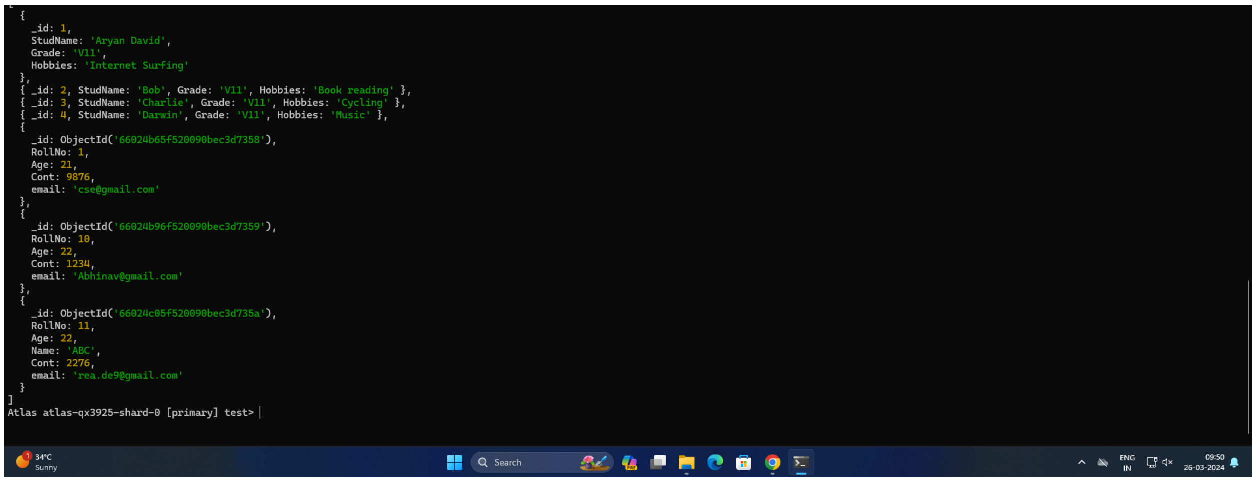
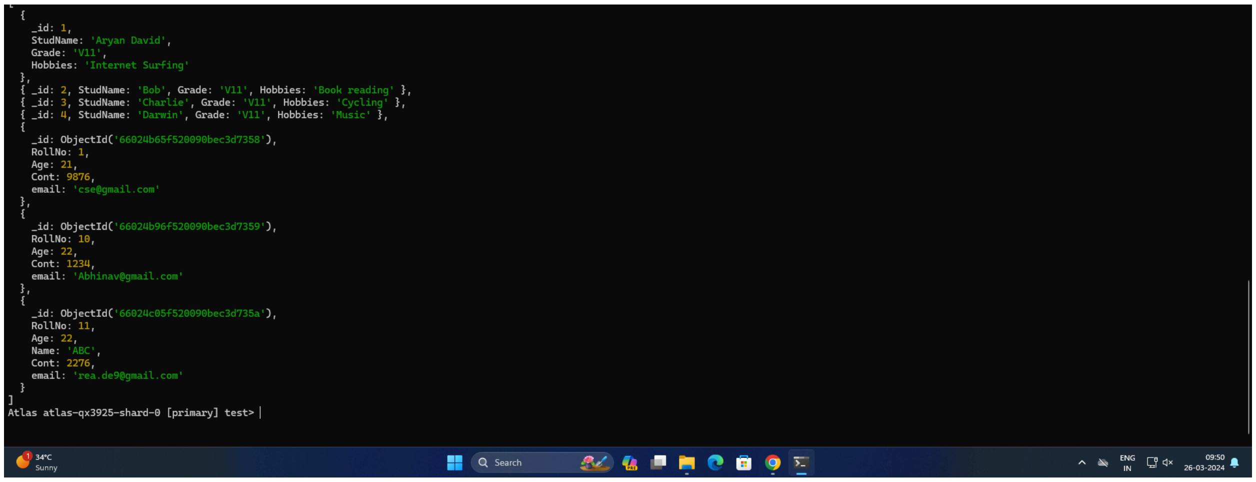
WEEK 3

