Design a Key-Value Store

了解需要设计的Scope

- the key must be unique
- the value associated with the key can be accessed through the key.
- Keys can be plain text or hashed values or composite keys. For performance reasons, a short key works better

NoSQL Storage

APACHE ZooKeeper™

redis

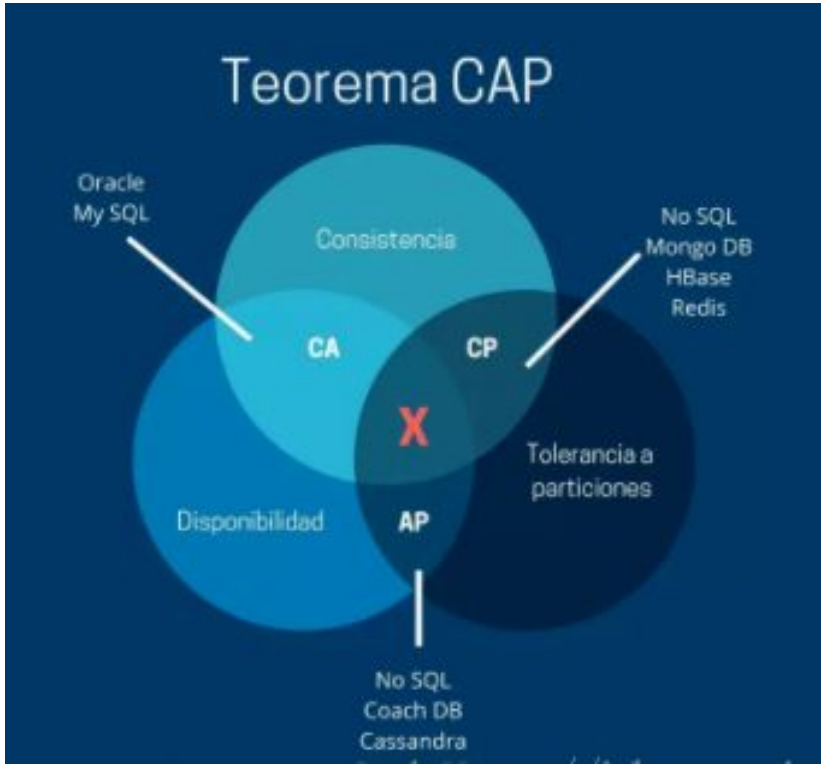
cassandra

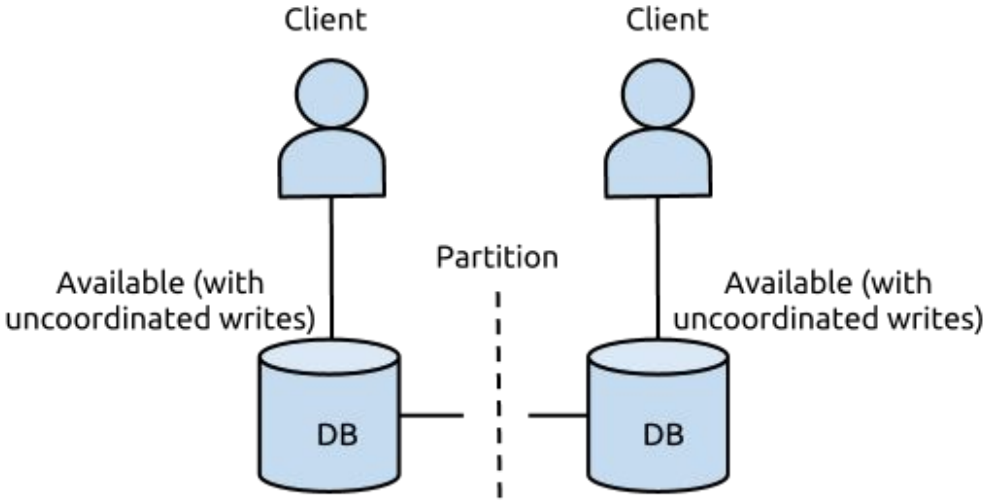
etcd

Single Node

- Big HashTable
  - Data compression
  - Store only frequently used data in memory and the rest on disk

