**MONGO DB – The Document DB**

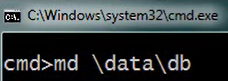
<https://university.mongodb.com/courses/M001/about>



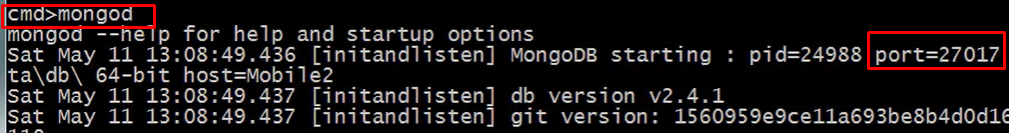
mongod.exe – run Mongo DB

mongo.exe – the Shell. Will use the Shell to connect to connect to the Mongo Server and perform some operation.

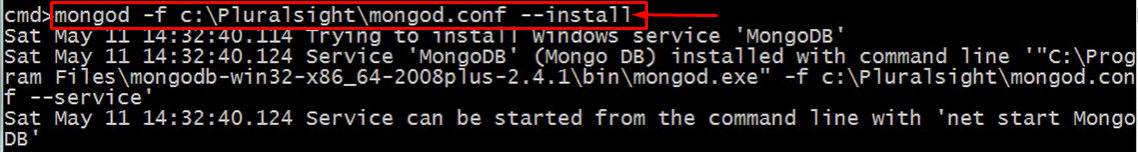
* Before starting Mongo DB need to create the location where all data will be saved. Default path is \data\db



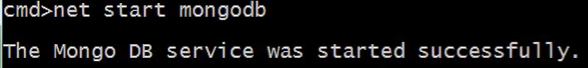
* Start running MongoDB on some server – the default port is 27017



* Install Mongo DB as a Service



* Start Mongo DB that was installed as a Service



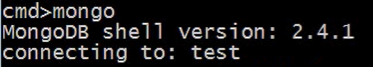
* List all the DB running on this machine



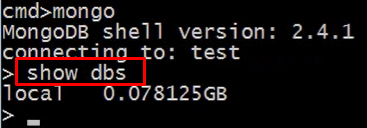
* Stop Mongo DB Service



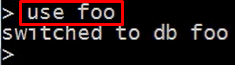
* Run Shell to connect to the Mongo Service with default name and default port. After this command we are in the Shell.



* List of the DB running in Mongo Service - After starting Shell once it is connected to the Mongo Service, now all commands in shell are running against Mongo Service



* In Shell switch from one DB, running under Mongo Service, to another - switch to the foo DB. Any operation after this switch will be done against this DB



* The name that Shell is currently connected to. It means that any operation t



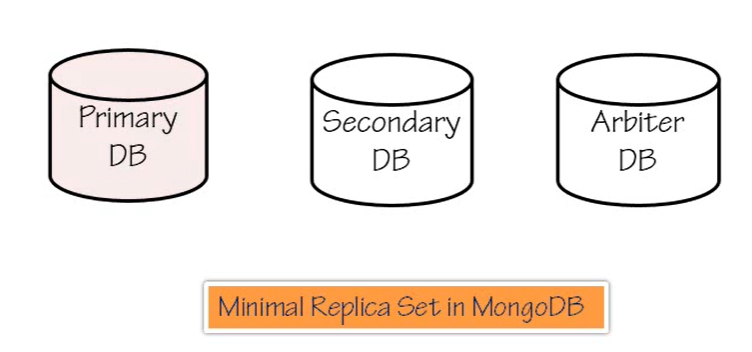
* Store the data file in specific data directory



Change Logging Verbosity

Where to store the Log File - supply its name and the location

**Replica Set**



Mongo DB supports the arrangement calls **REPLICA SET**.

