The Linked Data Benchmark Council (LDBC): Driving Competition and Collaboration in the Graph Data Management Space

Gábor Szárnyas[1*], Brad Bebee[2], Altan Birler[3], Alin Deutsch[4,5], George Fletcher[6], Henry A. Gabb[7], Denise Gosnell[2], Alastair Green[8], Zhihui Guo[9], Keith W. Hare[8], Jan Hidders[10], Alexandru Iosup[11], Atanas Kiryakov[12], Tomas Kovatchev[12], Xinsheng Li[13], Leonid Libkin[14], Heng Lin[9], Xiaojian Luo[15], Arnau Prat-Pérez[16], David Püroja[1], Shipeng Qi[9], Oskar van Rest[17], Benjamin A. Steer[18], Dávid Szakállas[19], Bing Tong[20], Jack Waudby[21], Mingxi Wu[5], Bin Yang[13], Wenyuan Yu[15], Chen Zhang[20], Jason Zhang[13], Yan Zhou[20], and Peter Boncz[1]

[1] CWI, the Netherlands, [2] Amazon Web Services, [3] Technische Universität München, Germany, [4] UC San Diego, [5] TigerGraph, [6] TU Eindhoven, [7] Intel Corporation, [8] JCC Consulting, [9] Ant Group, [10] Birkbeck, University of London, [11] VU Amsterdam, the Netherlands, [12] Ontotext AD, [13] Ultipa, [14] University of Edinburgh; RelationalAI, [15] Alibaba Damo Academy, [16] *work done while at UPC Barcelona and Sparsity*, [17] Oracle, USA, [18] Pometry Ltd., [19] *individual contributor*, [20] CreateLink, [21] Newcastle University, School of Computing

* Corresponding author, `gabor.szarnyas@ldbcouncil.org`

Abstract. Graph data management is instrumental for several use cases such as recommendation, root cause analysis, financial fraud detection, and enterprise knowledge representation. Efficiently supporting these use cases yields a number of unique requirements, including the need for a concise query language and graph-aware query optimization techniques. The goal of the Linked Data Benchmark Council (LDBC) is to design a set of standard benchmarks that capture representative categories of graph data management problems, making the performance of systems comparable and facilitating competition among vendors. LDBC also conducts research on graph schemas and graph query languages. This paper introduces the LDBC organization and its work over the last decade.

1 Introduction

The graph data management space. The category of data management software with *graph features* has grown steadily in the last 15 years [36]. This includes graph databases [10], relational DBMSs with graph extensions [44], graph analytics libraries [26], and graph streaming systems [12]. While graph data management systems are already popular for several use cases, such as financial fraud detection, recommendation, and data integration [35], they did not yet reach mass adoption. We believe the two key obstacles to this are: (1) the lack of standardized query languages and APIs [35], (2) limited and/or unpredictable performance in systems [36]. LDBC makes significant efforts to address these problems.

Query languages. The adoption of graph processing systems, particularly those supporting the *property graph data model*, is considerably hindered by the lack of a standard query language [35]. Currently, systems use several different query languages, including Cypher, Gremlin, GSQL, PGQL, and DQL, which causes (potential) customers concern over lack of portability. Starting in 2017, a concentrated effort was launched to create standard query languages. The SQL/PGQ (Property Graph Queries) extension was released as part of SQL:2023 and the standalone GQL (Graph Query Language) is scheduled to be released in 2024. Both of these languages have been influenced by LDBC's G-CORE design language and LDBC has been involved in their design via its liaison with ISO.

Performance challenges. Graph processing problems, including graph pattern matching [32], graph traversal (navigation) [4], and graph mining [13], have irregular memory access patterns and provide little spatial locality or opportunities for data reuse [17]. Contemporary CPUs are ill-suited to handle these workloads, leading to performance problems [37]. Moreover, while there were attempts to harness modern hardware such as GPUs [38] and FPGAs [11], these only proved beneficial for narrow domains and did not generalize to a wider set of use cases.

The importance of benchmarks. To expedite the speed of progress in graph data management systems, a group of industry and academic organizations founded the Linked Data Benchmark Council (LDBC). LDBC is as an independent benchmarking organization, which defines standard benchmarks to make graph query performance measurable and thus facilitate competition between vendors. In this sense, LDBC aims to fulfill a role similar to the Transaction Processing Performance Council (TPC), which defined a number of influential benchmarks. LDBC uses TPC's design and auditing processes as inspiration for its operations.

LDBC benchmarks. LDBC has six main benchmark workloads covering different aspects of graph processing with different transactional characteristics, set of CRUD operations, and data distributions. With the exception of Graphalytics, a leaderboard-style benchmark, all benchmarks define stringent *auditing processes* for ensuring that implementations are faithful to the specification and the derived results are reproducible. As of August 2023, LDBC published 45 audited results.

Paper structure. This paper is structured as follows. Section 2 gives an overview of the LDBC organization, including its history and structure. Section 3 presents LDBC's benchmarks, Section 4 describes the benchmark creation and auditing processes, and Section 5 summarizes our benchmark design experiences. Section 6 introduces LDBC's working groups and Section 7 outlines future directions.

