International Journal of Artificial Intelligence, Data Science, and Machine Learning
Grace Horizon Publication | Volume 7, Issue 1, 91-99, 2026
ISSN: 3050-9262 | https://doi.org/10.63282/3050-9262.IJAIDSML-V7I1P118

Original Article

Federated AI-Driven Query Optimization for Distributed Cloud Databases

Parameswara Reddy Nangi[1], Chaithanya Kumar Reddy Nala Obannagari[2]
[1,2]Independent Researcher, USA.

Received On: 25/12/2025 Revised On: 27/01/2026 Accepted On: 31/01/2026 Published On: 03/02/2026

Abstract *- The rapid adoption of distributed cloud databases across multi-cloud and hybrid environments has exposed significant inefficiencies in traditional query optimization techniques, primarily due to data heterogeneity, dynamic workloads, and strict data privacy constraints that limit centralized analysis. Conventional cost-based and machine-learning–driven optimizers struggle to scale effectively in such environments, as they rely on static statistics or require access to globally aggregated query execution data. To address these challenges, this paper proposes a federated AI-driven query optimization framework that enables intelligent and privacy-preserving optimization across distributed cloud databases without sharing raw data. The proposed approach employs federated learning to collaboratively train local cost estimation models using query workload characteristics and execution feedback at each database node, while a global model is iteratively refined through secure aggregation of model updates. An AI-based cost modeling mechanism is integrated with adaptive query plan selection to dynamically optimize execution strategies under varying workload and resource conditions. Extensive experimental evaluations conducted on distributed cloud testbeds using benchmark workloads demonstrate that the proposed framework achieves significant reductions in query latency, improved resource utilization, and enhanced scalability compared to centralized and traditional optimization approaches. The results confirm that federated AI-driven query optimization offers a practical and effective solution for next-generation distributed cloud database systems, balancing performance optimization with data privacy and system autonomy.*

Keywords *- Federated Learning, Query Optimization, Distributed Databases, Cloud Computing, AI-Driven Systems, Data Privacy.*

1. Introduction

The rapid evolution of cloud computing has led to the widespread adoption of distributed cloud databases deployed across multi-cloud, hybrid-cloud, and geographically dispersed environments. [1,2] Modern data-intensive applications such as real-time analytics, large-scale transaction processing, and AI-driven services increasingly rely on these distributed database systems to ensure scalability, availability, and fault tolerance. However, the decentralized nature of data storage and query execution introduces significant complexity in achieving efficient query optimization, particularly under dynamic workloads and heterogeneous infrastructure conditions. Traditional query optimization techniques are largely designed for centralized or tightly coupled database systems, where global statistics and execution plans can be readily accessed. In distributed cloud environments, centralized query optimization becomes inefficient due to high communication overhead, network latency, and limited visibility into remote execution contexts. Moreover, stringent data privacy regulations, organizational policies, and cross-cloud boundaries restrict the sharing of raw query logs and execution statistics, making centralized learning-based optimization impractical.

Existing cost-based optimizers rely on static statistics and simplified cost models that fail to adapt to workload variability and resource heterogeneity, while recent machine learning–based optimizers often assume centralized access to training data. These assumptions limit their applicability in distributed cloud databases, where scalability, autonomy of database nodes, and privacy preservation are critical requirements. Consequently, there is a clear need for a query optimization approach that can learn from distributed execution environments without violating data locality constraints or incurring excessive coordination costs. The primary objective of this research is to design a federated AI-based query optimization framework that enables collaborative learning across distributed database nodes while preserving data privacy. By leveraging federated learning, the proposed approach aims to train accurate cost estimation and optimization models using local query workloads and execution feedback, without transferring sensitive data. The framework seeks to improve query performance, scalability, and adaptability in distributed cloud database systems.

2. Related Work

2.1. Query Optimization in Distributed Databases

Query optimization in distributed database systems has traditionally relied on rule-based and cost-based approaches. Rule-based optimizers apply predefined heuristics, [3-5] such as join reordering and predicate pushdown, to reduce query execution cost, while cost-based optimizers evaluate alternative execution plans using statistical estimates and analytical cost models. In distributed environments, these methods are extended to account for data placement, network latency, and inter-node communication overhead. Although effective in relatively stable systems, their performance strongly depends on the accuracy of global statistics and static assumptions about system behavior. As distributed cloud databases become more dynamic and heterogeneous, maintaining up-to-date statistics and accurate cost models becomes increasingly difficult. Frequent workload variations, elastic resource provisioning, and cross-cloud execution introduce uncertainty that traditional optimizers are not designed to handle efficiently. Consequently, these approaches often result in suboptimal query plans and limited scalability in modern cloud-based deployments.

