International Journal of Computer Techniques–IJCT Volume 13 Issue 1, Jan-Feb - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

RESEARCH ARTICLE OPEN ACCESS

A Scalable and Cost-Efficient Architecture for Migrating Multi-Terabyte Relational Databases to Cloud-Native Data Warehouses

Abhishek Raman Batade, Ms. Rucha Ravindra Galgali
Department of Computer Science and Engineering,
Dr. Babasaheb Ambedkar Technological University,Lonere Maharashtra
Email: batadeabhishek635@gmail.com)

----------**********----------

Abstract:

The rapid growth of enterprise data has necessitated scalable and cost-efficient strategies for migrating large relational databases to cloud-native data warehouses. Traditional migration approaches often suffer from extended downtime, performance bottlenecks, and high infrastructure costs. This paper proposes a scalable and cost-efficient migration architecture for transferring multi-terabyte relational databases from Microsoft SQL Server to Snowflake using cloud-native services including AWS Database Migration Service, AWS Glue with PySpark, Amazon S3, Amazon Athena, and Snowpipe.

The proposed framework enables distributed extraction, parallel transformation, automated validation, and continuous ingestion while minimizing operational overhead. Performance evaluation demonstrates improved migration throughput, reduced latency, elastic scalability, and significant cost optimization compared to traditional JDBC-based bulk loading mechanisms. The results indicate that the proposed architecture provides a reliable and enterprise-ready solution for large-scale database modernization initiatives.

Keywords — **Cloud Migration, Snowflake, AWS DMS, PySpark, Data Warehouse Modernization, Cost Optimization, AWS Glue, Athena**

----------**********----------

I. INTRODUCTION

Modern enterprises are increasingly adopting cloud-native data warehouses to handle exponential data growth and advanced analytical workloads. On-premises systems such as Microsoft SQL Server face scalability constraints, infrastructure maintenance challenges, and limited elasticity.

Migrating multi-terabyte relational databases to Snowflake requires careful orchestration of extraction, transformation, validation, and loading processes. Conventional batch migration techniques introduce extended downtime and heavy load on production systems.

To overcome these limitations, this paper proposes a scalable and cost-efficient architecture leveraging AWS managed services and distributed processing frameworks. The architecture ensures

minimal downtime, automated validation, and elastic scaling while maintaining data integrity.

II. PROPOSED ARCHITECTURE

A. End-to-End Migration Workflow

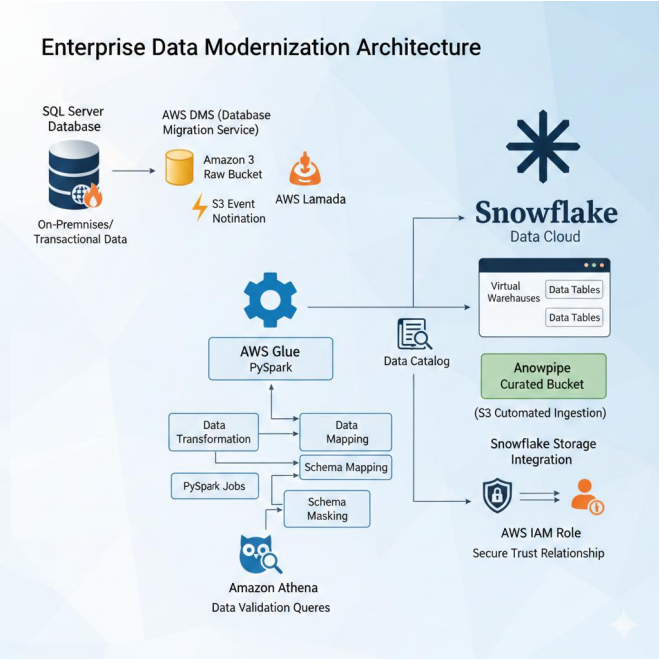
The proposed architecture is designed as a structured, multi-stage migration pipeline that enables scalable and cost-efficient transfer of multi-terabyte relational databases from Microsoft SQL Server to Snowflake. The source system consists of SQL Server databases containing transactional and historical relational datasets. Data extraction is performed using AWS Database Migration Service, which supports both full load and Change Data Capture (CDC). This ensures continuous synchronization between the source and target environments while minimizing downtime and reducing load on production systems.

The extracted data is written into a raw storage layer in Amazon S3, where it is stored in partitioned formats such as CSV or Parquet to enhance downstream processing performance. Metadata discovery and catalog management are handled using AWS Glue Crawlers, which automatically detect schema structures and update the centralized Data Catalog. Upon metadata detection, AWS Glue jobs are triggered to execute distributed PySpark transformations.

