LDBC WC-2023-01
LEX-044 Public version
https://doi.org/10.54285/ldbc.SSIF9351

LDBC Extended GQL Schema (LEX) Work Charter 1.1

Working Group Name

LDBC Extended GQL Schema (LEX) Working Group

Basecamp project

Contents

Lead

Alastair Green (JCC)

Initial members

There are ten member companies of LDBC represented, and X individual associate members. The full list is in **Appendix 2. Initial membership**.

Work group

Ad-Hoc (approximate lifetime of one and a half years, to r)

Charter version date

12 October 2023

8 November 2022 (Original)

Charter version

LDBC LEX-044 Revised timeline, revised related work

LDBC EGS-001 Revision 4 (Final, for presentation to the LDBC BoD, with fix for lost update re GSL, see footnote 4).

Date charter version agreed by Members Policy Council

16 October 2023

Mission

Propose a concrete extended GQL schema language design, as an input to WG3.

Motivation

Narrowing the gap between GQL graph types and graph schema requirements

The draft GQL standard includes optional graph types enclosing sets of node types and edge types. The experience of users of graph database languages indicates that this (vital) ground work is not enough to describe and prescribe the structure and values of practical data graphs, to at least the same degree of complexity as SQL schema.

Specifically, experience in enterprise graph deployments indicates that the following familiar features of conceptual modelling, document schema, RDF schema and SQL schema are sometimes desirable:

S1. Domains for user-defined refinements of datatypes, e.g. ranges and patterns

S2. Ability to specify nested record (tree, document) types for graph element attribution

S3. Ability to define subtyping relations for attribution types

S4. Default and computed values

S5. Keys for collections of nodes or edges (or other subgraphs) of a specific kind

S6. Cardinality and participation constraints for entities and relationships

S7. Integrity constraints (which translate into sub-graph integral units, see below)

S8. N-ary relationships

The following graph-specific schema features have also shown up as expressed needs:

G1. Ability to specify sub-graphs which are integral units. for the purpose of insertion or maintenance of a data graph, or for the purpose of access control, or as the target of constraints

G2. Schema inclusion, or “schema unions”: ability to define nodes in a schema graph which are the same or forcibly different from nodes of the same type in a second schema graph. This allows schema graphs to be composed into larger schema graphs in a modular fashion.

This permits “simple sub-graph views”, or “composite graphs”[1], but is a technique that can also be used to build up a single-graph schema with more complex topologies.

G3. Standard sub-graph topologies (e.g. trees and linked lists).

These lists could be added to, or disputed in terms of priority, but they reflect experiences of group participants in real deployments and/or existing features of some products or projects, —including in the RDF space, which obviously overlaps the property graph domain.

One symptom of the gap between “simple graph schema” and the needs of users is the proposal to use SHACL to define the schema of property graphs[2]. The wide implementation of SHACL in RDF products, in itself, shows a desire for more complicated graph schema[3]. However, this group should examine SHACL’s feature set critically: it would be useful to gather information on user and vendor experience of the use of SHACL “in the wild”.

Gathering experiences and expertise from users, vendors and researchers

A proposal by LDBC for a concrete GQL schema language would allow wide global participation by vendors and academics in the framing of the next stage of GQL as a full-featured database language. It would also be an ideal vehicle for greater involvement of end-users in our membership and work. Such a project is compatible with, and complementary to continuing research efforts in the same broad area.

The anticipated completion, in the first half of 2023, of technical work in WG3 for a Draft International Standard of the first version of the GQL language makes an effort of this kind timely.

[1] The concepts of unioning and intersecting graphs are illustrated by TigerGraph’s MultiGraph feature.

[2] https://neo4j.com/labs/neosemantics/4.0/validation/

[3] However, SHACL itself is far away from the style and target data model and type system of GQL, as a property graph database language. It builds on a profuse ecosystem of RDF standards that would also be a burden to users and implementers of a graph schema language.

