Apache Doris 在电商场景的最佳实践

王昌业 | 抖音集团 实时数据开发工程师
何宇彤 | 抖音集团 实时数据开发工程师

目录

背景介绍-业务背景

抖音电商

- **数据基建：**实时数据覆盖交易、流量、内容等电商全业务领域；
- **应用场景：**面向内部运营实时监控、运营提效和数据挖掘；面向外部商家实时诊断和实时经营策略调整；
- **数据产品：**数据透出产品包括对内的运营类产品，对外的商家产品以及分析报表平台等。

背景介绍-实时场景

实时大屏

- 高性能
- 高时效性

上图为Demo

实时预警

- 高性能
- 高时效性

实时分析

- 多维度
- 长周期
- 近实时场景

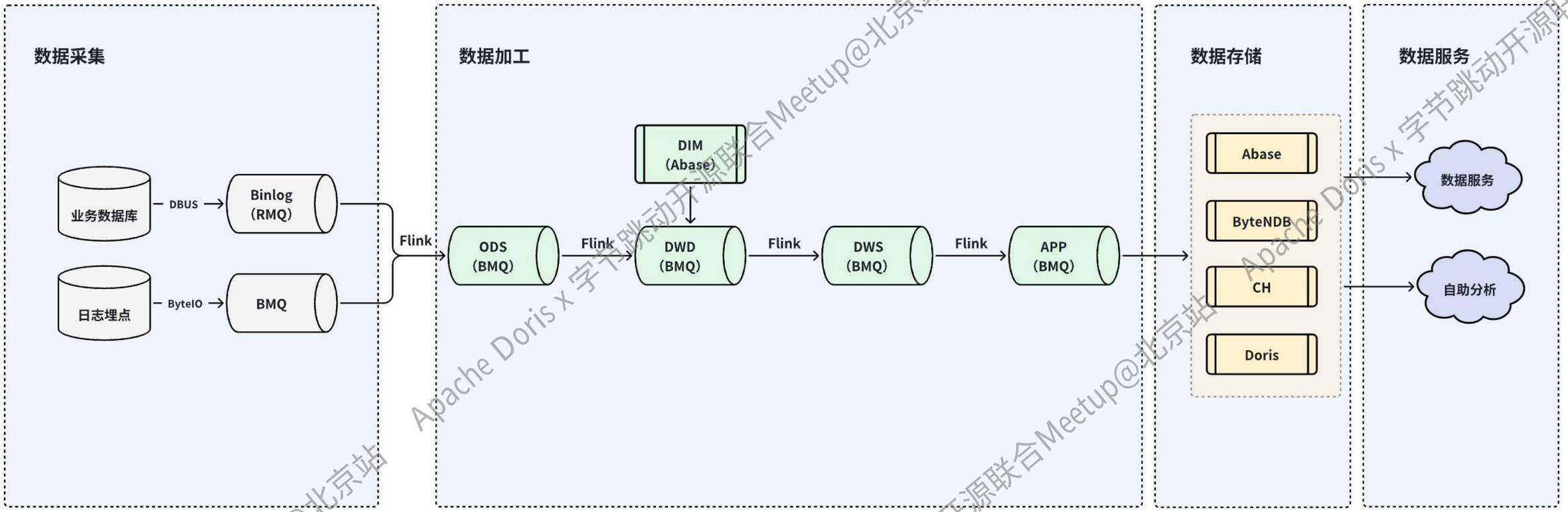
上图为Demo

实时榜单

- 多数据源
- 长周期
- 近实时场景

目录

架构演进- 基于 Flink 的实时计算架构



数据采集

- DBUS业务系统数据采集
- ByteIO埋点数据采集

数据加工

- Flink作为核心计算引擎
- Abase维度表
- BMQ消息通道

数据存储

- Doris/CH等OLAP分析引擎
- ByteNDB列表查询
- Abase点查询

数据服务

数据服务平台对外提供接口

- BI 平台对外提供分析服务

架构演进 - 架构 1.0

问题和挑战

资源成本高

- Flink 常驻任务资源利用率低于 60%，且需要预留资源
- Yarn 队列资源利用率受业务波动影响，潮汐调度困难
- 外部依赖如消息队列 BMQ、高速缓存 Abase 等成本高

数据质量保障难

- 实时数据受维表更新频率、状态周期 TTL、消息顺序等影响
- 部分离线算子在实时侧难以实现，需要折中处理(group by、全局 distinct 等)

