NAME:- Bryce Ferreira
ENROLLMENT NO:- MITU20BTCS0076
ROLL NO.:- 2203129
Class:- TY CSE CORE:- 3

Practical:-6

Part A

Practical Objective:

i) To learn how to perform aggregate function on database
ii) To learn how to perform indexing function on database
iii) To learn how to administer the database

Prerequisite: Understanding of concept of aggregate function,indexing and administration

Software: Mongodb

CO Mapping:
CO3: To get hands on exposure on NOSQL(Mongo) DB.

Practical Outcomes: At the end of this practical student will be able to:
Perform Aggregate function, how to do indexing on database using mongo db

Theory:

1. Aggregate Function

a) $match
b) $project
c) $group
d) $sort
e) $unwind
f) $count

2. Index

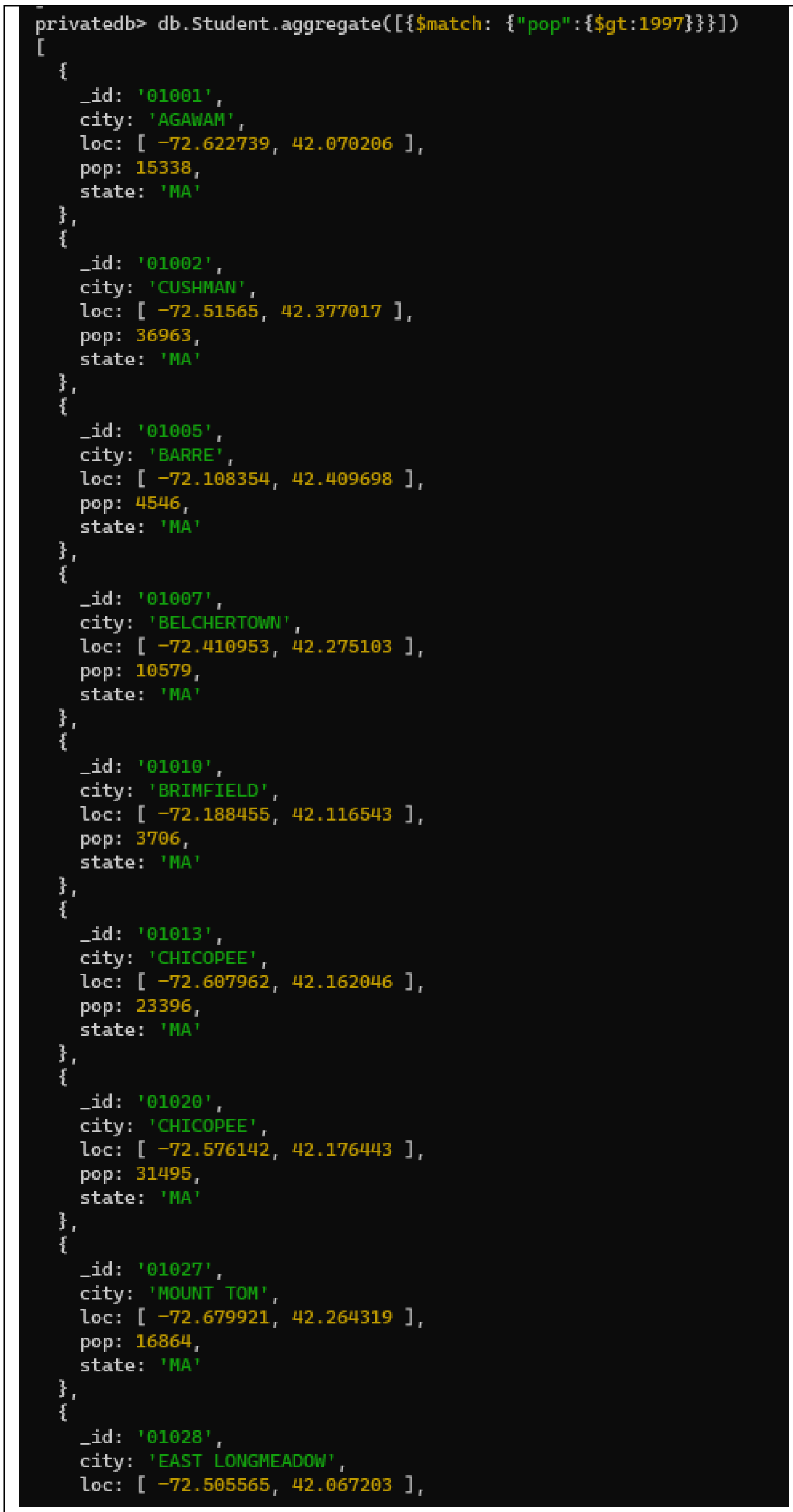
Create an index in Indexing section in collection of your own database

