Diễn giải: demodb2.mongodb

16 Tháng Mười 2025 1:42 SA

🧱 1️⃣ Khởi tạo Database & Collection

```
use("demodb2")
db.createCollection("products")
```

📘 Giải thích

- `use("demodb2")`:

  → chọn hoặc tạo database tên **demodb2**.

  Nếu chưa tồn tại, MongoDB sẽ **tạo mới** khi bạn thêm dữ liệu đầu tiên.

- `db.createCollection("products")`:

  → tạo một **collection** (bảng dữ liệu) tên là `products`.

  Tương đương với CREATE TABLE trong SQL.

🧩 2️⃣ Thêm dữ liệu

```
db.products.insertMany(
    [
        { name: "Laptop", price: 1500,brand: "Dell", stock: 30,
            tags: ["electronics", "computer"],discount: 10 },
        { name: "Phone", price: 800, brand: "Apple", stock: 50,
            tags: ["electronics", "mobile"], discount: 5 },
        { name: "TV", price: 1200, brand: "Sony", stock: 15,
            tags: ["electronics", "home"], discount: 15 },
        { name: "Shoes", price: 120, brand: "Nike", stock: 80,
            tags: ["fashion", "sport"], discount: 20 },
        { name: "Watch", price: 300, brand: "Casio", stock: 0,
            tags: ["fashion", "accessory","sport"], discount: 0 }
    ]
)
```

📘 Giải thích

- `insertMany([...])`: thêm nhiều document cùng lúc.
- Mỗi `{ ... }` là một document (tương tự một record trong SQL).
- Các field như `name`, `price`, `brand`, `stock`, `tags`, `discount` là các **key-value pair**.

📗 Kết quả

Tạo 5 sản phẩm trong collection `products`.

🧭 3️⃣ Truy vấn cơ bản (Find)

```
db.products.find()
```

- `find()` → lấy tất cả document trong collection.
- Kết quả hiển thị dưới dạng JSON (mỗi dòng một document).

🔍 4️⃣ Query Operators (toán tử truy vấn)

4.1 Lấy sản phẩm có `price > 1000`

```
db.products.find({ price: { $gt: 1000 } })
```

- `$gt` = *greater than* → "lớn hơn".
- Cú pháp:

  ```
  { field: { $operator: value } }
  ```

- Ý nghĩa: lọc các document có giá trị `price` lớn hơn 1000.

👉 Kết quả: Laptop (1500), TV (1200)

4.2 Lấy sản phẩm có `price` trong khoảng 500–1500

```
db.products.find({ price: { $gte: 500, $lte: 1500 } })
```

- `$gte`: lớn hơn hoặc bằng
- `$lte`: nhỏ hơn hoặc bằng
- Cả hai điều kiện gộp chung trên cùng một field.

  → MongoDB hiểu là **AND** giữa `$gte` và `$lte`.

  👉 Kết quả: Laptop (1500), Phone (800), TV (1200)

4.3 Lấy sản phẩm có `discount = 0`

```
db.products.find({ discount: { $eq: 0 } })
```

- `$eq`: *equal to* → bằng

  → Tương tự `WHERE discount = 0` trong SQL.

  👉 Kết quả: Watch (vì discount = 0)

4.4 Lấy sản phẩm có brand là “Apple” hoặc “Dell”

```
db.products.find({ brand: { $in: ["Apple", "Dell"] } })
```

- `$in`: giá trị nằm trong danh sách.

  → tương tự `WHERE brand IN ('Apple','Dell')`

  👉 Kết quả: Phone (Apple), Laptop (Dell)

4.5 Lấy sản phẩm có discount trong danh sách 0, 5, 10

```
db.products.find({ discount: { $in: [0, 5, 10] } })
```

→ Chọn tất cả sản phẩm có `discount` bằng 0, 5, hoặc 10.

👉 Kết quả: Laptop (10), Phone (5), Watch (0)

4.6 Lấy sản phẩm có tag là “mobile” hoặc “fashion”

```
db.products.find({ tags: { $in: ["mobile", "fashion"] } })
```

- `$in` cũng hoạt động được với **mảng**.

  → Nếu mảng `tags` có chứa bất kỳ phần tử nào trong danh sách → thỏa điều kiện.

  👉 Kết quả: Phone (mobile), Shoes (fashion), Watch (fashion)

⚙️ 5️⃣ Logic Operators (Toán tử logic)

5.1 `$or`: sản phẩm có `price < 500` **hoặc** có tag là “fashion”

```
db.products.find({
  $or: [
    { price: { $lt: 500 } },
    { tags: "fashion" }
```

```
  ]
})
```

- `$or`: mảng chứa nhiều điều kiện → chỉ cần **1 điều kiện đúng**.
- `$lt`: nhỏ hơn.

👉 Kết quả: Shoes (vì có tag “fashion”), Watch (vì giá 300 < 500).

