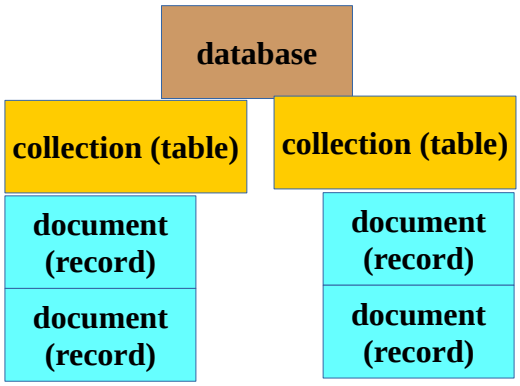
MongoDB

update old mongodb / install new

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 7F0CEB10
echo 'deb http://downloads-distro.mongodb.org/repo/ubuntu-upstart dist 10gen' | sudo tee /etc/apt/sources.list.d/mongodb.list
sudo apt-get update
sudo apt-get install mongodb-org
```

Start mongodb → $ sudo service mongodb start
Stop mongodb → **$** sudo service mongodb stop
Restart mongodb → **$** sudo service mongodb restart
Running mongodb → **$** mongo

Help → > db.help
Statistik → > db.stats()



Create database → > use database_name *// create*
Check currently selected database → > db
Check all databases list → > show dbs
Drop database
> use database_name *// Use one of database*
> db.dropDatabase() *// Delete database*

Create collection
> use test *// test → database name*
> db.createCollection("mycol") *// mycol → collection name*
Check all collection in selected database
> use test *// test → database name*
> show collections
Create collection automatically when insert document
> db.language.insert({"name" : "javascript"}) *// language → collection name*
Drop collection
> show collections
> db.mycol.drop() *// mycol → collection name (delete collection)*

Insert document to collection

```
> db.language.insert({
... name : "Javascript",
... tags : ["js", "nodejs", "mongodb"],
... by : "WOKKI"
... })
```

Check document in collection

```
> db.language.find()
{ "_id" : ObjectId("5742fa3edf139eca8b14e864"), "name" : "Javascript", "tags" : [  "js", "nodejs",  "mongodb" ], "by" : "WOKKI" }
```

Alternatif check document

```
> db.language.find().pretty()
```

mongodb

Add multiple document to collection

```
> db.language.insert([
... {
... name : "php",
... tags : ["php", "mysql"],
... by : "WAKKA"
... },
... {
... name : "c++",
... tags : ["c++", "c"],
... by : "OSN"
... }
... ])
```

Find field name and field value certainly document in collection

```
> db.language.find({"by" : "WAKKA"}) // find field "by" : "WAKKA" in language collection
{ "_id" : ObjectId("57430060df139eca8b14e866"), "name" : "php", "tags" : [ "php", "mysql" ], "by" : "WAKKA" }
```

Find field name and field value using <, <=, >, >=, !=

```
> use stock // create database → stock
> db.createCollection("phone") // create collection → phone
> db.phone.insert({ merk : "sony", amount : 100 }) // insert document
> db.phone.insert({ merk : "samsung", amount : 50 }) // insert document
> db.phone.insert({merk : "asus", amount : 20}) // insert document
> db.phone.insert({merk:"apple", amount:10}) // insert document
```

Find amount< 50 : > db.phone.find({"amount":{$lt:50}})
Find amount <= 50 : > db.phone.find({"amount":{$lte:50}})
Find amount > 50 : > db.phone.find({"amount":{$gt:50}}) **.pretty()**
Find amount >=50 : > db.phone.find({"amount":{$gte:50}}) **.pretty()**
Find amount !=50 : > db.phone.find({"amount":{$ne:50}}) **.pretty()**

Find some document on a collection

```
> db.phone.find({$or:[{"merk":"asus"},{"merk":"apple"}]}) // find field merk document
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
```

Find document based on some field

```
> db.phone.find({"merk":"samsung","amount":50}) // if there are two document that have some merk
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
```

Update Field

```
> db.phone.find() // first, see all document
{ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "sony", "amount" : 100 }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
> db.phone.update({"merk":"sony", "amount":100},{$set:{"merk":"sony","amount":150}})
> db.phone.find() // check, all document again after update
{ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "sony", "amount" : 150 }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
```

Update all document based on field

