

# Apache Doris 在 物业城市的场景实践

王凯

万物云 大数据研发专家

# 目录

1. 业务背景
2. 架构演进
3. 实践应用
4. 总结展望

# 1 业务背景



## 公司介绍

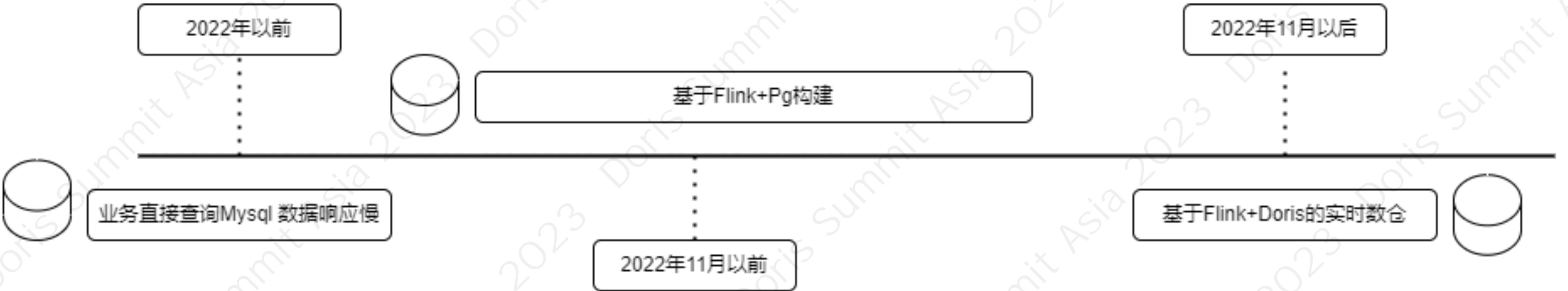


物业城市致力于城市空间智慧运营服务，整合物联网、云平台、大数据等多重力量，合力构建新型数字空间底座，在遵循市场化的原则下，通过数字化、机械化、专业化的运营手段打通孪生的数字空间与线下实体空间的联结。通过线上线下运营，智慧化地服务包括站城一体化、产城融合等多业态在内的城市复合空间，极大提升城市治理效率。目前物业城市的实践探索已经包括横琴粤澳深度合作区【放管服改革后，解决人少事多】、成都高新区【品质逐年提升 成本逐年下降】、厦门鼓浪屿【碎片化整合，高效应急处突】、武汉市江汉区【街区+社区，探索老旧小区长效运营】、深圳沙头街道【城中村共建共治】等。

