PingCAP

Building a Cloud-Native NewSQL Database

shenli@PingCAP

About me

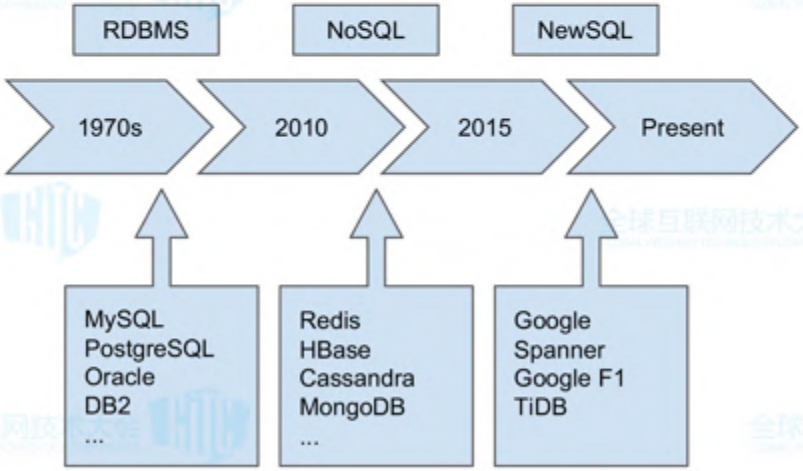
- Shen Li (申砾)
- Tech Lead of TiDB, VP of Engineering
- Netease / 360 / PingCAP
- Infrastructure software engineer

PingCAP

Why do we need a new database?

Brief History

- Standalone RDBMS
- NoSQL
- Middleware & Proxy
- NewSQL



NewSQL database

- Horizontal Scalability
- Transaction
- High Availability & Auto-Failover
- SQL at scale

How do we build a NewSQL database?

What is TiDB

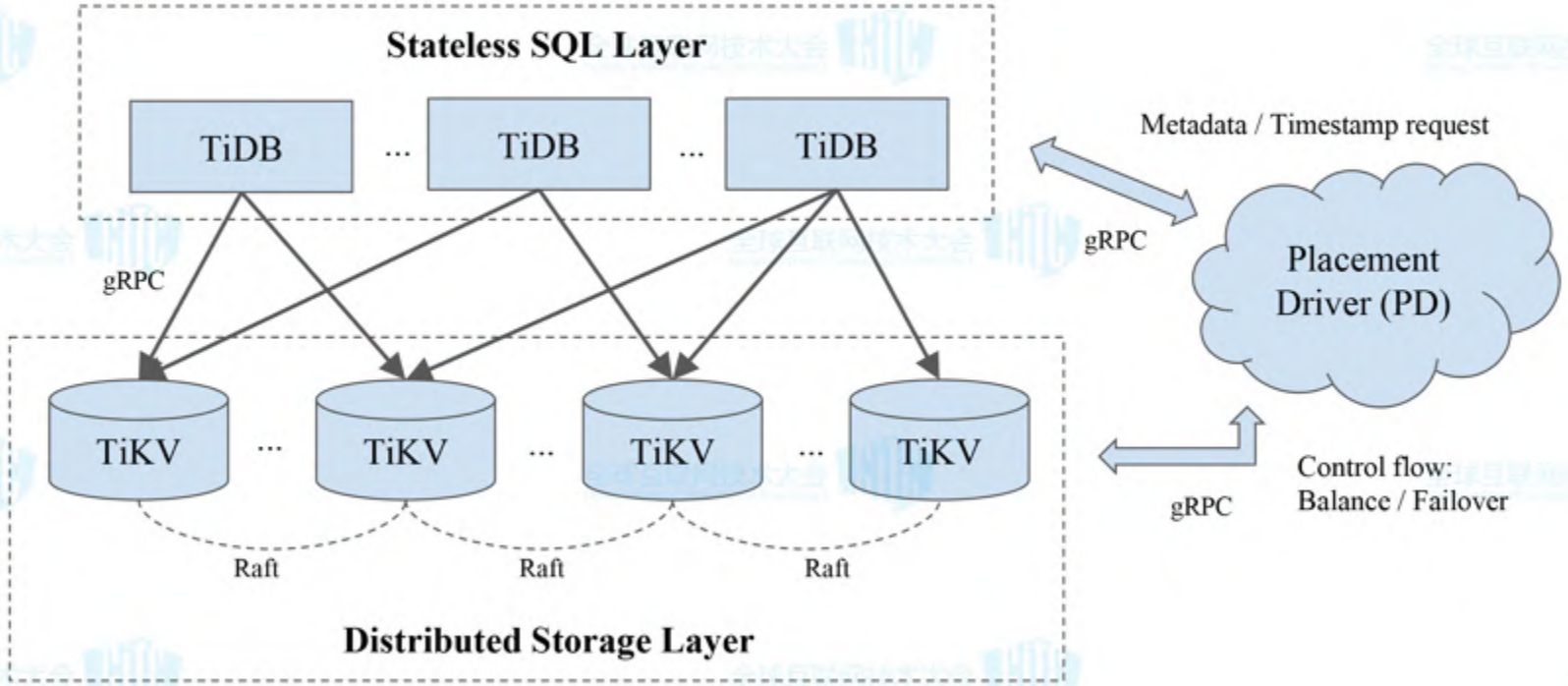
- Scalability as the first class feature
- SQL is necessary
- Compatible with MySQL, in most cases
- OLTP + OLAP = HTAP (Hybrid Transactional/Analytical Processing)
- 24/7 availability, even in case of datacenter outages
- Open source, of course

TiDB

A Distributed SQL Database

PingCAP

Architecture



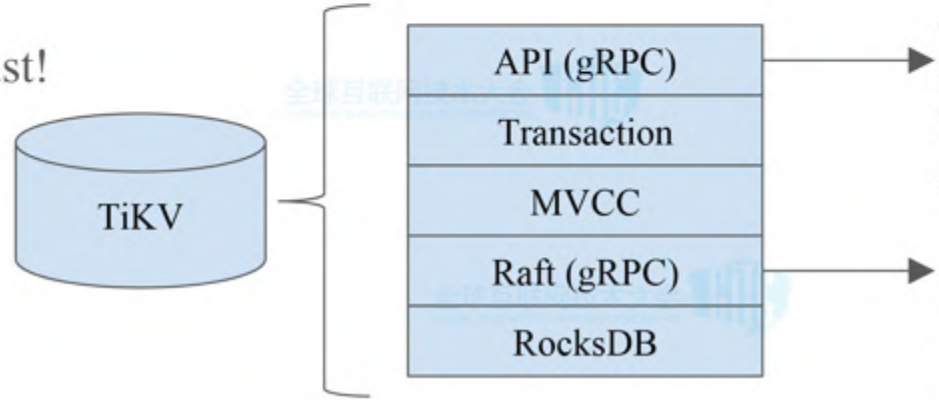
PingCAP

Data distribution

- Hash Based Partition
  - Redis
  - Scale well
  - Bad for scan
- Range Based Partition
  - Hbase
  - Good for SQL workload
  - Range size should be small enough and large enough

