**Langchain-Chatchat 源代码部署/开发部署指南**

**0. 拉取项目代码**

git clone https://github.com/chatchat-space/Langchain-Chatchat.git

**1. 初始化开发环境**

Langchain-Chatchat 自 0.3.0 版本起，为方便支持用户使用 pip 方式安装部署，以及为避免环境中依赖包版本冲突等问题， 在源代码/开发部署中不再继续使用 requirements.txt 管理项目依赖库，转为使用 Poetry 进行环境管理。安装前自行切换到清华源！！！

**1.1 安装 Poetry**

在安装 Poetry 之前，使用 Conda，请创建并激活一个新的 Conda 环境，例如使用 conda create -n chatchat python=3.10 创建一个新的 Conda 环境。

pip install poetry

在安装Poetry之后， 使用如下命令使 Poetry 使用 virtualenv python environment (poetry config virtualenvs.prefer-active-python true)

**1.2 安装源代码/开发部署所需依赖库**

安装过程成会碰到需要CPython情况，请先安装CPython

pip install cpython

进入主项目目录，并安装 Langchain-Chatchat 依赖

cd Langchain-Chatchat/libs/chatchat-server/  
poetry source add --priority=default mirrors https://pypi.tuna.tsinghua.edu.cn/simple/  
poetry install --with lint,test -E xinference

Poetry install 后会在你的虚拟环境中 site-packages 路径下生成一个 chatchat-<version>.dist-info 文件夹带有 direct\_url.json 文件，这个文件指向你的开发环境

**1.3 更新开发部署环境依赖库**

当开发环境中所需的依赖库发生变化时，一般按照更新主项目目录(Langchain-Chatchat/libs/chatchat-server/)下的 pyproject.toml 再进行 poetry update 的顺序执行。

**1.4 将更新后的代码打包测试**

如果需要对开发环境中代码打包成 Python 库并进行测试，可在主项目目录执行以下命令：

poetry build

命令执行完成后，在主项目目录下会新增 dist 路径，其中存储了打包后的 Python 库。

**2. 设置源代码根目录**

如果您在开发时所使用的 IDE 需要指定项目源代码根目录，请将主项目目录(Langchain-Chatchat/libs/chatchat-server/)设置为源代码根目录。

执行以下命令之前，请先设置当前目录和项目数据目录：

cd Langchain-Chatchat/libs/chatchat-server/chatchat  
export CHATCHAT\_ROOT=/parth/to/chatchat\_data

**3. 关于 chatchat 配置项**

在执行一下命令前需要先安装[Xinference](https://inference.readthedocs.io/zh-cn/latest/index.html)，在安装前需要先安装Microsoft C++ 生成工具

下载地址：

https://visualstudio.microsoft.com/zh-hans/visual-cpp-build-tools/

安装时只需勾选一项就行



再执行：

pip install "xinference[transformers]"

如需安装"xinference[all]"，需要在linux下配环境，因为pynini没有win版本，安装完成后需要设置环境变量，避免模型下载在C盘，XINFERENCE\_HOME=F:\XinferenceCache，XINFERENCE\_MODEL\_SRC=modelscope，设置完后重启电脑

安装好之后执行

# 1.运行 Xinference 框架，默认9997端口  
xinference-local --host 127.0.0.1 --port 9997  
# 此时可以通过访问 http://127.0.0.1:9997/ 进入 Xinference ui界面  
  
# 2. 自行下载 LLM 模型和 Embedding 模型（HuggingFace不通可以去ModelScope）  
# 如果是自定义的模型需要注册：xinference register --model-type <model\_type> --file model.json --persist  
  
# 3.加载 LLM 模型，这里使用了自定义的glm4-chat模型  
xinference launch --model-name custom-glm4-chat --model-format pytorch --model-engine Transformers  
  
# 4.加载 Embedding 模型，这里使用了自定义的bge-large-zh-v1.5模型  
xinference launch --model-name custom-bge-large-zh-v1.5 --model-type embedding

从 0.3.1 版本开始，所有配置项改为 yaml 文件，具体参考 [Settings](https://github.com/chatchat-space/Langchain-Chatchat/blob/master/docs/contributing/settings.md)。

执行以下命令初始化项目配置文件和数据目录：

python cli.py init

**4. 初始化知识库**

Warning

这个命令会清空数据库、删除已有的配置文件，如果您有重要数据，请备份。

python cli.py kb --recreate-vs

如需使用其它 Embedding 模型，或者重建特定的知识库，请查看 python cli.py kb --help 了解更多的参数。

**5. 启动服务**

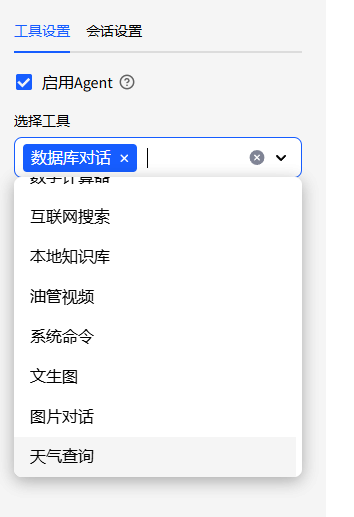
python cli.py start -a

如需调用 API，请参考 [API 使用说明](https://github.com/chatchat-space/Langchain-Chatchat/blob/master/docs/contributing/api.md)

