Week 1 –

MONGO Database(No SQL)

Name : Suhan B Revankar
SRN : PES2UG19CS412
Section : G SECTION

Problem Statement	Use MongoDB and create a sample database and perform the following operations on it 1. Create documents with varying fields. 2. Use the insert and save commands. 3. Drop an existing database and a collection and use remove() to delete documents. 4. Update an existing document in a collection. (update()) 5. Use find() and findOne() for querying the database.

MongoDB CRUD Operations

CRUD operations create, read, update, and delete .

1.Create Operations

Create or insert operations add new documents to a collection. If the collection does not currently exist, insert operations will create the collection.

MongoDB provides the following methods to insert documents into a collection:

db.collection.insertOne()

db.collection.insertMany()

ex: db.collection.insertMany() can insert multiple documents into a collection. Pass an array of documents to the method.

The following example inserts three new documents into the inventory collection. If the documents do not specify an _id field, MongoDB adds the _id field with an ObjectId value to each document

Insert one :

```
db.inventory.insertOne(

  { item: "canvas", qty: 100, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm" } }

)
```

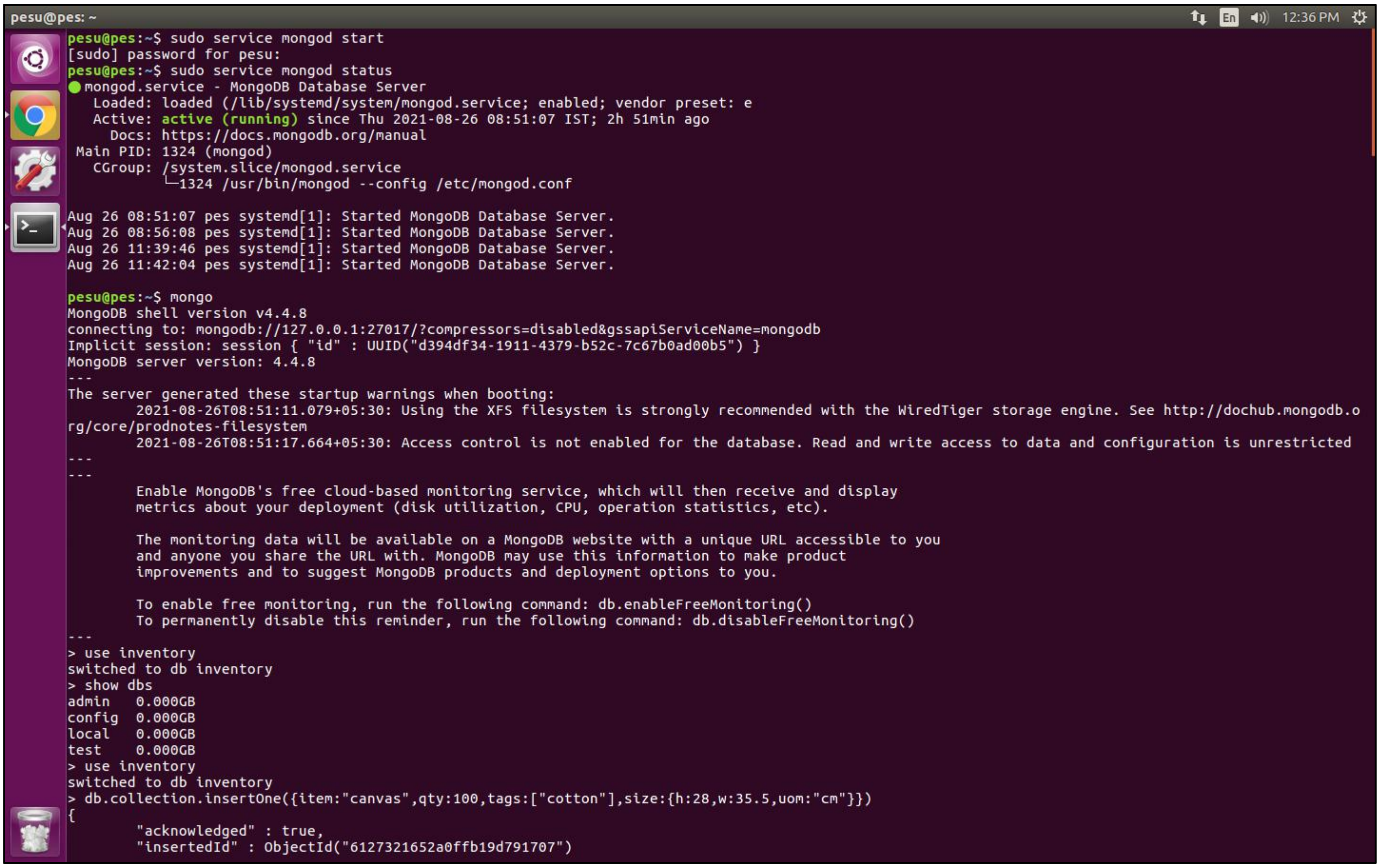
Insert many :

```
db.inventory.insertMany([

  { item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } },

  { item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },

  { item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } }])
```



```
pesu@pes:~$ sudo service mongod start
[sudo] password for pesu:
pesu@pes:~$ sudo service mongod status
● mongod.service - MongoDB Database Server
   Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: e
   Active: active (running) since Thu 2021-08-26 08:51:07 IST; 2h 51min ago
     Docs: https://docs.mongodb.org/manual
 Main PID: 1324 (mongod)
   CGroup: /system.slice/mongod.service
           └─1324 /usr/bin/mongod --config /etc/mongod.conf

Aug 26 08:51:07 pes systemd[1]: Started MongoDB Database Server.
Aug 26 08:56:08 pes systemd[1]: Started MongoDB Database Server.
Aug 26 11:39:46 pes systemd[1]: Started MongoDB Database Server.
Aug 26 11:42:04 pes systemd[1]: Started MongoDB Database Server.

pesu@pes:~$ mongo
MongoDB shell version v4.4.8
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("d394df34-1911-4379-b52c-7c67b0ad00b5") }
MongoDB server version: 4.4.8
---
The server generated these startup warnings when booting:
        2021-08-26T08:51:11.079+05:30: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.o
rg/core/prodnotes-filesystem
        2021-08-26T08:51:17.664+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use inventory
switched to db inventory
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
> use inventory
switched to db inventory
> db.collection.insertOne({item:"canvas",qty:100,tags:["cotton"],size:{h:28,w:35.5,uom:"cm"}})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("6127321652a0ffb19d791707")
```

