DBT Assignment-4

ASHWIN KRISHNA P

PES1201801465

SEM: 5 , SECTION: F

MongoDB(Document Database)

show dbs command shows all available databases.

Command **use DBT_A4** to create new database called DBT_A4.

Command **show collections** to see all collections.

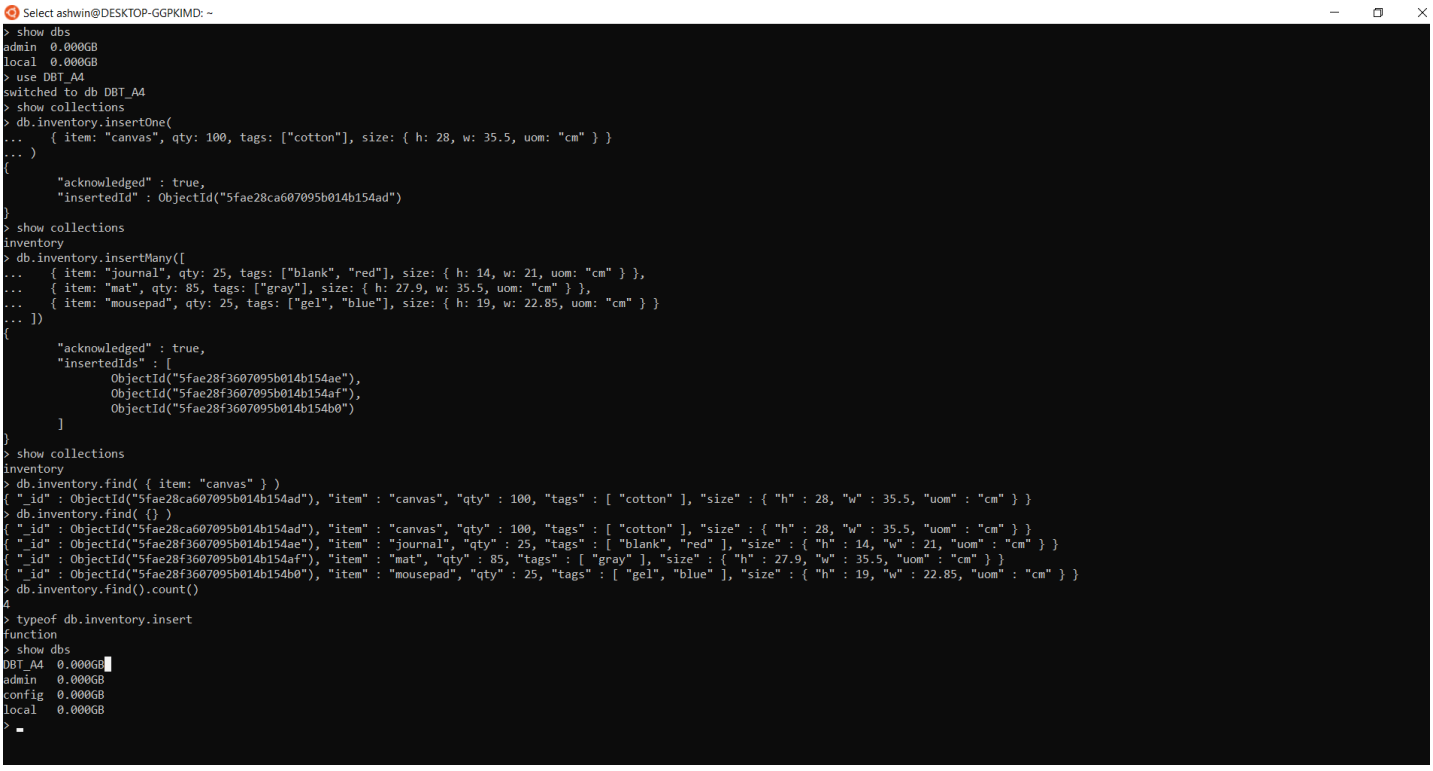
INSERT:

If the collection does not currently exist, insert operations will create the collection.

db.collection.insertOne() : Inserts single document into a collection

db.collection.insertMany() : Inserts multiple documents into the collection

Here I have created and inserted documents to the collection named **inventory**.



db.inventory.find({ item: "canvas" }) : To retrieve the document where item=”canvas”

db.inventory.find({}) : To retrieve all the documents in a collection. It is similar to **select * from inventory** in SQL

