NoSQL Implementations

Let's take a quick look at two popular NoSQL databases

MongoDB

MongoDB is a NoSQL “Document” database.

- Stores collections of documents in a key:value pair format
- What MongoDB is ***NOT***:
    - It is not relational
    - It does not store data in tables
    - It does not use the SQL query language

MongoDB

What MongoDB ***IS*** :

- MongoDB uses a JavaScript-like query language
- Community edition is free, open source
- Enterprise edition users can purchase support, advanced features, cloud deployment
- Very fast, very scalable
- The most widely used NoSQL database

MongoDB

MongoDB Concepts

- A MongoDB DATABASE can contain one or more COLLECTIONS of documents
- A MongoDB collection can contain many individual DOCUMENTS
    - Each **document** has a primary key
    - Primary keys (and any other field) can be indexed for faster performance

MongoDB

MongoDB Concepts

- Documents are stored in JSON format (semi-structured)

  "Java Script Object Notation"

- JSON grew out of the Java Script programming language
- JSON is gradually replacing XML
  - "Extended MarkUp Language"
  - XML is more verbose; JSON is easier for humans to work with

MongoDB

MongoDB Documents

- Documents are “polymorphic” (i.e. “multiple shapes”…)
  - Not all documents in a collection must have all the same key:value pairs.
- There is no need to declare the structure of a document
  - It is “self-describing”
- The database stores/retrieves the JSON very efficiently
- You can add fields to one document in a collection without modifying any other documents in the same collection

MongoDB

MongoDB relies on REPLICATION and SHARDING

- MongoDB provides horizontal scaling
- You can configure a scalable number of nodes in a cluster
- A cluster can be spread across geographically separate data centers
- MongoDB can easily scale READ operations across the cluster (parallelization)
- MongoDB has automated node failure detection and failover

MongoDB

MongoDB does Primary-to-Secondary Replication

- If the Primary node fails for any reason, the other member nodes vote to elect a new primary from among the secondary nodes
- All WRITES go to the Primary and are replicated to the secondary replica nodes
- Replication is asynchoronous (the primary does not wait for an acknowlegement from the replica)
- The number of replicas is configurable

MongoDB

No Database Downtime for Upgrades

- Administrators can take a node offline line, upgrade it, and put it back online
- Replication/Redundancy allows work to continue even if a node is offline for maintenance – there are multiple copies of the same data

MongoDB

Quick MongoDB Demo

Cassandra

Cassandra is a popular NoSQL database

- Stores data in a key:value pair format within a wide-row, column-family structure
- What Cassandra is ***NOT***:
    - It is not relational
    - It does not store data in rigidly structured tables
    - It does not use the SQL query language
    - (But it DOES use a similar "CQL", Cassandra Query Language)

Cassandra

What Cassandra ***IS*** :

- Massively scalable, free, open source
  - Like MongoDB , the Community Edition is free
  - Enterprise Cassandra users can purchase support and add-on features through a vendor like DataStax
- Designed for High Performance and High Scalability
- Designed for High Availability, fault tolerant with no SPOF

Cassandra

Cassandra's Heritage. Based on:

- Google Big Table which is the core foundation for many Google services and is the foundation for Cassandra's internal storage model
- Amazon Dynamo: Which supports many of Amazon's core services and is the foundation for Cassandra's distributed backbone
- Facebook -- which developed and open-sourced Cassandra

Cassandra

Cassandra

Provides the benefits of these technologies, but, It improved them for Cassandra's needs

Cassandra's database model is a partitioned row store (with roots in BigTable)

Uses a Peer-to-Peer distributed architecture (roots in Dynamo)

Cassandra

Cassandra Concepts

Although not really stored in a "table", data in Cassandra is

- **Row-oriented:** Each row is an aggregate with column families representing meaningful, related chunks of data within that aggregate.

- **Column-oriented:** Each column family defines a record with sets of related data. You can think of a row as a collection of related records across all column families.

Cassandra

Cassandra is a peer-to-peer, fully distributed system where

- All nodes are equal (no primary/secondary)
- Data is partitioned (replicated & sharded) among multiple nodes in a cluster
- Sharding and replication are configurable
- No node is a single point of failure (SPOF)
- Any node may be read from or written to

Cassandra

A Cassandra "Instance" = A collection of independent nodes

- Configured into a cluster
- All nodes are peers (i.e. they all serve the same function)
- Data is distributed across all nodes in a cluster
- All nodes store data and service client compute requests
- A client may read/write to any node, which becomes the coordinator for servicing that particular request
- Nodes can have different capacity/resources available (e.g. memory, CPU, disk)
- Cassandra distributes data and query workload based on the available resources

Cassandra

Cassandra is very configurable

- The number of replicas of each partition
- The level of consistency
  - Can be configured for updates to wait for an acknowledgement or not
  - "Eventual consistency"

Cassandra

Quick Cassandra Demo