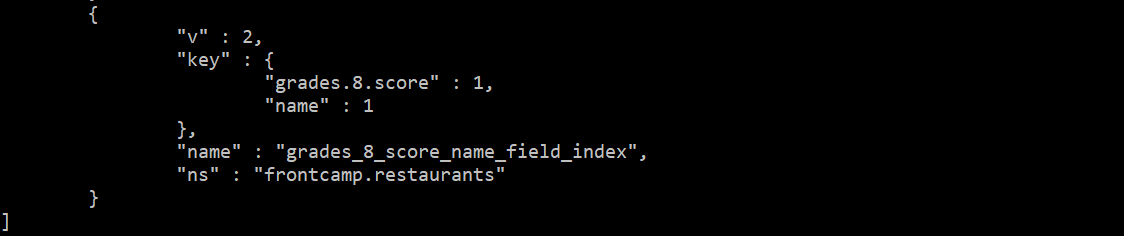


db.restaurants.find({ borough: "Queens", cuisine: "Pizza" }) – does not use index



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





db.restaurants.find( { "grades.8.score": { $lt: 7 }}, { "\_id":0, "name": 1 })

