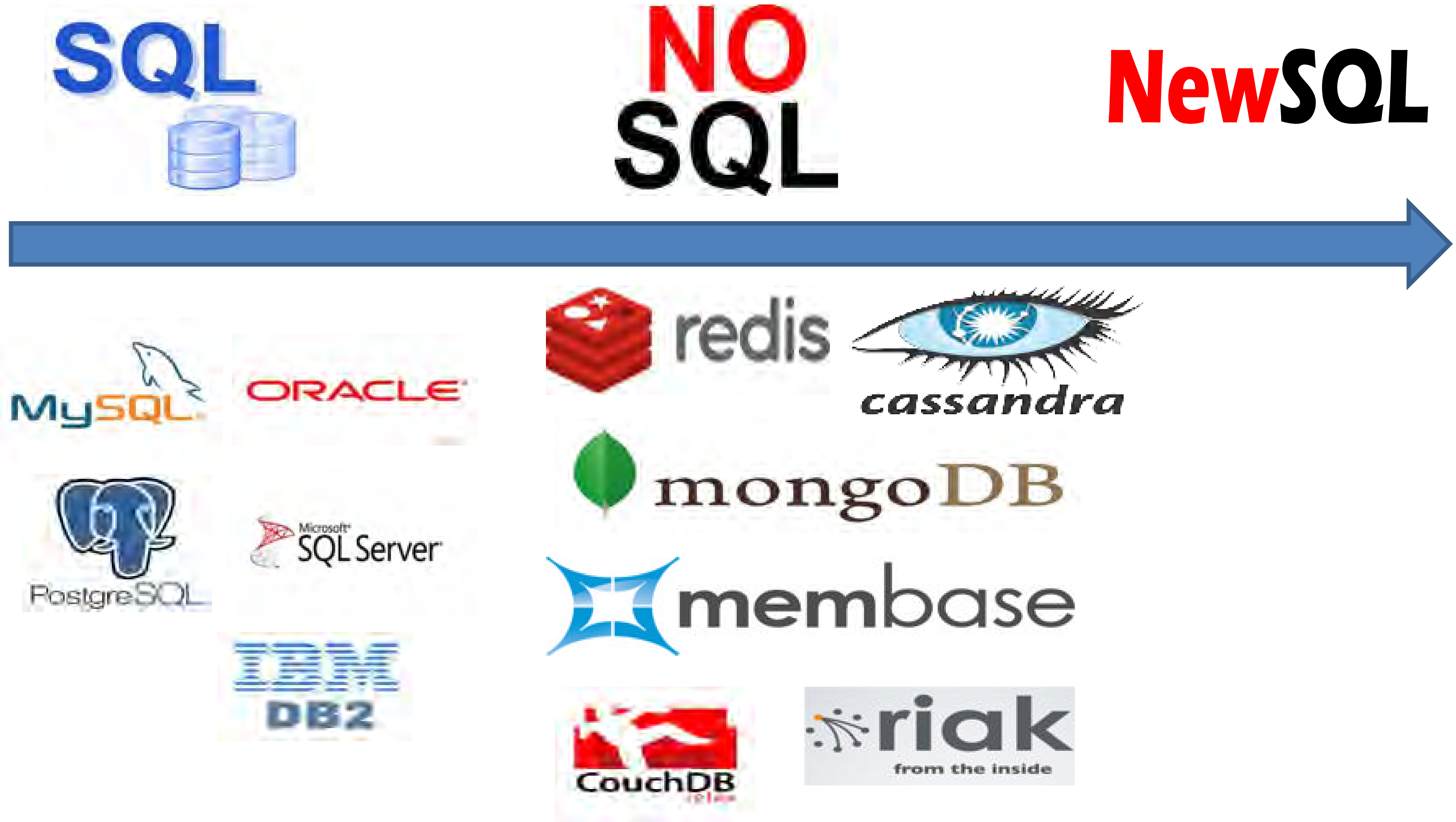
深入理解Raft及在NewSQL中的使用

京东数据库研发团队—张成远

Contents

- What is transaction
- What is consistency
- What is NewSQL
- What is Raft
- How Raft works
- Summary

What is NewSQL



What is transaction

Atomicity
——requires that each transaction be “all or nothing”

Consistency
——the consistency property ensures that any transaction will bring the database from one valid state to another

Isolation
—— the isolation property ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed serially, i.e., one after the other

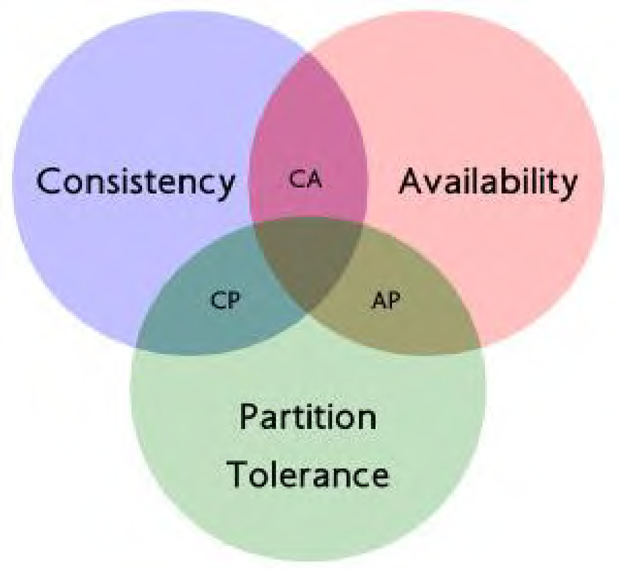
Durability
——the durability properity that once transaction has been commited, it will remain so, even in the event of power less, crashes or errors

What is consistency(CAP)

