1. FIELD NAME AUR USKE AAGE EK VALUE .
SLIDE NUMBER : 2

2. FIELD NAME AUR USKE AAGE EK DOCUMENT .
SLIDE NUMBER : 4

3. FIELD NAME AUR USKE AAGE EK ARRAY .
SLIDE NUMBER : 4

4. FIELD NAME AUR USKE AAGE EK ARRAY OF DOCUMENTS.
SLIDE NUMBER: 3

= :													
_													

DISCUSSING EACH TYPE ONE BY ONE

TYPE 1 :: EK FIELD NAME AUR USKE AAGE EK SIMPLE CHEEZ

SLIDE NUMBER 2

1. Create a Database

2. Create a Collection

3. Insert a Document

Create:

Adding new documents to the collection.

Read:

Retrieving documents from collection.

Update:

Updating documents in collection.

Delete:

Removing documents from collection.

1. CREATE OPERATION

Feature	Description
Purpose	Adds/Inserts a new document into a collection.
Method Used	<pre>insertOne(), insertMany().</pre>
Collection Handling	If the collection does not exist, MongoDB automatically creates it when inserting a document.
Bulk Insertion	Supports inserting multiple documents at once.

FIRST OF ALL .. UNDERSTAND ID WALI BAKCHODI ...

1. IF NOT ASSIGNED :: — THIS IS DEFAULT

_id: ObjectId("98232303df34948b4

2. IF ASSIGNED MANUALLY

_id: "user123"

_id: 1001

YAA TOH MANUALLY DE SAKTE HO YAA FIR AAPKI JO MONGO HAI WO KHUD HI ASSIGN KARDEGI ...

CREATE HAS 2 THINGS ::: --

- 1. INSERTONE
- 2. INSERTMANY

1. INSERT ONE SYNTAX ::—

DATABASENAME.COLLECTIONNA ME.insertOne({ name : "harry" , age : 34 })

JEHRA CONTENT HAI USS CH PEHLA TAH SIMPLE ROUND BRACKET ... TE USS TOH BAAD ANDER DOCUMENT LAGAN LAYI CURLY BAS KHATAM KYUNKI EK HI DOCUMENT INSERT KAR SAKTE HEIN

AGAR STRING HAI TOH DOUBLE QUOTES AND AGAR NUMBER HAI TOH NOTHING ...

2. INSERTMANY

CREATE MULTIPLE DOCUMENTS..
DB_NAME.COLL_NAME.
insertMany([{},{},{},{}])
CURLY BRACKETS KE ANDER
DOCUMENTS BAN SAKTE HEIN
(END CH SEMICOLON)

2. UPDATE OPERATIONS

- updateOne()
- updateMany()
- replaceOne()

A) UPDATEONE

db.collection	Specifies the collection in which documents will be updated.
<filter></filter>	A query that selects the documents to update.
<update></update>	The update operation using \$set , \$currentDate , \$inc , etc.

This method updates the first matching document.

```
db.users.updateOne(
    { name: "Bob" },
    {
        $set: { course: "CSE" },
        $currentDate: { lastModified: true }
    }
};
```

B) UPDATEMANY

This method updates all documents that match the filter criteria.

UPDATE OPERATORS

```
1. Using $currentDate (Sets the field to the current date)
Before Update:
                                                                         〇 Copy 🎖 Edit
   name: "Alice",
Update Query:
                                                                         ☐ Copy ಶ Edit
  db.users.updateOne(
   { name: "Alice" },
   { $currentDate: { lastModified: true } }
  );
After Update:
   name: "Alice",
   lastModified: "2025-02-06T15:25:00Z" // Updated to current timestamp
```

3. Using \$min (Updates a field only if the new value is smaller than the existing one)

Before Update:

Update Query:

```
db.users.updateOne(
    { name: "Charlie" },
    { $min: { salary: 55000 } }
);
```

After Update (Salary Updated Since 55000 < 60000):

4. Using \$max (Updates a field only if the new value is greater than the existing one) **Before Update:** ☐ Copy 🄣 Edit name: "David", **Update Query:** db.users.updateOne({ name: "David" }, { \$max: { salary: 80000 } }); After Update (Salary Updated Since 80000 > 70000): ☐ Copy 🄣 Edit { salary: 80000 // Updated because 80000 > 70000 } \downarrow

2. Using \$inc (Increments a field by a specified value) **Before Update:** { **Update Query:** ☐ Copy 🄣 Edit db.users.updateOne({ name: "Bob" }, { \$inc: { age: 5 } }); After Update: ☐ Copy 🄣 Edit { age: 30 // Age increased by 5 }

5. Using \$mul (Multiplies the field value by a specified number) **Before Update:** ☐ Copy 🄣 Edit { _id: 5, name: "Eve", bonus: 2000 } **Update Query:** db.users.updateOne({ name: "Eve" }, { \$mul: { bonus: 2 } }); After Update (Bonus Multiplied by 2): ☐ Copy 🎖 Edit { _id: 5, name: "Eve", } \downarrow



New field vi add kar salsa if not present.

```
6. Using $rename (Renames a field)
Before Update:
  {
Update Query:
  db.users.updateOne(
   { $rename: { department: "team" } }
  );
After Update (Field department Renamed to team ):
  {
                                           \downarrow
```







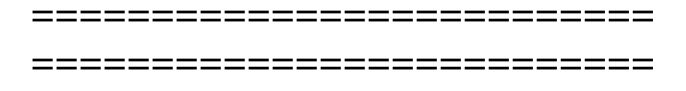
REPLACE ONE

(Id wali bakchodi yaar rakhi hai)

POORA DA POORA DOC HI NAVA BAN JANDA ... PICCHE AALE DA KOI ASTITV NI REHNDA ..

3. DELETE OPERATION

- The deleteOne() method removes a single document from a collection. It has a single required parameter which is a filter criteria to match a specific document to delete.
- db.courses.deleteOne({module: "java"})
- The .deleteMany() method removes all documents that match a given filter criteria. It takes in a single required parameter, the filter criteria to match multiple documents.
- db.courses.deleteMany({module: "java"})



4. FIND OPERATIONS

```
db.Rockers.find({ gender: "F", age: 19 })
```

```
db.Rockers.findOne({ gender: "F" })
```

ONLY FIRST IN DOCUMENT
