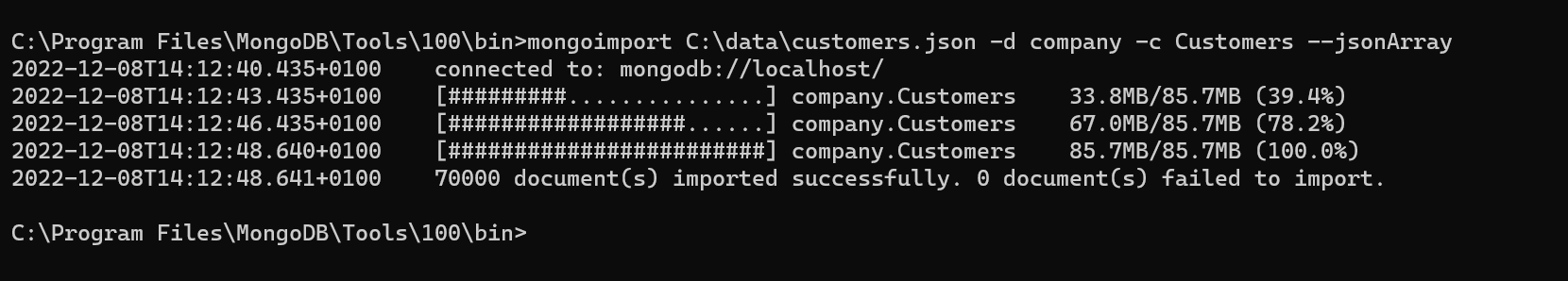
Assignment #7 - MongoDB Part 3

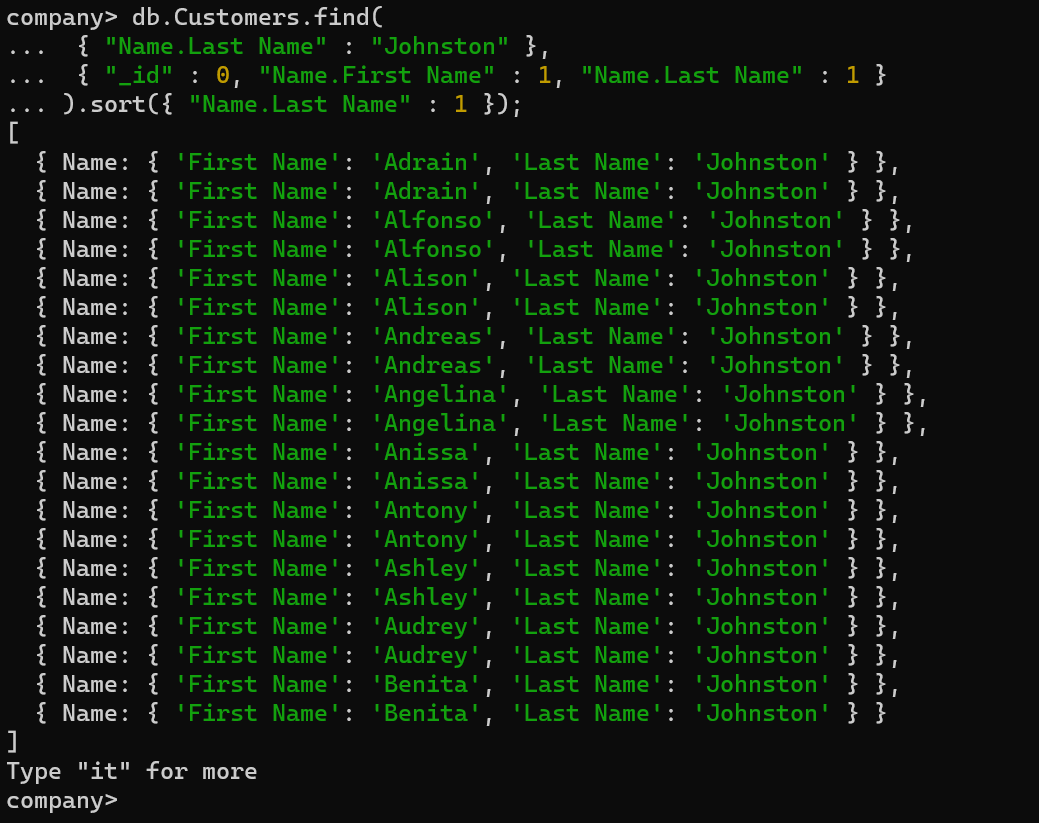
Complete the assignments that are included in MongoDB Part 3 file by submitting a text document with all required commands (you can include screenshots of the result of each task as well).

