EasyChair Preprint

№ 16035

Unified Heterogeneous Database Workload Representation

Vishal Sharma

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

November 22, 2026

Unified Heterogeneous Database Workload Representation

Vishal Sharma
(Last Affiliation, Feb 2024 - July 2026)
Department of Healthcare Administration and Policy
University of Nevada, Las Vegas
vishal_sharma@live.com

Abstract

The choice of a DBMS engine (*e.g.,* PostgreSQL) for a data type (e.g., relational) is crucial for an effective storage, retrieval, and operations on data. Many modern computer applications are supported by heterogeneous databases. Such applications uses various database management systems (DBMSs *e.g.,* Postgres, mongoDB) for running several programs/applications, *e.g.,* an application supported by a relational DBMS, another by a graph DBMS, and another by a key/value store. Database workloads from such modern applications consist of queries and workloads from several databases. While analyzing workloads from a single platform databases (Definition 1.1) can improve performance of a single database, unifying and analyzing workloads from heterogeneous databases will improve both individual/single (self-supporting) and inter-dependent (system-supporting) DBMSs performance.

This paper outlines the importance of unifying heterogeneous database workloads, formulates & defines the problem of representing heterogeneous workloads. The paper proposes an approach for heterogeneous database workload representation with LLMs by generating workload embeddings and performs clustering to measure LLMs capability in identifying clusters of similar workloads.

1 Introduction

The complexity of a large, real-world computer applications are better represented by diverse DBMS to support heterogeneous data types for efficient storage and operations as *one DBMS does not fit all* [35]. Applications in the fields of airlines, hospitals and petroleum companies often have hundreds of databases [17], supported by various DBMSs. Diverse DBMS in the real-world applications with heterogeneous databases include the following:

R1 Scientific data such as astronomy data uses several DBMSs to store large amount of images, text, key-values, and graphs to support search and applications [22, 27].

R2 Oceanography (ocean metagenomics) collects many TB of data to store genomics sequences, text, sensor metadata and streaming data. The data is stored in various databases to support data search and applications [13, 26].

R3 Real-time electrical power failure diagnostics at Siemens collects data from heterogeneous data streams, stored in diverse databases, produced by various sensors from turbines, generators and compressors at power plants [19].

R4 In healthcare the availability of biobank datasets has laid the groundwork for multimodal AI solutions to understand diseases from a broader perspective [31]. Healthcare data such as a electronic health record is stored using document model (semi-structured with unstructured text), patient information is stored in relational model and unstructured storage format for diagnostics imaging [1, 28].

The growing number of large modern computer applications are supported by diverse data types utilizing heterogeneous databases and has resulted in accumulation of *heterogeneous database workloads*. Heterogeneous and single database platform workloads are defined as follows:

Definition 1.1 (Single-platform database workload)**.** A database workload that can ONLY be executed entirely on a single database engine, *e.g.,* a workload from a PostgreSQL database. ■

Definition 1.2 (Heterogeneous database workload)**.** A collection of database workloads from several (heterogeneous) DBMSs *e.g.,* PostgreSQL, MongoDB, and Neo4j. ■

Example 1.1. *Ⓢ is a social media platform with user profile, network of friends, users posts/images/videos/text, scroll activities and a marketplace for buying and selling items. Ⓢ platform is similar to Facebook. Data generated from Ⓢ is stored on diverse database such as* ***transaction*** *records stored in RDBMS,* ***unstructured*** *user's posts and multimedia stored in MongoDB, real-time* ***streaming*** *users activity information in timeScaleDB,* ***vectors*** *for recommendation engines in pgVector and* ***graphs*** *of social connections stored in Neo4j. Configuration tuning of Neo4j will improve storage, search and retrieval of social network and similarly tuning of MongoDB will improve storage, search and retrieval of user's posts and multimedia. Such configuration tuning hits a limitation in terms of optimized overall system performance and require at least a minimal co-operation between databases for further system improvement. Below are two scenarios to support our claims:*

i) Single database configuration tuning does not take into the account the workload patterns (such as short and long lived temporal patterns) across all of the databases e.g., in case of a viral content, breaking news or major system failure.

ii) Single database configuration tuning does not support tuning of complex queries requiring data integration from other databases. For example, queries from Ⓢ that combine data from MongoDB, Neo4j and timeScaleDB.

