MongoDB Sharding fundamentals

What is sharding?

- A mechanism for horizontal scaling

- Distributes the dataset over multiple servers (shards)

- Each shard is an independent database

- All shards consists a single logical database

Why Sharding?

- Increases cluster throughput – Read/Write Scaling
- Reduces costs - Many small servers VS one big box
- Eliminates HW and SW hard limits

MongoDB Sharding

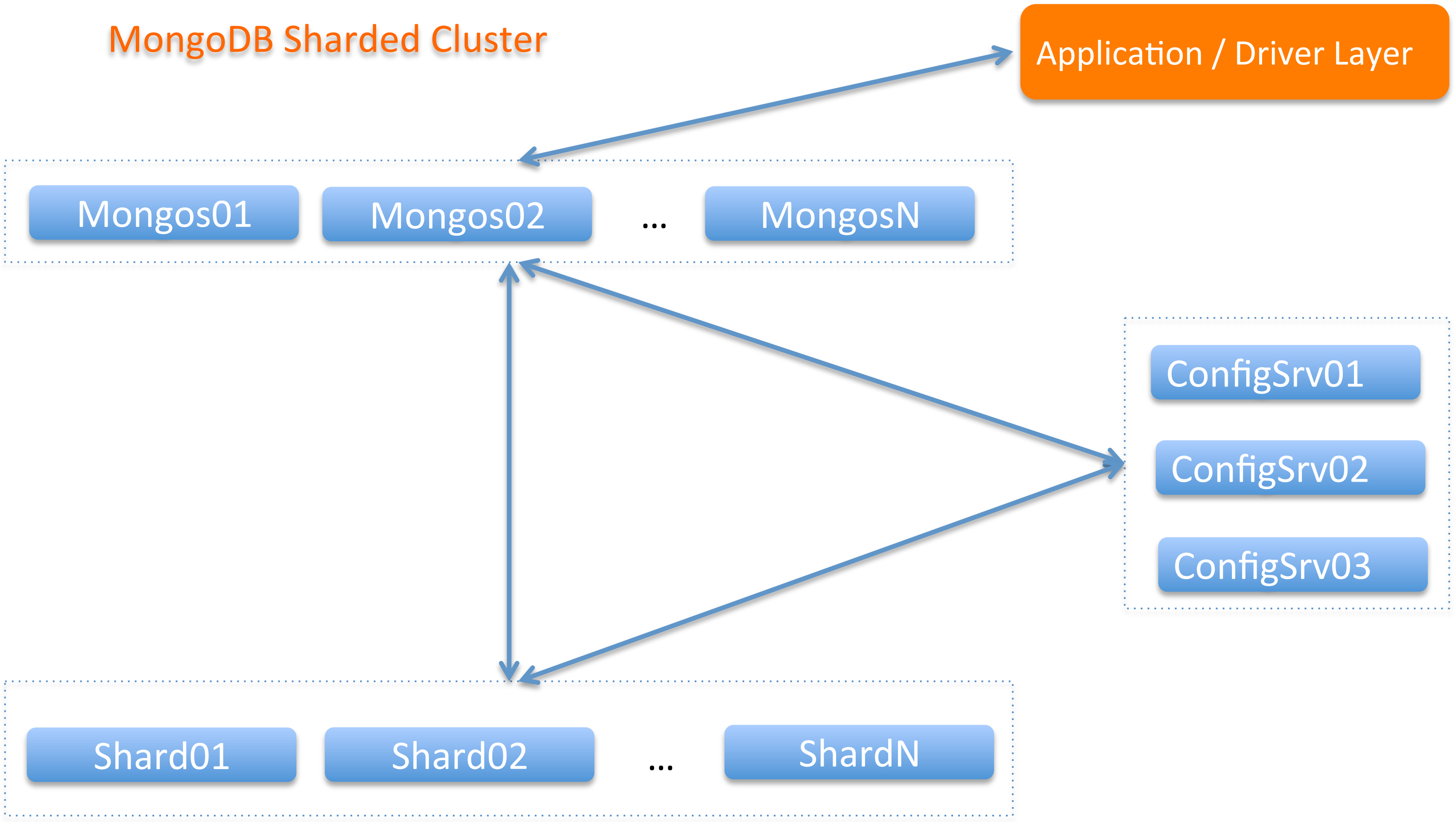
- Consists of three elements: Shards, Config Servers and Mongos

