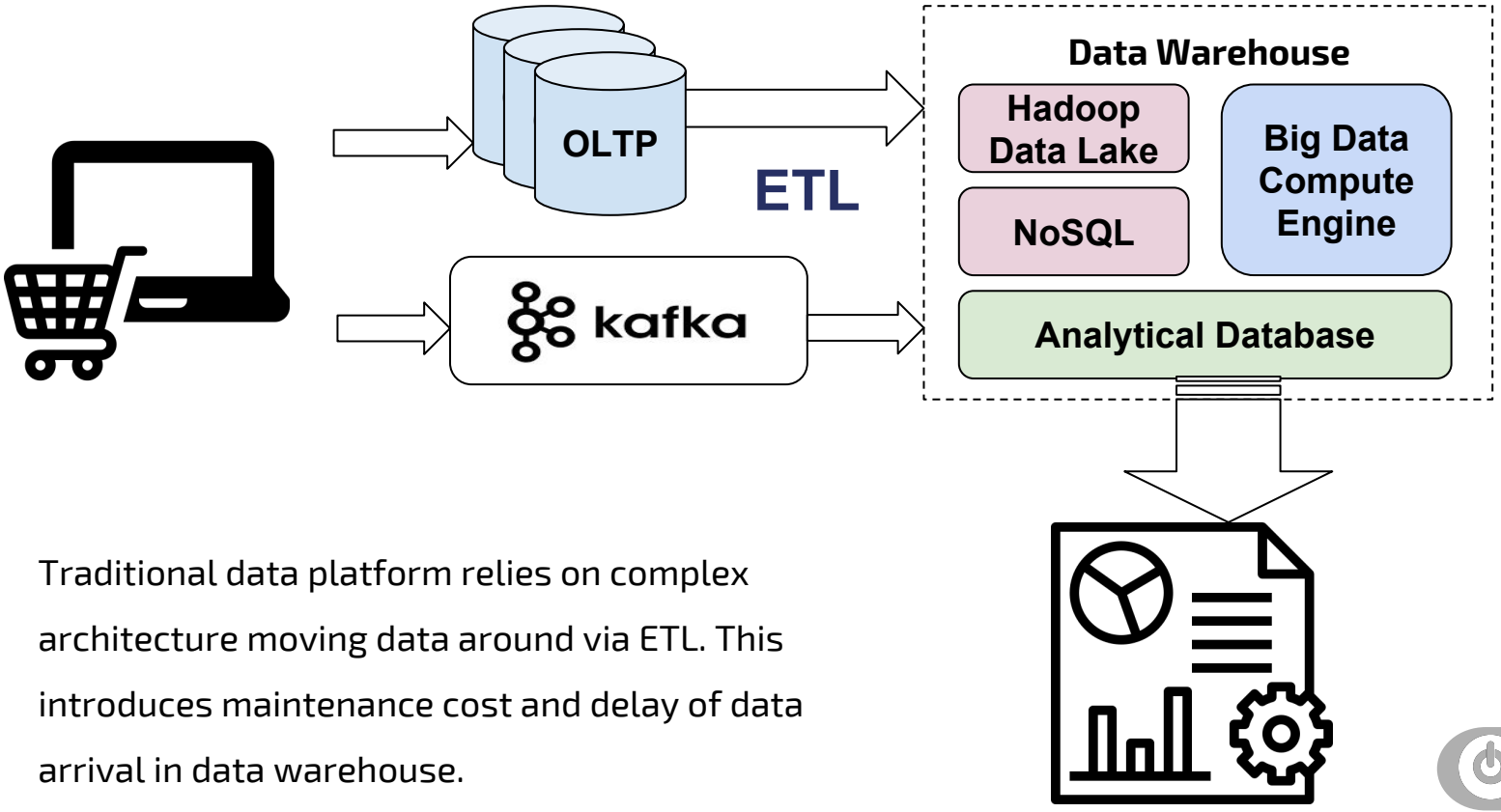
# Make HTAP Real with TiFlash

A TiDB native Columnar Extension

# About me

- Liu Cong, 刘聪
- Technical Director, Analytical Product@PingCAP
- Previously
  - Principal Enginer@QiniuCloud
  - Technical Director@Kingsoft
- Focus on distributed system and database engine

# Traditional Data Platform




Traditional data platform relies on complex architecture moving data around via ETL. This introduces maintenance cost and delay of data arrival in data warehouse.

# Fundamental Conflicts

- Large / batch process vs point / short access
  - Row format for OLTP
  - Columnar format for OLAP
- Workload Interference
  - A single large analytical query might cause disaster for your OLTP workload

# A Popular Solution

- Use different types of databases
  - For live and fast data, use an OLTP specialized database
  - For historical data, use Hadoop / analytical database
- Offload data via the ETL process into your Hadoop cluster or analytical database
  - Maybe once per day

