AI-Driven Database Management: A Survey

www.surveyx.cn

Abstract

This survey paper explores the transformative role of artificial intelligence (AI) and machine learning (ML) in enhancing database management systems (DBMS). By integrating AI-driven methodologies, modern DBMS have achieved significant advancements in query optimization, resource allocation, and anomaly detection, addressing the limitations of traditional systems. Key innovations include learned indexes, such as the PGM-index, which improve query latency and space efficiency, and AI-driven frameworks like openGauss that enhance performance through innovative architectural designs. The paper also examines the dual perspective of AI for databases (AI4DB) and databases for AI (DB4AI), highlighting how AI augments database functionalities and supports AI model deployment. Additionally, the survey discusses the integration of multi-task meta-learning frameworks, which facilitate transferable knowledge across tasks and databases, and the evolution of AI-driven text-to-SQL systems. Challenges such as model interpretability and the integration of heterogeneous data sources are addressed, emphasizing the need for more dynamic and adaptable data management solutions. The paper concludes by underscoring the potential of AI and ML to revolutionize database management, paving the way for more intelligent, scalable, and efficient data-driven applications.

1 Introduction

1.1 Significance of AI and ML in Database Management

The integration of artificial intelligence (AI) and machine learning (ML) into database management systems (DBMS) marks a significant advancement in modern data handling frameworks. Traditional systems often face challenges with scalability, reliability, and efficiency, particularly in big data and dynamic cloud environments [1]. AI and ML address these issues by automating complex tasks such as query optimization and resource allocation, enhancing the overall efficiency and scalability of DBMS.

Techniques like learned query superoptimization allow query optimizers to improve performance by learning from historical data, thus enhancing efficiency in repetitive queries within modern analytics systems [2]. Frameworks such as Hydro utilize real-time execution data to dynamically adjust query plans, which reduces execution time and improves resource utilization [3]. Furthermore, the application of deep learning for cardinality estimation significantly enhances the accuracy and efficiency of query optimization processes, which is vital for environments requiring precise execution plans for complex data interactions [4].

The adoption of AutoML techniques automates machine learning processes, particularly in IoT data analytics, thereby improving efficiency in managing dynamic data [5]. This automation is essential for handling the vast datasets typical of contemporary DBMS [6]. However, challenges such as model interpretability and the integration of diverse data sources persist, as highlighted by limitations in existing tabular question answering systems [7]. Addressing these challenges is crucial for the effective deployment of AI and ML in DBMS, paving the way for more adaptable data management solutions [8].

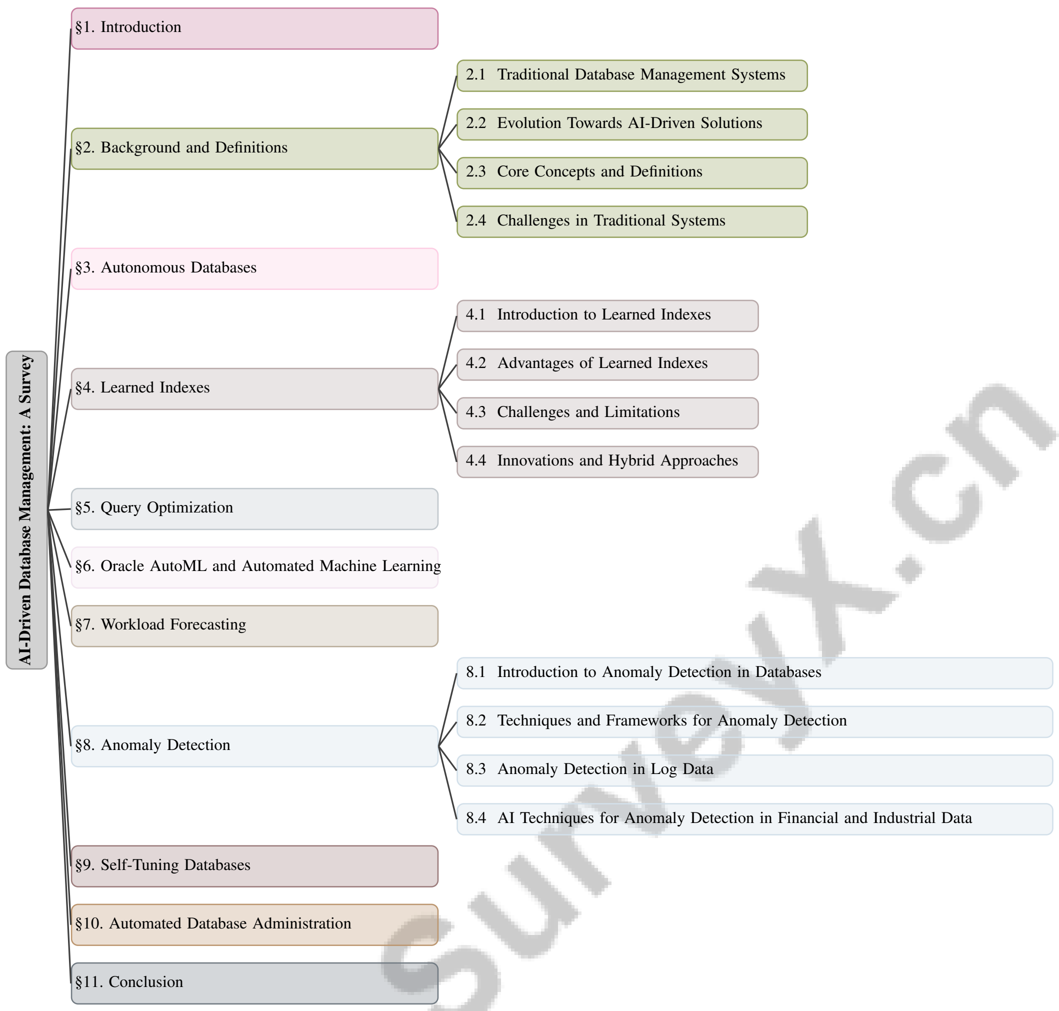

AI-Driven Database Management: A Survey
§1. Introduction
§2. Background and Definitions
2.1 Traditional Database Management Systems
2.2 Evolution Towards AI-Driven Solutions
2.3 Core Concepts and Definitions
2.4 Challenges in Traditional Systems
§3. Autonomous Databases
§4. Learned Indexes
4.1 Introduction to Learned Indexes
4.2 Advantages of Learned Indexes
4.3 Challenges and Limitations
4.4 Innovations and Hybrid Approaches
§5. Query Optimization
§6. Oracle AutoML and Automated Machine Learning
§7. Workload Forecasting
§8. Anomaly Detection
8.1 Introduction to Anomaly Detection in Databases
8.2 Techniques and Frameworks for Anomaly Detection
8.3 Anomaly Detection in Log Data
8.4 AI Techniques for Anomaly Detection in Financial and Industrial Data
§9. Self-Tuning Databases
§10. Automated Database Administration
§11. Conclusion

Figure 1: chapter structure

The transformative impact of AI and ML in database management is evident in their ability to modernize DBMS through automation and optimization, enhancing performance and scalability while overcoming traditional system limitations. Nonetheless, the lack of large-scale public datasets has hindered AIOps development, complicating fair comparisons of approaches and ensuring their generality and real-world performance [9]. Additionally, integrating heterogeneous production data is vital for improving industrial analytics, emphasizing the need for AI-driven solutions in managing diverse data sources effectively [10].

1.2 Objectives of the Survey

This survey elucidates the transformative role of AI-driven methodologies in enhancing database management systems (DBMS) by integrating AI and ML to address traditional system limitations. Key objectives include improving incident management and system supervision, crucial components of modern IT operations [11]. The survey highlights the dual perspective of AI for databases (AI4DB) and databases for AI (DB4AI), showcasing how AI can enhance database functionalities and how database techniques can optimize AI model deployment [12].

Another critical objective is to explore multi-task meta-learning frameworks that capture transferable knowledge across tasks and databases, promoting efficient learning processes [13]. This aligns with the broader goal of developing interconnected frameworks that enhance big data quality through improved assessment, anomaly detection, and correction processes [14].

The survey also aims to improve the interpretability of machine learning models by synthesizing k-nearest neighbors with information theory [15]. Addressing challenges faced by DBMS users will uncover opportunities for automating DBMS administration [16]. Additionally, the survey examines the challenges of applying ML techniques to IoT data analytics, focusing on model selection, tuning, and updating processes [5].

Furthermore, the survey provides a comprehensive overview of the evolution of AI-driven text-to-SQL systems, emphasizing advancements in large language model architectures and the significance of datasets in fostering progress [17]. It investigates approximation schemes for many-objective query optimization (MOQO), which efficiently generate near-optimal plans, significantly reducing the time required for exhaustive optimization [18]. The survey also addresses challenges in machine learning-based resource management in cloud computing, highlighting the necessity for intelligent resource management strategies that adapt to dynamic workloads [6]. It encompasses the evolution of database systems, including object-relational integration, web services, transaction processing, and data mining [8].

1.3 Overview of Key Terms

In AI-driven database management, several key terms are essential for understanding the integration of AI and ML technologies within database systems. AI for Databases (AI4DB) encompasses learning-based techniques for configuration tuning, query optimization, and enhancing security measures [12]. These methodologies aim to automate and optimize database operations by leveraging AI capabilities to improve performance and resilience.

Conversely, Databases for AI (DB4AI) refers to the development of AI-oriented declarative languages and data governance practices that facilitate efficient data management, accelerating both training and inference phases of AI models [12]. The synergy between AI4DB and DB4AI underscores the dual role that databases play in supporting AI applications while benefiting from AI-driven enhancements.

AutoML signifies the automation of machine learning processes, particularly relevant in IoT data analytics, involving the automatic selection, tuning, and updating of models to manage concept drift and ensure analytics remain accurate amid changing data patterns [5]. The application of AutoML in database systems is crucial for managing the dynamic and complex datasets characteristic of modern IoT environments.

The integration of AI and ML into DBMS holds transformative potential, enabling adaptive and efficient systems that intelligently respond to the complex demands of contemporary data-driven applications. Innovations such as multi-task training models and automated data quality frameworks demonstrate how AI enhances DBMS functionalities, optimizes query performance, and improves data quality management. These advancements streamline database administration and facilitate better decision-making by ensuring high-quality data, fostering a new generation of intelligent data systems across various industries [14, 19, 16, 17, 13].

1.4 Structure of the Survey

The survey is structured to provide a comprehensive exploration of AI-driven database management, beginning with an introduction that establishes the significance of integrating AI and ML into DBMS. This section highlights the transformative impact of these technologies on modernizing data handling frameworks. Following this, the survey delineates its objectives, emphasizing the dual perspective of AI4DB and DB4AI, and explores the integration of multi-task meta-learning frameworks and the evolution of AI-driven text-to-SQL systems.

The second major section provides a background on traditional DBMS and their evolution towards AI-driven solutions, defining core concepts such as autonomous databases, learned indexes, and query optimization, while addressing the challenges faced by traditional systems that AI solutions aim to overcome.

Subsequent sections delve into specific aspects of AI-driven database management. Section three focuses on autonomous databases, discussing their self-management capabilities and the AI-driven frameworks that support these functionalities. The fourth section examines learned indexes, highlighting their role in improving data retrieval speeds and recent innovations.

The fifth section explores query optimization through AI and ML, discussing learned models, neuro-symbolic approaches, and reinforcement learning. Section six introduces Oracle AutoML, detailing its role in automating machine learning tasks within databases and its impact on workload efficiency.

In section seven, the survey discusses workload forecasting techniques, emphasizing the limitations of traditional methods and introducing innovative models like QueryBot 5000. Section eight examines anomaly detection methods in database operations, focusing on techniques applicable to log data and specific industry contexts.

Section nine explores the concept of self-tuning databases, highlighting frameworks that enable automatic configuration adjustments for optimal performance. Finally, section ten discusses automated database administration, exploring tools and techniques that reduce manual intervention and the challenges involved in transitioning to automated systems.

The survey concludes with a summary of key points, reflecting on the current state of AI-driven database management and suggesting future research directions. This structured approach facilitates a comprehensive analysis of how AI and ML technologies are integrated into DBMS, highlighting both challenges and opportunities identified through qualitative research with industry experts. The insights gained are invaluable for researchers and practitioners, informing the development and adoption of automation tools that enhance DBMS administration and optimize query performance across various applications [17, 16, 20, 13].The following sections are organized as shown in Figure 1.

2 Background and Definitions

2.1 Traditional Database Management Systems

Traditional database management systems (DBMS) have been pivotal in data storage and retrieval, relying on structured query language (SQL) and predefined schemas to maintain data consistency and integrity [8]. These systems are characterized by a monolithic architecture that emphasizes transactional consistency, durability, and isolation, adhering to ACID (Atomicity, Consistency, Isolation, Durability) properties for reliable transaction processing.

