March 2019 W3C workshop in Berlin on graph data management standards

Towards an International Standard for the GQL Graph Query Language

by Stefan Plantikow, Language Standards Engineer and Product Manager at Neo4j (for the Neo4j Query Languages Team)

1 Biography

Stefan is a language standards engineer and product manager for the Cypher implementation at Neo4j and has a research background in distributed fault-tolerant systems and transaction processing.

Stefan is passionate about computer language design and how languages as a medium enable access to new technology. As a principal designer of the Cypher language at Neo4j, he built the first cost-based query planner for property graph databases, pioneered the architecture of the first big data implementation of Cypher called Cypher for Apache Spark, and helped introducing labels to the property graph world. He is now working on the creation of an International Standard of the GQL Graph Query Language and property graph extensions to SQL by acting as Neo4j representative to the US national body (INCITS DM32.2) as well as the SQL Standards Committee (ISO JTC1/SC32 WG3).

Stefan has a degree in computer science from Humboldt University, Berlin. He is still based in Berlin and nowadays continuously works on expanding the scope, the ease of use, and the applicability of graph technology for users.

2 GQL Graph Query Language

2.1 Background

Following the growing interconnectedness of data, property graphs are becoming increasingly popular as a primary data model for enterprise applications. Their success rests on the combination of (1) an intuitive ("whiteboard-friendly") model that is geared towards application developers, (2) support for rapid prototyping with schema-optionality (3) availability of native graph databases and (4) declarative query languages like openCypher and PGQL that focus on ease of use via graph pattern matching with visual ("ascii-art") syntax, top-down linear statement order, and language composability. However, the availability of many languages has also led to some fragmentation, to the detriment of users. The SQL Standards Committee (ISO JTC1/SC32 WG3) acknowledges these developments around the property graph model and is actively working on adding property graph extensions to SQL:2020 to accommodate them.

While these property graph extensions to SQL:2020 will provide existing RDBMS with standardized graph pattern matching over table-projected graphs, they do not fulfill the vision of the GQL Manifesto:
A composable declarative graphs-first query language based on pattern matching and graph operations.

2.2 GQL Scope and Features

The property graph community is pursuing the creation of a new graph query language that combines the best ideas of existing languages, incorporates new ideas from research (e.g. G-Core, Cypher 10) and next-generation implementations (e.g. Neo4j Morpheus), and is being built on the same foundations as SQL. This has mainly been initiated by representatives from industry, academia, and the open source community and is being pursued via the Ad Hoc on SQL Extensions Property Graphs, the GQL Community event calls and actively supported by the SQL Standards Committee.

This process has now begun to explore both the scope of features for GQL as well as pursue initial discussions on syntax and semantics. Neo4j has submitted a substantial proposal (of which I am a primary author) that gives initial answers to these questions and is called "*GQL Scope and Features*". This proposal focuses on

(1) pattern matching and essential language features,
(2) composition of parameterized queries, e.g. for the construction and consumption of whole graphs and graph views (making GQL closed under graphs)
(3) graph typing,
(4) schema-optionality,
(5) graph updates,
(6) language interoperability, security models, error handling, and similar concerns.

This proposal will have been discussed by the SQL Standards Committee at the time of the W3C workshop.

2.3 Topic Proposal

I'd like to present these ongoing developments around the creation of an International Standard for the GQL Graph Query Language, the features detailed in "*GQL Scope and Features*", and the state of the discussion around this and other proposals to the audience of the W3C workshop.

I'd like to facilitate a discussion that focuses on

- feedback on the direction of the presented features and the scope of GQL,
- opportunities for interoperability and synergy between GQL and other languages (esp. SPARQL),
- mapping between the property graph data model and other data models,
- other aspects such as the relationship between graph views and reasoning, leveraging computational statistics for graph analytics, integration of graph computational models and artificial intelligence

Furthermore, I'm interested in fostering links between the Semantic Web and GQL communities, and exploring research and implementation opportunities around GQL.

3 Other Topics of Interest

Path pattern languages and matching semantics. I'm interested in recent discussion on improving the power of pattern matching and exploring alternative pattern matching semantics (configurable morphism, bisimilar matching, context free pattern languages, approximate matching and costing).

Comparative analysis of graph languages. The field of graph computational and query languages recently has seen an increase in new approaches and technologies. I'm interested in presentations and discussions that provide an overview of trends and emerging technologies.

Graph view construction. Having been involved in the topic of graph view construction in the context of Cypher 10 language designs and G-Core, I would like to see continued discussion on the use of views for reasoning, foreign data model integration, and application security.

Multi-model integration. The rise of multiple data models (graph, table, nested, column, ...) and different data type systems (RDF, SQL, JSON) makes integration and conversion an important issue. In particular, I'm interested in approaches for bridging between Semantic Web and property graph data models but also in how to better integrate nested data into the database world.

(926 words)

4 References and Links

[1] GQL Scope and Features, Revision 3, https://s3.amazonaws.com/artifacts.opencypher.org/website/materials/sql-pg-2018-0046r3-GQL-Scope-and-Features.pdf
[2] GQL Manifesto, https://gql.today/
[3] The GQL Standards Website, https://www.gqlstandards.org/
[4] The Cypher 9 Language Reference, openCypher, "Cypher Query Language Reference", http://s3.amazonaws.com/artifacts.opencypher.org/openCypher9.pdf, Mar 2018, Originally published Sep 2017
[5] R. Angles, M. Arenas, P. Barcelo, P. Boncz, G. Fletcher, C. Gutierrez, T. Lindaaker, M. Paradies, S. Plantikow, J. Sequeda, O. van Rest, H. Voigt, "G-CORE: A Core for Future Graph Query Languages", https://arxiv.org/pdf/1712.01550.pdf, Dec 2017
[6] N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, M. Rydberg, M. Schuster, P. Selmer, A. Taylor, "Formal Semantics of the Language Cypher", https://arxiv.org/pdf/1802.09984.pdf, Mar 2018
[7] O. van Rest, "PGQL 1.1 Specification",http://pgql-lang.org/spec/1.1/, Sep 2017
[8] ISO JTC1/SC32 (parent committee of SQL standards committee), https://www.iso.org/committee/45342.html