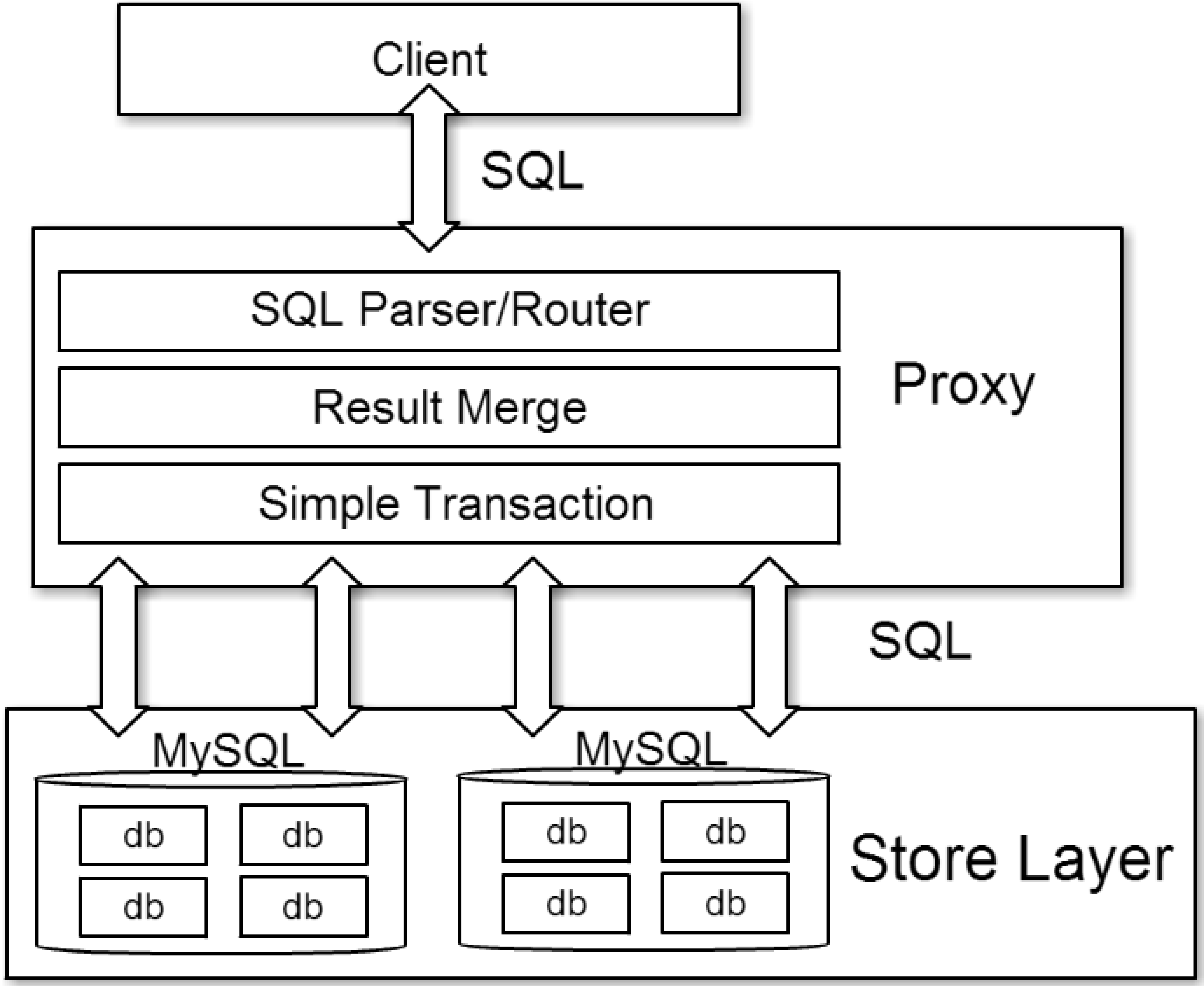
C
——consistency equivalent to having a single up-to-date cope of data.

