HTAP for MySQL在腾讯云数据库的演进

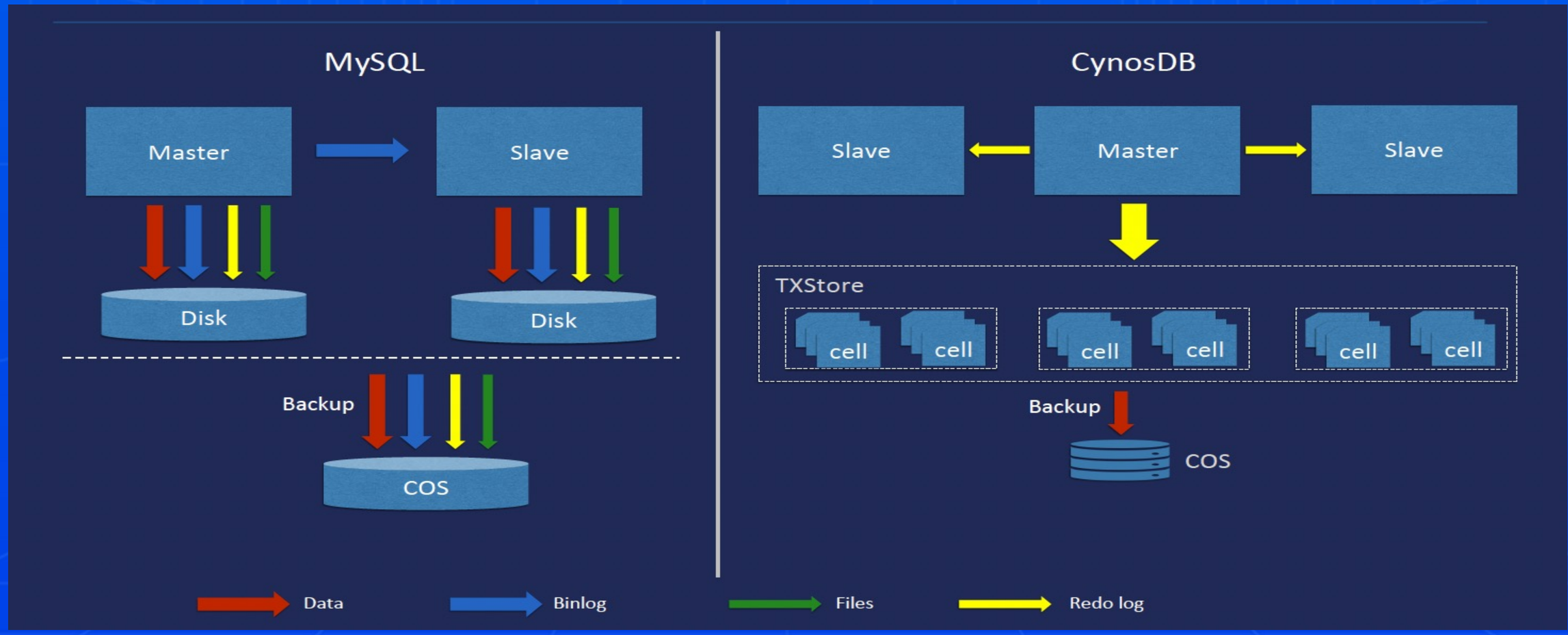
腾讯云数据库 陈开旺

日程

- **背景**
- HTAP在腾讯云数据库的演进
  - 并行查询
  - 列存索引
- 未来展望

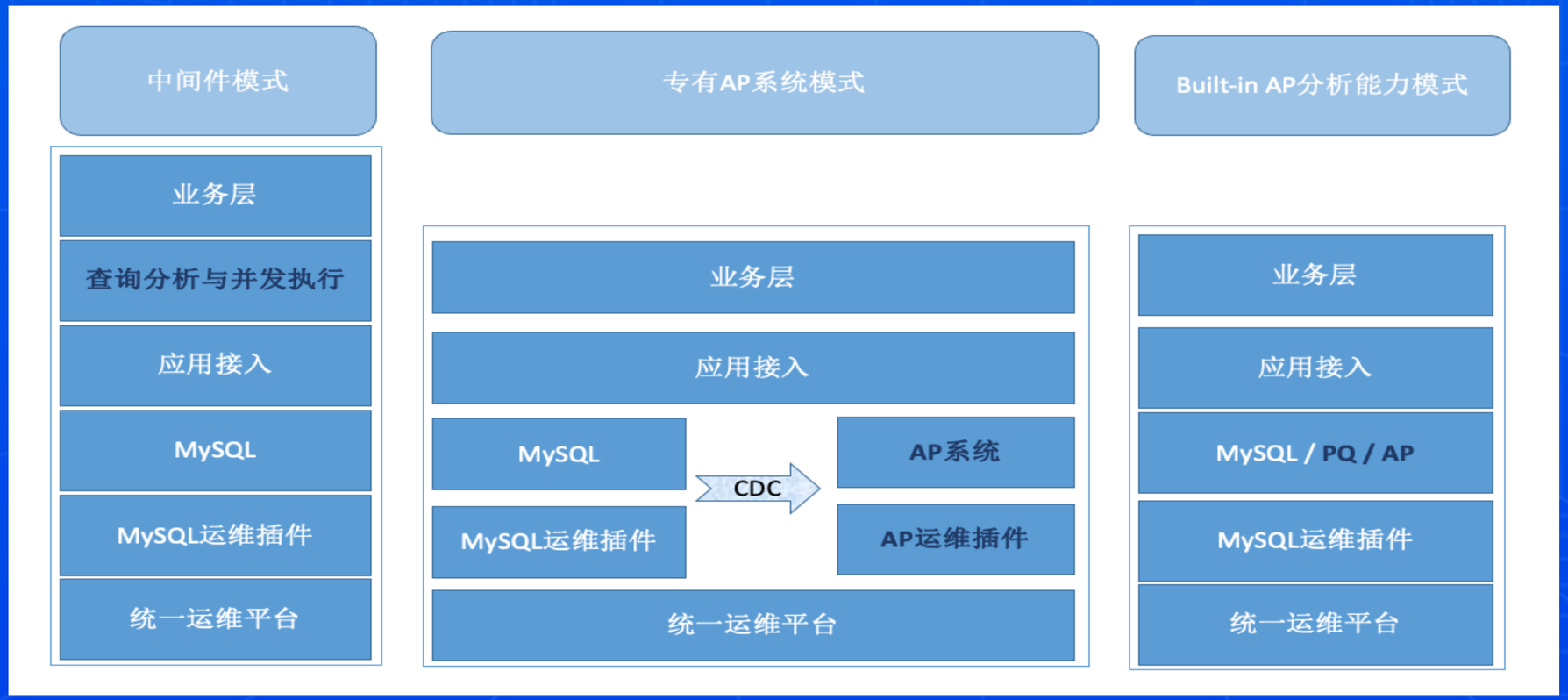
背景 – 产品介绍

云上 MySQL 形态	AWS	腾讯云	说明
用户自建	on EC2	on CVM	自选发行版，自行维护
托管数据库 (RDS)	Amazon RDS	TencentDB	本地存储，基于 binlog 复制
云原生数据库	Amazon Auroa	TDSQL-C (CynosDB)	共享存储，基于 redolog 复制
分布式数据库	-	TDSQL	Shared-Nothing ， 分库分表



