

如何使用RAG搭建智能问答助手

--以PolarDB社区问答机器人为例

兰韵诗、李翔
2024年6月

华东师范大学

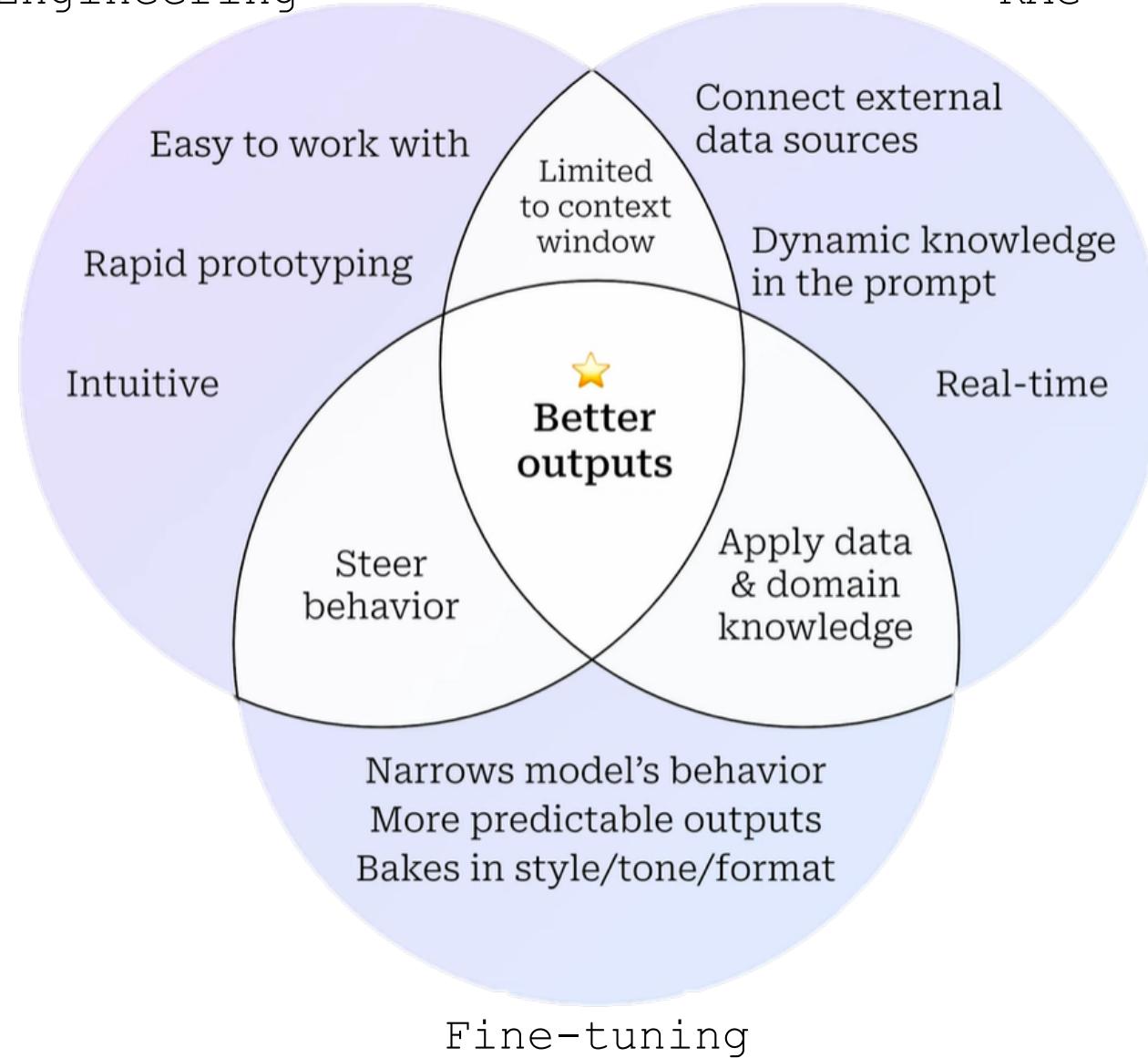
- 检索增强生成（RAG）理论介绍
 - 基础介绍
 - 优化方案
- PolarDB社区问答机器人项目实践
 - 项目背景
 - 实施方案

- 检索增强生成（RAG）理论介绍
 - 基础介绍
 - 优化方案
- PolarDB社区问答机器人项目实践
 - 项目背景
 - 实施方案

Prompt Engineering, RAG, and Fine-tuning

Prompt Engineering

RAG



RAG vs Finetuning

Table 1: Comparison between RAG and Fine-Tuning

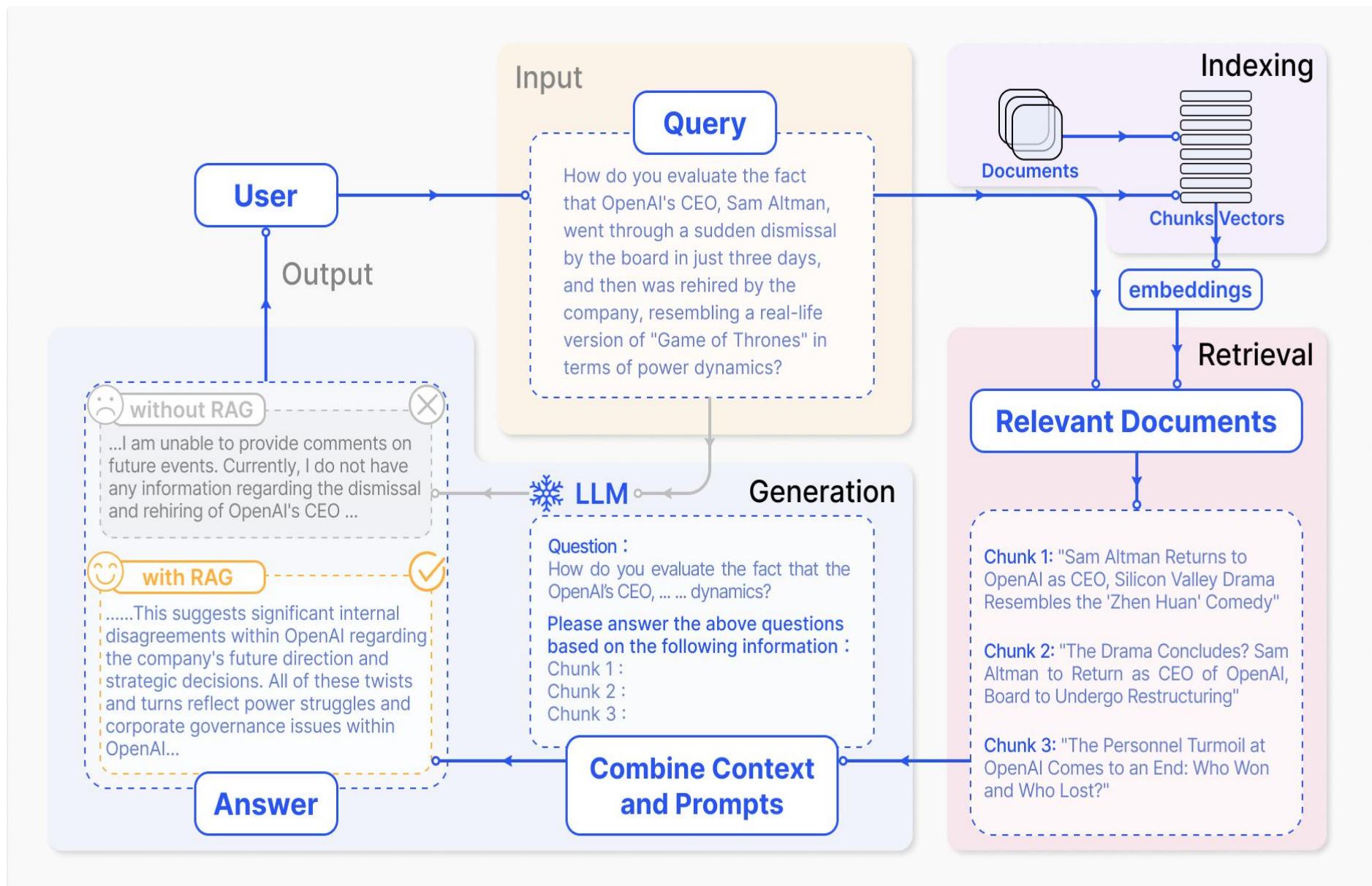
Feature Comparison	RAG	Fine-Tuning
Knowledge Updates	<p>Directly updating the retrieval knowledge base ensures that the information remains current without the need for frequent retraining, making it well-suited for dynamic data environments.</p>	<p>Stores static data, requiring retraining for knowledge and data updates.</p>
External Knowledge	<p>Proficient in leveraging external resources, particularly suitable for accessing documents or other structured/unstructured databases.</p>	<p>Can be utilized to align the externally acquired knowledge from pretraining with large language models, but may be less practical for frequently changing data sources.</p>
Data Processing	<p>Involves minimal data processing and handling.</p>	<p>Depends on the creation of high-quality datasets, and limited datasets may not result in significant performance improvements.</p>
Model Customization	<p>Focuses on information retrieval and integrating external knowledge but may not fully customize model behavior or writing style.</p>	<p>Allows adjustments of LLM behavior, writing style, or specific domain knowledge based on specific tones or terms.</p>
Interpretability	<p>Responses can be traced back to specific data sources, providing higher interpretability and traceability.</p>	<p>Similar to a black box, it is not always clear why the model reacts a certain way, resulting in relatively lower interpretability.</p>

RAG vs Finetuning

Computational Resources	Depends on computational resources to support retrieval strategies and technologies related to databases. Additionally, it requires the maintenance of external data source integration and updates.	The preparation and curation of high-quality training datasets, defining fine-tuning objectives, and providing corresponding computational resources are necessary.
Latency Requirements	Involves data retrieval, which may lead to higher latency.	LLM after fine-tuning can respond without retrieval, resulting in lower latency.
Reducing Hallucinations	Inherently less prone to hallucinations as each answer is grounded in retrieved evidence.	Can help reduce hallucinations by training the model based on specific domain data but may still exhibit hallucinations when faced with unfamiliar input.
Ethical and Privacy Issues	Ethical and privacy concerns arise from the storage and retrieval of text from external databases.	Ethical and privacy concerns may arise due to sensitive content in the training data.

- RAG 就像给模型一本教科书，用于定制的信息检索，所以要求模型本身要有基于知识解决问题的能力。
- FT 就像一个学生随着时间的推移内化知识，更适合模仿特定的结构、风格或格式。FT 可以通过增强基础模型知识、调整输出和教授复杂指令来提高模型的性能和效率。然而，它不那么擅长整合新知识或快速迭代新的用例。