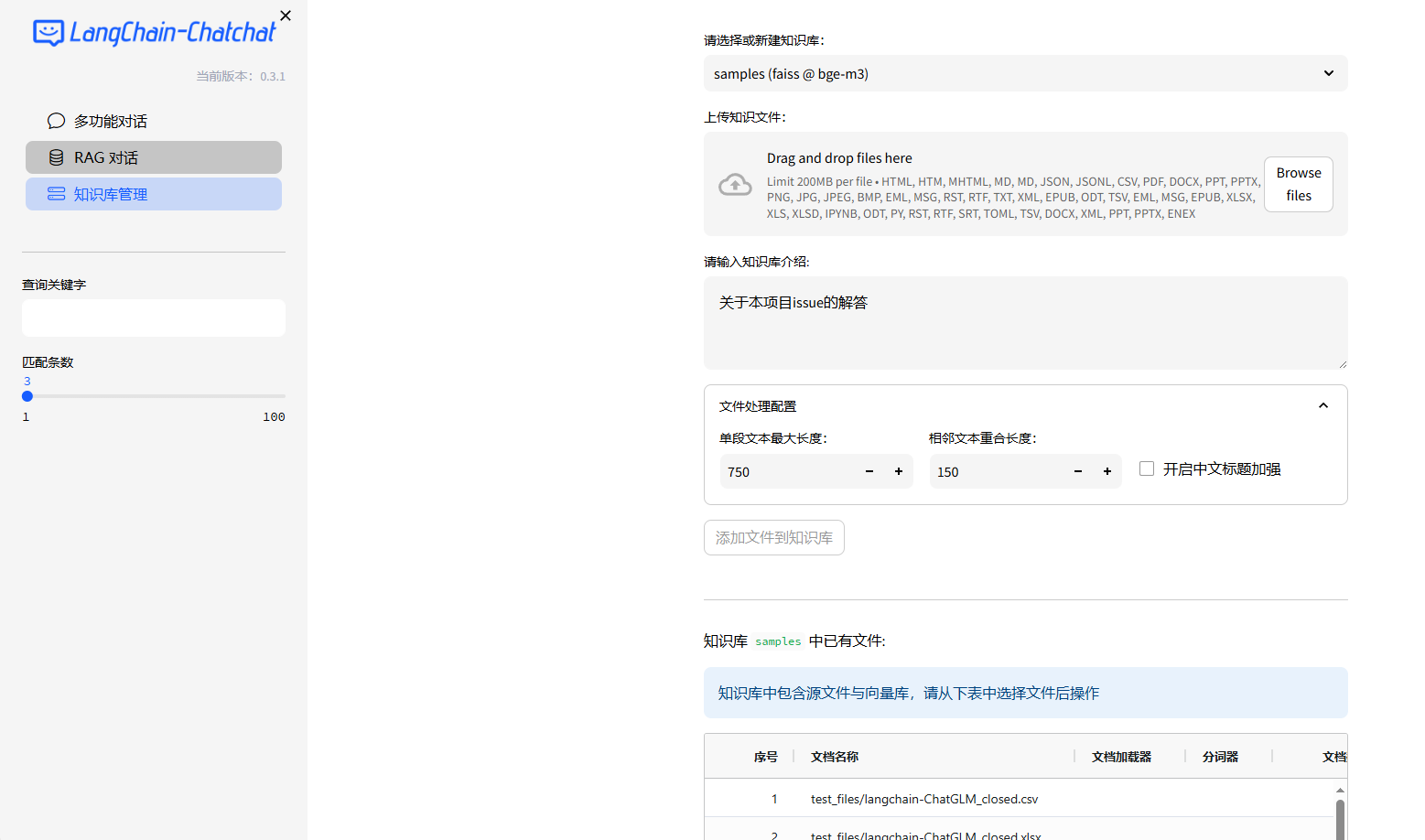
**6. 功能展示**



选择启用agent，可以实现各种功能：



通过不同的工具实现不同的功能，还可以配置知识库：



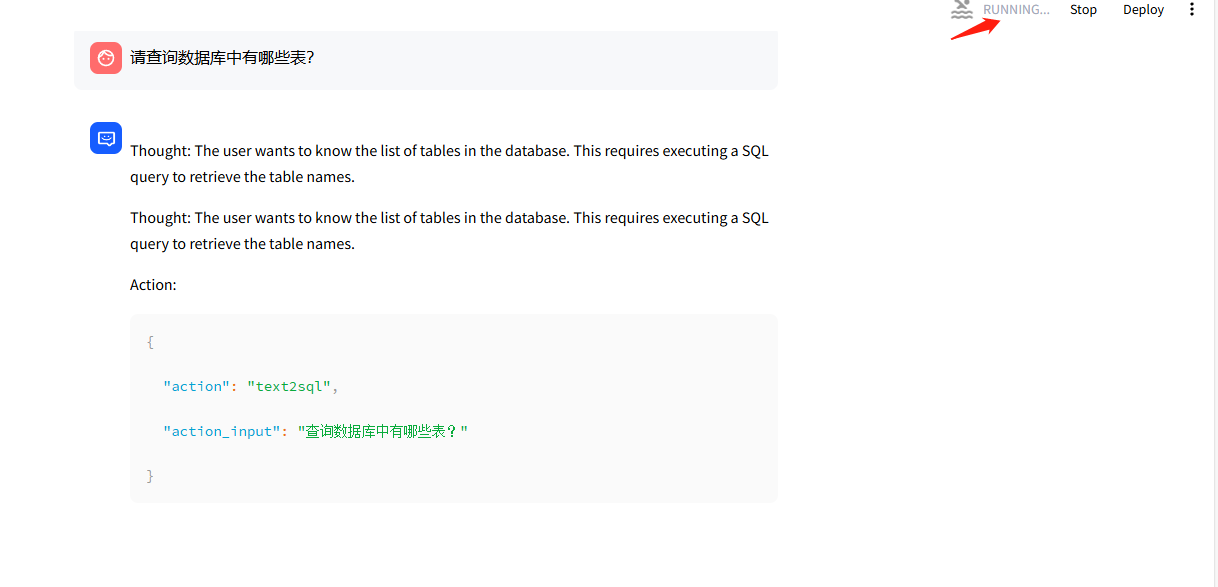
**7. 验证SQL指令功能**

### 1.配置文件修改

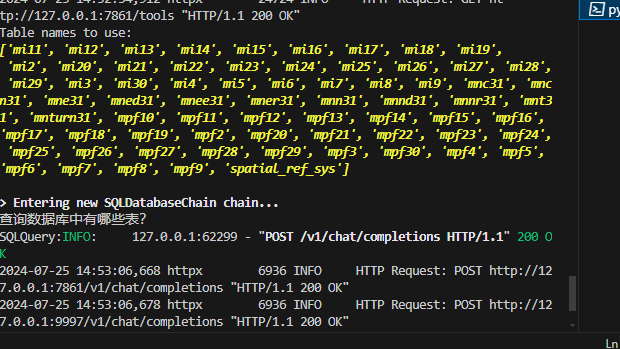


### 2.功能测试

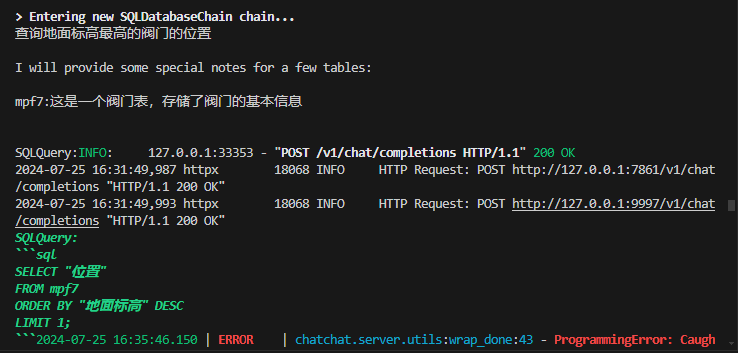
#### 1.bug的定位与修复



模型能够正确生成SQL语句，但该框架存在bug,对于SQL语句的切片不正确，导致无法输出结果

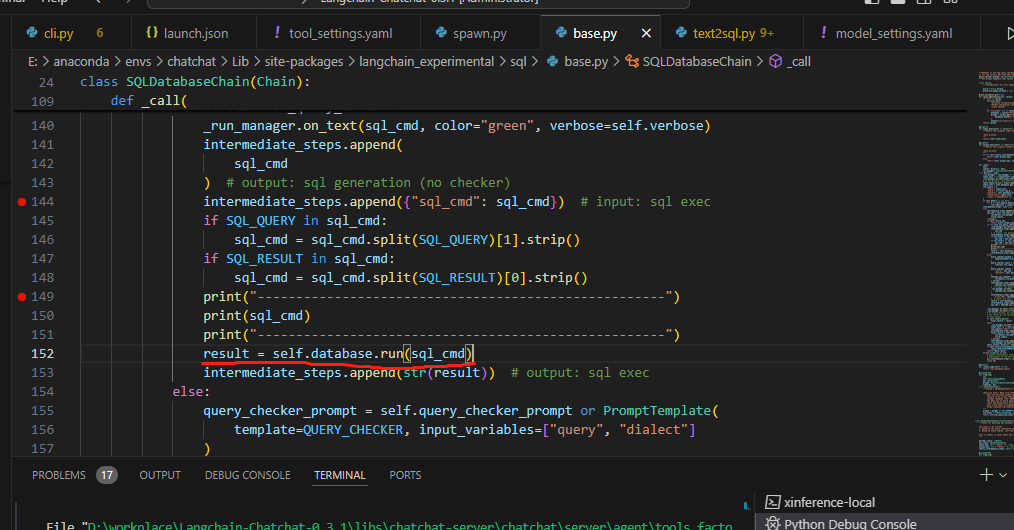