2. Read Operations

Read operations retrieve documents from a collection; i.e. query a collection for documents. MongoDB provides the following methods to read documents from a collection:

db.collection.find()

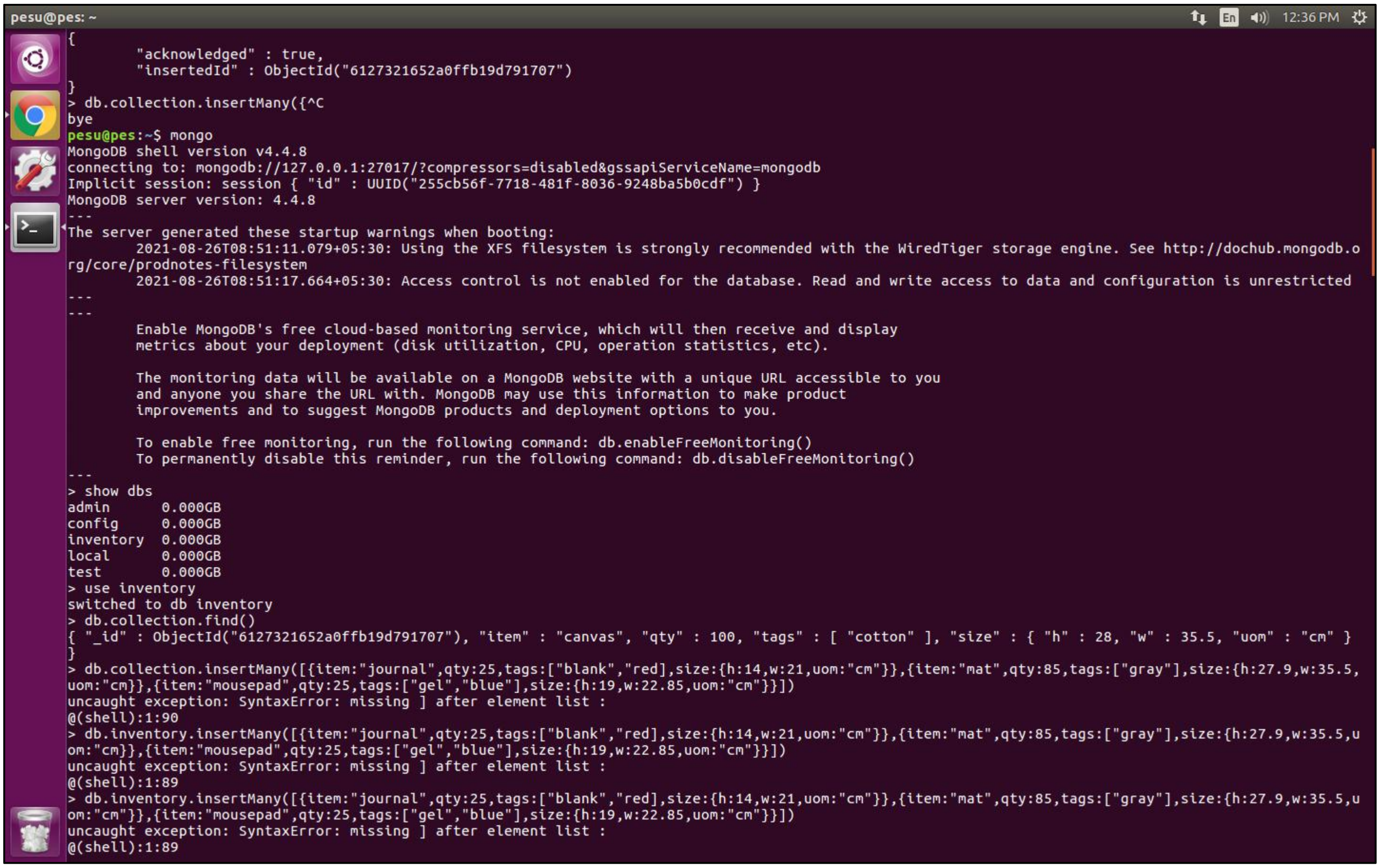
example :