2.2. Machine Learning for Query Optimization

To overcome the limitations of traditional optimizers, recent studies have explored machine learning–based query optimization techniques. Supervised learning models, including regression and deep neural networks, have been proposed for tasks such as cardinality estimation and cost prediction, while reinforcement learning approaches aim to learn optimal query plans by interacting with the query execution environment. These methods have demonstrated improved accuracy and adaptability compared to handcrafted cost models, particularly in complex query workloads. Despite their promise, most ML-based query optimization approaches rely on centralized training using aggregated query logs and execution metrics. This centralized assumption introduces scalability bottlenecks and raises concerns regarding data privacy and compliance, especially in multi-tenant and cross-organizational cloud environments. As a result, their applicability to distributed cloud databases remains limited.

2.3. Federated Learning in Database Systems

Federated learning has gained attention as a privacy-preserving distributed learning paradigm, allowing multiple participants to collaboratively train models without sharing raw data. In the context of database systems, federated learning has been applied to distributed analytics, workload prediction, and performance monitoring, enabling decentralized data processing while capturing global patterns. These approaches typically involve local model training at each node followed by secure aggregation of model updates.However, the application of federated learning to query optimization is still in its early stages. Existing federated database solutions primarily focus on analytical tasks rather than core query planning and execution. Furthermore, challenges such as system heterogeneity, communication overhead, and integration with database optimizers are not fully addressed in current federated approaches.

2.4. Research Gaps

Although prior work has made progress in distributed query optimization, machine learning–based optimization, and federated database systems, there is a notable lack of federated AI-driven frameworks specifically designed for query optimization in distributed cloud databases. Current solutions either assume centralized data access or do not effectively integrate federated learning with query planning mechanisms. This gap highlights the need for a unified approach that combines AI-driven optimization with federated learning to achieve scalability, adaptability, and privacy preservation.This paper aims to address these limitations by proposing a federated AI-based query optimization framework that enables collaborative learning across distributed database nodes without sharing sensitive data, thereby advancing the state of the art in distributed cloud database optimization.

3. System Model and Problem Formulation

3.1. Distributed Cloud Database Architecture

The system considered in this study consists of a distributed cloud database architecture deployed across multi-cloud and hybrid-cloud environments, [6-8] where multiple database nodes operate under different administrative domains. Each node maintains its own data partitions and executes queries locally or cooperatively with other nodes depending on data placement and query requirements. The architecture supports both intra-cloud and cross-cloud query execution, enabling scalability and fault tolerance while introducing challenges related to coordination and performance optimization. Data is horizontally or vertically partitioned across database nodes, and queries may require accessing data from multiple locations. A distributed query execution engine decomposes incoming queries into sub-queries, which are executed in parallel across relevant nodes. Intermediate results are exchanged over the network and combined to produce final query results. Due to differences in hardware resources, network conditions, and database configurations, query execution performance can vary significantly across nodes.

3.2. Query Optimization Problem Definition

The objective of query optimization in this distributed setting is to select an execution plan that minimizes overall query cost while satisfying system and application requirements. The cost of a query plan is modeled using multiple metrics, including query latency, system throughput, and resource utilization such as CPU, memory, and network bandwidth. These metrics often exhibit trade-offs, requiring the optimizer to balance performance and resource efficiency. Formally, given a query workload and a set of possible execution plans, the optimization problem involves estimating the cost of each plan under dynamic runtime conditions and selecting the plan that yields the best expected performance. In distributed cloud databases, this problem is further complicated by uncertainties in network latency, workload interference, and elastic resource provisioning, making accurate cost estimation a challenging task.

3.3. Assumptions and Constraints

The proposed system operates under several practical assumptions and constraints inherent to distributed cloud environments. First, data privacy constraints prevent the sharing of raw query data, execution logs, or sensitive statistics across database nodes, necessitating decentralized learning and optimization. Second, network latency and bandwidth variability introduce communication overhead that limits frequent synchronization and centralized coordination among nodes. Additionally, the system assumes heterogeneity across database nodes in terms of hardware capabilities, storage systems, and database engines. This heterogeneity affects query execution behavior and complicates the construction of a unified optimization model. The proposed framework is designed to operate within these constraints by enabling local learning and federated model aggregation, ensuring scalability and adaptability without violating data locality or privacy requirements.

