时间序列与图数据的结合：现有系统与方法综述 *

Mouna Ammar 莱比锡大学，ScaDS.AI Leipzig，德国
ammar@informatik.uni-leipzig.de

Marvin Hofer 莱比锡大学，ScaDS.AI Leipzig，德国
hofer@informatik.uni-leipzig.de

Erhard Rahm 莱比锡大学，ScaDS.AI Leipzig，德国
rahm@uni-leipzig.de

arXiv:2601.00304v1 [cs.DB] 1 Jan 2026

摘要

我们全面概述了当前结合图与时间序列数据的方法和系统。我们将现有系统划分为四种架构类别，并分析这些系统如何满足不同需求，以及如何通过不同的实现特性以统一方式支持这两种数据类型。本综述旨在帮助读者了解并评估当前的可选方案及其权衡因素，例如跨模型集成程度、成熟度和开放性。

1. 引言

许多领域都需要图数据和时间序列数据，而它们通常由彼此独立的数据管理系统进行管理。图数据应用广泛，例如用于社交网络、微出行和金融领域[69, 87, 90, 70], ，以建模复杂关系，并开展从模式匹配和遍历到图机器学习等分析。图数据既可以由基于属性图模型的原生系统（如 Neo4j 或 JanusGraph）管理，也可以由更通用的数据存储系统管理，例如键值存储或关系数据库管理系统（DBMS） [11]。时间序列是另一种重要的数据结构，用于记录随时间变化的测量值或事件（例如股价、传感器读数），并支持时间相关分析，例如预测和异常检测 [72, 78]。同样，这类数据既可以由原生存储系统（例如时序数据库管理系统 TimescaleDB [85]和 InfluxDB [29]）管理，也可以由更通用的数据存储系统管理。

应用日益需要结合分析图数据和时间序列 [79, 46,60], ，尤其是在机器学习领域。例如，交通网络中的连接（如道路路段）可以扩展为包含时间序列数据，以捕获车辆计数或平均速度，从而识别交通拥堵及其对邻近区域的影响。同样，在物联网（IoT）环境中分析传感器时间序列（例如进行异常检测），也可以通过考虑传感器拓扑以及传感器之间潜在的依赖关系来改进。

将图数据和时间序列数据存储在彼此分离的数据存储系统中，是满足此类应用需求的最低限度方案，因为它们必须管理不同的系统和查询语言。此外，还必须实现所有需要同时处理这两种数据类型的查询和分析任务。因此，我们认为有必要在单一系统中更紧密地整合图数据与时间序列数据。该系统不仅应针对这两种数据类型提供专门支持，还应支持对它们进行新的（混合）操作。[2], 中提出了这种强大混合方案的愿景，但目前仍缺乏完整的实现。

因此，在本研究中，我们着重综述现有的图和时间序列数据融合方法。我们将这些方案归纳为若干架构集成类型，并依据一系列需求和系统特征评估不同系统。我们考察了四种集成架构：**(i)** 单模型**(SM)**系统，使用单一数据模型（原生图模型、时间序列模型或其他数据模型）来管理图和时间序列数据；**(ii)** 扩展单模型**(E-SM)** 系统，通过专用扩展增强主模型，以支持图/时间序列；**(iii) Polyglot** 系统，通过中间件集成多个引擎；以及 **(iv)** 多模型**(MM)** 系统，这类系统没有单一的首选数据模型，而是在一个引擎中支持多个同等重要的模型。本综述旨在帮助读者了解当前可选方案及其优势与局限。本文所展示的方法

*本文已提交至 SIGMOD Record。

也可应用于本文未涵盖的系统。我们公开了用于评估执行跨模型查询的四种架构的代码库，地址为 [1]。

我们作出以下贡献：

1. 我们提出了一组针对图和时间序列数据联合管理与分析的需求，以及相关的系统特征

2. 我们描述了四种结合图与时间序列的架构方法，并阐述了各自的优缺点。

3. 针对所提出的需求和特征，我们对不同类别的20个系统进行了比较评估。

本文其余部分的组织如下。第2节介绍相关综述，第3节概述比较评估所需的需求和系统特征，第4节介绍集成架构。第5节综述具有代表性的系统和研究原型。我们在第6节总结开放性挑战，并在第7节总结本文工作。

2. 相关综述

已有多项综述分别专门探讨时间序列数据库 [35, 34, 9, 38, 31, 88, 95] 和图数据库 [11, 17, 3, 30, 94, 44]。Besta等人 [12]根据数据模型、存储布局和查询执行策略对 50 多个系统进行分类，并指出了处理动态大规模图时面临的根本性设计权衡。这些综述都没有探讨如何同时支持时间序列数据和图数据，而这正是本文的研究内容。针对扩展属性图（[28, 21, 27, 57, 58, 92, 93]）或资源描述框架图数据模型 [98, 97]，已有多种时态图数据库系统的模型和实现。这些方法能够管理和分析图数据随时间的演变，但并未涵盖我们所关注的时间序列。Lu 和 Holubová´a [47]概述了多模型数据库，并对不同系统进行了比较，但没有考虑对时间序列数据的支持，也没有像本文这样纳入具体需求。

3. 比较标准

我们定义了一组关键需求和理想的系统特性，适用于同时支持图数据和时间序列数据的数据管理系统。这本质上是一份对完美解决方案的愿望清单，而现有解决方案只能部分满足这些要求。尽管如此，我们认为，这份清单为比较现有系统并确定当前缺失的功能提供了合理的依据。

3.1 系统需求

需求组1： 数据模型与流处理

D1 单一模型覆盖：应提供支持图数据的数据模型，例如支持带标签的数据节点和边的属性图，以及时间序列数据。

D2 统一混合数据模型：理想情况下，应采用统一数据模型，无缝集成作为一等公民的图和时间序列数据，并支持两者的组合表示（例如图节点的时间序列属性或互相关联的时间序列）。

D3 时间演化：需要支持当前数据和历史数据，以捕捉图和时间序列随时间的演变。

D4 流式数据支持：系统应支持对无界高速数据流进行摄取、处理和查询，尤其是持续到达的时间序列和图更新，并实现低延迟处理、妥善处理乱序或迟到事件，以及与存储状态集成 [80, 89]

需求组 2： 查询与转换

Q1 跨模型查询引擎：系统应能够查询图数据和时间序列数据，并在单个查询中混合使用两者。用户必须能够跨这两种模型进行连接、筛选、聚合和模式匹配，而无需手动协调多个查询引擎。除了支持混合查询（例如时间感知分组）之外，还应支持提取算子（例如从图数据中提取时间序列）以及跨越两种数据模型且具有一致语义的混合更新。理想情况下，此功能应通过统一的查询抽象或查询语言提供 [26, 41]

