

Apache Doris in 2023 与创新者同行

衣国垒

Apache Doris PMC 成员、飞轮科技技术副总裁

目录

1. Apache Doris in 2023
2. 技术进展回顾
3. 走向实时分析的下一步

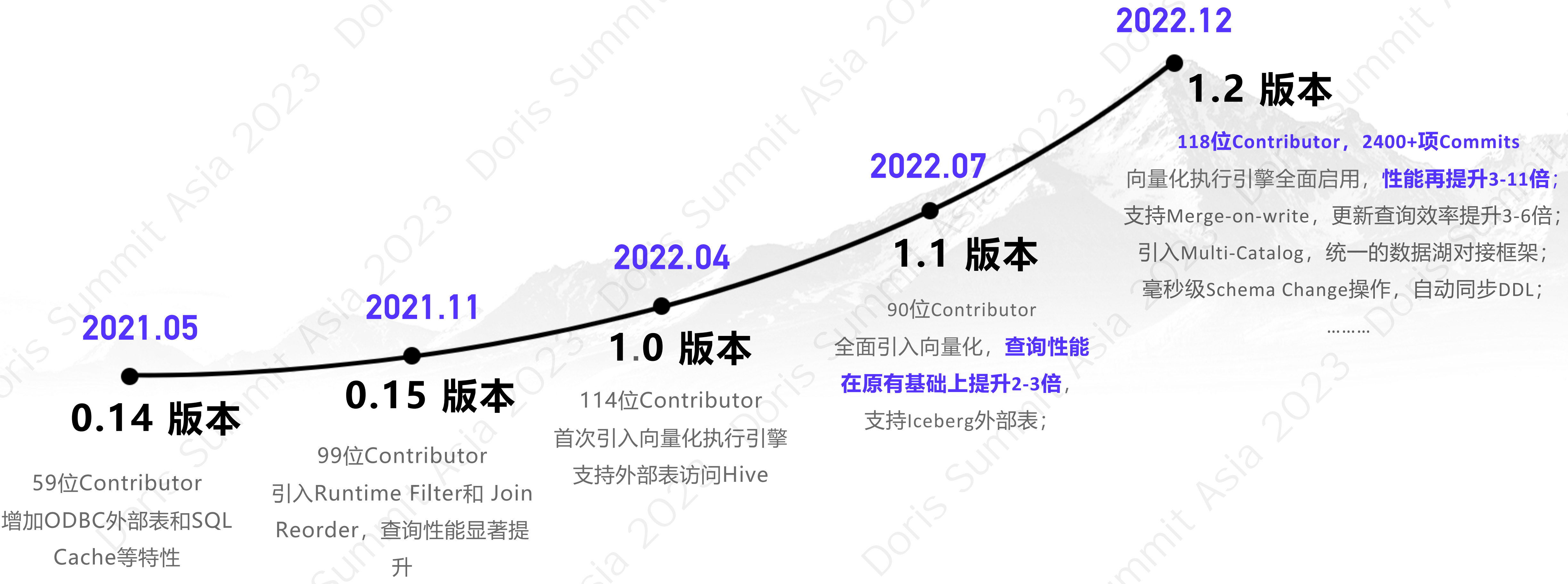
1 Apache Doris in 2023

重要版本迭代

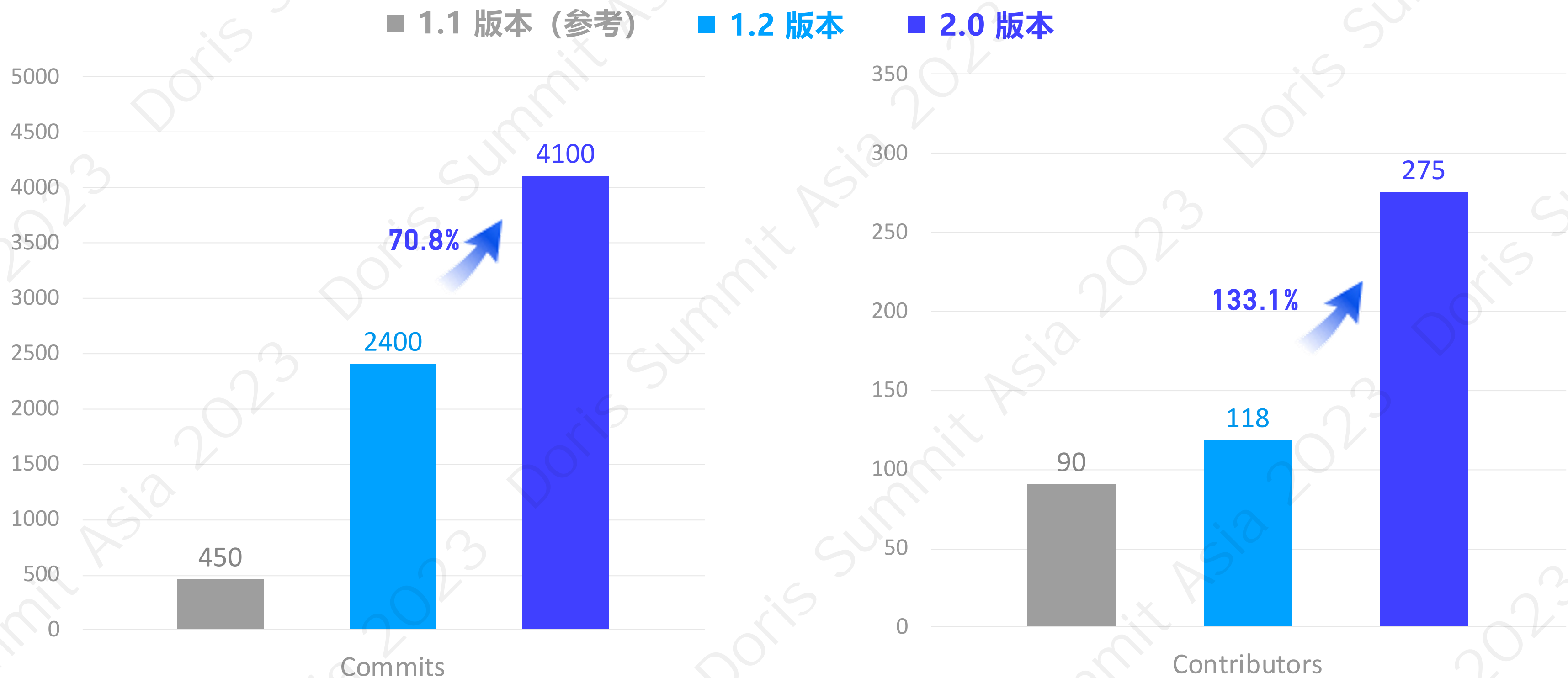
繁荣的社区生态

用户规模

回顾 Apache Doris 的过去版本，一直在加速进化



2023 年，Apache Doris 全面进入 2.0 时代

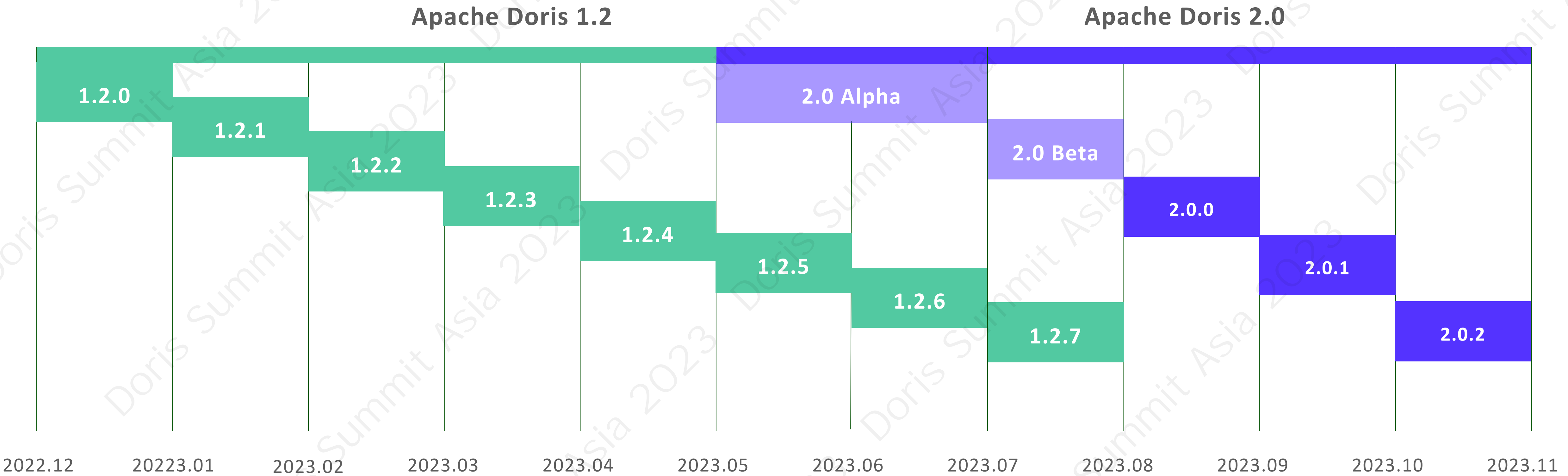


2.0 版本在 Apache Doris 发展历程中具有重要的里程碑意义：

- 引入自适应的并行执行模型和全新查询优化器，盲测性能提升**10**倍，多表关联提升**13**倍，单表场景提升**10**倍、高并发点查询提升**20**倍；
- 从报表和Ad-hoc等典型OLAP场景拓展到湖仓一体、高并发数据服务以及日志检索与分析，支撑更统一多样的分析场景；
- 支持实时数据高吞吐写入、秒级时延，对各类数据更新都有完备的支持，构建更高效易用且稳定的实时数据处理和分析链路；

建立了更加成熟稳定的版本迭代机制

- **更稳定的版本体验：** 经历 Alpha、Beta 两个验证性版本并经过大规模邀测后，2.0 版本正式 GA，版本稳定性更加契合企业用户生产环境的要求；
- **周期性发版：** 在大版本基础上以每月1个小版本的节奏稳定迭代，共发布 9 个小版本，累计优化功能及修复问题超过 1969 个；



