GQL Implementation WG

Eighteenth TUC Meeting

Michael Burbidge, August 30, 2024

Mission

Create tooling and documentation to assist in and accelerate the implementation and adoption of GQL

Working group outputs

Output	Original timeframe	Status
ANTLR grammar	May 2024	In progress
Railroad diagrams	May 2024	Complete
Technology Compatibility Kit	December 2024	Planning
Specification feedback mechanism	February 2024	Complete
Technical report	First draft June 2024, second December	In progress

ANTLR Grammar

Interim

- Repository
- Drives Code Editor and railroad diagrams

ANTLR Grammar

Strumenta

- Kickoff meeting with Strumenta
  - Shared what we've learned developing our grammar
- They have built a candidate grammar
- Weekly updates on their progress

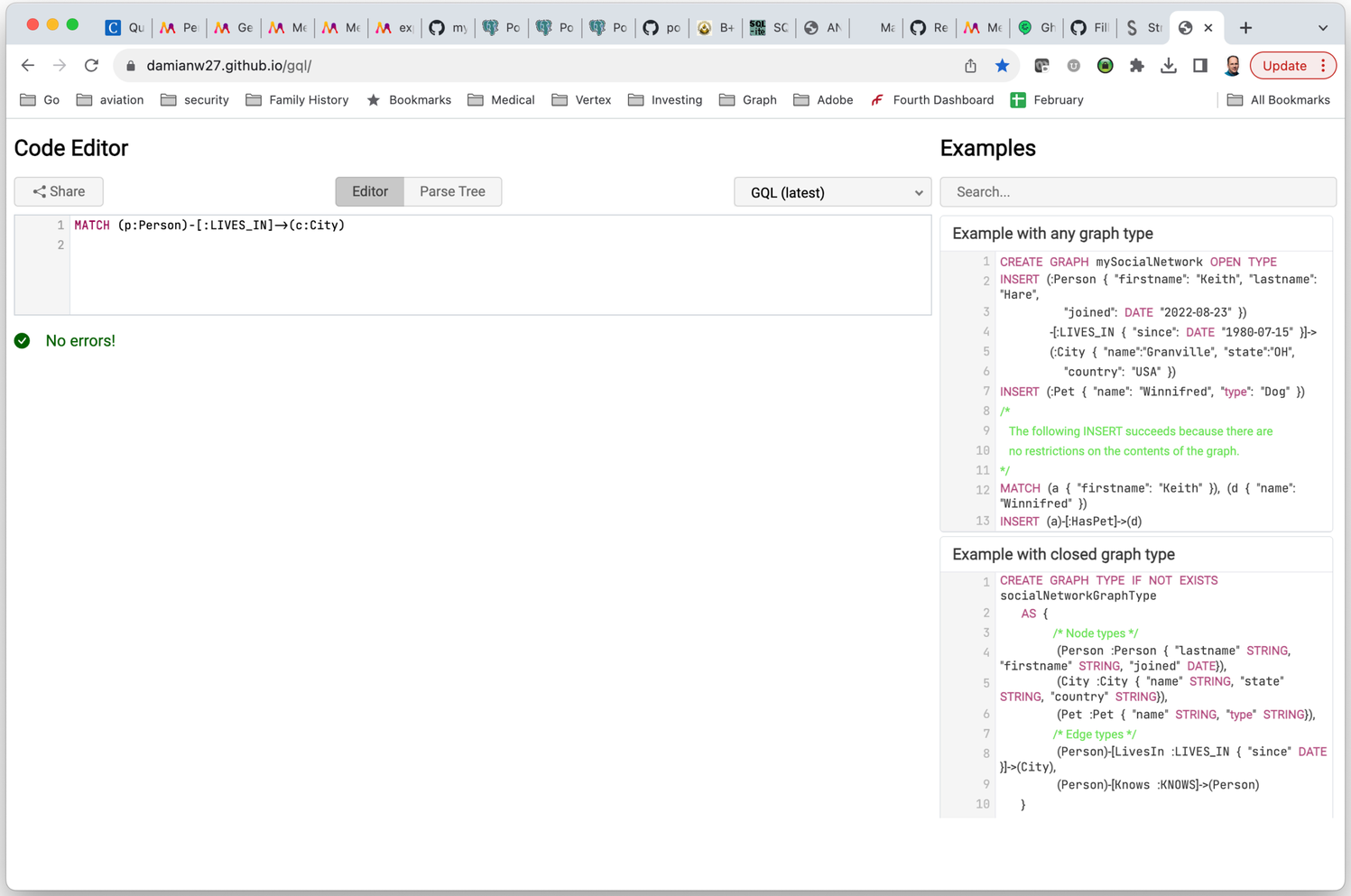
ANTLR Grammar

Strumenta video

ANTLR Grammar

Code Editor

- Automated deployment fixes
- Editor bug fixes
- Autocomplete



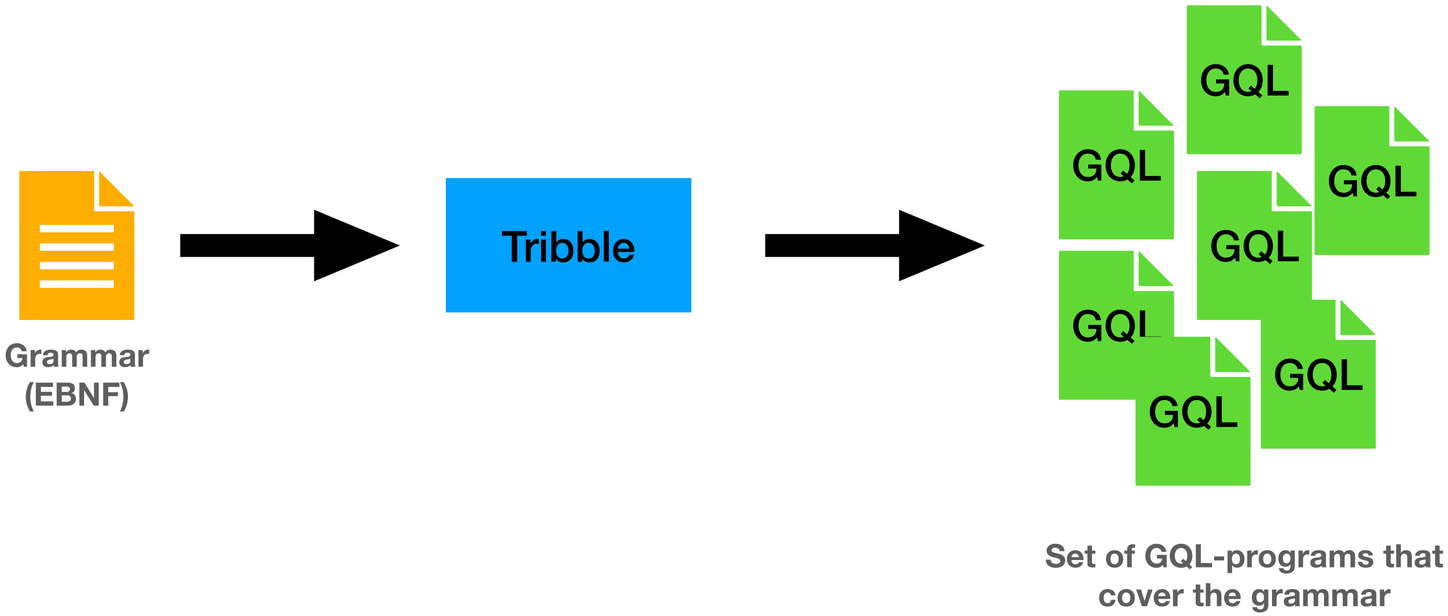
ANTLR Grammar

Grammar-based Fuzzer

- Systematically Covering Input Structure
- Implementation: Tribble (fork that contains GQL tribble grammar)
- Solves two problems
  - Limits recursion
  - Covers the grammar

