DORIS × TiDB × NebulaGraph

AI-Native 数据库玩家集结

聊技术、拆案例、拓生态

观看直播

01月08日 周四 19:00-21:15

Hybrid Search and Analytical Processing

AI 时代的新数据库范式

朱伟 SelectDB 内核产品负责人

目录

01 Hybrid Search and Analytical Processing (HSAP)

02 为什么 Apache Doris 是 HSAP 的最佳选择

03 Apache Doris for HSAP 典型案例

搜索与分析分离的时代

在传统架构中，搜索与分析往往由两套完全不同的系统分别承载

搜索 (Search)

快速定位 | 相关性排名

专注于从海量非结构化数据中快速找到特定记录。常用于文档检索、日志查询等场景。

典型代表：Elasticsearch

分析 (Analytics)

聚合计算 | 趋势洞察

侧重于大规模数据的扫描、聚合与复杂计算。常用于报表生成、业务趋势分析等场景。

典型代表：Apache Doris (OLAP)

AI 时代的新挑战：搜索与分析的融合需求

随着 AI 应用落地，单一的搜索或分析已无法满足复杂的业务场景

AI 可观测性

先通过语义搜索快速定位相关的 LLM 回答日志，再基于这些日志分析情感分数的时序变化趋势。

需求：搜索 + 时序分析

SaaS 客户分析

先通过文本特征搜索特定用户群体，再关联事件数据，分析其使用模式与转化漏斗。

需求：文本搜索 + 关联分析

生成式 AI 应用

根据描述搜索相关产品，并实时关联其元数据与历史销量，为 LLM 生成准确的上下文。

需求：向量搜索 + 结构化查询

传统混合架构的四大痛点