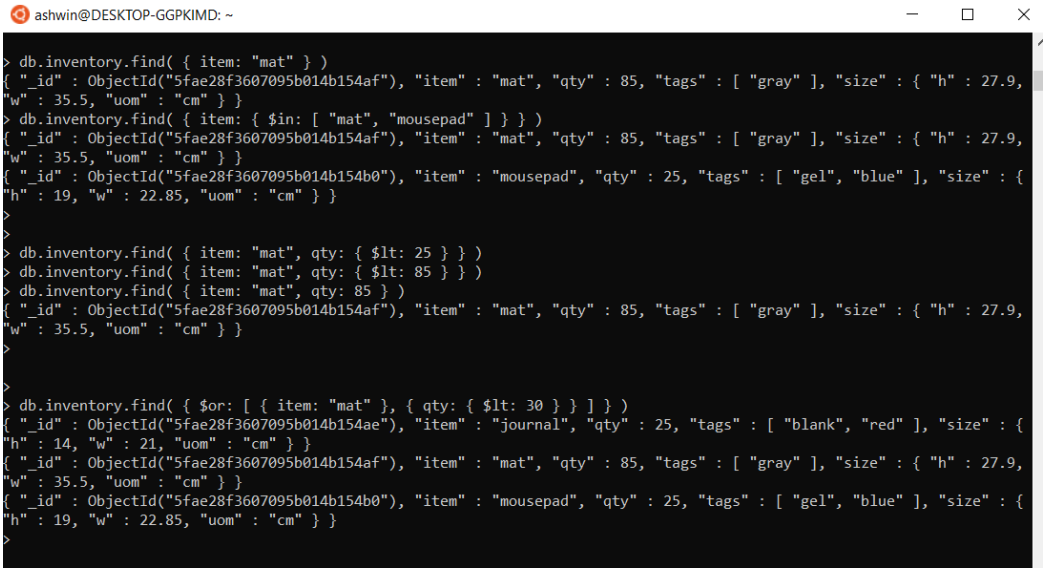
Queries:

db.inventory.find({ item: "mat" }) : To retrieve the document where item=”mat”

db.inventory.find({ item: { $in: ["mat", "mousepad"] } }) : To retrieve the document from collection if item is in “mat” or “mousepad”. Similar to **SELECT * FROM inventory WHERE item in ("mat", "mousepad")**

db.inventory.find({ item: "mat", qty: { $lt: 25 } }) : Similar to **SELECT * FROM inventory WHERE item= "mat" AND qty < 25**

db.inventory.find({ $or: [{ item: "mat" }, { qty: { $lt: 30 } }] }) : Similar to **SELECT * FROM inventory WHERE item = "mat" OR qty < 30**

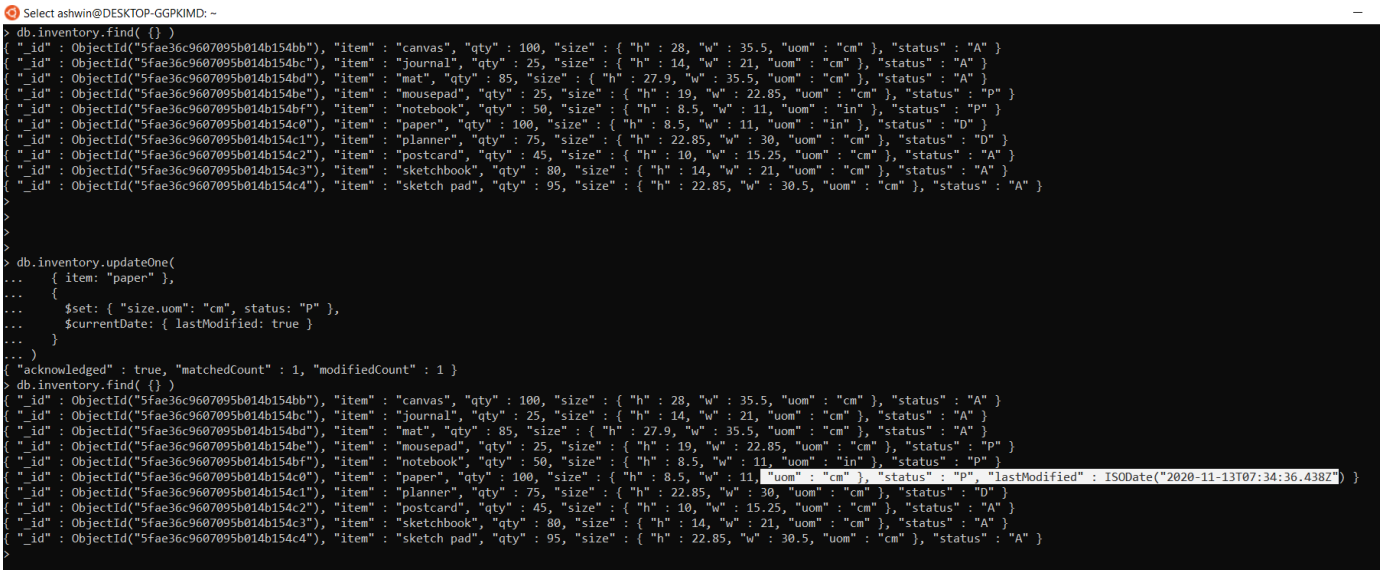


UPDATE:

db.collection.updateOne() : Updates a single document.

