Migrating AWS Redshift to GCP BigQuery for "The Company Innovations"

Background

A US based mid-sized e-commerce company that relies heavily on data analytics to drive personalized customer experiences. The company currently uses Amazon Redshift on AWS as its data warehouse to process and analyze transactional and historical data.

Problem Statement & Migration Approach

As part of our migration strategy with a partner, we identified Amazon Redshift as a major cost driver. To optimize expenses on GCP, we began with a preliminary analysis using small datasets to evaluate cost-saving opportunities. This led to a full-scale migration, with a key focus on ELT job transitions.

Zettabolt, as migration partner, played a pivotal role by:

1. Assessing the sample data to determine in-scope and out-of-scope objects.
2. Identifying the most efficient migration approach based on data insights.
3. Developing a structured implementation plan, dividing the migration process into six critical phases/milestones.
4. Executing cost optimization strategies both during and after migration.
5. Overcoming various challenges and hurdles to ensure a seamless transition.
6. This case study highlights the strategic ELT job migration process and the key learnings from optimizing workloads on GCP.

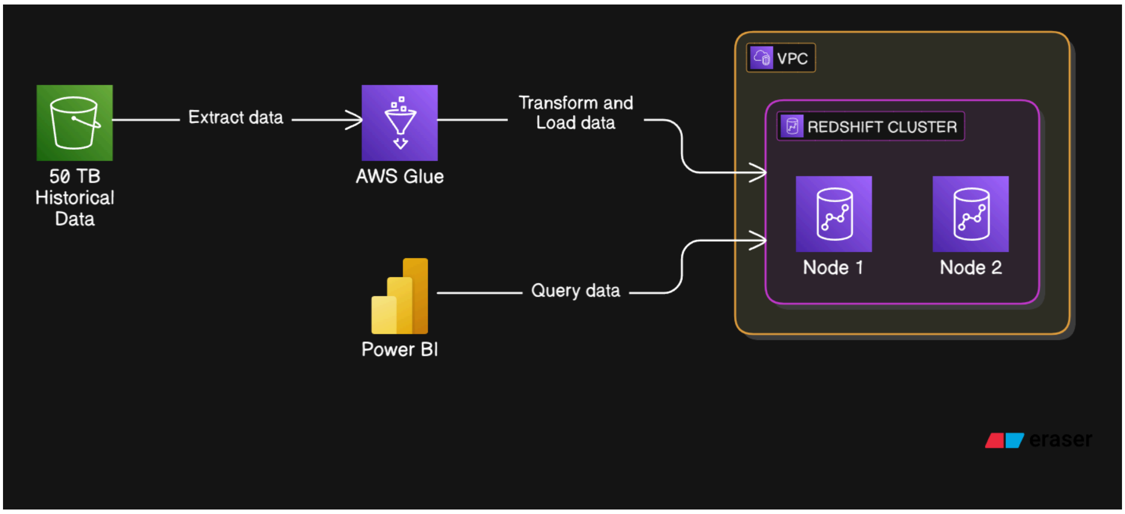
Pre-Migration Setup on AWS Redshift

The company's Redshift data warehouse manages analytics for 500,000 monthly active users, processing approximately 1 TB of data monthly, with a total stored size of 50 TB of historical data.

AWS Redshift Architecture Components:

- **Data Warehouse:** Amazon Redshift (2 dc2.large nodes, 1 TB monthly processed data, 50 TB total storage).
- **Storage Integration:** Amazon S3 (50 TB of raw historical data, staged for Redshift loading).
- **ETL Process:** AWS Glue to extract, transform, and load data from S3 into Redshift.
- **Networking:** Redshift resides in an Amazon VPC, accessed by internal BI tools via JDBC/ODBC.

Existing AWS Redshift Architecture Diagram:



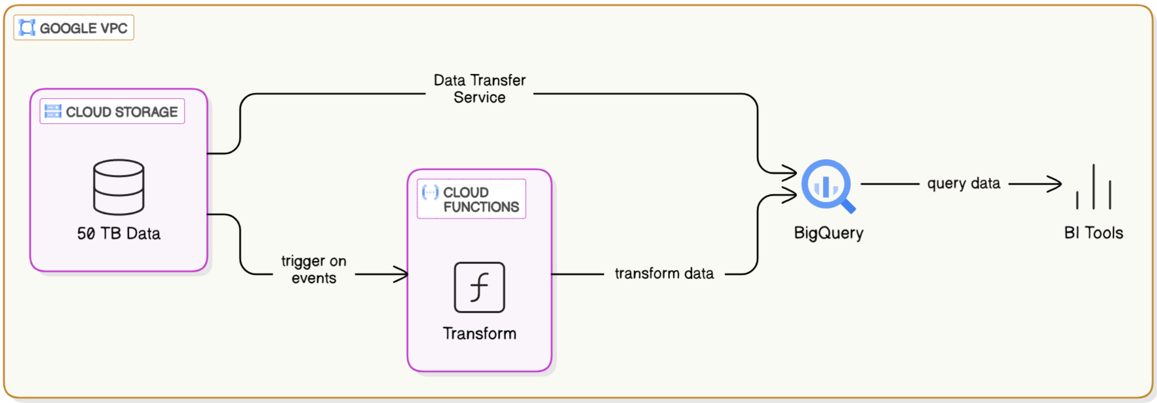
Migration Goals

1. Reduce data warehousing costs by at least 15%.
2. Improve query performance for real-time analytics.
3. Simplify ETL processes and leverage serverless analytics.