强行组合 "ES + 分析数据库" 的模式，带来了不可忽视的系统负担

1. 开发与运维复杂

维护多套系统与数据管道，技术栈割裂。协调成本极高，架构变得臃肿，成为业务迭代的瓶颈。

2. 基础设施成本高昂

同一份数据在不同系统中重复存储与摄取。计算资源与存储资源的双重浪费，导致 TCO 居高不下。

3. 数据不一致风险

系统间同步延迟导致数据状态不同步。分析洞察往往滞后于实时搜索，影响决策质量与用户体验。

4. 查询逻辑割裂

无法在单一查询中高效结合搜索与分析。开发者被迫拆分业务逻辑，应用层聚合代价昂贵且性能低下。

新范式：HSAP（混合搜索与分析处理）

一个执行引擎

统一优化器与执行器，同时高效处理点查、全文检索、多维分析与复杂关联查询。

一份数据

数据仅存储一次，告别冗余与同步延迟。确保搜索结果与分析报表基于同一时刻的鲜活数据，实现强一致性。

一个接口

使用标准 SQL 统一访问，大幅降低学习曲线。简化应用开发，无需维护多套客户端逻辑。

一套运维

极简架构，显著减少组件数量。大幅降低运维复杂度与硬件资源开销，TCO 节省可达 50% 以上。

DORIS SELECTDB

目录

01 Hybrid Search and Analytical Processing (HSAP)

02 为什么 Apache Doris 是 HSAP 的最佳选择

03 Apache Doris for HSAP 典型案例

一体化执行引擎：HSAP 中的高效分析的关键

极速分析性能：化解“前过滤”瓶颈

混合检索通常先做结构化过滤。如果底层过滤太慢，就会拖累整个查询。Doris 通过三级加速机制解决此问题：

1. 分区/分桶裁剪

物理级过滤：优化器根据谓词条件，自动剔除无关的分区和 Tablet，大幅缩小扫描范围。

2. 多层索引加速