A
——high availability of that data

P
—— tolerance to network partitions

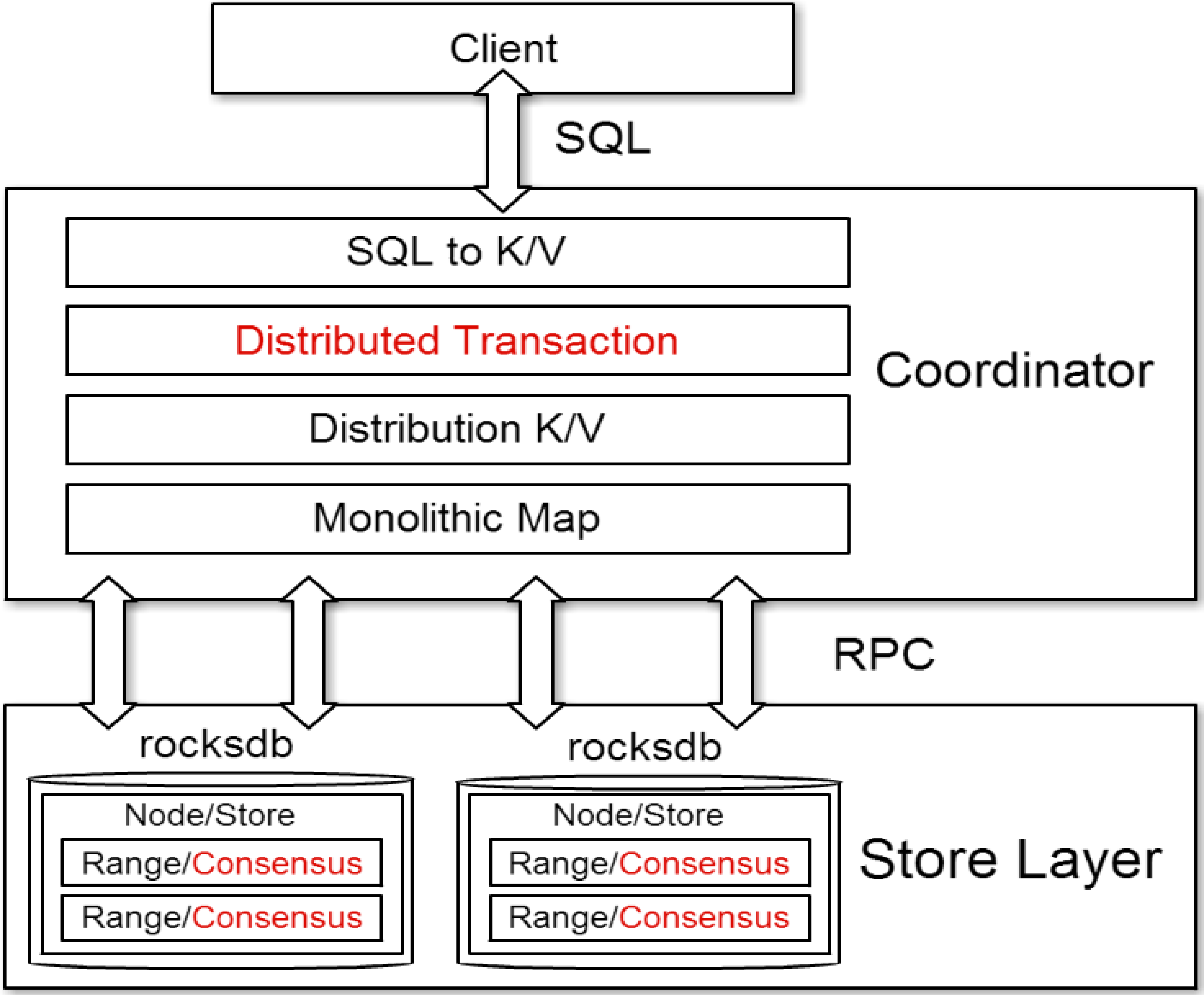


Distribution MySQL Cluster

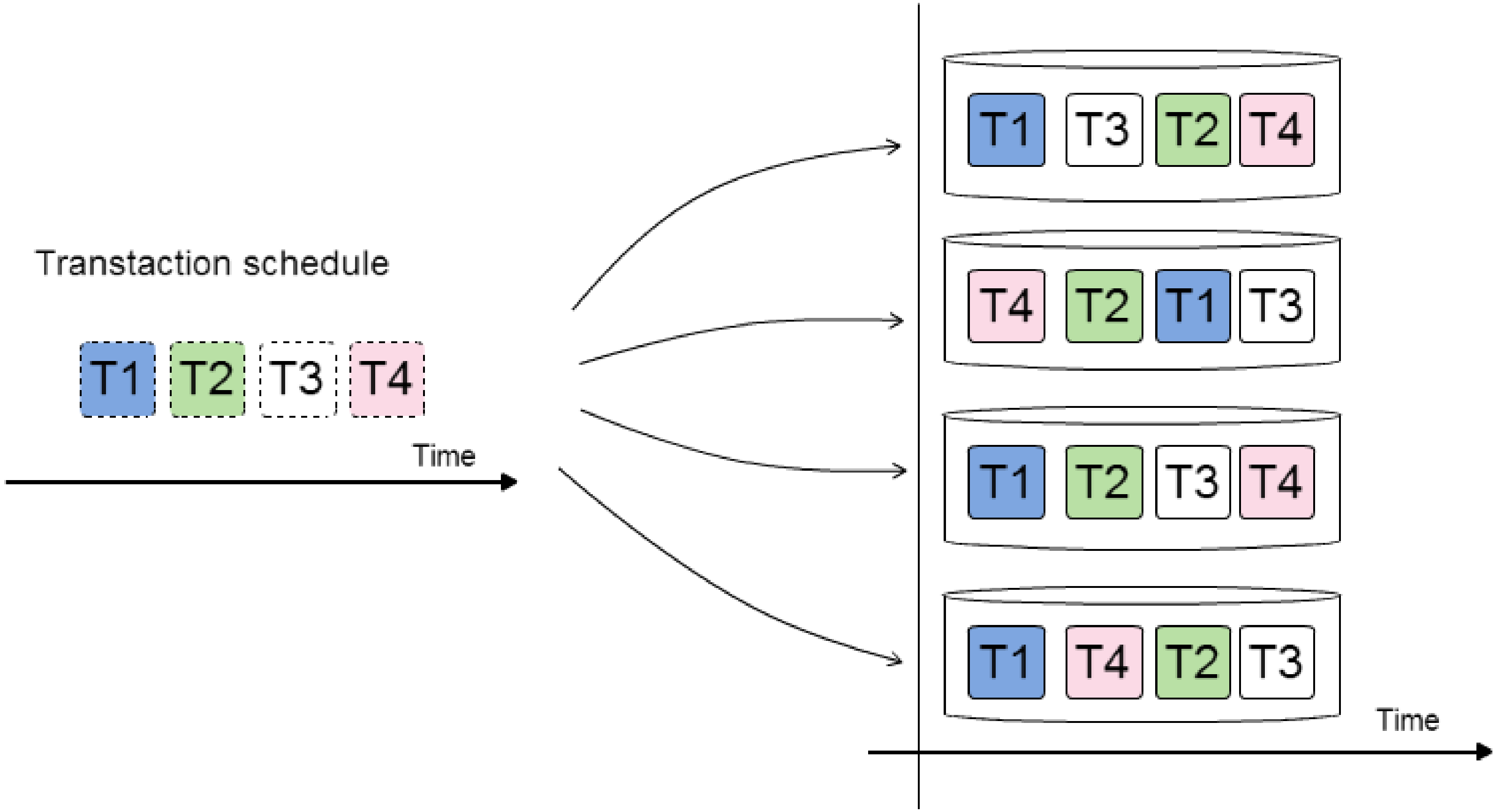


An example of NewSQL Architecture

APMCon

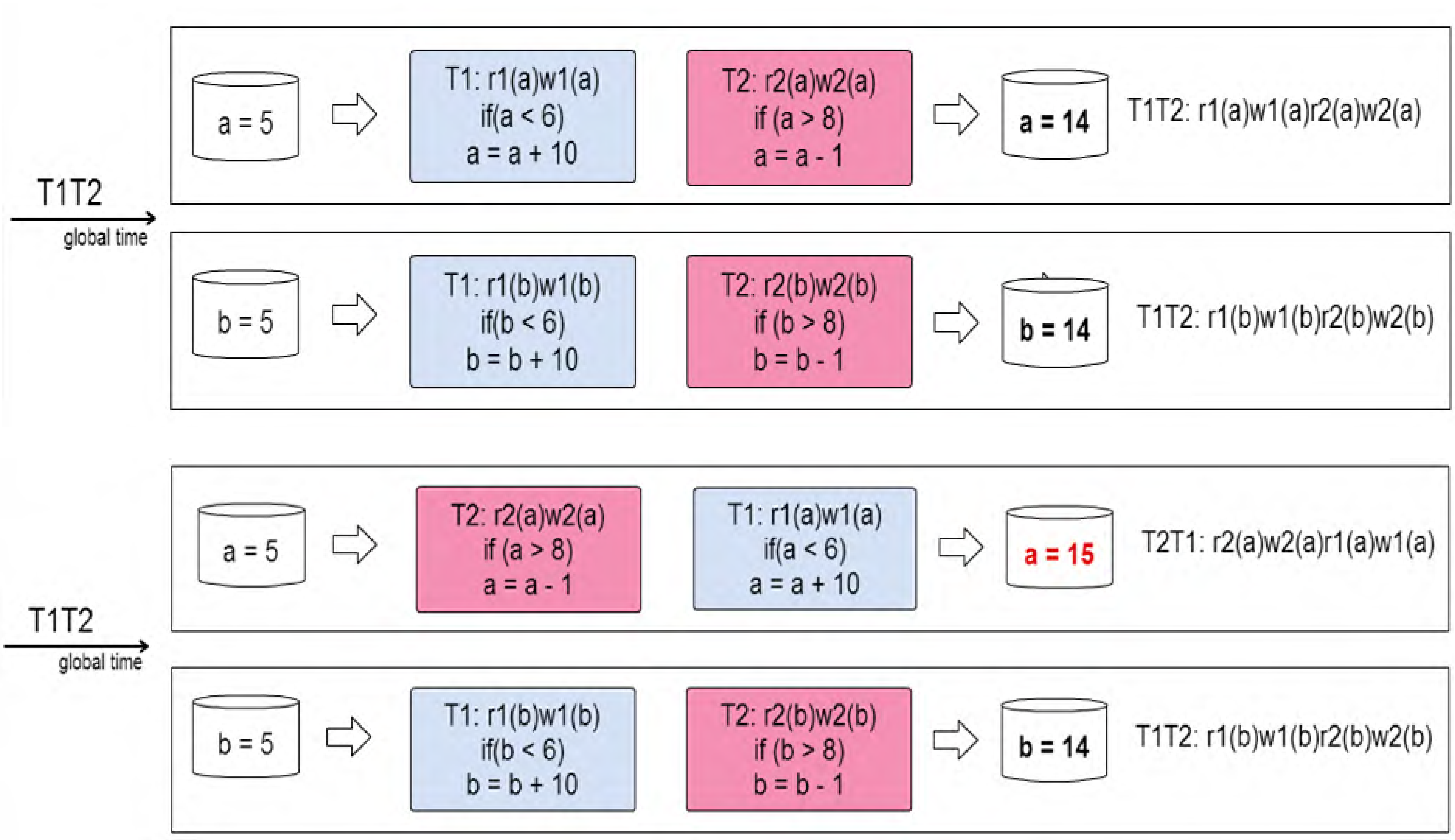


Transaction — Serializable Schedule



Isolation

Transaction — Serializable Schedule



T1T2

global time

a = 5

T1: r1(a)w1(a)
if(a < 6)
a = a + 10

T2: r2(a)w2(a)
if (a > 8)
