# 02 TiDB HTAP**技术**

[02 TiDB HTAP**技术**](#_bookmark0)

[1**、**HTAP**技术**](#_bookmark1)

* 1. [HTAP技术简介](#_bookmark2)
     1. [OLTP和OLAP](#_bookmark3)
     2. [什么是HTAP](#_bookmark6)
     3. [HTAP技术需求](#_bookmark5) [TiDB的HTAP架构](#_bookmark4)
  2. [MPP介绍](#_bookmark7)
     1. [MPP适合的场景](#_bookmark8)
     2. [MPP计算简介](#_bookmark9)
     3. [使⽤MPP模式](#_bookmark10)
     4. [MPP 模式算法⽀持](#_bookmark11)
  3. [HTAP场景](#_bookmark12)
     1. [混合负载场景](#_bookmark13)
     2. [实时流处理场景](#_bookmark14)

[1.2.3 数据中枢场景](#_bookmark15)

[2**、**TiFlash](#_bookmark16)

* 1. [TiFlash架构](#_bookmark17)
  2. [核⼼特性](#_bookmark18)
     1. [异步复制](#_bookmark20)
     2. [⼀致性读取](#_bookmark19)
     3. [智能选择](#_bookmark21)
  3. [TiFlash的使⽤](#_bookmark22)
     1. [部署&扩缩容](#_bookmark23)
        1. [部署](#_bookmark25)
        2. [扩容](#_bookmark24)
        3. [缩容](#_bookmark29)
     2. [使⽤](#_bookmark28)
        1. [按表构建 TiFlash 副本](#_bookmark27)
        2. [按库构建 TiFlash 副本](#_bookmark26)
        3. [Engine 隔离](#_bookmark33)
        4. [⼿⼯ Hint](#_bookmark32)
     3. [TiFlash调优](#_bookmark31)
     4. [TiFlash 集群监控](#_bookmark30)
     5. [TiFlash计算下推与不⽀持场景](#_bookmark35)
     6. [TiFlash常⻅问题的处理](#_bookmark34)

1**、**HTAP**技****术**

* 1. HTAP**技术简介** 
     1. OLTP**和**OLAP

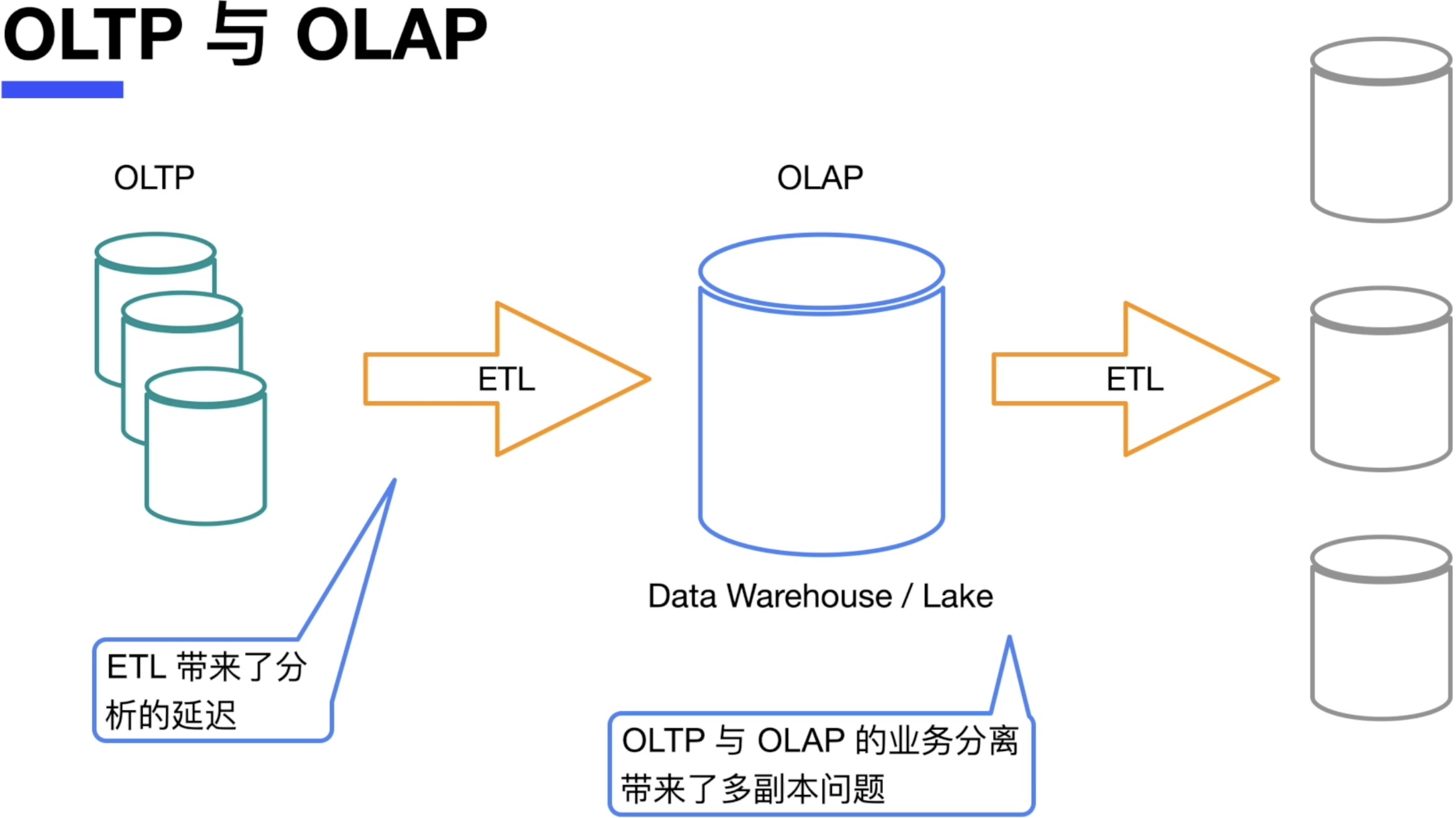


OLAP（On-Line Analytical Processing）联机分析处理，应⽤在数据仓库，强调的是数据分析。

OLTP（On-Line Transaction Processing）联机事务处理，主要应⽤是传统关系型数据库。OLTP系统强调的是内存效率，实时性⽐较⾼。

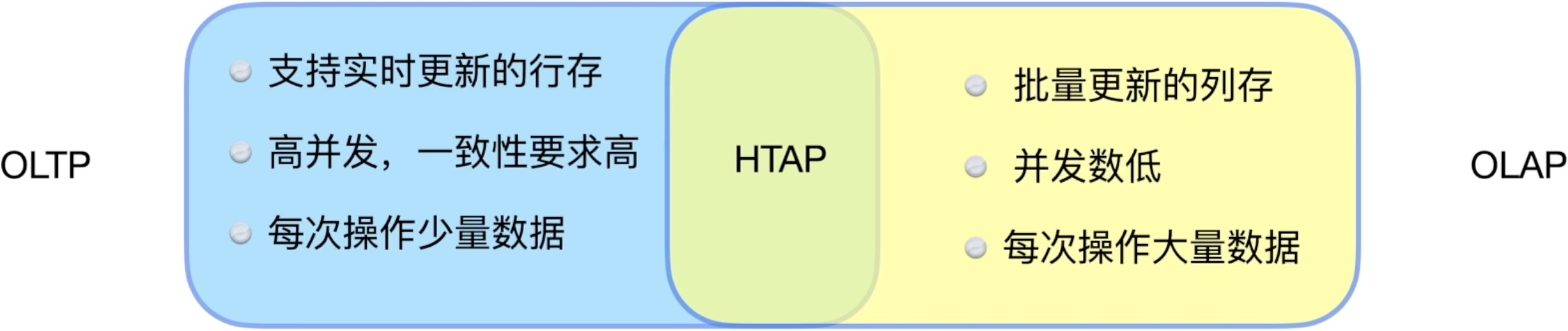
传统⽅案：

1. 线上OLAP数据库，⾏存，⾼并发
2. 数仓负责ETL+OLAP进⾏离线分析和近实时分析
   * ETL:根据技术种类不同延时,T+1,or 批处理(30min)
   * 带来了⼤量的数据冗余