逻辑级过滤：利用 ZoneMap、Bloom Filter 和倒排索引，在扫描前快速判定并跳过无关数据块。

3. 向量化执行引擎

指令级加速：Pipeline 引擎充分利用多核并行与 SIMD 指令，极大提升算子吞吐与并发度。

虚拟列机制

计算下推，消除重复计算

OPTIMIZATION FLOW

FE 规划层：将打分函数映射为虚拟列

BE **Scan 节点直接计算虚拟列结果并填充**

无需上层节点重复计算，减少数据传输

TopN 延迟物化

避免宽表查询的读放大

PHASE 1

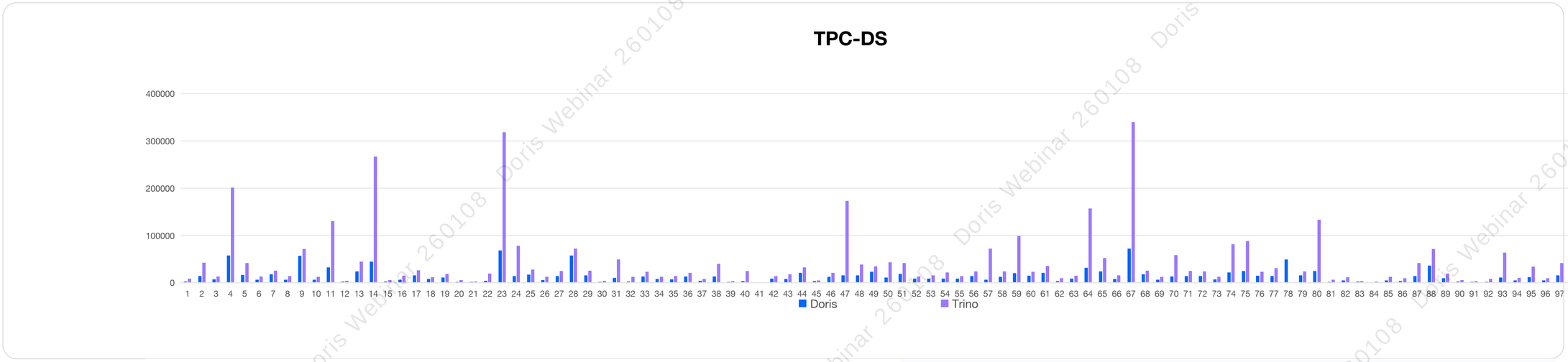
仅读排序列 & ID

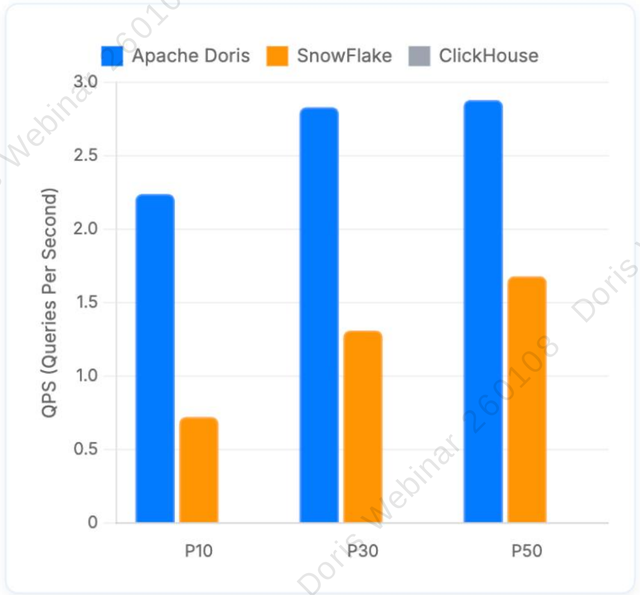
PHASE 2

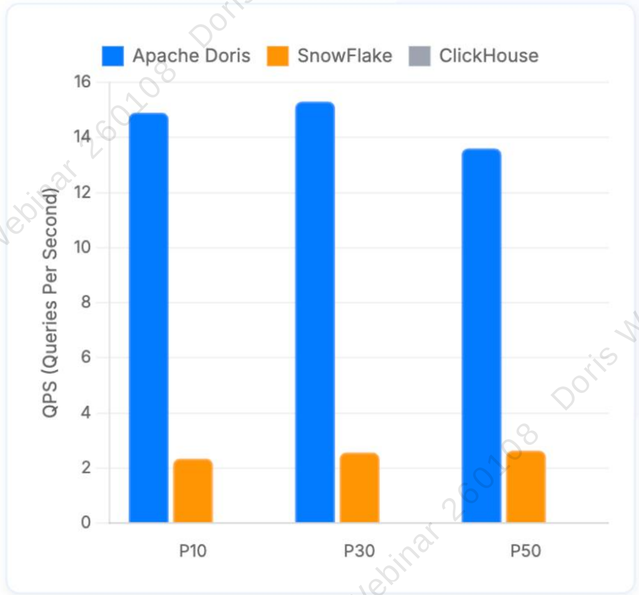
精准回表读取

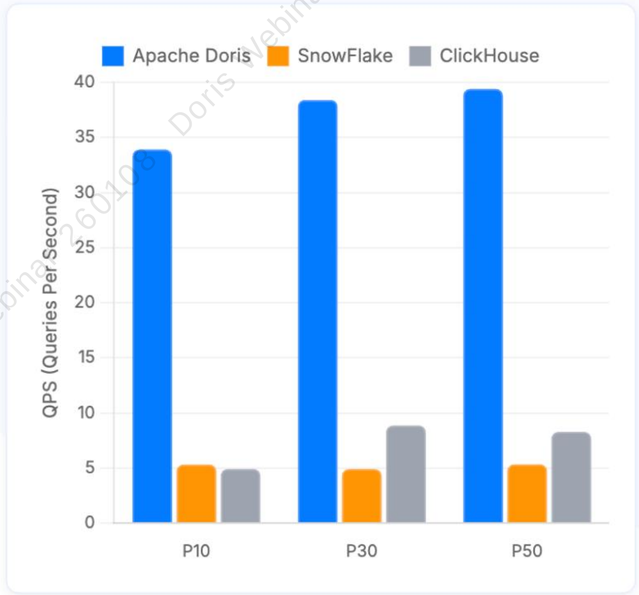
宽表场景下 TopN 查询效率提升 **几十倍**

Apache Doris实时分析性能

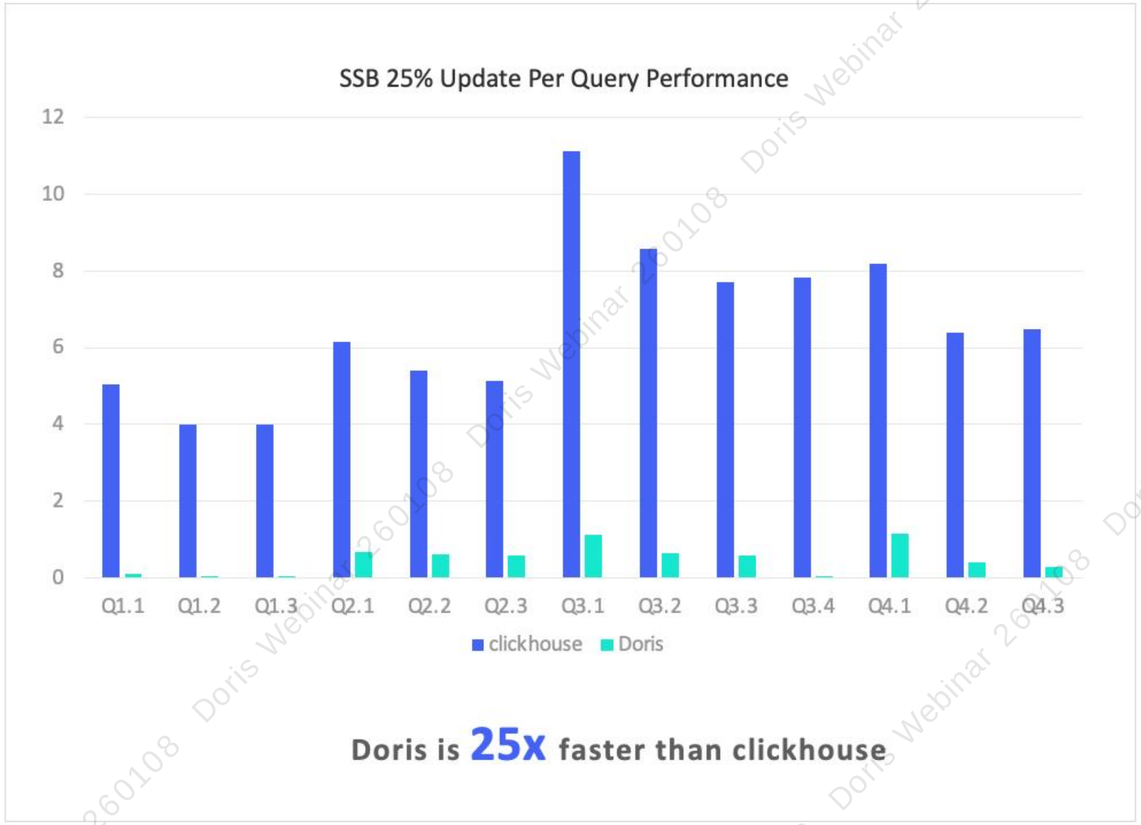


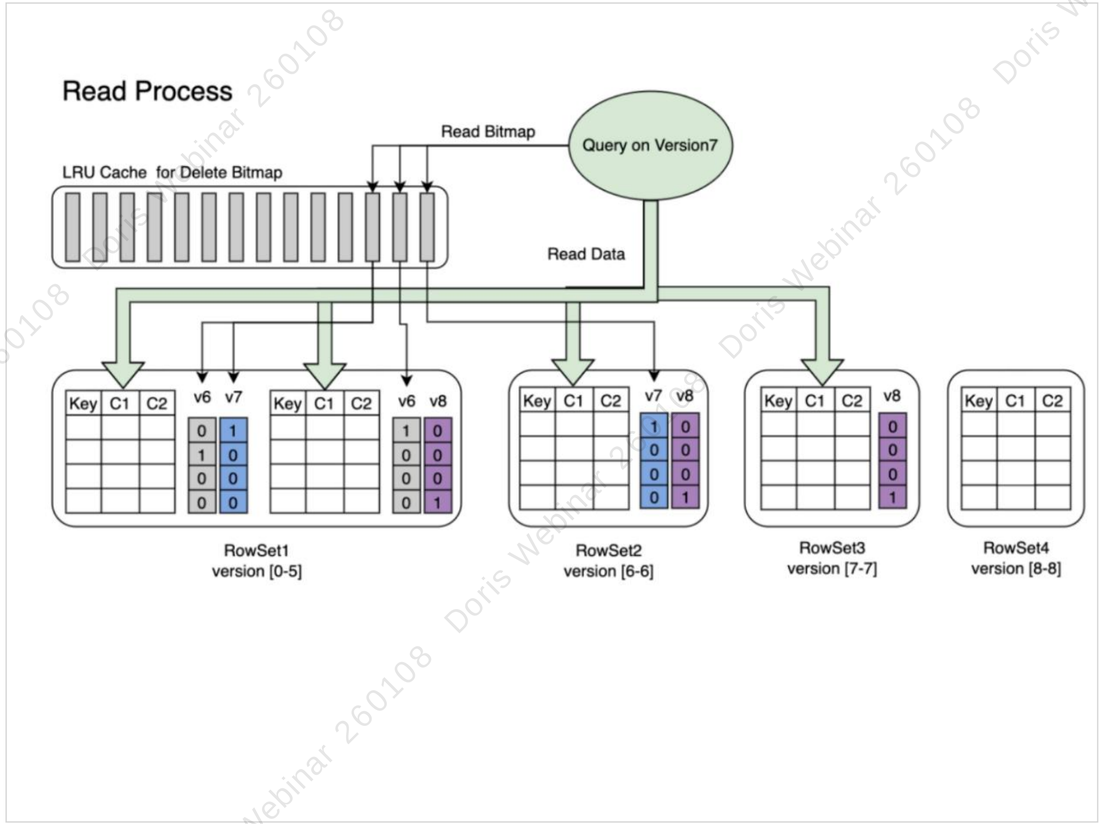






Apache Doris实时分析性能





DORIS SELECTDB

Apache Doris HSAP 能力演进

Apache Doris 正是基于这种 HSAP 模型演进而来：它通过统一的存储格式、统一的执行引擎和统一的 SQL 工作流，把结构化分析、倒排索引、向量索引三大能力整合为一个系统。这样做避免了多系统架构的冗余数据、重复建设与高延迟，天然契合 HSAP 所要求的混合搜索与实时分析能力。

