**Date: 22-07-2025**

**Update, Delete, Projection, Embedded documents in MongoDB**

**Update Operations:**

1. updateOne()

db.people.updateOne(

{ name: "Matthew" }, // Filter condition

{ $set: { status: "Completed" } } // Update operation

)

*Updates the first document that matches the filter. In this case, it changes Matthew’s status to "Completed".*

2. updateMany()

db.people.updateMany(

{ status: "Pending" }, // Filter condition

{ $set: { status: "In Progress" } } // Update operation

)

*Updates all documents where status is "Pending" to "In Progress".*

3. replaceOne()

db.people.replaceOne(

{ name: "Griffin" }, // Filter condition

{ name: "Griffin", age: 2 } // Entire replacement document

)

*Replaces the entire document for Griffin with the new one — old fields like status will be lost unless included.*

**Delete Operations:**

1. deleteOne():

db.people.deleteOne({ name: "Matthew" })

*removes only the first match.*

1. deleteMany():

db.people.deleteMany({ status: "Pending" })

*for bulk deletion.*

1. Delete all Documents:

db.people.deleteMany({})

1. Drop whole Collection:

db.people.drop()

**Projection in MongoDB:**

Projection in MongoDB is used to control which fields are returned in the results of a query.

**Basic Syntax:**

db.collection.find(query, projection)

* query: filter conditions (like status: "Pending")
* projection: specifies which fields to include or exclude

**Inclusion Example:**

db.people.find({}, { name: 1, age: 1 })

*Returns only the name, age, and default \_id fields from all documents.*

**Exclusion Example:**

db.people.find({}, { age: 0 })

*Returns all fields except age.*

* Also, remember, You **cannot mix inclusion and exclusion** in the same projection **(except \_id)**.

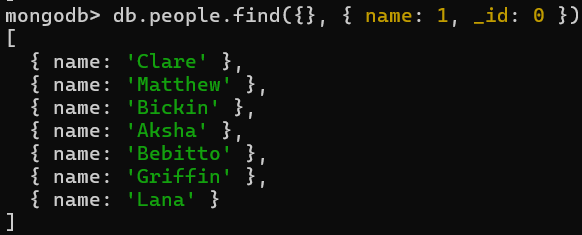
db.people.find({}, {name:1, \_id:0, age:0}) – It throws error.

* \_id is the **only field** that can be excluded alongside inclusions.
* Use 1 to **include** fields, 0 to **exclude.**

**Example Use cases:**

1. Return Only Names:

db.people.find({}, { name: 1, \_id: 0 })



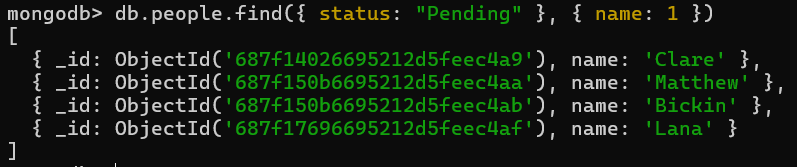
1. Exclude Age:

db.people.find({}, { age: 0 })



1. Conditional + Projection:

db.people.find({ status: "Pending" }, { name: 1 })



**Embedded Documents in MongoDB:**

In MongoDB, embedded documents (also called nested documents) are documents that are stored as values inside other documents.

Example of an Embedded Document:

{

name: "Clare",

age: 21,

address: {

street: "123 Main St",

city: "Mumbai",

pincode: 400001

}

}

*Here, the field address is an embedded document inside the main document.*

**Why Use Embedded Documents?**

* Keep related data together (like address with a person).
* Faster reads, since everything is in one place.
* Simpler queries for related fields (no joins required).
* Ideal for one-to-few relationships.

**Querying Embedded Documents:**

1. Match Entire Embedded Document:

db.people.find({

address: { street: "123 Main St", city: "Mumbai", pincode: 400001 }

})

*Must match all fields and order exactly.*

1. Query Specific Field in Embedded Document (Dot Notation):

db.people.find({ "address.city": "Mumbai" })

*Matches documents where address.city equals "Mumbai".*

1. Project Specific Nested Field:

db.people.find({}, { "address.city": 1, \_id: 0 })

*Only shows the city field inside the address, excluding \_id.*

1. Query Using Comparison Operator on Embedded Field:

db.people.find({ "address.pincode": { $gt: 400000 } })

*Finds people whose pincode is greater than 400000.*