2 The LDBC organization

2.1 History of the organization

Research project (2012–2015). LDBC started as a European Union-funded research project by the same name[1] with the participation of 4 academic and

[1] FP7-ICT grant ID 317548, `https://cordis.europa.eu/project/id/317548`

4 industry partners [14,3]. The project was coordinated by Josep Larriba Pey from Universitat Politècnica de Catalunya and Peter Boncz (CWI & VU Amsterdam). The consortium designed the Social Network Benchmark suite (SNB, Section 3.2), releasing its first workload, SNB Interactive [19], and the Semantic Publishing Benchmark [28] (SPB, Section 3.3). The non-profit company "Linked Data Benchmark Council" was established and registered in the UK.[2]

Sustained research efforts (2016–2018). After the EU project concluded, research efforts continued with the participation of industry partners and resulted in G-CORE [5], a declarative language designed to formulate composable graph queries. The Graphalytics benchmark (Section 3.4) [25] was released, and a draft version of the SNB Business Intelligence workload was published [40].

Expansion and auditing ramp-up (2019–2022). LDBC's membership increased from 7 organizations in 2019 to 22 organizations in 2022 (Section 2.2). While in the previous phase LDBC was economically supported by CWI and Sparsity, one of the organizational improvements realized by Alastair Green was to get LDBC a bank account, to start collecting membership fees.[3] New working groups were established to research property graph schemas and query language semantics (Section 6). The SNB Business Intelligence workload was completed [41] (Section 3.2). A new Task Force was set up to design the FinBench [24] (Section 3.5). LDBC's benchmark adoption process (Section 4.1) and auditing processes crystallized (Section 4.2) with multiple audits occurring per year. The term "LDBC benchmark result" was trademarked (Section 4.3).

Restructuring and new benchmarks (2023–). In early 2023, the organization was restructured to simplify governance (Section 2.2). The benchmark Task Forces released the initial version of the FinBench, updated the SNB Interactive workload (Section 3.2), and organized a Graphalytics competition (Section 3.4).

2.2 Organizational structure and operations

Historical structure (2013–2022). Until 2023, LDBC member organizations could appoint a director to the *Board of Directors*, which at its peak consisted of 20+ members, an unusual and unwieldy structure for a small non-profit company.

Current structure (2023–). To simplify its governance, LDBC was restructured in 2023, resulting in new articles of association [29] and updated byelaws [30]. The new structure (Figure 1) has *Voting Members*, who contribute via the *Members Policy Council*[4] and *Associate Members*, who pay no fees (and have no vote) but contribute to the day-to-day work of LDBC. There is a new, smaller *Board of Directors* (3–5 members), who are also part of the Members Policy Council.

Membership. As of August 2023, LDBC has 24 member organizations, including database, hardware, and cloud computing vendors, and academic institutes.

[2] `https://find-and-update.company-information.service.gov.uk/company/08716467`

[3] This seemingly trivial matter posed a practical hurdle for an organization with many directors (one per member at the time) located in different parts of the world.

[4] The *Members Policy Council* is called the *Members Council* in official documents.

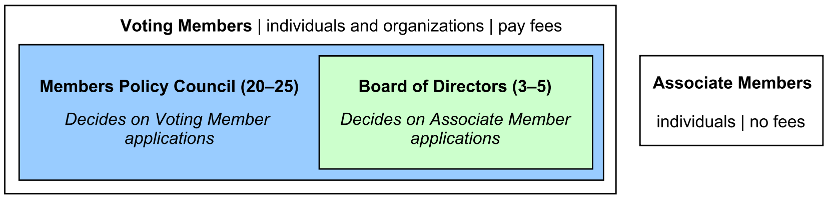
Voting Members | individuals and organizations | pay fees
Members Policy Council (20–25)
Decides on Voting Member applications
Board of Directors (3–5)
Decides on Associate Member applications
Associate Members
individuals | no fees

Fig. 1: Organizational structure of the LDBC from May 2023.

There are 3 sponsor companies[5], 18 member companies[6], and 3 non-commercial institutions[7]. LDBC has 3 individual voting members[8] and 60+ associate members.
Intellectual property rights. The intellectual property rights policies of LDBC cover assets created by members while participating in LDBC activities, including software components, written specifications, discussion proposals, academic papers, etc. [30]. Software contributions are licensed under a licence substantively identical to the Apache Software Licence 2.0; copyright for documentary or pictorial contributions are licensed to LDBC with a right to sublicence. Typically LDBC publishes contributions either to its members or to external standards groups like ISO, or generally to the public under the Creative Commons CC-BY licence. Participants in activities of LDBC are also required to comply with the LDBC Patent Rules, which include disclosing patents that may be infringed by the implementation of an LDBC benchmark specification, or by an external standard that incorporates contributions from LDBC.
Teams. LDBC's members form teams that work on specific aspects of graph processing. *Task forces* design, implement, and maintain benchmarks (Section 3). *Working groups* conduct research on graph query languages and graph schema (Section 6). The establishment of these teams is initiated by the creation of a *work charter* and is voted on by the Members Policy Council.
Finances. LDBC's sources of revenue are its membership and auditing fees. As of 2023, membership costs 1,100 GBP for non-commercial institutes, 2,200 GBP for commercial companies, and 8,800 GBP for sponsor members. A fee of 2,000 GBP is applied to audits commissioned by non-sponsor members. Individual associate membership is free. LDBC uses its funds to pay for cloud compute, storage, and collaboration services in addition to usual company overheads.