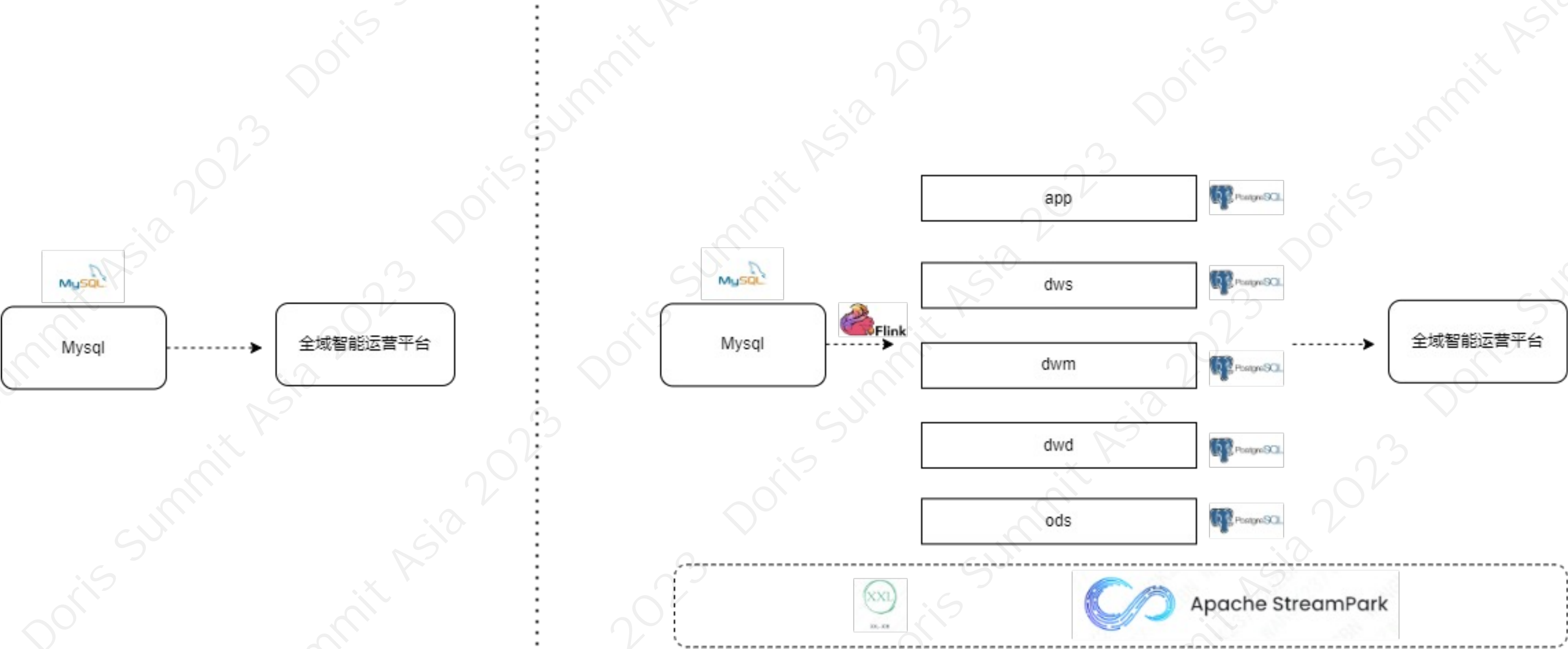


## 2 架构演进

# 架构演进



# 架构演进



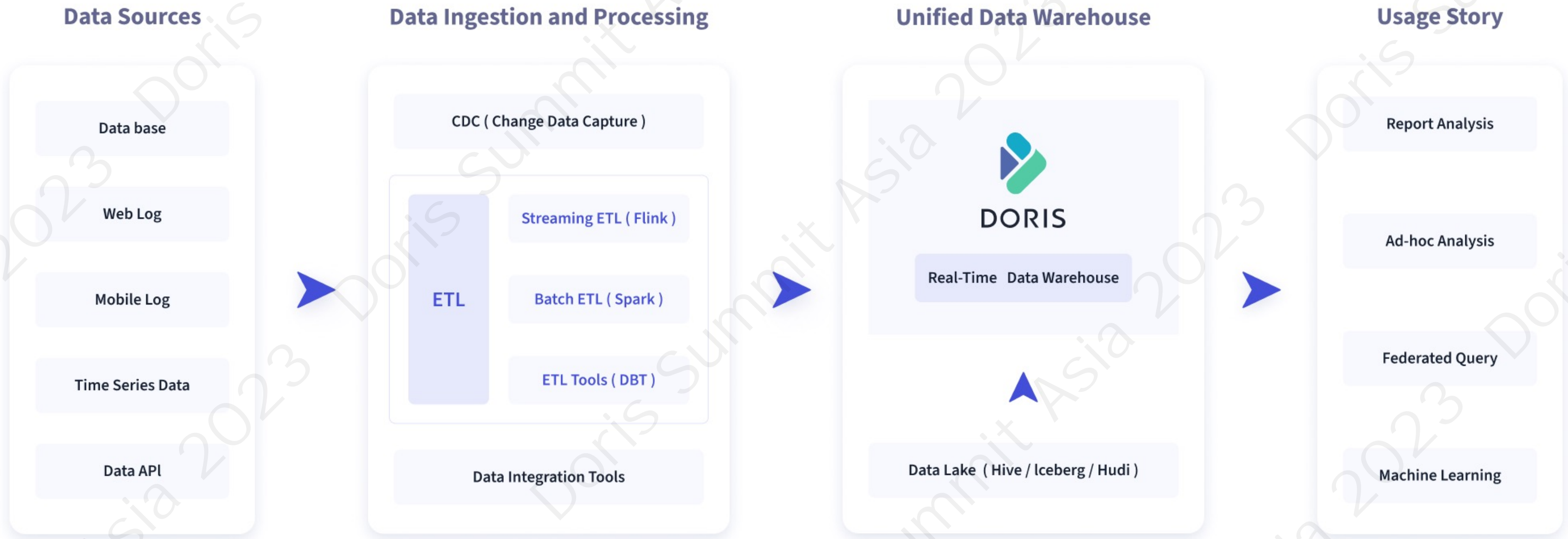
## 架构演进

### 旧有架构存在如下问题：

- 旧有架构部署成本高，查询响应慢
- 维护组件多，运营成本高
- 数据没有明确分层，很多开发都是烟囱式开发，计算口径不统一

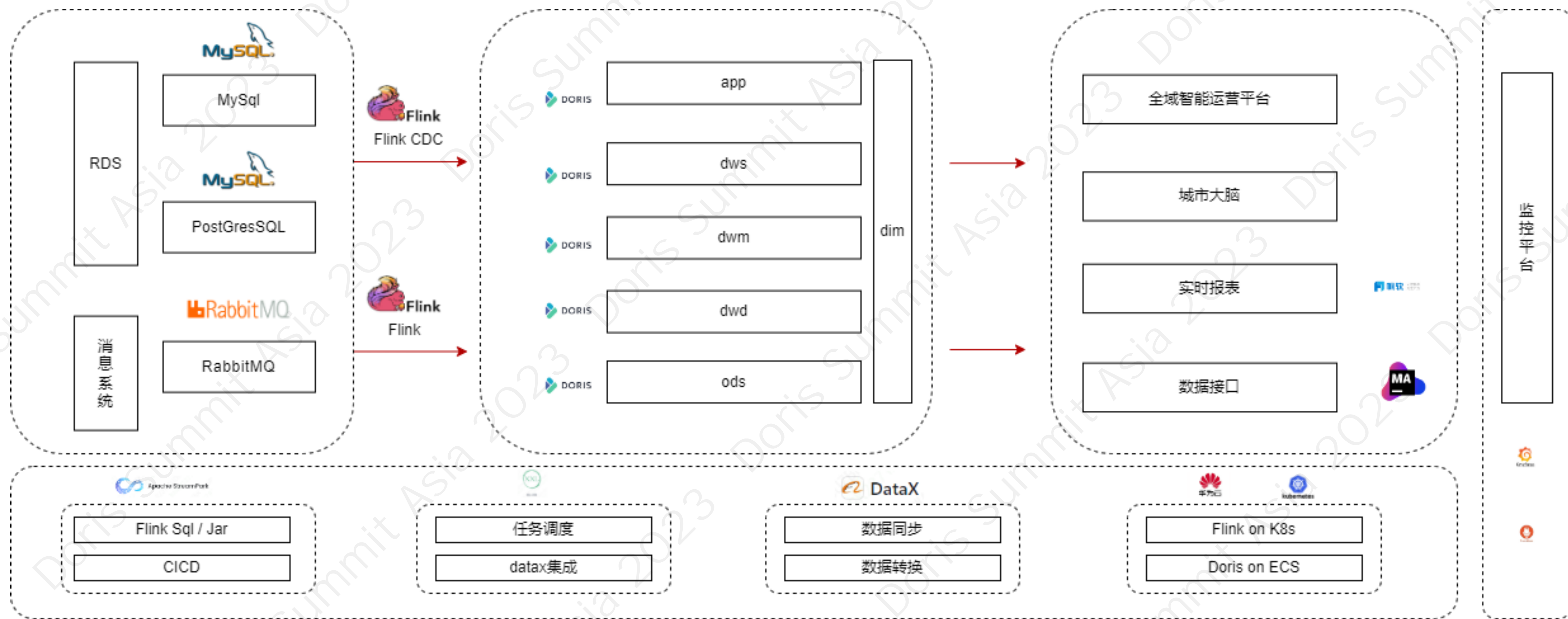


# 架构演进



- 实时看板（Dashboards），符合物业城市大脑实时大屏展示的需求
- 面向企业内部分析师和管理者的报表，符合物业城市实时报表的需求
- 即席查询（Ad-hoc Query）
- 统一数仓构建
- 数据湖联邦查询
- 极简运维，只需要维护FE BE两个进程，不需要引入额外组件

