# Querying Restaurants Collection

## Task 1

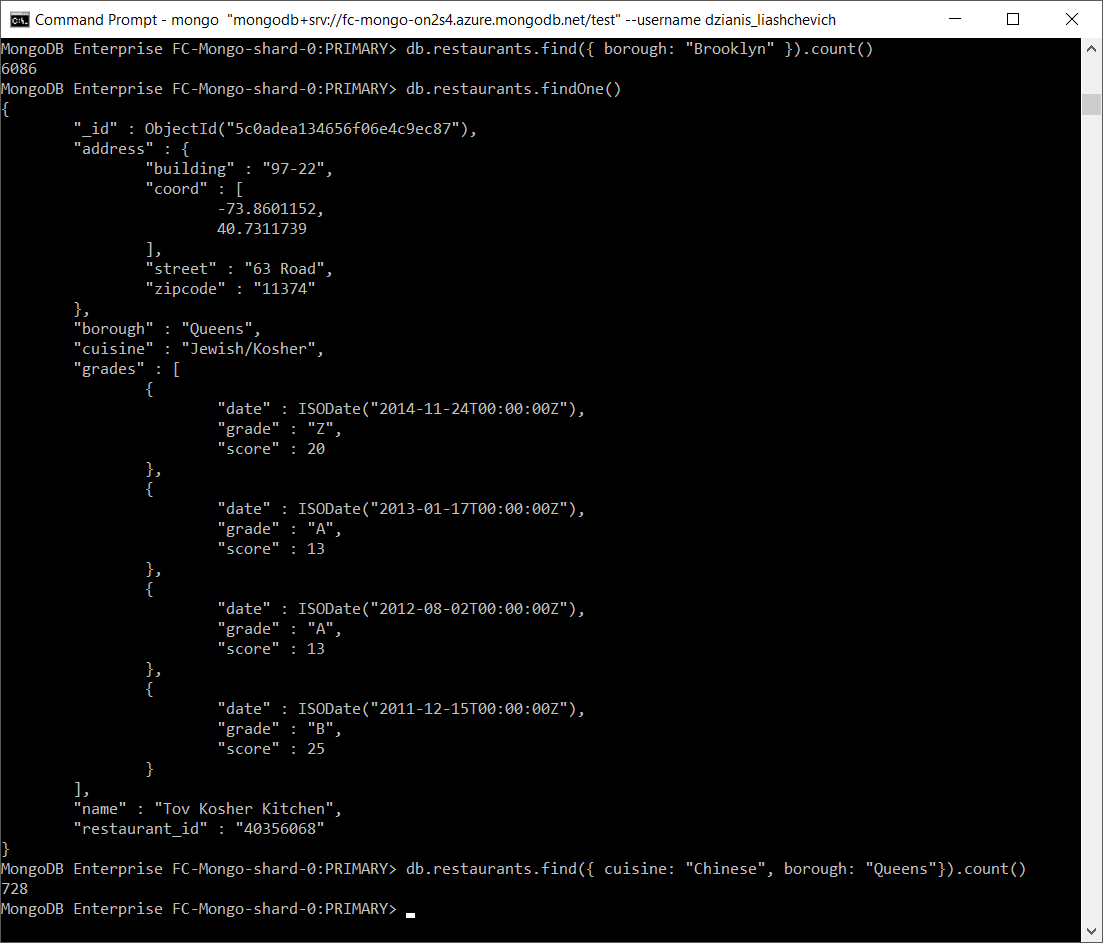
### Task:

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

### Query:

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

### Result:



## Task 2

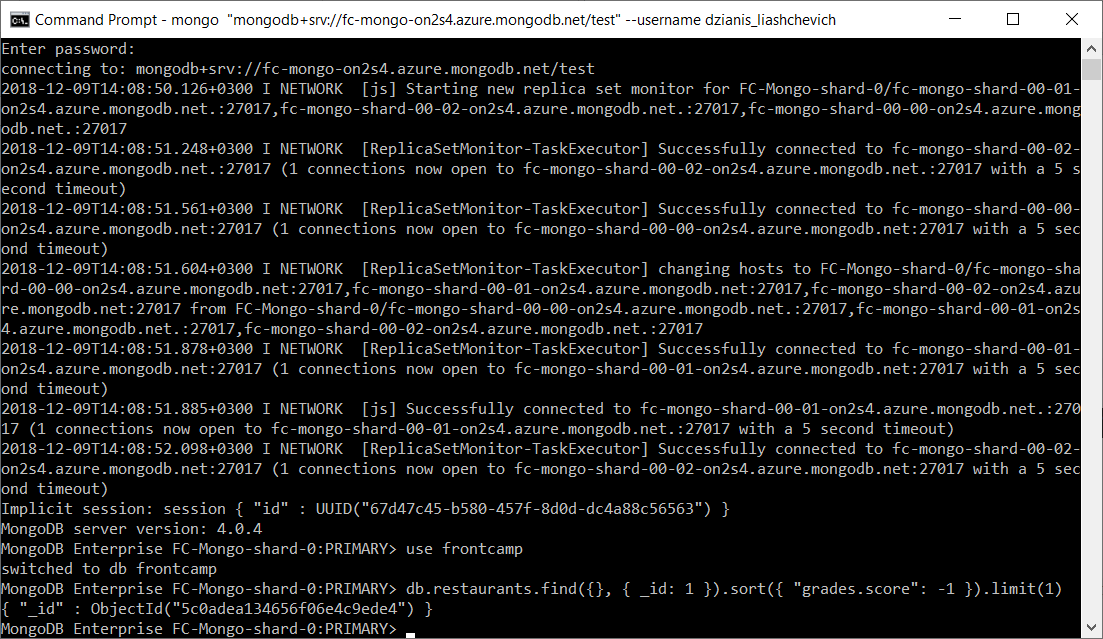
### Task:

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

### Query:

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

### Result:



## Task 3

### Task:

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

### Query:

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

### Result:



## Task 4

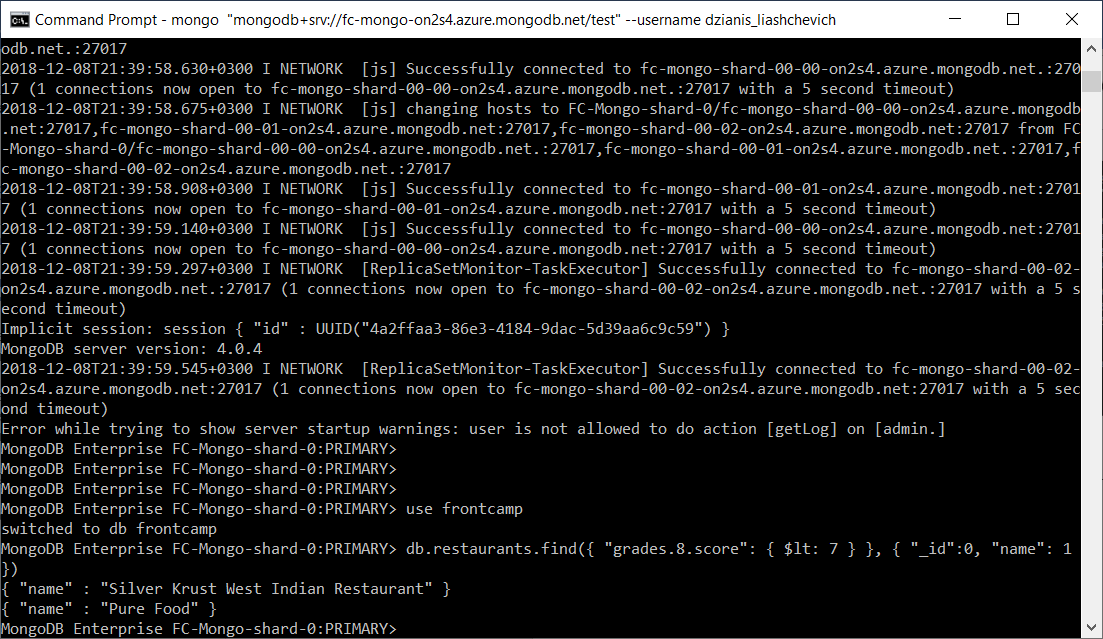
### Task:

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.

### Query:

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

### Result:



## Task 5

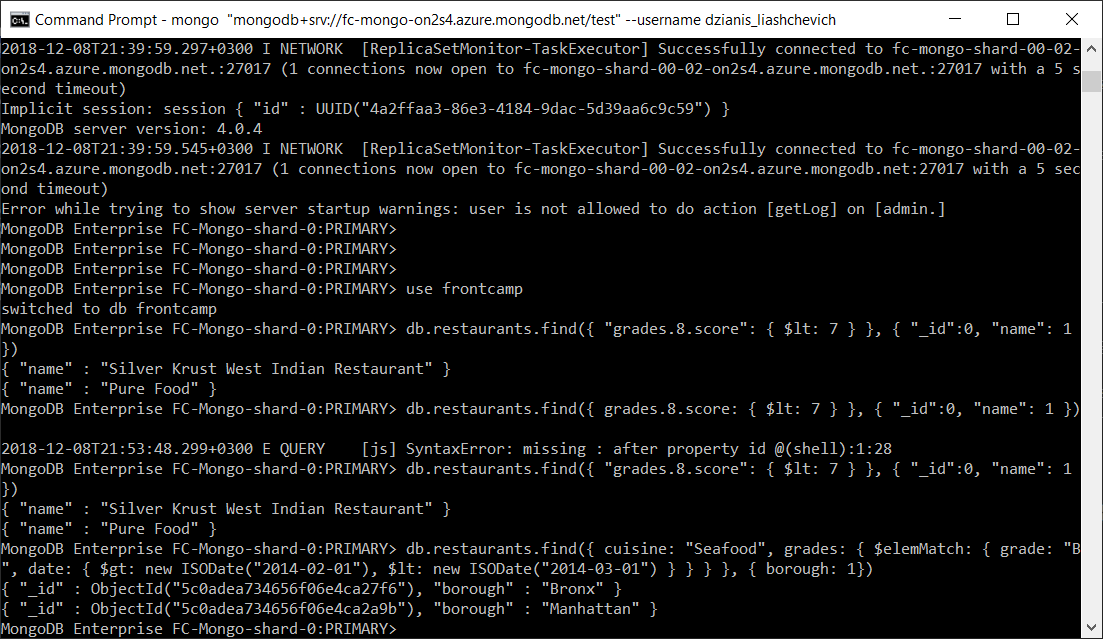
### Task:

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.

### Query:

db.restaurants.**find**({ cuisine: "Seafood", grades: { $elemMatch: { grade: "B", date: { $gt: new **ISODate**("2014-02-01"), $lt: new **ISODate**("2014-03-01") } } } }, { borough: 1 })