Bringing a significant informal design for enhanced GQL schema into existence can help to motivate and inform an extension of the GQL project, which is likely to address the twin themes of property graph schema, and composable queries with graph projection.

Scope of Work

This work will extend the GQL Draft International Standard, anticipated to be agreed by JTC/1 in 2023, to define a proposed schema definition language (LEX GQL Schema Language[4]), which allows for more restrictive constraints on the permitted values of GQL property graphs than can be imposed by graph types as defined in the current GQL drafts.

This means that the schema language should be expressed as GQL DDL statements, which either supplement the DDL statements for graph type definition and maintenance or which can, among other things, replicate the meaning of that DDL (could be transpiled to those statements).

The schema language should allow the creation, alteration and deletion of graph schemas in a GQL catalog, including the relationship of an instance (data) graph to a graph schema during the life cycles of both.

The GQL data model

The schema language we define will avoid proposing extensions to the GQL PGDM beyond the restoration of graph attributes (which are necessary to hold summary values for the graphs used in the highly influential graph networks use case described in the 2018 "Deep Mind paper"[5], and are useful for holding searchable, classifying or processing metadata).

Relationship of schema to DML

As GQL is a database language that includes data querying and modification, any schema definition language for GQL must address not only the definition of schema conformance but the efficient maintenance of conformance as a result of transactional updates. For example, DML statements could reference schema sub-graphs explicitly to reduce the scope of validation in executing a mutation.

Partial (extensible) schema

The history of property graph databases shows the worth of schema-free operation for many use cases, including the prototyping of databases, or extensions to databases, which are intended to ultimately be governed by schema.

A GQL schema language should allow a "sliding scale of schema": the ability to partially mandate a graph topology or an element's attribution, and to describe the PGDM itself as the freest, most permissive LEX GSL schema at one end of a spectrum that extends to schemas that are as, or more restrictive than, SQL schemas. This is an area where the

[4] The acronym for this language could be LEX or it could be GSL, which "rhymes" with GQL.

[5] Battaglia, Peter W., Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, et al. 2018. "Relational Inductive Biases, Deep Learning, and Graph Networks." *arXiv [cs.LG]*.

graph schema language is likely to innovate with respect to existing industrial and standard schema languages.

Open and closed worlds

Partial schema could be interpreted as meaning that there is some part of a database whose value cannot contradict the schema. What exactly that means in the context of property graphs is worthy of discussion. This issue was discussed in the PGS WG to some extent, and there is some research work that intersects this question[6].

Three component parts of a proposed schema definition language

There are three main topics of the proposed work:

- attribute type system, including domains and default/computed values
- topology of graphs, including subgraphs which must be created or kept whole during database mutation
- constraints on subgraphs that concern the permitted and necessary values of attributes.

Four substantial points of reference will be taken into account

- Schema capabilities of existing property graph products, e.g. TigerGraph, JanusGraph, TuGraph, and of existing data modelling approaches
- SQL schema
- SHACL and its antecedents like OWL
- Extended Entity Relationship and UML class models.

These parameters focus effort on “levelling up” with database schema as known to millions of SQL practitioners; with the current state of the industrial art in graph schema; and with common industrial practice in conceptual data modelling.

The attribute type system (extending GQL’s label and property type system) and topology descriptions will combine to allow the definition of subgraph types, including subtyping relations. The re-use of existing schema languages for nested record data (like JSONSchema or GraphQL will be considered, in this regard.

Prior work in the PGS WG in 2020-21 discussed mapping between data graphs and schema graphs as a means of establishing conformance. This enables a data graph as a whole to be validated.

A subgraph can range in complexity from a node to an edge to a path to an arbitrary graph, and is therefore an ideal building block for defining a graph schema. GQL’s node and edge

[6] Databases under the Partial Closed-world Assumption: A Survey, Simon Rasniewksi and Werner Nutt, 2014 foreshadows more recent continuing work in this area, including by one of our participants, Ognjen Savkovic.

types are, in this light, simple cases of schema subgraphs. Correspondingly, a schema graph is the union of a collection of schema subgraphs.

