The LDBC benchmark suite

Gábor Szárnyas

LDBC TUC meeting | 2024-08-30 | Guangzhou & virtual

Inspiration: TPC benchmarks

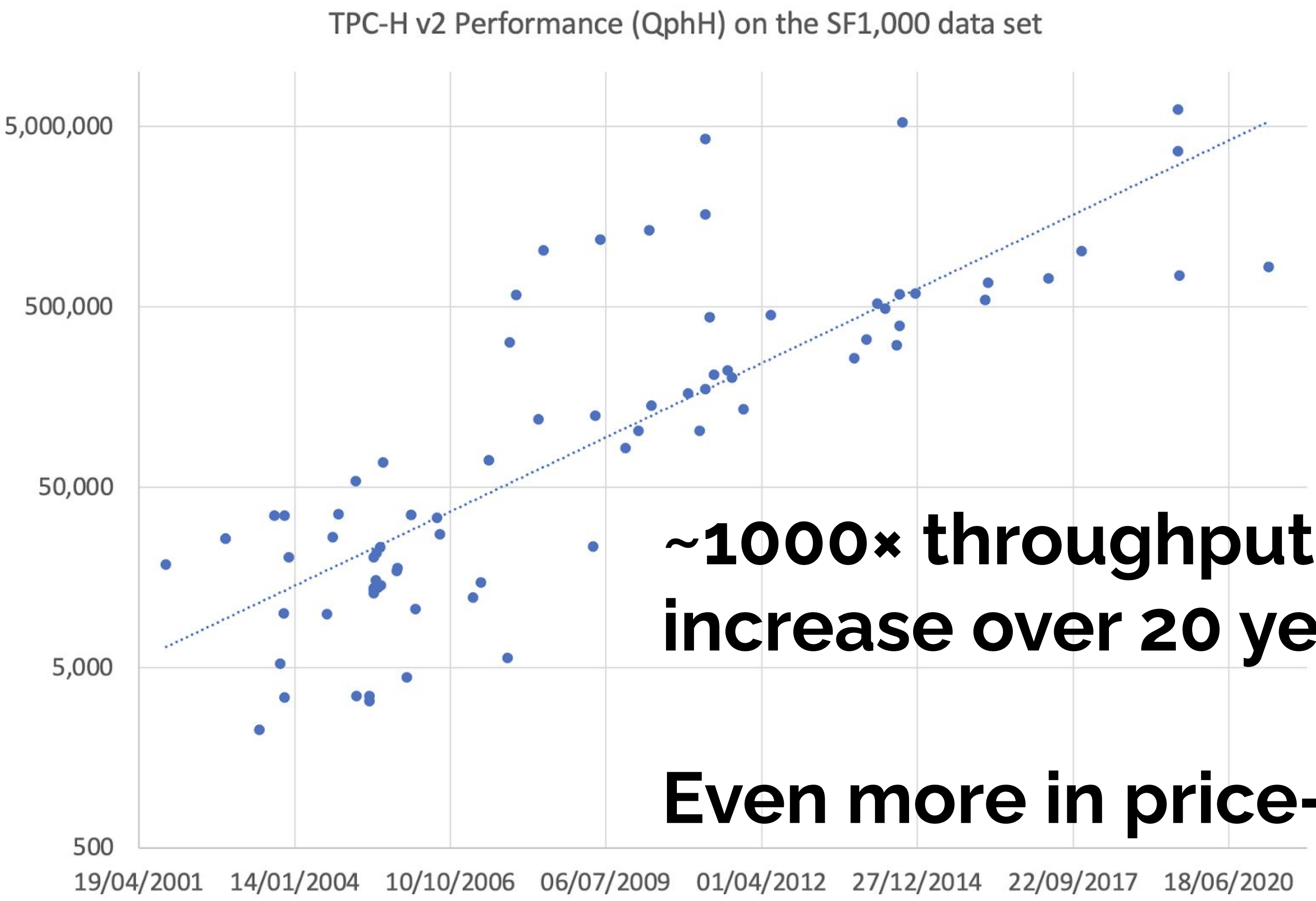
TPC: Transaction Processing Performance Council

Non-profit founded in 1988

Benchmark specifications

Stringent auditing process

Influential benchmarks: TPC-C, TPC-H, TPC-DS



~1000× throughput increase over 20 years

Even more in price-perf

LDBC benchmarks

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	**scale factors: SF30 = 30GiB CSV**	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

Similarities to TPC benchmarks

macro/application-level benchmarks	scale factors: SF30 = 30GiB CSV	flexible hardware and software setup
certified auditors	FDRs with metrics, e.g. throughput@SF	benchmark approval and renewal