a = a - 1

a = 14

T1T2: r1(a)w1(a)r2(a)w2(a)

b = 5

T1: r1(b)w1(b)
if(b < 6)
b = b + 10

T2: r2(b)w2(b)
if (b > 8)
b = b - 1

b = 14

T1T2: r1(b)w1(b)r2(b)w2(b)

T1T2

global time

a = 5

T2: r2(a)w2(a)
if (a > 8)
a = a - 1

T1: r1(a)w1(a)
if(a < 6)
a = a + 10

a = 15

T2T1: r2(a)w2(a)r1(a)w1(a)

b = 5

T1: r1(b)w1(b)
if(b < 6)
b = b + 10

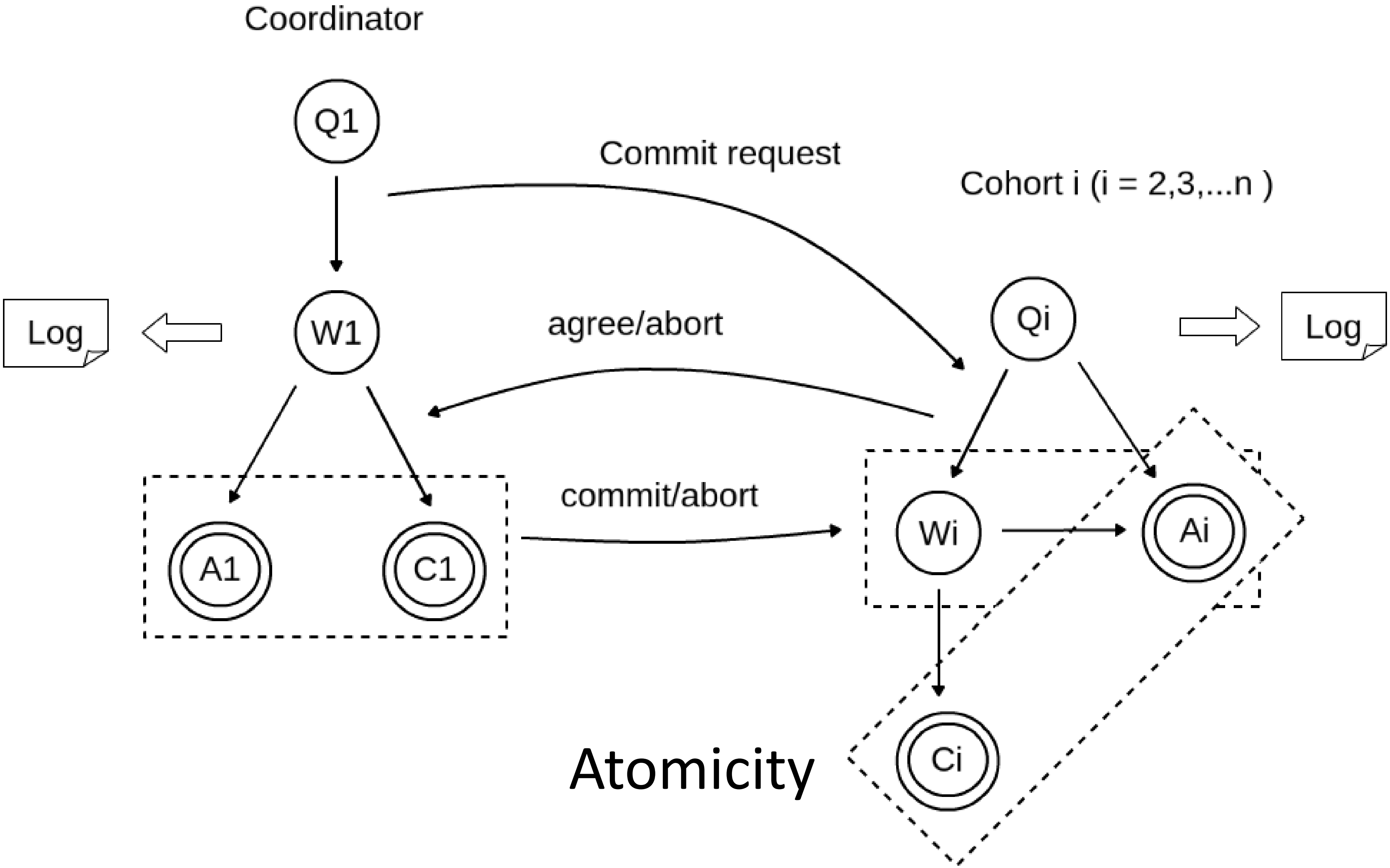
T2: r2(b)w2(b)
if (b > 8)
b = b - 1

b = 14

T1T2: r1(b)w1(b)r2(b)w2(b)