GQL and SQL/PGQ have developed a shared graph pattern matching language (GPML) that eloquently specifies subgraphs[7] of a property graph, using sets of intersecting path patterns (graph patterns). The use of variables to identify elements in a graph pattern allows graph unions to be finely controlled, and gives references to parts of a subgraph.

Validating proposed modifications of a graph involves checking a (potential) subgraph of a data graph against a subgraph of the schema graph. Subgraphs of a schema graph can be expressed in GPML, i.e. as subgraph-projecting queries over a schema graph. This approach will strongly leverage existing work in GQL.

Schema subgraphs are also suitable as targets for key and cardinality constraints (as posited in [PG-Keys]). In this perspective a key is a subgraph that functionally determines a larger subgraph. Cardinality constraints can equally be applied to any subgraph of a wider schema graph.

Using similar syntax to GPML to define schema subgraphs would benefit users, underscoring the consistent use of graphs (expressed through graph patterns) as a leitmotiv of the GQL language. This emphasis fits exactly with the aspiration to make GQL a compositional language, closed over graphs.

Intended output documents

It would be valuable to catalogue **use cases and requirements**, including review of the SHACL UCR document from 2017. Prior work by Denise Gosnell and Matthias Broechler examining 50 query and schema use cases for their book can be leveraged, for example. In a related vein we should be able to draw on work by Gábor Syárnaz and Dominik Tomaszuk[8]. An **examination of SHACL** and its pluses and minuses may be produced.

A document or documents including **EBNF for DDL and examples, which informally describe a graph schema language that extends and complements the graph types defined in the proposed GQL draft**. The level of informality is intended to be no less than Doug Chamberlin's SQL++ handbook, and not much greater than GraphQL's first draft specification.

Other likely outputs would include the definition of a **visual representation** (like ERDs, UML models or ORM diagrams) and a **graph representation** of a schema.

It is a non-goal to produce a draft specification in a format that resembles the GQL specification. The focus of this work is useful functionality, viewed from a rigorous database practitioner's standpoint.

[7] A GPML query projects an object that contains a subgraph, but includes additional material reflecting the process of pattern matching and the desire to project tabular results. In the context of this Charter we will focus on the subgraph that can be projected from a GPM result object: strictly, when we say that a "GPML query defines a subgraph", we mean that **a GPML query followed by a graph projection operation defines a subgraph of its data graph**.

[8] Gábor Szárnyas and Dominik Tomaszuk produced a very comprehensive analysis for the old LDBC Property Graph Schema WG, reproduced as EGS-004 -- [PGSWG] Comparison / survey of graph schema and data model features.

It is a non-goal to provide a formal mathematical semantics of the proposed language, or to focus in the first instance on an analysis of its computational complexity. However, we hope to encourage, liaise with, and be informed by theoretical work on property graph schema that furthers that side of the work carried out by the old Property Graph Schema Working Group, and through that cooperation to consider and address complexity issues.

The outputs of this Working Group will be contributed to WG3 through the LDBC liaison with that body, but will be published as a developing, living resource open to the world for read access during the lifetime of the working group, at least once an initial coherent core design has been produced.

Other intended outputs/work product

Any ancillary documents, technical reports or academic papers that support the design and dissemination of the primary document(s) may be produced as additional outputs.

Sample data graphs, schema graphs and queries in e.g. Cypher or SQL/PGQ that illustrate the functionality of EGS are likely to be produced.

There are proposals under discussion in WG3 to define the property graph data model of GQL, and property graph schema metadata, by mappings from atomic facts in 6NF or so-called “graph normal form”. We will keep this possible evolution under review. The idea of describing a property graph schema by some kind of mapping from information held in 6NF is obviously one possible design approach for our work.

The relationship of a GQL schema language to schema languages for other data models is within our scope. As we define a proposed GQL schema language, and as time permits, we will have the secondary goal of describing its intersection with the features of other noteworthy languages (e.g. ORM, RDFS, OWL, SHACL, SQL/PGQ graph view definitions),

Production of a parser capable of validating syntax of examples, and of supporting experimental transpilation backends will be attempted if the skills and time are available form working group contributors.

