

# 哪吒科技: SelectDB 实时数仓在智慧港口中的应用实践

邓宇超

哪吒科技 数据平台负责人

## 分享嘉宾 - 哪吒科技



**邓宇超**

哪吒科技-数据平台负责人

曾就职于趣头条、观安信息

丰富的大数据开发及架构经验

# 目录

01 哪吒科技数仓发展历程

02 实时数仓技术选型与架构

03 实时数仓在智慧港口的实战应用

04 收益与展望

01

# 哪吒科技数仓发展历程









## 1.2 哪吒科技业务介绍-产品&服务

### 咨询服务

顶层设计  
细节设计

### 项目实施

项目管理  
软件部署  
硬件集成

### 运维服务

运维中心

### 成熟产品

SMART系列

全生命周期

全解决方案





## 1.3 哪吒科技数仓发展历程

### 第一阶段

业务场景：港口运营数据枢纽

数仓能力：单业务单场景数据整合与决策赋能

挑战：数据融合性、实时性

### 第二阶段

业务场景：数据驱动码头作业

数仓能力：混合架构下数据整合与分析赋能

挑战：开发效率、维护成本、实时性

### 第三阶段

业务场景：智慧港口全生命周期与全解决方案（SMART 系列）

数仓能力：SelectDB 统一实时数仓

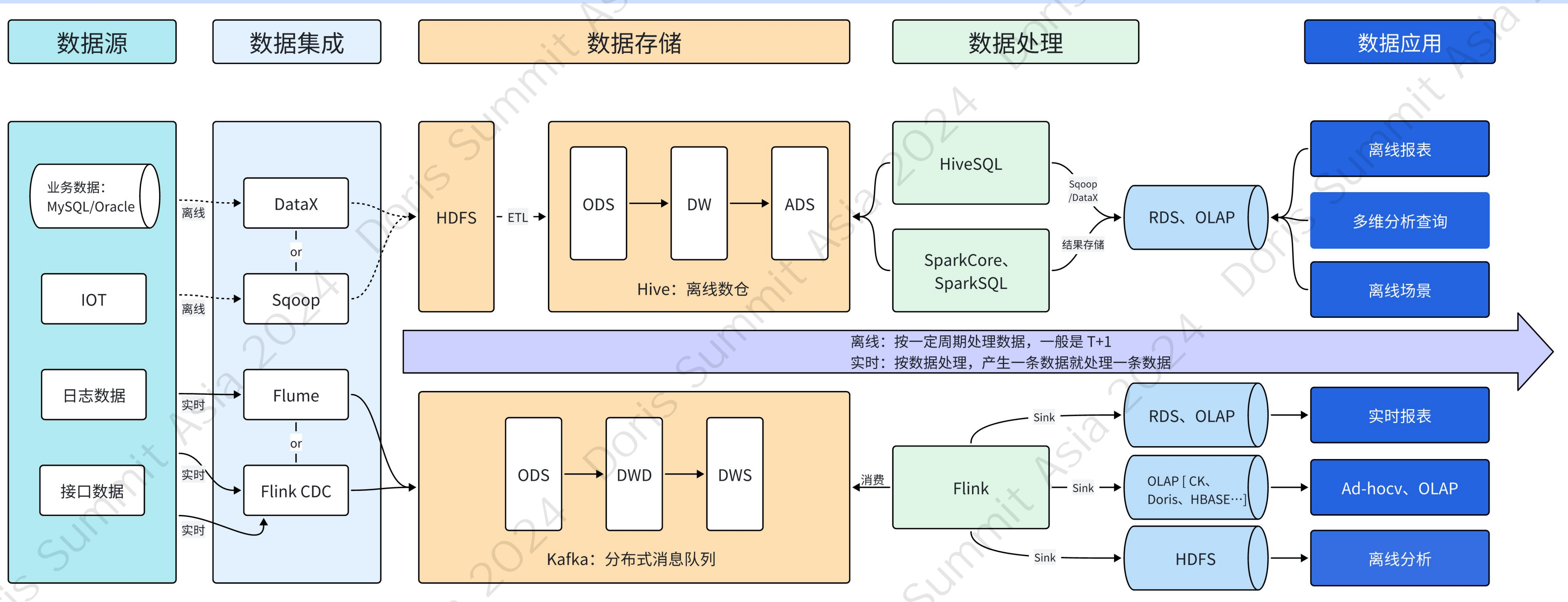
挑战：复杂场景、查询效率

## 1.4 哪吒科技数仓架构演变（V1.0）





# 1.5 哪吒科技数仓架构演变（V2.0）



# 1.6 哪吒数仓应用场景面临的挑战

应用场景	时效性要求	数据特点	面临的问题
RTG 效率分析（堆场机械效率分析）	离线跑批（H+1）	数据维度复杂，下钻层次多	存在异常数据，数据不准，任务资源占用高
某码头交接班分析	离线跑批（H+1）	数据维度比较单一，数据量少	数据量不大的情况下，单个任务资源占用高
集疏运数字孪生	实时性要求高，数据产生到落地在1s内	上海港8大码头数据量大，单表数据量10亿以上，表字段多（100+）	实时任务性能瓶颈问题，时效性无法满足

