INTERVIEW QUESTIONS ON MONGODB

1. What are some of the advantages of MongoDB?

Some advantages of MongoDB are as follows:

- MongoDB supports field, range-based, string pattern matching type queries. for searching the data in the database
- MongoDB support primary and secondary index on any fields
- MongoDB basically uses JavaScript objects in place of procedures
- MongoDB uses a dynamic database schema
- MongoDB is very easy to scale up or down
- MongoDB has inbuilt support for data partitioning (Sharding).

2. What is a Document in MongoDB?

A Document in MongoDB is an ordered set of keys with associated values. It is represented by a map, hash, or dictionary. In JavaScript, documents are represented as objects:
{"greeting" : "Hello world!"}

Complex documents will contain multiple key/value pairs:
{"greeting" : "Hello world!", "views" : 3}

3. What is a Collection in MongoDB?

A collection in MongoDB is a group of documents. If a document is the MongoDB analog of a row in a relational database, then a collection can be thought of as the analog to a table.
Documents within a single collection can have any number of different "shapes.", i.e. collections have dynamic schemas.
For example, both of the following documents could be stored in a single collection:

```
{"greeting" : "Hello world!", "views": 3}

{"signoff": "Good bye"}
```

4. What are Databases in MongoDB?

MongoDB groups collections into databases. MongoDB can host several databases, each grouping together collections.
Some reserved database names are as follows:
admin
local
config

5. What is the Mongo Shell?

It is a JavaScript shell that allows interaction with a MongoDB instance from the command line. With that one can perform administrative functions, inspecting an instance, or exploring MongoDB.

To start the shell, run the mongo executable:

```
$ mongod

$ mongo

MongoDB shell version: 4.2.0

connecting to: test

>
```

The shell is a full-featured JavaScript interpreter, capable of running arbitrary JavaScript programs. Let's see how basic math works on this:

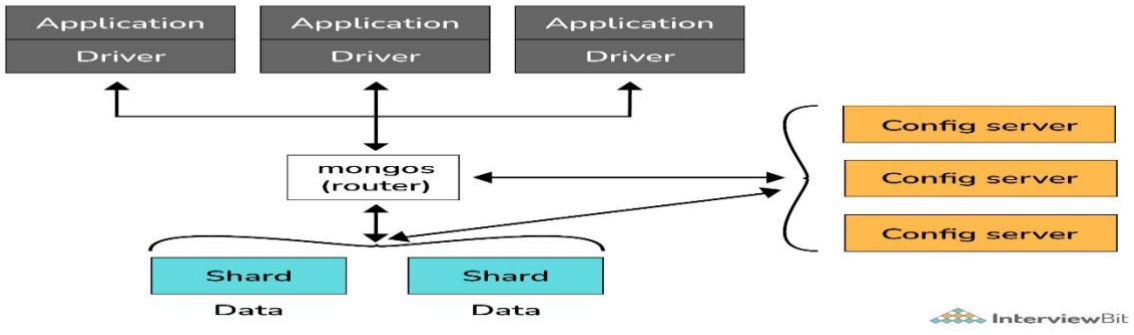
```
> x = 100;

200

> x / 5;

20
```

6. How does Scale-Out occur in MongoDB?

The document-oriented data model of MongoDB makes it easier to split data across multiple servers. Balancing and loading data across a cluster is done by MongoDB. It then redistributes documents automatically.

The mongos acts as a query router, providing an interface between client applications and the sharded cluster.

Config servers store metadata and configuration settings for the cluster. MongoDB uses the config servers to manage distributed locks. Each sharded cluster must have its own config servers.



7. What are some features of MongoDB?

- **Indexing:** It supports generic secondary indexes and provides unique, compound, geospatial, and full-text indexing capabilities as well.
- **Aggregation:** It provides an aggregation framework based on the concept of data processing pipelines.
- **Special collection and index types:** It supports time-to-live (TTL) collections for data that should expire at a certain time
- **File storage:** It supports an easy-to-use protocol for storing large files and file metadata.
- **Sharding:** Sharding is the process of splitting data up across machines.

8. How to add data in MongoDB?

The basic method for adding data to MongoDB is “inserts”. To insert a single document, use the collection’s insertOne method:

```
> db.books.insertOne({"title" : "Start With Why"})
```

For inserting multiple documents into a collection, we use insertMany. This method enables passing an array of documents to the database.

9. How do you Update a Document?

Once a document is stored in the database, it can be changed using one of several update methods: updateOne, updateMany, and replaceOne. updateOne and updateMany each takes a filter document as their first parameter and a modifier document, which describes changes to make, as the second parameter. replaceOne also takes a filter as the first parameter, but as the second parameter replaceOne expects a document with which it will replace the document matching the filter.

For example, in order to replace a document:

```
{
  "_id" : ObjectId("4b2b9f67a1f631733d917a7a"),
  "name" : "alice",
  "friends" : 24,
  "enemies" : 2
}
```

10. How do you Delete a Document?

The CRUD API in MongoDB provides deleteOne and deleteMany for this purpose. Both of these methods take a filter document as their first parameter. The filter specifies a set of criteria to match against in removing documents.

For example:
> db.books.deleteOne({"_id" : 3})

11. How to perform queries in MongoDB?

The find method is used to perform queries in MongoDB. Querying returns a subset of documents in a collection, from no documents at all to the entire collection. Which documents get returned is determined by the first argument to find, which is a document specifying the query criteria.

Example:
> db.users.find({"age" : 24})

12. What are the data types in MongoDB?

MongoDB supports a wide range of data types as values in documents. Documents in MongoDB are similar to objects in JavaScript. Along with JSON's essential key/value–pair nature, MongoDB adds support for a number of additional data types. The common data types in MongoDB are:

- Null
  {"x" : null}
- Boolean
  {"x" : true}
- Number
  {"x" : 4}
- String
  {"x" : "foobar"}
- Date
  {"x" : new Date()}
- Regular expression
  {"x" : /foobar/i}
- Array
  {"x" : ["a", "b", "c"]}
- Embedded document
  {"x" : {"foo" : "bar"}}

- Object ID
  {"x" : ObjectId()}
- Binary Data
  Binary data is a string of arbitrary bytes.
- Code
  {"x" : function() { /* ... */ }}

13. When to use MongoDB?

You should use MongoDB when you are building internet and business applications that need to evolve quickly and scale elegantly. MongoDB is popular with developers of all kinds who are building scalable applications using agile methodologies.
MongoDB is a great choice if one needs to:

- Support a rapid iterative development.
- Scale to high levels of read and write traffic - MongoDB supports horizontal scaling through Sharding, distributing data across several machines, and facilitating high throughput operations with large sets of data.
- Scale your data repository to a massive size.
- Evolve the type of deployment as the business changes.
- Store, manage and search data with text, geospatial, or time-series dimensions.

14. How is Querying done in MongoDB?

The find method is used to perform queries in MongoDB. Querying returns a subset of documents in a collection, from no documents at all to the entire collection. Which documents get returned is determined by the first argument to find, which is a document specifying the query criteria.

For example: If we have a string we want to match, such as a "username" key with the value "alice", we use that key/value pair instead:

> db.users.find({"username" : "alice"})

15. Explain the term “Indexing” in MongoDB.

In MongoDB, indexes help in efficiently resolving queries. What an Index does is that it stores a small part of the data set in a form that is easy to traverse. The index stores the value of the specific field or set of fields, ordered by the value of the field as specified in the index.

MongoDB’s indexes work almost identically to typical relational database indexes.

Indexes look at an ordered list with references to the content. These in turn allow MongoDB to query orders of magnitude faster. To create an index, use the createIndex collection method.

For example:

> db.users.find({"username": "user101"}).explain("executionStats")

Here, executionStats mode helps us understand the effect of using an index to satisfy queries.

16. What are Geospatial Indexes in MongoDB?

MongoDB has two types of geospatial indexes: 2dsphere and 2d. 2dsphere indexes work with spherical geometries that model the surface of the earth based on the WGS84 datum. This datum model the surface of the earth as an oblate spheroid, meaning that there is some flattening at the poles. Distance calculations using 2sphere indexes, therefore, take the shape of the earth into account and provide a more accurate treatment of distance between, for example, two cities, than do 2d indexes. Use 2d indexes for points stored on a two-dimensional plane.

2dsphere allows you to specify geometries for points, lines, and polygons in the GeoJSON format. A point is given by a two-element array, representing [longitude, latitude]:

```
{
  "name" : "New York City",
  "loc" : {
     "type" : "Point",
     "coordinates" : [50, 2]
  }
}
```

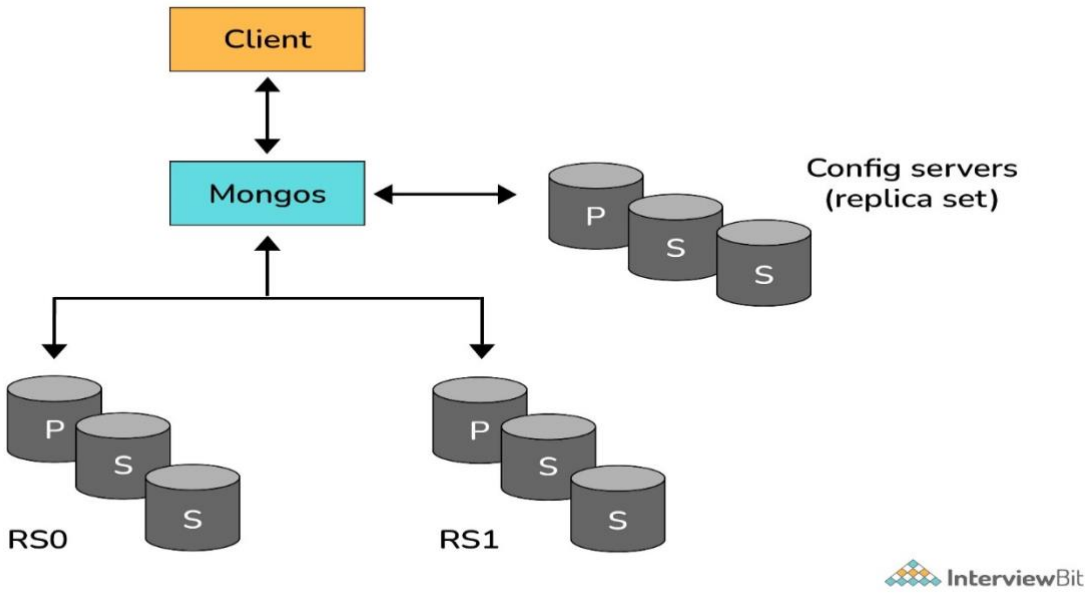
A line is given by an array of points:

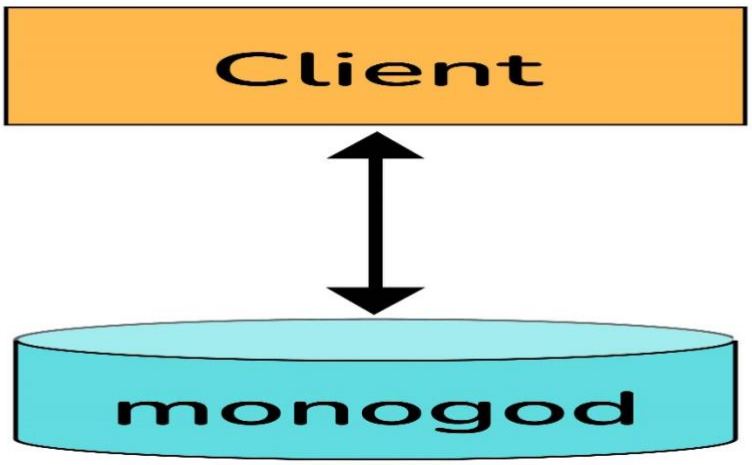
```
{
  "name" : "Hudson River",
  "loc" : {
     "type" : "LineString",
```

```
    "coordinates" : [[0,1], [0,2], [1,2]]
  }
}
```

17. Explain the process of Sharding.

Sharding is the process of splitting data up across machines. We also use the term “partitioning” sometimes to describe this concept. We can store more data and handle more load without requiring larger or more powerful machines, by putting a subset of data on each machine.
In the figure below, RS0 and RS1 are shards. MongoDB’s sharding allows you to create a cluster of many machines (shards) and break up a collection across them, putting a subset of data on each shard. This allows your application to grow beyond the resource limits of a standalone server or replica set.





Non Sharded Client Connection

18. Explain the SET Modifier in MongoDB?

If the value of a field does not yet exist, the "$set" sets the value. This can be useful for updating schemas or adding user-defined keys.

Example:

```
> db.users.findOne()
{
  "_id" : ObjectId("4b253b067525f35f94b60a31"),
  "name" : "alice",
  "age" : 23,
  "sex" : "female",
  "location" : "India"
}
```

To add a field to this, we use “$set”:

```
> db.users.updateOne({"_id" :
ObjectId("4b253b067525f35f94b60a31")},
... {"$set" : {"favorite book" : "Start with Why"}})
```

19. What do you mean by Transactions?

A transaction is a logical unit of processing in a database that includes one or more database operations, which can be read or write operations. Transactions provide a useful feature in MongoDB to ensure consistency.

MongoDB provides two APIs to use transactions.

- **Core API:** It is a similar syntax to relational databases (e.g., start_transaction and commit_transaction)
- **Call-back API:** This is the recommended approach to using transactions. It starts a transaction, executes the specified operations, and commits (or aborts on the error). It also automatically incorporates error handling logic for "TransientTransactionError" and"UnknownTransactionCommitResult".

20. What are MongoDB Charts?

MongoDB Charts is a new, integrated tool in MongoDB for data visualization.

MongoDB Charts offers the best way to create visualizations using data from a

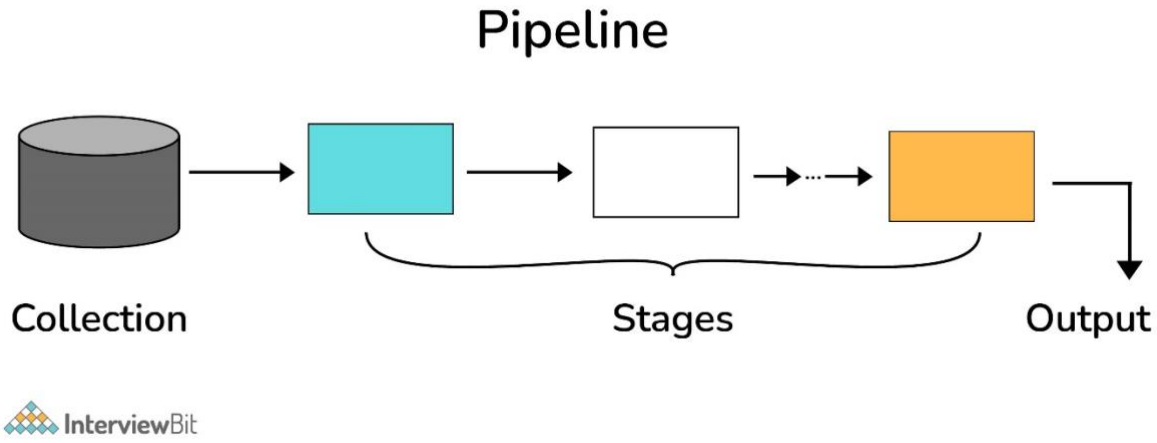
MongoDB database.
It allows users to perform quick data representation from a database without writing code in a programming language such as Java or Python.

The two different implementations of MongoDB Charts are:

- MongoDB Charts PaaS (Platform as a Service)
- MongoDB Charts Server

21. What is the Aggregation Framework in MongoDB?

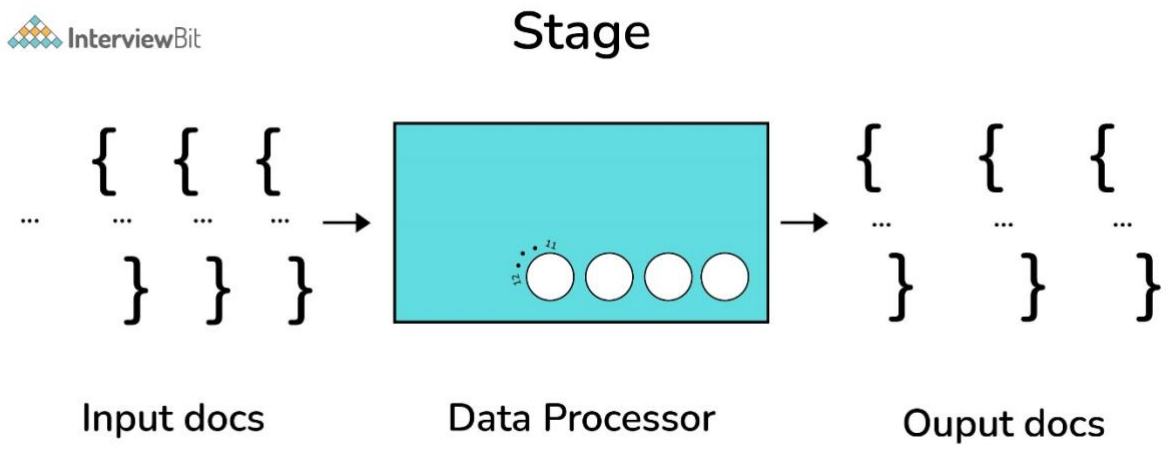
- The aggregation framework is a set of analytics tools within MongoDB that allow you to do analytics on documents in one or more collections.
- The aggregation framework is based on the concept of a pipeline. With an aggregation pipeline, we take input from a MongoDB collection and pass the documents from that collection through one or more stages, each of which performs a different operation on its inputs (See figure below). Each stage takes as input whatever the stage before it produced as output. The inputs and outputs for all stages are documents—a stream of documents.



InterviewBit

22. Explain the concept of pipeline in the MongoDB aggregation framework.

An individual stage of an aggregation pipeline is a data processing unit. It takes in a stream of input documents one at a time, processes each document one at a time, and produces an output stream of documents one at a time (see figure below).



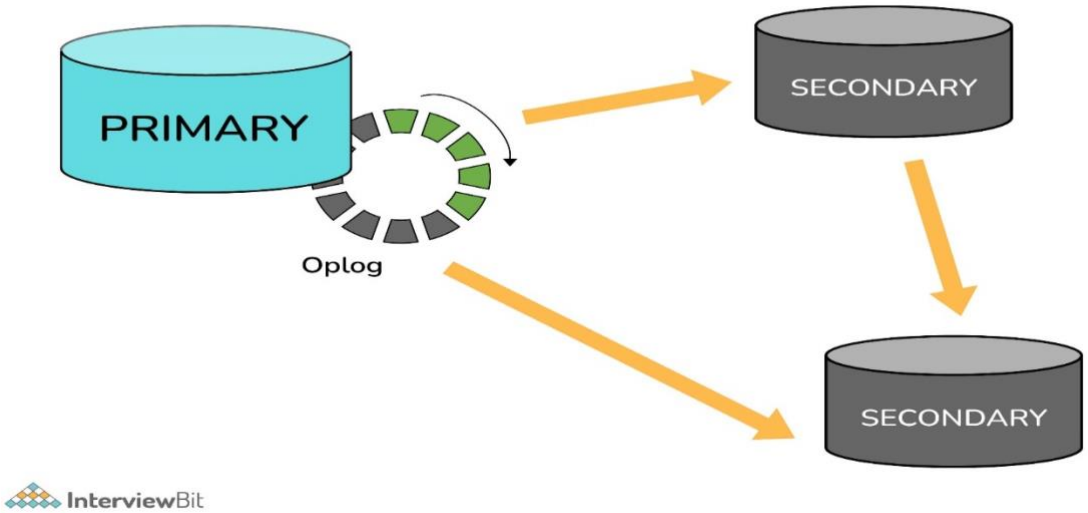
23. What is a Replica Set in MongoDB?

To keep identical copies of your data on multiple servers, we use replication. It is recommended for all production deployments. Use replication to keep your application running and your data safe, even if something happens to one or more of your servers.

Such replication can be created by a replica set with MongoDB. A replica set is a group of servers with one primary, the server taking writes, and multiple secondaries, servers that keep copies of the primary's data. If the primary crashes, the secondaries can elect a new primary from amongst themselves.

24. Explain the Replication Architecture in MongoDB.

The following diagram depicts the architecture diagram of a simple replica set cluster with only three server nodes – one primary node and two secondary nodes:



- In the preceding model, the PRIMARY database is the only active replica set member that receives write operations from database clients. The PRIMARY database saves data changes in the Oplog. Changes saved in the Oplog are sequential—that is, saved in the order that they are received and executed.
- The SECONDARY database is querying the PRIMARY database for new changes in the Oplog. If there are any changes, then Oplog entries are copied from PRIMARY to SECONDARY as soon as they are created on the PRIMARY node.
- Then, the SECONDARY database applies changes from the Oplog to its own datafiles. Oplog entries are applied in the same order they were inserted in the log. As a result, datafiles on SECONDARY are kept in sync with changes on PRIMARY.
- Usually, SECONDARY databases copy data changes directly from PRIMARY. Sometimes a SECONDARY database can replicate data from another SECONDARY. This type of replication is called Chained Replication because it is a two-step replication process. Chained replication is useful in certain replication topologies, and it is enabled by default in MongoDB.

25. What are some utilities for backup and restore in MongoDB?

The mongo shell does not include functions for exporting, importing, backup, or restore. However, MongoDB has created methods for accomplishing this, so that no scripting work or complex GUIs are needed. For this, several utility scripts are provided that can be used to get data in or out of the database in bulk. These utility scripts are:

- mongoimport
- mongoexport
- mongodump
- mongorestore

1) Explain what is MongoDB?

Mongo-DB is a document database which provides high performance, high availability and easy scalability.

2) What is "Namespace" in MongoDB?

MongoDB stores BSON (Binary Interchange and Structure Object Notation) objects in the collection. The concatenation of the collection name and database name is called a namespace.

3) What is sharding in MongoDB?

The procedure of storing data records across multiple machines is referred as Sharding. It is a MongoDB approach to meet the demands of data growth. It is the horizontal partition of data in a database or search engine. Each partition is referred as shard or database shard.

4) How can you see the connection used by Mongos?

To see the connection used by Mongos use db_adminCommand ("connPoolStats");

5) Explain what is a replica set?

A replica set is a group of mongo instances that host the same data set. In replica set, one node is primary, and another is secondary. From primary to the secondary node all data replicates.

6) How replication works in MongoDB?

Across multiple servers, the process of synchronizing data is known as replication. It provides redundancy and increase data availability with multiple copies of data on different database server. Replication helps in protecting the database from the loss of a single server.

7) While creating Schema in MongoDB what are the points need to be taken in consideration?

Points need to be taken in consideration are

- Design your schema according to user requirements
- Combine objects into one document if you use them together. Otherwise, separate them
- Do joins while write, and not when it is on read
- For most frequent use cases optimize your schema
- Do complex aggregation in the schema

8) What is the syntax to create a collection and to drop a collection in MongoDB?

- Syntax to create collection in MongoDB is db.createCollection(name,options)
- Syntax to drop collection in MongoDB is db.collection.drop()

9) Explain what is the role of profiler in MongoDB?

MongoDB database profiler shows performance characteristics of each operation against the database. You can find queries using the profiler that are slower than they should be.

10) Explain can you move old files in the moveChunk directory?

Yes, it is possible to move old files in the moveChunk directory, during normal shard balancing operations these files are made as backups and can be deleted once the operations are done.

11) To do safe backups what is the feature in MongoDB that you can use?

Journaling is the feature in MongoDB that you can use to do safe backups.

12) Mention what is Objecld composed of?

Objectld is composed of

- Timestamp
- Client machine ID
- Client process ID
- 3 byte incremented counter

13) Mention what is the command syntax for inserting a document?

For inserting a document command syntax is database.collection.insert (document).

14) Mention how you can inspect the source code of a function?

To inspect a source code of a function, without any parentheses, the function must be invoked.

15) What is the command syntax that tells you whether you are on the master server or not? And how many master does MongoDB allow?

Command syntax Db.isMaster() will tell you whether you are on the master server or not. MongoDB allows only one master server, while couchDB allows multiple masters.

16) Mention the command syntax that is used to view Mongo is using the link?

The command syntax that is used to view mongo is using the link is db._adminCommand(“connPoolStats.”)

17) Explain what are indexes in MongoDB?

Indexes are special structures in MongoDB, which stores a small portion of the data set in an easy to traverse form. Ordered by the value of the field specified in the index, the index stores the value of a specific field or set of fields.

18) Mention what is the basic syntax to use index in MongoDB?

The basic syntax to use in MongoDB is >db.COLLECTION_NAME.ensureIndex ({KEY:1}). In here the key is the the name of the COLUMN (or KEY:VALUE pair) which is present in the documents.

19) Explain what is GridFS in MongoDB?

For storing and retrieving large files such as images, video files and audio files GridFS is used. By default, it uses two files fs.files and fs.chunks to store the file's metadata and the chunks.

20) What are alternatives to MongoDB?

Cassandra, CouchDB, Redis, Riak, Hbase are a few good alternatives.

1) What do you understand by NoSQL databases? Is MongoDB a NoSQL database? explain.

At the present time, the internet is loaded with big data, big users, big complexity etc. and also becoming more complex day by day. NoSQL is answer of all these problems, It is not a traditional database management system, not even a relational database management system (RDBMS). NoSQL stands for "Not Only SQL". NoSQL is a type of database that can handle and sort all type of unstructured, messy and complicated data. It is just a new way to think about the database.

Yes. MongoDB is a NoSQL database.

2) Which are the different languages supported by MongoDB?

MonggoDB provides official driver support for C, C++, C#, Java, Node.js, Perl, PHP, Python, Ruby, Scala, Go and Erlang.

You can use MongoDB with any of the above languages. There are some other community supported drivers too but the above mentioned ones are officially provided by MongoDB.

3) What are the different types of NoSQL databases? Give some example.

NoSQL database can be classified as 4 basic types:

1. Key value store NoSQL database

2. Document store NoSQL database
3. Column store NoSQL database
4. Graph base NoSQL databse

There are many NoSQL databases. MongoDB, Cassandra, CouchBD, Hypertable, Redis, Riak, Neo4j, HBASE, Couchbase, MemcacheDB, Voldemort, RevenDB etc. are the examples of NoSQL databases.

4) Is MongoDB better than other SQL databases? If yes then how?

MongoDB is better than other SQL databases because it allows a highly flexible and scalable document structure.

For example:

- One data document in MongoDB can have five columns and the other one in the same collection can have ten columns.
- MongoDB database are faster than SQL databases due to efficient indexing and storage techniques.

5) What type of DBMS is MongoDB?

MongoDB is a document oriented DBMS

6) What is the difference between MongoDB and MySQL?

Although MongoDB and MySQL both are free and open source databases, there is a lot of difference between them in the term of data representation, relationship, transaction, querying data, schema design and definition, performance speed, normalization and many more. To compare MySQL with MongoDB is like a comparison between Relational and Non-relational databases.

7) Why MongoDB is known as best NoSQL database?

MongoDb is the best NoSQL database because, it is:

Document Oriented

Rich Query language

High Performance

Highly Available

Easily Scalable

8) Does MongoDB support primary-key, foreign-key relationship?

No. By Default, MongoDB doesn't support primary key-foreign key relationship.

9) Can you achieve primary key - foreign key relationships in MongoDB?

We can achieve primary key-foreign key relationship by embedding one document inside another. For example: An address document can be embedded inside customer document.

10) Does MongoDB need a lot of RAM?

No. There is no need a lot of RAM to run MongoDB. It can be run even on a small amount of RAM because it dynamically allocates and de-allocates RAM according to the requirement of the processes.

11) Explain the structure of ObjectID in MongoDB.

ObjectID is a 12-byte BSON type. These are:

- 4 bytes value representing seconds
- 3 byte machine identifier
- 2 byte process id
- 3 byte counter

12) Is it true that MongoDB uses BSON to represent document structure?

Yes.

13) What are Indexes in MongoDB?

In MondoDB, Indexes are used to execute query efficiently. Without indexes, MongoDB must perform a collection scan, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.

14) By default, which index is created by MongoDB for every collection?

By default, the_id collection is created for every collection by MongoDB.

15) What is a Namespace in MongoDB?

Namespace is a concatenation of the database name and the collection name. Collection, in which MongoDB stores BSON objects.

16) Can journaling features be used to perform safe hot backups?

Yes.

17) Why does Profiler use in MongoDB?

MongoDB uses a database profiler to perform characteristics of each operation against the database. You can use a profiler to find queries and write operations

18) If you remove an object attribute, is it deleted from the database?

Yes, it be. Remove the attribute and then re-save() the object.

19) In which language MongoDB is written?

MongoDB is written and implemented in C++.

20) Does MongoDB need a lot space of Random Access Memory (RAM)?

No. MongoDB can be run on small free space of RAM.

21) What language you can use with MongoDB?

MongoDB client drivers supports all the popular programming languages so there is no issue of language, you can use any language that you want.

22) Does MongoDB database have tables for storing records?

No. Instead of tables, MongoDB uses "Collections" to store data.

23) Do the MongoDB databases have schema?

Yes. MongoDB databases have dynamic schema. There is no need to define the structure to create collections.

24) What is the method to configure the cache size in MongoDB?

MongoDB's cache is not configurable. Actually MongoDb uses all the free spaces on the system automatically by way of memory mapped files.

25) How to do Transaction/locking in MongoDB?

MongoDB doesn't use traditional locking or complex transaction with Rollback. MongoDB is designed to be light weighted, fast and predictable to its performance. It keeps transaction support simple to enhance performance.

26) Why 32 bit version of MongoDB are not preferred ?

Because MongoDB uses memory mapped files so when you run a 32-bit build of MongoDB, the total storage size of server is 2 GB. But when you run a 64-bit build of MongoDB, this provides virtually unlimited storage size. So 64-bit is preferred over 32-bit.

27) Is it possible to remove old files in the moveChunk directory?

Yes, These files can be deleted once the operations are done because these files are made as backups during normal shard balancing operation. This is a manual cleanup process and necessary to free up space.

28) What will have to do if a shard is down or slow and you do a query?

If a shard is down and you even do query then your query will be returned with an error unless you set a partial query option. But if a shard is slow them Mongos will wait for them till response.

29)Explain the covered query in MongoDB.

A query is called covered query if satisfies the following two conditions:

- The fields used in the query are part of an index used in the query.
- The fields returned in the results are in the same index.

30) What is the importance of covered query?

Covered query makes the execution of the query faster because indexes are stored in RAM or sequentially located on disk. It makes the execution of the query faster.

Covered query makes the fields are covered in the index itself, MongoDB can match the query condition as well as return the result fields using the same index without looking inside the documents.

31) What is sharding in MongoDB?

In MongoDB, Sharding is a procedure of storing data records across multiple machines. It is a MongoDB approach to meet the demands of data growth. It creates horizontal partition of data in a database or search engine. Each partition is referred as shard or database shard.

32) What is replica set in MongoDB?

A replica can be specified as a group of mongo instances that host the same data set. In a replica set, one node is primary, and another is secondary. All data is replicated from primary to secondary nodes.

33) What is primary and secondary replica set in MongoDB?

In MongoDB, primary nodes are the node that can accept write. These are also known as master nodes. The replication in MongoDB is single master so, only one node can accept write operations at a time.

Secondary nodes are known as slave nodes. These are read only nodes that replicate from the primary.

34) By default, which replica sets are used to write data?

By default, MongoDB writes data only to the primary replica set.

35) What is CRUD in MongoDB?

MongoDB supports following CRUD operations:

- Create
- Read
- Update
- Delete

36) In which format MongoDB represents document structure?

MongoDB uses BSON to represent document structures.

37) What will happen when you remove a document from database in MongoDB? Does MongoDB remove it from disk?

Yes. If you remove a document from database, MongoDB will remove it from disk too.

38) Why are MongoDB data files large in size?

MongoDB doesn't follow file system fragmentation and pre allocates data files to reserve space while setting up the server. That's why MongoDB data files are large in size.

39) What is a storage engine in MongoDB?

A storage engine is the part of a database that is used to manage how data is stored on disk.

For example: one storage engine might offer better performance for read-heavy workloads, and another might support a higher-throughput for write operations.

40) Which are the storage engines used by MongoDB?

MMAPv1 and WiredTiger are two storage engine used by MongoDB.

41) What is the usage of profiler in MongoDB?

A database profiler is used to collect data about MongoDB write operations, cursors, database commands on a running mongod instance. You can enable profiling on a per-database or per-instance basis.

The database profiler writes all the data it collects to the system. profile collection, which is a capped collection.

42) Is it possible to configure the cache size for MMAPv1 in MongoDB?

No. it is not possible to configure the cache size for MMAPv1 because MMAPv1 does not allow configuring the cache size.

43) How to configure the cache size for WiredTiger in MongoDB?

For the WiredTiger storage engine, you can specify the maximum size of the cache that WiredTiger will use for all data. This can be done using storage.wiredTiger.engineConfig.cacheSizeGB option.

44) How does MongoDB provide concurrency?

MongoDB uses reader-writer locks for concurrency. Reader-writer locks allow concurrent readers shared access to a resource, such as a database or collection, but give exclusive access to a single write operation.

45) What is the difference between MongoDB and Redis database?

Difference between MongoDB and Redis:

- Redis is faster than MongoDB.
- Redis has a key-value storage whereas MongoDB has a document type storage.
- Redis is hard to code but MongoDB is easy.

46) What is the difference between MongoDB and CouchDB?

Difference between MongoDB and CouchDB:

- MongoDB is faster than CouchDB while CouchDB is safer than MongoDB.
- Triggers are not available in MongoDB while triggers are available in CouchDB.
- MongoDB serializes JSON data to BSON while CouchDB doesn't store data in JSON format.

47) What is the difference between MongoDB and Cassandra?

Difference between MongoDB and Cassandra:

- MongoDB is cross-platform document-oriented database system while Cassandra is high performance distributed database system.
- MongoDB is written in C++ while Cassandra is written in Java.

- MongoDB is easy to administer in the case of failure while Cassandra provides high availability with no single point of failure.

48) Is there any need to create database command in MongoDB?

You don't need to create a database manually in MongoDB because it creates automaically when you save the value into the defined collection at first time.

1) What is MongoDB?

MongoDB is a cross-platform document-based database. Categorized as a NoSQL database, MongoDB avoids the conventional table-oriented relational database structure in support of the JSON-like documents with the dynamic schemas, making the data integration in specific kinds of applications quicker and simpler.

MongoDB was developed by a software company “10gen”, in October 2007 as an element of the planned platform as the service product. After that, the company was shifted to a freeware deployment model in 2009, providing sales assistance and other services.

2) What are the features of MongoDB?

Following are the important features of MongoDB:

- A compliant data model in the format of documents.
- Agile and extremely scalable database.
- Quicker than traditional databases.
- Demonstrative query language.

3)What type of NoSQL database MongoDB is?

MongoDB is a document-oriented database. It stores the data in the form of the BSON structure-oriented databases. We store these documents in a collection.

4) Explain Namespace?

A namespace is the series of the collection name and database name.

5)Differentiate MongoDB and MySQL?

Despite MySQL and MongoDB being freeware and open source databases, there are several differences between them in terms of a data relationship, transaction, performance speed, querying data, schema design, normalization, etc. Comparison between MongoDB and MySQL is similar to the comparison between Non-relational and Relational databases.

6) Explain about Indexes in MongoDB?

In MongoDB, we use Indexes for executing the queries efficiently; without using Indexes, MongoDB should carry out a collection scan, i.e., scan all the documents of a collection, for selecting the documents which match the query statement. If a suitable index is available for a query, MongoDB will use an index for restricting the number of documents it should examine.

7) Why MongoDB is the best NoSQL database?

MongoDB is the best NoSQL database due to the following features:

- High Performance
- High Availability
- Easily Scalable
- Rich Query Language
- Document Oriented

8) Explain the significance of the covered query?

A covered query makes the query implementation quicker as we store the indexes in the RAM or consecutively located on the disk. It makes query execution quicker. The covered query covers all the fields in the index, MongoDB matches the query condition along with returning the result fields.

9) What is a replica set?

We can specify the replica as a set of the mongo instances which host a similar data set. In the replica set, one node will be primary, and another one will be secondary. We replicate all the data from the primary to the secondary nodes.

10) Differentiate MongoDB and Cassandra?

MongoDB	Cassandra
It is a cross-platform document-oriented database system	It is a high-performance distributed database system.
It is developed in C++	It is developed in Java
It is simple to administer in the failure case	It offers high availability

11) Explain the primary and secondary replica set?

In MongoDB, primary nodes are the nodes that accept write. Primary nodes are also called master nodes. Replication in MongoDB is a single master. Therefore, only one node will accept the write operations at once.

12) Which languages can we use with MongoDB?

At Present, MongoDB offers driver support to c++, java, PHP, Perl, Python, Go, Scala, Ruby.

13) Explain Storage Encryption?

Storage encryption encodes all the MongoDB data over the storage or over the operating systems for assuring that only authenticated processes will access the safeguarded data.

14) Explain Primary and Secondary Replica Sets?

Primary Replica Set receives all the write operations from the clients. Secondary replica sets replicate the primary replica sets and implement the operations for their datasets so that secondary datasets affect the primary datasets.

15) What is the importance of GridFS and Journaling?

- GridFS: We use GridFS to retrieve and store large files like images, videos, and audio files.
- J
- ournaling: We use Journaling for secure backups in MongoDB.

16) How to do locking or transactions in MongoDB?

MongoDB does not use traditional locking with the reduction because it is high-speed, knowable, and light in the presentation. We can consider it as the MyISAM, MySQL auto entrust script. Through the simpler business sustain, we can enhance the performance, specifically in the structure with various servers.

17) How to do Journaling in MongoDB?

We save the write operations in the memory while journaling is taking place. The on-disk journal files are dependable for the reason that journal writers are usual. In the DB path, MongoDB designs a journal subdirectory.

18) How does MongoDB provides concurrency?

MongoDB utilizes the reader-writer locks, enabling concurrent readers to access any supply such as collection or database though it provides private access to individual writers.

19) Explain Sharding and Aggregation in MongoDB?

- Aggregation: Aggregations are the activities that handle the data records and give the record results.
- Sharding: Sharding means storing the data on multiple machines.

20) What is the importance of profiler in MongoDB?

MongoDB contains the database profiler that shows the performance characteristics of every operation against the database. Through the profiler, we can identify the queries that are slower than they should be and use this data to determine when we require an index.

21) Define Collection?

The collection is a set of MongoDB documents.

22) Explain Aggregation Pipeline?

The aggregation Pipeline acts as a framework to perform aggregation tasks. We use this pipeline for transforming the documents into aggregated results.

23) Explain MapReduce?

MapReduce is a standard multi-phase data aggregation modality which we use to process the data quantities.

24) Explain Splitting?

Splitting is the background process that we use to store chunks from increasing too large.

25) What is the purpose of the save() method?

We use the save() method for replacing the existing documents with new documents.

26) What is the use of MongoDB?

- Generally, we use MongoDB as the main data store for the operational requirements with live needs. Generally, MongoDB is suitable for 80% of the applications which we develop today. MongoDB is simple to operate and extent in ways that are tough if they are not possible with the relational databases.
- MongoDB stands out in various use cases where the relational databases are not suitable, like applications with semi-structured, structured, along with the big scalability needs or the multi-datacenter deployments.
- MongoDB cannot be suitable for some applications. For instance, applications that need complex transactions and scan-based applications that access huge subsets of the data largely cannot be suitable for MongoDB.
- Some general uses of MongoDB comprise product catalogs, mobile apps, content management, real-time personalization, and applications providing individual views throughout several systems.

27) What is the purpose of the DB command?

We use the "DB" command to get the name of the presently selected database.

28) What are the restrictions of the MongoDB 32-bit versions?

When we run a 32-bit version of MongoDB, the total storage size of the server, containing indexes and data, is 2GB. Due to this reason, we will not deploy MongoDB to the production on the 32-bit machines. If we deploy a 64-bit version of MongoDB, there is no virtual restriction to the storage size. For the creation deployments, we strongly recommend 64-bit operating systems and builds.

29) When should we normalize the data in MongoDB?

It relies on our objectives. Normalization provides an updated effective data representation. Denormalisation makes data reading effective. Generally, we utilize embedded data models when:

- When we have "contains" relationships between the entities.
- When we have one-to-many relationships between the entities. In the relationships, "many" or the child documents display in the context of the parent documents.

Generally, we use normalized data models:

- When embedding results in duplication of the data yet they will not give enough read performance advantages to prevail the duplication implications.
- For representing more difficult many-to-many relationships.
- For modeling the big hierarchical data sets.

30) How do we perform sorting and Explain Project in MongoDB?

For finding any data in MongoDB, we use the find() method. The discovery () method returns the collection's documents over which we invoked this method. We can use the "Where" clause in the MongoDB query in order to restrict the

output by using MongoDB projection. Anytime we execute the find() method, MongoDB returns all the documents associated with a particular collection.

```
db.<collection_name>.find({ }, {<key_Name>:<Flag to display>})
```

31) How can MongoDB simulate subquery or join?

We have to find the best method for structuring the data in MongoDB for simulating what would be the simple subquery or join in SQL. For example, we have users and posts, with the users in one collection and posts in another collection. We have to find all the posts by the users whose city is “Hyderabad”.

32) Define oplog(operational log)?

Operational log(oplog) is a special kind of limited collection that stores a rolling record of all the operations which change the data we store in our databases. Primarily, it applies all the database operations over the primary and, after that, records these operations on the oplog of the primary. After that, the secondary members replicate and apply the operations in the asynchronous process.

33) How do we create a database in MongoDB?

When I want to create a database in MongoDB, I faced the following error:

```
:~$mongo


MongoDB shell version:1.65


Connecting to: test


Error: Could not connect to the server


Exception: connect failed
```

The solution to the above error:

1. cd/var1/lib1/MongoDB

2. We remove the mongod. lock from the folder

3. Sudo start MongoDB

4. Mongo

34) What is the syntax of the skip() method?

skip() method syntax is:

```
db.COLLECTION_NAME.find().limit(NUMBER).skip(NUMBER)
```

35) How do we delete everything from the MongoDB database?

By using the following code, we can delete everything from the MongoDB database:

```
use [database];
db.dropDatabase();
Ruby code should be pretty similiar.
Also, from the command line:
mongo [Database] -eval "db.dropDatabase();"
use
[databaseName]
db.Drop+databasename();
drop colllection
use databaseName
db.collectionName.drop();
```

36) Which command we use for creating the backup of the database?

We use the mongodump command for creating the database backup.

37) Which command we use for restoring the backup?

We use mongorestore for restoring the backup.

38) Explain the importance of the dot notation?

In MongoDB, we use dot notation for accessing the array elements and the fields of an embedded document.

39) What is the syntax of the limit() and sort() method?

Syntax of the limit() method is:

```
>db.COLLECTION_NAME.find().limit(NUMBER)
```

Syntax of the sort() method is:

```
>db.COLLECTION_NAME.find().sort({KEY:1})
```

40) What do you know about NoSQL databases? What are the various types of NoSQL databases?

NoSQL refers to “Not Only SQL”. NoSQL is a kind of database that handles and sorts all kinds of structured, massive, and difficult data. It is a new method to think about databases. Kinds of NoSQL databases:

- Key-Value
- Graph
- Column Oriented
- Document Oriented

41) Which command we use for dropping a database?

We use the “DB.drop database” command for dropping a database.

42) Explain MongoDB Projection

In MongoDB, we use Projection for selecting only the required data. It will not select the complete data of a document.

43) Why do we use the pretty() method?

We use the pretty() method for displaying the results in a formatted way.

44) How do we remove a document from the collection?

By using the remove() method, we remove a document from the collection.

45) What are the points we should consider while creating a schema in MongoDB?

We must consider the following points while creating a schema:

- Designing the Scheme based on the user requirements.
- Combining the objects into one document, if we have to use them jointly, or else, separate them.
- Perform joins while on write, and not while it is on reading.

- For most general application scenarios, maximize the schema.
- Perform complex aggregations in the schema.

46) What does ObjectId contain?

ObjectId contains the following:

- Client machine ID
- Client process ID
- Byte incremented counter
- Timestamp

47) How do we use the select * group by MongoDB aggregation?

For instance, if we have to select all the attributes and groups by name throughout the records. For example:

```
{Name: George, x: 5, y: 3}
{Name: George, z: 9}
{Name: Rob, x: 12, y: 2}
```

We can do MongoDB aggregation as follows:

```
db.example.aggregate(
  {
    $group:{
      _id:'$name',
x: {$addToSet: "$x"    },
y: {$addToSet: "$y"    },
z: {$addToSet: "$z"    },
 }
}
)
```

48) Explain Vertical Scaling and Horizontal Scaling?

- **Vertical Scaling:** Vertical Scaling increases storage and CPU resources for expanding the capacity.
- **Horizontal Scaling:** Horizontal Scaling splits the datasets and circulates the data over multiple shards or servers.

49) What are the elements of the Sharded Cluster?

Following are the elements of the Sharded Cluster:

- Query routers
- Shards
- Config servers

50) What are the substitutes to MongoDB?

Following are the substitutes to MongoDB:

- Hbase
- CouchDB
- Cassandra
- Redis
- Riak

51) How can we old files in the moveChunk directory?

In the course of general shard balancing operations, we make the old files as backups, and we can delete them when those operations are completed.

52) What is a Storage Engine?

Storage Engine is a component of the database that is accountable to manage how we store on the disk. For instance, one storage engine may provide better performance for the read-heavy workloads, and another one may support a great throughput for the write operations.

53) Does MongoDB require plenty of RAM?

No, MongoDB does not require plenty of RAM. It can run on a small amount of memory. MongoDB dynamically assigns and unassigns RAM according to the needs of other processes.

Advanced MongoDB Interview Questions And Answers

54) Differentiate MongoDB and CouchDB?

MongoDB	**CouchDB**
MongoDB is quicker than CouchDB	CouchDB is more secure than MongoDB
Triggers do not exist in MongoDB.	Triggers exist in CouchDB
MongoDB serializes the JSON Data to the BSON	CouchDB does not store the data in JSON

55) Explain Capped Collection?

In MongoDB, the Capped collection is a special kind of collection. This indicates that in this collection, we can restrict the collection size. Syntax of Capped Collection is as follows:

```
db.createCollection(<collection_name>, {capped: Boolean, autoIndexId: Boolean, size: Number, max : Number})
```

In the Capped Collection syntax, we have the following fields:

- Collection_Name: This field is the collection name that we create as the capped collection.
- Capped: Capped is a boolean field; it is true if we create a capped collection. By default, its value is false.
- auto indexed: It is a boolean flag which we use for auto-indexing. If this flag is true, indexes will be created automatically. If the flag is false, indexes will not be created automatically.

- Size: Size is the parameter that represents the maximum amount of documents in bytes. It is the required field in the context of capped collections.
- Max: Max is the parameter that represents the highest number of documents that permit the collection.

56) How do we perform the Join operations in MongoDB?

From MongoDB3.2, we can perform the Join operation. The new $lookup operator included with the aggregation pipeline is the same as the left outer join. Example:

```
{
  $lookup:
    {
      from: <collection to join>,
      localField: <field from the input documents>,
      foreignField: <field from the documents of the "from" collection>,
      as: <output array field>
    }
}
```

57) What are the storage engines used by MongoDB?

WiredTiger and MMAPv1 are the two storage engines used by MongoDB.

58) How do we configure the cache size in MongoDB?

In MongoDB, we cannot configure the cache. MongoDB utilizes the free spaces over the system automatically by using memory-mapped files.

59) How do we control the MongoDB Performance?

We can control the MongoDB Performance by:

- Locking the Performance
- Identifying the number of connections

- Database Profiling
- Full-time Diagnostic Data Capture

60) What are the aggregate functions of MongoDB?

Following are the aggregate functions of MongoDB:

- AVG
- Sum
- Min
- Max
- First
- Push
- addTo Set
- Last

61) What are the CRUD operations of MongoDB?

Following are the CRUD operations of MongoDB:

```
Create-db.collection.insert();

Read-db.collection.find();

Update-db.collection.update();

Delete-db.collection.remove();
```

62) What are the datatypes of MongoDB?

Following are the datatypes of MongoDB:

- Integer
- String
- Boolean
- Array
- Double
- Date
- Timestamp
- Regular Expression

63) Is it required to invoke “get last error” for making a write durable?

No, it is not required to invoke “get last error”. The server acts as if it has been invoked. “get last error” enables us to acquire confirmation that a write operation is committed. You will get the confirmation, yet the durability and safety of the writer are independent.

64) What happens when the Shard is slow or down while querying?

When the Shard is slow, the query returns an error until partial query options are fixed. When the shard is reacting slowly, MongoDB waits for it.

65) How do we use a primary key in MongoDB?

“_id field” is reticent for a primary key in MongoDB. And it is a distinct value. If we do not set anything to the “_id”, it will systematically fill it with the “MongoDB Id Object”. Yet, we can store any distinct information in that field.

66) How do we see the connections utilized by MongoDB?

For seeing the connections utilized by MongoDB, we use db_adminCommand(”connPoolStats”).

67) When a “moveChunk” fails, is it required to clean up partly moved docs?

No, it is not required to clean up the partly moved docs because chunk moves are deterministic and consistent. The move will try again, and when finished, data will be on the latest Shard.

68) Explain how to start the MongoDB Instance or Server?

We have to follow the below steps for starting the MongoDB Server:

- First, open the command prompt and execute the “mongod.exe” file.
- On the other hand, we move to the path where we installed MongoDB.
- Go to the bin folder, find the “mongod.exe” file, and double click the file for executing it.
- We can go to the folder, for instance, “C: MongoDB/bin” and type mongo for connecting MongoDB by using the Shell.

69) Differences between MongoDB and RDBMS

Basis for Comparison	MongoDB	RDBMS
Definition	It is a non-relational database	It is a relational database management
Working	It works over relationships among the tables, which use columns and rows	It is a document-oriented database system through fields and documents
Scalability	It is horizontally and vertically scalable	It is vertically scalable
Performance	Performance enhances with the rise in the processors	Performance enhances with the rise in the RAM capacity
Hierarchical Data Storage	It has a built-in provision to store the hierarchical data	It is hard to store the hierarchical data
Support to Joins	It does not support difficulty Joins	It supports complex joins

Query Language	It uses BSON for database querying	It uses SQL to query the database
Javascript Support	It provides support to javascript-based clients for querying the database	It does not provide support to the javascript-based clients to query the database

70) How applications access the real-time data modifications in MongoDB?

Applications access the real-time data modifications through the Change streams that serve as the subscriber for every collection operation like delete, insert, and update.

71) What are the different kinds of Indexes in MongoDB?

Following are the different kinds of Indexes in MongoDB:

- Default: It is the "_id" which MongoDB creates.
- Compound: It is useful for multiple fields.
- Multi-key: It indexes the array data.
- Single field: It sorts and indexes over a single field.
- Geospatial: It is useful for querying the location data.
- Hashed: It indexes the hashes of the multiple fields.

71) Define BSON?

Binary JSON or BSON is a binary-encoded format of the JSON. BSON extends the JSON and offers various data fields and types.

72) How MongoDB stores the data?

As it is a document-based database, MongoDB stores the documents in Binary Javascript Object Notation or BSON, which is a binary-encoded format of JSON.

73) Does MongoDB support ACID Transaction? Define ACID Transaction?

Yes, MongoDB supports ACID Transaction. ACID refers to Atomicity, Consistency, Isolation, and Durability. Transaction manager assures that we handle these attributes.

74) Explain Composing elements or Structure of ObjectID in MongoDB?

In MongoDB, ObjectID is associated with the "_id" field, and MongoDB uses it as the default value of the "_id" in the documents. For generating "ObjectID", we use the following Syntax:

```
ObjectId([SomeHexaDecimalValue])
```

Example:

```
ObjectId() = newObjectId
```

ObjectID has the following methods:

- Str: This method provides the string representation of the object id.
- valueOf()- This method returns hexadecimal representation of the ObjectId.
- getTimeStamp()- This method returns timestamp of the ObjectId.
- toString()- This method returns the string representation of the ObjectId in "ObjectId(haxstring)".

75) How do we find array elements with multiple criteria?

For example, if we have the below documents:

```
{ _id: 1, numbers: [1000, -1000]]
{ _id: 2, numbers: [500]]
```

When we execute the following command:

```
db.example.find( { numbers: { $elemMatch: { $gt: -10, $lt: 10 } } } );
```

76) How can we sort the user-defined function? For example, x and y are integers, and how do we calculate "x-y"?

By executing the following code, we calculate x-y.

```
db.eval(function() {
return db.scratch.find().toArray().sort(function(doc1, doc2) {
return doc1.a – doc2.a
})
});


Versus the equivalent client-side sort:
db.scratch.find().toArray().sort(function(doc1, doc2) {
return doc1.a – doc2.b
});
```

By using the aggregation pipeline and "$orderby" operator, it is possible to sort.

77) Upto Which extent does the data expand to multi-slice?

MongoDB shred stands on the collection. Therefore, we store all the substances in a mass or a lump. When we have an additional time slot, then we will have few slice data achievement options, yet when we have multiple lumps, data will be extended to numerous slices.

78) How do we retrieve MongoDB databases in Javascript Array?

In the MongoDB terminal, we can run "Show DBS" to retrieve the existing databases. To get the MongoDB databases programmatically, we execute the following code:

```
use admin
dbs = db.runCommand({listDatabases: 1})
dbNames = []
for (var i in dbs.databases) { dbNames.push(dbs.databases[i].name) }
Hopefully this will help someone else.
The below will create an array of the names of the database:
var connection = new Mongo();
var dbNames = connection.getDBNames();
```

79) How do we update the object in the Nested Array?

By executing the following code, we update the object:

```
Skip code block
{
“_id” : ObjectId(“4faaba123412d654fe83hg876”),
“user_id” : 123456,
“total” : 100,
“items” : [
{
“item_name” : “my_item_one”,
“price” : 20
},
{
“item_name” : “my_item_two”,
“price” : 50
},
{
“item_name” : “my_item_three”,
“price” : 30
}
]
}
```

80) How do we retrieve a particular embedded document in a MongoDB collection?

I have the collection that has an embedded document known as notes.

```
Skip code block
{
“_id” : ObjectId(“4f7ee46e08403d063ab0b4f9”),
“name” : “MongoDB”,
```

```
“notes” : [
{
“title” : “Hello MongoDB”,
“content” : “Hello MongoDB”
},
{
“title” : “ReplicaSet MongoDB”,
“content” : “ReplicaSet MongoDB”
}
]
}
```

81) How do we query a nested Join?

To query the nested join, we use “tested”. For example:

```
{“_id” : ObjectId( “abcd” ),
“className” : “com.myUser”,
“reg” : 12345,
“test” : [
{ “className” : “com.abc”,
“testid” : “pqrs” } ] }
```

82) Can we run more than one Javascript Operation in one mongod instance?

Yes, we can run multiple javascript operations in one mongod instance.

What are NoSQL databases? What are the different types of NoSQL databases?

A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases (like SQL, Oracle, etc.).

Types of NoSQL databases:

Document Oriented

Key Value

Graph

Column Oriented

What kind of NoSQL database MongoDB is?

MongoDB is a document oriented database. It stores data in the form of BSON structure based documents. These documents are stored in a collection.

Which are the most important features of MongoDB?

Flexible data model in form of documents

Agile and highly scalable database

Faster than traditional databases

Expressive query language

What is a Namespace in MongoDB?

A Namespace is the concatenation of the database name and collection name. For e.g. school.students with school as the database and students as the collection

Which all languages can be used with MongoDB?

Currently, MonggoDB provides official driver support for C, C++, C#, Java, Node.js, Perl, PHP, Python, Ruby, Scala, Go and Erlang. MongoDB can easily be used with any of these languages. There are some other community supported drivers too but the above mentioned ones are officially provided by MongoDB.

Compare SQL databases and MongoDB at a high level.

SQL databases store data in form of tables, rows, columns and records. This data is stored in a pre-defined data model which is not very much flexible for today's real-world highly growing applications. MongoDB in contrast uses a flexible structure which can be easily modified and extended.

How is MongoDB better than other SQL databases?

MongoDB allows a highly flexible and scalable document structure. For e.g. one data document in MongoDB can have five columns and the other one in the same collection can have ten columns. Also, MongoDB database are faster as compared to SQL databases due to efficient indexing and storage techniques.

Compare MongoDB and CouchDB at high level.

Although both of these databases are document oriented, MongoDB is a better choice for applications which need dynamic queries and good performance on a very big database. On the other side, CouchDB is better used for applications with occasionally changing queries and pre-defined queries.

Does MongoDB support foreign key constraints?

No. MongoDB does not support such relationships.

Does MongoDB support ACID transaction management and locking functionalities?

No. MongoDB does not support default multi-document ACID transactions. However, MongoDB provides atomic operation on a single document.

How can you achieve primary key - foreign key relationships in MongoDB?

By default MongoDB does not support such primary key - foreign key relationships. However, we can achieve this concept by embedding one document inside another. Foe e.g. an address document can be embedded inside customer document.

Does MongoDB need a lot of RAM?

No. MongoDB can be run even on a small amount of RAM. MongoDB dynamically allocates and de-allocates RAM based on the requirements of other processes.

Does MongoDB pushes the writes to disk immediately or lazily?

MongoDB pushes the data to disk lazily. It updates the immediately written to the journal but writing the data from journal to disk happens lazily.

Explain the structure of ObjectID in MongoDB.

ObjectID is a 12-byte BSON type with:

4 bytes value representing seconds

3 byte machine identifier

2 byte process id

3 byte counter

MongoDB uses BSON to represent document structures. True or False?

True

If you remove a document from database, does MongoDB remove it from disk?

Yes. Removing a document from database removes it from disk too.

Mention the command to insert a document in a database called school and collection called persons.

use school;

db.persons.insert({ name: "kadhir", dept: "CSE" })

What are Indexes in MongoDB?

Indexes support the efficient execution of queries in MongoDB. Without indexes, MongoDB must perform a collection scan, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.

How many indexes does MongoDB create by default for a new collection?

By default, MongoDB created the _id collection for every collection.

Can you create an index on an array field in MongoDB? If yes, what happens in this case?

Yes. An array field can be indexed in MongoDB. In this case, MongoDB would index each value of the array.

What is a covered query in MongoDB?

A covered query is the one in which:

fields used in the query are part of an index used in the query, and

the fields returned in the results are in the same index

Why is a covered query important?

Since all the fields are covered in the index itself, MongoDB can match the query condition as well as return the result fields using the same index without looking inside the documents. Since indexes are stored in RAM or sequentially located on disk, such access is a lot faster.

Does MongoDB provide a facility to do text searches? How?

Yes. MongoDB supports creating text indexes to support text search inside string content. This was a new feature which can introduced in version 2.6.

What happens if an index does not fit into RAM?

If the indexes do not fit into RAM, MongoDB reads data from disk which is relatively very much slower than reading from RAM.

Mention the command to list all the indexes on a particular collection.

db.collection.getIndexes()

At what interval does MongoDB write updates to the disk?

By default configuration, MongoDB writes updates to the disk every 60 seconds. However, this is configurable with the commitIntervalMs and syncPeriodSecs options.

How can you achieve transaction and locking in MongoDB?

To achieve concepts of transaction and locking in MongoDB, we can use the nesting of documents, also called embedded documents. MongoDB supports atomic operations within a single document.

What is Aggregation in MongoDB?

Aggregations operations process data records and return computed results. Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the aggregation pipeline, the map-reduce function, and single purpose aggregation methods and commands.

What is Sharding in MongoDB? Explain.

Sharding is a method for storing data across multiple machines. MongoDB uses sharding to support deployments with very large data sets and high throughput operations.

What is Replication in MongoDB? Explain.

Replication is the process of synchronizing data across multiple servers. Replication provides redundancy and increases data availability. With multiple copies of data on different database servers, replication protects a database from the loss of a single server. Replication also allows you to recover from hardware failure and service interruptions.

What are Primary and Secondary Replica sets?

Primary and master nodes are the nodes that can accept writes. MongoDB's replication is 'single-master:' only one node can accept write operations at a time.

Secondary and slave nodes are read-only nodes that replicate from the primary.5

By default, MongoDB writes and reads data from both primary and secondary replica sets. True or False.

False. MongoDB writes data only to the primary replica set.

Why are MongoDB data files large in size?

MongoDB preallocates data files to reserve space and avoid file system fragmentation when you setup the server.

When should we embed one document within another in MongoDB?

You should consider embedding documents for:

'contains' relationships between entities

One-to-many relationships

Performance reasons

Why MongoDB is not preferred over a 32-bit system?

When running a 32-bit build of MongoDB, the total storage size for the server, including data and indexes, is 2 gigabytes. For this reason, do not deploy MongoDB to production on 32-bit machines.

If you're running a 64-bit build of MongoDB, there's virtually no limit to storage size.

What is a Storage Engine in MongoDB

A storage engine is the part of a database that is responsible for managing how data is stored on disk. For example, one storage engine might offer better performance for read-heavy workloads, and another might support a higher-throughput for write operations.

Which are the two storage engines used by MongoDB?

MongoDB uses MMAPv1 and WiredTiger.

What is the role of a profiler in MongoDB? Where does the writes all the data?

The database profiler collects fine grained data about MongoDB write operations, cursors, database commands on a running mongod instance. You can enable profiling on a per-database or per-instance basis.

The database profiler writes all the data it collects to the system.profile collection, which is a capped collection.

How does Journaling work in MongoDB?

When running with journaling, MongoDB stores and applies write operations in memory and in the on-disk journal before the changes are present in the data files on disk. Writes to the journal are atomic, ensuring the consistency of the on-disk journal files. With journaling enabled, MongoDB creates a journal subdirectory within the directory defined by dbPath, which is /data/db by default.

Mention the command to check whether you are on the master server or not.

db.isMaster()

Can you configure the cache size for MMAPv1? How?

No. MMAPv1 does not allow configuring the cache size.

Can you configure the cache size for WiredTiger? How?

For the WiredTiger storage engine, you can specify the maximum size of the cache that WiredTiger will use for all data. This can be done using storage.wiredTiger.engineConfig.cacheSizeGB option.

How does MongoDB provide concurrency?

MongoDB uses reader-writer locks that allow concurrent readers shared access to a resource, such as a database or collection, but give exclusive access to a single write operation.

How can you isolate your cursors from intervening with the write operations?

You can use the snapshot() method on a cursor to isolate the operation for a very specific case. snapshot() traverses the index on the _id field and guarantees that the query will return each document no more than once.

Can one MongoDB operation lock more than one databases? If yes, how?

Yes. Operations like copyDatabase(), repairDatabase(), etc. can lock more than onne databases involved.

How can concurrency affect replica sets primary?

In replication, when MongoDB writes to a collection on the primary, MongoDB also writes to the primary's oplog, which is a special collection in the local database. Therefore, MongoDB must lock both the collection's database and the local database.

What is GridFS?

GridFS is a specification for storing and retrieving files that exceed the BSON-document size limit of 16MB. Instead of storing a file in a single document, GridFS divides a file into parts, or chunks, and stores each of those chunks as a separate document.

Can you run multiple Javascript operations in a single mongod instance?

Yes. The V8 JavaScript engine added in 2.4 allows multiple JavaScript operations to run at the same time.

Which command can be used to provide various information on the query plans used by a MongoDB query?

The explain() command can be used for this information. The possible modes are: 'queryPlanner', 'executionStats', and 'allPlansExecution'.