# 



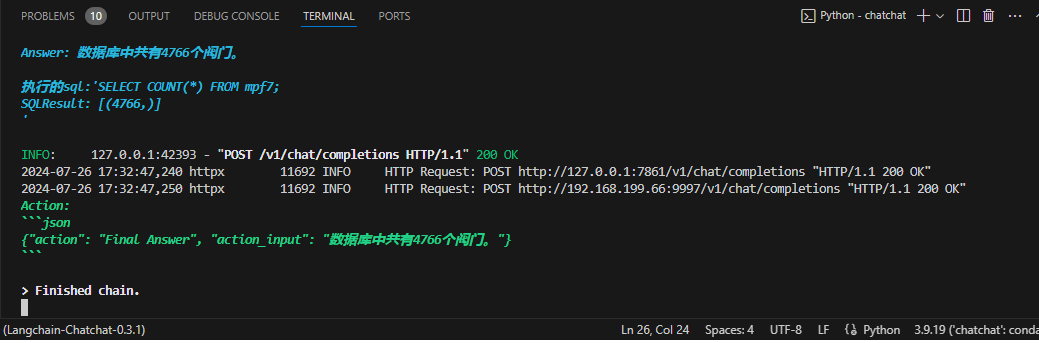
# 

经过对报错的分析，定位到存在问题的代码：

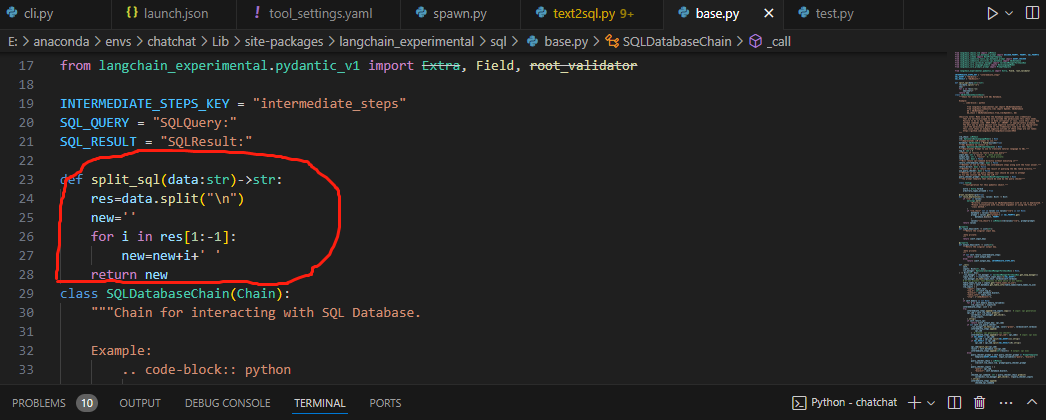


修复bug后的结果图：



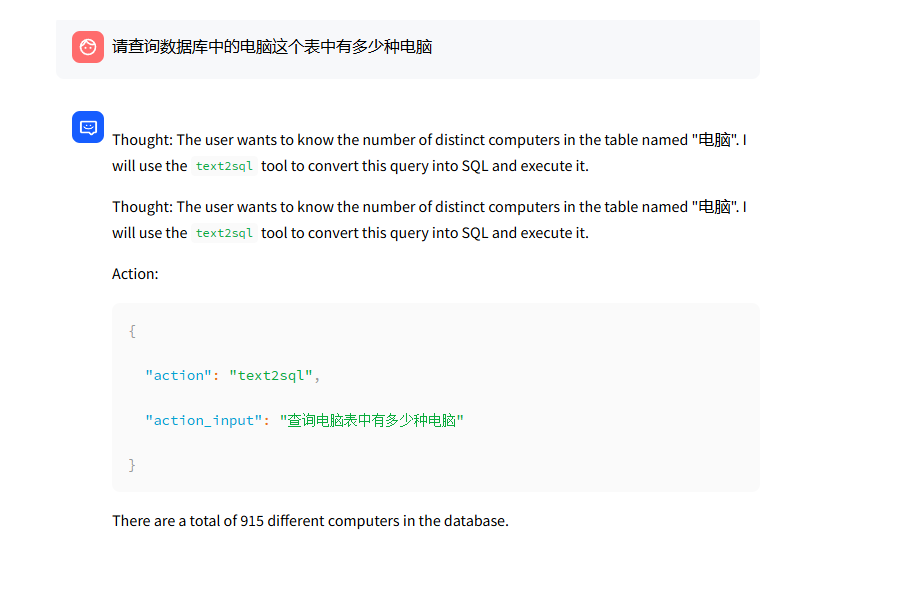


修复方法：



添加split\_sql函数处理生成的sql语句

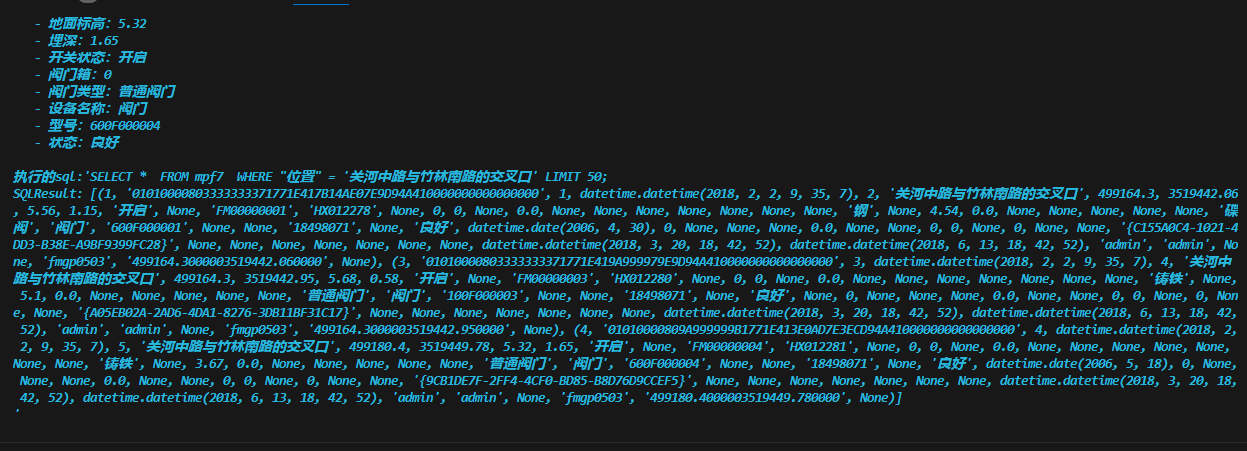
#### 2.验证输入错误信息，sql是否会报错



测试的数据库中不存在电脑这个表，但仍然输出了结果。结论是不会报错，会输出错误信息

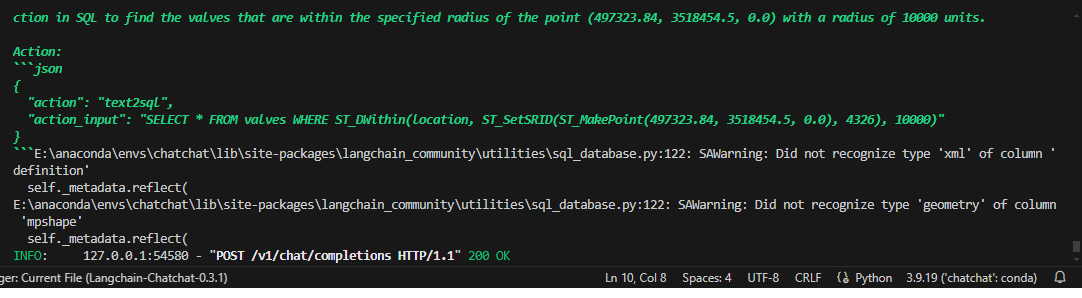
#### 3.测试sql的属性字段查询





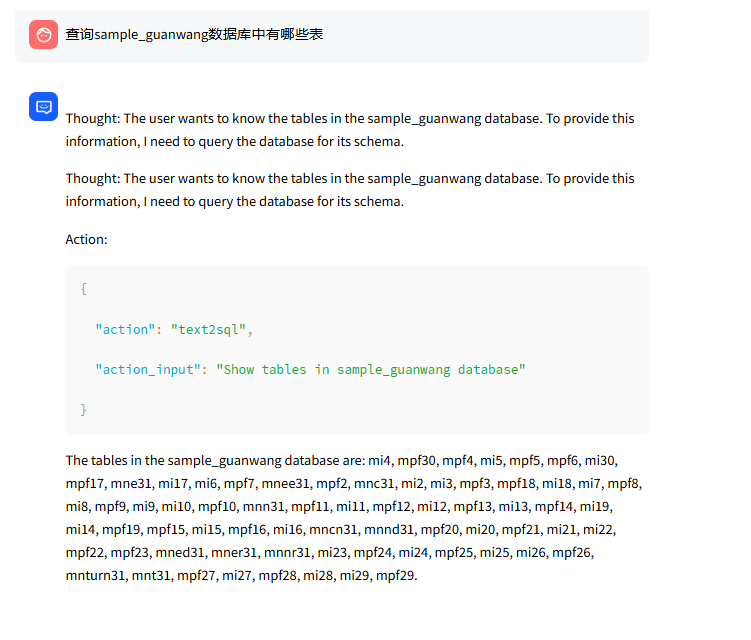