Storage stack 1/2

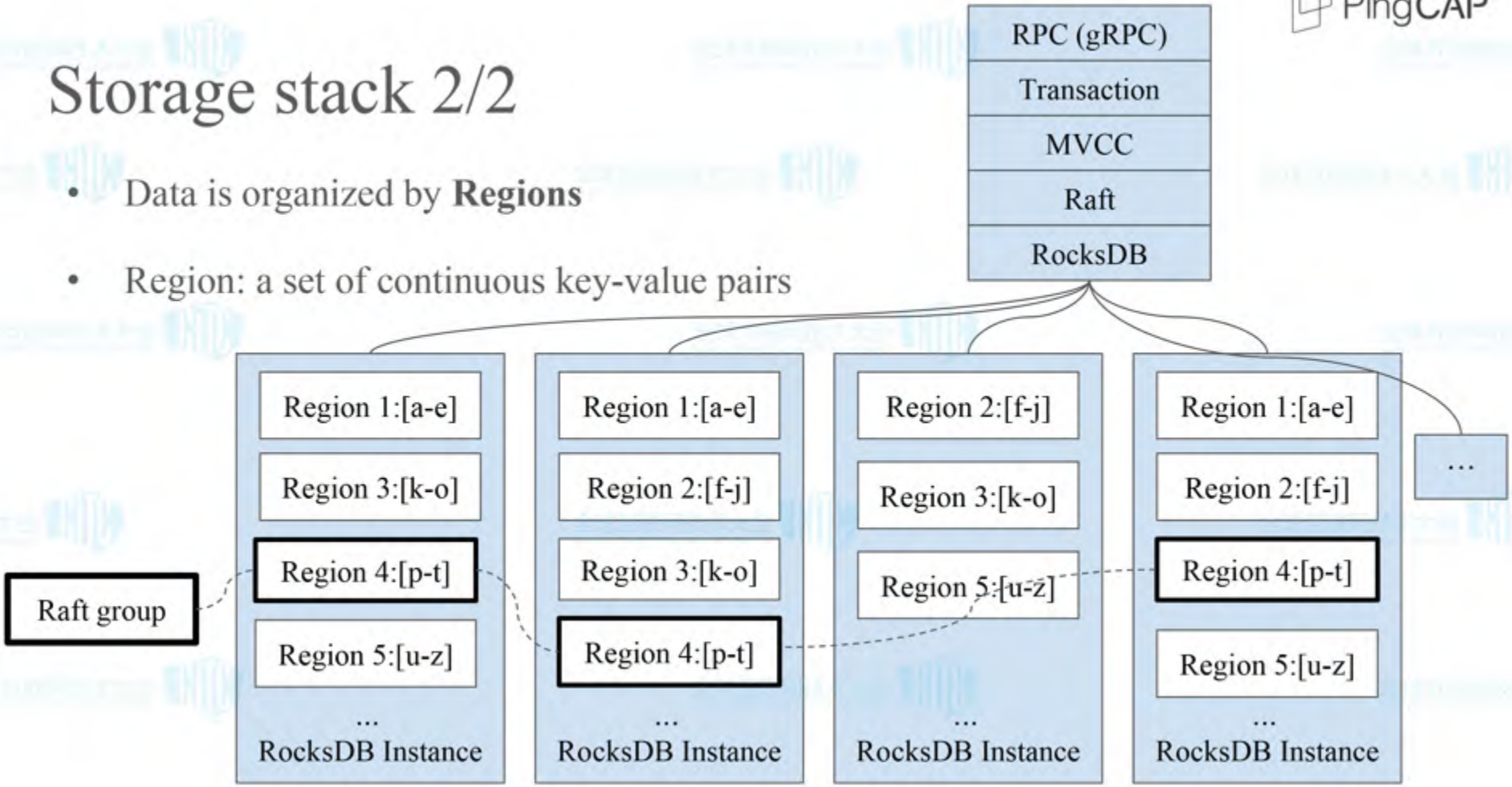
- TiKV is the underlying storage layer
- Physically, data is stored in RocksDB
- We build a Raft layer on top of RocksDB
  - What is Raft?
- Written in Rust!