1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.
2. Insert appropriate values
3. Write query to update Email-Id of a student with rollno 10.
4. . Replace the student name from “ABC” to “FEM” of rollno 11
5. Display Student Name and grade(Add if grade is not present)where the _id column is 1.
6. Update to add hobbies
7. Find documents where hobbies is set neither to Chess nor to Skating
8.Find documents whose name begins with A

```
db.createCollection("Student");

db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});
db.Student.insert({RollNo:2,Age:22,Cont:9976,email:"anushka.de9@gmail.com"});
db.Student.insert({RollNo:3,Age:21,Cont:5576,email:"anubhav.de9@gmail.com"});
db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});

db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})
db.Student.insert({RollNo:11,Age:22,Name:"ABC",Cont:2276,email:"rea.de9@gmail.com"});
db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})
```



```
  {
    _id: 1,
    StudName: 'Aryan David',
    Grade: 'V11',
    Hobbies: 'Internet Surfing'
  },
  { _id: 2, StudName: 'Bob', Grade: 'V11', Hobbies: 'Book reading' },
  { _id: 3, StudName: 'Charlie', Grade: 'V11', Hobbies: 'Cycling' },
  { _id: 4, StudName: 'Darwin', Grade: 'V11', Hobbies: 'Music' },
  {
    _id: ObjectId('66024b65f520090bec3d7358'),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'cse@gmail.com'
  },
  {
    _id: ObjectId('66024b96f520090bec3d7359'),
    RollNo: 10,
    Age: 22,
    Cont: 1234,
    email: 'Abhinav@gmail.com'
  },
  {
    _id: ObjectId('66024c05f520090bec3d735a'),
    RollNo: 11,
    Age: 22,
    Name: 'ABC',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
Atlas atlas-qx3925-shard-0 [primary] test>
```



```
  {
    _id: 1,
    StudName: 'Aryan David',
    Grade: 'V11',
    Hobbies: 'Internet Surfing'
  },
  { _id: 2, StudName: 'Bob', Grade: 'V11', Hobbies: 'Book reading' },
  { _id: 3, StudName: 'Charlie', Grade: 'V11', Hobbies: 'Cycling' },
  { _id: 4, StudName: 'Darwin', Grade: 'V11', Hobbies: 'Music' },
  {
    _id: ObjectId('66024b65f520090bec3d7358'),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'cse@gmail.com'
  },
  {
    _id: ObjectId('66024b96f520090bec3d7359'),
    RollNo: 10,
    Age: 22,
    Cont: 1234,
    email: 'Abhinav@gmail.com'
  },
  {
    _id: ObjectId('66024c05f520090bec3d735a'),
    RollNo: 11,
    Age: 22,
    Name: 'ABC',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
Atlas atlas-qx3925-shard-0 [primary] test>
```