背景 – 分析型能力

- 业务挑战：查询数据量大，单核处理瓶颈
- 解决方案：多核处理（并行查询），指令效率（向量化执行，JIT编译执行），信息密度（列式存储）

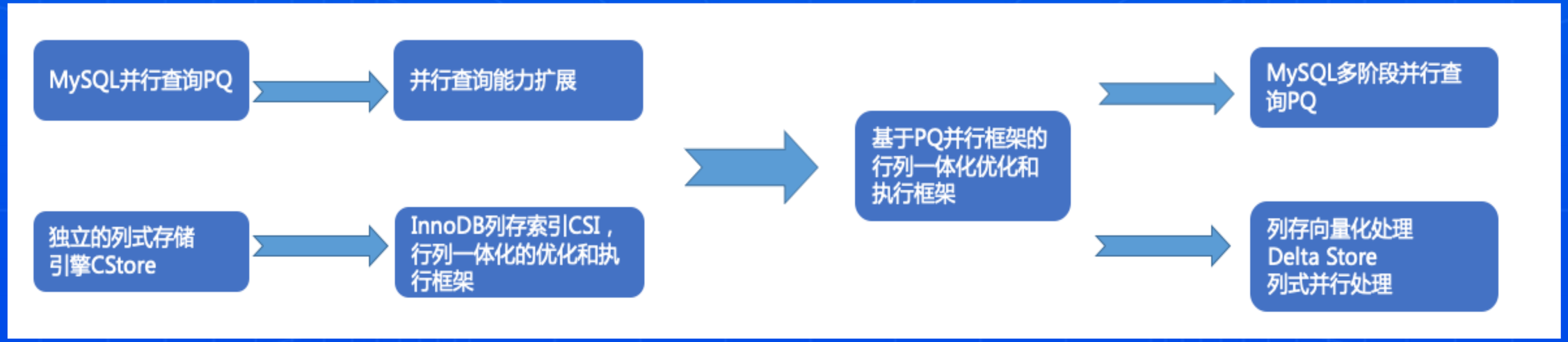


Abadi, D. J., Madden, S. R., & Hachem, N. (2008, June). Column-stores vs. row-stores: how different are they really?

日程

- 背景
- **HTAP在腾讯云数据库的演进**
  - 并行查询
  - 列存索引
- 未来展望

HTAP for MySQL在腾讯云数据库的演进



日程

- 背景
- HTAP在腾讯云数据库的演进
  - **并行查询**
  - 列存索引
- 未来展望

并行查询示例

```
mysql> select l_returnflag, l_linestatus, sum(l_quantity) as sum_qty
    -> from lineitem
    -> where l_shipdate <= '1998-09-02'
    -> group by l_returnflag, l_linestatus
    -> order by l_returnflag, l_linestatus;
+--------------+--------------+--------------+
| l_returnflag | l_linestatus | sum_qty      |
+--------------+--------------+--------------+
| A            | F            | 377518399.00 |
| N            | F            |   9851614.00 |
| N            | O            | 743124873.00 |
| R            | F            | 377732830.00 |
+--------------+--------------+--------------+
4 rows in set (1 min 4.77 sec)

mysql> set global txsql_max_parallel_worker_threads = 32;
Query OK, 0 rows affected (0.00 sec)

mysql> set txsql_parallel_degree = 4;
Query OK, 0 rows affected (0.00 sec)

mysql> select l_returnflag, l_linestatus, sum(l_quantity) as sum_qty
    -> from lineitem
    -> where l_shipdate <= '1998-09-02'
    -> group by l_returnflag, l_linestatus
    -> order by l_returnflag, l_linestatus;
+--------------+--------------+--------------+
| l_returnflag | l_linestatus | sum_qty      |
+--------------+--------------+--------------+
| A            | F            | 377518399.00 |
| N            | F            |   9851614.00 |
| N            | O            | 743124873.00 |
| R            | F            | 377732830.00 |
+--------------+--------------+--------------+
4 rows in set (16.07 sec)
```

可用并行线程总数

查询申请线程数量(缺省值)

原生执行计划

```
-> Sort: lineitem.L_RETURNFLAG, lineitem.L_LINESTATUS
  -> Table scan on <temporary>
    -> Aggregate using temporary table
        -> Filter: (lineitem.L_SHIPDATE <= DATE'1998-09-02')  (cost=6090688.90 rows=29680004)
            -> Table scan on lineitem  (cost=6090688.90 rows=59360009)
```

并行执行计划

```
-> Sort: lineitem.L_RETURNFLAG, lineitem.L_LINESTATUS
  -> Table scan on <temporary>
    -> Final Aggregate using temporary table
      -> PX Receiver (slice: 0; workers: 1)
        -> PX Sender (slice: 1; workers: 4)
            -> Table scan on <temporary>
              -> Aggregate using temporary table
                  -> Filter: (lineitem.L_SHIPDATE <= DATE'1998-09-02')  (cost=6090688.90 rows=29680004)
                    -> Parallel table scan on lineitem  (cost=6090688.90 rows=59360009)
```