To improve configuration tuning of DBMS and attain improved performance there is a need for both individual (self-supporting) and inter-dependent (system-supporting) configurations tuning. To achieve both individual and inter-dependent tuning, it requires analysis of heterogeneous database workloads. The challenge persist in unified representation of heterogeneous database workloads for analysis. Our proposed framework overcomes this challenge and is a way forward that demonstrates a framework for unified representation of heterogeneous database workloads.

To enable a comprehensive understanding of a system-wide workload, applications involving heterogeneous databases necessitate a *paradigm shift from the analysis of a single database workload to a heterogeneous database workload* [25]. The proposed paradigm is orthogonal to federated database systems (FDBSs). A FDBS is a

The work was done during the time at University of Nevada, Las Vegas where author Vishal Sharma was an Assistant Professor of Big Data Science.

Conference'17, Washington, DC, USA
2026. ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

heterogeneous collection of databases for which a common (federated) schema has been created and a common query system is used. Most real-world databases may not have common schema as shown in real-world examples of (R1-R4), due to variety and sources of data. Moreover, an FDBS requires upfront deployment of system infrastructure [33], in contrast our approach does not need specialized database deployment.To motivate the proposed paradigm from single to heterogeneous analysis of workloads, we explore an example in Example 1.1 and a few scenarios with real-world utility.

Moreover, heterogeneous database workload analysis overcomes the following limitations of analysis of single platform workload:

L1 *Inadequate representation of real-world complexity*: Workloads from a single-platform database presents a micro-view of the broader workload patterns that exist in the larger system. A heterogeneous database workload includes both micro (single-platform workload) and macro (heterogeneous database workload) views of larger system and is crucial for holistic understanding of patterns among and across workloads.

L2 *Missing cross-system dependencies*: Single-platform workload does not explore the inter-relation of heterogeneous databases. Databases in an application are often related and a heterogeneous workload representation and analysis can extract and explore the patterns of such relationships.

L3 *Performance improvement bottlenecks*: For an improved system performance configuring database for both individual (self-supporting) and inter-dependence (system-supporting) is a way forward. Configuring a single database may lead to tuning that limits overall system performance since optimal system performance may have inter-database dependencies.

Work related to database workload representation and analysis with compression or reduction [9–11, 34] and a recent work on real-time workload pattern analysis [40] are focused on a single database engine. There is a very recent work on data unification for LLMs, however the proposed work is not in the context of heterogeneous database workloads [24]. To the best of our knowledge heterogeneous DBMS workload representation and analysis has received less attention; the vast majority of the previous work has focused on single DBMS workloads.

The differences in the syntax of database query languages pose a challenge to creating a unified heterogeneous database workload. As an example consider the "same" queries in SQL (a standard query language for relational databases), mongoDB (the most popular JSON data store), and GQL (a standard query language for graph databases):

```
SQL: select count(*) from lineitem where id <= 7323842
mongoDB: mongodb.lineitem.find({id:{$lte:7323842}}).count()
GQL: match (i:lineitem) where i.id <= 7323842 return count(i)
```

Semantically, the queries are similar, all find data from `lineitem` that have an `id` value in a range. There are syntactic differences: `select` vs. `find` and `id <= 7323842` vs. `id:{ $lte:7323842}`). These differences are a challenge in unified representation of workloads from diverse DBMS platforms. We observe that compared to linguistics that suffers from subjectivity, structured languages do not and that this is a critical observation to overcome the challenges of unified representation of heterogeneous workloads. LLMs have been shown to be effective at unsupervised encoding [5]. Specifically, by embedding workloads to a vector space a shared unified heterogeneous representation can be achieved with LLMs. The idea of vector representation of a workload from a single database has been proposed and evaluated previously [8] and has been applied for single DB tuning such as online index tuning [32] with word2vec and pre-trained transformer models respectively. This paper proposes both pre-trained and fine-tuned LLMs in its capability to effectively learn the syntactic (and semantic) textual relationships in creating a unified database workload representation. Though we focus on high-level queries, our work equally applies to query plans. For instance the same SQL query compiled using Postgres vs. MySQL will yield different (syntactically, and sometimes operationally) query plans. But the strategy proposed in this paper would also work for unifying query plans (though there is extra database computation overhead to first generate the query plans and then unify at that level).

This paper makes following **vision** contribution:

i) The paper formulates the problem and proposes an approach for unified heterogeneous database workload representations with pre-trained and fine-tuned LLMs.