PingCAP

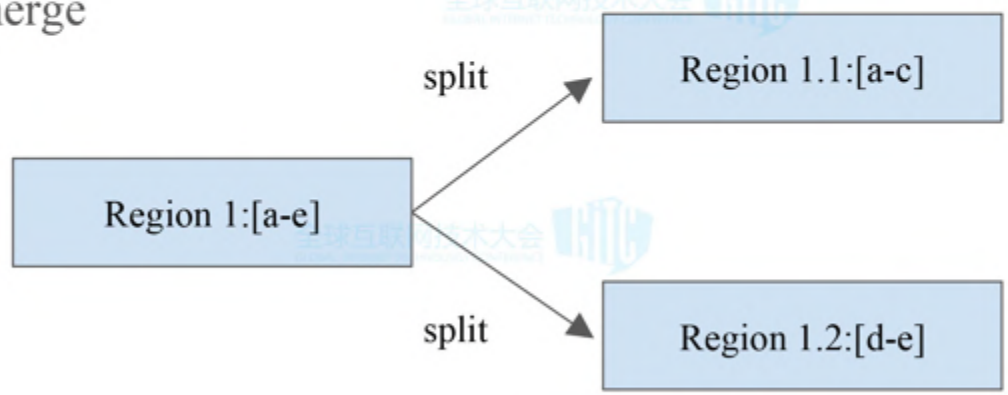
Storage stack 2/2

- Data is organized by **Regions**
- Region: a set of continuous key-value pairs

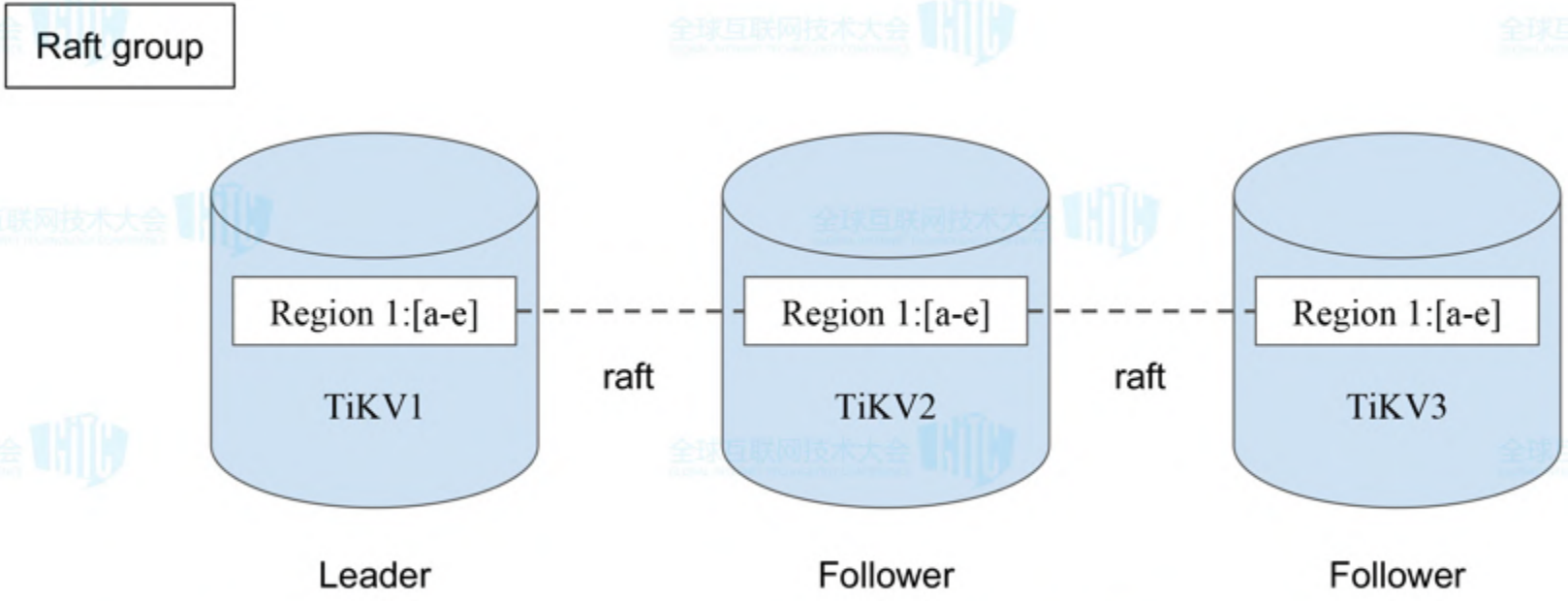


Dynamic Multi-Raft

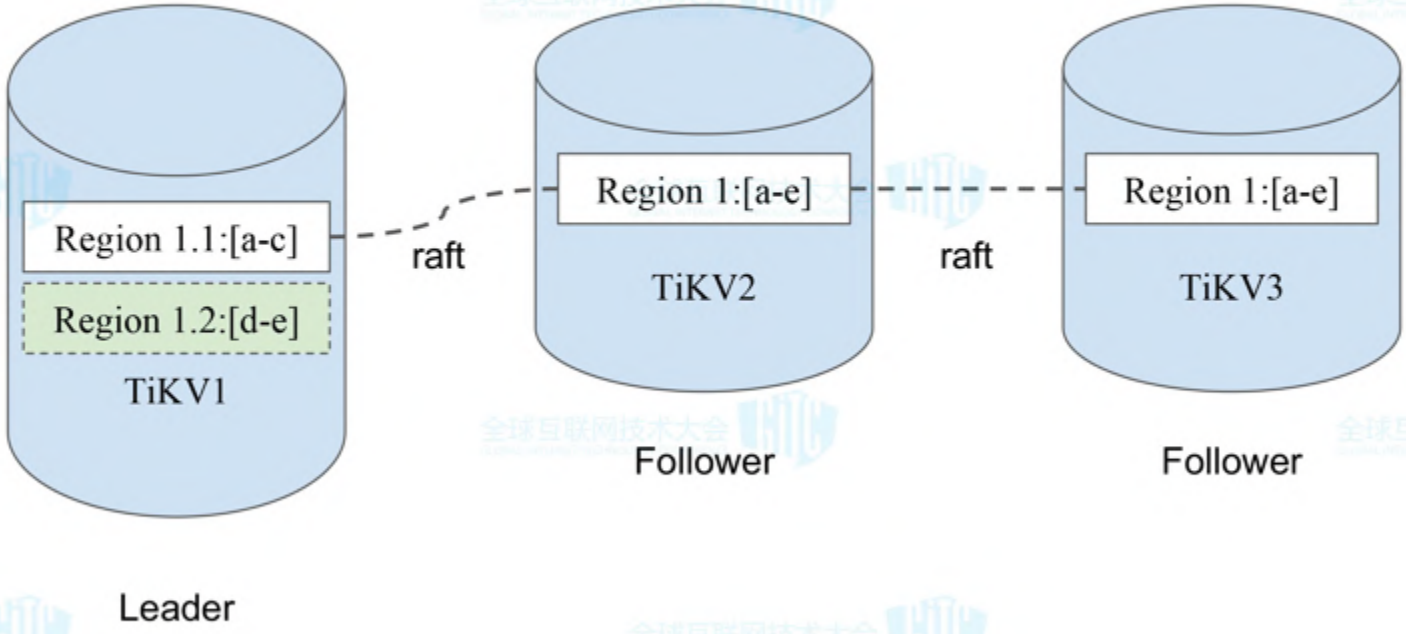
- What's Dynamic Multi-Raft?
  - Dynamic split / merge
- Safe split / merge