The Social Network Benchmark (SNB) suite

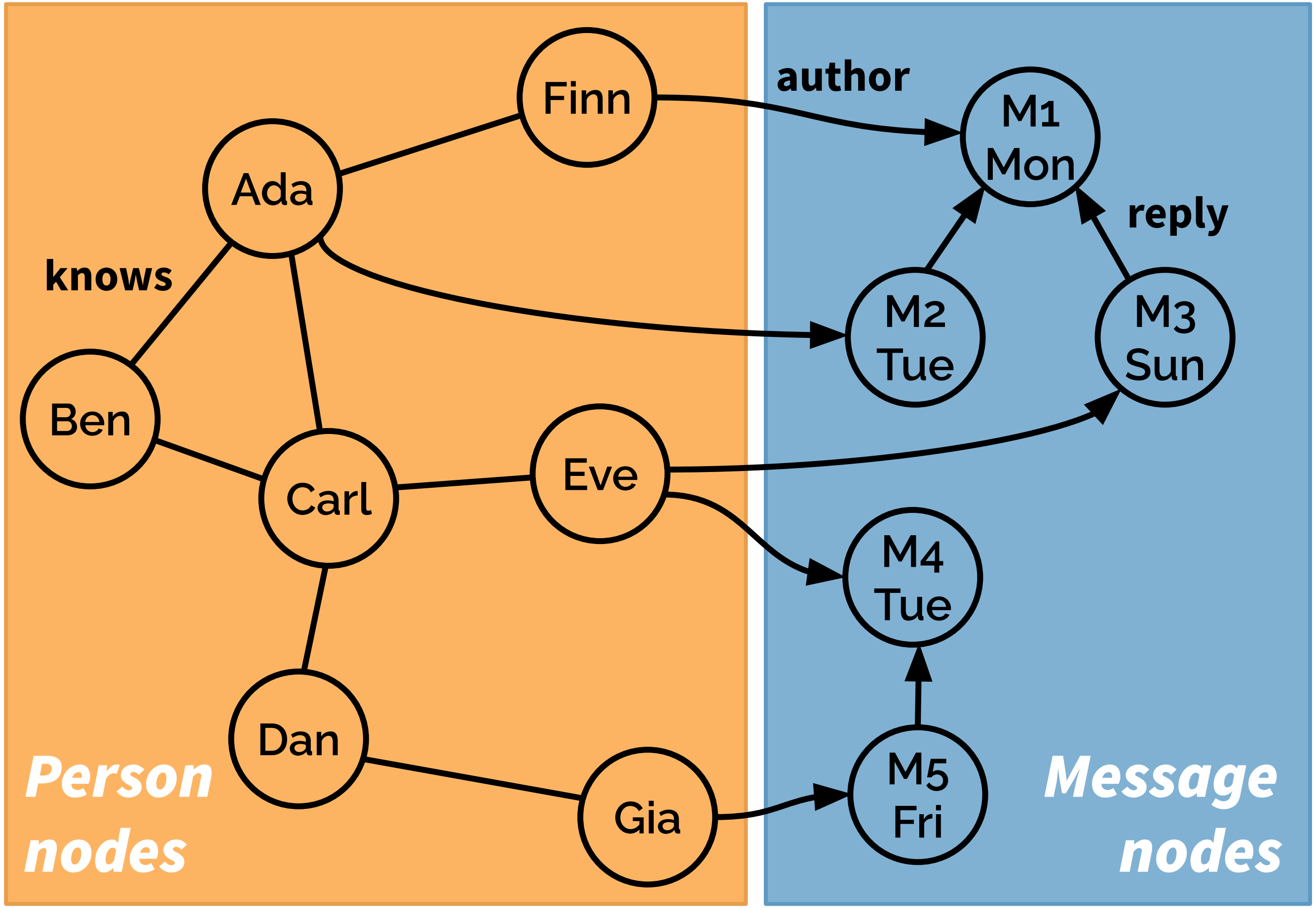
Data set and queries

Data set

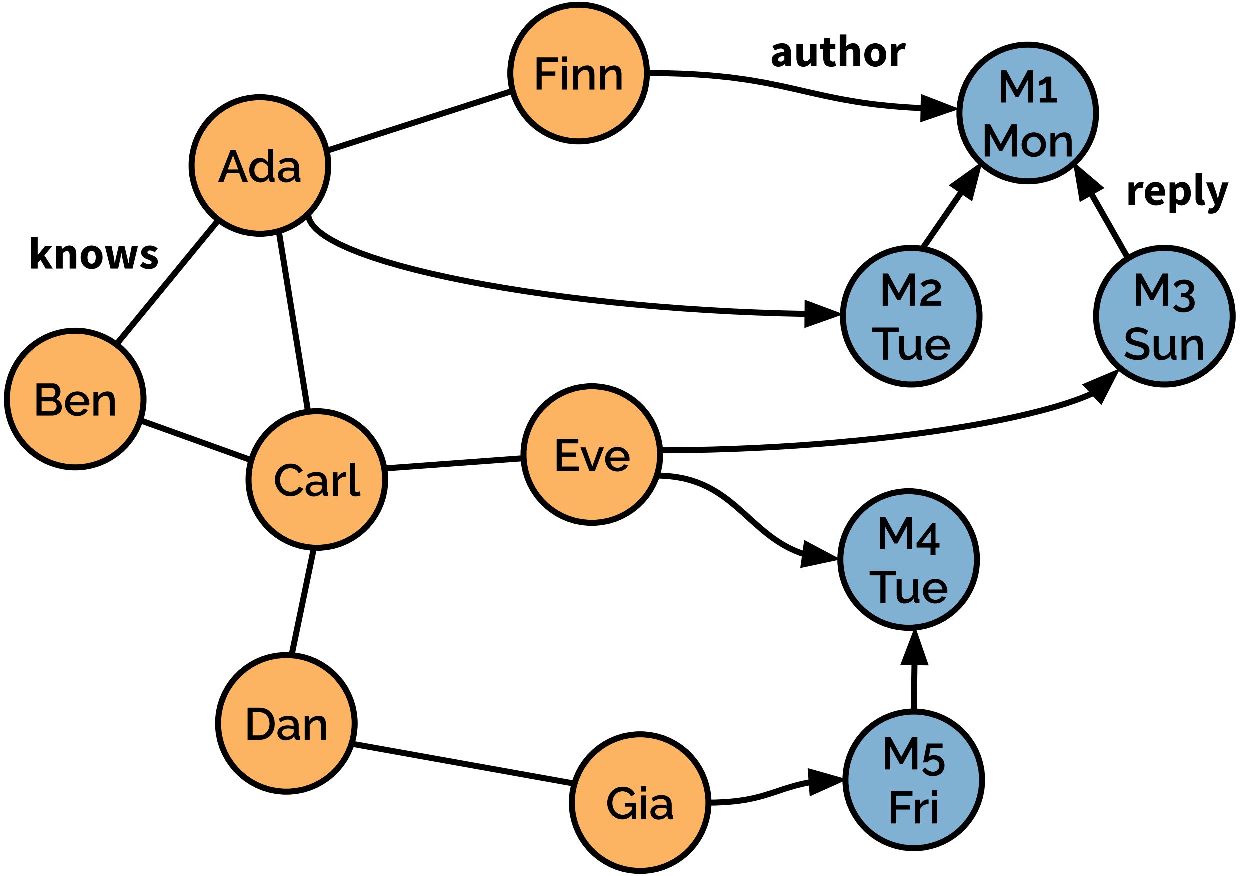
Queries

Updates

Data set | Queries | Updates



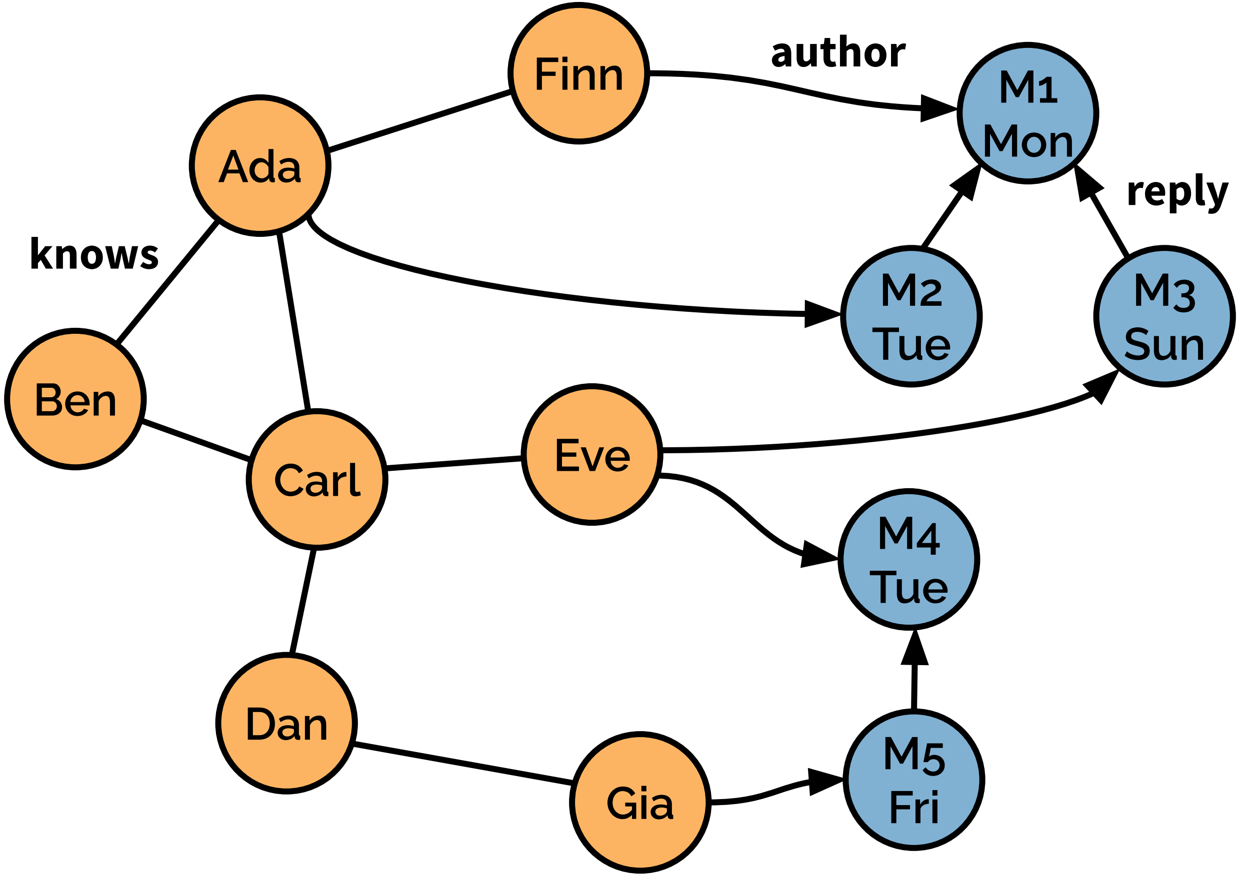
Data set | **Queries** | Updates

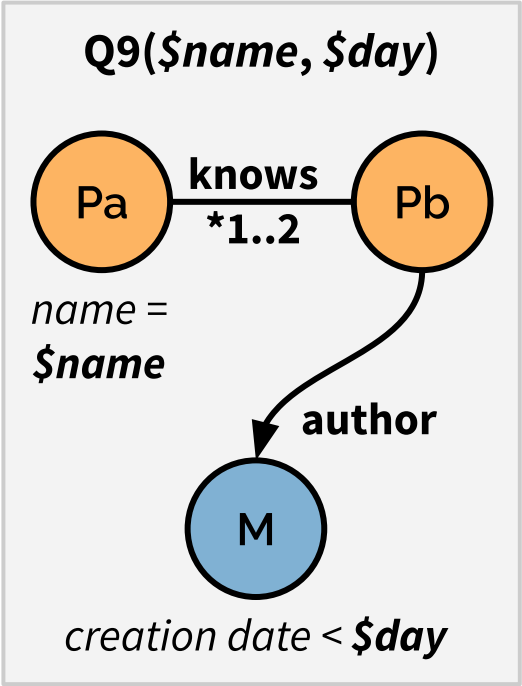


Data set

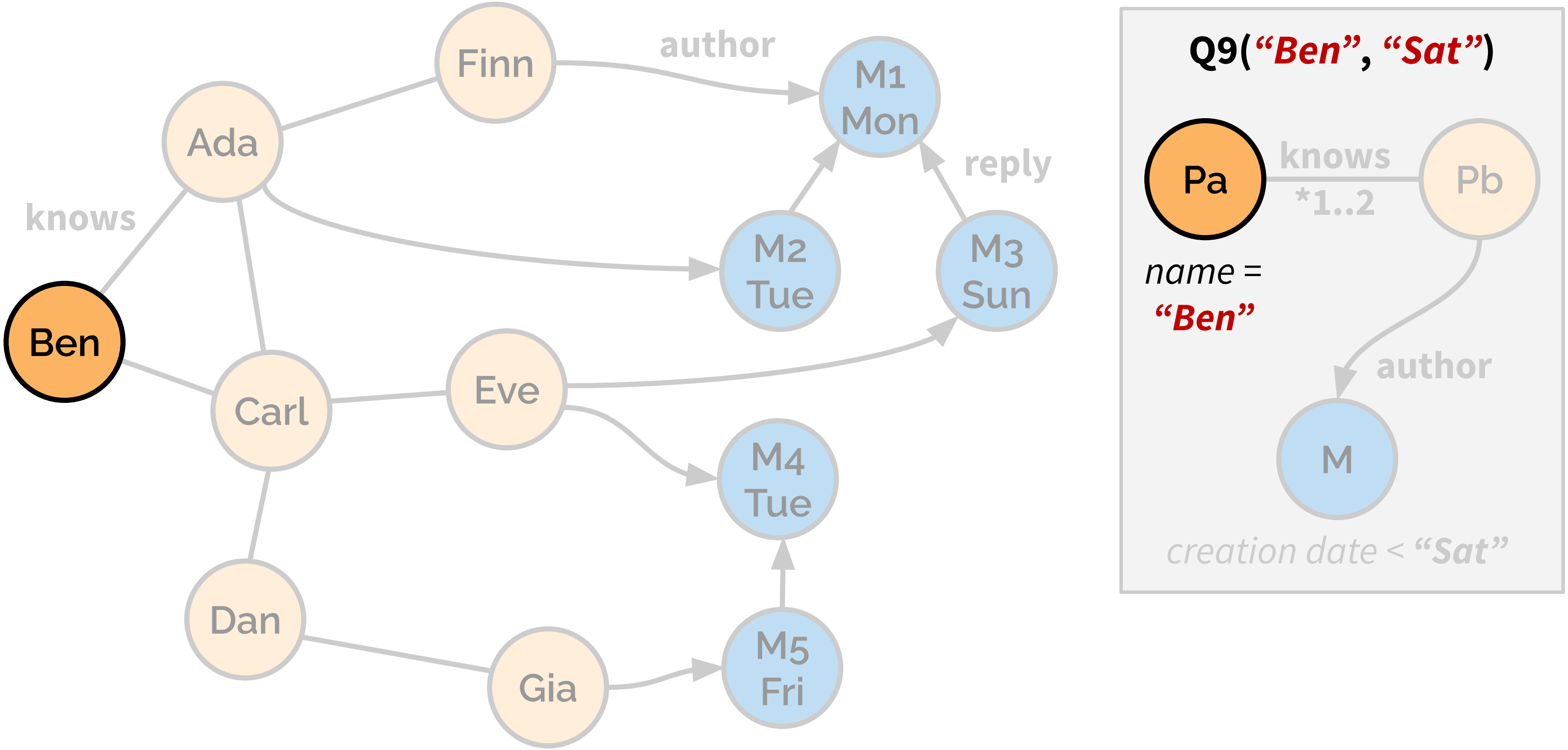
Queries

Updates

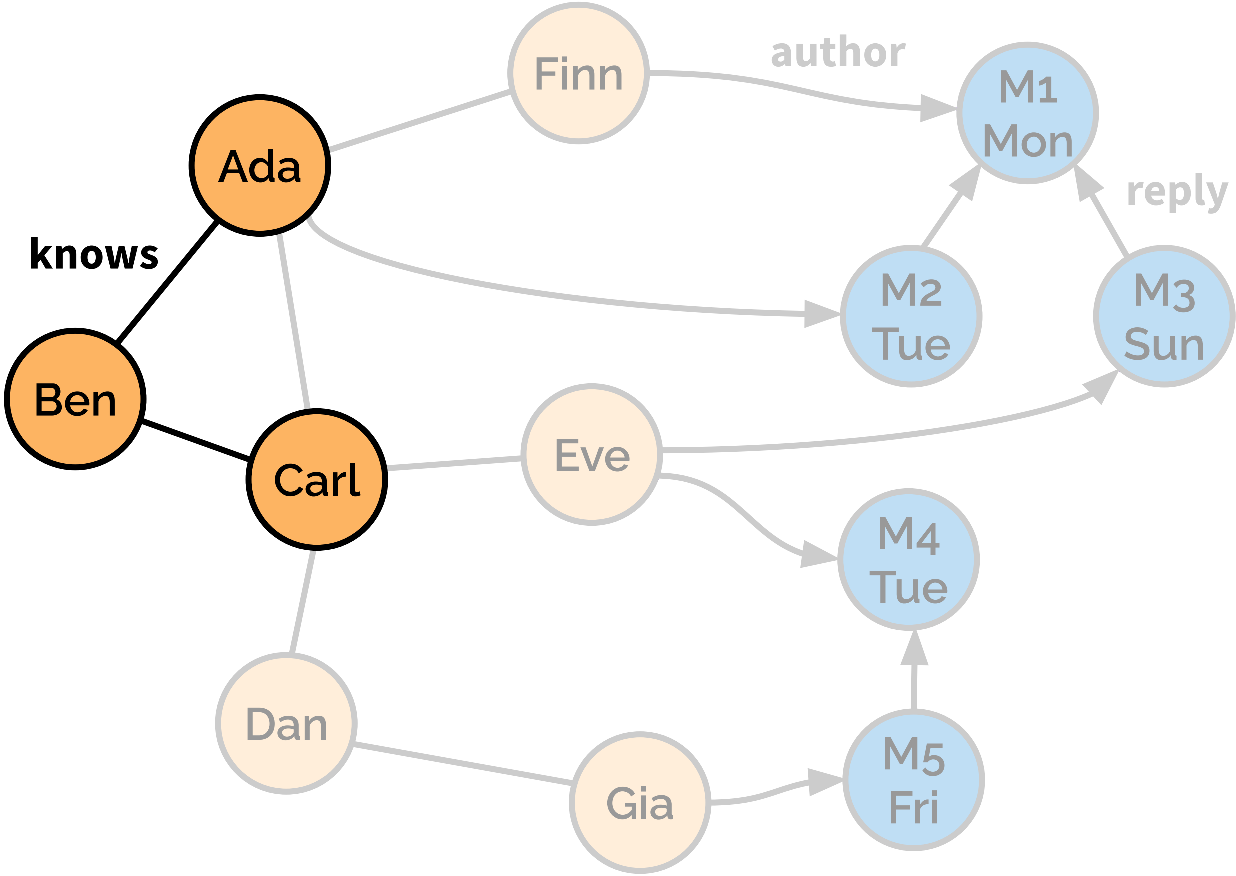


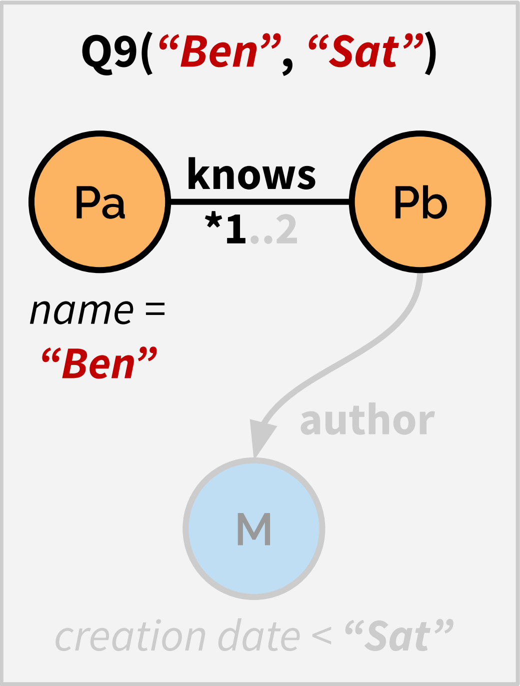


Data set | **Queries** | Updates

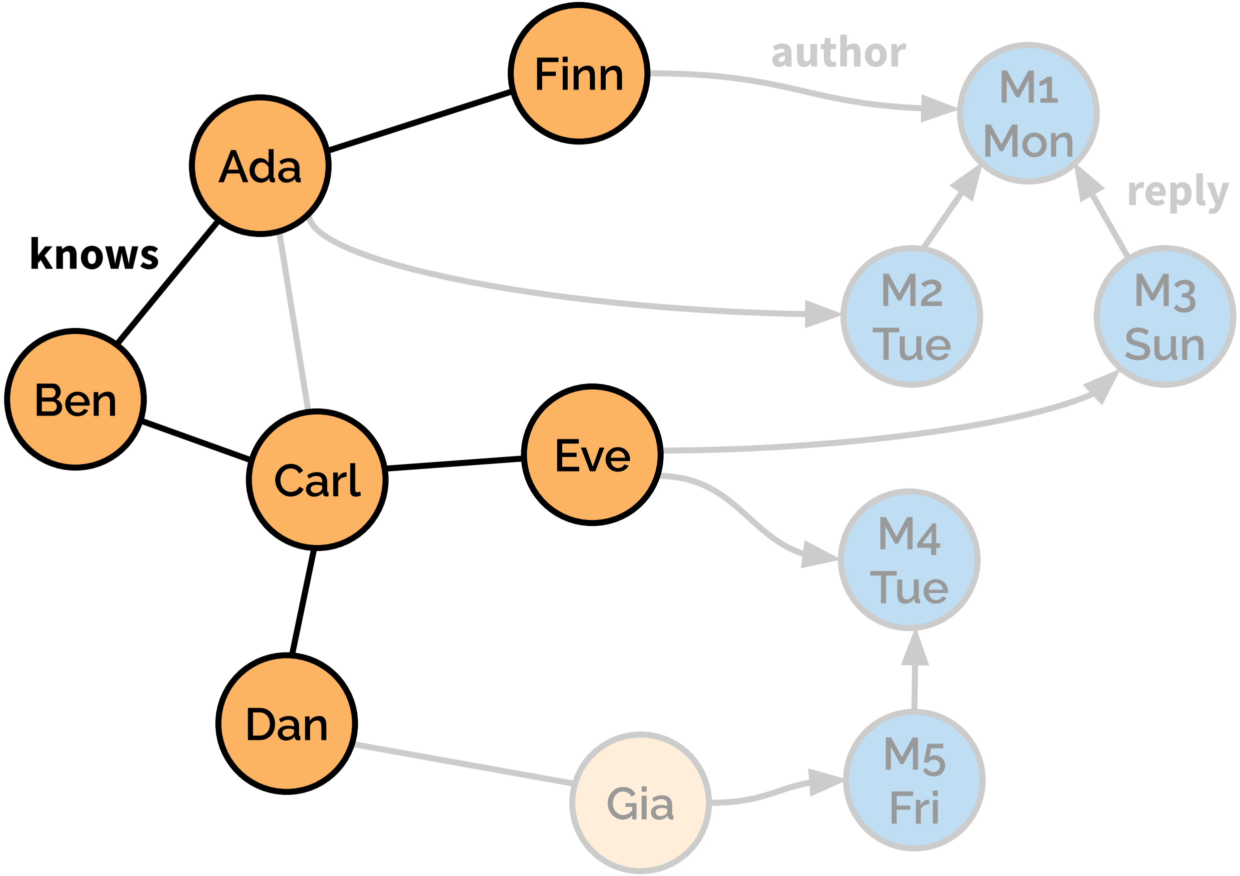


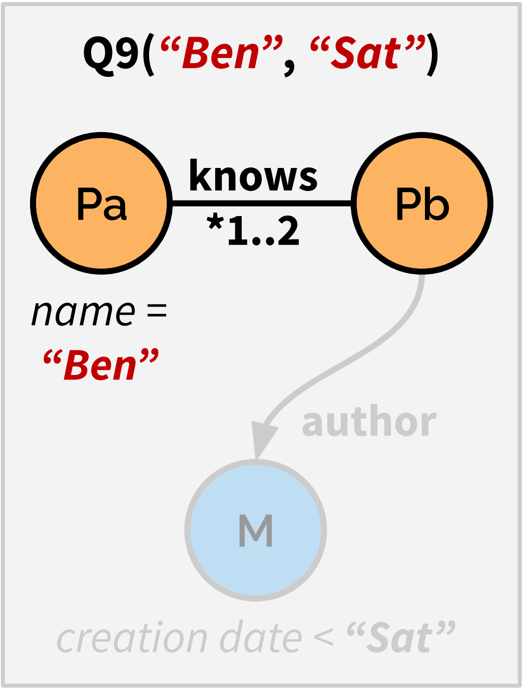
Data set | **Queries** | Updates



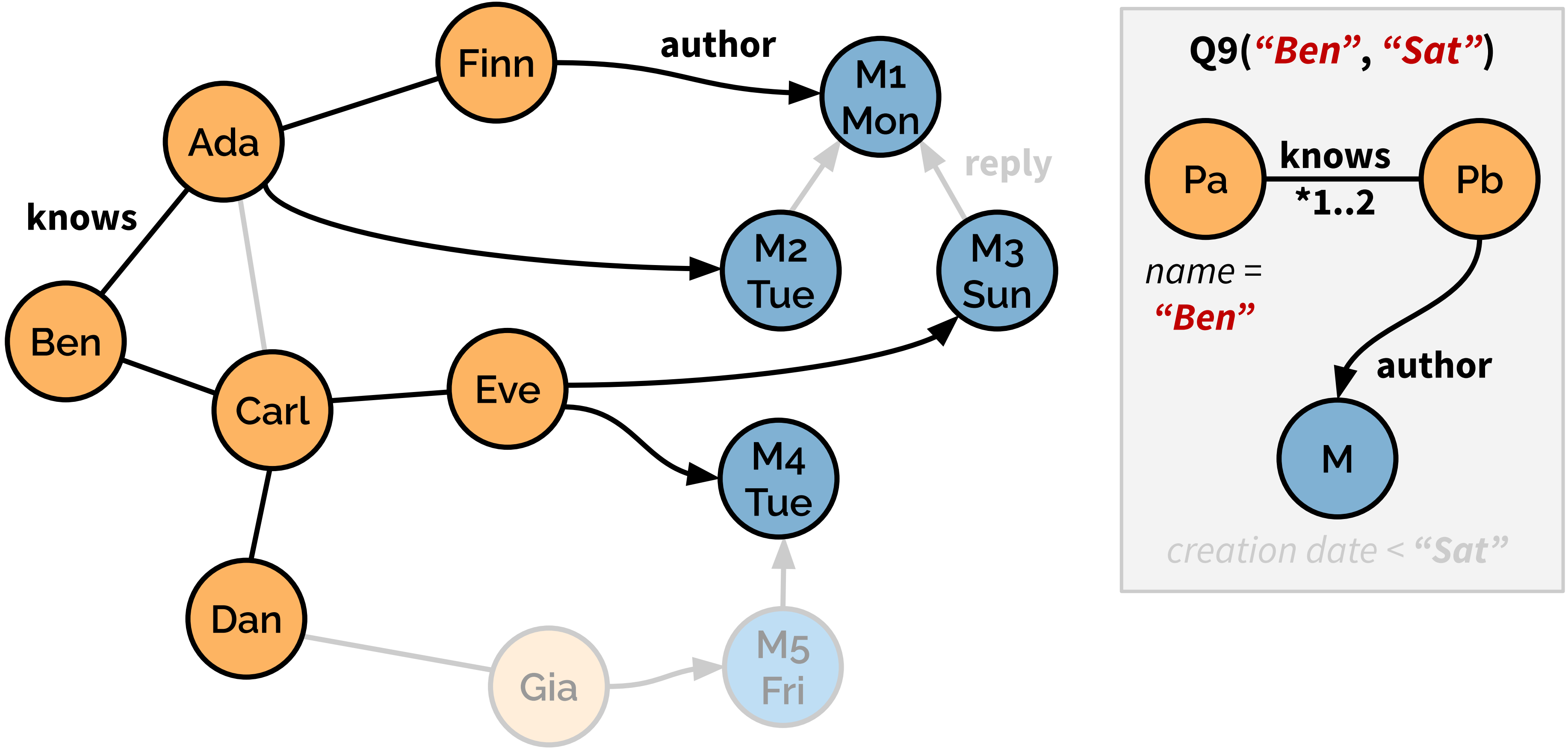


Data set | Queries | Updates

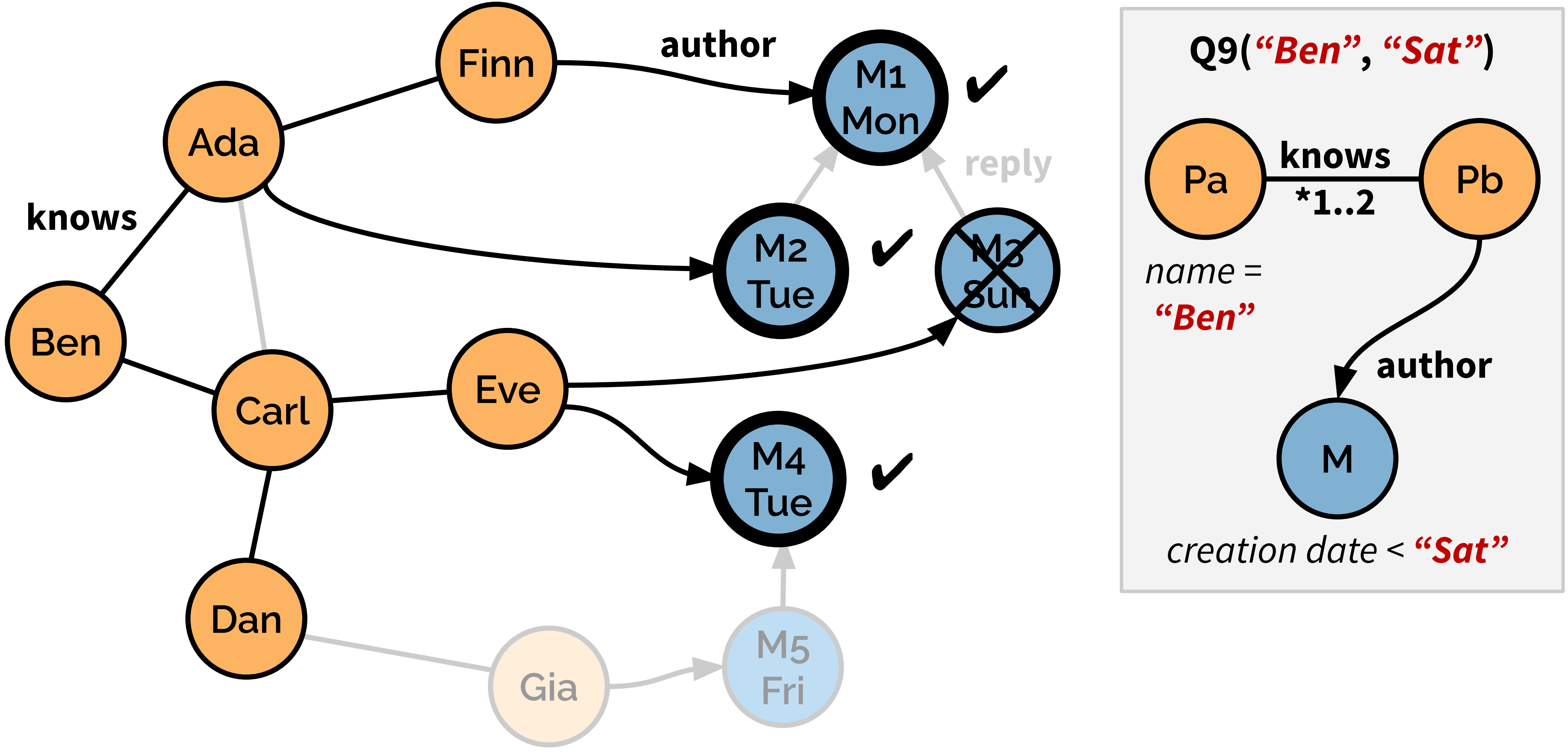




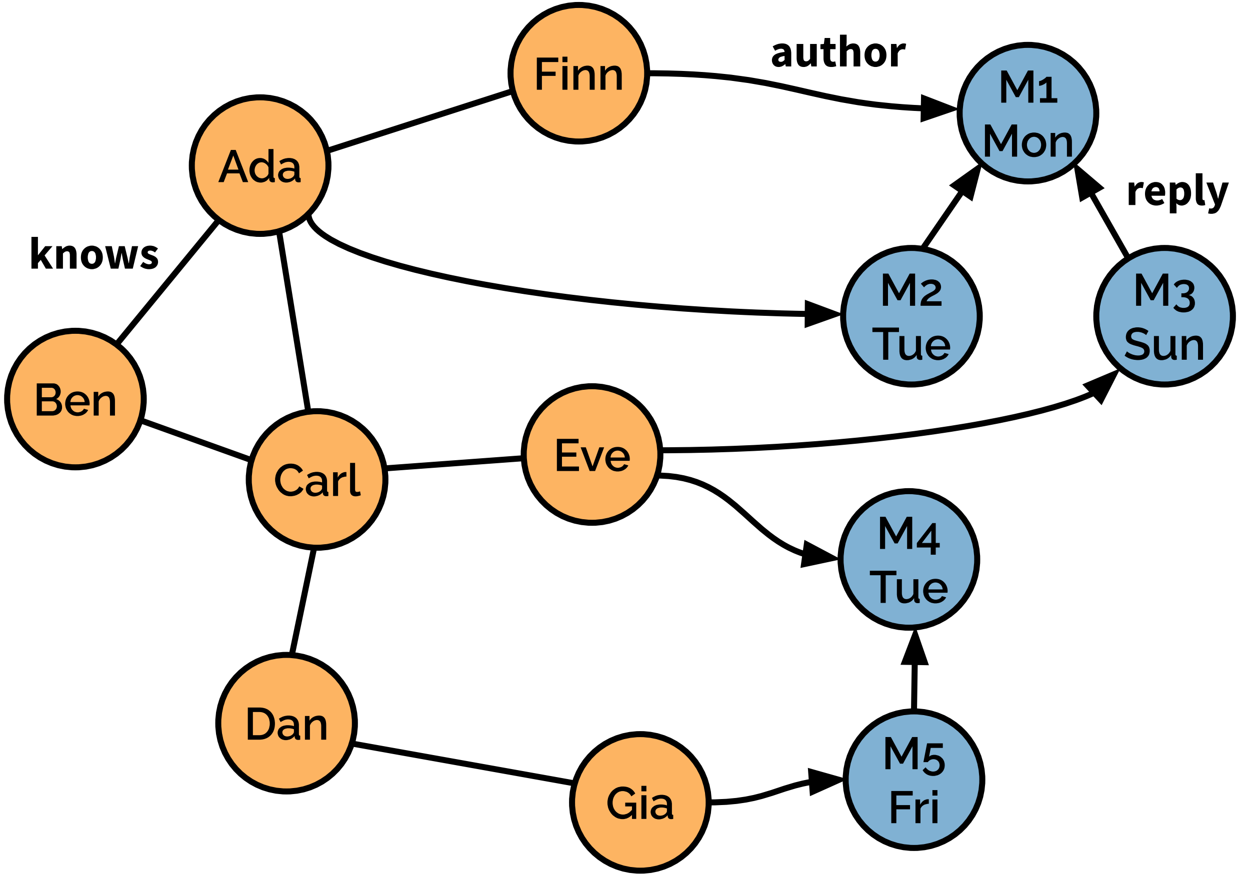
Data set | **Queries** | Updates