2.3 Liaison with ISO on standard query languages (GQL, SQL/PGQ)

In 2017, the international standards committee responsible for the SQL database language standard (ISO/IEC/JTC 1/SC 32/WG 3 "Database Languages")[9],

[5] Ant Group, Beijing Volcano Engine Technology Co., and Oracle Labs
[6] Amazon, Alibaba Damo Academy, ArangoDB, Beijing Haizhi Xingtu Company, CreateLink, Fabarta, Intel, JCC Consulting, Katana Graph, Memgraph, Neo4j, Ontotext, Pometry, RelationalAI, Sparsity Technologies, TigerGraph, Ultipa, and vesoft
[7] FORTH; Birkbeck, University of London; and Zhejiang Lab
[8] Peter Boncz, Alexandru Iosup, and Gábor Szárnyas; all co-authors of this paper
[9] `https://www.iso.org/organization/6720817.html`

established a Category C Liaison relationship with LDBC. This liaison allows WG 3 to share its working documents, draft specifications, and draft digital artifacts with LDBC participants. This access gives LDBC members early visibility into standards development efforts.

Peter Boncz, Alastair Green, and Jan Hidders are LDBC's liaison participants on the official ISO roster for WG 3, and can participate in any WG 3 meeting. This liaison relationship has helped to make the SC 32/WG 3 work more visible to organizations participating in LDBC and give the LDBC work more visibility in the Database Language standards process.

WG 3 has been working on two projects that are of interest to LDBC members:

- ISO/IEC 9075-16 Information technology — Database languages SQL — Part 16: Property Graph Queries (SQL/PGQ)
- ISO/IEC 39075 Information technology — Database languages — GQL

SQL/PGQ is completed and was published by ISO at the end of May 2023. GQL is currently undergoing a Draft International Standard (DIS) ballot that started on 2023-05-23 and ends on 2023-08-15. WG 3 aims to resolve any issues identified during the DIS ballot and to have the GQL standard ready for publication in early 2024.

Within the database language standards committees, there is ongoing work to expand the Graph Pattern Matching (GPM) language [18] in areas such as cheapest path queries. This GPM work will be integrated into the next editions of both SQL/PGQ and GQL.

The LDBC Extended Schema (LEX) working group [21] (Section 6.4) aims to propose expanded schema capabilities in a future edition of the GQL standard.

2.4 Technical User Community (TUC) meetings

Table 1: LDBC Technical User Community meetings between 2018 and 2023.

#	Year	Date	Location	Format	Program
16	2023	June 23–24	Seattle, WA	hybrid	31 talks
15	2022	June 17–18	Philadelphia, PA	hybrid	26 talks
14	2021	August 16	Copenhagen, Denmark	hybrid	19 talks
13	2020	June 30–July 1	*online*	online	4 sessions
12	2019	June 5	Amsterdam, the Netherlands	in-person	13 talks
11	2018	June 8	Austin, TX	in-person	11 talks

Since 2012, LDBC organizes Technical User Community (TUC) meetings.[10] These are 1–2 day informal workshops, where LDBC's leaders, task forces, and working groups report on their progress. Additionally, member companies give updates of their products, researchers in the graph space discuss their latest

[10] `https://ldbcouncil.org/tags/tuc-meeting/`

results, and users of graph data management systems present their use cases. The meetings provide an opportunity for members to contribute to LDBC's *choke points* (Section 3.1), and to influence the future direction of LDBC. In recent years, the TUC meetings have been steadily gaining popularity (Table 1).

3 Benchmarks

Table 2: Key characteristics of LDBC benchmarks. *Scale:* size of the largest data set, *GP lang.:* is the use of general-purpose programming languages allowed for implementations? *Req. isol.:* required isolation level (SI: snapshot isolation, RC: read committed). Legend: ⊗ yes, ○ no, ⊘ optional; ⊛ the benchmark is under design and audits are not yet possible; (i) larger sizes can be generated using the Graphalytics graph generator, but are not part of the standard benchmark.

Benchmark	Year	Workload	Scale	Min. scale	#Queries	#Audits	Inserts	Deletes	GP lang.	ACID test	Req. isol.
SNB BI	2022	analytical	30,000	30	20	4	⊗	⊗	○	⊘	SI
SNB Interactive v1	2015	transactional	1,000	30	21	24	⊗	○	⊗	⊗	RC
SNB Interactive v2	⊛	transactional	30,000	30	21	⊛	⊗	⊗	⊗	⊗	SI
SPB	2015	transactional	5	1	12	17	⊗	⊗	○	○	RC
FinBench	2023	transactional	10	0.1	40	0	⊗	⊗	⊗	⊗	RC
Graphalytics	2016	algorithms	320$^{(i)}$	–	6	–	○	○	⊗	–	–

In this section, we describe LDBC's benchmarks. We first present LDBC's common benchmark terminology (Section 3.1). We then describe the workloads of the Social Network Benchmark suite (SNB, Section 3.2), followed by the Semantic Publishing Benchmark (SPB, Section 3.3), the FinBench (Section 3.5), and Graphalytics (Section 3.4). The benchmarks are summarized in Table 2. Systems that implement at least two LDBC benchmarks are shown in Table 3.