Despite their robustness, traditional DBMS face challenges in big data and cloud computing contexts. Scalability is a significant concern, as these systems often struggle with the vast data volumes generated by modern applications, leading to performance bottlenecks [1]. The static nature of schemas further limits flexibility, complicating adaptation to evolving data requirements [8].

Traditional DBMS typically require substantial manual intervention for configuration, tuning, and maintenance, making them resource-intensive and error-prone [21]. This reliance on manual processes restricts dynamic adjustments to workload variations and real-time resource optimization [3]. Furthermore, these systems often lack the advanced analytics capabilities needed to extract insights from complex and unstructured data sources prevalent in contemporary environments [10].

The limitations of traditional DBMS underscore the need for more adaptive and intelligent solutions that integrate seamlessly with modern data ecosystems. This demand is driving the evolution of AI-driven DBMS, which leverage machine learning and automation to enhance performance, scalability, and operational efficiency. Such systems can autonomously manage complex data workloads, optimize resource allocation, and predict future demands, reducing administrative burdens and improving system responsiveness in dynamic environments [22, 16, 12].

2.2 Evolution Towards AI-Driven Solutions

The transition to AI-driven database management systems (DBMS) marks a shift from static architectures to dynamic, intelligent frameworks, driven by the increasing complexity and volume of modern data [22]. Self-driving DBMS exemplify this transformation by autonomously optimizing performance without human intervention [22].

A key advancement in this evolution is the development of learned indexes, which employ machine learning models to enhance data retrieval by adapting to data distribution and query patterns, offering superior performance over traditional indexing methods [23, 24]. AI technologies have also been integrated into query optimization, with learning-to-rank approaches addressing inefficiencies in traditional methods that rely on inaccurate absolute cost predictions [25].

AI-driven methodologies extend to multimodal data retrieval platforms, tackling challenges of exploiting diverse data sources and managing large datasets through intelligent data handling and rich query options [26]. Systems like openGauss demonstrate AI integration in enhancing database functionalities to meet modern demands for high performance, availability, security, and intelligence [27].

AI-driven solutions address limitations in handling complex queries involving multiple tables, particularly in text-to-SQL systems where efficiency has been a challenge [7]. By leveraging AI, these systems improve query processing efficiency and accuracy, expanding their real-world applicability.

The integration of AI technologies into DBMS represents a transformative advancement, addressing challenges faced by traditional systems. As AI-driven methodologies evolve, they promise to bridge the gap between theoretical performance improvements and practical deployment, enabling more intelligent, scalable, and efficient data management solutions [8].

2.3 Core Concepts and Definitions

AI-driven database management encompasses core concepts essential for integrating AI and ML technologies within database systems. Autonomous databases, learned indexes, and advanced query optimization methods each significantly enhance efficiency and performance. Autonomous databases optimize operations without human intervention, adapting dynamically to workload trends. Learned indexes use ML techniques to improve data retrieval by predicting access patterns, while advanced query optimization methods, including learned cardinality estimation, refine execution strategies by accurately modeling complex data relationships and workload characteristics. These innovations represent a transformative approach to database technology, facilitating improved resource utilization and faster query processing [28, 29, 30, 22, 31].

Autonomous databases exemplify significant advancements in database technology, characterized by self-management, self-tuning, and self-repair capabilities. By employing AI and ML algorithms, these databases automate routine tasks, minimizing manual oversight and reducing human error. This autonomy enables dynamic adaptation to changing workloads and real-time resource optimization, enhancing system performance and reliability [32].

Learned indexes offer a novel approach to data indexing, employing ML models to predict data positions within sorted arrays. Unlike static traditional indexing methods, learned indexes adapt to data distribution and query patterns, improving retrieval speeds and reducing memory consumption [33]. The Piecewise Geometric Model index (PGM-index) exemplifies this innovation by optimizing key indexing through linear models, enhancing both query time and space efficiency [23]. Open-source benchmarks and real-world datasets facilitate fair comparisons and insights into learned index performance, though challenges remain in handling real-world data distributions, which can lead to high lookup times compared to traditional methods [34].

Query optimization, a fundamental aspect of database management, aims to determine the most efficient execution plan for queries. AI and ML techniques have revolutionized this field by introducing models that predict query performance metrics and select optimal execution strategies. Cardinality estimation benefits from machine learning approaches that provide accurate predictions of query result sizes [35]. Learned models can identify the most suitable cardinality estimation model for various datasets, ensuring optimal query performance [36]. Neural networks enhance learned indexes' performance through improved time and space efficiency, advancing query optimization capabilities [37].

The convergence of AI and ML within DBMS offers significant opportunities to enhance operational efficiency and adaptability amidst increasing data complexity. By leveraging AI-driven frameworks like unified multi-task models for ML-enhanced DBMS, organizations can overcome traditional methods' limitations, which often require extensive retraining and lack generalization across tasks or databases. This shift facilitates automated data quality management, improves decision-making accuracy, and empowers non-technical users with tools like text-to-SQL systems that simplify database interactions. Consequently, AI and ML integration within DBMS is poised to create intelligent, responsive systems meeting contemporary data-driven applications' evolving demands across diverse industries [14, 19, 16, 17, 13].

2.4 Challenges in Traditional Systems

Traditional database management systems (DBMS) face numerous challenges that hinder efficient management and adaptation to rapidly evolving data environments. A primary challenge is the static nature of query optimization, leading to performance bottlenecks, particularly with user-defined functions (UDFs) where execution costs and selectivity vary significantly based on input data [3]. Fixed parameter values exacerbate this issue; deviations from assumed values can result in suboptimal query plans [38].

Cardinality estimation is another area where traditional systems struggle, as conventional methods often fail to model complex data distributions accurately, leading to inefficiencies, especially with increasing query joins that cause underestimation errors and inefficient execution plans. These systems also struggle to capture dependencies between multiple missing values, contributing to prediction inaccuracies and suboptimal results [39].

Learned indexes, while promising, present challenges, as they are primarily designed for in-memory use and do not effectively reduce block reads and writes crucial for disk operations [40]. This limitation is evident in traditional schema-on-write systems, multi-model databases, vector databases, and data lakes, which face significant difficulties managing multimodal data and executing rich hybrid queries [26].

Traditional query optimizers often make locally optimal decisions, missing globally optimal plans, especially with common subexpressions [41]. Probabilistic inference methods used in these systems either operate in exponential time or require numerous iterations to converge, making them impractical for large-scale applications [42].

Additional challenges include high read latencies, inefficient resource utilization during failover, and inadequate support for modern hardware architectures, significantly limiting traditional methods' effectiveness in meeting evolving user requirements [27]. Existing benchmarks, often based on synthetic data, fail to accurately reflect real-world system behavior, complicating the evaluation and comparison of traditional systems [9].

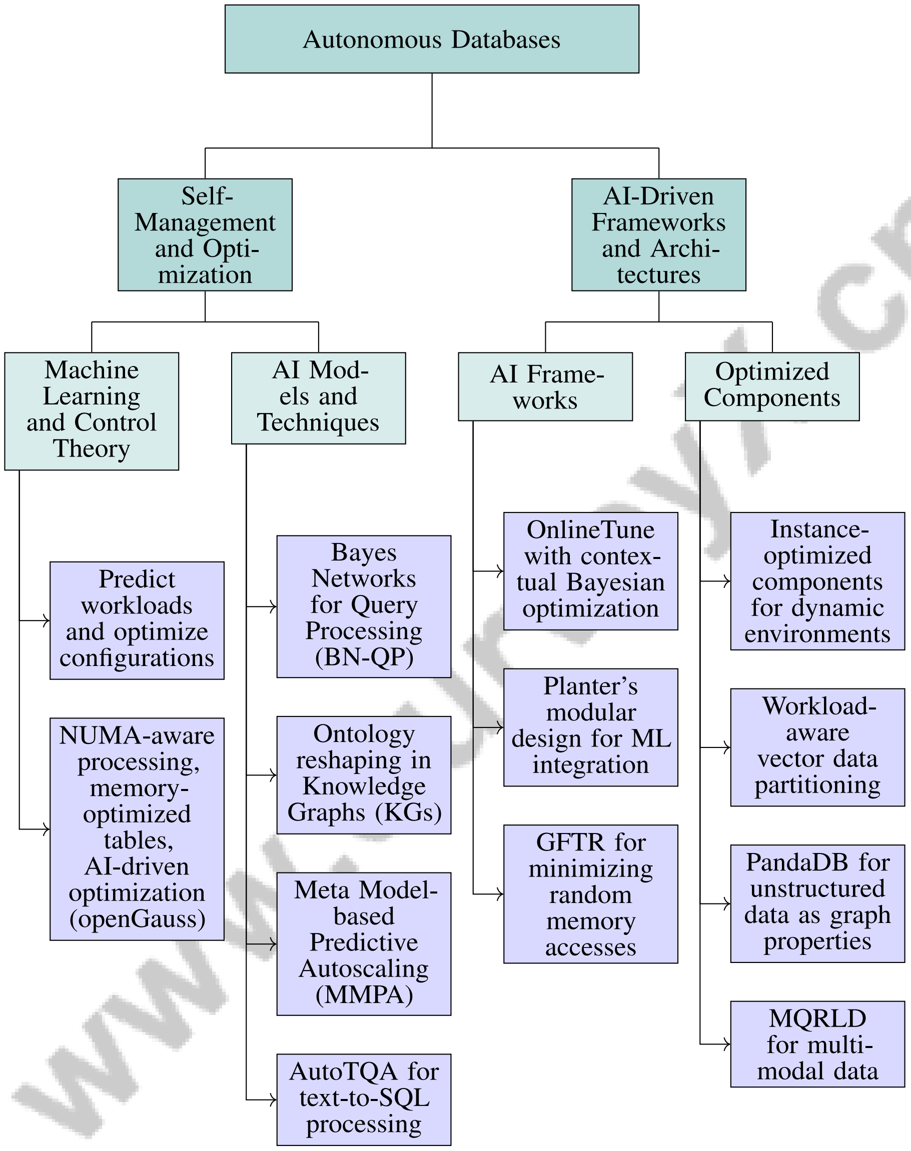
Moreover, traditional data integration methods face challenges aligning ontologies with industrial data, leading to mismatches that complicate data analytics and integration processes [10]. These challenges emphasize the urgent need for AI-driven solutions to enhance DBMS adaptability, efficiency, and intelligence, addressing traditional approaches' inherent limitations and paving the way for advanced data management frameworks.

3 Autonomous Databases

Category	Feature	Method
Self-Management and Optimization	Adaptive Learning Techniques	MMPA[43]
	Memory and Data Optimization	BN-QP[39]
	Knowledge and Query Optimization	OR[10]
AI-Driven Frameworks and Architectures	Optimization and Efficiency	OT[44], IODC[45], OG[27]
	Design and Integration	N/A[46], N/A[47]
	Data Management and Retrieval	AutoTQA[7], PDB[48], MQRLD[26], NDB[19]
	Memory and Access Optimization	GFTR[49], HQI[50]

Table 1: This table provides a comprehensive summary of various methods employed in autonomous databases for self-management and optimization, as well as AI-driven frameworks and architectures. It highlights specific features and corresponding methods, showcasing the integration of machine learning and AI techniques to enhance database performance and efficiency.

Autonomous databases signify a major advancement in data management by reducing human intervention through advanced automation and optimization. This section delves into their self-management and optimization features, highlighting the role of machine learning and control theory in predicting workloads and optimizing configurations. As illustrated in Figure 2, the hierarchical structure of autonomous databases emphasizes self-management and optimization through machine learning and AI-driven frameworks and architectures. This figure highlights the integration of machine learning models and AI techniques in optimizing database performance and efficiency, as well as the role of AI frameworks and optimized components in managing complex data environments. Table 1 presents a detailed overview of the methods utilized in autonomous databases, focusing on self-management, optimization, and AI-driven frameworks, thereby illustrating the advanced integration of machine learning and AI techniques in modern data management systems. Additionally, Table 3 presents a comprehensive comparison of methods utilized in autonomous databases, focusing on self-management, optimization, and AI-driven frameworks, thereby elucidating the integration of machine learning and AI techniques in enhancing database performance and efficiency. We explore the self-management processes crucial to these databases' operations.



Autonomous Databases
Self-Management and Optimization
AI-Driven Frameworks and Architectures
Machine Learning and Control Theory
AI Models and Techniques
AI Frameworks
Optimized Components
Predict workloads and optimize configurations
NUMA-aware processing, memory-optimized tables, AI-driven optimization (openGauss)
Bayes Networks for Query Processing (BN-QP)
Ontology reshaping in Knowledge Graphs (KGs)
Meta Model-based Predictive Autoscaling (MMPA)
AutoTQA for text-to-SQL processing
OnlineTune with contextual Bayesian optimization
Planter's modular design for ML integration
GFTR for minimizing random memory accesses
Instance-optimized components for dynamic environments
Workload-aware vector data partitioning
PandaDB for unstructured data as graph properties
MQRLD for multi-modal data

Figure 2: This figure illustrates the hierarchical structure of autonomous databases, focusing on self-management and optimization through machine learning and AI-driven frameworks and architectures. It highlights the integration of machine learning models and AI techniques in optimizing database performance and efficiency, as well as the role of AI frameworks and optimized components in managing complex data environments.

3.1 Self-Management and Optimization

Autonomous databases excel in self-management and optimization, minimizing human oversight through advanced machine learning and control theory integration. These systems predict workloads and optimize configurations proactively. For example, openGauss uses NUMA-aware processing, memory-optimized tables, and AI-driven optimization to enhance performance [27].

Machine learning models improve query planning and execution, enhancing database efficiency. The Bayes Networks for Query Processing (BN-QP) optimizes query execution by inferring missing attribute values through networked variable influences [39]. Ontology reshaping in Knowledge Graphs (KGs) simplifies query structures, improving interaction and optimizing complex industrial analytics queries [10].

Meta Model-based Predictive Autoscaling (MMPA) exemplifies AI's role in optimizing databases through reinforcement learning, ensuring dynamic adaptation to workload changes for optimal performance [43]. The multimodal data retrieval platform addresses the need for efficient handling of diverse data types and complex queries in modern systems [26].

AutoTQA leverages multi-agent large language models for efficient text-to-SQL processing across multiple tables, highlighting AI methodologies' importance in enhancing query processing [7]. These advancements illustrate how AI and machine learning empower autonomous databases to adapt to changing workloads and optimize resource usage, advancing performance, reliability, and efficiency.

3.2 AI-Driven Frameworks and Architectures

Method Name	Optimization Techniques	Integration Flexibility	Data Management Strategies
OT[44]	Contextual Bayesian Optimization	Modular Designs	Advanced Indexing
N/A[46]	Minimal Latency	Modular Framework	Advanced Indexing
GFTR[49]	Clustered Gathers	-	Virtual Identifiers
IODC[45]	Instance-optimization	Seamlessly Incorporate Algorithms	Advanced Indexing
HQI[50]	Multi-query Optimization	Workload-aware Indexing	Vector Data Partitioning
PDB[48]	Cost Model	Unified Framework	Optimized Indexing
MQRLD[26]	Learned Index	Not Applicable	Data Lake
OG[27]	Ai-driven Optimization	AI Optimization Capabilities	Memory-optimized Tables
AutoTQA[7]	Dynamic Interaction Scheduling	Multi-agent Architecture	Multi-table Tqa
NDB[19]	Dynamic Adaptation	AI Model Selection	In-database Analytics
N/A[47]	Shadow Planner	Seamless Integration	Learned Indices

Table 2: Overview of AI-Driven Frameworks and Architectures for Autonomous Databases. This table presents various methods along with their corresponding optimization techniques, integration flexibility, and data management strategies, highlighting the diverse approaches employed to enhance database performance and adaptability. The methods are referenced by their respective studies, showcasing the integration of AI and machine learning in advancing data management systems.

AI-driven frameworks and architectures are crucial for autonomous databases, enabling efficient management and optimization of complex data environments. OnlineTune, using contextual Bayesian optimization and safety assessment, minimizes unsafe recommendations, enhancing database reliability [44].

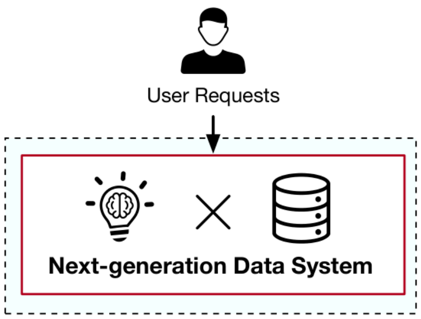
Planter's modular design allows seamless integration of machine learning algorithms, offering flexibility in managing diverse workloads [46]. The GFTR framework enhances performance by minimizing random memory accesses during materialization, improving throughput [49].

Instance-optimized components adjust for optimal performance based on datasets and workloads, maintaining efficiency in dynamic environments [45]. Workload-aware vector data partitioning and multi-query optimization significantly improve throughput, surpassing traditional methods [50].

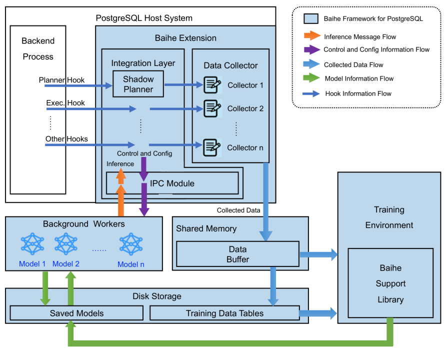
PandaDB uses advanced indexing and caching to treat unstructured data as graph properties, accelerating query execution [48]. MQRLD supports transparent storage and rich hybrid queries for multimodal data [26].

The openGauss system features NUMA-aware processing and adaptive symmetric multiprocessing for enhanced performance and security [27]. AutoTQA applies large language models for efficient multi-table user inquiry processing [7].

These frameworks and architectures enable autonomous databases to operate with minimal oversight, ensuring high performance and adaptability across complex environments. By leveraging AI and machine learning, these systems implement sophisticated optimizations, self-monitoring, and self-configuration, redefining data management and analytics with automated insights [19, 22, 8, 31]. Table 2 provides a comprehensive comparison of AI-driven frameworks and architectures, detailing their optimization techniques, integration flexibility, and data management strategies as applied to autonomous databases.



(a) User Requests and Next-generation Data System[19]



(b) A Diagram of a Machine Learning Infrastructure[47]

Figure 3: Examples of AI-Driven Frameworks and Architectures

As illustrated in Figure 3, AI-driven frameworks and architectures are integral to autonomous databases, enhancing data management and processing. The first image, "User Requests and Next-generation Data System," depicts a flowchart where user requests trigger a cycle of innovation and data storage, symbolized by a lightbulb and database icon, highlighting the interaction between user inputs and innovative solutions. The second image, "A Diagram of a Machine Learning Infrastructure," details AI architectures with a PostgreSQL host and BAIHE extension, managing data flow from inference to ensure optimal machine learning infrastructure operation. These examples show the synergy between user engagement and AI-driven systems, emphasizing their potential to revolutionize data handling and machine learning processes [19, 47].

Feature	Self-Management and Optimization	AI-Driven Frameworks and Architectures
Optimization Technique	Ai-driven Optimization	Contextual Bayesian Optimization
Integration Flexibility	Not Specified	Modular Design
Performance Enhancement	Workload Prediction	Throughput Improvement

Table 3: This table provides a comparative analysis of self-management and optimization techniques versus AI-driven frameworks and architectures in autonomous databases. It highlights key features such as optimization techniques, integration flexibility, and performance enhancement strategies, illustrating the advanced capabilities of these systems in modern data management.

4 Learned Indexes

4.1 Introduction to Learned Indexes

Learned indexes represent a paradigm shift in data indexing by employing machine learning models to predict data positions within datasets, offering a dynamic alternative to traditional indexing methods such as B-trees and binary search. Traditional indexing methods are often inefficient for dynamic or complex datasets due to their static nature, which limits adaptability to varying data distributions or query patterns [23]. Learned indexes, however, treat indexing as a machine learning problem, training models to map keys to positions in a sorted array based on rank, thus providing a flexible and adaptive indexing solution [51].

A key advantage of learned indexes is their adaptability to diverse data distributions and query workloads. The PGM-index, for example, adapts to dictionary key distributions and access frequencies, making it a distribution-aware learned index [23]. Additionally, learned indexes such as Doraemon are designed for dynamic workloads, incorporating access patterns and caching trained models to boost performance [51].

Extensions to multi-dimensional data, like LIMS, enhance query processing efficiency by clustering data and using pivot-based mapping in metric spaces [26]. Challenges in external-memory contexts, where I/O operations dominate cost structures, are addressed by approaches like AULID, which optimize on-disk learned indexes to minimize block reads and writes [26].

Despite their benefits, the learning processes of learned indexes are under-explored, lacking formal frameworks for evaluating learning objectives and effectiveness as datasets grow larger and more complex. Integrating learned indexes into database management systems marks a significant technological leap, enabling intelligent, adaptive, and efficient solutions. These predictive models can enhance search speeds by up to three times compared to traditional B+Trees and reduce memory footprint significantly. Innovations like the ALEX learned index address challenges associated with dynamic workloads, outperforming traditional methods by up to 4.1 times while maintaining a smaller index size. This evolution in indexing technology optimizes performance across various workloads and adapts to the complexities of dynamic data environments [52, 53].

4.2 Advantages of Learned Indexes

Learned indexes enhance database management by utilizing machine learning models to improve data retrieval efficiency and adaptability, surpassing traditional indexing methods. They achieve O(1) expected query time with linear space usage, making them more efficient than traditional structures like B+Trees [54]. This efficiency is beneficial in environments with dynamic and complex data distributions [23].

Their adaptability to specific data distributions results in faster and more memory-efficient indexing. Innovations like the AirIndex optimize hierarchical index designs to create heterogeneous structures that enhance query performance and adaptability [55]. Frameworks like Doraemon enhance efficiency by caching previously trained models and incrementally adapting them for similar access patterns, significantly reducing re-training times [51].

Learned indexes excel in handling diverse multimodal data types and supporting rich hybrid queries, as demonstrated by the MQRLD platform, which optimizes retrieval through a learned index [26]. The AULID design introduces a B+-tree-styled leaf node layout and a novel inner node structure, significantly reducing I/O costs compared to existing learned indexes, making them suitable for disk-based operations [40].

In dynamic environments, frameworks like UpLIF utilize reinforcement learning to adaptively self-tune model structures in response to updates, ensuring sustained optimal performance over time [56]. This adaptability is crucial for maintaining high performance without manual intervention. Learned indexes like WaZI adapt to varying query workloads, reducing query latencies and improving data retrieval efficiency [57].

The adoption of learned indexes in database management systems offers efficient, scalable, and adaptable solutions, addressing limitations of traditional indexing methods and meeting modern data environment demands. By considering factors such as memory and disk utilization, learned indexes provide a comprehensive approach to managing complex data distributions [6].

4.3 Challenges and Limitations

Implementing learned indexes in database management systems presents challenges and limitations affecting performance and applicability. A significant challenge is the dependency on probabilistic assumptions regarding data distribution, which may not hold in all scenarios, potentially leading to suboptimal performance [54]. This reliance complicates their use in environments with highly dynamic and unpredictable data distributions.

A core obstacle is the computational complexity of retraining, which increases linearly with the number of keys and their lengths, causing performance bottlenecks [58]. This complexity is pronounced in dynamic environments with frequent data updates, necessitating constant retraining to maintain accuracy. Moreover, current benchmark methodologies focus on read-only workloads and do not account for mixed read/write scenarios, limiting applicability in real-world applications [34].

Challenges in disk-based learned indexes like AULID arise with high write workloads, where I/O overhead may increase due to frequent structural modifications [40]. Inaccuracies in block size predictions can also lead to performance bottlenecks [59].

Managing learned index structures, as seen in MQRLD, requires careful tuning based on varying query workloads, which can be resource-intensive [26]. The method's reliance on cached model accuracy may not always align with new data distributions [51]. AutoIndex's focus on read-only indexes may not effectively address write-oriented scenarios, limiting utility in environments requiring frequent data modifications [60].

Key challenges include defining effective error-correction mechanisms for mispredictions and arranging data layouts for effective learning by machine learning models [61]. These challenges highlight the need for ongoing research to enhance learned indexes' adaptability, efficiency, and robustness, enabling them to fully realize their potential in diverse and dynamic data environments.

4.4 Innovations and Hybrid Approaches

Recent innovations in learned indexes have improved their adaptability, efficiency, and applicability across various data environments. The Shift-Table method exemplifies advancements in optimizing learned index structures, achieving up to a threefold performance improvement on real-world datasets compared to existing methods [62]. This method highlights the potential for enhancing learned indexes to achieve superior performance in practical applications.

Hybrid approaches are being explored, combining the strengths of learned indexes with traditional indexing methods to address their respective limitations. These hybrid models leverage the adaptability of learned indexes while retaining the reliability of traditional indexes, offering a balanced solution for complex data environments [61]. Future research is expected to focus on these hybrid structures, particularly in disk-resident settings, to optimize performance and resource utilization [52].

Integrating reinforcement learning into learned index structures is another innovative approach. UpLIF dynamically optimizes index structures through balanced model adjustment and reinforcement learning, ensuring optimal performance in dynamic environments [56]. This integration allows continuous adaptation to changing data patterns and workloads, enhancing overall database operation efficiency.

Frameworks proposed by Li et al. demonstrate significant improvements in both learning efficiency and effectiveness of learned indexes, achieving up to 78x construction speedup and up to 1.59x query speedup [63]. Such advancements mark a significant step forward in enhancing scalability and reducing overhead, particularly in dynamic data environments where frequent updates are necessary.

The experimental setup of MQRLD includes comparative analyses against other methods for rich hybrid queries, utilizing various multimodal datasets to assess query performance [26]. This highlights the potential of learned indexes to support diverse data types and complex query environments.

Future research directions include optimizing learned indexes for external-memory contexts and exploring hybrid approaches that blend learned and traditional indexing methods to leverage their strengths [34]. These innovations and hybrid approaches collectively advance the field of learned indexes, offering promising solutions for more efficient, scalable, and adaptable data management systems. As research continues to explore these areas, the potential for learned indexes to transform database management practices becomes increasingly evident, paving the way for more intelligent and responsive data systems.

5 Query Optimization

5.1 AI and Machine Learning in Query Optimization

The integration of artificial intelligence (AI) and machine learning (ML) into query optimization has revolutionized database management systems (DBMS), enhancing the adaptability and efficiency of data querying. Traditional optimization methods, dependent on static heuristics and fixed cost models, fall short in handling complex queries with multiple joins and intricate selection conditions [29]. In contrast, AI and ML introduce dynamic learning mechanisms and adaptive tuning capabilities, significantly improving query optimization.

Learned query optimizers (LQOs) exemplify this evolution by leveraging deep learning to capture complex data patterns, thus generalizing across unseen queries [36, 4]. By utilizing historical query execution data, LQOs can predict accurate execution plans, enhancing both efficiency and performance. Methods like the PGM-index use linear models to approximate key positions in sorted arrays, facilitating efficient data retrieval and improving query optimization [23]. In industrial analytics, ontology reshaping aligns data representations with analytical needs, optimizing query execution [10].

The COOOL framework applies learning-to-rank techniques, significantly reducing execution latencies and minimizing performance regressions compared to existing methods [25]. This approach ensures the selection of the most efficient plans, enhancing overall DBMS performance. The integration of AI and ML into query optimization represents a groundbreaking evolution in DBMS, enhancing intelligence and adaptability through techniques such as reinforcement learning and large language model (LLM) embeddings [28, 3, 20, 29]. As research progresses, AI and ML's role in optimizing data querying is poised to expand, promising significant enhancements in efficiency, scalability, and adaptability.

5.2 Learned Models and Cardinality Estimation

Learned models have become essential in enhancing cardinality estimation, a crucial aspect of query optimization in DBMS. Traditional techniques often falter with complex data distributions and query patterns, leading to inefficient execution plans. In contrast, learned models utilize advanced ML techniques to provide dynamic and adaptable solutions, significantly improving accuracy and efficiency, especially in scenarios involving data skew and correlated predicates [64, 65, 66, 30].

The Monotonic Cardinality Estimation of Similarity Selection (MCESS) method employs a deep learning-based regression model to enhance cardinality estimation, thus improving query optimization efficiency [4]. BitE (Bias-tuning Ensemble Model) leverages database statistics to classify workloads, optimizing query execution by distinguishing between Light and Heavy workloads [67]. Despite advancements, query-driven learned cardinality estimators often struggle with out-of-distribution queries, necessitating domain knowledge integration to enhance prediction accuracy [25]. The Mixed Approach for Query Optimization (MAQO) combines learned models with traditional techniques to improve execution plans, ensuring accurate and reliable cardinality estimates [68].

The COOOL framework reinforces learning-to-rank techniques, achieving significant performance improvements by transforming cost estimation into predicting relative cost orders, ensuring stable and robust query performance in dynamic environments [25]. The incorporation of learned models into cardinality estimation marks a significant advancement in query optimization, effectively accounting for complex factors often overlooked by traditional methods. Despite challenges in dynamic environments, ongoing research aims to refine these models for reliable deployment in modern DBMS, enhancing adaptability and efficiency [64, 65, 30].

5.3 Neuro-symbolic and Hybrid Approaches

Neuro-symbolic and hybrid approaches in query optimization represent a transformative advancement in DBMS, merging symbolic reasoning and neural networks to enhance adaptability and efficiency. These approaches combine machine learning's predictive capabilities with symbolic methods' logical precision, resulting in flexible frameworks adept at navigating large-scale data environments, such as knowledge graphs. By integrating neural models that capture non-linear relationships with traditional symbolic reasoning, these hybrid systems address standard methods' limitations that often misestimate in complex datasets [69, 70]. The Fully Observed Optimizer (FOOP) exemplifies this integration, utilizing deep reinforcement learning (DRL) to create a data-adaptive optimizer that enhances execution efficiency by avoiding exhaustive enumeration of join orders.

Parallel processing capabilities further enhance query optimization by facilitating efficient exploration of large plan spaces, reducing optimization time, and improving query plans' quality, leading to faster execution of complex queries in relational DBMS [71, 72, 20, 73, 74]. The MPDP algorithm exemplifies this by effectively pruning the search space and leveraging cluster resources to expedite query processing.

The FactorJoin method highlights hybrid approaches' advantages in estimating join cardinalities, significantly outperforming traditional and contemporary learning-based techniques by combining classical join histogram methods with learning-based techniques, achieving faster estimation times and smaller model sizes while maintaining accuracy [75, 76, 77]. Integrating domain knowledge into learned cardinality models, as seen in CORDON, enhances accuracy and robustness by introducing differentiable constraints, improving query execution plans' performance [41, 30].

The COOOL framework exemplifies learning-to-rank techniques in query optimization by framing cost estimation as predicting relative cost orders, enhancing stability and robustness against execution latencies variations [78, 79, 80]. Despite progress, challenges remain, such as reliance on well-tuned cost models impacting optimizer performance. LEC query optimization seeks to minimize expected execution costs by accounting for relevant parameter distributions [38], while BitE enhances optimization by adapting its approach based on workload complexity [67].

Neuro-symbolic and hybrid approaches in query optimization offer promising advancements in enhancing adaptability, efficiency, and accuracy in DBMS. As research evolves in innovative methodologies like Retrieval-Augmented Generation (RAG) and learned cardinality estimation, their impact on query optimization processes is expected to grow, leading to more intelligent and responsive data systems capable of efficiently handling complex queries and optimizing execution plans. Techniques such as RAG have shown potential in improving language models' accuracy by guiding them to retrieve factual information, while optimization methods for large-scale knowledge graphs have demonstrated substantial speed improvements in query processing [81, 30, 82, 83].

5.4 Reinforcement Learning and Adaptive Optimization

Reinforcement learning (RL) has emerged as a transformative approach in adaptive query optimization, enabling DBMS to dynamically refine and enhance performance through interactions with their environment. This methodology allows databases to manage fluctuating workloads and complex data environments by continuously adjusting optimization strategies based on real-time feedback. MADB employs independent learning, allowing multiple agents to adapt their policies based on shared observations while pursuing individual performance goals [84].

Frameworks like Hydro illustrate RL's application by introducing an adaptive query processing (AQP) framework designed for machine learning queries, allowing for dynamic reordering of predicate evaluations and improved resource allocation, thereby enhancing execution efficiency [3]. The challenge of high training and inference costs associated with learned methods can impede real-time applicability; addressing this requires developing more efficient algorithms and exploring hybrid models that combine neural and symbolic strengths [85]. The Neo system suggests enhancing row vector encoding and transfer learning for unseen schemas to improve RL models' adaptability and scalability [86].

Innovative methods like FOOP utilize deep reinforcement learning to adaptively optimize join orders without exhaustive searches, significantly reducing optimization time [87]. Additionally, the CardIndex method exemplifies RL integration with other optimization strategies by combining learned indexing and cardinality estimation into a lightweight structure, reducing latency and storage needs [88]. This integration underscores RL's potential to enhance query performance and resource efficiency, particularly when combined with advanced indexing techniques.

The effectiveness of learned query superoptimization highlights RL's capacity to improve performance in repetitive workloads, presenting a promising avenue for achieving significant efficiency gains [2]. RL integration in adaptive query optimization is revolutionizing DBMS by enhancing adaptability, efficiency, and accuracy. Recent advancements indicate that machine learning techniques, especially deep reinforcement learning and specialized algorithms like Bao, empower query optimizers to learn from past errors and adapt to evolving workloads and data structures. These innovative strategies not only enhance performance metrics—such as execution speed and tail latency—but also leverage insights from large language models (LLMs) to guide decision-making processes. Consequently, RL application in this domain promises to streamline query optimization, leading to more effective and responsive database management solutions [89, 90, 2, 29]. As research evolves, these methodologies are anticipated to play an increasingly pivotal role in transforming query optimization processes, paving the way for more intelligent and responsive data systems.

6 Oracle AutoML and Automated Machine Learning

Oracle AutoML introduces a significant transformation in managing data-driven insights by enhancing DBMS with machine learning techniques. It addresses traditional ML limitations in DBMS, such as task-specific optimization and data retraining challenges [91, 16, 92, 13]. Understanding its core functionalities is crucial for optimizing performance across various applications.

6.1 Introduction to Oracle AutoML

Oracle AutoML advances the automation of ML tasks in database systems by streamlining model selection, tuning, and deployment. Utilizing Oracle's infrastructure, it reduces human intervention, particularly in large-scale data environments like IoT and DBMS. Traditional ML workflows face challenges in model selection and tuning, compounded by dynamic data leading to concept drift. AutoML simplifies these processes, enhancing efficiency and reducing reliance on skilled data scientists [5, 93, 16, 14].

A key feature of Oracle AutoML is automated feature selection and hyperparameter tuning, crucial for optimizing model performance. Advanced algorithms navigate the parameter space to enhance predictive accuracy in applications such as anomaly detection and query optimization [93, 66, 20, 94, 82]. This accelerates model development and ensures robustness tailored to specific data characteristics.

Oracle AutoML's integration with Oracle's database systems allows direct application of ML models to stored data, facilitating real-time analytics and decision-making while enhancing data security by eliminating the need for data export [22, 16, 14, 13]. It incorporates mechanisms for continuous model monitoring and updating, ensuring adaptability to evolving data patterns and addressing concept drift challenges, enhancing efficiency and accuracy in dynamic environments like IoT systems [5, 16, 92, 13].

Oracle AutoML provides a comprehensive approach to automating ML workflows, equipping users to manage the entire ML lifecycle within Oracle's ecosystem. By addressing complexities in model development and deployment, it enables organizations to leverage data effectively, fostering innovation and actionable insights, particularly in DBMS where ML integration enhances task efficiency [16, 13].

6.2 Automation of Machine Learning Tasks

Oracle AutoML enhances ML task automation in database systems by streamlining model selection, hyperparameter tuning, and updating processes. This is crucial in dynamic environments like IoT, where adaptability and efficiency are vital [5]. Sophisticated algorithms identify optimal model configurations, enhancing predictive accuracy without extensive human intervention.

Oracle AutoML's automation capabilities facilitate continuous monitoring and updating of models, addressing challenges like concept drift. This ensures model accuracy and relevance, reducing manual effort in complex systems like IoT and DBMS [5, 16, 92, 13]. Automation accelerates the model development cycle and reduces computational overhead associated with manual tuning.

Oracle AutoML's integration with Oracle's database infrastructure allows direct application of ML models to stored data, optimizing database management tasks and enhancing processing efficiency. This enables real-time analytics and decision-making, improving operations and decision-making across various database scenarios [16, 20, 13].

6.3 Integration with Database Systems

Integrating Oracle AutoML into DBMS advances ML workflow automation in data-centric environments, enhancing data processing and analysis efficiency. This integration enables real-time analytics and decision-making by minimizing data transfer latency. Recent advancements, such as MTMLF, enhance these systems by capturing transferable knowledge across tasks and databases, addressing traditional ML limitations [16, 13].

A key advantage is executing ML models directly on stored data, ensuring data security and integrity. This accelerates model execution, enhances efficiency, and strengthens data governance. This is essential given the increasing complexity and volume of data in modern DBMS, allowing for automated tools that optimize performance and predict workload trends without human intervention [95, 22, 16, 14].

AutoML's integration optimizes model selection and tuning tailored to specific datasets, enhancing performance and addressing challenges like extensive retraining. This ensures precise and actionable insights, improving data-driven decision-making efficiency in complex environments [16, 20, 13].

Moreover, AutoML's seamless integration with DBMS facilitates continuous monitoring and updating of models, allowing adaptation to evolving data patterns. Transfer learning techniques, like multi-task training, optimize model performance across tasks and databases, reducing retraining needs. This adaptability is critical in IoT-driven environments, where concept drift impacts model efficacy [5, 13].

Oracle AutoML's integration into DBMS provides a comprehensive framework for automating ML processes, addressing data complexity in organizations. This streamlines tasks and leverages advanced techniques for proactive optimization, improving data management efficiency [20, 16, 96, 22, 13]. Embedding ML capabilities within the database infrastructure enhances analytics, drives innovation, and supports informed decision-making across applications.

6.4 Impact on Workload Efficiency

Oracle AutoML's integration into DBMS enhances workload efficiency by automating and optimizing ML processes. Automating model selection, tuning, and deployment reduces manual intervention, streamlining the ML lifecycle. This is crucial in dynamic environments like IoT, where rapid data generation necessitates effective model management. AutoML frameworks intelligently select and optimize models, addressing concept drift challenges to ensure robust performance. AI-driven data quality management ensures accurate and reliable data, leading to effective ML applications across domains [5, 81, 93, 14]. Oracle AutoML efficiently explores parameter space to identify optimal configurations, ensuring robustness for specific data characteristics.

Oracle AutoML's impact on workload efficiency is evident in its capability to continuously monitor and update models, ensuring accuracy as data evolves. This adaptability is essential in IoT environments, where sensor data requires frequent processing and analysis. Addressing concept drift is crucial, highlighting the demand for AutoML solutions that autonomously select, tune, and update models to maintain optimal performance [5, 14]. Automating model recalibration reduces computational overhead, enhancing overall system efficiency.

Furthermore, Oracle AutoML's integration with database systems facilitates real-time analytics and decision-making by enabling direct execution of ML models within the database. This reduces latency and enhances efficiency by allowing real-time data analysis, eliminating the need for data export. This integration streamlines workflows and improves data-driven decision accuracy by leveraging AI-driven frameworks for data quality management and anomaly detection [78, 80, 14, 97]. The interaction between ML models and database infrastructure allows efficient data processing and analysis, driving insights and innovation across domains.

Oracle AutoML's impact on workload efficiency is exemplified by Meta Model-based Predictive Autoscaling (MMPA) in Alipay's cloud environment, achieving approximately 50% resource savings compared to traditional methods [43]. This resource reduction underscores Oracle AutoML's potential to optimize resource allocation and improve workload efficiency in large-scale data environments.

Oracle AutoML's automation and optimization capabilities enhance workload efficiency, enabling organizations to leverage data while maintaining security and governance. By improving ML workflows and resource allocation, Oracle AutoML empowers organizations to make data-driven decisions and foster innovation across diverse applications, addressing data management complexities in a rapidly evolving technological landscape [5, 92, 20, 16, 13].

7 Workload Forecasting

7.1 Limitations of Traditional Forecasting Techniques

Traditional forecasting methods in database management often fall short in predicting resource usage due to their inability to handle the complexities of multi-resource environments and real-time workload fluctuations. This inadequacy leads to inefficient resource allocation, machine underutilization or overloading, and Quality-of-Service (QoS) violations, hindering cloud computing efficiency. Enhanced workload forecasting models, particularly those leveraging machine learning, are essential to address these challenges [14, 98]. The high variability and dimensionality of cloud workloads further complicate predictions.

These conventional methods typically employ static models that fail to utilize historical data and system call sequences for accurate, context-aware predictions, lacking the granularity needed for effective resource allocation [99]. Their adaptability is limited, rendering them ineffective in rapidly changing data patterns of modern environments.

Moreover, traditional techniques often rely on predefined assumptions about data distributions, which may not reflect real-world scenarios, leading to ineffective predictions. They focus on average trends, neglecting atypical data points crucial for understanding complex systems. The challenges posed by big data—its volume, velocity, and variety—underscore the need for AI-driven frameworks capable of adapting to dynamic environments and offering nuanced analyses of underlying patterns and anomalies [69, 14, 78, 99, 64]. Advanced forecasting methods incorporating machine learning and AI technologies are vital for enhancing prediction accuracy and adaptability.

7.2 Introduction to QueryBot 5000

QueryBot 5000 represents a significant advancement in workload forecasting by leveraging historical data and query logic to predict query arrival rates in dynamic environments [100]. This approach addresses the limitations of traditional techniques that struggle with the variability and complexity of modern workloads.

QueryBot 5000's predictive capabilities arise from its ability to analyze and categorize queries based on their logical structures, facilitating accurate and context-aware forecasts. This aligns with employing machine learning models to improve resource management efficiency in cloud environments, as evidenced by recent advancements [101]. By integrating these models, QueryBot 5000 anticipates workload fluctuations, optimizing resource allocation and system performance.

Complementing QueryBot 5000, systems like Alibaba Workload Miner (AWM) classify queries by business logic and identify real-time patterns to optimize query processing [102]. This synergy enhances QueryBot 5000's precision in workload forecasting, contributing to more efficient resource management in cloud-based systems.

QueryBot 5000 exemplifies the transformative potential of machine learning-driven solutions in workload forecasting, offering a robust framework for managing dynamic and complex data environments [98].

7.3 Query Clustering and Forecasting Models

Query clustering and forecasting models enhance workload prediction by categorizing similar queries and applying predictive techniques. The QB5000 framework exemplifies this by predicting expected query arrival rates through clustering and forecasting models [100]. This method leverages query logic to deliver accurate forecasts, addressing limitations of traditional techniques in dynamic environments.

Integrating machine learning models into workload forecasting enhances prediction precision and adaptability. A comprehensive survey categorizes research into five machine learning model classes: Evolutionary, Deep, Hybrid, Ensemble, and Quantum Learning [101]. Each offers unique strengths; for instance, deep learning captures complex patterns in workloads, while ensemble methods enhance accuracy by combining models.

Systems like Alibaba Workload Miner (AWM) enhance query clustering by processing streaming query logs to discover and optimize workload patterns in real-time [102]. This allows for dynamic resource allocation adjustments, improving performance and efficiency. By merging query clustering with advanced forecasting models, these approaches provide robust solutions for managing modern workload variability and complexity.

Query clustering and forecasting models significantly advance database management, improving accuracy and adaptability in resource allocation and performance. As research progresses, machine learning techniques will increasingly optimize workload forecasting and enhance efficiency in data-driven applications, particularly within cloud computing. These techniques predict future workloads, addressing challenges like underutilization, machine overloading, and QoS violations. By integrating comprehensive system telemetry and employing models like recurrent neural networks, researchers aim to develop sophisticated approaches for resource usage prediction and anomaly detection, leading to more effective cloud resource management and enhanced application performance [98, 99].

Benchmark	Size	Domain	Task Format	Metric
RBM[103]	500,000	User Interaction Analysis	Multi-Modal Classification	F1-score, Accuracy
FPTP[80]	100,000	Database Management	Query Optimization	Accuracy, Performance Impact
SOSD[104]	2,000,000	Data Structures	Search Query	Mean Query Time
LogEval[92]	4,000	Log Analysis	Log Parsing	F1-score, Accuracy
AIOps[9]	1,000,000	Kpi Anomaly Detection	Anomaly Detection	F1-score
E2E-Benchmark[36]	6,191	Database Optimization	Query Optimization	Execution Time, Inference Time

Table 4: This table presents a selection of representative benchmarks used in evaluating database management systems and related applications. It includes details on benchmark size, domain, task format, and the metrics used for performance assessment, providing a comprehensive overview of the diverse evaluation criteria employed in real-world scenarios.

7.4 Evaluation and Real-World Application

Evaluating workload forecasting techniques in real-world applications highlights their critical role in enhancing database management system efficiency and adaptability. QueryBot 5000 has demonstrated significant efficacy in predicting future workloads with minimal accuracy loss, enabling self-driving DBMSs to optimize performance proactively [100]. This predictive capability facilitates dynamic resource allocation and improves system responsiveness, addressing challenges posed by fluctuating workloads.

A comprehensive analysis of forecasting techniques reveals research gaps and emphasizes accurate workload predictions' importance [98]. Integrated approaches considering multiple resources enhance resource management strategies' effectiveness. Advanced machine learning models, such as Quantum Neural Networks, further optimize forecasting techniques, providing reliable and interpretable predictions [101].

In real-world applications, integrating forecasting models into DBMSs enables informed decision-making, allowing systems to anticipate and adapt to workload changes efficiently. This adaptability is crucial for maintaining optimal performance in cloud environments characterized by high variability and complexity. Optimizing machine learning models and incorporating Explainable AI principles can enhance workload forecasts' reliability and interpretability, driving innovation and efficiency in data-driven applications [101].

Evaluating workload forecasting techniques in practical scenarios demonstrates their potential to enhance DBMS efficiency by enabling autonomous future query arrival rate predictions and proactive system performance optimization. QueryBot 5000, for instance, uses historical data and logical query compositions to forecast workloads, facilitating timely index selections and reducing manual tuning by database administrators. This innovative approach streamlines system management, minimizing reliance on costly exploratory testing and reactionary measures, transforming modern DBMS operational landscapes [96, 100]. As research advances, these techniques will play an increasingly pivotal role in enhancing modern data environments' adaptability and efficiency, ensuring effective workload management. Table 4 offers a detailed overview of various benchmarks employed in the evaluation of workload forecasting techniques and their applications in database management systems.

8 Anomaly Detection

8.1 Introduction to Anomaly Detection in Databases

Anomaly detection is critical in database management systems for identifying irregularities that may indicate system malfunctions, security breaches, or data integrity issues. Modern data environments' complexity, with high dimensionality and diverse data types, requires sophisticated methodologies to effectively address data anomalies. In financial auditing, machine learning algorithms are increasingly used to detect irregularities within heterogeneous datasets. Recent studies suggest that Large Language Model (LLM) embeddings enhance anomaly detection in financial records by encoding non-semantic categorical data, improving model performance by addressing challenges related to feature sparsity and heterogeneity [69, 105].

Integrating Knowledge-Based Temporal Abstraction (KBTA) with temporal pattern mining significantly advances anomaly detection, identifying normal behavior patterns to detect deviations indicative of anomalies [106]. This framework is particularly beneficial where temporal data is crucial for understanding system behavior.

In log data analysis, anomaly detection is essential for maintaining system reliability. A recent taxonomy categorizes log data anomalies into point anomalies, such as Template and Attribute Anomalies, and contextual anomalies, providing a structured methodology for addressing these issues in complex IT systems [107, 108]. Anomalies in computational workflows can reveal underlying hardware or system process issues, necessitating sophisticated detection techniques to navigate feature dimension heterogeneity and sparsity, especially in general ledger data [93, 105].

At an industrial scale, anomaly detection methods effectively identify cases arising from distinct mechanisms compared to the majority of the dataset, highlighting their capability in large-scale environments [109]. These techniques are crucial for interpreting atypical data within extensive datasets, a challenge often overlooked in traditional statistical analyses [69]. Developing advanced anomaly detection techniques is essential for enhancing database system reliability and security, enabling proactive identification and resolution of anomalies to foster resilient and intelligent database management solutions, ultimately improving operational efficiency and decision-making in complex data environments [110, 69, 14, 111, 16].

8.2 Techniques and Frameworks for Anomaly Detection

Anomaly detection in database management systems is crucial for identifying irregularities that may indicate security breaches, data integrity issues, or system malfunctions. Advanced techniques, including machine learning and deep learning, are employed to analyze log data and financial records effectively. Deep learning models often outperform traditional data mining methods, particularly in detecting contextual anomalies. LLMs have shown promise in enhancing anomaly detection capabilities in financial data by addressing feature sparsity and improving model performance. A comprehensive anomaly detection framework safeguards data integrity and optimizes system performance and user experience in database management [107, 110, 69, 105].

Knowledge-Based Temporal Abstraction with Temporal Pattern Mining (KBTA-TPM) effectively identifies anomalies in time-oriented data by combining KBTA with temporal pattern mining [106]. This approach is particularly effective in environments where temporal data patterns are fundamental for understanding normal system behavior.

The ADLILog framework exemplifies an unsupervised log-based anomaly detection method, utilizing log instructions from public code projects alongside target system logs to train a deep learning model [108]. This method addresses the challenge of anomaly detection in log data without requiring labeled data, which can be costly and time-consuming to generate [107].

Innovative approaches, such as encoding non-semantic categorical data into fixed-size dense vectors using pre-trained sentence-transformer models, have also emerged as effective techniques for anomaly detection, particularly in financial journal entries [105]. Traditional rule-based systems and machine learning techniques often necessitate extensive preprocessing and expert knowledge, limiting their effectiveness in detecting novel anomalies [93]. Scalable methods capable of handling heterogeneous data types are increasingly adopted for industrial applications [109]. These methods enhance the robustness and scalability of anomaly detection systems.

Furthermore, the principle that atypical data can be described with fewer bits than typical data informs certain anomaly detection methods, suggesting the potential of data compression techniques for identifying irregularities [69]. Advancing techniques and frameworks for anomaly detection is essential for maintaining database system reliability and security. By systematically identifying and resolving anomalies, these methods enhance the integrity and performance of data-driven applications, fostering resilient and intelligent database management solutions to address the complexities of modern data environments [69, 111, 16, 14].

8.3 Anomaly Detection in Log Data

Anomaly detection in log data is critical for identifying irregularities that may indicate potential issues such as security breaches, data integrity problems, or system malfunctions. The complexity of contemporary data environments, characterized by high-dimensional and diverse log data, necessitates advanced methodologies to manage and analyze this complexity effectively. These methodologies must address challenges in symbolic data representation within SQL queries while leveraging statistical techniques like feature mapping, kernel methods, and Bayesian models to unlock log data's potential for practical applications like query optimization and user experience enhancement. A well-defined taxonomy of log data anomalies is essential for selecting suitable algorithms, particularly as deep learning approaches have demonstrated superior performance in identifying contextual anomalies. Integrating approximate computation techniques can enhance big data analytics efficiency without compromising effectiveness [107, 110, 78].

The ADLILog framework exemplifies an unsupervised log-based anomaly detection method, employing a two-phase learning procedure that uses log instructions from public code projects to train a deep learning model [108]. This approach effectively addresses the challenge of detecting anomalies in log data without requiring labeled data, often costly and time-consuming to create.

Evaluating anomaly detection methods typically involves using real-world data to derive normal patterns and detect deviations. For example, Shabtai et al. utilized data collected from a real server over two weeks, employing the first week to establish normal patterns and the second to identify anomalies [106]. This underscores the importance of realistic data environments in assessing the effectiveness of anomaly detection techniques.

Deep learning-based methods, such as DeepLog and A2Log, have outperformed traditional data mining techniques in detecting anomalies across various types, with template anomalies being the easiest to predict [107]. The comparative analysis indicates the potential of deep learning models to provide more accurate and reliable anomaly detection in log data.

In industrial applications, the computational intensity of processes like Singular Value Decomposition (SVD) presents challenges for scaling with large datasets [109]. Addressing these scalability issues is crucial for implementing effective anomaly detection systems in large-scale environments. Innovative methods that define atypicality axiomatically offer a unique perspective on anomaly detection, distinguishing themselves from traditional techniques reliant on likelihood measures [69].

Developing advanced methods for anomaly detection in log data is essential for ensuring the reliability and security of database systems. By effectively identifying and addressing anomalies within database management systems (DBMS), these techniques uphold the integrity and performance of data-driven applications while enhancing the resilience and intelligence of database management solutions. This is particularly crucial in the context of increasing data complexity and volume, as automation tools like PinSQL demonstrate their effectiveness in real-time anomaly detection and root cause analysis, streamlining performance issue resolution in cloud environments. Innovative methodologies, such as universal source coding and advanced statistical techniques for query analysis, pave the way for more efficient and automated DBMS administration, ultimately fostering improved operational outcomes and user experiences [69, 111, 16, 110].

8.4 AI Techniques for Anomaly Detection in Financial and Industrial Data

Applying artificial intelligence (AI) techniques for anomaly detection in financial and industrial data has significantly improved the accuracy and efficiency of identifying irregularities that may indicate issues like fraud, operational failures, or system inefficiencies. In financial contexts, leveraging large language models (LLMs) for embedding financial data has resulted in substantial enhancements in anomaly detection capabilities. These embeddings capture complex patterns and relationships within the data, significantly outperforming traditional methods and providing a deeper understanding of financial anomalies [105].

In industrial settings, AI-driven methodologies have proven effective in identifying atypical patterns within large datasets. Techniques such as Knowledge-Based Temporal Abstraction (KBTA) combined with temporal pattern mining have successfully detected anomalies by identifying time intervals that deviate from established normal patterns [106]. This approach is particularly valuable in environments where temporal data is critical for understanding operational behaviors and detecting potential disruptions.

Using log data for anomaly detection in both financial and industrial domains has been enhanced by advanced AI techniques. The ADLILog framework, for example, leverages log instructions from public code projects to train deep learning models, significantly outperforming both supervised and unsupervised methods regarding the F1 score [108]. This framework addresses the challenge of detecting anomalies in log data without relying on costly labeled data, thus enhancing the scalability and applicability of anomaly detection systems.

Moreover, adapting LLMs through supervised fine-tuning and in-context learning allows for direct anomaly detection in log files generated during computational workflows, providing a powerful tool for identifying irregularities in complex data environments [93].

Future research in anomaly detection should focus on refining classification methods, exploring additional anomaly types, and improving the understanding of currently unclassifiable log messages [107]. Developing methods that successfully identify atypical subsequences in large datasets lays a solid theoretical foundation for future applications in data analysis, ensuring that AI-driven anomaly detection techniques continue to evolve and adapt to the complexities of modern data environments [69].

Integrating AI techniques into anomaly detection frameworks significantly enhances the management of financial and industrial data by improving the accuracy and efficiency of anomaly detection processes. Recent advancements utilizing LLMs have shown promise in identifying irregularities in financial records, outperforming traditional machine learning models by effectively addressing feature sparsity. Furthermore, AI-driven frameworks have been developed to enhance data quality in big data ecosystems, focusing on comprehensive quality anomaly detection and correction methods, which are crucial for maintaining the integrity of data-driven decision-making. Innovative approaches, such as the Knowledge-Based Temporal Abstraction method, have been introduced to analyze temporal data and identify abnormal patterns through intelligent interpretation. Collectively, these advancements underscore the transformative potential of AI in refining data analysis capabilities across various domains [69, 106, 105, 14]. As research progresses, these methodologies are expected to play an increasingly pivotal role in ensuring the reliability and security of data-driven applications across diverse fields.

9 Self-Tuning Databases

9.1 Frameworks and Methods for Self-Tuning

Self-tuning databases mark a transformative advancement in database management by dynamically adjusting configurations to optimize performance. These systems leverage machine learning algorithms, optimization techniques, and adaptive methodologies to analyze telemetry data, such as resource usage metrics and application interactions, enabling precise predictions of future resource demands. This facilitates effective workload management and real-time resource allocation, addressing issues like underutilization, overloading, and quality-of-service violations in cloud environments. Techniques such as recurrent neural networks enhance adaptability to changing workloads, significantly improving system efficiency [6, 98, 99].

The TGD-rewrite method exemplifies self-tuning by optimizing queries through factorization and rewriting, thus minimizing redundancy and enhancing processing efficiency [112]. Integrating anomaly detection is crucial for maintaining performance, as demonstrated by Caithness et al., who identify anomalies through dataset partition analysis and joint distribution evaluations [109]. This integration prevents performance degradation and ensures efficient resource use.

Self-tuning databases also employ reinforcement learning and predictive modeling to anticipate workload patterns and adjust configurations accordingly. Techniques such as semantic query optimization, learned indexing, and autonomous systems enable efficient management of dynamic data environments. For instance, approximate query answering provides timely responses under challenging conditions, while learned indexes like ALEX optimize data retrieval and minimize memory usage. Systems like Peloton utilize deep learning for autonomous optimization, reducing the administrative burden on database managers [22, 16, 113, 53].

The evolution of self-tuning frameworks signifies a shift toward fully autonomous solutions, utilizing advanced algorithms for proactive performance optimization. By integrating comprehensive planning components, these architectures enhance resource utilization and introduce new optimization techniques, reducing the administrative load on database administrators and improving system efficiency

[22, 96, 100, 8]. As research advances, these methodologies are expected to significantly enhance modern database systems' capabilities and efficiency.

9.2 Resource and Query Optimization

Self-tuning databases employ sophisticated methodologies to optimize resource allocation and query performance, ensuring efficient data management. By combining machine learning models with adaptive algorithms, these databases dynamically adjust configurations in response to evolving workloads and data patterns. Generative models, which capture joint probability distributions, provide accurate query estimates, including those targeting rare subpopulations [114].

In resource allocation, these databases predict future workload patterns, adjusting resources to maintain optimal performance. Anomaly detection techniques identify deviations from normal behavior, triggering necessary adjustments. Analyzing dataset partitions and evaluating joint distributions facilitate anomaly detection, ensuring efficient resource use and minimizing performance degradation [109].

Future research aims to enhance optimization techniques through improved SVD implementations and exploration of statistical properties. Bootstrapped resampling is expected to enhance robustness and accuracy in resource and query optimization [109]. These advancements will contribute to the development of more intelligent and adaptive self-tuning databases capable of efficiently managing resources and optimizing query performance in dynamic environments.

9.3 Anomaly and Atypicality Detection

Anomaly and atypicality detection are vital for optimizing self-tuning database performance by identifying irregular patterns that may indicate system malfunctions, inefficiencies, or security threats. Advanced techniques, such as large language models and contextual Bayesian optimization, analyze complex data patterns to adaptively tune database configurations in real-time, ensuring reliable management in dynamic settings [93, 107, 69, 105, 44]. These mechanisms enable proactive resolution of performance bottlenecks and optimize resource allocation.

Anomaly detection frameworks in self-tuning databases continuously monitor system behavior, identifying deviations that may signal underlying issues. Techniques like Knowledge-Based Temporal Abstraction (KBTA) with temporal pattern mining enhance anomaly detection in time-oriented data [106]. This is particularly effective where temporal patterns are crucial for understanding operations and identifying deviations.

Log data analysis is crucial for maintaining database performance. Frameworks like ADLILog use log instructions from public code projects to train deep learning models, providing an unsupervised method for anomaly detection without costly labeled data [108]. This is vital for detecting anomalies in complex log data, ensuring optimal performance in self-tuning databases.

Using large language models (LLMs) for embedding financial data has improved anomaly detection by offering nuanced insights into atypical patterns [105]. These embeddings capture complex relationships, enhancing databases' ability to detect and respond to anomalies effectively.

Integrating anomaly and atypicality detection mechanisms is essential for maintaining performance and ensuring data-driven applications' reliability and security. By accurately identifying and addressing performance anomalies in SQL queries, systems like PinSQL enhance database management efficiency through structured approaches, including real-time anomaly detection and root cause analysis. These capabilities optimize resource allocation and prevent performance degradation, ensuring prompt resolution of potential issues and maintaining the integrity and responsiveness of cloud database services. Extensive testing has shown that PinSQL achieves an 80

10 Automated Database Administration

The progression of database management increasingly prioritizes automation technologies to boost operational efficiency. This section delves into various automation techniques and tools designed to streamline database administration, emphasizing their role in minimizing manual intervention and optimizing system performance.

10.1 Automation Techniques and Tools

Advanced tools and techniques leveraging artificial intelligence (AI) and machine learning (ML) significantly enhance automated database administration by reducing manual tasks. AI-driven models like PUNQ and GenCrd improve cardinality estimation, thus decreasing the necessity for manual oversight and enhancing system performance [115]. The AirIndex framework exemplifies automated index optimization through a guided graph search, systematically refining index structures to boost data retrieval efficiency without manual input [55]. This adaptability is crucial in dynamic data environments where efficient indexing is pivotal.

Learning-based techniques further illustrate AI's transformative potential in automating administrative tasks. For instance, the openGauss system employs these techniques to adjust dynamically to varying workloads, minimizing manual tuning and maintaining high performance [31]. In IoT data analytics, AutoML techniques automate machine learning processes, promoting scalable and efficient database administration [5]. Automated tuning methods, such as knob tuning, reduce the time and expertise required for configuration adjustments, allowing databases to optimize settings in response to changing conditions [116]. The Baihe framework demonstrates the integration of machine learning models into databases, enhancing automation capabilities while maintaining robustness and stability [47].

Utilizing relational algebra in database process automation reduces execution time and resource usage by eliminating intermediate storage steps, thereby enhancing overall system efficiency [117]. These advancements in tools and techniques for automated database administration signify a major leap in database management, providing intelligent, adaptive solutions that optimize performance and reduce manual intervention. As AI and ML continue to reshape Database Management Systems (DBMS), methodologies fostering self-managing and self-healing systems become increasingly crucial for effectively handling diverse data types and optimizing query performance. The integration of sophisticated algorithms and frameworks for data mining and machine learning within DBMS architectures is set to revolutionize application development and data processing workflows, encouraging broader adoption of these advanced methodologies in the industry [110, 16, 8, 82].

10.2 Unified and Framework-Based Approaches

Unified and framework-based approaches are essential in advancing automated database administration, providing cohesive solutions that enhance efficiency and scalability. These approaches consolidate diverse elements of database management systems (DBMS) into a unified framework, improving data administration and facilitating better interaction among functionalities, particularly regarding automation and semantic parsing challenges [91, 16].

A significant aspect of unified approaches is the integration of AI-driven methodologies into DBMS, which automates processes and reduces manual intervention. The Baihe framework exemplifies this integration, allowing seamless incorporation of machine learning models into existing infrastructures without substantial modifications, enabling databases to adapt dynamically to changing workloads [47]. Framework-based approaches emphasize modular and extensible architectures that support new technologies and methodologies. For instance, the Planter framework facilitates the integration of various models, targets, and architectures, supporting a range of state-of-the-art in-network machine learning algorithms [46]. This flexibility is vital for managing diverse database workloads and ensuring efficient handling of complex data environments.

The application of relational algebra in database process automation also contributes to significant reductions in execution time and resource usage, enhancing overall efficiency by eliminating intermediate storage steps [117]. This underscores the potential of unified frameworks to streamline operations and optimize resource utilization. Unified and framework-based approaches represent a considerable advancement in automated database administration, offering intelligent and adaptive solutions that enhance performance and reduce manual effort. As research progresses in DBMS, particularly in automation driven by AI and ML, methodologies integrating these technologies are expected to be increasingly vital for improving performance, scalability, and operational efficiency of modern DBMS architectures. These advancements address the growing data complexity and volume, facilitating dynamic query optimization and self-managing capabilities, ultimately transforming application structures and enhancing database administration practices [16, 8].

10.3 Challenges and Human Factors

The shift to automated database systems presents several challenges and human factors that need addressing for successful implementation. A primary challenge is the dynamic nature of workloads, complicating virtual machine (VM) consolidation and energy consumption predictions. Traditional systems often struggle to adapt to these fluctuations, necessitating advanced methodologies for optimal resource allocation and energy efficiency [6]. Human factors also significantly influence the adoption of automation in database management. User skepticism and unfamiliarity with automated systems can hinder the transition, as current studies often overlook these critical elements. Addressing these human factors is crucial for fostering a receptive environment for automation, ensuring users are adequately trained and confident in utilizing new technologies [16].

The integration of automated systems into DBMS faces resistance from stakeholders; many users are hesitant to abandon established manual processes due to concerns about losing control, hindering the adoption of advanced automation tools despite their efficiency-enhancing potential [5, 14, 81, 16, 7]. Effective communication and demonstration of automation benefits, such as increased efficiency and reduced error rates, are essential for gaining user buy-in and facilitating a smoother transition.

Addressing both technical and human challenges is vital for the successful implementation of automated database systems. By focusing on automating database management processes and leveraging advanced AI frameworks, organizations can significantly enhance the adaptability and efficiency of their data management systems. This approach not only addresses the increasing volume and complexity of data but also improves data quality through innovative metrics, anomaly detection, and correction methods. Consequently, these advancements pave the way for more intelligent, responsive, and data-driven solutions essential for informed decision-making across various industries [16, 14].

11 Conclusion

The integration of artificial intelligence and machine learning into database management systems signifies a pivotal transformation in data processing, enhancing performance and adaptability. AI-driven techniques optimize various aspects of DBMS operations, including query processing and resource allocation, with innovations like learned indexes exemplifying the potential for revolutionizing data management. Current AI-enhanced systems, such as openGauss, demonstrate the capability of intelligent architectures to navigate complex data environments efficiently. The advancement of hybrid systems that merge AI with traditional methods has notably improved performance, as seen in self-driving DBMS architectures that reduce human intervention while boosting operational efficiency. Reinforcement learning applications further underscore AI's role in resource management optimization, enhancing CPU utilization and resource efficiency. Future research should focus on expanding the application of machine learning models and exploring dynamic datasets to refine optimization processes. Developing models that leverage database values and foreign-key relationships offers a promising direction for innovation. Additionally, building user trust in automation tools and ensuring adequate training for DBMS administration automation are crucial for progress. Simplifying queries and improving the efficiency of Knowledge Graph generation through ontology reshaping are vital for optimizing industrial analytics. The integration of AI into DBMS is poised to enhance efficiency, scalability, and adaptability, shaping the future of data-driven applications and leading to more responsive and intelligent database solutions. As these technologies continue to evolve, they are expected to address modern data environment challenges, ensuring that DBMS can effectively meet contemporary workload demands.

References

[1] Alekh Jindal, Lalitha Viswanathan, and Konstantinos Karanasos. Query and resource optimizations: A case for breaking the wall in big data systems, 2019.

[2] Ryan Marcus. Learned query superoptimization, 2023.

[3] Gaurav Tarlok Kakkar, Jiashen Cao, Aubhro Sengupta, Joy Arulraj, and Hyesoon Kim. Hydro: Adaptive query processing of ml queries, 2024.

[4] Yaoshu Wang, Chuan Xiao, Jianbin Qin, Xin Cao, Yifang Sun, Wei Wang, and Makoto Onizuka. Monotonic cardinality estimation of similarity selection: A deep learning approach, 2021.

[5] Li Yang and Abdallah Shami. Iot data analytics in dynamic environments: From an automated machine learning perspective, 2022.

[6] Tahseen Khan, Wenhong Tian, Guangyao Zhou, Shashikant Ilager, Mingming Gong, and Rajkumar Buyya. Machine learning (ml)-centric resource management in cloud computing: A review and future directions. *Journal of Network and Computer Applications*, 204:103405, 2022.

[7] Jun-Peng Zhu, Peng Cai, Kai Xu, Li Li, Yishen Sun, Shuai Zhou, Haihuang Su, Liu Tang, and Qi Liu. Autotqa: Towards autonomous tabular question answering through multi-agent large language models. *Proceedings of the VLDB Endowment*, 17(12):3920–3933, 2024.

[8] Jim Gray. The revolution in database system architecture, 2004.

[9] Zeyan Li, Nengwen Zhao, Shenglin Zhang, Yongqian Sun, Pengfei Chen, Xidao Wen, Minghua Ma, and Dan Pei. Constructing large-scale real-world benchmark datasets for aiops, 2022.

[10] Zhuoxun Zheng, Baifan Zhou, Dongzhuoran Zhou, Gong Cheng, Ernesto Jiménez-Ruiz, Ahmet Soylu, and Evgeny Kharlamo. Query-based industrial analytics over knowledge graphs with ontology reshaping, 2022.

[11] Youcef Remil. *A data mining perspective on explainable AIOps with applications to software maintenance*. PhD thesis, INSA de Lyon, 2023.

[12] Xuanhe Zhou, Chengliang Chai, Guoliang Li, and Ji Sun. Database meets artificial intelligence: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(3):1096–1116, 2020.

[13] Ziniu Wu, Pei Yu, Peilun Yang, Rong Zhu, Yuxing Han, Yaliang Li, Defu Lian, Kai Zeng, and Jingren Zhou. A unified transferable model for ml-enhanced dbms, 2021.

[14] Widad Elouataoui. Ai-driven frameworks for enhancing data quality in big data ecosystems: Error$_detection, correction, andmetadataintegration$, 2024.

[15] Christopher J. Hazard, Christopher Fusting, Michael Resnick, Michael Auerbach, Michael Meehan, and Valeri Korobov. Natively interpretable machine learning and artificial intelligence: Preliminary results and future directions, 2019.

[16] Yifan Wang, Pierre Bourhis, Romain Rouvoy, and Patrick Royer. Challenges & opportunities in automating dbms: A qualitative study. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pages 2013–2023, 2024.

[17] Aditi Singh, Akash Shetty, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. A survey of large language model-based generative ai for text-to-sql: Benchmarks, applications, use cases, and challenges, 2025.

[18] Immanuel Trummer and Christoph Koch. Approximation schemes for many-objective query optimization, 2014.

[19] Beng Chin Ooi, Shaofeng Cai, Gang Chen, Yanyan Shen, Kian-Lee Tan, Yuncheng Wu, Xiaokui Xiao, Naili Xing, Cong Yue, Lingze Zeng, et al. Neurdb: an ai-powered autonomous data system. *Science China Information Sciences*, 67(10):200901, 2024.

[20] Zhiwei Fan, Rathijit Sen, Paraschos Koutris, and Aws Albarghouthi. A comparative exploration of ml techniques for tuning query degree of parallelism, 2020.

[21] Phanwadee Sinthong and Michael J. Carey. Polyframe: A retargetable query-based approach to scaling dataframes (extended version), 2021.

[22] Andrew Pavlo, Gustavo Angulo, Joy Arulraj, Haibin Lin, Jiexi Lin, Lin Ma, Prashanth Menon, Todd C Mowry, Matthew Perron, Ian Quah, et al. Self-driving database management systems. In *CIDR*, volume 4, page 1, 2017.

[23] Giorgio Vinciguerra, Paolo Ferragina, and Michele Miccinesi. Superseding traditional indexes by orchestrating learning and geometry, 2019.

[24] Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures, 2018.

[25] Xianghong Xu, Zhibing Zhao, Tieying Zhang, Rong Kang, Luming Sun, and Jianjun Chen. Coool: A learning-to-rank approach for sql hint recommendations, 2023.

[26] Ming Sheng, Shuliang Wang, Yong Zhang, Kaige Wang, Jingyi Wang, Yi Luo, and Rui Hao. Mqrld: A multimodal data retrieval platform with query-aware feature representation and learned index based on data lake, 2025.

[27] Guo-Liang Li, Jiang Wang, and Guo Chen. opengauss: An enterprise-grade open-source database system. *Journal of Computer Science and Technology*, 39(5):1007–1028, 2024.

[28] Siyuan Liu, Sourav S Bhowmick, Wanlu Zhang, Shu Wang, Wanyi Huang, and Shafiq Joty. Neuron: Query optimization meets natural language processing for augmenting database education, 2018.

[29] Peter Akioyamen, Zixuan Yi, and Ryan Marcus. The unreasonable effectiveness of llms for query optimization, 2024.

[30] Peizhi Wu, Ryan Marcus, and Zachary G. Ives. Adding domain knowledge to query-driven learned databases, 2023.

[31] Guoliang Li, Xuanhe Zhou, Ji Sun, Xiang Yu, Yue Han, Lianyuan Jin, Wenbo Li, Tianqing Wang, and Shifu Li. opengauss: An autonomous database system. *Proceedings of the VLDB Endowment*, 14(12):3028–3042, 2021.

[32] Oliver Schulte, Hassan Khosravi, Flavia Moser, and Martin Ester. Learning class-level bayes nets for relational data, 2009.

[33] Yao Tian, Tingyun Yan, Xi Zhao, Kai Huang, and Xiaofang Zhou. A learned index for exact similarity search in metric spaces, 2022.

[34] Ryan Marcus, Andreas Kipf, Alexander van Renen, Mihail Stoian, Sanchit Misra, Alfons Kemper, Thomas Neumann, and Tim Kraska. Benchmarking learned indexes, 2020.

[35] Yuxing Han, Haoyu Wang, Lixiang Chen, Yifeng Dong, Xing Chen, Benquan Yu, Chengcheng Yang, and Weining Qian. Bytecard: Enhancing data warehousing with learned cardinality estimation. *arXiv preprint arXiv:2403.16110*, 2024.

[36] Claude Lehmann, Pavel Sulimov, and Kurt Stockinger. Is your learned query optimizer behaving as you expect? a machine learning perspective, 2024.

[37] Domenico Amato, Giosue' Lo Bosco, and Raffaele Giancarlo. On the suitability of neural networks as building blocks for the design of efficient learned indexes, 2022.

[38] Francis C. Chu, Joseph Y. Halpern, and Praveen Seshadri. Least expected cost query optimization: an exercise in utility, 1999.

[39] Rohit Raghunathan, Sushovan De, and Subbarao Kambhampati. Bayes networks for supporting query processing over incomplete autonomous databases, 2012.

[40] Hai Lan, Zhifeng Bao, J. Shane Culpepper, Renata Borovica-Gajic, and Yu Dong. A simple yet high-performing on-disk learned index: Can we have our cake and eat it too?, 2023.

[41] Prasan Roy, S. Seshadri, S. Sudarshan, and Siddhesh Bhobe. Efficient and extensible algorithms for multi query optimization, 1999.

[42] Wolfgang Gatterbauer and Dan Suciu. Dissociation and propagation for approximate lifted inference with standard relational database management systems, 2016.

[43] Siqiao Xue, Chao Qu, Xiaoming Shi, Cong Liao, Shiyi Zhu, Xiaoyu Tan, Lintao Ma, Shiyu Wang, Shijun Wang, Yun Hu, Lei Lei, Yangfei Zheng, Jianguo Li, and James Zhang. A meta reinforcement learning approach for predictive autoscaling in the cloud, 2022.

[44] Xinyi Zhang, Hong Wu, Yang Li, Jian Tan, Feifei Li, and Bin Cui. Towards dynamic and safe configuration tuning for cloud databases. In *Proceedings of the 2022 International Conference on Management of Data*, pages 631–645, 2022.