计划分拆

数据交换

用户线程执行

并行线程执行

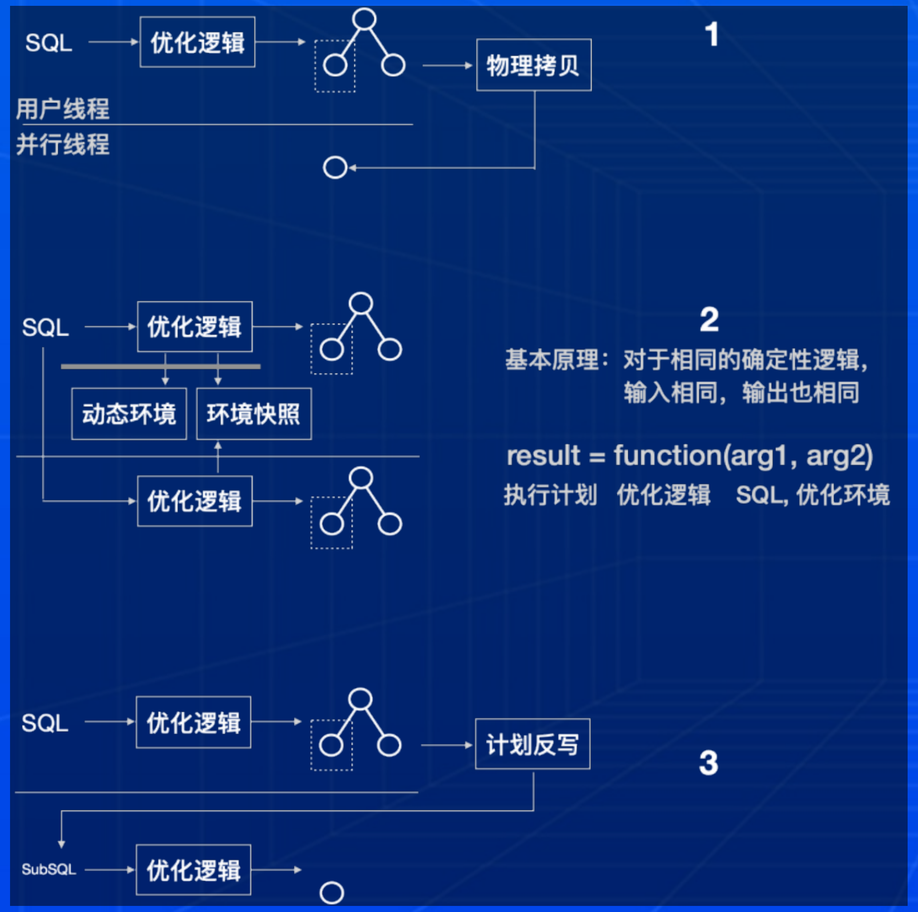
数据划分，并行读取

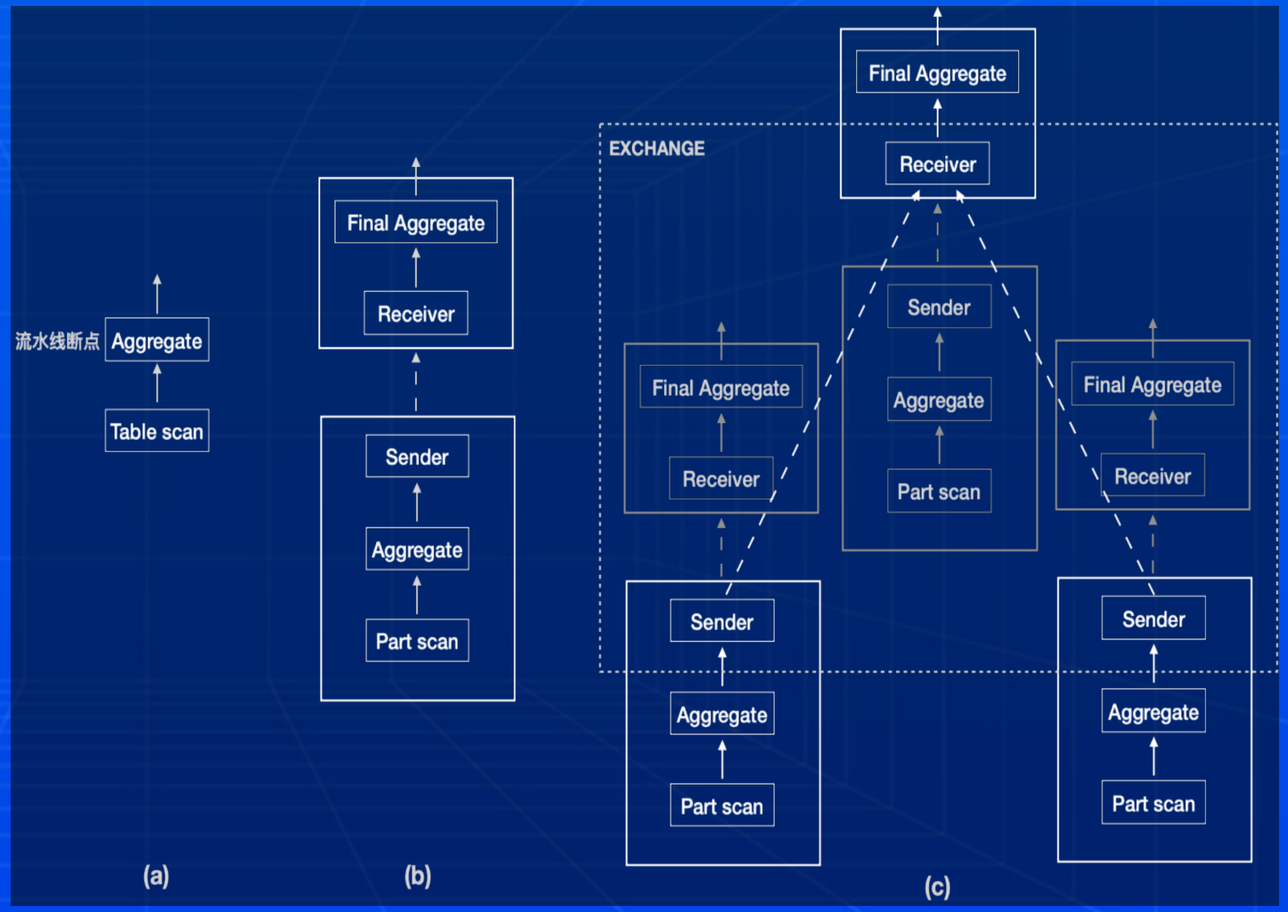
并行线程状态

```
mysql> show parallel processlist;
+-------+-------------+-----------+---------+---------+------+-------------------------------------+-------------------------------------+
| Id    | User        | Host      | db      | Command | Time | State                               | Info                                |
+-------+-------------+-----------+---------+---------+------+-------------------------------------+-------------------------------------+
| 45735 | tencentroot | localhost | tpch10g | Query   |    3 | Waiting workers generate physical plan | select l_returnflag, l_linestatus, |
sum(l_quantity) as sum_qty
from lineitem
where l_shipdate <= '199 |
| 46065 |             |           |         | Task    |    3 | Task runing                         | connection 45735, worker 0, task 1  |
| 46066 |             |           |         | Task    |    3 | Task runing                         | connection 45735, worker 1, task 1  |
| 46067 |             |           |         | Task    |    3 | Task runing                         | connection 45735, worker 2, task 1  |
| 46068 |             |           |         | Task    |    3 | Task runing                         | connection 45735, worker 3, task 1  |
+-------+-------------+-----------+---------+---------+------+-------------------------------------+-------------------------------------+
5 rows in set (0.00 sec)
```

