```
!pip install          le -
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.4/50.4 kB 3.4 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 990.3/990.3 kB 25.0 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.3/2.3 MB 72.0 MB/s eta
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 293.5/293.5 kB 20.9 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 103.5/103.5 kB 8.2 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 377.3/377.3 kB 26.1 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.8/139.8 kB 11.3 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 5.6 MB/s et
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 77.9/77.9 kB 5.7 MB/s et
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.2/49.2 kB 3.8 MB/s et
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 141.1/141.1 kB 11.9 MB/s
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 4.6 MB/s et
```

```
## Graphdb configuration
NEO4J_URI="neo4j+s://b31
NEO4J_USERNAME="neo4j"
NEO4J_PASSWOR          w_k

```

```
import os
os.environ["NEO4J_URI"]=
os.environ["N          ERNA
os.environ["N          SSWO
```

```
from langchain_community
graph=Neo4jGraph(
    url=NEO4J_URI,
    username=NEO4J_USERN
    password=          ASSW
)
```

```
graph
```

```
<langchain_community.graphs.neo4j_graph.Neo4jGraph at
0x780895f33d60>
```

```
groq_api_key=" "
```

```python
from langchain_groq import ChatGroq

llm=ChatGroq(groq_api_key=groq_api_key,model_name="Gemma2-9b-It
llm
```

```
ChatGroq(client=<groq.resources.chat.completions.Completions object
at 0x780892c69870>, async_client=
<groq.resources.chat.completions.AsyncCompletions object at
0x780892c6a530>, model_name='Gemma2-9b-It',
groq_api_key=SecretStr('**********'))
```

```python
from langchain_core.documents import Document
text="""
Elon Reeve Musk (born June 28, 1971) is a businessman and inves
company SpaceX and automotive company Tesla, Inc. Other involve
formerly Twitter, and his role in the founding of The Boring Co
He is one of the wealthiest people in the world; as of July 202
US$221 billion.Musk was born in Pretoria to Maye and engineer E
the University of Pretoria before immigrating to Canada at age 
his Canadian-born mother. Two years later, he matriculated at Q
Musk later transferred to the University of Pennsylvania and re
 and physics. He moved to California in 1995 to attend Stanford
  two days and, with his brother Kimbal, co-founded online city
 """
documents=[Document(page_content=text)]
documents
```

```
[Document(page_content="\nElon Reeve Musk (born June 28, 1971) is a
businessman and investor known for his key roles in space\ncompany
SpaceX and automotive company Tesla, Inc. Other involvements
include ownership of X Corp.,\nformerly Twitter, and his role in
the founding of The Boring Company, xAI, Neuralink and OpenAI. \nHe
is one of the wealthiest people in the world; as of July 2024,
Forbes estimates his net worth to be \nUS$221 billion.Musk was born
in Pretoria to Maye and engineer Errol Musk, and briefly attended
\nthe University of Pretoria before immigrating to Canada at age
18, acquiring citizenship through \nhis Canadian-born mother. Two
years later, he matriculated at Queen's University at Kingston in
Canada.\nMusk later transferred to the University of Pennsylvania
and received bachelor's degrees in economics\n and physics. He
moved to California in 1995 to attend Stanford University, but
dropped out after\n  two days and, with his brother Kimbal, co-
founded online city guide software company Zip2. \n ")]
```

```
!pip install --upgrade --quiet langchain_experimental
```

```
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 203.2/203.2 kB 5.2 MB/s
```

```python
from langchain_experimental.graph_transformers import LLMGraphT
llm_transformer=LLMGraphTransformer(llm=llm)
```

```python
graph_documents=llm_transformer.convert_to_graph_documents(docu
```

```python
graph_documents
```

```
[GraphDocument(nodes=[Node(id='Elon Reeve Musk', type='Person'),
Node(id='Maye', type='Person'), Node(id='Errol Musk',
type='Person'), Node(id='Pretoria', type='Location'),
Node(id='University Of Pretoria', type='Educationalinstitution'),
Node(id='Canada', type='Location'), Node(id="Queen'S University At
Kingston", type='Educationalinstitution'), Node(id='University Of
Pennsylvania', type='Educationalinstitution'),
Node(id='California', type='Location'), Node(id='Stanford
University', type='Educationalinstitution'), Node(id='Zip2',
type='Company'), Node(id='Spacex', type='Company'), Node(id='Tesla,
Inc.', type='Company'), Node(id='X Corp.', type='Company'),
Node(id='Twitter', type='Company'), Node(id='The Boring Company',
type='Company'), Node(id='Xai', type='Company'),
Node(id='Neuralink', type='Company'), Node(id='Openai',
type='Company'), Node(id='Forbes', type='Organization')],
relationships=[Relationship(source=Node(id='Elon Reeve Musk',
type='Person'), target=Node(id='Maye', type='Person'),
type='PARENT'), Relationship(source=Node(id='Elon Reeve Musk',
type='Person'), target=Node(id='Errol Musk', type='Person'),
type='PARENT'), Relationship(source=Node(id='Elon Reeve Musk',
type='Person'), target=Node(id='Pretoria', type='Location'),
type='BORN_IN'), Relationship(source=Node(id='Elon Reeve Musk',
type='Person'), target=Node(id='University Of Pretoria',
type='Educationalinstitution'), type='ATTENDED'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Canada', type='Location'), type='IMMIGRATED_TO'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id="Queen'S University At Kingston",
type='Educationalinstitution'), type='ATTENDED'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='University Of Pennsylvania',
type='Educationalinstitution'), type='ATTENDED'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='California', type='Location'), type='MOVED_TO'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Stanford University',
type='Educationalinstitution'), type='ATTENDED'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Kimbal', type='Person'), type='SIBLING'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Zip2', type='Company'), type='FOUNDED'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Spacex', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Tesla, Inc.', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='X Corp.', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='X Corp.', type='Company'),
target=Node(id='Twitter', type='Company'),
type='IS_FORMER_NAME_OF'), Relationship(source=Node(id='Elon Reeve
Musk', type='Person'), target=Node(id='The Boring Company',
type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Xai', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Neuralink', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Openai', type='Company'), type='INVOLVED_IN'),
Relationship(source=Node(id='Forbes', type='Organization'),
```

```
graph_documents[0].nodes
```

```
[Node(id='Elon Reeve Musk', type='Person'),
 Node(id='Maye', type='Person'),
 Node(id='Errol Musk', type='Person'),
 Node(id='Pretoria', type='Location'),
 Node(id='University Of Pretoria', type='Educationalinstitution'),
 Node(id='Canada', type='Location'),
 Node(id="Queen'S University At Kingston",
type='Educationalinstitution'),
 Node(id='University Of Pennsylvania',
type='Educationalinstitution'),
 Node(id='California', type='Location'),
 Node(id='Stanford University', type='Educationalinstitution'),
 Node(id='Zip2', type='Company'),
 Node(id='Spacex', type='Company'),
 Node(id='Tesla, Inc.', type='Company'),
 Node(id='X Corp.', type='Company'),
 Node(id='Twitter', type='Company'),
 Node(id='The Boring Company', type='Company'),
 Node(id='Xai', type='Company'),
 Node(id='Neuralink', type='Company'),
 Node(id='Openai', type='Company'),
 Node(id='Forbes', type='Organization')]
```

```
graph_documents[0].relationships
```

```
[Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Maye', type='Person'), type='PARENT'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Errol Musk', type='Person'), type='PARENT'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Pretoria', type='Location'), type='BORN_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='University Of Pretoria',
type='Educationalinstitution'), type='ATTENDED'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Canada', type='Location'), type='IMMIGRATED_TO'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id="Queen'S University At Kingston",
type='Educationalinstitution'), type='ATTENDED'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='University Of Pennsylvania',
type='Educationalinstitution'), type='ATTENDED'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='California', type='Location'), type='MOVED_TO'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Stanford University',
type='Educationalinstitution'), type='ATTENDED'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Kimbal', type='Person'), type='SIBLING'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Zip2', type='Company'), type='FOUNDED'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Spacex', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Tesla, Inc.', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='X Corp.', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='X Corp.', type='Company'),
target=Node(id='Twitter', type='Company'),
type='IS_FORMER_NAME_OF'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='The Boring Company', type='Company'),
type='INVOLVED_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Xai', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Neuralink', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='Elon Reeve Musk', type='Person'),
target=Node(id='Openai', type='Company'), type='INVOLVED_IN'),
 Relationship(source=Node(id='Forbes', type='Organization'),
target=Node(id='Elon Reeve Musk', type='Person'),
type='ESTIMATED_NET_WORTH')]
```

```
### Load the dataset of movie

movie_query="""
LOAD CSV WITH HEADERS FROM
'https://raw.githubusercontent.com/tomasonjo/blog-datasets/main

MERGE(m:Movie{id:row.movieId})
SET m.released = date(row.released),
    m.title = row.title,
    m.imdbRating = toFloat(row.imdbRating)
FOREACH (director in split(row.director, '|') |
    MERGE (p:Person {name:trim(director)})
    MERGE (p)-[:DIRECTED]->(m))
FOREACH (actor in split(row.actors, '|') |
    MERGE (p:Person {name:trim(actor)})
    MERGE (p)-[:ACTED_IN]->(m))
FOREACH (genre in split(row.genres, '|') |
    MERGE (g:Genre {name:trim(genre)})
    MERGE (m)-[:IN_GENRE]->(g))
"""
```

```
graph
```

```
<langchain_community.graphs.neo4j_graph.Neo4jGraph at
0x780895f33d60>
```

```
graph.query(movie_query)
```

```
[]
```

```
graph.refresh_schema()
print(graph.schema)
```

```
Node properties:
Person {born: INTEGER, name: STRING}
Movie {title: STRING, released: INTEGER, id: STRING, imdbRating: FLO
Genre {name: STRING}
Relationship properties:

The relationships:
(:Person)-[:ACTED_IN]->(:Movie)
(:Person)-[:DIRECTED]->(:Movie)
(:Movie)-[:IN_GENRE]->(:Genre)
```

```
from langchain.chains import GraphCypherQAChain
chain=GraphCypherQAChain.from_llm(llm=llm,graph=graph,verbose=T
chain
```

```
GraphCypherQAChain(verbose=True, graph=
<langchain_community.graphs.neo4j_graph.Neo4jGraph object at
0x780895f33d60>,
cypher_generation_chain=LLMChain(prompt=PromptTemplate(input_variabl
['question', 'schema'], template='Task:Generate Cypher statement to
query a graph database.\nInstructions:\nUse only the provided
relationship types and properties in the schema.\nDo not use any
other relationship types or properties that are not
provided.\nSchema:\n{schema}\nNote: Do not include any explanations
or apologies in your responses.\nDo not respond to any questions
that might ask anything else than for you to construct a Cypher
statement.\nDo not include any text except the generated Cypher
statement.\n\nThe question is:\n{question}'), llm=ChatGroq(client=
<groq.resources.chat.completions.Completions object at
0x780892c69870>, async_client=
<groq.resources.chat.completions.AsyncCompletions object at
0x780892c6a530>, model_name='Gemma2-9b-It',
groq_api_key=SecretStr('**********'))),
qa_chain=LLMChain(prompt=PromptTemplate(input_variables=['context',
'question'], template="You are an assistant that helps to form nice
and human understandable answers.\nThe information part contains
the provided information that you must use to construct an
answer.\nThe provided information is authoritative, you must never
doubt it or try to use your internal knowledge to correct it.\nMake
the answer sound as a response to the question. Do not mention that
you based the result on the given information.\nHere is an
example:\n\nQuestion: Which managers own Neo4j stocks?\nContext:
[manager:CTL LLC, manager:JANE STREET GROUP LLC]\nHelpful Answer:
CTL LLC, JANE STREET GROUP LLC owns Neo4j stocks.\n\nFollow this
example when generating answers.\nIf the provided information is
empty, say that you don't know the
answer.\nInformation:\n{context}\n\nQuestion: {question}\nHelpful
Answer:"), llm=ChatGroq(client=
<groq.resources.chat.completions.Completions object at
0x780892c69870>, async_client=
<groq.resources.chat.completions.AsyncCompletions object at
0x780892c6a530>, model_name='Gemma2-9b-It',
groq_api_key=SecretStr('**********'))), graph_schema='Node
properties are the following:\nPerson {born: INTEGER, name:
STRING},Movie {title: STRING, released: INTEGER, id: STRING,
imdbRating: FLOAT},Genre {name: STRING}\nRelationship properties
are the following:\n\nThe relationships are the
following:\n(:Person)-[:ACTED_IN]->(:Movie),(:Person)-[:DIRECTED]->
(:Movie),(:Movie)-[:IN_GENRE]->(:Genre)')
```

```
response=chain.invoke({"query":"Who was the director of the mov

response

```

```
> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (m:Movie {title:"GoldenEye"})<-[:DIRECTED]-(p:Person) RETURN p

Full Context:
[{'p.name': 'Martin Campbell'}]

> Finished chain.
{'query': 'Who was the director of the moview GoldenEye',
 'result': 'Martin Campbell  \n'}
```

```
response=chain.invoke({"query":"tell me the genre of th movie G

response
```

```
> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (m:Movie {title: "GoldenEye"})-[:IN_GENRE]->(g:Genre) RETURN g

Full Context:
[{'g.name': 'Adventure'}, {'g.name': 'Action'}, {'g.name': 'Thriller

> Finished chain.
{'query': 'tell me the genre of th movie GoldenEye',
 'result': 'Adventure, Action, Thriller \n'}
```

```
response=chain.invoke({"query":"Who was the director in movie C

response
```

```


> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (m:Movie{title:"Casino"})<-[:DIRECTED]-(p:Person)
RETURN p.name

Full Context:
[{'p.name': 'Martin Scorsese'}]

> Finished chain.
{'query': 'Who was the director in movie Casino',
 'result': 'Martin Scorsese  \n'}
```

```
response=chain.invoke({"query":"Which movie were released in 20

response
```

```


> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (m:Movie) WHERE m.released = 2008 RETURN m.title

Full Context:
[{'m.title': 'Ironman'}]

> Finished chain.
{'query': 'Which movie were released in 2008',
 'result': "I don't know the answer. \n"}
```

```
response=chain.invoke({"query":"Give me the list of movie havin
response
```

```


> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (m:Movie) WHERE m.imdbRating > 8 RETURN m.title

Full Context:
[{'m.title': 'Toy Story'}, {'m.title': 'Heat'}, {'m.title': 'Casino'

> Finished chain.
{'query': 'Give me the list of movie having imdb rating more than
8',
 'result': "I don't know the answer. \n"}
```

```
examples = [
    {
        "question": "How many artists are there?",
        "query": "MATCH (a:Person)-[:ACTED_IN]->(:Movie) RETURN
    },
    {
        "question": "Which actors played in the movie Casino?",
        "query": "MATCH (m:Movie {{title: 'Casino'}})<-[:ACTED_
    },
    {
        "question": "How many movies has Tom Hanks acted in?",
        "query": "MATCH (a:Person {name: 'Tom Hanks'})-[:ACTED_
    },
    {
        "question": "List all the genres of the movie Schindler
        "query": "MATCH (m:Movie {{title: 'Schindler\\'s List'}
    },
    {
        "question": "Which actors have worked in movies from bo
        "query": "MATCH (a:Person)-[:ACTED_IN]->(:Movie)-[:IN_G
    },
    {
        "question": "Which directors have made movies with at l
        "query": "MATCH (d:Person)-[:DIRECTED]->(m:Movie)<-[:AC
    },
    {
        "question": "Identify movies where directors also playe
        "query": "MATCH (p:Person)-[:DIRECTED]->(m:Movie), (p)-
    },
    {
        "question": "Find the actor with the highest number of
        "query": "MATCH (a:Actor)-[:ACTED_IN]->(m:Movie) RETURN
    },
]
```