



**National University of Computer & Emerging Sciences, Karachi**  
**Computer Science Department**  
**Fall 2021, Lab Manual – 11**



<b>Course Code: CL-2005</b>	<b>Course : Database Systems Lab</b>
<b>Instructor(s) :</b>	<b>Muhammad Nadeem, Amin Sadiq, Erum, Fizza, Mafaza, Ali Fatmi</b>

## Contents:

1. Overview of MongoDB
2. Difference in terminology of MongoDB
3. Installation of MongoDB
4. Designing Schema in MongoDB
5. Some important methods in MongoDB
6. Creating Database
7. Creating collections
8. Inserting single/multiple Documents
9. Querying, Deleting & updating of Documents
10. Logical Operations
11. Implementation of where clause

## Overview of MongoDB

MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++. This manual will give you great understanding on MongoDB concepts needed to create and deploy a highly scalable and performance-oriented database.

MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.

## Difference in Terminology of MongoDB

<b>RDBMS</b>	<b>MongoDB</b>
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by mongodb itself)
<b>Database Server and Client</b>	
Mysqld/Oracle	mongod
mysql/sqlplus	mongo

**Figure 1(Difference between RDBMS & MongoDB)**

## **Database**

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

## **Collection**

Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

## **Document**

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

# **Install MongoDB on Windows**

To perform the installation of MongoDB do following the below steps:

1. First of all, open your windows command prompt.

A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The text inside shows "Microsoft Windows [Version 10.0.19042.1348]" and "(c) Microsoft Corporation. All rights reserved." The current directory is "C:\Users\AminSadiq>".

```
Command Prompt
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AminSadiq>
```

2. Change the directory to the MongoDB directory. As I have MongoDB in C: Program Files

A screenshot of a Windows Command Prompt window showing the steps to navigate to the MongoDB directory. The title bar reads "Command Prompt". The text inside shows "Microsoft Windows [Version 10.0.19042.1348]" and "(c) Microsoft Corporation. All rights reserved." The commands entered are "cd ..", "cd ..", "cd \"Program Files\"", and "cd MongoDB". The current directory is "C:\Program Files\MongoDB>".

```
Command Prompt
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AminSadiq>cd ..
C:\Users>cd ..
C:\>cd "Program Files"
C:\Program Files>cd MongoDB
C:\Program Files\MongoDB>
```

### 3. Now further move to the sub sub directory of the mongodb i.e. bin directory

```

Command Prompt
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AminSadiq>cd ..

C:\Users>cd..

C:\>cd "Program Files"

C:\Program Files>cd MongoDB

C:\Program Files\MongoDB>cd Server

C:\Program Files\MongoDB\Server>cd 3.4

C:\Program Files\MongoDB\Server\3.4>cd bin

C:\Program Files\MongoDB\Server\3.4\bin>

```

### 4. Now run **dir** command on bin directory i.e. sub sub directory of mongoDB and match the number files as shown in the given screenshot

```

Command Prompt

C:\Program Files\MongoDB\Server\3.4\bin>dir
Volume in drive C has no label.
Volume Serial Number is 0026-F645

Directory of C:\Program Files\MongoDB\Server\3.4\bin

11/12/2021  11:16 AM  <DIR>          .
11/12/2021  11:16 AM  <DIR>          ..
10/25/2017  05:55 PM             7,403,853 bsondump.exe
11/12/2021  11:16 AM  <DIR>          data
12/19/2016  06:30 PM             2,000,384 libeay32.dll
10/25/2017  06:08 PM             11,315,712 mongo.exe
10/25/2017  06:12 PM             27,473,888 mongod.exe
10/25/2017  06:12 PM             256,471,040 mongod.pdb
10/25/2017  05:57 PM              9,517,219 mongodump.exe
10/25/2017  05:56 PM              7,668,491 mongoexport.exe
10/25/2017  05:56 PM              7,581,879 mongoimport.exe
10/25/2017  05:56 PM              7,763,191 mongoimport.exe
10/25/2017  05:58 PM              7,402,600 mongooplog.exe
10/25/2017  06:13 PM             23,440,384 mongoperf.exe
10/25/2017  05:57 PM             10,863,507 mongorestore.exe
10/25/2017  06:12 PM             13,987,328 mongos.exe
10/25/2017  06:12 PM             127,266,816 mongos.pdb
10/25/2017  05:55 PM              7,733,636 mongostat.exe
10/25/2017  05:58 PM              7,537,006 mongotop.exe
12/19/2016  06:30 PM             325,120 ssleay32.dll
               17 File(s)          535,452,054 bytes
               3 Dir(s)          91,610,632,192 bytes free

C:\Program Files\MongoDB\Server\3.4\bin>

