Fundamentals of GCP

Google Cloud Platform Fundamentals: Big Data and Machine Learning

Version #1.1

Notes:

30 minutes + 15 minutes lab

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

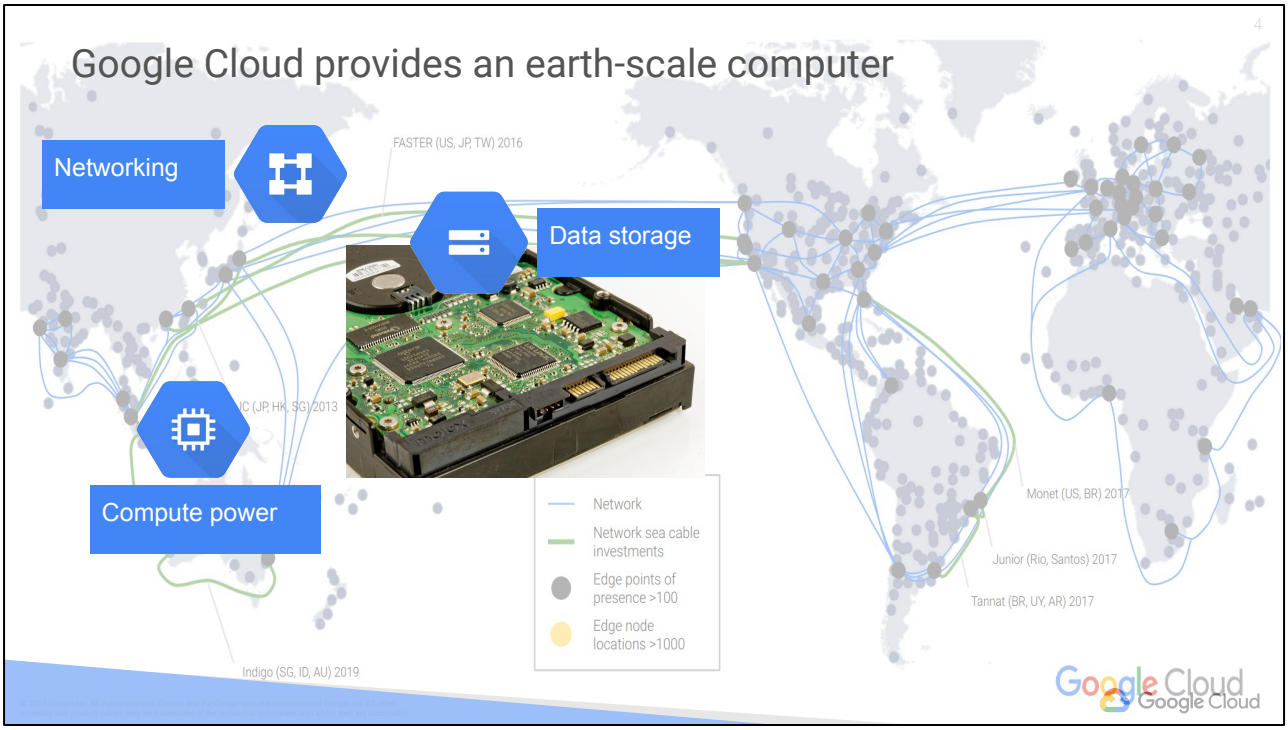
Please feel free to use the appendixes for self-study.
In the morning, we will get halfway through Module 3.

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

Agenda

CPUs on demand + Lab

A global filesystem + Lab

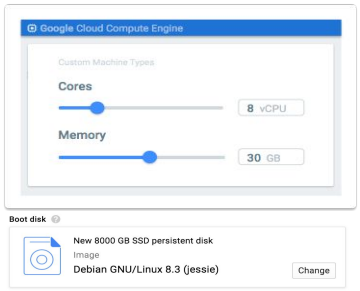


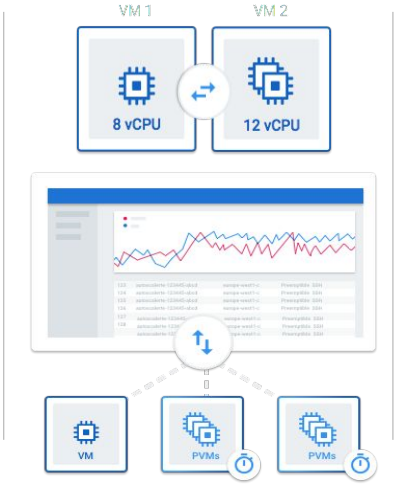
Notes:
https://pixabay.com/en/hard-drive-hdd-technology-digital-870699/ (cc0)

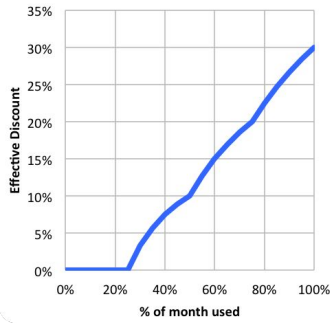
Any computer requires cpus, disks, and networking. Google Cloud Platform's computer is earth-scale. The cpu here is Compute Engine and the disk here is Cloud Storage. Google also provides the connectivity that you need to move data around and serve your users.

Let's talk about Compute Engine first—The design criteria are that Google wants things to be no-ops (no Gmail developer ever spins up a VM), scalable, and built on the best architectures and open standards.

Custom/changeable machine types, preemptible machines, and automatic discounts lead to simplicity and agility







Google Cloud

Notes:
https://cloud.google.com/custom-machine-types/
https://cloud.google.com/compute/pricing

Things you can customize: cores, memory, disk size, operating system. Load balancing, networking, and so on all come “baked in.” Then enjoy pricing simplicity and agility (The ability to change your initial choices leads into the next slide).

This is a good point to go to the pricing page, scroll down to the https://cloud.google.com/products/calculator/ and demo it.

https://cloud.google.com/compute/docs/instances/changing-machine-type-of-stopped-instance
https://cloud.google.com/preemptible-vms/

When running fault-tolerant software such as Hadoop, it is helpful to be able to supplement with PVMs.
You can tie this into the flexibility and customization. On Google Cloud Platform, preemptible instances cost 80% less. Always. There is no bidding.

Lab: Create a Compute Engine instance

Google Cloud

Lab 1, Part 1: Create a Compute Engine Instance

In this lab you will :

1. Create a Compute Engine instance
2. SSH into the instance
3. Install the software package git (for source code version control)

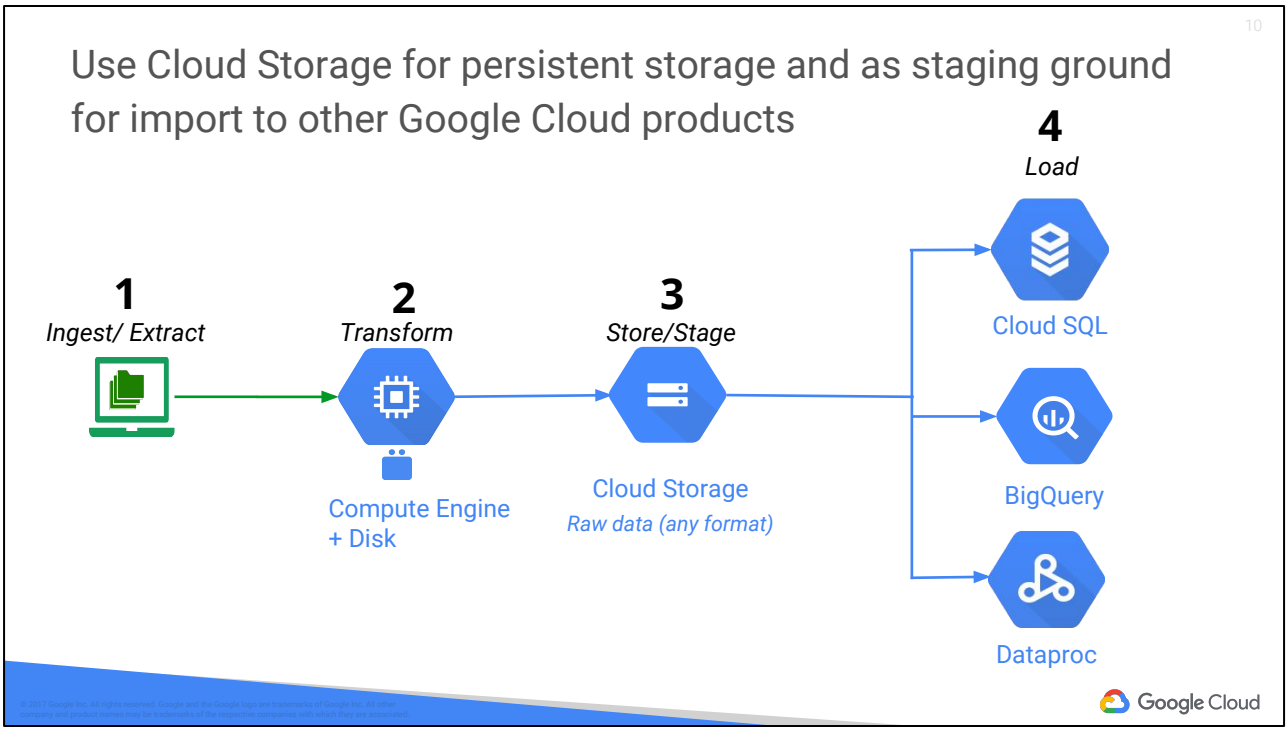
Notes:

If you miss changing the access and security options, you will not be able to copy the Cloud Storage file in the next lab. If you get permissions errors in a later lab, this is the reason.

Agenda

CPUs on demand + Lab

A global filesystem + Lab



Notes:

Disks attached to Compute Engine instances are fast, but ephemeral. Cloud Storage is durable but it is replicated and accessible from any machine. Use Cloud Storage as staging ground for import into other Google Cloud Platform products.

Let's walk through this flow-chart of how data gets processed. It (1) gets ingested, (2) cleaned up, transformed, and so on, (3) saved into cloud storage and (4) imported back into a compute node or into BigQuery, Dataproc, and so on for analysis. Ingest/Compute/Store would be for real-time data processing. Extract/Transform/Load would be for batch processing. It's the same idea, just different terminology.

What kinds of storage would you use in these stages?

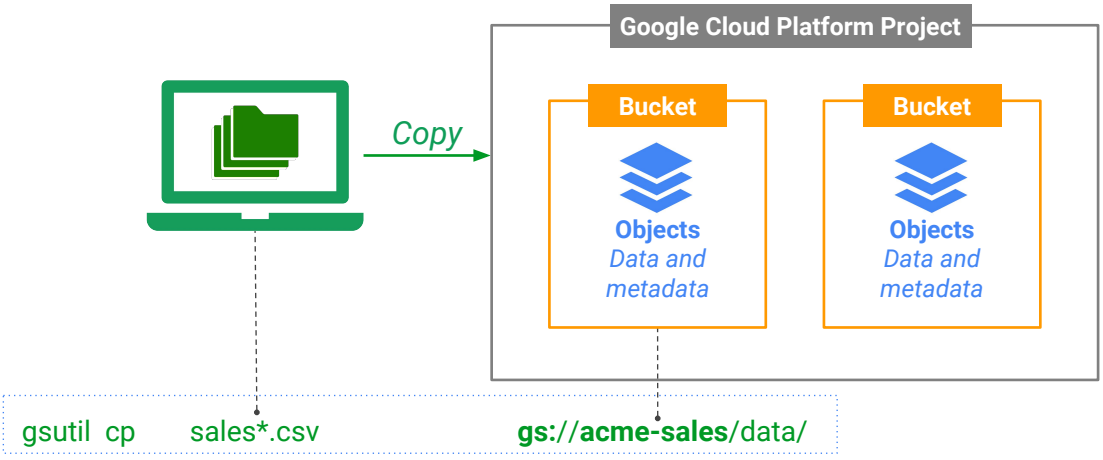
When you launch a Compute Engine instance, you can attach one or more disks. These are dedicated and so provide very fast read/write. However, when the instance is shut down, typically the disk also goes away (You can keep the disks and attach them to other instances, but disks are relatively expensive). The right way to save raw data so that it can be accessed at some future point in time or from other instances is to put the data in Cloud Storage -- raw object

storage. Then, when you are ready to process the data, you can copy it back to a disk for high-speed access or import the data into CloudSQL or BigQuery or DataProc. For the most part, Cloud Storage is an HDFS-replacement, so if you are using Dataproc, just run it off Cloud Storage directly -- there is no need to load it into HDFS. With the drop in prices for BigQuery storage, you might also think of BigQuery as a durable data warehouse.

Details:

- https://cloud.google.com/compute/docs/disks: Use disks for speed, but they are typically not durable beyond the life of the VM(s), although you can ask the disks to be retained after the instance is shut down. Local disks can be attached to only one instance. Cloud Storage provides durability and global access.
- You can also load from Cloud Storage buckets into BigQuery, CloudSQL, and so on. (You do this in this course).
- Use SSDs for fast writes.
- Use Persistent HDDs for fast sequential reads (as often happens with Hadoop) -- you could use SSDs, but HDDs are cheaper. The persistent storage is redundant (3x).

Create a bucket and copy the data over using the Cloud SDK; blobs are referenced through a gs://.../ URL



Google Cloud

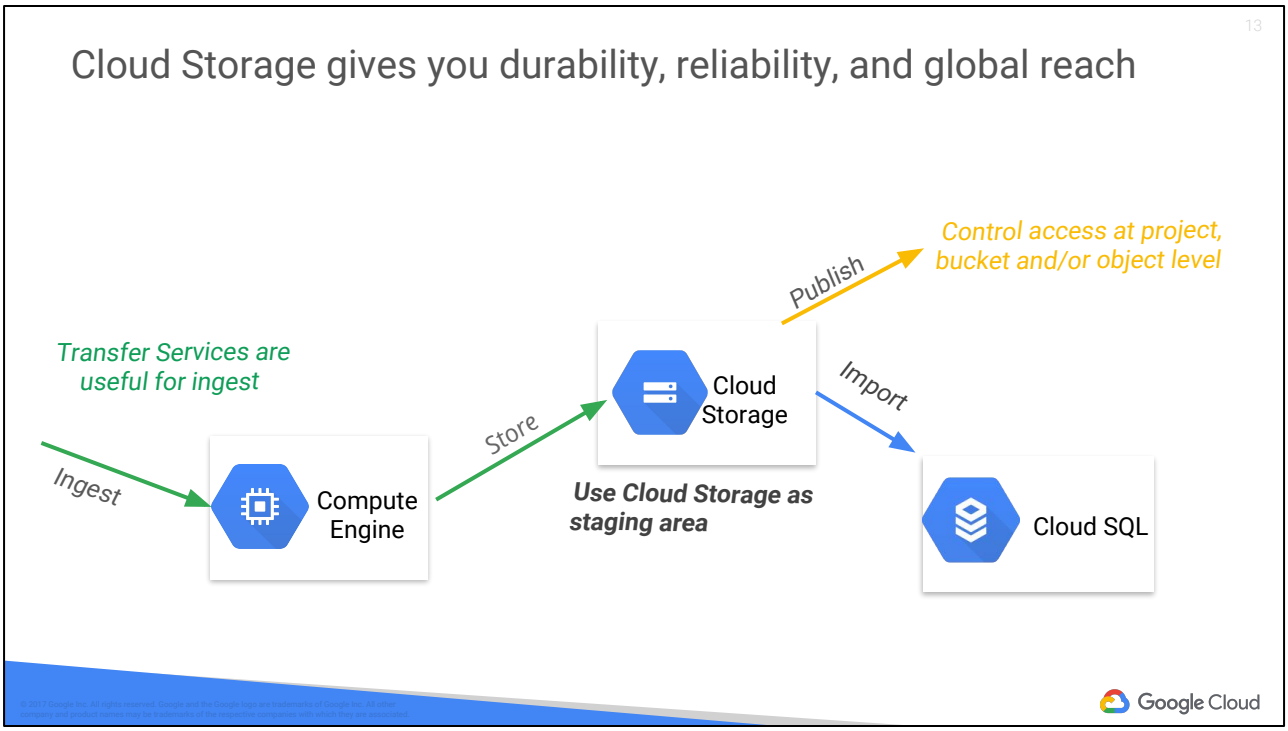
Notes:
https://cloud.google.com/storage/docs/overview

Follow-on from the previous slide: The purpose of Cloud Storage is to give you a durable, global, file system. How is it organized (talk about projects/buckets/objects)?

How do you work with it? You can use gsutil.

If you spin up a Compute Engine instance, gsutil is already available. On your laptop, you can download the Google Cloud SDK to get gstuil. gsutil uses a familiar UNIX command syntax. mb, rb are make-bucket and remove-bucket, for example. Instead of command-line, you can also use GCP Console, programming language or REST API

The way to get a globally unique bucket name is to use a reverse domain name (in which case Google Cloud Platform will ask you to prove you own the domain name in question) or simply use your project-id.



Notes:

Besides cp, you can configure a Transfer Service to grab data from a URL or S3 bucket and save it to Cloud Storage. You can also use a transfer service to automatically copy data from Standard storage to Nearline.

Common question: Can you treat Cloud Storage like a disk? Yes, for the most part, using https://cloud.google.com/storage/docs/gcs-fuse. Some things like hard links are not supported. A better metaphor for Cloud Storage is that a bucket is a hashmap of names to raw-bytes.

Transfer Services (one-time or recurring) useful for ingest

- Can be setup from Google Cloud console
- Google Cloud provides object change notification

Use Cloud Storage as staging area

- For import into analysis tools & databases
- For staging to disk for fast access

Can control access at project, bucket and/or object level

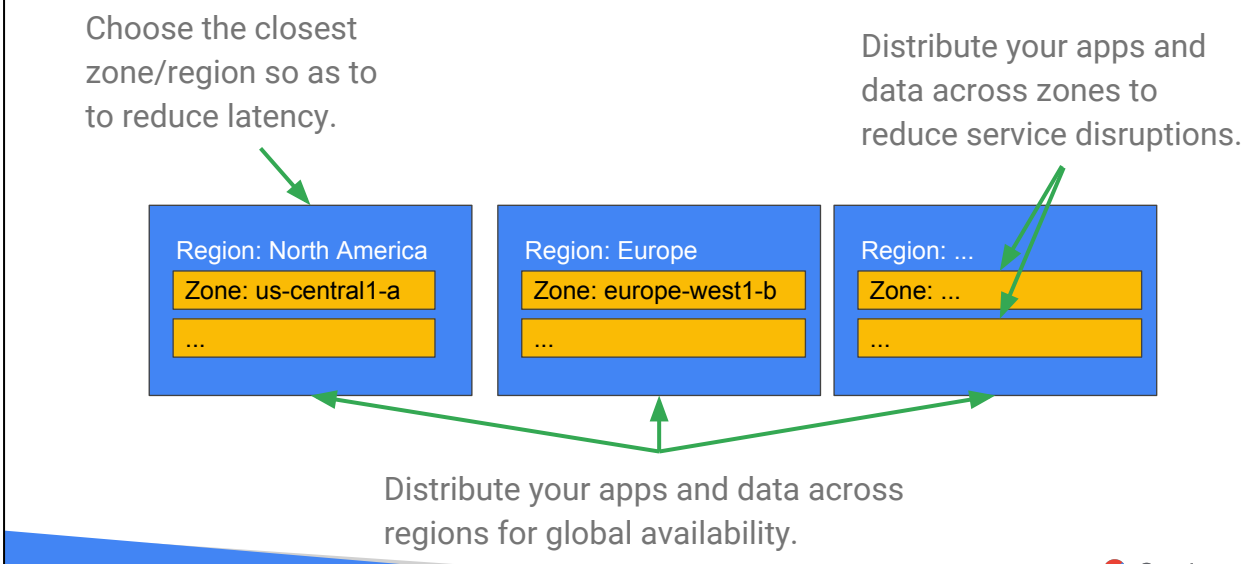
- Useful for ingest, transform and publish workflows
- Including read access to anyone with HTTP URL

Google Cloud gives you durability, reliability and global reach

- Versioning, redundancy, edge-caching

Use redundancy to lead into the next slide. How do you get redundant storage? What kinds of things do you want in a redundant storage?

Control latency and availability with zones and regions



Google Cloud

Notes:
https://cloud.google.com/about/datacenters/

Lead into this slide from the previous slide using the word “redundancy”, talking about edge-caching and failover.

Just because Cloud Storage is a global file system doesn’t mean that you can forget about latency considerations. You are better off storing the data close to your compute nodes. However, you should also distribute your apps and data across multiple zones to protect yourself against service disruptions. Leverage zones in different regions for additional redundancy. Distribute your apps and data across regions for global availability.

A zone is an isolated location within a region. It’s named <region-name>-<zone-letter>.

Lab: Interact with Cloud Storage

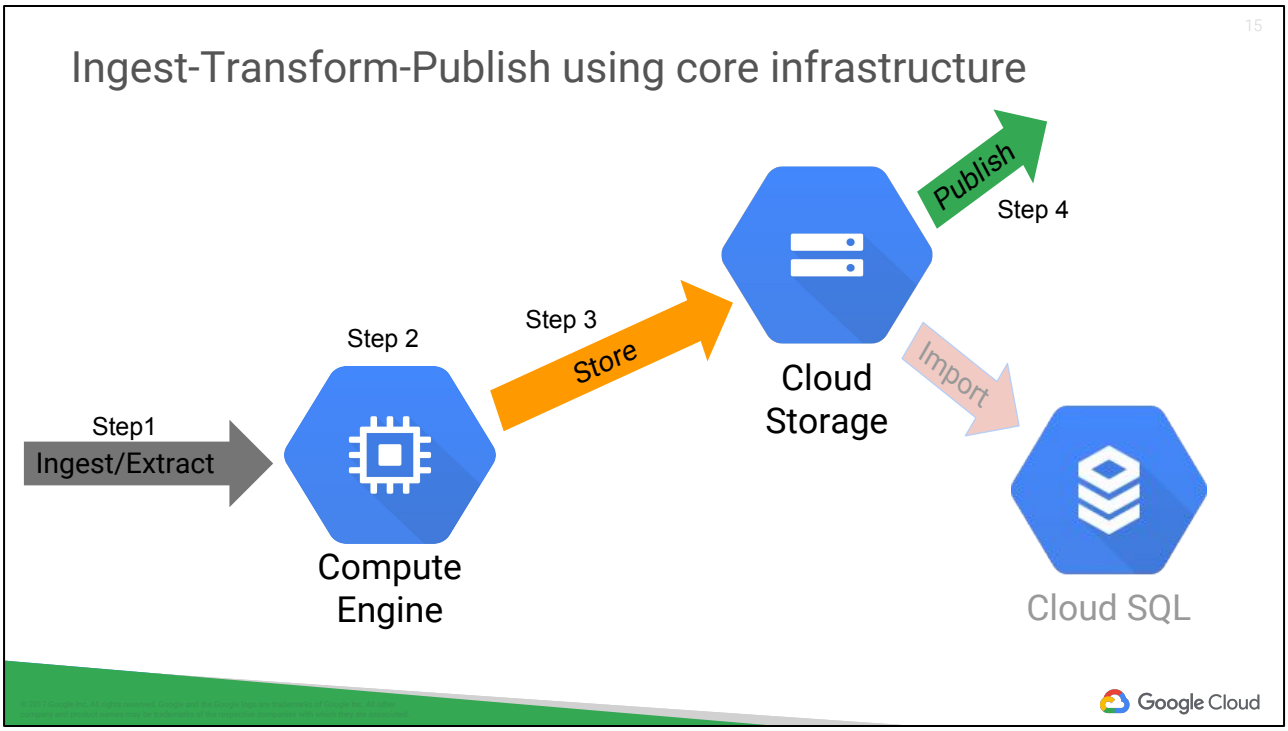
Google Cloud

Lab 1, Part 2: Interact with Cloud Storage

In this lab, you carry out the steps of an ingest-transform-and-publish data pipeline manually:

1. Ingest data into a Compute Engine instance
2. Transform data on the Compute Engine instance
3. Store the transformed data on Cloud Storage
4. Publish Cloud Storage data to the web

Google Cloud

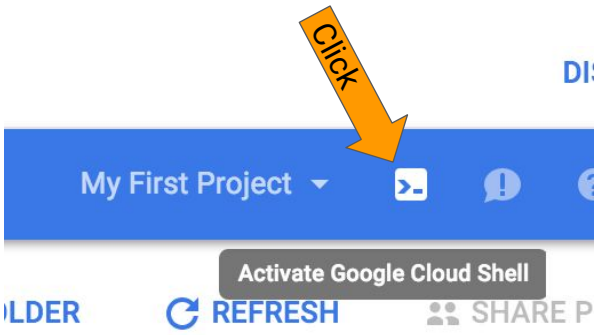


Notes:

Essentially, you are doing manually what will be done automatically within Dataflow, and so on. Point out that if you automate the creation of the VM and this job in the startup-script (you can show them this on the console), you essentially have an automated process to ingest, transform, and publish to the web. Later in this course, you learn how to import data from Cloud Storage into Cloud SQL, BigQuery, and so on.

Note: gsutil cp fails with AccessDeniedException: 403 Insufficient Permission if you skipped the step in Lab 1 where you are asked to allow read/write to Cloud Storage from your instance. If this happens, you have to create another instance with the right permissions (You can't just edit the permissions after-the-fact. Permissions affect the allocated hardware).

Cloud Shell gives you an easy command-line



Cloud Shell comes pre-installed with the tools, libraries, and so on you need to interact with Google Cloud Platform

Notes:

Ask them to do it with you.

CloudShell is container-based and is very convenient. You don't need to spin up Compute Engine instances to do quick one-off operations.

One cool thing is that it also has the permissions you need. So, you can make and remove buckets for example. You don't want to use it for what you did in the lab -- You shouldn't be using CloudShell for actual jobs because it may not have enough memory/storage/power, and the machine is liable to get shut down when you walk away to get coffee (it is a temporary VM).

Module Review

Google Cloud

Module review (1 of 2)

Compute nodes on GCP are:
(select the correct option)

- ❑ Allocated on demand, and you pay for the time that they are up.
- ❑ Expensive to create and teardown
- ❑ Pre-installed with all the software packages you might ever need.
- ❑ One of ~50 choices in terms of CPU and memory

Module review answers (1 of 2)

Compute nodes on GCP are:
(select the correct option)

- ✓ Allocated on demand, and you pay for the time that they are up.
- ❑ Expensive to create and teardown
- ❑ Pre-installed with all the software packages you might ever need.
- ❑ One of ~50 choices in terms of CPU and memory

Notes:

1: True.
2: False. You pay for a minimum of 10 minutes.
3: False. You used apt-get install in the lab.
4: False (customizable).

Module review (2 of 2)

Google Cloud Storage is a good option for storing data that:
(select all of the correct options)

- ❑ Is ingested in real-time from sensors and other devices
- ❑ Will be frequently read/written from a compute node
- ❑ May be required to be read at some later time
- ❑ May be imported into a cluster for analysis

Module review (2 of 2)

Google Cloud Storage is a good option for storing data that:
(select all of the correct options)

- ❏ Is ingested in real-time from sensors and other devices
- ❏ Will be frequently read/written from a compute node
- ✓ May be required to be read at some later time
- ✓ May be imported into a cluster for analysis

1: False, GCS is not low-latency. Use Pub/Sub
2. False. Use persistent disks.
3. True. for persistent storage
4. True. For staging

Resources

Compute Engine	https://cloud.google.com/compute/
Storage	https://cloud.google.com/storage/
Pricing	https://cloud.google.com/pricing/
Cloud Launcher	https://cloud.google.com/launcher/
Pricing Philosophy	https://cloud.google.com/pricing/philosophy/

Google Cloud

Notes:

This is a good opportunity to demonstrate the Pricing page and Cloud Launcher for pre-built instances.


cloud.google.com
Image by Connie Zhou