* + 1. **什么是**HTAP

1. ⽀持OLTP和OLAP业务
   * 报表分析:AP
   * 订单,交易:TP



* + 1. HTAP**技术需求**

#1.可扩展

* 分布式事务
* 分布式存储

#2.同时⽀持TP和AP

* 同时⽀持⾏存和列存
* TP和AP业务隔离

#3.实时性

* ⾏存和列存数据实时同步

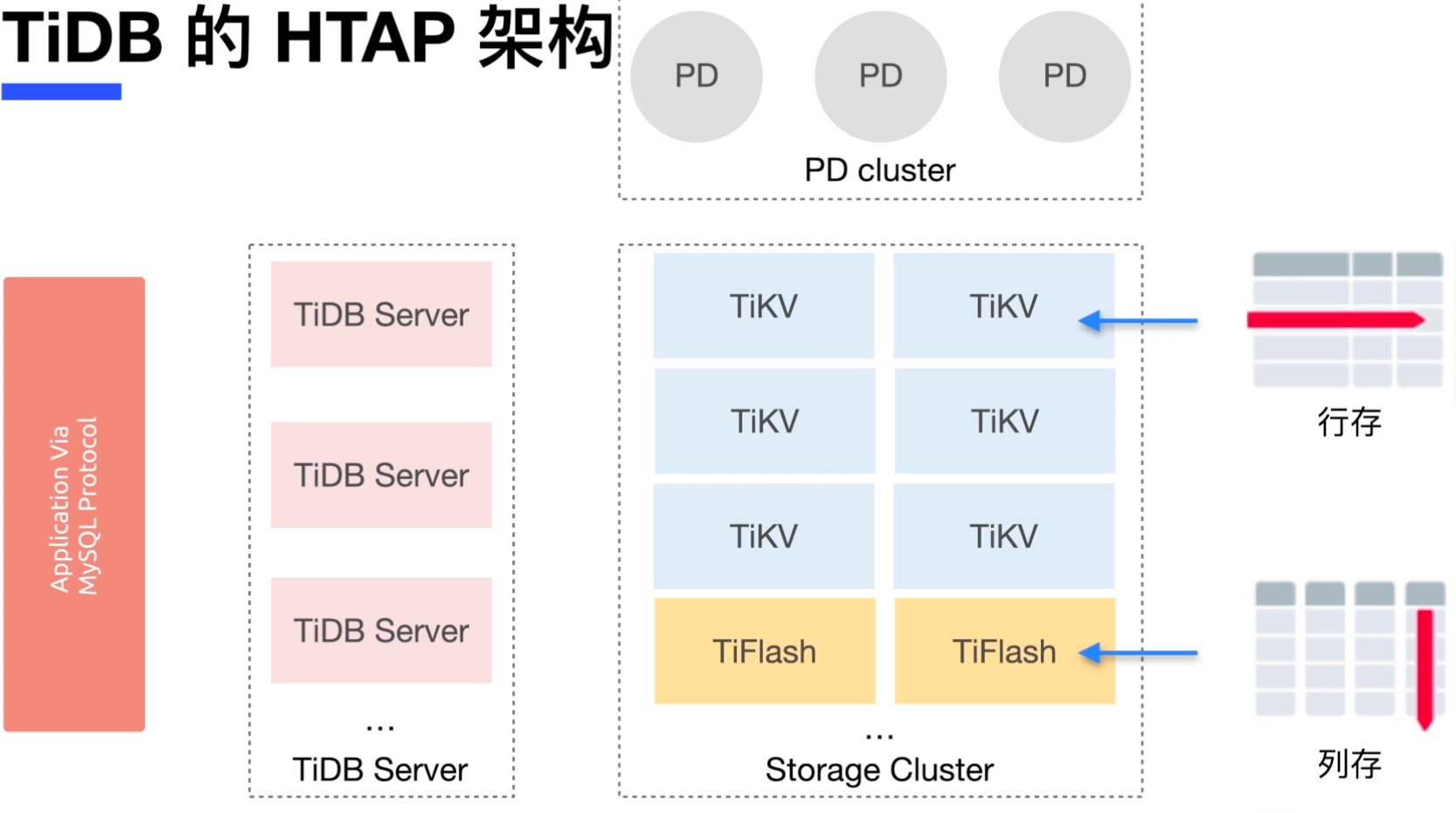
TiDB**的**HTAP**架构**

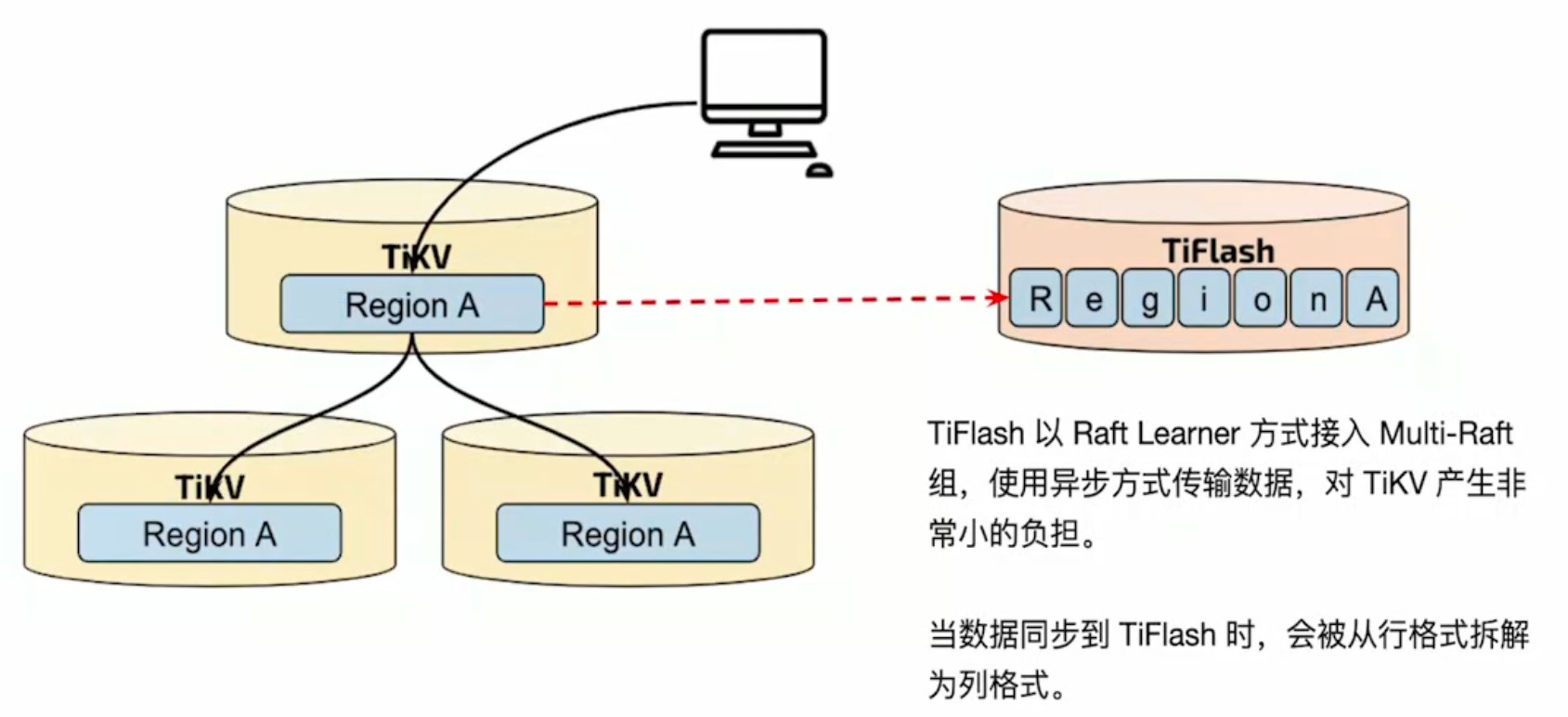
通过引进tiflash+ raft learner

1. 复制tikv数据(准实时)
2. ⾏存数据转化列存数据
3. TiDB 优化器根据SQL特征选择存储引擎(可⼈⼯⼲预),但实际应⽤需要⼤量的经验铺垫

#特性

1. ⾏列混合
   * 基于主键的tiflash实时更新
   * 作为副本
   * 业务隔离
2. CBO or ⼈⼯选择
3. MPP架构



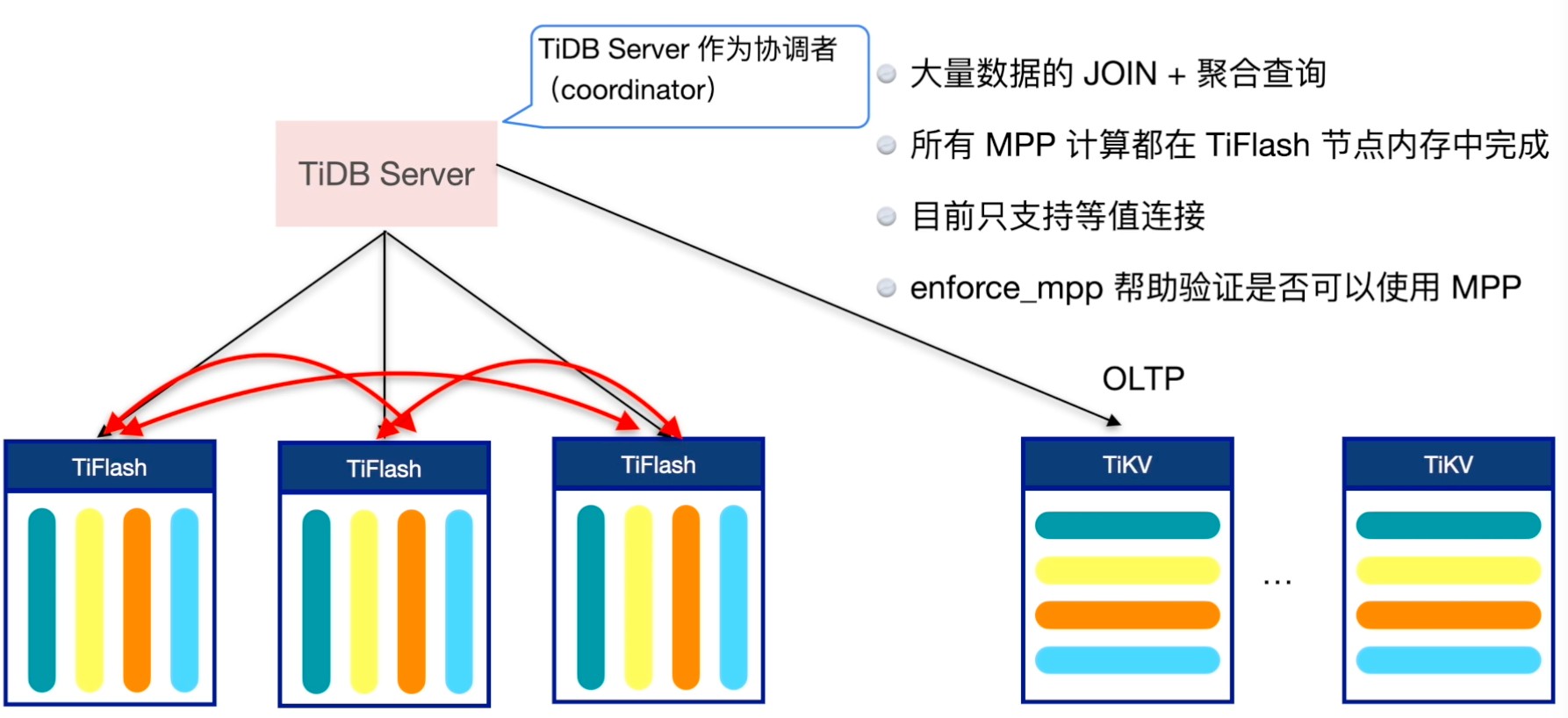


* 1. MPP**介****绍** 
     1. MPP**适合的场景**

1. 每⼀个tiflash负责过滤 数据交换 表连接 聚合,
2. 每个tiflash叫mpp worker

#适合场景

1. ⼤量数据的join+聚合查询
2. 所有mpp计算都在tiflash节点"内存中"完成,则tiflash宕机会导致本次查询中断
3. 仅⽀持等值连接，eg： a join b on a.id=b.id 4.ecforce\_mpp帮助验证是否亏使⽤mpp



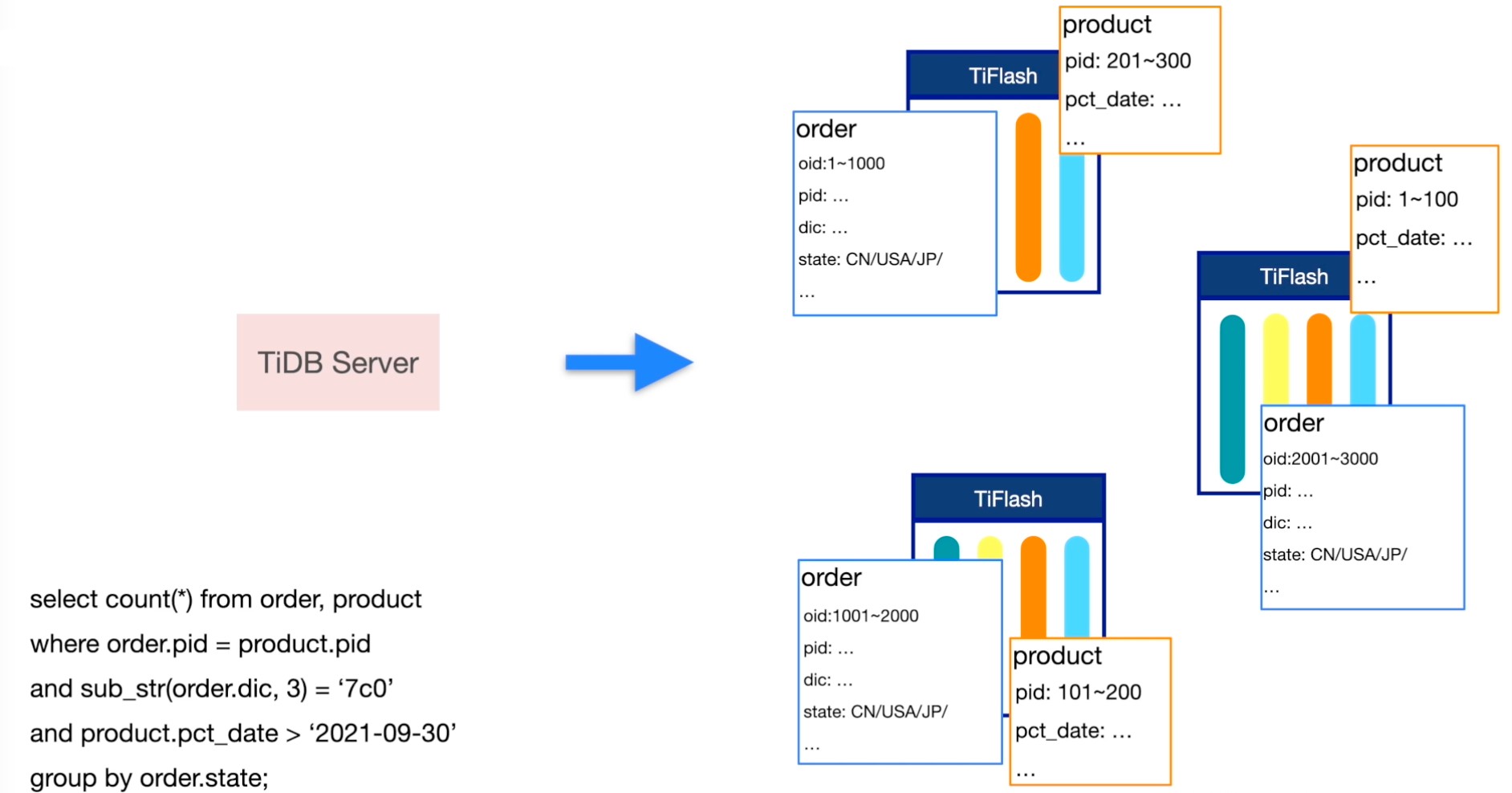
* + 1. MPP**计算简介**

1.sql count(*\*)*聚合 *join pid*

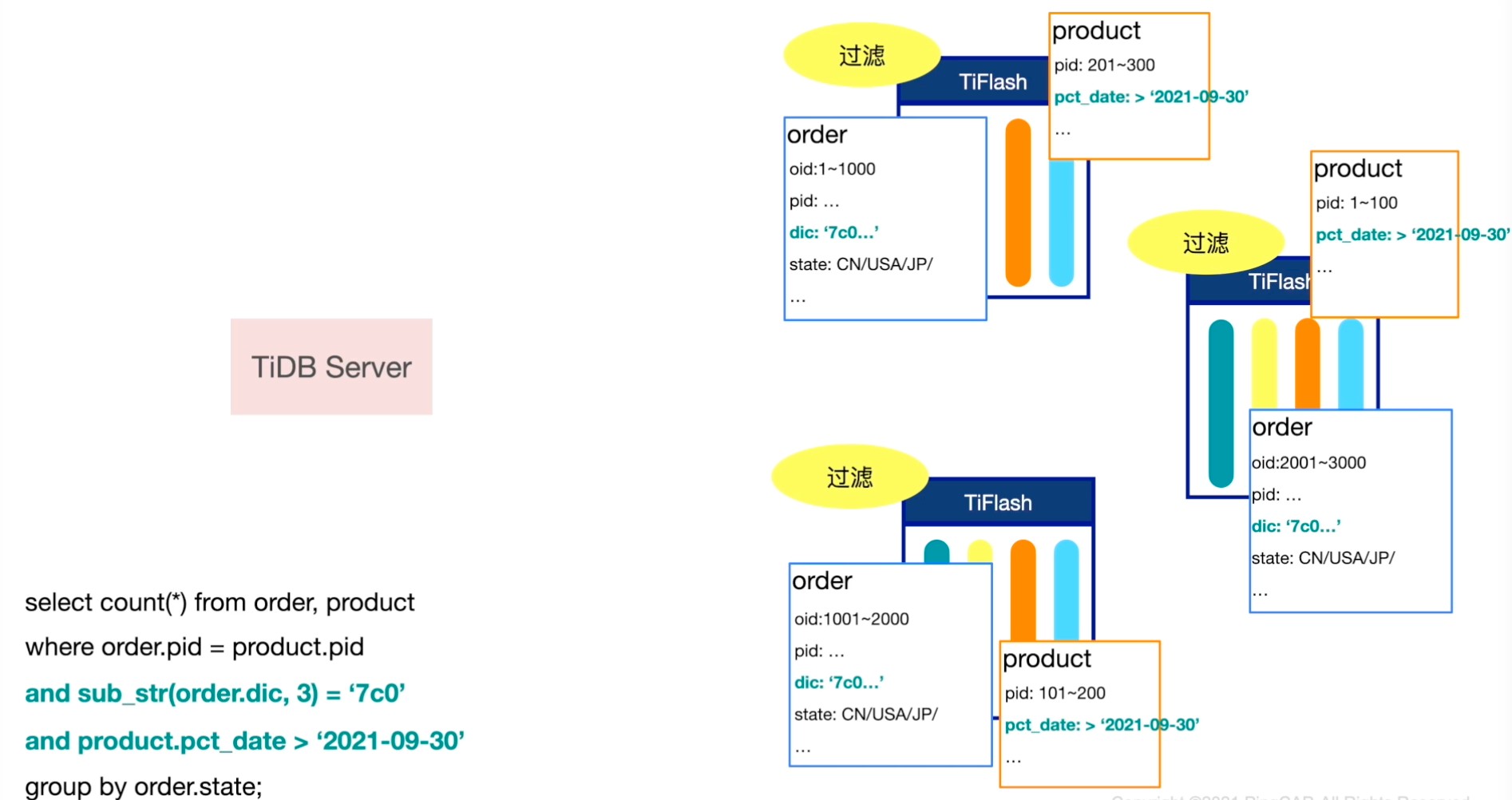
过滤*:sub\_str*，*pct\_date>''*分组*:group\_by*