Apache Doris in 2023

重要版本迭代

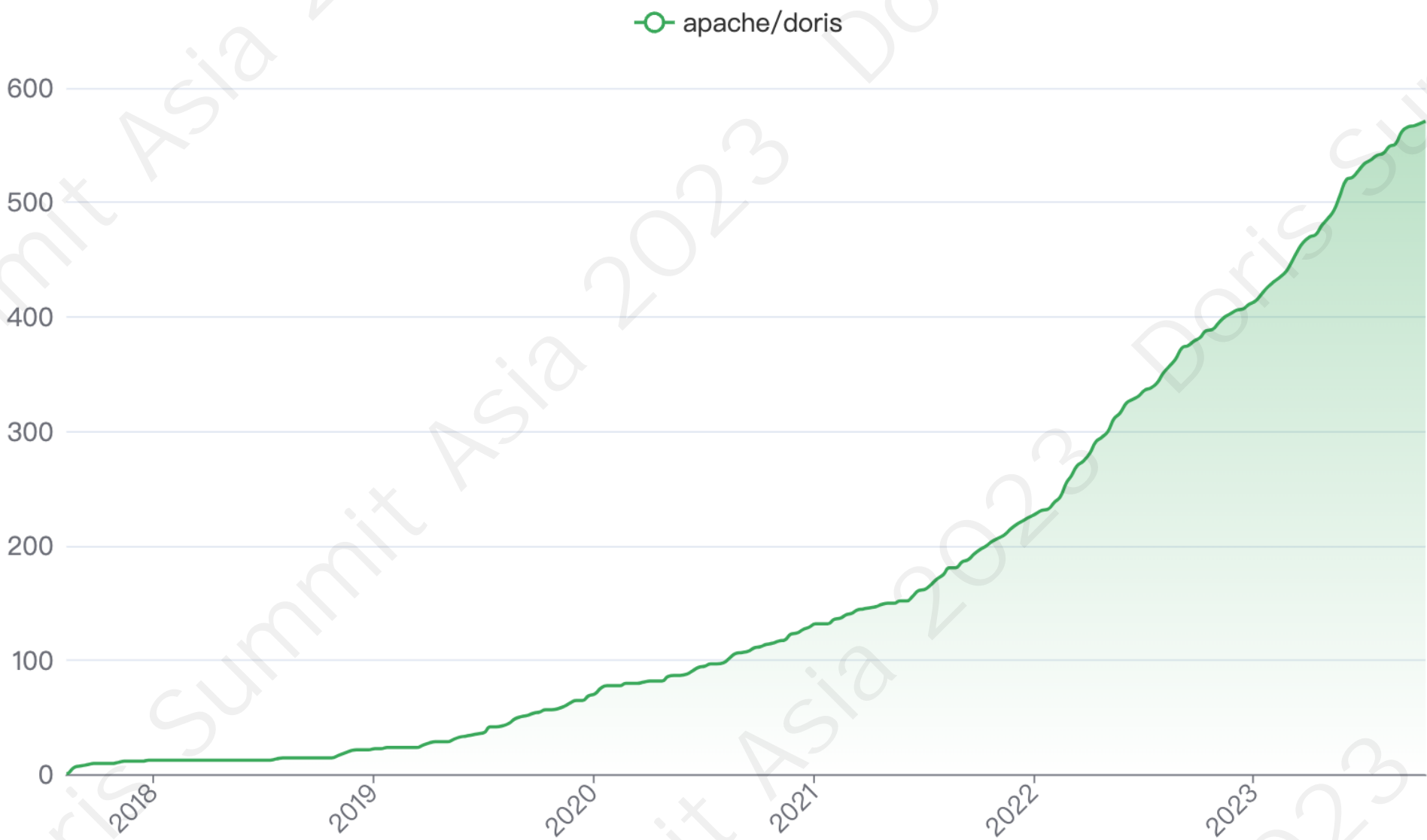
繁荣的社区生态

用户规模

更加繁荣的社区生态，更大的开发者规模和更高的开发者活跃度

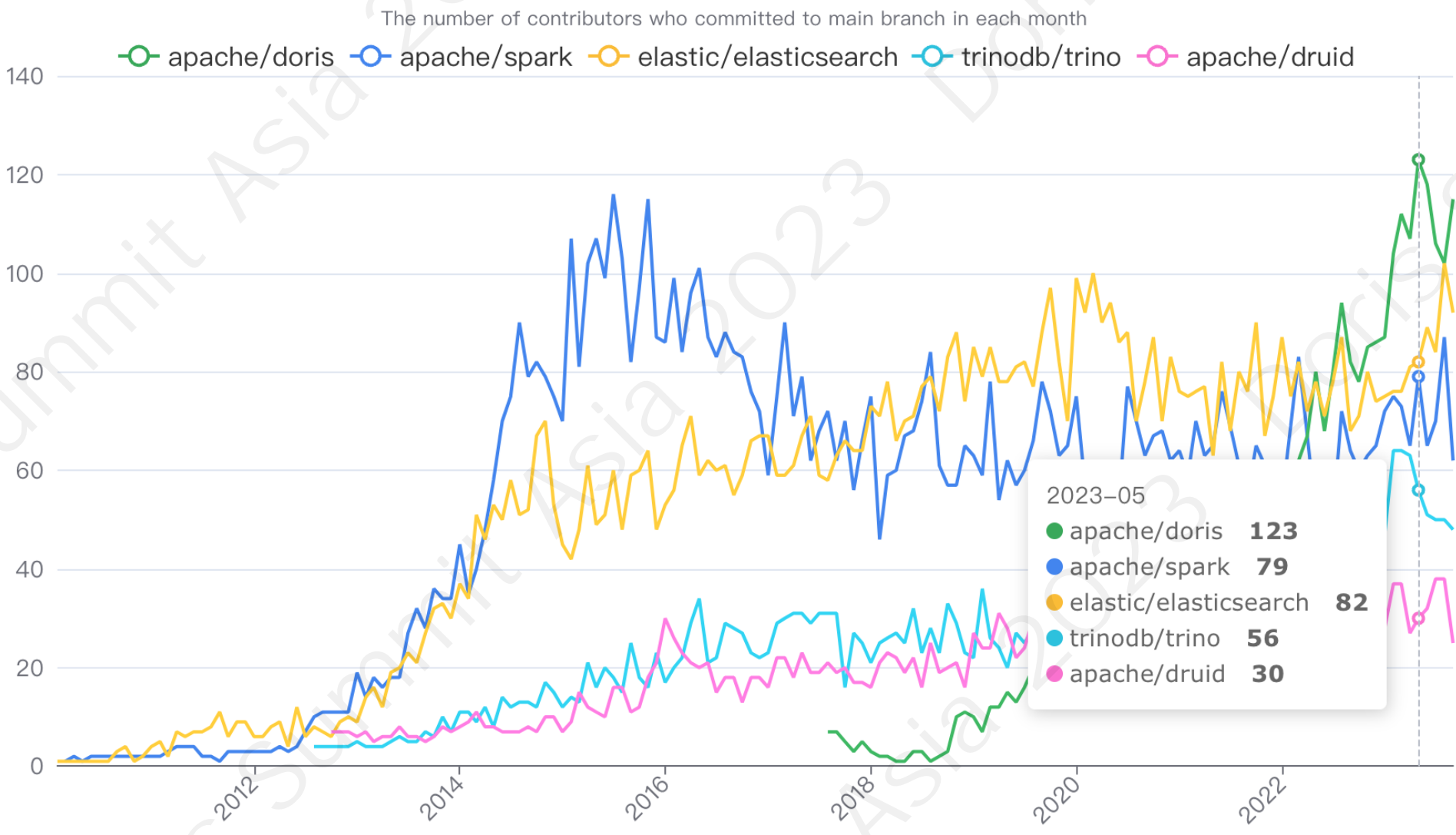
社区贡献者规模稳步增长，活跃贡献者数稳居全球前列

Contributor Over Time



- Contributor数量增长至 574人，较去年增长 67%

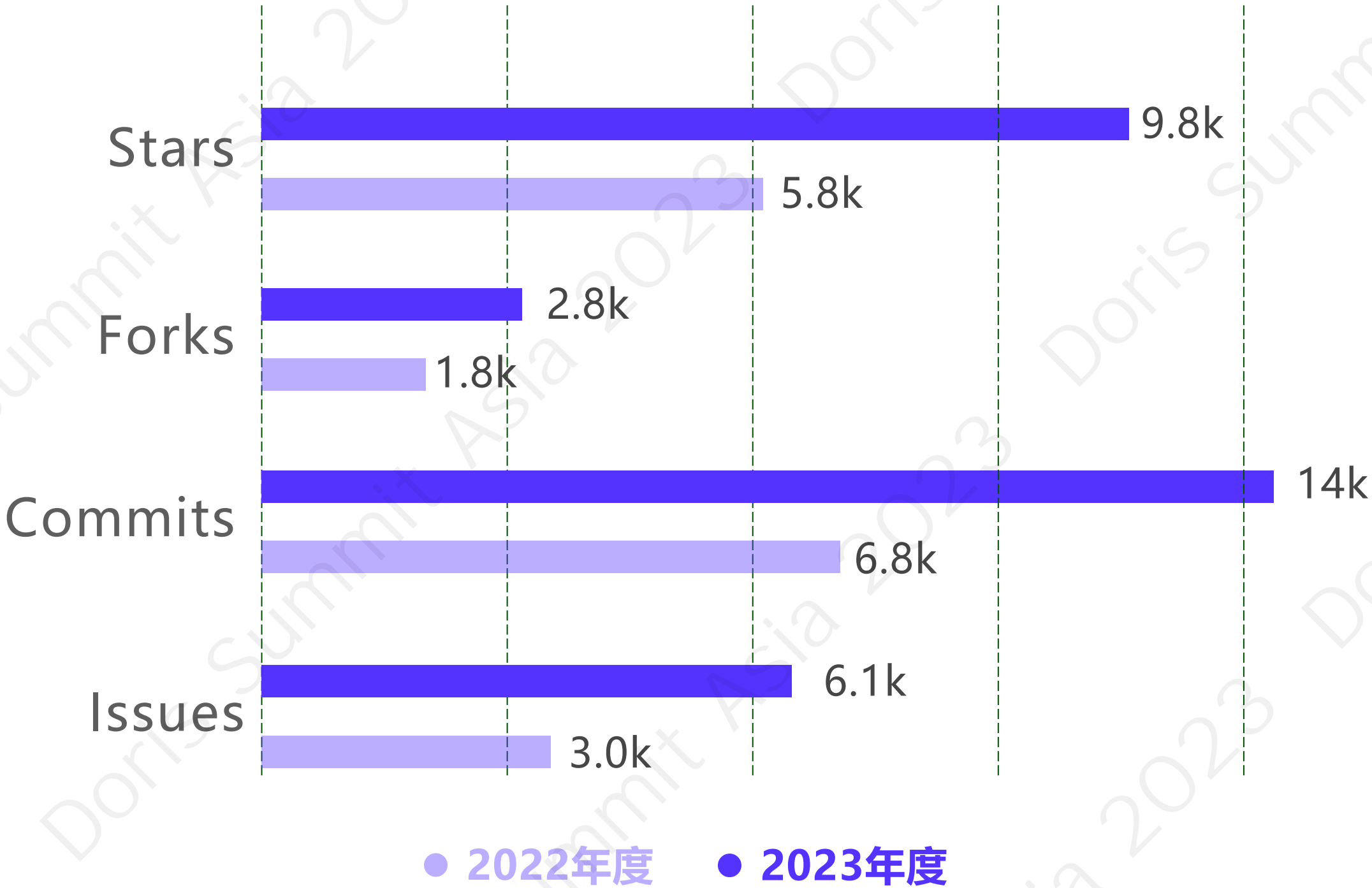
Monthly Active Contributors



- 近一年活跃贡献者规模稳居全球开源大数据项目第一位；

更加繁荣的社区生态，更大的开发者规模和更高的开发者活跃度

与去年同期相比，社区开发者活跃度全面提升

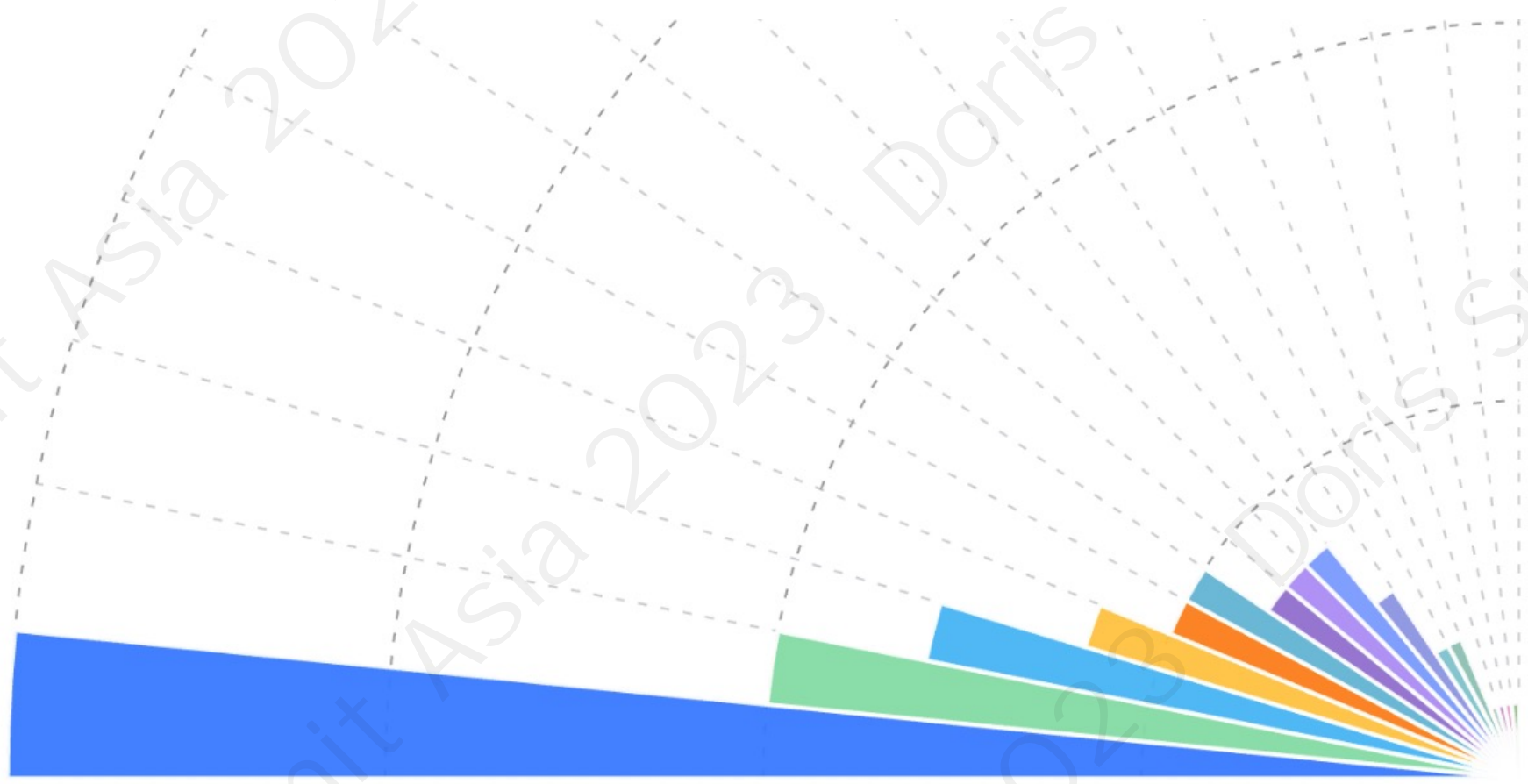


Stars增长 **69%** Commits 增长 **106%**
每周PRs **160+** 活跃度全球 **Top4**

- GitHub Star 较去年增长 49%，增速位列全球数据库前五；
- 近半年平均周合入Commits160+，活跃度位居OSSRank前四；
- Fork 较去年同期增长55%，更多开发者开始加入社区开发；
- 平均PR/Issue相应周期较去年提速300%；

更加繁荣的社区生态，更大的开发者规模和更高的开发者活跃度

贡献者来源更加多元化，国内顶尖云厂商纷纷投入共建



- 贡献者来源更加多元化，覆盖数十行业的 100 余家企业



- 国内顶尖云厂商投入共建，相关产品几乎覆盖国内外所有主流云平台

Apache Doris in 2023

重要版本迭代

繁荣的社区生态

用户规模

Apache Doris 已成为开源实时数据仓库领域的事实标准！



企业用户规模已超过 4000 家，在众多中大型企业核心分析业务中得到广泛应用。

2 我们如何应对实时分析的挑战

当我们重新思考 Apache Doris 的定位

02 融合统一

在一套系统中提供对多种分析负载的支持，简化复杂架构带来的运维使用成本

湖仓联邦分析

日志存储与检索分析

ETL/ELT加速

高并发数据服务

报表分析

即席分析

01 实时分析

在大规模实时数据上实现极致的查询性能

极致查询性能

实时高吞吐写入

实时存储与更新

03 云原生

面向云计算基础设施进行革新，利用云的极致弹性降低存储和计算成本

存储计算分离

多计算集群

弹性扩缩容

K8s 容器化部署

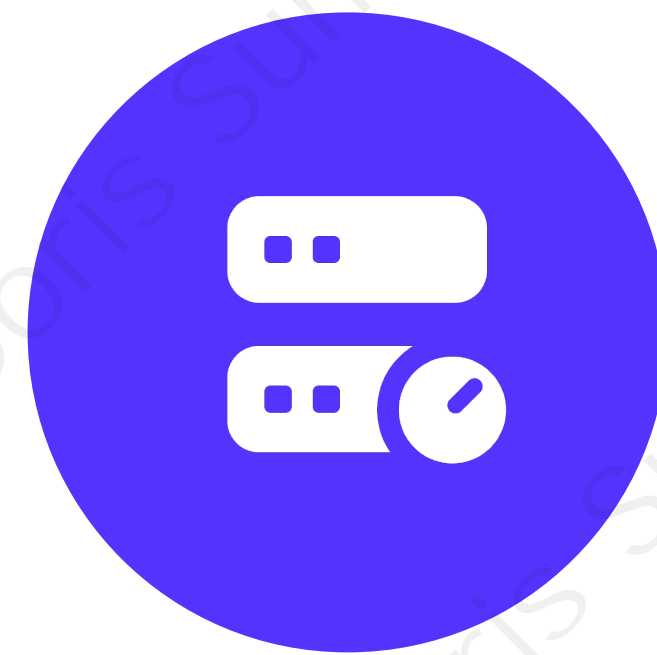
Open-source Data Warehouse for Real-time Analytics