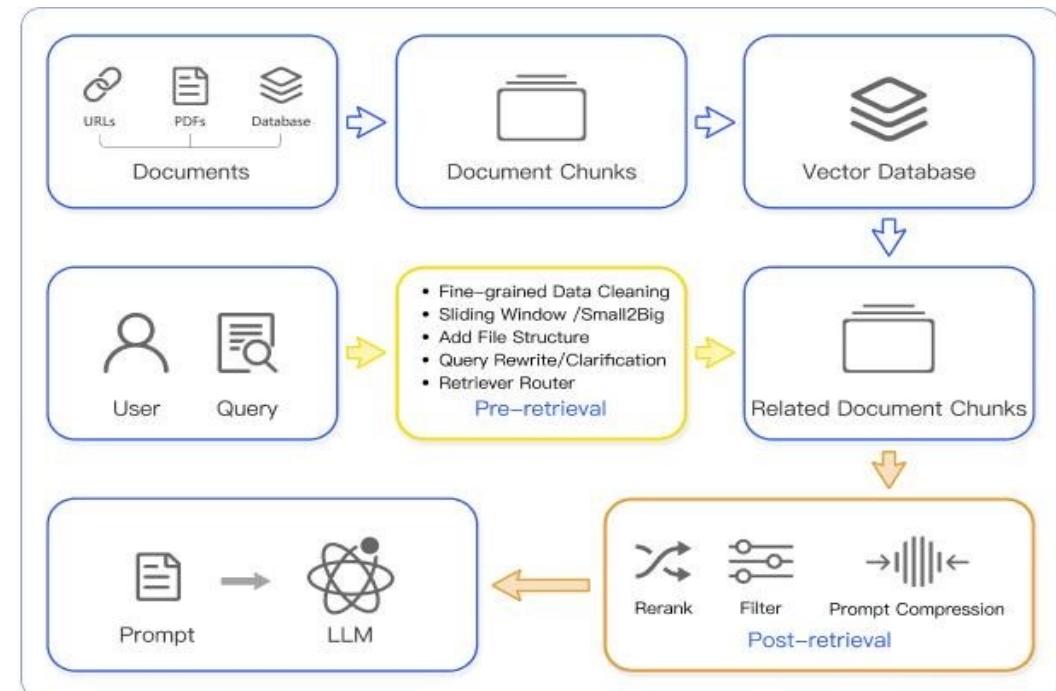
A typical case of RAG



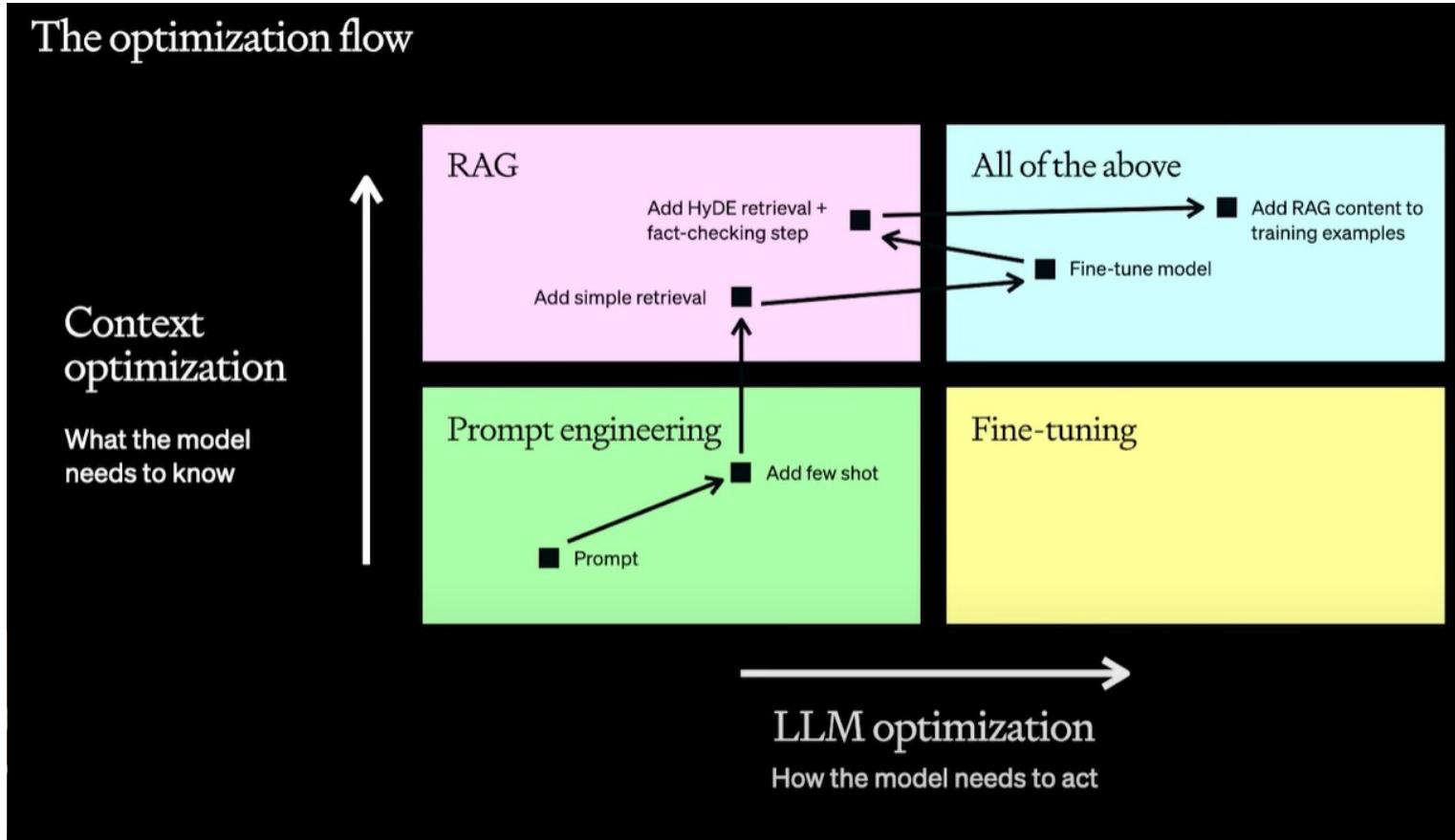
RAG pipeline

Index Optimization → Pre-Retrieval Process → Retrieval →
Post-Retrieval Process → Generation

- **Optimizing Data Indexing:** sliding window, fine-grained segmentation、adding metadata
- **Pre-Retrieval Process:** retrieve routes, summaries, rewriting, and confidence judgment
- **Post-Retrieval Process:** reorder, filter content retrieval



The optimization flow



首先从Prompt Engineering开始，通过添加一些shot来验证模型具有解决问题的能力
验证模型能力之后，开始加入简单的检索逻辑
之后在检索逻辑上进行优化，针对特定的任务优化、微调模型

- 检索增强生成（RAG）理论介绍
 - 基础介绍
 - 优化方案
- PolarDB社区问答机器人项目实践
 - 项目背景
 - 实施方案

The three key questions of RAG

What to retrieve ?

- Token
- Phrase
- Chunk
- Paragraph
- Entity
- Knowledge graph

When to retrieve ?

- Single search
- Each token
- Every N tokens
- Adaptive search

How to use the retrieved information ?

- Input/Data Layer
- Model/Intermediate Layer
- Output/Prediction Layer

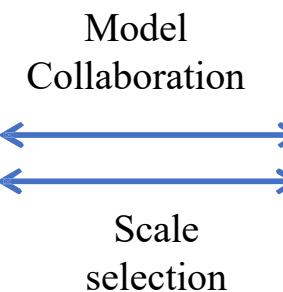
Other Issues

Augmentation stage:

- Pre-training
- Fine-tuning
- Inference

Retrieval choice:

- BERT
- Roberta
- BGE
-



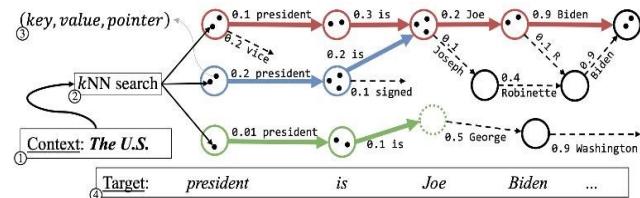
Generation choice:

- GPT
- Llama
- T5
-

What to retrieve

coarse

Chunk | In-Context RAG 2023



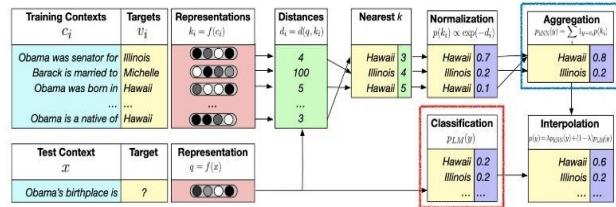
The search is **broad**, recalling a large amount of information, but with low **accuracy**, high coverage but includes much **redundant information**.

Retrieval granularity

Phrase | NPM 2023



Token | KNN-LMM 2019

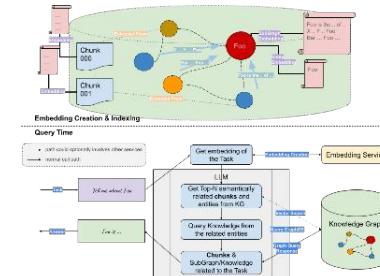


meticulous

low

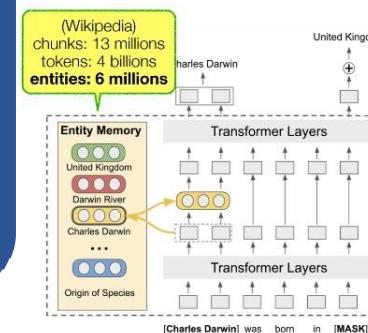
level of structuration

Knowledge Graph | 2023



Richer semantic and **structured information**, but the retrieval efficiency is lower and is limited by the quality of KG.

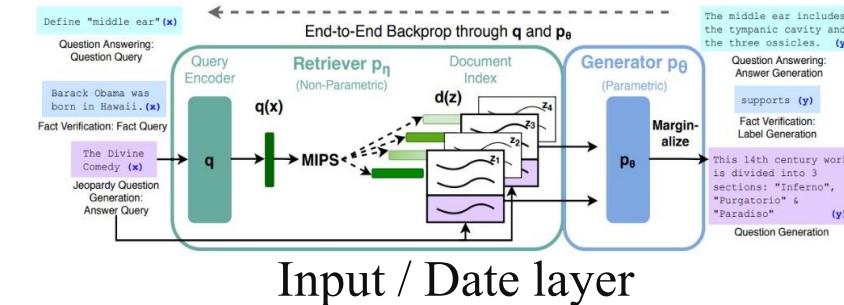
Entity | EasE 2022



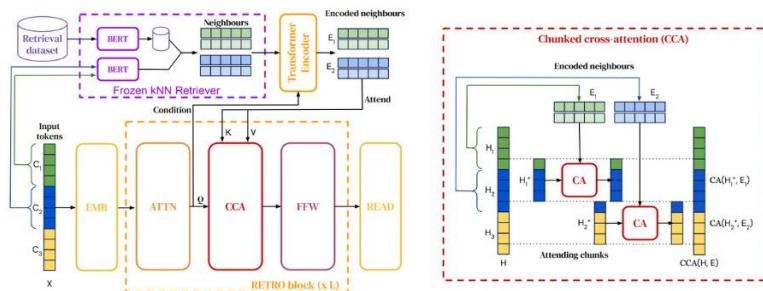
High

How to use the retrieved content

Integrating the retrieved information into different layers of the generation model, during inference process.

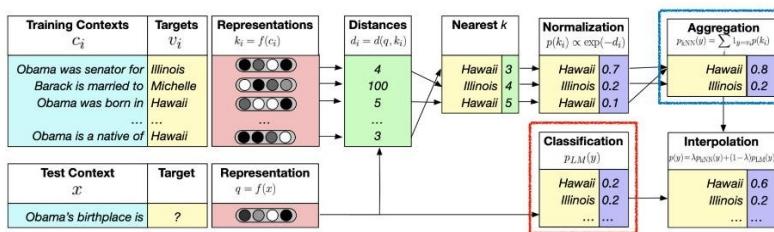


Integrate retrieval positions.



Using simple, but unable to support the retrieval of **more knowledge blocks**, and the **optimization space** is limited.

Supports the retrieval of more knowledge blocks, but introduces **additional complexity** and **must be trained**.



Ensuring the output results are **highly relevant** to the retrieval content, but the efficiency is low.

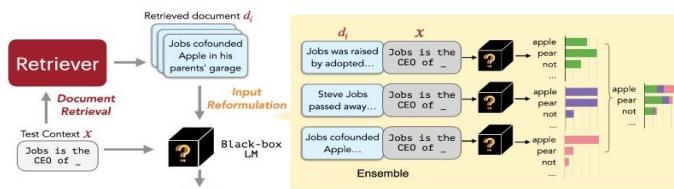
When to retrieve

High efficiency, but low relevance of the retrieved documents

Balancing efficiency and information might not yield the optimal solution

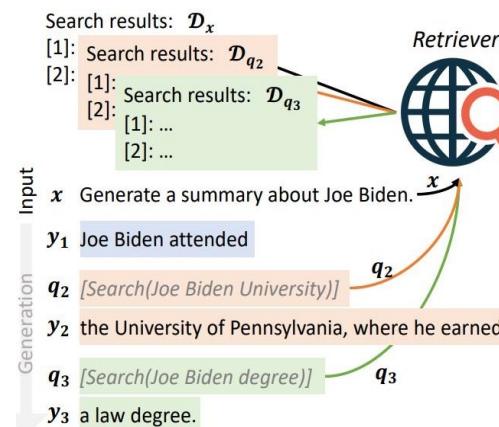
A large amount of information with low efficiency and redundant information.

Once | Replug 2023



Conducting once search during the reasoning process.

Adaptive | Flare 2023



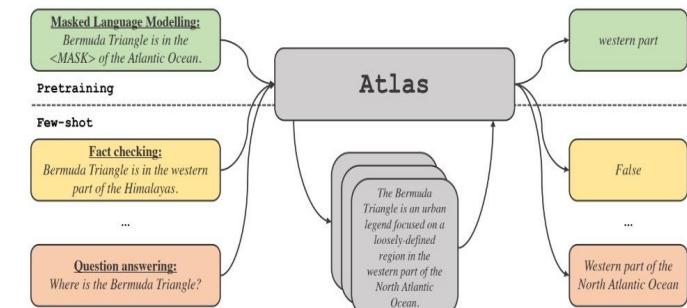
Adaptively conduct the search.

Low

Retrieval frequency

High

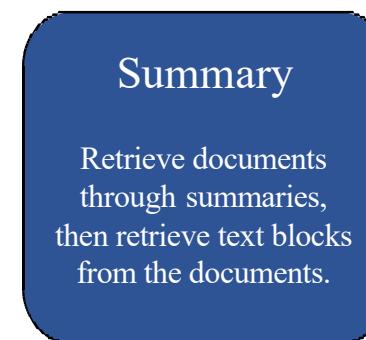
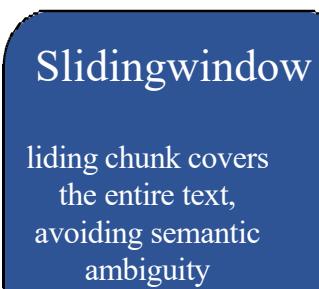
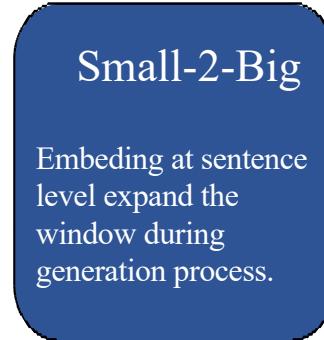
Every N Tokens | Atlas 2023



Retrieve once for every N tokens generated.

Data indexing optimization

Chunk Optimization



Small-2-Big

Question:
What are the concerns surrounding the AMOC?

Embed Sentence → Link to Expanded Window

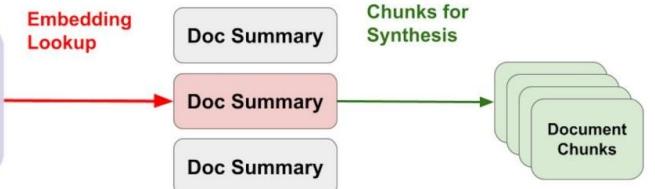
Continuous observation of the Atlantic meridional overturning circulation (AMOC) has improved the understanding of its variability (Frajka-Williams et al., 2019), but there is low confidence in the quantification of AMOC changes in the 20th century because of low agreement in quantitative reconstructed and simulated trends. Direct observational records since the mid-2000s remain too short to determine the relative contribution of natural climate forcing, internal forcing and anthropogenic forcing to AMOC change (high confidence). Over the 21st century, AMOC will very likely decline for all SSP scenarios but will not involve an abrupt collapse before 2100. 3.2.2.4 Sea Ice Changes

Sea ice is a key driver of polar marine life, hosting unique ecosystems and affecting diverse marine organisms and food webs through its impact on light penetration and supplies of nutrients and organic matter (Arrigo, 2014).