https://cloud.tencent.com/document/product/1003/81872

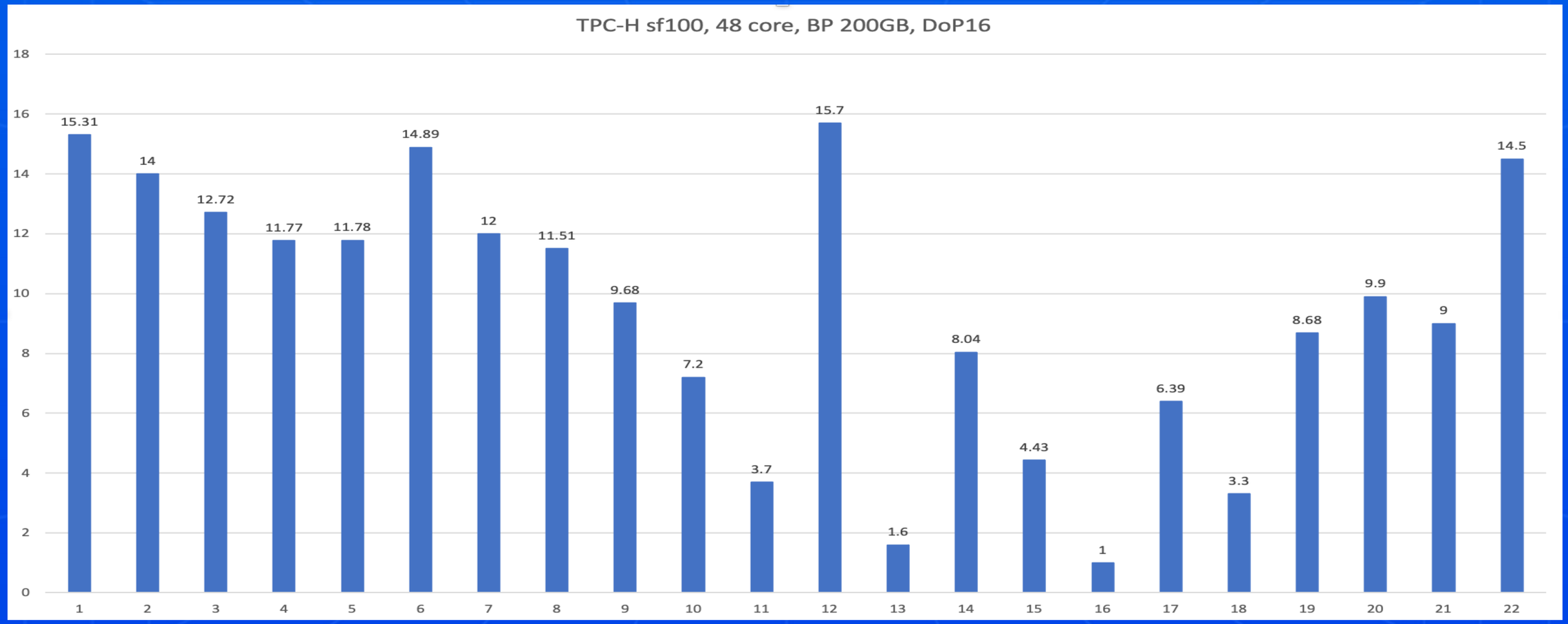
并行查询技术方案 – 计划切分





https://mp.weixin.qq.com/s/a7WM67ZWov24svTAyvwFLw

并行查询 TPC-H sf100 性能



Amdahl's law

$$S_{latency}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

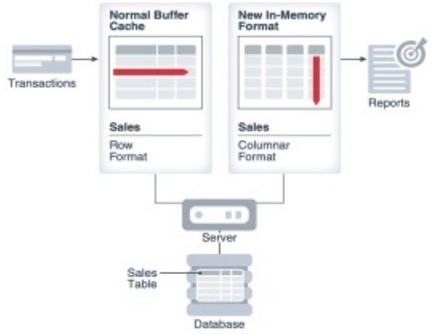
p - 可并行部分占总计算量的比例

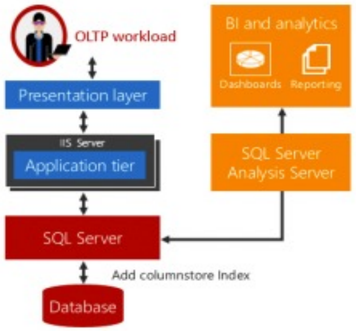
s – 上述部分的局部性能提升

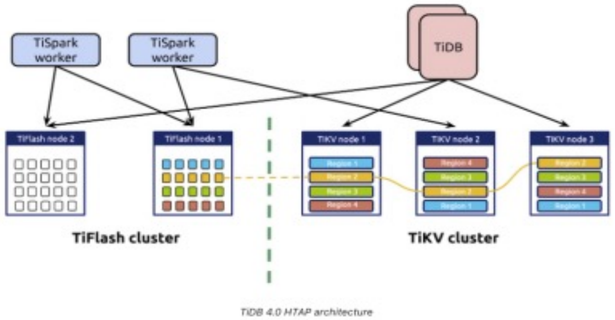
日程

- 背景
- HTAP在腾讯云数据库的演进
  - 并行查询
  - **列存索引**
- 未来展望

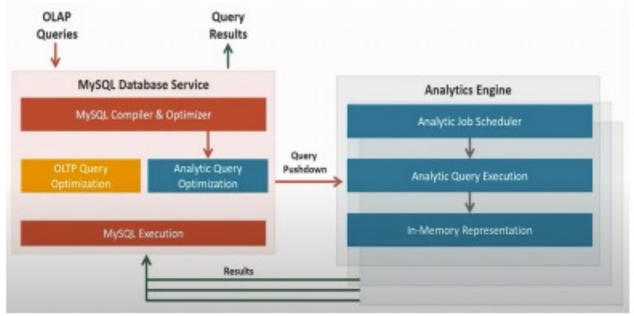
列存竞品现状







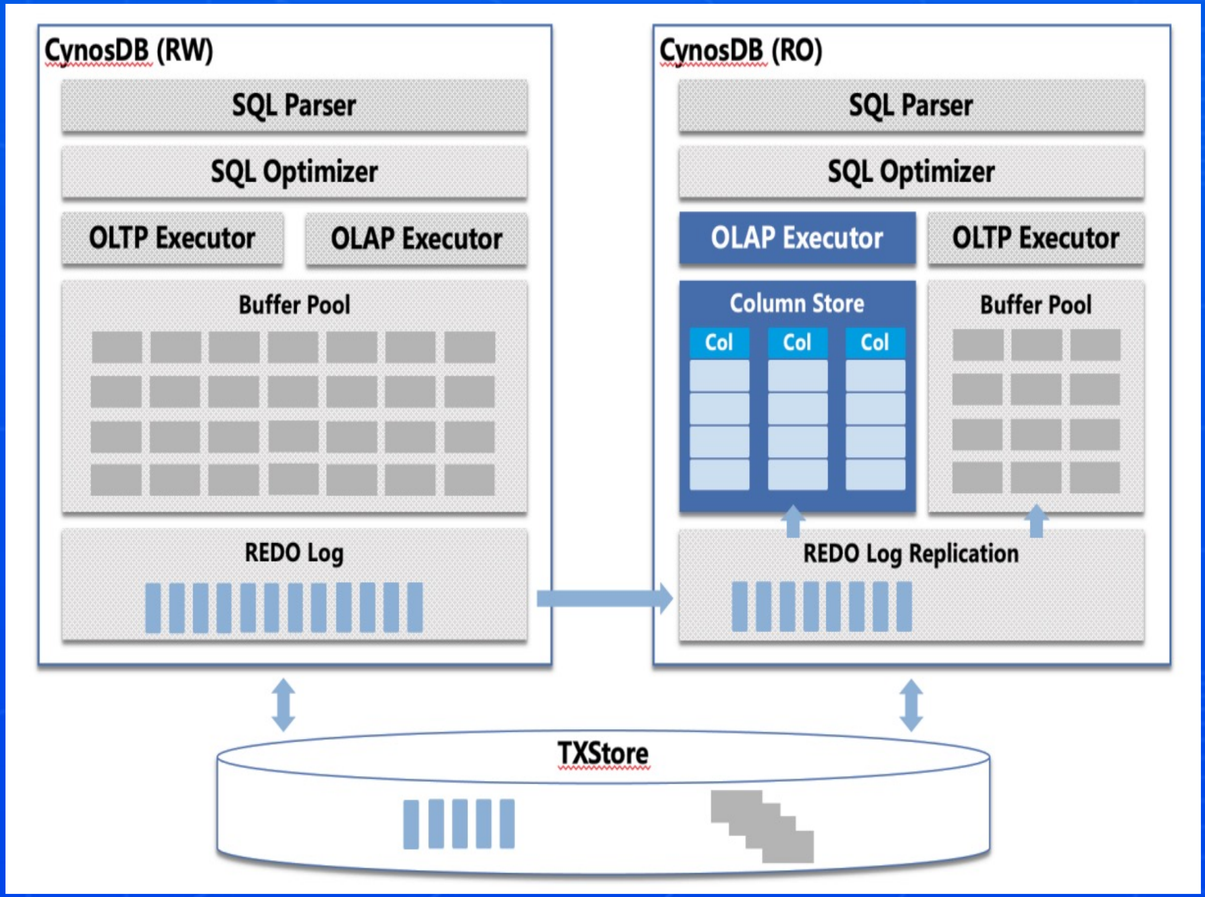