Q2 双向模型转换（互操作性和灵活性）：需要支持在单一数据模型之间（从图到时间序列，反之亦然）以及集成混合模型与单模型（仅图或仅时间序列）之间转换数据

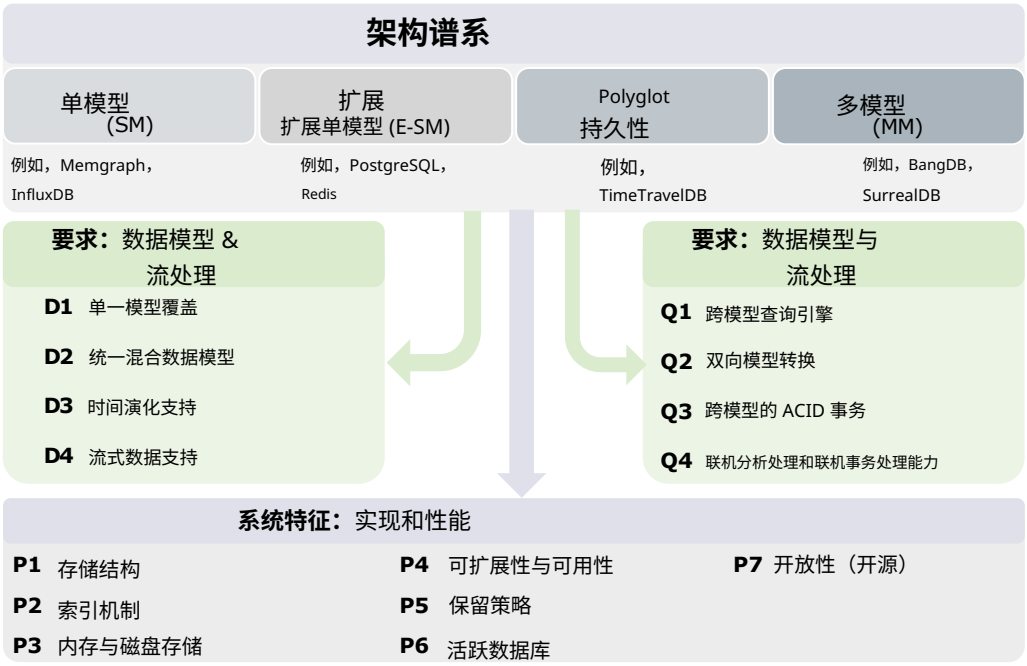


图1：Ove　　用于结合图和时间序列数据的系统比较方法概述

格式，以及单模型数据和混合数据的导入与导出。

Q3 跨数据类型的 ACID 事务：在同一事务中跨图实体和时间序列实体进行原子、一致的更新。

Q4 OLTP 和 OLAP 能力：同时支持事务型 OLTP 工作负载（点查询、更新、模式匹配、快照查询）和分析型 OLAP 工作负载（聚合、全局图和时间序列算法）且无需单独的 ETL [8]或系统。这种混合 HTAP 能力 [45] 很有优势，因为混合图和时间序列应用本质上结合了事务操作（例如图遍历）与分析运算符（例如时间序列聚合）。

3.2 系统特征

所提出的数据模型和查询需求难以满足，需要采用先进的实现技术，才能在大型数据集上实现快速处理和可扩展性。鉴于有多种系统方案可以部分满足所提出的需求，我们不规定实现方面的要求。相反，我们指出与实现和性能相关、会影响这些需求满足程度的系统特征。我们也会在对当前实现进行比较评估时考虑这些方面。

实现与性能特征

P1 图和时间序列数据的存储结构，例如邻接矩阵、B^+-树、日志结构合并树（LSM 树）、列式存储。

P2 图-时间序列索引机制：支持对图和时间序列数据进行索引，以及复合索引，例如联合捕获图实体及其时间序列属性。

P3 内存与磁盘存储：支持内存中存储和处理（例如，用于快速分析），并支持外部存储以实现持久性和数据持久性。

P4 可扩展性与可用性机制：通过数据分布（例如，分片）支持水平扩展，以处理不断增长的图数据和时间序列数据，同时保持查询性能，并通过复制实现容错。

P5 保留策略。内置的历史数据生命周期规则（例如，生存时间、降采样）控制图数据和时间序列数据的保留时长。

P6 活跃数据库支持对变化做出响应，例如 ECArules（事件-条件-动作）、触发器或持续/连续查询 [49]。

P7 开放性。商业版与开源许可证。根据 OSI 批准的许可证 [63]（例如 Apache 2.0、GPL、PostgreSQL 许可证、MIT、BSD）发布的系统被视为开源系统。采用 SSPL/BSL 等许可证的其他系统，无论是否提供源代码，均被视为闭源系统。

图1总结了我们比较方法中引入的标准和特征。

4. 集成架构

我们将用于组合图和时间序列数据的现有系统划分为四种架构方法：单模型（*SM*）、扩展单模型（*E-SM*）、多语言持久化系统和多模型数据库（*MMDB*）[1]。图2展示了这些解决方案的大致架构。表1总结了这四种架构的一些优势和挑战，并列出了示例实现。本节将更一般地讨论这些架构，而下一节将对四种集成架构中的选定系统进行比较评估。在我们的代码仓库 [1], 中，我们提供了多种数据库管理系统架构的解决方案，用于在移动出行用例（共享单车）中存储和分析图和时间序列数据。

4.1 单模型系统（SM）

这种方法使用单模型数据库管理系统来存储其他类型的数据。可以区分出这种方法的三种变体：

• 存储时间序列数据的图数据库系统。这种方法原生支持图数据处理，例如支持属性图模型（如 Neo4j [62], 、JanusGraph [33], 、TigerGraph [84], 、Memgraph [55]）或时序图方法（ [56,53] ），同时也存储和管理时间序列数据，例如将其存储在节点或边的属性中。

•存储图数据的时间序列数据库管理系统。这是对称的方法：原生时间序列数据库管理系统（如 InfluxDB [29] ）也致力于存储图数据，例如将节点和边视为带时间戳事件。

•其他用于存储图和时间序列数据的数据库系统。 在这种方法中，使用不原生支持图数据或时间序列数据的数据库管理系统（例如关系型或 NoSQL 数据库管理系统）来存储和处理这些类型的数据。在关系型数据库管理系统（例如 MySQL [61]）中，可以将这些数据存储在不同的表中，并使用 SQL 分析图和时间序列数据，但这种做法通常在性能和分析方面存在显著局限。对象关系数据库管理系统通过针对非关系型数据的专用扩展解决了其中一些局限，从而实现扩展单模型架构。在 MongoDB [59]）等文档存储中，每个时间序列点都可以建模为包含时间戳的文档。随后可以将这些文档链接（通过引用/外键）或嵌入，以形成图结构，详见我们在此存储库中提供的解决方案 [1]。键值存储（例如 RockDB [76]）可以将图邻接表编码为以节点标识符为键的值，并将时间序列编码为有序集合或列表。

单模型系统通常最难满足我们提出的要求，因为它们至多只对两种数据类型中的一种提供专门支持，因此既缺乏对另一种数据类型的高级查询支持，也缺乏跨模型查询支持。即使是采用 T-GQL [21] 和 T-PGQL [77]等查询语言的时序图数据库管理系统，也不支持时间序列分析。同样，非图数据库管理系统既不专门支持复杂的图查询（例如 k跳模式匹配），也不支持跨模型查询。

4.2 扩展单模型（E-SM）

采用 E-SM 架构的系统基于成熟的核心数据模型（例如关系模型），并对其进行扩展（例如通过插件或模块），以支持其他数据类型，包括 JSON、图和时间序列数据。这使系统具备多模型能力，但与架构类型 MM 不同，各模型的重要性并不相同。此外，新增的数据类型通常会映射到原始模型的存储结构中；也就是说，数据最终仍按照核心模型的范式存储，这可能会限制可执行的查询和系统性能。

关系模型的对象关系扩展明确支持 E-SM 方法，可通过用户定义数据类型和过程来支持非关系型数据等。SQL 标准已为文档数据（XML、JSON）和图数据定义了此类扩展[22], 但尚未为时间序列数据定义（尽管 SQL 窗口谓词有助于时间序列分析）。扩展数据类型的运算符可用于 SQL 查询，从而支持跨模型查询和数据类型转换。现有数据库管理系统（如 PostgreSQL [71] 和 Oracle）提供了支持图数据和时间序列数据的扩展，从而扩展了 SQL 标准——