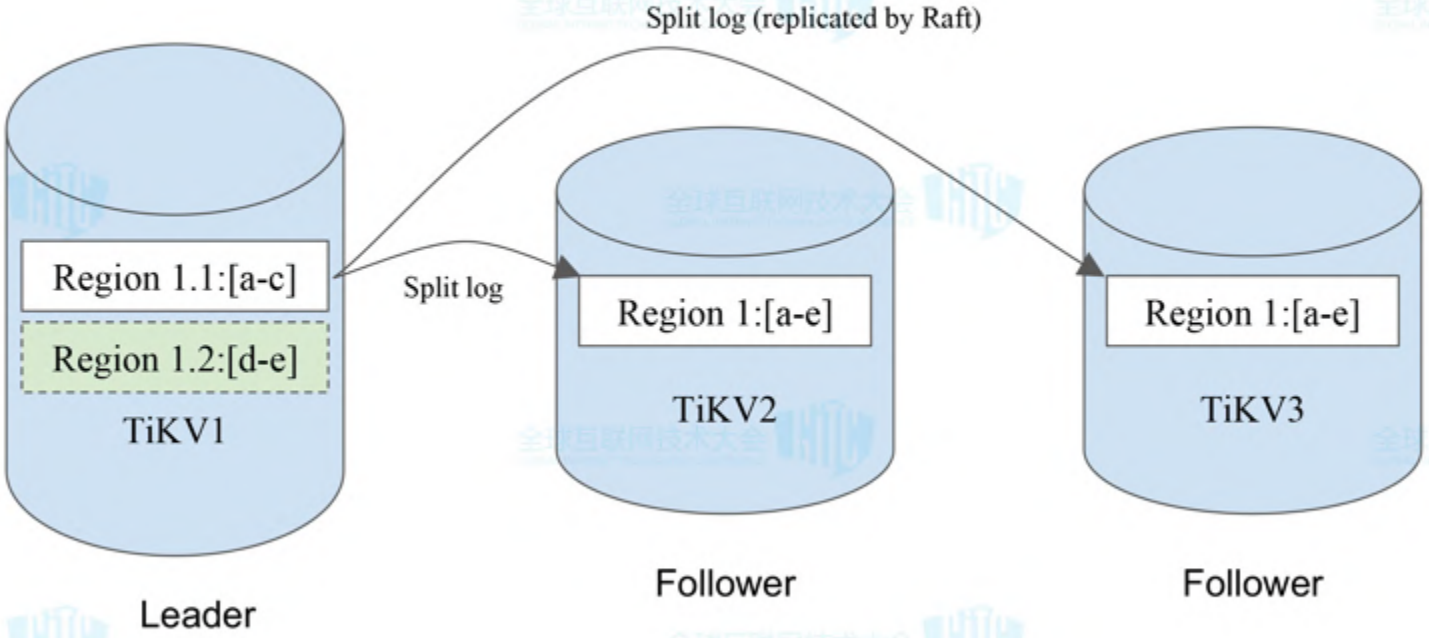
Safe Split: 1/4



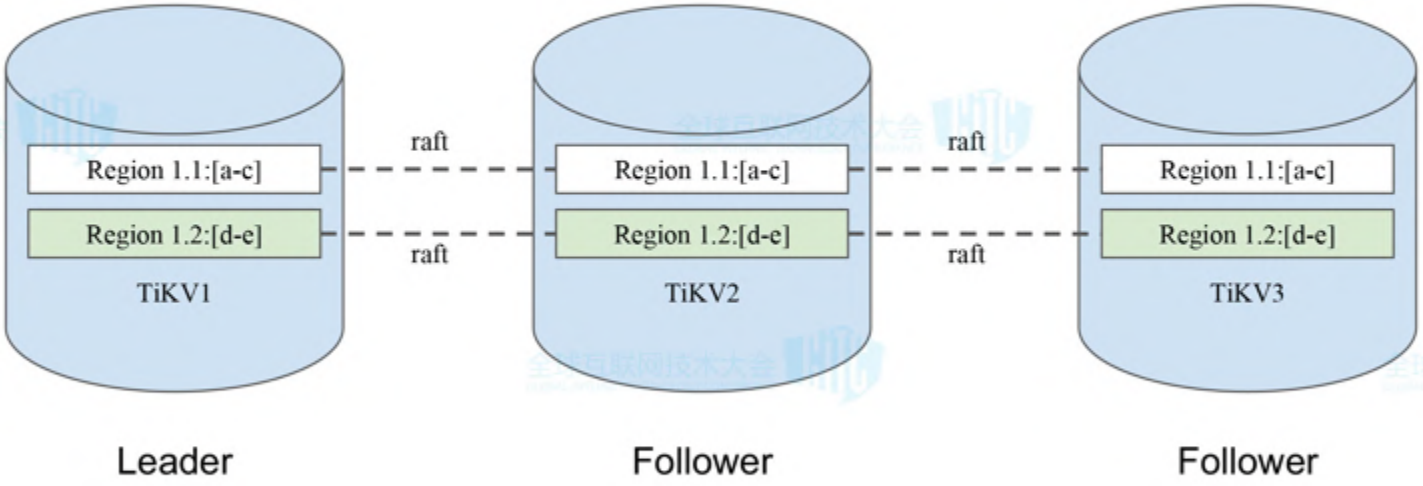
Safe Split: 2/4



Safe Split: 3/4

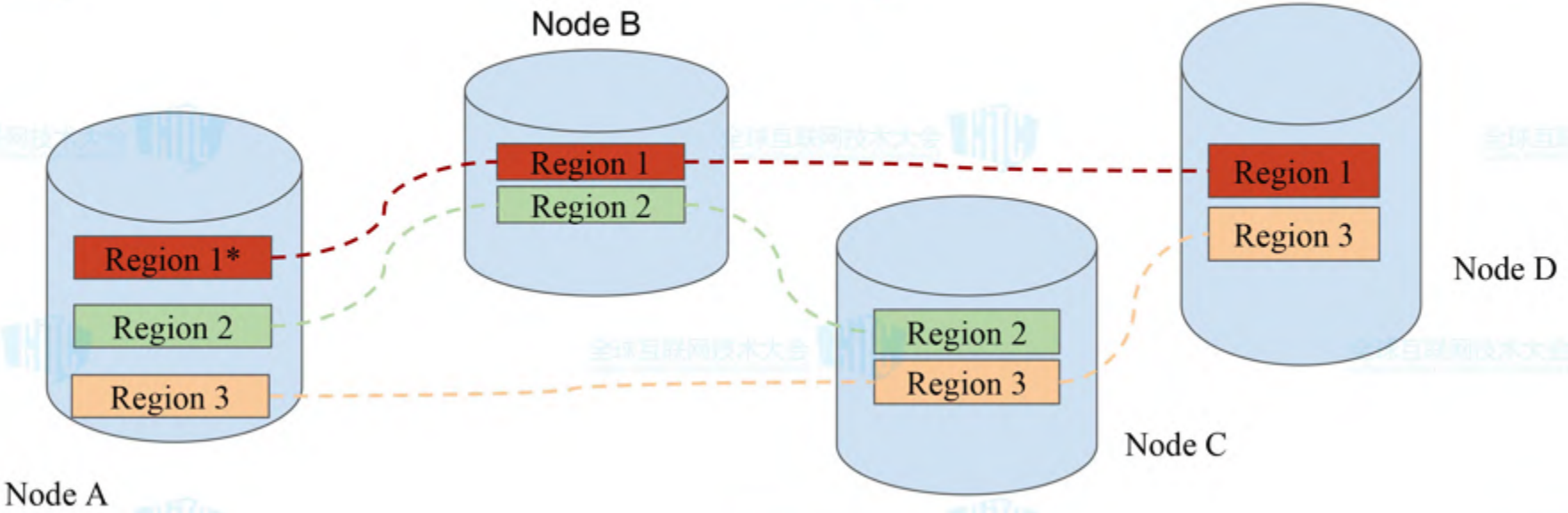


Safe Split: 4/4

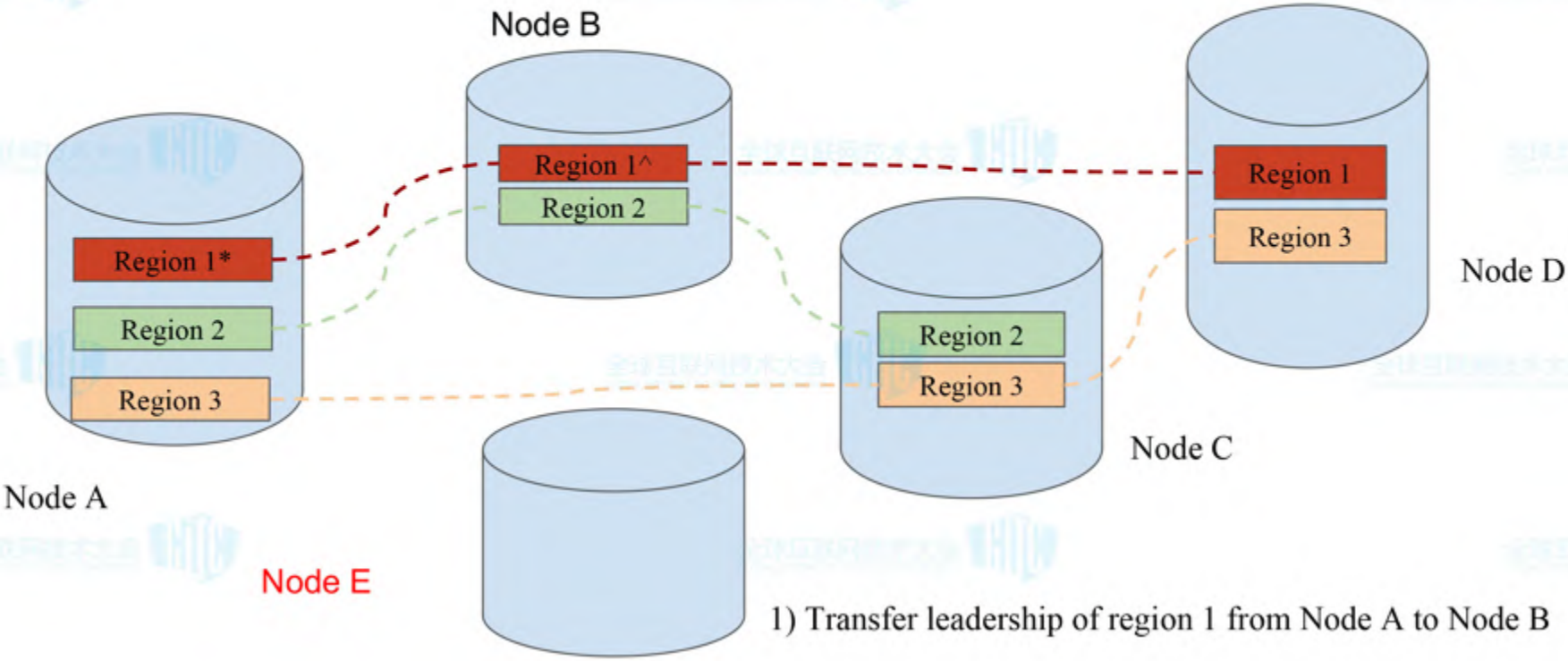


PingCAP

Scale-out (initial state)



