



Enterprise Architecture

Persistence



Scaling Database

Use case: Cassandra



What are the trade offs of persistence/DB ?



Speed vs durability

- Pure in-memory, no persistence at all, such as memcached or Scalaris
- In-memory with periodic snapshots, such as Oracle Coherence or Redis
- Disk-based with update-in-place writes, such as MySQL ISAM or MongoDB
- Commitlog-based, such as all traditional OLTP databases (Oracle, SQL Server, etc.)



Data Model ?

key-value

Amazon
DynamoDB (Beta)

ORACLE
BERKELEY DB 11g

 redis

graph

 Neo4j
the graph database

 InfiniteGraph


sones

column


HBASE


 riak


Cassandra

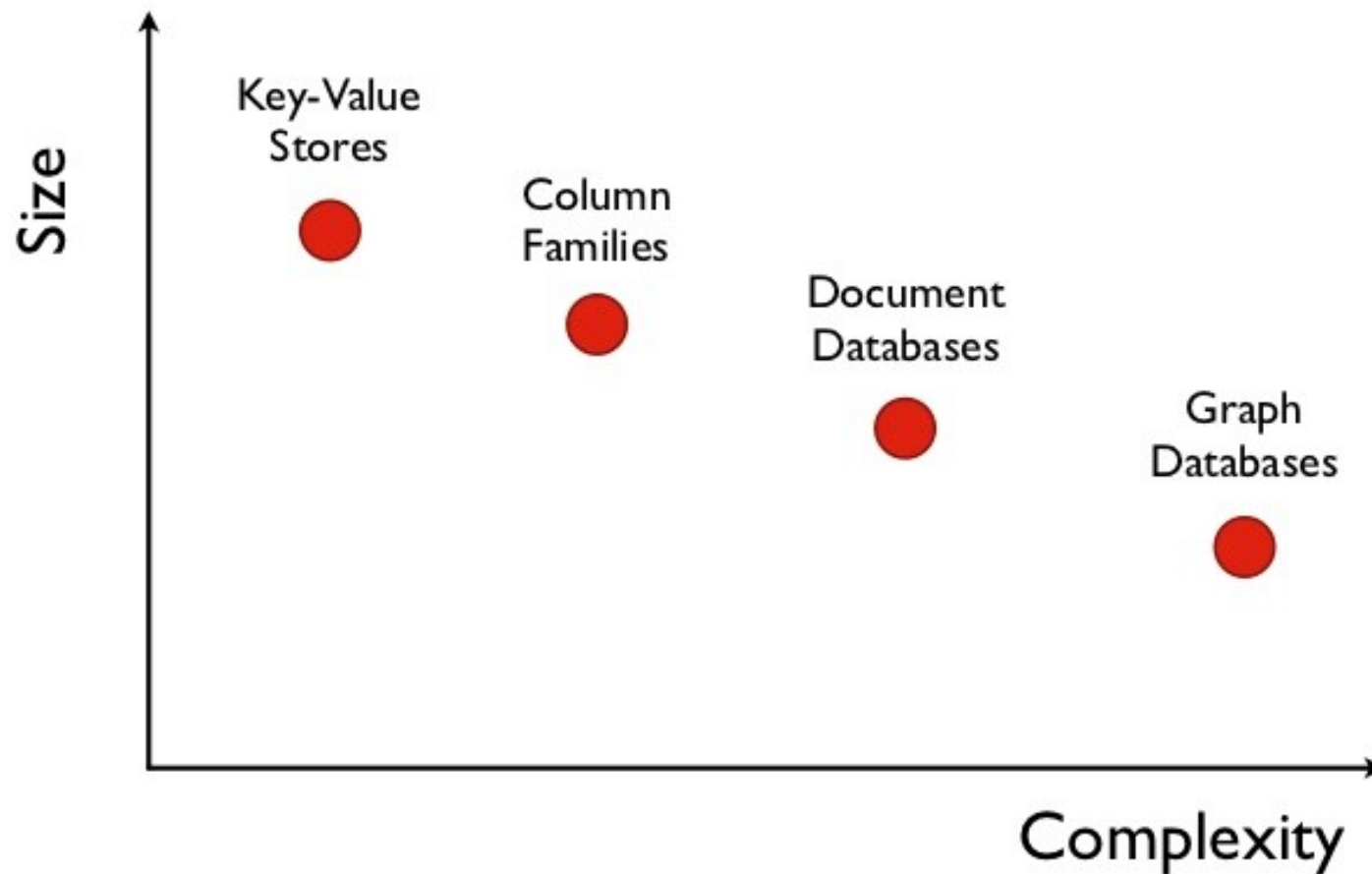
document


CouchDB
relax

 mongoDB


terrastore

Focus Of Different Data Models





Trade offs

- Read
 - <https://www.datastax.com/blog/2010/10/what-persistence-and-why-does-it-matter>



Cassandra

- Claim
 - A massively scalable open source NoSQL database.
 - Distributed across multiple data centers with no single point of failure.
 - Low latency queries



Cassandra

- Claim
 - A massively scalable open source NoSQL database.
 - Distributed across multiple data centers with no single point of failure.
 - Low latency queries

How would you build such system ?



Cassandra

- We can't offer ACID RDBMS constraints
 - Two phase commit is anti-availability
- We need to compromise :(

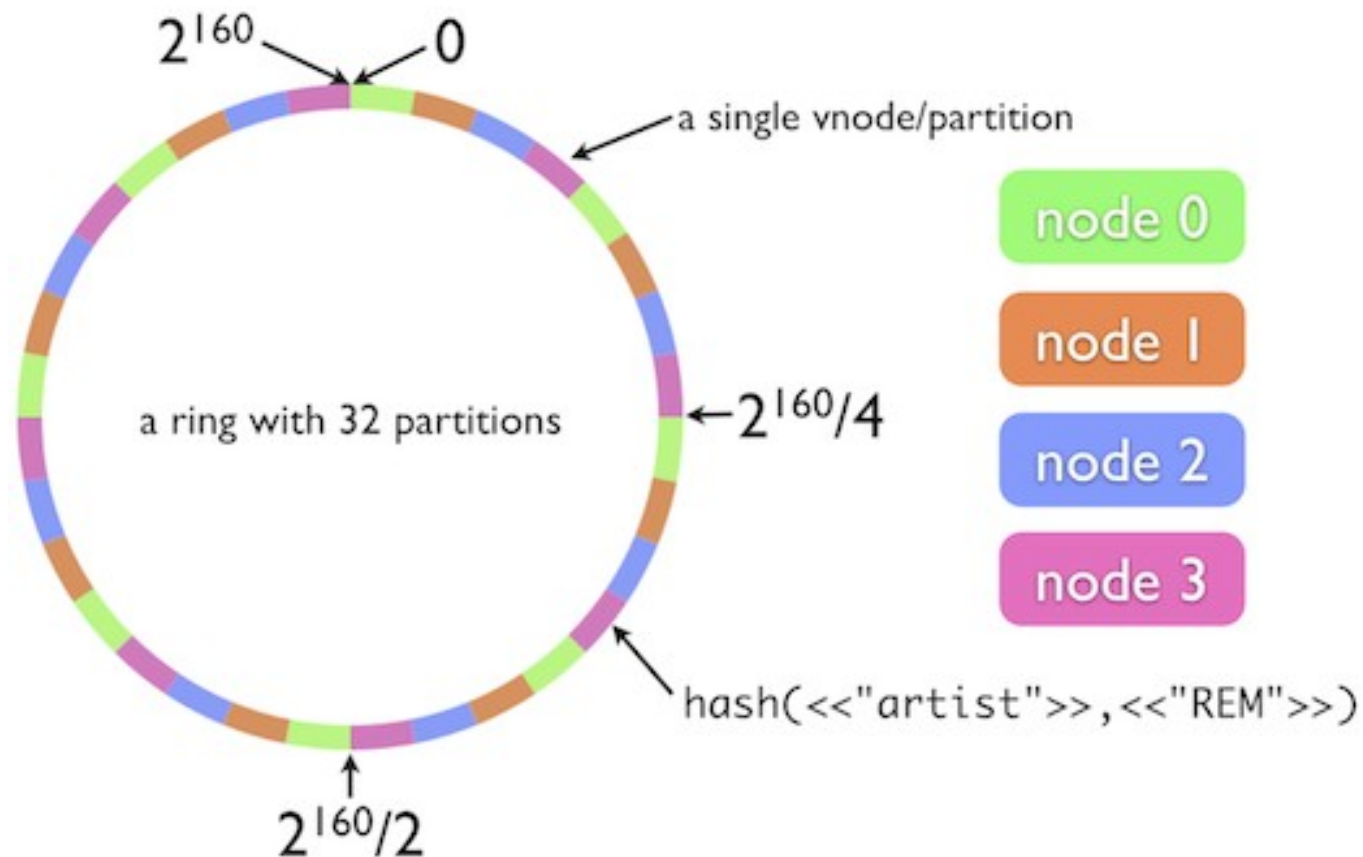
Sharding

- A developer can select an attribute and shard the table between different nodes
 - e.g. shard users table by country code

Do you see the
problem ?