The following query uses $gt to return documents where the value of qty is greater than 4.

db.collection.find({ qty: { $gt: 4 } })

Try using findone()

You can specify query filters or criteria that identify the documents to return.



```
{
        "acknowledged" : true,
        "insertedId" : ObjectId("6127321652a0ffb19d791707")
}
> db.collection.insertMany({^C
bye
pesu@pes:~$ mongo
MongoDB shell version v4.4.8
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("255cb56f-7718-481f-8036-9248ba5b0cdf") }
MongoDB server version: 4.4.8
---
The server generated these startup warnings when booting:
        2021-08-26T08:51:11.079+05:30: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.o
rg/core/prodnotes-filesystem
        2021-08-26T08:51:17.664+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin      0.000GB
config     0.000GB
inventory  0.000GB
local      0.000GB
test       0.000GB
> use inventory
switched to db inventory
> db.collection.find()
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" } 
}
> db.collection.insertMany([{item:"journal",qty:25,tags:["blank","red],size:{h:14,w:21,uom:"cm"}},{item:"mat",qty:85,tags:["gray"],size:{h:27.9,w:35.5,
uom:"cm}},{item:"mousepad",qty:25,tags:["gel","blue"],size:{h:19,w:22.85,uom:"cm"}}])
uncaught exception: SyntaxError: missing ] after element list :
@(shell):1:90
> db.inventory.insertMany([{item:"journal",qty:25,tags:["blank","red"],size:{h:14,w:21,uom:"cm"}},{item:"mat",qty:85,tags:["gray"],size:{h:27.9,w:35.5,u
om:"cm}},{item:"mousepad",qty:25,tags:["gel","blue"],size:{h:19,w:22.85,uom:"cm"}}])
uncaught exception: SyntaxError: missing ] after element list :
@(shell):1:89
> db.inventory.insertMany([{item:"journal",qty:25,tags:["blank","red"],size:{h:14,w:21,uom:"cm"}},{item:"mat",qty:85,tags:["gray"],size:{h:27.9,w:35.5,u
om:"cm"}},{item:"mousepad",qty:25,tags:["gel","blue"],size:{h:19,w:22.85,uom:"cm"}}])
uncaught exception: SyntaxError: missing ] after element list :
@(shell):1:89
```

3.Update Operations

Update operations modify existing documents in a collection. MongoDB provides the following methods to update documents of a collection:

db.collection.updateOne() New in version 3.2

db.collection.updateMany() New in version 3.2

db.collection.replaceOne() New in version 3.2

The updateOne() method has the following syntax:

```
db.collection.updateOne(
  <filter>,
  <update>,
  {
    upsert: <boolean>,
    writeConcern: <document>,
    collation: <document>,
    arrayFilters: [ <filterdocument1>, ... ],
```