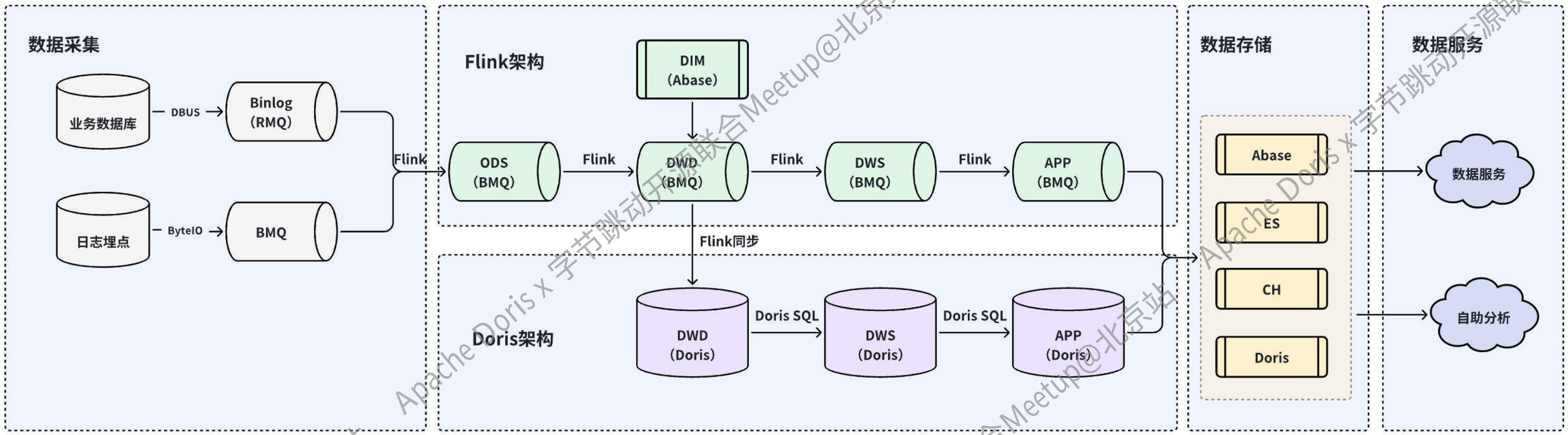
开发效率低

- 方案设计复杂，外部依赖多(Abase、BMQ)
- 数据验证、回溯等需要额外开发配套任务

任务稳定性差

- Flink 任务受依赖组件稳定性影响
- 业务潮汐对任务有冲击，高时效敏感任务报警频繁

架构演进 - Doris 准实时数仓架构

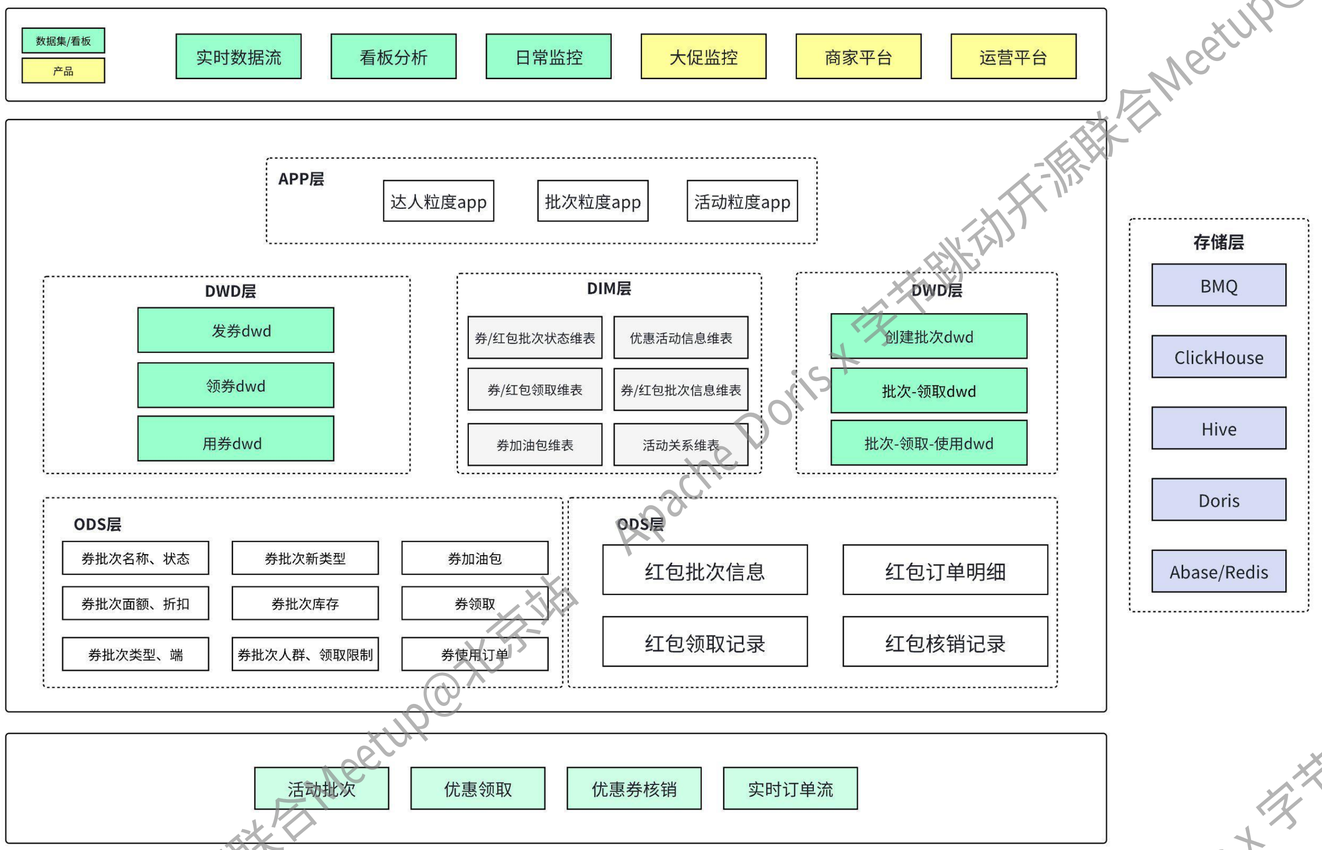


Doris 架构的优势

- **功能特性：**依托列式存储引擎、MPP 架构、向量化查询引擎、预聚合物化视图、数据索引等特性，在低延迟和高吞吐查询上，达到了极致性能。
- **简单易用：**Doris SQL 简单易用，上手难度低，函数支持丰富。
- **效率提升：** 作为 OLAP 存储和分析引擎, 结合秒级调度, 通过合理的数仓分层, 可以类似离线开发一样处理数据; 同时可以清晰地进行问题排查以及修复, 提升开发运维效率。

目录

应用实践 I -红包优惠券投放监控



红包优惠券投放业务架构图

业务背景

营销就是圈选一批商品（Product），基于一定的策略（Strategy）对价格（Price）进行干预，并在指定的渠道（Place）进行宣传的活动。红包和优惠券是最常见的营销方式。 1

痛点问题

1 业务周期较长

红包和优惠券数据周期长，长达 **90+** 天

2 数据存在热点

大促期间促销，部分红包优惠券成为热点数据。

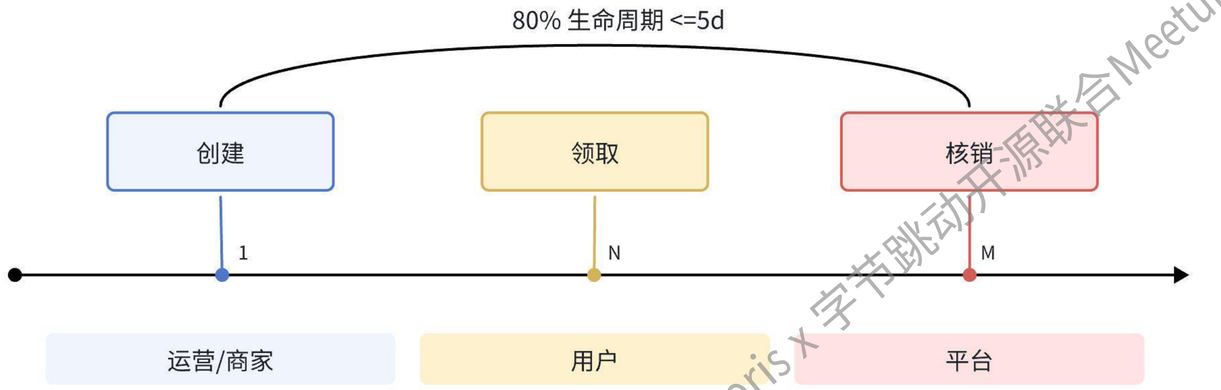
3 任务稳定性差

任务周期性 CP 失败，大促高峰期延迟严重

4 计算成本高昂

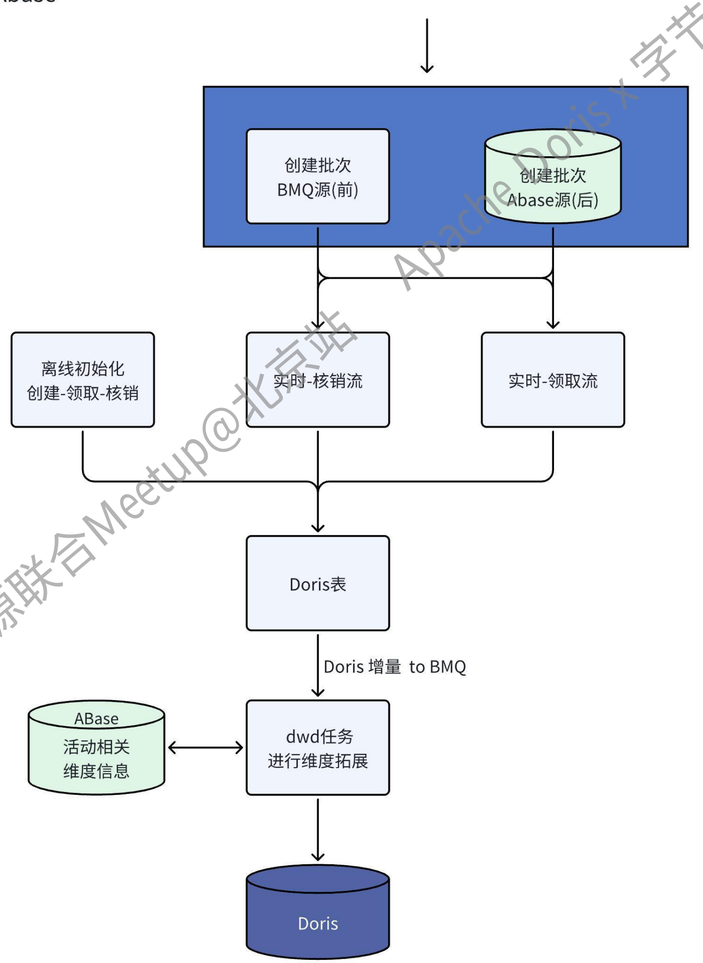
数据 TPS 较低，但 Flink 单个任务消耗 **CPU2000+**，内存 **10T+**

应用实践 I -红包优惠券投放监控



创建批次相关字段取值逻辑

IF(BMQ.field IS NOT NULL AND BMQ.update_time > Abase.update_time , BMQ.field , Abase.field)
FROM 主表
LEFT JOIN BMQ
LEFT JOIN Abase