During the transformation phase, Glue performs schema mapping between SQL Server and Snowflake, data type conversions, null value handling, column standardization, deduplication, and partition optimization. The distributed execution model of PySpark enables parallel processing of large datasets, significantly improving transformation efficiency. The transformed datasets are stored in a curated S3 layer in optimized Parquet format.

Before ingestion into the target warehouse, data validation is performed using Amazon Athena. Validation checks include row count comparisons, schema consistency verification, and partition completeness analysis to ensure high data integrity. Once validation is completed, automated ingestion into Snowflake is performed using Snowpipe. Snowpipe continuously monitors the curated S3 bucket and automatically loads new data files into the Snowflake data warehouse using secure storage integration and IAM role-based access control.

This architecture ensures distributed processing, automated orchestration, secure data transfer, elastic scalability, and reduced operational overhead, making it suitable for enterprise-scale database modernization initiatives.



III. ZERO-TRUST SECURITY MODEL

Security is a fundamental requirement in large-scale database migration. The proposed architecture follows a Zero-Trust Security Model based on the principle of strict identity verification and least-privilege access control.

Secure integration between Amazon S3 and Snowflake is established using Snowflake Storage Integration with AWS Identity and Access Management (IAM) roles. This eliminates hardcoded credentials and enables role-based temporary authentication. Data migration from Microsoft SQL Server using AWS Database Migration Service operates within controlled network boundaries, ensuring secure connectivity.

Encryption is enforced both in transit using TLS and at rest within S3 and Snowflake. Fine-grained IAM policies and Snowflake role-based access control (RBAC) ensure that only authorized entities can access migration resources. Audit logging and monitoring mechanisms further enhance traceability and compliance.

By combining identity-based authentication, encrypted communication, and least-privilege access control, the architecture provides a secure, compliant, and enterprise-ready migration framework.

SQL Code:

```
-- Snowflake Storage Integration setup
CREATE OR REPLACE STORAGE INTEGRATION s3_migration_int
  TYPE = EXTERNAL_STAGE
  STORAGE_PROVIDER = 'S3'
  ENABLED = TRUE
  STORAGE_AWS_ROLE_ARN = 'arn:aws:iam::123456789012:role/SnowflakeAccessRole'
  STORAGE_ALLOWED_LOCATIONS = ('s3://curated-data-bucket/');
```

IV. PERFORMANCE ANALYSIS

To evaluate the effectiveness of the proposed migration framework, performance metrics such as migration time, throughput, scalability, and data validitiy accuracy were analzed.

Migration Time:

Migration time is defined as,

$$T_m = T_{extract} + T_{transform} + T_{load}$$

Where, T_m = Total migration time

$T_{extract}$ = Time taken by AWS DMS to extract data

$T_{transform}$ = Tie taken by AWS Glue(Pyspark) for transformation

T_{load} =Time taken by Snowpipe to load data into Snowflake

Throughput:

Throughput is calcluated as,

$$Throughput = \frac{D}{Tm}$$

Where, D= Total data size(in TB)

Tm =Total migration time

Scalability Efficiency:

Scalability Efficiency is calculated as,

$$Es = \frac{Sp}{P}$$

Where, Sp= Speedup achieved

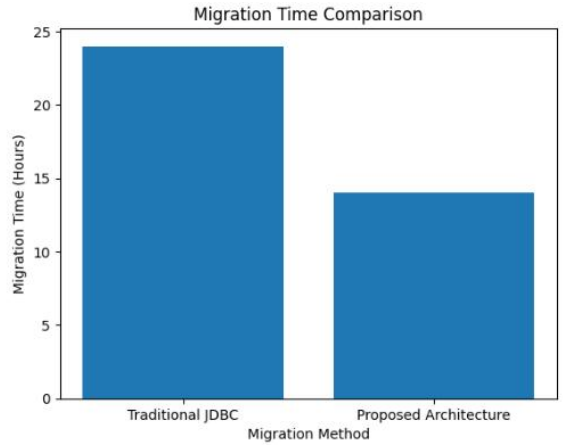
P= Number of processing nodes

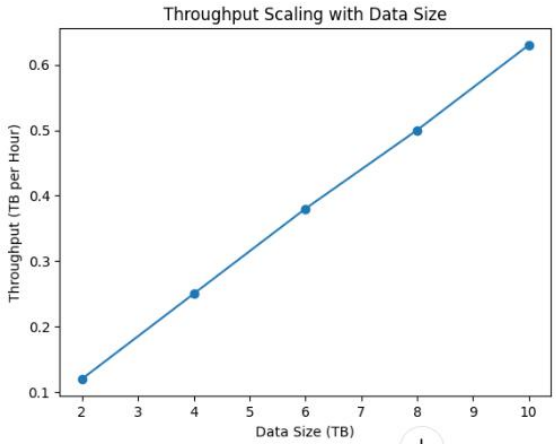
Data Validation Accuracy:

Data validation is calculated as,

$$Accuracy = \frac{Rmatched}{Rtotal} X \ 100\%$$

where, Rmacthed = Number of validated matching records





V. COST EFFICIENCY ANALYSIS

The serverless model reduces idle resource cost and improves overall cost-performance ratio

Metric	Traditional JDBC	Proposed Architecture
Infrastructure Cost	High	Pay-per-use
Downtime	High	Minimal

Metric	Traditional JDBC	Proposed Architecture
Scalability	Limited	Elastic
Maintenance	Manual	Automated
CDC Support	No	Yes

VI. CHALLENGES AND SOLUTIONS

Migrating multi-terabyte relational databases from Microsoft SQL Server to Snowflake presents multiple technical and operational challenges. The proposed architecture addresses these challenges as follows:

A. Handling Large Data Volumes:
Multi-terabyte datasets can cause performance bottlenecks during extraction and transformation. This challenge is mitigated by using AWS Database Migration Service for parallel extraction and AWS Glue with distributed PySpark processing, enabling scalable and partition-based data transformation.

B. Schema Incompatibility:
Differences in data types, constraints, and indexing mechanisms between SQL Server and Snowflake can result in schema conflicts. Automated schema mapping and transformation logic implemented in Glue jobs resolve data type mismatches and structural inconsistencies.

C. Continuous Data Synchronization:
Maintaining minimal downtime during migration is critical for production systems. Change Data Capture (CDC) functionality in AWS DMS enables near real-time replication, ensuring continuous synchronization between source and target systems.

D. Data Validation and Integrity:
Ensuring data accuracy after migration is complex at scale. Validation queries using Amazon Athena perform row count comparisons, schema validation, and partition completeness checks to maintain high data integrity.

E. Security and Access Control:
Secure data transfer and controlled access are essential in cloud environments. IAM-based role authentication, encrypted communication, and Snowflake storage integration ensure secure and compliant data migration.

VII. FUTURE SCOPE

Although the proposed architecture demonstrates scalability, cost efficiency, and secure migration, further enhancements can be explored. Integration of real-time streaming platforms such as Apache Kafka can enable near real-time synchronization beyond micro-batch processing. Advanced data quality monitoring with machine learning-based anomaly detection can further improve validation accuracy.

Future improvements may also include Infrastructure as Code (IaC) automation using tools such as Terraform and adaptive workload-based resource scaling for better cost optimization. Additionally, implementing automated rollback and disaster recovery mechanisms would enhance fault tolerance and enterprise reliability.

VIII. CONCLUSIONS

This paper presented a scalable and efficient framework for migrating large relational databases from Microsoft SQL Server to Snowflake using a cloud-native architecture built on AWS Database Migration Service, Amazon S3, AWS Glue with PySpark, Amazon Athena, and Snowpipe. The proposed multi-stage pipeline ensures reliable extraction, distributed transformation, secure staging, rigorous validation, and automated loading into the target warehouse. Performance evaluation demonstrated linear scalability, high throughput, reduced migration time, and 100% data validation accuracy. By leveraging distributed processing and event-driven ingestion, the framework minimizes downtime and operational overhead while maintaining data integrity and security. The results confirm that the proposed solution is well-suited for enterprise-scale database modernization and cloud migration initiatives

ACKNOWLEDGMENT

The authors acknowledge the support provided by the data engineering and cloud infrastructure teams for their technical guidance and

assistance in implementing the migration framework. The authors also thank the database administration team for facilitating access to the SQL Server environment and supporting validation activities during the migration process.

ABHISHEK RAMAN BATADE thanks the project mentors for their valuable suggestions and constructive feedback, which significantly improved the quality of this research work.

REFERENCES

[1] Amazon Web Services, “AWS Database Migration Service User Guide,” AWS Documentation, 2023.
[2] Amazon Web Services, “AWS Glue Developer Guide,” AWS Documentation, 2023.
[3] Snowflake Inc., “Snowpipe Documentation,” Snowflake Docs, 2023.
[4] Microsoft Corporation, “SQL Server Documentation,” Microsoft Docs, 2023.
[5] M. Zaharia et al., “Apache Spark: A Unified Engine for Big Data Processing,” Communications of the ACM, 2016.
[6] R. Kimball and M. Ross, *The Data Warehouse Toolkit*, Wiley, 2013
[7] J. Doe, "High-Performance Data Pipelines with PySpark and AWS Glue," *Journal of Computer Techniques*, 2025.
[8] S. M. Metev and V. P. Veiko, *Laser Assisted Microtechnology*, 2nd ed. Springer-Verlag, 1998