#### 4.测试sql的空间查询

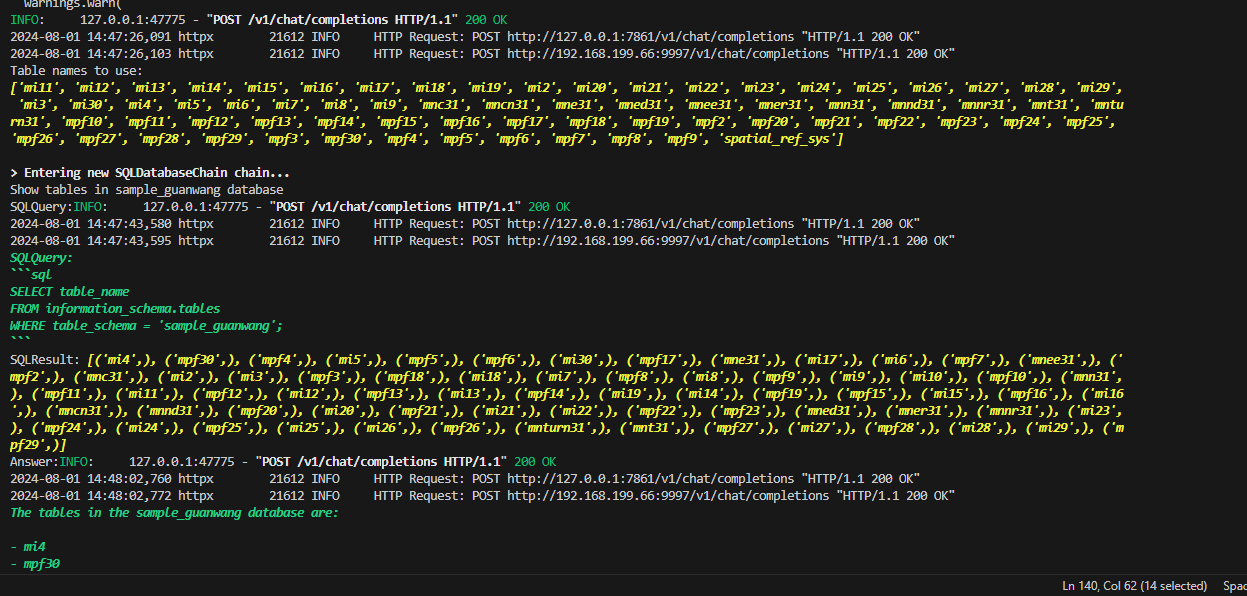




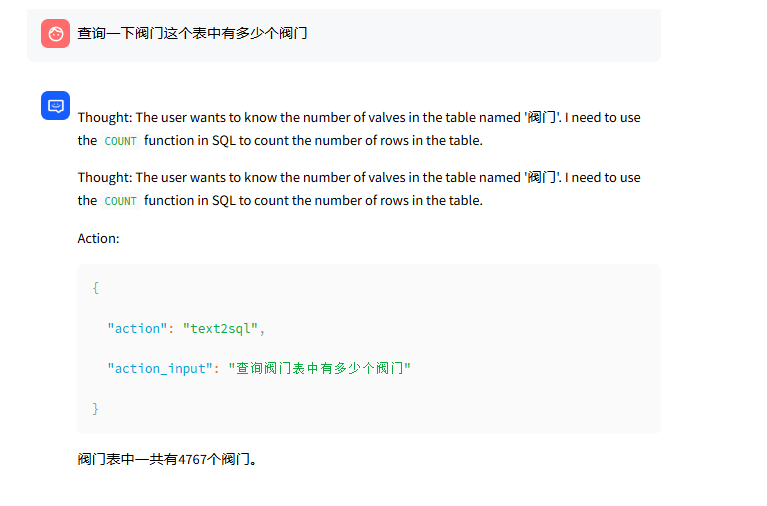
测试sql的空间查询，成功输出sql语句，但不正确

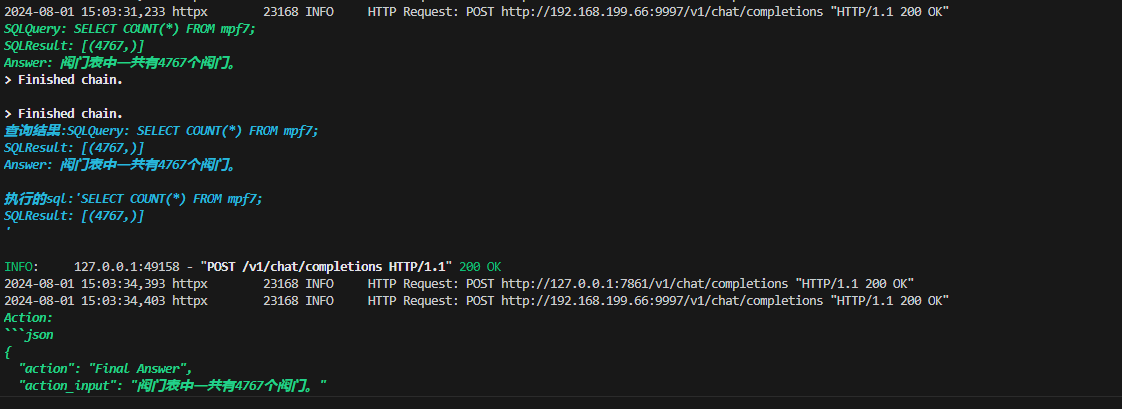
#### 5.查询数据库中有哪些表





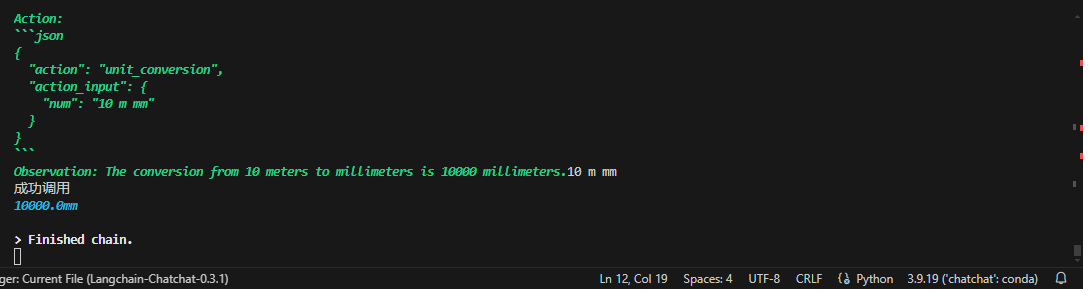
#### 6.查询表内的信息统计结果

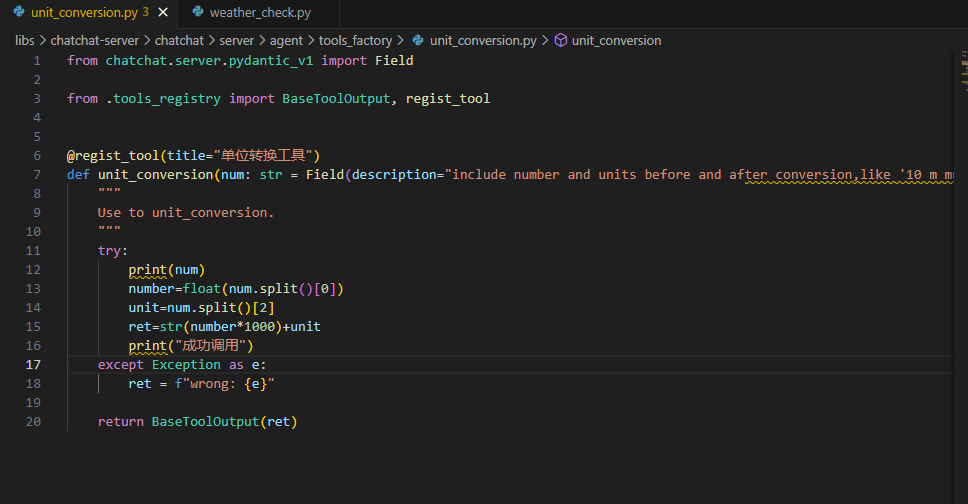




**8. 增加tool的demo**

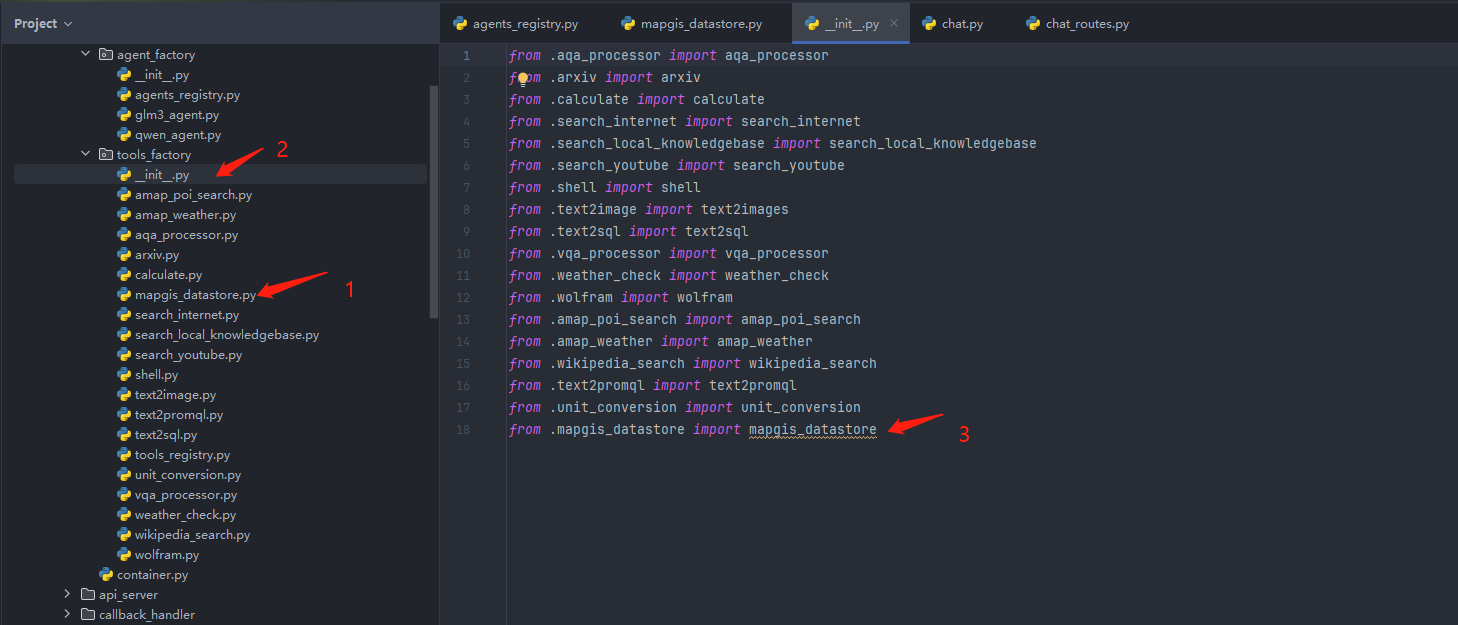