# Good enough, really?

PingCAP

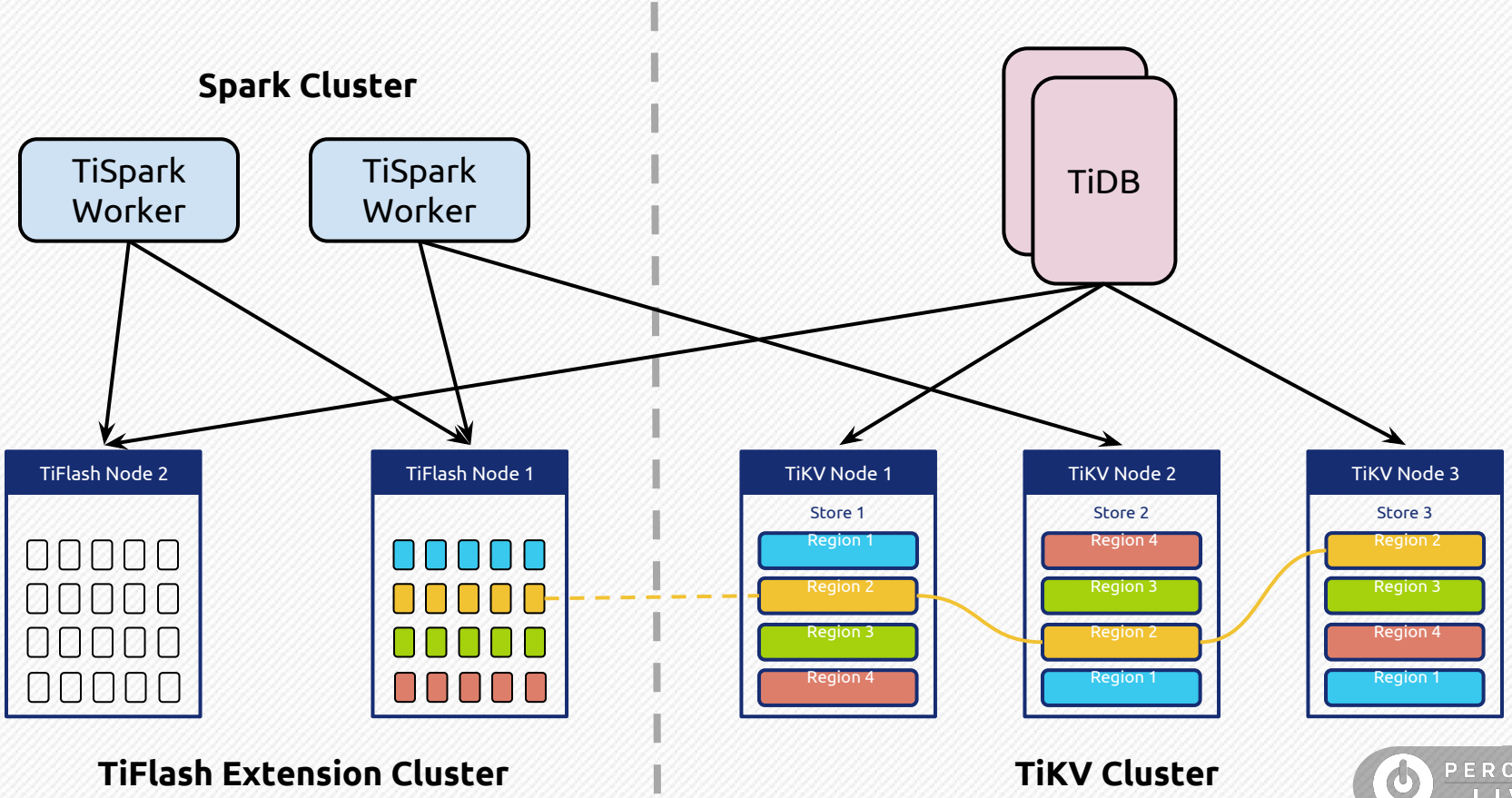
# TiFlash Extension

PingCAP

# What's TiFlash Extension

- TiFlash is an extended analytical engine for TiDB
- Powered by columnar storage and vectorized compute engine
- Tightly integrated with TiDB
- Clear isolation of workload not impacting OLTP
- Partially based on ClickHouse with tons of modifications
- Speed up read for both TiSpark and TiDB

# Architecture



# Columnstore vs Rowstore

- Columnar Storage stores data in columns instead of rows
  - Suitable for analytical workload
    - Possible for column pruning
  - Compression made possible and further IO reduction
    - ⅕ of average storage requirement
  - Bad small random IO
    - Which is the typical workload for OLTP
- Rowstore is the classic format for databases
  - Researched and optimized for OLTP scenario for decades
  - Cumbersome in analytical use cases

# Columnstore vs Rowstore

**Rowstore**

| id | name | age |
|---|---|---|
| 0962 | Jane | 30 |
| 7658 | John | 45 |
| 3589 | Jim | 20 |
| 5523 | Susan | 52 |

SELECT avg(age) FROM employee;

**Columnstore**

| id |
|---|
| 0962 |
| 7658 |
| 3589 |
| 5523 |

| name |
|---|
| Jane |
| John |
| Jim |
| Susan |

| age |
|---|
| 30 |
| 45 |
| 20 |
| 52 |

Usually you don't read all columns in a table performing analytics.

