Google Cloud

Building the analytics lakehouse on Google Cloud

Rachel Levy, Steve Thill, Wissem Khlifi, Ravi Bhatt, Franklin Whaite, Ragi Mahil, Firat Tekiner

Table of Contents

Foreword to the reader

Organizations have been searching for optimal ways to store and analyze vast amounts of structured, unstructured, and semi-structured data that can handle the increasing volume, latency, resilience, and data access requirements demanded by cross-functional teams. Historically, data architectures like data warehouses and data lakes have been used to resolve these challenges. Data warehouses stored structured aggregate data used primarily for business intelligence (BI) and reporting. Data lakes stored unstructured and semi-structured data in large volumes, primarily used for machine learning (ML) workloads allowing organizations to run any computational framework or processing engines such as Spark. This siloed architecture approach, however, often resulted in extensive data movement, processing, and duplication, requiring complex extract, transform, load (ETL) pipelines requiring specific expertise on the team.

The lakehouse architecture, which combines the key benefits of data warehouses and data lakes, emerged to address some of these issues. It delivers:

- Decoupled storage and compute to empower organizations to choose the compute engine that best suits their specific use case or workload
- Accelerated ability to run advanced workloads, data science, and modeling
- Real-time and consistent data lake to reduce data tiers and movement by bringing parity across data lakes and warehouses (ACID)
- Low-cost storage in an open format accessible by processing engines like Spark, while also providing management and optimization features
- Eliminated the need to create and manage multiple data copies, moving speed to insight.

The standard lakehouse architecture, however, doesn't solve the issue of cross-cloud analytics, real-time processing and democratizing AI/ML. And building an additional data lakehouse platform only adds another siloed layer as there is no one-size-fits all solution.

Google Cloud is extending the benefits of the lakehouse architecture, into an end-to-end analytics lakehouse. The analytics lakehouse enables organizations to **extract data in real-time regardless of which cloud or datastore the data resides** and **use it in aggregate for greater insight and artificial intelligence (AI)**, with **governance and unified access** across teams.

Benefits of Google Cloud's analytics lakehouse include:

- Democratized environment for data analysts (SQL), data engineers (Spark/Beam) and data scientists (ML) with built-in and managed application stack without needing to move the data around.
- Serverless solutions and integrated open-source tools provide flexibility for data personas to bring their tool of choice for analyzing and processing data either in real-time or in batch.
- In-Database ML **with an inference engine** allowing ML models to run where the data is rather and integrate with MLOps pipelines to productionize ML workloads than moving the data
- Centralized cataloging and fine-grained security for data management and governance enabling organizations to unify datastores and associated metadata to simplify permissions and exploration

This paper outlines the technical processes of implementing an analytics lakehouse, leveraging Google Cloud's **open**, **unified**, and **intelligent** data platform. The paper covers each architectural layer and highlights how many stakeholders interact with the platform along with the benefits of the analytics lakehouse along the way.

Chapter 1

Introduction

For more than a decade the technology industry has been searching for optimal ways to store and analyze vast amounts of data that can handle the volume, latency, resilience, and varying data access requirements of organizations. To tackle these issues, companies have been making the best of existing technology stacks. This typically involves trying to either make a data lake behave like an interactive data warehouse or make a data warehouse act like a data lake — processing and storing vast amounts of semi-structured data. Both approaches have resulted in unhappy users, high costs, and data duplication across the enterprise. Google Cloud is uniquely positioned to help customers innovate faster with an open, unified and intelligent data platform. This data platform is the foundation for Google's analytics lakehouse which blurs the lines between traditional data warehouses and data lakes to provide customers the benefits of both.

Historically for analytics, organizations have implemented separate solutions for different data use cases: data warehouses for storing and analyzing structured aggregate data primarily used for business intelligence (BI) and reporting, and data lakes for unstructured and semi-structured data, in large volumes, primarily used for machine learning (ML) workloads. This approach often resulted in extensive data movement, processing, and duplication requiring complex extract, transform, load (ETL) pipelines. Operationalizing and governing this architecture was challenging, costly, and reduced agility. As organizations are moving to the cloud, they want to break these silos.

To address these issues, a new choice has emerged: the analytics lakehouse, which combines key benefits of data lakes and data warehouses, without all the overhead of each. This approach offers low-cost storage in an open format that is accessible by a variety of processing engines like Spark and structured query language (SQL) engines, while also providing powerful management and optimization features. The unification of these systems allows powerful governance that enables more productive use of data across an organization.

At Google Cloud we believe in providing choice to our customers — the option of an open platform that minimizes dependencies on a specific vendor or file format. Organizations that want to build their analytics lakehouse using only open-source technologies can easily do so by using low-cost object storage provided by Google Cloud Storage or across other clouds — storing data in open formats like Parquet, Iceberg, and Delta. Processing engines and frameworks like Spark and Hadoop use these (and many other) file types and can be run on Dataproc or regular virtual machines (VMs) to enable transactions. This open-source-based solution has the benefits of portability, community support, and flexibility, though it requires extra effort for configuration, tuning, and scaling. Alternatively, Dataproc is a managed version of the Hadoop ecosystem, minimizing the management overhead of these systems while still benefiting from non-proprietary data types.

Google Cloud also offers cloud-native tools to build an analytics lakehouse with the cost and performance benefits of the Cloud. These include a few key pieces that we will discuss throughout this paper:

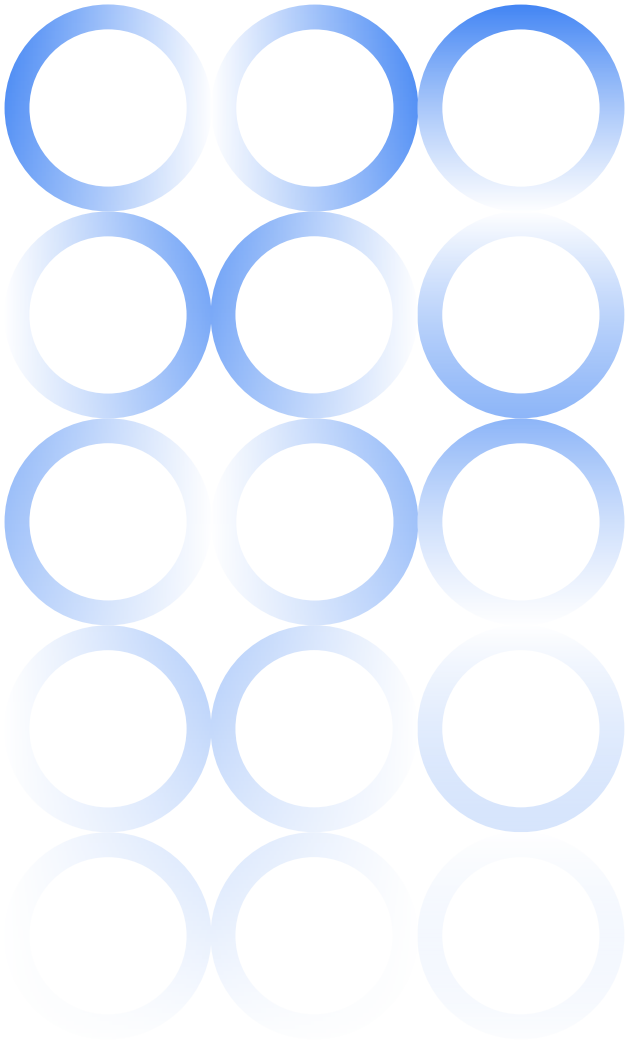
- Different storage options and optimizations depending on the data sources and end users consuming the data
- Several serverless (and stateful) compute engines to balance the benefits of speed and costs as required by each use case for processing and analytics
- Democratized and self-service BI and ML tools that can be used to maximize the value of the data stored in the lakehouse
- Powerful governance to ensure productive and accountable use of data in a way that ensures bureaucracy does not inhibit innovation and enablement

Underpinning these important pieces are thoughtful and rigorous automation and orchestration of repeatable tasks. This helps ensure that as data moves through the system, its timeliness and accuracy allow end users to trust it, making them more likely to both interact with and evangelize the analytics lakehouse. As more users start to adopt an analytics lakehouse, a concern for many organizations will be cost. Google's data cloud can intelligently manage capacity and optimize billing so you don't pay for more than what you use. This includes dynamic autoscaling, in combination with right-fitting, saves up to 50% in infrastructure for streaming, open source autoscaling capabilities across regional deployments, and granular billing options to only pay job duration instead of infrastructure time.

As the data landscape continues to grow and evolve at an exponential rate, it is important to have flexible patterns and limitless scale to ensure that value can be extracted from the data cost effectively.
The lakehouse enables this thoughtfulness because it's flexible and dynamic, much like the data and use cases we are seeing across organizations today.

In the ever-evolving world of data architectures and ecosystems, there is a growing suite of tools being offered to enable data management, governance, scalability, and machine learning.
With promises of digital transformation and evolution, organizations often find themselves with sophisticated solutions that have a considerable number of bolt-on features. However, the goal should be to streamline use of use by simplifying the underlying infrastructure, and thus enabling teams to focus on bringing value
to the business. This means that data engineers can focus on evolving data assets and exposing them to be useful for analysts and data scientists, while the data scientists focus on building and operationalizing models, not sourcing their own data. Analytics lakehouses provide end users with a job-specific set of flexible tools so they are no longer constrained by technical capabilities or technical infrastructure.

This paper will focus on the core technical foundations and capabilities of building an analytics lakehouse on Google Cloud. It will first unpack the definition of an analytics lakehouse and detail the requirements for how to make it useful, focusing on the flexibility that comes with separating storage and compute and the complexity of these decisions. Then the paper will cover these pieces using Google Cloud-native tools and governance recommendations. Finally, the paper will present an example use case of how these all work in a real-world example.



Chapter 2

Analytics lakehouse conceptual architecture

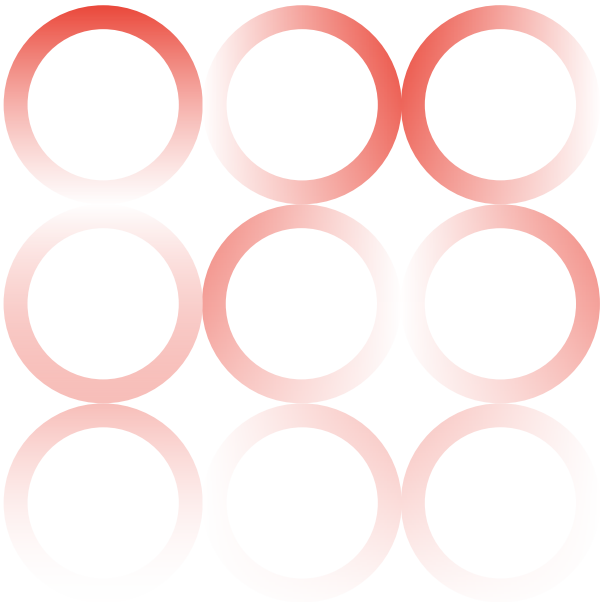
The analytics lakehouse combines the scalability and diverse processing of a data lake with the structure and performance SQL of a data warehouse. This is enabled by the separation of storage and compute, allowing them to scale separately and meet the needs of different types of end users. This allows users to bring the processing they want to the data they need. Historically, data lakes provided limitless compute and storage to process petabytes of data stored in proprietary and open file formats. Data lakes add support for unstructured data and streaming use cases, while often being best suited for data science and ML use cases. On the other hand, data warehouses provide a structured and cleansed view of enterprise data that is built on a foundation of data governance enforcement. The features of a traditional data warehouse include data quality, granular data access control and schema on write, SQL interactivity, time travel, ACID compliance, and data manipulation language (DML). Historically, data warehouses have been best suited for business intelligence and ad hoc SQL interaction use cases.

The analytics lakehouse is an evolved architecture that combines the best of both worlds, supporting both business intelligence and AI/ML use cases, thus breaking data silos. It also enables data engineers, data scientists, and other data users to share data, collaborate more effectively, and deliver more business value faster. Also, most of today's data lake setups require copying from data lakes to data

warehouses to enable BI and reporting use cases. This copying of data creates disparate data puddles which results in slower data delivery, increased variability, and decreased confidence in the data as results from the data lake may not match results from the data warehouse.

Google Cloud's analytics lakehouse replaces the need for separate data lakes and data warehouses and enables open and direct access to data stored in standard data formats, while enabling scale for data science workloads and low latency and reliability for BI workloads.

Google's analytics lakehouse is not a completely new product, but is built on Google's proven services such as Google Cloud Storage, BigQuery, Dataproc, Looker, and others. Our analytics lakehouse logical architecture, described below, shows how the analytics lakehouse is fed by the data ingestion layer, how data is stored through the storage layer, and how it is processed and activated in the data consumption layers. Management of the lakehouse occurs through a set of services that provide governance, observability, and workflow orchestration.



Analytics lakehouse components

Unified team access
Across all data practitioners and line of business owners (LOBs)

Extract and work with data in real-time
Regardless of which cloud or datastore the data resides

Unified data across clouds
Regardless of format and location of the data

Trusted data quality
Consistent fine-grained governance, both federated and domain-specific

Access full fidelity data
To use in aggregate for accurate insight and AI/ML modelling

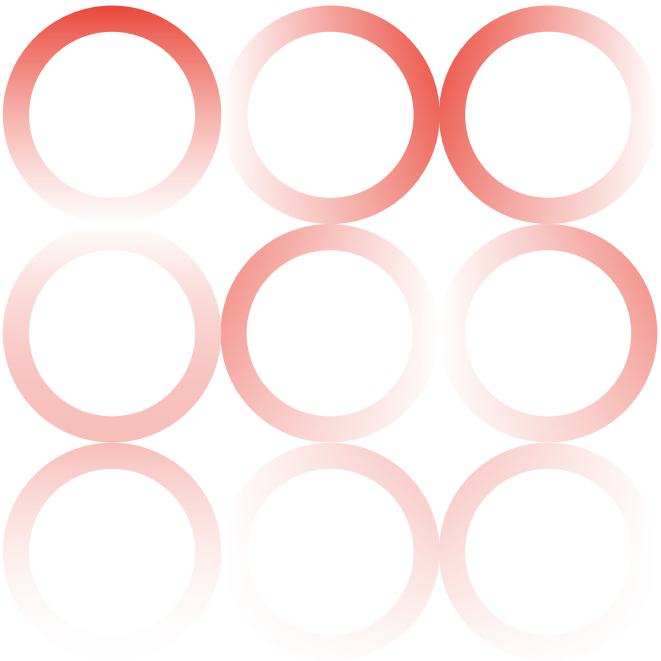
Figure 1: Analytics Lakehouse components

Analytics lakehouse architecture layers

The lakehouse architecture diagram (See Figure 2.) can be decomposed into various layers for ingestion, processing, storage, consumption, and governance. The goal of each layer is as follows:

Data ingestion

Bringing data into the lakehouse can occur in any format and at any velocity. For example, there might be a need to ingest unstructured data in the form of videos or images into the lakehouse or bring high-speed, real-time streaming data and bulk data assets from Internet of Things (IoT) devices. Each pattern of ingestion requires different tools and technologies to satisfy the technical requirements (velocity, volume, and variety). Tooling for ingesting data into the lakehouse can vary due to source systems, end-user requirements, and network connectivity. There are different (but unified) solutions for batch and streaming data.



Data storage

In the storage layer, structured data is stored in a certain data model, which can be the same as the data source or transformed to ensure quality, accuracy, and usability. The storage layer also stores semi- and unstructured data from file stores and non-traditional data sources like images and conversational text. The data storage layer consists of one or more zones that denote the level of transformation and governance applied to incoming data.

In Figure 2, we are considering three zones: Raw Zone, Enriched Zone, and Curated Zone. Each zone consists of a physical or logical representation of the data. A physical representation of data may be a database table, while a logical representation of data may be a database view. The initial data ingestion processes will land data in the Raw Zone of a lakehouse, initial transformations will build the Enriched Zone, and final transformations and security will be applied to the Curated Zone. In general, most end users, analytics teams, and applications will be pointed to the Curated Zone while other teams such as data scientists and data engineers may have access to data in the Raw or Enriched Zones. Because of decouplement of storage and compute in the lakehouse, there is no need to duplicate data to serve different access patterns and needs, thus there is a unified data storage layer. This is a key benefit to the analytics lakehouse architecture.

Data processing

When transforming data between zones, from Raw to Enriched to Curated, the data processing layer brings the compute required to perform the transformation. Often, this processing takes the form of SQL. The SQL approach is leveraged as the SQL engine is based on a serverless and highly scalable platform, with the ability to process in parallel. In addition, structured programming languages, such as Spark, can be leveraged to perform transformations that do not readily fit in an SQL construct.

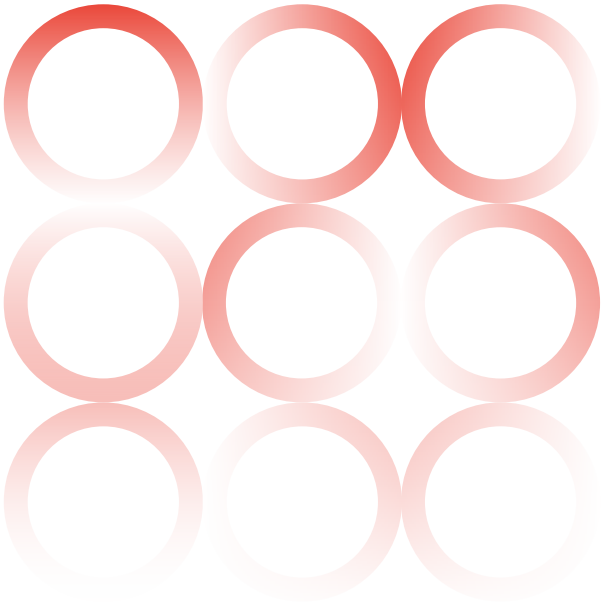
The breadth of needs to process unstructured or file-based data is immense. Analytics lakehouses aim to unify that processing, whether through document processing for analytics or making video and image data usable and searchable for ML use cases.c

Data consumption

Throughout the data lifecycle across the different storage zones, data is transformed into consumable datasets. This allows the analytics lakehouse to provide the unification of the wide-ranging needs to process unstructured data. The consumable datasets are served through the consumption layer and used for different types of applications. These applications, which may require analytics of ML uses, will choose to run queries against the lakehouse and get a filtered and/or aggregated view of the data, or access analytics lakehouse storage directly and perform processing in their specific application. The consumption layer is driven largely by the end users and their technical skill sets and use case needs.

Data governance

The analytics lakehouse empowers federated governance when it comes to responsibilities and roles. From a tooling perspective, federated governance enables centralized data governance, data quality, and data lifecycle. As different domains, personas, and skills are required to govern the lakehouse, a unified data governance layer is needed to provide a centralized place to manage, monitor, and govern your data in the lakehouse, and make this data securely accessible to a variety of analytics and data science tools.



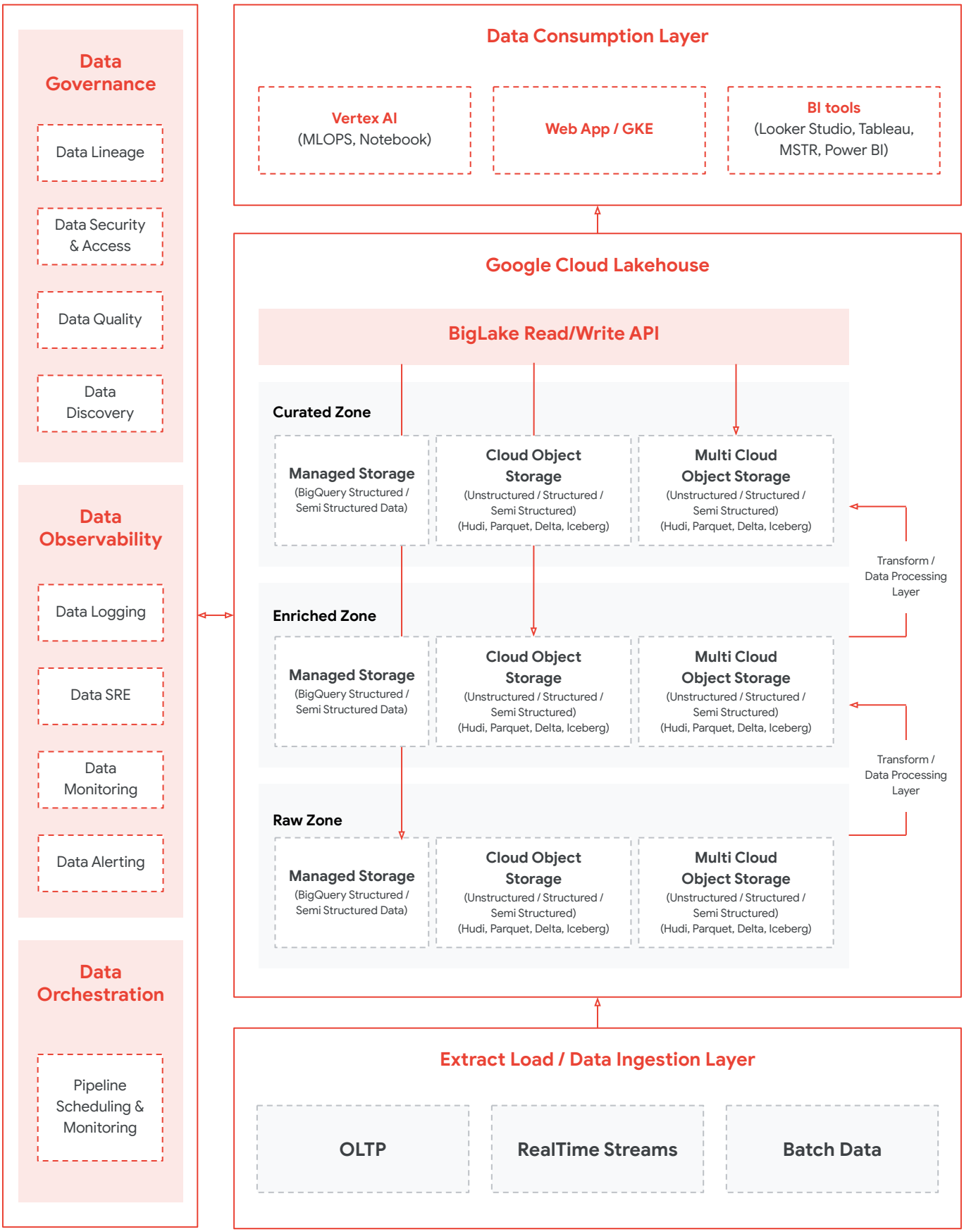


Figure 2: Analytics Lakehouse high level architecture

Chapter 3

Analytics lakehouse on Google Cloud

In the following sections, we will discuss each of the analytics lakehouse layers, the decisions that affect the design of each layer, and the solutions Google Cloud offers to enable customers to innovate faster with an open, unified, and intelligent data platform.

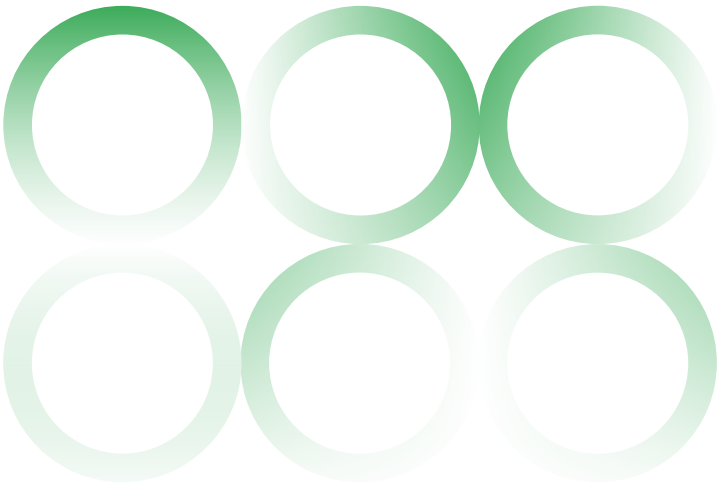
Google Cloud has leveraged our planet-scale analytics platform to bring together two foundational solutions for enterprise data operations — data lakes and data warehouses — and unified their core capabilities with one tool, simplifying management tasks while increasing value. BigQuery, the centerpiece of this architectural revolution, enables the analytics lakehouse, combining the best of data lakes and data warehouses without the overhead of both.

Google Cloud is especially well suited to enable the analytics lakehouse due to our unique combination of advanced data center capabilities and industry-leading services. For example, our data center with a network achieving petabit bisectional bandwidth enables the separation of compute and storage. Separating storage and compute means decoupling storage formats from processing frameworks, meaning that compute is moving to data, rather than data moving and copying to compute. This allows, for example, a Dataproc environment to connect to either Google Cloud Storage or the BigQuery storage subsystem to read/write data at attached storage speeds. This enables Spark developers to leverage data inside BigQuery without the need for data duplication and cumbersome ETL operations to move the data to a different compute cluster. The speed of the internal Google network enables our customers to bring the processing t o the data and avoid data duplication which, in turn, reduces data latency, processing time, data discrepancies, and cost.

In traditional environments, storage administrators would manage blob or file storage, while database administrators would control access to systems such as BigQuery. If data is stored across different mechanisms, such as Google Cloud Storage and BigQuery, this could present additional management complexity. To unify the management, and thus reduce the complexity, Google Cloud offers Dataplex, our intelligent data fabric, which enables you to manage your distributed data assets while making data securely accessible to all your analytics tools.

The following sections will walk through the Google Cloud tooling for designing an analytics lakehouse. They are broken down by operation layer (ingestion, storage, processing, consuming, and governing).

Figure 2 is a summary of Google's analytics lakehouse building blocks. This paper will touch on some of the services in this diagram and how they build on each other to deliver a highly extensible and scalable lakehouse solution.



Ingestion

The purpose of data Ingestion into the analytics lakehouse is to bring together disparate data from across your organization into one unified platform that can curate the data to support a variety of data consumption use cases. Data ingestion into a lakehouse can be challenging due the volume, variety, and velocity of data coming from various data sources. There may be additional complexity based on the different storage sinks as well, such as Google Cloud Storage or BigQuery.

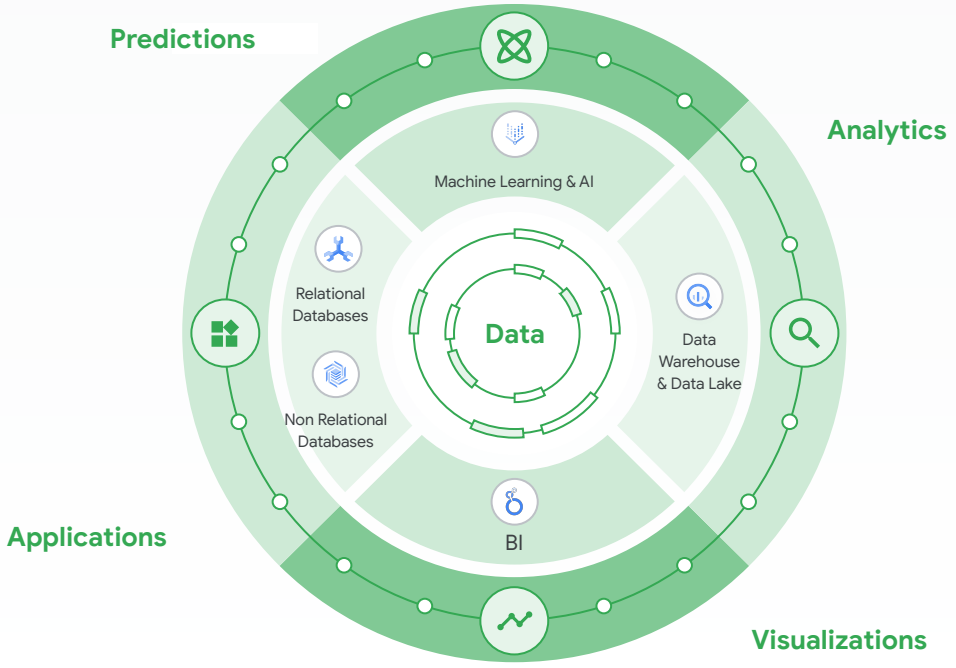


Figure 3: A unified, open and intelligent data ecosystem

To handle the wide variety of data sources required to build a holistic analytics lakehouse, organizations should try to standardize the smallest number of data ingestion solution(s) and design patterns that fit their needs. Choosing the data ingestion solution for your data ingestion layer requires knowing your data sources and targets. The lakehouse supports ingestion from any type of source, including operational data stores, social media, enterprise resource planning (ERP) applications, customer relationship management (CRM) applications, mobile devices, sensors, and video streams. These data sources can be internal or external to your organization. They will generate different types of data: structured, semi-structured, or unstructured and will be available for ingestion into an analytics lakehouse at different times (batch) or continuously (streaming). Data integration tools need to be able to handle ingestion via traditional time-based scheduling as well as support event processing (file arrival, dependency compilation, etc.) and streaming sources.

Before loading data into the lakehouse storage layer from internal or external data sources, consider the source type, structure, volume and arrival rate (batch vs. real time) for the data you would like to ingest. Google Cloud provides multiple options to ingest them into the lakehouse storage layer. In the following section, we will describe these options.

Data Ingestion Methods

One of the first questions to ask when considering a data ingestion solution is the expected arrival rate of the data source you need to ingest. Typically, this is broken down into two categories: real-time data ingestion and batch data ingestion with two sinks. There are often two sinks to choose from: an object-based store such as Google Cloud Storage and a managed datastore such as BigQuery. See Figure 4.

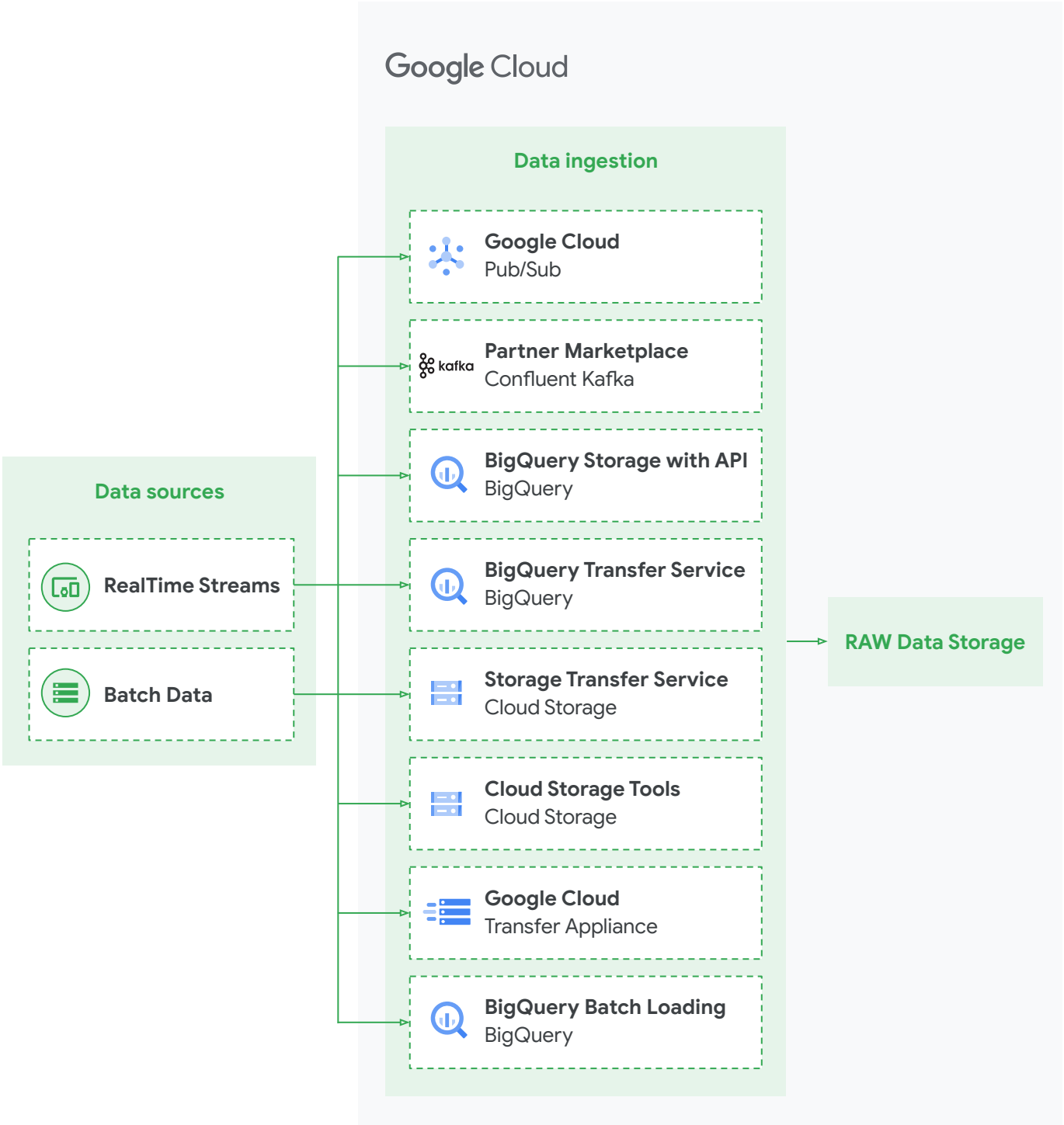


Figure 4: Data ingestion methods overview

Real-time data ingestion

Real-time data ingestion is typically leveraged when ingesting data from event/streaming data patterns. These data patterns include ingesting IoT data, Change Data Capture (CDC), monitoring data, and application data. These types of data sources typically write their data to an intermediate messaging service to overcome the latency associated with file-based processing. Two of the most common messaging services utilized in Google Cloud are Google's Pub/Sub service and Confluent Kafka. Both Pub/Sub and Confluent Kafka can write directly to the analytics lakehouse when leveraging BigQuery's storage API. If more complex streaming transformations are required, Google provides Cloud Dataflow. Cloud Dataflow will read off a message service, perform transformations, aggregations, and/or joins, and write the streaming results to the lakehouse. See Figure 5.

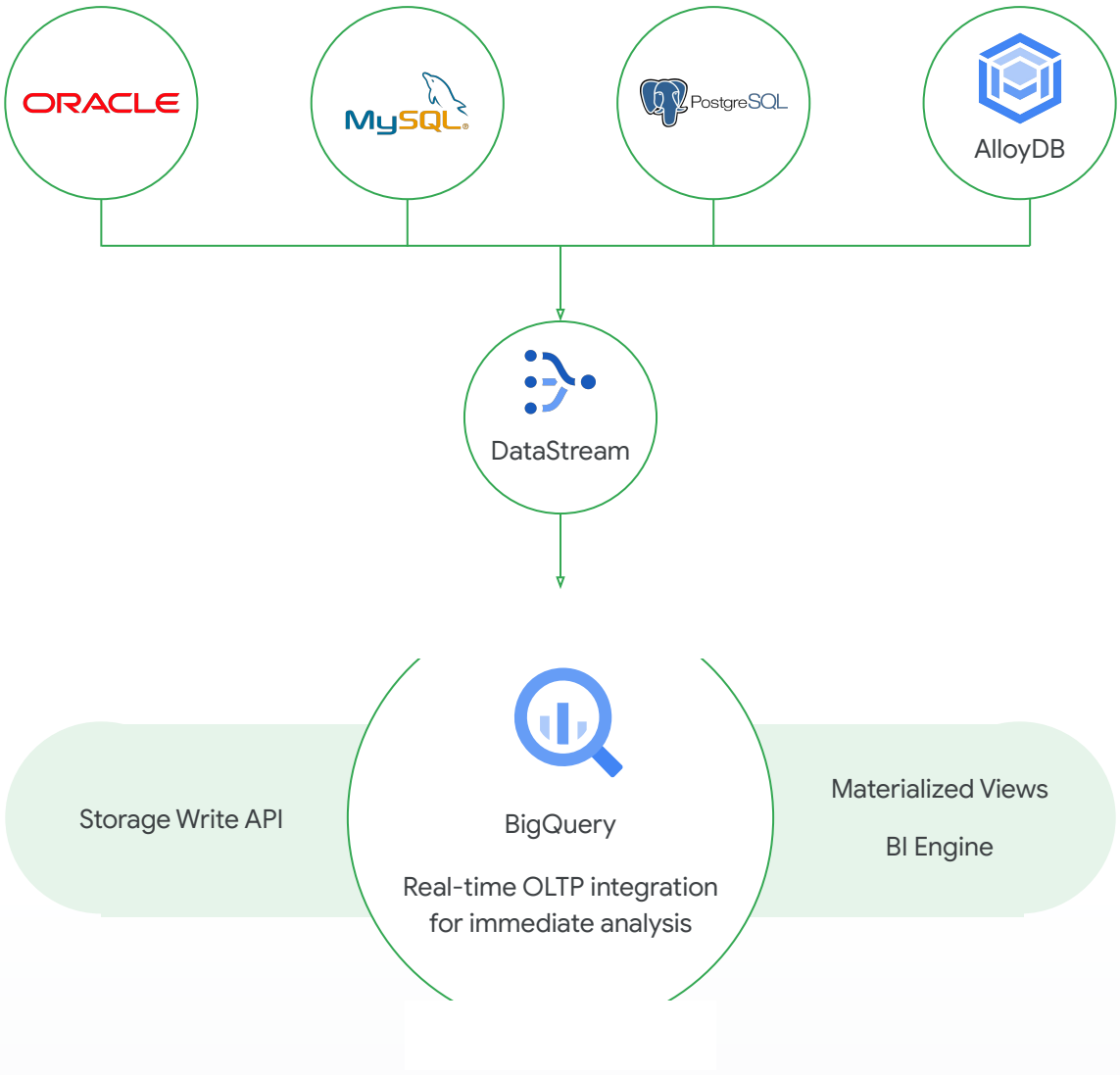


Figure 5: Real-time data ingestion with Datastream

Pub/Sub and Confluent Kafka

Google's Pub/Sub is a fully managed, highly available global messaging service with ≥99.95% of monthly uptime service-level agreement (SLA). Pub/Sub is used for integration pipelines to ingest and distribute data. It is serverless and auto scales as needed. With Pub/Sub, you can create up to 10,000 subscriber apps per topic and it supports a pull as well as a push delivery model to your microservice applications. By default, Pub/Sub retains unacknowledged messages in persistent storage for seven days from the time of publication. There is no limit on the number of retained messages. If subscribers don't use a subscription, the subscription expires. The default expiration period is 31 days. With Pub/Sub, unless you have to transform the data before ingestion, no pipelines are needed to stream data directly to BigQuery with a new type of Pub/Sub subscription called a "BigQuery subscription" that writes directly from Pub/Sub to BigQuery.

Pub/Sub is a key part of the analytics lakehouse because it allows for streaming ingest at a large scale, which is a requirement for streaming data use cases such as IoT, CDC, and application events.

If there are certain requirements that include the ability to seek a particular offset in a topic, possibly to the start of all the data captured or to retain per-key ordering, Confluent Kafka is available from the Google Cloud partner marketplace. It is fully managed with a common practice to use Kafka and Pub/Sub together, especially to leverage the high availability and global presence of the Pub/Sub messaging service.

Cloud Dataflow

While Pub/Sub and Kafka can write data to the BigQuery storage with the lakehouse as a sink, their ability to perform transformations on data before writing is limited. To achieve streaming transformations, Google Cloud provides Cloud Dataflow, a fully managed data processing service. Often used as an extract, transform, load (ETL) tool, Dataflow minimizes latency, processing time, and cost while being built on the OSS community-driven Apache Beam SDK. To handle massive fluctuations in streaming data, Dataflow employs horizontal and vertical autoscaling, which empowers the Dataflow service to automatically choose the appropriate number of worker instances and adjust worker instance size based on the processing required to run a given job. Dataflow employs a rich programming language, Apache Beam, that gives developers the flexibility to code in Java, Python, or Go and allows programmers to easily build complex transformation logic in scale-out pipelines. Dataflow has sources and sinks to a variety of storage, message bus, and database solutions.

Dataflow is an important component of the analytics lakehouse. Like Pub/Sub, it unlocks streaming use cases and provides the capability of more complex transformations before writing to any data storage layer. Because it is a massively scalable and parallel service, the managed service allows data engineers to focus on the data products and productivity, rather than the infrastructure of moving and transforming data.

Batch data ingestion

Batch data ingestion typically involves the movement of data via files structured in common file formats such as CSV, newline delimited JSON, ORC, Parquet, avro, or iceberg to Cloud Storage. Once the data is in Cloud Storage, the data can be accessed directly from clients or BigLake tables, or it can be loaded into BigQuery. Common Google batch data ingestion tools include gsutil (to access data from the command line), Cloud Storage Connector, Storage Transfer Service, Data Transfer Service and Transfer Appliance, and BigQuery batch loading.

Cloud CLI

The Google Cloud Command Line Interface (CLI) is a set of tools to create, manage, and interact with Google resources. One command that is part of the Google Cloud CLI package is gsutil. The gsutil command is an open-source application used to copy the data from any file system location (SAN, NAS, local storage, etc.) into Google Cloud Storage. Data can be copied within Google Cloud locations and remote locations, including an on-premises data center or other cloud provider. In addition, gsutil offers the ability to rsync data for incremental copies with transfer multithreading options. gsutil is an important part of the analytics lakehouse for scripting batch uploads and doing ad hoc data ingestion.

Cloud storage Connector

Frequent sources for lakehouses are existing data lakes running on Hadoop-based clusters. To simplify data movement, Google's Cloud Storage Connector allows the Hadoop DistCp utility to migrate data from the Hadoop Distributed File System (HDFS) directly to Google Cloud Storage without the need to land the data to an external file system first. Due to the large amount of data in Hadoop clusters, it is important to consider where the transfer will run and the network impact of moving large volumes of data. A good discussion of these considerations can be found here.

Storage Transfer Service

Storage Transfer Service is a fully managed service allowing data transfer into Google Cloud Storage from other clouds, on-premises sources, and from one storage bucket to another. Depending on network connectivity bandwidth, the service can scale up to 10 Gbps. The service offers scheduling, retries automation, logging, and intelligent incremental transfers. Storage Transfer Service currently does not provide transfer performance or latency SLA because it is highly dependent on your network connectivity and setup.

Storage Transfer Service moves large amounts of data "over the wire" to Google Cloud Storage. This is especially useful when migrating an existing data lake from another cloud to a Google Cloud lakehouse or moving large amounts of data from on-premises NAS/SAN storage to Google Cloud Storage for ingestion into the lakehouse.

Data Transfer Service

The BigQuery Data Transfer Service is a managed service with a data delivery service SLA of less than or equal to 24 hours. Use this option if BigQuery storage is going to be the storage layer of your analytics lakehouse or if you want to use a Google built-in supported transfer solution to BigQuery. BigQuery Data Transfer Service can be used for third-party SaaS services such as YouTube channels, Google Ads, Amazon S3, Redshift, and Teradata. Because the analytics lakehouse excels at bringing together data from different sources, the Data Transfer Service is one of the easiest ways to ingest data into BigQuery.

Transfer Appliance

Transfer Appliance is a high-capacity storage device that enables you to transfer and securely ship your data to a Google upload facility, where we upload your data to Cloud Storage. This option is often used when ingesting large datasets into Google Cloud that would take more than one week to upload over the network. With a typical network bandwidth of 100 Mbps, one petabyte of data takes about three years to upload. However, with Transfer Appliance, you can receive the appliance and capture a petabyte of data in under 25 days. Your data can be accessed in Cloud Storage within another 25 days, all without consuming any outbound network bandwidth.

When building a lakehouse, Transfer Appliance is especially useful for migrating a massive historical data store to Google Cloud Storage. Often, customers will use Transfer Appliance in conjunction with Storage Transfer Service and migrate large amounts of static history with Transfer Appliance while keeping current data in sync with Storage Transfer Service.

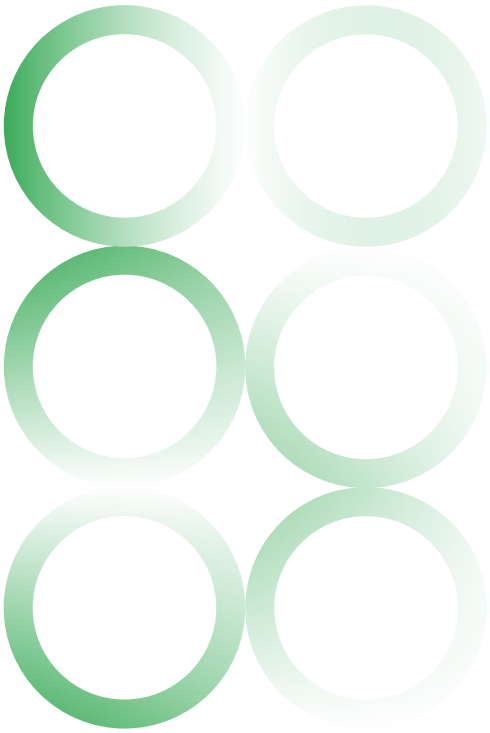
BigQuery Batch Loading

BigQuery batch loading is the throughput approach to load file-based data from Google Cloud Storage into BigQuery Storage. BigQuery batch loading utilizes the BigQuery load API that can be called from the console, command line, Cloud Composer, BigQuery Data Transfer Service, or from seven different programming languages.

Data to be loaded into BigQuery can be in a variety of open formats, including Avro, Parquet, ORC, CSV, or JSON (newline-delimited). Load performance can vary between file formats, with Avro being the fastest followed by Parquet, ORC, CSV, and JSON. Codec-based compression (such as Snappy) with Avro, Parquet, and ORC can optimize load performance by decreasing file sizes while still allowing parallel reads. When possible, avoid external compression (such as gzip) with large CSV or JSON files as externally compressed files cannot be split and read in parallel. That being said, consider the full path length of a data ingestion pipeline and optimize for the overall benefit of external compression for data transmission vs. the serialization penalty incurred when loading externally compressed files.

BigQuery utilizes a pool of compute dedicated to performing load operations. This means that data can be loaded into BigQuery while users are running queries with no impact on query performance. BigQuery batch loads in an on-demand project, uses a shared pool of compute, and is a free operation. If a strict SLA is required for loading data, consider reserving a pool of dedicated compute for load operations.

BigQuery Batch Loading is an important ingestion mechanism for lakehouse pipelines as it allows data engineers to ingest vast amounts of data into BigQuery using the highest throughput method available. This data ingestion approach supports the Extract Load Transform (ELT) processing paradigm that will be discussed in the BigQuery Compute section below. While batch loading provides the highest throughput approach, data processed via batches incurs latency during extraction of data from the source and the transmission of the files to Cloud Storage where they will be loaded. If a low latency approach is required, consider a streaming approach described in the Real-time section above.



Data Ingestion Design Considerations

There are many decisions to make based on the sources and sinks for an analytics lakehouse. While the following are not comprehensive, they highlight some of the questions and ideas you must consider when designing the ingestion layer for your analytics lakehouse:

- Pub/Sub Lite is up to 90% cheaper but less reliable than Pub/Sub. As opposed to Pub/Sub, Pub/Sub Lite requires you to manage your resource capacity and reservations. Refer to the documentation for differences between Pub/Sub and Pub/Sub Lite.
- If you are loading your data into BigQuery, consider using Avro formats for the following reasons:
    - It is faster to load. The data can be read in parallel, even if the data blocks are compressed.
    - It doesn't require typing or serialization.
    - It is easier to parse because there are no encoding issues found in other formats such as ASCII.
    - When you load Avro files into BigQuery, the table schema is automatically retrieved from the self-describing source data.
- When designing your data ingestion and transfers, reduce egress cost and client latency by compressing your data upload.
- If you are ingesting data from compute instances, ensure that communication between your services within a zone is routed through their internal IP addresses, and not routed externally.
- When possible, choose cloud services that are close to your consumer and data source locations. For example, if your data sources are in the Frankfurt on-premises data center, choose your landing storage bucket in the Frankfurt region to reduce the latency and cost of the data transfers.
- To reduce the cost of moving high volumes of data to the cloud frequently, use a direct connection between the on-premises and Google Cloud networks. Consider using Dedicated Interconnect or Partner Interconnect.

Figure 6: Data ingestion methods overview

Data Storage

For an analytics lakehouse, some of the most important decisions pertain to where the data will be stored and how it will be structured. It is important to consider data domains, ingestion patterns, level of transformation required, and consumption patterns when designing your analytics lakehouse storage.

In the following section, we will explain what data storage solutions we recommend for your use cases.

Lakehouse Data Storage Solutions

When storing data in your lakehouse, consider different types of storage solutions. This section will discuss the pros and cons of storing data in various lakehouse storage subsystems.

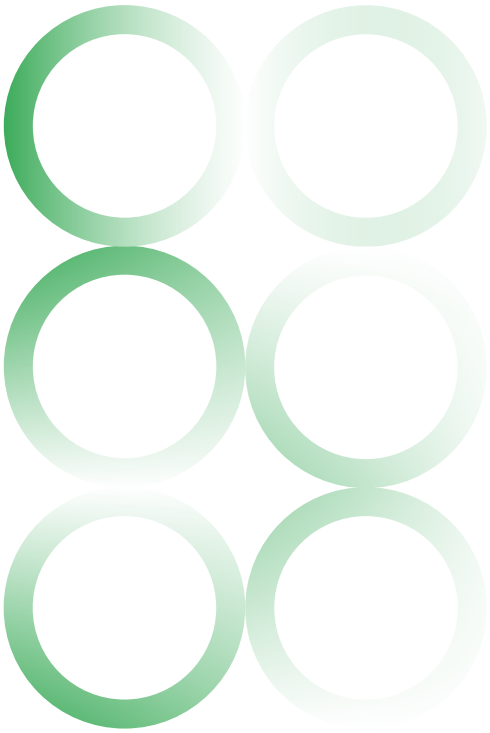
BigQuery Storage

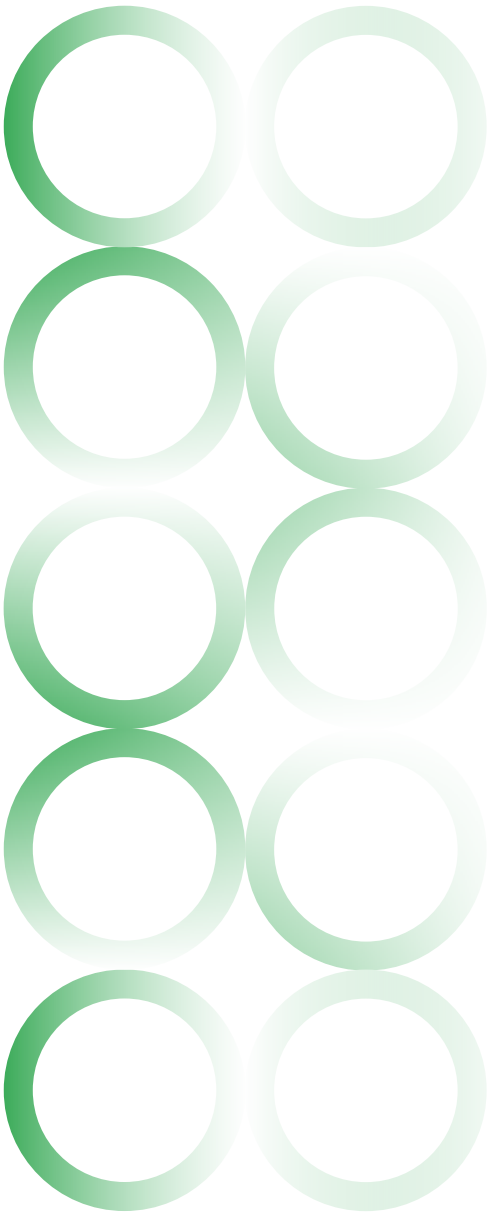
When building a lakehouse, BigQuery Storage is the recommended solution for most structured data storage use cases. BigQuery Storage is an exabyte scale and cost-effective storage subsystem designed for efficiency to drive business agility. Data in BigQuery storage are stored in a columnar format which makes it highly performant for analytics workloads. All table modifications in BigQuery, including DML operations, queries with destination tables, and load jobs, are ACID-compliant and all table maintenance activities such as encryption, compaction, and replication are handled automatically by the storage subsystem. BigQuery Storage supports CDC and IoT-style streaming ingestion patterns natively leveraging the BigQuery Storage Write API.

Per design, BigQuery decouples storage from compute, which means you can access the storage layer of BigQuery from any processing and analytic engine like Spark, BigQuery SQL, Presto, Flink, Tensorflow, or Trino in either batch or streaming fashion. With embedded streaming and machine learning capabilities, BigQuery is an ideal solution for both data lake and data warehouse use cases. Storing data in BigQuery is cost-effective and comparable to the cloud object storage cost in the market. In addition, BigQuery storage is highly durable with an SLA of 99.999999999%. Conceptually, if you store 1 million objects in BigQuery storage for 10 million years, you could expect to lose 1 object.

Last, BigQuery lets you use time travel to access data stored in BigQuery that has been changed or deleted. You can access the data from any point within the time travel window, which covers the past seven days. Time travel lets you query data that was updated or deleted, restore a table that was deleted, or restore a table that expired.

Unless you have a hard requirement to use big data open-source data formats or your data is highly unstructured, you should consider BigQuery Storage as your lakehouse storage solution.





Google Cloud Storage

Another storage solution Google Cloud offers is Google Cloud Storage. Google Cloud Storage is a reliable, highly available, durable object storage. It has a durability SLA of 99.999999999% and an availability SLA of 99.95%. Cloud storage can be resident to a Google region or distributed to multiple or dual zones for high availability. Lastly, Google Cloud Storage has various storage classes with the same API access so you can cost optimize your storage without changing the way you access the data.

Since cloud storage is an object store, you can store a wide variety of highly structured big data formats such as json, Parquet, avro, iceberg, and orc; semi-structured data formats such as pdfs, docs, and spreadsheets; and unstructured data formats such as binary, video, and image data. Historically, one of the challenges of using cloud storage to store data lake/lakehouse data was the level of granularity of security. Often data is secured at the bucket level with the option to secure at the file level, so data access tends to be "all or none" at the bucket or file level. To provide fine-grained (row/column) level access controls to structured data in your Cloud Storage, BigLake table support was built to bring BigQuery granular data access methods to structured data in Cloud Storage. We will talk more about BigLake tables in the next section.

Overall, we recommend Google Cloud Storage as your analytics lakehouse storage solution for use cases with requirements like the following:

- You are required to use big data open-source data formats such as Parquet, Avro, Iceberg, Delta, or Hudi.
- Your data are highly unstructured (e.g., videos, audio files, or images).
- You are required to store infrequently used data in a highly cost-effective Cloud Storage class (e.g., coldline or archive storage).

BigLake Tables

BigLake's mission is to bring the power of BigQuery Storage and Google Cloud Storage to open lakehouse. BigLake acts as a layer on top of the storage engine that unifies data warehouses and lakes by providing uniform fine-grained access control and performance acceleration for open formats stored on Google Cloud Storage and multi-Cloud Storage. It is an open API interface for analytics engines spanning across Google Cloud and open-source runtimes to access your distributed data with consistent security and governance controls. BigLake helps reduce cost, scales across more workloads efficiently, and expands BigQuery to multi-cloud data lakes leveraging BigQuery Omni.

The Figure 7 shows you where the BigLake sits within the analytics lakehouse.

From usage perspective, different personas can access the BigLake layer from any processing and analytic engine like Spark, BigQuery SQL, BigQuery ML, Presto, Trino, Flink, or Tensorflow. The vectorized runtime layer of BigLake allows faster columnar scans with user predicates filtering and security evaluation. Internally, in BigLake, data is served in Apache Arrow format with policies consistently enforced.

BigLake provides a unified experience to different personas, for example, data scientists or data engineers, so they can choose and mix their lakehouse storage solutions depending on their use cases.

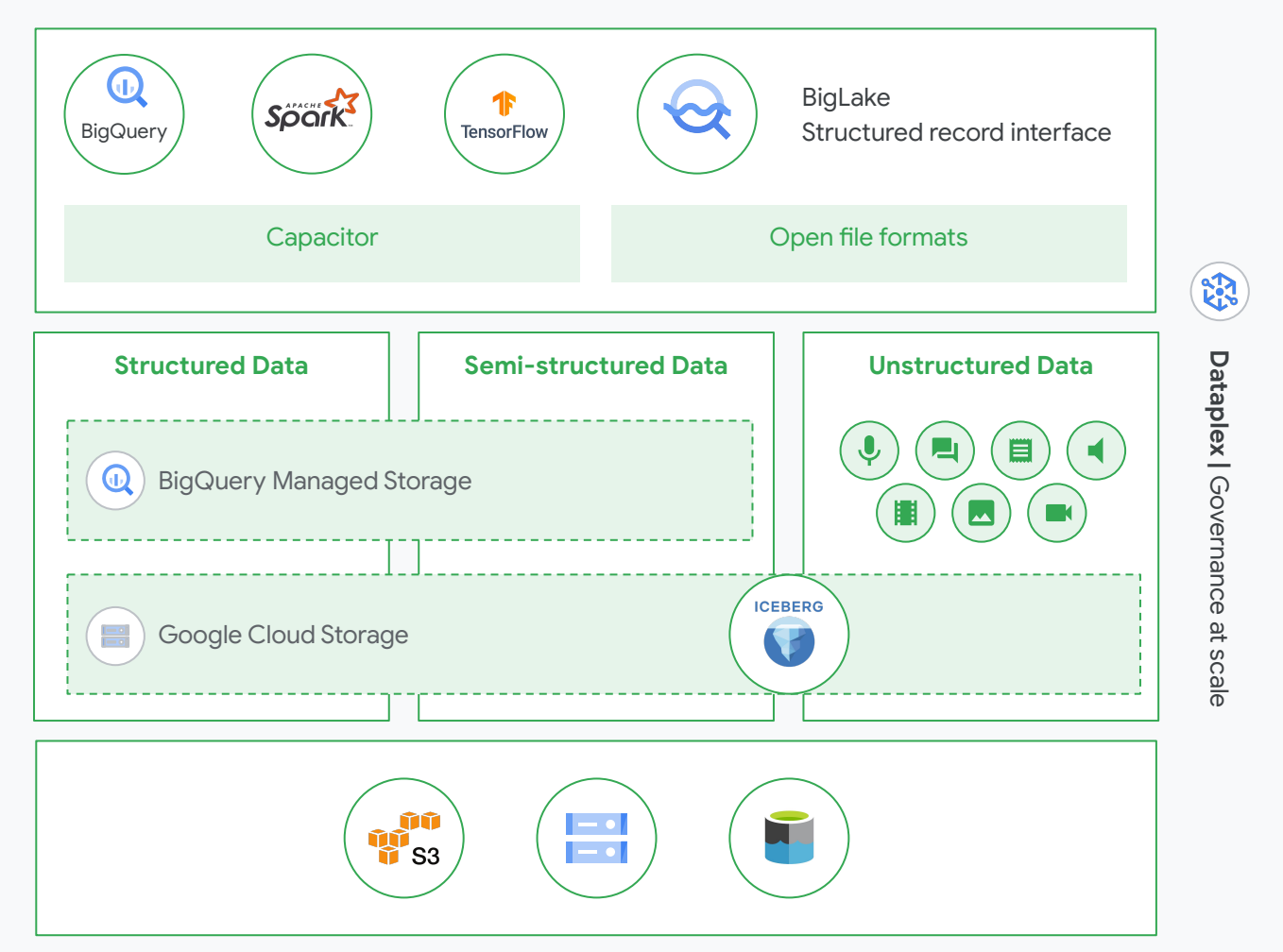


Figure 7: BigLake Virtualization Layer

Lakehouse Data Storage Considerations

When trying to decide between ingesting data into BigQuery Storage, leaving data in cloud storage, and utilizing multi-Cloud Storage, consider the following:

Use BigQuery Storage when:

- You are looking for a managed storage solution that reduces operational overhead and can serve many use cases.
- Your applications require a highly optimized analytical store.
- Full DML, ACID, and advanced CDC support is required.
- You want to store ML models in BigQuery and leverage BigQuery ML.
- You want to leverage BI Engine sub-second query response time with minimal load times and intelligent caching for data stored in BigQuery.
- It is required to have high throughput streaming into BigQuery by using BigQuery Storage Write API.

Use Cloud Storage when:

- You are required to use open formats such as Parquet, Iceberg, Hudi, or Delta.
- Your applications require unstructured data.
- You are migrating from a Hadoop or a datalake and you need the Hadoop as an intermediate storage option (temporary staging storage).
- You want to leverage BigLake tables to provide fine-grained access control to open file formats.

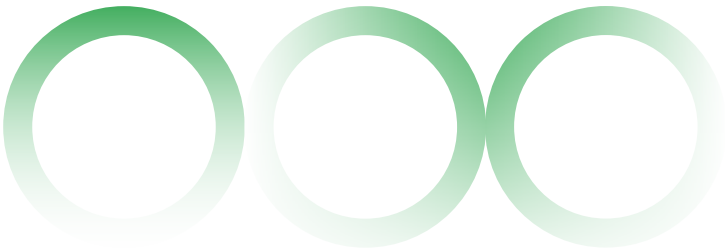
Use Multi cloud storage (S3, Azure Storage) when:

You cannot move the data into managed storage such as BigQuery and data needs to stay in another Cloud for operational reasons.

While far from a comprehensive list, some requirements to consider when deciding how to store your data are:

- Real-time analytics use case that requires very low latency consumption
- Migration of your already on-premises or cloud data lake to the Google Cloud with the requirement to use big data open-source data formats
- Data archiving for infrequently accessed data
- Need for time travel to let users roll back table mutations and use point-in-time snapshots
- Need for automatic table optimizations (e.g., compaction or shuffling optimization)
- Improve schema evolution by supporting
- column renames, reordering, and partition evolution support
- Multi-statement, multi-table transactions support
- High-volume, record-level DML support
- Change Data Capture

By now, we have explained what storage solutions you could use for your use case and the importance of having BigLake enabled in your lakehouse environment. In the following section, we will zoom in to each of your storage solutions and explain why we need to have different structures and zones within the lakehouse storage layer.

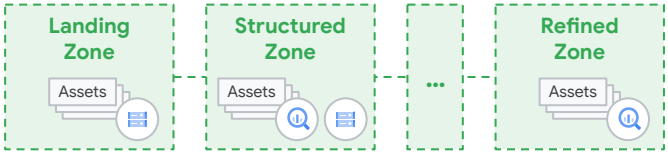


Lakehouse Storage Zones

As discussed in the Conceptual Architecture section, when you are ingesting your data into the lakehouse storage, your data will be stored in different zones depending on the stage of the processing layer and the value of your data. The data can be stored in the following zones:

Raw Zone

The Raw Zone typically contains the immutable data as it was ingested from the data source. This data usually needs some processing before most of your organization can leverage it, but occasionally, data scientists and data analysts use it as part of their data discovery. Data engineers generally build the data pipelines and use standardized processes and tools to create automated pipelines for data rationalization, cleansing, and quality. Because standardized processing is required to transform data in this zone, the data is usually accessible by automated pipelines and service accounts instead of IAM users. It is recommended to organize your Raw Zone by source as that can be useful for data lifecycle management and billing. The Raw Zone for batch files is generally Cloud Storage, while raw streaming data typically arrives in BigQuery tables.



Enriched Zone

The Enriched Zone is an evolved storage of the Raw Zone. Data in this zone is typically evolved by structured data engineering pipelines. Occasionally, data scientists or data analytics may use this data and apply transformations using their analytics tools of choice (e.g., Vertex AI Notebooks or SQL in BigQuery) for targeted analysis. In the Enriched Zone, schema is well enforced, data governance and quality rules have been applied on this data (e.g., sensitive data is anonymized), and data is cleansed and optimized for most common consumption patterns.
It is important to note that just because data has had transformations applied in this layer does not mean that the data needs to be physically copied to the Enriched Zone. If transformations are simple and/or the data is arriving with a high frequency, such as streaming, it can be beneficial to create database views or materialized views to apply transformations without having to make a physical copy of the data. Data analysts and data scientists usually have access to this data to further prepare the data, create their ML models or BI reports, and extract business values from it to further develop their data products.
The Enriched Zone can be either BigQuery datasets or cloud storage buckets.

Curated Zone

The Curated Zone is highly structured, enriched, and validated, which makes it particularly useful for traditional analytics. It is usually accessible to business users to drive business decisions (using Data Marts for LoBs). Data in this zone is aggregated (feeding dashboards with KPIs) and cross joined with other data in the lakehouse. This data should be easily discoverable by business users with a high degree of quality as it drives business decisions. The Curated Zone can be either BigQuery datasets or cloud storage buckets. As with the Enriched Zone, Curated Zone objects do not need to be physically copied to this layer as long as database views and materialized views are sufficiently performant.
The data in this zone can either be evolved using data engineering pipelines or data scientists/data analytics transformations using their tools of choice.
Now that we have explained the lakehouse data storage and ingestion solutions, in the next section we will explain the lakehouse data processing layer in detail.

Compute / processing

The power of Google's lakehouse is the separation of storage and compute across our various services. The separation of storage and compute gives the user the flexibility to store the data in the format and repository that best suits their workload and choose the compute needed to process it efficiently using tools and languages that are familiar to them. This section will focus on the processing of data in the analytics lakehouse; the various tools we can use to accomplish this task, including BigQuery, BigQuery Omni, Dataproc, Serverless Spark, Spark Stored Procedures, and Dataflow; and how to enable various tools to take advantage of BigQuery storage. See Figure 8.

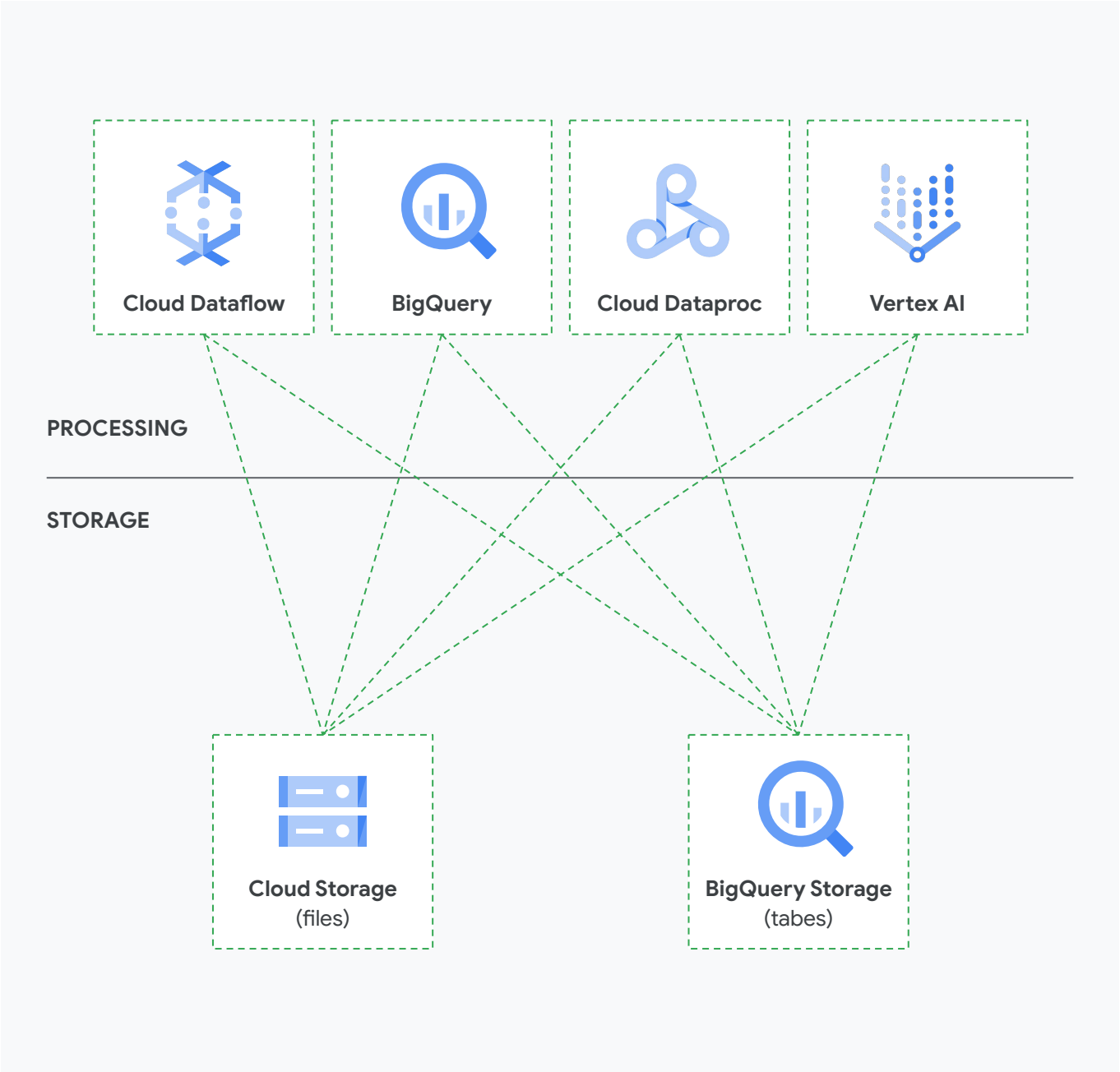


Figure 8: Separation of storage and compute through high speed network

BigQuery

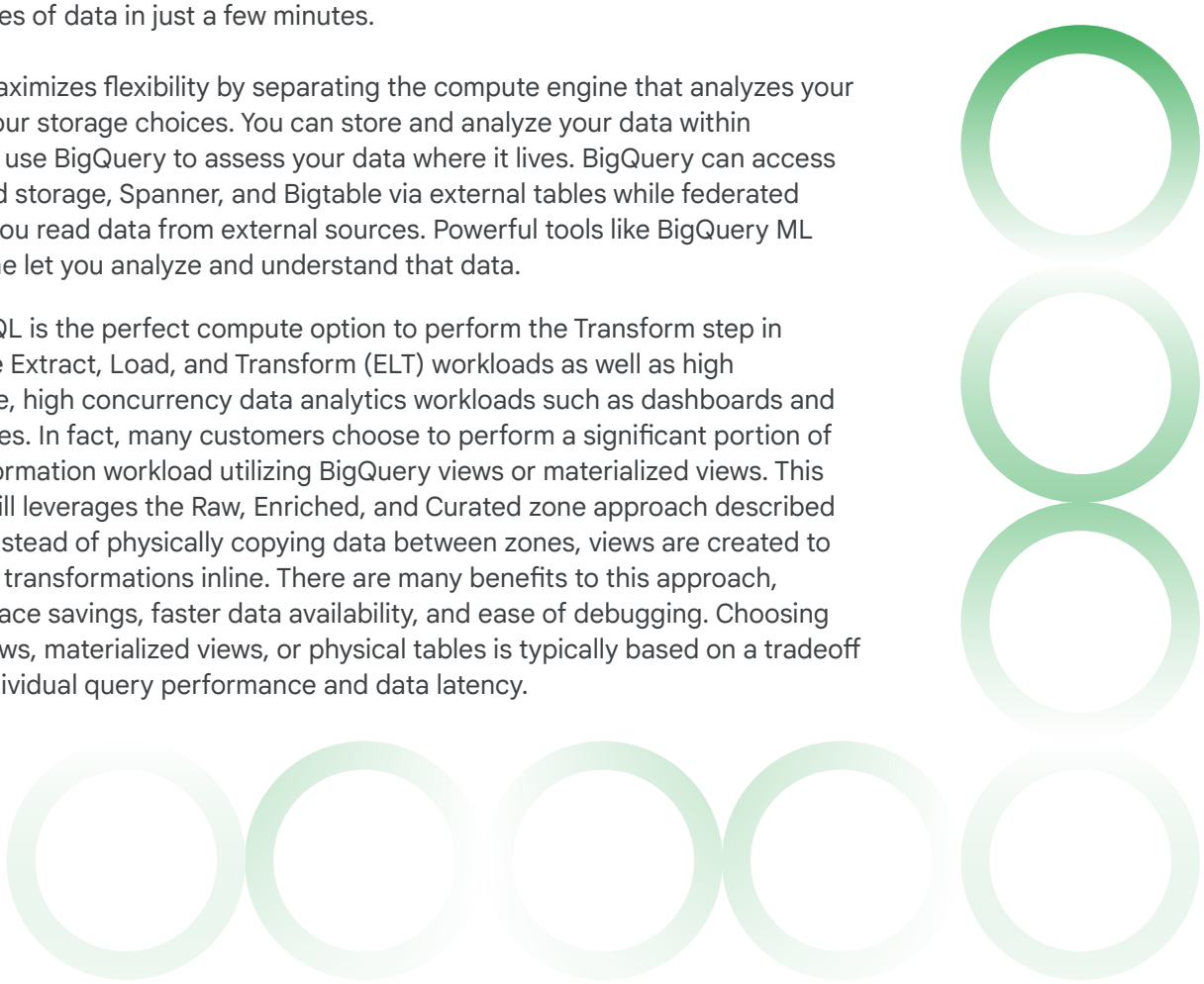
BigQuery's serverless architecture lets you use SQL queries to answer your organization's biggest questions with zero infrastructure management. BigQuery's scalable, distributed analysis engine lets you query terabytes in seconds and petabytes in minutes. BigQuery's serverless architecture makes it highly resilient to rack and node downtime and the ephemeral nature of its compute allows for online rolling software upgrades.

BigQuery's processing is accomplished via BigQuery slots. A slot is a unit of compute in BigQuery which consists of some amount of CPU, memory, and network used to service requests. Slots are dynamically allocated at the step level of a query. This granular application of slots ensures that you are only applying the compute needed to process a step. BigQuery can even dynamically increase or decrease the number of slots applied to subsequent steps of a query based on the results of the previous step to ensure queries are efficiently processed.

Loading of BigQuery is accomplished via a dedicated pool of load slots. These slots also consist of CPU, memory, and network but are dedicated to the job of loading data from Google Cloud Storage to BigQuery storage. This separation of slots ensures that load jobs do not impact query performance and enables BigQuery to load terabytes of data in just a few minutes.

BigQuery maximizes flexibility by separating the compute engine that analyzes your data from your storage choices. You can store and analyze your data within BigQuery or use BigQuery to assess your data where it lives. BigQuery can access data in cloud storage, Spanner, and Bigtable via external tables while federated queries let you read data from external sources. Powerful tools like BigQuery ML and BI Engine let you analyze and understand that data.

BigQuery SQL is the perfect compute option to perform the Transform step in high-volume Extract, Load, and Transform (ELT) workloads as well as high performance, high concurrency data analytics workloads such as dashboards and adhoc queries. In fact, many customers choose to perform a significant portion of their Transformation workload utilizing BigQuery views or materialized views. This approach still leverages the Raw, Enriched, and Curated zone approach described above but instead of physically copying data between zones, views are created to perform the transformations inline. There are many benefits to this approach, including space savings, faster data availability, and ease of debugging. Choosing between views, materialized views, or physical tables is typically based on a tradeoff between individual query performance and data latency.



BigQuery Stored procedures for Apache Spark

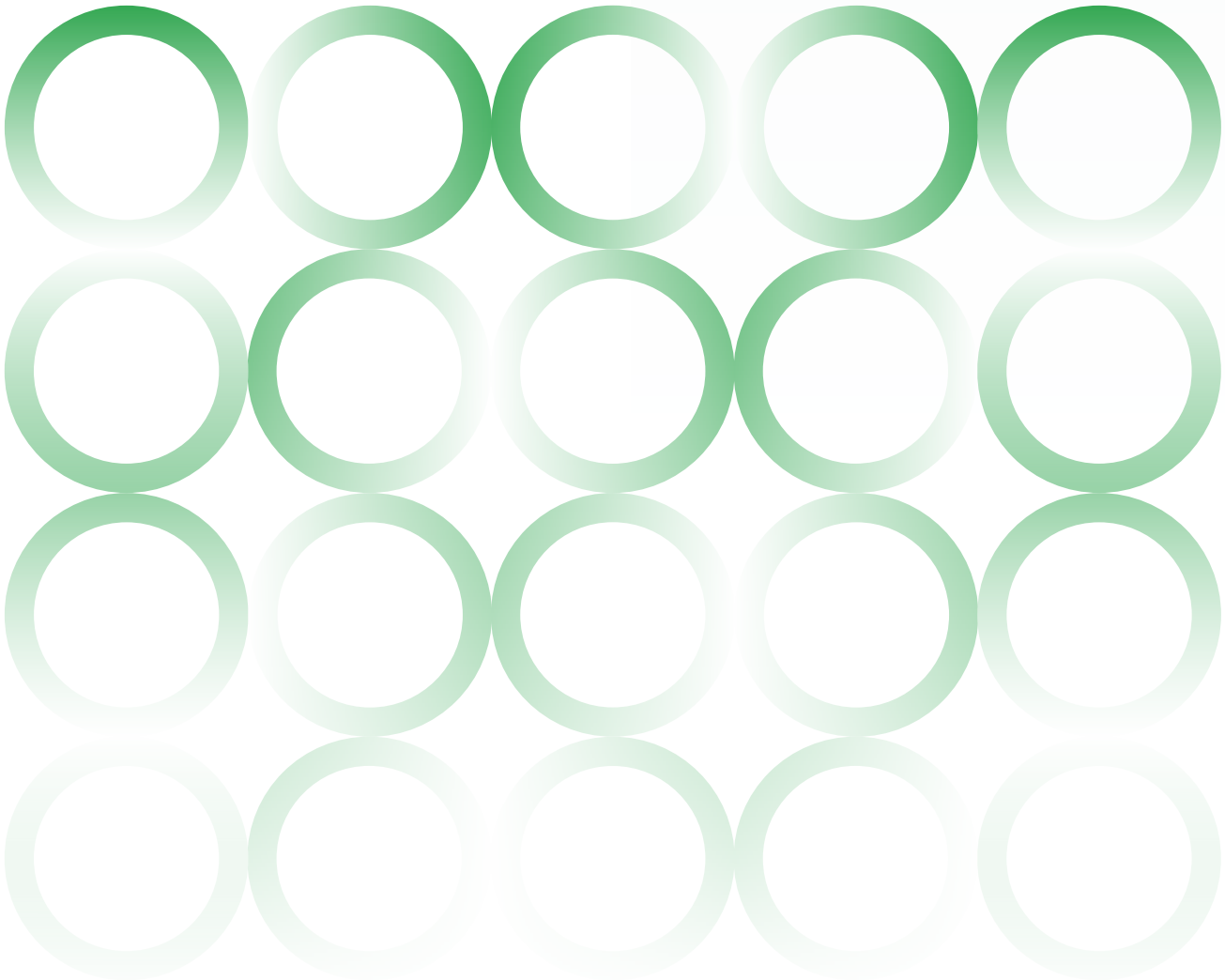
With BigQuery stored procedures for Apache Spark, you can create and run Apache Spark stored procedures written in Python. Once you create a stored procedure you can then execute it through BigQuery using a Google Standard SQL query, similar to running SQL or JavaScript-based stored procedures.

BigQuery stored procedures for Apache Spark allow developers to operate on entire tables worth of data to perform complex transformations that are not well suited to standard SQL. Encapsulation of business logic in a stored procedure allows non-spark savvy SQL users to invoke Spark logic with a common user interface.

BigQuery Remote Functions

A BigQuery remote function lets you incorporate Google Standard SQL functionality with software outside of BigQuery by providing a direct integration with Cloud Functions and Cloud Run. With BigQuery remote functions, you can deploy your functions in Cloud Functions or Cloud Run implemented with any supported language, and then invoke them from Google Standard SQL queries.

BigQuery remote functions greatly expand the functionality of BigQuery UDFs, allowing integrations with external web services to accomplish tasks such as name and address cleansing, and allow you to leverage popular open-source libraries such as RDKit to apply Cheminformatics transforms, such as determining molecular weight, or Quantlib to apply financial modeling and risk transformations. Remote functions increase the types of data transformations that can be performed within BigQuery and expand what can be accomplished with the recommended ELT design pattern.



BigQuery Omni

BigQuery Omni extends BigQuery's fully managed and serverless compute paradigm to AWS and Azure clouds. BigQuery Omni leverages the external table concept to connect BigQuery compute running in AWS and Azure to open format files stored in each cloud's object store. This allows BigQuery to run queries on other clouds against data resident in that cloud while providing a single pane of glass to all your data assets. See Figure 9.

BigQuery Omni allows you to bring the compute to the storage and avoid large data egress charges. This functionality is especially useful when you have large amounts of data in another cloud and want to filter and/or aggregate the data down to a manageable size before performing a cross cloud load of the data into BigQuery storage.

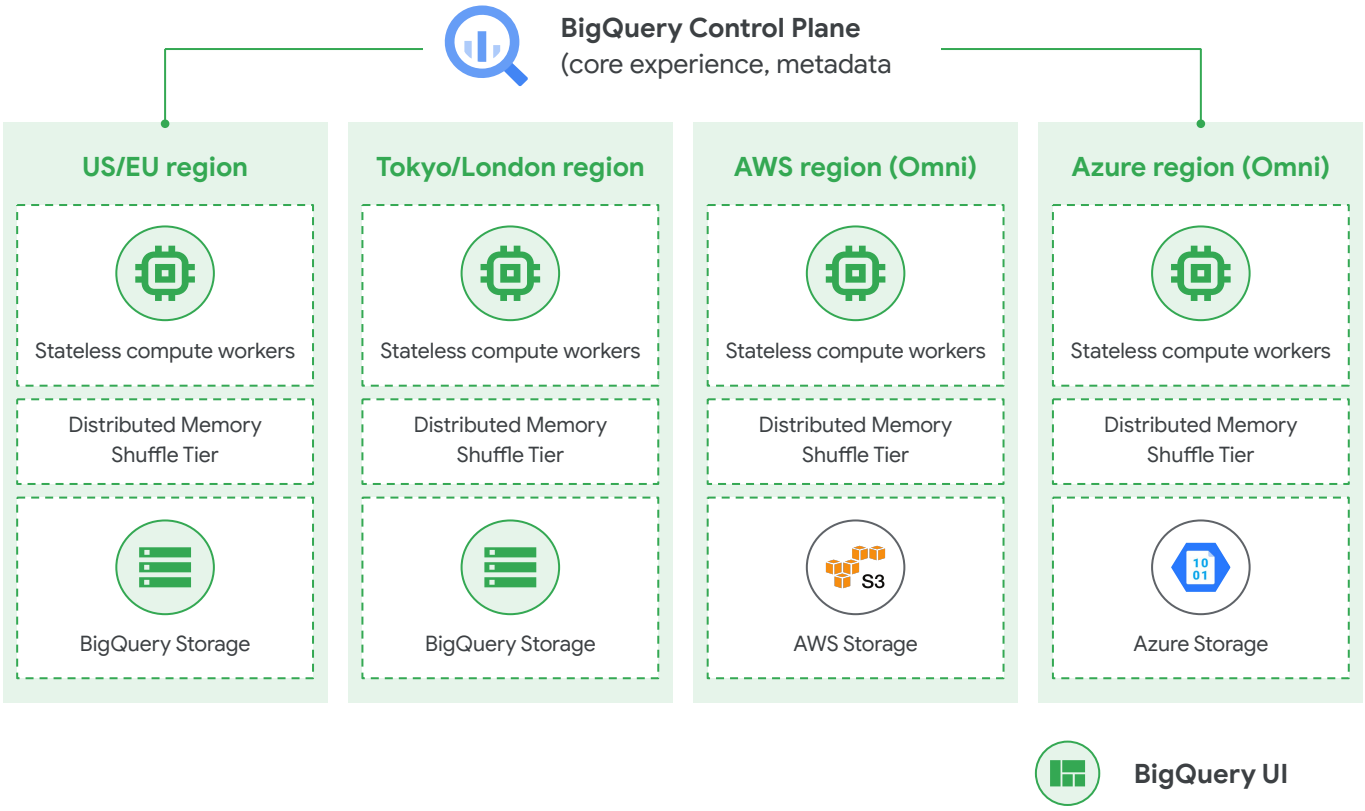


Figure 9: BigQuery Omni

BigQuery Analytics Hub

Traditional data-sharing ecosystems can introduce significant challenges. They lead to expensive, fragile data pipelines with an increased cost of unnecessary data replication. Furthermore, data replicated in many systems leads to governance and security risks as it decreases visibility and control of data.

Analytics Hub helps organizations unlock the value of data sharing, leading to new insights and business value. With Analytics Hub, you create a rich data ecosystem by publishing and subscribing to analytics-ready datasets. Because data is shared in place, data providers can monitor and control how their data is being used.

Analytics Hub provides a self-service way to access valuable and trusted data assets, including data provided by Google.

Finally, Analytics Hub provides an opportunity for organizations to monetize their data assets. We hear from customers who have valuable data but don't want to deal with the overhead of building the infrastructure required for monetization. Our goal is to enable you to offer data to your customers and efficiently manage the data delivery process.

Figure 10 shows the end-to-end architecture of Analytics Hub.

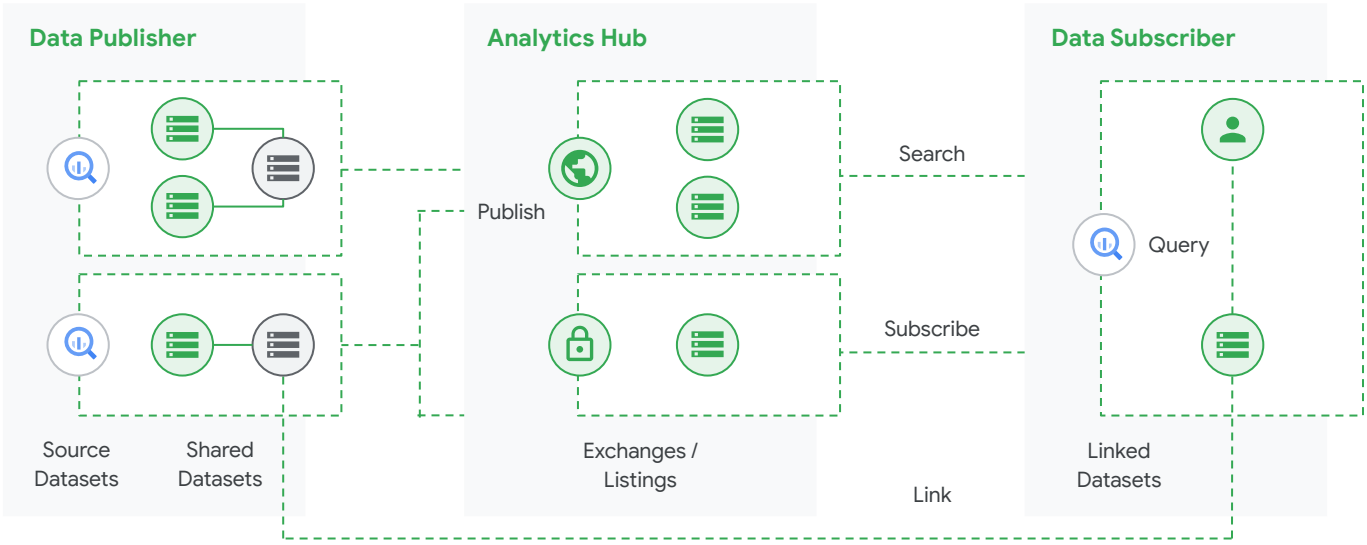


Figure 10: Data Sharing with Analytics Hub

Starting on the left, data publishers bring data into BigQuery and create shared datasets which contain the data that they want to share. This not only includes data in BigQuery managed storage, but data stored in open formats accessed through the recently announced BigLake platform.

They publish these datasets into an exchange with additional information about the data, for example how often it is updated or sample queries to run against the data.

Finally, on the right you can see the data subscriber. They search through the listings across multiple exchanges and then subscribe to relevant datasets. This creates a linked dataset in their project that they can query and join with their own datasets.

Dataproc

Dataproc is a managed Spark and Hadoop service that lets you take advantage of open-source data tools for batch processing, querying, streaming, and machine learning. Dataproc runs on Google Compute Engine (GCE) or Google Kubernetes Engine (GKE) and can be deployed as long running clusters or short-lived, ephemeral clusters using provided automation that helps you create clusters quickly, manage them easily, and save money by turning clusters off when you don't need them.

Dataproc's separation of storage and compute is accomplished by storing the input and output to Dataproc processes on Google Cloud Storage or in BigQuery Storage. This is accomplished via the Google provided Cloud Storage Connector included in Dataproc images and allows you to reference cloud storage data as you would HDFS data, but instead of using hdfs:// you specify gs://. In addition, Dataproc Spark jobs can reference BigQuery Storage directly using the BigQuery Connector for Spark to leverage BigQuery as a source.

Dataproc is generally leveraged when there are well-known workloads that are either long running or ephemeral and may benefit from Hadoop tooling outside of Spark. While Serverless Spark provides a super flexible platform for development or adhoc requests, building a right-size Dataproc cluster for well-known workloads avoids latency introduced when scaling a cluster up to the needed size.

Serverless Spark

With Serverless Spark, data engineers, data scientists, and analysts can spend more time building code and logic and less time managing platforms. The Serverless Spark offering means that they do not need to manage clusters or tune infrastructure. SQL or PySpark jobs can be managed from the interface of choice, including BigQuery console, Dataplex, Vertex AI Managed Notebooks, and the command line and processing is auto scaled to match the needs of the job. Serverless Spark jobs are ephemeral in nature and do not have their own long-term storage. Instead, they leverage cloud storage, BigQuery, or both to bring together data wherever it lives for processing and storing results.

Given the ease of Spark job creation and submittal, Serverless Spark makes an ideal iterative development platform and a great way to inject Spark processing into a diverse workflow. Figure 11 shows the breakdown of serverless Spark versus a non-serverless offering.

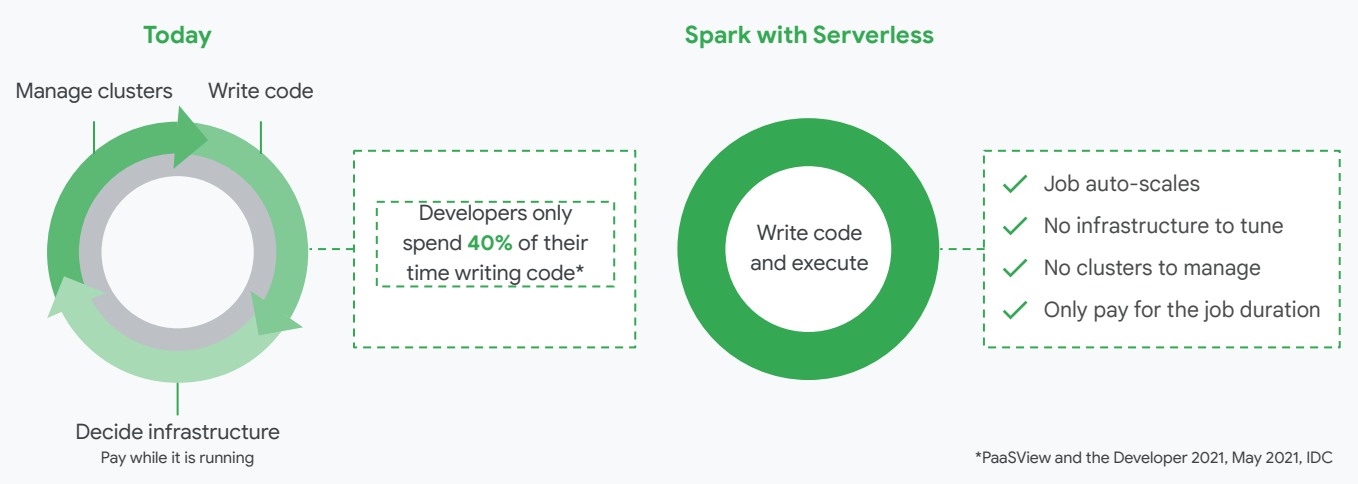
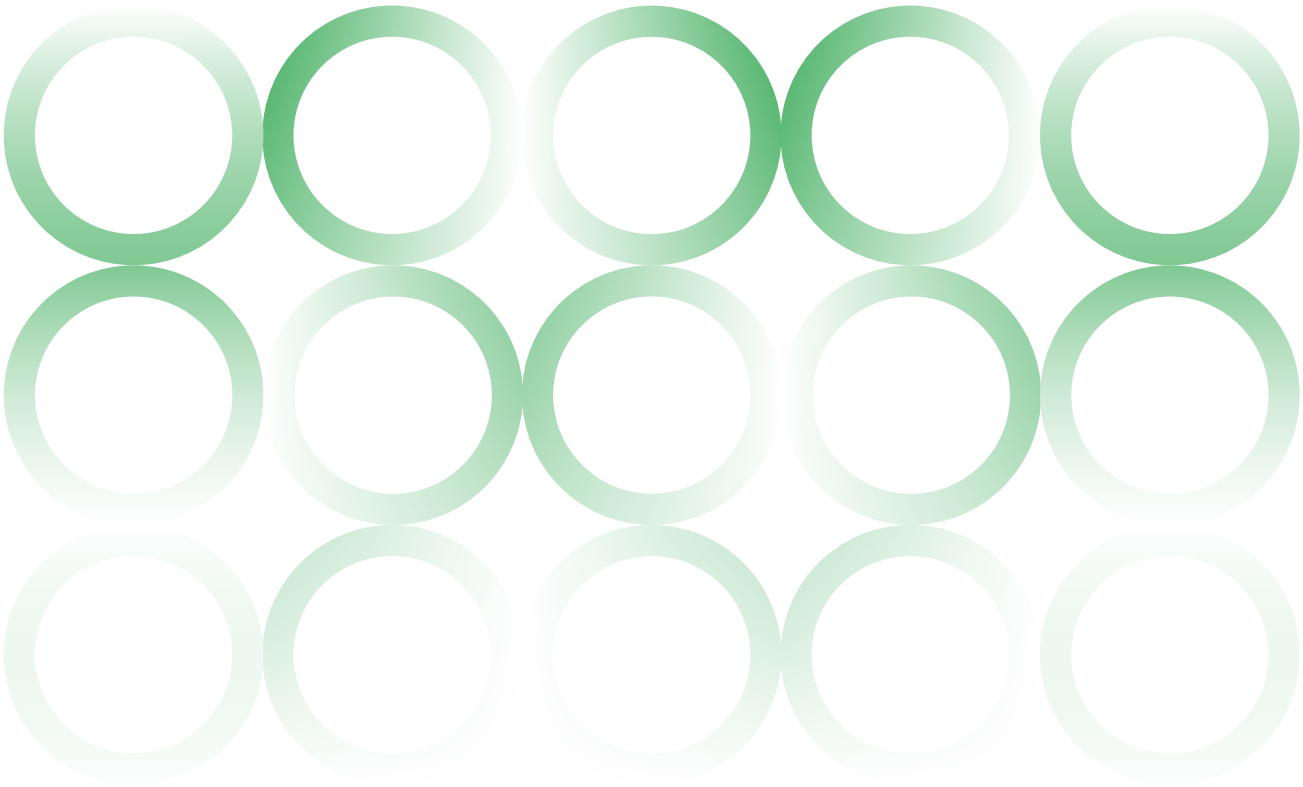


Figure 11: Infrastructure vs. Serverless Spark



Dataflow

Dataflow is Google's fully managed service for Apache Beam pipelines. Dataflow provides automated provisioning and management of compute resources. Furthermore, Dataflow unifies batch and streaming in a single programming model, so it is an ideal option for cases requiring some level of support for streaming. Use Dataflow templates to cut down development time by leveraging pre-implemented solutions for data import and export for most common data sources.

Dataflow processing is typically used as a dynamically scalable transform and ingest engine for streaming pipelines. Dataflow streaming jobs will read off a message bus such as Pub/Sub or Kafka, transform the data, and write it to a sink such as BigQuery. In addition, Dataflow can be leveraged as a scalable batch data ingest engine to BigQuery when preprocessing is required.

Compute Considerations

Google's analytics lakehouse processing stack includes a diverse set of tools to accommodate a wide variety of processing paradigms, languages, and end users. These tools are especially powerful because they are decoupled from storage and can read and write data where it is stored, avoiding the need to extract and load data into various systems and solutions.

Consuming/activating the data

Traditional reporting, with data in a warehouse, involves looking back at historical data over the past week, month, or quarter. While there is value in understanding these trends in the business, it is also important to use analytics to look forward, so that real-time actions can be taken to correct issues before or once they arrive. This section will cover how data is consumed and activated for analytical and data science use cases leveraging Google Cloud's tools.

BI Use Cases

There are a few ways to use the data that is stored in BigQuery, and the access method should be based on an end user's skill set. Meeting users at their level of data access, including SQL, python, or more GUI-based methods, means that technological skills do not limit their ability to use data for any job. Scientists may be working outside traditional SQL-based or BI-type tools. Because BigQuery has the storage API, tools such as Spark running on Dataproc or AI notebooks can easily be integrated into the workflow. The paradigm shift here is that the analytics lakehouse supports bringing the compute to the data, rather than moving the data around. In addition to the native BigQuery SQL engine, Figure 12 demonstrates other computation frameworks.

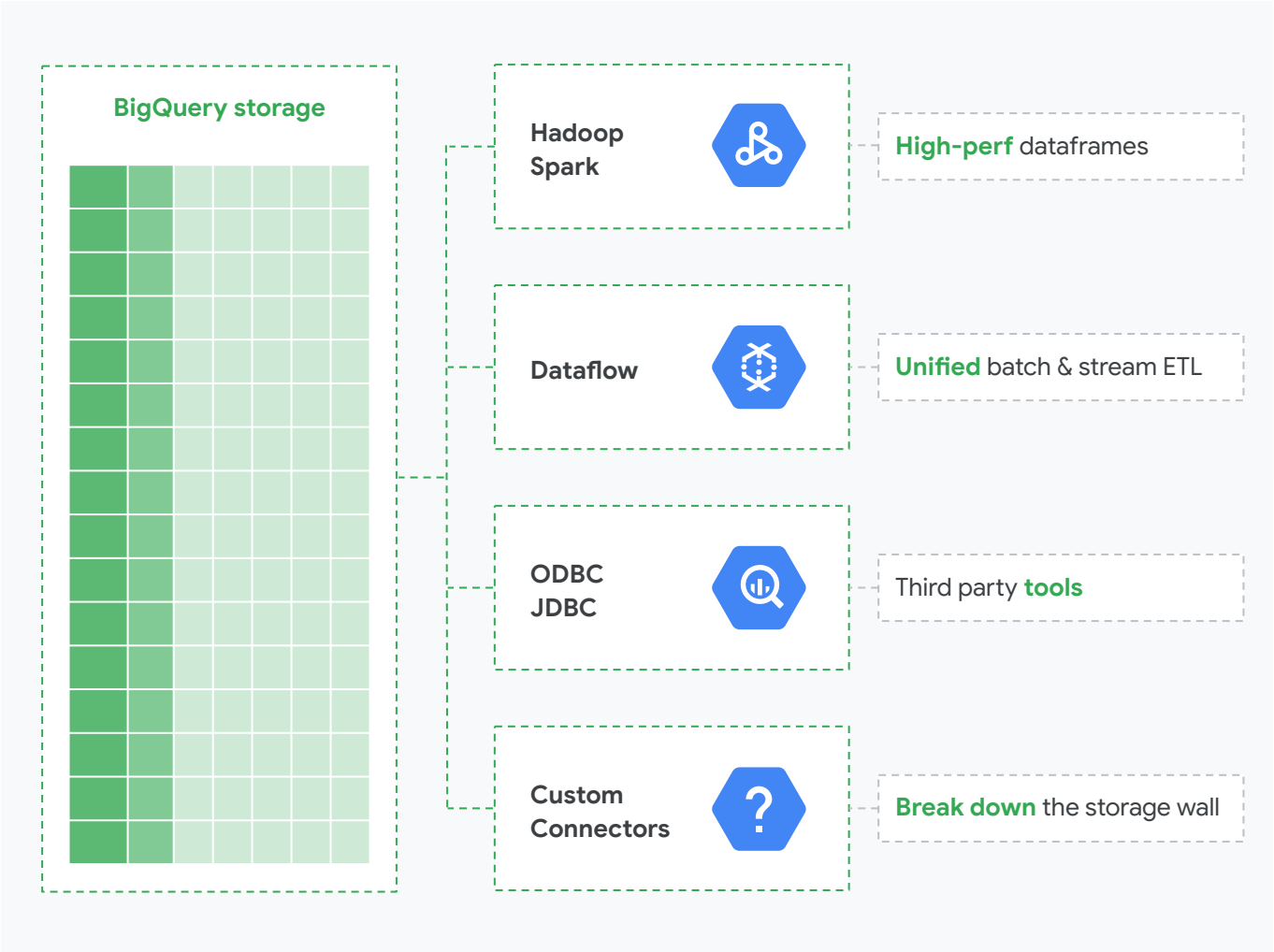


Figure 12: BigQuery Storage API

BigQuery ad hoc queries

When analysts are first working with data, they generally feel most comfortable using a familiar SQL interface. BigQuery has a native interface or can be leveraged with third-party tools for exploratory data analysis. This usually happens when analysts from different lines of business are looking for insights in the data that may end up in reports or applications. The ad hoc queries can be used as an interactive query incubator. Queries can be tested for speed and cost, then later codified in scheduled or automated batch queries.

Ad hoc queries are often performed in sandbox projects. In looking at the analytics lakehouse zones in BigQuery, ad hoc queries exit in the Curated Layer. These are virtual representations of the data logic (unless there are materialized views as part of the dataset) and can be used BigQuery on demand or with reserved slots.

The analytics lakehouse enables this because of the nature of the separated storage and compute. Individuals can be running queries in their own projects and/or sandboxes without the need to create another physical copy of the data.

BigQuery Storage API

BigQuery offers an API to use data stored in its managed service for things like Spark or Hadoop jobs. The BigQuery API is a structured parallel API to read the data at full speed and provides additional features like projection, filtering, and dynamic rebalancing. To query the data via the BigQuery API, the query is sent by making a direct HTTP request to the server which exposes a traditional JSON/REST interface. HTTP is a stateless protocol. REST services are well designed for web serving. JSON maps directly to in-memory objects in languages like JavaScript and Python. To interact with BigQuery in ways that are not directly in the console or CLI, it is recommended to use the client libraries (there are seven available languages for this) over an ODBC/JDBC driver because these drivers only offer a subset of BigQuery's capabilities.

One of the advantages of an analytics lakehouse built on BigQuery and Google Cloud is the flexibility and extensibility of the selected storage and compute frameworks. Using the BigQuery API allows for parity in the speed and capabilities when using client libraries rather than only using SQL or the console to interface with the data.

BI Tools

Nearly every organization will have both legacy and cloud-native BI tools with which analysts and other data users will interact with data. Analytics lakehouses are about enabling every user to be a data user, regardless of their technical skill set and capabilities. This is why it is important to have an analytics lakehouse that allows interactivity and connections to native and third-party BI tools.

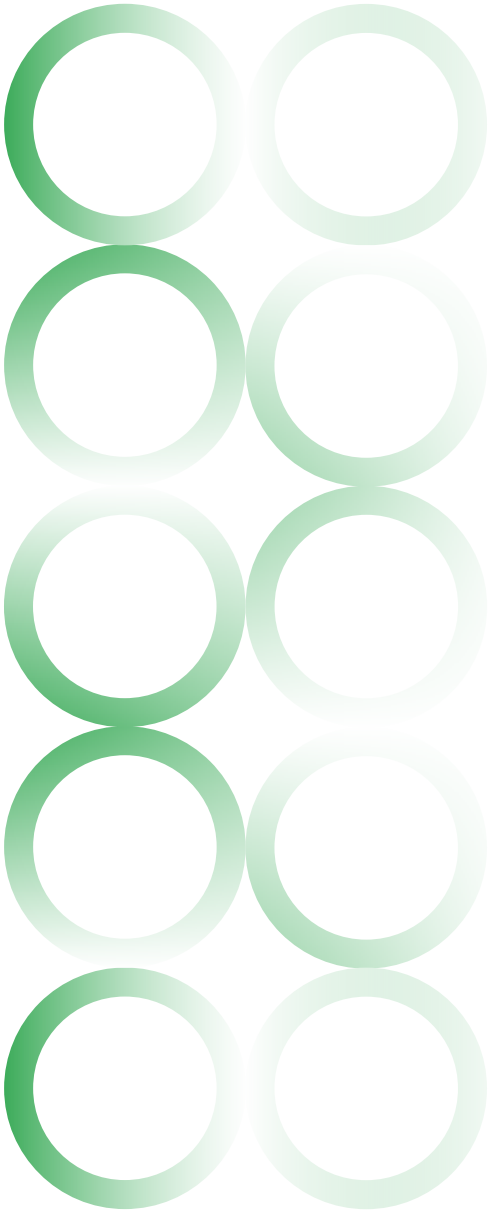
Looker / Looker Studio

Google Cloud offers a spectrum of options when it comes to visualization and reporting tools that are natively integrated to BigQuery and other Google Cloud services. With free tools like Looker Studio, Google Cloud sees self-service BI use cases as a key element for ensuring maximum value from the analytics lakehouse. With Looker Studio, teams can access all of Looker's governed data along with Looker Studio's existing 600+ data sources. Looker Studio Pro is a new paid version of Looker Studio that provides enterprise user management features, team collaboration features, support, and SLAs. It is offered as a GCP product, under the GCP terms of service.

Looker Studio can be used for "data sketching," a way to quickly visualize data interactions and understand the relationship between values. It can be accessed directly from a query result in BigQuery or by navigating to the tool. Looker Studio is lightweight compared to the enterprise version of Looker as it is more focused on not requiring a centralized data model or even just visualizing data from a spreadsheet.

Conversely, Looker unlocks valuable optionality when it comes to governance and building a trusted semantic model for exploration and reporting. It provides a unified semantic layer across data assets for repeatability and can be a datasource in Looker Studio. Looker enables teams to build data products and applications, which further supports consumption and activation of data from the lakehouse.

Security and governance will be covered in later sections of this paper, but it is important to note that Looker plays a key part and integrates with the standardized governance strategy that is designed for the analytics lakehouse. It has features like alerting, scheduling, and sharing — all of which create more value as more people are productively using the data. Because Looker was born in the cloud, it leverages the power of BigQuery or other databases to power the analytics, rather than copying a set of data to a separate server and running analytics in a siloed fashion.



Third party BI Tools

Because BigQuery acts like and is used as a warehouse for structured data, it can be convenient if database-agnostic APIs like Java Database Connectivity (JDBC) and Open Database Connectivity (ODBC) can be employed by a Java or .NET application to communicate with a BigQuery database driver. This is not recommended for new applications — use the client library instead. If, however, you have a legacy application that communicates with a database today and needs to be converted with minimal code changes to communicate with BigQuery, the use of a JDBC/ODBC driver might be warranted. The JDBC/ODBC drivers mean that the analytics lakehouse can interact with almost any existing database or applications as long as they enable these types of connections. Many organizations also use tools like Tableau or PowerBI for data visualization, which use the native BigQuery API. In evaluating tools for future use cases, it is worth noting that prioritizing the BigQuery API capability will ensure the highest functionality of how the data interacts with the additional BI/third party tools.

BI Engine

BigQuery BI Engine is a fast, in-memory analysis service to improve performance for BI and reporting tools. By using BI Engine, customers can analyze data stored in BigQuery with sub-second query response times using any BI tool that supports BigQuery and optimize their BQ computation cost using a combination of in-memory data and BigQuery slots. BI Engine also supports horizontal scaling for higher concurrency. In thinking about how BigQuery does smart caching overall, BI Engine offers the lowest latency for super-fast, self-tuning queries. Its architecture is simple, as BI Engine is essentially reserved hot machines that serve as a distributed in-memory execution engine. Because it is native to BigQuery, there is no need to build and manage online analytical processing (OLAP) cubes and it has an open API for partner integrations. Figure 13 is a diagram that highlights the simplified architecture of BigQuery using BI Engine.

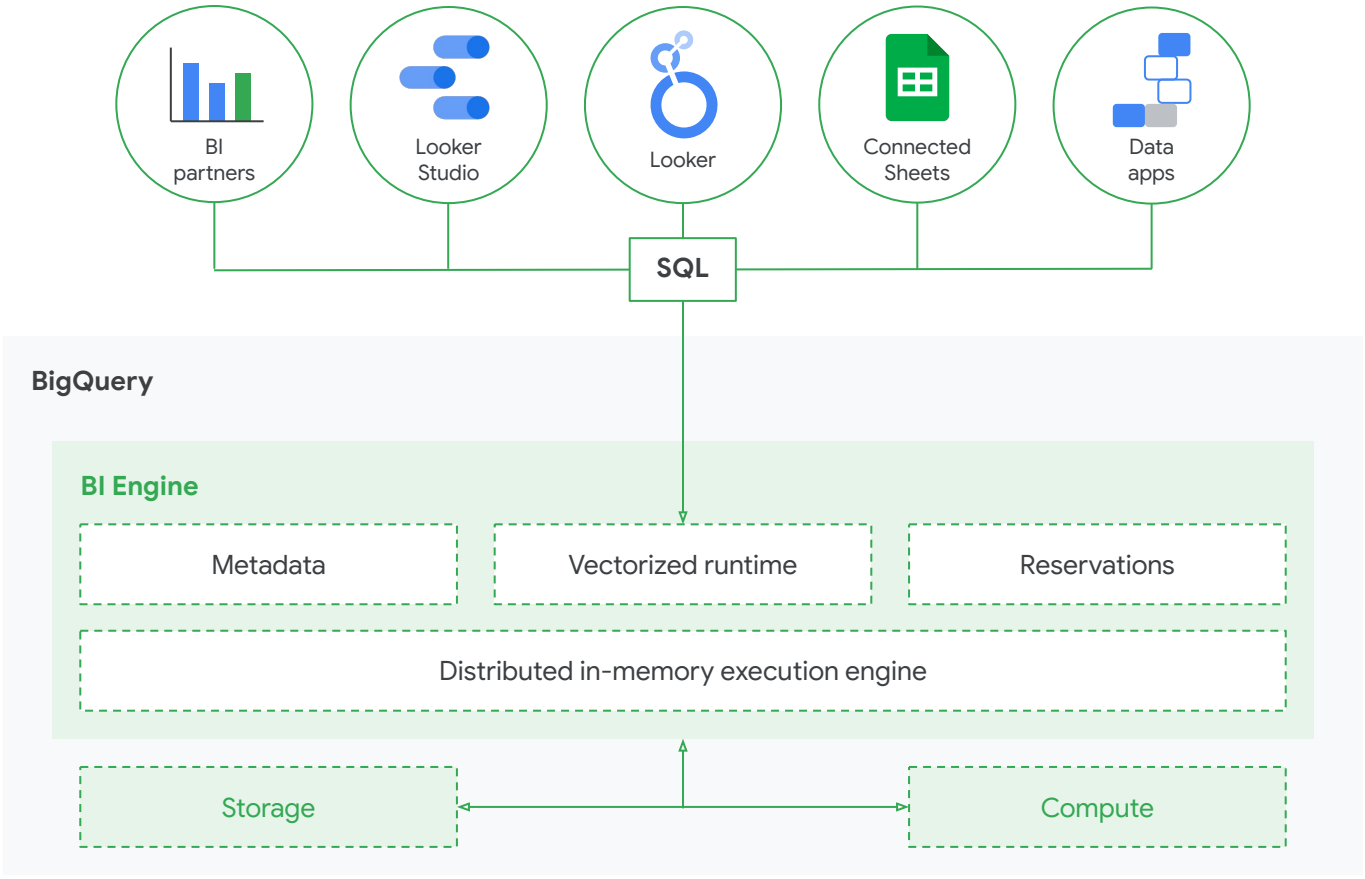


Figure 13: Simplified architecture of BigQuery using BI Engine

Machine Learning

ML has been and continues to be one of the most revolutionary opportunities for businesses to leverage their data for advanced analytics and predictive use cases. Google sees the democratization of this capability as a necessity for all organizations to benefit from these technologies. Building and deploying ML models is a key investment companies should continue to make in order to compete in the future of almost every industry.

Analytics lakehouses store data without the need to copy it several times, regardless of the use case. Data lakes were traditionally built for data scientists to have all the access to raw data they need to build models, but they were difficult and expensive to manage. Analytics lakehouses aim to minimize the burden, and Google Cloud provides tools to democratize the use of ML across every and any business.

BigQuery Machine Learning

BigQuery Machine Learning (BQML) is an SQL-based ML capability built into BigQuery that leverages the same powerful compute that powers analytics. Because storage and compute are decoupled, BQML makes building models simple and fast. Much like writing a stored procedure, a BQML model is built using a few lines of SQL. An end user just needs to structure the ML problem to essentially ask a question and know what kind of question they are trying to answer — whether it is a regression, classification, recommendation, clustering, or even unstructured data (like vision or language). BQML is a powerful tool for iteratively building a model without having to read the data into memory — a common problem when a dataset is too large. The model becomes available to serve immediately after it is trained because predictions can be made in the very tool in which the model is built. Essentially the entire ML process can be done in BigQuery, including hyperparameter tuning and feature engineering.

Recently, BQML has even started supporting models for unstructured data like documents and images. It essentially creates a powerful metadata table that points to a storage bucket where images or other data are stored and kicks off an AutoML training job all from the BigQuery interface. More about AutoML can be found in the following section, but what is important to note is how Google Cloud is working to seamlessly integrate every aspect of the data ingestion through activation workflow to give developers and users choices in how they want to accomplish their goals.

BQML is an important part of how users will activate data in an analytics lakehouse because it unlocks the capability for SQL developers and other analysts without the need to get a PhD in statistics. The models can be served directly from BigQuery, further breaking down barriers between data scientists and analysts.

Vertex AI and notebooks

Vertex AI is Google Cloud's unified ML and artificial intelligence (AI) platform and it is seamlessly integrated with the analytics lakehouse. It is where data scientists and developers can build, deploy, and scale ML models with fully managed tools. Within Vertex AI is a tool called AutoML — a low-code/no-code option to train models with minimal required expertise. This is a point-and-click tool that leverages the data stored in BigQuery to build models and create an end point to serve the models. Unlike BQML, which is an iterative way to build models, AutoML is focused on letting the tool fully automate the build process with no interaction required from the users. AutoML models, while low-code and fully automated, should not be confused in the quality of the models. It is often with ever increasing and high accuracy that models are built using this tool. See Figure 14.

Additionally, for data scientists looking to develop more custom models out of the analytics lakehouse, Google Cloud offers Vertex AI Workbench, a Jupyter Notebook-based development environment for the entire data science workflow. Other cloud services, such as Dataproc and BigQuery, can be accessed directly from the notebook, furthering the lakehouse's goal of breaking down data silos. Vertex AI notebooks provide a commonly used tool (Jupyter notebooks) that analysts and data scientists can leverage to easily interact with the data in the lakehouse. They can also process remarkably high volumes of data by running the notebook on a Dataproc cluster. Data scientists often work in a notebook environment to do exploratory data analysis, create visualizations, and perform feature engineering. Within a managed Workbench notebook instance in Vertex AI, you can directly access your BigQuery data with an SQL query or download it as a Pandas Dataframe for analysis in Python.

Like many features of the analytics lakehouse, there are several ways for the tools to connect and interact with the data. This includes data that is produced as predictions from ML models. BigQuery is a place to store predictions and do analysis of predictions for actuals as the models are released into production. The analytics lakehouse aims to provide the tools for each user in ways that do not require copies upon copies of data to be produced, but rather leverage the power of storage and compute in the format that makes the most sense for the use case.

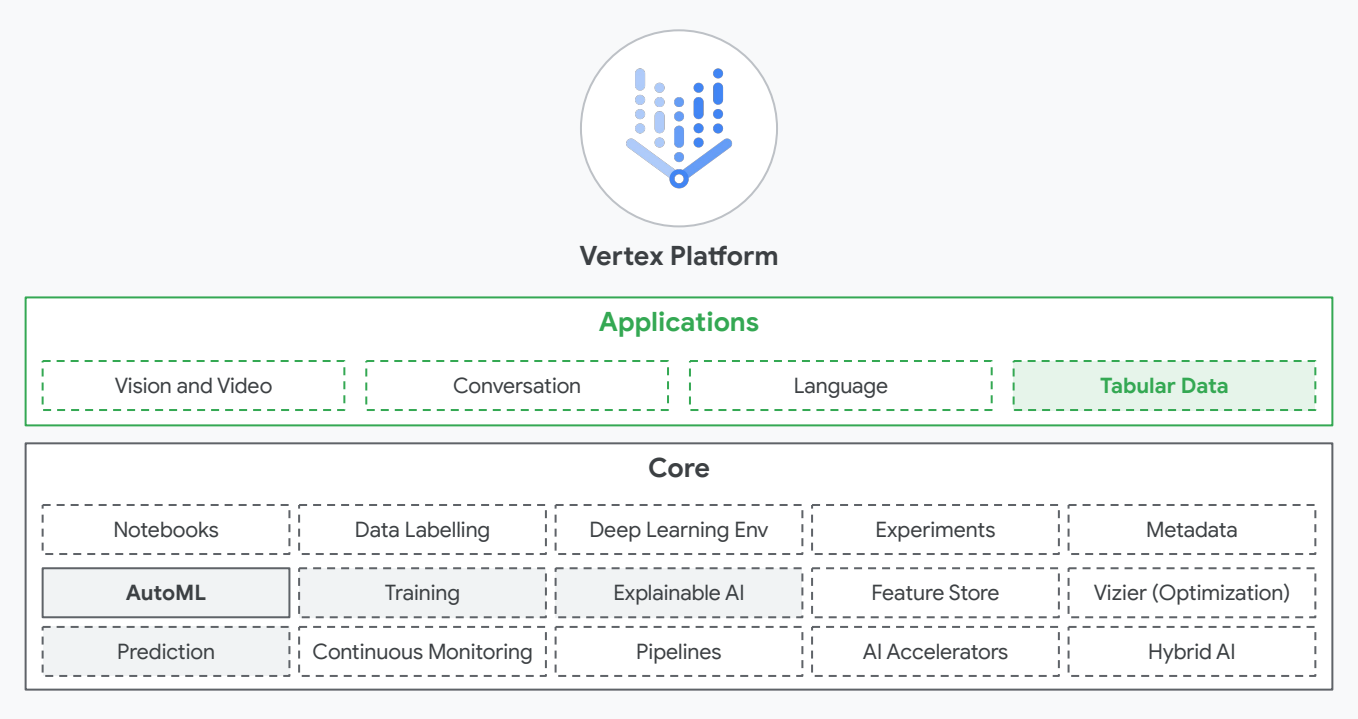


Figure 14: Vertex AI platform architecture

Governance

Ensuring an analytics lakehouse has clean, consistent, documented, and secure data is essential as organizations leverage the lakehouse for critical business operations and insights. Mature data organizations adopt and integrate rigorous processes, tools, and people to enable governance. The Google analytics lakehouse provides tooling to simplify governance by enabling features such as centralized administration, data cataloging, consistent orchestration, and enabling CI/CD for change control.

Dataplex

Dataplex is an intelligent data fabric that unifies distributed data storage solutions and automates data management and governance to power analytics at scale. Dataplex provides metadata-driven data management with built-in data quality and governance capabilities.

Historically, managing an analytics lakehouse with data in a data warehouse and object storage meant having multiple teams to manage both BigQuery permissions and Google Cloud Storage permission. Having two separate teams apply permissions to two different storage systems with differing levels of permission granularity meant that the level of access was often inconsistent across solutions. Dataplex was built to address this challenge and provides an integrated administration framework to apply permissions to BigQuery and Google Cloud Storage objects in a unified and consistent manner.

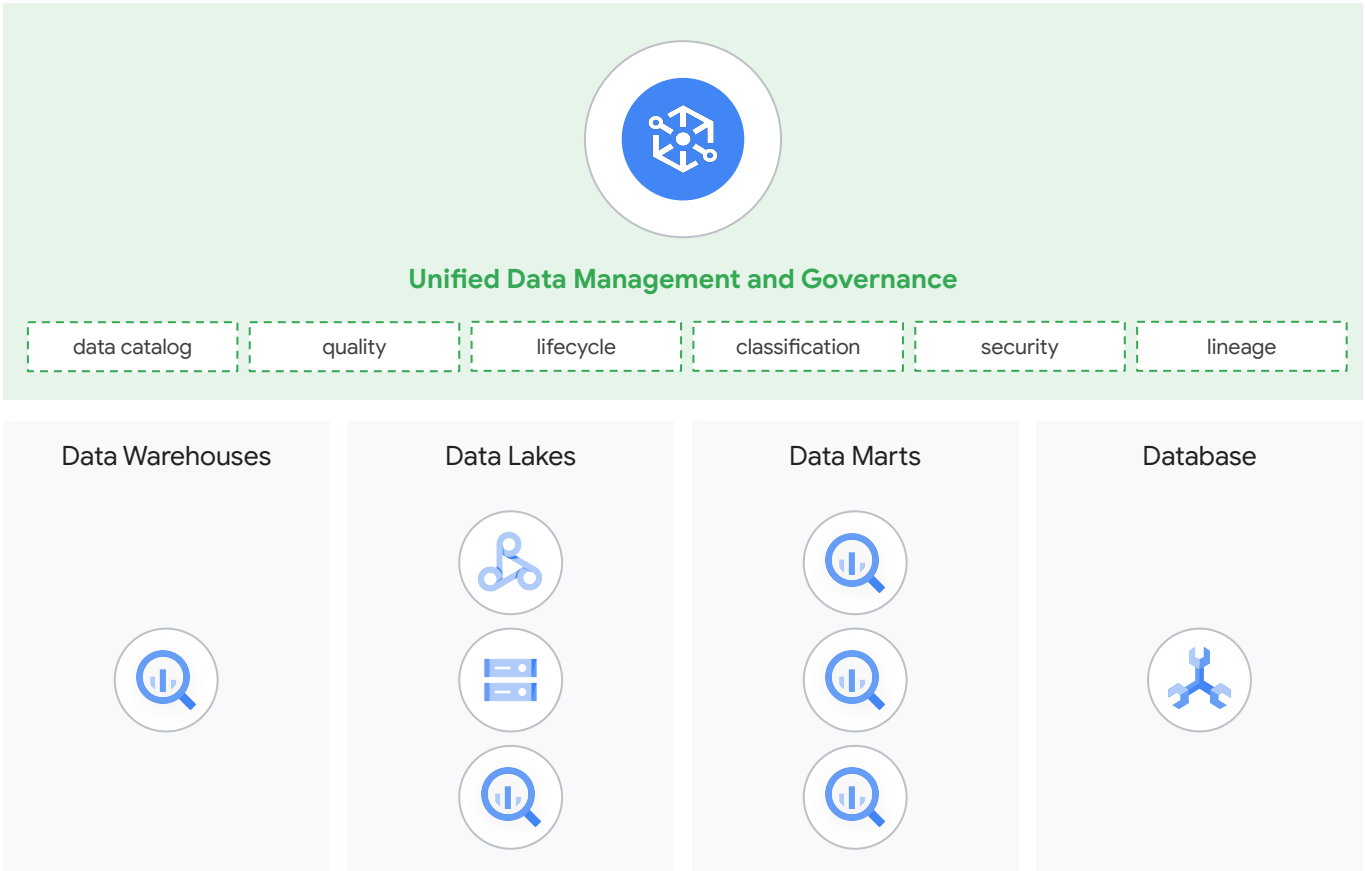


Figure 15: Unified Data Management and Governance with Dataplex

Dataplex enables you to build virtual "lakes" that consist of "zones" and "assets." A lake can be a single subject area or group of subject areas for which a data engineering team is responsible for building data ingestion pipelines. A zone correlates to the level of processing performed on the data and is analogous to the lakehouse storage zones of Raw, Enriched, and Curated. Each zone has various assets including BigQuery Datasets and Google Cloud Storage buckets. User groups can be mapped to zones for consistent application of permissions. Figure 16 is an example of a lake with zones, assets, and users.

Building separate zones for data product domains enables a data mesh-style approach to building a decentralized data architecture. This allows teams to build data products across projects and data storage systems without any additional movement to bring the domains together.

Dataplex removes administrative burden by automating metadata discovery and makes incoming data assets in cloud storage securely accessible and available for querying via BigQuery as external tables and open-source applications such as SparkSQL, Presto, and HiveQL.

Lastly, Dataplex provides a set of templates to help organizations build and run common data quality and data lifecycle management tasks, including Serverless Spark tasks.

In summary, Dataplex is not the "lakehouse," but Dataplex brings together multiple services and templates to make the lakehouse easier to build, manage, explore, and govern.

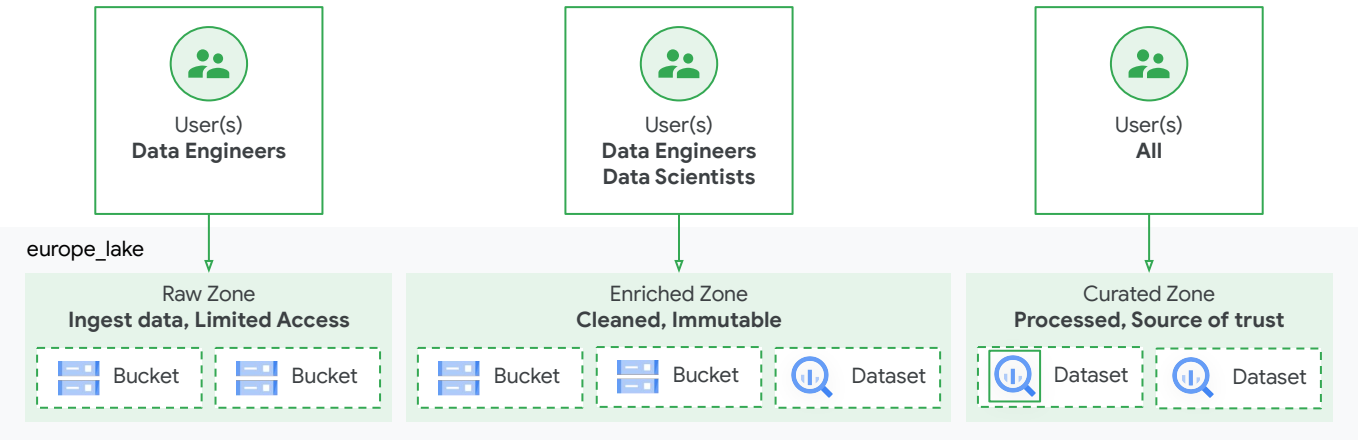


Figure 16: Example of a lake with zones, assets and users

Data Catalog

Data Catalog is a fully managed, scalable metadata management service within Dataplex. Data Catalog is API driven and will discover assets, tag data entities with metadata, and help secure data within BigQuery through Column Level Security and Dynamic Data Masking.

Discovering data assets will happen automatically with Google Cloud sources such as BigQuery, Cloud Storage, Pub/Sub, and Dataproc Metastore. When new data is found, technical metadata is captured and added to the Data Catalog. Once technical metadata is in the Data Catalog, users can bring business metadata and tags to enrich the meaning of the data and provide a framework for granular security application. Once data is discovered by the data catalog, tags may be applied either manually or programmatically to help categorize data.For example, when ingesting new tables, you may choose to apply tags to PII columns to restrict access to this sensitive data by non-authorized users. In addition, the Google Data Loss Prevention product can be configured to scan inbound data for sensitive data and apply tags to the Data Catalog based on the contents of the fields. This is especially useful for guarding against sensitive data being added to free form fields such as “notes” fields.

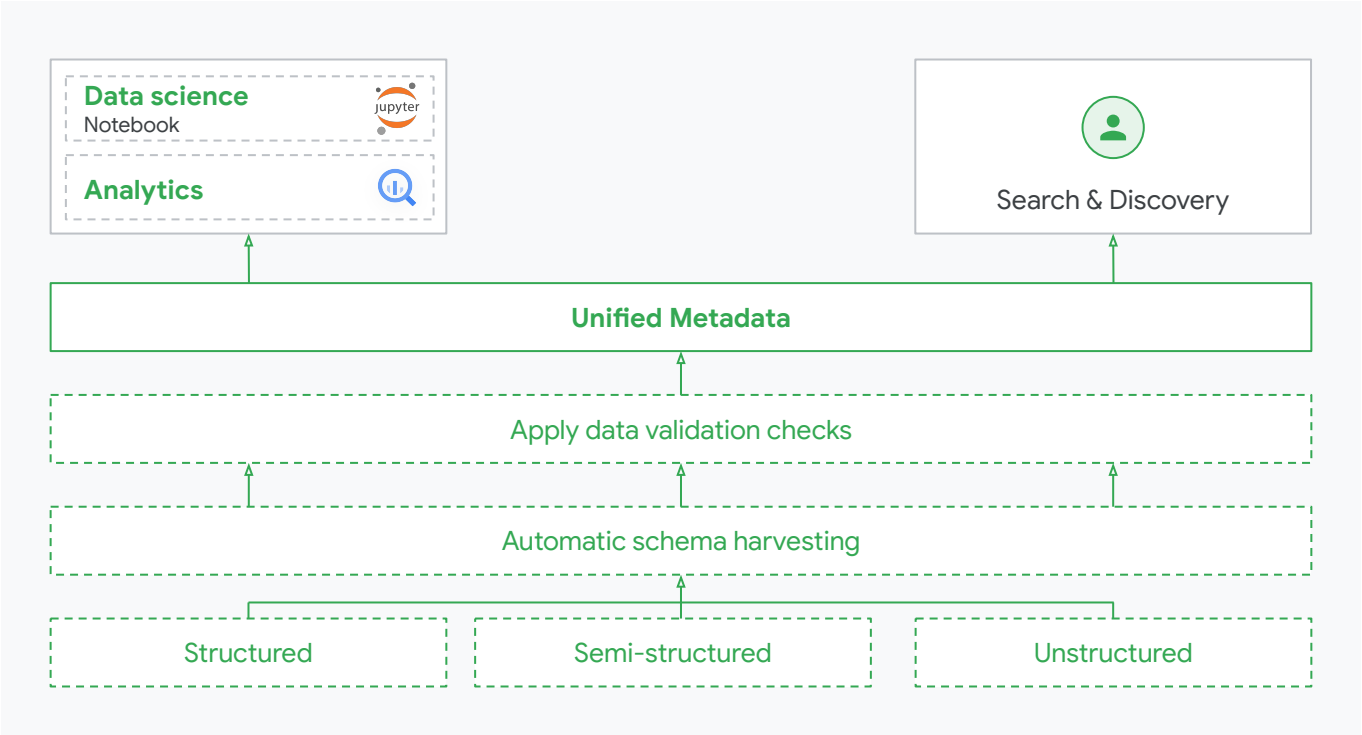


Figure 17: Data Catalog

Access

Security

Securing data in the lakehouse can be one of the most important yet challenging tasks when building a lakehouse. Securing a lakehouse is complicated due to the wide heterogeneity of data storage mechanisms, varying levels of access controls for each data storage mechanism, and the limitations of tools to effectively handle granular access controls.

Securing data in a data warehouse such as BigQuery is a relatively easy endeavor as BigQuery supports complex permissioning at the service, job, dataset (schema), table, column, and row along with built-in Dynamic Data Masking. This allows for application of fine-grained access controls to help secure sensitive data assets.

Securing data in object storage, such as Google Cloud Storage, can be more challenging. Data in Cloud Storage can be secured at the bucket, folder, and file level but is complicated by the wide variety and structure of data that can be placed in a Cloud Storage bucket. For example, you may have an extract in Parquet format that contains structured data but may have hundreds of physical files that make up one logical object. When securing this data you would likely secure the entire folder or bucket of files together with a set of ACLs. This is effective but can be imprecise. What happens if you want to give the marketing team access to the Parquet file but want to omit two columns of sensitive data? Historically, you would secure this data by creating a copy of the file without the sensitive columns, loading the data into the warehouse to apply fine-grained access control, or trying to control access at the application layer. While mostly effective, each of these mechanisms has a tradeoff, including data duplication, data latency, and access rules applied in too many locations (storage, database, application, etc.).As mentioned in the Data Storage section, the BigLake tables feature was developed for the lakehouse to specifically address the issue of fine-grained access control to data in Cloud Storage.

BigLake allows administrators and data engineers to leave the data in its original format (Parquet, Iceberg, ORC, Avro, CSV, JSON) while enabling database-style security granularity including dataset, table, row, column, and Dynamic Data Masking. This is enabled through the BigQuery Storage API and makes these objects available in BigQuery for processing with SQL but also makes these objects available to other lakehouse workloads such as Spark, Presto, and Trino. In summary, having diverse data stored in a variety of locations can pose challenges when trying to apply a consistent level of permissioning on your data assets. BigLake tables helps solve this problem by providing database-style fine-grained access controls on top of open standard file format data residing in Cloud Storage.

Observability

With a wide range of services and data storage options, observing who accessed what data and when can pose a significant challenge when managing an analytics lakehouse. To that end, Google has implemented the Google Cloud Operations Suite, including Cloud Logging and Cloud Monitoring. BigQuery Information_Schema tables can also help administrators understand how their environments are being utilized.

Cloud Logging

Cloud Logging is a fully managed service that allows you to store, search, analyze, monitor, and alert on logging data and events from Google Cloud. Logs are generated in response to particular events or actions. Each log is an append-only collection of log entries and many Google Cloud services, including BigQuery, create a type of log called an audit log. These logs record events such as administrative activity, data access, and system events. Audit logs are written in a consistent structured JSON format. Raw logs can be accessed via cloud logging, which includes a robust set of filtering and querying to help identify anomalies. In addition, Cloud Logging data can be written to BigQuery to enable additional analytics and dashboarding.

Having a consistent and granular logging framework across services gives administrators and auditors the detailed information required to monitor, alert, and govern the analytics lakehouse.

Cloud Monitoring

Cloud Monitoring provides predefined service-level dashboards and metrics based on Cloud Logging data. While Cloud Logging is great at providing detail-level data, Cloud Monitoring can help make sense of the data by enabling the exploration of metrics, generation of alerts, building of uptime checks, and grouping of resources to understand end-to-end application health.

While Cloud Logging provides the detailed data, Cloud Monitoring provides the tooling and templates to get started building, reporting, and alerting for your analytics lakehouse.

INFORMATION_SCHEMA

The BigQuery Information_Schema (IS) tables are read-only, system-defined views that provide metadata information about BigQuery objects and activity.

The most commonly used IS tables are the jobs_by* views that contain real-time and historic job information (queries, loads, scripts, etc.) with a retention period of 180 days. This data is accessible at the user, project, folder, or organization level and is frequently used as a starting point for query tuning. For example, a BigQuery administrator can write simple queries against the IS views to aggregate bytes_billed to see the highest consuming user in a project or find queries for a folder that take more than two minutes to run. In short, if you are looking for historical utilization at the job level, this is your source.

The object-specific IS views (Datasets, Tables, Columns, Partitions, Views, Routines, etc.) can be leveraged to obtain metadata and consumption information for database objects. For example, you can leverage the table_storage* views to understand the size of a table, or the partitions* views to understand how many partitions are in a table and the size of each partition. Overall, these objects can help the administrator audit, monitor, and automate tasks in the lakehouse (BigQuery + BigLake) environment.

In summary, your Google lakehouse has wide and deep logging capabilities to help you understand how your lakehouse is being used and by whom. These services provide out-of-the-box monitoring, dashboarding, and alerting and offer a foundation to build custom metrics based on governance needs.

Composer

Google's Cloud Composer is a fully managed Apache Airflow solution that allows users to build and orchestrate data pipelines across services, from on-prem to cloud and across clouds. While not strictly a data governance tool, Google's Cloud Composer can help organize your data orchestration pipelines into a scalable tool to support DevOps.

With Cloud Composer you can instantiate fully managed Airflow environments and leverage them to author, schedule, and monitor pipelines across services. Airflow pipelines, called Directed Acyclic Graphs (DAGs), are authored in Python and Google has provided over 150 Airflow operators to simplify the interaction with various Google Cloud services. Airflow operators contain the logic of how data is processed in a pipeline and how the pipeline interacts with other services. For example, you can create a DAG that has the following steps: extract data from Cloud SQL, spin up a Dataproc cluster, submit a Dataproc job to transform the extracted data, delete the Dataproc cluster, and load the data into BigQuery. Each one of those steps can be expressed with a Google provided operator, saving users the time and hassle of building integration points with each service.

It is Cloud Composer's ability to orchestrate pipelines across a variety of internal and external services that makes it a key component of the analytics lakehouse.

Dataform

transformation pipelines through a common SQL interface and is an excellent tool to build BigQuery transformation steps. Dataform leverages SQLX to define variable-driven SQL files in which you define your workflow, dependencies, and data quality checks. Dataform leverages third-party Git provider repositories to deliver full version-control support. For ease of use, Dataform is embedded in the BigQuery Google Cloud Console with a GUI designed for ease of development and deployment.

Dataform workspaces can be compiled from SQLX into executable SQL and executed via Google Cloud Workflows and Scheduler or through Cloud Composer for deeper integration with surrounding services such as load and extract jobs.

While not strictly governance, Dataform adds reusability, CI/CD rigor, and data quality checks to your analytics lakehouse transformations while leveraging the serverless and scalable nature
of the BigQuery SQL engine.

Governance Considerations

Enabling your organization to develop, deploy, monitor, audit, and secure data products is a crucial step toward building trust in and activating your lakehouse. Google provides a robust set of services and features to help simplify your processes and enable your organization to spend more time analyzing data and less time on maintenance and administration.

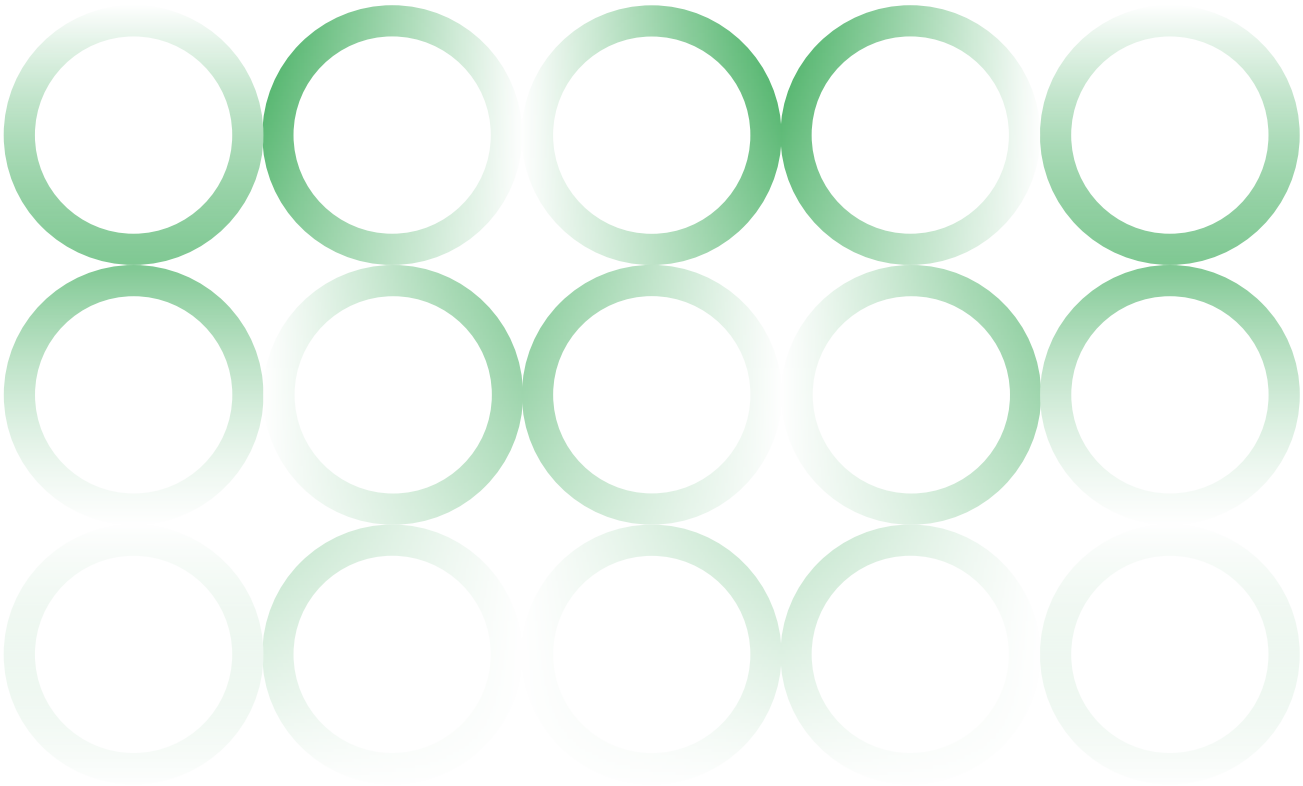
Who uses a lakehouse?... Everyone!

The intention of the analytics lakehouse is to make sure that every user becomes a data user in a way that aligns with their technical capabilities and enables productive and secure use of enterprise data. Analysts may use data to build and share dashboards in Looker, while data warehouse teams might focus on the SQL development of modeling the data and managing resources.

There are several roles and teams that will use or support the analytics lakehouse in any organization. The following table highlights the key roles and tasks for ensuring a successful analytics lakehouse. These might not all align with how your teams operate but are intended to provide a framework for operating an analytics lakehouse at scale. See Figure 18.

There are a few key personas that we will focus on in the following sections, highlighting how they would support the analytics lakehouse operations and value.

The primary roles of the analytics lakehouse will be those directly connected with data operations and analysis, including data engineers, data analysts, and data scientists. The key piece of the analytics lakehouse is that the tools and methods required for each of these roles is part of the same architecture. There is no need to build something different for data analysts and data scientists, even though the ways they each use data may differ. In this way, the analytics lakehouse really is built for all users.



Role	Responsibilities	Skills	Workload
Platform Admin	Maintains infrastructure hygiene Logging/monitoring and observability components	Cloud infrastructure administration	Site reliability engineering
Data Warehouse Admin	Designs and develops enterprise data warehouse	SQL development ELT using SQL	Dimensional modeling of transactional data
Data Engineer	Develops and maintains lakehouse infrastructure comprising of ingestion pipelines, ELT/ETL scripts, CI/CD with guardrails Develops and maintains workflows to integrate pipeline steps	Data engineering Software engineering	Creates ingestion pipelines Creates ETL/ELT scripts Designs and maintains CI/CD practices Standardized and reusable workflows
Data Governance Specialist	Defines and ensures data quality policies Defines and ensures data IAM compliance Defines and ensures data security standards Performs data audits	Data stewardship Data security and data compliance SME	Data privacy regulations/standards implementation Metadata standards
Data Analyst	Searches for, identifies, and evaluates cross-domain datasets to create a business intelligence framework	Analytics Engineering SQL development	Creates and maintains clean, curated, and aggregated datasets for visualization teams to consume Defines best practices for creating analytics datasets
Data Scientist	Data discovery Feature engineering ML model creation, training, testing, and deployment to optimize business domain processes	Data wrangling Machine learning Deep learning	With business stakeholders, translate business-driven needs into testable hypotheses, make sure that value is derived from machine learning workloads, and create reports to demonstrate value from the data
BI / Data visualization Specialist	Translate business requirements into system requirements Create and monitor reports that describe strategic business goals	Data visualization Requirement analysis	Dashboard development

Figure 18: Key roles and tasks for ensuring a successful analytics lakehouse

The primary roles that use the analytics lakehouse will likely be the data analysts, the data engineers, and the data scientists. Each of these roles have a optionality around which tools they can leverage to do their jobs most effectively.

Data engineers have a special role, often overlooked, though critical to successful data operations. They build data pipelines that make data available to the other users of the analytics lakehouse.

As noted in the previous table, the data analysts is focused on answering questions about the business using several datasets. They are often great SQL-izers, and will use BigQuery's robust SQL support to perform this analysis, potentially leveraging a data visualization tool to build and share the results in reporting use cases.

Finally, data scientists have a role similar to that of data analysts, but rather than answering specific questions, they are focusing on leveraging data to find questions users don't know to ask. They use data to build predictive models, and keep the business looking to the future.

Data engineers

Data engineers are the team that helps the data get from source to sink. They will partner with the lines of business to ensure that the data they need to run their operations is available in the analytics lakehouse. They will use the ingestion and transformation tools to bring the data together and move it across the different types of storage and layers. Some organizations have the data engineers sit closer to IT, while others have them sit closer to the lines of business. Regardless of where they sit in an organization, the analytics lakehouse will often end up empty or swampy without the expertise of this team.

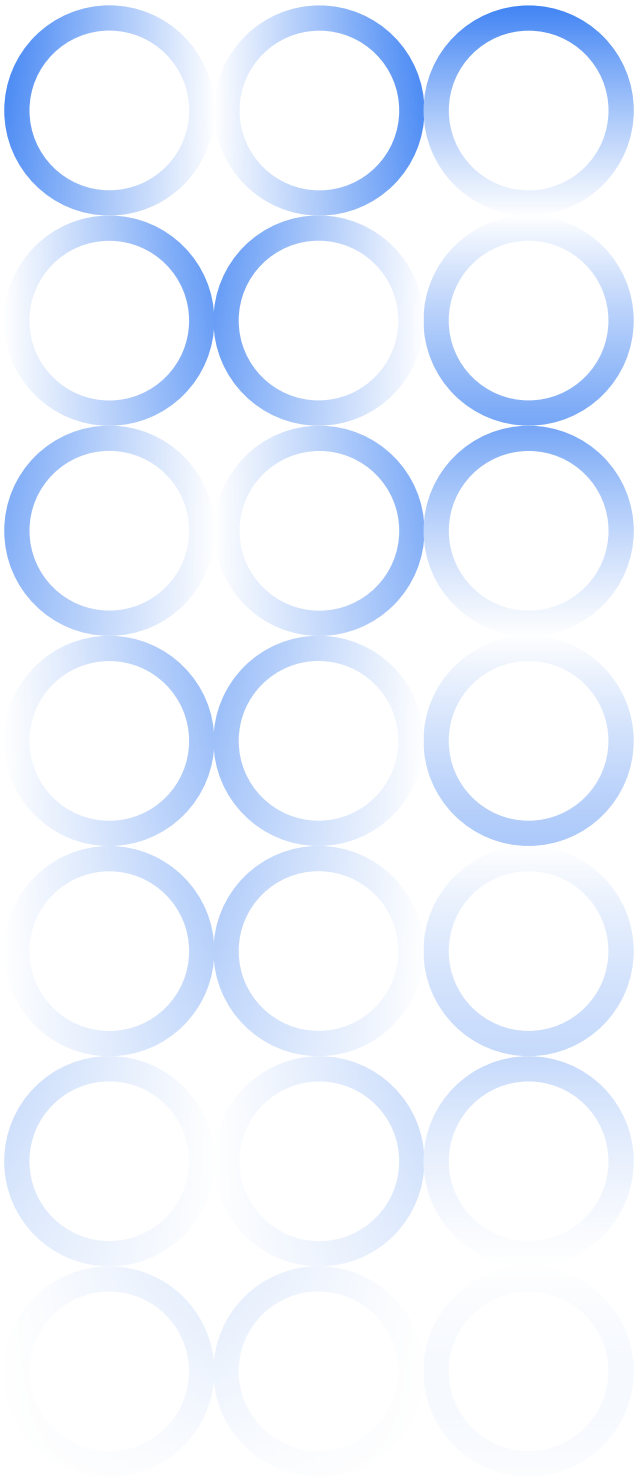
Data analysts

Data analysts are some of the primary users of the analytics lakehouse. They focus on deriving value from data stored in the lakehouse, using the set of tools that aligns most closely with their skills and the business' needs. Data analysts may structure and finesse datasets in BigQuery or use Looker to analyze and share insights. With the advent of AutoML and BQML, they may even start to blur the line between data analysts and scientists.

Data scientists

Data scientists or ML engineers should primarily focus on building, deploying, and scaling ML models. In many cases, when a good analytics lakehouse is not architected, these teams will spend too much time preparing or curating datasets for training and building models. These personas should not have to focus on that, but rather spend time building and scaling their own operations.

While every organization will have slightly different roles and responsibilities, it is important that these teams either have the ability to build and deploy software using CI/CD pipelines for their models, or partner with the correct development teams to productionalize the models. This will be a space where value is either lost or gained from the analytics lakehouse. This is one set of end users that will be consuming data, and the analytics lakehouse should make that simple, achievable, and agnostic to the tools they choose to use.



Chapter 4

Bringing it all together

We are in a transformative era for data in the Cloud. As data volumes increase and companies become more data driven, they need to break down data silos and make data more accessible to numerous users across the business. There is an increased need for the analytics lakehouse to meet the evolving demands of a data-driven world. With flexibility and planet-scale, Google Cloud is uniquely positioned to provide an open, dynamic, and future-ready analytics lakehouse that organizations and users will find valuable.

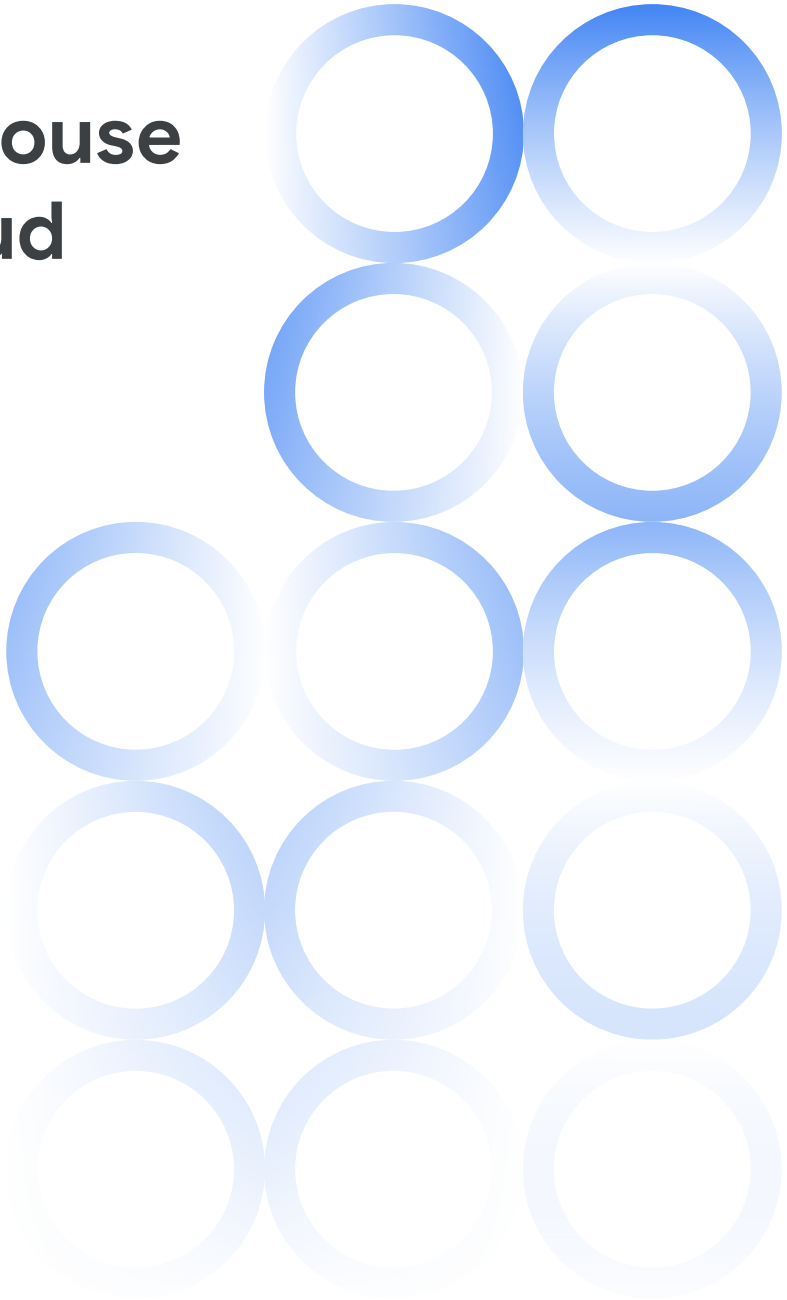
Google is a data company with a world-class suite of analytics products. But our secret sauce lies in our planet-scale, intelligent infrastructure upon which our products are built. Not only can you develop a lakehouse that meets your data users' needs, but we have you covered when it comes to unique hardware and networking, integration that enables streaming at unlimited scale, a serverless data platform with an unmatched 99.99% uptime SLA, and flexible and intelligent compute that takes the guesswork out of provisioning servers.

Building an analytics lakehouse on Google Cloud

April 2023

Interested in getting started?

Contact us to learn more.



Google Cloud