Exercise #1

Exit the mongo shell and import the data again, but this time into the Customers collection

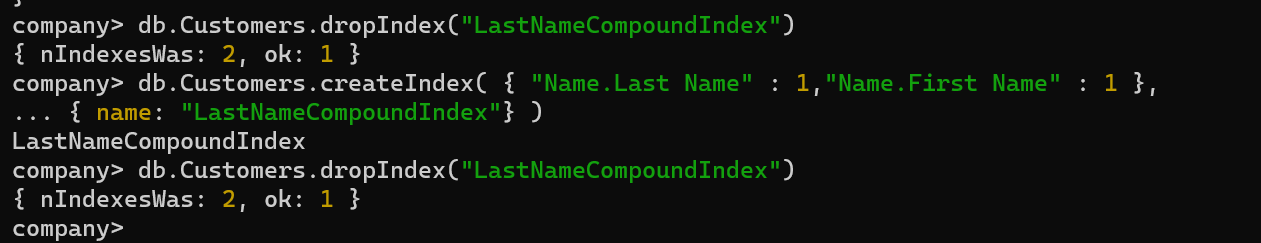


mongoimport C:\data\customers.json -d company -c Customers --jsonArray



Exercise #2

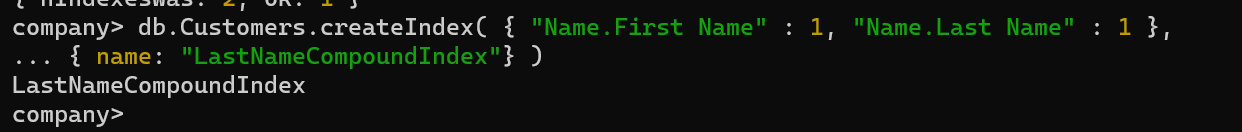
- Drop the LastNameCompoundIndex index



db.Customers.createIndex( { "Name.Last Name" : 1,"Name.First Name" : 1 },{ name: "LastNameCompoundIndex"} )

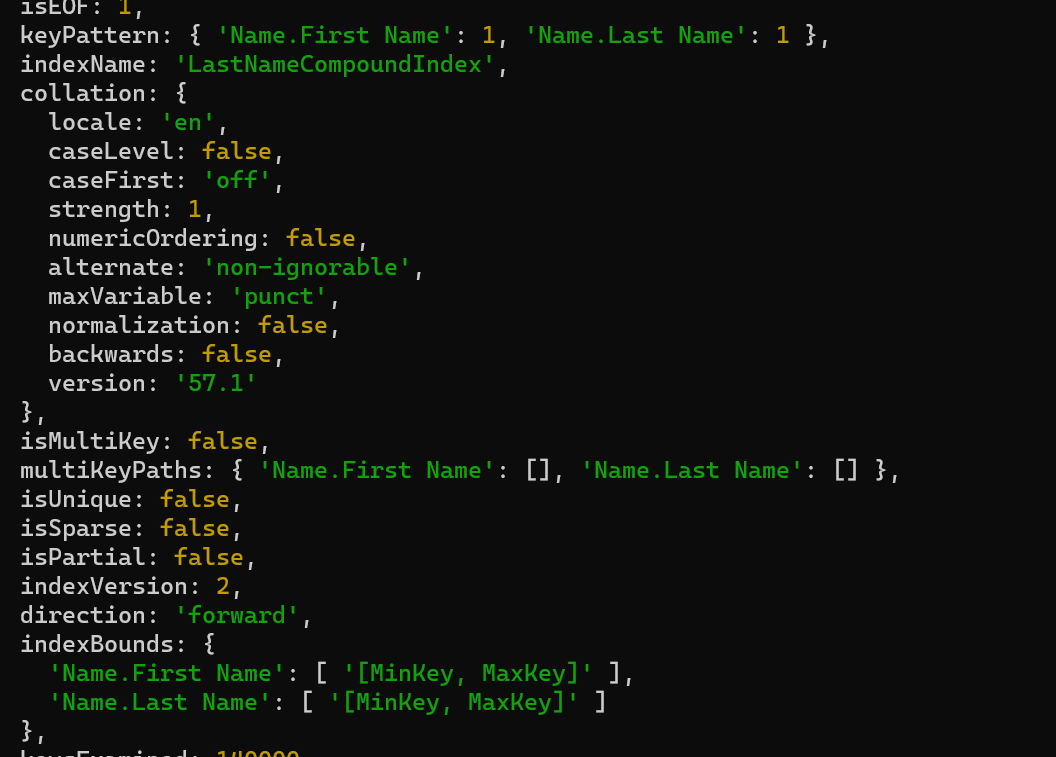
db.Customers.dropIndex("LastNameCompoundIndex")