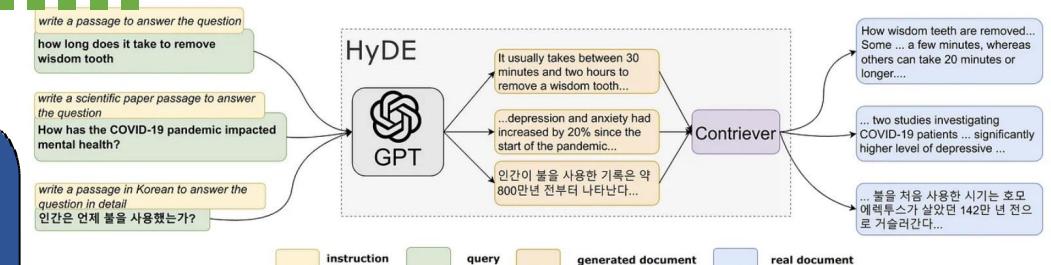
What the LLM Sees
What the LLM Sees

Abstract

Question:
What are the concerns surrounding the AMOC?



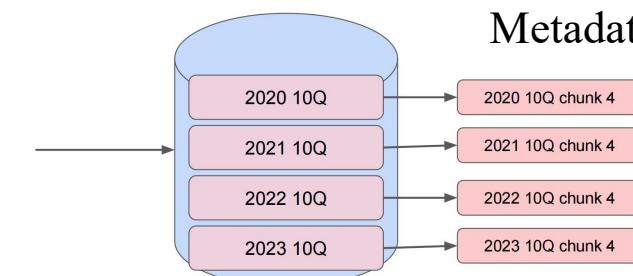
Pseudo Metadata



Metadata filter

query_str:
<query_embedding>

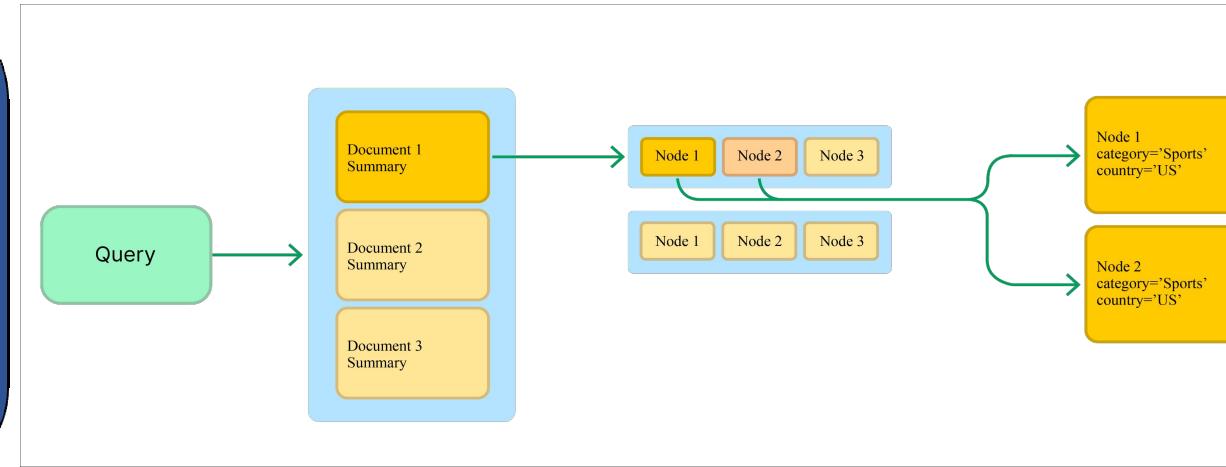
Metadata tags:
<metadata_tags>



Structured Corpus

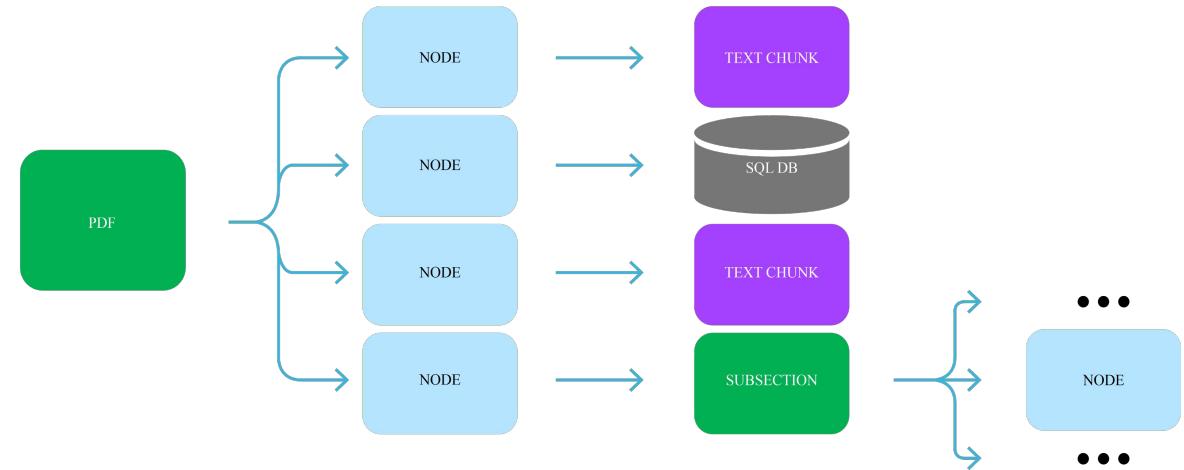
Hierarchical Organization of Retrieval Corpora

- Summary → Document
Replace document retrieval with summary retrieval, not only retrieving the most directly relevant nodes, but also exploring additional nodes associated with those nodes.



- Document → Embedded Objects

Documents have embedded objects (such as tables, charts), first retrieve entity reference objects, then query underlying objects, such as document blocks, databases, sub-nodes.



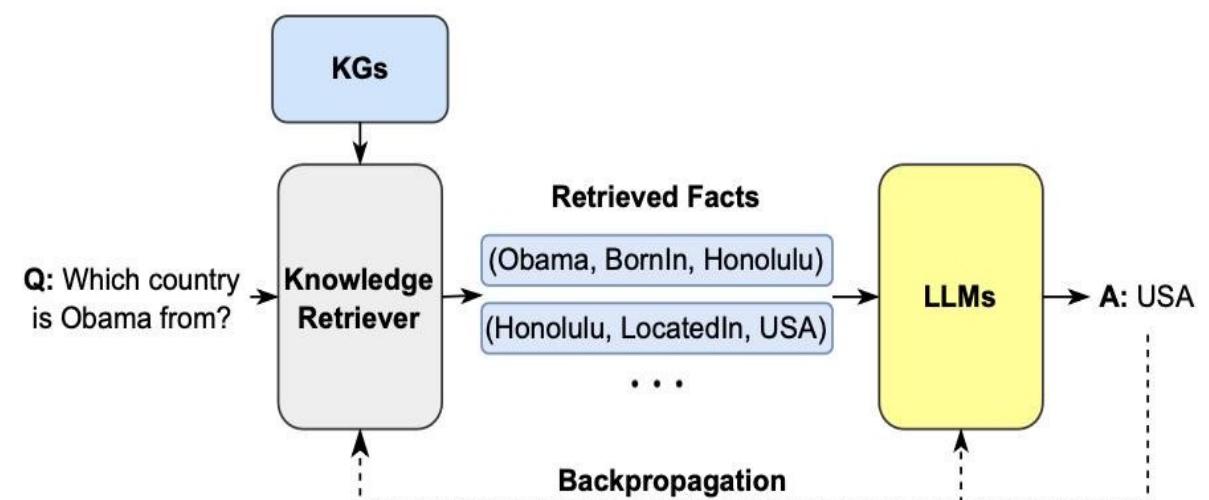
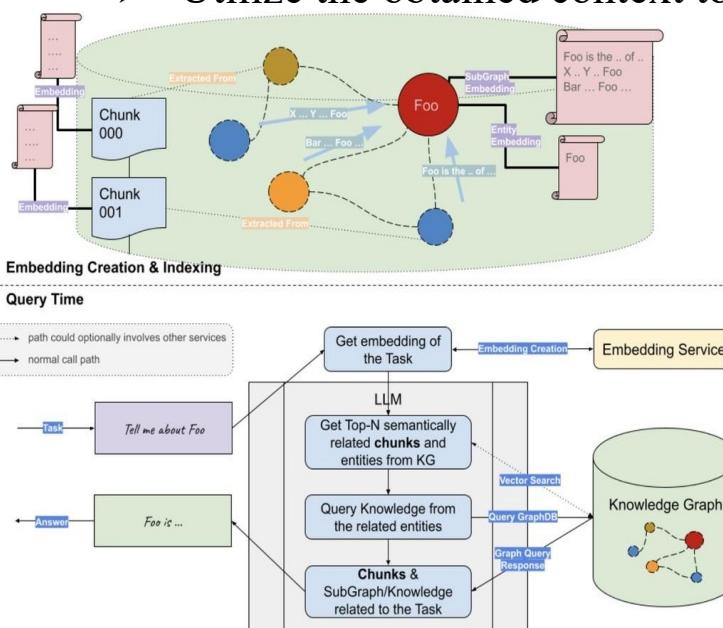
KG as a Retrieval Data Source

➤ GraphRAG

- Extract entities from the user's input query, then construct a subgraph to form context, and finally feed it into the large model for generation.

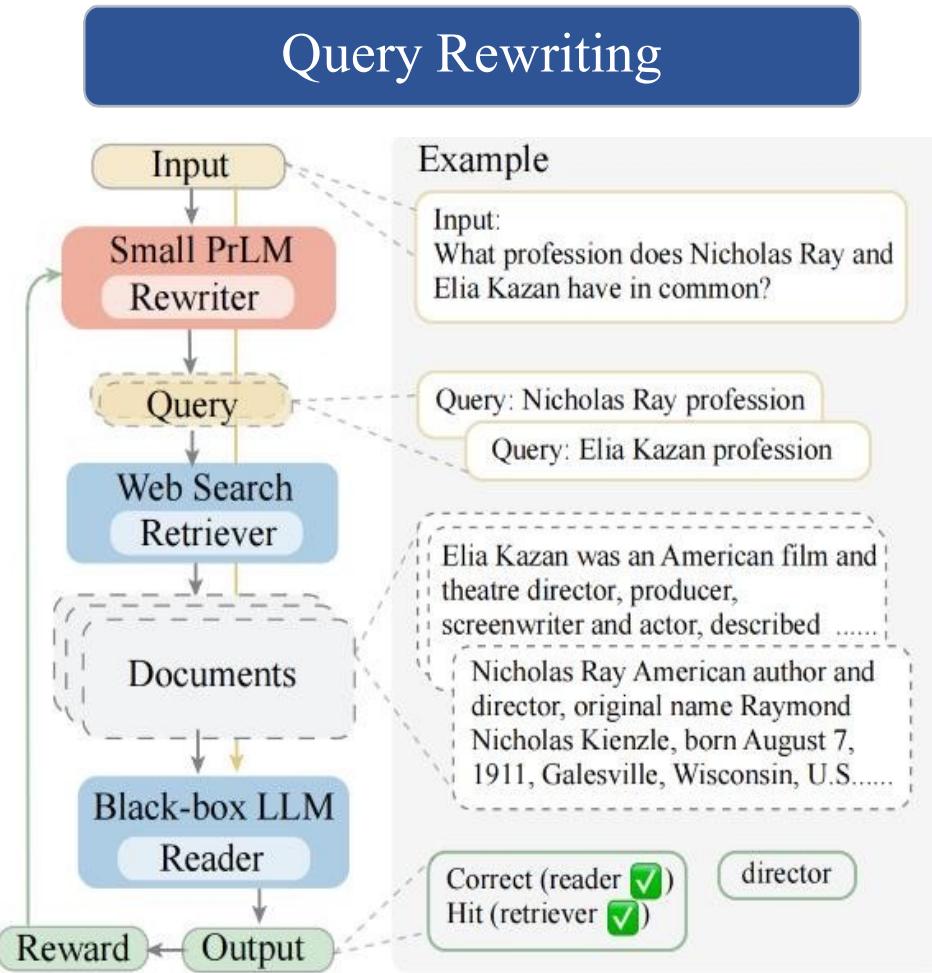
➤ Implementation

- Use LLM (or other models) to extract key entities from the question.
- Retrieve subgraphs based on entities, delving to a certain depth, such as 2 hops or even more.
- Utilize the obtained context to generate answers through LLM.

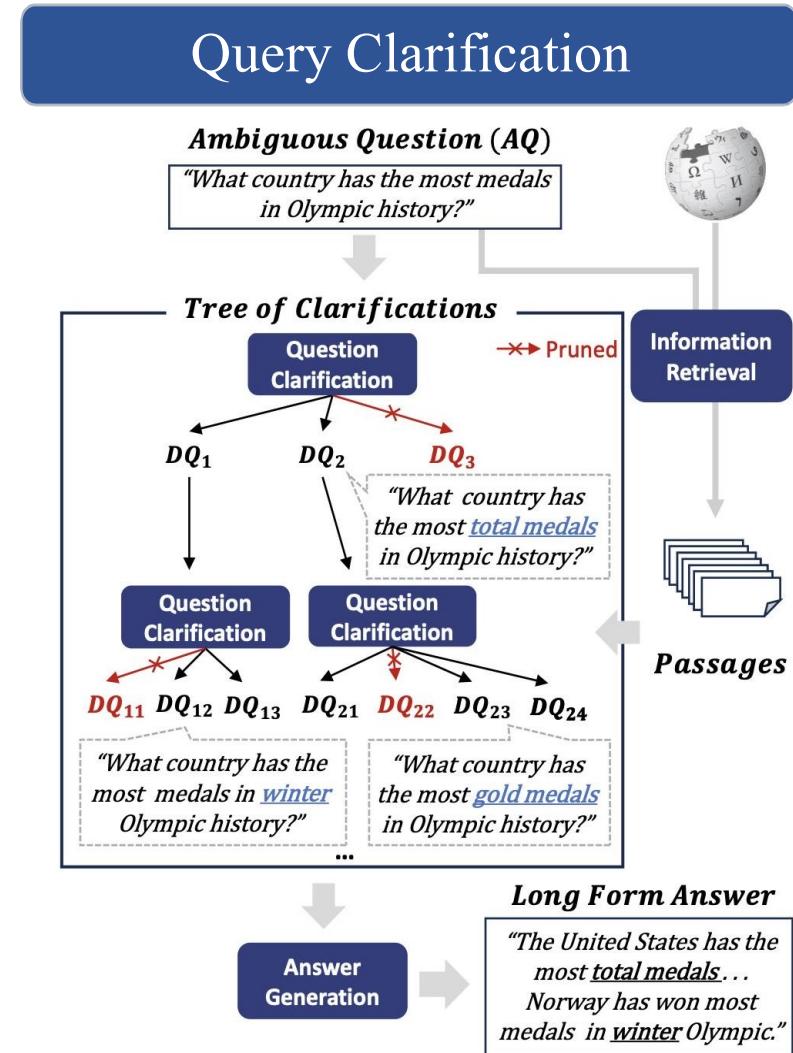


Query Optimization

Questions and answers do not always possess high semantic similarity;
adjusting the query can yield better retrieval results.



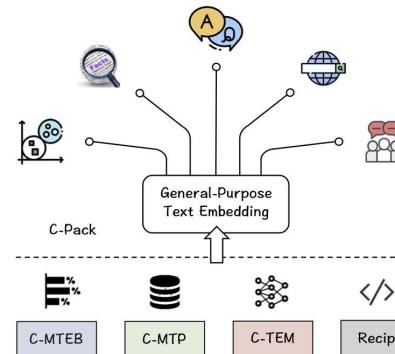
Rewrite-Retrieve-Read [Ma et al., 2023]



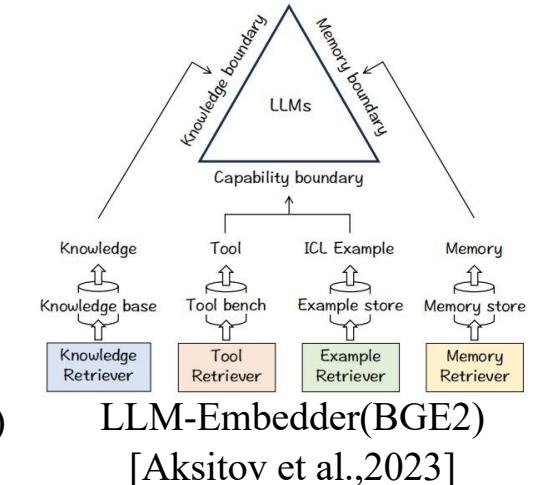
Tree of Clarifications (TOC) [Kim et al., 2023]

Embedding Optimization

Selecting a More Suitable Embedding Provider

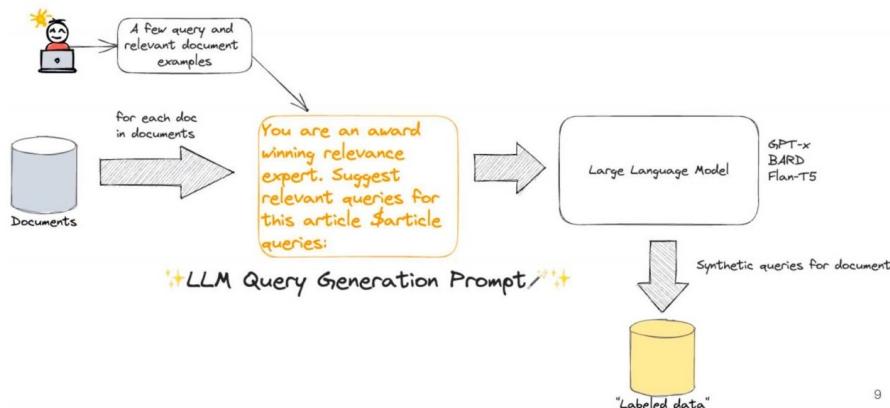


BAAI-General-Embedding (BGE)



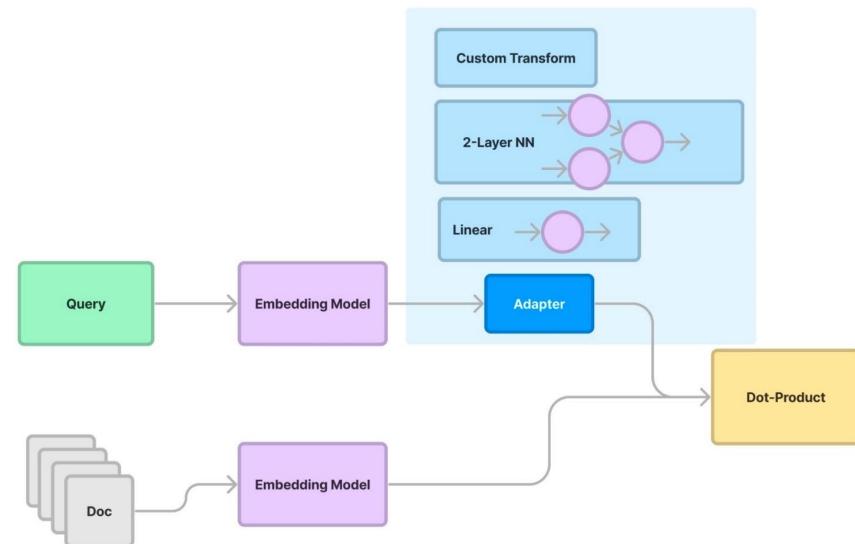
LLM-Embedder(BGE2)
[Aksitov et al., 2023]

Fine-tuning the Embedding Model



9

Fine-tuning According to Domain-Specific Repositories and Downstream Tasks

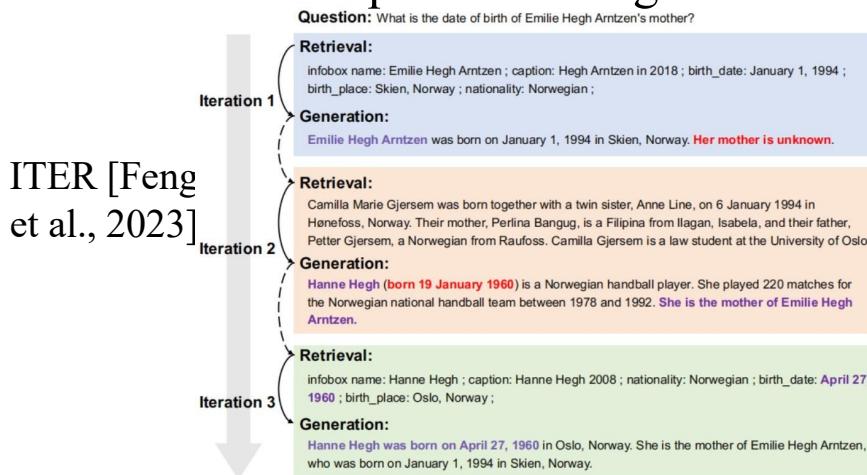


Fine-tuning the Adapter Module to Align the Embedding Model with the Retrieval Repository

Retrieval Process Optimization

Iterative

Iteratively Retrieving from the Corpus to Acquire More Detailed and In-depth Knowledge



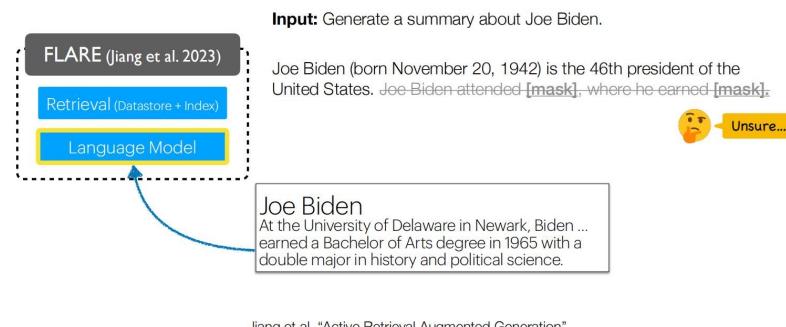
ITER [Feng et al., 2023]



IRCoT [Trivedi et al., 2022]

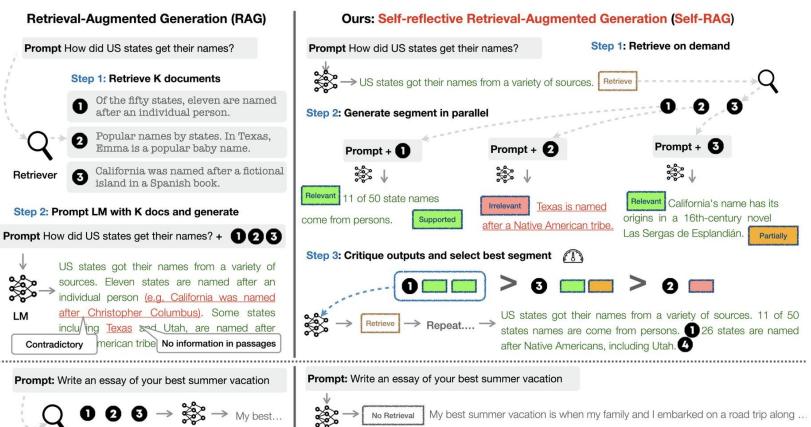
Adaptive

Dynamically Determined by the LLM, the Timing and Scope of Retrieval



Jiang et al. "Active Retrieval Augmented Generation"

FLARE [Jiang et al., 2023]



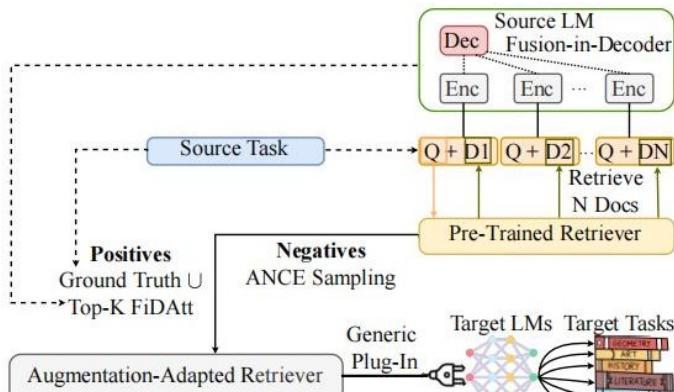
Self-RAG [Asai et al., 2023]

Hybrid (RAG + Fine-tuning)

Retriever Fine-Tuning

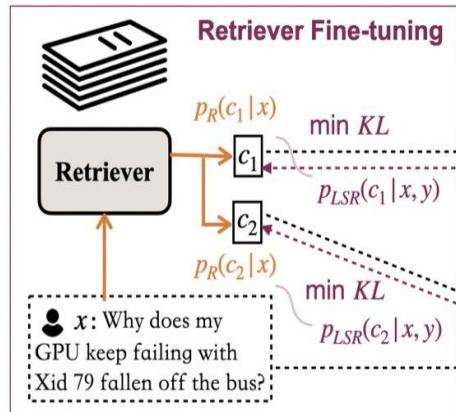
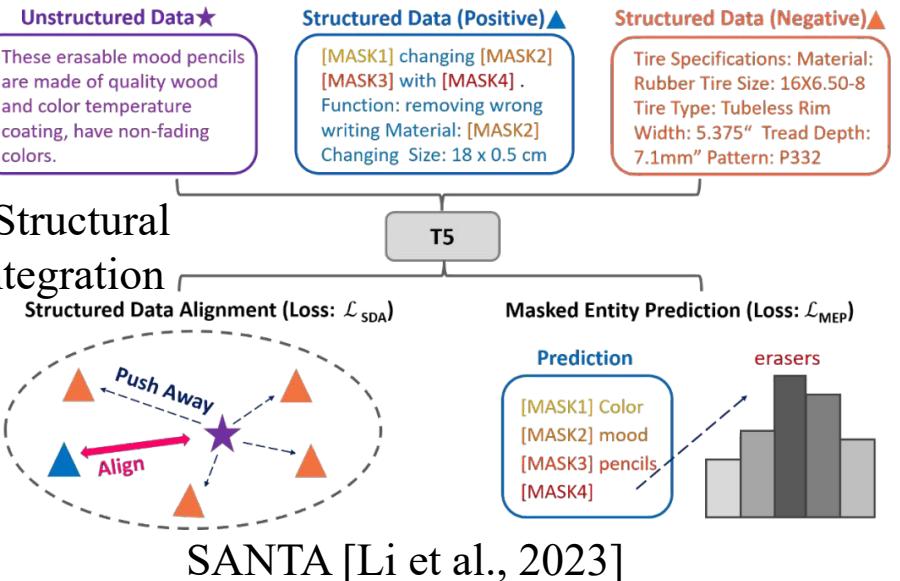
Highly Adaptive General-Purpose Retrieval Plugin

Generator Fine-Tuning



Augment with Structural Information Integration
AAR
[Yu et al., 2023]

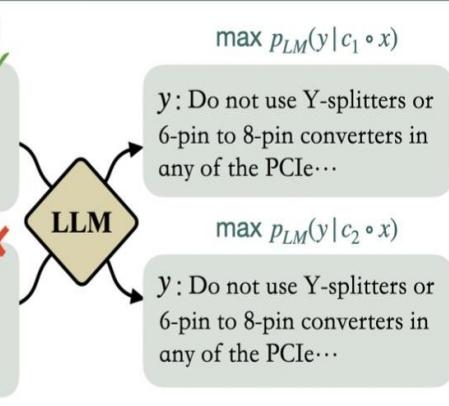
Collaborative Fine-Tuning



RA-DIT [Lin et al., 2023]

Retrieval-augmented Instruction Tuning

- 1 **Background:** I assume that the BGA chip has damage to the substrate level ... \n\n**Q:** Why does my GPU keep failing with Xid 79 fallen off the bus? **A:**
- 2 **Background:** Microsoft should withdraw from the hardware market ... \n\n**Question:** Why does my GPU keep failing with Xid 79 fallen off the bus? **Answer:**



- **R-FT**

Minimizing the KL Divergence Between the Retriever Distribution and LLM Preferences

- **LM-FT**

Maximizing the Likelihood of the Correct Answer Given Retrieval-Augmented Instructions

How to Evaluate

Evaluation Methods

Retriever

Evaluate the Quality of Text Blocks Retrieved by the Query Metrics:
MRP, Hit Rate, NDCG

Independent Evaluation

Generation/Synthesis

Quality of Context Enhanced with Retrieved Documents Evaluation Metrics: Context Relevance

End-to-End Evaluation

Evaluate the content ultimately generated by the model.