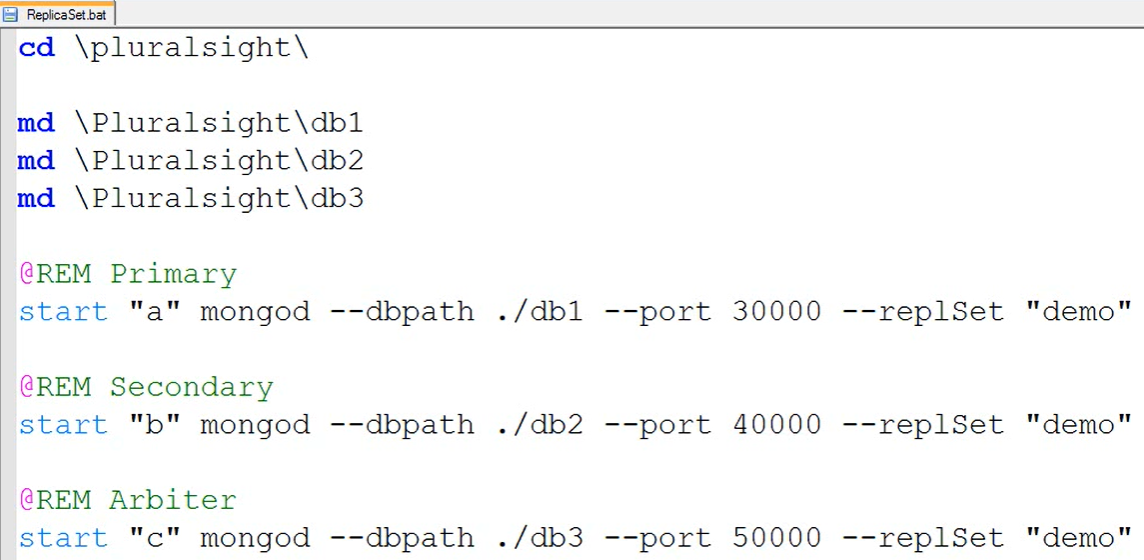
This REPLCICA SET contains the Primary DB, one or number of Secondary DBs and Arbiter DB.

Primary DB – is the only one writable instance. All clients that want to write to the DB, have to be connected to the Primary DB. The attempt to write to the Secondary DB will fail.

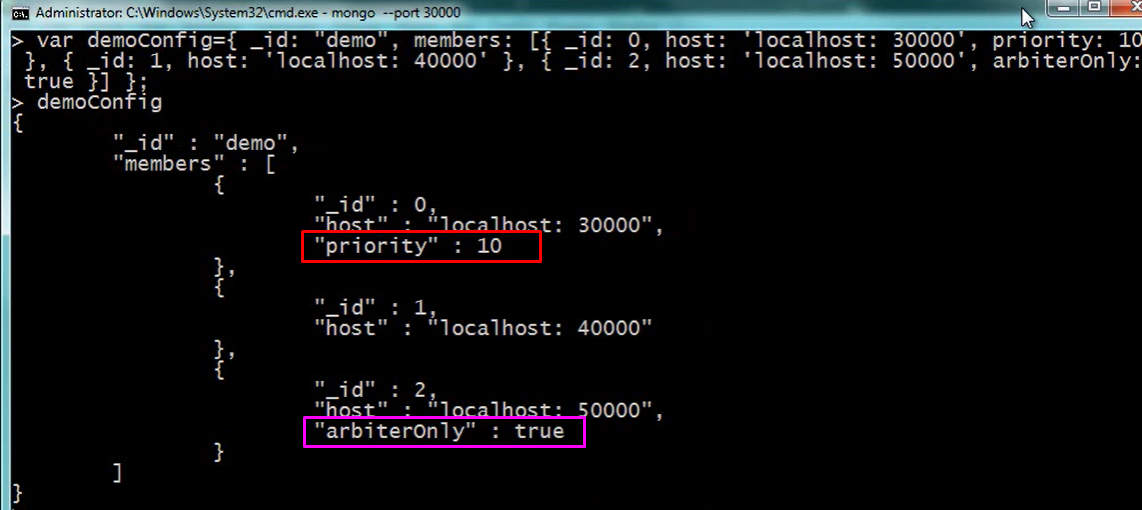
Secondary DB – it is read-only instance. The data in Secondary DBs will be replicated from the Primary DB eventually ( that was we called Eventual Consistency). At some point if Primary DB will fail, one of the Secondary DB will become the Primary one.

Arbiter DB – Arbiter provides the additional vote to decide who will be the Primary DB once the Primary DB failed. Arbiter DB has no data. It's purpose is to break a tie in an election. You may not be required an Arbiter DB machine if you have the odd number of DB in your farm.