- Change the order of the two fields in the index so that the Name.First Name comes before Name.Last Name



db.Customers.createIndex( { "Name.First Name" : 1, "Name.Last Name" : 1 },{ name: "LastNameCompoundIndex"} )

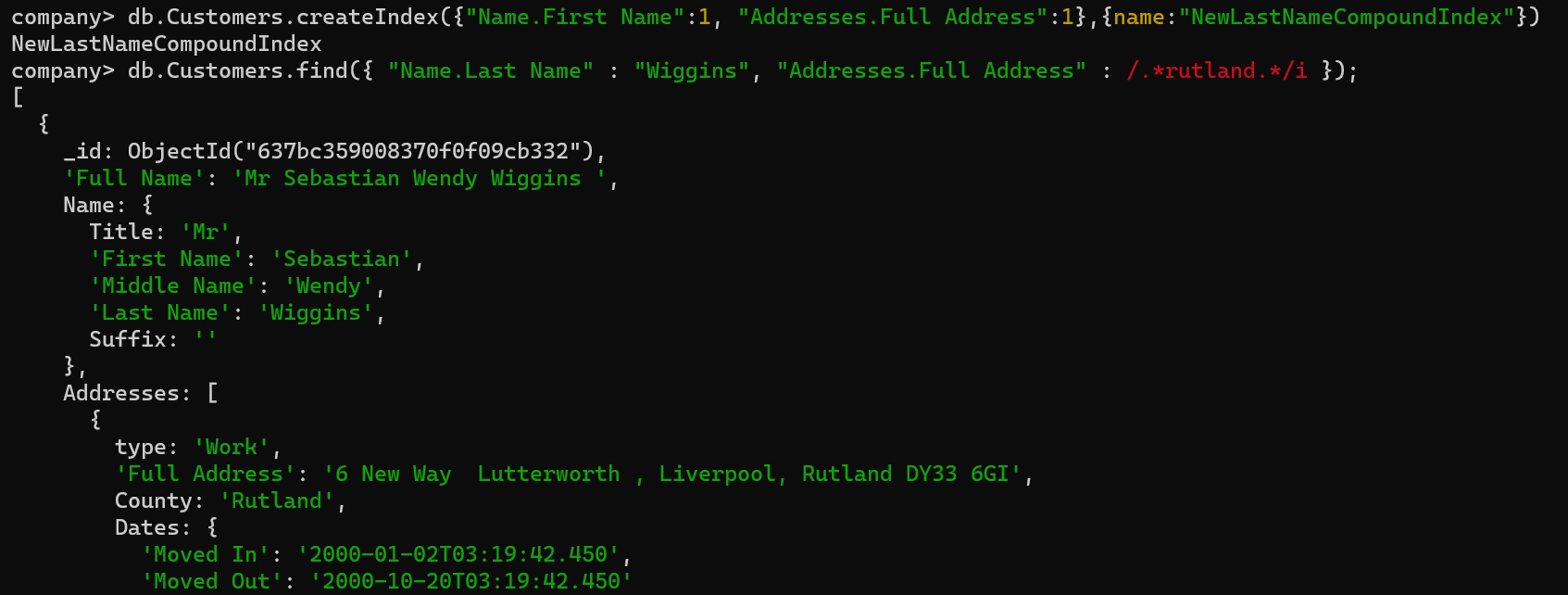
- Run the query we’ve been using. You will observe that the execution time shoots

