accenture

Telecom Data Pipeline on Google Cloud

Automating Telecom Data Processing Using Google Cloud Services

Project Proposal / Technical Specification Document

By

Alok Ranjan

June 12, 2025

Under the Guidance of

1. Sunil Kaduba Pawar (Data Eng, Mgmt and Governance Manager)

2. Saikat Kumar Pani (Data Eng, Mgmt & Governance Assoc. Manager)

Document Version: 1.0

Last Updated: June 12, 2025

Confidentiality Notice

The document contains proprietary information intended for internal use. Unauthorized distribution is strictly prohibited.

accenture

Table of Contents

SL. No	Sections
1.	Executive Summary
2.	Introduction
3.	System Architecture
4.	Cloud Spanner Schema Design
5.	Ingestion Pipeline
6.	CDC with Spanner Change Streams
7.	BigQuery Integration
8.	Results and Future Scopes
9.	References
10.	Appendix

accenture

Executive Summary

This document presents a comprehensive, end-to-end telecom data pipeline engineered on **Google Cloud Platform (GCP),** designed to efficiently ingest, transform, and manage high-volume telecom data. The primary objective was to establish a scalable, real-time solution capable of handling diverse JSON data, ensuring data integrity through a robust transactional store, and enabling advanced analytics.

The pipeline leverages core GCP services, including Google Cloud Pub/Sub for real-time data ingestion, **Apache Beam** (implemented via Dataflow) for flexible data parsing and transformation, and Google Cloud Spanner for globally distributed, transactional storage. A key innovation is the utilization of **Spanner Change Streams** for efficient Change Data Capture (CDC), allowing real-time propagation of data modifications to Google BigQuery for analytical reporting and business intelligence.

This project showcases expertise in designing and implementing cloud-native data architectures, emphasizing data normalization, transactional consistency, and seamless integration across various GCP services. The resulting solution provides a professional, highly available, and scalable foundation for telecom data management and analysis, demonstrating advanced **data engineering** capabilities.

Introduction

The **telecommunications industry** generates an immense volume of diverse data daily, encompassing call detail records, billing events, network usage, customer interactions, and device telemetry. Extracting actionable insights from this high-velocity, high-volume JSON data is paramount for informed decision-making, operational efficiency, and enhanced customer experience. However, processing and integrating such disparate, semi-structured data sources into a unified, analyzable format presents significant technical challenges regarding scalability, real-time processing, and data consistency.

This project addresses these challenges by outlining an end-to-end cloud-native data pipeline built on Google Cloud Platform (GCP). The primary goals of this solution are to:

- Achieve reliable, low-latency ingestion of high-volume telecom JSON data streams.
- Perform robust data parsing and transformation, converting raw JSON into a structured, normalized schema.
- Store transactional data in a globally consistent, highly available, and scalable database.
- Implement efficient Change Data Capture (CDC) to track and propagate data modifications in near real-time.

- Enable comprehensive business intelligence and analytical reporting by integrating with a powerful data warehouse.

This document details the architectural design, implementation specifics, and technical decisions behind this pipeline, showcasing how it effectively solves the complexities of telecom data management and enables a foundation for advanced analytics.

System Architecture

The end-to-end data pipeline is meticulously designed to handle the ingestion, transformation, storage, and analytical processing of high-volume telecom data. Leveraging Google Cloud Platform's managed services, the architecture ensures scalability, reliability, and real-time capabilities from raw data ingestion to insightful analytics. The data flow is structured to provide a clear separation of concerns across different layers, each optimized for its specific function.

The overall system architecture diagram below illustrates the interconnectedness of these components, showcasing the comprehensive flow of data:

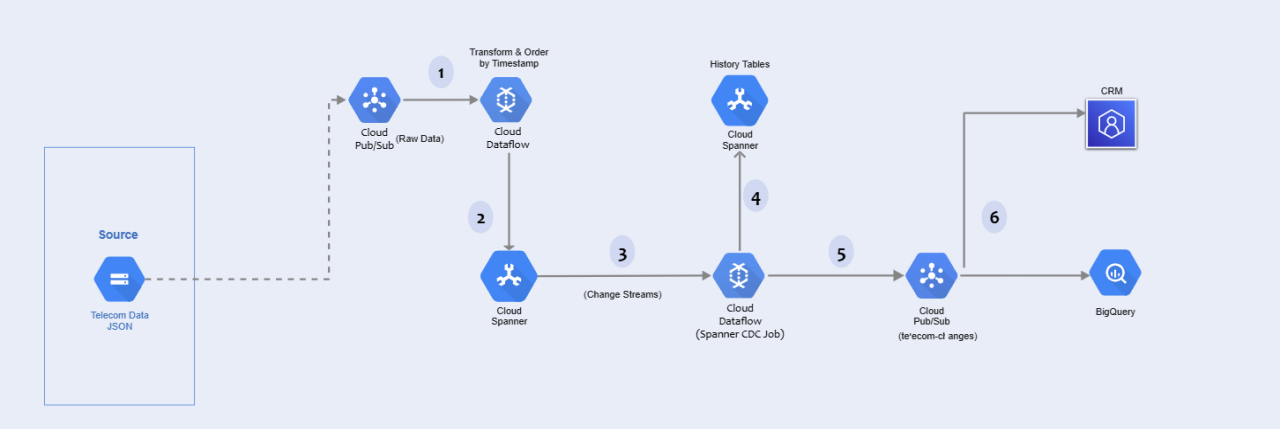


Fig 1. System Architecture

The core components and their roles in facilitating the data flow are as follows:

- **Ingestion Layer: Google Cloud Pub/Sub**

  Google Cloud Pub/Sub serves as the primary entry point for **raw telecom JSON data**. As a fully managed, highly scalable, and asynchronous messaging service, it effectively decouples data producers from consumers. This layer is crucial for reliable, low-latency ingestion of high-volume event streams, ensuring that spikes in data traffic are seamlessly absorbed without overwhelming downstream processing systems.

- **Transformation Layer: Apache Beam pipelines on Google Cloud Dataflow**

  This layer utilizes **Apache Beam**, executed on Google Cloud Dataflow, as the robust engine for data parsing and transformation. A primary Dataflow pipeline consumes raw JSON messages from Pub/Sub, performing complex schema parsing, data cleaning, validation, and normalization. Dataflow's serverless and autoscaling capabilities ensure efficient and fault-tolerant execution of these pipelines, dynamically adjusting resources to meet varying data loads and transforming the semi-structured JSON into a structured format suitable for relational storage.

- **Operational Storage Layer: Google Cloud Spanner**

  Google Cloud Spanner functions as the central operational storage for the structured telecom data. As a globally distributed, strongly consistent, and highly available relational database, Spanner provides ACID transactional properties crucial for maintaining data integrity across multiple normalized tables (**e.g., Customers, Accounts, Plans, Usage, Billing, Devices, Tickets**). Its ability to scale horizontally to petabytes of data and millions of transactions per second makes it ideal for handling the growing operational dataset of a telecom provider.

- **Change Data Capture Layer: Spanner Change Streams**

  Spanner Change Streams are instrumental in enabling real-time Change Data Capture (CDC) from the operational database. This feature provides a low-latency, continuous stream of data modifications (INSERT, UPDATE, DELETE operations) occurring within Spanner tables. The pipeline transforms them into structured rows for subsequent analytical integration. This ensures that downstream systems are kept up-to-date with the latest transactional changes.

- **Analytical Storage Layer: Google BigQuery**

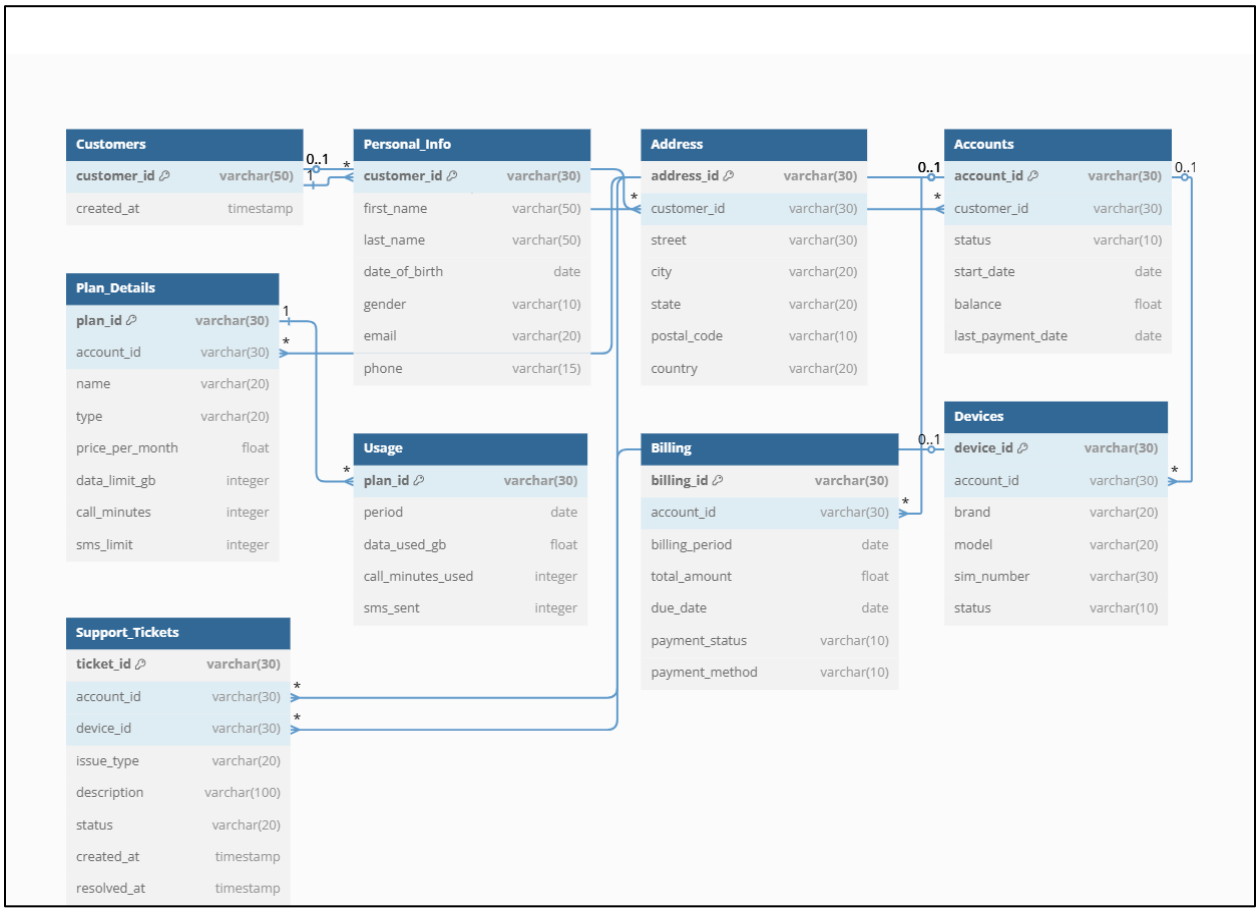
  Google BigQuery serves as the final destination for analytical storage and business intelligence. As a serverless, highly scalable, and cost-effective enterprise data warehouse, BigQuery is optimized for large-scale analytical queries. Data from Spanner (via change streams) is landed in BigQuery, enabling comprehensive reporting, complex aggregations, and integration with Business Intelligence (BI) tools. This layer supports deep insights into customer behavior, service performance, and operational efficiency.

Cloud Spanner Schema Design

Google Cloud Spanner was selected as the primary operational database for this telecom data pipeline due to its unique combination of **global distribution**, strong consistency, and **horizontal scalability**. As a fully managed relational database service, Spanner offers ACID transactional properties, making it ideal for managing the highly critical and interrelated operational data of a telecom provider. Its ability to scale write and read capacity across continents while maintaining transactional integrity was paramount for supporting a growing, real-time data ingestion workload.

Normalized Schema Overview

The schema designed for Spanner adheres to a normalized relational model, focusing on reducing data redundancy and ensuring data integrity across various telecom entities. This approach facilitates efficient querying, simplifies data updates, and supports complex transactional operations inherent to customer, account, and service management within a telecom environment. The design specifically models the core entities and their relationships, laying a robust foundation for operational processes and subsequent analytical integration.



accenture

Key Tables and Relationships

The structured telecom data is organized into several distinct tables within Spanner, each representing a key business entity. Relationships between these tables are established through primary and foreign keys, allowing for comprehensive data retrieval and transactional consistency:

- **Customers:** Stores core customer demographic and contact information.
  - Primary Key: CustomerId (UUID)
  - Purpose: The central entity around which all other telecom activities revolve.
- **Accounts:** Represents individual service accounts held by a customer. A customer can have multiple accounts.
  - Primary Key: AccountId (UUID)
  - Foreign Key: CustomerId
  - Relationship: One-to-many from Customers to Accounts.
- **Plans:** Details various service plans (e.g., mobile, internet, TV packages) offered by the telecom provider.
  - Primary Key: PlanId (UUID)
  - Purpose: Defines service offerings and their attributes.
- **Usage:** Captures detailed usage records such as call duration, data consumption, and SMS counts.
  - Primary Key: UsageId (UUID)
  - Foreign Keys: AccountId, PlanId (if usage is plan-specific)
  - Relationship: Many-to-one with Accounts and Plans.
- **Billing:** Records billing cycles, invoice details, payments, and outstanding balances for each account.
  - Primary Key: BillingId (UUID)
  - Foreign Key: AccountId
  - Relationship: One-to-many from Accounts to Billing.
- **Devices:** Stores information about devices associated with accounts (e.g., mobile phones, routers).
  - Primary Key: DeviceId (UUID)
  - Foreign Key: AccountId
  - Relationship: One-to-many from Accounts to Devices.
- **Tickets:** Manages customer support tickets, issues, and their resolution status.
  - Primary Key: TicketId (UUID)
  - Foreign Keys: AccountId, CustomerId (optional, for broader customer-level issues)
  - Relationship: Many-to-one with Accounts and Customers.

Important Design Considerations

- **Interleaving for Performance:** A key optimization in Spanner is the use of table interleaving. For instance, the Accounts table is interleaved within the Customers table. This co-locates rows for a customer and their associated accounts on the same servers or splits, significantly improving read performance for queries that retrieve all accounts for a given customer, as it reduces network hops and distributes data efficiently.

- **Primary Key Choices:** UUIDs (Universally Unique Identifiers) are utilized as primary keys for most tables (e.g., CustomerId, AccountId). This strategy ensures even distribution of data across Spanner's distributed architecture, preventing hotspotting that can occur with monotonically increasing keys. This design supports Spanner's ability to scale horizontally and globally without performance bottlenecks related to key ranges.

- **Transaction Management:** Spanner's strong consistency guarantees simplify transactional logic. The normalized schema leverages this by allowing multi-table transactions (e.g., updating an account and its associated billing records) with ACID properties, ensuring data integrity even in a highly distributed environment.

- **Scalability Considerations:** The normalized schema, combined with careful primary key selection and strategic interleaving, inherently supports Spanner's horizontal scaling capabilities. This ensures the database can seamlessly handle increased data volumes and transaction rates as the telecom business grows, without requiring significant architectural changes.

Ingestion Pipeline (Pub/Sub to Spanner)

The initial phase of the data pipeline focuses on ingesting raw telecom JSON messages and transforming them into a structured format suitable for transactional storage in Google Cloud Spanner. This critical component is implemented as an Apache Beam pipeline, deployed and managed efficiently on Google Cloud Dataflow. It serves as the primary data processing engine, ensuring that high-volume, semi-structured data is accurately parsed, validated, and normalized before being committed to the operational database.

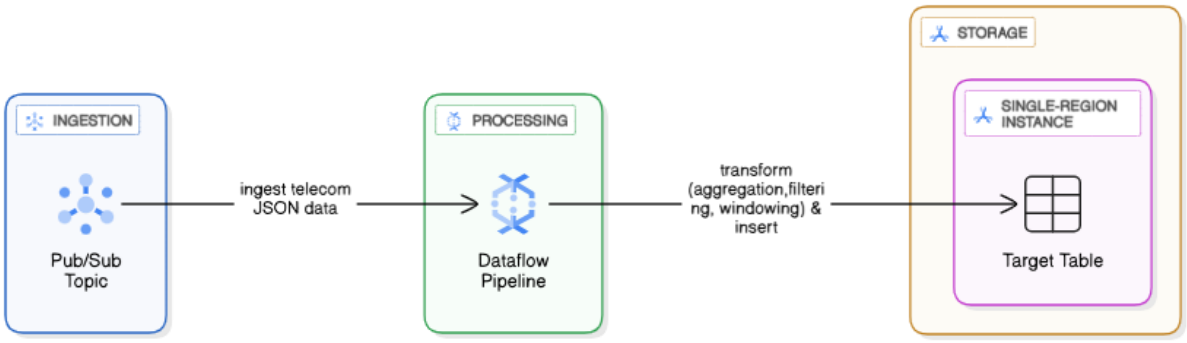


Fig 3. Data Ingestion Pipeline

Pub/Sub Subscription and Message Consumption

The Dataflow pipeline begins by consuming raw telecom JSON messages from a dedicated Google Cloud Pub/Sub topic via a subscription. Pub/Sub's at-least-once delivery guarantee ensures that no messages are lost, even during transient network issues or processing failures. The Apache Beam Pub/Sub IO connector is leveraged to establish a reliable stream of data. Dataflow's managed service handles message acknowledgment automatically; once a bundle of messages is successfully processed by the pipeline, Dataflow sends acknowledgments to Pub/Sub, allowing the messages to be purged from the subscription. In cases of processing failures, messages are not acknowledged and are re-delivered, supporting robust retry mechanisms.

Data Parsing and Validation

Upon receiving raw JSON messages, the pipeline performs rigorous data parsing and validation. Given the complex and often nested nature of telecom JSON data, robust parsing logic is implemented using Apache Beam's transformation capabilities (e.g., MapElements, FlatMapElements). Each JSON string is parsed into a structured object,

allowing individual fields to be extracted. Validation rules are applied to ensure data quality and integrity, such as checking for required fields, validating data types, and ensuring referential integrity within the JSON structure. For malformed or invalid JSON messages that cannot be successfully parsed or validated, a dead-letter queue mechanism is employed. These problematic messages are routed to a separate Pub/Sub topic or a Google Cloud Storage bucket for further investigation and manual remediation, preventing data loss and ensuring the main pipeline continues processing valid data without interruption.

Data Transformation and Normalization

Following successful parsing and validation, the data undergoes extensive transformation and normalization to conform to the predefined Cloud Spanner schema. This involves several key steps:

- **Data Type Conversions:** Converting string representations of numbers, booleans, and timestamps into their appropriate Spanner data types.
- **Data Enrichment:** Adding derived fields, such as timestamps for ingestion or processing, or generating unique identifiers (UUIDs) for new records.
- **Normalization Logic:** Breaking down the potentially monolithic raw JSON record into multiple entities that map to the normalized tables in Spanner (e.g., extracting customer details for the Customers table, account details for the Accounts table, and usage events for the Usage table).
- **Business Rule Application:** Implementing specific business rules, aggregations, or conditional logic necessary to prepare the data for transactional storage.

Writing to Cloud Spanner

The final step involves writing the transformed and normalized data into the appropriate Cloud Spanner tables. The pipeline constructs Spanner mutations (INSERT or UPDATE operations) for each record, ensuring that data is consistently committed across multiple normalized tables (e.g., Customers, Accounts, Usage, Billing, etc.). To achieve high throughput and efficiency, the pipeline leverages Dataflow's inherent ability to process data in parallel and utilizes batching for Spanner writes. Mutations are grouped into batches, significantly reducing the number of round trips to the database. Error handling for Spanner writes is critical; the pipeline is designed to retry transient errors and log or route permanent failures to a separate error queue for asynchronous handling, ensuring data durability and integrity within the transactional layer.

accenture

Change Data Capture with Spanner Change Streams

Change Data Capture (CDC) is a crucial pattern for modern data architectures, enabling the real-time synchronization of data modifications from transactional systems to analytical platforms, data warehouses, or other downstream applications. It is fundamental for use cases such as real-time analytics, operational dashboards, data replication, auditing, and maintaining up-to-date caches. Google Cloud Spanner

Change Streams provide a robust, low-latency, and highly scalable mechanism for implementing CDC directly from the operational database, ensuring data consistency and freshness across the entire data ecosystem.

Spanner Change Streams deliver a continuous, ordered stream of all data modifications (INSERTs, UPDATEs, DELETEs) made to designated tables. This built-in capability eliminates the need for complex, often fragile, application-level CDC implementations or log scraping, offering a highly reliable and performant solution that integrates seamlessly with Spanner's distributed architecture.

Change Stream Configuration

To enable Change Data Capture, Spanner Change Streams were explicitly configured on all relevant operational tables. This configuration ensures that every Data Manipulation Language (DML) operation—whether an INSERT of a new customer record, an UPDATE to a billing account, or a DELETE of a device entry—is captured and emitted as a change record. For this pipeline, change streams were enabled across key tables including Customers, Accounts, Plans, Usage, Billing, Devices, and Tickets. This comprehensive capture allows for a complete historical record of data modifications and supports full synchronization with analytical environments.

The configuration typically involves a simple DDL statement, specifying which tables and columns to track, along with whether to include old and new values for updates.

Mod JSON Parsing and Structured Row Generation

Each record read from a Spanner Change Stream is presented as a "Mod JSON" object. This JSON format contains rich metadata and the actual data modifications, providing a comprehensive view of the change event. The Apache Beam pipeline performs critical parsing and transformation steps to convert this Mod JSON into structured rows:

- **Metadata Extraction:** The pipeline extracts essential metadata embedded within the Mod JSON, including:

  - _table_name: The name of the Spanner table where the change occurred (e.g., 'Customers', 'Billing').
  - _commit_timestamp: The exact timestamp when the transaction was committed in Spanner.
  - _mod_type: Indicates the type of DML operation ('INSERT', 'UPDATE', 'DELETE').

- **Structured Row Transformation:** Finally, the extracted metadata is combined and transformed into a flat, structured row format. Each Mod JSON record typically translates into one or more rows, with distinct columns for the change type, commit timestamp, table name, and the individual columns of the modified record. pThis structured output is then ready for ingestion into staging tables in BigQuery, enabling efficient querying and downstream analytical processes.

Change Stream Reader Pipeline

A dedicated Apache Beam pipeline, also deployed on Google Cloud Dataflow, functions as the Spanner Change Stream reader. This second pipeline is specifically designed to continuously consume the stream of data modifications emitted by Spanner Change Streams in near real-time. It leverages the Apache Beam Spanner Change Stream connector, which efficiently handles the complexities of reading from Spanner's distributed change stream system, managing checkpoints and ensuring exactly-once processing semantics.

This pipeline is always-on, constantly monitoring for new change events. Its primary role is to process these raw change records and transform them into a structured, queryable format suitable for analytical consumption, primarily within Google BigQuery.

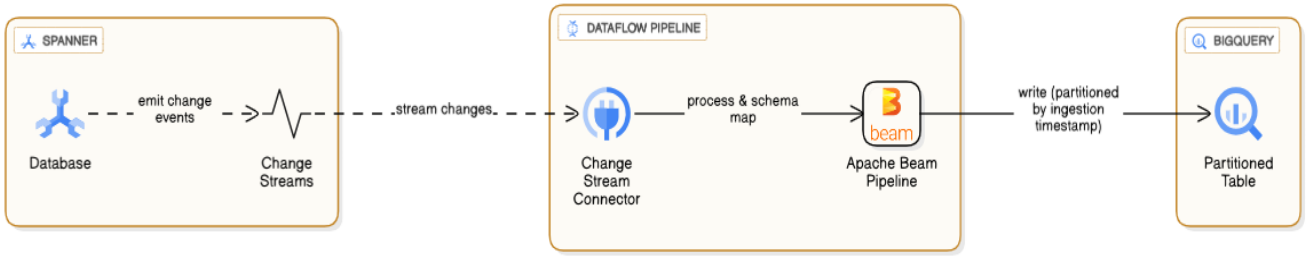


Fig 4. Change Stream(CDC) Pipeline

accenture

BigQuery Integration

The final stage of the telecom data pipeline is the integration with Google BigQuery, serving as the enterprise-grade analytical data warehouse. BigQuery was chosen for its unparalleled scalability, serverless architecture, columnar storage, powerful SQL interface, and cost-effectiveness when dealing with large-scale datasets. It provides a robust platform for business intelligence, reporting, and advanced analytics, allowing organizations to derive deep insights from their operational data without managing underlying infrastructure.

Staging Data in BigQuery

The Apache Beam pipeline dynamically ingests structured change data from **Spanner Change Streams** and writes it directly into **individual BigQuery staging tables**. For each operational Spanner table being tracked (e.g., Customers, Accounts, Billing), a corresponding staging table in BigQuery is created (e.g., telecom_dataset.customers_changes, telecom_dataset.billing_changes).

This is accomplished using **Beam's dynamic destination feature**, which routes each change record to the appropriate BigQuery table based on the originating Spanner table name. Each row includes metadata fields like:

- **record_timestamp –** when the change was committed in Spanner,
- **mod_type –** the type of change (INSERT, UPDATE, or DELETE),
- And the structured row-level data extracted from the change.

This approach provides several benefits:

- **Intermediate Storage:** Acts as a buffer before integration into final analytical tables.
- **Auditing and Replay:** Provides a complete, immutable audit log of all changes occurring in the operational Spanner database, invaluable for debugging, data lineage, or replaying historical states.
- **Schema Flexibility:** Allows for schema evolution by storing the raw change records before they are transformed into a more denormalized analytical format.

Enabling Business Intelligence and Analytics

The culmination of this pipeline is the rich, structured dataset residing in BigQuery, which serves as the foundation for diverse analytical applications. BigQuery's columnar storage and petabyte-scale processing capabilities enable:

- **Complex Ad-Hoc Queries:** Data analysts can execute sophisticated SQL queries on large volumes of telecom data to identify trends, patterns, and anomalies.
- **Dashboarding and Reporting:** Seamless integration with Business Intelligence tools like Looker Studio or Looker allows for the creation of interactive dashboards and automated reports on key performance indicators (KPIs) such as customer churn rates, service usage, billing accuracy, and network performance.
- **Machine Learning Applications:** The clean, structured data in BigQuery is ideal for training machine learning models for predictive analytics, such as churn prediction, personalized service recommendations, or fraud detection. BigQuery ML further simplifies this by allowing models to be built directly within BigQuery using SQL.

This integration transforms raw telecom data into actionable intelligence, empowering stakeholders to make data-driven decisions that enhance operational efficiency and drive business growth.

Results and Future Scope

This project successfully delivers a robust, scalable, and real-time telecom data pipeline on Google Cloud Platform, fully meeting its initial objectives. The end-to-end solution demonstrates significant technical capabilities in managing high-volume, semi-structured data from ingestion to advanced analytical insights.

Key Achievements and Impact

The implemented pipeline successfully:

- **Achieved High-Volume Ingestion:** Reliably ingested large volumes of diverse telecom JSON data via Google Cloud Pub/Sub, ensuring data durability and low latency.
- **Ensured Data Integrity:** Implemented robust parsing and normalization with Apache Beam on Dataflow, storing structured data in Google Cloud Spanner with strong consistency.
- **Enabled Real-time CDC:** Leveraged Spanner Change Streams for near real-time capture and propagation of data modifications to BigQuery.
- **Facilitated Analytical Insights:** Integrated seamlessly with Google BigQuery, enabling powerful ad-hoc querying, reporting, and business intelligence.

This solution provides immediate business impact by enabling proactive operational monitoring, improving data quality for reporting, and laying the foundation for predictive

analytics. The inherent scalability of GCP services ensures the pipeline can accommodate increasing data volumes and velocity, future-proofing the infrastructure for evolving business needs.

Future Enhancements

Potential enhancements and extensions for the pipeline include:

- **Advanced Visualization:** Integrating with business intelligence tools like Looker for interactive dashboards and self-service analytics platforms.
- **Machine Learning Integration:** Exploring Google BigQuery ML for use cases such as customer churn prediction, anomaly detection in network usage, or personalized service recommendations.
- **Additional Data Sources:** Onboarding new data streams, such as IoT device telemetry or customer interaction logs, to further enrich the analytical dataset.
- **Data Governance:** Implementing Google Cloud Data Catalog for enhanced metadata management and data discovery across the organization.

References

A list of official documentation and resources used during the development and deployment of the pipeline:

- Google Cloud Spanner Change Streams Documentation
- Apache Beam Programming Guide
- BigQuery IO Connector in Apache Beam
- Google Cloud Dataflow Documentation
- Google Cloud Pub/Sub Documentation
- Cloud Spanner: Schema and Table Design
- Cloud Spanner Client Libraries (Java)
- Gson Library for JSON Parsing
- Beam College 2025 Sessions
- Apache Beam Java SDK Documentation
- Beam IO Transforms - BigQueryIO
- Beam IO Transforms - SpannerIO and Change Streams

- Using DynamicDestinations in Apache Beam

Supplementary Learning

- Enterprise Data Management
- Understanding Cloud Spanner
- Hands on with Dataflow