The batch below create three Mongo DB on the same server listening on the different port and will participate in Replica Set named "demo"

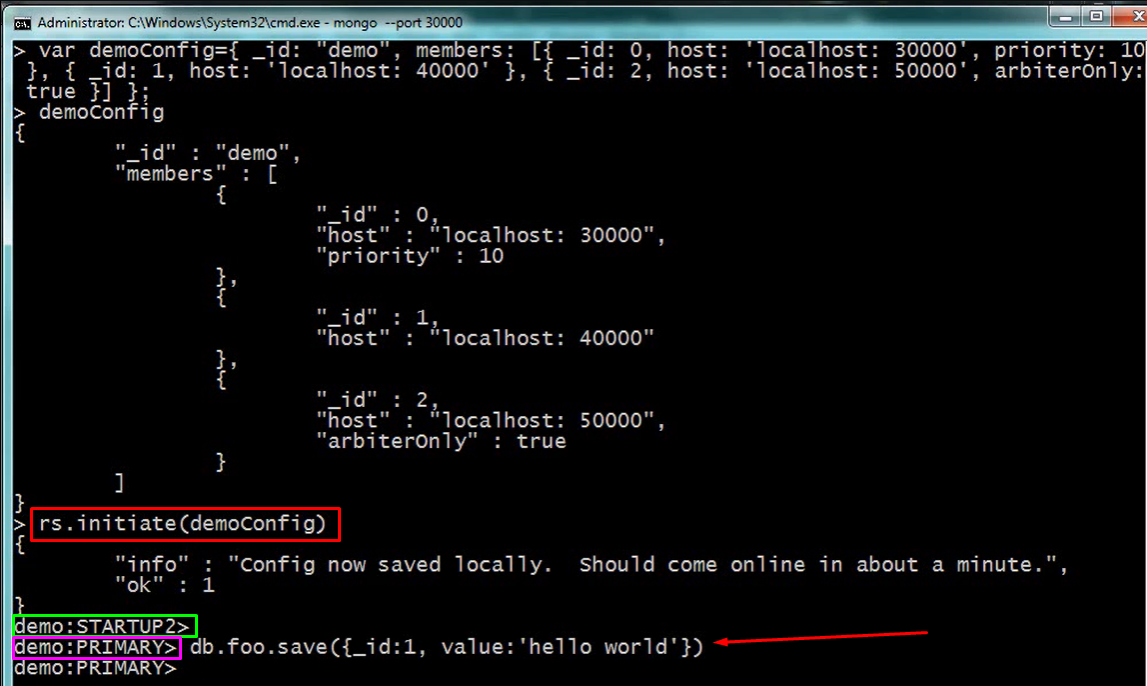


Now we can open Shell for some of the mongo DB (using command mongo –port …) and define the config where the DB with highest priority will be define as Primary DB and also there is an option to define which DB is the Arbitrary one.

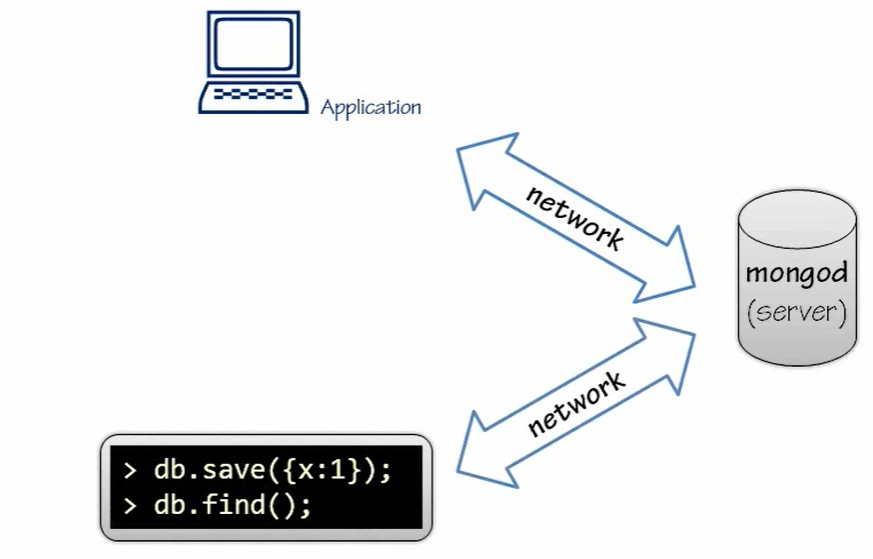


Now we want to initialize the replication – the red command. after starting the Replication the prompt was changed. Now we have Replica Name followed by status of the current server were are connected to (Replication Status)

The pink one display that the Replica Set was successfully initiated and this server we are connected to is the Primary and it can accept write commands



**Shell**



**Shell** - is the application that allows you interactively get inside into what Mongo Server is doing. The Shell is using the same protocol as you application has. It simply another app that talks to Mongo DB.