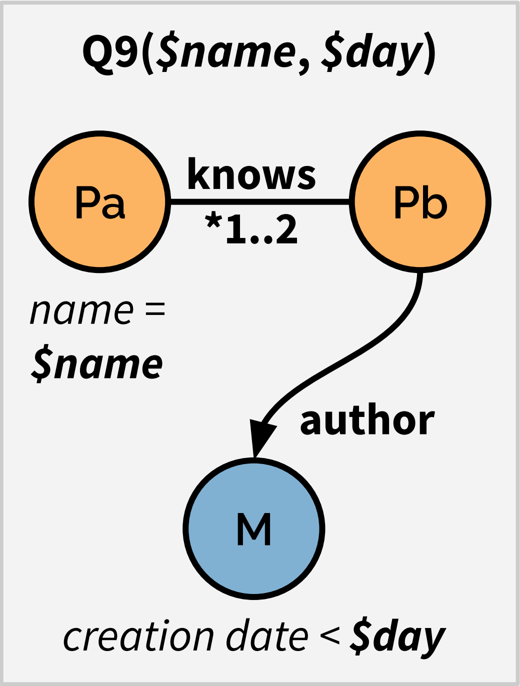


Data set | **Queries** | Updates

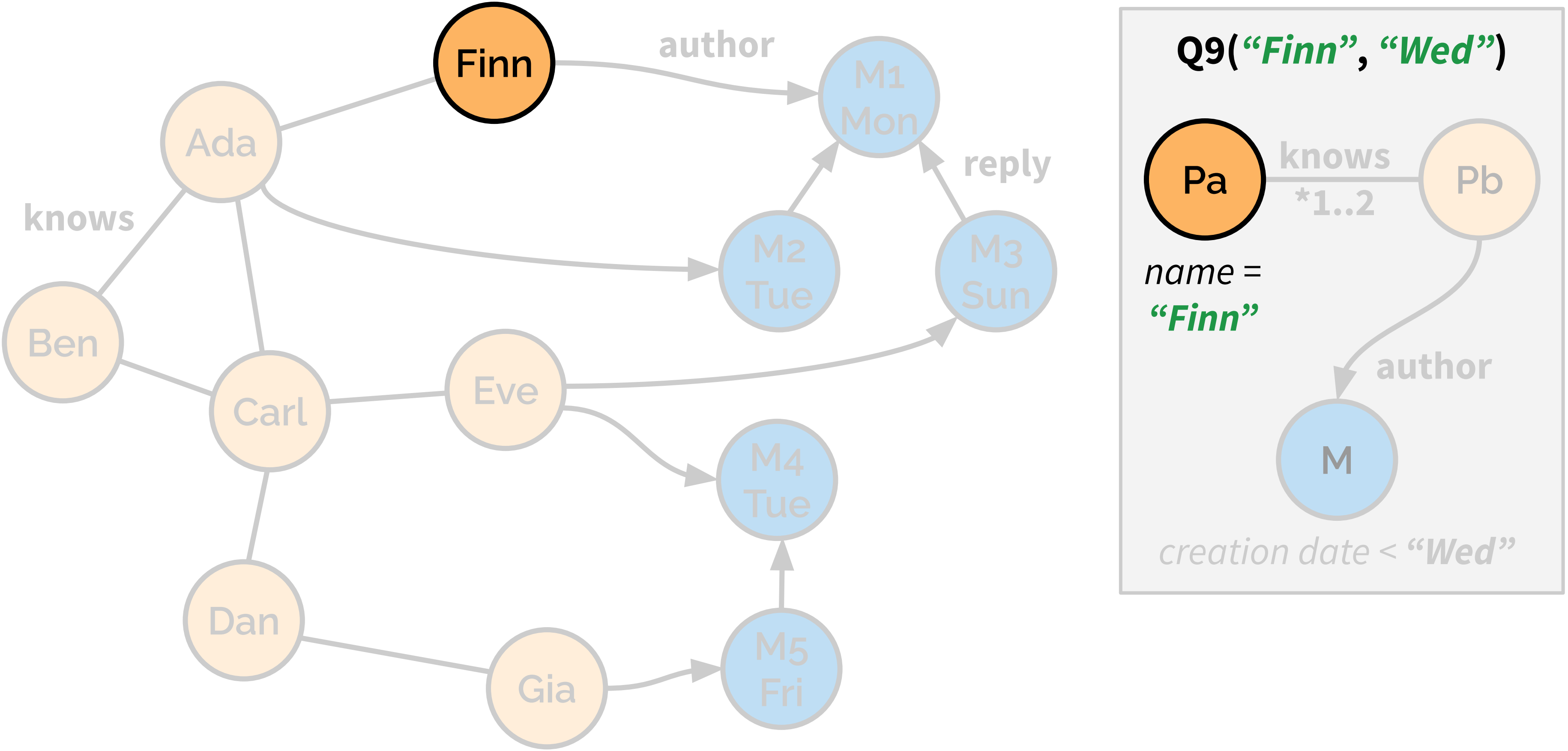


Data set Queries Updates

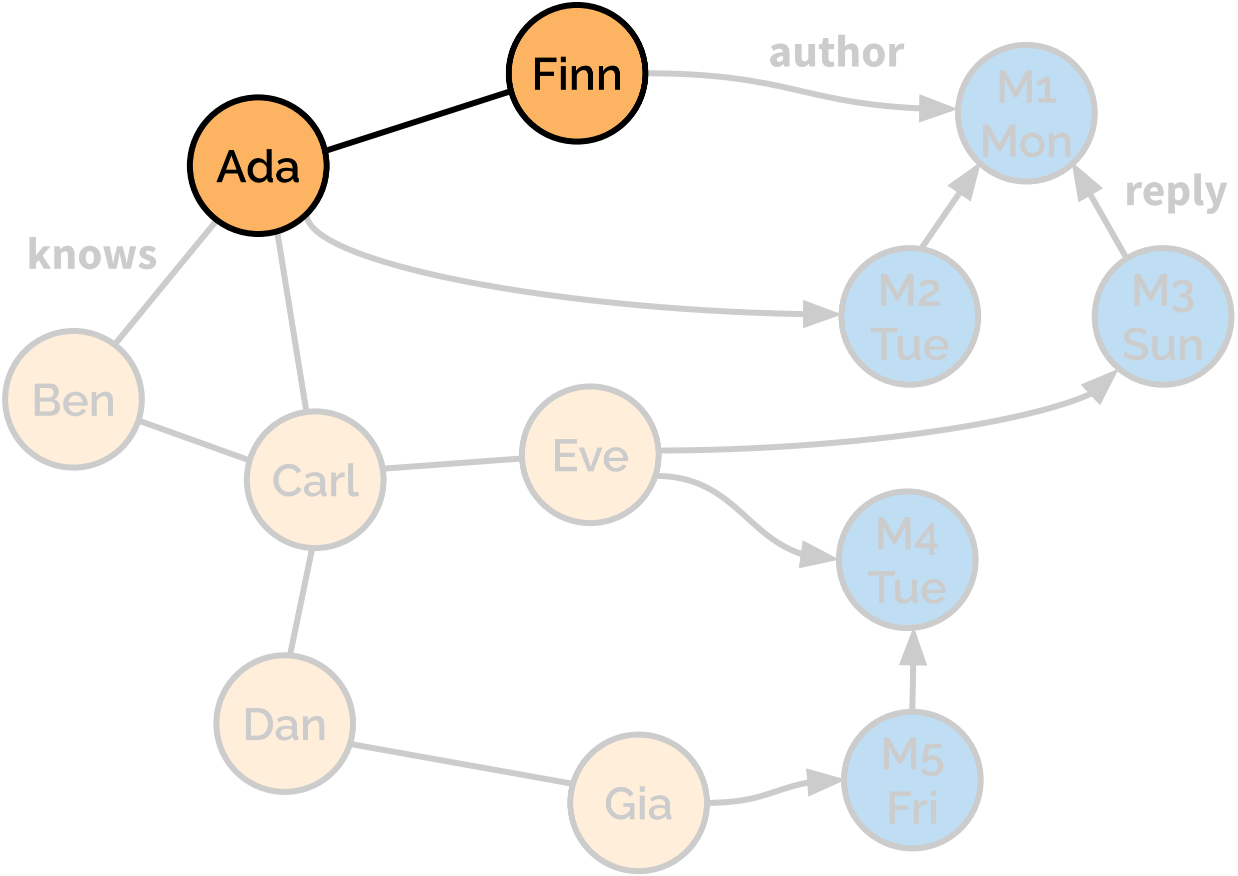


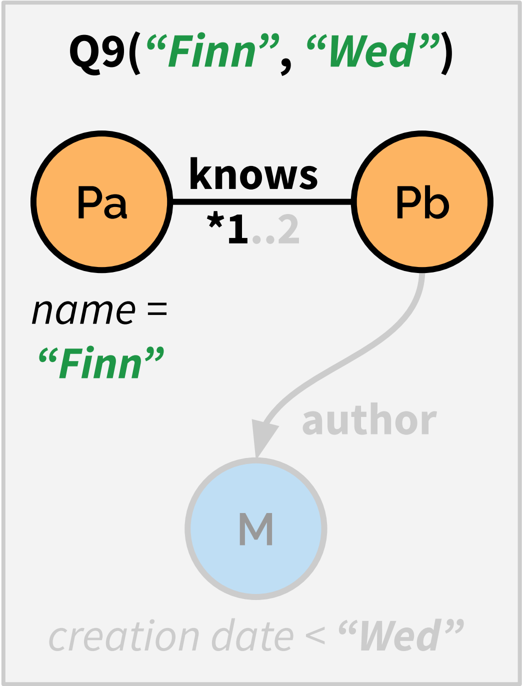


Data set | **Queries** | Updates

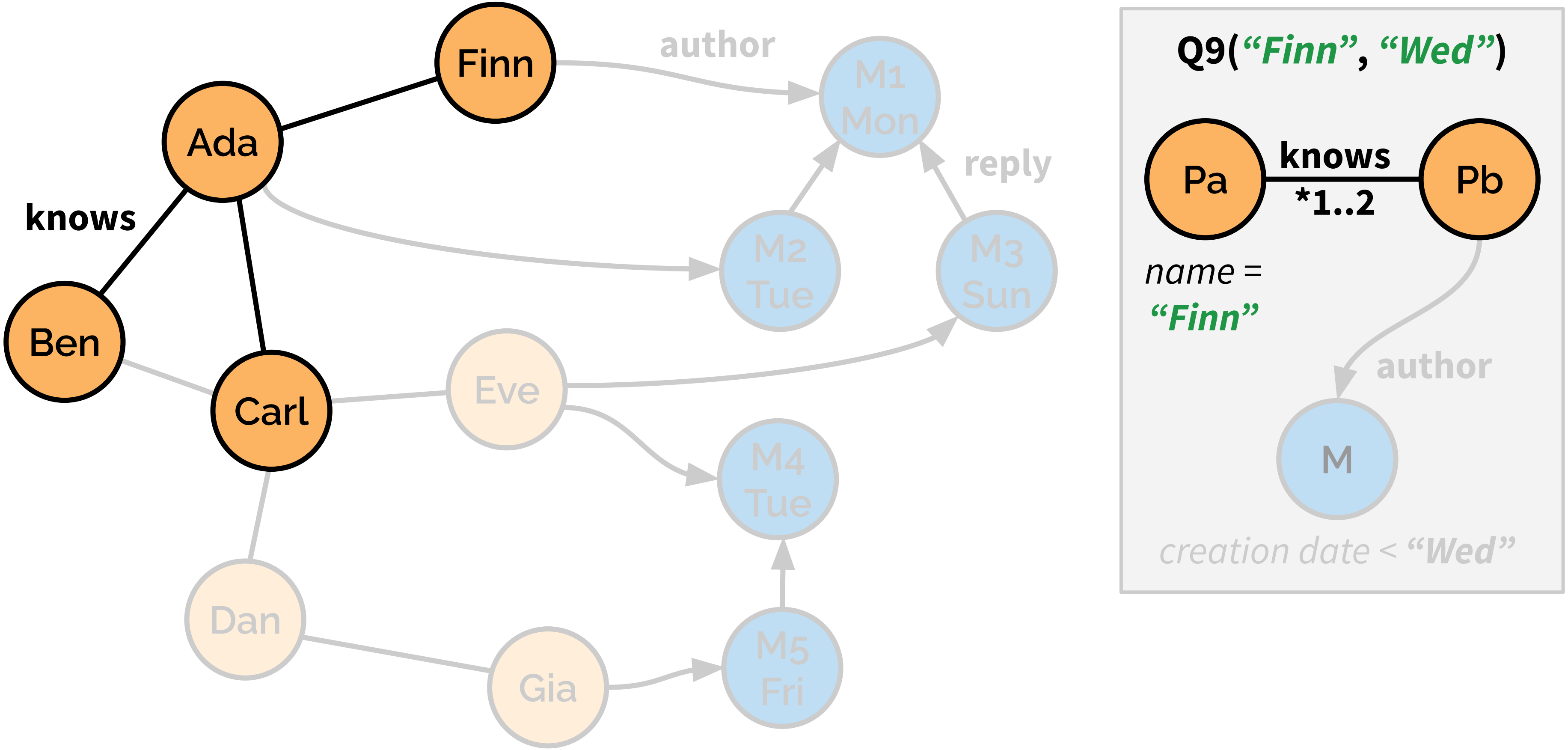


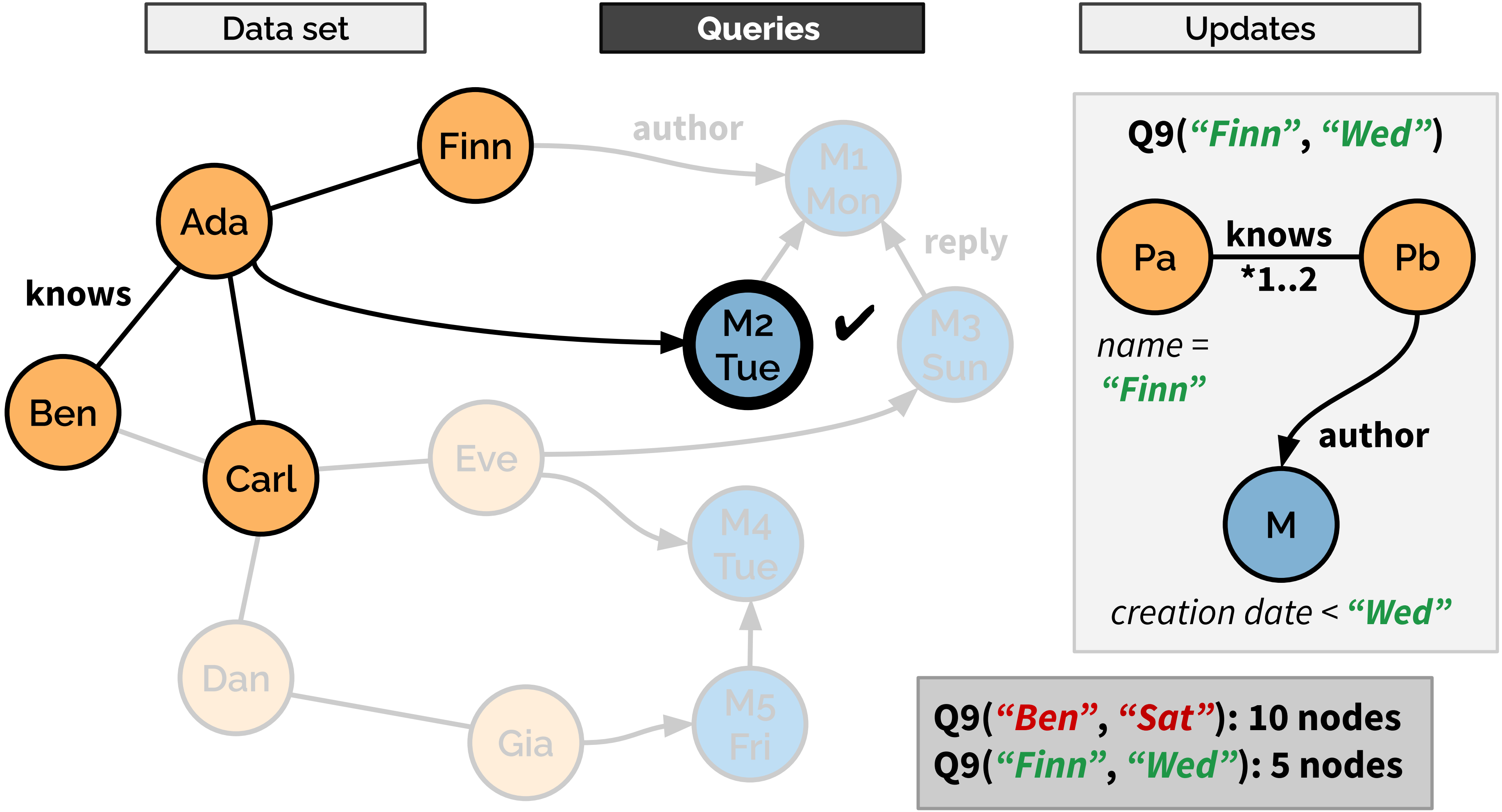
Data set | **Queries** | Updates





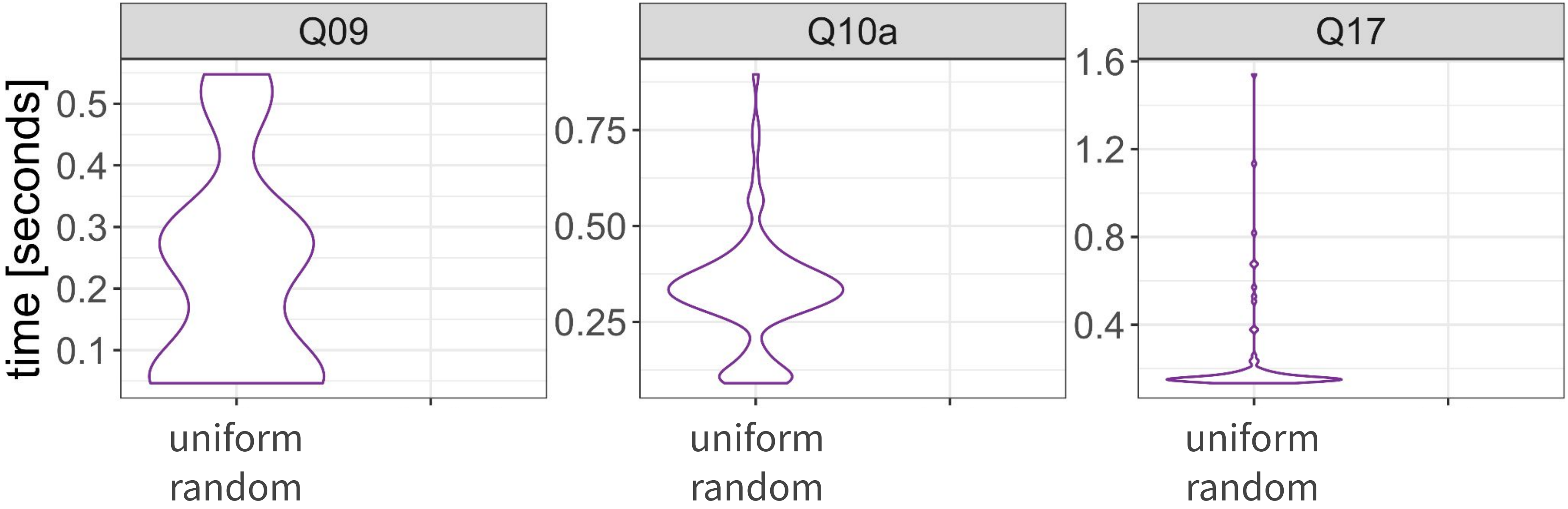
Data set | **Queries** | Updates



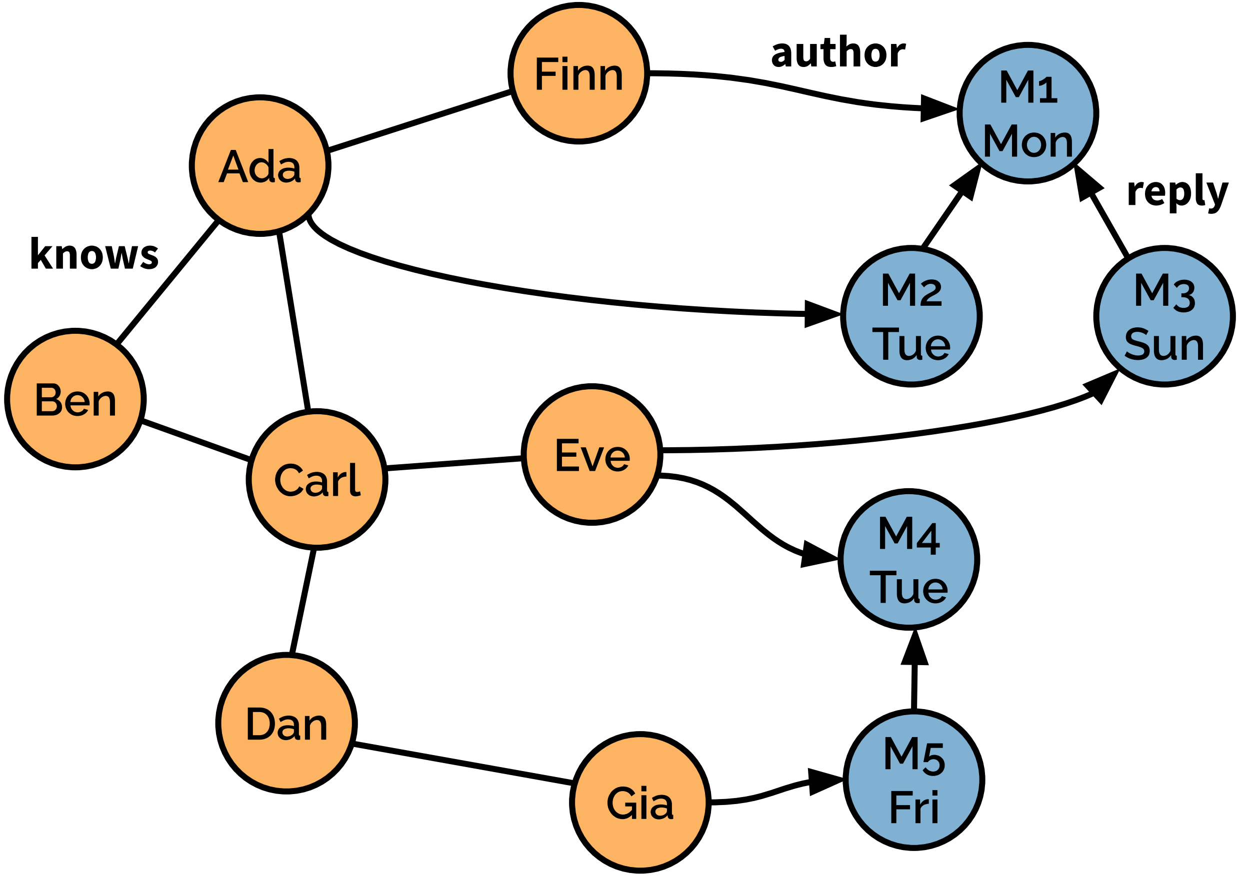

Data set
Queries
Updates
Finn
Ada
knows
Ben
Carl
Eve
Dan
Gia
author
M1 Mon
reply
M2 Tue
✔
M3 Sun
M4 Tue
M5 Fri
Q9("Finn", "Wed")
Pa
knows *1..2
Pb
name = "Finn"
author
M
creation date < "Wed"
Q9("Ben", "Sat"): 10 nodes
Q9("Finn", "Wed"): 5 nodes

Parameter selection

- ***Uniform random parameters*** → unstable distributions