1. 三个tiflash节点数据分布
   * 过滤:sub\_str and pct\_date>'' ,在tiflash中过滤数据放在内存中
   * 数据交换:pid相等都到⼀个节点(耗费⽹络Io)
     + 对pid进⾏hash,order和produce两个表相等的交换到⼀个tiflash节点
   * Join:并⾏做join计算
   * 聚合:
     + 数据交换:根据state做hash每个节点都放state相关数值,保证聚合是单⼀值做聚合
   * 结果
     + 在TiDB中把计算结果做⼀个排序 结果聚合即可

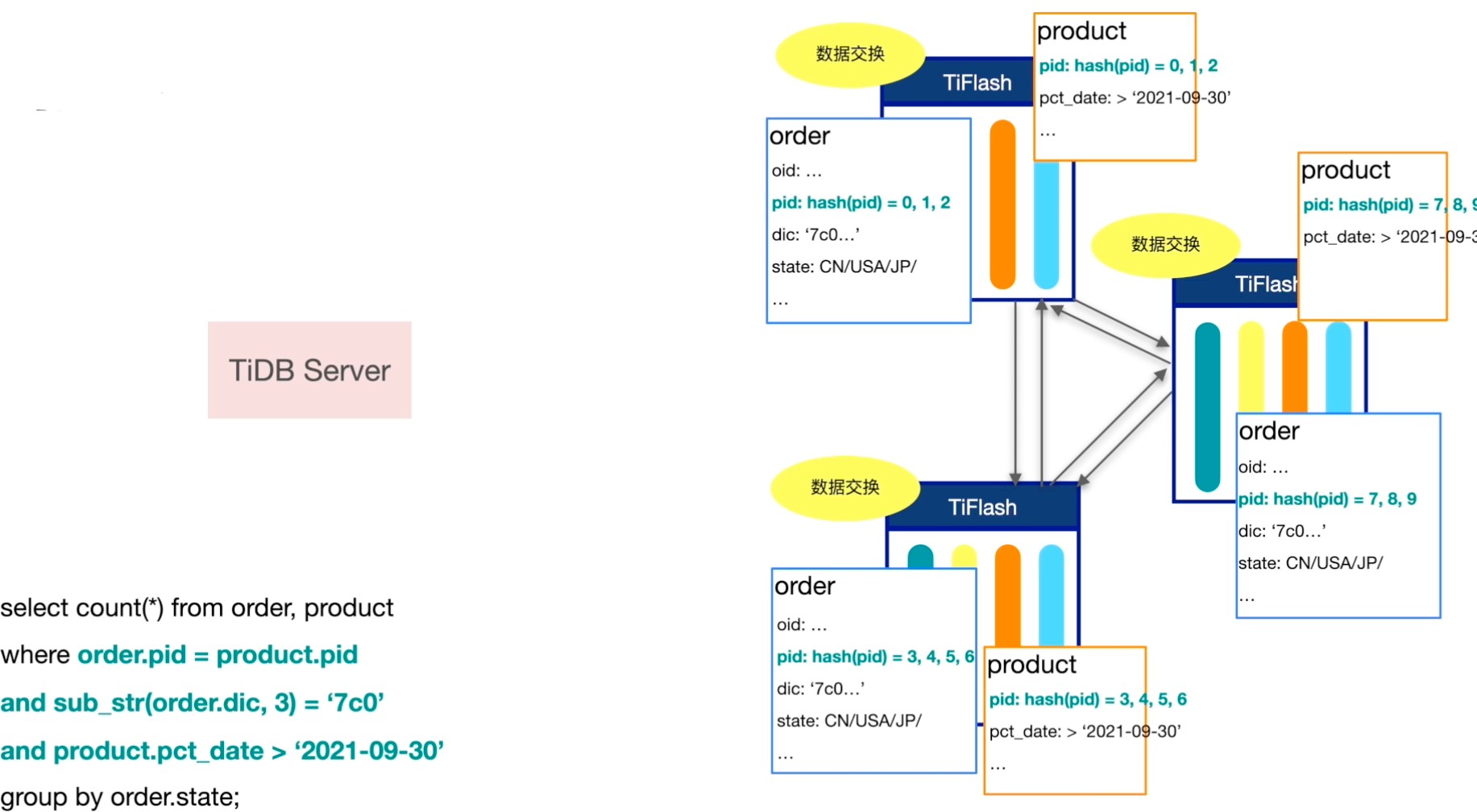
聚合sql请求



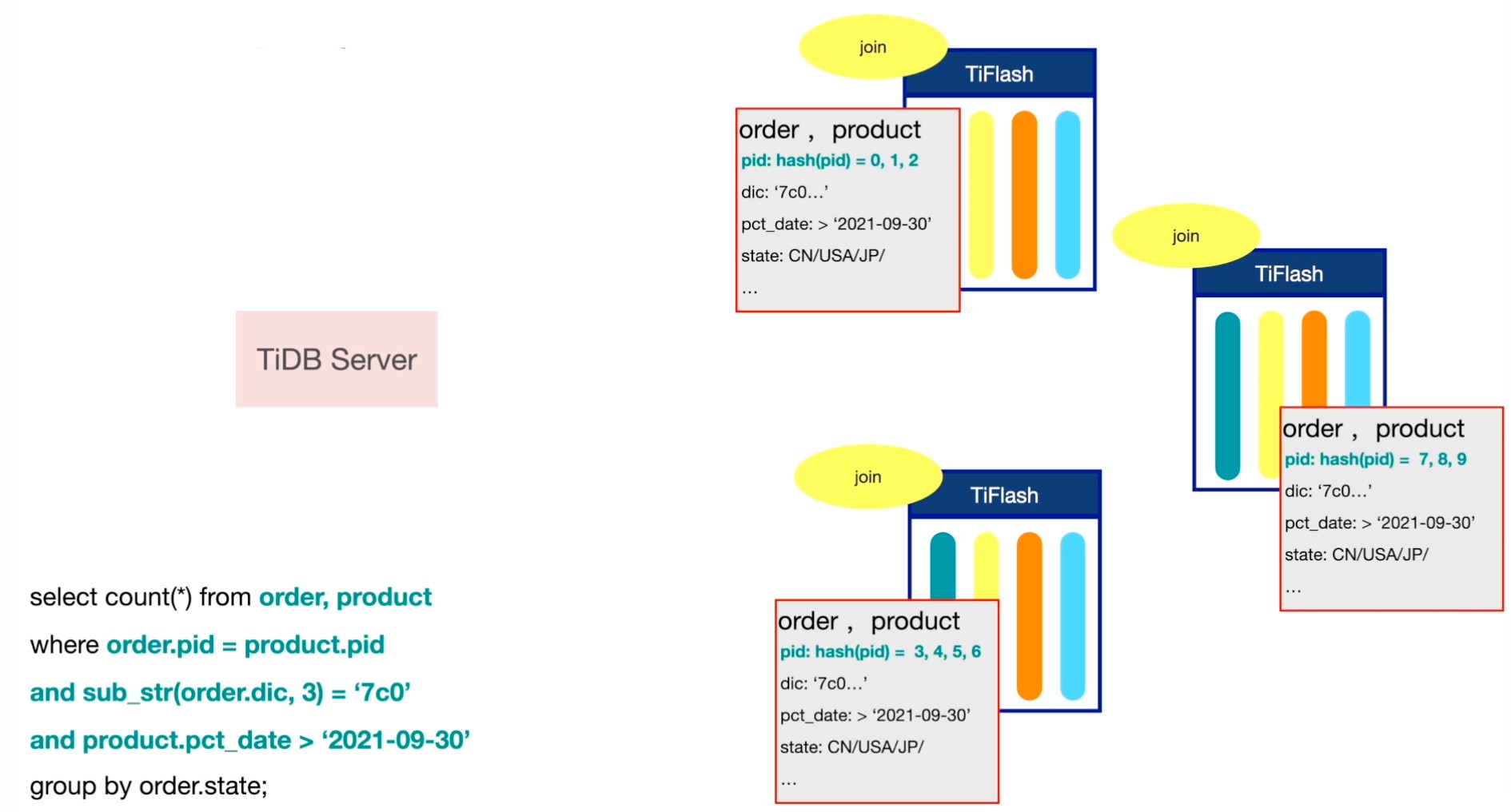
过滤



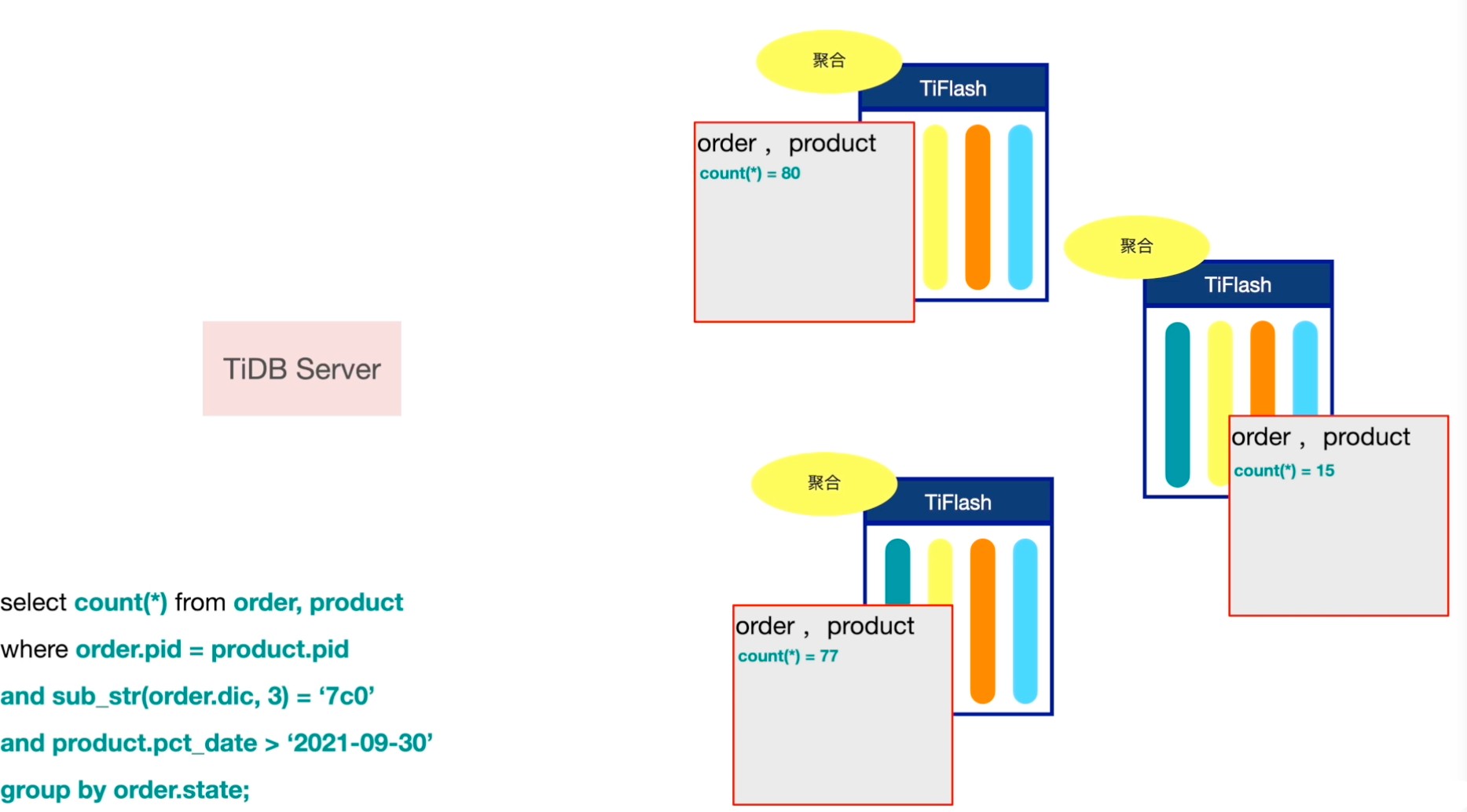
数据交换



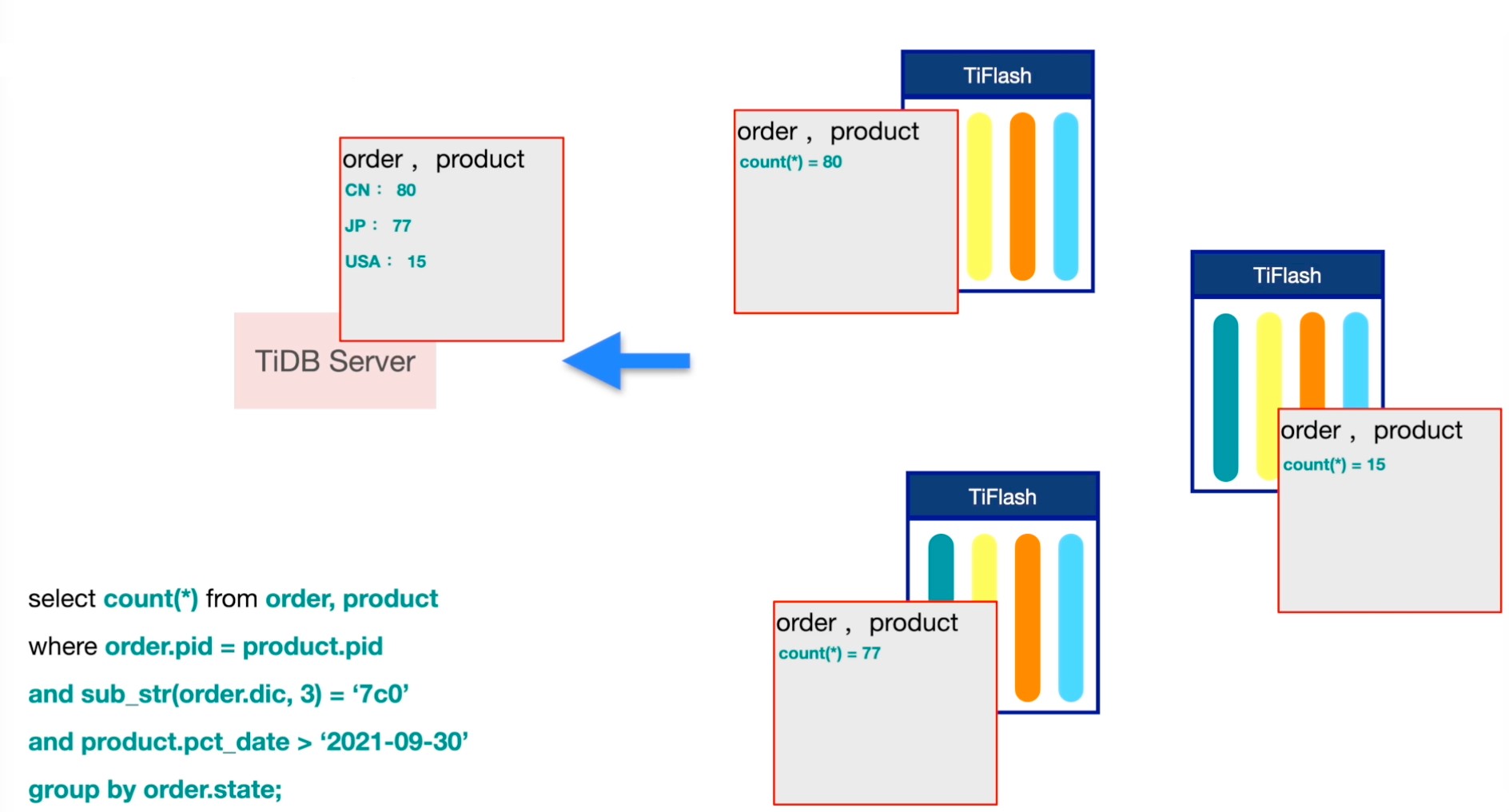
并⾏计算join



经过数据交换后做聚合



给出结果



* + 1. **使⽤**MPP**模式**

控制是否选择 MPP 模式的参数：

* tidb\_allow\_mpp
* tidb\_enforce\_mpp

参数设置：

不想使⽤ MPP 模式：

set @@session.tidb\_allow\_mpp=0;

如果想要通过优化器代价估算来智能选择是否使⽤ MPP（默认情况）： set @@session.tidb\_allow\_mpp=1;

set @@session.tidb\_enforce\_mpp=0;

没有⾛mpp的示例：

set @@session.tidb\_enforce\_mpp=1; create table t(a int);

explain select count(\*) from t; show warnings;

+---------+------+---------------------------------------------------------------------

- +

| Level | Code | Message

|

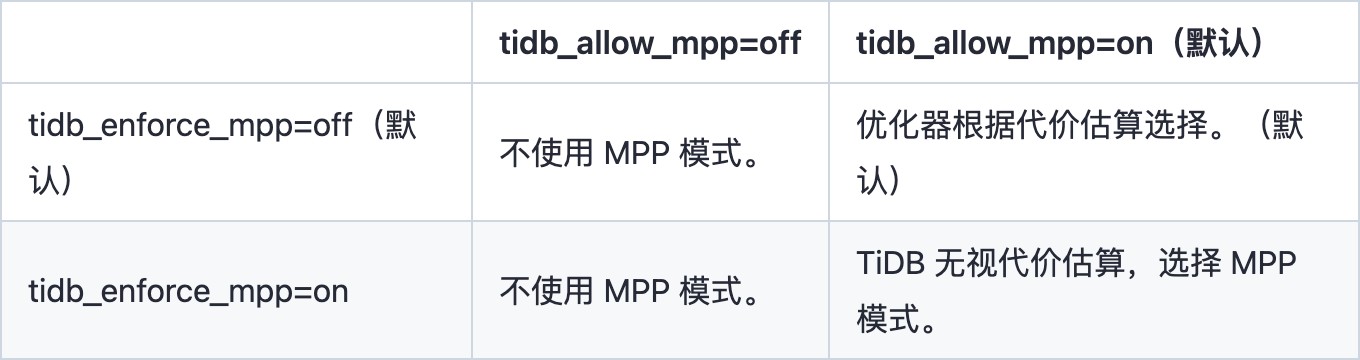
+---------+------+---------------------------------------------------------------------

- +

| Warning | 1105 | MPP mode may be blocked because there aren't tiflash replicas of table `t`. |

+---------+------+---------------------------------------------------------------------

- +



* + 1. MPP **模式算法⽀持**

1、MPP 模式⽬前⽀持的物理算法：

* Broadcast Hash Join
* Shuffled Hash Join
* Shuffled Hash Aggregation
* Union All
* TopN
* Limit

例⼦：