ANTLR Grammar

Grammar-based Fuzzer



ANTLR Grammar

Grammar-based Fuzzer

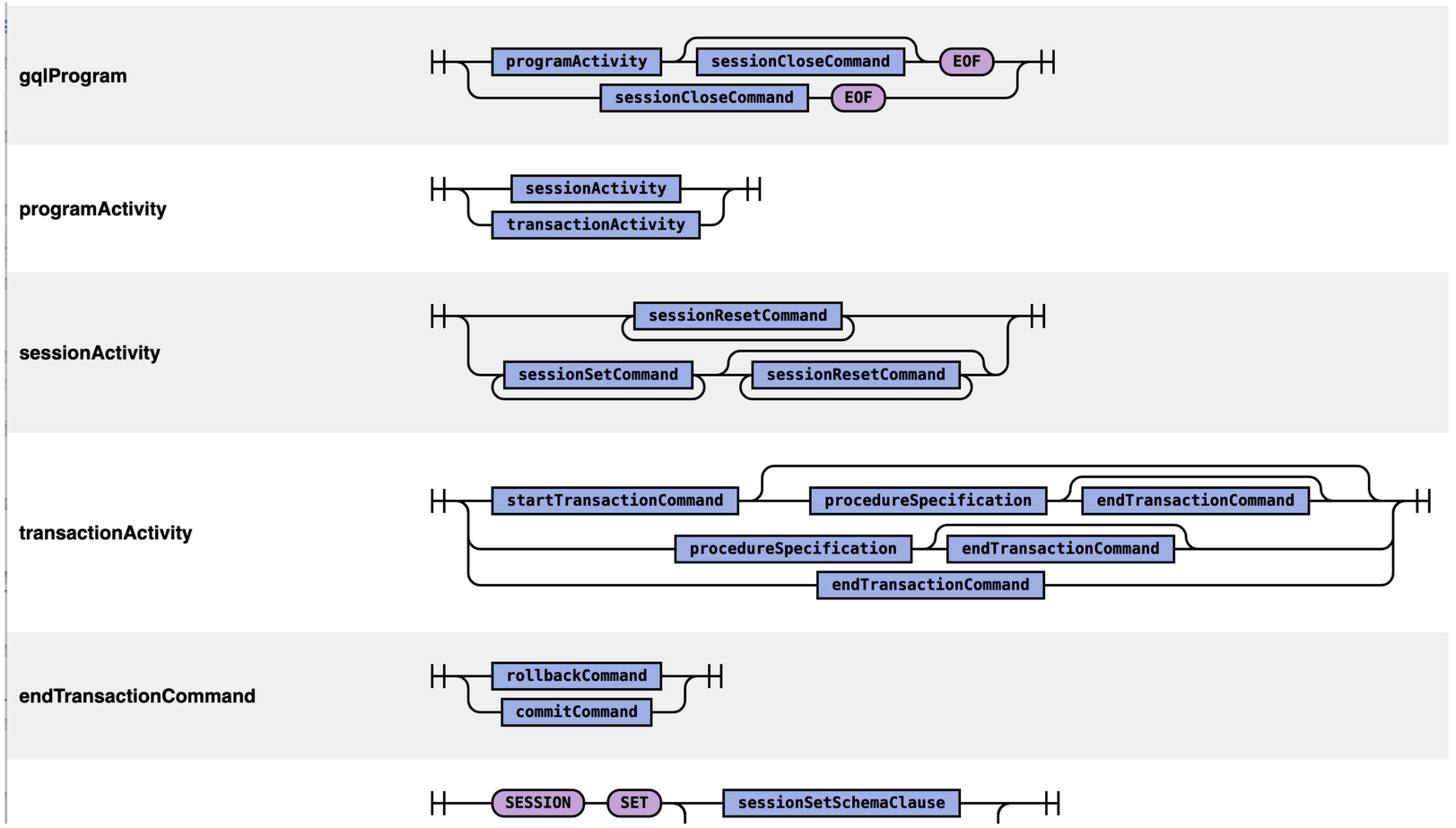
ANTLR Grammar

Grammar-based Fuzzer

- Running tribble with a k-path of 5 generates 14K GQL-programs
- Among other things, k-path limits recursion
- Greater values for k-path, result in more recursion

Railroad diagram

- Railroad diagrams
- Driven by ANTLR grammar
- Link-based navigation
- Back button support
- Auto-deployment



GQL Standard feedback mechanism

- Google Doc: LDBC GQL Improvements proposals (GQL-IP)
- Tracks improvement proposals and their status
- Regularly monitored and reviewed by ISO/WG3

Technical Report

Two publications

- 6 - 8 page summary of GQL
  - Target publication: Communications of the ACM
  - Draft available mid-september
- 25 - 35 page in depth summary of GQL
  - Target publication: Transactions of Databases, LDBC Website
  - Example-driven
  - In progress

GQL Technology Compatibility Kit (TCK)

openGQL relationship to openCypher

- Neo4j roadmap for openCypher
  - openCypher is the road to GQL
  - openCypher 9 will be frozen
  - GQL features will gradually be incorporated into openCypher, via the openCypher CIP process
  - No new non-GQL CIPs will be accepted
  - Eventually openCypher will have incorporated the whole of the GQL language

GQL Technology Compatibility Kit (TCK)

openGQL relationship to openCypher

- Neo4j is the primary driver/owner of openCypher
- Its likely that GQL features will be sequenced in a way that makes sense for existing Neo4j customers
- This is a good thing…more people using GQL features drives GQL adoption

GQL Technology Compatibility Kit (TCK)

openGQL

- Some vendors may want to move faster, or in a different sequence towards GQL
- Some vendors do not currently support openCypher and will be GQL from the ground up
- Thus, we believe that our working group should take a GQL-first approach to the libraries, tools and test cases we build
- We still want to leverage openCypher tests as much as possible to accelerate our work
- But we do not intend to strategically or technically interlock with the openCypher project

GQL Technology Compatibility Kit (TCK)

openGQL TCK priorities

- Catalog test scenarios
- DML test scenarios
- Transformed openCypher test scenarios
  - Low-hanging fruit
  - Queries (non-quantified paths)
  - Expressions
  - Create openCypher CIPs for changes that make it easier to reuse scenarios
  - Copy/modify as opposed to reference
- Work with Neo4j to make it easier to move scenarios from openGQL to openCypher and vice versa