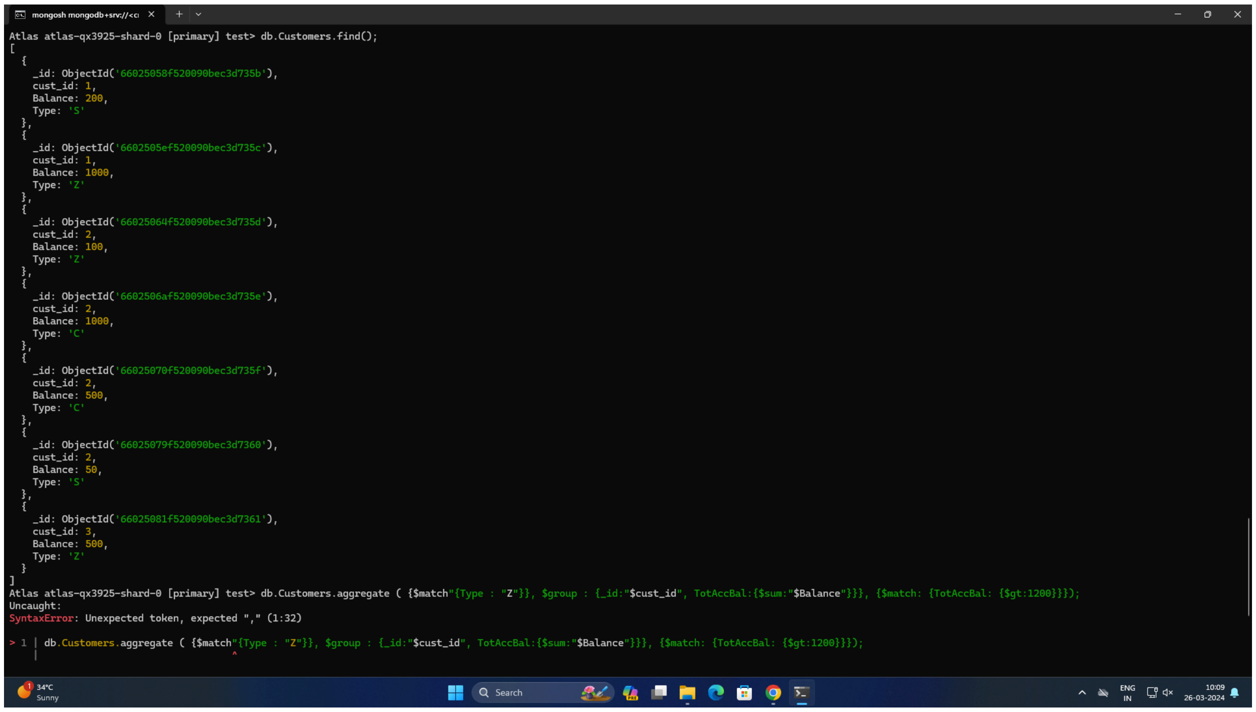
II. Perform the following DB operations using MongoDB.

1. Create a collection by name Customers with the following attributes.
Cust_id, Acc_Bal, Acc_Type
2. Insert at least 5 values into the table
3. Write a query to display those records whose total account balance is greater than 1200 of account type 'Z' for each customer_id.
4. Determine Minimum and Maximum account balance for each customer_i
5. Sort the documents based on Customer ID in ascending order and Account Balance in descending
order
6. Display only 2 nd and 3 rd records from the collection

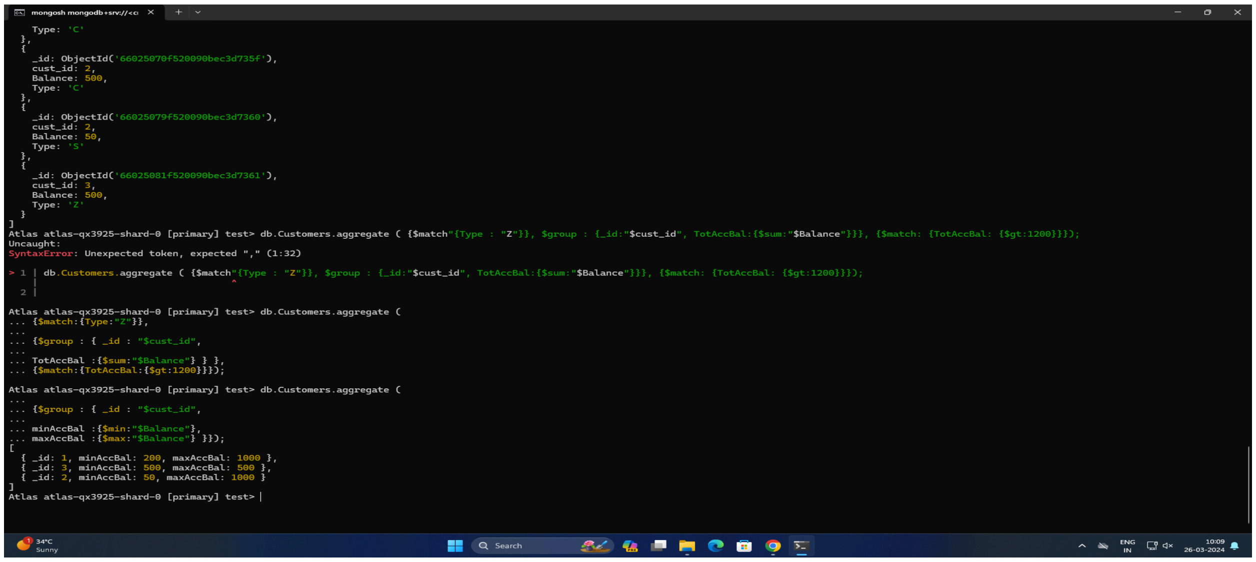
```
db.createCollection("Customers");

db.Customers.insert({cust_id:1,Balance:200, Type:"S"});

db.Customers.insert({cust_id:1,Balance:1000, Type:"Z"})

db.Customers.insert({cust_id:2,Balance:100, Type:"Z"});

db.Customers.insert({cust_id:2,Balance:1000, Type:"C"});

db.Customers.insert({cust_id:2,Balance:500, Type:"C"});

db.Customers.insert({cust_id:2,Balance:50, Type:"S"});

db.Customers.insert({cust_id:3,Balance:500, Type:"Z"});
```

