QCon
全球软件开发大会

InfoQ 极客传媒

火山引擎 Prometheus 面向大模型场景的优化实践

郭刚平

火山引擎托管 Prometheus 产品研发负责人

目录

InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

01

大模型场景指标观测需求和挑战

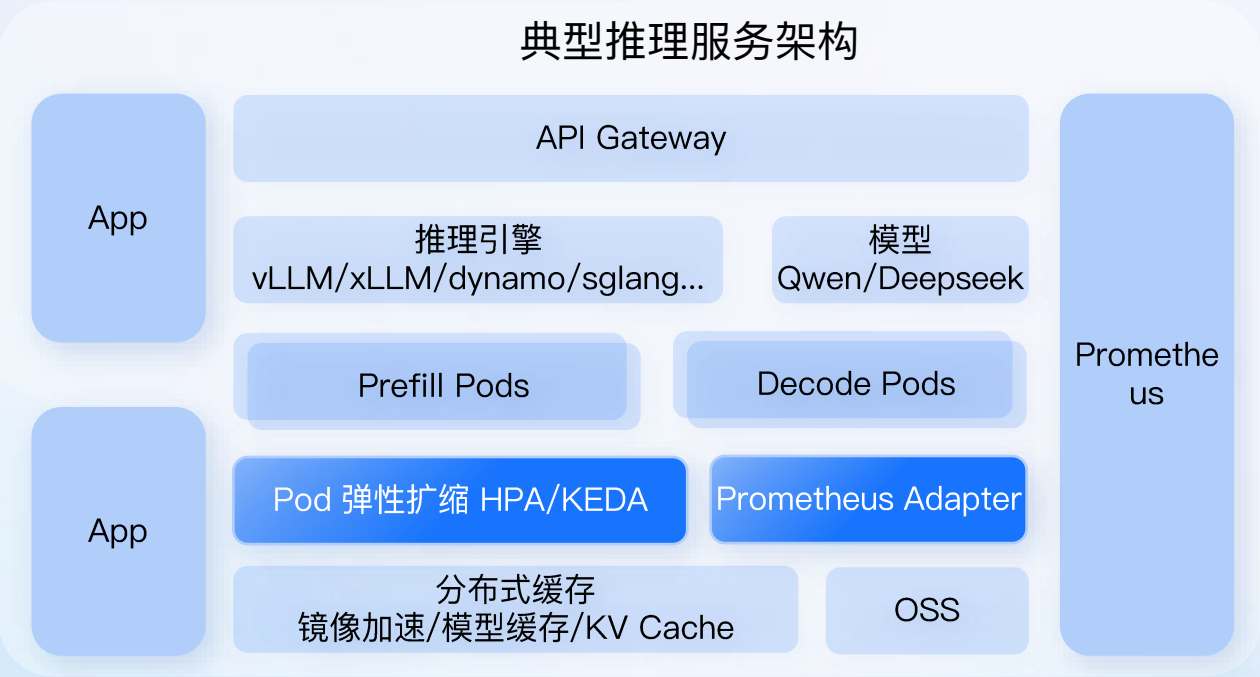
大模型场景 Prometheus 全栈监控

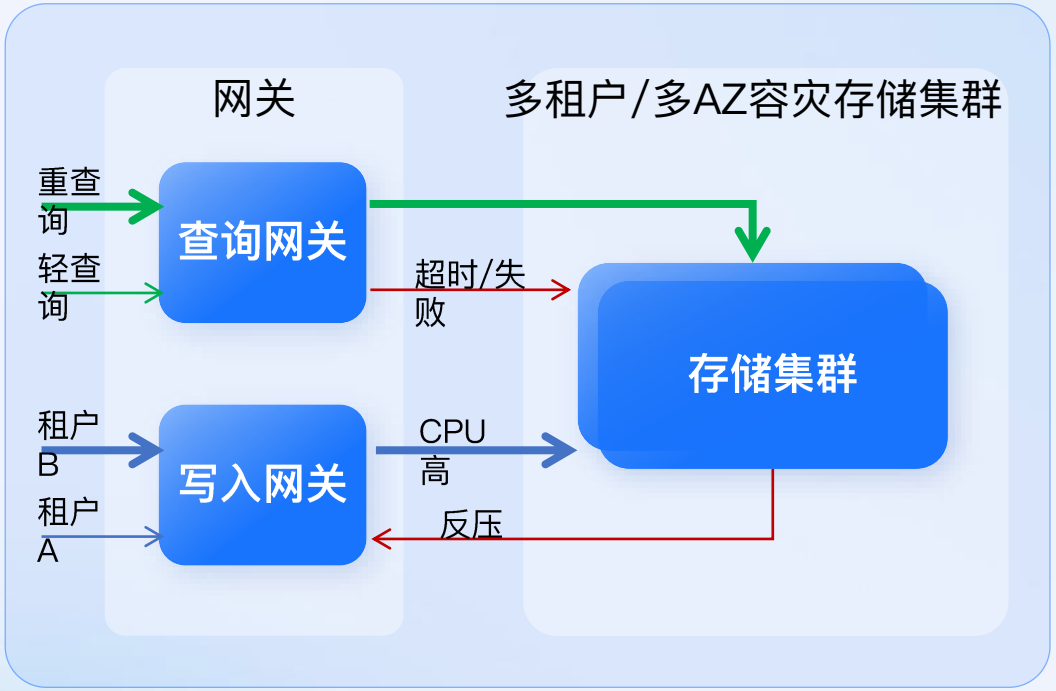
大模型场景指标观测需求和挑战

案例一：火山方舟平台在线推理服务

特点 1：面向ToD/ToC的推理API服务　　指标规模随用户数线性增长　　高基数、重查询问题，影响其他小租户

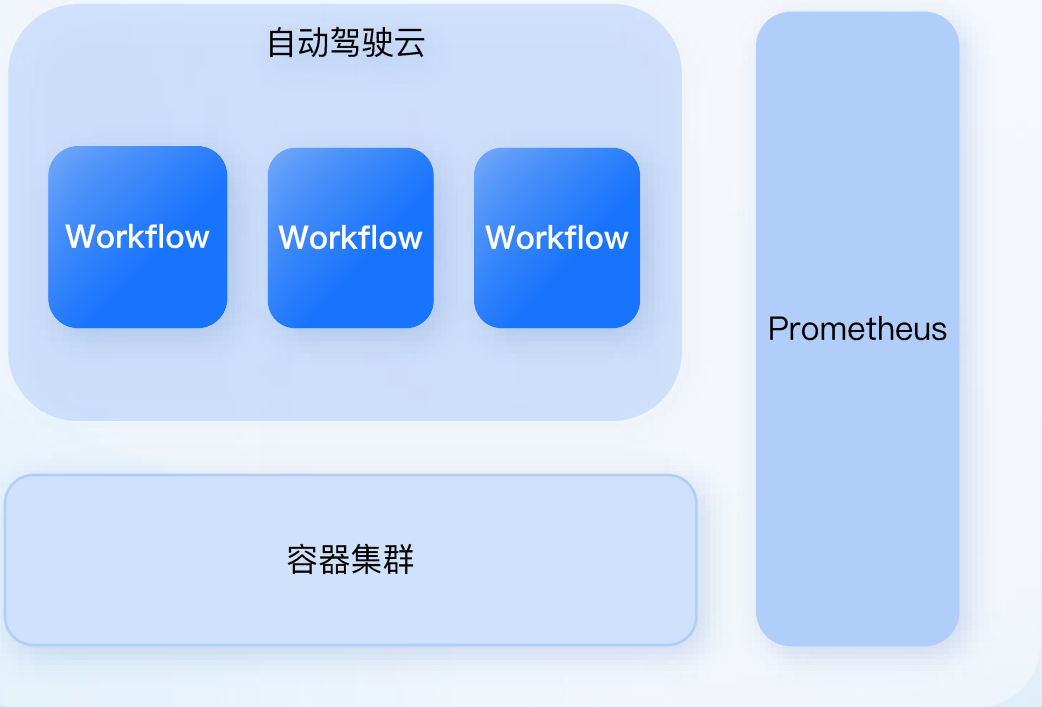
特点 2：SLO、成本效率要求高　　重度依赖基于自定义指标的HPA　　可用性要求高

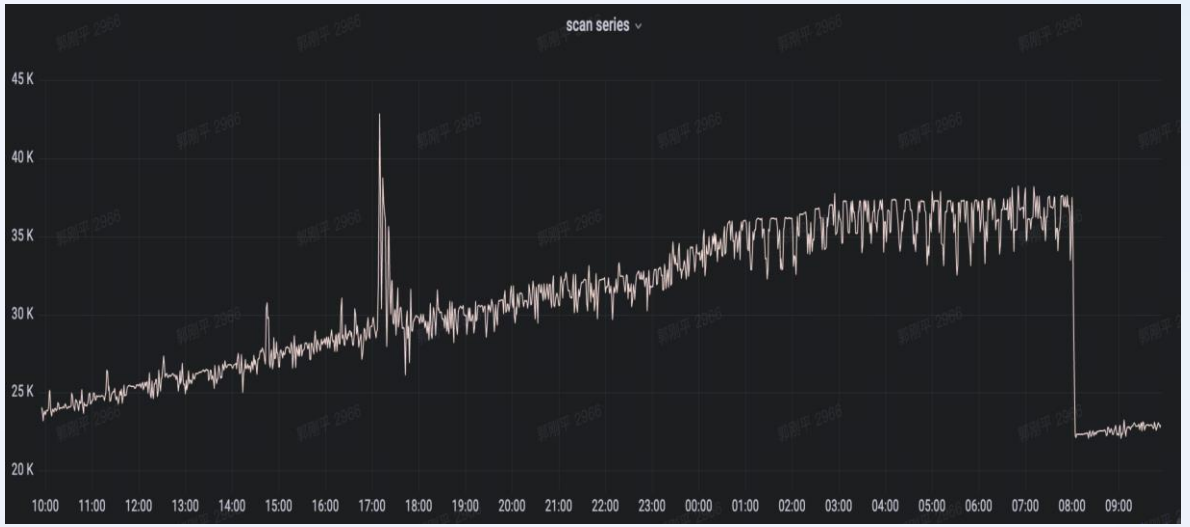




案例二：自动驾驶云平台

特点：大量短生命周期的任务　　高基数问题　　每天下午开始RecordingRule执行失败





02

火山引擎 Prometheus 优化思路