By generated content

With labels: EM, Accuracy
Without labels: Fidelity,
Relevance, Harmlessness

By evaluation method

Human evaluation Automatic evaluation (LLM judge)

Key Metrics & Capabilities

Key Metrics

Answer Relevance Is the answer relevant to the query?

Query

Context Relevance: Is the context enhanced with retrieved documents relevant to the query?

Answer

Context

Answer Fidelity:

Is the answer based on the given context?

Key Capabilities

Noise Robustness Can the model extract useful information from noisy documents?

Negative Rejection When required knowledge is not existing in the retrieved documents, the answer should be refused.

Info Integration Can the model answer complex questions that require integrating information from multiple documents?

Counterfactual Robustness Can the model recognize the risk of known factual errors in the retrieved documents?

Assessment Framework

TruLens

RAGAS

ARES

Use LLM as the adjudicator judge.

Based on handwritten prompt

Synthetic dataset + Fine-tuning + Ranking using confidence intervals

Evaluation

- Answer Fidelity
- Answer Relevance
- Contextual Relevance

- 检索增强生成（RAG）理论介绍
 - 基础介绍
 - 优化方案
- PolarDB社区问答机器人项目实践
 - 项目背景
 - 实施方案



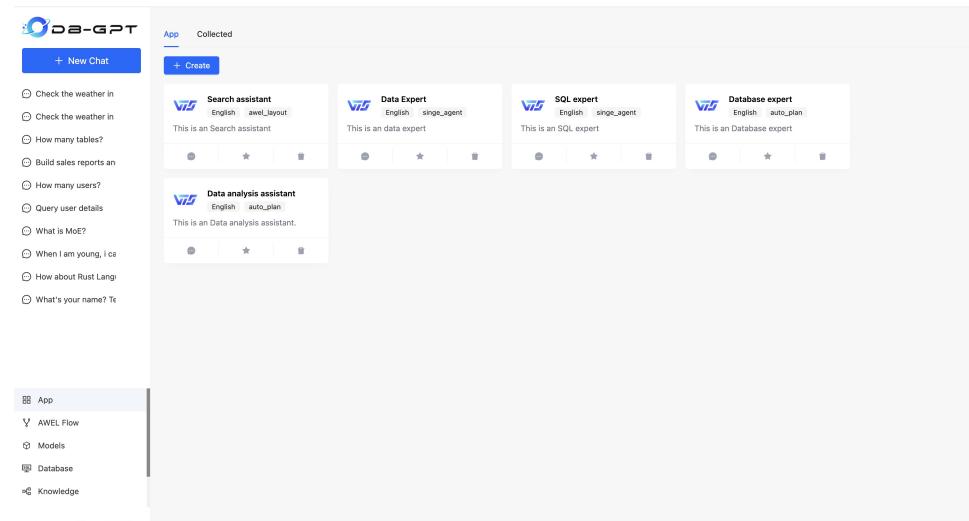
PolarDB

PolarDB开源社区是阿里云数据库开源产品PolarDB的技术交流平台。作为一款开源的数据库产品，离不开用户和开发者的支持，大家可以在社区针对PolarDB产品提问题、功能需求、交流使用心得、分享最佳实践、提交issue、贡献代码等。

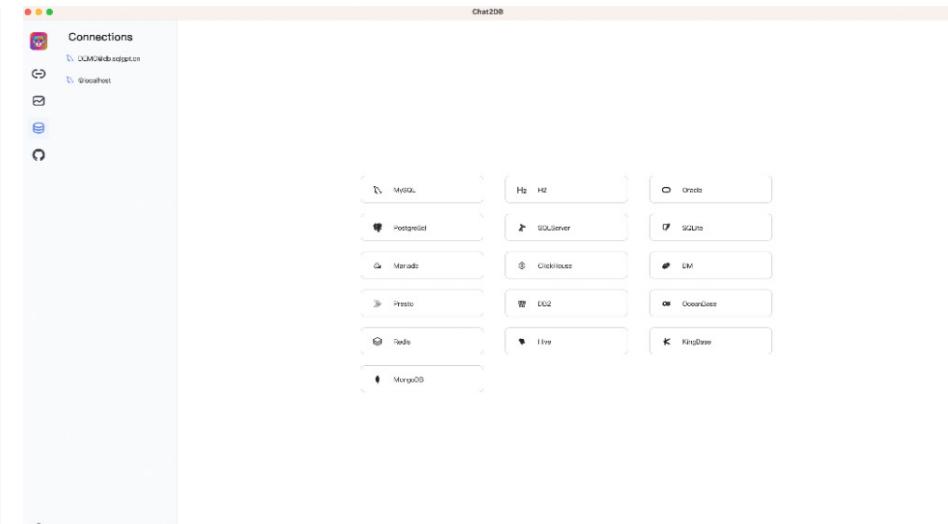


随着以ChatGPT为代表的大语言模型的出现，人工智能迎来了新一轮的高潮。其在各个领域的应用都受到了关注。我们思考：是否能够将PolarDB开源社区和大语言模型结合起来，用以支撑开源社区的问答业务？

结合PolarDB 开源社区代码、文档、Issue、PR comment 等数据，开发一款领域知识定制化的社区智慧问答助手工具，能够给 PolarDB 旗下的多个项目提供智慧问答以及辅助代码开发服务。



DB-GPT



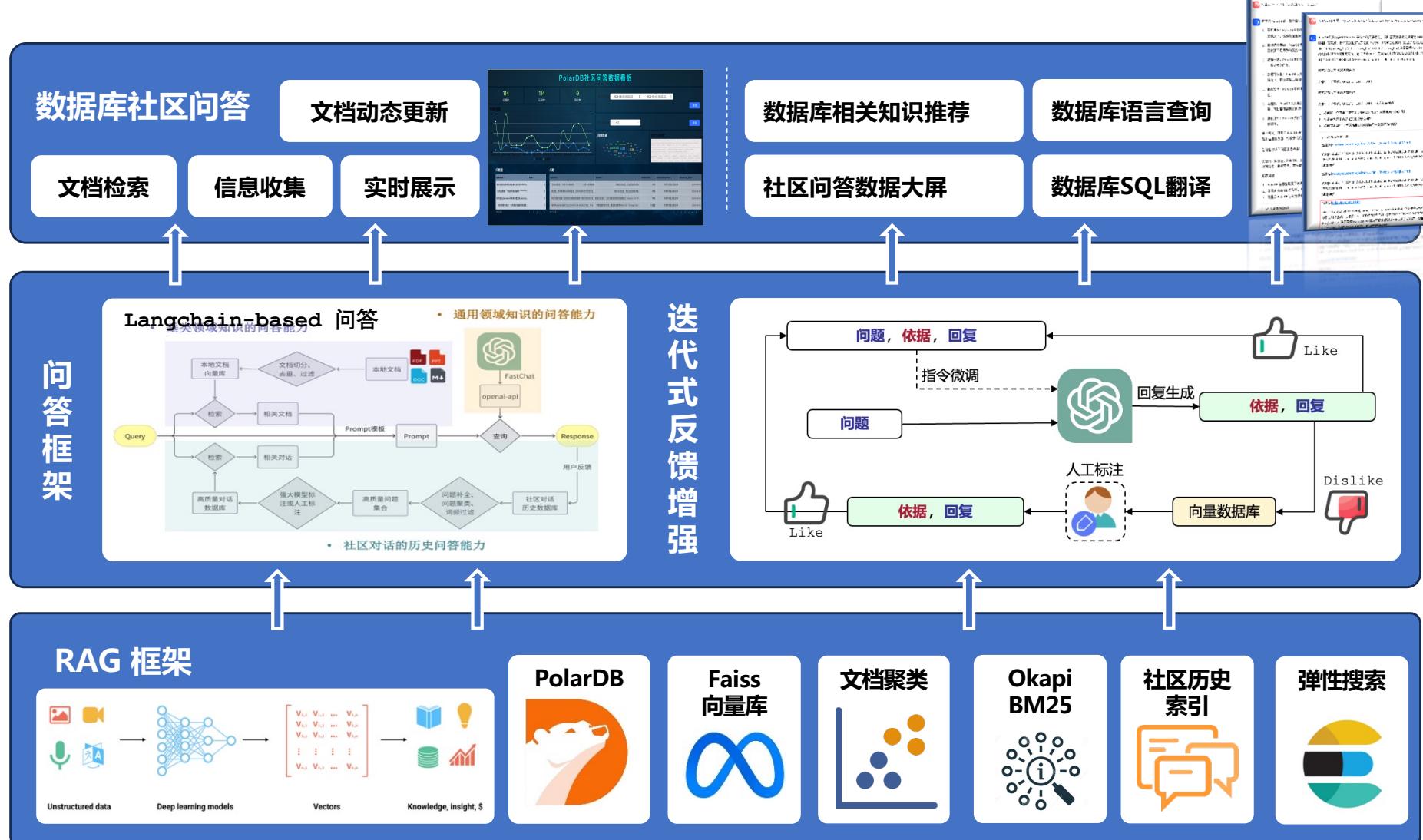
Chat2DB

- 能否使用大模型快速运用到社区这样的数据碎片化、结构繁多的垂类领域是值得探究的问题。
- 如何借助社区的力量提升模型表现

- 开源**数据库使用、编码、PolarDB社区相关问题**的回答能力。支持对于传统SQL代码转换为PolarDB-X的分布式查询代码
- 具备对语料**良好准确的理解、总结、概括、阐述能力**，针对用户发起的问题，提供**专业准确的回答和畅通的对话**
- 两种服务场景：**群问答机器人**和**网页端服务**
- 指定场景下自动向值班人员发送消息推送能力
- 数据统计用于效果分析
- 支持**自主训练，持续迭代**，不断提高准确度，支持开放API能力
- 围绕机器人项目，有详细的手册和课程介绍，打造**垂类领域智能问答的标准开源解决方案**，通过开源不断迭代，通过开源推广整个项目

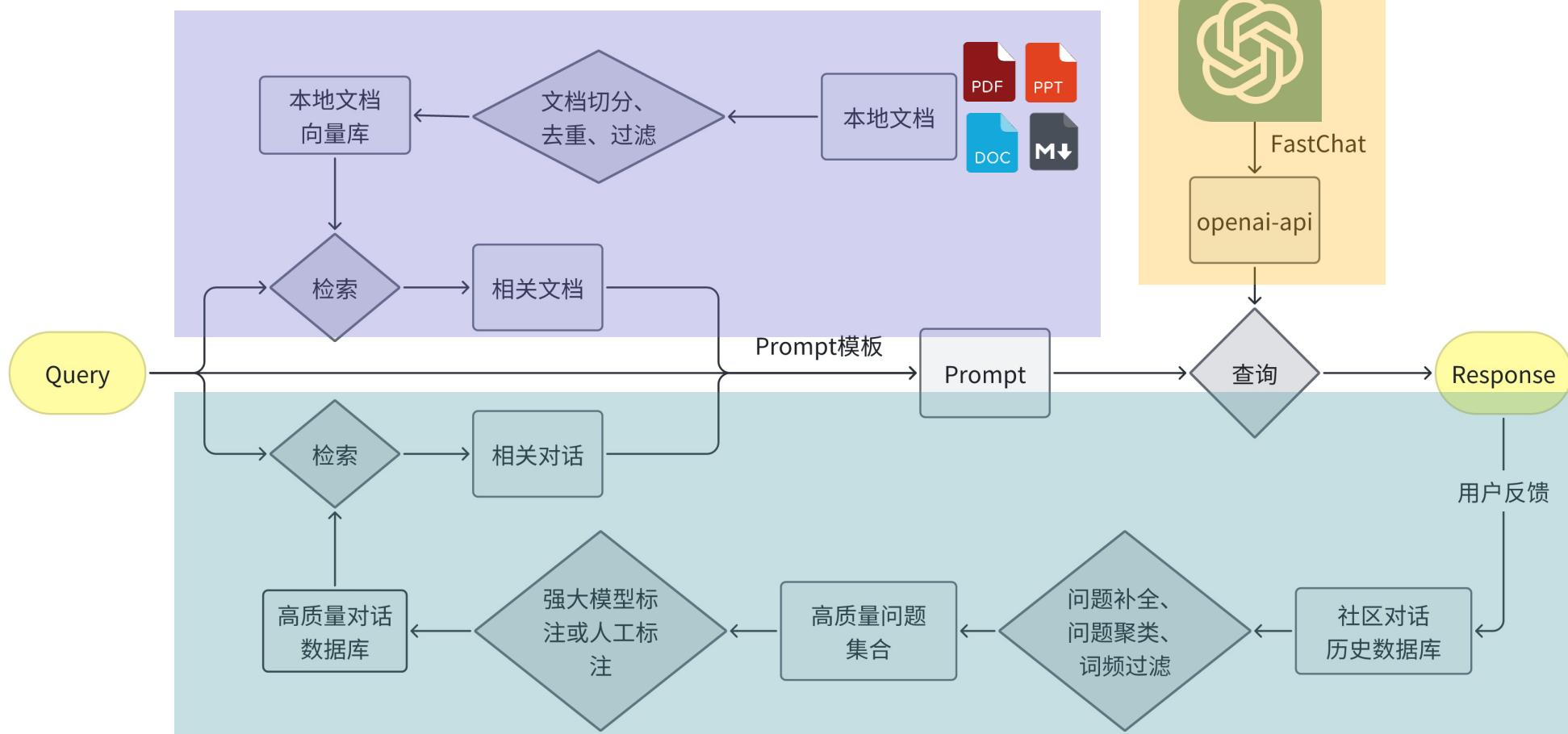
- 检索增强生成（RAG）理论介绍
 - 基础介绍
 - 优化方案
- PolarDB社区问答机器人项目实践
 - 项目背景
 - 实施方案

项目方案：融合大语言模型的社区问答系统



项目方案——基于langchain的静态问答框架

- 垂类领域知识的问答能力
- 通用领域知识的问答能力



- 社区对话的历史问答能力

- 基于文档类型读取文档并切分数据
 - 支持HTML, MD, JSON, JSONL, CSV, PDF, PNG, JPG, JPEG, BMP, EML, MSG, EPUB, XLSX, XLSD, IPYNB, ODT, PY, RST, RTF, SRT, TOML, TSV, DOCX, DOC, XML, PPT, PPTX, TXT, HTM多种格式数据
 - 基于不同数据类型选择合适的切分方式，例如
 - MarkDown数据#后为标题数据
 - HTML数据中根据标签切分、合并数据
- 切分后处理
 - 文档去重
 - 文档质量过滤
 - 文档冲突处理
- 文档数据存储在向量数据库中
 - 通过embedding model获得文档的语义向量
 - 将文档和语义向量存入PolarDB数据库中

- Rewrite
 - 多轮对话Query重写（问题拼接、大模型生成）
 - 对话实体抽取与意图识别（搜索引擎的常用方法）
 - 实体消歧
 - Pseudo-Relevance Feedback（基于LLM的query改写）
- Retrieval
 - 向量相似度召回
 - 计算两条（问题和文档）语义向量的余弦相似度作为文档得分（Bi-Encoder）
 - BM25召回
 - 基于TF-IDF的召回策略，可以更好的保存实体信息
- Rerank
 - Cross-Encoder 模型
 - LLM PPL（Perplexity）

Pseudo-Relevance Feedback

Pseudo-Relevance Feedback是在信息检索领域进行Query重写的常见方法

通常使用得到的伪文档和query一起拼接成为新的query进行查询

New query = D1+D2+D3+query

伪文档通常包含高度相关的信息，这些信息可以帮助消除查询歧义并指导Retriever

不同的伪文档的获取方法会影响query重写的结果

- LLM生成伪文档
- 检索得到伪文档

LLM Prompts

Write a passage that answers the given query:

Query: what state is this zip code 85282

Passage: Welcome to TEMPE, AZ 85282.
85282 is a rural zip code in Tempe, Arizona.
The population is primarily white...

...

Query: when was pokemon green released

Passage:

LLM Output

Pokemon Green was released in Japan on February 27th, 1996. It was the first in the Pokemon series of games and served as the basis for Pokemon Red and Blue, which were released in the US in 1998. The original Pokemon Green remains a beloved classic among fans of the series.

[Query2doc: Query Expansion with Large Language Models](#)

- 向量相似度召回
 - Embedding 是用一个低维稠密向量来表示一个对象，使得这个向量能够表达相应用对象的某些特征，同时向量之间的距离能反应用对象之间的相似性。
 - Word2Vec、GloVe、FastText、ELMo、GPT和BERT（BGE系列模型）
 - Scores = $q_embeddings^T p_embeddings$
- BM25召回
 - BM25是信息索引领域用来计算query与文档相似度得分的经典算法。
 - BM25的公式主要由三个部分组成：
 - query中每个单词q与文档d之间的相关性
 - 单词q与query之间的相似性
 - 每个单词的权重
 - $Score(Q, d) = \sum_i^n W_i R(q_i, d)$
 - 相比于相似度检索，BM25可以更好的保存实体信息，但是检索效果不如向量相似度

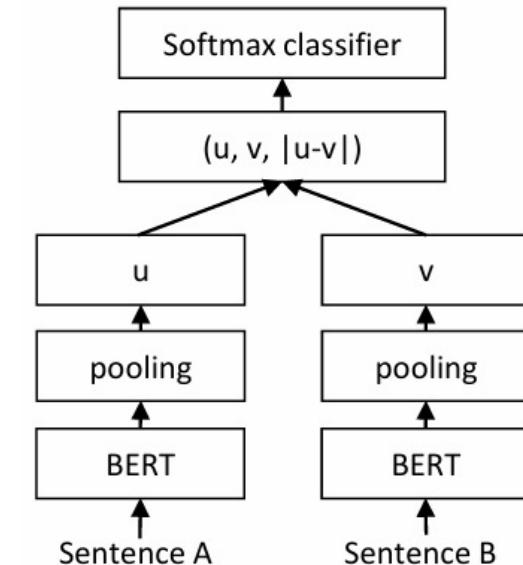
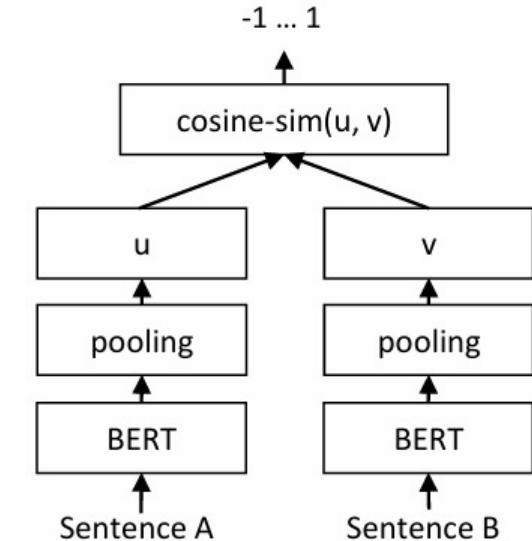
Cross-Encoder 模型

Bi-Encoders：给定句子生成句子的embedding。我们将句子A和B独立地传递给BERT，从而产生句子嵌入 u 和 v ，然后可以使用余弦相似度比较。

Cross-Encoder：同时将两个句子传递给Transformer网络。它产生一个介于0和1之间的输出值，表示输入句对的相似性。**不产生句子的embedding**。并且，无法将单个句子传递给Cross-Encoder。

Cross-Encoder比Bi-Encoder具有更高的性能，但是也会带来更高的复杂度。Bi-Encoders在检索时只需要提前对文档库进行一次embedding操作，而Cross-Encoder需要在针对不同问题在每次提问时将所有的文档重新组合并计算相似度

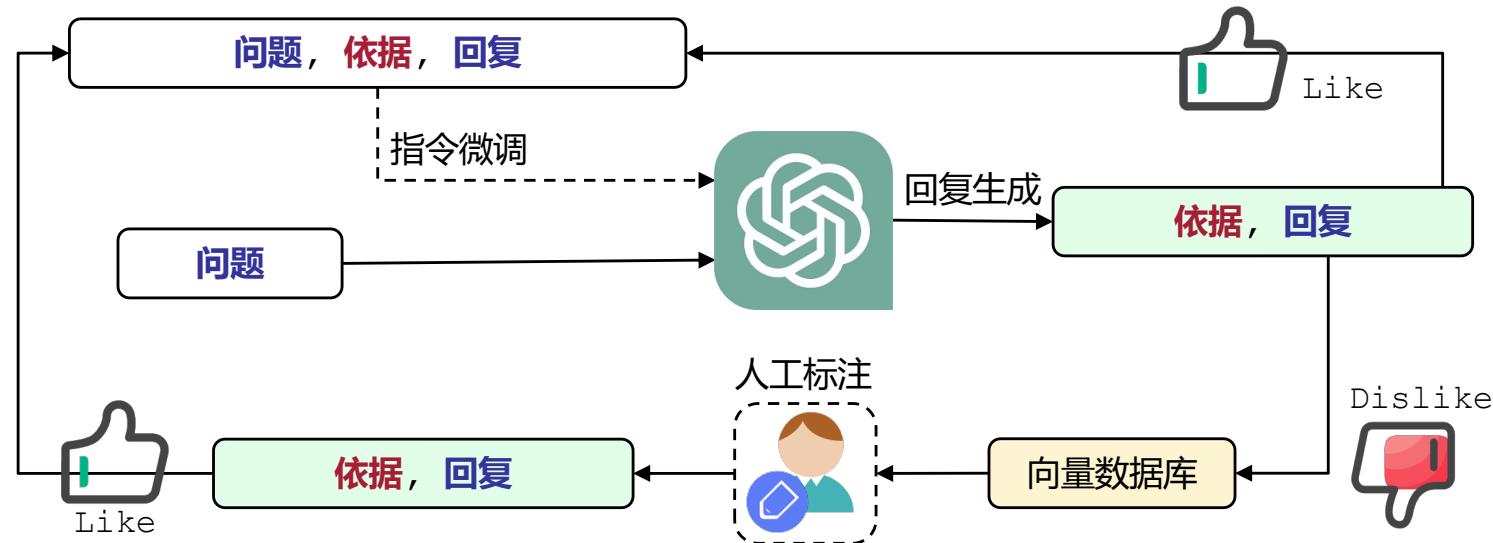
所以Bi-Encoder效率更高，更适合在retrieval中从海量数据中检索到一部分相关文档。而Cross-Encoder准确率更高，更适合在retrieval后的相关文档中重新排序，找出真正相关的文档



LLM PPL (Perplexity)

- GPT系列的causal language model在生成内容时，会根据一句话的前k个词，来预测第k+1个词是什么，循环以上步骤得到完整的生成内容。
- 即给出一个第k+1个词可能出现的概率的分布
 - $p(x_{k+1}|x_1, x_2, \dots, x_k)$
- 根据联合概率分布，模型生成一句话的概率为
 - $p(x_1, x_2, \dots, x_k) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\dots p(x_k|x_1, x_2, \dots, x_{k-1})$
- 困惑度 (perplexity) 的基本思想是：给测试集的句子赋予较高概率值的语言模型较好
 - $ppl = p(x_1, x_2, \dots, x_N)^{-\frac{1}{N}}$
- 而使用PPL计算文档和query的相关程度的基本思想是：更相关的文档可以提高LLM生成query的概率
- 即计算给定文档的基础上，生成query的ppl
 - $scores = p(x_1, x_2, \dots, x_N | D)^{-\frac{1}{N}}$

- 通过指令微调引入用户反馈机制



polardb是什么

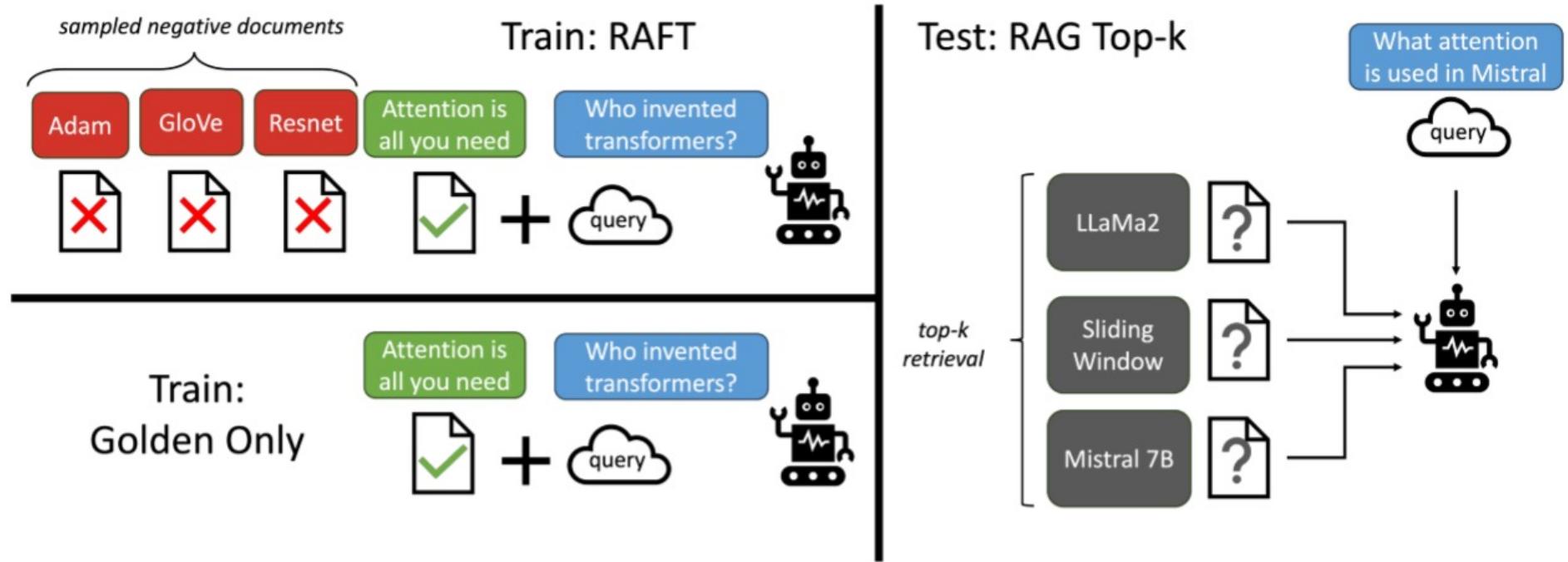
PolarDB是阿里巴巴云推出的一种新型关系型数据库服务，它是基于分布式架构设计的，可以提供高可用性、可扩展性和高性能等特点。PolarDB采用了存储计算分离的技术，支持在线扩容和缩容，能够满足不同业务场景的需求。此外，它还提供了丰富的数据库管理功能，如备份恢复、监控告警、安全审计等，方便用户进行数据管理和运维。总之，PolarDB是一种高效、稳定、易用的云数据库服务，适合于各种互联网和企业级应用。

✓ 对话历史匹配结果



欢迎反馈您打分的理由

SUBMIT



$$\mathbf{P} \% \text{ of data: } \mathbf{Q} + \mathbf{D}^* + \mathbf{D}_2 + \dots + \mathbf{D}_k \rightarrow \mathbf{A}^*$$

$$(1 - \mathbf{P}) \% \text{ of data: } \mathbf{Q} + \mathbf{D}_1 + \mathbf{D}_2 + \dots + \mathbf{D}_k \rightarrow \mathbf{A}^*$$