4. Federated AI-Driven Query Optimization Framework

4.1. Extracted Features Used for Local AI-Based Query Performance Modeling

Table1: Feature Set for Local Query Performance Modeling

Feature Category	Feature Name	Description
Query Structure	Join Count	Number of joins in query
Query Structure	Predicate Complexity	Filter conditions
Data Statistics	Table Cardinality	Size of referenced tables
Runtime Metrics	Execution Time	Operator-level latency
Resource Metrics	CPU Utilization	Processor usage
Network Metrics	Data Transfer Size	Inter-node communication

The table summarizes the key features extracted for local query performance modeling in the proposed federated AI-driven query optimization framework. [9,10] Each feature captures specific aspects of query behavior, data characteristics, or system resource usage that influence execution performance. Under the Query Structure category, Join Count measures the number of joins in a query, while Predicate Complexity quantifies the filtering conditions, both of which impact query execution cost and plan selection. Data Statistics, represented by Table Cardinality, provides information about the size of the tables involved, which is critical for estimating intermediate result sizes and operator costs. Runtime Metrics, such as Execution Time, record operator-level latency observed during query execution, offering direct feedback on plan efficiency. Resource Metrics, including CPU Utilization, capture the processing load on each node, helping the AI models account for contention and hardware limitations. Finally, Network Metrics, exemplified by Data Transfer Size, quantify the volume of inter-node communication, which is particularly important in distributed cloud environments where network latency can significantly affect query performance. Collectively, these features provide a comprehensive and privacy-preserving representation of local query workloads, enabling the federated AI models to learn accurate cost estimations without sharing raw data across nodes.

4.2. Overview of the Proposed Framework

The proposed federated AI-driven query optimization framework is designed to enable intelligent and scalable optimization across distributed cloud database systems while preserving data privacy. The framework follows a decentralized architecture in which each database node independently observes local query workloads and execution behavior. Instead of transferring raw data to a centralized optimizer, each node trains a local AI model that captures query performance characteristics specific to its environment. A federated coordination layer aggregates model updates to construct a global optimization model that reflects system-wide execution patterns. The overall workflow consists of iterative rounds of local model training, secure model update exchange, and global model aggregation. The aggregated model is then redistributed to participating nodes to guide query plan selection and cost estimation. This collaborative learning process allows the system to continuously adapt to workload changes and infrastructure dynamics while minimizing communication overhead and maintaining node autonomy.

4.3. Local Query Performance Modeling

At each database node, local query performance modeling begins with feature extraction from query workloads and execution traces. Relevant features include query structure characteristics, such as join types and predicate complexity, as well as runtime metrics, including execution time, resource consumption, and data access patterns. These features provide a compact representation of local query behavior without exposing sensitive data. Using the extracted features, each node trains a local AI model to estimate query execution costs and predict performance under different execution plans. The model is periodically updated using newly observed workloads, enabling continuous learning and adaptation to changing conditions. By keeping training localized, the framework ensures that sensitive execution data remains within the originating database environment.

4.4. Federated Learning Mechanism

The federated learning mechanism coordinates the collaborative training process across distributed database nodes. In each training round, nodes compute model updates based on locally trained parameters and transmit only these updates to a federated aggregator. The aggregator employs a model aggregation strategy, such as weighted averaging, to combine local updates into a global model that captures cross-node performance trends while accounting for data and workload heterogeneity.To reduce communication overhead and support scalability, the framework adopts an efficient communication protocol that limits the frequency and size of model exchanges. Model synchronization can be performed asynchronously to accommodate network variability and node availability. The updated global model is then disseminated back to participating nodes, where it is used to improve local query optimization decisions.

4.5. Federated Learning Model Architecture

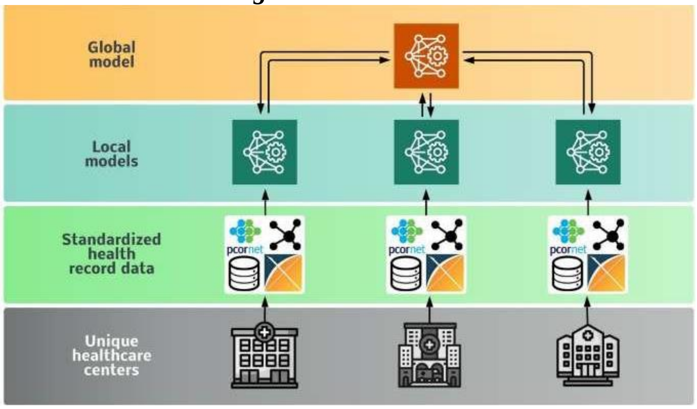

Fig 1: Federated Learning Model Architecture

The figure illustrates the federated learning architecture used for collaborative AI-driven optimization in distributed environments. At the bottom layer, [11,12] multiple unique healthcare centers represent decentralized nodes, each possessing sensitive local data. The next layer shows standardized health record data, indicating that although the underlying data is heterogeneous, it is formatted into a uniform schema to facilitate learning without sharing raw records. Each node trains a local AI model using its own data, depicted in the middle layer, capturing local query execution patterns and workload characteristics. These local models periodically transmit model updates to the global model at the top layer, where updates are aggregated using secure techniques to create a shared global representation. The global model is then redistributed to all nodes, enabling continuous improvement in query optimization performance while preserving data privacy and complying with regulatory constraints. This layered architecture effectively demonstrates how federated learning enables collaborative intelligence across distributed nodes without exposing sensitive raw data.

4.6. Privacy and Security Considerations

Privacy preservation is a core design principle of the proposed framework. Data locality is strictly maintained by ensuring that raw query data, execution logs, and sensitive statistics never leave their respective database nodes. Only model parameters or gradients are shared during the federated learning process, significantly reducing the risk of data leakage. To further enhance security, model update protection mechanisms are incorporated to prevent inference attacks and unauthorized access. These include secure aggregation techniques and controlled access to the federated coordination layer. Together, these measures enable collaborative query optimization while complying with data protection regulations and organizational privacy policies in distributed cloud environments.

5. AI Models and Optimization Techniques

5.1. Learning Models for Cost Estimation

Accurate cost estimation is a critical component of effective query optimization in distributed cloud databases. The proposed framework employs machine learning–based cost models, [13,14] including regression techniques and neural networks, to predict query execution costs under varying execution plans and system conditions. These models learn complex, non-linear relationships between query features, data characteristics, and resource utilization that are difficult to capture using traditional analytical cost models. Lightweight regression models are used for scenarios requiring low training overhead, while deeper neural network architectures are employed to model more complex query workloads and execution patterns. The models are trained using historical query execution data and continuously refined through federated learning, enabling them to adapt to workload changes and infrastructure dynamics without relying on static statistics or centralized data collection.

5.2. Reinforcement Learning for Plan Selection

To complement cost estimation, the framework integrates reinforcement learning (RL) techniques for query plan selection. In this formulation, the query optimization process is modeled as a sequential decision-making problem. The state represents the current query and system context, including query structure, data distribution, and resource availability. The action corresponds to selecting or modifying an execution plan, such as join ordering or operator placement across nodes. The reward function is designed to capture execution performance objectives, such as minimizing query latency or resource consumption. Through repeated interactions with the execution environment, the RL agent learns policies that balance short-term execution efficiency with long-term system performance. This adaptive learning process allows the optimizer to explore alternative plans and converge toward optimal strategies under dynamic conditions.

5.3. Adaptive Query Plan Generation

The proposed framework supports adaptive query plan generation by incorporating runtime feedback into the optimization process. During query execution, performance metrics such as operator latency and resource utilization are monitored and compared against predicted costs. Significant deviations trigger re-evaluation of execution plans, allowing the system to adjust optimization decisions in response to unforeseen runtime conditions. This feedback-driven re-optimization mechanism enables the framework to handle workload variability, data skew, and resource contention more effectively than static optimization approaches. By combining predictive cost modeling, reinforcement learning–based decision making, and continuous runtime adaptation, the proposed system achieves robust and efficient query optimization in distributed cloud database environments.

6. Experimental Setup

6.1. End-to-End AI Workflow for Monitoring, Training, and Decision Support

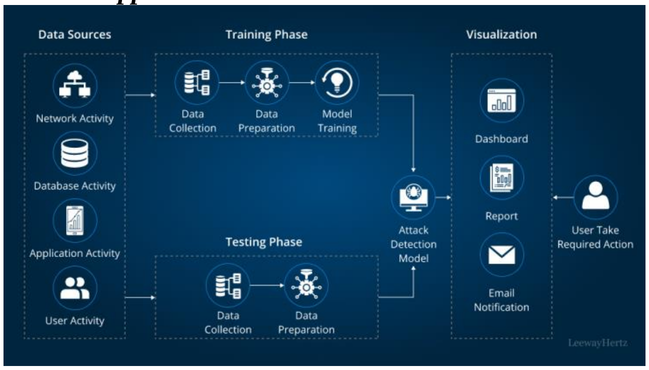

Fig 2: End-to-End AI Workflow for Monitoring, Training, and Decision Support

The figure illustrates an end-to-end AI workflow that begins with multiple data sources, including network activity, database activity, application activity, and user activity. [15,16] These heterogeneous data streams are collected and processed during the training phase, where data collection, data preparation, and model training are performed to build an effective analytical model. The trained model is subsequently evaluated during the testing phase, which follows a similar pipeline of data collection and preparation to validate model performance under unseen conditions. The output of both phases feeds into a central detection or decision model, which represents the system's intelligence layer. The results generated by this model are presented through a visualization layer, including dashboards, reports, and email notifications, enabling users to take informed and timely actions. Overall, the figure highlights how raw operational data is transformed into actionable insights through structured AI pipelines, emphasizing automation, continuous learning, and user-centric decision support.

6.2. Testbed Configuration

The experimental evaluation was conducted on a distributed cloud testbed deployed across multi-cloud and hybrid-cloud environments to reflect real-world deployment scenarios. The testbed consists of multiple database nodes hosted on heterogeneous cloud platforms, each configured with varying compute, memory, and storage resources. This setup enables the assessment of query optimization performance under diverse infrastructure conditions, including differences in network latency and resource availability. Industry-standard relational database engines were used at each node, configured to support distributed query execution and parallel processing. The federated coordination component was deployed as a lightweight service responsible for model aggregation and synchronization. All experiments were performed under controlled network conditions while allowing dynamic workload variations to evaluate the adaptability and scalability of the proposed framework.

6.3. Datasets and Workloads

To ensure reproducibility and comparability, the evaluation employed widely used benchmark datasets, including TPC-H and TPC-DS, which represent decision-support and analytical workloads. [17,18] These benchmarks provide a diverse set of complex SQL queries involving joins, aggregations, and nested subqueries, making them suitable for evaluating distributed query optimization techniques. The datasets were partitioned across database nodes using horizontal and hybrid partitioning strategies to simulate realistic data distribution scenarios. Query workloads were generated at varying scales and arrival rates to assess system behavior under both steady-state and dynamic workload conditions.

6.4. Baseline Methods

The performance of the proposed federated AI-driven query optimization framework was compared against multiple baseline approaches. Traditional optimizers, based on rule-based and cost-based techniques provided by the underlying database engines, were used as the primary baseline. These optimizers rely on static statistics and analytical cost models without adaptive learning capabilities. In addition, centralized machine learning–based optimizers were implemented as advanced baselines, where query execution data from all nodes is aggregated to train a global cost model. This comparison highlights the trade-offs between centralized learning and the proposed federated approach in terms of performance, scalability, and privacy preservation.

6.5. Evaluation Metrics

The evaluation focused on multiple performance metrics to comprehensively assess the effectiveness of the proposed framework. Query latency was measured as the end-to-end execution time of queries, while system throughput was evaluated in terms of the number of queries successfully processed per unit time. These metrics capture the impact of optimization decisions on user-perceived performance. Additionally, optimization overhead was measured to quantify the computational and communication costs introduced by the federated learning process. This includes model training time, aggregation latency, and communication overhead. Together, these metrics provide a balanced assessment of performance gains relative to the cost of optimization.

7. Results and Discussion

7.1. Performance Comparison Results

Table 2: Performance Comparison of Query Optimization Methods

Method	Avg. Latency (ms)	Throughput (q/s)	Improvement (%)
Rule-Based Optimizer	1450	18	–
Cost-Based Optimizer	1280	22	11.7
Centralized ML Optimizer	980	29	32.4
Federated AI Optimizer	820	34	43.4

The results presented in this table compare the performance of different query optimization methods in terms of average query latency, system throughput, and overall performance improvement. The rule-based optimizer serves as the baseline and exhibits the highest average latency (1450 ms) and the lowest throughput (18 queries per second), reflecting the limitations of static heuristic-based optimization in dynamic distributed cloud environments. The cost-based optimizer shows moderate improvement, reducing latency to 1280 ms and increasing throughput to 22 q/s, which corresponds to an 11.7% performance gain; however, its reliance on static statistics restricts further optimization. The centralized machine learning–based optimizer achieves significantly better performance, with average latency reduced to 980 ms and throughput increased to 29 q/s, demonstrating the effectiveness of data-driven cost estimation. Nevertheless, this approach requires centralized access to execution data, which introduces scalability and privacy concerns. In contrast, the proposed federated AI optimizer delivers the best overall performance, achieving the lowest latency (820 ms) and highest throughput (34 q/s), resulting in a 43.4% improvement over the baseline. These results highlight that federated learning can achieve performance comparable to or better than centralized ML approaches while preserving data locality and scalability in distributed cloud database environments.

7.2. Performance Comparison

The performance of the proposed federated AI-driven query optimization framework was evaluated against traditional and centralized machine learning–based optimization approaches. Experimental results show that the federated approach consistently achieves lower query latency and higher throughput compared to traditional cost-based optimizers. This improvement is primarily attributed to the adaptive learning of execution costs that more accurately reflects dynamic workload and system conditions. When compared with centralized ML-based optimizers, the federated framework demonstrates comparable or improved query performance while avoiding the overhead and privacy risks associated with centralized data aggregation. The results indicate that federated learning effectively captures global optimization patterns through collaborative model training, enabling efficient query plan selection without direct access to distributed execution data.

7.3. Scalability Analysis

Scalability was evaluated by increasing both the number of participating database nodes and the size of the query workload. The federated framework maintains stable performance as the system scales, with query latency increasing sub-linearly relative to the number of nodes. This behavior highlights the framework’s ability to adapt to larger distributed environments without incurring significant coordination overhead. As workload intensity increases, the proposed approach continues to outperform baseline methods by dynamically adjusting query plans based on updated cost estimates. These results demonstrate that federated AI-driven optimization is well-suited for large-scale cloud deployments where workload and resource conditions vary over time.

7.4. Communication and Training Overhead

The communication and training overhead introduced by federated learning was carefully analyzed to assess its practical feasibility. Results indicate that the overhead associated with model aggregation and synchronization remains modest compared to overall query execution time. By limiting the frequency and size of model updates, the framework effectively balances learning accuracy and communication efficiency. Training overhead at individual nodes is also manageable, as local model updates are lightweight and can be performed asynchronously. This design ensures that optimization processes do not interfere with normal query execution, making the approach suitable for production environments.

7.5. Privacy and Robustness Evaluation

The proposed framework was evaluated for privacy preservation and robustness under data isolation constraints. Since raw query data and execution logs remain localized at each database node, the federated approach inherently prevents sensitive information from being exposed during the optimization process. Experimental results confirm that this data isolation does not negatively impact optimization effectiveness. Furthermore, the framework demonstrates robustness to heterogeneous data distributions and partial node participation. Even when some nodes contribute limited updates, the global model continues to converge and support effective query optimization. These findings validate the practicality of federated AI-driven optimization in privacy-sensitive and heterogeneous cloud database environments.

8. Challenges and Limitations

8.1. Model Convergence Issues

One of the primary challenges in federated AI-driven query optimization is ensuring reliable model convergence across distributed database nodes. Variations in local data distributions, query workloads, and execution environments can lead to non-independent and non-identically distributed (non-IID) training data, which may slow convergence or result in suboptimal global models. Additionally, asynchronous updates and partial node participation can further complicate the learning process. Although the proposed framework mitigates these effects through iterative aggregation and adaptive learning strategies, convergence guarantees remain limited under highly skewed workloads. Further investigation is required to develop convergence-aware aggregation techniques that can better handle extreme heterogeneity and dynamic participation.

8.2. Network Overhead

Federated learning introduces additional network overhead due to periodic exchange of model updates between database nodes and the aggregation service. In geographically distributed cloud environments, variable network latency and bandwidth constraints can affect the timeliness of model synchronization. Excessive communication may also compete with query execution traffic, potentially impacting overall system performance. While the framework minimizes overhead by transmitting compact model updates at controlled intervals, network costs cannot be entirely eliminated. In environments with strict latency requirements or limited connectivity, communication-efficient federated learning strategies are necessary to further reduce overhead.

8.3. Heterogeneous Database Engines

Another significant limitation arises from the presence of heterogeneous database engines across distributed cloud deployments. Differences in query planners, execution operators, and internal cost metrics make it challenging to develop a unified optimization model that generalizes effectively across systems. Execution behavior may vary even for identical queries, reducing the accuracy of shared models. Although the proposed framework accommodates heterogeneity through local model training, cross-engine generalization remains an open challenge. Future extensions should explore engine-aware feature representations and modular optimization strategies to better support diverse database technologies.

9. Future Research Directions

9.1. Cross-Engine Optimization

Future research should explore cross-engine query optimization to enable federated AI models to operate effectively across heterogeneous database systems. Different database engines employ distinct query planners, execution operators, and optimization strategies, which complicates the development of generalized cost models. Advancing engine-agnostic feature representations and abstraction layers would allow federated models to better capture common execution patterns while preserving engine-specific optimizations. Additionally, incorporating transfer learning techniques could help leverage knowledge learned from one database engine to improve optimization performance on others. Such approaches would enhance the applicability of federated query optimization frameworks in multi-engine cloud environments.

9.2. Online Federated Learning

Another promising direction is the adoption of online federated learning, where models are continuously updated in near real time as new query workloads and execution feedback become available. Unlike batch-based federated learning, online learning enables faster adaptation to workload shifts, data skew, and resource fluctuations commonly observed in cloud databases. However, online federated learning introduces challenges related to model stability, communication frequency, and convergence guarantees. Future work should investigate adaptive synchronization strategies and incremental learning algorithms that balance responsiveness with system overhead.

9.3. Integration with Serverless Databases

The growing adoption of serverless and cloud-native database architectures presents new opportunities and challenges for federated AI-driven query optimization. Serverless databases offer elastic scaling and fine-grained resource management but introduce highly dynamic execution environments with ephemeral compute resources. Integrating federated optimization frameworks with serverless databases requires rethinking model training, state management, and cost estimation under transient execution contexts. Future research should focus on lightweight, stateless learning mechanisms and adaptive optimization techniques that align with the principles of serverless computing.

10. Conclusion

This paper investigated the problem of inefficient query optimization in distributed cloud databases and proposed a federated AI-driven framework to address the limitations of traditional and centralized optimization approaches. By combining machine learning–based cost estimation, reinforcement learning–based plan selection, and federated learning, the proposed system enables collaborative optimization across distributed database nodes without sharing raw query execution data. Experimental results demonstrated that the framework effectively reduces query latency, improves throughput, and maintains stable performance under dynamic workloads and scalable cloud deployments. The key contributions of this work include the design of a novel federated query optimization architecture tailored for multi-cloud and hybrid-cloud environments, the integration of AI-driven cost modeling with privacy-preserving federated learning, and a comprehensive experimental evaluation using benchmark workloads. Unlike existing solutions, the proposed approach achieves competitive or superior performance compared to centralized machine learning–based optimizers while preserving data locality and system autonomy. These contributions advance the state of the art in intelligent query optimization for distributed database systems. From a practical perspective, the proposed framework offers a viable solution for organizations operating privacy-sensitive and geographically distributed cloud databases. By eliminating the need for centralized data aggregation, the approach aligns with regulatory requirements and cross-organizational constraints while delivering performance benefits. The framework can be integrated into modern cloud database platforms to support adaptive, scalable, and secure query optimization, paving the way for next-generation intelligent database management systems.

Reference

[1] Forresi, C., Francia, M., Gallinucci, E., & Golfarelli, M. (2023). Cost-based optimization of multistore query plans. Information systems frontiers, 25(5), 1925-1951.

[2] Chaudhuri, S. (1998, May). An overview of query optimization in relational systems. In Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems (pp. 34-43).

[3] Dritsas, E., & Trigka, M. (2025). A Survey on Database Systems in the Big Data Era: Architectures, Performance, and Open Challenges. IEEE Access.

[4] Graefe, G. (1993). Query evaluation techniques for large databases. ACM Computing Surveys (CSUR), 25(2), 73-169.

[5] Neumann, T. (2011). Efficiently compiling efficient query plans for modern hardware. Proceedings of the VLDB Endowment, 4(9), 539-550.

[6] Ortiz, J., Balazinska, M., Gehrke, J., & Keerthi, S. S. (2018, June). Learning state representations for query optimization with deep reinforcement learning. In Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning (pp. 1-4).

[7] Mikhaylov, A., Mazyavkina, N. S., Salnikov, M., Trofimov, I., Qiang, F., & Burnaev, E. (2022). Learned query optimizers: Evaluation and improvement. IEEE Access, 10, 75205-75218.

[8] Kraska, T., Beutel, A., Chi, E. H., Dean, J., & Polyzotis, N. (2018, May). The case for learned index structures. In Proceedings of the 2018 international conference on management of data (pp. 489-504).

[9] Pavlo, A., Angulo, G., Arulraj, J., Lin, H., Lin, J., Ma, L., ... & Zhang, T. (2017, January). Self-Driving Database Management Systems. In CIDR (Vol. 4, p. 1).

[10] Mao, H., Schwarzkopf, M., Venkatakrishnan, S. B., Meng, Z., & Alizadeh, M. (2019). Learning scheduling algorithms for data processing clusters. In Proceedings of the ACM special interest group on data communication (pp. 270-288).

