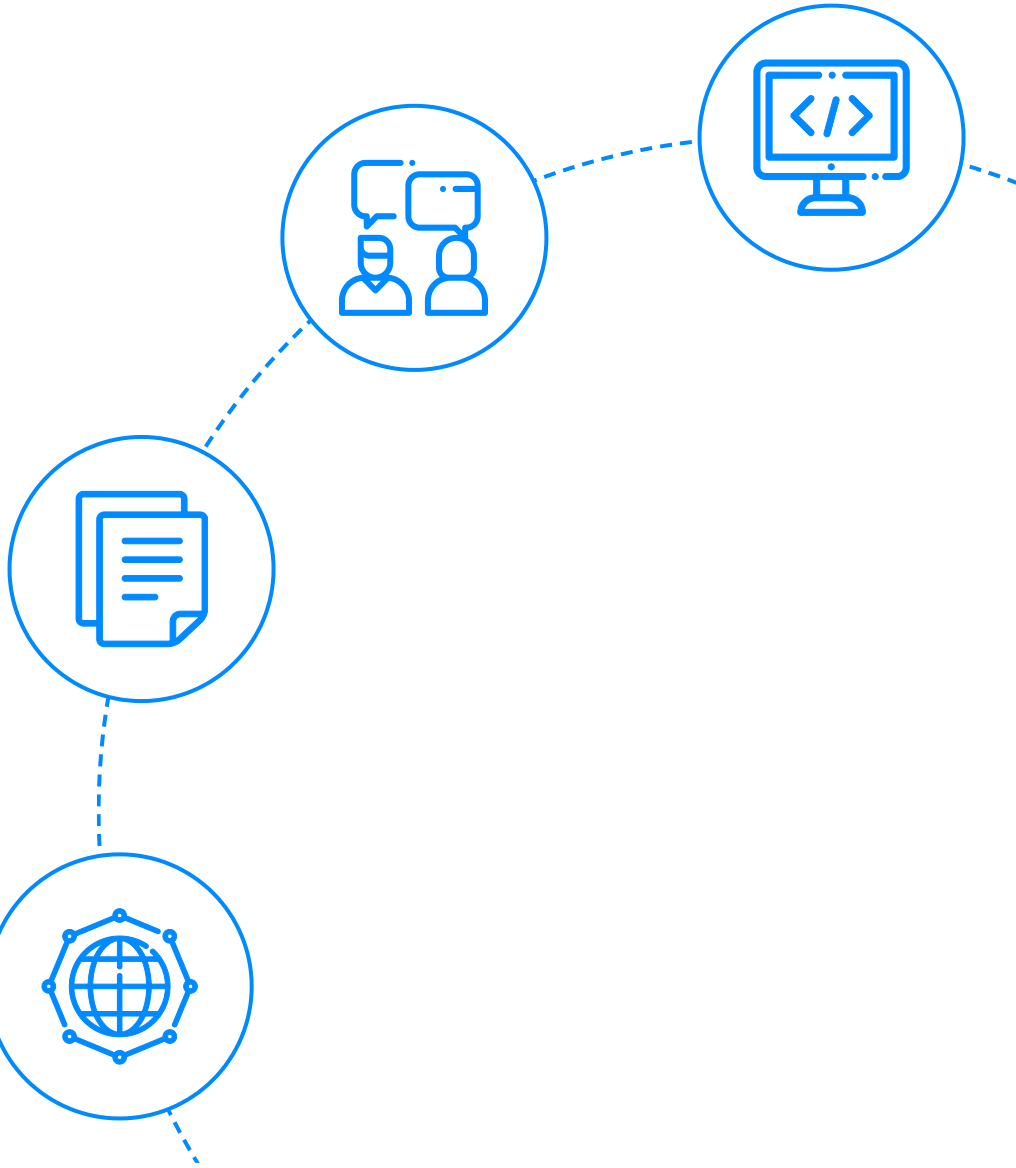
InterviewBit

MongoDB Cheat Sheet



To view the live version of the page, click here.