[1]当需要结合其他类型的数据时，这些架构同样适用。

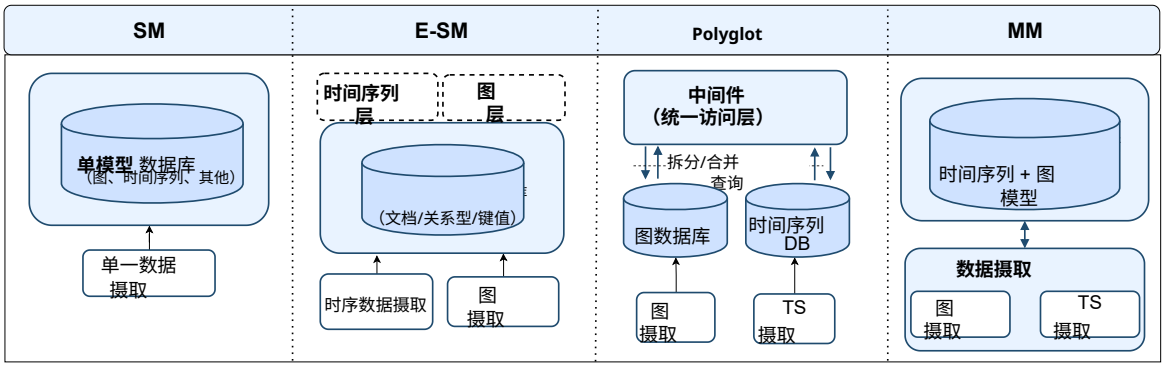


图 2：集成架构示意图：SM、E-SM、Polyglot 和 MM

准。

键值存储 Redis [73]提供了支持图数据和时间序列数据的扩展。对于宽列数据存储 Cassandra [15, 16], , 存储这两种数据类型有多种选择。Cassandra 既可用作存储后端，在其上构建 JanusGraph（用于图数据） [32]和 KairosDB [37]（用于时间序列数据）；也可使用 DataStax Graph [20], ——这是商业版 Cassandra 发行版的原生扩展，可将图数据和时间序列数据直接存储在 Cassandra 中，而 Cassandra 已提供经过优化的存储和一定的分析功能。示例查询见 [1]。讨论表明，E-SM 类别中存在截然不同的实现，即便是在关系型数据库管理系统之间也是如此。E-SM 方法能够支持跨模式等关键需求

1

查询，但表达能力和性能取决于基于核心数据模型所选择的映射和实现。

4.3 Polyglot 持久性系统

多语言持久化 [23, 91] 是一种架构概念，它集成多种数据类型和技术，通过利用不同数据库管理系统的优势，针对特定的数据模型、查询或可扩展性需求进行优化，从而支持具有多样化数据处理需求的应用程序 [24]。多语言持久化背后的核心动机是，没有单一数据库能够很好地适配所有数据类型和工作负载 [39]。就我们的用途而言，这种方法可用于将图数据库管理系统（例如 Neo4J）与时间序列数据库管理系统（例如 InfluxDB）集成。

缺点在于，所有跨系统功能都必须由原生系统之上的额外中间件层提供（图 2（Polyglot））。对于查询处理，跨模型查询要么由应用程序实现，要么由中间件支持特定的跨模型查询功能，甚至基于合适的跨模型模式提供混合查询语言。在这种情况下，中间件必须将跨模型查询拆分为单模型子查询，并合并这些子查询的结果。显然，这极具挑战性，因此难以在可接受的性能水平下支持复杂的跨模型查询 [39,40, 42]

据我们所知，目前没有商业版的 Polyglot 系统能够结合图数据库管理系统和时间序列数据库管理系统。因此，我们纳入了莱比锡大学开发的研究原型 TimeTravelDB。该系统旨在以原生方式存储图数据和时间序列数据，并配有一种新的查询语言——Time-Travel Query Language（TTQL），用于管理模型及其组合。

此类 Polyglot 方法的查询示例见 [1]。

4.4 多模型数据库（MM）

多模型数据库管理系统（MMDB） [96] 提供了一个统一平台，可在单一系统中管理多种模型和数据类型，例如文档、图、关系表和时间序列。多语言持久化依赖于针对不同数据模型的独立专用数据存储系统，而 MMDB 则将这些模型集成起来，以更灵活且一致的方式进行数据管理，并减少数据同步问题 [47]。与 E-SM 系统不同，所有受支持的数据模型都被平等地视为“一等公民”，从而确保所有模型具备一致的性能和功能 [26, 47]。MMDB系统使用单一引擎，以原生方式处理所有受支持的数据模型，而不依赖额外的层、模块、插件或中间件 [5, 26]。虽然可以针对不同模型支持多种查询语言，但理想情况下应有一种统一的查询语言，能够无缝管理和查询所有

表 1：图 +时间序列系统架构概览

类别	主要优势	主要挑战	示例系统
单模型	利用成熟的数据库管理系统。跨模型实现 ACID，因为一个引擎。	最多针对以下模型之一进行了优化：模型（图或时间序列），不支持混合查询	Neo4j [62], InfluxDB [29], Memgraph [55], MongoDB [59], Aion [83], ClockG [53]
扩展单一模型	例如单模型加上专门的对图/时间序列数据的支持数据	扩展必须与 core 模型保持一致; 潜在 性能开销	PostgreSQL [71], Redis [73], Couchbase [18], MariaDB [50], Cassandra [15],Oracle [66]
多语言持久化	原生支持 两者数据模型。	high 集成 复杂性, 两者数据查询延迟高 类型的数据 ，不支持跨模型事务，也没有成熟的实现—— 实现	TimeTravelDB 原型类型 [86]
MMDB	单一引擎支持多种模型，跨越 ACID 模型	复杂 to 实现; re- 当前实现尚不成熟 实现；图/时间序列 处理 优化程度较低 相比原生数据库管理系统	ArangoDB [6], OrientDB [68], ArcadeDB [7], BangDB [10], SurrealDB [82]

受支持的数据模型。这样无需在不同查询语言之间切换，即可支持复杂的多模型查询 [47]。例如，ArcadeDB 除了提供类 SQL 功能外，还支持用于图查询的 Cypher 和 Gremlin。相比之下，ArangoDB 提供了一种名为 AQL 的复杂统一查询语言 [54]。

尽管多模型数据库系统前景可期，但要实现功能完善且性能出色的系统并非易事，这取决于系统整合不同模型数据的能力。由于缺乏整合和优化各种数据模型的既定标准和最佳实践，这一问题也更加复杂 [47,41, 48]。我们将看到，当前的多模型数据库在满足我们的需求方面仍受到诸多限制。正如 [2], 所设想的那样，先进的多模型数据库可以支持图和时间序列数据的紧密整合，并采用统一的查询语言，但这一方案尚未实现。

5. 比较评估

在评估中，我们根据所提出的需求（表 2）和实现特征（表 3），比较了四种集成架构中的 15 个商业版数据管理系统和五个研究原型[2]。虽然无法涵盖所有相关系统，但我们找到了四类架构中具有代表性的解决方案，它们至少能部分满足同时处理图数据和时间序列数据的要求。下文将分别讨论这四种架构的评估结果。