火山引擎 Prometheus 监控平台架构

火山引擎 Prometheus 优化思路

全层次视角

从集群 -> 租户 -> 请求粒度 逐层分析看

端到端视角

从sdk -> agent -> 网关 -> 存储引擎 整个链路看

思路与原则

可慢不可瘫

宁可查询请求慢一点，也要尽量避免整个服务挂掉影响更多用户

优、治结合

从水平、垂直视角，合理优化提升系统上限

辅以治理解决方案，帮助用户发现、治理不合理使用姿势

火山引擎 Prometheus 优化思路

具体解法					
问题	整体方向	端到端视角	全层次视角	可慢不可瘫	优、治结合
高可用： 租户QoS保障	加强隔离，设置兜底	网关：识别大流量租户并拆分独立集群	集群粒度：识别大流量租户并拆分独立集群 租户粒度：查询请求shuffle sharding，降低故障域 请求粒度：设置自我保护性资源限制	计算节点：Never OOM 设计	慢查询分析 查询封禁 子用户限流 细化Qos限流项
高可用： HPA可靠性	预测能力	HPA：增加预测能力，降低延迟	集群粒度：多AZ容灾		
高性能/高扩展： 高基数问题	治理为主，降低基数	打点SDK：不活跃时序自动过期删除 引擎：小时粒度索引，减少无效扫描	集群粒度：数据跨集群分流，统一聚合查询，提升扩展性		打点姿势最佳实践 采集Agent：Agent侧写入预聚合，降低指标维度