Contents

MongoDB Tutorial: Beginners and Experienced

MongoDB Tutorial: Beginners and Experienced

(.....Continued)

21. Databases and Collections
22. Some other Handy commands

Let's get Started

Introduction

MongoDB is a data management platform that enables quick and easy query development and deployment of online, real-time data applications. It is a distributed, not-backed store that runs on a collection of servers and uses a JSON-like data model.

MongoDB Replica Management allows you to easily and cost-effectively scale your MongoDB architecture. MongoDB provides a rich set of analytical tools for data profiling, load analysis, and monitoring. It can be used for a variety of purposes including data mining, Big Data, and online analytical processing.

Internet and enterprise application developers that require flexibility and scaling efficiently may consider using MongoDB. MongoDB is particularly suited to developers of varied types who are creating scalable applications using agile approaches.

Advantages and Disadvantages of MongoDB

MongoDB has both pros and cons just like other NoSQL databases.

Pros:

- Any type of data can be stored in MongoDB, which gives users the flexibility to create as many fields in a document as they desire.
- Documents map to native data types in many programming languages, which provides a means of adding to data. Sharding, which involves dividing data across a cluster of machines, is also achieved by this.
- MongoDB includes its own file system, similar to the Hadoop Distributed File System (HDFS), called GridFS. The file system is primarily used to store files that exceed MongoDB's 16 MB per document BSON size limit.
- MongoDB is also compatible with Spark, Hadoop, and other data processing frameworks like SQL.

Cons:

- When a MongoDB master node goes down, another node will automatically become the new master. Despite the fact that it promises continuity, the automatic failover strategy is not instantaneous - it may take up to a minute. In contrast, the Cassandra NoSQL database supports multiple master nodes, so that if a master goes down, another one is ready to run a highly available database infrastructure.
- Although MongoDB's single master node restricts how fast data can be written to the database, it also limits how much data can be written. Because of this, data writes must be recorded on the master, and new information cannot be added to the database quickly.
- MongoDB doesn't provide full referential integrity using foreign-key constraints, which could affect data consistency.
- User authentication isn't enabled by default in MongoDB databases. Because of this, there is a default setting that blocks networked connections to databases if they've not been configured by a database administrator.
- There have also been instances of ransomware attacks that forced the setting to be turned on by the database administrator.

Features of MongoDB

1. Replication: The MongoDB replica set feature is known for providing high availability. Two or more copies of data constitute a replica set. A replica-set acts as a primary or a secondary replica. Secondary replicas keep a copy of the data of the primary, preserving it in an orderly manner, as part of a replicated MongoDB system. Whenever a primary replica crashes, the replica set automatically determines which secondary should become the primary and conducts an election if necessary. Secondary replicas may additionally serve read operations, but the data is only eventually consistent by default. To resolve the election of the new primary, three standalone servers must be added as secondary servers.

2. Indexing: A MongoDB field can be indexed with primary and secondary indices or indexes. A MongoDB index stores a small portion of the data set in a form that is convenient to traverse. The index stores the value of a particular field, or set of fields, ordered by their value. In MongoDB, indexes assist in efficiently resolving queries by storing a small portion of the data set in a convenient form. A MongoDB index is similar to a typical relational database index.

3. File storage: GridFS, which uses MongoDB as a file system, can be used to balance and replicate data across multiple machines. A file can be stored in MongoDB as a grid file system. It has features similar to a file system such as load balancing and data replication.

4. Aggregation: The aggregation pipeline, the map-reduce function, and single-purpose aggregation methods are available in MongoDB. According to MongoDB's documentation, the Aggregation Pipeline provides better performance for most aggregation operations over map-reduce. With the aggregation framework, users can obtain the kind of results for which the SQL GROUP BY clause is used. The aggregation framework includes $lookup and standard deviation like statistical operators.

5. Sharding: Sharding is the splitting up of data among machines. To permit this, we refer to it as "partitioning" or "sharding." We may store more data and handle more load without upgrading our machines, by dividing data across them. MongoDB's sharding allows you to split up a collection among many machines (shards), allowing it to grow beyond resource limitations.

The following cheat sheet is filled with some handy tips and commands for quick reference:

MongoDB Tutorial: Beginners and Experienced

1. To show all databases

```
show dbs
```

2. To show current database

```
db
```

3. Switch or create database

```
use acme
```

4. Drop a database

```
db.dropDatabase()
```

5. Inserting Document

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

```
db.docx.insert({name:'Enterprise',operator:'Star',type:'Explorer',class:'Universe',crew
db.docx.insert({name:'Prometheus',operator:'Star',class:'Prometheus',crew:40,codes:[10,
```

6. Insert Row

The MongoDB shell provides the following methods to insert rows:

```
db.docx.insert({
 title: 'Post Five',
 body: 'Body of post five,
 category: 'Information',
 tags: ['Information', 'events'],
 user: {
   name: 'David',
   status: 'author'
```

7. Insert Multiple Row

The MongoDB shell provides the following methods to insert multiple rows:

```
db.docx.insertMany([
{
  title: 'Post six,
  body: 'Body of post six,
  category: 'Science',
  date: Date()
},
{
  title: 'Post seven',
  body: 'Body of post seven',
  category: 'Information',
  date: Date()
},
{
  title: 'Post eight',
  body: 'Body of post eight',
  category: 'Sports',
  date: Date()
}
])
```

8. Finding document

The MongoDB shell provides the following methods to find documents:

S. No.	Commands	Description
1.	`db.docx.findOne()`	Finds one random document.
2.	`db.docx.find().prettyPrint()`	Finds all documents.
3.	`db.docx.find({}, {name:true, _id:false})`	Displays only the names of the document Docx.
4.	`db.docx.find({}, {name:true, _id:false})`	Can find one document by attribute among many documents.

9. Finding Documents using Operators

The MongoDB shell provides the following methods to find documents using operators:

Operator	Description	Commands
$gt	greater than	`db.docx.find({class: {$gt:'T'}`
$gte	greater than equals	`db.docx.find({class: {$gt:'T'}`
$lt	lesser than	`db.docx.find({class: {$lt:'T'}`
$lte	lesser than equals	`db.docx.find({class: {$lte:'T'}`
$exists	does an attribute exist or not	`db.docx.find({class: {$gt:'T'}`
$regex	Matching pattern in pearl-style	`db.docx.find({name: {$regex:'^USS\\sE'}})`
$type	search by type of an element	`db.docx.find({name : {$type:4}})`

10. Find one row

To find a row, use-

```
db.docx.findOne({ category: 'Science' })
```

11. Delete a document

deleteOne and deleteMany can be used for this purpose. Both of these methods take a filter document as their first parameter.

```
db.docx.deleteOne({"_id" : 6})
```

12. Delete row

To delete a row, use:

```
db.docx.remove({ title: 'Post six' })
```

13. Sort rows

The MongoDB shell provides the following methods to sort rows:

```
# asc
db.docx.find().sort({ title: 5 }).pretty()
```

```
# desc
db.docx.find().sort({ title: -5}).pretty()
```

14. Count Rows

To count number of rows, use:

```
db.docx.find().count()
```

15. Limit rows

To limit the number of rows, use:

```
db.docx.find().limit(5).pretty()
```

16. Update one document

A document stored in the database can be changed using one of several update methods: updateOne, updateMany, and replaceOne.

updateOne and updateMany take a filter document as their first parameter and a modifier document, which describes changes to make, as the second parameter.

Command: `db.docx.updateOne({"_id": 2}, {$set: {"title": 'revised title'}})`

To update multiple document, use:

```
db.docx.update({"category": "Information"}, {$set: {"category": 'Sports'}}  )
```

18. Update Row

The MongoDB shell provides the following method to update row:

```
db.docx.update({ title: 'Post three' },
{
 title: 'Post three,
 body: 'New body for post 3',
 date: Date()
},
{
 upsert: true
})
```

19. Indexes

- **List Indexes:** It can be done by using: `db.docx.getIndexes()`
- **Create Index**

```
db.docx.createIndex({"name": 2})                // single field index
db.docx.createIndex({"name": 2, "date": 2})     // compound index
db.docx.createIndex({foo: "text", bar: "text"}) // text index
db.docx.createIndex({"$**": "text"})            // wildcard text index
db.docx.createIndex({"userMetadata.$**": 1})    // wildcard index
```

- **Drop Index:** `db.docx.dropIndex("name_3")`
- **Hide/Unhide Indexes**
    - To Hide: `db.docx.hideIndex("name_3")`
- To Unhide: `db.docx.unhideIndex("name_3")`
- **Creating a compound index:** `db.docx.ensureIndex({name : 3, operator : 1, class : 0})`
- **Dropping a compound index:** `db.docx.dropIndex({name : 3, operator : 1, class : 0})`

20. Aggregation

1.

Operator	Description	Command
$sum	Sum up values	`db.docx.aggregate([{$group : {_id : "$operator", num_docx : {$sum : "$value"}}}])`
$avg	Calculates average values	`db.docx.aggregate([{$group : {_id : "$operator", num_docx : {$avg : "$value"}}}])`
$min / $max	Find min/max values	`db.docx.aggregate([{$group : {_id : "$operator", num_docx : {$min : "$value"}}}])`
$push	Push values to a result array	`db.docx.aggregate([{$group : {_id : "$operator", classes : {$push: "$value"}}}])`
$addToSet	Push values to a result array without duplicates	`db.docx.aggregate([{$group : {_id : "$operator", classes : {$addToSet : "$value"}}}])`
$first / $last	To get the first / last document	`db.docx.aggregate([{$group : {_id : "$operator", last_class : {$last : "$value"}}}])`

21. Databases and Collections

- **Drop:** `db.docx.drop()` `// removes the collection and its index definitions`
- **Create Collection:**

```
// Create collection with a $jsonschema
db.createCollection("contacts", {
validator: {$jsonSchema: {
bsonType: "object",
required: ["gadget"],
properties: {
phone: {
bsonType: "string",
description: "must be a string and is required"
},
email: {
bsonType: "string",
pattern: "@mongodb\.com$",
description: "must be a string and match the regular expression pattern"
 },
    status: {
     enum: [ "Unknown", "Incomplete" ],
     description: "can only be one of the enum values"
     }
     }
     }}
      })
```

- **Other Collection Functions**
  - In order to create a statistical structure and to copy a pointer into a user-specified memory location, use: `db.docx.stats()`
  - The total amount of storage in bytes allocated to the document for document storage can be known by using: `db.docx.storageSize()`
  - To report the total size used by the indexes in a collection document, use: `db.docx.totalIndexSize()`
  - The total size in bytes of the data in the collection plus the size of every index can be known by using: `db.docx.totalSize()`

22. Some other Handy commands

- **use admin**

```
db.createUser({"user": "major", "pwd": passwordPrompt(), "roles": ["major"]})
db.dropUser("major")
db.auth( "user", passwordPrompt() )
```

- **use test**

```
db.getSiblingDB("dbname")
db.currentOp()
db.killOp(345) // opid
```

- **Change Streams**

```
watchCursor = db.docx.watch( [ { $match : {"operationType" : "insert" } } ] )
while (!watchCursor.isExhausted()){
  if (watchCursor.hasNext()){
     print(tojson(watchCursor.next()));
  }
}
```

- **Search in a MongoDb Database:**

```
db.comments.find({lang:'Python'})
```

- **Showing All Collections in a Database:**

```
db.getCollectionNames()
```

- **Listing a Collection's Records:**

```
db.collectionname.find()
```

- **Listing Records with Matching Values of Specific Fields:**

```
db. collectionname.find({"field2": "secondmatching value"})
```

- **Multiple Matching Values:**

```
db. collectionname.find({"field2": "second matching value", "field3":
"thirdmatchingvalue"})
```

- **Finding a Single Record:**

```
db. collectionname.findOne({"field2": "content"})
```

Conclusion

MongoDB is one of the world's most popular document databases. It has some powerful capabilities like full-text search, data aggregation etc. One should have a solid knowledge of what MongoDB is. In this document, we've covered the basics of MongoDB, its features, and some of the important cheat sheets. We've also explored the common database operations of MongoDB. Now, it's time for you to head out and try what we've covered here and more.

Useful Resources

- Technical Interview Questions
- Coding Interview Questions
- Interview Resources
- DSA- Programming
- Mock Interview

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions