Who are we

- Chairs: Juan Sequeda (data.world) and Jan Hidders (Birkbeck, U. of London)
- “A group of smart people from industry and academia, stuck in a room, discussing what property graphs schemas should look like”
  - 75 people on Basecamp / Slack
  - 50 Industry: Google, Tigergraph, Uber, Neo4j, VeracityID, Amazon, Oracle, DataStax, ..
  - 25 Academia

- Inspired by the LDBC Query Language Task Force which produced G-CORE

- **Goal**: Provide community-based recommendations and analysis
  - Influence Standards; Foster scientific discussion
  - Open source implementations; ...
History

- After G-CORE (Summer 2018), several participants wanted to continue the same spirit of the work for Property Graph Schema
- Unofficial Working Group started December 2018
- W3C Workshop on Web Standardization for Graph Data (March 2019) where we presented our position: “Property Graphs should have a well-defined schema language”
  - [https://www.w3.org/Data/events/data-wsj-2019/assets/position/Juan%20Sequeda.txt](https://www.w3.org/Data/events/data-wsj-2019/assets/position/Juan%20Sequeda.txt)
  - Proposal to turn the working group as an official LDBC Task Force
- Linked to formal GQL working group WG3 (part of ISO SQL)
  - Document exchange & LDBC Liaisons at WG3 meetings
Property Types
- Records, collection types, basic types, ...
- Partial alignment with SQL types
- **Metaproperties**: all property values and their subvalues can be annotated with meta-properties

Key constraints
- **Simple key constraints**: sets of properties
- **Complex key constraints**: nodes and edges are identified by combinations of directly or indirectly connected properties and nodes
- Accepted SIGMOD 2021 paper
- Submitted VLDB paper

Cardinality constraints
- **Simple cardinality constraints**: relationship cardinality constraints (upper/lower bounds)
- **Complex cardinality constraints**: upper and lower bounds for results of graph patterns
Schema flexibility

- In node types / edge types / record types we can mark properties / fields as **optional**
- We can indicate that a record type is **open**: extra fields / properties are allowed
- We can indicate that properties allow values that conform to subtypes

Combining types

- The semantics does **by default not allow combining types**
- An analogue to combining / overlapping subtypes in conceptual data models (e.g., EER and UML diagrams) is considered
- **Journal paper in preparation**
Nominalised vs structural typing
- Types with different names do not overlap
- SQL does both
- Can we combine this elegantly and effectively?

Union Types
- **Tagged union** vs **untagged union**
- Alternative for NULL values?
- Too powerful?
- Necessary for deriving descriptive schemas?

NULL value(s)
- **SQL/PGQ** will have to deal with them
- Covered by optional properties?
- 3-valued logic?

Just-started discussions (1/2)
Derivation of descriptive schema
• Derive a schema if there is none
• For type inference
• For starting prescriptive schema

Type inference for schemas
• Determine well-typedness of query
• Determine structure of query result

Just-started discussions (2/2)
To be continued ..