```

### 5. Now run **mongod** command. There is an error in the given screenshot while running **mongod** command so in order to handle this problem first we need to run **mkdir \data\db** then **mongod** command

```

Command Prompt

C:\Program Files\MongoDB\Server\3.4\bin>mongod
2021-11-20T10:09:54.981+0500 I CONTROL [initandlisten] MongoDB starting : pid=16028 port=27017 dbpath=C:\data\db\ 64-bit host=DESKTOP-EIS8P32
2021-11-20T10:09:54.982+0500 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2021-11-20T10:09:54.983+0500 I CONTROL [initandlisten] db version v3.4.10
2021-11-20T10:09:54.983+0500 I CONTROL [initandlisten] git version: 078f28920cb24de0dd479b5ea6c66c644f6326e9
2021-11-20T10:09:54.983+0500 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips 22 Sep 2016
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] allocator: tcmalloc
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] modules: none
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] build environment:
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten]     distmod: 2008plus-ssl
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten]     distarch: x86_64
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten]     target_arch: x86_64
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] options: {}
2021-11-20T10:09:54.984+0500 I STORAGE [initandlisten] exception in initAndListen: 29 Data directory C:\data\db\ not found., terminating
2021-11-20T10:09:54.984+0500 I NETWORK [initandlisten] shutdown: going to close listening sockets...
2021-11-20T10:09:54.984+0500 I NETWORK [initandlisten] shutdown: going to flush diaglog...
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] now exiting
2021-11-20T10:09:54.984+0500 I CONTROL [initandlisten] shutting down with code:100

C:\Program Files\MongoDB\Server\3.4\bin>

```

## 6. Running `mkdir \data\db` command

```

C:\Program Files\MongoDB\Server\3.4\bin>mkdir \data\db

```

## 7. Now again run `mongod` command then connection is now waiting to connect as shown in the screenshot

```

2021-11-20T11:28:38.111+0500 I CONTROL [initandlisten] MongoDB starting : pid=9360 port=27017 dbpath=C:\data\db\ 64-bit host=DESKTOP-EIS8P32
2021-11-20T11:28:38.112+0500 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] db version v3.4.10
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] git version: 078f28920cb24de0dd479b5ea6c66c644f6326e9
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips 22 Sep 2016
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] allocator: tcmalloc
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] modules: none
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] build environment:
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] distmod: 2008plus-ssl
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] distarch: x86_64
2021-11-20T11:28:38.113+0500 I CONTROL [initandlisten] target arch: x86_64
2021-11-20T11:28:38.114+0500 I CONTROL [initandlisten] options: {}
2021-11-20T11:28:38.114+0500 I - [initandlisten] Detected data files in C:\data\db\ created by the 'wiredTiger' storage engine, so setting the active storage engine to 'wiredTiger'.
2021-11-20T11:28:38.114+0500 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=3531M,session_max=20000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000),checkpoint=(wait=60,log_size=2GB),statistics_log=(wait=0),
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]
2021-11-20T11:28:38.482+0500 W FTDC [initandlisten] Failed to initialize Performance Counters for FTDC: WindowsPdhError: PdhExpandCounterPathW failed with 'The specified object was not found on the computer.' for counter '\Memory\Available Bytes'
2021-11-20T11:28:38.483+0500 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory 'C:\data\db\diagnostic.data'
2021-11-20T11:28:38.485+0500 I NETWORK [thread1] waiting for connections on port 27017

```

## 8. Now we are establishing the `mongod` and `mongo` connection in the given screenshot

```

2021-11-20T11:28:38.485+0500 I NETWORK [thread1] waiting for connections on port 27017
2021-11-20T11:33:11.476+0500 I NETWORK [thread1] connection accepted from 127.0.0.1:1338 #1 (1 connection now open)
2021-11-20T11:33:11.477+0500 I NETWORK [conn1] received client metadata from 127.0.0.1:1338 conn1: { application: { name: "MongoDB Shell" }, driver: { name: "MongoDB Internal Client", version: "3.4.10" }, os: { type: "Windows", name: "Microsoft Windows 8", architecture: "x86_64", version: "6.2 (build 9200)" } }

```

```

C:\Program Files\MongoDB\Server\3.4\bin>mongo
MongoDB shell version v3.4.10
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.10
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
http://docs.mongodb.org/
Questions? Try the support group
http://groups.google.com/group/mongodb-user
Server has startup warnings:
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]

```

## 9. After successful connection we will check the already made Database

```

Command Prompt - mongo
C:\Program Files\MongoDB\Server\3.4\bin>mongo
MongoDB shell version v3.4.10
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.10
Server has startup warnings:
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten] **           Read and write access to data and configuration is unrestricted.
2021-11-20T11:28:38.331+0500 I CONTROL [initandlisten]
> db
test
> db.test.save
function (obj, opts) {
  if (obj == null)
    throw Error("can't save a null");

  if (typeof(obj) == "number" || typeof(obj) == "string")
    throw Error("can't save a number or string");

  if (typeof(obj._id) == "undefined") {
    obj._id = new ObjectId();
    return this.insert(obj, opts);
  } else {
    return this.update({_id: obj._id}, obj, Object.merge({upsert: true}, opts));
  }
}
> db.test.save({a:1})
WriteResult({ "nInserted" : 1 })
> db.test.find()
{ "_id" : ObjectId("619897dada279eeb82cd6ed3"), "a" : 1 }
>

```

## 10. Now running **db.help()** to see different helping functions

```

Command Prompt - mongo
> db.help()
DB methods:
  db.adminCommand(nameOrDocument) - switches to 'admin' db, and runs command [ just calls db.runCommand(...) ]
  db.auth(username, password)
  db.cloneDatabase(fromhost)
  db.commandHelp(name) returns the help for the command
  db.copyDatabase(fromdb, todb, fromhost)
  db.createCollection(name, { size : ..., capped : ..., max : ... })
  db.createView(name, viewOn, [ { $operator: {...}}, ... ], { viewOptions })
  db.createUser(userDocument)
  db.currentOp() displays currently executing operations in the db
  db.dropDatabase()
  db.eval() - deprecated
  db.fsyncLock() flush data to disk and lock server for backups
  db.fsyncUnlock() unlocks server following a db.fsyncLock()
  db.getCollection(cname) same as db['cname'] or db.cname
  db.getCollectionInfos([filter]) - returns a list that contains the names and options of the db's collections
  db.getCollectionNames()
  db.getLastErrorMessage() - just returns the err msg string
  db.getLastErrorMessageObj() - return full status object
  db.getLogComponents()
  db.getMongo() get the server connection object
  db.getMongo().setSlaveOk() allow queries on a replication slave server
  db.getName()
  db.getPrevError()
  db.getProfilingLevel() - deprecated
  db.getProfilingStatus() - returns if profiling is on and slow threshold
  db.getReplicationInfo()
  db.getSiblingDB(name) get the db at the same server as this one
  db.getWriteConcern() - returns the write concern used for any operations on this db, inherited from server object if set
  db.hostInfo() get details about the server's host
  db.isMaster() check replica primary status
  db.killOp(opid) kills the current operation in the db
  db.listCommands() lists all the db commands
  db.loadServerScripts() loads all the scripts in db.system.js
  db.logout()

```

## 11. Now run **db.stats()** to see the statistics of the database

```

Command Prompt - mongo
> db.stats()
{
  "db" : "test",
  "collections" : 1,
  "views" : 0,
  "objects" : 1,
  "avgObjSize" : 33,
  "dataSize" : 33,
  "storageSize" : 16384,
  "numExtents" : 0,
  "indexes" : 1,
  "indexSize" : 16384,
  "ok" : 1
}
>

```

## Some Consideration While Designing Schema in MongoDB

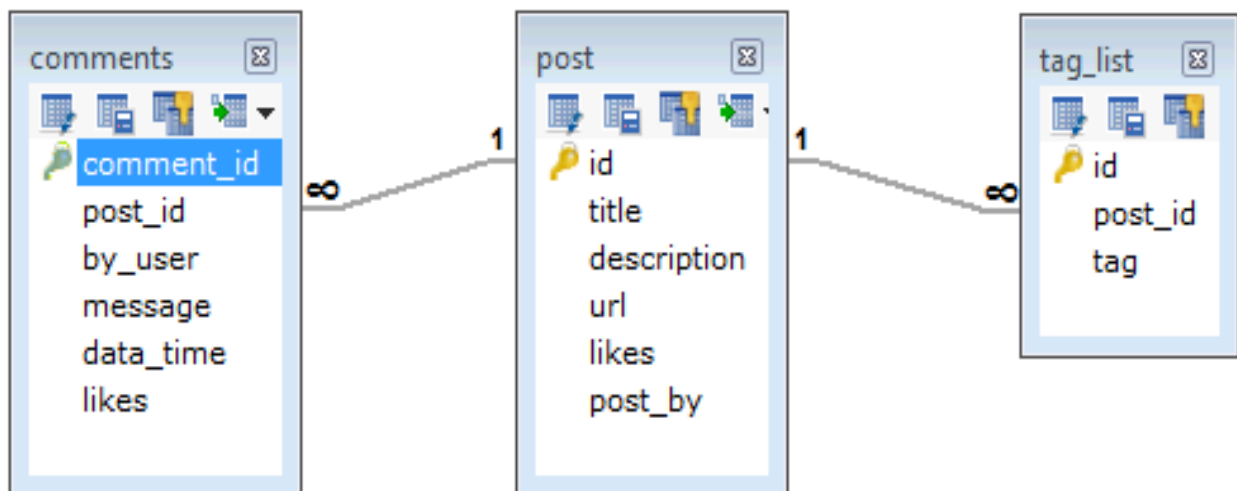
1. Design your schema according to user requirements.
2. Combine objects into one document if you will use them together. Otherwise separate them (but make sure there should not be need of joins).
3. Duplicate the data (but limited) because disk space is cheap as compare to compute time.
4. Do joins while write, not on read.
5. Optimize your schema for most frequent use cases.
6. Do complex aggregation in the schema