<table>
<tr><th>Parameter</th><th>Type</th><th>Description</th></tr>
<tr><td>filter</td><td>document</td><td>The selection criteria for the update. The same query selectors as in the find() method are available.
Specify an empty document { } to update the first document returned in the collection.</td></tr>
<tr><td>update</td><td>document or pipeline</td><td>The modifications to apply. Can be one of the following:
<table>
<tr><th></th><th></th></tr>
<tr><td>Update document</td><td>Contains only update operator expressions.
For more information, see Update with an Update Operator Expressions Document</td></tr>
</table>
</td></tr>
</table>

		Aggregation pipeline (*Starting in MongoDB 4.2*)	Contains only the following aggregation stages: $addFields and its alias $set $project and its alias $unset $replaceRoot and its alias $replaceWith. For more information, see Update with an Aggregation Pipeline.
		To update with a replacement document, see db.collection.replaceOne().	
upsert	Boolean	Optional. When true, updateOne() either: Creates a new document if no documents match the filter. For more details see upsert behavior. Updates a single document that matches the filter. To avoid multiple upserts, ensure that the filter field(s) are uniquely indexed. Defaults to false, which does *not* insert a new document when no match is found.	

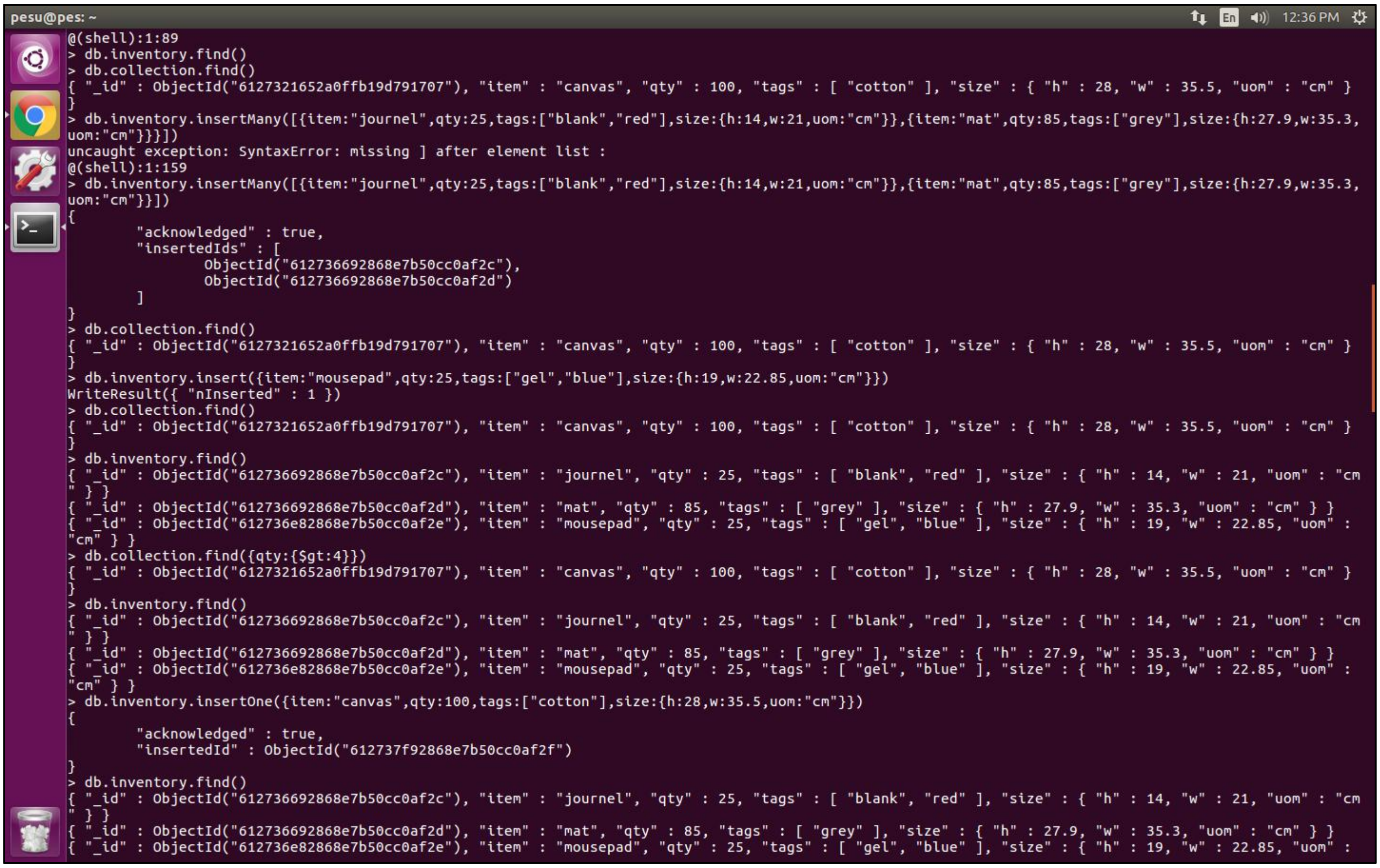
}

)

Ex:

```
 db.people.update(
  { name: "Andy" },
  {
  name: "Andy",
  rating: 1,
  score: 1
  },
  { upsert: true }
)
```

In MongoDB, update operations target a single collection. All write operations in MongoDB are atomic on the level of a single document.