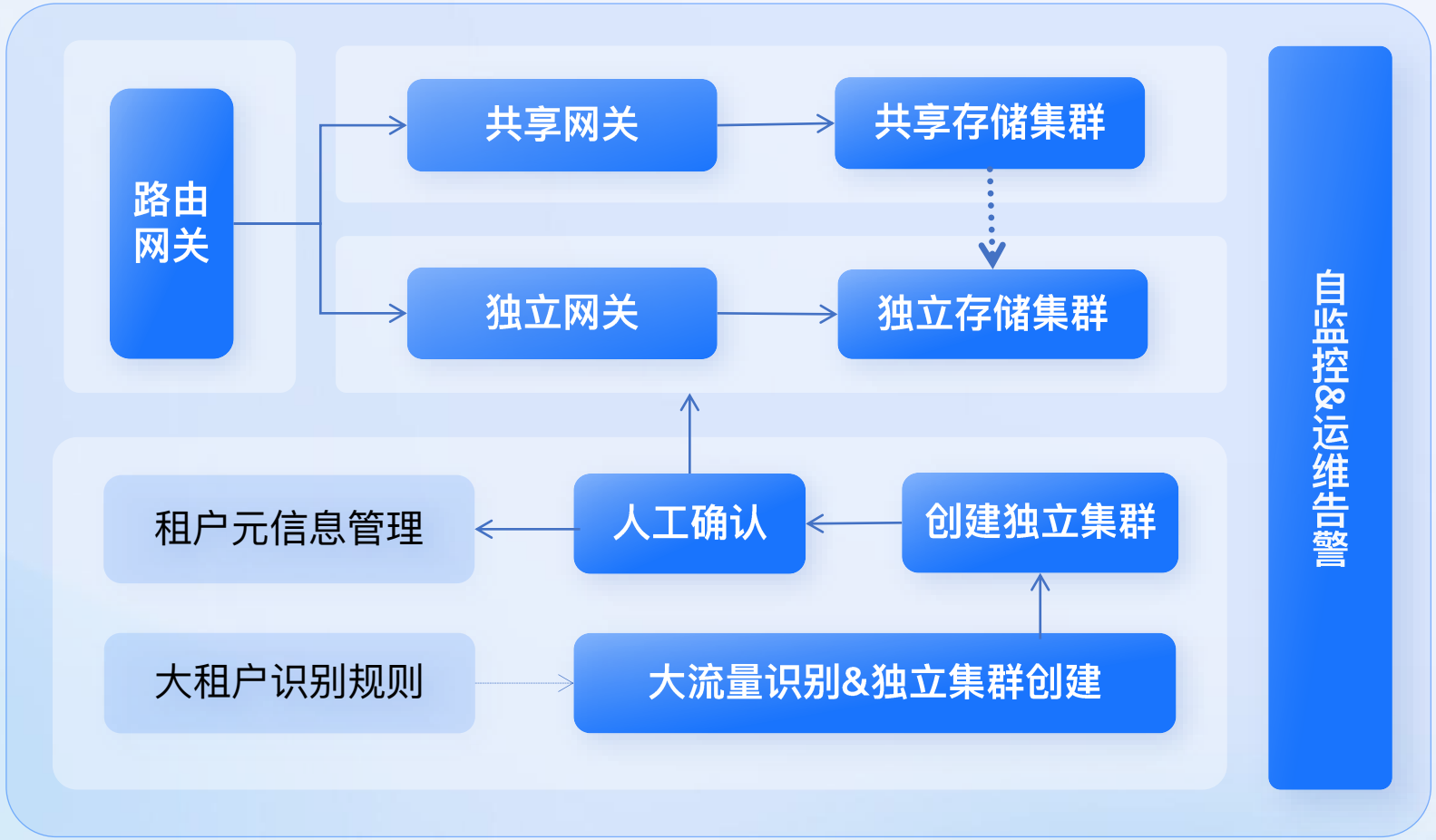
03

火山引擎 Prometheus 优化实践

租户 QoS 保障

租户 QoS 保障

1. 大流量租户自动识别&独立拆分



实现形式：

创建独立集群是一个k8s CRD，由一个专用的controller负责状态更新

CR的状态分为：Auditing、Creating、Created、Routed

逻辑步骤：

定义大租户规则(活跃时序超5亿或写入速率超15M samples/s)

持续监控指标跟踪

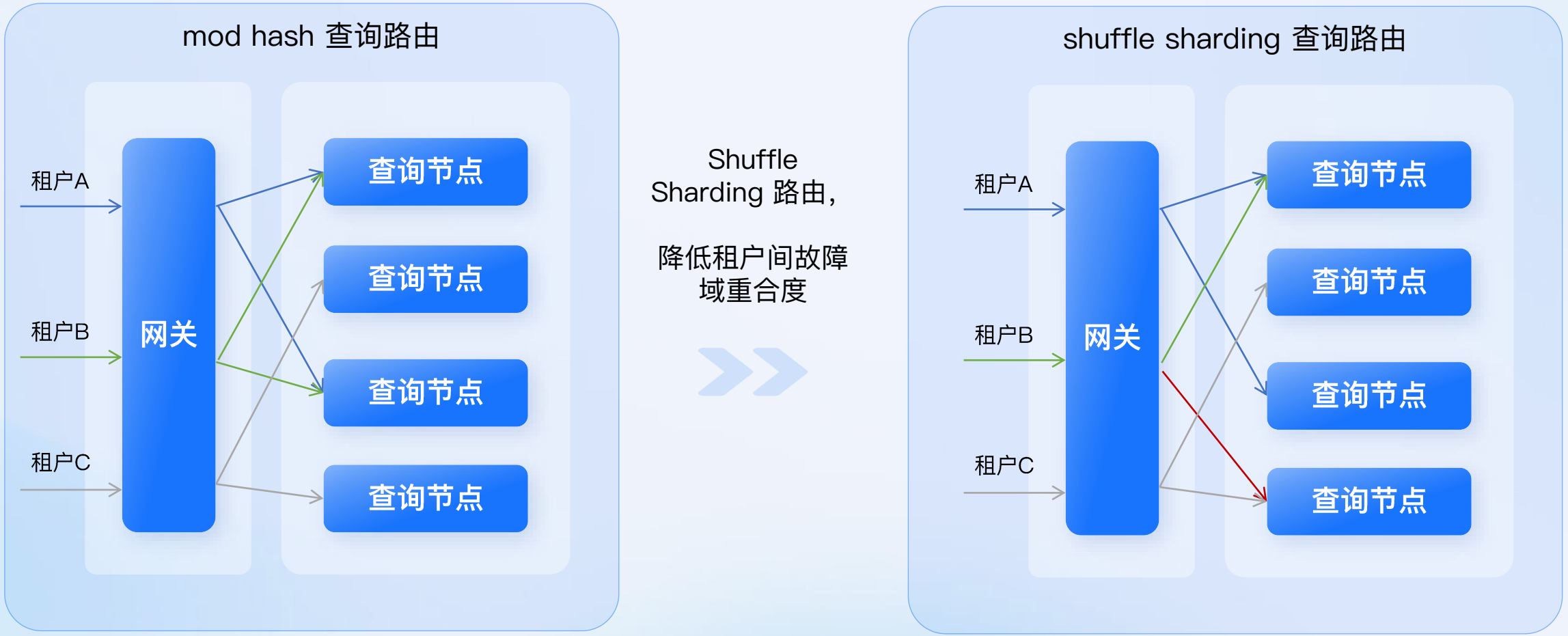
创建独立集群CR，通过告警触发通知人工确认

人工确认，创建独立集群组件

独立集群ready后，更新路由关系

租户 QoS 保障

2. 故障域隔离



租户 QoS 保障

2. 故障域隔离

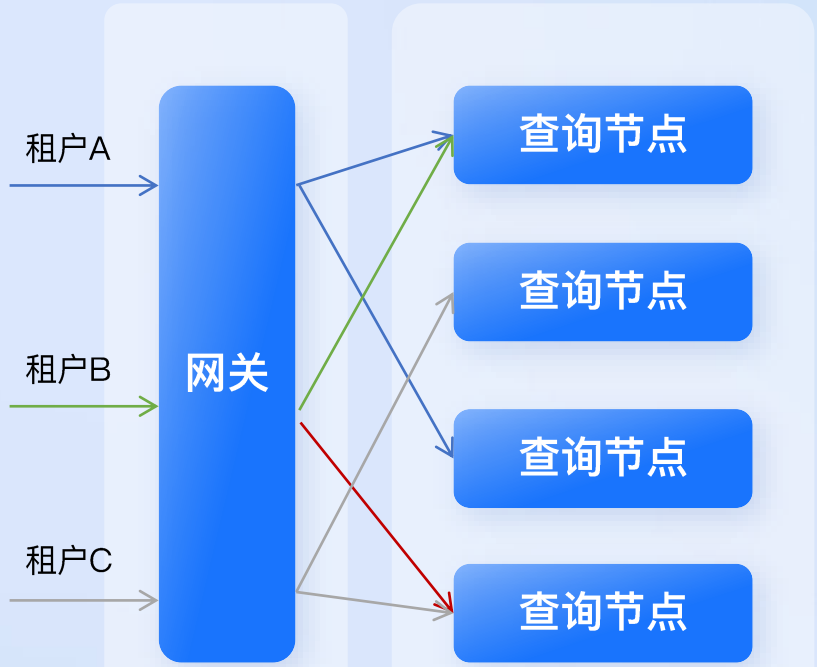
代码实现

```
func SimpleShuffle(seed int, identifier []byte, endpointsPerCell int)
(endpoints []string) {
    r := rand.New(seed * hash(identifier))

    eps := GetAllEndpoints()
    r.Shuffle(len(eps), func(x, y int) {
        eps[x], eps[y] = eps[y], eps[x]
    })

    return eps[:endpointsPerCell]
}
```

shuffle sharding 查询路由



租户 QoS 保障

3. Never OOM 设计

1. 主动设限：实例 & 请求 粒度
2. 用前检查：申请内存前应该判断是否有余量
3. 用后清理：及时清理非活跃内存
4. 溢出磁盘：超出容量设计部分，考虑溢出磁盘
5. 过载保护：使用自适应限流，避免服务过载
6. 监控优化：持续监控各组件内存用量，不断优化

租户 QoS 保障

4. 精细、高效限流

限流项	示例	实现方式	限流效果
写入速率	10M samples/s	分布式限流	返回429
查询 QPS	100 count/s	分布式限流	返回429
查询扫描时序速率	20000 series/s	分布式限流	窗口内查询熔断
查询扫描数据速率	2亿 samples/s	分布式限流	窗口内查询熔断

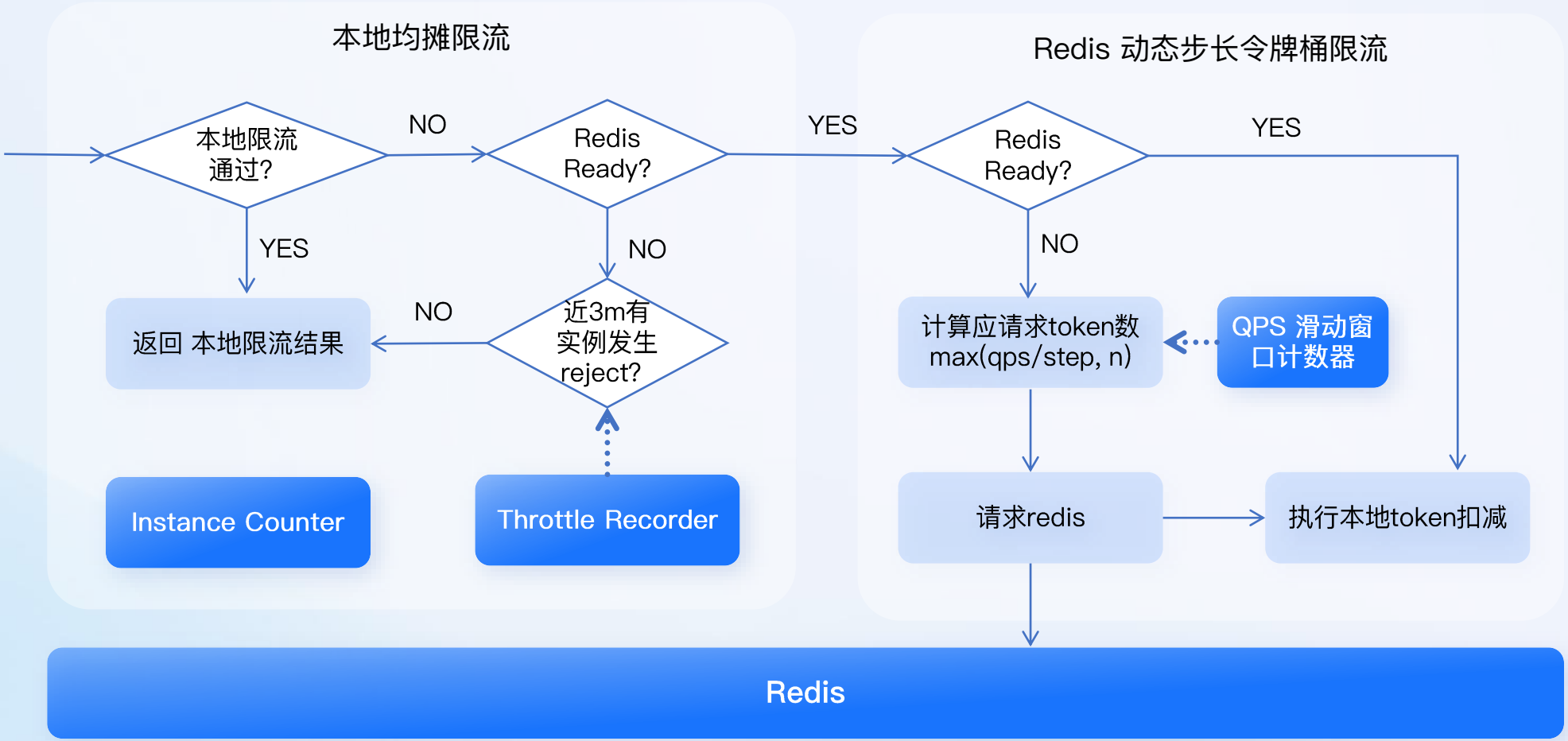
租户级别，SLA 预算

限流项	示例	限流效果
单次查询内存占用	1GB	返回422
单次查询扫描时序数	10w	返回422
单次查询扫描点数	10亿	返回422
单实例并发请求数	500	返回503

实例级别，兜底保护

租户 QoS 保障

4. 精细、高效限流　如何实现高效的分布式限流?