I deleted all documents and inserted some new documents into the collection inventory.

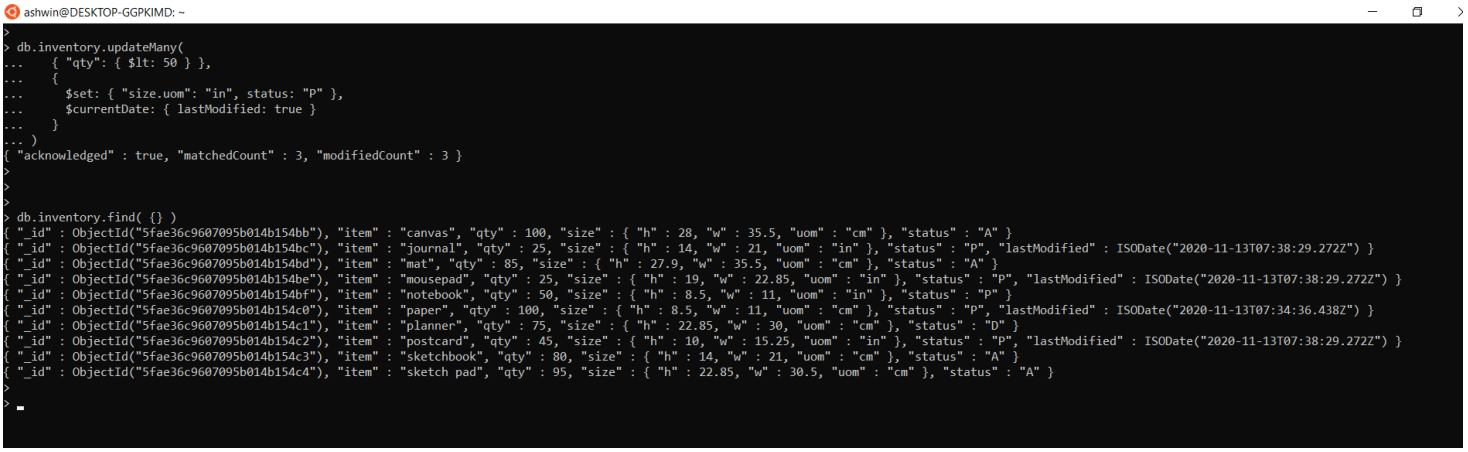
And then using updateOne() method on the inventory collection to update the *first* document where item equals "paper":-



We can see the change after the update and last modified timestamp. I have highlighted it.

db.collection.updateMany() : To update multiple documents.

Using this method on the inventory collection to update all documents where qty is less than 50:-