添加噪点信息提升模型的鲁棒性

RAFT: Adapting Language Model to Domain Specific RAG

- 计算机常识问答
 - 例：请阐述作业、进程、程序三者之间的关系？
- PolarDB 问答
 - 例：在PolarDB中，如何有效地管理并优化存储引擎的读取性能，特别是在高并发和大规模数据场景下？
- PolarDB SQL 纠错
 - 例：PolarDB SQL 纠错：UPDATE t1SETc 1=(SELECT c2FROMt2WHEREt 1.c 1=t 2.c 1) LIMIT 10 ?
- 多轮对话沟通
 - 例：Q1：PolarDB能增加节点吗？
Q2：怎么增加节点？ Q3：具体怎么操作？

项目方案——模型基座的选择

- 在计算机常识问答、PolarDB 问答、
PolarDB SQL 纠错与多轮对话任务评测
- Qwen-14B-Chat 目前是最符合我们要求的
模型

计算机常识问答							
	使用知识库的执行时间	不使用知识库的执行时间	cos_sim	bleu4			
chatglm2-6b	1388.389226436615	979.0877907276154	0.6291	0.010436571437401806			
Qwen-14B-Chat	12 03	PolarDB问答					
baichuan-13b-chat	16 27	使用知识库的执行时间	不使用知识库的执行时间	cos_sim	BLEU4	使用知识库回答总长度	答案的平均长度
vicuna-13b-v1.5	10 chatglm2-6b	416.77390456199646	625.9088687896729	0.7205	0.009844454616382423	31908	343.10
chatglm3-6b	11 Qwen-14B-Chat	642.6580767631531	多轮对话				
	baichuan-13b-chat	1099.2815444469452	model	BLEU1	BLEU2	BLEU3	
	vicuna-13b-v1.5	1349.2528622150421	chatglm2-6b	0.14344014065805855	0.05161342273818349	0.028914945919139165	
	chatglm3-6b	557.3187487125397	chatglm3-6b	0.1404026151466831	0.053235349376040936	0.02956100022190767	
			vicuna-13b-v1.5	0.1433787442087	0.053128487250324485	0.029801093011219727	
			baichuan-13b-chat	0.1569741680495669	0.0644386685660481	0.03639450310489442	
			Qwen-14B-Chat	0.18037539055564028	0.08151790357279665	0.048873824669316154	

Text2sql纠错 Execution accuracy					
method	easy (163)	medium (298)	hard (114)	extra (90)	all (665)
chatglm2-6b	0.135 (0.153)	0.057 (0.054)	0.061 (0.053)	0.011 (0.011)	0.071 (0.072)
chatglm3-6b	0.239 (0.153)	0.101 (0.054)	0.114 (0.053)	0.000 (0.011)	0.123 (0.072)
baichuan-13b-chat	0.215 (0.178)	0.124 (0.114)	0.079 (0.079)	0.011 (0.011)	0.123 (0.110)
vicuna-13b-v1.5	0.117 (0.129)	0.044 (0.030)	0.009 (0.000)	0.000 (0.000)	0.050 (0.045)
Qwen-14B-Chat	0.356 (0.313)	0.221 (0.218)	0.219 (0.228)	0.067 (0.089)	0.233 (0.226)
codegen25-7b-multi	0.061 (0.043)	0.023 (0.040)	0.018 (0.009)	0.000 (0.000)	0.029 (0.030)
Text2sql纠错 Exact matching accuracy					
codegeex2-6b	method	easy (163)	medium (298)	hard (114)	extra (90)
sqlcoder2	chatglm2-6b	0.129 (0.153)	0.037 (0.044)	0.061 (0.053)	0.000 (0.000)
	chatglm3-6b	0.264 (0.153)	0.094 (0.044)	0.088 (0.053)	0.000 (0.000)
	baichuan-13b-chat	0.233 (0.172)	0.077 (0.077)	0.070 (0.070)	0.000 (0.000)
	vicuna-13b-v1.5	0.129 (0.147)	0.034 (0.023)	0.009 (0.000)	0.000 (0.000)
	Qwen-14B-Chat	0.380 (0.331)	0.191 (0.171)	0.211 (0.202)	0.011 (0.022)
	codegen25-7b-multi	0.061 (0.086)	0.017 (0.057)	0.026 (0.009)	0.000 (0.000)
	codegeex2-6b	0.000 (0.000)	0.003 (0.000)	0.000 (0.000)	0.002 (0.000)
	sqlcoder2	0.172 (0.160)	0.037 (0.034)	0.061 (0.044)	0.000 (0.000)

项目方案——外部文档的整理

• PolarDB 垂类知识文档：

This screenshot shows the official technical documentation for PolarDB PostgreSQL. It features a navigation bar with links like '产品', '应用实践', '文档', '博客', '社区', '生态', '下载', '活动', and '关于我们'. Below the navigation is a search bar. The main content area is divided into sections: '备份恢复' (Backup and Recovery), '架构简介' (Architecture Introduction), '文件系统图' (File System Diagram), and '数据目录结构' (Data Directory Structure). The architecture diagram illustrates a distributed system where multiple PolarDB nodes (PolarDB-PG) interact with a central PostgreSQL node, all connected to a shared storage system (Shared Storage). The file system diagram shows the directory structure for PostgreSQL, including paths for binlogs, checkpoints, and shared memory.

PolarDB PostgreSQL 官方技术文档

This screenshot shows the official technical documentation for PolarDB-X. The layout is similar to the PostgreSQL version, with a navigation bar and search bar. The main content area includes sections for '高可用性 PolarDB-X Operator 介绍' (High Availability PolarDB-X Operator Introduction), '操作手册和 CPU 架构' (Operation Manual and CPU Architecture), and '多机房部署' (Multi-region Deployment). A detailed diagram shows a multi-region deployment setup with multiple PolarDB-X clusters (labeled 'PolarDB-X Operator') connected via 'PolarDB-X API' to a central 'PolarDB-X Operator' instance. The 'CPU 架构' section provides specific details about the architecture, including CPU usage statistics for different components.

PolarDB-X 官方技术文档

• 数据库基础知识文档：

This screenshot displays two pages from a database handbook. The left page is for MySQL, titled '7.2 表表达式' (Table Expressions), which covers topics like 'CREATE TABLE AS SELECT' and 'SELECT语句' (SELECT Statement). The right page is for PostgreSQL, titled '7.2.7. SELECT语法' (SELECT Syntax), which covers 'SELECT语句' (SELECT Statement) and '13.2.7.1. JOIN语法' (JOIN Syntax). Both pages include code snippets and explanatory text.

MySQL 中文文档 & PostgreSQL 技术手册

• 数据库学习材料：

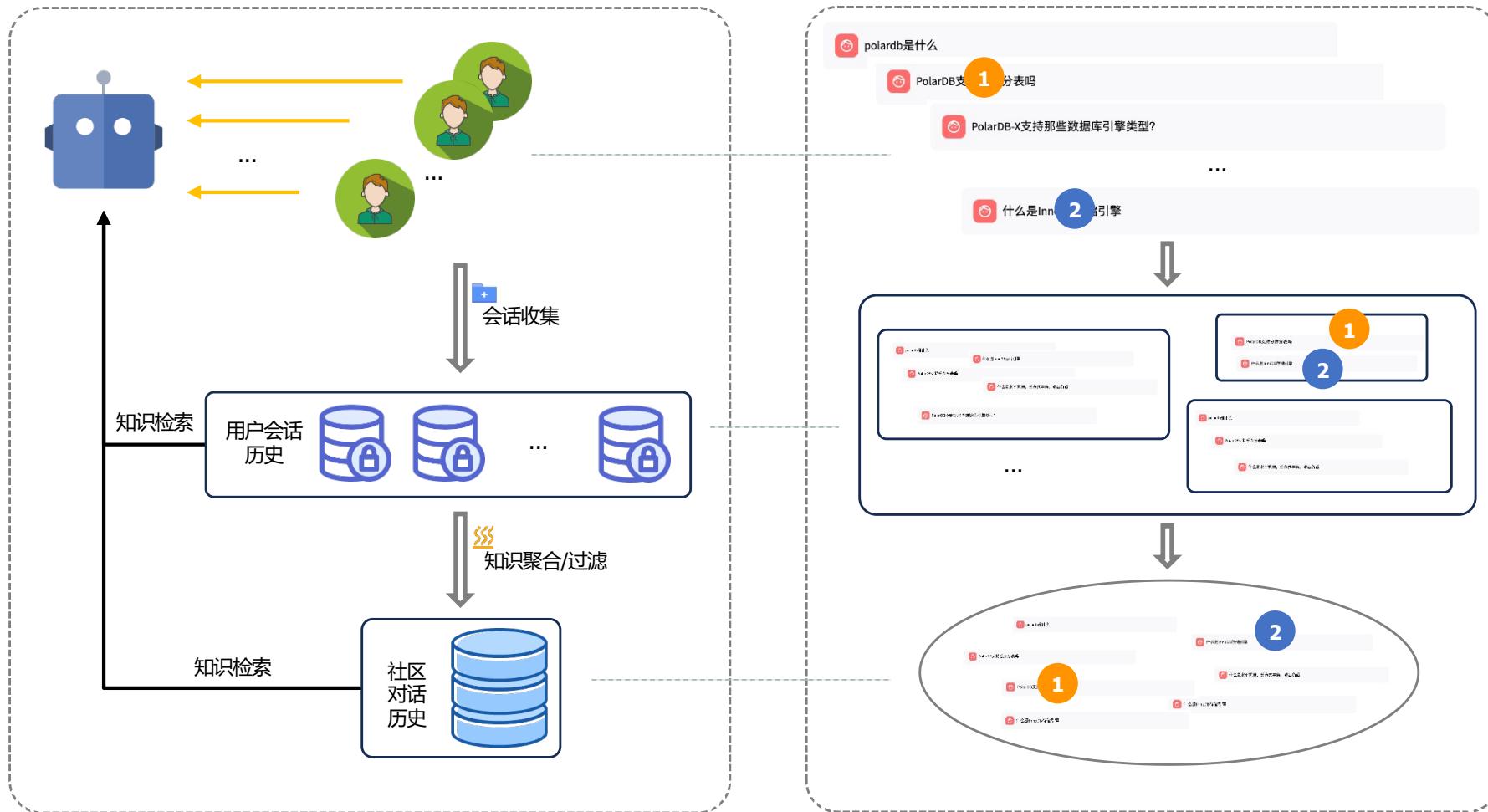
This screenshot shows a page from the PostgreSQL documentation titled 'PG 额外类型' (PostgreSQL Extra Types). It discusses various data types, including '内部类型' (Internal Types) and '扩展类型' (External Types). A diagram illustrates the relationship between internal types and external types. The page also includes a table showing the mapping between MySQL and PostgreSQL data types.

This screenshot shows a collection of PostgreSQL-related resources. On the left is a screenshot of a PostgreSQL book titled 'Creating an audit trail'. In the center is a screenshot of a PostgreSQL training slide titled 'Manual debugging with RAISE NOTICE'. The right side contains several code snippets and examples related to PostgreSQL functions and triggers, such as 'CREATE OR REPLACE FUNCTION running_queries(rows int, qid int) RETURNS SETOF running_queries AS \$\$.

PostgreSQL 书籍 & 培训 PPT

- 高质量对话数据收集及辅助问答

- 1 PolarDB 支持分库分表吗
- 2 什么是 InnoDB 存储引擎



- 问题补全
 - 将多轮对话中用户省略的问题内容进行补全
- 问题聚类
 - 通过语义向量对问题进行聚类
 - 同一个问题在中文中有许多种不同的问法
 - 通过聚类的方法将同个问题的不同问法写成同一个
- 问题过滤
 - 保留用户高频问题
 - 过滤掉低质量问题
- 答案生成
 - 生成可信的答案来降低幻觉现象

功能—— PolarDB 社区问答助手



华东师范大学数据科学与工程学院
School of Data Science and Engineering at ECNU

- 基于 FastAPI – WebUI 进行网页问答对话
- 依托钉钉自定义机器人实现钉钉群聊助手

The screenshot shows a web-based AI conversational interface. On the left, there are configuration options like '对话' (Conversation), '知识库管理' (Knowledge庫 Management), '当前会话' (Current Session: default), '请选择对话模式' (Select Conversation Mode: 默认), '选择 LLM 模型' (Select LLM Model: qwen-api (Running)), '请选择 Prompt 模板' (Select Prompt Template: default), 'Temperature' slider (0.00 to 1.00, set to 0.70), '历史对话轮数' (Number of Previous Conversations: 3), and '知识库配置' (Knowledge Base Configuration). The main area displays a conversation about PolarDB's architecture, mentioning its distributed design, cost efficiency, and performance improvements over MySQL. It also lists several questions related to PolarDB's architecture and performance.

