MongoDB basic

1. Mongodb তে একটা ডাটা insert করানোর জন্য insertOne() ফাংশান ব্যবহার করা হয়। যেমনঃ

```
db.test.insertOne({name : 'Jonathan'})
```

2. Mongodb তে একের অধিক বা অনেকগুলো ডাটা insert করার জন্য insertMany() ফাংশান ব্যবহার করা হয়। যেমনঃ

```
db.test.insertMany([
    {name : 'abir rahman'},
    {name : 'shafiqul islam'}
    ])
```

3. specific কিছু ডাটা পাওয়ার জন্য আমরা field filtering ব্যবহার করতে পারি। উদাহরনঃ

```
db.test.find({age : 17}, {name : 1, address : 1})
```

এখানে আমরা কোন ডাটাবেজের age : 17 এর সাথে মিল পাওয়া সকল ডাটা পাবো। কিন্তু পরের ব্রাকেটে `{name : 1, address : 1})` দেওয়ার কারণে age : 17 এর সাথে মিল পাওয়া ডাটাগুলোর শুধুমাত্র name এবং address ডাটাগুলো পাবো।
উপরের কোডটি আমরা project মেথড ব্যবহার করেও করতে পারি। তবে project মেথড শুধু find এর ক্ষেত্রে কাজ করে। findOne এর ক্ষেত্রে কাজ করেনা।

```
db.test.find({age : 17}).project({name : 1, address : 1})
```

3. mongodb operator দিয়েও আমরা ফিল্ড ফিল্টারিং করতে পারি। যেমনঃ

$eq :

```
    db.test.find({age : {$eq : 18}}) //-- give the all data that matches age=18
```

$gt :

```
    db.test.find({age : {$gt : 18}}) //-- give the all data that matches age>18
```

$gte :

```
    db.test.find({age : {$gte : 18}}) //-- give the all data that matches age>=18
```

$lt :

```
    db.test.find({age : {$lt : 18}}) //-- give the all data that matches age<18
```

$lte :

```
    db.test.find({age : {$lte : 18}}) //-- give the all data that matches age<=18
```

sort :

```
    db.test.find({age : {$eq : 18}}).sort({birthday : 1}) //-- give the all data that
matches age=18 and will be sorted by Birthday in ascending order
```

4. $in এর ব্যবহারঃ

```
db.test.find({age : {$gt : 18, $lt : 40}}, {age : 1}).sort({age:1})

//-- give the all data that matches
```

`age : {$gt : 18, $lt : 40} =` //-- age 18 এর উপরে এবং 40 এর নিচের ডাটাগুলো দেখাবে।

`{age : 1} =` //-- শুধু মাত্র age ডাটা দেখাবে।

`.sort({age:1})=` //-- ডাটাগুলো age অনুসারে ascending order এ sorting হয়ে দেখাবে।

```
//-- same code with one more field
db.test.find({gender :'female' ,age : {$gt : 18, $lt : 40}}, {age : 1}).sort({age:1})
```

`{gender :'female' ,age : {$gt : 18, $lt : 40}}` //-- age 18 এর উপরে এবং 40 এর, এবং শুধুমাত্র female ডাটাগুলো দেখাবে।

```
db.test.find(
    {age : {$in: [18,19,20,22,32]}},
    {age : 1})
    .sort({age:1}
    )
```

`{age : {$in: [18,19,20,22,32]}}` //-- যেসব ডাটার age এর ভ্যালু 18, 19, 20, 22, 32 সেগুলো শো করবে।

```
db.test.find(
    {age : {$nin: [18,19,20,22,32]}},
    {age : 1})
    .sort({age:1}
    )
```

`{age : {$nin: [18,19,20,22,32]}}` //-- যেসব ডাটার age এর ভ্যালু 18, 19, 20, 22, 32 সেগুলো বাদে আর সবগুলো ডাটা শো করবে।

```
db.test.find(
    {
        age: { $nin: [18, 19, 20, 22, 32] },
        interests: 'Writing'
    },
    { age: 1 })
    .sort({ age: 1 }
    )
```