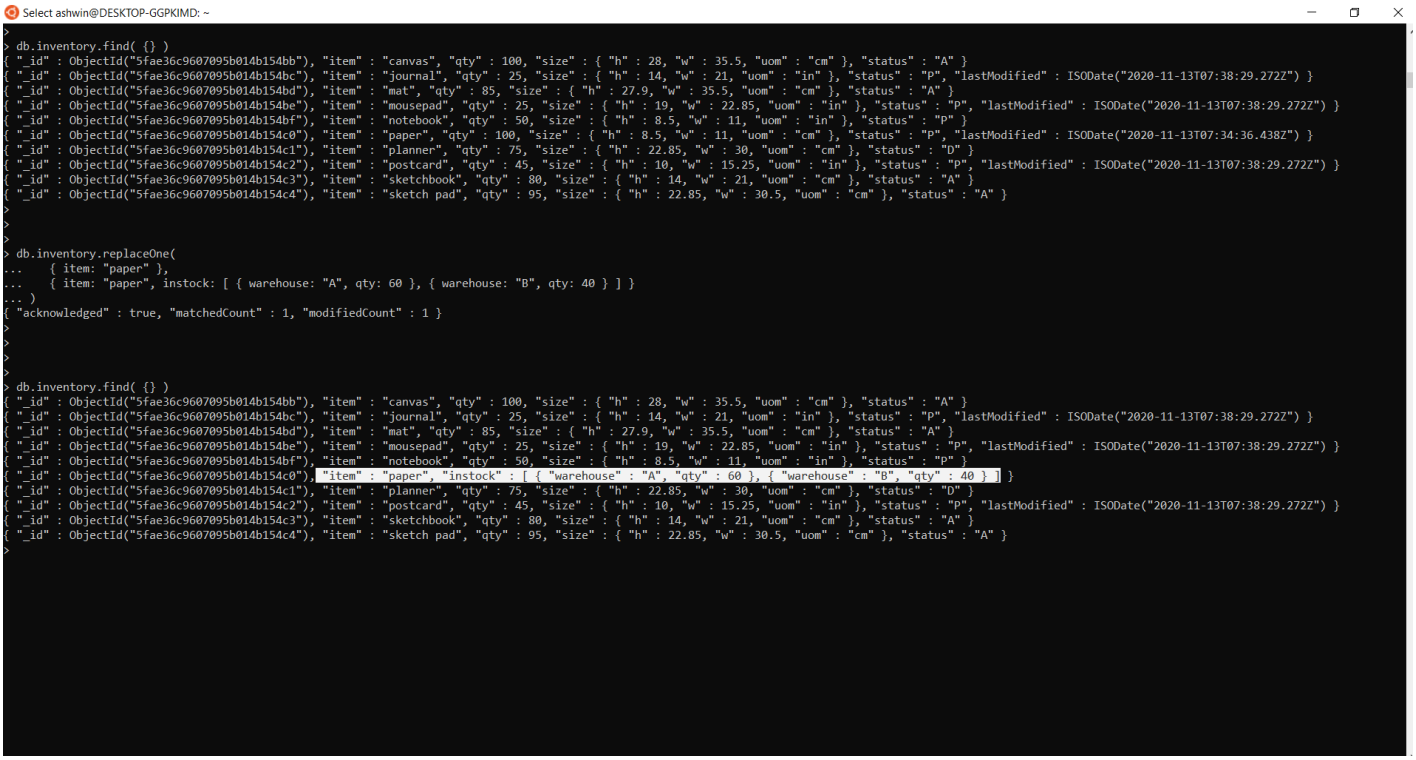


We can see 3 more documents getting updated.

Update by Replacing a document:

To replace the entire content of a document except for the _id field, pass an entirely new document as the second argument to **db.collection.replaceOne()**

To replace the *first* document from the inventory collection where item="paper" :-

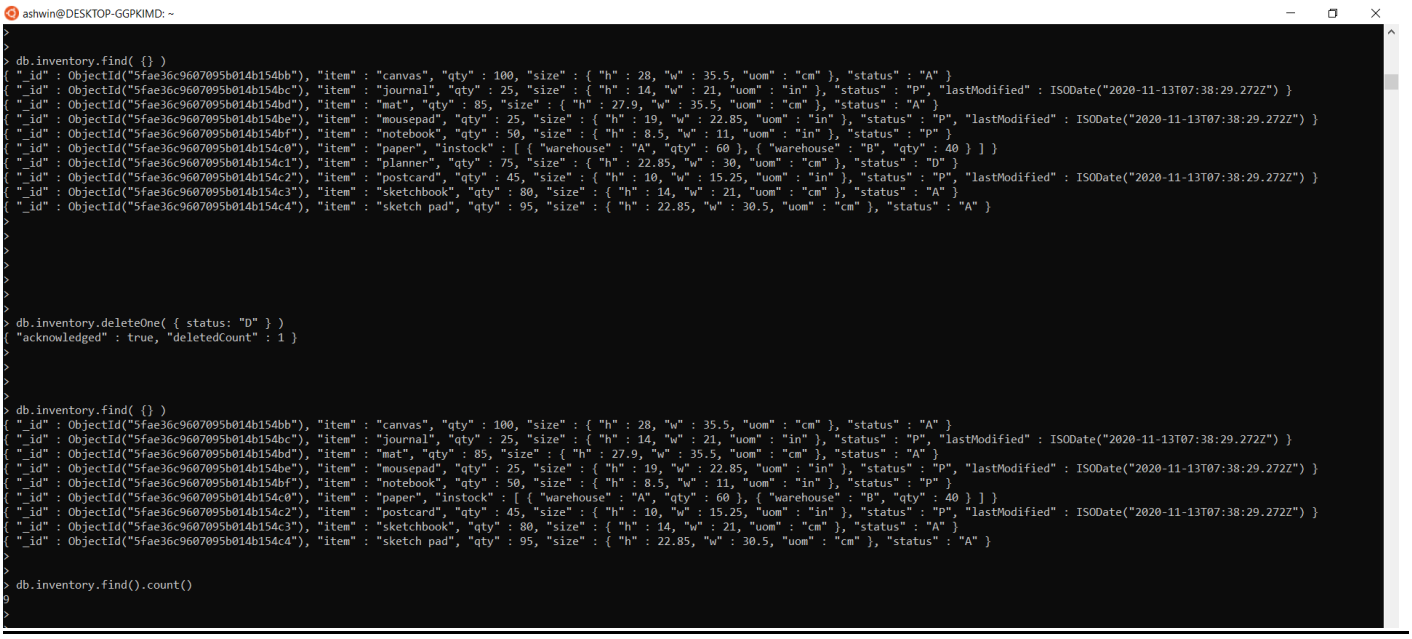


We can see that the *first* document from the inventory collection where item="paper" gets replaced by a new document.