减少对 Redis 压力:

1. 优先本地均摊

2. 动态均摊实例

3. Redis 令牌桶根据QPS 动态请求令牌

租户 QoS 保障

5. 慢查询分析&治理

用户在会在不同的场景使用到Promethues的指标，比如业务HPA、告警、大盘，为不同的使用场景分配不同的子账号，并且配置不同的查询Qouta，就很有必要

实现原理：分布式限流

子用户限流

对于识别到的慢查询，我们希望用户能够感知到并且持续去治理优化，因此我们会对用户近7天的重查询进行展示，便于用户去优化

实现原理：慢查询日志&统计

慢查询分析

对于用户识别到的一些问题子用户或者慢查询语句，需要给到用户一种能力去封禁

实现原理：查询PromQL正则匹配

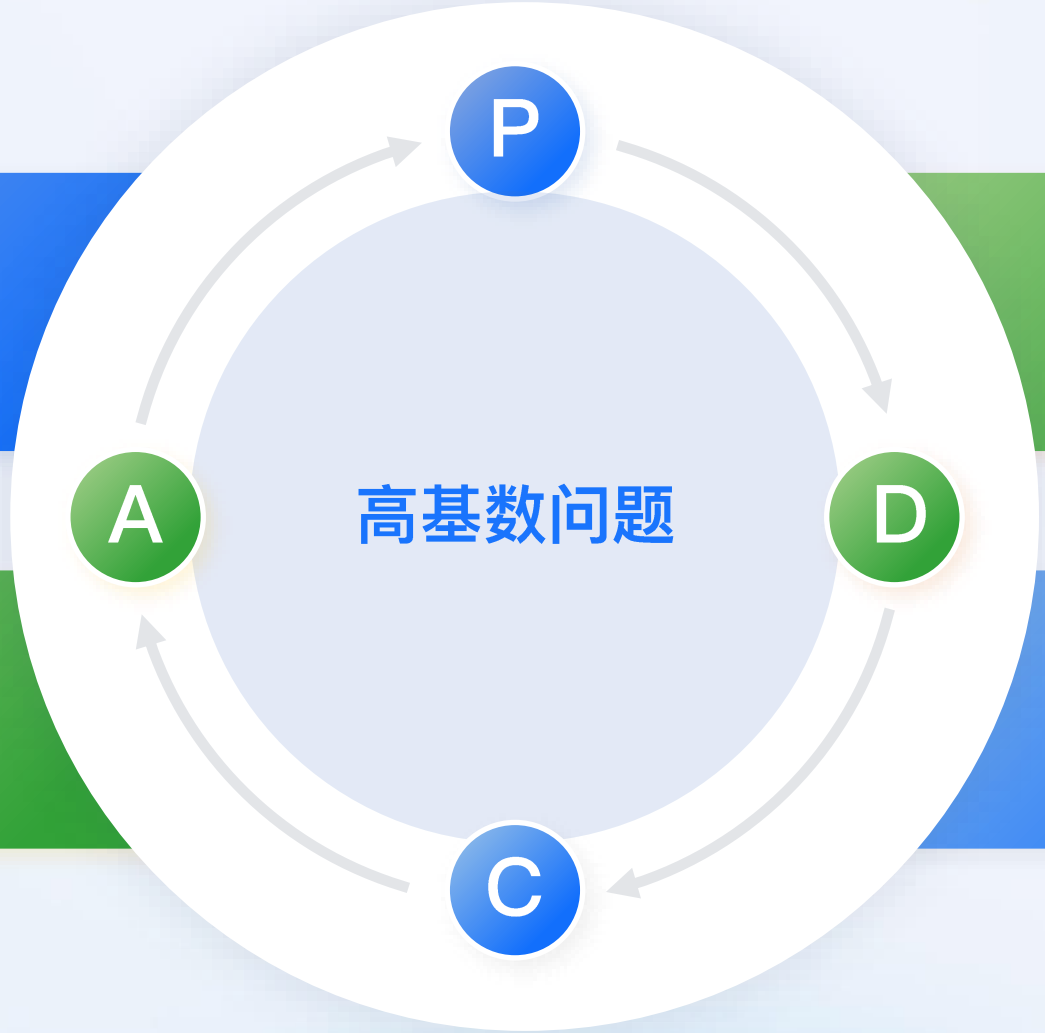
手动查询封禁

对于一些比较重的查询，比如查询超时、或者被兜底限流阈值限流的查询，如果不进行封禁，会导致持续的消耗资源

实现原理：查询PromQL + 查询时间范围匹配

自动查询封禁

高基数问题



1. 提供打点最佳实践

最佳实践文档和工具

2. 采集侧预聚合

写入前降低维度

4. 写入分片&聚合查询

分治原则

3. 查询优化

特定场景优化

高基数问题

1. 打点最佳实践

1. 合理设计：去除非必要标签，详细信息考虑log或trace
2. 指标拆分：根据使用场景，对指标标签进行合理划分，避免所有标签都放到同一指标
3. 活跃感知：及时删除不再活跃时序，提供sdk实现不活跃指标自动回收

http_request_total	维度	基数
instances	100	100*10*10*10=10w
path	10	
response_code	10	
response_size_level	10	

指标拆分

http_request_total	维度	基数
instances	100	100*10*10=1w
path	10	
response_code	10	

http_request_size	维度	基数
instances	100	100*10*10=1w
path	10	
response_size_level	10	