### Example

Suppose a client needs a database design for his blog/website and see the differences between RDBMS and MongoDB schema design. Website has the following requirements.

1. Every post has the unique title, description and url.
2. Every post can have one or more tags.
3. Every post has the name of its publisher and total number of likes.
4. Every post has comments given by users along with their name, message, data-time and likes.
5. On each post, there can be zero or more comments

In RDBMS schema, design for above requirements will have minimum three tables.



While in MongoDB schema, design will have one collection post and the following structure:

```
{
  _id: POST_ID,
  title: TITLE_OF_POST,
  description: POST_DESCRIPTION,
  by: POST_BY,
  url: URL_OF_POST,
  tags: [TAG1, TAG2, TAG3],
  likes: TOTAL_LIKES,
  comments: [
    {
      user: 'COMMENT_BY',
      message: TEXT,
      dateCreated: DATE_TIME,
      like: LIKES
    },
    {
      user: 'COMMENT_BY',
      message: TEXT,
      dateCreated: DATE_TIME,
      like: LIKES
    }
  ]
}
```

So while showing the data, in RDBMS you need to join three tables and in MongoDB, data will be shown from one collection only.

## MongoDB – Create Database

MongoDB **use DATABASE\_NAME** is used to create database. The command will create a new database if it doesn't exist, otherwise it will return the existing database.

### Syntax

Basic syntax of **use DATABASE** statement is as follows – use DATABASE\_NAME



```
Command Prompt - mongo
> use myfirstdb
switched to db myfirstdb
>
```

### Command:

>show dbs

If you want to check your databases list, use the command **show dbs**.



```
Command Prompt - mongo
> use myfirstdb
switched to db myfirstdb
> show dbs
admin  0.000GB
local  0.000GB
test   0.000GB
>
```

Your created database (myfirstdb) is not present in list. To display database, you need to insert at least one document into it.

```
> db.myfirstdb.insert({"name":"Amin Sadiq"})
WriteResult({"nInserted" : 1 })
```



```

Command Prompt - mongo
> use myfirstdb
switched to db myfirstdb
> show dbs
admin    0.000GB
local    0.000GB
test     0.000GB
> db.myfirstdb.insert({"name":"Amin Sadiq"})
WriteResult({"nInserted" : 1 })
> show dbs
admin    0.000GB
local    0.000GB
myfirstdb 0.000GB
test     0.000GB
>

```

## The dropDatabase() Method

MongoDB `db.dropDatabase()` command is used to drop an existing database

### Syntax

Basic syntax of `dropDatabase()` command is as follows –

```
> db.dropDatabase()
```

This will delete the selected database. If you have not selected any database, then it will delete default 'test' database.

### Example

If you want to delete new database <mydb>, then `dropDatabase()` command would be as follows –

```

>use myfirstdb
// Switched to db mydb
>db.dropDatabase()
{ "dropped" : "myfirstdb", "ok" : 1 }

```

Now check list of databases.

```
>show dbs
```



```

Command Prompt - mongo
> use myfirstdb
switched to db myfirstdb
> show dbs
admin    0.000GB
local    0.000GB
test     0.000GB
> db.myfirstdb.insert({"name":"Amin Sadiq"})
WriteResult({"nInserted" : 1 })
> show dbs
admin    0.000GB
local    0.000GB
myfirstdb 0.000GB
test     0.000GB
> db.dropDatabase()
{ "dropped" : "myfirstdb", "ok" : 1 }
> show dbs
admin    0.000GB
local    0.000GB
test     0.000GB
>

```

## The createCollection() Method

MongoDB `db.createCollection(name, options)` is used to create collection.

### Syntax

Basic syntax of `createCollection()` command is as follows –

```
db.createCollection(name, options)
```

In the command, name is name of collection to be created. Options (Optional Parameter) is a document and is used to specify configuration of collection.



```
Command Prompt - mongo
> use myfirstdb
switched to db myfirstdb
> show dbs
admin    0.000GB
local    0.000GB
test     0.000GB
> db.movie.insert({"name":"Amin Sadiq"})
WriteResult({"nInserted" : 1 })
> show dbs
admin    0.000GB
local    0.000GB
myfirstdb 0.000GB
test     0.000GB
> db.createcollection("myCollection")
2021-11-20T12:16:51.393+0500 E QUERY    [thread1] TypeError: db.createcollection is not a function :
@(:shell):1:1
> db.createCollection("myCollection")
{ "ok" : 1 }
> show collections
movie
myCollection
>
```

## MongoDB – Drop Collection

### The drop() Method

MongoDB's **db.collection.drop()** is used to drop a collection from the database

### Syntax

Basic syntax of **drop()** command is as follows –

**db.COLLECTION\_NAME.drop()**

```
> show collections
movie
myCollection
> db.movie.drop()
true
> show dbs
admin    0.000GB
local    0.000GB
myfirstdb 0.000GB
test     0.000GB
> show collections
myCollection
>
```

## MongoDB – Insert Document

### The insert() Method

To insert data into MongoDB collection, you need to use MongoDB's insert() or save() method

### Syntax

The basic syntax of insert() command is as follows –

**>db.COLLECTION\_NAME.insert(document)**

## Example

```
> db.mycollection.insert({
  title: 'MongoDB Overview',
  description: 'MongoDB is no sql database',
  by: 'Amin Sadiq',
  url: 'http://www.gmail.com',
  tags: ['mongodb', 'database', 'NoSQL'],
  likes: 100
})
```



```

Command Prompt - mongo
> db.mycollection.insert({
... title: 'MongoDB Overview',
... description: 'MongoDB is no sql database',
... by: 'Amin Sadiq',
... url: 'http://www.gmail.com',
... tags: ['mongodb', 'database', 'NoSQL'],
... likes: 100
... })
WriteResult({ "nInserted" : 1 })

```

Notepad: Here **mycollection** is our collection name, as created in the previous slide. If the collection doesn't exist in the database, then MongoDB will create this collection and then insert a document into it.

\_id parameter

In the inserted document, if we don't specify the \_id parameter, then MongoDB assigns a unique ObjectId for this document.

## Insert Document multiple document

To insert multiple documents in a single query, you can pass an array of documents in insert() command.

```

db.mycollection.insert([
  {
    title: 'MongoDB Overview',
    description: 'MongoDB is no sql database',
    by: 'Amin Sadiq',
    url: 'http://www.gmail.com',
    tags: ['mongodb', 'database', 'NoSQL'],
    likes: 100
  },

  {
    title: 'NoSQL Database',
    description: "NoSQL database doesn't have tables",
    by: 'Ali Shah Fatmi',
    url: 'http://www.gmail.com',
    tags: ['mongodb', 'database', 'NoSQL'],
    likes: 20,
    comments: [
      {
        user: 'user1',
        message: 'My first comment',
        dateCreated: new Date(2021,11,10,2,35),
        like: 0
      }
    ]
  }
])

```

```

Command Prompt - mongo
> db.mycollection.insert([
...   {
...     title: 'MongoDB Overview',
...     description: 'MongoDB is no sql database',
...     by: 'Amin Sadiq',
...     url: 'http://www.gmail.com',
...     tags: ['mongodb', 'database', 'NoSQL'],
...     likes: 100
...   },
...   {
...     title: 'NoSQL Database',
...     description: 'NoSQL database doesn't have tables',
...     by: 'Ali Shah Fatmi',
...     url: 'http://www.gmail.com',
...     tags: ['mongodb', 'database', 'NoSQL'],
...     likes: 20,
...     comments: [
...       {
...         user: 'user1',
...         message: 'My first comment',
...         dateCreated: new Date(2021,11,10,2,35),
...         like: 0
...       }
...     ]
...   }
... ])
BulkWriteResult({
  "writeErrors" : [ ],
  "writeConcernErrors" : [ ],
  "nInserted" : 2,
  "nUpserted" : 0,
  "nMatched" : 0,
  "nModified" : 0,
  "nRemoved" : 0,
  "upserted" : [ ]
})

```

# MongoDB – Query Document

## The find() Method

To query data from MongoDB collection, you need to use MongoDB's find() method.

### Syntax

- The basic syntax of find() method is as follows –
- >db.COLLECTION\_NAME.find()

```

Command Prompt - mongo
> db.mycollection.find()
{ "_id" : ObjectId("6198f3f41388304875b893e7"), "title" : "MongoDB Overview", "description" : "MongoDB is no sql database", "by" : "Amin Sadiq", "url" : "http://www.gmail.com", "tags" : [ "mongodb", "database", "NoSQL" ], "likes" : 100 }
{ "_id" : ObjectId("6198f3f41388304875b893e8"), "title" : "NoSQL Database", "description" : "NoSQL database doesn't have tables", "by" : "Ali Shah Fatmi", "url" : "http://www.gmail.com", "tags" : [ "mongodb", "database", "NoSQL" ], "likes" : 20, "comments" : [ { "user" : "user1", "message" : "My first comment", "dateCreated" : ISODate("2021-12-09T21:35:00Z"), "like" : 0 } ] }

```

Note: **find()** method will display all the documents in a non-structured way.

## The pretty() Method

To display the results in a formatted way, you can use pretty() method.

### Syntax

- >db.mycol.find().pretty()

```

Command Prompt - mongo
> db.mycollection.find().pretty()
{
  " _id" : ObjectId("6198f3f41388304875b893e7"),
  "title" : "MongoDB Overview",
  "description" : "MongoDB is no sql database",
  "by" : "Amin Sadiq",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 100
}
{
  " _id" : ObjectId("6198f3f41388304875b893e8"),
  "title" : "NoSQL Database",
  "description" : "NoSQL database doesn't have tables",
  "by" : "Ali Shah Fatmi",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 20,
  "comments" : [
    {
      "user" : "user1",
      "message" : "My first comment",
      "dateCreated" : ISODate("2021-12-09T21:35:00Z"),
      "like" : 0
    }
  ]
}

```

# RDBMS Where Clause Equivalents in MongoDB

Operation	Syntax	Example	RDBMS Equivalent
<b>Equality</b>	{<key>:<value>}	db.mycollection.find({"by":"Amin Sadiq"}).pretty()	where by = 'Amin Sadiq'
<b>Less Than</b>	{<key>:{\$lt:<value>}}	db.mycollection.find({"likes":{\$lt:50}}).pretty()	where likes < 50
<b>Less Than Equals</b>	{<key>:{\$lte:<value>}}	db.mycollection.find({"likes":{\$lte:50}}).pretty()	where likes <= 50
<b>Greater Than</b>	{<key>:{\$gt:<value>}}	db.mycollection.find({"likes":{\$gt:50}}).pretty()	where likes > 50
<b>Greater Than Equals</b>	{<key>:{\$gte:<value>}}	db.mycollection.find({"likes":{\$gte:50}}).pretty()	where likes >= 50
<b>Not Equals</b>	{<key>:{\$ne:<value>}}	db.mycollection.find({"likes":{\$ne:50}}).pretty()	where likes != 50

## AND in MongoDB

### Syntax

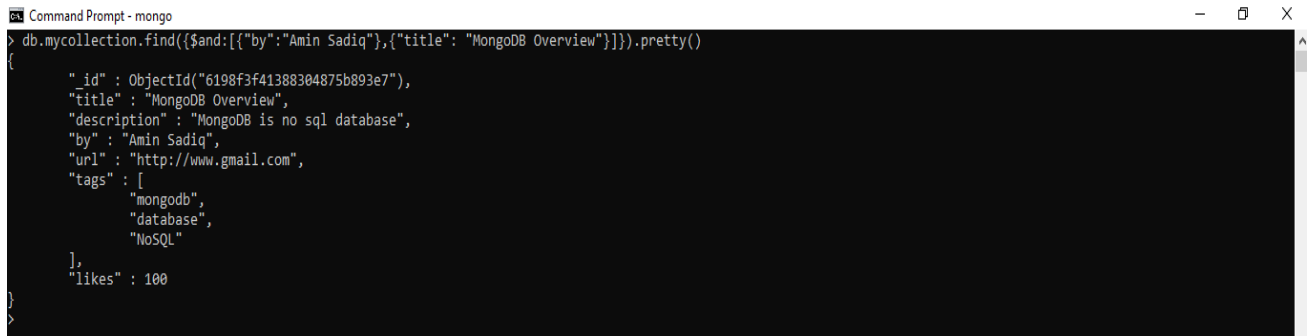
In the find() method, if you pass multiple keys by separating them by ',' then MongoDB treats it as AND condition. Following is the basic syntax of AND –

```
db.mycol.find(
{
  $and: [
    {key1: value1}, {key2:value2}
  ]
})
).pretty()
```

### Example

Following example will show all the tutorials written by 'Amin Sadiq' and whose title is 'MongoDB Overview'.

```
db.mycollection.find({$and:[{"by":"Amin Sadiq"},"title": "MongoDB Overview"]}).pretty()
```



```
Command Prompt - mongo
> db.mycollection.find({$and:[{"by":"Amin Sadiq"},"title": "MongoDB Overview"]}).pretty()
{
  "_id" : ObjectId("6198f3f41388304875b893e7"),
  "title" : "MongoDB Overview",
  "description" : "MongoDB is no sql database",
  "by" : "Amin Sadiq",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 100
}
```

## OR in MongoDB

### Syntax

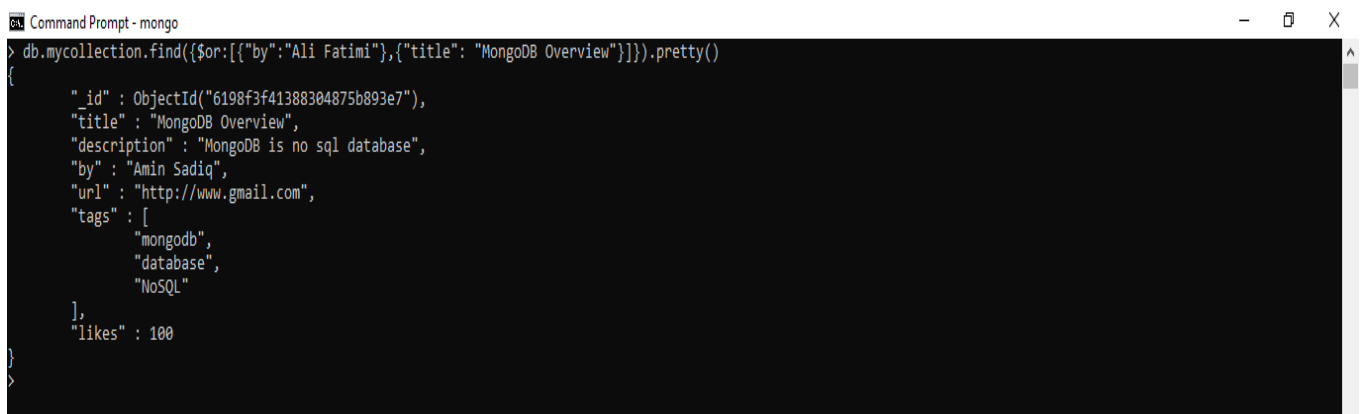
- To query documents based on the OR condition, you need to use \$or keyword. Following is the basic syntax of OR –

```
db.mycol.find(
{
  $or: [
    {key1: value1}, {key2:value2}
  ]
}
).pretty()
```

### Example

Following example will show all the tutorials written by 'Amin Sadiq' or whose title is 'MongoDB Overview'.

```
db.mycollection.find({$or:[{"by":"Ali Fatimi"},"title": "MongoDB Overview"]}).pretty()
```

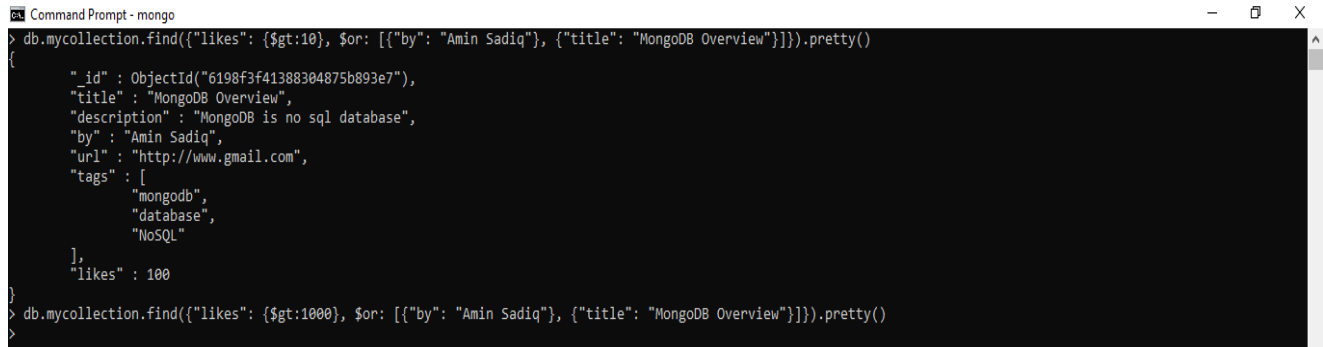


```
Command Prompt - mongo
> db.mycollection.find({$or:[{"by":"Ali Fatimi"},"title": "MongoDB Overview"]}).pretty()
{
  "_id" : ObjectId("6198f3f41388304875b893e7"),
  "title" : "MongoDB Overview",
  "description" : "MongoDB is no sql database",
  "by" : "Amin Sadiq",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 100
}
```

## Using AND & OR together in MongoDB

The following example will show the documents that have likes greater than 10 and whose title is either 'MongoDB Overview' or by is 'Amin Sadiq'. Equivalent SQL where clause is 'where likes>10 AND (by = 'Amin Sadiq' OR title = 'MongoDB Overview')'

```
db.mycollection.find({"likes": {$gt:10}, $or: [{"by": "Amin Sadiq"}, {"title": "MongoDB Overview"}]}).pretty()
```



```
Command Prompt - mongo
> db.mycollection.find({"likes": {$gt:10}, $or: [{"by": "Amin Sadiq"}, {"title": "MongoDB Overview"}]}).pretty()
{
  "_id" : ObjectId("6198f3f41388304875b893e7"),
  "title" : "MongoDB Overview",
  "description" : "MongoDB is no sql database",
  "by" : "Amin Sadiq",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 100
}
> db.mycollection.find({"likes": {$gt:1000}, $or: [{"by": "Amin Sadiq"}, {"title": "MongoDB Overview"}]}).pretty()
>
```

## MongoDB – Update Document

### MongoDB Update() Method

The update() method updates the values in the existing document

#### Syntax

- The basic syntax of update() method is as follows –

```
db.COLLECTION_NAME.update(SELECTION_CRITERIA, UPDATED_DATA)
```

#### Example

Consider the mycollection collection has the following data.

```
{"_id" : ObjectId(5983548781331adf45ec5), "title":"MongoDB Overview"}
{"_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{"_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}
```

Following example will set the new title 'New MongoDB Tutorial' of the documents whose title is 'MongoDB Overview'.

```
db.mycollection.update({'title':'MongoDB Overview'},{$set: {'title':'New MongoDB Tutorial'}})
```

```

Command Prompt - mongo
> db.mycollection.update({'title':'MongoDB Overview'},{$set:{'title':'New MongoDB Tutorial'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.mycollection.find().pretty()
{
  "_id" : ObjectId("6198f3f41388304875b893e7"),
  "title" : "New MongoDB Tutorial",
  "description" : "MongoDB is no sql database",
  "by" : "Amin Sadiq",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 100
}
{
  "_id" : ObjectId("6198f3f41388304875b893e8"),
  "title" : "NoSQL Database",
  "description" : "NoSQL database doesn't have tables",
  "by" : "Ali Shah Fatmi",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 20,
  "comments" : [
    {
      "user" : "user1",
      "message" : "My first comment",
      "dateCreated" : ISODate("2021-12-09T21:35:00Z"),
      "like" : 0
    }
  ]
}

```

## MongoDB – Delete Document

### MongoDB remove() Method

- MongoDB's remove() method is used to remove a document from the collection. remove() method accepts two parameters. One is deletion criteria and second is justOne flag.
- deletion criteria – (Optional) deletion criteria according to documents will be removed.
- justOne – (Optional) if set to true or 1, then remove only one document.

### Syntax

Basic syntax of remove() method is as follows –

```
db.COLLECTION_NAME.remove(DELETION_CRITERIA)
```

Example

Let's consider the mycollection collection has the following data.

```

{"_id" : ObjectId(5983548781331adf45ec5), "title":"New MongoDB Tutorial"}
{"_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{"_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}

```

Following example will remove all the documents whose title is 'MongoDB Overview'.

```
db.mycollection.remove({'title':'New MongoDB Tutorial'})
```

```
Command Prompt - mongo
> db.mycollection.remove({'title':'New MongoDB Tutorial'})
WriteResult({ "nRemoved" : 1 })
> db.mycollection.find().pretty()
{
  "_id" : ObjectId("6198f3f41388304875b893e8"),
  "title" : "NoSQL Database",
  "description" : "NoSQL database doesn't have tables",
  "by" : "Ali Shah Fatmi",
  "url" : "http://www.gmail.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL"
  ],
  "likes" : 20,
  "comments" : [
    {
      "user" : "user1",
      "message" : "My first comment",
      "dateCreated" : ISODate("2021-12-09T21:35:00Z"),
      "like" : 0
    }
  ]
}
```