```
> db.shephone.find()
{ "_id" : ObjectId("57613ee257b79693d720f05b"), "merk" : "samsung", "type" : "s7" }
{ "_id" : ObjectId("5761493447f8593a35aa46af"), "merk" : "samsung", "type" : "k1" }
{ "_id" : ObjectId("5746cced25def5360e6f535c"), "merk" : "mito", "type" : "new1" }
{ "_id" : ObjectId("57613ee257b79693d720f05a"), "merk" : "samsung", "type" : "j10" }

> db.shephone.update({"merk":"samsung"},{$set:{"condition":"new"}},{multi:true}) // multi will set all field
> db.shephone.find()
{ "_id" : ObjectId("5746cced25def5360e6f535c"), "merk" : "mito", "type" : "new1" }
{ "_id" : ObjectId("57613ee257b79693d720f05a"), "condition" : "new", "merk" : "samsung", "type" : "j10" }
{ "_id" : ObjectId("57613ee257b79693d720f05b"), "condition" : "new", "merk" : "samsung", "type" : "s7" }
{ "_id" : ObjectId("5761493447f8593a35aa46af"), "condition" : "new", "merk" : "samsung", "type" : "k1" }
```

If document found replace but if not insert new document

```
> db.shephone.update({"merk":"nokia"},{$set:{"design":"flat"}},{upsert:true, multi:true}) // find nokia
> db.shephone.find()
{ "_id" : ObjectId("5746cced25def5360e6f535c"), "merk" : "mito", "type" : "new1" }
{ "_id" : ObjectId("57613ee257b79693d720f05a"), "condition" : "new", "design" : "flat", "merk" : "samsung",
"type" : "j10" }
{ "_id" : ObjectId("57613ee257b79693d720f05b"), "condition" : "new", "design" : "flat", "merk" : "samsung",
"type" : "s7" }
{ "_id" : ObjectId("5761493447f8593a35aa46af"), "condition" : "new", "design" : "flat", "merk" : "samsung",
"type" : "k1" }
{ "_id" : ObjectId("57615c4557b79693d720f05c"), "design" : "flat", "merk" : "nokia" } //insert new document
// upsert will insert new document if field that searched doesn't exist
```

Update field using _id

```
> db.phone.update({ "_id" : ObjectId("57430d88df139eca8b14e868"),"amount":150},{$set:{"amount":500}})
> db.phone.find()
{ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "sony", "amount" : 500 } // just only update amount
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
```

Replace old document with new document

```
> db.phone.find() // first, see all document
{ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "sony", "amount" : 150 } //we will replace red
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }

> db.phone.save({ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "oppo", "amount" : 150})
> db.phone.find() // check, all document again after replace
{ "_id" : ObjectId("57430d88df139eca8b14e868"), "merk" : "oppo", "amount" : 150 }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
```

Delete document based on field

```
> db.phone.remove({"merk":"sony"}) // remove all document that have field merk : sony
```

Delete document based on id

```
> db.phone.remove({"_id" : ObjectId("5743e61fc553296fdfc2946a")}) // only one document will deleted
```

Remove all document

```
> db.phone.remove()
```

Add new field to document

```
> db.phone.find() // example data
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }

> db.phone.update({ _id:ObjectId("5743151ddf139eca8b14e86d")},{$set:{"condition":"new"}})
> db.phone.find() // after add new field
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "condition" : "new", "merk" : "apple" }
```

Delete a field in document

```
> db.phone.find() // example data
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "condition" : "new", "merk" : "apple" }

> db.phone.update(
... { "_id" : ObjectId("5743151ddf139eca8b14e86d") },  // delete based on id
... { $unset: { condition: ""} } ) // delete field condition (actually not delete but replace element with null)

> db.phone.find() // after delete a field
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony", "amount" : 100 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "merk" : "apple" }
```

Add new array field to document

```
> db.phone.find() // example data before add field array color
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 } // add field array color
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony", "amount" : 100 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "merk" : "apple" }
> db.phone.update(    {"_id" : ObjectId("57430df1df139eca8b14e869")   },
                      {   $set:{"color":["red","blue","yellow"]}              }        )

> db.phone.find() // after add field array color
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony", "amount" : 100 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "merk" : "apple" }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "amount" : 50, "color" : [  "red",  "blue",  "yellow" ], "merk" : "samsung" }
```

Show document (record) only some field (projection)

```
> db.phone.find({},{"merk":1}) // show document only id & merk (1 = show, 0=hide)
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung" }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus" }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple" }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony" }
```

Hide ID

