NoSQL Datastores

UCSB Nov 4th 2014

Josep M. Blanquer

blanquer@gmail.com

Goals and Agenda

"Broad overview of NoSQL technologies"

- Intro
- 100K feet introduction to 5 NoSQL DBs
- Deeper dive, in terms of:
  - Storage
  - Interface
  - System architecture
  - Storage architecture

Intro

"Um…and who are you again?"

- **Academia**
  - BSCS: Multi-process scheduling
  - MSCS: Networking
  - Ph.D: Scalable Web services
- **Industry**
  - Manager Internet Service Provider
  - Research @Bell-Labs
  - Advanced development @ Citrix
  - Chief Architect @ RightScale

NoSQL technologies under study:

Cassandra

MongoDB

Redis

Kafka

Elasticsearch

Preface: Cassandra

Apache Cassandra is an ***open source, distributed database*** *management system designed to* ***handle large amounts of data*** *across* ***many commodity servers, providing high availability*** *with* ***no single point of failure****. Cassandra offers robust support for* ***clusters spanning multiple datacenters,*** *with* ***asynchronous masterless replication*** *allowing low latency operations for all clients.*

Preface: MongoDB

MongoDB (from "humongous") is a cross-platform ***document-oriented database****. Classified as a NoSQL database, MongoDB eschews the traditional table-based relational database structure in favor of* ***JSON-like documents*** *with* ***dynamic schemas*** *(MongoDB calls the format BSON), making the integration of data in certain types of applications easier and faster.*

Preface: Redis

*Redis is an open source, BSD licensed**, advanced key-value cache and store**. It is often referred to as **a data structure server** since keys can contain **strings, hashes, lists, sets, sorted sets, bitmaps and hyperloglogs**.*

Redis: Remote DIctionary Server

Preface: Kafka

*Apache Kafka is an open-source **message broker** project developed by the Apache Software Foundation written in Scala. The project aims to provide a **unified, high-throughput, low-latency platform** for **handling real-time data feeds**. The design is heavily influenced by transaction logs.*

Preface: Elasticsearch

elasticsearch.

*Elasticsearch is a **search server** based on **Lucene**. It provides a **distributed, multitenant-capable full-text search engine** with a **RESTful** web interface and **schema-free JSON documents.***

Deeper Dive

Cassandra

- Most similar to a RDBMS
- From a tabular and querying stance
- No relationships or joins possible
- Designed for large datasets
- Similar to:
  - HBase (Google)
  - DynamoDB (AWS)

Cassandra: Storage

- Has tables called ColumnFamilies:
  - like distributed Hashes
- Each table (ColumnFamily) has rows:
  - Value of a row is like a “big” Hash
  - Values per row can be schemaless
  - Very long rows (many values / row)

Users ColumnFamily:

row key	columns …			
jbellis	name	email	address	state
	jonathan	jb@ds.com	123 main	TX
dhutch	name	email	address	state
	daria	dh@ds.com	45 2nd St.	CA
egilmore	name	email		
	eric	eg@ds.com		

Cassandra: Storage

- Static Column Family

  Example: Users

row key	columns …			
jbellis	name	email	address	state
	jonathan	jb@ds.com	123 main	TX
dhutch	name	email	address	state
	daria	dh@ds.com	45 2nd St.	CA
egilmore	name	email		
	eric	eg@ds.com		

Known column names

- Dynamic Column Family

  Example: friends

row key	columns …			
jbellis	dhutch	egilmore	datastax	mzcassie
dhutch	egilmore			
egilmore	datastax	mzcassie		

Dynamic column names

Cassandra: Interface

- CQL interface, similar to SQL
- But very limited features
  - Mostly key value queries
  - Can create 2ndary indexes on column values
  - Fixed sort order within row
- No transactions or joins:
  - Batch atomicity possible
  - High amount of denormalization
- Can control consistency per statement

- <quick look at CQL slides>

Cassandra: System arch.

- Distributed and Highly Available
- Masterless system (all are coordinators)
- Data automatically split across nodes
- Each datum can be replicated N times
- Supports datacenter groups
- Reads are eventually consistent:
  - But it can be made strictly consistent

- <peek at architecture slides>

Cassandra: Storage arch.