2023 年聚焦开发方向



极致查询性能

自适应的并行执行模型
基于代价的查询优化器
多维度快速检索
高并发点查询能力



数据实时写入/更新

主键模型写时合并
完备的数据更新支持
导入性能优化



更多分析场景支持

湖仓一体 Lakehouse
更高性价比的日志分析平台
高并发数据服务场景



高可用/低成本

冷热数据分层
跨集群数据同步
多租户资源隔离

技术进展回顾

查询性能

实时写入/更新

更多分析场景

高可用、低成本

全新查询优化器与 Pipeline 并行执行模型

CBO
查询优化器



Pipeline
并行执行
模型

阻塞操作异步化

灵活的CPU
调度策略

并行化改造与
资源池化

资源组调度

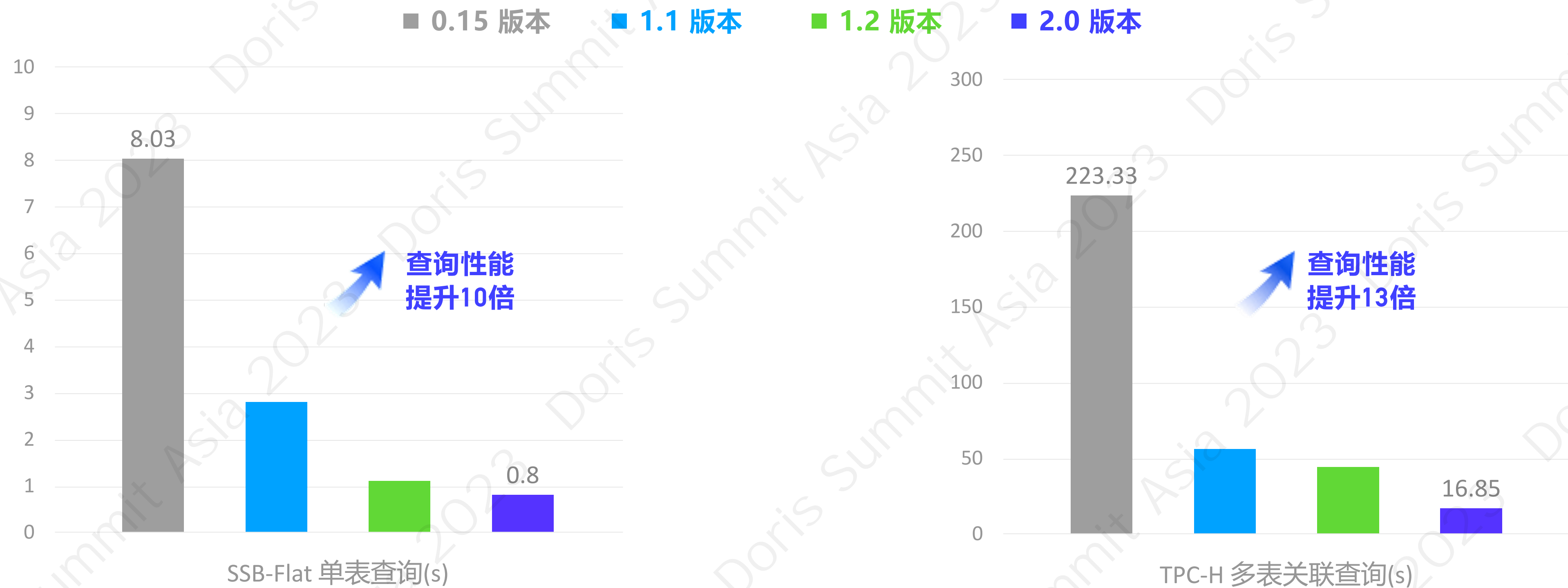


Workload Group 避免资源抢占

大小查询都可以分配到CPU资源
执行计算结果，混合负载下查询
性能更高

资源队列、查询排队

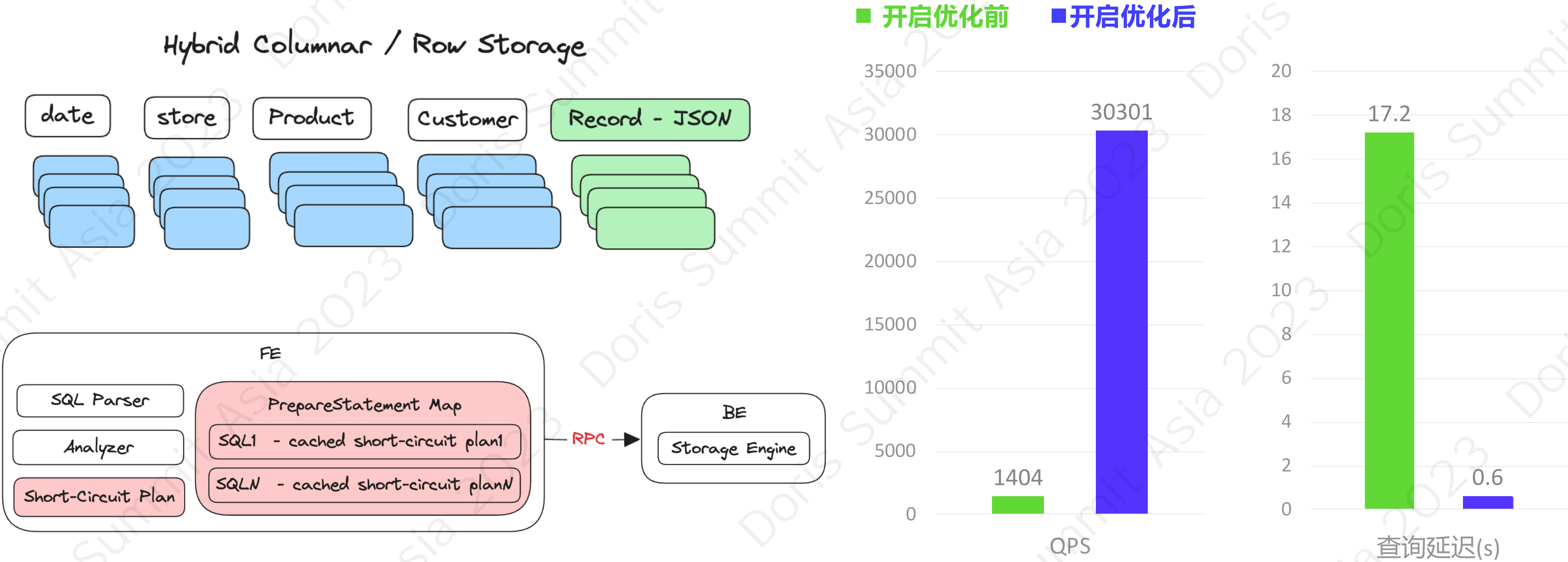
在多表关联和单表场景下均取得10倍以上的查询性能提升



在同等规模数据量和机器配置下，性能超越一众同类项目：

- 3 台 16 Core、64GB 云主机测试，SF100
- 多表关联复杂查询场景 Doris 2.0 性能相比 Doris 0.15 提升 13 倍，相比其他的MPP 数据库有明显优势
- 单表场景 Doris 2.0 性能相比 Doris 0.15 提升 10 倍，相比擅长单表的 ClickHouse 更有优势；

点查询并发能力提升20倍，单节点 3w+ QPS

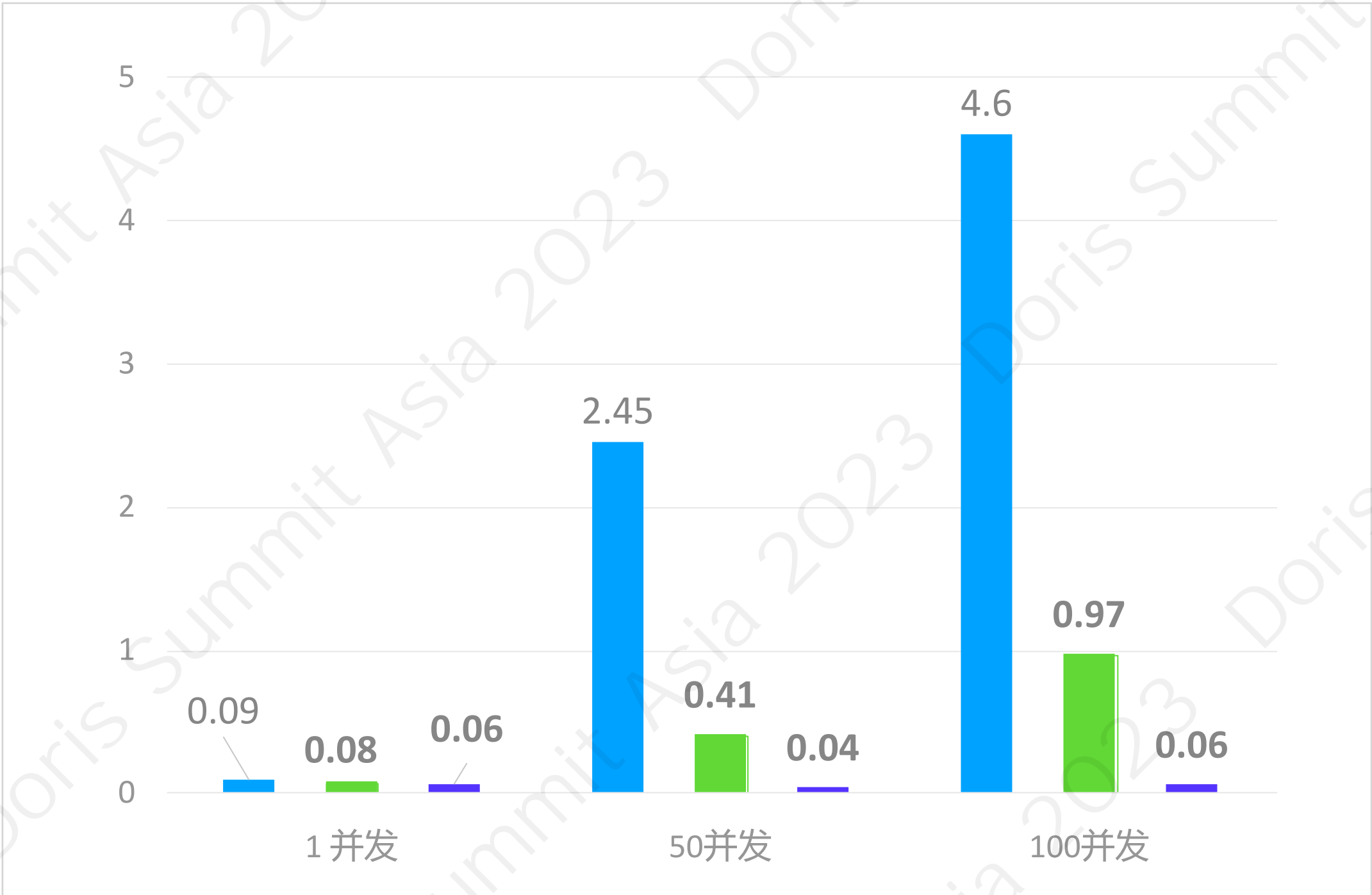
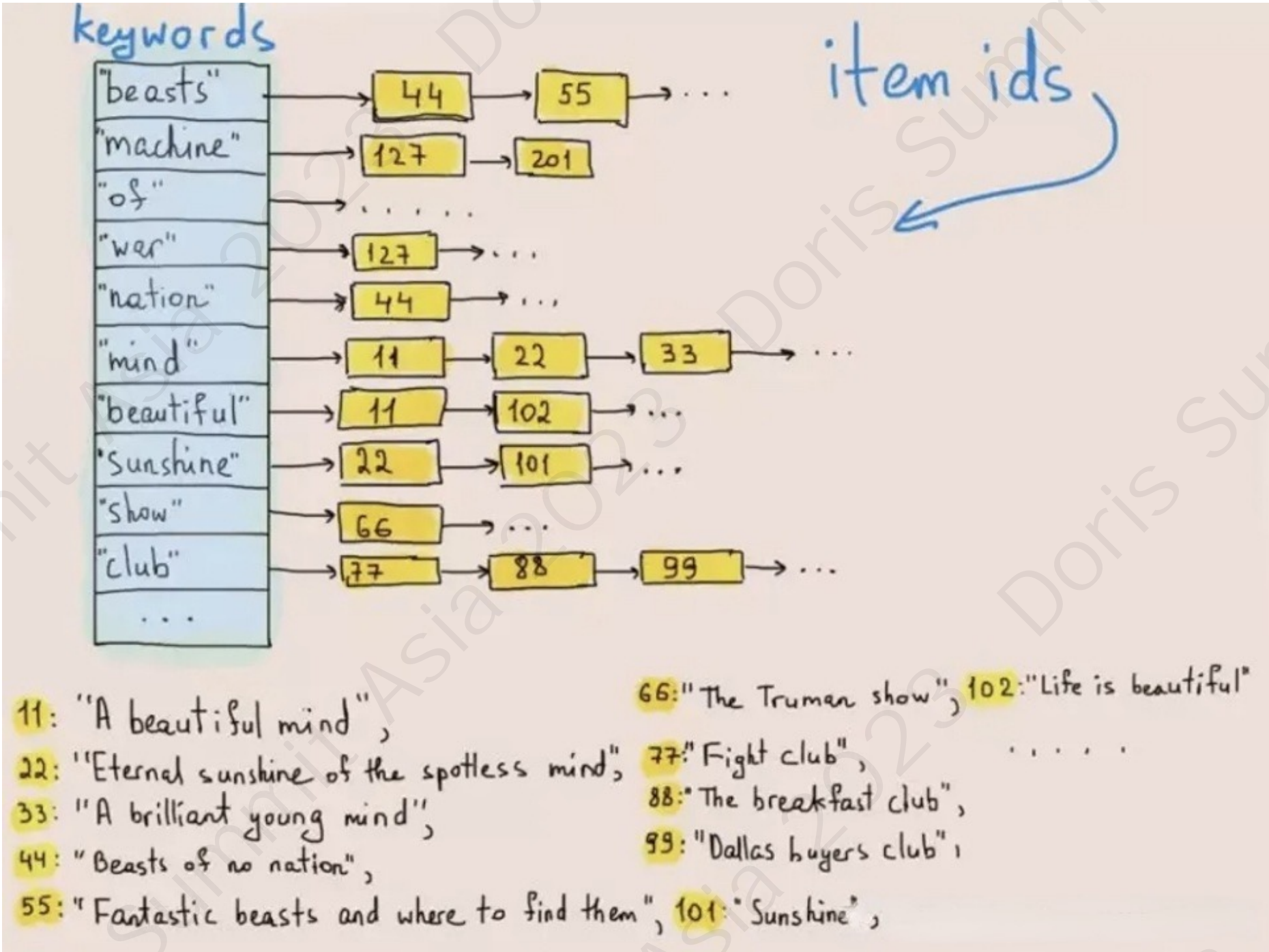