Procedure:

1. Formulate the function for given problem.
2. Write the NOSQL query with proper input.
3. Execute the query.

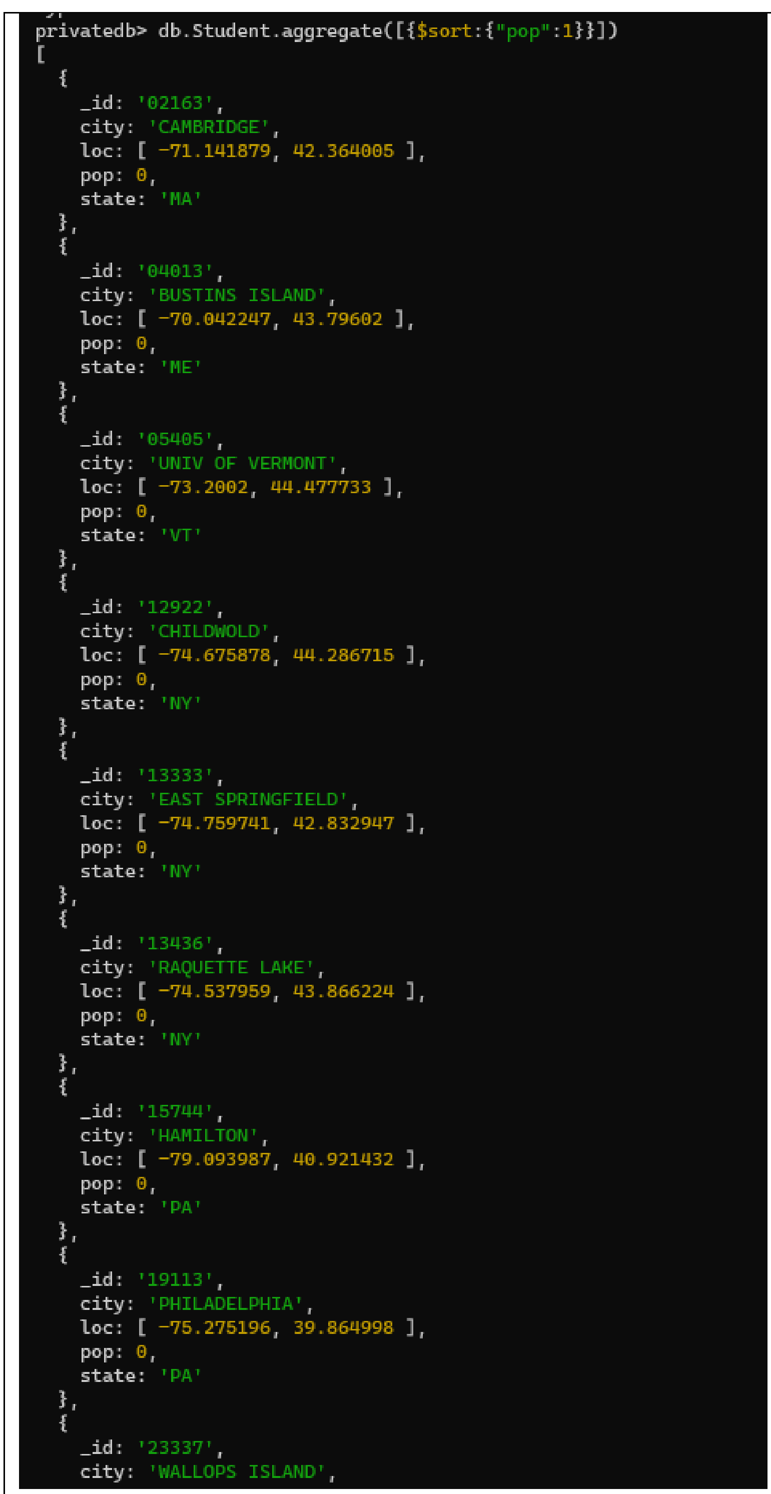
Practice Exercise:

1. Display the result for pop > 1997



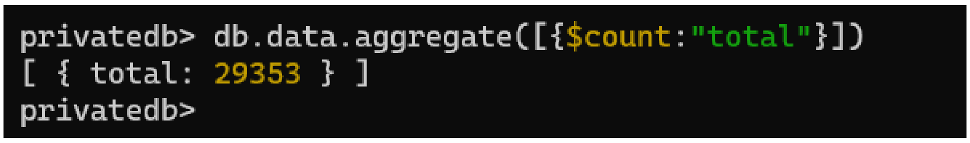
```
privatedb> db.Student.aggregate([{$match: {"pop":{$gt:1997}}}])
[
  {
    _id: '01001',
    city: 'AGAWAM',
    loc: [ -72.622739, 42.070206 ],
    pop: 15338,
    state: 'MA'
  },
  {
    _id: '01002',
    city: 'CUSHMAN',
    loc: [ -72.51565, 42.377017 ],
    pop: 36963,
    state: 'MA'
  },
  {
    _id: '01005',
    city: 'BARRE',
    loc: [ -72.108354, 42.409698 ],
    pop: 4546,
    state: 'MA'
  },
  {
    _id: '01007',
    city: 'BELCHERTOWN',
    loc: [ -72.410953, 42.275103 ],
    pop: 10579,
    state: 'MA'
  },
  {
    _id: '01010',
    city: 'BRIMFIELD',
    loc: [ -72.188455, 42.116543 ],
    pop: 3706,
    state: 'MA'
  },
  {
    _id: '01013',
    city: 'CHICOPEE',
    loc: [ -72.607962, 42.162046 ],
    pop: 23396,
    state: 'MA'
  },
  {
    _id: '01020',
    city: 'CHICOPEE',
    loc: [ -72.576142, 42.176443 ],
    pop: 31495,
    state: 'MA'
  },
  {
    _id: '01027',
    city: 'MOUNT TOM',
    loc: [ -72.679921, 42.264319 ],
    pop: 16864,
    state: 'MA'
  },
  {
    _id: '01028',
    city: 'EAST LONGMEADOW',
    loc: [ -72.505565, 42.067203 ],
```

2. **On the basis of state calculate average of pop,maximum of pop and sum of pop**
3. **Sort the records in increasing order**



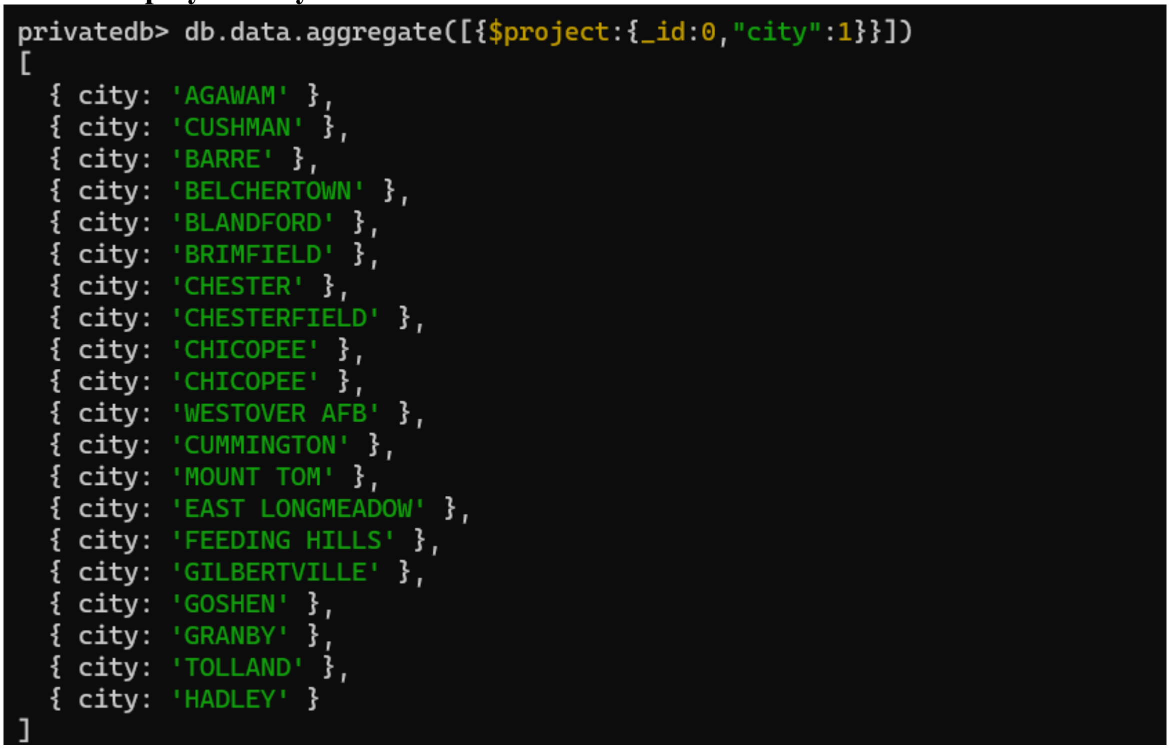
```
privatedb> db.Student.aggregate([{$sort:{"pop":1}}])
[
  {
    _id: '02163',
    city: 'CAMBRIDGE',
    loc: [ -71.141879, 42.364005 ],
    pop: 0,
    state: 'MA'
  },
  {
    _id: '04013',
    city: 'BUSTINS ISLAND',
    loc: [ -70.042247, 43.79602 ],
    pop: 0,
    state: 'ME'
  },
  {
    _id: '05405',
    city: 'UNIV OF VERMONT',
    loc: [ -73.2002, 44.477733 ],
    pop: 0,
    state: 'VT'
  },
  {
    _id: '12922',
    city: 'CHILDWOLD',
    loc: [ -74.675878, 44.286715 ],
    pop: 0,
    state: 'NY'
  },
  {
    _id: '13333',
    city: 'EAST SPRINGFIELD',
    loc: [ -74.759741, 42.832947 ],
    pop: 0,
    state: 'NY'
  },
  {
    _id: '13436',
    city: 'RAQUETTE LAKE',
    loc: [ -74.537959, 43.866224 ],
    pop: 0,
    state: 'NY'
  },
  {
    _id: '15744',
    city: 'HAMILTON',
    loc: [ -79.093987, 40.921432 ],
    pop: 0,
    state: 'PA'
  },
  {
    _id: '19113',
    city: 'PHILADELPHIA',
    loc: [ -75.275196, 39.864998 ],
    pop: 0,
    state: 'PA'
  },
  {
    _id: '23337',
    city: 'WALLOPS ISLAND',
```