Data set | **Queries** | Updates



<u>Parameter curation using statistics ("factors")</u>

numFriendsOfFriends

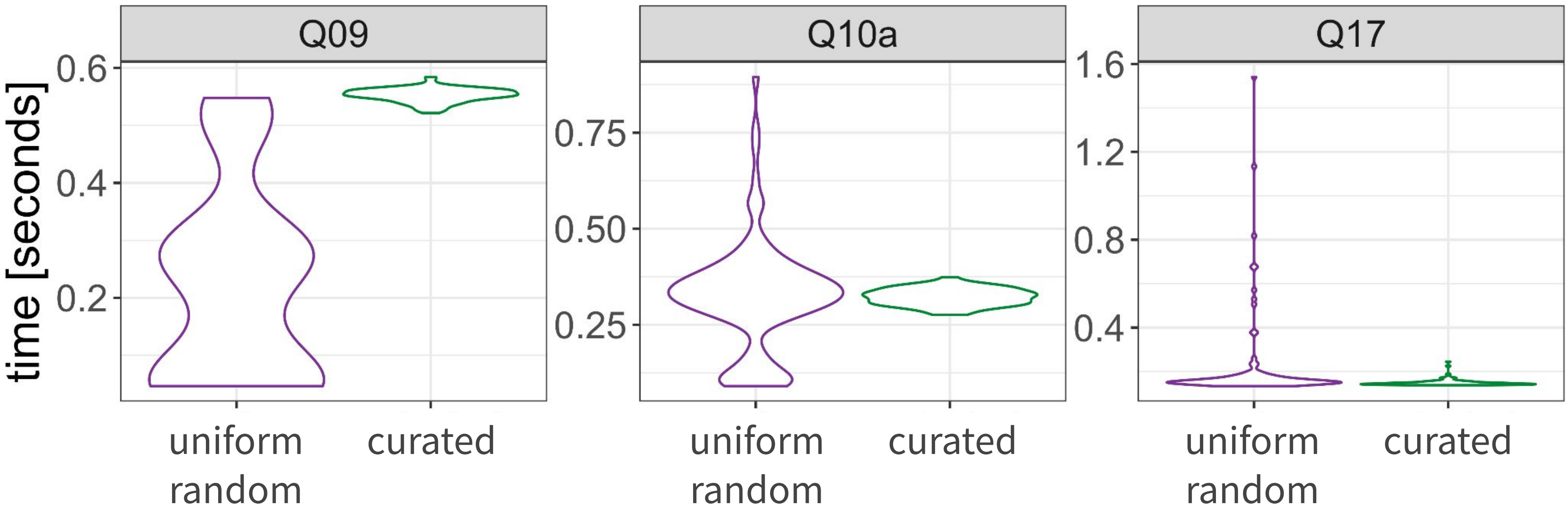
name	#1-hop	#2-hop
Ben	2	3
Carl	4	2
Ada	3	2
…		

numMessagesPerDay

day	#
Mon	1
Tue	2
…	

Parameter selection

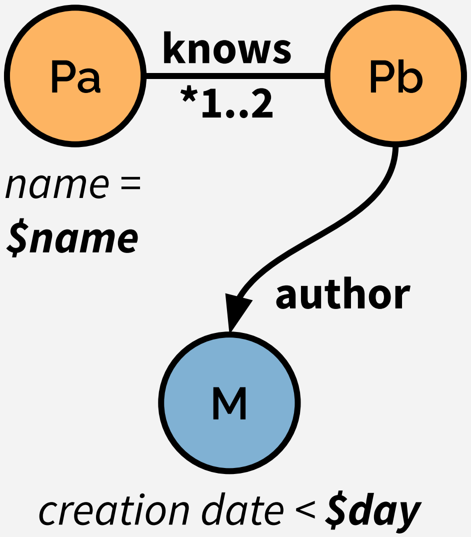
- ***Uniform random parameters*** → unstable distributions
- ***Curated parameters*** → tighter distributions, closer to bell curves



SQL:1992

```
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 (SQL:2023)

```
SELECT id
FROM GRAPH_TABLE (socialNetwork
 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

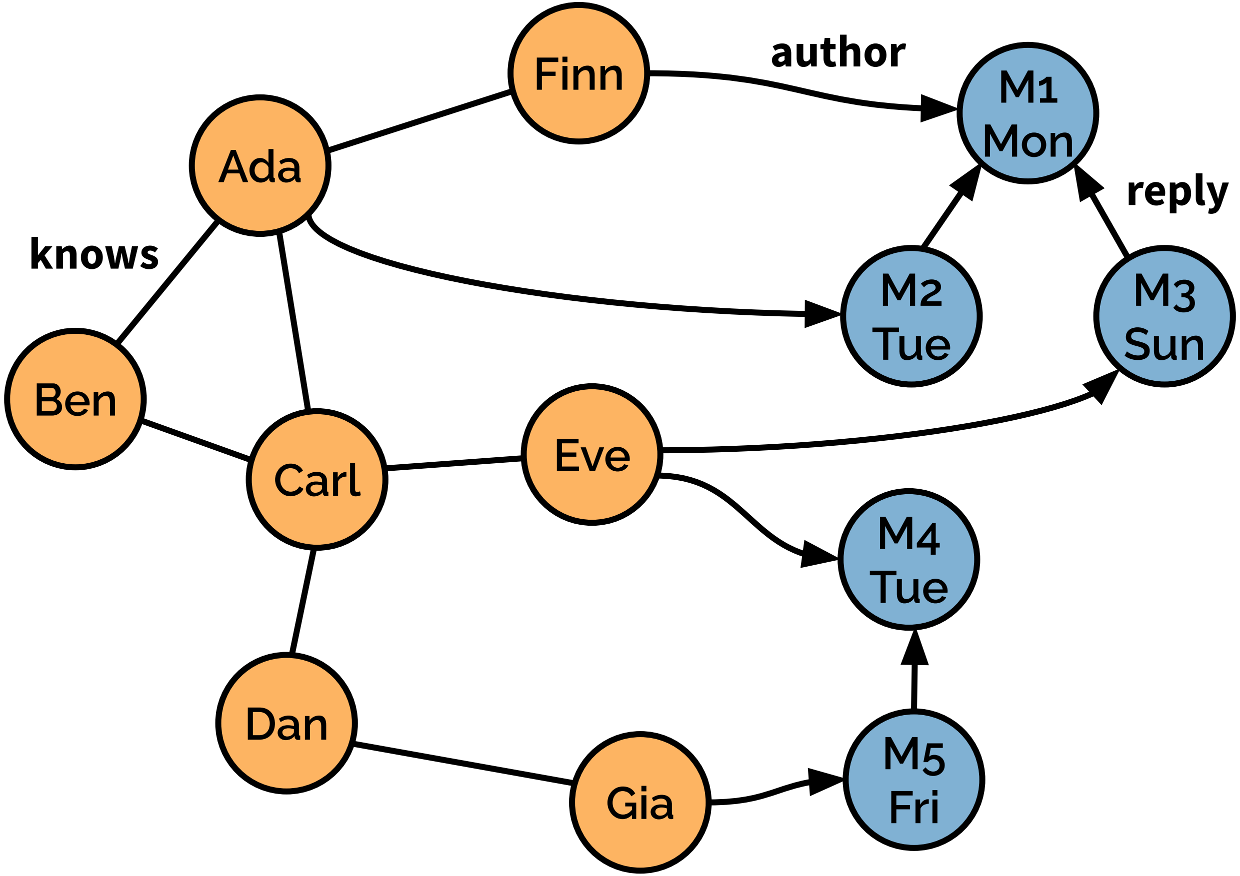
```
MATCH ANY ACYCLIC
  (Pa:Person WHERE Pa.name = $name)
  -[:knows]-{1,2} (Pb:Person)
  -[:author]-> (m:Message)
WHERE m.creationDate < $day
RETURN DISTINCT m.id
```