在表 2 中，我们将具有原生流处理能力的系统归类为完全支持流处理 （例如，支持实时摄取和分析管道）；将依赖外部流处理引擎并通过连接器接入的系统归类为部分支持流处理；将除批处理操作外不具备明确流处理能力的系统归类为不支持流处理。对于双向模型转换，我们将支持两个方向（从图/时间序列 → 混合数据转换而来，并可转换回去）的系统归类为完全支持；对于仅支持一个方向的系统（例如，可以导入混合数据但不能导出，或只能导出单一模型），则归类为部分支持。在表 3 中，完全支持 *Graph-TS 索引* 要求对两种数据类型都建立索引，并支持复合索引。相反，如果只对其中一种数据类型建立索引，或缺少复合 Graph-TS 索引，则表示部分支持。若干工业系统提供多个版本（例如社区版和企业版）。在我们的比较中，如果某系统提供 OSI 许可的社区版，且表中报告的功能无需付费许可证即可在该社区版中使用，我们就将其视为开源。如果所报告的功能依赖仅限企业版功能，则在我们的评估中，该系

[2]出于篇幅考虑，我们将对联机事务处理和联机分析处理等部分需求移至表 3

统不被视为开源。例如，Neo4j 同时提供社区版和企业版。尽管社区版支持大多数图查询操作，但只有企业版支持可扩展性。因此，我们认为 Neo4j 是开源的，但不具备可扩展性。相比之下，Oracle 数据库采用商业许可证发布。尽管存在有限的免费版本，但它们并非开源。因此，Oracle 被视为闭源系统，所报告的功能对应于其企业版产品。

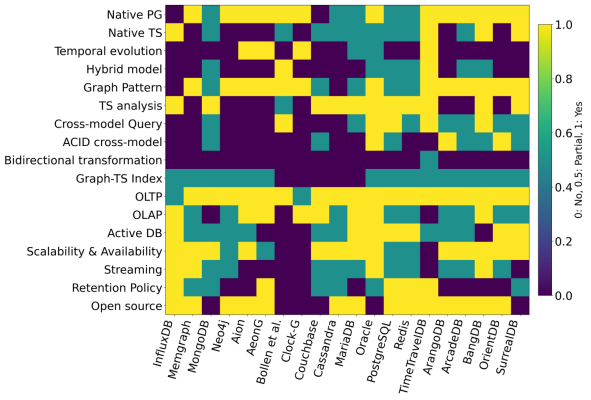


图 3：各系统对需求和特征的覆盖情况

5.1 单模型系统结果

对于**单模型**类型的系统，我们纳入了原生属性图数据库管理系统（Neo4j、Memgraph）、原生时间序列数据库管理系统（InfluxDB）以及文档存储（MongoDB）。此外，我们还考察了三个针对时序图数据的研究原型（Aion、AeonG、Clock-G），以及一个支持时间序列的属性图原型（Bollen 等人 [13]）。不出所料，如表 2 所示，没有任何单模型系统能全面支持属性图（PG）和时间序列（TS）。这些系统大多不支持跨模型查询和 ACID 事务，因此无法满足基本需求。

所考察的图系统 Neo4J 和 Memgraph 均不内置时间序列支持。可以通过属性值对时间序列建模。不过，如果只使用简单的属性值，就需要为长度为 n 的时间序列设置大量 $2n$ 属性值（每个事件对应一个属性值，分别用于存储时间戳和值）。在 Neo4j 中，也可以将整个时间序列表示为由事件节点串成的图，或使用 GraphAware Timetree 插件 [25]。这种编码方式支持混合查询（受时态谓词或聚合约束的模式匹配），但代价是查询语句冗长复杂，因而可用性较差。Bollen 等人的研究原型通过其查询语言 GQL-TS 支持时间序列和跨模型查询，并将时间序列表示为由节点组成的链表 [13]。Neo4J、Memgraph 和原型 [13]侧重于联机事务处理，但也支持类似联机分析处理的查询。内存系统 Memgraph 支持流处理（图）数据。

时序图原型 保留历史图数据，因此能够支持演化分析和快照分析，但不原生支持时间序列分析或混合查询。查询语言通常基于 Neo4J 的 Cypher，并扩展了时间谓词，例如用于获取特定时间点的快照。两者均支持联机事务处理和联机分析处理查询。原型 AeonG[3]采用 BSL 1.1 许可证，这意味着它是闭源的，而 Aion[4] 是开源的。Clock-G 和 Bollen 等人的方法均没有可用的代码实现。

时间序列数据库管理系统 InfluxDB 没有内置的图数据存储与分析支持，也不支持跨模型查询。时间序列事件（测量值）可以通过元数据进行丰富，以便存储节点状态或节点间关系。因此，用户可以通过一些方式表示和分析随时间演化的图数据，但对于结构图、复杂图遍历和图分析的支持则相对有限。InfluxDB 是一款面向时间序列数据的联机分析处理系统，并支持流数据。

其他关系型和 NoSQL 单模型系统均未原生支持图数据和时间序列数据。不过，仍可利用底层数据模型和查询语言来表示和分析这两种数据。在所考察的文档数据库管理系统 MongoDB 中，图节点和边可以表示为类型化集合中的文档，并提供了一个用于简单图分析的运算符$graphLookup 。时间序列数据可以存储在 MongoDB 5.0 引入的时间序列集合中，并使用聚合运算符进行分析。该系统支持时间序列数据压缩，也支持用于数据保留的 TTL（生存时间）策略。因此，虽然可以实现对这两种数据类型和混合查询的一些基本支持，但缺乏高级或优化的功能。

实现特征（表 3）差异很大，侧重于磁盘存储，并部分支持分片和复制。

[3]https://github.com/hououou/AeonG[4] https://github.com/aeon-toolkit/aeon

表 2：各系统的建模与查询能力。图例： ✓=支持； ○=通过变通方案或扩展/插件支持； ×= 不支持。RP =研究原型。KV = 键值。符号： ✓ = 原生/一等支持； ○ = 通过扩展或变通方案支持； × = 不支持。

Architecture	System	Primary Model	Native PG	Native TS	Temporal evolution	Hybrid model	Streaming	Query Language	Graph Pattern	TS analysis	Cross-model Query	Bidirectional transformation	Acid cross-model
SM	InfluxDB	时间序列	×	✓	×	×	✓	InfluxQL	×	聚合、分组、连接等	×	×	×
	Memgraph	属性图	✓	×	×	×	✓	OpenCypher	✓	×	×	×	×
	MongoDB	文档	○	○	×	○	○	MQL	○	`$bucket`, `$window` 等，位于聚合管道中	○	×	○
	Neo4j	属性图	✓	×	×	×	○	Cypher	✓	×	×	×	×
SM (RP)	Aion	时序图	✓	×	✓	×	×	时态 Cypher	✓	×	×	×	×
	AeonG	时序图	✓	×	✓	×	×	时态 Cypher	✓	×	×	×	×
	Bollen 等人	带有属性图的时间序列	✓	○	×	✓	×	GQL-TS	✓	窗口、聚合	✓	×	×
	Clock-G	时序图	✓	×	✓	×	×	T-Cypher	✓	和时间序列模式 ×	×	×	×
E-SM	Couchbase	文档	×	○	×	×	○	SQL++	○	`_TIMESERIES()` 聚合函数	×	×	○
	Cassandra*	宽列存储	○	○	×	×	○	CQL	×	`$merge`, `$out`, `$group` 等。	×	×	×
	MariaDB	关系型	○	○	○	×	○	SQL	○	聚合，统计-窗口上的统计函数 dow	○	×	×
	Oracle	关系型	✓	○	○	○	✓	SQL、PGQL	✓	机器学习、时间序列回归、预测统计学	✓	×	✓
	PostgreSQL*	关系型	○	○	×	○	○	SQL	○	Timescale：分组，时间筛选器、聚合	✓	×	○
	Redis*	键值	○	○	×	○	○	命令 / Cypher（图）	○	RedisTS：TS.GET, TS.RANGE 等。	○	×	×
Polyglot (RP)	TimeTravelDB	TTPGM (1)	✓	✓	✓	✓	×	TTQL（Ex-扩展 Cypher）	✓	原生（FROM、TO, SHALLOW）、聚合	✓	○	×
MM	ArangoDB	文档 +图 + KV	✓	×	×	×	○	AQL	✓	×	○	×	✓
	ArcadeDB	文档 + 图 + 时间序列 +键值	✓	○	×	○	○	类 SQL	✓	×	○	×	○
	BangDB	文档 +图 + 时间序列	✓	✓	×	○	✓	C++, 类 SQL	✓	连接、筛选、聚合, 滑动窗口等。	✓	×	○
	OrientDB	文档 + 图 + KV	✓	×	×	×	○	类 SQL	✓	×	○	×	✓
	SurrealDB	文档 + 图 + 时间序列（+其他）	✓	✓	×	×	×	SurrealQL	✓	聚合, 分组等。	○	×	○