### Result:



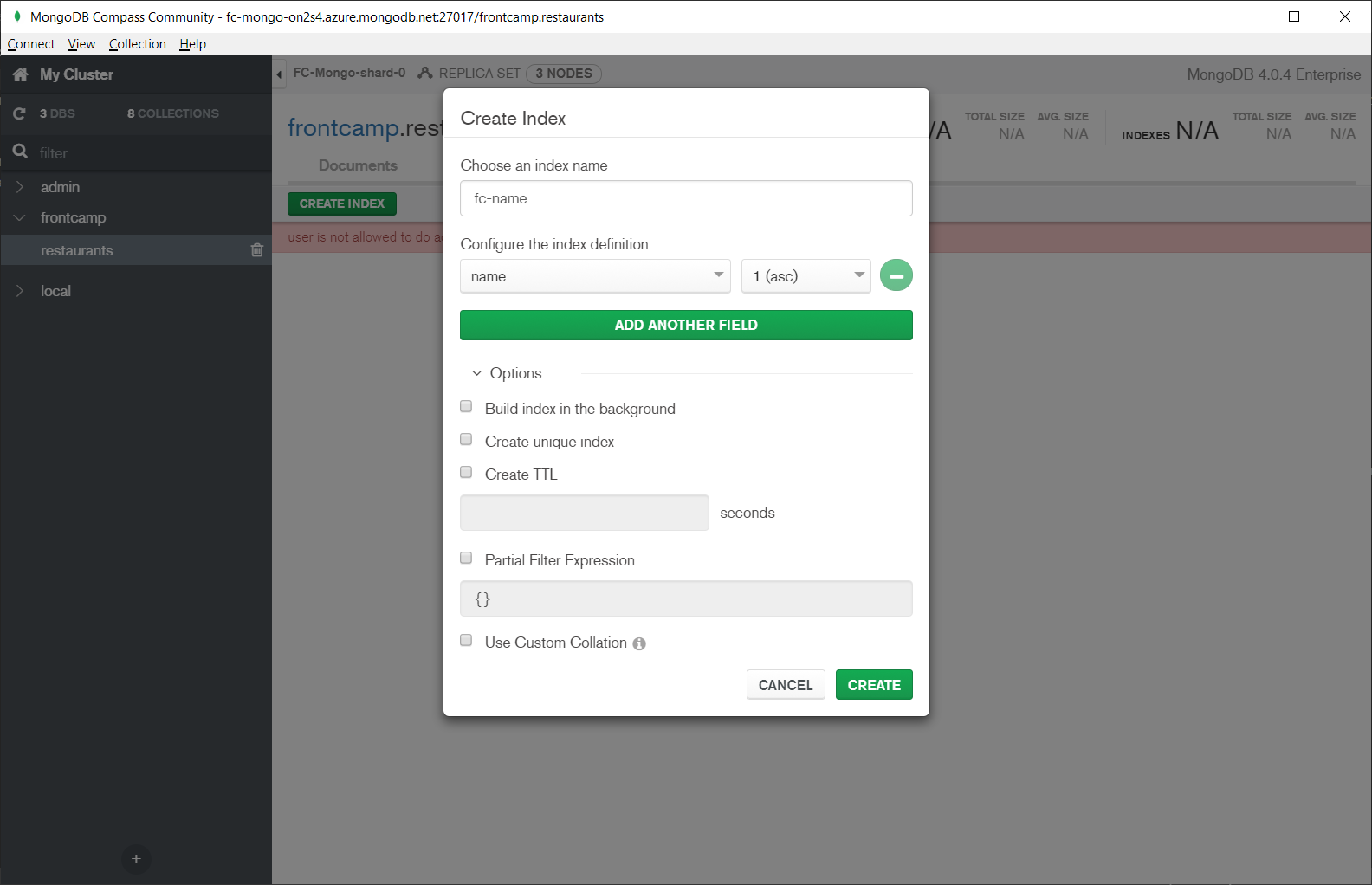
# Indexing Restaurants Collection

## Task 1

### Task:

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" })

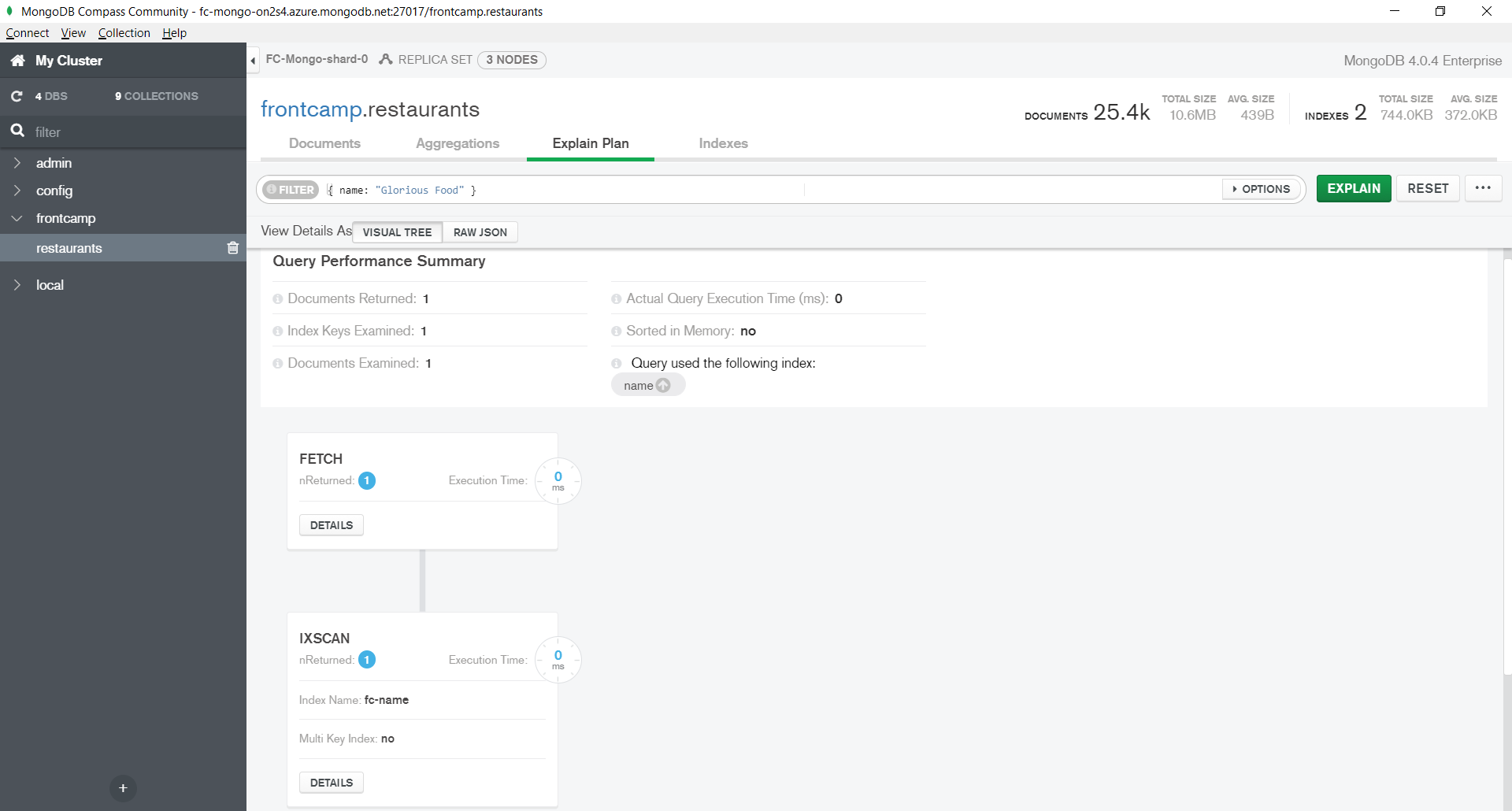
### Index:



### Index query:

db.restaurants.**createIndex**({ name: 1 }, { name: "fc-name" })

