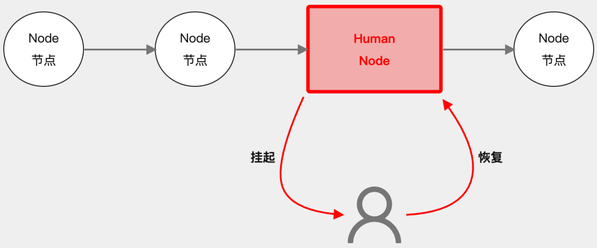
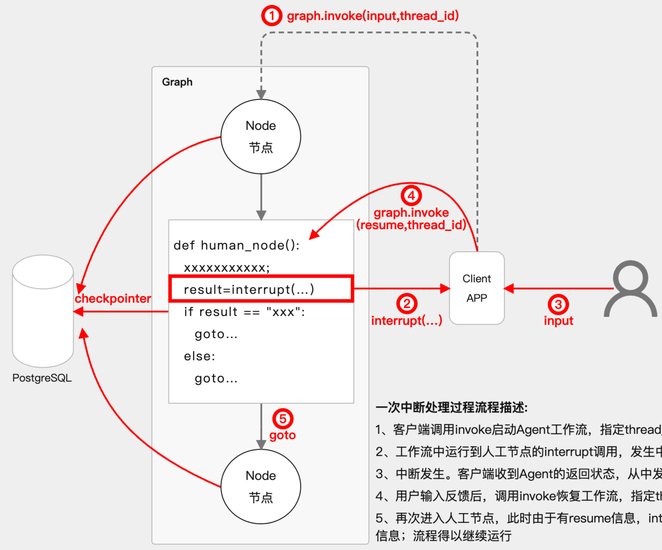


要想实现带有HIL的Agent系统，其关键在于流程的中断和恢复，以及为了支持它所需要的状态持久化机制简单说，就是需要一种机制，将流程“挂起”在特定节点，等待人类参与和反馈，然后能从中断点恢复运行

LangGraph提供的解决方案是:Interrupt(中断) + Command Resume(命令恢复) + Checkpointer(检查点)三大机制

- Interrupt(中断) 暂停LangGraph工作流的执行的同时，返回一个中断数据对象，其中包含给人类的信息，比如需要审核的内容
- Command Resume(命令恢复) 要恢复工作流，需要获得人类反馈信息并注入工作流状态(State)，然后发出继续执行的命令(Command(resume=value)来反馈并恢复)
- Checkpointer(检查点) 为了实现“断点续跑”，必须要实现Agent的状态持久化存储，用来在做恢复时“重建现场”



一次中断处理过程流程描述:

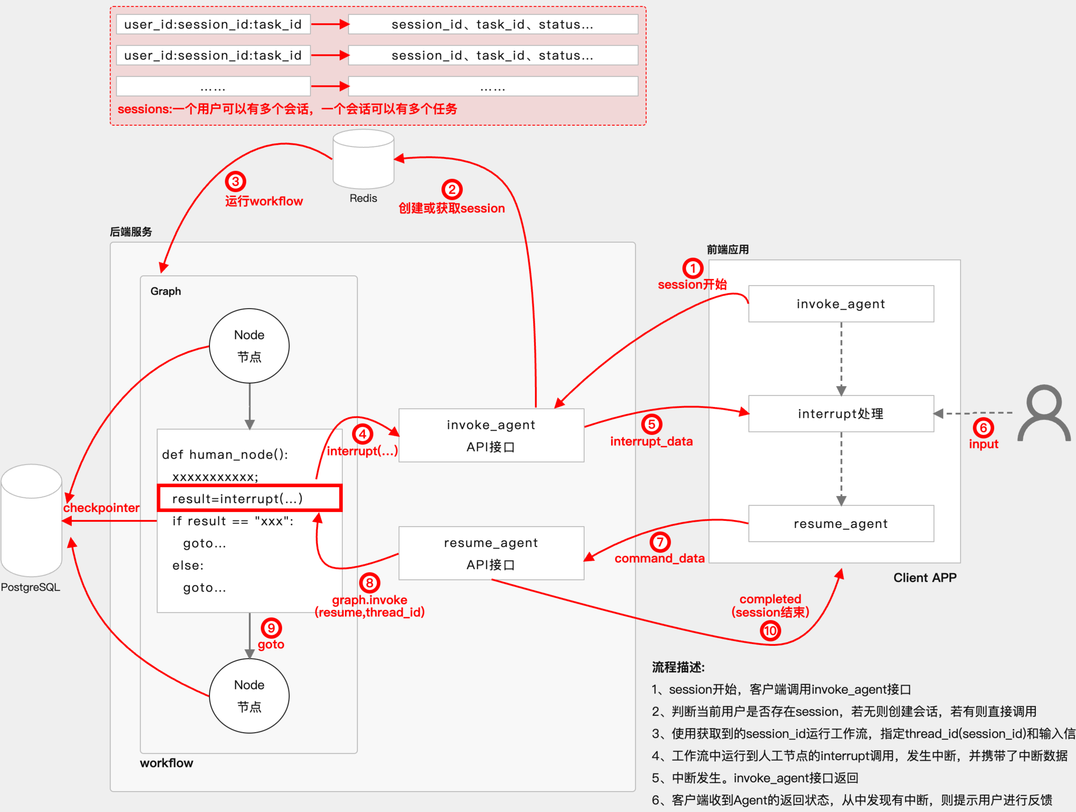
1、客户端调用invoke启动Agent工作流，指定thread_id和输入信息

2、工作流中运行到人工节点的interrupt调用，发生中断，并携带了中断数据

3、中断发生。客户端收到Agent的返回状态，从中发现有中断，则提示用户进行反馈

4、用户输入反馈后，调用invoke恢复工作流，指定thread_id和resume信息

5、再次进入人工节点，此时由于有resume信息，interrupt函数不会触发中断，直接返回resume信息；流程得以继续运行



流程描述:

1、session开始，客户端调用invoke_agent接口

2、判断当前用户是否存在session，若无则创建会话，若有则直接调用

3、使用获取到的session_id运行工作流，指定thread_id(session_id)和输入信息

4、工作流中运行到人工节点的interrupt调用，发生中断，并携带了中断数据

5、中断发生。invoke_agent接口返回

6、客户端收到Agent的返回状态，从中发现有中断，则提示用户进行反馈

7、用户输入反馈后，调用resume_agent接口恢复工作流

8、恢复工作流，指定thread_id和resume信息

9、再次进入人工节点，此时由于有resume信息，interrupt函数不会触发中断，直接返回resume信息；流程得以继续运行

10、至此，一次中断过程处理结束，会话结束