



Level 2

Next Level Web Development

▼ Mission 2 (MongoDB and Mongoose)

- ▼ Module 5: In-depth exploration of MongoDB queries
- ▼ Video 1 (Install MongoDB shell and noSQL Booster)

MongoDB Compass download. (search Google download mongoDB)

সেখানে shell এর মধ্যে আমরা আমাদের query গুলো লিখতে পারবো।

যেমনঃ

```
show dbs
```

এটি লিখলে আমাদের যেগুলো ডাটাবেস আছে সেগুলো দেখাবে।

```
localhost:27017 mongosh: localhost:27017 +  
>_MONGOSH  
> show dbs  
< admin 40.00 KiB  
config 108.00 KiB  
local 40.00 KiB  
test>
```

কোন ডাটাবেস তৈরি করতে চাইলে প্রথমে use লিখে ডাটাবেস এর নাম দিতে হবে।

```
>_MONGOSH
> show dbs
< admin   40.00 KiB
  config  108.00 KiB
  local    40.00 KiB
> use practice
< switched to db practice
practice>
```

ধরি collection তৈরি করতে চাই তাহলে

```
db.createcollection("test")
```

```
>_MONGOSH
> show dbs
< admin   40.00 KiB
  config  108.00 KiB
  local    40.00 KiB
> use practice
< switched to db practice
> db.createCollection("test")
< { ok: 1 }
practice>
```

ধরি কোন একটি ডাটা অ্যাড করতে চাই তাহলে

```
db.getcollection("test").insertOne({name:"Next level web development"})
```

```
>_MONGOSH
> use practice
< switched to db practice
> db.getCollection('test').insertOne({name:"Next level web developement"})
< {
  acknowledged: true,
  insertedId: ObjectId('673065af57fe790328908b11')
}
practice>
```

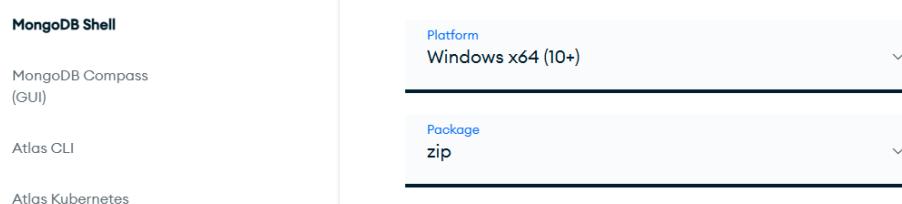
এখন যদি সবগুলো ডাটা দেখতে চাই তাহলে

```
db.getcollection("test").find()
```

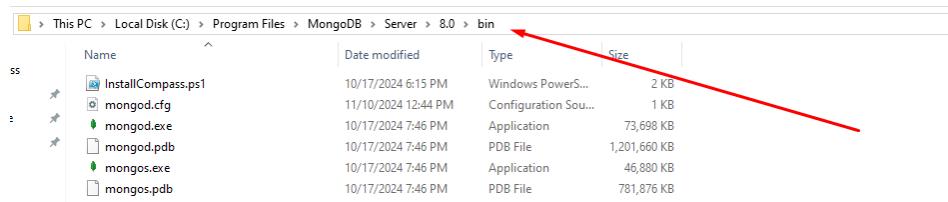
```
> db.getCollection("test").find()
< {
    _id: ObjectId('673065af57fe790328908b11'),
    name: 'Next level web development'
}
practice>
```

যদি চাই MongoDB shell থেকে কোড না লিখে তার কোডগুলো CMD থেকে লিখবো তাহলে mongoose shell টা path এ অ্যাড করতে হবে ।

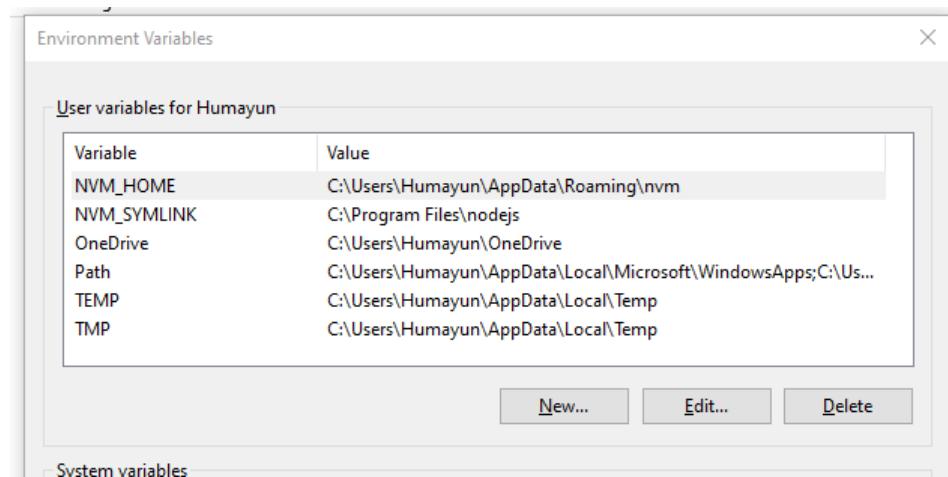
তার জন্য Environment variable এ path অ্যাড করতে হবে ।



তার জন্য MongoDB Shell download হবে ।



এই path টা কপি করবো ।



Edit environment variable এ যাবো Path এ ডাবল ক্লিক করবো

```
C:\Users\Humayun>mongod --version
db version v8.0.3
Build Info: {
    "version": "8.0.3",
    "gitVersion": "89d97f2744a2b9851ddfb51bdf22f687562d9b06",
    "modules": [],
    "allocator": "tcmalloc-gperf",
    "environment": {
        "distmod": "windows",
        "distarch": "x86_64",
        "target_arch": "x86_64"
    }
}
C:\Users\Humayun>
```

mongod —version দিলে version দেখাবে ।

```
C:\Users\Humayun>mongosh
Current Mongosh Log ID: 67306b74c28b055fc60d818f
Connecting to:      mongod://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2
.3.3
Using MongoDB:     8.0.3
Using Mongosh:     2.3.3

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

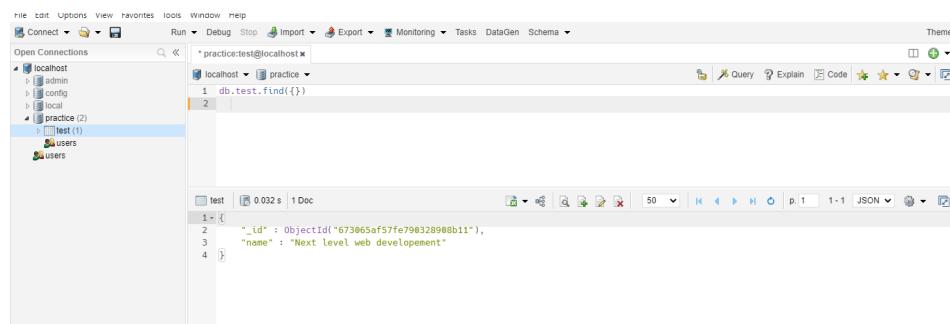
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2024-11-10T12:44:17.734+06:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----
test>
```

এখন CMD তে mongosh দিলে রান হবে ।

অন্য আরো একটি গ্রাফিকাল ইউজার ইন্টারফেস ব্যবহার করবো ।

studio 3t ব্যবহার করা যায় । আমরা noSQL booster ব্যবহার করবো ।



প্রগ্রাম টি রান করতে চাইলে ctrl + enter চাপতে হবে ।

▼ Video 2 (insertOne, find, findOne, field filtering, project)

insertOne

একটি ডাটা যোগ করার জন্য **insertOne**

```
* practice:test@localhost x
localhost ▾ practice ▾
1 db.test.insertOne({name:"Something"})
2

🕒 0.036 s
1 [
2   "acknowledged" : true,
3   "insertedId" : ObjectId("67307dc44b7729402ef17b86")
4 ]
```

insertMany

একাধিক ডাটা একসাথে যোগ করার জন্য insertMany ব্যবহার করা হয়। তার জন্য [] এর মধ্যে ডাটা গুলো দিতে হয় একটার থেকে অন্য টা কমা(,) দিয়ে আলাদা করতে হবে।

```
* practice:test@localhost x | practice:test@localhost (1) x | practice:test@localhost (2) x
localhost ▾ practice ▾
1 db.test.insertMany([
2   {name:"Complete Web Development"},
3   {name:"Next Level Web Development"}
4 ])
5

🕒 0.032 s
1 [
2   "acknowledged" : true,
3   "insertedIds" : [
4     ObjectId("673080704b7729402ef17b87"),
5     ObjectId("673080704b7729402ef17b88")
6   ]
7 ]
```

আগের ডাটা গুলো সব ডিলিট করে দিলাম।

প্রাক্টিস এর জন্য ডাটা:

GitHub - HumayunKabirSobuj/mongodb-practice
 Contribute to HumayunKabirSobuj/mongodb-practice development by creating an account on GitHub.

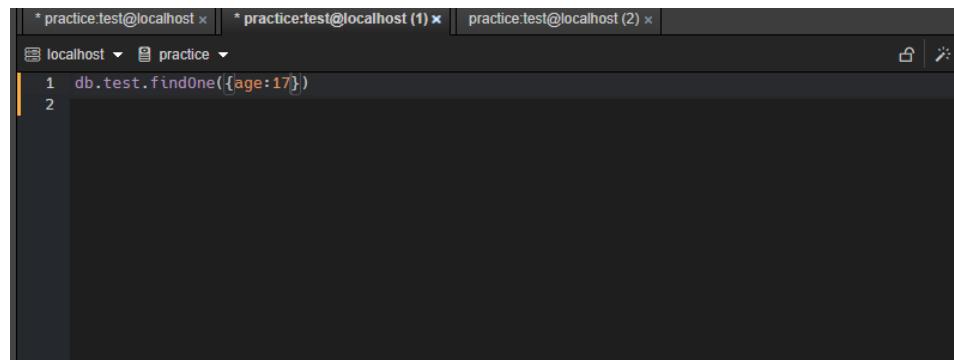
<https://github.com/HumayunKabirSobuj/mongodb-practice>

1 Contributor
0 Issues
0 Stars
0 Forks

এইখানের practice-data.json

findOne

এর মাধ্যমে একটি ডাটা খুজে দেয়।



```
* practice:test@localhost x * practice:test@localhost (1) x practice:test@localhost (2) x
localhost <-- practice <--
```

```
1 db.test.findOne({age:17})
2
```

age এর ভালু 17 এইরকম মিল হওয়া প্রথম ডাটাটি রিটার্ন করবে ।

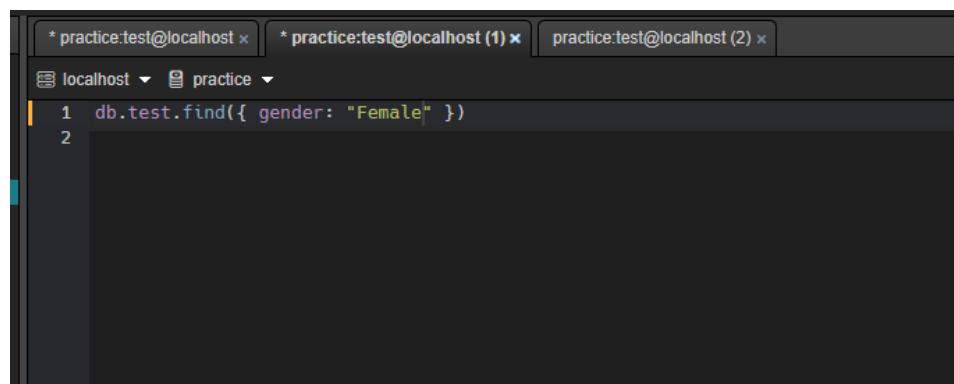
find

এর মাধ্যমে অনেকগুলো ডাটা একসাথে দেয় ।

যদি () এর মধ্যে কিছু না দেই তাহলে সবগুলো রিটার্ন করবে ।

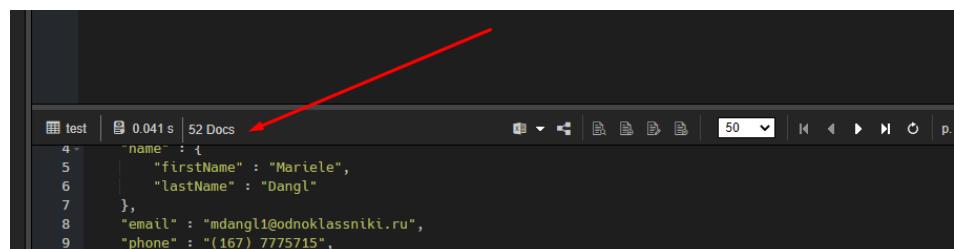
আর যদি ফিল্টার করতে চাই তাহলে findOne এর মতো করে দিতে পারি যেগুলো মিলবে সবগুলো রিটার্ন করবে ।

যেমনঃ



```
* practice:test@localhost x * practice:test@localhost (1) x practice:test@localhost (2) x
localhost <-- practice <--
```

```
1 db.test.find({ gender: "Female" })
2
```



```
test | 0.041 s | 52 Docs
```

```
4 "name" : {
5   "firstName" : "Marielle",
6   "lastName" : "Dangl"
7 },
8 "email" : "mdangl1@odnoklassniki.ru",
9 "phone" : "(167) 7775715",
```

এক্ষেত্রে যেগুলো ম্যাচ হবে তার সব ইনফরমেশন দেখাবে ।

যদি মনে হয় সবগুলো ডাটা আমার দরকার শুধু name, email, phone ,gender এই field গুলো দরকার ।

তাহলে {} এর পরে কমা দিয়ে {} এর মধ্যে বলে দিতে হবে আমার যেগুলো যেগুলো দরকার ।

A screenshot of the MongoDB Compass interface. The top navigation bar includes 'Tools', 'Window', and 'Help' along with standard application icons. Below the bar, there are tabs for 'practice:test@localhost', 'practice:test@localhost (1)', and 'practice:test@localhost (2)'. The main area shows a query in the 'localhost' database named 'practice':

```
1 db.test.find({ gender: "Male" }, { gender: 1 })
```

A red arrow points from the text 'gender' in the first document projection field to the word 'gender' in the query's filter clause.