面向高并发 Data Serving 场景的优化方案，主键点查询能力提升 20 倍

- 引入行列混存，解决 IOPS 瓶颈；引入 PreparedStatement 解决FE 模块高并发下的解析规划瓶颈；
- 引入点查询短路径优化，跳过执行引擎和查询优化器对于简单查询的框架开销；

多维度检索并发性能最高提升90倍

1.1 版本 1.2 版本 2.0 版本



引入倒排索引来应对多维度快速检索的需求，在真实业务场景中取得性能最高90倍提升：

- 在关键字模糊查询、等值查询和范围查询等场景中均取得了显著的查询性能和并发能力提升；
- 开启倒排索引前，并发量的提升带来查询耗时的大幅上升，开启倒排索引后始终保持毫秒级；

技术进展回顾

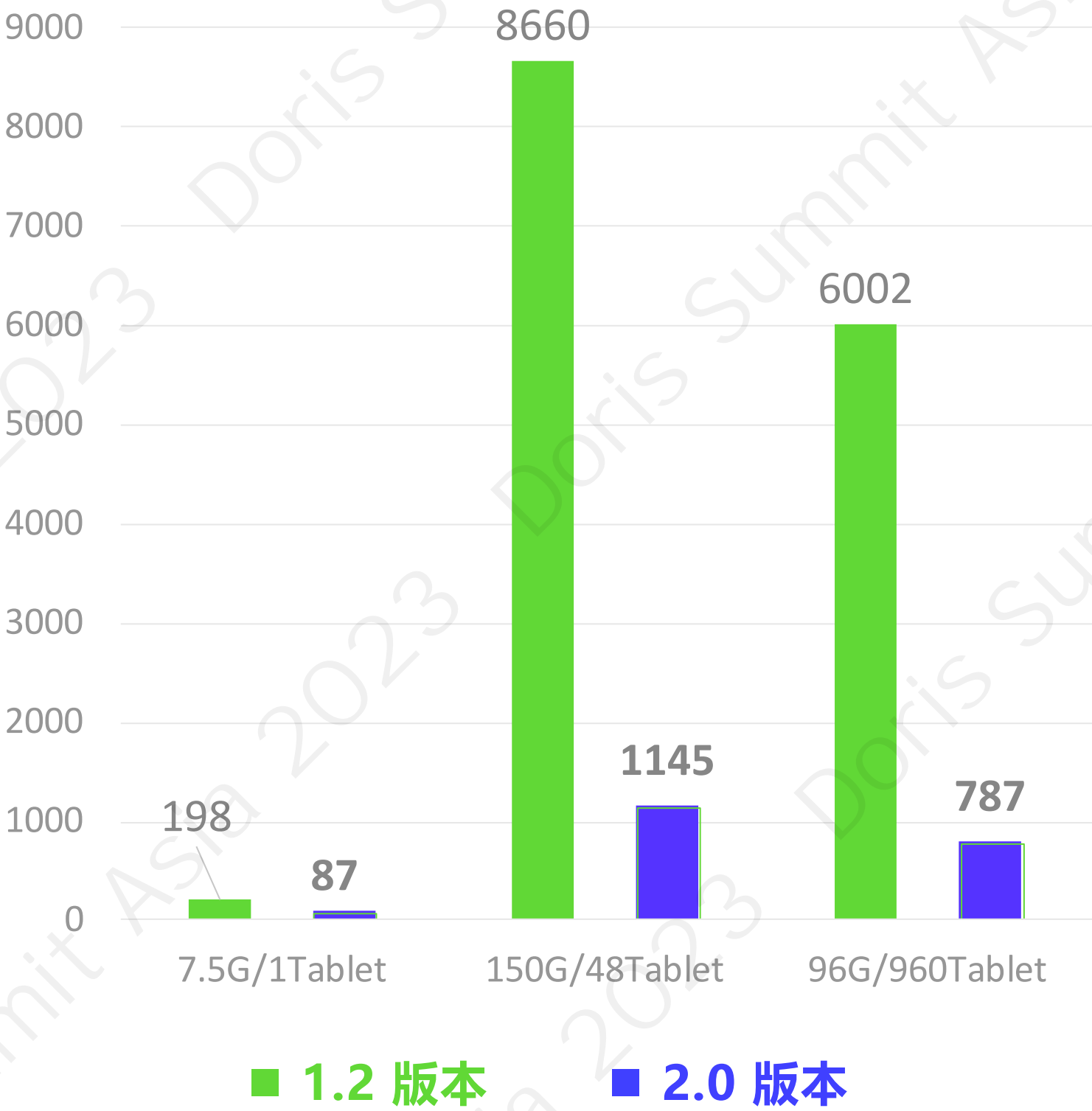
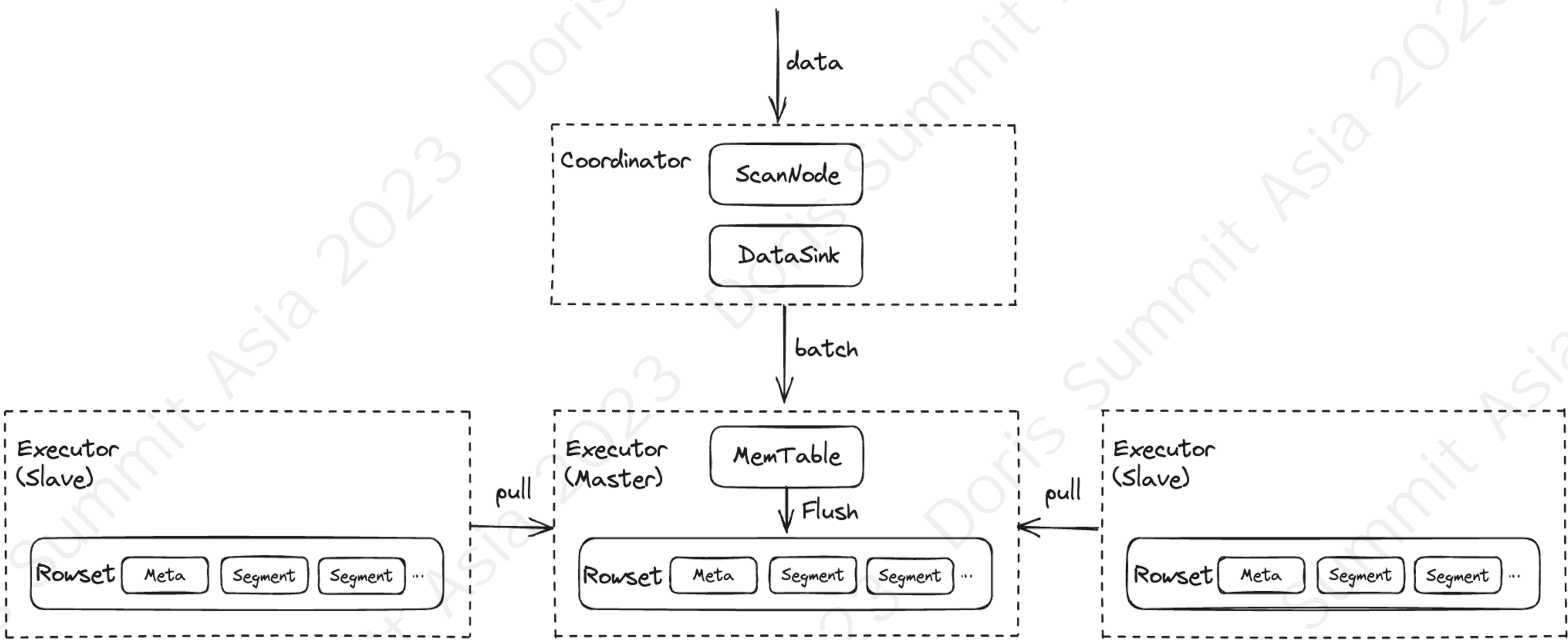
查询性能

实时写入/更新

更多分析场景

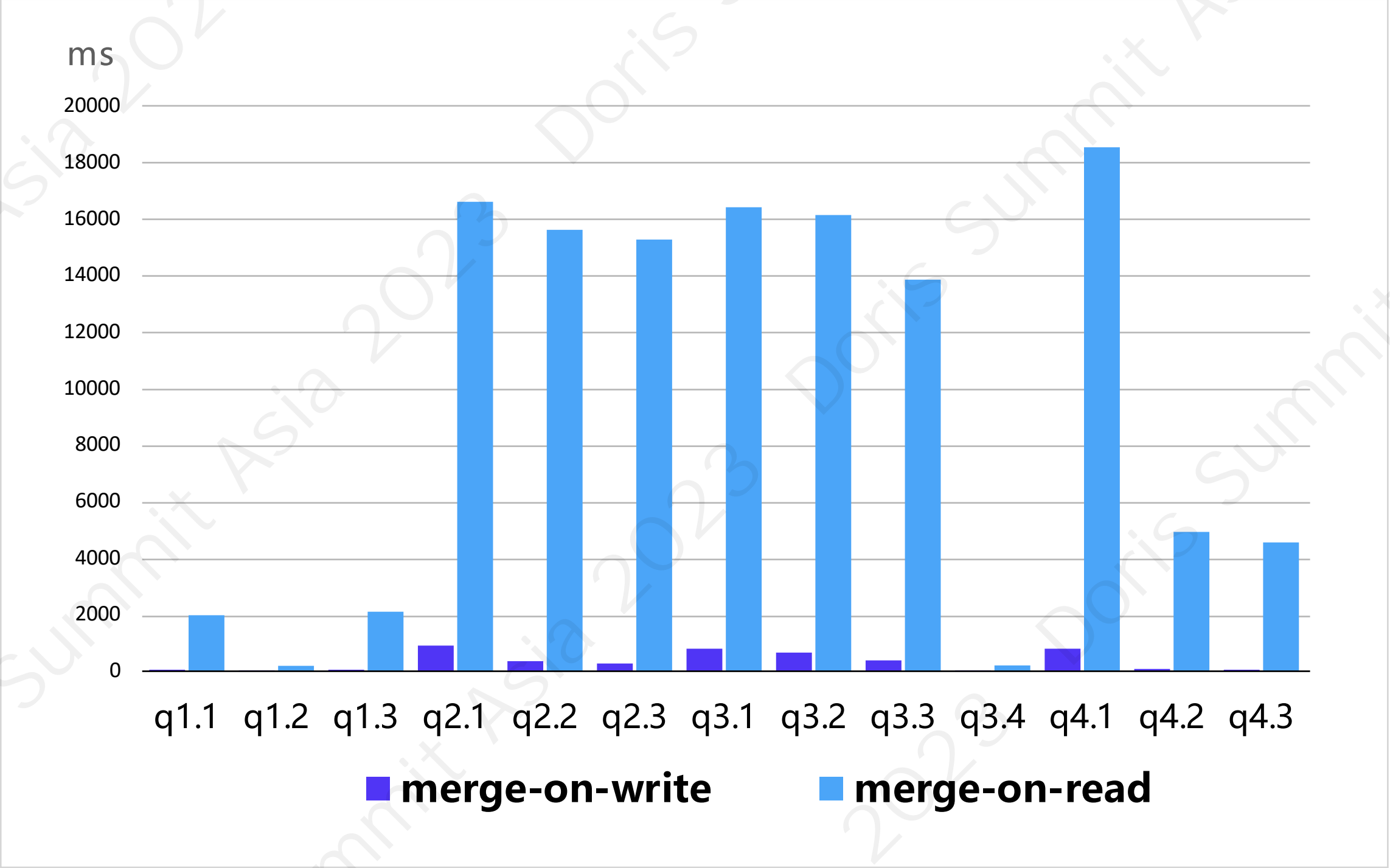
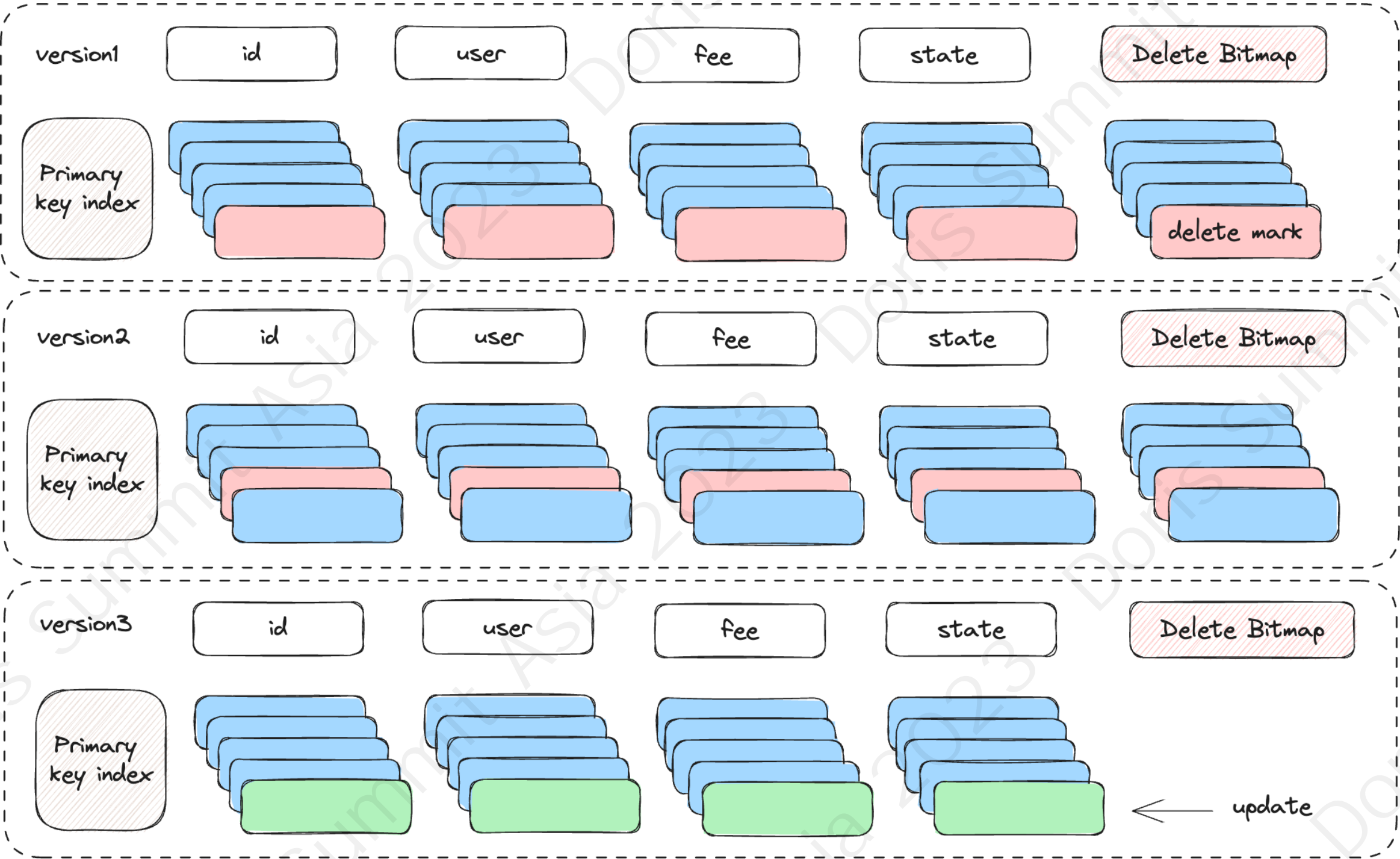
高可用、低成本