Updates

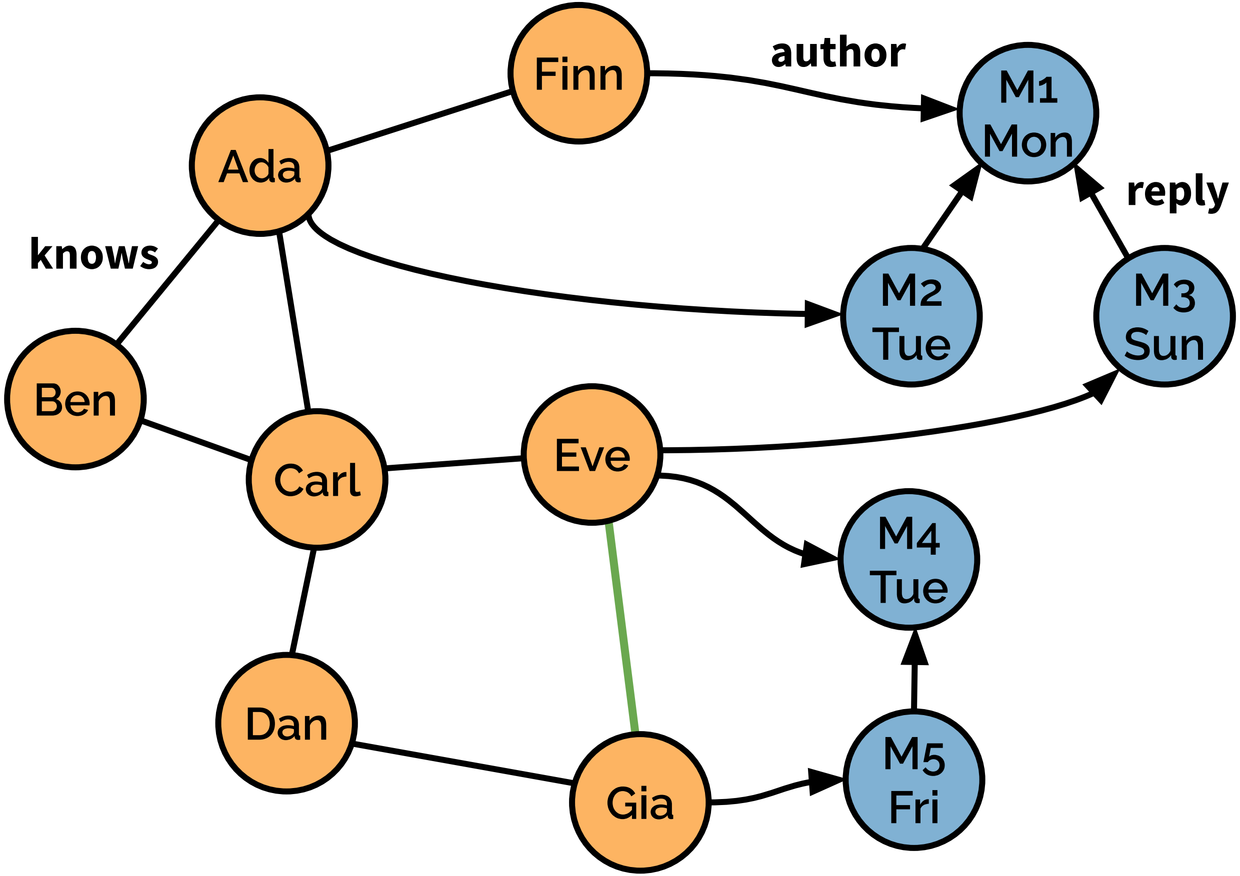
Data set

Queries

Updates



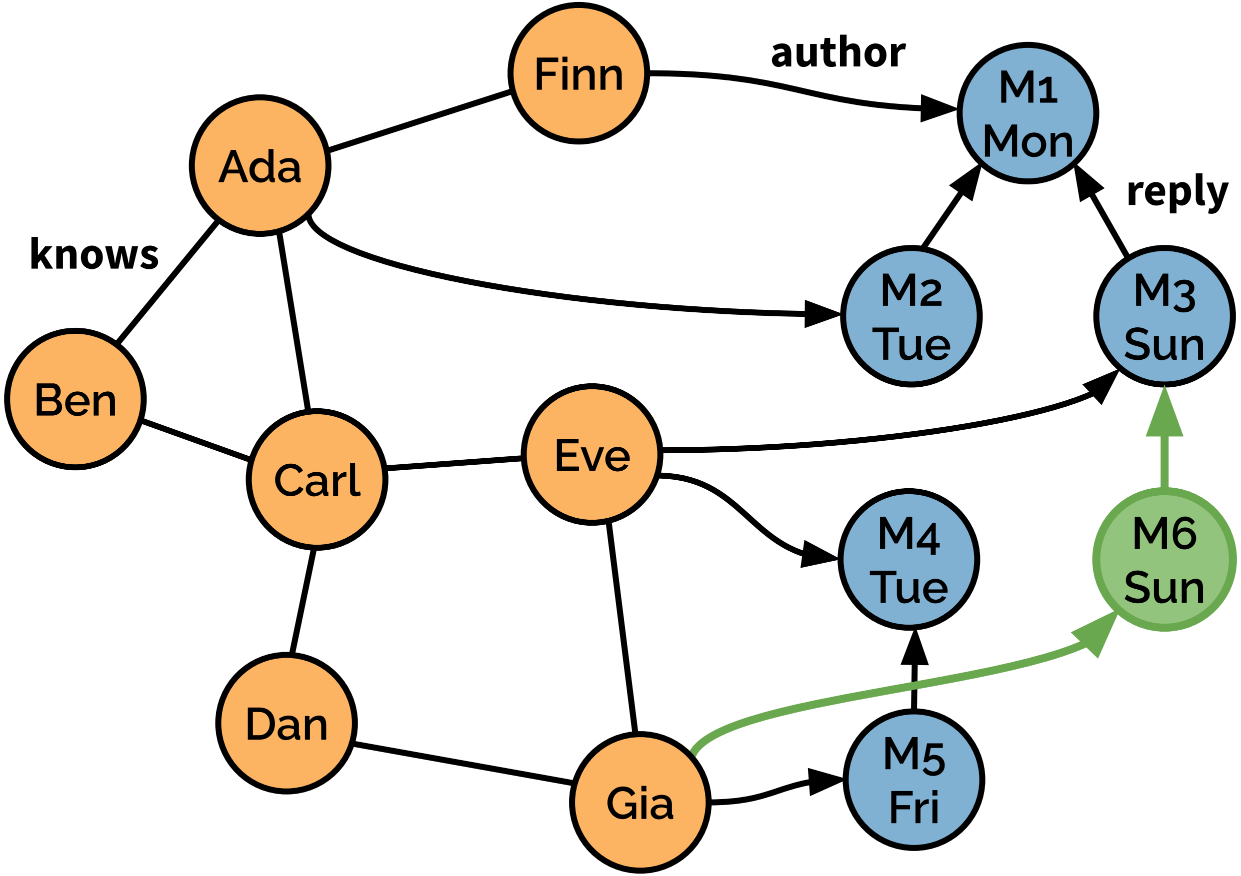
Data set | Queries | **Updates**



Updates

+ knows(“Eve”, “Gia”)

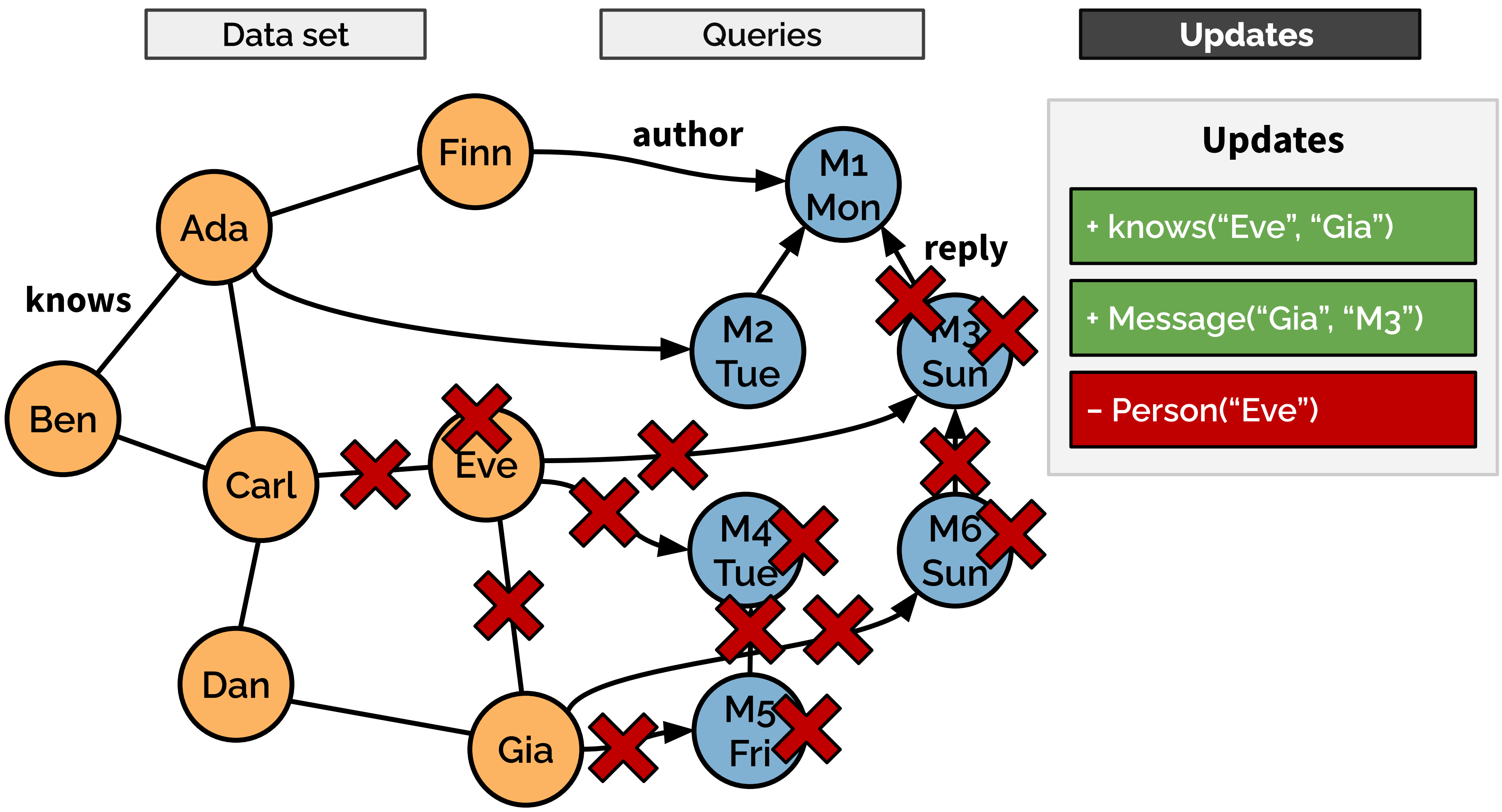
Data set | Queries | **Updates**



Updates

+ knows(“Eve”, “Gia”)

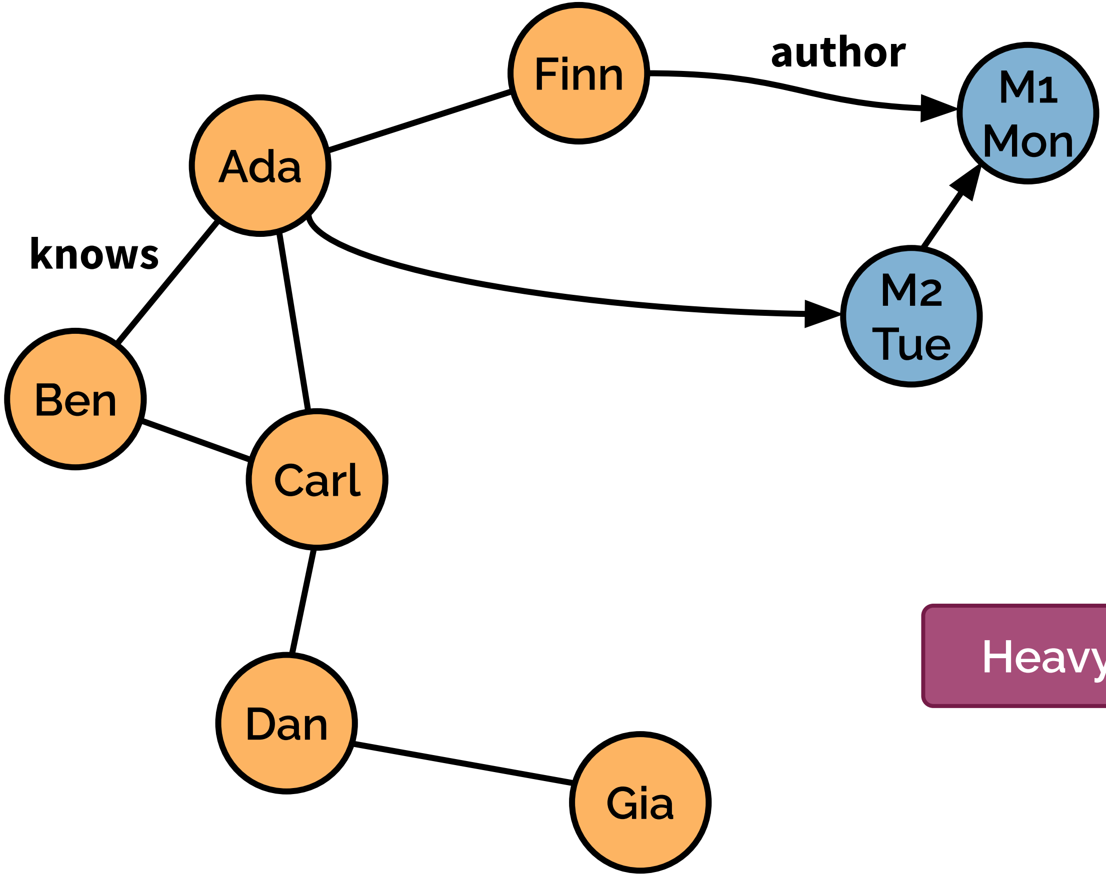
+ Message(“Gia”, “M3”)


Data set
Queries
Updates
Finn
author
M1
Mon
Ada
reply
knows
M2
Tue
M3
Sun
Ben
Eve
Carl
M4
Tue
M6
Sun
Dan
Gia
M5
Fri
Updates
+ knows("Eve", "Gia")
+ Message("Gia", "M3")
– Person("Eve")

Data set

Queries

Updates



Updates

+ knows(“Eve”, “Gia”)

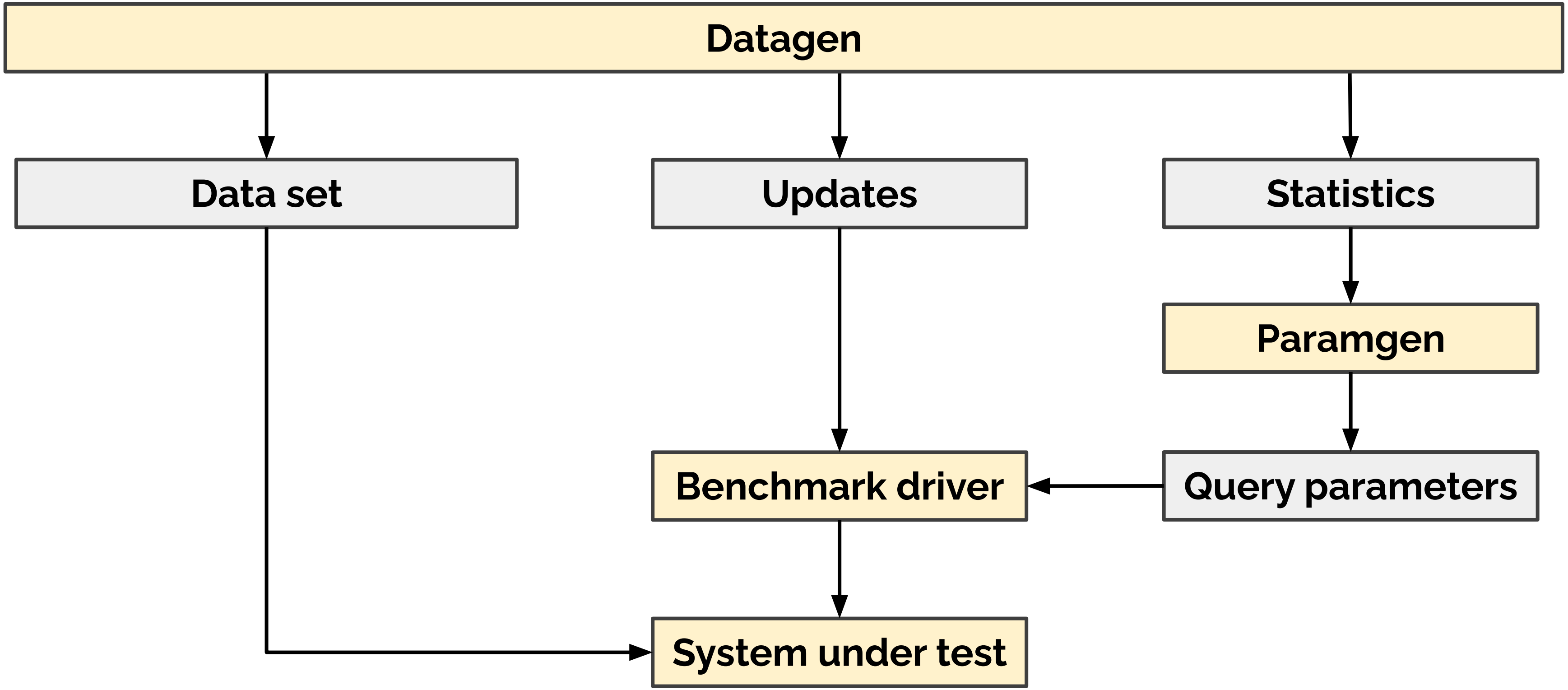
+ Message(“Gia”, “M3”)

– Person(“Eve”)

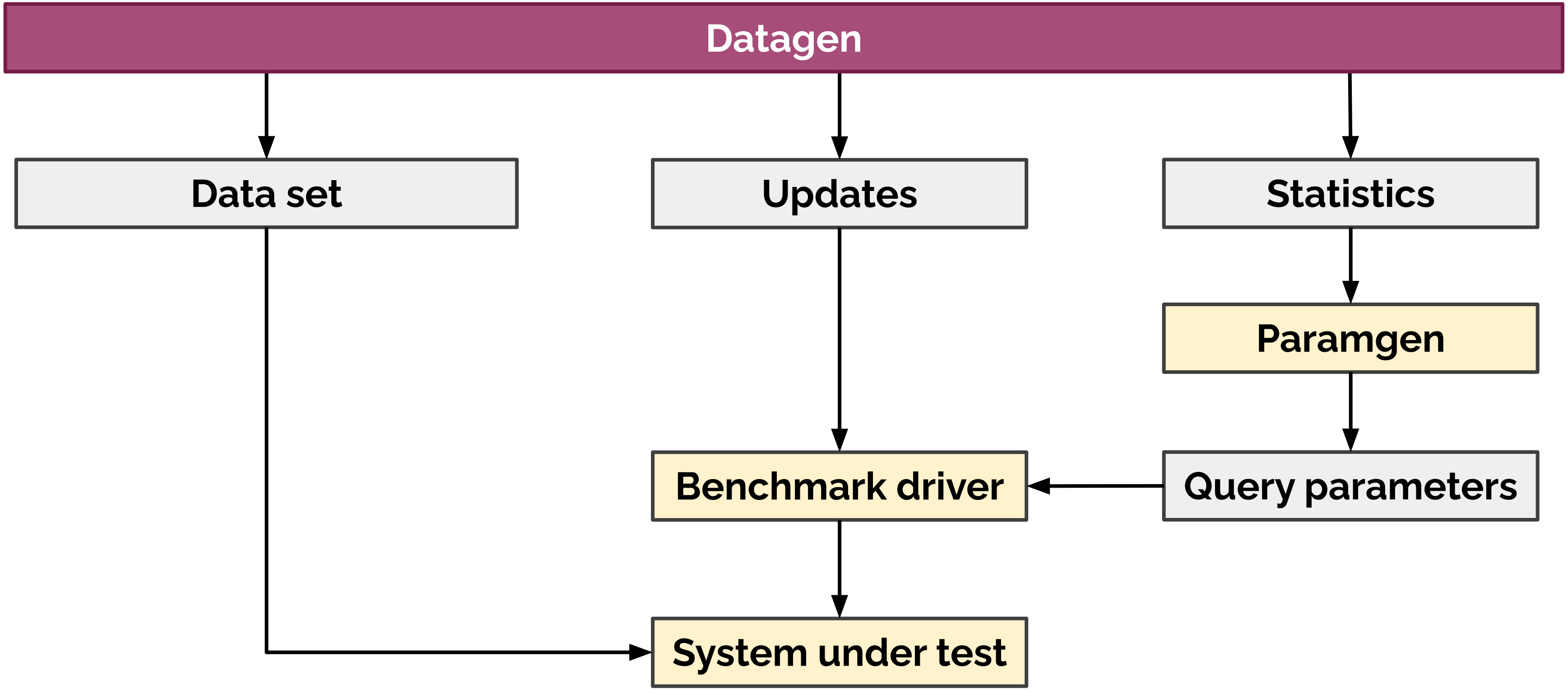
Heavy-hitting operation!