Distributed KV Store - CAP/BASE Recall



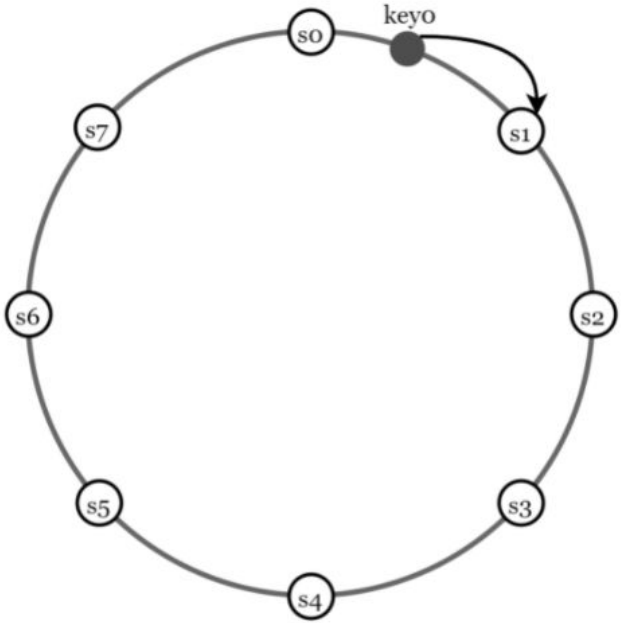


Distributed KV Store - Data Partition

Split the data into smaller partitions and store them in multiple servers

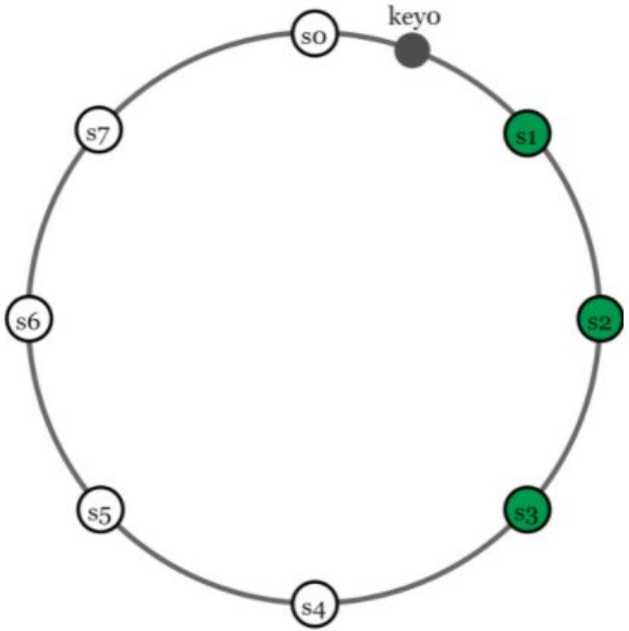
- Distribute data across multiple servers evenly.
- Minimize data movement when nodes are added or removed.

Distributed KV Store - Data Partition



DDIA Chapter 6：
1）基于关键字区间分区
2）哈希分区

Data Replication - High Availability



To achieve high availability and reliability, data must be replicated asynchronously over *N* servers, *N* is a configurable parameter.

DDIA Chapter 5:

1) 主从复制
2) 多主节点复制
3) 无主节点复制

Distributed KV Store - Consistency

Quorum consensus - tradeoff

N-W-R

W = 1; R = 1

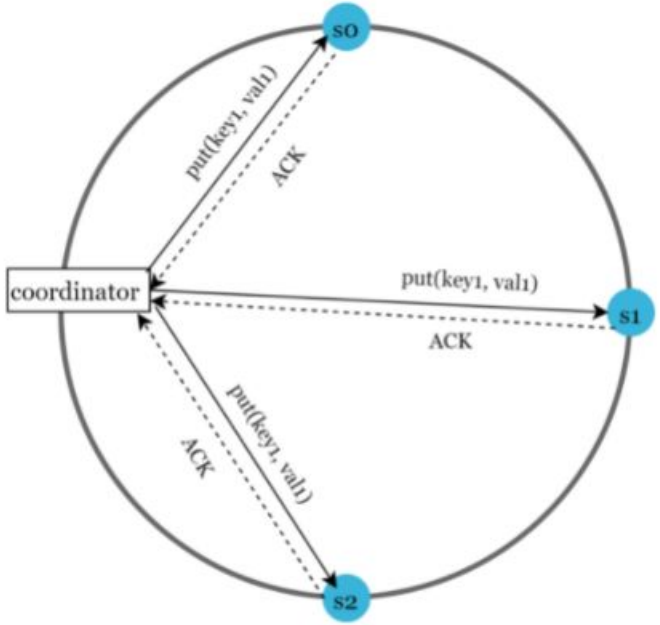
W or *R* > *1*:
If *W* + *R* > *N*, strong consistency is guaranteed because there must be at least one overlapping node that has the latest data to ensure consistency.

Some possible setups:
If *R* = *1* and *W* = *N*, the system is optimized for a fast read.

If *W* = *1 and R* = *N*, the system is optimized for fast write.

If *W* + *R* > *N*, strong consistency is guaranteed (Usually *N* = *3, W* = *R* = *2*).

If *W* + *R* <= *N*, strong consistency is not guaranteed.



Distributed KV Store - Consistency Models

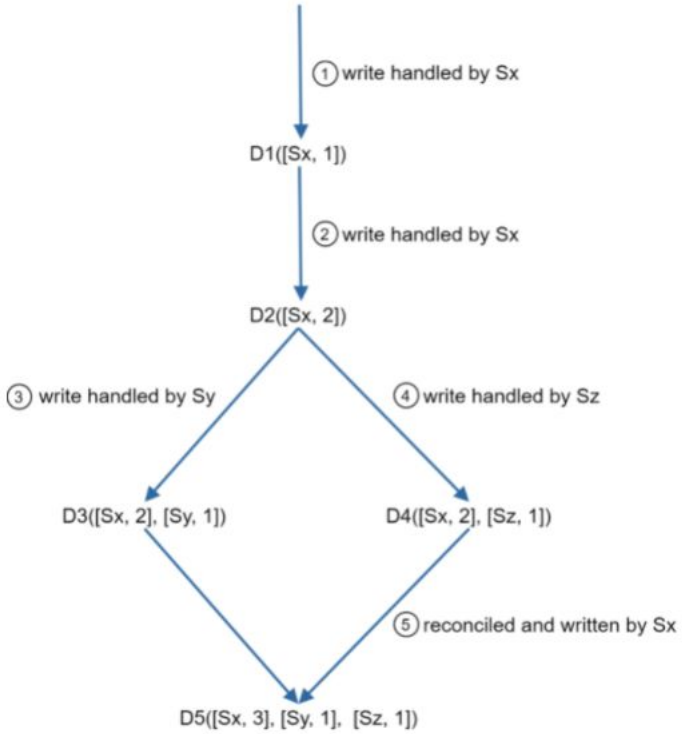
- **Strong consistency**: any read operation returns a value corresponding to the result of the most updated write data item. A client never sees out-of-date data.
- **Weak consistency**: subsequent read operations may not see the most updated value.
- **Eventual consistency**: this is a specific form of weak consistency. Given enough time, all updates are propagated, and all replicas are consistent.

