Neo4j Database with Cypher Queries using Python Documentation

Table of Contents

1. Introduction

This documentation provides a step-by-step guide on using Neo4j, a graph database, with Cypher queries in Python. The document assumes you have imported the movie dataset into a Neo4j Sandbox and aims to establish a connection with your local Jupyter Notebook using Python.

2. Setup

2.1 Neo4j Sandbox

Neo4j Sandbox is a cloud-based environment provided by Neo4j that allows users to explore and experiment with Neo4j without the need for local installation. It provides a pre-configured Neo4j instance with sample datasets. Follow these steps:

Create a Neo4j Sandbox account.
Import the movie dataset into the Neo4j Sandbox.

2.2 Local Environment

Before connecting to the Neo4j Sandbox, ensure you have Neo4j and the py2neo library installed in your local Python environment. Install py2neo using the following command:

```
pip install py2neo
```

3. Establishing Connection

In your Jupyter Notebook or Python script, use the following code to establish a connection:

```
from py2neo import Graph

# Neo4j Sandbox credentials
url = "bolt://52.55.96.244:7687"
pwd = "spindle-sidewalks-armory"

# Establishing connection
graph = Graph(url, auth=("neo4j", pwd))
```

Replace the url and pwd variables with the actual credentials provided by your Neo4j Sandbox.

4. Running Cypher Queries

4.1 Basic Cypher Queries

Once the connection is established, you can run Cypher queries on the Neo4j database. Here's an example of a basic Cypher query:

```
# Example Cypher query
query = """
MATCH (m:Movie)
RETURN m.title AS MovieTitle, m.release_year AS ReleaseYear
LIMIT 5
"""

# Running the query
result = graph.run(query)

# Displaying the result
for record in result:
    print(record)
```

Modify the query variable to suit your specific needs.

4.2 Node and Relationship Concepts

In Cypher queries, nodes are represented using parentheses (), and relationships are represented using arrows -->. For example:

```
# Example Cypher query with nodes and relationships
query = """
MATCH (actor:Person)-[:ACTED_IN]->(movie:Movie)
WHERE actor.name = 'Tom Hanks'
RETURN actor, movie
"""

# Running the query
result = graph.run(query)

# Displaying the result
for record in result:
    print(record)
```

This query retrieves actors and movies connected by the "ACTED_IN" relationship, specifically for the actor with the name 'Tom Hanks'.

5. Conclusion

This documentation guides you through the process of connecting to a Neo4j database using Python and running Cypher queries. It also introduces basic concepts of nodes and relationships in Cypher queries, allowing you to explore and analyze graph data in your local Jupyter Notebook environment.

https://www.loom.com/share/d06a7f4c7c2048c0ba813040a1e9ece2