Benchmark framework

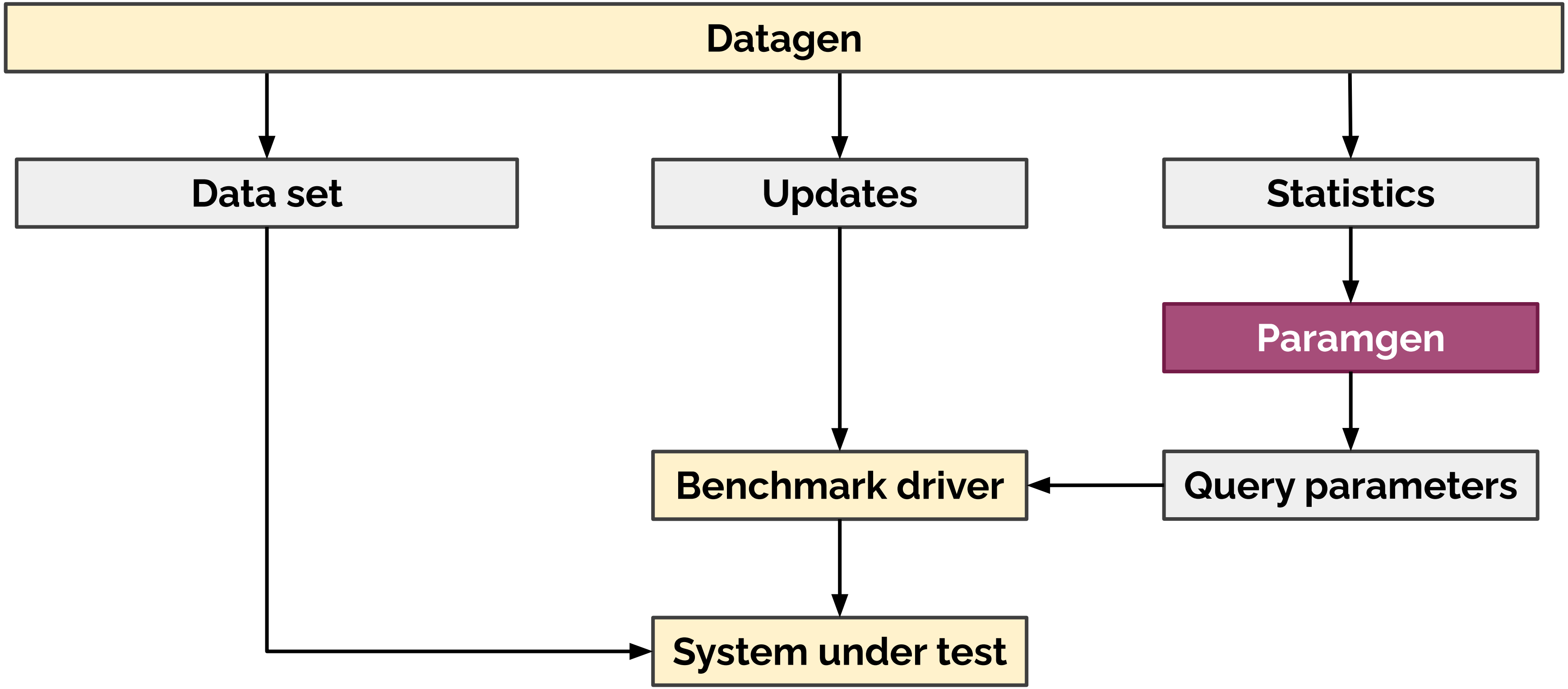
Benchmark workflow



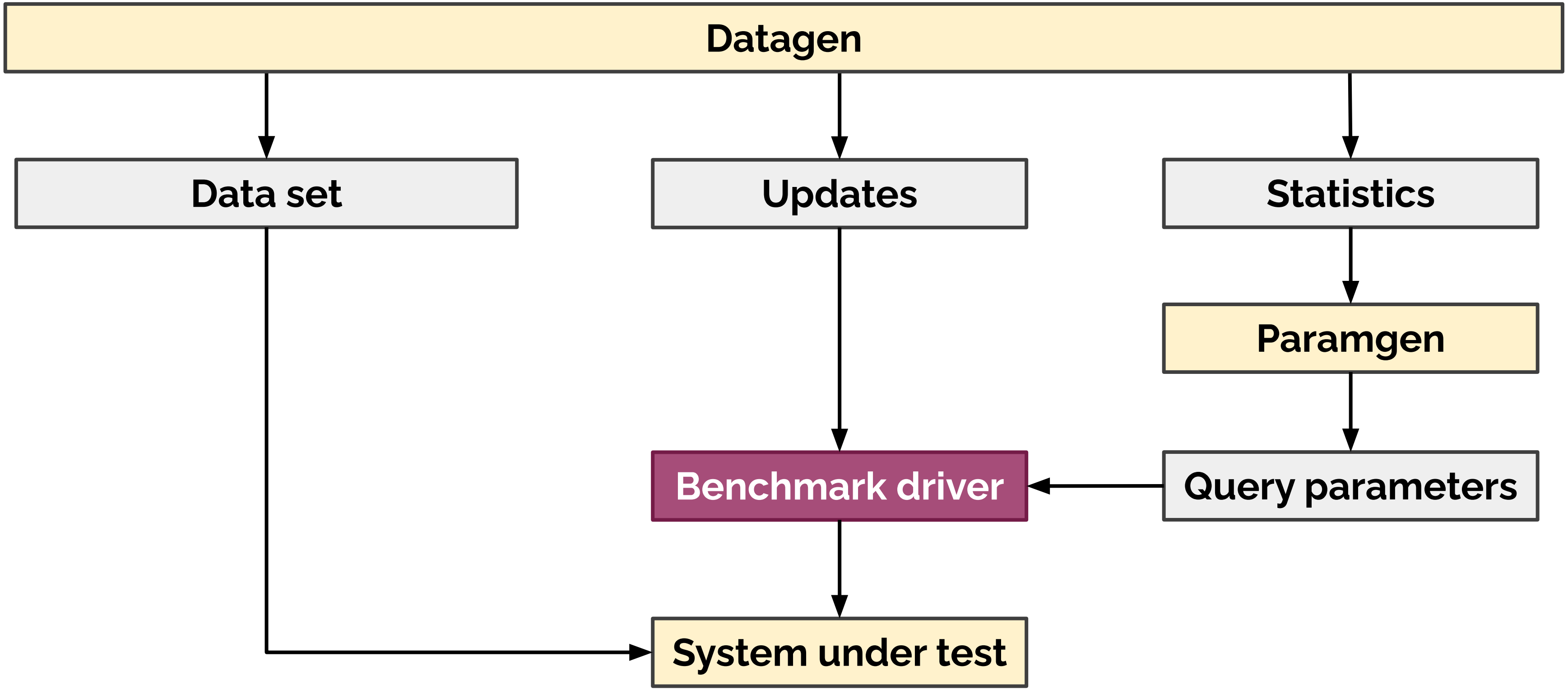
Benchmark workflow



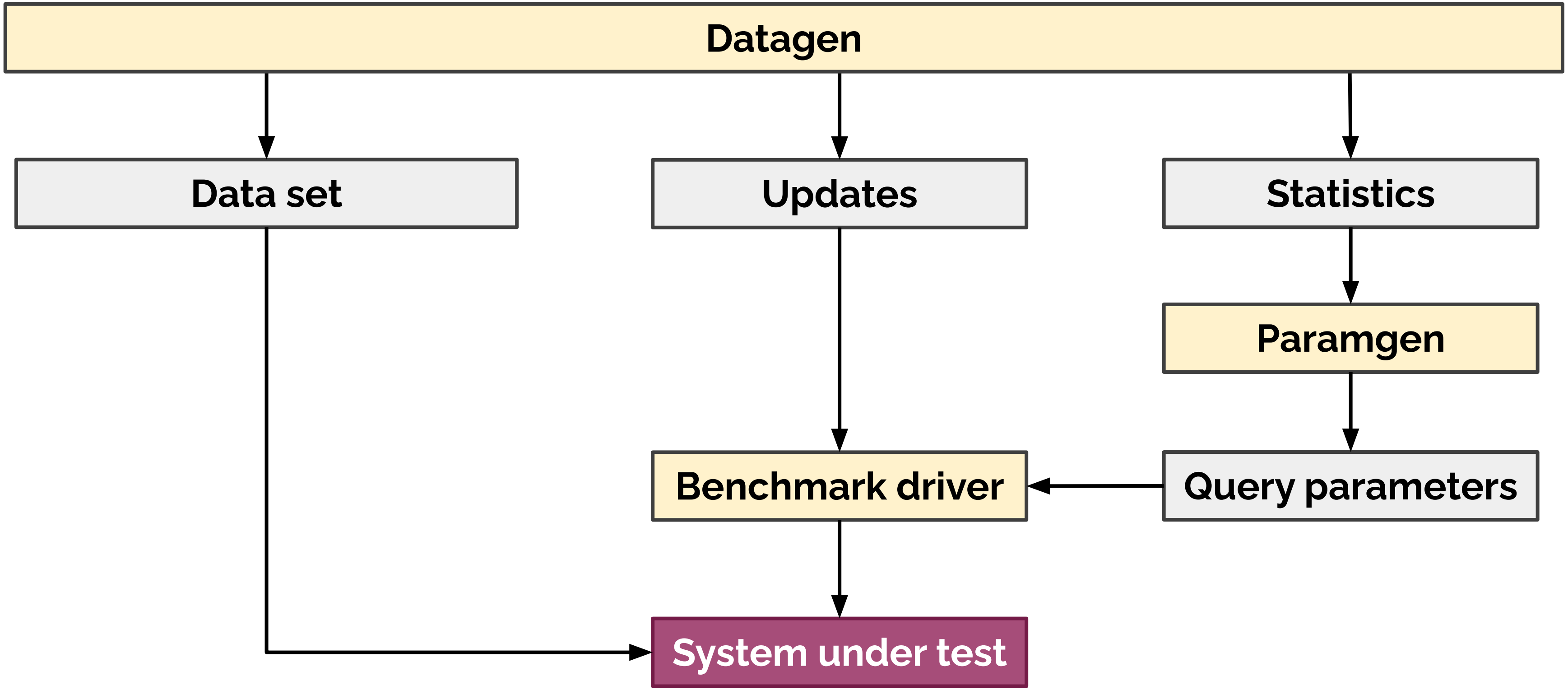
Benchmark workflow



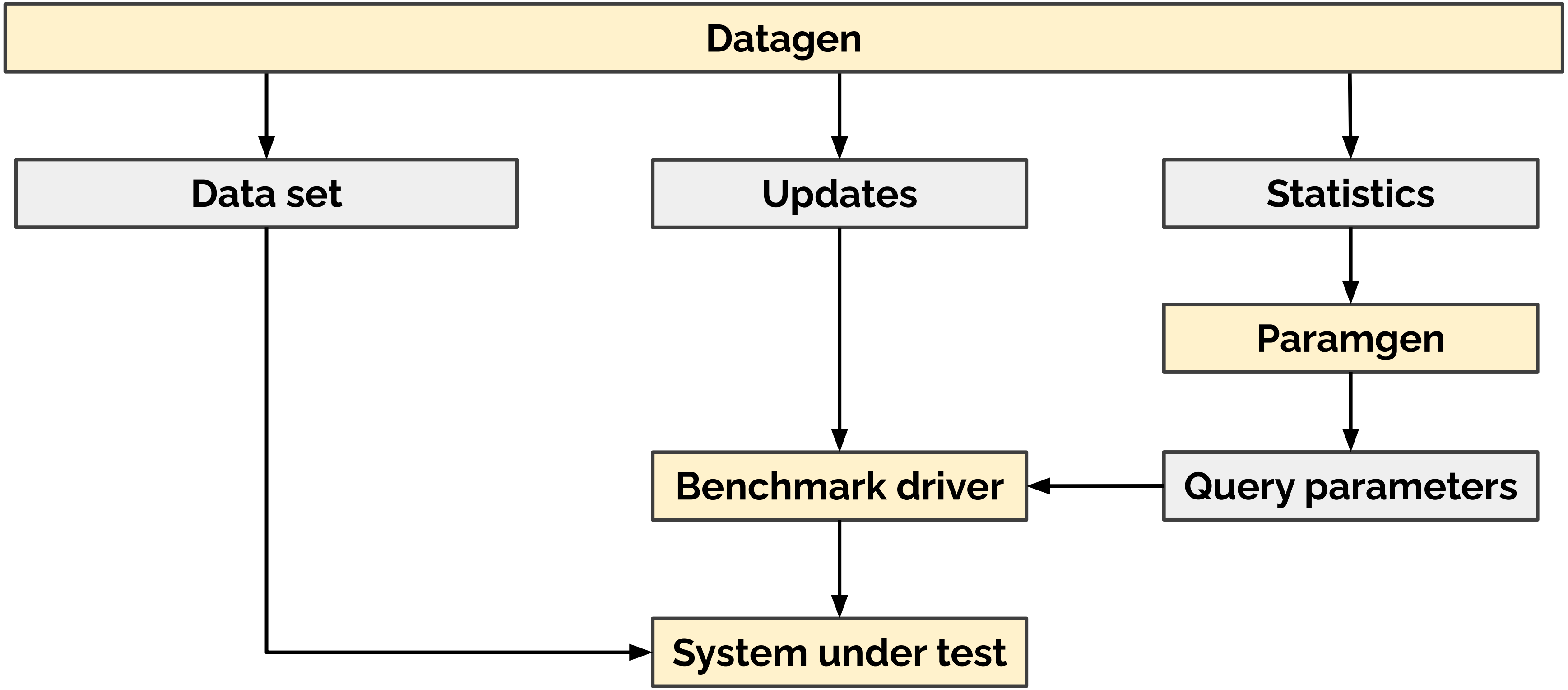
Benchmark workflow



Benchmark workflow



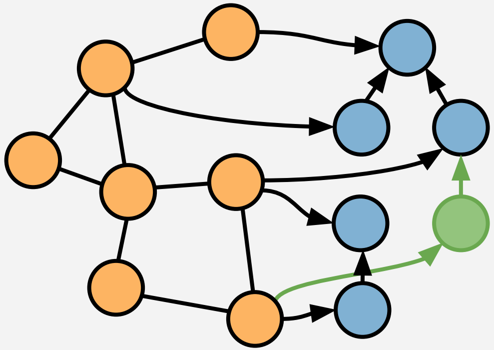
Benchmark workflow

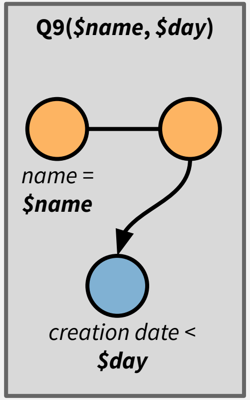


SNB workloads

- OLTP: Interactive
- OLAP: Business Intelligence

SNB Interactive v1 (2015)





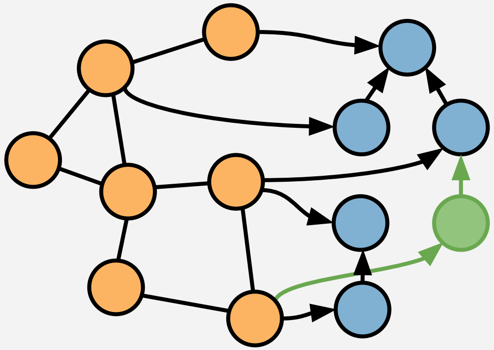
Queries start in 1–2 person nodes

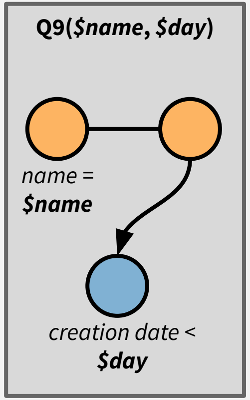
14 complex reads, 7 short reads

8 insert operations run concurrently

Goal: High throughput (ops/s)

SNB Interactive v1 (2015)



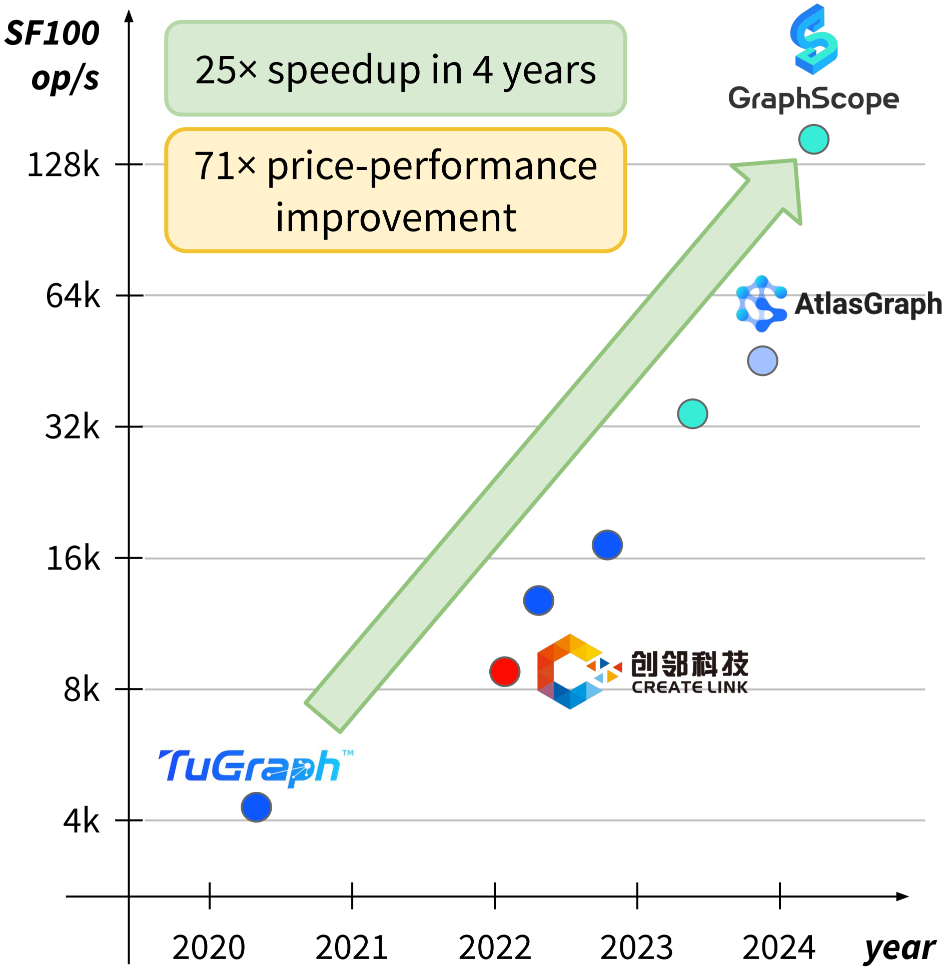


Queries start in 1–2 person nodes

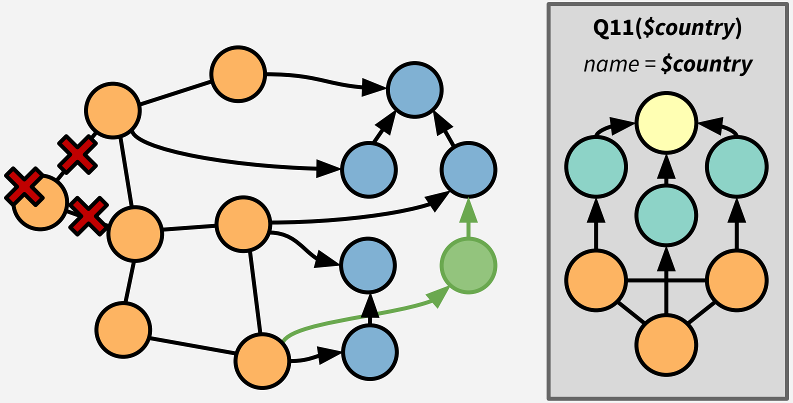
14 complex reads, 7 short reads

8 insert operations run concurrently

Goal: High throughput (ops/s)



SNB Business Intelligence (2022)