- Shared Nothing Architecture, scales well!
- Check Pros and Cons:
 - [https://en.wikipedia.org/wiki/Shard_\(database_architecture\)](https://en.wikipedia.org/wiki/Shard_(database_architecture))

Consistent Hashing



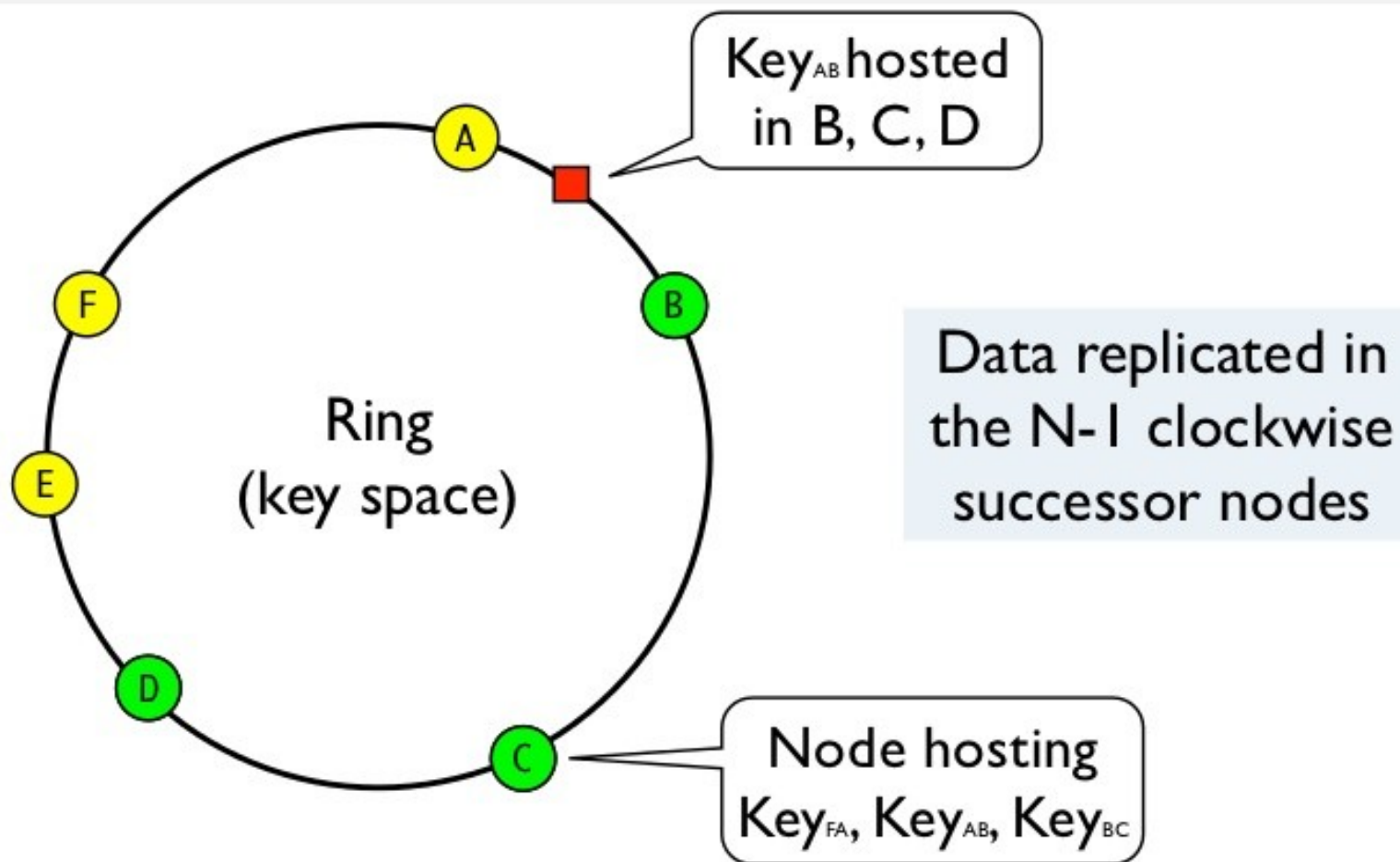
<https://dzone.com/articles/simple-magic-consistent>



Replication

- Storing copies of data on multiple nodes
- Replication of data ensures fault tolerance and reliability.