[11] Federated Learning: A Paradigm Shift in Data Privacy and Model Training, Medium, 2024. Online. https://medium.com/@cloudhacks_/federated-learning-a-paradigm-shift-in-data-privacy-and-model-training-a41519c5fd7e

[12] Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., ... & Ng, A. (2012). Large scale distributed deep networks. Advances in neural information processing systems, 25.

[13] Konečný, J., McMahan, B., & Ramage, D. (2015). Federated optimization: Distributed optimization beyond the datacenter. arXiv preprint arXiv:1511.03575.

[14] McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017, April). Communication-efficient learning of deep networks from decentralized data. In Artificial intelligence and statistics (pp. 1273-1282). PMLR.

[15] Data security in AI systems: Types of threats, principles and techniques to mitigate them and best practices, leewayhertz. Online. https://www.leewayhertz.com/data-security-in-ai-systems/

[16] Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST), 10(2), 1-19.

[17] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., ... & Seth, K. (2017, October). Practical secure aggregation for privacy-preserving machine learning. In proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1175-1191).

[18] Dimakis, A. G., Kar, S., Moura, J. M., Rabbat, M. G., & Scaglione, A. (2010). Gossip algorithms for distributed signal processing. Proceedings of the IEEE, 98(11), 1847-1864.

[19] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., ... & Stoica, I. (2016). Apache spark: a unified engine for big data processing. Communications of the ACM, 59(11), 56-65.

[20] Melnik, S., Gubarev, A., Long, J. J., Romer, G., Shivakumar, S., Tolton, M., & Vassilakis, T. (2010). Dremel: interactive analysis of web-scale datasets. Proceedings of the VLDB Endowment, 3(1-2), 330-339.

[21] Dantuluri, V. N. R. (2025). AI-Powered Query Optimization in Multitenant Database Systems. Journal of Computer Science and Technology Studies, 7(4), 802-813.

[22] Jayaram, Y., & Bhat, J. (2025). Autonomous AI Agents for Campus Knowledge Hubs: A Secure and Intelligent

System Architecture. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, *6*(4), 150-161. https://doi.org/10.63282/3050-9262.IJAIDSML-V6I4P120

[23] Bhat, J., & Sundar, D. (2025). Leveraging Generative AI in ERP Systems: Use Cases for Higher Education and Public Sector Operations. American International Journal of Computer Science and Technology, 7(6), 57-69. https://doi.org/10.63282/3117-5481/AIJCST-V7I6P106

[24] Sundar, D., & Jayaram, Y. (2025). AI-Powered Credential Intelligence and Degree Discovery Frameworks for Academic Pathway Analysis. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, *6*(2), 161-171. https://doi.org/10.63282/3050-9262.IJAIDSML-V6I2P118

[25] Jayaram, Y., & Sundar, D. (2025). Multi-Cloud ECM/WCM Orchestration with AI: A Scalable and Intelligent Enterprise Architecture. *American International Journal of Computer Science and Technology*, *7*(1), 96-110. https://doi.org/10.63282/3117-5481/AIJCST-V7I1P108

[26] Sundar, D. (2025). Reinforcement Learning Techniques for Autonomous Cloud Optimization and Adaptive Resource Management. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, *6*(3), 134-145. https://doi.org/10.63282/3050-9262.IJAIDSML-V6I3P119

[27] Bhat, J., & Jayaram, Y. (2025). AI-Enhanced Integrations: Secure API Management for Multi-Cloud ERP Environments. *International Journal of Emerging Trends in Computer Science and Information Technology*, *6*(3), 94-103. https://doi.org/10.63282/3050-9246.IJETCSIT-V6I3P115

[28] Jayaram, Y. (2025). AI-Powered ECM Automation with Agentic AI for Adaptive, Policy-Driven Content Processing Pipelines. International Journal of Artificial Intelligence, Data Science, and Machine Learning, 6(3), 125-134. https://doi.org/10.63282/3050-9262.IJAIDSML-V6I3P118

[29] Bhat, J. (2025). Augmenting the Public Sector Workforce with AI Assistants and Intelligent Automation. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, *6*(4), 162-171. https://doi.org/10.63282/3050-9262.IJAIDSML-V6I4P121

[30] Sundar, D., & Bhat, J. (2025). Lakehouse-Integrated Graph Risk Scoring Architectures for Advanced Fraud Detection. American International Journal of Computer Science and Technology, 7(6), 70-80. https://doi.org/10.63282/3117-5481/AIJCST-V7I6P107