3.1 Benchmark terminology

Choke points. LDBC's benchmark design process uses *choke points* [15], i.e. well-chosen technical difficulties that are challenging for the present generation of data processing systems and whose optimization likely results in significant overall performance improvements. The choke points are identified by expert data systems architects and also subject to feedback received at the Technical User Community meetings (Section 2.4). LDBC workloads are designed to cover the set of choke points triggered by a given workload category.
Auditing. Similarly to TPC's *Enterprise Class benchmarks* [34], most of LDBC's benchmarks must undergo an auditing process conducted by a certified auditor

Table 3: Systems with implementations for 2+ LDBC benchmarks. Legend: $\otimes$ full, $\oslash$ partial, $\bigcirc$ no implementation; (a) audited implementation; (s) the implementation was used for a standard-establishing audit.

Benchmark	**CreateLink Galaxybase**	**Ontotext GraphDB**	**Graphscope-Flex**	**Neo4j**	**PostgreSQL**	**Sparsity Sparksee**	**Microsoft SQL Server**	**TigerGraph**	**Ant Group TuGraph**	**Umbra**	**OpenLink Virtuoso**
SNB BI	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\otimes$	$\otimes$	$\oslash$	$\bigcirc$	$\otimes^{(a)}_{(s)}$	$\bigcirc$	$\otimes_{(s)}$	$\oslash$
SNB Interactive v1	$\otimes^{(a)}$	$\otimes^{(a)}$	$\otimes^{(a)}$	$\otimes$	$\otimes$	$\otimes_{(s)}$	$\oslash$	$\otimes$	$\otimes^{(a)}$	$\otimes$	$\otimes_{(s)}$
SNB Interactive v2	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\otimes$	$\otimes$	$\bigcirc$	$\otimes$	$\bigcirc$	$\oslash$	$\otimes$	$\bigcirc$
SPB	$\bigcirc$	$\otimes^{(a)}$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\otimes^{(a)}$
FinBench	$\otimes_{(s)}$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\otimes_{(s)}$	$\bigcirc$	$\bigcirc$
Graphalytics	$\otimes$	$\bigcirc$	$\otimes$	$\otimes$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\bigcirc$	$\otimes$	$\oslash$	$\bigcirc$

before they can be published as official results. The auditors check compliance with the specification, run the benchmark independently and present their findings in a *full disclosure report* (FDR). The FDR documents the benchmark setup and the derived results in detail, typically spanning over 20–50 pages. Additionally, audited benchmark results are accompanied by a *supplementary package*, which includes the benchmark implementation and the binary of the system-under-test (SUT), ensuring that the results are reproducible.

Scale factors. LDBC's benchmark suites include data generators that produce synthetic data sets of increasing sizes. Each data set is characterized by its *scale factor* (SF) which corresponds to the data set's disk usage when serialized in CSV (comma-separated values) format, measured in GiB.

ACID compliance. Several of LDBC's benchmarks require the SUT to comply with ACID properties. An important aspect of this is *durability*: the SUT must be able to recover from a crash or power outage without losing any committed data.[11] For the SNB workloads and FinBench, the *isolation* properties are tested with an ACID test suite.

3.2 Social Network Benchmark (SNB) suite

The Social Network Benchmark suite pioneered a number of techniques used in LDBC benchmarks: choke point-driven design [15], scalable correlated dynamic graph generation [33,43], and parameter curation for stable query runtimes [22]. The detailed specification of the SNB workloads is available at [6].

SNB data generators. The first version of the SNB data generator was implemented in Hadoop and only supported insert operations [33]. In 2020, it was ported to Spark for improved scalability,[12] and was extended with support for

[11] Unlike TPC, LDBC does not require systems to tolerate hardware failures.

[12] `https://github.com/ldbc/ldbc_snb_datagen_spark`

producing deep (cascading) delete operations [43]. To the best of our knowledge, its ability to generate a scalable graph where structure and values correlate, with flashmob-style spikes and deep delete operations are features unique to the SNB data generator [16].

SNB Interactive workload v1

The SNB Interactive v1 workload was published in 2015 [19]. It is a transactional benchmark that targets OLTP systems with graph features (e.g. path-finding). The workload consists of three types of operations: 14 complex read queries, 7 short read queries, and 8 inserts. The workload has a balanced mix of operations with approximately 8% complex reads, 72% short reads, and 20% inserts.

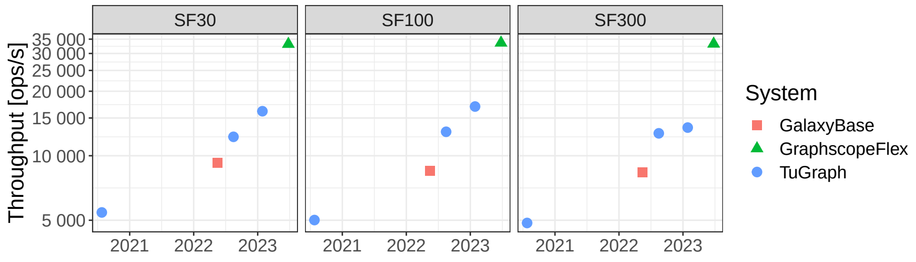

Fig. 2: Throughput of the audited results over time for the SNB Interactive v1 workload for scale factors 30, 100, 300 for implementations using general-purpose programming languages (C++ and Java), obtained between 2021 and 2023.

SNB Interactive v1 has been influential in the graph data management space: as of June 2023, 24 audited results were published using this workload.[13] Figure 2 shows the top 15 audited results published between 2021 and 2023, which demonstrate a performance increase of more than 6× over this period.

