

基于 Apache Doris 数仓实时离线一体化探索

杨志宇 大数据开发工程师

目录

01 背景介绍

02 架构演进

03 最佳实践

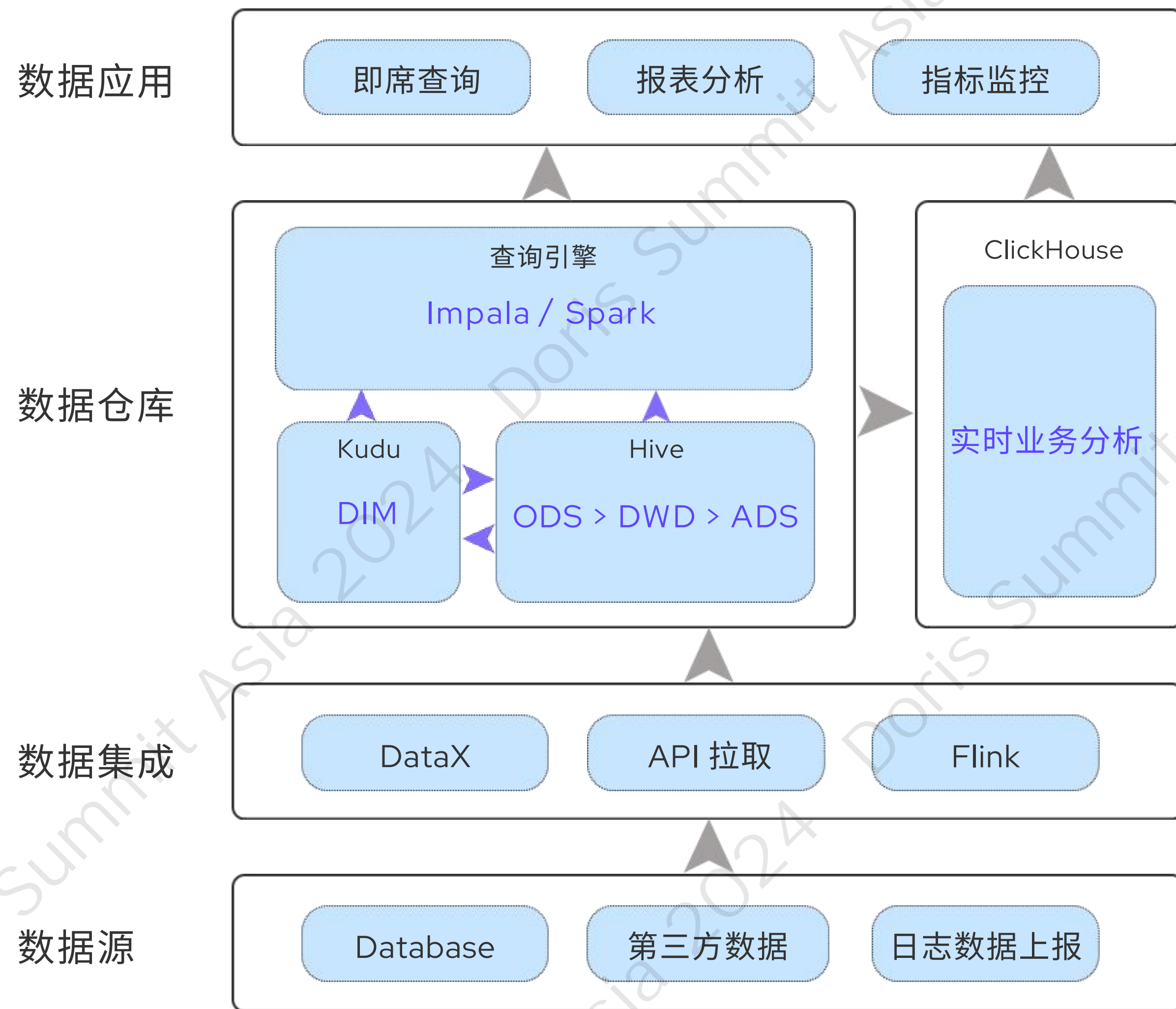
04 未来规划

01 背景介绍

数仓职能的基本介绍

- 01 数据集成**：把多个数据源的数据整合到一起，形成统一的数据存储链路。
- 02 数据存储**：存储大量的数据，对于一些业务大表日志同步到数仓进行备份，定时删除业务库历史日志数据，减少业务库负载。
- 03 数据查询**：项目内部大表大范围在数仓上进行查询，分担业务数据库压力。
- 04 数据处理**：对数据进行清洗转化聚合处理，将数据转化为统一的格式，提高数据利用效率。
- 05 数据分析**：根据需求进行业务分析，为业务决策提供数据支持。

历史架构



痛点

- 组件太多，架构复杂，运维困难+
- 对于开发者技能要求高，研发成本高
- 查询效率低，存储成本高

架构升级的核心诉求

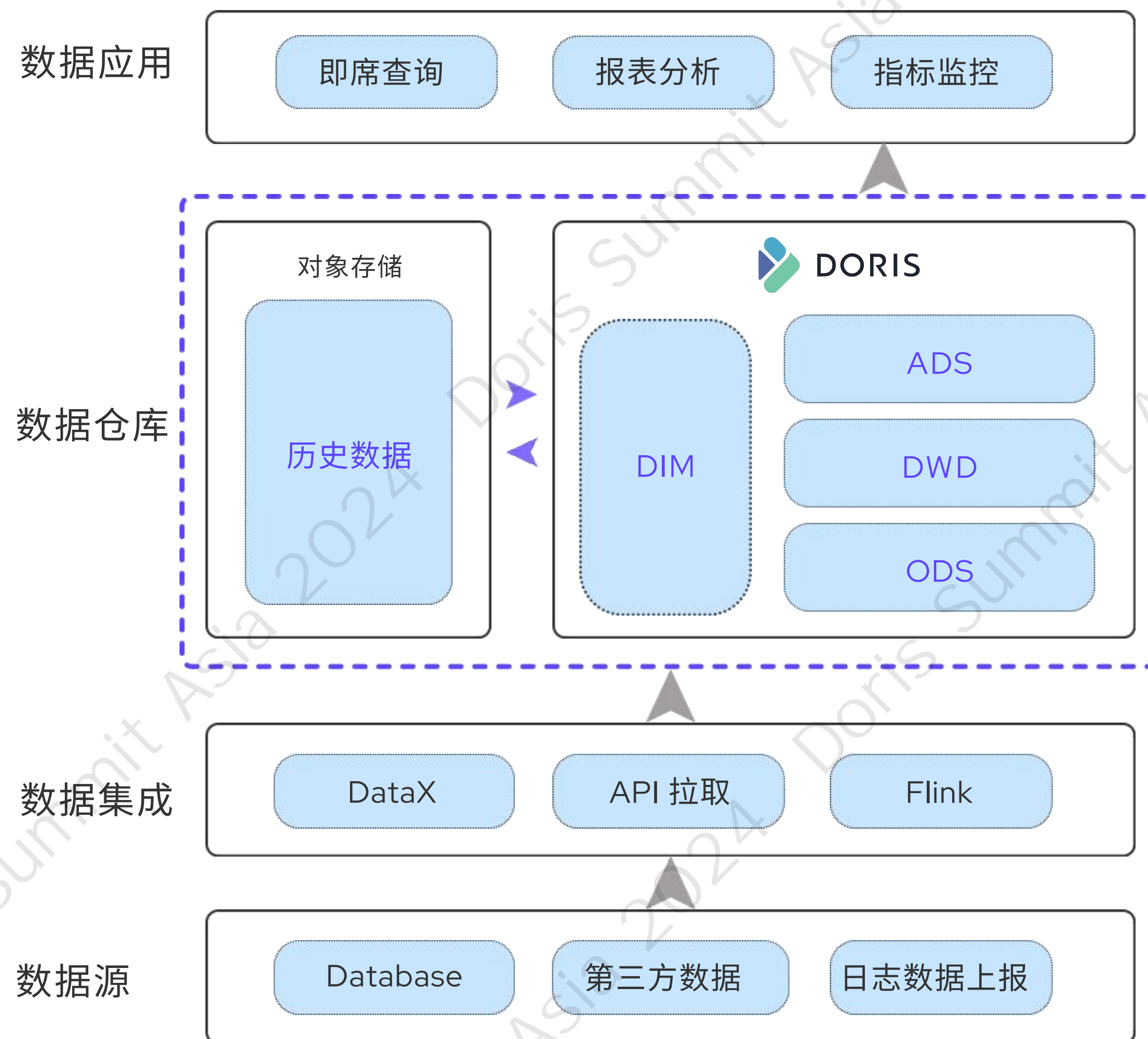
- 组件减少：降低架构复杂度，降低运维成本
- 统一查询：降低学习和开发成本
- 效率提升：查询支持秒级别返回
- 节约存储：历史数据能够存储在 oss，降低存储成本

