How Discord Indexes Billions of msgs?

Subscribe YouTube channel - 'MsDeep Singh' for more such content.

Requirements

① Cost Effective - Search is accessary feature so price should not exceed actual storage.

② Fast & Intuitive

③ Self healing

④ Linearly scalable

⑤ Lazily Indexed - don't index msgs unless someone attempts to search them atleast once.

Finalized Solution - Elastic Search

Why?

① No extra setup required for Node discovery. Solr requires Zookeep

② ES supports automatic shard rebalancing on addition/removal of nodes.

③ Structured query DSL built in.

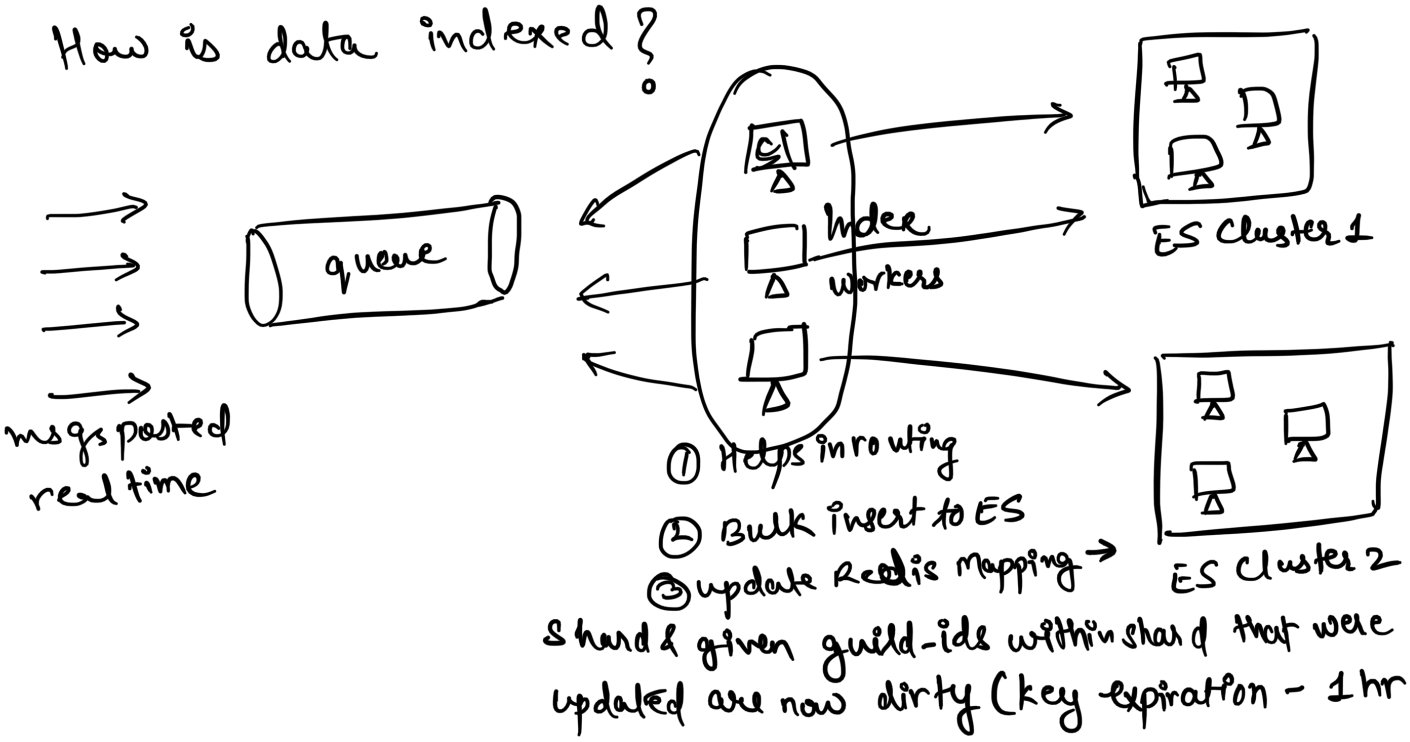
④ Previous work experience with ES.

How can you avoid large ES clusters as msg data grows?

① Delegate sharding and routing logic to application layer - this will help to index msgs into pool of smaller ES clusters.

② Multiple clusters also help to reduce blast radius.

How is data indexed?



Mapping of Discord Server's message to ES cluster + index

It comes in two layers -

① Persistent shard mapping - put in Cassandra.

② Shard mapping cache - while ingesting msgs on workers, querying Cassandra for shard is slow operation. So cache these mappings in Redis.

{ES cluster + index}

Don't confuse with ES shard.

Select shard to Discord server's messages (when indexed first time)

Shard Allocator : ① Use sorted set in Redis, keep set of shards with score that represent their load.

② The shard with lowest load should be allocated next.

③ Score gets incremented on every new allocation.

Discovery of ES clusters and hosts within them –

etcd is used. – nodes in ES cluster can announce themselves onto etcd.

Indexing and mapping the Data

① ES cluster → consists of indices → contains no. of shards within it

Shard is lucene index.

② ES holds responsibility of data distribution within index to a shard belonging to that index.

↳ Discord has its own sharding logic in application layer (as described above in document)

(a) index contains one shard (no sharding required)

(b) index replication to one node.

(c) refresh interval - 60 minutes

(d) index contains single document type - message.

→ msg is transformed into bunch of fields containing metadata → index & search on this.

index template snippet → refer on discord blog

→ Message data is not duplicated in ES cluster, actual msg is fetched from Cassandra.

↳ saves additional disk space.

↳ to highlight text matches, tokenizers and language analyzers are built into client application.

→ No separate search service. Created a library as a wrapper of routing and querying logic to ES cluster.

Why refresh interval is 60 minutes and not default 1sec

(1) 1sec refresh interval provides near time search ability.

(2) So every second for across the indices, ES flushes the in-memory buffer to Lucene Segment and opening the segment to make it searchable.

↳ It will require high CPU.

↳ More disk usage

↳ At less load time, ES merges massive amount of tiny segments into larger segments [which are space efficient]

Search Life cycle

① look up shard that needs to queried for guild-id

② Check Redis - if shard & guild-id is dirty.

③ If dirty, do a refresh of shard's ES index and mark entire shard as clean.

④ Execute search query & return results.

This helps in identifying if index should be refreshed. → since it is set to 1hr but if refresh is required prior to that for specific discord's server, it can be achieved.

what is guild-id - Helps in scoping a serach to given server.

Guid - represents collection of users and channels. From client perspective, guild means server.

How does search query looks like?

```
SearchQuery
( guild_id = 12345678989,
  content = "Hey Jake",
  channel_ids = [1667089, 38225979]
)
```

Happy Learning 😊

subscribe for more

YouTube - MsDeep Singh