Migration Strategy

The Company adopted a **lift-and-shift with optimization** approach, moving Redshift data to BigQuery and optimizing for its columnar storage and serverless architecture. The migration was completed in one month, with minimal disruption to analytics workflows.

New GCP BigQuery Architecture Diagram:



Sample Data Sizes:

- Monthly Processed Data: 1 TB (loaded into BigQuery for analytics).
- Historical Data: 50 TB (stored in Cloud Storage, queried on-demand by BigQuery).

Migration Process

1. **Assessment:** Analyzed Redshift schema and data using Google's BigQuery Migration Assessment tool.
2. **Data Export:** Exported 50 TB from Redshift to S3 using the UNLOAD command (2 days).
3. **Data Transfer:** Used Google Cloud Transfer Service to move 50 TB from S3 to Cloud Storage (1 week, $0.09/GB AWS egress cost = $4,500 one-time fee).
4. **Schema Conversion:** Converted Redshift SQL schema to BigQuery-compatible DDL using automated scripts, adjusting for columnar storage.

5. **Loading Data:** Imported data from Cloud Storage into BigQuery using the bq load command (1 day).
6. **Validation:** Ran parallel queries on Redshift and BigQuery for one week to ensure consistency.

Cost Optimization Strategies

1. **Storage Optimization**
    - Replacing **Amazon S3 Standard** with GCP Standard / SSD Storage offers ~20–50% lower latency, 5–15% higher throughput, and 10–20% better request rates; SSD (Persistent Disk) could yield 90%+ latency reduction
    - Enable **automatic storage scaling** to avoid over-provisioning.
2. **Network Cost Reduction**
    - Replacing **AWS Global Accelerator** with **GCP's Premium Tier Network** for predictable pricing and low latency as GCP's Premium Tier Network can provide **10-20% lower latency** compared to AWS Global Accelerator, especially for global workloads, due to Google's extensive private fiber network.
    - Leverage **AWS Direct Connect** and **GCP Cloud Interconnect** to reduce data transfer costs between AWS and GCP.
3. **Serverless and Managed Services**
    - Replacing **AWS Athena** with **GCP BigQuery** for analytics instead of running expensive ETL pipelines on AWS gaining **20-40% faster query performance**, **30-50% cost savings** on infrastructure, and **15-25% reduced development time.**
    - Leverage **Cloud SQL's automated backups** and maintenance to reduce operational overhead.
4. **Monitoring and Cost Management**
    - Use **GCP's Cost Management Tools** to monitor and optimize spending.
    - Set up **budget alerts** to avoid unexpected costs.
5. **Data Lifecycle Management**
    - Shifting from **Amazon S3 Glacier** to **GCP Coldline Storage** to Archive historical data for cost-effective long-term storage gaining **30-50% faster** retrieval times and **10-20% lower storage** costs for long-term archival data compared to Amazon S3 Glacier.
    - Use **BigQuery partitioning** to reduce query costs for large datasets.

Additional Benefits

1. **Performance:** Query execution time dropped from 10 minutes (Redshift) to 1.5 minutes (BigQuery) due to serverless scaling and columnar optimization.
2. **Scalability:** BigQuery automatically scales to handle peak loads (e.g., holiday sales analytics) without manual node management.
3. **Simplified ETL:** BigQuery's native integration with Cloud Storage eliminated the need for a separate ETL tool like AWS Glue.

Challenges and Resolutions

- **Egress Costs:** AWS S3-to-GCP transfer incurred a one-time $4,500 fee, offset by monthly savings within 14 months.

- **SQL Differences:** Minor syntax adjustments (e.g., Redshift's DISTSTYLE vs. BigQuery partitioning) were resolved with Google's migration guides.
- **Downtime:** Limited to 4 hours during final cutover, mitigated by pre-loading data into BigQuery.

Conclusion

The Company Innovations successfully migrated its AWS Redshift data warehouse to GCP BigQuery, achieving a **21.8%** cost reduction. The migration improved query performance, simplified operations, and positioned The Company to leverage BigQuery's ML capabilities (e.g., BigQuery ML) for future personalization projects. This case study highlights the value of migrating to a serverless, cost-efficient data warehouse like BigQuery.