mysql> explain select count(\*) from customer c join nation n on c.c\_nationkey=n.n\_nationkey;

+------------------------------------------+------------+-------------------+----------

-----+ +

| id | estRows | task | access object | operator info |

+------------------------------------------+------------+-------------------+----------

-----+ +

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| | HashAgg\_23 | | | 1.00 | | | root | | |  |
| | funcs:count(Column#16)->Column#15  | └─TableReader\_25 | | | 1.00 | | | root | | | | |
| | data:ExchangeSender\_24  | └─ExchangeSender\_24 | | | 1.00 | | | batchCop[tiflash] | | | | |
| | ExchangeType: PassThrough  | └─HashAgg\_12 | | | 1.00 | | | batchCop[tiflash] | | | | |
| | funcs:count(1)->Column#16  | └─HashJoin\_17 | | | 3000000.00 | | | batchCop[tiflash] | | | | |

| inner join, equal:[eq(tpch.nation.n\_nationkey, tpch.customer.c\_nationkey)] |

| ├─ExchangeReceiver\_21(Build) | 25.00 | batchCop[tiflash] |

| |

| │ └─ExchangeSender\_20 | 25.00 | batchCop[tiflash] |

| ExchangeType: Broadcast |

| │ └─TableFullScan\_18 | 25.00 | batchCop[tiflash] | table:n

| keep order:false |

| └─TableFullScan\_22(Probe) | 3000000.00 | batchCop[tiflash] | table:c

| keep order:false |

+------------------------------------------+------------+-------------------+----------

-----+ +

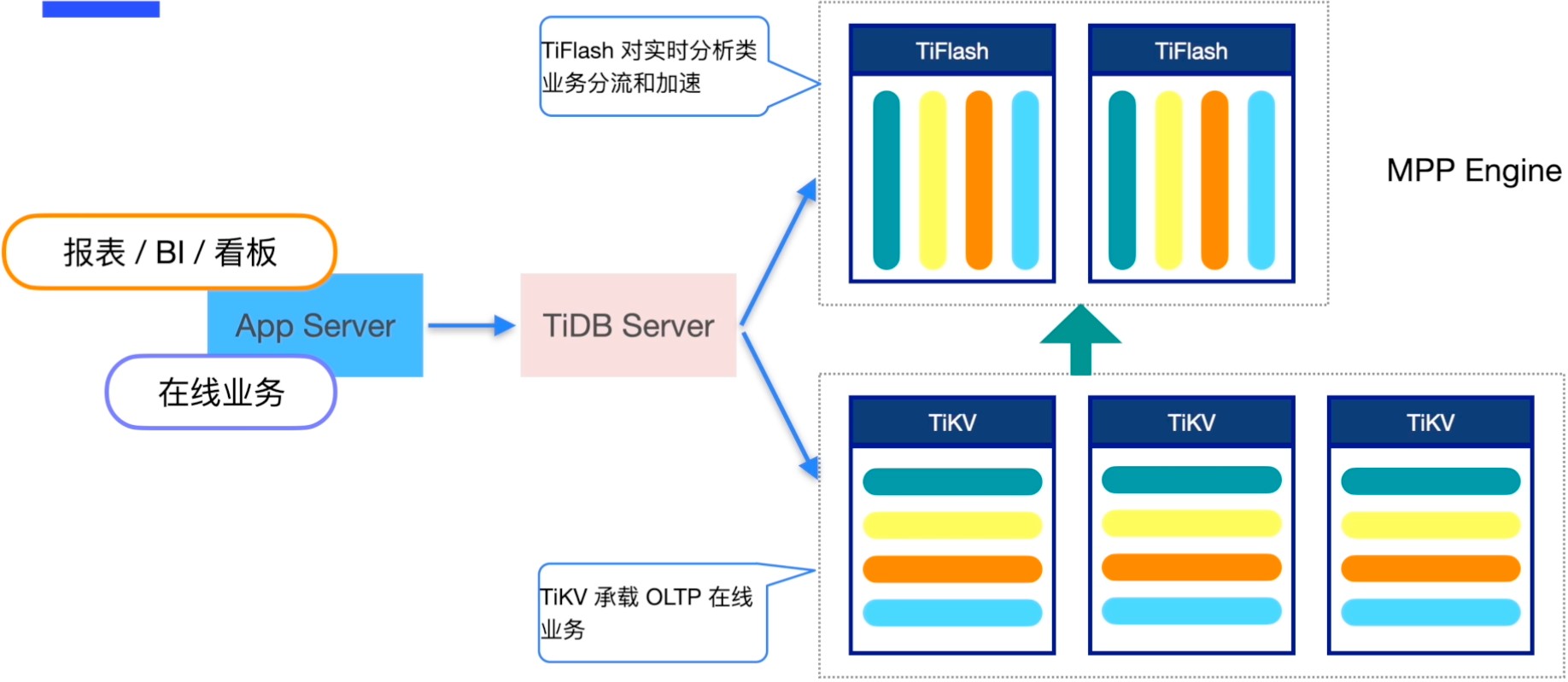
9 rows in set (0.00 sec)

2、MPP不⽀持的功能

* 分区表
* Window Function
* Collation
* 部分内置函数
* 读取 TiKV 数据
* OOM Spill
* Union
* Full Outer Join
  1. HTAP**场****景**