## 架构演进-现有架构

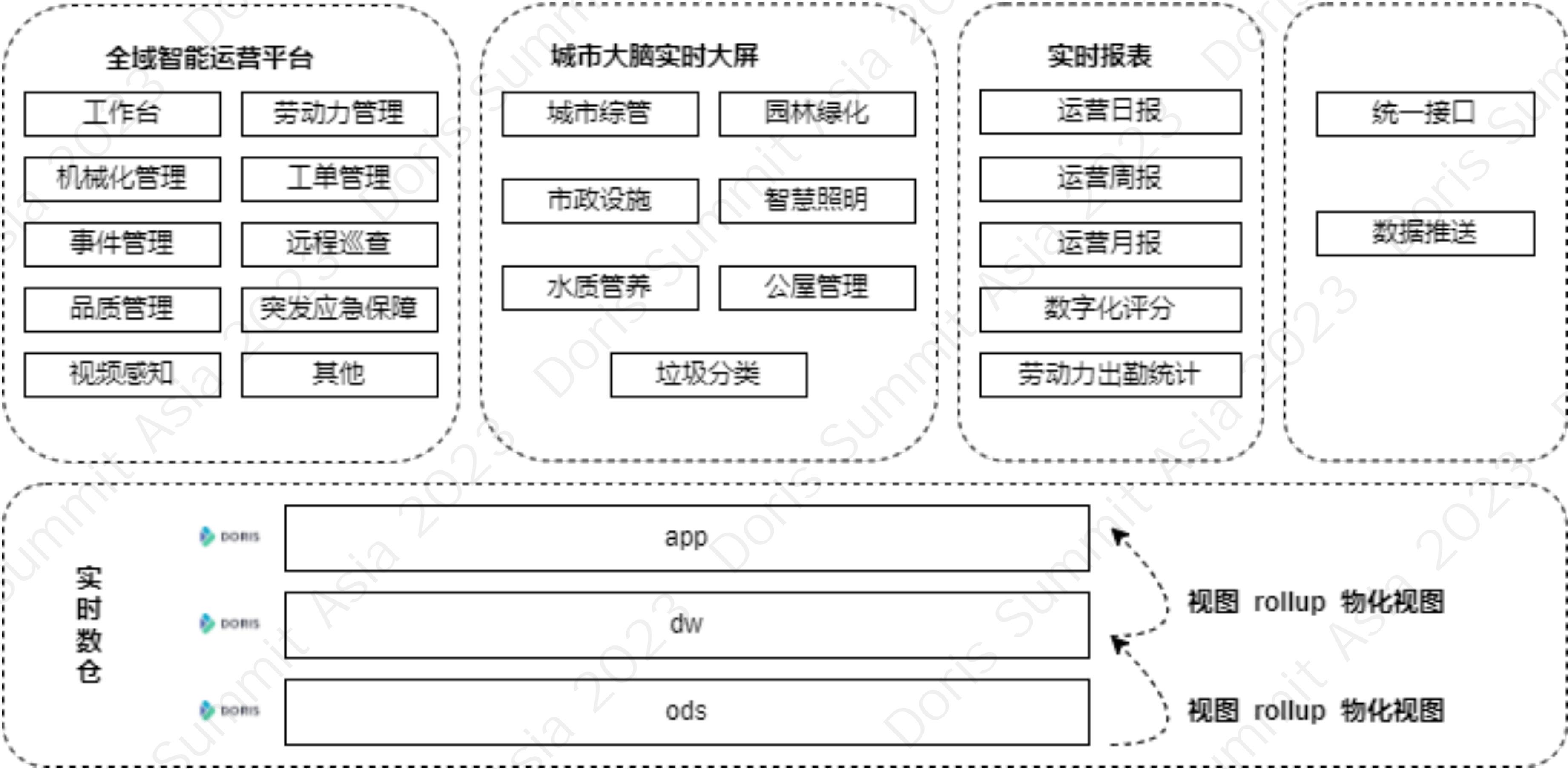


## 架构演进-基于 Doris 的实时数仓的改进

- 基于上述架构构建了以 Doris 为核心的物业城市实时数据底座，极大的提高了数据时效性，数据可见性从原来的 T+15min，缩短到了毫秒级
- 全域智能运营平台车辆轨迹数据从原来的数据库 3 个月可见到现在的历史可见，查询速度成倍提升
- 全域智能运营平台工单数据报表实时性得到了成倍提升
- 构建了基于 Doris 的城市大脑
- 基于 Doris 构建的实时数仓，由于不需要额外维护其他组件，极大降低了维护成本