实时写入性能提升2-8倍



- 对导入过程中 MemTable 的攒批、排序和落盘等流程进行优化，增加 memtable 并行下刷，提高了上下游之间数据传输的效率；
- 引入“单副本导入”的数据分发模式，多副本数据导入时无需在多个 BE 上重复进行排序，直接复制主副本就行，有效提升集群计算和内存资源的利用率，提升导入的总吞吐量，导入性能提升2-8倍；

主键模型写时合并，实现高效的数据更新



引入写时合并模式，实现小批量数据的实时更新，查询性能提升5-10倍：

- 在写入时通过标记删除做轻量级 Merge，从而提高写入和查询效率；
- 执行 upsert 写入操作，单节点实现 40w 行/s的峰值吞吐；

主键模型写时合并，实现高效的数据更新

对于各种类型的数据更新都有完备的支持



Upsert



条件更新



条件删除



部分列更新



分区覆盖

```
UPDATE test SET v1 = v1 + 1 WHERE k1=1
```

```
UPDATE t1
```

```
  SET t1.c1 = t2.c1, t1.c3 = t2.c3 * 100
```

```
FROM t2 INNER JOIN t3 ON t2.id = t3.id
```

```
WHERE t1.id = t2.id;
```

```
DELETE FROM my_table PARTITIONS (p1, p2)
```

```
  WHERE k1 >= 3 AND k2 = "abc";
```

```
DELETE FROM t1
```

```
  USING t2 INNER JOIN t3 ON t2.id = t3.id
```

```
  WHERE t1.id = t2.id;
```

技术进展回顾

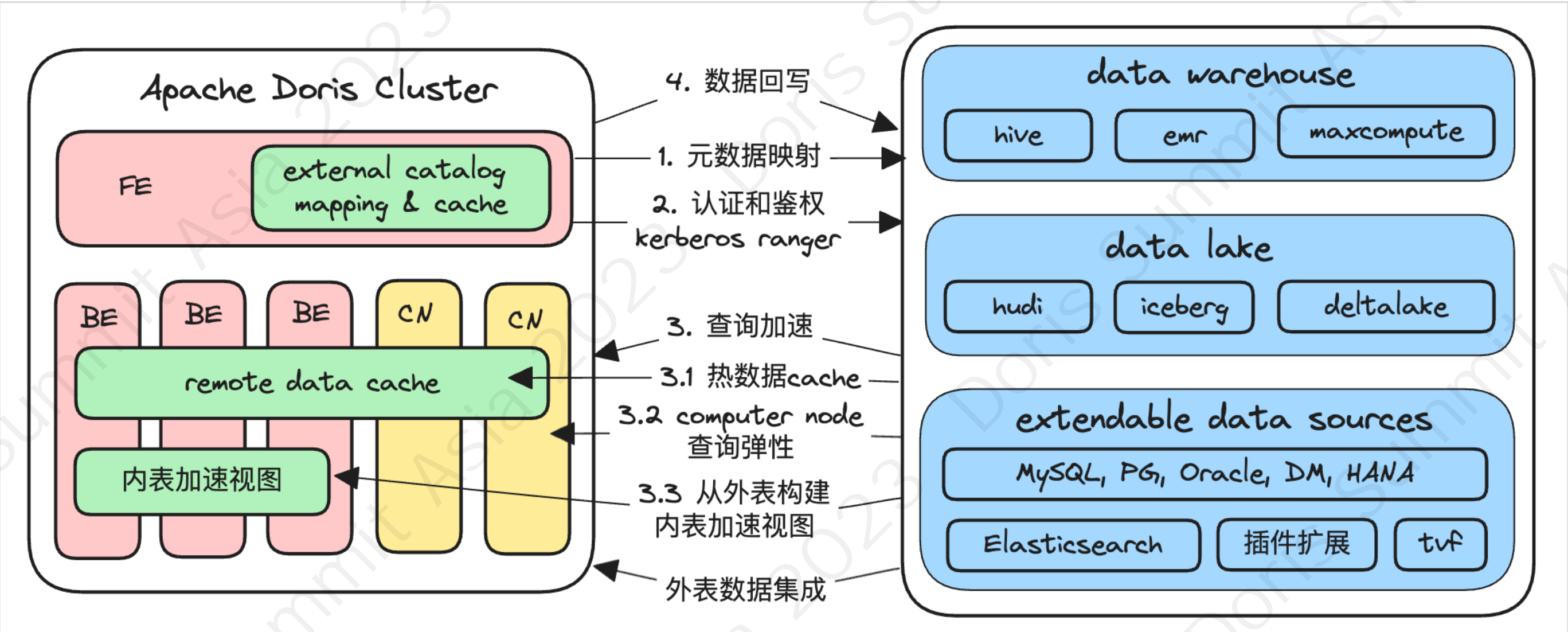
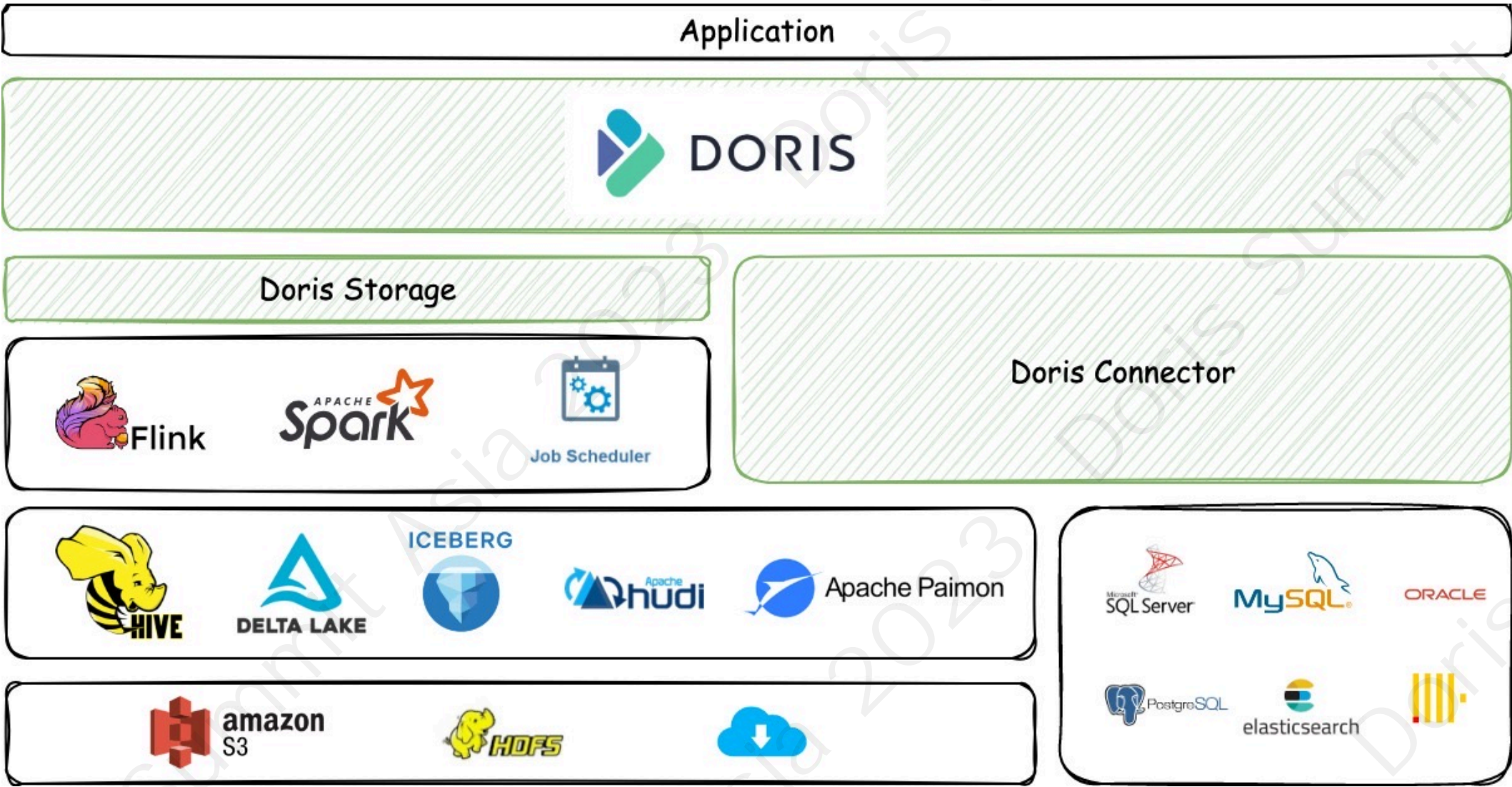
查询性能

实时写入/更新

更多分析场景

高可用、低成本

更加开放的湖仓一体解决方案



可扩展的数据源连接框架和丰富的数据源支持，查询性能较 Trino/Presto 提升 3-10 倍，构建湖仓一体更得心应手：

- 湖仓查询加速：为数据湖、Elasticsearch 以及各类关系型数据库提供优秀的查询加速能力，相比 Hive、Presto、Spark 等查询引擎实现数倍的性能提升。
- 数据导入与集成：基于可扩展的连接框架，增强 Apache Doris 在数据集成方面的能力，让数据更便捷的被消费和处理。用户可以通过 Apache Doris 对上游的多种数据源进行统一的增量、全量同步，并利用 Apache Doris 的数据处理能力对数据进行加工和展示，也可以将加工后的数据写回到数据源，或提供给下游系统进行消费。
- 统一数据分析网关：利用 Apache Doris 构建完善可扩展的数据源连接框架，便于快速接入多类数据源。提供基于各种异构数据源的快速查询和写入能力，将 Apache Doris 打造成统一的数据分析网关。

更高性价比的日志检索分析平台

5倍
写入吞吐提升

- 利用CPU向量化指令，提升数据解析、构建索引的性能
- 简化去掉正排等索引结构，降低构建索引开销

80%
存储成本降低

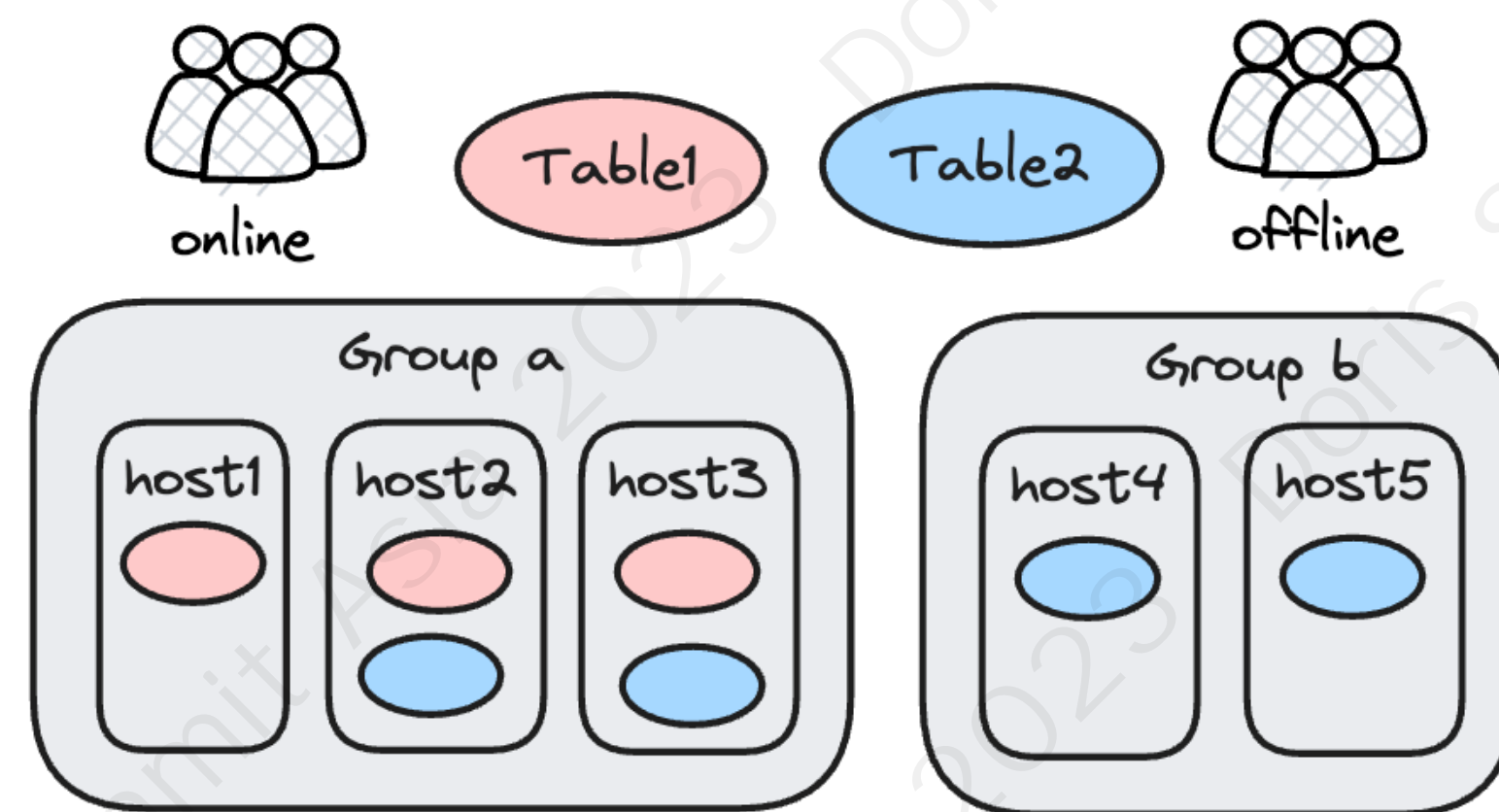
- 简化去掉正排等索引结构，减少倒排索引数据量30%
- 列式存储与ZSTD压缩算法，提供5-10倍压缩比
- 冷热分层，降低冷数据存储成本60%

稳定性提升

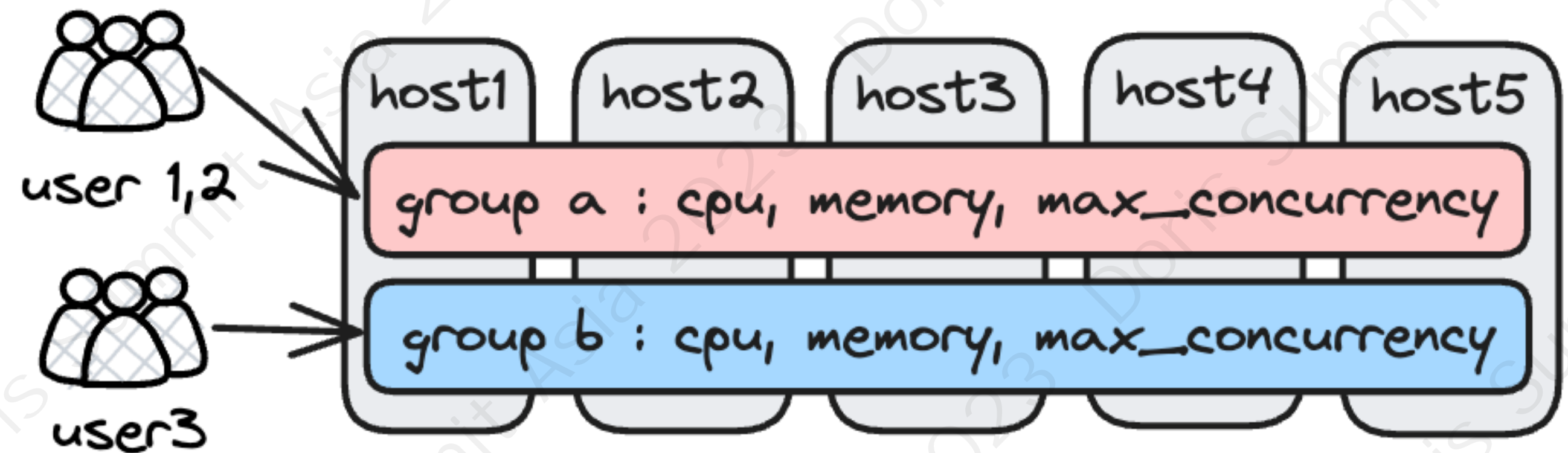
- 基于资源队列的隔离机制，解决负载间相互影响
- 异常查询Kill机制，避免单个查询影响整个集群
- 中间数据落盘，支持大查询内存不足运行失败

指标	Apache Doris	Elasticsearch
写入速度 (MB/s)	~550	~130
存储空间 (GB)	~4	~19
查询耗时 (s)	~95	~230

更精细化的多租户与资源隔离方案



Resource Tag 资源硬隔离



Workload Group 资源软限制

基于 Workload Group 实现了更加精细化的资源隔离方案：

- 细化到进程内的资源隔离，实现更精细化的资源分配和调度，避免进程内的资源冲突和抢占；
- 支持设置资源组优先级与超限使用，保证隔离性的同时实现了资源的充分利用；
- 通过查询队列和任务排队机制进一步保证了在高工作负载场景下的系统稳定性。

技术进展回顾

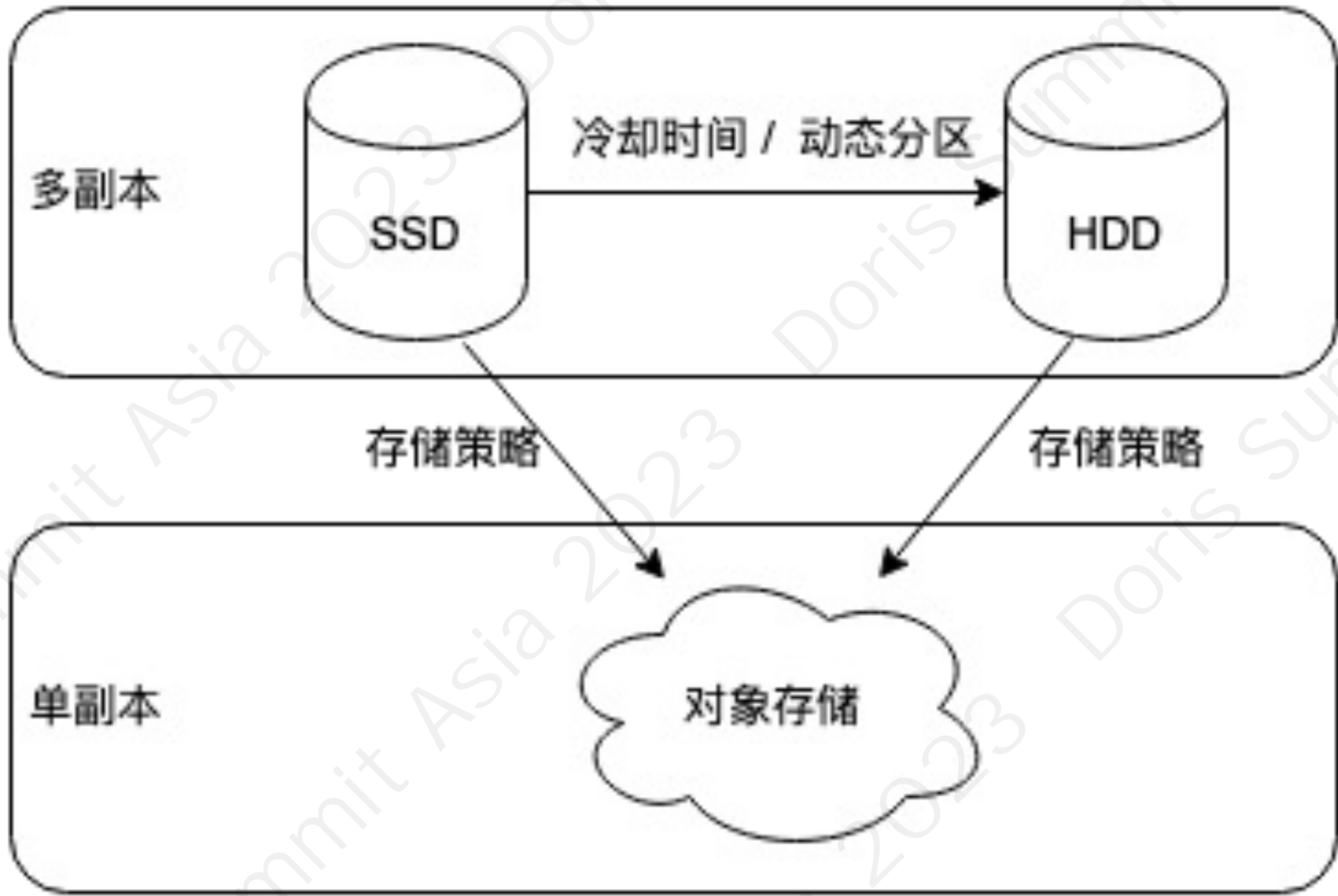
查询性能

实时写入/更新

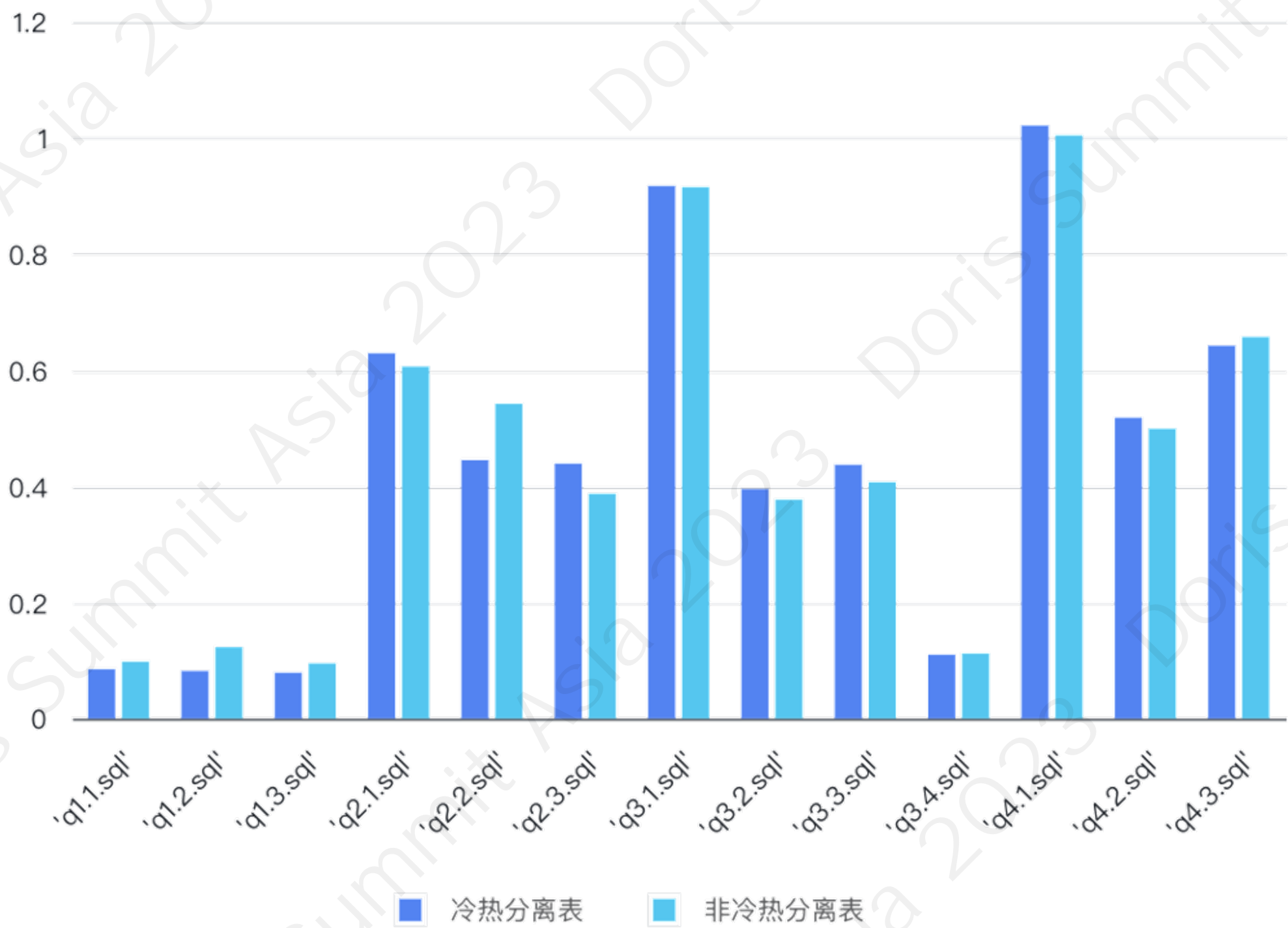
更多分析场景

高可用、低成本

基于对象存储实现冷热数据分层，存储成本降低70%

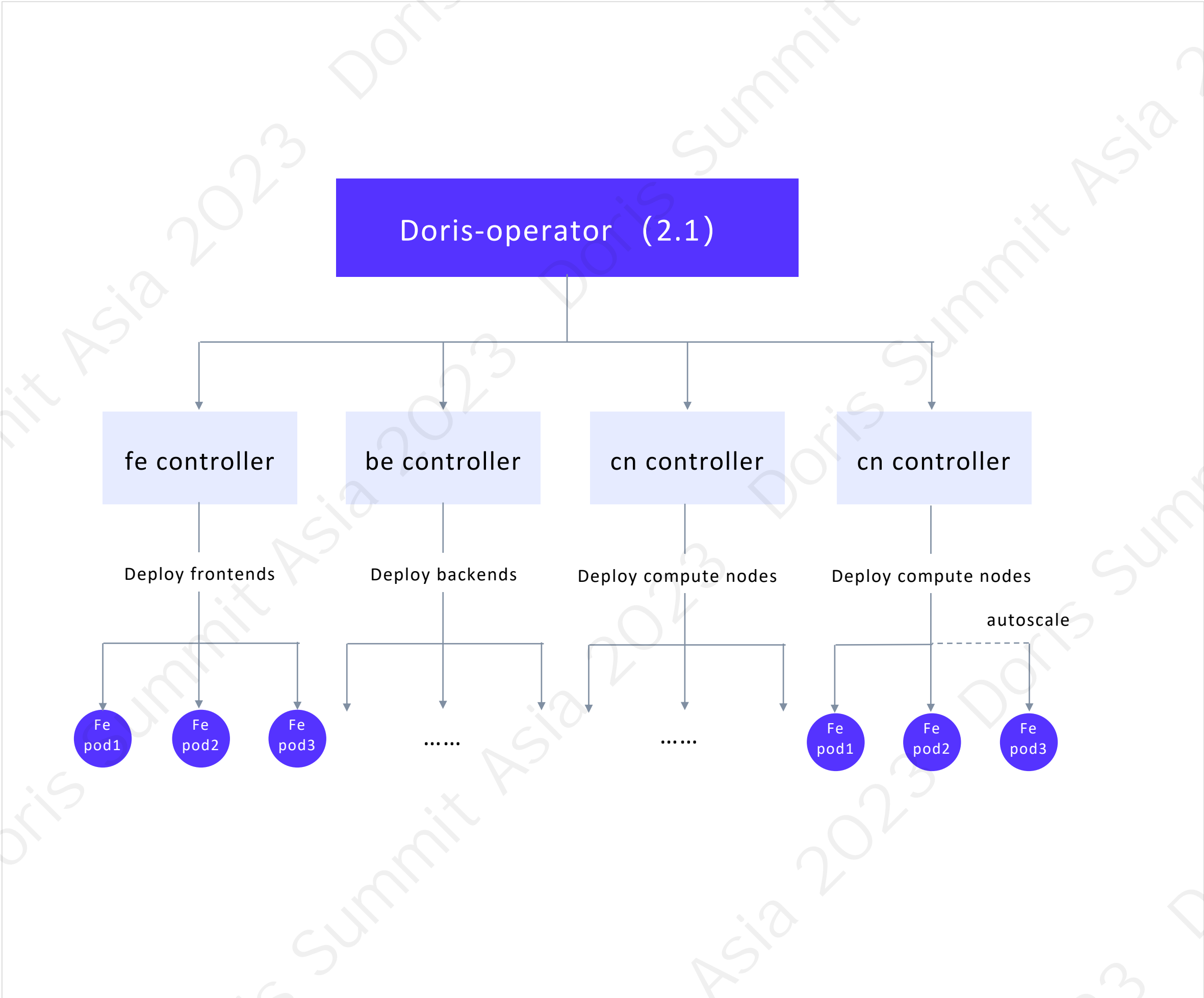


查询性能对比



- 根据将冷热数据分别存储在成本不同的存储介质上，从原本的 SSD->HDD 增加到 SSD -> HDD -> OS 三层；
- 云磁盘的价格通常是对象存储的 5-10 倍，如果可以将 80% 的冷数据保存到对象存储中，**存储成本至少可降低 70%**；
- 通过冷数据 Compaction 实现数据的高效压缩，提供冷数据 Cache 加速冷热数据查询，降低成本的同时保持性能不受影响；

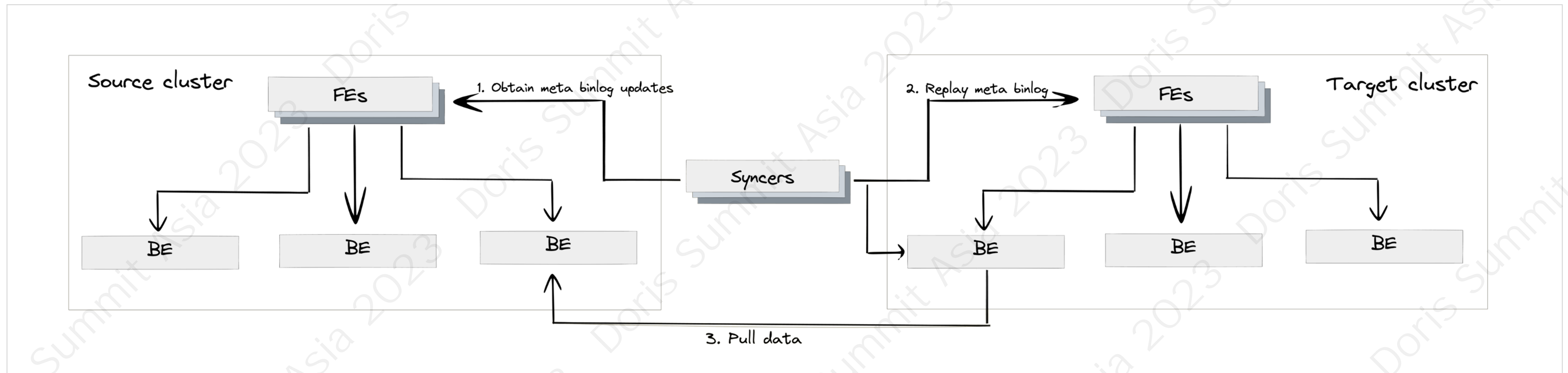
支持部署在公有云/私有云/K8s上



更简易的K8s 部署运维模式

- 支持 FE/BE/Compute Node/Broker 所有组件的部署、扩容、缩容、健康检查等运维工作
- 支持 Compute Node 根据机器负载自动扩容缩容；
- 支持 Prometheus 监控；
- 支持服务滚动升级；
- 支持多种persistent Volume的管理能力；

跨集群数据复制，集群可用性保证的利器



- 容灾备份：将企业的数据备份到另一个集群与机房中，当突发事件导致业务中断或丢失时，可以从备份中恢复数据或快速进行主备切换。一般在对 SLA 要求比较高的场景中，都需要进行容灾备份，比如在金融、医疗、电子商务等领域中比较常见。
- 读写分离：读写分离是将数据的查询操作和写入操作进行分离，目的是降低读写操作的相互影响，保证数据库的性能及稳定性。
- 隔离升级：当对集群升级时，为了避免不兼容和未知Bug，提前构建备集群进行双跑验证。
- 性能数据：最快可以做到分钟级数据延时，数据同步速度可以触及网卡和磁盘上限。

3 走向实时分析的下一步

实时分析


**极致分析
性能**

自适应的查询引擎

全自动统计信息收集

丰富hint语法

TPC-DS 性能

大查询落盘

小查询性能提升

Union 算子并行执行

Join速度提升2倍

数据自适应

多表物化视图

全量/增量构建

透明改写

聚合改写与上卷

智能运维


实时更新

主键模型优化

写时合并/灵活数据更新

写时合并默认开启

建表制定Cluster by

灵活的任意列更新

Key列与排序列分离，任意列排序

基于MOW的数据模型统一

行列混存

更加通用的查询

prefix scan

Batch read

更加智能的Cache机制

行级cache

Block级cache

淘汰策略优化


实时写入

写入语义统一，All is Relation！

Insert into select 统一多种写入方式

MySQL协议

HTTP协议

Kafka Load

S3/HDFS/数据湖

写入更加便捷

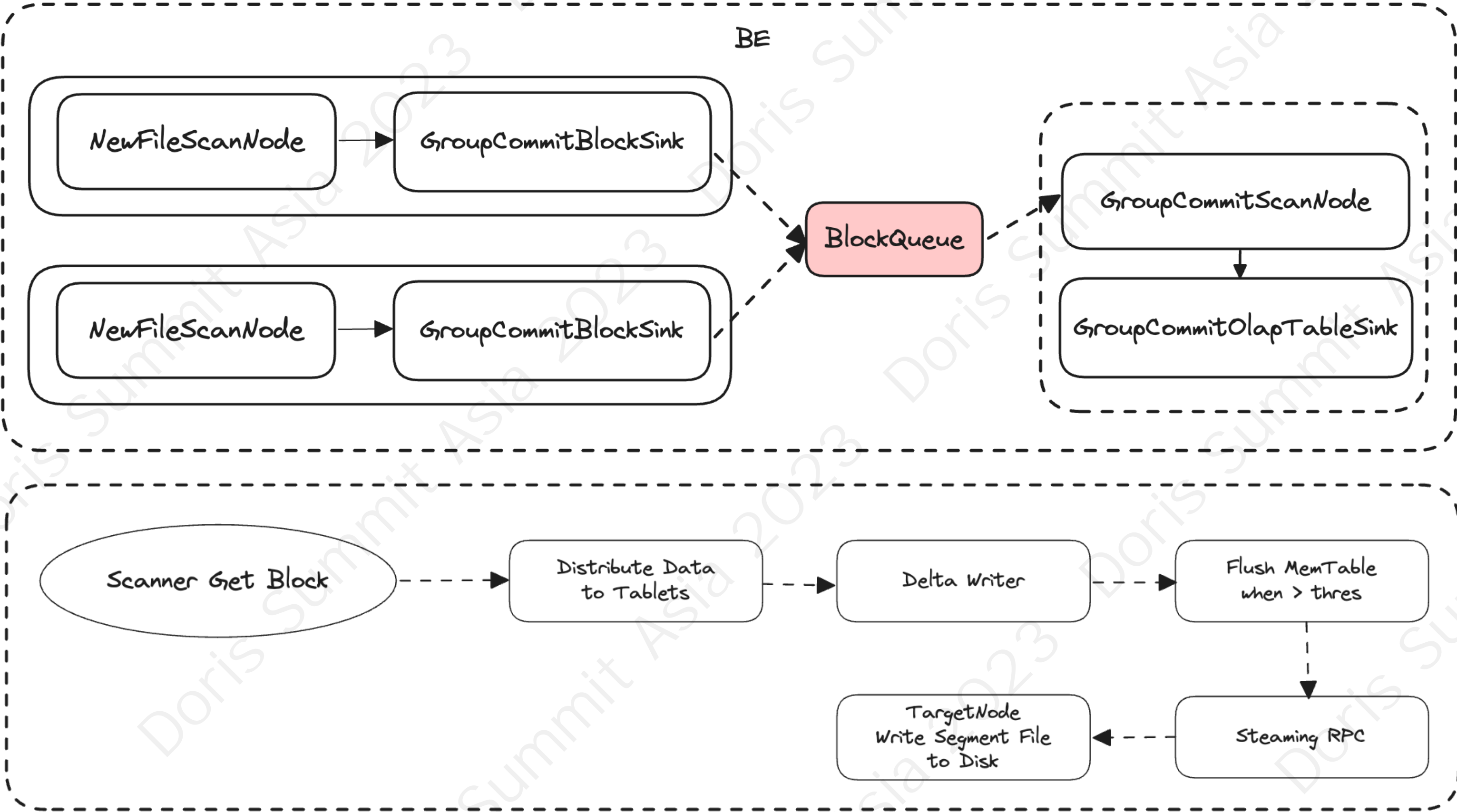
更简易、更稳定、更高性能的实时数据写入

服务端攒批，毫秒级实时

Memtable下刷前置

内置流式TP数据库同步

服务端攒批，高并发写入更快更稳定



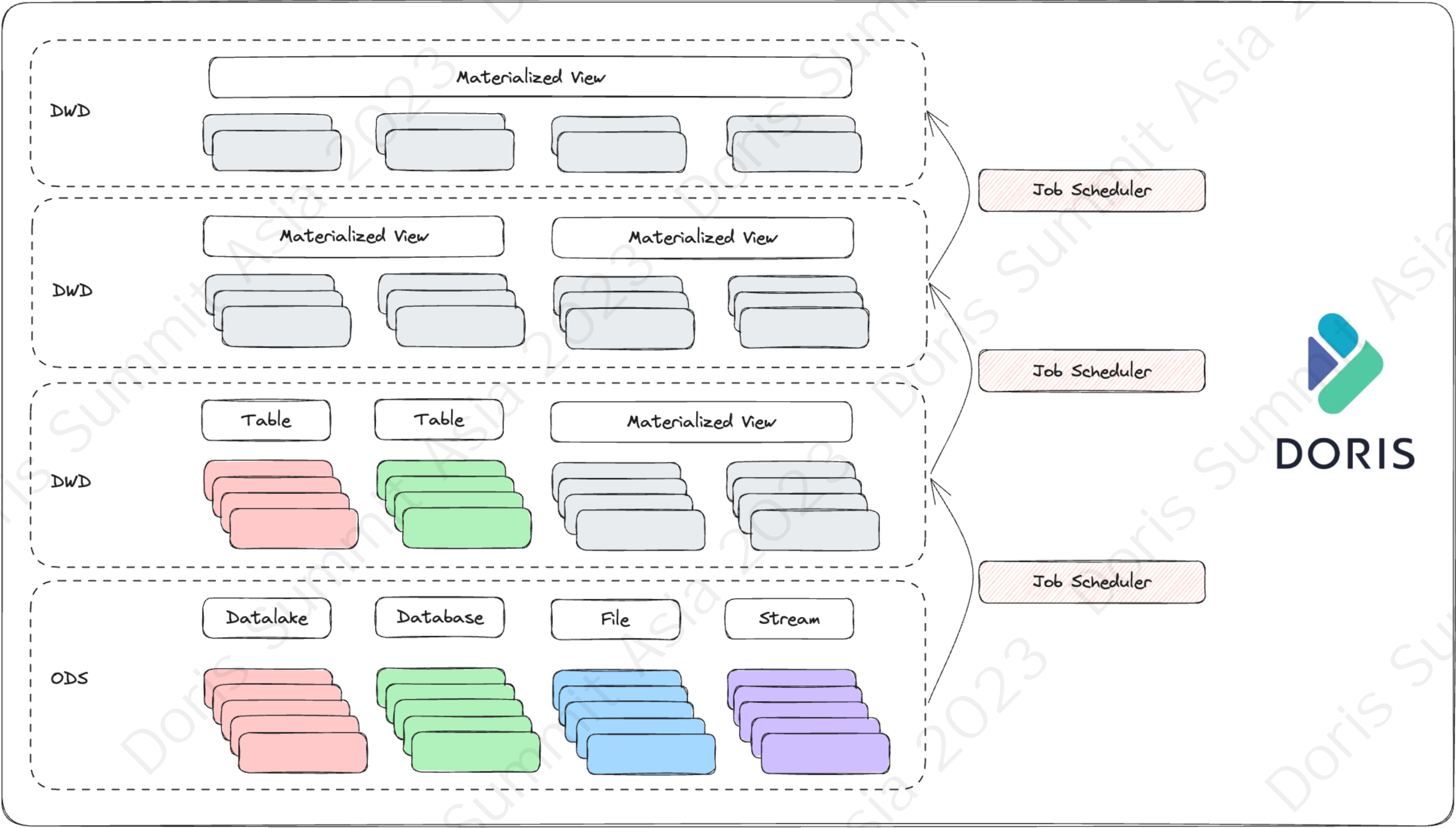
服务端自动攒批，毫秒级高并发写入

- 更稳定，降低高并发写入的小文件问题；
- 更智能，减少写入端攒批带来的维护成本；
- 在8c16g单机上使用服务端攒批，写入吞吐可以达到10w行/s，秒级数据可见；

