**账号：qq邮箱**

密码：1593511814abc（mac) 1593511814(window10)

**使用RAGFow构建本地知识库与之间使用DS区别**

解决数据隐私泄露问题（如ds需要上传第三方)

解决文件上传限制问题：数量限制，算力限制（如ds只能上传几个）

解决使用繁琐的问题：每次都需要重新上传文件，增删改复杂，上下文限制

疑问❓：Rag技术的上下文限制问题: 指定知识库全库检索

**实现方案**

隐私保护：通过本地部署deepseek解决隐私问题

个性化知识库构建：使用RAG技术（Retrieval-Augmented-Generation, 检索增强生产）构建个人知识库。

本地部署RAG技术所需要的开源框架爱RAGFlow;

本地部署Embedding大模型（或者之间不熟自带Embending模型的RAGFlow版本(有隐私泄露风险））

**RAG和模型微调**

共同点：解决大模型对未知知识解答时出行的“幻觉”问题

区别：

微调：在已有预训练模型的基础上，结合特定任务的数据集进一步训练，让模型在该领域表现的更好（靠前复习）

RAG：在生产回答之前，通过信息检索外部知识库相关信息，增强生产过程中的信息来源，从而提升生产信息准确性（开卷考试，变考边抄）

RAG技术原理：

检索（Retrieval): 提问后，系统从外部知识库搜索相关内容

增强（Augmentation): 将搜索到的信息和用户输入内容结合，扩展模型上下文。

生成（Generation): 生成模型基于增强后的输入生成最终答案。

**Embedding模型**

embedding模型在整个问答环节的作用：

**对外部知识库文件解析**：文本等外部知识是由自然语言组成，不利于机器直接计算相似度。embedding模型就是要将自然语言转换为高维向量，然后通过向量捕获到单词或句子背后的语义信息。

**对用户提问内容解析，生成高维向量**。

**然后拿用户提问解析后的高维向量信息，去匹配外部知识库解析后的高维向量信息**：使用这个用户输入生成的这个高纬向量，去查询知识库中有无相关的文档片段。系统会利用某些相似度度量（如余弦相似度）去判断这个相似度。这一整个过程可以理解为，上传并解析了知识库之后，相当于给知识库中每一个小的文段都生成了一个指纹；然后用户输入问题之后，这个问题同样也会生成一个独一无二的指纹，接着RAGFLOW系统就会拿着用户输入的这个指纹，在指纹库也就是知识库中匹配，找到相似的指纹，然后把将检索到的这些相关文段与用户的输入结合，扩展模型的上下文，再喂给对话模型DeepSeek。