- Shards: Hold the cluster data, databases, collections, documents (Data nodes)

- Config Servers: Hold the cluster metadata, map the cluster architecture.

- Mongos: Serve all drivers requests. Route each request to a shard or shards (Router nodes)

MongoDB Sharded Cluster



How Sharding works?

- Range partitioning per collection (chunks)
- Shard key to define chunks (field(s))
- Chunks are “metadata” on the config servers
- Chunks can move, split and merge

How Sharding works? - Example

```
{ "name" : "Angelina", "surname" : "Jolie", "position" : "Windows Eng.", "phone" : "555-5555" }
{ "name" : "Emma", "surname" : "Stone", "position" : "Windows Eng.", "phone" : "555-5555" }
{ "name" : "Charlize", "surname" : "Theron", "position" : "Linux Eng.", "phone" : "555-5555" }
{ "name" : "Olivia", "surname" : "Wilde", "position" : "Linux Eng.", "phone" : "555-5555" }
{ "name" : "Jessica", "surname" : "Alba", "position" : "Sr Linux Eng.", "phone" : "555-5555" }
{ "name" : "Scarlett", "surname" : "Johansson", "position" : "Sr Windows Eng.", "phone" : "555-5555" }
{ "name" : "Megan", "surname" : "Fox", "position" : "Networks Eng.", "phone" : "555-5555" }
{ "name" : "Mila", "surname" : "Kunis", "position" : "Sr Networks Eng.", "phone" : "555-5555" }
{ "name" : "Natalie", "surname" : "Portman", "position" : "Database Eng", "phone" : "555-5555" }
{ "name" : "Anne", "surname" : "Hathaway", "position" : "Sr Database Eng", "phone" : "555-5555" }
```

- Collection employees for an IT company
- Shard key “**position**”

How Sharding works? - Example

{ "min" : { "position" : { "$minKey" : 1 } }, "max" : { "position" : "Database Eng" }, "shard" : ”Shard01" }

{ "min" : { "position" : "Database Eng" }, "max" : { "position" : "Sr Database Eng" }, "shard" : "Shard01" }

{ "min" : { "position" : "Sr Database Eng" }, "max" : { "position" : "Windows Eng." }, "shard" : "Shard02" }

{ "min" : { "position" : "Windows Eng." }, "max" : { "position" : { "$maxKey" : 1 } }, "shard" : "Shard02" }

- Lower/upper bound and shard (server)

Choose a shard key

- High Cardinality
- Not Null values
- Immutable field(s)
- Not Monotonically increased fields

Choose a shard key

- Even read/write distribution

- Even data distribution

- Read targeting

- Read locality

Choose a shard key

- Hashed shard keys for randomness
- Compound shard keys for cardinality
- Unique indexes are good
- {_id:”hashed”} scales writes

Limitations of Sharding

- Unique indexes – Just one…

- Initial collection size – Avoid collections > 256G, hard limit is a function of key and chunk size , for 64MB chunk/512B key is more than 1TB

- Number of documents per chunk (250K)

Limitations of Sharding

- Shard key size < 512 bytes

- Multikey,text, geo indexes are prohibited

- Some operations won’t run (for example group, db.eval(), $isolated, $snapshot, geoSearch)

“Sharding” – Other players

- Application level sharding
- Mysql (MaxScale, Fabric,…)
- Postgres (pg_shard)
- ElasticSearch (Document ID or routing)
- Cassandra (Hash-based - Ring topology)