02 架构演进

新一代数仓选型

关键要素	说明
数据查询	亿级别单表数据查询能够秒级别返回，多表 Join 性能好
存储成本	可以使用对象存储来对数据进行冷热分析，减少存储成本
使用成本	兼容 MySQL 协议，传统 DBA 也能进行相应的数据开发
社区活跃度	官方文档详细，社区活跃度高，版本更新快
数据导入	开源社区有成熟的离线和实时导入工具
运维部署	有官方的集群 Manager 工具，运维升级简单

离线实时一体化数仓



Impala/Spark+Kudu+Hive+ClickHouse



基于 Apache Doris 的离线实时一体化数仓

- 实时离线一体，架构简单，方便集群维护和数据治理
- 冷热存储，实现成本与效率的均衡

应用收益

1、60% 的数据存储在对象存储上，数仓投入成本减少了 50%~60%

2、查询结果返回达到秒级别，简单查询甚至达到毫秒级别，大大提高了用户体验

3、离线和实时数据进行整合，减少数据处理链路长度，代码复杂度和任务失败率

4、集群故障追踪简单，不需要在多个组件上去查找问题，运维成本大大降低

03 最佳实践

冷数据查询优化

问题定位

表现：对象存储上面的大表冷数据查询慢，需要 3-5 分钟才能返回结果

原因分析：

- 天数据 30GB 左右，按天分区，每个区只分了 4 个桶，tablet 达到 8~10GB
- 高频查询字段，例如物品 id，角色 id 等，未放在 key 中

