The LDBC Social Network Benchmark

Gábor Szárnyas, David Püroja
(CWI Amsterdam, LDBC)

FOSDEM 2023 Graph devroom

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

Sponsor Members

ANT GROUP

ORACLE Labs

火山引擎 Beijing Volcano (ByteDance)

Regular Members

aws

intel

neo4j

TigerGraph

NebulaGraph

JCC Consulting, Inc.

KATANA GRAPH

MEM GRAPH

ontotext

创邻科技 CREATE LINK

pometry

ULTIPA™

*Sparsity

海致星图 StarGraph

達摩院 ALIBABA DAMO ACADEMY

ArangoDB

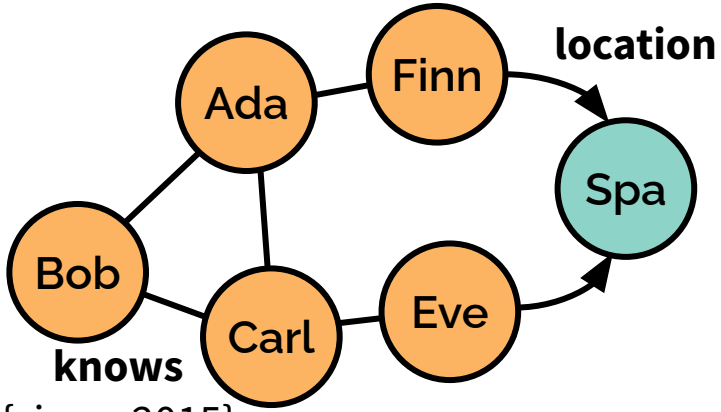
ENS ÉCOLE NORMALE SUPÉRIEURE

FORTH

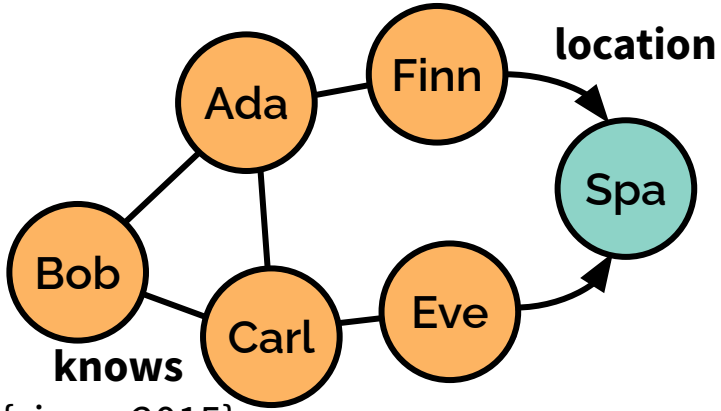
Birkbeck UNIVERSITY OF LONDON

Modern Graph Database Systems

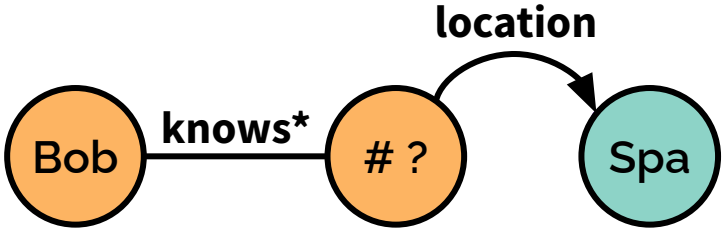
Data model: Property graph



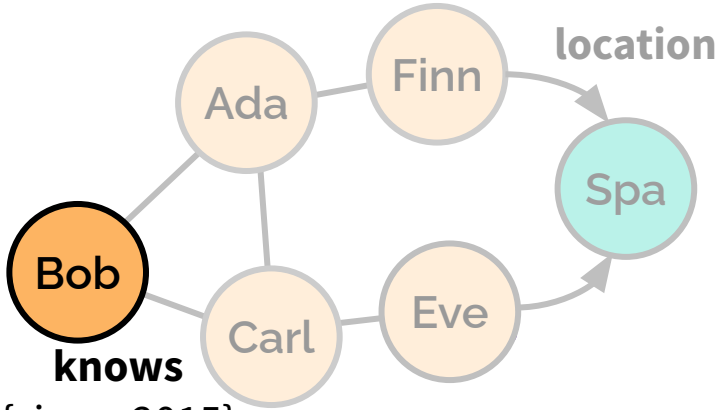
Data model: Property graph



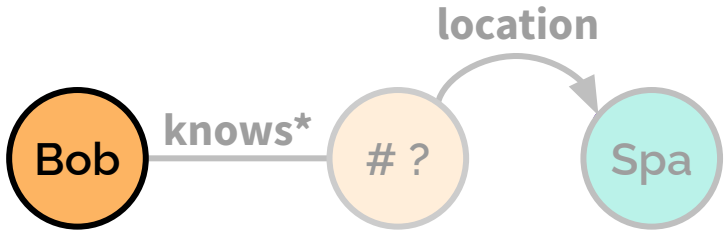
Graph query



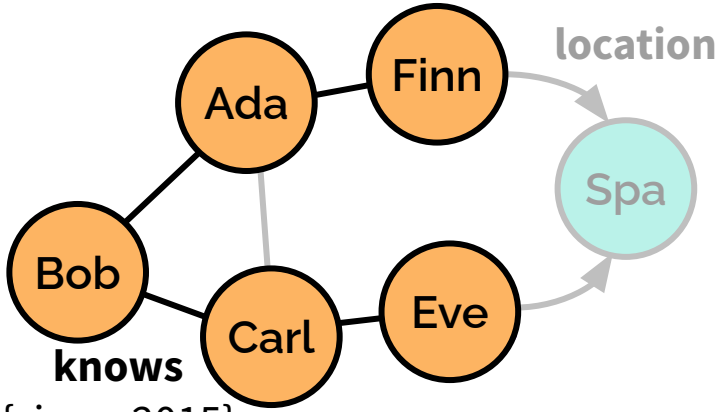
Data model: Property graph



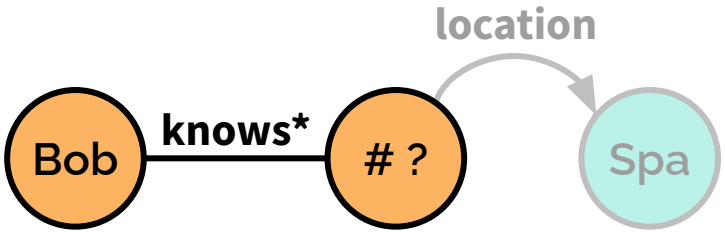
Graph query



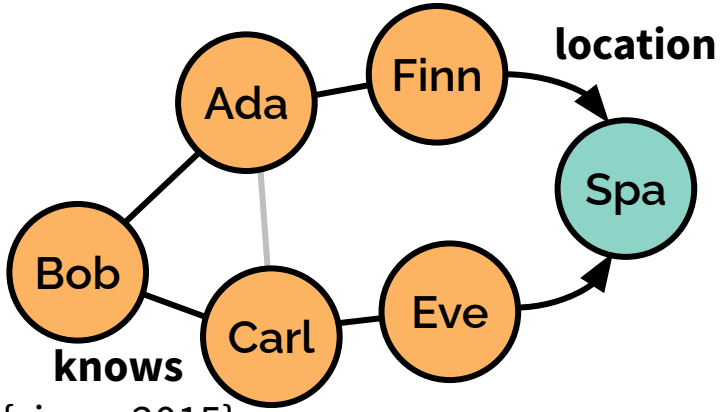
Data model: Property graph



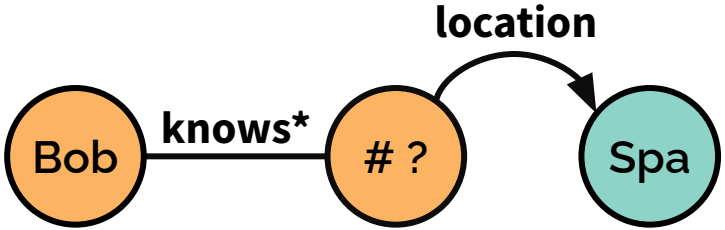
Graph query



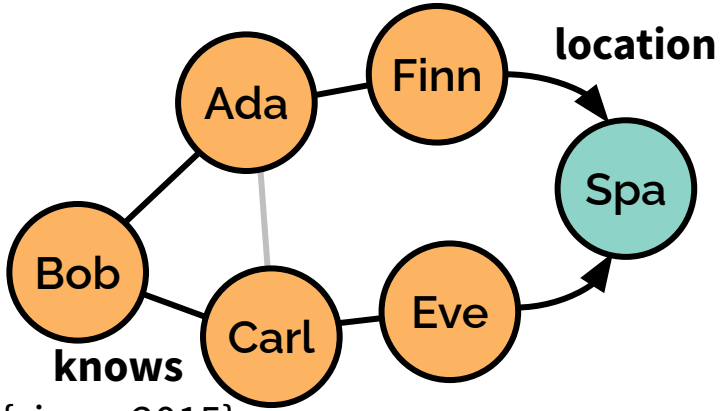
Data model: Property graph



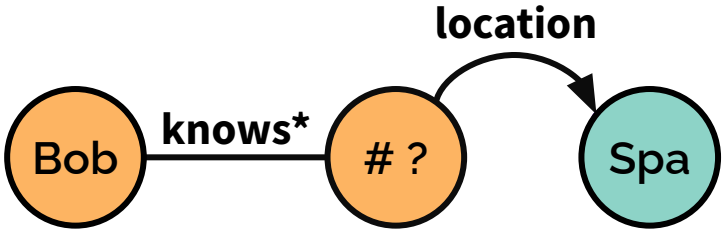
Graph query



Data model: Property graph

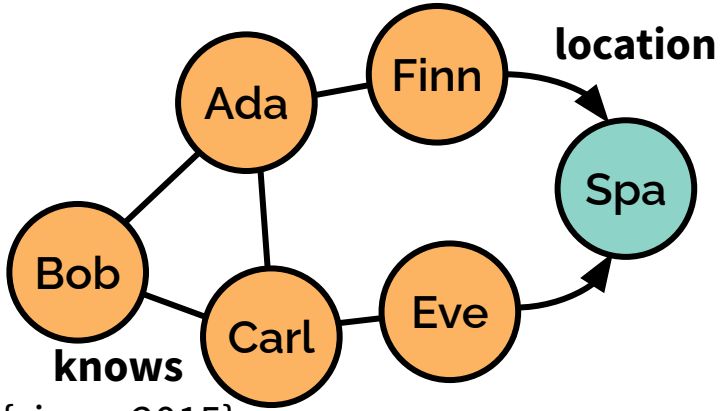


Graph query

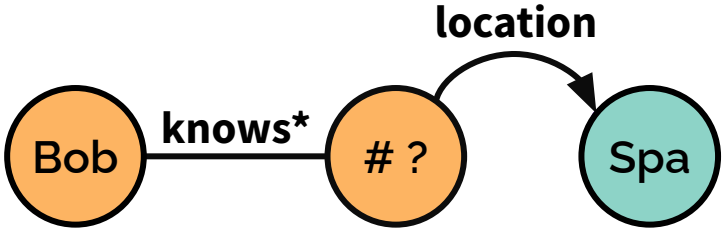


Result: 2

Data model: Property graph

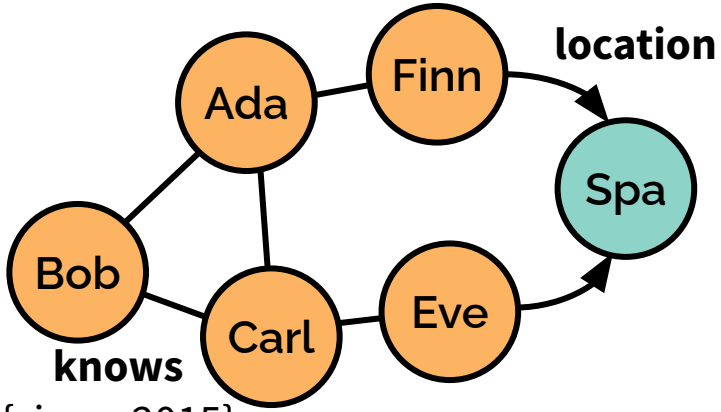


Query language: Visual graph syntax (e.g. Cypher)

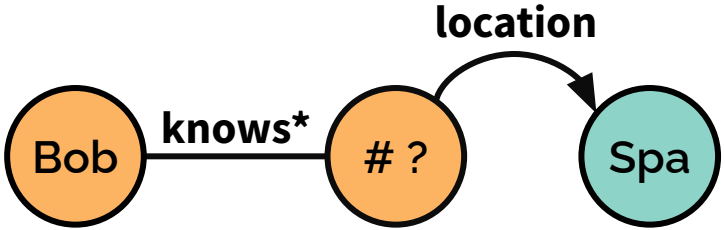



```
MATCH
  (p1:Person {name: 'Bob'})
  -[:knows*]-(p2:Person)
  -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
```

Data model: Property graph



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



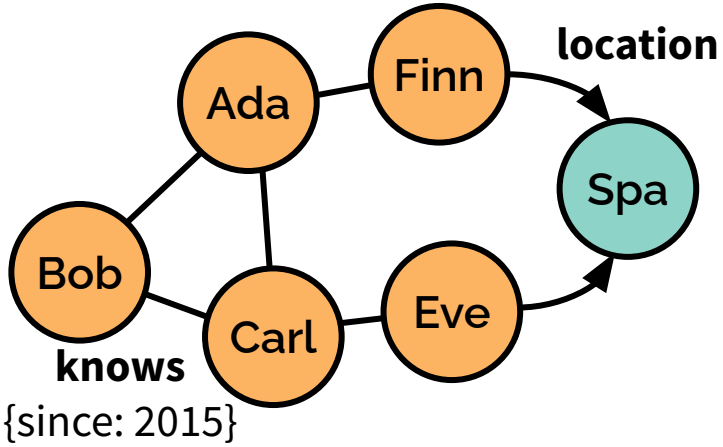

```
MATCH
  (p1:Person {name: 'Bob'})
  -[:knows*]-(p2:Person)
  -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
```

relational operators

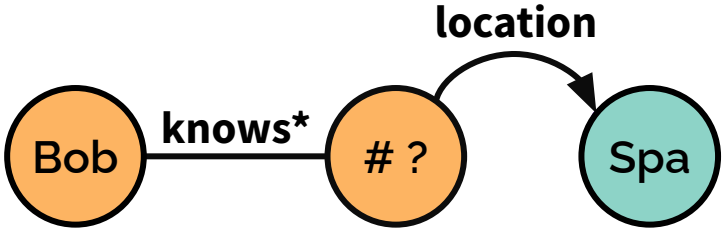
path finding

pattern matching

Data model: Property graph



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



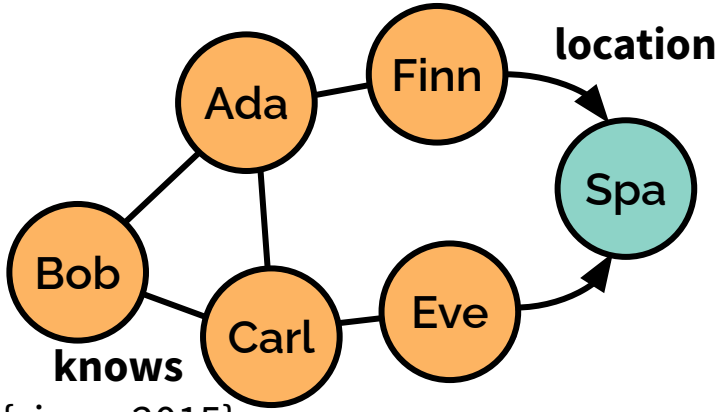

```
MATCH
  (p1:Person {name: 'Bob'})
  -[:knows*]-(p2:Person)
  -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
```

relational operators

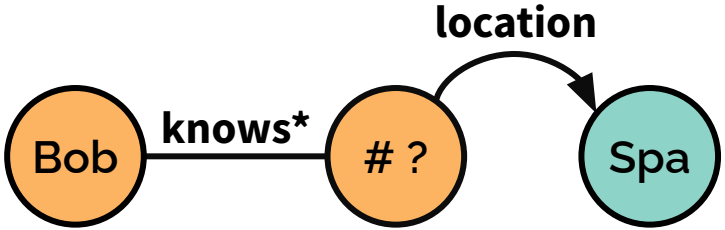
path finding

pattern matching

Data model: Property graph



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



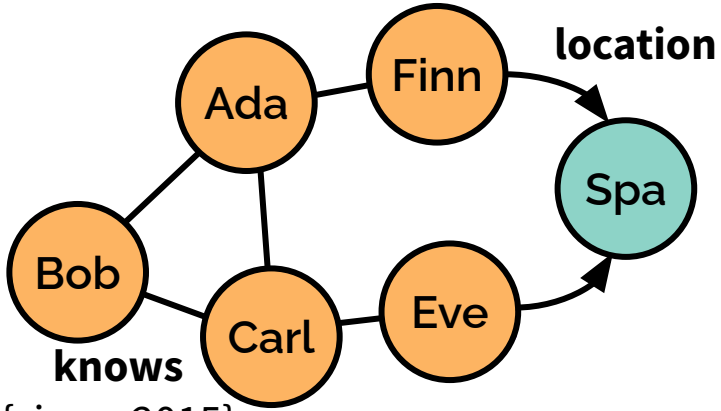

```
MATCH
  (p1:Person {name: 'Bob'})
  -[:knows*]-(p2:Person)
  -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
```

relational operators

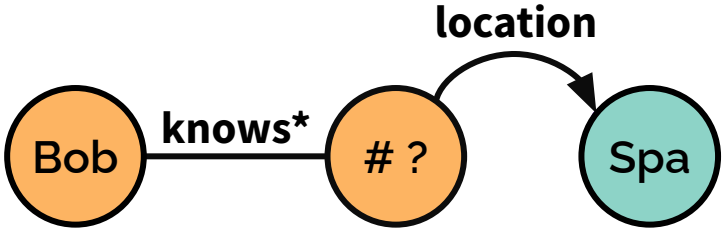
path finding

pattern matching

Data model: Property graph



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



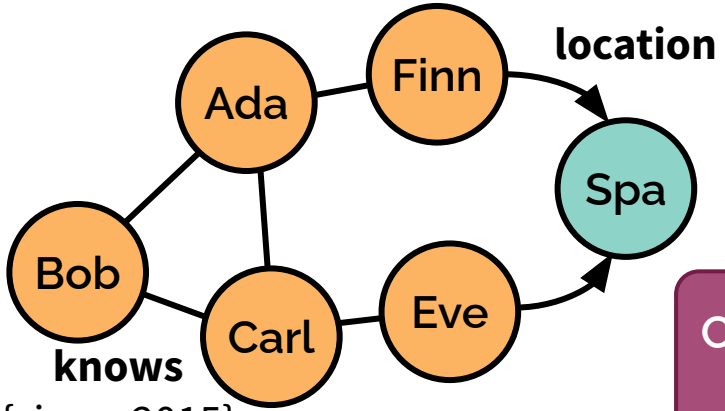

```
MATCH
  (p1:Person {name: 'Bob'})
  -[:knows*]-(p2:Person)
  -[:location]->(c:City {name: 'Spa'})
RETURN count(DISTINCT p2) AS cp
```

relational operators

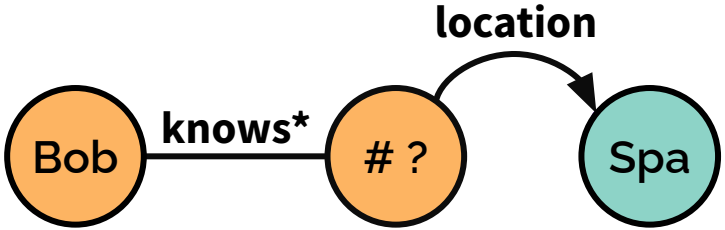
path finding

pattern matching

Data model: Property graph



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




```
MATCH
```

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

})

relationa
operato

direction-optimizing BFS (2012)

batched multi-source BFS (2014)

factorization (2012)

multi-way joins (2013)

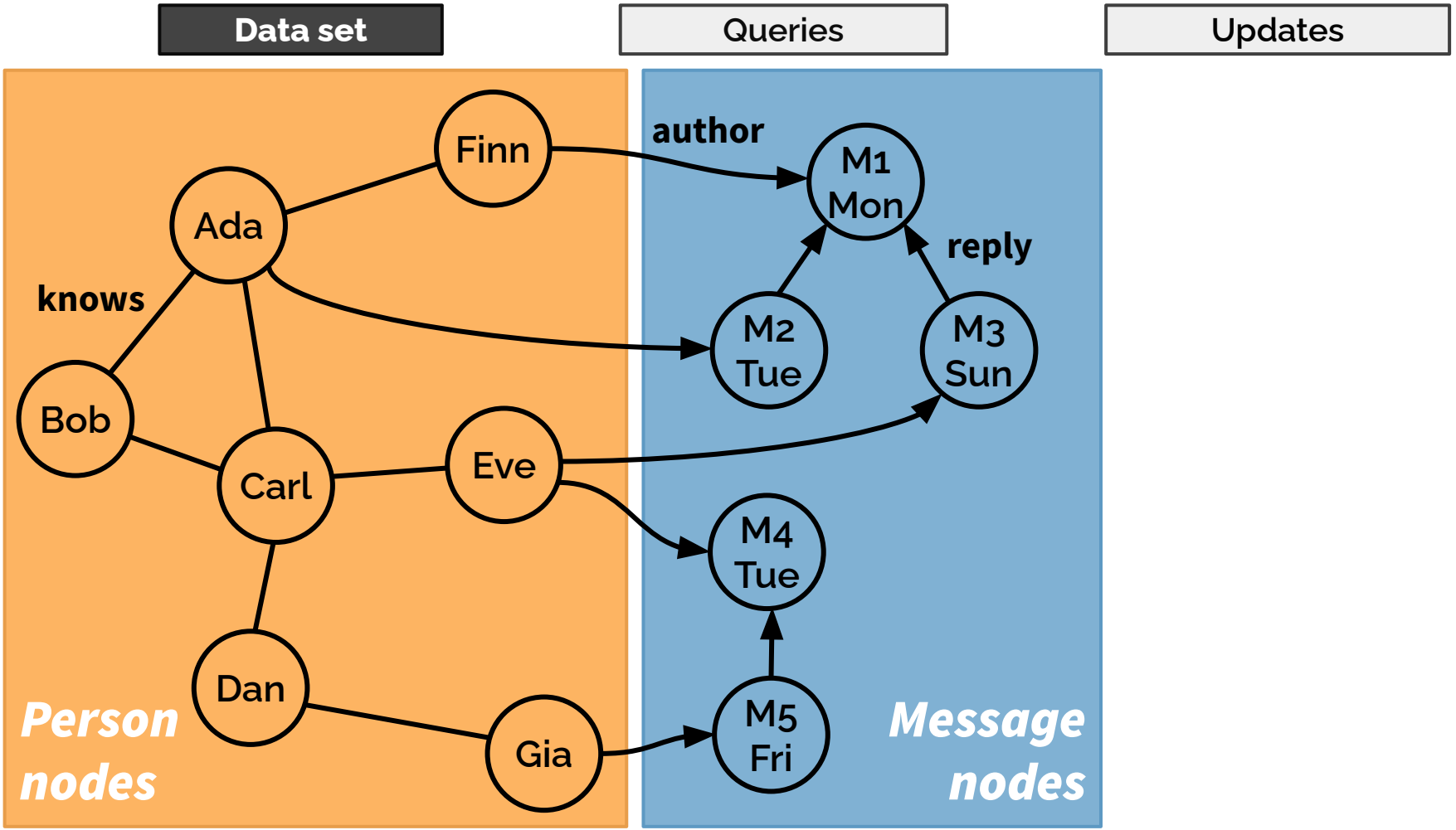
LDBC Social Network Benchmark

Graph(-capable) database systems

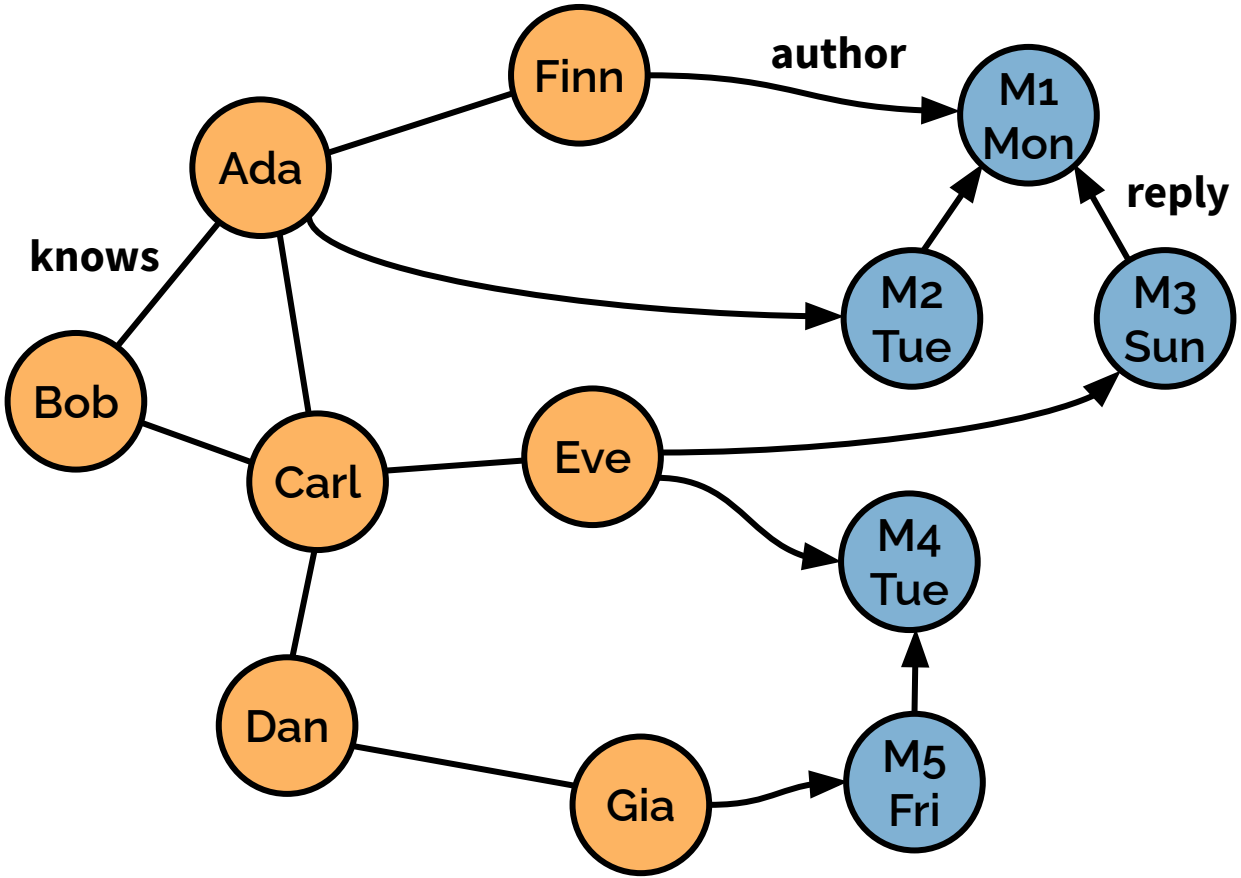
Data set

Queries

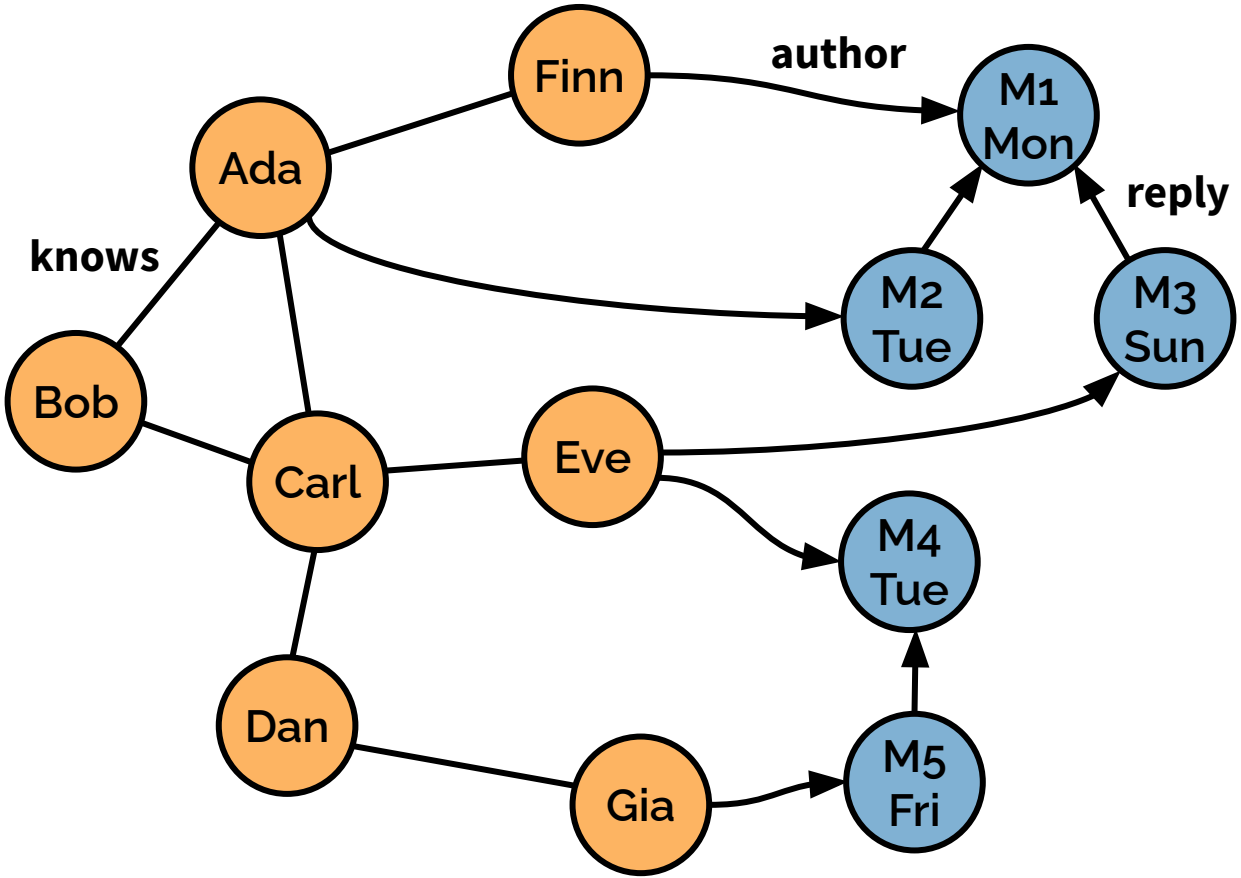
Updates


Data set
Queries
Updates
Finn
author
M1
Mon
Ada
reply
knows
M2
Tue
M3
Sun
Bob
Carl
Eve
M4
Tue
Dan
Gia
M5
Fri
Person
nodes
Message
nodes

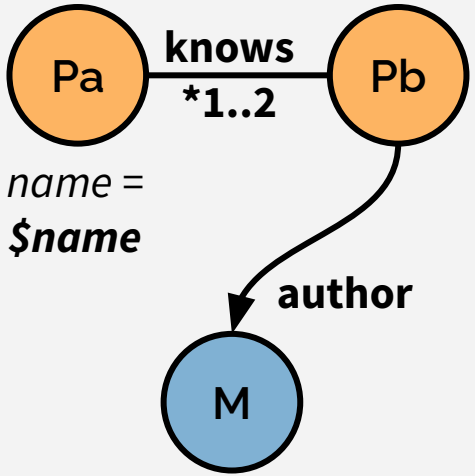
Data set | **Queries** | Updates



Data set | **Queries** | Updates



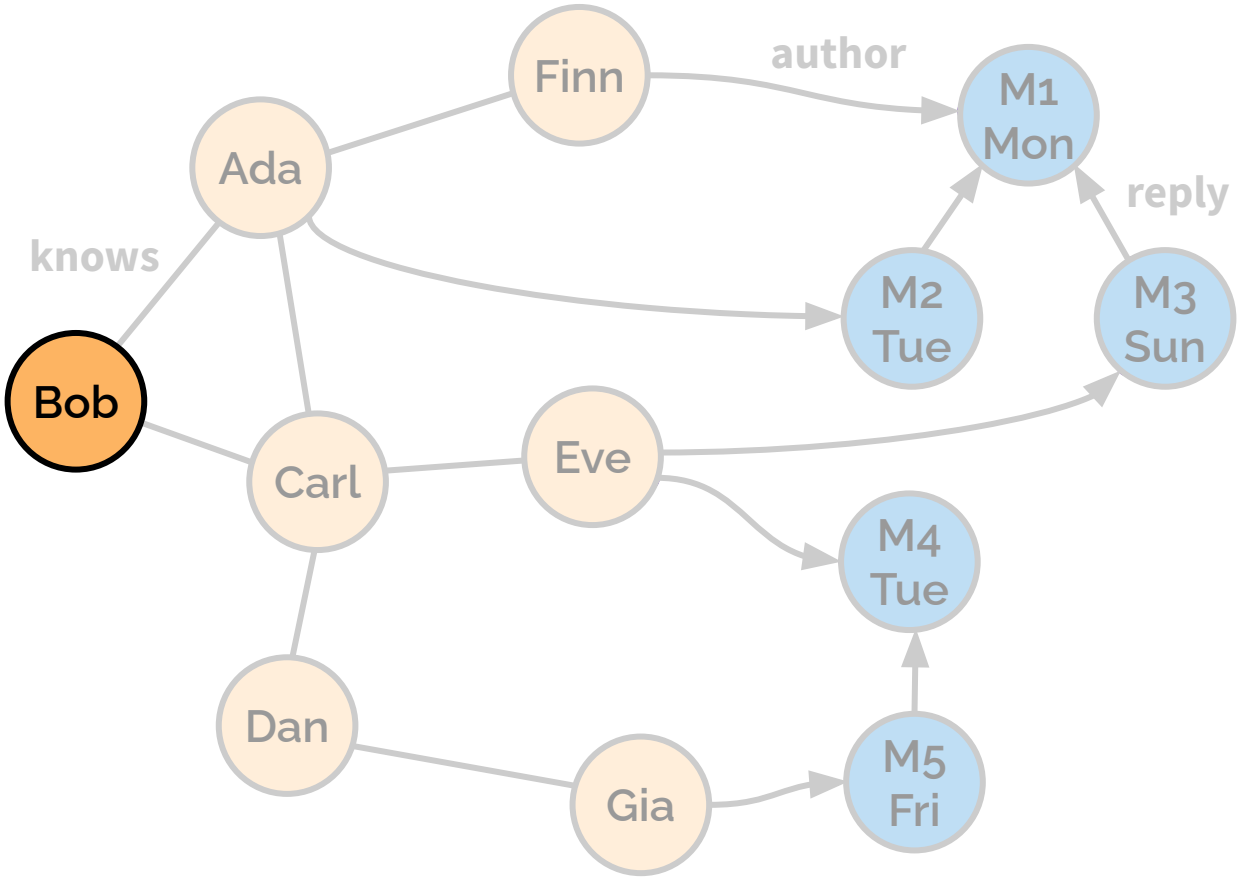
Q9(*$name*, *$day*)



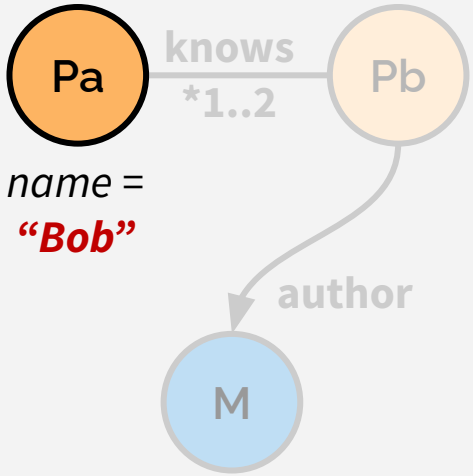
Data set

Queries

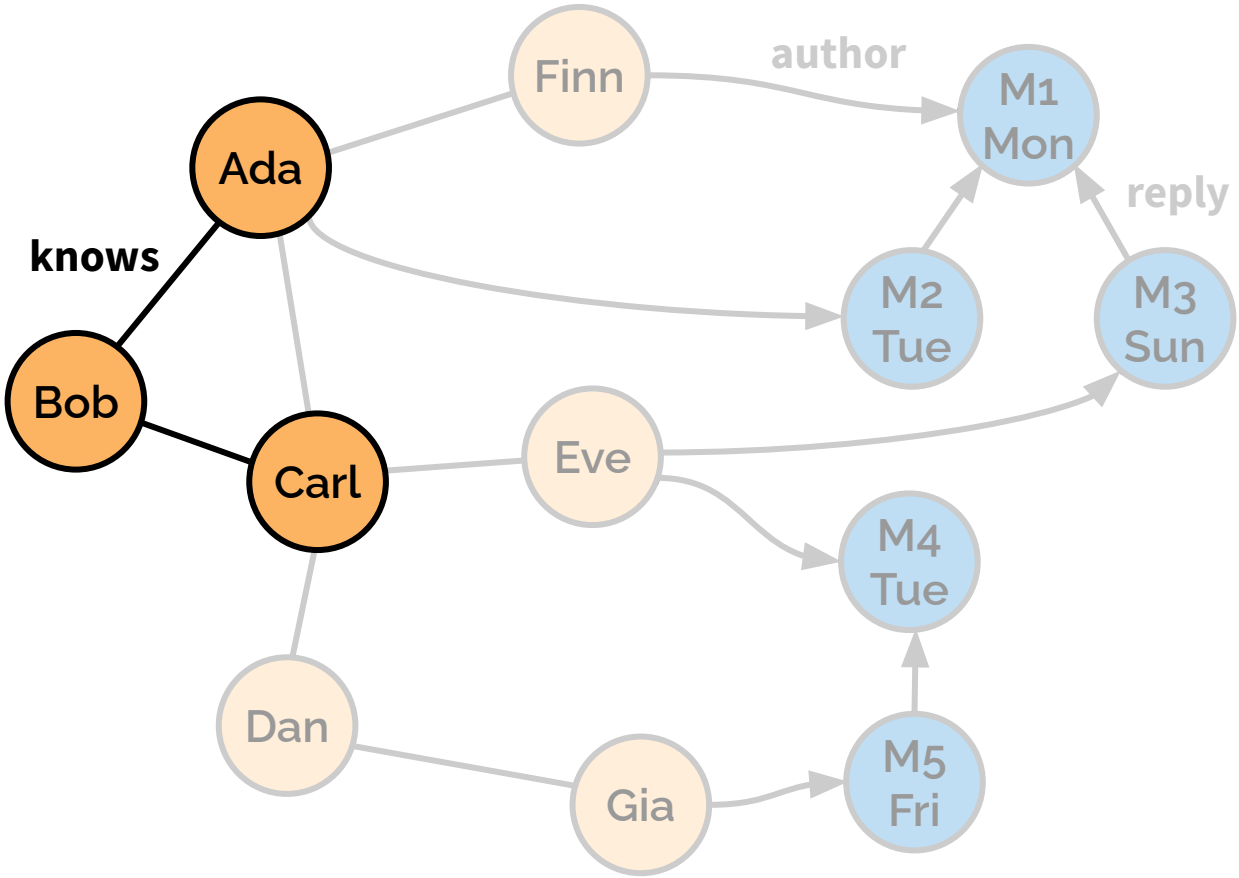
Updates



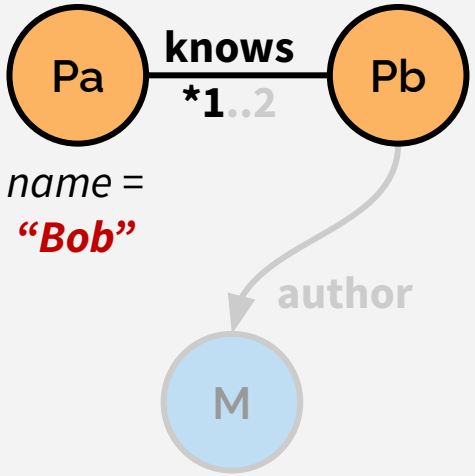
Q9(*"Bob"*, *"Sat"*)



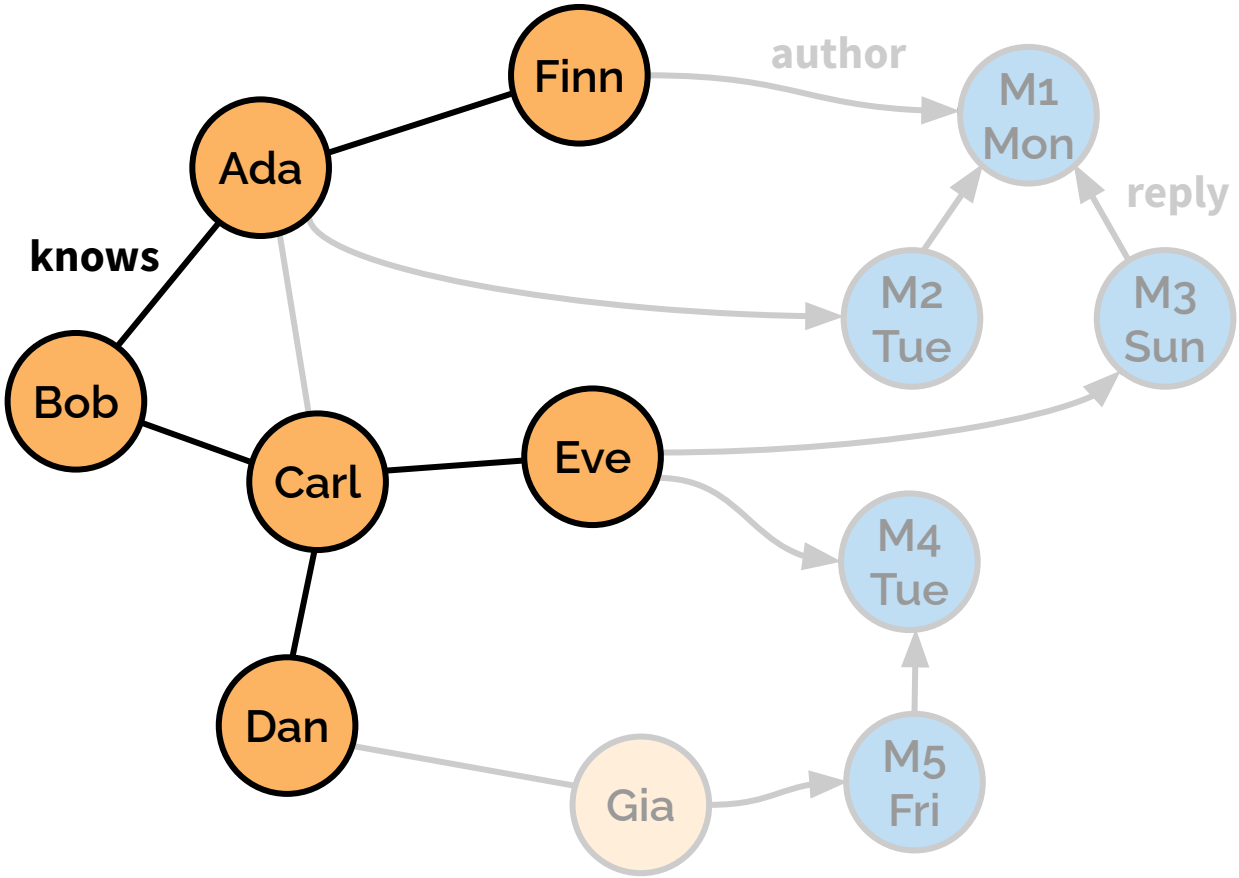
Data set | **Queries** | Updates



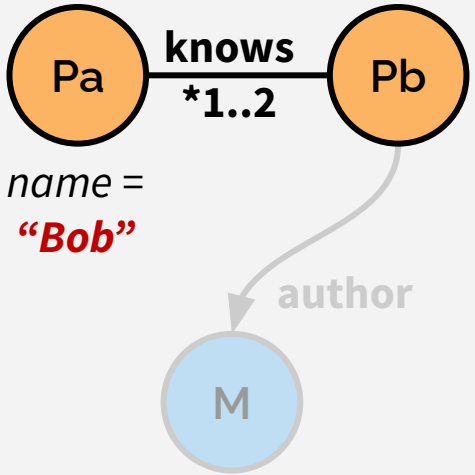
Q9(*"Bob"*, *"Sat"*)



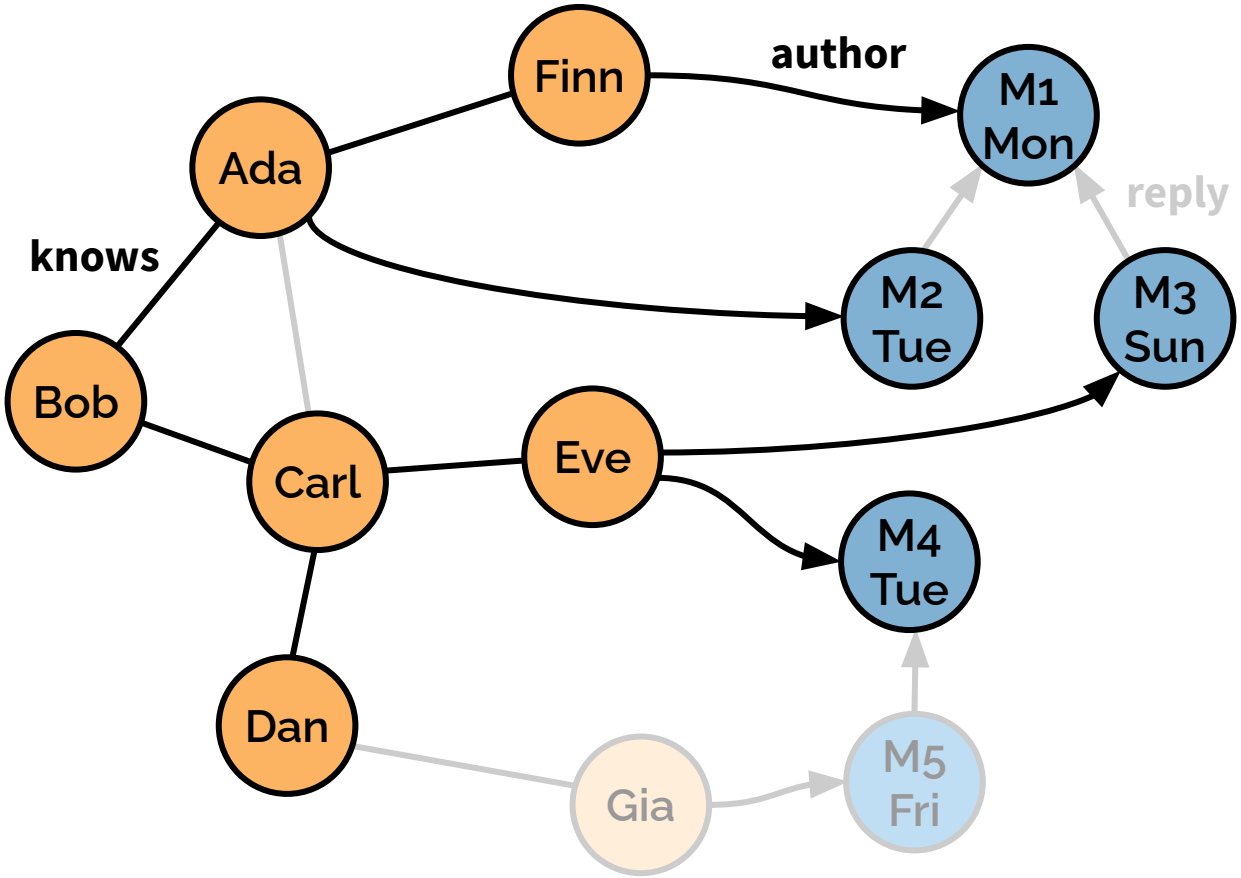
Data set | **Queries** | Updates



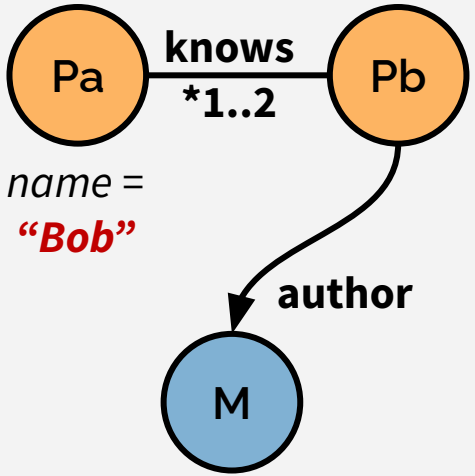
Q9(*"Bob"*, *"Sat"*)



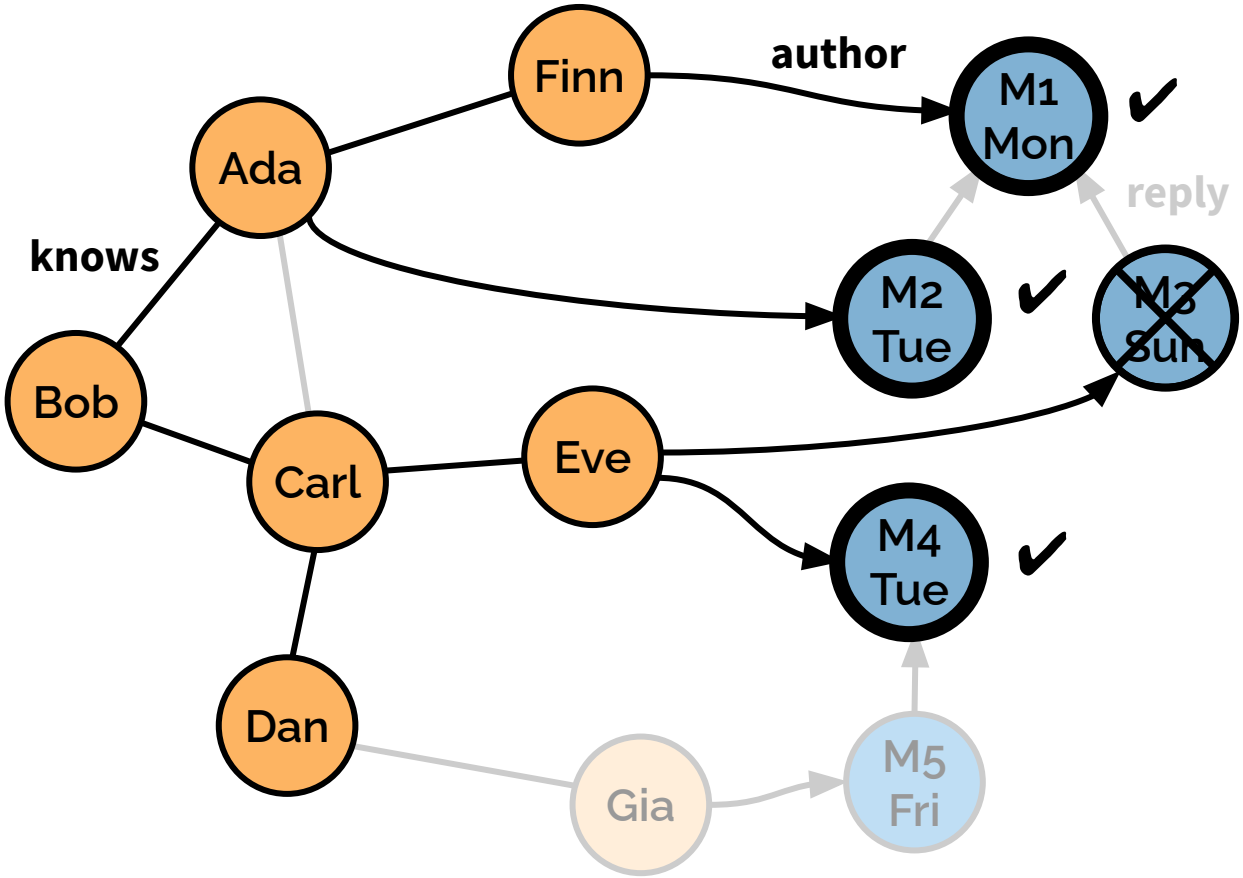
Data set | **Queries** | Updates



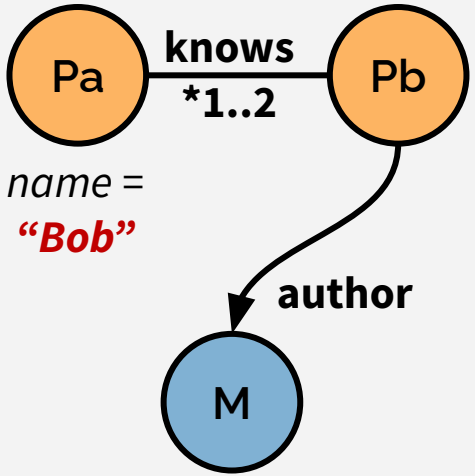
Q9(*"Bob"*, *"Sat"*)



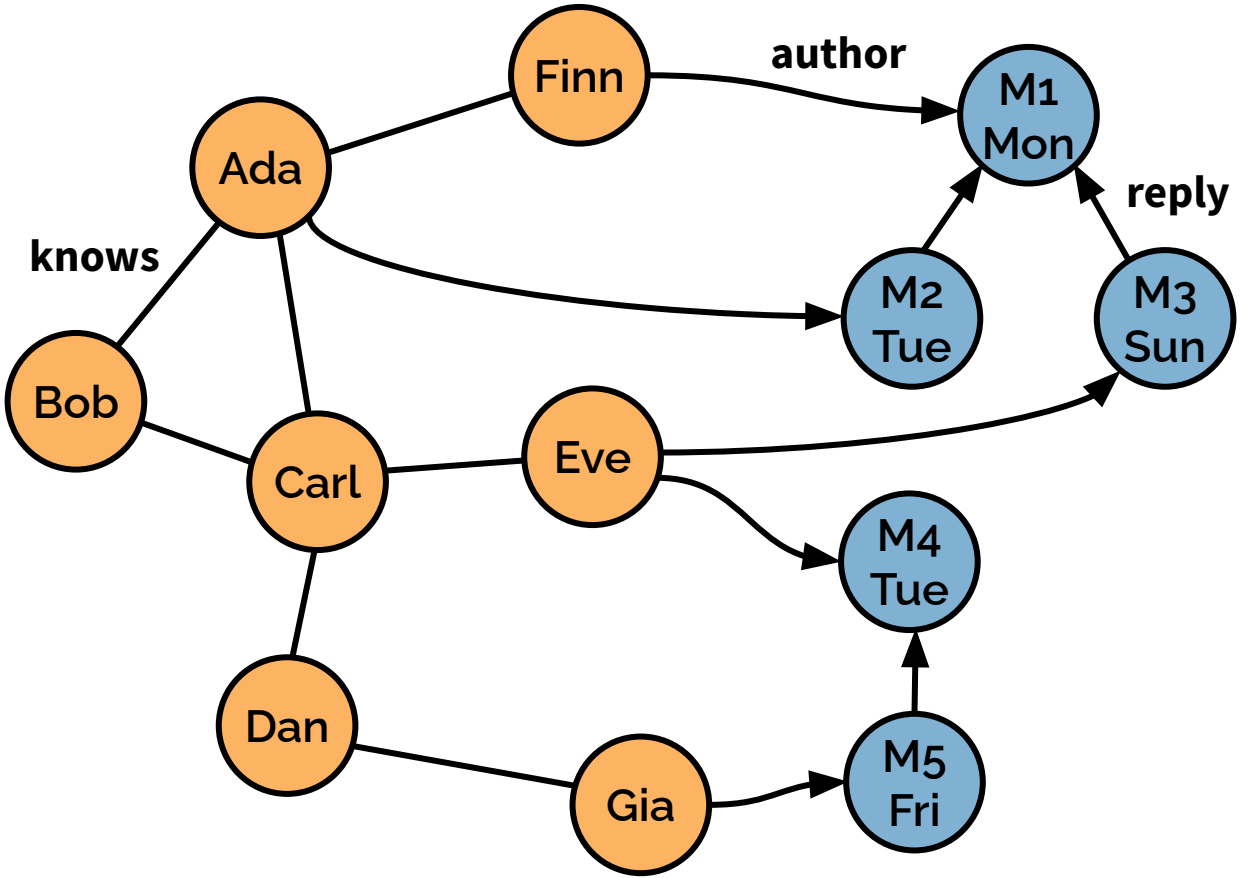
Data set | **Queries** | Updates



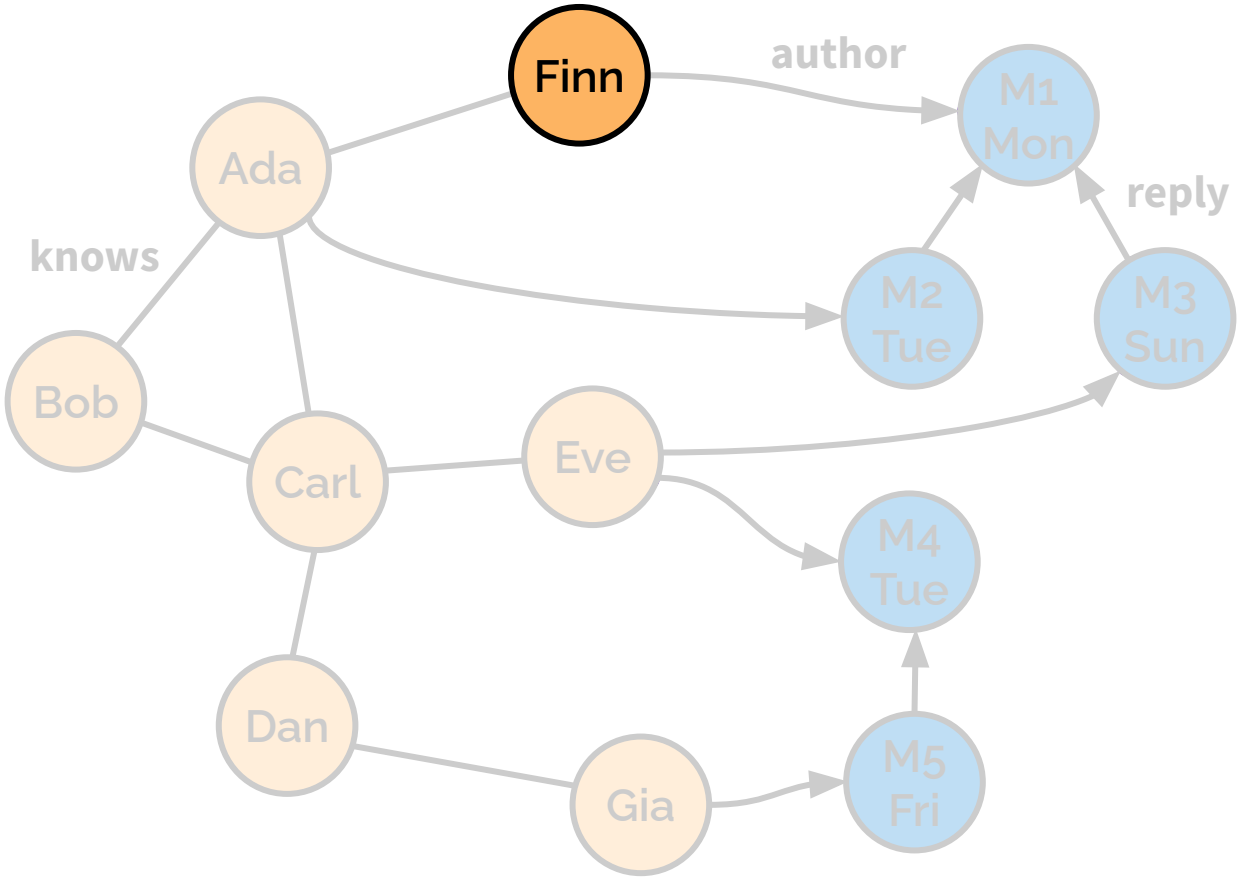
Q9(*"Bob"*, *"Sat"*)



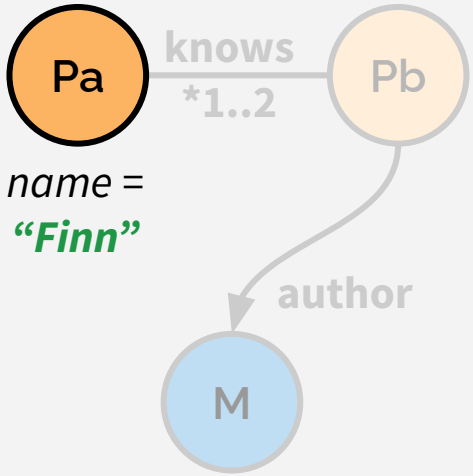
Data set | **Queries** | Updates



Data set | **Queries** | Updates



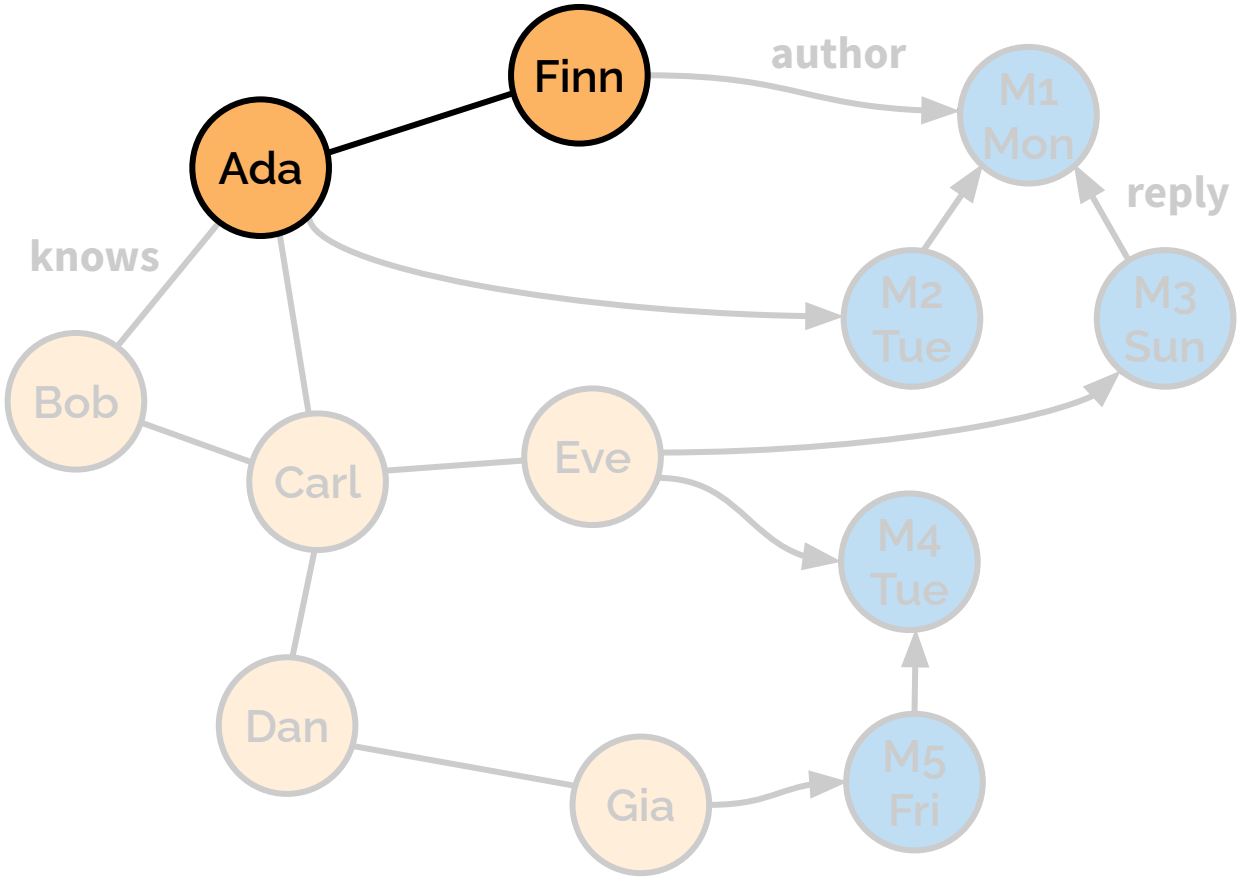
Q9(*"Finn"*, *"Wed"*)



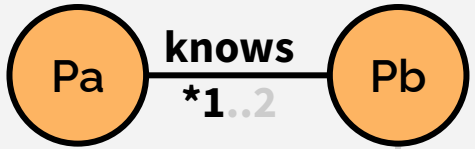
Data set

Queries

Updates



Q9(*"Finn"*, *"Wed"*)



name =
"Finn"