**embeding 模型：对知识库内容进行解析，转化为机器可以理解的高维向量；**

**文本数据——Embedding——向量数据**

**本地部署流程**

**下载ollama, 通过ollama将DeepSeek模型下载到本地运行；**

**下载RAGFlow源代码和Docker，通过Docker部署RAGFlow**

**在RAGFLow中构建个人知识库并实现基于个人知识库的对话问答**

**非本地部署方式**

下载RAGFlow源代码和docker，通过docker本地部署RAGFlow（RAGFlow目前没有官方的网页版）；

在RAGFlow中配置任意的Chat模型和Embedding模型（你需要到这些模型对应的官网去付费申请apiKey）

**windows10 部署流程：**

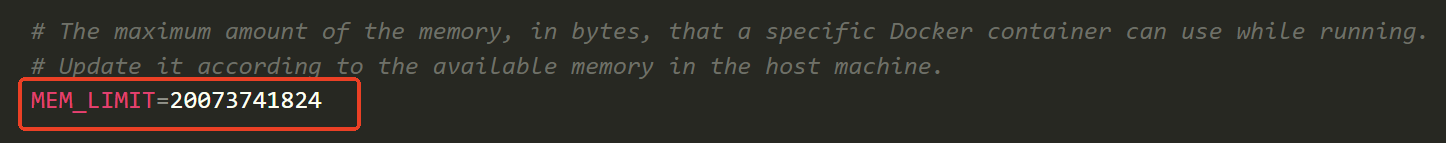
下载安装docker desktop

地址：<https://www.docker.com/>

下载RAGFlow 源码：

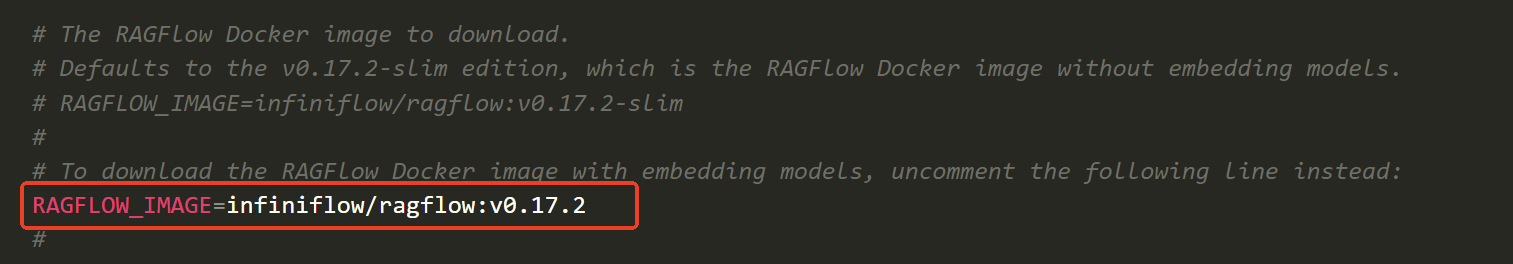
地址：<https://github.com/infiniflow/ragflow>

提高内存限制：



选择全量版本（包含embedding模型）

**注：选择默认镜像，需要开启梯子，docker也保持默认配置，这个方式最稳定**



拉取镜像并启动RAGFlow （开启梯子环境）

cd ragflow/docker

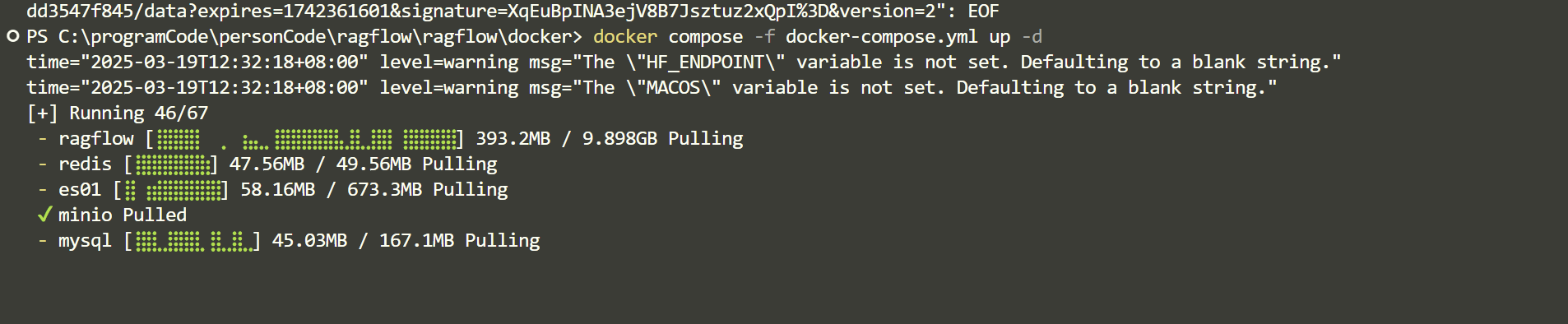
docker compose -f docker-compose.yml up -d

报错信息：可能梯子或者未使用国内镜像的原因（可重试或者切换国内华为或阿里镜像）

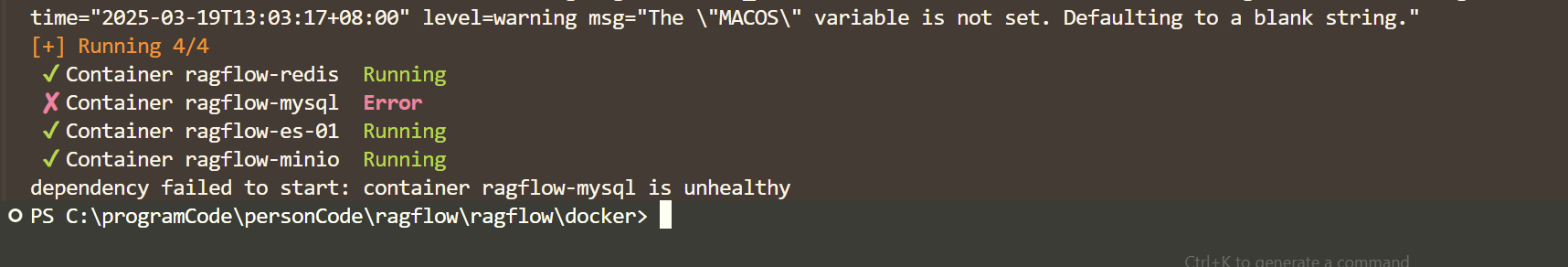


重新执行命名成功：

注意：看到9GB ragFlow模型就是全量版本，如果是2GB版本则是轻量版；



报错信息：mysql安装失败



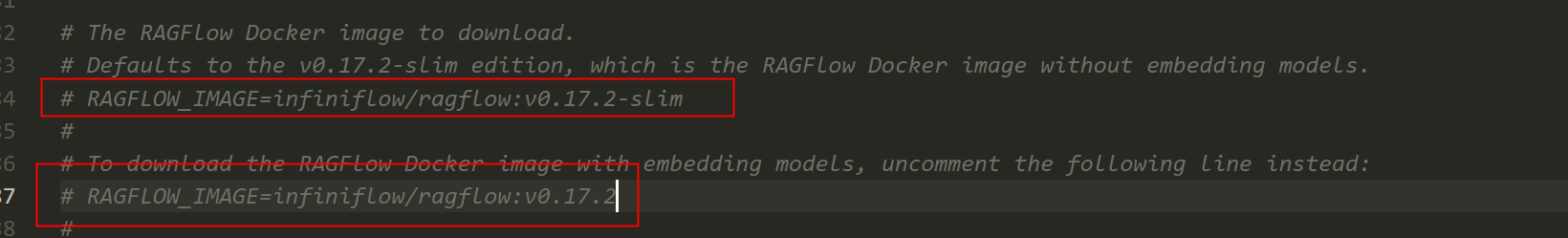
使用curosr claude3.7 thinking 解决 agent模式：



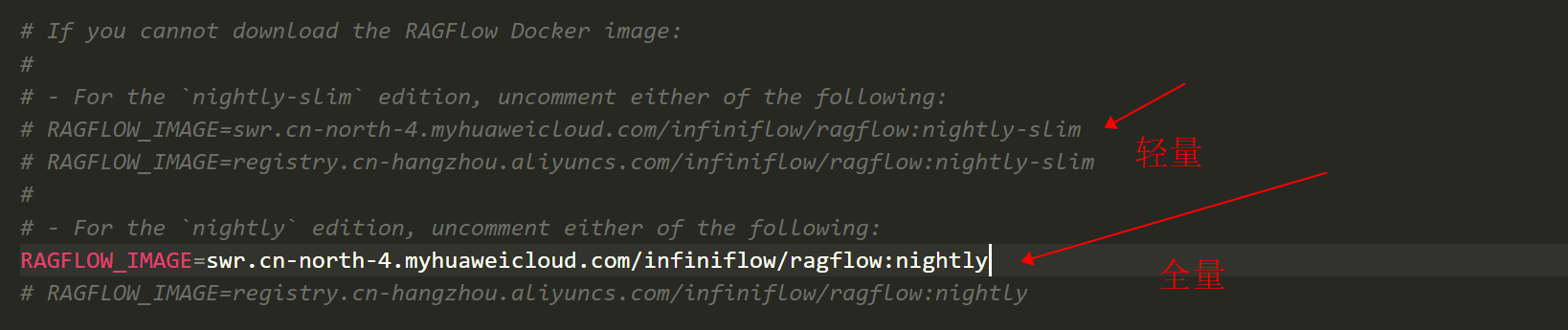
切换国内镜像方式：

注意：如果切换国内镜像，还需要给docker配置国内镜像地址，否则拉取会有各种链接失败；

打开docker/.env文件，先注释掉默认镜像：



打开阿里货华为镜像：nightly-slim是轻量版，nightly是全量版



启动通过访问本地服务80端口访问：

