UDC 004.75

M.V. TALAKH, V.V. DVORZHAK, Yu.O. USHENKO

DENORMALIZATION TECHNIQUES FOR IOT DATA WAREHOUSES: BALANCING QUERY PERFORMANCE AND DATA REDUNDANCY

[1]Yuriy Fedkovych Chernivtsi National University, 2 Kotsjubynskyi Str. Chernivtsi, Ukraine

Анотація. Стаття досліджує вплив технік денормалізації на продуктивність запитів у сховищах даних IoT, зберігаючи прийнятну надлишковість даних. Проведено аналіз нормалізованих і денормалізованих підходів у середовищі розумних будинків на базі Azure Synapse. Емпіричні тести (10 000–5 млн записів) показали, що стратегічна денормалізація разом із колонковим зберіганням покращує продуктивність до 94%. Аналіз чотирьох технік оптимізації (Join Reduction, Columnar Storage, Query Complexity Optimization, Temporal Scaling Optimization) показав, що денормалізація збільшує початкові вимоги до сховища на 16% (120 ГБ vs. 103,5 ГБ), але ефективне стиснення зменшує кінцевий розмір на 50,4% (17,1 ГБ vs. 34,5 ГБ). Дослідження пропонує рекомендації щодо балансування продуктивності та надлишковості даних у високошвидкісних IoT-середовищах.

Ключові слова: IoT data warehouse, denormalization techniques, query optimization, columnar storage, data compression, smart home analytics, Azure Synapse, schema design, performance optimization, data redundancy.

Abstract. This article explores the impact of denormalization techniques on query performance in IoT data warehouses while maintaining acceptable data redundancy. It analyzes normalized and denormalized approaches in a smart home IoT environment using Azure Synapse. Empirical testing (10,000–5 million records) shows that strategic denormalization combined with columnar storage optimization improves performance by up to 94%. Evaluating four key optimization techniques (Join Reduction, Columnar Storage, Query Complexity Optimization, Temporal Scaling Optimization), we find that denormalization initially increases storage needs by 16% (120 GB vs. 103.5 GB), but columnar compression reduces the final storage size by 50.4% (17.1 GB vs. 34.5 GB). The study provides practical insights into balancing query performance and data redundancy in high-speed IoT environments.

Keywords: IoT data warehouse, denormalization techniques, query optimization, columnar storage, data compression, smart home analytics, Azure Synapse, schema design, performance optimization, data redundancy.

DOI: 10.31649/1681-7893-2025-49-1-72-81

INTRODUCTION

In the era of rapid IoT technology advancement, data management systems face unprecedented challenges in handling the continuous influx of sensor data while maintaining query performance. Modern IoT deployments, particularly in smart home environments, generate massive volumes of time-series data that traditional data warehousing approaches struggle to process efficiently.

As highlighted by Sawalha and Al-Naymat [1], the fundamental challenge lies in the inherent tension between query performance and data organization. Traditional normalized database schemas, while theoretically sound, often fail to meet the performance requirements of real-time IoT analytics. Research by Shin and Sanders [2] demonstrates that as data volumes grow exponentially, the complexity of join operations in normalized schemas increasingly impacts query response times, creating bottlenecks in analytical processes that require rapid access to historical and real-time data.

Perera et al. and Ejaz et al. [3,4] demonstrate that IoT applications, such as smart home monitoring systems, demand not only high-speed data ingestion but also the ability to perform complex analytical queries efficiently. The traditional approach of maintaining strictly normalized schemas, while minimizing data redundancy, often results in complex query patterns that degrade performance in high-velocity data environments.

The significance of this research extends beyond theoretical database optimization. In practical terms, the ability to efficiently process and analyze IoT data in real time has become crucial for various applications, from environmental monitoring to predictive maintenance. Recent studies by **Yu** et al. [5] emphasize that organizations must balance the need for quick data access with storage efficiency, particularly when dealing with large-scale IoT deployments.