This paper is organized as follows. The next section describes previous approaches for workload representation, formulates the problem and proposes an approach for unified workload representation. The Section 3 covers related work and Section 4 is conclusion.

2 Database Workload Representation

2.1 String Literal

In previous research, queries in a database workload were represented as a string literal. String literal representation has limitations as identified in previous work in [3]. For example. consider the following three SQL queries from [3],

```
select * from Employers where birthyear < 1980
select * from Employees where birthyear < 1980
select * from Employed where birthyear < 1980
```

The queries extract similar information from *different tables*. The names of the tables differ only in two letters and measuring queries similarity represented as strings results in very high scores such as with SequenceMatcher [1] it will result in the range of 0.82-0.88 similarity scores. However, when evaluated on a database the queries access three different tables and may yield different outputs. Hence they do not have high similarity as database queries.

2.2 Feature Set

Some of the limitations of query representation as a string literal could be overcome with feature based query representation.The features could be numeric and/or categorical as previously reported [11]. Categorical features such as counts of tables referenced, functions used, columns referenced, group by and order by columns

[1] https://docs.python.org/3/library/difflib.html

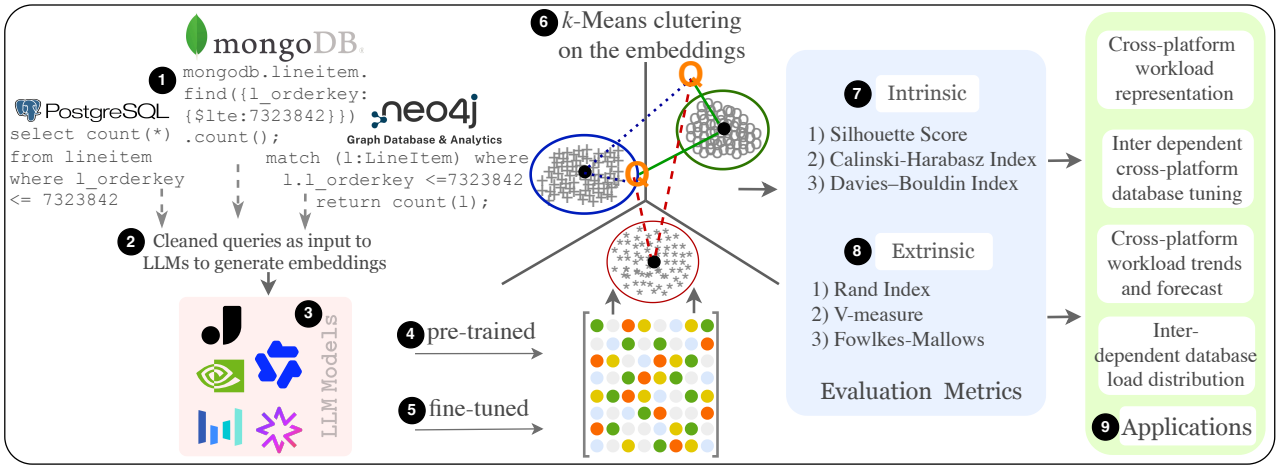

Figure 1: The proposed framework of heterogeneous representations with LLM models. The input queries of heterogeneous database are cleaned and fed as triplets (SQL, MongoDB, Neo4j) to pre-trained and fine-tuned LLM models to generate embeddings. *k*-Means clusters of the embedding from (i) pre-trained (ii) fine-tuned LLMs are generated. To measure the performance of LLM representation, the clusters are evaluated using intrinsic and extrinsic metrics. The Q represents a query with respective distance from dimensions.

are common. Numerical feature such as execution time, planning time, total size of the input to the query, total output rows, CPU time and number of joins are commonly used. Feature set-based representation is confined to pre-defined rules. It does not consider the values (such as filtering) used in the queries. Two queries that produce the same results with one as a subquery and the other using join syntax will have different feature sets. This remains the limitation of feature set based workload representation. With the limitations, neither the string literal approach nor the feature set representation posses the capacity to unify database workloads.

2.3 Query Embedding

Every query in a workload could be viewed as an *independent* statement. A workload is a collection of statements, where each statement is a database query. Pre-trained transformers, such as BERT, that are trained on a very large corpus of textual information from the web makes it possible to represent textual information in Euclidean space. This process is known as textual embedding and is represented by a fixed size length of integer vector (*e.g.,* 512). Specifically for a query, SentenceTransformer[2] can embed a statement to a fixed length vector. Unlike limitations of previous approaches of representing query as a string literal [3] or feature set [11], representing queries in Euclidean space opens the opportunity of various findings with vector analysis such as for identifying clusters and patterns.

Embeddings are high-dimensional vectors in a mathematical space that captures the syntactic and semantic meaning of the input data. The process involves converting text, images or any other types of data into fixed-length vectors to facilitate downstream tasks. In the context of databases, previous researches has demonstrated the effectiveness of embeddings from pre-trained transformer based NLP models, also known as large language models (LLMs), on the semantic textual information of SQL queries [6, 8, 15, 16, 32]. LLMs take a sequence of tokens as an input for standardization and meaningful representation. The input text is standardized with preprocessing steps of marking the beginning with [CLS] and [SEP] at the end of a statement[3]. LLMs are trained on a very large corpus of data and are generally an unsupervised multitask learner [29]. During the training an LLM builds a vocabulary from training data and uses a tokenization algorithm (*e.g.,* Byte Pair Encoding (BPE)), which converts an input text into its corresponding integer ID. These tokens are converted to token embeddings which is learned during training. All token embeddings are pooled to form an embedding for a sentence (in our case, for a query). This paper uses the mean pooling strategy, which performed the best, as proposed in the paper on sentence transformers [30]. In this paper, the workloads are represented as embeddings to overcome the challenges of unified database workload representation.

2.4 Unified Workload Representation

Definition 2.1 (Heterogeneous database workload representation)**.** A heterogeneous database workload representation generates a unified representation that is database engine independent. A heterogeneous workload representation is:

	Symbol	Description
D	$=\{D_1, .., D_n\}$	n database engine
$\mathcal{W}$	$=\{\mathcal{W}^1_{D_1}, .., \mathcal{W}^m_{D_n}\}$	m workloads from n databases
$\mathcal{W}^k_{D_1}$	$=\{q^1_{D_1}, .., q^j_{D_1}\} \in Q_{D_1}$	j queries in $\mathcal{W}^k_{D_1}$ workloads from D_1
$\mathcal{F}_\theta$	$=\{Q_{D_1} \cup Q_{D_2} .. \cup Q_{D_n}\}$	workload representation function

$$\mathcal{F}_\theta = \bigcup_{i=1}^{n} Q_{D_i} \longrightarrow \mathbb{R}^d \tag{1}$$

where $\mathcal{F}_\theta$ function maps a workload Q_{D_i} from D_i database into a common vector space with d dimensions. The common vector space

[2]https://sbert.net/

[3][CLS] and [SEP] are special tokens for BERT and similar models; other models may use a different special token.

d is achieved through the output dimension of the embeddings from LLM models. k represents number of workloads from a database engine. The total number of workloads m ($m > k$) and number of database engines n are different in numbers as there could be many workloads from a database engine.

2.4.1 Unified Workload Representation with LLMs. The proposed framework begins with generating dataset that are triplet queries from three different database platforms: PostgreSQL (SQL), MongoDB (document-based), and Neo4j (graph database), the triplet performs the same operation on the same dataset, however on different database engine. The purpose of triplet queries is to measure effectiveness of LLM's performance for heterogeneous representation. The process involves cleaning these queries and feeding them into LLMs to generate embeddings. The generated embeddings are passed through a k-means algorithm, the output clusters are evaluated using two categories of metrics to measure structural and semantic performance with intrinsic (silhouette, calinski-harabasz, davies-bouldin) and extrinsic (rand index, v-measure, fowlkes-mallows) evaluation. The cluster evaluations are effectiveness of LLMs capacity to identify and represent heterogeneous database workloads. Figure 1 displays the proposed methodology for heterogeneous database workload representation with LLMs.

3 Related Work

Related work falls into two categories: heterogeneous data storage and workload representation.

3.1 Heterogeneous Data Storage

Managing heterogeneous data types from various sources has been an active area of research under the topic of *polystore* [4], *polyglot*, *multi-modal* [5] and *heterogeneous* data management systems [13, 20]. There has been previous work related to processing heterogeneous data types from different sources and a few heterogeneous data processing systems have also been proposed [2, 21]. Recently, the Apache Wayang framework was proposed to solve the problem of data unification and analytics for heterogeneous data types [4, 18]. However, such systems need a mediator that communicates with the diverse databases to fulfill user queries, *what remains a challenge for a mediator framework is a unified universal data and user workload representation.* There have been several recent advancements in a universal representation for database workloads [15, 16, 37] and database relational representation [12, 14, 23, 39] with graphs, deep learning and large language models (LLMs). This seems to be a current trend and a future direction towards relational and graph database representation. The real-world DB system that runs on diverse databases brings one of the unexplored challenges that is of *heterogeneous database workload representation and analysis.* Our work extends previous work to support workload analysis by introducing a framework for heterogeneous database workload representation.

[4]access to diverse data types and sources with native query language
[5]access to diverse data types and sources with single query language

3.2 Vectorized Workload Representation

Workload representation based on deep learning and large language models has gained tremendous popularity. Embedding a query in a workload to a vector representation is the current technique to standardize workload representation. An alternative method is feature-based extraction and query representation. But feature-based representation does not extract semantic and syntactic relationship between queries, which is captured by the vectorization method. Transforming queries to vectors with *word2vec* and *GloVe* models was proposed for various tasks such as to support similarity search using cosine similarity measures [6–8]. *Query2Vec* was proposed using Doc2Vec and LSTM (Long Short Term Memory) based methods to vectorize queries for workload summarization and to identify errors with workload classification [15, 16]. Pre-trained language models have been used for annotating the columns of a table in the multi-task learning framework *DoDuo* [36]. The goal was to predict column types and relationships. LLMs have also been used for column correlation prediction [38]. Another work *PreQR*, a system for query vectorization with graph neural networks to encode workload enriched with database schema [37]. To the best of our knowledge previous approaches on query vectorization for workload representation have focused on workloads from a single database engine and heterogeneous database workload representation has received less attention. Defining, executing and evaluating cross platform workload representation remains an under-researched problem. With the growing number of real-world applications with heterogeneous databases, heterogeneous database workload representation is of high importance for improved systems configuration and workload analysis.

4 Conclusion

This paper identifies the problem of unifying heterogeneous database workloads, formulates the problem and proposes a method for unifying the workload representation. This work proposes a framework for unified database workload representation for modern large, heterogeneous database systems.

References

[1] Julián N Acosta, Guido J Falcone, Pranav Rajpurkar, and Eric J Topol. 2022. Multimodal biomedical AI. *Nature Medicine* 28, 9 (2022), 1773–1784.

[2] Divy Agrawal, Sanjay Chawla, Bertty Contreras-Rojas, Ahmed Elmagarmid, Yasser Idris, Zoi Kaoudi, Sebastian Kruse, Ji Lucas, Essam Mansour, Mourad Ouzzani, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Nan Tang, Saravanan Thirumuruganathan, and Anis Troudi. 2018. RHEEM: enabling cross-platform data processing: may the big data be with you! *Proc. VLDB Endow.* 11, 11 (jul 2018), 1414–1427. https://doi.org/10.14778/3236187.3236195

[3] Natalia Arzamasova, Klemens Böhm, Bertrand Goldman, Christian Saaler, and Martin Schäler. 2019. On the usefulness of SQL-query-similarity measures to find user interests. *IEEE Transactions on Knowledge and Data Engineering* 32, 10 (2019), 1982–1999.

[4] Kaustubh Beedkar, Bertty Contreras-Rojas, Haralampos Gavriilidis, Zoi Kaoudi, Volker Markl, Rodrigo Pardo-Meza, and Jorge-Arnulfo Quiané-Ruiz. 2023. Apache Wayang: A Unified Data Analytics Framework. *ACM SIGMOD Record* 52, 3 (2023).

[5] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961* (2024).

[6] Rajesh Bordawekar, Bortik Bandyopadhyay, and Oded Shmueli. 2017. Cognitive database: A step towards endowing relational databases with artificial intelligence capabilities. *arXiv preprint arXiv:1712.07199* (2017).

[7] Rajesh Bordawekar and Oded Shmueli. 2016. Enabling cognitive intelligence queries in relational databases using low-dimensional word embeddings. *arXiv preprint arXiv:1603.07185* (2016).

[8] Rajesh Bordawekar and Oded Shmueli. 2017. Using Word Embedding to Enable Semantic Queries in Relational Databases. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning* (Chicago, IL, USA) *(DEEM'17)*. Association for Computing Machinery, New York, NY, USA, Article 5, 4 pages. https://doi.org/10.1145/3076246.3076251

[9] Matteo Brucato, Tarique Siddiqui, Wentao Wu, Vivek Narasayya, and Surajit Chaudhuri. 2024. Wred: Workload Reduction for Scalable Index Tuning. 2, 1, Article 50 (mar 2024), 26 pages. https://doi.org/10.1145/3639305

[10] Surajit Chaudhuri, Ashish Kumar Gupta, and Vivek Narasayya. 2002. Compressing SQL workloads *(SIGMOD '02)*. Association for Computing Machinery, New York, NY, USA, 488–499. https://doi.org/10.1145/564691.564747

[11] Shaleen Deep, Anja Gruenheid, Paraschos Koutris, Jeffrey Naughton, and Stratis Viglas. 2020. Comprehensive and efficient workload compression. *Proc. VLDB Endow.* 14, 3 (nov 2020), 418–430. https://doi.org/10.14778/3430915.3430931

[12] Haoyu Dong and Zhiruo Wang. 2024. Large Language Models for Tabular Data: Progresses and Future Directions. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 2997–3000. https://doi.org/10.1145/3626772.3661384

[13] Jennie Duggan, Aaron J Elmore, Michael Stonebraker, Magda Balazinska, Bill Howe, Jeremy Kepner, Sam Madden, David Maier, Tim Mattson, and Stan Zdonik. 2015. The bigdawg polystore system. *ACM Sigmod Record* 44, 2 (2015), 11–16.

[14] Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. 2024. Position: Relational Deep Learning - Graph Representation Learning on Relational Databases. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (Eds.). PMLR, 13592–13607. https://proceedings.mlr.press/v235/fey24a.html

[15] Shrainik Jain, Bill Howe, Jiaqi Yan, and Thierry Cruanes. 2018. Query2Vec: An Evaluation of NLP Techniques for Generalized Workload Analytics. *Proceedings of the VLDB Endowment* 11, 5 (2018).

[16] Shrainik Jain, Jiaqi Yan, Thierry Cruane, and Bill Howe. 2019. Database-agnostic workload management. In *Conference on Innovative Data Systems Research (CIDR)*.

[17] Zoi Kaoudi and Jorge-Arnulfo Quiané-Ruiz. 2018. Cross-platform data processing: use cases and challenges. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 1723–1726.

[18] Zoi Kaoudi and Jorge-Arnulfo Quiané-Ruiz. 2022. Unified data analytics: state-of-the-art and open problems. *Proceedings of the VLDB Endowment* (2022), 3778–3781.

[19] Evgeny Kharlamov, Theofilos Mailis, Konstantina Bereta, Dimitris Bilidas, Sebastian Brandt, Ernesto Jiménez-Ruiz, Steffen Lamparter, Christian Neuenstadt, O Özçep, Ahmet Soylu, et al. 2016. A semantic approach to polystores. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 2565–2573.

[20] Felix Kiehn, Mareike Schmidt, Daniel Glake, Fabian Panse, Wolfram Wingerath, Benjamin Wollmer, Martin Poppinga, and Norbert Ritter. 2022. Polyglot data management: state of the art & open challenges. *Proceedings of the VLDB Endowment* 15, 12 (2022), 3750–3753.

[21] Sebastian Kruse, Zoi Kaoudi, Jorge-Arnulfo Quiané-Ruiz, Sanjay Chawla, Felix Naumann, and Bertty Contreras-Rojas. 2019. Optimizing cross-platform data movement. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1642–1645.

[22] Nicholas M Law, Shrinivas R Kulkarni, Richard G Dekany, Eran O Ofek, Robert M Quimby, Peter E Nugent, Jason Surace, Carl C Grillmair, Joshua S Bloom, Mansi M Kasliwal, et al. 2009. The Palomar Transient Factory: system overview, performance, and first results. *Publications of the Astronomical Society of the Pacific* 121, 886 (2009), 1395.

[23] Jure Leskovec. 2023. Databases as Graphs: Predictive Queries for Declarative Machine Learning. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems* (Seattle, WA, USA) *(PODS '23)*. Association for Computing Machinery, New York, NY, USA, 1. https://doi.org/10.1145/3584372.3589939

[24] Hao Liang, Xiaochen Ma, Zhou Liu, Zhen Hao Wong, Zhengyang Zhao, Zimo Meng, Runming He, Chengyu Shen, Qifeng Cai, Zhaoyang Han, et al. 2025. DataFlow: An LLM-Driven Framework for Unified Data Preparation and Workflow Automation in the Era of Data-Centric AI. *arXiv preprint arXiv:2512.16676* (2025).

[25] Jiaheng Lu and Irena Holubová. 2019. Multi-model databases: a new journey to handle the variety of data. *ACM Computing Surveys (CSUR)* 52, 3 (2019), 1–38.

[26] Tim Mattson, Vijay Gadepally, Zuohao She, Adam Dziedzic, and Jeff Parkhurst. 2017. Demonstrating the BigDAWG Polystore System for Ocean Metagenomics Analysis. In *Conference on Innovative Data Systems Research ()*.

[27] Manoj Poudel, Rashmi P Sarode, Shashank Shrestha, Wanming Chu, and Subhash Bhalla. 2019. Development of a polystore data management system for an evolving big scientific data archive. In *Heterogeneous Data Management, Polystores, and Analytics for Healthcare: VLDB 2019 Workshops, Poly and DMAH, Los Angeles, CA, USA, August 30, 2019, Revised Selected Papers 5*. Springer, 167–182.

[28] Abdul Quamar, Jannik Straube, and Yuanyuan Tian. 2020. Enabling Rich Queries Over Heterogeneous Data From Diverse Sources In HealthCare. In *Conference on Innovative Data Systems Research (CIDR)*.

[29] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[30] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[31] Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. 2011. Multiparameter Intelligent Monitoring in Intensive Care II: a public-access intensive care unit database. *Critical care medicine* 39, 5 (2011).

[32] Vishal Sharma and Curtis Dyreson. 2022. Indexer++: workload-aware online index tuning with transformers and reinforcement learning. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing* (Virtual Event) *(SAC '22)*. Association for Computing Machinery, New York, NY, USA, 372–380. https://doi.org/10.1145/3477314.3507691

[33] Amit P. Sheth and James A. Larson. 1990. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.* 22, 3 (Sept. 1990), 183–236. https://doi.org/10.1145/96602.96604

[34] Tarique Siddiqui, Saehan Jo, Wentao Wu, Chi Wang, Vivek Narasayya, and Surajit Chaudhuri. 2022. ISUM: Efficiently Compressing Large and Complex Workloads for Scalable Index Tuning. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22)*. Association for Computing Machinery, New York, NY, USA, 660–673. https://doi.org/10.1145/3514221.3526152

[35] Michael Stonebraker, Samuel Madden, Daniel J. Abadi, Stavros Harizopoulos, Nabil Hachem, and Pat Helland. 2018. *The end of an architectural era: it's time for a complete rewrite*. Association for Computing Machinery and Morgan & Claypool, 463–489. https://doi.org/10.1145/3226595.3226637

[36] Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. 2022. Annotating columns with pre-trained language models. In *Proceedings of the 2022 International Conference on Management of Data*. 1493–1503.

[37] Xiu Tang, Sai Wu, Mingli Song, Shanshan Ying, Feifei Li, and Gang Chen. 2022. PreQR: Pre-training Representation for SQL Understanding. In *Proceedings of the 2022 International Conference on Management of Data* (Philadelphia, PA, USA) *(SIGMOD '22)*. Association for Computing Machinery, New York, NY, USA, 204–216. https://doi.org/10.1145/3514221.3517878

[38] Immanuel Trummer. 2023. Can Large Language Models Predict Data Correlations from Column Names? *Proc. VLDB Endow.* 16, 13 (sep 2023), 4310–4323. https://doi.org/10.14778/3625054.3625066

[39] Boris van Breugel and Mihaela van der Schaar. 2024. Position: Why Tabular Foundation Models Should Be a Research Priority. In *Forty-first International Conference on Machine Learning*.

[40] Jiaqi Wang, Tianyi Li, Anni Wang, Xiaoze Liu, Lu Chen, Jie Chen, Jianye Liu, Junyang Wu, Feifei Li, and Yunjun Gao. 2023. Real-Time Workload Pattern Analysis for Large-Scale Cloud Databases. *Proc. VLDB Endow.* 16, 12 (aug 2023), 3689–3701. https://doi.org/10.14778/3611540.3611557