✅ ONE CORE RULE (VERY IMPORTANT)

MongoDB always works with DOCUMENTS

- **One document** → { }
- **Multiple documents** → [{ }, { }]

This idea appears in **insert, update, filters, find, aggregation**.

🔹 INSERT

insert ONE document

```
db.books.insertOne({ title: "Java", price: 500 });
```

insert MANY documents

```
db.books.insertMany([
  { title: "Java", price: 500 },
  { title: "Python", price: 450 }
]);
```

🔹 FIND (READ)

find with ONE filter object

```
db.books.find({ price: 500 }).toArray();
```

find with MULTIPLE conditions (still ONE object)

```
db.books.find({
  price: { $gt: 300 },
  writer: "TG"
}).toArray();
```

⚠️ **find never takes [{}, {}] directly**

🔹 UPDATE

update ONE document

```
db.books.updateOne(
  { title: "Java" },       // filter (one object)
  { $set: { price: 600 } } // update object
);
```

update MANY documents

```
db.books.updateMany(
  { writer: "TG" },
  { $set: { price: 400 } }
);
```

✓ Filter = {}
✓ Update = { $set: {} }

DELETE

delete ONE

```
db.books.deleteOne({ title: "Java" });
```

delete MANY

```
db.books.deleteMany({ writer: "TG" });
```

FILTER OPERATORS (inside {})

```
db.books.find({
  price: { $gt: 300, $lt: 600 }
}).toArray();
```

Operators always go **inside the object**.

AGGREGATION (SPECIAL CASE)

Aggregation **REQUIRES ARRAY**:

```
db.books.aggregate([
  { $match: { price: { $gt: 300 } } },
  { $project: { title: 1, price: 1 } }
]);
```

👉 Each {} = one stage
👉 [ ] = pipeline

🧠 QUICK SUMMARY TABLE (SAVE THIS)

Operation	**{}**	**[{}]**
Insert	one doc	many docs
Find	filter	❌
Update	filter & update	❌
Delete	filter	❌
Aggregation	❌	✓ REQUIRED

🟢 GOLDEN LINE (INTERVIEW READY)

MongoDB = document based
Array is used only when multiple documents or stages are involved