4. **Count the number of records**



```
privatedb> db.data.aggregate([{$count:"total"}])
[ { total: 29353 } ]
privatedb>
```

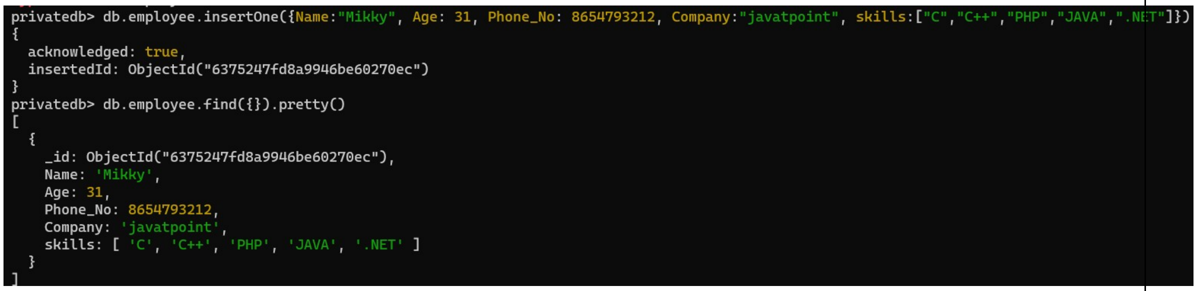
5. Display the city records but not with the id field



```
privatedb> db.data.aggregate([{$project:{_id:0,"city":1}}])
[
  { city: 'AGAWAM' },
  { city: 'CUSHMAN' },
  { city: 'BARRE' },
  { city: 'BELCHERTOWN' },
  { city: 'BLANDFORD' },
  { city: 'BRIMFIELD' },
  { city: 'CHESTER' },
  { city: 'CHESTERFIELD' },
  { city: 'CHICOPEE' },
  { city: 'CHICOPEE' },
  { city: 'WESTOVER AFB' },
  { city: 'CUMMINGTON' },
  { city: 'MOUNT TOM' },
  { city: 'EAST LONGMEADOW' },
  { city: 'FEEDING HILLS' },
  { city: 'GILBERTVILLE' },
  { city: 'GOSHEN' },
  { city: 'GRANBY' },
  { city: 'TOLLAND' },
  { city: 'HADLEY' }
]
```

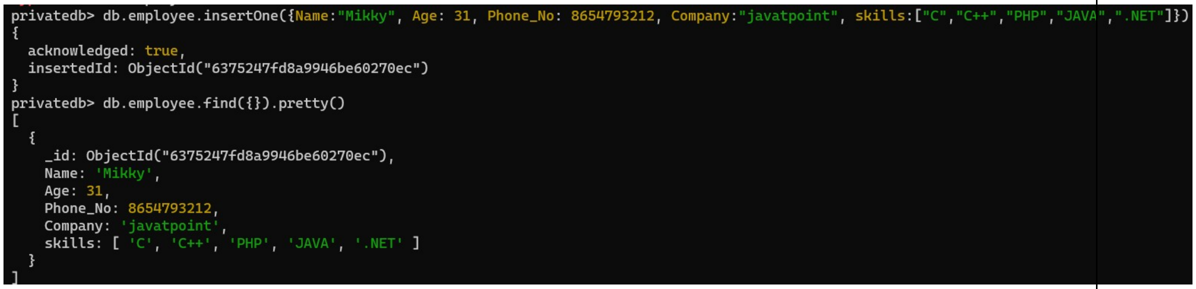
6. Create an employee collection.

```
db.employee.insertOne(
   {
      "name" : "Mikky",
      "age" : 31,
      "phone_no" : 8654793212
      "company" : "javatpoint",
      "skills" : ["C", "C++", "PHP", "Java", ".Net", ]
   }
);
```



```
privatedb> db.employee.insertOne({Name:"Mikky", Age: 31, Phone_No: 8654793212, Company:"javatpoint", skills:["C","C++","PHP","JAVA",".NET"]})
{
  acknowledged: true,
  insertedId: ObjectId("6375247fd8a9946be60270ec")
}
privatedb> db.employee.find({}).pretty()
[
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    Phone_No: 8654793212,
    Company: 'javatpoint',
    skills: [ 'C', 'C++', 'PHP', 'JAVA', '.NET' ]
  }
]
```