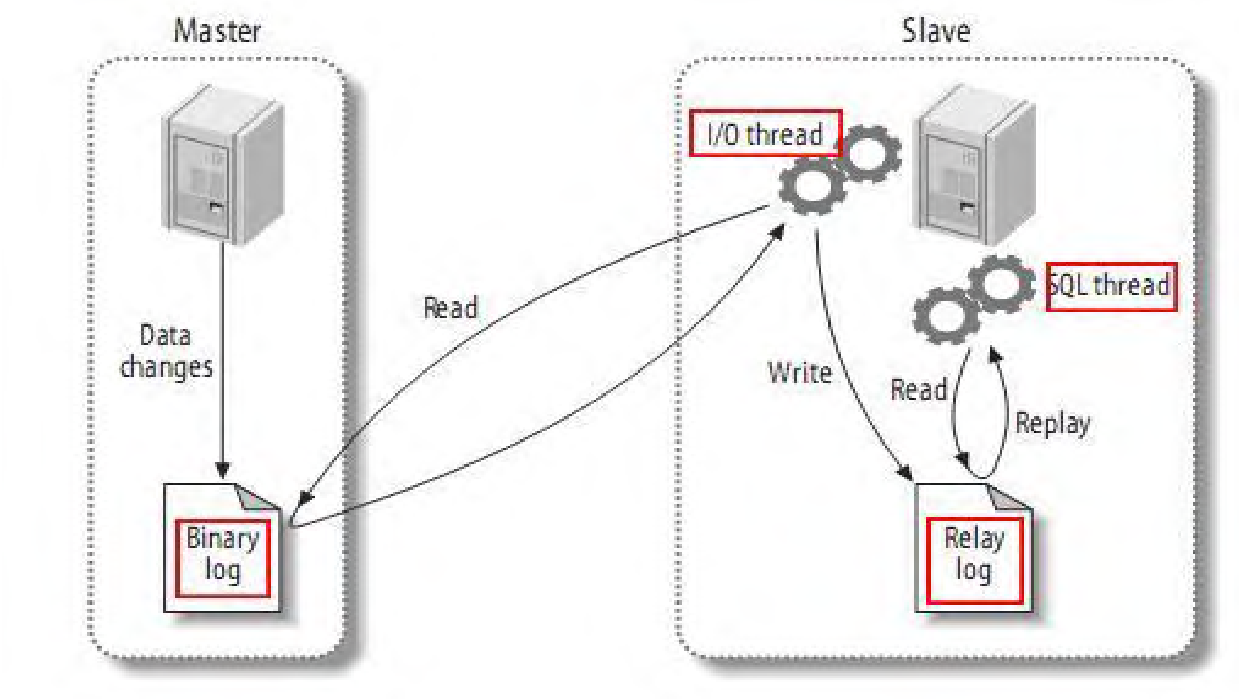
Isolation

Two-Phase commit protocol



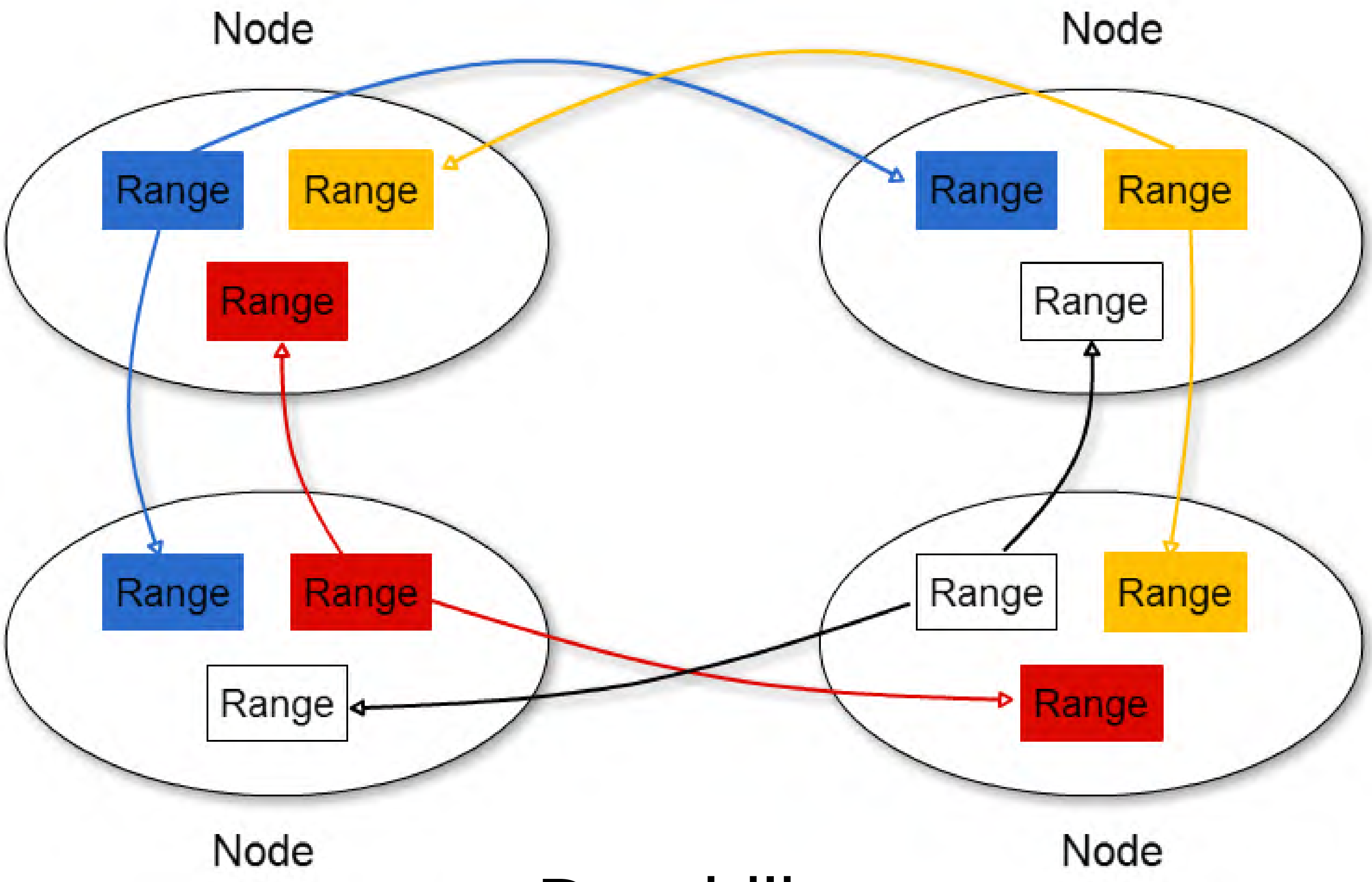
Atomicity

Consistency(CAP)