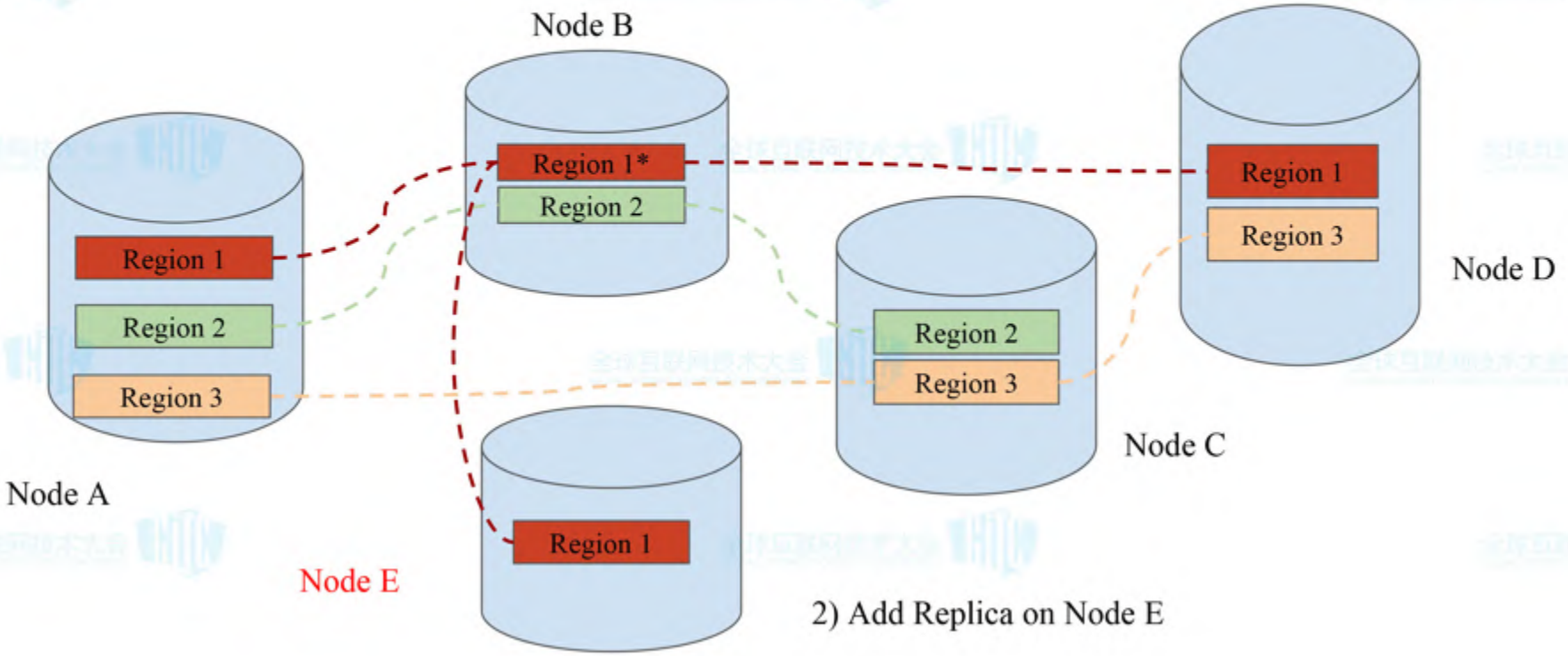
Scale-out (add new node)



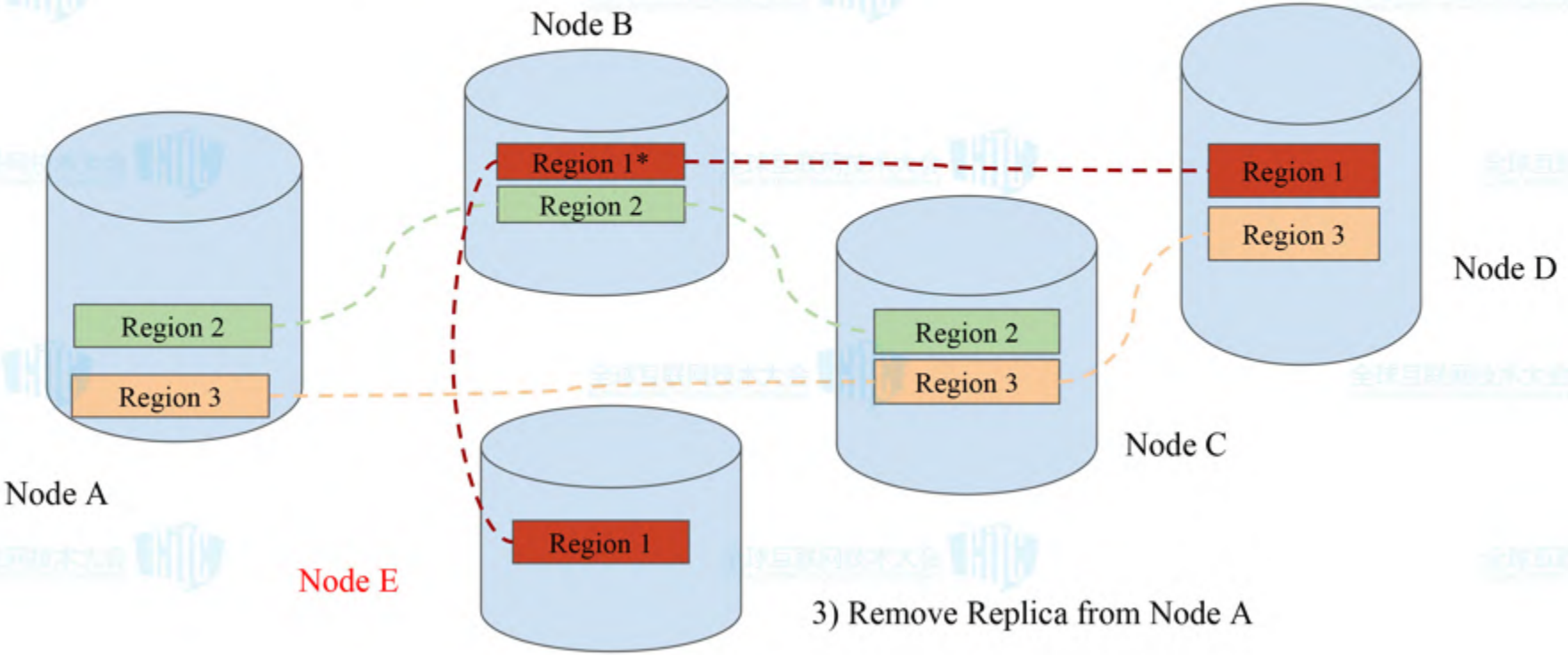
1) Transfer leadership of region 1 from Node A to Node B

PingCAP

Scale-out (balance)



Scale-out (balance)