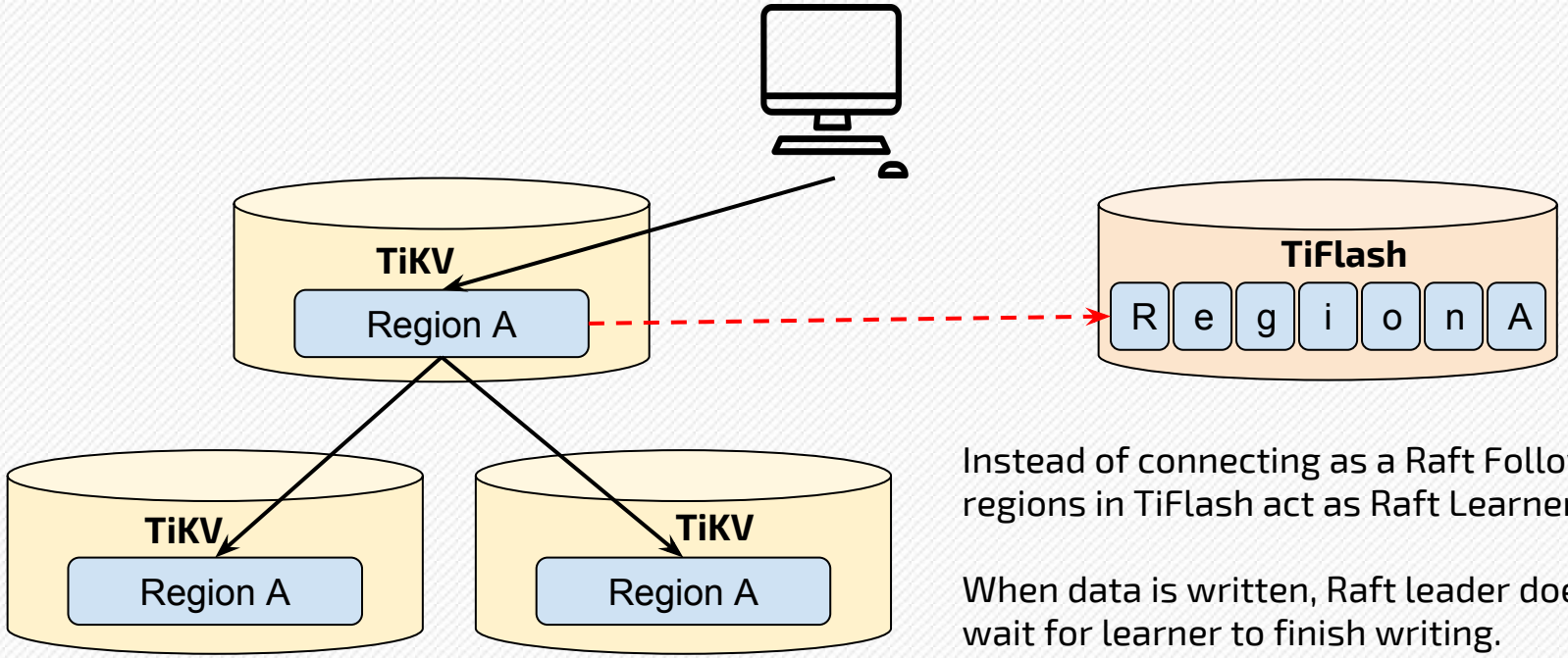
In columnstore, you avoid unnecessary IO while you have to read them all in rowstore.

# Raft Learner

TiFlash synchronizes data in columnstore via Raft Learner

- Strong consistency on read enabled by the Raft protocol
- Introduce almost zero overhead for the OLTP workload
  - Except the network overhead for sending extra replicas
  - Slight overhead on read (check Raft index for each region in 96 MB by default)
  - Possible for multiple learners to speed up hot data read

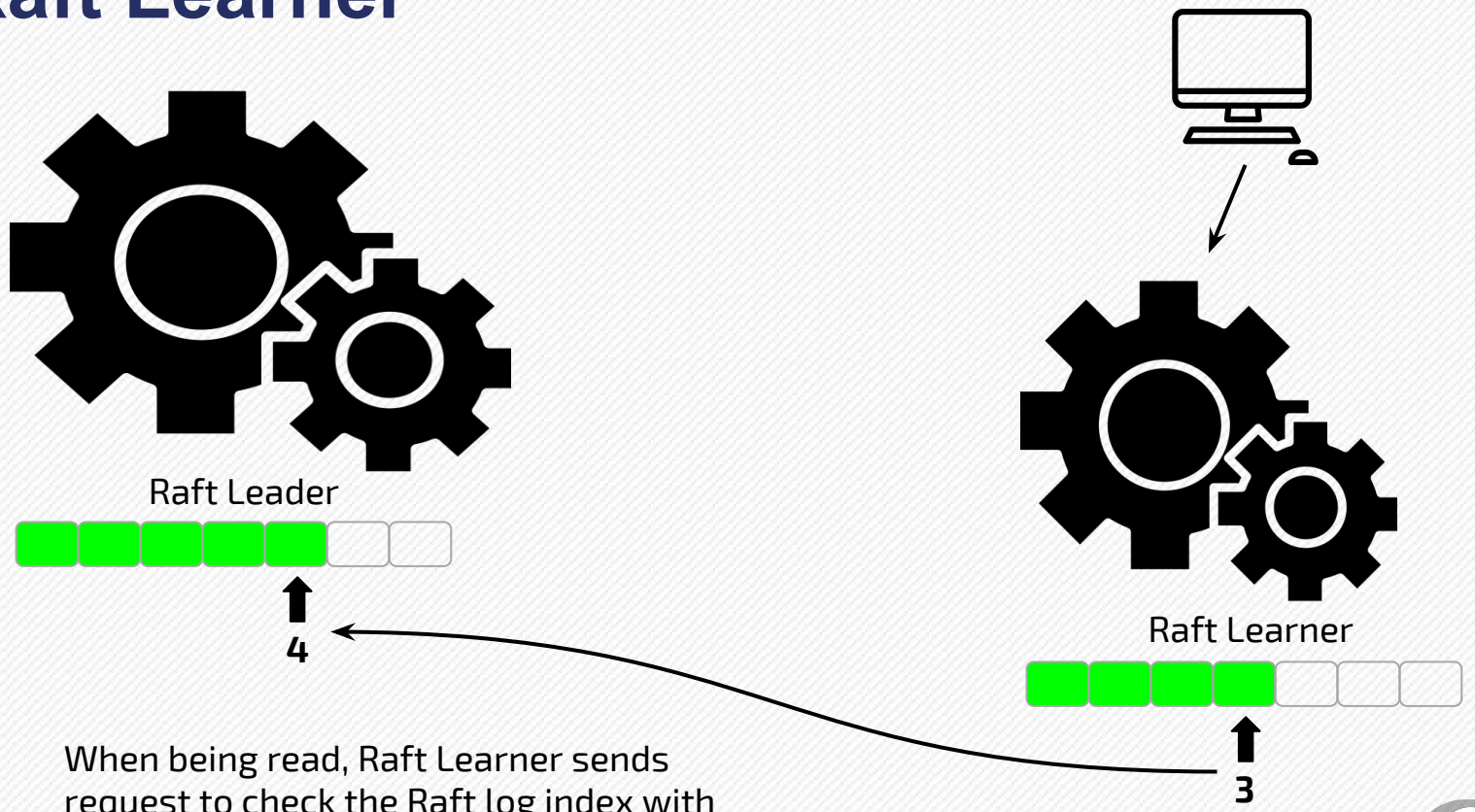
# Raft Learner




Instead of connecting as a Raft Follower, regions in TiFlash act as Raft Learner.

