

# 图连万物 NebulaGraph 在互联网的应用实践

2025 · 北京站 nMeetup

 2025 年 6 月 28 日

 北京市海淀区中关村创业大街 6 号楼 B1 层

# 关系型数据库没有“关系”？

论图数据库如何重构人际关系的存储逻辑

张潇

# 目录

CONTENTS

01

为何“关系”  
成为数据库的  
阿克琉斯之踵？

02

图数据库：  
存储逻辑的  
重构哲学

03

**NebulaGraph**  
如何重构  
关系存储

01

为何“关系”成为数据库的阿克琉斯之踵？

# 关系型数据库的“关系”悖论

外键≠关系，无语义描述能力，无法表达关系的强度、类型等属性，难以满足复杂关系的存储需求

3度以上JOIN引发性能悬崖，复杂度呈指数增长

customers			
customer_id	customer_name	customer_contact	customer_region
1001	客户 A	contact_a@company.com	华东
1002	客户 B	contact_b@company.com	华南
1003	客户 C	contact_c@company.com	华北
1004	客户 D	contact_d@company.com	西南

managers			
manager_id	manager_name	manager_email	managed_region
2001	张经理	manager_zhang@company.com	华东
2002	李经理	manager_li@company.com	华南
2003	王经理	manager_wang@company.com	华北
2004	刘经理	manager_liu@company.com	西南

manager_assistants			
assistant_id	assistant_name	assistant_email	associated_manager_id
3001	小张助理	assistant_xiaozhang@company.com	2001
3002	小李助理	assistant_xiaoli@company.com	2002
3003	小王助理	assistant_xiaowang@company.com	2003
3004	小刘助理	assistant_xiaoliu@company.com	2004

mer manager relationship		
relationship_id	customer_id	manager_id
4001	1001	2001
4002	1002	2002
4003	1003	2003
4004	1004	2004

```

SELECT
  c.customer_name AS 客户名称,
  m.manager_name AS 客户经理姓名,
  ma.assistant_name AS 助理姓名,
  m.manager_email AS 客户经理邮箱,
  ma.assistant_email AS 助理邮箱
FROM
  customers c
JOIN
  customer_manager_relationship cmr ON c.customer_id = cmr.customer_id
JOIN
  managers m ON cmr.manager_id = m.manager_id
JOIN
  manager_assistants ma ON m.manager_id = ma.associated_manager_id
WHERE
  c.customer_name = '客户 A'; -- 查询客户 A 的客户经理及其助理信息
  
```

# 人际关系的核心特征

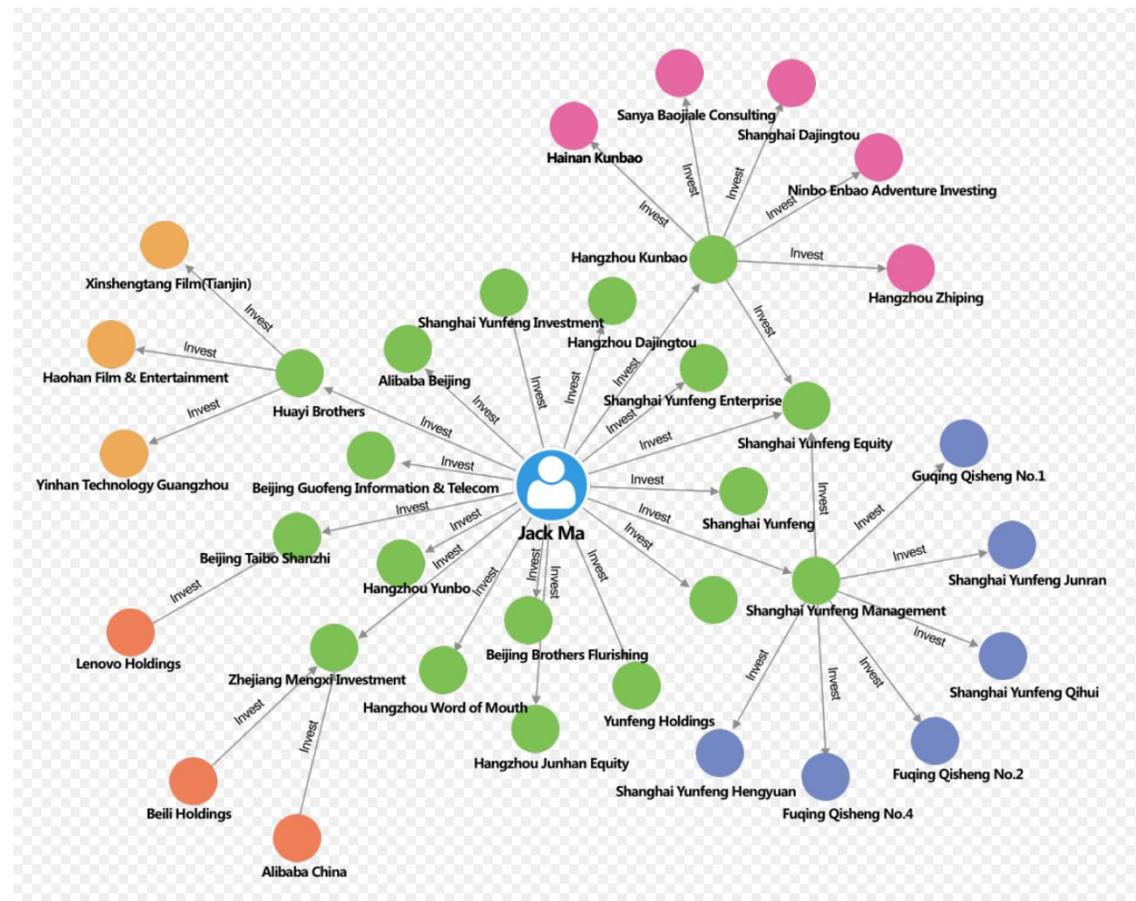


## 动态网状结构

人际关系是动态的网状结构，非静态层级，随时可能发生变化，难以用传统的表结构来准确表示。

## 多维度属性

关系具有多维度属性，如关系强度、类型、时效性等，传统表结构难以完整存储这些信息，导致信息丢失。



02

## 图数据库：存储逻辑的重构哲学

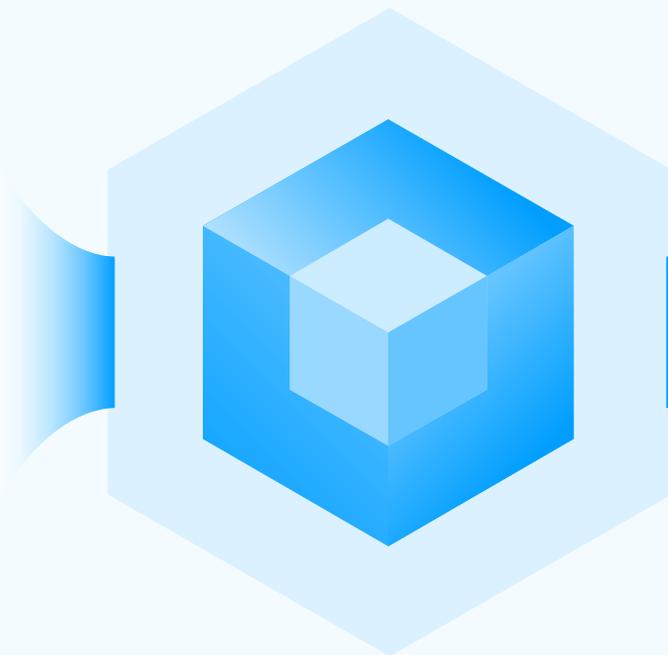
## 模型革命：从表到图

### 属性图三要素

**顶点：**人/物（携带属性），如姓名、年龄等，是关系的主体。

**边：**关系（可携带权重、类型），如亲密度、时间等，是关系的连接。

**属性：**为顶点和边提供详细信息，丰富关系的语义。



### 与关系型的关键差异

**关系表达：**关系型数据库隐式（外键约束），图数据库显式（一等公民），直接存储关系。

**查询模式：**关系型数据库集合操作，图数据库图遍历，更适合关系查询。

# 查询语言：从集合论到图论

## SQL的局限

查询“朋友的朋友”需显式指定JOIN路径，路径固化，难以应对动态关系查询。

## 图查询的突破

声明式路径查询（支持可变深度），路径自由探索（深度/方向/类型过滤），实时响应，对比关系型分钟级延迟。



03

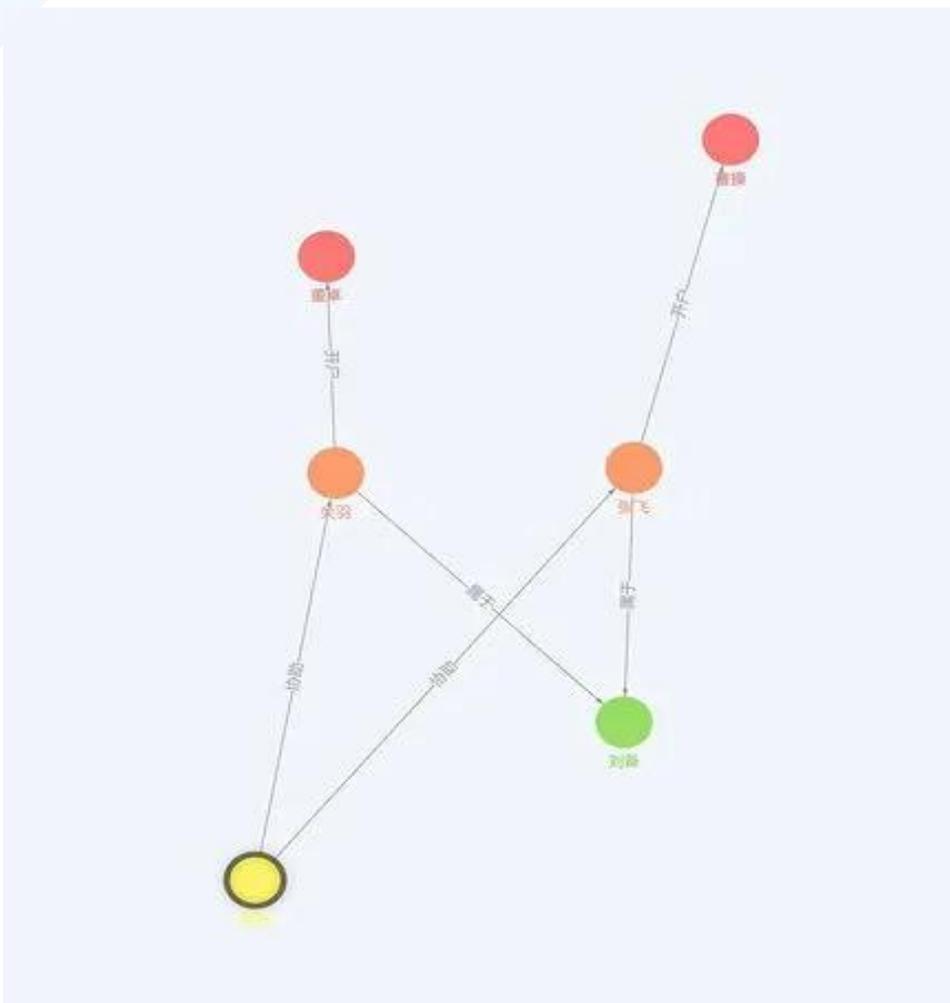
# NebulaGraph 如何重构关系存储

## “关系”悖论的破局

标签 (Tag)	属性 (Properties)	说明	
manager	manager_id,manager_name,manager_email,managed_region	客户经理实体	
assistant	assistant_id,assistant_name,assistant_email,associated_manager_id	客户经理助理实体	
customer	customer_id,customer_name,customer_contact,customer_region	客户实体	
关系类型 (Edge)	起点 → 终点	属性 (Properties)	说明
MANAGED_BY	manager → customer	start_date: date, is_primary: bool	经理服务客户
ASSIST_BY	assistant → manager	department: string	助理支持经理

-- 查询客户A的客户经理及其助理信息

```
MATCH (c:Customer {customer_name: '客户A'})-[:Managed_By]->(m:Manager)-
[:Assisted_By]->(a:Assistant)
RETURN c.customer_name AS 客户名称, m.manager_name AS 客户经理姓名, a.assistant_name
AS 助理姓名, m.manager_email AS 客户经理邮箱, a.assistant_email AS 助理邮箱;
```



### Vertex Details

客户

vid: "曹操"

客户.姓名: "曹操"

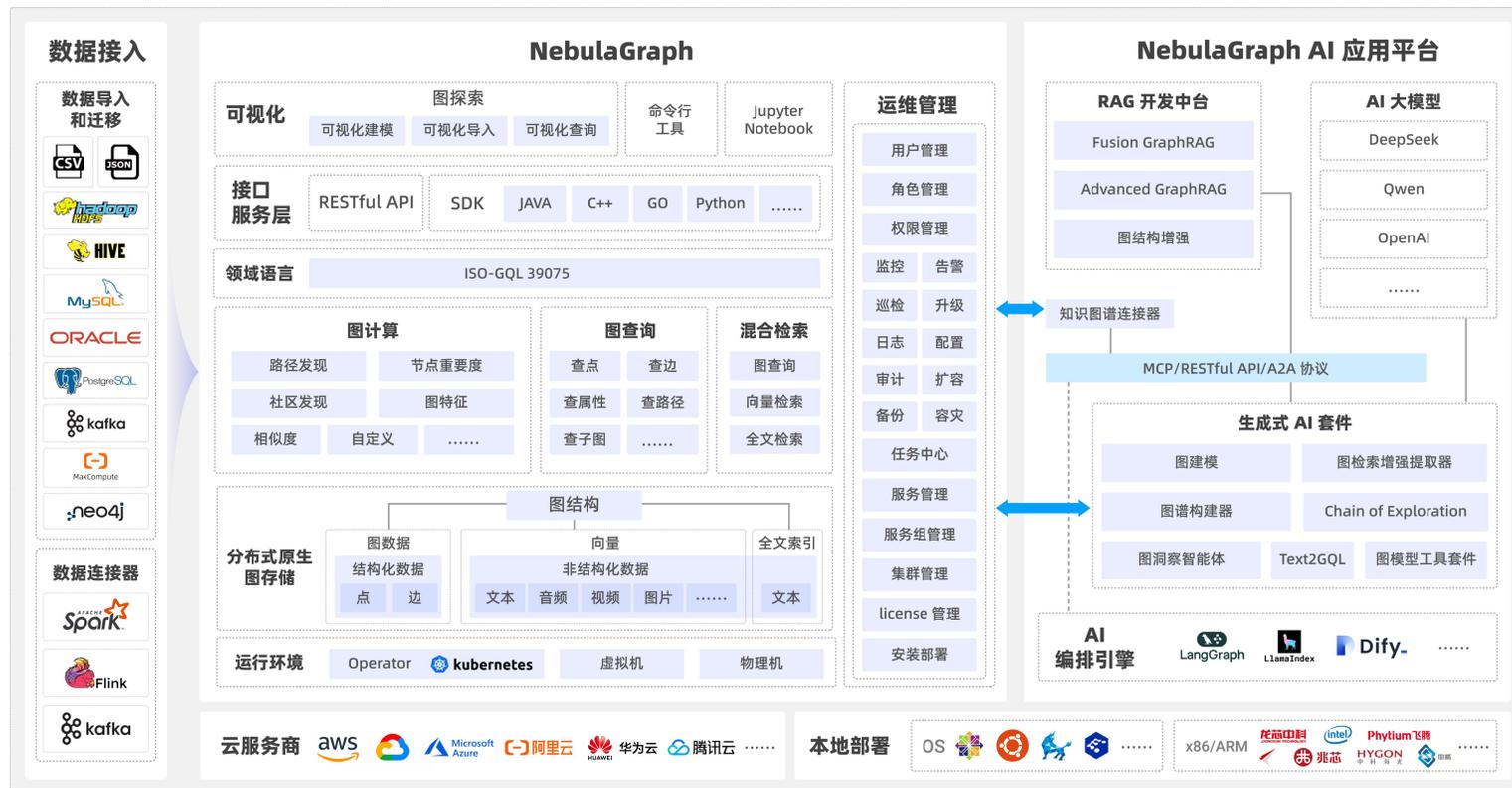
客户.联系方式: "000000"