```
> db.phone.find({},{"merk":1, _id:0}) // show document only merk (1 = show), and hide _id (0=hide)
{ "merk" : "samsung" }
{ "merk" : "asus" }
{ "merk" : "apple" }
{ "merk" : "sony" }
```

Limit record showed

```
> db.phone.find({},{merk:1, _id:0}).limit(2) // only show two record
{ "merk" : "samsung" }
{ "merk" : "asus" }
```

Skip record

```
> db.phone.find().limit(1).skip(2) // limit(1) → only show 1 record , skip(2) →  start at record 3
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10
> db.phone.find({},{merk:1}).limit(2).skip(1) // limit(2) → show 2 record , skip(1) →  start at record 2
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus" }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple" }
```

Sort Document

```
> db.phone.find().sort({"merk":1}) // sort ascending based on  field merk (1=ascending; -1=descending)
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony", "amount" : 100 }
```

Indexing (make fast search in document)

```
> db.phone.ensureIndex({"merk":1}) // make index for field merk (1= ascending; -1 = descending)
> db.phone.ensureIndex({"merk":1,"amount":-1}) // make index merk (1) and amount (-1)
```

mongodb

Checking if a field using Index

```
> db.myphone.find({merk:"SAMSUNG"}).explain()
{
    "cursor" : "BtreeCursor merk_1",
    "isMultiKey" : false,
    "n" : 3,
    "nscannedObjects" : 3,
    "nscanned" : 3,
    "nscannedObjectsAllPlans" : 3,
    "nscannedAllPlans" : 3,
    "scanAndOrder" : false,
    "indexOnly" : false,
    "nYields" : 0,
    "nChunkSkips" : 0,
    "millis" : 0,
    "indexBounds" : {
        "merk" : [
            [
                "SAMSUNG",
                "SAMSUNG"
            ]
        ]
    },
    "server" : "Aspire:27017"
}
```

example data array for import : myphone.json

```
{
    "_id" : ObjectId("5745b5f49ec46985e4dc0990"),
    "merk" : "APPLE",
    "type" : "7",
    "price" : 6000000,
    "amount" : 2,
    "desc" : [  {  "color" : "red",  "width" : "150 cm" } ]
}
```

DeleteIndex

```
> db.myphone.dropIndex({merk:1});
```

Count document

```
> db.myphone.find().count() // count how many document (record) on collection (table)
> db.myphone.find({"merk":"SAMSUNG"}).count() // find samsung on collection & count
```

Import file type json to mongodb collection (use for 1 or 2 document)

phonetype.json

```
[ // may not used
    { merk: "samsung", type: "j1", price: 1500000},
    { merk: "samsung", type: "s7", price: 4500000}
] // may not used
```

```
// no nedd to enter mongodb, just enter direktori where phonetype.json there and automatically create collection phonetype
f@Aspire:~/Documents/mongodb$ mongoimport --db stock --collection phonetype --file phonetype.json
                                                --jsonArray
```

Export mongodb collection to file type jsonArray

```
// just enter direktori where collection want to place (file export name is phone.json)
f@Aspire:~/Documents/mongodb$ mongoexport --db stock --collection phone --out phone.json
```

Backup mongodb collection (use for all document in collection)

```
// just enter direktori where collection want to save
f@Aspire:~/Documents/mongodb$ mongodump --db stock --collection myphone
```

Restore mongodb collection to database (warning this operation will inserts not updates)

```
f@Aspire:~/Documents/mongodb$ mongorestore --collection myphone --db stock dump/stock/myphone.bson
```

Aggregation (sum)

```
> db.phone.find() // example data
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "merk" : "samsung", "amount" : 50 }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "merk" : "asus", "amount" : 20 }
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "merk" : "apple", "amount" : 10 }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "merk" : "sony", "amount" : 100 }
{ "_id" : ObjectId("574452fedd0538586aaa5d51"), "amount" : 35, "merk" : "samsung" }
 // search all document in collection
> db.phone.aggregate([ {$group:{_id:"$merk",num:{$sum:1}}} ])  // sum samsung group by merk
{
    "result" : [
        {
            "_id" : "sony",
            "num" : 1
        },
        {
            "_id" : "apple",
            "num" : 1
        },
        {
            "_id" : "asus",
            "num" : 1
        },
        {
            "_id" : "samsung",
            "num" : 2
        }
    ],
    "ok" : 1
}
```

Aggregation (uppercase)