# 基于 Doris 的全域智能运营平台



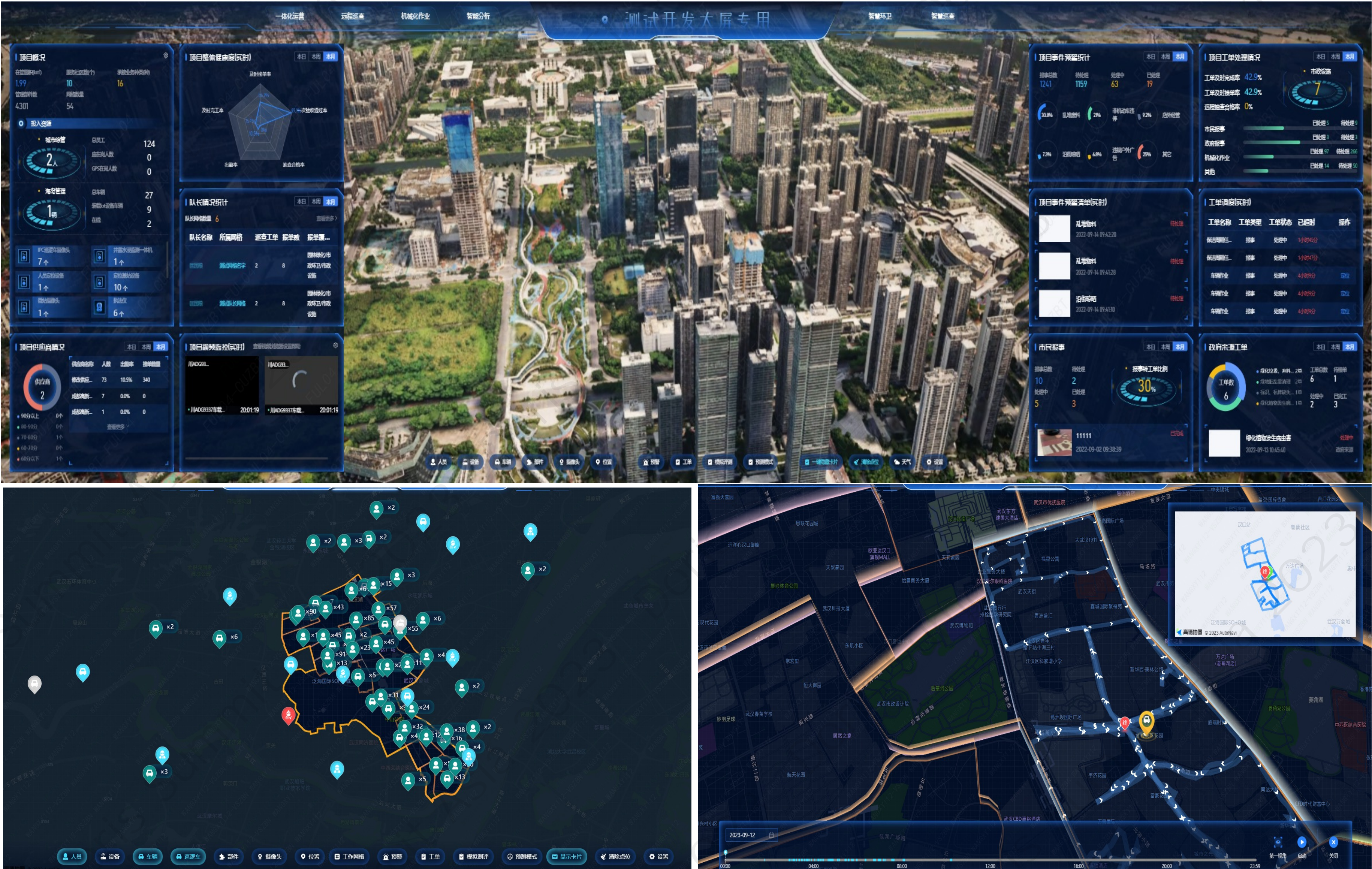
## 基于 Doris 的实时数仓的优势

- 基于 Doris 的实时数仓构建的数据底座，支撑了全域智能运营平台，城市大脑实时大屏，实时报表以及统一对外数据接口、数据推送平台，目前核心数据服务基本都迁移到了 Doris
- 工单、车辆轨迹等业务，以前线上服务几乎不可用，现在线上服务可以达到秒级甚至毫秒级响应，极大提升了用户体验
- 由于 Doris 只需要维护 FE 和 BE，不需要依赖额外组件，运维非常高效
- Doris支持 Rollup、物化视图，基于物化视图构建的实时数仓，不仅可以对原始明细数据的任意维度进行分析，也能快速的对固定维度进行分析查询

# 3 实践应用

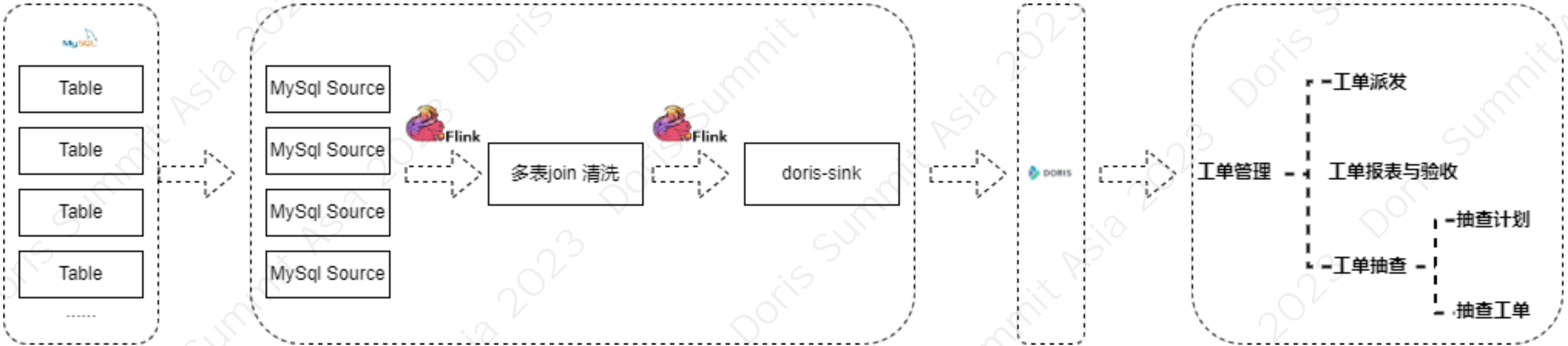


实时大屏





# 工单



# 工单建表优化

```
1 CREATE TABLE if not exists dwd.dwd_ (
2 )
3 ENGINE=OLAP
4 UNIQUE KEY(plan_start_id)
5 PARTITION BY RANGE(plan_time) (
6 DISTRIBUTED BY HASH(plan_start_time, id) BUCKETS 4
7 PROPERTIES (
8   "replication_allocation" = "tag.location.default: 3",
9   "compression"="zstd",
10  "light_schema_change" = "true",
11  "enable_unique_key_merge_on_write" = "true",
12  "bloom_filter_columns" = "id, tririga_no, project_id, issue_type, biz_source, issue_status_value",
13  "dynamic_partition.time_zone" = "Asia/Shanghai",
14  "dynamic_partition.enable" = "true",