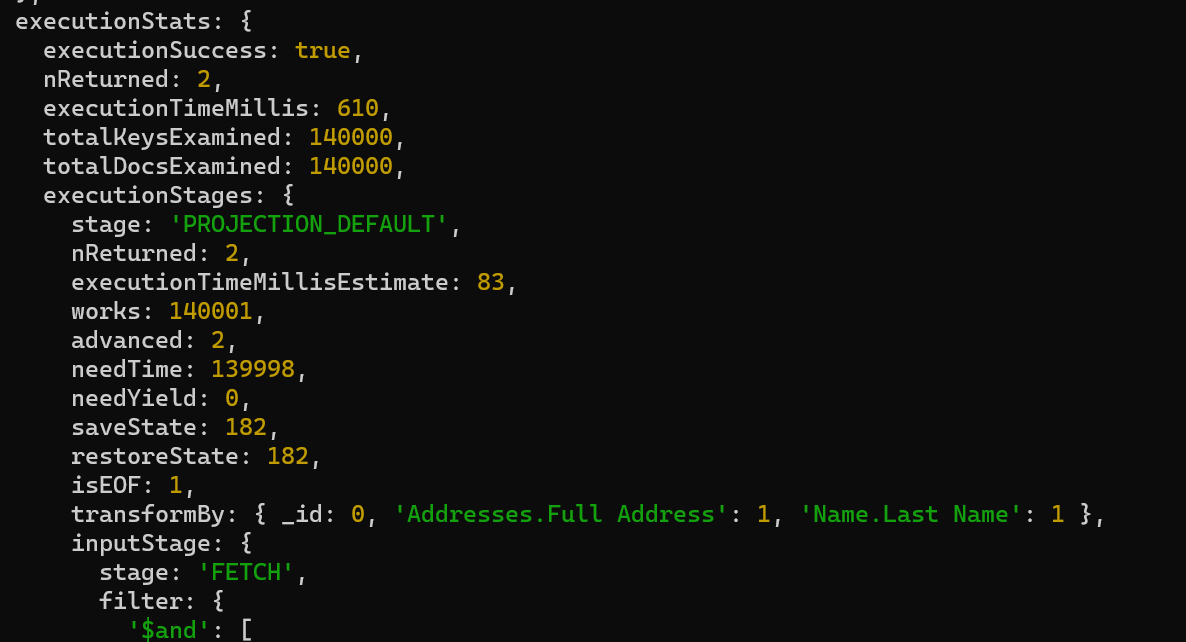


db.Customers.find( { "Name.Last Name": "Johnston" }, { "\_id": 0, "Name.First Name": 1, "Name.Last Name": 1 }).sort({ "Name.First Name": 1 }).explain("executionStats")

Exercise #3:

Create an index over the “Name.Last Name” and “Addresses.Full Address” fields to speed up the following query that returns information about a customer called Wiggins who lives in Rutland:





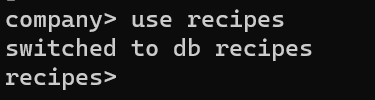
db.Customers.createIndex({"Name.First Name":1, "Addresses.Full Address":1}, {name:"NewLastNameCompoundIndex"})

db.Customers.find({ "Name.Last Name" : "Wiggins", "Addresses.Full Address" : /.\*rutland.\*/i });

db.Customers.find({"Name.Last Name":"Wiggins", "Addresses.Full Address":/.\*rutland.\*/i}, {"\_id": 0, "Addresses.Full Address":1, "Name.Last Name":1 }).sort({ "Name.First Name":1}).explain("executionStats")