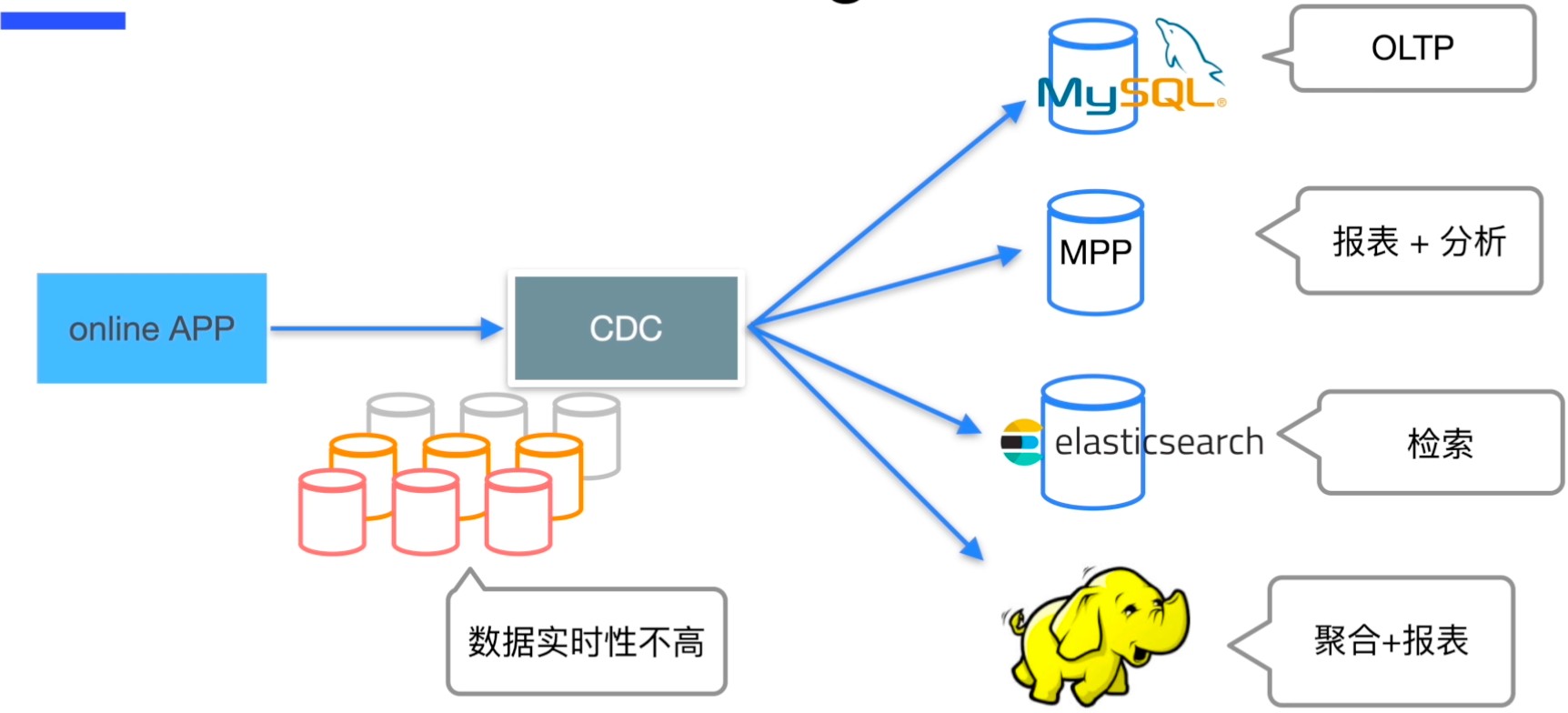
## 混合负载场景

1. 在线⾛tp,tikv
2. 实时分析⾛ap,tiflash



* + 1. **实时流处理场景**

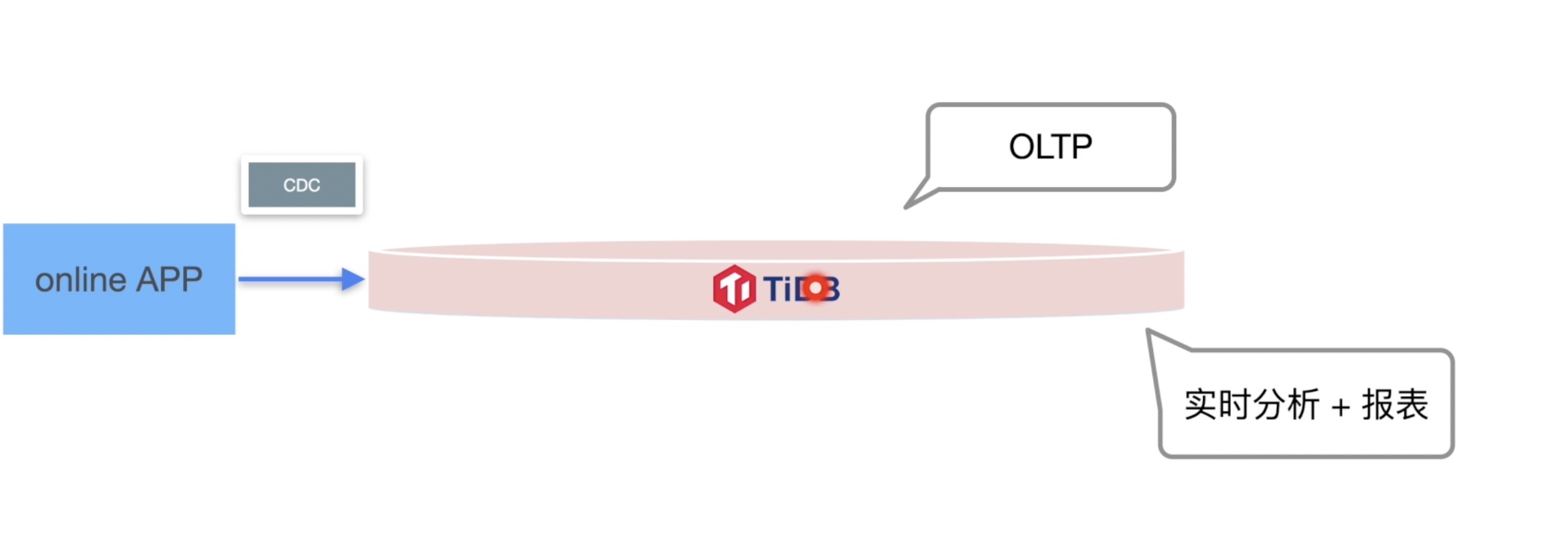
传统⽅案



传统⽅案：

- cdc抽取变化增量数据到下游进⾏计算

TiDB流式计算

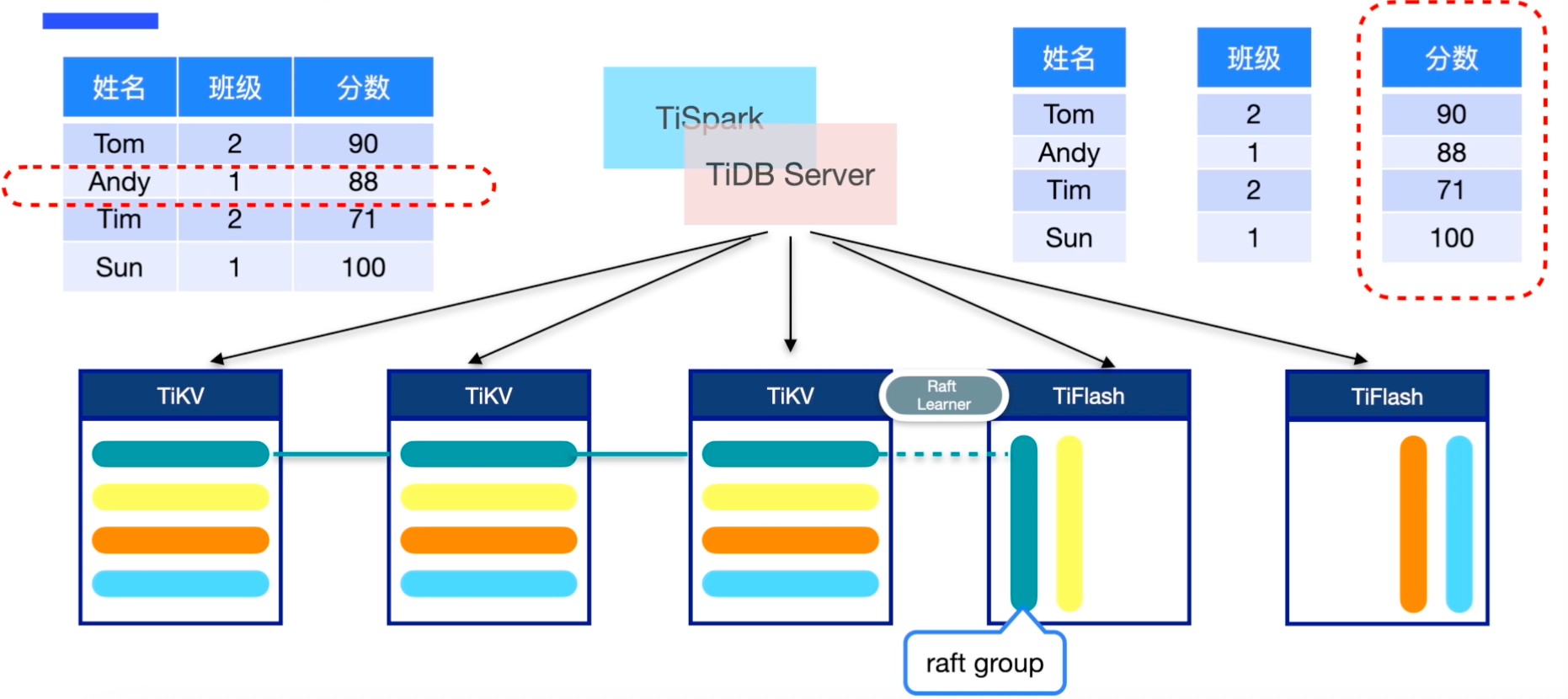


1.2.3 **数据中枢场景**

⽆缝连接数据业务层和数据仓库层

# 2**、**TiFlash

* 1. TiFlash**架构**



1. 加⼊tiflash后,tikv数据列存版本
2. tikv与tiflash的region 副本与分区⼀致(包括分裂等操作)
3. 数据通过 raft leaner同步(仅接受raft⽇志)
   1. **核⼼特****性**

1. 异步复制
   * tiflash宕机不影响tikv
   * 随时拉去新副本 取消副本

2.⼀致性读取

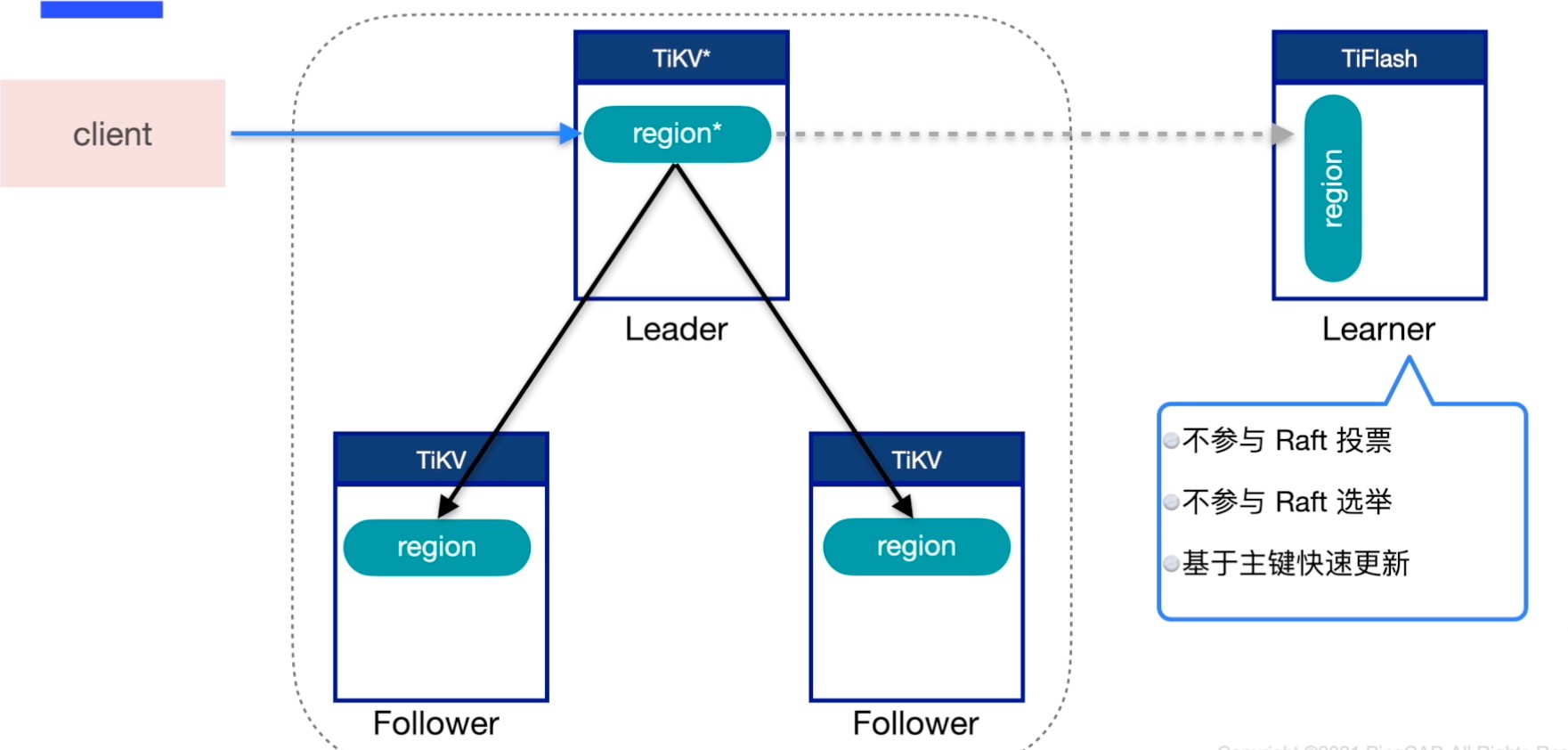
- ⽀持SI隔离级别,存储数据多个版本

3.智能选择

- CBO(贪⼼算法模型):根据执⾏代价不同算⼦执⾏不同计划

1. 计算加速
   * 列存:MPP+统计分析
   * 计算下推:过滤 表连接 聚合等

## 异步复制



* + 1. **⼀致性读取**

T0时刻:

1.TIKV raft log

- idx=101 and idx=22 标记时间戳

2.tiflash,写⼊未到tiflash

* 绿⾊ idx =95
* ⻩⾊ idx = 18

T1:时刻 tiflash

* 绿⾊: idx=108（已经读取了idx=101的tikv数据 可以读取）
* ⻩⾊: idx=20(未到29,raft log没读取到)

- client端发出查询,要key=1和key=999，tiflash准备询问tikv raft log⽇志位置

T2:tiflash准备向tikv确认,client写⼊绿⾊ value-200，⽣产122

T3:

region发出⼀个轻量级的确认(raft log写⼊位置,如果⼀致则数据确定都有)

* 去绿⾊region确认tikv idx=125，⻩⾊tikv 31，对⽐⾃⼰的tiflash
* 等待此刻获取的SI raft log同步

T4:

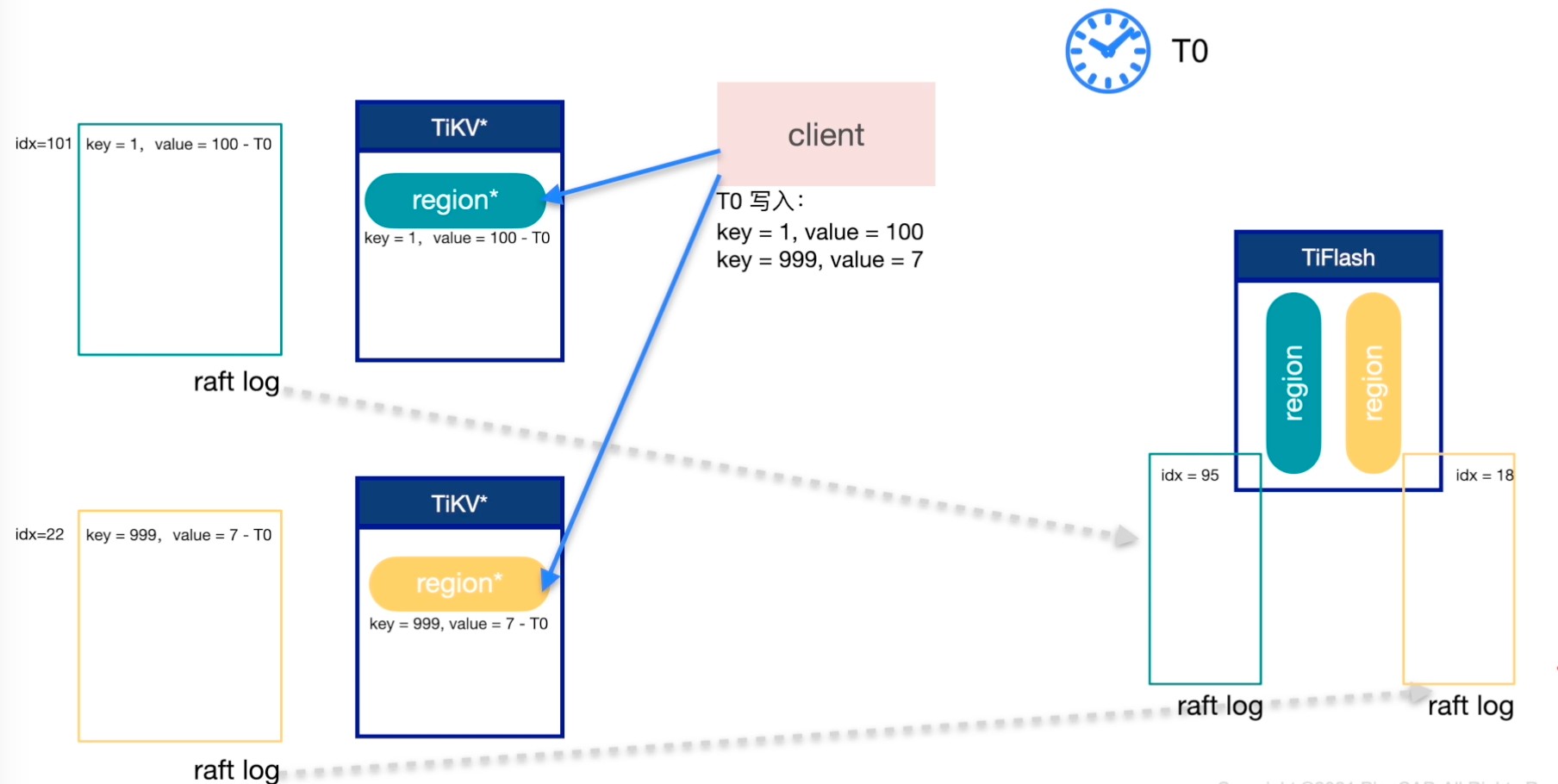
tiflash等待SI记录应⽤了idx=125和idx=31,

T5:raft log全部应⽤完毕

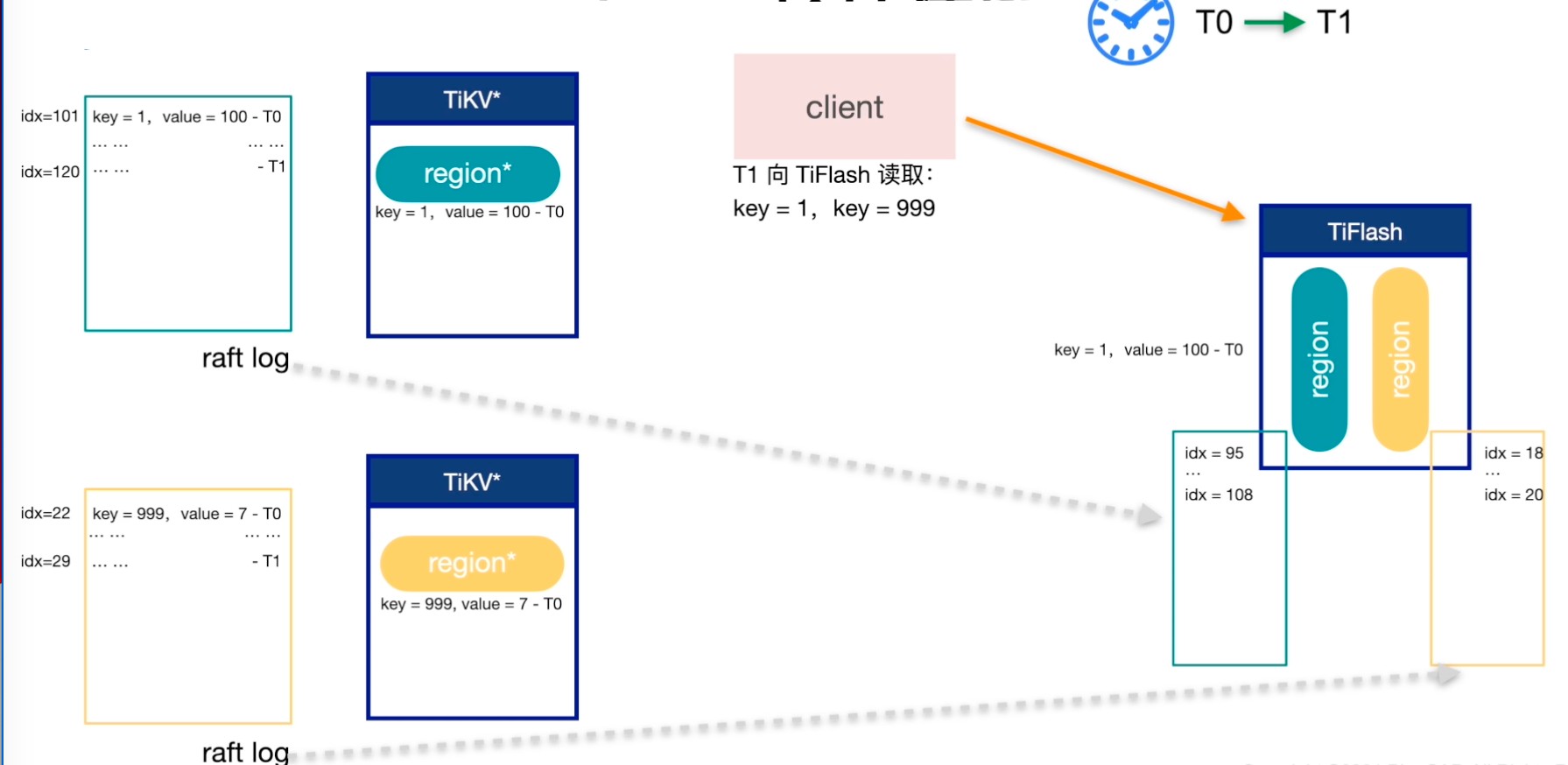
1. tiflash应⽤ idx=125 idx=31
2. client在t1时刻读取,按照此时的SI读取数据,绿⾊的value=200在发出读取请求之后,所以读取的是

