

基于 Apache Doris 构建多场景极速分析体验

王天宜

飞轮科技 资深解决方案架构师

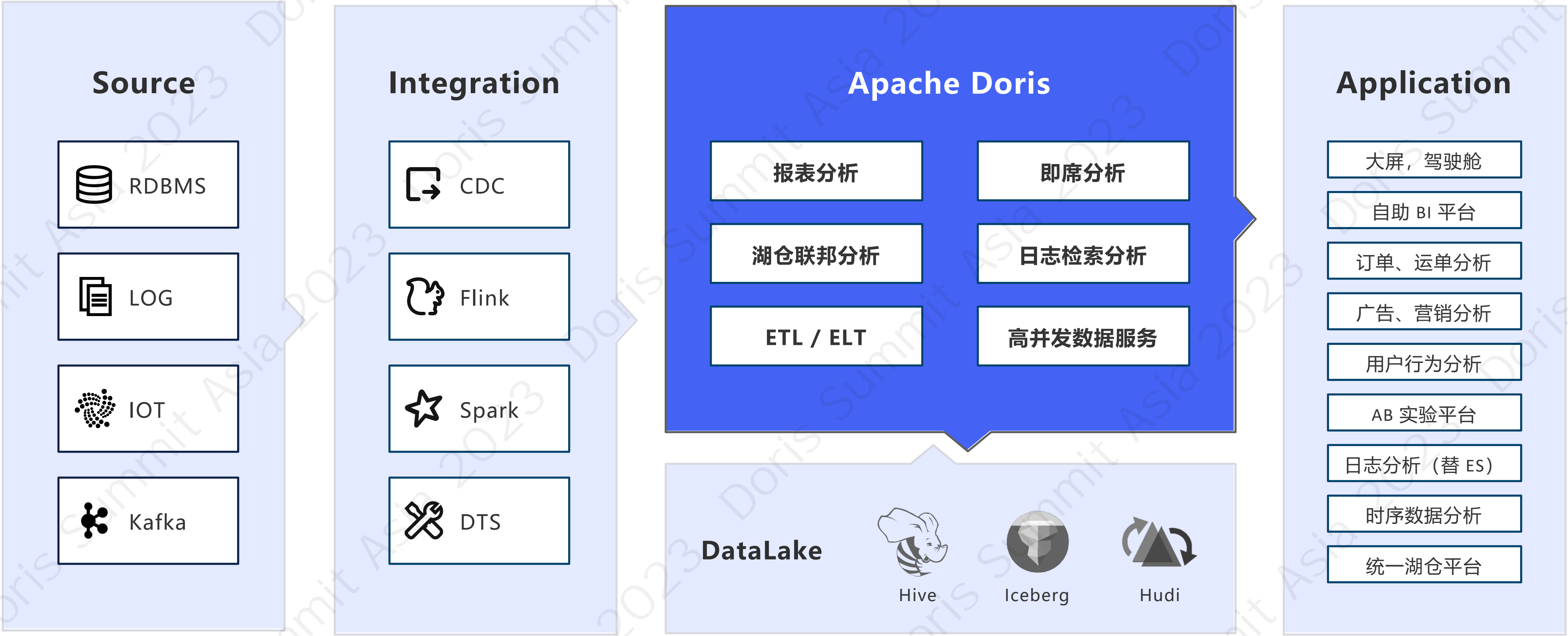
目录

1. Apache Doris 在大数据生态中的定位
2. Apache Doris 核心能力与特点
3. Apache Doris 多场景实时分析方案

1

Apache Doris 在大数据生态中的定位

Apache Doris 在数据分析中的定位



Apache Doris 主要特性

高效

- 极速的分析性能
- 高效的数据更新能力
- 丰富的数据集成能力
- 极致弹性与存算分离

简单

- 高可用与高可靠
- 多租户管理能力
- 可视化管理工具
- 丰富周边工具

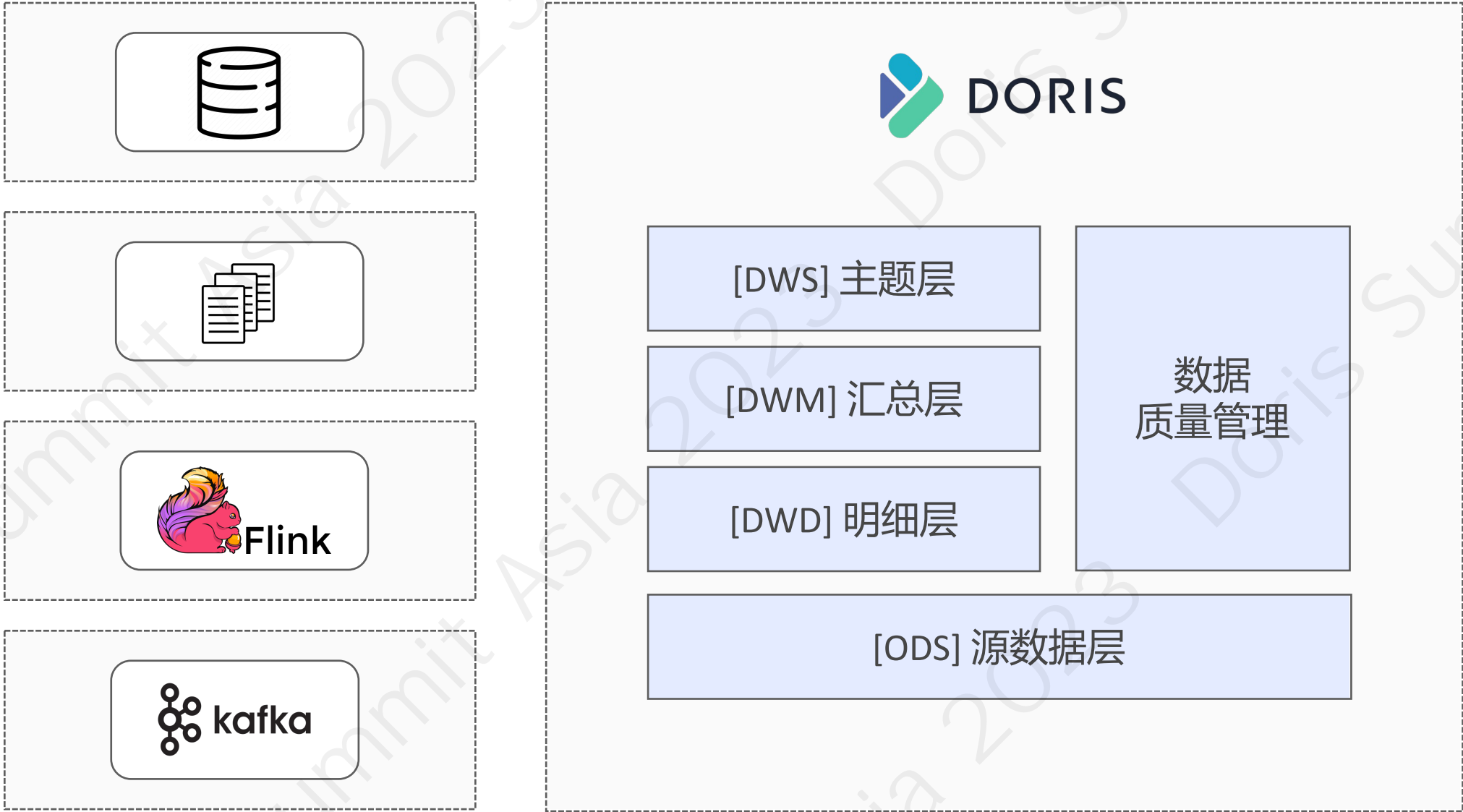
统一

- 结构化与半结构化统一
- 在线与离线数据统一
- 湖仓统一
- 多元数据建模统一

2

Apache Doris 实时AdHoc分析场景

实时报表痛点问题



数据入库及更新问题

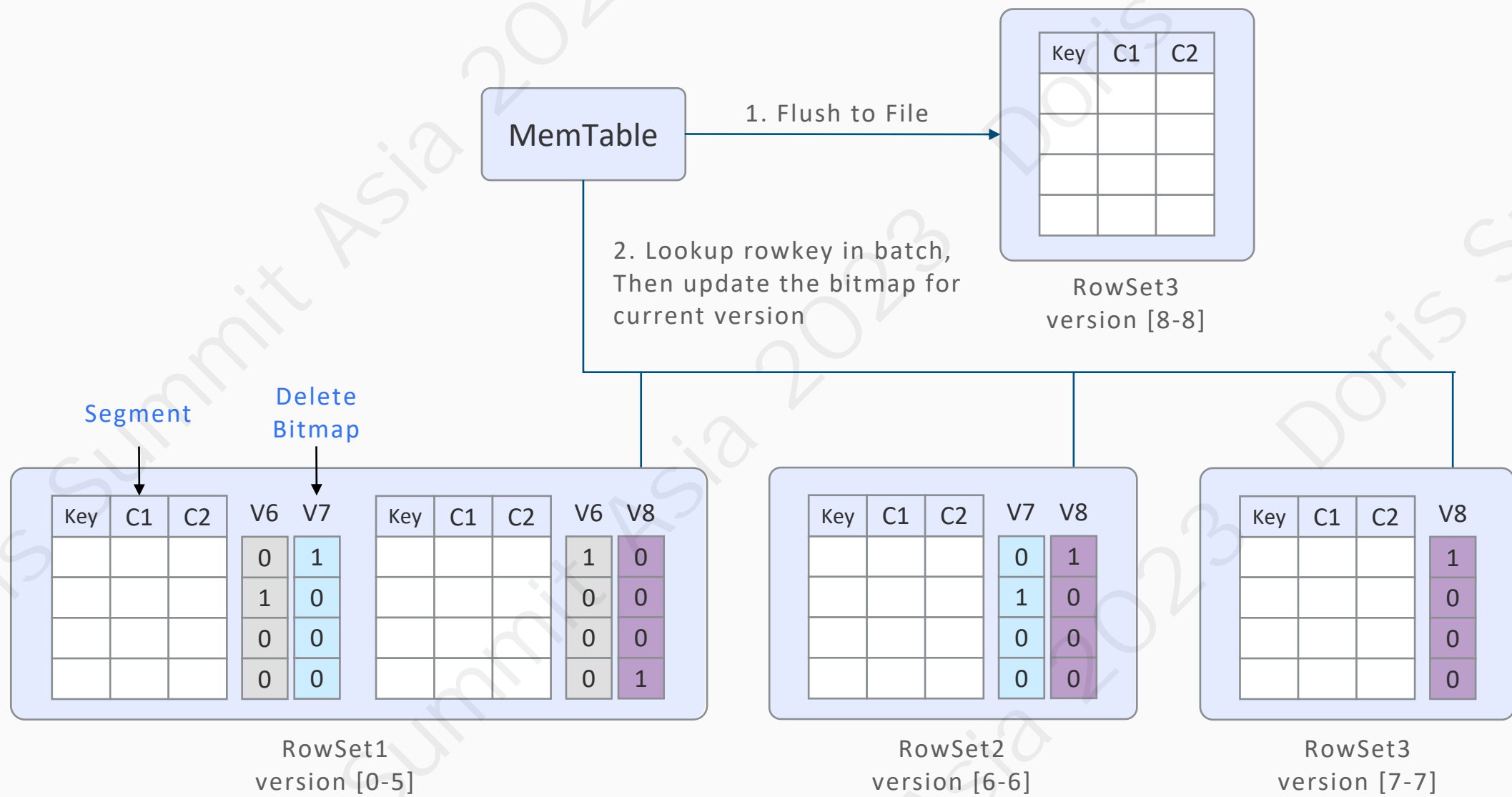
数据导入时效性要求高，秒级别数据可见
基于上游 TP 库的主键需要实时更新
数据需要不丢不重，重要数据原子性导入

数据查询时效性问题

使用人数众多，并发量告
查询延迟要求高，延迟控制在秒级别
基于场景不同，要求多样的建模方式

Apache Doris 高性能更新能力

Merge on Write 模型



Merge on Write 更新能力

实现原理:

主键索引 + Delete Bitmap 实现
导入过程成数据删除标记 delete bitmap
基于 mow 模型实现查询谓词下推

适用场景:

适用于小批量实时高频导入，基于主键做高频数据更新
支持 UPSERT、条件更新、条件删除、部分列更新、分区覆盖

TPCH 标准测试集，使用 MOW 模型，性能提升近 50 倍

100 并发	无 Cache	有 Cache
MoR	23.3	14.3
MoW	5.6	0.3
提升倍数	4.2x	47.6x

Apache Doris 高性能更新能力



游戏行为分析场景

存量数据 300 亿，单副本 80TB、70字段
15 并发 Upsert, TPS 40w/s



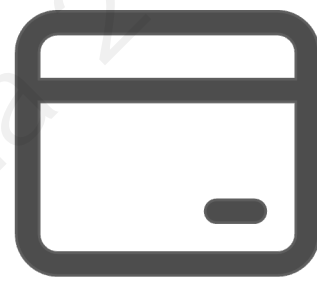
游戏行为分析场景

半年物流订单分析，200 字段宽表
8 并发 Upsert, TPS 6w/s



游戏行为分析场景

千亿数据月度统计，万亿数据年度统计
10 并发 Upsert, TPS 10w/s



消费金融场景

宽表更新场景，基于 MOW 的部分列更新
200 列，20 并发，每个并发更新 10 列
数据可见性明显降低，30s -> 10s



某客户 PoC 压力测试

40 并发 Upsert, 10s Checkpoint
MOW 表稳定导数, TPS 7.5w/s

Apache Doris 数据建模能力

AGGREGATE KEY

报表统计、指标计算

```
CREATE TABLE doris_agg_tab(  
  store_id INT,  
  date INT,  
  amount BIGINT SUM DEFAULT '0',  
) AGGREGATE KEY(store_id, date)  
DISTRIBUTED BY HASH(store_id) BUCKETS 32
```

原表

store_id	date	amount
1	02-14	200
4	02-15	4000

新增

store_id	date	amount
1	02-14	2000
2	02-15	1500

结果

store_id	date	amount
1	02-14	2200
2	02-15	1500
4	02-15	4000

UNIQUE KEY

订单状态、用户状态

```
CREATE TABLE doris_uni_tab(  
  order_id INT,  
  date INT,  
  status VARCHAR(20) SUM DEFAULT '0',  
) UNIQUE KEY(order_id)  
DISTRIBUTED BY HASH(order_id) BUCKETS 32
```

原表

order_id	date	status
1	02-14	待支付
2	02-15	完成支付

新增

order_id	date	status
1	02-14	完成支付

结果

order_id	date	status
1	02-14	完成支付
2	02-15	完成支付

DUPLICATE KEY

明细数据、行为数据

```
CREATE TABLE doris_dup_tab(  
  user_id INT,  
  date INT,  
  action VARCHAR(60) SUM DEFAULT '0',  
) UNIQUE KEY(order_id)  
DISTRIBUTED BY HASH(order_id) BUCKETS 32
```

原表

user_id	date	action
1	02-14	浏览
2	02-15	收藏

新增

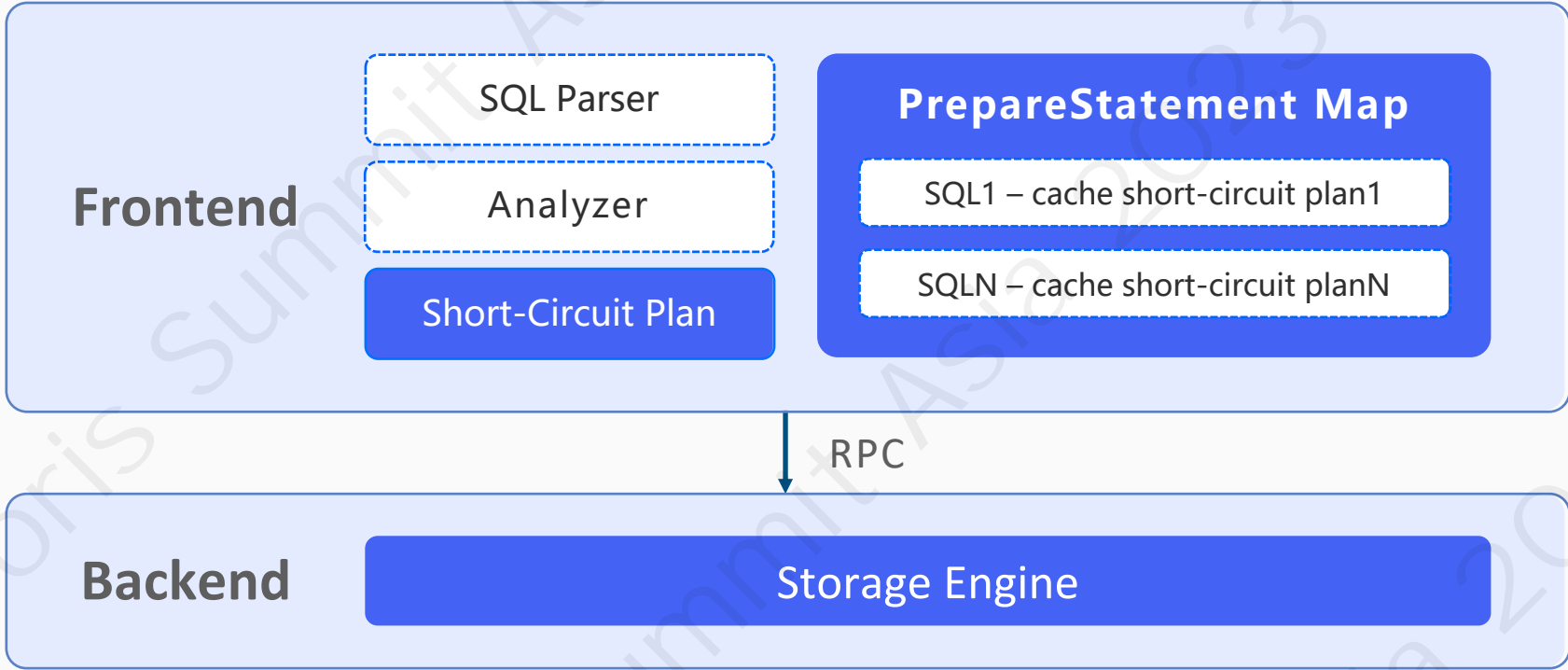
user_id	date	action
1	02-16	购买

结果

user_id	date	action
1	02-14	浏览
2	02-15	收藏
1	02-16	购买

Apache Doris 极速分析能力

date	store	prod	cust	行存
1001	201	X01	CUA	{"date":"1001","store":"201","prod":"X01"}
1002	202	X02	CUB	{"date":"1002","store":"202","prod":"X02"}
1003	203	X03	CUC	{"date":"1003","store":"203","prod":"X03"}
1004	204	X04	CUD	{"date":"1004","store":"204","prod":"X04"}
1005	205	X05	CUE	{"date":"1005","store":"205","prod":"X05"}
1006	206	X06	CUF	{"date":"1006","store":"206","prod":"X06"}



高并发数据服务

高并发查询痛点：

以点查 `SELECT * FROM user_table WHERE id = xxx` 为例
宽表带来 IOPS 放大的问题
执行引擎和优化器对于点查操作过重
SQL 解析规划由 FE 负责，高并发易形成瓶颈

Apache Doris 面向高并发数据服务优化方案：

引入行列混存，解决 IOPS 瓶颈
引入 PreparedStatement，解决解析瓶颈规划
点查短路径优化，减少框架开销

某手机厂商实时行报表查询场景，使用倒排索引，性能提升 45 倍

100 并发	V1.1.5	V1.2.4	V2.0.0(倒排)
SQL1 平均执行时间	0.130	0.140	0.023
SQL1 平均执行时间	1.178	0.397	0.048
SQL1 平均执行时间	0.653	0.284	0.028
SQL1 平均执行时间	4.601	0.972	0.055
SUM(单位ms)	6.562	1.793	0.154

Apache Doris 解决方案收益

高并发查询	极致查询性能	多维度分析	不丢不重
分区分桶剪裁 行存储加速	向量化 MPP 查询 Pipeline 执行引擎	多表物化视图 基于 AGG 模型的预聚合	事务性导入 2PC 数据多副本存储