Consistent Hashing - Replication

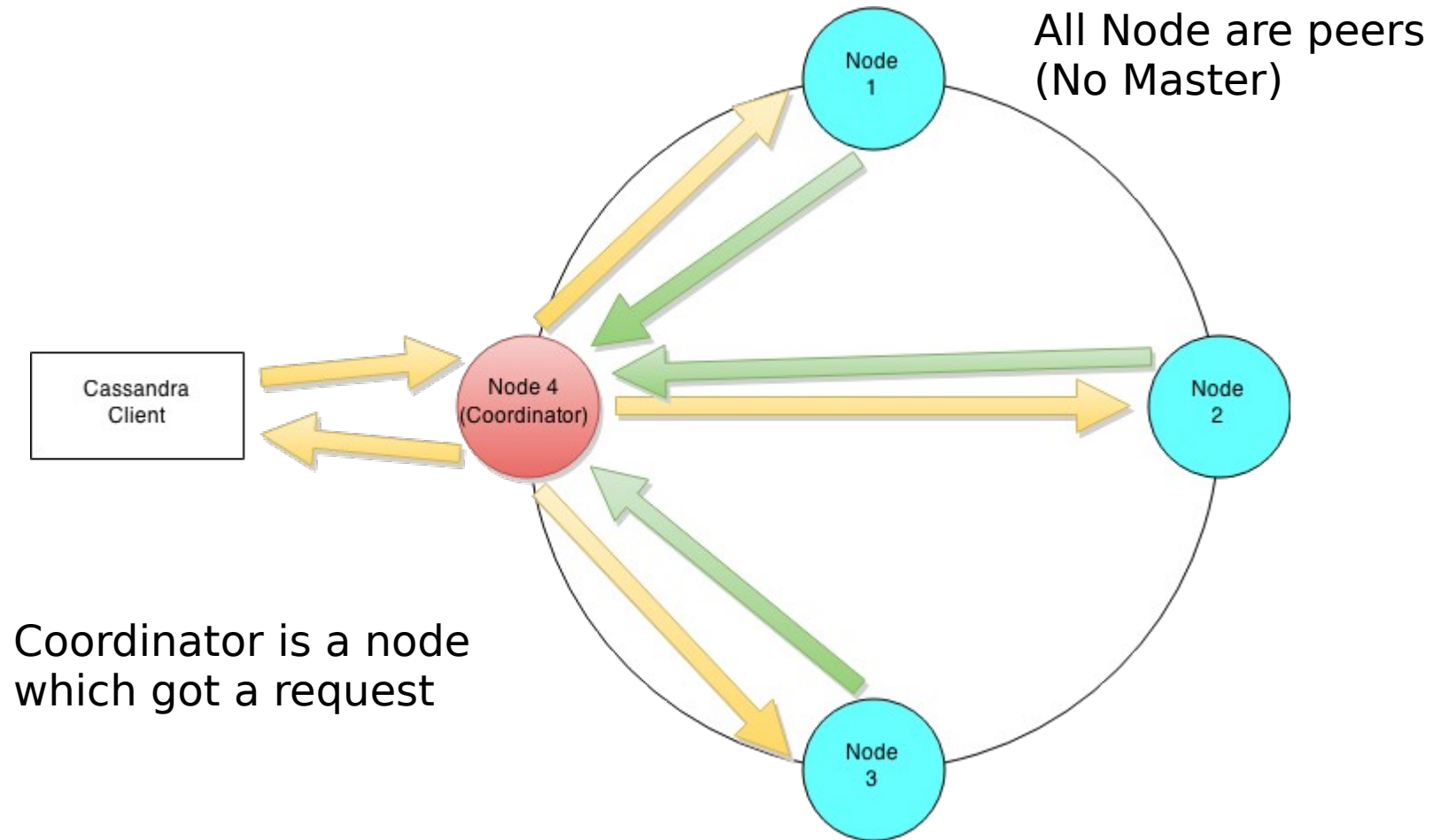




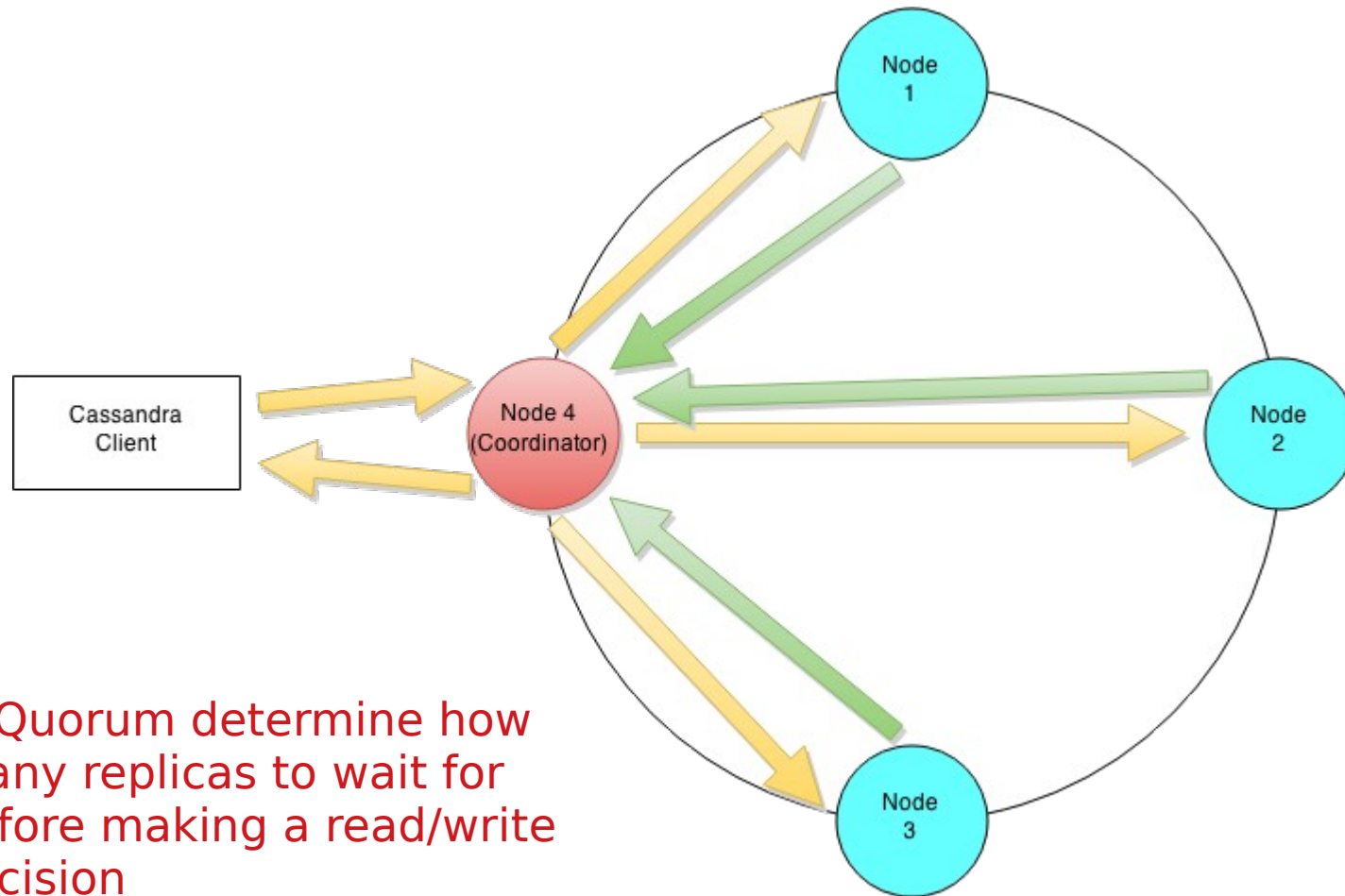
Cassandra

- Eventual Consistency
 - Typically it's AP system (remember CAP?)
- Tunable Consistency by setting quorum level
 - From "writes never fail" → A
 - to "block for all replicas to be readable" → C

Coordination



Coordination





How to choose

- Watch:
 - How to Choose the Right Database? - MongoDB, Cassandra, MySQL, HBase | Frank Kane
 - https://www.youtube.com/watch?v=v5e_PasMdXc



Final thoughts

- Know your data!
- Replication, sharding and consistent hashing are important concepts
 - Nothing is free :(
- We omitted several Cassandra specifics
 - SStable, tombstone ,...