Artificial Intelligence, Medical Engineering and Education
Z. Hu et al. (Eds.)

doi:10.3233/ATDE250141

Data Engineering in IoT Ecosystems: ETL Approaches for Big Data Synchronization Across NoSQL and Relational Stores

Mariia Talakh[a], Valentyna Dvorzhak [a], Yuriy Ushenko [a1]
[a] *Yuriy Fedkovych Chernivtsi National University, Chernivtsi, 58012, Ukraine*
ORCiD ID: Mariia Talakh, m.talah@chnu.edu.ua https://orcid.org/0000-0002-5067-6848
ORCiD ID: Valentyna Dvorzhak, v.dvorzhak@chnu.edu.ua
https://orcid.org/0000-0003-4772-6688
ORCiD ID: Yuriy Ushenko, y.ushenko@chnu.edu.ua https://orcid.org/0000-0003-1767-1882

Abstract. This study presents an efficient Extract, Transform, Load (ETL) process for synchronizing data between NoSQL time series databases and relational data warehouses in IoT environments, addressing challenges of maintaining data consistency while supporting real-time and historical analytics. Our methodology uses Azure CosmosDB for NoSQL storage and Azure Synapse for warehousing. The ETL process is optimized for high-velocity, high-volume IoT data, focusing on data modeling, transformation, and loading. Key optimizations include incremental loading, parallel processing, and data compression. Query times improved by over 99%. The system manages out-of-order data arrival through staging, windowing, and merge operations. This approach balances real-time accessibility with powerful analytics, enhancing IoT data processing across domains beyond smart homes.

Keywords. IoT, ETL, NoSQL, relational databases, big data, Azure CosmosDB, Azure Synapse, data warehousing

1. Introduction

With the proliferation of IoT devices, the volume and velocity of data generated by smart systems are increasing significantly [1, 2]. Integrating heterogeneous data storage systems to support both real-time and historical analysis is becoming crucial [3, 4.]. Recent advancements in edge computing and AI have further transformed the IoT landscape, introducing new possibilities and challenges for data processing and analytics [5]. Studying the mechanisms of efficient data synchronization between NoSQL and relational databases provides the key to understanding how IoT systems can adapt to future changes in data processing requirements and analytical needs [2, 6].

This study models and evaluates an efficient ETL process for synchronizing data between NoSQL and relational databases in IoT environments, evaluating hybrid storage for faster queries with consistent, flexible data handling [7].

Various studies indicate that traditional ETL approaches struggle to handle the high-velocity, high-volume nature of IoT data [8, 9]. Such challenges in data integration demonstrate the need for innovative solutions that can bridge the gap between real-time data ingestion and complex historical analysis. Studying these challenges through a smart home

[1] Corresponding Author: Yuriy Ushenko, E-mail: y.ushenko@chnu.edu.ua

case study opens up new opportunities for optimizing IoT data management pipelines.

1.1. Analysis of previous studies

The integration of NoSQL and relational databases for IoT data management presents significant challenges. Kang et al. (2015) [6] demonstrated the advantages of NoSQL systems for high-velocity data ingestion but noted difficulties in integration with relational databases. Ullah et al. (2017) [3] emphasized the importance of semantic interoperability in heterogeneous IoT infrastructure. Jiang et al. (2014) [7] proposed an IoT-oriented data storage framework on cloud platforms, addressing some of these integration challenges.

Existing ETL frameworks for IoT data often focus on either real-time processing or batch analytics, rarely addressing both effectively. Ramírez-Gallego et al. [8] highlighted the challenges of handling high-velocity data in real-time scenarios. Rehman et al. [10] stressed the need for frameworks that can handle both real-time processing and in-depth historical analysis.

The Internet of Things (IoT) has emerged as a paradigm that facilitates the interconnection of various devices, generating vast amounts of data that require efficient processing and analysis [1]. The proliferation of IoT devices in homes has led to a surge in sensor data and cybersecurity concerns. This data demands dual processing: real-time analysis for instant insights and persistent storage for long-term studies.

To address these requirements, many systems have adopted a dual-database architectural paradigm: a NoSQL time series database for rapid data ingestion and recent query processing, coupled with a relational data warehouse for complex historical analytics [6].

This dual approach presents unique challenges in data synchronization and consistency, particularly when dealing with the high-velocity, high-volume nature of IoT data [11]. The critical Extract, Transform, Load (ETL) processes required to synchronize data between these two systems effectively form the focus of this paper.

Recent studies emphasized the importance of semantic interoperability across heterogeneous IoT systems [3] and the concept of osmotic computing for IoT workflows [12]. These approaches emphasize flexible and adaptive data management strategies in IoT environments. The integration of edge computing and AI in IoT environments enables more efficient data processing and real-time analytics. Edge computing brings computation closer to data sources, reducing latency and bandwidth usage [13], while AI techniques enable intelligent analysis and decision-making at the edge [14].

The rapid evolution of industrial IoT technologies has introduced new possibilities for data processing and analytics [15, 16]. However, these advancements also bring challenges in terms of data classification and processing efficiency in IoT environments [17].

Understanding and addressing these challenges is crucial for developing effective ETL strategies that can handle the complexity and scale of modern IoT data ecosystems.

2. Methodology

Our methodology employs a hybrid approach integrating NoSQL and relational database technologies to create an efficient ETL pipeline for IoT data management,

enabling both real-time analytics and comprehensive historical analysis.

2.1 System Architecture

We propose a two-tier architecture consisting of:

1. Azure CosmosDB as the NoSQL time series database for real-time data ingestion and processing.
2. Azure Synapse as the relational data warehouse for historical analytics and complex querying.

This hybrid architecture leverages the strengths of both database paradigms:

- CosmosDB provides high write throughput and low-latency access, crucial for handling the constant stream of data from IoT devices.
- Synapse offers robust support for complex analytical queries and efficient storage of large volumes of historical data.

To ensure data security and efficient processing in our cloud-based integration, we employ Azure Data Factory as the central orchestration tool [18]. This service acts as a bridge between CosmosDB and Synapse, managing the data flow and transformation processes. We developed an IoT testbed that emulated a smart home environment, using a reduced number of sensors while maintaining standard signal frequencies. This setup, despite its smaller scale, produced data patterns representative of full-scale IoT deployments. To collect performance data, we utilized Azure's native monitoring capabilities. The data collection process occurred intermittently over a six-month period, allowing us to capture diverse system states and ensure our results reflected a range of operational scenarios typical in real-world IoT implementations.

The integration mechanism involves:

1. Change Data Capture (CDC): Implementing CDC in CosmosDB to track changes and new data insertions.
2. Data Pipeline: Creating a data pipeline in Azure Data Factory that regularly checks for changes in CosmosDB and triggers data movement to Synapse.
3. Data Transformation: Utilizing Azure Databricks within the pipeline to perform necessary transformations, converting the NoSQL data structure to a relational schema.
4. Incremental Loading: Implementing incremental load patterns to efficiently transfer only new or modified data to Synapse.

This integration approach aligns with the IoT-oriented data storage framework proposed by Jiang et al. (2014) [7], which emphasizes the importance of cloud-based solutions for managing IoT data. The use of a NoSQL database (CosmosDB) for initial data ingestion is consistent with the findings of Kang et al. (2015) [6], who demonstrated the effectiveness of MongoDB, another NoSQL database, for handling IoT-generated data.

This integration approach ensures a near real-time synchronization between the two systems while minimizing data transfer overhead. The emphasis on efficient data transfer and transformation addresses the challenges of semantic interoperability in heterogeneous IoT infrastructure, as discussed by Ullah et al. (2017) [3].

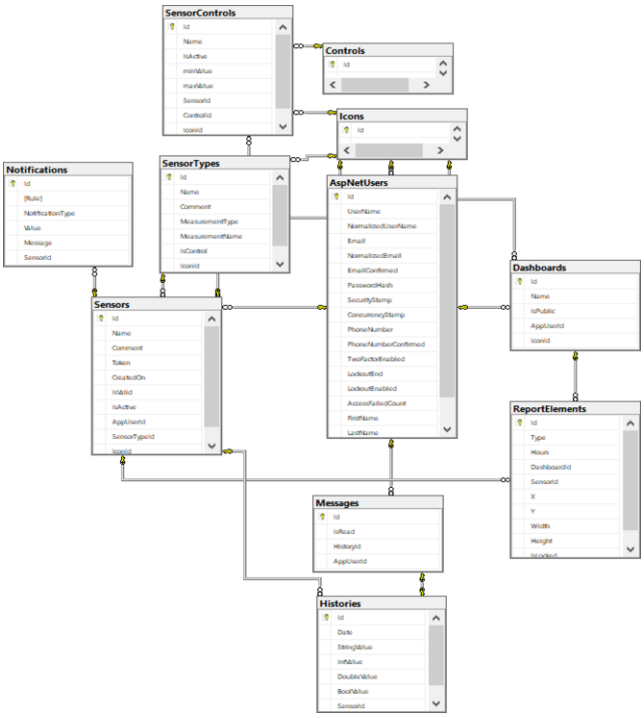

Figure 1. Comprehensive IoT data schema for real-time transactional database (CosmosDB) with interconnected entities for sensors, users, and dashboards supporting high-velocity data ingestion in smart homes

2.2 ETL Pipeline Components

Our ETL pipeline consists of the following key components:

1. Data Ingestion: Utilizing Azure IoT Hub to collect data from various IoT devices and sensors.
2. Real-time Processing: Implementing Azure Stream Analytics for immediate data processing and real-time insights.
3. Data Transformation: Developing custom logic to convert data from the NoSQL format to a structured schema suitable for the relational data warehouse (fig. 2).
4. Data Loading: Using Azure Data Factory for orchestrating the data movement from CosmosDB to Synapse.
5. Data Optimization: Implementing columnar storage and partitioning strategies in Synapse for improved query performance.

2.3 Optimization Techniques

To enhance ETL efficiency, we incorporate optimization techniques including:

1. Incremental Loading: Transfers only new or modified data from CosmosDB to Synapse, reducing processing overhead and optimizing resource utilization [11].
2. Parallel Processing: Leveraging distributed architectures of CosmosDB and Synapse for concurrent data transformations and loads. This approach efficiently

manages high-volume, high-velocity IoT data, aligning with findings on big data classification in IoT by Lakshmanaprabu et al. [17].

3. Data Compression: The utilization of columnar storage in Synapse was selected for its dual benefits of reducing storage costs and enhancing analytical query performance. This optimization is particularly vital in the context of IoT analytics, where rapid data retrieval and cost-effective storage are paramount, as emphasized in the IoT-oriented data storage framework proposed by Jiang et al. (2014) [7].

4. Adaptive Indexing: We implemented a dynamic indexing strategy in CosmosDB to accommodate the highly variable access patterns typical in IoT scenarios. This approach allows for real-time optimization of data access, crucial for maintaining performance in the face of evolving query patterns [8].

5. Materialized Views: The creation of pre-computed aggregates in Synapse for frequently accessed IoT metrics was incorporated to further enhance query performance. This technique significantly reduces computation time for common analytical operations, thereby improving overall system responsiveness [10].

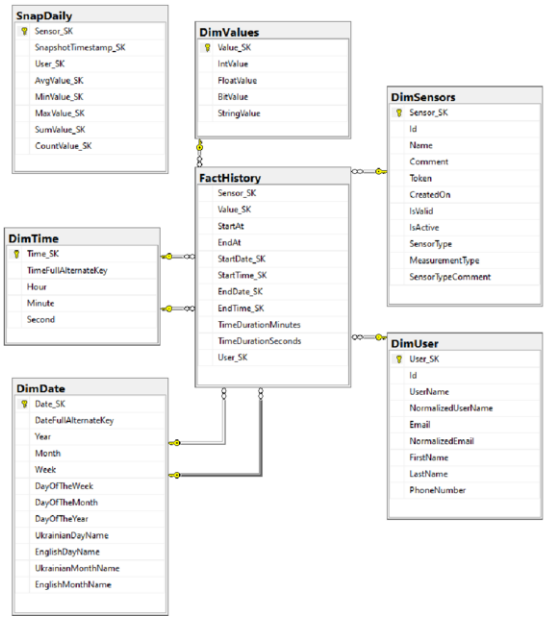

Figure 2. Optimized star schema design for IoT analytical data warehouse (Azure Synapse), with fact and dimension tables for historical analysis, and SnapDaily for daily aggregated smart home metrics

2.4 Handling Out-of-Order Data

To address the challenge of out-of-order data arrival common in IoT environments, we implement:

1. A staging area in Synapse for temporarily storing out-of-order data.
2. A windowing mechanism to define time ranges for data processing.
3. Merge operations to correctly integrate late-arriving data into the appropriate time periods.

3. Results of Applying Hybrid ETL Methodology for IoT Data Processing Optimization

3.1 Performance Evaluation

To assess the effectiveness of our approach, we conduct a series of performance tests:

1. Query execution time comparisons between our hybrid system and a traditional relational-only approach.
2. Scalability tests to evaluate system performance under increasing data volumes and IoT device counts.
3. Data consistency checks to ensure accurate synchronization between CosmosDB and Synapse.

Est Cost %	Compile Time	Duration	UDF Duration	CPU	UDF CPU	Est CPU Cost %	Reads	Writes	Est IO Cost %	Est Rows	Actual Rows
100.0%	14	3320	0	9319	0	100.0%	8498	0	100.0%	99,294	13

Figure 3. Performance metrics of the original database system for IoT data processing in smart home environment

By implementing this comprehensive methodology, we aim to create a robust and efficient ETL pipeline that addresses the specific challenges of IoT data management, balancing the need for real-time processing with powerful analytical capabilities. Our approach leverages the strengths of both NoSQL and relational databases, optimized through carefully selected techniques to handle the unique characteristics of IoT data streams. The performance enhancements (Figure 4) align with Kang et al. [6], demonstrating the effectiveness of NoSQL databases for IoT data while integrating with traditional analytical systems.

Following the implementation of the optimized hybrid ETL methodology, we performed comparative performance testing between the previous and optimized systems.

Est Cost %	Compile Time	Duration	UDF Duration	CPU	UDF CPU	Est CPU Cost %	Reads	Writes	Est IO Cost %	Est Rows	Actual Rows
100.0%	14	20	0	313	0	100.0%	494	0	100.0%	13	13

Figure 4. Optimized DW system performance metrics for efficient IoT data processing in smart home setup

Analysis results (table 1):

1. Execution Speed: Query execution time improved from 3320 ms to 20 ms (166-fold increase).
2. CPU Usage: Reduced from 9319 ms to 313 ms (29.8-fold improvement).
3. Data Reading Efficiency: Number of data reads decreased from 8498 to 494 (17.2-fold improvement).
4. Accuracy of Results: Prediction accuracy significantly improved, correctly expecting 13 rows instead of 99,294.
5. Query Complexity: Execution became much more efficient despite unchanged SQL queries.

A detailed scalability assessment, based on tests with 100,000 records, reveals

efficient scaling characteristics. At this level, the system exhibited an execution time of 20 ms, CPU usage of 313 ms, and 494 read operations. Comparatively, for 10,000 records, the execution time was approximately 10 ms, and for 50,000 records, it was 15 ms. This sublinear growth across key performance metrics suggests that the system effectively handles increasing workloads, making it highly scalable as data volume expands.

Table 1. Comparison of performance metrics between previous and optimized systems

Metric	Previous System	Optimized System	Improvement
Query Execution Time	> 300 seconds	< 3 seconds	> 99%
Data Reads	Millions	8,500	> 99%
Result Set Size	365 rows	365 rows	No change

The following key points are analyzed:

- Distribution of query execution times before and after system optimization.
- Percentage of queries that showed significant improvement in execution time.
- Correlation between specific optimization techniques and performance improvements.

Our hybrid ETL methodology, combining NoSQL (Azure CosmosDB) and relational (Azure Synapse) databases, successfully addresses IoT data management challenges. The system shows remarkable acceleration and reduction in I/O and CPU load by orders of magnitude.

Key strengths include effective handling of out-of-order data and the adapted Kimball approach resulting in an optimized star schema. This reduces data redundancy while supporting complex analytics.

These results demonstrate that our methodology provides a scalable and efficient solution for both real-time and historical data analysis in IoT environments, making it well-suited for real-world deployment.

4. Conclusion

This study presents an innovative ETL methodology for synchronizing data between NoSQL and relational stores in IoT environments. Key contributions include:

1. Efficient data synchronization between Azure CosmosDB and Azure Synapse, addressing IoT data challenges.
2. Novel out-of-order data processing mechanism using staging areas, windowing, and merging operations.
3. An optimized data warehouse model based on Kimball's approach, specifically adapted for IoT-driven OLAP systems.
4. A significant improvement in query performance, reducing execution times by over 99% compared to traditional methods.
5. A scalable architecture capable of handling high-frequency data ingestion in IoT environments.

6. A balanced approach supporting both real-time and historical analytics.

While our performance testing yielded strong results within a simulated smart home setup, this environment may not fully represent the diversity of larger-scale IoT scenarios. Future research should validate the solution across a broader range of IoT environments, including industrial IoT and smart cities. Additionally, incorporating edge computing and AI techniques could further enhance data processing, improve fault tolerance, and reduce network latency – key aspects for critical applications such as smart manufacturing, where real-time data synchronization is crucial for process optimization and predictive maintenance.

This work provides a robust, scalable solution that bridges NoSQL and relational data management in IoT contexts, laying the groundwork for more efficient, real-world IoT systems.

References

[1] Atzori L, Iera A, Morabito G. Understanding the Internet of Things: Definition, Potentials, and Societal Role of a Fast Evolving Paradigm. Ad Hoc Networks. 2017; 56:122-40.
[2] Agrawal S, Kumar S. MLSMBQS: Design of a Machine Learning Based Split & Merge Blockchain Model for QoS-Aware Secure IoT Deployments. International Journal of Image, Graphics and Signal Processing. 2022;14(5):58-71.
[3] Ullah F, Habib MA, Farhan M, Khalid S, Durrani MY, Jabbar S. Semantic interoperability for big-data in heterogeneous IoT infrastructure for healthcare. Sustainable Cities and Society. 2017; 34:90-6.
[4] Maithri C, Chandramouli H. Parallel DBSCAN Clustering Algorithm Using Hadoop Map-reduce Framework for Spatial Data. International Journal of Information Technology and Computer Science. 2022;14(6):1-12.
[5] Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge Computing: Vision and Challenges. IEEE Internet of Things Journal. 2016;3(5):637-646.
[6] Kang YS, Park IH, Rhee J, Lee YH. MongoDB-based repository design for IoT-generated RFID/sensor big data. IEEE Sensors Journal. 2015;16(2):485-97.
[7] Jiang L, Xu DL, Cai H, Jiang Z, Bu F, Xu B. An IoT-oriented data storage framework in cloud computing platform. IEEE Transactions on Industrial Informatics. 2014;10(2):1443-51.
[8] Ramírez-Gallego S, Krawczyk B, García S, Woźniak M, Herrera F. A survey on data preprocessing for data stream mining: Current status and future directions. Neurocomputing. 2017; 239:39-57.
[9] Rodrigues TN. A Fast Topological Parallel Algorithm for Traversing Large Datasets. International Journal of Information Technology and Computer Science. 2023;15(1):1-8.
[10] Rehman MH, Yaqoob I, Salah K, Imran M, Jayaraman PP, Perera C. The role of big data analytics in industrial Internet of Things. Future Generation Computer Systems. 2019; 99:247-59.
[11] Chen CP, Zhang CY. Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data. Information Sciences. 2014; 275:314-47.
[12] Nardelli M, Nastic S, Dustdar S, Villari M, Ranjan R. Osmotic flow: Osmotic computing + IoT workflow. IEEE Cloud Comput. 2017;4(2):68-75.
[13] Mao Y, You C, Zhang J, Huang K, Letaief KB. A Survey on Mobile Edge Computing: The Communication Perspective. IEEE Communications Surveys & Tutorials. 2017;19(4):2322-2358.
[14] Li H, Ota K, Dong M. Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing. IEEE Network. 2018;32(1):96-101. doi:10.1109/MNET.2018.1700202.
[15] Khan WZ, Rehman MH, Zangoti HM, Afzal MK, Armi N, Salah K. Industrial internet of things: Recent advances, enabling technologies and open challenges. Computers & Electrical Engineering. 2020; 81:106522.
[16] Matta JCP, Siddaiah P. Channel Estimation of massive MIMO Using Code Shift Keying Pilot Symbols (CSK-PS). International Journal of Image, Graphics and Signal Processing. 2022;14(3):23-31.
[17] Lakshmanaprabu SK, Shankar K, Ilayaraja M, Nasir AW, Vijayakumar V, Chilamkurti N. Random forest for big data classification in the internet of things using optimal features. International Journal of Machine Learning and Cybernetics. 2019;10:2609-18.
[18] Tiwari CS, Jha VK. Enhancing Security of Medical Image Data in the Cloud Using Machine Learning Technique. International Journal of Image, Graphics and Signal Processing. 2022;14(4):13-31.