客户.证件到期: "2025-01-01"

整个团队关系以图的形式展现，支持拖拉拽调整布局。

点击节点，查看具体信息。

# 技术架构优势



能力维度	传统关系型数据库	NebulaGraph
关系查询	JOIN 爆炸，深度查询性能骤降	免索引邻接，邻居访问
扩展性	垂直扩展为主，成本高	存储计算分离，线性水平扩展
数据模型	表结构僵化	属性图灵活建模（点、边、属性）
实时性能	分钟级延迟	毫秒级路径查询

# 重构人际关系逻辑的实践

## • 社交网络分析

- 合规检测：识别洗钱或黑产团伙（设备/IP/账号关联）
- 人际关系分析：关键节点挖掘（高净值客户识别）

## • 金融风险

- 实时反欺诈：闭环交易检测（ $A \rightarrow B \rightarrow C \rightarrow A$ 路径分析）
- 知识图谱构建：企业股权穿透（多层持股关系可视化）

## • Graph+AI

- **GraphRAG**：知识图谱增强大模型推理
- **Text2GQL**：自然语言转查询语句

# Thanks!

## Q&A

-  <https://nebula-graph.com.cn>
-  GitHub: [vesoft-inc/nebula](https://github.com/vesoft-inc/nebula)
-  Twitter: [@NebulaGraph](https://twitter.com/NebulaGraph)
-  Facebook: [@NebulaGraph](https://www.facebook.com/NebulaGraph)
-  <https://discuss.nebula-graph.com.cn>



微信公众号



开源项目

BOSS  
直聘

# 基于 NebulaGraph 构建异常调用 实现根因定位

分享人：万佳飞

# 目录

1

背景&问题

2

技术选型

3

实现方案

4

未来展望

# 背景&问题

## 视角割裂、链路不清

- 监控、日志、链路等数据未打通，缺一视角
- 服务调用链不清晰，跨团队排查无头绪
- 异常无法自动聚合到业务视角或根因层面

## 定位路径无体系

- 排查路径无标准，完全靠人经验
- 根因归因逻辑难沉淀，轮番接锅
- 无自动化辅助手段，工具缺失

## 责任认定无机制

- 故障责任无法基于数据自动判定
- 影响范围和服务边界不清，容易踢皮球
- 故障复盘无抓手，经验难沉淀

问题

结果

## 难以快速定界

- 故障首发难识别，延误响应
- 调查从头开始，重复排查
- 效率低，影响范围被放大

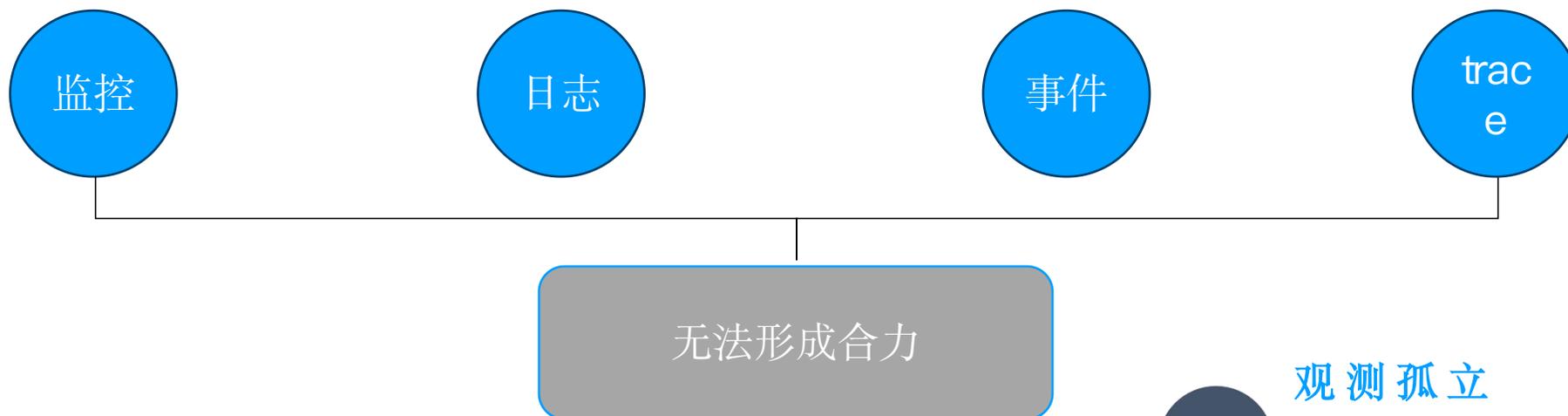
## 故障靠人扛

- 故障定界慢、根因靠猜
- 关键路径问题频繁复现
- 专家负担重，效率难扩展

## 没人对问题兜底

- 定责模糊，团队推诿
- 问题复发无治理
- 稳定性难持续提升

## 背景&问题



### 建设思路

1. 工具整合, 图谱串联
2. 根因建模, 提高效率
3. 快速定位, 及时止损

### 有效解决问题

1. 排障工具多, 使用难度大
2. 故障种类多, 排查方向多
3. 定位根因难, 止损不及时

### 观测孤立

01

这些通用的工具都只是对自身进行观测, 缺乏关联性分析

### 角度片面

02

单一工具只能描述一类现象, 比如监控可能偏趋势观测, 事件偏根因类观测, 日志偏细节定位观测, trace偏链路分析。各有优缺点

# 技术选型

维度	Neo4j	NebulaGraph	对根因定位的影响
数据模型	支持灵活的属性图模型	同样支持属性图，且支持自定义属性、标签及边类型，模型更贴近实际业务结构	都可满足建模需求，但 Nebula 对标签与边类型管理更清晰，适合维护大规模业务图谱
性能与扩展性	单机/集群版，适合中等规模数据	原生分布式，水平扩展性强，支持千万节点、亿级边	根因定位场景下，数据量持续积累，NebulaGraph 可轻松应对未来增长
查询语言	Cypher: 语义表达能力强，学习成本较低	nGQL: 类似SQL，学习成本低，支持图计算表达	两者都足够强大，但 nGQL 更贴近工程实践
时序能力	不擅长处理时序数据	支持边的多版本存储 (TimeRank) 机制	在根因定位中，服务依赖与性能指标有明显时间关联，Nebula 的 TimeRank 支持非常关键
生态与集成	丰富，工具和社区成熟	生态快速发展，兼容 Spark、Flink、Prometheus、Grafana 等	Nebula 在云原生和可观测性集成方面更适配我们的系统
运维与成本	社区版功能受限，企业版授权成本高	完全开源，资源占用少，部署简洁	Nebula 成本更低，利于中台统一建设和横向推广

## 实现方案-部署 NebulaGraph

类别	项目	配置说明
硬件	处理器	Intel(R) Xeon(R) Gold 6342 CPU @ 2.10GHz, 80 核
	内存	DDR4,128G
	存储	2TB NVME
软件	NebulaGraph 版本	v3.8.0
	操作系统	CentOS Linux release 7.3.1611 (Core)
	节点部署情况	3 个 metad, 5 个 graphd, 5 个 storaged (共 5 个节点)

# 实现方案-现象建模

## 水平层

01

建设应用与应用，应用与服务，应用和组件的依赖关系

## 垂直层

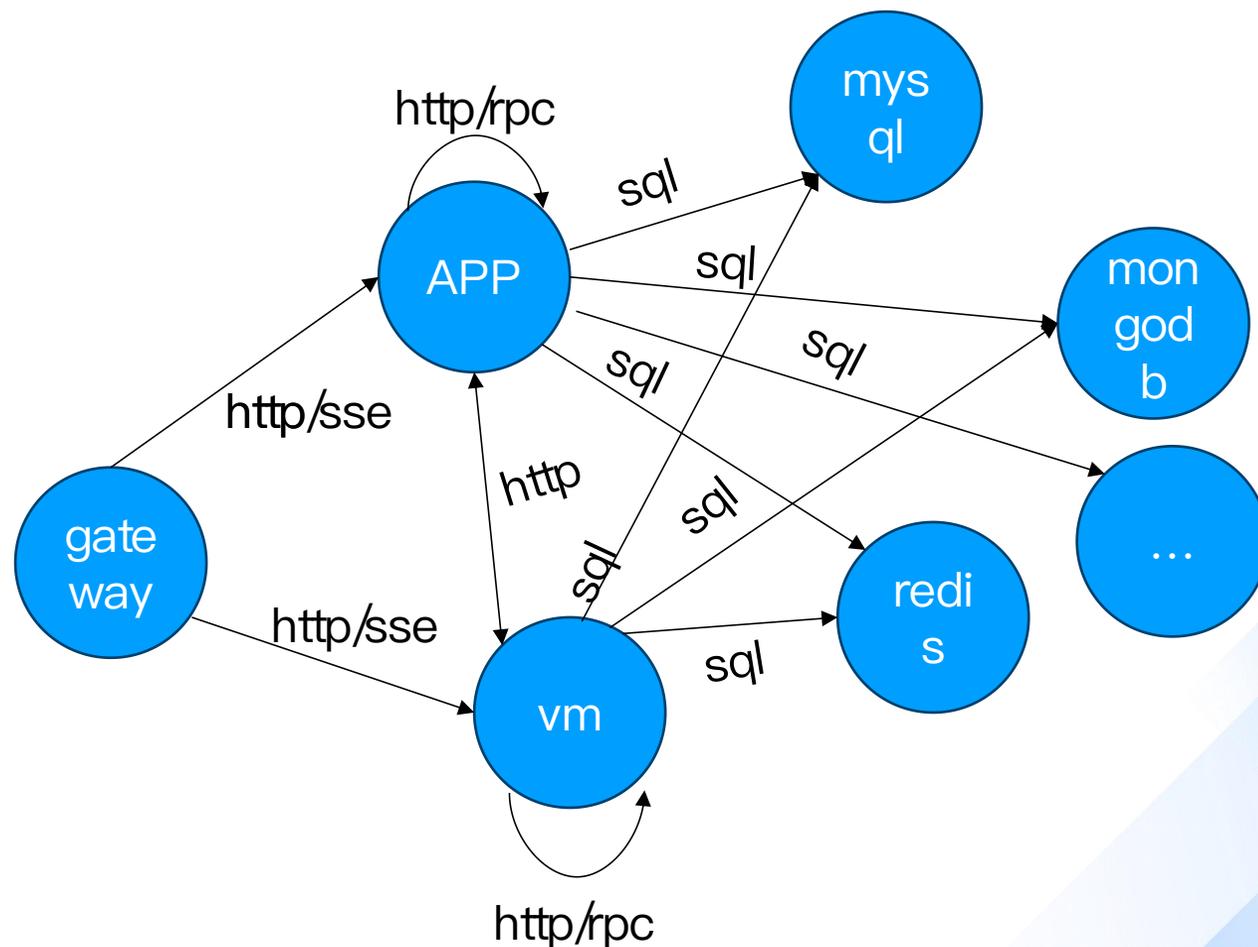
02

建设从网关到应用、从应用到基础设施的依赖关系

## 依赖设计

03

不同实体间的依赖是不同的，主要有http,sse,rpc,tcp,sql等



# 实现方案-根因建模

按照需要进行依赖建模，以符合实际价值需要。

**01 应用层**

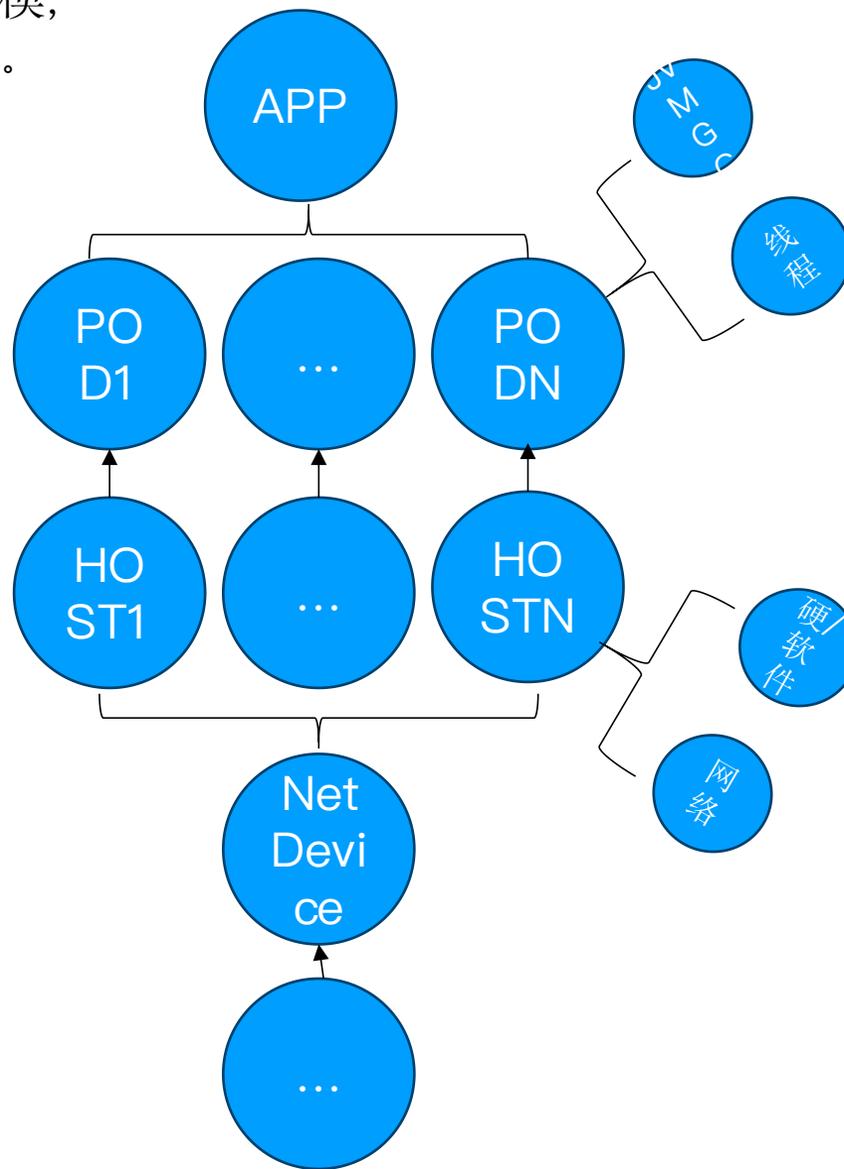
建设应用，服务等根因模型，例如发布变更类、配置类等

**02 软件层**

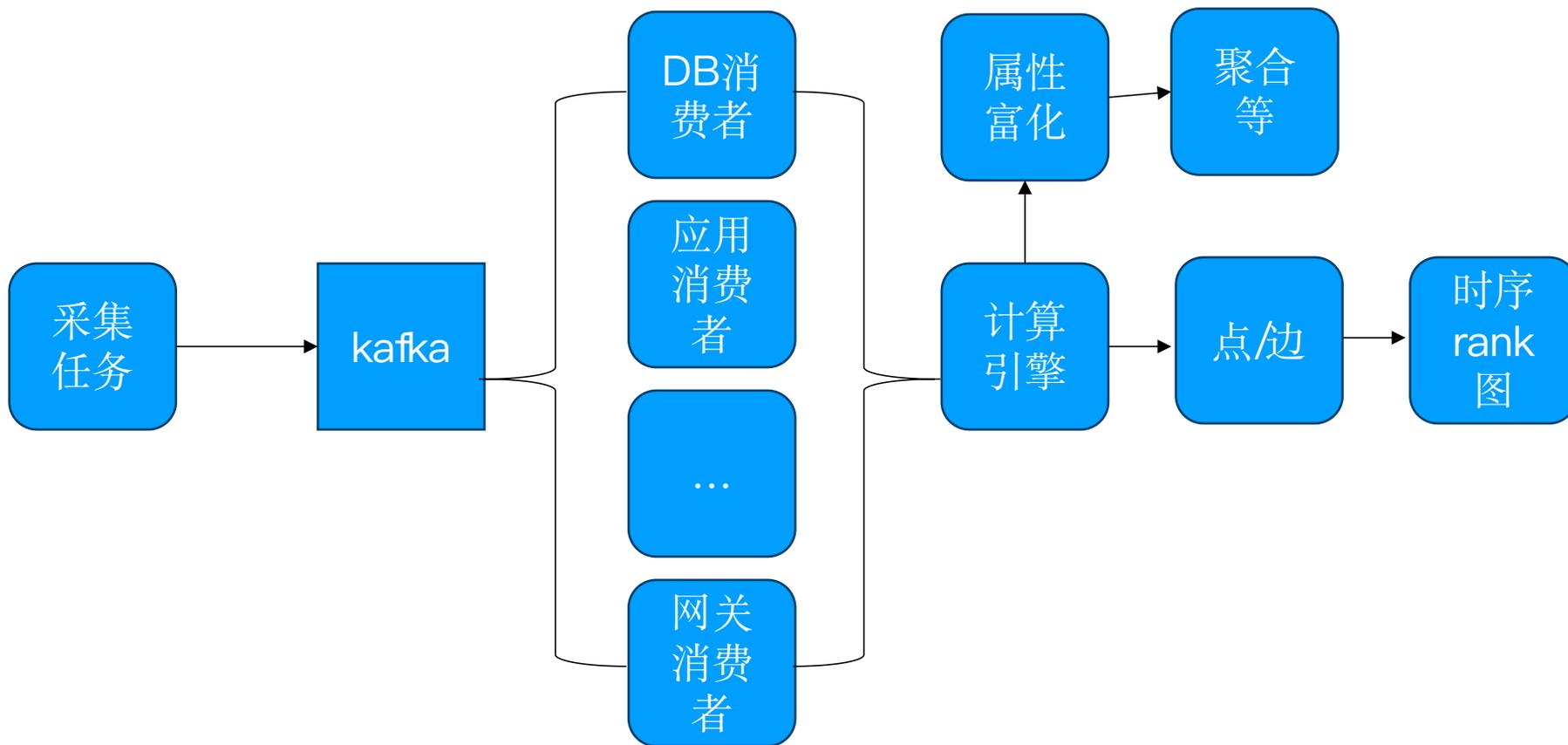
建设容器、kvm等根因模型，例如宕机，CPU，内存等

**03 硬件层**

建设物理机、网络设备等根因模型。例如硬件故障等



# 实现方案-数据采集与存储



# 实施方案-故障定位

实施方案已手动删除  
如有需要可以联系  
BOSS 直聘-小万老师  
邮箱: wanjfwjf@ gmail.com

# 实施方案-根因定位

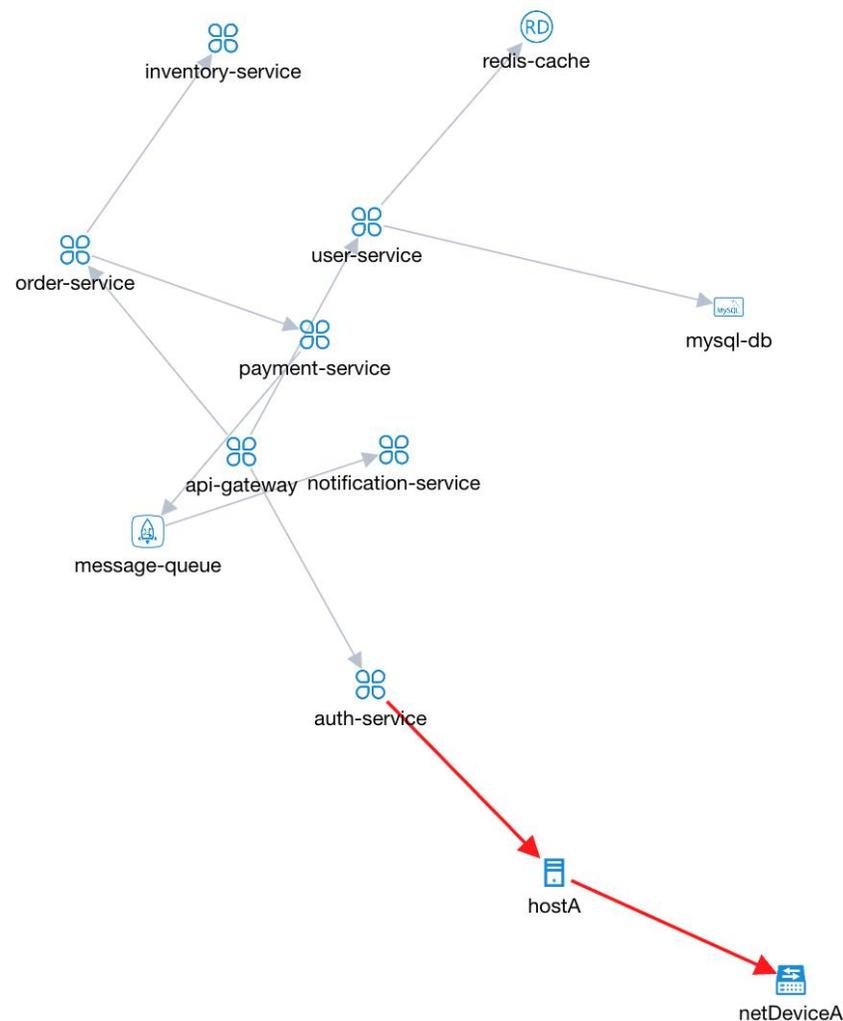
实施方案已手动删除  
如有需要可以联系  
BOSS 直聘-小万老师  
邮箱: wanjfwjf@ gmail.com

## 实现方案-展示

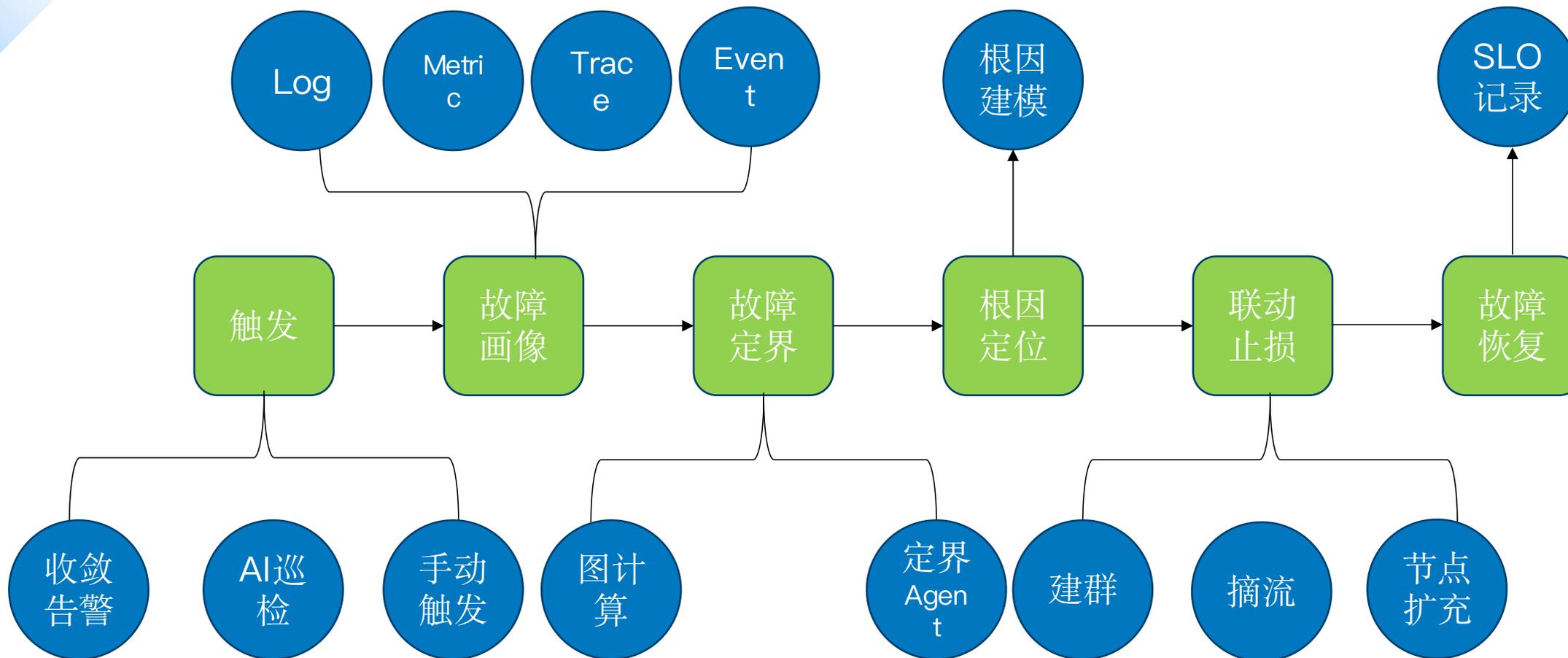
网络设备告警 【网络异常】 设备名称: netDeviceA 异常类型: 网络端口故障 错误信息: Interface eth0/1 is down, link status changed 影响范围: 网络连接中断, 数据传输异常 时间: 2025-06-27 10:05:30

根因: 该图描述了由于网络设备A端口故障, 影响了主机, 进而影响了应用auth-service。

影响: 进而链式传递影响了数个应用, 风暴眼位auth-service。根因位网络设备A



# 实现方案-展示



## 实现方案-收益

严格按照故障响应的三个阶段，用于缩短故障恢复时长MTTR

故障阶段	时长(Max/Avg)	应对措施
故障发现	1 min(平均收敛时间20s)	故障定位，迅速找到根因
故障处理	5min(平均响应时间40s)	根据根因协调负责人进入止损方案制定
故障恢复	10min(平均响应时间2min)	开启止损，恢复业务

# 未来展望

01

## 数据建模增强

覆盖更全面、更细节的建模数据。并预计算与存储，构造高质量建模

02

## Data Agents

专门分析关联形数据结构的AI助手，从关联形数据中提取重点

03

## 根因定位Agent

从海量的日志、监控等中挖掘目前无法发现的根因

BOSS  
直聘

# Thanks! Q&A

-  <https://nebula-graph.com.cn>
-  GitHub: [vesoft-inc/nebula](https://github.com/vesoft-inc/nebula)
-  Twitter: [@NebulaGraph](https://twitter.com/NebulaGraph)
-  Facebook: [@NebulaGraph](https://www.facebook.com/NebulaGraph)
-  <https://discuss.nebula-graph.com.cn>



微信公众号



开源项目

# 大模型与 NebulaGraph 结合的 知识库搭建

中科数睿

张哲源

大模型算法与用工程师

# 背景介绍

---

随着信息技术的飞速发展，各行各业积累了海量的数据和知识资源。然而，这些知识往往以非结构化或半结构化的形式存在，导致信息孤岛、知识碎片化等问题，严重影响了知识的共享和利用效率。因此，构建高效、智能的知识库系统，成为了提升组织决策能力和创新能力的关键。

# 向量 RAG (Vector RAG)

## 原理

- 通过将文档和查询转化为向量，使用向量数据库进行相似度检索，再将检索到的文本片段与查询一起输入大语言模型（LLM）生成答案。

## 优势

- 实现简单，适用于大多数文本检索任务。
- 支持快速的语义匹配。

## 局限性

- 无法精准切片，尤其是按行按段或者按照标点符号切片都会导致上下文丢失。
- 大模型并没有参与到文本检索的过程中，仅仅在向量库检索的答案中。
- 如果涉及到全局问题，比如领导想问某个工作组的这周完成的事项。向量rag只能检索一部分员工的工作事项。

# GraphRAG

## 1、原理：

构建知识图谱，将实体及其关系表示为图结构，在检索阶段通过图遍历获取相关子图，提供更丰富的上下文信息供 LLM 生成答案。

## 2、优势：

能够捕捉实体间的多层次关系，提升推理能力。在处理复杂查询和多跳推理时表现优异。减少模型幻觉，提高生成内容的准确性和一致性。

## 3、局限性：

构建和维护知识图谱成本较高。  
图遍历和子图检索计算开销较大，可能影响响应速度。  
指标数据难以维护和修改。比如我想修改整个图关于某个指标比如成绩权重的分布，那么可能就需要人工写 cypher 去核对。



## 我司各小组上周工作情况

### 思考过程

“嗯，用户询问了各小组的上周工作情况。根据 Nebula 知识图谱中“每周二至下周一”的汇报机制，当前时间为2025年6月9日星期一，本次筛选的时间范围就定为 2025年6月3日至6月9日。接下来，我将严格按照 Nebula 知识图谱设计的分析路径依次完成以下步骤：首先，解析各小组周报的文档元数据，确定报告的日期范围、撰写人和标题，以明确本次分析的数据基础；其次，构造并执行 Nebula 的 nGQL 查询；随后，构造并执行 MySQL 的 SQL 查询；接着，我会对获取到的数据进行清洗与对齐，统一数值精度与格式，以便于后续的文本处理和阅读理解；紧接着进行文本格式化处理，将数据结构化地呈现为清晰易读的报告形式；最终，我会结合以上全部步骤的结果，输出一份内容精准、格式规范、逻辑清晰的最终报告，以支持各组的工作进度讨论与决策制定。

1. 解析文档元数据：读取三个周报文件的日期范围、汇报人及报告标题

- NLP组：2025-06-03至2025-06-09，由张三撰写
- 强化学习组：2025-06-03至2025-06-09，由王五撰写
- 计算机视觉组：2025-06-03至2025-06-09，由李四撰写

2. 构造并执行 Nebula nGQL 查询获取本周所有模型信息：

```
Nebula 返回列表 = [
  {'model':'NL2SQL','group':'NLP'},
  {'model':'RAG','group':'NLP'},
  {'model':'TransformerRL','group':'RL'},
  {'model':'MAML','group':'RL'},
  {'model':'MultiAgent','group':'RL'},
  {'model':'ConvNeXt','group':'CV'}
]
```

3. 构造并执行 MySQL 查询获取核心指标：

```
MySQL 返回 = {
  'NLP': {
    'data_internal':5000,'data_spider':10000,
    'nl2sql_acc_increase':0.08,'cot_acc_increase':0.12,
    'rag_docs':2000,'rag_chars':1500000,'rag_precision@10':0.87,'rag_recall@10':0.92
  },
  'RL': {
    'transformer_speedup':0.20,'representation_improve':0.18,'adaptation_speedup':3.0,
    'multiagent_success':0.92,'baseline_success_rate':0.95,'avg_reward_increase':0.12
  },
  'CV': {
    'images_augmented':5000,'top1':0.887,'focal_improve':0.05,
    'size_reduction':0.40,'speedup_tensorrt':1.30
  }
}
```

4. 数据清洗与对齐：百分比保留一位小数，倍数保留一位小数

5. 文本格式化：直接填充数据字段，不做额外分析

6. 输出最终报告：

```
### 最终报告
【NLP 组】
- 内部数据量：5000；Spider 扩展数据量：10000
- NL2SQL 准确率提升：8.0%；CoT 提升：12.0%
- RAG 文档数：2000；字符数：1500000；Precision@10: 87.0%；Recall@10: 92.0%

【强化学习组】
- Transformer 训练速度提升：20.0%
- 表征质量提升：18.0%；环境适应速度提升：3.0x
- 多智能体成功率：92.0%；基线成功率：95.0%；平均奖励提升：12.0%

【计算机视觉组】
- 增强图像数量：5000
- Top-1 准确率：88.7%；Focal Loss 提升：5.0%
- 模型体积减小：40.0%；TensorRT 推理加速：1.3x”
```

### 2025年6月3日至6月9日 各小组工作进展最终报告

#### NLP 组

#### 数据规模：

- 内部数据量：5000
- Spider 扩展数据量：10000

#### 模型性能提升：

NL2SQL 准确率提升：8.0%

我们思考过程的  
两大优势：

1.思考依托

NebulaGraph 构建的知识图谱

2.思考给客户看到完整的过程以及数据，但没有

具体库表、SQL、NGQL

适应速度提升: 3.0×\n- 多智能体成功率: 92.0%; 基线成功率: 95.0%; 平均奖励提升: 12.0%\n\n【计算机视觉组】\n- 增强图像数量: 5000\n- Top-1 准确率: 88.7%; Focal Loss 提升: 5.0%\n- 模型体积减小: 40.0%; TensorRT 推理加速: 1.3×"

## 2025年6月3日至6月9日 各小组工作进展最终报告

### NLP 组

#### 数据规模:

- 内部数据量: 5000
- Spider 扩展数据量: 10000

#### 模型性能提升:

- NL2SQL 准确率提升: 8.0%
- CoT 准确率提升: 12.0%

#### RAG 相关指标:

- 处理文档数: 约2000
- 处理字符数: 约1500000
- Precision: 87.0%
- Recall: 92.0%

### 强化学习组

#### Transformer 模型:

- 训练速度提升: 20.0%

#### 模型优化效果:

- 表征质量提升: 18.0%
- 环境适应速度提升: 3.0 倍

#### 多智能体系统表现:

- 多智能体成功率: 92.0%
- 基线成功率: 95.0%
- 平均奖励提升: 12.0%

### 计算机视觉组

#### 数据处理:

- 增强图像数量: 5000

#### 模型性能指标:

- Top-1 准确率: 88.7%
- Focal Loss 改进: 5.0%

#### 模型优化效果:

- 模型体积减小: 40.0%
- TensorRT 推理速度提升: 1.3 倍

我们回答的两大优势:

1.统筹图社区的知识点, 更适合领导进行规划

2.回答统一关系型数据库、图数据库和文档的内容

我们思考流程可解释，比如上面的意图为查询流程，我们会让大模型按照 NebulaGraph 中涉及的步骤执行。确保大模型精确执行

The screenshot shows the NebulaGraph Studio interface. The top navigation bar includes the NebulaGraph logo, 'Schema', '导入' (Import), '控制台' (Console), and 'Schema 草稿' (Schema Draft). The main area displays a query execution result for the query: `[查询流程]> MATCH (n)-[r]->(e) RETURN n,r,e;`. The result is shown in a table view with two columns: 'n' and 'r'. The table contains five rows of data, each representing a step in a process flow.

n	r
("step1":Step(step_name: "明确数据基础"))	[next "step1"->"step2" @0 {name: "next"}]
("step4":Step(step_name: "对获取到的数据进行清洗与对齐"))	[next "step4"->"step5" @0 {name: "next"}]
("step2":Step(step_name: "构造并执行Nebula的nGQL的查询"))	[next "step2"->"step3" @0 {name: "next"}]
("step5":Step(step_name: "文本格式化处理"))	[next "step5"->"step6" @0 {name: "next"}]
("step3":Step(step_name: "构造并执行Mysql的sql的查询"))	[next "step3"->"step4" @0 {name: "next"}]

# 今天的主角—大模型+ NebulaGraph +Mysql 的 RAG

## 模型微调

工欲善其事必先利其器，我们对模型进行了两个部分微调。

能力	技术要点	目的
① 动态链路规划	对 DeepSeek 等模型做轻量指令微调，加入“反思-预估”prompt： → 模型先估计问题复杂度 $C$ ，再用规则 $N=f(C)$ 决定要展开的思考链 (Chain-of-Thought) 长度	既避免“大材小用”，又保证复杂任务有足够推理步
② 占位符式调用	训练模型在遇到 <code>function_call</code> / 外部 MCP 服务时，以 <code>§FUNC_i§</code> 等占位符输出，并返回参数草稿	思维过程与外部计算解耦： <ul style="list-style-type: none"><li>• 减少幻觉</li><li>• 前端可异步并行调用</li><li>• 日志中保留完整推理链</li></ul>

# 今天的主角—大模型+NebulaGraph+Mysql 的 RAG

模型微调的后端效果展示

01

02

极其简单的问题直接回答，同时仍可以在ollama和vllm上进行跑。

```
root@e86a32d6b6c3:/# ollama run deepseek-r1_zheyuan:32b
>>> 1+1=?
<think>

</think>

1 + 1 = **2**.
```

```
C:\Users\wzzha>curl -X POST "http://127.0.0.1:8000/complete" -H "Content-Type: application/json" -d "{\"messages\": [{\"role\": \"user\", \"content\": \"公司的周报给我总结一下?\"}]}\"
{\"id\": \"chatcmpl-xyz-nonstream\", \"object\": \"chat.completion\", \"created\": 1749640295, \"model\": \"deepseek-reasoner\", \"choices\": [{\"index\": 0, \"message\": {\"role\": \"assistant\", \"content\": \"本周三组工作总结：\\n\\n **NLP 组** 完成“模块 A”迭代，详细维度见 $GRAPH_URL；最新 NL2SQL 准确率提升约 $SQL_URL[nlp_accuracy]；\\n\\n **强化学习组** 基于 Transformer 策略网络，多智能体完成率达到 $SQL_URL[rl_success]；\\n\\n **计算机视觉组** 选用 ConvNeXt 模型 (Top-1: $SQL_URL[cv_accuracy])，推理速度提升约 $SQL_URL[cv_speedup]；\\n\\n建议将这些 $GRAPH_URL 和 $SQL_URL[...] 嵌入监控大屏，实时拉取并替换占位符，以自动化生成每周报告。\", \"reasoning_content\": \"我已经知道各组的实体和关系，上一步应从图数据库拉取维度详情 ($GRAPH_URL)；再从 SQL 报表获取最新指标，如 NLP 准确率 ($SQL_URL[nlp_accuracy])、RL 完成率 ($SQL_URL[rl_success])、CV 推理加速比 ($SQL_URL[cv_speedup])；最后将两部分数据整合到回答里。\", \"finish_reason\": \"stop\", \"logprobs\": null}}]}\"
C:\Users\wzzha>
```

# 核心流程

## 文本分块 & 第一轮抽取

- 固定大小的文本块送入大模型，输出三等长数组，组成〈实体 1, 关系, 实体 2〉三元组。

## 不完整句检测

- 若发现句子未完或出现代词，取最近 k 个三元组作为上下文，拼接后续文本。

## 第二轮解析--补充上下文

- 模型在新上下文中完成指代消解，再次抽取并与第一次结果去重、合并。

## 分层存储 --不用在担心维度数据难以修改的问题

- 实体节点、关系与维度属性写入Nebula；数值指标落到 MySQL。

## 流式循环

- 写库后立即清空数组、复位下标，进入下一文本块，保持解析流水线。

## 动态检索 --抑制模型幻觉

- 用户提问时，LLM 先评估问题复杂度，在Nebula中查询是否存在相应实体，
- 再决定是否调用向量检索、图查询或外部 API。

## 占位符输出--抑制模型幻觉

- 推理中遇到外部调用，模型用 `§GRAPH§`、`§FUNC§`、`§MYSQL§` 等占位符输出草稿答案。

## 回填呈现

- 前端并行执行实际查询，将结果回填占位符，生成完整、可解释的最终回答。

## 优势与局限性

01

**优势:适合对准确率和速度要求较高的领域(金融、政务、供应链)**

- 1.减少模型幻觉 —— 模型只能回答知识图谱中存在的实体问题+占位符回填
- 2.检索并行加速 —— 生成与外部调用同步执行
- 3.维度和指标数据准确率几乎为100%——占位符回填。
- 4.支持多跳推理与复杂查询 —— NebulaGraph 子图遍历
- 5.任一组件可独立迭代 —— 检索或指标 API 随时替换

02

**局限性:不适合趋势分析、创意写作、军事等领域**

- 1.无法处理回答类似发展趋势之类的问题——模型生成过程中采用的是占位符。
- 2.内存溢出风险——人物传记等长指代场景
- 3.前期微调与建库成本高 —— Token 与时间耗费大
- 4.系统运维复杂度高 —— LLM、Mysql、NebulaGraph 都需要维护
- 5.隐私与权限治理更困难 —— 占位符回填需要精细脱敏

# Thanks!

## Q&A

-  <https://nebula-graph.com.cn>
-  GitHub: [vesoft-inc/nebula](https://github.com/vesoft-inc/nebula)
-  Twitter: [@NebulaGraph](https://twitter.com/NebulaGraph)
-  Facebook: [@NebulaGraph](https://www.facebook.com/NebulaGraph)
-  <https://discuss.nebula-graph.com.cn>



微信公众号



开源项目

# 图连万物 NebulaGraph 在互联网的应用实践

2025 · 北京站 nMeetup

 2025 年 6 月 28 日

 北京市海淀区中关村创业大街 6 号楼 B1 层

## 茶歇环节

5 ~ 10 min 后精彩继续↓

- NebulaGraph 在阿里妈妈营销场景的应用实践
- NebulaGraph 在京东物流一站式分析平台 UData 的探索与实践



本场活动群聊  
禁广告 仅用于活动交流

# NebulaGraph 在阿里妈妈 营销场景的应用实践

承开（宋铭涛）

1. 自我介绍
2. 引入NebulaGraph的背景
3. 业务实践 Case
4. 服务架构
5. 遇到问题和目前的解决办法
6. 未来规划

## 1. 自我介绍

我是承开，来自阿里妈妈营销引擎团队，我们团队深耕广告领域，依托淘内和全网海量的用户行为数据，不断完善和构建一套基于阿里大数据，可以支持广告主实时进行投放策略智能洞察分析、投放人群定向、投放效果优化分析的交互式分析系统，合作方众多，面临场景case复杂，技术挑战大。

## 2.引入 NebulaGraph 的背景

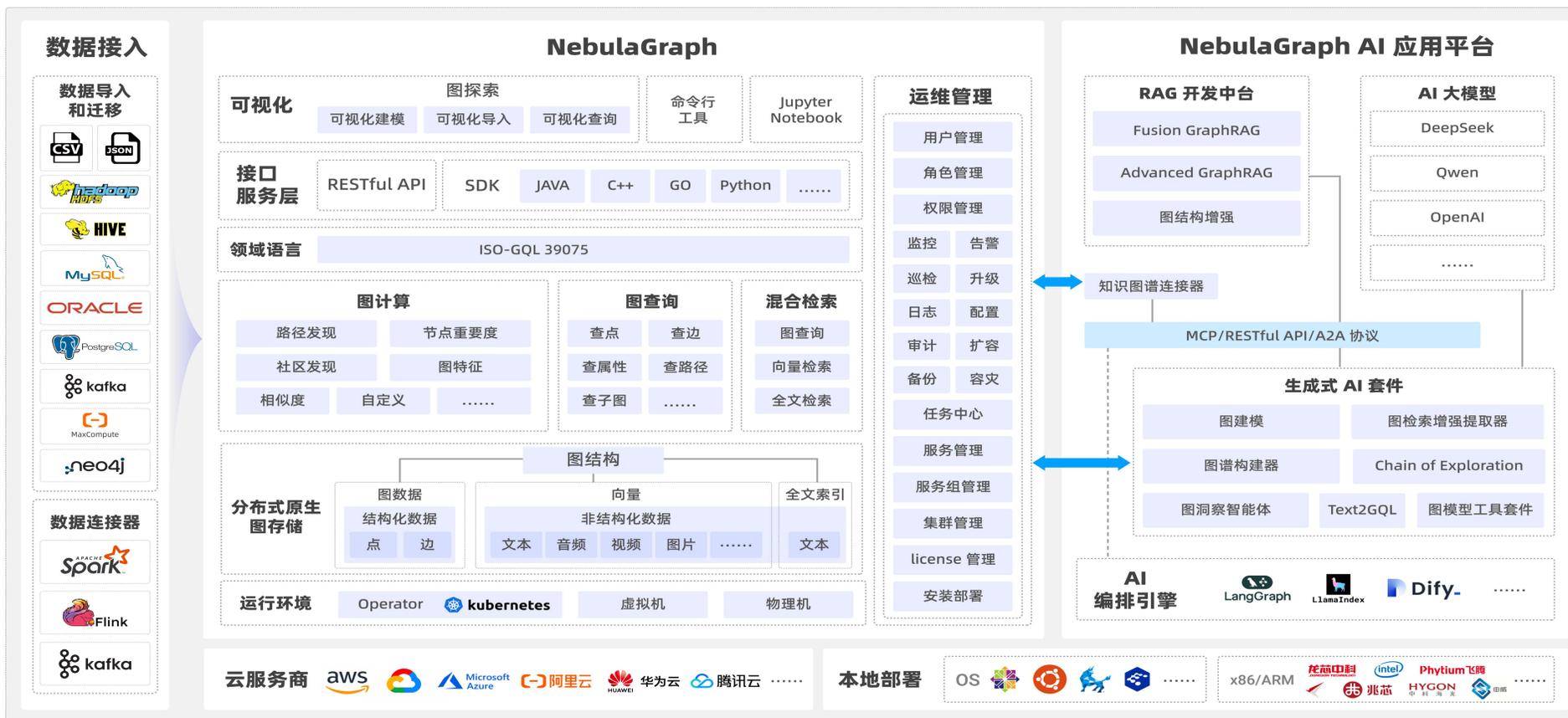
2024年末尾，我们和相关团队对某些问题场景进行沟通，发现很多可以方便使用图数据使用的场景，后续又有大模型的RAG方面也对图数据有诉求。

因此我们对 Neo4j、BlazeGraph、DGraph、NebulaGraph 等相关数据进行测试和调研。

最终我们选择了 NebulaGraph

# 2.1 为什么选择 NebulaGraph

1. 我们测试使用了 Neo4j、DGraph、TuGraph 等这些同类竞品，NebulaGraph 综合性能优秀
2. 架构的优势，NebulaGraph 具备存储和计算的横向扩展能力，更加符合我们场景下的使用需求。
3. NebulaGraph 的基建相对的更加丰富，可以参考和整合、以及二次进行开发

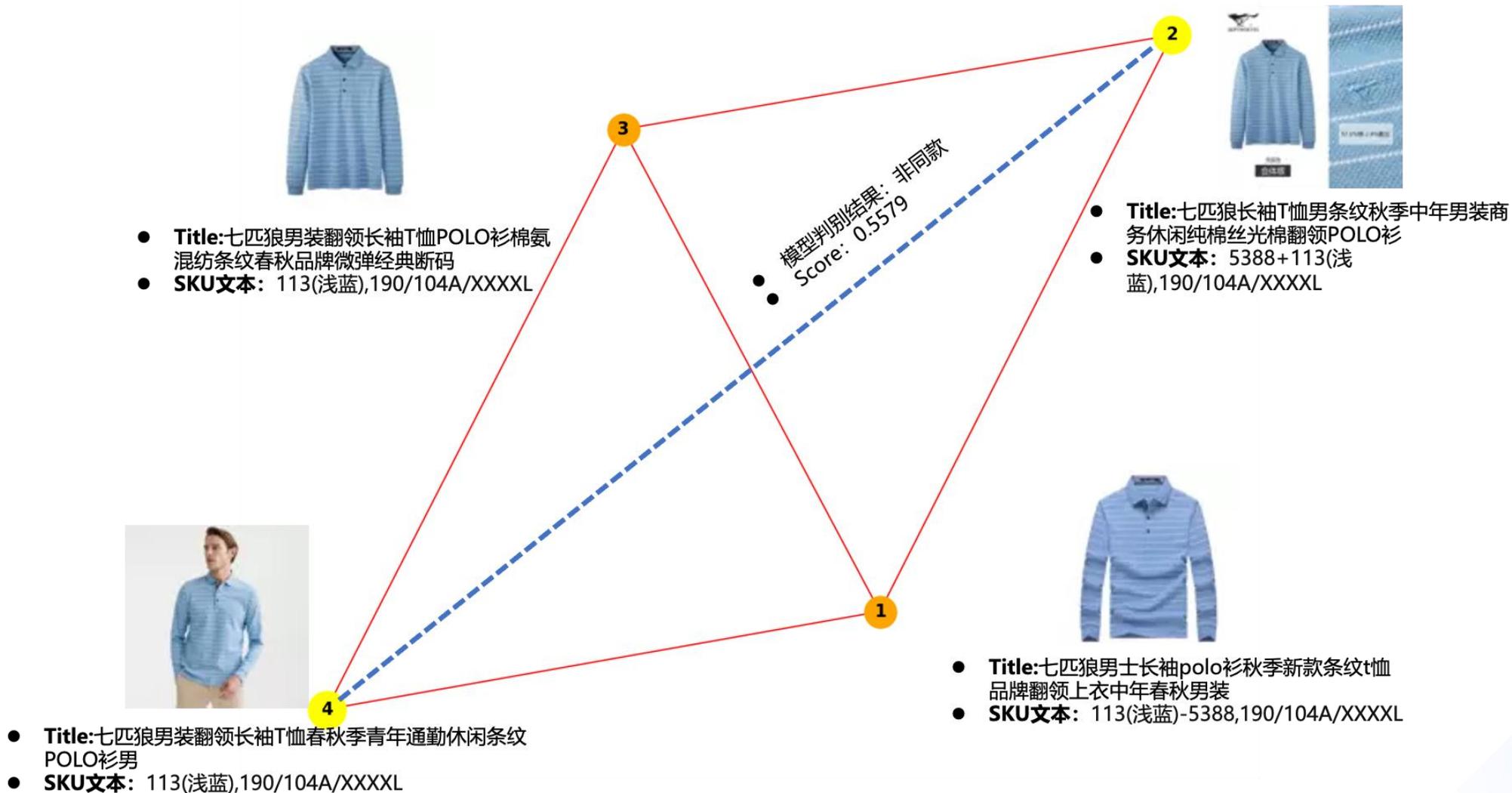


### 3.业务实践-同款异常点检测

淘宝或者天猫的需求场景中，同款和相似款的应用场景很多，其中一个实现方式是，使用分类器对于同款商品集做分类提取，在算法理想 work 的情况下，同款商品簇内的两两商品直接互为同款。

但是现实是算法模型本身的运行设置、机器运算、数据问题等原因导致误差传递。进而使得本应该判断为同款的商品，得到置信度很低的值，成为异常点。

### 3.业务实践-同款异常点检测



## 同款异常点检测

### • Schema

- 点 (Vertex) : 代表一个商品
- 点 (Vertex) Tag: 点的属性, 可以包括商品多模态表征、图像表征等
- 边 (Edge) : 商品之间是否为同款, 已经对应的 score, 后续可以添加图像相似度等信息

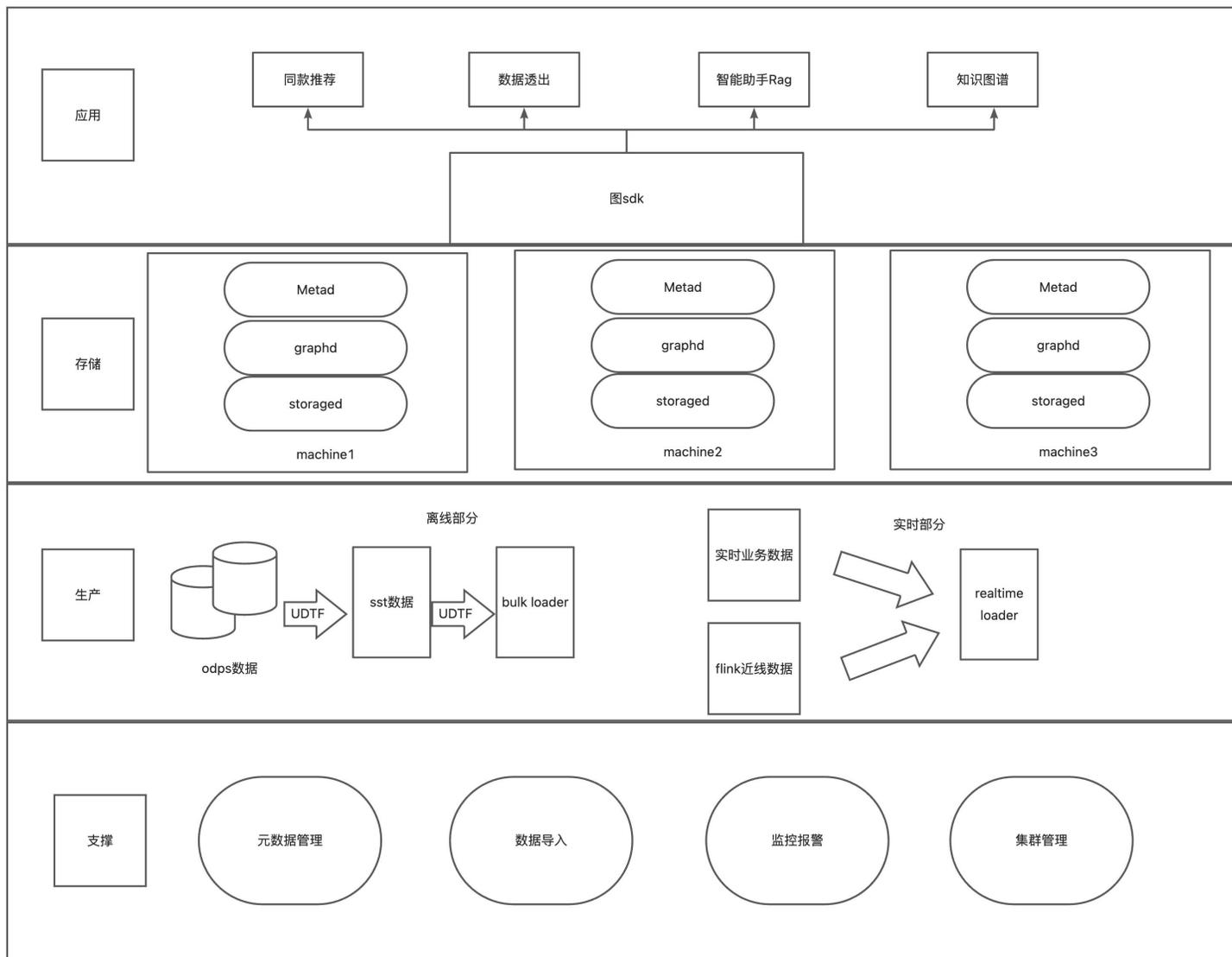
### • k-hop子图查询

- 输入: 待预测的pair
- 输出: k-hop子图 (主要关注1-hop和2-hop) 和子图节点的feature
- 目前已完成小批量图数据库构建, 单个pair的子图提取。

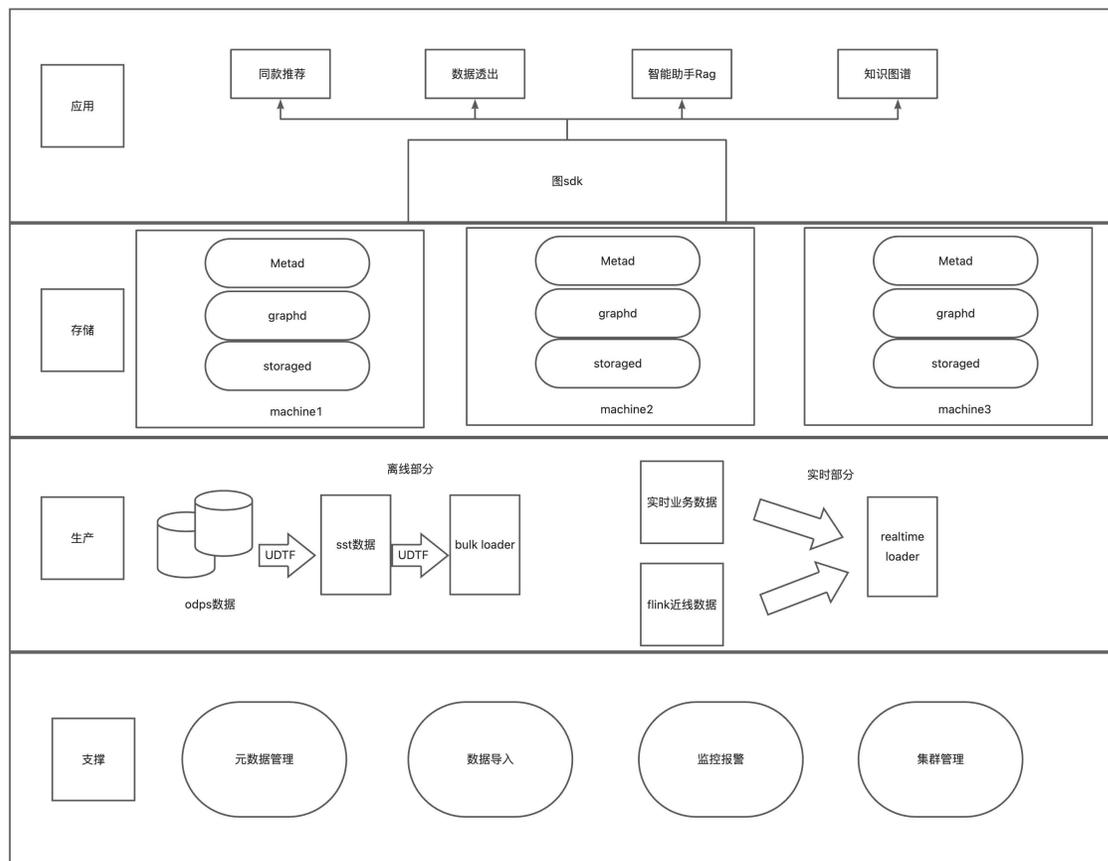
## 其他 case

1. GraphRAG
2. 风控算法团队

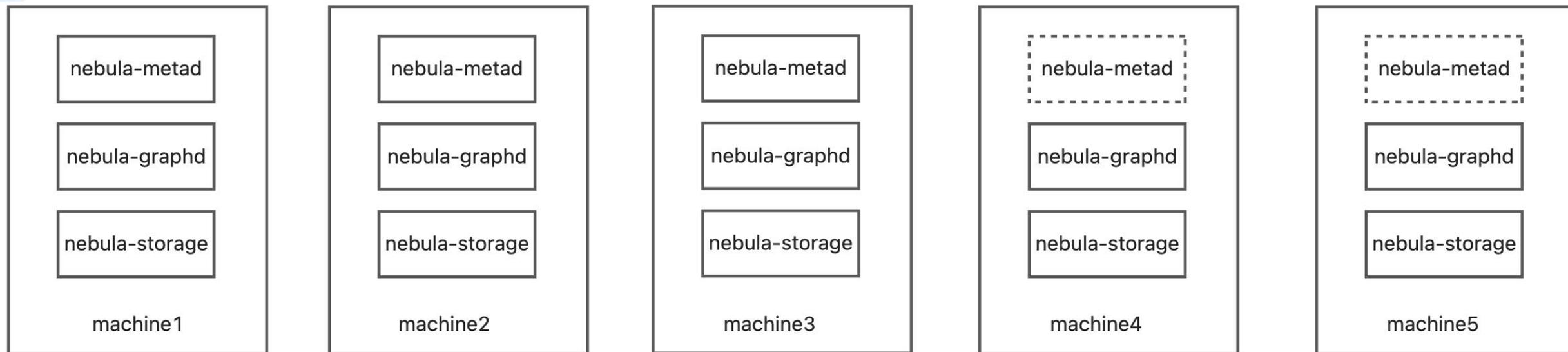
# 4. NebulaGraph 在妈妈实践的架构分层



- 应用**：提供图的sdk，业务方可以在业务服务中引入实时地对图数据进行业务操作
- 存储**：基于hippo(k8s) 的高可用服务物理集群和服务系统。
- 生产**：这里主要进行导入图相关数据的处理。图数据主要有两种来源，第一种是业务方把odps数据etl成转成点和边的表，然后离线导入到图数据库中；第二种是业务线上实时产生的数据、或者通过Flink 等流式处理产生的近线数据，调用在线批量写接口实时灌到图数据库中。
- 支撑平台**：提供了元数据管理、数据导入管理、监控报警、集群管理等依赖功能。



## 4.1 主要存储 (NebulaGraph 集群)

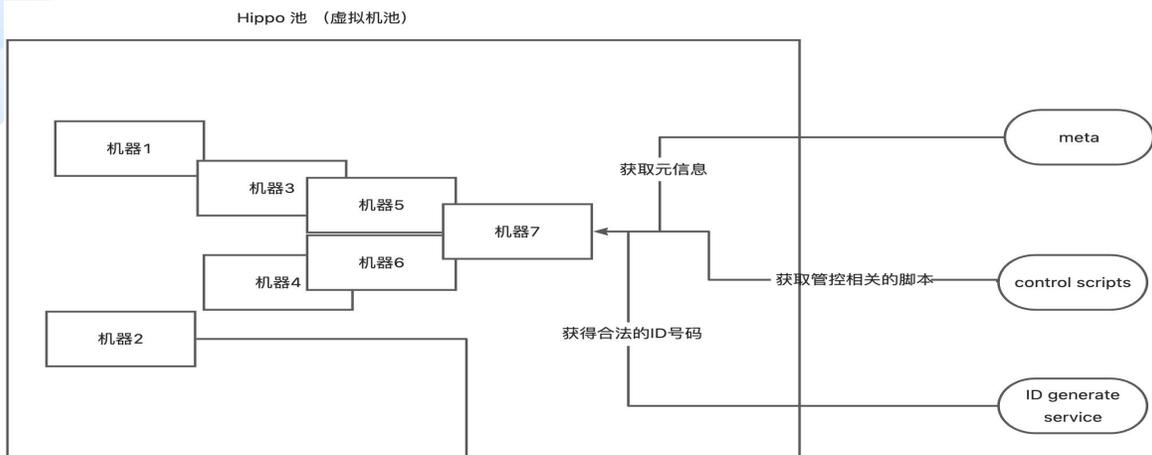


NebulaGraph 集群如何灵活组织编排上面的进程?

哪些机器是 GraphMeta, 哪些是 NebulaGraph? 哪些是 NebulaGraph?

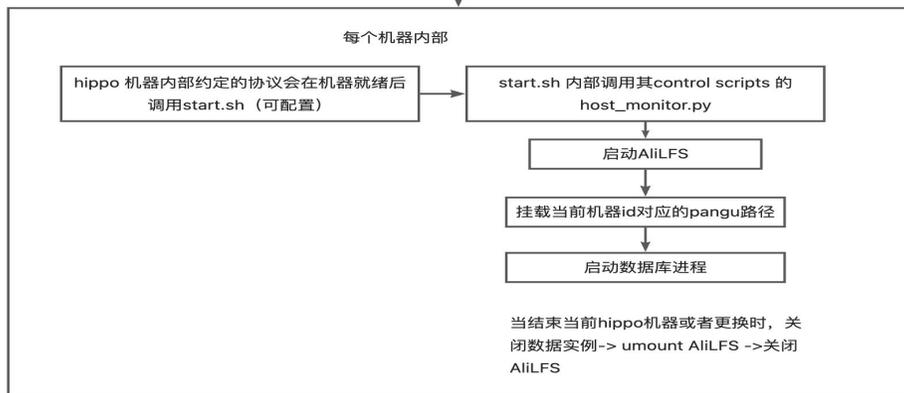
机器都是 hippo(docker) 拉起的机器

## 4.2 集群管控系统



```

"meta_server_shards": "0,1,2,3,4",
"graph_server_shards": "5,6,7,8,9,10,11,12,13,14,15,16,17,18,19",
"storage_server_shards": "5,6,7,8,9,10,11,12,13,14,15,16,17,18,19"
    
```



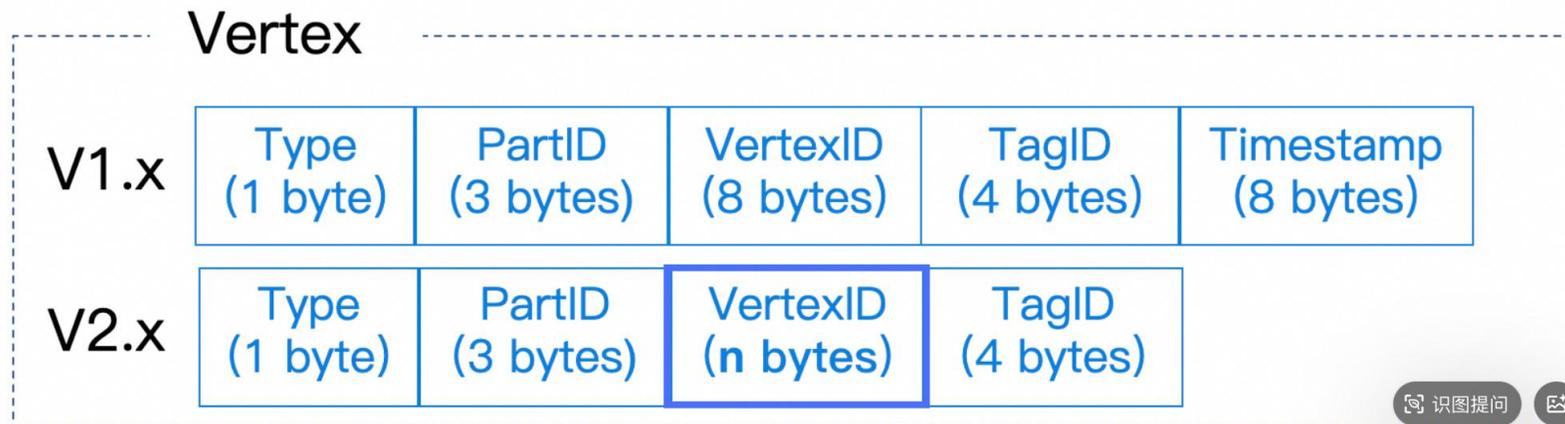
通过管控系统我们会为每个拉起的机器分配一个标识  
 这里使用数字 id 标识, 通过组织编排 id 就可以灵活组建  
 出来不同规格的机器进而满足不同的诉求

## 4.3 存储和计算节点分离

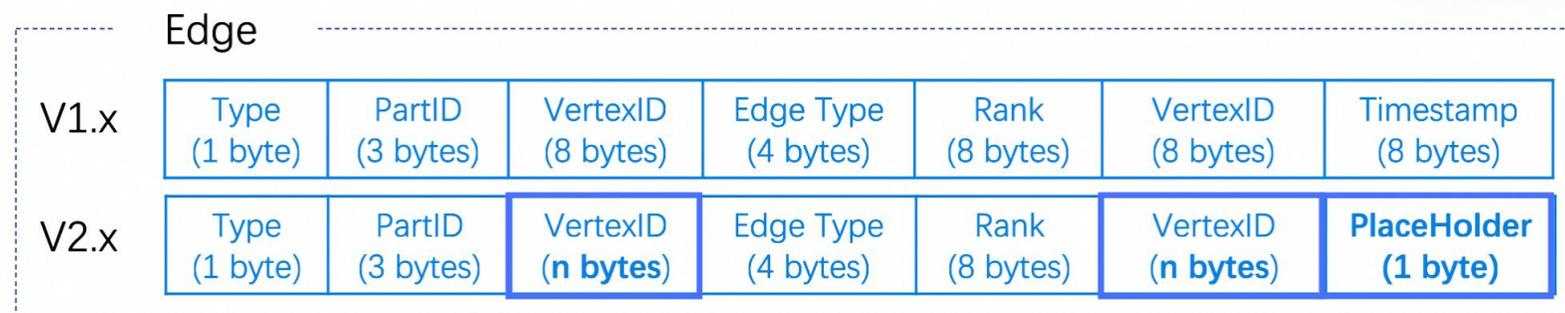
这些运行的集群都是 hippo 拉起的机器，中间调度系统都好几层。如何保存数据和恢复服务。

我们的架构利用中间件alilfs(类似开源的JuiceFS)，它将远端的pangu(阿里云存储高效的存储系统)，以提供posix接口挂载到拉起的机器上，这样已经挂载的机器可以像访问本地磁盘的方式一样访问远端的pangu存储。结合前面的集群管控系统，将每个id的机器分配并绑定不同存储路径。等hippo拉起机器的时候很根据自己的加载属于自己的数据。

## 4.4 生产-数据Bulk-load方案



Nebula 的存储数据格式



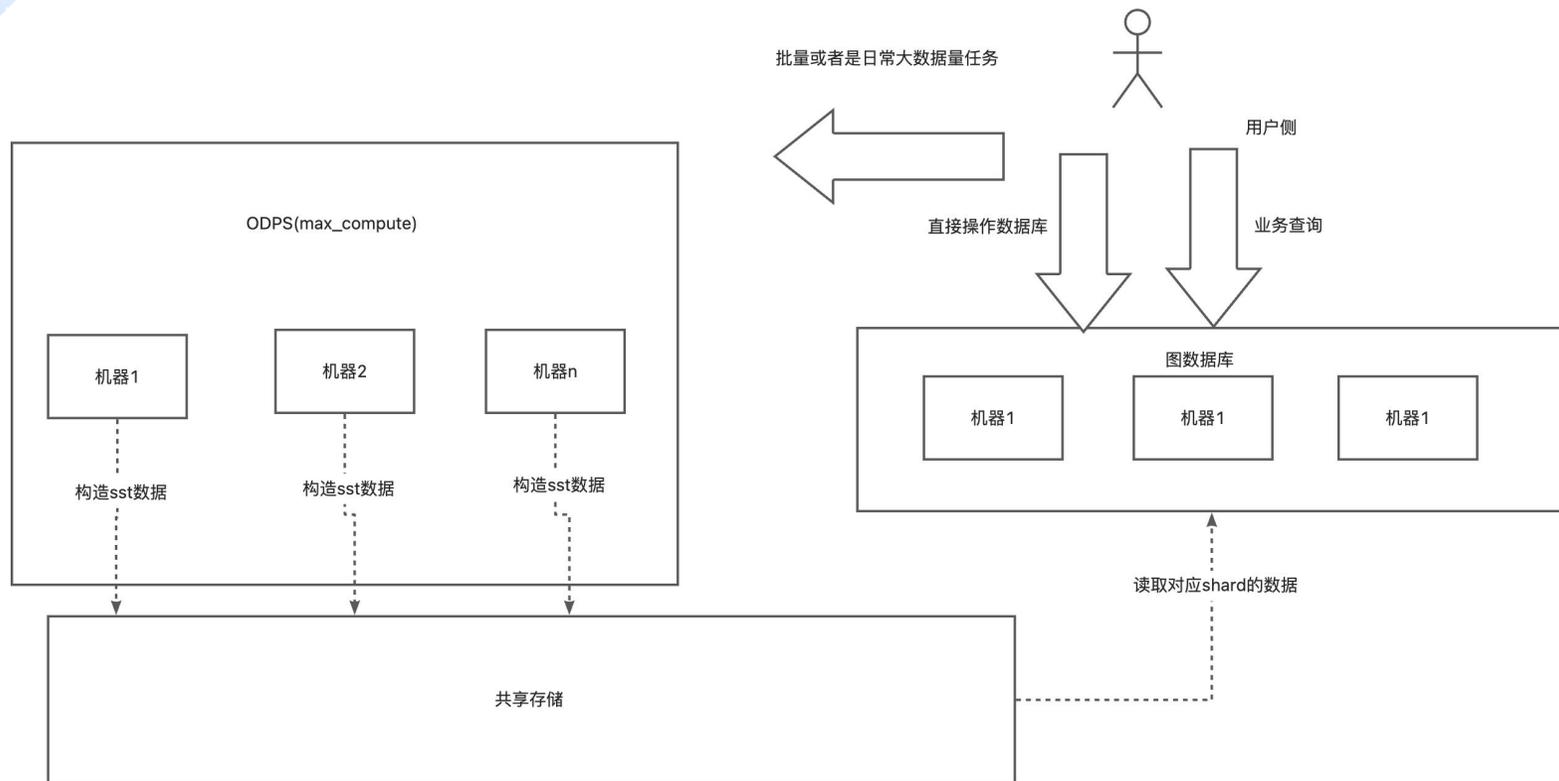
Nebula 的存储数据格式

## 4.4 生产-数据 Bulk-load 方案

底层使用 RocksDB 存储，数据编码格式如上，利用以上信息和对官方 NebulaGraph Exchange 的研究，我们设计实现大批量数据导入方案是：

利用离线 odps 资源构造图数据内部的二进制文件，之后利用图数据库的封装的 RocksDB 的 ingest 函数操作 load 构造的二进制文件，以实现批量大数据导入图数据库。

