Streaming Databases with Bigtable and Apache Beam

Christopher Crosbie

Group Product Manager

Databases @ Google Cloud

BEAM SUMMIT
NYC 2025

Real-time Data Cloud: Products

Real-time Data Movement	Real-time Data Warehouse	Real-time Processing	Real-time Analytics Database
Pub/Sub	BigQuery	Dataflow	Bigtable
Managed Kafka		Dataproc (Spark & Flink)	BigQuery
Datastream			

Agenda

- Why non-relational databases for streaming
  - Use Case: Feature stores
- Bigtable and Apache Beam Integrations

Choosing a Google Cloud Database Model for Apache Beam

Relational				Non-Relational			Multimodal
Oracle DB @ Google Cloud	Cloud SQL	AlloyDB	BigQuery	Bigtable	Firestore	Memorystore	Spanner

State-of-the-art gen AI and vector capabilities

Natural language in SQL | Vector Search | AI models in SQL

Vertex AI | MCP Toolbox for Databases | LangChain

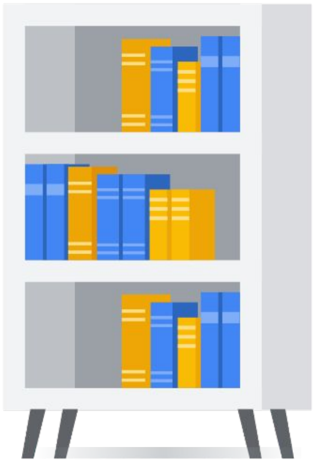
Ecosystem integration

Relational databases

- Store and provide access to data in tables that are joined by relationships
- Built-in mechanisms to ensure the consistency and integrity of your database structure

What’s wrong with this approach?

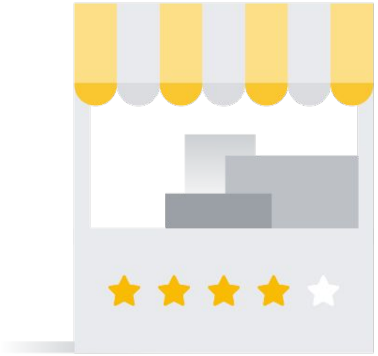
NOTHING.



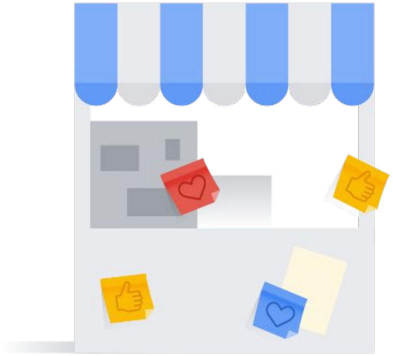
The right question to ask

What problem do you have?

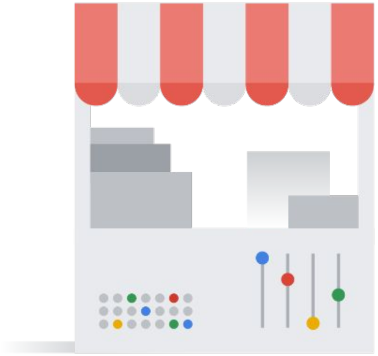
The problem with streaming pipelines



Customer reviews



Social media posts



Sensor readings

Non-relational databases

- Store data in a single table with keys and values, or in a document format such as JSON
- Ideal for data types that change frequently or for applications that handle diverse types of schema

Characteristics of a non-relational database

Near-unlimited scalability	High batch and streaming throughput	Flexible data-model	Superior price-performance
Need for high-availability and fault tolerance	Low latency for reads and writes	Denormalization of data for access	Flexible deployment topology

Bigtable

Low latency NoSQL database service for machine learning, operational analytics, and user-facing applications at scale

Fully managed key-value database

Schemaless, eventual consistency

High throughput

Millions of RPS, predictable single-digit ms latency

Flexible and open

Topologies from a single zone to 8 regions of your choice with SSD or HDD storage; HBase API compatible now with SQL support

Industry leading 99.999% SLA

Regional and multi-regional replicas

Bigtable has more than 10 exabytes of data under management and serves over 7 billion queries per second.

Example use cases

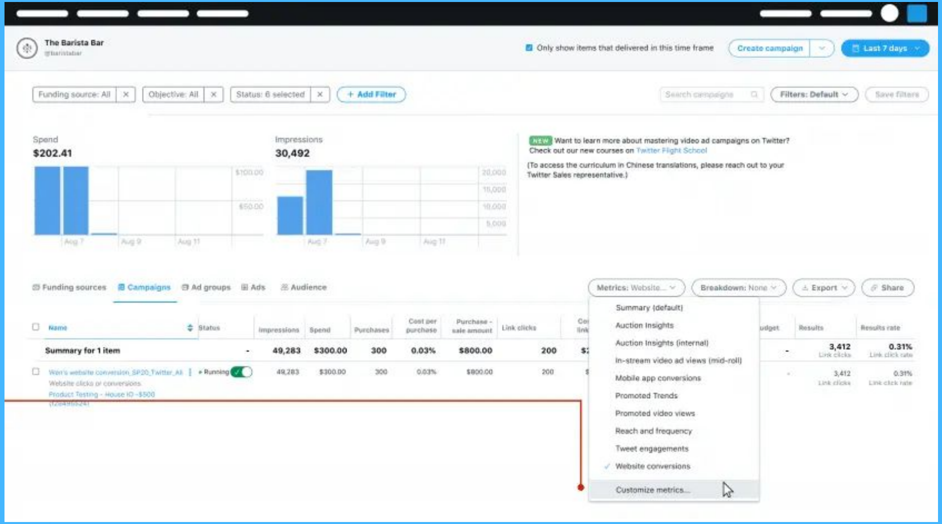
- Personalization
- Fraud detection
- IoT/machine data
- Customer/product metadata
- Data fabric

Bigtable and Apache Beam

Public Architectures of real-world applications

Advertiser Dashboards at X

- Allows advertisers to view their ad performance **in real time**
  - eg: Twitter aimed for a ~1 minute average lag between logged event and dashboard update
- ~1s latency target (for the entire dashboard)
  - "User interactive"
- Should be correct, but slight inaccuracy can be tolerated as long as it's corrected eventually
  - eg: Spend shown must match amount billed "eventually"



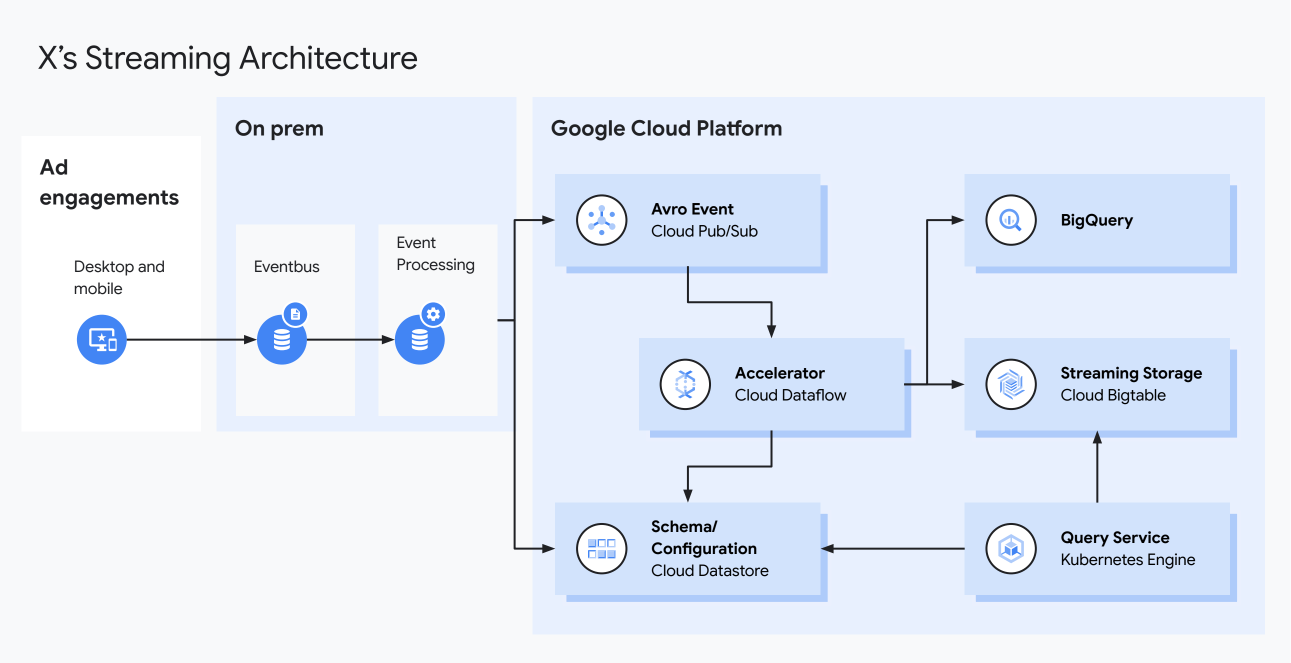
X: Streaming analytics at scale

~3 million
events / second

~2 million
mutations / sec to Bigtable

~1 billion
active aggregations / hour

1.5 GB/s
events / second



Problem statement

We needed an operational data store that would support low latency access for both real-time and historical data with a flexible schema

- Flexible schema for evolving telematics with new vehicles, new sensors, over the air (OTA) updates
- Real-time notifications and actionable insights to our customers
- Low latency, < 100ms (p99)
- Support for AI based insights and recommendations
- Privacy and Security and GDPR compliance
- Able to self-heal based on data trends and anomalies

Journey to Bigtable

MongoDB

We found it highly varying document size.
Heavy use of memory and not cost effective.

BigQuery

Serverless and cost-effective data warehouse.
Would not support low latency for real-time and historical data for serving API requirements

Postgres

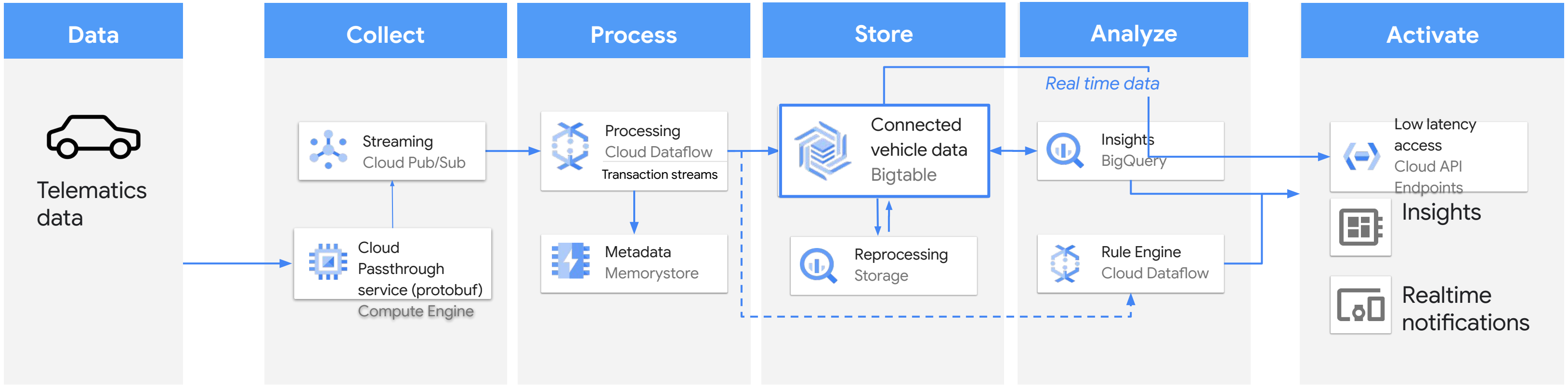
Would not support a flexible schema, or the scale we need.

Memorystore + BigQuery

Would not scale to the data volume required.

Platform architecture

Bigtable is at the heart of our platform providing high performance data capture and low latency real time insights for our customers



1 billion + messages per day
600k writes per second with bulk data upload

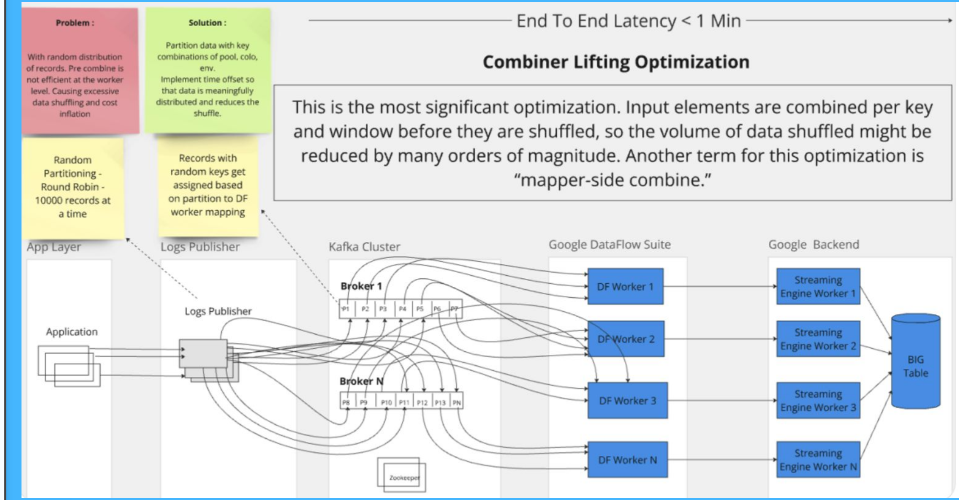
Daily average
75,000 writes per second
22,000 reads per second

Log Processing at PayPal

- Observablity team is responsible for providing a telemetry platform that produces three petabytes of logs per day
- Migrated from self-managed Apache Flink to Dataflow

Implementing a high-throughput, low-latency streaming platform is critical to providing high cardinality analytics to business, developers and our command center teams. The dataflow integration has now empowered our engineering teams with a strong platform to monitor paypal.com 24 x 7 thereby ensuring PayPal is highly available for our consumers and merchants.

Varun Raju, Architect, Observability Platform, PayPal



3 Petabytes of logs per day

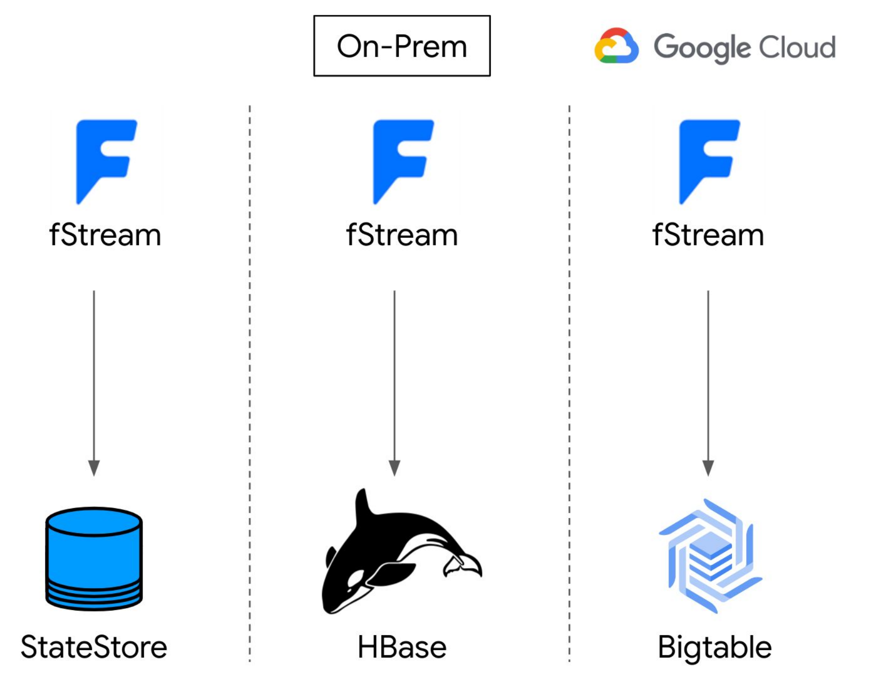
FStream at FlipKart

Indian e-commerce company, Flipkart

- 450 million users
- 1.4 million sellers
- 150 million products
- Millions of shipments daily

fStream, is an in-house common streaming platform and state store. fStream operates seamlessly on Apache Spark and Apache Flink using Dataproc.

Moving from HBase to Bigtable made it simple to scale up the platform 4x for their Billion Day event and reduced replication lag and maintenance overhead



AI/ML and real-time are driving non-relational patterns

With a non-relational system, customers can delay decisions about how their data will be consumed

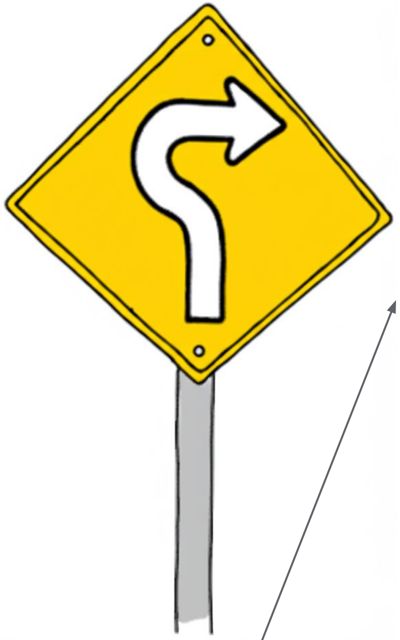
"The process of data modeling for a relational system is extremely time-consuming. While a relational system offers very good performance for specific types of queries, data preparation is too labor-intensive for **frequent changes** to be practical and too expensive and difficult to be scalable. . ."

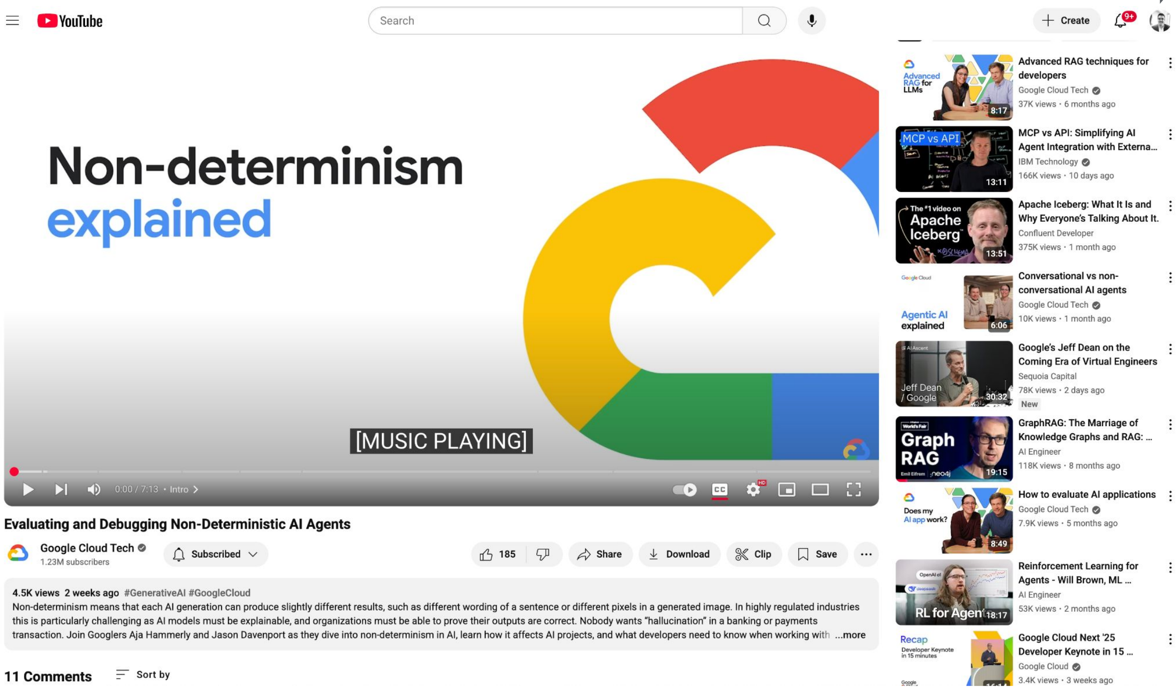
- **Ted Dunning & Ellen Friedman, *AI and Analytics at Scale: Lessons from Real-World Production Systems*, O'Reilly Publishing, 2021**

Example use cases

- Personalization
- Fraud detection
- Data fabric
- IoT/Machine data
- Customer/product metadata
- ML training

Case Study: Navigating Changing Requirements in a Feature Store





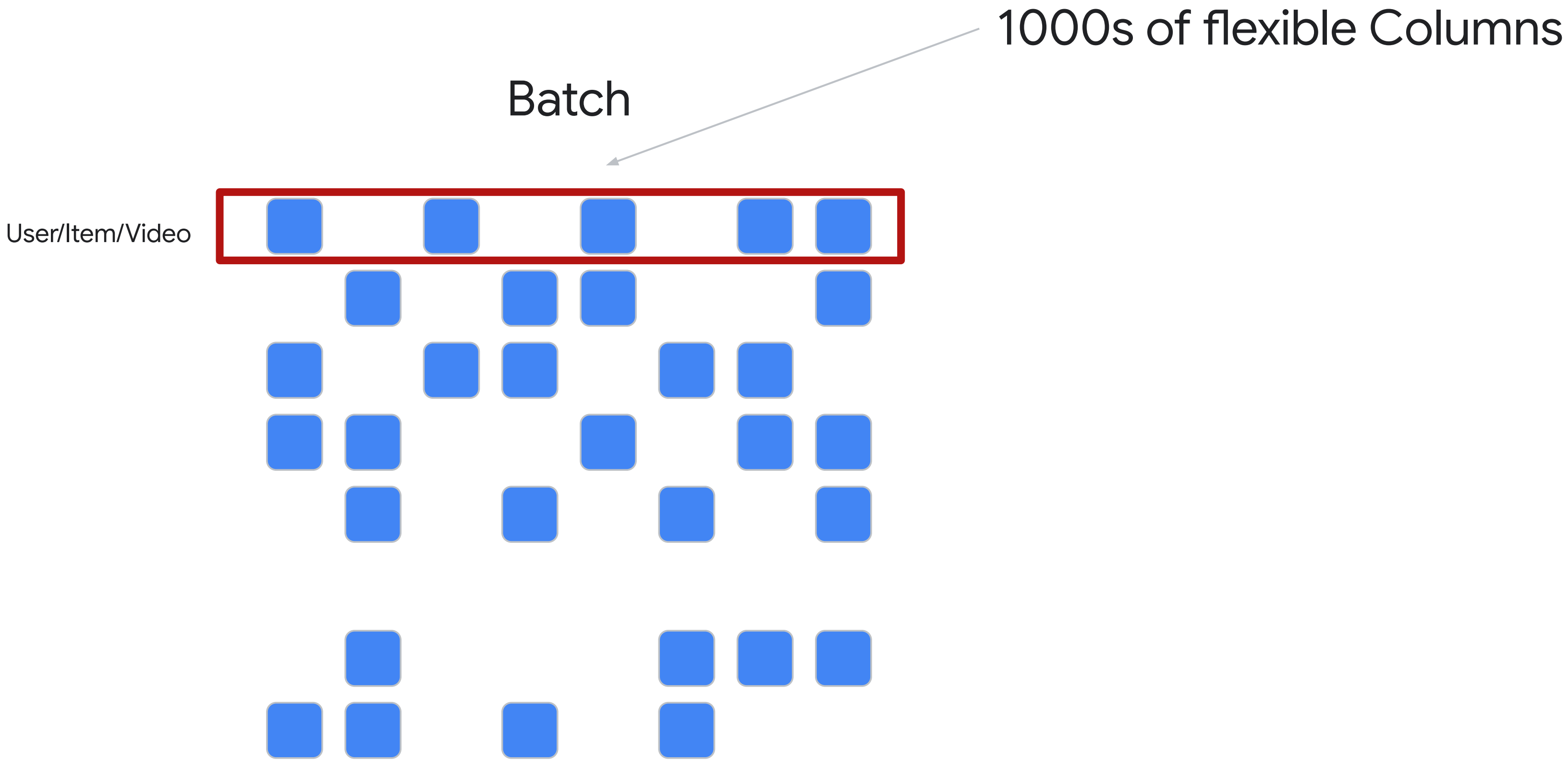
User
user_time_spent
User_video_plays
user_dob

Video
video_likes_10m
video_shares_30days

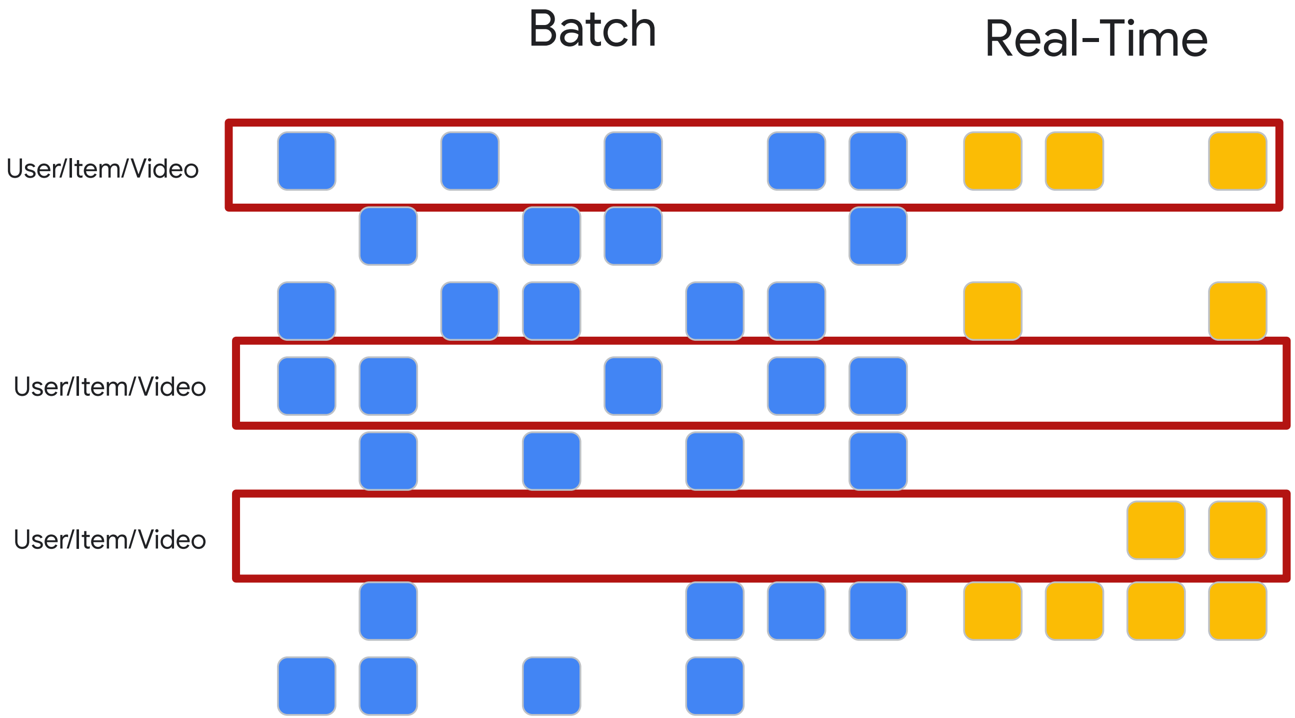
Comments
comment_count
comment_sentiment_neg_count

Recommendation
user_clicks_7day

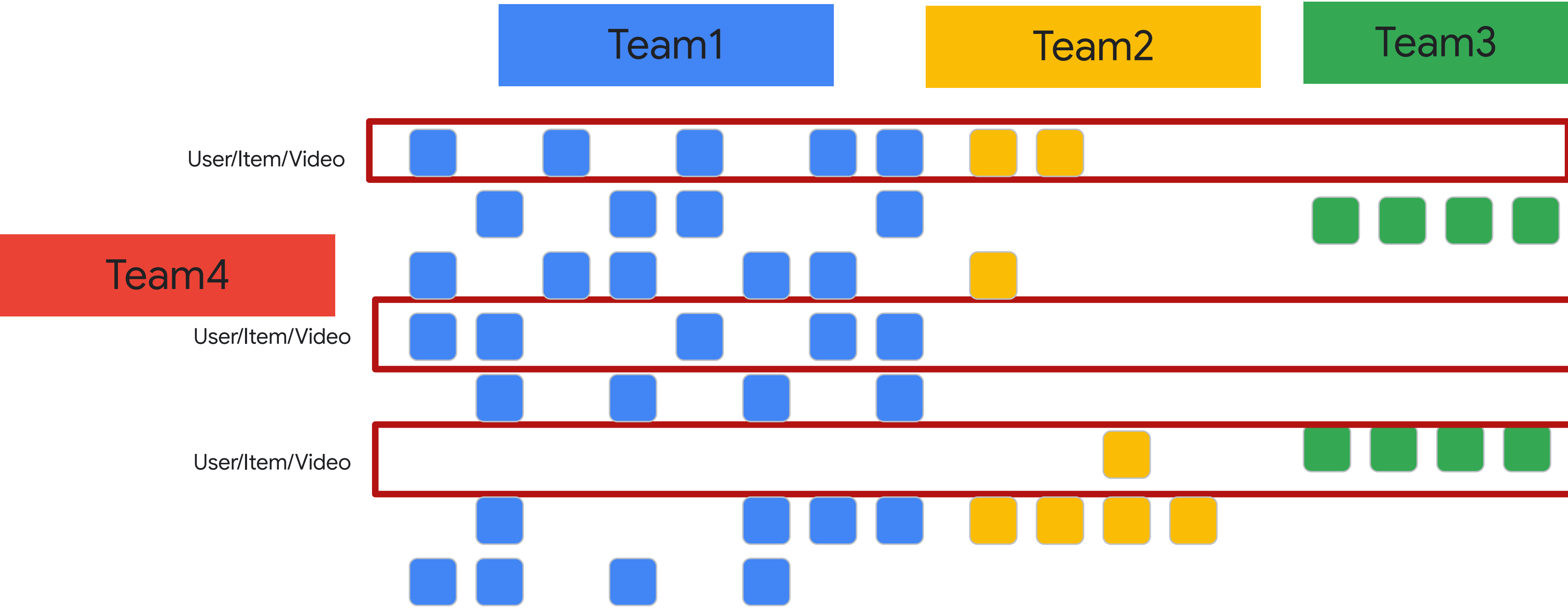
The case study of Feature Stores



The case study of Feature Stores
(new Real-Time Features, Needed)



The case study of Feature Stores
(Multiple Teams Collaborate on Features)



Feature Store Needs

Different Workloads, different teams, schema flexibility, high scalability

Online Mode

Focused on now

Short term retention with TTL

Real-time live statistics

Fast retrieval on lookups

Offline Mode

Historical data for training

Long term storage

Batch processing

Mix multiple sources

Data Boost: Unified batch & real-time processing

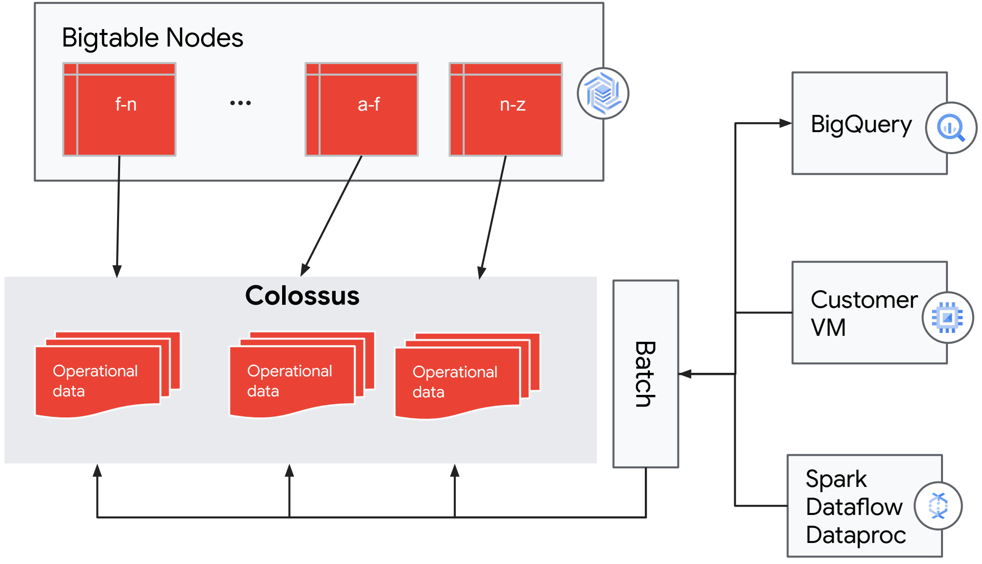
Traditional feature stores separate online and offline processing thus increasing cost and introducing skew

01 **No data movement or duplication**
Faster time-to-market and higher productivity for for ad-hoc queries, or ETL

02 **Workload isolation**
No performance impact on production serving

03 **Data sharing**
Share data among teams without worrying about impacting serving performance

04 **Unified Feature Store**
Reduce training/serving skew and storage costs by training and serving over the same data



Bigtable Counters for Real-time ML features

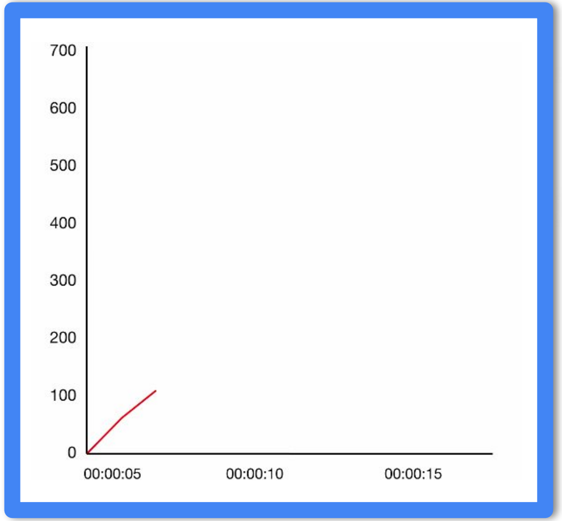
Traditional feature stores write the events into offline store and update features in batches resulting in outdated metrics

With Bigtable counters you get

- ***Instantly*** updated metrics with < 3 millisecond writes without complex Lambda/Kappa architectures
- Aggregations for sum/count, min/max, approximate count distinct
- Timestamps for hourly, daily, weekly etc. ***tumbling windows***
- Costs a **single write** to update multiple counters in a row
- Global scale without performance compromises

Ideal for calculating metrics such as

- First, last time an action is taken, total time spent by user
- Impression and unique counts for ads, promotions, content, product
- Hourly, daily, weekly, monthly dollars spent
- Number of unique credit cards or ip addresses by user



productXviewed_1day

hoursactive_lifetime

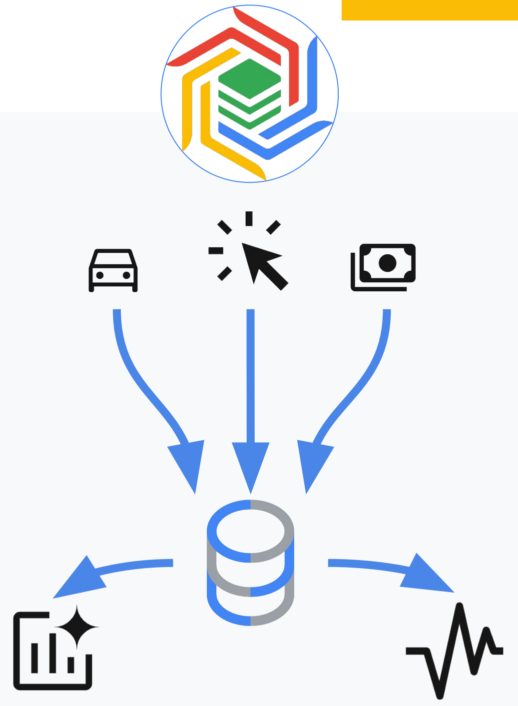
failedlogins_1hour

uniqueIPaddresses_15min

Bigtable Continuous Materialized Views

Preview

Turn data streams into immediate insights



SQL aggregations on dynamic data
Generate aggregations and new groupings on flexible and changing schema directly in the database

Globally scalable
Distributed and synchronized metrics at global scale with seamless regional and global replication, with support for high fanout writes that converge

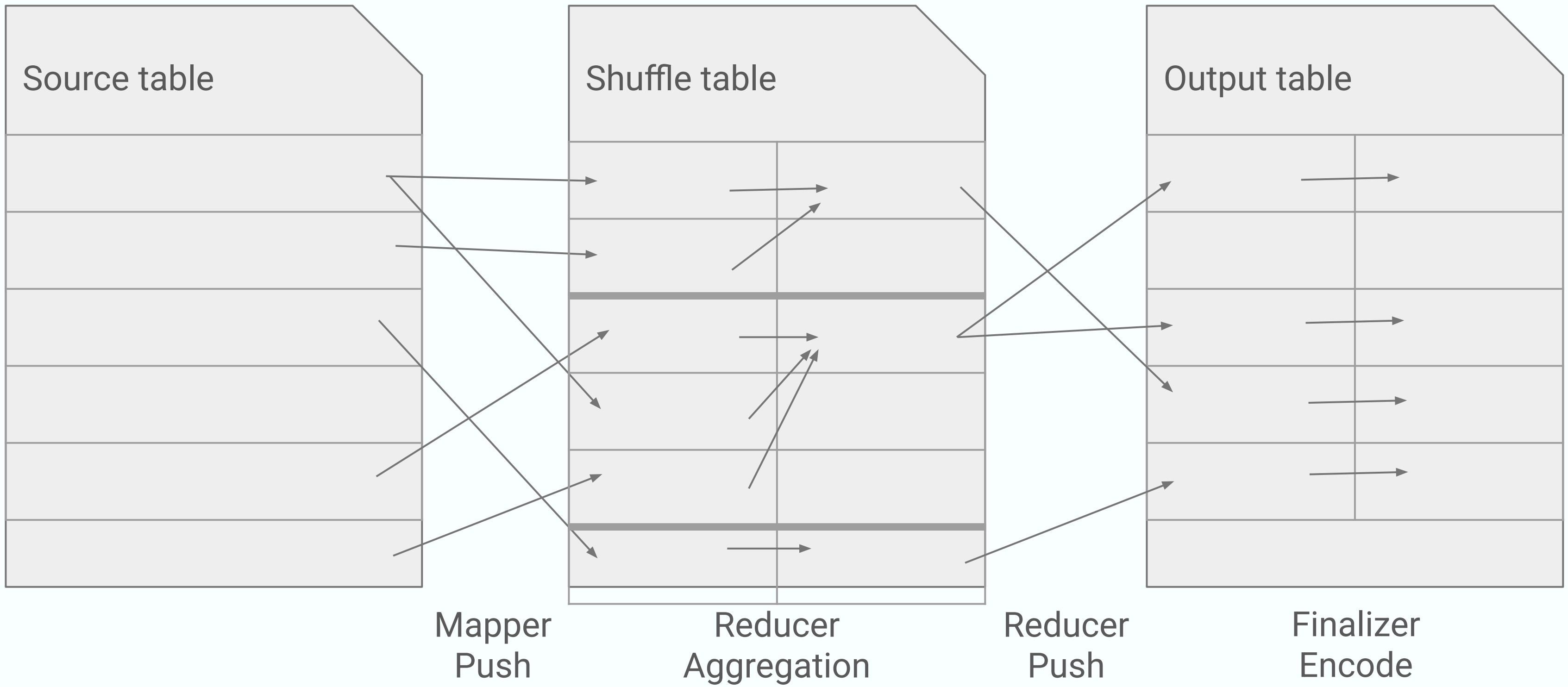
Fully managed data pipelines for fresh data
Simplified administration, data is processed (typically in seconds) without impacting application queries

Automatically generate datasets for known query patterns
Pre-aggregate data for real-time dashboards, re-key data by groups for alternative query patterns, rollup time series data, extend session windows

Example use cases

- Online feature store metrics to detect fraud and anomalies
- Interactive analytics for real-time tracking
- Gaming: real-time leaderboards and in-game content ranking
- Rollup tables across time for telemetry data

How does this work?



Bigtable as a feature store

Low latency, fully managed database with flexible topologies and SQL support for dynamic schema

Online Mode

High throughput

Millions of RPS, predictable single-digit ms latency

Industry leading 99.999% SLA

Regional and multi-regional replicas

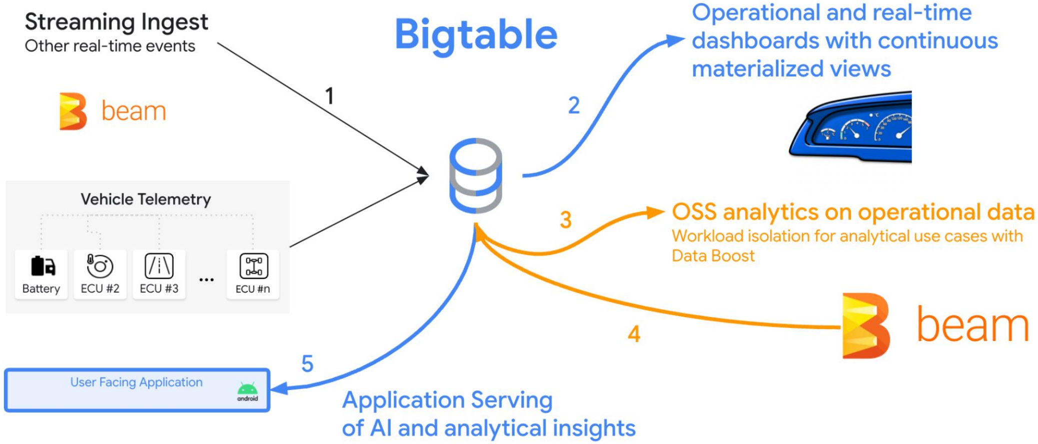
Offline Mode

Offline Analytics with Data Boost

Serverless compute for high-throughput batch jobs from Apache Beam and Dataflow

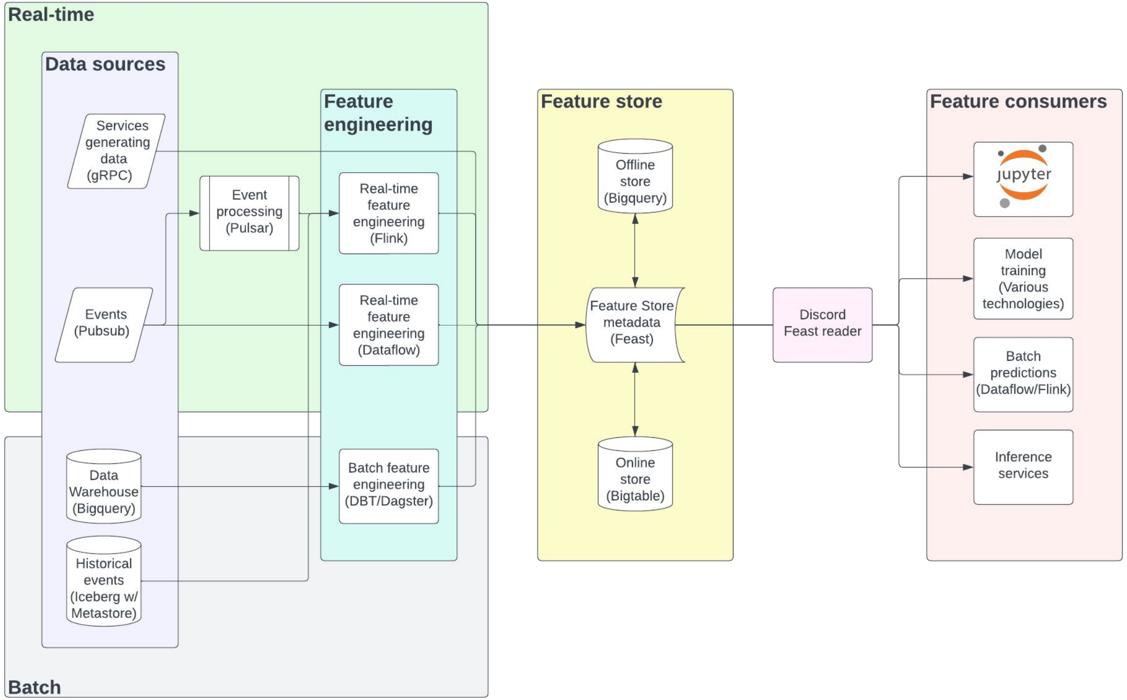
Tiered Storage

Automatically migrate data from "hot" (SSD) serving to "cold" (HDD) storage for long term retention



Building a feature store with open source flexibility

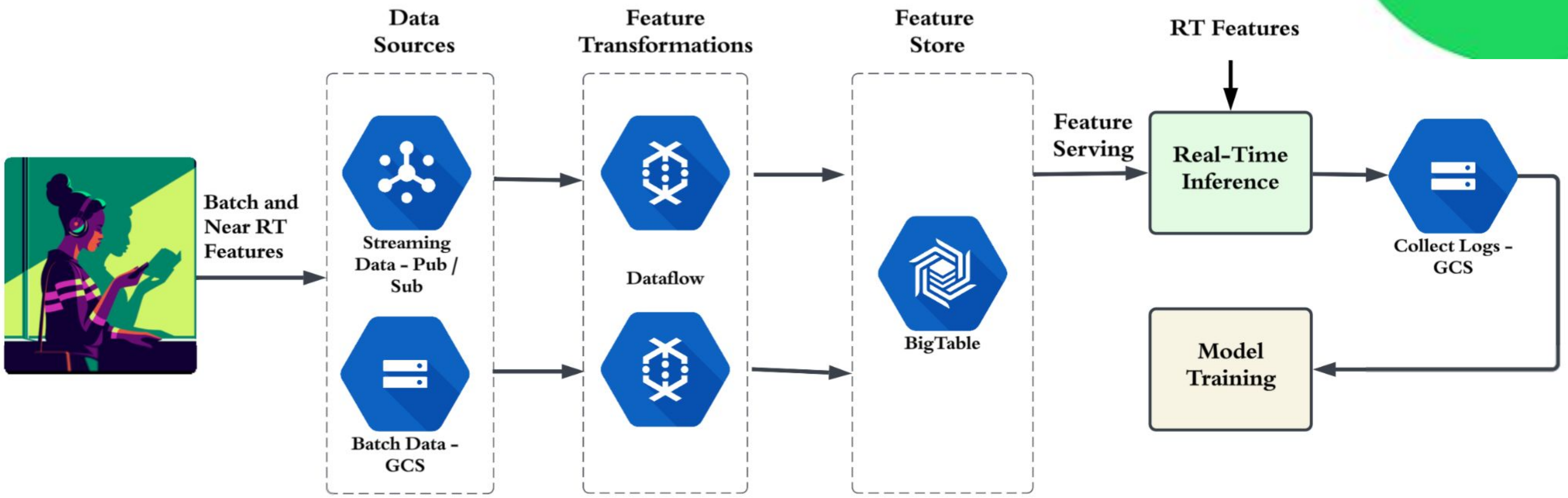
- **Discord** built a customized feature store that provides a one stop shop for ML features to streamline model development of social interactions across hundreds of millions of users
- **Feast** provides a unified development and deployment platform for data science and machine learning while letting you bring databases like Databricks and Snowflake.



Blog Series: Streamlining ML Development with Feast

Bigtable as Music Recommendation Feature Store

Learn more in [Bigtable and BigQuery in Spotify's music recommendation engine]



Bigtable and Apache Beam

Specialized capabilities for Apache Beam and Bigtable

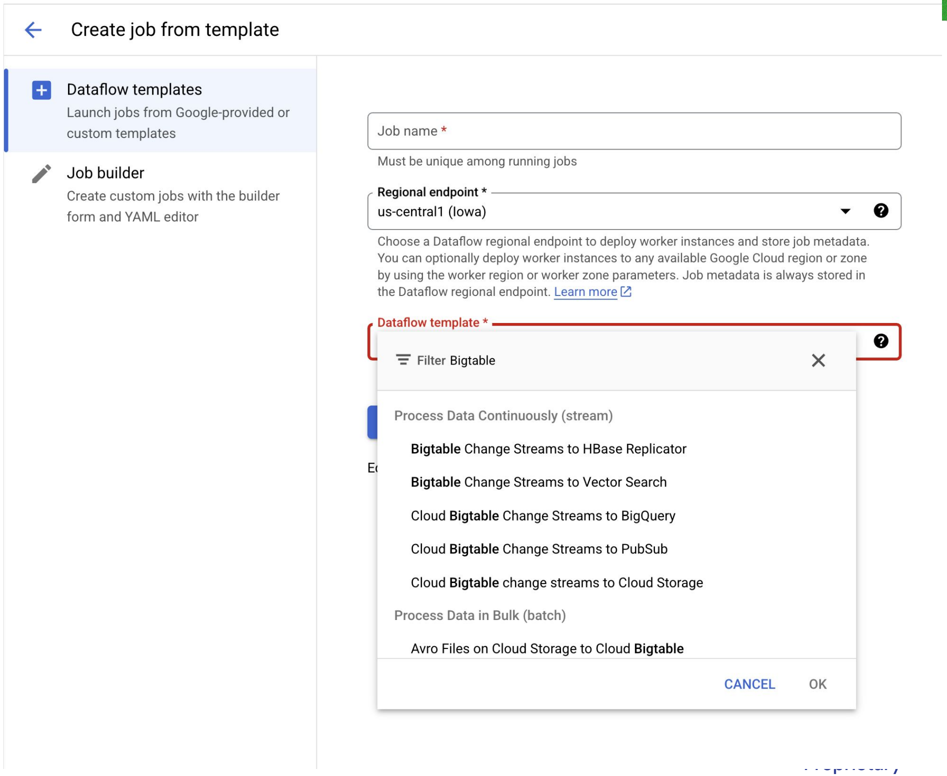
Easy to Use: Bigtable Templates

Dataflow templates for Bigtable

Streamline data pipelines and unlock faster insights with Bigtable Dataflow templates

- **Bigtable Change Streams (continuous):** HBase Replicator, Vector Search, BigQuery, PubSub, Cloud Storage
- **Bulk uploads to Bigtable (sink):** Avro on Cloud Storage, BigQuery, Cassandra, Parquet, SequenceFile
- **Bulk uploads FROM Bigtable (source):** Avro on Cloud Storage, JSON, Parquet on Cloud Storage, SequenceFiles on Cloud Storage, Vector Embeddings, Parquet Files, SequenceFile

Explore the Dataflow templates and start optimizing your Bigtable data pipelines.



Integrations: For streaming and batch processing

Apache Beam connector

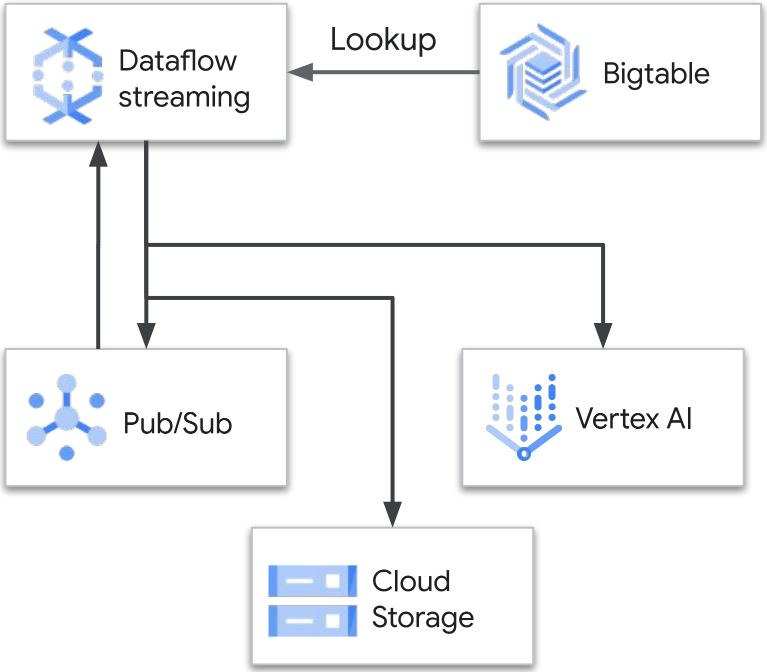
Enrich streaming data with fast key-value lookups with ease using **Apache beam.io.enrichment** package

Real-time recommendations

Join 'live' clickstream with the historic clickstream or user attributes.

Real-time fraud detection

Link purchase activity to past purchases and fraud indicators from Bigtable online feature store.



Enrichment

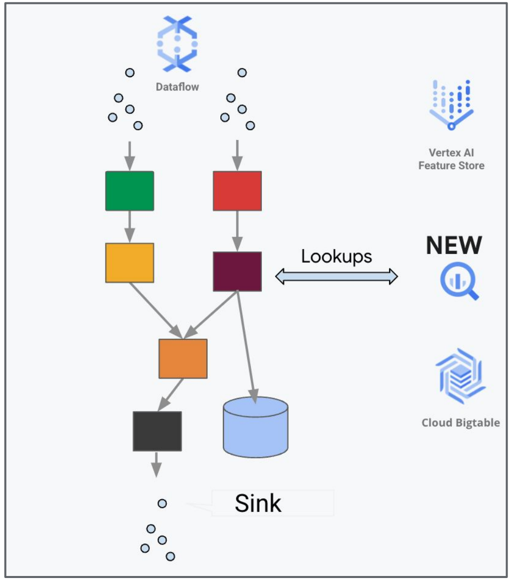
What

- Low code declarative turn key transforms for joining streaming data to data stores
- **Out of the box support for Bigtable**

```
output = (p
          ...
          | "Create" >> beam.Create(data)
          | "Enrich with Bigtable" >> Enrichment(bigtable_handler)
```

Benefits

- **Declarative:** Define the what, not the how
- **Rate limiting capabilities:** Automatic back-offs and autoscaler integration



Use cases for Enrichment

Real time recommendations
Requires the joining of users 'live' clickstream with the historic clickstream data.

Anomaly detection
Join the IOT telemetry data, with historic information

FSI Index building:
Join the current instrument ticks, with historic information metrics to build near real time indexes

Insightful, Intelligent, and Open: Built-in RAG Support

Accelerating ML developer productivity

Dataflow ML for RAG applications: Knowledge Ingestion and Real-time Streaming

What

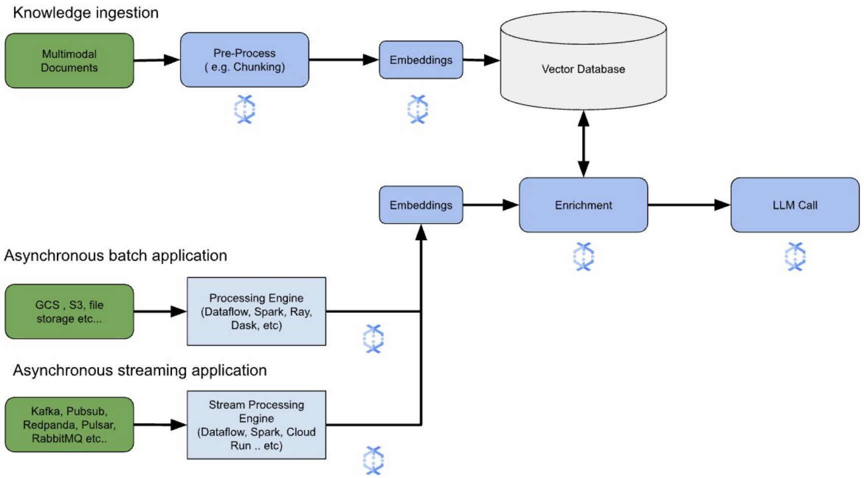
- Built-in transforms for creating embeddings (single LOC to integrate embedding generation in your pipeline)
- Choice of using Vertex AI or BYOM for embedding creation
- Built in support for writing embeddings to AlloyDB, Bigtable, Spanner and other Vector databases

Benefits

- Simplified preprocessing of content - choice of chunking methodologies combined with powerful data processing capabilities
- Leverage fast/light models (e.g Gemma) for local embedding generation
- Leverage same code for knowledge ingestion and real-time serving (async streaming)
- Eliminate code changes when moving from one vector database to another or changing ML models/systems

```
beam.ml.MLTransform(rag.embedding...)
```

Note: API Signature may change



QUESTIONS?

Email: BigChris@google.com