


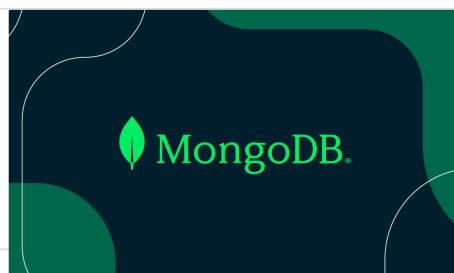
MongoDB Queries

Resource Link:

Comparison Query Operators

.leafygreen-ui-zzkys8{font-size:16px;line-height:28px;font-family:'Euclid Circular A','Helvetica Neue',Helvetica,Arial,sans-serif;display:-webkit-inline-block;display:-webkit-inline-flex;display:-ms-inline-

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



যখনই কোন একটা অপারেটর আসবে তখনই দুই পাশে একটা সেকেন্ড ব্র্যাকেট {} হবে

Comparison Query Operators

<u>\$eq</u>	Matches values that are equal to a specified value.
<u>\$gt</u>	Matches values that are greater than a specified value.
<u>\$gte</u>	Matches values that are greater than or equal to a specified value.
<u>\$in</u>	Matches any of the values specified in an array.
<u>\$lt</u>	Matches values that are less than a specified value.
<u>\$lte</u>	Matches values that are less than or equal to a specified value.
<u>\$ne</u>	Matches all values that are not equal to a specified value.
<u>\$nin</u>	Matches none of the values specified in an array.

▼ \$eq

Syntax:

{ <fieldName>: { \$eq: <fieldValue> } }

Example:

```
db.collection.find({
  age: { $eq: 19 }
})
```

▼ \$gt

Syntax:

{ field: { \$gt: value } }

Example:

```
db.collection.find({
  age: { $gt: 19 }
})
```

▼ \$gte

Syntax:

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

Example:

```
db.collection.find({
  age: { $gte: 19 }
})
```

▼ \$in

Array of values এর মধ্যে যেকোনো একটি ভেলোর সাথে \$in এর ভেলোর যেকোনো একটি ভ্যালু কে এক্সিস্ট করলেই হবে।

Syntax:

{ <fieldName>: { \$in: <fieldValue> } }

Example:

```
db.collection.find({
  friends: { $in: ["Abdur Rakib", "Emdadullah"] }
}).projection({ friends: 1 })
```

▼ \$lt

Syntax:

{ field: { \$lt: value } }

Example:

```
db.collection.find({
  age: { $lt: 19 }
})
```

▼ \$lte

Syntax:

```
{ field: { $lte: value } }
```

Example:

```
db.collection.find({
  age: { $lte: 19 }
})
```

▼ \$ne

Syntax:

```
{ field: { $ne: value } }
```

Example:

```
db.collection.find({
  age: { $ne: 19 }
})
```

▼ \$nin

\$nin ভেলুগুলো যদি কোন এ রাতে একজিস্ট না করে

Syntax:

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

Example:

```
db.collection.find({
  friends: { $nin: ["Abdur Rakib", "Emdadullah"] }
}).projection({ friends: 1 })
```

Logical Query Operators

▼ \$and

Array of expression এর এক বা একাধিক এক্সপ্রেশন সত্য হতে হবে

Syntax:

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

Example:

```
db.collection.find({
  $and: [
    { age: { $lt: 19 } },
    { friends: { $in: ["Fahim Ahammed Firoz", "Nahid Hasan Bulbul"] } }
  ]
}).projection({ friends: 1, age: 1 })
```

▼ \$or

Array of expression এর যেকোনো একটি এক্সপ্রেশন সত্য হলেই হবে

Syntax:

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

Example:

```
db.collection.find({
  $or: [
    { age: { $lt: 20 } },
    { salary: 100 }
  ]
}).projection({ salary: 1, age: 1 })
```

Array Query Operators

▼ \$all

Array এর এলিমেন্টের পজিশন যাই হোক না কেন ডাটা ম্যাচ করলেই সেটা কাজ করবে, এটি অনেকটা \$and অপারেটর এর মত কাজ করে

Syntax:

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

Example:

```
db.collection.find({
  friends: { $all: ["Mir Hussain", "Abdur Rakib", "Fahim Ahammed Firoz", "Tanmoy Parvez"] }
})
.projection({ friends: 1 })
```

▼ \$elemMatch — Array of object

Array of object এর এলিমেন্ট ম্যাচ করাতে চাইলে **\$elemMatch** ব্যবহার করতে হবে, এটি পজিশন ম্যাটার করে না

Syntax:

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

Example:

```
db.collection.find({
  education: { $elemMatch: {
    major: "Art",
    year: 2003
  } }
})
.projection({education: 1})
```

▼ \$size

এটা array সাইজ চেক করে এবং কোন একটা array এমটি কিনা সেটা চেক করে

Syntax:

```
db.collection.find( { field: { $size: 1 } } );
```

Example:

```
db.collection.find({  
  education: { $size: 3 }  
})  
  .projection({ education: 1 })
```

Element Query Operators

▼ \$exists

কোন একটা ফিল্ড ডকুমেন্ট এর মধ্যে এক্সিস্ট করে কিনা সেটা চেক করার জন্য \$exists ব্যবহার করা হয়।

Syntax:

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

Example:

```
db.collection.find({  
  education: { $exists: true }  
})  
  .projection({ education: 1 })
```

▼ \$type

\$type অপারেটর ব্যবহার করে কোন একটা ফিল্ডের টাইপ চেক করা হয়

Syntax:

```
{ field: { $type: <Enter checking type> } }
```

Example:

```
db.collection.find({
  "education.year": { $type: "number" }
})
.projection({ education: 1 })
```

Field Update Operators

▼ \$Set

কোন একটা ফিল্ডার এর ভ্যালু কে নতুন ভ্যালু দিয়ে রিপ্লেস করে

Syntax:

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

Example:

```
db.collection.updateOne(
  { _id: ObjectId("6406ad63fc13ae5a40000065") },
  {
    $set: { salary: 24000 }
  },
  { new: true }
)
```

Array Update Operators

▼ \$addToSet

এটি array তে নতুন ভ্যালু ইনসার্ট করার আগে চেক করে ভ্যালু টি এর মধ্যে অলরেডি এক্সিস্ট করে কিনা। যদি এক্সিস্ট না করে তাহলে নতুন ভ্যালু ইনসার্ট করবে এবং এটি সব সময় নতুন ভ্যালু ইনসার্ট করে কিন্তু Push মেথড ডুপ্লিকেট ভ্যালু ইনসার্ট করে থাকে। এই হলো এর অ্যাডভান্টেজ।

Syntax:

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

Example:

```
db.data.updateOne(  
  { _id: ObjectId("6406ad63fc13ae5a40000065") },  
  { $addToSet: { friends: "emdad" } }  
)
```