### Result:



## Task 2

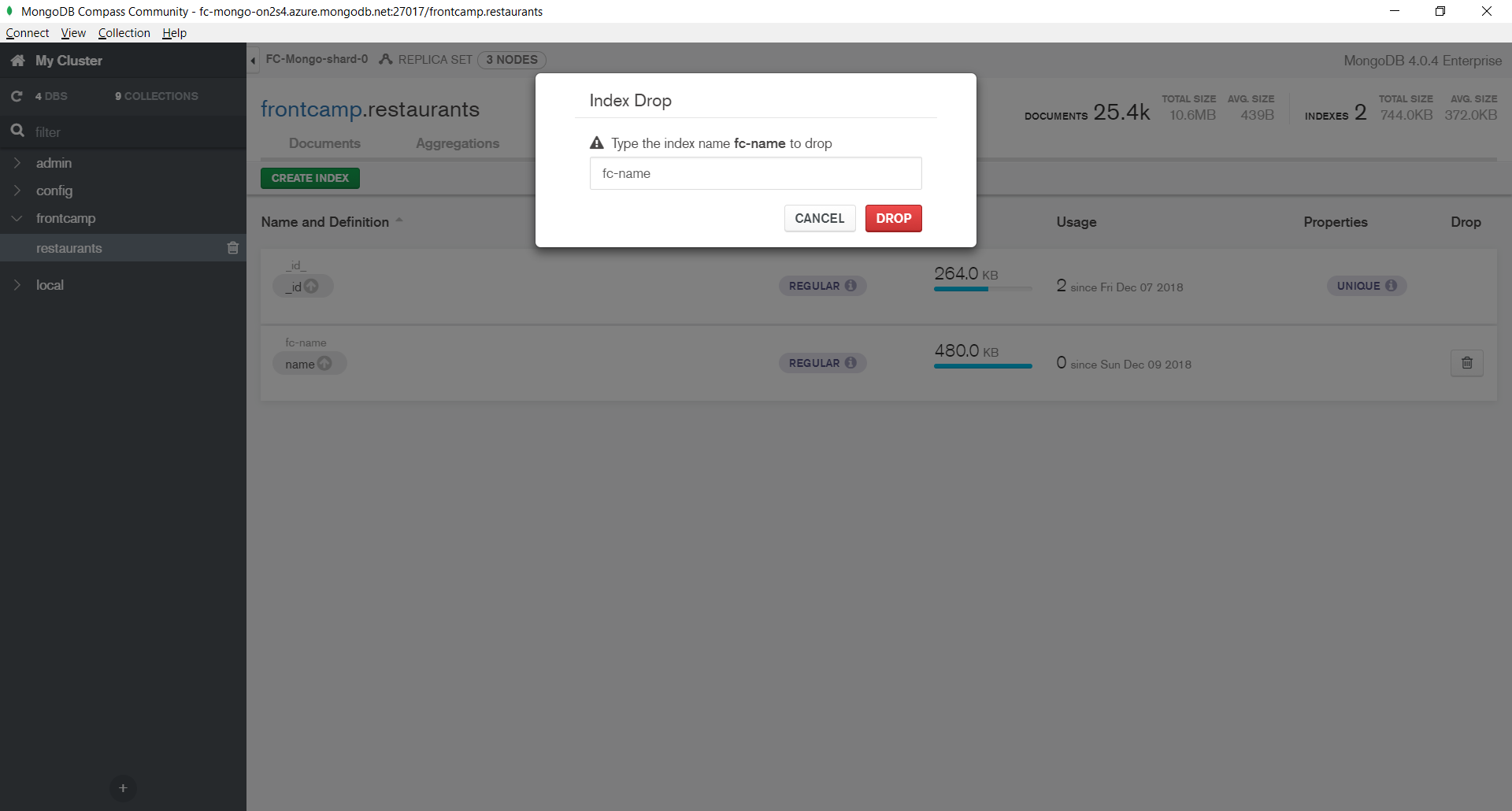
### Task:

Drop index from task 4.1

### Query:

db.restaurants.**dropIndex**("fc-name")