15  "dynamic_partition.time_unit" = "MONTH",
16  "dynamic_partition.end" = "24",
17  "dynamic_partition.create_history_partition" = "true",
18  "dynamic_partition.history_partition_num" = "36",
19  "dynamic_partition.prefix" = "p_",
20  "dynamic_partition.buckets" = "4"
21 );
22 CREATE INDEX IF NOT EXISTS id ON dwd.dwd_ (id) USING BITMAP COMMENT 'id';
23 CREATE INDEX IF NOT EXISTS tririga_no ON dwd.dwd_ (tririga_no) USING BITMAP COMMENT 'tririga_no';
24 CREATE INDEX IF NOT EXISTS project_id ON dwd.dwd_ (project_id) USING BITMAP COMMENT 'project_id';
25 CREATE INDEX IF NOT EXISTS issue_type ON dwd.dwd_ (issue_type) USING BITMAP COMMENT 'issue_type';
26 CREATE INDEX IF NOT EXISTS biz_source ON dwd.dwd_ (biz_source) USING BITMAP COMMENT 'biz_source';
27 CREATE INDEX IF NOT EXISTS issue_status_value ON dwd.dwd_ (issue_status_value) USING BITMAP COMMENT 'issue_status_value';
28 CREATE INDEX IF NOT EXISTS etl_timestamp ON dwd.dwd_ (etl_timestamp) USING BITMAP COMMENT 'etl_timestamp';
```