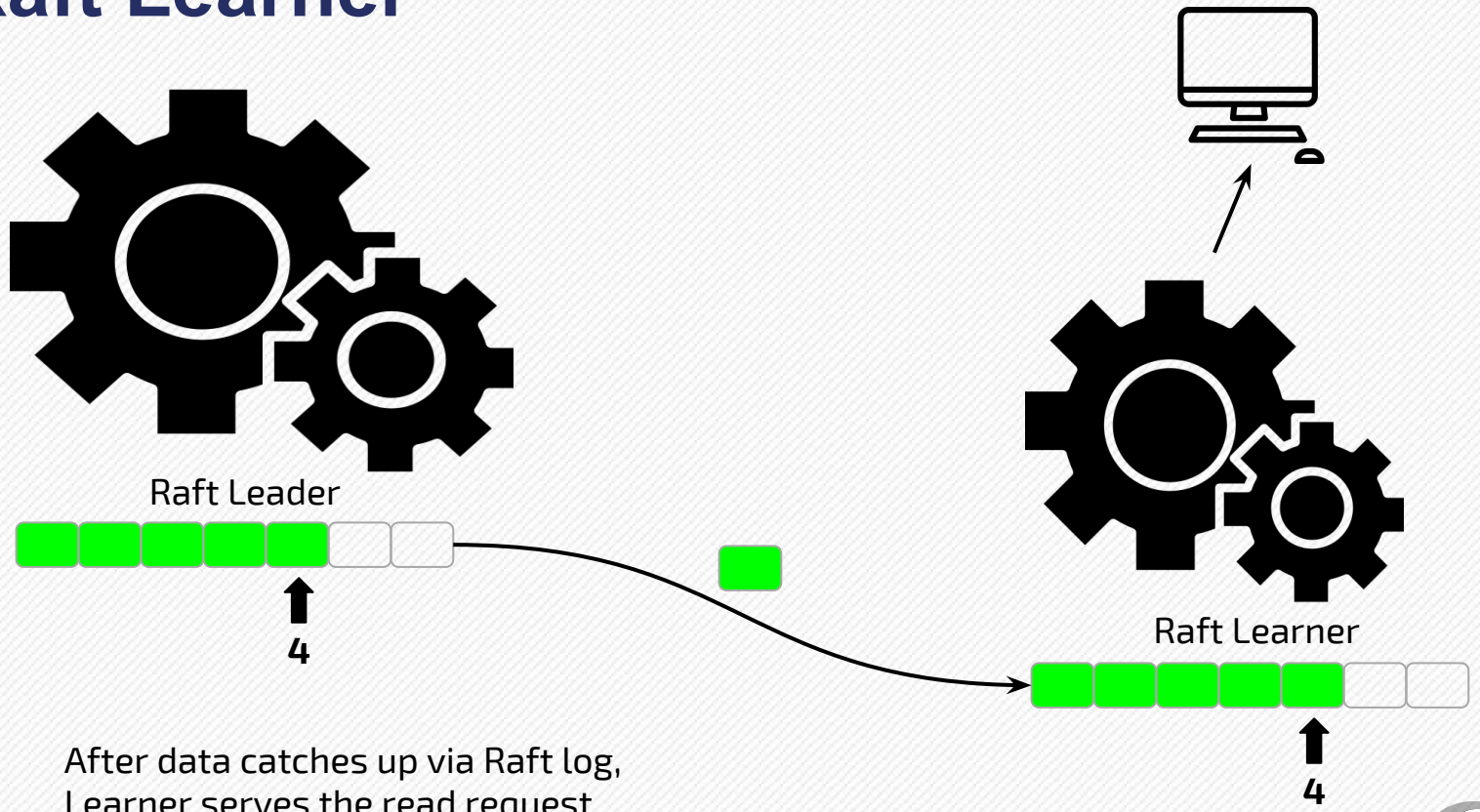
When data is written, Raft leader does not wait for learner to finish writing. Therefore, TiFlash introduces almost no overhead replicating data.

# Raft Learner




When being read, Raft Learner sends request to check the Raft log index with Leader to see if its data is up-to-date.

# Raft Learner




After data catches up via Raft log, Learner serves the read request then.