优化措施

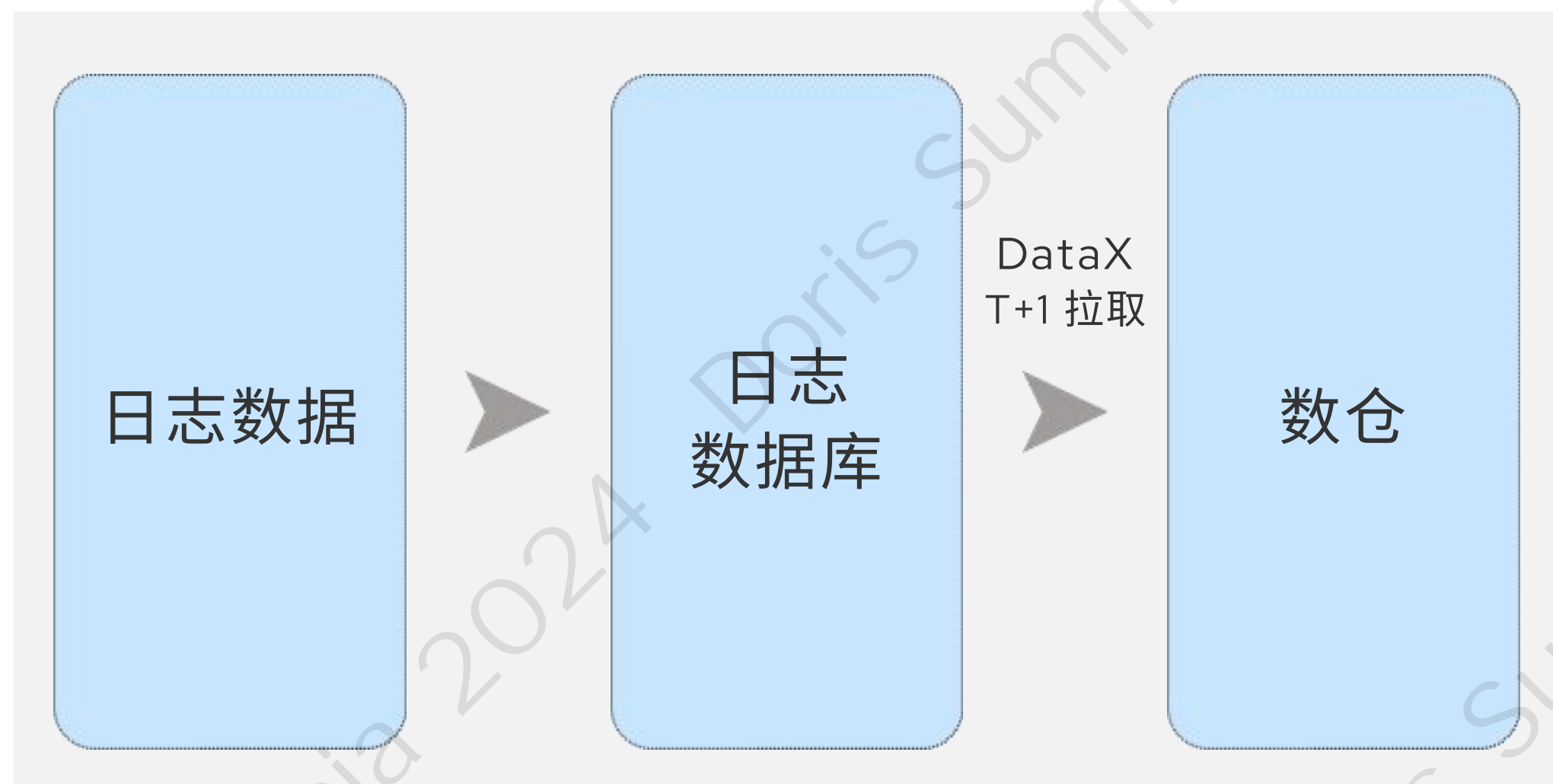
- 重新设计表，将 tablet 大小规划到 1~2GB，单日数据分为 25 个桶
- 充分利用前缀索引的功能，将高频查询字段根据查询频率从高到低，依次从左到右放入 key 中
- 对于经常查的字段，使用布隆过滤器