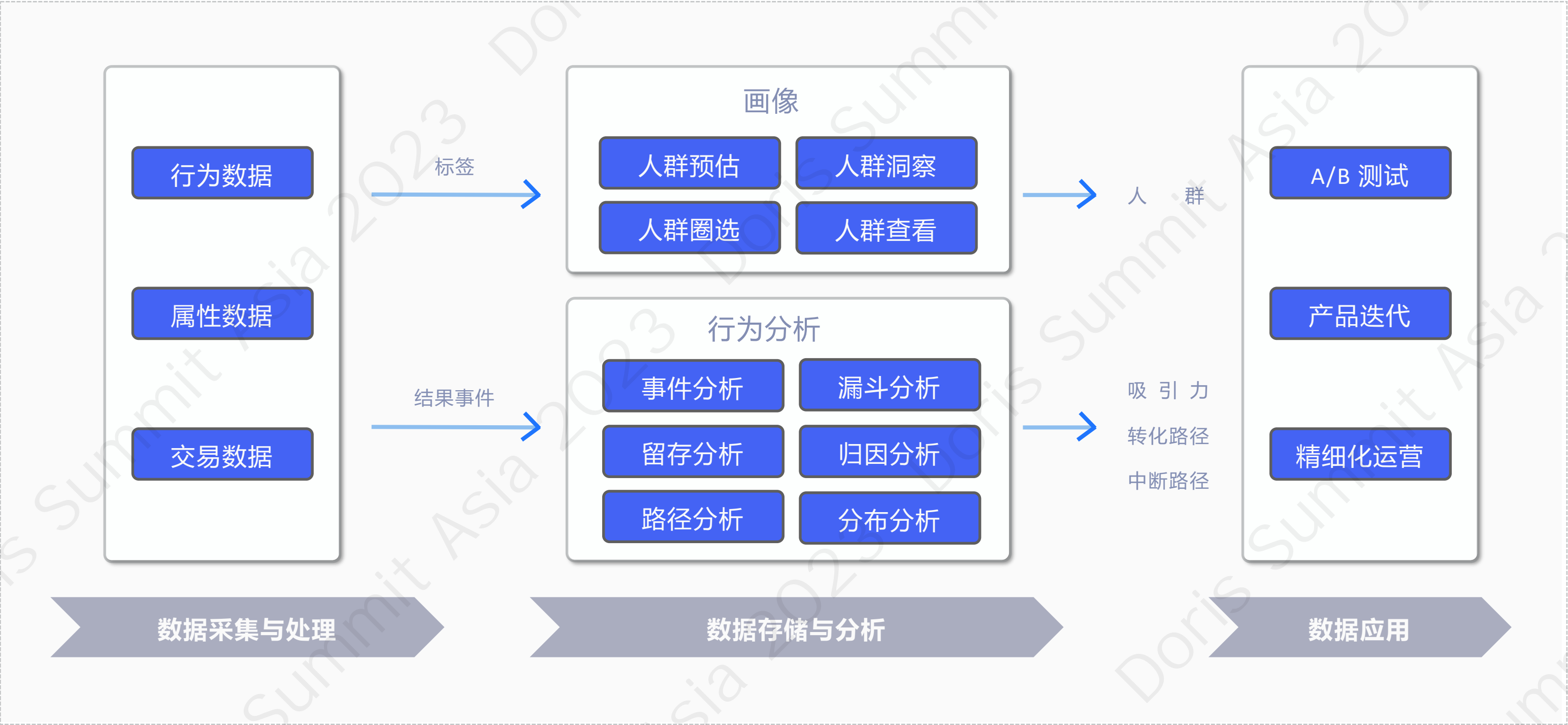
某头部短视频公司面向广告主在线报表平台，2000 台高配物理机，并发 QPS 数万每秒。

3

Apache Doris

用户行为分析及人群圈选

用户行为分析业务特点



画像业务特点：

标签生成时间不统一，有实时更新需求
快速圈选、生成人群包及 TGI 计算

行为分析业务特点：

业务属性频繁变动
SQL 复杂，效率低
数据量大，交互式分析，要求秒级延迟

画像和行为分析 Apache Doris 解决方案



丰富的行为分析函数

典型举例

window_funnel
retention
sequence_match
sequence_count
array函数

不使用分析函数

```
SELECT
  a.day as day,
  count(distinct if(a.day = b.day, a.user_id, NULL)) as active_user_num,
  count(distinct if(a.day = days_add(b.day, 1), b.user_id, NULL)) as active_user_num_1day,
  count(distinct if(a.day = days_add(b.day, 3), b.user_id, NULL)) as active_user_num_3day,
  count(distinct if(a.day = days_add(b.day, 7), b.user_id, NULL)) as active_user_num_7day
FROM
(
  SELECT day, user_id
  FROM login_event
  WHERE (day <= days_add(to_date('2022-11-01'), 7)) AND (day >= '2022-11-01')
) AS a
INNER JOIN
(
  SELECT DISTINCT user_id, day
  FROM login_event
  WHERE day = '2022-11-01'
) AS b ON a.user_id = b.user_id
GROUP BY a.day
```

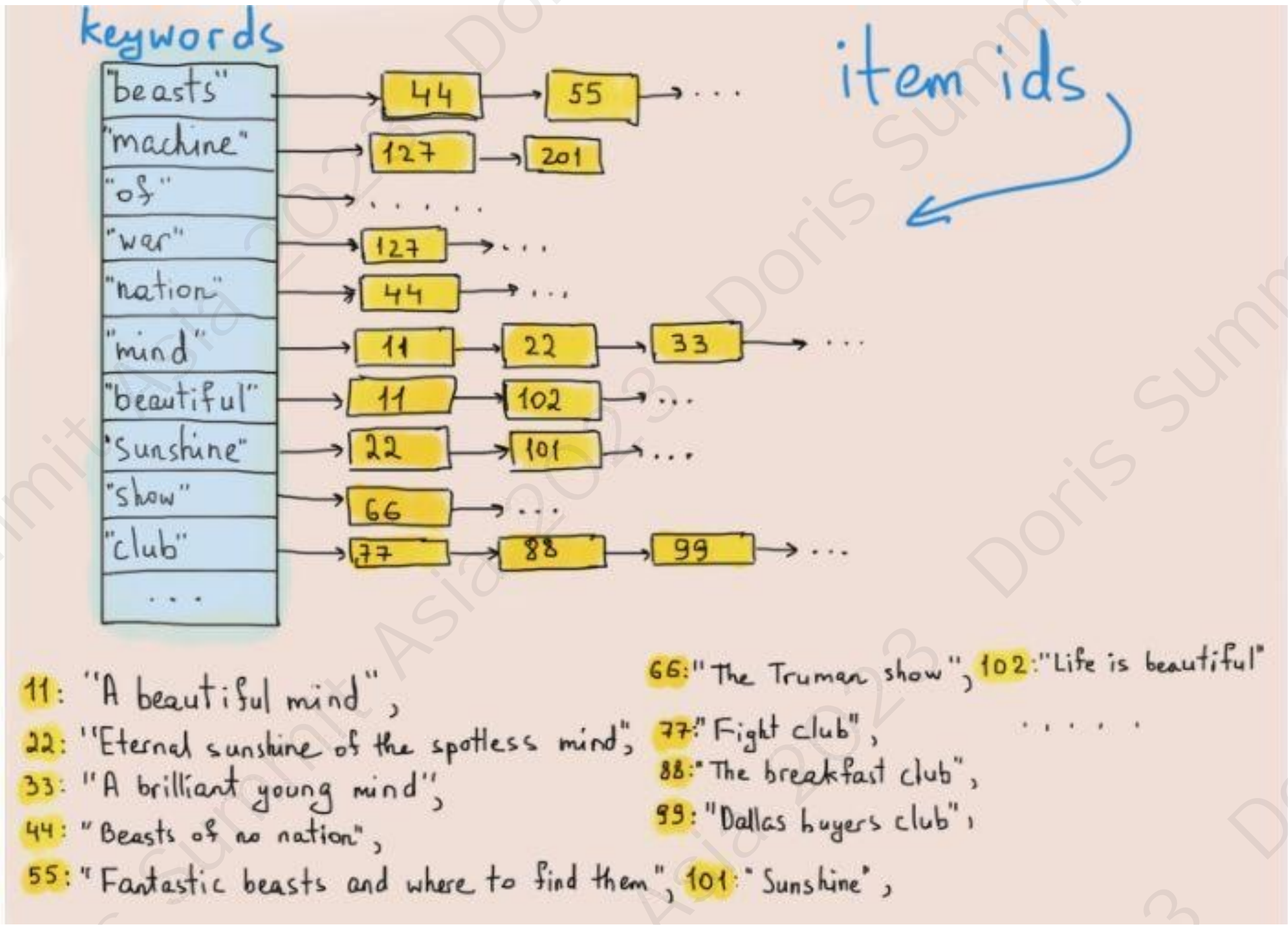


使用分析函数

```
SELECT
  user_id,
  retention(
    day = '2022-11-01',
    day = '2022-11-02',
    day = '2022-11-04',
    day = '2022-11-07') as r
FROM login_event
WHERE (day >= '2022-11-01') AND (day <= '2022-11-21')
GROUP BY user_id
) a
```