### Result:

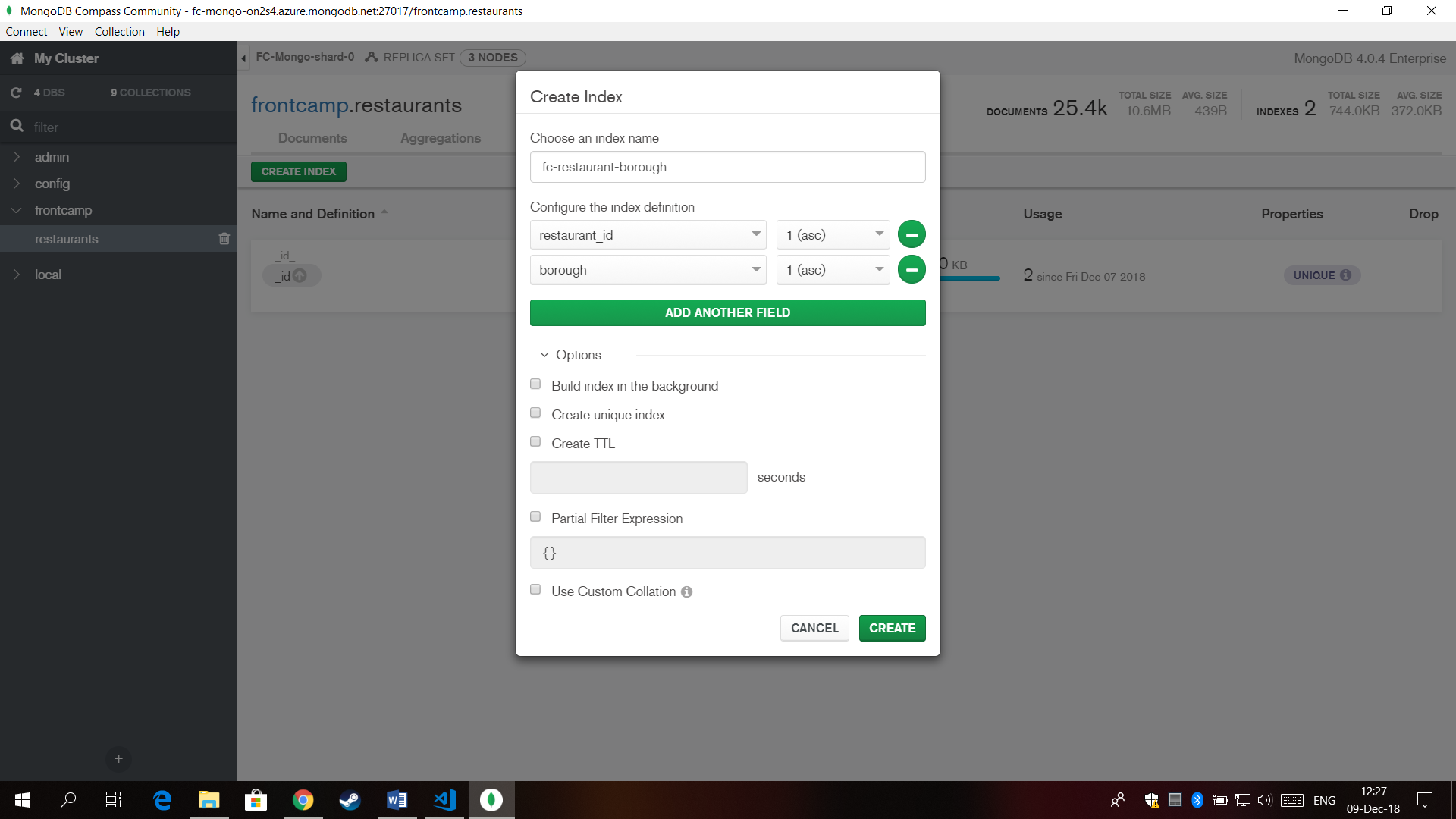


## Task 3

### Task:

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 })

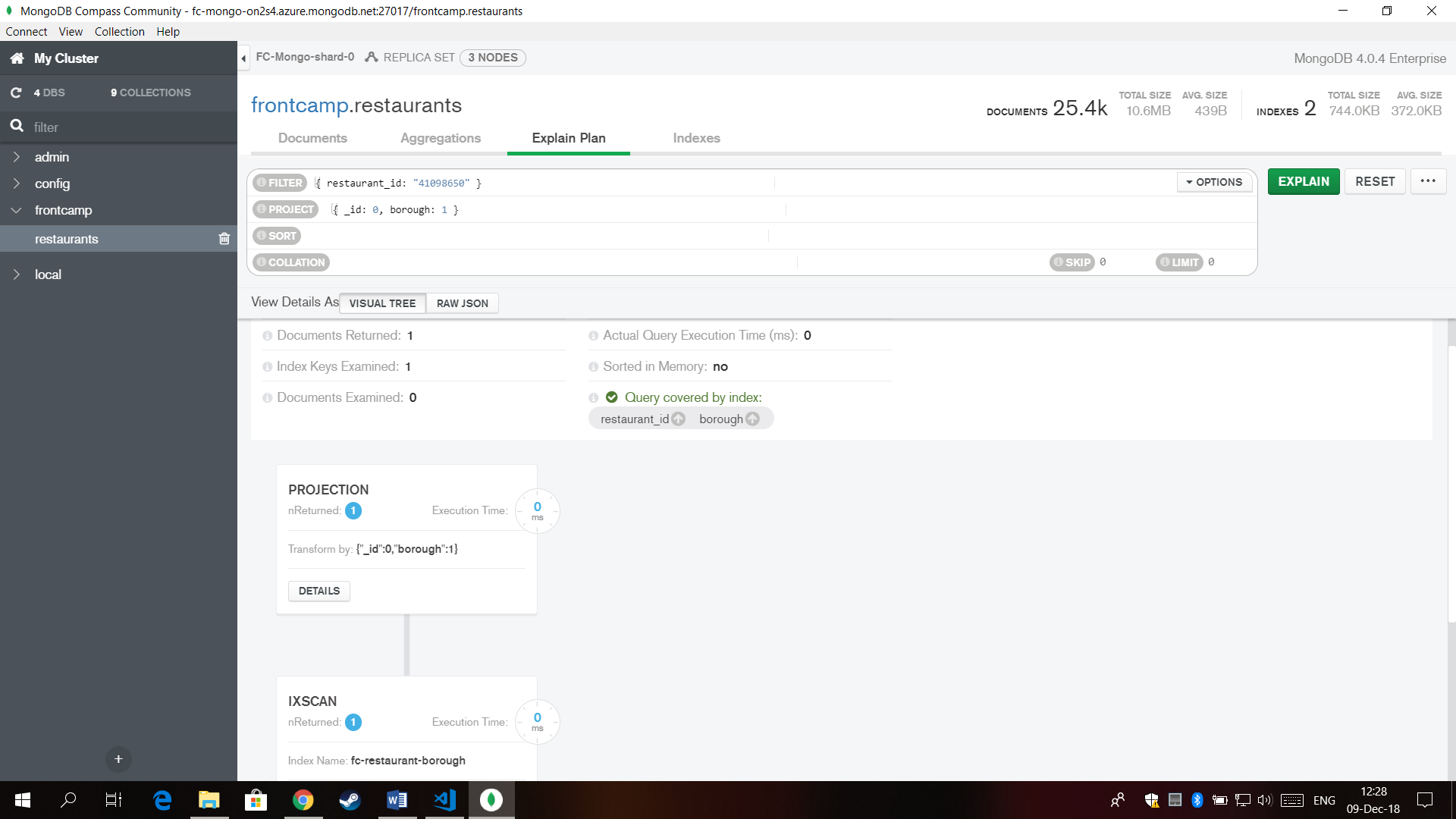
### Index:



### Index query:

db.restaurants.**createIndex**({ restaurant\_id: 1, borough: 1 }, { name: "fc-restaurant-borough" })