`interests: 'Writing'` //-- যেসব ডাটার age এর ভ্যালু 18, 19, 20, 22, 32 সেগুলো , সেইসাথে সেই ডাটাগুলোর মধ্যে যেগুলাতে interests ফিল্ডে 'Writing' আছে সেগুলো শো করবে।

```
db.test.find(
    {
        age: { $nin: [18, 19, 20, 22, 32] },
        interests: { $in: ['Writing', 'Cooking'] }
    },
    {
        age: 1,
        interests : 1
```

```
    })
    .sort({ age: 1 }
    )
```

interests: 'Writing' //-- যেসব ডাটার age এর ভ্যালু 18, 19, 20, 22, 32 সেগুলো , সেইসাথে সেই ডাটাগুলোর মধ্যে যেগুলাতে interests ফিল্ডে 'Writing' অথবা 'Cooking' যে কোন একটি অথবা ২টিই আছে সেগুলো শো করবে।

---

5. $and, $or, implicit vs explicit :

```
db.test.find({
    $and: [
        { age: { $gt: 3, $lt: 30 } },
        { gender: 'Female' }
    ]
}).project({
    age: 1,
    gender: 1

}).sort(
    { age: 1 }
)
```

```
$and: [
        { age: { $gt: 3, $lt: 30 } },
        { gender: 'Female' }
    ]
```

//-- $and অপারেটর দ্বারা বুঝাচ্ছে এখানে [ ] ব্রাকেটের ভেতর থাকা সবগুলো শর্ত পূরণ করা ডাটাগুলো দেখাবে।

//-- তদ্রুপভাবে এখানে $and এর জায়গায় $or থাকলে বোঝাতো [ ] ব্রাকেটের মধ্যে থাকা যেকোন একটা শর্ত পূরণ করা ডাটাগুলো দেখাবে। অবশ্য $or এর কাজ আমরা $in দিয়েও করতে পারি যেটা উপরে করেছি। তবে $in শুধুমাত্র একই ফিল্ডের ক্ষেত্রে কাজ করে।

---

6. array উপর mongodb অপারেশানঃ

একটি array-র যেকোন একটা ডাটা অনুযায়ী সার্চের নিয়মঃ

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

এখানে "interests" : ["Cooking", "Writing", "Reading"] এর মত ডাটার ভেতর cooking থাকলেই সেগুলো শো করবে।

আর যদি এরকম লিখিঃ

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

তবে interests এরের মধ্যে ঠিক `[ "Cooking", "Writing", "Reading" ]` ঠিক এইভাবে সিরিয়াল অনুযায়ী সাজানো আছে যে ডাটাগুলো , শুধু সেগুলো দেখাবে। এখন আমরা যদি চাই জাস্ট এই ৩টা ডাটা থাকলেই হবে , সিরিয়াল যেমন ইচ্ছা তেমন থাকুক তবে সেটা $all দিয়ে করতে হবে ,

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

---

7. array of objects [{ }] এর উপর mongodb অপারেশানঃ

```
db.test.find(
    {
        "skills.name" : 'JAVASCRIPT'
    })
    .project({
        skills : 1
    })
```

উপরের কোডটি skills নামক array-র ভেতর যেসব অবজেক্ট আছে সেসব অবজেক্টের মধ্যে name প্রপার্টিতে JAVASCRIPT পেলেই সেগুলো শো করবে।

```
db.test.find(
    {
        skills: {
            "name": 'JAVA',
            "level": 'Expert',
            "isLearning": false
        }
    })
    .project({
        skills: 1
    })
```

উপরের কোডটি এরের মত এক্স্যাক্ট সিরিয়াল অনুযায়ী ডাটা দিবে । অর্থাৎ যে ডাটাগুলোতে

```
skills: {
            "name": 'JAVA',
            "level": 'Expert',
            "isLearning": false
        }
```

সিরিয়াল মত থাকবে, শুধু সেই ডাটাগুলোই দিবে। কিন্তু আমরা যদি চাই এখানে array এর $all মেথডের মত করে ডাটা শো করতে অর্থাৎ query তে দেওয়া ডাটাগুলো থাকলেই হলো, সিরিয়াল মেইন্টেইন ম্যান্ডাটরি না , তবে আমরা $elemMatch অপারেটর ব্যবহার করতে পারি,

```
db.test.find(
    {
```

```
        skills: {
            $elemMatch: {
                "level": 'Expert',
                "name": 'JAVA'
            }
        }
    })
    .project({
        skills: 1
    })
```