注： 星号（*）表示基于插件的功能。RedisGraph 将于 **2025年1月31日**结束支持。（1）时间旅行属性图模型

表 3：实现特性。图例： ✓=原生/一等支持。 ○=部分支持或通过扩展/插件实现。×= 不支持。CQ = 连续查询。CQN =连续查询通知。可扩展性与可用性代码：S = 分片；R = 复制；SR = 分片 + 复制。KV=键值

Architecture	System	Storage Structure	Graph-TS Index	Disk/In-Memory	OLTP	OLAP	Active DB	Scalability & Availability	Retention Policy	License
SM	InfluxDB	时间结构化合并树 (TSM)	○	Disk	○	✓	✓ CQ	R (1)	✓	MIT（开源）
	Memgraph	内存中（跳过）表）	○	内存中	✓	○	○ 触发器	R	○	BSL 1.1（源代码可用）
	MongoDB	B 树（WiredTiger）	○	Disk	✓	×	○ 触发器	SR	○	SSPL（源代码可用）
	Neo4j	固定大小记录存储	○	Disk	✓	○	○ 触发器	× (1)	×	GPL v3（社群版）
SM (RP)	Aion	TimeStore + LineageStore	○	Both	✓	✓	○ 事件监听器	SR	×	开源
	AeonG	当前（多版本并发控制）+ 历史的 KV 存储 (anchor+delta)	○	Disk	✓	✓	×	S（4）	✓	RP
	Bollen 等人	固定大小记录存储（Neo4j）	×	Disk	✓	×	×	×	×	RP
	Clock-G	δ-复制+日志（Cassandra 后端）	×	Disk	○	✓	×	×	×	RP
E-SM	Couchbase	B 树（Couchstore/-Magma）	×	Both	✓	✓	○事件服务	SR	○	BSL 1.1（源代码可用）
	Cassandra*	日志结构合并树（SSTables）	×	Disk	✓	○	○Java 触发器	SR	○	Apache 2.0
	MariaDB	RocksDB（2）存储（Inn-	×	Both	✓	✓	✓ 触发器	SR	×	GPL v2
	Oracle	行存储表 (5)	○	Both	✓	✓	✓ 触发器/CQN	SR	○	商业版
	PostgreSQL*	行存储（B 树 / 堆）	○	Disk	✓	○	✓ 触发器	S (3)	✓	PostgreSQL License
	Redis*	内存中（哈希 + 跳表）	○	内存中	✓	○	✓ 触发器	S	✓	BSD 三条款许可证
Polyglot (RP)	TimeTravelDB	记录存储 + 列式	○	Disk	✓	×	○ 触发器 + 连续查询	S	✓	开源
MM	ArangoDB	LSM 树 (RocksDB)	○	Disk	✓	○	○Spring 事件	SR	×	BSL 1.1
	ArcadeDB	LSM 树	○	Disk	✓	○	○ Java 监听器	R	×	Apache 2.0
	BangDB	B 树 + 哈希	○	Both	✓	✓	×	SR	×	BSD 三条款许可证
	OrientDB	分页本地存储	○	Both	✓	○	✓ 触发器	SR	×	Apache 2.0（社区版）
	SurrealDB	LSM tree (RocksDB)	○	Both	✓	○	✓ 实时查询	SR	○	munity) BSL 1.1（源代码可用）

注释： (1) 企业版 = HR。(2) 提供用于时间序列的列存储引擎。(3) 使用 Citus 的 H。(4) 分布式变体 AeonG-D。(5) 提供内存列存储

5.2 扩展单模型（E-SM）结果

对于 E-SM 类型，我们考察了三个关系型数据库管理系统（Oracle、PostgreSQL、MariaDB）和三个 NoSQL 系统（Redis、Couchbase、Cassandra）。这些系统主要通过扩展支持图和时间序列数据（要求 D1），部分支持跨模型查询（Q1），也部分支持不同数据类型之间的 ACID 特性（Q3），但仍缺乏统一的混合模型（D2），无法在单一数据结构中整合这两种数据类型。系统支持流数据（D4），但对时间演化（D3）的支持有限。在 E-SM 系统中，核心引擎提供的可扩展性保证不一定适用于图和时间序列扩展，反之亦然。因此，在本综述中，只有当核心引擎及其扩展都具备可扩展性时，我们才认为系统完全支持可扩展性。

关系型数据库管理系统 Oracle 对属性图以及对存储在关系表中的时间序列数据进行分析提供了强有力的支持。它还支持管理图数据和时间序列数据的时间演化，并可进行历史查询（D3）。Oracle Database 23ai 的图功能包括一个丰富的图算法库（超过 80 [64] 个）。时间序列分析由 SQL 函数 [65]（例如 TIME_BUCKET、AGO、TODATE）以及用于时间序列预测的 Oracle Machine Learning for SQL [67] 提供支持。因此，Oracle 可同时支持联机事务处理和联机分析处理工作负载。借助 Oracle Cloud Infrastructure 和兼容 Kafka 的事件流服务 [36]，Oracle 也可支持流处理；该服务可用于将图和时间序列数据实时摄取到 Oracle 中。

PostgreSQL 是一种关系数据库，通过 TimescaleDB [85]为时间序列数据提供扩展，并通过 Apache AGE [4]为图数据提供扩展。若 Timescale 和 AGE 表由同一个 Postgres 实例管理，则还支持跨模型查询（Q1）以及部分 跨模型 ACID（Q3）。不过，扩展边界和异构存储路径往往会限制混合支持的深度。PostgreSQL 仅通过其 TimescaleDB 扩展提供部分数据流支持（D4），可实现实时数据摄取和查询。

关系型数据库管理系统 MariaDB 支持在同一服务器上使用多种存储引擎，包括用于处理类似图结构复杂数据的引擎*OQGRAPH* [51], ，以及支持有效时间和事务时间的双时态表 [14]。它没有专为时间序列数据设计的存储适配方案，但可以使用针对分析查询优化的 *ColumnStore* 引擎 [52]处理此类数据。对跨模型查询的支持较弱。查询处理需要类似 ETL 的步骤，以便在不同引擎之间传递数据：先从一个引擎导出结果，再将其导入另一个引擎，以执行组合查询。

Redis 是一种开源的内存键值存储，包含用于时间序列数据的扩展（RedisTimeSeries [75]）和用于图数据的扩展（RedisGraph [74],，后者使用 OpenCypher 查询语言），不过 RedisGraph 已不再受支持。每个扩展都为其相应的数据模型提供原生查询能力，但不内置跨模型查询支持。Redis 通过 Redis Streams 扩展 [81]提供部分流处理支持。数据保留支持（P5）仅由 RedisTimeSeries（P5）提供。