TiDB 4.0 HTAP architecture



列存引擎技术

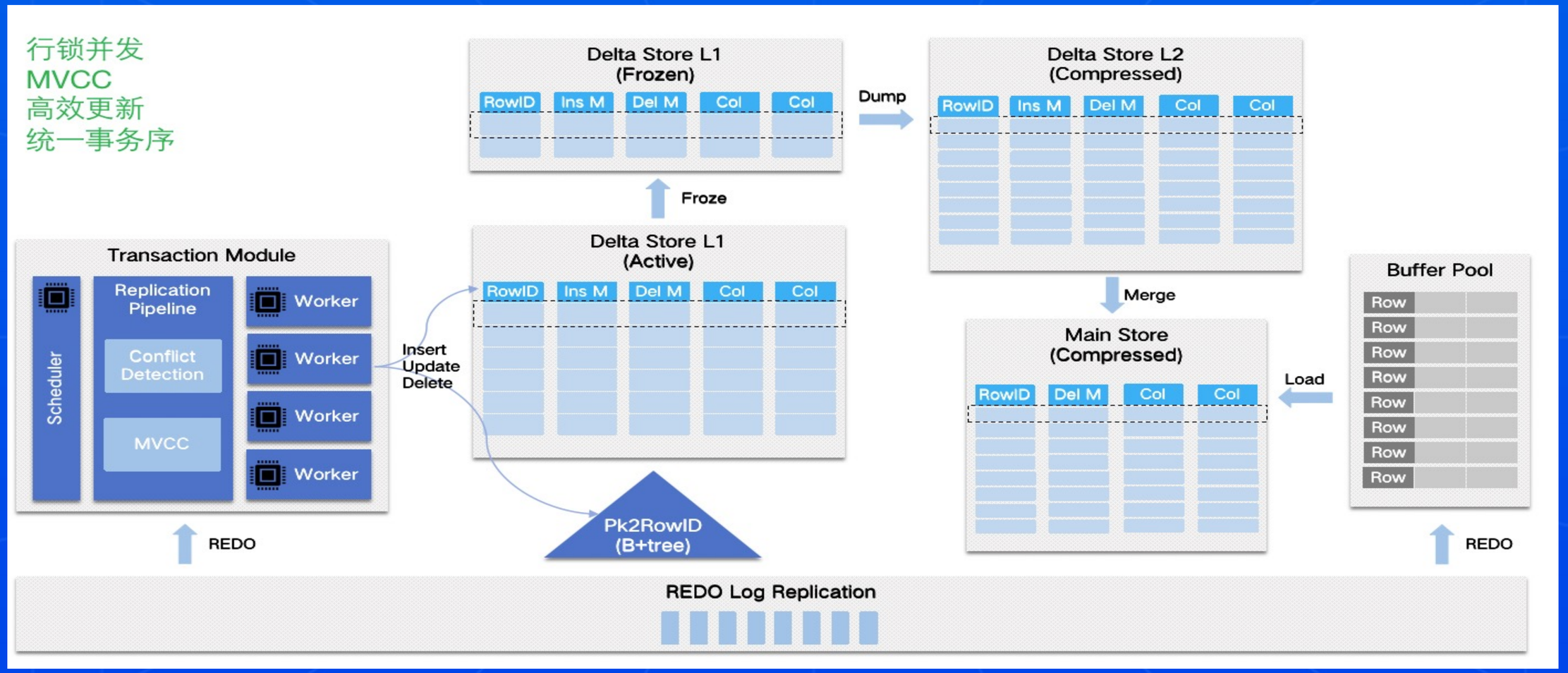
特性	Oracle	SQL Server	TiFlash	MySQL HeatWave
数据格式	一份数据两种格式 基于Row Group存储 列式数据全内存 数据选择性持久化	索引数据格式(支持聚簇和非聚簇列存索引) 基于Row Group存储 数据持久化	两份独立数据 纯列式文件存储 数据持久化	基于Row Group存储 数据多节点分区，内存存储 数据不持久化
向量化处理	支持	支持	支持	支持
部署形式	一体化方案 一个执行计划可以跨行存和列存两个引擎	一体化方案 一个执行计划可以跨行存和列存两个引擎	独立的RO节点 一个执行计划可以跨行存和列存两个引擎	独立的分析集群 MySQL优化器决定下推的列存计划 通过Secondry Engine接口和MySQL Server交互
并行处理能力	和Oracle的并行框架结合	和SQL Server的并行框架结合	CK并行框架和TiDB并行框架独立	支持MPP框架
数据同步和更新	通过内存列存数据(基线)和事务信息(增量)保证数据一致性	Columnstore+delta store保持一执性	基于Binlog复制保持一致	Binlog复制