写入性能进一步提升2倍

引入memtable前移，简化导入路径、节省RPC序列化、压缩、解压开销。

多表物化视图+内置调度，加速查询、简化数据建模与加工逻辑



加速多表关联查询、简化数据建模

- 空间换时间，通过预计算在高频查询或代价昂贵的查询中显著提升查询速度；
- 提供数据仓库分层建模、透明改写能力，减少查询加速的人工成本；

内置任务调度模块，减少外部依赖

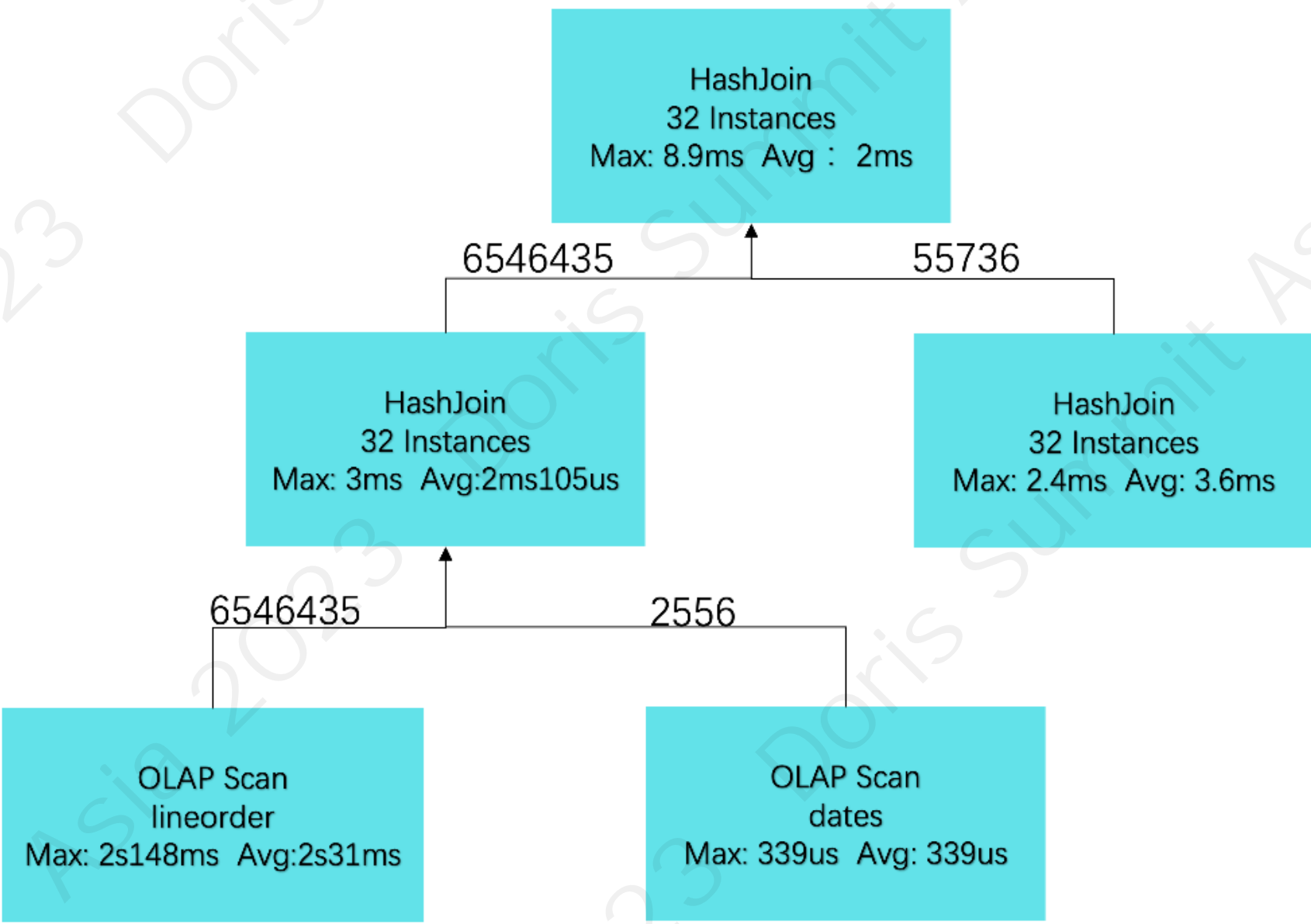
- 轻量级 ETL 任务编排
- 简化数据加工逻辑

Profile全面优化，更易读、可视化展示、动态更新

```
VAGGREGATION_NODE (id=3):(Active: 376.91us, % non-child: 0.00%)
- AggInfos:: (_is_merge: true, _needs_finalize: true, Streamin
- BuildConvertToPartitionedTime: 0ns
- BuildTime: 0ns
- DeserializeAndMergeTime: 1.121us
- ExecTime: 0ns
- ExprTime: 0ns
- GetResultsTime: 139.940us
- HashTableComputeTime: 22.787us
- HashTableInputCount: 1
- HashTableIterateTime: 1.448us
- HashTableSize: 1
- InsertKeysToColumnTime: 40.777us
- MaxRowSizeInBytes: 77
- MemoryUsage:
  - HashTable: 24.00 B
  - PeakMemoryUsage: 1.39 MB
  - SerializeKeyArena: 1.39 MB
- MergeTime: 49.752us
- ProjectionTime: 0ns
- RowsReturned: 1
- RowsReturnedRate: 2.658K /sec
- SerializeDataTime: 0ns
- SerializeKeyTime: 7.425us
- SerializeResultTime: 0ns
- StreamingAggTime: 0ns
```



```
1051:VHASH JOIN
|
| join op: INNER JOIN(BUCKET_SHUFFLE)[]
| equal join conjunct: id = expr_cast(id as BIGINT)
| runtime filters: RF001[in_or_bloom] <- expr_cast(id as BIGINT)(1/1/1048576)
| cardinality=4987134
|
| vec output tuple id: 7
| vIntermediate tuple ids: 6
| hash output slot ids: 8
| BlocksReturned(Operator): 0
| CloseTime(Operator): avg 9.451us, max 9.451us, min 9.451us
|
| Sink:
|   InputRows(Sink): 1078423
|   OpenTime(Sink): avg 212.114us, max 212.114us, min 212.114us
|   TotalTime(Sink): avg 50.175ms, max 100.175ms, min 1.175ms
|
| Operator:
|   OpenTime(Operator): avg 79.514us, max 79.514us, min 79.514us
|   RowsReturned(Operator): 4781291
|   TotalTime(Operator): avg 40.75ms, max 80.52ms, min 1.001ms
```



- Profile 很大(几MB~几十MB)，几百上千行，FE 内存占用太多；
- 跟查询执行逻辑强绑定，用户理解成本过高，不便分析性能瓶颈；

- 算子的详细信息
- 合并的 Instance 的统计信息
- 更加贴近用户的指标，OutputRows, TotalTime

- 可视化的展示方式，更容易发现瓶颈
- 根据执行过程动态更新

融合统一



数据湖分析

多表mv+内置调度，简化数据建模与加工逻辑

- 异步构建mv，实现分层建模/透明改写/查询加速
- 轻量级ETL作业编排与数据加工

数据导出与写回，实现数据加工闭环

Parquet/ORC文件

RDBMS

Hive/Iceberg/Hudi/Paimon

倒排索引优化

支持更多数据类型

array

map

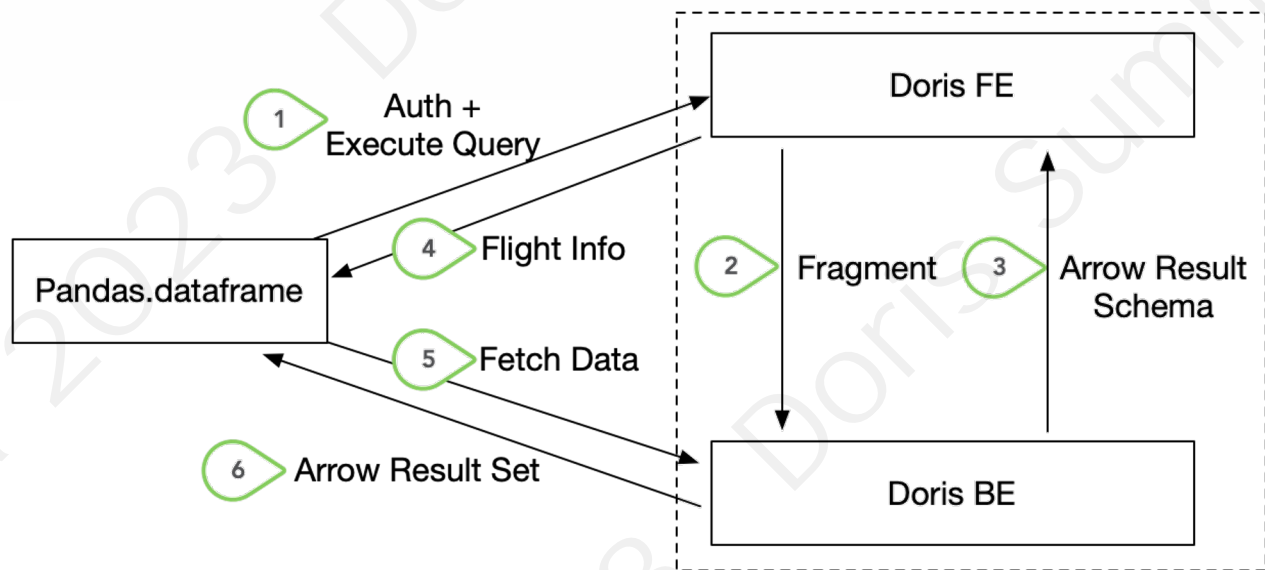
variant

geo

更加通用查询加速

- Count、Join等加速
- 自适应选择索引

高速数据读取



- 数据直接从BE 传递到Pandas 客户端
- 列式数据传输，数据吞吐性能提升100倍！

Variant类型

更加彻底的Schemafree

任意嵌套JSON

子字段类型变化

嵌套结构变化

自动拆分子列

支持倒排索引

灵活的混合负载管理

- 通过SQL 来创建和管理Workload Group，调整资源配置；一个集群可以建立多个 Workload Group；可实现查询并发控制、排队，计算资源软限/硬限管理

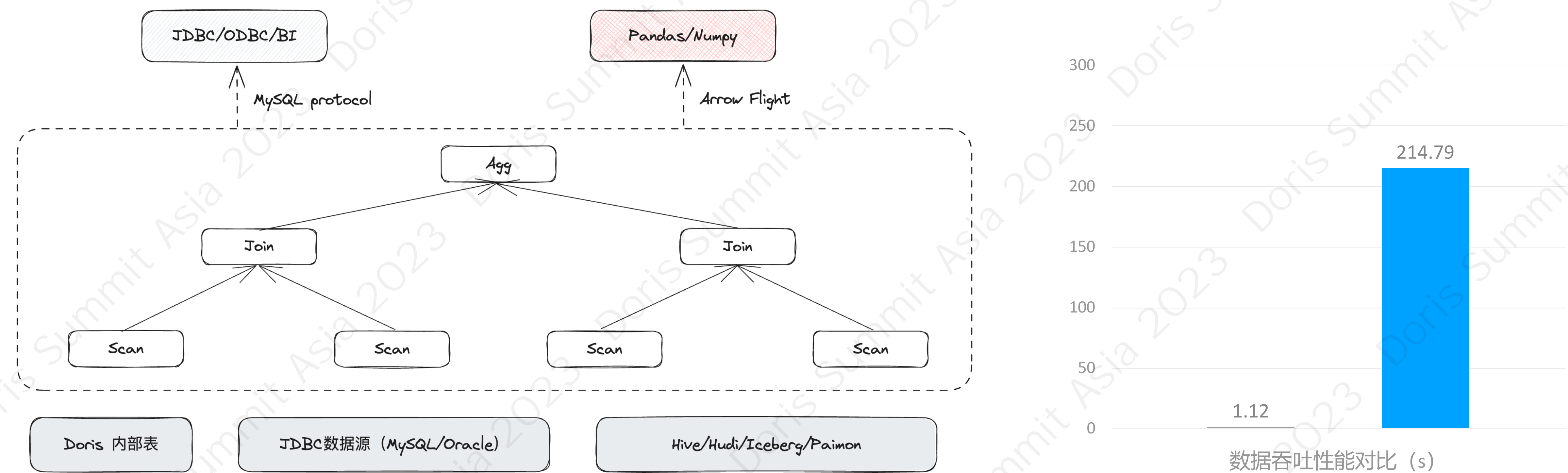


日志分析与检索



负载管理

高速数据读取，数据吞吐提高100倍



- MySQL 具有良好兼容性和广泛的工具支持，但在数据科学、大规模数据导出场景，FE 容易成为瓶颈、文本协议效率差；
- 引入 Arrow Flight 实现高速数据读取、数据直接通过 BE 传递到 Pandas 客户端，列式数据传输；
- Pandas 测试数据吞吐提升 100 倍！

统一的湖仓分析和数据集成



- 湖仓加速：高速的湖上数据查询加速；
- 统一数据分析网关：各类异构数据源的查询和写入能力；
- 统一数据集成：多数据源的数据同步、加工处理、数据导出；

Schema less 的 Variant 类型

```
CREATE TABLE IF NOT EXISTS ${table_name} (  
    k bigint,  
    v variant  
)  
DUPLICATE KEY(`k`)  
DISTRIBUTED BY RANDOM BUCKETS 5  
properties("replication_num" = "1");  
  
// insert into 写入或是stream load  
insert into ${table_name} values(1, '{"a" : 1}');  
  
// 查询需要带上cast, 存储层尽可能消除cast加速查询  
select cast(v:a as int) from ${table_name} where cast(v:a as int) > 1
```

Schema Less

- 可以支持任意类型、任意形状的json格式文档数据
- 自动动态地处理列增加、类型变更
- 不需要繁琐的DDL操作以及Schema Change操作

高性能

- 根据数据类型自动推断类型进行列式存储
- 与普通字段一样的查询效率

云原生化，存储计算分离，实现极致弹性



多计算集群

元数据与数据共享，计算负载隔离

共享存储与本地缓存

共享存储系统，热数据本地手动/自动缓存

计算弹性扩缩容

手动/自动扩缩容，集群自动启停

- 在 2.1 版本中完成代码结构调整，在 2.2 版本正式面向社区可用；

与创新者同行





获取更多社区动态与最佳实践

Apache Doris 官方平台:

- Apache Doris 官网: doris.apache.org
- Apache Doris GitHub: github.com/apache/doris/

获取更多峰会资料:

- Doris Summit 峰会官网: doris-summit.org.cn
- Doris Summit 峰会回放: <https://space.bilibili.com/1196172099/channel/collectiondetail?sid=1824324>