Big Data de Google

Ismael Yuste Varona
Strategic Cloud Engineer, Google
KSchool / Febrero 2021

Agenda

- Big Data
- Pub-Sub
- DataProc
- DataFlow
- BigQuery Intro
- AI Notebooks
- Composer
- Data Fusion
- Data Catalog

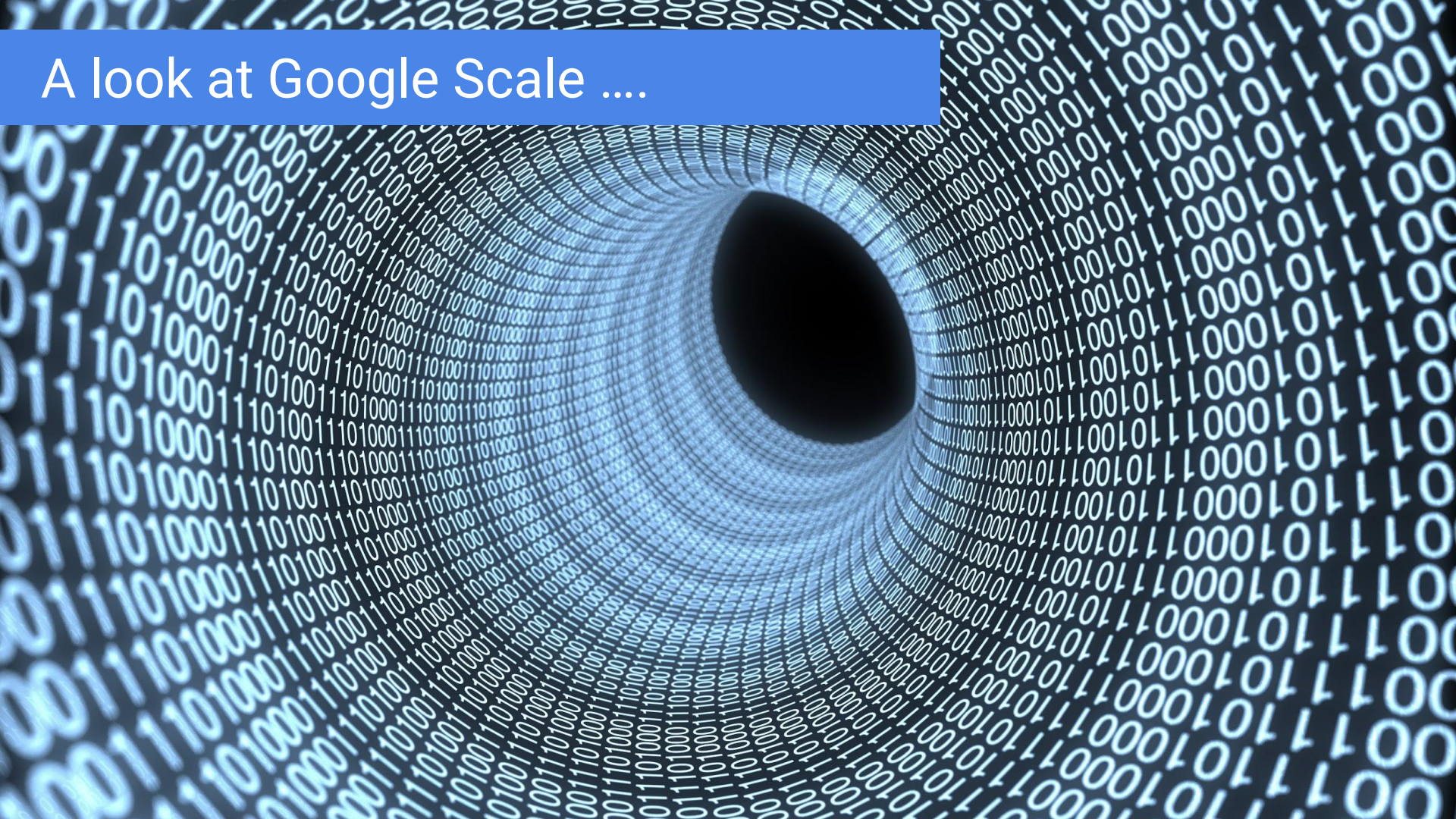
Google Cloud Platform - Trial

Periodo de Prueba de 90 días y 300 $ -

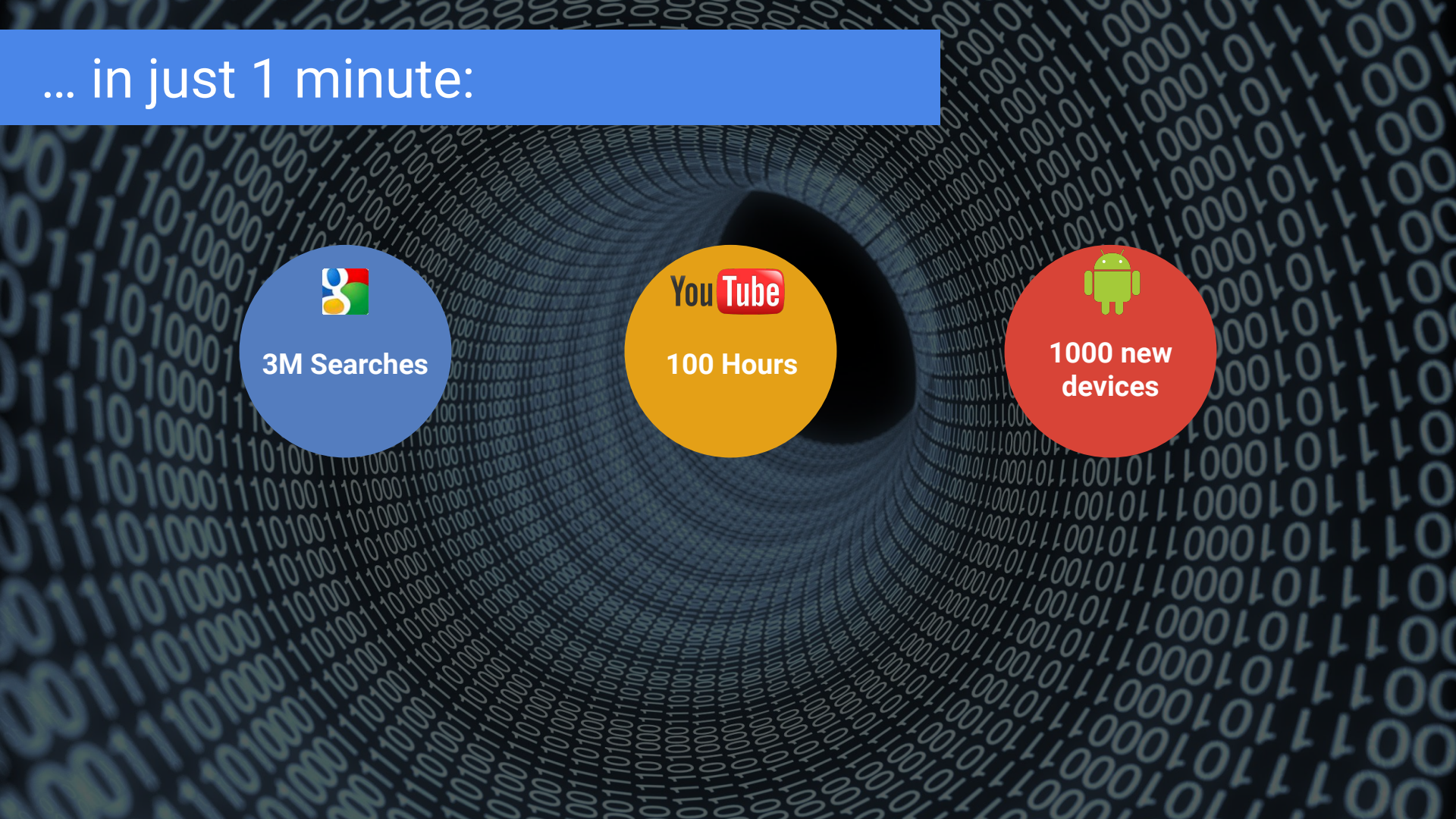
https://cloud.google.com/free/

GCP Big Data

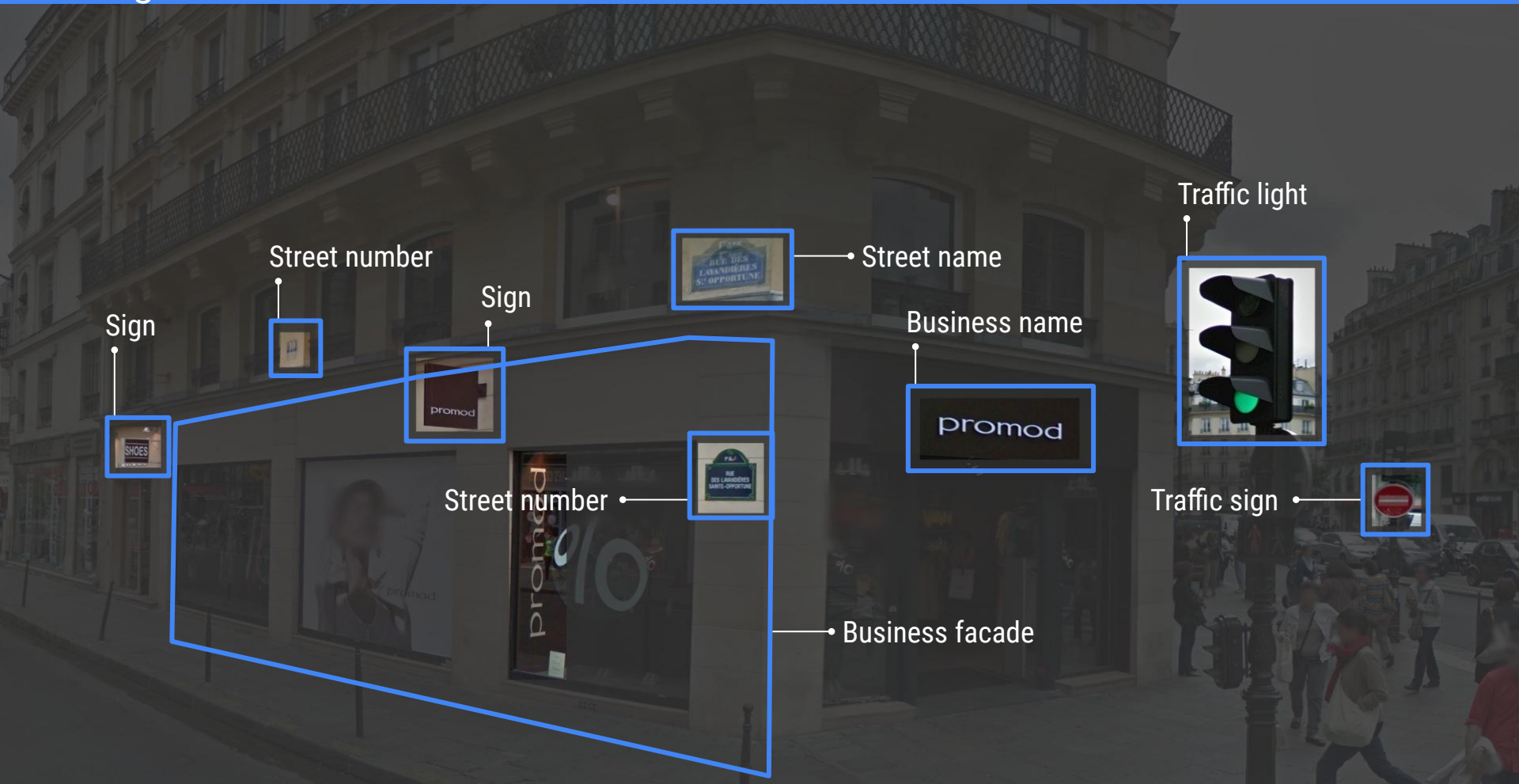
A look at Google Scale ….



… in just 1 minute:



Finding new value in data



Data Lifecycle Steps

Ingest

The first stage is to pull in the raw data, such as streaming data from devices, on-premises batch data, application 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.

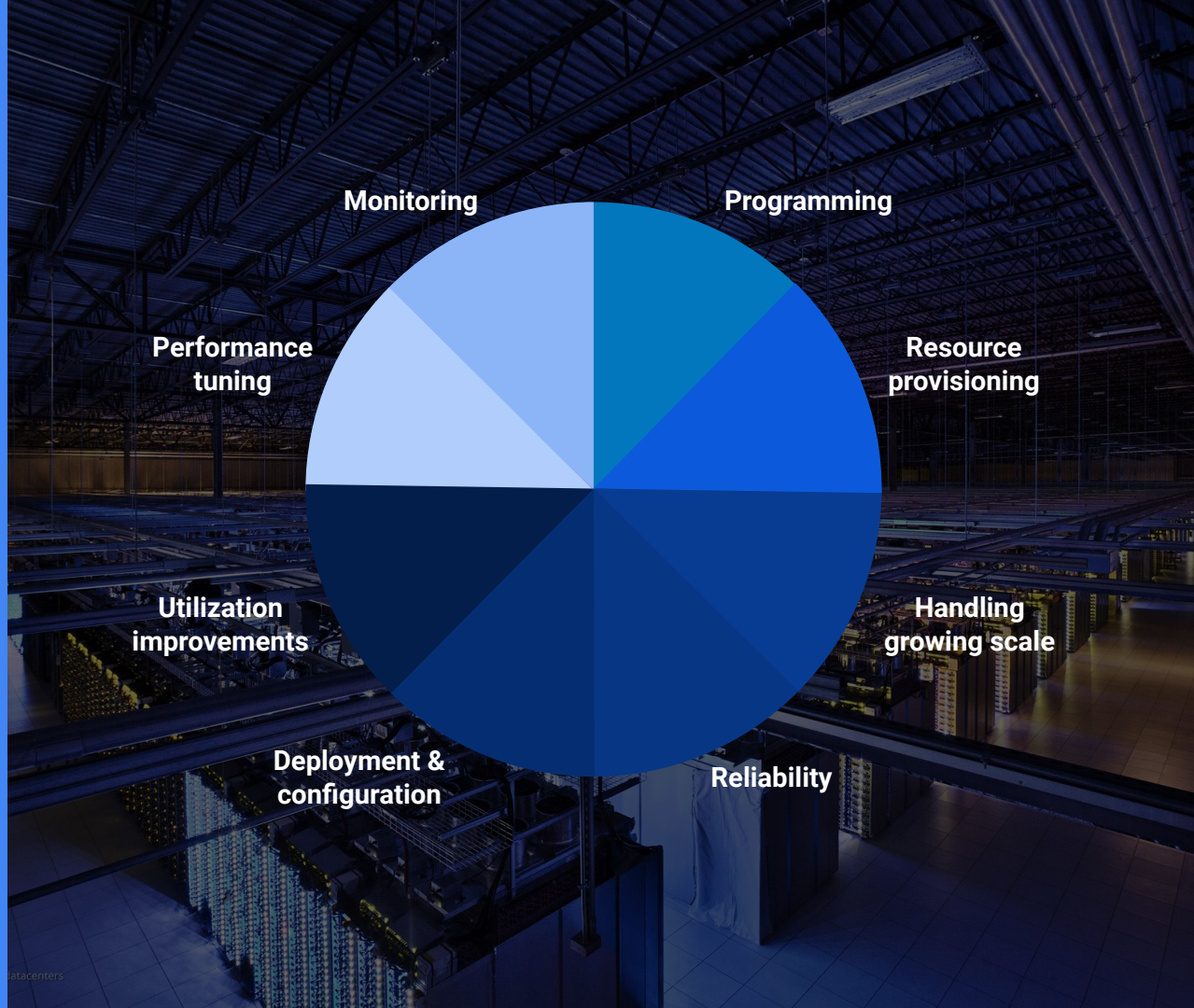
Process & Analyze

In this stage, the data is transformed from raw form into actionable information.

Explore & Visualize

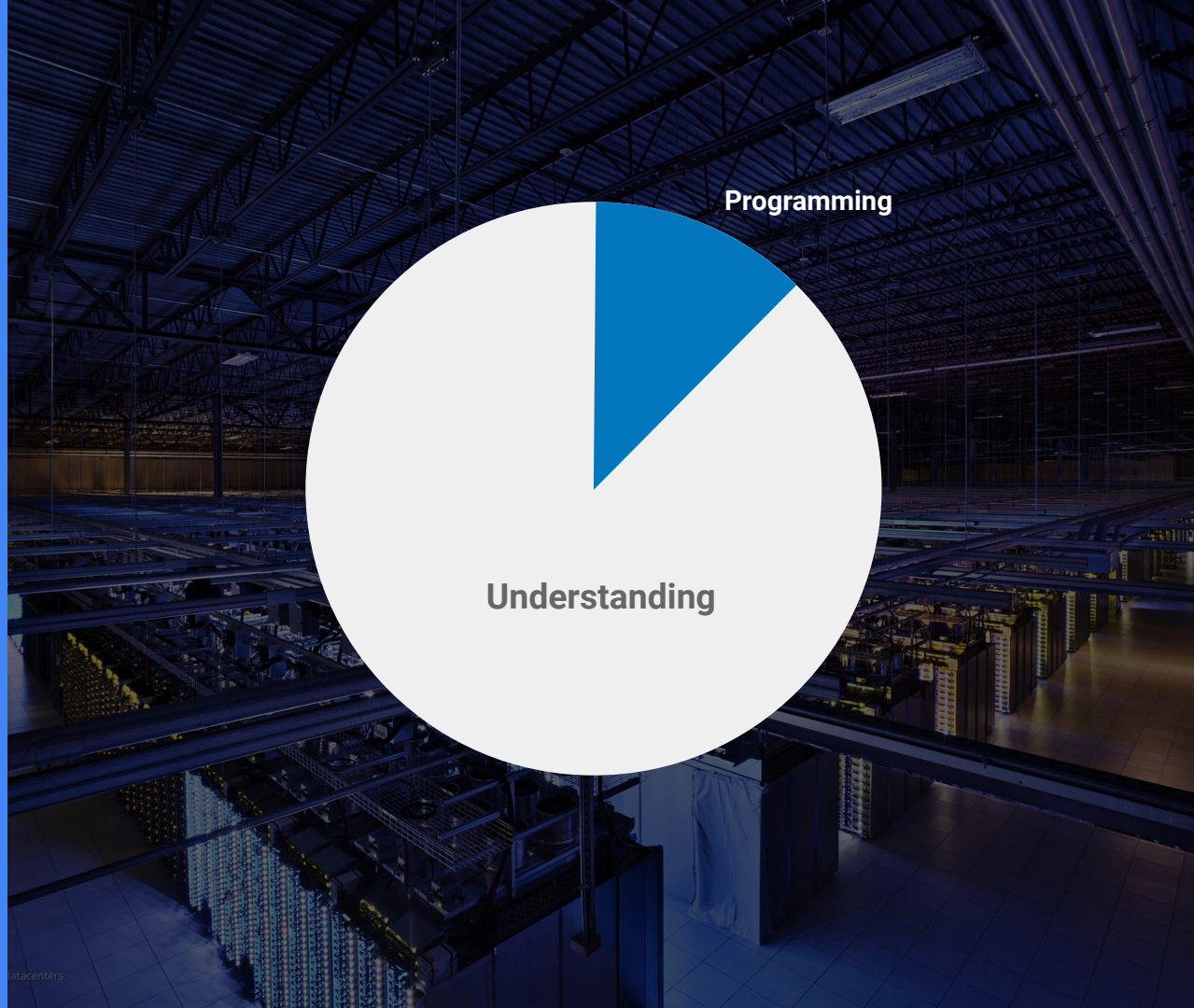
The final stage is to convert the results of the analysis into a format that is easy to draw insights from and to share with colleagues and peers.

Typical Big Data Jobs



Big Data with Google

Focus on insights.
Not infrastructure.
From batch to real-time.

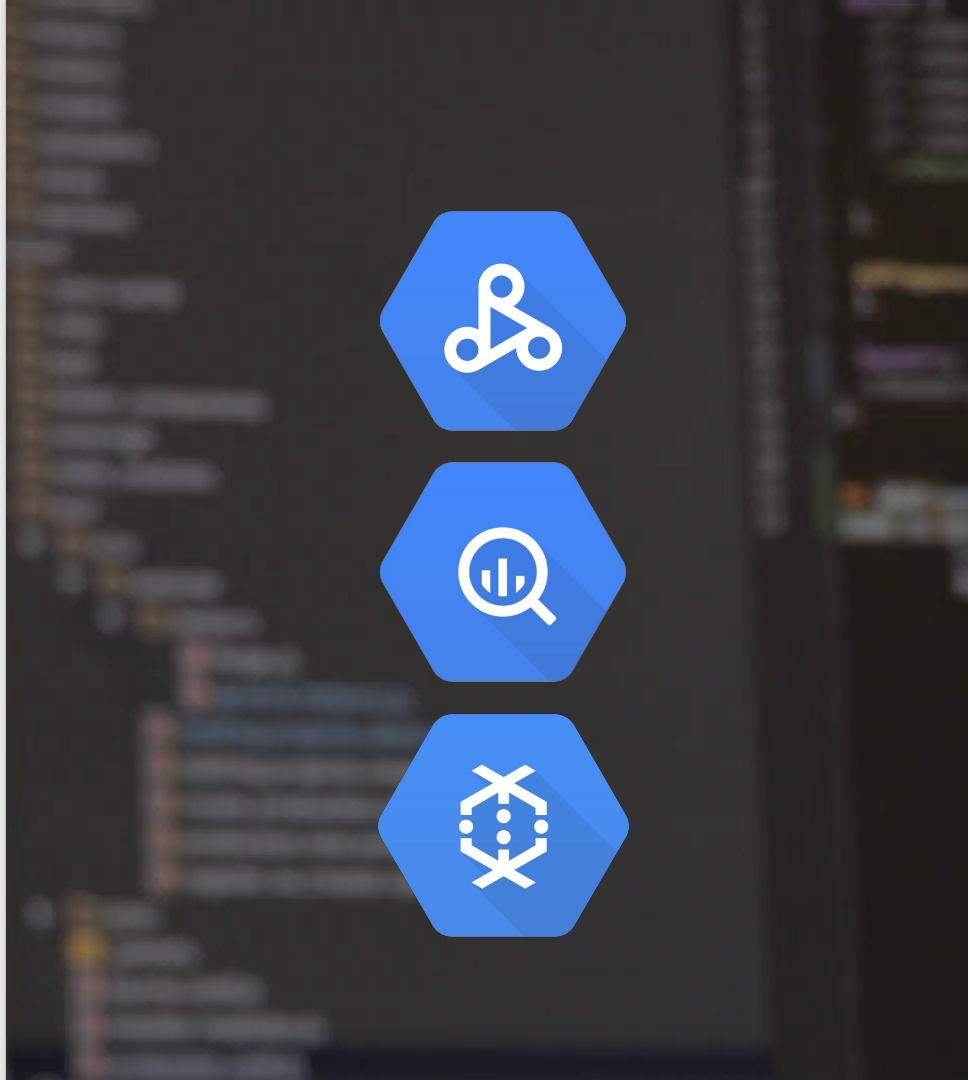


Data & Analytics

Cloud Dataproc
Fully managed Hadoop and Spark with industry-leading performance

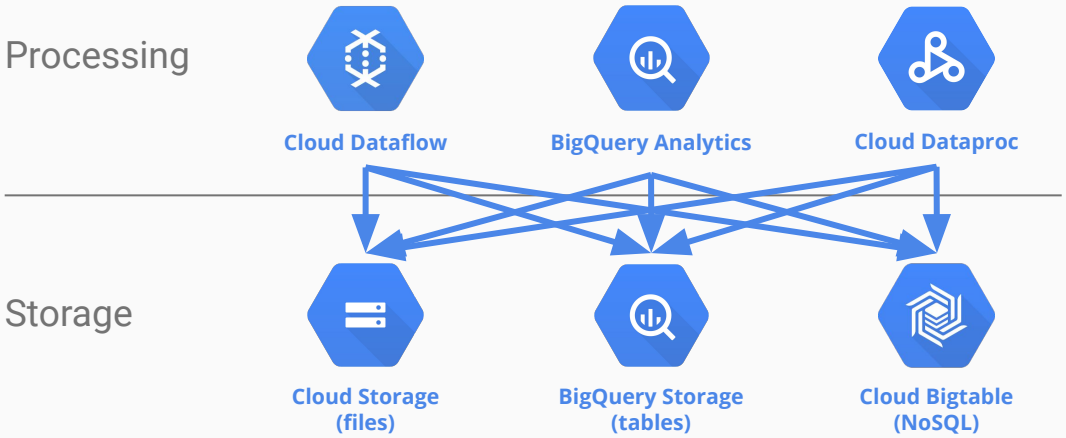
BigQuery
Fully managed data warehouse for large-scale analytics

