Logical Operators

Logical operators allow you to combine multiple query conditions. They help you create more complex queries by combining or negating conditions.

Operator	Meaning	Example Usage
`$and`	All conditions must be true	`{ $and: [ {a:1}, {b:2} ] }`
`$or`	At least one condition must be true	`{ $or: [ {a:1}, {b:2} ] }`
`$not`	Negate a condition	`{ field: { $not: { $eq: value } } }`
`$nor`	None of the conditions can be true	`{ $nor: [ {a:1}, {b:2} ] }`

1. `$and` - All conditions must be true

- Selects documents where **every** condition inside `$and` is true.

Example 1:
Find products that are in category `"Electronics"` **AND** have `stock` greater than 50.

```
db.products.find({
  $and: [
    { category: "Electronics" },
    { stock: { $gt: 50 } }
  ]
});
```

Result: `[`

```
  { _id: 1, name: "Wireless Mouse", category: "Electronics", stock:
200, ... },
  { _id: 5, name: "Bluetooth Speaker", category: "Electronics", stock:
120, ... }
]
```

Example 2:
Find products priced between 20 and 100 **AND** are available (`is_available: true`).

```
db.products.find({
  $and: [
    { price: { $gte: 20, $lte: 100 } },
    { is_available: true }
  ]
});
```

Result: `[`

```
  { _id: 3, name: "Yoga Mat", price: 45, is_available: true, ... },
  { _id: 7, name: "Wireless Charger", price: 80, is_available: true, ... }
]
```

---

2. `$or` - At least one condition must be true

- Selects documents where **any** one (or more) of the conditions inside `$or` is true.

Example 1:
Find products that are either `"Fitness"` category **OR** tagged as `"wireless"`.

```
db.products.find({
  $or: [
    { category: "Fitness" },
    { tags: "wireless" }
  ]
});
```

Result:`[`

```
  { _id: 2, name: "Dumbbells", category: "Fitness", ... },
  { _id: 1, name: "Wireless Mouse", tags: ["wireless", "computer"], ... }
]
```

Example 2:
Find products with price less than 20 **OR** stock less than 80.

```
db.products.find({
  $or: [
    { price: { $lt: 20 } },
    { stock: { $lt: 80 } }
  ]
});
```

Result: `[`

```
  { _id: 4, name: "Notebook", price: 15, stock: 100, ... },
  { _id: 6, name: "Smart Light", price: 25, stock: 40, ... }
]
```

---

3. `$not` - Negates a condition

- Selects documents **where the condition is NOT true**.

Example 1:
Find products where category is **NOT** `"Electronics"`.

```
db.products.find({
  category: { $not: { $eq: "Electronics" } }
```

```
});
```

Result: `[`

```
  { _id: 2, name: "Dumbbells", category: "Fitness", ... },
  { _id: 3, name: "Yoga Mat", category: "Fitness", ... }
]
```

Example 2:
Find products with price **NOT** greater than 50 (so price ≤ 50).

```
db.products.find({
  price: { $not: { $gt: 50 } }
});
```

Result: `[`

```
  { _id: 3, name: "Yoga Mat", price: 45, ... },
  { _id: 4, name: "Notebook", price: 15, ... }
]
```

---

4. `$nor` - None of the conditions should be true

- Selects documents where **none** of the specified conditions are true.

Example 1:
Find products that are **neither** in category `"Smart Home"` **nor** tagged `"gaming"`.

```
db.products.find({
  $nor: [
    { category: "Smart Home" },
    { tags: "gaming" }
  ]
});
```

Result: `[`

```
  { _id: 1, name: "Wireless Mouse", category: "Electronics", tags:
["wireless"], ... },
  { _id: 2, name: "Dumbbells", category: "Fitness", tags: [], ... }
]
```

Example 2:
Find products that have **neither** price greater than 90 **nor** stock less than 50.

```
db.products.find({
  $nor: [
    { price: { $gt: 90 } },
```

```
    { stock: { $lt: 50 } }
  ]
});
```

Result: [

```
  { _id: 1, name: "Wireless Mouse", price: 60, stock: 200, ... },
  { _id: 7, name: "Wireless Charger", price: 80, stock: 120, ... }
]
```