Distributed KV Store - Inconsistency Solution Versioning (vector clock)

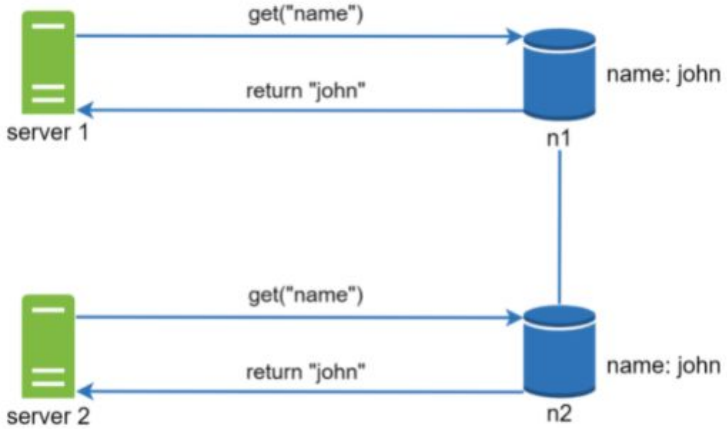
A vector clock is a *[server, version]* pair associated with a data item. It can be used to check if one version precedes, succeeds, or in conflict with others.

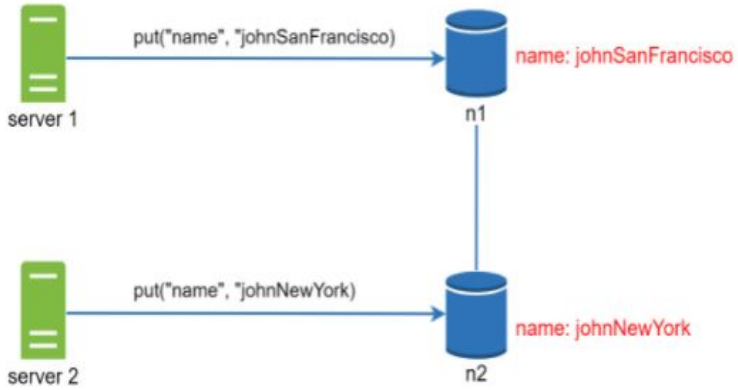
e.g.,
D([S1, v1], [S2, v2], …, [Sn, vn])

- Increment *vi* if *[Si, vi]* exists.
- Otherwise, create a new entry *[Si, 1]*.



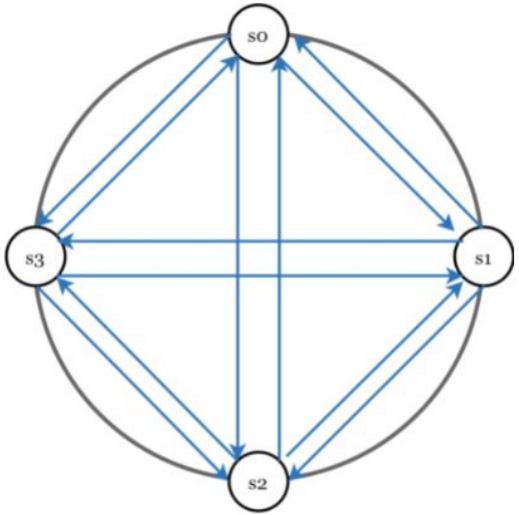
Distributed KV Store - Inconsistency Solution Versioning (vector clock)





