OpenTenBase

# OpenAtom OpenTenBase

## AI 多模态介绍

腾讯云数据库
杨承禹
2025年9月

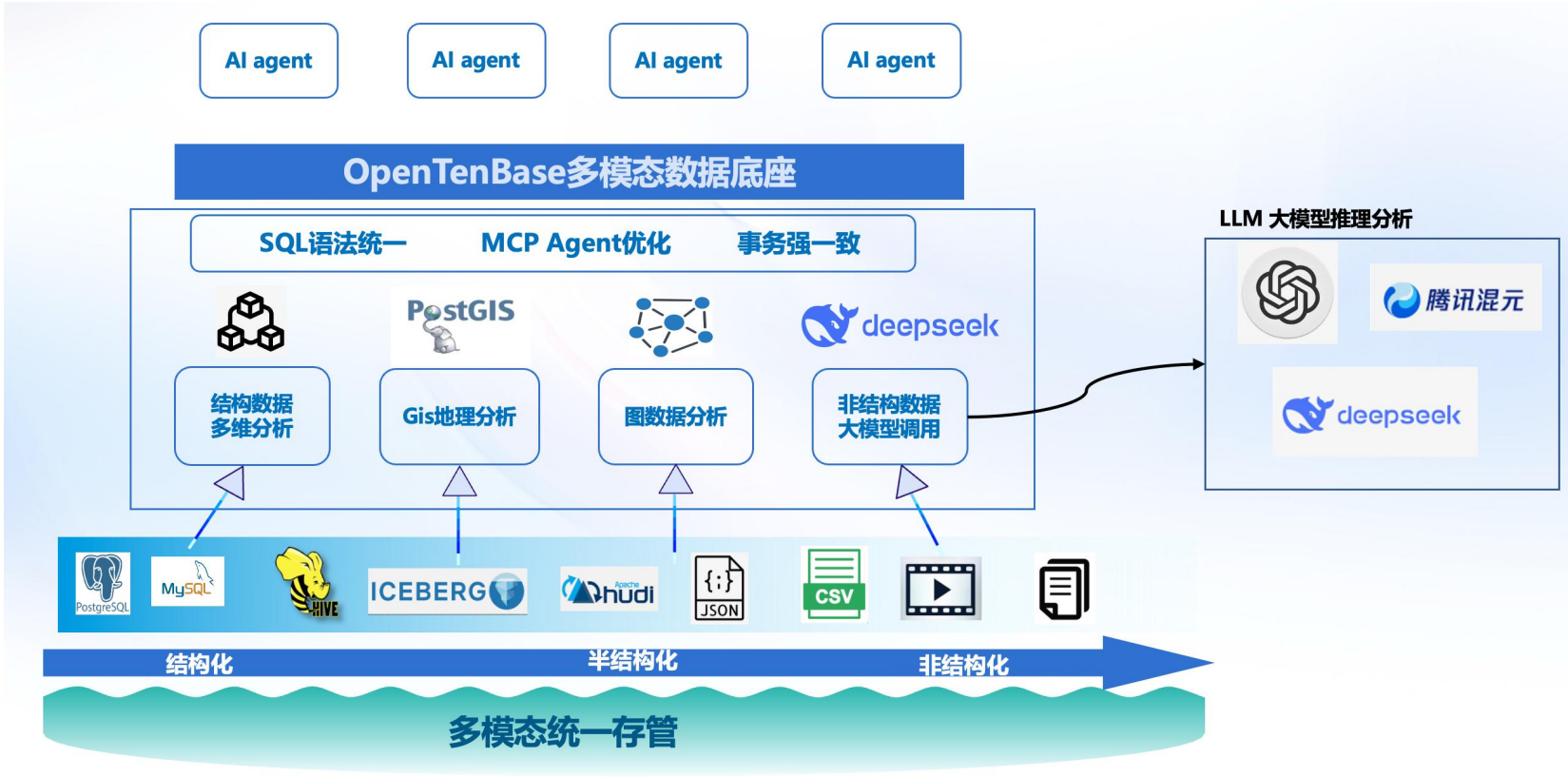
# 目录

01

# OpenTenBase
# 多模态分析

# 赛题

## Data + AI

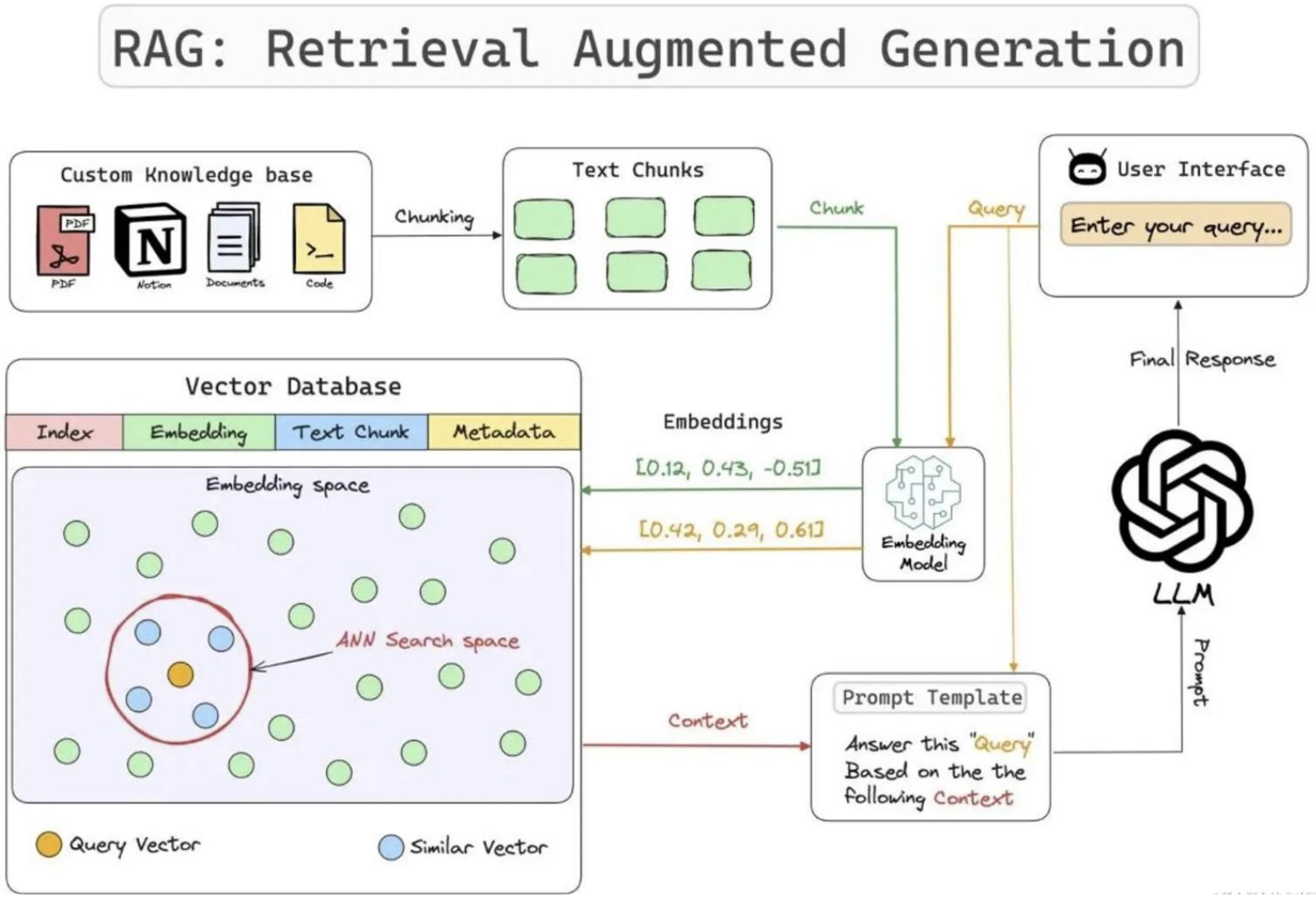



**技术价值**：探索Data+AI数据库重要前沿方向，**打造OpenTenBase开源分布式多模态能力。