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)

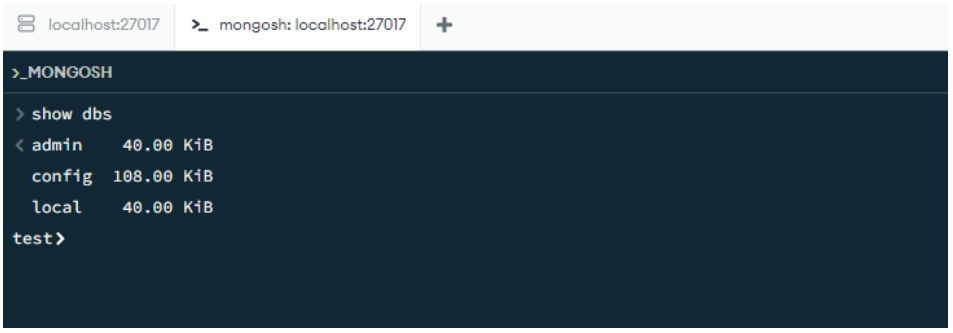
MongoDB Compass downlode. (search Google download mongoDB)

সেখানে shell এর মধ্যে আমরা আমাদের query গুলো লিখতে পারবো ।

যেমনঃ

```
show dbs
```

এটি লিখলে আমাদের যেগূলো ডাটাবেস আছে সেগুলো দেখাবে ।



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>
```

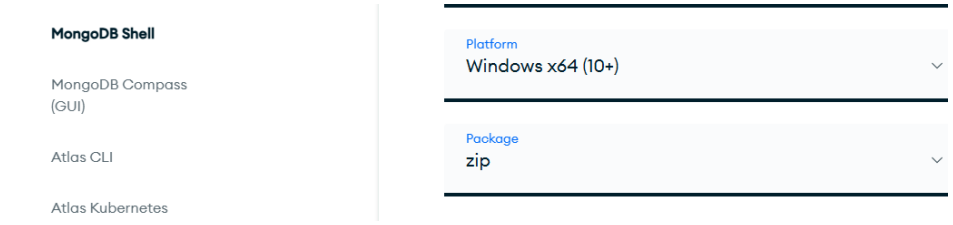
এখন যদি সবগুলো ডাটা দেখতে চাই তাহলে

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

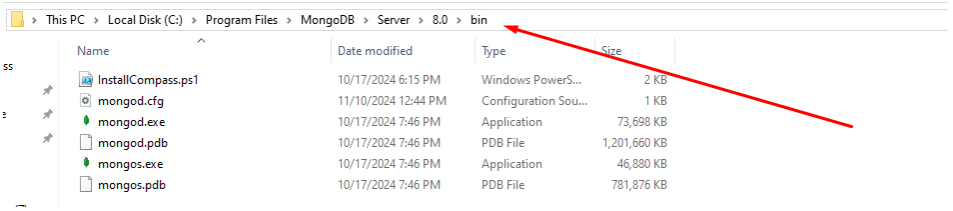
```
> db.getCollection("test").find()
< {
    _id: ObjectId('673065af57fe790328908b11'),
    name: 'Next level web developement'
  }
practice>
```

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

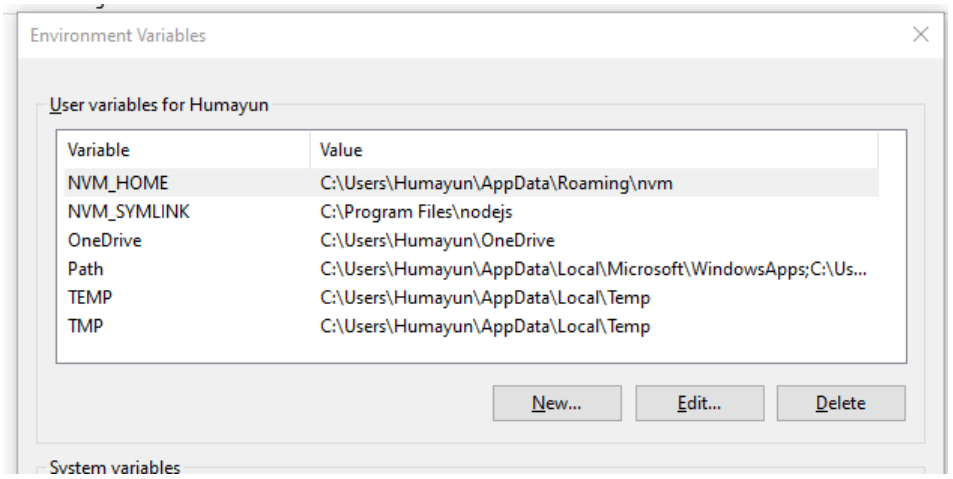
তার জন্য Environment veriable এ path অ্যাড করতে হবে ।



তার জন্য MongoDB Shell download হবে ।



এই path টা কপি করবো ।



Edit environment veriable 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>
```

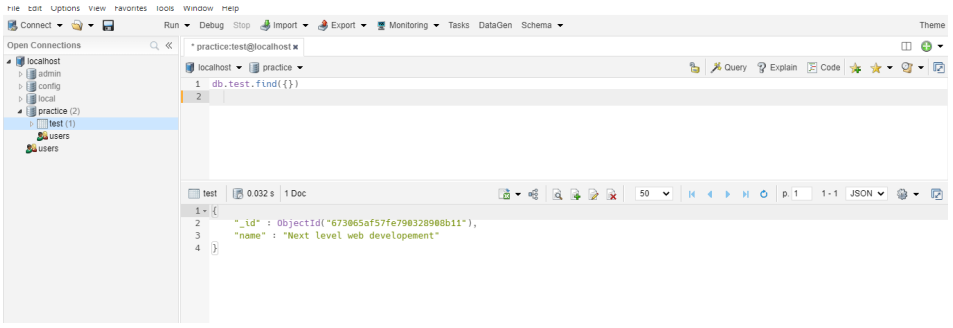
mongod —version দিলে version দেখাবে ।

```
C:\Users\Humayun>mongosh
Current Mongosh Log ID: 67306b74c28b055fc60d818f
Connecting to:          mongodb://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.co
m/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 conf
iguration 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**

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 ।

আগের ডাটা গুলো সব ডিলিট করে দিলাম ।

প্রাক্টিস এর জন্য ডাটাঃ

GitHub - HumayunKabirSobuj/mongodb-practice

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

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

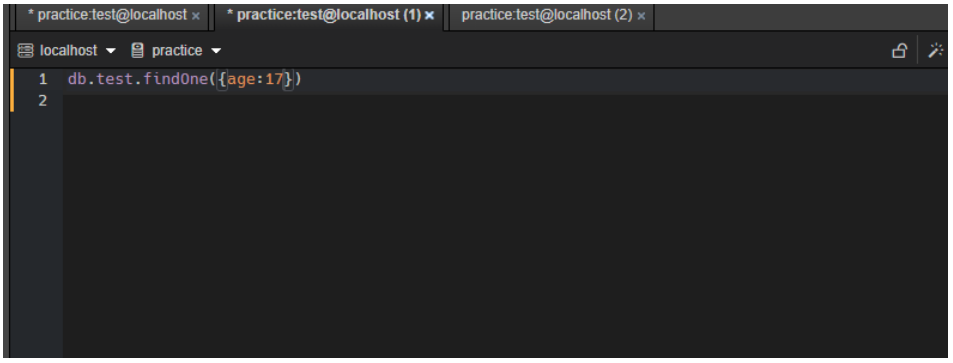
HumayunKabirSobuj/
mongodb-practice

1 Contributor · 0 Issues · 0 Stars · 0 Forks

এইখানের practice-data.json

findOne

এর মাধ্যমে একটি ডাটা খুজে দেয় ।



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

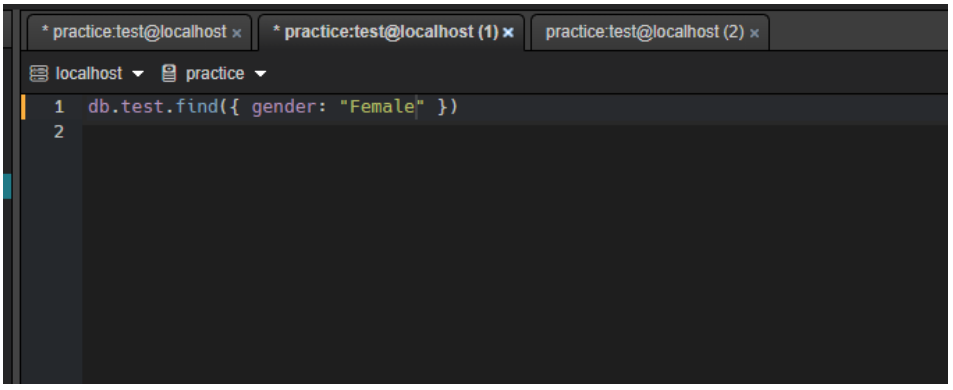
find

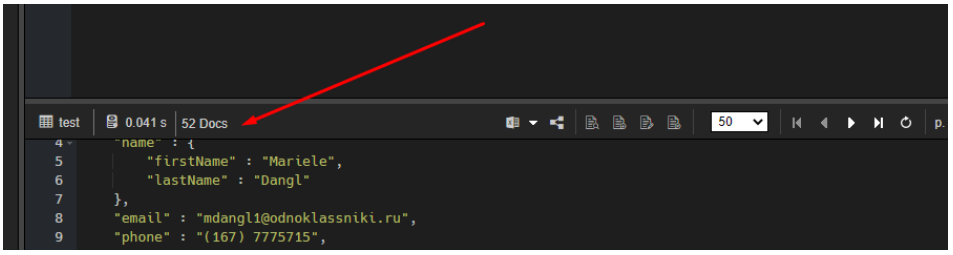
এর মাধ্যমে অনেকগুলো ডাটা একসাথে দেয় ।

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

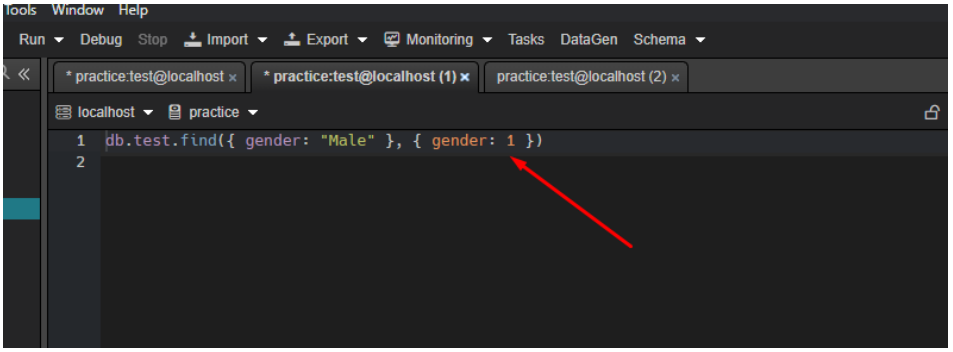




এক্ষেত্রে যেগুলো ম্যাচ হবে তার সব ইনফরমেশন দেখাবে ।

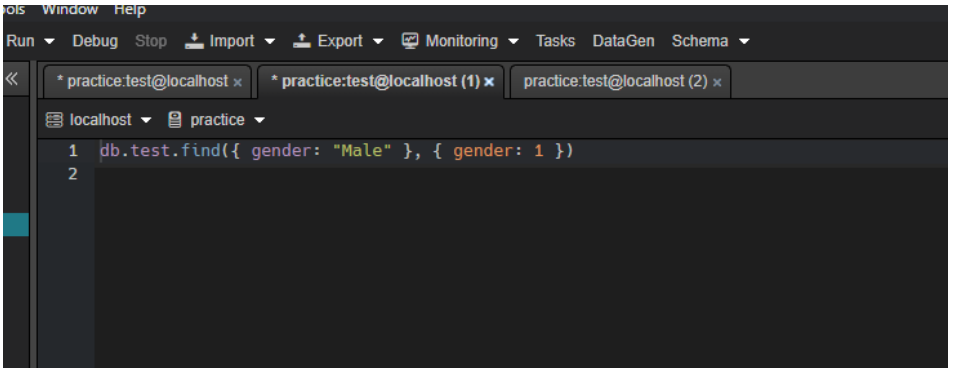
যদি মনে হয় সবগুলো ডাটা আমার দরকার শুধু name, email, phone ,gender এই field গুলো দরকার ।

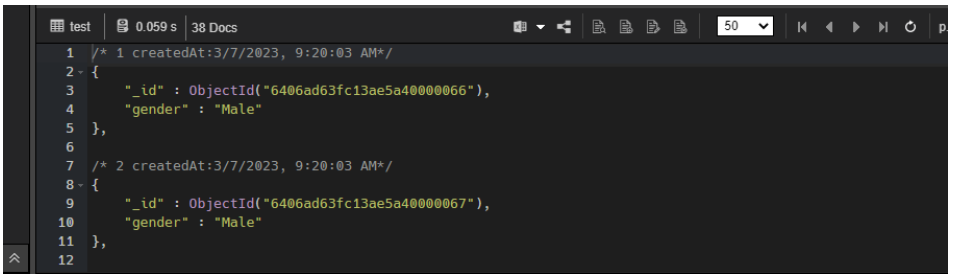
তাহলে { } এর পরে কমা দিয়ে { } এর মধ্যে বলে দিতে হবে আমার যেগুলো যেগুলো দরকার ।



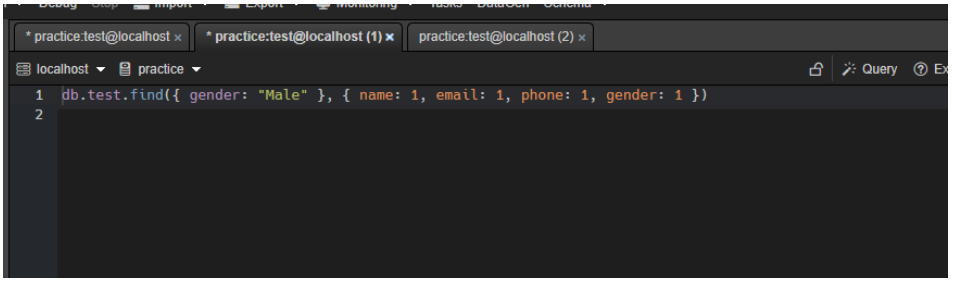
১ মানে হলো এটা আমার লাগবে ।

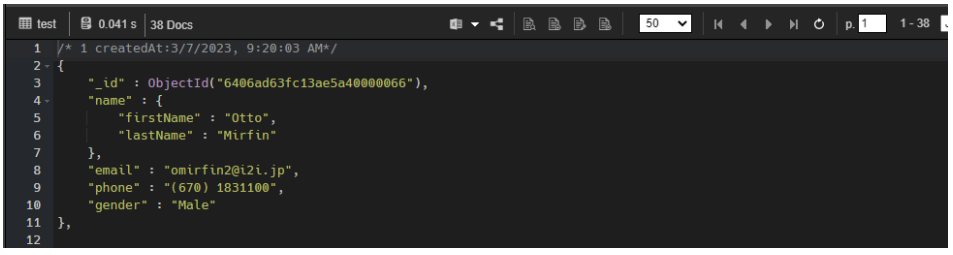
তাহলে শুধু gender দিবে ।





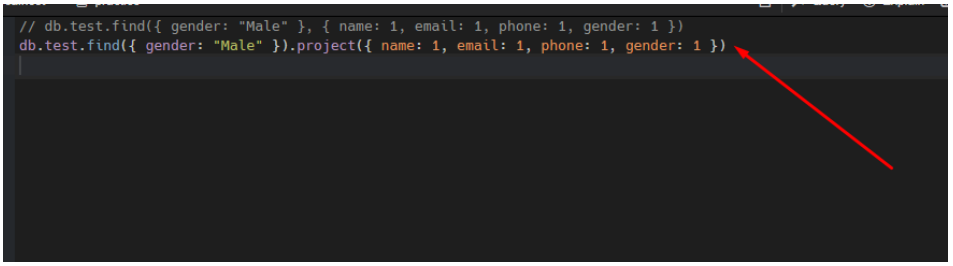
এটাকে বলা হয় **field filtering** .





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

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



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

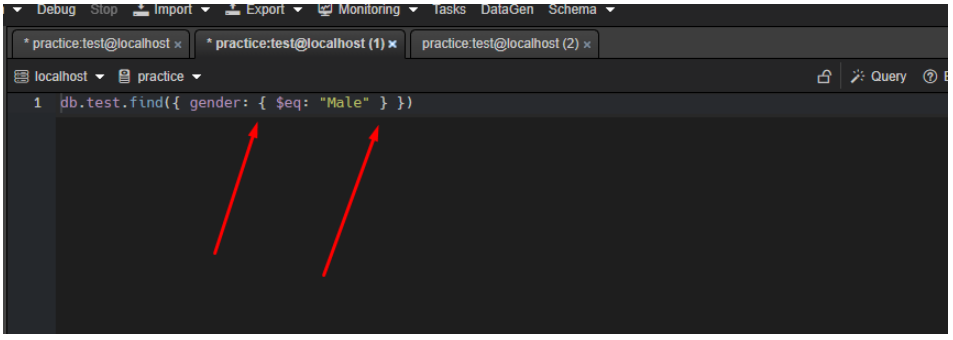
▼ Video 3 ($eq, $neq, $gt, $lt, $gte, $lte)

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

প্রথমেঃ **Comparison Query Operators**

Query শেখার thumb rule :

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



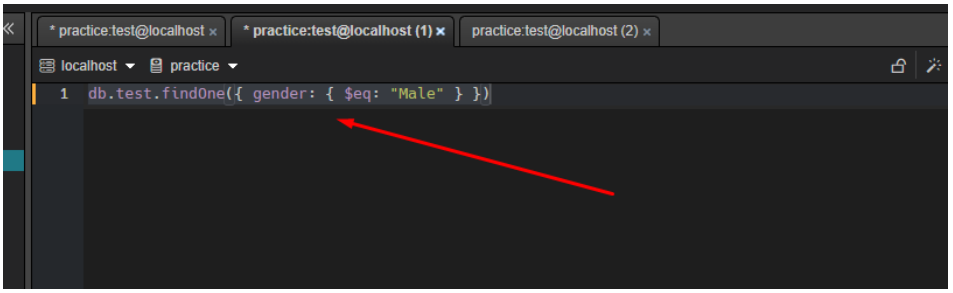
name, email, phone, gender প্রভৃতি এক একটি field।

এগুলো field filtering এর মতো কাজ করে।

$eq

syntax :

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



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

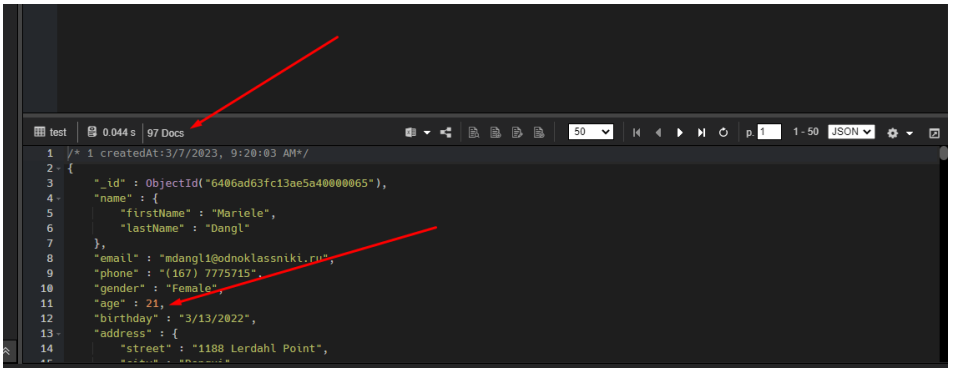
eq = equal অর্থ সমান।

$eq এর বিপরীত হলো $ne

$ne

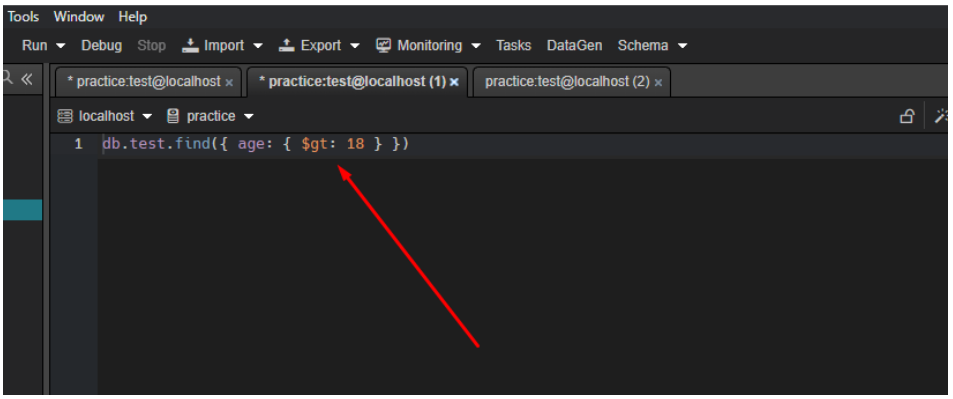
ne = not equal অর্থ অসমান।

age 12 এর সমান নয় এমন সবগুলো দিবে।



$gt

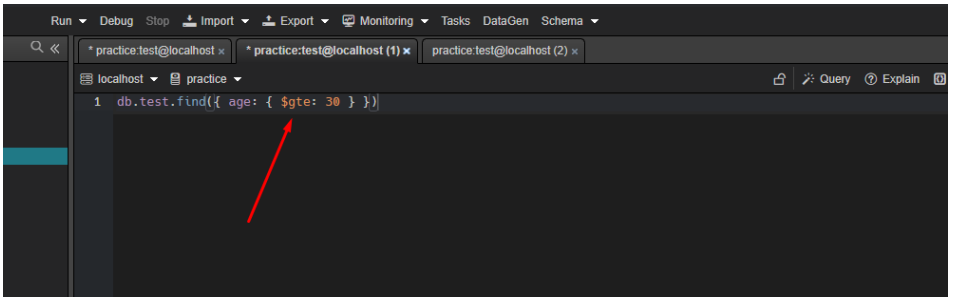
gt = greater than



বয়স ১৮ এর উপরে যাদের সেগুলো সব দিবে ।

$gte

gte = greater than or equal

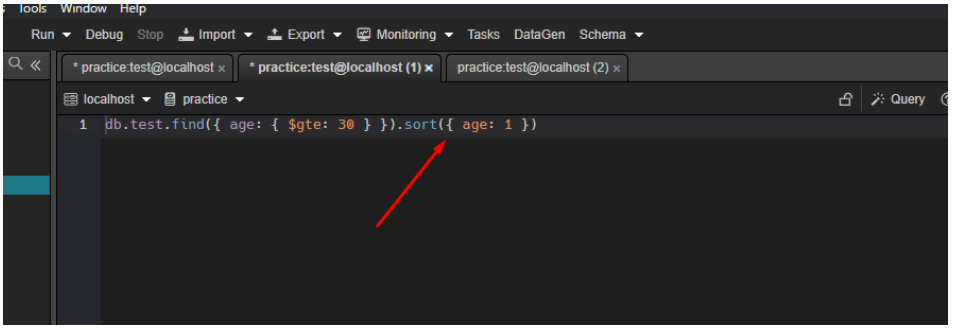


বয়স ৩০ অথবা ৩০ এর উপরে সেগুলো সব দিবে । অর্থাৎ ৩০ সহ নিবে ।

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 .



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

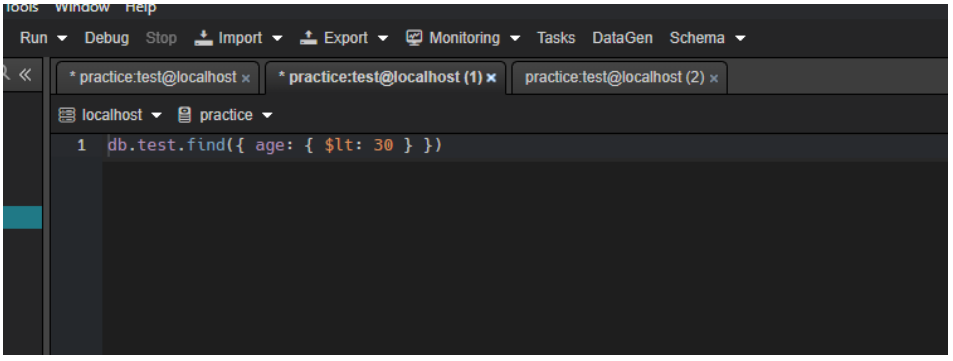
বড় থেকে ছোট হলে অর্থাৎ Disendingহলে -1 দিতে হবে ।

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

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

$lt

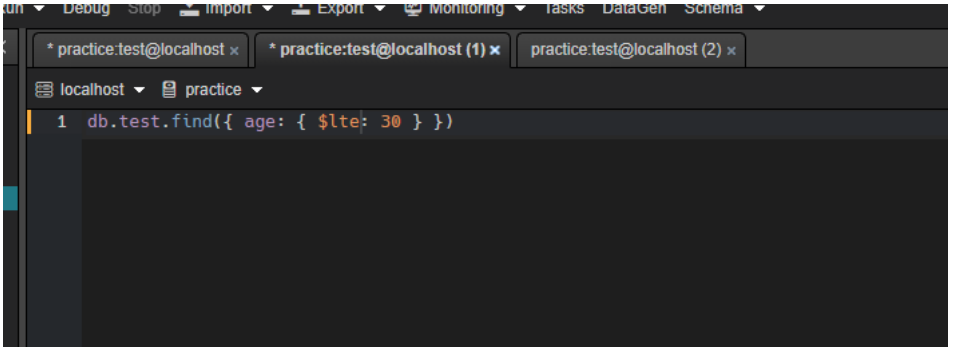
lt = less than



বয়স ৩০ এর থেকে কম যাদের তাদের গুলো দিবে ।

$lte

lte = less than or equal



যাদের বয়স ৩০ বা তার থেকে কম ।

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

$in

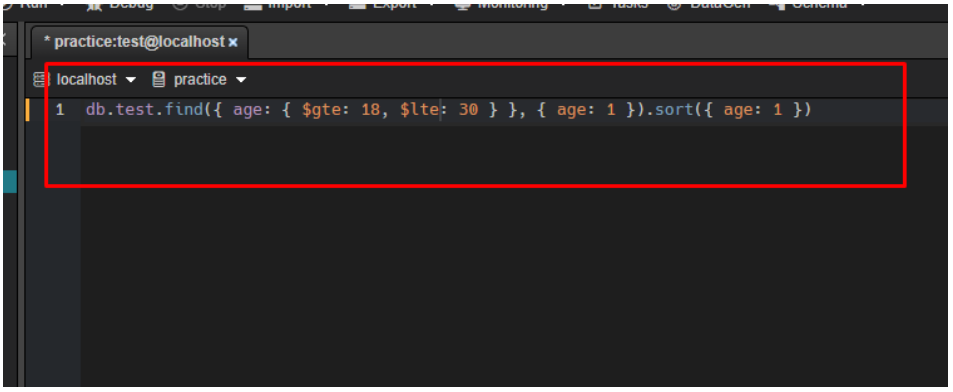
The **$in** operator selects the documents where the value of a field equals any value in the specified array.

$in এর মধ্যে যে Array value হিসেবে দিবো সেই value গুলোর উপর ভিত্তি করে সে query করে নিয়ে আসবে । ঐ 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)



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** .

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 রিটার্ন করবে ।

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

সেক্ষেত্রে আমরা **$in** ব্যবহার করতে পারবো ।