### Result:



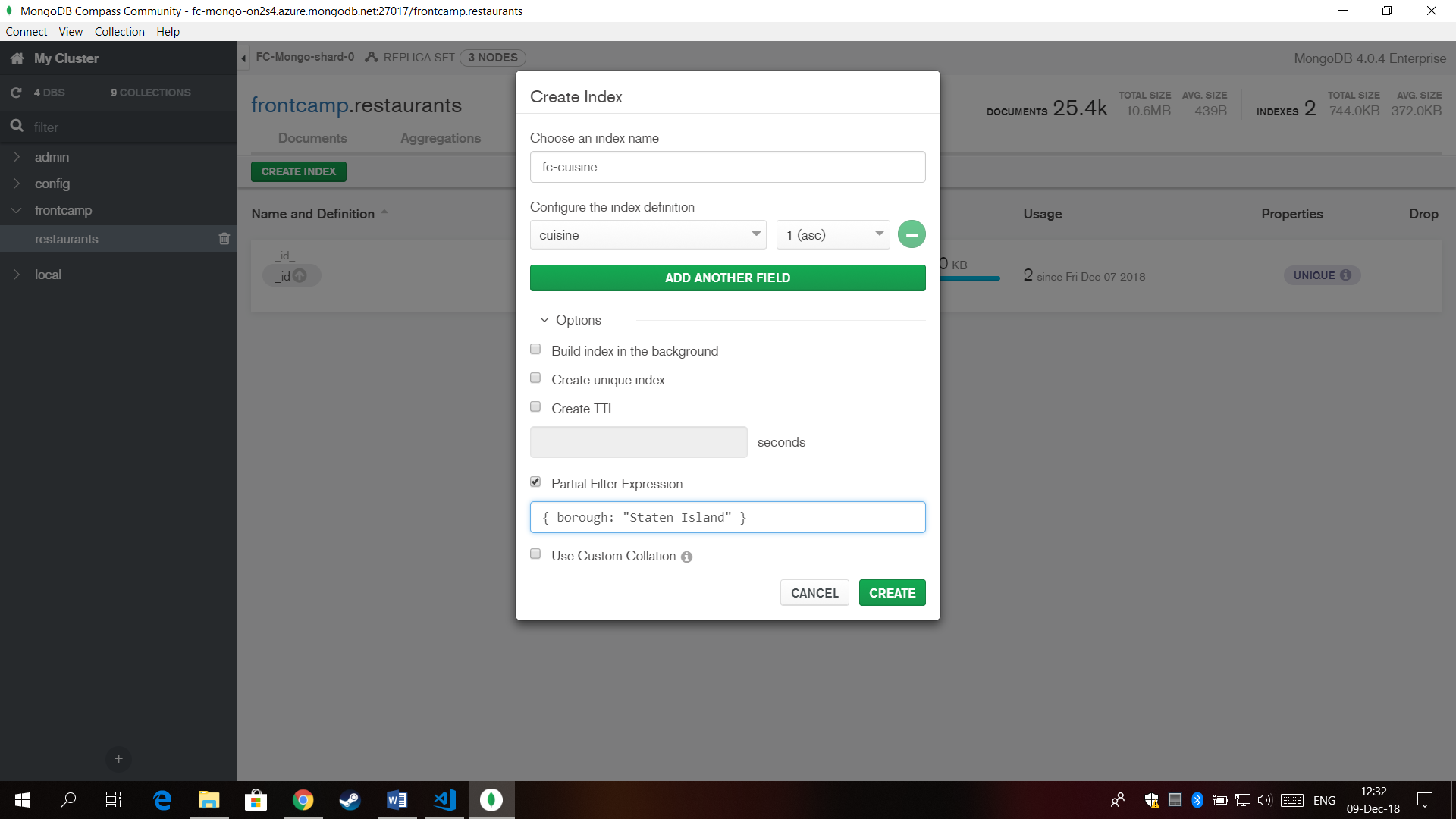
## Task 4

### Task:

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

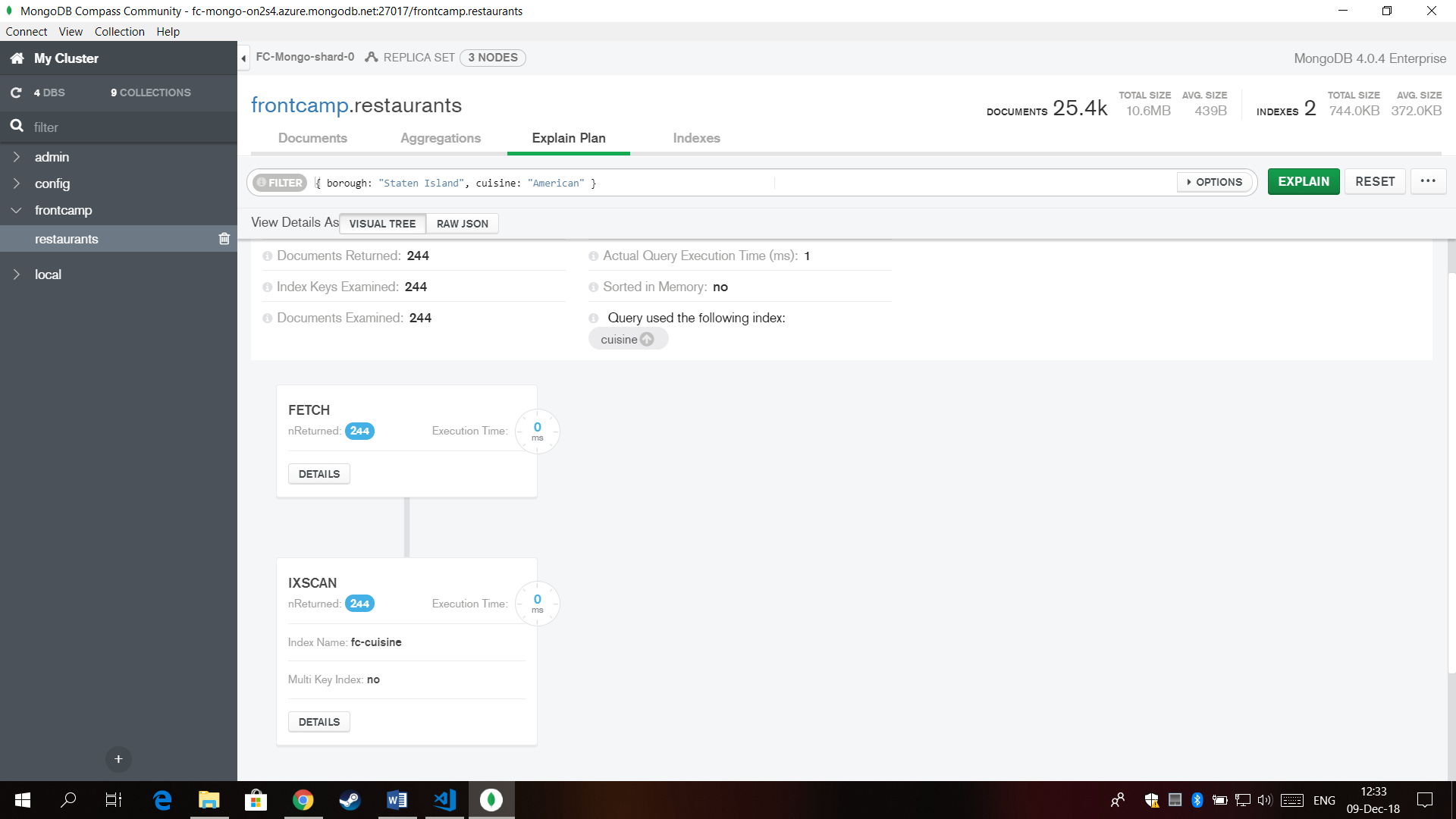
### Index:

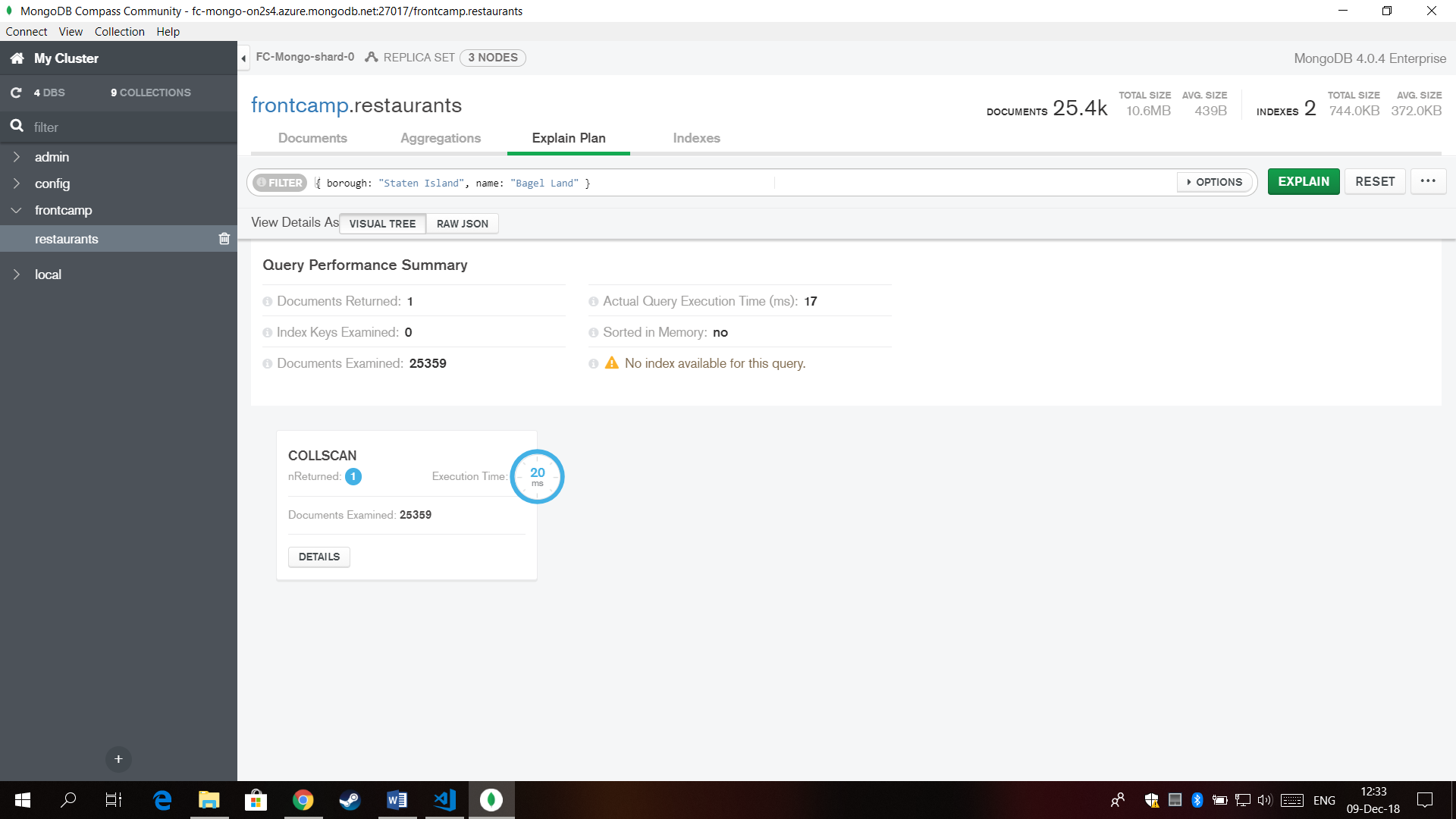


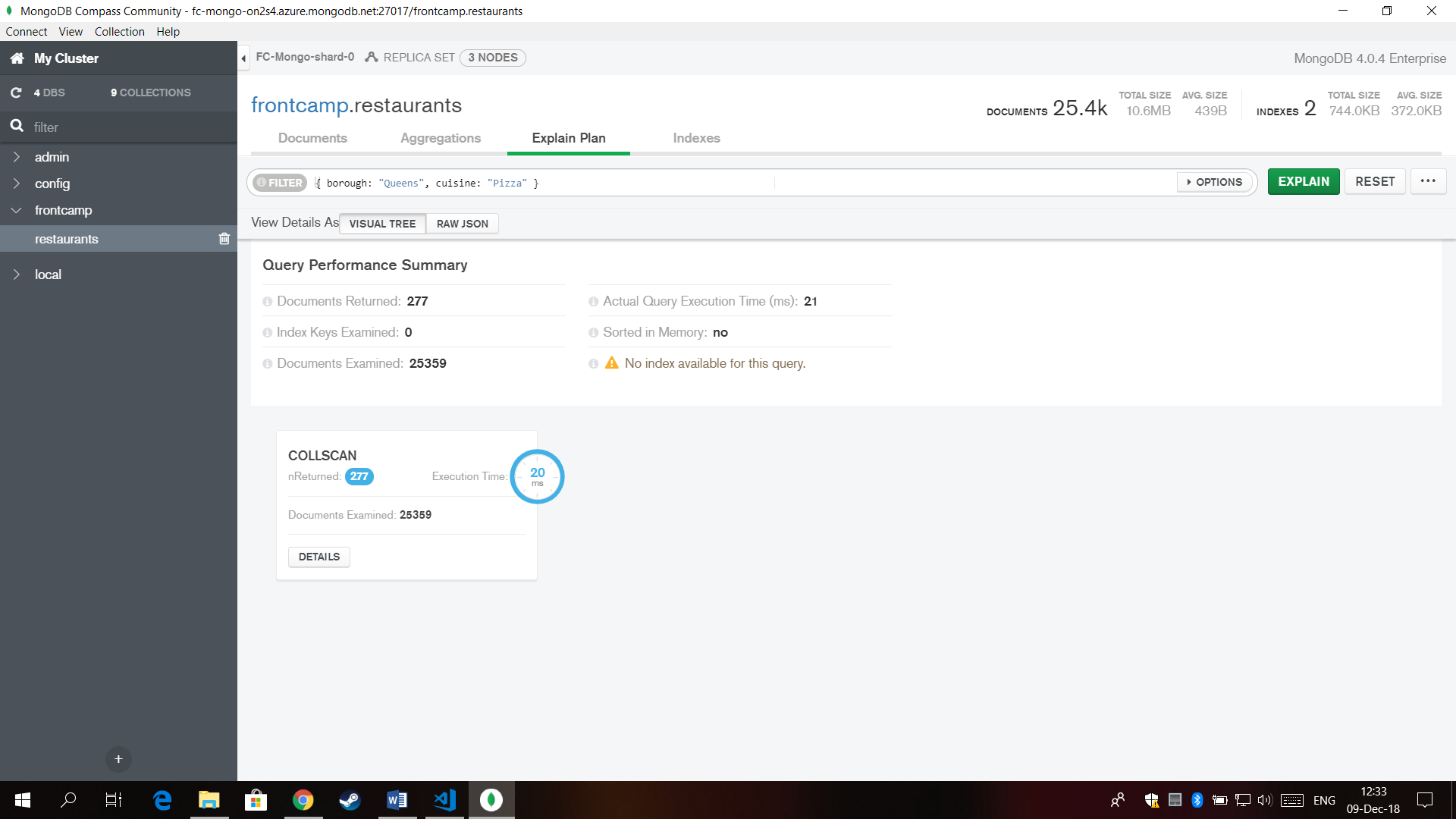
### Index query:

db.restaurants.**createIndex**({ cuisine: 1 }, { name: "fc-cuisine", partialFilterExpression: { borough: "Staten Island" } })