১ মানে হলো এটা আমার লাগবে ।

তাহলে শুধু gender দিবে ।

A screenshot of the MongoDB Compass interface, similar to the previous one but with a different query:

```
1 db.test.find({ gender: "Male" }, {})
```

The projection field is now an empty object {}, indicating that no fields will be returned in the documents.

A screenshot of the MongoDB Compass interface showing the results of the query. The results table has a header row with columns for '_id', 'name', 'email', 'phone', and 'gender'. Below the header, two documents are listed:

_id	name	email	phone	gender
6406ad63fc13ae5a40000066	Otto	omirfin2@i2i.jp	(670) 1831100	Male
6406ad63fc13ae5a40000067	Mirfin			Male

এটাকে বলা হয় **field filtering** .

A screenshot of the MongoDB Compass interface showing a more complex query:

```
1 db.test.find({ gender: "Male" }, { name: 1, email: 1, phone: 1, gender: 1 })
```

The projection field includes name, email, phone, and gender.

A screenshot of the MongoDB Compass interface showing the results of the complex query. The results table has a header row with columns for '_id', 'name', 'email', 'phone', and 'gender'. Below the header, two documents are listed:

_id	name	email	phone	gender
6406ad63fc13ae5a40000066	Otto	omirfin2@i2i.jp	(670) 1831100	Male
6406ad63fc13ae5a40000067	Mirfin			Male

এটা খুবেই গুরুত্বপূর্ণ কারণ FrontEnd এ সব ডাটা সবসময় প্রয়োজন হবে না।

এই কাজটি chaining ব্যবহার করেও করা যাবে। project method ব্যবহার করতে হবে।



```
// db.test.find({ gender: "Male" }, { name: 1, email: 1, phone: 1, gender: 1 })
db.test.find({ gender: "Male" }).project({ name: 1, email: 1, phone: 1, gender: 1 })
```

A screenshot of a MongoDB shell window. The command `db.test.find({ gender: "Male" }).project({ name: 1, email: 1, phone: 1, gender: 1 })` is entered. A red arrow points from the text above to the word `project` in the command.

তবে project method শুধুমাত্র find এর ক্ষেত্রে কাজ করতে পারে findOne এর ক্ষেত্রে কাজ করতে পারে না।

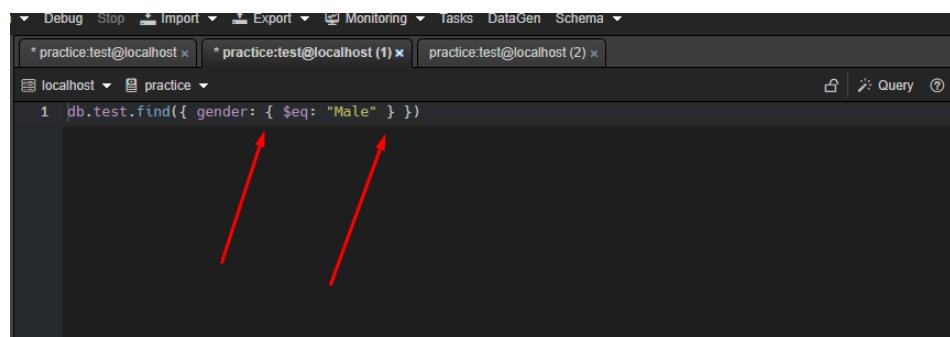
▼ Video 3 (\$eq, \$neq, \$gt, \$lt, \$gte, \$lte)

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/query/>

প্রথমেং Comparison Query Operators

Query শেখার thumb rule :

- যখন নতুন একটি অপারেটর আনবো তখন আমরা দুইপাশে দ্বিতীয় বন্ধনী {} দিয়ে দিবো।



```
1 db.test.find({ gender: { $eq: "Male" } })
```

A screenshot of a MongoDB shell window. The command `db.test.find({ gender: { \$eq: "Male" } })` is entered. Two red arrows point from the text above to the field names `gender` and the value `Male` in the query.

name, email, phone, gender প্রত্তি এক একটি field।

এগুলো field filtering এর মতো কাজ করে।

\$eq

syntax :

```
{ <field>: { $eq: <value> } }
```

```
* practice:test@localhost * practice:test@localhost (1) * practice:test@localhost (2)
localhost practice
1 db.test.findOne({ gender: { $eq: "Male" } })
```

A red arrow points from the text '\$eq' in the query to the '\$ne' section below.

name, email, phone, gender প্রতুতি এক একটি field।

eq = equal অর্থ সমান।

\$eq এর বিপরীত হলো \$ne

\$ne

ne = not equal অর্থ অসমান।

```
* practice:test@localhost * practice:test@localhost (1) * practice:test@localhost (2)
localhost practice
1 db.test.find({ age: { $ne: 12 } })
```

age 12 এর সমান নয় এমন সবগুলো দিবে।

#	createdAt	Docs
1	2023-07-03T09:20:03.000Z	97
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		

```
1 /* 1 createdAt:2023-07-03T09:20:03.000Z */
2 {
3   "_id": ObjectId("6406ad63fc13ae5a40000065"),
4   "name": {
5     "firstName": "Mariele",
6     "lastName": "Dangl"
7   },
8   "email": "mdangli@odnoklassniki.ru",
9   "phone": "(167) 7775715",
10  "gender": "Female",
11  "age": 21, ----->
12  "birthday": "3/13/2022",
13  "address": {
14    "street": "1188 Lerdahl Point",
```

\$gt

gt = greater than

```
Tools Window Help
Run Debug Stop Import Export Monitoring Tasks DataGen Schema
* practice:test@localhost * practice:test@localhost (1) * practice:test@localhost (2)
localhost practice
1 db.test.find({ age: { $gt: 18 } })
```

বয়স ১৮ এর উপরে যাদের সেগুলো সব দিবে ।

\$gte

gte = greater than or equal

```
Run Debug Stop Import Export Monitoring Tasks DataGen Schema
* practice:test@localhost * practice:test@localhost (1) * practice:test@localhost (2)
localhost practice
1 db.test.find({ age: { $gte: 30 } })
```

বয়স ৩০ অথবা ৩০ এর উপরে সেগুলো সব দিবে । অর্থাৎ ৩০ সহ নিবে ।

সেটা দেখানোর জন্য আমরা sort ব্যবহার করতে পারি ।

আমরা জানি sort দুইভাবে কাজ করে।

1. ছোট থেকে বড়
2. বড় থেকে ছোট

ছোট থেকে বড় হলে তাকে বলা হয় Assending.

বড় থেকে ছোট হলে তাকে বলা হয় Disending .

```
1 db.test.find({ age: { $gte: 30 } }).sort({ age: 1 })
```

age এর উপর ভিত্তি করে sort . ছোট থেকে বড় হলে অর্থাৎ Assending হলে 1 দিতে হবে ।

বড় থেকে ছোট হলে অর্থাৎ Disending হলে -1 দিতে হবে ।

```
db.test.find({ age: { $gte: 30 } }).sort({ age: 1 })
```

```
db.test.find({ age: { $gte: 30 } }).sort({ age: -1 })
```

\$lt

lt = less than

```
1 db.test.find({ age: { $lt: 30 } })
```

বয়স 30 এর থেকে কম যাদের তাদের গুলো দিবে ।

\$lte

lte = less than or equal

```
1 db.test.find({ age: { $lte: 30 } })
```

যাদের বয়স ৩০ বা তার থেকে কম।

▼ Video 4 (\$in, \$nin, implicit and condition)

\$in

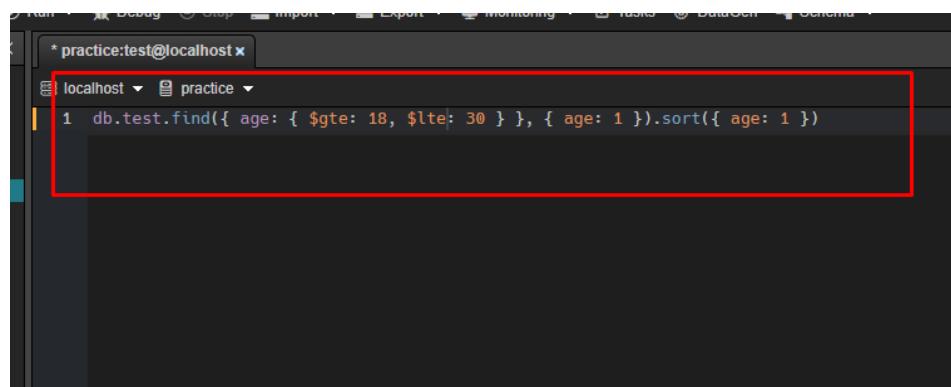
The **\$in** operator selects the documents where the value of a field equals any value in the specified array.

\$in এর মধ্যে যে Array value হিসেবে দিবো সেই value গুলোর উপর ডিটি করে সে query করে নিয়ে আসবে। এই value গুলোর মধ্যে যদি একটাও মিলে যায় তাহলে সে Document return করবে।

syntax:

```
{ field: { $in: [<value1>, <value2>, ... <valueN> ] } }
```

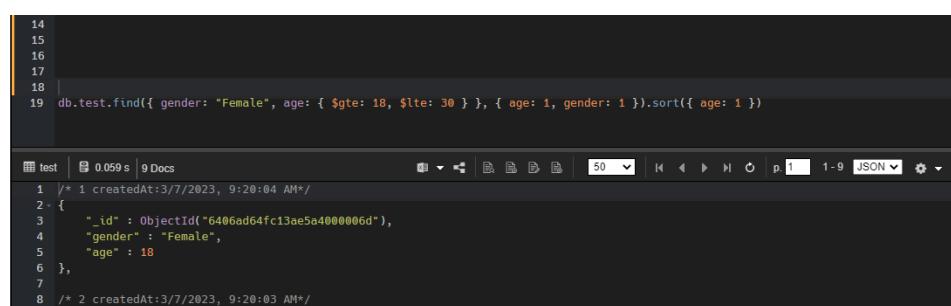
\$in এর মধ্যে যেসব value দিবো সে সেসব value থাকা document পাইলে সেটা return করবে। (যেকোন একটা মিলে গেলেই সেগুলো দিয়ে দিবে)



```
* practice:test@localhost x
localhost > practice
1 db.test.find({ age: { $gte: 18, $lte: 30 }, { age: 1 }}).sort({ age: 1 })
```

বয়স ১৮ থেকে ৩০ এর মধ্যে এইভাবে filter করা যাবে। একই field হলে কমা (,) দিয়ে Logical AND অপারেশন ব্যবহার করতে পারছি।

এই condition টিকে বলা হয় **implicit AND** .



```
14
15
16
17
18 | db.test.find({ gender: "Female", age: { $gte: 18, $lte: 30 } }, { age: 1, gender: 1 }).sort({ age: 1 })
19

test | 0.059 s | 9 Docs
1 /* 1 createdAt:3/7/2023, 9:20:04 AM*/
2 {
3   "_id" : ObjectId("6406ad64fc13ae5a4000006d"),
4   "gender" : "Female",
5   "age" : 18
6 },
7
8 /* 2 createdAt:3/7/2023, 9:20:03 AM*/
```

এইখানে বয়স ১৮ থেকে ৩০ এর মধ্যে এবং gender হচ্ছে Female .

এই condition টি ও **implicit AND** .

