**Basic CRUD Operations**

1. **Insert a Single Document**: Insert a new user document into a users collection.
2. **Insert Multiple Documents**: Insert multiple product documents into a products collection.
3. **Find a Document by ID**: Retrieve a user document by its \_id from the users collection.
4. **Update a Document**: Update the email address of a user in the users collection.
5. **Delete a Document**: Remove a product from the products collection by its \_id.
6. **Find Documents with a Condition**: Retrieve all users who are older than 30 from the users collection.
7. **Update Multiple Documents**: Increase the price of all products in the products collection by 10%.
8. **Delete Multiple Documents**: Remove all users who have not logged in for over a six months from the users collection.
9. **Upsert Operation**: Insert a new user if it does not exist, or update the existing user’s information.
10. **Regular Expression**: Find places that have the word “of” in their names using a regular expression.
11. **Compound Index**: Create a compound index on firstName and lastName in the users collection.
12. **Text Index**: Create a text index on the description field in the products collection for full-text search.
13. **Unique Index**: Ensure that the username field in the users collection is unique.
14. **Find text**: Using the text index, find the products with “laptop” word in their description.
15. **Geospatial Index**: Create a 2dsphere index on the location field in the places collection.
16. **Indexed search**: Retrieve all products with prices higher than 1000 and with the word “laptop” in the description.
17. **Find Documents Near a Point**: Retrieve all places within 5 kilometers of a given point.
18. **Geospatial Sorting**: Sort places by their distance from a given point.
19. **Find documents using $or** : Find places with the word “the” or “of” in their names.
20. **Find documents using $in** : Find products of Electronics and Furniture type.
21. **Find documents by array value**: Find places with both coordinates less than 0.

**Additional:**

**Execution Time:**

In order to see the time the system spend executiong your query, you can use the command “explain”.

Example:

db.coll.find().explain(“executionStats”)[“executionStats”]

You will obtain a JSON result, and in the filed “executionTimeMillis” you will find the time in milliseconds that the system spent on your query execution.

**Query Regular Expressions**: db.col.find( {description : {$regex : /the/}})

**Query Texts** (on a text index): db.col.find( {$text : {$search: “shop”}})

You do not set the column, because it is set by the index creation. There will be only one text index. However you can create a text index with more than one column.

Example: db.col.createIndex({name : “text”, description : “text})

**Query with specific position of an array**: You can access to the specific position of an array by putting its position after a “.”, the position starts with 0.

Example: db.col.find({“location.coordinates.1” : {$lt : 40}})

**Count the number of outputs**: You can count the number of outputs by adding the function “.count()” at the end:

Example: db.col.find({“location.coordinates.1” : {$lt : 40}}).count()

**Sort the outputs**: You can sort the outputs by adding the function “.sort()” at the end and the attribute by which are you sorting between the round brackets:

Example: db.col.find({“location.coordinates.1” : {$lt : 40}}).count(“location.coordinates.1”)