

Migrating from traditional data warehouses or data lakes to a modern **lakehouse** architecture is more than a technology upgrade; it's a substantial paradigm shift that reshapes how organizations store, process, and analyze data. One of the top concerns during this transition is *data quality*: ensuring that the insights you derive continue to be accurate, consistent, and trustworthy.



In this blog, we'll dive deep into the landscape of data quality validation during a lakehouse migration, with a focus on Azure and Databricks-based implementations. We'll discuss:

- The essential differences between data lakes, warehouses, and lakehouses
- How Databricks and Snowflake deliver depth in lakehouse and warehouse paradigms
- Leveraging data quality frameworks like **Great Expectations** and **reconciliation checks**
- Governance, lineage, and semantic modeling as foundations for high data quality
- Real-world considerations and best practices I've learned from Azure/AWS migrations

Lakehouse vs Warehouse vs Data Lake: What You Need to Know

Before we drill into testing data quality, it's critical to understand the foundational concepts of the architectures involved:

Data Lake

A data lake is a centralized repository that stores raw, unstructured, or semi-structured data at scale.

- **Strengths:** Low cost, flexible schema-on-read, great for exploration
- **Weaknesses:** Often lacks strong governance, schema enforcement, and can lead to data swamp scenarios without strict controls

Data Warehouse

Warehouses store structured data optimized for analytics and reporting, with enforced schemas and ACID guarantees.



- **Strengths:** Rigorous data quality, performance optimized for BI
- **Weaknesses:** Less flexible for semi-structured data, slower ingestion for complex pipelines

Lakehouse

The lakehouse combines the scalability and flexibility of data lakes with the structured governance and performance of warehouses.

- Enables ACID transactions on data stored in open formats like Parquet
- Supports both streaming and batch workloads with governance
- Includes semantic layers, lineage, and centralized data quality frameworks

In practice, a lakehouse aims to unify the best of both worlds, but this also means testing data quality requires considering both raw data characteristics and structured governance mechanisms.

Depth in Delivery: Databricks and Snowflake

In recent years, two platforms stand out in lakehouse and warehouse innovation: **Databricks** and **Snowflake**.

Criteria Databricks Snowflake Architecture Lakehouse, built on open formats (Delta Lake), supports streaming & batch Cloud data warehouse with some lakehouse features (external tables via Snowflake Elastic Data Lake) Platform Support Available on Azure, AWS, GCP Available on Azure, AWS, GCP Governance Strong lineage & data catalog via Unity Catalog, supports fine-grained access Strong data sharing, governance via Snowflake Data Marketplace & Data Clean Room Data Quality Tools Integration with Great Expectations, supports reconciliation via notebooks Built-in analytics functions, third-party integrations for data quality

From my experience running migrations on both Azure and AWS clouds, Databricks' open framework and robust governance features through Unity Catalog make it more natural to build end-to-end lineage and semantic layers. Snowflake's managed warehouse experience simplifies data ingestion pipelines but often requires additional tooling for comprehensive lineage and quality tests.

Implementing Data Quality Testing in Lakehouse Migrations

1. Establish Clear Governance and Lineage

Two critical questions I always ask in vendor evaluations or internal projects are:

- **Where does lineage live?** Is the data lineage captured automatically and accessible to data custodians and data consumers?
- **Who owns the data quality tests?** Are they team-owned, automated, and maintainable via CI/CD pipelines?

Successful lakehouse migrations leverage tools like Databricks Unity Catalog on [suffolknewsherald.com](https://www.suffolknewsherald.com) Azure or AWS Lake Formation to build automated lineage graphs. This visibility is the baseline for effective quality testing:

- Enables impact analysis before and after migration
- Supports semantic modeling that standardizes domain definitions
- Allows monitoring data quality metrics over time

2. Use Data Quality Frameworks: Great Expectations

Great Expectations (GE) is an open-source data quality framework widely adopted in lakehouse projects.

Key attributes of GE in migrations:

- **Declarative Expectations:** Define rules for data such as null checks, value ranges, distribution matches, and uniqueness
- **Integration with Data Platforms:** GE integrates natively with Databricks Delta Lake and Azure Synapse, allowing validation during ETL/ELT jobs
- **Data Docs and Dashboards:** Automatically generates human-readable documentation for data quality results, making it accessible to different stakeholders

In one large Azure migration I led, we implemented GE expectation suites linked to ingestion pipelines in Synapse and Databricks notebooks. The result was a reduction in production incidents caused by data schema drift and anomalies.

3. Reconciliation Checks

Besides rule-based validation, reconciliation checks compare row counts, aggregates, and hash sums across source and target systems.

Typical reconciliation steps include:

1. **Row Counts:** Ensure the same number of rows ingested between the old warehouse and new lakehouse tables
2. **Aggregate Metrics:** Totals and averages on key business metrics should match, within defined tolerances
3. **Hash Totals:** Row-level hash comparisons can confirm record integrity

Automated reconciliation scripts should run as part of your CI/CD pipeline with clear logging and alerting on mismatches. Databricks notebooks combined with Azure DevOps pipelines (or AWS CodePipeline) work well for this.

Governance and Semantic Modeling: Foundations for Trust

In my experience, no amount of data quality testing can fix foundational gaps in governance and semantic modeling.

- **Governance:** Role-based access, data cataloging, policy enforcement, and audit logging must be baked in. Databricks Unity Catalog or Microsoft Fabric data governance features offer enterprise-grade options.
- **Semantic Modeling:** Creating reusable business views and domain models ensures consistent metrics, labels, and definitions across teams. Lakehouse migrations are an ideal time to clean up and enforce semantic layers—don't defer this.

If your architecture diagrams show only raw tables and omit semantic models—or if the vendor talk is “AI-ready” without addressing who owns and audits these layers—that’s a red flag. Trustworthy data quality emerges from disciplined semantic management.

Real-World Lessons From Azure and AWS Lakehouse Migrations

Having led 11 years of migrations on Azure and AWS, here are some practical tips you won't find in marketing materials:

1. **Never trust pilot-only success stories:** Early pilots often bypass data governance and lineage. Real data quality issues surface once scaled.
2. **Implement CI/CD and Infrastructure as Code (IaC):** Automate deployment of data pipelines, quality tests, and governance configurations. Manual processes inevitably cause drift.
3. **Lineage visibility is non-negotiable:** You need automated lineage updated in real time, not Excel sheets or manual documentation.
4. **Data quality must be a continuous guardrail:** Set expectations early that data quality tests are part of the product, with ownership clear and visibility empowered.

Summary Table: Data Quality Activities During Lakehouse Migration

Activity	Tools / Platform	Why Important
Lineage Capture	Databricks Unity Catalog, Azure Purview, AWS Lake Formation	Visibility into data flow for impact analysis and root cause debugging
Data Quality Tests	Great Expectations	Integrated into Databricks / Synapse pipelines. Automated validation prevents bad data promotion to downstream consumers.
Reconciliation Checks	Custom SQL scripts, Databricks notebooks, Azure DevOps pipelines	Verify correctness and completeness post-migration
Semantic Layer Modeling	Databricks SQL Dashboards, Microsoft Fabric semantic models	Ensures consistent business definitions and trust in metrics
Governance and Access Controls	Unity Catalog, Azure Active Directory, Snowflake RBAC	Protects sensitive data and enforces compliance

Closing Thoughts

Testing data quality during a lakehouse migration requires bridging the rigor of traditional data warehouses with the flexibility of data lakes, enabled by new platforms like Databricks and Snowflake on Azure/AWS clouds.

Don't fall for vague “AI-ready” or “next-gen” catchphrases—demand clear ownership of lineage, automated, declarative data quality frameworks such as Great Expectations, and continuous reconciliation and governance baked into your pipelines.

With thorough quality tests, clear semantic modeling, and transparent lineage, your lakehouse migration will not just be a technology change, but a trusted transformation that empowers your organization with confidence in its data-driven decisions.

Feel free to reach out if you want a no-nonsense conversation about your lakehouse migration and how to build a bulletproof data quality framework.