কমা (,) দ্বারা implicit AND করলে সেটা Logical AND এর মতো কাজ করবে যেগুলো মিলবে শুধুমাত্র সেগুলোই রিটুর্ন করবে।

😎 এখন আমি চাই তাদের খুজতে যাদের বয়স জোড় সংখ্যাক। যেমন ১৮,২০ ইত্যাদি।

সেক্ষেত্রে আমরা \$in ব্যবহার করতে পারবো।

```
3
4 - db.test.find(
5     { gender: "Female", age: { $in: [18, 20] } },
6     { age: 1, gender: 1 }
7     .sort({ age: 1 })
```

test | 0.040 s | 1 Doc

```
1 - [
2     "_id" : ObjectId("6406ad64fc13ae5a4000006d"),
3     "gender" : "Female",
4     "age" : 18
5 ]
```

১৮,২০ value একটা ডাটার সাথে মিলেছে সেটা দিয়েছে

```
1
2
3
4- db.test.find(
5   { gender: "Female", age: { $in: [18, 20, 22, 24, 26, 28, 30] } },
6   { age: 1, gender: 1 }
7   .sort({ age: 1 })

test | 0.036s | 3 Docs
200 | p 1 | 1-3 | JSON

1 /* 1 createdAt:3/7/2023, 9:20:04 AM*/
2 {
3   "_id" : ObjectId("6406ad64fc13ae5a4000006d"),
4   "gender" : "Female",
5   "age" : 18
6 },
7
8 /* 2 createdAt:3/7/2023, 9:20:04 AM*/
9 {
10  "_id" : ObjectId("6406ad64fc13ae5a40000082"),
11  "gender" : "Female",
12  "age" : 24
13 },
14
15 /* 3 createdAt:3/7/2023, 9:20:05 AM*/
16 {
17  "_id" : ObjectId("6406ad65fc13ae5a400000b0"),
18  "gender" : "Female",
19  "age" : 26
20 }
```

\$nin

\$nin হচ্ছে \$in এর বিপরীত।

```
2
3
4- db.test.find(
5    { gender: "Female", age: { $nin: [18, 20, 22, 24, 26, 28, 30] } },
6    { age: 1, gender: 1 })
7        .sort({ age: 1 })
```

যেগুলো রিটার্ন করবে সেগুলোতে একটাও age ভালু গুলোর সাথে মিলবে না।

▼ Video 5 (\$and, \$or, implicit vs explicit)

Documentation : <https://www.mongodb.com/docs/manual/reference/operator/query-logical/>

```
1 db.test.find({ age: { $ne: 15 }, age:{$lte:30} })
```

এটা লিখলে Error দিবে Duplicate Identifier .

কারণঃ একই field এ কাজ করার সময় একই field এর উপরে আলাদা ভাবে implicit AND (,) করে ব্যবহার করা যাবে না ।

সেক্ষেত্রে একই bracket এর মধ্যে নিয়ে যেতে হবে ।

solution :

```
db.test.find({ age: { $ne: 15, $lte: 30 } })
```

```
1 db.test.find({ age: { $ne: 15, $lte: 30 } })
2
3
```

test | 0.044 s | 31 Docs

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/

আর একটা পদ্ধতি আছে সেটা হচ্ছে Explicid AND.

\$and ⇒ Explicid AND

syntax :

```
{ $and: [ { <expression1> }, { <expression2> } , ... , { <expressionN> } ] }
```

এখনঃ

```
db.test.find({ age: { $ne: 15, $lte: 30 } })
```

এই কাজটি Explicid AND এর মাধ্যমে

```
db.test.find({
  $and: [
    { age: { $ne: 15 } },
    { age: { $lte: 30 } }
```

```
        ]  
    })
```

\$or ⇒ Explicit OR

```
db.test.find({  
  $or: [  
    {interests:"Travelling"},  
    {interests:"Cooking"},  
  ]  
}).project({interests:1}).sort({ age: 1 })
```

এক্ষেত্রে যেকোন একটা Condition সত্য হলেই সেটি return করবে ।

```
256   "friends" : [ "Mizanur Rahman", "Abdur Rakib", "Najmus Sakib", "Mir Hussain", "Mezbaul Abedin" ],  
257   "occupation" : "VP Sales",  
258   "interests" : [ "Gaming", "Cooking", "Writing" ],  
259   "skills" : [  
260     {  
261       "name" : "JAVASCRIPT",  
262       "level" : "Intermediate",  
263       "isLearning" : false  
264     },  
265     {  
266       "name" : "C#",  
267       "level" : "Beginner",  
268       "isLearning" : false  
269     },  
270     {
```

Skills এর উপর যদি query করতে চাই তাহলেঃ "skills.name" এইভাবে access করতে হবে ।

The screenshot shows the MongoDB Compass interface. In the top-left, there's a code editor with the following query:

```
1 // db.test.find({  
2   age: { $ne: 15, $lte: 30 } })  
3 - db.test.find({  
4   $or: [  
5     {"skills.name": "JAVASCRIPT"},  
6     {"skills.name": "PYTHON"}  
7   ]  
8 }).project({skills:1}).sort({ age: 1 })  
10
```

A red arrow points to the "\$or" operator in line 4. In the bottom-right, the "Results" tab displays the query results in JSON format:

```
4 {  
5   "SKILLS": [  
6     {  
7       "name": "RUBY",  
8       "level": "Beginner",  
9       "isLearning": true  
10      },  
11      {  
12        "name": "JAVASCRIPT",  
13        "level": "Expert",  
14        "isLearning": false  
15      }  
16    ],  
17    "age": 20,  
18    "friends": [  
19      "Mizanur Rahman",  
20      "Abdur Rakib",  
21      "Najmus Sakib",  
22      "Mir Hussain",  
23      "Mezbaul Abedin"  
24    ],  
25    "interests": [  
26      "Gaming",  
27      "Cooking",  
28      "Writing"  
29    ],  
30    "occupation": "VP Sales"  
31  }  
32
```

এই কাজটি যদি Implicit OR দিয়ে করতে চাই তাহলেঃ

```

13
14
15
16
17 db.test.find({ "skills.name": { $in: ["JAVASCRIPT", "PYTHON"] } }).project({ skills: 1 }).sort({ age: 1 })

```

test | 0.039 s | 30 Docs | 200 | 1-30 JSON

21 "skills" : []

এইখানে আমার \$not আর \$nor নিজে থেকে বুঝে অ্যাড করতে হবে

▼ Video 6 (\$exists, \$type, \$size)

Element Query Operator

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/query-element/>

\$exists

কোন একটি field কোন একটি document এ exist করে কিনা সেটা দেখাবে।

syntax:

```
{ field: { $exists: <boolean> } }
```

value হিসেবে true , false যেকোনো টি দেওয়া যাবে।

সেগুলো ডাটা দেখাও যেগুলোতে age আছে। তাহলেঃ

```

1
2 db.test.find({ age: { $exists: true } })

```

test | 0.052 s | 99 Docs | 200

1 /* 1 createdAt:3/7/2023, 9:20:03 AM */

সেগুলো ডাটা দেখাও যেগুলোতে age নেই। তাহলেঃ

```

1
2 db.test.find({ age: { $exists: false } })

```

test | 0.035 s | 0 Doc | 200

1 []

এখন ,

_id	postalCode	company	favouriteColor	friends	oc
1 6406ad63fc13ae5a40000065		null	Aquamarine	Array[5]	Fo
2 6406ad63fc13ae5a40000066	830-247	Aivee	Violet	Array[5]	Se
3 6406ad63fc13ae5a40000067		Bubblebab	Maroon	Array[4]	Re
4 6406ad63fc13ae5a40000068	4150-000	Twimho	Indigo	Array[5]	Ad

একটি company এর field null আছে।

```

localhost practice
1
2 db.test.find({ company: { $exists: false } })

```

test | 0.040 s | 0 Doc

```

1 []

```

কিন্তু এই ক্ষেত্রে exists operator 0 return করে।

exists operator null, undefined এবং empty array [] বের করে দেয় না।

\$type

কোন কিছুর টাইপ বের করার জন্য।

_id	gender	age	birthday	street	city
1 6406ad63fc13ae5a40000065	Female	42	3/13/2022	1188 Lerdahl Point	Dongxi
2 6406ad63fc13ae5a40000066	Male	12	9/2/2022	47 Hanover Trail	Léguá
3 6406ad63fc13ae5a40000067	Male	38	9/7/2022	54 Shasta Center	Qandala
4 6406ad63fc13ae5a40000068	Male	65	10/21/2022	94975 Harbor Street	Malas Olímpio
5 6406ad63fc13ae5a40000069	Female	70	1/14/2023	070 Clyde Gallagher Alley	Santa Magdalena
6 6406ad63fc13ae5a4000006a	Female	48	12/22/2022	08126 Blaine Street	Marina Roshcha
7 6406ad63fc13ae5a4000006b	Male	18	9/30/2022	9233 Artisan Terrace	Aniana

ধরি এই age এর ডাটাটি string ফরম্যাট এ আছে বা বানিয়ে নিলাম।

এখন আমি যদি চাই নির্দিষ্ট কোন টাইপ এর ডাটা নিয়ে আসতে তাহলে \$type operator ব্যবহার করতে হবে।

```

localhost practice
1
2 db.test.find({ age: { $type: "string" } }, { age: 1 })

```

test | 0.042 s | 2 Docs

```

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3   "_id" : ObjectId("6406ad63fc13ae5a40000065"),
4   "age" : "42"
5 },
6
7 /* 2 createdAt:3/7/2023, 9:20:05 AM*/
8 {
9   "_id" : ObjectId("6406ad65fc13ae5a400000c6"),
10  "age" : "45"
11 }

```

তাহলে সে যেগুলো ডাটা তে age এর value string type এর সেগুলো রিটার্ন করবে।

```

* practice:test@localhost | practice.test@localhost (1) *
localhost <--> practice
1
2 db.test.find({ company: { $type: "null" } }, { company: 1 })

test | 0.035 s | 2 Docs
1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3     "_id" : ObjectId("6406ad63fc13ae5a40000065"),
4     "company" : null
5 },
6
7 /* 2 createdAt:3/7/2023, 9:20:05 AM*/
8 {
9     "_id" : ObjectId("6406ad65fc13ae5a400000c7"),
10    "company" : null
11 }

```

company এর value null সেক্ষেত্রে \$type operator ব্যবহার করতে হবে।

empty array [] যাচাই করতে চাইলে \$size operator ব্যবহার করতে হবে।

\$size

documentation: <https://www.mongodb.com/docs/manual/reference/operator/query/size/>

syntax :

```
db.collection.find( { field: { $size: value নাম্বার এ } } );
```

যদি value 0 দেই তাহলে সেটা empty array

```

localhost <--> practice
1
2 db.test.find({ friends: { $size: 4 } }, { friends: 1 })

test | 0.037 s | 41 Docs
1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3     "_id" : ObjectId("6406ad63fc13ae5a40000067"),
4     "friends" : [ "Mir Hussain", "Abdur Rakib", "Najmus Sakib", "Rasel Ahmed" ]
5 },
6
7 /* 2 createdAt:3/7/2023, 9:20:03 AM*/
8 {
9     "_id" : ObjectId("6406ad63fc13ae5a4000006b"),
10    "friends" : [ "Jhankar Mahbub", "Rasel Ahmed", "Tanmoy Parvez", "Mezbaur Abedin" ]
11 }

```

যেসব array এর সাইজ 4 সেগুলো দেখাবে।

সেক্ষেত্রে বলা যায় \$size operator শুধুমাত্র Array এর ক্ষেত্রে।

▼ Video 7 (\$all , \$elemMatch)

```

15     "city" : "Dongxit",
16     "country" : "China"
17   },
18   "company" : null,
19   "favouriteColor" : "Aquamarine",
20   "friends" : [ ],
21   "occupation" : "Food Chemist",
22   "interests" : [ "Cooking", "Writing", "Reading" ],
23   "skills" : [
24     {
25       "name" : "JAVASCRIPT",
26       "level" : "Expert",
27       "isLearning" : false
28     },
29     {
30       "name" : "C#",
31       "level" : "Expert",
32       "isLearning" : true

```

interests object এর মধ্যে Element হিসেবে Cooking আছে।

এখন যদি চাই আমাকে এমন Document গুলো দাও যেগুলোর মধ্যে interests হিসেবে Cooking থাকবে।

The screenshot shows the MongoDB Compass interface. In the top-left, there's a search bar and a navigation bar with tabs like Run, Debug, Stop, Import, Export, Monitoring, Tasks, DataGen, and Schema. Below that, it says "localhost > practice". The main area has two tabs: "Query" and "Results". The "Query" tab contains the following code:

```
1 db.test.find({ interests: "Cooking" }).project({ interests: 1 })
```

An arrow points from the text "interests object এর মধ্যে Element হিসেবে Cooking আছে।" to the word "Cooking" in the query. The "Results" tab shows the output of the query:

```

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3   "_id" : ObjectId("6406ad63fc13ae5a40000065"),
4   "interests" : [ "Cooking", "Writing", "Reading" ]
5 },
6
7 /* 2 createdAt:3/7/2023, 9:20:03 AM*/
8 {
9   "_id" : ObjectId("6406ad63fc13ae5a40000069"),
10  "interests" : [ "Gaming", "Cooking", "Writing" ]
11 },
12
13 /* 3 createdAt:3/7/2023, 9:20:03 AM*/
14 {
15   "_id" : ObjectId("6406ad63fc13ae5a4000006a"),
16   "interests" : [ "Gardening", "Gaming", "Cooking" ]

```

output এ কিছু কিছু cooking আছে last এর দিকে।

ধরি লাস্ট এর দিকে যেসব Cooking আছে সেগুলো আমার দরকার।

সেক্ষেত্রে position হিসাব করতে হবে।

আমারা array এর পজিশন হিসাব করি index এর মাধ্যমে।

২ নাম্বার index এ যেসব ডাটার cooking আছে সেগুলো দেখতে চাইলে

The screenshot shows the MongoDB Compass interface. In the top-left, there's a search bar and a navigation bar with tabs like Run, Debug, Stop, Import, Export, Monitoring, Tasks, DataGen, and Schema. Below that, it says "localhost > practice". The main area has two tabs: "Query" and "Results". The "Query" tab contains the following code:

```
1 db.test.find({ "interests.2": "Cooking" }).project({ interests: 1 })
```

An arrow points from the text "২ নাম্বার index এ যেসব ডাটার cooking আছে সেগুলো দেখতে চাইলে" to the index number "2" in the query. The "Results" tab shows the output of the query:

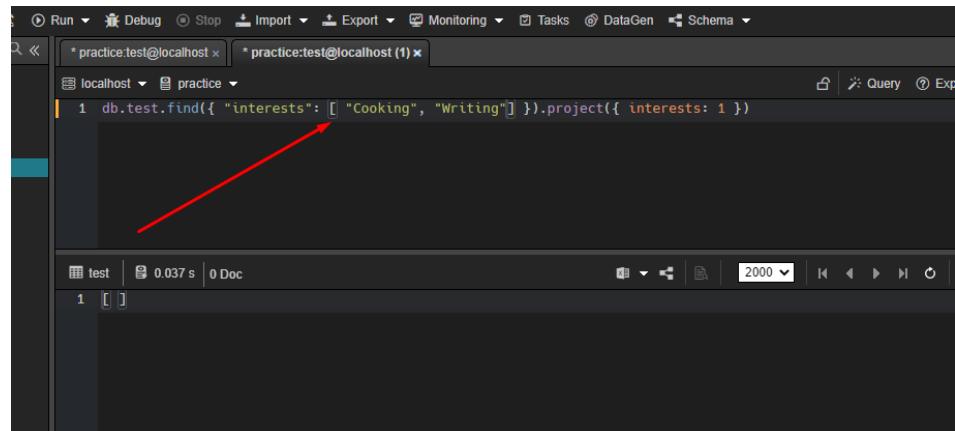
```

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3   "_id" : ObjectId("6406ad63fc13ae5a4000006a"),
4   "interests" : [ "Gardening", "Gaming", "Cooking" ]
5 },
6
7 /* 2 createdAt:3/7/2023, 9:20:04 AM*/
8 {
9   "_id" : ObjectId("6406ad64fc13ae5a40000075"),
10  "interests" : [ "Travelling", "Reading", "Cooking" ]
11 },

```

যেহেতু ডট (.) syntax ব্যবহার করেছি তাই কোটেশন এর মধ্যে নিতে হবে ।

এখন ধরি Cooking এর সাথে Gardening লাগবে ।



A screenshot of the MongoDB Compass interface. The top navigation bar shows 'Run', 'Debug', 'Stop', 'Import', 'Export', 'Monitoring', 'Tasks', 'DataGen', and 'Schema'. Below the navigation is a search bar with 'localhost' and 'practice' selected. The main area has a code editor with the following query:

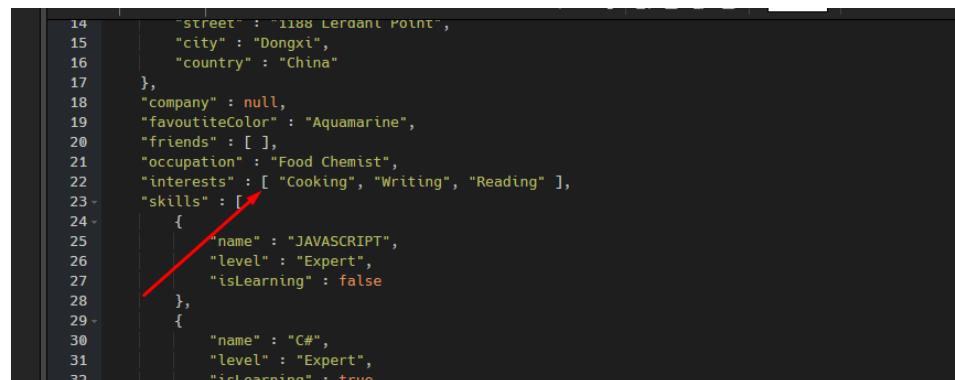
```
1 db.test.find({ "interests": [ "Cooking", "Writing" ] }).project({ interests: 1 })
```

The line '1 db.test.find({ "interests": ["Cooking", "Writing"] })' is highlighted with a red arrow pointing to it.

At the bottom, the results pane shows 'test' collection, 0.037s execution time, 0 documents found, and a result table with one row containing an empty array: [].

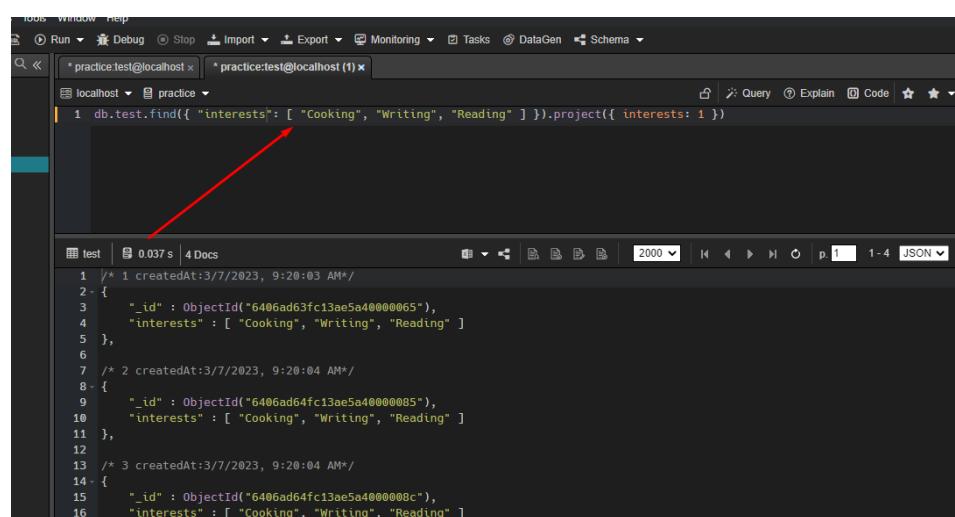
এইভাবে লিখলে হবে না ।

কারণঃ



A screenshot of a JSON document. The document contains several fields like street, city, country, company, favoriteColor, friends, occupation, interests, and skills. The 'skills' field is a nested array with two objects. A red arrow points to the 'skills' field.

```
14     "street" : "1188 Lerdant Point",
15     "city" : "Dongxi",
16     "country" : "China"
17 },
18 "company" : null,
19 "favoriteColor" : "Aquamarine",
20 "friends" : [ ],
21 "occupation" : "Food Chemist",
22 "interests" : [ "Cooking", "Writing", "Reading" ],
23 "skills" : [
24     {
25         "name" : "JAVASCRIPT",
26         "level" : "Expert",
27         "isLearning" : false
28     },
29     {
30         "name" : "C#",
31         "level" : "Expert",
32         "isLearning" : true
33     }
34 ]
```



A screenshot of MongoDB Compass showing a query in the Query tab. The query is:

```
1 db.test.find({ "interests": [ "Cooking", "Writing", "Reading" ] }).project({ interests: 1 })
```

The line '1 db.test.find({ "interests": ["Cooking", "Writing", "Reading"] })' is highlighted with a red arrow pointing to it.

At the bottom, the results pane shows 'test' collection, 0.037s execution time, 4 documents found, and a result table with four rows, each containing an array of three elements: ["Cooking", "Writing", "Reading"].

Array এর মধ্যে দিলে পজিশন এবং value একই হতে হবে তবেই তা রিটার্ন করবে ।

কিন্তু আমার দরকার "Travelling", "Gaming", "Reading" এই তিনটি থাকলেই হবে যে কোন পজিশন এ বসুক না কেন।

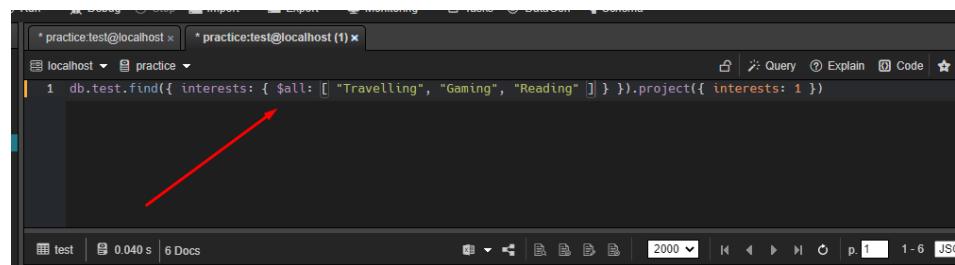
সেক্ষেত্রে \$all operator ব্যবহার করতে হবে।

documentation : <https://www.mongodb.com/docs/manual/reference/operator/query-array/>

\$all operator

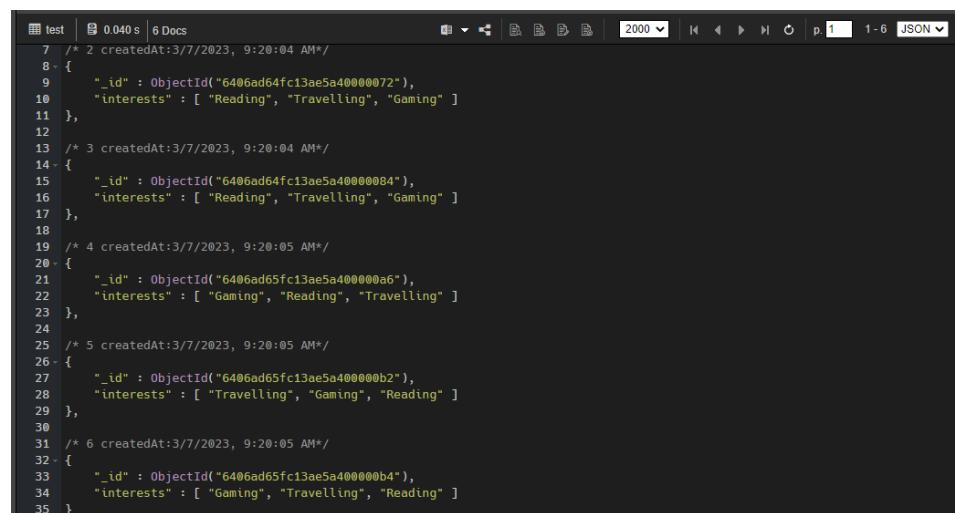
syntax:

```
{ <field>: { $all: [ <value1> , <value2> ... ] } }
```



```
1 db.test.find({ interests: { $all: [ "Travelling", "Gaming", "Reading" ] } }).project({ interests: 1 })
```

Output :



```
7 /* 2 createdAt:3/7/2023, 9:20:04 AM*/
8 {
9     "_id" : ObjectId("6406ad64fc13ae5a40000072"),
10    "interests" : [ "Reading", "Travelling", "Gaming" ]
11 },
12
13 /* 3 createdAt:3/7/2023, 9:20:04 AM*/
14 {
15     "_id" : ObjectId("6406ad64fc13ae5a40000084"),
16    "interests" : [ "Reading", "Travelling", "Gaming" ]
17 },
18
19 /* 4 createdAt:3/7/2023, 9:20:05 AM*/
20 {
21     "_id" : ObjectId("6406ad65fc13ae5a400000a6"),
22    "interests" : [ "Gaming", "Reading", "Travelling" ]
23 },
24
25 /* 5 createdAt:3/7/2023, 9:20:05 AM*/
26 {
27     "_id" : ObjectId("6406ad65fc13ae5a400000b2"),
28    "interests" : [ "Travelling", "Gaming", "Reading" ]
29 },
30
31 /* 6 createdAt:3/7/2023, 9:20:05 AM*/
32 {
33     "_id" : ObjectId("6406ad65fc13ae5a400000b4"),
34    "interests" : [ "Gaming", "Travelling", "Reading" ]
35 }
```