DELETE:

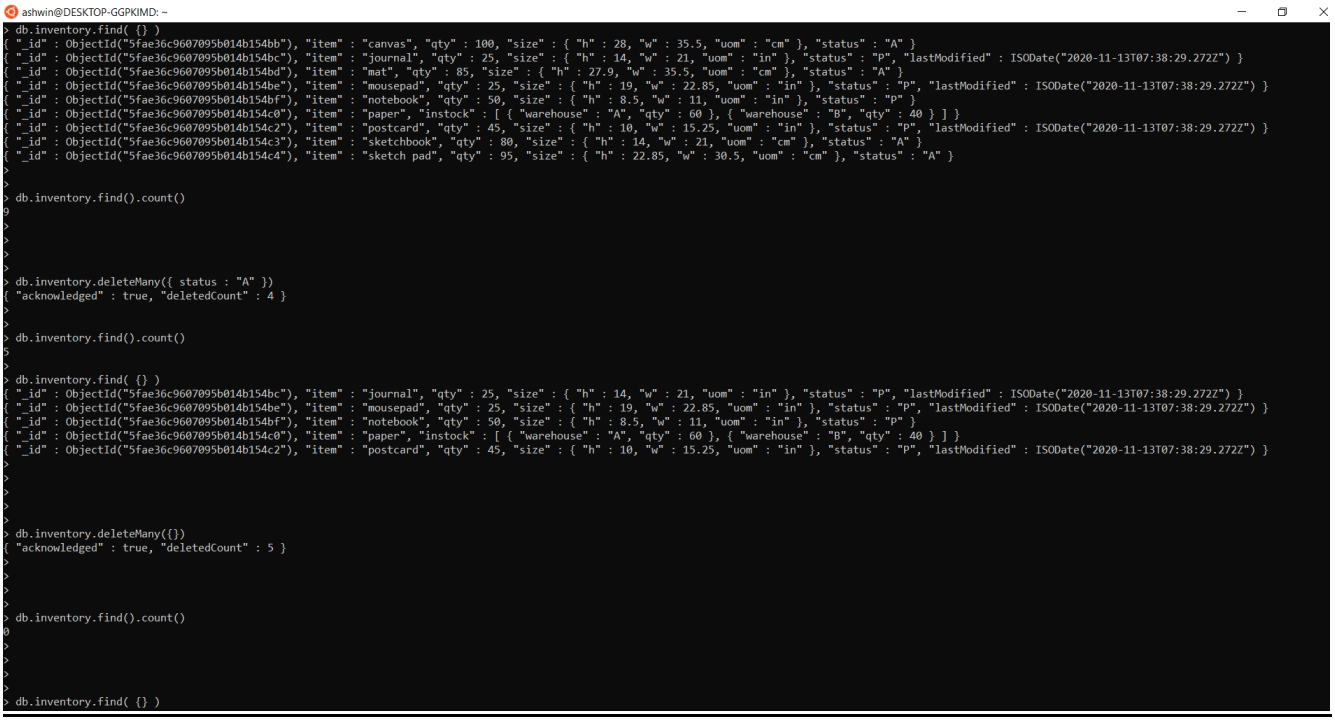
db.collection.deleteOne() : To delete at most a single document from collection that matches a specified filter (even though multiple documents may match the specified filter)

To delete the *first* document where status is "D" :-



db.collection.deleteMany() : To delete all documents from collection that match a deletion criteria.

To delete all documents from the inventory collection where the status field equals "A":-