- 工单表创建使用基于时间的按月动态分区
- 开启了写时合并，消除掉了读时合并中的数据聚合过程
- 新增 BloomFilter 索引，BITMAP索引，加速查询速度
- 基于上述优化，工单的查询速度较之前提升了 5 倍左右



## Doris 优化

Doris 在使用一段时间后，会出现大量慢查询，导致生产不可用，通过排查以后发现如下问题：

- 建表分区、分桶不合理，很多表只使用了 Doris 默认索引，而没有使用 BITMAP 等索引来加速查询
- 业务方 SQL 不规范，出现大量 `select *` 写法
- 数仓分层基本很多依赖视图，导致谓词不能下推
- 使用 API 写入报错

## Doris 优化

基于上述问题，在使用过程中做了如下优化：

- 增加 BITMAP 索引，加速查询
- 设置合理的分区分桶
- 优化 SQL 写法
- 增加慢查询监控
- 增加 fe be节点自动拉起
- 将部分多表视图放到 Flink 中计算后写入 Doris，直接基于 Doris 查询
- Doris 监控及预警，及时发现doris的慢查询等问题

## Doris 优化

Flink CDC 使用 API 写入数据报错，与社区同学沟通，判断是 Doris 1.2.1 连接器 bug，需要在字段映射里面新增隐藏的逻辑删除标识

```
tableEnv.createTemporaryView("status", processStream);
String sql = "select license_plate_number,offline_start_time,offline_end_time,project_id,project_name,car_id,status,longitude,latitude,0 as __DORIS_DELETE_SIGN__ from status where " +
    "date_format(offline_end_time, 'yyyy-MM-dd') between FROM_UNIXTIME(UNIX_TIMESTAMP(cast(CURRENT_DATE as string), 'yyyy-MM-dd') - 220 * 24 * 3600, 'yyyy-MM-dd') and CURRENT_DATE";
Table sqlResult = tableEnv.sqlQuery(sql);

String[] fields = {"license_plate_number", "offline_start_time", "offline_end_time", "project_id", "project_name", "car_id", "status", "longitude", "latitude", "__DORIS_DELETE_SIGN__"};
DataType[] types = {DataTypes.STRING(), DataTypes.TIMESTAMP(), DataTypes.TIMESTAMP(), DataTypes.BIGINT(), DataTypes.STRING(), DataTypes.BIGINT(), DataTypes.INT(), DataTypes.DOUBLE(), DataTypes.DOUBLE(), DataTypes.INT()};
DataStream<RowData> rowDataStream = tableEnv.toAppendStream(sqlResult, RowData.class);
```



## Doris 数据备份

Doris 数据备份到华为云

```
CREATE REPOSITORY `doris_bak`  
WITH S3  
ON LOCATION "s3://d[REDACTED]"  
PROPERTIES  
(  
  "AWS_ENDPOINT" = "[REDACTED]",  
  "AWS_ACCESS_KEY" = "[REDACTED]",  
  "AWS_SECRET_KEY" = "[REDACTED]",  
  "AWS_REGION" = "[REDACTED]"  
);
```

```
BACKUP SNAPSHOT ods.ods_snapshot_20230831  
TO doris_bak
```

## 4 总结展望

## 总结展望

升级 Doris 2.0+

- 全新查询优化器
- 倒排索引
- 点查询并发能力提升 20 倍
- 自适应的并行执行模型
- 高效的数据更新





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](https://doris.apache.org)
- Apache Doris GitHub: [github.com/apache/doris/](https://github.com/apache/doris/)

### 获取更多峰会资料:

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