SNB Business Intelligence workload

The SNB Business Intelligence (BI) workload [41] captures an OLAP scenario with heavy-hitting analytical queries that touch on large portions of the graph (e.g. aggregating all Messages created within a 100-day period or exploring the neighbourhoods of the Persons living in China). The queries include multi-source *cheapest path-finding* (also known as *weighted shortest paths*), cyclic graph patterns, and have correlated and anti-correlated variants. The workload uses an improved data generator which produces fully dynamic graphs with insert and delete operations [43], and scales to SF30,000. The BI workload targets both DBMSs and data analytical systems such as Spark. To this end, updates can be applied in two modes: in *concurrent read–write mode*, reads and writes

[13] https://ldbcouncil.org/benchmarks/snb-interactive/

are executed concurrently, while in *disjoint read–write mode*, alternating blocks of reads and writes (accounting for one day's worth of updates) are executed. The BI workload was officially approved in November 2022. Since then, it has accumulated 4 audited results, including one for SF10,000. [14]

SNB Interactive workload v2

In 2022, the SNB task force initiated the renewal of the Interactive v1 workload. The Interactive v2 workload adopted several features from the BI workload: support larger scale factors and delete operations, and coverage of the *cheapest path-finding* algorithm. Moreover, the updated workload introduces a new temporal parameter curation algorithm to ensure stable runtimes for path queries evaluated on a fully dynamic graph, and features a complete refactoring of the driver along with several usability improvements. The balance of the read and insert queries are similar to Interactive v1 workload with only 0.2% delete operations added – motivated by the fact that deletes are rare compared to inserts in most real-life systems [2]. As of June 2023, the workload is in draft phase with four complete implementations already available (Table 3).

3.3 Semantic Publishing Benchmark (SPB)

The Semantic Publishing Benchmark (SPB) targets RDF database engines and is inspired by the BBC's Dynamic Semantic Publishing approach [28]. The scenario behind the benchmark considers a media that deals with a large volume of streaming content, namely articles and other "creative works". This content is enriched with metadata that describes it and links it to reference knowledge – taxonomies and databases that include relevant concepts, entities, and factual information. This metadata allows publishers to efficiently retrieve relevant content and increase engagement.

The main interactions with the repository are (i) updates, which add new creative works or alter the repository, and (ii) aggregation queries, which retrieve content. The engine should handle updates executed concurrently with a massive amount of aggregation queries. It must be able to deal with graph pattern matching, reasoning, geospatial constraints, and full-text search. The SPB includes ontologies, a data generator, query patterns, as well as test and benchmark drivers. The detailed specification of SPB version 2.0 is available at [27].

3.4 Graphalytics

The LDBC Graphalytics benchmark targets systems that evaluate graph algorithms on static graphs. While the scope of Graphalytics is similar to the influential GAP Benchmark Suite [9], Graphalytics has slightly different algorithms, more data sets (38 vs. 5), and enforces determinism in its algorithms.

[14] `https://ldbcouncil.org/benchmarks/snb-bi/`

Algorithms. The selection of graph algorithms for Graphalytics was motivated by input received during the TUC meetings (Section 2.4) and a literature survey of over 100 academic papers [23]. The benchmark includes the following six algorithms: breadth-first search (BFS), community detection using label propagation (CDLP), local clustering coefficient (LCC), PageRank (PR), single-source shortest paths (SSSP), and weakly connected components (WCC). Algorithms were adjusted if necessary such that they are *deterministic:* BFS returns the *distance* (level) for each node in the traversal (instead of picking one of the parent nodes), while the tie-breaking strategy employed in the CDLP algorithm always picks the smallest label (instead of picking randomly).
Data set. Graphalytics uses untyped and unattributed graphs. The data set of the benchmark includes both directed and undirected graphs. Some graphs have edge weights, which are used exclusively by the SSSP algorithm.
Graphalytics competition. Presently, it is not possible to commission audits for Graphalytics. Instead, LDBC organizes competitions with leaderboards, in the style of the Top500[15] and Graph500[16] competitions of the high-performance computing community. Solutions compete for both performance (execution time) and scalability. Implementers are also required to report system prices following the TPC Pricing Specification [42].

3.5 FinBench

The FinBench (short for "Financial Benchmark") is a benchmark, new in 2023, specifically designed to evaluate the performance of graph database systems in financial scenarios, such as anti-fraud and risk control, by employing financial data patterns and query patterns. This collective effort led by Ant Group adopts its rich experience in financial services, making it more applicable to graph users in the financial industry than previous LDBC benchmarks. Sharing some framework similarities with SNB, it distinguishes itself through differences in datasets and workloads.
Data pattern. Financial graphs are distinguished from the social network in SNB by having hub vertices with higher degrees and allowing edge multiplicity. Financial graphs are asymmetric directed graphs causing more unbalanced load and less spatial locality. The maximum degrees of hub vertices are in the magnitude of thousands in the social network generated by the SNB data generator, while the degree of hub vertices in the financial graphs generated by the FinBench data generator[17] may scale up to millions in large data scales. The higher degree of hub vertices poses new challenges to the performance of systems. The edge multiplicity means multiple edges of the same type can exist between the same source vertex and destination vertex. This requires support in the system storage and provides optimization opportunities for filtering during traversal.