# 



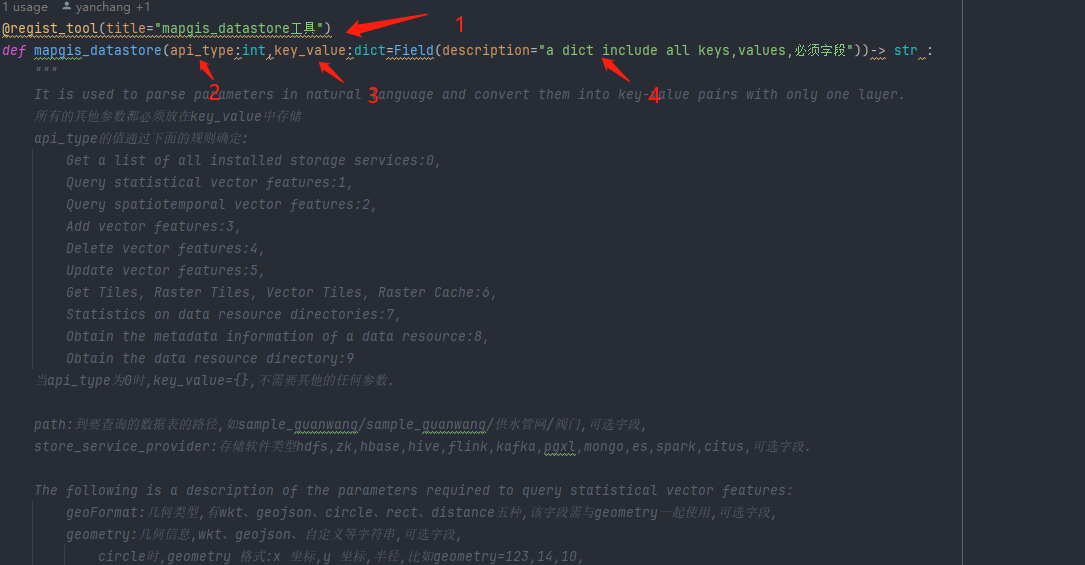


field中的description必须严格描述，不能乱写，否则llm无法正确处理数据

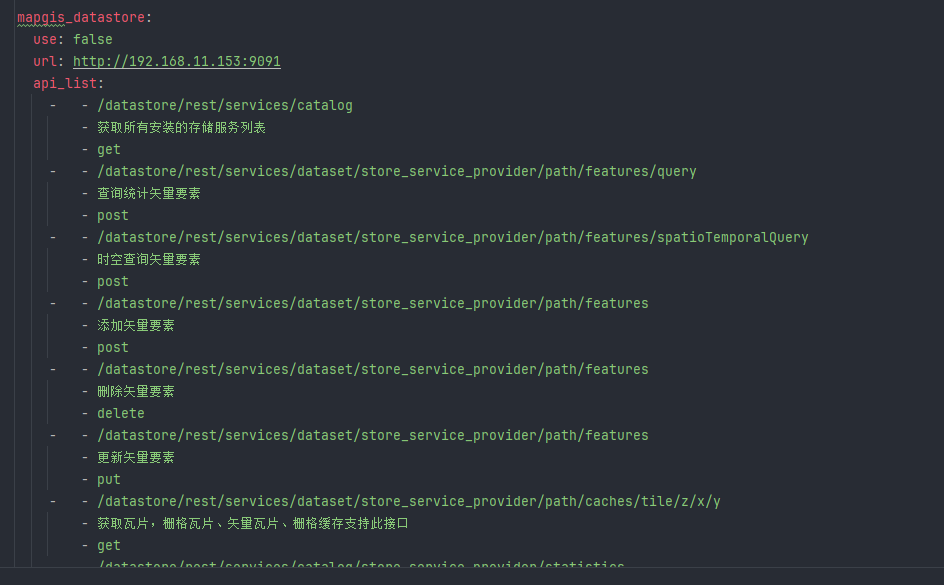
**9. 增加agent调用mapgis\_datastore api**



先在此目录下创建mapgis\_datastore.py文件，然后在init.py中导入包，函数名需要跟工具名一致



然后使用regist\_tool这个装饰器，注册该工具。在函数中需要定义好需要的参数列表，同时需要在函数的注释中添加对函数的说明，对参数字段的说明。自然语言一定要正确



最后需要在添加tool的配置

# 一些可能遇到的问题：

**在chatchat问答页面中出现peer closed connection without