author

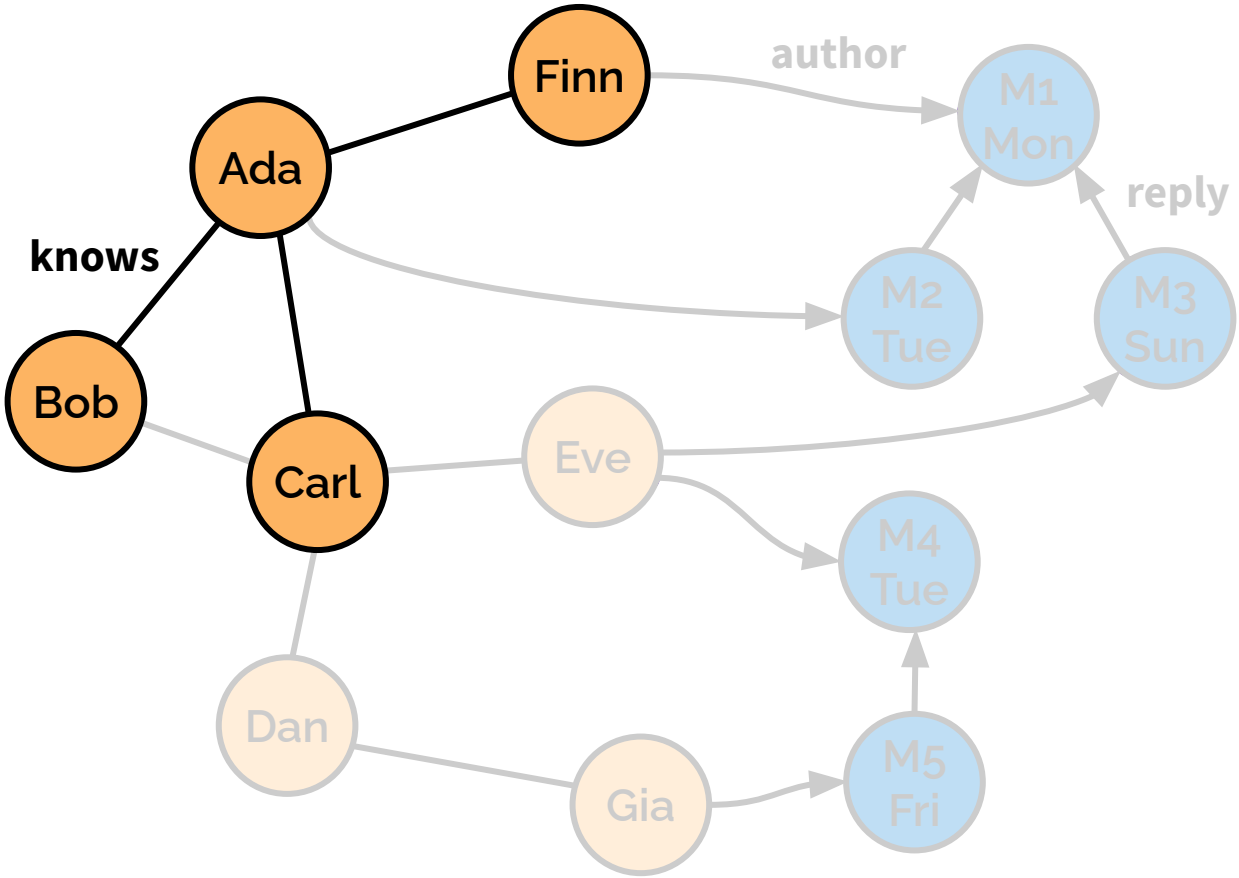
M

*creation date < **"Wed"***

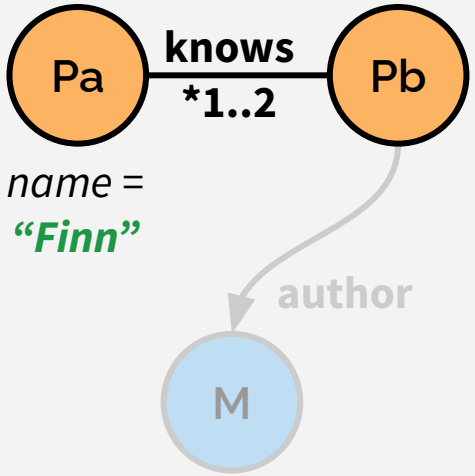
Data set

Queries

Updates



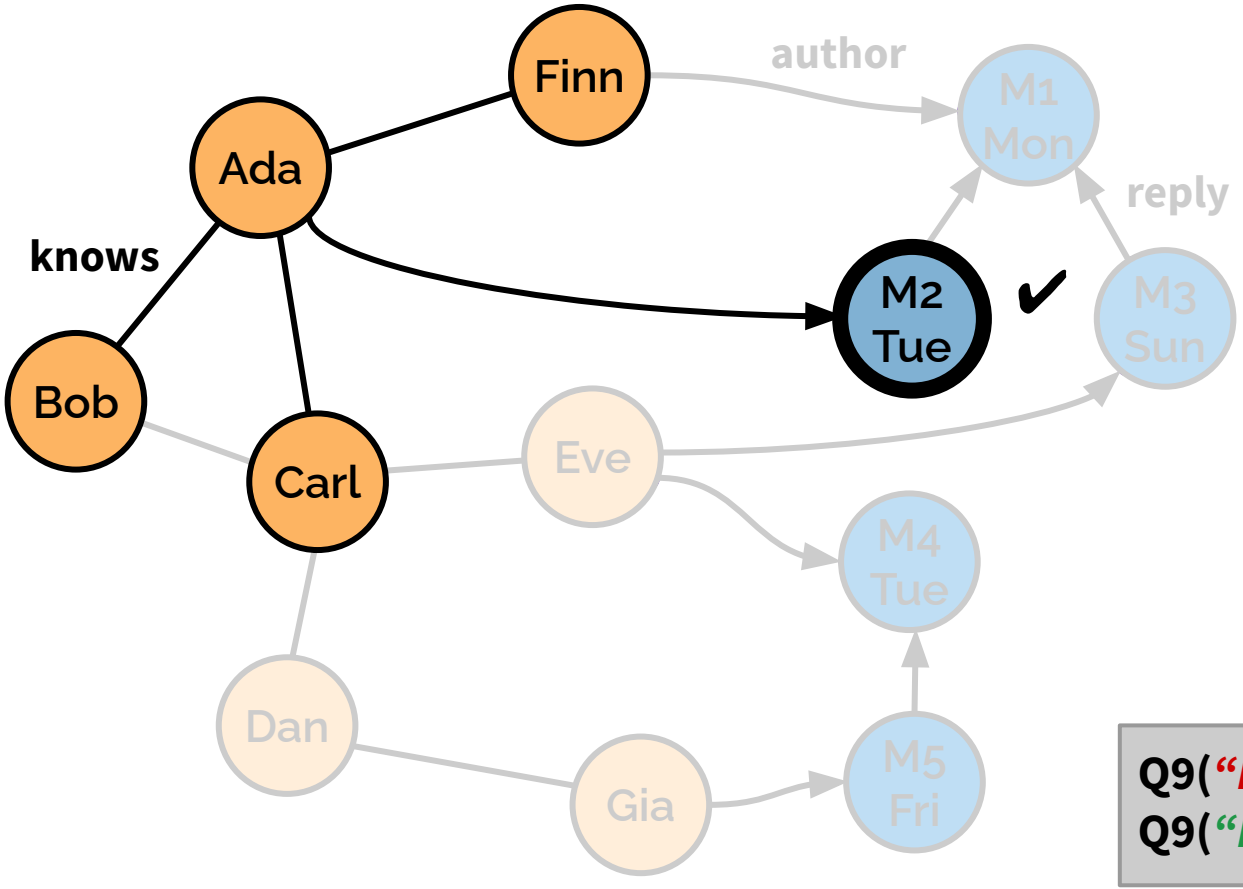
Q9(*"Finn"*, *"Wed"*)



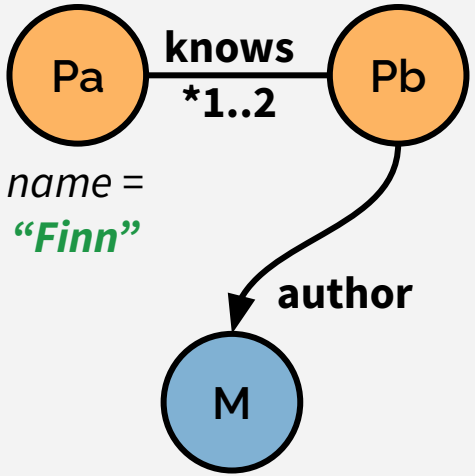
Data set

Queries

Updates



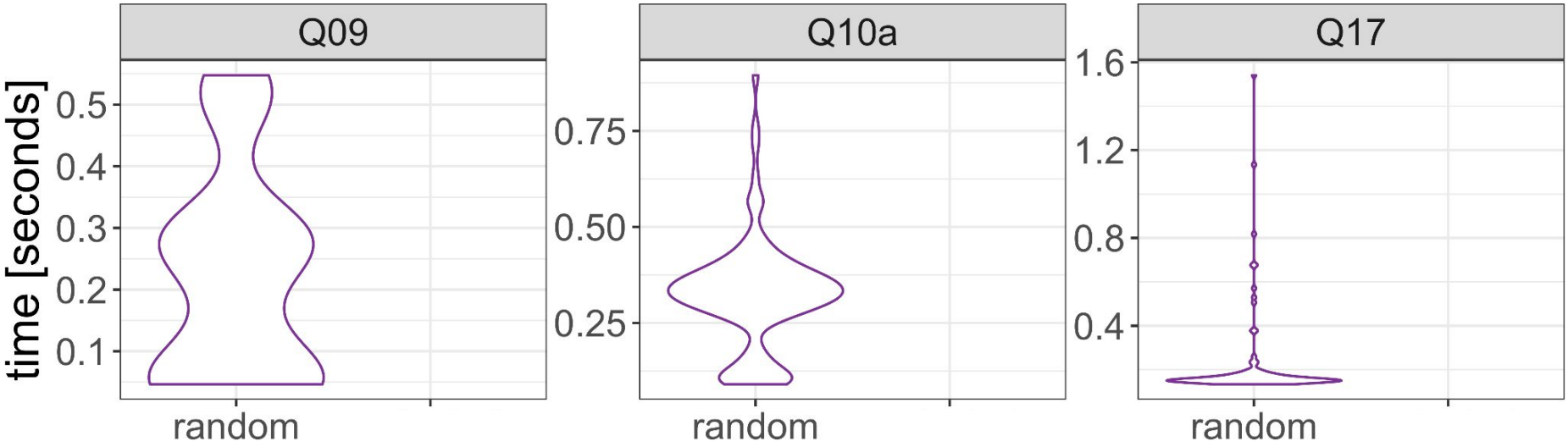
Q9(*"Finn"*, *"Wed"*)



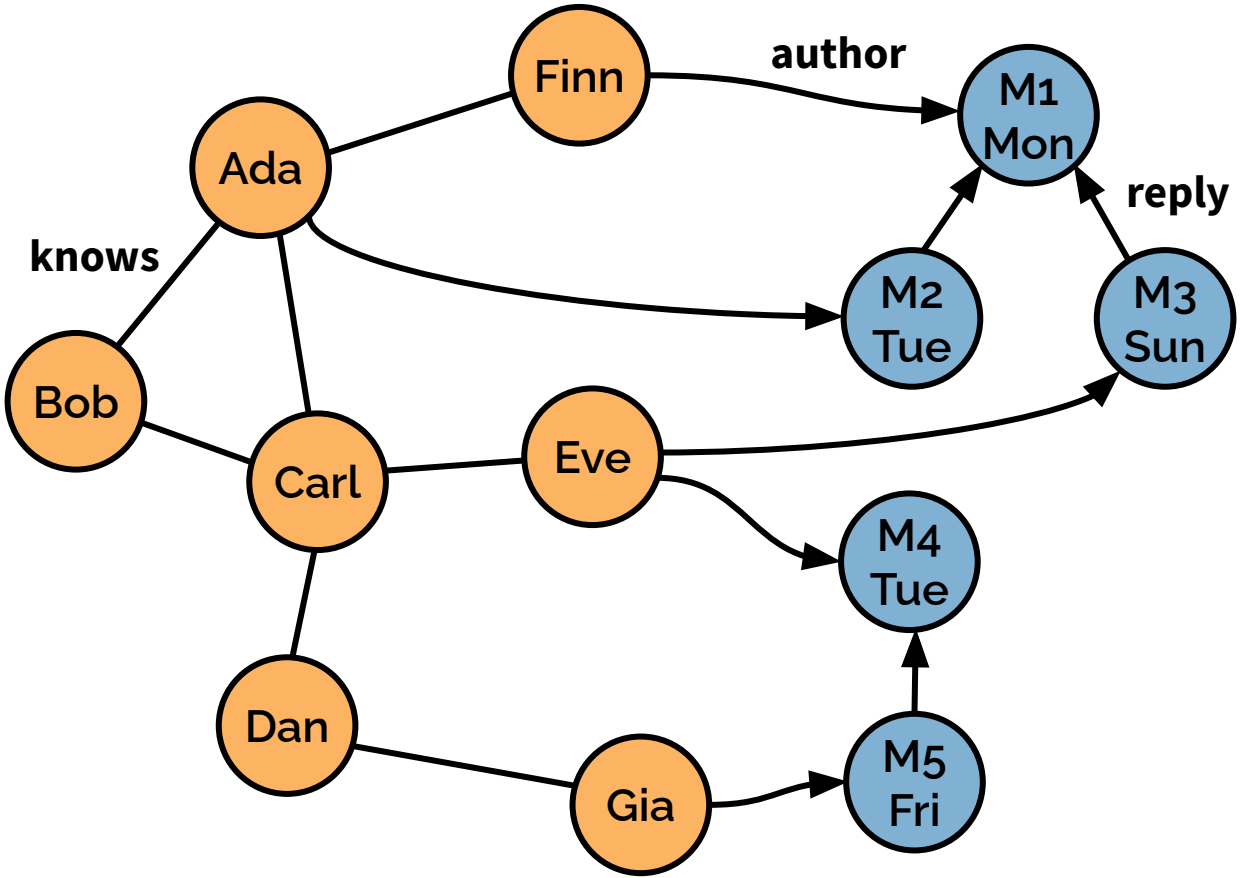
Q9(*"Bob"*, *"Sat"*): 10 nodes
Q9(*"Finn"*, *"Wed"*): 5 nodes

Parameter selection

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



Data set | **Queries** | Updates

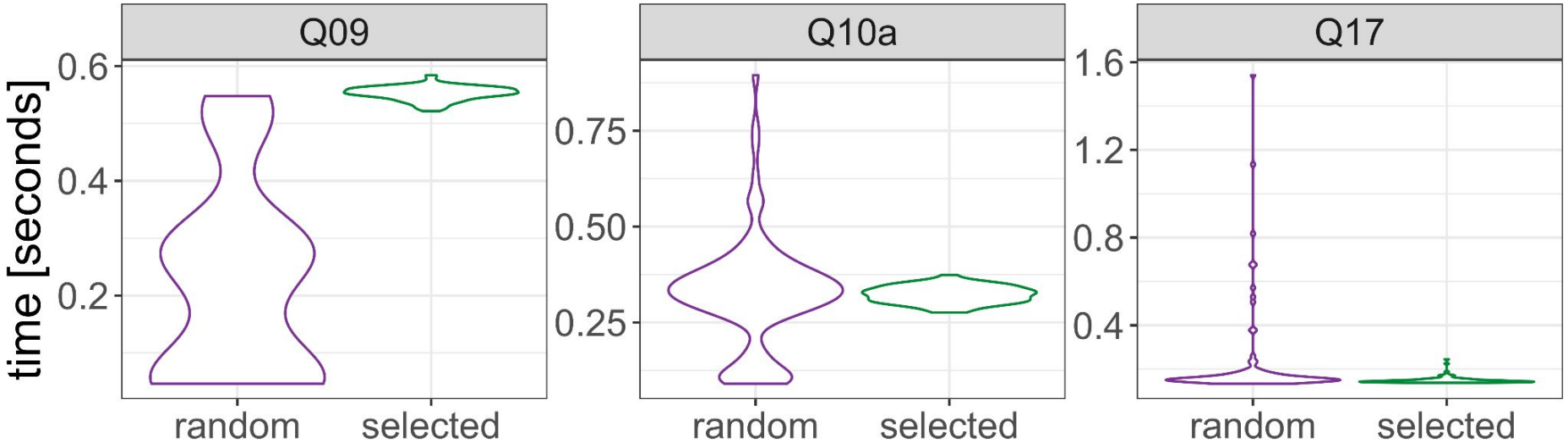


Statistics

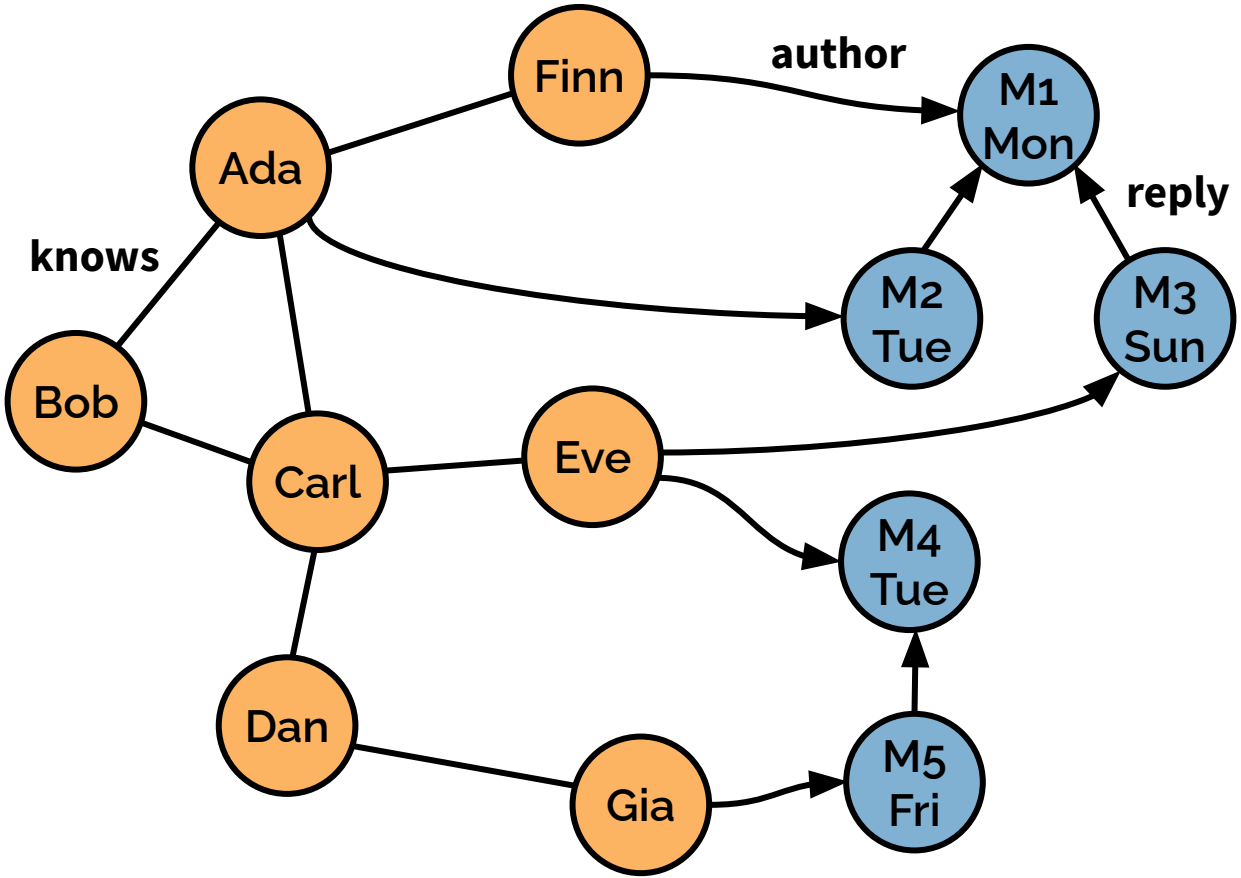
name	1-hop friends	2-hop friends
Bob	2	3
Carl	4	2
Ada	3	2
…		

Parameter selection

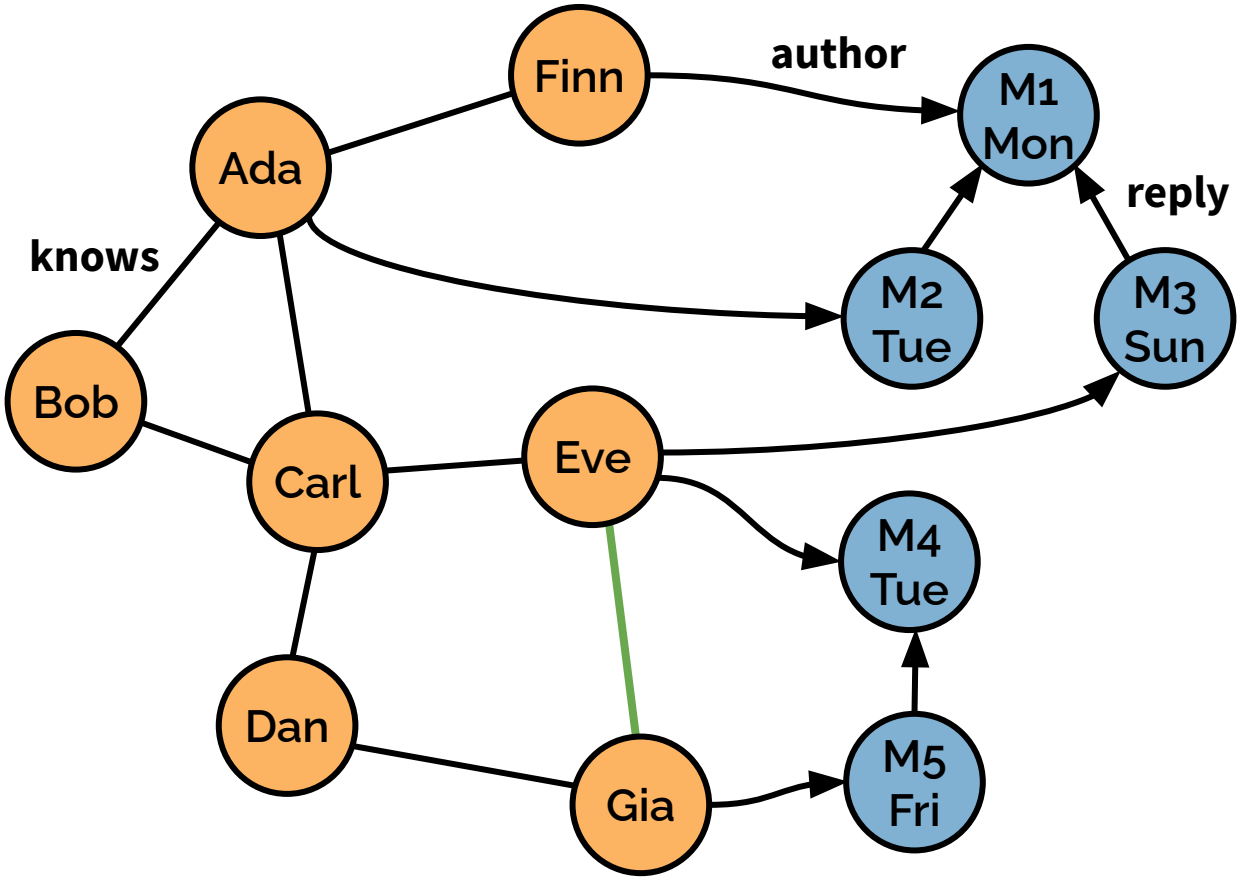
- ***Uniform random parameters*** result in unstable distributions
- ***Parameters selected*** using the statistics results tighter distributions



Data set | Queries | **Updates**



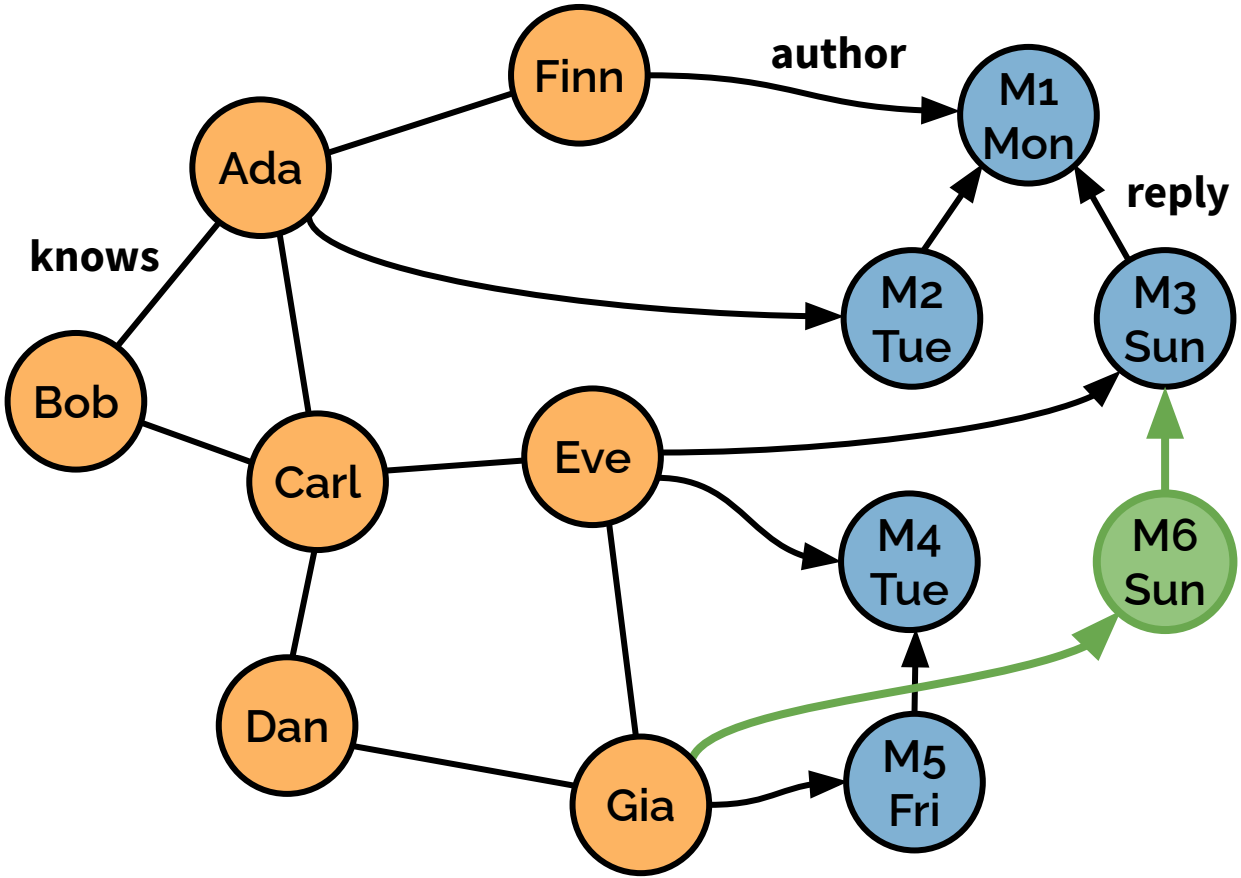
Data set | Queries | **Updates**



Updates

+ knows(“Eve”, “Gia”)

Data set | Queries | **Updates**



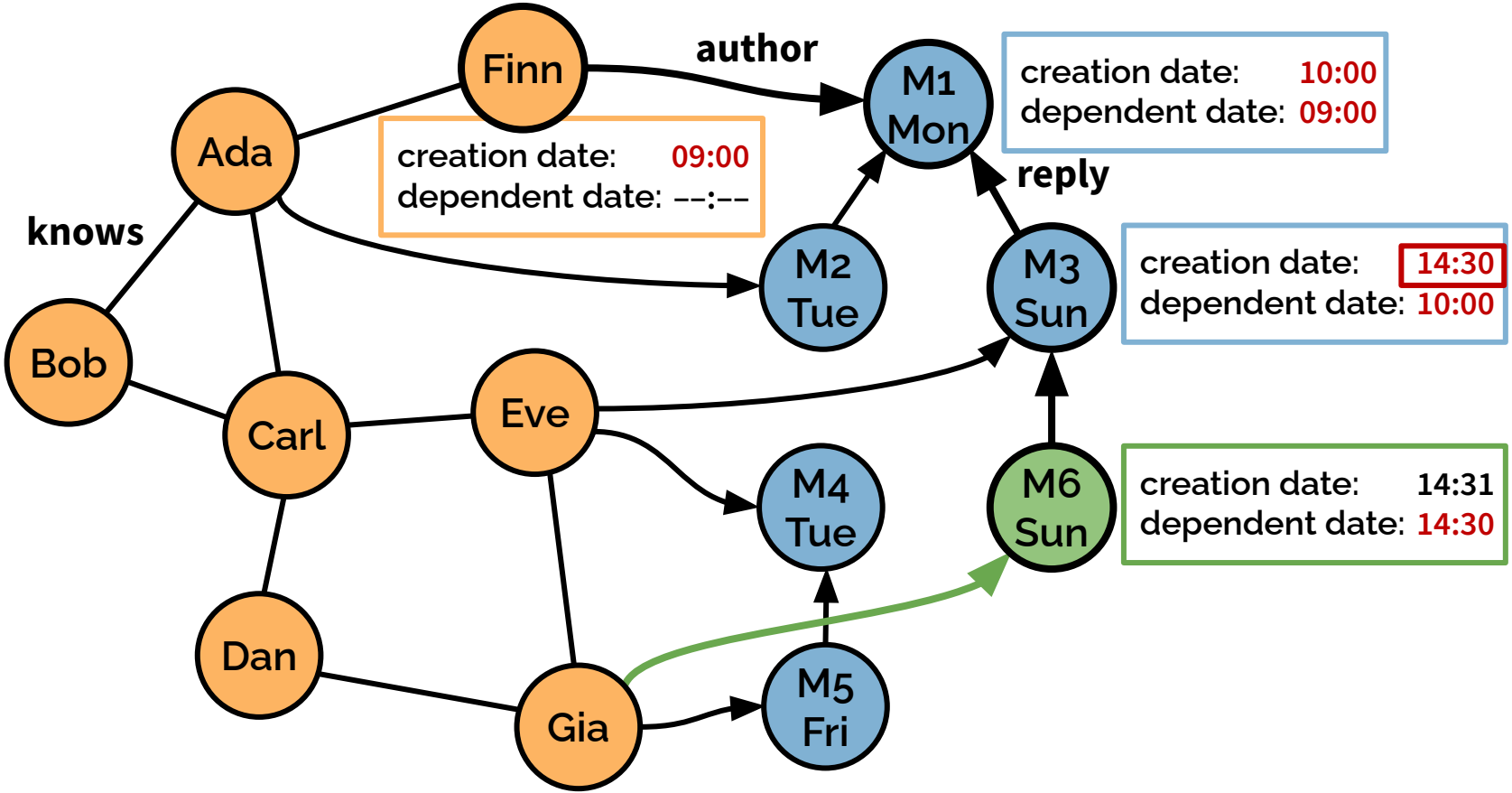
Updates

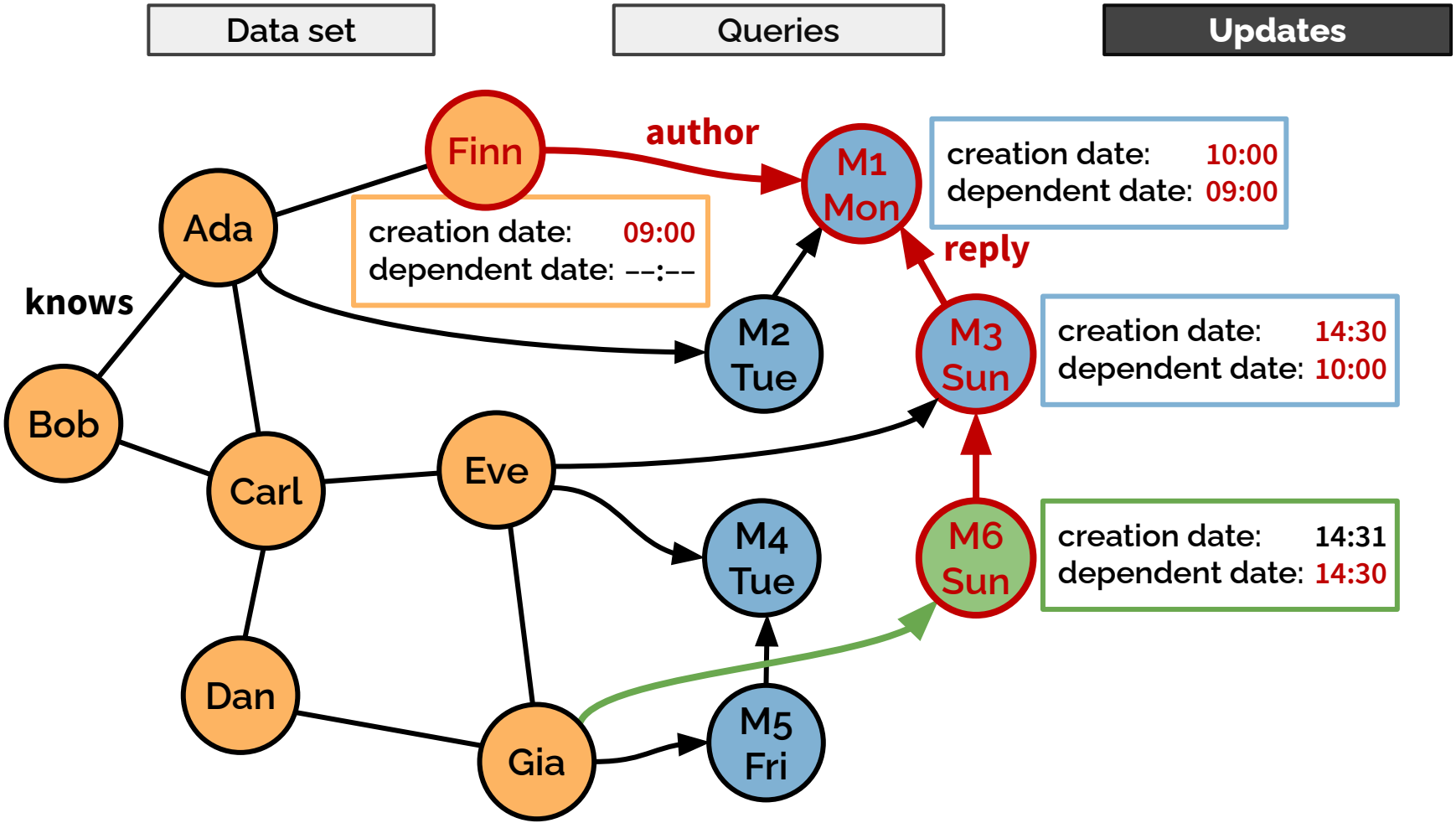
+ knows("Eve", "Gia")

+ Comment("Gia", "M3")

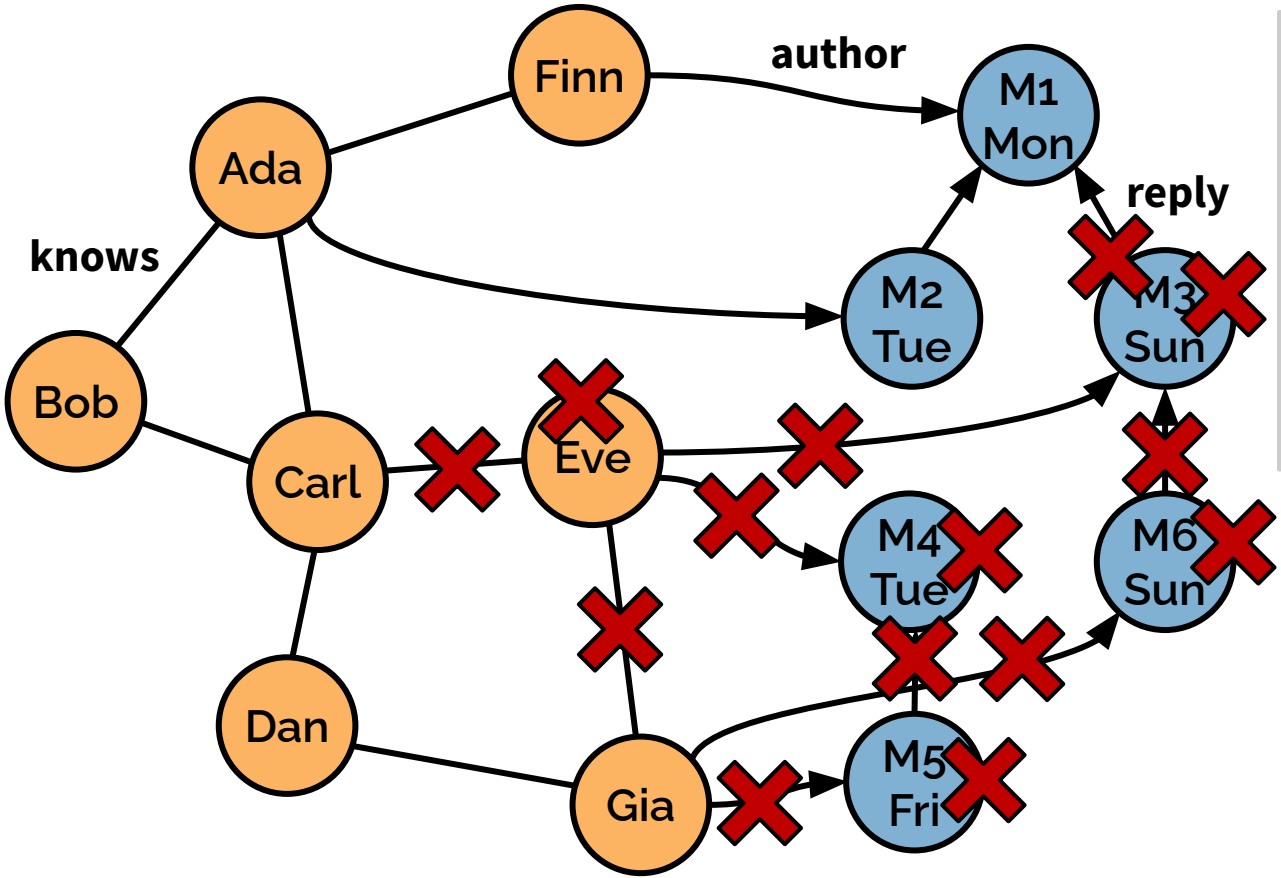
When is this operation executable?

Data set | Queries | **Updates**




Data set
Queries
Updates
Finn
author
M1
Mon
creation date: 10:00
dependent date: 09:00
Ada
creation date: 09:00
dependent date: --:--
reply
knows
M2
Tue
M3
Sun
creation date: 14:30
dependent date: 10:00
Bob
Carl
Eve
M4
Tue
M6
Sun
creation date: 14:31
dependent date: 14:30
Dan
Gia
M5
Fri

Data set | Queries | **Updates**



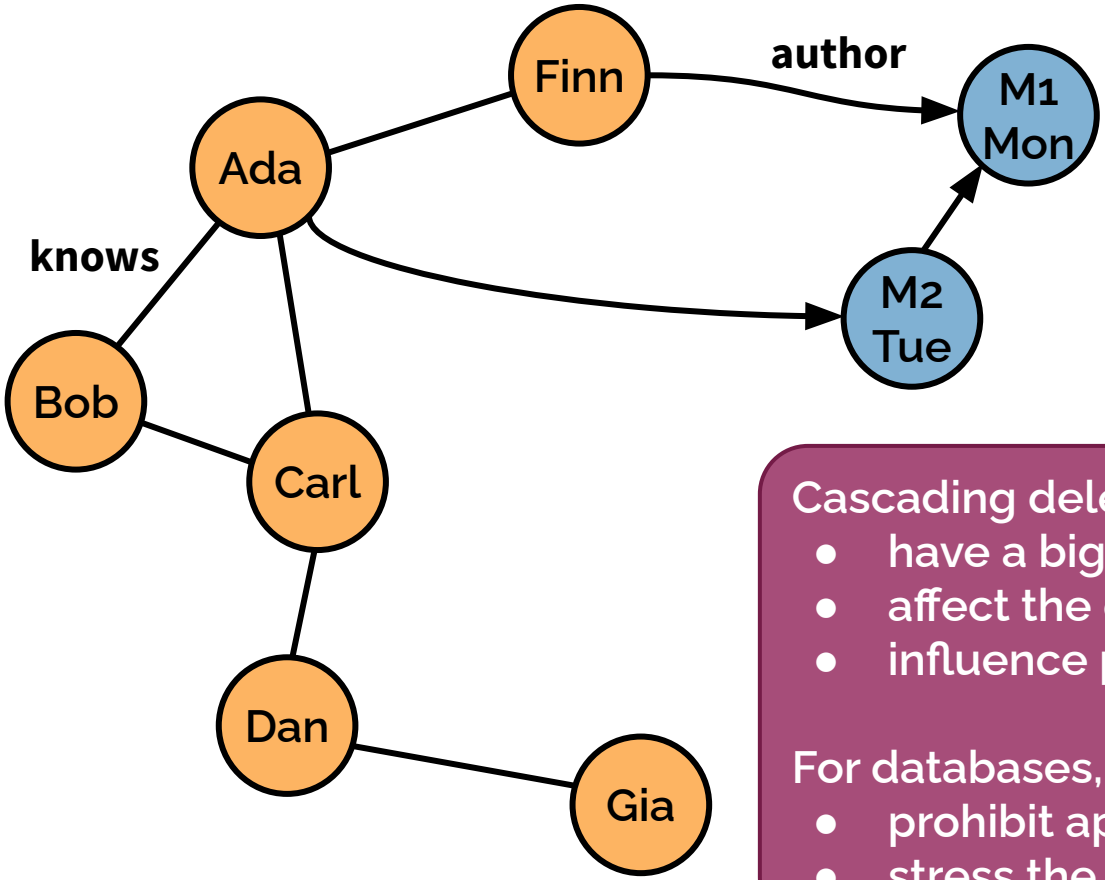
Updates

+ knows("Eve", "Gia")

+ Comment("Gia", "M3")

– Person("Eve")

Data set | Queries | **Updates**



Updates

+ knows(“Eve”, “Gia”)

+ Comment(“Gia”, “M3”)

– Person(“Eve”)

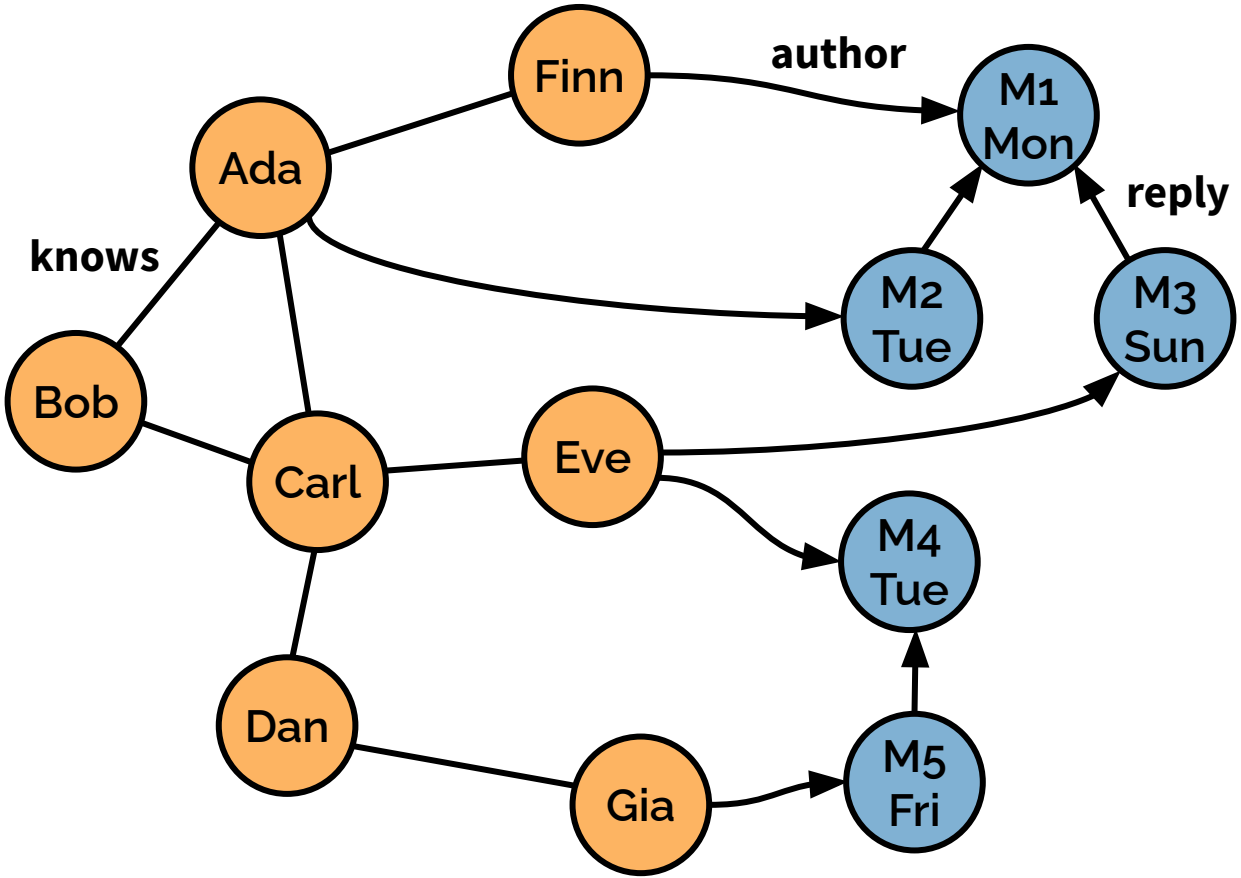
Cascading deletes remove lots of entities:

- have a big impact on the data distribution
- affect the executability of operations
- influence parameter selection

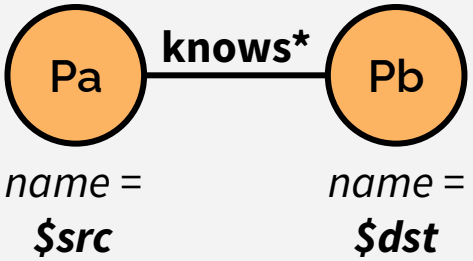
For databases, deletes:

- prohibit append-only data structures
- stress the garbage collector

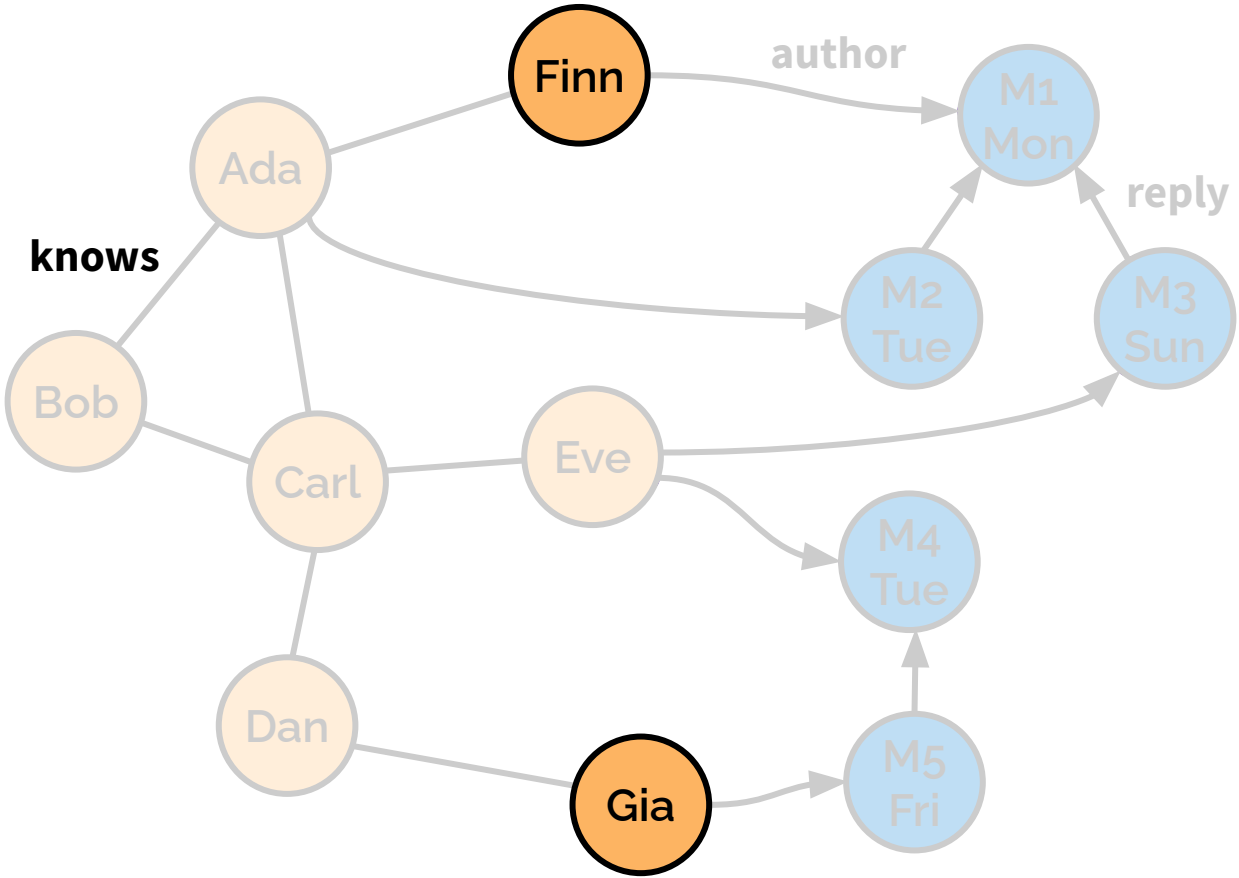
Data set | **Queries** | Updates



Q13($src, $dst)