Cloud Dataflow
Real-time data pipelines, with open source SDK via Apache Beam

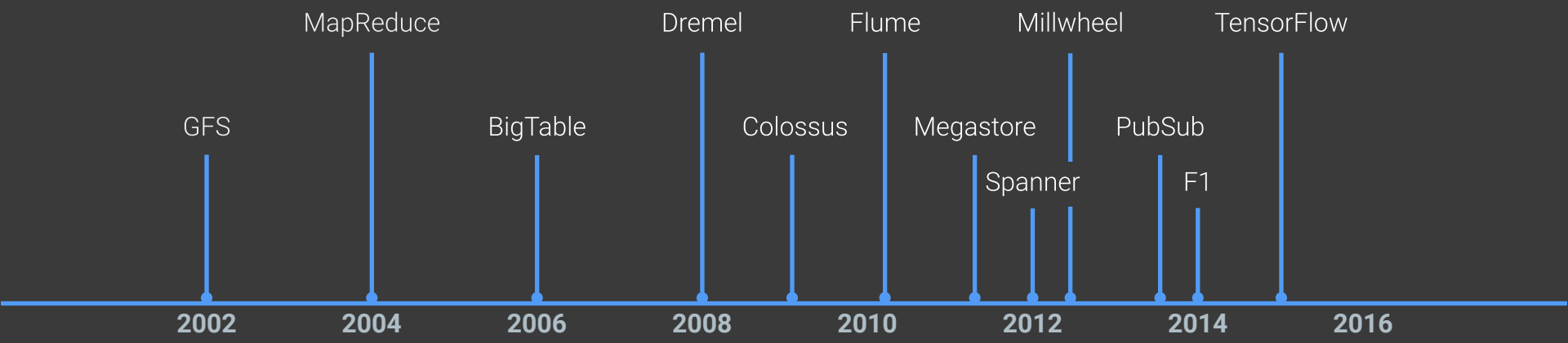


Separation of Storage and Compute

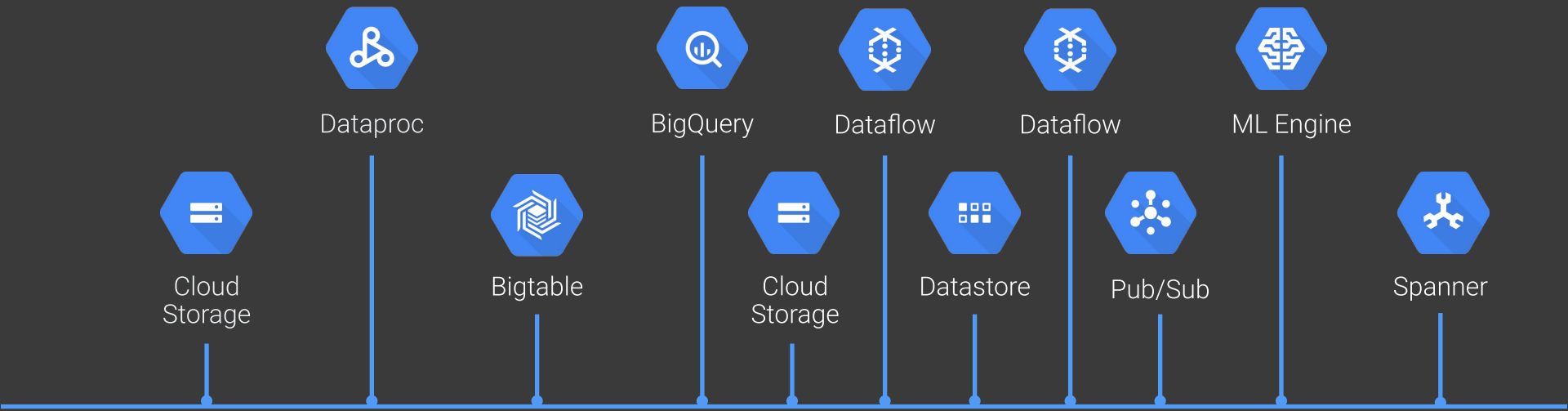
- Access any storage system from any processing tool
- Keep as much data as you want, economically
- Share data in place, no more FTP and copying



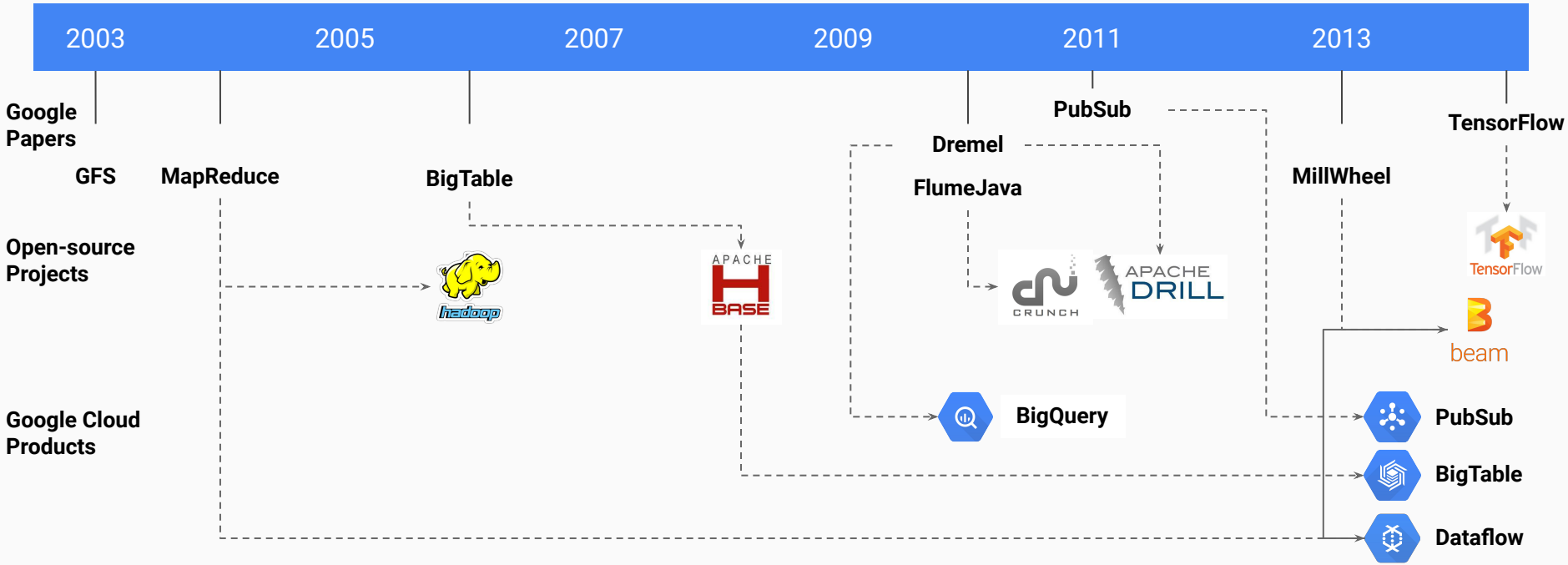
Google's Data Research



Google's Data Products



10+ years of Big Data innovation - Open Source



Product Mapping

Pub/Sub

kafka

RabbitMQ

BigQuery

Spark SQL

APACHE DRILL

HORTONWORKS

TERADATA

Dataflow

Spark Streaming

APEX

beam

Flink

Dataproc

hadoop

APACHE Spark

Cloud Data Fusion

CDAP

AI Platform Notebooks

jupyter

Cloud Composer

Apache Airflow

amazon web services

Microsoft Azure

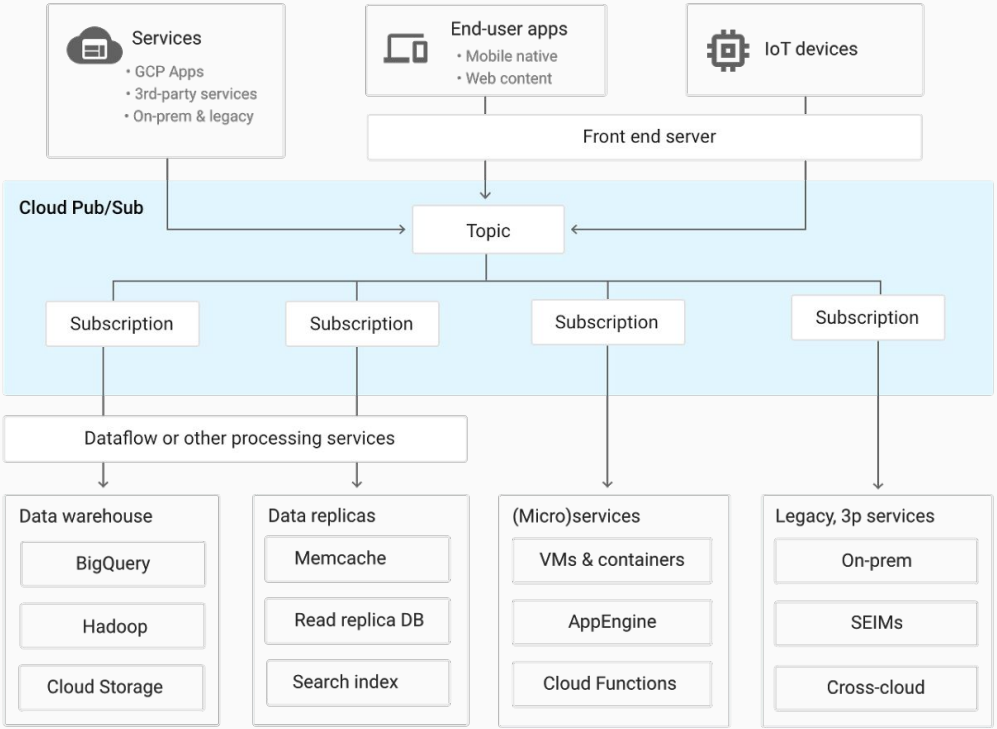
Pub Sub

Pub/Sub: 100% serverless event delivery

Google Cloud Pub/Sub

- Scalable, in-order message delivery with pull and push modes
- Auto-scaling and auto-provisioning with support from zero to hundreds of GB/second
- Independent quota and billing for publishers and subscribers
- Global message routing to simplify multi-region systems

Pub/Sub: Google Cloud Pub/Sub integrates services

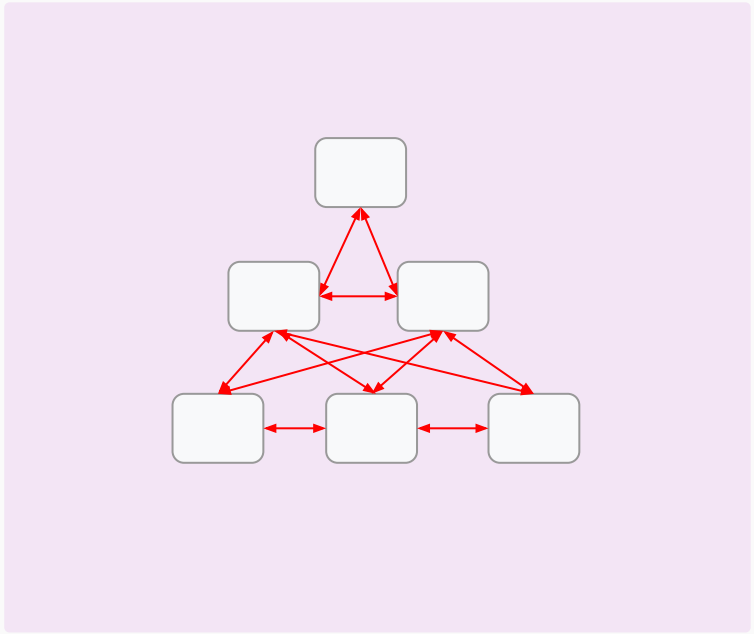


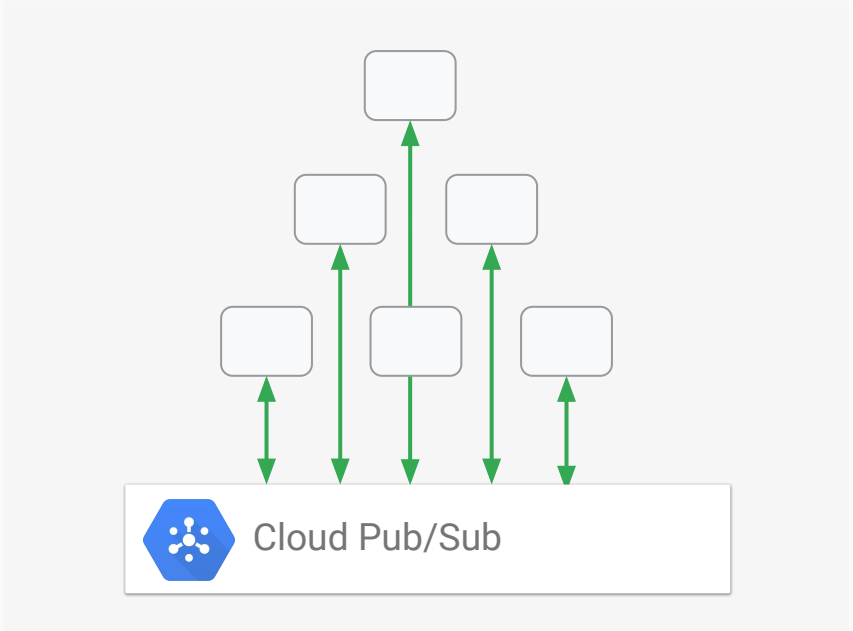
Pub/Sub: What is Google Cloud Pub/Sub

- Cloud Pub/Sub is a service to capture and rapidly pass massive amounts of data (or messages) between software applications with world-class security.
- It uses the **Publish - Subscribe** pattern:
  - **Publisher** applications can send **messages** to a **topic**.
  - **Subscriber** applications can subscribe to that topic to receive the message when the subscriber is ready (asynchronously).
- Pub/Sub acts as a buffer between sending and receiving software applications, making it easier for developers to connect applications.

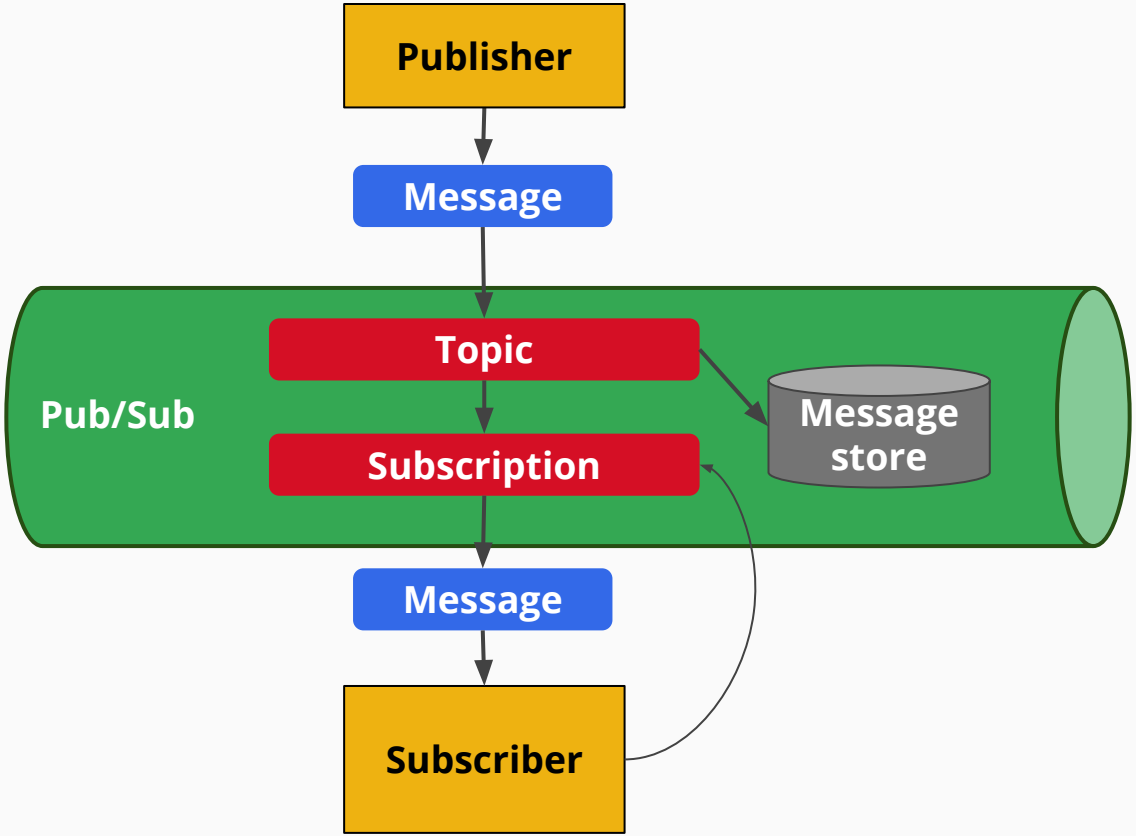
- Cloud Pub/Sub provides:
  - **Scale**—It supports Google's services, such as Gmail and ads.
  - **Reliability**—Dedicated resources in every Google Cloud Platform region enhance availability without increasing latency.
  - **Performance**—Sub-second notifications even when tested at over 1 million messages per second.
  - **Cost-efficiency**—A "pay for what you use" service.
  - **Ease of use and implementation**—Because it's a fully managed service, there's no need to manage your own open source software implementation. Get started in minutes, not days.

Pub/Sub: Point-to-point communication grows complex, flaky

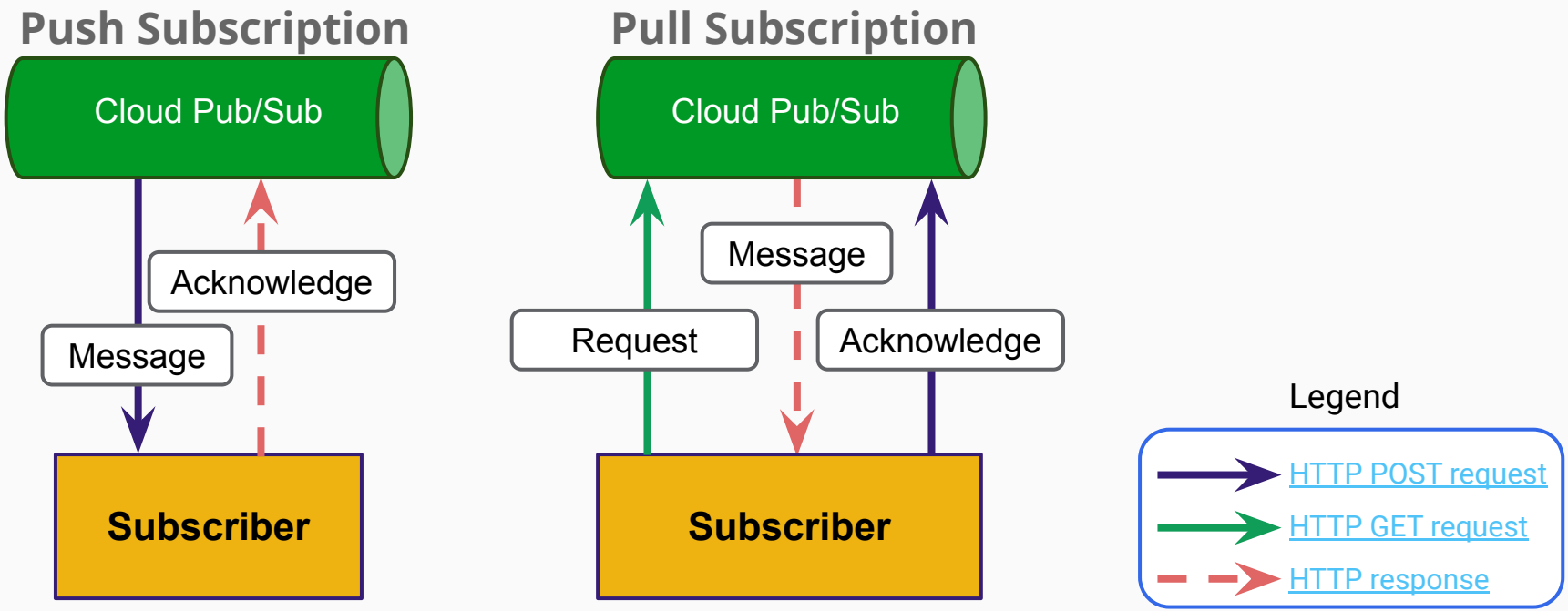




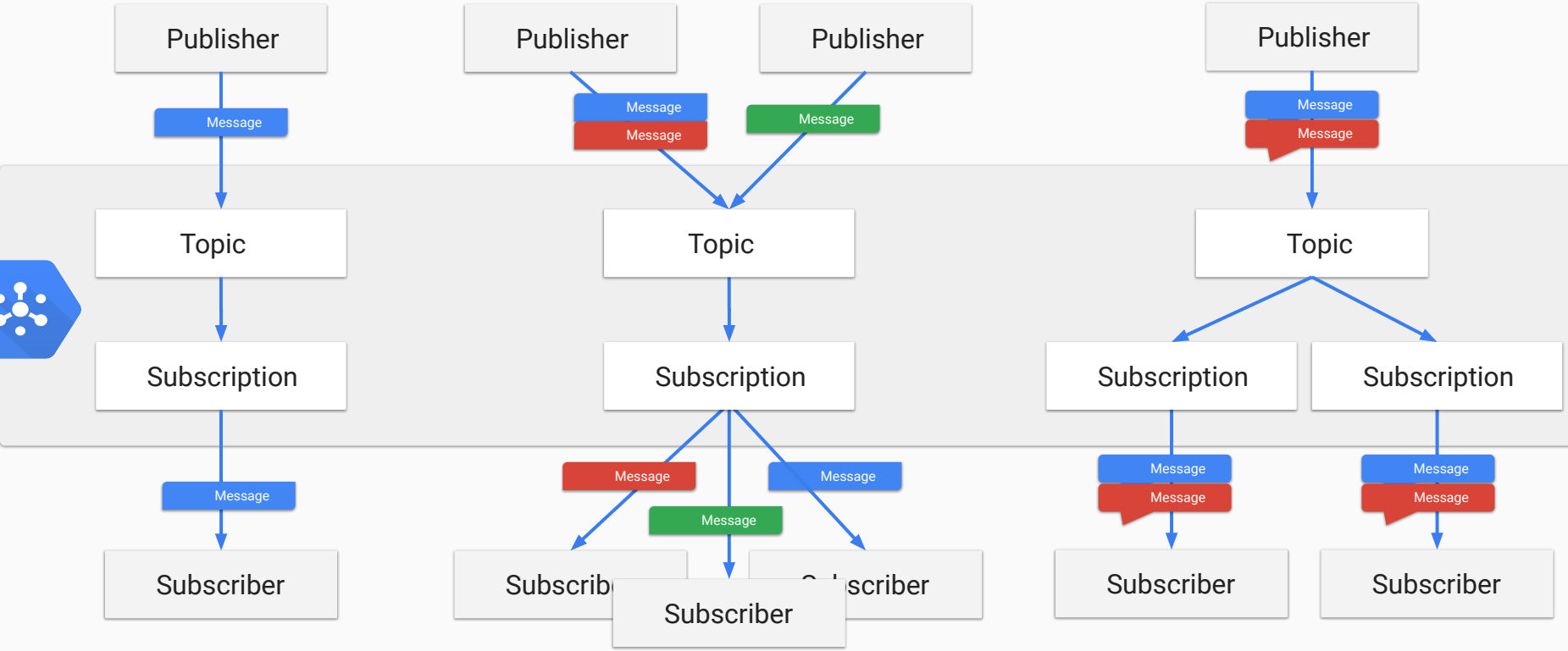
Pub/Sub: Architecture



Pub/Sub: Architecture



Pub/Sub: Publish/Subscribe patterns



Pub/Sub: Messaging is a shock-absorber

Availability

- Buffer messages and requests during outages
- Prevent message overloads that cause outages
- Redirect requests to recover from outages

Throughput

- Smooth out spikes in new request rate
- Balance load across multiple servers
- Balance arrival rate with service rate
- “Fan-in” from many devices

Latency

- Accept requests closer to the network edge
- Optimize message flow across regions

Pub/Sub: Messaging is a shock-absorber

Sources

- New data sources can plug into old data flows
- New data sources can use new schemas
- Common security policies for all sources

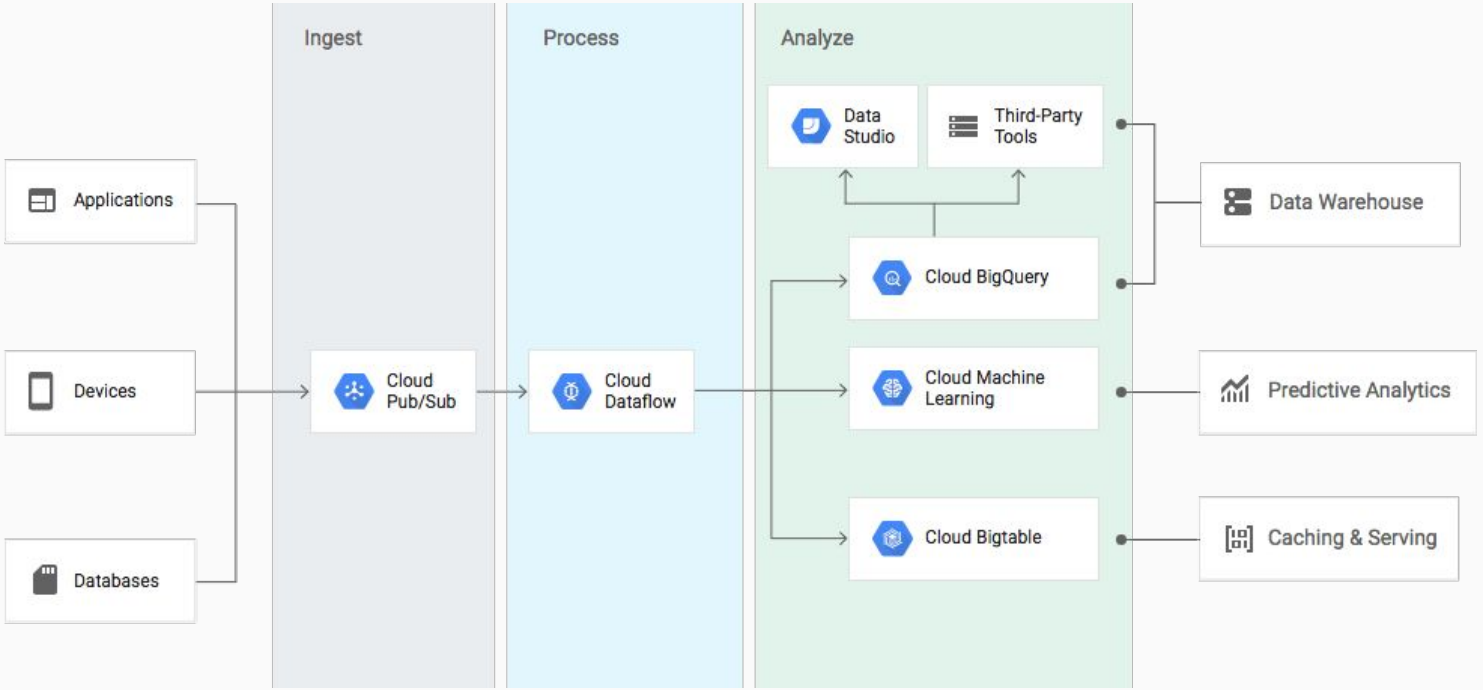
Sinks

- Data can be sent to new destinations
- “Push” and “pull” delivery are both available
- Spans organizational boundaries

Transforms

- Can merge streams into new topics
- Sends messages to Dataproc and Dataflow for transformation

Pub/Sub: GCP Stream Analytics: making event accessible and useful at scale



Pub/Sub: examples in use at Google

Chat & Mobile

Every time your Gmail displays a new message, it's because of a push notification to your browser or mobile device.

