**Architecture of Mongo**

Once we installed MongoDB

Mongod - DataBase Server Engine

Mongo - MongoShell or command Prompt

Mongos - Routing Service for MongoDB Sharding

**Request Processing at MongoDB**

Application

Mongo Shell

(CUI)

Mongo Booster

(GUI)

**DataFormat**

{

Key: value,

Key : value

}

**Mongod Process**

--dbpath

Runs on Port - 27017

**MongoDB Server**

D:/data/db

{

Id: value

---

}

BSON Format

Documents

The Applications or from IDE the data will sent to the Mongod which as Database Server in JSON Format

The JSON again converted into BSON Format and Saved as a document at the Data Store Folder

BSON Provides Some Special Features Like Data types for JSON Data

BSON Provides the Facility to Build Indexes So that Mongo Can Easily traverse the Data

BSON is Light weight and allows the MongoDB to enter inside the BSON so that Performance will increase

Mongo Stores a record in the format of a Document

Each Document will have a unique ID 12 bit Code

Each Document is of max 16 MB Size . Will helps maintain efficient bandwidth in the network . if need a document More than 16MB , we should use GRIDFS Api

**Write And Read in Mongo**

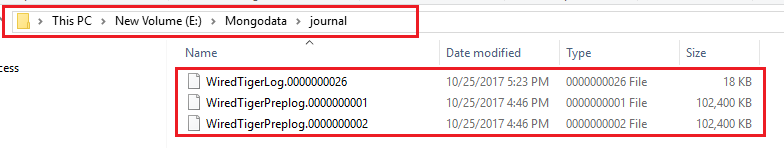
Writing in Mongo is Reliable and Faster

For Write Purpose Mongo Uses In-Memory . Maintains the write ahead log , so that can recover the data even on Sudden Exits

MongoDB Storage Engine create one Journal for each write Operation from each Client to perform the write operation

Journal Includes the Write Operations caused by the Initial Write

The Physical Location for the Journal is **dbpath/Journal**



MongoDB Storage Engine asks the OS to Sync the buffered Journal Data into the physical Disk under some conditions

* For the interval of 60Secs or storage of 2GB , ie after Every 60 secs journal Log will be written to the Disk or after the journal size become 2gb
* Generally the Size of the Journal is 100MB , It will create new Journal if exceeds 100MB . When a new Journal File is created , Mongo Storage Engine Syncs the Previous Journal Data to Disk
* If the Write Operation Include an acknowledgement , by setting j:true then Journal Data will be forced to sync to the Disk

**DB**

(**Physical Disk**)

**Storage**

**Engine**

Requests

**OS**

**Private**

**View**

**Shared**

**View**

**(Flushed by OS)**

Client Performs Write Operation

Mongod

60 Secs

Exceeds 2GB

200 milli Secs

**Journal**

(In Memory)

upto 100 MB for Each Jounal File.

Only Operations

Using journal Log may Reduce the write Performance when writing the data . But data is can recovered to the last check point .

Maximum data loss will be for 60 Secs as it is the interval is of Pushing the data from Journal to Physical disk , even we can reduce the interval by setting the interval

Steps

1. Once the write Operation requested Mongod Place the data into the Private View
2. From the Private View it will be pushed to Journal. Once the data Available at Journal , even I Mongo Exits Sudden , we can get the data from Journal for the last CheckPoint
3. Form the Journal Mogod asks the OS to push it to Shared view after every 60secs or once reached 2GB
4. From the Shared View The Data Will be Saved to the DB at a high speed
5. And last the shared will be mapped with private so that to avoid repeat writes

While reading

When you start up mongod, it maps your data files to a shared view. Basically, the operating system says: “Okay, your data file is 2,000 bytes on disk. I’ll map that to memory address 1,000,000-1,002,000. So, if you read the memory at memory address 1,000,042, you’ll be getting the 42nd byte of the file.