[15] https://www.top500.org/
[16] https://graph500.org/
[17] https://github.com/ldbc/ldbc_finbench_datagen

Transaction workload. The *transaction workload* is the first workload included in the initial version of FinBench targeting OLTP data management systems, as the SNB Interactive workload [19] does. The key features of transaction workload include read-write query, time-window filtering, and recursive path filtering. The read-write query is a new query type, which wraps a complex query in a transaction. The write query to execute is determined by the result of the complex read. Time-window filtering is a pattern when financial businesses focus on the data, especially the edges, in a specific time range. Recursive path filtering is a pattern used by financial businesses to find fund traces in the graph. A fund trace may match the filter that the timestamp increases and the amount decreases of the edges sourcing from the origin vertex hop-by-hop. These typical features bring new challenges to the performance of systems. The transaction workload has 12 complex read, 6 simple read, 19 write, and 3 read-write queries.

The initial version of FinBench was collaboratively developed by nine leading graph system vendors. It underwent cross-validation on three systems: TuGraph, Galaxybase, and UltipaGraph. For more detailed information, please refer to the repositories on GitHub: FinBench specification[18], FinBench driver[19], reference implementation of the FinBench transaction workload[20], FinBench ACID suite[21].

4 Benchmark processes

4.1 Defining new LDBC benchmarks

LDBC has a strict process for proposing new benchmarks. Based on our experience, the initial benchmark completion (phases 1 to 3) takes at least 2 years, followed by adoption and maintenance (phases 4 and 5), which may span 5+ years.

Phase 1: Benchmark proposal. The benchmark proposer shall create a draft proposal. The benchmark must be motivated by real-world use cases and target a category of data processing systems that tackle some aspect of graph processing. The designers must reason why the benchmark is significantly different from existing LDBC benchmarks by identifying new performance challenges and formulating their choke points, providing data with unique characteristics (e.g. distribution, frequency/type of updates), etc. The benchmark draft shall be presented the benchmark to the Members Policy Council to gather feedback.

Phase 2: Collaboration setup. The proposer shall gather agreements from 2+ member companies who are willing to contribute to the benchmark specification and create reference implementations. They shall create a work charter for the benchmark task force (e.g. [24,39]), which includes the list the members interested in working on the benchmark. This shall be presented to the Members Policy Council, which votes *on the establishment of a new benchmark task force.*

[18] `https://github.com/ldbc/ldbc_finbench_docs`
[19] `https://github.com/ldbc/ldbc_finbench_driver`
[20] `https://github.com/ldbc/ldbc_finbench_transaction_impls`
[21] `https://github.com/ldbc/ldbc_finbench_acid`

Phase 3: Detailed benchmark design. The task force shall create the detailed benchmark specification, implement the data generator and the benchmark driver. The creation of at least 2+ complete reference implementations is required. The task force shall ensure that the specification contains the description of the data set, queries, and workload as well as the detailed auditing guidelines. They shall publish the specification in an open repository and release the software components as open-source. The task force shall conduct 2+ *standard-establishing audits.* These include the complete execution of the benchmark and the creation of FDR that detail their outcomes. The task force shall submit all resulting documents to the Members Policy Council, which votes *on the acceptance of the benchmark.*
Phase 4: Adoption and auditing. The task force shall help adoption attempts by working closely with the benchmark's early users. They should create training material and exam questions for auditor exams, then train and certify auditors. Certified auditors can fulfill incoming audit requests, initially with close collaboration with the benchmark task force.
Phase 5: Maintenance and renewal. The task force shall maintain the benchmark and optionally assist in further adoption and auditing attempts. If the benchmark remains popular, the task force should consider renewing it after 5–10 years to ensure its continued relevance.

4.2 Auditing process

Phase 1: Preparation (1–2 weeks). The Test Sponsor shall be an LDBC member company. If the Test Sponsor is not an LDBC sponsor member company, it shall pay LDBC an auditing fee of 2,000 GBP (as of 2023). The Test Sponsor shall create an initial version of the supplementary package of the benchmark and send it to an LDBC-certified Auditor for a preliminary review. The Test Sponsor shall establish the costs of its system setup using the TPC Pricing Specification [42].
Phase 2: Audit setup (3–6 weeks). The Test Sponsor and the Auditor shall negotiate the timeline and pricing of the audit, and sign a contract. The Test Sponsor shall hand over the supplementary package to the Auditor. Continuous communication in the form of emails, online meetings, DMs, etc. between the Auditor and the Test Sponsor is recommended for status updates and clarifications.
Phase 3: Auditing (3–10 weeks). For the SNB workloads, an audit consists of the following steps: (1) set up system, (2) run cross-validation, (3) perform code review, (4) run ACID isolation tests, (5) perform recovery tests, (6) conduct benchmark runs on the scale factors requested by the Test Sponsor, (7) write the FDR and the executive summary. The Test Sponsor then reviews the FDR and executive summary documents. If everything is in order, the Auditor, the Test Sponsor, and the leader of the task force sign the FDR.
Phase 4: Dissemination of results (1–2 weeks). The audit results are announced on the LDBC website, on mailing lists and on social media.
Timespan. The overall time required for an audit is between 8 and 20 weeks.

4.3 Trademark

To prevent misuse of the benchmarks, the term “LDBC benchmark result” is trademarked and is only allowed to be used for results that were achieved by an LDBC-certified Auditor in an official audit. That said, LDBC encourages use (including derived use) of its benchmarks provided that users comply with the *fair use policies*, described in LDBC’s Byelaws [30].