Distributed KV Store - Failure Detection

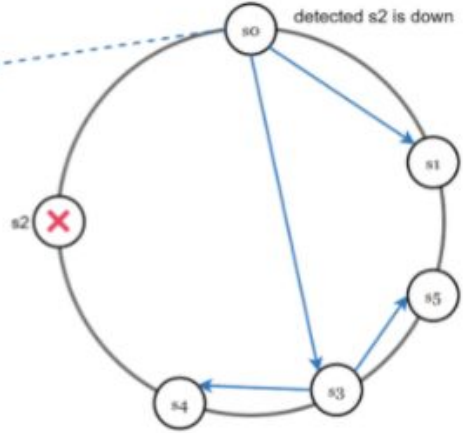
all-to-all multicasting



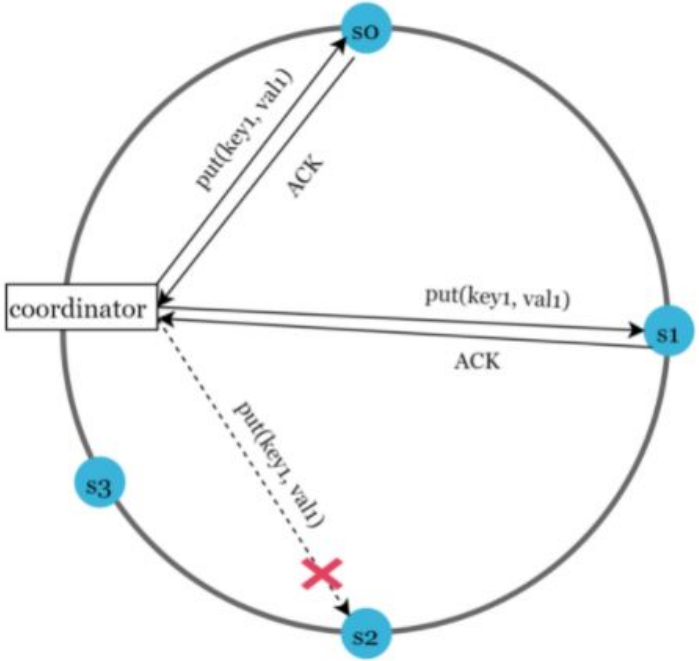
gossip protocol detection

s0's membership list

Member ID	Heartbeat counter	Time
0	10232	12:00:01
1	10224	12:00:10
2	9908	11:58:02
3	10237	12:00:20
4	10234	12:00:34