Shell Invoking Syntax

* "Blind Command" – Shell will execute the command against Mongo Server without parting the interactive Shell



* "Blind" Script – run the JavaScript file. Shell will load and execute this js file



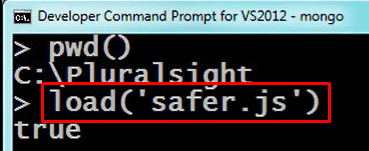


* Run the script and remain in the Shell

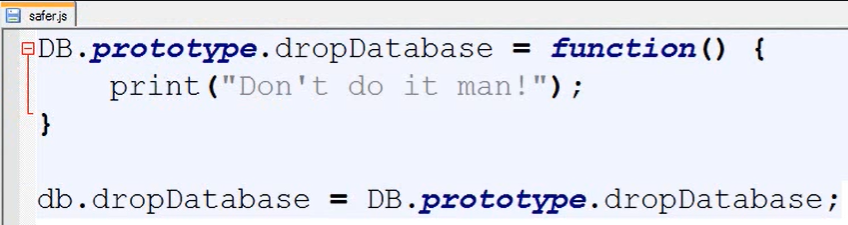




* Load a script into the Shell, which is already existing as a file(this command is executed once we are already in the Shell)



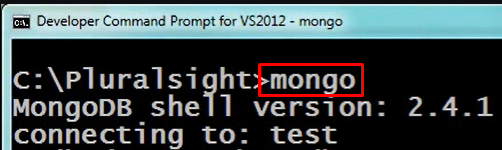
* Run script each time we enter the Shell – for example there is JS file contains code that protect from dropping DB once we are running inside the Shell
  + Take some Javascript code



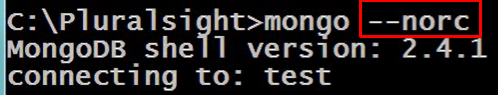
* + Save the content that you want to run each time that Shell is run with the under next default name (.mongorc.js) in the default location below



* + Now when I start some shell, this js code from the .mongorc.js will be run

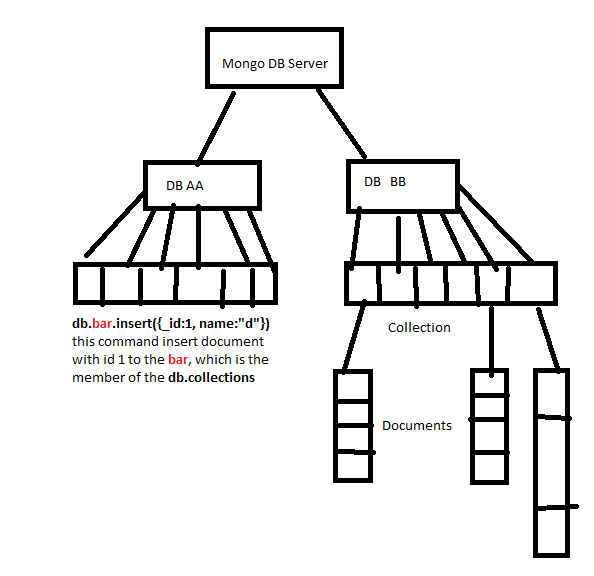


* Disable default running script on starting Shell



**Saving Data**

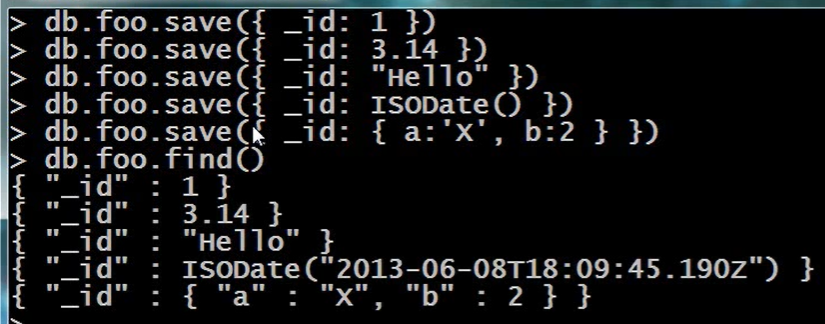
BSON - Efficient data storage serialization format



