

# Data Processing Architecture

Google Cloud Platform Fundamentals: Big Data and Machine Learning

---

Version #1.1



© 2017 Google Inc. All rights reserved. Google and the Google logo are trademarks of Google Inc. All other company and product names may be trademarks of the respective companies with which they are associated.

## Notes:

60 minutes lecture + 30 minutes lab

# Agenda



## Notes:

### 1. Introduction

Overview of GCP as a whole, but with emphasis on the data-handling aspects of the platform

- GCP, GCP Big Data
- Usage scenarios
- Create an account on GCP

### 2. Foundation of GCP

Compute and Storage with a focus on their value in data ingest, storage, and federated analysis

- Compute Engine
- Cloud Storage
- Start GCE instance
- Upload data to GCS

### 3. Data analytics on the Cloud

Common use cases that Google manages for you and for which there is an easy migration path to the Cloud

- Cloud SQL
- Dataproc
- Import data into and query Cloud SQL
- Machine Learning with Dataproc

In the morning, we will complete Modules 1 and 2 and get halfway through Module 3.

#### 4a. Scaling data analysis

Change how you compute, not just where you compute with GCP

- Datalab
- Datastore, Big Table
- BigQuery

#### 5. TensorFlow

Change how you compute, not just where you compute with GCP

- TensorFlow
- Datalab instance
- BigQuery
- Demand forecasting with ML

#### 6. Data processing architectures

Scaleable, reliable data processing on GCP

- Pub/Sub
- Dataflow

#### 7. Summary

Course summary

- Resources

Please feel free to use the appendixes for self-study.

In the morning, we will get halfway through Module 3.

Please feel free to use the appendixes for self-study.

Please take 5 minutes to give us feedback

[g.co/CloudTrainEval](https://g.co/CloudTrainEval)

This URL is case-insensitive

The class code is always a tag in the QL class. Instructor can always find the class code (if they don't have it already ) when they are in the class in QL, click Edit Class and the code is in the tags. Not ideal - but we'll figure out how to make it more prominent

# Agenda

## Message-oriented architectures

---

Serverless data pipelines

---

GCP Reference Architecture

---

## Notes:

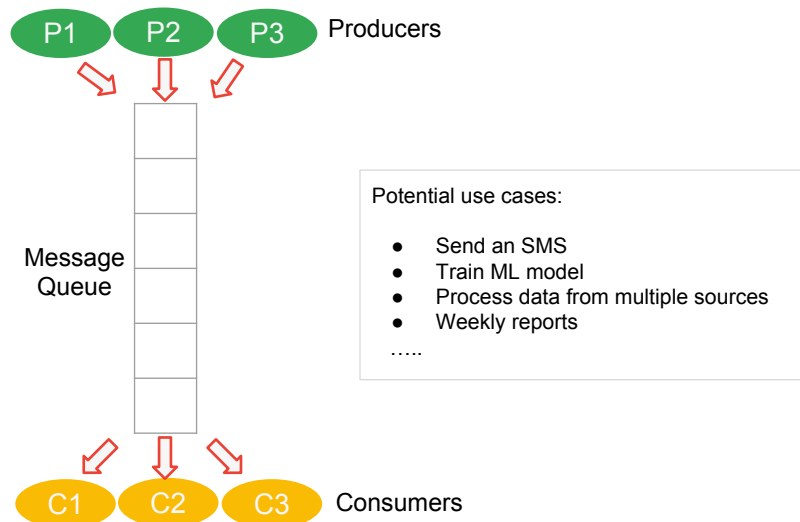
This chapter is going to be transformational use cases taking advantage of GCP. It's about database technologies, picking up from the previous chapter. Then we'll look at notebooks, large-scale querying of structured data and finally machine learning.

Now, why do you use a database? [ask, and most people will answer that it is to store data]

No, really, the reason you use a database rather than simply store the data on a tape drive is that you want fast random access to it. It's about fast queries.

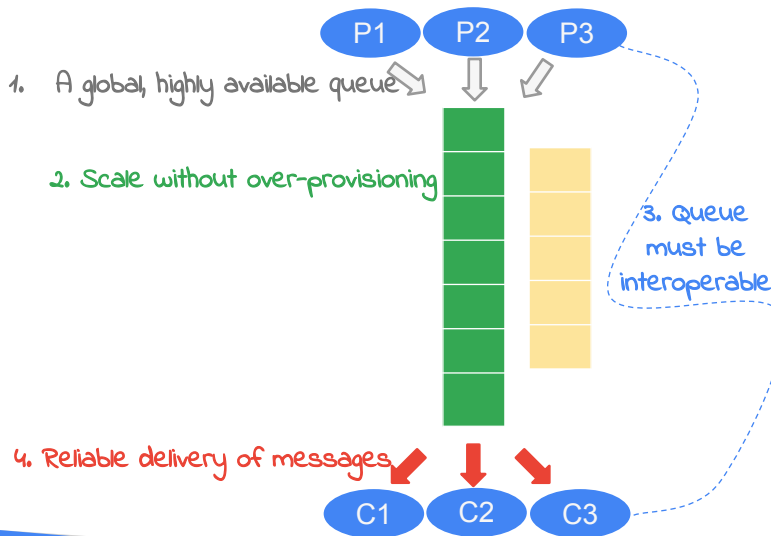
Yet, the solution of using your typical relational database has certain drawbacks as we looked at in the review to module 3 – for example, if the data are not structured, it leads to a mismatch.

Asynchronous processing is useful for long-lived tasks or to have loose coupling between two systems



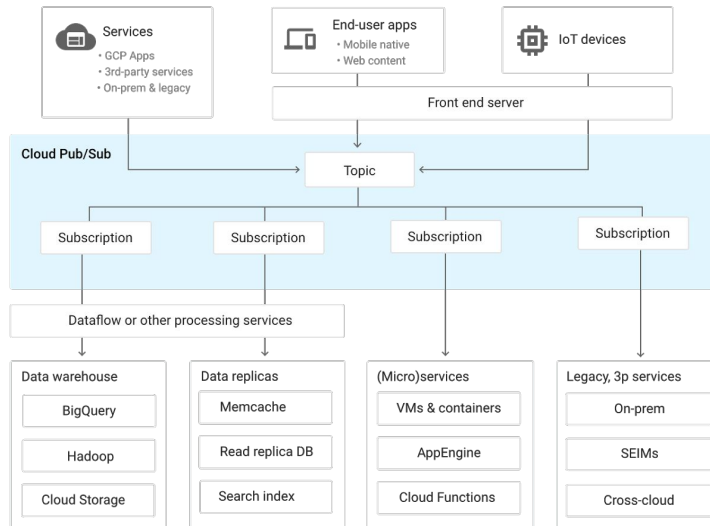
any processing that would take more than a second or two will ultimately be far too slow for synchronous execution. In addition, there is often processing that needs to be scheduled in the future and/or processing that needs to affect an external service. In these cases when we have a task that needs to execute but that is not a candidate for synchronous processing, the best course of action is to **move the execution outside the request/response cycle**. Specifically, we can have the synchronous web app simply notify another separate program that certain processing needs to be done at a later time.

For robust asynchronous processing, you need:



1. So that producers can always access the queue and be short-lived processes.
2. As the # of messages increases, the queue needs to manage storage and compute so as to not fall over; but don't overprovision for traffic surges
3. P3 might be Java on Linux and C3 might be PHP on Windows. Still has to work.
4. Reliable delivery of messages amid producer/consumer/hardware failures

# Pub/Sub provides a no-ops, serverless global message queue





# Agenda

Message-oriented architectures

---

Serverless data pipelines

---

GCP Reference Architecture

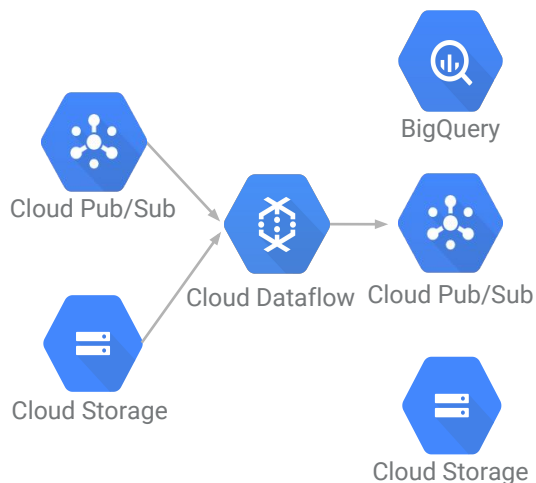
---

## Dataflow offers NoOps data pipelines in Java and Python



Dataflow is a superset of MapReduce. Can do MapReduce + a whole lot more. Note that in this workflow, we are doing a Map (speed-by-sensor), followed by a GroupBy (the shuffle stage in MapReduce), followed by a reduce (the average)

## Same code does real-time and batch



```

options = PipelineOptions(pipeline_args)
options.view_as(StandardOptions).streaming = True
p = beam.Pipeline(options=options)

lines = p | beam.io.ReadStringsFromPubSub(input_topic)
traffic = (lines
    | beam.Map(parse_data).with_output_types(unicode)
    | beam.Map(get_speedbysensor) # (sensor, speed)
    | beam.WindowInto(window.FixedWindows(15, 0))
    | beam.GroupByKey() # (sensor, [speed])
    | beam.Map(avg_speed) # (sensor, avg_speed)
    | beam.Map(lambda tup: '%s: %d' % tup))
traffic | beam.io.WriteStringToPubSub(output_topic)

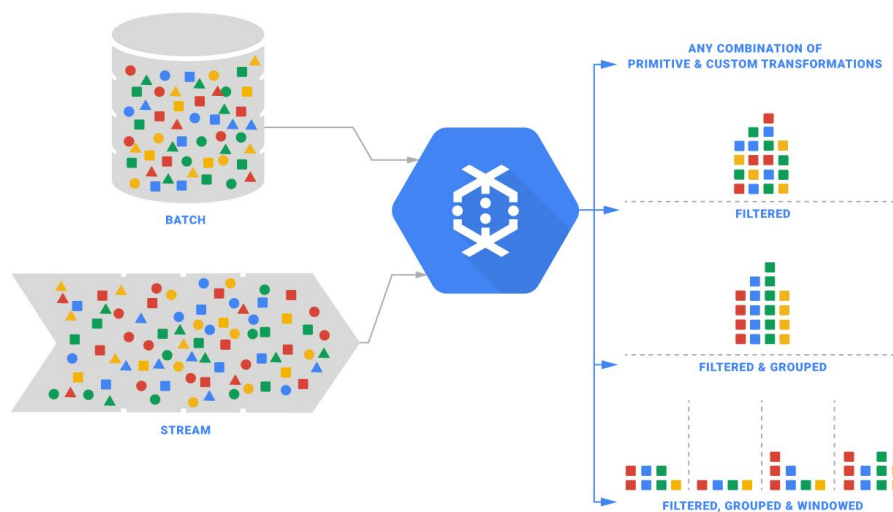
p.run()
  
```

### Notes:

You can get input from any of several sources, and you can write output to any of several sinks. The pipeline code remains the same.

You can put this code inside a servlet, deploy it to App Engine, and schedule a cron task queue in App Engine to execute the pipeline periodically.

Dataflow does ingest, transform, and load; consider using it instead of Spark



### Notes:

You can replace all the various data handling tools with just Dataflow. Many Hadoop workloads can be done easily and more maintainably with Dataflow. Plus, Dataflow is NoOps.

# Agenda

Message-oriented architectures

---

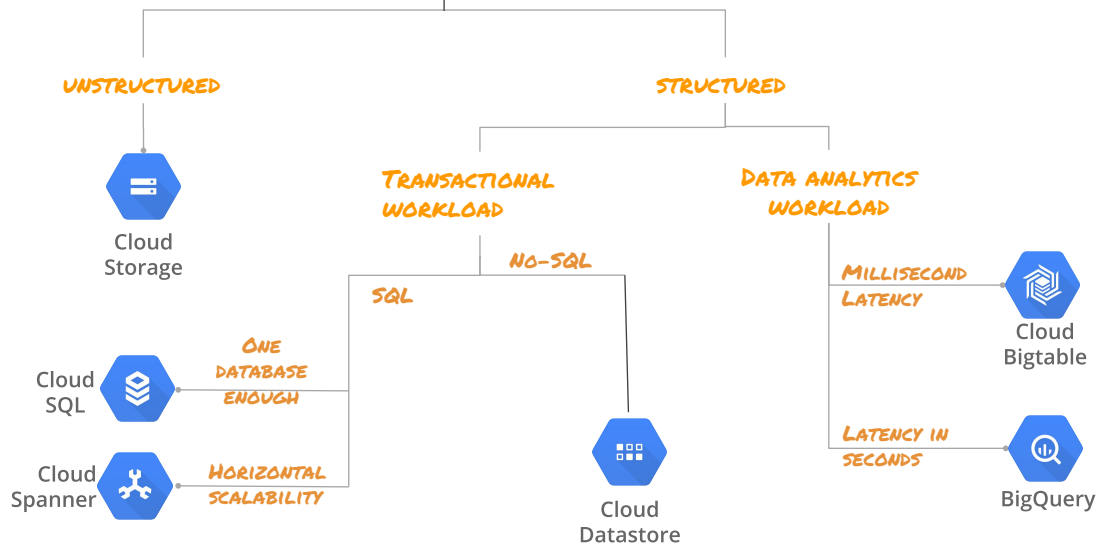
Serverless data pipelines

---

[GCP Reference Architecture](#)

---

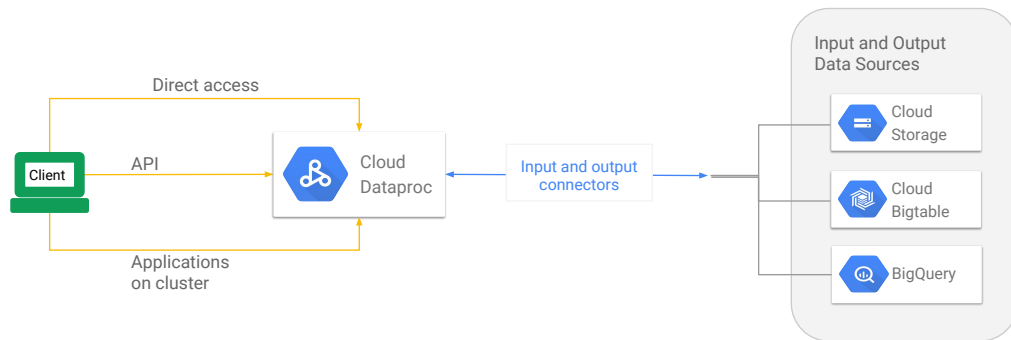
# Choosing where to store data on GCP



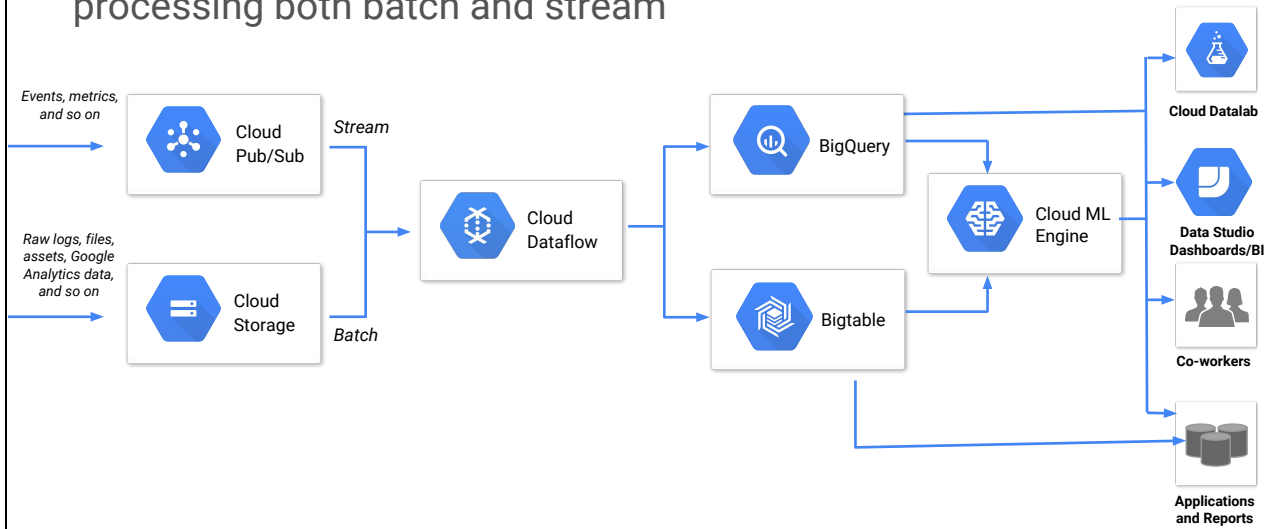
© 2017 Google Inc. All rights reserved. Google and the Google logo are trademarks of Google Inc. All other marks and names may be trademarks of the respective companies with which they are associated.

Google Cloud

## Run Spark/Hadoop jobs on Cloud Dataproc



## On GCP, you can have the same data processing pipeline for processing both batch and stream



©2017 Google Inc. All rights reserved. Google and the Google logo are trademarks of Google Inc. All other names and product names may be trademarks of their respective companies. All rights reserved.

Google Cloud



# Module Review

## Module review



Match the use case on the left with the product on the right

- A. Decoupling producers and consumers of data in large organizations and complex systems
- B. Scalable, fault-tolerant multi-step processing of data

- 1. Cloud Dataflow
- 2. Cloud Pub/Sub

## Module review answers

Match the use case on the left with the product on the right

- |  |   |                   |
|--|---|-------------------|
| A. Decoupling producers and consumers of data in large organizations and complex systems |  | 1. Cloud Dataflow |
| B. Scalable, fault-tolerant multi-step processing of data                                |  | 2. Cloud Pub/Sub  |

### Notes:

- A. Cloud Pub/Sub
- B. Cloud Dataflow

## Resources (1 of 2)

---

Cloud Pub/Sub

<https://cloud.google.com/pubsub/>

---

Cloud Dataflow

<https://cloud.google.com/dataflow/>

---

Processing media using  
Cloud Pub/Sub and  
Compute Engine

<https://cloud.google.com/solutions/media-processing-pub-sub-compute-engine>

---

## Resources (2 of 2)

---

Reverse Geocoding of  
Geolocation Telemetry in  
the Cloud Using the Maps  
API

<https://cloud.google.com/solutions/reverse-geocoding-geolocation-telemetry-cloud-maps-api>

---

Using Cloud Pub/Sub for  
Long-running Tasks

<https://cloud.google.com/solutions/using-cloud-pub-sub-long-running-tasks>

---