http://localhost:80

http://localhost/knowledge

**使用流程：**

**设置LLM模型：使用本地启动的模型，或者使用三方api**

**创建知识库-上传文件-文件解析向量化**

**创建ai智能体-配置知识库-配置ai智能体模型**

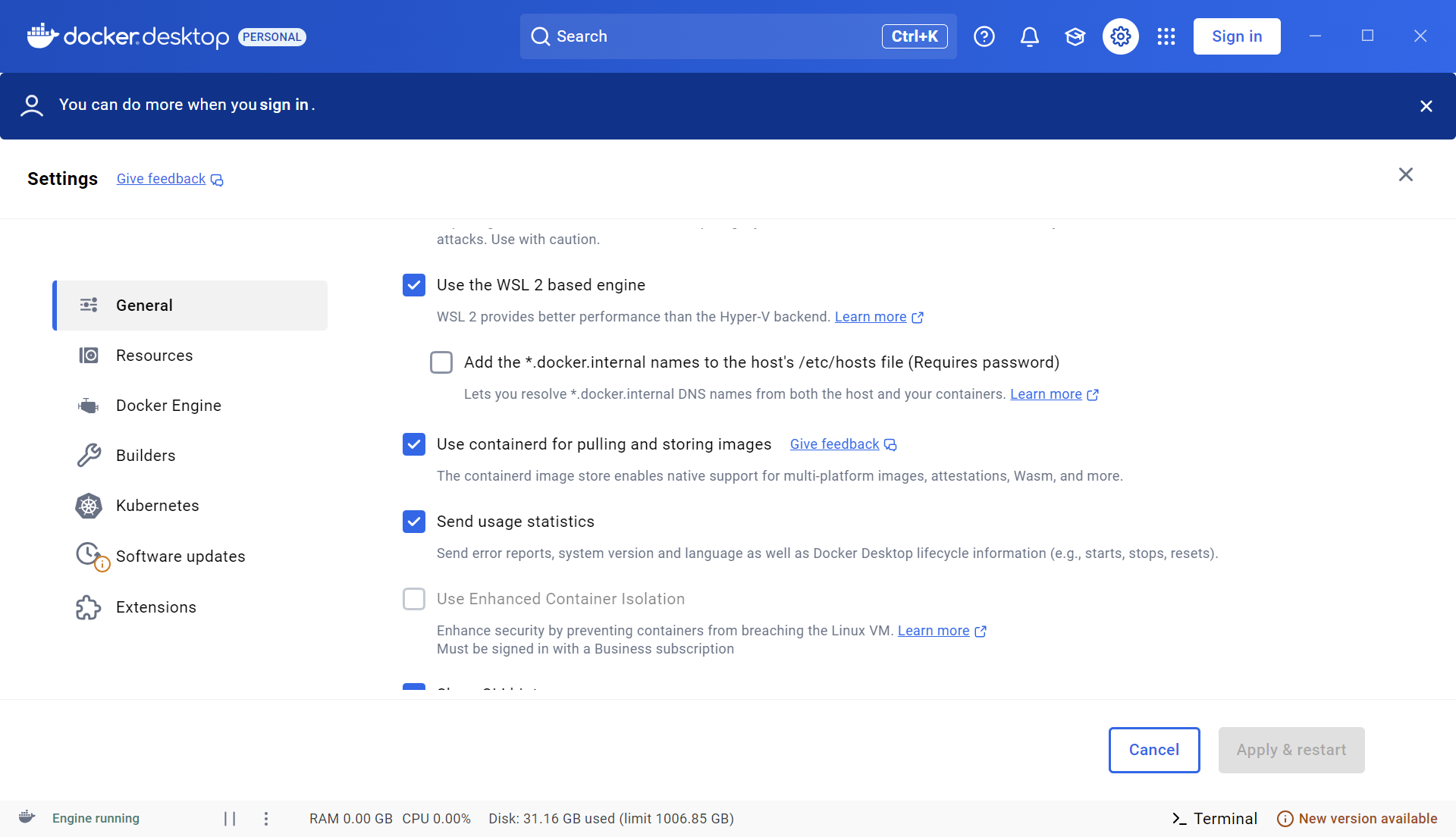
**注意点：promot提示词需要优化，加知识库检索不到答案，比如加上提示词：“如果知识库未找到相关内容，以你自己的经验继续解答”；**

**测试效果**

**docker配置**

docker最稳定的模式是使用 wsl模式，无效手动配置内存限制，数据库、镜像文件目录等；

取消“ Use the WSL 2 based engine” 勾选，重启以后可手动配置内存限制，数据库、镜像文件目录等，但实测会引起各种报错，需要对docker使用比较熟悉再尝试这种方式。



**聊天助手测试**

deepseek r1搜索速度太慢，等待时间1分钟+，输出效果最详细；

qianwen 32B最快，但是不稳定，有时候输出有报错

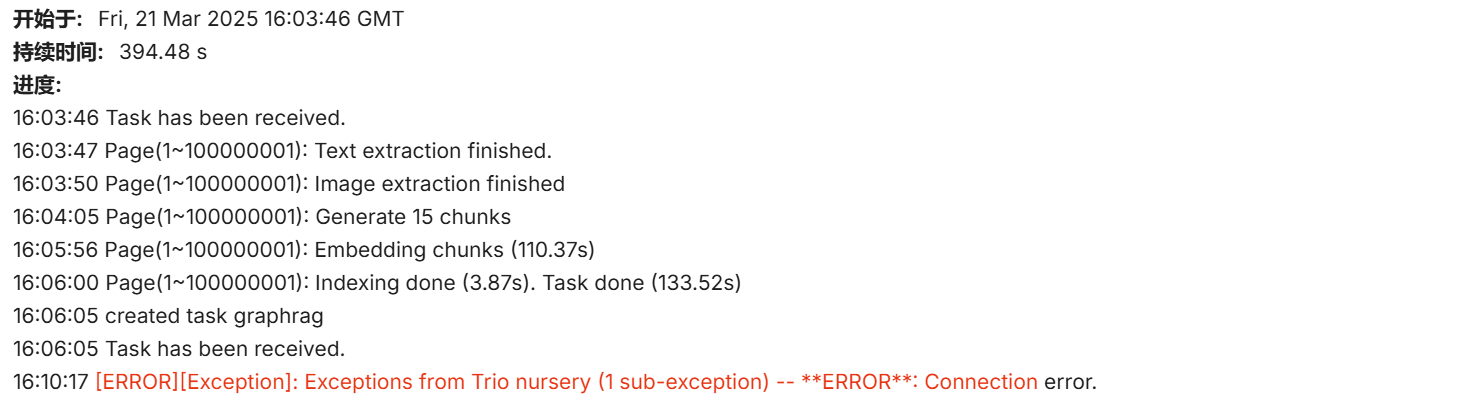
deepseek v3输出速度适中，需要等待10秒+，输出能力没有qianwen详细，但是发挥最稳定

提示词优化只能加{knowledge}不能加其他字段

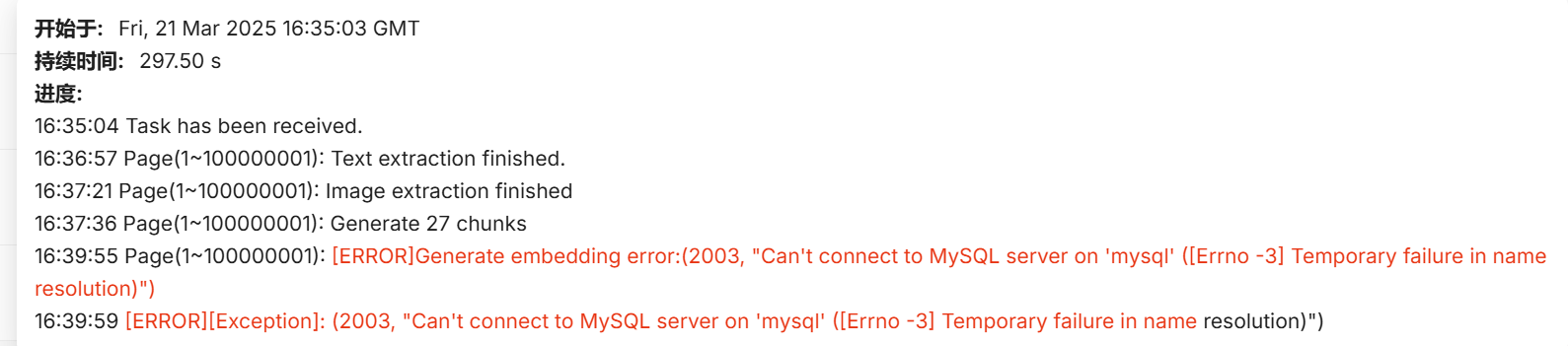
**知识库和解析测试**

解析文件过程中，配置开启知识图谱后，会大量消耗deepseek r1；原因，联调模型配置了deepseek，解析文件过程也会调用聊天模型。

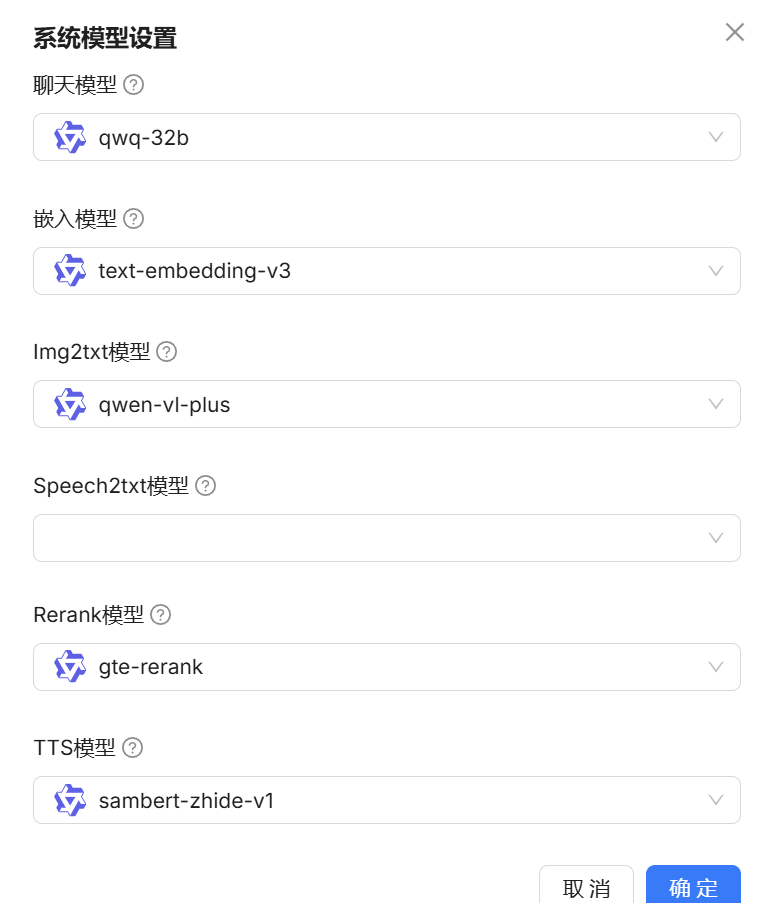
解析文件，对ppt文件或大文件，会解析报错，但观察数据内容都解析成功了，并且不影响查询使用，只是解析状态会标记失败。应该是大文件导致的。



系统默认模型切换到qianwen32B后解析速度加快，但有新的报错



最终使用以下配置后，文件解析速度是最快的



使用qianwen模型配置后解析文件出现新的报错：

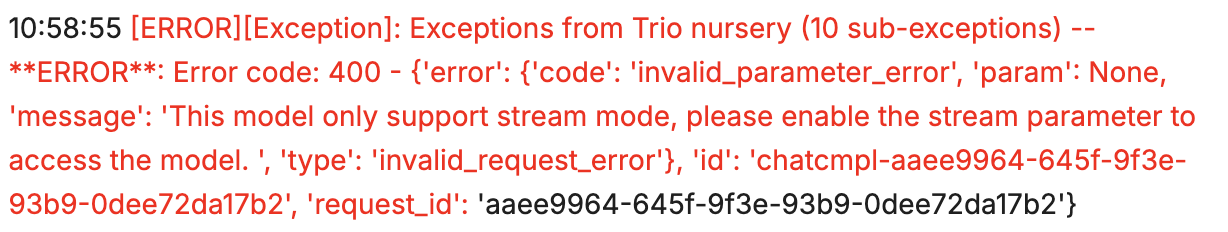
应该是模型api配置问题，但界面中没有api参数配置，尝试定位代码没有成功；（社区查询结果：不支持qianwen模型stream模式）

解决方法：

联调模型改为deepseek v3后解决；（缺点：解析速度慢了很多,可能是因为正常进入了知识图谱生成环节）

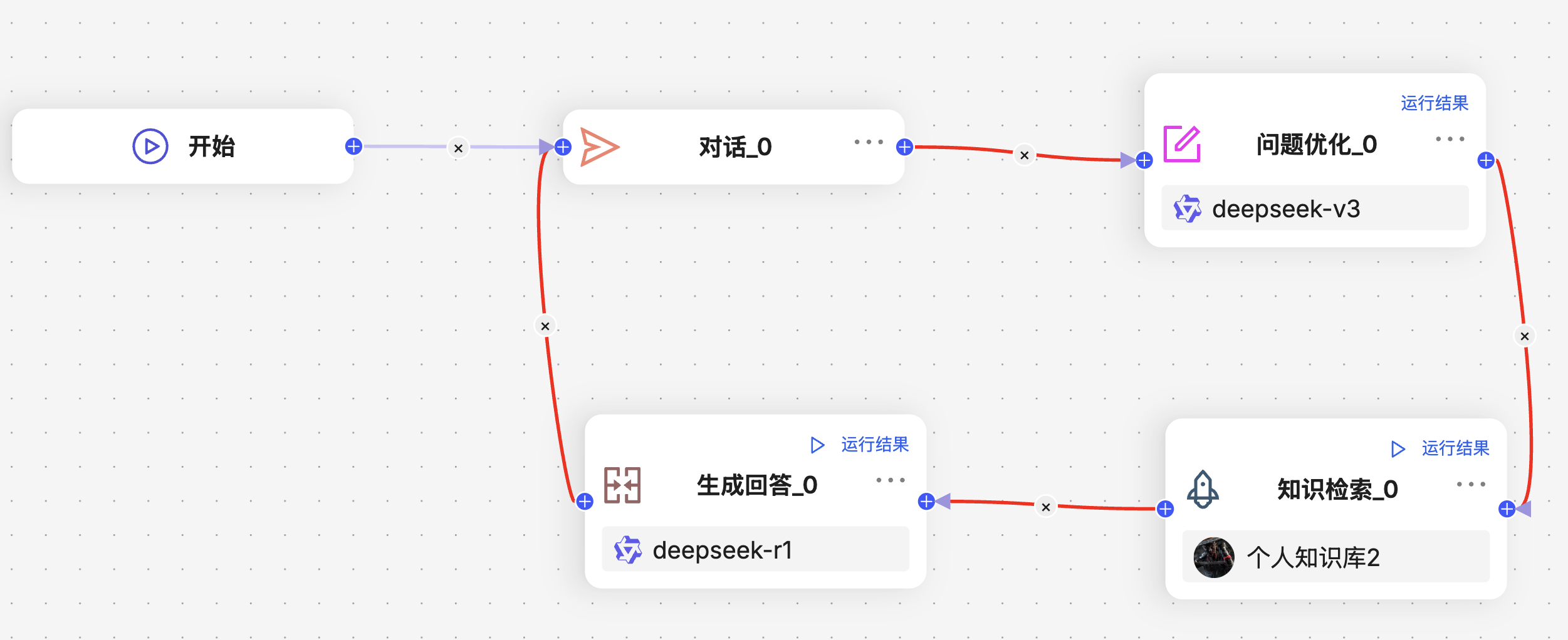
v3上下文长度限制：大文件需要分段上传解析

qinawen模型目标不支持流输出，使用qianwen chat模型解析必须关闭知识图谱；



**agent模式测试**

生成回答系统提示词配置：”请总结以下对以下段落做分析和优化，并分析问题中有哪些要求继续完善结果；“（会更具检索后的内容继续自主发挥）



**相关专业词汇解析**

**知识库-检索测试**

**检索测试**

完成召回测试：确保你的配置可以从数据库召回正确的文本块。如果你调整了这里的默认设置，比如关键词相似度权重，请注意这里的改动不会被自动保存。请务必在聊天助手设置或者召回算子设置处同步更新相关设置。

**相似度阈值**

我们使用混合相似度得分来评估两行文本之间的距离。 它是加权关键词相似度和向量余弦相似度。 如果查询和块之间的相似度小于此阈值，则该块将被过滤掉。（可调节）

**关键字相似度权重**

我们使用混合相似性评分来评估两行文本之间的距离。它是加权关键字相似性和矢量余弦相似性或rerank得分（0〜1）。两个权重的总和为1.0。（可调节）

**使用知识图谱**

它将检索相关实体、关系和社区报告的描述，这将增强多跳和复杂问题的推理。（可开启）

**知识库-配置**

**文档解析器**

使用视觉模型进行 PDF 布局分析，以更好地识别文档结构，找到标题、文本块、图像和表格的位置。 如果选择 Naive 选项，则只能获取 PDF 的纯文本。请注意该功能只适用于 PDF 文档，对其他文档不生效。（默认选择deepDoc)

**嵌入模型**

用于嵌入块的嵌入模型。 一旦知识库有了块，它就无法更改。 如果你想改变它，你需要删除所有的块。（embedding模型：用于向量化解析）

**切片方法**

"General" 分块方法说明（默认值：Genneral)

支持的文件格式为**DOCX、EXCEL、PPT、IMAGE、PDF、TXT、MD、JSON、EML、HTML**。

此方法将简单的方法应用于块文件：

系统将使用视觉检测模型将连续文本分割成多个片段。

接下来，这些连续的片段被合并成Token数不超过“Token数”的块。

**块Token数**

它大致确定了一个块的Token数量。（默认值500左右）

**分段标识符**

支持多字符作为分隔符，多字符分隔符用`包裹。如配置成这样： `##`;那么就会用换行，两个#以及分号先对文本进行分割，然后按照“ token number”大小进行拼装。（默认值：\n!?;。；！？）（个人理解：防止切片截断问题）

