# MongoDB - Sharding

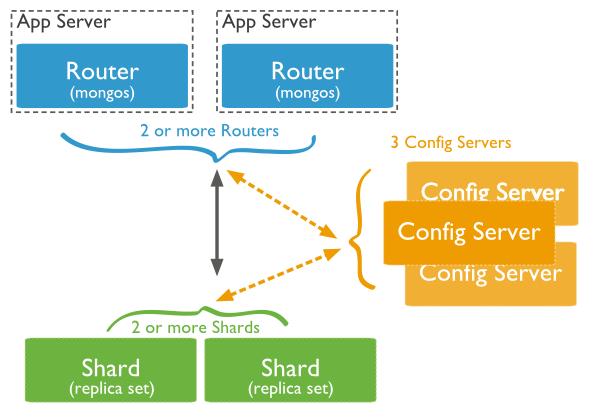
Sharding is the process of storing data records across multiple machines and it is MongoDB's approach to meeting the demands of data growth. As the size of the data increases, a single machine may not be sufficient to store the data nor provide an acceptable read and write throughput. Sharding solves the problem with horizontal scaling. With sharding, you add more machines to support data growth and the demands of read and write operations.

## Why Sharding?

* In replication, all writes go to master node
* Latency sensitive queries still go to master
* Single replica set has limitation of 12 nodes
* Memory can't be large enough when active dataset is big
* Local disk is not big enough
* Vertical scaling is too expensive

## Sharding in MongoDB

The following diagram shows the sharding in MongoDB using sharded cluster.



In the following diagram, there are three main components −

* **Shards** − Shards are used to store data. They provide high availability and data consistency. In production environment, each shard is a separate replica set.
* **Config Servers** − Config servers store 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. In production environment, sharded clusters have exactly 3 config servers.
* **Query Routers** − Query routers are basically mongo instances, interface with client applications and direct operations to the appropriate shard. The query router processes and targets the operations to shards and then returns results to the clients. A sharded cluster can contain more than one query router to divide the client request load. A client sends requests to one query router. Generally, a sharded cluster have many query routers.

**MongoDB Sharded Cluster - Step by Step Implementation**

Sharding is a concept in MongoDB, which splits large data sets into small data sets across multiple MongoDB instances.

Sometimes the data within MongoDB will be so huge, that queries against such big data sets can cause a lot of CPU utilization on the server. To tackle this situation, MongoDB has a concept of Sharding, which is basically the splitting of data sets across multiple MongoDB instances.

The collection which could be large in size is actually split across multiple collections or Shards as they are called. Logically all the shards work as one collection. 

**Implementing Sharding**

Shards are implemented by using clusters which are nothing but a group of MongoDB instances.

The components of a Shard include

1. **A Shard** – This is the basic thing, and this is nothing but a MongoDB instance which holds the subset of the data. In production environments, all shards need to be part of replica sets.
2. **Config server** – This is a mongodb instance which holds metadata about the cluster, basically information about the various mongodb instances which will hold the shard data.
3. **A Router** – This is a mongodb instance which basically is responsible to re-directing the commands send by the client to the right servers.

**Step 1)** Create a separate database for the config server.

mkdir /data/configdb

**Step 2)** Start the mongodb instance in configuration mode. Suppose if we have a server named Server D which would be our configuration server, we would need to run the below command to configure the server as a configuration server.

mongod –configdb ServerD: 27019

**Step 3)** Start the mongos instance by specifying the configuration server

mongos –configdb ServerD: 27019

**Step 4)** From the mongo shell connect to the mongo's instance

mongo –host ServerD –port 27017

**Step 5)** If you have Server A and Server B which needs to be added to the cluster, issue the below commands

sh.addShard("ServerA:27017")

sh.addShard("ServerB:27017")

**Step 6)** Enable sharding for the database. So if we need to shard the Employeedb database, issue the below command

sh.enableSharding(Employeedb)

**Step 7)** Enable sharding for the collection. So if we need to shard the Employee collection, issue the below command

Sh.sharedCollection("db.Employee" , { "Employeeid" : 1 , "EmployeeName" : 1})

**Summary:**

* Sharding is a concept in MongoDB, which splits large data sets into small data sets across multiple MongoDB instances.