5.2 `$not`: phủ định một điều kiện (NOT)

```
db.products.find({
  tags: { $not: { $eq: "electronics" } }
})
```

- `$not` chỉ áp dụng cho **1 field cụ thể**.

→ Chọn những document mà `tags` **không chứa** giá trị `"electronics"`.

👉 Kết quả: Shoes, Watch.

5.3 `$nor`: phủ định nhiều điều kiện (NOT OR)

```
db.products.find({
  $nor: [{ tags: "electronics" }]
})
```

- `$nor`: trả về document **không thỏa bất kỳ điều kiện nào trong mảng**.

→ Nghĩa tương đương “WHERE NOT (tags = 'electronics')”.

👉 Kết quả: Shoes, Watch.

5.4 `$and`: kết hợp nhiều điều kiện đồng thời

```
db.products.find({
  $and: [{ stock: { $gt: 0 } }, { discount: { $gt: 0 } }]
})
```

- `$and`: tất cả điều kiện đều phải đúng.

→ Tìm sản phẩm có `stock > 0` **và** `discount > 0`.

👉 Kết quả: Laptop, Phone, TV, Shoes.

🔢 6️⃣ Element Operators (Kiểm tra field)

6.1 `$exists`: kiểm tra field có tồn tại không

```
db.products.find({ discount: { $exists: true } })
```

→ Trả về mọi document **có trường “discount”**.

6.2 `$type`: kiểm tra kiểu dữ liệu của field

```
db.products.find({ price: { $type: "number" } })
```

→ Lấy tất cả document có `price` kiểu `Number`.

🧺 7️⃣ Array Operators (mảng)

7.1 `$all`: chứa tất cả giá trị cụ thể

```
db.products.find({ tags: { $all: ["fashion", "sport"] } })
```

→ Lấy sản phẩm có **cả hai tag** "fashion" và "sport".

👉 Kết quả: Shoes.

7.2 `$elemMatch`: khớp phần tử trong mảng

```
db.products.find({ tags: { $elemMatch: { $eq: "sport" } } })
```

→ Tìm sản phẩm có phần tử `sport` trong mảng `tags`.

👉 Kết quả: Shoes, Watch.

🧮 8️⃣ Projection Operator (Giới hạn cột)

```
db.products.find({}, { name: 1, price: 1, _id: 0 })
```

- Đối số 1️⃣ : `{}` → không có điều kiện lọc (lấy tất cả).
- Đối số 2️⃣ : `{ name: 1, price: 1, _id: 0 }`
  → Chỉ hiển thị `name` và `price`, ẩn `_id`.
  (`1` = hiện, `0` = ẩn)

🖍️ 9️⃣ Update Operators

9.1 `$inc`: tăng/giảm giá trị số

```
db.products.updateOne({ brand: "Dell" }, { $inc: { stock: 50 } })
```

→ Tăng `stock` thêm 50 cho sản phẩm có brand "Dell".

(Nếu stock ban đầu là 30 → mới = 80)

9.2 `$push`: thêm phần tử vào mảng

```
db.products.updateOne({ name: "Watch" }, { $push: { tags: "new" } })
```

→ Thêm `"new"` vào cuối mảng `tags` của Watch.

Trước: ["fashion", "accessory", "sport"]

Sau: ["fashion", "accessory", "sport", "new"]

9.3 `$unset`: xóa field

```
db.products.updateOne({ name: "Watch" }, { $unset: { stock: "" } })
```

→ Xóa field `stock` khỏi document.

9.4 `$rename`: đổi tên field

```
db.products.updateOne({ name: "Watch" }, { $rename: { "brand": "thuonghieu" } })
```

→ Đổi tên field `brand` thành `thuonghieu`.

9.5 $pop: xóa phần tử đầu/cuối của mảng

```
db.products.updateOne({ name: "Watch" }, { $pop: { tags: 1 } })
```

- `$pop: { field: 1 }` → xóa phần tử **cuối cùng**
- `$pop: { field: -1 }` → xóa phần tử **đầu tiên**

9.6 $pull: xóa phần tử cụ thể trong mảng

```
db.products.updateOne({ name: "Watch" }, { $pull: { tags: "accessory" } })
```

→ Xóa mọi phần tử `"accessory"` trong `tags`.

🧩 Tổng kết cú pháp quan trọng

Nhóm	Toán tử	Ý nghĩa
So sánh	`$gt`, `$gte`, `$lt`, `$lte`, `$eq`, `$ne`	Lớn hơn, nhỏ hơn, bằng, khác
Logic	`$and`, `$or`, `$nor`, `$not`	Kết hợp điều kiện
Mảng	`$in`, `$nin`, `$all`, `$elemMatch`, `$push`, `$pull`, `$pop`	Làm việc với mảng
Field	`$exists`, `$type`, `$unset`, `$rename`	Kiểm tra, đổi tên, xóa trường
Cập nhật	`$inc`, `$set`, `$push`, `$pull`	Thay đổi dữ liệu