结果

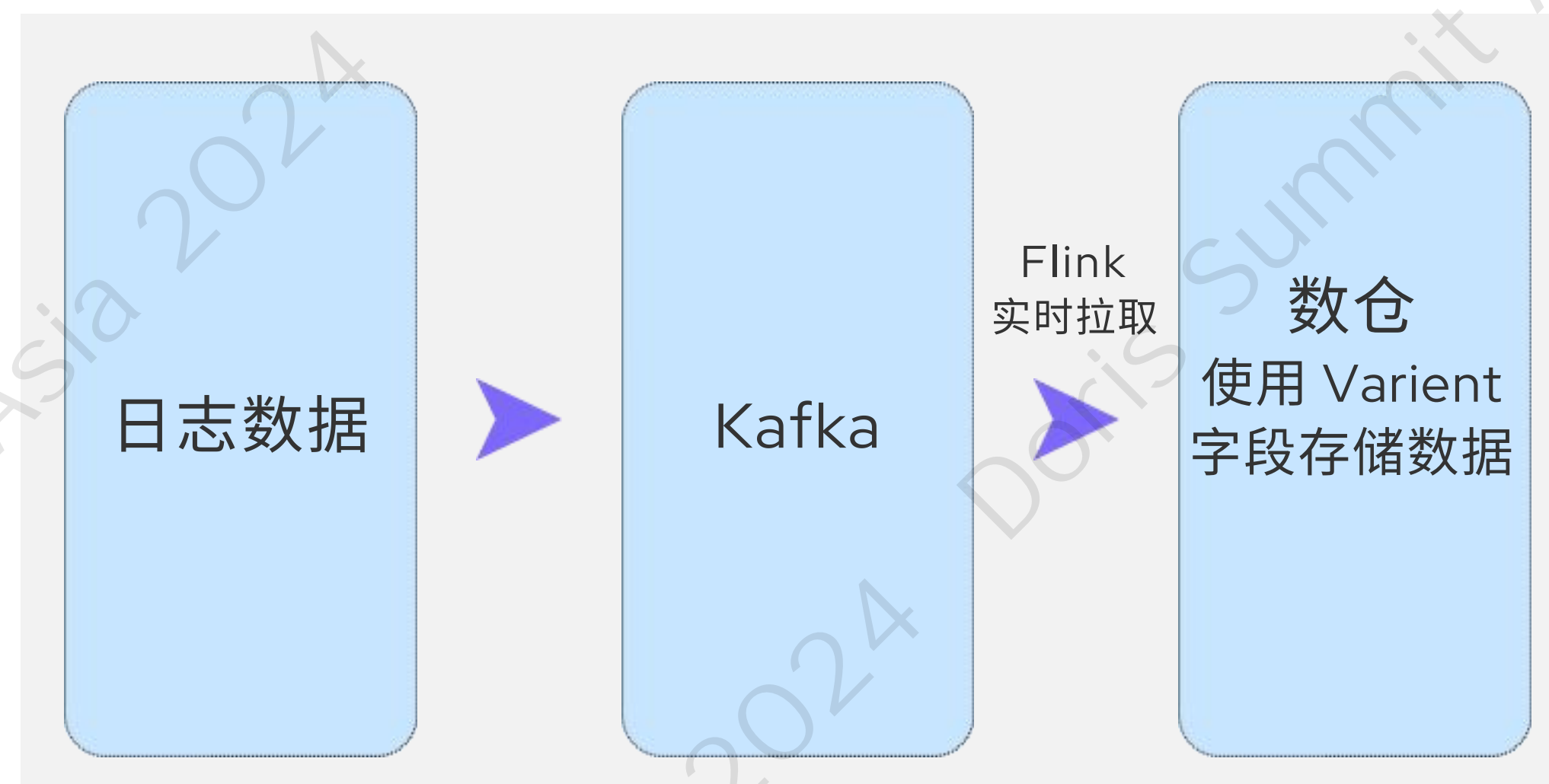
- 大表冷数据查询能达到秒级别的返回速度，查询效率提升数十倍

游戏日志接入重构

老架构



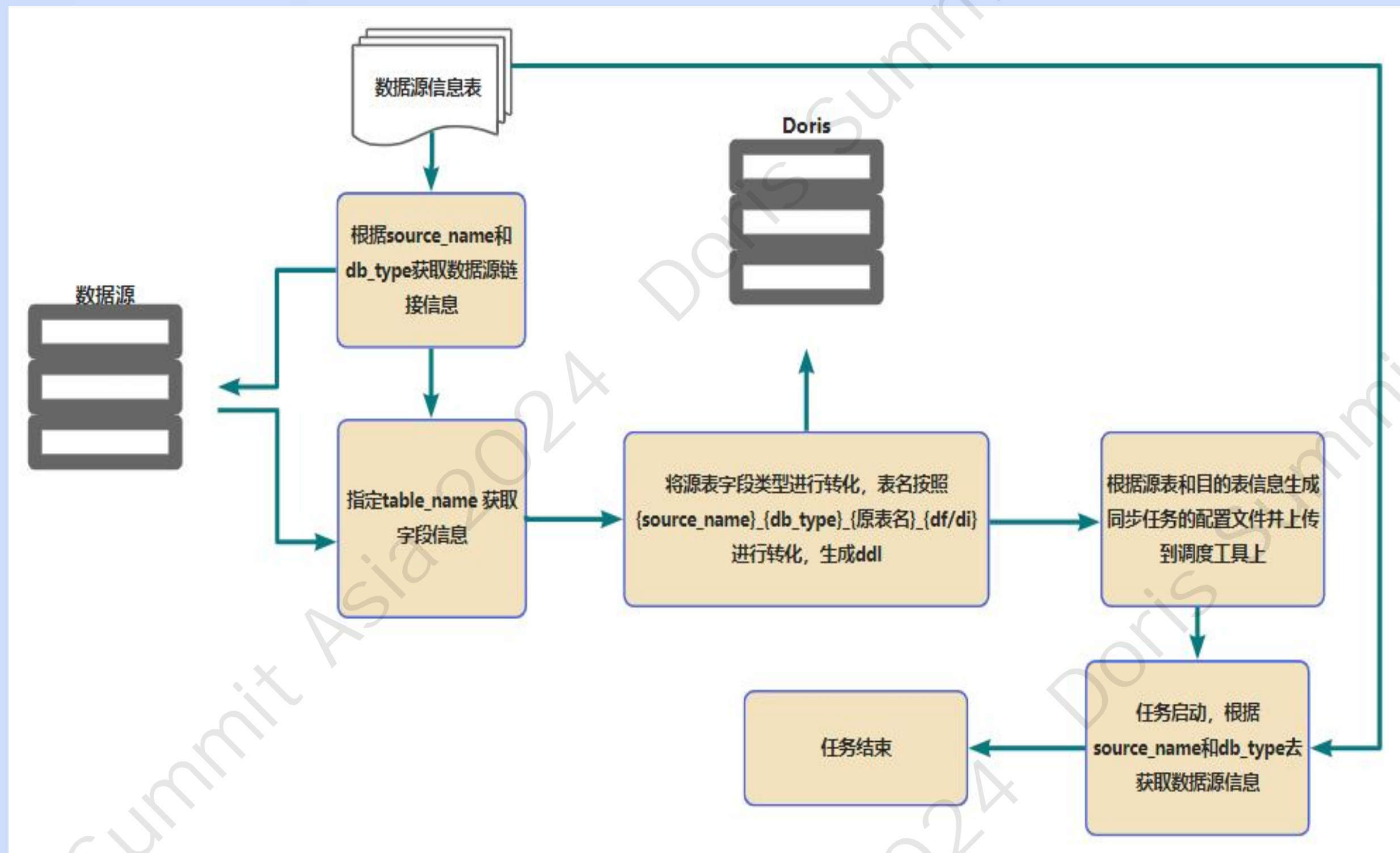
新架构



结果

- Flink 消费 Kafka 写入 Doris 实时数仓，数据延迟在秒级别
- 实时日志每日接入数据量 100GB+，单表数据查询在毫秒和秒级别，成功替换老版内部业务日志数据查询功能，减少服务器成本
- 日志数据 json 结构修改，不需要修改库表结构，Flink 任务不需要上下线，减少开发和运维成本

数据治理



问题

- 数据源众多，导致 ODS 层表多，命名不规范复杂
- 表所有字段都为 string 类型，浪费存储
- 有些业务存在分库分表情况，数据库维护时会进行合库和分库操作，导致任务拉取失败

解决措施

- 统一管理数据源
- 规范 ODS 表命名规则: 按照 {source_name}_{db_type}_{源表名}_{di/df/ri/rf} 格式命名
- 对源库字段类型和 Doris 字段类型进行映射，自动化建表
- 数据同步数据源信息懒加载：任务启动的时候，根据 source_name 和 db_type，取 ip，port 和 dbname 信息

04 未来规划

未来规划

1. 调研 3.0 版本的存算分离架构，考虑将数据全部存储到对象存储
2. 使用异步物化视图，减少数据链路和代码维护
3. 调研倒排索引，Elasticsearch 数据迁移，持续优化存储成本

Thanks for Watching !