实体关系

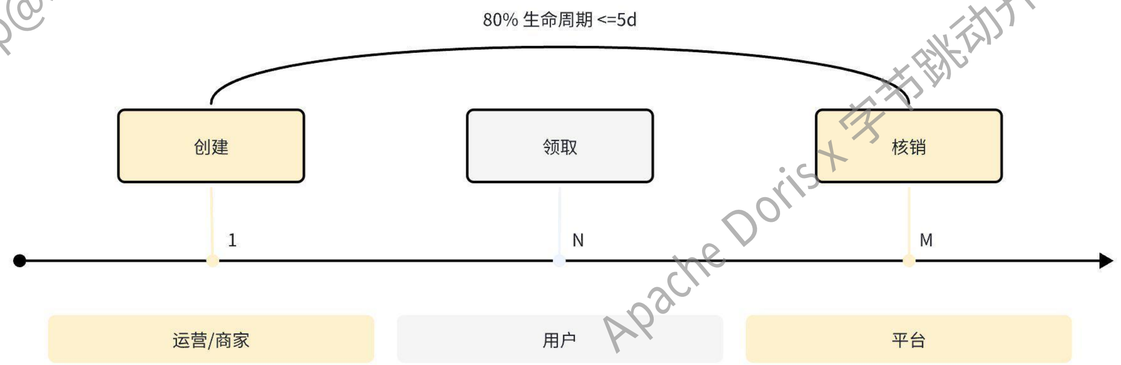
- 1 - N - M (N>=M), 即创建一个批次对应多次领取及核销, 领取后仅支付未退款才会触发核销

数据分布

- 用户券使用生命周期 <= 1d 占比 70.1%
- 用户券使用生命周期 <= 5d 占比 89.5%, 从次日起每日增加约5%

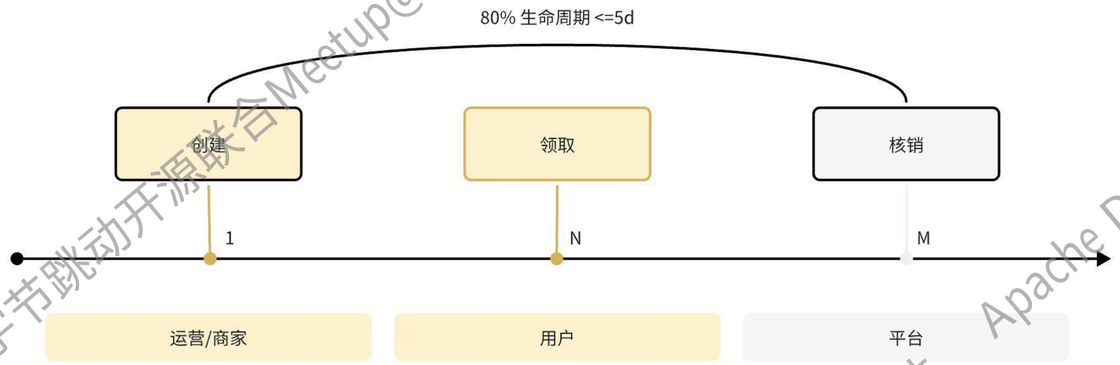
应用实践 I -红包优惠券投放监控

```
SELECT  redpack_meta_id,
            redpack_id
        ...
        FROM    ods_ecom_usr_rpk_get_event  AS redpack_get_event
LEFT JOIN
        dim_redpack_meta for SYSTEM_TIME AS of redpack_get_event.proc AS
dim_redpack_meta
ON  redpack_get_event.redpack_meta_id = dim_redpack_meta.redpack_meta_id
LEFT JOIN
bmq_redpack_meta AS bmq_redpack_meta
ON      redpack_get_event.redpack_meta_id = bmq_redpack_meta.redpack_meta_id
WHERE   (
        dim_redpack_meta.create_time IS NOT NULL
        OR bmq_redpack_meta.create_time IS NOT NULL);
```



红包/优惠券核销 x 批次写入 Doris 表

红包/优惠券领取 x 批次写入 Doris 表




```
SELECT  redpack_meta_id,
            redpack_id
        ...
        FROM    ods_ecom_usr_rpk_get_event  AS redpack_get_event
LEFT JOIN
        dim_redpack_meta for SYSTEM_TIME AS of redpack_get_event.proc AS
dim_redpack_meta
ON  redpack_get_event.redpack_meta_id = dim_redpack_meta.redpack_meta_id
LEFT JOIN
bmq_redpack_meta AS bmq_redpack_meta
ON      redpack_get_event.redpack_meta_id = bmq_redpack_meta.redpack_meta_id
WHERE   (
        dim_redpack_meta.create_time IS NOT NULL
        OR bmq_redpack_meta.create_time IS NOT NULL);
```