高基数问题

2. 采集侧预聚合

优先推荐在 Agent 端配置预聚合

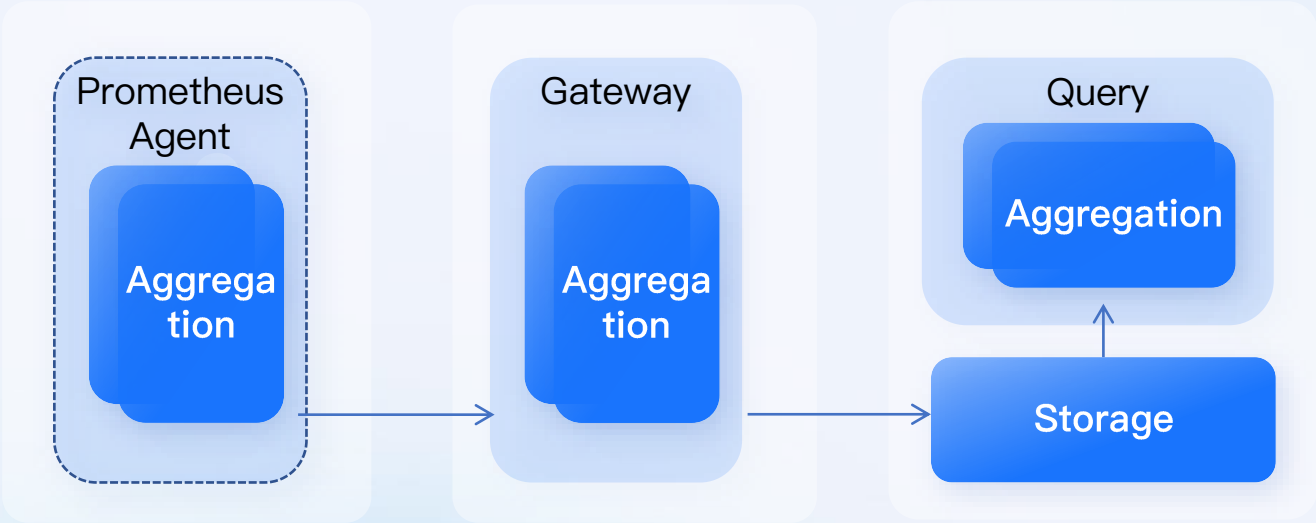
1. 预聚合的资源消耗分散，不易致服务端处理性能成为瓶颈

2. 服务端不用做过重的窗口聚合，对整体稳定性和端到端时延有帮助

cpu_usage_seconds_total	维度	基数
cluster	2	2*100*1000=20w
node	100	
pod	1000	

预聚合 by node

cpu_usage_seconds_total	维度	基数
cluster	2	2*100=200
node	100	



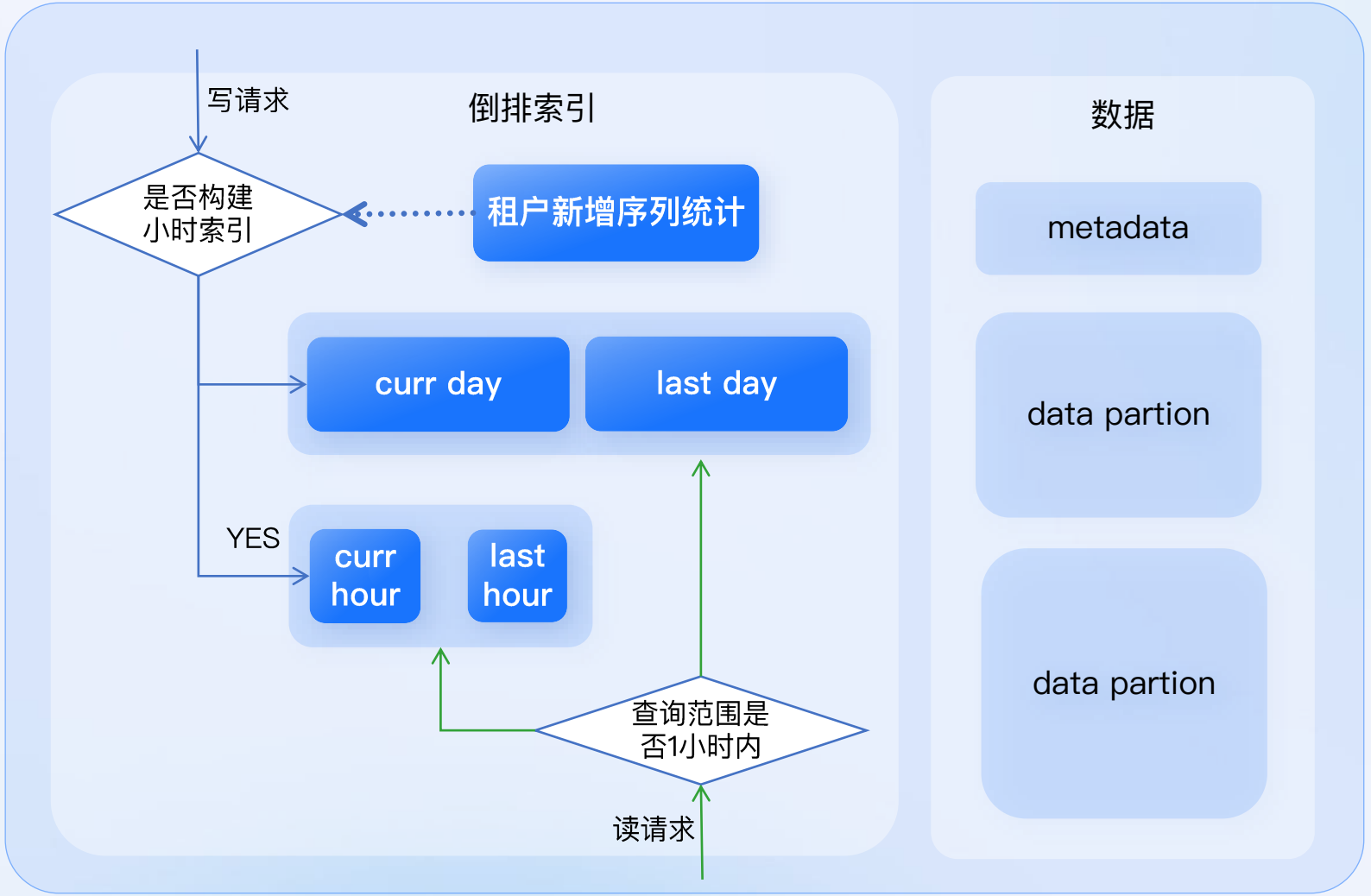
高基数问题

3. 查询优化

增加小时粒度索引

1. 只保留近2小时，覆盖告警和大部分查询场景

2. 小时粒度索引构建是启发式的，只对序列新增速率超过一定比例的用户开启，一方面节省存储，另一方面减少对写入吞吐的影响

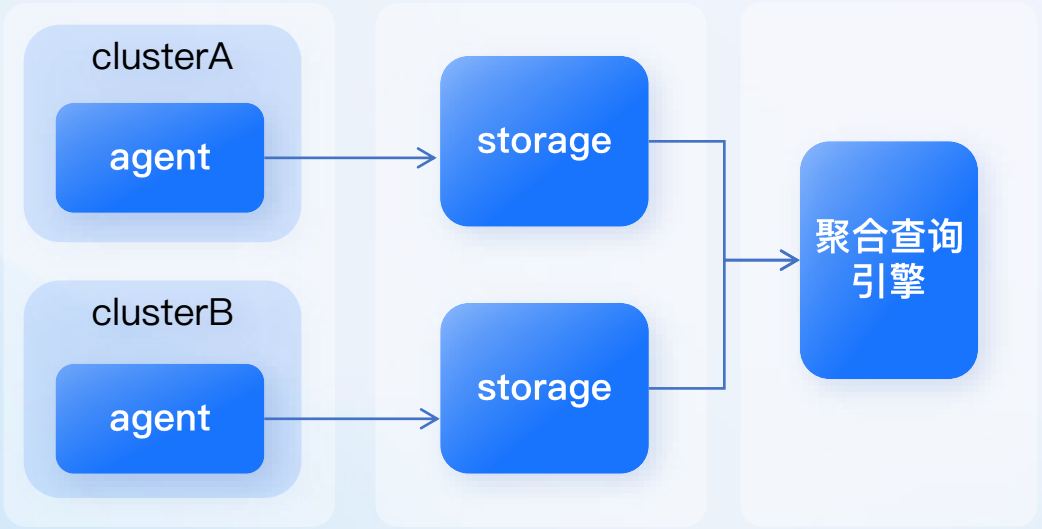