Ads and budgets

One of the most important real-time information streams in the company is advertising revenue—we use Pub/Sub to broadcast budgets to our entire fleet of search engines.

Push notifications

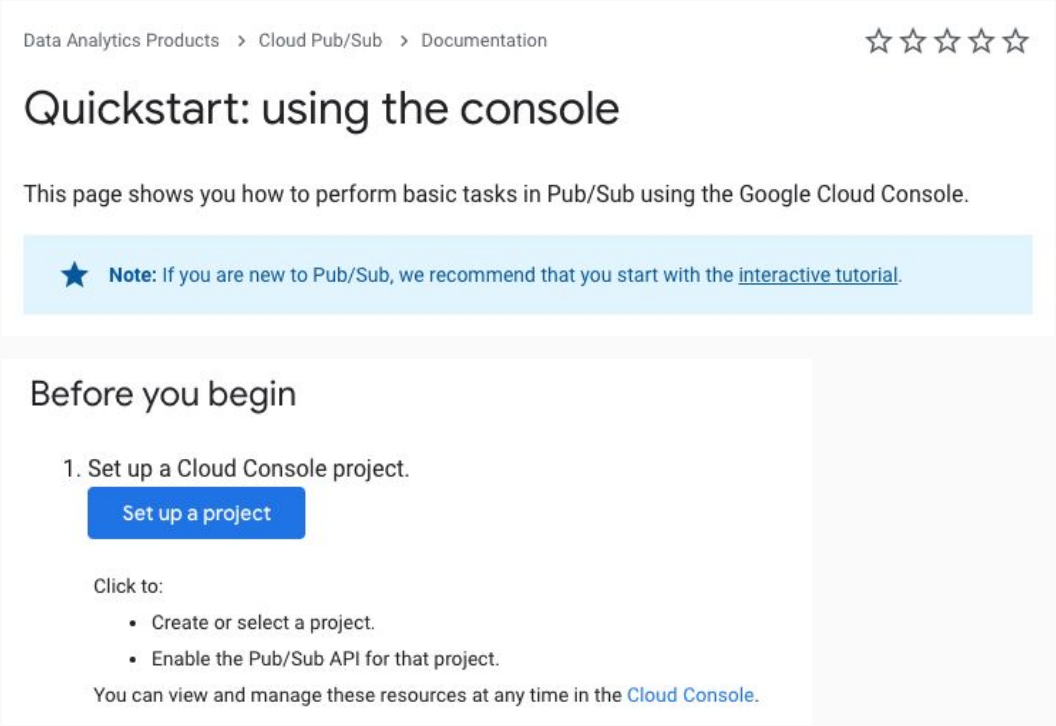
Google Cloud Messaging for Android delivers billions of messages a day, reliably and securely, for Google's own mobile apps and the entire developer community.

Instant search

Updating search results as you type is a feat of real-time indexing that depends on Pub/Sub to update caches with breaking news.

Ejercicio Pub-Sub

Inicio



Data Analytics Products > Cloud Pub/Sub > Documentation

Quickstart: using the console

This page shows you how to perform basic tasks in Pub/Sub using the Google Cloud Console.

Note: If you are new to Pub/Sub, we recommend that you start with the interactive tutorial.

Before you begin

1. Set up a Cloud Console project.

   Set up a project

   Click to:

   - Create or select a project.
   - Enable the Pub/Sub API for that project.

   You can view and manage these resources at any time in the Cloud Console.

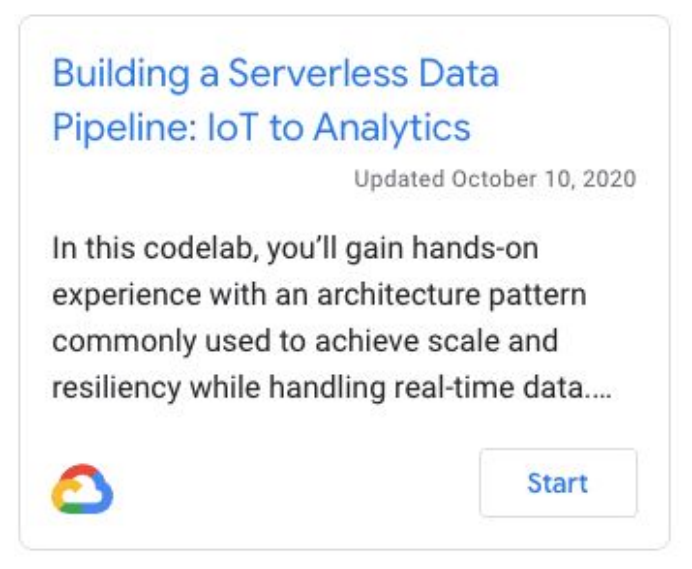
Bonus

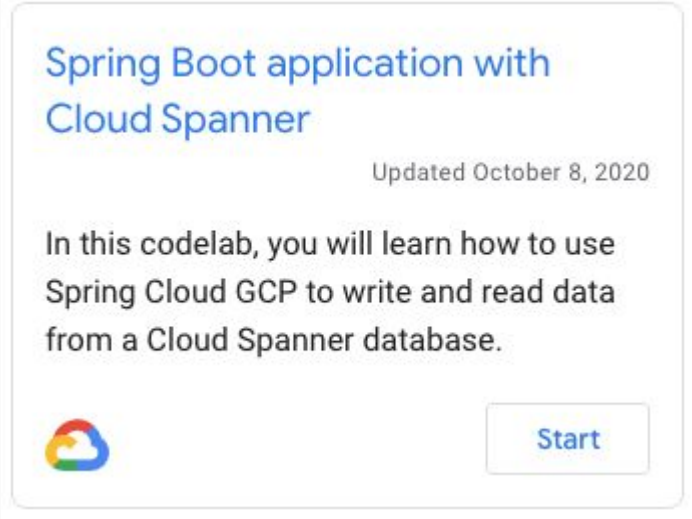
Quickstart: Using the gcloud Command-Line Tool

Pub/Sub is a messaging service for exchanging event data among applications and services. A producer of data publishes a messages to a Pub/Sub topic. A consumer creates a subscription to that topic. From this point on, Pub/Sub guarantees that the message will be delivered to every consumer of the message at least once. Consumers either pull messages from a subscription or are configured as webhooks for push subscriptions. Every subscriber must acknowledge each message within a configurable time window. Unacknowledged messages are redelivered. Pub/Sub is geographically global and does not require sharding or additional configuration to scale with demand.

This page shows you how to perform basic tasks in Google Cloud Pub/Sub.

Bonus

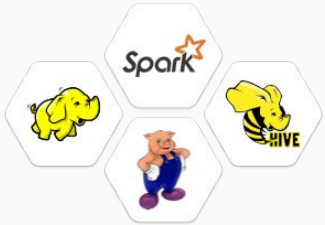

Building a Serverless Data Pipeline: IoT to Analytics
Updated October 10, 2020
In this codelab, you'll gain hands-on experience with an architecture pattern commonly used to achieve scale and resiliency while handling real-time data....
Start


Spring Boot application with Cloud Spanner
Updated October 8, 2020
In this codelab, you will learn how to use Spring Cloud GCP to write and read data from a Cloud Spanner database.
Start

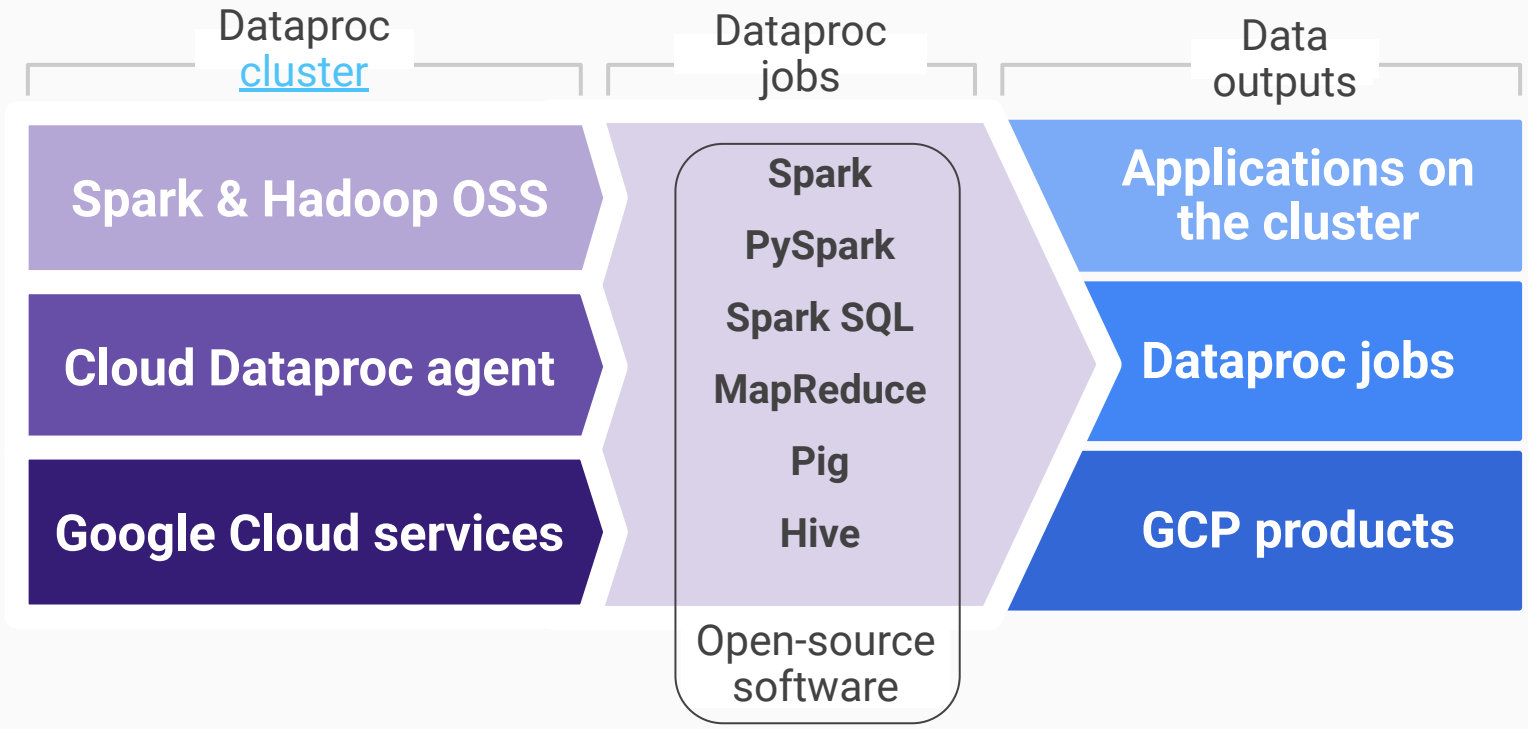
Dataproc

Dataproc

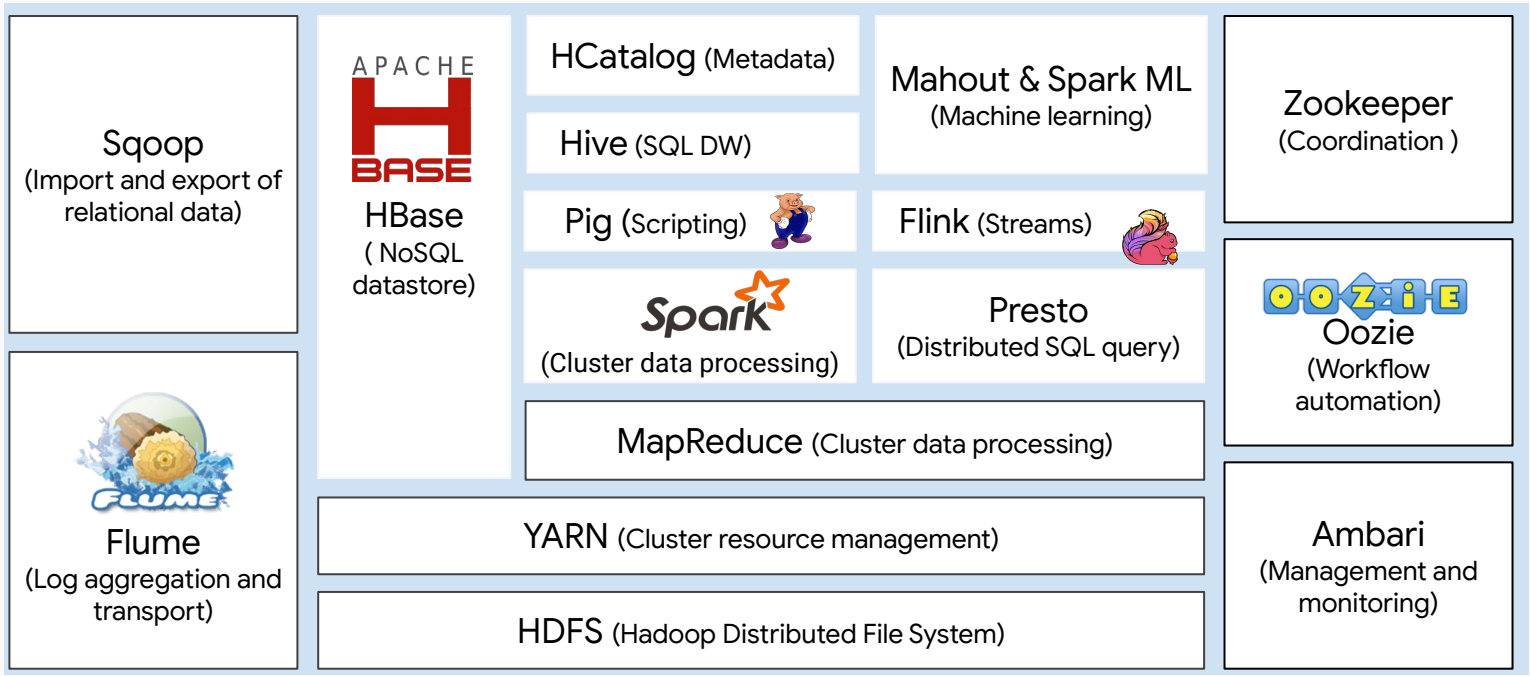
- It is a managed Hadoop MapReduce, Spark, Pig, and Hive service, to easily process big datasets at low cost
- Is a fast, easy to use, low cost and fully-managed service, powered by Google Cloud Platform



Dataproc


Dataproc cluster
Dataproc jobs
Data outputs
Spark & Hadoop OSS
Cloud Dataproc agent
Google Cloud services
Spark
PySpark
Spark SQL
MapReduce
Pig
Hive
Open-source software
Applications on the cluster
Dataproc jobs
GCP products

Dataproc - Hadoop Ecosystem



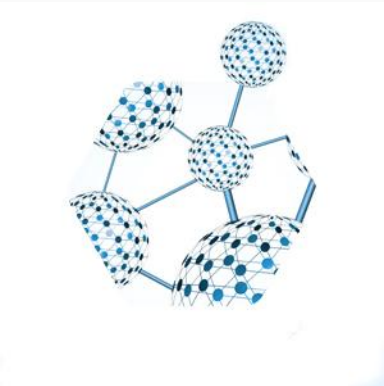
Dataproc

- **Apache Spark™** is a fast and general engine for large-scale data processing.
- The Spark Python API (**PySpark**) exposes the Spark programming model to Python.
- **Spark SQL** is Apache Spark's module for working with structured data.
- **MapReduce** is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster.
- **Apache Pig** is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.
- The **Apache Hive** ™ data warehouse software facilitates reading, writing, and managing large datasets residing in distributed storage using SQL. Structure can be projected onto data already in storage. A command line tool and JDBC driver are provided to connect users to Hive.

Dataproc: Features

Native Spark and Hadoop

Run Spark and Hadoop applications out of the box without modification.



Cloud Integrated

Integrated with Cloud Storage, Cloud Logging, Cloud Monitoring, and more.

Minute-by-Minute Billing

While active, Dataproc clusters are billed minute-by-minute.

Preemptible VMs

Dataproc clusters can make use of low-cost preemptible Compute Engine VMs.

Dataproc: Features

Anytime Scaling

Manually scale clusters up or down based on need, even when jobs are running.

Initialization Actions

Execute scripts on cluster creation to quickly customize and configure clusters.

Developer Tools

REST API and Integration with Google Cloud SDK for rapid development.

Easily Configured

Select between multiple Spark and Hadoop versions; configure properties easily.

Dataproc: Benefits

Low-cost

Enjoy lower total cost of ownership due to low prices and minute-based billing.

Superfast

Spend less time waiting for things to happen and more time hands-on with data.

Customizable

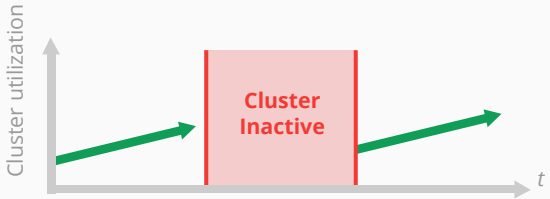
Customizable virtual machine types let you create right-sided clusters.

Easy

Vanilla Spark and Hadoop supported by purpose-built Cloud products.

Dataproc: Scaling

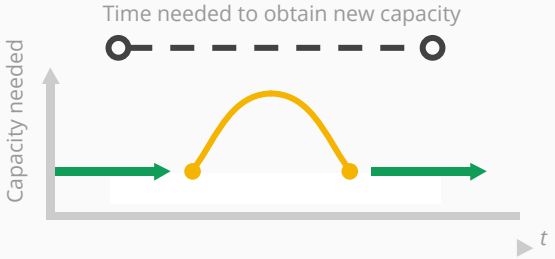
Idle clusters



Most traditional clusters are utilized only a portion of the time they're online

Idle cluster capacity (time, money, computing capacity) is wasted

Scaling inflexibility

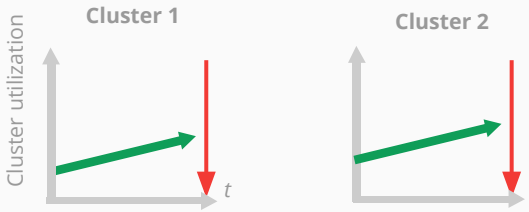


Job demands can be hard to predict, and scaling can take considerable time

Clusters may be constrained at the time when you need them most

Dataproc: Scaling

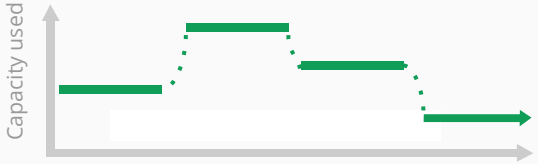
Anytime clusters



Clusters aren't idle. When work is done, simply turn off the cluster and create a new one when required.

Run clusters only when you need them

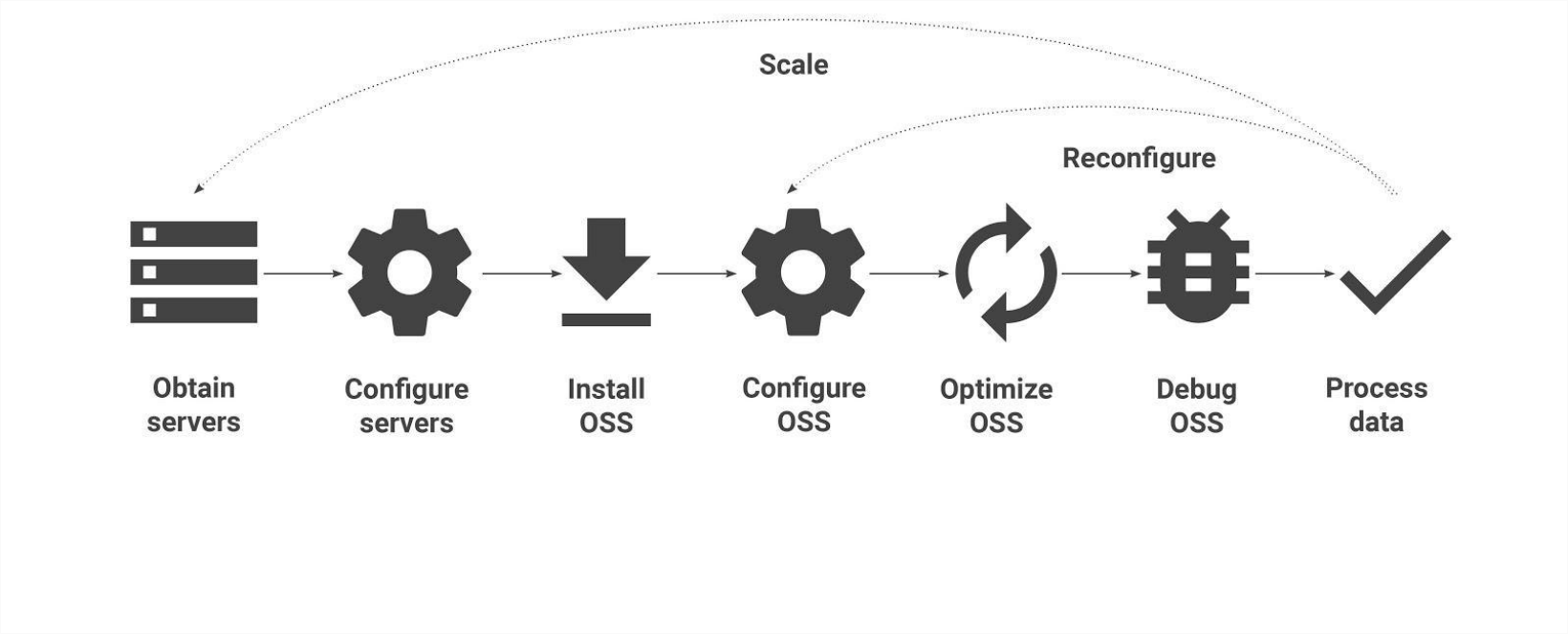
Flexible scaling



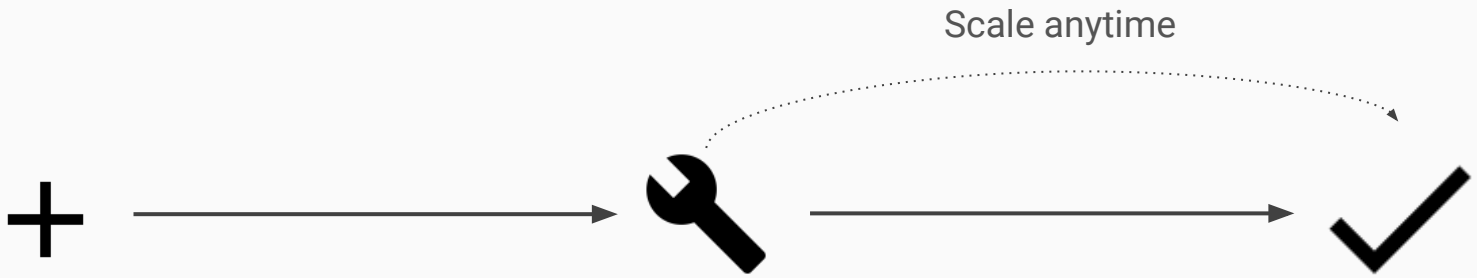
By scaling clusters at anytime, your jobs can get exactly the resources they need when required.

Clusters are the right size, anytime

Typical Spark and Hadoop deployments



Cloud Dataproc Clusters



Create cluster

(0 seconds)

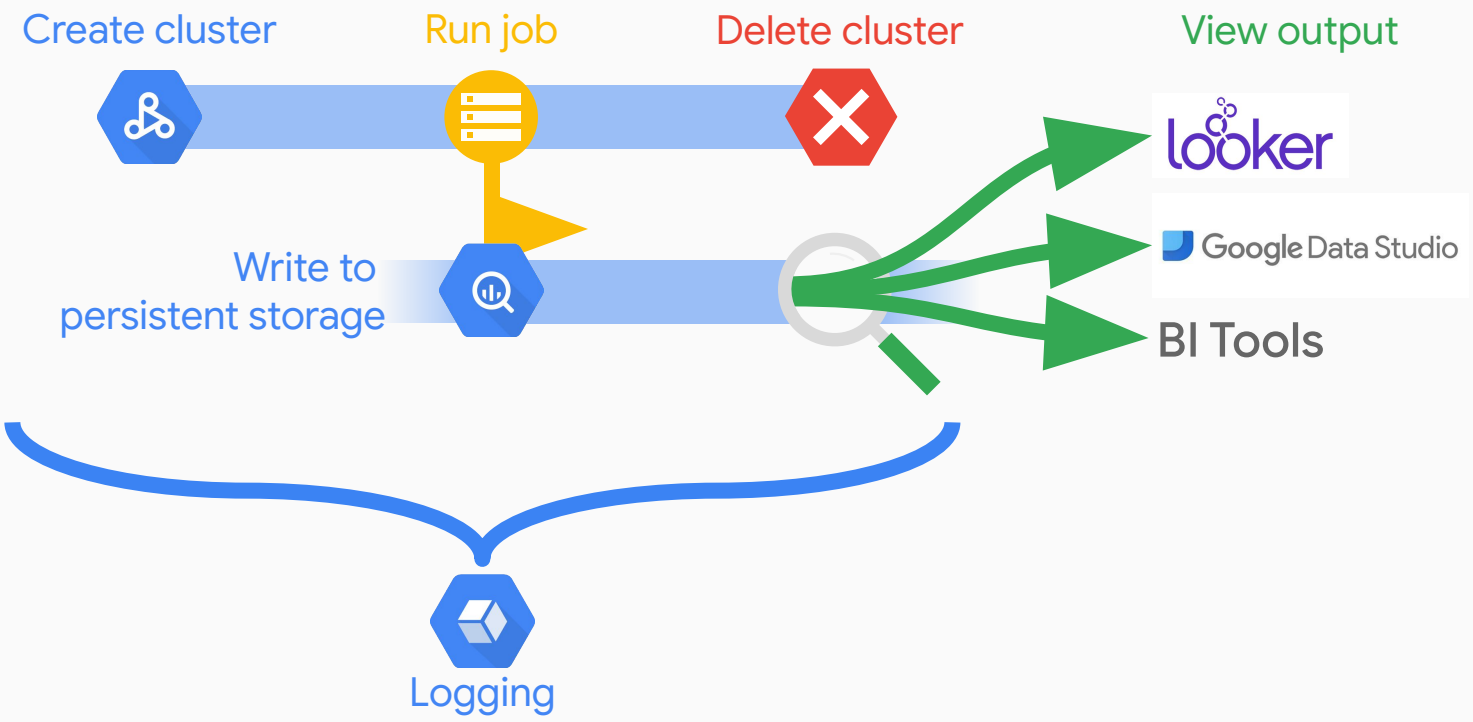
Configure cluster

(20 seconds)

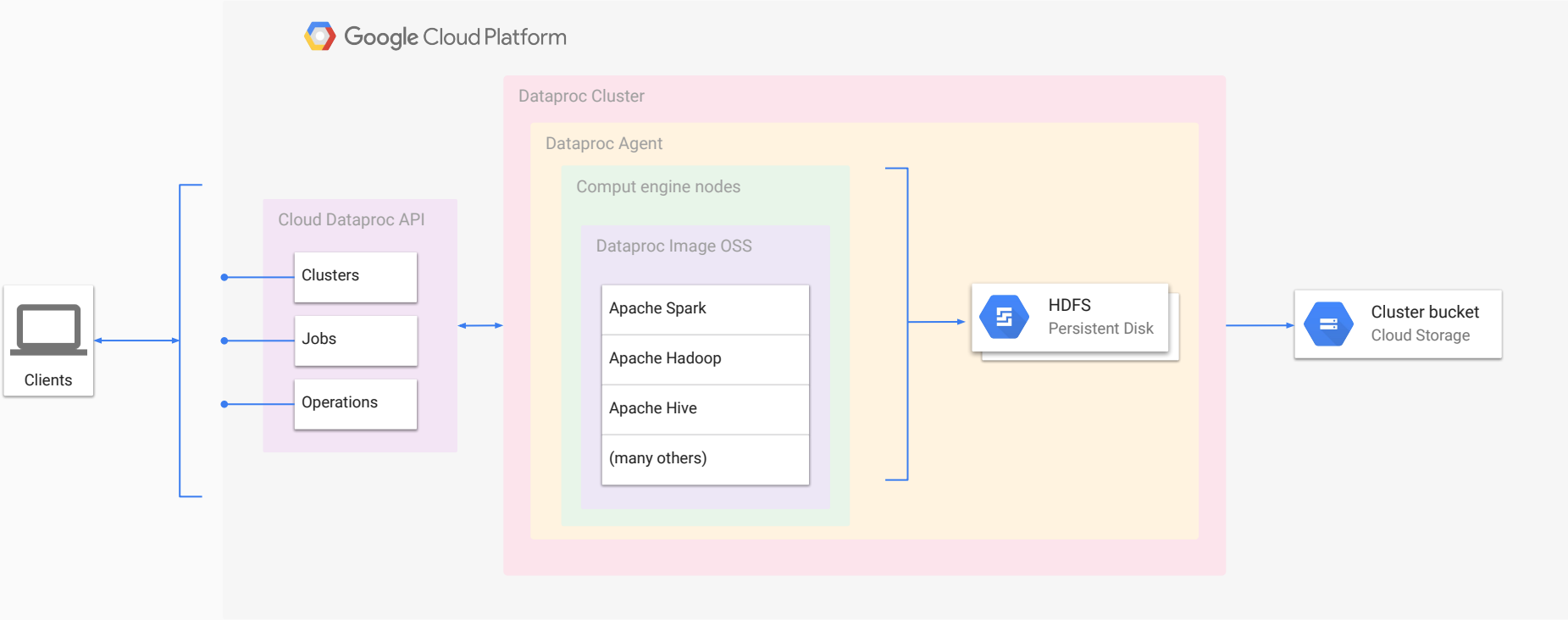
Use cluster

(90 seconds)

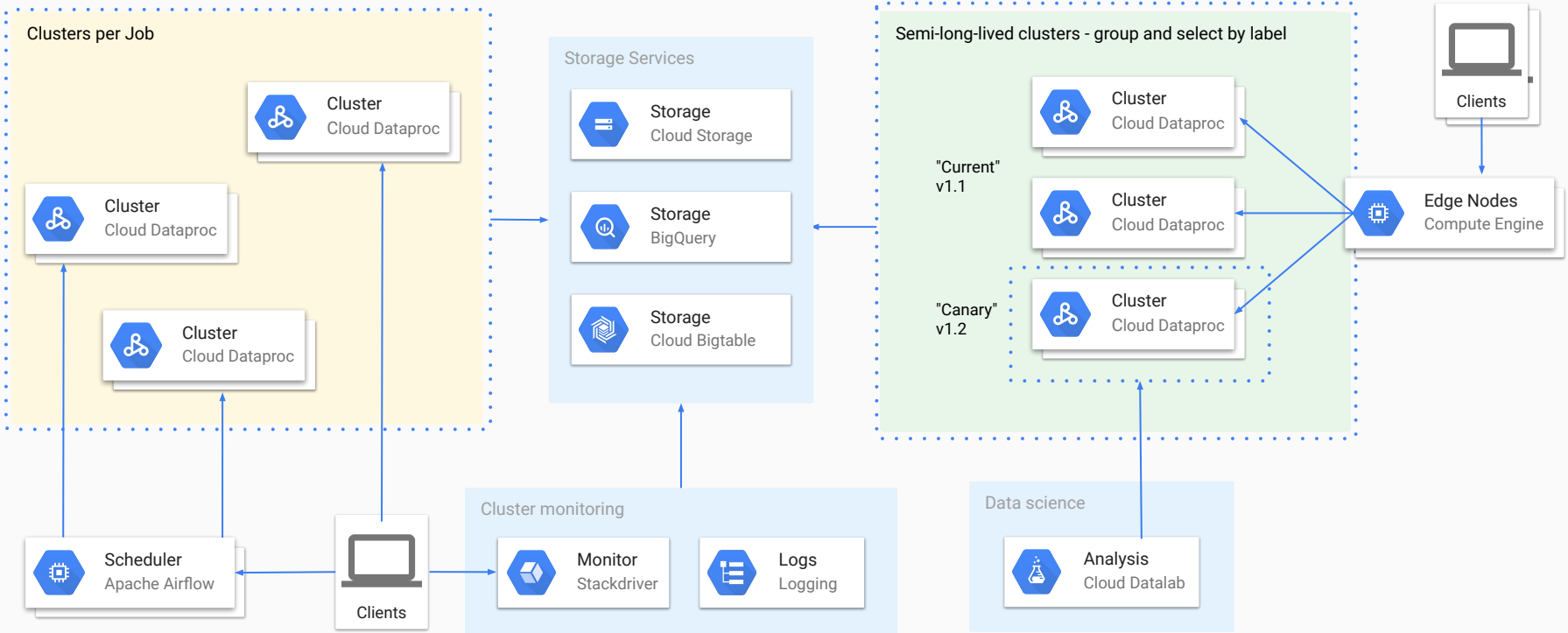
Dataproc: Use ephemeral clusters for one job's lifetime



Dataproc: Architecture



Dataproc: Architecture



Dataproc: Use Case #1: Log Processing

Need

A customer processes 50 gigabytes of (text) log data per day to produce aggregated metrics. They have used a persistent on-premise cluster to store and process the logs with MapReduce.

How Dataproc addresses this need

Google Cloud Storage can act as a landing zone for the log data for low-cost and high-durability storage. A Dataproc cluster can be created in less than 2 minutes to process this data with their existing MapReduce. Once finished, the Dataproc cluster can then be removed immediately.

Dataproc value

Instead of running all the time and incurring costs even when not used, Dataproc only runs to process the logs which saves money and reduces complexity.

Dataproc: Use Case #2: Ad-hoc Data Mining & Analysis

Need

A customer uses Spark standalone on one computer to perform data mining and analysis. The data is stored locally and they are using the Spark shell to examine the data along with Spark SQL.

How Dataproc addresses this need

Dataproc can create clusters that scale for speed and mitigate any single point of failure. Since Dataproc supports Spark, Spark SQL, and PySpark, they could use the web interface, Cloud SDK, or the native spark shell via SSH to perform their analysis safe from a single machine failure.

Dataproc value

Dataproc quickly unlocks the power of the cloud for anyone without added technical complexity. Running complex computations now take seconds instead of minutes or hours.

Dataproc: Use Case #3: Machine Learning

Need

A customer uses the Spark Machine Learning Libraries (MLlib) to run classification algorithms on very large datasets. They rely on cloud-based machines where they install, and customize Spark.

How Dataproc addresses this need

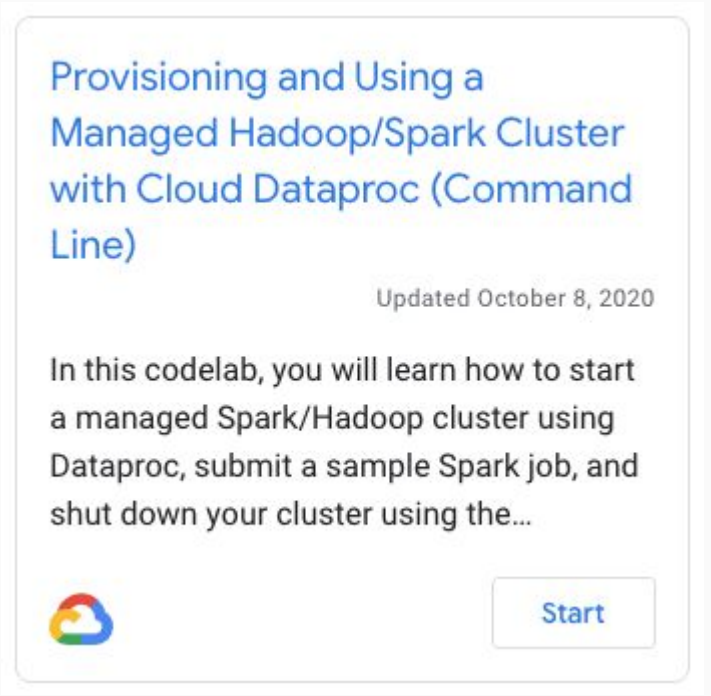
Since Spark and the MLlib installed on any Dataproc cluster, the customer can save time by quickly creating Dataproc clusters. Any additional customizations can be applied easily to the entire cluster via initialization actions. To keep an eye on workflows, they can use the built-in Cloud Logging and Monitoring.

Dataproc value

With Dataproc, resources spent on cluster creation and management can now be focused on the data. Integrations with new Google Cloud products unlock new features for Spark clusters.

Ejercicio DataProc

Inicio


Provisioning and Using a Managed Hadoop/Spark Cluster with Cloud Dataproc (Command Line)
Updated October 8, 2020
In this codelab, you will learn how to start a managed Spark/Hadoop cluster using Dataproc, submit a sample Spark job, and shut down your cluster using the...
Start

Bonus

Apache Spark and Jupyter Notebooks on Cloud Dataproc

Updated October 11, 2020

This lab shows you how to set up Apache Spark and Jupyter Notebooks on Cloud Dataproc using Optional Components and Component Gateway.

Start

Preprocessing BigQuery Data with PySpark on Dataproc

Updated October 8, 2020

This lab shows you how to use PySpark on Dataproc to load data from BigQuery and save it to Google Cloud Storage.

Start

Dataflow

Dataflow

Google Cloud Dataflow

- Dataflow is a unified programming model and a managed service for developing and executing a wide range of data processing patterns including ETL, batch computation, and continuous computation.
- Cloud Dataflow frees you from operational tasks like resource management and performance optimization.

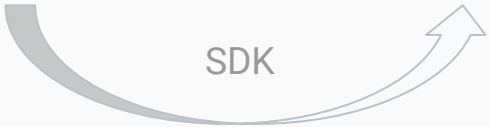
Dataflow

beam

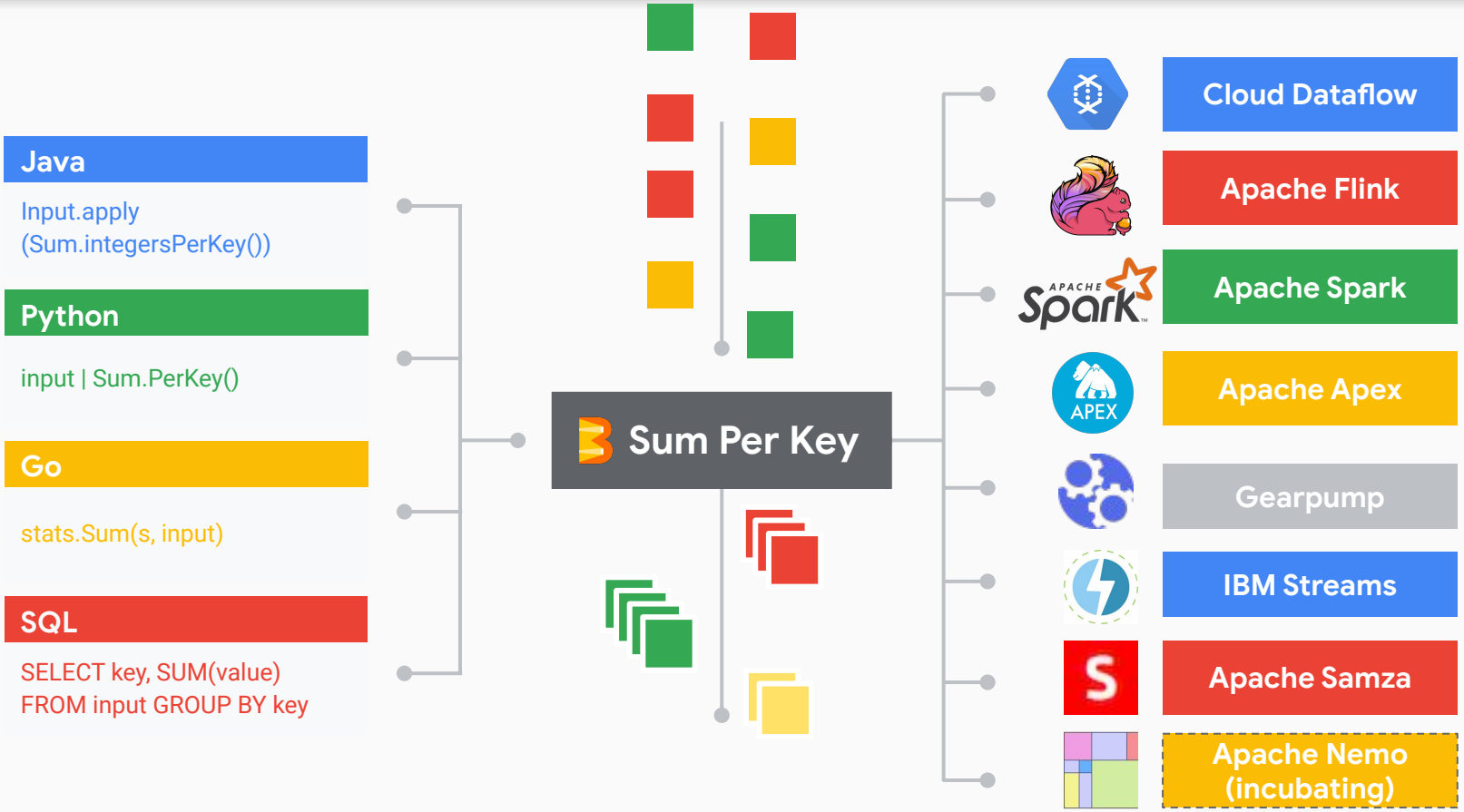
- Open source programming model
- Top Apache project by dev@ activity
- Unified batch and streaming
- Runner and language portability

Cloud Dataflow

- Serverless, fully managed data processing
- Exactly-once streaming semantics
- Automatic optimizations scale to millions of QPS
- State storage in Shuffle and Streaming Engine



The Beam Vision



Dataflow: Managing Massive Data In/Out

ETL

- Movement
- Filtering
- Enrichment
- Shaping

Analysis

- Reduction
- Batch computation
- Continuous computation

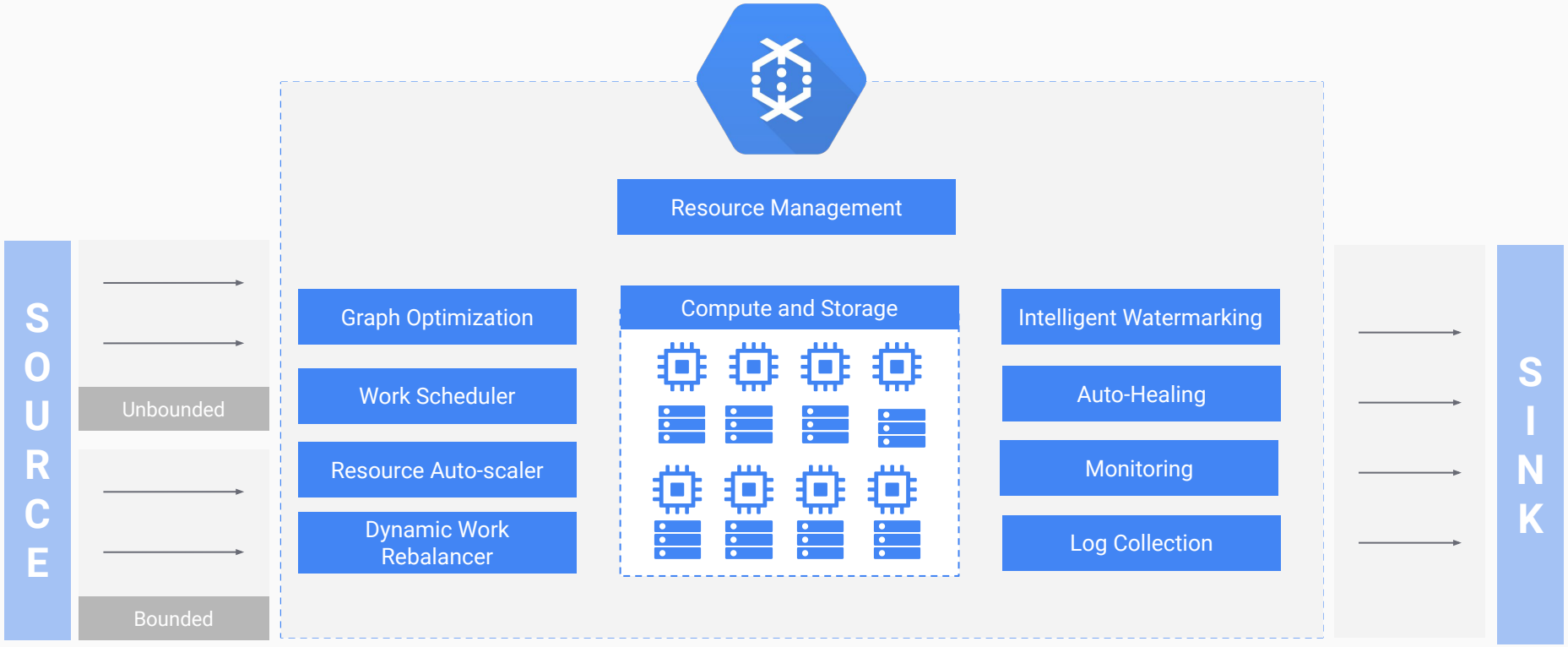
Orchestration

- Composition
- External orchestration
- Simulation

What is Cloud Dataflow?

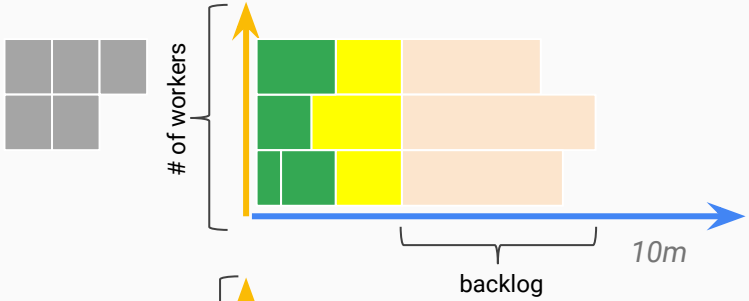
- Google Cloud Dataflow is a is a tool for developing and executing a wide range of data processing patterns—for example extract, transform, and load (ETL)—on very large data sets.
- Use Cloud Dataflow for nearly any kind of data processing task, encompassing both batch and streaming data processing.
- Dataflow can handle an unbounded or “infinite” data set from a continuously updating source such as Google Cloud Pub/Sub. That is, Dataflow can process practically any amount of data arriving at any time.
- Dataflow is particularly useful for embarrassingly parallel data processing tasks, in which the problem can be decomposed into many smaller bundles of data that can be processed independently, making it very fast (in the same way MapReduce works).

Cloud Dataflow: Under the Hood

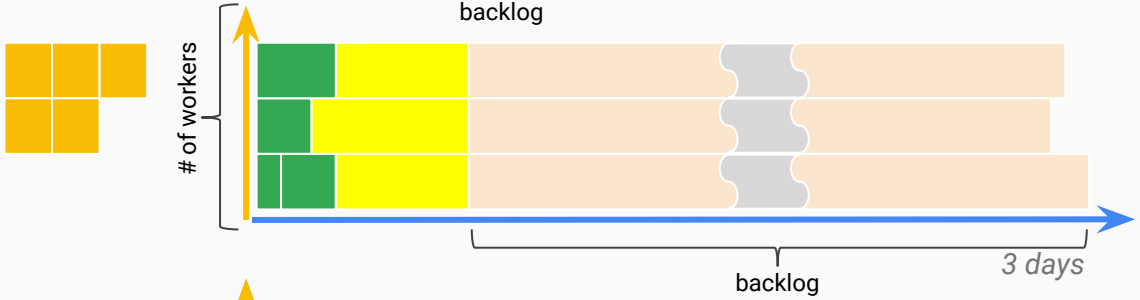


Cloud Dataflow: Autoscaling

Start off with **3** workers,
things are looking okay

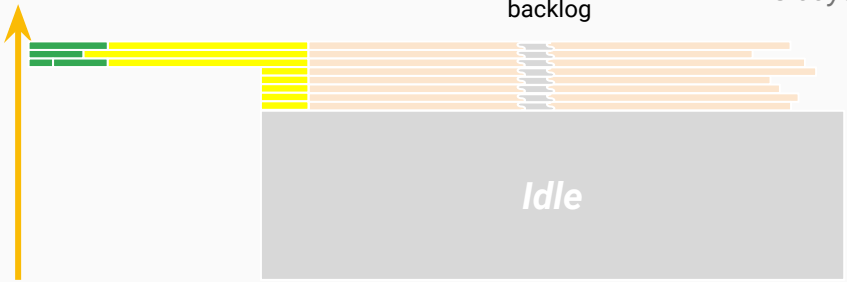


Re-estimation shows there's
orders of magnitude more work:
need **100** workers!



You have 100 workers
but you don't have 100 pieces of work!

...and that's really the most important part



Cloud Dataflow: why do people use it?

Usually **the goal** of big data efforts is to reduce the time required to answer questions for making faster, better decisions.

Examples of questions prospects want answered:

- “How many online sales did I make in the last hour due to advertising conversion?”
- “What was the average viewing time over the past seven days compared to last year?”
- “Which version of my web page do people like better?”
- “Which transactions look fraudulent?”

Cloud Dataflow **speeds up the rate at which questions can be answered** by:

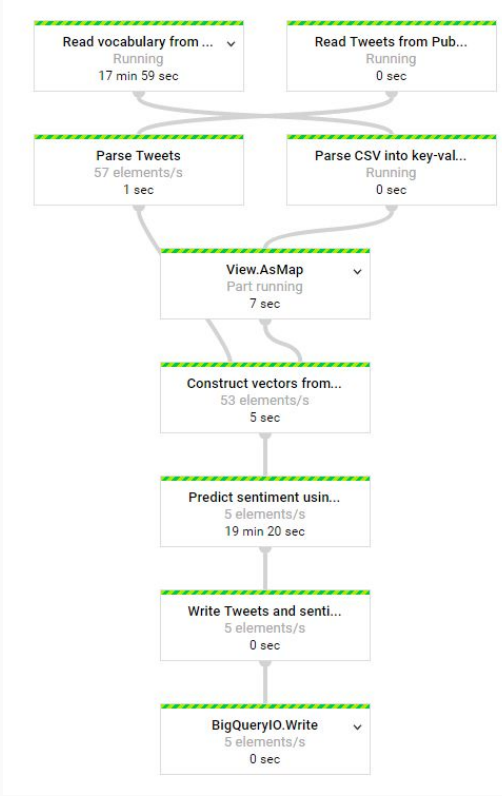
- Integrating data from multiple sources and preparing it for analysis.
- Analyzing event data streams using the Dataflow service.

How Cloud Dataflow works ?

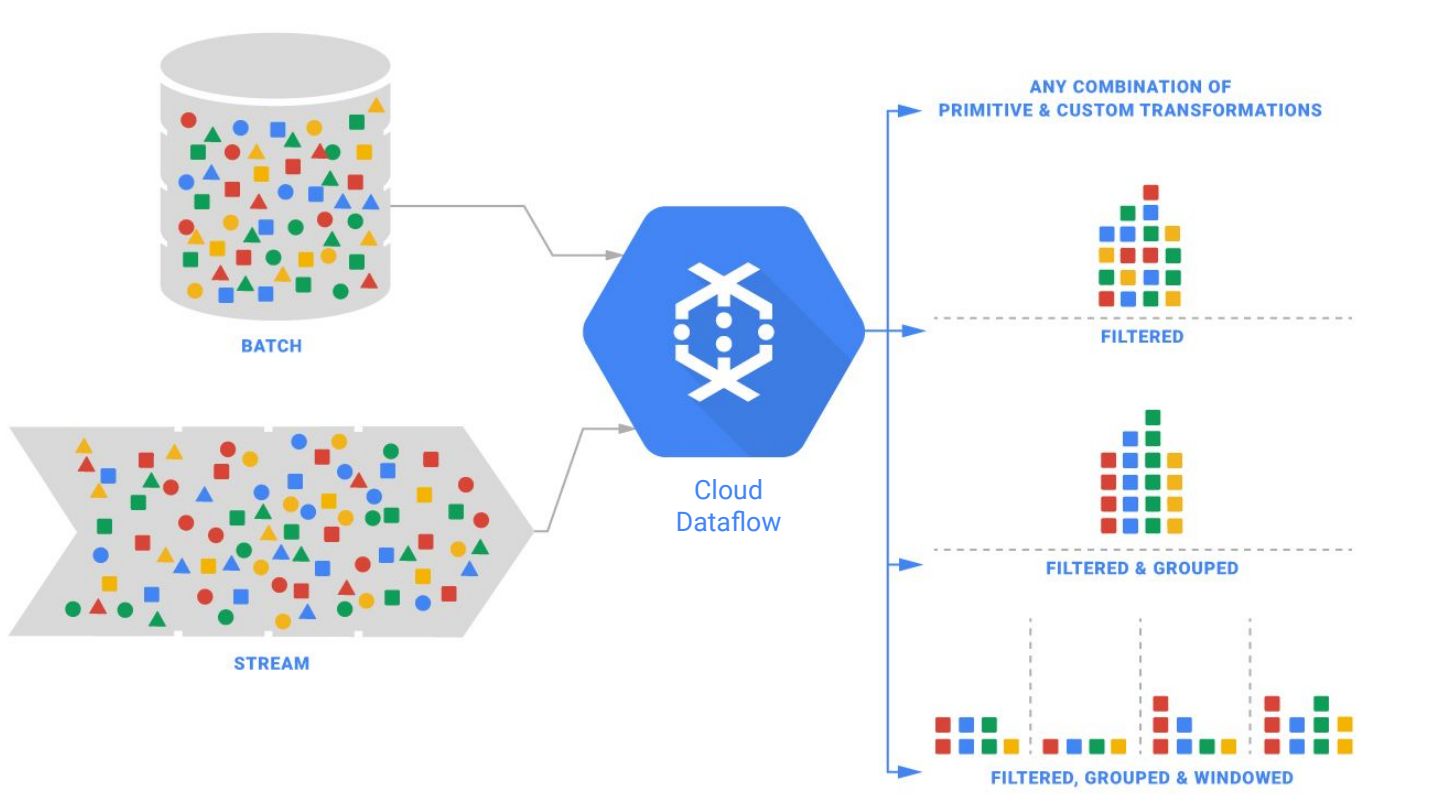
- Dataflow provides an easy way to create your data processing jobs. Each job is represented by a data processing **pipeline** that you create by writing a program.

- Each pipeline reads some input data, performs some transforms on that data to gain useful or actionable intelligence about it, and produces some resulting output data.

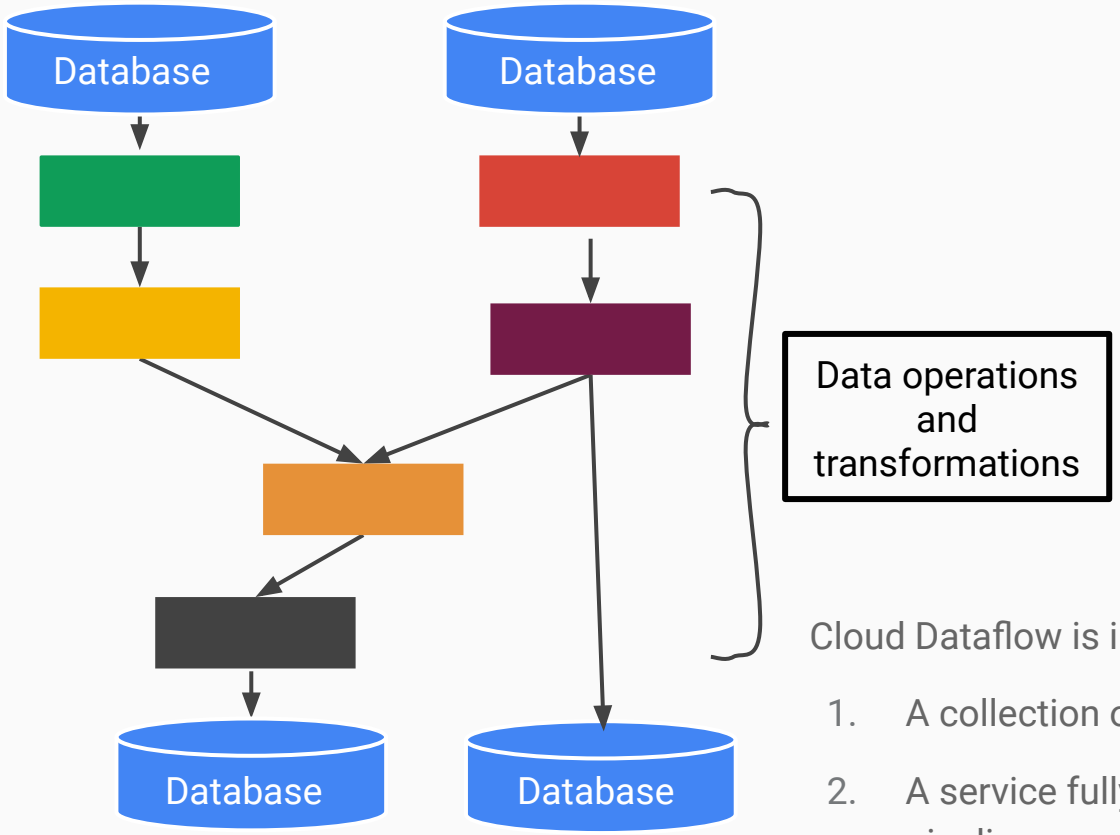
- A pipeline's transforms might include filtering, grouping, comparing, or joining data.



How Cloud Dataflow works ?



Dataflow: What are data pipelines?



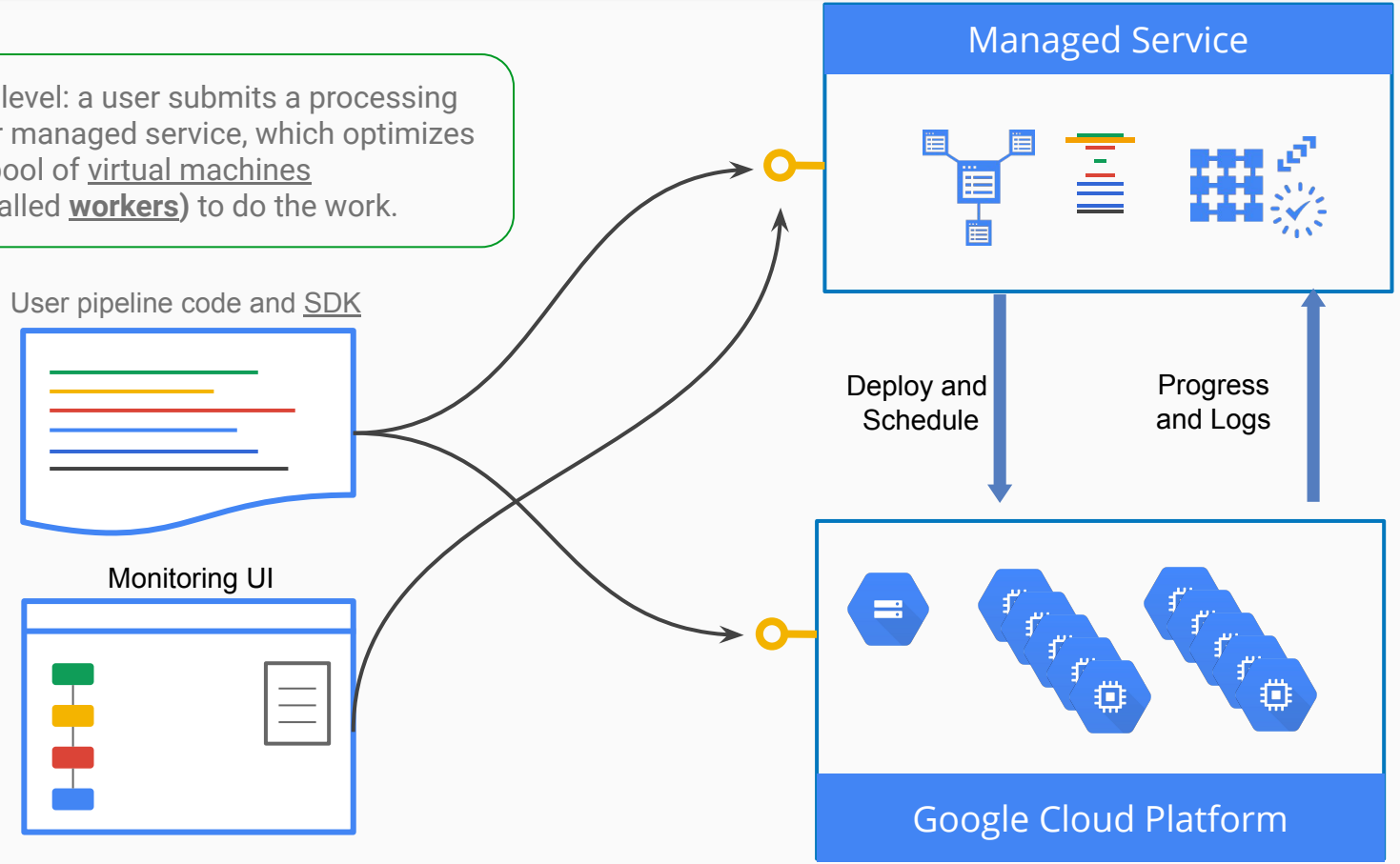
- A pipeline is a defined set of data processing transformations
- Optimized and executed as a unit
- Can include multiple inputs and multiple outputs
- Can perform many mathematical, logical, or transformation operations
- PCollections conceptually flows through the pipeline

Cloud Dataflow is implemented as two distinct elements:

1. A collection of software development kits.
2. A service fully managed by Google for running data pipelines.

Dataflow: Life of a Pipeline

At a very high level: a user submits a processing pipeline to our managed service, which optimizes it and runs a pool of virtual machines (sometimes called **workers**) to do the work.



Dataflow Python code

```python
def run(argv=None):
    """Main entry point; defines and runs the pipeline."""

    parser = argparse.ArgumentParser()
    parser.add_argument('--input',
                        dest='input',
                        default='gs://table/table.csv',
                        help='Input file to process.')

    parser.add_argument('--output',
                        dest='output',
                        default='project:dataset.table01',
                        help='Output BigQuery table for results specified as: PROJECT:DATASET.TABLE
    known_args, pipeline_args = parser.parse_known_args(argv)

    # Run the pipeline (all operations are deferred until run() is called).

    p = beam.Pipeline(options=PipelineOptions(pipeline_args))

    (p
    # Read the text file[pattern] into a PCollection.
    | 'Read from a File' >> beam.io.ReadFromText(known_args.input,skip_header_lines=1)
    | 'String To BigQuery Row' >> beam.ParDo(FormatDoFn())
    # Write the output using a "Write" transform
    | 'Write to BigQuery' >> beam.io.WriteToBigQuery(
        known_args.output,
        schema=TABLE_SCHEMA,
        create_disposition=beam.io.BigQueryDisposition.CREATE_IF_NEEDED,
        write_disposition=beam.io.BigQueryDisposition.WRITE_APPEND)

    # Run the pipeline (all operations are deferred until run() is called).
    )
    p.run()
```

Dataflow technologies that customers value

Automated input "sharding"

Removing the requirement to pre-sort input data

Dynamic work rebalancing

Removing the investment in manually balancing the load between resources

Auto-scaling of work resources

Removing the investment in manually scaling resources to match needs

Unified programming model

Developers can express the computation needs regardless of batch or streaming data input

Logical monitoring

Provides developers a logical view of pipeline behavior vs. a control view

On-demand resourcing

All resources are provided on demand, providing nearly limitless resource scale

Dataflow use cases

Batch data movement

Moving at rest data from one system to another, such as from Google Cloud Storage to BigQuery

Data reduction and enrichment

Reduce, compress, re-shape existing data into smaller, computed values, such as log files and geo tags

Continuous computation

Analyze real-time streaming inputs, such as click streams

Continuous data movement

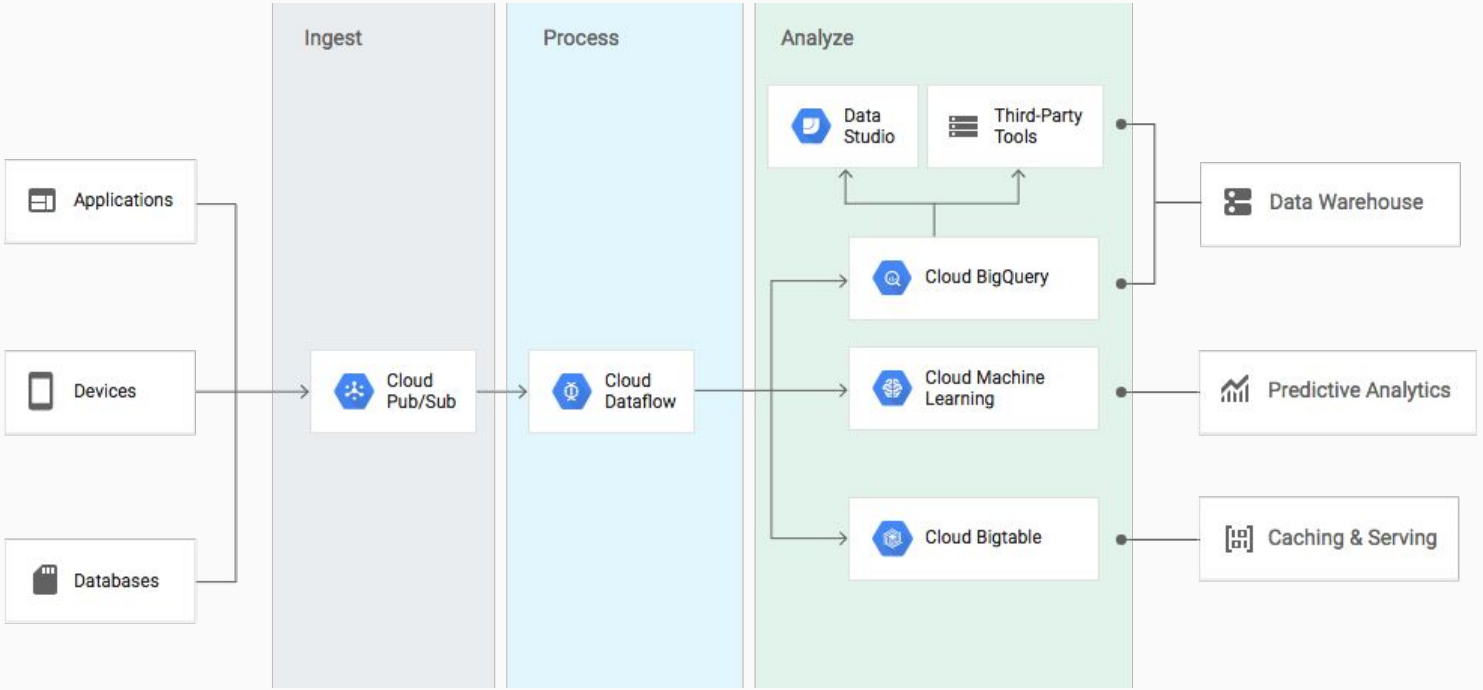
Real-time ETL over streaming inputs

Dataproc vs Dataflow

Recommended Workloads

WORKLOADS	CLOUD DATAPROC	CLOUD DATAFLOW
Stream processing (ETL)		✓
Batch processing (ETL)	✓	✓
Iterative processing and notebooks	✓	
Machine learning with Spark ML	✓	
Preprocessing for machine learning		✓ (with Cloud ML Engine)

GCP Stream Analytics: making event accessible and useful at scale



Ejercicio DataFlow

Inicio


Using Notebooks with Google Cloud Dataflow
Updated October 11, 2020
Setting up and running a notebook with interactive Beam
Start

Bonus

Running your first SQL statements using Google Cloud Dataflow

Updated October 11, 2020

The page explains how to use Dataflow SQL and create Dataflow SQL jobs.

Start

Run a big data text processing pipeline in Cloud Dataflow

Updated October 12, 2020

You will use Cloud Dataflow, create a Maven project with the Cloud Dataflow SDK, and run a distributed work count pipeline using the Google Cloud Platform...

Start

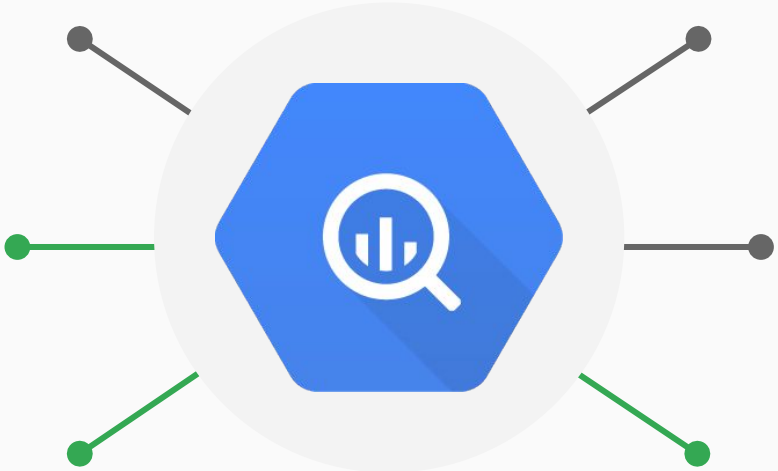
BigQuery

BigQuery: 100% serverless data warehouse

Google Cloud Platform's **enterprise data warehouse** for analytics

Built-in **ML and GIS**
Unique!

Fully managed and **serverless**
Unique!



Gigabyte- to **petabyte-scale** storage and SQL queries

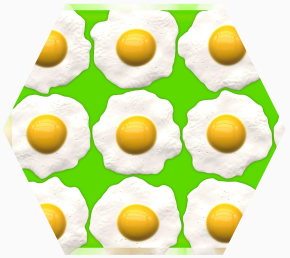
Encrypted, durable, and highly available

Real-time analytics on streaming data
Unique!

BigQuery: Is a great choice because

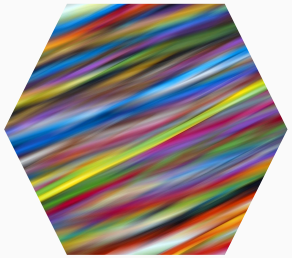
Near-real time analysis of massive datasets

No-ops; Pay for use



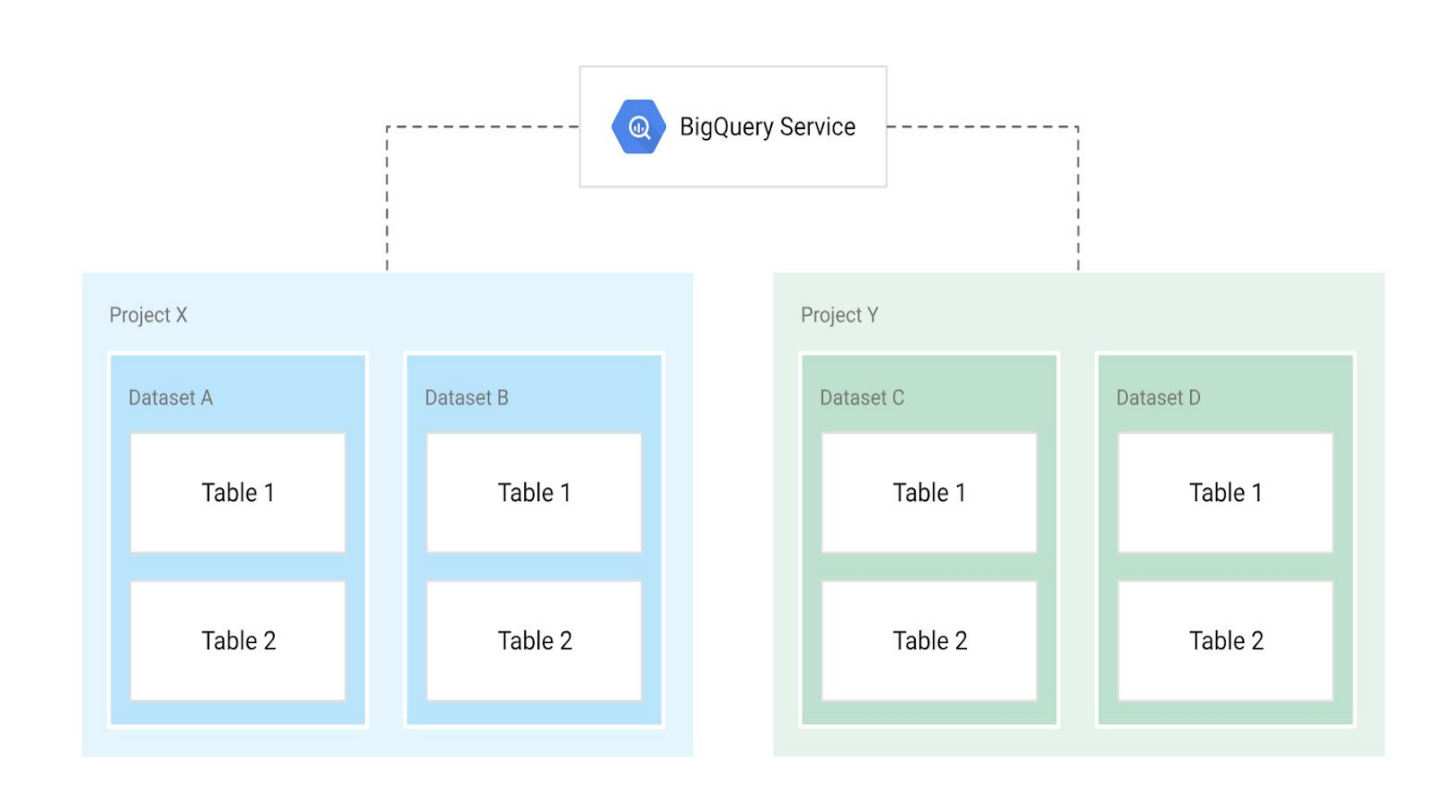
Durable (replicated), inexpensive storage

Immutable audit logs



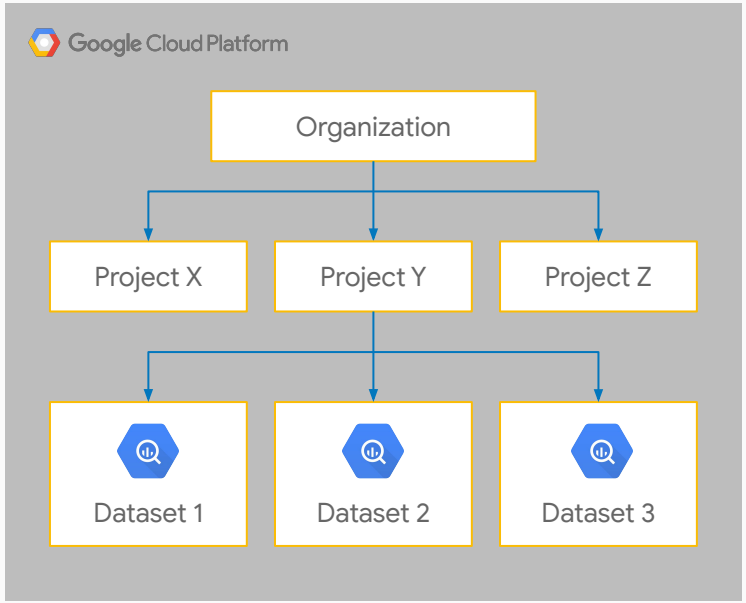
Mashing up different datasets to derive insights

BigQuery: Projects



BigQuery: uses Cloud IAM

Permission inheritance



Permission inheritance

User

Group

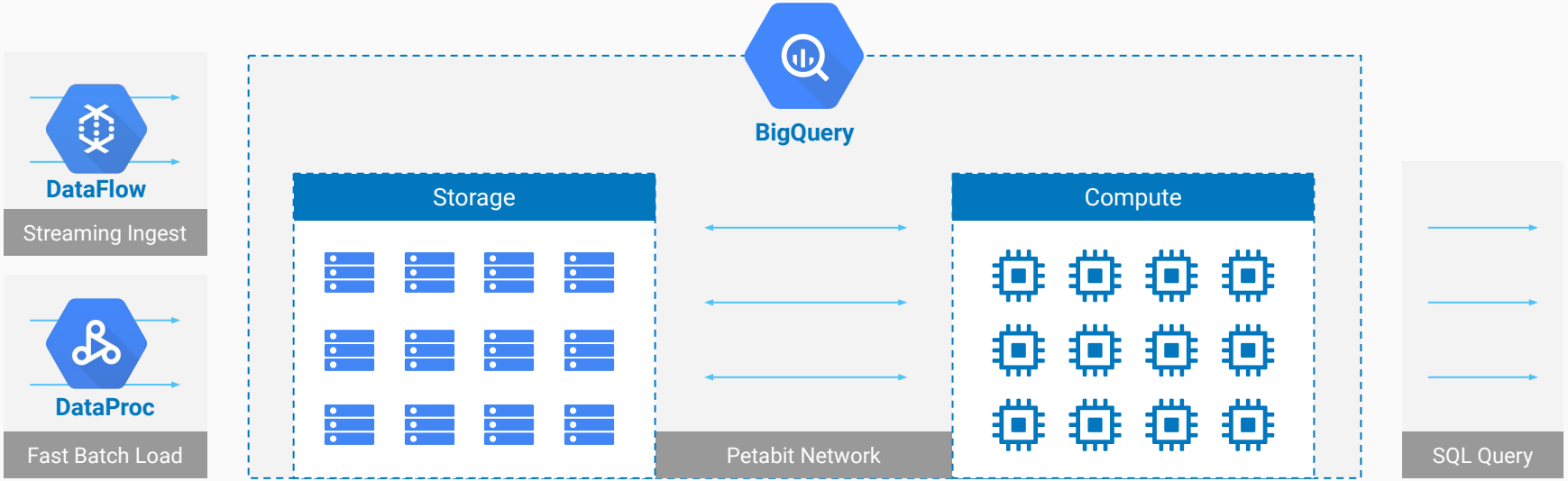
Domain

Public*

Authorized views*

BigQuery: Massive Parallel Processing Engine

BigQuery = **Massively Parallel Processing** query with the petabit network and thousands of servers



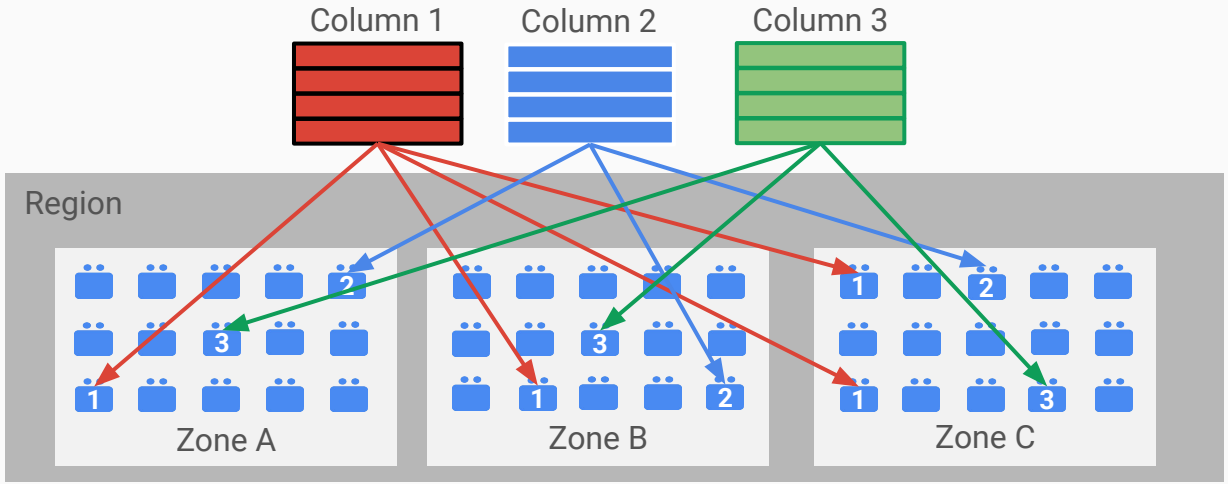
BigQuery: Managed Storage

Each column is stored in its own file

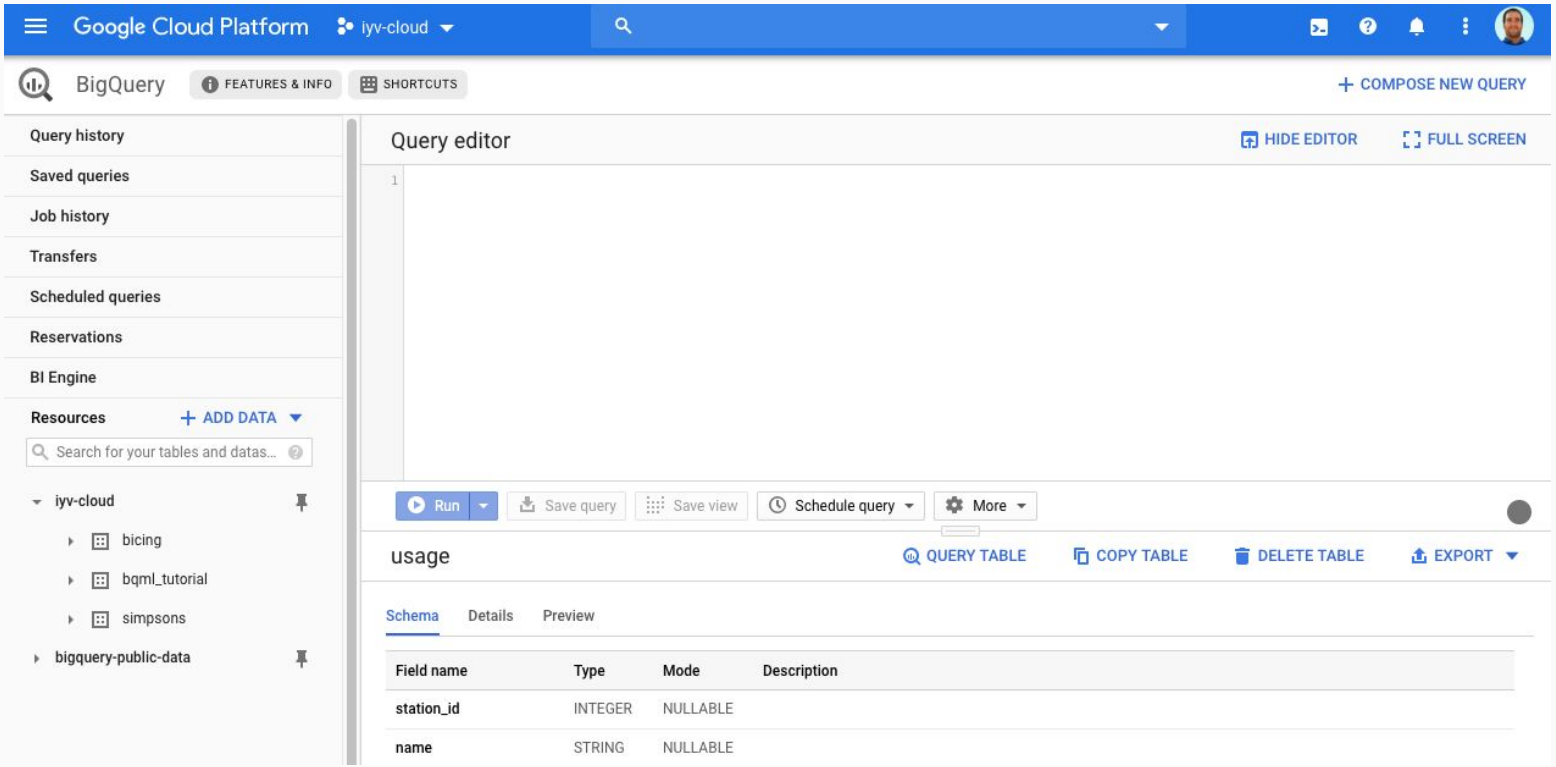
Each file is compressed and encrypted on disk

Storage is durable

Each file is replicated across datacenters



BigQuery - UI


Google Cloud Platform
iyv-cloud
BigQuery
FEATURES & INFO
SHORTCUTS
COMPOSE NEW QUERY
Query history
Saved queries
Job history
Transfers
Scheduled queries
Reservations
BI Engine
Resources
ADD DATA
Search for your tables and datas...
iyv-cloud
bicing
bqml_tutorial
simpsons
bigquery-public-data
Query editor
HIDE EDITOR
FULL SCREEN
1
Run
Save query
Save view
Schedule query
More
usage
QUERY TABLE
COPY TABLE
DELETE TABLE
EXPORT
Schema
Details
Preview
Field name
Type
Mode
Description
station_id
INTEGER
NULLABLE
name
STRING
NULLABLE

BigQuery - Sample Query

```
SELECT COUNT(*)
FROM
`bigquery-samples.wikipedia_benchmark.Wiki1B`
WHERE REGEXP_CONTAINS(title,"G.*o.*o.*g")
```

BigQuery - Query Plan Explanation

Query complete (5.1 sec elapsed, 37.2 GB processed)

Job information | Results | JSON | Execution details

For help debugging or optimizing your query, check our documentation. Learn more

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
5.1 sec	14 min 37.097 sec	1.93 KB	0 B

Stages		Worker timing: Wait	Read	Compute	Write		Rows
S00: Input	Avg:	792 ms	853 ms	2242 ms	3 ms	Input:	1,249,541,131
	Max:	1585 ms	1397 ms	2757 ms	38 ms	Output:	220
S01: Output	Avg:	1799 ms	0 ms	5 ms	14 ms	Input:	220
	Max:	1799 ms	0 ms	5 ms	14 ms	Output:	1

Bigquery: Pricing

Storage

- Amount of data in table
- Ingest rate of streaming data
- Automatic discount for old data

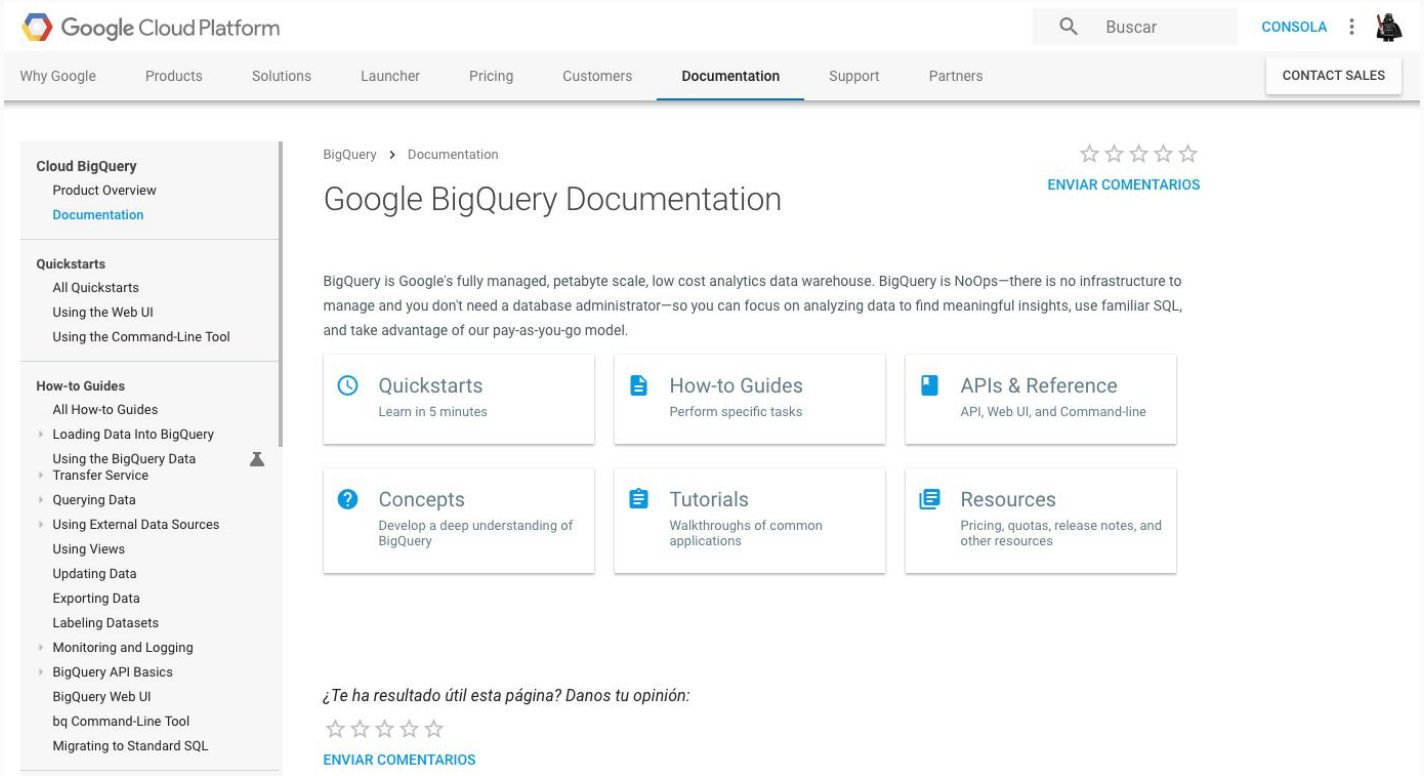
Processing

- On-demand OR Flat-rate plans
- On-demand based on amount of data processed
- 1 TB/month free
- Have to opt-in to run high-compute queries

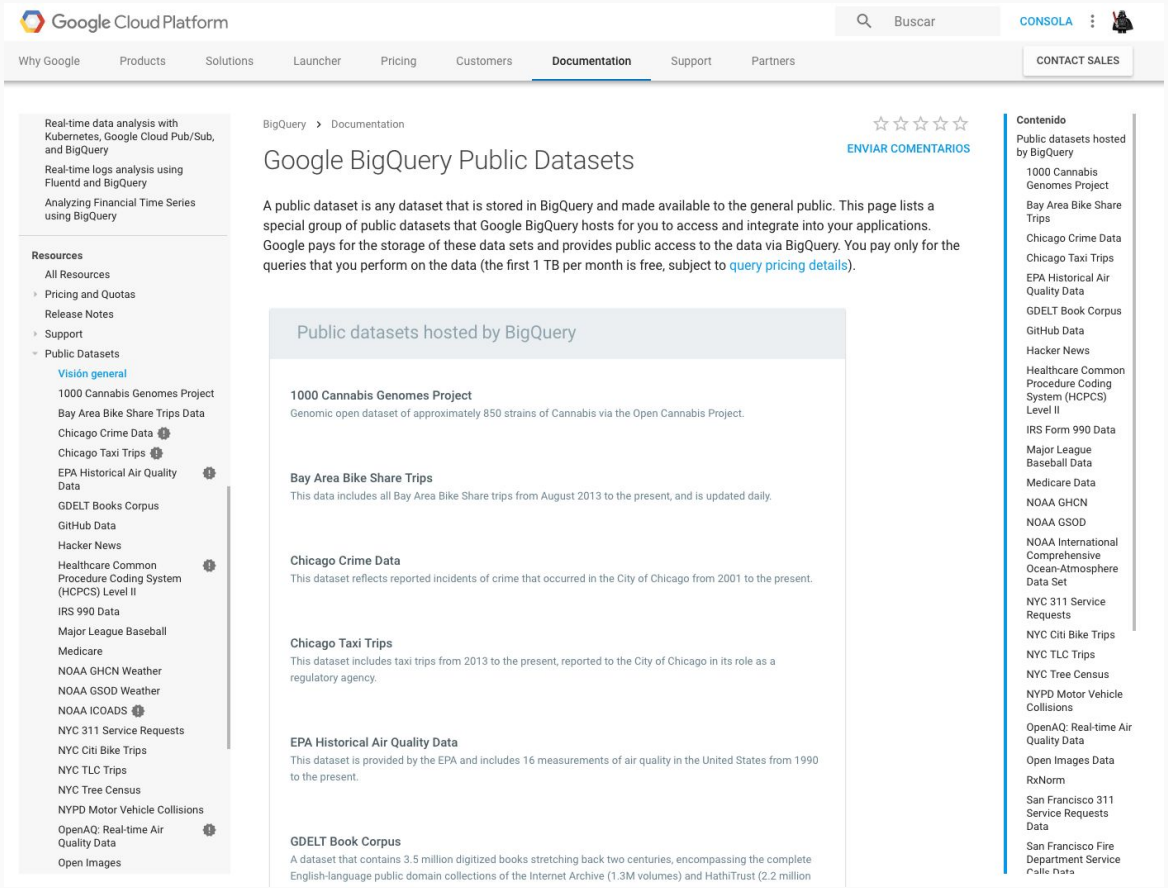
Free:

- Loading
- Exporting
- Queries on metadata
- Cached queries
- Queries with errors
- 10GB storage

Bigquery: Documents

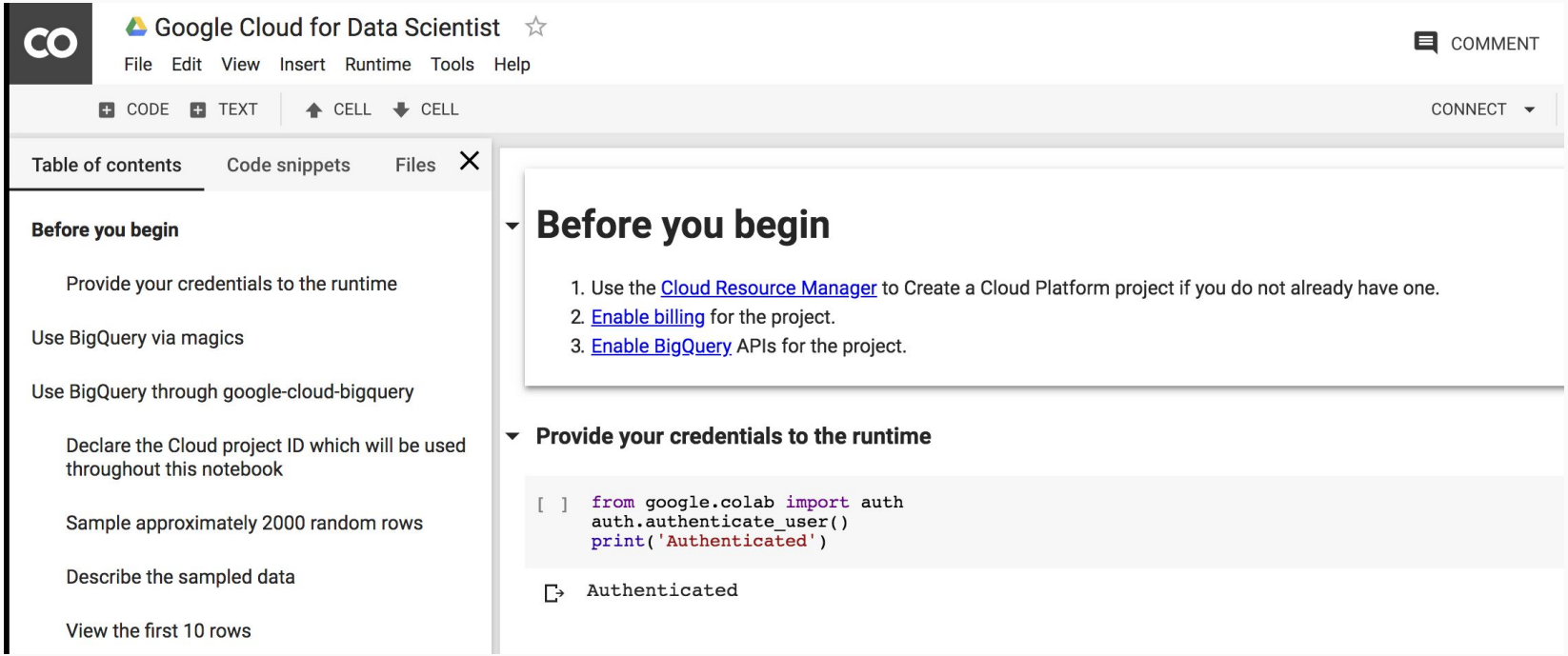


Bigquery: Public Dataset



AI Notebooks

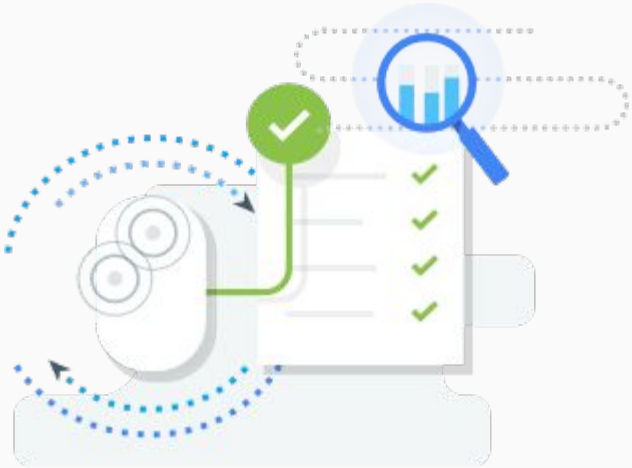
Colab


Google Cloud for Data Scientist
File Edit View Insert Runtime Tools Help
COMMENT
CODE TEXT CELL CELL
CONNECT
Table of contents
Code snippets
Files
Before you begin
Provide your credentials to the runtime
Use BigQuery via magics
Use BigQuery through google-cloud-bigquery
Declare the Cloud project ID which will be used throughout this notebook
Sample approximately 2000 random rows
Describe the sampled data
View the first 10 rows
Before you begin
1. Use the Cloud Resource Manager to Create a Cloud Platform project if you do not already have one.
2. Enable billing for the project.
3. Enable BigQuery APIs for the project.
Provide your credentials to the runtime
[ ]
from google.colab import auth
auth.authenticate_user()
print('Authenticated')
Authenticated

Instancias administradas de notebooks de JupyterLab

Notebooks de AI Platform es un servicio administrado que ofrece un entorno integrado de JupyterLab en el que los desarrolladores de aprendizaje automático y los científicos de datos pueden crear, en un solo clic, instancias que ejecutan JupyterLab y tienen preinstalados los marcos de trabajo de ciencia de datos y aprendizaje automático más recientes. Notebooks está integrado en BigQuery, Cloud Dataproc y Cloud Dataflow, lo que facilita pasar de la transferencia de datos al procesamiento previo y la exploración, y, luego, el entrenamiento y la implementación de modelos.

AI Notebooks



Ponte en marcha rápidamente

Puedes implementar instancias nuevas de JupyterLab con un clic y comenzar a analizar tus datos de inmediato. Cada instancia viene preconfigurada con versiones optimizadas de las bibliotecas más populares de ciencia de datos y aprendizaje automático, de modo que no tengas que preocuparte por crear y administrar las VM.

AI Notebooks

Usa frameworks de trabajo populares de código abierto

Notebooks no tiene una curva de aprendizaje, ya que usa la interfaz estándar de la industria de JupyterLab y tiene preinstaladas versiones optimizadas de bibliotecas populares, como TensorFlow, PyTorch, scikit-learn, Pandas, NumPy, SciPy y Matplotlib.

AI Notebooks

Scale on Demand

Puedes comenzar en un nivel bajo y escalar verticalmente con solo agregar CPU, RAM y GPU. Cuando tengas demasiados datos para una sola máquina, podrás cambiar sin interrupciones a servicios distribuidos como BigQuery, Cloud Dataproc, Cloud Dataflow y Entrenamiento y predicción de AI Platform.

AI Notebooks

AI Platform Notebooks te permite crear y administrar instancias de máquina virtual (VM) que vienen ya empaquetadas con JupyterLab. Las instancias de AI Platform Notebooks son compatibles con los marcos de trabajo de TensorFlow y PyTorch, y tienen un conjunto ya instalado de paquetes de aprendizaje profundo de Python y R. Puedes configurar instancias solo para CPU o habilitadas para GPU a fin de optimizar tu flujo de trabajo.

AI Platform Notebooks facilita el trabajo de crear y configurar una máquina virtual de aprendizaje profundo, ya que proporciona imágenes verificadas, optimizadas y probadas para el marco de trabajo elegido.

Tus instancias de notebook están protegidas con la autenticación y autorización de Google Cloud Platform (GCP) y se encuentran disponibles mediante una URL de instancia de notebook. Las instancias de notebook también se integran a GitHub, de modo que puedas sincronizar tu notebook con un repositorio de GitHub sin dificultades.

Ejercicio AI Notebooks

Inicio


Analyze Clinical Data using BigQuery and AI Platform Notebooks
Updated October 7, 2020
In this codelab, we demonstrate a solution to access and analyze clinical data in GCP using BigQuery and AI Platform Notebooks.
Start

Composer

Cloud Composer

Managed Apache Airflow to make workflow creation and management easy, powerful, and consistent.

Apache Airflow

- Apache Airflow is a top-level project in the Apache Software Foundation (ASF)
- Airflow has become very popular within the open-source community
- Designed to enable users to author, schedule and monitor workflows
- Workflows are authored as Directed Acyclic Graphs (DAGs), and can be configured in Python

Composer - Benefits

How Cloud Composer helps you

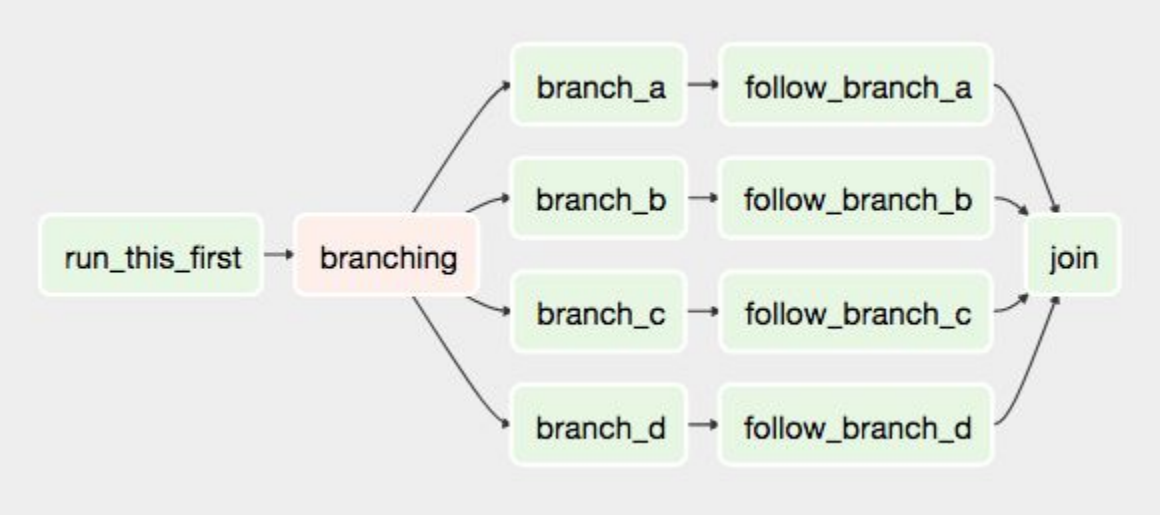
- **Comprehensive GCP integration:** Orchestrate your full GCP pipeline through Cloud Composer
- **Hybrid and multi-cloud environments:** Connect your pipeline and break down silos through a single orchestration service
- **Easy workflow orchestration:** Get moving quickly by coding your workflows using simple Python
- **Open source at its core:** Ensure freedom from lock-in through Apache Airflow's open source portability

Key Cloud Composer features

- Client tooling including the Google Developer Console and Cloud SDK
- Easy and controlled access to the Airflow web UI through Cloud Identity-Aware Proxy
- Streamlined Airflow runtime and environment configuration, such as plugin support
- Stackdriver logging and monitoring
- Identity access management (IAM)
- Simplified DAG (workflow) management
- Python (PyPi) package management

Composer - Airflow DAGs

DAG = Directed Acyclic Graph(Vertex, Edge) where
V = Task | DAG (SubDagOperator)
E = Dependency (e.g., v1 >> v2 : v1 is the upstream dependency of v2)



Understanding Airflow DAGs

Airflow DAGs are Python files with 5 main sections

1. Imports
2. Arguments
3. Setup
4. Tasks
5. Ordering

Imports

Import Python dependencies needed for the workflow.

```
import datetime
from airflow import DAG
from airflow import models
from airflow.contrib.operators import bigquery_operator
from airflow.utils import trigger_rule
```

Arguments

Define default and DAG-specific arguments. Can utilize environment variables and Jinja templating.

```
default_dag_args = {
    # Setting start date as yesterday starts the DAG
    # immediately when it is detected in the Cloud
    # Storage bucket.
    'start_date': yesterday,
    'email_on_failure': False,
    'email_on_retry': False,
    # If a task fails, retry it once after waiting
    # at least 5 minutes
    'retries': 1,
    'retry_delay': datetime.timedelta(minutes=5),
}

bq_dataset_name = 'airflow_bq_dataset_{{ ds_nodash }}'
bq_githib_table_id = bq_dataset_name + '.github_commits'
output_file = 'gs://my_bucket/github_commits.csv'
```

Setup

Give the DAG a name, configure the schedule, and set the DAG settings.

```
dag = DAG(
      'demo_dag',
      # Continue to run DAG once per day
      schedule_interval = datetime.timedelta(days = 1),
      default_args = default_dag_args
)
```

Tasks

Tasks that do two things:

1. Run a BigQuery query that exports the results to a new dataset
2. Exports the new dataset to Cloud Storage

```
# Perform query of Airflow GitHub commits
bq_airflow_commits_query =
bigquery_operator.BigQueryOperator(
      task_id = 'bq_airflow_commits_query',
      bql = """      SELECT commit, subject, message
      FROM [bigquery-public-data:github_repos.commits]
      WHERE repo_name contains 'airflow'
       """,
       destination_dataset_table = bq_githib_table_id
)

# Export query result to Cloud Storage
export_commits_to_gcs =
bigquery_to_gcs.BigQueryToCloudStorageOperator(
      task_id = 'export_airflow_commits_to_gcs',
      source_project_dataset_table = bq_githib_table_id,
      destination_cloud_storage_uris = [output_file],
      export_format = 'CSV'
)
```

Ordering

Set the order the tasks should be executed in. Ordering can be simple (all at once, in serial) or complex (graph.)

```
# Define task ordering
bq_airflow_commits_query >> export_commits_to_gcs
```

Composer - Steps

Composer - Operators

An operator describes a single task in a workflow.

Types:

- Action operator that performs or tells the system to perform an action
- Transfer operator that moves data from one system to another
- Sensor are a special type of operators that keeps running until a certain criteria is met (e.g. availability of data).

Airflow already supports BigQuery, Dataflow, Dataproc, Datastore, Cloud Storage, Pub/Sub and Cloud ML engine.

Composer - Architecture

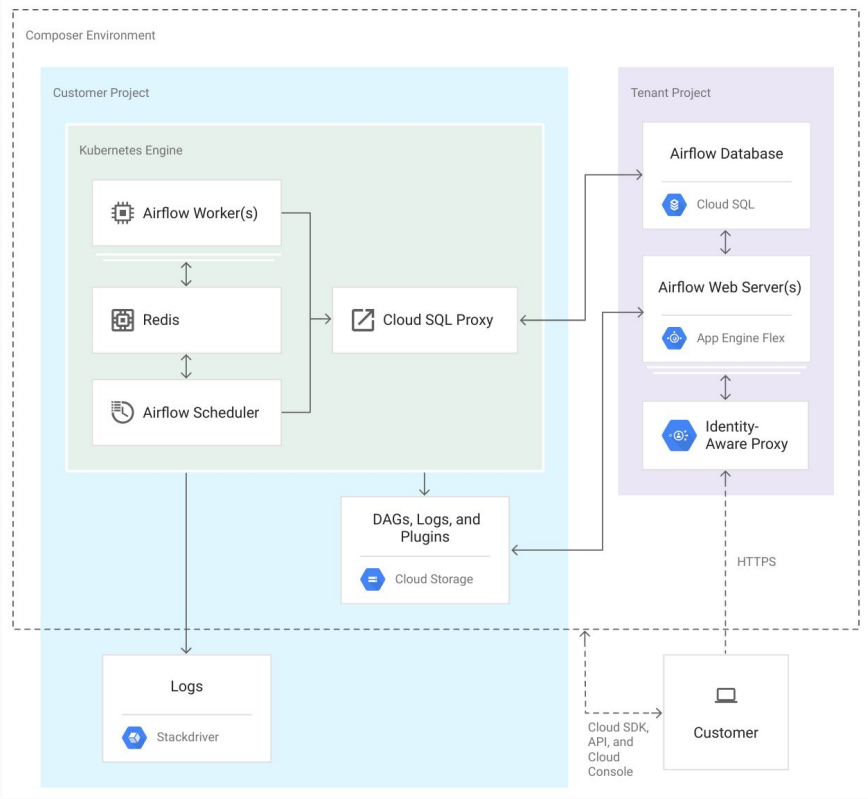
As discussed earlier, a Composer Environment runs fully managed instance of Apache Airflow in a GCP Project.

Airflow Workers and Scheduler are hosted in GKE and communicate/authenticate with the Airflow Database Google's Cloud SQL Proxy.

Airflow Web Server hosts the Airflow UI.

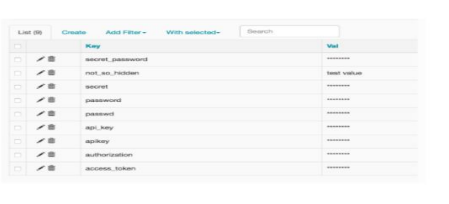
The DAGs, Logs and Plugins are all hosted in Cloud Storage in the customer project.

Logs can be viewed in Stackdriver and through Cloud Storage for DAG logs.

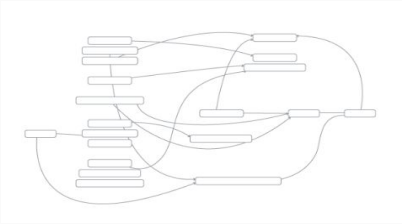


Composer - DAG Views on Airflow UI

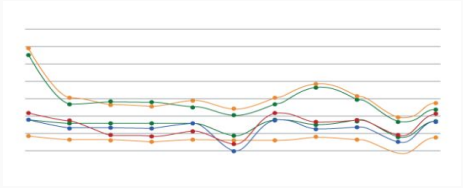
Tree views show a DAG across time, aiding in identifying blockers for late pipelines.



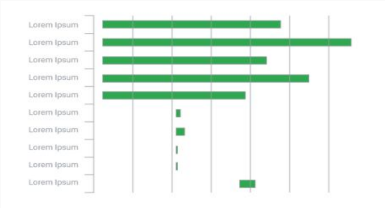
Variable views allow you to inspect the key-value pair of variables used in jobs



Graph views give a comprehensive look at dependencies and status of a work-flow.



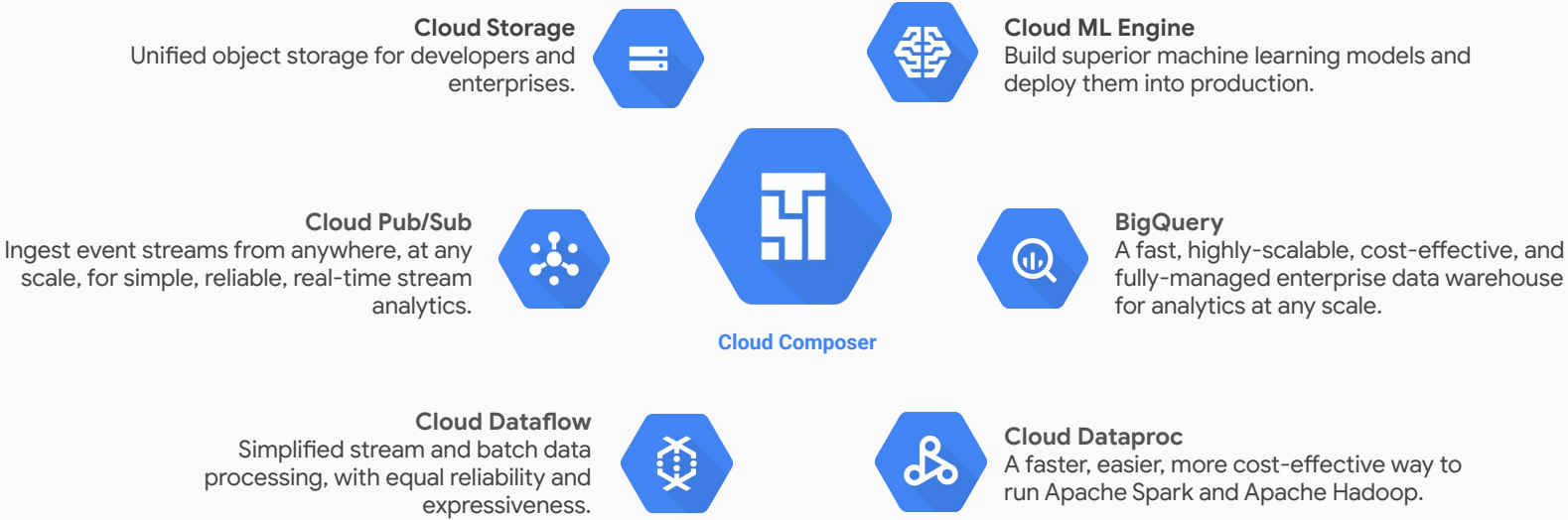
Gantt views allow you to analyze task duration to identify bottlenecks



Gantt views allow you to analyze task duration to spot bottlenecks

Code views give you the clearest view of what is happening within your pipeline

Composer - Integrations

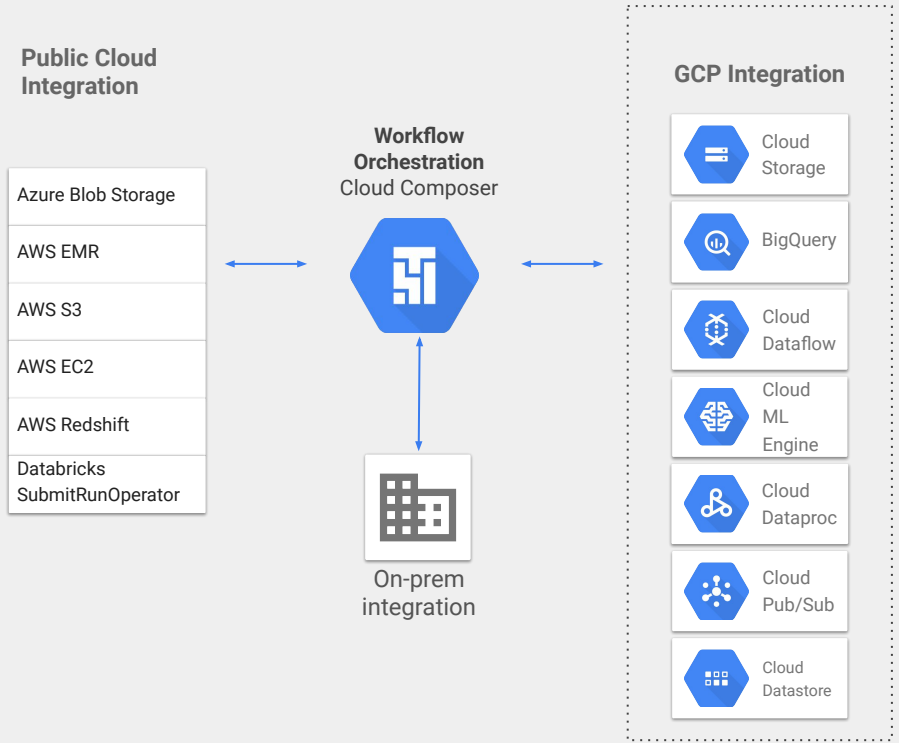


Composer - Connectors

Broad community of development
Diverse needs of the OSS community ensure that connectors are built to a variety of services, with more added on an ongoing basis

Easy-to-use experience
Drop do-it-yourself orchestration between services that hamper future flexibility by using established connectors

Connect data across environments
Take advantage of what you feel are best in breed solutions from a mix of your on-premises and cloud environments



Cloud Composer Data Workflow Orchestration

End-to-End GCP Integration

Author full workflows for GCP with a single tool, Integration with BigQuery, Dataflow, Dataproc, Datastore, Cloud Storage, Pub/Sub, Cloud ML Engine

Hybrid and Multi-Cloud Environments

Break down data silos and unite data sources in one tool, Connect data pipeline between different clouds and different on-premises tools

Easy Workflow Orchestration

Author workflows in Python, One-click deployment and auto-synchronization, Rich library of connectors, Easy troubleshooting and increased reliability through graph-based (DAG) UI

Open Source at its Core

Built upon Apache Airflow, Freedom from lock-in, Community continuing contributions for non-GCP platforms GCP contributing back our developments

Ejercicio Composer

Inicio


Running a Hadoop wordcount job on a Dataproc cluster
Updated October 12, 2020
This codelab shows you how to create and run an Apache Airflow workflow in Cloud Composer that completes the following tasks:
Start

Data Fusion

Cloud Data Fusion

Cloud Data Fusion is a fully-managed, cloud native, enterprise data integration service for quickly building and managing data pipelines.

Data Fusion - CDAP

Cloud Data Fusion is Google's cloud native manage service for enterprise data integration powered by CDAP.

CDAP

cdap.io

Google acquired Cask - May 14th 2018

CDAP is an 100% open-source framework to build data analytics applications for on-premise and cloud.

API | Runtime | Pre-built Accelerators | Library | Apache 2.0 License | In production at scale for +4.5 years / Dev +7 years

Developer, Data scientist and Business Analyst

1

Need to cleanse, match, de-dupe, blend, transform, partition, transfer, standardize, automate, & monitor

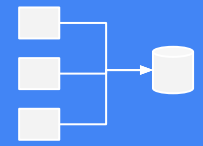
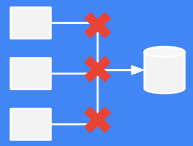
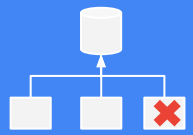
2

Use Cloud Data Fusion to visually build integration pipeline, test, debug and deploy

3

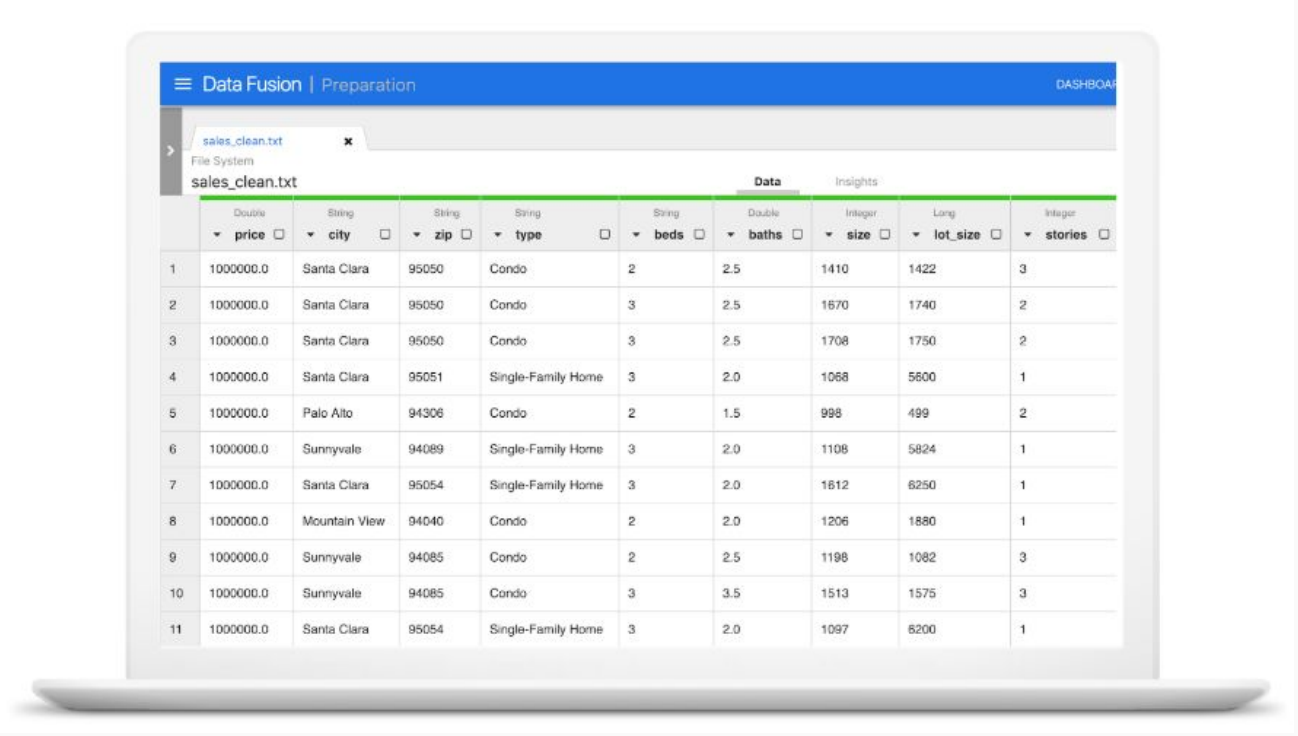
Run it at scale on GCP, operationalize (monitor, report) pipelines, inspect rich integration metadata

Data Fusion

Business Imperatives	Improve Decisions & Regulatory Compliance	Modernize Business and Reduce IT costs	Acquire and Merge	Increase Business Profitability	Extend or Migrate or both to Cloud
IT Initiatives	Business Intelligence / Analytics / AI	Legacy Retirement	Data and Application Consolidation	Customer, Supplier, Product Hubs	iPaaS, SaaS
Data Fusion Use cases	 Data Warehouse	 Data Migration	 Data Consolidation	 Master Data Management	 Data Consistency
Data Services					

Data Fusion - Benefits

- Integrate with any data
- Increase productivity
- Reduce complexity
- Increase flexibility



	price	city	zip	type	beds	baths	size	lot_size	stories
1	1000000.0	Santa Clara	95050	Condo	2	2.5	1410	1422	3
2	1000000.0	Santa Clara	95050	Condo	3	2.5	1670	1740	2
3	1000000.0	Santa Clara	95050	Condo	3	2.5	1708	1750	2
4	1000000.0	Santa Clara	95051	Single-Family Home	3	2.0	1068	5600	1
5	1000000.0	Palo Alto	94306	Condo	2	1.5	998	499	2
6	1000000.0	Sunnyvale	94089	Single-Family Home	3	2.0	1108	5824	1
7	1000000.0	Santa Clara	95054	Single-Family Home	3	2.0	1612	6250	1
8	1000000.0	Mountain View	94040	Condo	2	2.0	1206	1880	1
9	1000000.0	Sunnyvale	94085	Condo	2	2.5	1198	1082	3
10	1000000.0	Sunnyvale	94085	Condo	3	3.5	1513	1575	3
11	1000000.0	Santa Clara	95054	Single-Family Home	3	2.0	1097	6200	1

Data Fusion - Development

Rich graphical interface

100+ plugins - connectors, transforms & actions

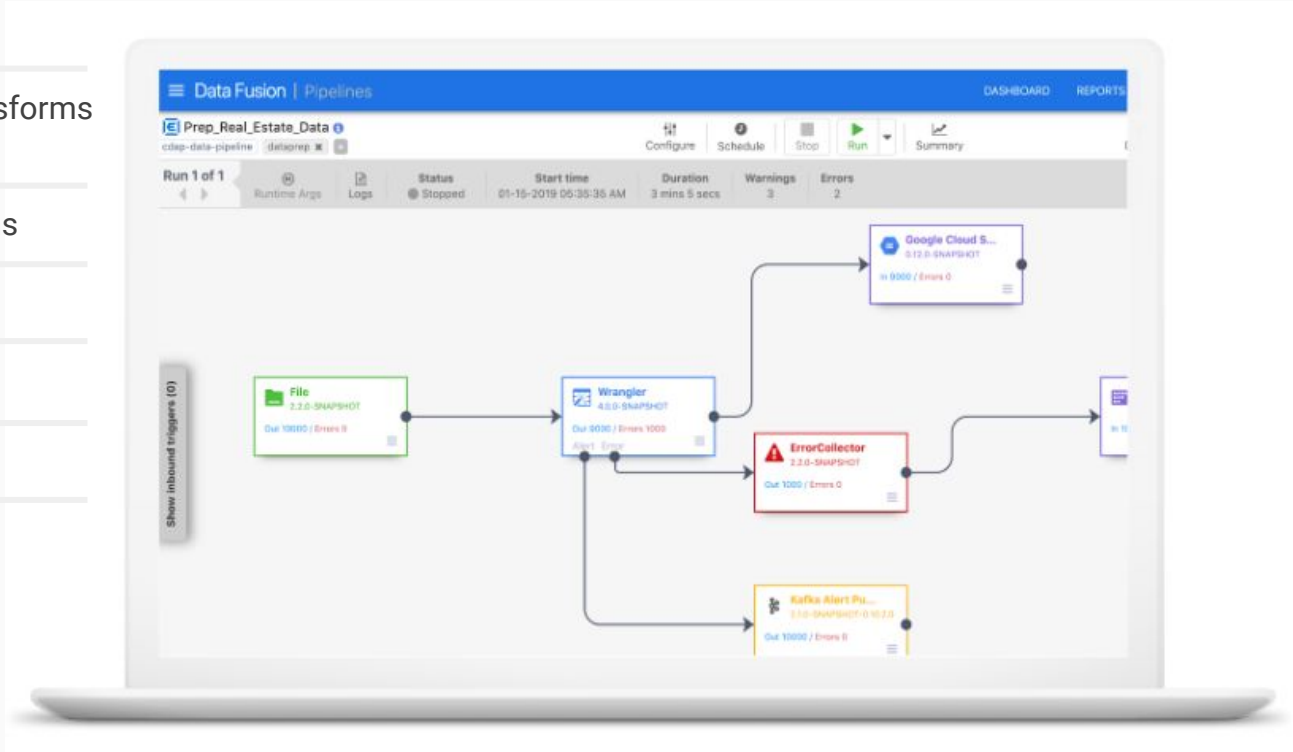
Code free visual transformations

1000+ transforms, data quality

Test and debug pipeline

Pre-built pipelines

Developer SDK



Data Fusion - Execution & Management

Cloud Dataproc - Batch & Realtime

Schedules and Triggers

Control and data flow pipelines support

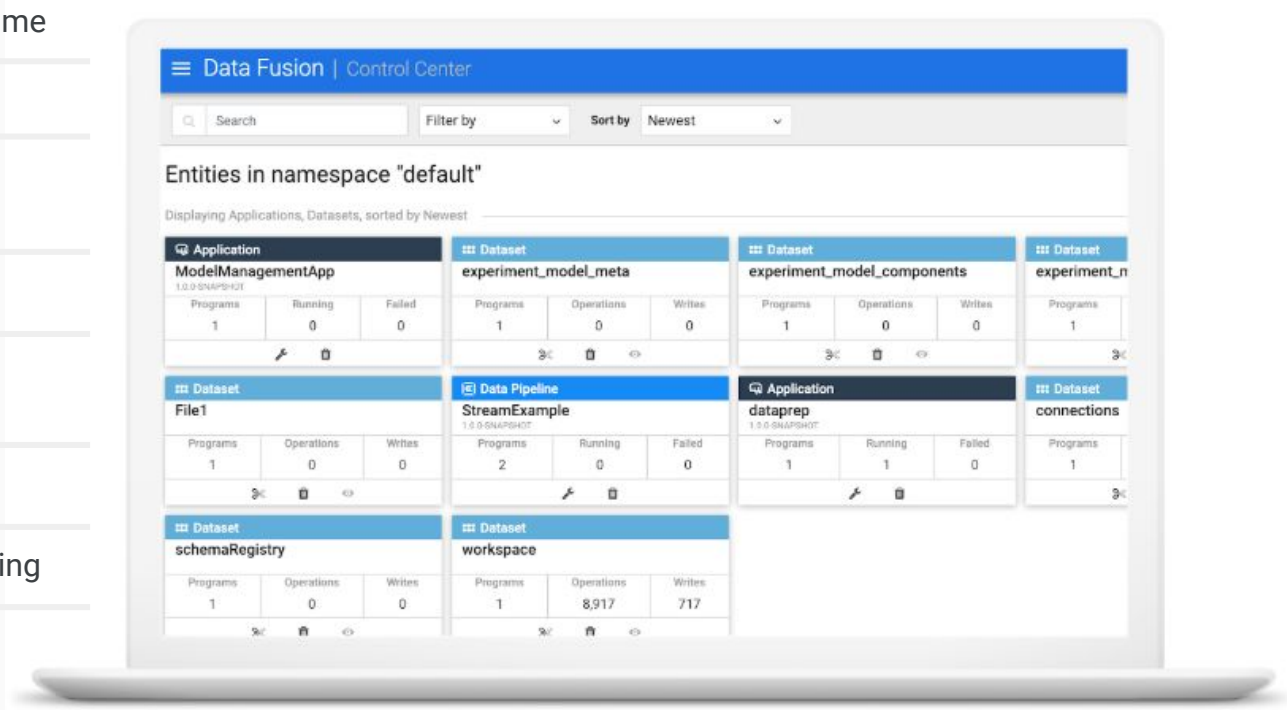
Dashboard and Reports

Execute existing **Spark** and **MapReduce** jobs

Private IP support

Service Networking - VPC Peering

REST API and CLI



Data Fusion - Integration Metadata

Tags and Properties support

Pipeline

Dataset

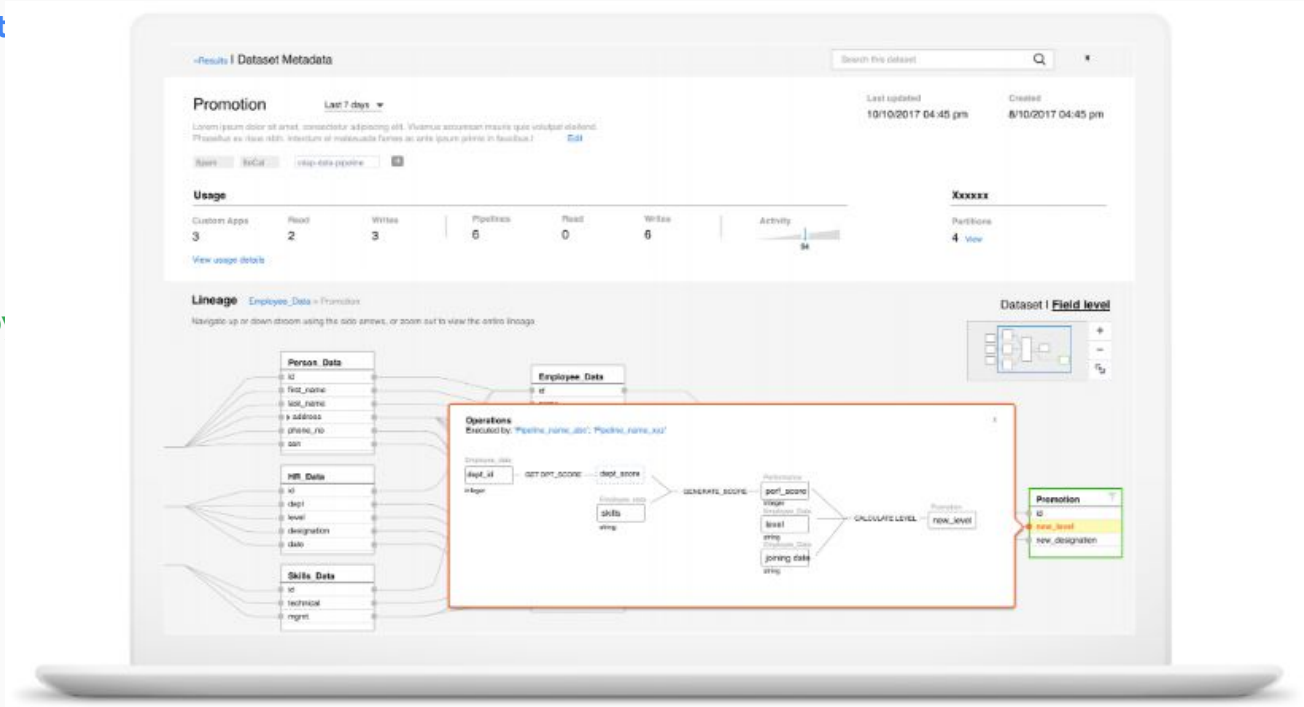
Schema

Search integrated entities b

Keyword

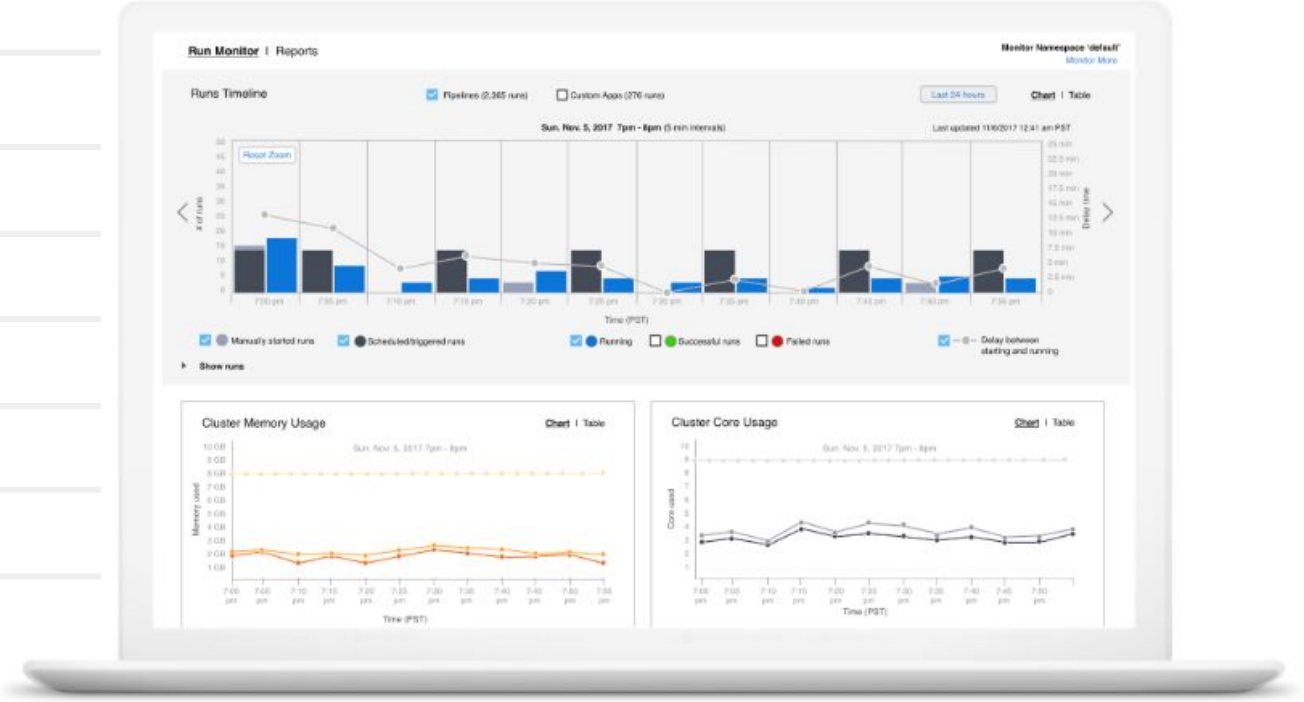
Schema name and type

Dataset level and Field level Lineage



Data Fusion - Extensible

- Pipeline templatization
- Conditional Pipeline Triggers
- Plugin Management
- Plugin templatization
- Plugin UI Widget
- Custom Provisioners
- Custom Compute Profiles
- Hub Integration



Ejercicio Data Fusion

Inicio


Ingest CSV data to BigQuery using Cloud Data Fusion - Batch ingestion
Updated October 7, 2020
In this codelab, we will implement a data ingestion pattern to load CSV formatted healthcare data into BigQuery using Cloud Data Fusion.
Start

Data Catalog

Data Catalog is a fully managed and highly scalable data discovery and metadata management service

Organizations faced with a wealth of data spread across disjoint systems need an **effective solution for data discovery**

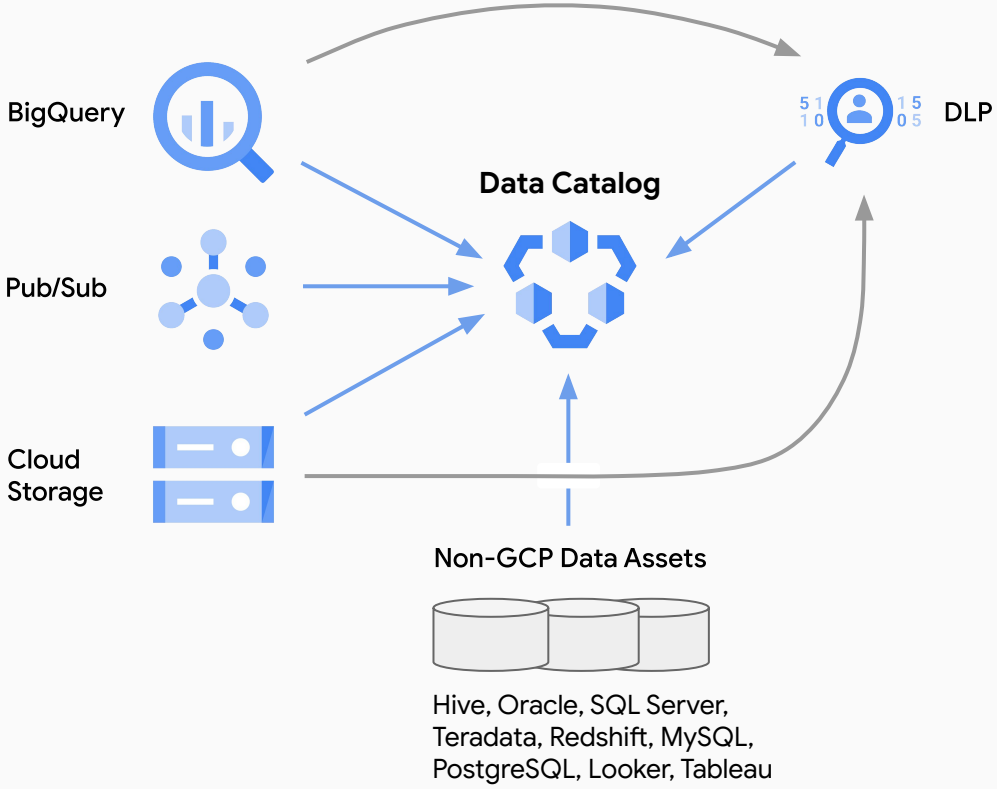
Offers **unified data discovery** of all data assets, spread across multiple projects and systems

Empowers users to **annotate business metadata** in a collaborative manner

Provides the **foundation** for data **governance**

Data Catalog - Features

1. Auto-syncs ‘technical metadata’ from GCP data assets in near real-time
2. Auto-tags PII data through DLP integration
3. Supports non-GCP data assets through open-source connectors



Data Catalog - Architecture

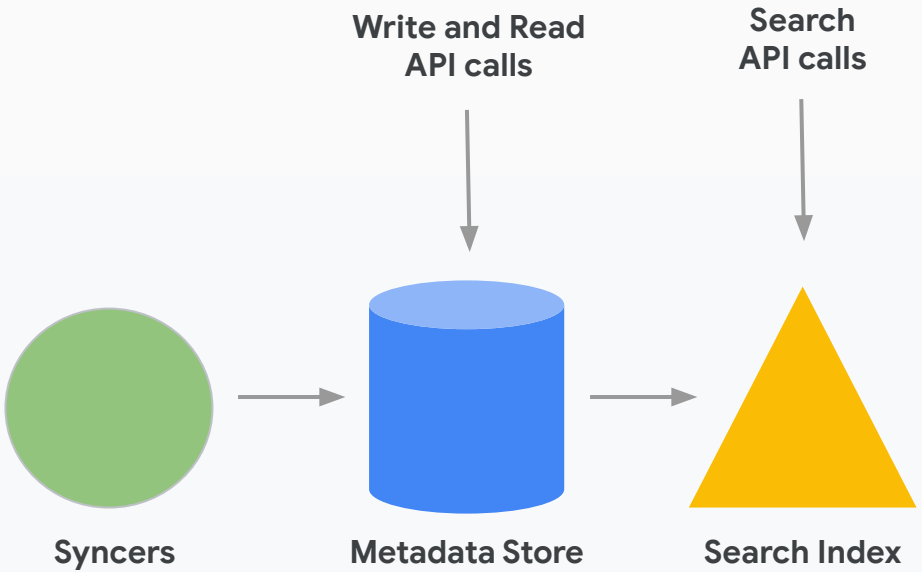
Data Catalog leverages the architecture and innovation of Google's internal metadata management service that is in wide use for many years.

Spanner, the globally distributed, strongly consistent database for storing all metadata entries

Real-time and batch syncers for auto-ingestion of technical metadata

Google search index with built-in ACL checks for data discovery -

- Leverages the same technology that powers **Gmail** and **Google Drive**

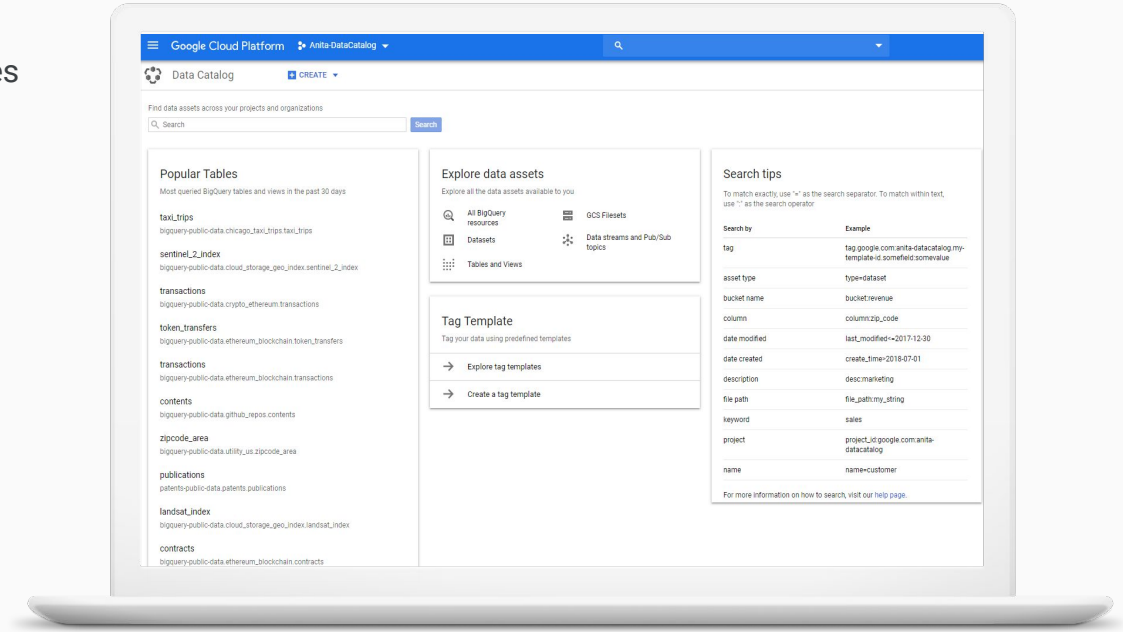


Data Catalog - Data Discovery

Simple keyword search interface enables both, business and technical users

Facet search enables power users

- **projectid**:my_proj_id
- **system**=bigquery
- **type**=table.view
- **column**:keyword
- **description**:keyword
- **tag**:has_pii=true
- **createtime**>2020-01
- **Updatetime**:2020-02-02



Programmatic access through API

Read, Write, and Search API for full metadata access

API powers bulk metadata updates

- Beta API with Python, Java, and Node.js language libraries

API enables Enterprise Applications and Custom Frontends

- GOJEK and others

```
# Create Tag Template
template = datacatalog_v1beta1.types.TagTemplate()
template.display_name = 'A test for v1beta1 using only primitive types'

template.fields['a-boolean'].display_name = 'BOOL field'
template.fields['a-boolean'].type.primitive_type = \
    datacatalog_v1beta1.enums.FieldType.PrimitiveType.BOOL.value

template.fields['a-double'].display_name = 'DOUBLE field'
template.fields['a-double'].type.primitive_type = \
    datacatalog_v1beta1.enums.FieldType.PrimitiveType.DOUBLE.value

template.fields['a-string'].display_name = 'STRING field'
template.fields['a-string'].type.primitive_type = \
    datacatalog_v1beta1.enums.FieldType.PrimitiveType.STRING.value

template.fields['a-timestamp'].display_name = 'TIMESTAMP field'
template.fields['a-timestamp'].type.primitive_type = \
    datacatalog_v1beta1.enums.FieldType.PrimitiveType.TIMESTAMP.value

template.fields['an-enum'].display_name = 'ENUM field'
template.fields['an-enum'].type.enum_type.allowed_values.append('VALUE_A')
template.fields['an-enum'].type.enum_type.allowed_values.append('VALUE_B')

client.create_tag_template(
    parent=client.project_path('YOUR_PROJECT_ID'),
    tag_template_id='YOUR_TEMPLATE_ID',
    tag_template=template)

# Create Tags
# ================================================================
tag = datacatalog_v1beta1.types.Tag()
tag.template = client.tag_template_path('YOUR_PROJECT_ID', 'YOUR_TEMPLATE_ID')
tag.fields['a-boolean'].bool_value = True
tag.fields['a-double'].double_value = 3.1415
tag.fields['a-string'].string_value = 'tag-string'
tag.fields['a-timestamp'].timestamp_value.FromJsonString('2019-03-29T15:45:00-07:00')

bq_dataset_entry_id = 'YOUR_BQ_DATASET_ENTRY_ID'
client.create_tag(parent=client.entry_path('YOUR_PROJECT_ID', 'global', '@bigquery', bq_dataset_entry_id), tag=tag)
```

Technical and Business Metadata

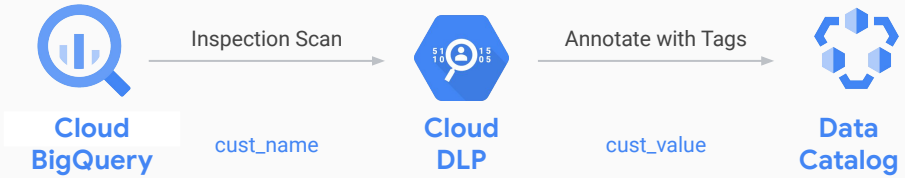
Technical Metadata (auto-ingested from data source)

- Table names, column names
- Table descriptions, column descriptions
- Date created, date modified

Business Metadata (user provided / inferred)

- Table has PII
- Data quality owner
- ‘Delete by’ date
- ‘Retain till’ date
- Business logic used for computing a column
- Data quality score

Auto-tagging PII data in BigQuery with DLP integration

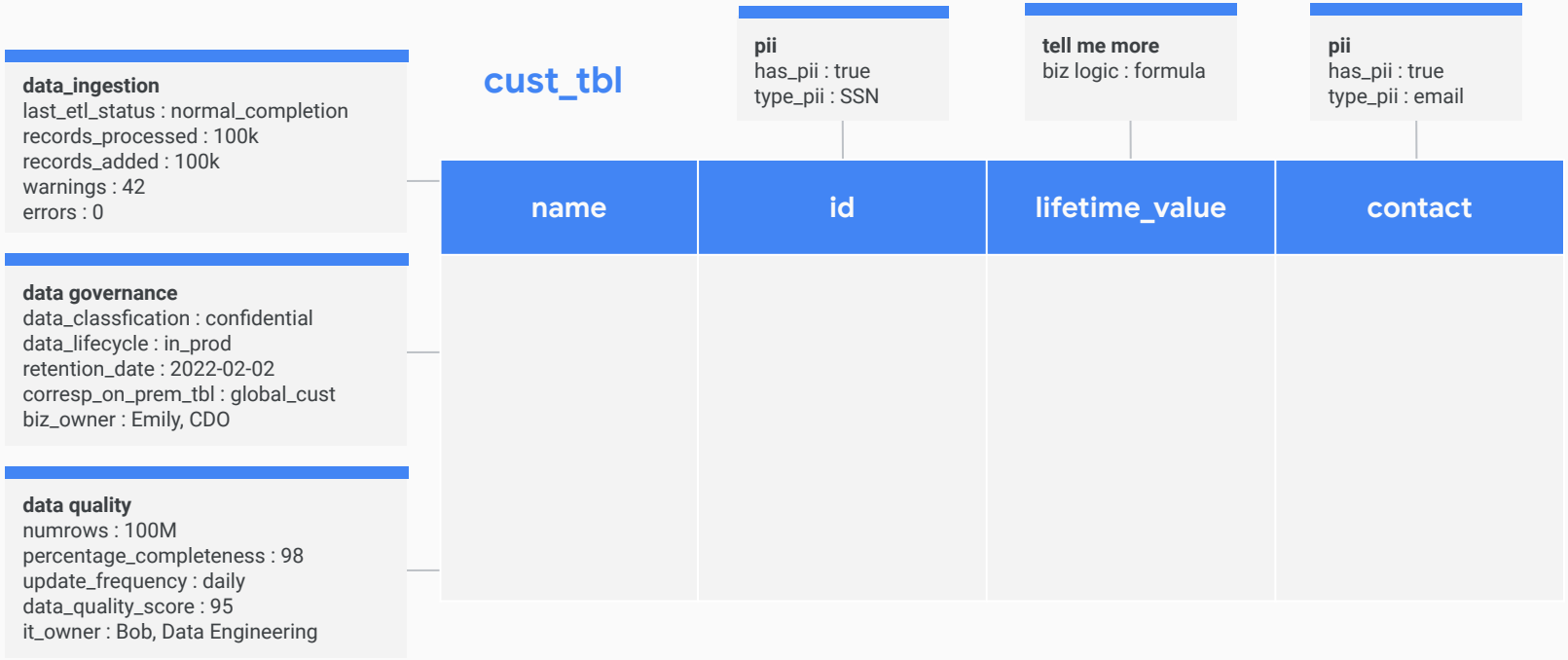


column:credit_card_number: DLP Result Scan Template (travel-bookings-236421.dlp_result_scan_template)

Attribute	Display name	Value
scan_date_time	Date Time of Scan	04/04/2019 04:44
result	Result	CREDIT_CARD_NUMBER
has_pii	Has PII Information	true
num_rows_scan	Number of Rows to Scan in DLP	1000
result_percentual	Result percentual	1
all_result	all result	|CREDIT_CARD_NUMBER(LIKELY : 100.0%);|
num_rows_table	Number of Rows in the table scanned	13000010
result_likelihood	Results likelihood	LIKELY
num_match	Number of Results	1000

Auto-tagging of PII data using DLP

Data Catalog - Structured Tags for Business Metadata



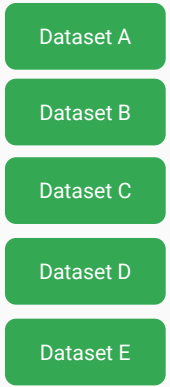
Metadata access governed by IAM

User-1 permissions

- read access to all datasets in BQ

User-1 view in Data Catalog

- Discover all datasets
- Read access to all datasets

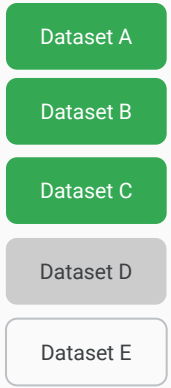


User-2 permissions

- read access to A, B, and C
- metadata-read only access to dataset D
- no read or metadata-read access to dataset E

User-2 view in Data Catalog

- Can discover datasets A, B, C, and D
- Discovers dataset D but needs to request read access to dataset D
- No metadata exfiltration for dataset E



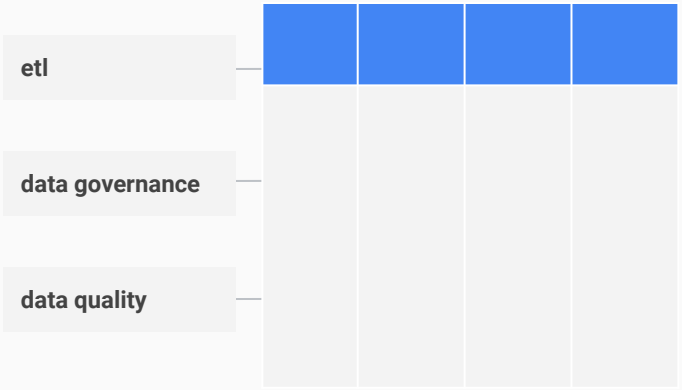
ACL controls on Business Metadata

Data governor permissions

- access to all tags

Data Governor view in Data Catalog

- can discover all tags

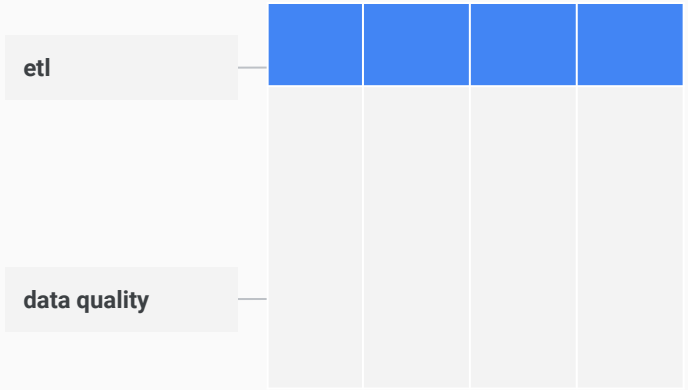


Data Analyst permissions

- no access to ‘data governance’ tags

Data Analyst view in Data Catalog

- can discover all accessible tags



¿ Preguntas ?

Ismael Yuste
linkedin.com/in/ismaelyuste/
@IsmaelYuste