```
db.test.find(
    { gender: "Female", age: { $in: [18, 20] } },
    { age: 1, gender: 1 })
    .sort({ age: 1 })
```

test | 0.040 s | 1 Doc

```
{
    "_id" : ObjectId("6406ad64fc13ae5a4000006d"),
    "gender" : "Female",
    "age" : 18
}
```

১৮,২০ value একটা ডাটার সাথে মিলেছে সেটা দিয়েছে ।

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

test | 0.036 s | 3 Docs

```
/* 1 createdAt:3/7/2023, 9:20:04 AM*/
{
    "_id" : ObjectId("6406ad64fc13ae5a4000006d"),
    "gender" : "Female",
    "age" : 18
},

/* 2 createdAt:3/7/2023, 9:20:04 AM*/
{
    "_id" : ObjectId("6406ad64fc13ae5a40000082"),
    "gender" : "Female",
    "age" : 24
},

/* 3 createdAt:3/7/2023, 9:20:05 AM*/
{
    "_id" : ObjectId("6406ad65fc13ae5a400000b0"),
    "gender" : "Female",
    "age" : 26
}
```

$nin

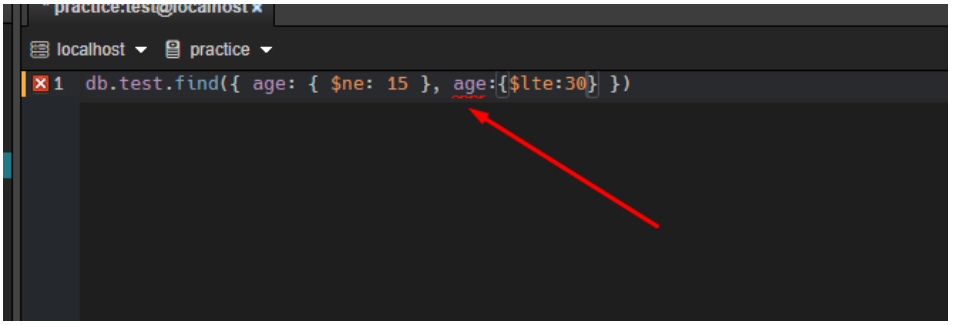
$nin হচ্ছে $in এর বিপরীত ।

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

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

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

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



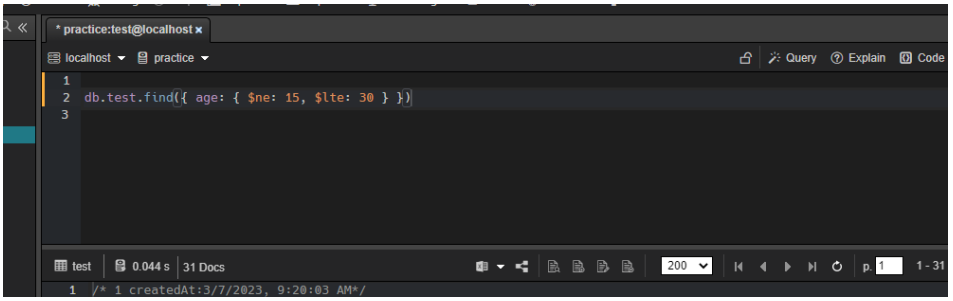
এটা লিখলে Error দিবে Duplicate Identifier .

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

সেক্ষেত্রে একই bracket এর মধ্যে নিয়ে যেতে হবে ।

solution :

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



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 } }
```

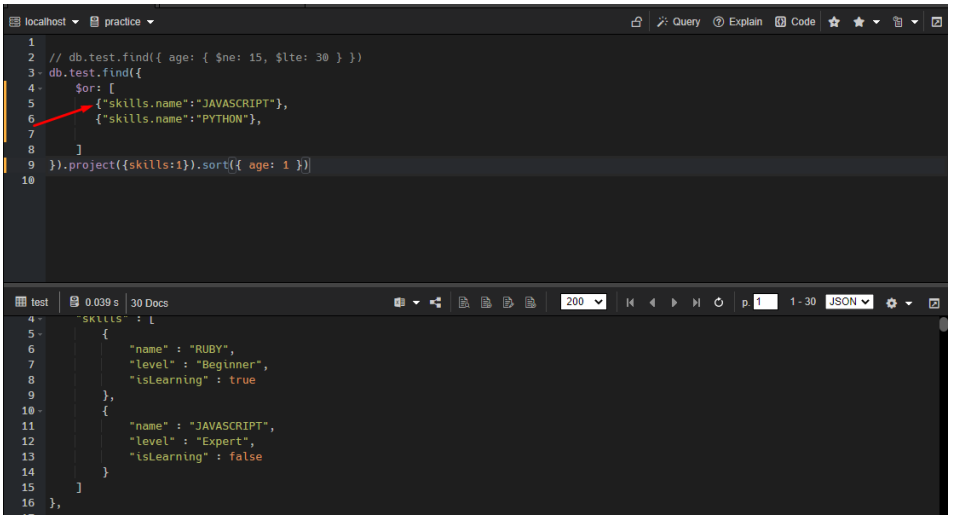
```
    ]
})
```

$or ⇒ Explicid OR

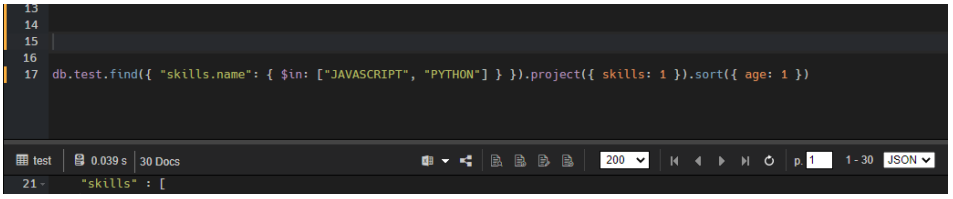
```
db.test.find({
    $or: [
        {interests:"Travelling"},
        {interests:"Cooking"},

    ]
}).project({interests:1}).sort({ age: 1 })
```

এক্ষেত্রে যেকোন একটা Condition সত্য হলেই সেটি return করবে ।

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



এই কাজটি যদি Implicid OR দিয়ে করতে চাই তাহলেঃ



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

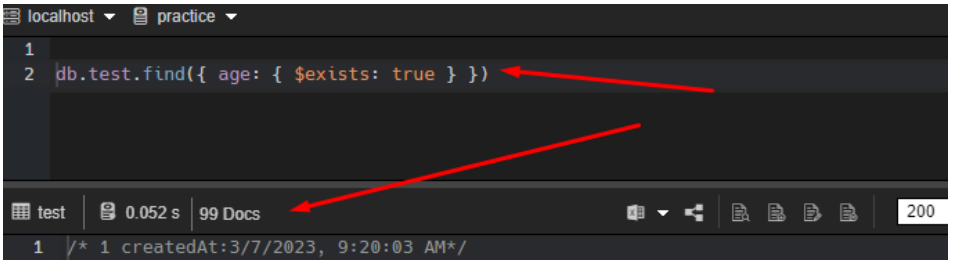
কোন একটি field কোন একটি document এ exist করে কিনা সেটা দেখাবে ।

syntax:

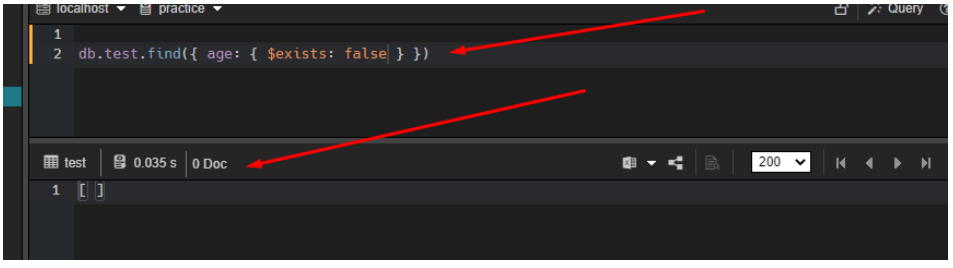
```
{ field: { $exists: <boolean> } }
```

value হিসেবে true , false যেকোনো টি দেওয়া যাবে ।

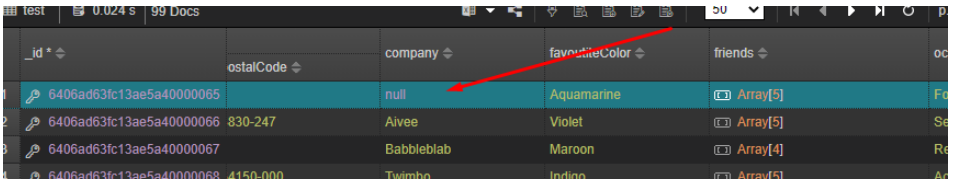
সেগুলো ডাটা দেখাও যেগুলোতে age আছে । তাহলেঃ



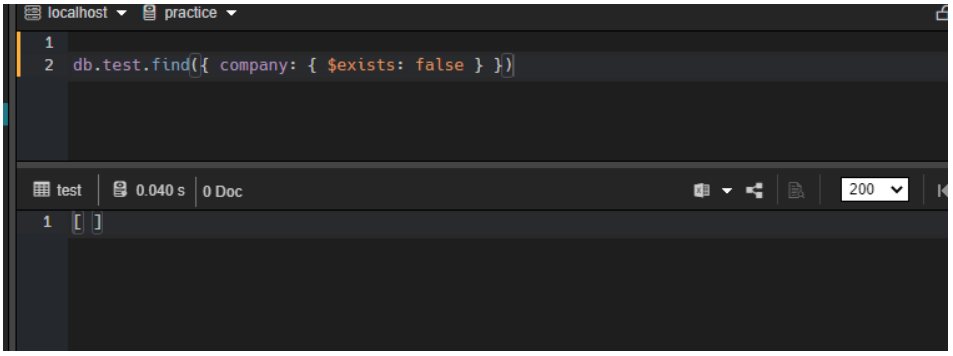
সেগুলো ডাটা দেখাও যেগুলোতে age নেই। তাহলেঃ



এখন ,



একটি company এর field null আছে।

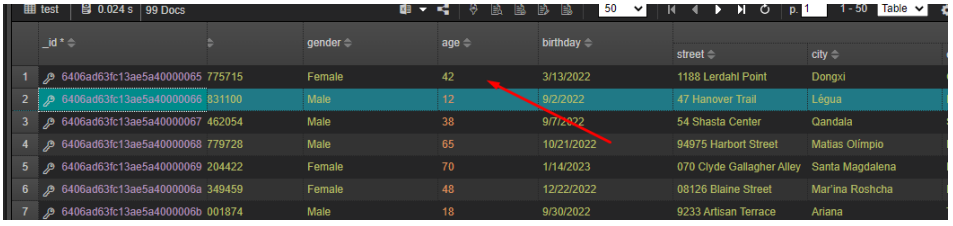


কিন্তু এই ক্ষেত্রে exists operator 0 return করে।

exists operator null, undefine এবং empty arrap [ ] বের করে দেয় না।

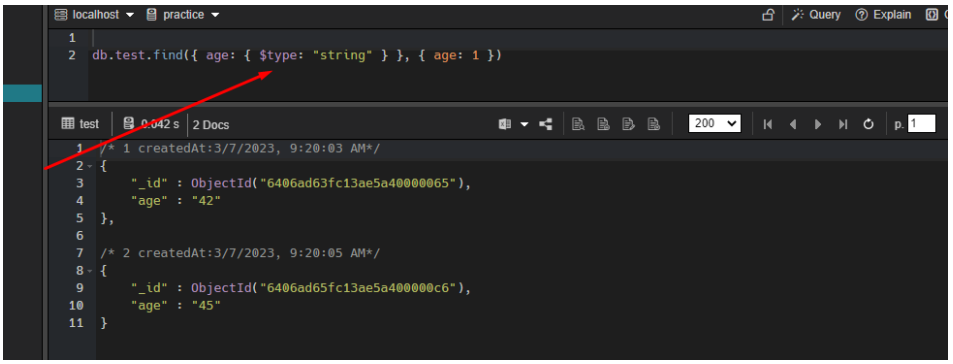
$type

কোন কিছুর টাইপ বের করার জন্য।

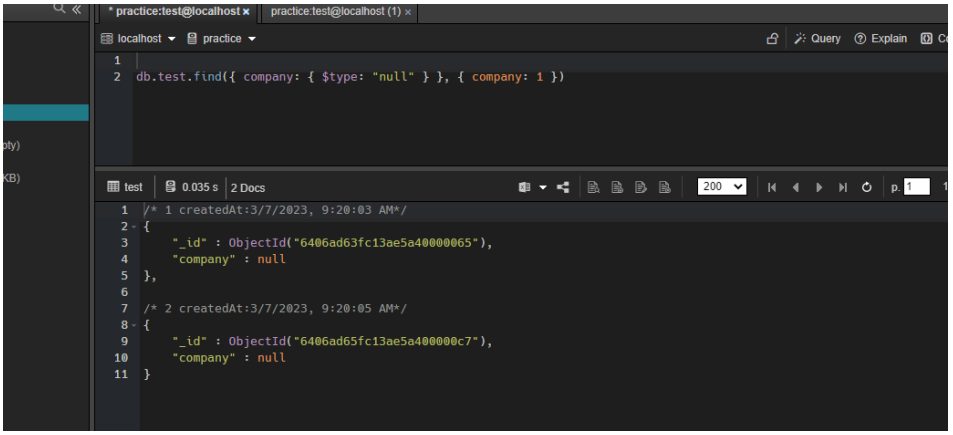


ধরি এই age এর ডাটাটি string ফরম্যাট এ আছে বা বানিয়ে নিলাম।

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



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



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

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

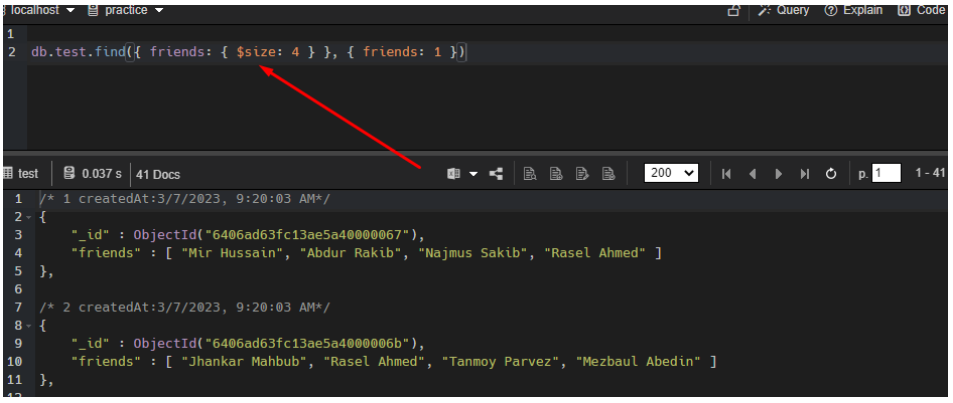
$size

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



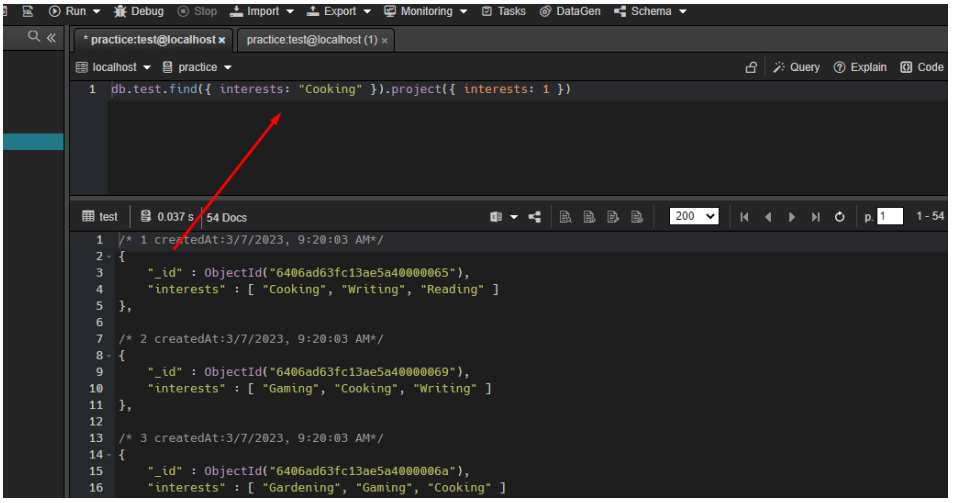
যেসব array এর সাইজ ৪ সেগুলো দেখাবে ।

সেক্ষেত্রে বলা যায় $size operator শুধুমাত্র Array এর ক্ষেত্রে ।

▼ Video 7 ($all , $elemMatch)

interests object এর মধ্যে Element হিসেবে Cooking আছে।

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



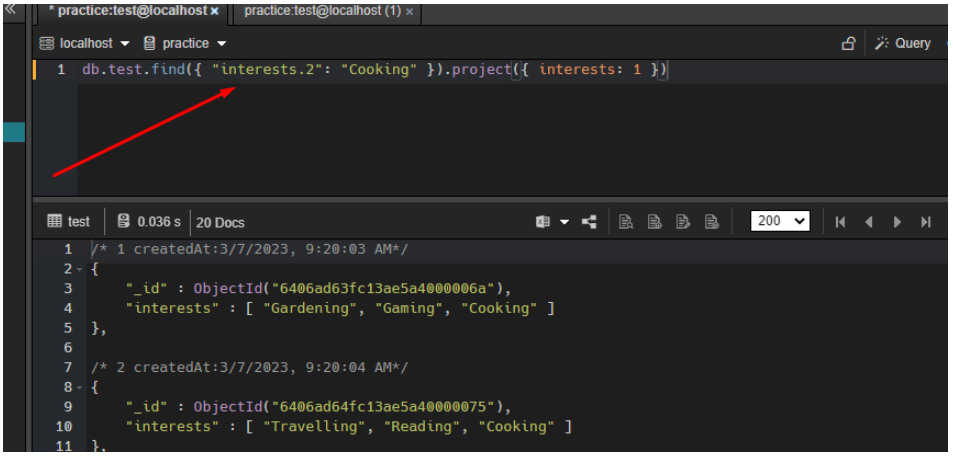
output এ কিছু কিছু cooking আছে last এর দিকে।

ধরি লাস্ট এর দিকে যেসব Cooking আছে সেগুলো আমার দরকার।

সেক্ষেত্রে position হিসাব করতে হবে।

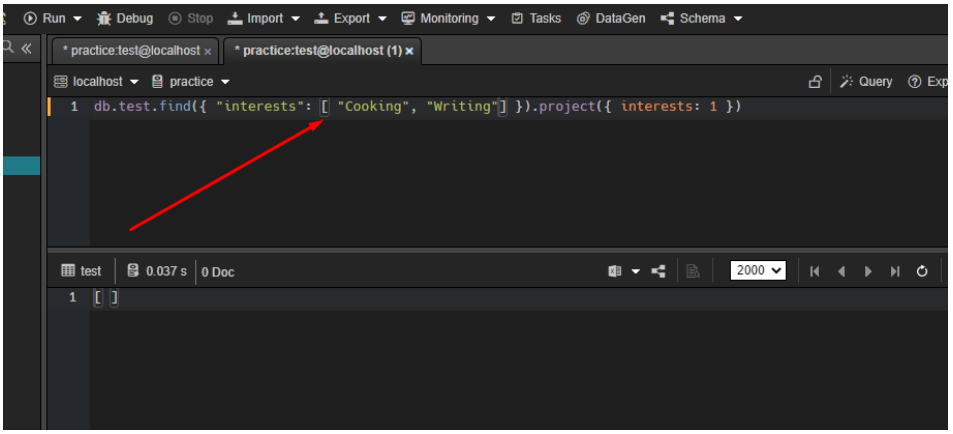
আমারা array এর পজিশন হিসাব করি index এর মাধ্যমে।

২ নাম্বার index এ যেসব ডাটার cooking আছে সেগুলো দেখতে চাইলে



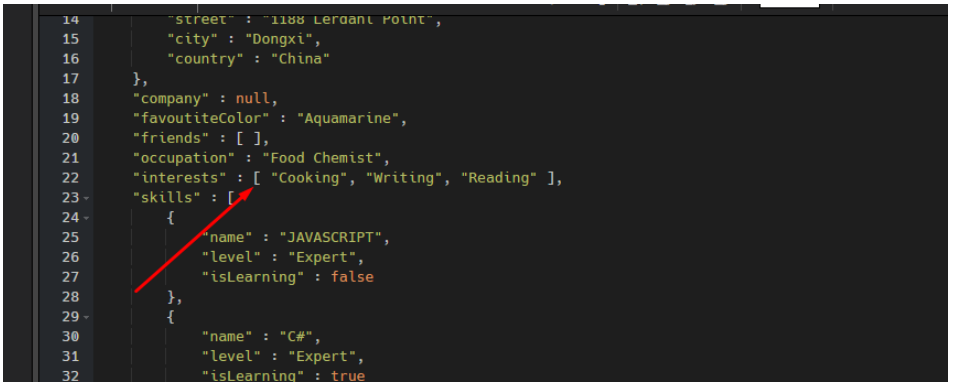
যেহেতু ডট (.) syntax ব্যবহার করেছি তাই কোটেশন এর মধ্যে নিতে হবে ।

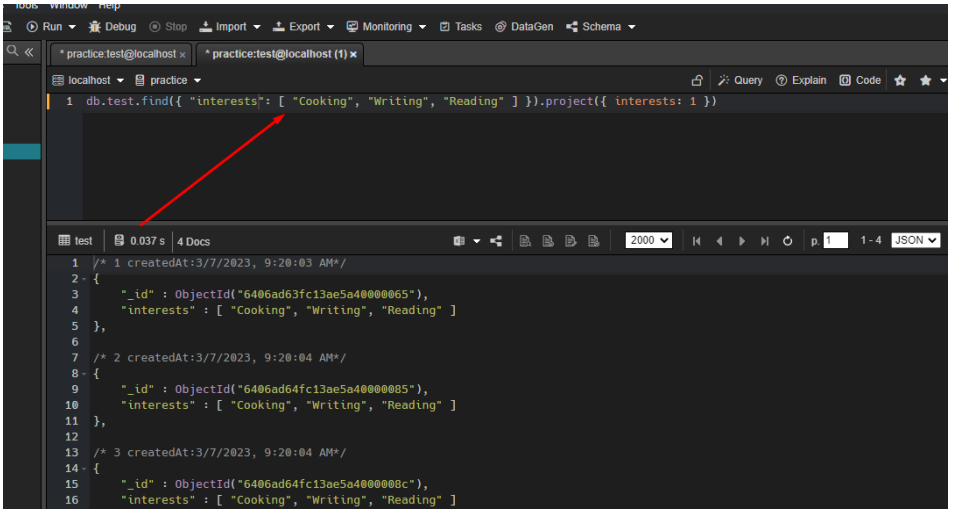
এখন ধরি Cooking এর সাথে Gardening লাগবে ।



এইভাবে লিখলে হবে না ।

কারণঃ





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 ।

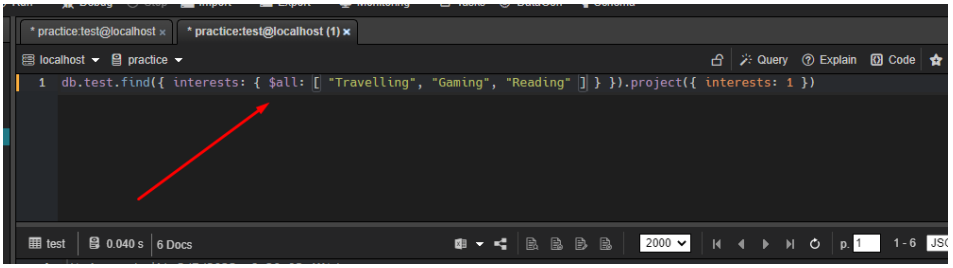
সেক্ষেত্রে $all operator ব্যবহার করতে হবে ।