列存索引架构

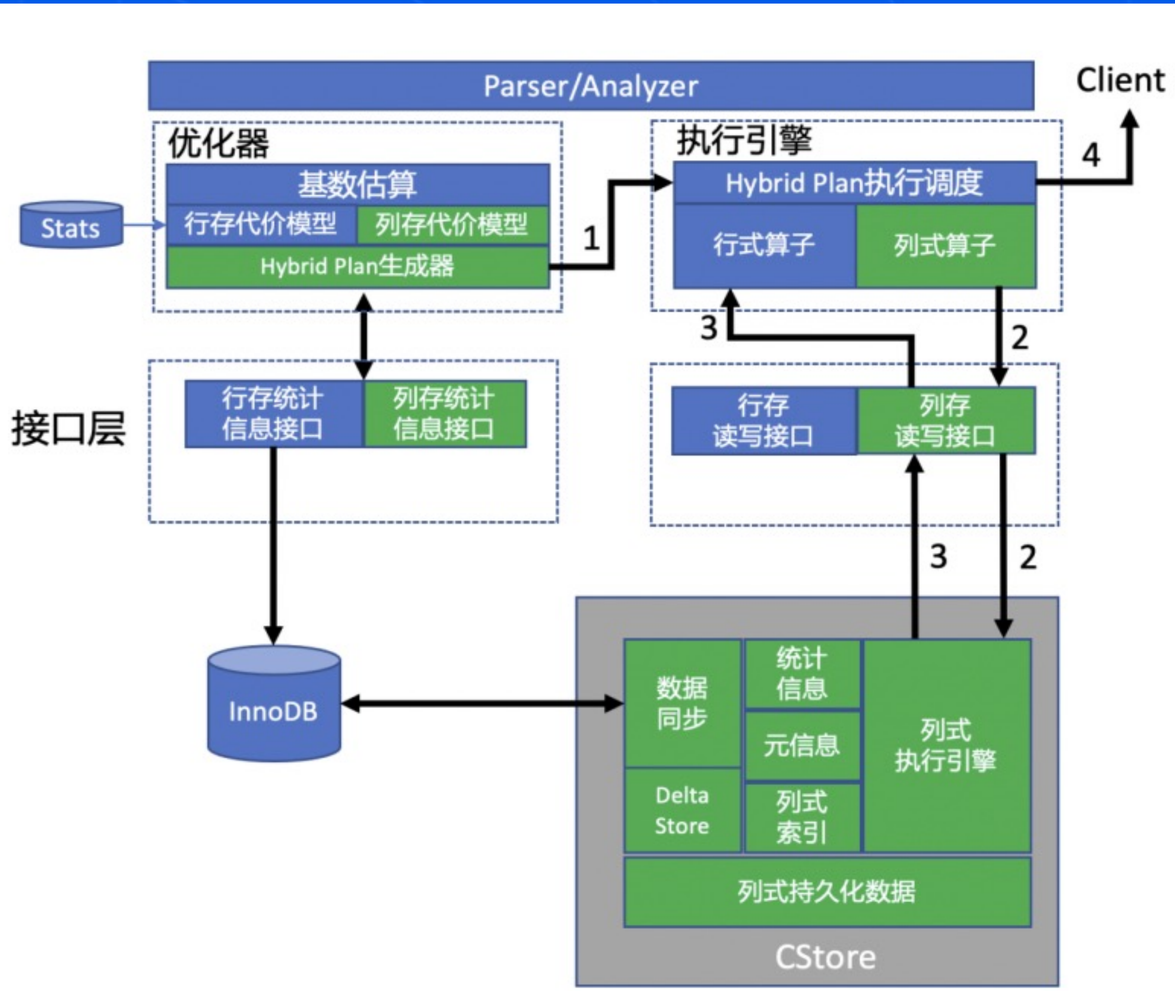


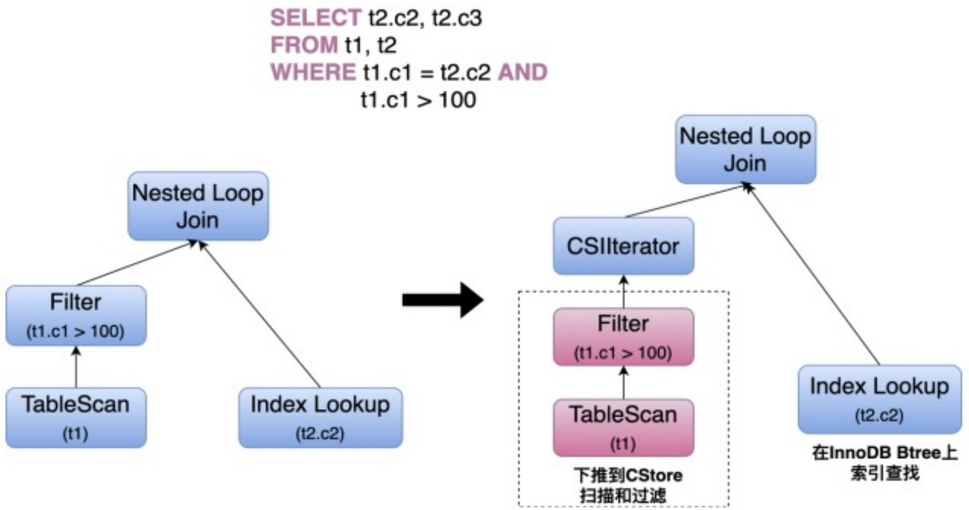
- 混合执行
  - Hybrid优化器
  - Hybrid执行调度
- 高效计算
  - 并行执行*
  - 基于AVX512的向量化
- 混合存储
  - 实现DeltaStore,支持高效写入和并发事务

列存索引 – Delta Store



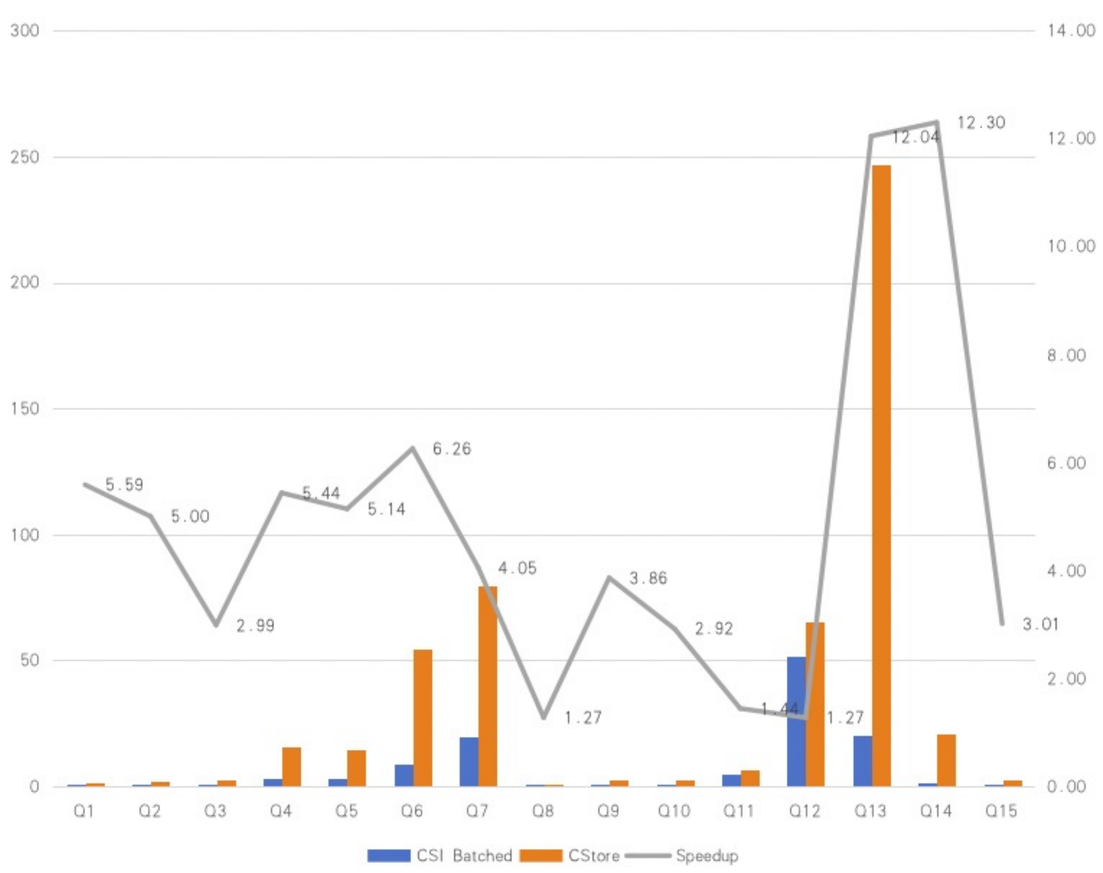
列存索引引擎执行流程





1. 优化器生成Hybrid Plan，由执行引擎调度执行
2. 列式算子通过接口层下推到Cstore执行
3. 数据以行式投影到Server层，继续执行行式算子
4. 最终结果写回client