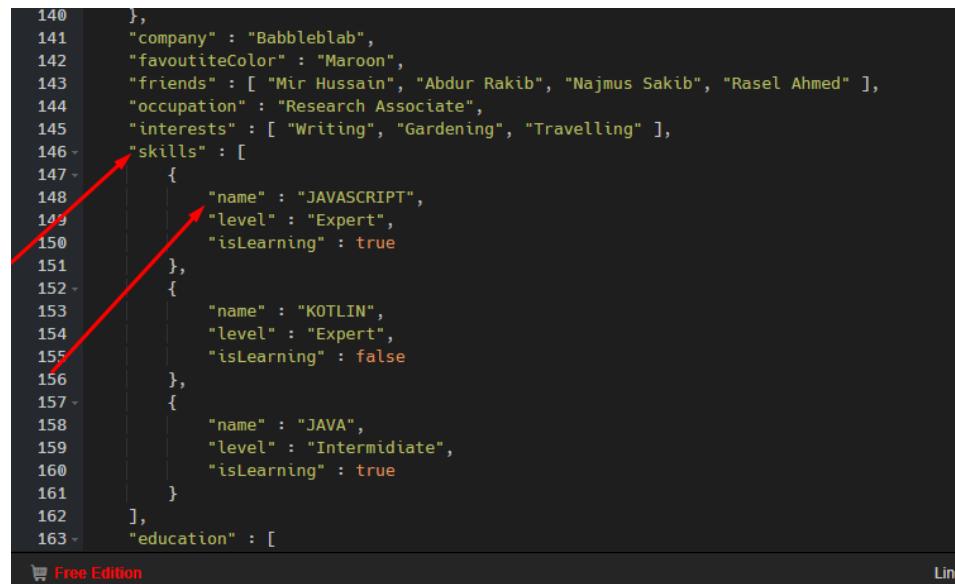
এক্ষেত্রে এই তিনটি থাকলেই হবে যেকোন পজিশন এ থাকতে পারে কোন সমস্যা নাই।

Array এর মতো করে Object ও Find করতে পারি।

```

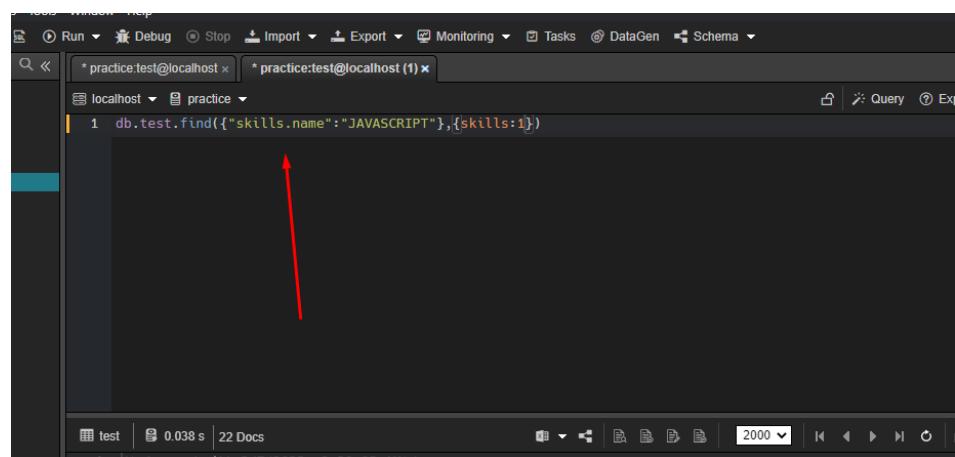
140 },
141 "company" : "Babbleblab",
142 "favouriteColor" : "Maroon",
143 "friends" : [ "Mir Hussain", "Abdur Rakib", "Najmus Sakib", "Rasel Ahmed" ],
144 "occupation" : "Research Associate",
145 "interests" : [ "Writing", "Gardening", "Travelling" ],
146 "skills" : [
147   {
148     "name" : "JAVASCRIPT",
149     "level" : "Expert",
150     "isLearning" : true
151   },
152   {
153     "name" : "KOTLIN",
154     "level" : "Expert",
155     "isLearning" : false
156   },
157   {
158     "name" : "JAVA",
159     "level" : "Intermediate",
160     "isLearning" : true
161   }
162 ],
163 "education" : [

```



skills এর মধ্যে name হিসেবে JAVASCRIPT আছে।

এখন আমি Javascript কে find করতে চাই।



```

db.test.find({skills.name:'JAVASCRIPT'}, {skills:1})

```

এইভাবে লিখতে হবে।

এখন ধরি আমার JAVASCRIPT দরকার আর level দরকার Intermediate .

```

db.test.find({"skills.name": "JAVASCRIPT", "skills.level": "Intermediate"}, {"skills:1"})

```

test | 0.036 s | 16 Docs

```

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3   "_id" : ObjectId("6406ad63fc13ae5a40000067"),
4   "skills" :
5   [
6     {
7       "name" : "JAVASCRIPT",
8       "level" : "Expert",
9       "isLearning" : true
10    },
11    {
12      ...
13    }
14  ]
15 }
16 [
17   {
18     "name" : "KOTLIN",
19     "level" : "Expert",
20     "isLearning" : false
21   },
22   {
23     ...
24   }
25 ],
26 /* 2 createdAt:3/7/2023, 9:20:03 AM*/

```

এইভাবে লিখলেও আসবে কিন্তু একটা সমস্যা আছে।

দেখা যাবে level:Expert ও দিয়ে দিচ্ছে তার মানে ঠিক মতো কাজ করে নাই।

```

db.test.find(
  {
    skills: {
      name: "JAVASCRIPT",
      level: "Intermediate",
      isLearning: true
    }
  }, {"skills:1"})

```

test | 0.036 s | 16 Docs

```

11 [
12   {
13     "name" : "KOTLIN",
14     "level" : "Expert",
15     "isLearning" : false
16   },
17   {
18     ...
19   }
20 ],
21 /* 2 createdAt:3/7/2023, 9:20:03 AM*/

```

এইভাবে দিলে Array এর মতো সমস্যা। index position ঠিক থাকতে হবে তবেই আসবে।

```

db.test.find(
  {
    skills: [
      {
        name: "JAVASCRIPT",
        level: "Intermediate",
        isLearning: true
      }
    ]
  }, {"skills:1"})

```

test | 0.041 s | 3 Docs

```

1 /* 1 createdAt:3/7/2023, 9:20:04 AM*/
2 [
3   {
4     "_id" : ObjectId("6406ad64fc13ae5a40000097"),
5     "skills" :
6     [
7       {
8         "name" : "C",
9         "level" : "Intermediate",
10        "isLearning" : true
11      },
12      {
13        ...
14      }
15    ],
16    ...

```

যদিও isLearning:true এটা তুলে দেই তাহলেঃ

A screenshot of the NoSQLBooster interface for MongoDB. The left sidebar shows a database named 'practice' with a single collection 'test'. The main panel displays a query in the 'Query' tab:

```
localhost > db.test.find({  
  skills:{  
    name:"JAVASCRIPT",  
    level:"Intermediate"  
  },  
}, {skills:1})
```

The results pane at the bottom shows one document:

test	0.036 s 0 Doc
1 []	

Two red arrows point from the text 'যদিও isLearning:true এটা তুলে দেই তাহলেঃ' to the 'isLearning:true' part of the query.

A screenshot of the NoSQLBooster interface for MongoDB. The left sidebar shows a database named 'practice' with a single collection 'test'. The main panel displays a query in the 'Query' tab:

```
localhost > db.test.find({  
  skills:{  
    name:"JAVASCRIPT",  
    isLearning:true,  
    level:"Intermediate",  
  },  
}, {skills:1})
```

The results pane at the bottom shows one document:

test	0.037 s 0 Doc
1 []	

Two red arrows point from the text 'উপর নিচে করলেও হবে না।' to the 'isLearning:true' part of the query.

উপর নিচে করলেও হবে না।

এই সমস্যা সমাধান করার জন্য **\$elemMatch operator** ব্যবহার করতে হবে।

\$elemMatch

documentation :

<https://www.mongodb.com/docs/manual/reference/operator/query/elemMatch/>

syntax :

```
{ <field>: { $elemMatch: { <query1>, <query2>, ... } } }
```

Solution :

The screenshot shows the MongoDB Compass interface with a query results pane. The query is:

```

1 db.test.find({
2   skills: {
3     $elemMatch: { name: "JAVASCRIPT", level: "Intermediate" }
4   }, { skills: 1 }
5 })

```

The results pane displays 10 documents. One document is expanded, showing:

```

1 /* 1 createdAt:3/7/2023, 9:28:03 AM*/
2 {
3   "_id": ObjectId("6406ad63fc13ae5a40000069"),
4   "skills": [
5     {
6       "name": "JAVASCRIPT",
7       "level": "Intermediate",
8       "isLearning": false
9     },
10    {
11      "name": "C#",
12      "level": "Beginner",
13      "isLearning": false
14    },
15  ]
}

```

▼ Video 8 (\$set, \$addToSet, \$push)

এখন আপডেট অপারেটর সম্পর্কে জানবো।

The screenshot shows the MongoDB Compass interface with a query results pane. The query is:

```

1 db.test.find({})
2

```

The results pane displays 99 documents. One document is expanded, showing:

```

1 /* 1 createdAt:3/7/2023, 9:20:03 AM*/
2 {
3   "_id": ObjectId("6406ad63fc13ae5a40000065"),
4   "name": {
5     "firstName": "Marielle",
6     "lastName": "Dangl"
7   },
8   "email": "mdangl1@odnoklassniki.ru",
9   "phone": "(167) 7775715",
10  "gender": "Female",
11  "age": "42",
12  "birthday": "3/13/2022",
13  "address": {
14    "street": "1188 Lerdahl Point",
15    "city": "Dongxi",
16    "country": "China"
}

```

ধরি এই ডকুমেন্ট টি আপডেট করবো।

The screenshot shows the MongoDB Compass interface with a query results pane. The query is:

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065" ) })
2

```

The results pane displays 1 document. One document is expanded, showing:

```

1 {
2   "_id": ObjectId("6406ad63fc13ae5a40000065"),
3   "name": {
4     "firstName": "Marielle",
5     "lastName": "Dangl"
6   },
7   "email": "mdangl1@odnoklassniki.ru",
8   "phone": "(167) 7775715",
9   "gender": "Female",
10  "age": "42",
11  "birthday": "3/13/2022",
12  "address": {
13    "street": "1188 Lerdahl Point",
14    "city": "Dongxi",
15    "country": "China"
}

```

`_id` দিয়ে `find` করে নিয়ে আসলাম।

ধরি `age` কে আপডেট করবো।

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/update-field/>

Docs Home / MongoDB Manual / Reference / Operators / Update

Field Update Operators

NOTE
For details on a specific operator, including syntax and examples, click on the link to the operator's reference page.

\$set

syntax:

```
{ $set: { <field1>: <value1>, ... } }
```

```
localhost:~$ practice
1 db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000065") },
3   {
4     $set: {
5       age: 60
6     }
7   }
8 )
9
10
```

0.028 s

```
1 [{}]
2   "acknowledged" : true,
3   "matchedCount" : 1,
4   "modifiedCount" : 1
5 ]
```

- "modifiedCount" : 1 অর্থ `Modify` হয়েছে।
- "matchedCount" : 1 অর্থ কোন ডকুমেন্ট এর সাথে ম্যাচ হয়েছে। কোন ডকুমেন্ট এর সাথে ম্যাচ হয়েছে কিনা বলে দেয়। যদি `value` 1 হয় তাহলে ম্যাচ হয়েছে।

`$set` এর কিছু সমস্যা রয়েছে।

যেমনঃ

```
test | 0.036 s | 1 Doc | 50 | p 1 | 1-1 | JSON
14     "city" : "wongxt",
15     "country" : "China"
16   },
17   "company" : null,
18   "favouriteColor" : "Aquamarine",
19   "friends" : [ ],
20   "occupation" : "Food Chemist",
21   "interests" : [ "Cooking", "Writing", "Reading" ], -----^
22   "skills" : [
23     {
24       "name" : "JAVASCRIPT",
25       "level" : "Expert",
26       "islearning" : false
27     },
28     {
29       "name" : "C#",
30       "level" : "Intermediate",
31       "islearning" : true
32     }
33   ]
34 }
```

এইখানে আছে Array , এখন আমি চাইতেছি interesses কে আপডেট করবো ।

The screenshot shows the MongoDB Compass interface with two tabs open: "localhost" and "practice". The "localhost" tab contains a query to update a document in the "test" collection. The query is as follows:

```
1 - db.test.updateOne(  
2 -   { _id: ObjectId("6406ad63fc13ae5a40000065") },  
3 -   {  
4 -     $set: {  
5 -       interests:"Gaming"  
6 -     }  
7 -   }  
8 - )  
9 -  
10 )
```

A red arrow points from the text "Gaming" in the query to the corresponding field in the response. The response is:

```
0.028 s  
1 - {  
2 -   "acknowledged" : true,  
3 -   "matchedCount" : 1,  
4 -   "modifiedCount" : 1  
5 - }
```

ধরি এইভাবে আপডেট করলাম।

The screenshot shows the MongoDB Compass interface. At the top, there are tabs for "localhost" and "practice". Below the tabs, a query is run:

```
db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
```

The results pane shows one document:

```
test { "_id": "6406ad63fc13ae5a40000065", "name": "Vongxi", "country": "China", "company": null, "favouriteColor": "Aquamarine", "friends": [], "occupation": "Food Chemist", "interests": "Gaming", "skills": [ { "name": "JAVASCRIPT", "level": "Expert", "isLearning": false } ] }
```

A red arrow points from the "interests" field in the query results back to the "interests" field in the JSON document.

এখন কিন্তু আর Array টা নাই।

- যখন কোন একটি Field কে আপডেট করবো সেটা যদি Primitive Type হয় (Primitive এর একটি value হয়) তাহলে \$set ব্যবহারে কোন সমস্যা নেই ।
 - Non Primitive (Array, Object Etc) এর ক্ষেত্রে \$set ব্যবহার করলে সম্পূর্ণ জিনিসটিকে পরিবর্তন করে দেয়

কিন্তু যদি মনে হয় সম্পূর্ণ Array টি পরিবর্তন হয়ে নতুন Array হবে তাহলে \$set এর ব্যবহার ঠিক আছে। যেমনঃ

The screenshot shows the MongoDB Compass interface. In the top-left query editor, a command is being run:

```
1 db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000065") },
3   {
4     $set: {
5       interests: ["Gaming", "Reading,Writing"]
6     }
7   }
8 )
9
10
```

A red arrow points from the text "Reading,Writing" in the update command to the corresponding value in the output window below.

In the bottom-right results window, the output is:

```
0.031 s
1 [
2   {
3     "acknowledged" : true,
4     "matchedCount" : 1,
5     "modifiedCount" : 1
6   }
7 ]
```

Output :

The screenshot shows the MongoDB Compass interface. In the top-left query editor, a command is being run:

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
```

A red arrow points from the "interests" field in the output window below to the value in the update command above.

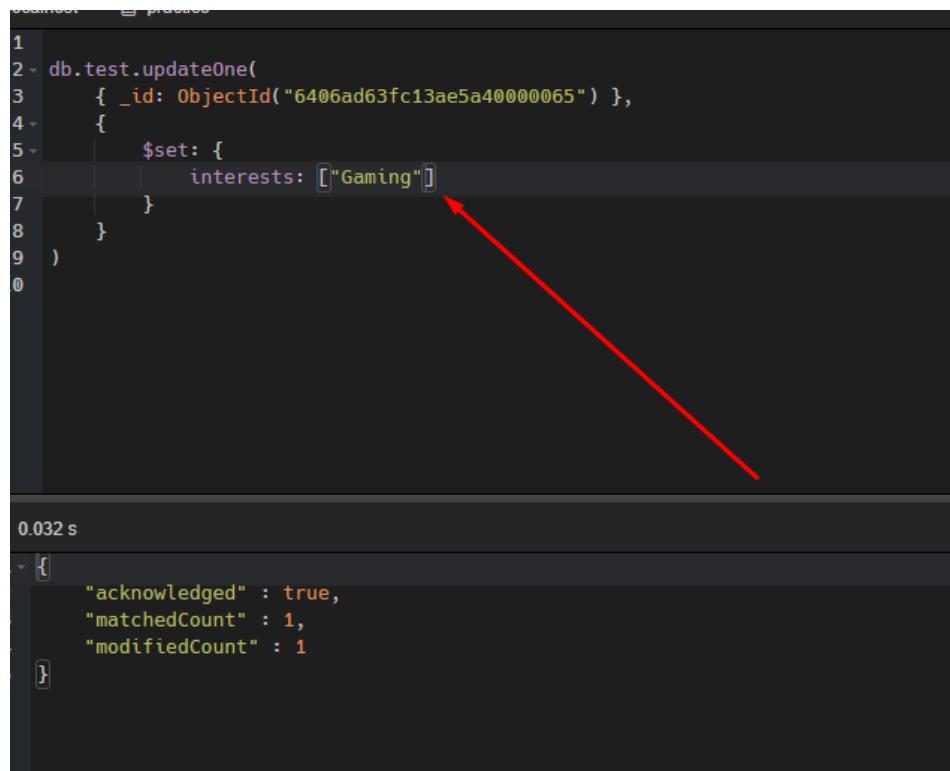
In the bottom-right results window, the output is:

```
test | 0.034 s | 1 Doc
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18   "favouriteColor" : "Aquamarine",
19   "friends" : [ ],
20   "occupation" : "Food Chemist",
21   "interests" : [ "Gaming", "Reading,Writing" ],
22   "skills" : [
23     {
24       "name" : "JAVASCRIPT",
25       "level" : "Expert",
26       "language" : "Frontend"
27     }
28   ]
29 }
```

কিন্তু যদি চাই নতুন value অ্যাড করতে সেক্ষেত্রে \$set ব্যবহার করলে কাজ হবে না।

তাহলে সম্পূর্ণ টা আপডেট করতে চাইলে \$set ব্যবহার করবো। Primitive এর ক্ষেত্রে কোন সমস্যা নেই।

এখন আমি চাচ্ছিঃ



```

1
2 db.test.updateOne(
3   { _id: ObjectId("6406ad63fc13ae5a40000065") },
4   {
5     $set: {
6       interests: ["Gaming"]
7     }
8   }
9 )
0

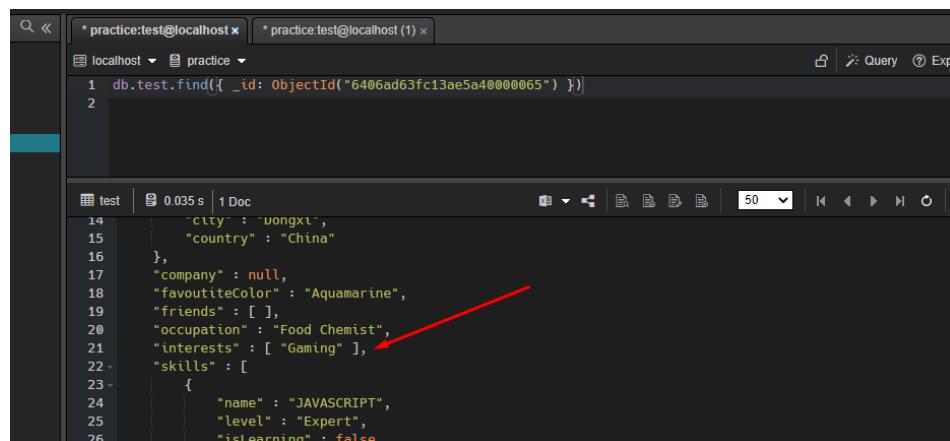
```

0.032 s

```

{
  "acknowledged" : true,
  "matchedCount" : 1,
  "modifiedCount" : 1
}

```



```

* practice:test@localhost x * practice:test@localhost (1) x
localhost <--> practice
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
2

```

test	0.035 s 1 Doc
14	"city" : "Dongxi",
15	"country" : "China"
16	},
17	"company" : null,
18	"favouriteColor" : "Aquamarine",
19	"friends" : [],
20	"occupation" : "Food Chemist",
21	"interests" : ["Gaming"],
22	"skills" : [
23	{
24	"name" : "JAVASCRIPT",
25	"level" : "Expert",
26	"isLearning" : false

এইখানে নতুন value অ্যাড করতে।

সেক্ষেত্রে ব্যবহার করতে হবে Array Update Operator .

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/update-array/>

\$addToSet Operator

Documentation:

<https://www.mongodb.com/docs/manual/reference/operator/update/addToSet/#mongodb-update-up.-addToSet>

syntax :

```
{ $addToSet: { <field1>: <value1>, ... } }
```

The screenshot shows the MongoDB shell interface. The command entered is:

```
1 db.test.updateOne
2   { _id: ObjectId("6406ad63fc13ae5a40000065") },
3   {
4     $addToSet: {
5       interests: "Writing"
6     }
7   }
8 )
9
10
```

The execution results are shown below:

```
0.031s
1 [
2   {
3     "acknowledged" : true,
4     "matchedCount" : 1,
5     "modifiedCount" : 1
6   }
7 ]
```

A red arrow points from the text '\$addToSet' in the command to the 'interests' field in the result document.

The screenshot shows the MongoDB shell interface. The command entered is:

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
2
```

The execution results are shown below:

```
test | 0.034 s | 1 Doc
11 {
12   "birthday" : "3/13/2022",
13   "address" : {
14     "street" : "1188 Lerdahl Point",
15     "city" : "Dongxi",
16     "country" : "China"
17   },
18   "company" : null,
19   "favouriteColor" : "Aquamarine",
20   "friends" : [ ],
21   "occupation" : "Food Chemist",
22   "interests" : [ "Gaming", "Writing" ],
23   "skills" : [
24     {
25       "name" : "JAVASCRIPT",
26       "level" : "Beginner"
27     }
28   ]
29 }
```

A red arrow points from the text '\$each' in the command to the 'interests' array in the result document.

\$addToSet Array এর মধ্যে value সেট করে দিবে Duplicate value নিবে না।

যদি চাই একাধিক value একসাথে যোগ করতে সেক্ষেত্রে \$each মডিফিয়ার ব্যবহার করতে হবে।

\$each

Documentation:

<https://www.mongodb.com/docs/manual/reference/operator/update/each/#mongodb-update-up-each>

syntax :

```
{ $addToSet: { <field>: { $each: [ <value1>, <value2> ... ] } } }
```

The screenshot shows the MongoDB Compass interface. In the top-left, there are two tabs: "practice: test@localhost" and "practice: test@localhost (1)". Below them, the database "localhost" and collection "practice" are selected. In the main area, a query is being run:

```
1 db.test.updateOne
2   { _id: ObjectId("6406ad63fc13ae5a40000065") },
3   {
4     $addToSet: {
5       interests: { $each: ["CopyWriter", "StoryWriter"] }
6     }
7   }
8 )
9
10
```

Below the query, the execution results are shown:

```
0.033 s
1 [
2   {
3     "acknowledged" : true,
4     "matchedCount" : 1,
5     "modifiedCount" : 1
6   }
7 ]
```

A red arrow points from the text "মনে হয় আমি Duplicate Entry দিবো সে ক্ষেত্রে \$push ব্যবহার করতে হবে।" to the line "interests: { \$each: ["CopyWriter", "StoryWriter"] }".

The screenshot shows the MongoDB Compass interface. In the top-left, there are two tabs: "practice: test@localhost" and "practice: test@localhost (1)". Below them, the database "localhost" and collection "practice" are selected. In the main area, a query is being run:

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
```

Below the query, the results are shown:

```
test | 0.057 s | 1 Doc
1
18   "favouriteColor" : "Aquamarine",
19   "friends" : [ ],
20   "occupation" : "Food Chemist",
21   "interests" : [ "Gaming", "Wrting", "CopyWriter", "StoryWriter" ],
22   "skills" : [
23     {
24       "name" : "JAVASCRIPT",
25       "level" : "Expert",
26       "isLearning" : false
27     },
28   ]
```

A red arrow points from the text "মনে হয় আমি Duplicate Entry দিবো সে ক্ষেত্রে \$push ব্যবহার করতে হবে।" to the line "interests" : ["Gaming", "Wrting", "CopyWriter", "StoryWriter"].

যদি মনে হয় আমি Duplicate Entry দিবো সে ক্ষেত্রে \$push ব্যবহার করতে হবে।

\$push

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/update/push/>

The screenshot shows the MongoDB Compass interface. In the top-left panel, there are two tabs: "practice:tesl@localhost" and "practice:tesl@localhost (1)". The right-hand panel displays the results of an update operation. The command is:

```

1 db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000065") },
3   [
4     {
5       $push: {
6         interests: { $each: ["CopyWriter", "StoryWriter"] }
7       }
8     }
9   ]
10

```

The result of the update is shown below, indicating a successful operation:

```

0.036 s
1 {
2   "acknowledged" : true,
3   "matchedCount" : 1,
4   "modifiedCount" : 1
5 }

```

The screenshot shows the MongoDB Compass interface. In the top-left panel, there are two tabs: "practice:tesl@localhost" and "practice:tesl@localhost (1)". The right-hand panel displays the results of a find operation. The command is:

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })
2

```

The result is a single document with the following fields and values:

```

test | 0.038 s | 1 Doc
18   "favouriteColor" : "Aquamarine",
19   "friends" : [ ],
20   "occupation" : "Food Chemist",
21   "interests" : [
22     "Gaming",
23     "Writing",
24     "CopyWriter",
25     "StoryWriter",
26     "CopyWriter",
27     "StoryWriter"
28   ],
29   "skills" : [
30     {
31       "name" : "JAVASCRIPT",
32       "level" : "Expert",
33       "isLearning" : false
34     }
35   ],

```

A red circle highlights the "interests" field, and a red arrow points from it to the "interests" field in the screenshot above.

▼ Video 9 (\$unset, \$pop, \$pull, \$pullAll)

DELETE OPERATION

\$unset

\$unset হচ্ছে \$set এর বিপরীত।

syntax:

```
{ $unset: { <field1>: "", ... } }
```

A screenshot of the MongoDB Compass interface. The top panel shows a query in the 'Query' tab: `db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000065") })`. The bottom panel shows the results in 'test':

```
1 - {
2 -   "_id": ObjectId("6406ad63fc13ae5a40000065"),
3 -   "name": {
4 -     "firstName": "Marielle",
5 -     "lastName": "Dang"
6 -   },
7 -   "email": "mdang1@odnoklassniki.ru",
8 -   "phone": "(167) 775715",
9 -   "gender": "Female",
10 -  "age": 60,
11 -  "birthday": "3/13/2022",
12 -  "address": {
13 -    "street": "1188 Lerdahl Point",
14 -    "city": "Dongxi",
15 -    "country": "China"
```

ধরি Birthday field টি ডিলিট করবো।

A screenshot of the MongoDB Compass interface. The top panel shows an update query in the 'Query' tab: `db.test.updateOne({ _id: ObjectId("6406ad63fc13ae5a40000065") }, { \$unset: { birthday: "" } })`.

The bottom panel shows the results: '0.032 s' and a response array:

```
1 - [
2 -   {
3 -     "acknowledged": true,
4 -     "matchedCount": 1,
5 -     "modifiedCount": 1
6 -   }
]
```

যদি age কে ডিলিট করতে চাই তাহলে?