5 Benchmarking lessons learnt

This section captures our key lessons learnt with designing and maintaining benchmarks, and conducting audits with them.

High development costs. We found that creating domain-specific application-level benchmarks is a big undertaking. Realistic benchmarks are bound to be complex as they require a (somewhat) realistic scalable data generator and a high-performance driver with reference implementations. To make matters more complicated, the graph domain has highly skewed and correlated data sets, causing queries to be sensitive to parameter selection [22]. Path-finding queries on fully dynamic graphs are particularly susceptible to this, necessitating expensive parameter generation steps [41]. Creating reference implementations is also labour-intensive due to the lack of a standard query language – fortunately, this is expected to improve with the introduction of SQL/PGQ and GQL.

Data set availability is important. We found that users prefer downloading pre-generated data sets from an official repository instead of generating them using the data generators. To help adoption, it is best to make the data sets available in multiple serialization formats (different layouts, datetime formats, etc.) for all scale factors. However, this requires tens of terabytes of storage, leading to high storage costs. Moreover, we transferring large data sets stored at public cloud providers can be prohibitively expensive due to high egress fees (i.e. fees paid for transferring data out of the cloud). To work around this problem, we use storage services which do not have egress fees. For long-term archiving, we store our data in the SURF Data Repository,[22] which is operated by the Dutch national HPC support center, and offers tape-based storage. For short-term data distribution, we use Cloudflare’s R2 service.

Shift to the cloud. We observed a shift to the cloud for database management [1]: approximately half of LDBC’s graph vendors have a cloud offering and some have cloud-native systems with no on-premise solutions. The use of the cloud is also popular for running audits (and is encouraged by LDBC for easier reproducibility): 35 out of 49 audits conducted so far were executed in the cloud.

Finding problems beyond peformance. While the main goal of a benchmark implementation is to measure performance and to identify bottlenecks, implementing a full workload often leads to the discovery of other issues. Namely, we have found several issues such as insufficient query language features, correctness bugs, concurrency issues on different CPU architectures, crashes on large data

[22] `https://github.com/ldbc/data-sets-surf-repository`

sets, durability errors, parameter handling errors, issues with datetime and string handling, and deadlocks caused by concurrent transactions. The availability of public benchmark data sets made these errors easy to reproduce and reason about, leading to significant improvements in the systems-under-test.

6 Working groups

LDBC's working groups conduct research on areas related to graph query languages, including formalization of (sub)languages and exploring possibilities for defining graph schemas.

6.1 Graph Query Languages working group

The *Graph Query Languages* working group created the G-CORE design language [5], which treats paths as first-class citizens and supports the composability of graph queries. While the working group ceased to exist after the publication of G-CORE, LDBC has a liaison with ISO (Section 2.3) that facilitates continued collaboration on query language standards.

6.2 Formal Semantics Working Group (FSWG)

The *Formal Semantics Working Group* (FSWG) gives formal treatment to standard graph query languages to prevent ambiguous interpretations. To this end, it formalized the Graph Pattern Matching (GPM) language of GQL and SQL/PGQ [18]. The group also produced a formal summary of the GQL language [31] and created a pattern calculus for property graphs [20] that serves as a theoretical basis of GPM.

6.3 Property Graph Schema Working Group (PGSWG)

Initial graph database systems were schemaless, which hindered their adoption in several enterprise domains. The *Property Graph Schema Working Group* (PGSWG) investigates the problem of defining schemas for the property graph data model [8]. The group also identified ways to define keys in property graphs [7].

6.4 LDBC Extended GQL Schema (LEX)

The *LDBC Extended GQL Schema* (LEX) working group was established in 2022 with an initial membership of 10 organizations and approximately 20 individuals [21]. The group aims to propose the addition of a future extended schema definition language to the GQL standard, which allows for more elaborate (and therefore more restrictive) constraints on the permitted values of GQL property graphs than can be imposed by graph types as defined in the GQL DIS. These additional constraints aim to establish parity with the constraints available in SQL schema, to incorporate features described in PG-Schema [8] and to support performant processing of incremental transactional updates of a graph database.

7 Conclusion and future outlook

In this paper, we summarized LDBC's history, organization structure, community management; as well as its benchmarks, working groups, and processes. At the time of writing (June 2023), LDBC has a healthy benchmark ecosystem, which is actively maintained and renewed. Our benchmarks are used by a number of database vendors for both internal benchmarking as well as impartial comparisons via audits, which are now performed routinely.

In the future, we plan to further improve our benchmarks, including support for larger scale factors (up to SF100,000 for SNB). We also plan to investigate the impact of incorporating long-running transactions in our transactional workloads. Finally, we are interested in creating benchmarks for new areas of graph data management such as streaming and machine learning on graphs.