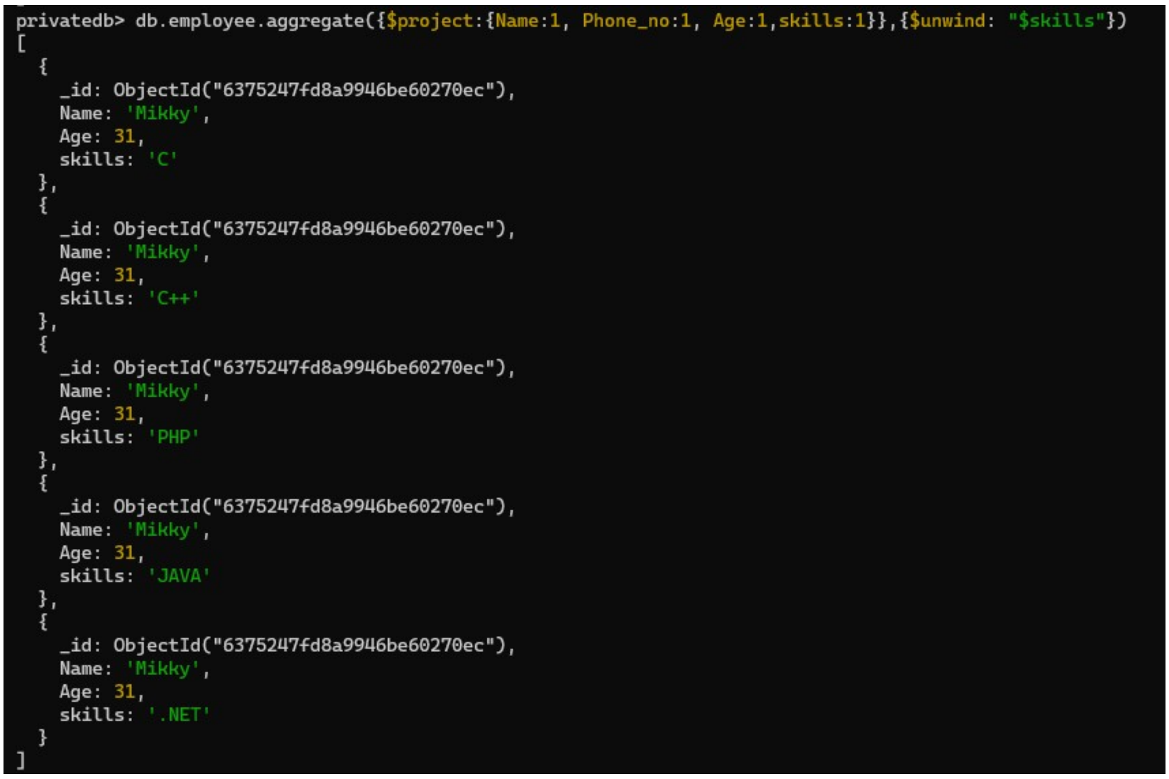
7. Now, display the documents from employee collection using the find() method.



```
privatedb> db.employee.insertOne({Name:"Mikky", Age: 31, Phone_No: 8654793212, Company:"javatpoint", skills:["C","C++","PHP","JAVA",".NET"]})
{
  acknowledged: true,
  insertedId: ObjectId("6375247fd8a9946be60270ec")
}
privatedb> db.employee.find({}).pretty()
[
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    Phone_No: 8654793212,
    Company: 'javatpoint',
    skills: [ 'C', 'C++', 'PHP', 'JAVA', '.NET' ]
  }
]
```