**Rules for saving data in Mongo DB:**

1. A document must have an **\_id** field- this is only the one scheme rule for the document.

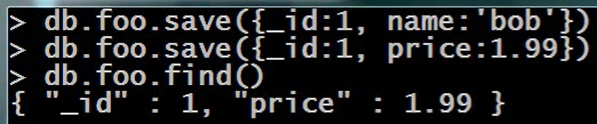
Legal id types in Mango DB



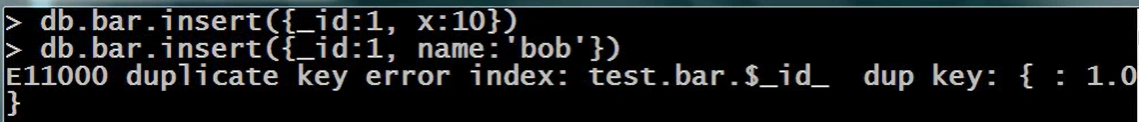
Not Legal is Array



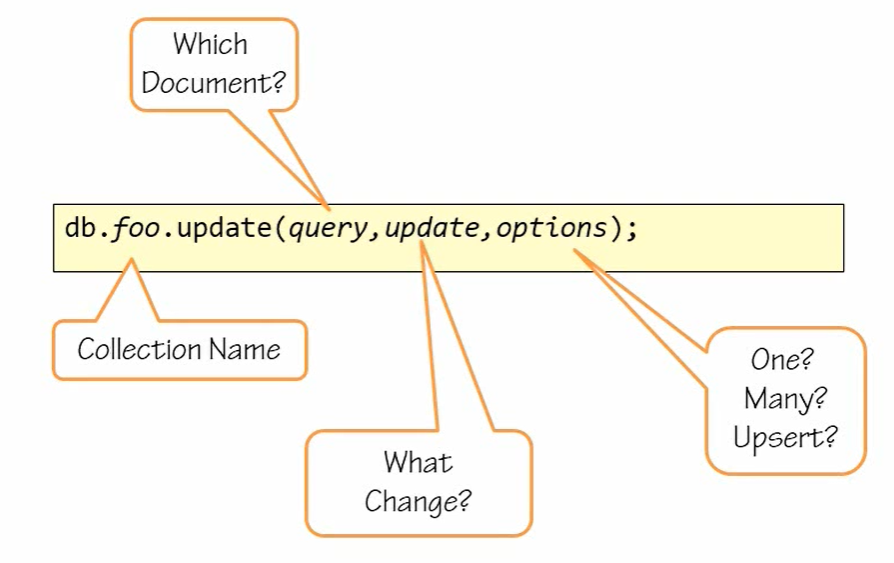
1. The max size of the document is limited to 16MB
2. Save command is override the another content with the same \_id



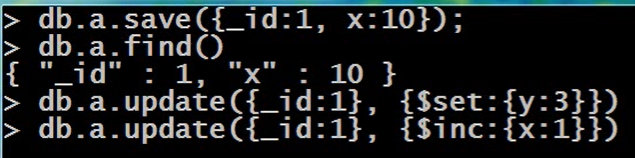
1. Insert will not allow to update the existing one document from the same collection



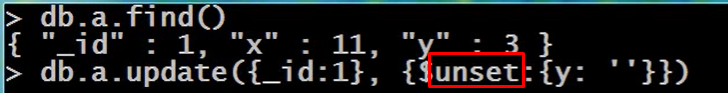
**Update – atomic command**



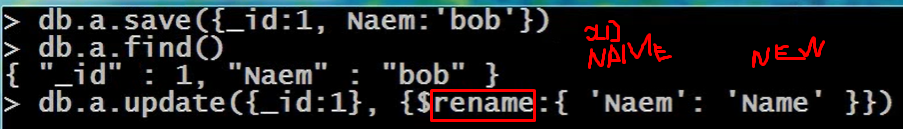
* Increment value + initialize new parameter in document



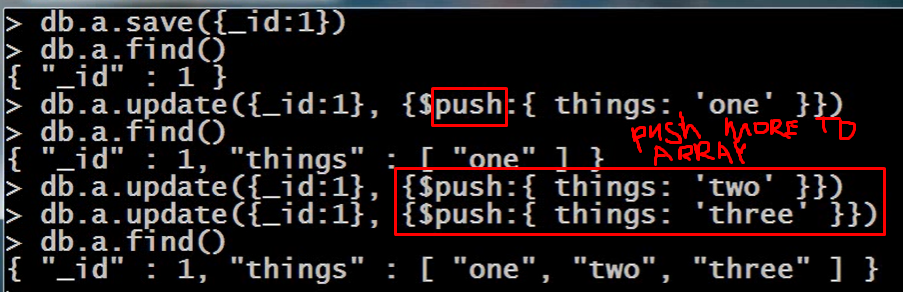
* Remove some parameter



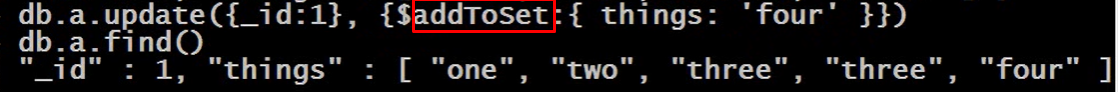
* Rename parameter in document



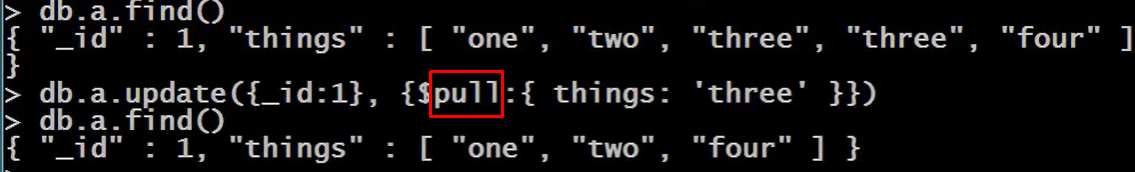
* Array Operation
  + Adding array with values to the document – does not check duplicates during inserting



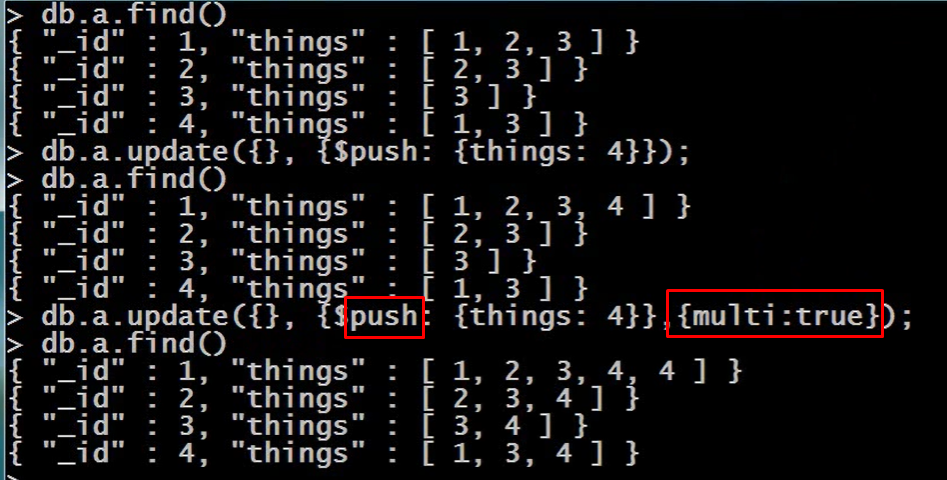
* Add to the array of the document some values just in case that they are not already exists in array



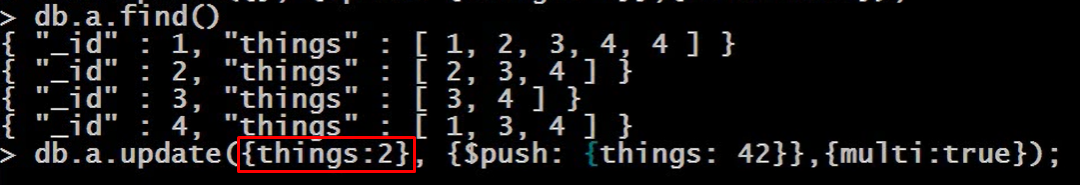
* Remove all instances of the supplies element from array



* Multiple update



* Update only documents which array names "things" has value 2. Just for this documents, the element 52 will be added to the array "things"



**Find and Modify**

\*\*\* By default this operator returns the document before applying changes on it



**Sort Order** – only the first document or the last document matching the "query" criteria may be modified

**Update** – what is the change that we want to make

**Upsert** – set it to true which meaning creating new record (document) if no one exists which matching the "query" criteria

**Remove** – delete a document that match the criteria defined in "query", but delete exactly one record

**New** – if set this parameter to True, it will return the document after the change was mode on it.

**Fields** – defined which part of the document should be returned as Return value from this function.

**MONGO DB Atlas**

is a multi-cloud database service by the same people that build MongoDB. Atlas simplifies deploying and managing your databases while offering the versatility you need to build resilient and performant global applications on the cloud providers of your choice.

<https://docs.atlas.mongodb.com/>