key=1,value 100

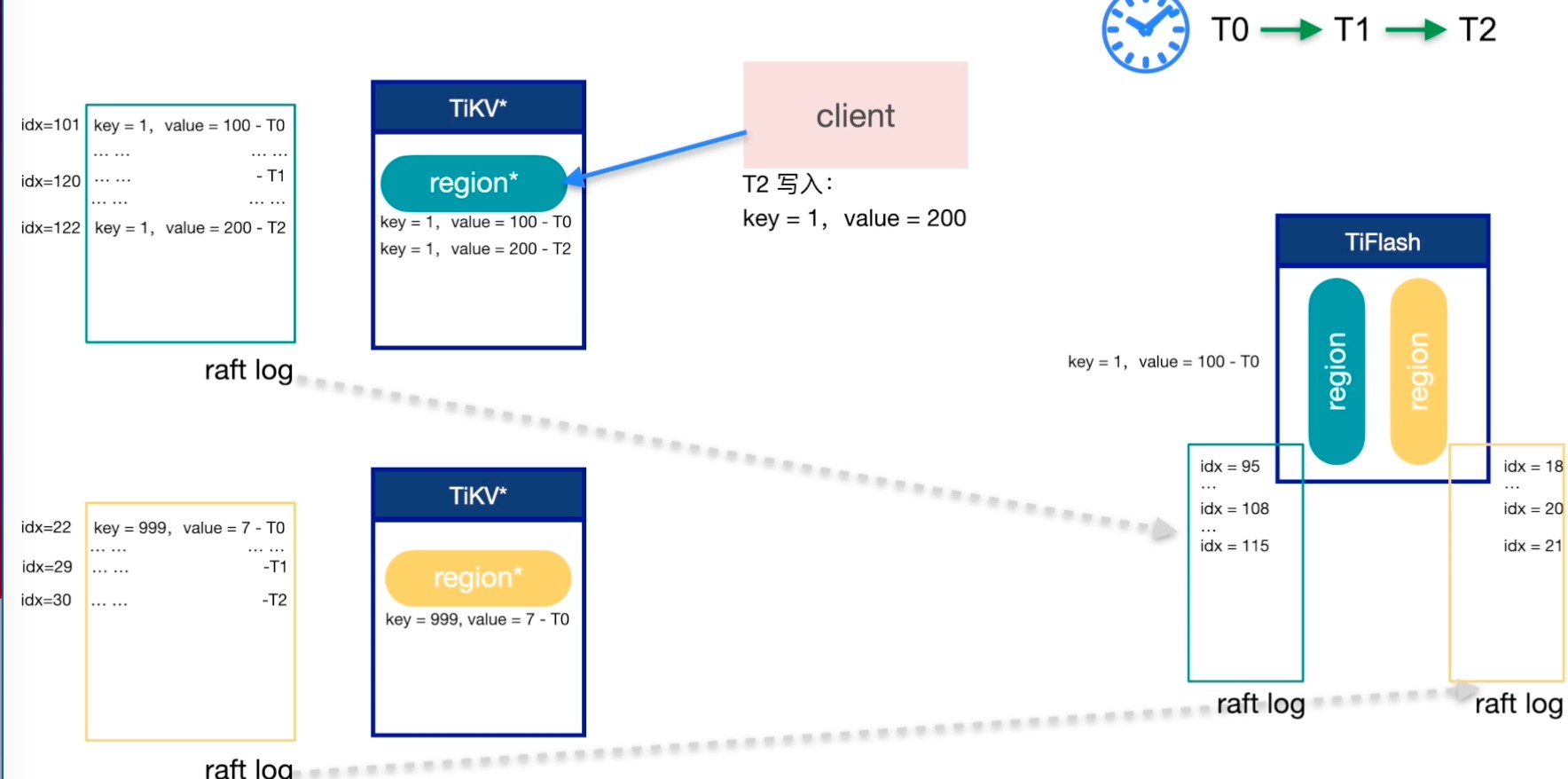
T0时刻



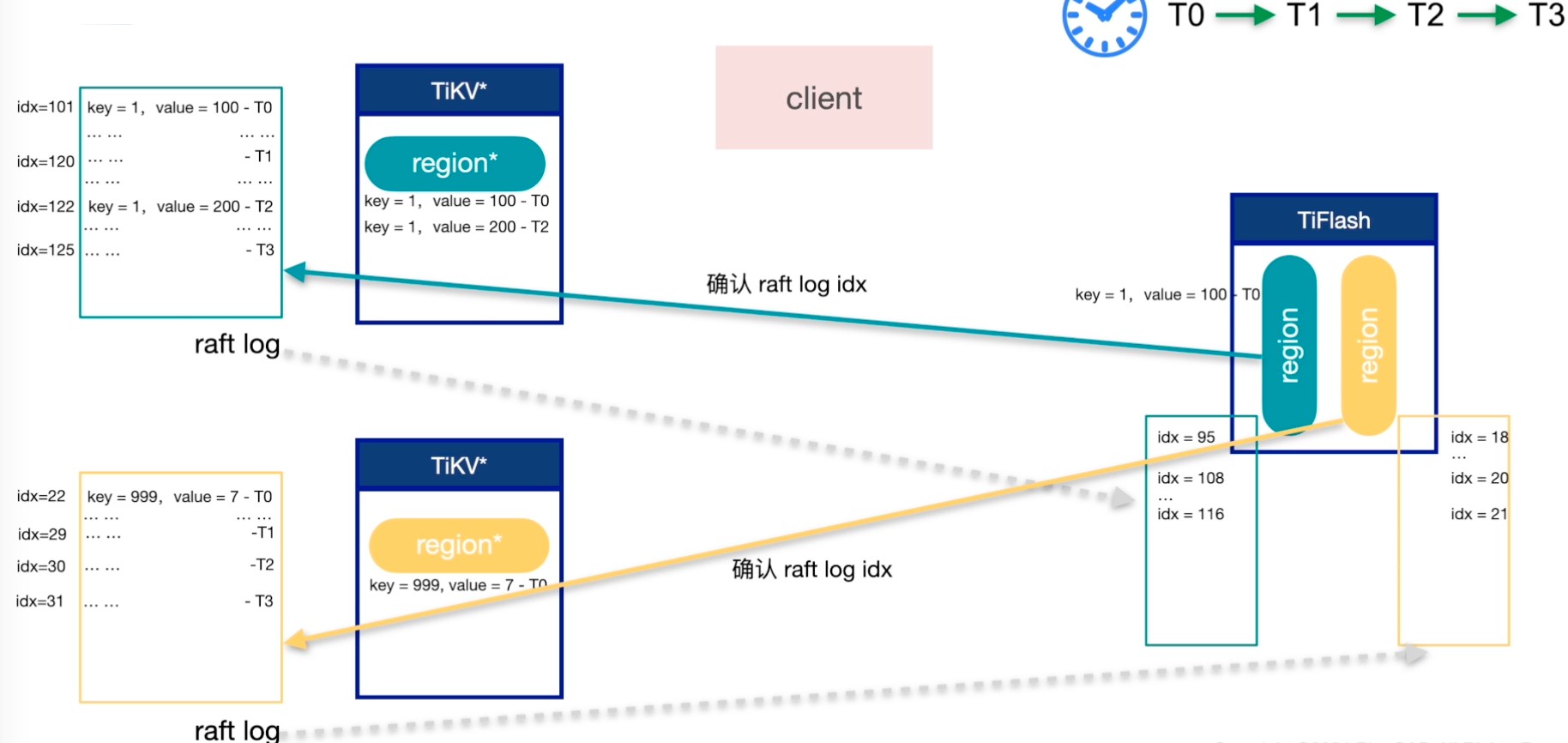
T1时刻



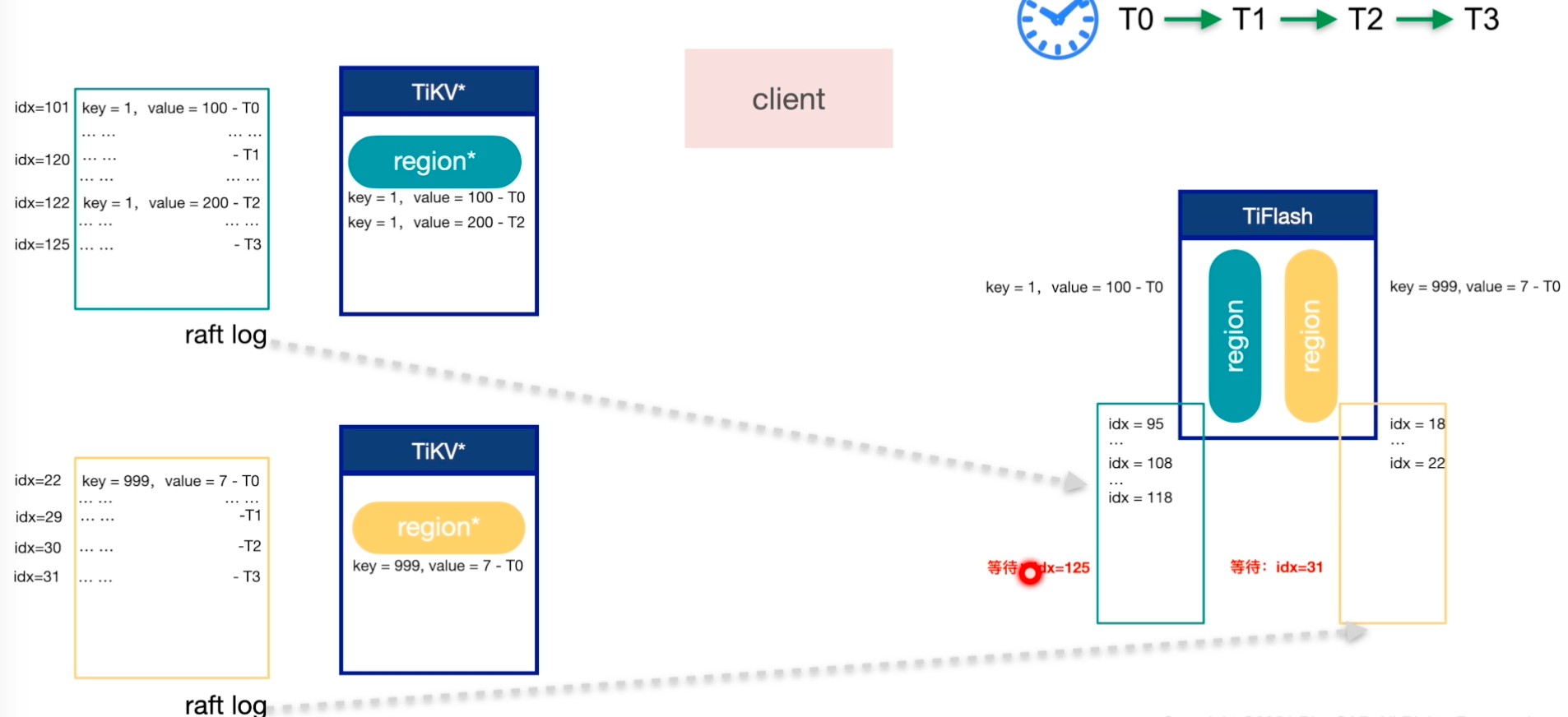
T2时刻 继续写⼊



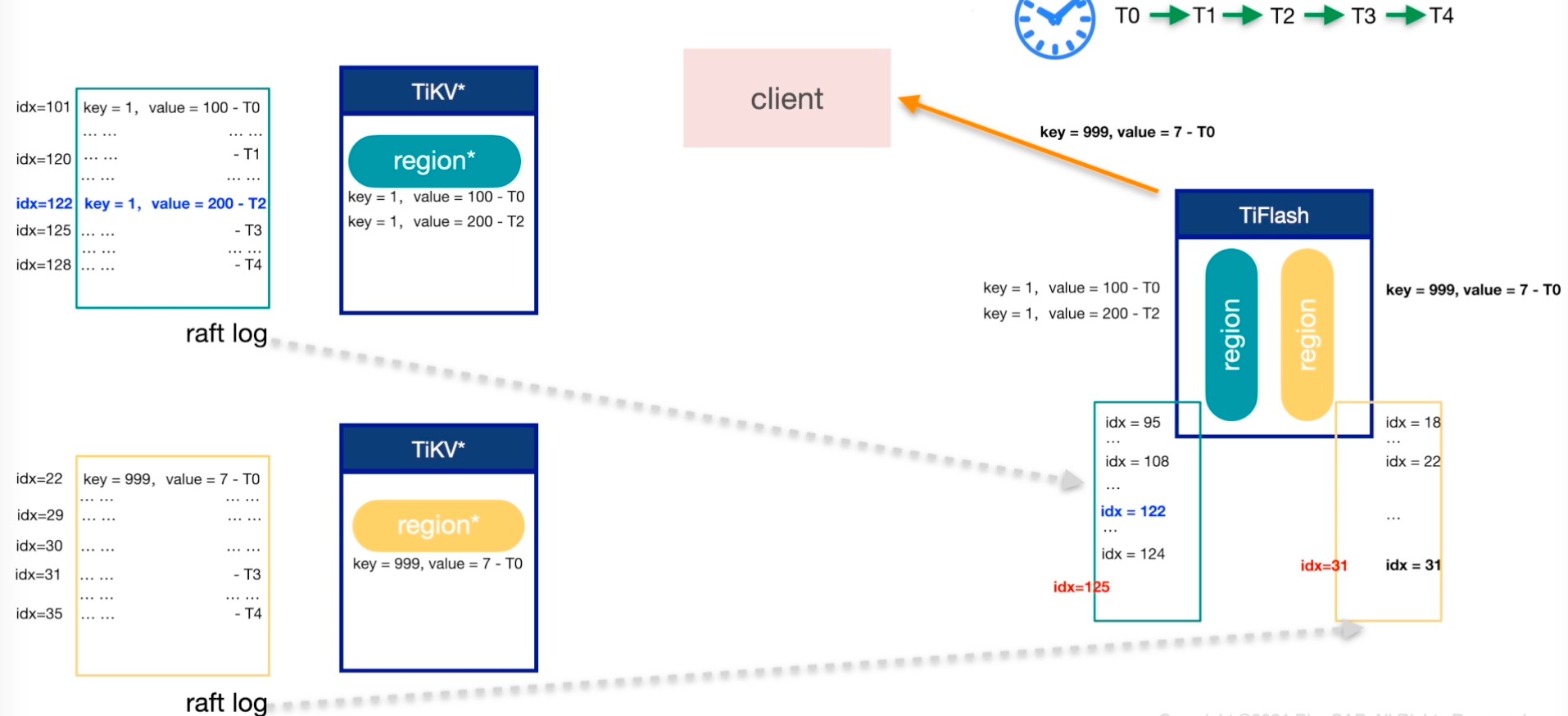
T3时刻



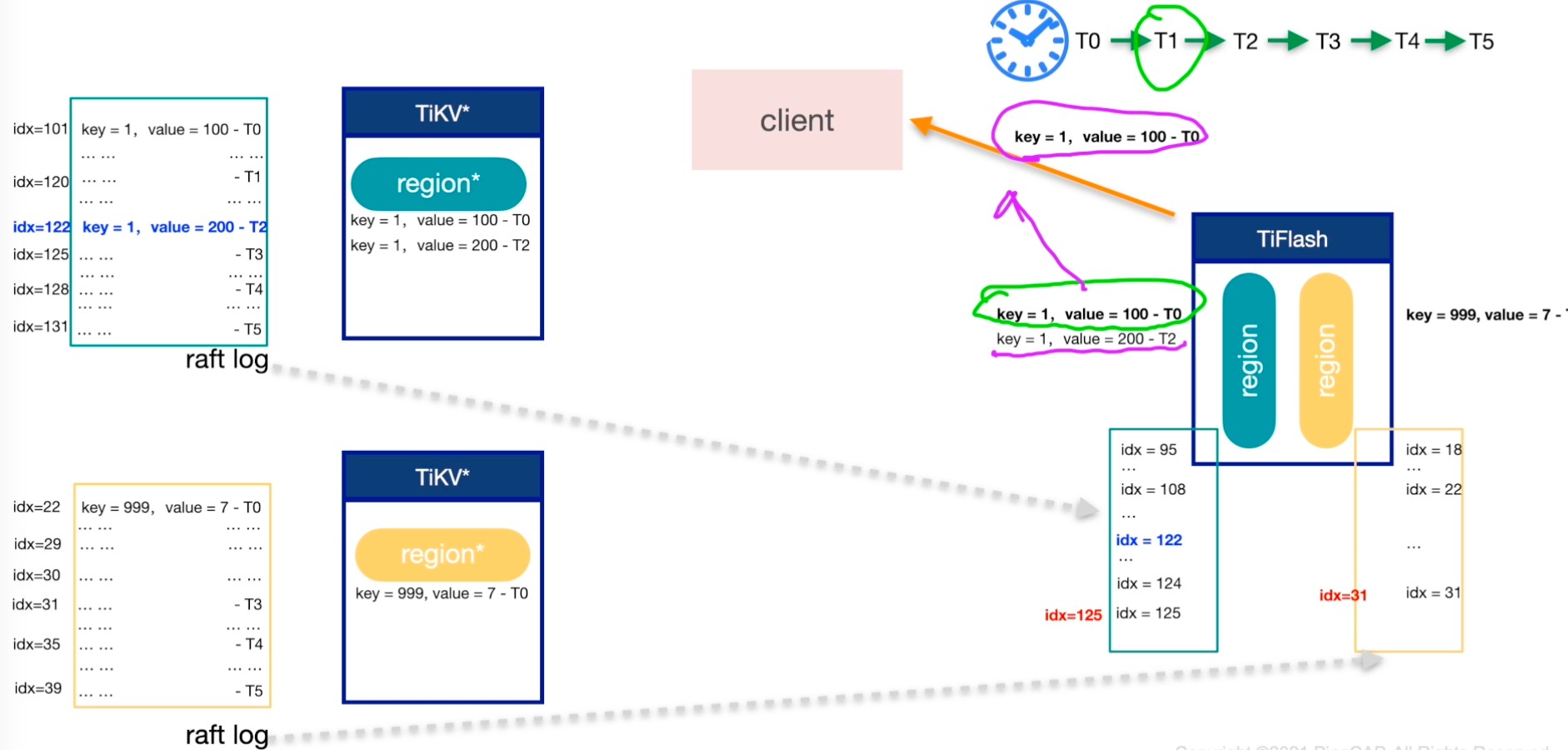
tiflash等待raft log应⽤



T4时刻

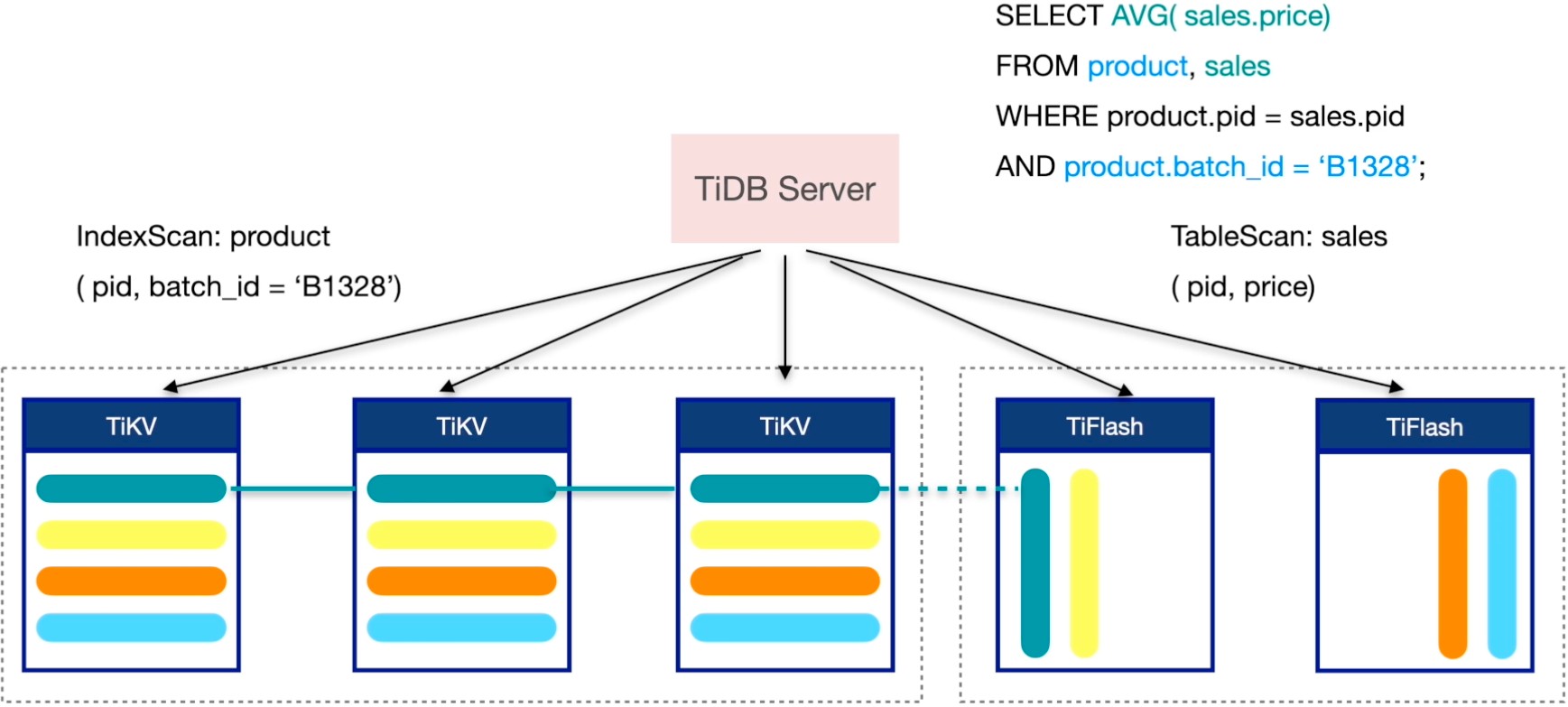


T5时刻：SI全部应⽤完毕



## 智能选择

1. tikv：索引扫描 pid+batch\_id-->tidb
2. tiflash：表扫 sales.price+pid-->tidb
3. tidb server做表连接操作，聚合汇总



* 1. TiFlash**的使****⽤**

## 部署&扩缩容

1. **部署**

# 1、全新部署

参照tidb集群部署⽅式，配置中增加关于tiflash相关的参数

tiflash\_servers:

- host:192.168.1.1

1. **扩容**

扩容部署

1. 确认当前tidb的版本⽀持tiflash，否则需要先升级tidb集群到5.0以上
2. enable-placement-rules参数开启
3. 编辑扩容配置⽂件（scale-out topology）

# vim scale-out.yaml tiflash\_servers:

- host:192.168.1.1

(4) 执⾏扩容命令

tiup cluster scale-out <cluster-name> scale-out.yaml

预期输出 Scaled cluster <cluster-name> out successfully 信息，表示扩容操作成功。

（5）检查集群状态

tiup cluster display <cluster-name>

\*\*注意：\*\*

此处假设当前执⾏命令的⽤户和新增的机器打通了互信，如果不满⾜已打通互信的条件，需要通过 `-p` 来输⼊新机器的密码，或通过 `-i` 指定私钥⽂件。

1. **缩容**

1、根据 TiFlash 剩余节点数调整数据表的副本数

1. 删除副本

alter table <db-name>.<table-name> set tiflash replica 0;

1. 等待相关表的 TiFlash 副本被删除（查不到相关表的同步信息时即为副本被删除）。

select \* from information\_schema.tiflash\_replica where table\_schame='xxx' and table\_name='xxx';

2、执⾏缩容操作

tiup cluster scale-in <cluster-name> --node 10.0.1.4:9000

## 使⽤

1. **按表构建** TiFlash **副本**

# 增加tiflash副本

ALTER TABLE test.dba\_test SET TIFLASH REPLICA 2;

# 删除tiflash副本

ALTER TABLE test.dba\_test SET TIFLASH REPLICA 0;

#查看表同步进度

SELECT \* FROM information\_schema.tiflash\_replica WHERE TABLE\_SCHEMA = '<db\_name>' and TABLE\_NAME = '<table\_name>';

1. **按库构建** TiFlash **副本**

#为test库新建2个tiflash副本

ALTER DATABASE test SET TIFLASH REPLICA 2;

#删除库test的tiflash副本

ALTER DATABASE test SET TIFLASH REPLICA 0;

#查看库同步进度

SELECT \* FROM information\_schema.tiflash\_replica WHERE TABLE\_SCHEMA = '<db\_name>'; #查看数据库中尚未设置 TiFlash Replica 的表名：

SELECT TABLE\_NAME FROM information\_schema.tables where TABLE\_SCHEMA = "<db\_name>" and TABLE\_NAME not in (SELECT TABLE\_NAME FROM information\_schema.tiflash\_replica where

TABLE\_SCHEMA = "<db\_name>");

1. ⼿动执⾏tiflash
   * 实例隔离,tidb实例级别配置

[isolation-read]

engines = ["tikv", "tidb", "tiflash"]