Apache Doris 极速分析能力



倒排索引

倒排索引实现多维度快速检索
支持自定义分词，实现全文检索，加速日志场景

```
CREATE INDEX idx_request ON httplogs(request) USING INVERTED;
-- search for request contains word 'login'
SELECT * FROM httplogs WHERE request MATCH 'login';
-- search for request contains word 'error' and 'error'
SELECT * FROM httplogs WHERE request MATCH_ALL 'login error';
```

100 并发	V1.1.5	V1.2.3	V2.0.0(倒排)
SQL1 平均执行时间	1755	1990	6011
SQL1 平均执行时间	218	610	763
SQL1 平均执行时间	687	1582	1127
SQL1 平均执行时间	661	1181	977
SUM(单位ms)	19007	43667	40862

Apache Doris 部分列更新能力

uid	name	age	tag1	tag2	tagN

主键 活跃标签

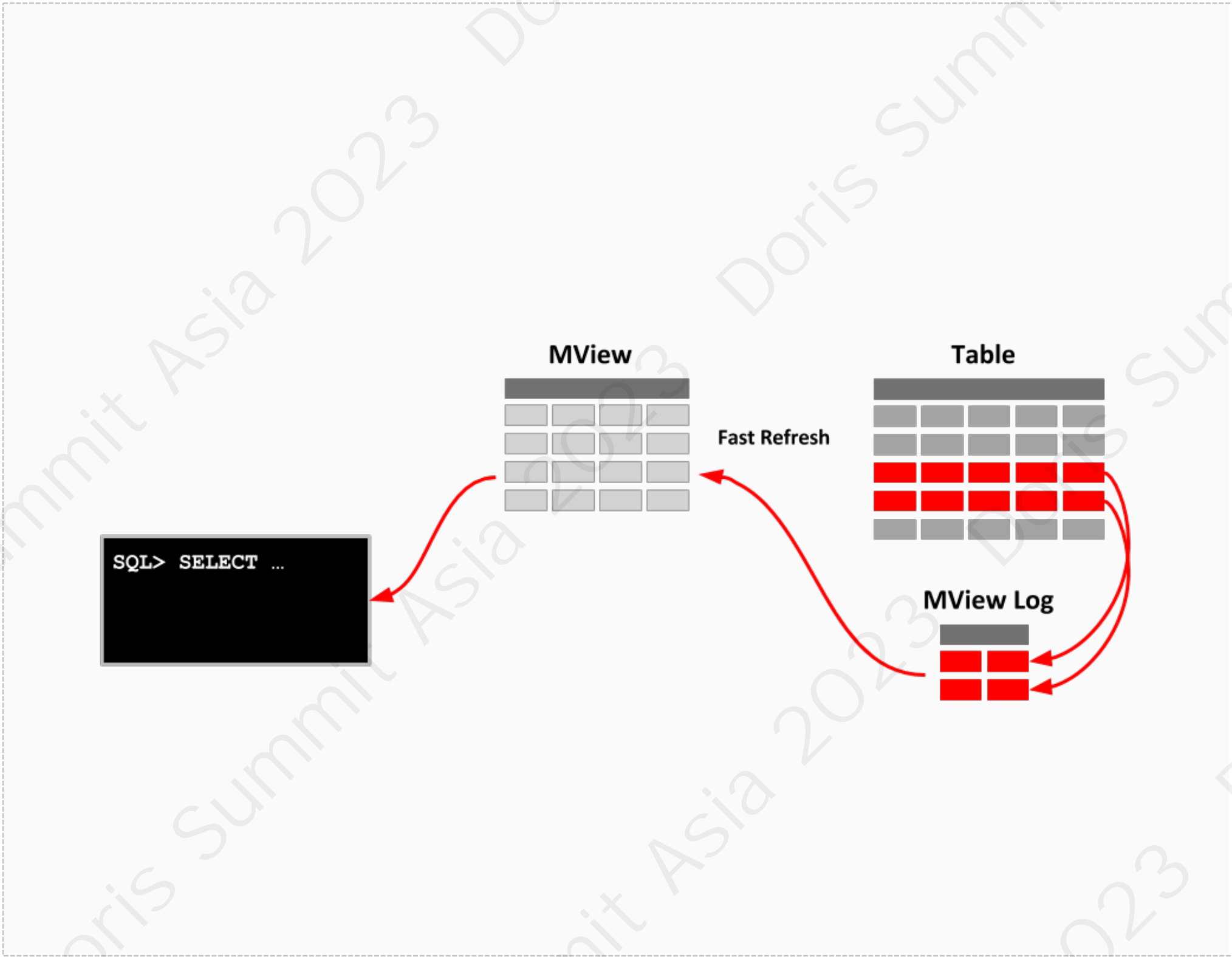
StreamLoad 部分列更新:

```
curl --location-trusted -u root: \
-H "partial_columns:true" \
-H "column_separator:," \
-H "columns:uid,tag2" \
-T /tmp/update.csv \
http://127.0.0.1:48037/api/db1/user_profile/_stream_load
```

InsertInto 部分列更新:

```
SET enable_unique_key_partial_update = true;
INSERT INTO user_profile(uid, tag2)
values (1, 'xxx');
```

Apache Doris 数据建模能力



基于物化视图的 UV 统计

```
CREATE TABLE advertiser_view_record(  
    click_time    DATE,  
    advertiser    VARCHAR(60),  
    channel       VARCHAR(20),  
    user_id       INT  
) DISTRIBUTED BY HASH(click_time) BUCKETS 32;  
  
CREATE MATERIALIZED VIEW advertiser_uv AS  
SELECT    advertiser,  
          channel,  
          bitmap_union(to_bitmap(user_id))  
  
FROM      advertiser_view_record  
  
GROUP BY advertiser, channel;
```


Apache Doris 标签宽表

```
CREATE TABLE orders_based_wide (  
  user_id INT,  
  tag_date DATE,  
  age INT,  
  gender CHAR(2),  
  city VARCHAR(50),  
  region VARCHAR(50),  
  tag1 BIGINT,  
  tag2 BIGINT,  
  tag3 BIGINT,  
  tag5 BIGINT,  
  ... ..  
  tagN BIGINT  
) UNIQUE KEY(user_id)  
DISTRIBUTED BY HASH(user_id) BUCKETS 64;
```

宽表圈人:

```
SELECT user_id  
FROM orders_based_table_wide  
WHERE tag1 = 23 AND tag2 = 18 AND tag3 = 27;
```

宽表人群预估:

```
SELECT COUNT(user_id)  
FROM orders_based_table_wide  
WHERE tag1 = 23 AND tag2 = 18 AND tag3 = 27;
```


Apache Doris 标签高表

```
CREATE TABLE user_tag_high_bitmap (  
  tag          VARCHAR(50),  
  tag_val      VARCHAR(50),  
  date_time    DATE,  
  user_ids     BITMAP BITMAP_UNION  
)  
AGGREGATE KEY(tag, tag_val, date_time)  
DISTRIBUTED BY HASH(tag) BUCKETS 8;
```