```

* practice:test@localhost x * practice:test@localhost (1) x
localhost practice
1
2 ~ db.test.updateOne(
3     { _id: ObjectId("6406ad63fc13ae5a40000065") },
4     {
5         $unset: { age: 1 }
6     }
7 )

```

0.031 s

```

1 ~ [
2     "acknowledged" : true,
3     "matchedCount" : 1,
4     "modifiedCount" : 1
5 ]

```

"" অথবা 1 একই কথা। অর্থাৎ true .

এখন আমরা চাই Array থেকে কোন কিছু Remove করবো। তাহলেঃ

\$pop

\$pop এর কাজ হচ্ছে Element কে রিমুভ করে দেওয়া।

syntax:

```
{ $pop: { <field>: <-1 | 1>, ... } }
```

- 1 হচ্ছে last element remove করা।
- আর -1 হচ্ছে first element remove করা।
-

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2
3
4
5
6
7
8
9
10
11
12
13
14   "city" : "Legua",
15   "state" : "01",
16   "postalCode" : "3830-247",
17   "country" : "Portugal"
18 },
19   "company" : "Aivee",
20   "favouriteColor" : "Violet",
21   "friends" : [ "Fahim Ahmed Firoz", "Nahid Hasan Bulbul", "Tanmoy Parvez", "Mir Hussain", "Mezbaur Abedin" ],
22   "occupation" : "Senior Cost Accountant",
23   "interests" : [ "Travelling", "Gaming", "Reading" ],
24   "skills" : [
25     {
26       "name" : "RUBY",
27       "level" : "Expert",
28       "isLearning" : true
29     },
30     {
31       "name" : "KOTLIN",
32       "level" : "Expert",
33       "isLearning" : false
34     }
35   ],
36   "education" : [

```

ধরি এইটা রিমোভ করবো।

```

1 db.test.updateOne([
2   { _id: ObjectId("6406ad63fc13ae5a40000066") },
3   { $pop: { friends: 1 } }
4 ]
5 )

```

Last Element Remove

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2
3
4
5
6
7
8
9
10
11
12
13
14   },
15   "company" : "Aivee",
16   "favouriteColor" : "Violet",
17   "friends" : [ "Fahim Ahmed Firoz", "Nahid Hasan Bulbul", "Tanmoy Parvez", "Mir Hussain" ],
18   "occupation" : "Senior Cost Accountant",
19   "interests" : [ "Travelling", "Gaming", "Reading" ],
20   "skills" : [
21     {
22       "name" : "RUBY",
23       "level" : "Expert",
24       "isLearning" : true
25     }
26   ],
27   "education" : [

```

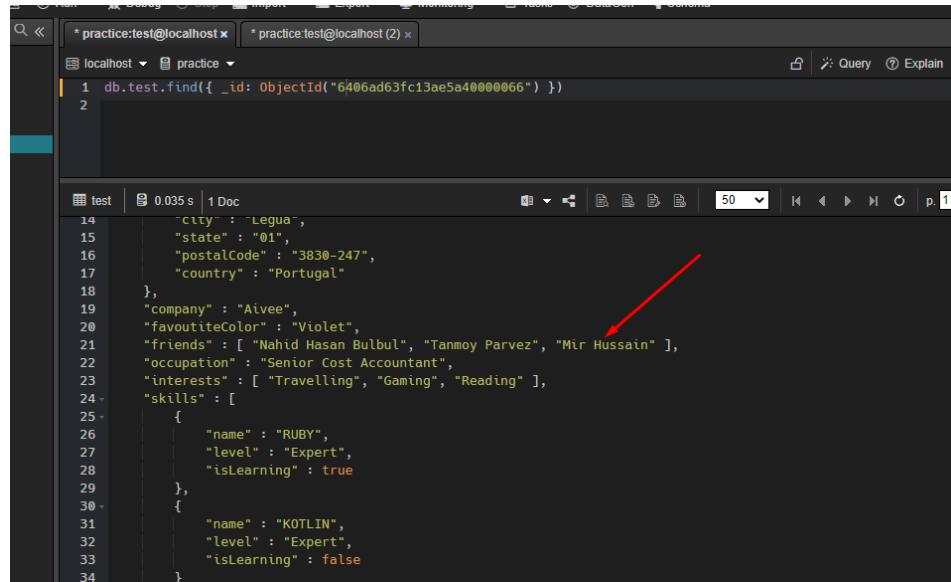
Remove Successfully

\$pull

\$pull ও \$pop এর মতোই।

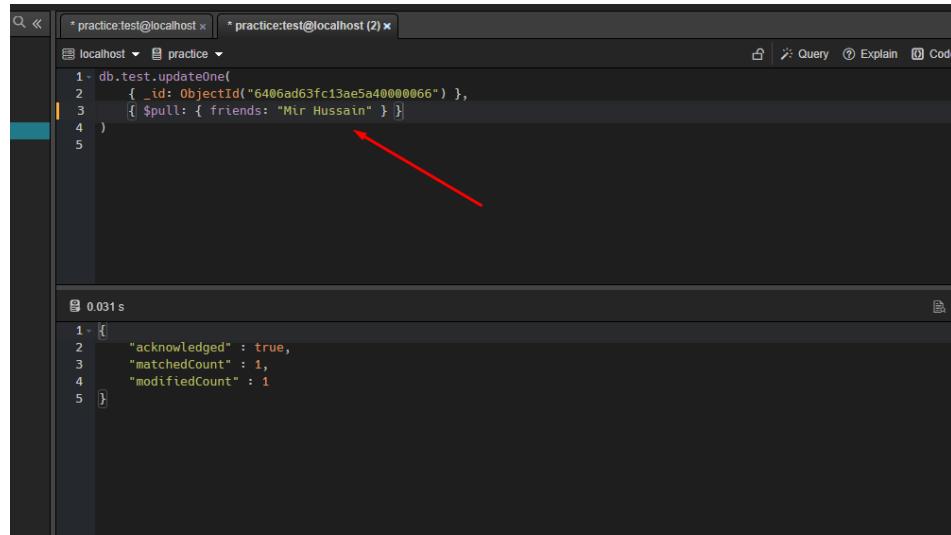
এটার কাজ কোন নির্দিষ্ট element কে পুল করে নিয়ে আসা।

documentation: <https://www.mongodb.com/docs/manual/reference/operator/update/pull/>



```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2
3
4
5
6
7
8
9
10
11
12
13
14     "city" : "Legua",
15     "state" : "01",
16     "postalCode" : "3830-247",
17     "country" : "Portugal"
18 },
19     "company" : "Aivee",
20     "favouriteColor" : "Violet",
21     "friends" : [ "Nahid Hasan Bulbul", "Tanmoy Parvez", "Mir Hussain" ],
22     "occupation" : "Senior Cost Accountant",
23     "interests" : [ "Travelling", "Gaming", "Reading" ],
24     "skills" : [
25         {
26             "name" : "RUBY",
27             "level" : "Expert",
28             "isLearning" : true
29         },
30         {
31             "name" : "KOTLIN",
32             "level" : "Expert",
33             "isLearning" : false
34     }
```

ধরি Mir Hussain কে pull করতে চাই তাহলেঃ



```
1 db.test.updateOne(
2     { _id: ObjectId("6406ad63fc13ae5a40000066") },
3     { $pull: { friends: "Mir Hussain" } }
4 )
5 
```

0.031s

```
1 { "acknowledged" : true,
2  "matchedCount" : 1,
3  "modifiedCount" : 1
4 }
```

```

* practice:test@localhost * practice.test@localhost (2) x
localhost practice
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2

test | 0.038 s | 1 Doc
18 },
19 "company" : "Aivee",
20 "favouriteColor" : "Violet",
21 "friends" : [ "Nahid Hasan Bulbul", "Tanmoy Parvez" ],
22 "occupation" : "Senior Cost Accountant",
23 "interests" : [ "Travelling", "Gaming", "Reading" ],
24 "skills" : [
25   {
26     "name" : "RUBY",
27     "level" : "Expert",
28     "isLearning" : true
29   },
30   {
31     "name" : "KOTLIN",
32     "level" : "Expert",

```

তাহলে pull এর মাধ্যমে আমরা নির্দিষ্ট element কে Delete করতে পারবো।

যদি একটি array থেকে একসাথে একাধিক element pull করতে হয় সেক্ষেত্রে \$pullAll ব্যবহার করা হয়।

\$pullAll

syntax:

```
{ $pullAll: { <field1>: [ <value1>, <value2> ... ], ... } }
```

```

test | 0.038 s | 1 Doc
18 },
19 "company" : "Aivee",
20 "favouriteColor" : "Violet",
21 "friends" : [ "Nahid Hasan Bulbul", "Tanmoy Parvez" ],
22 "occupation" : "Senior Cost Accountant",
23 "interests" : [ "Travelling", "Gaming", "Reading" ],
24 "skills" : [
25   {
26     "name" : "RUBY",
27     "level" : "Expert",
28     "isLearning" : true
29   },
30   {
31     "name" : "KOTLIN",
32     "level" : "Expert",
33     "isLearning" : false
34   }
35 ],
36 "education" : [
37   {
38     "degree" : "Master of Fine Arts",
39     "major" : "History"

```

ধরি এই দুইটিকে ডিলিট করতে চাই।

তাহলে:

```

* practice:test@localhost * practice:test@localhost (2) x
localhost practice
1 - db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000066") },
3   { $pullAll: { interests: ["Gaming", "Reading"] } }
4 )
5

0.031 s
1 - [
2   {
3     "acknowledged" : true,
4     "matchedCount" : 1,
5     "modifiedCount" : 1
6   }
7 ]

```



```

* practice:test@localhost * practice:test@localhost (2) x
localhost practice
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2

0.035 s | 1 Doc
21   "friends" : [ "Nahid Hasan Bulbul", "Tanmoy Parvez" ],
22   "occupation" : "Senior Cost Accountant",
23   "interests" : [ "Travelling" ],
24   "skills" : [
25     {
26       "name" : "RUBY",
27       "level" : "Expert",
28       "isLearning" : true
29     },
30   ]

```

▼ Video 10 (More about \$set, how to explore documentation)

\$set Advanced

```

* practice:test@localhost * practice:test@localhost (2) x
localhost practice
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2

0.035 s | 1 Doc
1 - [
2   {
3     "_id" : ObjectId("6406ad63fc13ae5a40000066"),
4     "name" : {
5       "firstName" : "Otto",
6       "lastName" : "Mirfin"
7     },
8     "email" : "omirfin2@zli.jp",
9     "phone" : "(670) 1851100",
10    "gender" : "Male",
11    "age" : 12,
12    "birthday" : "9/2/2022",
13    "address" : {
14      "street" : "47 Hanover Trail",
15      "city" : "Léguá",
16      "state" : "01",
17      "postalCode" : "3830-247",
18      "country" : "Portugal"
19    },
20    "company" : "Aivee",
21    "favouriteColor" : "Violet",
22    "friends" : [ "Nahid Hasan Bulbul", "Tanmoy Parvez" ],
23    "occupation" : "Senior Cost Accountant",
24    "interests" : [ "Travelling" ],
25    "hobbies" : [
26      "Reading"
27    ]
28  }
29 ]

```

ধরি address এর মধ্যে city এর value পরিবর্তন করবো ।

The screenshot shows two tabs in the MongoDB Compass interface: 'practice' and 'practice:test@localhost (2)'. In the 'practice' tab, a command is run:

```
1 db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000066") },
3   {
4     $set: {
5       "address.city": "dhaka"
6     }
7   }
8 )
```

The result of the update is shown in the '0.033 s' section:

```
1 [
2   {
3     "acknowledged" : true,
4     "matchedCount" : 1,
5     "modifiedCount" : 1
6   }
7 ]
```

In the 'test' tab, a find operation is run:

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
```

The results show the document with the updated address:

```
8   "phone" : "(670) 1831100",
9   "gender" : "Male",
10  "age" : 12,
11  "birthday" : "9/2/2022",
12  "address" : {
13    "street" : "47 Hanover Trail",
14    "city" : "dhaka",
15    "state" : "01",
16    "postalCode" : "3830-247",
17    "country" : "Portugal"
18  },
```

A red arrow points from the '\$set' part of the update query in the first tab to the 'city' field in the document's 'address' object in the second tab.