```
@(shell):1:89
> db.inventory.find()
> db.collection.find()
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.insertMany([{item:"journel",qty:25,tags:["blank","red"],size:{h:14,w:21,uom:"cm"}},{item:"mat",qty:85,tags:["grey"],size:{h:27.9,w:35.3,
uom:"cm"}}}])
uncaught exception: SyntaxError: missing ] after element list :
@(shell):1:159
> db.inventory.insertMany([{item:"journel",qty:25,tags:["blank","red"],size:{h:14,w:21,uom:"cm"}},{item:"mat",qty:85,tags:["grey"],size:{h:27.9,w:35.3,
uom:"cm"}}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("612736692868e7b50cc0af2c"),
                ObjectId("612736692868e7b50cc0af2d")
        ]
}
> db.collection.find()
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.insert({item:"mousepad",qty:25,tags:["gel","blue"],size:{h:19,w:22.85,uom:"cm"}})
WriteResult({ "nInserted" : 1 })
> db.collection.find()
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.find()
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
"cm" } }
> db.collection.find({qty:{$gt:4}})
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.find()
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
"cm" } }
> db.inventory.insertOne({item:"canvas",qty:100,tags:["cotton"],size:{h:28,w:35.5,uom:"cm"}})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("612737f92868e7b50cc0af2f")
}
> db.inventory.find()
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
```

4.Delete Operations

Delete operations remove documents from a collection. MongoDB provides the following methods to delete documents of a collection:

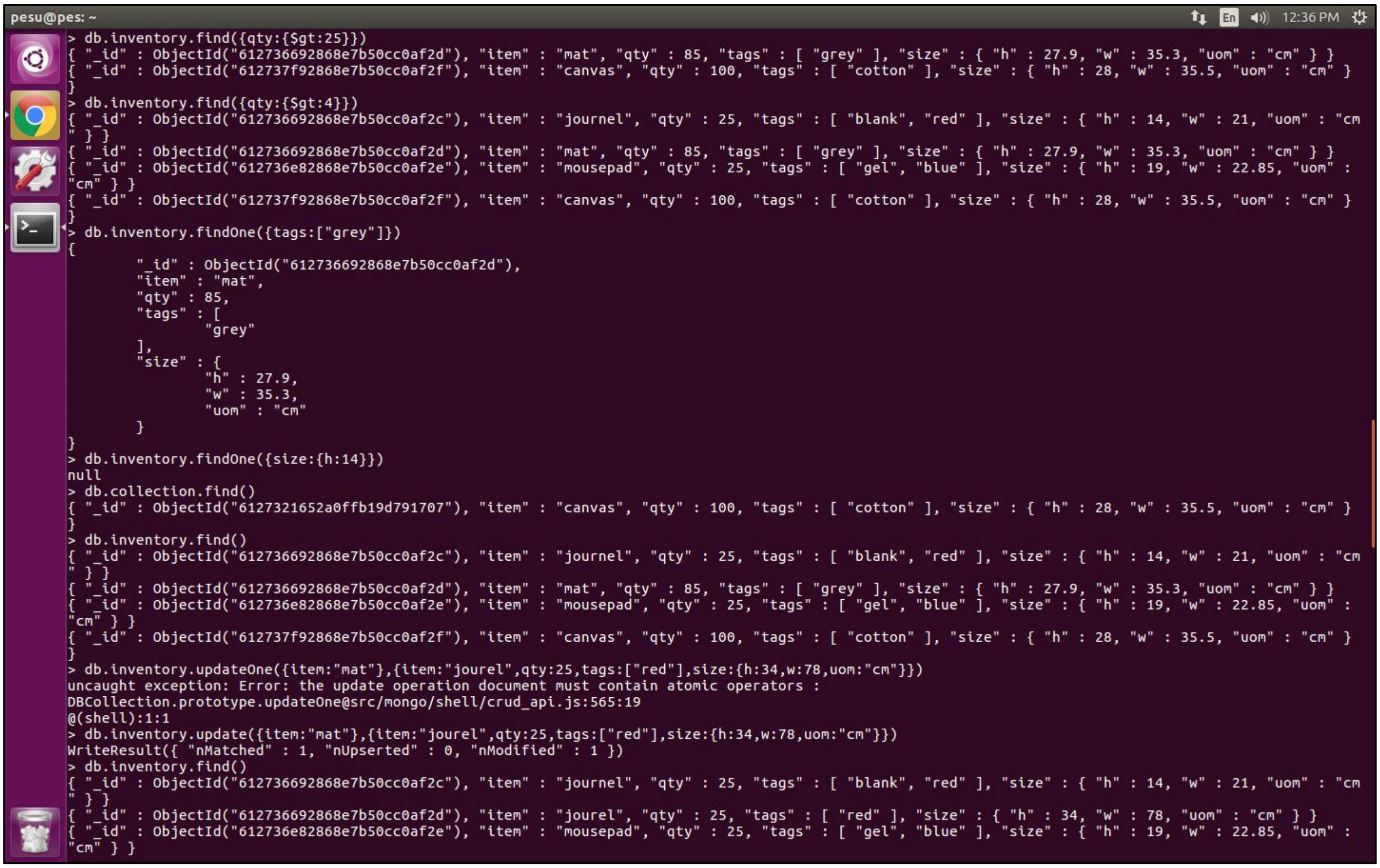
db.collection.deleteOne() New in version 3.2

db.collection.deleteMany() New in version 3.2

ex:

db.inventory.deleteMany({ status : "A" })

try deleteone()



```
> db.inventory.find({qty:{$gt:25}})
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612737f92868e7b50cc0af2f"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.find({qty:{$gt:4}})
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
"cm" } }
{ "_id" : ObjectId("612737f92868e7b50cc0af2f"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.findOne({tags:["grey"]})
{
        "_id" : ObjectId("612736692868e7b50cc0af2d"),
        "item" : "mat",
        "qty" : 85,
        "tags" : [
                "grey"
        ],
        "size" : {
                "h" : 27.9,
                "w" : 35.3,
                "uom" : "cm"
        }
}
> db.inventory.findOne({size:{h:14}})
null
> db.collection.find()
{ "_id" : ObjectId("6127321652a0ffb19d791707"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.find()
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "mat", "qty" : 85, "tags" : [ "grey" ], "size" : { "h" : 27.9, "w" : 35.3, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
"cm" } }
{ "_id" : ObjectId("612737f92868e7b50cc0af2f"), "item" : "canvas", "qty" : 100, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
}
> db.inventory.updateOne({item:"mat"},{item:"jourel",qty:25,tags:["red"],size:{h:34,w:78,uom:"cm"}})
uncaught exception: Error: the update operation document must contain atomic operators :
DBCollection.prototype.updateOne@src/mongo/shell/crud_api.js:565:19
@(shell):1:1
> db.inventory.update({item:"mat"},{item:"jourel",qty:25,tags:["red"],size:{h:34,w:78,uom:"cm"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.inventory.find()
{ "_id" : ObjectId("612736692868e7b50cc0af2c"), "item" : "journel", "qty" : 25, "tags" : [ "blank", "red" ], "size" : { "h" : 14, "w" : 21, "uom" : "cm
" } }
{ "_id" : ObjectId("612736692868e7b50cc0af2d"), "item" : "jourel", "qty" : 25, "tags" : [ "red" ], "size" : { "h" : 34, "w" : 78, "uom" : "cm" } }
{ "_id" : ObjectId("612736e82868e7b50cc0af2e"), "item" : "mousepad", "qty" : 25, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w" : 22.85, "uom" :
"cm" } }
```

*****