应用实践 I -红包优惠券投放监控

结果和收益

资源消耗降低

- 整体资源消耗降低 **95%+**，CPU 使用从数千降低至 **100+**

数据质量提升

- Doris 可以轻松实现千万级数据关联，长周期数据保证数据 **100%** 准确率

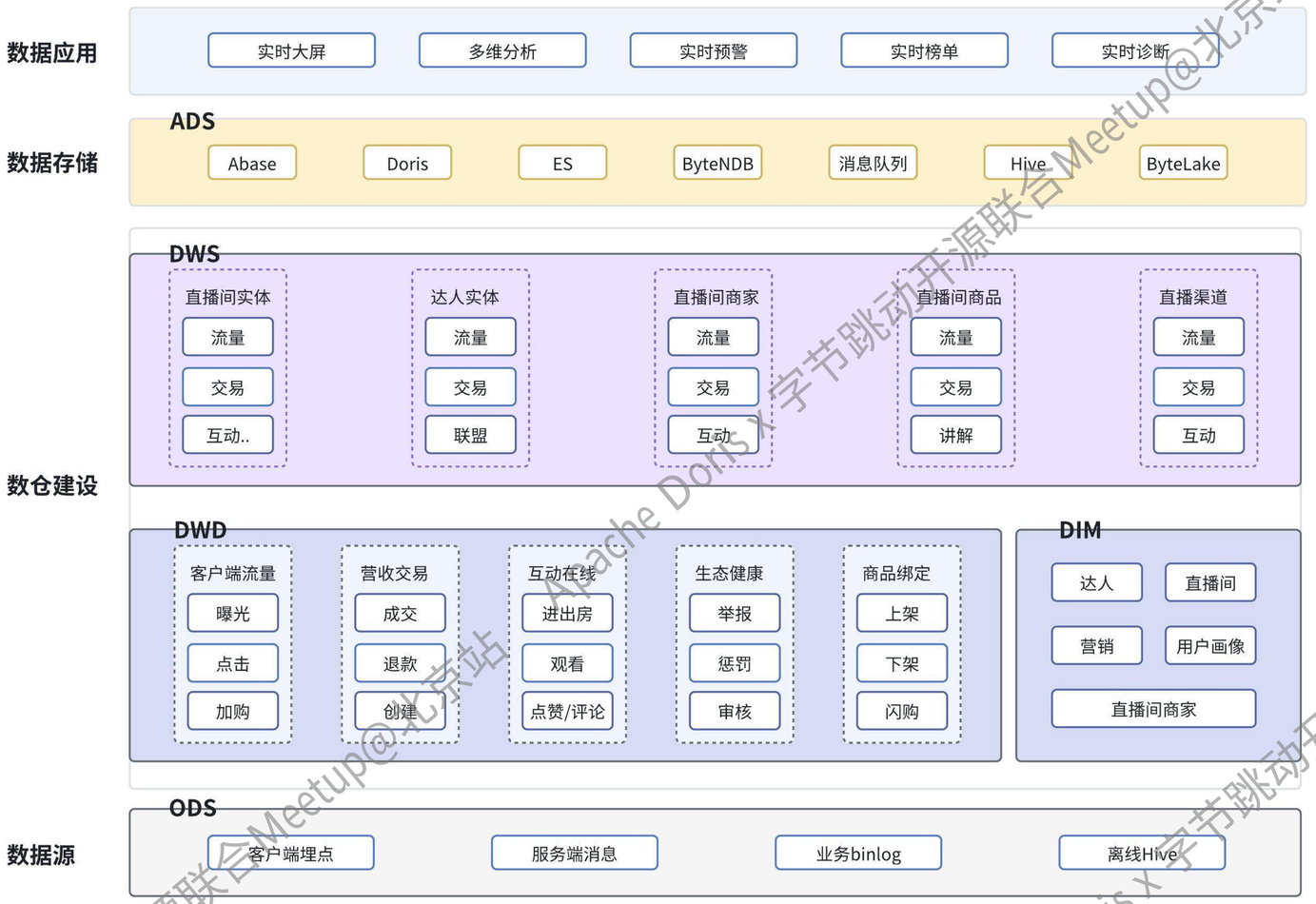
稳定性大幅提升

- 消灭 Flink 常驻大状态任务，报警数从 3-5 次/周降低至小于**1次/月**
- 任务数量减少至 2 个，运维成本显著降低

开发效率显著提升

- 基于 Doris SQL 轻松实现业务迭代，迭代效率提升 **30%+**，人效提升**50%+**

应用实践 II -抖音电商直播



抖音电商直播 - 业务架构图

业务背景

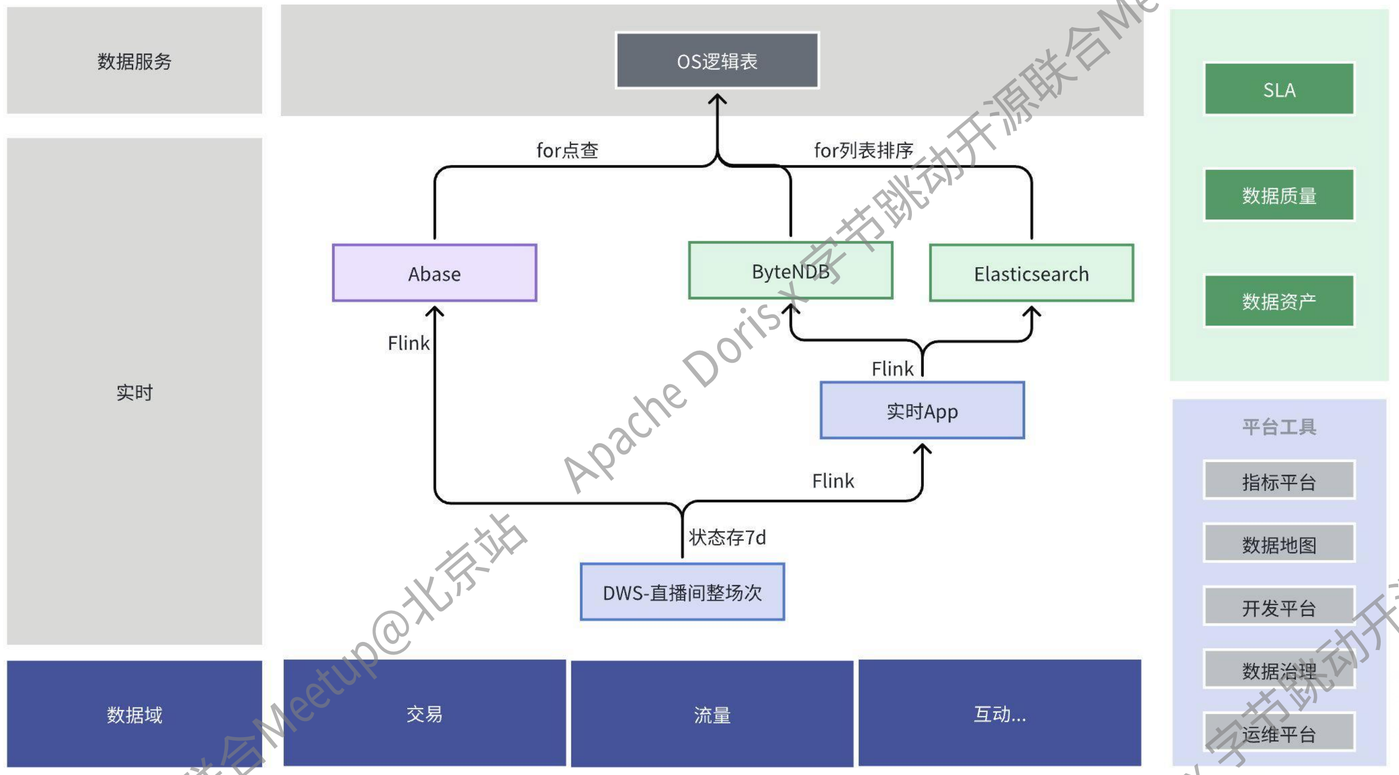
1 抖音电商直播

- 抖音电商以优质内容为基础，以直播为主要手段重构“人、货、场”零售三要素，挖掘用户潜在需求。系统通过精准推荐，吸引用户并最终完成商品购买。
- 对于 C 端用户：直播弥补了传统图文、视频题材中信息缺失、交互不及时等缺点。
- 对于 B 端商家：与用户有效互动讲解答疑，缩短交易链路。

2 直播数据产品

- 直播前计划
- 直播中盯盘
- 直播后复盘和诊断
- 帮助商家/达人以数据引领生意增长

应用实践 II -抖音电商直播



直播长周期历史解决方案

痛点问题

1 发布周期长

单次迭代双跑 7 天方能上线

2 任务状态大

数据量大, 状态大, eg.流量场景 :5T+，计算成本高

3 故障恢复慢

稳定性风险高,修数成本高

延迟恢复慢,eg.流量 3h+

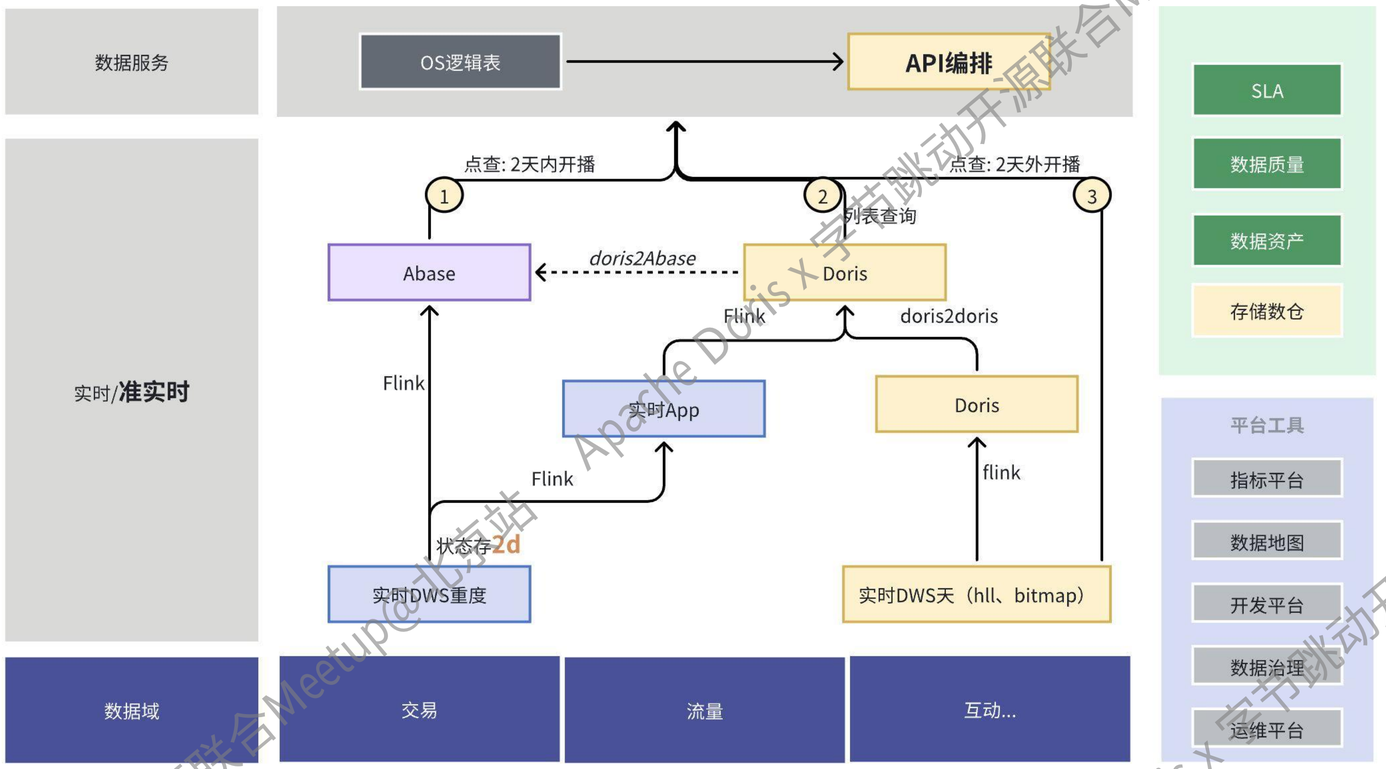
4 问题定位难

离线实时同存, 线上问题难区分

5 超长直播间问题

超长直播间超过 7 天无法更新

应用实践 II -抖音电商直播



直播长周期新的解决方案

解决方案

1 实时&准实时链路

- 实时-Flink:近2天内开播的直播间, 覆盖99%场景
- 准实时-Doris:2~20天内开播的直播间，分钟级更新
  - Doris 2 doris、Doris 2 abase(可选)调度
  - 查询加速: 交易-bitmap、流量-hll

2 点查场景

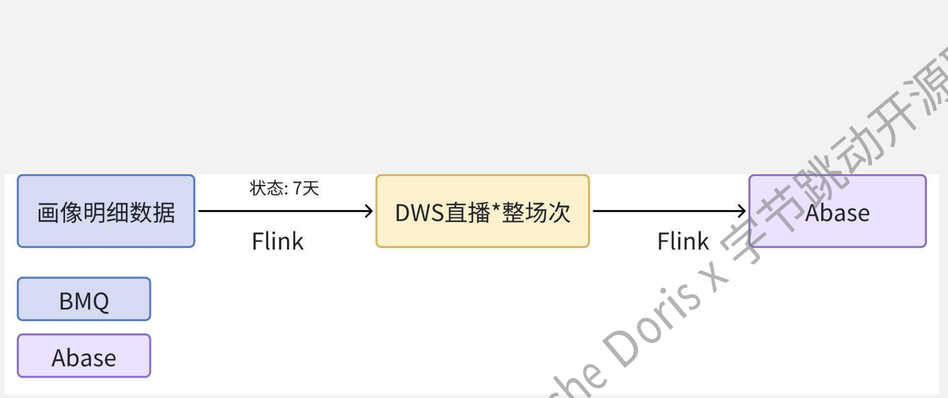
- for直播中盯盘：eg.直播大屏
- 数据服务的API编排-路由查询
  - 近2天开播->实时(Abase)
  - 2天外开播->准实时(Doris)数据
    - 可选: Doris2Abase->Abase

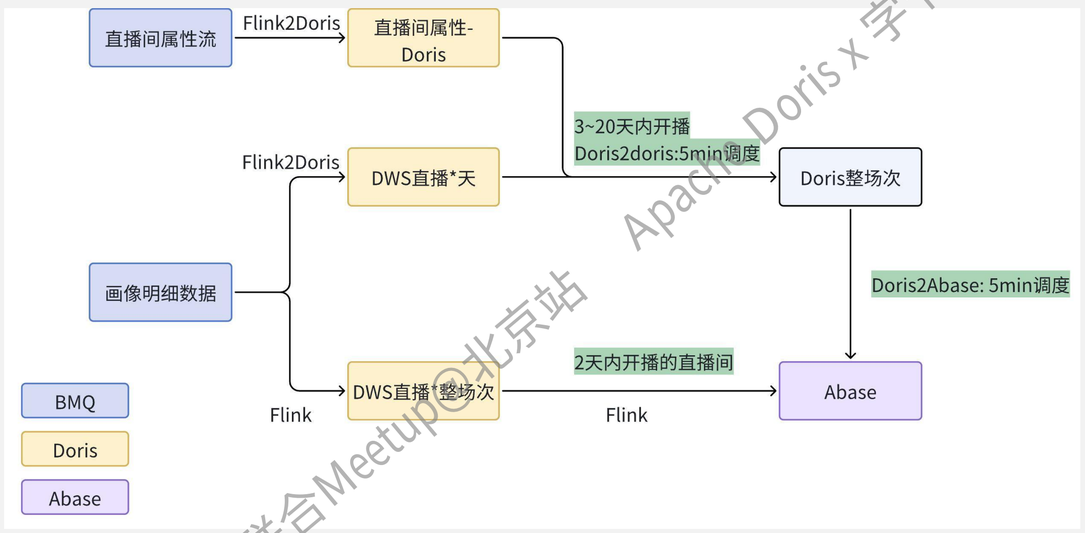
3 排序场景

- for 直播后复盘: eg.直播详情页
- 实时离线同存储 Doris

应用实践 II -抖音电商直播

方案落地-直播人群画像(点查场景)





旧架构

- Flink 任务, 限定 7 天长周期计算
- 资源损耗高: 状态大, 单场景 5T+
- 超过 7 天的直播间无法更新

新架构

- Flink 任务, 状态只需保留 2 天
- 资源损耗低: 状态变小, 单场景 5T+ -> 1T 以下
- 超过 7 天的直播间可以更新

应用实践 II -抖音电商直播

方案落地-直播间详情页(排序场景)

旧架构的痛点

1 ES 读写能力不足&成本高

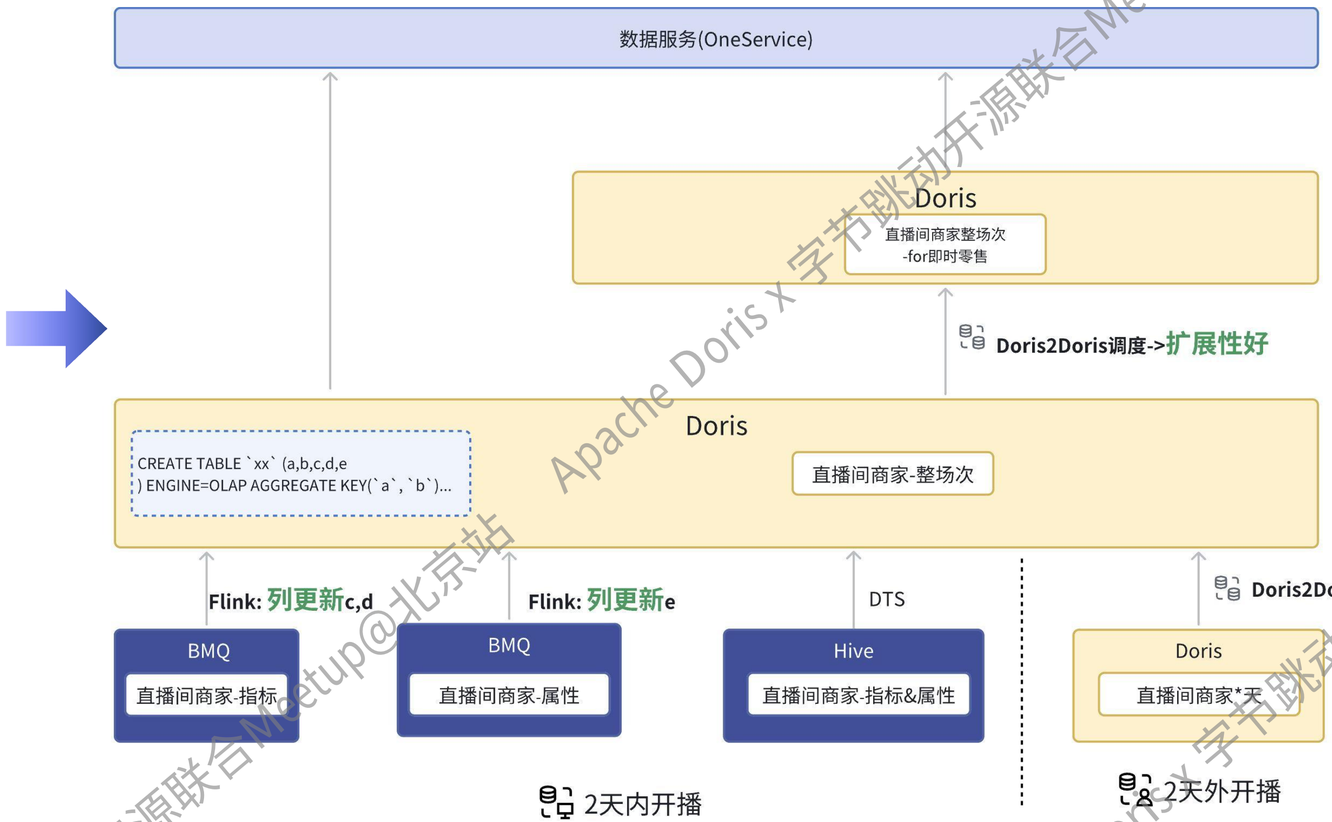
- ES 写入延迟，CPU 打满。原因: 索引、热点问题
- 聚合/排序场景体感支持 QPS <500
- 成本: ~1000w/year

2 扩展能力不足

- ES 的数据，不支持进一步的近实时加工

应用实践 II -抖音电商直播

方案落地-直播间详情页(排序场景)



新架构的方案

存储:ES->Doris 1

- 模型特点
  - 唯一主键约束 + 更新部分列
  - 按月分区，分区字段使用开播时间
  - Aggregate 模型，非主键列的聚合REPLACE_IF_NOT_NULL

能力扩展 2

- Doris2doris 调度、join doris 表

新架构的效果

1 读写能力&成本

- 写入能力: 30w/s -> **百万/s**
- 查询能力: QPS<500 -> **2000+**
- 成本: 降低 **90%**

2 扩展能力

- Doris sql 调度：近实时链路加工
- 支持了更多的业务场景

应用实践 II -抖音电商直播

结果和收益

发布周期降低

- 每次迭代需要跑 7 天才能上线 -> 最少 1 天
- 迭代效率极大的提升

任务状态降低

- 大流量场景的状态由 5T+ 降低至 1T 以下

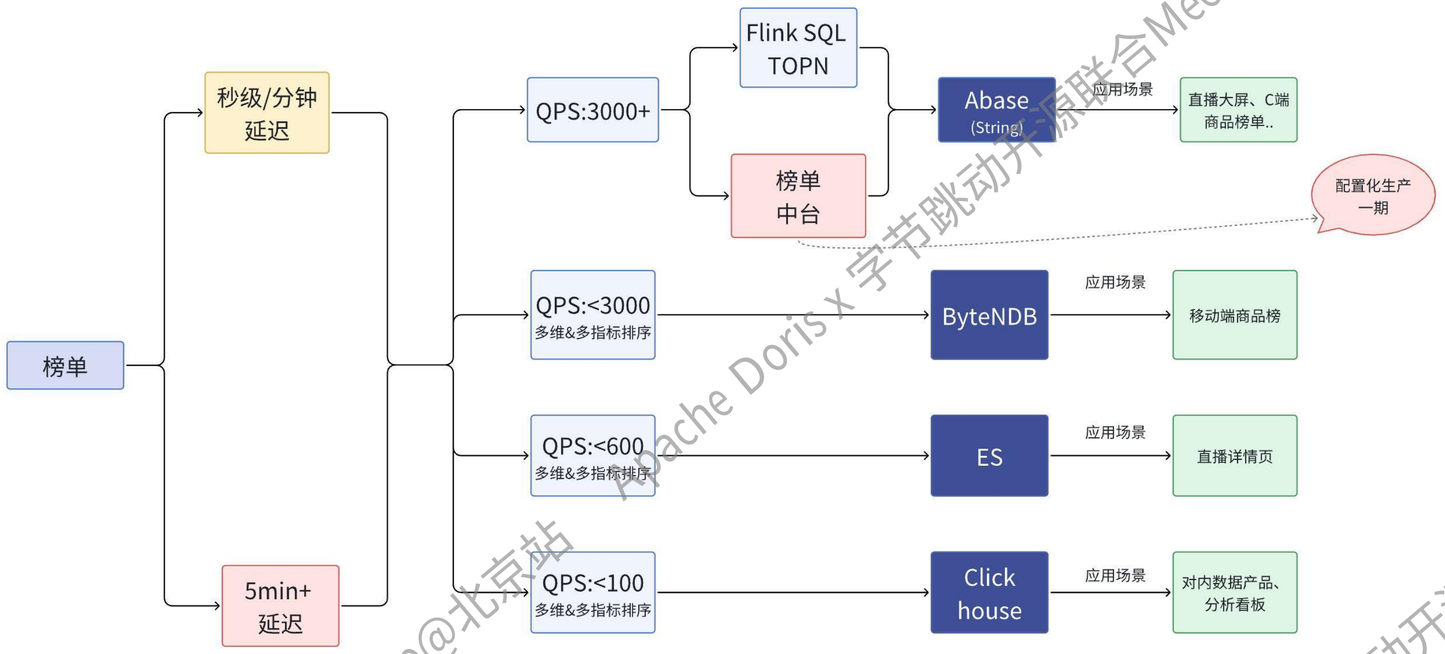
故障恢复加快

- 消除大状态, 异常时从状态恢复加快
- 修数成本降低

解决超长直播间问题

- 解决超长直播间超过 7 天无法更新的难题
- 业务客诉降低

应用实践III-实时榜单及配置化生产



实时榜单历史解决方案

业务背景

1 抖音电商实时体系，
榜单场景丰富，多种实现方式

2 统一按实时榜单建设

应用实践III-实时榜单及配置化生产

痛点问题

1 榜单开发成本较高

- 实现方式各异，各业务单独建设
- 选型依赖经验

2 计算资源消耗大

依赖 Flink 预计算,计算链路任务多

3 存储写入压力大

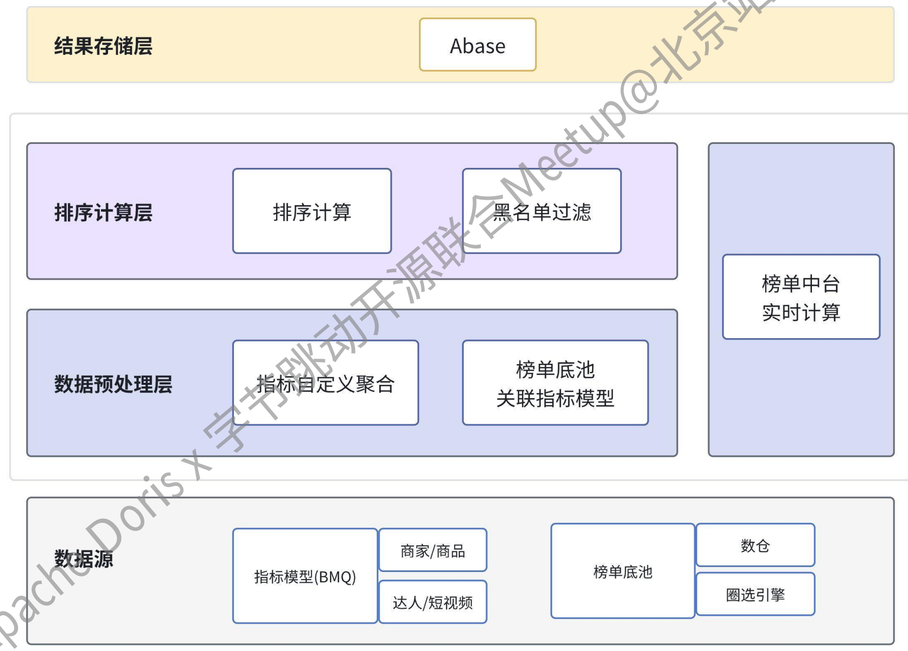
- 多维榜单使用 grouping-sets，数据量放大 N 倍
- 实时写入高

4 组件种类多且数据冗余存储

- 组件: Abase、ByteNDB、ES、CK
- 初期主要用 ES, 无法承载 QPS 的场景，冗余写 ByteNDB

配置化生产一期

- 依赖 Abase 将业务中榜单的公共计算逻辑，抽象成“数据接入” -> “指标预计算” -> “榜单结果排序” -> “结果干预” ->"数据存储"-> “榜单查询服务” 等环节，各个业务不再需要分别单独建设，而是把整个链路工具化和平台化，降低各个业务对于榜单需求的开发成本。

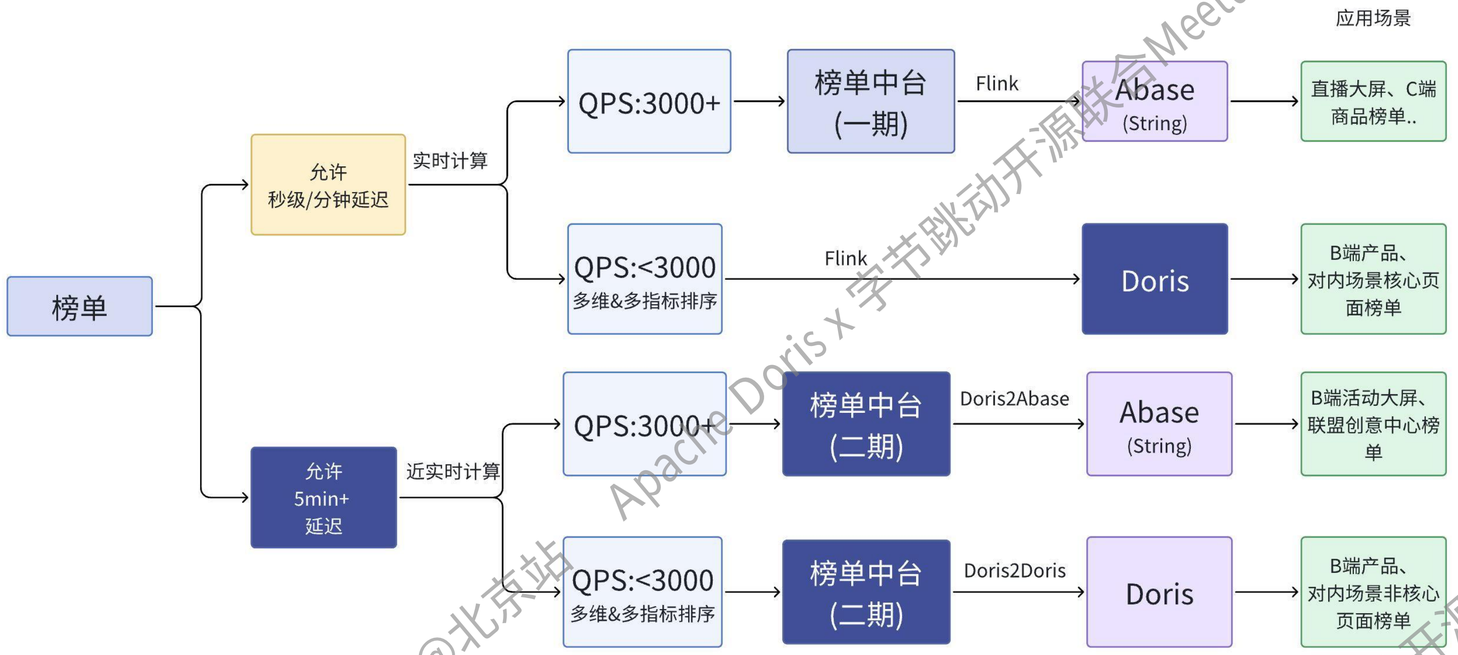


榜单中台-配置化生产一期

新的痛点

- 计算任务数量与榜单数量正比增长，任务量"爆炸"
- 长周期计算，回溯问题

应用实践III-实时榜单及配置化生产



实时榜单及配置化生产升级方案

解决方案

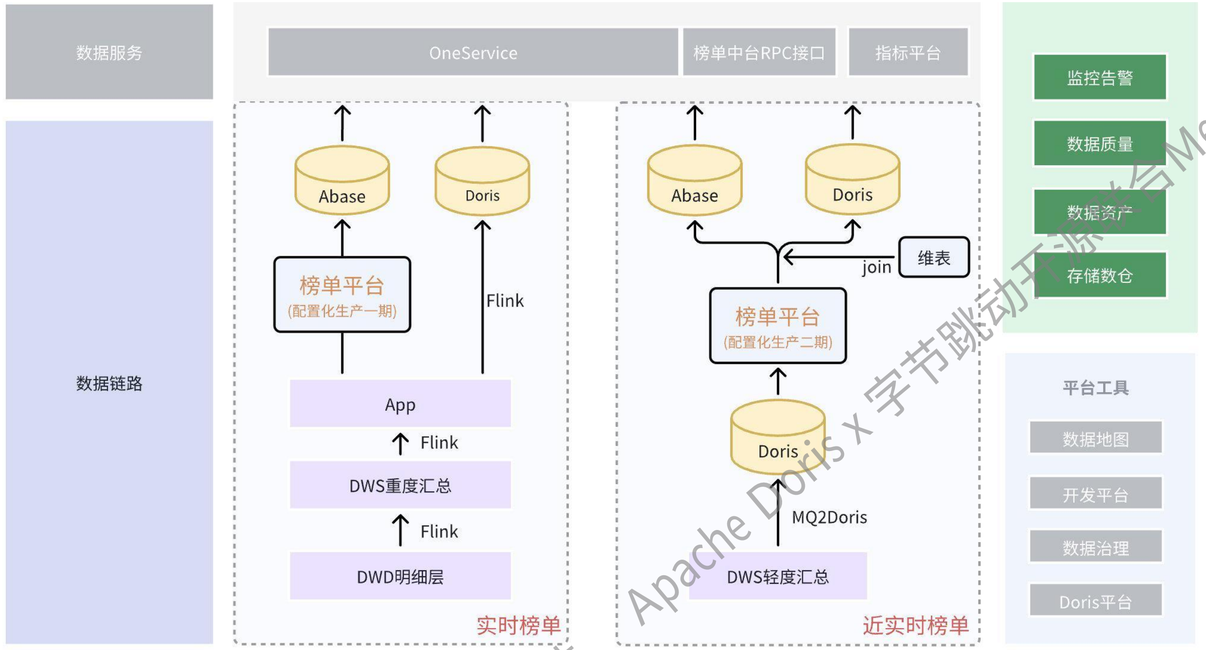
1 引入Doris，服务分析一体化

- 引入Doris，统一存储

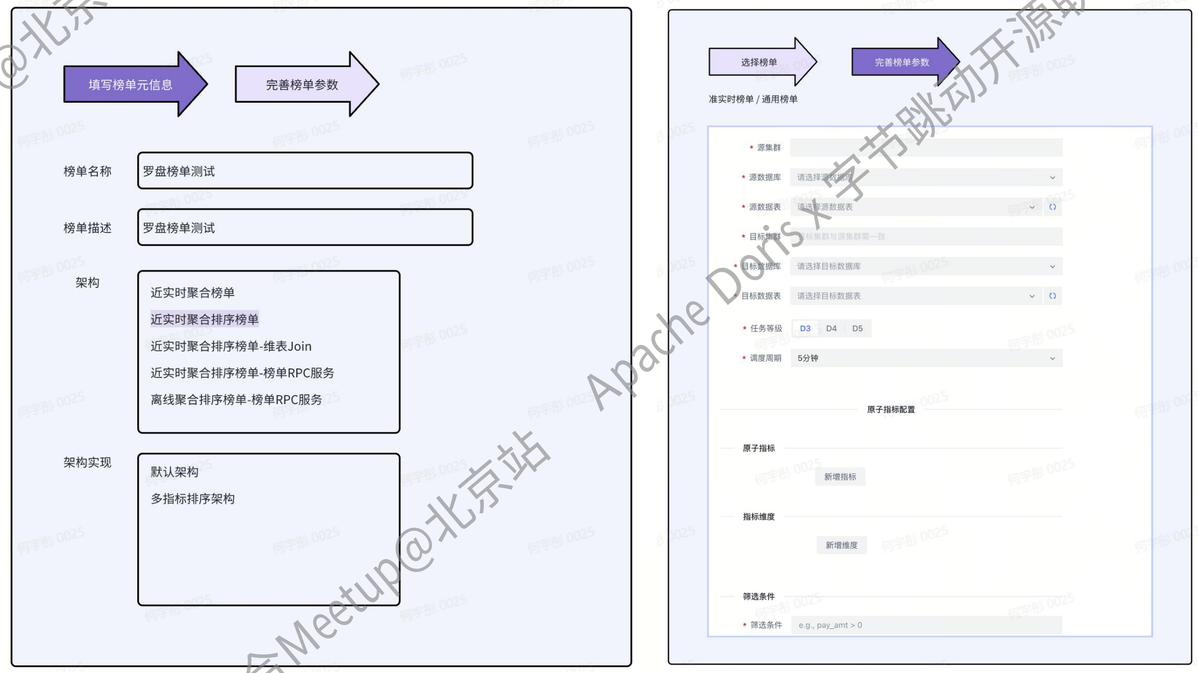
2 近实时榜单改造

- 当实效性要求不高：近实时计算
- Flink 实时榜单->基于 Doris 的准实时榜单

应用实践III-实时榜单及配置化生产



实时榜单及配置化生产系统图



榜单平台配置化生产-页面

解决方案

1 榜单平台配置化生产

- 支持 Abase 类型的近实时榜单: MQ->Doris->Abase
- 支持 Doris 类型的近实时榜单: MQ->Doris->Doris

2 降低开发成本

- 技术架构沉淀到平台，通过架构模板快速实现场景
- 优化参数和经验沉淀到平台
- 业务无关的配置沉淀到平台

应用实践III-实时榜单及配置化生产

--创建临时分区: 保证每次用最新的结果覆盖

```
alter table database_test.realtime_ranking_demo
add TEMPORARY PARTITION if not exists temp_p_${date} VALUES [('${date}'), ('${date+1}'));
```

--数据写入临时分区: insert into xx select xxx from table ...

```
insert into database_test.realtime_ranking_demo TEMPORARY
PARTITION(temp_p_${date})
(rank_type,index_type,date,key_1,key_2,value_1,value_2,property_3_list)
```

--聚合计算: 多维度, 多指标

```
WITH t_base AS (
  SELECT
    1 + IF(key_2 IS NULL, 0, 1) AS index_type,
    date AS date,
    key_1 AS key_1,
    IF(grouping(key_2) = 0, key_2, -1) AS key_2,
    SUM(value_1) AS value_1,
    SUM(value_2) AS value_2,
    GROUP_CONCAT(DISTINCT CAST(property_3 AS VARCHAR), ',') AS property_3_list
  FROM database_test.base_table
  WHERE date = '${date}'
  GROUP BY
    GROUPING SETS(
      (date, key_1),
      (date, key_1, key_2),
    )
),
```

--排序计算: 多维度, 多指标

```
t_value_1 AS (
  SELECT * FROM
  (SELECT
       *,'value_1' AS rank_type,
       ROW_NUMBER() over(
         PARTITION BY date,index_type,key_2
         ORDER BY value_1 DESC
       ) AS rn FROM t_base) t1
  WHERE rn <= 200
),
t_value_2 AS (
  SELECT * FROM (
     SELECT *, 'value_2' AS rank_type,
       ROW_NUMBER() over(
         PARTITION BY date,index_type,key_2
         ORDER BY value_2 DESC
       ) AS rn  FROM t_base) t1
  WHERE rn <= 200
)
select rank_type,index_type,date,key_1,key_2,value_1,value_2,property_3_list  from
t_value_1
union all
select rank_type,index_type,date,key_1,key_2,value_1,value_2,property_3_list from
t_value_2;
```

--正式分区用临时分区数据替换

```
alter table database_test.realtime_ranking_demo
replace partition(p${date}) with TEMPORARY PARTITION(temp_p_${date});
```

配置化生产生成代码示例

应用实践III-实时榜单及配置化生产

结果和收益

榜单开发效率提升

- 基于页面配置化，轻松构建榜单，开发效率提升 50%+
- 引导式的架构选型，各业务可统一榜单建设方式

资源消耗降低

- Flink 预计算链路任务极大减少
- 计算与存储资源降低 60%+

稳定性大幅提升

- 存储组件的写入压力显著降低，负载下降，稳定性提升
- 计算链路写入延迟报警极大的降低

服务分析一体化

- 统一数据存储
- 既能支持高并发、低延迟的在线 Serving，也能支持海量数据的实时复杂分析

目录

应用实践-结果和收益

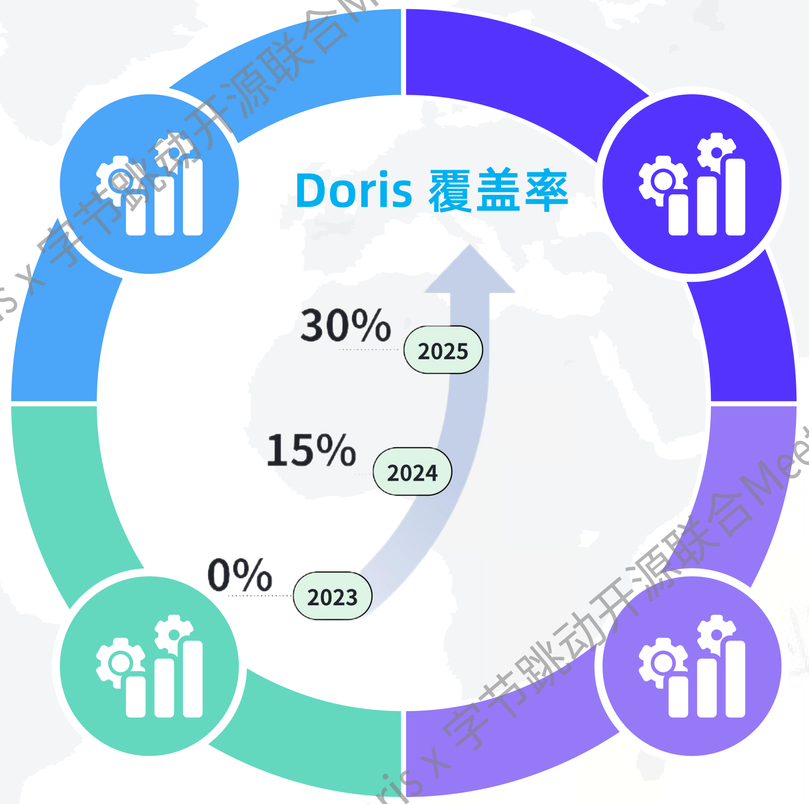
结果和收益

成本降低	数据质量提升	稳定性提升	效率提升
存储和计算成本降低 **80%+**	单季度数据质量case>3 降低至**< 1**	核心任务报警3 ~ 5/周 降低至 **< 1次/月**	开发和运维效率提升 **50%+**

未来展望

持续架构演进

- 基于 Doris 的近实时架构
- 基于 Doris 2.x 的服务分析一体化
- Doris 与数据湖融合的湖仓一体解决方案



业务与引擎侧密切合作

- 新功能推进：如 Doris Binlog、Doris SLA、跨集群 ETL 等
- 业务场景性能优化

Doris稳定性治理与保障

- 100%升级稳定版本
- 规范与治理
- 危险SQL封禁
- 慢查询治理
- 资源隔离

业务场景应用推广

- 近实时场景推广上量
- 服务端分析一体化场景应用推广

Thanks！