db.Customers.aggregate ({$match:{Type:"Z"}},{$group : { _id : "$cust_id",TotAccBal :{$sum:"$Balance"} } }, {$match:{TotAccBal:{$gt:1200}}});

db.Customers.aggregate ({$group : { _id : "$cust_id",minAccBal :{$min:"$Balance"}, maxAccBal :{$max:"$Balance"} }});



```
Atlas atlas-qx3925-shard-0 [primary] test> db.Customers.find();
[
  {
    _id: ObjectId('66025058f520090bec3d735b'),
    cust_id: 1,
    Balance: 200,
    Type: 'S'
  },
  {
    _id: ObjectId('6602505ef520090bec3d735c'),
    cust_id: 1,
    Balance: 1000,
    Type: 'Z'
  },
  {
    _id: ObjectId('66025064f520090bec3d735d'),
    cust_id: 2,
    Balance: 100,
    Type: 'Z'
  },
  {
    _id: ObjectId('6602506af520090bec3d735e'),
    cust_id: 2,
    Balance: 1000,
    Type: 'C'
  },
  {
    _id: ObjectId('66025070f520090bec3d735f'),
    cust_id: 2,
    Balance: 500,
    Type: 'C'
  },
  {
    _id: ObjectId('66025079f520090bec3d7360'),
    cust_id: 2,
    Balance: 50,
    Type: 'S'
  },
  {
    _id: ObjectId('66025081f520090bec3d7361'),
    cust_id: 3,
    Balance: 500,
    Type: 'Z'
  }
]
Atlas atlas-qx3925-shard-0 [primary] test> db.Customers.aggregate ( {$match"{Type : "Z"}}, $group : {_id:"$cust_id", TotAccBal:{$sum:"$Balance"}}}, {$match: {TotAccBal: {$gt:1200}}});
Uncaught:
SyntaxError: Unexpected token, expected "," (1:32)

> 1 | db.Customers.aggregate ( {$match"{Type : "Z"}}, $group : {_id:"$cust_id", TotAccBal:{$sum:"$Balance"}}}, {$match: {TotAccBal: {$gt:1200}}});
    |                                 ^
```



```
    Type: 'C'
  },
  {
    _id: ObjectId('66025070f520090bec3d735f'),
    cust_id: 2,
    Balance: 500,
    Type: 'C'
  },
  {
    _id: ObjectId('66025079f520090bec3d7360'),
    cust_id: 2,
    Balance: 50,
    Type: 'S'
  },
  {
    _id: ObjectId('66025081f520090bec3d7361'),
    cust_id: 3,
    Balance: 500,
    Type: 'Z'
  }
]
Atlas atlas-qx3925-shard-0 [primary] test> db.Customers.aggregate ( {$match"{Type : "Z"}}, $group : {_id:"$cust_id", TotAccBal:{$sum:"$Balance"}}}, {$match: {TotAccBal: {$gt:1200}}});
Uncaught:
SyntaxError: Unexpected token, expected "," (1:32)

> 1 | db.Customers.aggregate ( {$match"{Type : "Z"}}, $group : {_id:"$cust_id", TotAccBal:{$sum:"$Balance"}}}, {$match: {TotAccBal: {$gt:1200}}});
    |                                 ^
  2 |

Atlas atlas-qx3925-shard-0 [primary] test> db.Customers.aggregate (
... {$match:{Type:"Z"}},
...
... {$group : { _id : "$cust_id",
...
... TotAccBal :{$sum:"$Balance"} } },
... {$match:{TotAccBal:{$gt:1200}}});

Atlas atlas-qx3925-shard-0 [primary] test> db.Customers.aggregate (
...
... {$group : { _id : "$cust_id",
...
... minAccBal :{$min:"$Balance"},
... maxAccBal :{$max:"$Balance"} }});
[
  { _id: 1, minAccBal: 200, maxAccBal: 1000 },
  { _id: 3, minAccBal: 500, maxAccBal: 500 },
  { _id: 2, minAccBal: 50, maxAccBal: 1000 }
]
Atlas atlas-qx3925-shard-0 [primary] test> |
```