Doris 2.0版本

基础奠定

- 引入倒排索引与基础匹配
- 实现了高性能的全文搜索能力

Doris 3.x版本

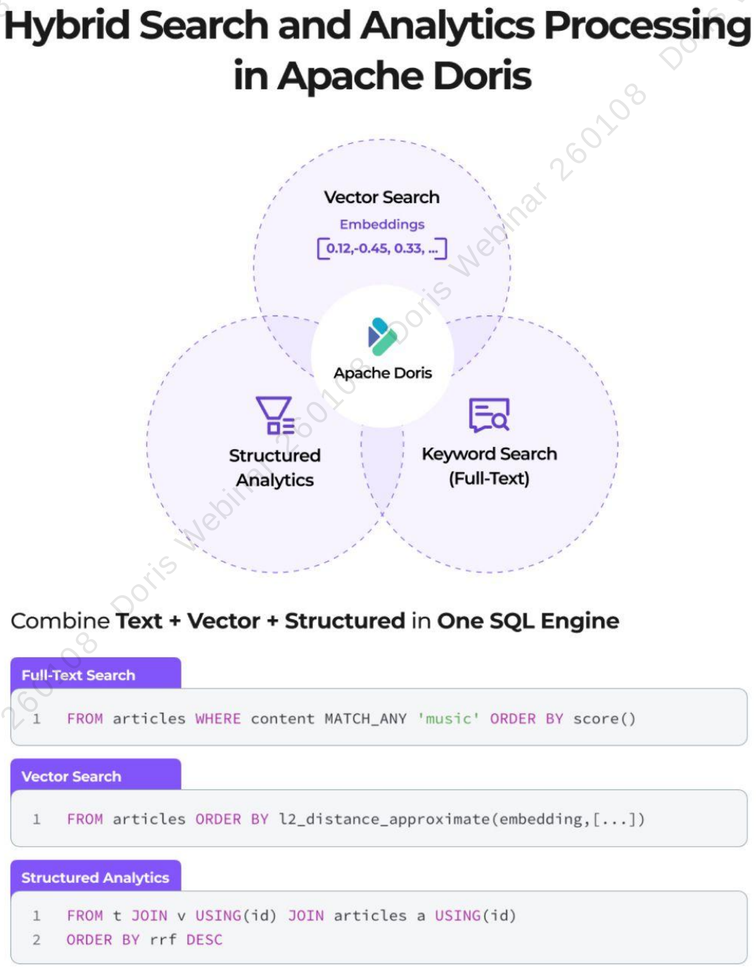
功能增强

- 引入更多文本搜索算子，比如 match_phrase_prefix, match_regex 等
- 引入自定义分词框架

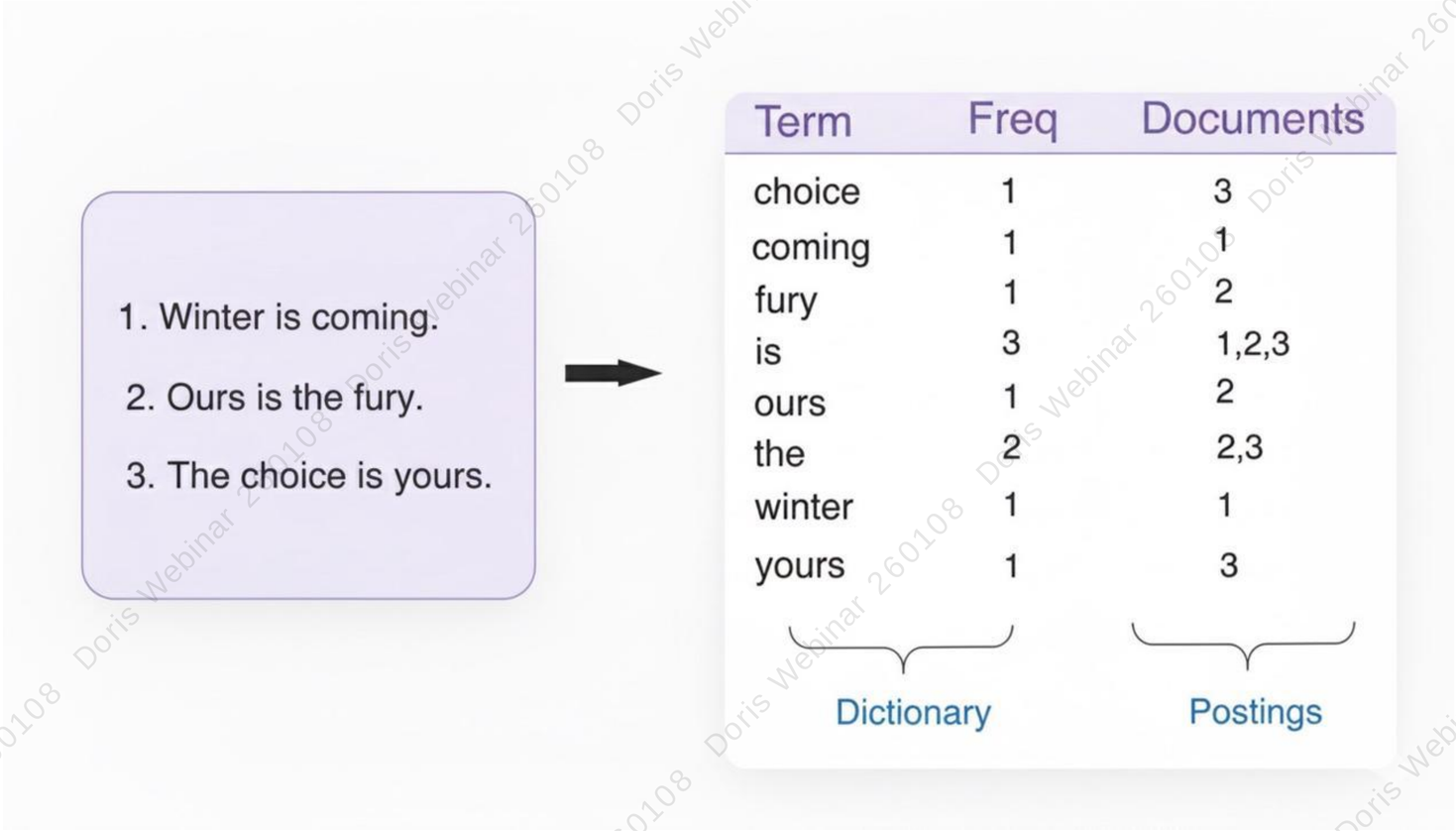
Doris 4.0版本

正式迈向混合搜索

- 引入 ANN 向量索引
- 引入文本相关性打分
- SEARCH函数统一文本搜索入口



文本搜索：基于倒排索引的高性能文本分析能力



文本搜索的核心是倒排索引

Apache Doris 在文本搜索的能力实现上，主要依赖于倒排索引、BM25 相关性打分等核心技术。

倒排索引原理

倒排索引的基本原理，是将文本拆分成词项（Term），并记录每个词项出现在哪些文档里。查询时无需逐行扫描，而是直接通过索引定位相关行号，从而将查询复杂度从 O(n) 下降到 O(log n) 级别。

Apache Doris 倒排索引的设计与优化

外挂式结构

- 索引独立与数据文件存储
  - ✓ 支持异步构建，不阻塞写入
  - ✓ 轻量级增删，降低索引管理成本
  - ✓ 支持索引compaction，降低系统资源消耗

查询和存储性能优化

- 索引查询不回表
  - ✓ 纯索引条件过滤或统计场景下，跳过数据文件读取，仅读索引即可完成计算，IO 开销降至最低。
- 双层缓存体系
- 词典、倒排表和位置信息自适应编码压缩

```
CREATE TABLE uba_event (
    ts      DATETIME,
    uid     BIGINT,
    action  STRING,
    detail  STRING
)
DUPLICATE KEY(ts, uid)
DISTRIBUTED BY HASH(uid) BUCKETS 10;

-- 异步创建索引，只对新增数据生效
CREATE INDEX idx_detail_ext
ON uba_event(detail)
USING INVERTED;

-- 历史数据构建索引
BUILD INDEX idx_detail_ext ON uba_event;

-- 查询和存储性能优化：索引查询不回表（纯索引过滤 / 统计）
SELECT   COUNT(*)
FROM uba_event
WHERE detail MATCH 'login failed';
-- 仅读取倒排索引完成过滤与聚合
```


DORIS SELECTDB

Apache Doris倒排索引的强大功能

功能完备的查询算子

算子	用途描述	SQL 示例
MATCH_ANY	包含任一关键词 (OR)	`WHERE content MATCH_ANY 'keyword1 keyword2'`
MATCH_ALL	包含所有关键词 (AND)	`WHERE content MATCH_ALL 'keyword1 keyword2'`
MATCH_PHRASE	短语精准匹配 (顺序+相邻)	`WHERE content MATCH_PHRASE 'hello world'`
MATCH_REGEXP	正则模式匹配 (不分词)	`WHERE content MATCH_REGEXP '^key_word.*'`

强大灵活的分词能力

- Doris 内置多种分词器，覆盖中英文、多语种 Unicode、ICU 国际化等主流语言。
- 同时支持自定义分析器，可配置字符清洗、分词模式、停用词、拼音转换、大小写规整等能力。