**页面排名**

这用于提高相关性得分。所有检索到的块的相关性得分将加上此数字。 当您想首先搜索给定的知识库时，请设置比其他知识库更高的 pagerank 得分。（未知）

**自动关键词**

在查询此类关键词时，为每个块提取 N 个关键词以提高其排名得分。在“系统模型设置”中设置的 LLM 将消耗额外的 token。您可以在块列表中查看结果。（未知）

**自动问题**

在查询此类问题时，为每个块提取 N 个问题以提高其排名得分。在“系统模型设置”中设置的 LLM 将消耗额外的 token。您可以在块列表中查看结果。如果发生错误，此功能不会破坏整个分块过程，除了将空结果添加到原始块。（未知）

**表格转HTML**

开启后电子表格会被解析为 HTML 表格，每张表格最多 256 行，否则会按行解析为键值对。（未知）

**标签集**

能够为数据集的文本块批量添加更多领域知识，从而显著提高检索准确性。该功能还能提升大量文本块的操作效率。（待测试）

**知识图谱-实体归一化**

解析过程会将具有相同含义的实体合并在一起，从而使知识图谱更简洁、更准确。应合并以下实体：特朗普总统、唐纳德·特朗普、唐纳德·J·特朗普、唐纳德·约翰·特朗普（建议勾选）

**社区报告生成**

区块被聚集成层次化的社区，实体和关系通过更高抽象层次将每个部分连接起来。然后，我们使用 LLM 生成每个社区的摘要，称为社区报告。（建议勾选）

**聊天-助理设置**

**空回复**

如果在知识库中没有检索到用户的问题，它将使用它作为答案。 如果您希望 LLM 在未检索到任何内容时提出自己的意见，请将此留空。(配置提示词使用）

**关键词分析**

应用 LLM 分析用户的问题，提取在相关性计算中要强调的关键词。(先使用大模型优化用户的提问，再去检索和生成内容）（建议开启）

**Tavily API Key**

如果 API 密钥设置正确，它将利用 Tavily 进行网络搜索作为知识库的补充。

**聊天-提示引擎**

**系统提示词**

当LLM回答问题时，你需要LLM遵循的说明，比如角色设计、答案长度和答案语言等。（实测只能加{knowledge}标签，追加未检索到内容使用模型能力输出内容可生效，其他调试效果不明显，需要后续尝试调优测试）

**相似度阈值**

我们使用混合相似度得分来评估两行文本之间的距离。 它是加权关键词相似度和向量余弦相似度。 如果查询和块之间的相似度小于此阈值，则该块将被过滤掉。（可调节）

**关键字相似度权重**

我们使用混合相似性评分来评估两行文本之间的距离。它是加权关键字相似性和矢量余弦相似性或rerank得分（0〜1）。两个权重的总和为1.0。（可调节）

**Top N**

并非所有相似度得分高于“相似度阈值”的块都会被提供给大语言模型。 LLM 只能看到这些“Top N”块。（未知）

**多轮对话优化**

在多轮对话的中，对去知识库查询的问题进行优化。会调用大模型额外消耗token。（可开启）

**使用知识图谱**

它将检索相关实体、关系和社区报告的描述，这将增强多跳和复杂问题的推理。（可开启）

**推理**

它将像Deepseek-R1 / OpenAI o1一样触发推理过程。将代理搜索过程集成到推理工作流中，允许模型本身在遇到不确定信息时动态地检索外部知识。

**Rerank模型**

非必选项：若不选择 rerank 模型，系统将默认采用关键词相似度与向量余弦相似度相结合的混合查询方式；如果设置了 rerank 模型，则混合查询中的向量相似度部分将被 rerank 打分替代。请注意：采用 rerank 模型会非常耗时。

**变量**

如果您使用对话 API，变量可能会帮助您使用不同的策略与客户聊天。 这些变量用于填写提示中的“系统提示词”部分，以便给LLM一个提示。 “知识”是一个非常特殊的变量，它将用检索到的块填充。 “系统提示词”中的所有变量都应该用大括号括起来。（默认knowledge，可新增词）

**聊天-模型设置**

**自由度**

“精确”意味着大语言模型会保守并谨慎地回答你的问题。 “即兴发挥”意味着你希望大语言模型能够自由地畅所欲言。 “平衡”是谨慎与自由之间的平衡。（默认精准）（其他更细粒度参数温度、处罚等参数需要进一步了解再做使用）