Data set | **Queries** | Updates



Q13(*"Finn"*, *"Gia"*)

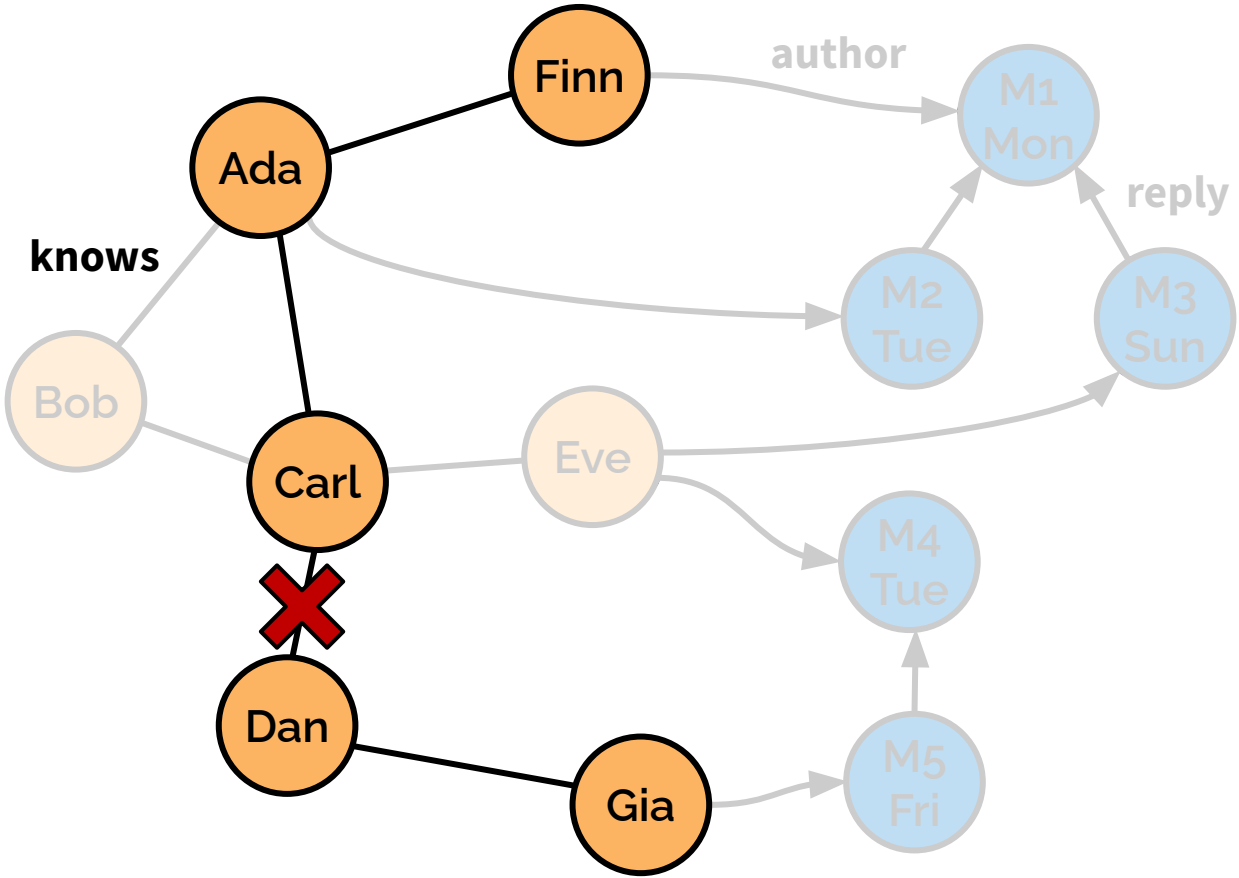
name = ***"Finn"***

name = ***"Gia"***

Data set

Queries

Updates



Q13(*"Finn"*, *"Gia"*)

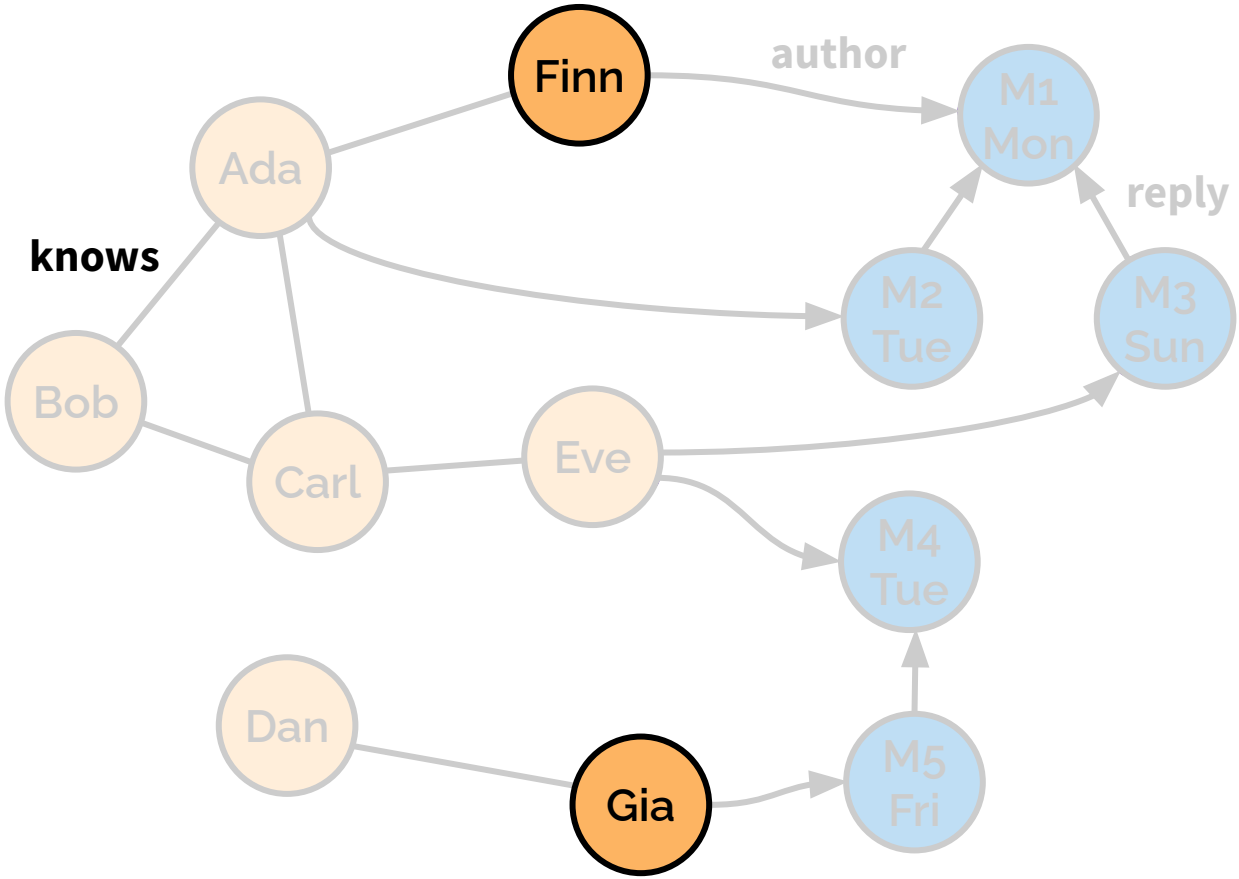
name = ***"Finn"***

name = ***"Gia"***

Updates

– knows("Carl", "Dan")

Data set | **Queries** | Updates



Q13(*"Finn"*, *"Gia"*)

name = ***"Finn"***

name = ***"Gia"***

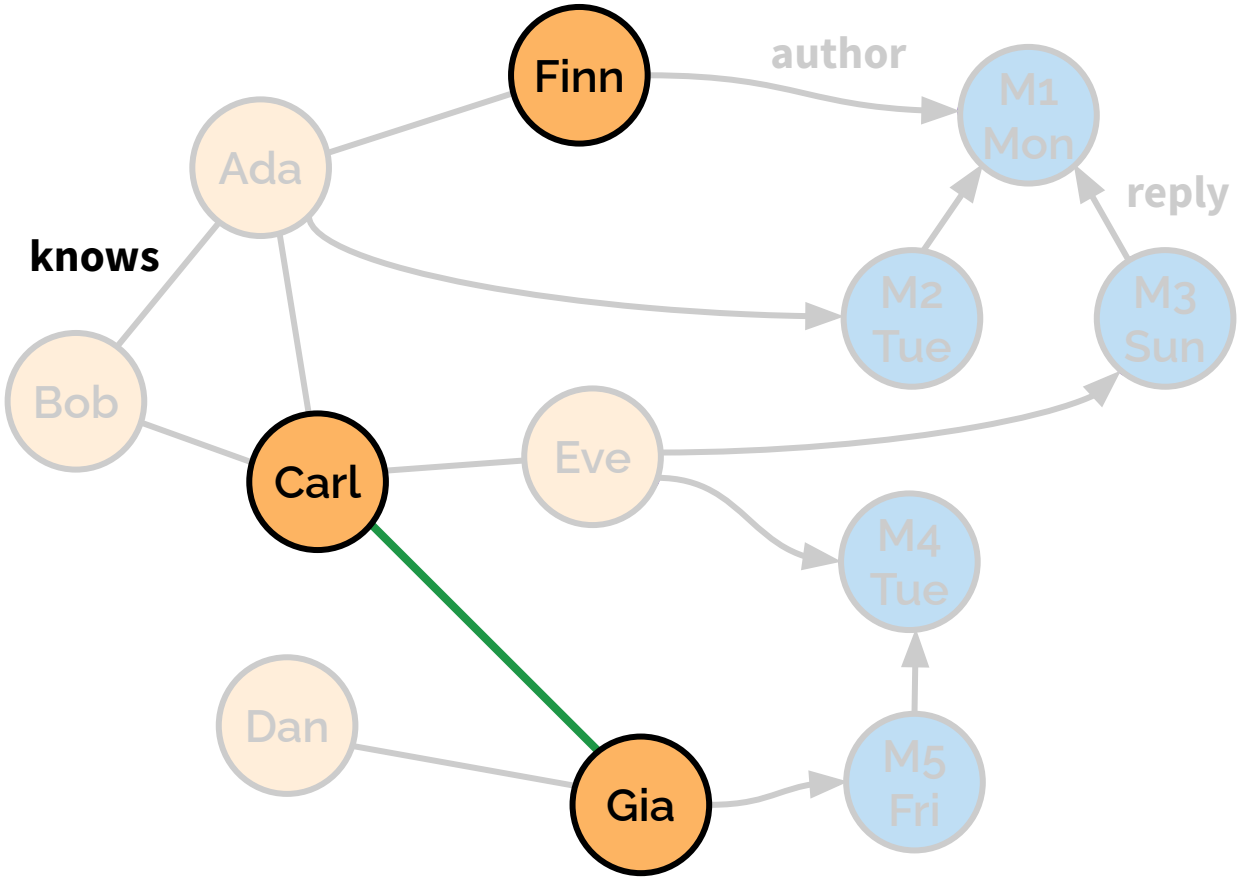
Updates

\- knows("Carl", "Dan")

Data set

Queries

Updates



Q13(*"Finn"*, *"Gia"*)

name = ***"Finn"***

name = ***"Gia"***

Updates

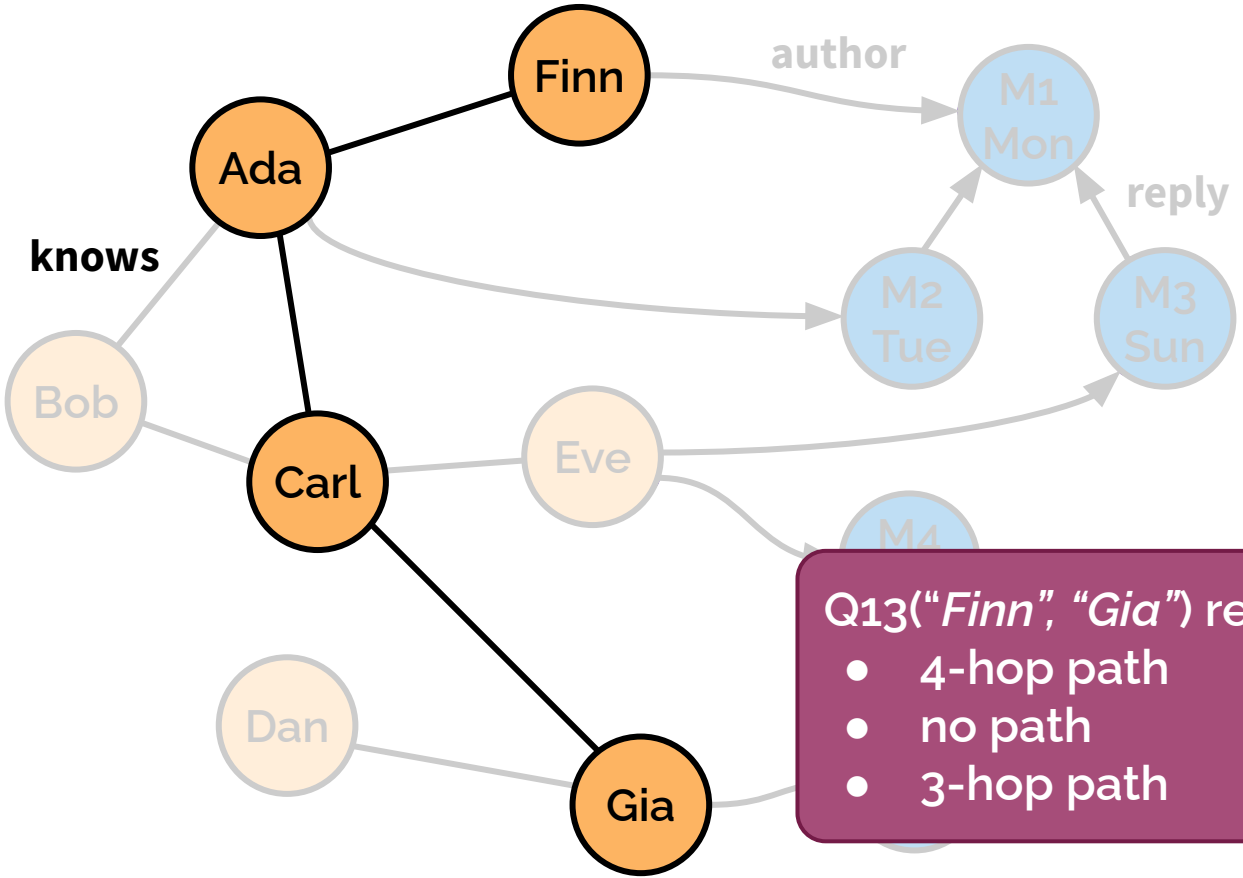
− knows("Carl", "Dan")

+ knows("Carl", "Gia")

Data set

Queries

Updates



Q13(*"Finn"*, *"Gia"*)

name = ***"Finn"***

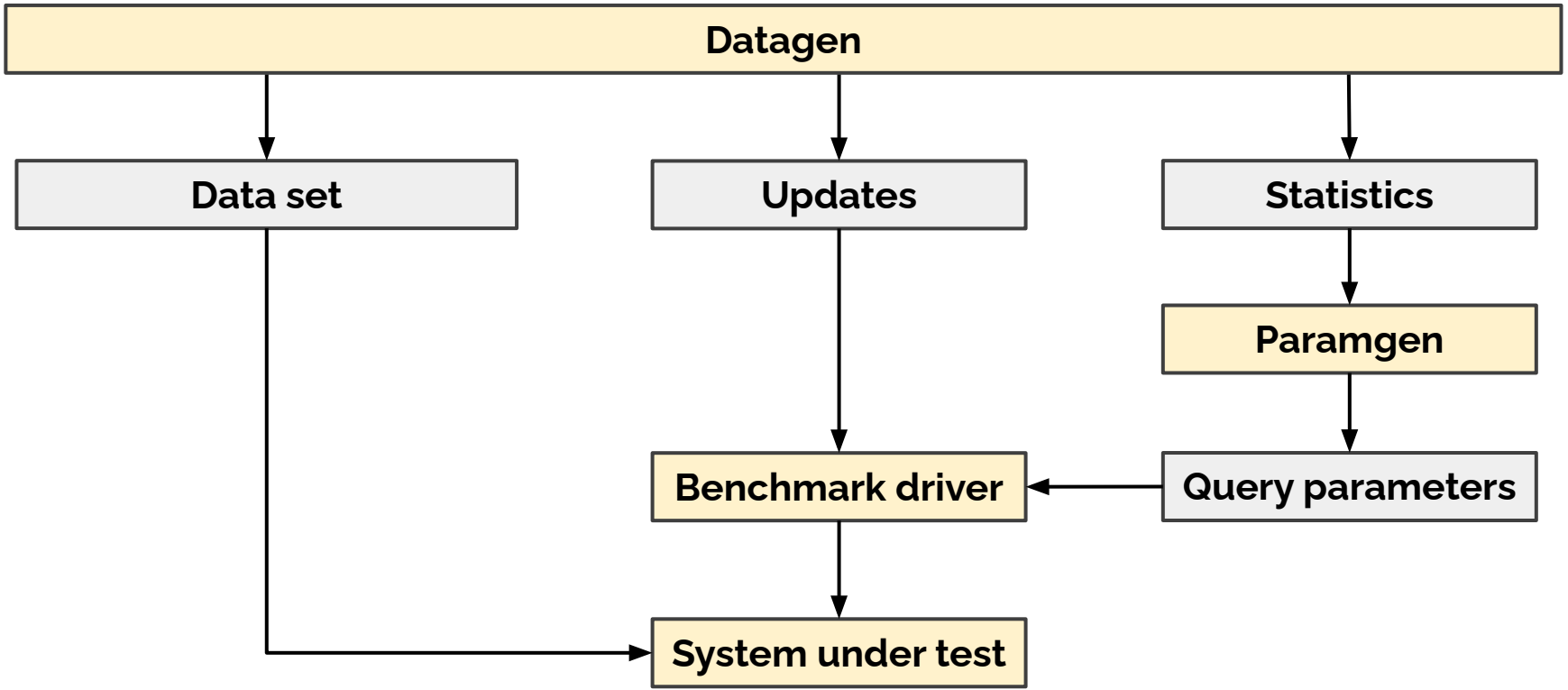
name = ***"Gia"***

Updates

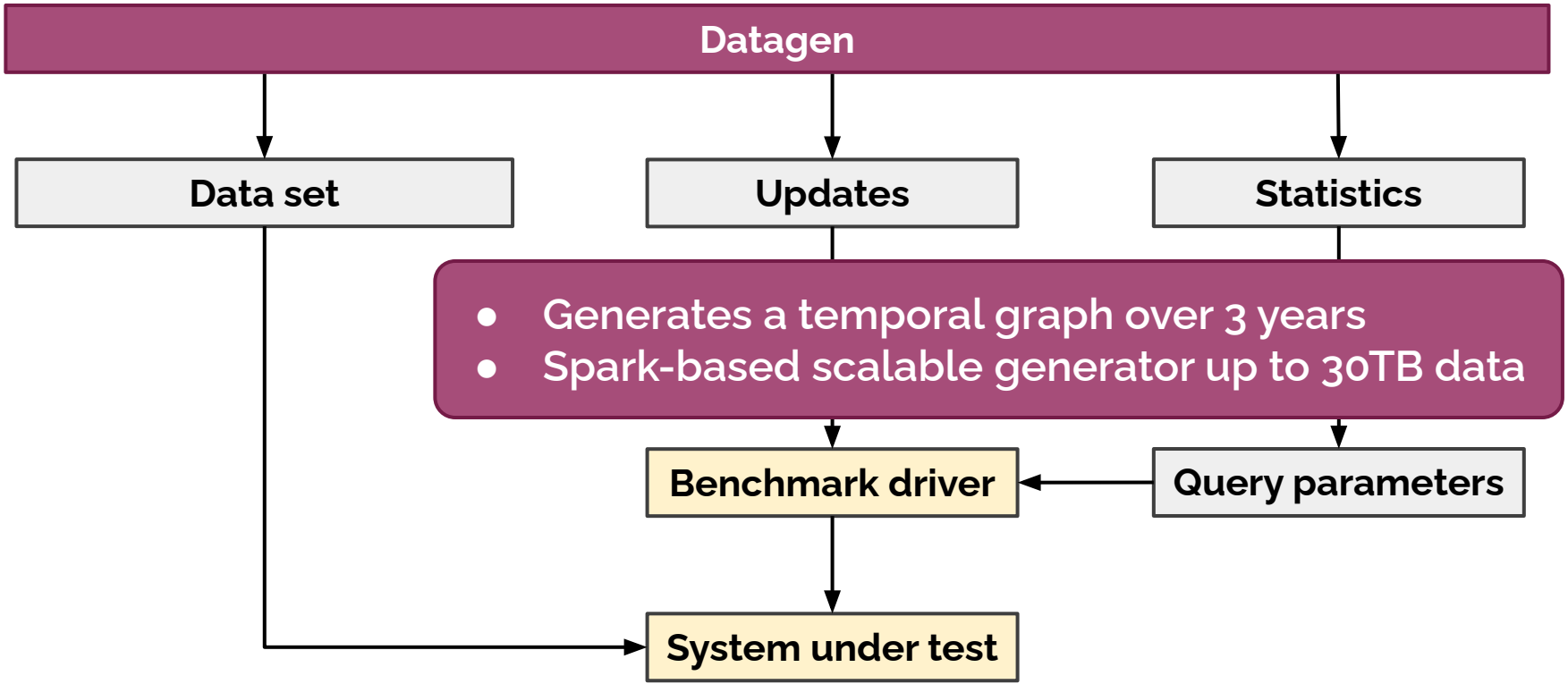
Q13(*"Finn"*, *"Gia"*) results change over time:

- 4-hop path
- no path
- 3-hop path

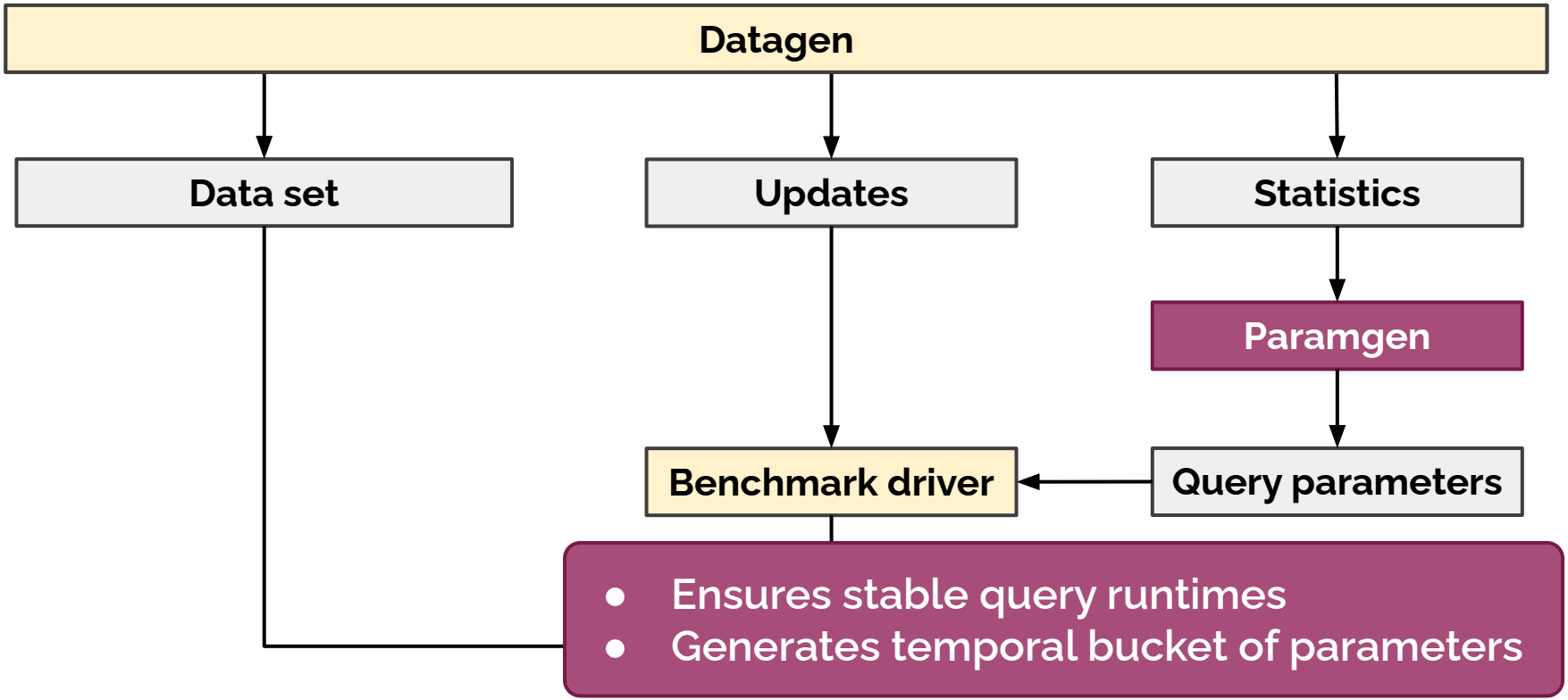
Benchmark workflow



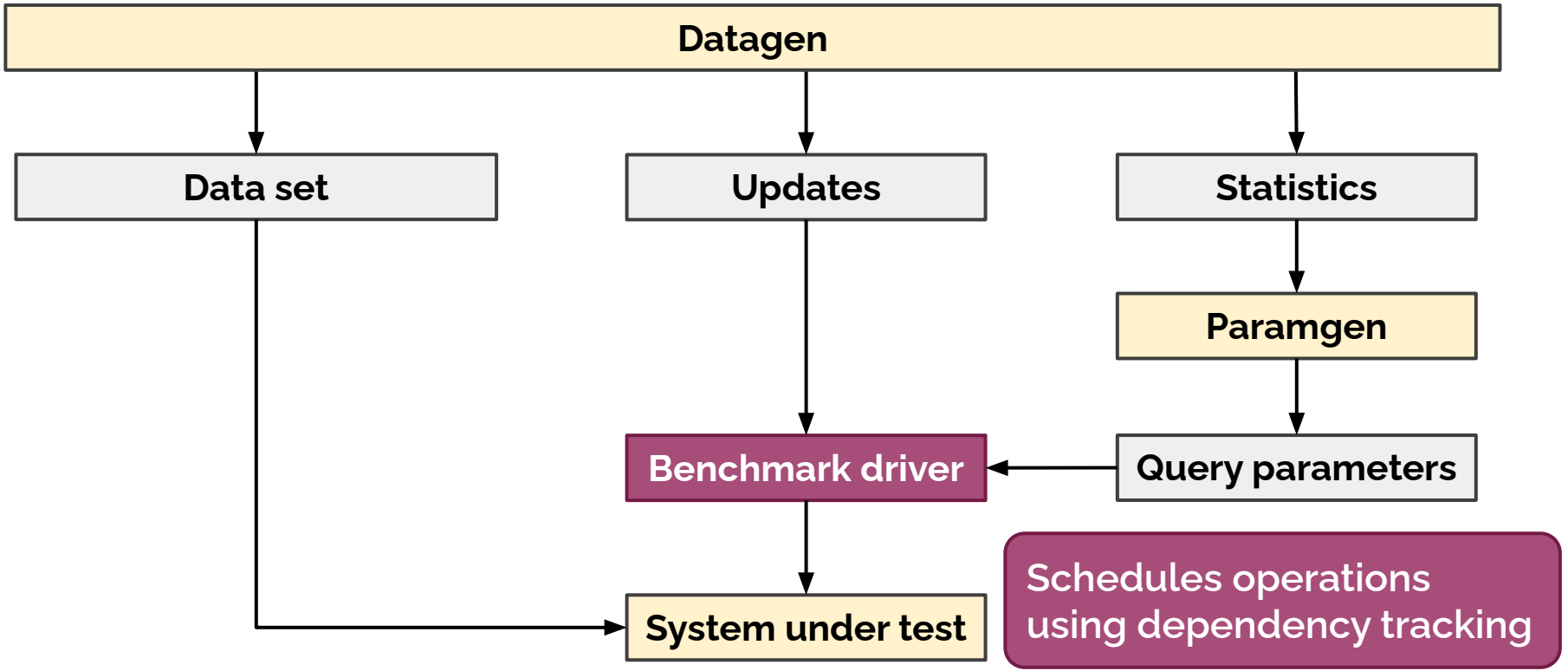
Benchmark workflow



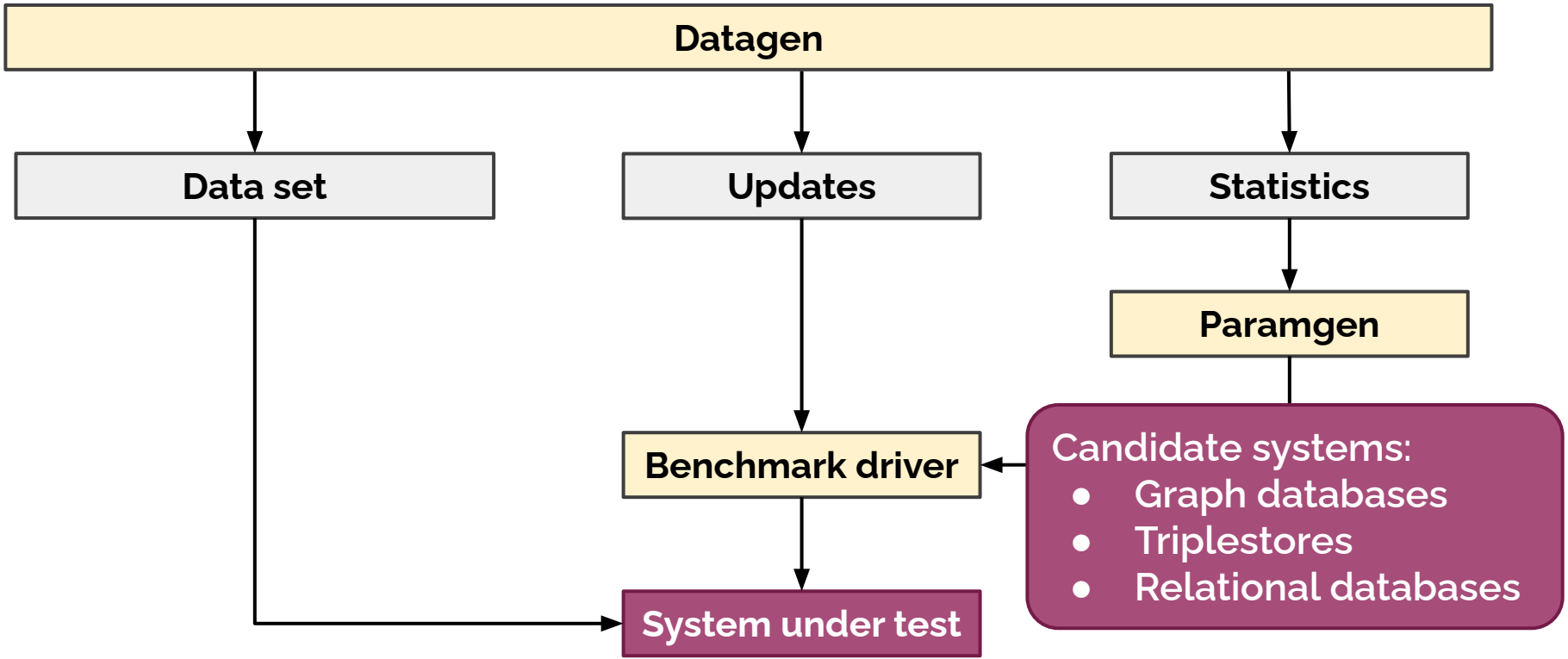
Benchmark workflow



Benchmark workflow

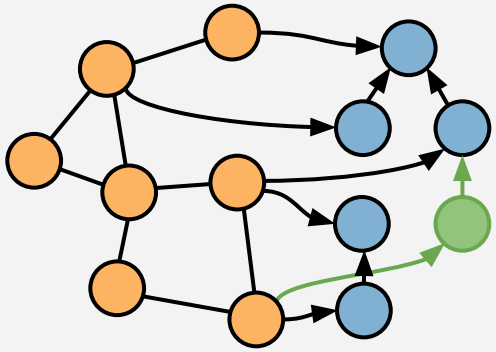


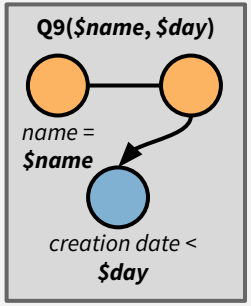
Benchmark workflow

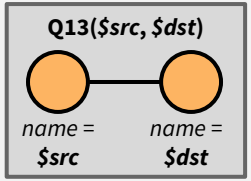


Social Network Benchmark Workloads

SNB Interactive





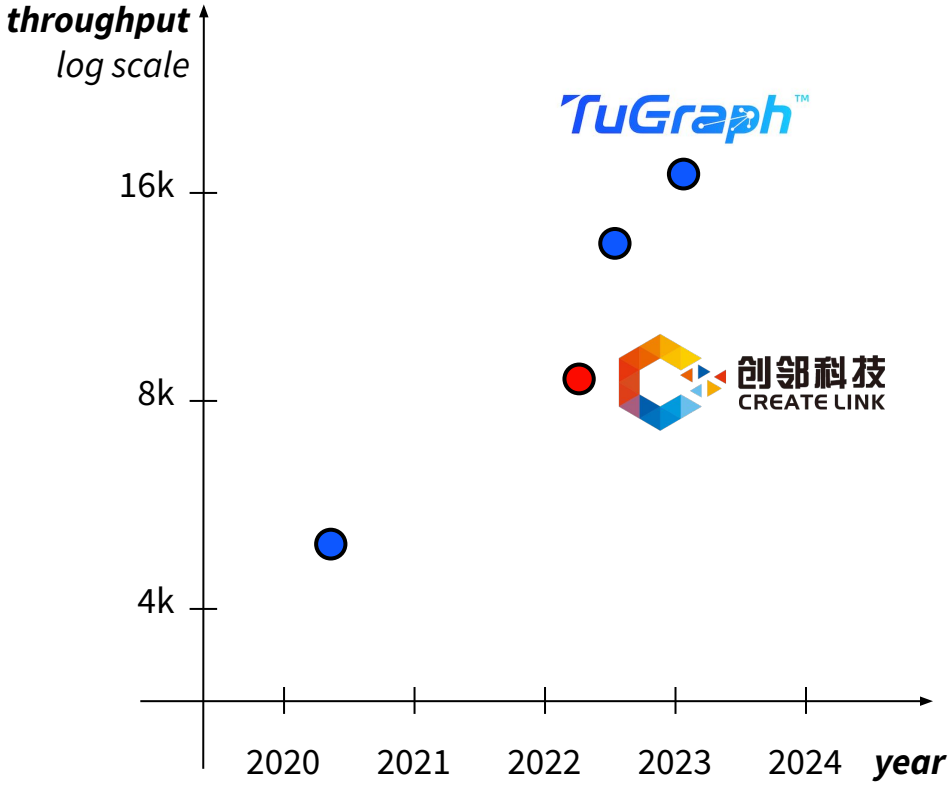


Queries start in 1–2 person nodes

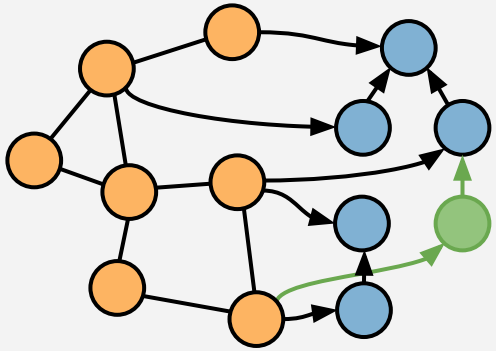
Queries and updates run concurrently

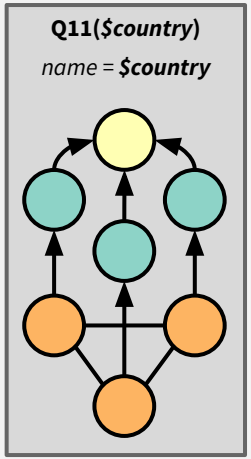
Goal: high throughput (ops/s)

Results on the 100GB data set



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

TigerGraph AMD

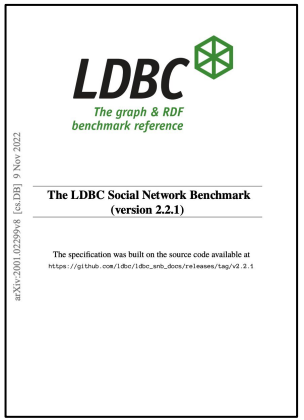
- Power@SF: 30,990
- Throughput@SF: 12,993

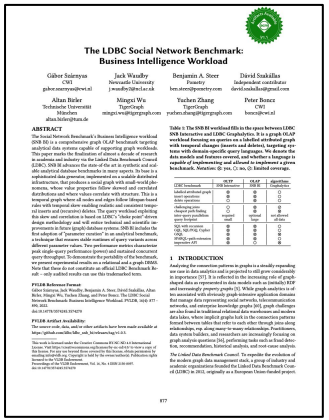
More and larger-scale results expected in 2023

Benchmark process

For each workload:

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





Guidelines:

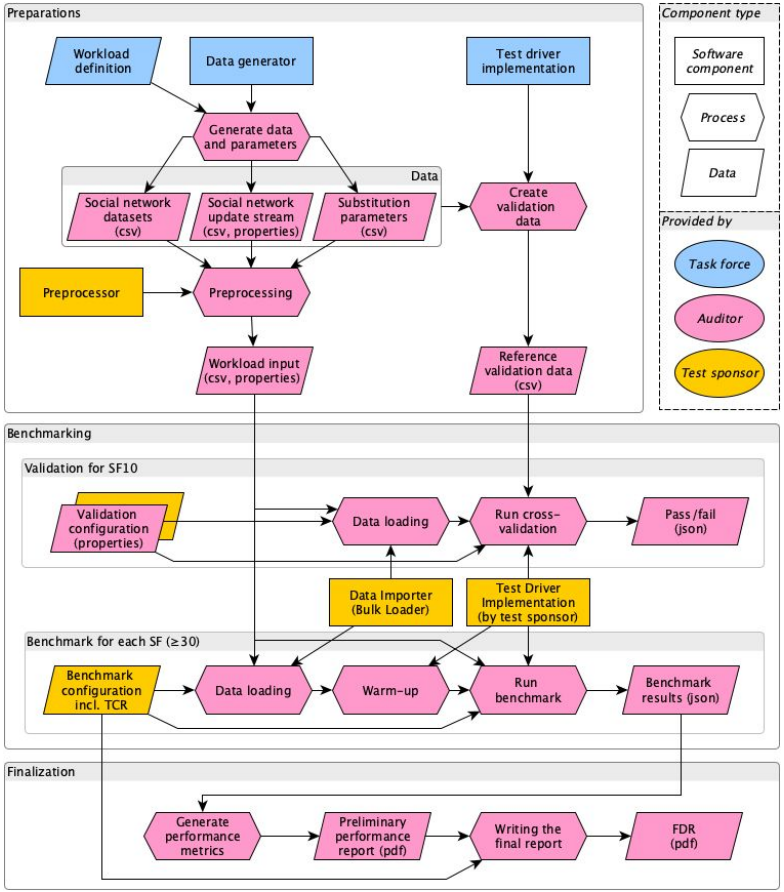
- How to execute the benchmark correctly
- Validate the results
- Verify ACID-compliance

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

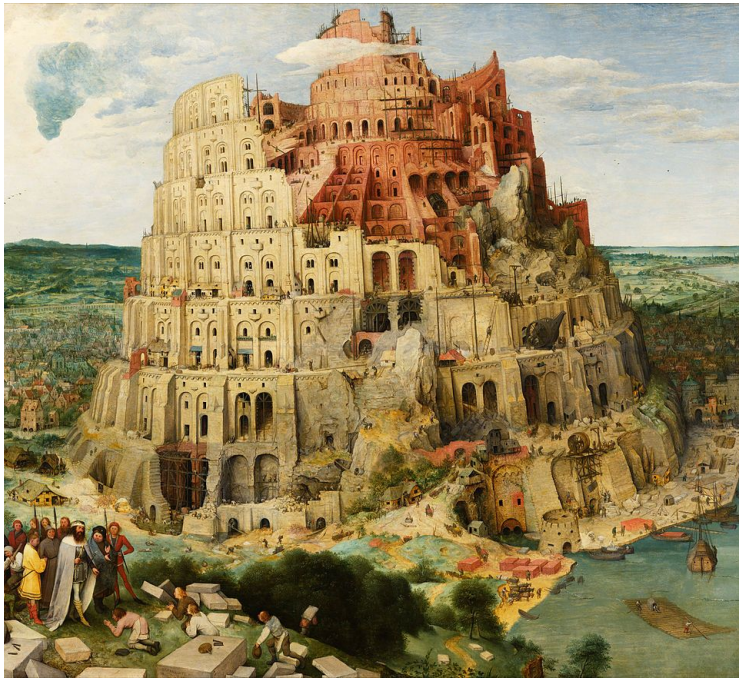
ArangoDB
AQL

Vaticle TypeDB
TypeQL

JanusGraph
Gremlin

NebulaGraph
nGQL

XTDB
Datalog



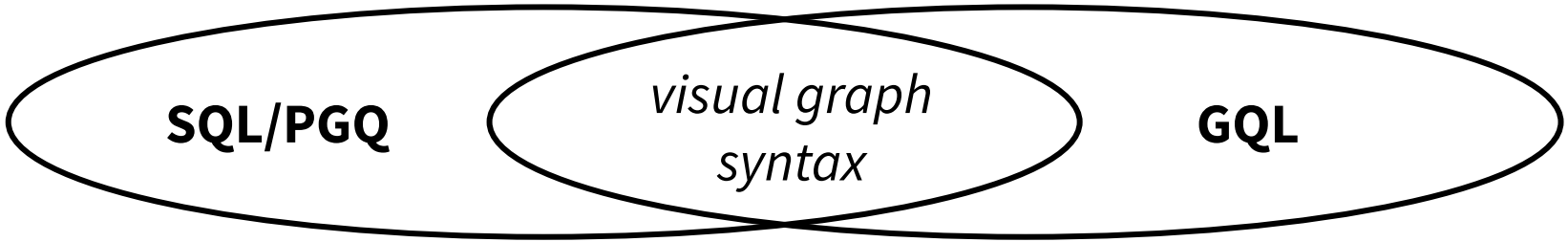
New ISO standard query languages

relational operators

path finding

pattern matching

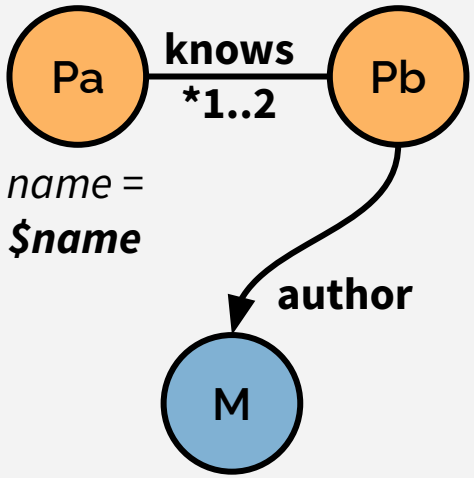
- **SQL/PGQ** (Property Graph Queries), June 2023
- **GQL** (Graph Query Language), March 2024



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
 UNION
 SELECT k2.p2id AS id
 FROM person Pa,
      knows k1,
      knows k2
 WHERE Pa.name = $name
   AND Pa.id = k1.p1id
   AND k1.p2id = k2.p1id
   AND k1.p1id <> k2.p2id
 ) Pb,
 Message m
WHERE Pb.id = m.authorId
  AND m.creationDate < $day
```

Q9(*$name*, *$day*)



SQL/PGQ

```
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

```
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
```

SQL/PGQ

```
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))
```

Q13($src, $dst)