8. Use the $unwind operator and see how the output looks like("$skills")

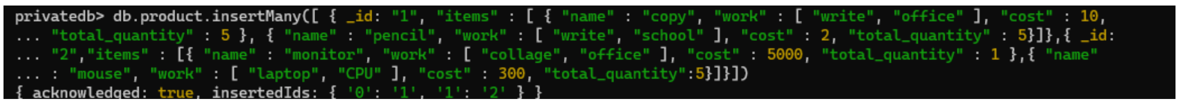
Using $unwind operator on embedded arrays



```
privatedb> db.employee.aggregate({$project:{Name:1, Phone_no:1, Age:1,skills:1}},{$unwind: "$skills"})
[
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    skills: 'C'
  },
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    skills: 'C++'
  },
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    skills: 'PHP'
  },
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    skills: 'JAVA'
  },
  {
    _id: ObjectId("6375247fd8a9946be60270ec"),
    Name: 'Mikky',
    Age: 31,
    skills: '.NET'
  }
]
```

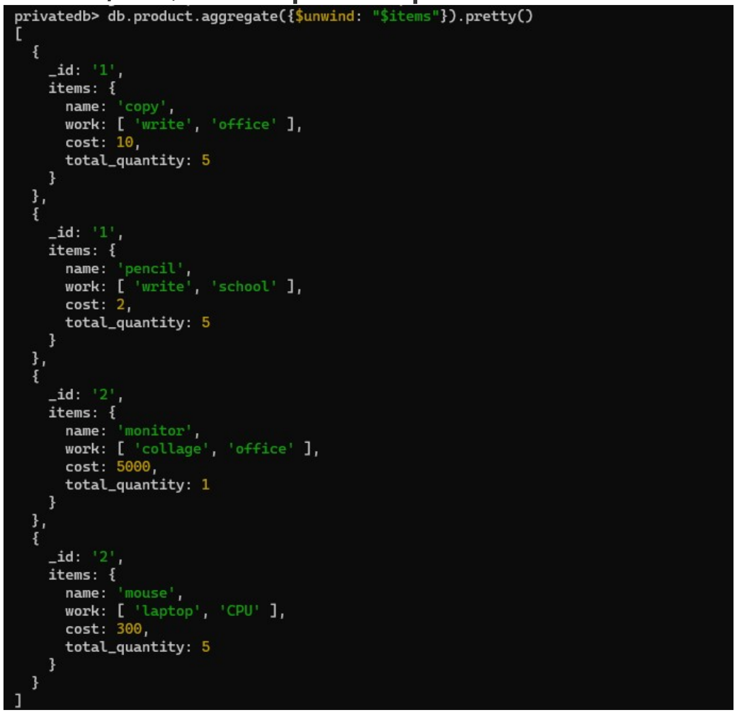
9. Now, create a product collection with the following documents.

```
db.product.insertMany([
 {
   _id: "1",
   "items" : [
     {
       "name" : "copy",
       "work" : [ "write", "office" ],
       "cost" : 10,
       "total_quantity" : 5
     },
     {
       "name" : "pencil",
                  "work" : [ "write", "school" ],
                  "cost" : 2,
                  "total_quantity" : 5
                }
              ]
            },
            {
              _id: "2",
              "items" : [
                {
                  "name" : "monitor",
                  "work" : [ "collage", "office" ],
                  "cost" : 5000,
                  "total_quantity" : 1
                },
                {
                  "name" : "mouse",
                  "work" : [ "laptop", "CPU" ],
                  "cost" : 300,
                  "total_quantity" : 5
              }
            ]
          }
        ])
```