**

**产业价值**：拓展OpenTenBase结构化数据分析到非结构数据分析，**统一多模态业务价值挖掘**，助力企业数智深度转型。

**人才价值**：培养既懂数据库内核又懂AI的跨领域人才，吸引高校AI人才加入数据库方向，促进人才的交流与合作。

# 多模态分析能力

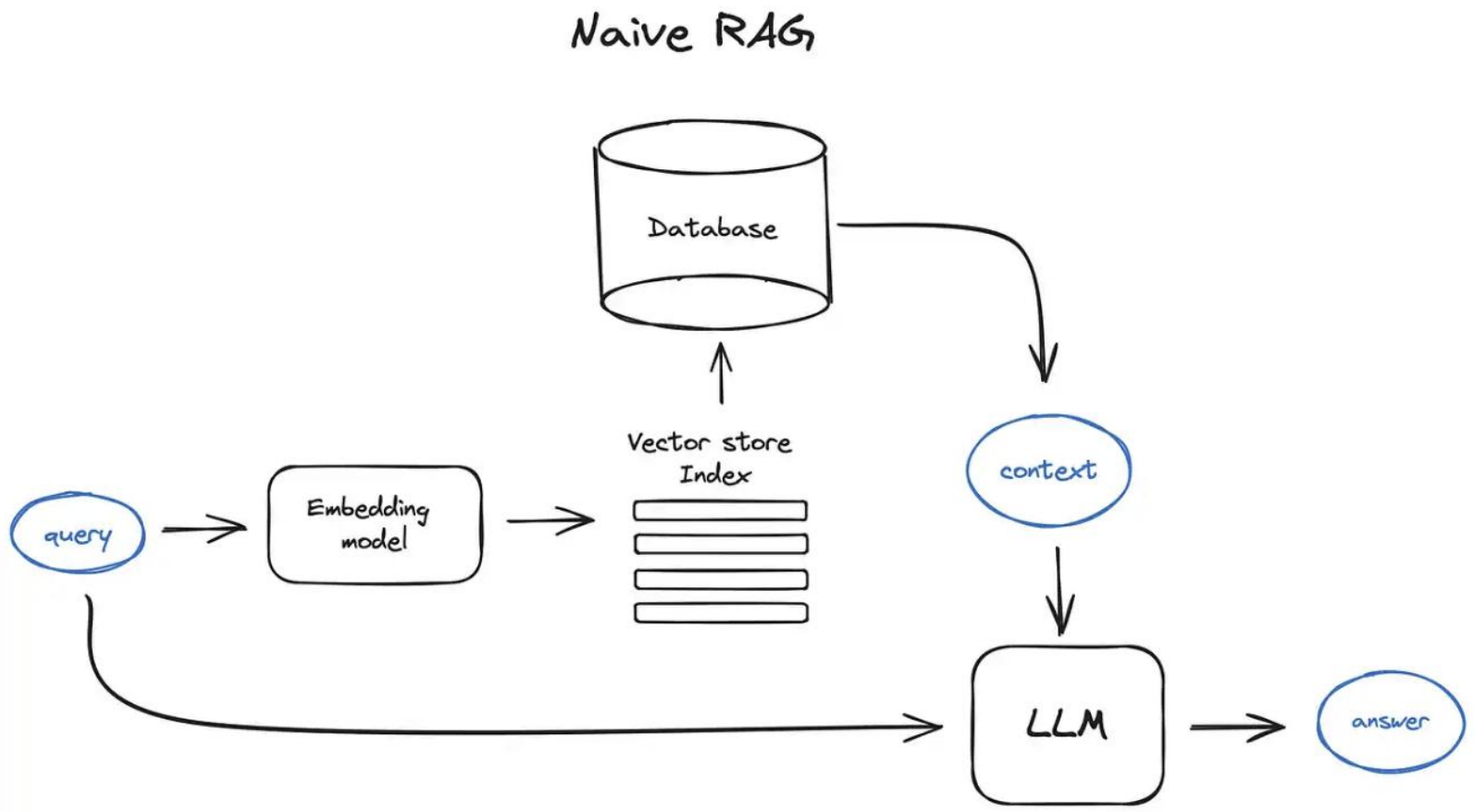


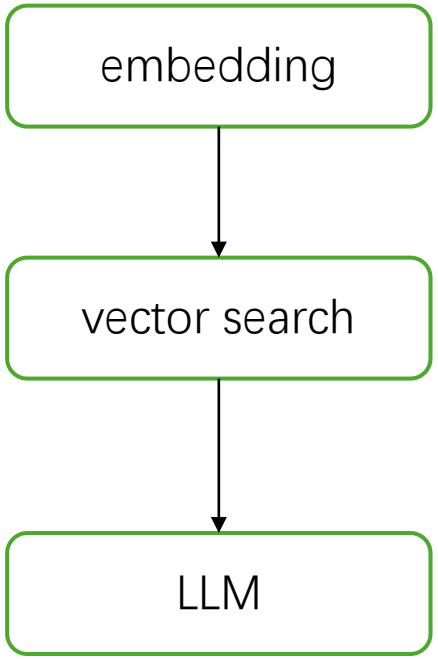

