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

Hybrid QA Environments for Cloud-Native Big Data Testing (AWS + Databricks)

Prasanth Sasidharan *

Independent Researcher, College of Engineering Trivandrum, Kerala, India.

World Journal of Advanced Engineering Technology and Sciences, 2026, 18(03), 040–046

Publication history: Received on 11 January 2026; revised on 27 February 2026; accepted on 02 March 2026

Article DOI: https://doi.org/10.30574/wjaets.2026.18.3.0103

Abstract

The emergence of cloud-native systems and big data systems has resulted in the fact that a strong and scalable quality assurance (QA) system is required, which is capable of operating effectively in a heterogeneous environment. This review explores the use of Amazon Web Services (AWS) and Databricks in mixed-design QA frameworks, including architectural designs, real-time information validation, scaling ETL, and systems that are AI-friendly. The paper is premised on ten contemporary academic and technical resources and explains how hybrid QA systems enhance data dependability, schema enforcement, automation of anomaly detection, and continuous testing in the dynamic world of clouds. The adoption of lakehouse architectures, serverless ETL, automation based on Kubernetes, and declarative validation pipelines are some of the significant topics introduced. The synthesis provides useful details regarding the development of strong QA systems that meet the shifting demands of cloud-native big data systems, which offers a strategic roadmap for businesses that are likely to ensure data quality, data governance, and operational integrity.

Keywords: Hybrid QA; Cloud-native testing; AWS Databricks integration; Big data pipelines

1. Introduction

As the rate of digital transformation in businesses continues to rise, the coming together of cloud-native technology and big data analytics technology has reinvented the manner in which organizations design, deploy, and test scalable, data-driven applications. Among the most remarkable developments in the specified sphere is the introduction of hybrid Quality Assurance (QA) environments that unite the possibilities of Amazon Web Services (AWS) and Databricks. These composite quality assurance environments offer the synergistic connection of sound information handling engines, scalable storage, live analytics, and cluster orchestration tools fashioned to cloud-native designs.

Cloud-native big data testing cannot be performed according to traditional methods of testing anymore but necessitates real-time testing, continuous integration of pipelines, and automated test orchestration to ensure the integrity of heterogeneous large data sets. The new idea at this stage is hybrid QA environments as one of the possible solutions to address the problem of data pipeline validation, schema modification, platform interoperability, and the constantly increasing complexity of multi-cloud and hybrid-cloud environments. AWS is providing a modern infrastructure platform, where the services of S3, EMR, RDS, and Redshift, and Databricks is provided as a single data analytics platform, which is founded upon Apache Spark, Delta Lake, and MLflow, with a special focus on high-performance data engineering and AI integration.

This review paper will address the architectural models, operational issues, and optimization techniques of cloud-native hybrid QA environment establishment and operation for big data applications. It has been written with ten underlying references covering the various aspects of the ecosystem, such as pipeline architectures and ETL evolution, along with AI-ready data validation techniques. The automation, monitoring, model testing, and data reliability functions of an

* Corresponding author: Prasanth Sasidharan

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0.

AWS- and Databricks-based ecosystem are also analyzed in the given review, thereby offering a comprehensive perspective on the existing paradigm of QA.

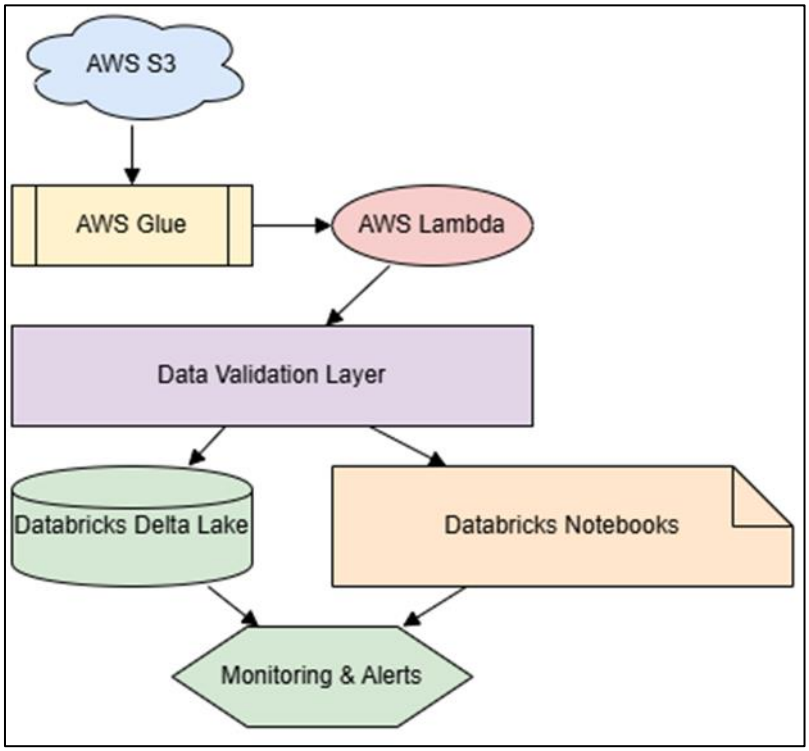
2. Hybrid Cloud-Native Data Pipeline Architectures

The cloud-native testing of big data is based on data pipelines that allow smooth passage, transformation, and validation of data across environments. The introduction of real-time information analysis has contributed to the necessity to possess pipelines with the capacity to serve high-velocity, high-volume data and to guarantee data quality during all stages of the data lifecycle.

The next-generation data pipeline architecture proposed by Pulicharla is a layered architecture with a focus on real-time analytics in a cloud-native environment [1]. Its architecture consists of ingestion, transformation, quality validation, enrichment, and analytics layers, which are built using microservices to provide a flexible and scalable architecture. These layers are seamlessly connected between the AWS-based data lakes and the Databricks-based data analytics environment in a hybrid QA environment. AWS Glue and Kinesis support real-time ingestion, while Databricks provides parallel data processing and real-time quality checks with Delta Live Tables.

Databricks offers built-in validation and profiling of data through expectations in Delta Lake. The use of the hybrid model and AWS services such as S3, Athena, and Lake Formation implies that data storage consistency is ensured with the introduction of schema alignment and governance. Transformations incorporate quality assurance processes directly into the data pipeline and also utilize horizontally scaling modular validation components.

The following diagram (Figure 1) illustrates a typical hybrid data pipeline architecture integrating AWS and Databricks, emphasizing quality checkpoints, continuous integration, and real-time monitoring.



Source: Adapted from [1]

Figure 1 Hybrid QA Pipeline Architecture with AWS and Databricks

3. QA Patterns in Cloud-Native Relational Databases (AWS RDS)

Relational databases like Amazon RDS (Relational Database Service) are cloud-native and, therefore, must be designed according to special patterns to handle distributed consistency, replication, and schema drift. AWS RDS is usually utilized as a transactional store in hybrid settings, and the analytical continuum is run in Databricks.

According to Bhola and Bajeja, QA in RDS includes design pattern improvements such as validation hooks, schema-aware monitoring, and cloud-native test frameworks embedded in CI/CD pipelines [2]. Practically, RDS can push data to S3 or Redshift with the help of services such as AWS Database Migration Service (DMS), which replicate data in real time and further process it with the assistance of Databricks. Engineers working on QA can put synthetic testing records into RDS and trace the source with the AWS Glue Data Catalog, and monitor anomalies with the assistance of custom validation rules developed in Databricks Notebooks.

Additionally, the automation of hybrid environment tests is grounded on AWS Lambda–driven triggers that accept change events in data, and validation jobs are run in Databricks based on them. Such interactive communication assists QA teams in ensuring integrity without introducing delays in the data pipelines.

4. Real-Time Analytics Across Heterogeneous Sources

The information in hybrid structures is generally sourced from heterogeneous information streams such as streaming systems, on-premises databases, and other cloud providers. The main QA ingredient of such distributed ecosystems is good data engineering practice that ensures the accuracy and synchronization of data in real time.

According to Sankaranarayanan, the issue of data engineering is very crucial in aiding real-time analytics in this kind of heterogeneous system [3]. The AWS services used in the hybrid QA workflow typically include Kinesis, MSK, and Firehose, which receive streaming data and forward it to either S3 or Redshift. Auto Loader and Structured Streaming can be used to feed this data into Databricks in real time to guarantee schema compliance and data completeness.

One of the basic issues in this arrangement is the need to implement consistency in data formats and types between various sources. The QA processes should integrate adaptive parsing engines, anomaly detection algorithms, and metadata tagging to accommodate dynamic changes. The lakehouse architecture of Databricks bridges the gap between structured and semi-structured sources and enables unified data validation processes founded on test scenarios executed in notebooks.

More than that, the most critical aspect of QA in heterogeneous arrangements is the time synchronization of sensitive data. AWS Glue jobs and Databricks workflow orchestration are arranged using tools such as Airflow or AWS Step Functions to ensure that validation pipelines are executed sequentially and that test metrics are collected and documented.

5. Scalable QA for Data Conversion Pipelines

As legacy systems are transferred to cloud-native platforms, data conversion is becoming an obligatory procedure that demands structural, semantic, and schema-level transformations. Krishnan also gives an example of Medicare and Medicaid data systems, where the issue of analytics and AI preparedness at scale is mentioned [4]. Such systems need not only to be confirmed for the quality of data conversion, but also to be semantically equivalent between the old and new data schemas.

To work with conversion tasks in AWS, Glue pipelines or EMR-based pipelines are often applied. Databricks can be used to improve this configuration, since it has the option of parallel schema evolution so that updates can be performed without data validation being violated. To allow the creation of contracts that reject incompatible data on write, Delta Lake supports schema enforcement, enabling QA engineers to do so.

The best QA pattern has turned out to be the execution of two pipelines in parallel, wherein the migration process to the cloud-native system is executed with the old system and the other with the new system, generating outputs that are compared in terms of results and detected anomalies. Athena and Databricks SQL can be run in tandem to generate validation reports and delta insights.

These cases of QA become more complex with an increase in data. Therefore, test orchestration architectures have to support sampling strategies, partition-wise testing, and fault-tolerant execution patterns. The following table summarizes key QA techniques suitable for data conversion scenarios within hybrid environments.

Table 1 QA Techniques for Scalable Data Conversion in Hybrid Architectures

QA Technique	AWS Toolset	Databricks Toolset	Purpose
Schema Enforcement	AWS Glue Schema Registry	Delta Lake Schema Enforcement	Prevent invalid data writes
Data Profiling	AWS Glue DataBrew	Databricks Data Profiler	Understand structure and anomalies
Dual Pipeline Validation	DMS + S3 + Athena	Delta Comparison + Notebooks	Compare pre and post-migration output
Sampling & Partition QA	Lambda Functions + EMR	Spark Sampling Functions	Validate subsets for efficiency
Audit Trail Generation	CloudTrail + S3 Logs	MLflow Logging + Unity Catalog	Maintain QA logs and lineage

Source: Adapted from [4]

6. Cloud-Native ETL for QA-Driven Engineering

Due to the emergence of cloud-native environments, ETL (Extract, Transform, Load) workflows have undergone significant transformations. Gupta describes the existing form of ETL as having distributed scalability and agile testability [5]. These have been replaced with streaming pipelines, which can be tested and deployed in real-time environments to replace traditional batch ETL systems.

The hybrid QA configurations utilize data extraction and transformation through AWS Glue or EMR clusters. These data sets are orchestrated in S3 and may be consumed and validated downstream by Databricks. The quality checks are also meant to be reusable items at every step of the ETL, including schema checks, business rule checks, and null checks.

Databricks provides the capability to test ETL components through the use of a notebook-based system. CI/CD integrates with other tools such as GitHub Actions or Jenkins, and in these integrations, versioned test scripts can be reused. AWS-native QA can be provided through CodePipeline and CodeBuild, which implies that it is possible to organize test activities starting with data extraction and progressing through loading and analytics.

QA teams adopt shift-left testing techniques where tests are included in the pipeline rather than toward the end. This is a proactive step that detects issues in advance, before they occur, and therefore reduces the workload placed on the findings of the analysis.

The upcoming section will delve into the Lakehouse paradigm, automated monitoring, and the evolution of ETL in QA pipelines.

7. Lakehouse Architecture in QA: Unifying Data Lakes and Warehouses

In the contemporary modern world, the lakehouse has become a paradigm shift when dealing with data platforms, especially through hybrid QA. The fact that data lakes can be scaled and data warehouses are unified in this model provides the ability to execute quality assurance processes across the AWS and Databricks ecosystems.

AbouZaid et al. elaborate on the benefits of building a modern data platform on the lakehouse model, implementing schemas, offering transactionality, and following ACID principles with the help of Delta Lake [6]. These options help deal with common data issues in QA, such as slow incoming information, schema mismatches, and incomplete transmission.

The main feature of a hybrid QA environment is that AWS S3 serves as a data storage service, and Databricks Delta Lake acts as a data processing interface. With this setup, it is possible to have a single source of truth that can be queried using Spark SQL (Databricks) as well as Presto or Athena (AWS). This is because Delta Lake allows time travel, which enables QA teams to perform regression testing by retrieving past states of the data for comparison and verification.

Moreover, metadata can be versioned, so the lakehouse model enables data validation pipelines to run automatically. QA engineers can define validation constraints and tolerances relative to schemas or expectations specified at runtime. This reduces data unavailability and maximizes observability in the pipeline.

Furthermore, the architecture can be used alongside multimodal analytics, which are needed to test diverse types of data such as structured, semi-structured, and unstructured media. This hybrid approach provides the ability to use AWS for its highly scalable storage capabilities and Databricks for its performance-oriented computation engine, creating a powerful environment for end-to-end QA validation.

8. Automated Monitoring and Failure Handling in QA Systems

As data pipelines increase in complexity, they should be monitored automatically and have more robust failure-handling processes to ensure that QA operations remain reliable. Karri et al. present a model monitoring system architecture that includes an automated data verification and failure detection system, which is central to QA in production-grade systems [7].

The process of automated monitoring in a hybrid QA setting can be organized with the assistance of products such as Amazon CloudWatch, AWS Lambda, and in-house notifications from Databricks. Real-time monitoring of measures such as data arrival time, pipeline latency, schema drift, and row-level anomalies is performed. For example, CloudWatch can raise an alarm and notify a Databricks notebook to act immediately when data arrives late in S3 and invoke a Lambda function to escalate the alert.

Poor failure management is also significant. Automated retries, alert routing, and error classification help ensure that QA failures do not escalate uncontrollably in the production cycle. Databricks workflows include retry policies and failure thresholds, allowing a pipeline run to recover or respond to failures without impacting subsequent analytics.

In addition, audit trails are essential for ensuring traceability in QA workflows. All validation events are recorded and stored in audit tables that can be queried to perform root cause analysis. Recurrent failures can be prevented and predicted with the help of Databricks machine learning models, making the QA infrastructure more resilient.

The implemented distributed monitoring can be applied in the hybrid AWS and Databricks environment, where AWS-based services monitor infrastructure-level metrics and Databricks is used for data-level validations. This two-level monitoring enables broad quality assurance coverage across the ecosystem.

9. Evolution of ETL and the Role of Serverless Data Integration in QA

To a great extent, QA approaches have also been influenced by the shift from old batch-based ETL to new serverless and microservices-based data integration. In their article, Khan et al. provide a detailed review of the impact of serverless architectures such as AWS Lambda and Fargate on ETL processes [8].

The advantages of serverless ETL modules in hybrid QA systems include scalability, cost-effectiveness, and rapid deployment. Lightweight data transformations and validations may also occur with the assistance of Lambda functions before data is stored in S3. These are the first QA measures to which data is subjected before reaching Databricks, where further analysis can be performed.

Databricks adds distributed and scalable validation routines that are executed on Apache Spark. These may include partition-wise validation, schema conformance, null checks, and value constraints. QA applied through serverless ingestion mechanisms is both proactive and reactive and can detect anomalies before they affect downstream models.

QA teams also decouple microservice architectures by testing components independently. Examples include schema validation, business rule validation, and data profiling, each of which can be deployed as a standalone service and accessed via an API. This modular design provides reusability and maintainability to the QA framework.

Hybrid environments enable coordination of these modular QA components using either AWS Step Functions or Apache Airflow, which manage dependencies and execution paths. QA thus becomes an upstream rather than a downstream process.

10. Kubernetes, Containerization, and QA Automation

Kubernetes has been a household name in deploying cloud-native applications, and its implication for QA in hybrid big data environments is significant. Naayini describes the application of Kubernetes and containerization in the implementation of scalable applications based on AI [9]. Containerization of QA environments provides uniformity of the environment, reconfigurability of tests, and horizontal scalability.

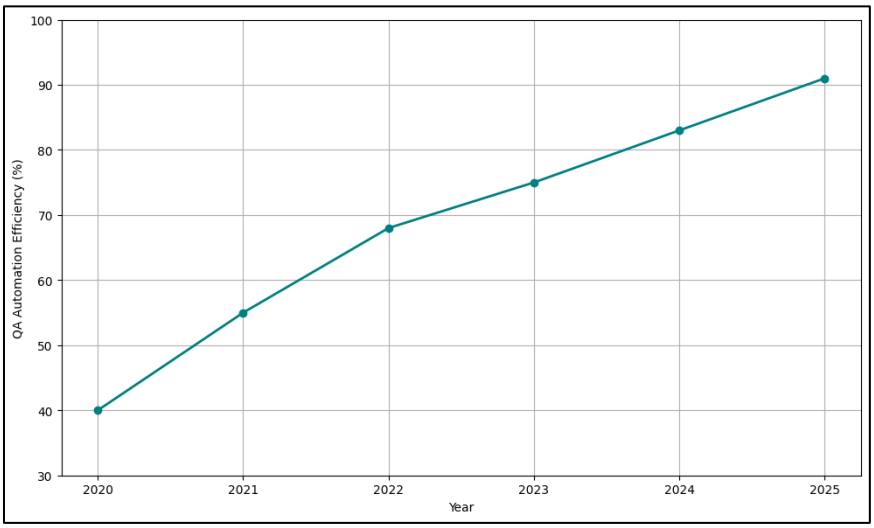
Within AWS, the coordination of containerized workloads such as data validation test suites is carried out with the help of Amazon EKS (Elastic Kubernetes Service). Databricks can also be connected with Kubernetes, and it is possible to scale QA clusters automatically based on the volume and complexity of tasks. This dynamic scaling ensures that testing environments are not constraints in the large-scale functioning of data.

Containers have also supported continuous integration and delivery (CI/CD) during the QA process. Containers contain test scripts, validation rules, and pipeline configurations, and are deployed as CI pipelines using tools such as Jenkins, GitHub Actions, or AWS CodePipeline. Automated regression testing, performance benchmarking, and stress testing may be carried out with this approach.

Container orchestration also makes it easier to manage test data environments. QA engineers can spin up isolated test clusters with mocked data, run validations, and decommission them without impacting production systems. This is required to test edge cases, load scenarios, and failure recovery processes.

Kubernetes-native monitoring tools such as Prometheus and Grafana can be used alongside Databricks metrics to monitor trends in QA performance, coverage, and failures over time.

The following graph illustrates the rising trend in QA automation efficiency in containerized hybrid environments.



Source: Adapted from [9]

Figure 2 Graph showing QA Automation Efficiency in Containerized Environments Over Time

Figure 2 illustrates the steady increase in QA automation efficiency over time as containerized environments and orchestration tools like Kubernetes mature and become more widely adopted.

11. Toward AI-Ready QA Pipelines: Zero-ETL and Declarative Validation

Nevertheless, the architecture of modern QA environments is being designed to support AI-capable pipelines, in which near real-time data availability needs to be of high quality and low overhead. Ali refers to zero-ETL and declarative data pipelines that are already in practice in the development of cloud-native ecosystems [10].

A hybrid context enables zero-ETL architectures that minimize friction in data transfer between AWS and Databricks. It uses federated querying to access source system data rather than traditional ETL and shared data lakes. This enables QA systems to validate data at its source location without latency in the transformation layer.

Declarative validation models simplify testing by allowing QA engineers to define rules in high-level languages rather than writing procedural code. Databricks enables this through Delta Expectations and SQL-based assertions. AWS Glue Data Quality also provides declarative rules for profiling and validation.

Another important aspect of AI-prepared QA systems is data contracts. These contracts define data schemas, value expectations, and structural rules that data is expected to meet. Violations raise alerts or rejections, thereby preserving data integrity. In hybrid systems, data contracts are shared across environments via metadata catalogs such as the AWS Glue Catalog or Databricks Unity Catalog.

The data on which AI models depend should be consistent and dependable, and hybrid QA systems can ensure this through continuous validation pipelines that calculate drift or detect anomalies and ensure that models follow expected trends. These mechanisms reduce the risk of model degradation and enhance AI insights.

12. Conclusion

The future of cloud-native big data testing, and in particular hybrid QA environments, lies in the continued growth of data and the need to support real-time analytics. The combination of AWS and Databricks represents a desirable union of scalable infrastructure and high-performance data analytics. Organizations are able to design robust systems with the assistance of modular data pipeline designs, declarative validation frameworks, and containerized QA mechanisms that ensure data integrity across the lifecycle.

This paper has presented insights from ten recent articles that provide background on hybrid QA, including pipeline structures, ETL development, lakehouse configurations, and AI-prepared validation frameworks. The move toward zero-ETL, serverless integration, and automated failure management indicates that hybrid QA ecosystems have matured. These innovations help QA teams proactively address data quality issues and deliver reliable analytics and consistent AI systems within a hybrid cloud setting.

References

[1] Pulicharla, M. R. (2025). Next-Generation Data Pipeline Architectures for Real-Time Cloud Analytics: A Novel Framework.

[2] Bhola, M., & Bajeja, S. (2025). Enhancing Cloud-Native Relational Database Systems: Proposed Design Patterns for AWS RDS Application. SN Computer Science, 6(5), 556.

[3] Sankaranarayanan, S. (2025). The Role of Data Engineering in Enabling Real-Time Analytics and Decision-Making Across Heterogeneous Data Sources in Cloud-Native Environments. International Journal of Advanced Research in Cyber Security (IJARC), 6(1).

[4] Krishnan, P. (2025). Cloud-native data conversion for Medicare & Medicaid: A scalable foundation for analytics and AI. Journal of Tianjin University Science and Technology, 58(8).

[5] Gupta, S. (2025, August). Modern Data Engineering for Cloud ETL, Migration, and Scalable Analytics. In Proceedings of the International Conference on Sustainability, Innovation & Technology (ICSIT) (Vol. 2025).

[6] AbouZaid, A., Barclay, P. J., Chrysoulas, C., & Pitropakis, N. (2025). Building a modern data platform based on the data lakehouse architecture and cloud-native ecosystem. Discover Applied Sciences, 7(3), 1-22.

[7] Karri, S. B. R., Devalla, V. K., Bojja, R. K., & Pandey, M. S. (2025, April). An Architecture for Model Monitoring System with Automated Data Validation and Failure Handling. In 2025 3rd International Conference on Communication, Security, and Artificial Intelligence (ICCSAI) (Vol. 3, pp. 1960-1966). IEEE.

[8] Khan, J., Liang, W., Mary, B. J., Hamzah, F., John, B., & Blessing, M. (2025). The Evolution of ETL Processes in the Age of Cloud Computing and Microservices: From Traditional Batch Loading to Serverless Data Integration.

[9] Naayini, P. (2025). Building AI-driven cloud-native applications with Kubernetes and containerization. International Journal of Scientific Advances (IJSCIA), 6(2), 328-340.

[10] ALI, Z. (2025). AI-Ready Data Infrastructure: A Review of Zero-ETL, Declarative Pipelines, and Data Contracts in Modern Data Engineering.