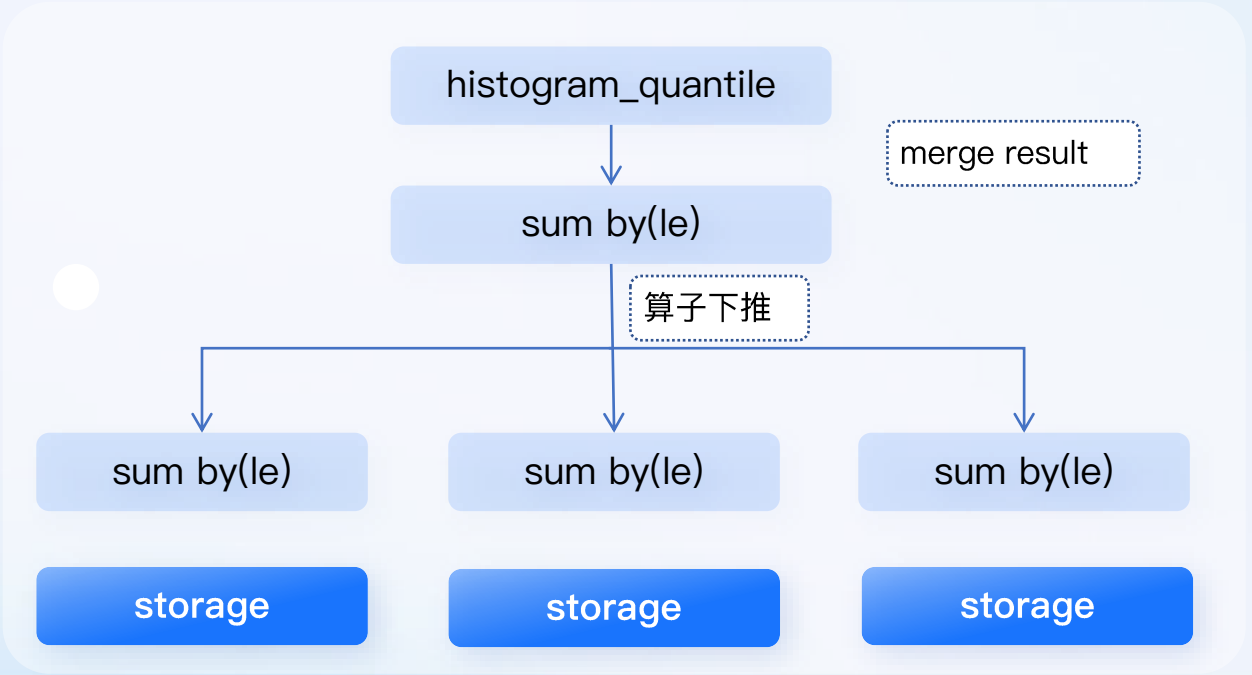
高基数问题

4. 分片写入&聚合查询

根据业务集群、指标名等拆分，降低单集群规模和运维压力

聚合查询，查询算子下推，并行执行加速查询





HPA 可靠性

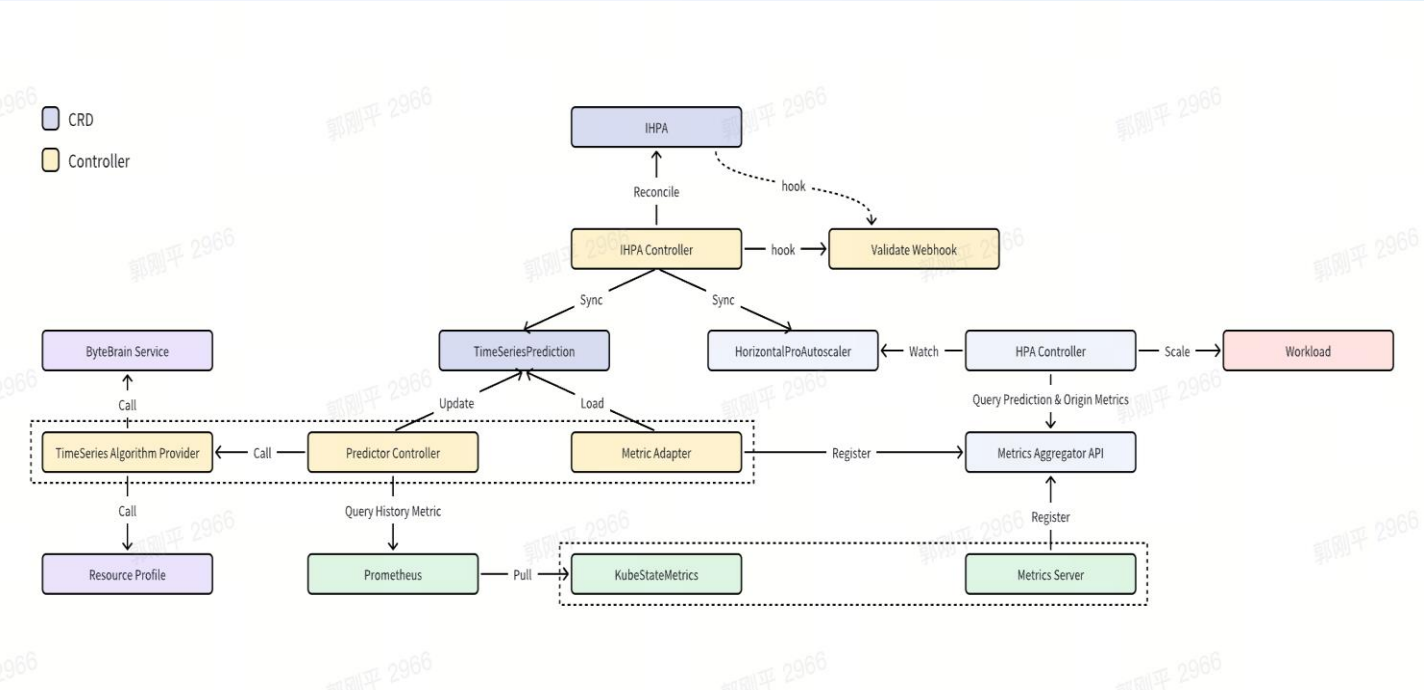
预测式 iHPA

iHPA 为与内场 ByteBrain 团队合作，基于其提供的时序预测模型 实现

iHPA 自身提供对于 Metric 的配置，Metric 配置完成后，Controller 会将每一个 iHPA Metric 转换为两个 HPA Metric，分别对应实时指标和预测指标。

1. 对于实时指标来说，其会直接透传到 HPA Metric；

2. 对于预测指标来说，iHPA Controller 会进行转换：
   – 预测指标类型均为 External，即由 Metric Adapter 提供；



04

总结与展望

总结与展望

总结

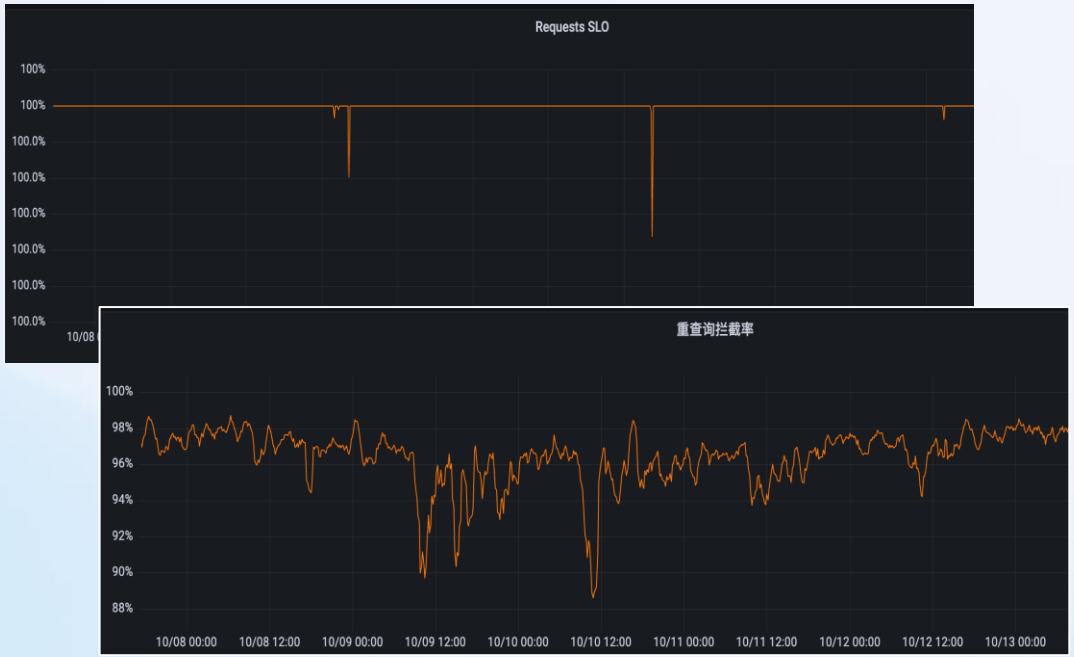
