A proposal to the database industry

Not three but one: GQL

A standard declarative property graph query language
A complement to SQL

The following proposal was first sent by Neo4j to vendors in the property graph data management space, and to research and industry experts active in

- Linked Database Benchmarking Council (LDBC) Graph Query Language Task Force,
- INCITS Ad Hoc for Property Graph Extensions working group (sub group of INCITS DM32.2, the U.S. national body responsible for proposing changes to the ISO SQL Standard).

Neo4j technical staff have actively contributed to both of these working groups.

GCORE *is a research language designed by LDBC collaborators, and described in a paper accepted for the industrial track of SIGMOD'18.*

PGQL 1.1 *is a language used by Oracle's PGX product, which has also been implemented in research contexts.*

Cypher *is a language originally authored by Neo4j and now maintained by the openCypher project, which has been implemented in Neo4j Server and Neo4j Morpheus for Apache Spark (beta), Redis Graph, Agens Graph (over PostgreSQL), SAP HANA Graph, Memgraph; in the open-source projects Cypher for Apache Spark and Cypher for Gremlin; and in research projects such as InGraph and Cytosm (Cypher to SQL). Its history and trajectory is described in a paper accepted for the industrial track of SIGMOD'18.*

These three languages have shared roots, much common syntax and semantics, and their authors share many technical goals for graph database query language evolution.

Friday, 11 May 2018

Dear [industry and research] colleagues,

We've been thinking about the various recent efforts on graph database query languages (SQL, LDBC, Cypher, PGQL).

I've been talking to Emil Eifrem, our CEO, and Philip Rathle, our head of Product, who are the executives sponsoring our work in this area, in the context of the very likely imminent decision to fix a limited working scope for SQL:2020 Property Graph Querying.

It's become clear that SQL:2020 will support defining graphs in DDL, their mappings to existing tables, and the ability to run a MATCH query and get out tabular results. It seems that at least Oracle, IBM, SAP and Neo4j all support that as a realistic target for SQL, judging by recent discussions and correspondence. We do feel that getting to that point in SQL would be a very worthwhile outcome.

At the same time, it's also clear that Cypher, Oracle's PGQL and the research language GCORE are all criss-crossing a broader functional space, with frequently overlapping content and perspectives.

LDBC GCORE advocates (in essence) something that looks like read-only Cypher, plus composable queries (closure over graphs), which implies multiple named graphs/graph construction, as well as regular path queries.

PGQL is very close to read-only Cypher syntactically and semantically, has multiple input graphs, and has regular path queries. But it does not yet support graph construction/closure over graphs, although we are sure that its authors would like to pursue that direction.

Cypher has expanded, with the features implemented in Cypher for Apache Spark, to cover multiple input graphs, output graphs and named graphs (closure over graphs). That's a very strong direction which we want to pursue and finalise, including updatable graphs and views. But Cypher doesn't yet have the level of regular path querying that PGQL has, which is a gap we'd like to fill.

Furthermore, many of those concerned, including SQL specialists, seem to be interested to some degree in evolving the thinking further on issues like controllable morphism and more sophisticated regular path queries.