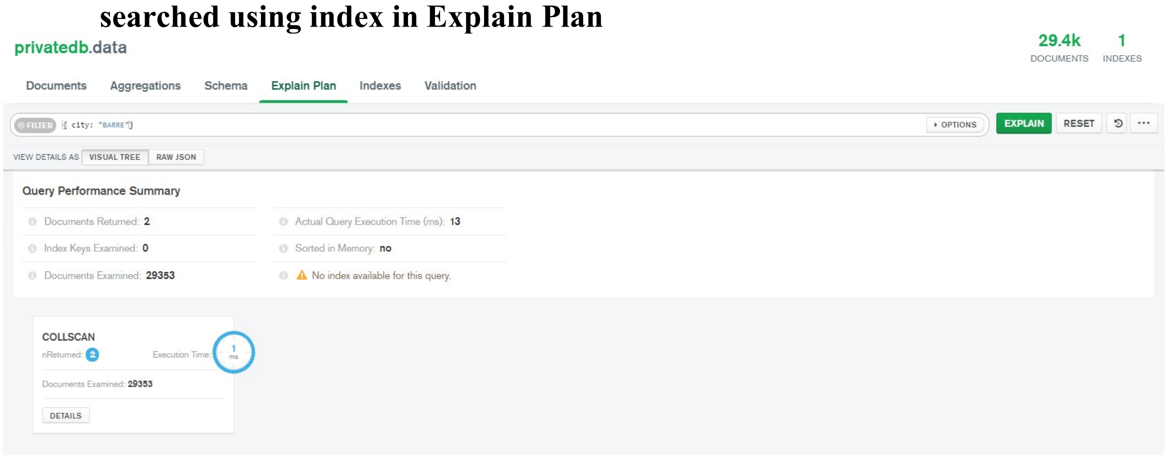
```
privatedb> db.product.insertMany([ { _id: "1", "items" : [ { "name" : "copy", "work" : [ "write", "office" ], "cost" : 10,
... "total_quantity" : 5 }, { "name" : "pencil", "work" : [ "write", "school" ], "cost" : 2, "total_quantity" : 5}]},{ _id:
... "2","items" : [{ "name" : "monitor", "work" : [ "collage", "office" ], "cost" : 5000, "total_quantity" : 1 },{ "name"
... : "mouse", "work" : [ "laptop", "CPU" ], "cost" : 300, "total_quantity":5}]}])
{ acknowledged: true, insertedIds: { '0': '1', '1': '2' } }
```

10. **Now, the $unwind operator is performed on the "items" and record the output**



```
privatedb> db.product.aggregate({$unwind: "$items"}).pretty()
[
  {
    _id: '1',
    items: {
      name: 'copy',
      work: [ 'write', 'office' ],
      cost: 10,
      total_quantity: 5
    }
  },
  {
    _id: '1',
    items: {
      name: 'pencil',
      work: [ 'write', 'school' ],
      cost: 2,
      total_quantity: 5
    }
  },
  {
    _id: '2',
    items: {
      name: 'monitor',
      work: [ 'collage', 'office' ],
      cost: 5000,
      total_quantity: 1
    }
  },
  {
    _id: '2',
    items: {
      name: 'mouse',
      work: [ 'laptop', 'CPU' ],
      cost: 300,
      total_quantity: 5
    }
  }
]
```

11. **Create an index for city and calculate the difference of timings and queries searched using index in Explain Plan**


privatedb.data

29.4k DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

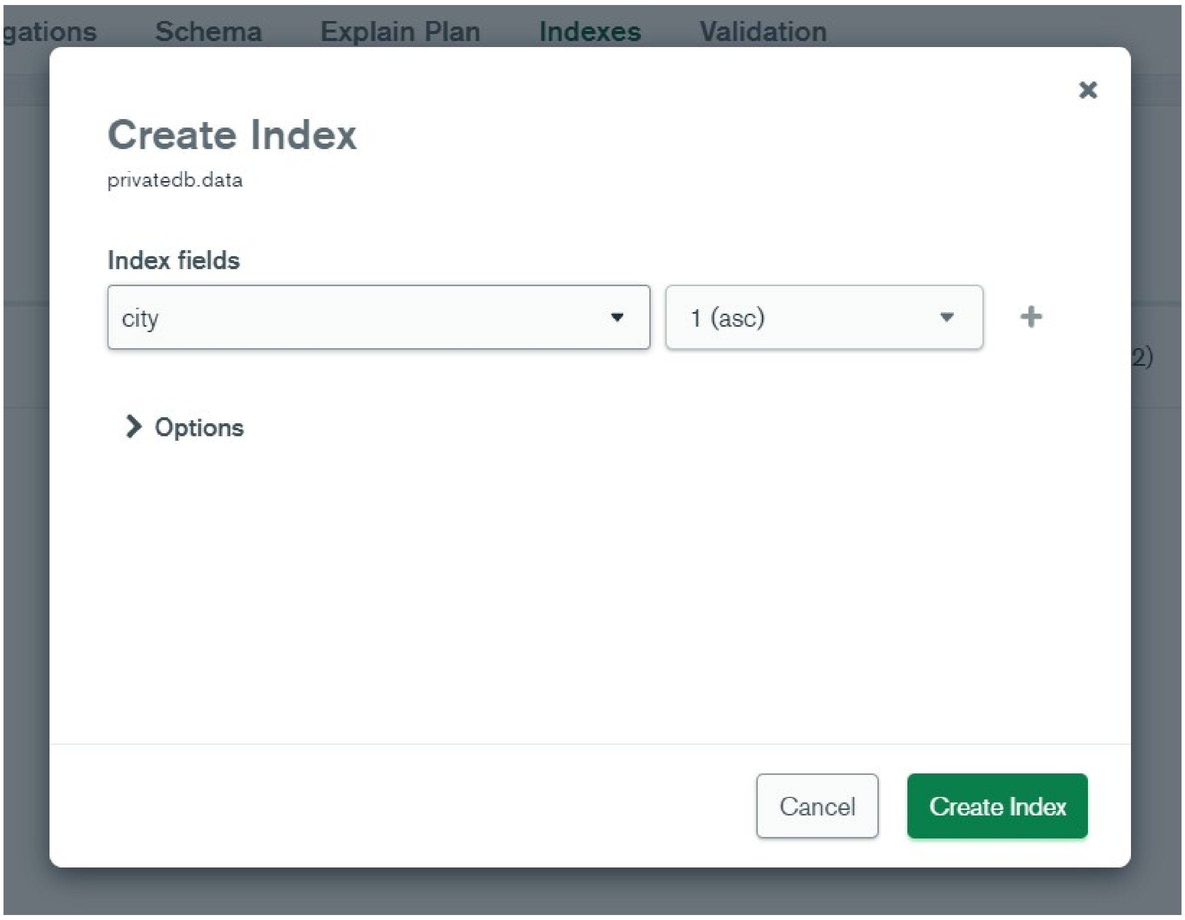
FILTER `{ city: "BARRE"}` OPTIONS EXPLAIN RESET

