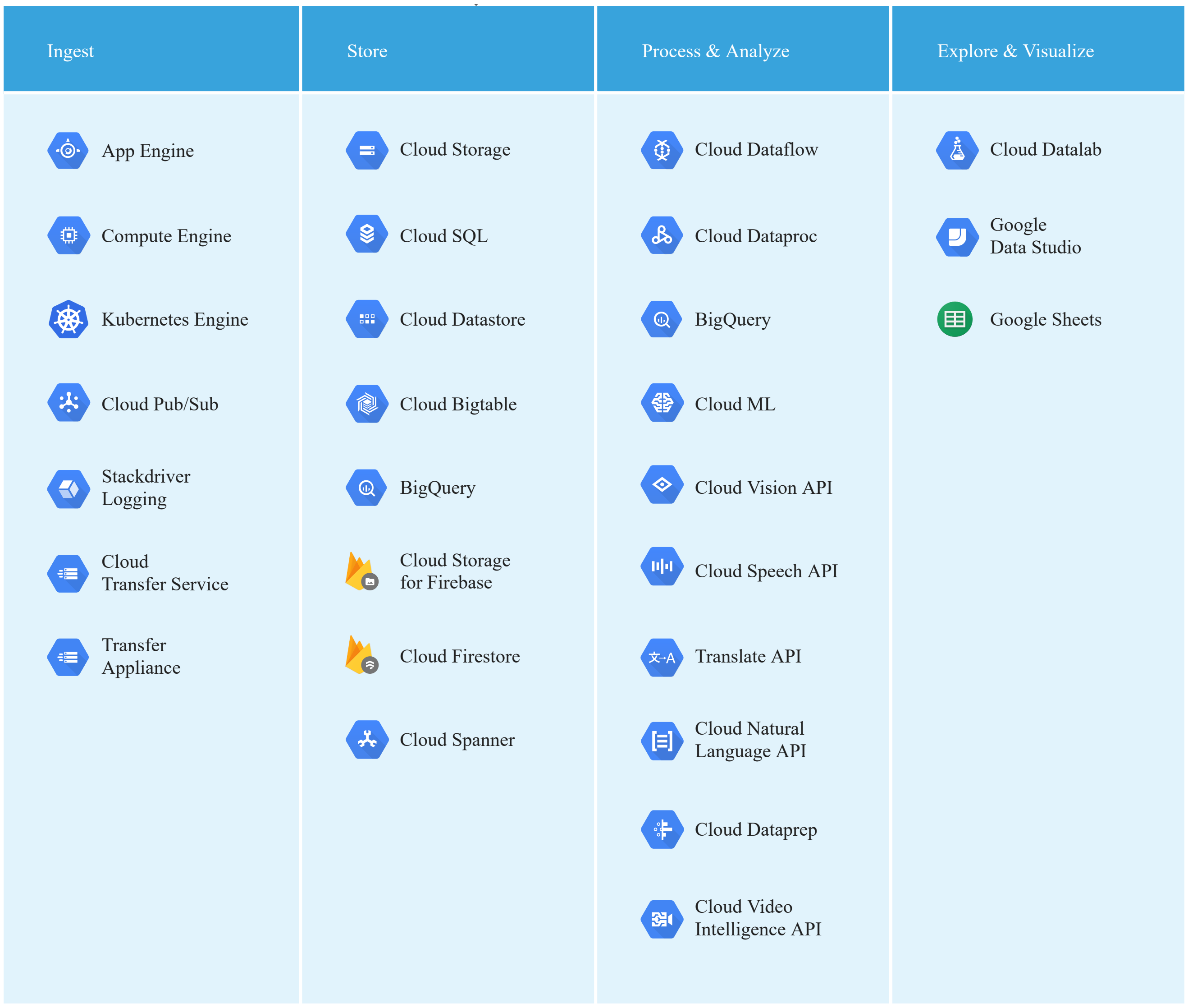
Data lifecycle

This article describes Google Cloud services you can use to manage data throughout its entire lifecycle, from initial acquisition to final visualization. You'll learn about the features and functionality of each service so you can make an informed decision about which services best fit your workload.

The data lifecycle has four steps.

- **Ingest**: The first stage is to pull in the raw data, such as streaming data from devices, on-premises batch data, app logs, or mobile-app user events and analytics.
- **Store**: After the data has been retrieved, it needs to be stored in a format that is durable and can be easily accessed.
- **Process and analyze**: In this stage, the data is transformed from raw form into actionable information.
- **Explore and visualize**: The final stage is to convert the results of the analysis into a format that is easy to draw insights from and to share with colleagues and peers.

At each stage, Google Cloud provides multiple services to manage your data. This means you can select a set of services tailored to your data and workflow.



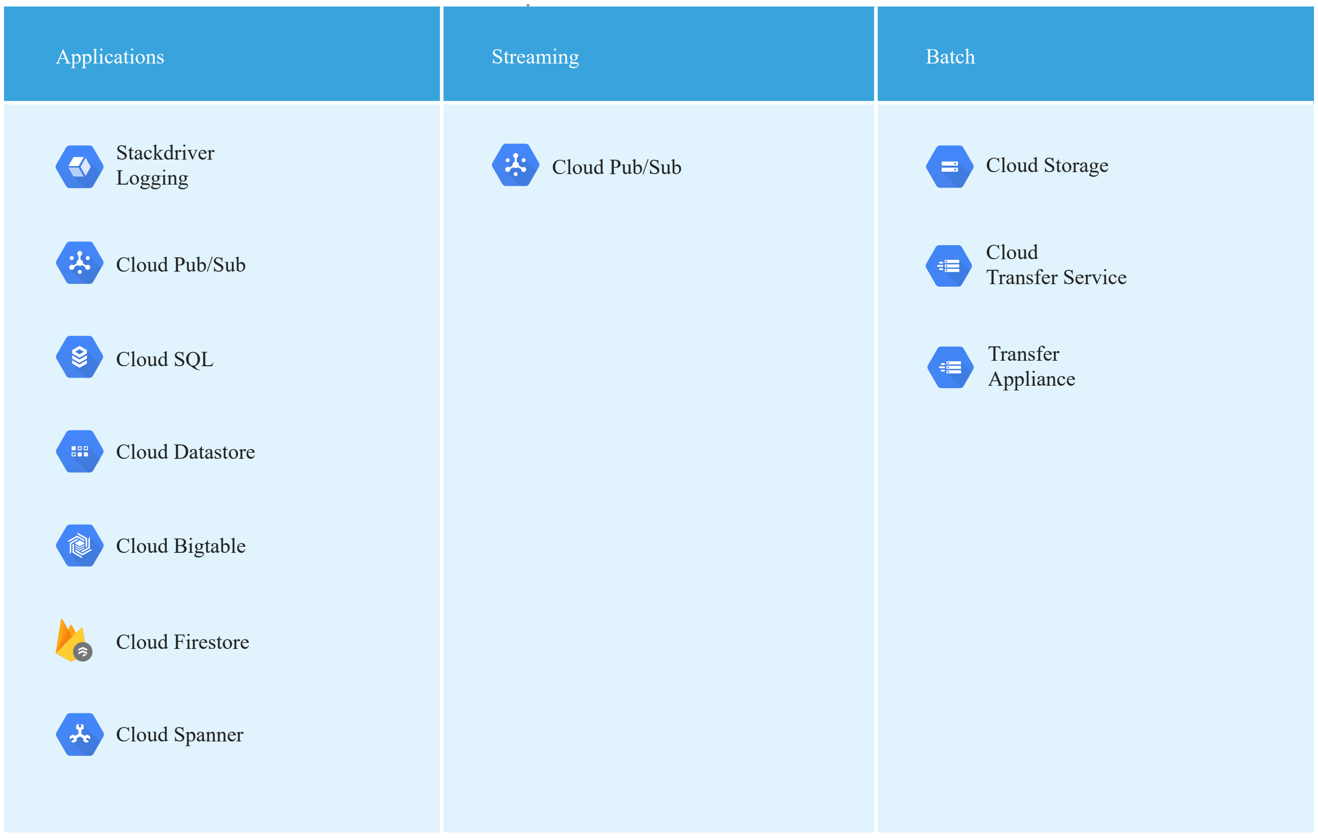
Ingest	Store	Process & Analyze	Explore & Visualize
App Engine	Cloud Storage	Cloud Dataflow	Cloud Datalab
Compute Engine	Cloud SQL	Cloud Dataproc	Google Data Studio
Kubernetes Engine	Cloud Datastore	BigQuery	Google Sheets
Cloud Pub/Sub	Cloud Bigtable	Cloud ML	
Stackdriver Logging	BigQuery	Cloud Vision API	
Cloud Transfer Service	Cloud Storage for Firebase	Cloud Speech API	
Transfer Appliance	Cloud Firestore	Translate API	
	Cloud Spanner	Cloud Natural Language API	
		Cloud Dataprep	
		Cloud Video Intelligence API	

Ingest

There are a number of approaches you can take to collect raw data, based on the data's size, source, and latency.

- **App**: Data from app events, such as log files or user events, is typically collected in a push model, where the app calls an API to send the data to storage.
- **Streaming**: The data consists of a continuous stream of small, asynchronous messages.
- **Batch**: Large amounts of data are stored in a set of files that are transferred to storage in bulk.

The following chart shows how Google Cloud services map to app, streaming, and batch workloads.



Applications	Streaming	Batch
Stackdriver Logging	Cloud Pub/Sub	Cloud Storage
Cloud Pub/Sub		Cloud Transfer Service
Cloud SQL		Transfer Appliance
Cloud Datastore		
Cloud Bigtable		
Cloud Firestore		
Cloud Spanner		

The data transfer model you choose depends on your workload, and each model has different infrastructure requirements.

Ingesting app data

Apps and services generate a significant amount of data. This includes data such as app event logs, clickstream data, social network interactions, and e-commerce transactions. Collecting and analyzing this event-driven data can reveal user trends and provide valuable business insights.

Google Cloud provides a variety of services you can use to host apps, from the virtual machines of Compute Engine (/compute/docs/), to the managed platform of App Engine (/appengine/), to the container management of Google Kubernetes Engine (GKE) (/kubernetes-engine/).

When you host your apps on Google Cloud, you gain access to built-in tools and processes to send your data to Google Cloud's rich ecosystem of data management services.

Consider the following examples:

- **Writing data to a file**: An app outputs batch CSV files to the object store of Cloud Storage (/storage/). From there, the import function of BigQuery (/bigquery/), an analytics data warehouse, can pull the data in for analysis and querying.
- **Writing data to a database**: An app writes data to one of the databases that Google Cloud provides, such as the managed MySQL of Cloud SQL (/sql/) or the NoSQL databases provided by Datastore (/datastore/) and Cloud Bigtable (/bigtable/).
- **Streaming data as messages**: An app streams data to Pub/Sub (/pubsub/), a real-time messaging service. A second app, subscribed to the messages, can transfer the data to storage or process it immediately in situations such as fraud detection.

Stackdriver Logging: Centralized log management

Logging (/logging/) is a centralized log-management service that collects log data from apps running on Google Cloud and other public and private cloud platforms. Export data collected by Logging by using built-in tools that send the data to Cloud Storage (/storage/), Pub/Sub (/pubsub/), and BigQuery (/bigquery/).

Many Google Cloud services automatically record log data to Logging. For example, apps running on App Engine (/appengine/) automatically log the details of each request and response to Logging. You can also write custom logging messages to `stdout` and `stderr`, which Logging automatically collects and displays in the Logs Viewer.

Logging provides a logging agent, based on fluentd (http://www.fluentd.org/), that you can run on virtual machine (VM) instances hosted on Compute Engine (/compute/docs/) as well as container clusters managed by GKE (/kubernetes-engine/). The agent streams log data from common third-party apps and system software to Logging.

Ingesting streaming data

Streaming data is delivered asynchronously, without expecting a reply, and the individual messages are small in size. Commonly, streaming data is used for *telemetry*, collecting data from geographically dispersed devices. Streaming data can be used for firing event triggers, performing complex session analysis, and as input for machine learning tasks.

Here are two common uses of streaming data.

- **Telemetry data**: Internet of Things (IoT) devices are network-connected devices that gather data from the surrounding environment through sensors. Although each device might send only a single data point every minute, when you multiply that data by a large number of devices, you quickly need to apply big data strategies and patterns.
- **User events and analytics**: A mobile app might log events when the user opens the app and whenever an error or crash occurs. The aggregate of this data, across all mobile devices where the app is installed, can provide valuable information about usage, metrics, and code quality.

Pub/Sub: Real-time messaging

Pub/Sub (/pubsub/) is a real-time messaging service that allows you to send and receive messages between apps. One of the primary use cases for inter-app messaging is to ingest streaming event data. With streaming data, Pub/Sub automatically manages the details of sharding, replication, load-balancing, and partitioning of the incoming data streams.

Most streaming data is generated by users or systems distributed across the globe. Pub/Sub has global endpoints and leverages Google's global front-end load balancer to support data ingestion across all Google Cloud regions, with minimal latency. In addition, Pub/Sub scales quickly and automatically to meet demand, without requiring the developer to pre-provision the system resources. For more details on how Pub/Sub scales, see the case-study by Spotify (https://labs.spotify.com/2016/03/03/spotifys-event-delivery-the-road-to-the-cloud-part-ii/).

Topics are how Pub/Sub organizes message streams. Apps streaming data to Pub/Sub target a topic. When it receives each message, Pub/Sub attaches a unique identifier and timestamp.

After the data is ingested, one or more apps can retrieve the messages by using a topic subscription. This can be done in either a pull or push model. In a push subscription, the Pub/Sub server sends a request to the subscriber app at a preconfigured URL endpoint. In the pull model, the subscriber requests messages from the server and acknowledges receipt. Pub/Sub guarantees message delivery at least once per subscriber.

Pub/Sub doesn't provide guarantees about the order of message delivery. Strict message ordering can be achieved with buffering (/pubsub/docs/ordering#order_of_processed_messages_matters), often using Dataflow (/dataflow/model/pubsub-io).

A common use of Pub/Sub is to move streaming data into Dataflow for real-time processing, per actual event time. When processed, you can move the data into a persistent storage service, such as Datastore (/datastore/) and BigQuery (/bigquery/), which support queries ordered by app timestamps.

Ingesting bulk data

Bulk data consists of large datasets where ingestion requires high aggregate bandwidth between a small number of sources and the target. The data could be stored in files, such as CSV, JSON, Avro, or Parquet files, or in a relational or NoSQL database. The source data could be located on-premises or on other cloud platforms.

Consider the following use cases for ingesting bulk data.

- **Scientific workloads**: Genetics data stored in Variant Call Format (VCF) text files are uploaded to Cloud Storage (/storage/) for later import into Genomics (/genomics/).
- **Migrating to the cloud**: Moving data stored in an on-premises Oracle database to a fully managed Cloud SQL (/sql/) database using Informatica.
- **Backing up data**: Replicating data stored in an AWS bucket to Cloud Storage using Cloud Storage Transfer Service (/storage/transfer/).
- **Importing legacy data**: Copying ten years worth of website log data into BigQuery (/bigquery/) for long-term trend analysis.

Google Cloud and partner companies provide a variety of tools you can use to load large sets of data into Google Cloud.

Storage Transfer Service: Managed file transfer

Storage Transfer Service (/storage/transfer/) manages the transfer of data to a Cloud Storage bucket. The data source can be an AWS S3 bucket, a web-accessible URL, or another Cloud

Storage bucket. Storage Transfer Service is intended for bulk transfer and is optimized for data volumes greater than 1 TB.

Backing up data is a common use of Storage Transfer Service. You can back up data from other storage providers to a Cloud Storage bucket. Or you can move data between Cloud Storage buckets, such as archiving data from a Standard Storage (/storage/docs/storage-classes#standard) bucket to an Archive Storage (/storage/docs/storage-classes#archive) bucket to lower storage costs.

Storage Transfer Service supports one-time transfers or recurring transfers. It provides advanced filters based on file creation dates, filename filters, and the times of day you prefer to import data. It also supports the deletion of the source data after it's been copied.

Transfer Appliance: Shippable, high-capacity storage server

Transfer Appliance (/transfer-appliance/) is a high-capacity storage server that you lease from Google. You connect it to your network, load it with data, and ship it to an upload facility where the data is uploaded to Cloud Storage. Transfer Appliance comes in multiple sizes. In addition, depending on the nature of your data, you might be able to use deduplication and compression to substantially increase the effective capacity of the appliance.

To determine when to use Transfer Appliance, calculate the amount of time needed to upload your data by using a network connection. If you determine that it would take a week or more, or if you have more than 60 TB of data (regardless of transfer speed), it might be more reliable and expedient to transfer your data by using the Transfer Appliance.

Transfer Appliance deduplicates, compresses, and encrypts your captured data with strong AES-256 encryption using a password and passphrase that you provide. When you read your data from Cloud Storage, you specify the same password and passphrase. After every use of Transfer Appliance, the appliance is securely wiped and re-imaged to help prevent your data from being available to the next user.

Cloud Storage `gsutil`: Command-line interface

Cloud Storage provides `gsutil` (/storage/docs/gsutil), a command-line utility that you can use to move file-based data from any existing file system into Cloud Storage. Written in Python, `gsutil` runs on Linux, macOS and Windows systems. In addition to moving data into Cloud Storage, you can use `gsutil` to create and manage Cloud Storage buckets, edit access rights of objects, and copy objects from Cloud Storage. For more information on how to use `gsutil` for batch data ingestion, see Scripting Production Transfers (/storage/docs/gsutil/addlhelp/ScriptingProductionTransfers).

Database migration tools

If your source data is stored in a database, either on-premises or hosted by another cloud provider, there are several third-party apps you can use to migrate your data, in bulk, to Google Cloud. These apps are often co-located in the same environment as source systems and provide both one-time and ongoing transfers. Apps such as Talend (https://www.talend.com/) and Informatica (https://www.informatica.com/products.html) provide extract-transform-load (ETL) capabilities with built-in support for Google Cloud.

Google Cloud has several target databases suitable for migrating data from external databases.

- **Relational databases**: Data stored in a relational database management system (RDBMS) can be migrated to Cloud SQL (/sql/) and Cloud Spanner (/spanner/).
- **Data warehouses**: Data stored in a data warehouse can be moved to BigQuery (/bigquery/).
- **NoSQL databases**—Data stored in a column-oriented NoSQL database, such as HBase or Cassandra, can be migrated to Bigtable (/bigtable/). Data stored in a JSON-oriented NoSQL database, such as Couchbase or MongoDB, can be migrated to Datastore (/datastore/).

Partner solutions

A number of Google Cloud partners provide complementary solutions focused on bulk data movement.

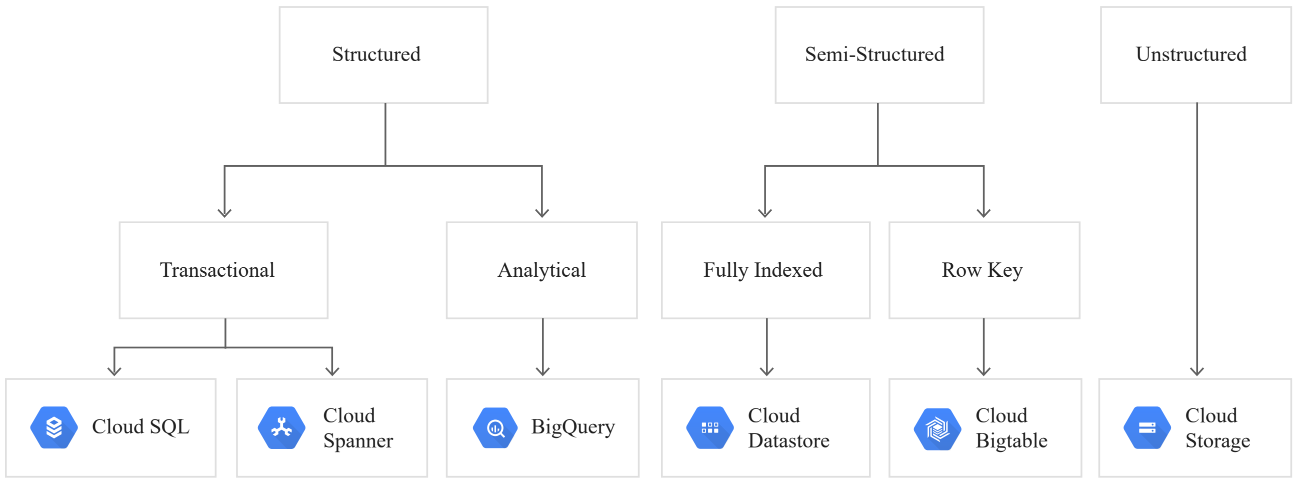
- WANDisco provides Google Active Migrator (https://www.wandisco.com/product/google-active-migrator), which automates the transfer of data from on-premises local and network storage into Dataproc clusters.
- Tervela offers Cloud FastPath (https://www.cloudfastpath.com/), for automating data migration and local file system synchronization with Cloud Storage.

- Iron Mountain (https://www.ironmountain.com/resources/general-articles/b/balancing-the-past-of-legacy-tape-with-the-future-of-disk-and-cloud-storage) and Prime Focus (http://www.primefocustechnologies.com/offline-media-import-export-form) offer the ability to load data into Cloud Storage from your physical media, such as hard disk drives, tapes, and USB flash drives.

For more information about partner solutions for data and analytics, see Partner Ecosystem for Data and Analytics (/solutions/data-analytics-partner-ecosystem).

Store

Data comes in many different shapes and sizes, and its structure is wholly dependent on the sources from which it was generated and the subsequent downstream use cases. For data and analytics workloads, ingested data can be stored in a variety of formats or locations.



Storing object data

Files are a common format for storing data, especially bulk data. With Google Cloud you can upload your file data to Cloud Storage, which makes that data available to a variety of other services.

Cloud Storage: Managed object storage

Cloud Storage (/storage/) offers durable and highly-available object storage for structured and unstructured data. For example, that data might be log files, database backup and export files, images, and other binary files. Files in Cloud Storage are organized by project into individual buckets. These buckets can support either custom access control lists (ACLs) or centralized identity and access management (IAM) controls.

Cloud Storage acts as a distributed storage layer, accessible by apps and services running on App Engine (/appengine/), GKE (/kubernetes-engine/), or Compute Engine (/compute/), and through other services such as Logging (/logging/).

Consider the following use cases for storing data.

- **Data backup and disaster recovery**: Cloud Storage offers highly durable and more secure storage for backing up and archiving your data.
- **Content distribution**: Cloud Storage enables the storage and delivery of content. For example, storing and delivering media files is scalable.
- **Storing ETL data**: Cloud Storage data can be accessed by Dataflow for transformation and loading into other systems such as Bigtable or BigQuery.
- **Storing data for MapReduce jobs**: For Hadoop and Spark jobs, data from Cloud Storage can be natively accessed by using Dataproc.
- **Storing query data**: BigQuery has the ability to import data from Cloud Storage into datasets and tables, or queries can be federated across existing data without importing. For direct access, BigQuery natively supports importing CSV, JSON, and Avro files from a specified Cloud Storage bucket.
- **Seeding machine learning**: Google Cloud machine learning APIs, such as the Cloud Vision API or the Cloud Natural Language API, can access data and files stored directly in Cloud Storage.
- **Archiving cold data**: Nearline, Coldline, and Archive Storage (/storage/archival) offer low latency, lower cost storage for objects that you plan to access less than once per month, less than once per quarter, or less than once per year, respectively.

Cloud Storage is available in multiple classes, depending on the availability and performance required for apps and services.

- Standard Storage (/storage/docs/storage-classes#standard) offers the highest levels of availability and is appropriate for storing data that requires low-latency access for frequently accessed data, such as data used by Compute Engine instances. Example use cases include serving website content, interactive storage workloads, data supporting mobile and gaming apps, data-intensive computations and big data processing.
- Nearline Storage (/storage/docs/storage-classes#nearline) is a low-cost, highly durable storage service for storing data that you access less than once per month. Nearline Storage offers fast access to data, on the order of sub-second response times and is useful for data archiving, online backup, or disaster recovery use cases.
- Coldline Storage (/storage/docs/storage-classes#coldline) provides a very-low-cost, highly durable storage service for storing data that you intend to access less than once per quarter. Coldline Storage offers fast access to data, on the order of sub-second response times, and is appropriate for data archiving, online backup, and disaster recovery.
- Archive Storage (/storage/docs/storage-classes#archive) provides a lowest-cost, highly durable storage service for storing data that you intend to access less than once per year. Archive Storage offers fast access to data, on the order of sub-second response times, and is ideal for data archiving, online backup, and disaster recovery.

Cloud Storage for Firebase: Scalable storage for mobile app developers

Cloud Storage for Firebase (https://firebase.google.com/docs/storage/) is a simple and cost-effective object storage service that's designed to scale with your user base. Cloud Storage for Firebase is a good fit for storing and retrieving assets such as images, audio, video, and other user-generated content in mobile and web apps.

Firebase SDKs for Cloud Storage perform uploads and downloads regardless of network quality. If they are interrupted due to poor connections, they restart where they stopped, saving you time and bandwidth. Out-of-the-box integration with Firebase Authentication (https://firebase.google.com/docs/auth/) allows you to configure access based on filename, size, content type, and other metadata.

Cloud Storage for Firebase stores your files in a Cloud Storage bucket, which gives you the flexibility to upload and download files from mobile clients that are using Firebase SDKs.

You can also perform server-side processing such as image filtering or video transcoding with Google Cloud.

To get started with Cloud Storage for Firebase, refer to the documentation (https://firebase.google.com/docs/storage/). Firebase has SDKs for iOS, Android, web, C++, and Unity clients.

Storing database data

Google Cloud provides a variety of databases, both RDBMS and NoSQL, that you can use to store your relational and nonrelational data.

Cloud SQL: Managed MySQL and PostgreSQL engines

Cloud SQL (/sql/) is a fully managed, cloud-native RDBMS that offers both MySQL and PostgreSQL engines with built-in support for replication. It's useful for low-latency, transactional, relational database workloads. Because it's based on MySQL and PostgreSQL, Cloud SQL supports standard APIs for connectivity. Cloud SQL offers built-in backup and restoration, high availability, and read replicas.

Cloud SQL supports RDBMS workloads up to 30 TB for both MySQL and PostgreSQL. Cloud SQL is accessible from apps running on App Engine (/appengine/), GKE (/kubernetes-engine/), or Compute Engine (/compute/). Because Cloud SQL is built on top of MySQL and PostgreSQL, it supports standard connection drivers, third-party app frameworks (such as Django and Ruby on Rails), and popular migration tools. Data stored in Cloud SQL is encrypted both in transit and at rest. Cloud SQL instances have built-in support for access control, using network firewalls to manage database access.

Cloud SQL is appropriate for typical online transaction processing (https://wikipedia.org/wiki/Online_transaction_processing) (OLTP) workloads.

- **Financial transactions**: Storing financial transactions requires ACID (https://wikipedia.org/wiki/ACID) database semantics, and data is often spread across multiple tables, so complex transaction support is required.
- **User credentials**: Storing passwords or other secure data requires complex field support and enforcement along with schema validation.

- **Customer orders**: Orders or invoices typically include highly normalized relational data and multi-table transaction support when capturing inventory changes.

Cloud SQL is not an appropriate storage system for online analytical processing (https://wikipedia.org/wiki/Online_analytical_processing) (OLAP) workloads or data that requires dynamic schemas on a per-object basis. If your workload requires dynamic schemas, consider Datastore. For OLAP workloads, consider BigQuery. If your workload requires wide-column schemas, consider Bigtable.

For downstream processing and analytical use cases, data in Cloud SQL can be accessed from multiple platform tools. You can use Dataflow or Dataproc to create ETL jobs that pull data from Cloud SQL and insert it into other storage systems.

Bigtable: Managed wide-column NoSQL

Bigtable (/bigtable/) is a managed, high-performance NoSQL database service designed for terabyte- to petabyte-scale workloads. Bigtable is built on Google's internal Bigtable database infrastructure that powers Google Search, Google Analytics, Google Maps, and Gmail. The service provides consistent, low-latency, and high-throughput storage for large-scale NoSQL data. Bigtable is built for real-time app serving workloads, as well as large-scale analytical workloads.

Bigtable schemas use a single-indexed row key associated with a series of columns; schemas are usually structured either as *tall* or *wide* and queries are based on row key. The style of schema is dependent on the downstream use cases and it's important to consider data locality and distribution of reads and writes to maximize performance. *Tall* schemas are often used for storing time-series events (/bigtable/docs/schema-design-time-series), data that is keyed in some portion by a timestamp, with relatively fewer columns per row. *Wide* schemas follow the opposite approach, a simplistic identifier as the row key along with a large number of columns. For more information, refer to the Bigtable Schema Design (/bigtable/docs/schema-design) documentation.

Bigtable is well suited for a variety of large-scale, high-throughput workloads such as advertising technology or IoT data infrastructure.

- **Real-time app data**: Bigtable can be accessed from apps running in App Engine flexible environment, GKE, and Compute Engine for real-time live-serving workloads.

- **Stream processing**: As data is ingested by Pub/Sub, Dataflow can be used to transform and load the data into Bigtable.
- **IoT time series data**: Data captured by sensors and streamed into Google Cloud can be stored using time-series schemas in Bigtable.
- **Adtech workloads**: Bigtable can be used to store and track ad impressions, as well as a source for follow-on processing and analysis using Dataproc and Dataflow.
- **Data ingestion**: Dataflow or Dataproc can be used to transform and load data from Cloud Storage into Bigtable.
- **Analytical workloads**: Dataflow can be used to perform complex aggregations directly from data stored in Bigtable, and Dataproc can be used to execute Hadoop or Spark processing and machine-learning tasks.
- **Apache HBase replacement**: Bigtable can also be used as a drop-in replacement for systems built using Apache HBase (https://hbase.apache.org/), an open source database based on the original Bigtable paper authored by Google. Bigtable is compliant with the HBase 1.x APIs so it can be integrated into many existing big-data systems. Apache Cassandra uses a data model based on the one found in the Bigtable paper, meaning Bigtable can also support several workloads that leverage a wide-column-oriented schema and structure.

While Bigtable is considered an OLTP system, it doesn't support multi-row transactions, SQL queries or joins. For those use cases, consider either Cloud SQL or Datastore.

Spanner: Horizontally scalable relational database

Spanner (/spanner/) is a fully managed relational database service for mission-critical OLTP apps. Spanner is horizontally scalable, and built for strong consistency, high availability, and global scale. This combination of qualities makes it unique as a service. Because Spanner is a fully managed service, you can focus on designing your app and not your infrastructure.

Spanner is a good fit if you who want the ease of use and familiarity of a relational database along with the scalability typically associated with a NoSQL database. Like relational databases, Spanner supports schemas, ACID transactions, and SQL queries (ANSI 2011). Like many NoSQL databases, Spanner scales horizontally in regions, but it can also scale *across* regions for workloads that have more stringent availability requirements. Spanner also performs automatic sharding while serving data with single-digit millisecond latencies.

Security features in Spanner include data-layer encryption, audit logging, and Cloud IAM integration.

To get started with Spanner, refer to the Spanner Documentation (/spanner/docs/).

The following are typical uses cases for Spanner.

- **Financial services**: Financial services workloads require strong consistency across read/write operations. Spanner provides this consistency without sacrificing high availability.
- **Ad tech**: Latency is a key consideration in the ad tech space. Spanner facilitates low-latency querying without compromising scale or availability.
- **Retail and global supply chain**: The need for global scale can force supply chain experts to make a trade-off between consistency and maintenance costs. Spanner offers automatic, global, synchronous replication with low latency, which means that data is always consistent and highly available.

Firestore: Flexible, scalable NoSQL database

Firestore (https://firebase.google.com/docs/firestore/) is a database that stores JSON data. JSON data can be synchronized in real time to connected clients across different platforms, including iOS, Android, JavaScript, IoT devices, and desktop apps. If a client does not have network connectivity, the Firestore API lets your app persist data to a local disk. After connectivity is reestablished, the client device synchronizes itself with the current server state.

Firestore provides a flexible, expression-based rules language, Firestore Security Rules (https://firebase.google.com/docs/firestore/security/get-started), that integrates with Firebase Authentication (https://firebase.google.com/docs/auth/) so you can define who has access to what data.

Firestore is a NoSQL database with an API that you can use to build a real-time experience serving millions of users without compromising responsiveness. To facilitate this level of scale and responsiveness, it's important to structure your data appropriately (https://firebase.google.com/docs/firestore/manage-data/structure-data). To get started with Firestore, refer to the documentation (https://firebase.google.com/docs/firestore/). Firestore has SDKs for iOS, Android, web, C++, and Unity clients.

Here are a couple of use cases for Firestore.

- **Chat and social**: Store and retrieve images, audio, video, and other user-generated content.
- **Mobile games**:Keep track of game progress and statistics across devices and device platforms.

Ecosystem databases

In addition to the database services provided by Google Cloud, you can deploy your own database software on high-performance Compute Engine virtual machines with highly scalable persistent storage. Traditional RDBMS such as EnterpriseDB (/marketplace/solution/public-edb-ppas/edb-postgres-enterprise) and Microsoft SQL Server (/sql-server/) are supported on Google Cloud. NoSQL database systems such as MongoDB (/solutions/deploy-mongodb) and Cassandra (https://cloudplatform.googleblog.com/2014/03/cassandra-hits-one-million-writes-per-second-on-google-compute-engine.html) are also supported in high-performance configurations.

Using Google Cloud Marketplace (/marketplace) you can deploy many types of databases onto Google Cloud using pre-built images, storage, and network settings. Deployment resources, such as Compute Engine instances, persistent disks, network configurations, can be managed directly and easily customized for different workloads or use cases.

Storing data warehouse data

A data warehouse stores large quantities of data for query and analysis instead of transactional processing. For data-warehouse workloads, Google Cloud provides BigQuery.

BigQuery: Managed data warehouse

For ingested data that will be ultimately analyzed in BigQuery (/bigquery/), you can store data directly in BigQuery, bypassing other storage mediums. BigQuery supports loading data through the web interface, command line tools, and REST API calls.

When loading data in bulk, the data should be in the form of CSV, JSON, or Avro files. You can then use the BigQuery web interface, command line tools, or REST API calls to load data from these file formats into BigQuery tables.

For streaming data, you can use Pub/Sub and Dataflow in combination to process incoming streams and store the resulting data in BigQuery. In some workloads, however, it might be appropriate to stream data directly into BigQuery without additional processing. You can also build custom apps, running on Google Cloud or on-premises infrastructure, that read from data sources with defined schemas and rows. The custom app can then stream that data into BigQuery tables using the Google Cloud SDKs or direct REST API calls.

Process and analyze

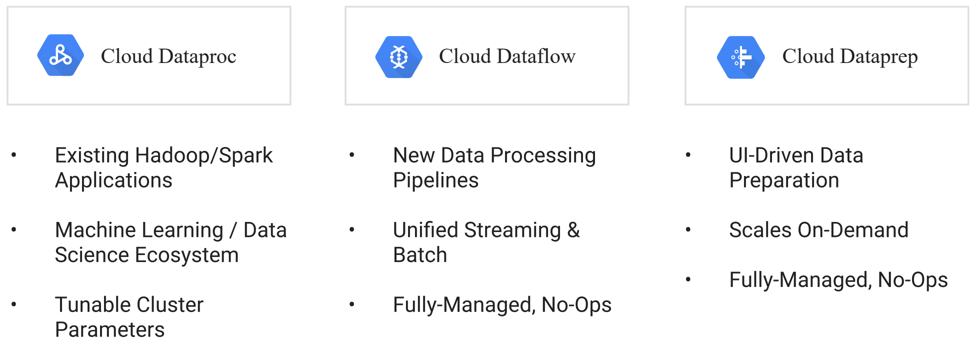
In order to derive business value and insights from data, you must transform and analyze it. This requires a processing framework that can either analyze the data directly or prepare the data for downstream analysis, as well as tools to analyze and understand processing results.

- **Processing**: Data from source systems is cleansed, normalized, and processed across multiple machines, and stored in analytical systems.
- **Analysis**: Processed data is stored in systems that allow for ad-hoc querying and exploration.
- **Understanding**: Based on analytical results, data is used to train and test automated machine-learning models.

Google Cloud provides services to process large-scale data, to analyze and query big data, and to understand data through machine learning.

Processing large-scale data

Large-scale data processing typically involves reading data from source systems such as Cloud Storage, Bigtable, or Cloud SQL, and then conducting complex normalizations or aggregations of that data. In many cases, the data is too large to fit on a single machine so frameworks are used to manage distributed compute clusters and to provide software tools that aid processing.



Cloud Dataproc	Cloud Dataflow	Cloud Dataprep
• Existing Hadoop/Spark Applications	• New Data Processing Pipelines	• UI-Driven Data Preparation
• Machine Learning / Data Science Ecosystem	• Unified Streaming & Batch	• Scales On-Demand
• Tunable Cluster Parameters	• Fully-Managed, No-Ops	• Fully-Managed, No-Ops

Dataproc: Managed Apache Hadoop and Apache Spark

The capability to deal with extremely large datasets has evolved since Google first published the MapReduce paper (http://research.google.com/archive/mapreduce.html) in 2004. Many organizations now load and store data in Hadoop Distributed File System (HDFS) and run periodic aggregations, reports or transformation using traditional batch-oriented tools, such as Hive or Pig. Hadoop has a large ecosystem to support activities such as machine learning using Mahout, log ingestion using Flume, and statistics using R, and more. The results of this Hadoop-based data processing are business critical. It is a non-trivial exercise for an organization dependent on these processes to migrate them to a new framework.

Spark has gained popularity over the past few years as an alternative to Hadoop MapReduce. Spark's performance is generally considerably faster than Hadoop MapReduce. Spark achieves this by distributing datasets and computation in memory across a cluster. In addition to speed increases, this distribution gives Spark the ability to deal with streaming data using Spark Streaming, as well as traditional batch analytics, transformations and aggregations using Spark SQL and a simple API. The Spark community is very active with several popular libraries including MLlib, which can be used for machine learning.

Running either Spark or Hadoop at an ever-growing scale, however, creates operational complexity and overhead as well as a continuous, and growing, fixed cost. Even if a cluster is only needed at discrete intervals, you still end up paying the cost of a persistent cluster. With Dataproc, you can migrate your existing Hadoop or Spark deployments to a fully-managed service that automates cluster creation, simplifies configuration and management of your cluster, has built-in monitoring and utilization reports, and can be shutdown when not in use.

Starting a new Dataproc cluster takes 90 seconds (/dataproc/#fast--scalable-data-processing) on average, which makes it easy to create a 10-node cluster or even a 1000-node cluster. This reduces the operational and cost overhead of managing a Spark or Hadoop deployment, while still providing the familiarity and consistency of either framework. Dataproc provides the ease and flexibility to spin up Spark or Hadoop clusters on demand when they are needed, and to terminate clusters when they are no longer needed. Consider the following use cases.

- **Log processing**: With minimal modification, you can process large amounts of text log data per day from several sources using existing MapReduce.
- **Reporting**: Aggregate data into reports and store the data in BigQuery. Then you can push the aggregate data to apps that power dashboards and conduct analysis.
- **On-demand Spark clusters**: Quickly launch ad-hoc clusters to analyze data is stored in blob storage using Spark (Spark SQL, PySpark, Spark shell).
- **Machine learning**: Use the Spark Machine Learning Libraries (MLlib), which are preinstalled on the cluster, to customize and run classification algorithms.

Dataproc also simplifies operational activities such as installing software or resizing a cluster. With Dataproc, you can natively read data and write results in Cloud Storage, Bigtable, or BigQuery, or the accompanying HDFS storage provided by the cluster. With Cloud Storage, Dataproc benefits from faster access to data and the ability to have many clusters seamlessly operate on datasets with no data movement, as well as removing the need to focus on data replication. This ability to store and checkpoint data externally makes it possible for you to treat Dataproc clusters as ephemeral resources with external persistence, which can be launched, consumed, and terminated as required.

Dataflow: Serverless, fully managed batch and stream processing

Being able to analyze streaming data has transformed the way organizations do business and how they respond in real-time. Having to maintain different processing frameworks to deal with batch and streaming analytics, however, increases complexity by necessitating two different pipelines. And spending time optimizing cluster utilization and resources, as you do with Spark and Hadoop, distracts from the basic objective of filtering, aggregating, and transforming your data.

Dataflow (/dataflow/) was designed to simplify big data for both streaming and batch workloads. It does this by unifying the programming model and the execution model. Instead of having to specify a cluster size and manage capacity, Dataflow is a managed service where on-demand resources are created, autoscaled, and parallelized. As a true *zero-ops* service, workers are added or removed based on the demands of the job. Dataflow also deals with the common problem of straggler workers found in distributed systems by constantly monitoring, identifying, and rescheduling (/blog/big-data/2016/05/no-shard-left-behind-dynamic-work-rebalancing-in-google-cloud-dataflow) work, including splits, to idle workers across the cluster.

Consider the following use-cases.

- **MapReduce replacement**: Process parallel workloads where non-MapReduce processing paradigms have led to operational complexity or frustration.
- **User analytics**: Analyze high-volume user-behavior data, such as in-game events, click stream data, and retail sales data.
- **Data science**: Process large amounts of data to make scientific discoveries and predictions, such as genomics, weather, and financial data.
- **ETL**: Ingest, transform, and load data into a data warehouse, such as BigQuery.
- **Log processing**: Process continuous event-log data processing to build real-time dashboards, app metrics, and alerts.

The Dataflow SDK has also been released as the open source project Apache Beam (http://beam.incubator.apache.org/), which supports execution on Apache Spark and Apache Flink. Because of its autoscaling and ease of deployment, Dataflow is an ideal location to run Dataflow/Apache Beam workflows.

Dataprep by Trifacta: Visual data exploration, cleaning, and processing

Dataprep (/dataprep/) is a service for visually exploring, cleaning, and preparing data for analysis. You can use Dataprep using a browser based-UI, without writing code. Dataprep automatically deploys and manages the resources required to perform the transformations, based on demand.

With Dataprep, you can transform data of any size stored in CSV, JSON, or relational-table formats. Dataprep uses Dataflow to scale automatically and can handle terabyte-scale datasets. Because Dataprep is fully integrated with Google Cloud, you can process data no matter where it resides: in Cloud Storage, in BigQuery, or on your desktop. After the data is processed, you can export clean data directly to BigQuery for further analysis. You can manage user access and data security with Cloud Identity and Access Management (/iam/docs/).

Here are some common use cases for Dataprep.

- **Machine Learning**: You can clean training data for fine-tuning ML models.
- **Analytics**: You can transform raw data so that it can be ingested into data warehousing tools such as BigQuery.

Analyzing and querying data

After data is ingested, stored, and processed, it needs to end up in a format that lets it be easily accessed and queried.

BigQuery: Managed data warehouse

BigQuery (/bigquery/) is a fully-managed data warehouse with support for ad-hoc SQL queries and complex schemas. You can use BigQuery to analyze, understand, and organize data. If you are accustomed to using a traditional data warehouse to run standard SQL queries or business intelligence and visualization tools, you will appreciate the power and familiar interface of BigQuery.

BigQuery is a highly-scalable, highly-distributed, low-cost analytics OLAP data warehouse capable of achieving a scan rate of over 1 TB/sec. It is a fully-managed service; computational nodes are spun up for each query entered in the system.

To get started with BigQuery you create a dataset in your project, load data into a table, and execute a query. The process of loading data can be simplified by using streaming ingestion from Pub/Sub and Dataflow, loading data from Cloud Storage, or by using the output from a processing job run on Dataflow or Dataproc. BigQuery can import CSV, Avro and JSON data formats and includes support for nested and repeated items in JSON.

All data in BigQuery is accessed over an encrypted channel and encrypted at rest. BigQuery is covered by Google's compliance programs including SOC, PCI, ISO 27001 and HIPAA

(/security/compliance), so it can be used to handle and query sensitive information. Access to data is controlled through your ACLs.

BigQuery calculates billing charges (/bigquery/pricing) along two independent dimensions: queries and storage. Storing data in BigQuery is comparable in cost with storing data in Cloud Storage, which means you don't need to choose between keeping log data in a bucket and in BigQuery. There is no upper limit to the amount of data that can be stored in BigQuery, in addition, if tables are not edited for 90 days, the price of storage for that table drops by 50% (/bigquery/pricing#long-term-storage).

A typical use case for BigQuery is to stream or periodically batch load log data from servers or other systems producing signals at a high rate, such as IoT devices. Native integration is available with several Google services. For example, Logging can be configured to deliver logging data directly into BigQuery.

When querying data in BigQuery, you have the option of two pricing models: on-demand or flat-rate. With on-demand pricing, query charges are priced according to terabytes processed. When using flat-rate pricing, BigQuery gives you consistent query capacity with a simpler cost model.

As a fully-managed service, BigQuery automates tasks such as infrastructure maintenance windows and data vacuuming. To improve the design of your queries, you can examine the query plan explanation (/bigquery/query-plan-explanation) of any given query. Data is stored in a columnar format, which is optimized for large-scale aggregations and data processing. In addition, BigQuery has built-in support for time-series partitioning of data. From a design perspective, this means you could design your loading activity to use a timestamp and then target queries in a particular date partition. Because BigQuery query charges are based on the amount of data scanned, proper partitioning (/bigquery/docs/partitioned-tables) of data can greatly improve query efficiency and reduce cost.

Running queries on BigQuery can be done using standard SQL (/bigquery/sql-reference/), which is compliant with SQL 2011 and has extensions to support querying nested and repeated data. There is a rich set of built-in functions and operators (/bigquery/query-reference) natively available in BigQuery, and support for user-defined functions (UDFs) (/bigquery/user-defined-functions).

You can leverage BigQuery in a variety of ways.

- **User analysis**: Ingest large amounts of user-generated activity (adtech, clickstream, game telemetry) and determine user behavior and characteristics.
- **Device and operational metrics**: Collect streaming information from IT systems, IoT devices, and so on. and analyze data for trends and variations.
- **Business intelligence**: Store business metrics as a data warehouse, and drive a BI tool or partner offering, such as Tableau, QlikView, or Looker.

Several tutorials and examples for using BigQuery are available on the BigQuery site (/bigquery/docs/tutorials).

Understanding data with machine learning

Machine learning has become a critical component of the analysis phase of the data lifecycle. It can be used to augment processed results, suggest data-collection optimizations, and predict outcomes in data sets.

Consider the following use cases.

- **Product recommendations**: You can build a model that recommends products based on previous purchases and site navigation.
- **Prediction**: Use machine learning to predict the performance of complex systems, such as financial markets.
- **Automated assistants**: Build automated assistants that understand and answer questions asked by users.
- **Sentiment analysis**: Determine the underlying sentiment of user comments on product reviews and news stories.

There are a number of options for leveraging machine learning in Google Cloud.

- **Task-specific machine learning APIs**: Google Cloud provides turn-key, managed, machine-learning services with pre-trained models for vision, speech, natural language, and text translation. These APIs are built from the same technologies that power apps such as Google Photos, the Google mobile app, Google Translate, and Inbox smart replies.

- **Custom machine learning**: AI Platform is a hosted, managed service that runs custom models at scale. In addition, Dataproc can also execute machine learning models built with Mahout or Spark MLlib.

The Vision API

You can use the Vision API (/vision/) to analyze and understand the content of an image using pre-trained neural networks. With the Vision API you can classify images, detect individual objects and faces, and recognize printed words. Additionally, you can use the Vision API to detect inappropriate content and to analyze emotional facial attributes of people.

The Vision API is accessible through REST endpoints. You can either send images directly to the service or upload them to Cloud Storage and include a link to the image in the request. Requests can include a single image, or multiple images can be annotated in a single batch. In a request, feature annotations can be selected for detection for each image enclosed. Feature detection includes labels, text, faces, landmarks, logos, safe search, and image properties (such as dominant colors). The response will contain metadata about each feature type annotation selected for each supplied in the original request. For more information about requests and response, refer to the Vision API documentation (/vision/docs/requests-and-responses).

You can easily integrate the Vision API into custom apps running on App Engine, GKE, Compute Engine, and mobile platforms such as Android and iOS. It can also be accessed from Google Cloud services such as Dataflow, Dataproc, and Datalab (/datalab/).

Speech-to-Text

The Speech-to-Text (/speech/) supports the ability to analyze audio and convert it to text. The API recognizes more than 80 languages and variants and is powered by deep-learning neural-network algorithms that constantly evolve and improve.

You can use the Speech-to-Text for different types of workloads.

- **Real-time speech-to-text**: The Speech-to-Text can accept streaming audio input and begin returning partial recognition results as they become available. This capability is useful for integrating real-time dictation or enabling command-and-control through voice in apps. The Speech-to-Text supports gRPC, a high-performance, open-source, general-purpose RPC framework, for streaming audio-speech analysis for custom apps running on App Engine, GKE, Compute Engine, and mobile platforms such as Android and iOS.

- **Batch analysis**: To process large numbers of audio files you can call the Speech-to-Text using REST endpoints and gRPC. Both synchronous and asynchronous speech-to-text capabilities are supported. The REST API can also be accessed from Google Cloud services such as Dataflow, Dataproc, and Datalab.

Natural Language API

The Natural Language API (/natural-language/) provides the ability to analyze and reveal the structure and meaning of text. The API can be used to extract information about people, places, events, the sentiment of the input text, and more. The resulting analysis can be used to filter inappropriate content, classify content by topics, or build relationships from the extracted entities found in the input text.

You can combine the Natural Language API with the Vision API OCR capabilities or the Speech-to-Text features to create powerful apps or services.

The Natural Language API is available through REST endpoints. You can either send text directly to the service, or upload text files to Cloud Storage and link to the text in your request. You can easily integrate the API into custom apps running on App Engine, GKE, Compute Engine, and mobile platforms such as Android and iOS. It can also be accessed from other Google Cloud services such as Dataflow, Dataproc, or Datalab.

Cloud Translation

You can use the Translation (/translate/) to translate more than 90 different languages. If the input language is unknown, the Translation automatically detects the language, with high accuracy.

The Translation can provide real-time translation for web and mobile apps, and supports batched requests for analytical workloads.

The Translation is available through REST endpoints. You can integrate the API into custom apps running on App Engine, GKE, Compute Engine, and mobile platforms such as Android and iOS. It can also be accessed from Google Cloud services such as Dataflow, Dataproc, or Datalab.

Video Intelligence API: Video search and discover

Video content has traditionally been opaque and has not easily lent itself to analysis. But with the Video Intelligence (/video-intelligence/), an easy-to-use REST API, you can now search, discover, and extract metadata from videos. Video Intelligence can detect entities (nouns) in video content, such as "dog," "flower," or "car." You can also look for entities in scenes in the video content.

You can annotate videos with frame-level and video-level metadata. (The service can extract data at a maximum granularity of 1 frame per second.) The API supports common video formats, including MOV, MPEG4, MP4, and AVI. Making a request to annotate a video is straightforward: you create a JSON request file with the location of the video and the type or types of annotation that you want to perform, and then submit the request to the API endpoint.

To get started, refer to the Video Intelligence Quickstart (/video-intelligence/docs/getting-started)
.

Here are some common use cases for Video Intelligence.

- **Glean insights from videos**: Extract insights from videos without having to use machine learning or implement computer vision algorithms.
- **Video catalog search**: Search through a catalog of videos to identify the presence and timestamp of entities of interest.

AI Platform: Managed machine learning platform

AI Platform (/ml/) is a managed platform you can use to run custom machine learning models at scale. You create models using the TensorFlow (https://www.tensorflow.org/) framework, an open source framework for machine intelligence, and then use AI Platform to manage preprocessing, training, and prediction.

AI Platform is integrated with Dataflow for data pre-processing, which can access data stored in both Cloud Storage and BigQuery. It also works with Cloud Load Balancing to serve online predictions at scale.

You can develop and test TensorFlow models completely in Google Cloud using Datalab and Jupyter (http://jupyter.org/) notebooks, and then use AI Platform for large-scale training and prediction workloads.

Models built for AI Platform are completely portable. By leveraging the TensorFlow framework, you can build and test models locally and then deploy them across multiple machines for distributed training and prediction. Finally, you can then upload the trained models to AI Platform and run them across multiple, distributed, virtual-machine instances.

The workflow of AI Platform consists of the following phases:

- **Preprocessing**: AI Platform converts features from input datasets into a supported format, and might also normalize and transform the data to enable more efficient learning. During preprocessing, the training, evaluation, and test data is stored in Cloud Storage. This also makes the data accessible to Dataflow during this phase for any additional required preprocessing.
- **Graph building**: AI Platform converts the supplied TensorFlow model into an AI Platform model with operations for training, evaluation, and prediction.
- **Training**: AI Platform continuously iterates and evaluates the model according to submitted parameters.
- **Prediction**: AI Platform uses the model to perform computations. Predictions can be computed in either batches or on-demand, as an online prediction service. Batch predictions are designed to be run against large datasets asynchronously, using services such as Dataflow to orchestrate the analysis. On-demand predictions are often used with custom apps running on App Engine, GKE, or Compute Engine.

General-purpose machine learning

In addition to the Google-built machine learning platform and APIs, you can deploy other high-scale machine-learning tools on Google Cloud. Mahout (http://mahout.apache.org/) and MLlib (http://spark.apache.org/mllib/) are two projects in the Hadoop and Spark ecosystems, that provide a range of general-purpose machine-learning algorithms. Both packages offer machine-learning algorithms for clustering, classification, collaborative filtering, and more.

You can use Dataproc to deploy managed Hadoop and Spark clusters, and bootstrap those clusters with additional software. This means you can run machine-learning workloads built with Mahout or MLlib on Google Cloud, and be able to scale the clusters using regular or preemptible VMs.

Explore and visualize

The final step in the data lifecycle is in-depth data exploration and visualization to better understand the results of the processing and analysis.

Insights gained during exploration can be used to drive improvements in the velocity or volume of data ingestion, the use of different storage mediums to speed analysis, and enhancements to processing pipelines. Fully exploring and understanding these data sets often involves the services of data scientists and business analysts, people trained in probability, statistics, and understanding business value.

Exploring data science results

Data science is the process of deriving value from raw data assets. To do so, a data scientist might combine disparate datasets, some public, some private, and perform a range of aggregation and analysis techniques. Unlike data warehousing, the types of analysis and the structure of the data vary widely and are not predetermined. Techniques include statistical methods, such as clustering, Bayesian, maximum likelihood, and regression, as well as machine learning, such as decision trees and neural networks.

Datalab: Interactive data insights

Datalab (/datalab/) is an interactive web-based tool that you can use to explore, analyze and visualize data. It is built on top of Jupyter (http://jupyter.org/) notebooks, which was formerly known as IPython. Using Datalab, you can, with a single click, launch an interactive web-based notebook where you can write and execute Python programs to process and visualize data. The notebooks maintain their state and can be shared between data scientists as well as published on sites such as GitHub, Bitbucket, and Dropbox.

Out of the box, Datalab includes support for many popular data-science toolkits, including pandas (http://pandas.pydata.org/), numpy (http://www.numpy.org/), and scikit-learn (http://scikit-learn.org/stable/), and common visualization packages, such as matplotlib (http://matplotlib.org/). Datalab also includes support for Tensorflow and Dataflow. Using these libraries and cloud services, a data scientist can load and cleanse data, build and verify models, and then visualize the results using matplotlib. This works both for data that fits on a single machine or for data that requires a cluster to store. Additional Python modules can be loaded using !pip install (http://ipython.readthedocs.io/en/stable/interactive/python-ipython-diff.html#shell-assignment) commands.

Data science ecosystem

Using high-performance Compute Engine instances, you can deploy many types of data science tools and use them to run large-scale analysis on Google Cloud.

The R programming language is commonly used by statisticians. If you want to use R for data exploration, you can deploy RStudio Server (https://www.rstudio.com/products/rstudio/) or Microsoft Machine Learning Server (https://docs.microsoft.com/machine-learning-server/what-is-machine-learning-server) on a Compute Engine instance. RStudio Server provides an interactive run-time environment to process and manipulate data, build sophisticated models, and visualize results. Microsoft Machine Learning Server is a high-scale and high-performance complement to R desktop clients for running analytical workloads.

Datalab is based on Jupyter and currently supports Python. If you want to do your data exploration in other languages such as R, Julia, Scala, and Java, you can deploy open-source Jupyter (http://jupyter.org/) or JupyterHub (https://jupyterhub.readthedocs.io/) on Compute Engine instances.

Apache Zeppelin (https://zeppelin.apache.org/) is another popular web-based, notebook-centric, data-science tool. Similar to Jupyter, Zeppelin provides support for additional language and data-processing backend systems such as Spark, Hive, R, and Python.

Both Jupyter and Zeppelin can be deployed using pre-built Dataproc initialization actions to quickly bootstrap common Hadoop- and Spark-ecosystem software packages.

Visualizing business intelligence results

During the analysis phase, you might find it useful to generate complex data visualizations, dashboards, and reports to explain the results of the data processing to a broader audience. To make this easier, Google Cloud integrates with a number of reporting and dashboarding tools.

Google Data Studio (https://www.google.com/analytics/data-studio/) provides a drag-and-drop report builder that you can use to visualize data into reports and dashboards that can then be shared with others. The charts and graphs in the reports are backed by live data, that can be shared and updated. Reports can contain interactive controls allowing collaborators to adjust the dimensions used to generate visualizations.

With Data Studio, you can create reports and dashboards from existing data files, Google Sheets, Cloud SQL, and BigQuery. By combining Data Studio with BigQuery, you can leverage the full computing and storage capacity of BigQuery without having to manually import data into Data Studio or create custom integrations.

If you prefer to visualize data in a spreadsheet, you can use Google Sheets (https://docs.google.com/spreadsheets), which integrates directly with BigQuery. Using Google Apps Script (https://www.google.com/script/start/), you can embed BigQuery queries and data directly inside Google Sheets. You can also export BigQuery query results into CSV files and open them in Google Sheets or other spreadsheets. This is useful to create smaller datasets for sharing or analysis. You can also do the reverse, use BigQuery to query across distributed data sets stored in Google Sheets or files stored in Google Drive.

BigQuery also supports a range of third-party business intelligence tools and integrations, ranging from SaaS to desktop apps. For more information, see BigQuery partners documentation (/bigquery/partners/).

Orchestration

Incorporating all of the elements of the data lifecycle into a set of connected and cohesive operations requires some form of orchestration. Orchestration layers are typically used to coordinate starting tasks, stopping tasks, copying files, and providing a dashboard to monitor data processing jobs. For example, a workflow could include copying files into Cloud Storage, starting a Dataproc processing job, and then sending notifications when processing results are stored in BigQuery.

Orchestration workflows can range from simple to complex, depending on the processing tasks, and often use a centralized scheduling mechanism to run workflows automatically. There are several open-source orchestration tools that support Google Cloud, such as Luigi (https://github.com/spotify/luigi) and Airflow (http://airflow.incubator.apache.org/). For custom orchestration apps, you can create an App Engine app that uses built-in scheduled tasks (/appengine/docs/python/config/cron) functionality to create and run workflows.

What’s next?

To learn more about how to manage your data on Google Cloud, see these reference architectures and use cases.

- Partner Ecosystem for Data and Analytics (/solutions/data-analytics-partner-ecosystem)
- Architecture: Complex Event Processing (/solutions/architecture/complex-event-processing)
- Architecture: Real-time Stream Processing for IoT (/solutions/architecture/real-time-stream-processing-iot)
- Building a Mobile Gaming Analytics Platform - a Reference Architecture (/solutions/mobile/mobile-gaming-analysis-telemetry)
- Processing Logs using Logging and Dataflow (/solutions/processing-logs-at-scale-using-dataflow)
- Using Machine Learning on Compute Engine to Make Product Recommendations (/solutions/recommendations-using-machine-learning-on-compute-engine)
- Cloud Big Data and Machine Learning Blog (/blog/big-data/)
- Cloud Big Data Solutions (/solutions/big-data/)
- Try out other Google Cloud features for yourself. Have a look at our tutorials (/docs/tutorials).