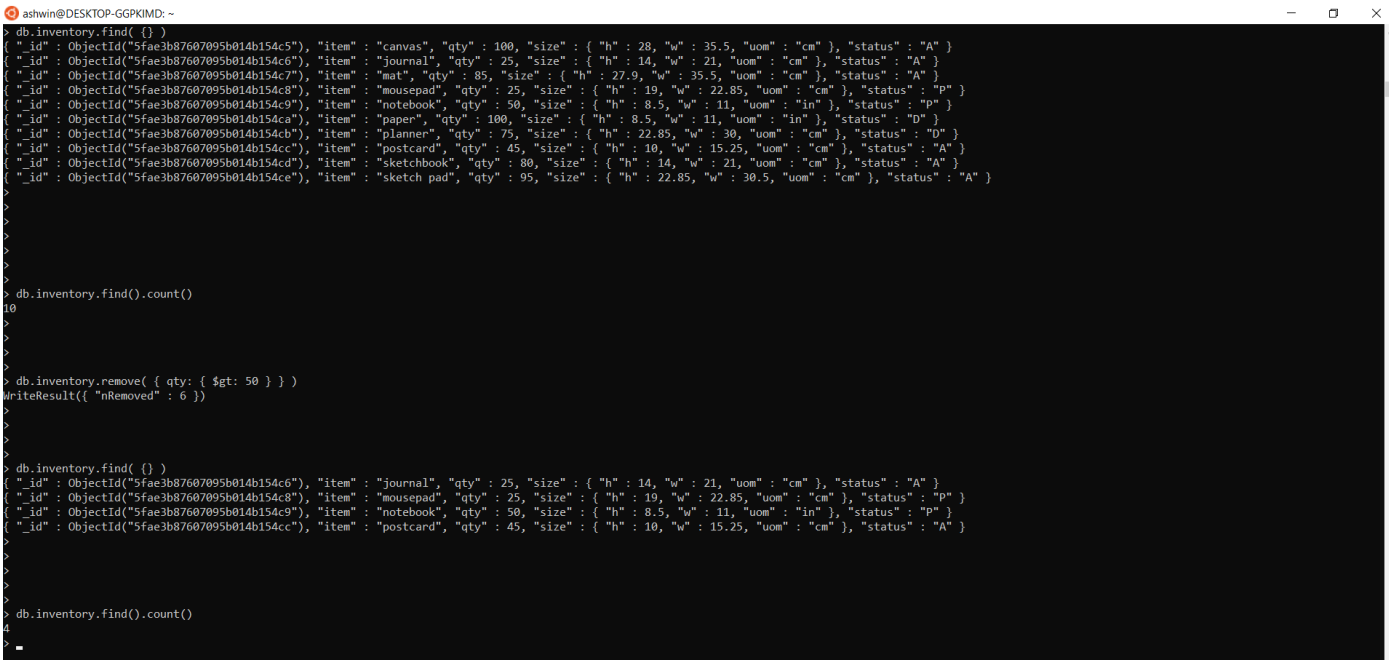


To delete all documents from a collection, pass an empty filter document {} to the db.collection.deleteMany() method.

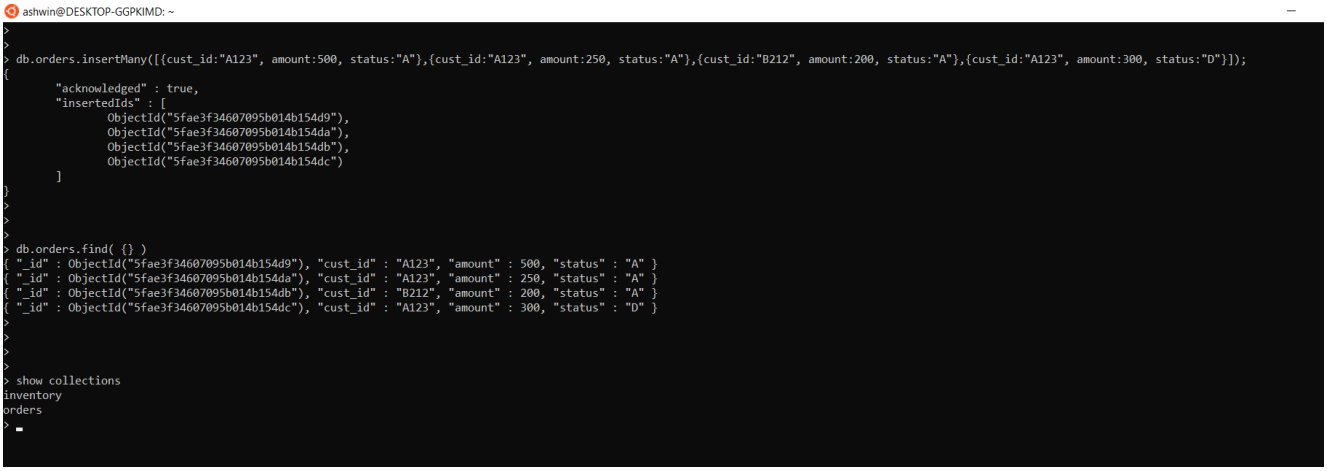
db.inventory.deleteMany({}) : deletes all documents from inventory collection. Document count will be 0 after this.

Deletion using remove() :

db.inventory .remove({ qty : { $gt : 50 } }) : removes all documents from colection inventory where qty is greater than 50 :-



Now I create a new collection called **orders** and insert some documents into it:-



Using this collection orders let us see Aggregation commands.