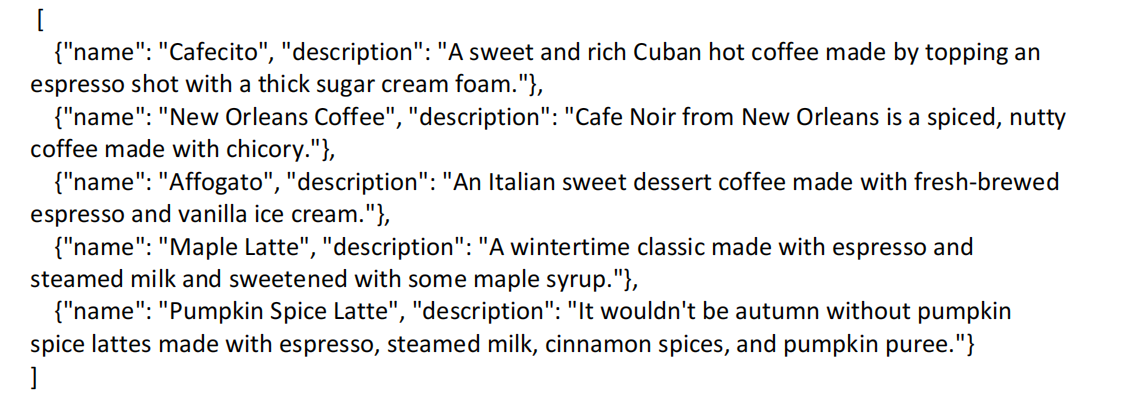
Exercise #4

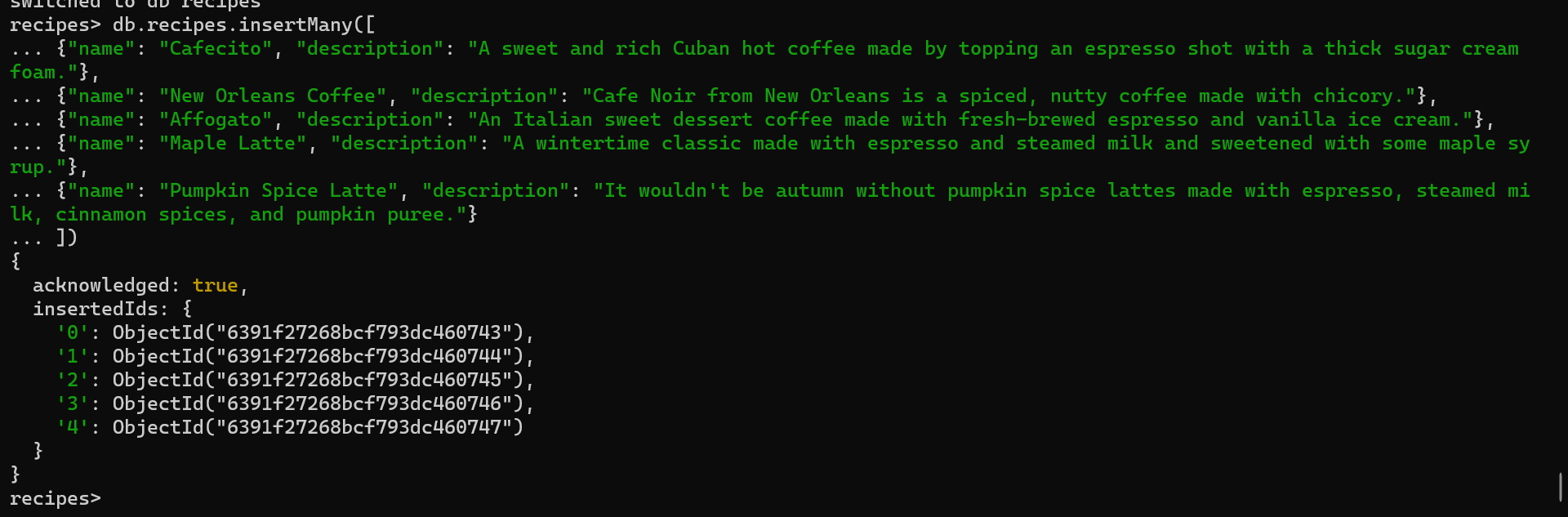
- Create a recipes database



use recipes

- Insert the following data:





db.recipes.insertMany([

{"name": "Cafecito", "description": "A sweet and rich Cuban hot coffee made by topping an espresso shot with a thick sugar cream foam."},