VIEW DETAILS AS VISUAL TREE RAW JSON

Query Performance Summary

Documents Returned: 2
Index Keys Examined: 0
Documents Examined: 29353
Actual Query Execution Time (ms): 13
Sorted in Memory: no
No index available for this query.

COLLSCAN
nReturned: 2 Execution Time: 1 ms
Documents Examined: 29353
DETAILS

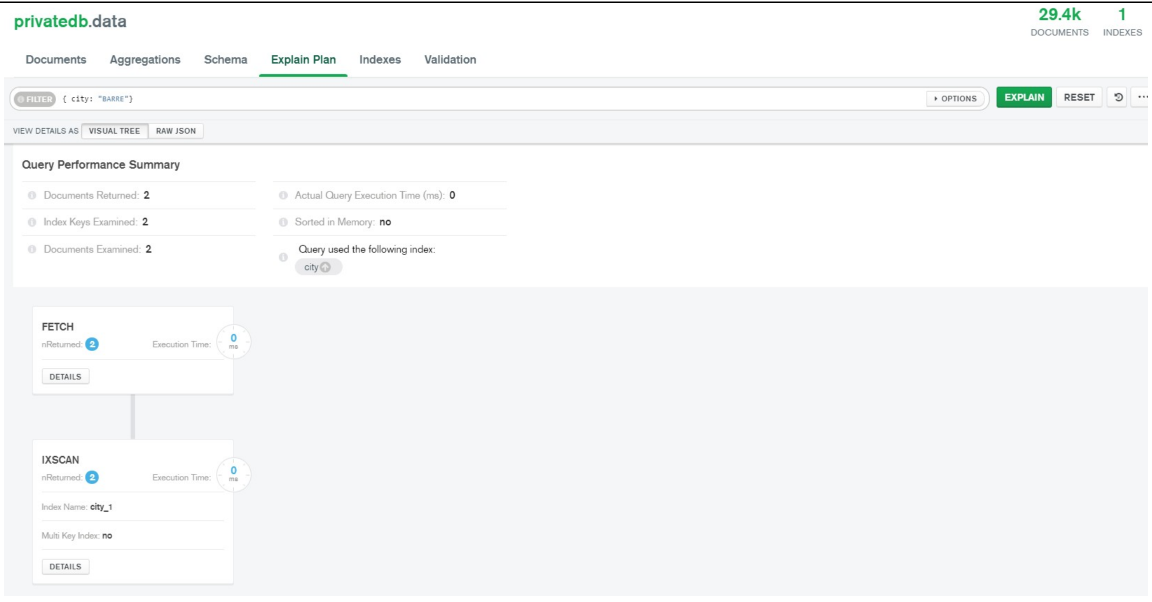


Create Index
privatedb.data

Index fields
city 1 (asc)

Options

Cancel Create Index



```
{ city: "BARRE"}
```

Query Performance Summary

Documents Returned: 2
Index Keys Examined: 2
Documents Examined: 2
Actual Query Execution Time (ms): 0
Sorted in Memory: no
Query used the following index:
city

FETCH
nReturned: 2
Execution Time: 0 ms

IXSCAN
nReturned: 2
Execution Time: 0 ms
Index Name: city_1
Multi Key Index: no

Instructions:

1. Write and execute the in MONGODB.
2. Paste the snapshot of the output in input & output section.

Part B

Code and Output:

Perform the operation and paste the running code here.

Observation & Learning:

Write your observation and learning after performing the task.

Conclusion:

Write statement of conclusion here.