```
> db.phone.aggregate ([ // conver to uppercase merk
... { $project : {phone_merk:{$toUpper:"$merk"}, _id:0, amount:1}}, // new field=phonemerk; dont show _id; show amount
... { $sort : { phone_merk:1} } // order ascending by phone_merk
... ])
{
    "result" : [
        {
            "amount" : 10,
            "phone_merk" : "APPLE"
        },
        {
            "amount" : 20,
            "phone_merk" : "ASUS"
        },
        {
            "amount" : 50,
            "phone_merk" : "SAMSUNG"
        },
        {
            "amount" : 100,
            "phone_merk" : "SONY"
        }
    ],
    "ok" : 1
}
```

Aggregation (unwind) → collect data array and sum

```
> db.phone.find()
{ "_id" : ObjectId("5743151ddf139eca8b14e86d"), "amount" : 10, "merk" : "apple" }
{ "_id" : ObjectId("57430df1df139eca8b14e869"), "amount" : 50, "color" : [  "red",  "blue",  "yellow" ], "merk" : "samsung" }
{ "_id" : ObjectId("57431442df139eca8b14e86b"), "amount" : 20, "color" : [  "black",  "blue",  "brown" ], "merk" : "asus" }
{ "_id" : ObjectId("5743e61fc553296fdfc2946b"), "amount" : 100, "color" : [  "white",  "brown",  "blue" ], "merk" : "sony" }


> db.phone.aggregate([
... { $unwind : "$color" },
... { $group : { _id:"$color", num : {$sum:1}}} ])
{
     "result" : [
          {
               "_id" : "white",
               "num" : 1
          },
          {
               "_id" : "brown",
               "num" : 2
          },
          {
               "_id" : "black",
               "num" : 1
          },
          {
               "_id" : "yellow",
               "num" : 1
          },
          {
               "_id" : "blue",
               "num" : 3
          },
          {
               "_id" : "red",
               "num" : 1
          }
     ],
     "ok" : 1
}
```

Aggregation (multiply field)

```
> db.myphone.find() // example data
{ "_id" : ObjectId("574565a7a91ef40a166fcaae"), "merk" : "APPLE", "type" : "7", "price" : 6000000, "amount" : 2 }
{ "_id" : ObjectId("574565a7a91ef40a166fcaaf"), "merk" : "SAMSUNG", "type" : "A7", "price" : 2300000, "amount" : 4 }
{ "_id" : ObjectId("574565a7a91ef40a166fcab0"), "merk" : "SAMSUNG", "type" : "J1", "price" : 1500000, "amount" : 10 }
{ "_id" : ObjectId("574565a7a91ef40a166fcab1"), "merk" : "ASUS", "type" : "Phonepad 7", "price" : 1750000, "amount" : 5 }
{ "_id" : ObjectId("574565a7a91ef40a166fcab2"), "merk" : "ASUS", "type" : "Phonepad 5", "price" : 2600000, "amount" : 6 }
{ "_id" : ObjectId("574565a7a91ef40a166fcab3"), "merk" : "SAMSUNG", "type" : "Galaxi Tab 3", "price" : 4500000, "amount" : 2 }
// we want to get total from price*amount from merk SAMSUNG
> db.myphone.aggregate ({
  $project:
 {
    _id:0, merk:1, price:1, amount:1,
     "Total": {$multiply:["$price","$amount"]}    // Total = new field
  }    })
```

Aggregation (find based on certainly field all document & then multiply two field)

```
> db.myphone.aggregate([
      { $match: {"merk":"SAMSUNG"} }, // find merk SAMSUNG in all document
      { $project: { _id:0, merk:1, price:1, amount:1,
        "Total": {$multiply:["$price","$amount"]} } } // multiply price with amount of SAMSUNG
  ])
{
    "result" : [
        {
            "merk" : "SAMSUNG",
            "price" : 2300000,
            "amount" : 4,
            "Total" : 9200000
        },
        {
            "merk" : "SAMSUNG",
            "price" : 1500000,
            "amount" : 10,
            "Total" : 15000000
        },
        {
            "merk" : "SAMSUNG",
            "price" : 4500000,
            "amount" : 2,
            "Total" : 9000000
        }
    ],
    "ok" : 1
}
```