**From:** **Oskar van Rest** oskar.van.rest@oracle.com
**Subject:** Position Statement for the W3C Workshop on Web Standardization for Graph Data
**Date:** 8 January 2019 at 07:20
**To:** group-data-ws-pc@w3.org
**Cc:** Jan Michels jan.michels@oracle.com

# SQL/PGQ and GQL: Two faces of the same coin

Oskar van Rest – Oskar is Oracle's lead developer for PGQL, the property graph query language that is used by Oracle Spatial and Graph option to Oracle Database, Oracle Big Data Spatial and Graph, and Oracle Labs' PGX. He has also been involved in various standards activities around property graph query languages.

Jan Michels – Jan has nearly 20 years of experience in the database industry and the formal SQL standards process. He represents Oracle in the ISO and US SQL standard committees and is the chair of the INCITS DM32.2 Ad Hoc for defining SQL extensions to support property graphs.

The formal standards committees responsible for standardizing SQL (ISO/IEC JTC1/SC32/WG3 internationally and INCITS DM32.2 within the USA) have taken up the task to enhance the well-known SQL standard with extensions to query property graphs (SQL/PGQ) [1] and are actively working on bringing about a new standard for a standalone query language for property graphs (GQL) [2].

SQL/PGQ will contain extensions for SQL to create, maintain, and query property graphs. Work is currently ongoing to define these extensions. SQL/PGQ is scheduled for publication together with the next revision of the SQL standard in 2020/21. The scope of the initial version of SQL/PGQ can be summarized as follows:

— Definition of a property graph over existing relational tables
— With directed edge support
— Support for multiple named graphs
— Read-only graph query
— With the ability to find and return shortest and cheapest paths
— Modification of graph elements (vertices and edges) can be done through existing SQL DML operations on the underlying tables

Possible future extensions beyond the first version of SQL/PGQ may include support for graph DML and graph construction, undirected edges and mixed graphs, graph analytics and procedural extensions, and foreign graph interfaces to allow an RDBMS to interface with a foreign graph database.

GQL, on the other hand, is a proposed new standalone property graph query language that will exist outside of SQL but is also meant to complement SQL/PGQ. ISO has not yet officially accepted the new work item proposal for GQL. But when it does, the following capabilities for GQL have been suggested by various interested parties:

— Graph query capabilities similar to what is proposed for SQL/PGQ
— Graph DML
— Graph construction and graph views
— Graph schema language

Though GQL and SQL/PGQ will progress on different schedules (the first version of GQL will trail the publication of SQL/PGQ), having a tight integration of SQL/PGQ and GQL is a must. This tight integration will be achieved by building upon the same framework and foundation that made SQL arguably one of the most successful standards of the past 30+ years.

As there is no point in "reinventing the wheel", the first version of GQL can draw upon the scalar data types, operations, functions, and expressions on those types, predicates, three-valued logic, and other things already defined by SQL, as well as the read-only graph query capabilities of SQL/PGQ. SQL/JSON can be leveraged to define the semantics of handling data with no or only partially defined schema in the context of property graphs. SQL/JSON can also be considered to model vertex and edge types. Conversely, subsequent versions of SQL/PGQ can draw upon the relevant specifications in GQL.

There will be certain "GQL-only" capabilities that SQL/PGQ may not adopt, such as vertex types, edge types, support for graphs that have either no predefined schema or that have only a partially defined schema. There will also be certain "SQL/PGQ-only" capabilities that GQL may not adopt, such as the API for invoking a property graph query.

In this workshop contribution, we will share the current status of SQL/PGQ (what is in the latest working draft and what may come in the near future) as well as our vision for GQL which includes an SQL-like syntax for the main query structure, and graph DML for modifying a graph using INSERT, UPDATE and DELETE clauses. The DML includes capability for performing modification based on vertices and edges of arbitrary-length paths that were matched through regular expressions.

[1] http://www.incits.org/committees/dm32.2-sql-property-graphs
[2] https://www.gqlstandards.org/