AGGREGATION:

Aggregation commands are ,

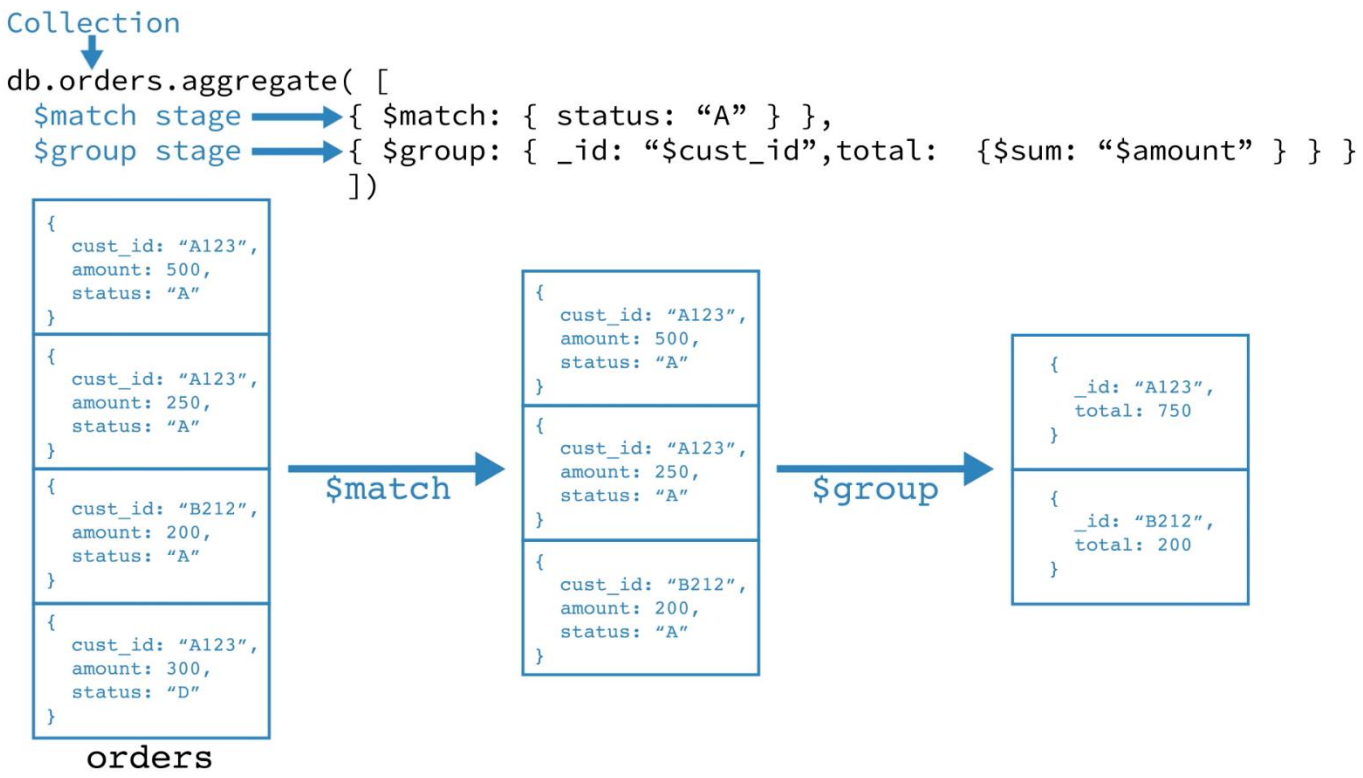
aggregate : Performs aggregation tasks such as group using the aggregation framework.

mapReduce : Performs map-reduce aggregation for large data sets.

distinct : Displays the distinct values found for a specified key in a collection or a view.

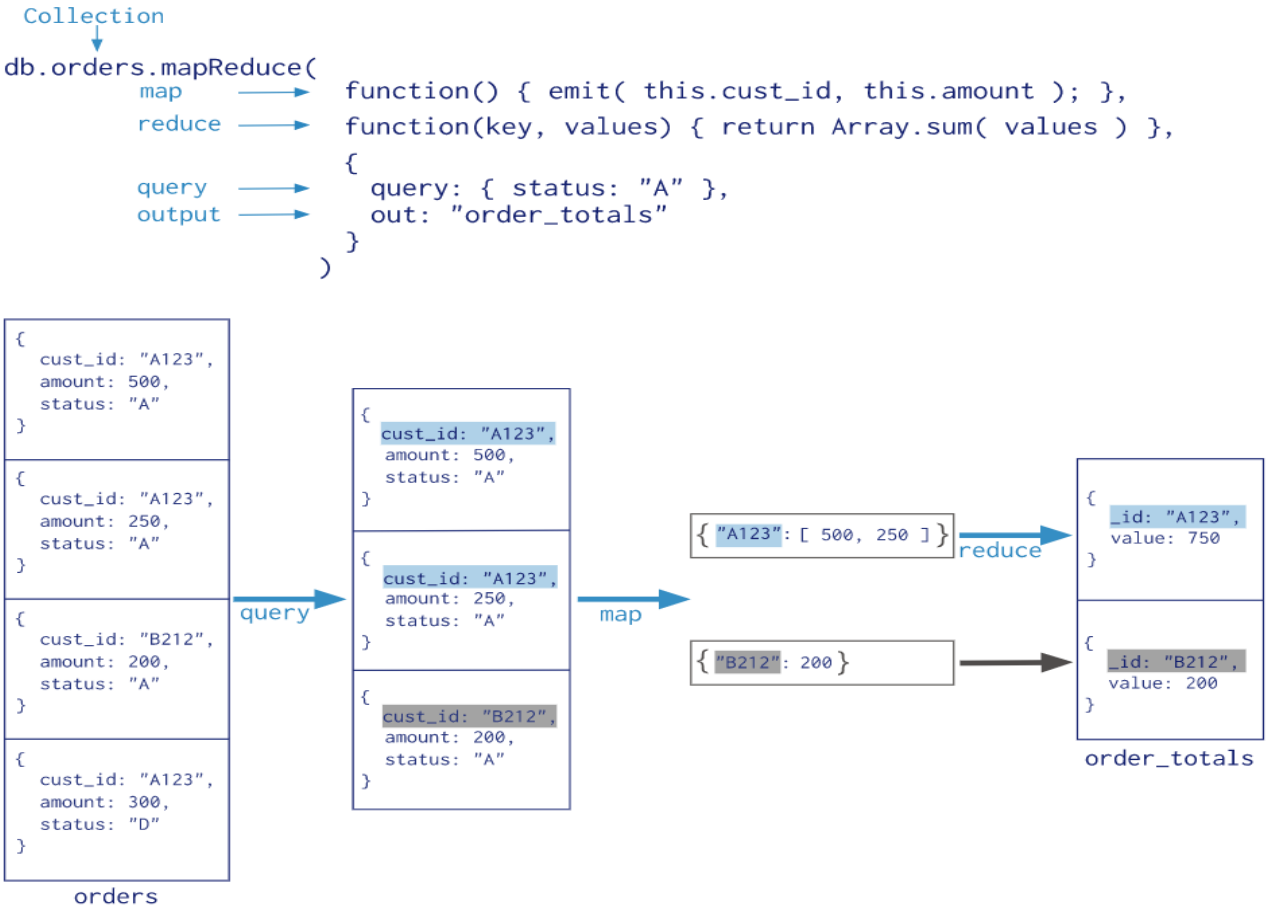
count : Counts the number of documents in a collection or a view.

aggregate

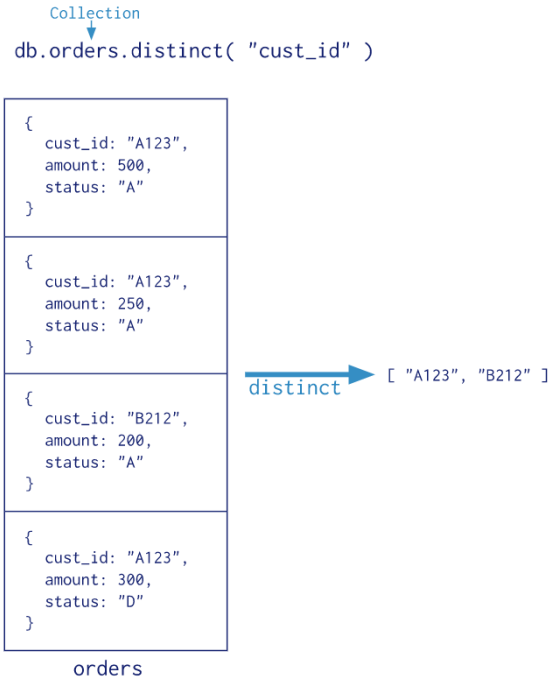


---

mapReduce



distinct



Commands executing these aggregation on orders collection:-

```
ashwin@DESKTOP-GGPKIMD: ~
>
>
>
>
> db.orders.aggregate([    { $match: { status: "A" } },    { $group: { _id: "$cust_id", total: { $sum: "$amount" } } } ])
{ "_id" : "B212", "total" : 200 }
{ "_id" : "A123", "total" : 750 }
>
>
>
>
> db.orders.distinct("cust_id")
[ "A123", "B212" ]
>
>
>
>
>
> db.orders.mapReduce( function() {emit(this.cust_id, this.amount);}, function(key, values) {return Array.sum(values)}, { query: {status: "A"}, out: "order_totals"} )
{
        "result" : "order_totals",
        "timeMillis" : 1511,
        "counts" : {
                "input" : 3,
                "emit" : 3,
                "reduce" : 1,
                "output" : 2
        },
        "ok" : 1
}
>
>
>
> db.order_totals.find( {} )
{ "_id" : "A123", "value" : 750 }
{ "_id" : "B212", "value" : 200 }
>
```

Thank you