THE UNIVERSITY OF SYDNEY

Comparing Graph Query Languages

Yuemeng Yin, School of Computer Science, the University of Sydney

Introduction

Graph data plays a vital role in numerous applications, leading to various proposed query languages for access.

This exploratory project will compare different approaches, by taking sample data schema and writing queries in various languages to extract the same information.

Materials and methods

Firstly, we queried two public GraphQL APIs. Next, we explored more graph-oriented queries and operations, aiming to express them in both GraphQL and Cypher.

Through this comparison and analysis, we aim to understand the capabilities of each language and identify potential limitations in utilizing GraphQL for complex graph queries.

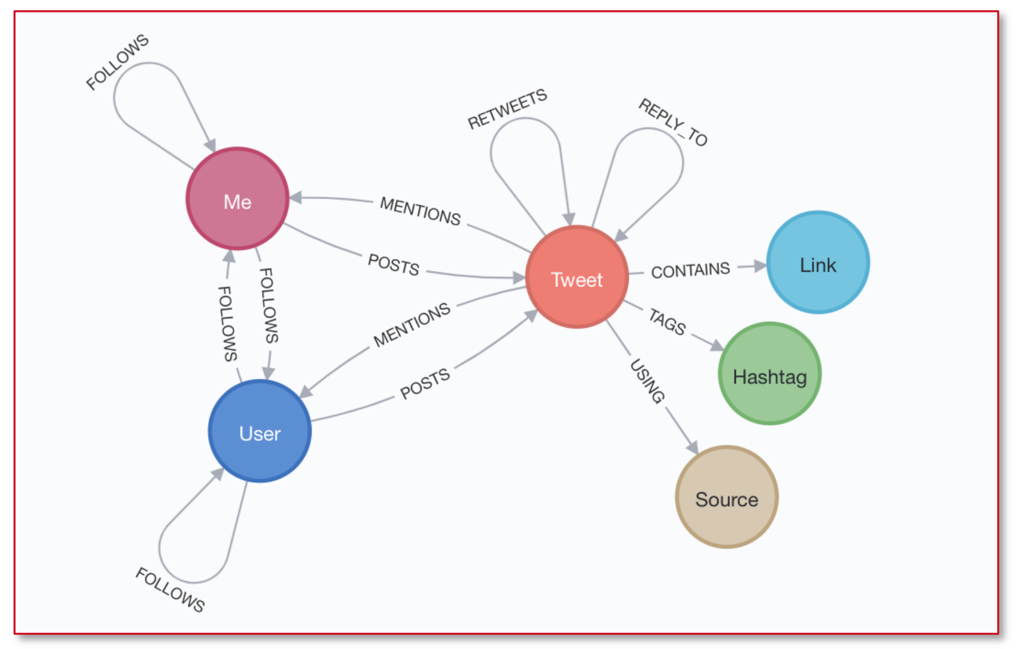
Results

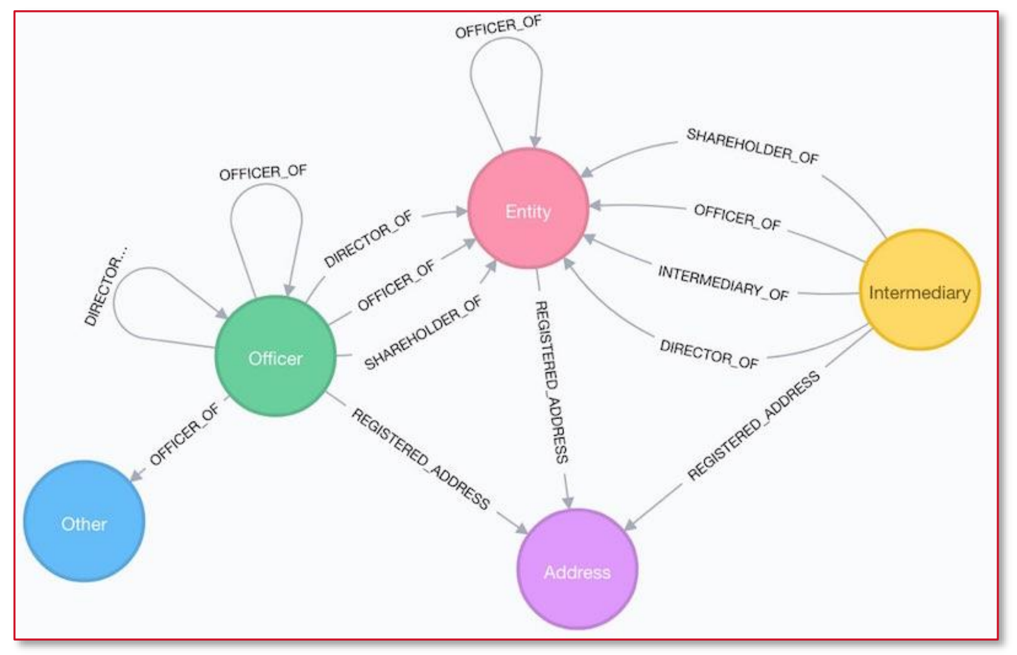
The Results section of the investigation comprises three sub-sections, each revealing essential findings and insights:

1. **"Finding from Expressing User Requirements in GraphQL Queries":**

In this part, the expressive capabilities of GraphQL were explored by expressing different levels of user requirements. The key observations include:

- GraphQL offers a flexible and intuitive approach to retrieve specific data fields and connections for each node in the graph.
- However, expressing aggregates, ordering, limits, or similar operations in GraphQL can be challenging.





"On one hand you want to take full advantage of GraphQL's graph traversal features ("show me the authors of the comments of the author of the post of ..." etc.). But on the other hand, you don't want your app to become slow and unresponsive.
—Sacha Greif."

2. **"Findings from Comparison of Cypher and GraphQL in Neo4j Graph Databases":**

This section compared Cypher and GraphQL using Neo4j graph databases for a social network dataset and the Offshore Leaks Database (relationships graph as shown above). Notable findings include:

- Cypher excels in expressing complex graph queries efficiently and concisely due to native support for variable-length path patterns.
- Queries involving non-deterministic path lengths may lead to deeply nested and cumbersome queries in GraphQL.

3. **"Summarized Findings and Identified Strengths, Limitations, and Potential Extensions":**

The overall conclusions and identified strengths and limitations of GraphQL are summarized. Key points include:.

- Potential extensions to GraphQL, such as introducing additional language features or directives, could enhance its support for graph-oriented operations. Through GraphQL schema directives, we can use the power of Cypher with GraphQL to map a GraphQL field to the result of an arbitrary Cypher query.

Conclusions

This project presents a comparison between GraphQL and Cypher for graph-oriented queries, shedding light on GraphQL's limitations, particularly concerning tasks involving non-deterministic path lengths.

Future work can extend the comparison to include other graph query languages, like G-CORE , for a more comprehensive evaluation of GraphQL's capabilities against various alternatives.

Addressing GraphQL's limitations and exploring ways to enable graph-oriented queries while maintaining backward compatibility with existing systems will empower users to:

- seamlessly transition to an enhanced GraphQL implementation,
- And benefiting from its graph query capabilities in an idiomatic and intuitive manner.

Literature cited

[1] Peter T. Wood. Query languages for graph databases. SIGMOD Rec., 41(1):50–60, apr 2012.
[2] Renzo Angles, Marcelo Arenas, Pablo Barcel´o, Aidan Hogan, Juan Reutter, and Domagoj Vrgoˇc. Foundations of modern query languages for graph databases. ACM Comput. Surv., 50(5), sep 2017.
[3] Antonio Quĩna Mera, Pablo Fernandez, Jos´e Mar´ıa Garc´ıa, and Antonio Ruiz-Cort´es. Graphql: A systematic mapping study. ACM Comput. Surv., 55(10), feb 2023.
[4] Thomas Frisendal. Visual design of Graphql Data: A practical introduction with legacy data and neo4j. Apress, 2018.

Acknowledgments

Thanks to Professor Alan Fekete and Dr Akon Dey for their support throughout the project. This work was created as part of the VRI Winter Vacation Program 2023 at the University of Sydney, Australia.

Further information

Contact details:

Yuemeng Yin
Yuemeng.yin@student.unsw.edu.au