- Memtables
- Commitlog
- SSTables

- Corollary:
  - Writes: really, really fast
    - *Delayed compactions and repairs*
  - Reads: fast
    - *But merges are performed on the fly*

MongoDB

- A document DB
- No intra-document relationships
- But some nesting and joins possible

MongoDB: Storage

- Has “tables” called document collections
- Each collection has JSON documents
  - Stored JSONB format
- A document
  - Is a JSON hash hash with (key: _id)
  - Schema does not need to be defined
  - Can nest other JSON documents
  - Can also have references to others
    - But app needs to do the “join”

MongoDB: Interface

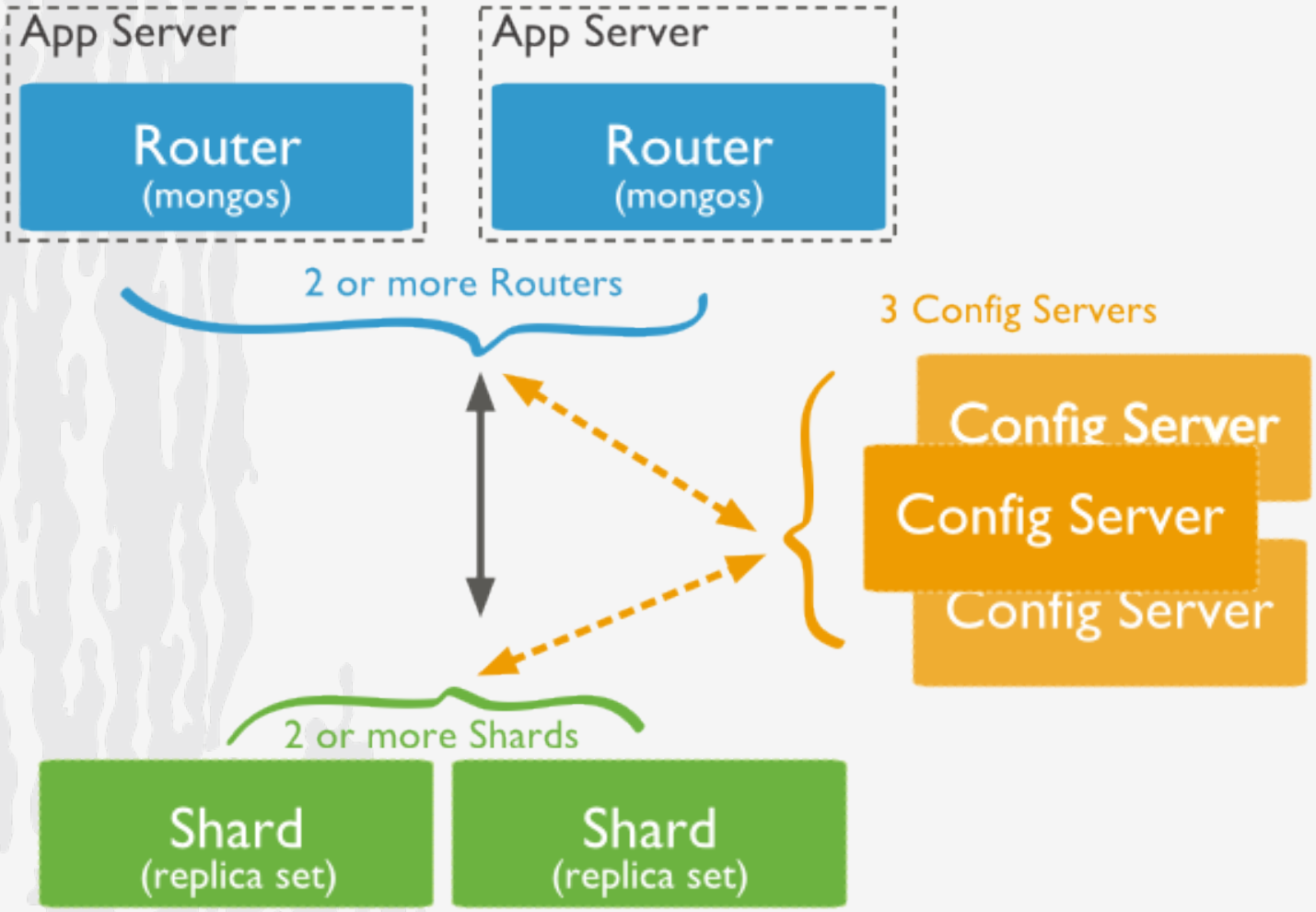
- Query by primary key: doc id
- Or by any indexed document fields
- No transactions or joins
- Consistent reads and atomic updates
- Custom query language

```
db.users.insert (          <-- collection
   {
     name: "sue",          <-- field: value
     age: 26,              <-- field: value
     status: "A"           <-- field: value
   }                       } document
)
```

MongoDB: System arch.

- Master-slave system (replica sets)
- Collections can be sharded (see next slide)
  - Each shard can have a replica set
  - Mongos will know what shard to access
  - Config servers map data to shards
- Consistency:
  - Strict when reading from primary
  - Eventual when reading from replicas

MongoDB: System arch.



MongoDB: Storage arch.

- Journaled writes
- Custom memory-mapped files
- Both for documents and indexes

Redis

- An in-memory DB
- Supports many datastructures
- Also supports pub/sub

Redis: Storage

- Native structures
  - Lists
  - (sorted)Sets
  - Hashes
  - Bitmaps
  - HyperLogLogs

Redis: Interface

- Basically the same you'd do in memory
  - Hashes by key, Lists by index
  - Top-K elem of sorted Sets
  - Push/pop
- Also subscribe/publish to topic by name
- Transactions are available
- Very simple TEXT protocol

Redis: System arch.

- Supports master-slave system
- Replication: Sentinel as of 2.8
  - It monitors the master/slaves
  - Can automatically promote and resync
  - Or can send a message to do that

Redis: Storage arch.

- All in-memory structures
- Possibility of persisting to Disk
  - RDB: Redis Database File
    - Forks and saves a full dump
  - AOF: Append-Only File
    - Saves updates to a log
    - Log is replayed upon start

Kafka

- Not similar to a DB at all
- It is a persistent queue/bus
- Very high-throughput
- Uses Zookeeper to coordinate:
  - Distributed kafka brokers
  - Consumer/ConsumerGroups offsets

Kafka: Storage

- Has topics (i.e., named queues)
- Topics have partitions (for parallelism)
  - fully ordered sequence of messages
  - Immutable sequence (append only)

Kafka: Interface

- Push N messages to topic/partition
  - Can control the consistency
  - Batching possible / async
  - Consumer-groups
- Read messages for topic/partition
  - Can start from offset O
  - Batching possible
- Binary TCP protocol
  - complex clients: need to use ZooKeeper

Kafka: System arch.

- Distributed system (multiple brokers)
- Partitions are split across brokers
- Each partition has a master (+slaves)
- Each message can be replicated N times
- Uses Zookeeper to manage cluster
  - Connected brokers to zk nodes
  - Notifications when brokers disconnect
  - Spreading and rebalancing partitions

Kafka: Storage arch.

- Partitions of topics stored in flat files
- Plus some metadata to map files o each topic/segment

- Corollary:
  - Writes: really, really fast (file write)
  - Reads: really really fast (sendfile)

Elasticsearch

elasticsearch.

- Deep JSON document indexing
- Geared towards flexible indexing terms
- And to perform analytics on data
- Multi-tenant

Elasticsearch: Storage

elasticsearch.

- Uses Lucene indexes underneath.

Elasticsearch: Interface

elasticsearch.

- Powerful search (incl. Lucene query)
- Full-text index, highlighting, more like this…
- Powerful analytics, faceting
- Percolator
- REST interface

- Query examples

Elasticsearch: System arch.

elasticsearch.

- Distributed nodes
- Data split across shards
- Shards can be replicated N times
- Uses Zen discovery to organize cluster

Elasticsearch: Storage arch.

elasticsearch.

- Partitions data into shards
- Each shard has a Lucene index
- Indexes are flushed periodically

NoSQL technologies

- Cassandra
- MongoDB
- Redis
- Kafka
- Elasticsearch

That’s a wrap!

Questions?