

Application 1

Retrieval Engine

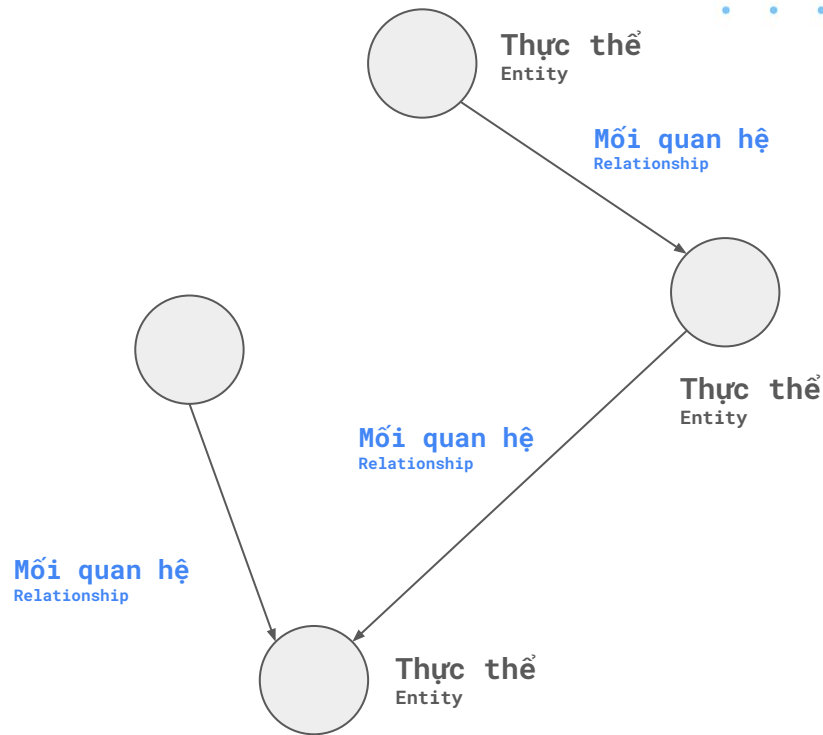


Llama Index + GraphRAG

Llama Index + GraphRAG

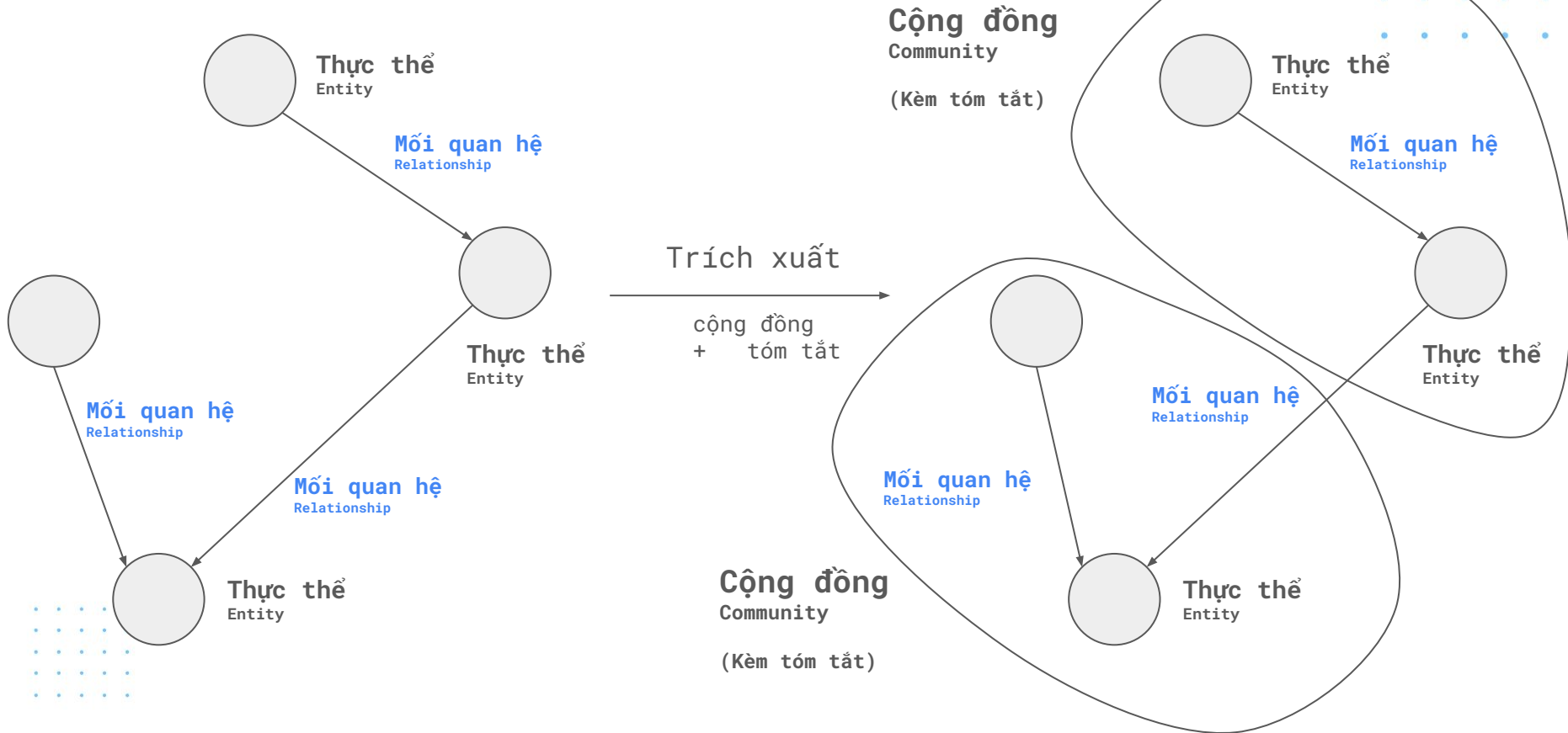


Chuyển thành
đồ thị



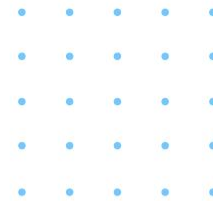
Trích xuất cộng đồng

Communities Extraction

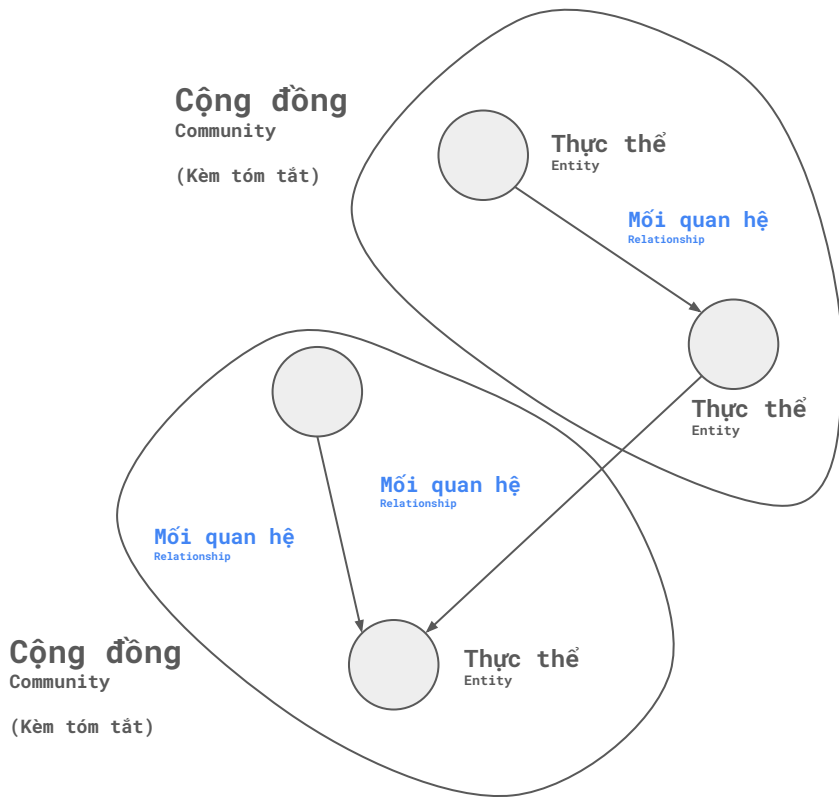


Truy vấn vào cộng đồng

Graph Communities Retrieval



Truy vấn
Query

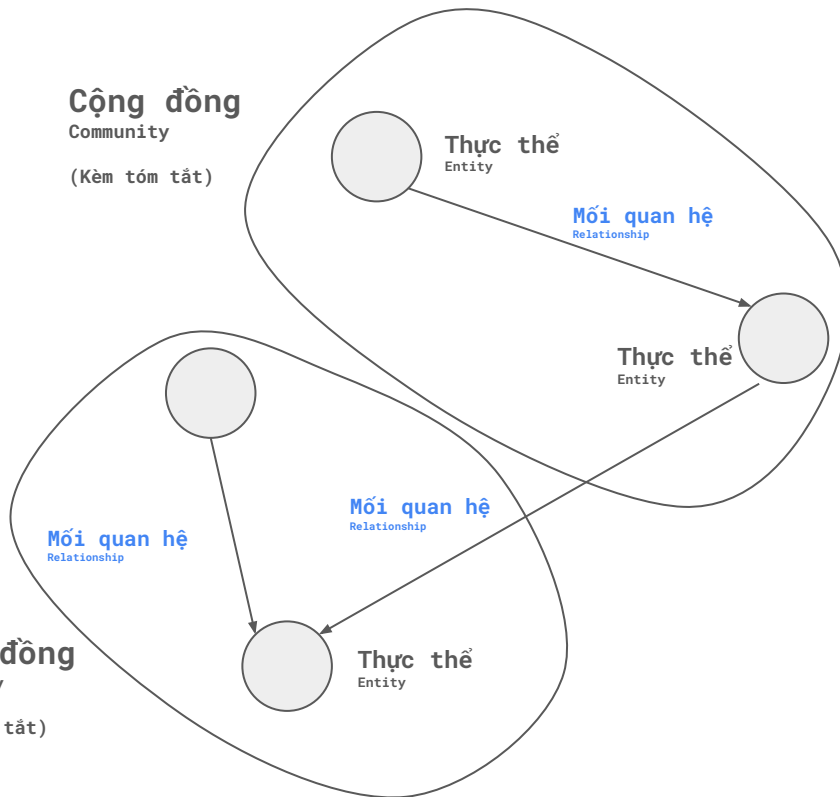
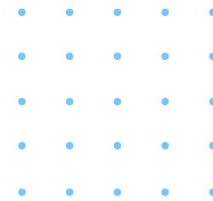


1. Truy xuất các bản tóm tắt cộng đồng
2. Trả lời truy vấn từ mỗi tóm tắt
3. Tổng hợp các câu trả lời này thành một phản hồi cuối cùng.



Truy vấn vào cộng đồng

Graph Communities Retrieval



Bài viết



```
[97] 1 query_engine = GraphRAGQueryEngine(  
2     graph_store=index.property_graph_store, llm=llm  
3 )
```

```
[98] 1 response = query_engine.query(  
2     "Ý tưởng chính của bài này là gì?"  
3 )  
4 display(Markdown(f"{response.response}"))
```

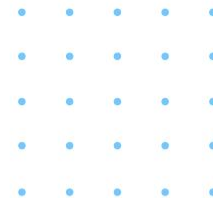


Bài viết chủ yếu nói về vai trò của Thái Tông trong việc hỗ trợ và bảo vệ Trần Quốc Tuấn, đặc biệt là trong việc đảm bảo an toàn khi ông rời cung điện. Ngoài ra, bài viết cũng nhắc đến việc Trần Quốc Tuấn đã kết hôn với Công chúa Thiên Thành.



Độ sâu của community

Community Levels



We compare six different conditions in our analysis, including Graph RAG using four levels of graph communities (**C0**, **C1**, **C2**, **C3**), a text summarization method applying our map-reduce approach directly to source texts (**TS**), and a naïve “semantic search” RAG approach (**SS**):

- **C0**. Uses root-level community summaries (fewest in number) to answer user queries.
- **C1**. Uses high-level community summaries to answer queries. These are sub-communities of C0, if present, otherwise C0 communities projected down.
- **C2**. Uses intermediate-level community summaries to answer queries. These are sub-communities of C1, if present, otherwise C1 communities projected down.
- **C3**. Uses low-level community summaries (greatest in number) to answer queries. These are sub-communities of C2, if present, otherwise C2 communities projected down.
- **TS**. The same method as in [subsection 2.6](#), except source texts (rather than community summaries) are shuffled and chunked for the map-reduce summarization stages.
- **SS**. An implementation of naïve RAG in which text chunks are retrieved and added to the available context window until the specified token limit is reached.

The size of the context window and the prompts used for answer generation are the same across all six conditions (except for minor modifications to reference styles to match the types of context information used). Conditions only differ in how the contents of the context window are created.

The graph index supporting conditions **C0-C3** was created using our generic prompts for entity and relationship extraction only, with entity types and few-shot examples tailored to the domain of the data. The graph indexing process used a context window size of 600 tokens with 1 gleaning for the Podcast dataset and 0 gleanings for the News dataset.

```
import networkx as nx
from graspologic.partition import hierarchical_leiden

# Create a sample graph
G = nx.karate_club_graph()

# Apply the hierarchical Leiden algorithm
hierarchical_partition = hierarchical_leiden(G)

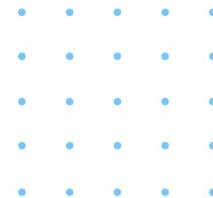
# Extracting different levels of community structures
C0 = hierarchical_partition[0] # Root-level communities
C1 = hierarchical_partition[1] if len(hierarchical_partition) > 1 else
C0 # High-level communities
C2 = hierarchical_partition[2] if len(hierarchical_partition) > 2 else
C1 # Intermediate-level communities
C3 = hierarchical_partition[3] if len(hierarchical_partition) > 3 else
C2 # Low-level communities
```

... Lý tưởng: Tuning độ sâu của community sẽ cho
... kết quả tốt hơn
...
...
...



Code

Code



Tạo Đồ Thị

Tạo Đồ Thị, xây dựng cộng đồng và các bản tóm tắt của chúng trên tài liệu đã cho.

Trả Lời Truy Vấn

Sử dụng các bản tóm tắt của các cộng đồng được tạo từ bước 1 để trả lời truy vấn.

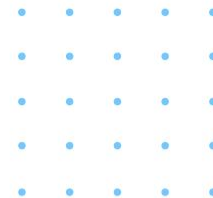
Code

GraphRAG + LlamaIndex



Cons

Cons



Rất tốn kém chi phí (cho LLMs) khi sử dụng Prompt dài để trích xuất Graph + tổng hợp thông tin từ Community

Đề xuất: Sử dụng GraphRAG để xây dựng Knowledge Graph rồi sử dụng Graph Query thay vì sử dụng tổng hợp thông tin từ Community

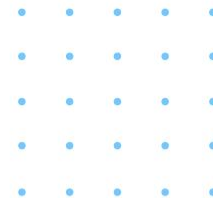




Application 2

Graph Database Building





Hưng Đạo Vương Trần Quốc Tuấn và màn cướp dâu chấn động lịch sử

© Thứ sáu, ngày 20/01/2023 10:31 AM (GMT+7)

Hưng Đạo Vương Trần Quốc Tuấn là vị tướng tài ba, nổi tiếng bậc nhất trong lịch sử Việt Nam. Thế nhưng ít ai biết rằng ngoài tận tụy với việc nước, ông còn là người hết lòng vì tình yêu. Chính vì vậy mà màn cướp dâu chấn động triều đình khi đó của Trần Quốc Tuấn vẫn được lưu truyền đến ngày nay.

[Chia sẻ](#) [Thích 6](#) [Bình luận 0](#)

Dân Việt trên [Google News](#)

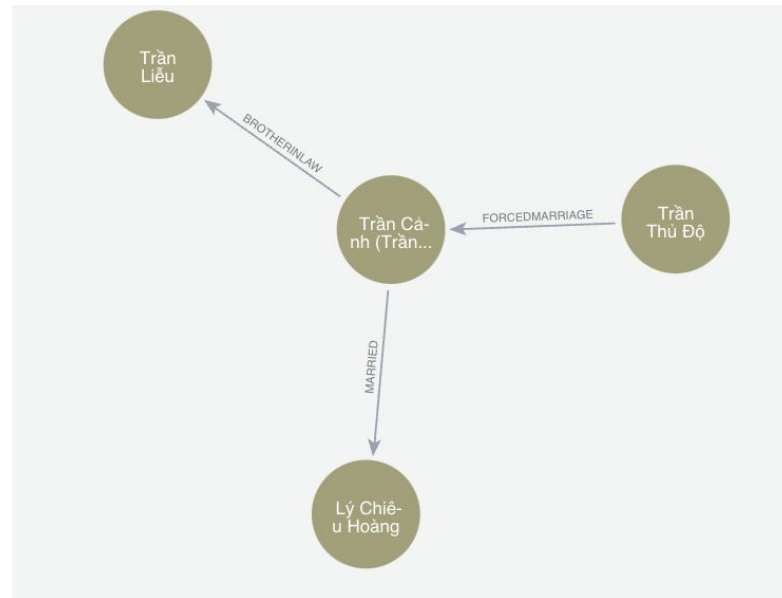
- 3 môn võ công nào trong Kim Dung tuy bất bại nhưng vẫn có điểm yếu?
- Chiếc vòng kim cô Tôn Ngộ Không đeo đáng giá bao nhiêu?
- Vị phi tần nào giành chiến thắng huy hoàng nhất nơi hậu cung nhà Thanh?
- Trả tự do cho Lưu Thiện, Tào Tháo nói 1 điều dự đoán vận mệnh của Thục Hán

Hưng Đạo Vương Trần Quốc Tuấn là người đã trực tiếp chỉ huy quân đội đánh tan hai cuộc xâm lược của quân Nguyên-Mông năm 1285 và năm 1288. Ông được biết đến với những chiến công lẫy lừng, là một nhà quân sự tài ba với gia thế hiển hách. Đứng là người giỏi, người tài nên cái gì ông cũng dám làm, dám chịu. Đau khổ vì người con gái mình yêu thương phải lấy người khác làm chồng, ông đã tạo ra một phi vụ cướp dâu chấn động nhất lịch sử Việt Nam.



GraphRAG

Prompt + LLMs



Graph Database lưu trữ trên [Neo4J](#)



```

Node(id='Hưng Đạo Vương Trần Quốc Tuấn', type='Person')
Node(id='quân Nguyên-Mông', type='Organization')
Node(id='Trần Cảnh', type='Person')
Node(id='Trần Thái Tông', type='Person')
Node(id='Lý Chiêu Hoàng', type='Person')
Node(id='công chúa Thuận Thiên', type='Person')
Node(id='Trần Liễu', type='Person')
Node(id='An Sinh Vương', type='Title')
Node(id='đất Yên Sinh', type='Location')
Node(id='Thủy Bà công chúa', type='Person')
Node(id='Thiên Thành công chúa', type='Person')
Node(id='Trung Thành Vương', type='Person')
Node(id='Nhân Đạo Vương', type='Person')
Node(id='Đại Việt sử ký toàn thư', type='Book')
Node(id='vương phủ Nhân Đạo Vương', type='Location')
Node(id='Thủy Bà công chúa', type='Person')
Node(id='Bảo Thánh Hoàng Hậu Trần Trinh', type='Person')
Node(id='Trần Nhân Tông', type='Person')
Node(id='Trần Anh Tông', type='Person')

```

Querying on a Graph Database

Querying on a Graph Database

Query

Mối quan hệ giữa Trần Cảnh và Lý Chiêu Hoàng là gì?

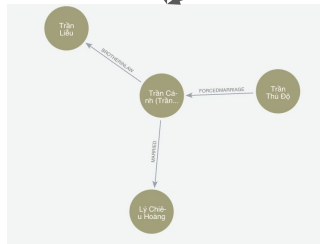
Chuyển thành
Neo4J query

```
MATCH (n {id: 'Trần Cảnh'})-[r]->(m)
RETURN 'Node ' + n.id + ' (label: ' +
labels(n)[0] + ') has relationship ' +
type(r) + ' with Node ' + m.id + ' (label:
' + labels(m)[0] + ')' AS Description
UNION
MATCH (n)->[r]-(m {id: 'Trần Cảnh'})
RETURN 'Node ' + m.id + ' (label: ' +
labels(m)[0] + ') has relationship ' +
type(r) + ' with Node ' + n.id + ' (label:
' + labels(n)[0] + ')' AS Description
```

```
MATCH (n {id: 'Lý Chiêu Hoàng'})-[r]->(m)
RETURN 'Node ' + n.id + ' (label: ' +
labels(n)[0] + ') has relationship ' +
type(r) + ' with Node ' + m.id + ' (label:
' + labels(m)[0] + ')' AS Description
UNION
MATCH (n)->[r]-(m {id: 'Lý Chiêu Hoàng'})
RETURN 'Node ' + m.id + ' (label: ' +
labels(m)[0] + ') has relationship ' +
type(r) + ' with Node ' + n.id + ' (label:
' + labels(n)[0] + ')' AS Description
```

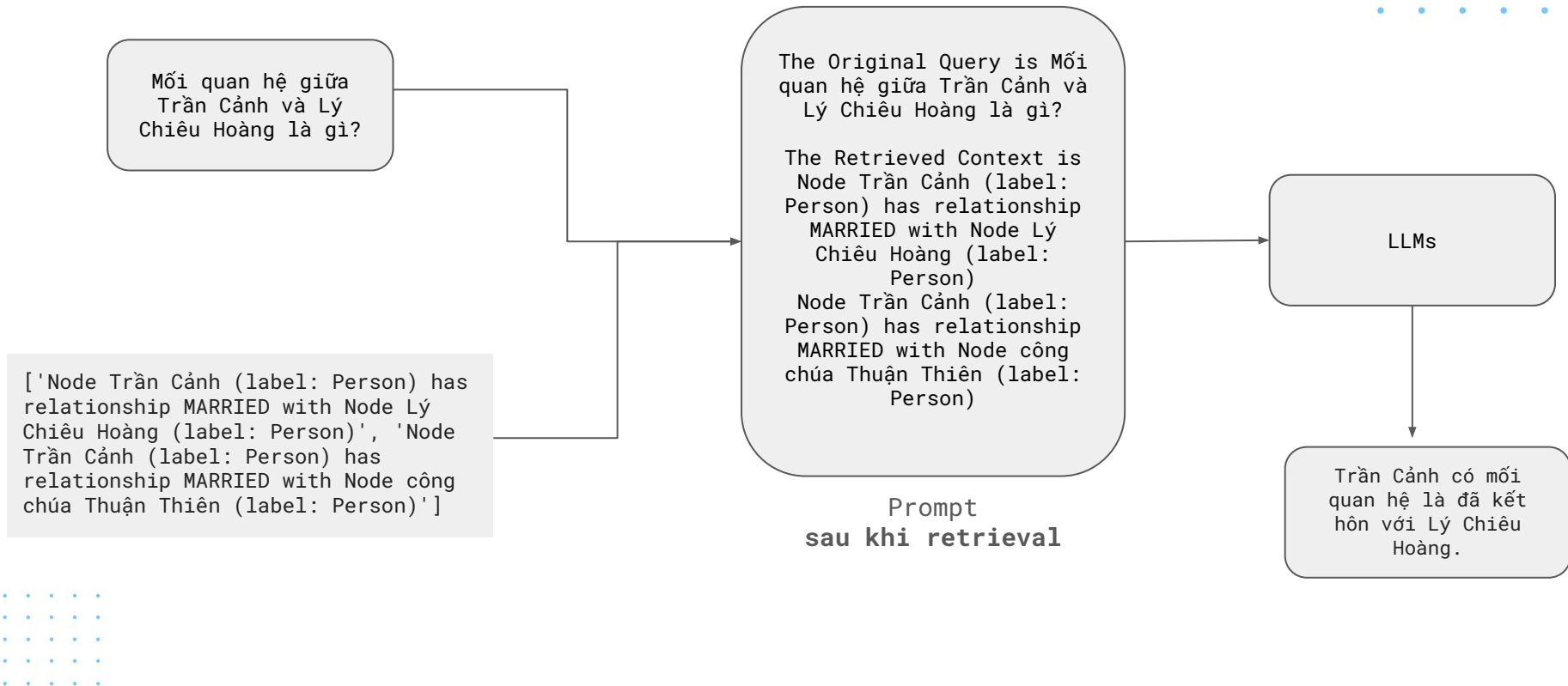
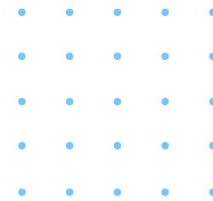
Query

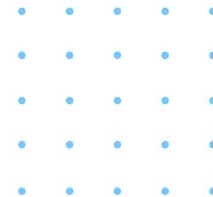
Query



Graph
Database

['Node Trần Cảnh (label: Person) has relationship MARRIED with Node Lý Chiêu Hoàng (label: Person)', 'Node Trần Cảnh (label: Person) has relationship MARRIED with Node công chúa Thuận Thiên (label: Person)']





Tạo Đồ Thị

Tạo Đồ Thị và lưu trữ vào DB

Trả Lời Truy Vấn

Từ Prompt chuyển thành truy vấn trên Graph Database và kết hợp kết quả này tạo thành prompt mới và đưa qua LLMs


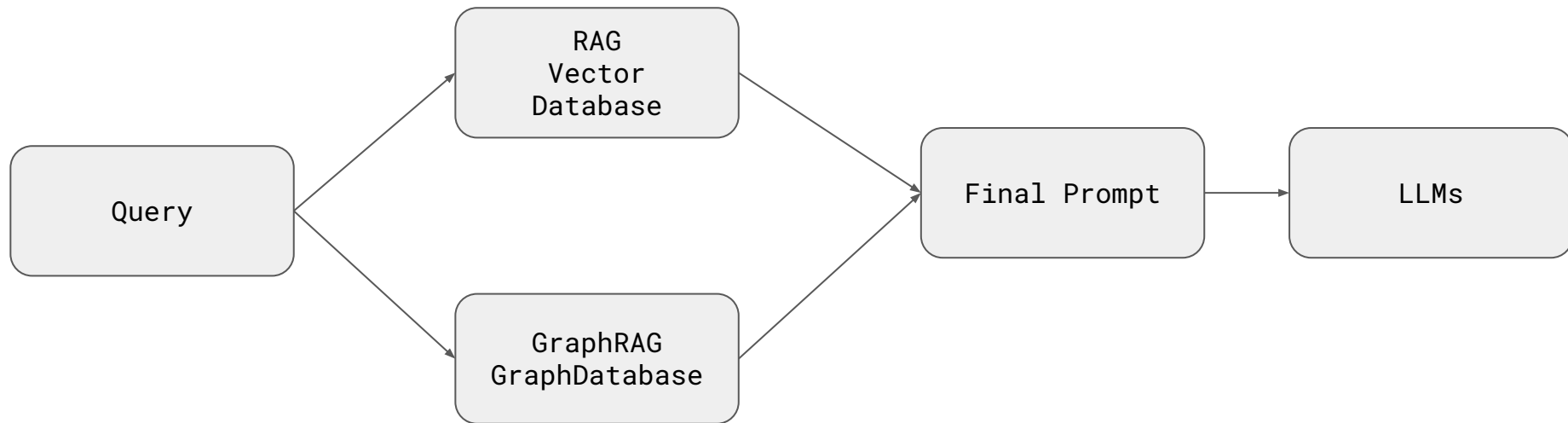
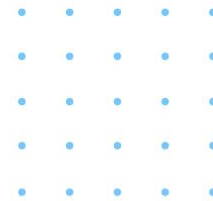
Code

GraphRAG + Neo4j + CamelAI



RAGGraph
+
Neo4J
+
CamelAI





Kết hợp kết quả của RAG qua **Vector Database** và RAG qua **Graph Database** để cho kết quả tốt nhất