विद्याधनं सर्वधन प्रधानम्।

MUMBAI EDUCATIONAL TRUST

MET Institute of Computer Science

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET
Bhujbal Knowledge City

Program No:	**2**
Roll No :	**1545**
Title of Program :	**MongoDB**
Objective :	**Performing CRUD operation and aggregate operation using MongoDB**

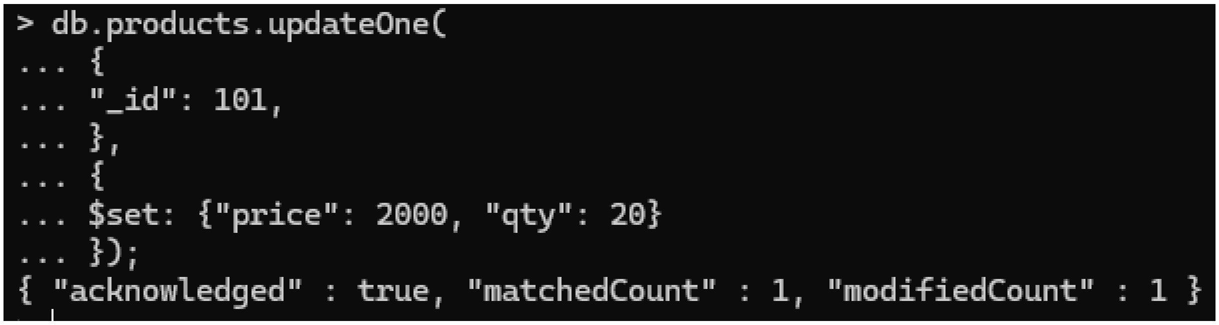
Source Code:

CRUD in mongoDB

1. Insert data into collection

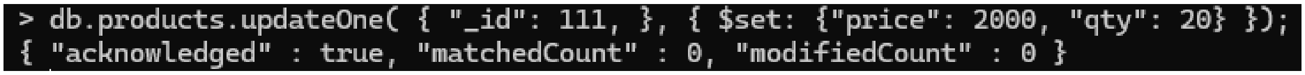
```
> db.products.insert( { "_id": 101, "name": "Paper" });
> db.products.insert( { "_id": 102, "name": "Paper", "brand": "Dunder Mifflin", "price":
1000, "rating": 4.5, "qty": 100 });
WriteResult({ "nInserted" : 1 })
> db.products.insert( { "_id": 103, "name": "Printer", "brand": "Dunder Mifflin", "price":
5000, "rating": 3.5, "qty": 10 });
> db.products.insert( { "_id": 104, "name": "Printer", "brand": "Dunder Mifflin", "price":
2000, "rating": 2.5, "qty": 20 });
> db.products.insert( { "_id": 105, "name": "Staple", "brand": "Dunder Mifflin", "price":
500, "rating": 3.0, "qty": 50 });
> db.products.insert( { "_id": 106, "name": "T-shirt", "brand": "USPA", "price": 1000,
"rating": 4.0, "qty": 10 });
> db.products.insert( { "_id": 107, "name": "T-shirt", "brand": "Peter England", "price":
1200, "rating": 4.5, "qty": 100 });
> db.products.insert( { "_id": 108, "name": "Phone", "brand": "One Plus", "price":
20000, "rating": 4.5, "qty": 500 });
> db.products.insert( { "_id": 109, "name": "Phone", "brand": "Samsung", "price":
80000, "rating": 5.0, "qty": 100 });
> db.products.insert( { "_id": 110, "name": "Phone", "brand": "Iphone", "price": 90000,
"rating": 2.0, "qty": 10 });
```