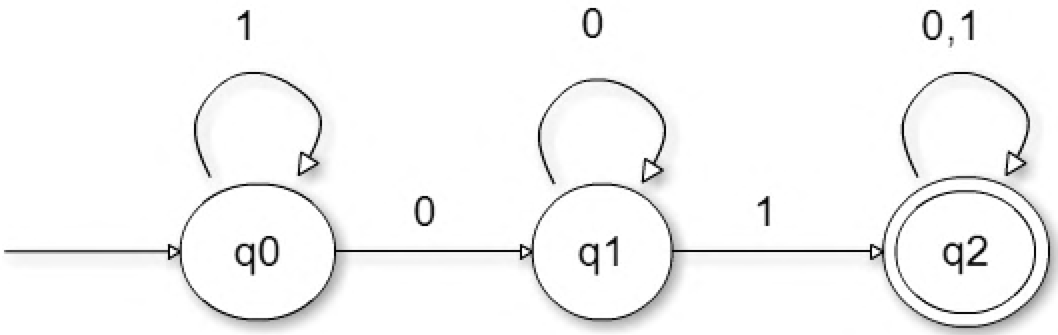
Durability

Consistency(CAP)



Durability

State machine



M = (Q, Σ, δ, q0, F)

- Q = {q0, q1, q2}
- Σ = {0, 1}
- δ : Q × Σ → Q
- q0 = {q0}
- F = {q2}

Q	0	1
q0	q1	q0
q1	q0	q2
q2	q2	q2

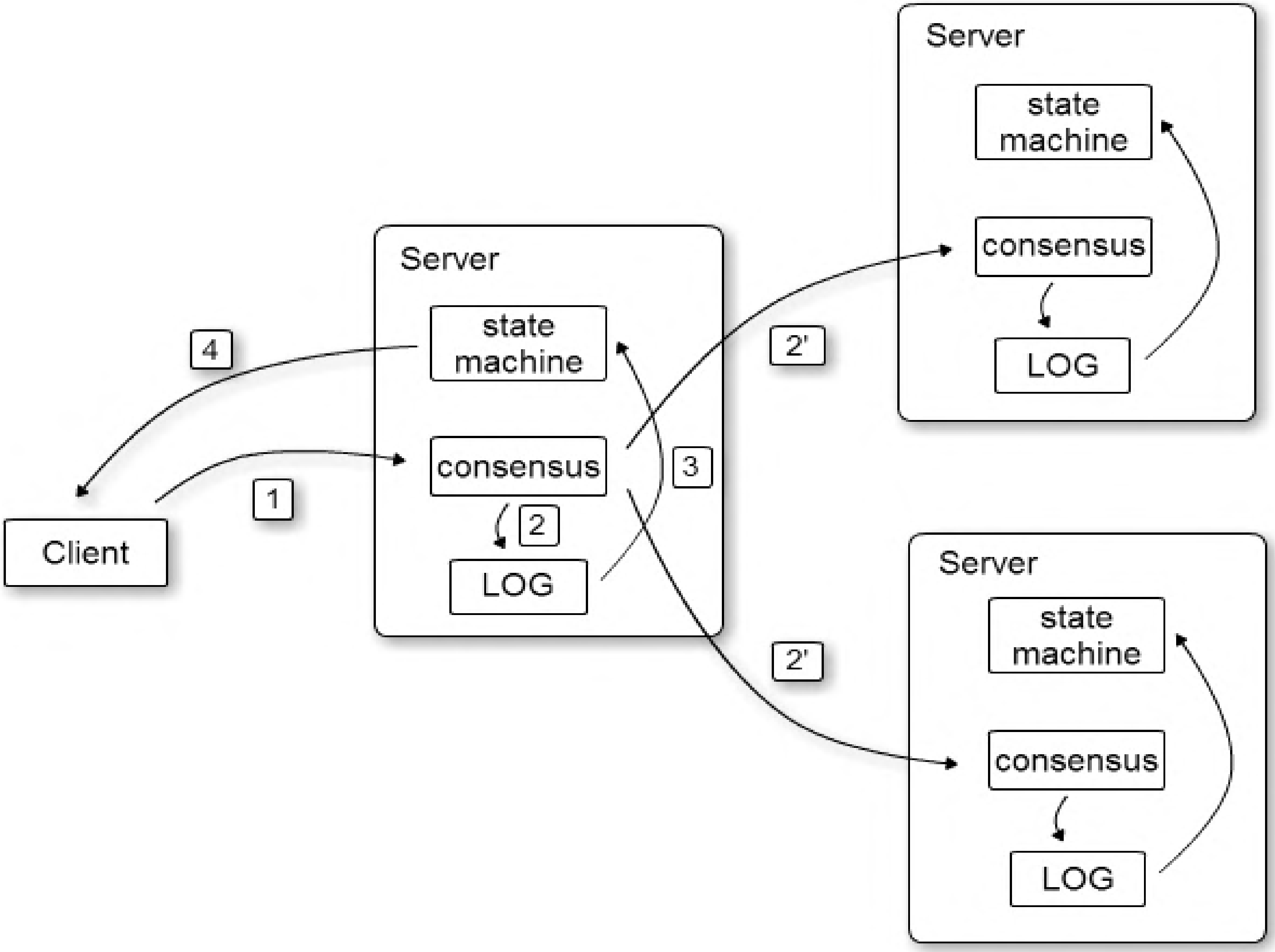
match： $1*0^{+}1(0|1)^{*}$

What is raft

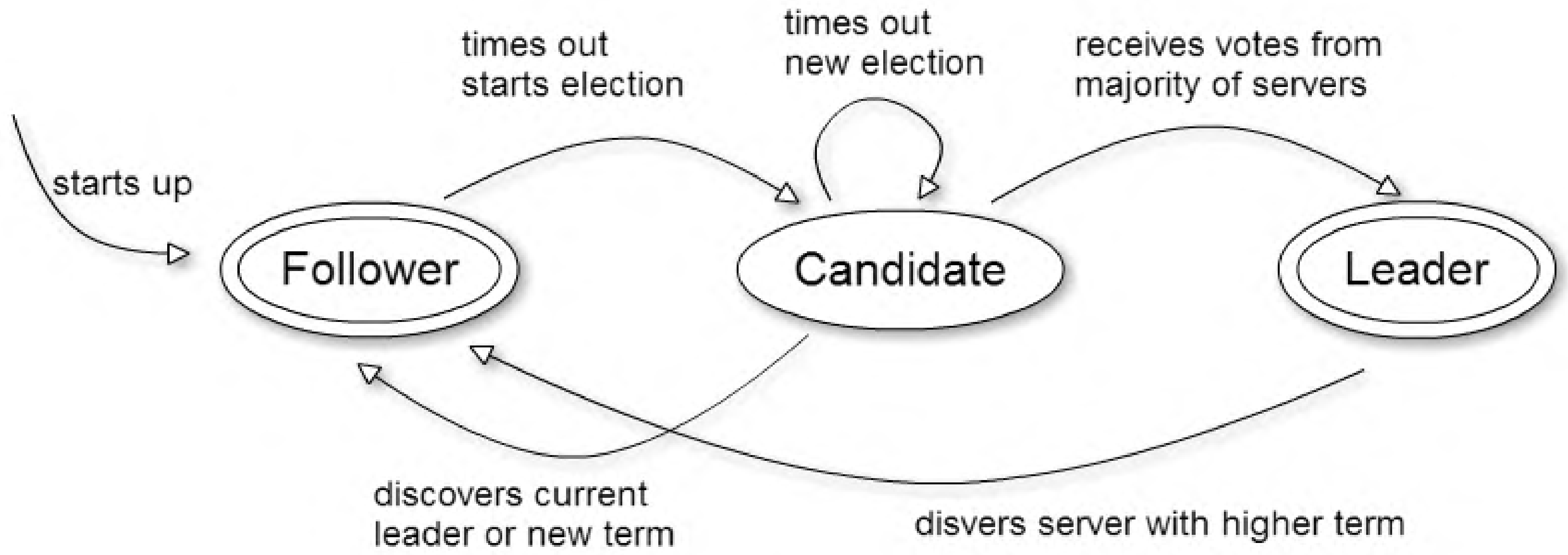
raft is an simple and understandability consensus algorithm