Create a collection by the name blogPosts and it has 3 fields id, title and comments.

In the collection the comments field is an array which consists of user details. Each collection

consists of two user details inside the comments array- user name and text

Demonstrate the following

1. Adding an element into array

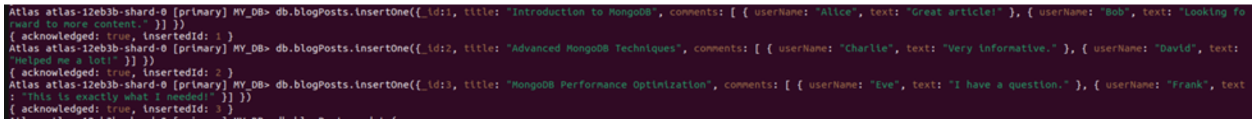
2. Display second element

3. Display size of the array

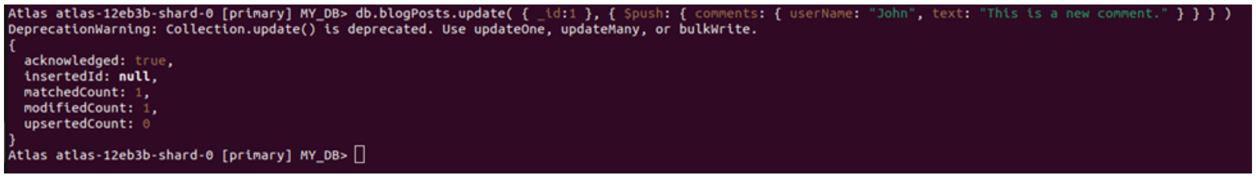
4. Display first two elements of the array