稳定支撑火山引擎方舟数十亿级时序的读写

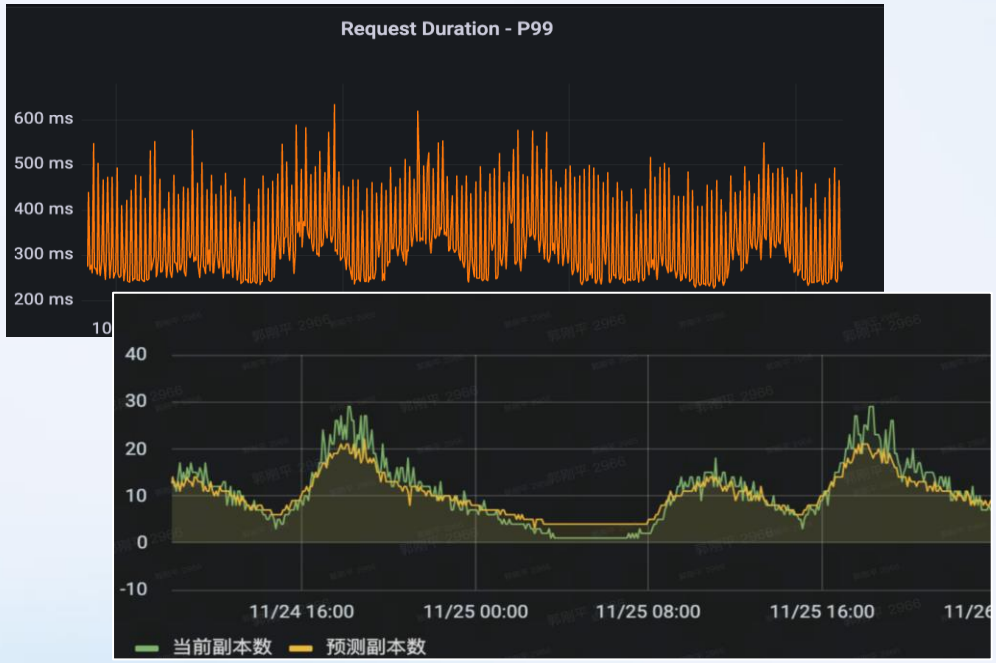
高基数查询P99时延 降低 50%+

重查询 拦截率 95%

读写可用性 99.99%

查询P99时延 600ms





总结与展望

展望

夯实基础，稳字当先

建设下一代更高性能、更低成本的时序存储

持续加强稳定性&运维自动化建设

统一观测，跨层联动

与火山引擎应用观测APMPlus产品联动，统一采集、统一观测，提升跨层数据联动质量

面向 AI，主动洞察

增强主动数据分析能力，内置主动异常检测、趋势预测等能力

加速AIOPS产品化建设，把字节跳动内部成熟能力输出到云上

InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

QCon
全球软件开发大会

InfoQ 极客传媒

THANKS

大模型正在重新定义软件

Large Language Model Is Redefining The Software