ACID Transaction

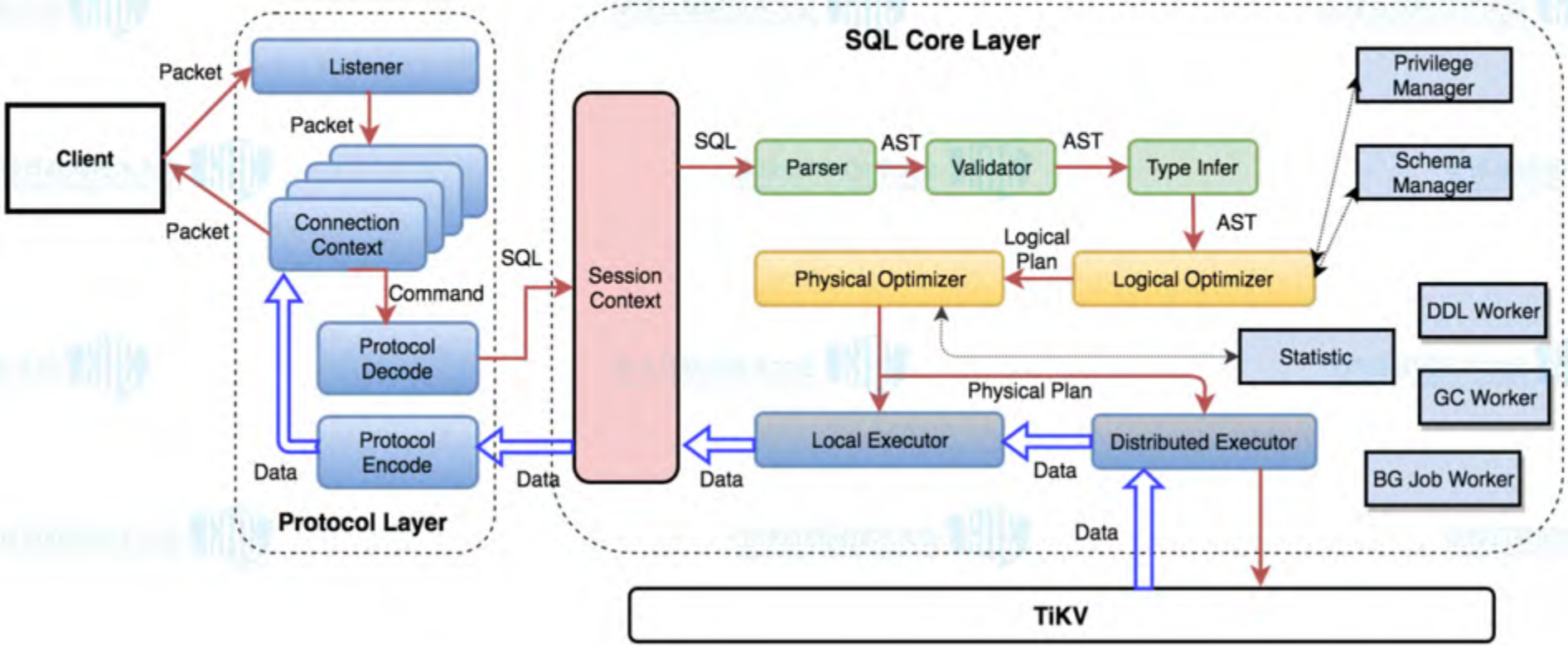
- Based on Google Percolator
- ‘Almost’ decentralized 2-phase commit
  - Timestamp Allocator
- Optimistic transaction model
- Default isolation level: Repeatable Read
- External consistency: Snapshot Isolation + Lock
  - SELECT … FOR UPDATE

Distributed SQL

- Full-featured SQL layer
- Predicate pushdown
- Distributed join
- Distributed cost-based optimizer (Distributed CBO)

PingCAP

TiDB SQL Layer overview

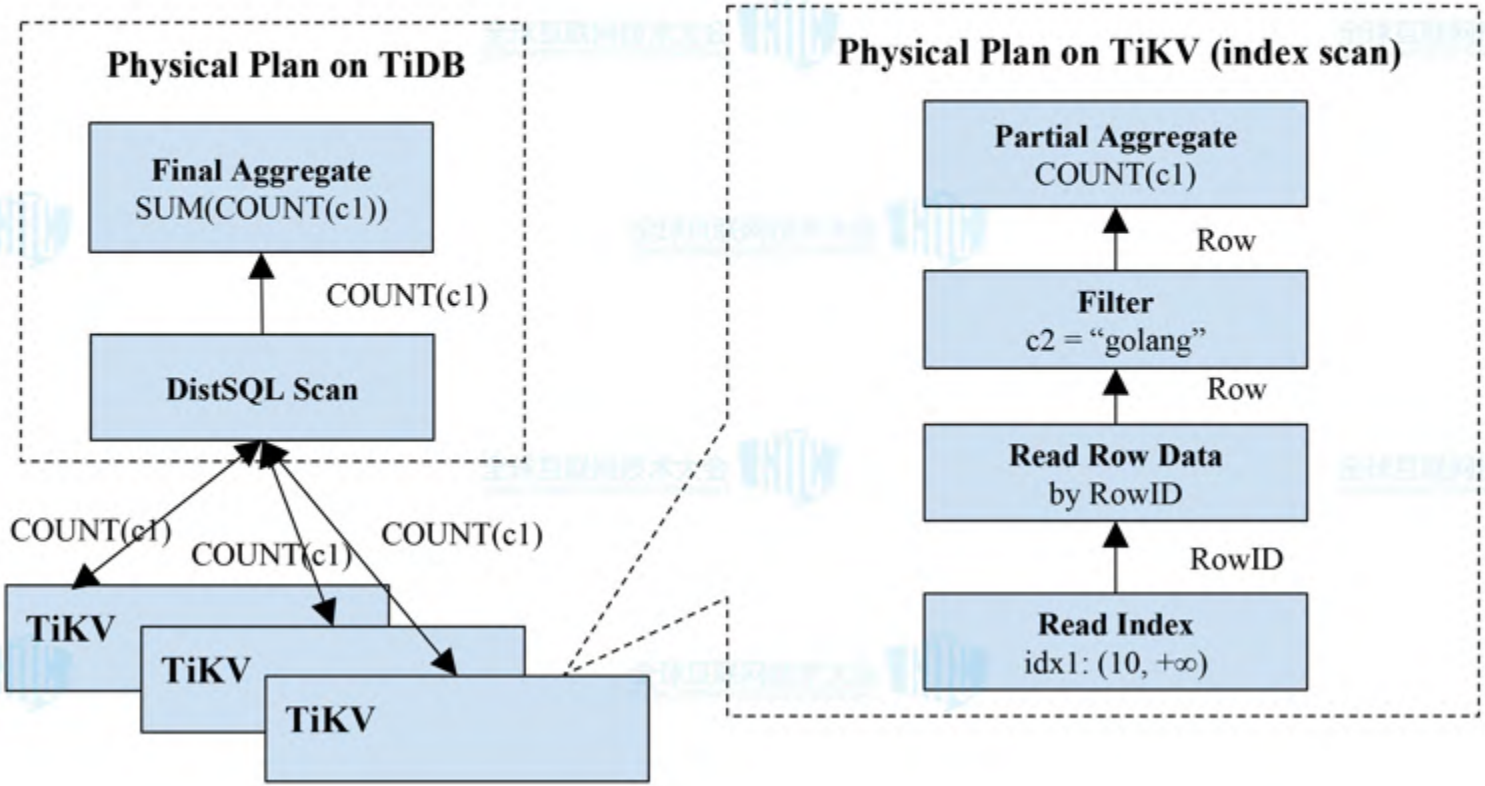


What happens behind a query

CREATE TABLE t (c1 INT, c2 TEXT, KEY idx_c1(c1));

