LDBC: Linked Data Benchmark Council

Non-profit company founded in 2012

Mission: accelerate progress in graph data management

Designs graph benchmarks & governs their use

Open-source under Apache v2

github.com/ldbc
Modern Graph Database Systems
Data model: Property graph

{since: 2015}
Data model: Property graph

Graph query
Data model: Property graph

Graph query

Bob knows Carl {since: 2015}

Bob knows* Spa

Bob location # ? Spa
Data model: Property graph

Bob \rightarrow Ada \rightarrow Finn \rightarrow Spa
Bob \rightarrow Carl \rightarrow Eve

Bob knows Eve since 2015

Graph query

Bob \rightarrow Spa
Bob knows* # ? Spa
Data model: Property graph

Graph query

Result: 2
MATCH (p1:Person {name: 'Bob'})-[[:knows*]]-(p2:Person)-[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
MATCH (p1:Person {name: 'Bob'}) -[:knows*]-(p2:Person) -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp

Data model: Property graph
Query language: Visual graph syntax (e.g. Cypher)

relational operators
path finding
pattern matching
MATCH (p1:Person {name: 'Bob'})-[[:knows*]]-(p2:Person)-[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
MATCH (p1:Person {name: 'Bob'})-[[:knows*]]-(p2:Person)-[[:location]]-(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp

**Data model:** Property graph

**Query language:** Visual graph syntax (e.g. Cypher)

**Pattern matching**

**Relational operators**

**Path finding**

**Ada**

**Finn**

**Bob**

**Carl**

**Eve**

**Spa**
MATCH (p1:Person {name: 'Bob'})-[[:knows*]]-(p2:Person) -[:location]->(c:Cit)
RETURN count(DISTINCT p2) AS cp
MATCH (p1:Person {name: 'Bob'}) -[:knows*]-(p2:Person) -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp

Compared to traditional SQL systems:
- more concise syntax
- better algorithms and implementations

relational operators

- direction-optimizing BFS (2012)
- batched multi-source BFS (2014)

factorization (2012)
multi-way joins (2013)
LDBC Social Network Benchmark

Graph(-capable) database systems
<table>
<thead>
<tr>
<th>Data set</th>
<th>Queries</th>
<th>Updates</th>
</tr>
</thead>
</table>

Data set

- Ada
- Bob
- Carl
- Dan
- Gia
- Finn

- knows

Queries

- Q9($name, $day)
- author
- reply

Updates

- Pa
- Pb

- *1..2
- name = $name
- creation date < $day
Q9("Bob", "Sat")
Knows relationships:
- Bob knows Ada
- Ada knows Bob
- Ada knows Carl
- Ada knows Dan
- Ada knows Gia
- Carl knows Eve
- Eve knows Finn
- Finn knows M1
- M1 knows M2
- M2 knows M3
- M4 knows M3
- M3 knows M4
- M4 knows M5
- M5 knows M3
- M5 knows M1

Queries:
Q9("Bob", "Sat")

Updates:
Pa knows Pb
name = "Bob"
creation date < "Sat"
**Data set**

- Ada
- Bob
- Carl
- Dan
- Gia

**Queries**

- Finn
- M1: Mon
- M2: Tue
- M3: Sun
- M4: Tue
- M5: Fri

**Updates**

- Q9(“Finn”, “Wed”)
- Pa
- Pb
- M

**Relationships**

- knows
- reply
Q9(“Finn”, “Wed”)
**Data set**

- **Q9(“Finn”, “Wed”):** 5 nodes
- **Q9(“Bob”, “Sat”):** 10 nodes

**Questions and Updates**

- **Q9(“Finn”, “Wed”):**
  - knows *1..2
  - name = “Finn”
  - creation date < “Wed”

- **Q9(“Bob”, “Sat”):** 10 nodes
- **Q9(“Finn”, “Wed”):** 5 nodes
Parameter selection

- *Uniform random parameters* result in unstable distributions
Parameter selection

- *Uniform random parameters* result in unstable distributions
- *Parameters selected* using the statistics results tighter distributions
Data set

- Ada
- Finn
- Bob
- Carl
- Eve
- Dan
- Gia

Queries

- M1 Mon
- M2 Tue
- M3 Sun
- M4 Tue
- M5 Fri

Updates

+ knows("Eve", "Gia")
Knows

Bob

Eve

Gia

Data set

Queries

Updates

+ knows("Eve", "Gia")

+ Comment("Gia", "M3")

When is this operation executable?
Ada
Bob
Dan
Eve
Finn
Carl
Gia

knows

author

reply

- Person("Eve")
+ knows("Eve", "Gia")
+ Comment("Gia", "M3")
Cascading deletes remove lots of entities:
- have a big impact on the data distribution
- affect the executability of operations
- influence parameter selection

For databases, deletes:
- prohibit append-only data structures
- stress the garbage collector
**Data set**

- Ada knows Bob
- Carl
- Eve
- Dan
- Gia

**Queries**

- Finn
- M1 Mon
- M2 Tue
- M3 Sun
- M4 Tue
- M5 Fri

**Updates**

- Q13($src, $dst)
- Pa knows* Pb

- name = $src
- name = $dst
Data set

- Ada
- Bob
- Carl
- Dan
- Finn
- Gia

queries

- Mon
- Tue
- Wed
- Thu
- Fri
- Sat
- Sun

knows

- knows

Updates

Q13("Finn", "Gia")

- Pa
  - knows*
  - Pb

name = "Finn"
name = "Gia"
Data set

Bob knows Ada

Ada knows Carl

Carl knows Dan

Finn knows Eve

Gia knows

Bob

Ada

Carl

Dan

Eve

Gia

knows

author

M1 Mon

M2 Tue

M3 Sun

M4 Tue

M5 Fri

reply

Q13("Finn", "Gia")

Pa knows*

Pb

name = "Finn"

name = "Gia"

Updates

- knows("Carl", "Dan")
Q13("Finn", "Gia")

Data set

- Bob
- Ada
- Carl
- Eve
- Gia
- Dan

Queries

- author
- reply

Updates

- knows("Carl", "Dan")
Q13("Finn", "Gia")

- knows("Carl", "Dan")
+ knows("Carl", "Gia")

Updates
Q13(“Finn”, “Gia”) results change over time:

- 4-hop path
- no path
- 3-hop path
Benchmark workflow

Datagen

- Data set
- Updates
- Statistics
- Paramgen
- Query parameters

Benchmark driver

System under test
Benchmark workflow

Datagen

Data set

Updates

Statistics

- Generates a temporal graph over 3 years
- Spark-based scalable generator up to 30TB data

Benchmark driver

Query parameters

System under test
Benchmark workflow

- Ensures stable query runtimes
- Generates temporal bucket of parameters
Benchmark workflow

Datagen

Data set

Updates

Statistics

Paramgen

Query parameters

Benchmark driver

System under test

Schedules operations using dependency tracking
Benchmark workflow

Datagen

Data set

Updates

Statistics

Paramgen

Benchmark driver

System under test

Candidate systems:
- Graph databases
- Triplestores
- Relational databases
Social Network Benchmark Workloads
Queries start in 1–2 person nodes

Queries and updates run concurrently

Goal: high throughput (ops/s)
SNB Business Intelligence

Queries touch on large portions of the data
Both bulk and concurrent updates allowed
Goal: high throughput & low query runtimes

Results on the 1TB data set

- Power@SF: 30,990
- Throughput@SF: 12,993

More and larger-scale results expected in 2023
Benchmark process

For each workload:

- Specification
- Academic paper
- Data generator
- Pre-generated data sets
- Benchmark driver
- 2+ reference implementations

Guidelines:

- How to execute the benchmark correctly
- Validate the results
- Verify ACID-compliance
Auditing
Auditing and trademark

Audited benchmark runs can be conducted by independent third-party auditors

- LDBC is **trademarked** worldwide
- Only a **result produced by a certified auditor** is an “LDBC benchmark result”
- Unofficial benchmark results can be reported with a disclaimer:
  “This is **NOT** an official LDBC benchmark result”
New Standard Query Languages
Graph query languages

- neo4j
  - Cypher

- TigerGraph
  - GSQL

- Amazon Neptune
  - SPARQL

- Dgraph
  - DQL

- ArangoDB
  - AQL

- TypeDB
  - TypeQL

- JanusGraph
  - Gremlin

- NebulaGraph
  - nGQL

- XTDB
  - Datalog
New ISO standard query languages

- SQL/PGQ (Property Graph Queries), June 2023
- GQL (Graph Query Language), March 2024

- relational operators
- path finding
- pattern matching

SQL/PGQ

GQL

visual graph syntax
### SQL

```sql
SELECT DISTINCT m.id
FROM ( 
  SELECT k.p2id AS id
  FROM person Pa,
  knows k
  WHERE Pa.name = $name
  AND Pa.id = k.p1id
) Pb,
Message m
WHERE Pb.id = m.authorId
AND m.creationDate < $day
```