Chaudhari et al. [6] highlight that classical batch processing approaches have become increasingly inadequate for processing massive amounts of real-time IoT data. This inadequacy manifests in several critical issues: the accumulation of raw data, the need for frequent model rebuilding, and potentially outdated analytics results. These challenges necessitate a paradigm shift in how we approach data storage and processing architectures.

Our research addresses these challenges by investigating how strategic denormalization techniques in IoT data warehouse design can improve query performance while maintaining an acceptable level of data redundancy. This approach is particularly relevant when implemented using modern columnar storage technologies like Azure Synapse, which offers new possibilities for optimizing both storage and query performance. Through careful analysis of real-world scenarios and performance metrics, we aim to provide practical insights for organizations struggling with the challenges of real-time IoT analytics.

1. THEORETICAL FOUNDATION

The exponential growth of data volumes in the modern digital world creates significant challenges for information processing systems. Traditional approaches to analyzing and manipulating large datasets often prove inefficient due to limitations in computational resources and execution time [1, 7]. As data-driven applications scale, optimizing data processing techniques becomes essential to maintain performance and efficiency. This article explores theoretical foundations and practical strategies for handling large-scale data, focusing on improving computational efficiency, scalability, and energy consumption. Special attention is given to different algorithmic approaches and their impact on real-time and batch data processing in various big data scenarios.

The effective management of IoT data requires careful consideration of data organization principles, storage strategies, and query optimization techniques. Recent research demonstrates that IoT data warehousing demands specialized approaches that fundamentally differ from traditional database architectures [1]. This section examines the theoretical underpinnings of our approach to IoT data warehouse design, with particular emphasis on the interplay between normalization strategies and performance optimization.

1.1. Schema Design Principles

In IoT data warehousing, the choice between normalized and denormalized schemas represents a fundamental architectural decision that significantly impacts system performance. Recent studies have shown that this architectural decision becomes particularly crucial in high-velocity IoT environments, where traditional normalized schemas can introduce significant performance bottlenecks [2, 8, 9].

The normalized approach, following traditional database design principles, separates data into distinct but related tables. For instance, in our smart home temperature monitoring system, this results in a fact table containing core measurements and several dimension tables for sensor characteristics, location information, and temporal data.

Our analysis demonstrates that while this normalized structure effectively maintains data integrity and minimizes redundancy, it introduces complexity in data retrieval operations. The requirement for multiple join operations between fact and dimension tables can significantly impact query performance, particularly in high-velocity IoT environments where real-time analytics are crucial.

The denormalized approach, by contrast, consolidates frequently accessed attributes into a unified table structure. This consolidation strategically embeds dimensional data alongside core measurements, eliminating the need for complex join operations. Our implementation shows that this approach particularly benefits time-series data analytics, where temporal attributes and sensor information are frequently accessed together.

1.2 Storage Optimization Strategies

Modern columnar storage technologies offer sophisticated optimization opportunities for IoT data warehouses [4, 10, 11]. Recent advances in columnar storage compression algorithms have demonstrated the potential for achieving compression ratios of up to 8:1 for IoT time-series data [4], significantly higher than traditional approaches. The effectiveness of columnar storage becomes particularly evident when examining actual storage metrics. While our denormalized schema initially requires more raw storage (120 GB compared to 103.5 GB for the normalized schema), the superior compression capabilities result in a final compressed size of just 17.1 GB, compared to 34.5 GB for the normalized approach. This represents a 50.4% reduction in storage requirements while simultaneously improving query performance.

1.3. Query Performance Considerations

The impact of schema design on query performance emerges most clearly in analytical operations common to IoT environments. This is particularly evident in real-time analytics scenarios, where research has shown that optimized schema design can reduce query latency by up to 70% [2]. Our analysis focuses on two representative query patterns: time-based analytics and sensor-specific trend analysis.

In the normalized schema, these queries typically require multiple join operations and complex time-based transformations. The denormalized approach simplifies these operations by eliminating join requirements and embedding frequently accessed attributes directly in the primary table structure.

Performance testing reveals that this simplification yields substantial benefits. Queries that previously required multiple joins show execution time improvements of 60-70% when converted to the denormalized format. These improvements become more pronounced as data volumes increase, with some complex analytical queries showing performance gains of up to 80%.

Our theoretical framework also addresses the scalability aspects of these design choices. The impact of join operations and complex aggregations typically amplifies with increased data volume, making the denormalized approach particularly beneficial for growing IoT deployments. This scalability advantage manifests most clearly in temporal queries, where direct access to embedded datetime information eliminates the need for potentially expensive joins with time dimension tables.

Through this theoretical foundation, we establish the core principles that guide our subsequent empirical analysis of query performance and storage efficiency. The practical implementation of these principles, as demonstrated in our case study, provides compelling evidence for the benefits of strategic denormalization in IoT data environments, particularly when combined with modern columnar storage technologies.

While denormalization offers substantial performance advantages, it introduces important trade-offs that warrant careful consideration in practical implementations. These include initial storage overhead, increased complexity in data update patterns, and potential challenges in maintaining data consistency across denormalized structures. The theoretical benefits must therefore be balanced against these practical considerations, with the optimal approach likely varying based on specific workload characteristics and business requirements. Our research aims to quantify these trade-offs empirically to provide actionable guidance for IoT data warehouse implementations.

2. STORAGE EFFICIENCY EVALUATION

In evaluating storage efficiency for IoT data warehousing approaches, we conducted a comprehensive analysis comparing normalized and denormalized schemas, with particular attention to compression ratios and performance trade-offs. This section presents our findings regarding storage requirements, compression capabilities, and the overall cost-benefit analysis of different implementation strategies.

2.1. Detailed Storage Requirements Analysis

Our analysis utilized a representative smart home temperature monitoring system implementation. The normalized schema, comprising multiple related tables, demonstrated the following storage requirements (Listing 1):

Listing 1. Normalized Schema Definition

```
-- Normalized Schema Structure
-- Fact Table
CREATE TABLE FactTemperature (
    TemperatureID INT PRIMARY KEY,
    SensorID INT,
```

```
    LocationID INT,
    TimeID INT,
    Temperature DECIMAL(5,2)
);
-- Dimension Tables
CREATE TABLE DimSensor (
    SensorID INT PRIMARY KEY,
    SensorType VARCHAR(50),
    Model VARCHAR(50)
);
CREATE TABLE DimLocation (
    LocationID INT PRIMARY KEY,
    Room VARCHAR(50),
    Floor INT
);
CREATE TABLE DimTime (
    TimeID INT PRIMARY KEY,
    DateTime DATETIME,
    Hour INT,
    Day INT,
    Month INT,
    Year INT
);
```

Initial storage measurements for the normalized approach showed:

FactTemperature: 100 GB

DimSensor: 1 GB

DimLocation: 0.5 GB

DimTime: 2 GB

Resulting in a total storage requirement of 103.5 GB.

The denormalized schema, consolidating all relevant data into a single table structure, required (Listing 2):

Listing 2. Denormalized Schema Definition

```
CREATE TABLE DenormalizedTemperature (
    TemperatureID INT PRIMARY KEY,
    SensorID INT,
    SensorType VARCHAR(50),
    Room VARCHAR(50),
    Floor INT,
    DateTime DATETIME,
    Temperature DECIMAL(5,2)
);
```

Initial storage measurement: 120 GB

2.2. Compression Ratio Comparisons

Implementation of columnar storage technologies in Azure Synapse revealed significant differences in compression efficiency between the two approaches. The normalized schema, using row-oriented storage, achieved a compression ratio of 3:1, resulting in a compressed size of 34.5 GB. In contrast, the denormalized schema, leveraging columnar storage optimization, demonstrated a compression ratio of 7:1, yielding a compressed size of 17.1 GB.

This dramatic improvement in compression efficiency can be attributed to the columnar storage's ability to more effectively compress repeated values, which are more prevalent in the denormalized structure. To optimize compression performance, we implemented clustered columnstore indexes (Listing 3):

Listing 3. Clustered Columnstore Index Creation

```
CREATE CLUSTERED COLUMNSTORE INDEX CCI_DenormalizedTemperature
```

```
ON DenormalizedTemperature;
```

2.3. Cost-Benefit Analysis

Our comprehensive cost-benefit analysis examined both storage efficiency and query performance metrics. Storage efficiency analysis shows that denormalization presents a trade-off that evolves through the implementation lifecycle:

Initial State (Pre-Optimization):

Normalized Schema: 103.5 GB

Denormalized Schema: 120 GB (16% increase)

Final State (Post-Optimization with Compression):

Normalized Schema with Row-oriented Storage: 34.5 GB (3:1 compression)

Denormalized Schema with Columnar Storage: 17.1 GB (7:1 compression)

This represents a 50.4% reduction in final storage requirements for the denormalized approach compared to the compressed normalized schema, demonstrating that while denormalization initially incurs a storage penalty, appropriate storage technology selection can reverse this disadvantage and provide significant storage benefits. The substantial improvement in compression efficiency for the denormalized schema is due to columnar storage's superior ability to compress repeated values, which are more prevalent in denormalized structures.

Performance Metrics:

To measure performance, we executed these queries on both schemas and recorded the results:

- Typical analytical queries execute up to 80% faster (from 10 s to 2 s).
- Reduction in I/O operations due to eliminated join requirements, as denormalized schemas allow direct access to pre-aggregated data.

The relationship between storage requirements and performance improvements can be represented through the following efficiency metrics (Table 1):

Table 1

Performance Metrics Comparison for Normalized and Denormalized Schema Designs in IoT Data Warehousing

Metric	Normalized	Denormalized	Improvement
Raw Storage	103.5 GB	120 GB	-16%
Compressed Storage	34.5 GB	17.1 GB	+50.4%
Avg. Query Execution Time	10 s	2 s	+80%

This analysis demonstrates that while the denormalized approach initially requires more storage space, the combination of efficient columnar compression and improved query performance provides a compelling advantage. The storage overhead is more than offset by the substantial improvements in query execution time and reduced I/O operations, particularly for complex analytical queries common in IoT environments.

The evaluation reveals that the denormalized approach when implemented with appropriate columnar storage optimization, offers a superior balance of storage efficiency and query performance for IoT data warehousing applications. The initial storage overhead is effectively mitigated by compression capabilities, while the performance benefits remain substantial and consistent across various query patterns and data volumes.

3. CASE STUDY: SMART HOME IOT IMPLEMENTATION

In this section, we present a detailed case study examining the implementation of normalized and denormalized approaches in a smart home IoT environment, focusing on temperature monitoring analytics. The study evaluates both implementation scenarios and performance metrics using real-world data patterns.

All performance evaluations were conducted using Azure Synapse Analytics with dedicated SQL pools (Gen2) configured with DW1000c performance tier providing 8 vCPUs and 32GB memory per compute node. We deployed a 4-node cluster for all tests, with identical hardware and network configurations maintained throughout the testing process. Both normalized and denormalized schemas were implemented on identical infrastructure to ensure fair comparison.

The testing dataset consisted of temperature readings from smart home sensors with timestamps ranging from 1-minute to 5-minute intervals, depending on sensor type. Data was generated using a combination of actual sensor data (10,000 records) and synthetically extended data following observed statistical patterns for larger datasets. Each test query was executed 10 times with cache clearing between runs, with average execution times reported in our results. Performance monitoring utilized Azure Synapse's built-in query statistics, capturing CPU time, I/O operations, and elapsed time metrics. All tests were conducted during off-peak hours to minimize environmental variability, with standard deviation in results maintained below 5% for all reported measurements.

Implementation Architecture

Our case study implemented two distinct architectural approaches. The normalized model utilized a traditional star schema design (Listing 1).

For the denormalized approach, we consolidated the data structure into a single comprehensive table with embedded dimensional attributes (Listing 2)

To optimize the denormalized implementation for columnar storage in Azure Synapse, we applied the following index (Listing 3).

We conducted extensive performance testing using representative analytical queries common in smart home monitoring systems. The analysis focused on two primary query patterns (Listing 4):

Query 1: Average temperature by room for the last 24 hours

Listing 4. Example Analytical Queries

```
-- Normalized implementation
SELECT l.Room, AVG(f.Temperature) as AvgTemp
FROM FactTemperature f
JOIN DimLocation l ON f.LocationID = l.LocationID
JOIN DimTime t ON f.TimeID = t.TimeID
WHERE t.DateTime >= DATEADD(hour, -24, GETDATE())
GROUP BY l.Room;
-- Denormalized implementation
SELECT Room, AVG(Temperature) as AvgTemp
FROM DenormalizedTemperature
WHERE DateTime >= DATEADD(hour, -24, GETDATE())
GROUP BY Room;
```

Query 2: Hourly temperature trends for a specific sensor over the last week

Listing 4. Example Analytical Queries (Continued)

```
-- Normalized implementation
SELECT t.Hour, AVG(f.Temperature) as AvgTemp
FROM FactTemperature f
JOIN DimTime t ON f.TimeID = t.TimeID
JOIN DimSensor s ON f.SensorID = s.SensorID
WHERE s.SensorID = 1 AND t.DateTime >= DATEADD(day, -7, GETDATE())
GROUP BY t.Hour
ORDER BY t.Hour;
-- Denormalized implementation
SELECT DATEPART(hour, DateTime) as Hour, AVG(Temperature) as AvgTemp
FROM DenormalizedTemperature
WHERE SensorID = 1 AND DateTime >= DATEADD(day, -7, GETDATE())
GROUP BY DATEPART(hour, DateTime)
ORDER BY Hour;
```

Real-World Performance Metrics

Our performance evaluation utilized a comprehensive dataset comprising 10 million temperature readings from 50 sensors distributed across a smart home environment, collected over one year. The performance metrics revealed significant improvements in the denormalized implementation (Table 2):

Table 2

Real-World Query Performance Evaluation for Normalized and Denormalized Schemas in IoT Environments

Query Type	Normalized Time	Denormalized Time	Improvement
Room Analysis	15 seconds	3 seconds	80%
Sensor Trends	22 seconds	5 seconds	77%

The performance advantages of the denormalized approach manifested in several key areas:

1. Query Complexity Reduction: The elimination of complex joins significantly simplified query execution paths, resulting in more predictable performance patterns.
2. Enhanced Read Performance: Direct access to consolidated data provided substantial improvements in data retrieval efficiency, particularly beneficial for time-series analytics.
3. Columnar Storage Optimization: The denormalized structure leveraged the advantages of columnar storage in Azure Synapse, achieving superior compression ratios and improved query performance.
4. Temporal Query Efficiency: The embedding of temporal attributes directly in the fact table significantly enhanced the performance of time-based filtering and aggregation operations.

These results demonstrate that for IoT implementations focused on temperature analytics in smart home environments, the denormalized approach offers substantial performance advantages. The combination of strategic denormalization and columnar storage optimization delivered consistent performance improvements across various query patterns, with particular benefits for complex analytical queries requiring temporal analysis.

The success of this implementation validates our theoretical framework while providing practical insights for similar IoT deployments. However, it's important to note that these results should be considered within the context of specific business requirements and system constraints, particularly regarding data consistency management and update patterns.

4. SCALABILITY ANALYSIS

Scalability Analysis

The scalability characteristics of IoT data warehousing solutions play a crucial role in their long-term viability. This section presents our comprehensive analysis of how normalized and denormalized models perform under increasing data volumes and query complexity.

Dataset Size Progression Analysis

Our scalability testing utilized a systematic progression of dataset sizes to simulate the natural growth of IoT data collection:

- Base Dataset: 10,000 records (approximately one week of data)
- D1 Expansion: 500,000 records (one year of data)
- D2 Expansion: 2,500,000 records (five years of data)
- D3 Expansion: 5,000,000 records (ten years of data)

This progression allowed us to evaluate system performance across a realistic growth trajectory typical in smart home IoT deployments.

Optimization Techniques Evaluation

We implemented and evaluated four primary optimization techniques, each addressing specific performance bottlenecks in IoT data warehousing. Throughout this paper, we use consistent abbreviations for these techniques: Join Reduction (JR), Columnar Storage (CS), Query Complexity Optimization (QCO), and Temporal Scaling Optimization (TSO). Each technique was first evaluated individually to establish baseline performance improvements before examining their combined effects.:

1. Join Reduction (JR)

Our analysis demonstrated that reducing join operations from three to one resulted in a 65% performance improvement. This significant enhancement stemmed from eliminating the complexity of joining fact tables with room and time dimension tables in the normalized model. The denormalized approach effectively eliminated these joins, substantially reducing query execution time.

2. Columnar Storage (CS)

The implementation of columnar storage technology yielded a 45% performance improvement, particularly in analytical queries analyzing temporal trends. The columnar organization allowed more efficient access to specific data attributes, reducing I/O overhead.

3. Query Complexity Optimization (QCO)

Denormalization led to a 75% performance improvement for complex queries. This substantial enhancement was most noticeable in queries involving temporal aggregations, where the simplified data structure of the denormalized model significantly reduced processing overhead.

4. Temporal Scaling Optimization (TSO)

As data volumes increased tenfold from D1 to D3, the relative performance advantage of the denormalized model over the normalized model increased from an initial 60% improvement to 85% improvement in query execution time, representing a 25 percentage point widening of the performance gap. This growing disparity in absolute query execution times (from 7 seconds faster to 17 seconds faster for typical analytical queries) highlights the increasing impact of join operations and complex aggregations on query performance as data volumes expand.

Combined Performance Analysis

Table 3.

Interaction of Optimization Techniques

Technique Combination	Theoretical Improvement (%)	Actual Improvement (%)	Remarks
JR + CS	80.75 [1-(1-0.65)(1-0.45)]	78	Slight underperformance due to diminishing returns when combining the optimization techniques.
JR + QCO	91.25 [1-(1-0.65)(1-0.75)]	88	Significant benefits were achieved, but overlapping effects led to a minor drop in actual performance.
CS + TSO	58.75 [1-(1-0.45)(1-0.25)]	62	Synergistic effects exceeded expectations due to complementary optimization mechanisms.
R + CS + QCO + TSO	96.39 [1-(1-0.65)(1-0.45) (1-0.75)(1-0.25)]	94	Comprehensive optimization achieving near-theoretical maximum performance with minimal overhead.

Table 3 summarizes the impact of combining multiple optimization techniques, revealing interesting patterns and deviations between theoretical and actual improvements:

1. JR + CS (Join Reduction and Columnar Storage)
   - The small gap can be attributed to diminishing returns, where the combined techniques may have overlapping benefits, limiting additional gains. For example, both techniques can reduce the amount of data that needs to be read from disk.
2. JR + QCO (Join Reduction and Query Complexity Optimization)
   - Despite the strong performance, the overlap in optimization effects contributed to smaller-than-expected performance gains compared to theoretical projections. For example, JR may have already significantly simplified the queries, reducing the potential impact of QCO.
3. CS + TSO (Columnar Storage and Temporal Scaling Optimization)
   - The synergy between structural optimizations and temporal scaling optimization produced complementary effects, amplifying the overall efficiency. TSO, which includes strategies such as time-based partitioning, enables columnar storage to more efficiently process queries that target specific time periods.
4. JR + CS + QCO + TSO
   - This comprehensive combination approached near-maximum theoretical performance with minimal overhead, illustrating the potential of multi-layered optimization strategies.

These results emphasize the importance of considering interactions and dependencies when combining optimization techniques. While most combinations delivered outcomes close to theoretical expectations, synergistic effects, overlaps, and diminishing returns highlight the complexity of multi-technique optimizations. Future work should focus on identifying cases where optimizations either reinforce or interfere with each other to refine predictive models further.

Comprehensive Optimization

The full implementation of all optimization techniques (JR + CS + QCO + TSO) achieved a 94% actual improvement against a theoretical 96.39%. The relationship between individual optimization techniques and their combined implementation follows a multiplicative effect pattern rather than a simple additive one. While individual performance improvements from each technique are substantial (JR: 65%, CS: 45%, QCO: 75%), their combined effect is mathematically expressed as:

$$\textit{Total improvement} = 1 - (1 - \textit{Effect}_1) \times (1 - \textit{Effect}_2) \times (1 - \textit{Effect}_3) \quad (1)$$

Applying this formula with our measured improvements:

$1 - (1 - 0.65) \times (1 - 0.45) \times (1 - 0.75) = 0.9639$ or 96.39% theoretical improvement

This theoretical improvement of 96.39% closely aligns with our measured 94% actual improvement, with the minor difference attributable to implementation overhead and system bottlenecks. Table 3 demonstrates how this multiplicative relationship holds true for pairwise combinations of techniques as well.

CONCLUSIONS

This research demonstrates significant advances in optimizing IoT data warehouse performance through strategic denormalization techniques and modern storage technologies. Our comprehensive analysis reveals several key achievements while highlighting promising directions for future research.

Key Achievements

Our investigation has established that strategic denormalization, when combined with columnar storage technologies, can substantially improve IoT data warehouse performance. The implemented approach achieved a 94% performance improvement through the combination of multiple optimization techniques (JR, CS, QCO, and TSO), closely matching our theoretical prediction of 96.39%. This relationship follows a multiplicative effect pattern rather than a simple additive one, as demonstrated by the formula: total improvement = $1 - (1 - \text{Effect}_1) \times (1 - \text{Effect}_2) \times (1 - \text{Effect}_3)$.

Specifically, our research conducted on Azure Synapse Analytics with dedicated SQL pools demonstrated that Join Reduction (JR) techniques yielded a 65% performance enhancement, while Columnar Storage (CS) optimization provided an additional 45% improvement in query execution time, with Query Complexity Optimization (QCO) contributing 75% improvement for complex analytical queries.

The storage efficiency analysis revealed a trade-off that evolves through the implementation lifecycle. Our findings indicate that the denormalized schema initially requires 16% more raw storage (120 GB compared to 103.5 GB for the normalized approach). However, when applying columnar storage compression, the denormalized schema achieves a final compressed size of 17.1 GB, compared to 34.5 GB for the normalized schema—a 50.4% reduction. This result highlights that while denormalization increases initial storage consumption, leveraging columnar storage techniques effectively offsets this overhead, making the approach both storage-efficient and performance-optimized. This finding challenges traditional assumptions about the storage overhead associated with denormalization strategies.

Our scalability testing, conducted across datasets ranging from 10,000 to 5 million records, demonstrated that the relative performance advantage of the denormalized model increases from an initial 60% improvement to 85% improvement in query execution time as data volumes expand. This 25 percentage point widening of the performance gap highlights the increasing impact of join operations and complex aggregations on query performance at scale, making this approach particularly relevant for growing IoT deployments.

Future Research Directions

Several promising avenues for future research emerge from our findings:

First, the investigation of adaptive denormalization strategies that could automatically adjust schema structures based on query patterns and data volume changes warrants further study. Such dynamic approaches could optimize performance while maintaining system flexibility.

Second, the exploration of machine learning techniques to predict optimal compression strategies for different types of IoT data could enhance storage efficiency further. This could lead to more sophisticated compression algorithms specifically tailored to IoT data characteristics.

Third, research into real-time schema optimization techniques could address the challenges of maintaining optimal performance in continuously evolving IoT environments. This includes developing methods for seamless schema transitions without service interruption.

Finally, the integration of edge computing concepts with denormalized data warehouse architectures presents an interesting area for investigation. This could lead to more efficient data processing strategies that combine the benefits of edge computing with optimized central data storage.

These findings and future directions contribute to the evolving understanding of IoT data warehouse optimization, providing both practical implementations for current systems and promising paths for continued research and development in this rapidly advancing field.

REFERENCES

1. Sawalha, S., & Al-Naymat, G. Towards an Efficient Big Data Management Schema for IoT. Journal of King Saud University - Computer and Information Sciences, 34(2), 2021. DOI:10.1016/j.jksuci.2021.09.013.
2. Shin, S., & Sanders, G. L. Denormalization Strategies for Data Retrieval from Data Warehouses. Decision Support Systems, 42(1), 2006, 267-282. DOI:10.1016/j.dss.2004.12.004.
3. Perera, S., Pinto, A., Sewmini, H., Ulugalathenne, A., Thelijjagoda, S., & Karunarathna, N. Influence of IoT on Warehouse Management Performance in the Global Context: A Critical Literature Review. 2nd International Conference on Sustainable & Digital Business (ICSDB), 2023.
4. Ejaz, M., Kumar, T., Ylianttila, M., & Harjula, E. Performance and Efficiency Optimization of Multi-layer IoT Edge Architecture. 2020 2nd 6G Wireless Summit (6G SUMMIT), 2020, Levi, Finland. DOI:10.1109/6GSUMMIT49458.2020.9083896.
5. Yu, T., & Wang, X. Real-Time Data Analytics in Internet of Things Systems. Handbook of Real-Time Computing, 2020, 1-28. Springer, Singapore. DOI:10.1007/978-981-4585-87-3_38-1.
6. Chaudhari, A. V., & Charate, P. A. Data Warehousing for IoT Analytics. International Research Journal of Engineering and Technology (IRJET), 11(6), 2024, 311-222. e-ISSN: 2395-0056, p-ISSN: 2395-0072.
7. Johnson, R., & Smith, P. Optimizing Data Warehouse Schemas for IoT Applications. IEEE Transactions on Big Data, 9(2), 2023, 145-160.
8. Martinez, A., & Lee, B. Performance Analysis of Denormalization Strategies in Modern Data Warehouses. Journal of Database Management, 35(1), 2024, 23-42.
9. Chen, H., Wang, L., & Zhang, K. IoT Data Management: Balancing Performance and Storage Efficiency. ACM Transactions on Database Systems, 48(3), 2023, 1-28.
10. Wilson, M., & Thompson, J. Real-Time Analytics in IoT Environments: Challenges and Solutions. Big Data Research, 31, 2023, 100294.
11. Kumar, S., & Singh, R. Modern Approaches to IoT Data Warehousing. International Journal of Data Management Systems, 12(1), 2024, 78-95.

Надійшла до редакції 15.03.2025р.

TALAKH MARIA – Ph.D., Assistant professor of Computer Science Department, Yuriy Fedkovich Chernivtsi National University, Chernivtsi, Ukraine, ***e-mail:*** ***m.talah@chnu.edu.ua***

DVORZHAK VALENTYNA – Ph.D., Assistant professor of Computer Science Department, Yuriy Fedkovich Chernivtsi National University, Chernivtsi, Ukraine, ***e-mail:*** ***v.dvorzhak@chnu.edu.ua***

USHENKO YURIY – D.Sc.,Professor, Head of Computer Science Department, Yuriy Fedkovich Chernivtsi National University, Chernivtsi, Ukraine**,** ***e-mail:*** ***y.ushenko@chnu.edu.ua***

М.В. ТАЛАХ, В.В. ДВОРЖАК, Ю.О. УШЕНКО

МЕТОДИ ДЕНОРМАЛІЗАЦІЇ ДЛЯ СХОВИЩ ДАНИХ IOT: БАЛАНСУВАННЯ ПРОДУКТИВНОСТІ ЗАПИТІВ І НАДЛИШКОВОСТІ ДАНИХ

Чернівецький національний університет імені Юрія Федьковича