SQL extensions for Property Graphs (PGs)

Oskar van Rest, Jan Michels – Oracle

Why Property Graphs with SQL?

- Users are using both SQL data and Property Graph data
- Application development is easier, better, quicker, faster if only one interface

Who is involved?

- ISO:
  - JTC 1 / SC32 / WG3
  - USA, Germany, Japan, UK, Canada, China

Aka the ISO SQL committee

- ANSI:
  - INCITS / DM32 / DM32.2 / DM32.2 Ad Hoc Group on SQL Extensions for Property Graphs
  - Oracle, Neo4j, TigerGraph, IBM, SAP/Sybase, JCC Consulting

Aka the US SQL committee

SQL extensions for Property Graphs (PGs)

- Goal: define extensions to query property graphs
    - Agree on one (or possibly more) representation of PGs in SQL
        - Most obvious, in tables
        - Maybe later, some “native” storage format
    - Agree on the way to query PGs in SQL
        - Query PGs “natively” (use the power of pattern matching)
        - Represent result as a table (unleash the power of SQL on the result)
        - Maybe later DML operations on a property graph directly
- Targeted for the next version of SQL (~2020/21)

Property Graph Definition (DDL) – Example

- Example:

```
CREATE PROPERTY GRAPH myGraph
   VERTEX TABLES (Person, Message)
   EDGE TABLES (
      Created SOURCE Person DESTINATION Message,
      Commented SOURCE Person DESTINATION Message )
```

Create a PG w/two vertex tables and two edge tables.

- Existing tables (or views): Person, Message, Created, Commented
- Implementation can infer keys from (primary/foreign keys) of underlying tables
- All columns of each table are exposed as properties of the corresponding vertex/edge (tables)
- Not shown here: constructs for fine-grained control over keys, labels, properties, etc.

Querying PGs – Example

Get the creationDate and content of the messages created by one person ("email1") and commented on by another person ("email2").

```
SELECT GT.creationDate, GT.content
FROM myGraph GRAPH_TABLE (
  MATCH
   (Creator IS Person WHERE Creator.email = :email1)
      -[ IS Created ]->
   (M IS Message)
      <-[ IS Commented ]-
   (Commenter IS Person WHERE Commenter.email = :email2)
      WHERE ALL_DIFFERENT (Creator, Commenter)
  ONE ROW PER MATCH
  COLUMNS (
      M.creationDate,
      M.content )
 ) AS GT
```