The screenshot shows a DingTalk-based AI bot interface. A message from the bot asks how to manage and optimize storage engines in PolarDB for high concurrency and large-scale data scenarios. Below the message, there are several numbered tips: 1. 分片 (Sharding) - 将数据分片，将数据分布到不同节点上执行的操作，提高系统的性能。 2. 读写分离 (Read-Write Separation) - 持续操作和写操作以有效地分散读写的性能。 3. 分布式事务 (Distributed Transactions) - PolarDB支持分布式事务，提高系统的性能。 4. 缓存 (Cache) - 热点数据缓存到内存中，减少访问次数，提高读取性能。 5. 优化查询 (Query Optimization) - 多种查询优化技术，生成合适的执行计划。 6. 硬件优化 (Hardware Optimization) - 持多种硬件资源，根据硬件配置，从而提升性能。

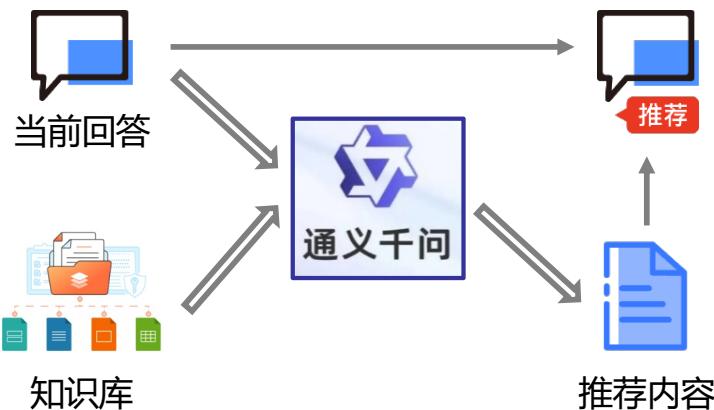
Below the tips, there is a code snippet for an UPDATE query:

```
UPDATE t1
SET c1 = (
    SELECT c2
    FROM t2
    WHERE t2.c1 = t1.c1
) AS t ON t.c1 = t1.c1
SET t1.c1 = t.c2
LIMIT 10;
```

There is also a note about potential side effects of this query. At the bottom, there are buttons for '内容不错' (Content is good) and '帮助不大' (Help is not much).

功能——知识推荐

- 根据当前回答推荐关键词与问题



polardb 是什么

阿里云 PolarDB 是一款云原生的数据库服务，提供 MySQL 协议的兼容能力，是云原生数据库的代表作。

您可能对以下问题还感兴趣：

关键词：阿里云、PolarDB、云原生、数据库服务、MySQL 协议、兼容能力、代表作。

拓展问题：

- 阿里云 PolarDB 和传统数据库有什么区别？
- PolarDB 的性能如何？能否满足大规模并发的需求？
- 阿里云 PolarDB 的安全性如何保障？

✓ 知识库匹配结果

阿里云 PolarDB 和传统数据库有什么区别？

阿里云 PolarDB 是一款云原生的数据库服务，相较于传统数据库，PolarDB 有以下几个特点：

- 高可用性：PolarDB 在设计上充分考虑了可用性，采用多副本策略，即使在部分节点故障的情况下，也能保证服务的高可用性。
- 自动弹性伸缩：PolarDB 支持自动弹性伸缩，可以根据业务需求自动调整服务规模，避免因资源不足导致的服务中断。
- 数据一致：PolarDB 支持分布式事务，可以保证跨多个节点的数据一致性，避免因数据不一致导致的问题。
- 数据可恢复：PolarDB 支持跨多个节点的数据备份和恢复，可以在节点故障或数据丢失的情况下，快速恢复业务服务。
- 数据安全：PolarDB 支持多种安全机制，如数据加密、访问控制等，可以保护数据的安全性。
- 高性能：PolarDB 采用最新的数据库技术和架构，如分布式存储、分布式计算、列式存储等，可以提供高性能的数据库服务。
- 简化运维：PolarDB 提供了丰富的运维工具和接口，可以简化数据库的运维工作，提高运维效率。

综上所述，阿里云 PolarDB 在可用性、弹性伸缩、数据一致、数据可恢复、数据安全、高性能和运维等方面，相较于传统数据库，都有较大的优势。

您可能对以下问题还感兴趣：

关键词：阿里云、PolarDB、云原生、数据库服务、高可用性、自动弹性伸缩、数据一致、数据可恢复、数据安全、高性能、运维。

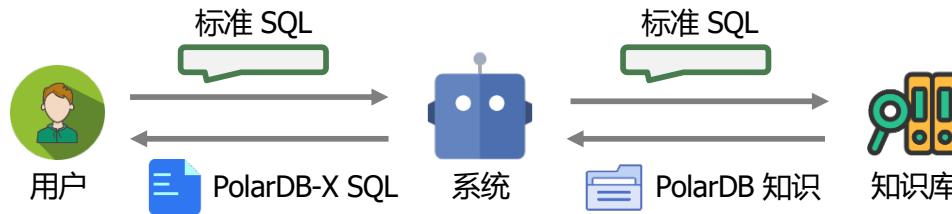
拓展问题：

- PolarDB 在哪些场景下表现最佳？
- 在使用 PolarDB 过程中，有哪些需要注意的地方？
- 阿里云 PolarDB 与传统数据库相比，有哪些不足之处？

✓ 知识库匹配结果

功能—— PolarDB-X SQL 纠错

- 分布式与单机数据库存在架构差异
- 利用 RAG 将标准 SQL 纠错为 PolarDB-X SQL



SQL语法使用限制		
SQL语法	操作	使用约束
自定义操作	自定义函数	暂不支持。
自定义操作	自定义类型	暂不支持。
自定义操作	触发器	暂不支持。
DQL	CREATE TABLE ... LIKE ...	暂不支持拆分表。
DQL	CREATE TABLE ... SELECT ...	暂不支持拆分表。
DQL	RENAME TABLE	暂不支持同时RENAME多表。
DQL	ALTER TABLE	暂不支持ALTER TABLE修改拆分字段类型。
DQL	Merge/Split/Add/Drop 二级分区表	暂不支持。
DQL	Merge/Split/Add/Drop 索引分区表	暂不支持。
DML	STRAIGHT_JOIN	暂不支持。
DML	NATURAL JOIN	暂不支持。
DML	INSERT DELAYED	暂不支持。
DML	对变量的引用和操作 (例如SET @c=1, @d=@c+1; SELECT @c, @d)	暂不支持。
DML	LOAD XML	暂不支持。
DQL	HAVING子句中包含子查询	暂不支持。
DQL	JOIN ON子句中包含子查询	暂不支持。
DQL	等号操作符的标量子查询 (The Subquery as Scalar Operand)	暂不支持。
数据库管理	SHOW WARNINGS	暂不支持LIMIT@COUNT的组合。
数据库管理	SHOW ERRORS	暂不支持LIMIT@COUNT的组合。
数据库管理	HELP	暂不支持。
运算符	:=	暂不支持。
函数	全文检索函数	暂不支持。
函数	XML函数	暂不支持。
函数	GTID语句	暂不支持。

子查询

本文介绍PolarDB-X支持的子查询类别及在PolarDB-X中使用子查询的相关限制和注意事项。

使用限制

相比原生MySQL，PolarDB-X在子查询使用上增加了如下限制：

- 不支持在HAVING子句中使用子查询。示例如下：

```
1 SELECT name, AVG( quantity )
2 FROM tb3
3 GROUP BY name
4 HAVING AVG( quantity ) > 2 * (
5   SELECT AVG( quantity )
6   FROM tb2
7 );
```

- 不支持在JOIN ON子句中使用子查询。示例如下：

```
SELECT * FROM tb1 p JOIN tb2 s on (p.id=s.id and p.quantity>All(select quantity from tb3))
```

- 等号操作符的标量子查询 (The Subquery as Scalar Operand) 不支持ROW语法。示例如下：

```
select * from tb1 where row(id, name) = (select id, name from tb2)
```

- 不支持在UPDATE SET子句中使用子查询。示例如下：

```
UPDATE t1 SET c1 = (SELECT c2 FROM t2 WHERE t1.c1 = t2.c1) LIMIT 10
```

PolarDB sql 纠错: UPDATE t1 SET c1 = (SELECT c2 FROM t2 WHERE t1.c1 = t2.c1) LIMIT 10

PolarDB不支持在UPDATE SET子句中使用子查询，因此需要将子查询改写为JOIN的形式来达到相同的效果。具体的SQL改写方法如下所示：UPDATE t1 JOIN (SELECT t2.c1,t2.c2 FROM t2 LIMIT 10) AS new_t2 ON t1.c1 = new_t2.c1 SET t1.c1 = new_t2.c2;注意事项PolarDB-X中部分子查询仅能以APPLY的方式执行，查询效率低下。在实际使用中请尽量避免如下例子中的低效SQL：SELECT * FROM tb1 WHERE row(id, name) = (select id, name from tb2);

您可能对以下问题还感兴趣：

关键词：子查询、UPDATE、LIMIT、JOIN。

您可能对以下问题还感兴趣：

关键词：子查询、UPDATE、LIMIT、JOIN。相关拓展问题：

- 如何将一个使用子查询的UPDATE语句改写为使用JOIN的语句？
- 在哪些情况下应该避免使用子查询？
- 如何使用LIMIT子句来限制UPDATE语句中受影响的行数？

✓ 对话历史匹配结果

出处 [1] dev.mysql.com/doc/refman/5.7/en/mysql-5.7/mysql-5.7.md

mysql> SELECT * FROM t1 : WHERE s1 IN (SELECT s2 FROM t2 ORDER BY s1 LIMIT 1); ERROR 1235 (42000): This version of MySQL doesn't yet support : 'LIMIT & IN/ALL/ANY/SOME subquery'

出处 [2] dev.mysql.com/doc/refman/5.7/en/mysql-5.7/mysql-5.7.md

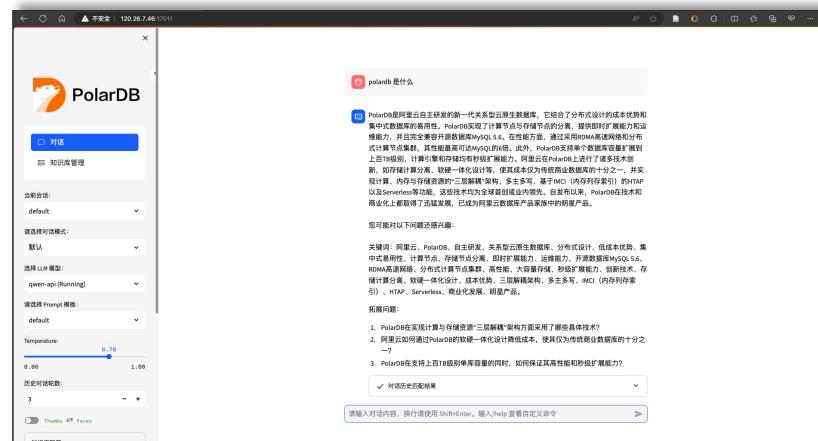
mysql> SELECT * FROM t1 : WHERE s1 IN (SELECT s2 FROM t2 ORDER BY s1 LIMIT 1); ERROR 1235 (42000): This version of MySQL doesn't yet support : 'LIMIT & IN/ALL/ANY/SOME subquery'

出处 [3] [PolarDB/PolarDB-X.txt](#)

select * from tb1 where row(id, name) = (select id, name from tb2) 不支持在UPDATE SET子句中使用子查询，示例如下：UPDATE t1 SET c1 = (SELECT c2 FROM t2 WHERE t1.c1 = t2.c1) LIMIT 10 注意事项PolarDB-X中部分子查询仅能以APPLY的方式执行，查询效率低下。在实际使用中请尽量避免如下例子中的低效SQL：

项目落地

- 本产品将在PolarDB社区平台落地，该平台涵盖的**用户类型丰富**，该平台具有**庞大的数据规模和坚实的用户基础**。
- PolarDB 社区问答系统在网页端与钉钉群同步上线。



@测试 在PolarDB中，如何有效地管理并优化存储引擎的读取性能，特别是在高并发和大规模数据场景下？

测试 机器人

在PolarDB中，如何有效地管理并优化存储引擎的读取性能，特别是在高并发和大规模数据场景下？

PolarDB在高并发和大规模数据场景下，可以通过多种手段来管理并优化存储引擎的读取性能：

- 分片 (Sharding)：PolarDB可以支持数据的分片存储，将数据分布在多个物理节点上，从而提高并发读取的性能。分片可以根据数据的分布规律或者自定义的规则进行。
- 读写分离 (Read-Write Separation)：PolarDB支持读操作和写操作在不同的节点上执行，这样可以有效地分散读取和写入操作的压力，提高系统的性能。
- 分布式事务 (Distributed Transactions)：PolarDB支持分布式事务处理，可以确保在多个节点上执行的操作能够一致地提交或者回滚，从而提高系统的性能和可靠性。
- 缓存 (Cache)：PolarDB支持缓存机制，可以将热点数据缓存到内存中，从而减少对数据库的访问次数，提高读取性能。
- 优化查询 (Query Optimization)：PolarDB支持多种查询优化技术，可以根据查询的特点选择合适的执行计划，从而提高查询性能。
- 硬件优化 (Hardware Optimization)：PolarDB支持多种硬件资源，可以根据实际需求选择合适的硬件配置，从而提高系统的性能。

您可能对以下问题还感兴趣：关键词：分片、读写分离、分布式事务

拓展问题：

1. PolarDB如何实现数据的分片存储？
2. PolarDB如何支持读写分离？这对系统性能有何影响？
3. PolarDB如何保证分布式事务的一致性？

内容不错

帮助不大

PolarDB sql纠错: UPDATE t1SETc 1=(SELECT c2FROMt2WHEREt 1.c 1=t 2.c 1) LIMIT 10 @测试

测试 机器人

PolarDB sql纠错: UPDATE t1SETc 1=(SELECT c2FROMt2WHEREt 1.c 1=t 2.c 1) LIMIT 10

UPDATE语句中不能直接使用子查询，可以尝试使用JOIN或者多次更新的方式来操作。例如：

```
UPDATE t1
JOIN (
  SELECT c2
  FROM t2
  WHERE t2.c1 = t1.c1
) AS t ON t.c1 = t1.c1
SET t1.c1 = t.c2
LIMIT 10;
```

或者：

```
UPDATE t1
SET c1 = (
  SELECT c2
  FROM t2
  WHERE t2.c1 = t1.c1
)
LIMIT 10;
```

需要注意的是，这种情况下可能会影响到更多的行数，因为在上述两种方式中，如果有两条t1记录与一条t2记录匹配，则这条t1记录会被更新多次。

您可能对以下问题还感兴趣：关键词：UPDATE语句、子查询、JOIN。

相关拓展问题：

1. 如何在SQL中使用JOIN进行复杂的数据查询？
2. UPDATE语句中如何使用多表更新？
3. SQL中的子查询有哪些使用限制和注意事项？

内容不错

帮助不大