Queries touch on large portions of the data

20 complex read queries, insert & delete ops

Both bulk and concurrent updates allowed

Goal: High throughput & low query runtimes

Audited results

TigerGraph

SF100
SF1,000
SF10,000

TuGraph™

SF30,000

Using the SNB benchmarks

Making benchmarks easy to use

Specification

Academic paper

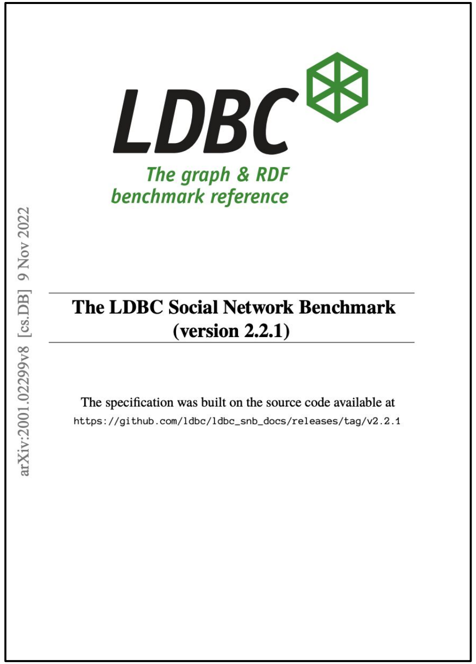
Data generator

Pre-generated data sets

Driver

2+ implementations

Guidelines



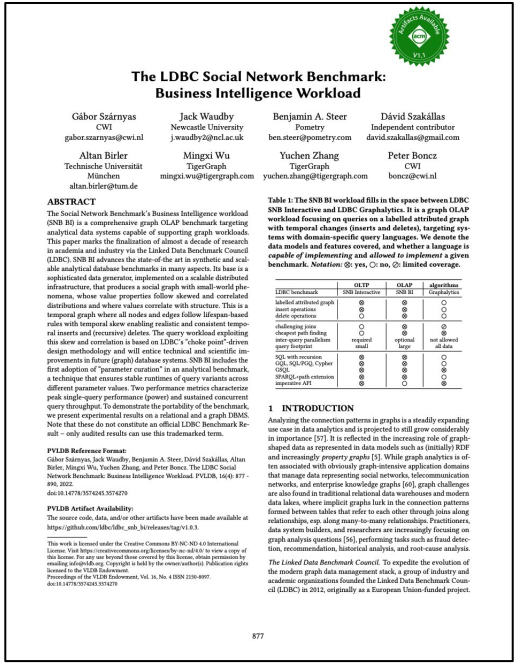
arXiv:2001.02299v8 [cs.DB] 9 Nov 2022

LDBC

The graph & RDF benchmark reference

The LDBC Social Network Benchmark (version 2.2.1)

The specification was built on the source code available at
https://github.com/ldbc/ldbc_snb_docs/releases/tag/v2.2.1



Artifacts Available acm V1.1

The LDBC Social Network Benchmark: Business Intelligence Workload

Gábor Szárnyas, CWI, gabor.szarnyas@cwi.nl
Jack Waudby, Newcastle University, j.waudby2@ncl.ac.uk
Benjamin A. Steer, Pometry, ben.steer@pometry.com
Dávid Szakállas, Independent contributor, david.szakallas@gmail.com
Altan Birler, Technische Universität München, altan.birler@tum.de
Mingxi Wu, TigerGraph, mingxi.wu@tigergraph.com
Yuchen Zhang, TigerGraph, yuchen.zhang@tigergraph.com
Peter Boncz, CWI, boncz@cwi.nl

ABSTRACT

The Social Network Benchmark's Business Intelligence workload (SNB BI) is a comprehensive graph OLAP benchmark targeting analytical data systems capable of supporting graph workloads. This paper marks the finalization of almost a decade of research in academia and industry via the Linked Data Benchmark Council (LDBC). SNB BI advances the state-of-the art in synthetic and scalable analytical database benchmarks in many aspects. Its base is a sophisticated data generator, implemented on a scalable distributed infrastructure, that produces a social graph with small-world phenomena, whose value properties follow skewed and correlated distributions and where values correlate with structure. This is a temporal graph where all nodes and edges follow lifespan-based rules with temporal skew enabling realistic and consistent temporal inserts and (recursive) deletes. The query workload exploiting this skew and correlation is based on LDBC's "choke point"-driven design methodology and will entice technical and scientific improvements in future (graph) database systems. SNB BI includes the first adoption of "parameter curation" in an analytical benchmark, a technique that ensures stable runtimes of query variants across different parameter values. Two performance metrics characterize peak single-query performance (power) and sustained concurrent query throughput. To demonstrate the portability of the benchmark, we present experimental results on a relational and a graph DBMS. Note that these do not constitute an official LDBC Benchmark Result – only audited results can use this trademarked term.

PVLDB Reference Format:
Gábor Szárnyas, Jack Waudby, Benjamin A. Steer, Dávid Szakállas, Altan Birler, Mingxi Wu, Yuchen Zhang, and Peter Boncz. The LDBC Social Network Benchmark: Business Intelligence Workload. PVLDB, 16(4): 877 - 890, 2022.
doi:10.14778/3574245.3574270

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/ldbc/ldbc_snb_bi/releases/tag/v1.0.3.

Proceedings of the VLDB Endowment, Vol. 16, No. 4 ISSN 2150-8097.
doi:10.14778/3574245.3574270

Table 1: The SNB BI workload fills in the space between LDBC SNB Interactive and LDBC Graphalytics. It is a graph OLAP workload focusing on queries on a labelled attributed graph with temporal changes (inserts and deletes), targeting systems with domain-specific query languages. We denote the data models and features covered, and whether a language is *capable of implementing* and *allowed to implement* a given benchmark. *Notation:* ⊗: yes, ○: no, ⊘: limited coverage.

	OLTP	OLAP	algorithms
LDBC benchmark	SNB Interactive	SNB BI	Graphalytics
labelled attributed graph	⊗	⊗	○
insert operations	⊗	⊗	○
delete operations	○	⊗	○
challenging joins	○	⊗	⊘
cheapest path finding	○	⊗	⊗
inter-query parallelism	required	optional	not allowed
query footprint	small	large	all data
SQL with recursion	⊗	⊗	○
GQL, SQL/PGQ, Cypher	⊗	⊗	○
GSQL	⊗	⊗	⊗
SPARQL+path extension	⊗	⊗	○
imperative API	⊗	○	⊗

1 INTRODUCTION

Analyzing the connection patterns in graphs is a steadily expanding use case in data analytics and is projected to still grow considerably in importance [57]. It is reflected in the increasing role of graph-shaped data as represented in data models such as (initially) RDF and increasingly *property graphs* [5]. While graph analytics is often associated with obviously graph-intensive application domains that manage data representing social networks, telecommunication networks, and enterprise knowledge graphs [60], graph challenges are also found in traditional relational data warehouses and modern data lakes, where implicit graphs lurk in the connection patterns formed between tables that refer to each other through joins along relationships, esp. along many-to-many relationships. Practitioners, data system builders, and researchers are increasingly focusing on graph analysis questions [56], performing tasks such as fraud detection, recommendation, historical analysis, and root-cause analysis.

The Linked Data Benchmark Council. To expedite the evolution of the modern graph data management stack, a group of industry and academic organizations founded the Linked Data Benchmark Council (LDBC) in 2012, originally as a European Union-funded project.

877

Auditing and trademark

Performed by certified auditors (>$20k + infrastructure costs, multiple weeks)

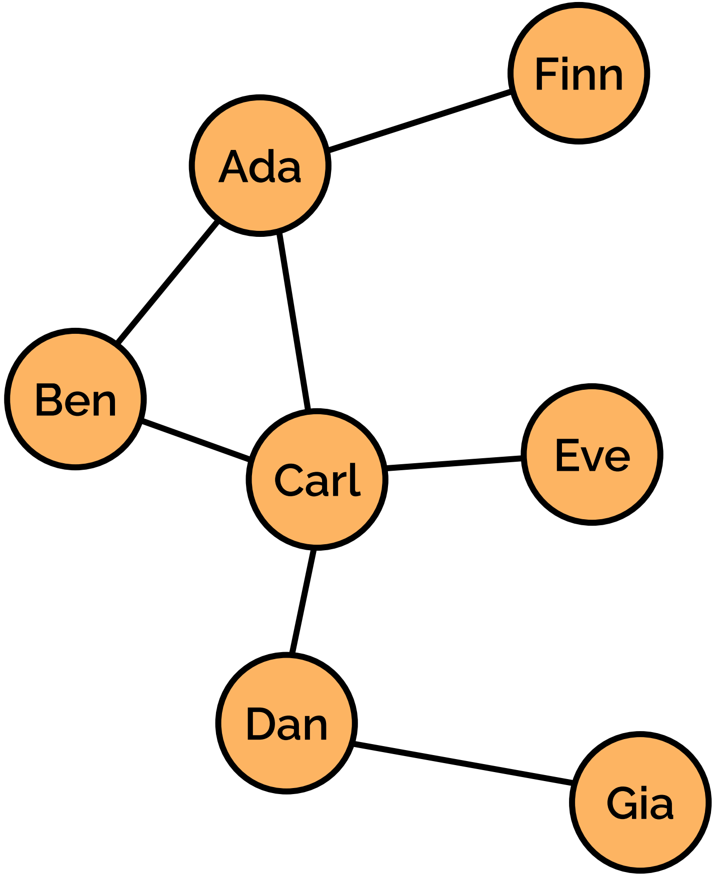
Trademark: only a result produced by an auditor is an "LDBC benchmark result"

Benchmark setup	SF	Hardware	Throughput	Documents
System: **GraphScope Flex 0.26.1** Test sponsor: **Alibaba Cloud** Date: **2024-05-14** Queries implemented in: **C++ stored procedures** System cost: **738,724 RMB**	100	Alibaba Cloud ecs.r8a.16xlarge, 512GiB RAM, 64×AMD EPYC 9T24 @ 3.7GHz vCPUs	130,098.36 ops/s	Full disclosure report Executive summary Signatures Supplementary package
	300	Alibaba Cloud ecs.r8a.16xlarge, 512GiB RAM, 64×AMD EPYC 9T24 @ 3.7GHz vCPUs	131,263.87 ops/s	
	1000	Alibaba Cloud ecs.r8a.16xlarge, 512GiB RAM, 64×AMD EPYC 9T24 @ 3.7GHz vCPUs	127,784.51 ops/s	

The LDBC Graphalytics Benchmark

LDBC Graphalytics

- Graphalytics = graph + analytics
- An LDBC benchmark for graph algorithm implementations
- A macrobenchmark
- No audits – competitions with leaderboard ranking
  (similar to HPC benchmarks such as Top500 and Graph500)



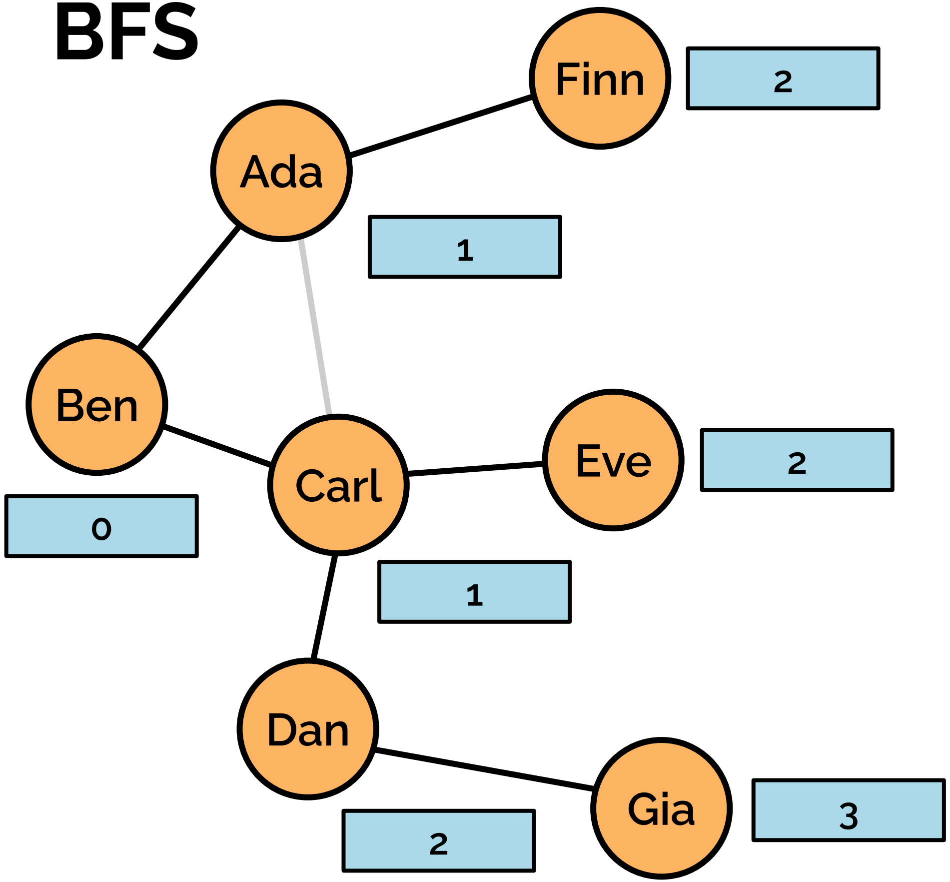
The data sets contain untyped, unattributed graphs with (optional) edge weights

LDBC SNB Datagen
Graph500
Twitter
Friendster
Patents
wiki-Talk
...

Largest graph:

- 450M vertices
- 34B edges

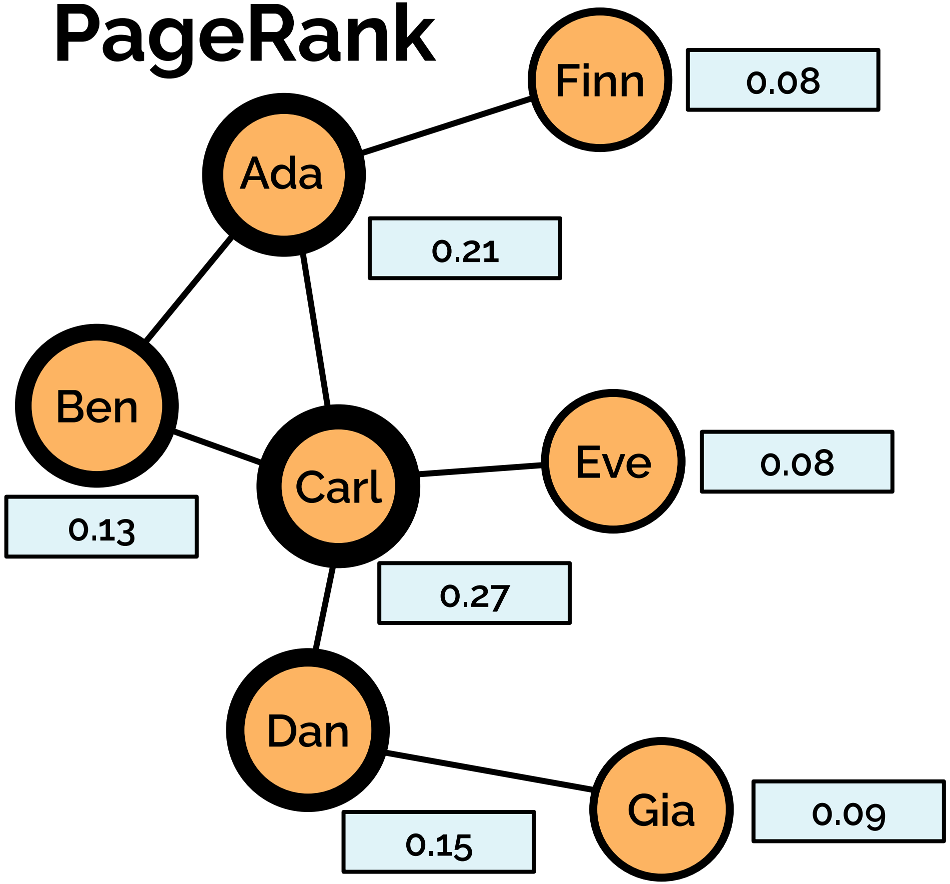
BFS



Graphalytics algorithms

- **Breadth-first search(** ***source*** **: “Ben”)**
- PageRank(*damping factor*: 0.85, *iterations*: 5)
- Local clustering coefficient
- Community detection using LP(*iterations:* 2)
- Weakly connected components
- Single-source shortest paths(source: “Ben”)

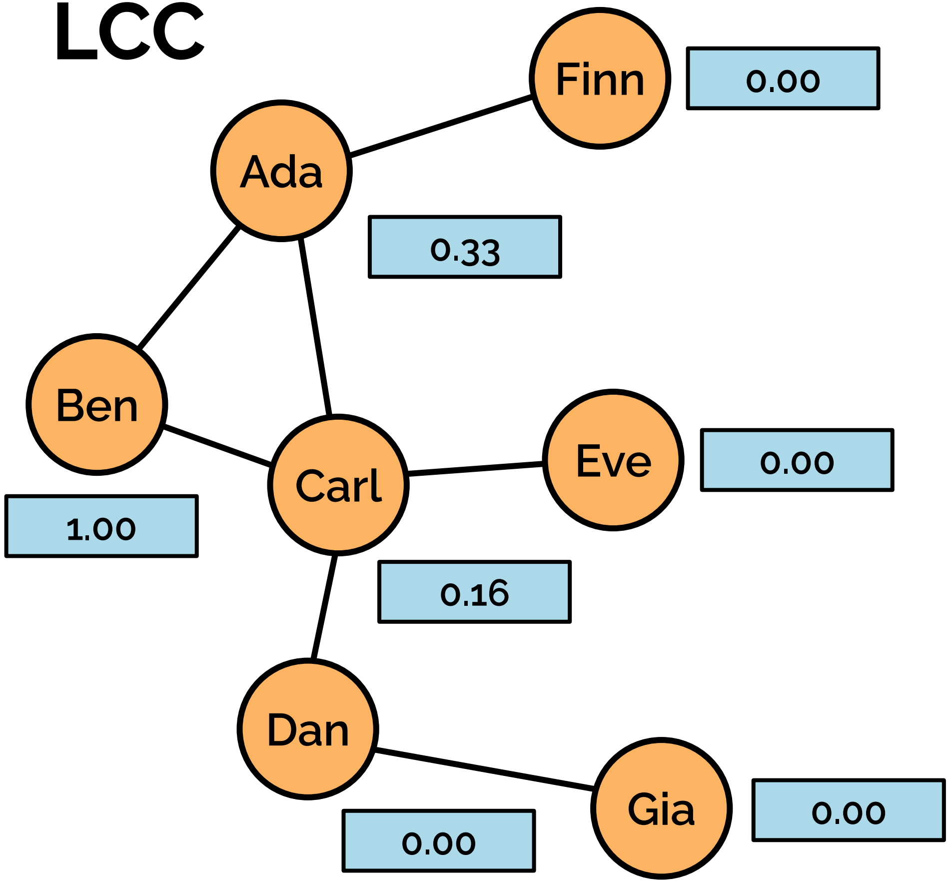
PageRank



Graphalytics algorithms

- Breadth-first search(*source*: “Ben”)
- **PageRank(*damping factor* : 0.85, *iterations* : 5)**
- Local clustering coefficient
- Community detection using LP(*iterations:* 2)
- Weakly connected components
- Single-source shortest paths(source: “Ben”)

LCC



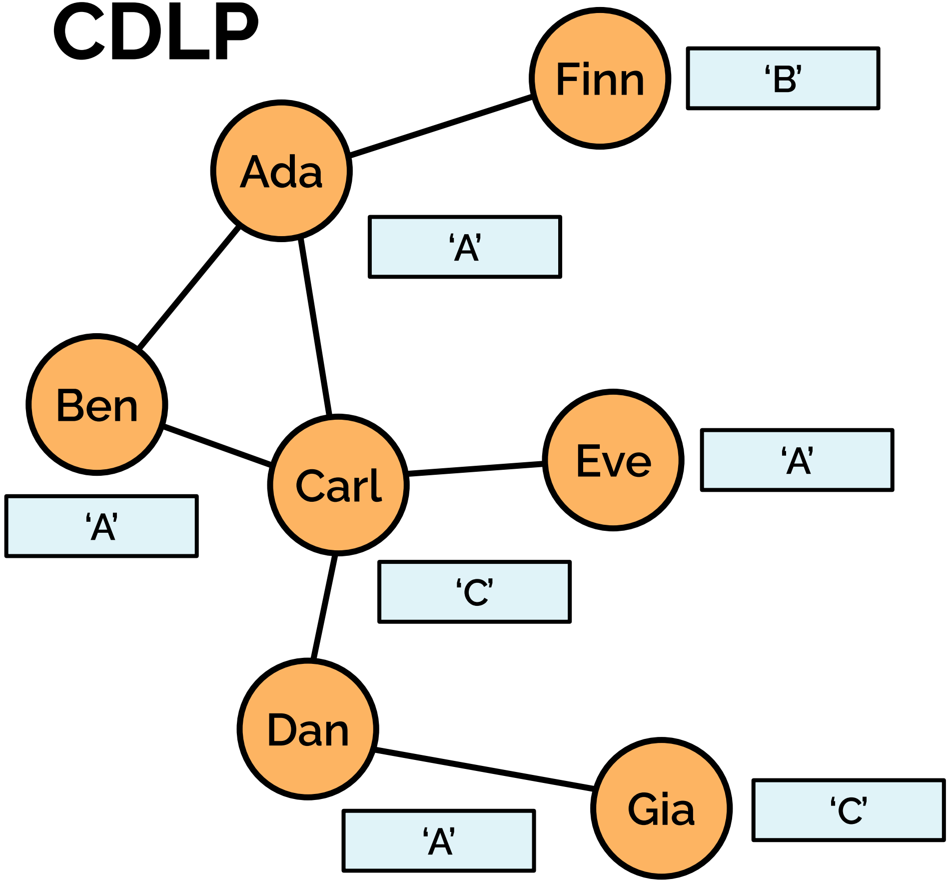
Graphalytics algorithms

Breadth-first search(*source*: “Ben”)
PageRank(*damping factor*: 0.85, *iterations*: 5)
Local clustering coefficient
Community detection using LP(*iterations:* 2)
Weakly connected components
Single-source shortest paths(source: “Ben”)

For each vertex, LCC is #triangles / #wedges.