sending complete message body**

RemoteProtocolError: peer closed connection without sending complete message body (incomplete chunked read)

**解决办法**

如果不行就删除现有虚拟环境，重新创建，还不行就基于python3.9重新创建

**Chat completion stream got an error: \_get\_logits\_warper() missing 1 required positional argument: 'device' #1752**

**解决办法**

pip install 'transformers==4.41.2'  
pip install sentence-transformers

**xinference载入glm4-chat模型无法正常回复, 报错'ChatGLMForConditionalGeneration' object has no attribute 'stream\_chat'**

**解决办法**

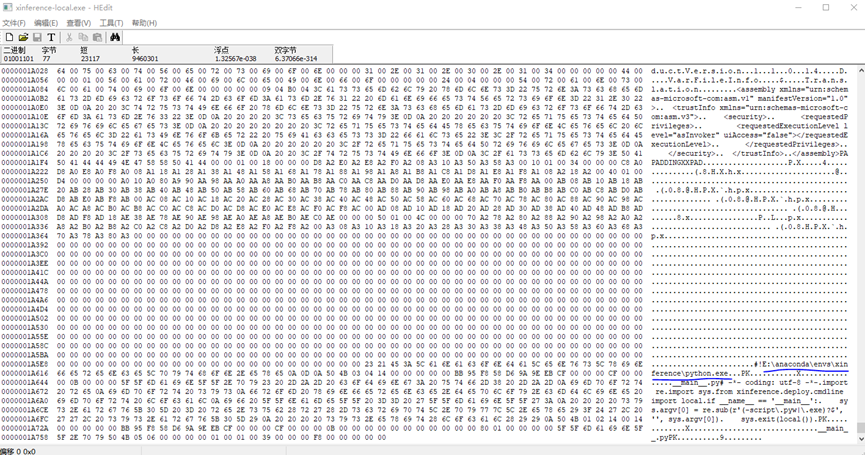
[xinference载入glm4-chat模型无法正常回复, 报错'ChatGLMForConditionalGeneration' object has no attribute 'stream\_chat' · Issue #1915 · xorbitsai/inference · GitHub](https://github.com/xorbitsai/inference/issues/1915)

将cnoda环境下的xinference/api/restful\_api.py，xinference/model/llm/llm\_family.json，xinference/model/llm/pytorch/chatglm.py，xinference/model/llm/utils.py这4个文件替换成上面issue中更新的

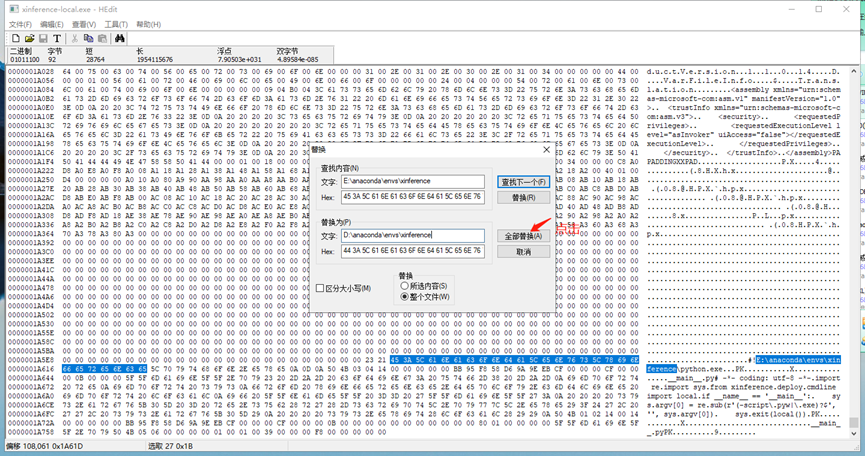
**Fatal error in launcher: Unable to create process using**

Xinference环境迁移导致的问题，需要在envs环境下envs\xinference\Scripts目录下找到

xinference-local.exe文件，用Hedit打开，修改路径为当前xinference环境下python解释器的位置



修改为迁移后的python解释器位置：



保存即可