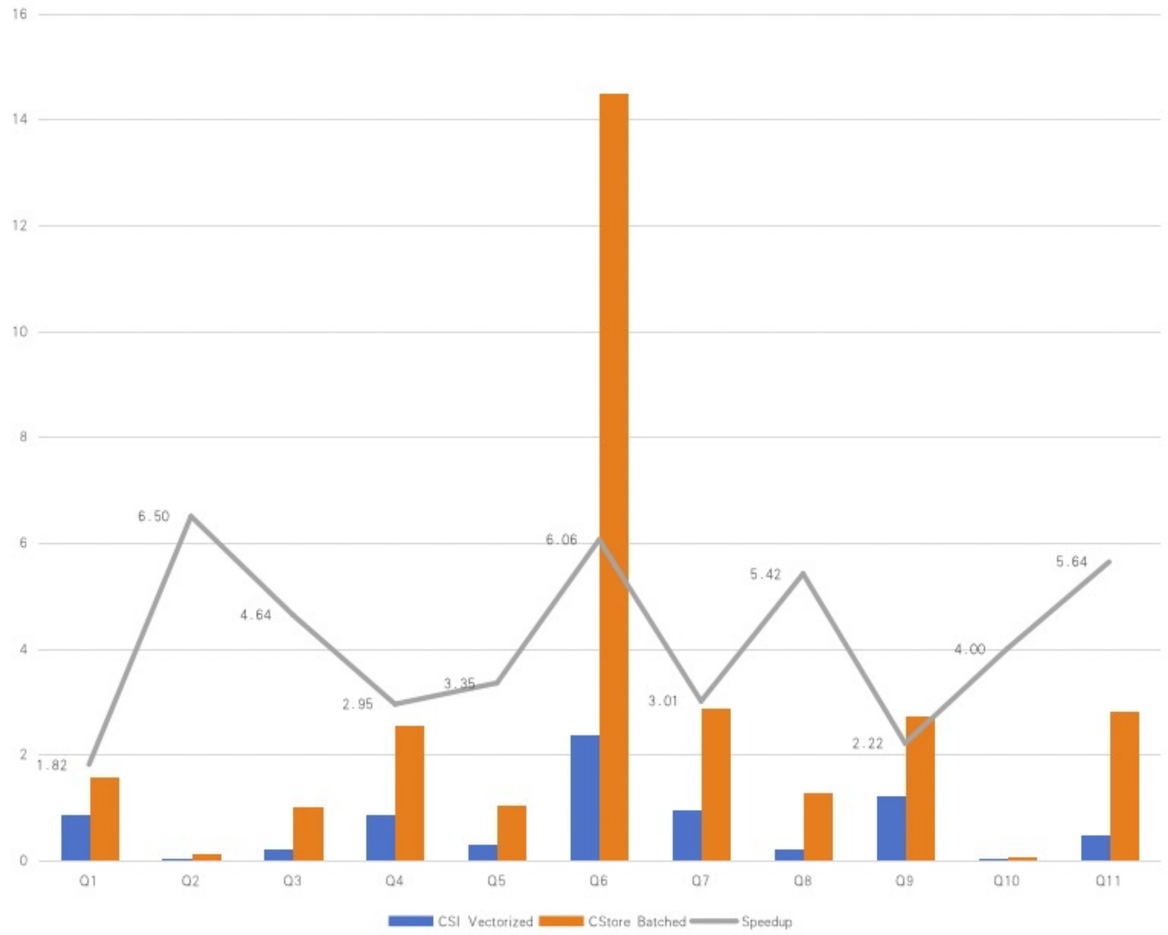
列存索引执行引擎 – 批处理




```
Q1
select count(*) from customer where c_mktsegment = 'BUILDING';
Q2
select count(*) from part where p_type = 'ECONOMY ANODIZED STEEL';
Q3
select count(*) from part where p_brand <> 'Brand#44'
Q4
select count(*) from orders where o_orderpriority = '1-URGENT';
Q5
select count(*) from orders where o_orderpriority <> '1-URGENT';
Q6
select count(*) from lineitem where l_returnflag = 'R';
Q7
select count(*) from lineitem where l_shipinstruct = 'DELIVER IN PERSON';
Q8
select count(*) from supplier where s_comment LIKE '%Customer%Complaints%';
Q9
select count(*) from part where p_name LIKE 'green%';
Q10
select count(*) from part where p_type NOT LIKE 'MEDIUM POLISHED%';
Q11
select count(*) from part where p_name LIKE '%green%';
Q12
select count(*) from orders where o_comment NOT LIKE '%pending%deposits%';
Q13
select count(*) from lineitem where l_shipmode IN ('MAIL', 'SHIP');
Q14
select count(*) from customer where substring(c_phone FROM 1 FOR 2) IN ('20', '40', '22',
'30', '39', '42', '21');
Q15
select count(*) from part where p_brand = 'Brand#44' AND p_container = 'WRAP PKG';
```

