International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V73I4P119

Volume 73 Issue 4, 135-139, April 2025

Original Article

Implementing Enterprise-Wide Lakehouse using Microsoft Azure Databricks and Delta Lake

Mehul K Bhuva

Master's in computer science from Georgia Institute of Technology, Atlanta, USA.
Data Platform Architect, Quality Consulting Inc (QCI) West Des Moines, IA-USA.

Corresponding Author : mehul.bhuva@gmail.com

Received: 20 March 2025 Revised: 18 April 2025 Accepted: 23 April 2025 Published: 30 April 2025

Abstract *- This article presents a practical and scalable approach for implementing an enterprise-wide Lakehouse using Azure Databricks and Delta Lake. As data grows in volume, variety, and velocity, organizations need a unified platform that combines the reliability of data warehouses with the scalability of data lakes. The Lakehouse paradigm fulfills this by enabling transactional data lakes with support for both analytical and operational workloads. This paper discusses the architecture, key components, implementation strategies, and real-world considerations for building such systems in Azure. The results showcase improved data governance, reduced duplication, and faster insights. This architecture has broad implications for digital transformation and advanced analytics.*

Keywords *- Azure databricks, Data lakehouse, Data pipeline, Delta lake, Enterprise data architecture.*

1. Introduction

Data is the new oil in the digital age. Enterprises generate vast amounts of structured and unstructured data from diverse sources, including IoT devices, operational databases, and third-party APIs. Traditional data warehouses are not equipped to handle this scale and variety. On the other hand, data lakes offer scalability but lack strong governance and performance for BI workloads. Lakehouse architecture emerges as a hybrid solution that combines the best of both. Azure Databricks, powered by Apache Spark and Delta Lake, is an open-source storage layer that offers a modern platform for realizing enterprise-wide lakehouses. This paper outlines how organizations can adopt this architecture to enable governed, high-performing, and cost-effective analytics at scale.

1.1. Research Gap

Despite advances in big data technologies, organizations continue to struggle with several critical challenges in their data architectures:

1.1.1. Architectural Complexity

Most enterprises maintain separate data lakes and data warehouses, resulting in data duplication, inconsistent governance, and high maintenance costs.

1.1.2. Data Reliability Issues

Traditional data lakes lack ACID transactions, schema enforcement, and data quality controls, leading to what industry practitioners call "data swamps."

1.1.3. Integration Inefficiencies

The separation between analytical and operational data platforms creates integration complexities and delays deriving actionable insights.

1.1.4. Governance Challenges

Distributed data across multiple platforms makes unified security, privacy, and compliance enforcement difficult. These challenges are particularly acute in regulated industries such as financial services, healthcare, and agriculture technology, where data integrity and governance are paramount. While previous research has explored data lake architectures and warehouse modernization separately, a significant gap exists in implementing unified architectures that deliver both reliability and scalability in a production environment.

1.2. Problem Statement

This research addresses how enterprise organizations can implement a unified Lakehouse architecture using Azure Databricks and Delta Lake technologies to overcome the limitations of traditional approaches while maintaining governance, performance, and cost-effectiveness at scale.

2. Literature Review

The evolution of enterprise data architecture has progressed through several generations, from traditional data warehouses to modern cloud-native platforms. This section reviews key contributions to the field.

2.1. Data Warehouse Systems

Inmon (1992) introduced the concept of the Enterprise Data Warehouse (EDW) as a subject-oriented, integrated, time-variant, and non-volatile collection of data supporting management decisions.

Kimball (1996) later proposed the dimensional modeling approach for data warehouses. However, as Russom (2017) noted, traditional EDWs struggle with unstructured data and the velocity requirements of modern data pipelines.

2.2. Big Data Platforms and Data Lakes

Hadoop and related technologies emerged to address the volume challenges, with Shvachko et al. (2010) describing HDFS as a distributed file system for big data. O'Leary (2014) defined data lakes as repositories storing raw data in native formats until needed.

However, Miloslavskaya and Tolstoy (2016) identified significant data quality and access control challenges in first-generation data lakes.

2.3. Emergence of the Lakehouse Paradigm

Armbrust et al. (2020) introduced Delta Lake as a storage layer enabling ACID transactions on cloud storage. Zaharia et al. (2016) described Apache Spark as a unified batch and streaming processing engine. The Lakehouse paradigm, formally articulated by Databricks (2020), combines these technologies to bridge the gap between data lakes and warehouses.

2.4. Cloud Implementation Approaches

Microsoft's documentation (2023) outlines reference architectures for data lakes on Azure, while Databricks documentation (2023) provides technical specifications for implementing Lakehouses.

However, there is limited research on production-level implementations of enterprise-wide Lakehouse architectures, particularly regarding governance models, performance optimization, and organizational adoption strategies.

This research addresses this gap by providing a comprehensive implementation approach validated through a real-world case study in the agricultural technology sector.

3. Architecture & Approach

3.1. Core Components of Lakehouse Architecture

- *Azure Data Lake Storage Gen2:* Scalable storage for raw and curated data that combines the power of a hierarchical namespace with the economics of blob storage.
- *Azure Databricks:* Unified analytics platform that supports batch and streaming data processing, providing an Apache Spark-based foundation for data engineering, data science, and machine learning workloads.
- *Delta Lake:* Open-source storage layer that provides ACID transactions, schema enforcement, time travel capabilities, and optimized data layout on top of ADLS.
- *Unity Catalog:* Centralized governance for data discovery, lineage tracking, and fine-grained access control across all data assets.
- *Azure Synapse and Power BI:* Downstream analysis tools that connect directly to processed data in the Lakehouse.

3.2. Implementation Layers

The architecture follows a mediated data flow pattern through well-defined layers:

- *Ingestion Layer:* Azure Data Factory (ADF) or Databricks Autoloader to ingest structured/unstructured data from various sources with fault-tolerant, incremental processing.
- *Bronze Layer:*Raw data storage with minimal transformation, preserving the original format and content for auditing and reprocessing needs.
- *Silver Layer:* Cleaned and enriched data using Delta format, with standardized schemas, quality validation, and business rules applied.
- *Gold Layer:* Aggregated business-level data models optimized for specific analytical domains and use cases.
- *Consumption Layer:* Power BI dashboards, Synapse queries, and ML workloads that provide business value through actionable insights.

3.3. Novel Aspects of the Proposed Architecture

The architecture proposed in this research offers several innovations compared to traditional approaches:

- *Unified Governance Model:* Leveraging Unity Catalog to provide a single control plane for security, privacy, and compliance across all data assets.
- *Incremental Processing Framework:* Custom implementing change data capture patterns using Databricks Autoloader and Delta Lake time travel capabilities.
- *Quality-by-Design Approach:* Embedding data quality validation at each layer transformation through Delta constraints and expectations framework.
- *Cost-Optimized Compute Strategy:* Dynamic allocation of compute resources based on workload patterns, with auto-scaling and spot instance integration.

4. Materials and Methods

4.1. Implementation Context

A pilot project was implemented for an agricultural company that develops and provides global seed and crop protection products and digital solutions to farmers.

The company had been operating with a traditional data warehouse and a separate Hadoop-based data lake, facing challenges with data inconsistency, governance overhead, and analytics time-to-value.

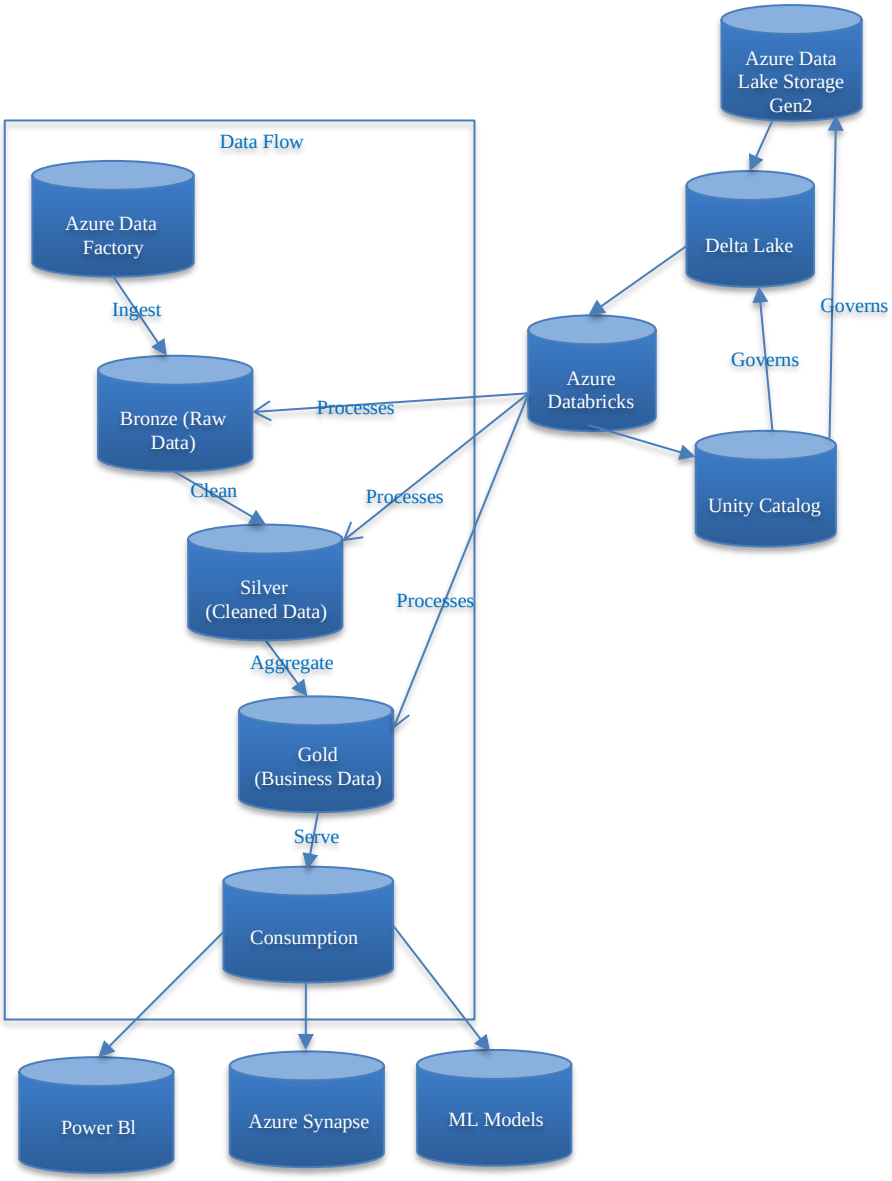

Fig. 1 Lakehouse architecture diagram

4.2. Technologies Employed

- *Azure Databricks Premium Tier:* For enterprise security features and advanced autoscaling
- *Delta Lake 2.0:* For ACID transactions and optimization features
- *Azure Data Factory:* For orchestration of data movement
- *Azure Data Lake Storage Gen2:* For hierarchical namespace storage
- *Unity Catalog (GA version):* For unified governance
- *Power BI Pro:* For visualization and self-service analytics

4.3. Implementation Methodology

The implementation followed an iterative approach:

- *Infrastructure Setup:* Deployment of core components with secure network integration
- *Data Migration:* Progressive migration of critical data domains from legacy systems
- *Pipeline Development:* Creation of medallion architecture data pipelines using Databricks notebooks
- *Governance Implementation:* Configuration of Unity Catalog with row/column-level security
- *Optimization:* Performance tuning and cost optimization
- *Knowledge Transfer:* Training and documentation for organizational adoption

4.4. Evaluation Metrics

The implementation was evaluated against several key performance indicators:

- Query performance compared to legacy systems
- Data freshness (time from source to availability)

- Cost per terabyte of data processed
- Governance compliance coverage
- Developer productivity metrics

5. Results and Discussion

5.1. Benefits Realized

5.1.1. Performance Gains

Query times reduced by 50% compared to legacy database applications, with complex analytical queries showing the most significant improvements. This was achieved through combining Photon engine optimization, Delta Lake indexing features, and columnar storage formats.

5.1.2. Data Quality:

Schema evolution and data validation using Delta constraints ensured consistency across the data lifecycle. The implementation detected and remediated over 2,000 data quality issues that existed in the legacy systems but were previously undetected.

5.1.3. Cost Optimization:

Leveraging Databricks Photon and spot instances reduced compute costs by 30% compared to the previous architecture. Storage costs were also reduced through intelligent data lifecycle management and compression.

5.1.4. Governance:

Unity Catalog provided lineage and Role Based Access Control (RBAC), satisfying compliance needs for both internal policies and regulatory requirements. The unified approach reduced security administration overhead by 40%.

5.1.5. Agility:

Teams independently built data products over standardized layers, reducing time-to-market for new analytics capabilities from weeks to days.

5.2. Quantitative Results

The following table summarizes key performance metrics before and after implementation:

Table 1. Comparison of key data metrics before and after system implementation

Metric	Before Implementation	After Implementation	Improvement
Complex Query Performance	45 mins	22 mins	51%
Standard Query Performance	3.5 mins	1.2 mins	66%
Data freshness	24 hrs	2 hrs	92%
Storage costs	$0.12/GB	$0.05/GB	58%
Compute costs	$850/TB processed	$595/TB processed	30%
Data quality issues	~2000 undetected	<50 detected & remediated	97.5%

5.3. Challenges

Several challenges were encountered during the implementation:

- *Learning Curve:* Teams new to Spark/Delta required training and ramp-up time to become effective with the new paradigm.
- *Data Modeling Discipline:* Maintaining semantic consistency across zones requires strong data modeling practices and governance.
- *Operational Monitoring:* Setting up comprehensive monitoring for jobs, costs, and access patterns needed significant effort.
- *Integration with Legacy Systems:* Some legacy systems require custom connectors and transformation

5.4. Comparison with Existing Approaches

Compared to previous research and industry approaches, this implementation offers several advantages. The key differentiator of our approach is the combination of strong governance through Unity Catalog with the flexibility and performance of Delta Lake, creating a truly unified platform for all data workloads.

Table 2. Comparative analysis of data management approaches

Approach	Traditional Warehouse	Cloud Warehouse	Databricks Lakehouse
Data Quality	High	High	High
Governance	Medium	Medium	High
Performance	Medium	High	High
Cost efficiency	Low	Medium	High
Unified Experience	Low	Medium	High

6. Conclusion

An enterprise-wide Lakehouse architecture implemented using Azure Databricks, and Delta Lake can transform how organizations manage and analyze data. This approach enables scalability, real-time insights, and strong governance by merging the capabilities of data lakes and warehouses. The architecture described offers a blueprint for digital-first organizations seeking to modernize their analytics platforms. The implementation demonstrated significant improvements in performance, data quality, governance, and cost-efficiency compared to traditional architecture. The approach is particularly valuable for organizations with diverse data types, complex analytical requirements, and strict governance needs.

6.1. Future Work

Future research could explore:

- Integration of AI/ML workflows within the Lakehouse architecture.
- Advanced multi-cloud implementation patterns.
- Automated data quality remediation techniques.
- Performance optimization for extremely large datasets (>100 petabytes).
- Enhanced metadata management and semantic layer integration.

Acknowledgements

The authors thank the Cloud Data Engineering team at his own employer for their contributions to the proof of concept and pilot implementation.

References

[1] Matei Zaharia et al., "Apache Spark: A Unified Engine for Big Data Processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56-65, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[2] Michael Armbrust et al., "Delta Lake: High-Performance ACID Table Storage Over Cloud Object Stores," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 3411-3424, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Databricks Documentation, Lakehouse Architecture, 2023. [Online]. Available: https://docs.databricks.com/lakehouse/

[4] Microsoft, Azure Data Lake Storage Gen2 Documentation, 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction

[5] Unity Catalog Documentation, Databricks, 2023. [Online]. Available: https://docs.databricks.com/data-governance/unity-catalog/index.html

[6] Konstantin Shvachko et al., "The Hadoop Distributed File System," *IEEE 26th Symposium on Mass Storage Systems and Technologies*, Incline Village, NV, USA, pp. 1-10, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[7] Daniel E. O'Leary, "Embedding AI and Crowdsourcing in the Big Data Lake," *IEEE Intelligent Systems*, vol. 29, no. 5, pp. 70-73, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[8] Natalia Miloslavskaya, and Alexander Tolstoy, "Big Data, Fast Data and Data Lake Concepts," *Procedia Computer Science*, vol. 88, pp. 300-305, 2016. [CrossRef] [Google Scholar] [Publisher Link]