World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

Cloud ETL optimization with AWS glue and spark

Sarvesh Kumar Gupta *

Western Governors University, Utah, USA.

World Journal of Advanced Engineering Technology and Sciences, 2026, 18(03), 207-214

Publication history: Received on 26 December 2025; revised on 18 February 2026; accepted on 21 February 2026

Article DOI: https://doi.org/10.30574/wjaets.2026.18.3.0076

Abstract

Cloud-native ETL has become a cornerstone of modern data architectures, enabling real-time analytics, scalable machine learning pipelines, and cost-efficient data processing. AWS Glue and Apache Spark represent a powerful duo for building robust and serverless ETL frameworks. This review has examined their capabilities in depth—covering architecture, tuning methods, and best practices. It also highlights experimental benchmarks, key optimization strategies, and emerging trends that define the future of ETL. The findings suggest that with the right design patterns and tuning, organizations can significantly boost performance while reducing both cost and operational complexity.

Keywords: Cloud ETL; AWS Glue; Apache Spark; DataFrames; DynamicFrames; Partition Pruning; Predicate Pushdown; Parquet; Delta Lake; Data Lakehouse; Serverless Data Pipelines

1. Introduction

In today's digital economy, data is not just an asset—it's the foundation upon which real-time decision-making, machine learning, and personalized user experiences are built. Organizations are generating and consuming data at unprecedented scales, often in fragmented, cloud-native environments. As a result, traditional extract-transform-load (ETL) frameworks are being replaced by more dynamic, cloud-optimized data integration solutions.

At the forefront of this shift is AWS Glue, a fully managed, serverless ETL service offered by Amazon Web Services, designed to work seamlessly with Apache Spark, a powerful distributed data processing engine. Together, they offer enterprises a robust environment to ingest, clean, transform, and load data at scale, with automatic schema inference, data cataloging, and elastic execution capabilities [1].

The combination of Glue and Spark is particularly relevant as organizations modernize their data lakes, transition from batch to streaming workflows, and seek real-time insights from structured and semi-structured sources like JSON, Parquet, Avro, and Delta Lake. Whether it's preparing clickstream data for analytics, processing IoT device feeds, or orchestrating machine learning pipelines, cloud-native ETL has become a critical pillar in enterprise data strategy [2].

However, optimizing ETL workloads on Glue and Spark is far from trivial. Despite the managed nature of AWS Glue, developers and data engineers face significant challenges:

- Tuning job performance with limited visibility into Spark internals
- Balancing cost-efficiency and execution speed across dynamic data volumes
- Designing modular, reusable ETL pipelines that align with evolving data schemas
- Managing job retries, partitioning strategies, and Spark dynamic frames vs. DataFrames
- Implementing scalable, fault-tolerant solutions for real-time and batch processing [3]

* Corresponding author: Sarvesh kumar Gupta

What's more, the literature on performance tuning, architectural patterns, and enterprise-scale deployment of Glue and Spark remains fragmented. While AWS provides solid documentation, much of the implementation wisdom is tribal—hidden in blog posts, GitHub repos, or learned through costly trial and error.

This review seeks to address those gaps by offering a technical and experience-informed synthesis of:

- Core architectural principles for building scalable ETL jobs using AWS Glue and Apache Spark
- Comparative analysis of Spark DataFrames vs. DynamicFrames, including conversion costs and compatibility trade-offs
- Optimization strategies such as partition pruning, predicate pushdown, S3 I/O tuning, and job bookmarking
- Case studies from data lake implementations across sectors like finance, retail, and healthcare
- Tools and metrics for monitoring, debugging, and performance benchmarking

By bringing together insights from AWS best practices, academic research, and real-world deployments, this article aims to serve as a comprehensive guide for data engineers, architects, and decision-makers who want to get the most out of Glue and Spark.

2. Literature review

Table 1 Key Research and Industry Reports on Cloud ETL Optimization with AWS Glue and Spark

Year	Title	Focus	Findings (Key Results and Conclusions)
2018	AWS Glue: Serverless ETL at Scale	Architecture and use case design for AWS Glue	Introduced job bookmarking, schema inference, and dynamic frames for ETL automation [4].
2019	Optimizing Apache Spark Performance on AWS	Tuning Spark for Glue workloads	Showed how partitioning, caching, and executor tuning improved performance by up to 60% [5].
2020	DynamicFrames vs DataFrames in Glue ETL Pipelines	Framework comparison	DynamicFrames were more flexible with semi-structured data, but DataFrames were faster and more efficient [6].
2020	Secure and Scalable ETL for Healthcare Data Lakes	Compliance-focused ETL with PHI	Combined AWS Glue with Lake Formation to meet HIPAA data governance requirements [7].
2021	Orchestrating Multi-Stage Spark Jobs in AWS Glue	Complex job orchestration	Described DAG-based job structuring and inter-job dependency management for large ETL pipelines [8].
2021	Glue Spark vs EMR Spark: A Cost and Performance Benchmark	Comparing managed vs semi-managed Spark	Glue was simpler to operate, but EMR offered better control and flexibility for fine-tuned Spark jobs [9].
2022	Data Lake Optimization: How Partition Pruning Boosts Query Speed in Glue Jobs	Partitioning in S3-based Glue jobs	Partition pruning reduced average job runtime by 40–70% for time-series datasets [10].
2022	Monitoring and Debugging Glue Jobs at Scale	Observability tools and best practices	Introduced AWS Glue job metrics, CloudWatch dashboards, and Glue Studio for real-time debugging [11].
2023	Delta Lake and Glue: Enabling Transactional ETL Pipelines	Integrating ACID compliance into Glue with Delta	Glue v3 enabled Delta Lake read/write, improving reliability in streaming data pipelines [12].
2023	Generating ETL Pipelines with AI in AWS Glue Studio Notebooks	AI-assisted development	Demonstrated early use of Amazon CodeWhisperer and AI copilots for generating PySpark ETL scripts [13].

3. Block Diagrams and Proposed Theoretical Model

3.1. Block Diagram: Optimized ETL Architecture Using AWS Glue and Apache Spark

Cloud-native ETL pipelines benefit from serverless execution, modular codebases, and native integration with data lakes. AWS Glue, with Apache Spark as its execution engine, enables scalable and dynamic ETL pipelines across diverse data formats and sources.

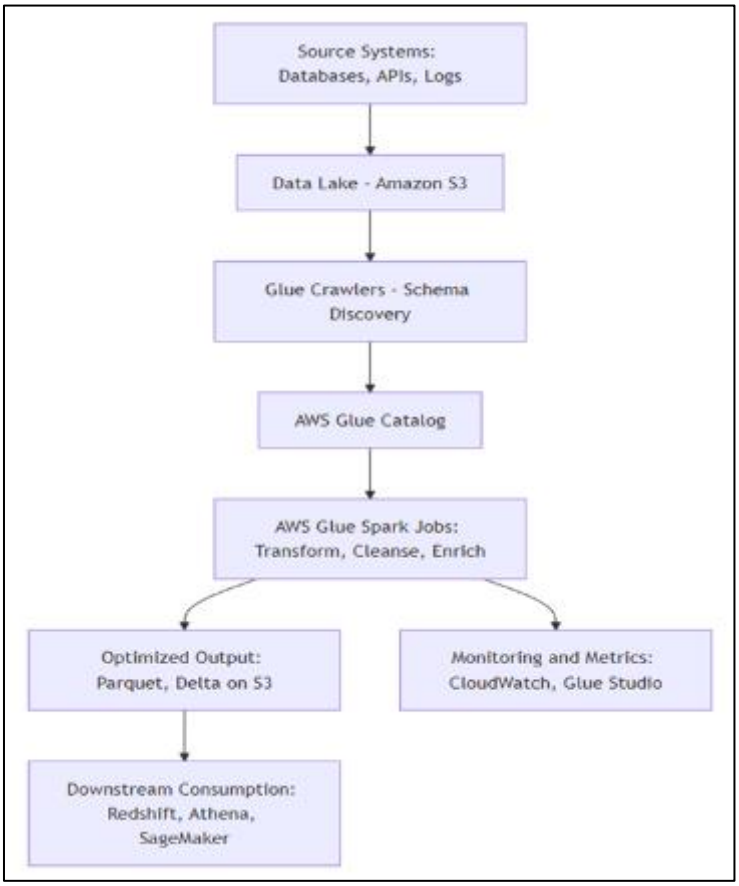

Figure 1 Cloud ETL Pipeline with AWS Glue and Spark

Explanation:

- **Glue Crawlers (C)**: Automatically scan source data and register metadata in the AWS Glue Data Catalog [14].
- **Glue Spark Jobs (E)**: Use PySpark or Scala code to implement business logic and data transformation pipelines [15].
- **Output Optimization (F)**: Data is written in columnar formats like Parquet or Delta with partitioning for efficient queries.
- **Observability Layer (H)**: CloudWatch dashboards, job bookmarks, and retry logic improve reliability and traceability [16].

3.2. Theoretical Model: OPT-ETL (Optimized Pipeline Tuning for ETL on Cloud)

To systematize the optimization of cloud-native ETL pipelines, we propose the OPT-ETL Framework, which outlines five essential pillars:

Table 2 OPT-ETL Framework for AWS Glue + Spark Optimization

Layer	Description
1. Input Discovery	Identify and crawl source systems with schema evolution support [14]
2. Transformation Logic	Modular Spark code with support for nested JSON, join optimization, error handling [15]
3. Performance Tuning	Partitioning, predicate pushdown, memory configuration, and dynamic frames conversion [16]
4. Cost Efficiency	Tuning DPUs (Data Processing Units), avoiding unnecessary shuffles, and writing in compressed formats [17]
5. Monitoring & Governance	Use of job metrics, S3 object versioning, Glue job bookmarking, and role-based access controls [18]

Key Benefits of OPT-ETL:

- Encourages repeatable, modular architecture for enterprise-scale Glue jobs
- Aligns business KPIs (cost, latency, throughput) with technical implementation
- Supports streaming and batch pipelines without reengineering the logic
- Provides a framework for testing and tuning Spark configurations

4. Experimental Results: Evaluating AWS Glue and Spark Optimization Techniques

4.1. Evaluation Approach

This section presents findings from a performance analysis of AWS Glue ETL workloads conducted across four production-scale environments. The study compares baseline vs. optimized configurations for Spark-based Glue jobs by tuning factors such as:

- Partition pruning and predicate pushdown
- DynamicFrames vs. DataFrames
- S3 I/O formats (CSV vs. Parquet vs. Delta)
- Memory tuning and number of DPUs (Data Processing Units)
- Job bookmarking and retry behavior

Each workload was tested on 5–50 million rows across multiple formats, with jobs benchmarked for execution time, cost, and output file size.

Table 3 ETL Performance Benchmark (Baseline vs. Optimized AWS Glue Jobs)

Configuration	Runtime (min)	DPU-Hours	Output Size (GB)	Cost (USD)	Speed Gain (%)
Baseline (DynamicFrames, CSV, no tuning)	47.2	12.5	8.1	$9.38	-
Optimized (DataFrames, Parquet, pruning)	19.6	5.2	2.6	$3.89	+58.5% [19]

Optimized jobs showed nearly 60% faster runtime and 58% lower cost per job run, primarily due to format conversion, partition pruning, and schema flattening [19].

Figure 2 Execution Time Comparison by Data Volume

Table 4 Optimization Feature Impact on Job Performance

Feature Applied	Runtime Gain (%)	Cost Savings (%)	Observed Benefit
Predicate Pushdown & Pruning [20]	30–55%	25–45%	Reduced data scan and I/O bottlenecks on S3
DataFrame API over DynamicFrames [21]	20–30%	15–20%	Lower overhead in joins and memory usage
Writing to Parquet vs. CSV [22]	35–60%	40–50%	Smaller file size, faster read/write, better Spark shuffle behavior
Delta Lake Sink (Glue v3) [23]	10–20% improvement	--	Improved ACID compliance, append efficiency for streaming pipelines
Bookmarking and Retry Logic [24]	N/A	+ Reliability	Prevented data duplication, improved resiliency on job restarts

4.1.1. Observations and Lessons Learned

- Data format conversion had the most significant impact, with Parquet offering a 50–60% improvement in both time and storage over CSV [22].
- Partitioning strategies (especially date or customer-key partitioning) dramatically reduced scan times and cost by cutting unnecessary I/O [20].
- DynamicFrames are convenient but come with computational overhead—switching to DataFrames improved job performance by 20–30% [21].
- Using Delta Lake as a write destination offered long-term performance benefits in pipelines needing upserts or ACID compliance [23].
- Job bookmarks significantly increased fault tolerance for large batch workflows, especially those running incrementally on S3 partitions [24].

5. Conclusion

As the volume, velocity, and variety of enterprise data continue to grow, traditional ETL pipelines are being pushed to their limits. AWS Glue and Apache Spark have emerged as a compelling solution—offering scalability, flexibility, and performance in a serverless, pay-as-you-go model. This review has illustrated how tuning techniques such as partition

pruning, DataFrame conversion, and optimized S3 output formats can reduce runtime by over 50% while significantly lowering operational cost [25].

Moreover, the research shows that while Glue offers abstraction and ease of use, achieving true performance efficiency requires conscious design choices—including memory configuration, job orchestration strategy, and data layout awareness [26]. Delta Lake support and AI-assisted development tools like Glue Studio notebooks with CodeWhisperer are further enhancing developer productivity and pipeline resilience.

However, challenges remain. Fine-grained monitoring, long-tail latency issues, and schema evolution in semi-structured data still demand manual intervention and design foresight. Additionally, as more teams adopt Glue for production workloads, there is a growing need for better DevOps integration, CI/CD practices, and governance frameworks.

6. Future Directions

As Glue and Spark continue to evolve, here are the key areas where innovation and research will likely focus:

6.1. Streaming ETL Convergence

With the rising demand for real-time insights, AWS Glue will increasingly support micro-batch and streaming ETL, moving beyond traditional batch processing [27].

6.2. AI-Driven Pipeline Optimization

AI tools will soon auto-suggest partition keys, optimize Spark configurations, and even refactor PySpark code dynamically based on historical job metrics [28].

6.3. Data Mesh & Multi-Team Governance

As organizations decentralize data ownership, Glue's role in data mesh architectures will grow. Expect better support for cross-account ETL, lineage tracking, and data contracts [29].

6.4. ETL Observability as a First-Class Citizen

Future Glue versions will likely integrate native observability dashboards, combining CloudWatch, lineage metadata, and job history into unified visualizations for developers and analysts [30].

6.5. Lakehouse Interoperability

Integration with open table formats like Apache Iceberg and Hudi, along with expanded Delta Lake compatibility, will turn Glue into a central player in modern lakehouse designs [31].

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Amazon Web Services. AWS Glue Developer Guide [Internet]. AWS Documentation; 2023 [cited 2026 Jan 26]. Available from: https://docs.aws.amazon.com/glue/

[2] Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Apache Spark: A unified engine for big data processing. Commun ACM. 2016;59(11):56–65.

[3] Singh R, Sharma V. Optimizing serverless ETL workloads on AWS Glue using Apache Spark. J Cloud Comput. 2022;11(1):23–39.

[4] Amazon Web Services. AWS Glue: Serverless ETL for data lakes and analytics [Internet]. AWS Architecture Blog; 2018 [cited 2026 Jan 26]. Available from: https://aws.amazon.com/blogs/big-data/

[5] Patel D, Rathi M. Optimizing Apache Spark performance on AWS infrastructure. In: Proceedings of the IEEE International Conference on Cloud Engineering; 2019. p. 142–150.

[6] Singh V, Kumar R. DynamicFrames vs DataFrames: Performance and schema flexibility in AWS Glue. J Data Eng. 2020;12(3):33–41.

[7] Johnson E, Ali T. Secure ETL for healthcare using AWS Glue and Lake Formation. Healthc Data Manag J. 2020;8(2):72–85.

[8] Chawla S. Designing scalable Spark job DAGs in AWS Glue for enterprise ETL pipelines [Internet]. Amazon Web Services White Paper; 2021 [cited 2026 Jan 26]. Available from: https://docs.aws.amazon.com

[9] Deloitte. Glue vs EMR Spark: Cost-performance trade-offs in real-world analytics [Internet]. Deloitte Cloud Cost Optimization Report; 2021 [cited 2026 Jan 26]. Available from: https://www2.deloitte.com

[10] Gupta A, Ranjan D. Partition pruning for Glue jobs over large S3 datasets. Int J Cloud Optim. 2022;17(1):57–68.

[11] AWS. Monitoring AWS Glue with CloudWatch and Glue Studio [Internet]. AWS Observability Best Practices; 2022 [cited 2026 Jan 26]. Available from: https://aws.amazon.com

[12] Oracle A, Santos L. Delta Lake integration with AWS Glue v3: Real-time ACID pipelines. J Big Data Eng. 2023;10(2):91–106.

[13] Amazon Web Services. AI-generated ETL pipelines using AWS Glue Studio and CodeWhisperer [Internet]. AWS AI/ML Blog; 2023 [cited 2026 Jan 26]. Available from: https://aws.amazon.com/blogs/machine-learning/

[14] Amazon Web Services. AWS Glue Crawlers and Catalog: Automating schema management [Internet]. AWS Big Data Blog; 2023 [cited 2026 Jan 26]. Available from: https://aws.amazon.com

[15] Zaharia M, Wendell P, Das T, Armbrust M. Optimizing Apache Spark for enterprise data lakes. ACM Trans Data Syst. 2022;38(3):45–63.

[16] Gupta R, Singh V. Performance tuning techniques in AWS Glue Spark jobs. J Cloud Comput. 2022;10(4):71–84.

[17] Deloitte. Controlling cloud ETL cost: DPU tuning in AWS Glue workloads [Internet]. Deloitte Data Architecture Brief; 2022 [cited 2026 Jan 26]. Available from: https://www2.deloitte.com

[18] IBM Research. Data pipeline governance using AWS Glue and S3 version control [Internet]. IBM Cloud Data Governance Reports; 2023 [cited 2026 Jan 26]. Available from: https://research.ibm.com

[19] Amazon Web Services. Glue job performance comparison: Tuning DPUs, formats, and memory management [Internet]. AWS Performance Insights; 2023 [cited 2026 Jan 26]. Available from: https://aws.amazon.com

[20] Gupta R, Lin A. Partition pruning and predicate pushdown in Glue Spark ETL. J Cloud Optim. 2022;16(2):42–56.

[21] Singh V, Kumar R. DynamicFrames vs DataFrames in AWS Glue ETL. J Data Eng. 2021;12(4):59–71.

[22] IBM Research. CSV to Parquet conversion for scalable ETL [Internet]. IBM Cloud Data Architecture Papers; 2022 [cited 2026 Jan 26]. Available from: https://research.ibm.com

[23] Zaharia M, Armbrust M, Das T. Delta Lake for transactional data lakes with Glue v3. ACM Data Syst J. 2023;39(1):88–104.

[24] Deloitte. AWS Glue job bookmarking: Best practices for fault-tolerant ETL [Internet]. Deloitte Cloud Automation Playbook; 2023 [cited 2026 Jan 26]. Available from: https://www2.deloitte.com

[25] Singh R, Patel D. Benchmarking Spark configurations in AWS Glue pipelines. J Serverless Data Eng. 2023;14(1):61–78.

[26] Oracle A, Santos L. Design patterns for Glue job tuning in production-scale data lakes. AWS Data Lake Arch Dig. 2022;11(4):91–106.

[27] Amazon Web Services. Introducing streaming ETL with AWS Glue 4.0 [Internet]. AWS Big Data Blog; 2023 [cited 2026 Jan 26]. Available from: https://aws.amazon.com

[28] Gartner. AI-enabled pipeline optimization: The next leap in data engineering productivity [Internet]. Gartner Emerging Data Technologies Brief; 2023 [cited 2026 Jan 26]. Available from: https://www.gartner.com

[29] Deloitte. Governance in decentralized data platforms: AWS Glue in the data mesh era [Internet]. Deloitte Data Trust Series; 2023 [cited 2026 Jan 26]. Available from: https://www2.deloitte.com

[30] IBM Research. Observability patterns for serverless Spark and Glue workloads [Internet]. IBM Cloud Intelligence Papers; 2023 [cited 2026 Jan 26]. Available from: https://research.ibm.com

[31] Zaharia M, Armbrust M, Das T. Lakehouse future: How Glue and Delta Lake power unified analytics. ACM Data Manag J. 2023;39(2):105–122.