RAG本质上是通过工程化手段，解决LLM知识更新困难的问题。

其核心手段是利用外挂于LLM的知识数据库（通常使用向量数据库）存储未在训练数据集中出现的新数据、领域数据等。

通常而言，RAG将知识问答分成三个阶段：索引、知识检索和基于内容的问答。

# 多模态分析能力


Naive RAG
Database
Vector store
Index
query
Embedding
model
context
LLM
answer


embedding
vector search
LLM

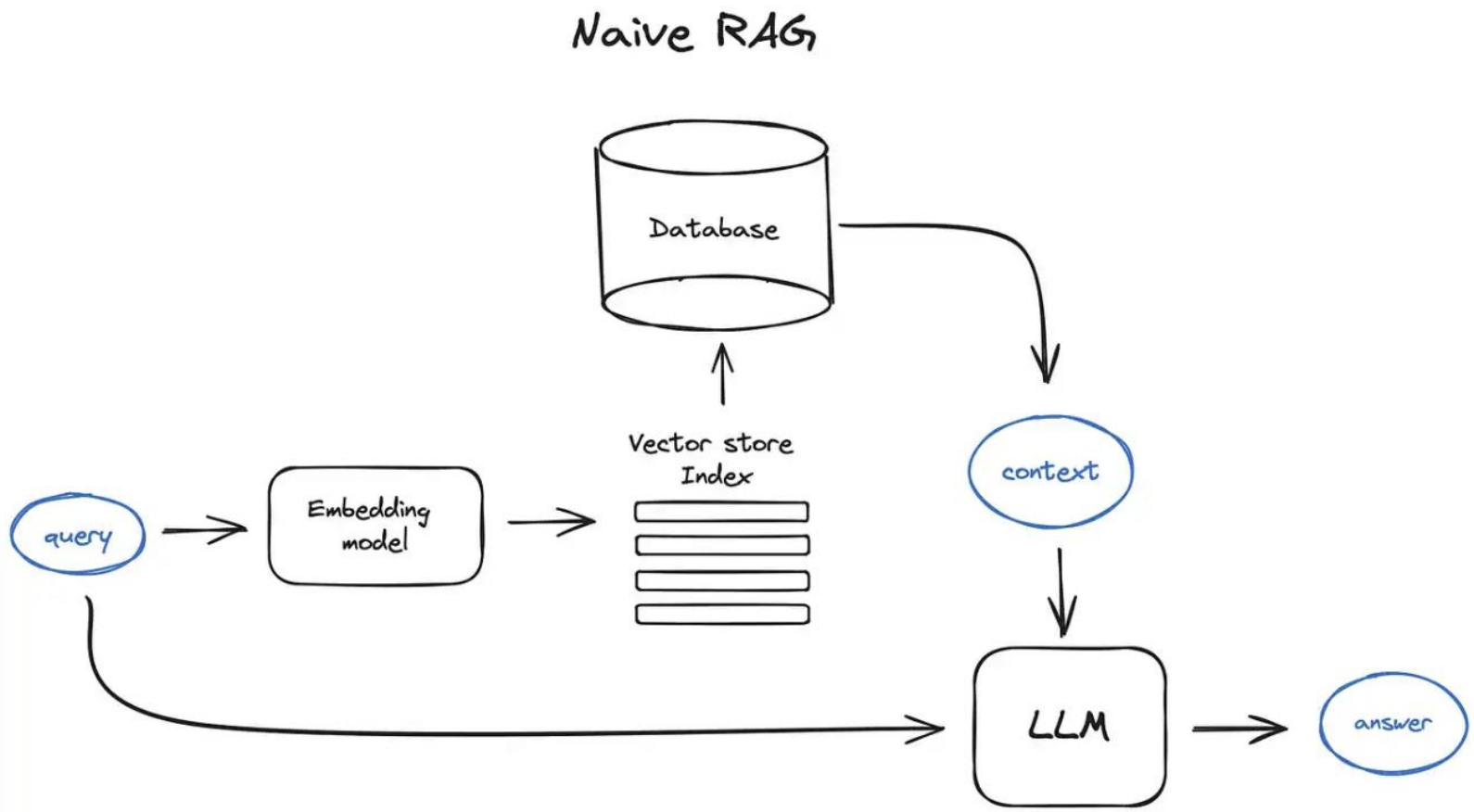
# 多模态分析能力

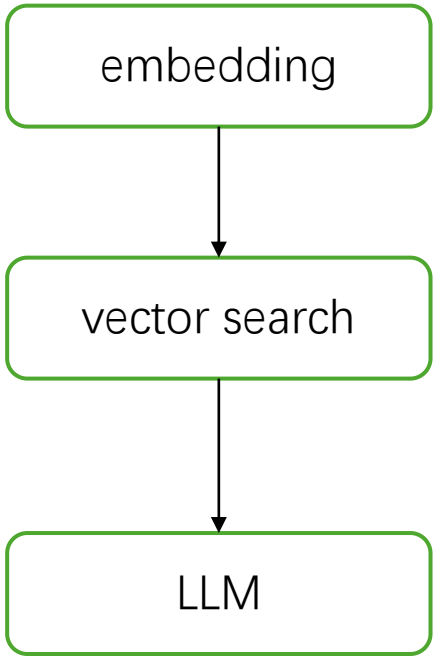
Prompt ?

Chat history ?

Full text search ?

Result cache ···



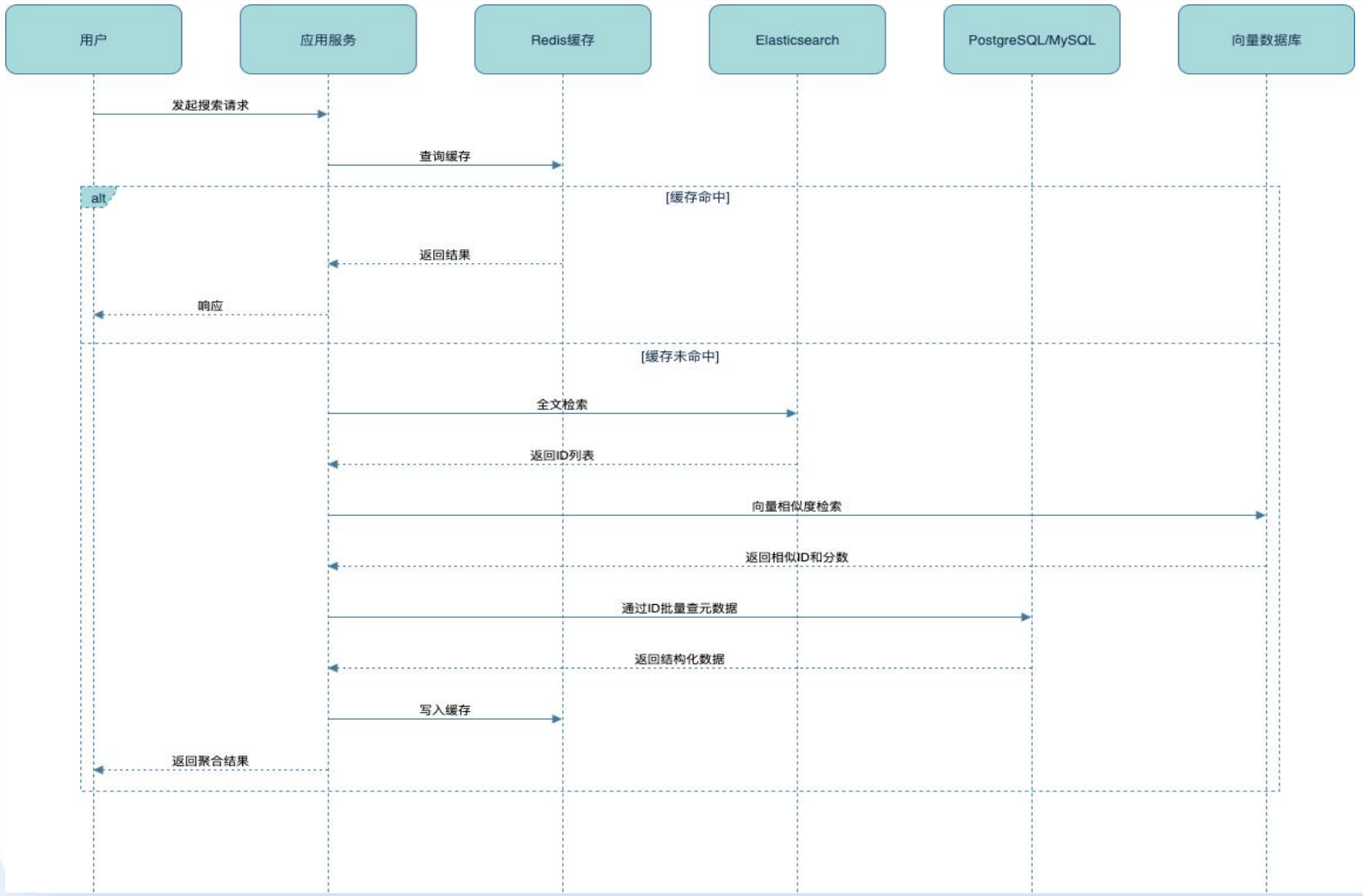
# 业务链路示意

Prompt ?

Chat history ?

Full text search ?

Result cache ……




调用链路长

技术栈广

维护系统多

运维难度高

# OpenTenBase
# 简化调用链路

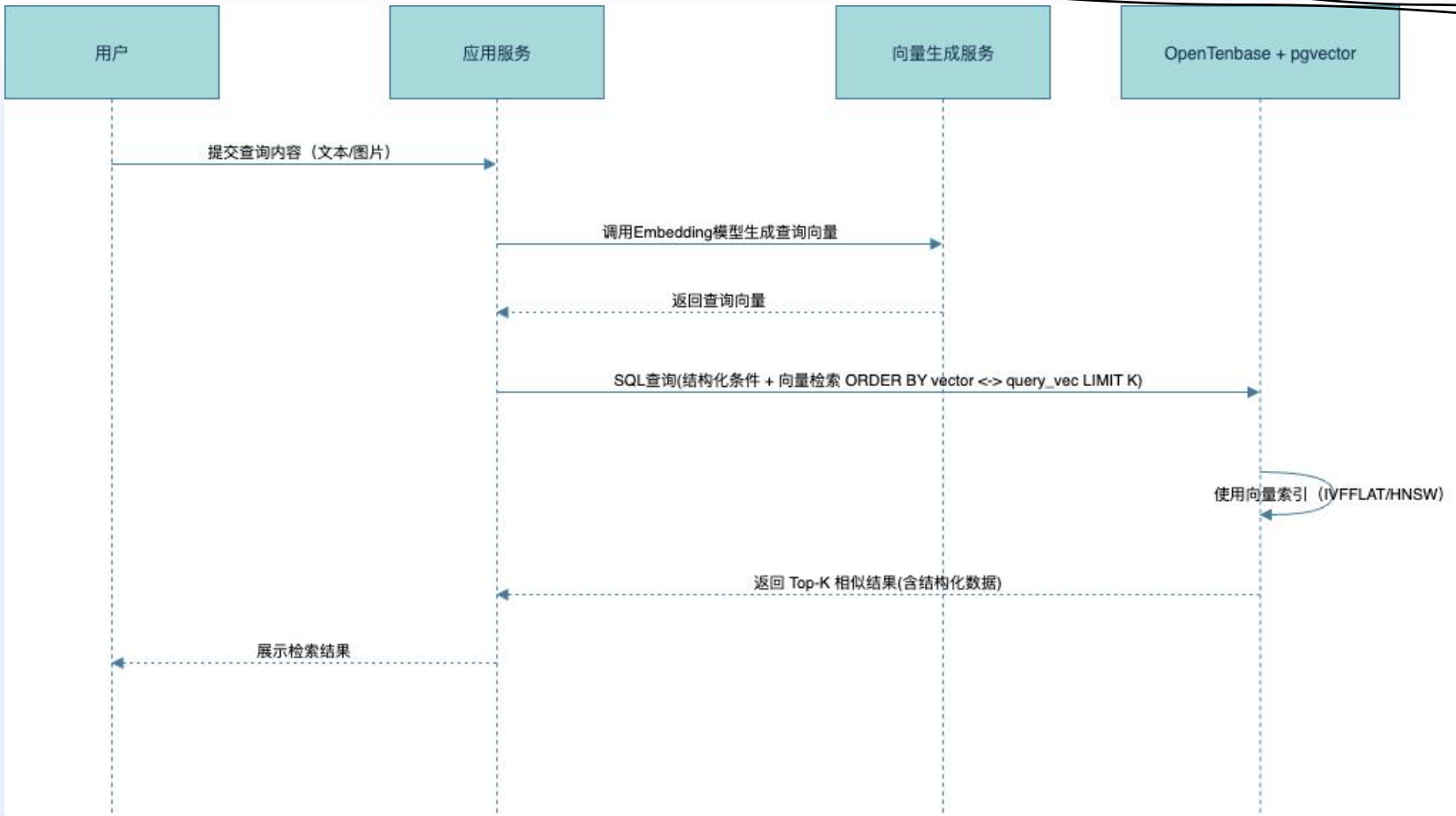
Prompt ?

Chat history ?

Full text search ?

Result cache ……

OpenTenBase




Graph ?

Json?

Gis?

# 多模态分析能力

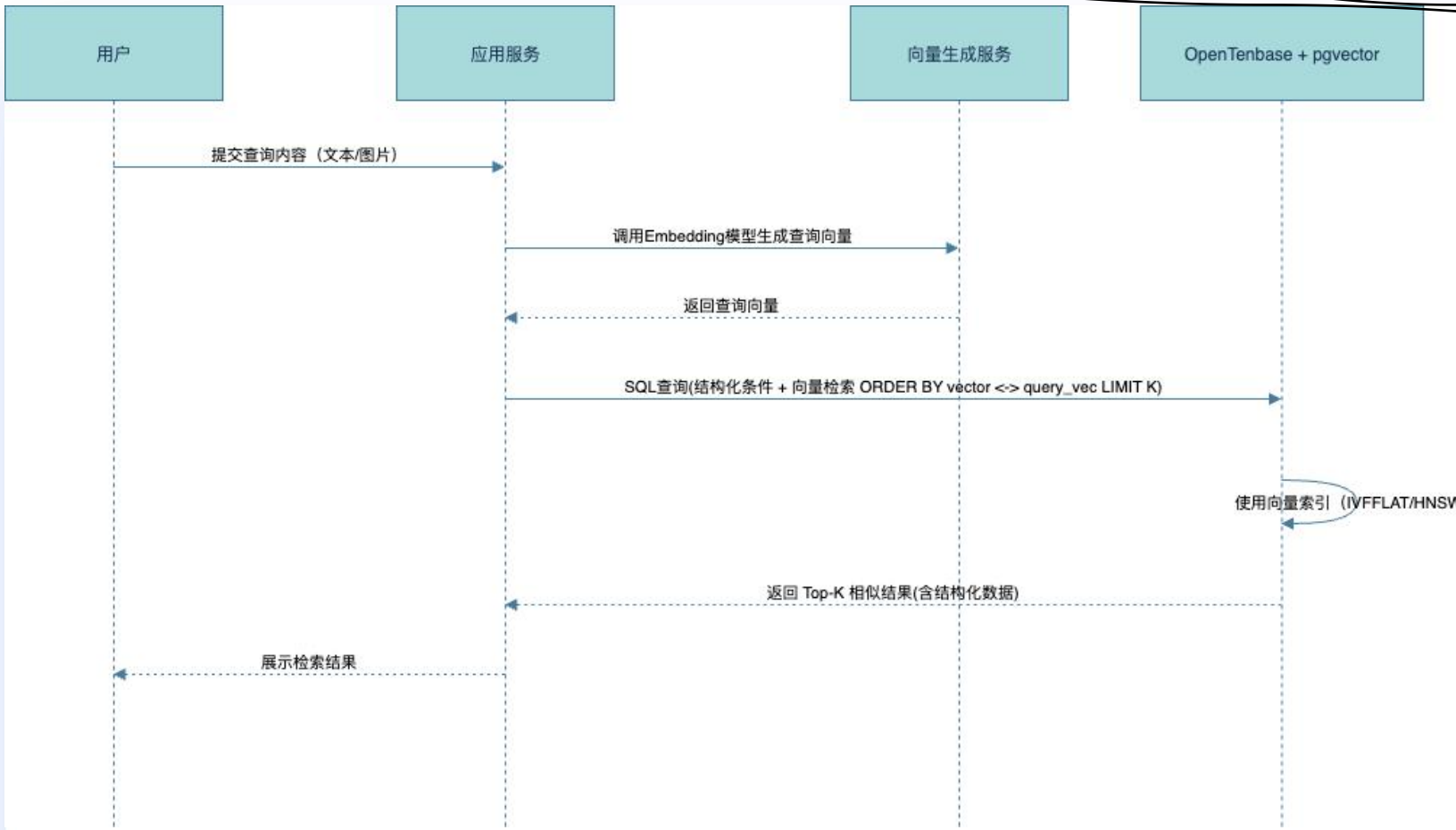
Prompt ?

Chat history ?

Full text search ?

Result cache …

OpenTenBase

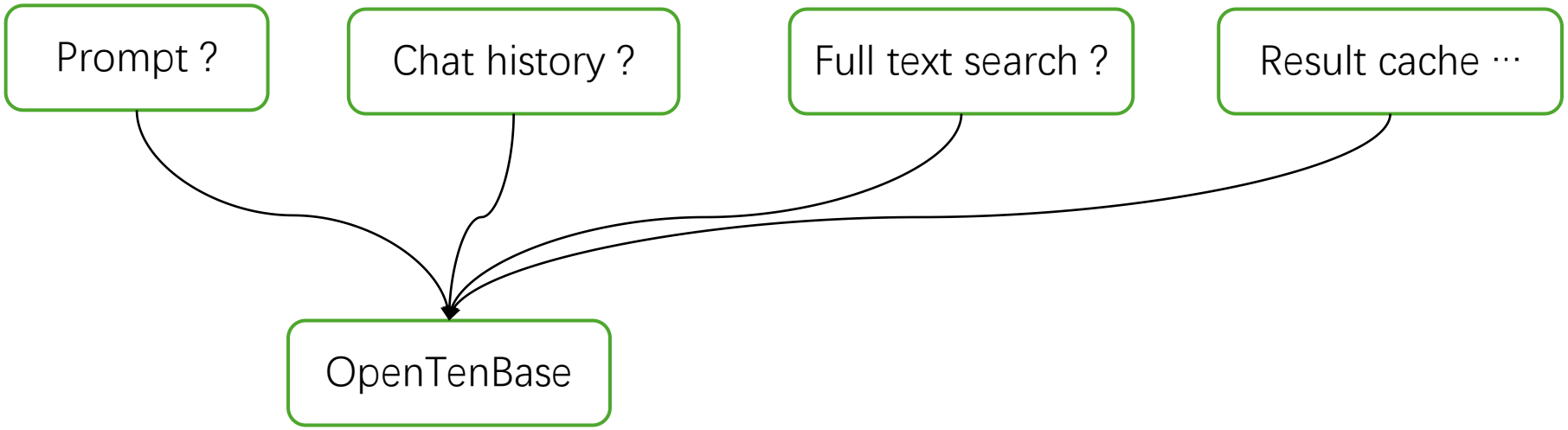



Graph ?

Json?

GIS?

# OpenTenBase 多模态融合



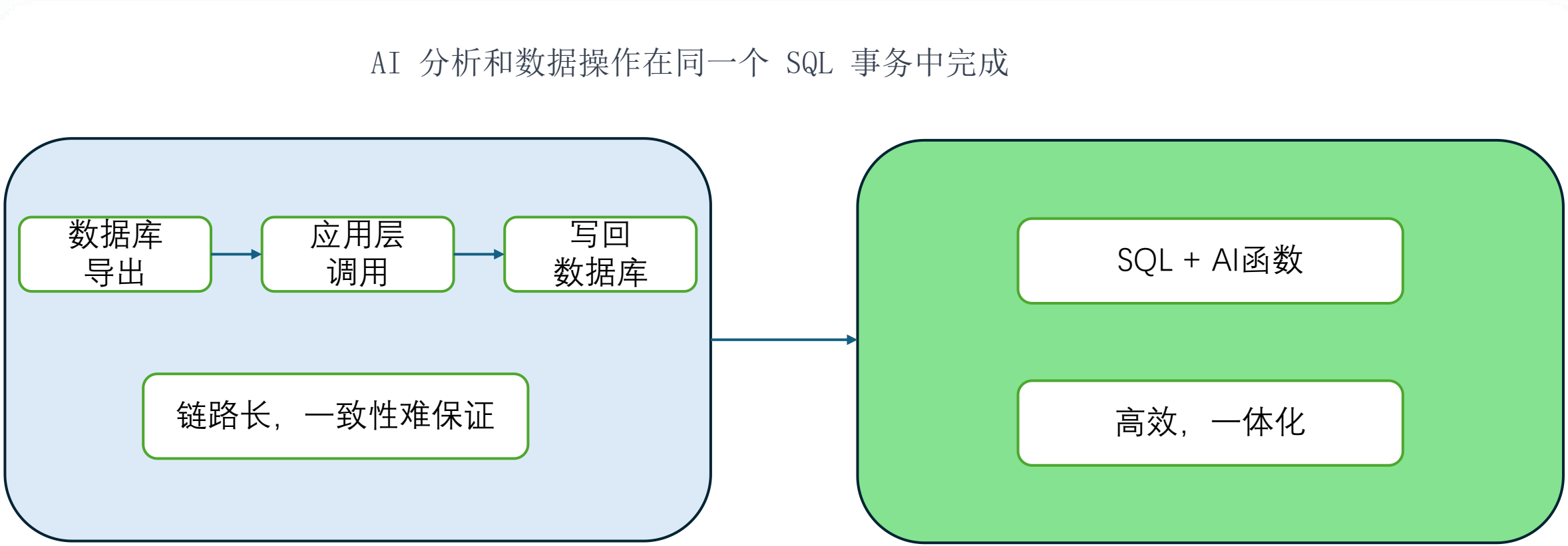

| | | | | |
|---|---|---|---|---|
| Graph | Json | Vector | GIS | Timeseries |
| Age | Json/Jsonb | pgvector | PostGIS | TimescaleDB |

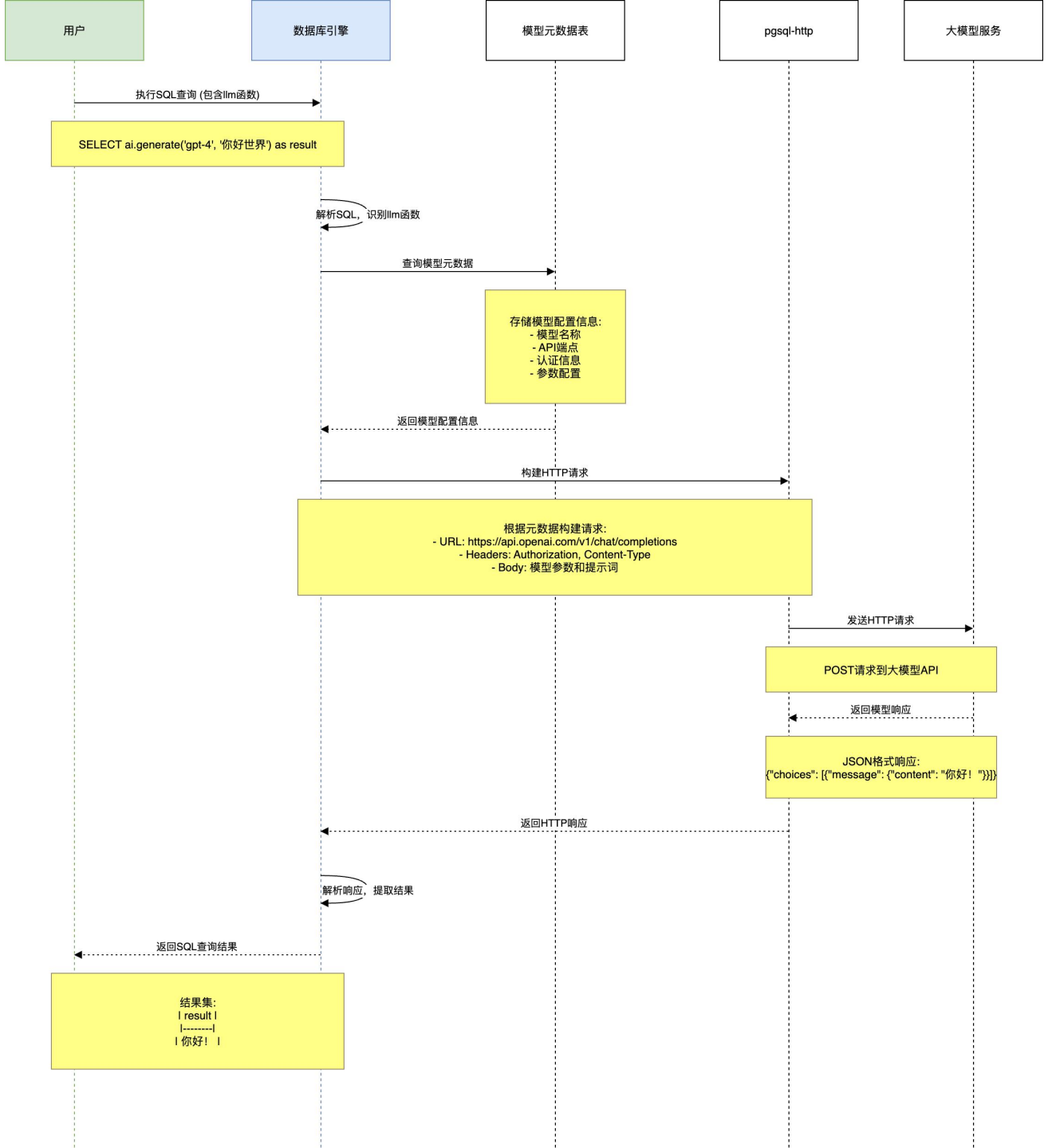
02

# OpenTenBase

# 数据库原生AI

# AI in SQL 原生集成 LLM 函数,将大语言模型的能力引入 SQL引擎






AI函数调用流程

对用户隐藏调用细节

# AI in SQL

**文本生成**

```
SELECT ai.generate_text('为以下产品写一段吸引人的描述：智能手表');
```

**情感分析**

```
SELECT ai.sentiment('这个产品非常好用，我很满意');
```

**文本摘要**

```
SELECT ai.summarize('这里是一段很长的产品说明文本...');
```

**指定返回类型生成**

```
SELECT ai.generate('9*9 = ?',NULL::integer);
```

简易使用

SQL调用 LLM

# AI in SQL

无需手动类型转换，可与 pgvector 无缝配合衔接使用

## 嵌入向量生成

```
-- 基本用法
SELECT ai.embedding('这是一段需要生成嵌入向量的文本');

-- 指定模型
SELECT ai.embedding('这是一段需要生成嵌入向量的文本', 'text-embedding-ada-002');
```

简易使用

SQL调用 LLM

# AI in SQL

## 图像分析

```sql
-- 使用图像 URL
SELECT ai.image('这张图片中有什么？', 'https://example.com/image.jpg');

-- 使用二进制图像数据
SELECT ai.image('这张图片中有什么？', image_data_column)
FROM images_table
WHERE id = 1;

-- 指定模型和配置
SELECT ai.image(
    '这张图片中有什么？',
    'https://example.com/image.jpg',
    'gpt-4-vision',
    '{"max_tokens": 500}'::jsonb
);
```

图生文

支持指定 url 或二进制数据

# AI in SQL

| 功能 | 函数 | 返回类型 |
| --- | --- | --- |
| 多态生成 | `ai.generate(prompt, dummy, model_name, config)` | anyelement |
| 文本生成 | `ai.generate_text(prompt, model_name, config)` | TEXT |
| 整数生成 | `ai.generate_int(prompt, model_name, config)` | INTEGER |
| 浮点数生成 | `ai.generate_double(prompt, model_name, config)` | DOUBLE PRECISION |
| 布尔值生成 | `ai.generate_bool(prompt, model_name, config)` | BOOLEAN |
| 文本摘要 | `ai.summarize(text_content, model_name, config)` | TEXT |
| 文本翻译 | `ai.translate(text_content, target_language, model_name, config)` | TEXT |
| 情感分析 | `ai.sentiment(text_content, model_name, config)` | TEXT |
| 问答提取 | `ai.extract_answer(question, context, model_name, config)` | TEXT |
| 嵌入向量生成 | `ai.embedding(input, model_name, config)` | TEXT |
| 图像分析 | `ai.image(prompt, image_url, model_name, config)` | TEXT |
| 图像分析 | `ai.image(prompt, image_bytea, model_name, config)` | TEXT |

功能丰富

类型安全

# AI in SQL 多模态分析

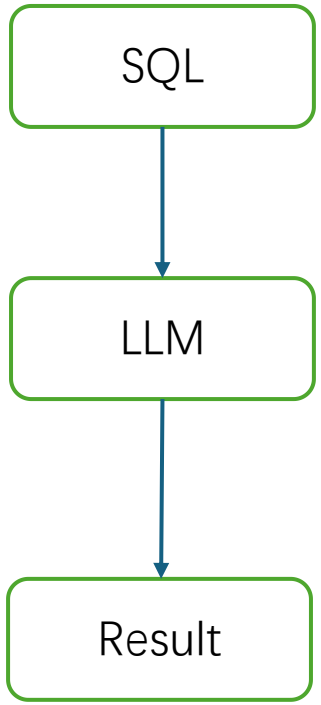
```sql
-- （例）多模态产品分析：结合产品属性、图片分析和客户评论
SELECT
    p.product_id,
    p.product_name,
    p.category,
    p.price,
    -- 图像分析：提取产品视觉特征
    ai.image('描述这个产品的主要外观特征', p.product_image) AS visual_features,

    -- 文本分析：评论情感
    ai.sentiment(r.review_text) AS review_sentiment,

    -- AI函数作为聚合函数参数：计算产品评论的平均情感分数
    avg(ai.generate_double('将此评论情感转换为-1到1之间的分数： ' || r.review_text)) AS avg_sentiment_score,

    -- AI函数作为聚合函数参数：提取评论中的关键主题
    array_agg(DISTINCT ai.extract_topics(r.review_text)) AS common_topics,

    -- AI函数作为聚合函数参数：生成产品评论摘要
    ai.summarize(string_agg(r.review_text, ' ')) AS product_review_summary
FROM
    products p
JOIN
    product_reviews r ON p.product_id = r.product_id
WHERE
    p.category = 'Electronics'
    -- AI函数作为过滤条件：只选择与产品描述相关的评论
    AND ai.generate_double('评估此评论与产品描述的关联度(0-1)：评论： ' || r.review_text || '，产品描述： ' || p.description) > 0.7
GROUP BY
    p.product_id, p.product_name, p.category, p.price, p.product_image
HAVING
    -- 使用简单的计数条件
    count(*) >= 3
ORDER BY
    avg_sentiment_score DESC
LIMIT 5;
```



# AI in SQL

```sql
-- 配置文本模型
SELECT ai.add_completion_model(
    model_name => 'gpt-4',
    uri => 'https://api.openai.com/v1/chat/completions',
    default_args => '{"model": "gpt-4", "temperature": 0.7}'::jsonb,
    token => 'your_api_key',
    model_provider => 'openai'
);

-- 配置嵌入向量模型
SELECT ai.add_embedding_model(
    model_name => 'text-embedding-ada-002',
    uri => 'https://api.openai.com/v1/embeddings',
    default_args => '{"model": "text-embedding-ada-002"}'::jsonb,
    token => 'your_api_key',
    model_provider => 'openai'
);
```

配置便捷

OpenAI 兼容接口

支持不同大模型服务商

# AI in SQL

```
SET ai.completion_model = 'hunyuan-chat';
SET ai.embedding_model = 'text-embedding-ada-002';
SET ai.image_model = 'gpt-4-vision';

-- 在查询中使用默认模型
SELECT ai.translate('hello world!', 'chinese');

-- 使用自定义参数
SELECT ai.translate(
    'hello world',
    'chinese'
    'deepseek-v3',
    '{"temperature": 0.2, "max_tokens": 500}'
);
```

动态模型选择

灵活参数配置

# 谢谢大家