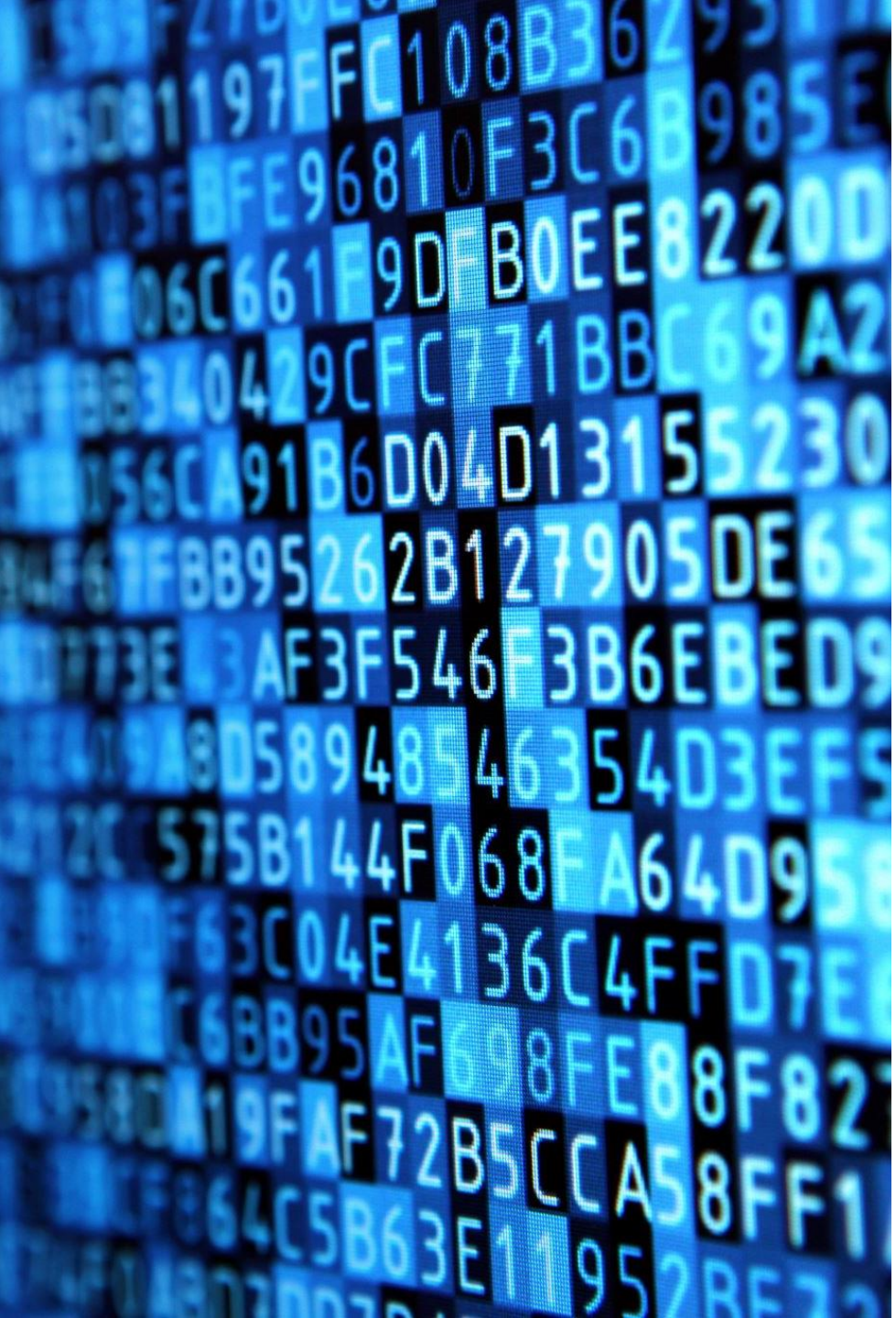

5602201

8. MongoDB Practice: Part II and Part III

CHUTIPORN ANUTARIYA

MongoDB Practice:

Part II

- Searching Products
  - Basic Query

Find Data

There are 2 methods to find and select data from a MongoDB collection, find() and findOne().

find()

- To select data from a collection in MongoDB, we can use the find() method.
- This method accepts a query object. If left empty, all documents will be returned.

findOne()

- To select only one document, we can use the findOne() method.
- This method accepts a query object. If left empty, it will return the first document it finds.

Practice 06: find() and findOne()

```
db.product.find()

db.product.findOne()
```

Basic Query

db.collection.find()

`db.collection.find(query, projection)`

- **query**: document
  - Optional. Specifies selection filter using query operators. To return all documents in a collection, omit this parameter or pass an empty document ({}).
- **projection**: document
  - Optional. Specifies the fields to return in the documents that match the query filter. To return all fields in the matching documents, omit this parameter.

https://docs.mongodb.com/manual/reference/method/db.collection.find/#db.collection.find

Practice 07: Projection

```
db.product.find({}, {id:1, price:1, _id:0})
```

SHOW

NOT SHOW

Results

```
> db.product.find({}, {name:1, price:1, _id:0})
< {
    name: 'Leather Sofa Set 6 Seater',
    price: 615.82
  }
  {
    name: 'Dining Table 4 Seater',
    price: 921.23
  }
  {
    name: 'Rocking Chair',
    price: 889.7
  }
  {
    name: 'Rocking Chair',
    price: 735.67
  }
  {
    name: 'Office Wooden Chair',
    price: 966.37
  }
  {
    name: 'Office Chair with Wheels',
    price: 698.06
  }
```

Practice 08: Query Conditions

```
db.product.find({name:"Table Lamp"}, {name:1, price:1, _id:0})
```

```
{
  name: 'Table Lamp',
  price: 537.23
}
{
  name: 'Table Lamp',
  price: 524.11
}
{
  name: 'Table Lamp',
  price: 815.79
}
{
  name: 'Table Lamp',
  price: 598.39
}
{
  name: 'Table Lamp',
  price: 904.19
}
{
  name: 'Table Lamp',
  price: 837.8
}
```

Query Operators

Comparison

The following operators can be used in queries to compare values:

- `$eq` : Values are equal
- `$ne` : Values are not equal
- `$gt` : Value is greater than another value
- `$gte` : Value is greater than or equal to another value
- `$lt` : Value is less than another value
- `$lte` : Value is less than or equal to another value
- `$in` : Value is matched within an array

Logical

The following operators can logically compare multiple queries.

- `$and` : Returns documents where both queries match
- `$or` : Returns documents where either query matches
- `$nor` : Returns documents where both queries fail to match
- `$not` : Returns documents where the query does not match

Practice 09: Query Operators

```
db.product.find({name: {$ne : "Vase"}})
```

```
db.product.find({name:"Table Lamp"}, {name:1, price:1, _id:0})
```

```
db.product.find({price: {$gt : 500, $lt : 550}}, {name:1, price:1, _id:0})
```

Practice 10: $and

```
db.product.find(
    { $and: [
        {price: { $gt: 500, $lt: 550 }},
        {name: "Table Lamp" }
    ] },
    { name: 1, price: 1, _id: 0 })
```

```
{
  name: 'Table Lamp',
  price: 537.23
}
{
  name: 'Table Lamp',
  price: 524.11
}
```

Practice 11: $or

```
db.product.find(
    {
        $or: [
            { "description.countrOfOrigin": "Japan" },
            { "description.countrOfOrigin": "Thailand" }
        ]
    })
```

Search product from Japan or Thailand

```
> db.product.find({$or: [{"description.countrOfOrigin": "Japan"}, {"description.countrOfOrigin":"Thailand"}]})
< { _id: ObjectId("632d29d89c05f50cc5b24164"),
    id: 1,
    name: 'Leather Sofa Set 6 Seater',
    status: 'Available',
    description:
     [ { countrOfOrigin: 'Japan',
         material: 'Oak Wood',
         dimension: [ { height: 11, width: 4, weight: 12049 } ] } ],
    price: 615.82 }
  { _id: ObjectId("632d29d89c05f50cc5b24167"),
    id: 4,
    name: 'Rocking Chair',
    status: 'Available',
    description:
     [ { countrOfOrigin: 'Thailand',
         material: 'Synthetic',
         dimension: [ { height: 7, width: 8, weight: 7184 } ] } ],
    price: 735.67 }
```

Practice 12: Sort()

```
db.product.find(
    {},
    {name:1, price:1, _id:0}
).sort({price: -1})
```

1 = ASC

-1 = DESC


```
{
  name: 'Office Wooden Chair',
  price: 966.37
}
{
  name: 'Vase',
  price: 965.38
}
{
  name: 'Leather Sofa Set 6 Seater',
  price: 956.71
}
{
  name: 'Office Wooden Chair',
  price: 946.91
}
{
  name: 'Leather Sofa Set 6 Seater',
  price: 937.99
}
{
  name: 'Leather Sofa Set 6 Seater',
  price: 932.61
}
```

MongoDB Practice:

Part III

- Searching Products
  - Advanced Query
  - Aggregate Query

Practice 13: Nested Field

Uses dot notation to access fields in an embedded document:

```
db.product.find({"description.dimension.height":11})
```

```
{
  _id: ObjectId("654766d1183ca4bb40f213f2"),
  id: 1,
  name: 'Leather Sofa Set 6 Seater',
  status: 'Available',
  description: [
    {
      countrOfOrigin: 'Japan',
      material: 'Oak Wood',
      dimension: [
        {
          height: 11,
          width: 4,
          weight: 12049
        }
      ]
    }
  ],
  price: 615.82,
  isBestSeller: true
}
```

Practice 14: Pattern Matching

Pattern Matching using Regular Expression ($regex):

```
db.product.find
    (
        { name: { $regex: /^wood/i } },
        { name: 1, _id: 0 }
    );
```

Start with the word “wood”, match with any cases.

```
{
  name: 'Wooden Chair'
}
{
  name: 'Wooden Chair'
}
{
  name: 'Wooden Chair'
}
{
  name: 'Wooden Chair'
}
{
  name: 'woodplate'
}
```

Practice 15: Aggregate Query

Group products by their status, count them and sum their prices

```
db.product.aggregate([
    {
    $group: {
        _id: "$status",
                    count: {
                  $sum: 1
                },
            totalValue: {
              $sum: '$price'
            }
    }
    }
]);
```

```
{
  _id: null,
  count: 3,
  totalValue: 529
}
{
  _id: 'Unavailable',
  count: 18,
  totalValue: 13127.64
}
{
  _id: 'Available',
  count: 32,
  totalValue: 24130.6
}
```

Bonus: Your Turn

1. Select products where their height less than 13 cm and the material is “Oak Wood”.
2. Group products by their status, count them, show the average, min, max and sum of their prices
3. What does the following query return:

```
db.product.find
    (
        { name: { $regex: /table/i } },
        { name: 1, _id: 0 }
    );
```


THANK YOU