02

# 实时数仓技术选型与架构

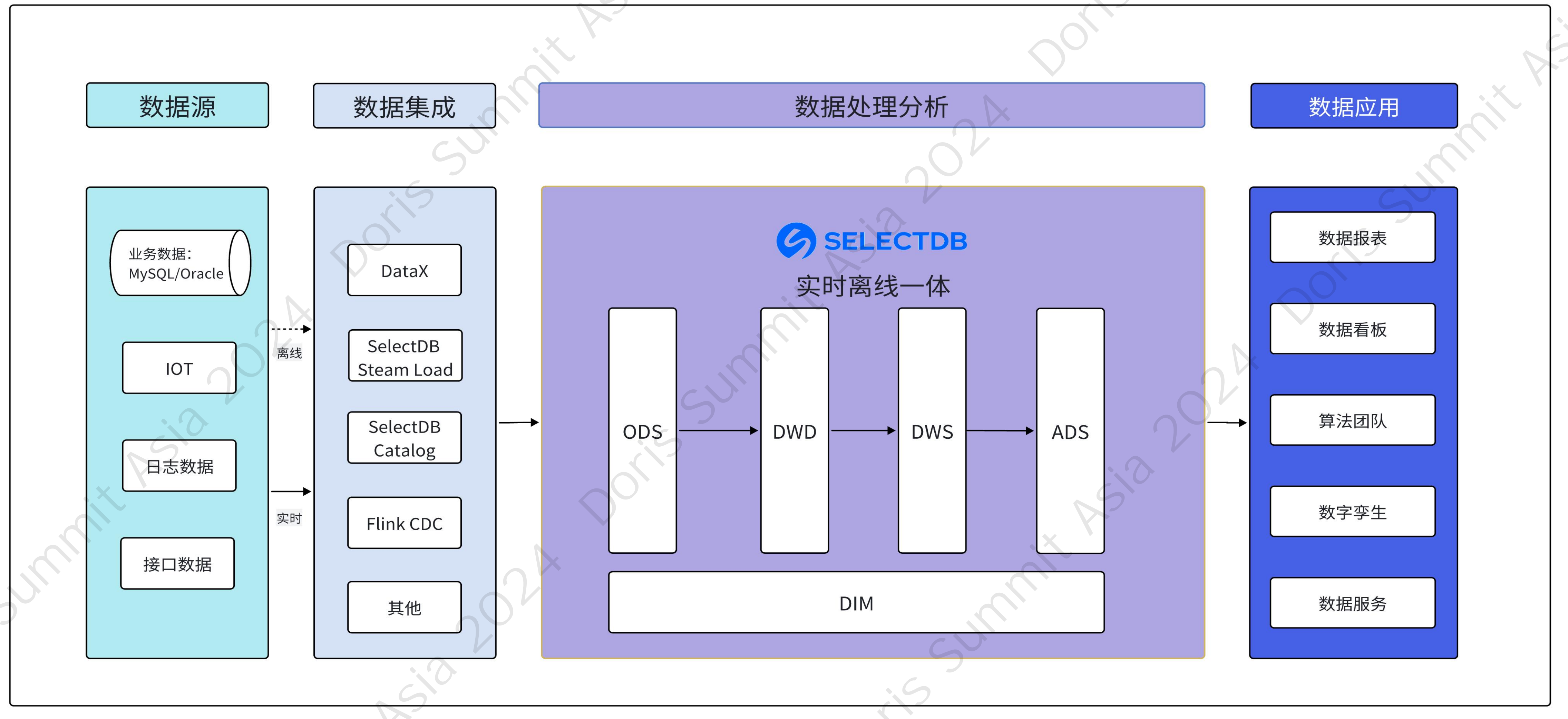


## 2.1 哪吒实时数仓选型因素

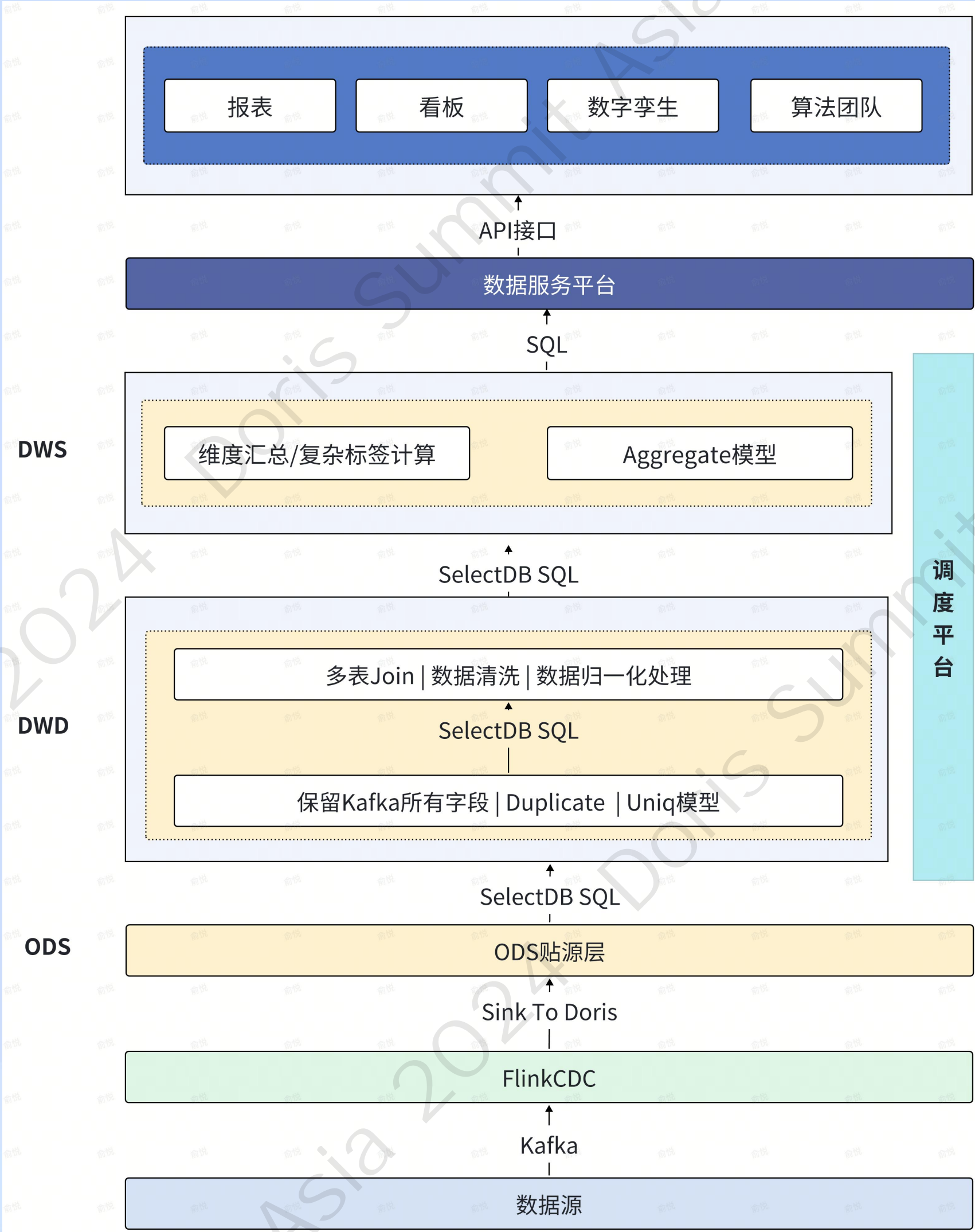
关键指标	SelectDB
数据导入	秒级数据导入，支持多种数据导入方式（Stream Load、Bocker Load、Routine Load），毫秒级轻量化表模式修改
查询性能	支持高并发点查、大宽表查询、多表 Join 复杂查询、数据湖查询，查询优化（向量化、物化视图、CBO优化器、丰富的索引）
弹性架构	支持计算隔离，分层存储，存算分离三种模式，实现高效灵活的资源管理
开放生态	基于 Apache Doris 构建，100% 与其兼容，兼容 MySQL 生态
安全合规	深度契合等保要求的各项指标，并严格遵循 ISO 标准的规范框架
自主可控、安全可靠	融合信创技术，构建多重安全防线，以高可靠性能，为数据构建安全防线



## 2.2 哪吒科技实时数仓架构



# 2.4 SelectDB在实时场景中的应用



## 实时数仓建模:

1. ODS: 通过 FlinkCDC 实时采集数据源的数据，落地到 SelectDB 实时数仓，为后续数据分析提供原始数据。
2. DWD: 基于清洗后的明细数据，做数据归一化处理，方便上游业务。
3. DWS: 基于聚合模型，对指标预聚合，供业务使用。

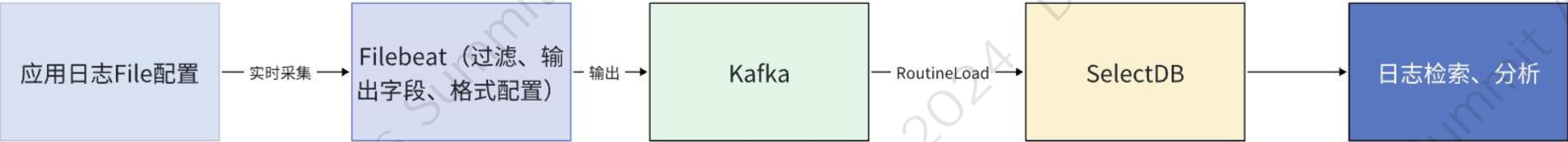
## 业务应用:

通过数据服务平台，提供查询 API 接口，赋能给上层业务使用。



# 2.5 SelectDB在日志收集分析中的运用-架构及效果

## 整体架构



2 倍

查询性能提升

- SQL 查询语言，支持 Join，相比 DSL简单，表达能力强
- 高效的执行引擎和优化器，高效的索引与文本分析

5 倍

写入吞吐提升

- 向量化指令，提升数据解析，索引构建的性能
- 简化去掉正排等索引结构，降低索引构建开销

80%

存储成本降低

- 简化去掉正排等索引结构，减少倒排数据量 30%
- 列式存储和ZSTD高效压缩，提供 5-10 倍压缩比
- 冷热分层，大幅降低冷数据成本

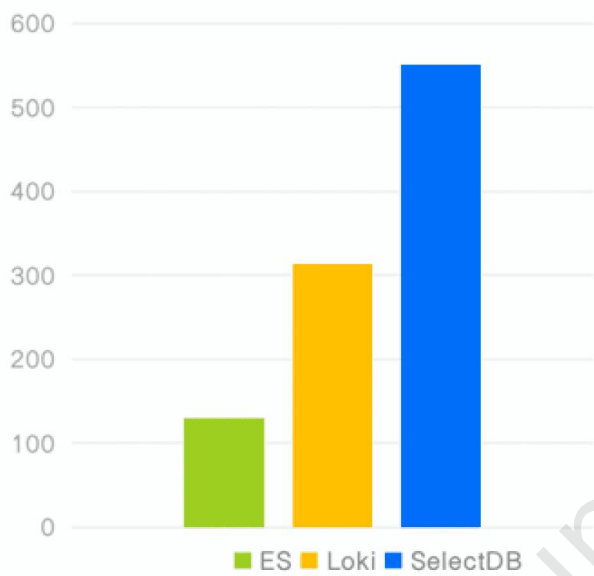
稳定性

大幅提升

- 资源隔离和大查询限制
- 高效内存管理，避免 Java GC 影响

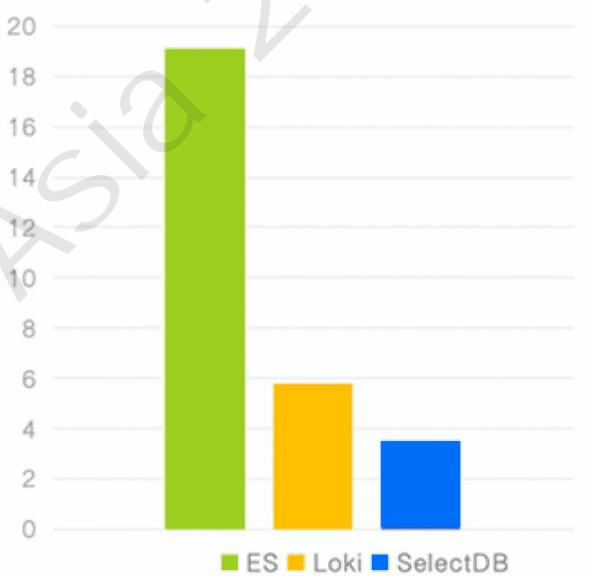
4.2倍

写入速度(MB/s)



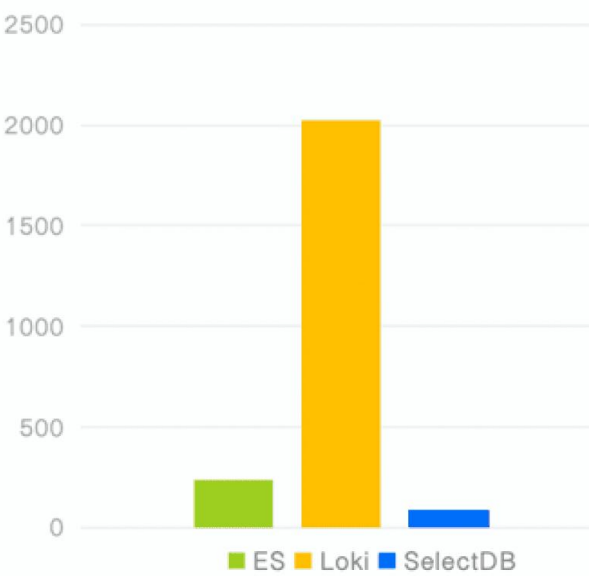
5.4倍

存储空间(GB)



2.3倍

查询时间(秒)



## 2.6 SelectDB案例-日志检索分析-建表示例

```
CREATE TABLE XXX_LOGS
(
  ts DATETIME,
  path TEXT,
  message TEXT,
  hostname VARCHAR(30),
  ip ARRAY<VARCHAR(20)>,
  INDEX idx_path (path) USING INVERTED,
  INDEX idx_host (hostname) USING INVERTED,
  INDEX idx_msg (message) USING INVERTED PROPERTIES("parser" = "unicode")
)
ENGINE = OLAP
DUPLICATE KEY(ts)
PARTITION BY RANGE(ts) ()
DISTRIBUTED BY RANDOM BUCKETS AUTO
PROPERTIES (
  "compression"="zstd",
  "compaction_policy" = "time_series",
  "dynamic_partition.enable" = "true",
  "dynamic_partition.create_history_partition" = "true",
  "dynamic_partition.time_unit" = "DAY",
  "dynamic_partition.start" = "-7",
  "dynamic_partition.end" = "3",
  "dynamic_partition.prefix" = "p"
);
```



## 2.6 SelectDB案例-日志检索分析-load 示例

```
CREATE ROUTINE LOAD database.test_job ON XXX_LOGS
  COLUMNS(ts, path, message, hostname, ip)
  PROPERTIES
  (
    "desired_concurrent_number"="1",
    "format" = "json",
    "strict_mode" = "false",
    "jsonpaths" =
      "[\"$.@timestamp\", \"$.log.file.path\", \"$.message\", \"$.host.hostname\", \"$.host.ip\"]"
  )
  FROM KAFKA
  (
    "kafka_broker_list" = "ip:9092",
    "kafka_topic" = "flink_logs",
    "property.group.id" = "t1_g",
    "property.kafka_default_offsets" = "OFFSET_BEGINNING"
  );
```

### 注意点:

#### (1) Doris 建表语句

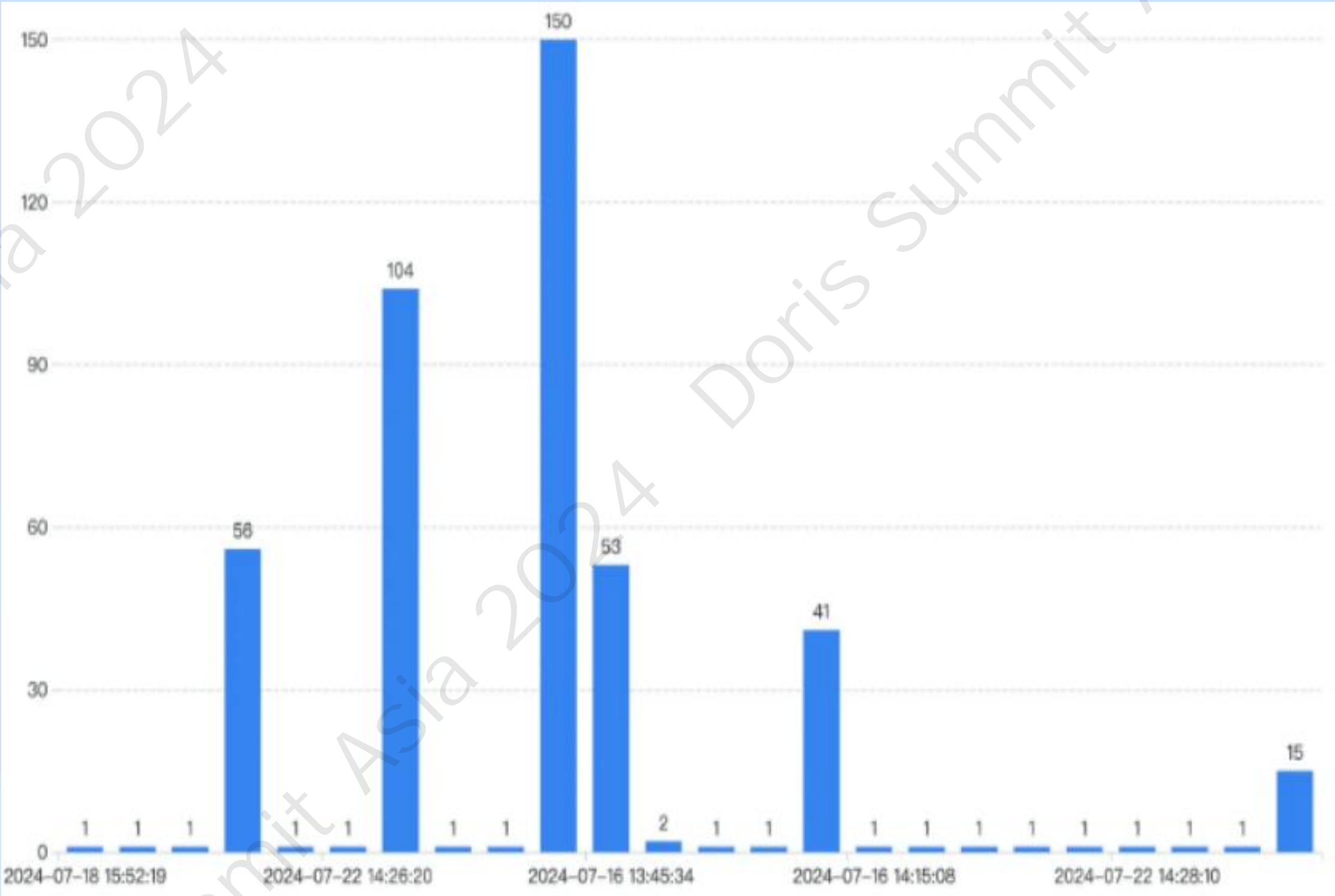
- 当使用 DATETIME 类型的时间字段作为主键 Key 时，查询最新 n 条日志的速度会得到显著提升。
- 使用基于时间字段的 RANGE 分区，并开启动态 Partiiton，以便按天自动管理分区，提升数据查询和管理的灵活性。
- 在分桶策略上，可以使用 RANDOM 进行随机分桶，分桶数量大致设置为集群磁盘总数的 3 倍。
- 对于经常需要查询的字段，建议构建索引以提高查询效率；而对于需要进行全文检索的字段，应指定合适的分词器参数 parser，确保检索的准确性和效率。
- 采用 ZSTD 压缩，可以获得更好的压缩效果，节省存储空间。
- 对需要全文检索的字段，将分词器（parser）参数赋值为 unicode，如有支持短语查询的需求，将 support\_phrase 参数赋值为 true；如不需要，则设置为 false，以降低存储空间。

#### (2) 导入语句

- 需要针对 filebeat 采集的 json 格式数据，做正确解析，此处与 Doris 建表字段类型要一致。

# 2.7 SelectDB案例-日志检索分析-实现效果

	timestamp	path	message
1	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:18:03,600 ERROR io.debezium.c
2	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:18:00,804 ERROR io.debezium.p
3	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:18:00,804 ERROR io.debezium.c
4	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:58,059 ERROR io.debezium.p
5	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:58,059 ERROR io.debezium.c
6	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:55,296 ERROR io.debezium.p
7	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:55,296 ERROR io.debezium.c
8	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:52,555 ERROR io.debezium.p
9	2024-07-18 15:50:22	/opt/module/flink-1.18.1/log/flink-root-taskexecutor-12-nz-cdh-master-02.log	2024-07-17 11:17:52,555 ERROR io.debezium.c





03

# 实时数仓在智慧港口的实战运用

# 3.1 SelectDB指标计算实践



## 业务场景（场地内的件散货货物作业动态数据）

智慧码头运营管理需多维度统计场地货物作业。对钢材类，筛选作业量超 500 吨数据，剖析其大规模作业态势及资源影响；对全场货物，统计进货 1 天内作业总量，把握流转与繁忙程度，助力计划调整；针对钢材进货 1 天内作业量大于 500 吨状况，据此制定专属策略，提升特定货物与时段的管理及资源利用效能，实现高效运营决策。

统计指标名称	指标描述
钢材大规模作业数据统计	聚焦于钢材类货物，精准筛选出作业量超过 500 吨的相关数据记录，分析其作业态势及对码头资源占用与作业效率的影响。
全场货物短期作业量汇总	针对场地所有货物，详细统计自进货起 1 天内的作业量总和，以掌握货物短期内流转速度与作业繁忙程度，为短期作业计划调整提供依据。
钢材特殊高效作业数据统计	着重于钢材类货物，精确统计其进货后 1 天内作业量大于 500 吨的情形，为制定钢材专属作业方案与资源配置策略提供数据支持，提升特定货物类别及作业时段运营管理效能与资源利用效率。



## 3.2 SelectDB指标计算实践-示例代码（表结构）

-- 场地货动态表

```
CREATE TABLE `TMP_ACTIVITIES` (  
  `g_id` varchar(32) NOT NULL COMMENT '动态id',  
  `g_w_id` string NOT NULL COMMENT '场地货id',  
  `g_gtwg` DECIMAL(13, 3) NOT NULL DEFAULT "0" COMMENT '重量',  
  `g_opdate` datetime NOT NULL COMMENT '作业日期',  
  `t_id` string NULL COMMENT '租户id', ...  
) ENGINE=OLAP  
UNIQUE KEY(`g_id`)  
COMMENT '场地货动态'  
DISTRIBUTED BY HASH(`g_id`) BUCKETS AUTO  
PROPERTIES ("enable_unique_key_merge_on_write" = "true", "store_row_column" = "true", ...);
```

-- 场地货表

```
CREATE TABLE `TMP_GOODS` (  
  `w_id` varchar(32) NOT NULL COMMENT 'id',  
  `w_gname` string NOT NULL COMMENT '货名',  
  `w_in_date` datetime NULL COMMENT '进货日期',  
  `t_id` string NULL COMMENT '租户id', ...  
) ENGINE=OLAP  
UNIQUE KEY(`w_id`)  
COMMENT '场地货'  
DISTRIBUTED BY HASH(`w_id`) BUCKETS AUTO  
PROPERTIES ("enable_unique_key_merge_on_write" = "true", "store_row_column" = "true", ...);
```

## 3.3 SelectDB指标计算实践-示例代码（指标计算）

--统计场地货货类为钢材且作业量大于500吨的情况

```
select w_id,sum(g_gtwg) from TMP_ACTIVITIES goa
join TMP_GOODS wyg on goa.g_w_id =wyg.w_id and wyg.t_id=goa.t_id
where w_gname='钢材'
group by w_id
having sum(g_gtwg)>5;
```

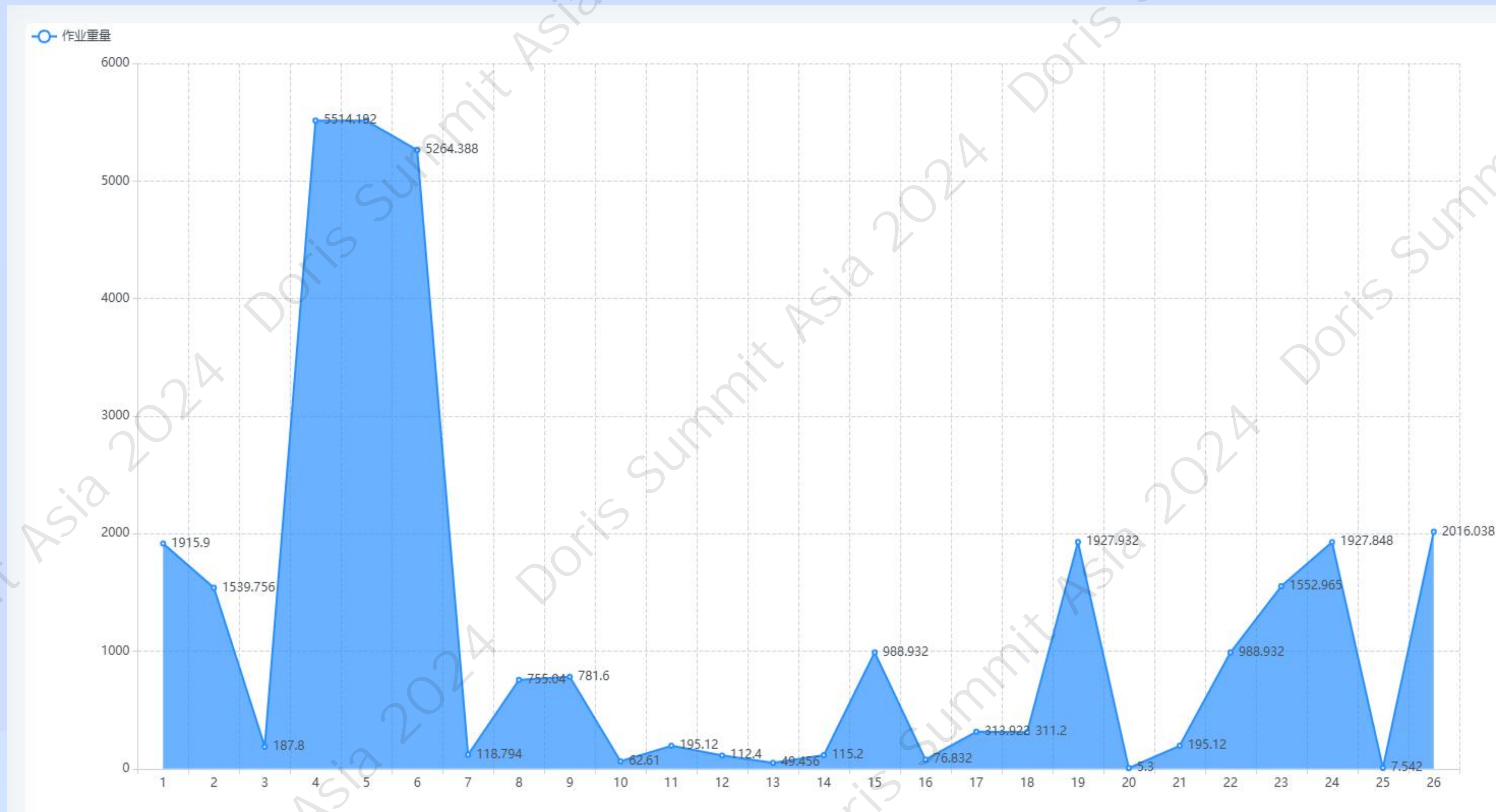
--统计场地货从进货开始1天内的作业量

```
select w_id,sum(g_gtwg) from TMP_ACTIVITIES goa
join TMP_GOODS wyg on goa.g_w_id =wyg.w_id and wyg.t_id=goa.t_id
where g_odate between w_in_date and date_add(w_in_date,interval 1 day)
group by w_id;
```