## 4.4 生产-数据 Bulk-load 方案

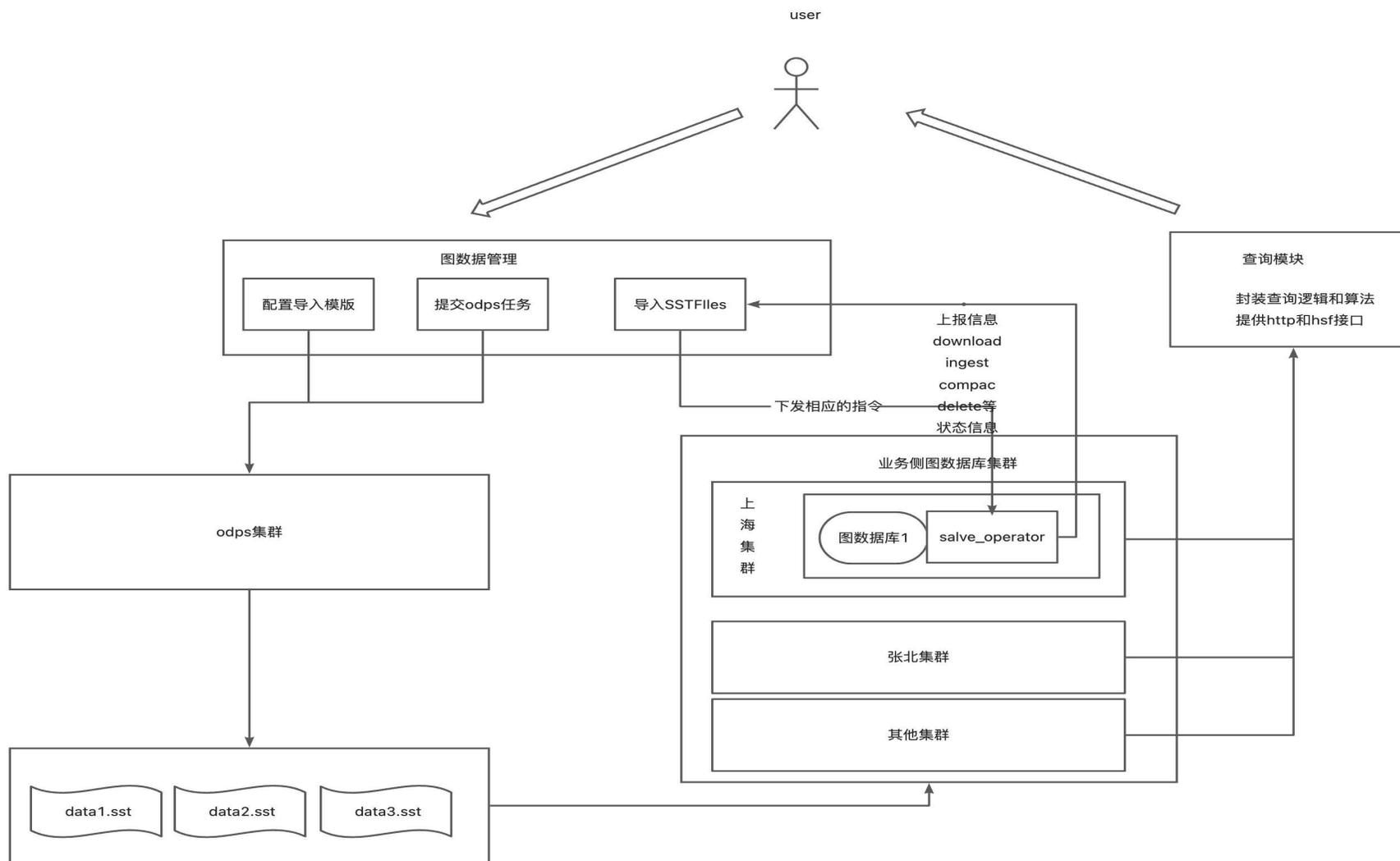


```

select
  NebulaSink(
    item_id,
    sku_id,
    cate_id,
    cate_level1_id,
    cate_name,
    feature
  ) as msg
from
  (
    SELECT
      item_id,
      sku_id,
      cate_id,
      cate_level1_id,
      cate_name,
      feature
    from
      (
        SELECT
          item_id,
          sku_id,
          cate_id,
          cate_level1_id,
          cate_name,
          feature
        FROM
          src_table
        WHERE
          ds='20241224'
      ) AS DISTRIBUTE BY rand()
  );

```

# 4.4 生产-数据 Bulk-load 方案



## 5. 遇到问题和目前的解决办法

1. 数据量巨大，导入时候，底层的 RocksDB 进行 compact，导致导入时间长尾严重，目前我们通过切分数据来减少 compact 的影响
2. bulk-load sst ingest之后，大批量数据导入，数据重建索引耗时和 mem 很大，分割数据 + 规格 + 配置调整

## 6. 后续规划

我们团队有对 RocksDB和其他 kv 数据库的研究和魔改（二次开发）  
在未来的一些成果可以和 NebulaGraph 结合，提升服务能力，最后看是否可以以 PR 的方式贡献社区

# Thanks Q&A

## 与我联系

# songmingtao.smt@taobao.com

-  <https://nebula-graph.com.cn>
-  GitHub: [vesoft-inc/nebula](https://github.com/vesoft-inc/nebula)
-  Twitter: [@NebulaGraph](https://twitter.com/NebulaGraph)
-  Facebook: [@NebulaGraph](https://www.facebook.com/NebulaGraph)
-  <https://discuss.nebula-graph.com.cn>



微信公众号



开源项目

# NebulaGraph 在京东物流一站式 分析平台 UData 的探索与实践

京东物流 李世豪

1、业务背景

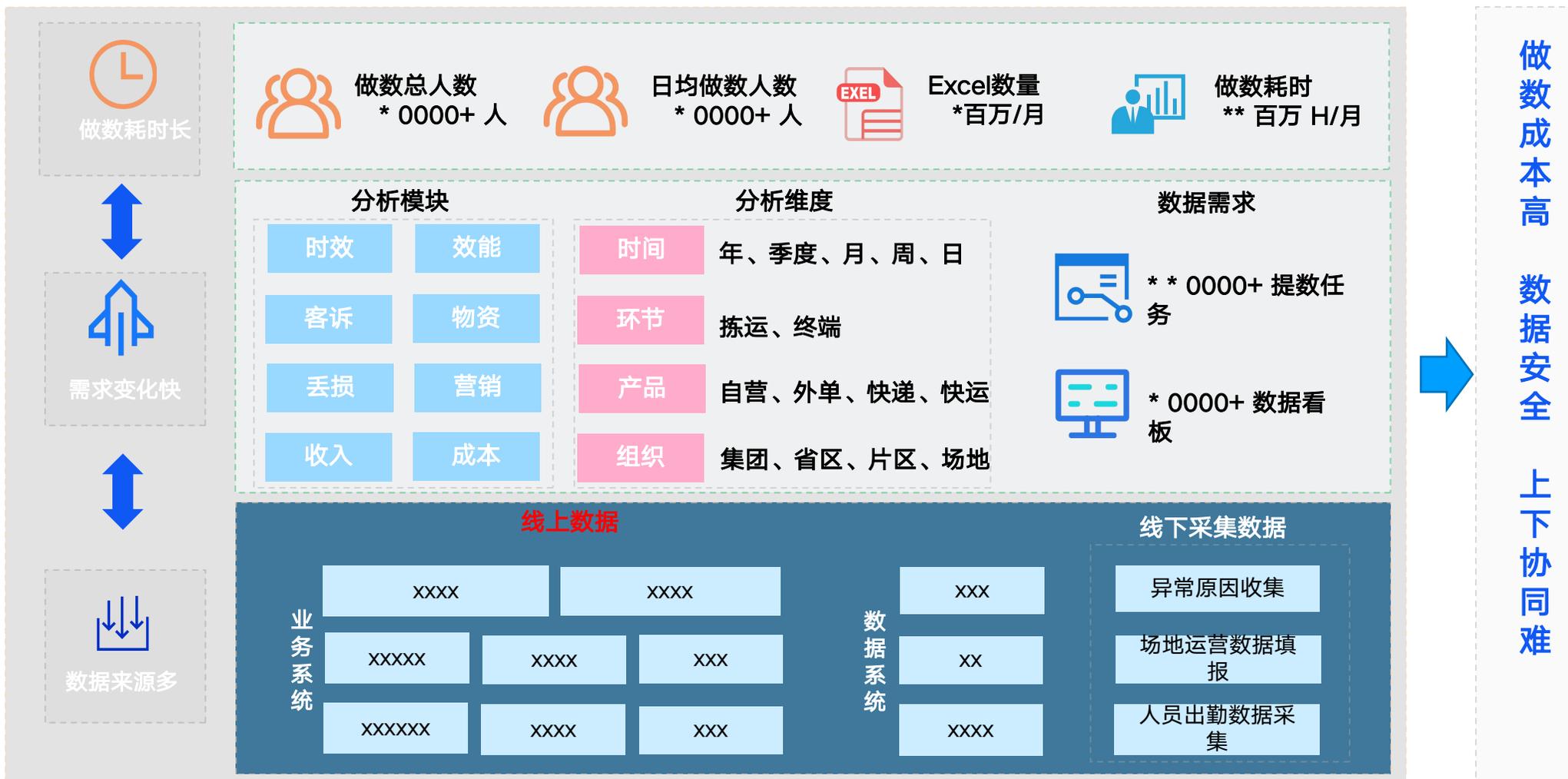
2、技术选型

3、实践案例

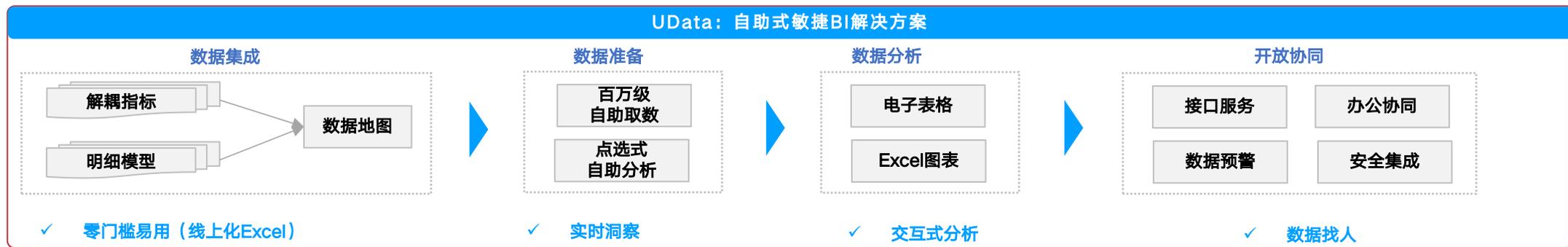
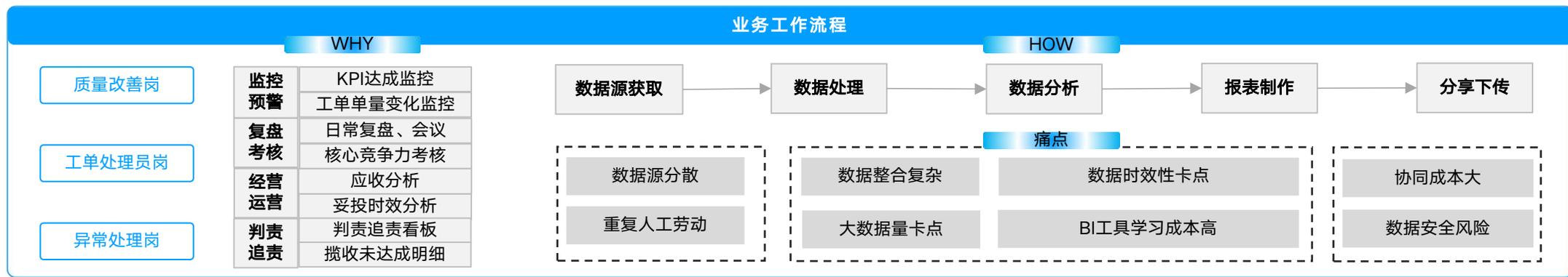
4、未来规划

# 业务背景

# 业务背景



# 京东物流一站式分析平台--UData



# 技术选型

# 为什么会选择使用 Nebula Graph?

数据获取

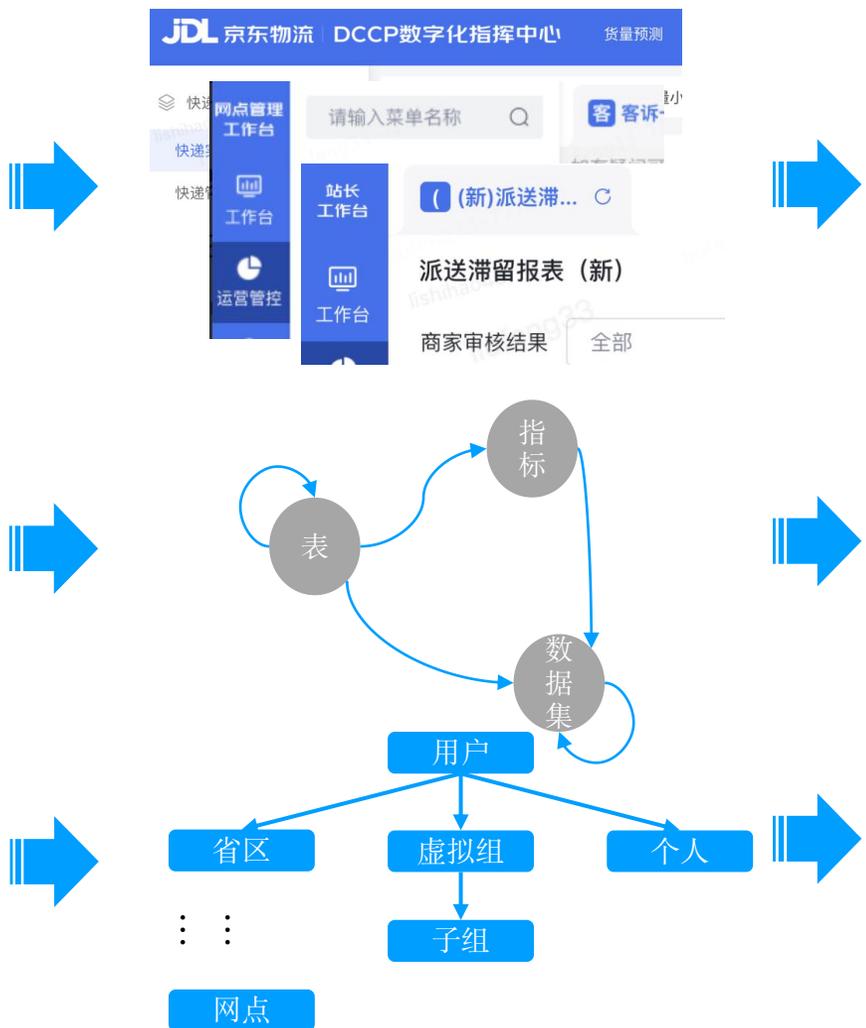
- 大件、中小件...
- 实时、离线数据...
- 仓储、分拣、配送...

数据处理

- 指标
- 视图
- 数据集

数据权限

- 个人
- 小组
- 组织机构



链路长

难溯源

多层嵌套

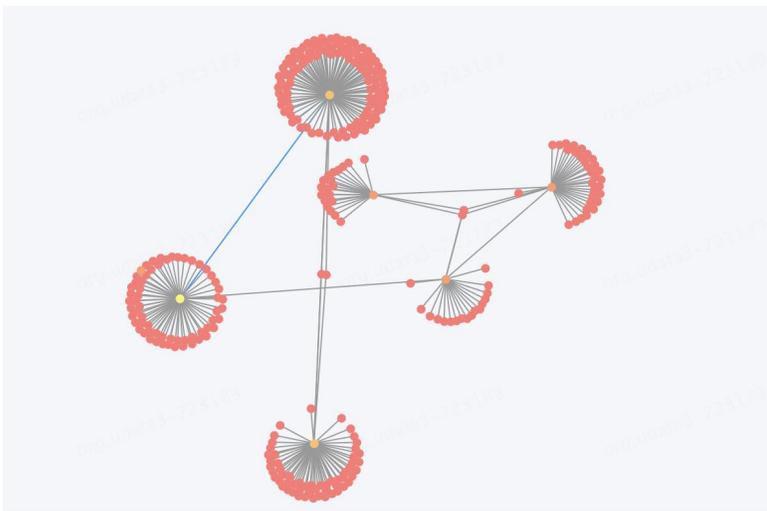
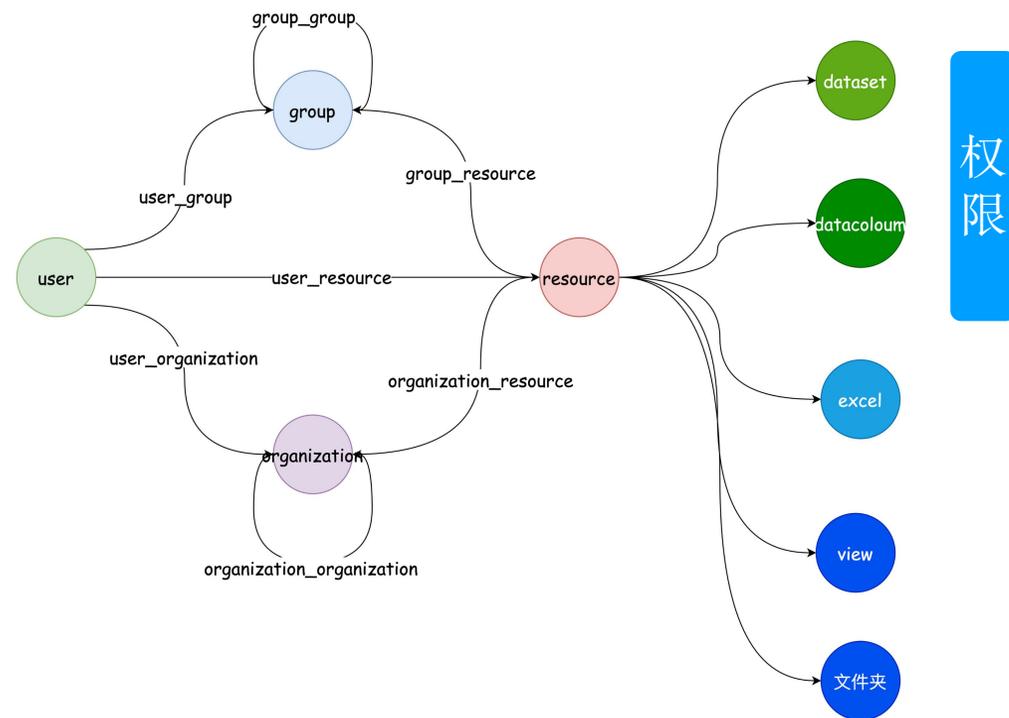
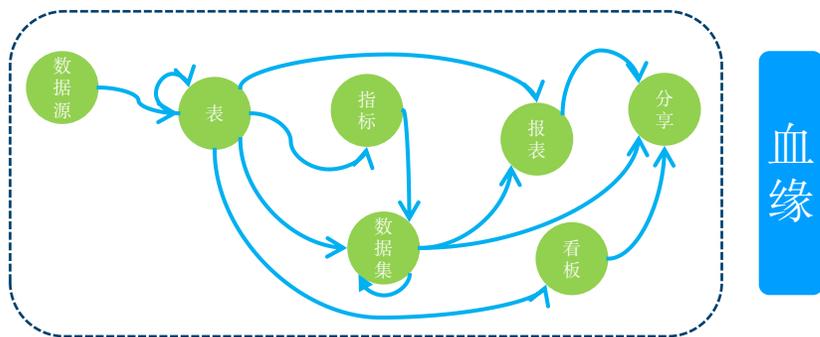
血缘

多层嵌套

扩展困难

Nebula Graph

# 为什么会选择使用 Nebula Graph?



# 为什么会选择使用 Nebula Graph?

**MYSQL**

**nGQL**

```
MATCH (userx:user)-[ug:user_organization]->(currentGroup:organization)-[gg:organization_organization+0..]->(parentGroup:organization)-[gr:organization_resource]->(resource:resource) WHERE id(userx)=='user_495' RETURN userx, resource, ug, currentGroup, gg, parentGroup, gr
```

数据量(条)	MySQL (毫秒)	nGQL (毫秒)
10000	33556	1707
1000	31895	413
100	32526	215
erp(1007)	222/234 (分页100条)	107/145 (不分页)
erp,group,org(1 20 3)	282/255/233/502(分页100条)	415/231 /235/275 (不分页)



# 为什么会选择使用 Nebula Graph?

## 热门回复

2023 年 9 月 · ❤️ 2

查询指定节点和另一种tag的节点之间的关系，但是关系可能是多跳，如何返回中间节点的信息呢

2023 年 10 月 · ❤️ 1

想查询环路径，ngql该如何写呢

2023 年 10 月 · ❤️ 1

已知一个树的父节点和叶子节点的类型，想查询从父节点到叶子节点的所有路径，有更好一点的方案吗？现在的速度是1.2s左右

2023 年 9 月 · ❤️ 1

删除某个节点的某种类型的所有入边怎么写呢

2023 年 9 月 · ❤️ 1

match 限制起始点，不能查询出结果

2023 年 7 月 · ❤️ 1

LISTENER单机部署失败

## 热门话题

2023 年 11 月 · ❤️ 3

这种情况算不算循环呢？

2023 年 9 月 · ❤️ 2

删除某个节点的某种类型的所有入边怎

2023 年 9 月 · ❤️ 2

查询指定节点和另一种tag的节点之间的如何返回中间节点的信息呢

2023 年 7 月 · ❤️ 2

全文搜索可以使用es的match搜索吗？

2023 年 7 月 · ❤️ 2

单机部署3.5.0 cmake时报错

2024 年 3 月 · ❤️ 1

想问一下如果这种情况，想返回边的话

 2 个回复 修改配置文件的ip重启后，没数据了

 修改配置文件的ip重启后，没数据了

 获得了 '周年纪念日'

 MuYi-方扬 想问一下如果这种情况，想返回边的话要怎么写呢？

 steam 想问一下如果这种情况，想返回边的话要怎么写呢？

 想问一下如果这种情况，想返回边的话要怎么写呢？

 steam NebulaGraph is nothing without you | 社区 2023 年度人物合集

 steam 关于感恩节活动小礼品邮寄

 获得了 '捉虫小能手'

 MuYi-方扬 这种情况算不算循环呢？

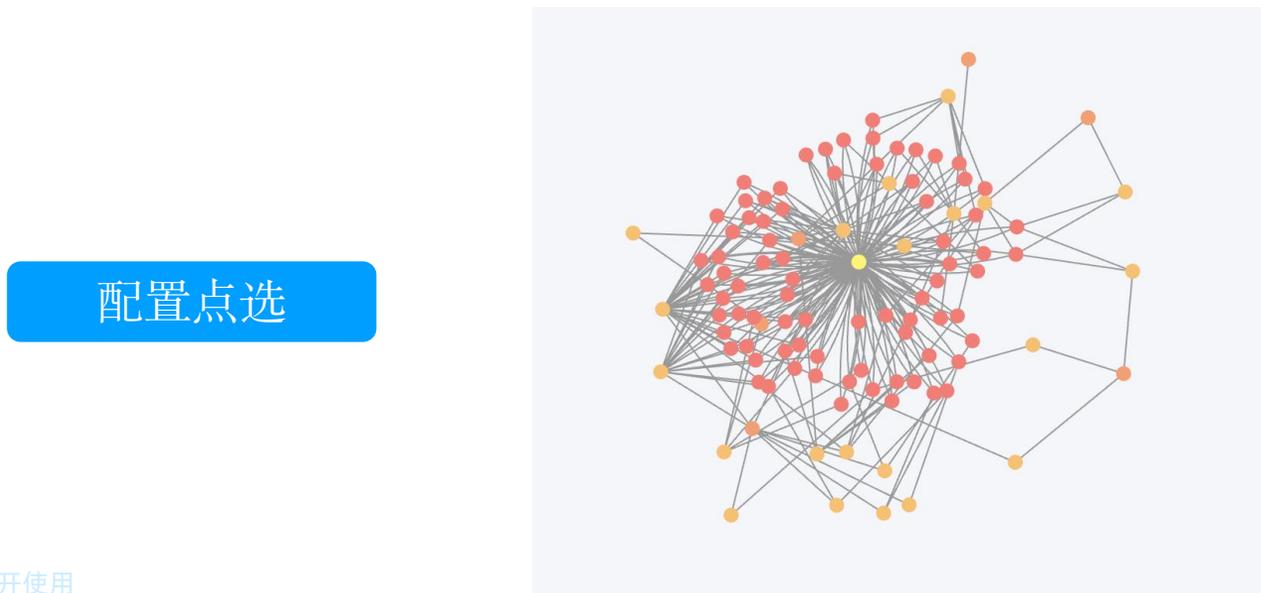
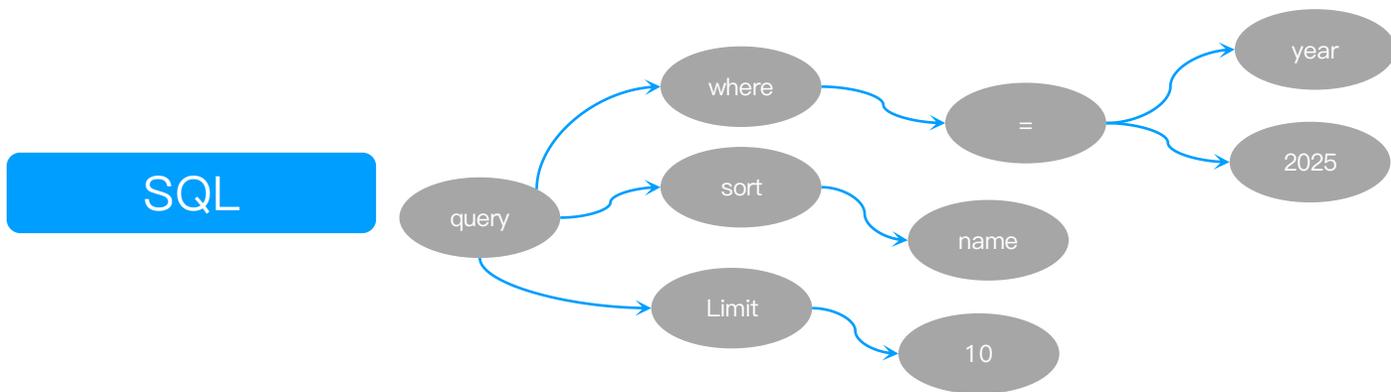
 wey 这种情况算不算循环呢？

 这种情况算不算循环呢？

 获得了 '火鸡先锋'

# 实践案例

# NebulaGraph 在 UData 的实践 --性能优化



冷热表替换



物化视图



缓存策略优化



查询时长下降30%

# NebulaGraph 在 UData 的实践

缓存预测

生成报表特征

生成时序图

图学习预测

提前缓存

命中率提高

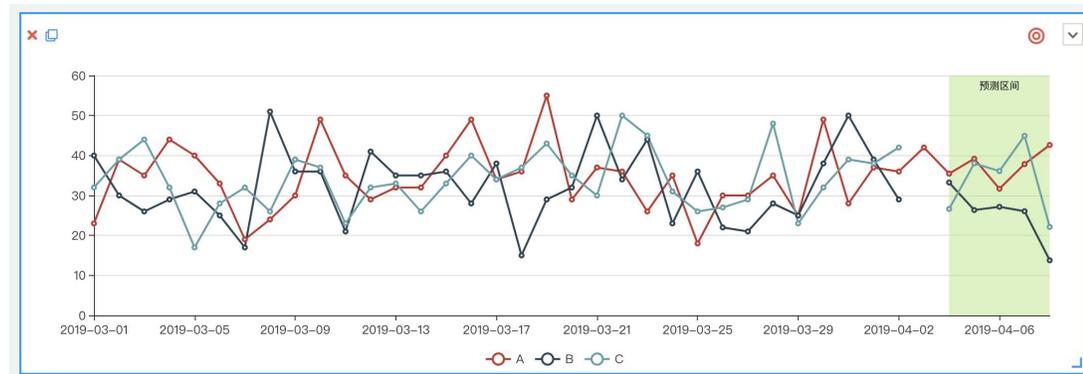
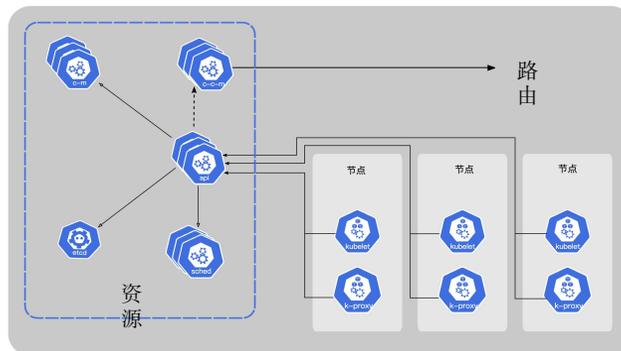
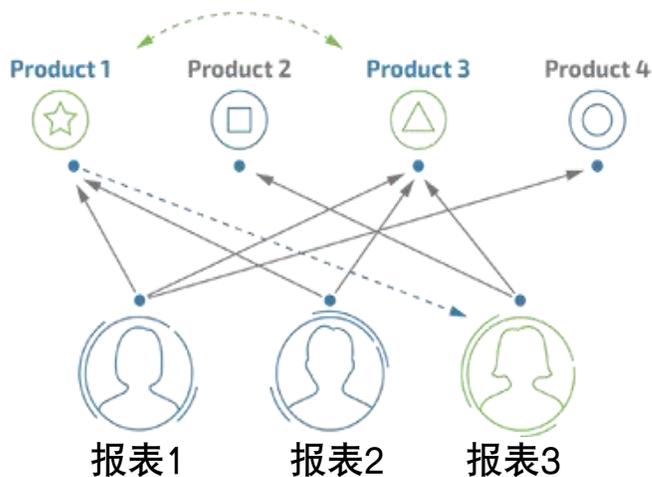
集群路由

寻找相似节点

生成策略

预测集群压力

提前预警



# 未来规划

# 未来规划



# Thanks

-  <https://nebula-graph.com.cn>
-  GitHub: [vesoft-inc/nebula](https://github.com/vesoft-inc/nebula)
-  Twitter: [@NebulaGraph](https://twitter.com/NebulaGraph)
-  Facebook: [@NebulaGraph](https://www.facebook.com/NebulaGraph)
-  <https://discuss.nebula-graph.com.cn>



微信公众号



开源项目