Similar to triangle count.

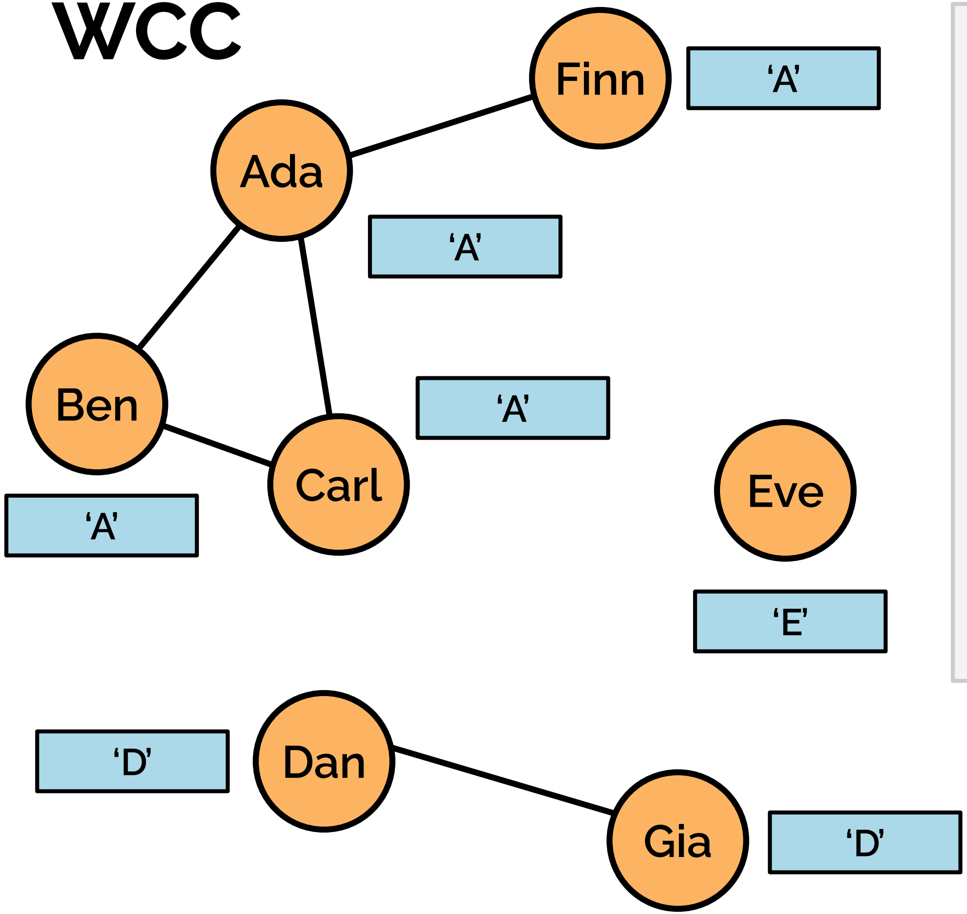
CDLP



Graphalytics algorithms
Breadth-first search(*source*: "Ben")
PageRank(*damping factor*: 0.85, *iterations*: 5)
Local clustering coefficient
Community detection using LP(*iterations:* 2)
Weakly connected components
Single-source shortest paths(source: "Ben")

In each iteration, the next label of a vertex is selected as *the minimum mode value among the labels of the neighbours.*

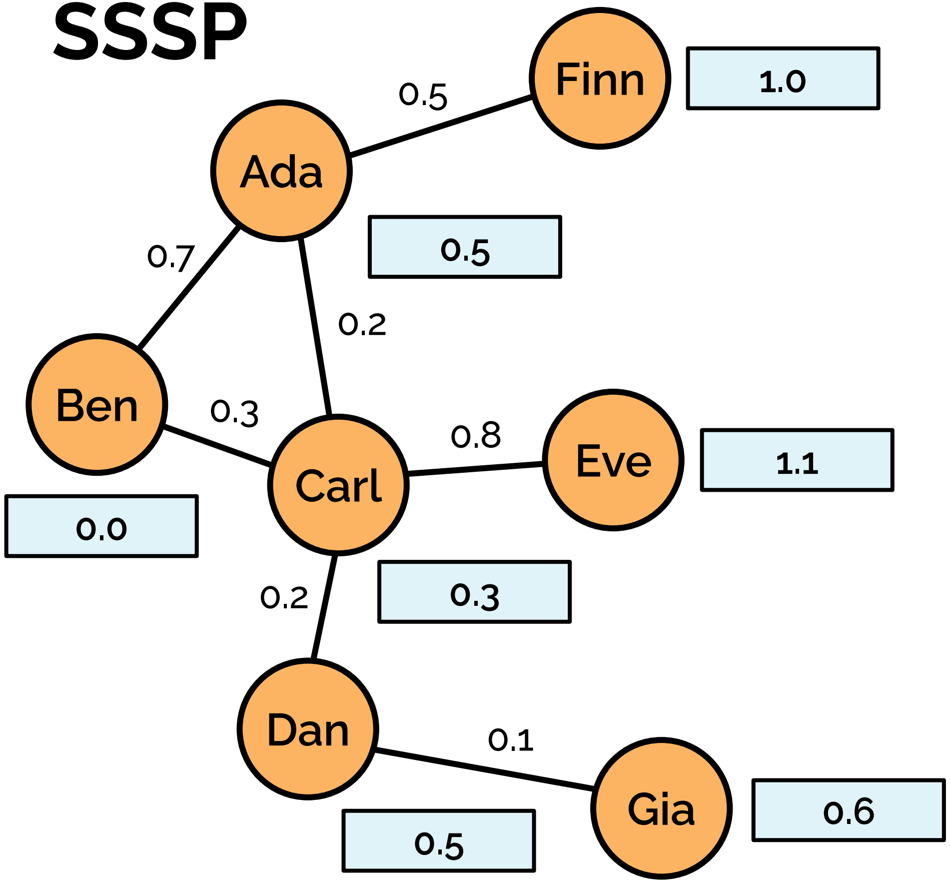
WCC



Graphalytics algorithms

- Breadth-first search(*source*: "Ben")
- PageRank(*damping factor*: 0.85, *iterations*: 5)
- Local clustering coefficient
- Community detection using LP(*iterations:* 2)
- **Weakly connected components**
- Single-source shortest paths(source: "Ben")

SSSP



Graphalytics algorithms
Breadth-first search(*source*: "Ben")
PageRank(*damping factor*: 0.85, *iterations*: 5)
Local clustering coefficient
Community detection using LP(*iterations:* 2)
Weakly connected components
Single-source shortest paths(*source*: "Ben")

Most implementations are expected to use the delta-stepping SSSP algorithm.

Provisional Graphalytics leaderboard (2024)

#	Size	Platform	Environment name	Makespan throughput per dollar	Processing throughput per dollar
1	S	libgrape-gpu	ecs.ebmgn7e.32xlarge	0.08	58.35
2	S	libgrape-lite	ecs.r7.16xlarge	1.45	37.40
3	S	GraphBLAS Intel Xeon Gold 6342	bare metal, dedicated server	6.26	17.99
4	S	GraphBLAS Intel Xeon Platinum 8369	bare metal, dedicated server	7.84	15.73
5	S	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	3.05	11.45
1	M	libgrape-gpu	ecs.ebmgn7e.32xlarge	0.03	37.13
2	M	libgrape-lite	ecs.r7.16xlarge	0.49	25.44
3	M	GraphBLAS Intel Xeon Gold 6342	bare metal, dedicated server	1.94	5.88
4	M	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	1.12	5.74
5	M	GraphBLAS Intel Xeon Platinum 8369	bare metal, dedicated server	2.66	5.32

#	Size	Platform	Environment name	Makespan throughput per dollar	Processing throughput per dollar
1	L	libgrape-gpu	ecs.ebmgn7e.32xlarge	0.03	16.58
2	L	libgrape-lite	ecs.r7.16xlarge	0.14	8.99
3	L	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	0.43	2.27
4	L	GraphBLAS Intel Xeon Gold 6342	bare metal, dedicated server	0.62	2.13
5	L	GraphBLAS Intel Xeon Platinum 8369	bare metal, dedicated server	0.92	1.94
1	XL	libgrape-gpu	ecs.ebmgn7e.32xlarge	0.01	3.98
2	XL	libgrape-lite	ecs.r7.16xlarge	0.06	2.07
3	XL	GraphBLAS Intel Xeon Platinum 8369	bare metal, dedicated server	0.24	0.39
4	XL	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	0.08	0.28
1	2XL	libgrape-gpu	ecs.ebmgn7e.32xlarge	0.01	0.59
2	2XL	libgrape-lite	ecs.r7.16xlarge	0.05	0.33
3	2XL	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	0.02	0.05
4	2XL	GraphBLAS Intel Xeon Platinum 8369	bare metal, dedicated server	0.03	0.04
1	3XL	libgrape-lite	ecs.r7.16xlarge	0.01	0.07
2	3XL	Geacompute	ecs.c8i.24xlarge / ecs.c8a.48xlarge	0.01	0.01

LDBC benchmarks – Challenges

Areas to improve in

LDBC benchmarks don’t sufficiently cover some important recent technologies:

- cloud infrastructure and cloud-native systems
  - serverless systems
  - elasticity
- binary file formats (e.g. Parquet)
- user-defined functions
- ML workloads
  - graph neural networks
  - knowledge graphs
  - vector databases

Benchmark development and audits

Developing a new LDBC benchmark can take 5+ person-years:

- Without a standard language, implementations took a long time
- Hard to obtain a good baseline system (chicken-or-egg problem)

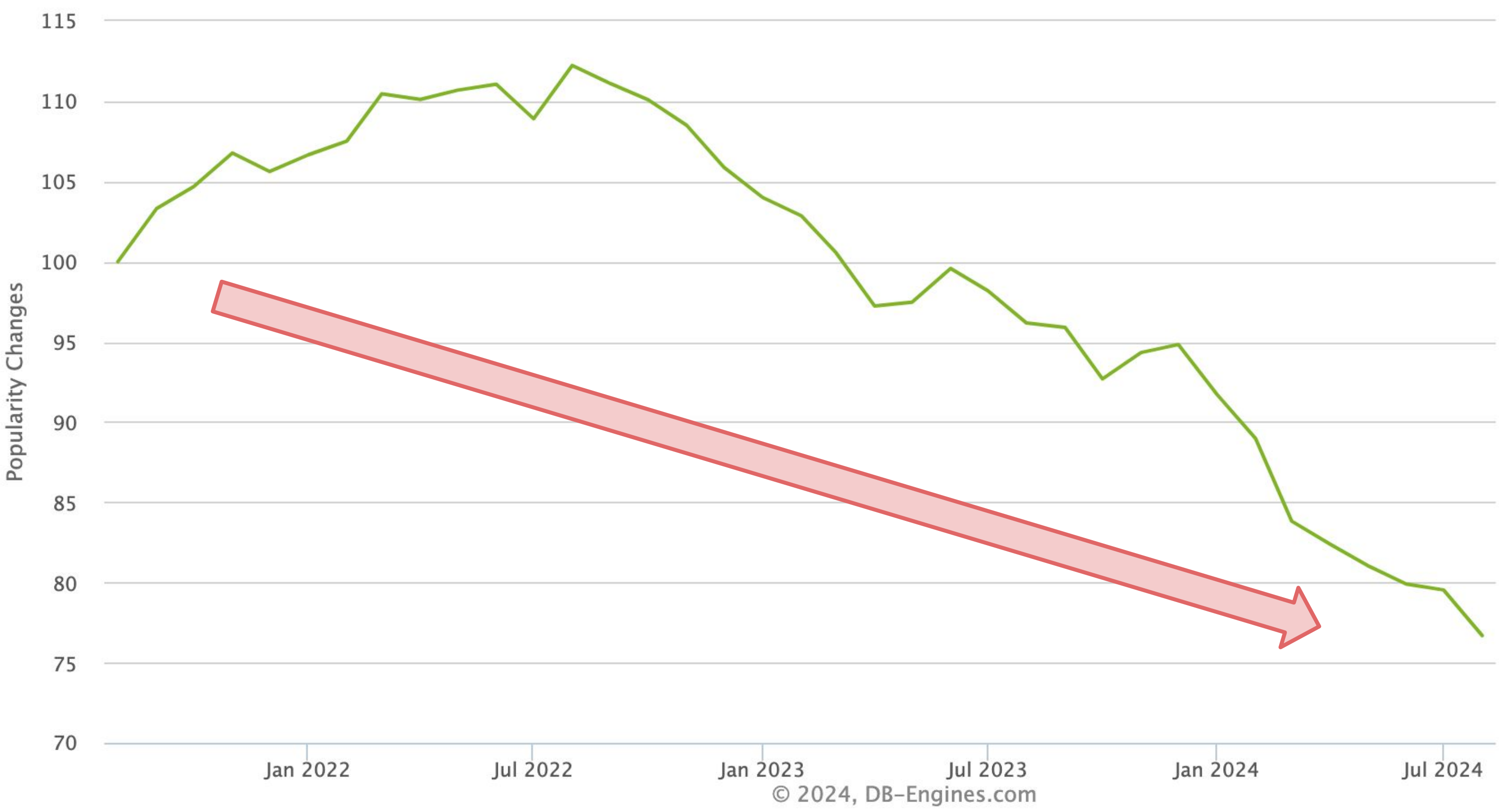
Audits:

- Most Interactive audited results use imperative languages
- Audits are long and expensive

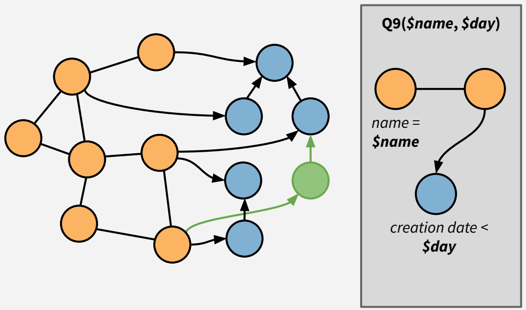
Data sets:

- Generate data for SF100k and beyond

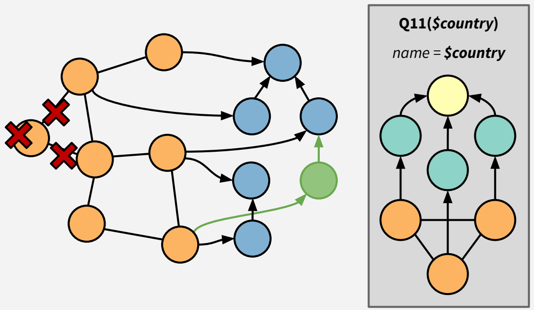
DB Engines Ranking for graph: ¼ drop in 3 years



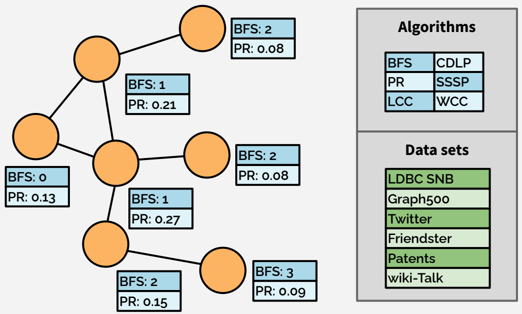
SNB Interactive v1



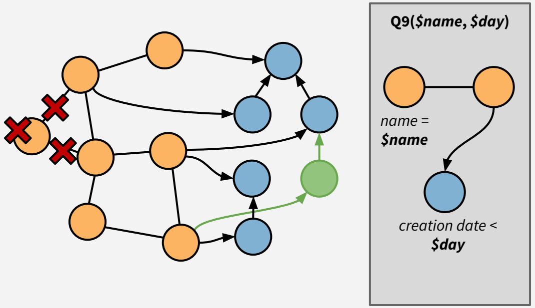
SNB Business Intelligence



Graphalytics



SNB Interactive v2



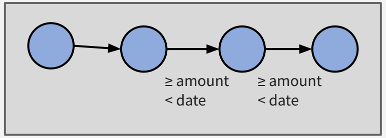
Semantic Publishing Benchmark

Target: **RDF/SPARQL**

Domain: **Media/publishing industry**

Inferencing & continuous updates

Financial Benchmark



Traversal with truncation

Strict latency bound (P99 < 100 ms)

LDBC

The graph & RDF
benchmark reference