* + 会话隔离

set @@session.tidb\_isolation\_read\_engines = "tiflash";

- INIT

select /\*+ read\_from\_storage(tiflash[table\_name]) \*/ ... from table\_name;

1. Engine **隔离**

1、TiDB 实例级别，即 INSTANCE 级别

[isolation-read]

engines = ["tikv", "tidb", "tiflash"]

2、会话级别，即 SESSION 级别

set @@session.tidb\_isolation\_read\_engines = "tikv,tiflash";

或者

set SESSION tidb\_isolation\_read\_engines = "tiflash";

1. **⼿⼯** Hint

select /\*+ read\_from\_storage(tiflash[table\_name]) \*/ ... from table\_name;

-- 如果在查询语句中对表设置了别名，在 Hint 语句中必须使⽤别名才能使 Hint ⽣效。⽐如：

select /\*+ read\_from\_storage(tiflash[alias\_a,alias\_b]) \*/ ... from table\_name\_1 as alias\_a, table\_name\_2 as alias\_b where alias\_a.column\_1 = alias\_b.column\_2;

* + 1. TiFlash**调优**

1、资源规划：不要与TiKV共享磁盘

2、TiDB 相关参数调优：

1. 对于 OLAP专属的 TiDB 节点，建议调⼤读取并发数 tidb\_distsql\_scan\_concurrency 到 80

set @@tidb\_distsql\_scan\_concurrency = 80;

1. 开启 Super batch 功能

set @@tidb\_allow\_batch\_cop = 1;

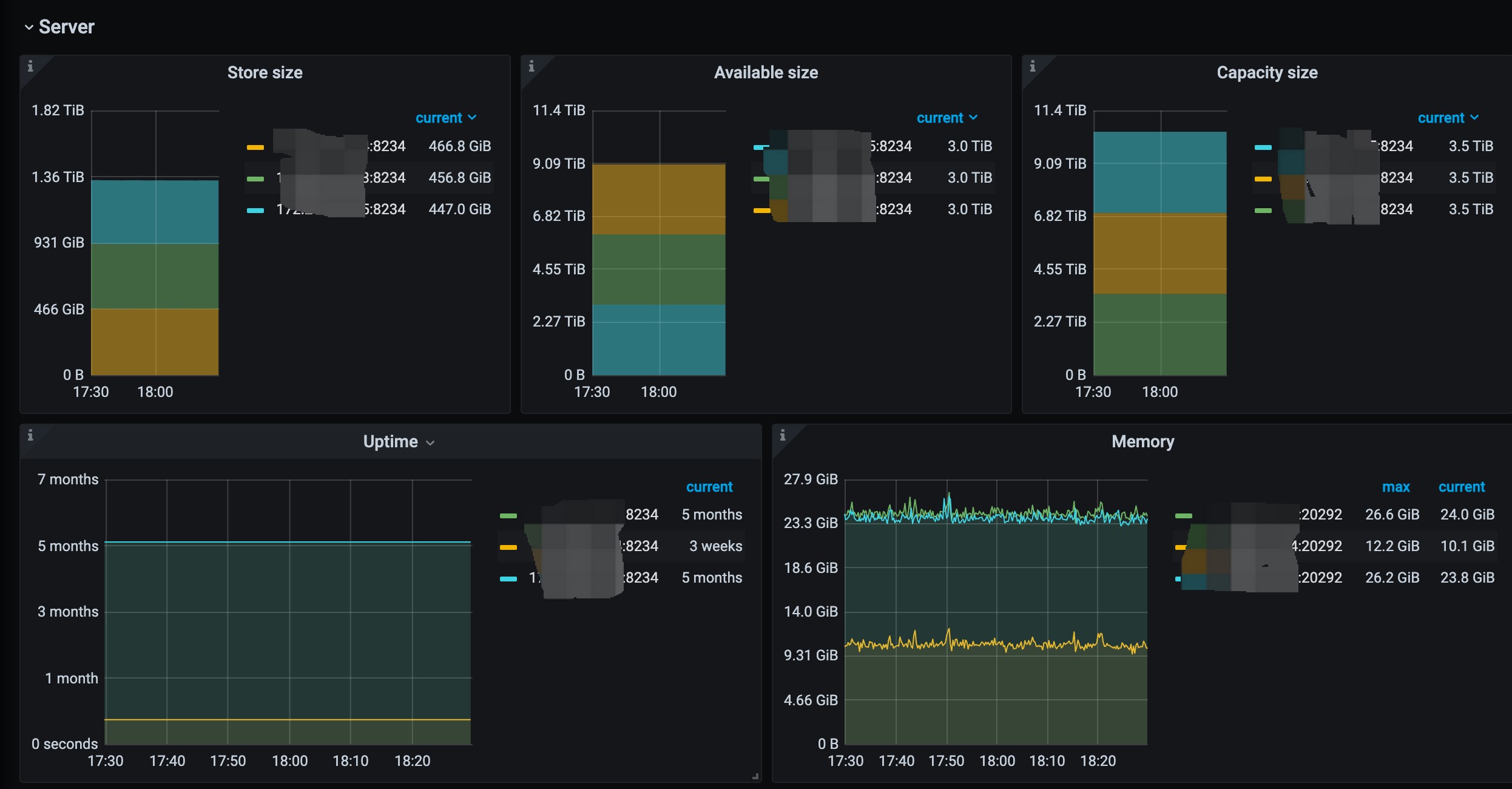
3、开启聚合推过 Join / Union 等 TiDB 算⼦的优化

set @@tidb\_opt\_agg\_push\_down = 1;

4、开启 Distinct 推过 Join / Union 等 TiDB 算⼦的优化

set @@tidb\_opt\_distinct\_agg\_push\_down = 1;

* + 1. TiFlash **集群监控**
* TiFlash-Summary
* TiFlash-Proxy-Summary -> Raft 层信息
* TiFlash-Proxy-Details -> Raft 层信息



* + 1. TiFlash**计算下推与不⽀持场景**

官⽹：https://docs.pingcap.com/zh/tidb/stable/tiflash-supported-pushdown-calculations

1、⽀持：

- 数学函数：+, -, /, \* , %, >=, <=, =, !=, <, >等

* 逻辑函数：and, or, not, case when, if, ifnull, isnull, in, like, coalesce, is
* 字符串函数：substr, char\_length, replace等

2、不⽀持：

* 字符串sum()不⽀持, select sum(cast(string\_col as double)) from t
  + 1. TiFlash**常⻅问题的处理**

**# 1、部分查询返回 Region Unavailable 的错误**

* TiFlash 负载压⼒过⼤, 数据同步落后
* 增加 TiFlash 节点

**# 2、TiFlash 分析慢**

* 检查语句中是否含有 MPP 模式不⽀持的算⼦或函数等

mysql> set @@session.tidb\_enforce\_mpp=1;

mysql> explain select count(*\*) from t where subtime(a, '12:00:00') > '2022-01-01' group by a;*

*mysql> show warnings;*

*Warning | 1105 | Scalar function 'subtime'(signature: SubDatetimeAndString, return type: datetime) is not supported to push down to tiflash now.*

**# 3、TiFlash 数据同步卡住**

* 检查磁盘空间
* 检查是否有 down peer （down peer 没有清理⼲净可能会导致同步卡住）

**# 4、数据同步慢**

* 调整调度参数取值
* 调整 TiFlash 侧负载