[45] Jialin Ding. *Instance-Optimized Database Indexes and Storage Layouts*. PhD thesis, Massachusetts Institute of Technology, 2022.

[46] Changgang Zheng, Mingyuan Zang, Xinpeng Hong, Riyad Bensoussane, Shay Vargaftik, Yaniv Ben-Itzhak, and Noa Zilberman. Automating in-network machine learning, 2022.

[47] Andreas Pfadler, Rong Zhu, Wei Chen, Botong Huang, Tianjing Zeng, Bolin Ding, and Jingren Zhou. Baihe: Sysml framework for ai-driven databases, 2021.

[48] Zihao Zhao, Zhihong Shen, Mingjie Tang, Chuan Hu, Huajin Wang, and Yuanchun Zhou. Pandadb: Understanding unstructured data in graph database, 2022.

[49] Bowen Wu, Dimitrios Koutsoukos, and Gustavo Alonso. Efficiently processing joins and grouped aggregations on gpus, 2025.

[50] Jason Mohoney, Anil Pacaci, Shihabur Rahman Chowdhury, Ali Mousavi, Ihab F. Ilyas, Umar Farooq Minhas, Jeffrey Pound, and Theodoros Rekatsinas. High-throughput vector similarity search in knowledge graphs, 2023.

[51] Chuzhe Tang, Zhiyuan Dong, Minjie Wang, Zhaoguo Wang, and Haibo Chen. Learned indexes for dynamic workloads, 2019.

[52] Hai Lan, Zhifeng Bao, J. Shane Culpepper, and Renata Borovica-Gajic. Updatable learned indexes meet disk-resident dbms – from evaluations to design choices, 2023.

[53] Jialin Ding, Umar Farooq Minhas, Jia Yu, Chi Wang, Jaeyoung Do, Yinan Li, Hantian Zhang, Badrish Chandramouli, Johannes Gehrke, Donald Kossmann, David Lomet, and Tim Kraska. Alex: An updatable adaptive learned index, 2020.

[54] Luis Croquevielle, Guang Yang, Liang Liang, Ali Hadian, and Thomas Heinis. Querying in constant expected time with learned indexes, 2024.

[55] Supawit Chockchowwat, Wenjie Liu, and Yongjoo Park. Airindex: Versatile index tuning through data and storage, 2023.

[56] Alireza Heidari, Amirhossein Ahmadi, and Wei Zhang. Uplif: An updatable self-tuning learned index framework, 2024.

[57] Sachith Pai, Michael Mathioudakis, and Yanhao Wang. Wazi: A learned and workload-aware z-index, 2024.

[58] Minsu Kim, Jinwoo Hwang, Guseul Heo, Seiyeon Cho, Divya Mahajan, and Jongse Park. Accelerating string-key learned index structures via memoization-based incremental training, 2024.

[59] Hussam Abu-Libdeh, Deniz Altınbüken, Alex Beutel, Ed H. Chi, Lyric Doshi, Tim Kraska, Xiaozhou, Li, Andy Ly, and Christopher Olston. Learned indexes for a google-scale disk-based database, 2020.

[60] Supawit Chockchowwat, Wenjie Liu, and Yongjoo Park. Automatically finding optimal index structure, 2022.

[61] Abdullah Al-Mamun, Hao Wu, Qiyang He, Jianguo Wang, and Walid G. Aref. A survey of learned indexes for the multi-dimensional space, 2024.

[62] Ali Hadian and Thomas Heinis. Shift-table: A low-latency learned index for range queries using model correction, 2021.

[63] Yaliang Li, Daoyuan Chen, Bolin Ding, Kai Zeng, and Jingren Zhou. A pluggable learned index method via sampling and gap insertion, 2021.

[64] Xiaoying Wang, Changbo Qu, Weiyuan Wu, Jiannan Wang, and Qingqing Zhou. Are we ready for learned cardinality estimation?, 2021.

[65] Rojeh Hayek and Oded Shmueli. Improved cardinality estimation by learning queries containment rates, 2019.

[66] Jintao Zhang, Chao Zhang, Guoliang Li, and Chengliang Chai. Autoce: An accurate and efficient model advisor for learned cardinality estimation, 2024.

[67] Yuri Kim, Yewon Choi, Yujung Gil, Sanghee Lee, Heesik Shin, and Jaehyok Chong. Bite : Accelerating learned query optimization in a mixed-workload environment, 2023.

[68] Debajyoti Mukhopadhyay, Dhaval Chandarana, Rutvi Dave, Sharyu Page, and Shikha Gupta. Query optimization over web services using a mixed approach, 2012.

[69] Anders Høst-Madsen, Elyas Sabeti, and Chad Walton. Data discovery and anomaly detection using atypicality: Theory, 2017.

[70] Maribel Acosta, Chang Qin, and Tim Schwabe. Neuro-symbolic query optimization in knowledge graphs, 2024.

[71] K. F. D. Rietveld and H. A. G. Wijshoff. Redefining the query optimization process, 2022.

[72] Tarun Kathuria and S. Sudarshan. Efficient and provable multi-query optimization, 2017.

[73] Riccardo Mancini, Srinivas Karthik, Bikash Chandra, Vasilis Mageirakos, and Anastasia Ailamaki. Efficient massively parallel join optimization for large queries, 2022.

[74] Immanuel Trummer and Christoph Koch. Parallelizing query optimization on shared-nothing architectures, 2015.

[75] Ibrahim Sabek and Tim Kraska. The case for learned in-memory joins, 2022.

[76] Ziniu Wu, Parimarjan Negi, Mohammad Alizadeh, Tim Kraska, and Samuel Madden. Factorjoin: A new cardinality estimation framework for join queries, 2022.

[77] Yuvaraj Chesetti and Prashant Pandey. Evaluating learned indexes for external-memory joins, 2024.

[78] Shuai Ma and Jinpeng Huai. Approximate computation for big data analytics, 2019.

[79] Immanuel Trummer and Christoph Koch. Probably approximately optimal query optimization, 2015.

[80] Dawei Tao, Enqi Liu, Sidath Randeni Kadupitige, Michael Cahill, Alan Fekete, and Uwe Röhm. First past the post: Evaluating query optimization in mongodb, 2024.

[81] Julien Pierre Edmond Ghali, Kosuke Shima, Koichi Moriyama, Atsuko Mutoh, and Nobuhiro Inuzuka. Enhancing retrieval processes for language generation with augmented queries, 2024.

[82] Jens Dörpinghaus and Andreas Stefan. Optimization of retrieval algorithms on large scale knowledge graphs, 2020.

[83] Sergey Zinchenko and Sergey Iazov. Hero: Hint-based efficient and reliable query optimizer, 2024.

[84] Chi Zhang, Olga Papaemmanouil, Josiah P. Hanna, and Aditya Akella. Multi-agent databases via independent learning, 2022.

[85] Yaoshu Wang, Chuan Xiao, Jianbin Qin, Rui Mao, Onizuka Makoto, Wei Wang, Rui Zhang, and Yoshiharu Ishikawa. Consistent and flexible selectivity estimation for high-dimensional data, 2021.

[86] Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. Neo: A learned query optimizer, 2019.

[87] Jonas Heitz and Kurt Stockinger. Join query optimization with deep reinforcement learning algorithms, 2019.

[88] Yingze Li, Hongzhi Wang, and Xianglong Liu. One stone, two birds: A lightweight multidimensional learned index with cardinality support, 2023.

[89] Ryan Marcus, Parimarjan Negi, Hongzi Mao, Nesime Tatbul, Mohammad Alizadeh, and Tim Kraska. Bao: Learning to steer query optimizers, 2020.

[90] Jennifer Ortiz, Magdalena Balazinska, Johannes Gehrke, and S. Sathiya Keerthi. Learning state representations for query optimization with deep reinforcement learning, 2018.

[91] Ben Eyal, Amir Bachar, Ophir Haroche, Moran Mahabi, and Michael Elhadad. Semantic decomposition of question and sql for text-to-sql parsing, 2023.

[92] Tianyu Cui, Shiyu Ma, Ziang Chen, Tong Xiao, Shimin Tao, Yilun Liu, Shenglin Zhang, Duoming Lin, Changchang Liu, Yuzhe Cai, Weibin Meng, Yongqian Sun, and Dan Pei. Logeval: A comprehensive benchmark suite for large language models in log analysis, 2024.

[93] Hongwei Jin, George Papadimitriou, Krishnan Raghavan, Pawel Zuk, Prasanna Balaprakash, Cong Wang, Anirban Mandal, and Ewa Deelman. Large language models for anomaly detection in computational workflows: from supervised fine-tuning to in-context learning, 2024.

[94] David Charte, Francisco Charte, María J. del Jesus, and Francisco Herrera. A showcase of the use of autoencoders in feature learning applications, 2020.

[95] Daniel Lindner, Daniel Ritter, and Felix Naumann. Enabling data dependency-based query optimization, 2024.

[96] Lin Ma. *Self-Driving Database Management Systems: Forecasting, Modeling, and Planning*. PhD thesis, Carnegie Mellon University, 2021.

[97] Supawit Chockchowwat, Chaitanya Sood, and Yongjoo Park. Airphant: Cloud-oriented document indexing, 2021.

[98] Deepika Saxena and Ashutosh Kumar Singh. workload forecasting and resource management models based on machine learning for cloud computing environments, 2021.

[99] Florian Schmidt, Mathias Niepert, and Felipe Huici. Representation learning for resource usage prediction, 2018.

[100] Lin Ma, Dana Van Aken, Ahmed Hefny, Gustavo Mezerhane, Andrew Pavlo, and Geoffrey J Gordon. Query-based workload forecasting for self-driving database management systems. In *Proceedings of the 2018 International Conference on Management of Data*, pages 631–645, 2018.

[101] Deepika Saxena, Jitendra Kumar, Ashutosh Kumar Singh, and Stefan Schmid. Performance analysis of machine learning centered workload prediction models for cloud, 2023.

[102] Jiaqi Wang, Tianyi Li, Anni Wang, Xiaoze Liu, Lu Chen, Jie Chen, Jianye Liu, Junyang Wu, Feifei Li, and Yunjun Gao. Real-time workload pattern analysis for large-scale cloud databases. *arXiv preprint arXiv:2307.02626*, 2023.

[103] Andreas Kipf, Ryan Marcus, Alexander van Renen, Mihail Stoian, Alfons Kemper, Tim Kraska, and Thomas Neumann. Sosd: A benchmark for learned indexes, 2019.

[104] Domenico Amato, Giosuè Lo Bosco, and Raffaele Giancarlo. Standard vs uniform binary search and their variants in learned static indexing: The case of the searching on sorted data benchmarking software platform, 2022.

AI-generated, for reference only.

[105] Alexander Bakumenko, Kateřina Hlaváčková-Schindler, Claudia Plant, and Nina C. Hubig. Advancing anomaly detection: Non-semantic financial data encoding with llms, 2024.

[106] Asaf Shabtai. Anomaly detection using the knowledge-based temporal abstraction method, 2016.

[107] Thorsten Wittkopp, Philipp Wiesner, Dominik Scheinert, and Odej Kao. A taxonomy of anomalies in log data, 2021.

[108] Jasmin Bogatinovski, Gjorgji Madjarov, Sasho Nedelkoski, Jorge Cardoso, and Odej Kao. Leveraging log instructions in log-based anomaly detection, 2022.

[109] Neil Caithness and David Wallom. Anomaly detection for industrial big data, 2018.

[110] Thibault Sellam and Martin Kersten. 80 new packages to mine database query logs, 2017.

[111] Xiaoze Liu, Zheng Yin, Chao Zhao, Congcong Ge, Lu Chen, Yunjun Gao, Dimeng Li, Ziting Wang, Gaozhong Liang, Jian Tan, et al. Pinsql: Pinpoint root cause sqls to resolve performance issues in cloud databases. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 2549–2561. IEEE, 2022.

[112] Georg Gottlob, Giorgio Orsi, and Andreas Pieris. Ontological queries: Rewriting and optimization (extended version), 2011.

[113] Federica Panella. Approximate query answering in inconsistent databases, 2014.

[114] Moritz Kulessa, Alejandro Molina, Carsten Binnig, Benjamin Hilprecht, and Kristian Kersting. Model-based approximate query processing, 2018.

[115] Rojeh Hayek and Oded Shmueli. Nn-based transformation of any sql cardinality estimator for handling distinct, and, or and not, 2020.

[116] Xinyang Zhao, Xuanhe Zhou, and Guoliang Li. Automatic database knob tuning: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12470–12490, 2023.

[117] Remco Dijkman, Juntao Gao, Paul Grefen, and Arthur ter Hofstede. Relational algebra for in-database process mining, 2017.