তাহলে বলা যায় non primitive এর মধ্যে object এর ক্ষেত্রে \$set ব্যবহার করা যাবে ।

এখন যদি array of object এর ক্ষেত্রে আপডেট করতে চাই তাহলেঃ

Documentation: <https://www.mongodb.com/community/forums/t/mongodb-update-a-value-in-array-of-object-of-array/208302>

Documentation: <https://sparkbyexamples.com/mongodb/update-objects-in-the-array-in-mongodb/>

এইখানে দুই জায়গা তেই \$ ব্যবহার হয়েছে । এইটা একটু বুঝাতে হবে ।

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/update/positional/>

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2

30 {
31   "name" : "KOTLIN",
32   "level" : "Expert",
33   "isLearning" : false
34 }
35 ],
36 "education" : [
37   {
38     "degree" : "Master of Fine Arts",
39     "major" : "History",
40     "institute" : "East West University",
41     "year" : 1994
42   },
43   {
44     "degree" : "Juris Doctor",
45     "major" : "Philosophy",
46     "institute" : "Kangnam University",
47     "year" : 1997
48   }
49 ],
50 "languages" : [ "Assamese", "Estonian", "Portuguese" ],
51 "ipAddress" : "213.32.106.50",
52 "salary" : 202
53 ]

```

ধরি এই major এর value পরিবর্তন করবো।

```

1 db.test.updateOne(
2   { _id: ObjectId("6406ad63fc13ae5a40000066") }, { "education.major": "History" },
3   {
4     $set: {
5       "education.$.major": "CSE"
6     }
7   }
8 )
9

```

এইখানে Access করে value assign
করা হলো

```

1 {
2   "acknowledged" : true,
3   "matchedCount" : 1,
4   "modifiedCount" : 1
5 }

```

এক্ষেত্রে যখন Query করবো তখন যদি একাধিক ওটার মতো থাকে প্রথমে যেটা থাকবে সেটাকে সিলেক্ট করবে।

Output:

```

1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
2

30 {
31   "name" : "KOTLIN",
32   "level" : "Expert",
33   "isLearning" : false
34 }
35 ],
36 "education" : [
37   {
38     "degree" : "Master of Fine Arts",
39     "major" : "CSE",
40     "institute" : "East West University",
41     "year" : 1994
42   },
43   {
44     "degree" : "Juris Doctor",
45     "major" : "Philosophy",
46     "institute" : "Kangnam University"
47   }
48 ],
49 ],
50 "languages" : [ "Assamese", "Estonian", "Portuguese" ],
51 "ipAddress" : "213.32.106.50",
52 "salary" : 202
53 ]

```

এখনঃ

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })  
2  
3 {  
4   "_id": ObjectId("6406ad63fc13ae5a40000066"),  
5   "name": {  
6     "firstName": "Otto",  
7     "lastName": "Mirfin"  
8   },  
9   "email": "omirfin2@i2i.jp",  
10  "phone": "(670) 1831100",  
11  "gender": "Male",  
12  "age": 12, ----->  
13  "birthday": "9/2/2022",  
14  "address": {  
15    "street": "47 Hanover Trail",  
16    "city": "dhaka",  
17    "state": "01",  
18    "postalCode": "3830-247",  
19    "country": "bangladesh"  
20  },  
21  "company": "Aivee",  
22  "favoutiteColor": "Violet",  
23  "friends": [ "Nahid Hasan Bulbul", "Tanmoy Parvez" ]  
24}
```

আমি চাইছি age এর মান প্রতিবার ১ করে বাড়বে।

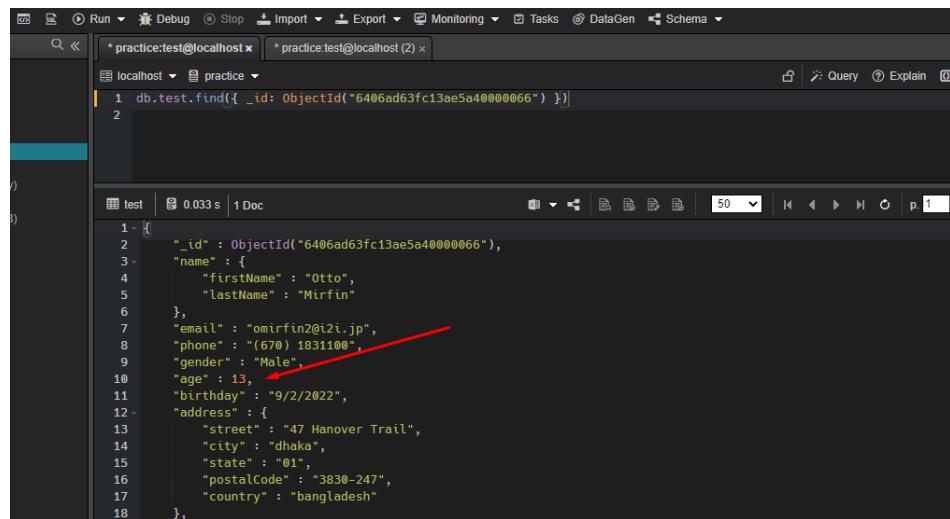
\$inc

Documentation: <https://www.mongodb.com/docs/manual/reference/operator/update/inc/>

syntax:

```
{ $inc: { <field1>: <amount1>, <field2>: <amount2>, ... } }
```

```
1 db.test.updateOne(  
2   { _id: ObjectId("6406ad63fc13ae5a40000066") },  
3   {  
4     $inc: {  
5       age: 1  
6     }  
7   }  
8 )  
9  
10 {  
11   "acknowledged": true,  
12   "matchedCount": 1,  
13   "modifiedCount": 1  
14 }
```

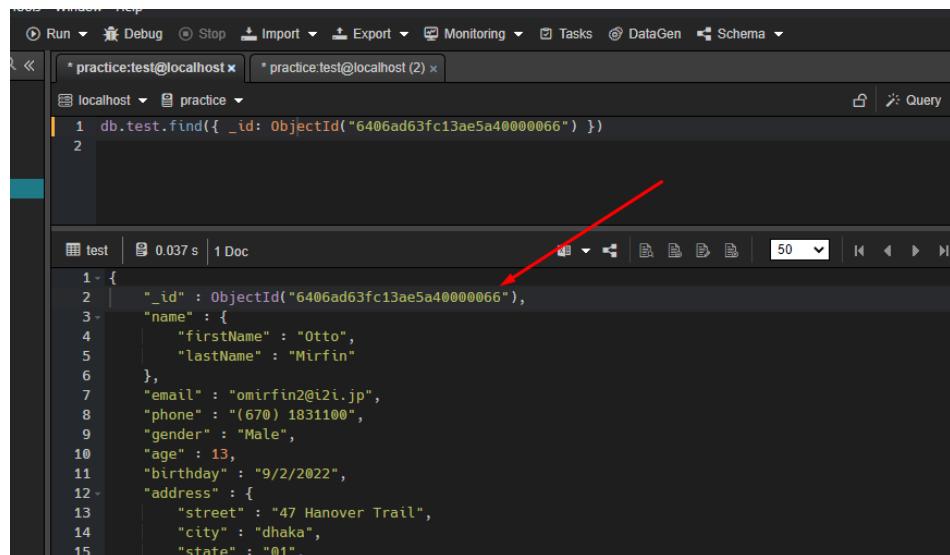


```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
```

```
1 [
2   {
3     "_id": ObjectId("6406ad63fc13ae5a40000066"),
4     "name": {
5       "firstName": "Otto",
6       "lastName": "Mirfin"
7     },
8     "email": "omirfin2@i2i.jp",
9     "phone": "(670) 1831100",
10    "gender": "Male",
11    "age": 13, ----->
12    "birthday": "9/2/2022",
13    "address": {
14      "street": "47 Hanover Trail",
15      "city": "dhaka",
16      "state": "01",
17      "postalcode": "3830-247",
18      "country": "bangladesh"
19    }
20  ]
21 ]
```

▼ Video 11 (delete documents, drop collection)

delete documents



```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
```

```
1 [
2   {
3     "_id": ObjectId("6406ad63fc13ae5a40000066"),
4     "name": {
5       "firstName": "Otto",
6       "lastName": "Mirfin"
7     },
8     "email": "omirfin2@i2i.jp",
9     "phone": "(670) 1831100",
10    "gender": "Male",
11    "age": 13,
12    "birthday": "9/2/2022",
13    "address": {
14      "street": "47 Hanover Trail",
15      "city": "dhaka",
16      "state": "01",
17    }
18  ]
19 ]
```

মনে করি পুরো ডকুমেন্ট টি ডিলিট করবো।

Documentation: <https://www.mongodb.com/docs/mongodb-shell/crud/delete/>

A screenshot of the MongoDB Compass interface. The top bar shows 'Tools', 'Window', and 'Help' with various menu items like 'Run', 'Import', 'Export', 'Monitoring', 'Tasks', 'DataGen', and 'Schema'. Below the bar, there are two tabs: '* practice:test@localhost' and '* practice:test@localhost (2)'. The main panel shows a query in the 'localhost' database's 'practice' collection. The query is:

```
1 db.test.deleteOne({ _id: ObjectId("6406ad63fc13ae5a40000066") })
```

The results pane shows the response with a duration of 0.037s:

```
1 { "acknowledged" : true, "deletedCount" : 1 }
```

Two red arrows point from the text 'Collection Name' in the 'Create Collection' section below to the word 'test' in the database name of the tabs and the collection name in the query.

A screenshot of the MongoDB Compass interface. The top bar shows 'Tools', 'Window', and 'Help' with various menu items like 'Run', 'Import', 'Export', 'Monitoring', 'Tasks', 'DataGen', and 'Schema'. Below the bar, there are two tabs: '* practice:test@localhost' and '* practice:test@localhost (2)'. The main panel shows a query in the 'localhost' database's 'practice' collection. The query is:

```
1 db.test.find({ _id: ObjectId("6406ad63fc13ae5a40000066") })
```

The results pane shows the response with a duration of 0.033s and 0 documents found:

```
1 [ ]
```

Create Collection

```
db.createCollection("posts")
```

A red arrow points from the text 'Collection Name' in the 'Create Collection' section below to the word 'posts' in the collection name of the query.

Delete Collection / Drop Collection

ধরি post collection টাকে ডিলিট করবো তাহলে

```
db.posts.drop( { writeConcern: { w: 1 } } )
```

▼ Module 6: Mastering MongoDB Aggregation & Indexing

▼ Video 0 (Intro to the powerful aggregation framework)

যদি আমার collection এর ডাটা কে different stage এ পাস করি তাকে বলা হয় aggregation .

stage গুলোকে বলা হয় Pipeline .

উদাহরণঃ

যদি ৮ জন কাজিন মিলে ঘুরতে যাবে তারিখ ঠিক হলো জানুয়ারি এর ৩১ তারিখ।

কিন্তু ২ জন এর পরিষ্কা তাহলে ২ জন বাদ। বাকি থাকলো ৬ জন।

কিছু দিন পর দেখা গেলো ২ জন এর টাকার সমস্যা তাহলে আরো ২ জন বাদ বাকি থাকলো ৪ জন।

তারপর দেখা গেলো ১ জন অসুস্থ। তাহলে বাকি থাকলো শেষে ৩ জন।

এখন টিকিট কাটতে যেয়ে দেখা গেলো টিকিট আছে ২ টা।

তাহলে যারা বড় তাদের sort করলো এবং বড় দুই জন গেলো তাদের বাজেট calculation করে।

এবং শেষে তারা দুইজন মিলে একটা গ্রুপ করলো এবং Tour এ গেলো।

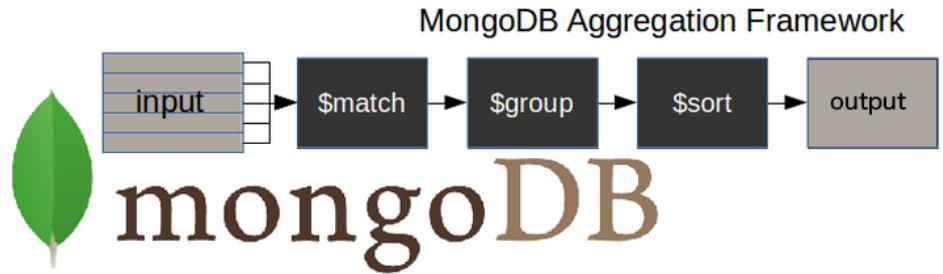
এই সম্পূর্ণ কাজটি aggregation এর মাধ্যমে করা যাবে।



```
db.cousins.aggregate([
    // filter out the cousins who have an exam
    { $match: { hasExam: { $ne: true } } },
    //filter out cousins who have a budget less than 500
    { $match: { budget: { $gte: 500 } } },
    //filter out cousins who are sick
    { $match: { isSick: false } },
    //sort by age
    { $sort: { age: -1 } },
    //limit by 2
    { $limit: 2 },
    // calculate the budget
    {
        $group: {
            _id: "null",
            totalBudget: { $sum: "$budget" },
            cousins: { $push: "$name" }
        }
    }
])
```

▼ Video 1 (\$match , \$project aggregation stage)

Common Flow :



\$match

The `$match` stage allows us to choose just those documents from a collection we want to work with. It does this by filtering out those that need to follow our requirements.

অর্থঃ কাঞ্চিত ডকুমেন্ট গুলো ফিল্টার করে ম্যাচ করে নিয়ে আসবে।