--统计场地货货类为钢材且从进货开始1天内作业量大于500吨的情况

```
select w_id,sum(g_gtwg) from TMP_ACTIVITIES goa
join TMP_GOODS wyg on goa.g_w_id =wyg.w_id and wyg.t_id=goa.t_id
where w_gname='钢材' and g_odate between w_in_date and date_add(w_in_date,interval 1 day)
group by w_id
having sum(g_gtwg)>5;
```

### 3.4 SelectDB指标计算实践-效果展示





## 3.5 SelectDB物化视图实践-示例代码

### 场景：

码头多用户同时对单表（数据量十亿级）进行维度聚合查询。通过不同维度组合和聚合方式，对比查询性能。

-- 有同步物化视图: 命中了 agg 条件过滤裁剪后的物化视图

```
select
year(orc_1stupddt)
, month(orc_1stupddt)
, day(orc_1stupddt)
, count(orc_id)
from TEST_MVW..xxx_records_rt
group by year(orc_1stupddt),month(orc_1stupddt),day(orc_1stupddt);
```

-- 无同步物化视图:

```
select
year(orc_1stupddt)
, month(orc_1stupddt)
, day(orc_1stupddt)
, count(orc_id)
from TEST_MVW..xxx_records_rt_tmp1
group by year(orc_1stupddt),month(orc_1stupddt),day(orc_1stupddt);
```

### 3.6 SelectDB物化视图实践-结果分析

查询平均响应时间差异对比：

并行度	无物化视图查询平均响应时间 (ms)	有同步物化视图查询平均响应时间 (ms)	差异值 (ms)	差异倍数	平均差异倍数
5	873	113	760	7.73	10.26
10	1608	164	1444	9.80	10.26
20	3182	240	2942	13.26	10.26

引入同步物化视图后，单表维度聚合查询性能得到了显著提升：

- 1. 查询响应时间减少：在不同并发场景下，查询平均响应时间平均提升约 10 倍，查询响应时间显著减少。
  - 2. 吞吐量提升：物化视图显著提升了系统的查询吞吐量，平均提升约 9.5 倍，系统并发处理能力大幅提升。
  - 3. CPU使用率降低：引入物化视图后，CPU使用率降低约 1.53 倍。
  - 4. 内存使用率变化较小：内存使用率的变化相对较小，平均差异倍数为 1.16，说明内存并不是该查询场景下的主要瓶颈。
- 整体来看，引入同步物化视图在单表维度聚合查询中有效地提升了查询效率、吞吐量，并显著减少了CPU 的使用率，对于聚合查询场景具有良好的优化效果。

04

# 收益与展望



## 4.1 收益-数据增长

>200<sub>↑</sub>

客户数量

>9000<sub>万</sub>

TOS 产品支撑国内箱量  
(TEU)

>100<sub>座</sub>

客户覆盖码头

## 4.2 收益-业务效果

### 显著的性能提升

- 1.核心报表数据实时性从 1-2 天延迟骤减至 5s 内。
- 2.80% 即席分析可在 2s 内返回结果，95% 的即席分析可在 5s 内返回结果。

### 智慧港口数据大脑

为 SMART 系列产品与全生命周期解决方案提供给数据大脑，推动港口业务创新。

### 极大降低成本

- 1.降低平台运维成本
- 2.SelectDB 极致的存储压缩比，存储成本降低 70%
- 3.降低人员开发成本

### 服务支持

提供 7\*24 小时技术支持服务、重点 BUG 天级快速修复及重大应急保障现场及时响应，有力支撑业务稳定运行，确保业务面对技术问题和突发状况时能持续高效开展。



## 4.3 展望

**1.SelectDB 存算分离实践：弹性资源配置，降低存储成本**

**2.SelectDB 增强多表物化视图业务运用：提升查询性能**

**3.集群管理工具运用：提升运维效率**

# Thanks for Watching!