{"name": "New Orleans Coffee", "description": "Cafe Noir from New Orleans is a spiced, nutty coffee made with chicory."},

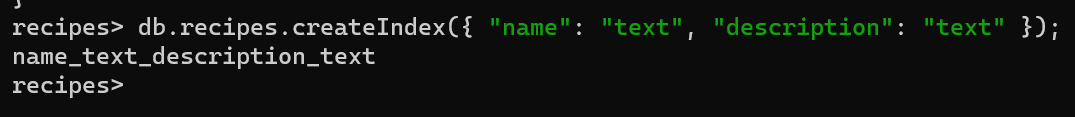
{"name": "Affogato", "description": "An Italian sweet dessert coffee made with fresh-brewed espresso and vanilla ice cream."},

{"name": "Maple Latte", "description": "A wintertime classic made with espresso and steamed milk and sweetened with some maple syrup."},

{"name": "Pumpkin Spice Latte", "description": "It wouldn't be autumn without pumpkin spice lattes made with espresso, steamed milk, cinnamon spices, and pumpkin puree."}

])

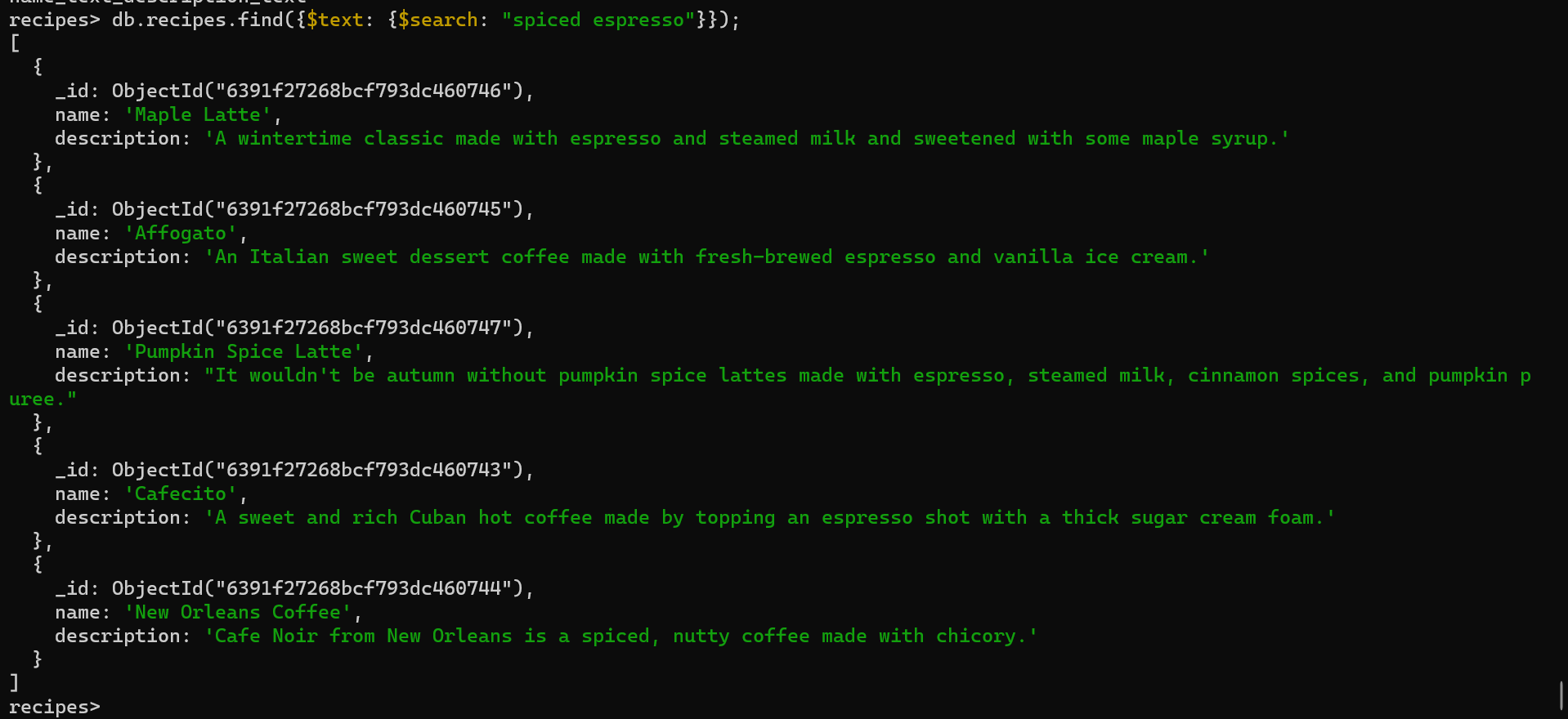
- Now create a text index that includes the name and description fields