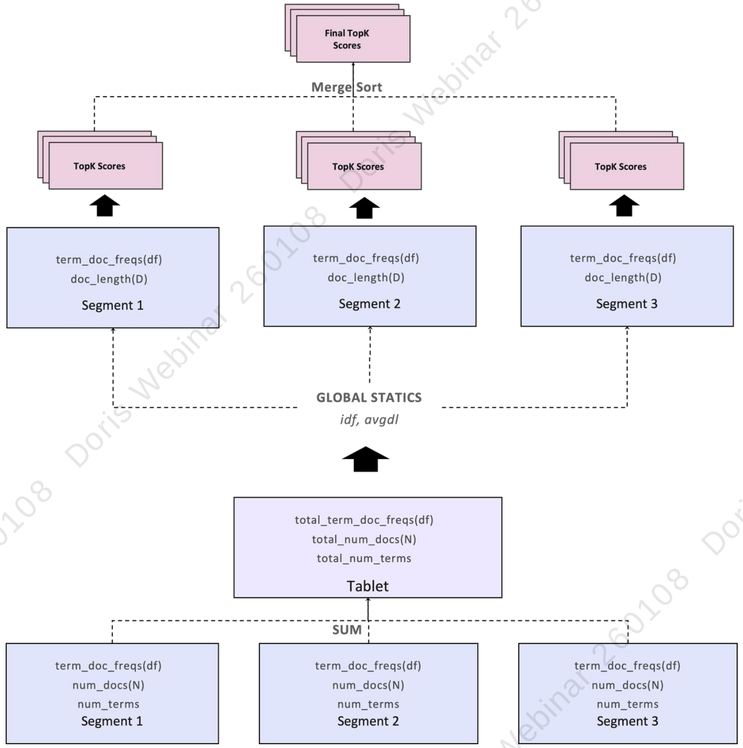
```
-- 1. 创建自定义 token filter
CREATE INVERTED INDEX TOKEN_FILTER IF NOT EXISTS
complex_word_splitter
PROPERTIES
(
    "type" = "word_delimiter",
    "type_table" = "[. => SUBWORD_DELIM], [_ =>
SUBWORD_DELIM]");

-- 2. 创建自定义分词器
CREATE INVERTED INDEX ANALYZER IF NOT EXISTS
complex_identifier_analyzer
PROPERTIES
(
    "tokenizer" = "standard",
    "token_filter" = "complex_word_splitter, lowercase"
);
```

BM25相关性评分

BM25（Best Matching 25）是一种基于概率的文本相关性评分算法，广泛应用于全文搜索引擎中。Doris 4.0 版本引入 BM25，为倒排索引查询提供了相关性评分功能。BM25 可根据文档长度动态调整词频权重，在长文本、多字段检索场景下（如日志分析、文档检索），显著提升结果相关性与检索准确性。

$$\mathrm{BM25}(Q,D)=\sum_{i=1}^{|Q|}\mathrm{IDF}(q_i)\times\frac{f(q_i,D)\times(k_1+1)}{f(q_i,D)+k_1\times(1-b+b\times|D|/\mathrm{avgdl})}$$



Doris 的 BM25 打分流程在分布式架构下，通过将统计信息收集层（Tablet 级别）与分数计算层（Segment 级别）解耦，从而有效确保了打分结果的稳定性。

阶段 1：Tablet 级别 - 收集全局统计信息

此阶段在 Tablet 级别执行，负责遍历所有 Segment 并收集 BM25 计算所需的全局元数据：

- 统计收集： 累加总文档数 (N) 和每个词项的全局文档频率 (f(qi,D))。
- 平均计算： 累加所有文档的总词数，计算出全局的平均文档长度 (avgdl)。
- 核心产出： 根据N和 f(qi,D)计算出每个查询词项的 全局IDF值。

阶段 2：Segment 级别 - 并行计算 BM25 分数并筛选 Top-K

此阶段是并行执行的打分环节。由于 BM25 的计算是文档独立的（一旦IDF确定，文档分数只依赖自身信息），每个 Segment 可以利用阶段 1 提供的全局统计信息，独立并行地完成打分和局部筛选：

- 并行打分： 每个 Segment 使用全IDF，结合自身的词频 (f(qi,D)) 和文档长度 (|D|)，计算所有匹配文档的 BM25 分数。
- 局部裁剪： 在 Segment 内部直接执行 Top-K 筛选，只保 b留和返回排名最高的文档 ID 和对应的分数，减少数据传输开销。

阶段 3：上层汇总 - 合并各 Segment 的 Top-K

查询的汇总节点（如 Backend 的聚合层）收集所有并行 Segment 返回的局部 Top-K 结果，进行全局归并排序，最终生成满足用户需求的 Top-K 结果集。