PingCAP

# TiFlash is beyond columnar format

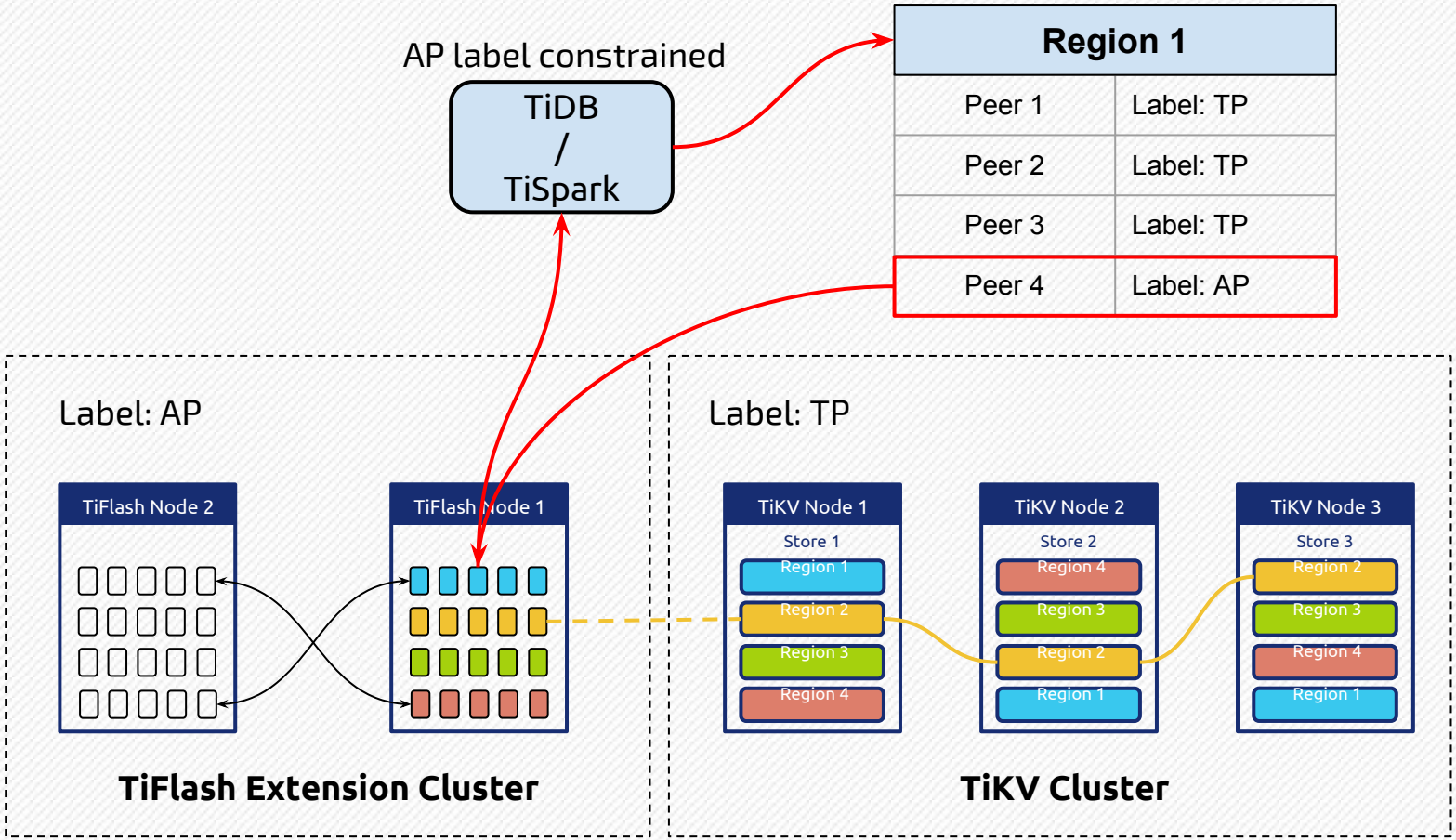
# Scalability

- An HTAP database needs to store huge amount of data
- Scalability is very important
- TiDB relies on multi-raft for scalability
  - One command to add / remove node
  - Scaling is fully automatic
  - Smooth and painless data rebalance
- TiFlash adopts the same design

# Isolation

- Perfect resource isolation
- Data rebalance based on the “label” mechanism
  - Dedicated nodes for TiFlash / Columnstore
  - TiFlash nodes have their own AP label
  - Rebalance between AP label nodes
- Computation Isolation is possible by nature
  - Use a different set of compute nodes
  - Read only from nodes with AP label

