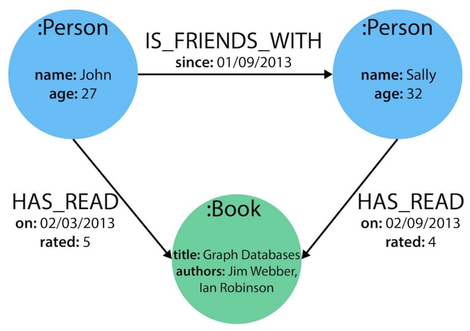
프로퍼티 그래프 질의 처리 엔진

2세부

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Property Graph

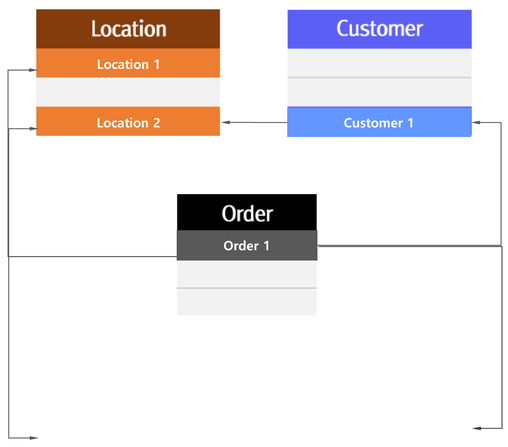
- Property Graph?
  - A graph composed of nodes and relationships, where every node/relationships can have their own properties (key-values)
- Difference with Relational Data
  - Schema
    - All tuples in a table (of RDBMS) have the same schema
    - In GDBMS, there is no constraint on schema by default
  - Relationships
    - GDBMS efficiently manages relationships between nodes, empowering ability to traverse between nodes



Why Graph DBMS?

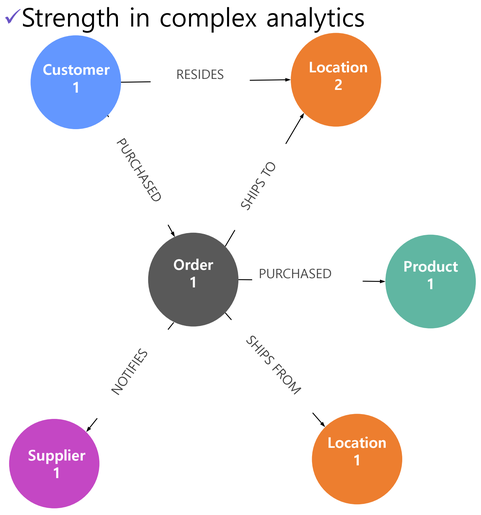
Relational DBMS

✓Strict Schema
✓High cost on complex query processing
✓Strength in simple analytics



Graph DBMS

✓Flexible Schema
✓High performance on complex queries
✓Strength in complex analytics



Limitations of the Existing GDBMS

- Inefficient Graph Query Processing
  - Non-native graph storage
    - RDBMS: Oracle PGX Spatial Graph, SAP HANA Graph
    - Document Store: Azure Cosmos DB
  - Absent of the worst-case optimal join
  - Schema-less data processing (Neo4J)
- Limited Scalability
  - In-memory Graph Query Processing (Oracle PGX)
  - Do not support distributed, parallel query processing (Neo4J, Amazon Neptune)

Our Ultimate Goals

- 100x performance improvement compared to the state-of-the-art commercial DBMS
- Supports flexible yet very fast schemaless computation
- A versatile DBMS for covering various, analytic workloads
  - Would even significantly outperform state-of-the-art RDBMSs for relational-friendly queries such as TPC-H or TPC-DS

TurboGraph-v3.0 vs Neo4J

	TurboGraph-v3.0	Neo4J
Native Graph Store	O	O
Execution Model	Push-based Pipeline (Many opt. opportunities)	Volcano, Pull-based Pipeline
Storage Format	Schemaless Columnar Format (Flexible, Fast)	Schemaless Row-major Format (Slow)
Type Inference	Extent-level (Fast) * Extent = a fixed-size set of tuples with similar schema	Vertex/Edge-level (Slow)
Distributed execution of the single query	O	X
Worst-case optimal query optimization	O	X

Main Components of the Storage

- Cache Manager
  - Implemented with a open-source in-memory object store, where multiple processed can access cached data via shared memory
  - Can cache variable-length data
- Catalog Manager
  - Manages information for the system catalog
- Extent Manager
  - Can create/delete and iterate extents
  - The storage APIs called from the execution engine can scan/seek data via Extent Manager

Execution Engine

- Vectorized, push-based execution model
  - Compared to traditional pull-based model, push-based model provides cache efficiency and is also much natural to parallelize query without much alteration
- We currently adapt expression evaluation component from that of DuckDB (a portable, high-speed HTAP database), but are planning to eventually replace them to generate LLVM IR codes