向量搜索：基于 ANN 索引拓展 Doris 语义搜索能力

在 Doris 4.0 中，向量索引采用了与倒排索引**统一的索引体系架构**。

统一执行框架与机制复用

得益于统一的索引架构，向量索引直接复用了倒排索引成熟的**底层文件管理接口**与**查询优化机制**。

优势 实现了架构上的灵活性与底层文件管理的透明性，大幅降低了软件工程的复杂度。

索引异步构建与灵活管理

基于这套统一架构，用户可以像管理倒排索引一样，对亿级 Embedding 数据进行**异步构建**。

优势 既避免了阻塞在线写入性能，也大大降低了索引重建与参数调整的运维成本，实现生产级的灵活管理。

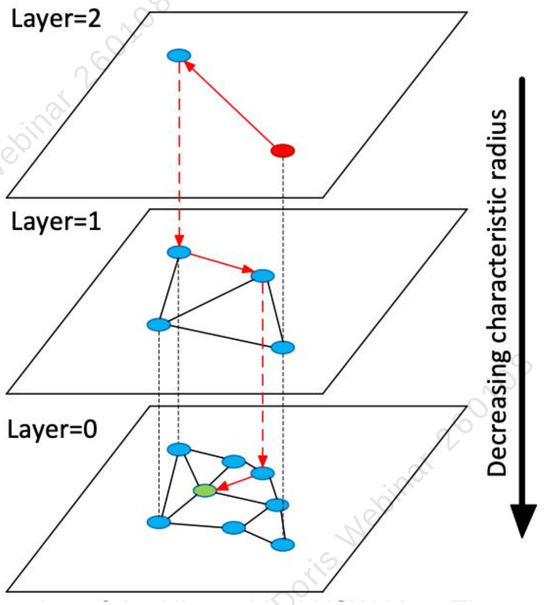
支持两类主流 ANN 索引，实现查询效率与构建成本的平衡

HNSW 索引

GRAPH BASED

基于分层图结构，快速从稀疏层导航至底层精细查找。

- **高召回、低延迟：**查询复杂度接近 O(log n)，保持高精度。
- **适用场景：**RAG、问答检索、对延迟敏感的在线服务。

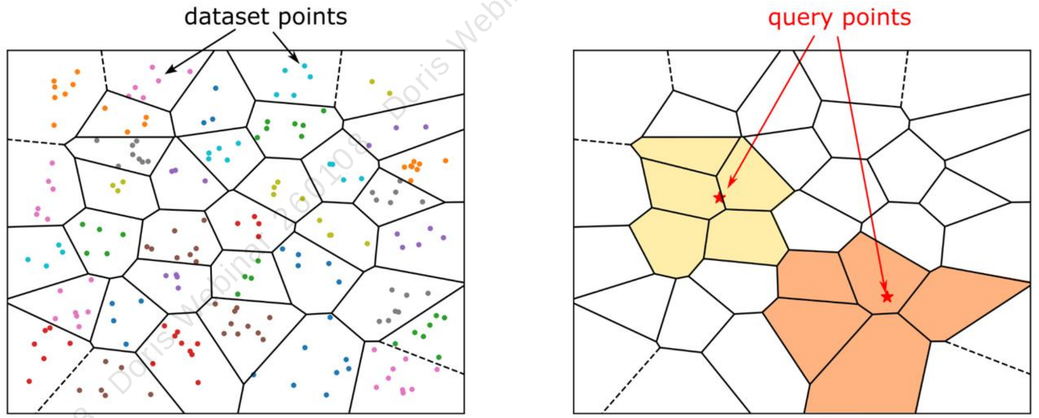


IVF 索引

CLUSTER BASED

基于倒排聚类分桶，只在最相关的分桶中进行搜索。

- **构建快、资源低：**适合海量数据，可通过参数平衡召回率。
- **适用场景：**千万级以上的日志、埋点、商品向量库。



DORIS SELECTDB

查询模式与量化压缩

多种向量查询模式

Top-K 最近邻检索

L2 / Inner Product

```
SELECT id, inner_product_approximate(
emb, [0.1, ...] ) AS dist FROM table
ORDER BY dist LIMIT 10
```

近似范围查询 (Range)

Radius Search

```
SELECT count(*) FROM table WHERE
l2_distance_approximate( emb, [0.1, ...]
) > 300
```

组合搜索 (Hybrid)

Filter + Vector

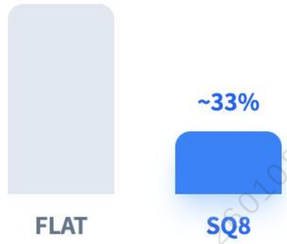
```
SELECT id, l2_distance_approximate(...)
AS dist FROM table WHERE
l2_distance_approximate(...) > 300 ORDER
BY dist LIMIT 10
```

量化与编码：打破内存瓶颈

标量量化 (SQ)

SQ8 / SQ4

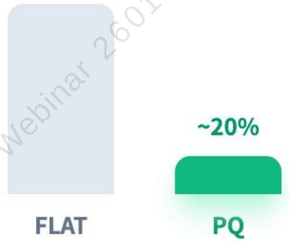
通过将 FLOAT32 压缩为低精度整数，显著减少内存占用。



乘积量化 (PQ)

Subspace

将高维向量分割为子空间分别量化，适合极致压缩场景。



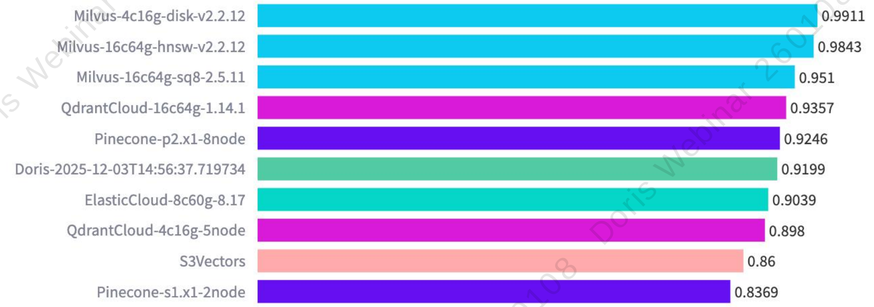
DORIS SELECTDB

VDBBench 测试结果

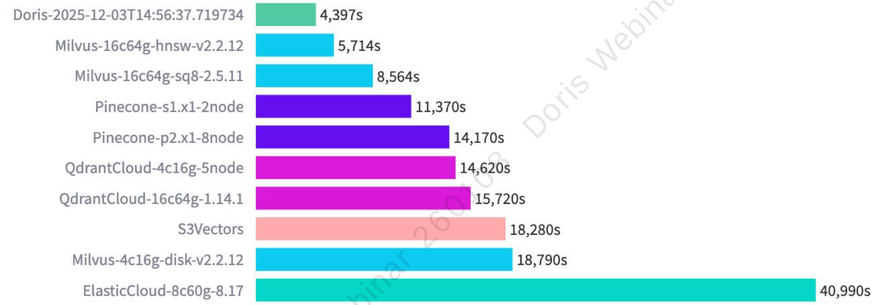
Search Performance Test (10M Dataset, 768 Dim)

Qps (more is better)

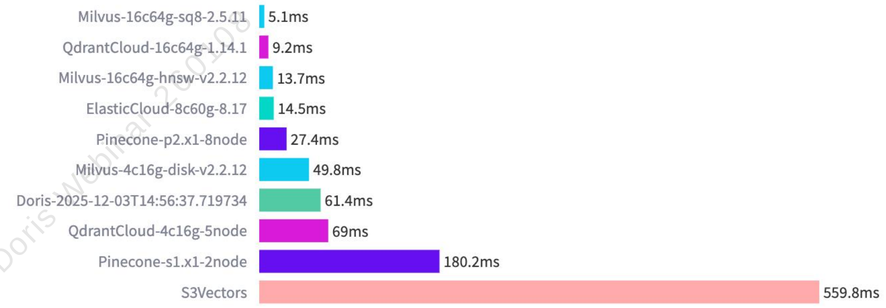
Recall (more is better)



Load_duration (less is better)



Serial_latency_p99 (less is better)