Mode of work, cadence and intended timescales

The focus of work will be discussion of documents held in a central register. Contributions will include examples and GQL-variant EBNF, where relevant, as standard practice. It is expected that there will be videoconference meetings every two weeks.

The following timetable of milestones was originally envisaged:

- Core design including first-cut DDL and sketch of semantics: 31 January 2023, available for circulation at the February 2023 meeting of WG3. This milestone will enable us to ascertain if we have a good consensus on the design thrust.

- Second draft including comprehensive examples, and related materials (e.g. visual representation concepts, parser capability): 31 May 2023, with the goal of presentation at or around the date of the June 2023 meeting of WG3

- Third draft, which will finalize scope, and may incorporate work on possible future directions, for submission to WG3: 30 October 2023.

It is now anticipated that some of the planned output of the Working Group will be incorporated in the first version of GQL as a result of submissions from LEX to WG3. This interaction, among other factors, has slowed our overall work.

The primary focuses of work in the first eleven months have been:

1. Use cases and requirements
2. JSON Schema for graph schema
3. Type keys, names and aliases

This work has been reflected in papers submitted to WG3 at the June 2023 meeting in Washiington D.C.

The outstanding known areas of work for the next eight months, to end-June 2024, as discussed at Eindhoven and subsequently are

4. Schema graph (including information schema)
5. Schema sub-graphs and associated constraints, including
    a. integrating PG-Keys capabilities
    b. cardinality constraints
6. DML and schema sub-graphs

Other topics and areas may be added.

It is expected that the LEX WG will interact with the planned LDBC GQL Implementation Working Group with respect to the schema aspects of a GQL Technical Report.

There are likely to be two or three face-face meetings, potentially co-located with and abutting onto meetings of WG3, finances allowing. One has already occurred in March 2023.

Related Task Forces or Working Groups

Any LDBC working group (or associated research initiative in which LDBC plays a role) where a more formal academic approach is taken to describing options for, and computational limitations of, property graph schemata.

The idea of a twin “Theory” group with such a remit has been proposed as the likely direction by the LDBC board.

This group will certainly have overlapping membership with a Theory group, and should appoint one or two formal liaison representatives to make sure that the two tracks are aligned, share external presentations etc.

References to relevant documents, standards etc

[WG3 GQL specification] GQL Working Draft, 10 October 2022

[WG3 SQL/PGQ specification] SQL/PGQ Working Draft, 10 October 2022

[WG3 Thomas Frisendal] "GQL Scaffolding for GQL Foundation and Framework", September 2022

WG3 papers and documents are not for redistribution outside LDBC membership, and not for publication. Do not share the three links above outside LDBC.

[GPML] "Graph Pattern Matching in GQL and SQL/PGQ", SIGMOD '22

[SHACL spec] Shapes Constraint Language (SHACL), W3C Editor's Draft 22 January 2021

[SHACL UCR] SHACL Use Cases and Requirements -- W3C Working Group Note July 2017

[MMX reports] Documents submitted to WG3 MMX meeting June 2020

[PG-Keys] PG-Keys: Keys for Property Graphs, SIGMOD 2021

Ant Group TuGraph Schema description, 26 September 2022

Appendix 2: Presentation from first pre-meeting

LDBC EGS-002r1 -- LDBC Extended GQL Schema Work Charter themes

Appendix 2. Initial membership

The following member organizations are represented in the initial membership of the Working Group:

Ant Group Co. Ltd.

ArangoDB Inc.

AWS Inc.

Birkbeck University of London

CreateLink Technology

ENS Paris

JCC Inc.

Neo4j Inc.

TigerGraph Inc.,

Vesoft Inc.

The convenor of JTC 1/SC 32 WG3 and the editor and co-editor of GQL are taking part.

The complete list of members follows: