**CRUD in NoSQL (MongoDB)**

CRUD stands for **Create, Read, Update, and Delete**, which are the basic operations performed on a database.

1. **Create** – Inserting new documents (records) into a collection (table equivalent in MongoDB).
2. **Read** – Retrieving documents from a collection using queries.
3. **Update** – Modifying existing documents.
4. **Delete** – Removing documents from a collection.

In NoSQL databases like MongoDB, CRUD operations are performed using methods like insertOne(), insertMany(), find(), updateOne(), updateMany(), deleteOne(), and deleteMany().

**How MongoDB Stores Data**

MongoDB is a **document-oriented** NoSQL database that stores data in **JSON-like BSON (Binary JSON) format**. Instead of tables with rows and columns, data is stored in **collections** (equivalent to tables), and each record is stored as a **document** (equivalent to rows).

Example of a MongoDB document:

{

"\_id": ObjectId("507f191e810c19729de860ea"),

"name": "Alice",

"age": 30,

"email": "alice@example.com"

}

* Each document is **schema-flexible**, meaning different documents in the same collection can have different structures.
* **Indexes** are used to optimize query performance.

**Creating and Inserting Documents in MongoDB**

MongoDB provides methods for inserting documents into a collection.

**Insert a Single Document**

db.users.insertOne({

"name": "John Doe",

"email": "john@example.com",

"age": 28

});

**Insert Multiple Documents**

db.users.insertMany([

{ "name": "Alice", "email": "alice@example.com", "age": 25 },

{ "name": "Bob", "email": "bob@example.com", "age": 27 }

]);

* insertOne() inserts one document.
* insertMany() inserts multiple documents.

**Removing Documents from a MongoDB Database**

To remove documents, MongoDB provides the following methods:

**Delete a Single Document**

db.users.deleteOne({ "name": "John Doe" });

* Deletes the **first matching** document.

**Delete Multiple Documents**

db.users.deleteMany({ "age": { "$gt": 25 } });

* Deletes all documents where age is greater than 25.

**Remove All Documents**

db.users.deleteMany({});

* Deletes **all documents** in the collection.

**How Read Operations Retrieve Data**

Read operations in MongoDB allow retrieving data using the find() method. MongoDB **retrieves** data based on conditions and can return one or multiple documents.

* Queries in MongoDB are performed using **key-value pairs**.
* If no criteria are specified, MongoDB returns **all documents** from a collection.

**How Cursors are Used in MongoDB**

A **cursor** in MongoDB is an iterator that allows traversing through a set of documents retrieved by a query.

1. When you use find(), MongoDB **returns a cursor**, which points to the result set.
2. The cursor allows iterating through the results one by one or all at once.

Example:

var cursor = db.users.find();

while (cursor.hasNext()) {

printjson(cursor.next());

}

* hasNext() checks if more documents exist.
* next() retrieves the next document.

MongoDB **automatically iterates** over the cursor in interactive environments like the MongoDB shell.

**Using the find() Method to Query Documents**

The find() method is used to retrieve documents from a MongoDB collection.

**Retrieve All Documents**

db.users.find();

**Retrieve Specific Fields**

db.users.find({}, { "name": 1, "email": 1, "\_id": 0 });

* Returns only name and email fields, excluding \_id.

**Find Documents with a Condition**

db.users.find({ "age": { "$gt": 25 } });

* Finds all users **older than 25**.

**Find a Single Document**

db.users.findOne({ "name": "Alice" });

* Returns the first document where name is "Alice".

**Using Operators in Queries**

* **$gt** (greater than), **$lt** (less than), **$eq** (equal to), **$ne** (not equal)

db.users.find({ "age": { "$gte": 25 } });

* Finds users **age 25 or older**.

**Using Logical Operators**

* **$and**, **$or**, **$not**

db.users.find({ "$or": [{ "age": 30 }, { "name": "Alice" }] });

* Finds users who are either **30 years old** or named **Alice**.

**Sorting and Limiting Results**

db.users.find().sort({ "age": -1 }).limit(5);

* Sorts users by age in **descending order**.
* Returns **only 5 documents**.

**Summary**

* **CRUD** operations allow managing data in MongoDB.
* MongoDB stores data in **BSON (Binary JSON)** format as **documents** in **collections**.
* Documents are **inserted** using insertOne() and insertMany().
* Documents are **removed** using deleteOne() and deleteMany().
* The **find()** method is used to retrieve data, which is returned via a **cursor**.
* **Cursors** allow efficient iteration over query results.
* **Querying** supports advanced conditions using operators like $gt, $lt, $or, and $and.