Single Core, Scale Factor = 100

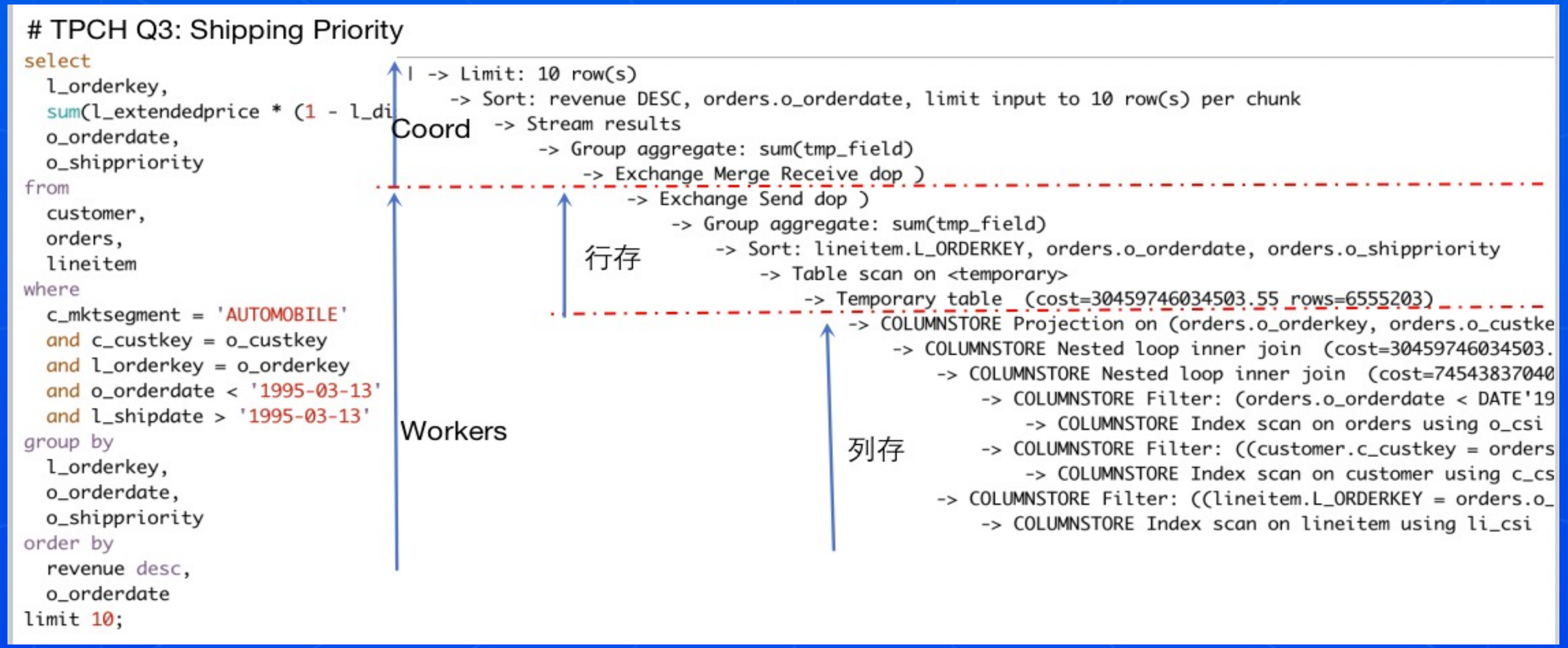
列存索引执行引擎 – SIMD指令的性能




```
Q1
SELECT count(*) FROM lineitem WHERE l_shipdate <= DATE '1998-12-01' - INTERVAL '90'
DAY;
Q2
SELECT count(*) FROM part WHERE p_size = 30;
Q3
SELECT count(*) FROM orders WHERE o_orderdate < DATE '1995-03-15';
Q4
SELECT count(*) FROM lineitem WHERE l_shipdate > DATE '1995-03-15';
Q5
SELECT count(*) FROM orders WHERE o_orderdate >= DATE '1994-01-01'
  AND o_orderdate < DATE '1994-01-01' + INTERVAL '1' YEAR;
Q6
SELECT count(*) FROM lineitem WHERE l_shipdate >= DATE '1994-01-01'
  AND l_shipdate < DATE '1994-01-01' + INTERVAL '1' YEAR
  AND l_discount BETWEEN .06 - 0.01 AND .06 + 0.01
  AND l_quantity < 24;
Q7
SELECT count(*) FROM lineitem WHERE l_shipdate BETWEEN DATE '1995-01-01' AND DATE
'1996-12-31';
Q8
SELECT count(*) FROM orders WHERE o_orderdate BETWEEN DATE '1995-01-01' AND DATE
'1996-12-31';
Q9
SELECT count(*) FROM lineitem WHERE l_receiptdate >= DATE '1994-01-01'
  AND l_receiptdate < DATE '1994-01-01' + INTERVAL '1' YEAR;
Q10
SELECT count(*) FROM part WHERE p_size BETWEEN 1 AND 5;
Q11
SELECT count(*) FROM lineitem WHERE l_quantity >= 1 AND l_quantity <= 1 + 10;
```

Single Core, Scale Factor = 100

列存索引+并行查询 示例


TPCH Q3: Shipping Priority
select
l_orderkey,
sum(l_extendedprice * (1 - l_di
o_orderdate,
o_shippriority
from
customer,
orders,
lineitem
where
c_mktsegment = 'AUTOMOBILE'
and c_custkey = o_custkey
and l_orderkey = o_orderkey
and o_orderdate < '1995-03-13'
and l_shipdate > '1995-03-13'
group by
l_orderkey,
o_orderdate,
o_shippriority
order by
revenue desc,
o_orderdate
limit 10;
Coord
Workers
行存
列存
| -> Limit: 10 row(s)
-> Sort: revenue DESC, orders.o_orderdate, limit input to 10 row(s) per chunk
-> Stream results
-> Group aggregate: sum(tmp_field)
-> Exchange Merge Receive dop)
-> Exchange Send dop)
-> Group aggregate: sum(tmp_field)
-> Sort: lineitem.L_ORDERKEY, orders.o_orderdate, orders.o_shippriority
-> Table scan on <temporary>
-> Temporary table (cost=30459746034503.55 rows=6555203)
-> COLUMNSTORE Projection on (orders.o_orderkey, orders.o_custke
-> COLUMNSTORE Nested loop inner join (cost=30459746034503.
-> COLUMNSTORE Nested loop inner join (cost=74543837040
-> COLUMNSTORE Filter: (orders.o_orderdate < DATE'19
-> COLUMNSTORE Index scan on orders using o_csi
-> COLUMNSTORE Filter: ((customer.c_custkey = orders
-> COLUMNSTORE Index scan on customer using c_cs
-> COLUMNSTORE Filter: ((lineitem.L_ORDERKEY = orders.o_
-> COLUMNSTORE Index scan on lineitem using li_csi

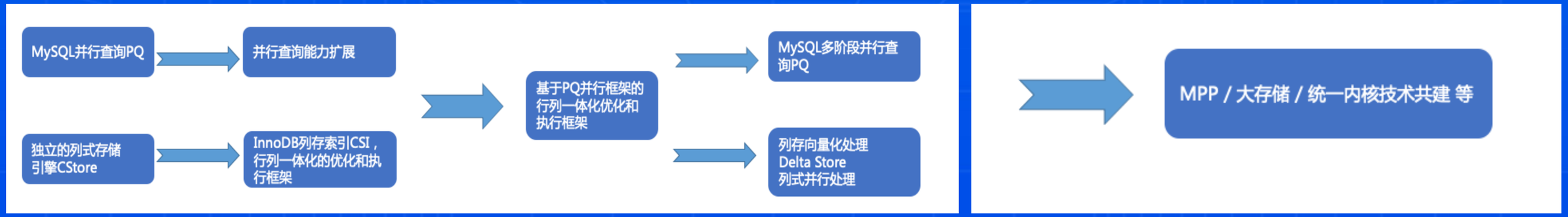
列存索引+并行查询性能

- 产品在灰度测试和内部客户试用中
- TPC-H 性能在 PQ 的基础上又有平均 3 倍左右的提升
- 预期在向量化完成之后 TPC-H 性能会再有更大的提升

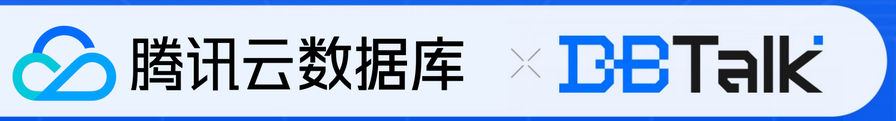
日程

- 背景
- HTAP在腾讯云数据库的演进
  - 并行查询
  - 列存索引
- **未来展望**

未来展望


MySQL并行查询PQ
并行查询能力扩展
独立的列式存储引擎CStore
InnoDB列存索引CSI，行列一体化的优化和执行框架
基于PQ并行框架的行列一体化优化和执行框架
MySQL多阶段并行查询PQ
列存向量化处理
Delta Store
列式并行处理
MPP / 大存储 / 统一内核技术共建 等



THANKS