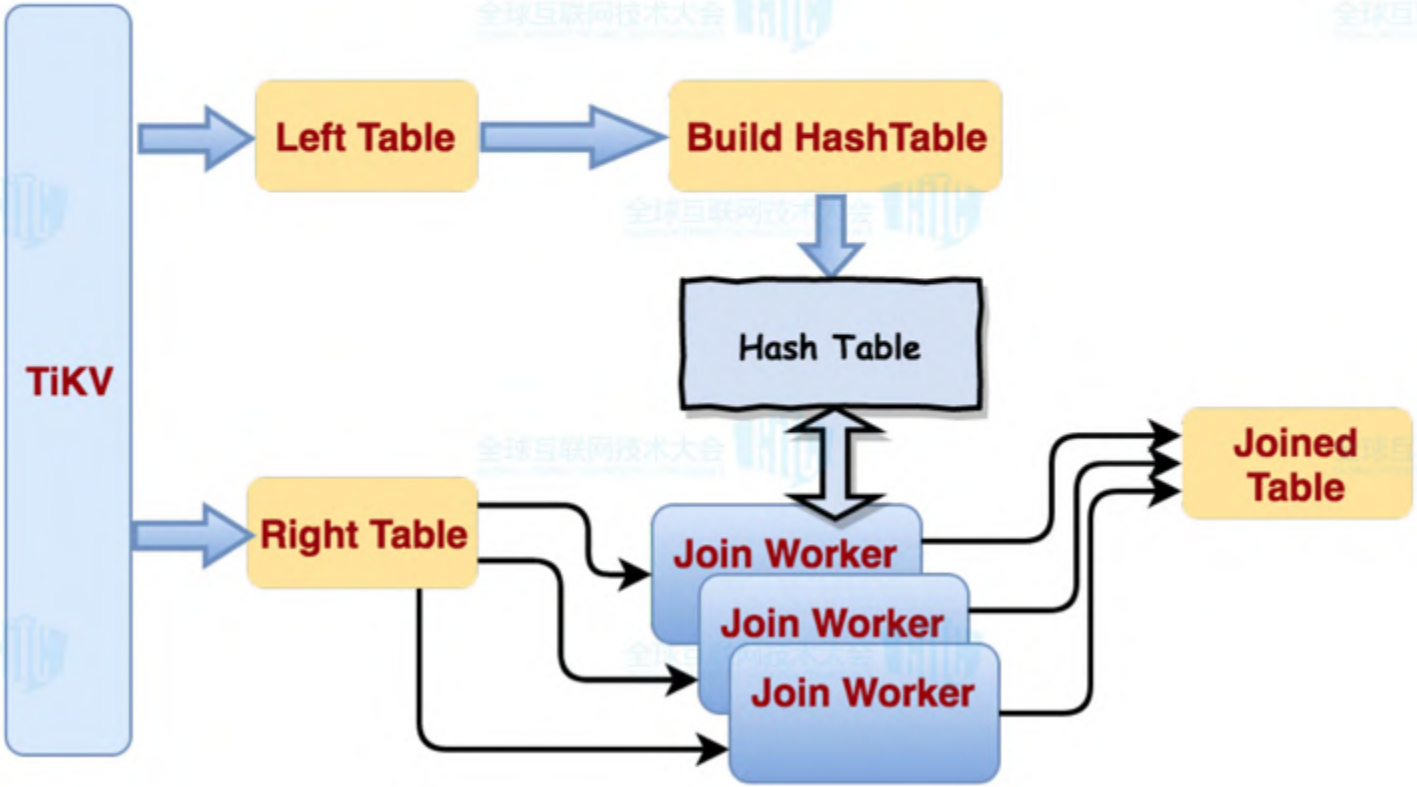
SELECT COUNT(c1) FROM t WHERE c1 > 10 AND c2 = 'golang';

Query Plan


Physical Plan on TiDB
Final Aggregate
SUM(COUNT(c1))
COUNT(c1)
DistSQL Scan
COUNT(c1)
COUNT(c1)
COUNT(c1)
TiKV
TiKV
TiKV
Physical Plan on TiKV (index scan)
Partial Aggregate
COUNT(c1)
Row
Filter
c2 = “golang”
Row
Read Row Data
by RowID
RowID
Read Index
idx1: (10, +∞)

PingCAP

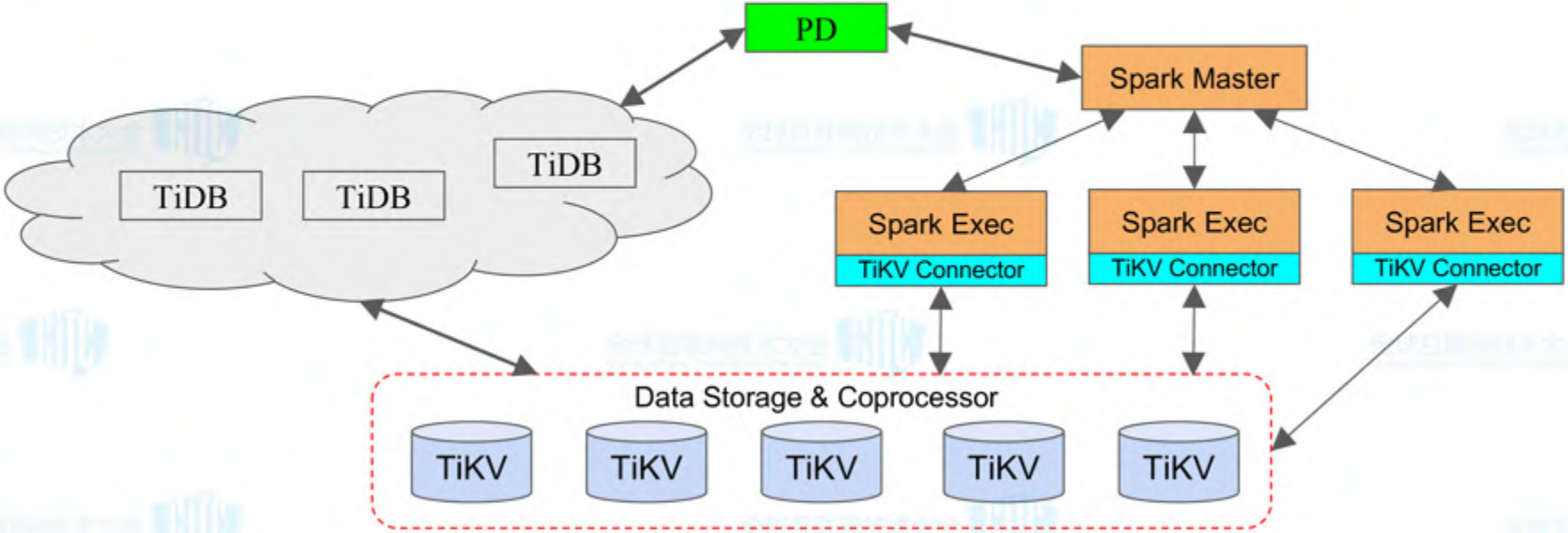
Distributed Hash Join



PingCAP

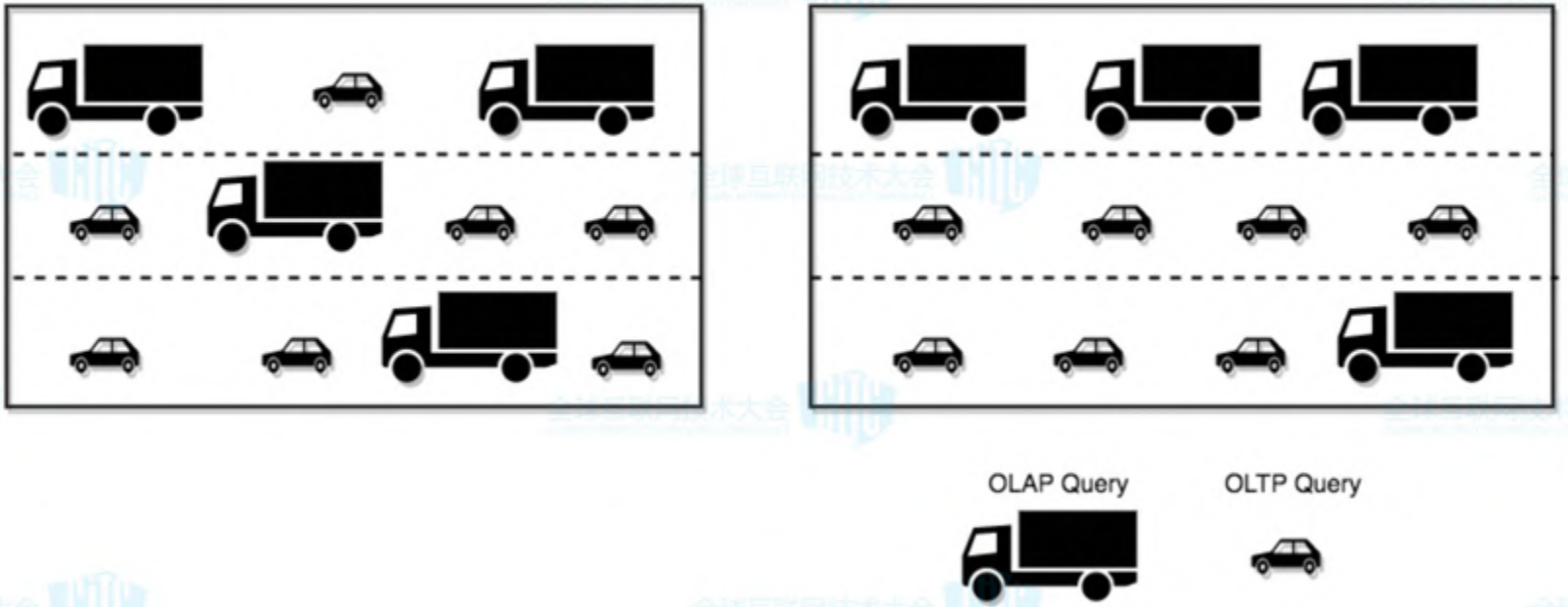
TiSpark

TiDB + SparkSQL = TiSpark



PingCAP

Hybrid Transactional/Analytical Processing



PingCAP

TiDB is a Cloud-Native Database

What is Cloud-Native

CNCF: Cloud Native Computing Foundation

Cloud native computing uses an open source software stack to be:

- Containerized
- Dynamically orchestrated
- Microservices oriented

Key points for a database:

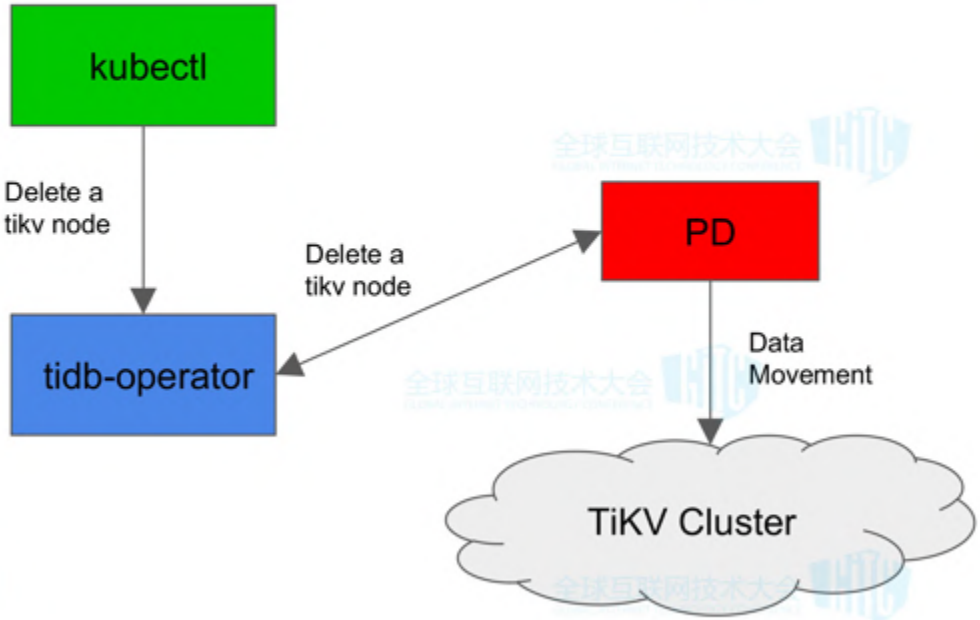
- Work well with container
- Horizontally scalable
- High availability & Auto-Failover
- Automate everything

TiDB with Kubernetes 1/3

- Kubernetes is winning the container war
- What's the hard part?
  - Stateless is Easy, Stateful is Hard
  - Application domain knowledge
  - IO Isolation
- tidb-operator (Inspired by etcd-operator)

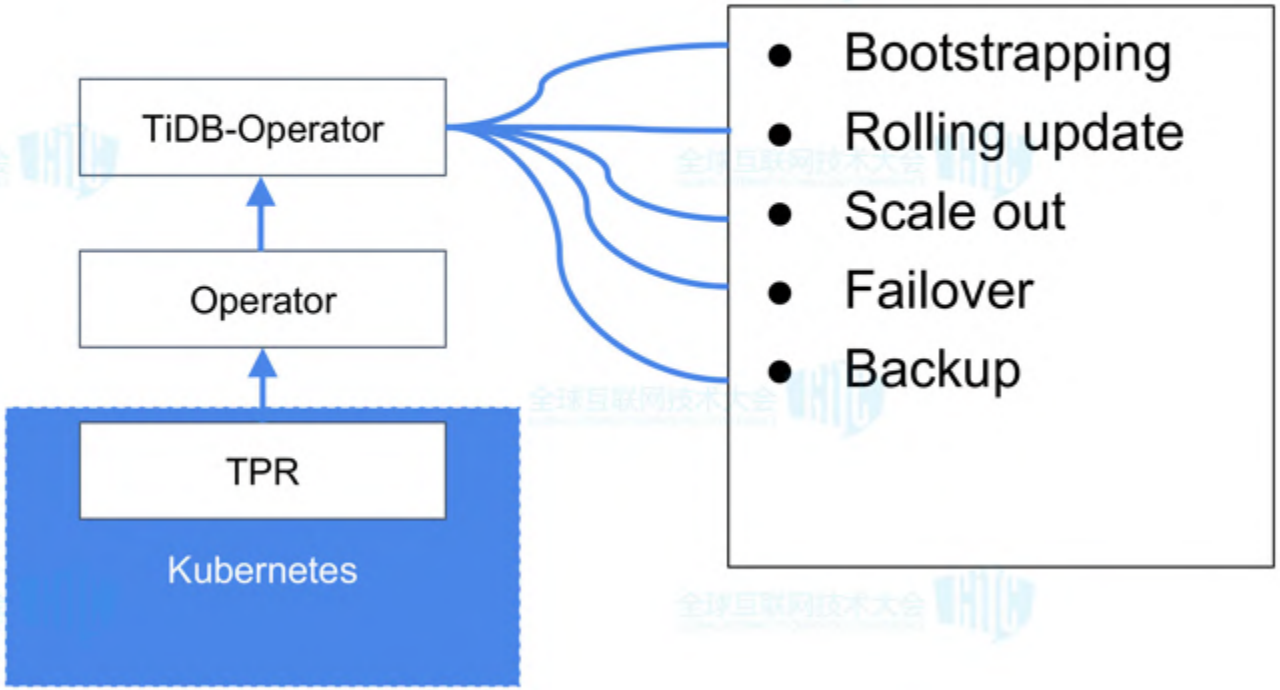
PingCAP

TiDB with Kubernetes 2/3



PingCAP

TiDB with Kubernetes 3/3



Roadmap

- TiSpark: Integrate TiKV with SparkSQL
- Better optimizer (Statistics && CBO)
- JSON type and document store for TiDB
  - MySQL 5.7.12+ X-Plugin
- Integrate with Kubernetes
  - TiDB Operator

PingCAP

Thanks

https://github.com/pingcap/tidb

https://github.com/pingcap/tikv

Contact me:

shenli@pingcap.com

GITC-PingCAP

Valid until 6/29 and will update upon joining group