db.recipes.createIndex({ "name": "text", "description": "text" });

Exercise #5:

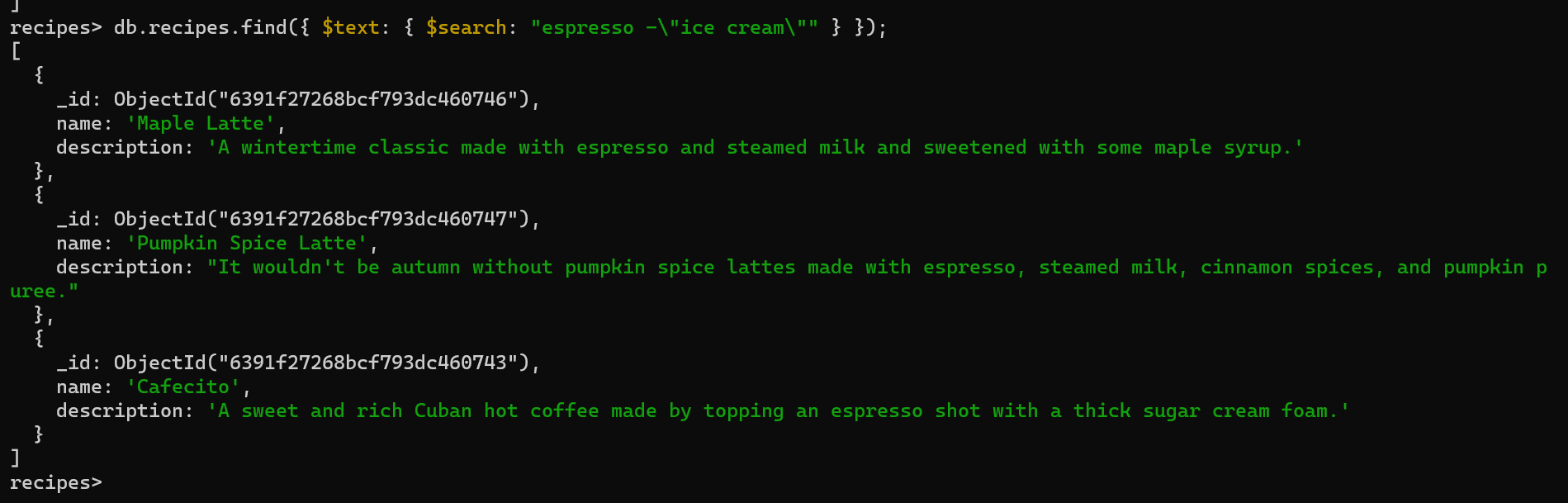
Try looking up documents with a two-word query, spiced espresso, to look for a spicy, espresso-based coffee.



db.recipes.find({$text: {$search: "spiced espresso"}});

Exercise #6:

You can also exclude full phrases. Try searching for espressos without “ice cream”. You need to escape the double quotes with backslashes.



db.recipes.find({ $text: { $search: "espresso -\"ice cream\"" } });