- Leader election
- Log replication
- Safety

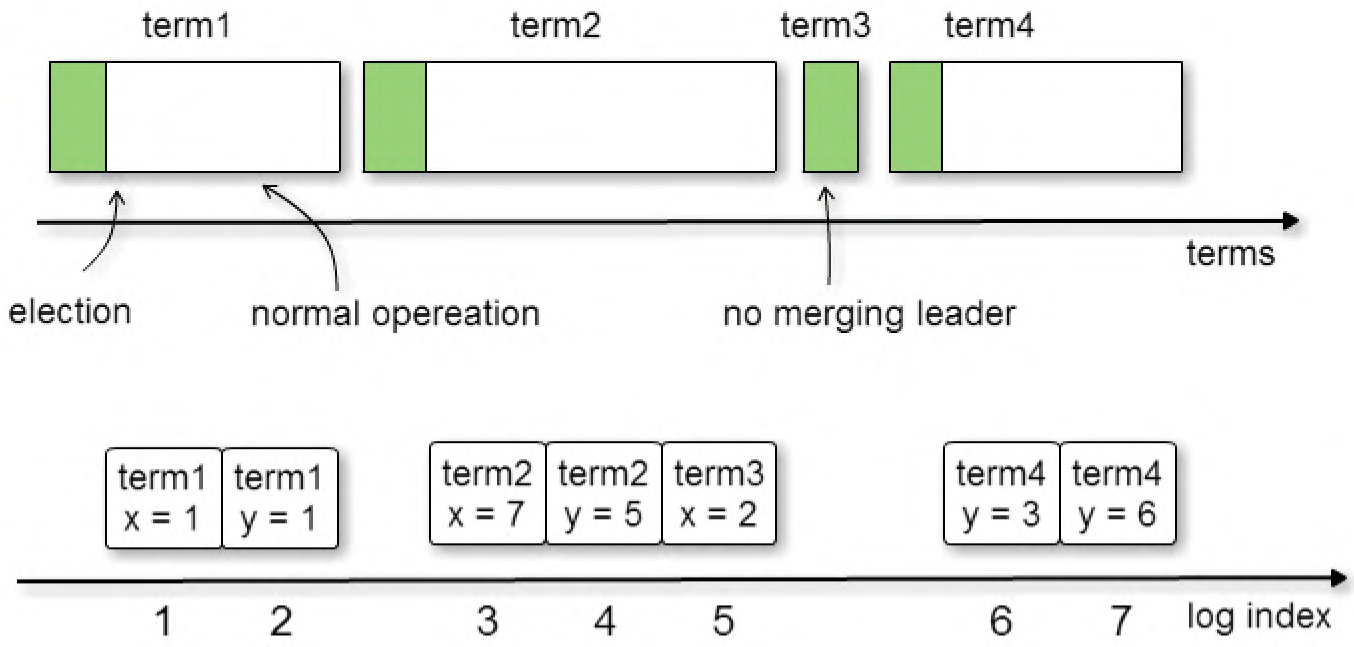
Replicated state machine



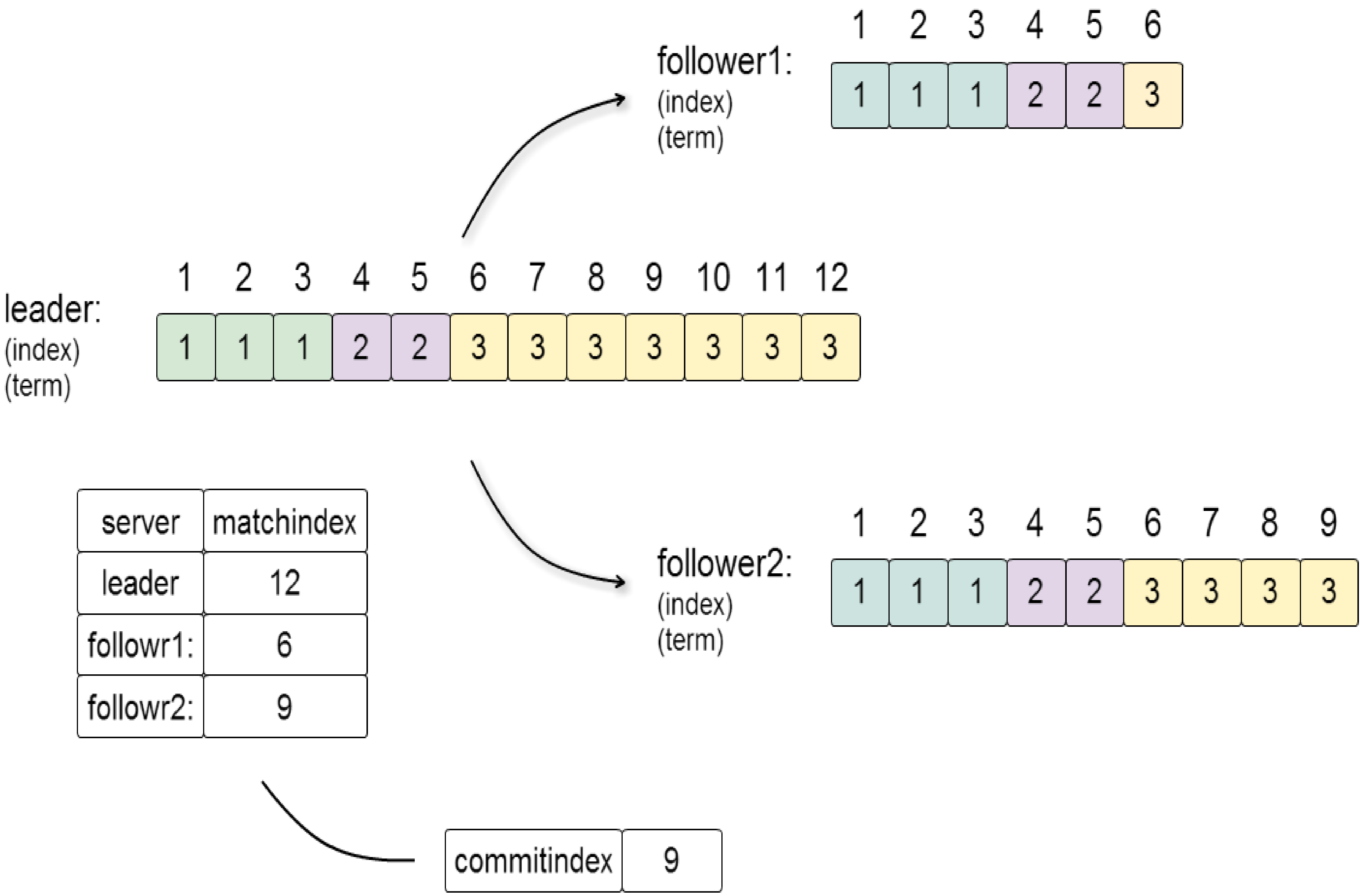
Server State



Term & log index



Log replication



follower1:

(index)	1	2	3	4	5	6
(term)	1	1	1	2	2	3

leader:

(index)	1	2	3	4	5	6	7	8	9	10	11	12
(term)	1	1	1	2	2	3	3	3	3	3	3	3

follower2:

(index)	1	2	3	4	5	6	7	8	9
(term)	1	1	1	2	2	3	3	3	3

server	matchindex
leader	12
followr1:	6
followr2:	9

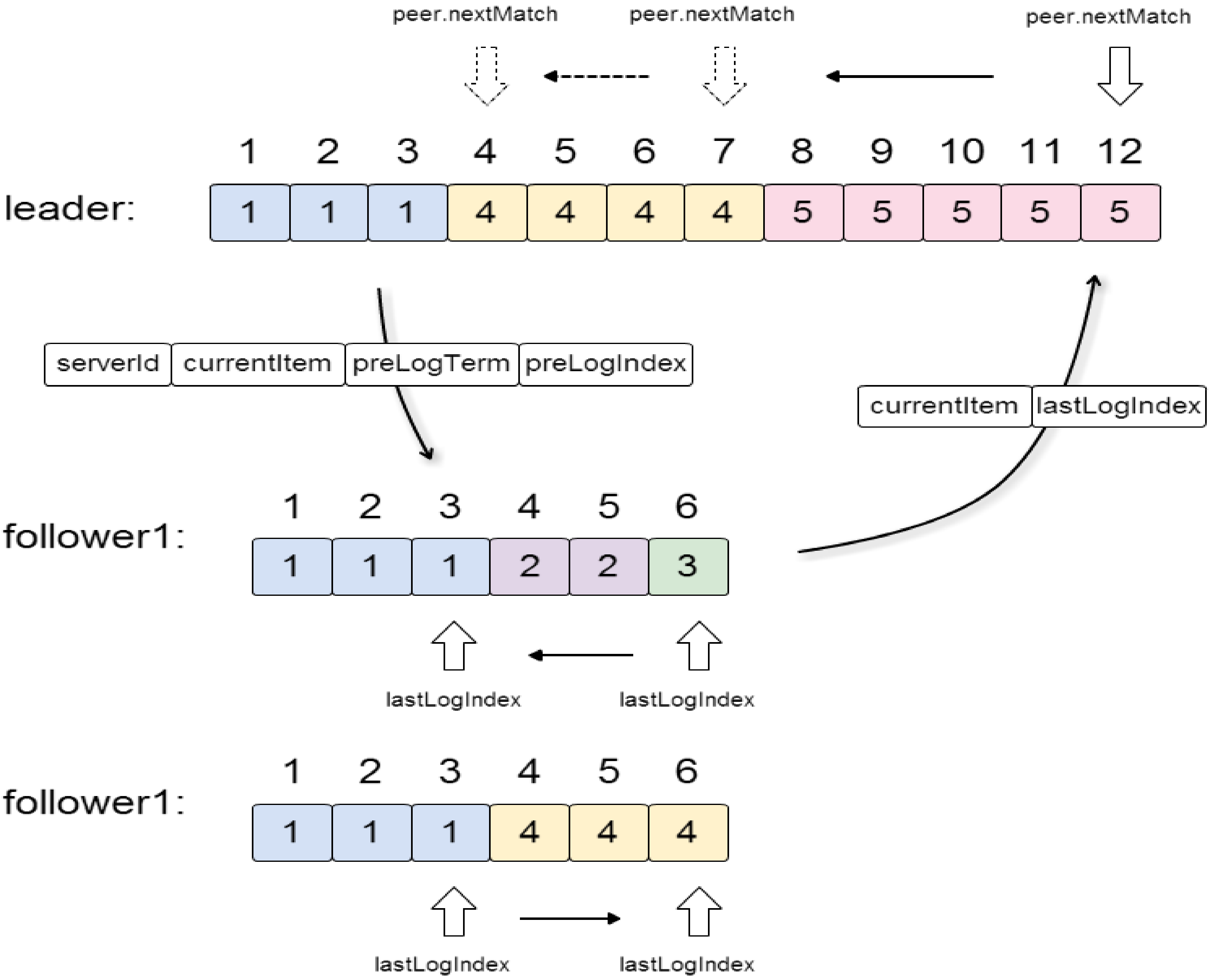
commitindex	9

Log replication

- If two entries in different logs have the same index and term, then they store the same command.

- If two entries in different logs have the same index and term, then the logs are identical in all preceding entries.

Log replication



leader:

1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	4	4	4	4	5	5	5	5	5

serverId	currentItem	preLogTerm	preLogIndex

currentItem	lastLogIndex

follower1:

1	2	3	4	5	6
1	1	1	2	2	3

follower1:

1	2	3	4	5	6
1	1	1	4	4	4

Summary

●How to support distribution transaction in NewSQL ?

Atomicity — Two-Phase commit protocol
Consistency — the same as in SQL
Isolation — Global Timestamp
Durability — Raft (Consistency<CAP>)

●How raft works?

leader election
log replication

THANK YOU