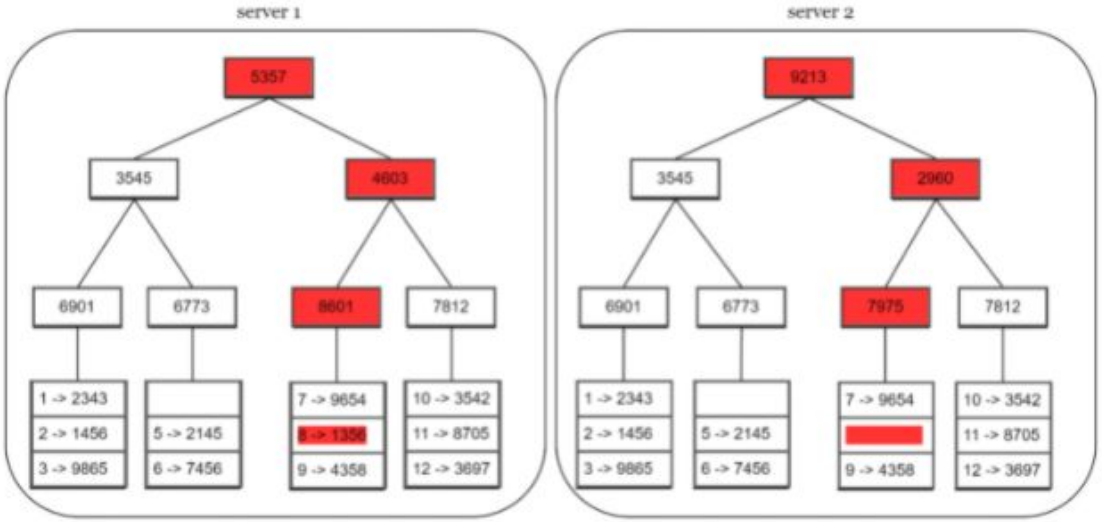


Distributed KV Store - Handling Temp Failures



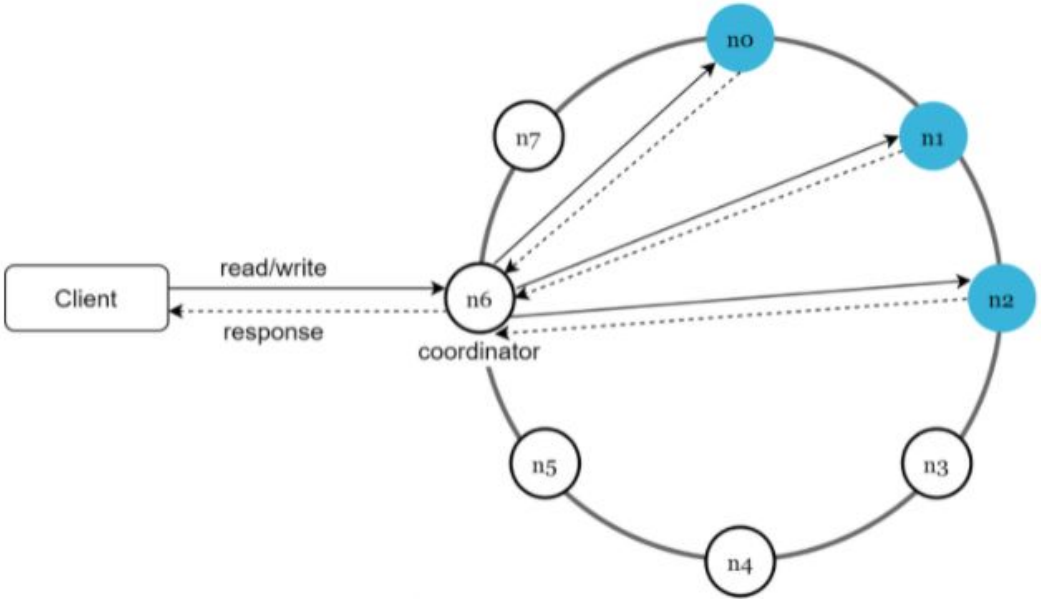
Sloppy Quorum
Hinted Handoff

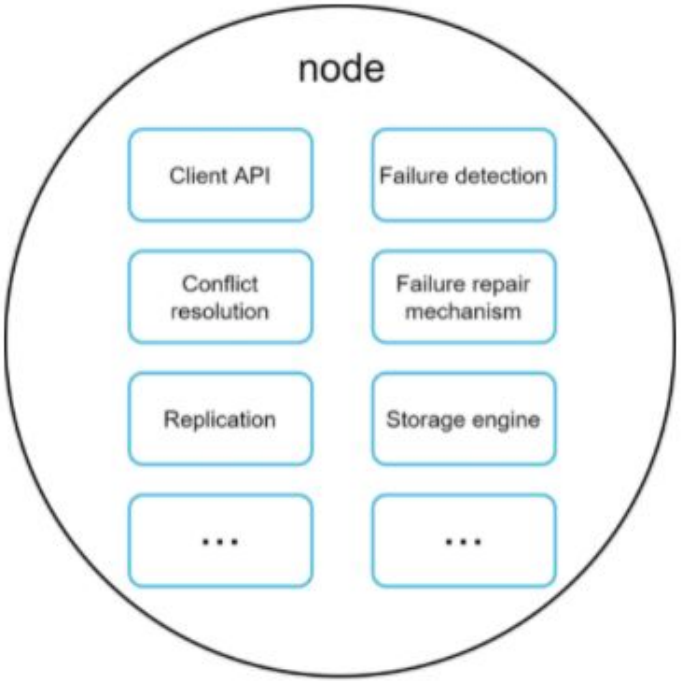
Distributed KV Store - Handling Perm Failures