DORIS SELECTDB

目录

01 Hybrid Search and Analytical Processing (HSAP)

02 为什么 Apache Doris 是 HSAP 的最佳选择

03 Apache Doris for HSAP 典型案例

某全球领先的智能 CRM 平台案例

该平台利用 AI Agent 重塑客户交互体验，核心业务高度依赖实时数据处理

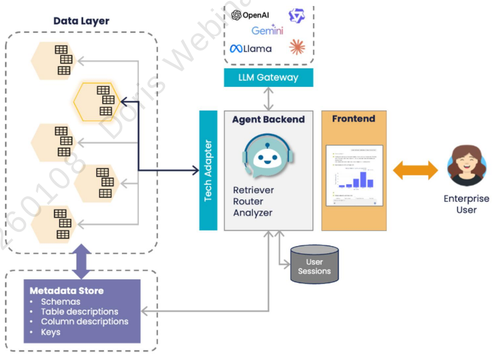
Customer Agent

全旅程智能客服

覆盖营销、销售、服务全渠道。提供 24/7 的个性化即时交互，需要极速检索客户历史以生成精准回复。

Data Agent

AI 驱动数据分析



通过自然语言提示，自动为 CRM 表格添加智能属性列。需要实时分析非结构化数据并更新结构化记录。

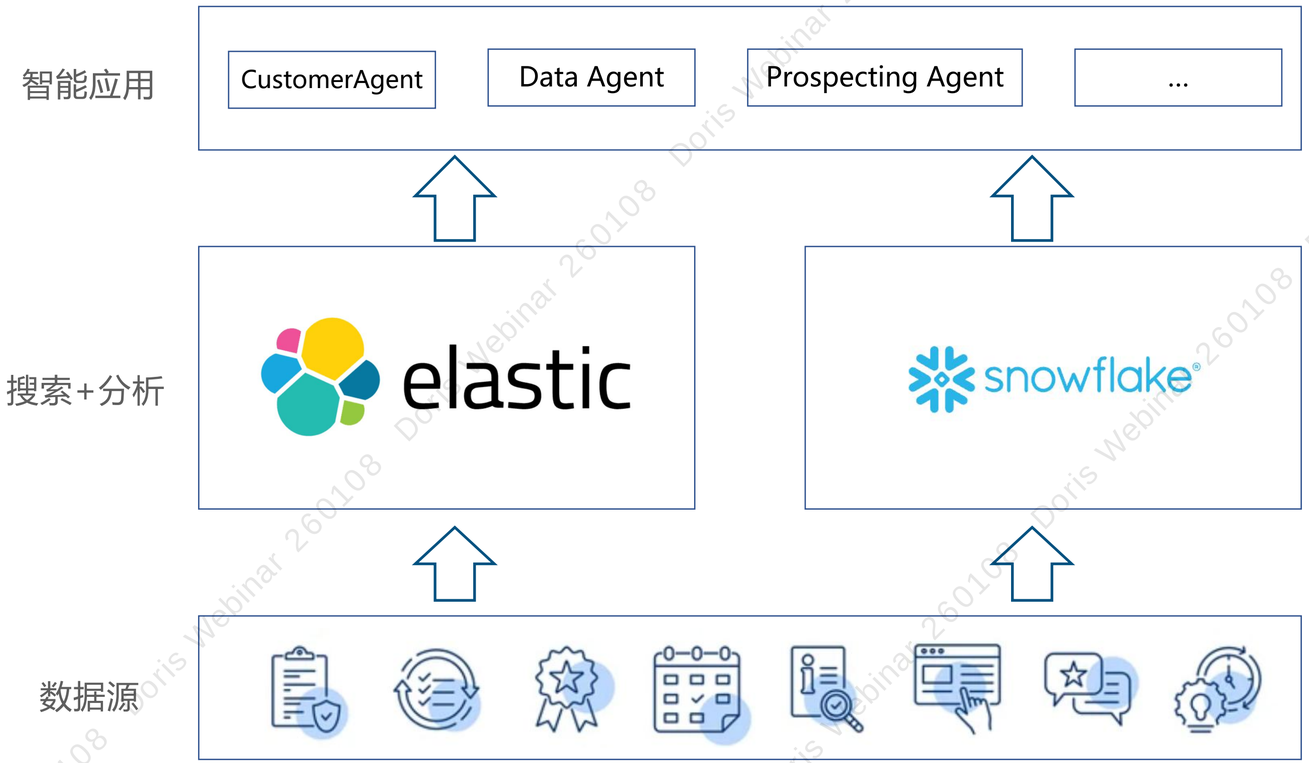
Prospecting Agent

智能个性化外联

结合内外部数据，自动生成定制化邮件并智能判断发送时机。依赖复杂的多源数据关联与分析。

DORIS SELECTDB

原有架构和痛点：Elasticsearch + Snowflake

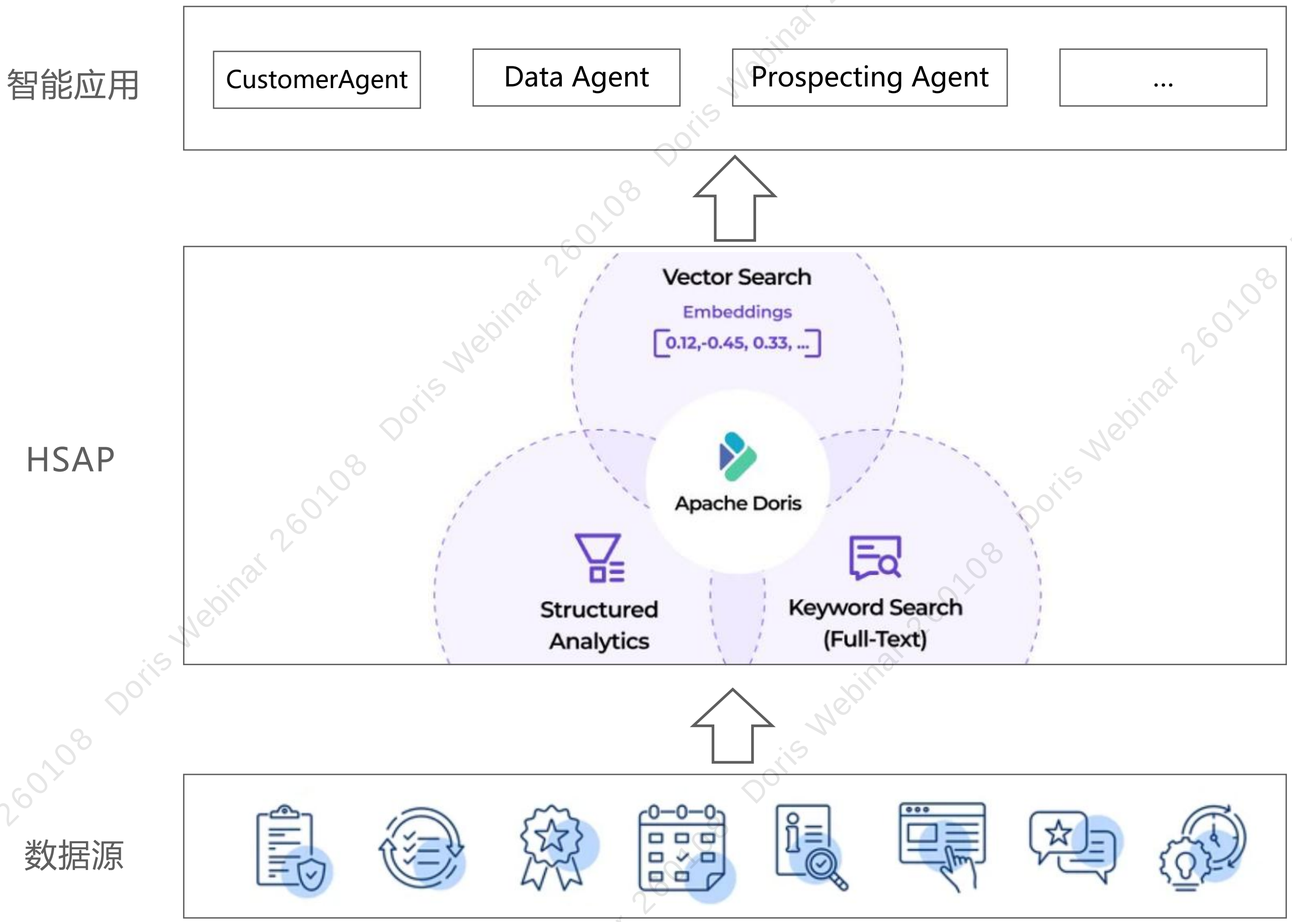


核心痛点

- **架构复杂**
  维护两套异构系统
- **能力局限**
  ES 无法关联分析，Snowflake复杂查询慢
- **体验割裂**
  数据不强一致，查询逻辑分散
- **成本高昂**
  双倍存储与计算开销

DORIS SELECTDB

基于 Apache Doris 的 HSAP 统一平台



全文搜索

支持倒排索引与自定义分词。完美满足电话、多语言文本的高性能精准搜索。

跨对象关联

原生 Join 能力。百毫秒级完成复杂关联搜索，替代 Snowflake 实时场景，弥补 ES 短板。

复杂分析 (OLAP)

高效列式存储与向量化引擎。复杂查询性能超越 Snowflake 5倍以上。

Variant 动态类型

动态 Schema 支持。高效存储万级动态子列 JSON，并自动创建智能索引。

DORIS SELECTDB

超宽 Variant 类型

```
CREATE TABLE IF NOT EXISTS tbl2 (
    k BIGINT,
    v VARIANT <
    'pattern1_*': STRING,
    -- batch-typing: all subpaths matching pattern1_* are STRING
    'pattern2_*': BIGINT,
    -- batch-typing: all subpaths matching pattern2_* are BIGINT
    properties("variant_max_subcolumns_count" = "2048") >,

    INDEX idx_p1 (v) USING INVERTED PROPERTIES(
        "field_pattern" = "pattern1_*",
        "parser" = "english"
    ),

    INDEX idx_p2 (v) USING INVERTED PROPERTIES("field_pattern" = "pattern2_*")
) DUPLICATE KEY(k);
```

1. 上万子列

- 业务中存在的高维度、多子列字段数据场景（包含 10000 以上子列的复杂结构数据）

2.高效查询