分布式文档存储 Couchbase 不原生支持图数据和时间序列数据。不过，它可以将这些数据类型存储在 JSON 文档中，并针对时间序列数据提供优化和专门的分析功能 [19]。其类 SQL 查询语言 SQL++也支持递归查询，例如可达性查询和图遍历。它不内置跨模型查询支持，因此用户必须自行实现图文档与时间序列文档的组合。流处理仅通过连接器接入外部流处理系统来支持。

纳入 Cassandra 的主要原因是，作为一种宽列存储，它具有很高的可扩展性，这得益于其去中心化存储架构 [43], ，以及它被用作时间序列数据库（如 KairosDB [37], ）和图数据库（如 JanusGraph）的后端存储。两者都可以在同一个 Cassandra 引擎中使用。

5.3 多语言持久化

由于缺乏合适的商业版系统，我们将研究原型 TimeTravelDB (TTDB) [2]作为多语言持久化方案进行考察。它结合了图数据库管理系统 Neo4j 和 PostgresSQL 的时间序列扩展 TimescaleDB。它在中间件层支持一种称为 TTPGM（时空属性图模型）的扩展时态属性图数据模型（D1），其中图节点可以包含时间序列属性。通用唯一标识符（UUID）用于引用图元素中的时间序列数据。这就是“混合链接”，用于维护两个相互独立的数据库之间的一致性（D2）。

对于查询处理，一种名为 TTQL（TimeTravel 查询语言）的扩展 Cypher 查询语言支持对时间序列数据 [86]（Q1）执行时态范围查询和聚合。查询处理还

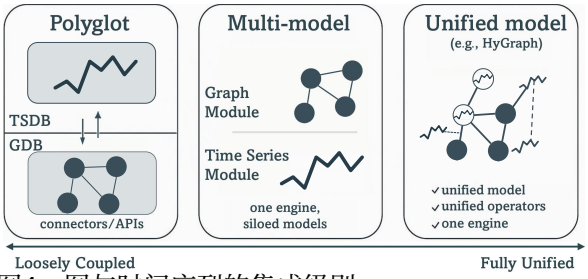


图4：图与时间序列的集成级别

包括将用户查询拆分为 Cypher 组件（图）和 SQL 组件（时间序列），分别执行这两个组件，并合并结果。TimeTravelDB 从其底层系统继承了联机事务处理能力，但在联机分析处理方面仍较为薄弱，仅提供带有时间序列约束的模式匹配。TT DB 原型通过为每个图元素设置有效性属性以及支持时态范围查询（关键字 FROM、TO）（D3）来实现时态性。

5.4 多模型系统（MM）

我们考察了五种多模型实现（ArangoDB、OrientDB、ArcadeDB、BangDB、SurrealDB）。它们都原生支持文档和图，但并非都支持时间序列，因为时间序列也可以存储在其他数据结构中，例如有序列表和键值映射。$BangDB$ 和$SurrealDB$ 提供原生时间序列支持，$ArcadeDB$也计划提供该支持，而$ArangoDB$|$OrientDB$则依靠通用文档集合来管理时间序列。它们通常不提供统一混合数据模型（D2）和时态演化（D3），但支持跨模型查询（Q1）和跨模型 ACID 事务（Q3）。ArangoDB、OrientDB 和 ArcadeDB 为大型图提供图分析（例如 PageRank、单源最短路径）和聚合功能。 [1]中给出了一个使用 ArcadeDB 的跨模型查询示例。

6. 尚待解决的挑战

图 3 中的热力图总结了受调查系统在多大程度上符合我们的建模、查询和实现标准。总体而言，没有任何系统满足所有要求。覆盖面最广的是 E-SM 系统 Oracle（约符合 73% 的既定标准），其次是多模型系统 BangDB（约 67%）以及带有扩展的 PostgreSQL（62%）。Polyglot 研究原型 TimeTravelDB 对这些标准的支持也很出色（约 70%），但它的成熟度当然不及前述系统。

Oracle 提供商业版解决方案；BangDB 提供社区版；而 PostgreSQL 可免费使用。

尽管现有系统种类繁多，并且支持图和时间序列数据，但在提出的需求和实现功能方面仍存在**重大缺口**，因此尚未真正实现这两种数据模型的统一。除了两个研究原型支持图中的时间序列属性之外，目前还不支持**统一混合数据模型**（需求 D2），无法将图和时间序列数据整合到一个复合数据模型中。因此，真正的混合查询语言仍然缺失，尚未完全支持**双向数据模型转换**（Q2）、混合数据的导入/导出，以及复合图/时间序列索引（P2）等实现功能。例如，考虑这样一种查询：需要返回满足特定时间序列约束的图路径，例如实体或链接上出现暂时性的尖峰模式。使用现有解决方案时，必须先手动将时间序列与其实体关联起来，然后通过图查询获取所有可能的路径，最后逐一检查每条路径在时间序列数据中是否满足条件。这种解决方案将耗费过多的时间和计算资源；而在统一环境中，则可以同时检查时间序列条件，并直接过滤掉无效路径。

如图4所示，不同的集成架构及其实现系统，在图数据与时间序列数据的统一程度上各有差异。在一端，多语言持久化系统松散地整合了基于两种（或更多种）数据模型的系统，但需要复杂的中间件来组合数据。多模型系统和 E-SM 系统则通过外部连接器或 API，在单个数据库管理系统及其扩展（例如，图数据库管理系统和时间序列数据库管理系统）中实现更紧密的集成，但仍将每种模型分开处理，既没有统一的数据模型，跨模型表达能力也有限。在另一端，则是构建真正统一的数据模型这一愿景：图数据和时间序列数据都作为一等公民，纳入具有共享语义、索引和查询能力的统一表示中。HyGraph 项目 [2]旨在提供这样的解决方案。

7. 结论

我们调查了采用四种架构方案组合图数据与时间序列数据的20个系统，并根据一组关键需求和实现特性对它们进行了比较评估，

该方法也可应用于其他未纳入本文的系统。定量评估有助于读者了解当前的发展现状，并为特定用例或研究方向找到最有前景的系统。

虽然目前还没有能够满足所有要求的完整解决方案，但我们发现，一些商业版数据库管理系统已提供了合理的支持，例如在单个查询中同时评估两种数据类型。这尤其适用于 Oracle 和 PostgreSQL 等 E-SM系统，以及 BangDB 等多模型方案。目前缺少的是对统一混合数据模型的支持，该模型能够更紧密地集成这两种数据类型，并支持混合算子和数据转换。作为未来工作，我们计划设计一个用于混合图与时间序列处理的基准测试，以便对选定系统进行定量评估。

8. 参考文献

[1] M. Ammar. 用于复现各种架构的跨模型查询示例的代码库，2025年。创建于：2025年8月。[2] M. Ammar、C. Rost、R. Tommasini、S. Agarwal、A. Bonifati、P. Selmer、E. Kharlamov 和 E. Rahm. 迈向混合图：统一属性图与时间序列。载于第*28*届数据库技术拓展国际会议，2025年。[3] S. Anuyah、V. Bolade 和 O. Agbaakin. 理解图数据库：全面的教程与综述。*arXiv* 预印本 *arXiv:2411.09999*，2024年。[4] apache age. 面向 PostgreSQL 的 Apache AGE™ 图数据库，2024年。[5] ArangoDB. ArangoDB 官方文档，2023年。[6] arangodb. ArangoDB 多模型数据库，2025年。[7] arcadedb. 新一代多模型数据库，支持图、键值、文档、搜索引擎、向量和时间序列，2021年。[8] M. Attobrah. ETL 流水线。载于《*Essential Data Analytics, Data Science, and AI: A Practical Guide for a Data-Driven World*》，第27–46页。Springer，2024年。[9] A. Bader、O. Kopp 和 M. Falkenthal. 开源时间序列数据库综述与比较。载于数据库系统 *f¨urBusiness,Technologieund Web(BTW2017)-Workshopband*，第249–268页。Gesellschaft f¨ur Informatik eV，2017年。[10] bangdb。Bangdb，面向现代应用的多模态数据库，2025。[11] M. Besta、R. Gerstenberger、E. Peter、M. Fischer、M. Podstawski、C. Barthels、G. Alonso 和 T. Hoefler。揭开图数据库的面纱：数据组织、系统设计和图查询的分析与分类。*ACM*计算调查，56(2):1–40，2023。[12] M. Besta、R. Gerstenberger、E. Peter、M. Fischer、M. Podstawski、C. Barthels、G. Alonso 和 T. Hoefler。揭开图数据库的面纱：数据组织、系统设计和图查询的分析与分类。*ACM Comput. Surv.*，56(2):31:1–31:40，2024。[13] E. Bollen、R. Hendrix 和 B. Kuijpers。使用图数据库管理配备传感器的交通网络数据。《地球科学仪器、方法与数据系统》，13(2):353–371，2024。[14] A. Bui。双时态建模：实现、性能与应用案例。*LU-CS-EX*，2022。[15] cassandra。Cassandra，开源 NoSQL 数据库，2025。[16] A. Cassandra。Planet Cassandra，2025。[17] M. E. Coimbra、L. Svit´akov´a、A. P. Francisco 和 L. Veiga。综述：图数据库。*arXiv* 预印本 *arXiv:2505.24758*，2025。[18] couchbase。Couchbase，分布式 NoSQL 开发者数据库平台，2025。[19] Couchbase。介绍 Couchbase 时间序列，2025。[20] Datastax。作为 Cassandra 扩展的 Datastax Graph（商业版），2025。[21] A. Debrouvier、E. Parodi、M. Perazzo、V. Soliani 和 A. Vaisman。时态图数据库的模型与查询语言。《*VLDB*期刊》，30(5):825–858，2021。[22] A. Deutsch、N. Francis、A. Green、K. Hare、B. Li、L. Libkin、T. Lindaaker、V. Marsault、W. Martens、J. Michels 等。GQL 和 SQL/PGQ 中的图模式匹配。载于*ACM SIGMOD*论文集，第2246–2258 页，2022。[23] M. Fowler 和 P. Sadalage。NoSQL 数据库与多语言持久化。个人网站：*http://martinfowler.com/articles/nosql-intro-original. pdf*，2012。[24] D. Glake、F. Kiehn、M. Schmidt、F. Panse 和 N. Ritter。迈向多语言数据存储——概述与开放研究问题。*ArXiv* 预印本，abs/2204.05779，2022。

[25] GraphAware。Graphaware Neo4j 时间树，2024。[26] Q. Guo、C. Zhang、S. Zhang 和 J. Lu。多模型查询语言：驾驭大数据的多样性。《分布式与并行数据库》，42(1):31–71，2024。[27] T. Hartmann、F. Fouquet、M. Jimenez、R. Rouvoy 和 Y. Le Traon。利用时态图大规模分析动态复杂数据。载于第*29*届国际软件工程与知识工程会议 *(SEKE'17)*，第 6 页。KSI Research，2017。[28] H. Huang、J. Song、X. Lin、S. Ma 和 J. Huai。Tgraph：时态图数据管理系统。载于第*25*届 *ACM* 国际信息与知识管理会议论文集，第 2469–2472 页，2016。[29] influxdb。InfluxData：InfluxDB 时间序列数据平台，2024。[30] A. Iosup、T. Hegeman、W. L. Ngai、S. Heldens、A. Prat-P´erez、T. Manhardt、H. Chafi、M. Capota、N. Sundaram、M. J. Anderson、I. G. Tanase、Y. Xia、L. Nai 和 P. A. Boncz。LDBC Graphalytics：面向并行与分布式平台的大规模图分析基准测试。*VLDB Endowment*论文集，9(13):1317–1328，2016。[31] M. Jalal、S. Wehbi、S. Chilingaryan 和 A. Kopmann。SciTS：科学实验与工业物联网中时间序列数据库的基准测试。(2022)，12: 1–12: 11，2022。[32] JanusGraph。Apache Cassandra 作为 JanusGraph 的存储后端，2025。[33] janusgraph。JanusGraph：分布式、开源、可大规模扩展的图数据库，2025。[34] S. K. Jensen、T. B. Pedersen 和 C. Thomsen。时间序列管理系统：综述。《*IEEE*知识与数据工程汇刊》，29(11):2581–2600，2017。[35] S. K. Jensen、T. B. Pedersen 和 C. Thomsen。时间序列管理系统：2022 年综述。载于《数据序列管理与分析》。计算机协会，2022。[36] Kafka。Kafka：开源分布式事件流平台，2025。[37] kairosDB。KairosDB：基于 Cassandra 的高速时间序列数据库，2025。[38] A. Khelifati、M. Khayati、A. Dign¨os、D. Difallah 和 P. Cudr´e-Mauroux。Tsm-bench：面向监控应用的时间序列数据库系统基准测试。*VLDB Endowment*论文集，16(11):3363–3376，2023。[39] P. P. Khine 和 Z. Wang。大数据领域中的多语言持久化综述。*Information*，10(4):141，2019。[40] F. Kiehn、M. Schmidt、D. Glake、F. Panse、W. Wingerath、B. Wollmer、M. Poppinga 和 N. Ritter。多语言数据管理：现状与开放挑战。*VLDB Endowment*论文集，15(12):3750–3753，2022。[41] P. Koupil、D. Crha 和 I. Holubov´a。Mm-quecat：用于统一查询多模型数据的工具。载于*EDBT*，第 831–834 页，2023。[42] O. Lajam 和 S. Mohammed。重新审视多语言持久化：从原则到实践。《国际高级计算机科学与应用杂志》，13(5)，2022。[43] A. Lakshman 和 P. Malik。Cassandra：一种去中心化结构化存储系统。*ACM SIGOPS*操作系统评论，44(2):35–40，2010。[44] ldbc。Ldbc金融基准，2024。[45] G. Li 和 C. Zhang。HTAP数据库：新进展与未来方向。载于 Z. G. Ives、A. Bonifati 和 A. E. Abbadi 编，*SIGMOD '22*：国际数据管理会议，美国宾夕法尼亚州费城，*2022* 年 *6* 月 *12* 日至 *17* 日，第 2483–2488 页。ACM，2022。[46] X. Liu、Y. Xia、Y. Liang、J. Hu、Y. Wang、L. Bai、C. Huang、Z. Liu、B. Hooi 和 R. Zimmermann。Largest：用于大规模交通预测的基准数据集。神经信息处理系统进展，36:75354–75371，2023。[47] J. Lu 和 I. Holubov´a。多模型数据库：处理数据多样性的新征程。*ACM*计算调查（*CSUR*），52(3):1–38，2019。[48] J. Lu、Z. H. Liu、P. Xu 和 C. Zhang。Udbms：多模型数据管理的统一之路。载于概念建模进展：*ER 2018*研讨会 *Emp-ER*、*MoBiD*、*MREBA*、*QMMQ*、*SCME*，中国西安，*2018* 年 *10* 月 *22* 日至 *25* 日，论文集 *37*，第 285–294 页。Springer，2018。[49] A. Margara、J. Urbani、F. Van Harmelen 和 H. Bal。流式处理网络：对动态数据进行推理。《网络语义学杂志》，