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

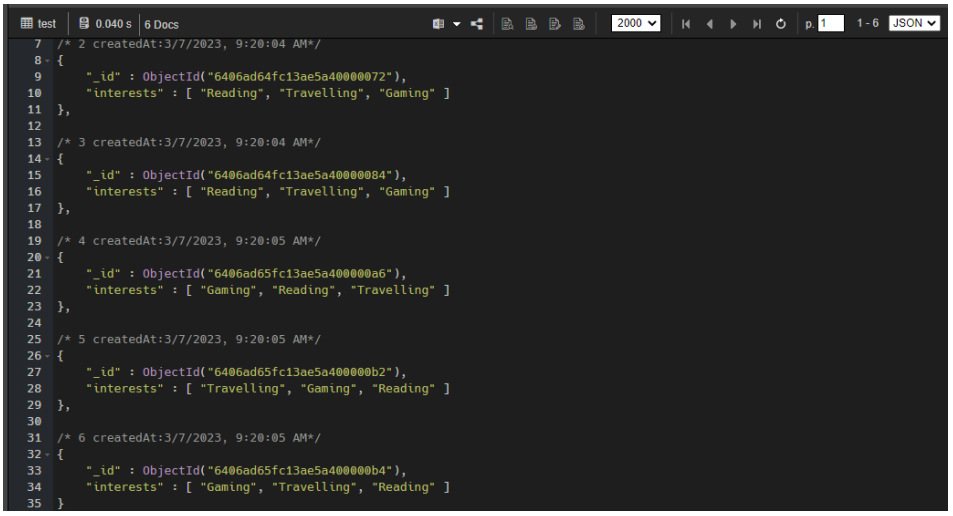
$all operator

syntax:

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



Output :

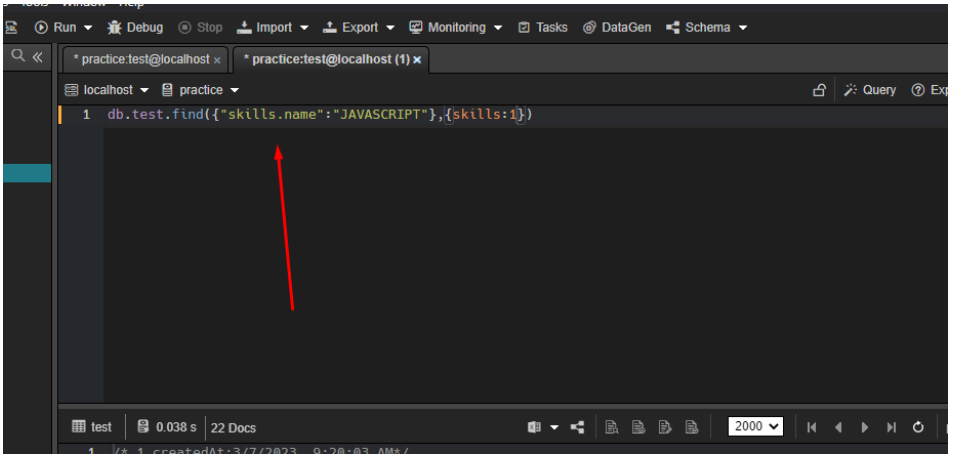


এক্ষেত্রে এই তিনটি থাকলেই হবে যেকোন পজিশন এ থাকতে পারে কোন সমস্যা নাই ।

Array এর মতো করে Object ও Find করতে পারি ।

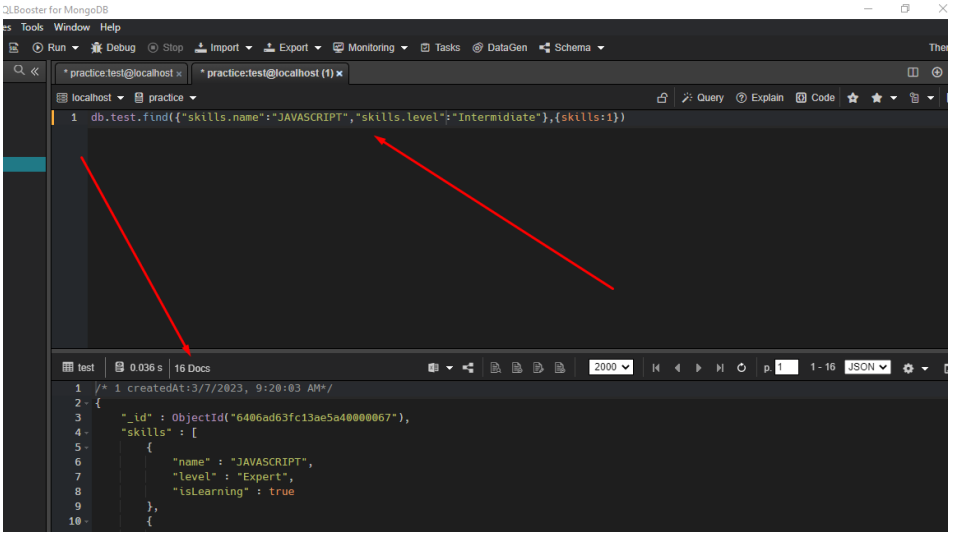
skills এর মধ্যে name হিসেবে JAVASCRIPT আছে।

এখন আমি Javascript কে find করতে চাই।



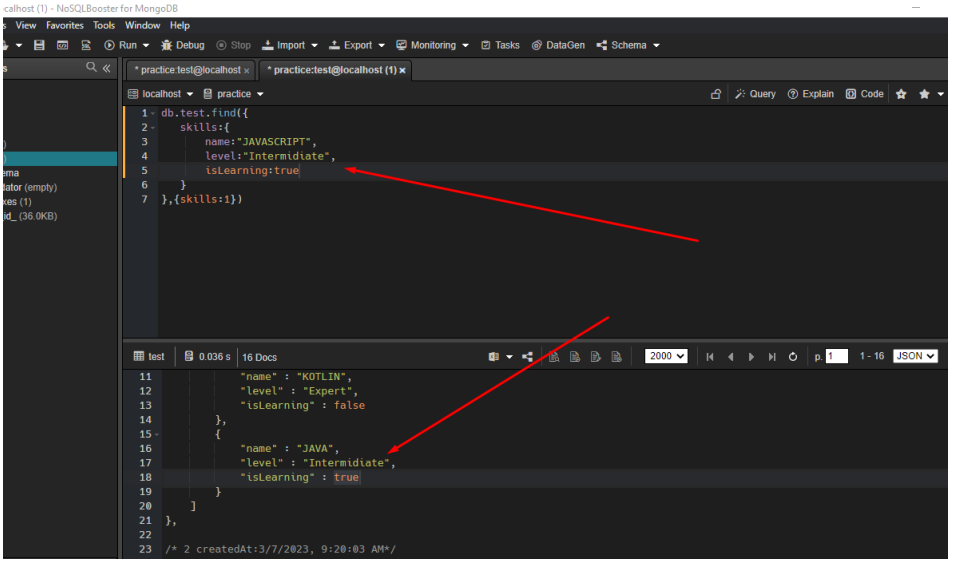
এইভাবে লিখতে হবে।

এখন ধরি আমার JAVASCRIPT দরকার আর level দরকার Intermidiate .

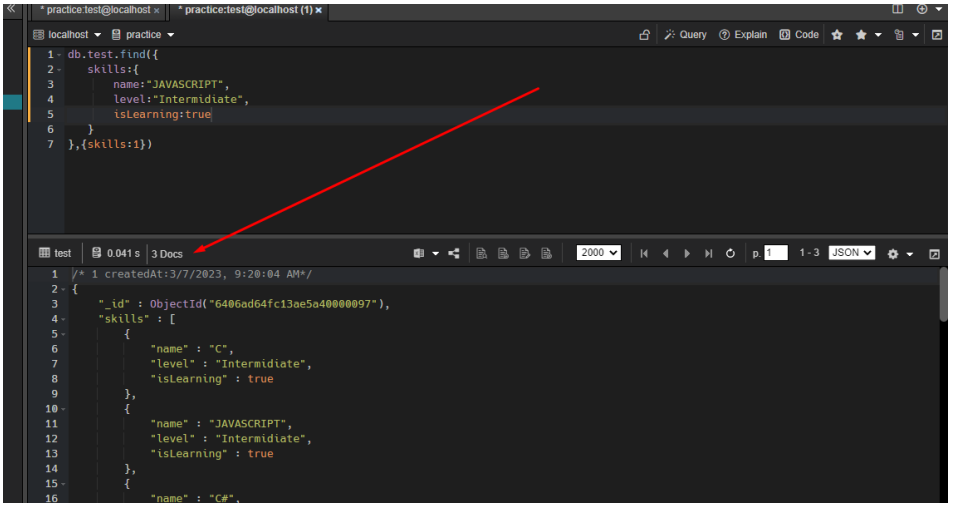


এইভাবে লিখলেও আসবে কিন্তু একটা সমস্যা আছে ।

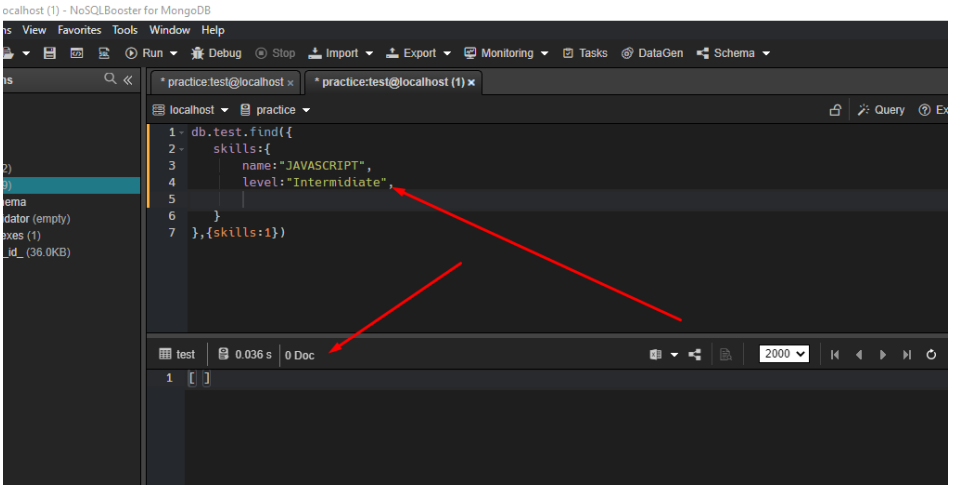
দেখা যাবে level:Expert ও দিয়ে দিছে তার মানে ঠিক মতো কাজ করে নাই ।

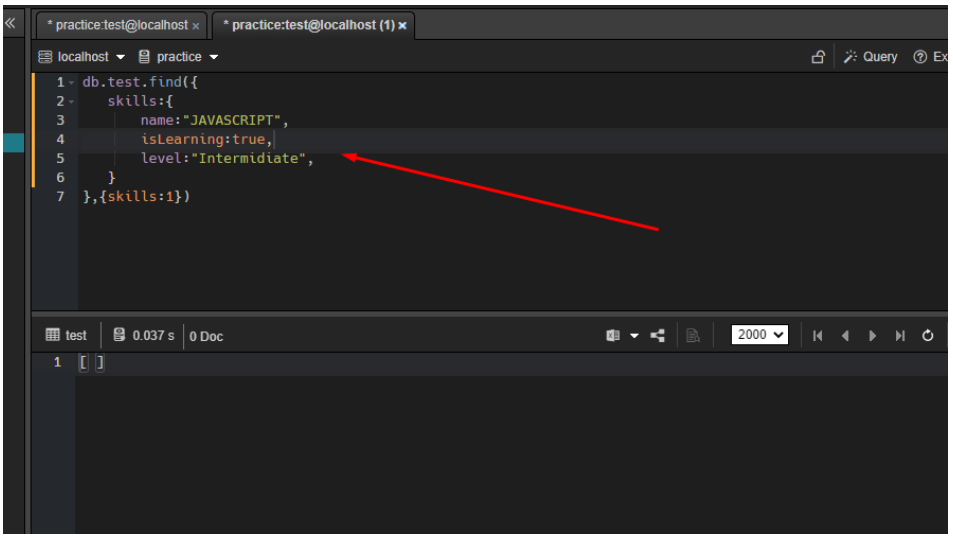


এইভাবে দিলে Array এর মতো সমস্যা । index position ঠিক থাকতে হবে তবেই আসবে ।



যদিও isLearning:true এটা তুলে দেই তাহলেঃ





উপর নিচে করলেও হবে না।

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

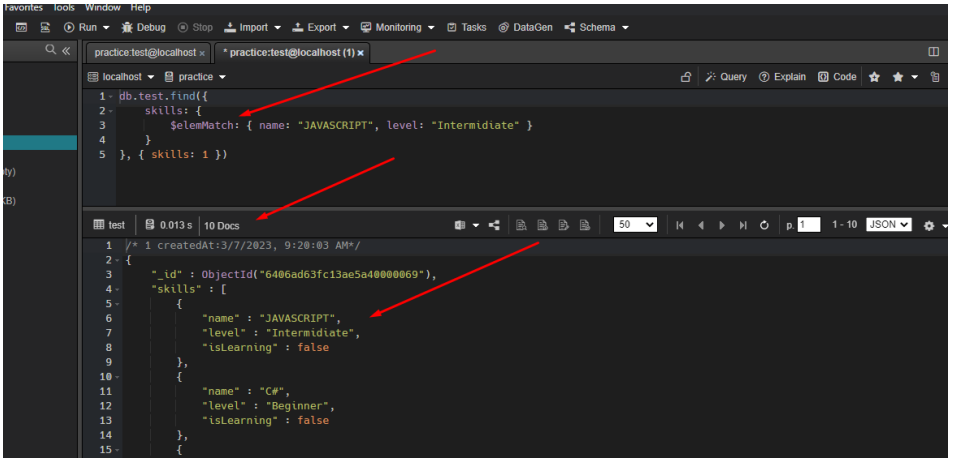
$elemMatch

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

syntax :

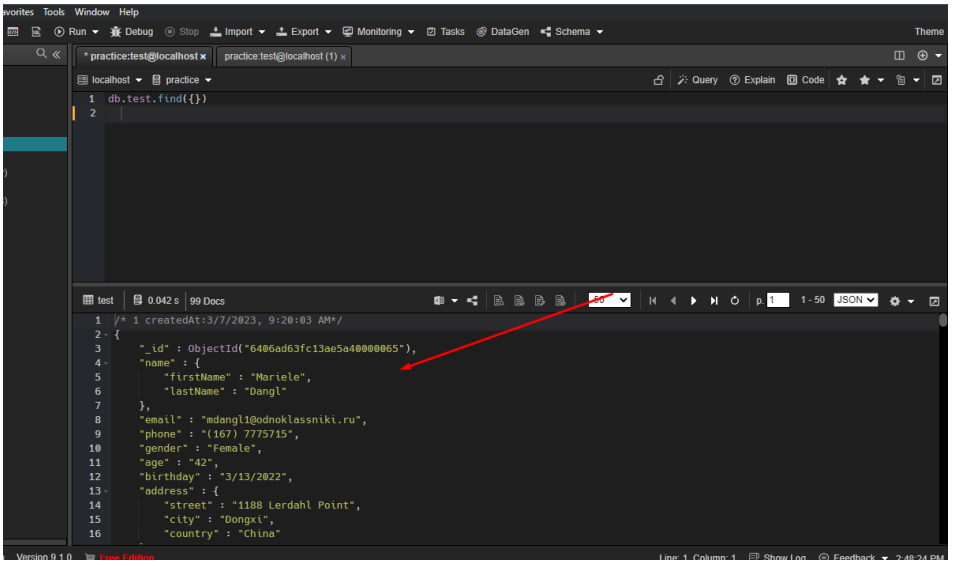
```
{ <field>: { $elemMatch: { <query1>, <query2>, ... } } }
```

Solution :

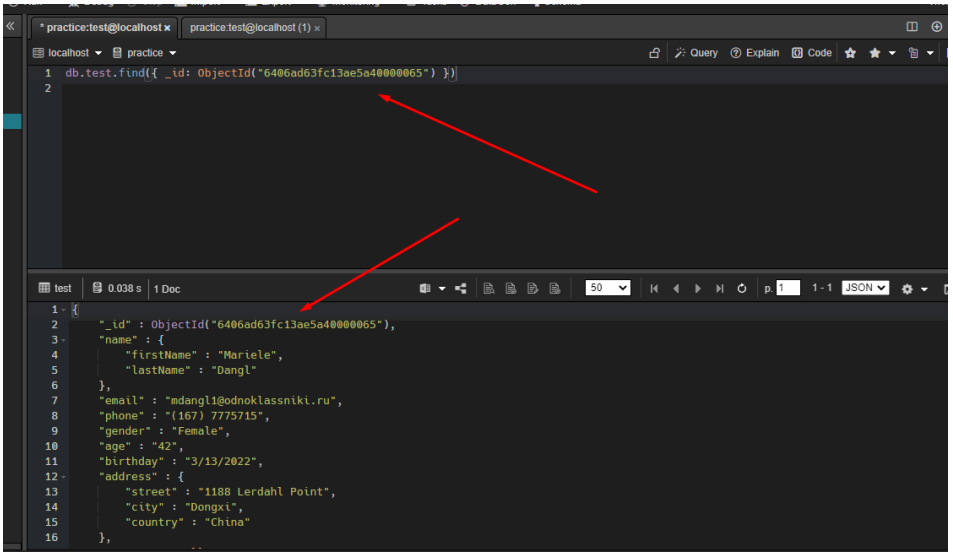


▼ Video 8 ($set, $addToSet, $push)

এখন আপডেট অপারেটর সম্পর্কে জানবো।



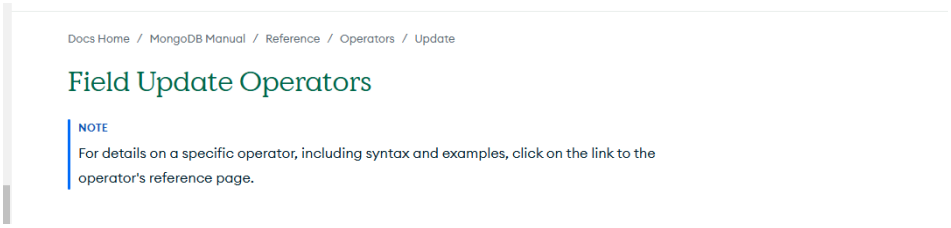
ধরি এই ডকুমেন্ট টি আপডেট করবো।



_id দিয়ে find করে নিয়ে আসলাম ।

ধরি age কে আপডেট করবো ।

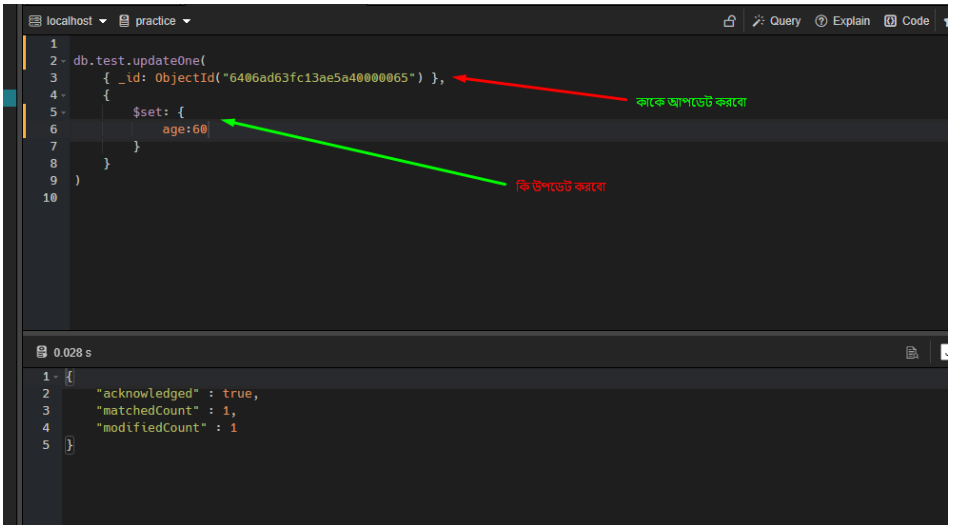
Documentation: https://www.mongodb.com/docs/manual/reference/operator/update-field/



$set

syntax:

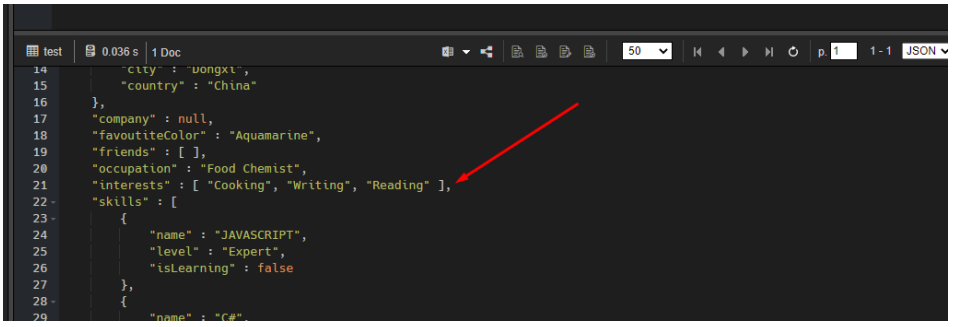
```
{ $set: { <field1>: <value1>, ... } }
```



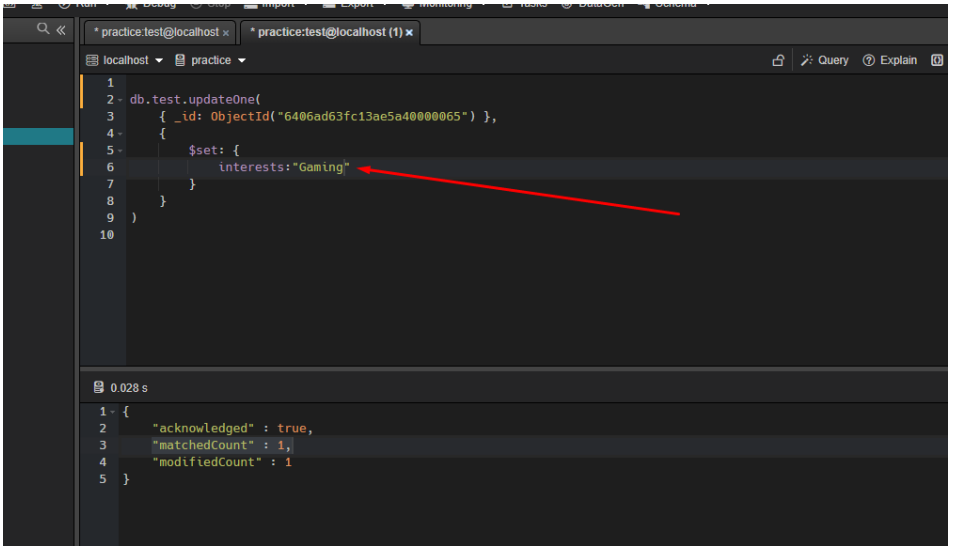
- "modifiedCount" : 1 অর্থ Modify হয়েছে ।
- "matchedCount" : 1 অর্থ কোন ডকুমেন্ট এর সাথে ম্যাচ হয়েছে ।কোন ডকুমেন্ট এর সাথে ম্যাচ হয়েছে কিনা বলে দেয় । যদি value 1 হয় তাহলে ম্যাচ হয়েছে ।

$set এর কিছু সমস্যা রয়েছে ।

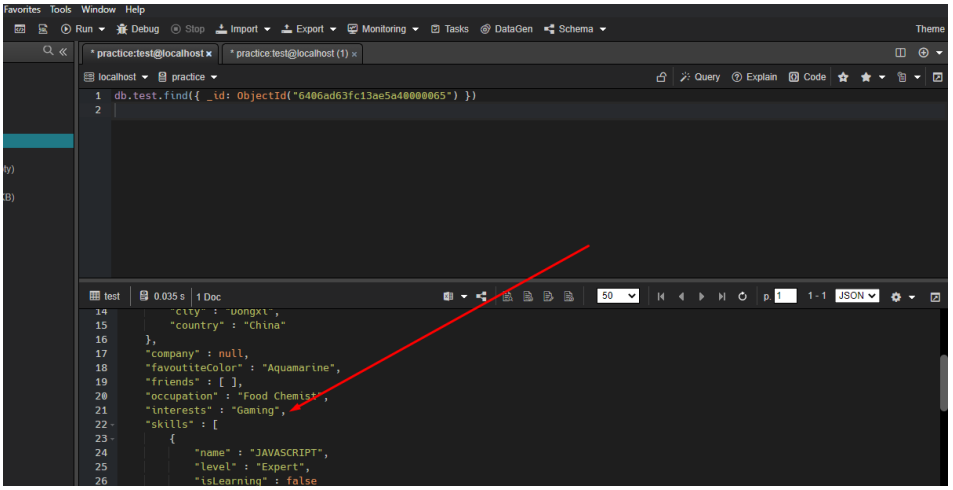
যেমনঃ



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



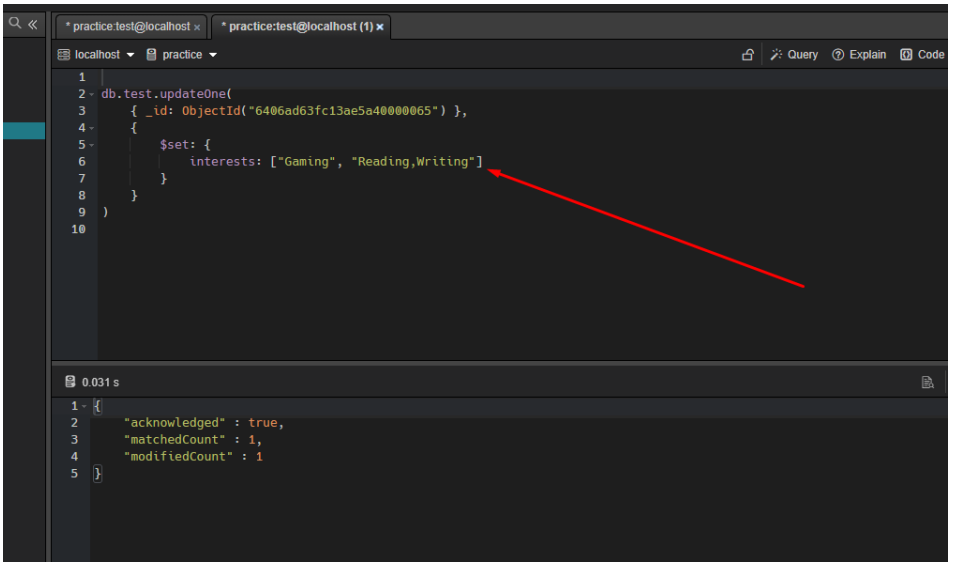
ধরি এইভাবে আপডেট করলাম ।



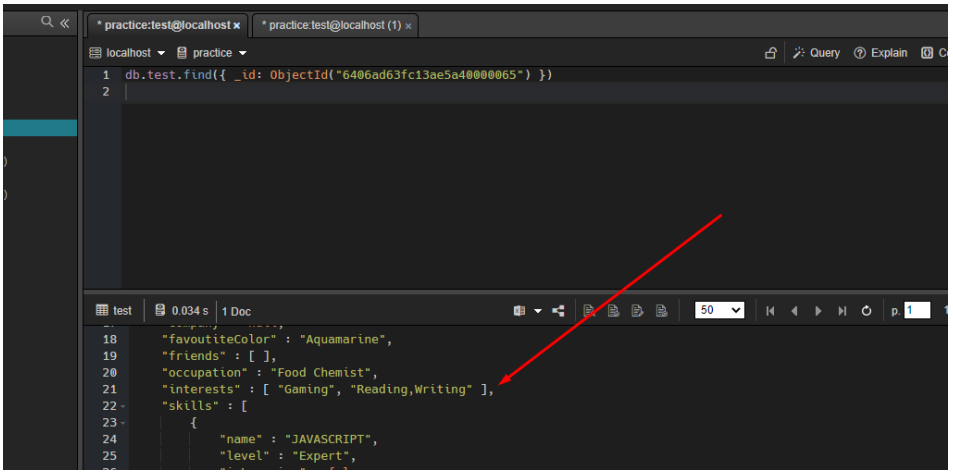
এখন কিন্তু আর Array টা নাই ।

- 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



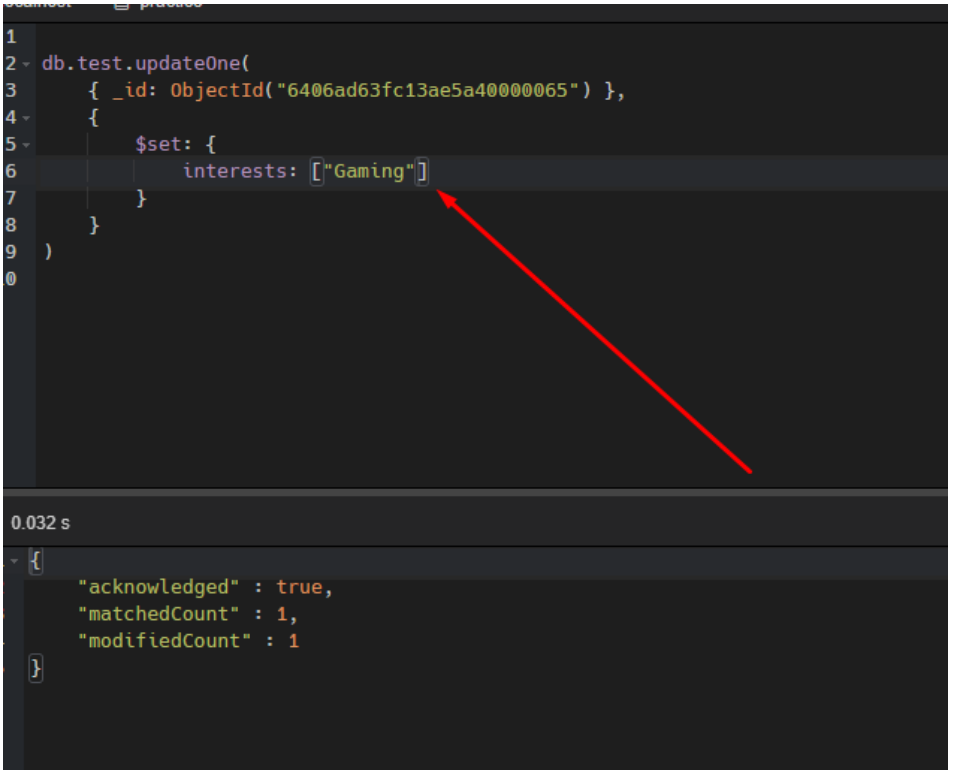
Output :

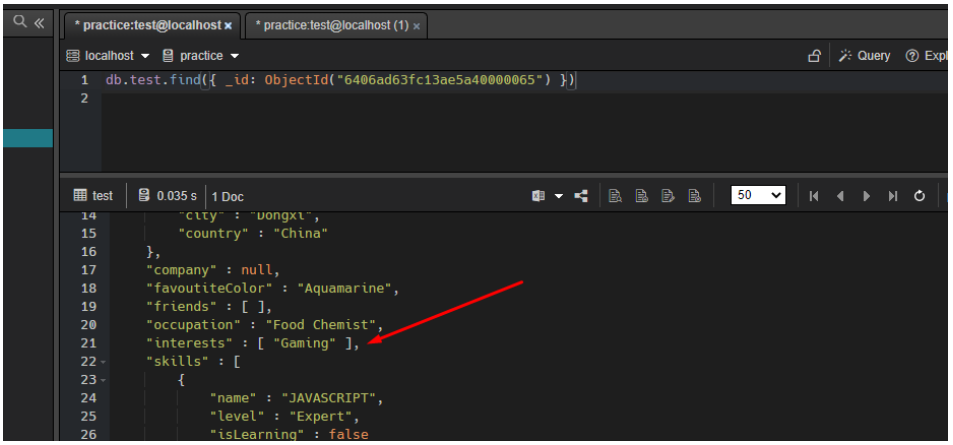


কিন্তু যদি চাই নতুন value অ্যাড করতে সেক্ষেত্রে $set ব্যবহার করলে কাজ হবে না।

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

এখন আমি চাচ্ছিঃ





এইখানে নতুন value অ্যাড করতে ।

সেক্ষেত্রে ব্যবহার করতে হবে Array Update Operator .

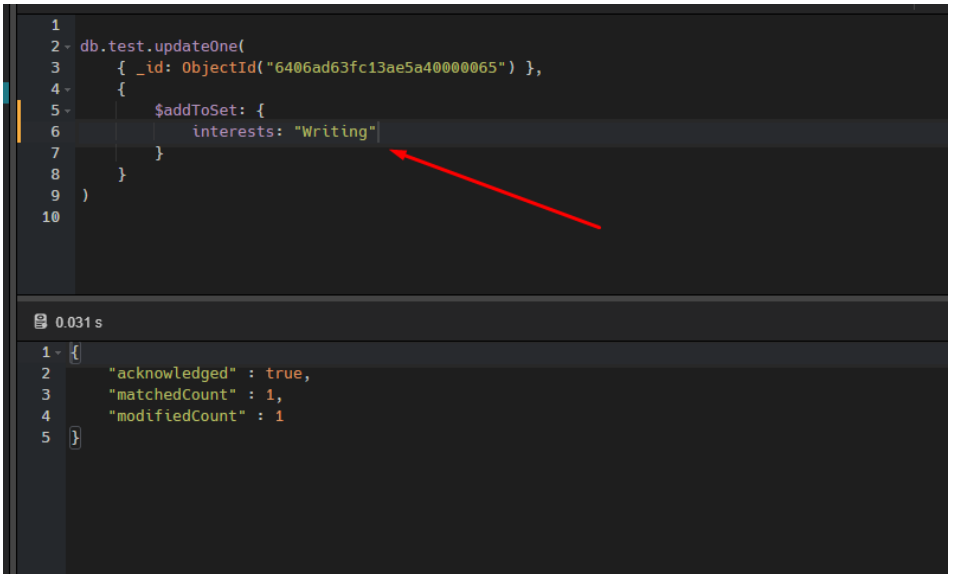
Documentation: https://www.mongodb.com/docs/manual/reference/operator/update-array/

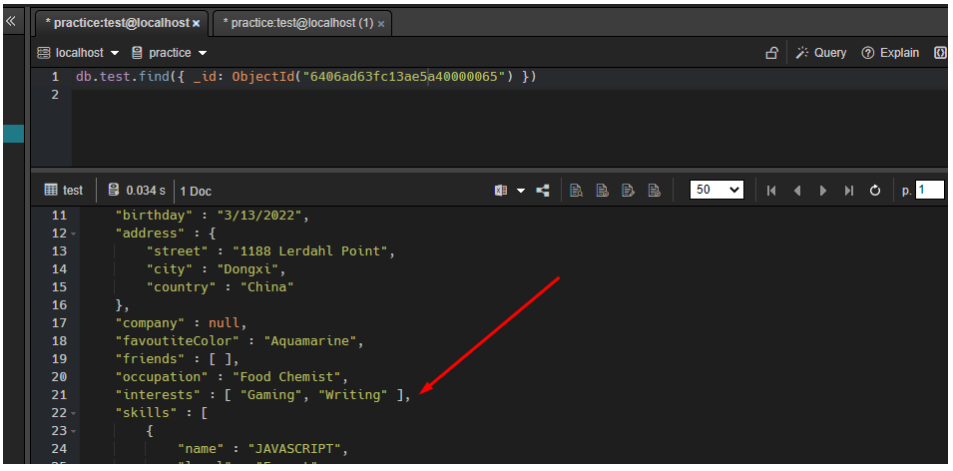
$addToSet Operator

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

syntax :

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





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

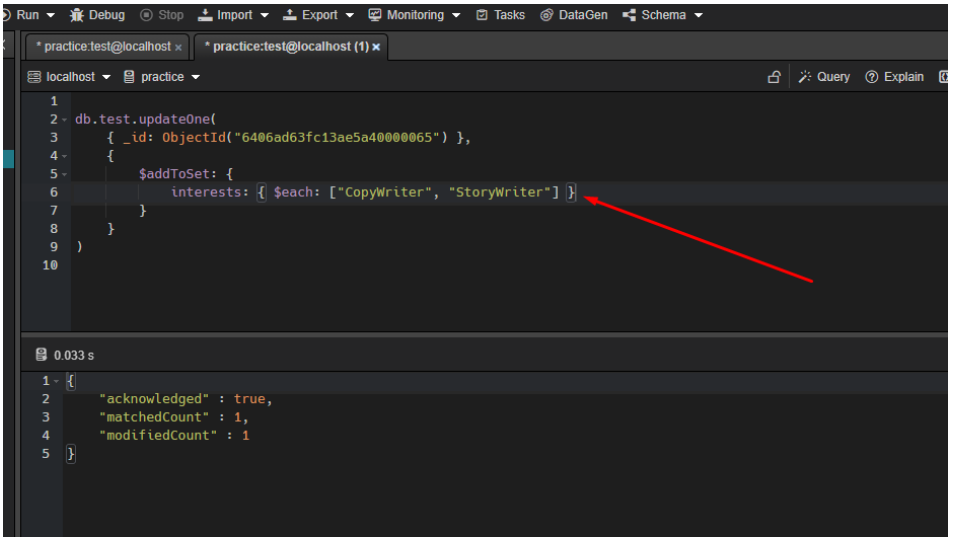
যদি চাই একাধিক value একসাথে যোগ করতে সেক্ষেত্রে $each মডিফায়ার ব্যবহার করতে হবে।

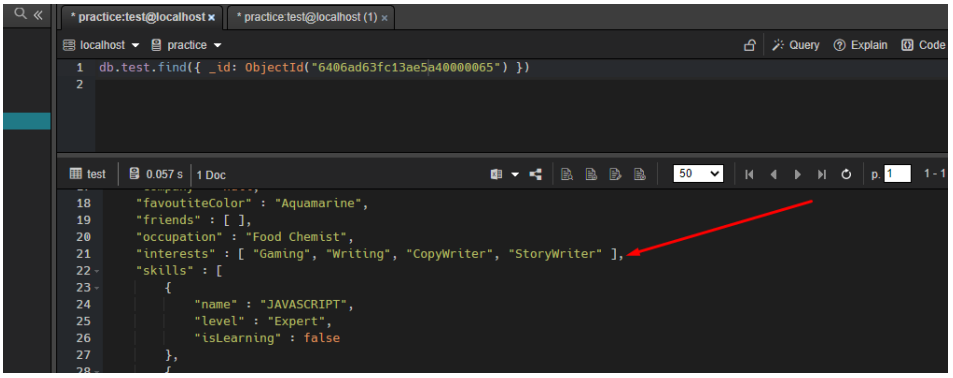
$each

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

syntax :

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

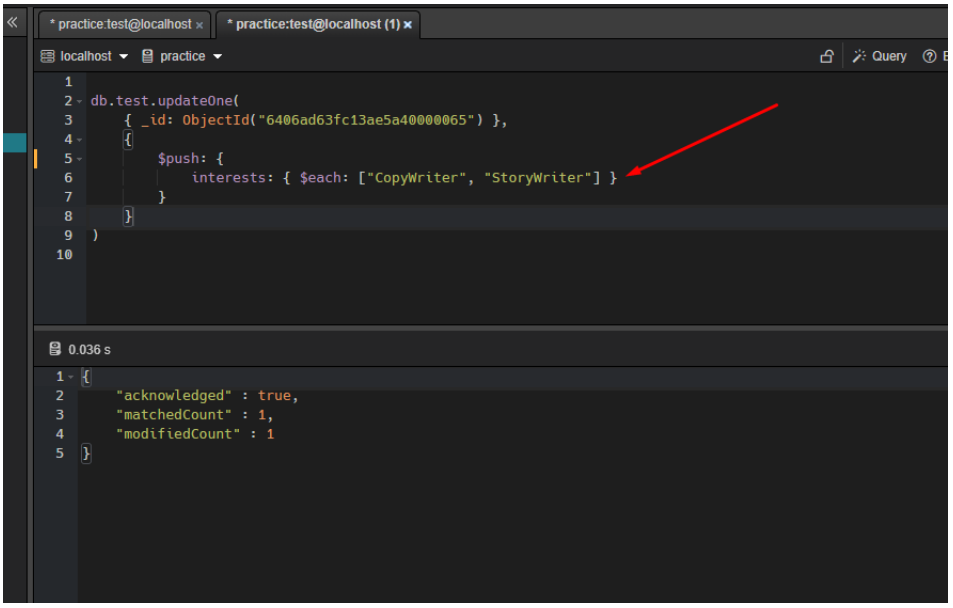


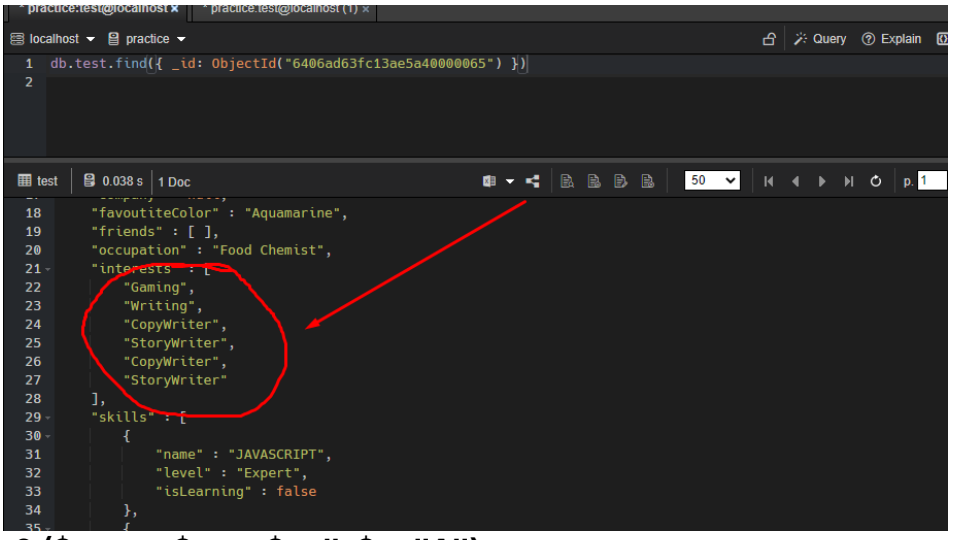


যদি মনে হয় আমি Duplicate Entry দিবো সে ক্ষেত্রে $push ব্যবহার করতে হবে ।

$push

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





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

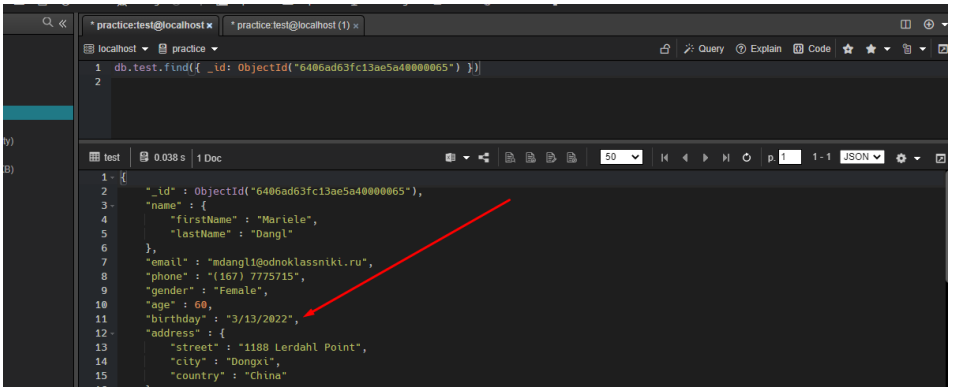
DELETE OPERATION

$unset

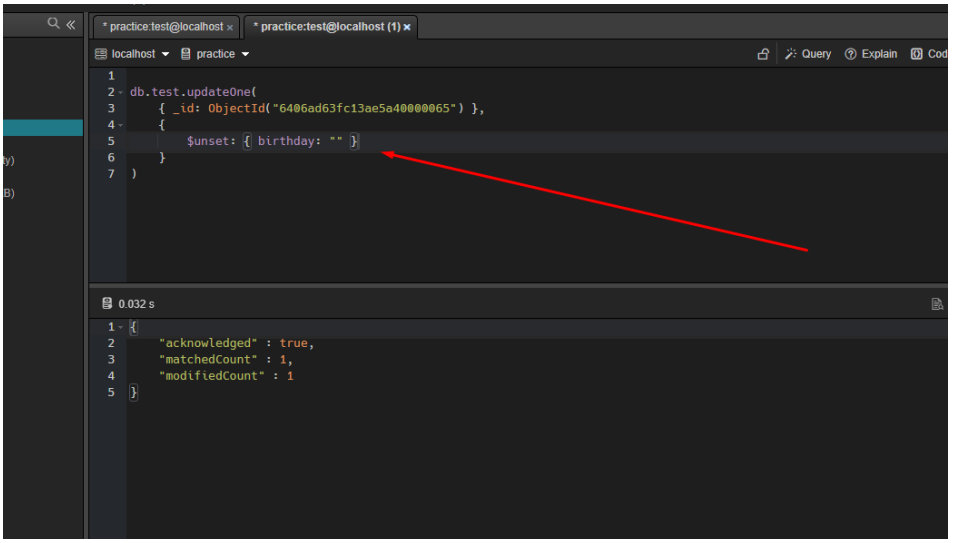
$unset হচ্ছে $set এর বিপরীত।

syntax:

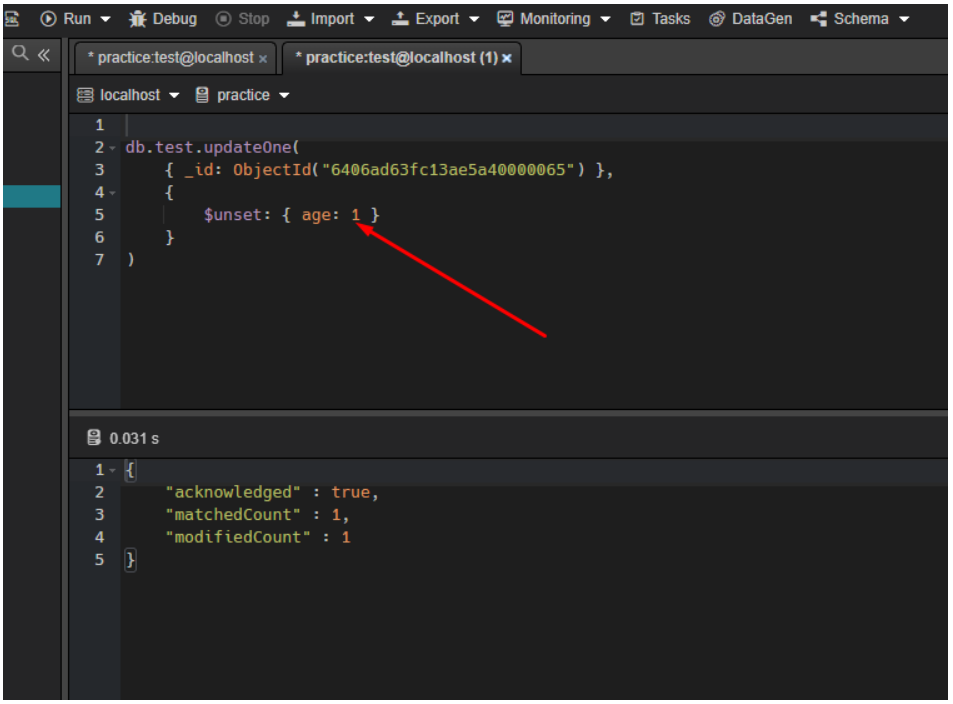
```
{ $unset: { <field1>: "", ... } }
```



ধরি Birthday field টি ডিলিট করবো।



যদি age কে ডিলিট করতে চাই তাহলেঃ



"" অথবা 1 একই কথা। অর্থাৎ true .

এখন আমরা চাই Array থেকে কোন কিছু Remove করবো। তাহলেঃ

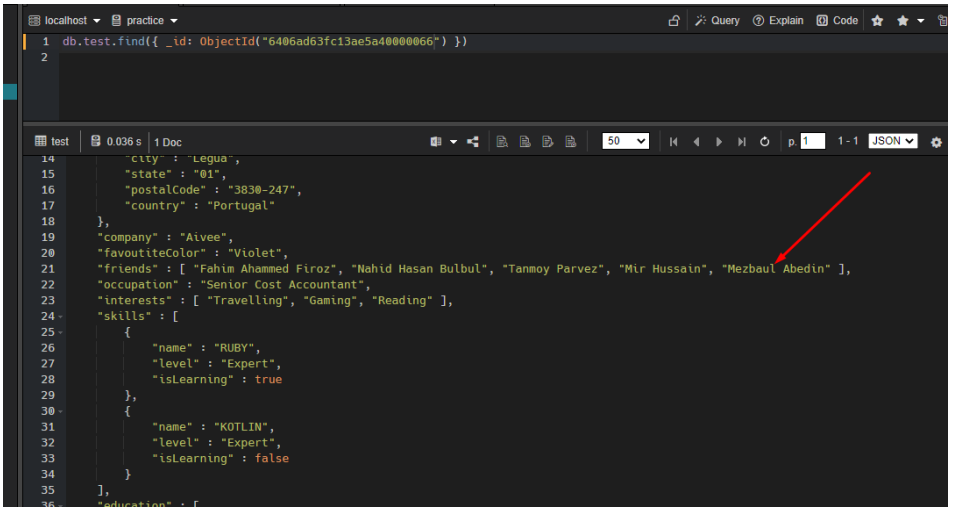
$pop

$pop এর কাজ হচ্ছে Element কে রিমুভ করে দেওয়া।

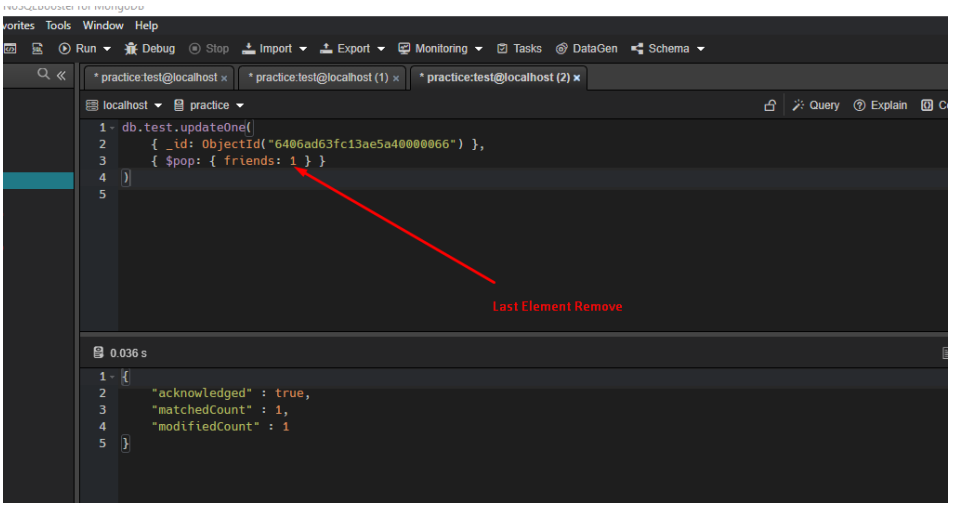
syntax:

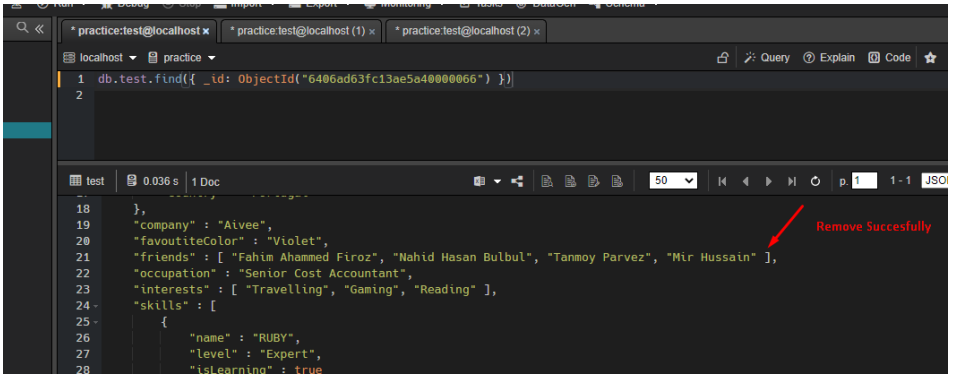
```
{ $pop: { <field>: <-1 | 1>, ... } }
```

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



ধরি এইটা রিমোভ করবো।



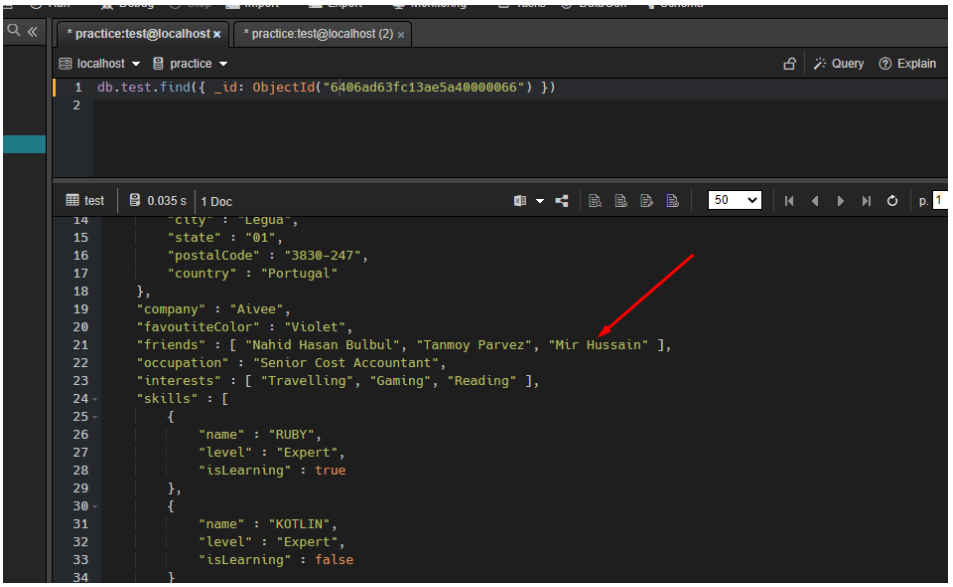


$pull

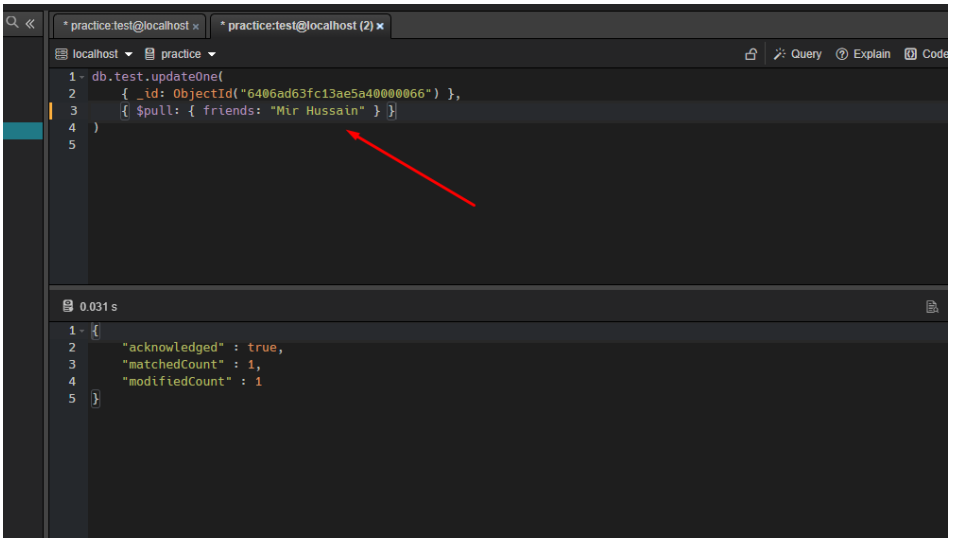
$pull ও $pop এর মতোই।

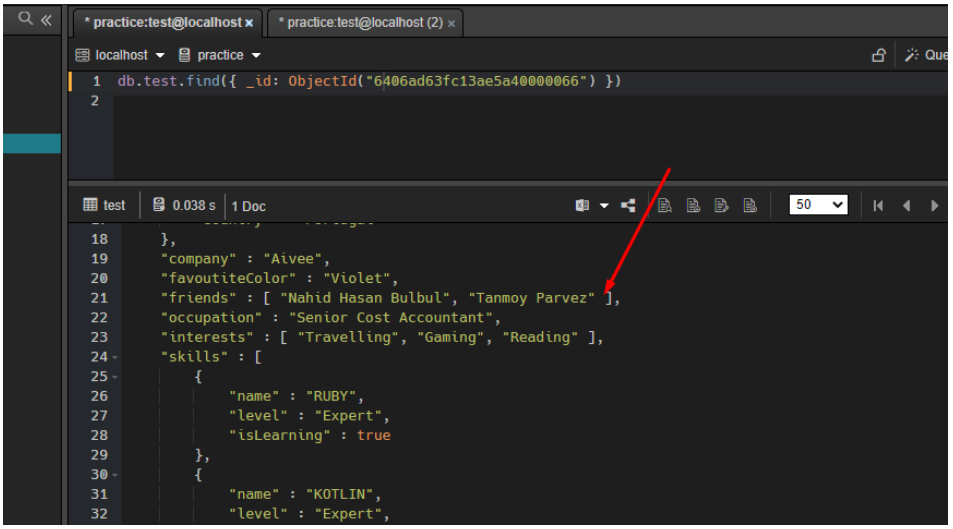
এটার কাজ কোন নির্দিষ্ট element কে পুল করে নিয়ে আসা।

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



ধরি Mir Hussain কে pull করতে চাই তাহলেঃ





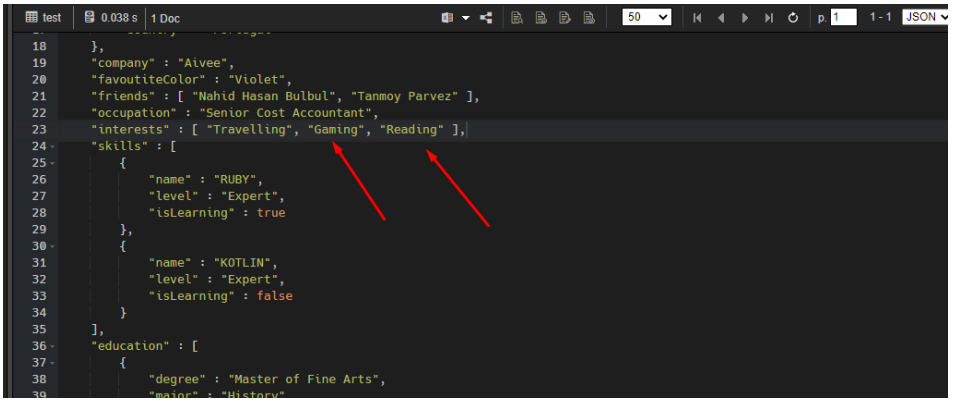
তাহলে pull এর মাধ্যমে আমরা নির্দিষ্ট element কে Delete করতে পারবো।

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

$pullAll

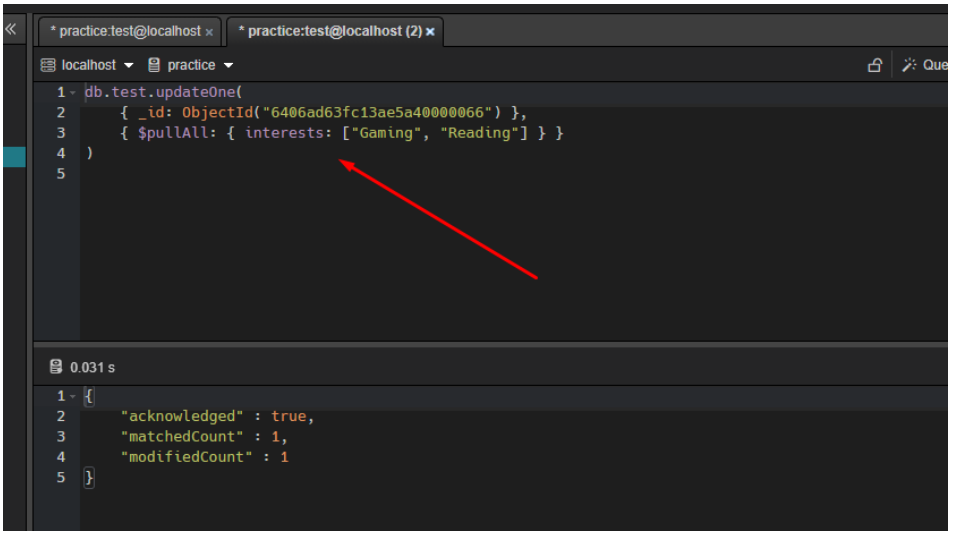
syntax:

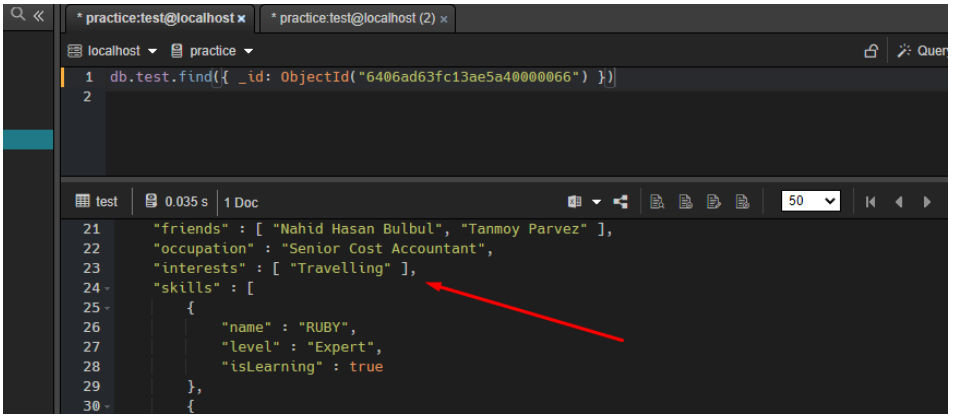
```
{ $pullAll: { <field1>: [ <value1>, <value2> ... ], ... } }
```



ধরি এই দুইটিকে ডিলিট করতে চাই।

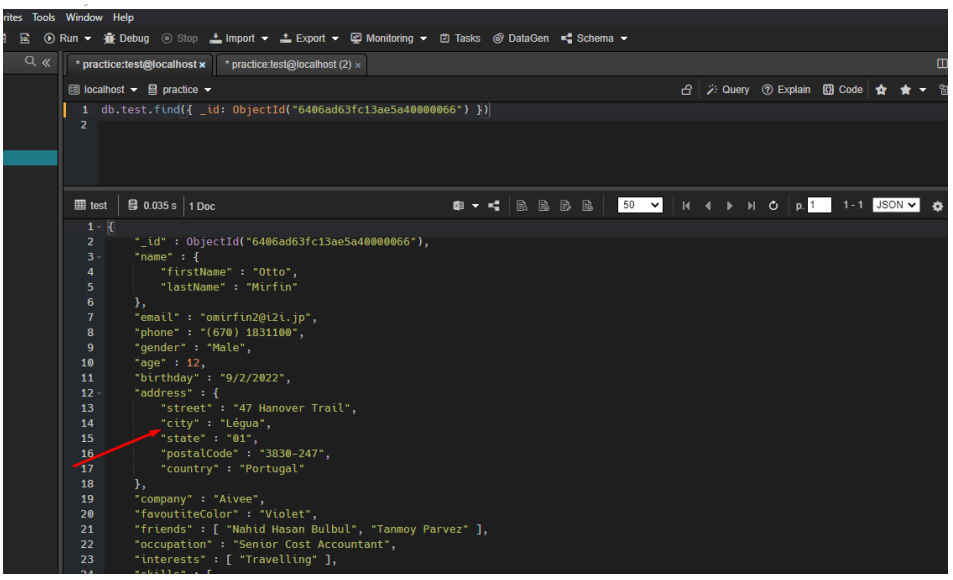
তাহলেঃ



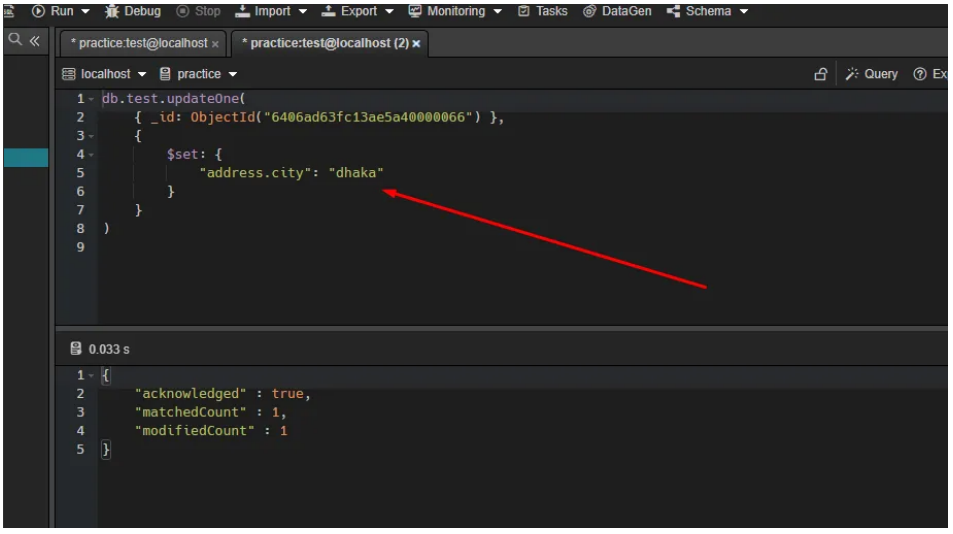


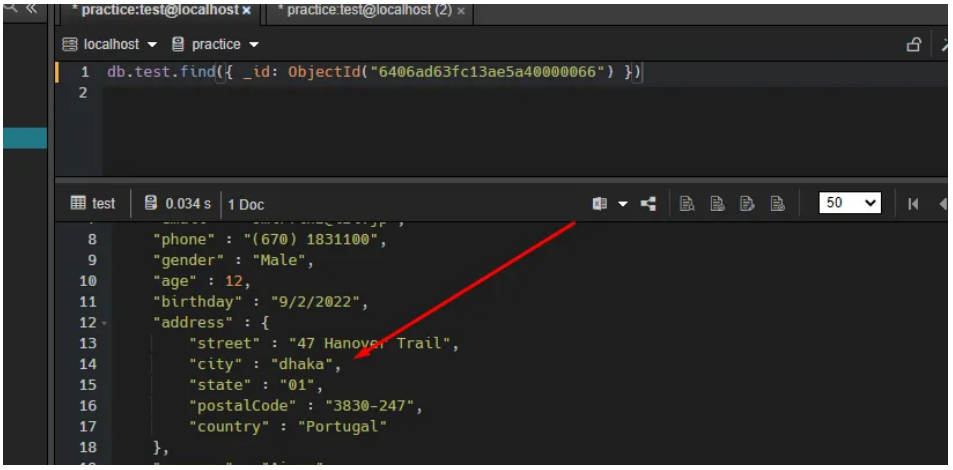
▼ Video 10 (More about $set, how to explore documentation)

$set Advanced



ধরি address এর মধ্যে city এর value পরিবর্তন করবো।





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

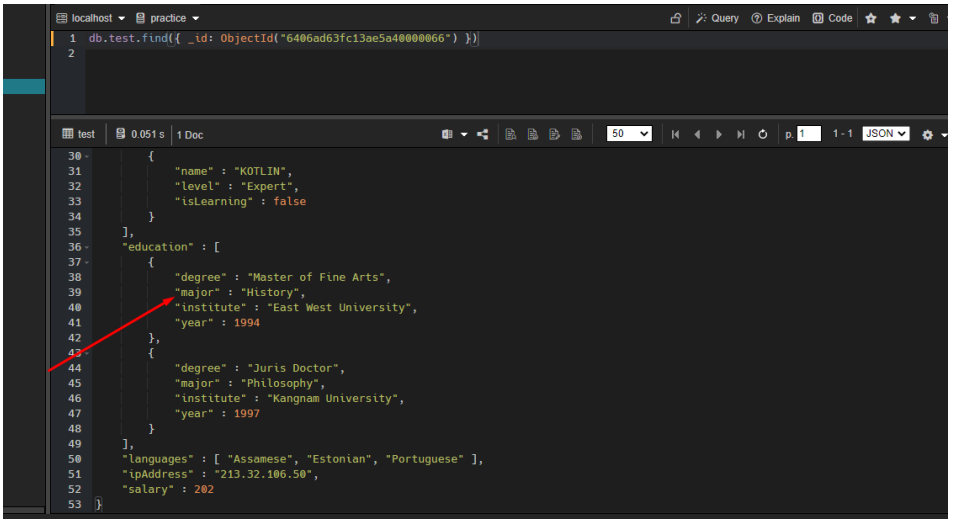
এখন যদি array of object এর ক্ষেত্রে আপডেট করতে চাই তাহলেঃ

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

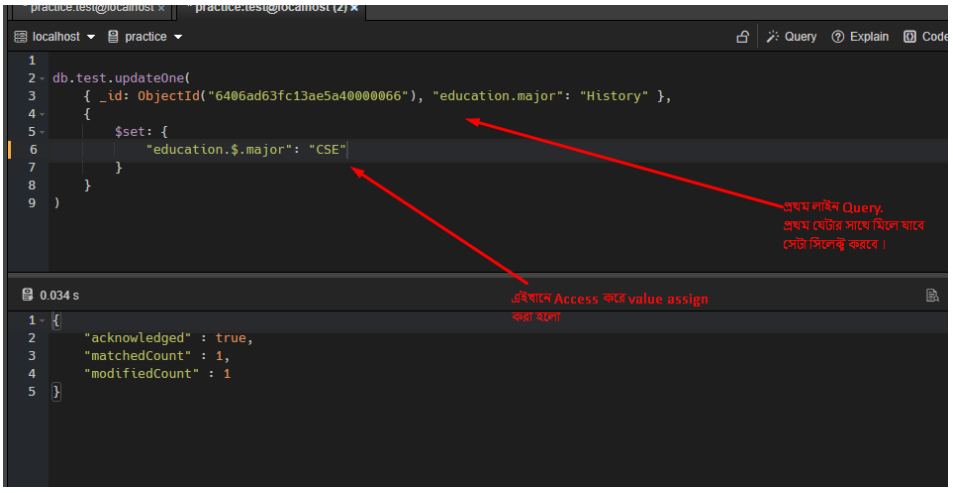
Documentation: https://sparkbyexamples.com/mongodb/update-objects-in-the-array-in-mongodb/

এইখানে দুই জায়গা তেই **$** ব্যবহার হয়েছে। এইটা একটু বুঝতে হবে।

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

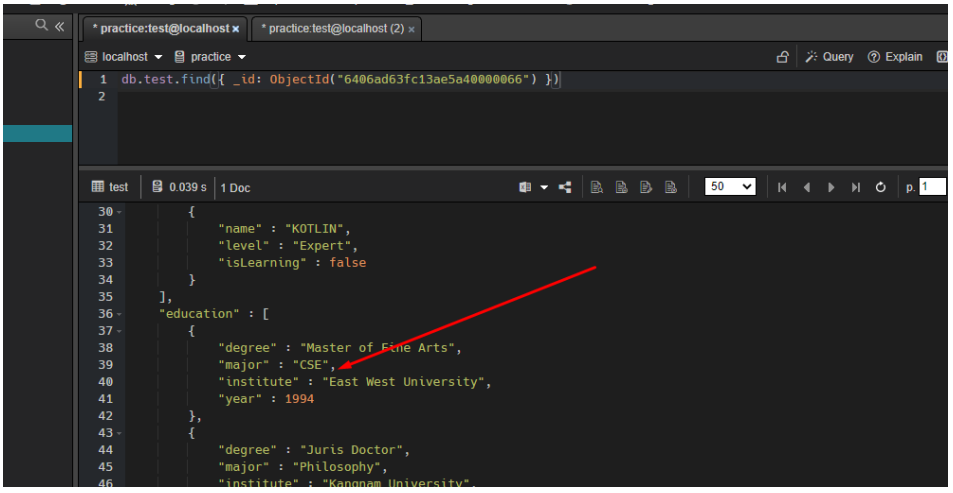


ধরি এই mejor এর value পরিবর্তন করবো।

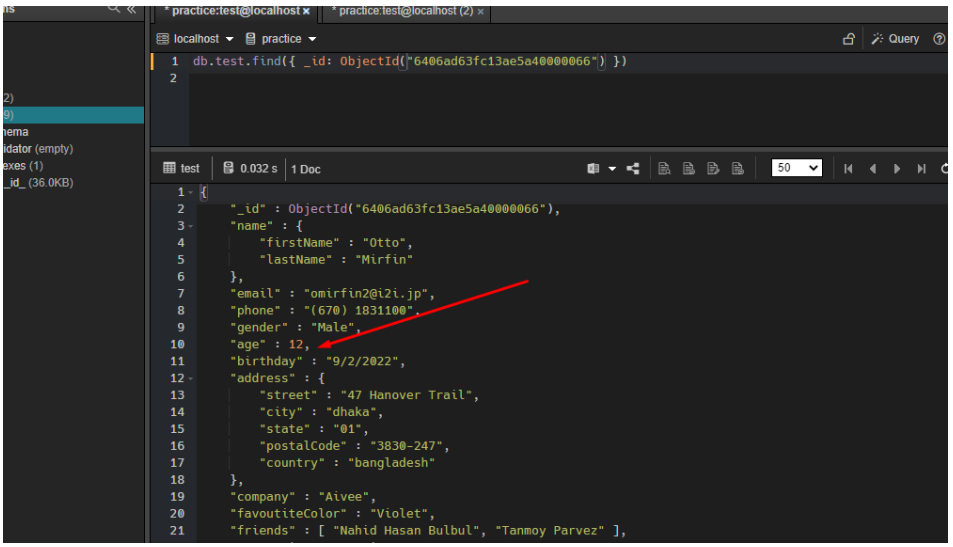


এক্ষেত্রে যখন Query করবো তখন যদি একাধিক ওটার মতো থাকে প্রথমে যেটা থাকবে সেটাকে সিলেক্ট করবে।

Output:



এখনঃ



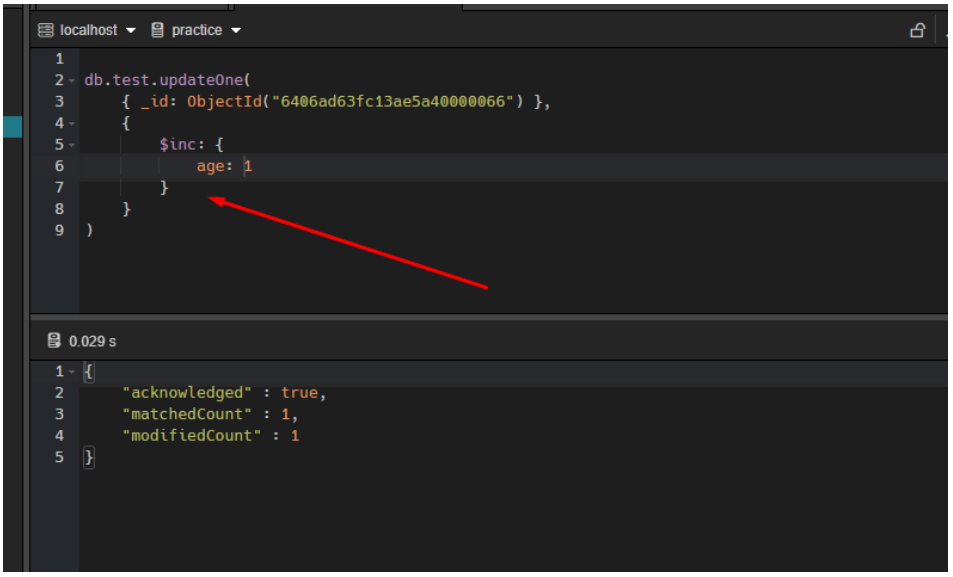
আমি চাচ্ছি age এর মান প্রতিবার ১ করে বাড়বে।

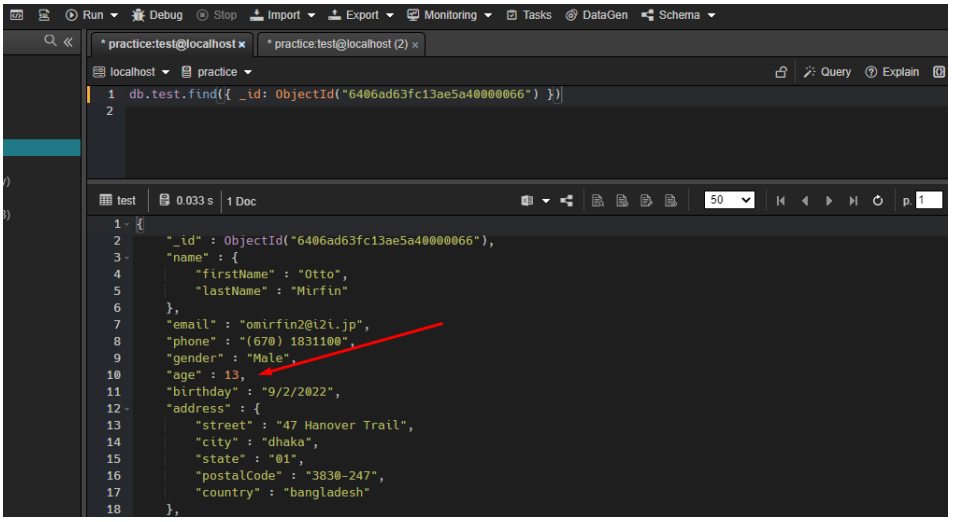
$inc

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

syntax:

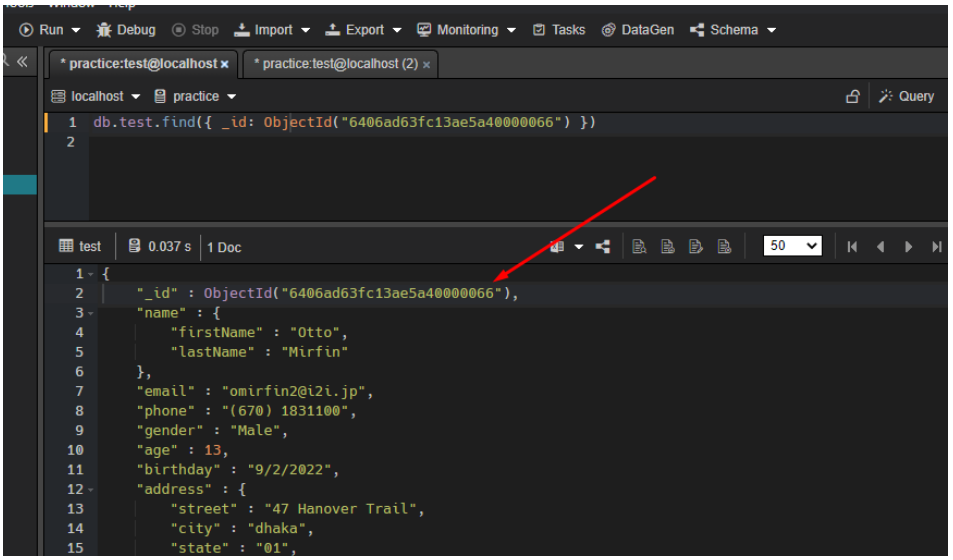
```
{ $inc: { <field1>: <amount1>, <field2>: <amount2>, ... } }
```





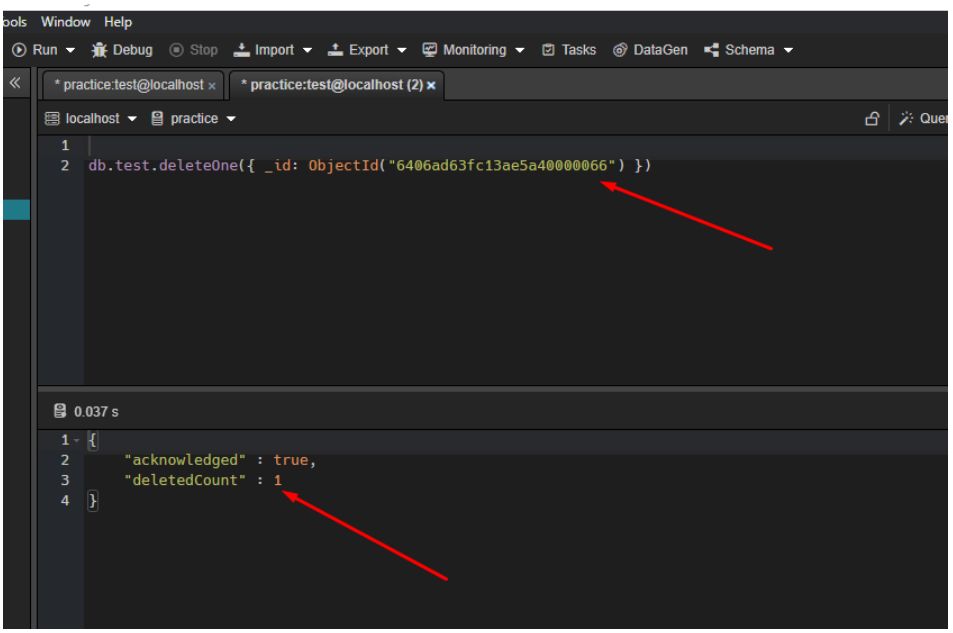
▼ Video 11 (delete documents, drop collection)

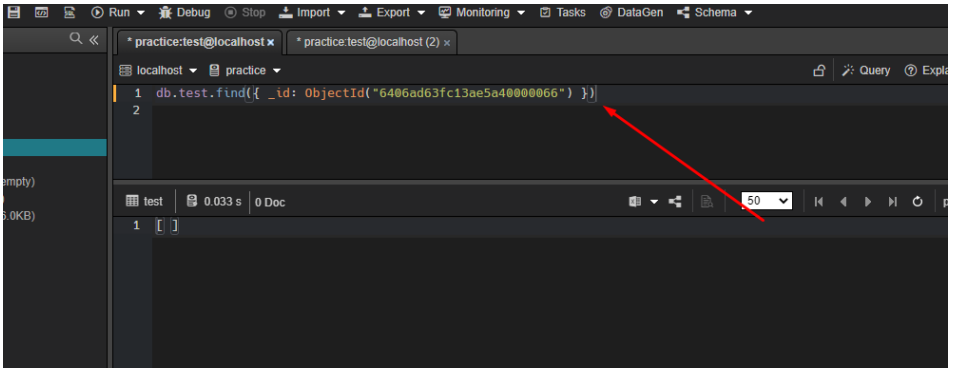
delete documents



মনে করি পুরো ডকুমেন্ট টি ডিলিট করবো ।

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





Create Collection

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

Collection Name

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 পরিক্ষা 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 ।

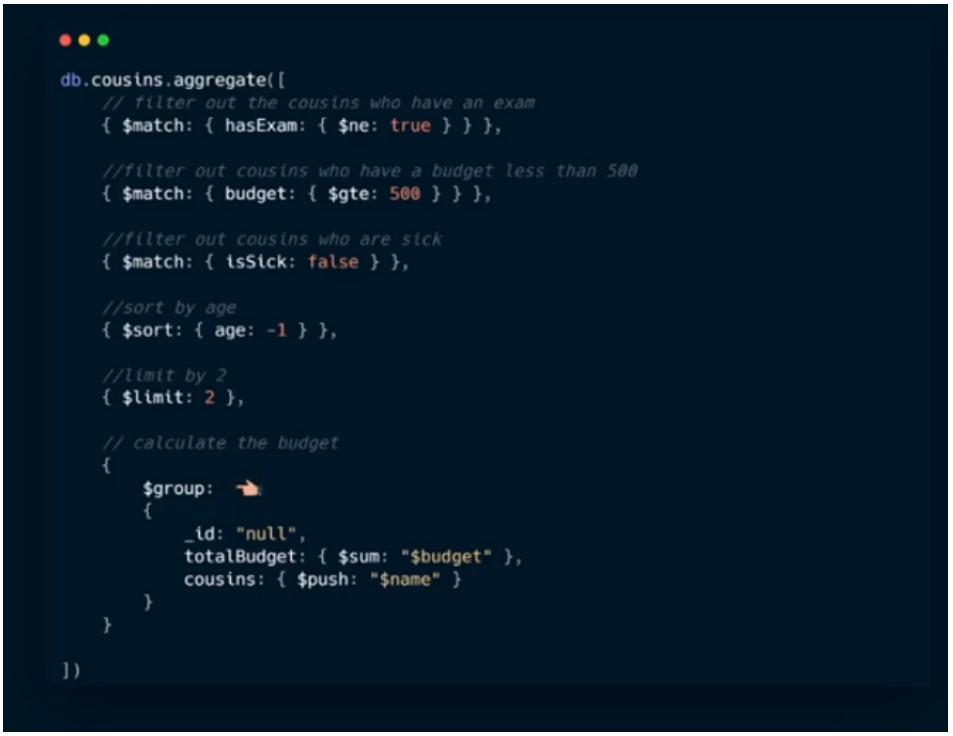
তারপর দেখা গেলো ১ জন অসুস্থ । তাহলে বাকি থাকলো শেষে ৩ জন ।

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 ।

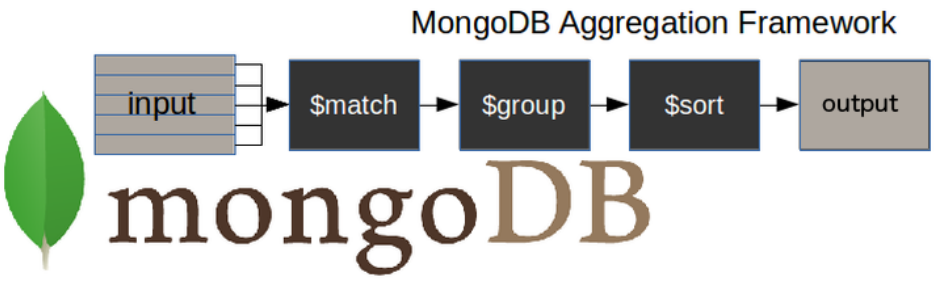
এই সম্পূর্ণ কাজটি aggregation এর মাধ্যমে করা যাবে ।



```
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 কাঙ্ক্ষিত ডকুমেন্ট গুলো ফিল্টার করে ম্যাচ করে নিয়ে আসবে ।