京东的基于langchain的知识问答和数据分析流程的开发：

- 京东的基于langchain的知识问答和数据分析流程的开发：
  - 开发一个结合**数据库查询**和**自然语言处理能力**的**智能聊天机器人**，以便**用户通过自然语言**，查询**结构化数据。**
  - 系统维护两类数据
    - **结构化数据**：保存在 **MySQL**。
    - **文档类数据**：以 **Embedding** 存入 **Faiss** 向量库。
  - **首先要对于结构化表元数据生成M-Schema 元数据**：将表结构转为半结构描述（数据库、表、列、类型、主键/外键、列描述与示例值等），统一作为 schema 的权威来源。
    - 例子

M-Schema

【DB_ID】 superhero
【Schema】
Table: hero_power
[
(hero_id:INTEGER, Primary Key, the id of the hero
Maps to superhero(id), Examples: [1, 2, 3]),
(power_id:INTEGER, the id of the power
Maps to superpower(id), Examples: [1, 18, 26])
]
Table: superpower
[
(id:INTEGER, Primary Key, the unique identifier of the
superpower, Examples: [1, 2, 3]),
(power_name:TEXT, the superpower name, Examples:
[Agility, Accelerated Healing, Lantern Power Ring])
]
【Foreign keys】
hero_power.power_id=superpower.id

  - **意图识别，最重要的环节，正确率瓶颈**
    - 收到用户问题后，先做意图判断：**知识问答** or **数据查询**。
    - **使用Schema 感知**
    - schema感知：词表加向量索引
      - 词表
        - 把业务里会出现的**规范名及它们的同义词、别名、缩写整理成一个映射表**
          - 比如：把“成交额/交易额/GMV”都归一到 GMV
        - 计算“**命中率**”（问题里有多少词能对上库内实体）

    - 向量索引：把M-schema元数据通过embeding 模型保存到vector store，计算“**语义相似度**”
    - 然后把（词表命中率）和（向量最大相似度）结合来得到一个Text2SQL 的概率
        - 大于60%，走text2sql，
- **对话记忆（Chat Memory）**
    - 采用 **Redis** 记录最近 **100 条**消息（含 user/assistant），
    - 作为 chat_history 注入到意图识别、问题改写与生成阶段，沿用已确认的时间范围/口径/维度等默认参数。
- 若判定为**知识问答（RAG链路）**：
    - 在向量库中多路召回：**bge Embedding** 相似度 + **BM25** 关键词检索并行执行，按预设权重融合打分，取前 **Top-N** 语义相关块。
    - 将「用户问题 + Top-N 检索结果 + chat_history」一并发送给大模型，生成自然语言回答（可附来源段落）。
- 若判定为**数据查询（Text2SQL 链路）**：
    - 和文档类数据不用同一个向量库，避免污染召回结果
    - **M-Schema 元数据**：将表结构转为半结构描述（表、列、类型、主键/外键、列描述与示例值等），统一作为 schema 的权威来源。
    - **Schema 向量筛选**：基于 Embedding 相似度，先识别与本问题最相关的**表/列**，只把这部分 **M-Schema 子集** 注入 Prompt，避免把全库元数据塞进模型，降低上下文负担。
    - **ReAct 代理 + 自我改正**：
        - prompt工程：设置sql具体方言，拦截敏感词（DDL），以及全表扫描的时候只返回前100行
        - 采用 **ReAct** 模式
        - 生成 SQL 前先给出【计划】（涉及的表、Join 路径、过滤条件、聚合口径）。
        - 生成后做【自我检查】（列名/表名是否存在、主外键是否正确、是否包含必要 **WHERE** 以避免全表扫描、是否需要 DISTINCT/NULL 处理等）。
        - 执行报错或结果异常时，根据报错信息进行【自我改正】并自动重试
        - 设置重试次数，超过则返回知识问答
    - **结果生成**：将查询结果 + 元数据说明 + 用户问题语境（含 chat_history）再次组织 Prompt，经 LCEL 链式工作流输出**结构化结论 + NLP 解读**