25:24–44，2014。[50] mariadb。MariaDB。端到端数据库解决方案，2025。[51] mariadb graph。MariaDB 服务器中的 OQGRAPH 存储引擎，2023。[52] mariadb ts。使用 MariaDB ColumnStore 分析金融时间序列数据，2023。[53] M. Massri、Z. Miklos、P. Raipin 和 P. Meye。Clock-G：一种采用空间高效存储技术的时态图管理系统。载于*2022 年 IEEE 第 38 届国际数据工程会议（ICDE）*，第 2263–2276 页。IEEE，2022。[54] K. Mavrogiorgos、A. Kiourtis、A. Mavrogiorgou 和 D. Kyriazis。MongoDB、ArangoDB 和 CouchDB 在大数据存储方面的比较研究。载于*2021 年第五届云计算与大数据计算国际会议论文集*，第 8–14 页，2021。[55] Memgraph。一种图数据库，2025。[56] Y. Miao、W. Han、K. Li、M. Wu、F. Yang、L. Zhou、V. Prabhakaran、E. Chen 和 W. Chen。Immortalgraph：用于存储和分析时态图的系统。*ACM 存储汇刊（TOS）*，11(3):1–34，2015。[57] O. Michail。时态图导论：算法视角。*InternetMathematics*，12(4):239–280，2016。[58] V. Z. Moffitt 和 J. Stoyanovich。时态图代数。载于 T. Rompf 和 A. Alexandrov 编，第 *16* 届数据库编程语言国际研讨会论文集（*DBPL 2017*），德国慕尼黑，*2017* 年 *9* 月 *1* 日，第 10:1–10:12 页。ACM，2017。[59] mongodb。MongoDB：一种文档数据库，2024。[60] S. M. Mousavi、Y. Sheng、W. Zhu 和 G. C. Beroza。斯坦福地震数据集（STEAD）：用于人工智能的全球地震信号数据集。*IEEE Access*，7:179464–179476，2019。[61] MySQL。MySQL：一种关系数据库，2025。[62] neo4j。Neo4j：一种图数据库，2024。[63] Opensource.org。OSI 批准的许可证，2025。[64] Oracle。集成图数据库功能，2025。[65] Oracle。时序数据建模，2025。[66] Oracle。Oracle：一种关系数据库管理系统（RDBMS），2025。[67] Oracle。时间序列机器学习，2025。[68] orientdb。OrientDB 文档与图数据库，2025。[69] A. Patel 和 J. Dharwa。图数据：各领域大数据建模的下一个前沿。《印度科学与技术杂志》，10(21):1–7，2017。[70] J. Pokorn`y。图数据库：其优势与局限。在*IFIP*计算机信息系统与工业管理国际会议，第58–69页。Springer，2015。[71] PostgreSQL。Postgresql：全球最先进的开源数据库Postgresql官方网站，2024。[72] S. Rani 等。时间序列数据库及时间序列挖掘最新研究趋势综述。在*2014*年第五届*Confluence*下一代信息技术峰会国际会议（*Confluence*），第109–115页。IEEE，2014。[73] redis。Redis：端到端数据库解决方案，2025。[74] RedisGraph。RedisGraph是构建于Redis之上的可查询图数据库。[75] RedisTimeSeries。使用Redis摄取和查询时间序列数据。[76] RockDB。RockDB：面向快速存储环境的持久化键值存储，2025。[77] C. Rost、K. Gomez、P. Christen 和 E. Rahm。大型时态图中度量指标的演化。在面向商业、技术与*Web*的数据库系统¨（*BTW 2023*），LNI第P-331卷，第485–507页。德国信息学协会，2023。[78] S. Sc hmidl、P. Wenig 和 T. Papenbrock。时间序列异常检测：全面评估。*VLDB Endowment*论文集，15(9):1779–1797，2022。[79] Z. Shao、Z. Zhang、F. Wang 和 Y. Xu。用于多变量时间序列预测的预训练增强型时空图神经网络。在第*28*届*ACM SIGKDD*知识发现与数据挖掘大会论文集，第1567–1577页，2022。[80] M. Stonebraker、U. ˛Cetintemel 和 S. Zdonik。实时流处理的8项要求。《*ACM SIGMOD*记录》，34(4):42–47，2005。[81] R. Streams。Redis Streams简介。[82] surrealdb。Surrealdb：面向人工智能代理和应用的终极多模型数据库，2025。[83] G. Theodorakis、J. Clarkson 和 J. Webber。Aion：高效的时态图数据管理。在*EDBT*，第501–514页，2024。[84] TigerGraph。一种图数据库，2025。

[85] Timescaldb。Timescaldb：一种开源时间序列 SQL 数据库，2024。[86] timetraveldb。Timetraveldb：结合Neo4j和TimescaleDB的多语言持久化方案，2025。[87] C. Vicknair、M. Macias、Z. Zhao、X. Nan、Y. Chen 和 D. Wilkins。图数据库与关系数据库的比较：数据溯源视角。在第*48*届*ACM*东南会议论文集，第1–6页，2010。[88] L. K. Visperas 和 Y. Chodpathumwan。面向功率测量数据的时间序列数据库基准测试框架。在*2021*年研究、发明与创新大会：电气与电子创新（*RI2C*），第25–30页。IEEE，2021。[89] Y. Watanabe、S. Yamada、H. Kitagawa 和 T. Amagasa。集成流处理引擎与数据库以实现持久化流数据管理。在数据库与专家系统应用国际会议，第414–423页。Springer，2007。[90] N. white paper。“克服SQL瓶颈与SQL痛点”，2015年4月。[91] L. Wiese。多语言数据库架构=多语言挑战。在LWA，第422–426页，2015。[92] H. Wu、J. Cheng、S. Huang、Y. Ke、Y. Lu 和 Y. Xu。时态图中的路径问题。*VLDB Endowment*论文集，7(9):721–732，2014。

[93] Y. Xu、J. Huang、A. Liu、Z. Li、H. Yin 和 L. Zhao。在大型时序图中的时间约束图模式匹配。载于亚太网络会议（*APWeb*）与 *Web* 时代信息管理（*WAIM*）联合会议：*Web* 与大数据，第 100–115 页。Springer，2017 年。[94] L. Yu。时态图基准的实证评估。$CoRR$，abs/2307.12510，2023 年。[95] H. Yuanzhe、Q. Xiongpai、C. Yueguo、L. Yaru、S. Xiaoguang、T. Yu、Z. Xiao 和 D. Xiaoyong。TS-Benchmark：时间序列数据库基准测试。载于*2021* 年 *IEEE* 第 *37* 届国际数据工程会议（*ICDE*），2021 年。[96] C. Zhang、J. Lu、P. Xu 和 Y. Chen。UniBench：多模型数据库管理系统基准测试。载于人工智能时代的性能评估与基准测试：第十届 *TPC* 技术会议（*TPCTC 2018*），巴西里约热内卢，*2018* 年 *8* 月 *27–31* 日，修订版精选论文 *10*，第 7–23 页。Springer，2019 年。[97] F. Zhang、Z. Li、D. Peng 和 J. Cheng。用于时态数据管理的资源描述框架（RDF）——综述。地球科学信息学，14(2):563–599，2021 年。[98] F. Zhang、K. Wang、Z. Li 和 J. Cheng。基于资源描述框架（RDF）的时态数据表示与查询。$IEEE\ Access$，7:85000–85023，2019 年。