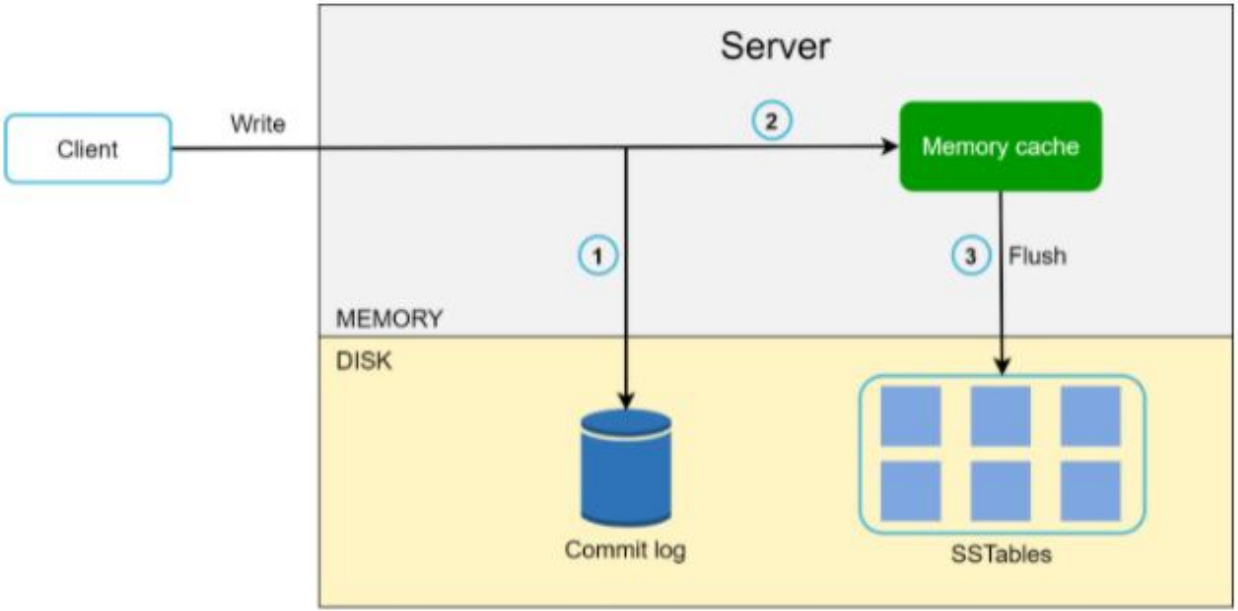
Anti-entropy protocol
Merkle tree

Distributed KV Store - Architecture Diagram

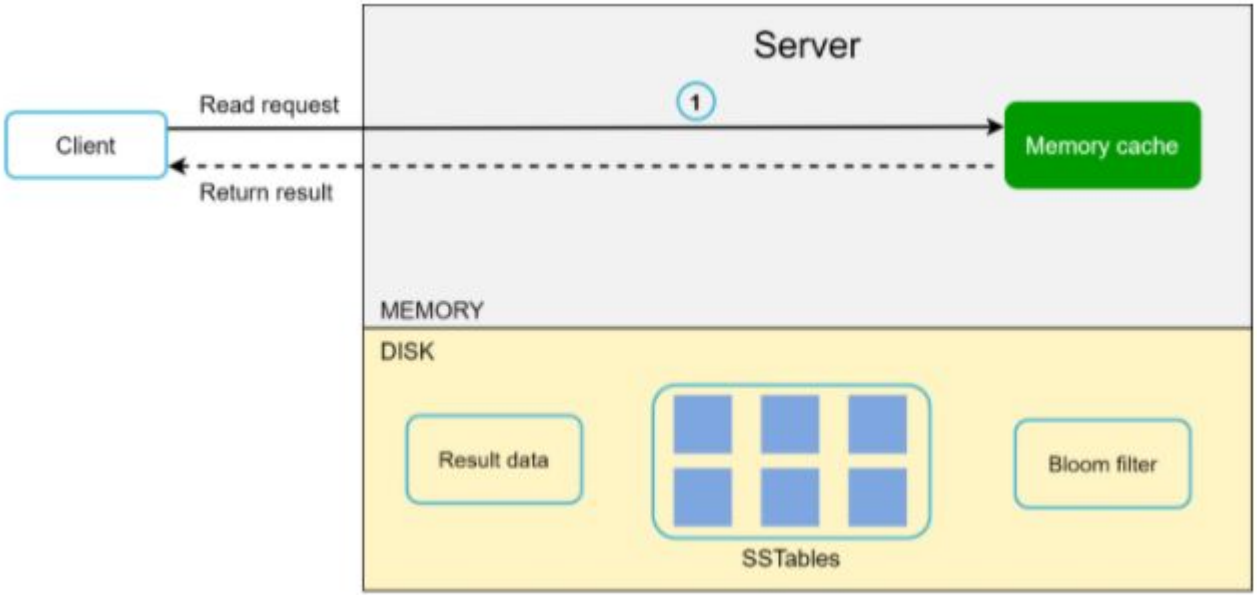




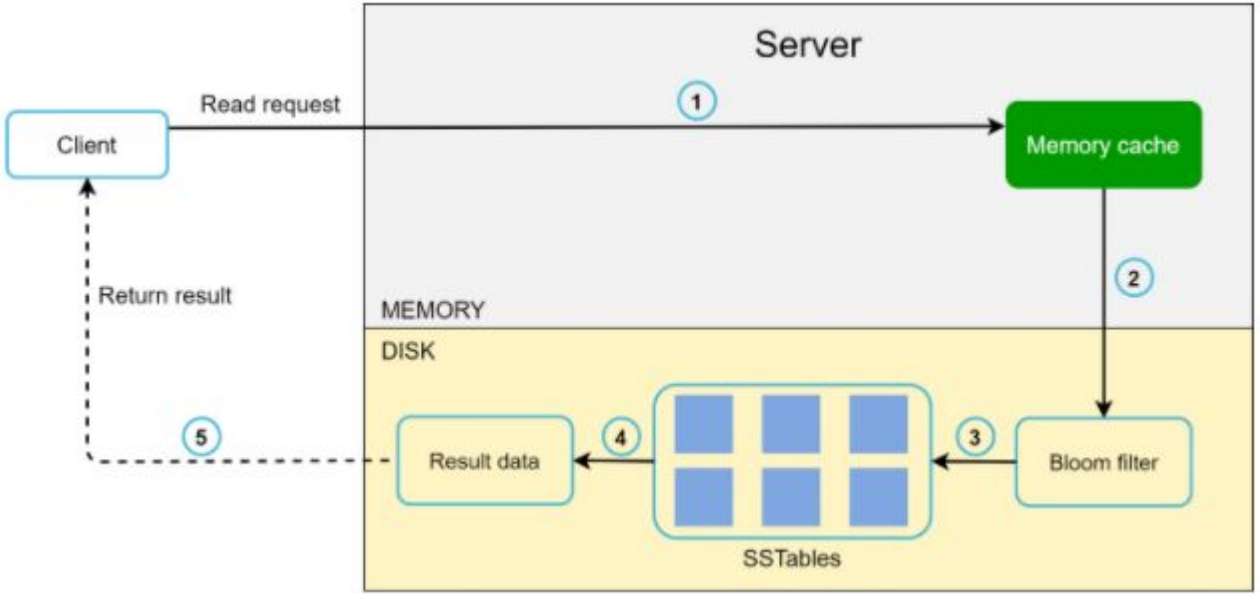
Distributed KV Store - Write Path



Distributed KV Store - Read Path



Distributed KV Store - Read Path



Furthermore

- Transaction
- Index
- Tombstone file

Fun/Non-Fun Requirements

Requirements	Techniques
Ability to store big data	Consistent Hashing
HA reads	Data replication/multi data center
HA writes	Versioning/conflict resolution/vector clock
Dataset partition	Consistent Hashing
Incremental scalability	Consistent Hashing
Heterogeneity	Consistent Hashing
Tunable consistency	Quorum consensus
Handling temp failures	Sloppy quorum/hinted handoff
Handling perm failures	Merkle Tree
Handling Data Center outage	Cross-data center replication

Reference

1. Dynamo: Amazon's Highly Available Key-value Store: https://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf
2. Merkle tree: https://en.wikipedia.org/wiki/Merkle_tree
3. Cassandra architecture: https://cassandra.apache.org/doc/latest/architecture/
4. SStable: https://www.igvita.com/2012/02/06/sstable-and-log-structured-storage-leveldb/
5. Bloom filter https://en.wikipedia.org/wiki/Bloom_filter