2. UpdateOne



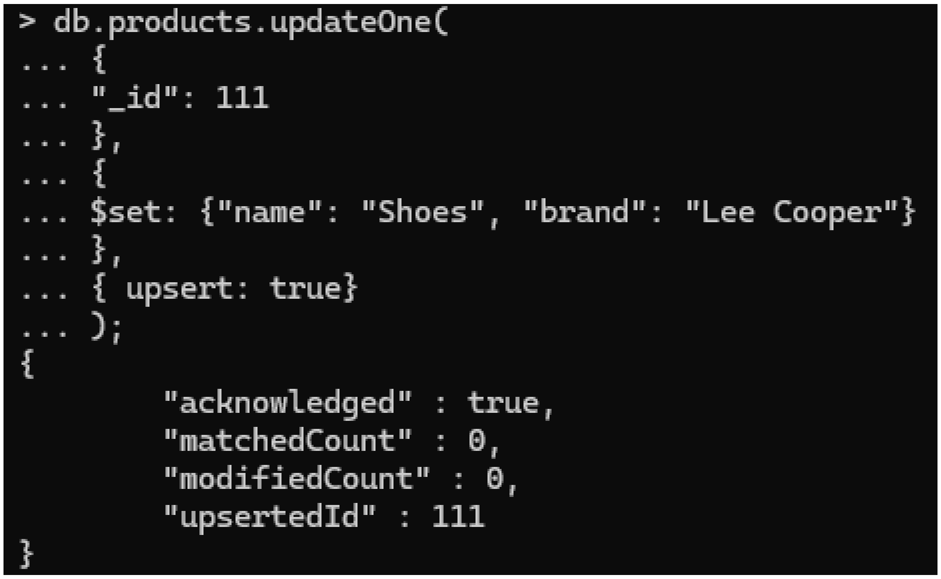
```
> db.products.updateOne(
... {
... "_id": 101,
... },
... {
... $set: {"price": 2000, "qty": 20}
... });
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

3. Match not found



```
> db.products.updateOne( { "_id": 111, }, { $set: {"price": 2000, "qty": 20} });
{ "acknowledged" : true, "matchedCount" : 0, "modifiedCount" : 0 }
```

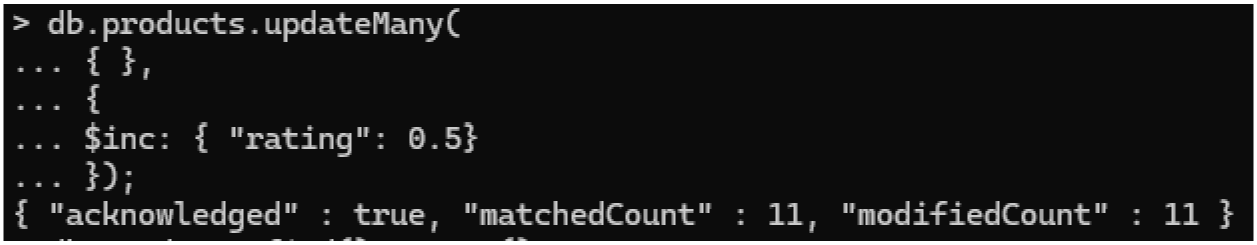
4. Upsert (if document exist update else insert)



```
> db.products.updateOne(
... {
... "_id": 111
... },
... {
... $set: {"name": "Shoes", "brand": "Lee Cooper"}
... },
... { upsert: true}
... );
{
        "acknowledged" : true,
        "matchedCount" : 0,
        "modifiedCount" : 0,
        "upsertedId" : 111
}
```

5. UpdateMany

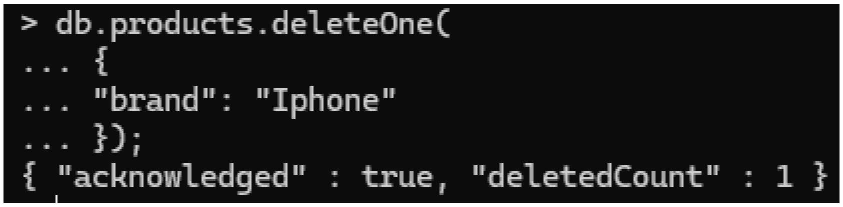
write a query to update the rating by .5 of all documents



