Memory-augmented Query Reconstruction for LLM-based Knowledge Graph Reasoning

Anonymous ACL submission

Abstract

Large language models (LLMs) have achieved remarkable performance on knowledge graph question answering (KGQA) tasks by planning and interacting with knowledge graphs. However, existing methods often confuse tool utilization with knowledge reasoning, harming readability of model outputs and giving rise to hallucinatory tool invocations, which hinder the advancement of KGQA. To address this issue, we propose **Mem**ory-augmented **Q**uery Reconstruction for LLM-based Knowledge Graph Reasoning (MemQ) to decouple LLM from tool invocation tasks using LLM-built query memory. By establishing a memory module with explicit descriptions of query statements, the proposed MemQ facilitates the KGQA process with natural language reasoning and memory-augmented query reconstruction. Meanwhile, we design an effective and readable reasoning to enhance the LLM's reasoning capability in KGQA. Experimental results that MemQ achieves state-of-the-art performance on widely used benchmarks WebQSP and CWQ. [1]

1 Introduction

Large language models (LLMs) have demonstrated impressive reasoning capabilities in knowledge graph question answering (KGQA) task (Yu et al., 2022; Huang and Chang, 2023; Jiang et al., 2022). Using planning and interactive strategies, current LLM-based KGQA methods conduct the reasoning process on the knowledge graph based on the SPARQL tools and achieve remarkable performance across benchmarks (LUO et al., 2024; Sun et al., 2024; Xu et al., 2024b). Typically, part of these studies directly strengthens the reasoning ability of the LLM to plan tool-based paths and retrieve information from the knowledge graph (Wang et al., 2023b; LUO et al., 2024). The others employ LLMs to construct knowledge reasoning agents that execute the reasoning process on the knowledge graph through continuous tool-based decision-making based on environmental observations (Gu et al., 2023; Jiang et al., 2024; Xu et al., 2024a). These methods have achieved impressive results in the KGQA task.

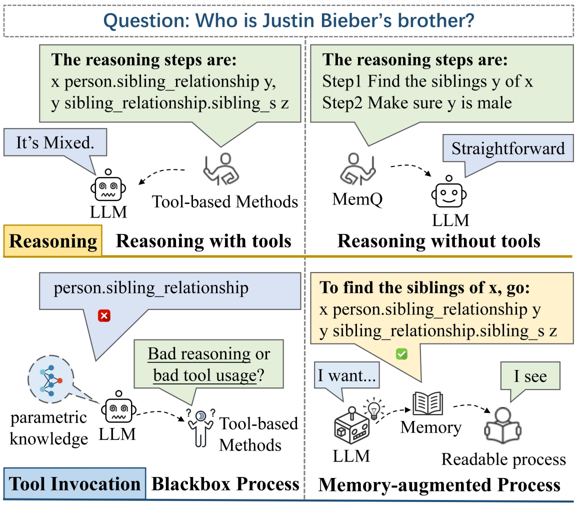


Figure 1: Comparing reasoning methods designed with knowledge graph query tools with proposed memory-augmented method MemQ.

However, the existing methods often confuse tool utilization with knowledge reasoning, harming readability and giving rise to hallucinatory tool invocations. As illustrated in Figure 1, when answering the question "*Who is Justin Bieber's brother?*" existing methods mixed tool invocation with knowledge reasoning tasks, which reduces the model's focus on the knowledge reasoning process (upper left). Furthermore, the mixed reasoning and tool invocation relies heavily on the LLM's parametric knowledge to utilizes the tool effectively, resulting in a black-box reasoning process with low interpretability (bottom left). Constructing a reasoning framework with tool

[1] Our code and data will be released upon acceptance.

invocation steps impairing readability and leading to erroneous tool invocations.

To address the issue, we propose **Mem**ory-augmented **Q**uery Reconstruction for LLM-based Knowledge Graph Reasoning (MemQ) to decouple LLM from the tool invocation task using an LLM-built query memory. To establish the query memory, we employ a rule-based strategy to decompose queries into statements, which are then described using the LLM's capabilities, facilitating an independent reasoning process. We design an effective reasoning strategy based on natural language, enhancing readability and generating explicit reasoning steps. Based on the developed steps, MemQ retrieves memory based on semantic similarity and reconstructs the final query to interact with the knowledge graph. By establishing this query memory, the MemQ approach enables the model to disengage from tool invocation and focus on generating readable knowledge reasoning steps. Our main contributions are:

- We propose MemQ, a memory-augmented LLM-based KGQA reasoning framework to decouple reasoning from tool invocation task in the KGQA process.
- The designed reasoning and memory construction strategies realize a readable LLM-based KGQA process, significantly alleviating the hallucinatory tool invocation issue.
- The proposed MemQ achieved state-of-the-art performance on two widely used benchmarks WebQSP and CWQ.

2 Related Works

Memory-augmented LLM Generation. Though large language models have demonstrated remarkable performance across tasks, they still struggle to achieve consistent performance on complex reasoning tasks (Wang et al., 2024b). In this context, the approach of constructing an external knowledge base to record key information has been proposed and shown to be beneficial (Hu et al., 2023; Anokhin et al., 2024). Researchers have proposed strategies to enhance LLM memory using external modules to support long-term dialogue history referencing (Lee et al., 2024; Rezazadeh et al., 2024). For tasks requiring extensive domain knowledge, methods for constructing memory banks either manually or using large models have also been proven effective (Cheng et al., 2024; Panda et al., 2024; Edge et al., 2024).

Knowledge Graph Question Answering. Early KGQA approaches focused on using networks like key-value memory and graph neural networks to represent inference paths (Miller et al., 2016; Yasunaga et al., 2021; Jiang et al., 2022), while other approaches teach models to build database queries such as SPARQL for direct answer retrieval (Gu and Su, 2022; Ye et al., 2022). With the rise of large language models (LLMs), methods utilize LLM's graph reasoning capability to enhance the reliability of reasoning process (Zhong et al., 2024; Wang et al., 2024a; Zhu et al., 2024). Certain approaches are developed to leverage scaled models to directly interact with Knowledge Graphs or for generating labels that assist smaller models in distilling reasoning abilities (Sun et al., 2024; LUO et al., 2024; Xu et al., 2024b). Other efforts focus on constructing decision datasets based on annotated data to perform a supervised fine-tuning process, which enhance LLM's understanding of the knowledge reasoning process and their ability to interact with knowledge graphs (Jiang et al., 2024). Since LLM-generated outputs are generally susceptible to hallucinatory behavior, some research has shifted to employing discriminative strategies instead of generative ones to reduce unfounded reasoning processes (Gu et al., 2023; Xu et al., 2024a).

However, the issue of confusing the tool invocation process with the knowledge reasoning process remains unresolved. The existing method often conducts reasoning based on SPARQL-formed edges like 'type.domain.property' or self-designed toolboxes, which diminishes the model's focus on the reasoning process and suffers from hallucinatory tool invocation behaviors. In this paper, we propose a memory-augmented KGQA reasoning method that effectively decouples the reasoning process from tool invocation.

3 Framework with Memory Construction

In this section, we introduce the framework of MemQ to decouple the reasoning process from tool invocation; the overall flow is illustrated in Figure 2. We propose to facilitate the KGQA process using three tasks including memory construction, knowledge reasoning and query reconstruction. Before discussing the three tasks, we first illustrate the memory construction process.

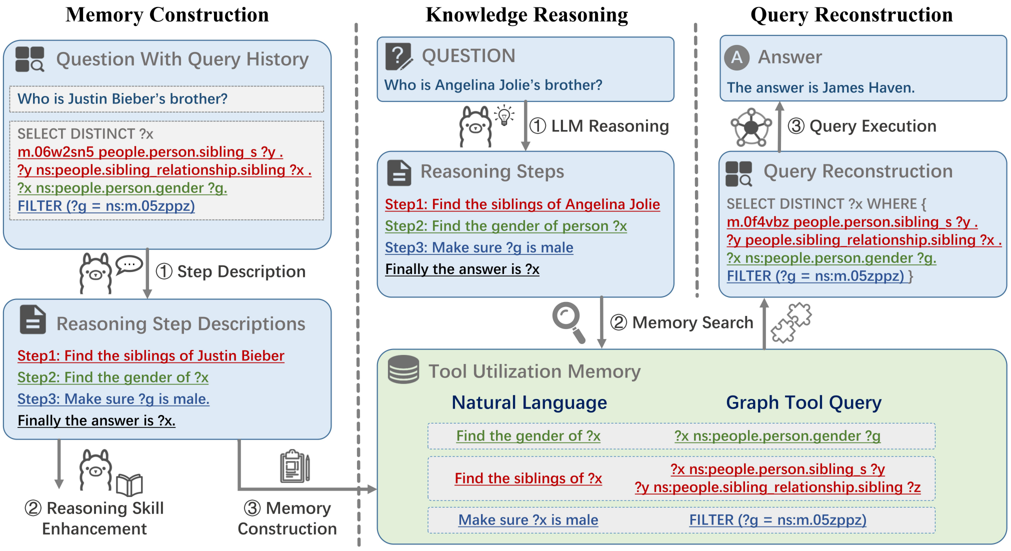

Memory Construction
Knowledge Reasoning
Query Reconstruction
Question With Query History
Who is Justin Bieber's brother?
SELECT DISTINCT ?x
m.06w2sn5 people.person.sibling_s ?y .
?y ns:people.sibling_relationship.sibling ?x .
?x ns:people.person.gender ?g.
FILTER (?g = ns:m.05zppz)
① Step Description
Reasoning Step Descriptions
Step1: Find the siblings of Justin Bieber
Step2: Find the gender of ?x
Step3: Make sure ?g is male.
Finally the answer is ?x.
② Reasoning Skill Enhancement
③ Memory Construction
QUESTION
Who is Angelina Jolie's brother?
① LLM Reasoning
Reasoning Steps
Step1: Find the siblings of Angelina Jolie
Step2: Find the gender of person ?x
Step3: Make sure ?g is male
Finally the answer is ?x
② Memory Search
Answer
The answer is James Haven.
③ Query Execution
Query Reconstruction
SELECT DISTINCT ?x WHERE {
m.0f4vbz people.person.sibling_s ?y .
?y people.sibling_relationship.sibling ?x .
?x ns:people.person.gender ?g.
FILTER (?g = ns:m.05zppz) }
Tool Utilization Memory
Natural Language
Graph Tool Query
Find the gender of ?x
?x ns:people.person.gender ?g
Find the siblings of ?x
?x ns:people.person.sibling_s ?y
?y ns:people.sibling_relationship.sibling ?z
Make sure ?x is male
FILTER (?g = ns:m.05zppz)

Figure 2: The overall framework of MemQ. During the memory construction stage, we describe the question with its query history using the LLMs to get the reasoning steps. In the inference stage, we reconstruct the query using the recalled query sentences based on the reasoning results.

3.1 Memory Construction

Given query history H the contains question q_i with its corresponding query $query_i$, the memory construction task asks the model to build a memory M to represent the mapping function from natural language descriptions n_i to query statements s_i:

$$s_i = M(n_i),\ s_i \in \text{query}_i. \tag{1}$$

For example, if we have a question "*what does x do?*" and its query "*select y where x people.person.career y.*", we can directly save this pair of query and question into the memory M. It represents a mapping relationship from '*one's job*' with query statement '*x people.person.career y*'. It represents a mapping relationship from 'x's career' with the query statement '*select y where x people.person.career y*'.

3.2 Knowledge Reasoning

Given the question Q, the mentioned entities E, the knowledge reasoning task asks the model to develop an n-step reasoning plan P to answer the question. Here we regulate P with the rule that each reasoning step p_i is limited to searching or examining only one entity. The n-step plan P can be represented as a set of reasoning steps:

$$P = \{p_i | i = 1, 2, ..., n\}. \tag{2}$$

For example, when answering the question "*Who is Justin Bieber's Brother?*", the ideal reasoning step will start with the only known entity "*Justin Bieber*" and search for the siblings of this person, and then we may figure out which one of the siblings is male to match with '*brother*'. Note that we also need to record the retrieved new entities for potential use in subsequent steps, so we will always expect an assignment statement "*and assign it to <variable>.*" in every search step. So we have p_1 ="*Find the siblings of Justin Bieber, assign it to x.*", following by p_2 ="*Find the gender of person x, assign it to g.*".

Reasoning steps that examine the answer or the value of a certain entity are often needed to meet the requirement of the question Q. In the previous example, we will have p_3="*Make sure g is male.*" and p_4="*The answer is x.*" to examine the value of g and the position of the answer among known entities. Thus, a 4-step plan $P = \{p_1, p_2, p_3, p_4\}$ is given for the question.

3.3 Query Reconstruction

Given the developed reasoning plan P and the query memory M, the query reconstruction task asks the model to first recall proper query statements s_i using M and then reconstruct the final query Q_f corresponding to the question Q using the set of collected statements:

$$\begin{gathered} s_i = M(p_i), \\ Q_f = \text{Re-con}(S), \\ p_i \in P,\ s_i \in S. \end{gathered} \tag{3}$$

Since the memory M is constructed in a key-value form, we can directly recall the most similar memory using M to reconstruct the new query. Referring to the example from the previous section, concerning the reasoning step "*find the gender of x*", we expect the most similar memory to be recalled as "*x people.person.gender g*."

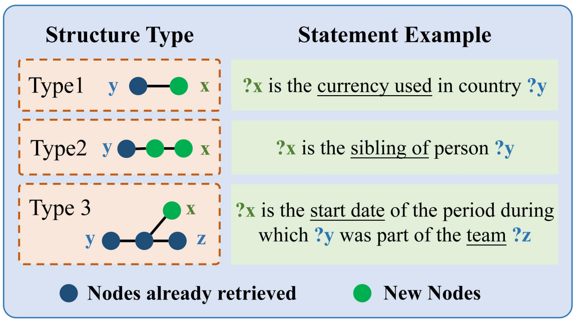
4 Approach

After we propose the MemQ framework, we are able to design efficient strategies to facilitate memory construction, knowledge reasoning, and query reconstruction. Based on the tasks, we model the KGQA process as illustrated in Figure 2.

4.1 Memory Construction Strategy

MemQ utilizes a rule-based strategy to decompose queries and then gather the description of each statement using the LLM. Based on the corresponding descriptions and statements, we establish a memory for the query statements to augment the query reconstruction process.

Rule-based Decomposing. Not every triplet in the knowledge graph conveys a readable meaning that can be described in natural language, which arises from the Compound Value Type (CVT) nodes that lack inherent semantic meanings. When splitting query statements, MemQ always uses non-CVT nodes as the starting or ending nodes, while regarding any encountered CVT nodes as intermediate nodes to ensure the semantic readability of individual statements. If no CVT node is encountered, the statement will contain only a single triplet.



Structure Type	Statement Example
Type1	?x is the currency used in country ?y
Type2	?x is the sibling of person ?y
Type 3	?x is the start date of the period during which ?y was part of the team ?z

Figure 3: Here we present the illustration of the 3 distinct graph structures.

Using the strategy introduced, we can get query statements each of which stands for an operation with an atomic semantic message, such as "someone's hometown". As illustrated in Figure 3, the established memory contains statements with three distinct structures. By decomposing the queries, we obtained a total of 481 statements for type 1, 371 for type 2, and 142 for type 3.

Description Collection. For each statement, we use the LLM to provide a natural language description and store them in the query memory in pairs. We provide task instructions and examples in the context of conducting few-shot generation to ensure the quality of the description and prevent excessive differences between descriptions. We adopt GLM-4 as the description model to generate the descriptions. The prompt templates are shown in Appendix B. The memory construction process is actually a summarization and compression of historical search queries, providing readable hints to future query reconstruction process.

4.2 LLM Reasoning in Natural Language

As shown in Figure 2, after obtaining the corresponding description of each query statement, MemQ uses those explanation-statement pairs to finetune the LLM to enhance its reasoning capabilities (bottom left). By adopting a memory-enhanced approach instead of using a model to directly generate query invocation content, MemQ only requires the LLM to focus on the reasoning process by generating reasoning steps based on the questions using natural language. The generated reasoning steps will be used for memory reconstruction process.

4.3 Query Reconstruction Strategy

During the query reconstruction process, MemQ iterates and alternates between the two sub-steps of memory recall and statements assembling according to the reasoning steps planned in the previous task, until the end of the reasoning steps is reached. As the query is reconstructed, it is executed to retrieve the final answer from the knowledge graph.

Adaptive Memory Recall Strategy. Given the developed reasoning steps, MemQ recalls relevant memory based on semantic similarity and employs rule-based methods to concatenate these statements to reconstruct a complete query. To measure the semantic similarity, we use Sentence-BERT to encode the reasoning steps and the explanations in the memory. Since the similarity scores of the top-N memory fragments can be nearly identical, MemQ adopts an adaptive recall strategy to retrieve

Method	**WebQSP**		**CWQ**	
	Hits@1	F1	Hits@1	F1
Llama2-7b zero-shot (Touvron et al., 2023)*	0.403	0.293	0.297	0.272
Llama3-8b zero-shot (Dubey et al., 2024)*	0.303	0.257	0.305	0.278
Qwen2.5-7b zero-shot (Yang et al., 2024)*	0.284	0.237	0.259	0.241
KV-Mem (Miller et al., 2016)	0.467	0.345	0.184	0.157
GraftNet (Sun et al., 2018)	0.664	0.604	0.368	0.327
QGG (Lan and Jiang, 2020)	0.730	0.738	0.369	0.374
NSM (He et al., 2021)	0.687	0.628	0.476	0.424
SR+NSM (Zhang et al., 2022)	0.689	0.641	0.502	0.471
SR+NSM+E2E (Zhang et al., 2022)	0.695	0.641	0.493	0.463
DECAF (DPR+FiD-3B) (Yu et al., 2022)	0.821	0.788	-	-
UniKGQA (Jiang et al., 2022)	0.751	0.702	0.507	0.480
KD-CoT (Wang et al., 2023a)	0.686	0.525	0.557	-
ToG w/ChatGPT (Sun et al., 2024)	0.758	-	0.589	-
ToG w/GPT-4 (Sun et al., 2024)	0.826	-	0.676	-
KG-Agent (Jiang et al., 2024)	0.833	0.810	0.722	0.692
RoG (Top-3 relation path) (LUO et al., 2024)*	0.795	0.701	0.567	0.547
MemQ (Ours)	**0.841**	**0.858**	**0.803**	**0.830**

Table 1: The results of our method compared with previous approaches on WebQSP and CWQ. The asterisk * denotes the results we reproduced. Note that the Hits@1 result reported in the original RoG paper (WebQSP 0.857, CWQ 0.626) is not calculated in the right way, see the author's response here.

the statements from the memory:

$$N = \begin{cases} 1 & \text{if top-1 similarity} \geq \gamma_1, \\ k & \text{if top-1 similarity} < \gamma_1, \end{cases} \qquad (4)$$
$$k = \text{count}_{case}(\text{similarity} \geq \gamma_2).$$

Rule-based Reconstruct Strategy. MemQ designs a rule-based reconstruction strategy where the most recently recalled sentence is appended to the end of the existing query. Note that we allow the LLM generate the names of unknown entities (e.g., "*person_n*") in the developed steps, the recalled statements will also be refilled using those names.

5 Experiment

In this section, we first introduce the datasets and evaluation methods used by MemQ. After presenting the main experimental results, we will follow up with reports on several analytical experiments to examine the characteristics of the MemQ method compared to previous methods from various perspectives.

5.1 Benchmarks and Baselines

Benchmarks. To evaluate the knowledge graph question-answering capability of the proposed method, we choose two widely used benchmarks, WebQSP (Yih et al., 2016) and CWQ (Talmor and Berant, 2018).

Metrics. We choose commonly used metrics Hits@1 and F1 for the evaluation process following previous works. For the definitions of metrics, please refer to Appendix A.

Baselines. We select previous SOTA approaches with tool-based strategies as baselines, including RoG with LLM planning and chain-of-thought reasoning strategy (LUO et al., 2024), ToG with interactive strategy (Sun et al., 2024). We also list representative methods and zero-shot performances of widely used LLMs for comparison. We also finetune the LLMs with SPARQL queries for ablation, see Section 5.4.

Base Model. To ensure fairness in comparison, we choose Llama2-7b (Touvron et al., 2023) as the base model following RoG (LUO et al., 2024). In analytical experiments, we adopt a stronger model Llama3-8b to better evaluate the effectiveness of our framework.

5.2 Main Result

The performance of our MemQ framework on the WebQSP and CWQ datasets is presented in Table 1. Our method achieves state-of-the-art results on both benchmarks, as demonstrated by significant improvements in Hits@1 and F1 metrics. The results show the efficiency of proposed framework to decouple reasoning from tool invocation. By

Total Hops	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**avg**
Edge Hitting Rate EHR								
RoG	**0.853**	0.644	0.390	0.276	0.249	0.230	0.283	0.377
MemQ	0.816	**0.844**	**0.854**	**0.851**	**0.854**	**0.861**	**0.939**	**0.860**
Graph Edit Distance with Golden Graph $GoldGED$								
RoG	0.479	2.494	3.764	4.505	5.499	7.193	10.438	4.910
MemQ	**0.158**	**0.465**	**0.909**	**1.364**	**1.611**	**2.531**	**2.250**	**1.327**

Table 2: We evaluate the Edge Hitting Rate and Graph Edit Distance with the golden graph for both our method and RoG. The results indicate that the reconstructed graphs achieve significantly higher accuracy and structural alignment compared to those generated by RoG.

adopting a memory-augmented strategy, MemQ provides a new way to enhance the LLM-based reasoning process.

5.3 Reasoning Capability Analysis

To investigate the improvements brought by our proposed reasoning framework, we conduct experiments to examine the discrepancies between the search graph of the reconstructed queries and that of the golden queries. We evaluate the quality of the developed subgraph from two aspects: 1) the structural accuracy and 2) the edge accuracy. Our analysis specifically targets these dimensions to identify the principal factors driving the observed performance improvements.

The structural accuracy GoldGED is defined as the Graph Edit Distance between the reconstructed graph G_{re} and the golden graph G_{gd}:

$$\text{GoldGED}(G_{re}) = \min_{\pi \in \Pi(G_{re}, G_{gd})} \text{num}(\pi). \tag{5}$$

The edge accuracy is quantified by the Edge Hitting Rate, which is computed using the hitting rate between edges in the golden graph G_{gd} and the edges in the reconstructed graph G_{re}:

$$\text{EHR}(G_{re}) = \frac{\text{num}(\{e | e \in G_{gd} \land e \in G_{re}\})}{\text{num}(\{e | e \in G_{gd}\})}. \tag{6}$$

The reuslts is featured in Table 2. Specifically, MemQ achieves a significantly lower GoldGED, indicating more accurate structural alignment with reference graphs, especially in complex multi-hop scenarios. Additionally, MemQ sustains a higher EHR, demonstrating robust edge accuracy even as the number of reasoning steps increases. Overall, these results emphasize MemQ's superior performance in producing accurate and structurally coherent graph-based reasoning across subgraphs.

5.4 Ablation Study

To further analyze the effectiveness of the proposed framework, we conduct experiments to ablate the strategies in MemQ and observe the change in performance. We design two finetune-based baselines to ablate our strategies. 1) For the query reconstruction process, we directly finetune the model utilizing the statements and the descriptions recorded in the memory (denoted as -w/o QRM) to evaluate the effectiveness of our proposed query memory; 2) For the whole MemQ framework, we finetune the model using queries to simulate a straightforward tool-based reasoning process (denoted as -w/o PE, QRM) to evaluate the effectiveness of the MemQ framework. The results are shown in Table 3.

Strategy	**WebQSP**			**CWQ**		
	Hits@1	F1	EHR	Hits@1	F1	EHR
MemQ	0.857	0.872	0.858	0.817	0.845	0.886
-w/o QRM	0.729	0.743	0.849	0.588	0.620	0.864
-w/o PE,QRM	0.733	0.731	0.739	0.556	0.570	0.806

Table 3: We conduct ablation studies to evaluate the impact of key components in our method by comparing it with two settings: 1) removing the Planning Expert (PE) and 2) removing both the Planning Expert (PE) and the Query Reconstruction Module (QRM).

According to the results, we can observe that: 1) Comparing MemQ with "-w/o QRM", the proposed memory-augmented strategy significantly improves the stability of tool utilization process compared with LLM-based finetuning strategy; 2) Comparing "-w/o QRM" with "-w/o PE, QRM", in the case of using a direct fine-tuning strategy, the method of direct fine-tuning that blends reasoning with tool invocation has lowered the overall F1 and EHR score. Furthermore, given that our method has also improved the overall Hits@1 and

Case Study

Question: who was richard nixon married to?

Generated Plan and Reconstruct Query:
Step1: Find the spouse of richard nixon, assign it to ?x.
Retrieved Query1: ns:m.06c97 ns:people.person.spouse_s ?cvt. ?cvt ns:people.marriage.spouse ?x
Step2: Find the type of union between richard nixon and ?x, assign it to ?type_of_union.
Retrieved Query2: ?cvt ns:people.marriage.type_of_union ?type_of_union
Step3: Make sure ?type_of_union should be Marriage.
Retrieved Query3: FILTER(?type_of_union = ns:m.04ztj)
Finally the answer is?x.
Reconstruct Sparql:

```
PREFIX ns: <http://rdf.freebase.com/ns/> SELECT DISTINCT ?x WHERE{
ns:m.06c97 ns:people.person.spouse_s ?cvt .
?cvt ns:people.marriage.spouse ?x .
?cvt ns:people.marriage.type_of_union ?type_of_union .
FILTER(?type_of_union = ns:m.04ztj).
FILTER (!isLiteral(?x) OR lang(?x) = '' OR langMatches(lang(?x), 'en')).
FILTER(?x != ns:m.06c97) }
```

Output:
Pat Nixon (m.023v03)

Figure 4: Case of MemQ, we retrieve memories based on the reasoning steps and reconstruct the final query.

F1 scores compared to previous tool-based SOTA work, these results demonstrate the enhancement of the proposed decoupling strategy on the reasoning process of LLMs.

5.5 Case Study

To demonstrate the readability of the proposed MemQ method, we present a detailed case that highlight its capability to produce clear, logically consistent reasoning plans and accurate reconstruction queries. Figure 4 provides an example question alongside the corresponding reasoning plan and reconstructed query, demonstrating its readability. Refer to Appendix C for more cases.

5.6 Reasoning Hallucination Analysis

To figure out the impact of our decoupled reasoning strategy on the hallucination issue, we manually check and evaluate the error cases of MemQ and the "-w/o PE, QRM" baseline proposed in the ablation study. To guarantee an objective evaluation, we established criteria to check with the cases: 1) **Correctness**: whether the main reasoning steps contain errors, 2) **Completeness**: whether the reasoning logic lacks necessary filtering conditions, and 3) **Redundancy**: whether the reasoning logic includes irrelevant or unnecessary filtering conditions. We randomly sample 100 cases from the test set to record the frequency of each of the errors. Note that one sample may contain multiple errors at a time.

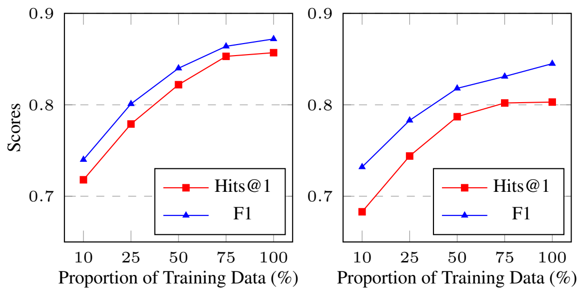
Strategy	Correctness	Completeness	Redundancy
MemQ	**8**	**16**	16
-w/o PE,QRM	39	41	**9**

Table 4: We manually assess the reasoning plans based on Consistency, Completeness, and Redundancy, documenting the number of plans that exhibit errors in each of these categories.

As shown in Table 4, our method significantly reduces the number of Correctness and Completeness errors, while errors in Redundancy slightly increase. The increment in Redundancy errors stems from our retrieval strategy, justified by the presence of edges with similar semantic meanings in the Knowledge Graph (see Appendix C for details). The result indicates that our proposed decoupled reasoning strategy significantly reduces the errors brought by the confusing tasks, indicating an alleviation of the hallucinatory tool invocation issue.

5.7 Data Efficiency Analysis

To assess the data efficiency of our MemQ method, we evaluate the performance of planning expert



(a) WebQSP

(b) CWQ

Figure 5: We evaluate the Hits@1 and F1 scores of the LLaMA-3 Reasoning LLM across varying proportions of training data.

LLM trained with varying levels of training data availability. In this experiment, we randomly selected 10%, 25%, 50%, 75%, and 100% of the step description data to fine-tune the LLaMA-3-8B-Instruct model. As illustrated in Figure 5, our method achieves an F1 score and Hits@1 of approximately 0.7 with only 10% of the training data, significantly outperforming the zero-shot baseline across both datasets. Furthermore, performance improves steadily as the proportion of training data increases, indicating the method's ability to scale effectively with additional data. These results show that our method can effectively utilize limited data, highlighting its strong data efficiency, with consistently improved performance among different volume of training data.

5.8 Model Universality Analysis

Base Model	WebQSP		CWQ	
	Hits@1	F1	Hits@1	F1
Vicuna-7b	0.828	0.846	0.796	0.826
Llama2-7b	0.841	0.858	0.803	0.830
Llama3-8b	0.858	0.872	0.818	0.845
Qwen2.5-7b	0.828	0.850	0.793	0.818

Table 5: We fine-tuned four widely-used LLMs to assess method versatility, with all models demonstrating strong performance, confirming the approach's robustness across diverse architectures.

To demonstrate the robustness and versatility of our MemQ, we conduct fine-tuning experiments on four distinct, widely-used large language models (LLMs) serving as the Planning Expert to generate the reasoning steps. The results in Table 5 demonstrate that all models achieved strong performance, indicating its adaptability to different LLM architectures and confirming its robustness as a model-agnostic solution for reasoning tasks.

5.9 Error Analysis

To conduct a detailed error analysis, we categorize errors into two distinct types: 1) **Main Path Error**, where the primary reasoning path is incorrect, and 2) **Filtering Error**, which includes cases of excessive or insufficient filtering. This classification allows for a systematic evaluation of the inaccuracies in the reasoning process.

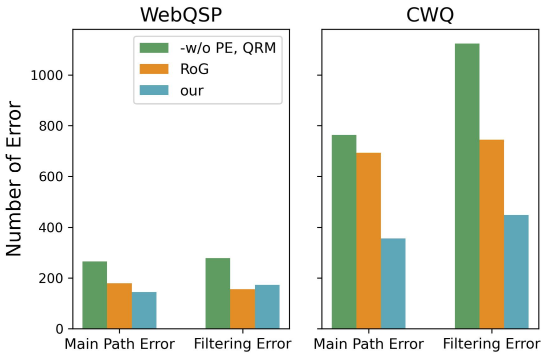


Figure 6: We compare our method with two baselines across two datasets, analyzing the number of errors categorized into two distinct types.

As shown in Figure 6, the Main Path Error of our method is significantly lower than the other two baselines in all datasets. In the CWQ dataset, our method achieves the lowest filtering error among all compared approaches. In the WebQSP dataset, our method achieves substantially lower filtering error compared to the setting without PE and QRM, though it is marginally higher than the RoG method. These results demonstrate the effectiveness of our method in reducing reasoning and filtering errors.

6 Conclusion

In this paper, we propose decoupling LLM from tool invocation tasks using an LLM-built query memory to alleviate hallucinatory tool invocation issues. By facilitating the KGQA process using three tasks, we established a memory module to augment the query reconstruction process in the KGQA task. Based on the framework, we design an effective and readable reasoning strategy to enhance the LLM's reasoning capability, which also alleviates hallucinatory behaviors in existing methods. Experimental results show that our proposed memory-enhanced framework has achieved the state-of-the-art (SOTA) performance on two commonly used benchmarks.

Limitation

Though our proposed MemQ framework has shown competitive KGQA performance and is proven to enhance the LLM's reasoning capability, we identify several limitations that require further improvement. In the future, we will focus on the following directions to extend the current work:
1) Usage of Labeled data: Although our method effectively enhances LLM-based KGQA reasoning process and alleviates the hallucinatory tool invocations, we assume that we have the gold queries to construct the memory. However, it is noteworthy that the decomposing process of the query can be replaced by gathering all the relations and examples of the usage of relations from the Freebase itself. In the future, we will analyze the possibility of model the whole Freebase into a memory to get rid of the demand of gold queries.
2) Plug-and-play Capability: The proposed framework possesses good plug-and-play capability since the constructed memory is a portable module that can be adopted with other reasoning strategies and other tools. In the future, we will conduct experiments to showcase this kind of capability and testify our proposed memory-based framework under multi-tool or task transfer conditions.

References

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Mikhail Burtsev, and Evgeny Burnaev. 2024. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. 2024. Lift yourself up: Retrieval-augmented text generation with self-memory. *Advances in Neural Information Processing Systems*, 36.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Yu Gu, Xiang Deng, and Yu Su. 2023. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949.

Yu Gu and Yu Su. 2022. ArcaneQA: Dynamic program induction and contextualized encoding for knowledge base question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1718–1731, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21. ACM.

Chenxu Hu, Jie Fu, Chenzhuang Du, Simian Luo, Junbo Zhao, and Hang Zhao. 2023. Chatdb: Augmenting llms with databases as their symbolic memory. *arXiv preprint arXiv:2306.03901*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. 2024. Kg-agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. *arXiv preprint arXiv:2402.11163*.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. *arXiv preprint arXiv:2212.00959*.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974, Online. Association for Computational Linguistics.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. 2024. A human-inspired reading agent with gist memory of very long contexts. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

LINHAO LUO, Yuan-Fang Li, Reza Haf, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409, Austin, Texas. Association for Computational Linguistics.

Pranoy Panda, Ankush Agarwal, Chaitanya Devaguptapu, Manohar Kaul, et al. 2024. Holmes: Hyper-relational knowledge graphs for multi-hop question answering using llms. *arXiv preprint arXiv:2406.06027*.

Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. 2024. From isolated conversations to hierarchical schemas: Dynamic tree memory representation for llms. In *The First Workshop on System-2 Reasoning at Scale, NeurIPS'24*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium. Association for Computational Linguistics.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023a. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *Preprint*, arXiv:2308.13259.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023b. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634.

Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. 2024b. Symbolic working memory enhances language models for complex rule application. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17583–17604.

Mufan Xu, Kehai Chen, Xuefeng Bai, Muyun Yang, Tiejun Zhao, and Min Zhang. 2024a. Llm-based discriminative reasoning for knowledge graph question answering. *arXiv preprint arXiv:2412.12643*.

Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Kang Liu, and Jun Zhao. 2024b. Generate-on-graph: Treat llm as both agent and kg in incomplete knowledge graph question answering. *arXiv preprint arXiv:2404.14741*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2022. RNG-KBQA: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6032–6043, Dublin, Ireland. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Yang Wang, Zhiguo Wang, and Bing Xiang. 2022. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. In *The Eleventh International Conference on Learning Representations*.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5773–5784, Dublin, Ireland. Association for Computational Linguistics.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731.

Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2024. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *World Wide Web*, 27(5):58.

A Metrics

In this section, we present the mathematical formulations and explanations for the metrics that were not fully elaborated in the main text.

Hits@1. Hits@1 quantifies the proportion of questions for which the top-ranked answer in the model's output is correct. Let $Answer$ represent the list of predicted answers, $Golden$ denote the list of ground truth answers, and $total\_num$ represent the total number of questions in the dataset. The formula is defined as follows: The formula of Hits@1 is defined as follows:

$$Hits@1 = \frac{count(Answer[0] \in Golden)}{total\_num}. \quad (7)$$

F1. Following previous methods, we use the Macro-F1 scoring method, which calculates the F1 for each test sample and then averages those F1 scores among the samples.

B Prompt Template

The used prompt templates are listed in the following tables. We designs 3 templates for the three types of queries shown in Table 6, Table 7 and Table 8. Besides, for the finetuning process to enhance the LLM's reasoning ability, we use the template in Table 9.

C More Cases

Here, we present two additional cases generated by our method. As shown in Table 10, our method accurately constructs queries with "Order By" and "Limit" clauses in Step 5, demonstrating its ability to interpret the temporal meaning of "last time" in the question, which a nuance often overlooked by previous methods.

In Table 11, our method retrieves multiple queries with similar semantic meanings. While this approach may introduce redundancy, we argue that it is justified given the nature of the Freebase Knowledge Graph, where edges with similar semantic meanings do exist and can be challenging even for humans to distinguish. Consequently, retrieving all such edges ensures comprehensive coverage of potentially relevant answers.

Prompt for Structure 1
Act as a SPARQL expert. I need you to explain the meaning and function of a specific part of a SPARQL query. You job is answer the Question for me. ONLY OUTPUT THE ANSWER, NOTING ELSE!! ### EXAMPLE1 Sparql: ?entity1 ns:location.country.currency_used ?entity2 . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the currency used in the country ?entity1. ### EXAMPLE2 Sparql: ?entity2 ns:location.country.currency_used ?entity1 . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the country that use ?entity1 as currency. ### EXAMPLE3 Sparql: ?entity2 ns:government.election_campaign.candidate ?entity1 . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the election campaign which ?entity1 is the candidate. ### EXAMPLE4 Sparql: ?entity1 ns:government.election_campaign.candidate ?entity2 . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the candidate in the election campaign ?entity1. ### EXAMPLE5 Sparql: { ?entity2 ns:sports.sports_championship_event.runner_up ?entity1 } UNION { ?entity2 ns:sports.sports_championship_event.champion ?entity1 } Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is either the runner-up or the champion of a sports championship event ?entity1. ### EXAMPLE6 Sparql: { ?entity1 ns:location.statistical_region.places_exported_to ?tmp0 . ?tmp0 ns:location.imports_and_exports.exported_to ?entity2 } UNION { ?entity1 ns:location.statistical_region.places_exported_from ?tmp1 . ?tmp1 ns:location.imports_and_exports.exported_from ?entity2 } Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the place that is either exported to or exported from the statistical region ?entity1. ### YOUR TURN Sparql: {sparql} Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer:

Table 6: The prompt to get the explanation of Structure 1 graph

Prompt for Structure 2
Act as a SPARQL expert. I need you to explain the meaning and function of a specific part of a SPARQL query. You job is answer the Question for me. ONLY OUTPUT THE ANSWER, NOTING ELSE!! ### EXAMPLE1 Sparql: ?cvt ns:government.government_position_held.office_holder ?entity1 . ?entity2 ns:government.governmental_body.members ?cvt . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the governmental body that is held by ?entity1. Answer: ?entity2 is the governmental body that has an office holder ?entity1. ### EXAMPLE2 Sparql: ?entity1 ns:film.actor.film ?cvt . ?cvt ns:film.performance.character ?entity2 . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the character played by the actor ?entity1. ### EXAMPLE3 Sparql: ?cvt ns:music.group_membership.member ?entity1 . ?entity2 ns:music.musical_group.member ?cvt . Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer: ?entity2 is the musical group that has the member ?entity1. Answer: ?entity2 is the group that includes the member ?entity1. ### YOUR TURN Sparql: {sparql} Question: How does ?entity2 related to ?entity1 ? Please answer the question with "?entity2 is [noun phrase]" . Answer:

Table 7: The prompt to get the explanation of Structure 2 graph

Prompt for Structure 3

```
Act as a SPARQL expert.
I need you to explain the meaning and function of a specific part of a SPARQL query.
You job is complete the answer for me. ONLY OUTPUT THE ANSWER, NOTING ELSE!!
### EXAMPLE1
Sparql:
?cvt ns:sports.sports_team_coach_tenure.position ?entity1 .
?cvt ns:sports.sports_team_coach_tenure.coach ?entity2 .
?entity3 ns:sports.sports_team.coaches ?cvt .
Question: How does ?entity3 related to ?entity1 and ?entity2 ?
Please answer the question with "?entity3 is [noun phrase]" .
Answer: ?entity3 is the sports team that has a coach ?entity2 who holds the position ?entity1 .
### EXAMLPE2
Sparql:
?entity1 ns:film.actor.film ?cvt .
?cvt ns:film.performance.character ?entity2 .
?cvt ns:film.performance.film ?entity3 .
Question: How does ?entity3 related to ?entity1 and ?entity2 ?
Please answer the question with "?entity3 is [noun phrase]" .
Answer: ?entity3 is the film in which the actor ?entity1 performs the character ?entity2.
Answer: ?entity3 is the film in which ?entity1 acted as a character ?entity2.
### EXAMLPE3
Sparql:
?entity1 ns:sports.pro_athlete.teams ?cvt .
?cvt ns:sports.sports_team_roster.team ?entity2 .
?cvt ns:sports.sports_team_roster.from ?entity3
Question: How does ?entity3 related to ?entity1 and ?entity2 ?
Please answer the question with "?entity3 is [noun phrase]" .
Answer: ?entity3 is the starting date when ?entity1 (the professional athlete) was part of the team ?entity2.
Answer: ?entity3 is the start date of the period during which ?entity1 was part of the team ?entity2.
### YOUR TURN
Sparql:
{sparql}
Question: How does ?entity3 related to ?entity1 and ?entity2 ?
Please answer the question with "?entity3 is [noun phrase]" .
Answer:
```

Table 8: The prompt to get the explanation of Structure 3 graph

Prompt for Plan Expert

You are given a problem to solve step by step. Each step should begin with either "Find", "Make sure" or "Rank". Finally, you need to output which one is the final answer.
The steps should logically follow from one another, where each step builds on the outcome of the previous steps.
Each step should be simple, clear, and directly related to achieving the overall goal.
Some topic entities you can use to start the plan are provided below.
Question:
{question}
Topic Entities:
{topic_entities}

Table 9: The prompt utilized for generating knowledge reasoning plans in the Planning Expert.

WebQTest-994

Question:
When was the last time the Dallas Cowboys won the superbowl?
Generated Plan and Reconstruct Query:
Step1: Find the championship won by the sports team dallas cowboys, assign it to ?x.
Retrieved Query1: ?entity1 ns:sports.sports_team.championships ?entity2
Step2: Find the end date of the event ?x, assign it to ?sk0.
Retrieved Query2: ?entity1 ns:time.event.end_date ?entity2
Step3: Find the championship associated with the sports championship event ?x, assign it to ?championship.
Retrieved Query3: ?entity1 ns:sports.sports_championship_event.championship ?entity2
Step4: Make sure ?championship should be super bowl.
Retrieved Query4: FILTER(?championship = ns:m.06x5s)
Step5: Sort the result based on ?sk0 in descending order and keep the first result.
Retrieved Query5: ORDER BY DESC(?sk0) LIMIT 1
Finally the answer is ?x.
Reconstruct Sparql:

```
PREFIX ns: <http://rdf.freebase.com/ns/> SELECT DISTINCT ?x WHERE{
ns:m.02896 ns:sports.sports_team.championships ?x .
?x ns:time.event.end_date ?sk0 .
?x ns:sports.sports_championship_event.championship ?championship .
FILTER(?championship = ns:m.06x5s).
FILTER (!isLiteral(?x) OR lang(?x) = ” OR langMatches(lang(?x), ’en’)).
FILTER(?x != ns:m.02896) }
ORDER BY DESC(?sk0) LIMIT 1
```

Output:
Super Bowl XXX (m.076wz)

Table 10: Case Study 1

WebQTest-212_3c65a82425ff4488504b53ba502d0eb7

Question:
What is the name of the state that has the elected official Jan Brewer and also has the
Colorado River flow through it?
Generated Plan and Reconstruct Query:
Step1: Find the location that partially contains *Colorado River*, assign it to?x.
Retrieved Query1: {?entity1 ns:location.location.partially_containedby ?entity2 } UNION
{?entity2 ns:location.location.partially_contains ?entity1 }UNION
{?cvt ns:location.partial_containment_relationship.partially_contains ?entity1 .
?entity2 ns:location.location.partiallycontains ?cvt }UNION
{?entity1 location.location.partially_contained_by ?cvt
?cvt ns:location.partial_containment_relationship.partially_contained_by ?entity2}
Step2: Find the official who appointed the governing official ?x, assign it to ?appointed_by.
Retrieved Query2: ?entity1 ns:government.governmental_jurisdiction.governing_officials ?cvt .
?cvt ns:government.government_position_held.appointed_by ?entity2 .
Step3: Make sure ?appointed_by should be Jan Brewer.
Retrieved Query3: FILTER(?appointed_by = ns:m.02pkb1c)
Finally the answer is ?x.
Reconstruct Sparql:
{ns:m.018qjq ns:location.location.partially_containedby ?x } UNION
{?x ns:location.location.partially_contains ns:m.018qjq } UNION
{?cvt ns:location.partial_containment_relationship.partially_contains ns:m.018qjq .
    ?x ns:location.location.partiallycontains ?cvt } UNION
{ns:m.018qjq ns:location.location.partially_contained_by ?cvt1 .
    ?cvt1 ns:location.partial_containment_relationship.partially_contained_by ?x }.
?x ns:government.governmental_jurisdiction.governing_officials ?cvt2 .
    ?cvt2 ns:government.government_position_held.appointed_by ?appointed_by .
FILTER(?appointed_by = ns:m.02pkb1c).
FILTER (!isLiteral(?x) OR lang(?x) = ” OR langMatches(lang(?x), ’en’)).
FILTER(?x != ns:m.018qjq) }
Output:
Arizona (m.0vmt)

Table 11: Case Study 2