- Doris 对 Variant 类型的优化查询能力，即使面对超大量子列，仍能保持高效的查询性能，例如：可直接通过 properties['string69'] 这种方式直接精准获取第 10000 个子列的值，同时利用索引机制高效过滤。

全文搜索

```
-- 1) 邮箱/枚举字符串：整词匹配 + 小写
 INDEX idx_enumString_analyzer (`PROPERTIES`) USING INVERTED PROPERTIES
( "field_pattern" = "enumString*", "analyzer" = "keyword_lowercase" );

-- 2) 一般字符串：分隔符切词 + 小写 + 短语检索
 INDEX idx_string_analyzer (`PROPERTIES`) USING INVERTED PROPERTIES (
"field_pattern" = "string*", "analyzer" = "lowercase_delimited" );

-- 3) 电话：首端 n-gram 前缀检索（适配国际/本地格式）
 INDEX idx_phone_analyzer (`PROPERTIES`) USING INVERTED PROPERTIES (
"field_pattern" = "phone*", "analyzer" = "edge_ngram_phone_number" );
```

1. 灵活的分词检索策略

- 电话号码字段：需要先去除括号、短横线等分隔符，再提取数字，并支持 edge n-gram 前缀检索。
- 多语言文本字段：需要进行 Unicode 归一化，同时结合多种分词器与停用词过滤，保证不同语言下的检索一致性。
- 产品编码字段：既要保留完整编码用于精确匹配，又要按照分隔符拆分以支持局部检索，同时统一大小写。

2.同一字段多种索引

- 同一业务字段可能同时承载“精确匹配、短语匹配、前缀/模糊匹配、排序/聚合”等多种检索意图。

跨对象关联 & 复杂分析

业务对象之间的关联查询需求

- 用户希望通过客户 ID 查询其最近签署的合同、期间关联的所有联系人活动、以及相关商机的状态变更记录。
- 这类查询通常涉及多个维度的对象（客户、合同、商机、联系人、活动日志等）之间的多表关联与条件过滤，属于典型的跨对象关联查询场景

复杂分析场景

- 从海量客户对象数据中，提取包含多类联系方式（邮箱、电话、传真等）、创建时间、公司名称等信息的结果集，且要通过多字段的精确匹配、模糊匹配及数组包含校验等条件进行过滤，随后对结果进行排序分页，并统计总条数。
- 这类分析涉及大量字段的类型转换（如将枚举类型转为数组、将日期转为长整数）、复杂的多条件组合过滤（包含精确匹配、范围匹配、模糊匹配和数组元素校验）、子查询嵌套与联合查询，以及排序、分页和聚合计算等操作

Apache Doris 的 HSAP 架构收益

1.5s

写入延迟

超过 12 万条/秒写入(含更新)
数据可见性达到秒级，远优于原有架构的 3 分钟。

实时性飞跃

5x

分析性能提升

文本搜索万级 QPS，99 分位查询延迟 <100ms。复杂分析性能提升超过 5 倍。

极致查询体验

50%

资源节省

集群规模从 15,000 核降至 8,000 核以下，显著降低拥有成本 (TCO)。

降本增效

Thanks !

DORIS × TiDB × NebulaGraph

回放与演讲资料获取

请关注 SelectDB 公众号发送 **20260108**

联系我们

www.selectdb.com

400-092-6099

关注 SelectDB

关注 TiDB

关注 NebulaGraph