```
> db.products.updateMany(
... { },
... {
... $inc: { "rating": 0.5}
... });
{ "acknowledged" : true, "matchedCount" : 11, "modifiedCount" : 11 }
```

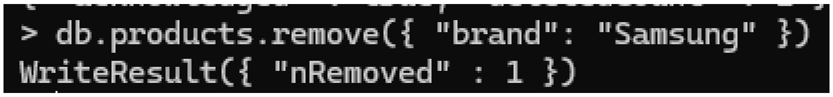
6. Delete document

deleteOne



```
> db.products.deleteOne(
... {
... "brand": "Iphone"
... });
{ "acknowledged" : true, "deletedCount" : 1 }
```

remove
if no criteria specified all documents are deleted

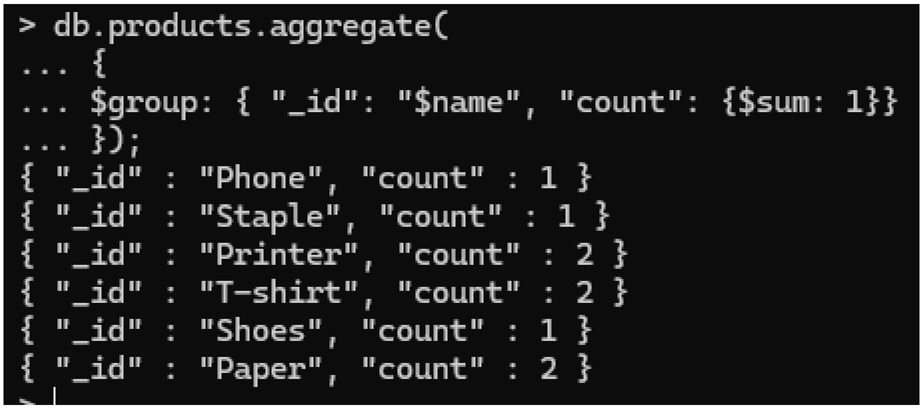


```
> db.products.remove({ "brand": "Samsung" })
WriteResult({ "nRemoved" : 1 })
```

Aggregation in MongoDB

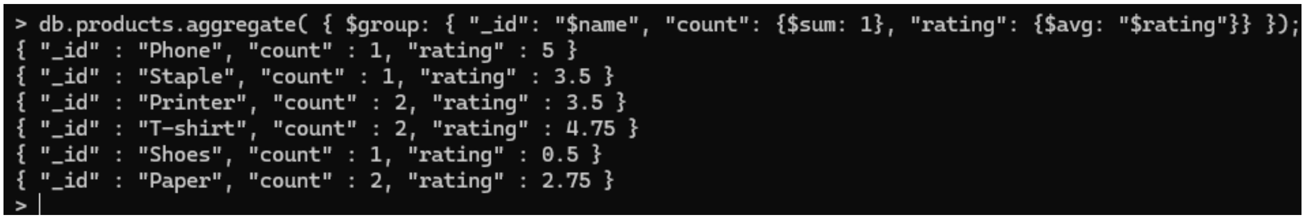
Aggregation are used to get summaries of data

1. Write a query to find the count of all the products according to their name



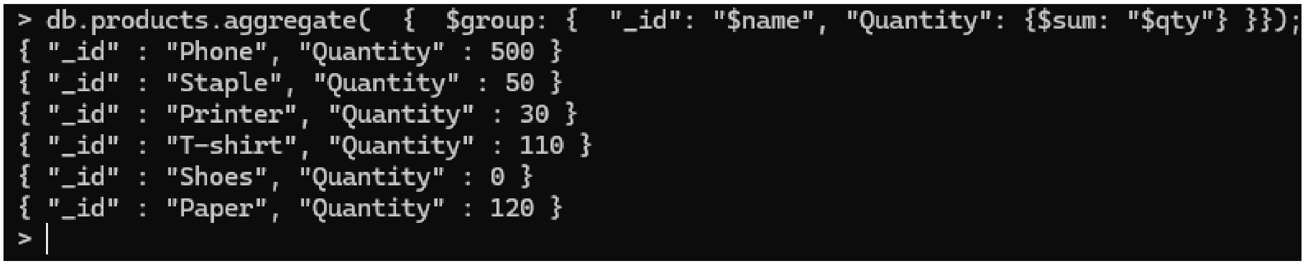
```
> db.products.aggregate(
... {
... $group: { "_id": "$name", "count": {$sum: 1}}
... });
{ "_id" : "Phone", "count" : 1 }
{ "_id" : "Staple", "count" : 1 }
{ "_id" : "Printer", "count" : 2 }
{ "_id" : "T-shirt", "count" : 2 }
{ "_id" : "Shoes", "count" : 1 }
{ "_id" : "Paper", "count" : 2 }
>
```

2. Write a query to group on the basis of name and count along with average rating of product



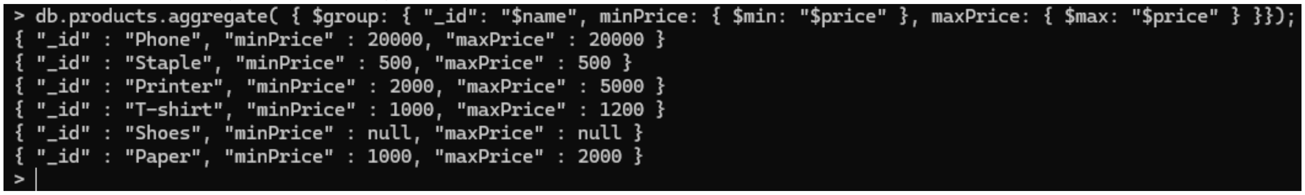
```
> db.products.aggregate( { $group: { "_id": "$name", "count": {$sum: 1}, "rating": {$avg: "$rating"}} });
{ "_id" : "Phone", "count" : 1, "rating" : 5 }
{ "_id" : "Staple", "count" : 1, "rating" : 3.5 }
{ "_id" : "Printer", "count" : 2, "rating" : 3.5 }
{ "_id" : "T-shirt", "count" : 2, "rating" : 4.75 }
{ "_id" : "Shoes", "count" : 1, "rating" : 0.5 }
{ "_id" : "Paper", "count" : 2, "rating" : 2.75 }
>
```

3. Write a query to display name of product and total quantity of that product



```
> db.products.aggregate(  {  $group: {  "_id": "$name", "Quantity": {$sum: "$qty"} }});
{ "_id" : "Phone", "Quantity" : 500 }
{ "_id" : "Staple", "Quantity" : 50 }
{ "_id" : "Printer", "Quantity" : 30 }
{ "_id" : "T-shirt", "Quantity" : 110 }
{ "_id" : "Shoes", "Quantity" : 0 }
{ "_id" : "Paper", "Quantity" : 120 }
>
```

4. Write a query to display min and max price of product



```
> db.products.aggregate( { $group: { "_id": "$name", minPrice: { $min: "$price" }, maxPrice: { $max: "$price" } }});
{ "_id" : "Phone", "minPrice" : 20000, "maxPrice" : 20000 }
{ "_id" : "Staple", "minPrice" : 500, "maxPrice" : 500 }
{ "_id" : "Printer", "minPrice" : 2000, "maxPrice" : 5000 }
{ "_id" : "T-shirt", "minPrice" : 1000, "maxPrice" : 1200 }
{ "_id" : "Shoes", "minPrice" : null, "maxPrice" : null }
{ "_id" : "Paper", "minPrice" : 1000, "maxPrice" : 2000 }
>
```