SQL

```
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

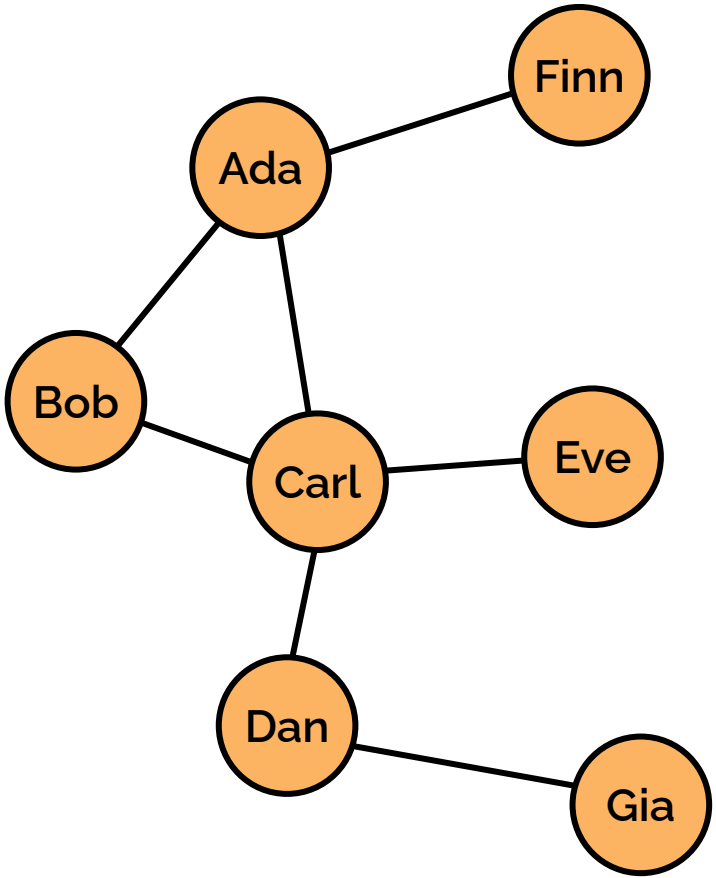
LDBC conducts research on graph schemas:

- 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

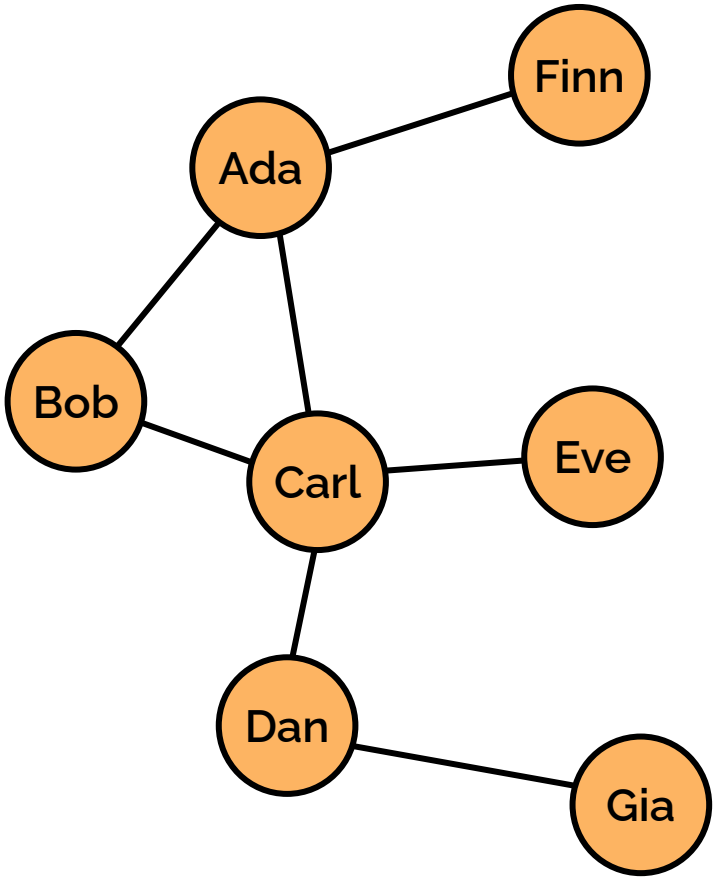


Algorithms

untyped, unattributed graphs

LDBC SNB
Graph500
Twitter
Friendster
Patents
wiki-Talk

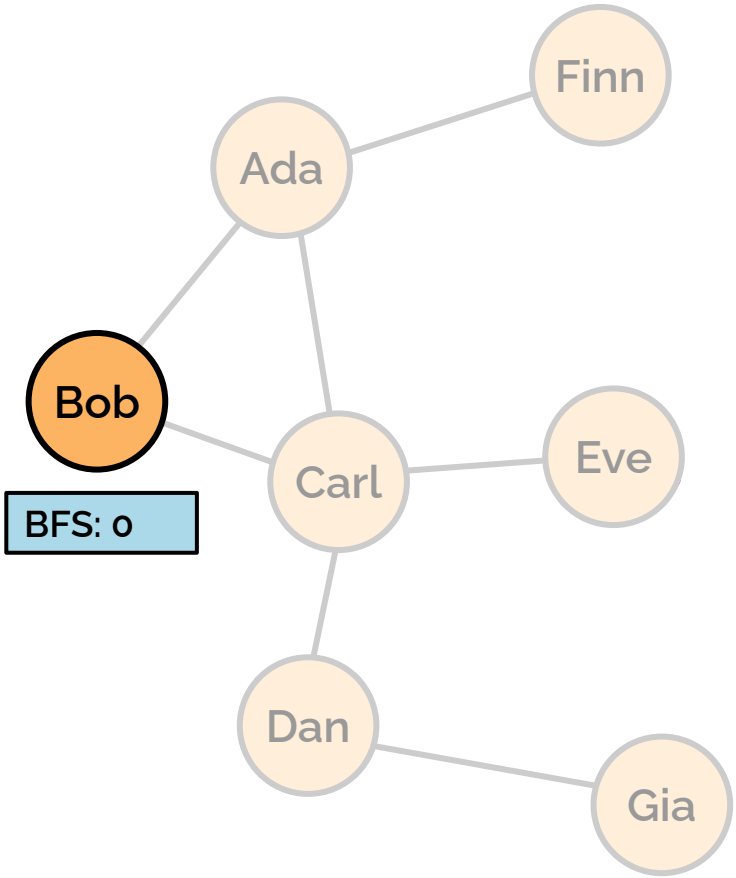
Data set



Algorithms

Graphalytics algorithms

Data set



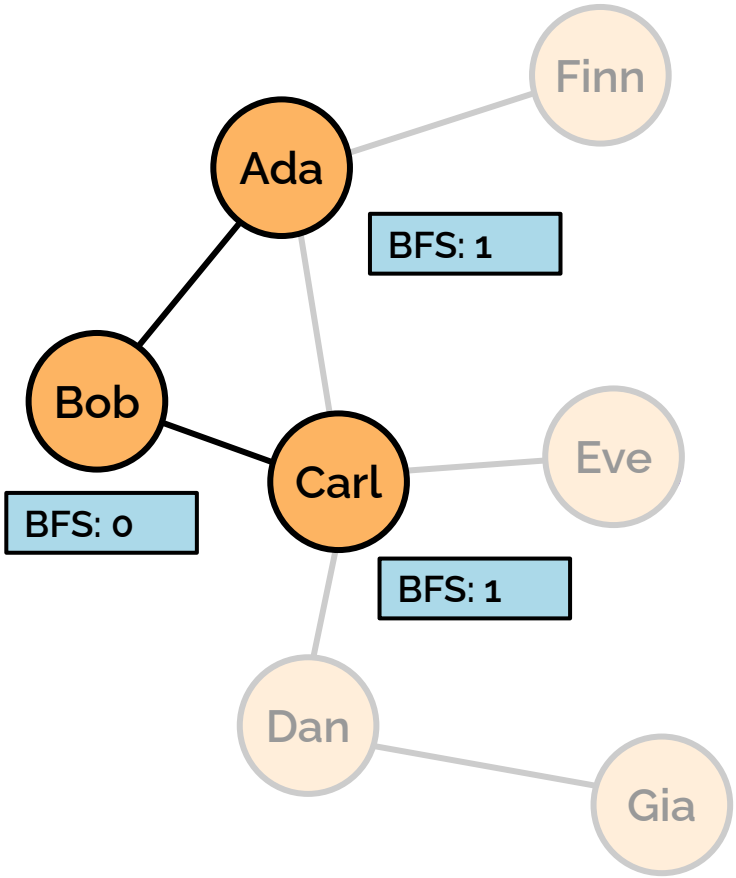
BFS: 0

Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

Data set

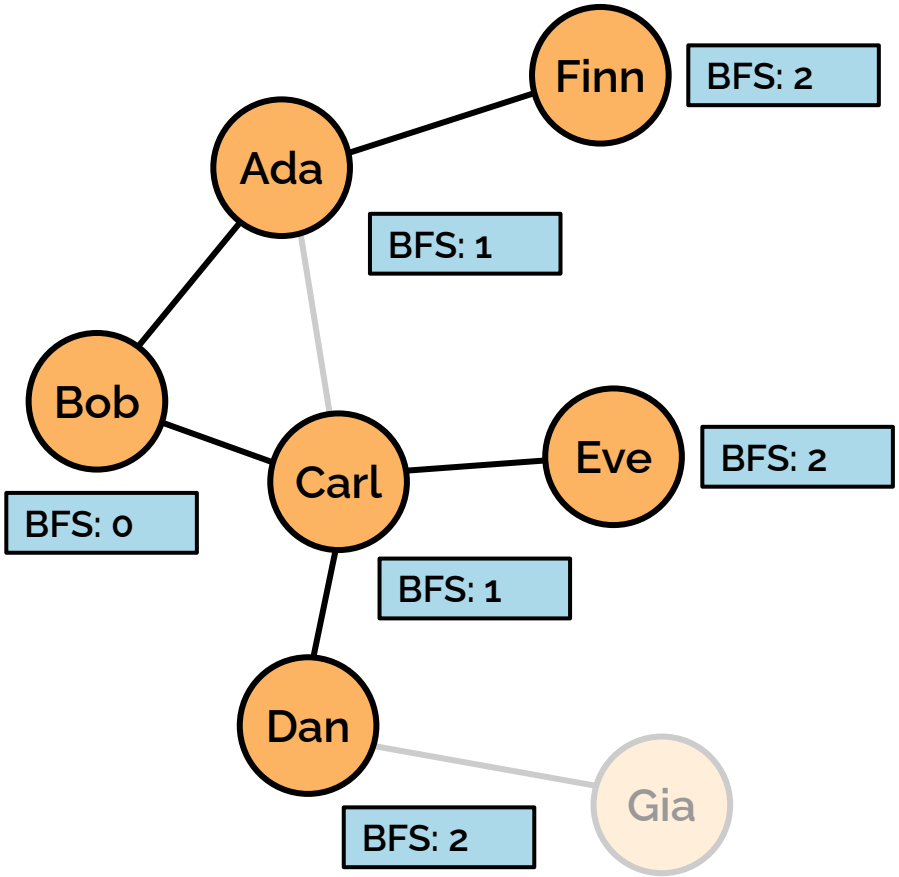


Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

Data set

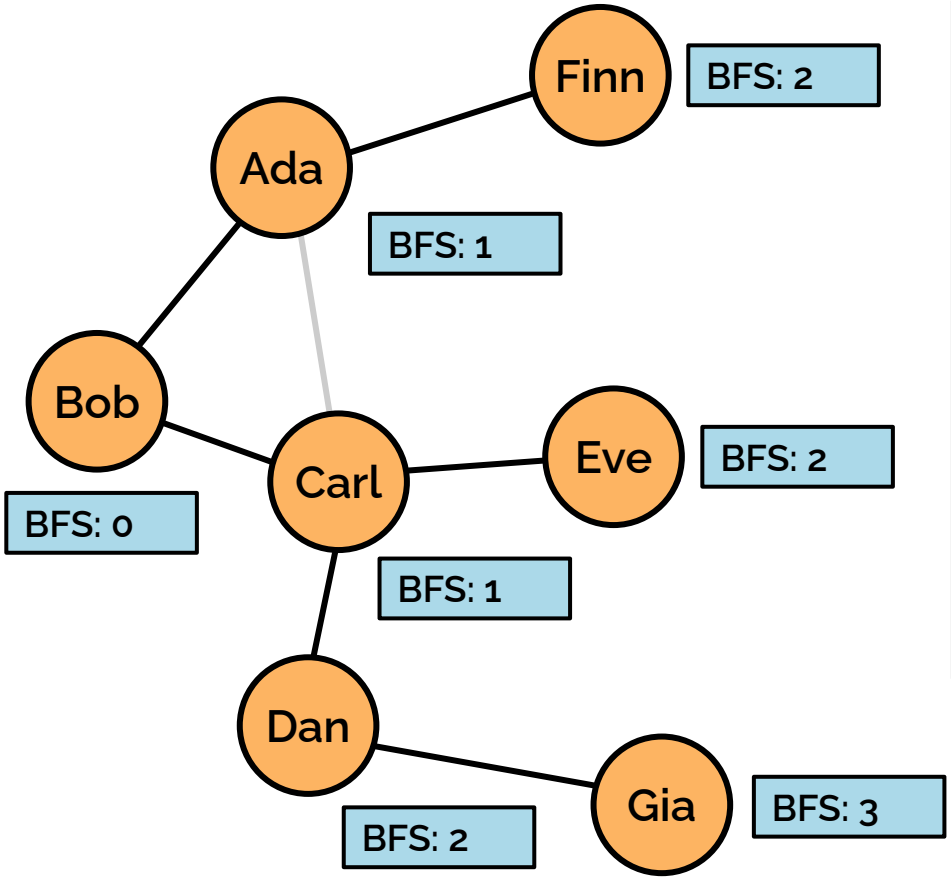


Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

Data set

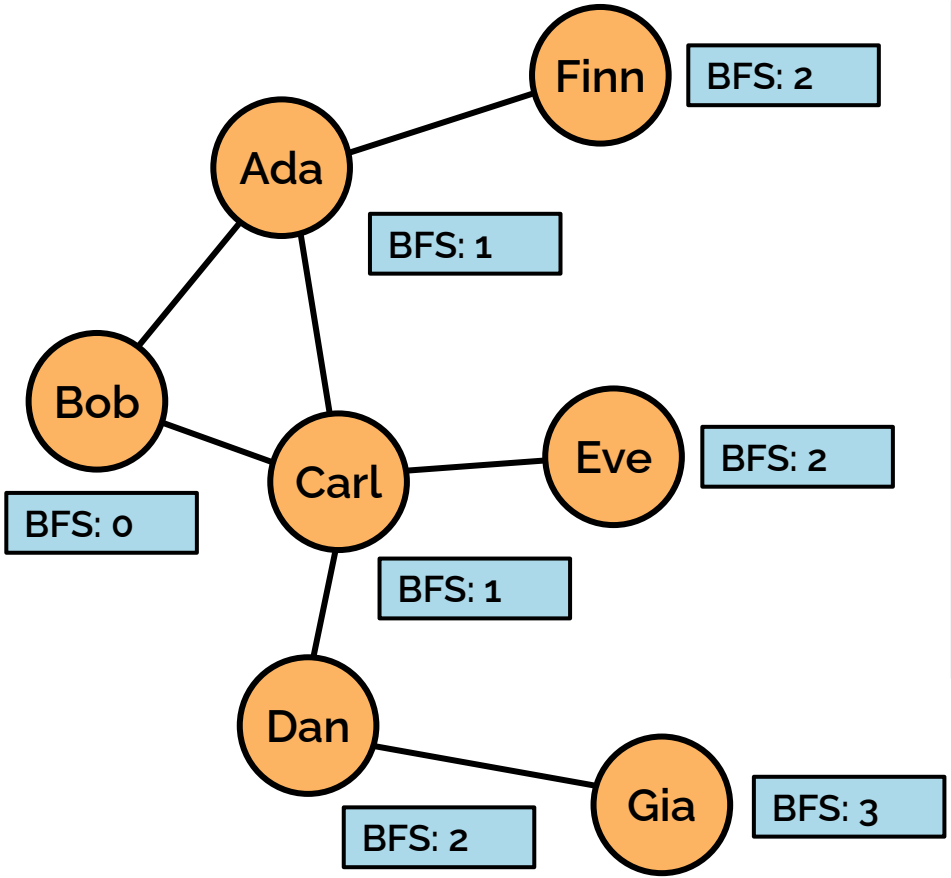


Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

Data set



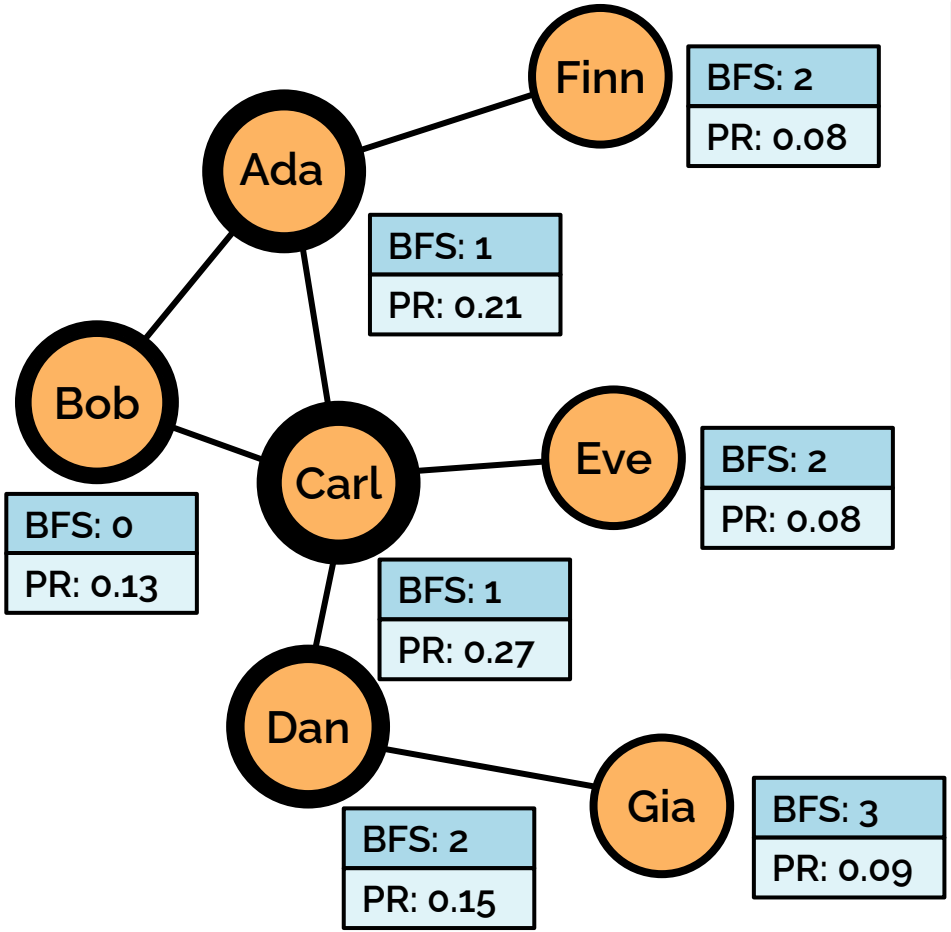
Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

PageRank(*damping factor*: 0.85, *iterations*: 5)

Data set



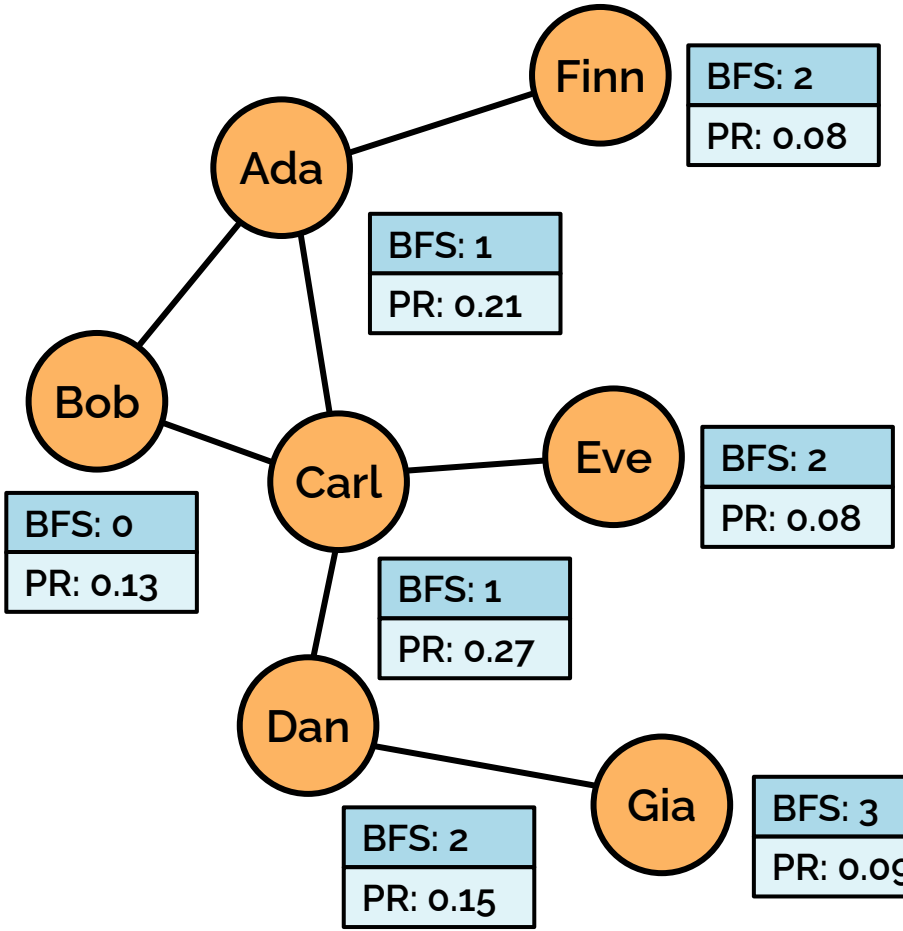
Algorithms

Graphalytics algorithms

Breadth-first search(*source*: “Bob”)

PageRank(*damping factor*: 0.85, *iterations*: 5)

Data set



Algorithms

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:

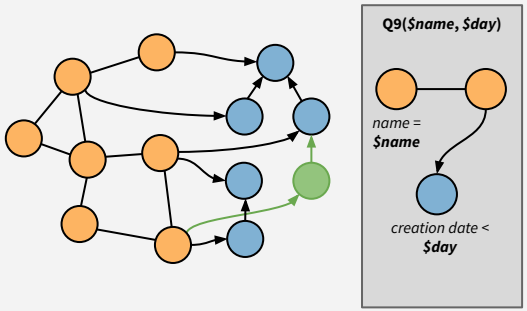
- Participate in benchmark design & research
- Commission audits
- Gain early access to ISO standard drafts, SQL/PGQ and GQL

Pricing:

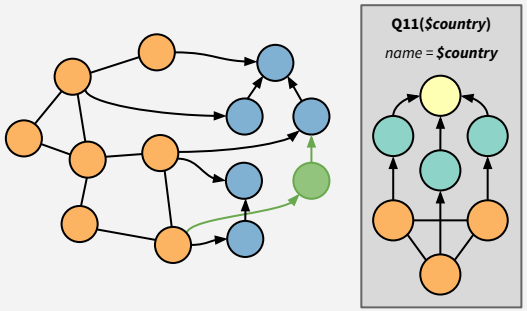
- Free for individuals
- 2,500 EUR/year for companies
- 10,000 EUR/year for sponsor companies

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

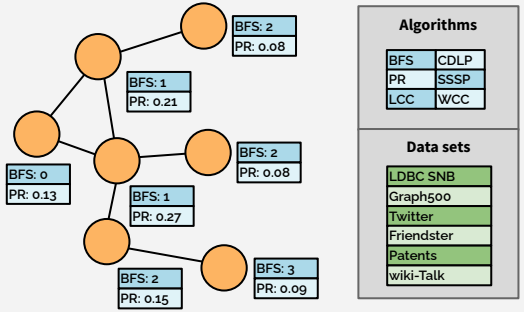
SNB Interactive



SNB Business Intelligence



Graphalytics



Semantic Publishing Benchmark

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

Financial Benchmark (to be released in 2023)

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

Future benchmark ideas

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


LDBC
The graph & RDF
benchmark reference