判断用户是否在人群包内:

```
SELECT bitmap_contains(user_ids, 13643)  
FROM   user_segments  
WHERE  seg_name = 'seg_1';
```

高表圈人:

```
SELECT orthogonal_bitmap_intersect_count(  
  user_ids, tag, 'tag1', 'tag2', ...)  
FROM   user_tag_bitmap  
WHERE  tag IN ('tag1', 'tag2');
```

Apache Doris 高表宽表结合圈人

```
WITH user_segment_act AS (  
    SELECT orthogonal_bitmap_intersect(  
        user_ids, tag, 'tag01', 'tag02', ..., 'tag102')  
    FROM user_tag_bitmap  
    WHERE tag IN ('tag01', 'tag02', ..., 'tag102')  
)  
SELECT      user_id, age, gender, city, region, phone, ... ..  
FROM        orders_based_table_wide  t0  
JOIN (        
    SELECT uid FROM user_tag_bitmap  
    LATERAL VIEW explode_bitmap(user_ids) tmp AS uid  
    ) as t1  
ON t0.user_id = t1.uid
```

STEP1: 高表通过正交函数圈人

STEP3: 高表结果 join 宽表结果

STEP2: 位图展开后行转列

Apache Doris 解决方案收益

丰富分析函数

留存、漏斗、归因函数
Bitmap 正交函数

多维度建模

宽表圈人
Bitmap 高表圈人

高性能导入

活跃标签部分列更新
多数据源导入方案

用户画像千亿数据，秒级人群预估，秒级 10 标签圈人，10 秒 100 标签圈人

4

Apache Doris 综合湖仓统一方案

数据湖与数据仓库能力

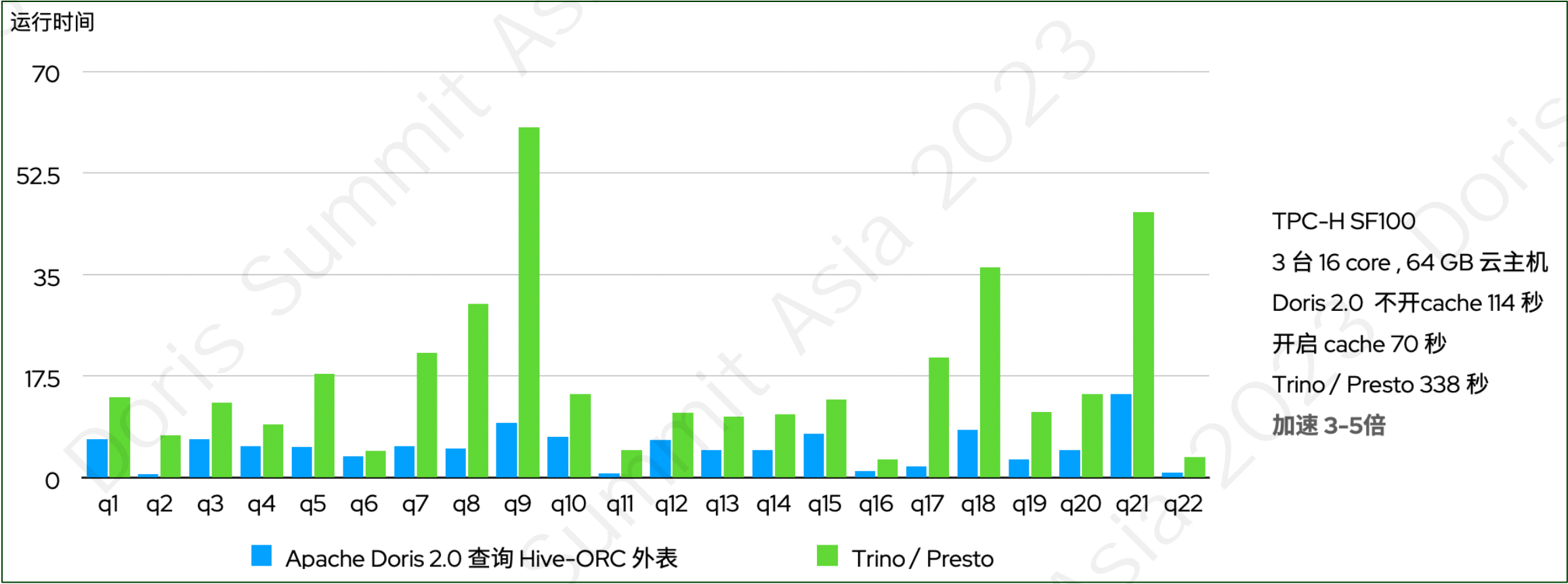
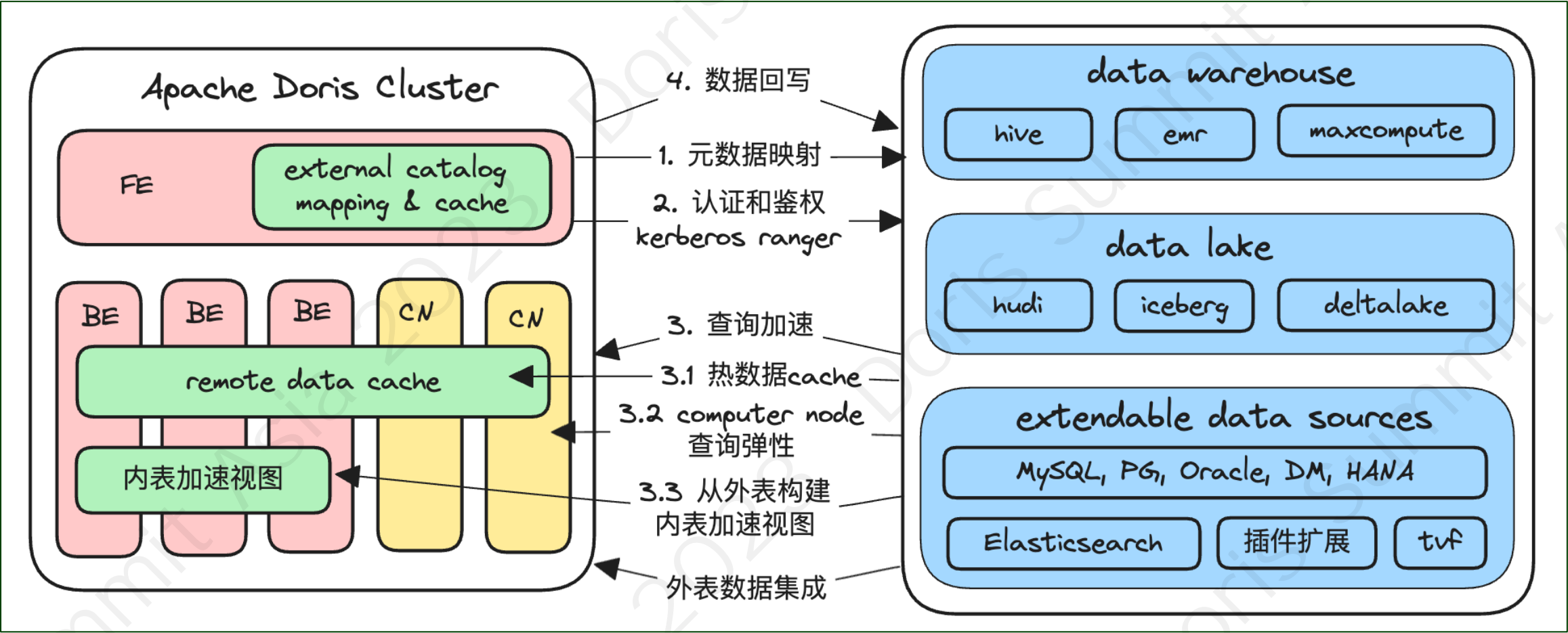
数据湖能力

- 开放的生态
- 灵活的数据访问方式
- 可扩展性
- 高性价比

数据仓库能力

- 高数据质量
- 极致查询性能
- 数据治理及模型分析
- 高实时性

Apache Doris 湖仓一体方案



湖仓一体架构

主要应用场景

湖仓查询加速
数据导入和集成
统一查询网关
ETL / ELT 加速, 写回开放湖仓存储格式

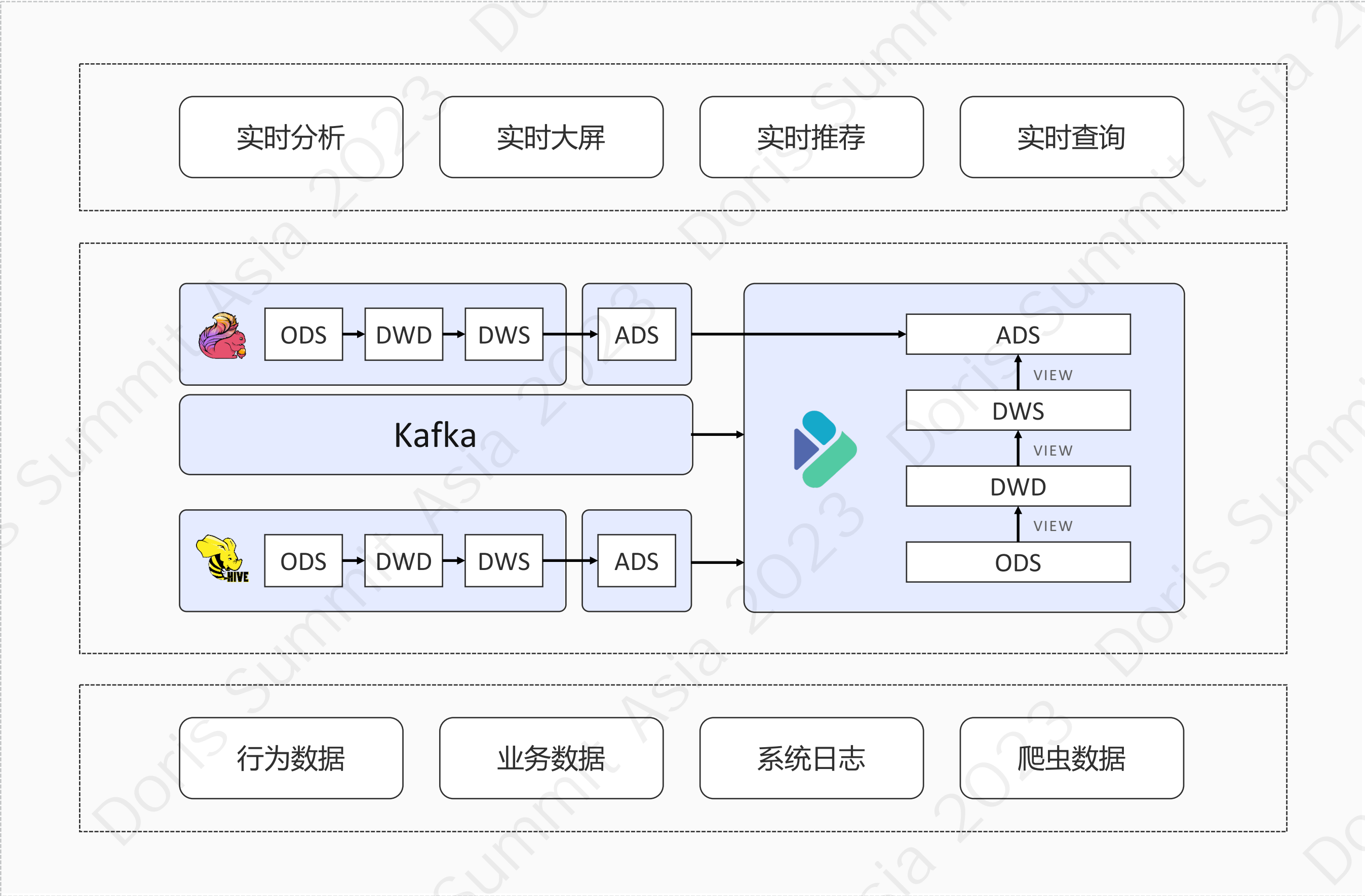
便捷的元数据和数据打通

元数据映射、cache 和 自动刷新
支持几乎所有开放湖仓格式 和 meta store
支持 ES 和关系型数据库, 并且插件扩展
支持外表的认证鉴权, 如 keberos, ranger