References

1. D. Abadi et al. The Seattle report on database research. *Commun. ACM*, 2022.
2. H. Almuhimedi et al. Tweets are forever: a large-scale quantitative analysis of deleted tweets. In *CSCW*, 2013.
3. R. Angles et al. The Linked Data Benchmark Council: A graph and RDF industry benchmarking effort. *SIGMOD Rec.*, 2014.
4. R. Angles et al. Foundations of modern query languages for graph databases. *ACM Comput. Surv.*, 2017.
5. R. Angles et al. G-CORE: A core for future graph query languages. In *SIGMOD*, 2018.
6. R. Angles et al. The LDBC Social Network Benchmark. *CoRR*, abs/2001.02299, 2020. http://arxiv.org/abs/2001.02299.
7. R. Angles et al. PG-Keys: Keys for property graphs. In *SIGMOD*, 2021.
8. R. Angles et al. PG-Schema: Schemas for property graphs. In *SIGMOD*. ACM, 2023.
9. S. Beamer et al. The GAP benchmark suite. *CoRR*, abs/1508.03619, 2015.
10. M. Besta et al. Demystifying graph databases: Analysis and taxonomy of data organization, system designs, and graph queries. *CoRR*, abs/1910.09017, 2019.
11. M. Besta et al. Graph processing on FPGAs: Taxonomy, survey, challenges. *CoRR*, abs/1903.06697, 2019.
12. M. Besta et al. Practice of streaming and dynamic graphs: Concepts, models, systems, and parallelism. *CoRR*, abs/1912.12740, 2019.
13. M. Besta et al. GraphMineSuite: Enabling high-performance and programmable graph mining algorithms with set algebra. *Proc. VLDB Endow.*, 2021.
14. P. A. Boncz et al. The Linked Data Benchmark Council project. *Datenbank-Spektrum*, 2013.
15. P. A. Boncz et al. TPC-H analyzed: Hidden messages and lessons learned from an influential benchmark. In *TPCTC*, 2013.
16. A. Bonifati et al. Graph generators: State of the art and open challenges. *ACM Comput. Surv.*, 2020.
17. F. Checconi et al. Breaking the speed and scalability barriers for graph exploration on distributed-memory machines. In *SC*, 2012.

18. A. Deutsch et al. Graph pattern matching in GQL and SQL/PGQ. In *SIGMOD*, 2022.
19. O. Erling et al. The LDBC Social Network Benchmark: Interactive workload. In *SIGMOD*, 2015.
20. N. Francis et al. GPC: A pattern calculus for property graphs. In *PODS*, 2023.
21. A. Green. LDBC Extended GQL Schema (LEX) Work Charter. Work Charter WC-2022-02, LDBC, Nov 2022. https://doi.org/10.54285/ldbc.VSBC2149.
22. A. Gubichev and P. A. Boncz. Parameter curation for benchmark queries. In *TPCTC*, 2014.
23. Y. Guo et al. How well do graph-processing platforms perform? An empirical performance evaluation and analysis. 2014.
24. Z. Guo. Work Charter for FinBench v1.0. Technical report, LDBC, 2022.
25. A. Iosup et al. LDBC Graphalytics: A benchmark for large-scale graph analysis on parallel and distributed platforms. *PVLDB*, 2016.
26. V. Kalavri et al. High-level programming abstractions for distributed graph processing. *IEEE Trans. Knowl. Data Eng.*, 2018.
27. V. Kotsev et al. LDBC Semantic Publishing Benchmark (SPB) - v2.0. Benchmark specification, LDBC, 2014. https://ldbcouncil.org/benchmarks/spb/ldbc-spb-v2.0-specification.pdf.
28. V. Kotsev et al. Benchmarking RDF query engines: The LDBC Semantic Publishing Benchmark. In *BLINK at ISWC*, 2016.
29. LDBC. Articles of Association, 2023. https://ldbcouncil.org/docs/LDBC.Articles.of.Association.ADOPTED.2023-03-30.pdf.
30. LDBC. Byelaws of the Linked Data Benchmark Council v1.4, 2023. https://ldbcouncil.org/docs/LDBC.Byelaws.1.4.ADOPTED.2023-05-02.pdf.
31. L. Libkin et al. A researcher's digest of GQL (invited talk). In *ICDT*, 2023.
32. A. Mhedhbi et al. LSQB: A large-scale subgraph query benchmark. In *GRADES-NDA at SIGMOD*, 2021.
33. M. Pham et al. S3G2: A scalable structure-correlated social graph generator. In *TPCTC*, 2012.
34. M. Poess. New initiatives in the TPC. In *TPCTC*, 2022.
35. S. Sahu et al. The ubiquity of large graphs and surprising challenges of graph processing: Extended survey. *VLDB J.*, 2020.
36. S. Sakr et al. The future is big graphs: A community view on graph processing systems. *Commun. ACM*, 2021.
37. B. Shao et al. Trinity graph engine and its applications. *IEEE Data Eng. Bull.*, 2017.
38. X. Shi et al. Graph processing on GPUs: A survey. *ACM Comput. Surv.*, 2018.
39. G. Szárnyas. LDBC Social Network Benchmark task force work charter. Technical report, LDBC, 2023.
40. G. Szárnyas et al. An early look at the LDBC Social Network Benchmark's Business Intelligence workload. In *GRADES-NDA*, 2018.
41. G. Szárnyas et al. The LDBC Social Network Benchmark: Business Intelligence workload. *PVLDB*, 2022.
42. TPC. TPC Pricing Specification, revision 2.8.0, 2022. http://www.tpc.org/tpc_documents_current_versions/pdf/pricing_v2.8.0.pdf.
43. J. Waudby et al. Supporting dynamic graphs and temporal entity deletions in the LDBC Social Network Benchmark's data generator. In *GRADES-NDA*, 2020.
44. K. Zhao and J. X. Yu. Graph processing in RDBMSs. *IEEE Data Eng. Bull.*, 2017.