### SQL/PGQ

```sql
SELECT id
FROM GRAPH_TABLE (sn
MATCH ANY ACYCLIC
  (Pa:Person WHERE Pa.name = $name)
  -[:knows]-(1..2) (Pb:Person)
  -[:author]-> (m:Message)
WHERE m.creationDate < $day
COLUMNS (m.id))
```

### GQL

```sql
MATCH ANY ACYCLIC
  (Pa:Person WHERE Pa.name = $name)
  -[:knows]-(1..2) (Pb:Person)
  -[:author]-> (m:Message)
WHERE m.creationDate < $day
RETURN DISTINCT m.id
```
SELECT length FROM GRAPH_TABLE (sn
MATCH p = ANY SHORTEST
(Pa:Person WHERE Pa.name = $src)-[:knows]-*
(Pb:Person WHERE Pb.name = $dst)
) COLUMNS (path_length(p) AS length))

WITH RECURSIVE ps(sp, ep, path, eR) AS (  
SELECT p1id AS sp, p2id AS ep, [p1id, p2id] AS path, (p2id = $dst) AS eR  
FROM knows WHERE sp = $src UNION ALL SELECT ps.sp AS sp, p2id AS ep,  
array_append(path, p2id) AS path, max(CASE WHEN p2id = $dst THEN 1 ELSE 0 END)  
OVER (ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS eR  
FROM ps JOIN knows ON ps.ep = p1id WHERE NOT EXISTS  
(SELECT 1 FROM ps pps WHERE list_contains(pps.path, p2id)) AND ps.eR = 0)  
SELECT min(length(path)) AS length FROM ps WHERE ep = $dst
LDBC’s involvement

The G-CORE design language (2018) influenced SQL/PGQ and GQL

Language semantics:

- Formal Semantics Working Group

LDBC conducts research on graph schemas:

- LDBC Extended GQL Schema (LEX) – industry-driven
- LDBC Foundations of Graph Schema (FOX) – theory-driven
Outlook:
The LDBC Graphalytics Benchmark

Graph processing frameworks
(Apache Giraph, NetworKit, GraphBLAS, etc.)
Data set

- Ada
- Finn
- Bob
- Carl
- Eve
- Dan
- Gia

Algorithms

- untyped, unattributed graphs

- LDBC SNB
- Graph500
- Twitter
- Friendster
- Patents
- Wiki-Talk
Graphalytics algorithms
Graphalytics algorithms

Breadth-first search(source: “Bob”)
Bob

Dan

Carl

Ada

Finn

Eve

Gia

BFS: 0

BFS: 1

Graphalytics algorithms

Breadth-first search(source: “Bob”)
Graphalytics algorithms

Breadth-first search(source: “Bob”)
Graphalytics algorithms

Breadth-first search (source: “Bob”)
Ada
Bob
Dan
Finn
Carl
Eve
Gia

BFS: 0
BFS: 1
BFS: 1
BFS: 2
BFS: 2
BFS: 2
BFS: 3

Data set

Graphalytics algorithms
Breadth-first search(source: “Bob”)
PageRank(damping factor: 0.85, iterations: 5)

Algorithms
Graphalytics algorithms

Breadth-first search (source: “Bob”)

PageRank (damping factor: 0.85, iterations: 5)
Graphalytics algorithms

- Breadth-first search (source: “Bob”)
- PageRank (damping factor: 0.85, iterations: 5)
- Clustering coefficient
- Community detection
- Connected components
- Shortest paths

Graphalytics spring 2023 competition – please reach out if interested
Wrapping Up...
Joining LDBC

Members can:

- Participate in benchmark design & research
- Commission audits
- Gain early access to ISO standard drafts, SQL/PGQ and GQL

Pricing:

- Free for individuals
- 2,500 EUR/year for companies
- 10,000 EUR/year for sponsor companies

Visit our website at ldbcouncil.org and reach out at info@ldbcouncil.org
### SNB Interactive

- **Query:** $Q_9(name, \$day)$
  - name = $name$
  - creation date < $\$day$

### SNB Business Intelligence

- **Query:** $Q_{11}(country)$
  - name = $country$

### Graphalytics

- **Algorithms:** BFS, CDLP, PR, SSSP, LCC, WCC
- **Data sets:** LDBC SNB, Graph500, Twitter, Friendster, Patents, wiki-Talk

### Semantic Publishing Benchmark

- **Target:** RDF/SPARQL
- **Domain:** Media/publishing industry
- **Inferencing & continuous updates**

### Financial Benchmark

- **Target:** Distributed systems
- **Domain:** Financial fraud detection
- **Strict latency bound (20 ms)**

### Future benchmark ideas

- **GNNs**
- **Graph mining**
- **Graph streaming**