# Isolation

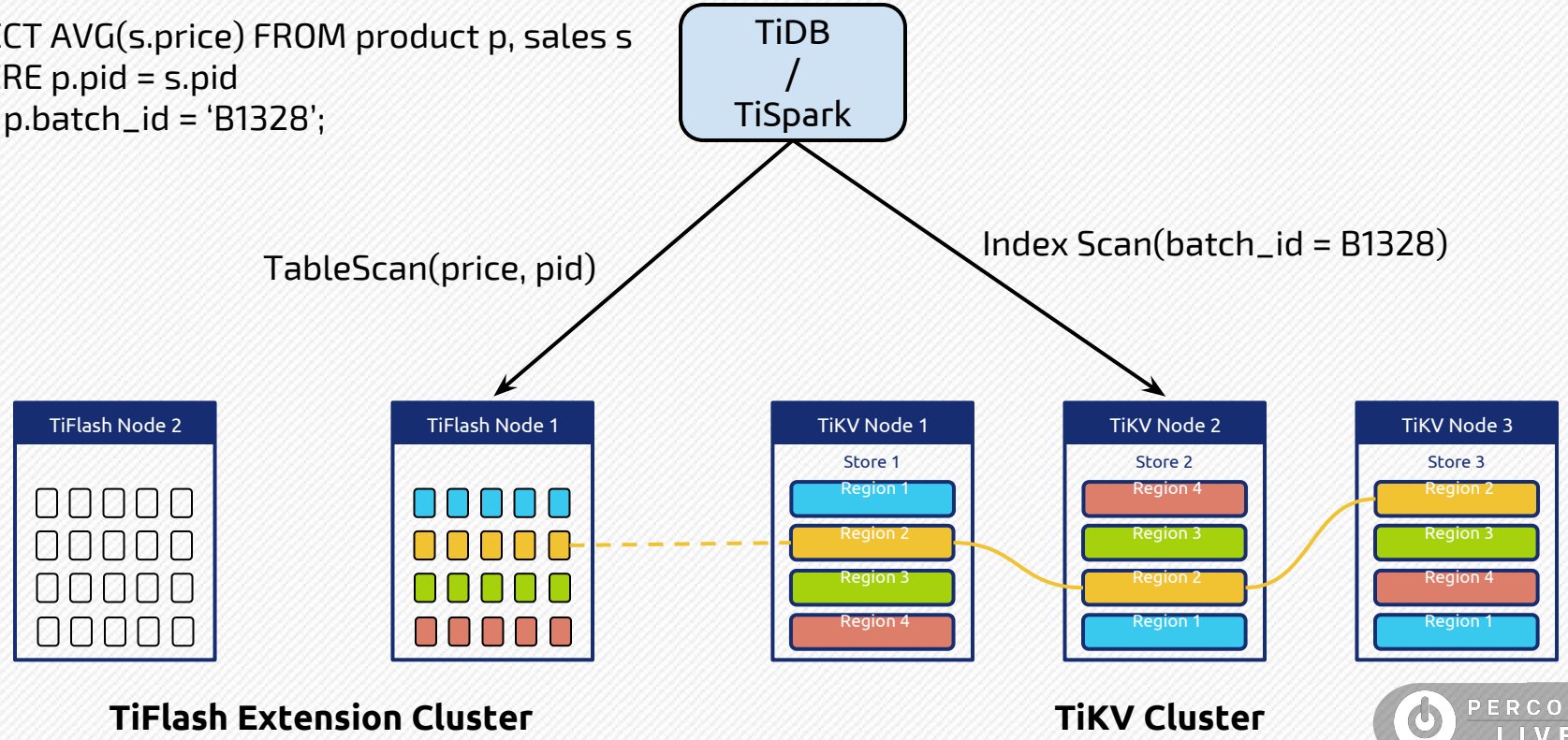


# Integration

- TiFlash Tightly Integrated with TiDB / TiSpark
  - TiDB / TiSpark might choose to read from either side
    - Based on cost
  - When reading TiFlash replica failed,  read TiKV replica transparently
  - Join data from both sides in a single query

# Integration

```
SELECT AVG(s.price) FROM product p, sales s
WHERE p.pid = s.pid
AND p.batch_id = 'B1328';
```



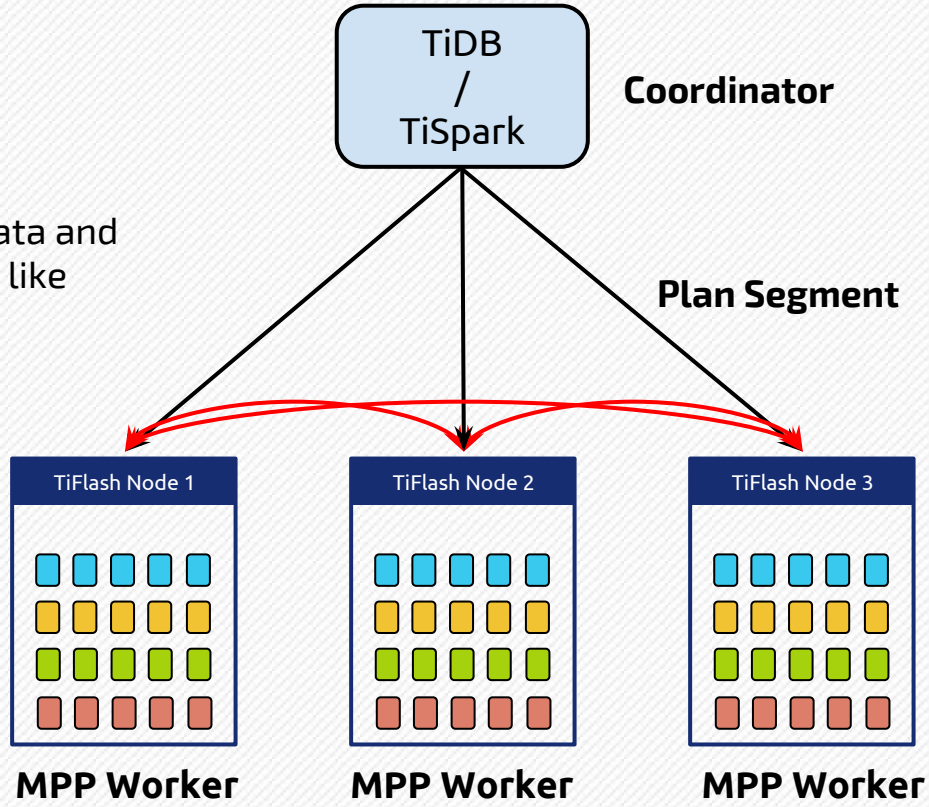
# MPP Support

- TiFlash nodes form a MPP cluster by themselves

- Full computation support on MPP layer
  - Speed up TiDB since it is not MPP design
  - Speed up TiSpark by avoiding writing disk during shuffle

# MPP Support

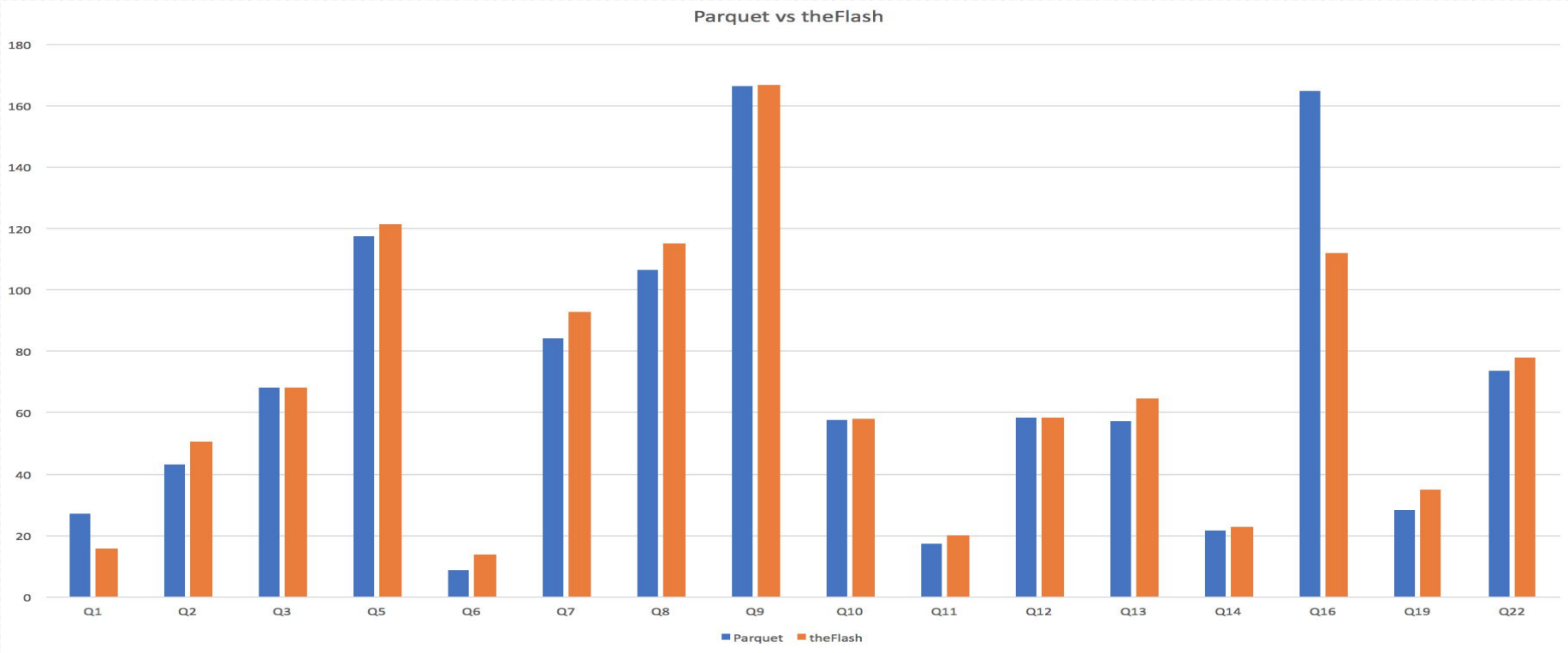
TiFlash nodes exchange data and enable complex operators like distributed join.



# Performance

- Comparable performance against vanilla Spark on Hadoop + Parquet
  - Benchmarked with Pre-Alpha version of TiFlash + Spark (without MPP support)
  - TPC-H 100

# Performance



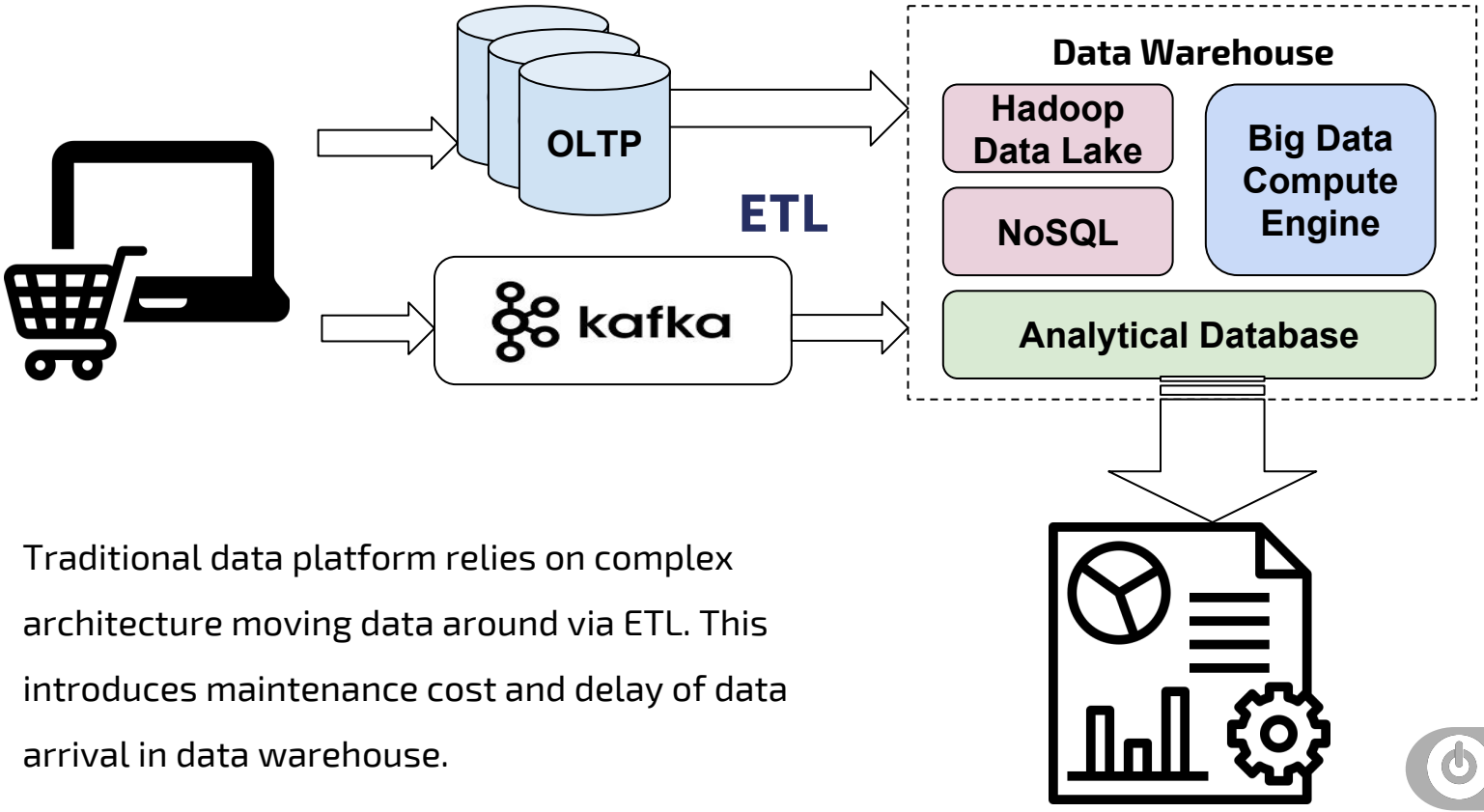

Parquet

TiFlash

PingCAP

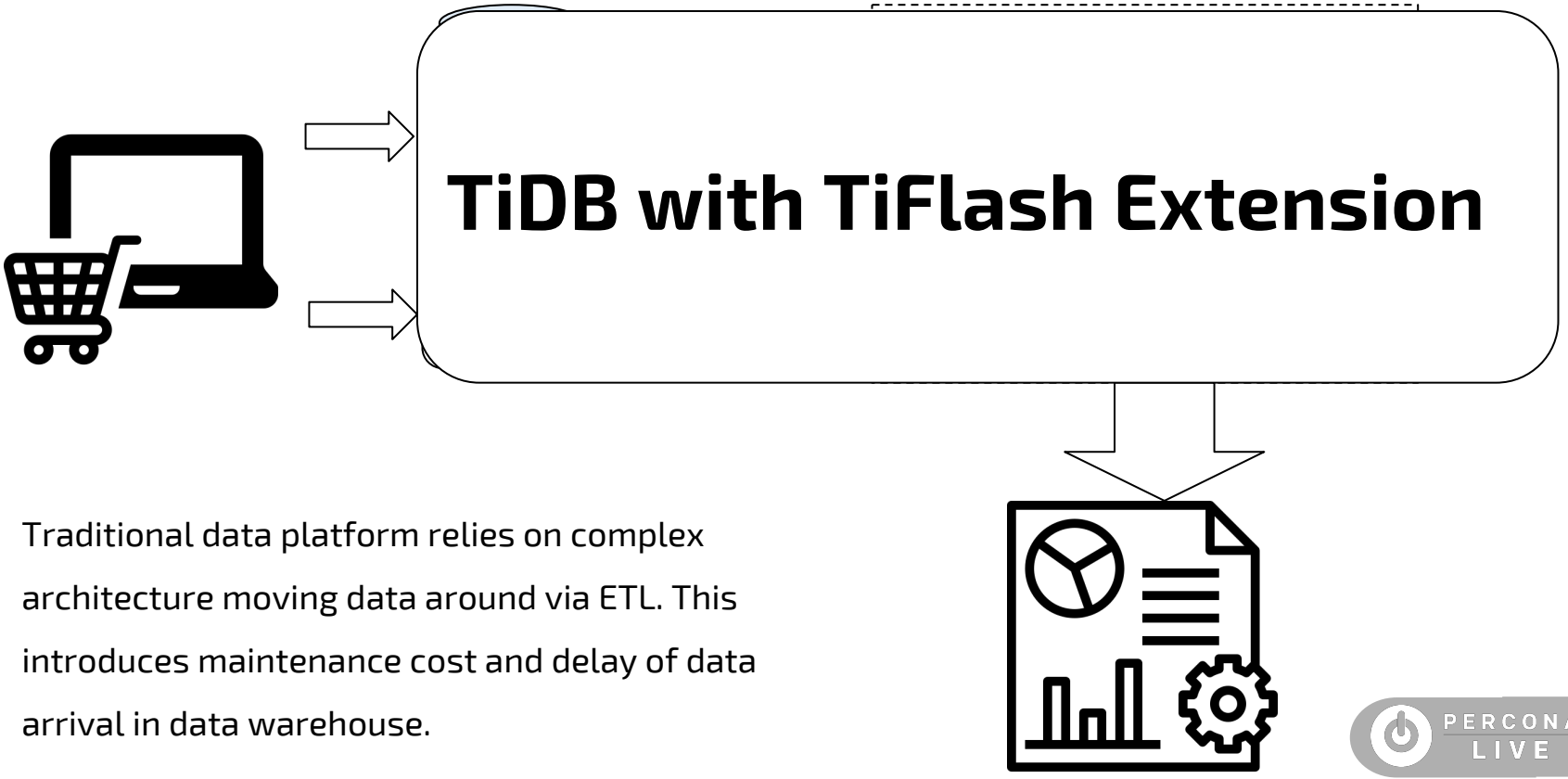
# TiDB Data Platform

# Traditional Data Platform




Traditional data platform relies on complex architecture moving data around via ETL. This introduces maintenance cost and delay of data arrival in data warehouse.

# TiDB Data Platform




Traditional data platform relies on complex architecture moving data around via ETL. This introduces maintenance cost and delay of data arrival in data warehouse.

PingCAP

# Fundamental Change

- “What happened yesterday” vs “What’s going on right now”
- Realtime report for sales campaign and adjust price in no time
  - Risk management with up-to-date info always
  - Very fast paced replenishment based on live data and prediction

# Roadmap

- Beta / User POC in May, 2019
  - With columnar engine and isolation ready
    - Access only via Spark
- GA, By the end of 2019
  - Unified coprocessor layer
    - Ready for both TiDB / TiSpark
    - Cost based access path selection
  - Possibly MPP layer done

# Thanks!

Contact us:

www.pingcap.com