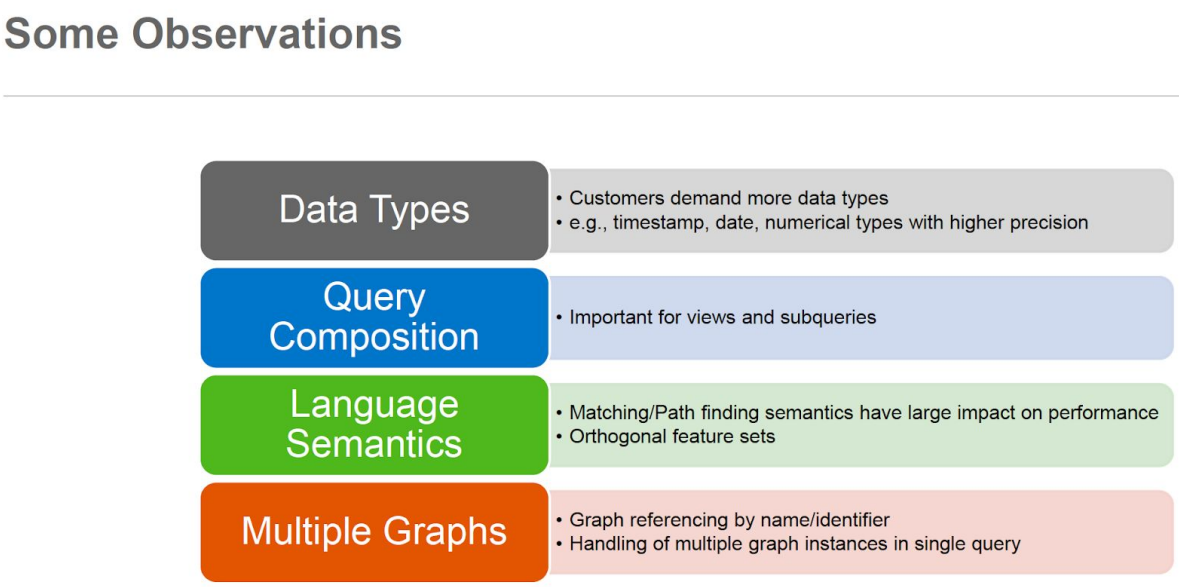
A little over a year ago at the Walldorf meeting of LDBC, Oracle's Jan Michels posed the possibility of two parallel initiatives in ISO: a native property graph query language, and extensions to SQL. By the time the current Ad Hoc was created in late April 2017 that had narrowed down to extensions to SQL alone.

We've put a lot of work into the SQL graph extensions project, and we think it's a positive thing, to be pursued.

But we also feel that there is still the parallel opportunity, to arrive at one standardised native (property) Graph Query Language (GQL) that fulfills the promise of GCORE, and which could complement SQL's more limited graph querying capabilities.

If we recall the SAP criticisms of Cypher that Marcus Paradies put forward at the first openCypher Implementers' meeting at SAP's offices in February last year (Image below),

then we can see that PGQL, GCORE and openCypher since then have all been pushing towards a bigger goal that SAP effectively advocated in his presentation:



Today we could summarize the logical outcome of that direction as: The union of PGQL and Cypher with the proposals from GCORE. Not three, but one, native property Graph Query Language.

To us, it doesn't feel that we are yet collectively acting to pursue that common logic. There is a real risk that the last fifteen or eighteen months of intensive, and largely convergent work in various groups will end up as a wasted opportunity. It's not clear that anyone intends that outcome. For our part we want to avoid that happening — and to firmly express our willingness to participate in a truly open, joint, process that could prevent that unhappy ending.

We are writing to you and all the other players, as we go into the first meeting of ISO WG3 to seriously discuss the area of property graph querying, to see whether you would support a fresh initiative, as follows:

1) Find **a formally constituted standards venue** capable of forming an international standard, or of being referenced by an international standard, with equal treatment of all participants, and appropriate IP protection for safe sharing of inputs and common work on outputs. ISO WG3 (which is in principle responsible for Database Languages in the plural, i.e. not necessarily just SQL) is one possibility, which we would be happy with. There are others, and we in Neo4j are not partisan about this point.

2) **Take as inputs PGQL, Cypher and GCORE**, and place no constraints on the nature of the output, although recognising that these languages a) intersect heavily in syntax, semantic intent and apparent roadmap, and b) in the case of PGQL and Cypher are well-established and implemented in (several) industrial products. If there are other

languages that should be considered, we would of course support their inclusion as further inputs.

3) Seek to **merge these inputs into one language — let's call it GQL** as a working title — aiming to have that adopted by the industry, including newer cloud services like Amazon Neptune and Azure CosmosDB, as a standard, to stand alongside SQL.

4) Seek to **define GQL's (supportive, interoperable, referenceable) relationship to SQL**, in collaboration with WG3.

We for our part would commit to submit without IP restriction all openCypher artefacts, software or textual, that we have generated; to seek the agreement and support of other openCypher partners; and to provide a significant group of senior Neo4j technical staff with expertise in this area for a period up to at least June 2020 to work alongside other participants on the proposed standard GQL, assuming a critical mass of industrial vendors in this space come together to pursue this initiative. (We would of course very much welcome the participation of expert researchers, as well.)

In our view three or four significant vendors, with graph database products in early adoption or generally available in the market, would constitute that critical mass.

We are considering raising this proposal at the ISO WG3/ INCITS DM32.2/Ad Hoc meetings next week in Toronto, and wanted to let you know our thinking in advance, and also to request your serious consideration of support for this initiative, which we believe to be of value to the whole industry and to all of our users.

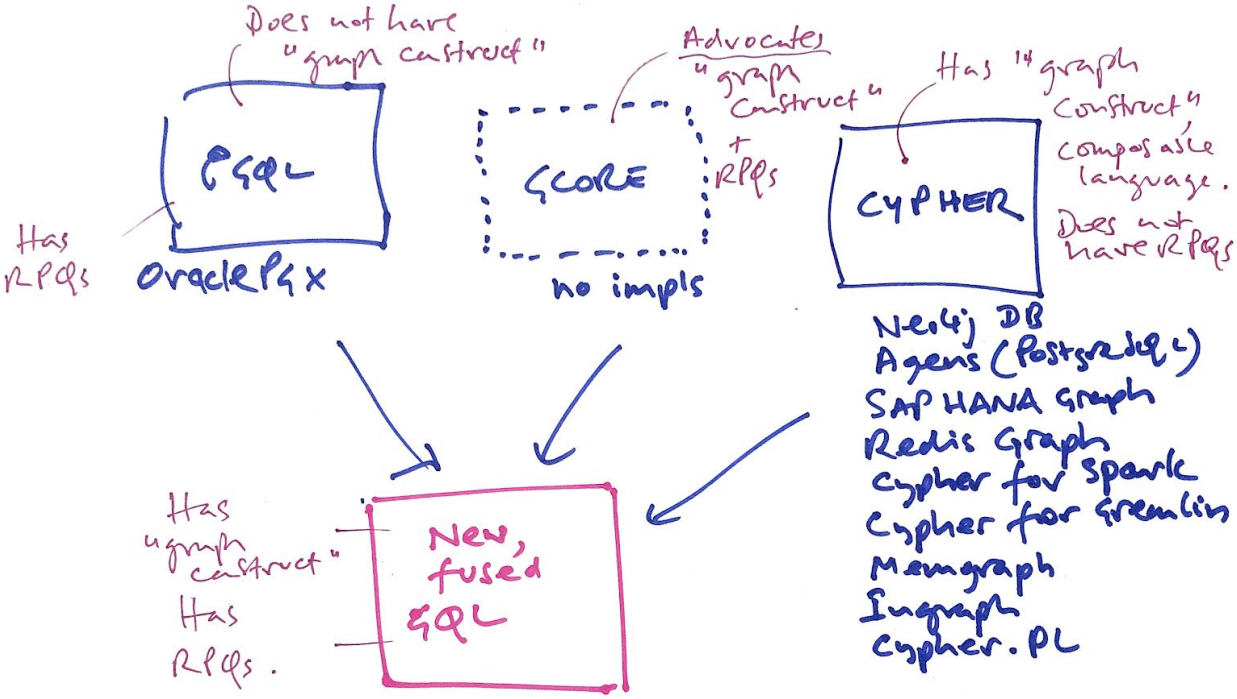
I attach a sketch that (despite my poor handwriting and drawing skills) may help to illustrate the essence of this proposal.

With warm regards,

Alastair

Alastair Green
Product Manager, Cluster and Cypher/Data Integration
Lead, Query Languages Standards and Research

Come to openCypher Implementers #4, CPH, 22-24 May


Does not have "graph construct"
PGQL
Has RPQs
OraclePG X
Advocates "graph construct"
GCORE
+ RPQs
no impls
Has "graph construct"
CYPHER
composable language.
Does not have RPQs
Neo4j DB
Agens (PostgreSQL)
SAP HANA Graph
Redis Graph
Cypher for Spark
Cypher for Gremlin
Memgraph
Inqraph
Cypher.PL
Has "graph construct"
Has RPQs.
New, fused GQL