MongoDB

Add, Delete and Update operations

AND Operator in MongoDB Queries

The AND operator in MongoDB queries allows you to specify multiple criteria that a document must satisfy to be included in the query results. It's used to combine multiple filtering conditions.

Below is the Command:

```
db.collection.find({ condition1: value1, condition2: value2, ... })
```

- db: The name of the MongoDB database.
- collection: The name of the collection you're querying.
- condition1, condition2: These are filter expressions that define the criteria documents must meet.
- value1, value2: The values to compare against the corresponding conditions.

Example:

```
db.products.find({ name: "T-Shirt", color: "Red" })
```

This query will find all documents in the products collection where the name field equals "T-Shirt" AND the color field equals "Red".

Note:

- We can combine multiple AND conditions using additional key-value pairs within the main query object.
- MongoDB also supports logical operators like OR ($or) and NOT ($not) for more complex filtering.

Update Operations in MongoDB

MongoDB offers several methods to update existing documents:

1. **updateOne()**
   Updates a single document that matches the specified filter criteria.
   Command:

```
db.collection.updateOne({ filter }, { update }, options)
```

- filter: A query document defining which document to update.
- update: A document specifying the changes to apply. You can use update operators like $set, $unset, $inc, etc.
- options (optional): Additional options like upsert (create a document if no match is found).

Example (updateOne):

```
db.users.updateOne({ username: "Ananya" }, { $set: { email: "ananya@ait.com" } })
```

This updates the email address for the user with the username "Ananya".

2. **updateMany()**
   Updates multiple documents that match the filter criteria.
   Command:
   ```
   db.collection.updateMany({ filter }, { update }, options)
   ```
   Similar arguments to updateOne().

3. **replaceOne()**
   Replaces a single document that matches the filter with a new document entirely.
   Command:
   ```
   db.collection.replaceOne({ filter }, replacement, options)
   ```
   replacement: The new document to replace the existing one.

Delete Operations in MongoDB

MongoDB provides methods to remove documents from collections:

1. **deleteOne()**
   Removes a single document matching the filter criteria.
   Command:
   ```
   db.collection.deleteOne({ filter })
   ```
   filter: A query document defining which document to delete.
   Example (deleteOne):
   ```
   db.products.deleteOne({ _id: ObjectId("1234567890abcdef") })
   ```
   This deletes the document with the specified _id from the products collection.

2. **deleteMany()**
   Removes multiple documents that match the filter criteria.
   Command:
   ```
   db.collection.deleteMany({ filter })
   ```
   filter: A query document defining which documents to delete.

Note:

- Update and delete operations in MongoDB are atomic, meaning they are completed entirely or not at all, ensuring data consistency.
- We have to use filters carefully to avoid unintended deletions or updates.