5. Update the document with id 4 and replace the element present in 1st index position of the

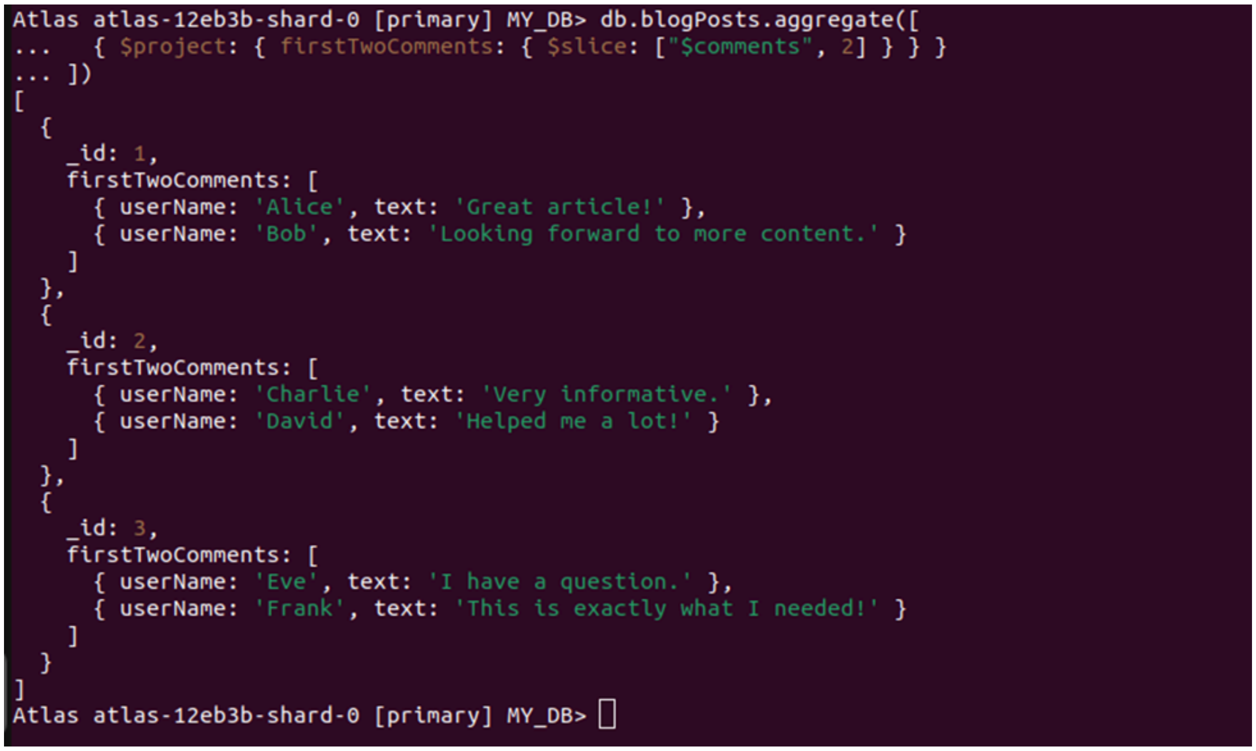
array with another array



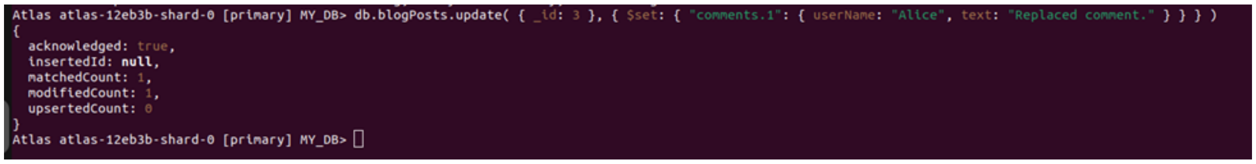
```
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.insertOne({_id:1, title: "Introduction to MongoDB", comments: [ { userName: "Alice", text: "Great article!" }, { userName: "Bob", text: "Looking fo
rward to more content." }] })
{ acknowledged: true, insertedId: 1 }
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.insertOne({_id:2, title: "Advanced MongoDB Techniques", comments: [ { userName: "Charlie", text: "Very informative." }, { userName: "David", text:
"Helped me a lot!" }] })
{ acknowledged: true, insertedId: 2 }
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.insertOne({_id:3, title: "MongoDB Performance Optimization", comments: [ { userName: "Eve", text: "I have a question." }, { userName: "Frank", text
: "This is exactly what I needed!" }] })
{ acknowledged: true, insertedId: 3 }
```



```
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.update( { _id:1 }, { $push: { comments: { userName: "John", text: "This is a new comment." } } } )
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> 
```



```
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.aggregate([
...   { $project: { firstTwoComments: { $slice: ["$comments", 2] } } }
... ])
[
  {
    _id: 1,
    firstTwoComments: [
      { userName: 'Alice', text: 'Great article!' },
      { userName: 'Bob', text: 'Looking forward to more content.' }
    ]
  },
  {
    _id: 2,
    firstTwoComments: [
      { userName: 'Charlie', text: 'Very informative.' },
      { userName: 'David', text: 'Helped me a lot!' }
    ]
  },
  {
    _id: 3,
    firstTwoComments: [
      { userName: 'Eve', text: 'I have a question.' },
      { userName: 'Frank', text: 'This is exactly what I needed!' }
    ]
  }
]
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> 
```



```
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> db.blogPosts.update( { _id: 3 }, { $set: { "comments.1": { userName: "Alice", text: "Replaced comment." } } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-12eb3b-shard-0 [primary] MY_DB> 
```