分析加速

利用 Doris 高效的分析引擎加速
热数据 Cache 到本地
支持弹性计算节点, 实现计算弹性加速
外表处理结果可写入内表, 形成加速视图

基于 DataLake 的实时业务改造



Doris 实时数仓改造方案

湖仓一体方案

数据下沉到 Hive、Iceberg、Hudi 等湖产品中
Doris 作为计算引擎，进行查询加速

离线数据入库方案

创建 Catalog 链接

```
INSERT INTO doris_tab SELECT * FROM Catalog_tab
```

分析加速

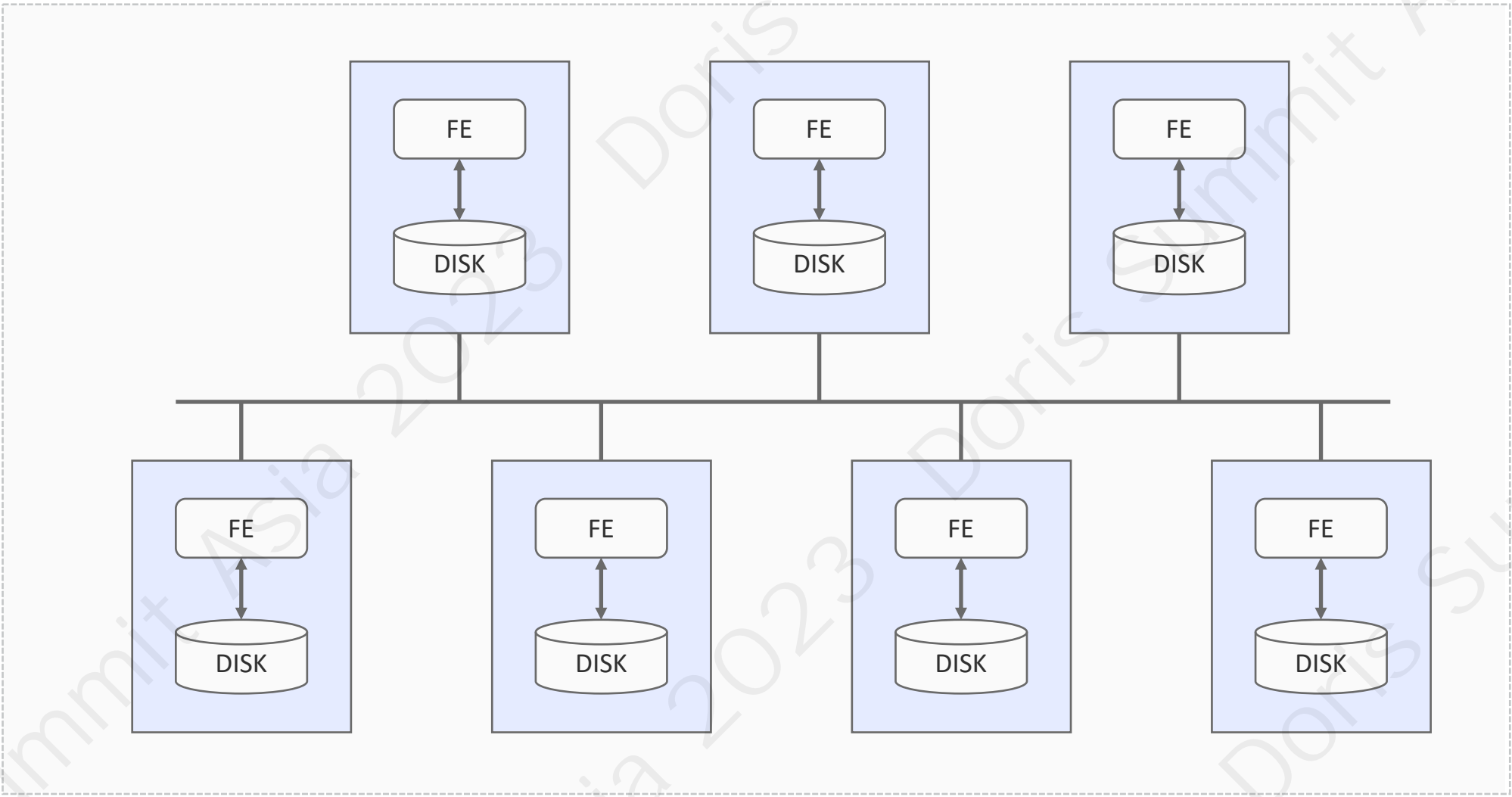
用户生产环境

Hive 查询 56min

Doris Hive Catalog 加速值 7min

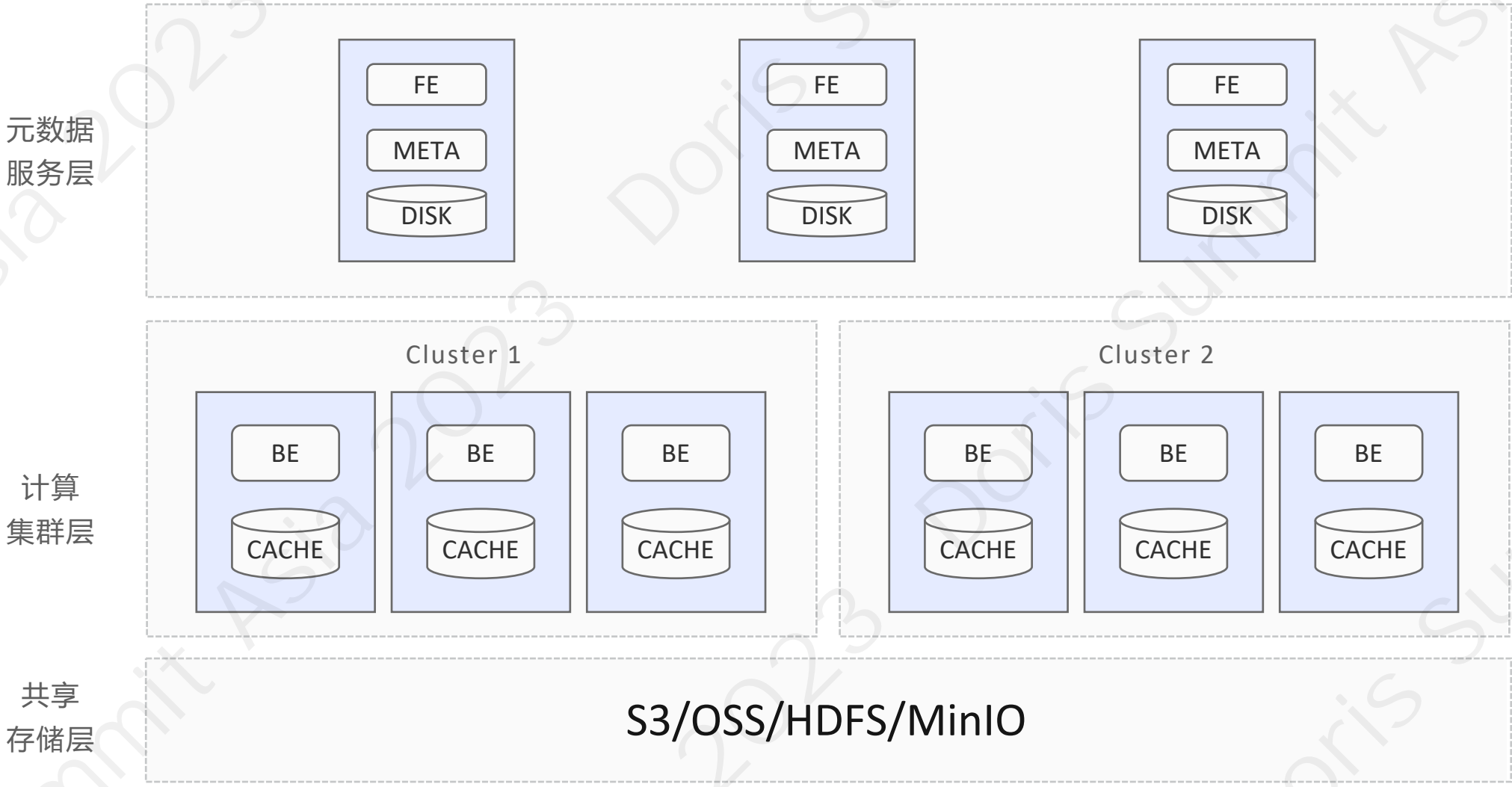
通过 Doris 内表查询 30s

Apache Doris 存算分离架构



存算一体架构

- 简单易部署，易运维，适合绝大多数用户
- FE 和 BE节点都可以灵活扩缩容
- 存储支持冷热数据分层，将冷存储下沉到对象存储或HDFS
- 可以支持弹性计算节点，快速实现计算弹性



存算分离架构

- 飞轮科技基于 Apache Doris 实现了存算分离模式的 SelectDB Cloud
- SelectDB Cloud 将在 V2.1 贡献给 Apache Doris
- 存储资源和计算资源分离，各自弹性，更极致性价比
- 依赖足够稳定、高吞吐的共享存储，通常公有云上才有
- 可以通过多 cluster 机制实现负载隔离，读写分离等机制

Apache Doris 解决方案收益

简单部署

无需数据迁移
简化导入流程

弹性扩展

高性能存算分离方案
计算层弹性可扩展

统一查询接口

支持多源联邦查询
避免数据孤岛产生

Hive 查询 56min, Doris Hive Catalog 加速值 7min, 通过 Doris 内表查询 30s



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>