Batch Loading Data to Google BigQuery using Google Data Fusion

Suhas Hanumanthaiah

suhas.h@hotmail.com

Abstract:

Efficient data ingestion into cloud-based data warehouses is critical for enabling timely analytics and informed decision-making in modern enterprises. This paper explores a practical and scalable solution for batch loading large volumes of structured and semi-structured data into Google BigQuery by leveraging Google Cloud Data Fusion (CDF). BigQuery, a serverless and highly scalable analytical database, excels at processing petabyte-scale datasets but requires efficient upstream data integration to unlock its full potential. Google Cloud Data Fusion, built on the Cask Data Application Platform (CDAP), offers a visual, code-free interface for designing, managing, and executing ETL pipelines. The research outlines how CDF integrates seamlessly with other GCP services such as Dataproc to orchestrate resource-optimized data pipelines. Through a well-defined architectural framework and deployment model, the paper demonstrates how CDF can be employed to create modular, reusable, and auto-scaling batch data workflows, delivering operational cost savings and performance benefits. Best practices such as namespace segregation, transformation pushdown, autoscaling clusters, and failure alerting are presented to enhance pipeline efficiency and governance. Additionally, this study identifies existing limitations in real-time data ingestion capabilities within CDF and proposes future work to evaluate its streaming performance using Pub/Sub and Spark Streaming. Overall, the approach provides a robust and cost-effective foundation for enterprise-grade data integration on Google Cloud, with strong potential for hybrid batch-streaming models in future research.

Keywords: Cloud Data Fusion (CDF), Google BigQuery, Batch Data Processing, Google Cloud Platform (GCP).

1. INTRODUCTION

1.1. Google Cloud Ecosystem

Google Cloud Platform (GCP) offers a comprehensive ecosystem of services tailored for big data management, with BigQuery serving as a cornerstone for analytical workloads [1]. Its highly scalable, serverless architecture enables organizations to process and analyze massive datasets with remarkable efficiency, thereby supporting critical business intelligence and data science initiatives [1]. However, the effective ingestion of large volumes of data into BigQuery, especially in batch operations, presents distinct challenges related to throughput, reliability, and cost-efficiency [3]. This paper delves into leveraging Google Cloud Data Fusion (CDF) as a robust orchestrator for batch loading data into BigQuery, emphasizing its capabilities in streamlining extract, transform, load processes and ensuring data integrity.

1.2. Google BigQuery

BigQuery's architecture is optimized for handling petabyte-scale datasets through its columnar storage format and massively parallel processing capabilities, which facilitate rapid querying and aggregation of complex data structures. This design allows for the segregation of compute and storage, providing inherent scalability and flexibility for diverse analytical requirements. The platform supports standard SQL queries, enabling seamless integration with existing data analysis tools and practices. Additionally, BigQuery supports executing Machine Learning and Artificial intelligence integration. Hence BigQuery is a natural choice for data warehousing in Google Cloud.

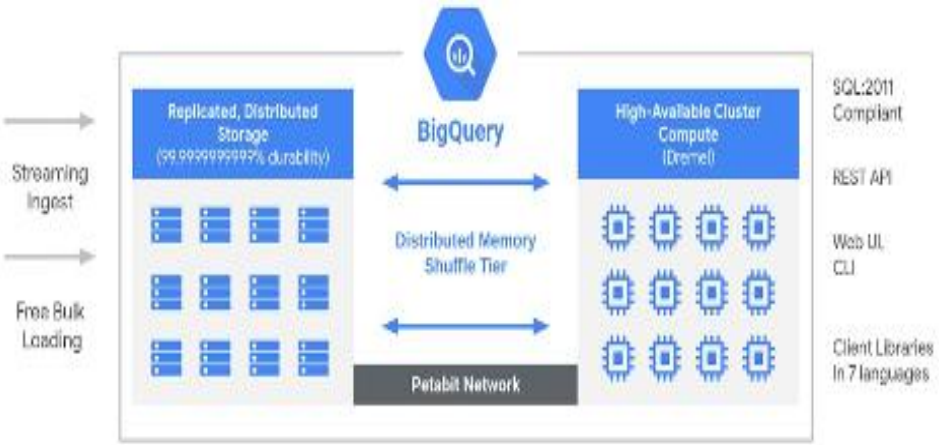
Streaming Ingest
Free Bulk Loading
Replicated, Distributed Storage (99.999999999% durability)
BigQuery
Distributed Memory Shuffle Tier
Petabit Network
High-Available Cluster Compute (Dremel)
SQL:2011 Compliant
REST API
Web UI, CLI
Client Libraries In 7 languages

Fig 1: BigQuery architecture [2]

Furthermore, BigQuery supports Business Intelligence semantic layering with Looker and other third-party tools, enhancing its utility for data visualization and reporting [4]. Its serverless nature abstracts away infrastructure management, allowing users to focus solely on data analysis rather than operational overheads.

1.4. Google Cloud Data Fusion

It is built on Cask Data Application Platform (CDAP), an open-source data integration platform, providing a managed service for developing and managing data pipelines.

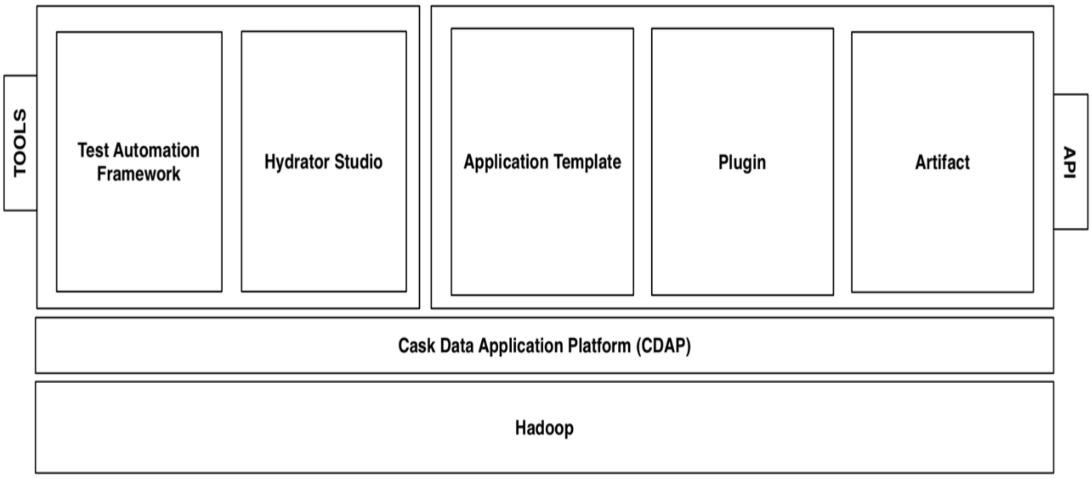

Figure 2.1: Functional Architecture of CDAP Pipelines [13]

As illustrated above, CDAP pipelines use application template to define consist of a definition of its different components, processing, and workflow, in the form of a configuration including MapReduce, Service, Spark Streaming, Worker and Workflow. This architecture facilitates the creation of portable and reusable data integration solutions, abstracting away the underlying infrastructure complexities and enabling seamless deployment across various environments.

In GCP, Hadoop and Spark workloads are managed through services like Dataproc, which provides fully managed clusters for executing big data frameworks [5]. Hence, Data Fusion leverages Dataproc to execute its pipelines, thereby benefiting from Dataproc's optimized resource management and scaling capabilities for large-scale data processing [6] [7].

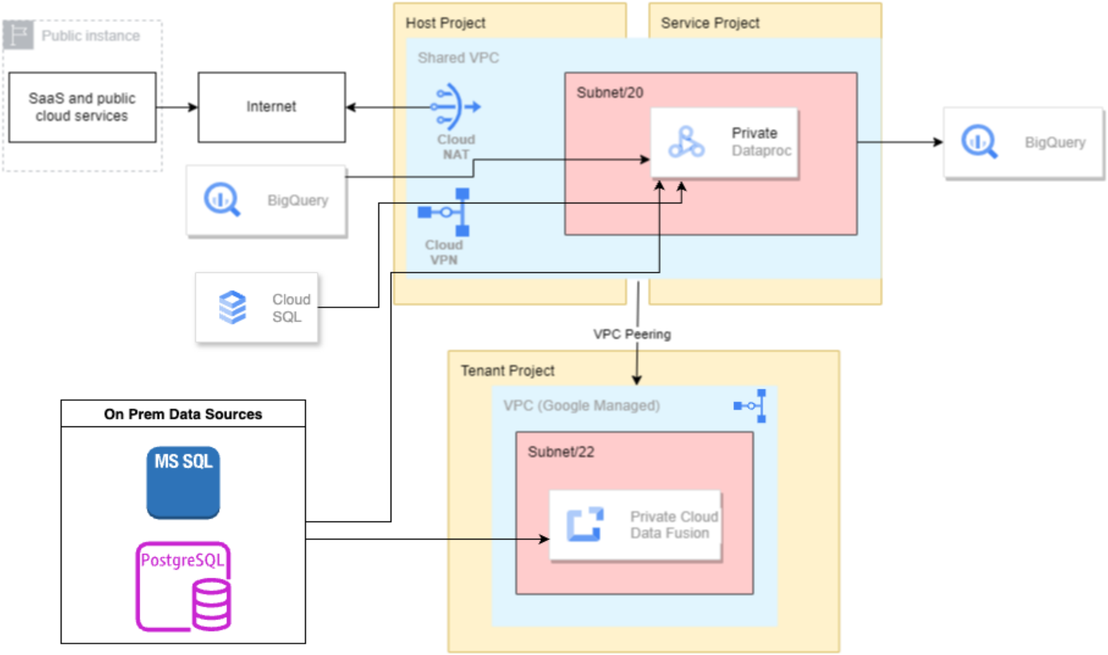

Fig 2.2: Cloud Data Fusion Architecture & Data Flow

The set of services required to build and orchestrate Cloud Data Fusion pipelines and store pipeline metadata are provisioned in a tenant project, inside a tenancy unit. A separate tenant project is created for each customer project, in which Cloud Data Fusion instances are provisioned. The tenant project inherits all the networking and firewall configurations from the customer project.

CDF Studio provides a user-friendly interface for designing, developing, and deploying data pipelines, complete with a drag-and-drop functionality for various data sources, sinks, and transformations. The Studio provides the following administrator controls:

- System Administration: Namespace management, User management, and compute configurations are managed here.
- Namespace Administration: The Studio offers comprehensive namespace management, allowing for the logical segregation of data pipelines and resources, which is crucial for managing multi-tenancy and access control in large organizations [8].

2. METHODOLOGY

The decision tree below details the considerations involved in selecting a data integration service on GCP.

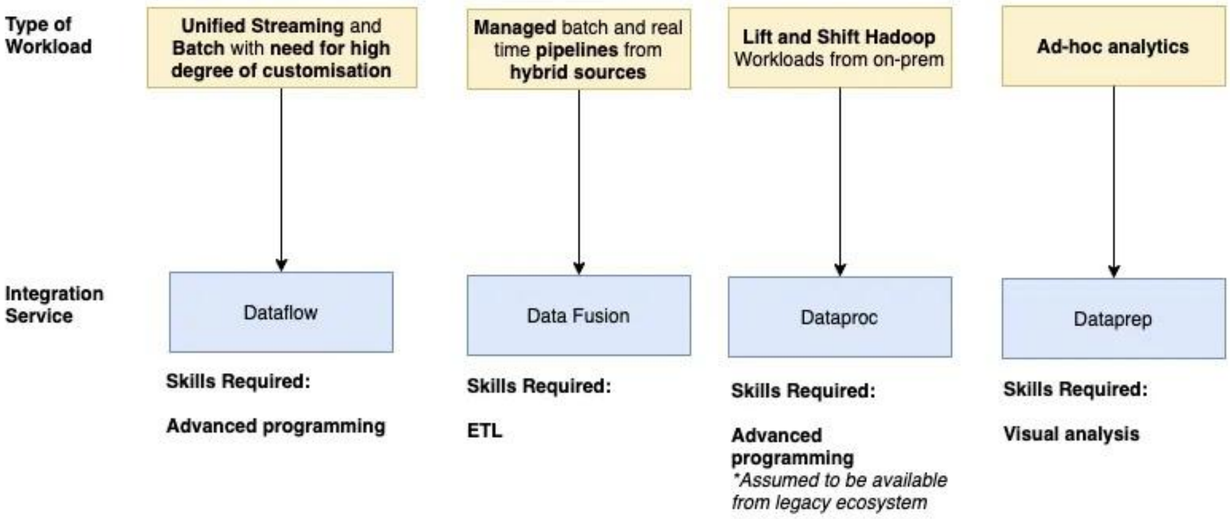
Type of Workload
Unified Streaming and Batch with need for high degree of customisation
Managed batch and real time pipelines from hybrid sources
Lift and Shift Hadoop Workloads from on-prem
Ad-hoc analytics
Integration Service
Dataflow
Data Fusion
Dataproc
Dataprep
Skills Required:
Advanced programming
Skills Required:
ETL
Skills Required:
Advanced programming
*Assumed to be available from legacy ecosystem
Skills Required:
Visual analysis

Figure 3: Data Integration Service offerings in Google Cloud [12]

For use-cases where data integration involves batch processing of diverse data sources with complex transformations, Google Data Fusion emerges as a particularly suitable solution, offering a visual, code-free interface for orchestrating ETL pipelines. All other integrators like Dataflow, Dataproc, Cloud Composer need programming expertise and are better suited for streaming or highly customized data pipelines. This visual approach significantly lowers the barrier to entry for data engineers and analysts, enabling them to construct sophisticated data pipelines without extensive coding knowledge. This can democratize data integration, empowering business analysts who may lack deep programming skills to manage and analyze data effectively, thereby accelerating time to value for data analytics [9].

3. RESULTS

This approach promotes a standardized yet flexible framework for data integration, enabling organizations to achieve a unified view of their disparate data assets, which is critical for informed decision-making and advanced analytics [10]. This unification is particularly crucial in bridging data silos, which often hinder comprehensive data analysis and lead to inconsistent insights across an enterprise [11].

The proposed data flow design from Figure 2.2 delivered cost effective ELT data integration that auto scales and turns off the Dataproc when not in use. This saves cost of operations considerably.

4. BEST PRACTICES

- To manage segregate pipelines to Development, User Acceptance Testing, and Production environments create different namespaces for each environment.
- Create re-usable connection definition at Namespace level
- Enable and use Transformation Pushdown feature for improved performance if the pipeline has multiple complex Join operations
- Use right number of cluster worker nodes for a workload. For provisioned Dataproc cluster, use all available CPU and memory
- Use autoscaling clusters to improve parallelism in pipelines
- Adjust resource configurations in the stages of pipeline where records are pushed or pulled from BigQuery during pipeline execution.
- In Joiner plugin high skewed input to re-sort it. This mitigates the resource utilization limits on BigQuery
- For static cluster use following recommended configurations
  - yarn.nodemanager.delete.debug-delay-sec: Retains YARN logs. Recommended value: 86400 (equivalent to one day)

- ○ yarn.nodemanager.pmem-check-enabled : Enables YARN to check for physical memory limits and kill containers if they go beyond physical memory. Recommended value: false
- ○ yarn.nodemanager.vmem-check-enabled: Enables YARN to check for virtual memory limits and kill containers if they go beyond physical memory. Recommended value: false.
- Use Cloud Data Loss Prevention (DLP) to encrypt PII information in transformation phase of Data Fusion
- Setup on error System alert for Batch pipeline to send email
- Inside each pipeline, on failure send notification on error details

5. CONCLUSION

This paper demonstrates how Google Cloud Data Fusion (CDF) serves as a powerful and flexible orchestration tool for batch data loading into Google BigQuery. By leveraging its graphical interface, pre-built connectors, and seamless integration with Google Cloud services such as Dataproc and BigQuery, CDF significantly reduces the complexity involved in constructing and maintaining enterprise-scale ETL pipelines. The architecture promotes modularity, reusability, and scalability, while offering namespace-driven pipeline segregation to support development, testing, and production environments independently. Moreover, applying best practices like transformation pushdown, autoscaling clusters, and workload-specific configurations allows teams to optimize resource usage and minimize operational costs.

The implementation outlined in this study proved to be cost-effective and robust for batch data ingestion scenarios, particularly where data latency requirements are flexible. However, despite these strengths, certain limitations remain. Notably, Google Cloud Data Fusion's capabilities for real-time or near real-time data processing are less mature compared to its batch processing features. While CDF supports streaming through integrations with Apache Kafka, Pub/Sub, and Spark Streaming, the user experience, performance tuning, and operational observability in these cases are not yet on par with those of specialized streaming tools such as Apache Beam or Google Cloud Dataflow.

Gaps and Future Research

A key gap identified in this research is the limited evaluation of real-time streaming data pipelines using Cloud Data Fusion. The current work focuses exclusively on batch processing, leaving open questions about latency, throughput, and failure recovery in low-latency data flows. Future research could extend this investigation by implementing and benchmarking streaming pipelines in CDF using Google Cloud Pub/Sub as the source and BigQuery as the sink. Comparative studies between CDF, Dataflow, and other native streaming platforms could help determine the most suitable approach for different real-time workloads.

Additionally, deeper exploration into hybrid models that combine real-time and batch ETL workflows using CDF could unlock new architectural patterns for organizations dealing with both high-frequency transactional data and large-volume historical data. Evaluation of event-driven pipeline triggers, stateful transformations, and streaming analytics capabilities in CDF remains an open area for further study. Incorporating observability, error handling, and compliance features into real-time pipelines will also be critical for enterprise readiness. As Google continues to enhance Data Fusion's support for streaming, ongoing research and experimentation will be essential to validate its effectiveness in production-grade real-time data integration scenarios.

REFERENCES:

[1] S. Deochake, V. Channapattan, and G. B. Steelman, “BigBird: Big Data Storage and Analytics at Scale in Hybrid Cloud,” arXiv (Cornell University), Jan. 2022, doi: 10.48550/arxiv.2203.11472.

[2] Rajesh Thallam, “BigQuery explained: An overview of BigQuery's architecture”, Sep. 2020. Available: https://cloud.google.com/blog/products/data-analytics/new-blog-series-bigquery-explained-overview

[3] N. Kumar and S. K. Sharma, “A Cost-Effective and Scalable Processing of Heavy Workload with AWS Batch,” International Journal of Electrical and Electronics Research, vol. 10, no. 2, p. 144, Jun. 2022, doi: 10.37391/ijeer.100216.

[4] T. Furche, G. Gottlob, L. Libkin, G. Orsi, and N. W. Paton, “Data Wrangling for Big Data: Challenges and Opportunities,” Extending Database Technology, p. 473, Nov. 2016, doi: 10.5441/002/edbt.2016.44.

[5] X. Wang and H. Shen, “A Scalable Deep Reinforcement Learning Model for Online Scheduling Coflows of Multi-Stage Jobs for High Performance Computing,” arXiv (Cornell University), Jan. 2021, doi: 10.48550/arxiv.2112.11055.

[6] F. Ullah, S. Dhingra, X. Xia, and M. A. Babar, “Evaluation of Distributed Data Processing Frameworks in Hybrid Clouds,” arXiv (Cornell University), Jan. 2022, doi: 10.48550/arxiv.2201.01948.

[7] U. U. Hafeez, M. Maas, M. Uysal, and R. McDougall, “Rethinking Storage Management for Data Processing Pipelines in Cloud Data Centers,” arXiv (Cornell University), Jan. 2022, doi: 10.48550/arxiv.2211.02286.

[8] M. Zasadziński, M. H. Theodoulou, M. Thurner, and K. Ranganath, “The Trip to The Enterprise Gourmet Data Product Marketplace through a Self-service Data Platform,” arXiv (Cornell University), Jan. 2021, doi: 10.48550/arxiv.2107.13212.

[9] M. Lenzerini, “Data integration,” p. 233, Jun. 2002, doi: 10.1145/543613.543644.

[10] R. Sherman, “Data Integration Design and Development,” in Elsevier eBooks, Elsevier BV, 2014, p. 275. doi: 10.1016/b978-0-12-411461-6.00011-3.

[11] J. Patel, “Bridging Data Silos Using Big Data Integration,” International Journal of Database Management Systems, vol. 11, no. 3, p. 1, Jun. 2019, doi: 10.5121/ijdms.2019.11301.

[12] Neha Joshi, “Designing a Data Lake on GCP with Data Fusion and Composer”, Feb. 2021. Available: https://cloud.google.com/blog/topics/developers-practitioners/architect-your-data-lake-google-cloud-data-fusion-and-composer

[13] CDAP, “How CDAP Data Pipelines Work”, Jan. 2021. Available: https://cdap.atlassian.net/wiki/spaces/DOCS/pages/480313949/How+CDAP+Data+Pipelines+Work

7. Abbreviation

Abbreviation	Full Form
GCP	Google Cloud Platform
CDF	Cloud Data Fusion
CDAP	Cask Data Application Platform
ETL	Extract, Transform, Load
ELT	Extract, Load, Transform
SQL	Structured Query Language
AI	Artificial Intelligence
ML	Machine Learning
DLP	Data Loss Prevention
PII	Personally Identifiable Information
UI	User Interface
CPU	Central Processing Unit
YAML	Yet Another Markup Language
UAT	User Acceptance Testing
IAM	Identity and Access Management