### Result:





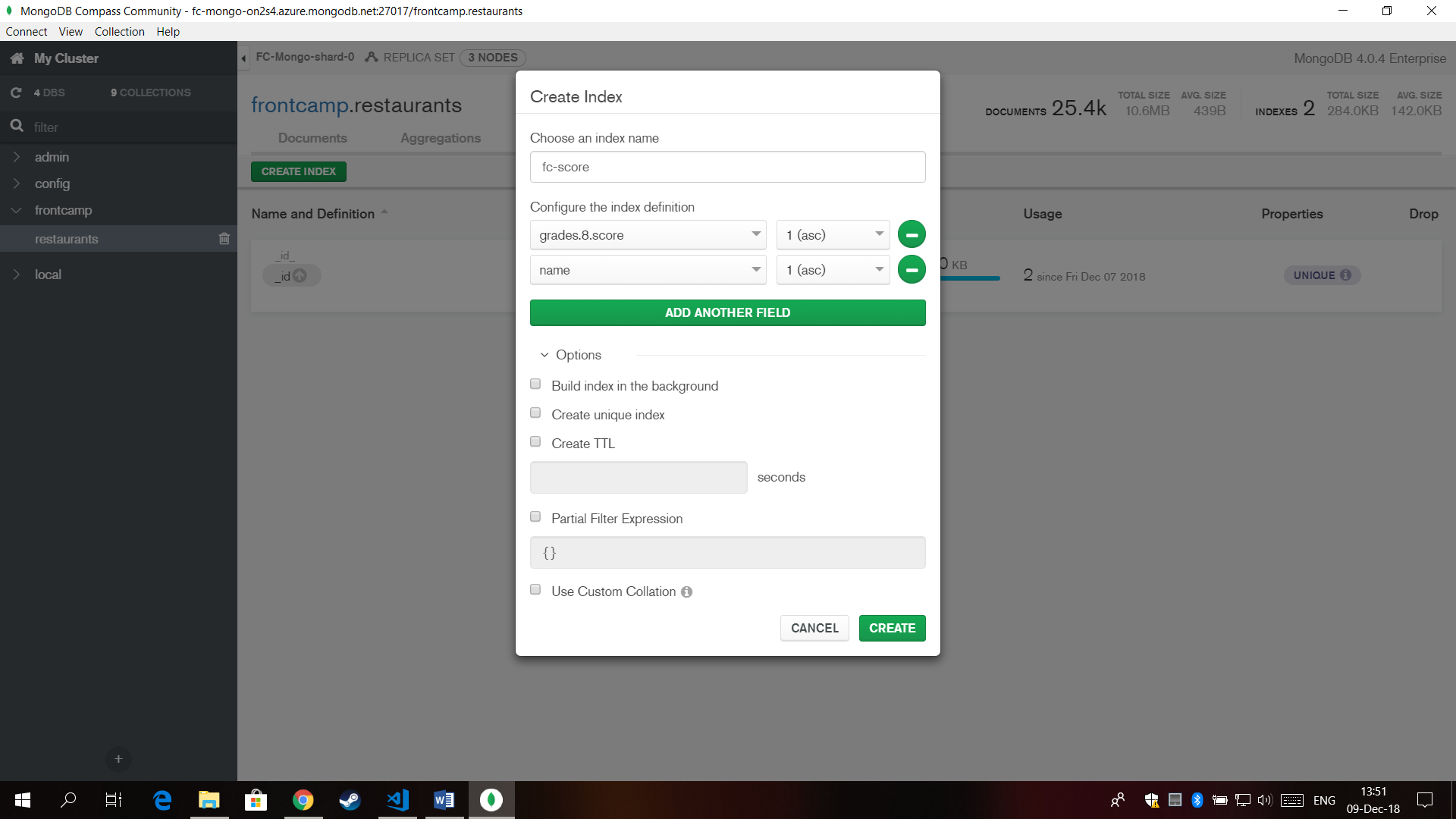


## Task 5

### Task:

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

### Index:



### Index query:

db.restaurants.**createIndex**({ "grades.8.score": 1, name: 1 }, { name: "fc-score" })

### Result:

