**Mongodb:-**

open source nosql document oriented database which is written in C++ .

MongoDB doesn’t supports joins or transactions.

Mongo drivers  are typically written in their respective languages, although some drivers use C extensions for better performance .

High performance(After increasing load it maintain performance)

* If load increases (more storage space, more processing power), it can be distributed to other nodes across computer networks. This is called as *sharding*.
* MongoDB supports rich query to fetch data from the database.
* MongoDB supports replacing an entire document (database) or some specific fields with it's update() command.
* MongoDB supports Map/Reduce framework for the batch processing of data and aggregation operation. Here is brief of how Map/Reduce works :
* Map : A master node takes an input. Splits it into smaller sections. Sends it to the associated nodes.
* These nodes may perform the same operation in turn to send those smaller section of input to other nodes. It processes the problem (taken as input) and sends it back to the Master Node.
* Reduce : The master node aggregates those results to find the output.
* *GridFS* specification of MongoDB supports storage of very large files.

**key aspects:**

1. Data Model:- MongoDB stores data in the form of BSON -Binary encoded JSON documents which supports a rich collection of types. Fields in BSON documents may hold arrays of values or embedded documents.
2. GridFS:- This 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, and stores each part as a separate document. GridFS uses two collections to store files.

When we query a GridFS store for a file, the client reassembles the chunks as needed. Information can also be accessed from any random section/s of files. This feature is what basically allows for “skipping” into the middle of a video or audio file.

1. Sharding:- Database systems with large data sets and high throughput applications can challenge the capacity of a single server in multiple ways such as:

High query rates put stress on the CPU capacity of the server.

Larger data sets exceed the storage capacity of a single machine.

Dataset sizes larger than the system’s RAM stress the I/O capacity of disk drives.

To address these issues of scale, database systems have two basic approaches:

**Vertical Scaling**

**Sharding or Horizontal Scaling**

a) Vertical scaling: It adds more CPU and storage resources to increase capacity. But such arrangements are disproportionately expensive. As a result there is a practical maximum capability for vertical scaling.

b) Sharding or Horizontal Scaling: By contrast, it divides the data set and distributes the data over multiple servers-shards. Each shard is an independent database and collectively shards make up a single database.

MongoDB supports sharding through the configuration of sharded clusters. Process of sharing has been explained in the image below where:

Shards are used to store the data.

Query Routers, or mongos instances, interface with client applications and direct operations to the appropriate shard or shards and then returns results to the clients.

Config servers stores the cluster’s metadata. This data contains a mapping of the cluster’s data set to the shards. The query router uses this metadata to target operations to specific shards.

1. Aggregation
2. Data partitioning:-
3. Indexes
4. Replication

**Refrence:-** <https://www.analyticsvidhya.com/blog/2015/06/beginners-guide-mongodb/>

**What is a ‘document’ in Mongo DB?**

A record in MongoDB is a document (shown below), which is a data structure composed of field and value pairs. MongoDB documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents.This is an important differentiation from RDBMS systems where each field must contain only one value.