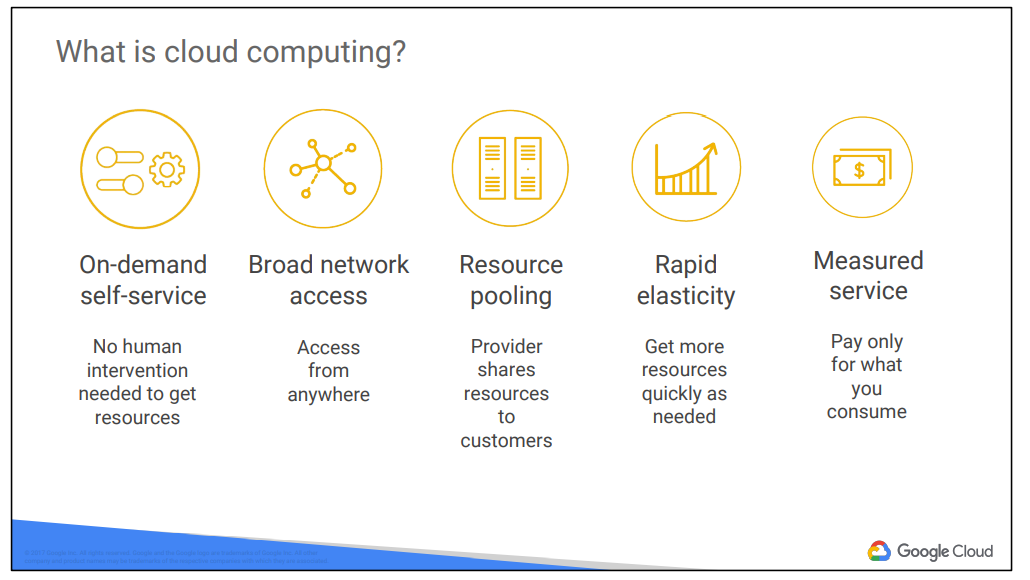
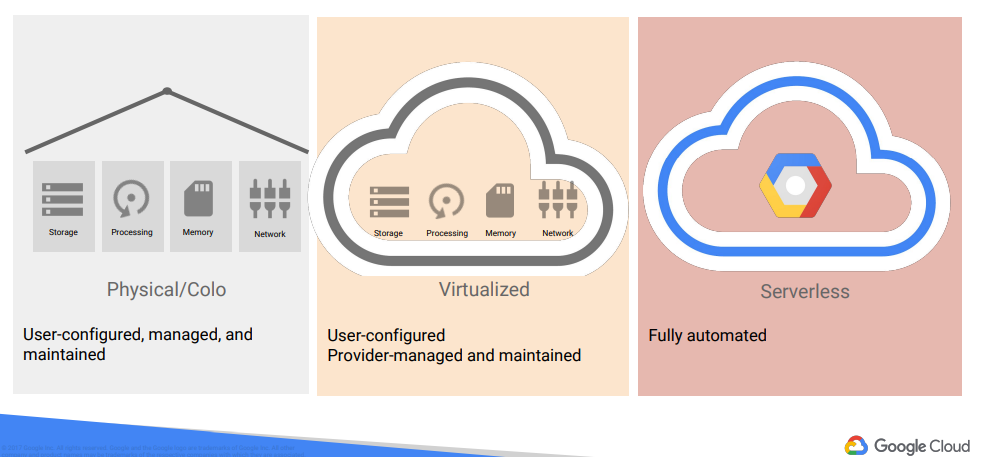
**Core Infrastructure**

**1.**



**How and Why Cloud Computing**

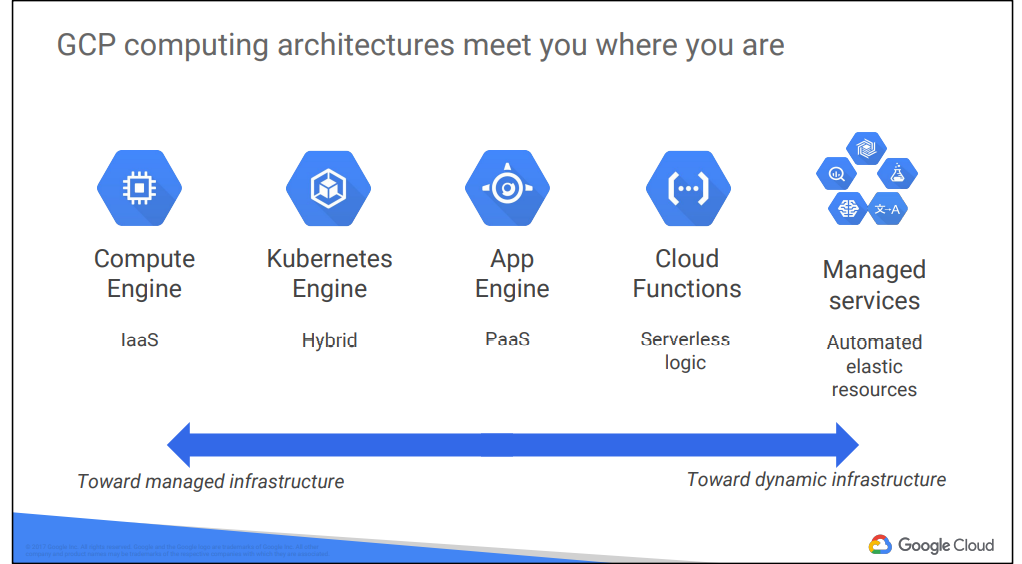


**First phase :** People were more interested in renting physical space instead of investing in data center real estate.

**Second Phase :** Virtualisation was the second phase. The components of virtualized data centers match the physical building blocks of hosted computing—servers, CPUs, disks, load balancers, and so on—but now they are virtual devices. Virtualization does provide a number of benefits: your development teams can move faster, and you can turn capital expenses into operating expenses. With virtualization you still maintain the infrastructure; it is still a user-controlled/user-configured environment.

**Third Phase :** Google switched to a container-based architecture—a fully automated, elastic third-wave cloud that consists of a combination of automated services and scalable data. Services automatically provision and configure the infrastructure used to run applications.

**Google Offerings :**



**Regions :** Regions are independent geographic areas that consist of zones. Locations within regions tend to have round-trip network latencies of under 5 milliseconds on the 95th percentile.

**Zone :** A zone is a deployment area for Google Cloud Platform resources within a region.

**Zonal resources**

Zonal resources operate within a single zone. If a zone becomes unavailable, all of the zonal resources in that zone are unavailable until service is restored.

* Google Compute Engine VM instance resides within a specific zone.

**Regional resources**

Regional resources are deployed with redundancy within a region. This gives them higher availability relative to zonal resources.

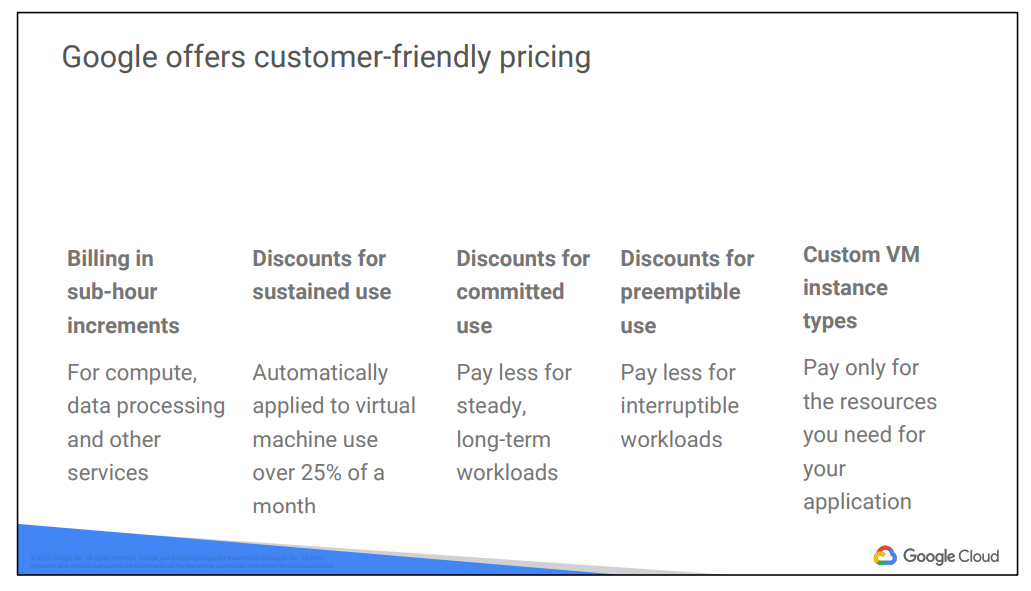
**Multi-regional resources**

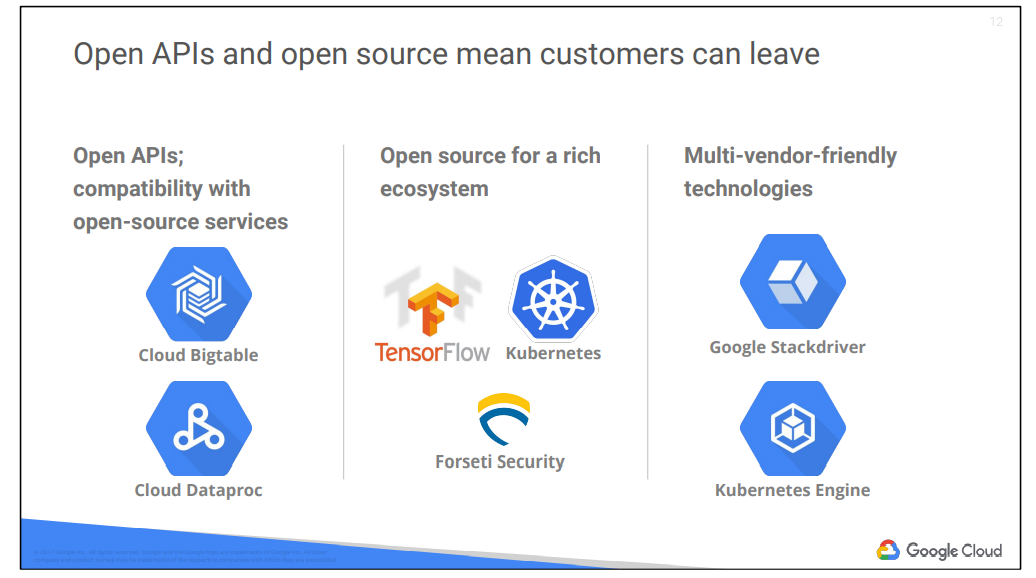
A few Google Cloud Platform services are managed by Google to be redundant and distributed within and across regions. These services optimize availability, performance, and resource efficiency.

* Google App Engine and its features
* Google Cloud Datastore
* Google Cloud Storage
* Google BigQuery



**Pricing :**





Google gives customers the ability to run their applications elsewhere if Google becomes no longer the best provider for their needs.

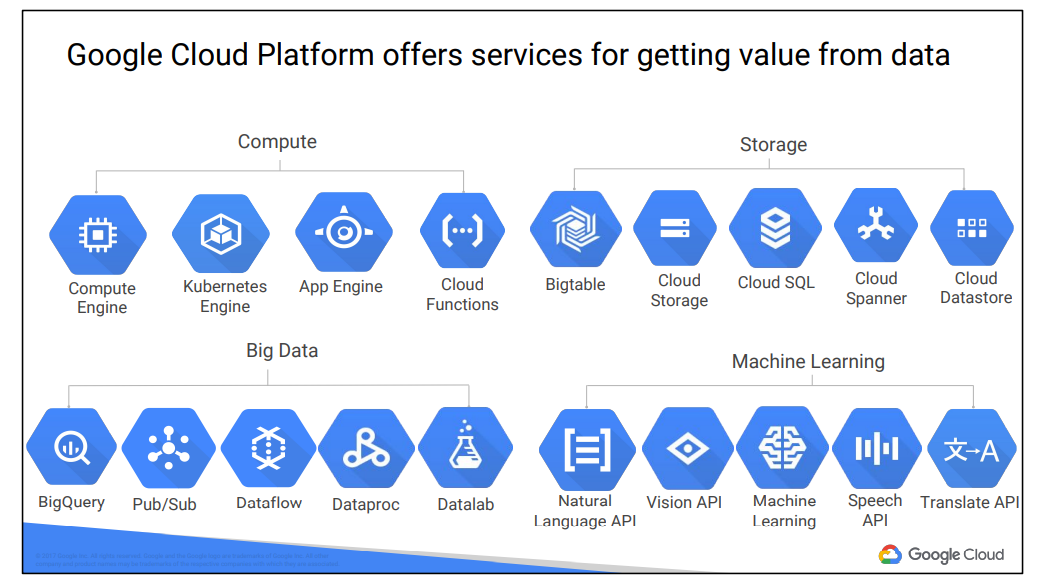
This includes:

● Using Open APIs.

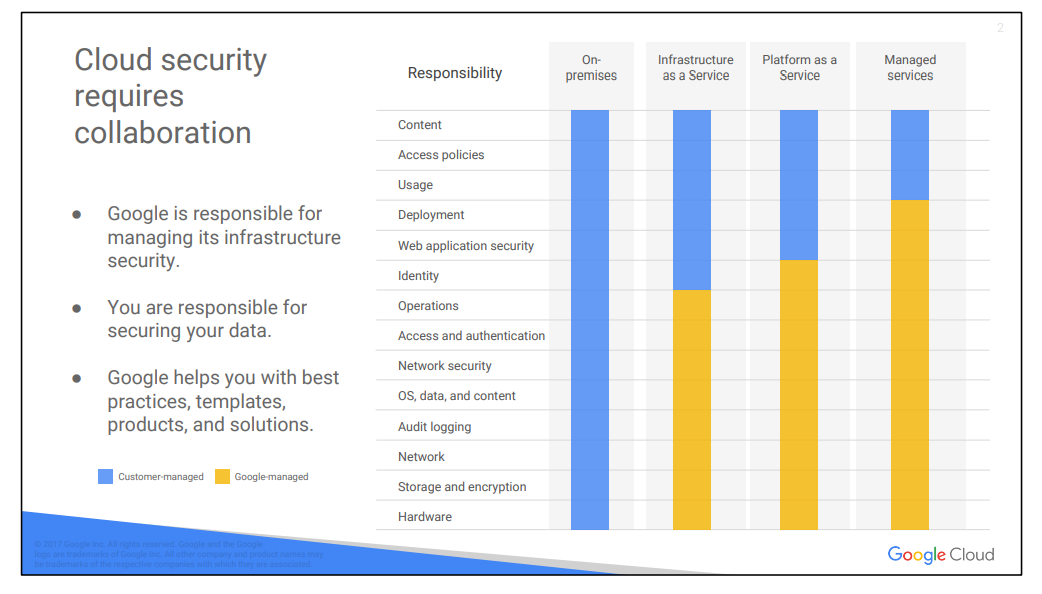
Google services are compatible with open-source products. For example, Google Cloud Bigtable, a horizontally scalable managed database: Bigtable uses the Apache HBase interface, which gives customers the benefit of code portability. Another example: Google Cloud Dataproc offers the open-source big data environment Hadoop as a managed service.

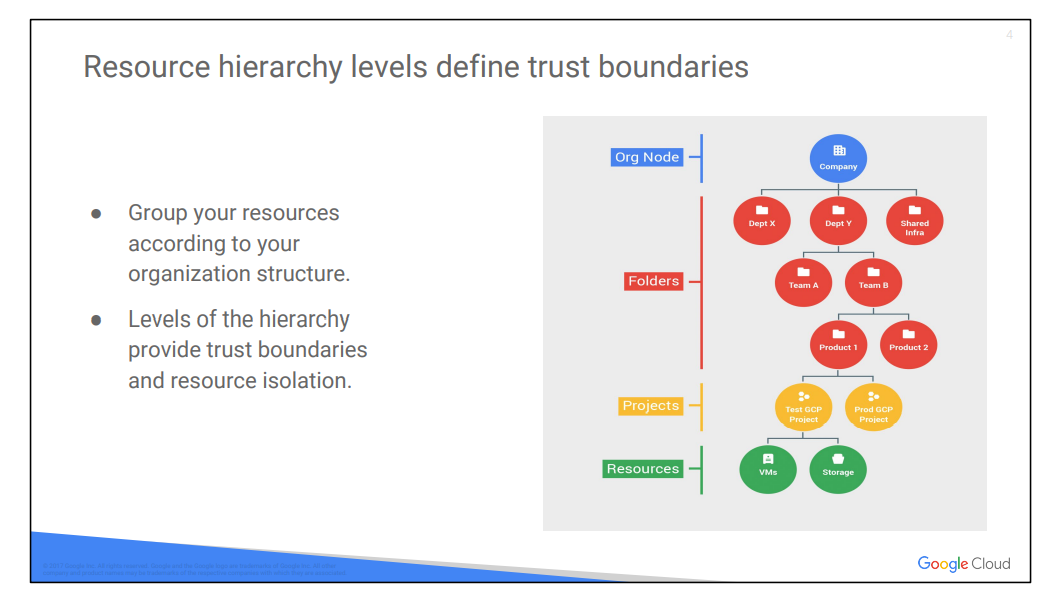
● Google publishes key elements of its technology, using open-source licenses, to create ecosystems that provide customers with options other than Google. For example, TensorFlow, an open-source software library for machine learning developed inside Google, is at the heart of a strong open-source ecosystem.

● Google provides interoperability at multiple layers of the stack. Kubernetes and Google Kubernetes Engine give customers the ability to mix and match microservices running across different clouds. Google Stackdriver lets customers monitor workloads across multiple cloud providers.

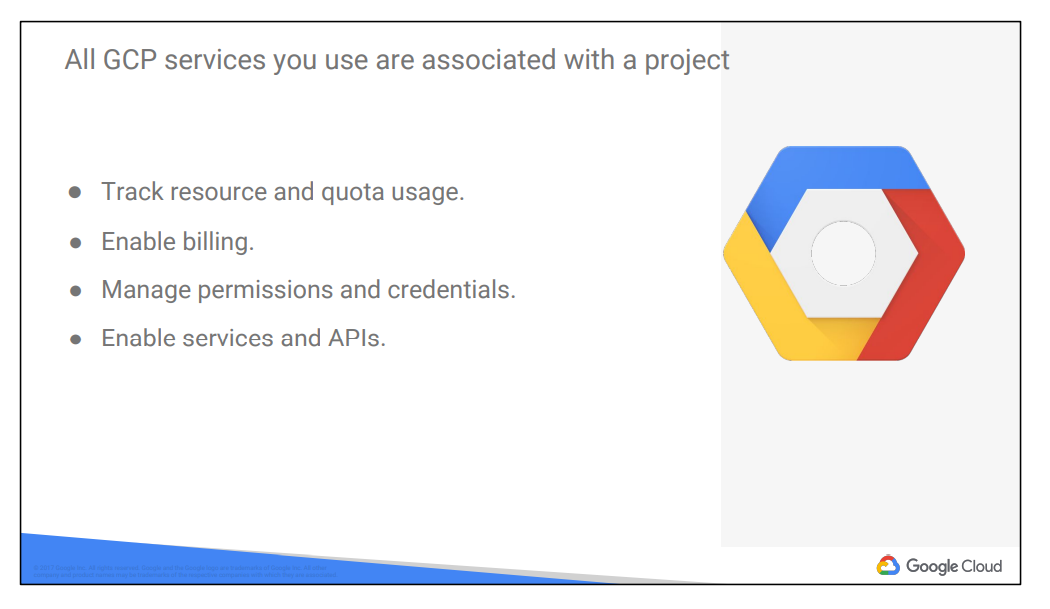


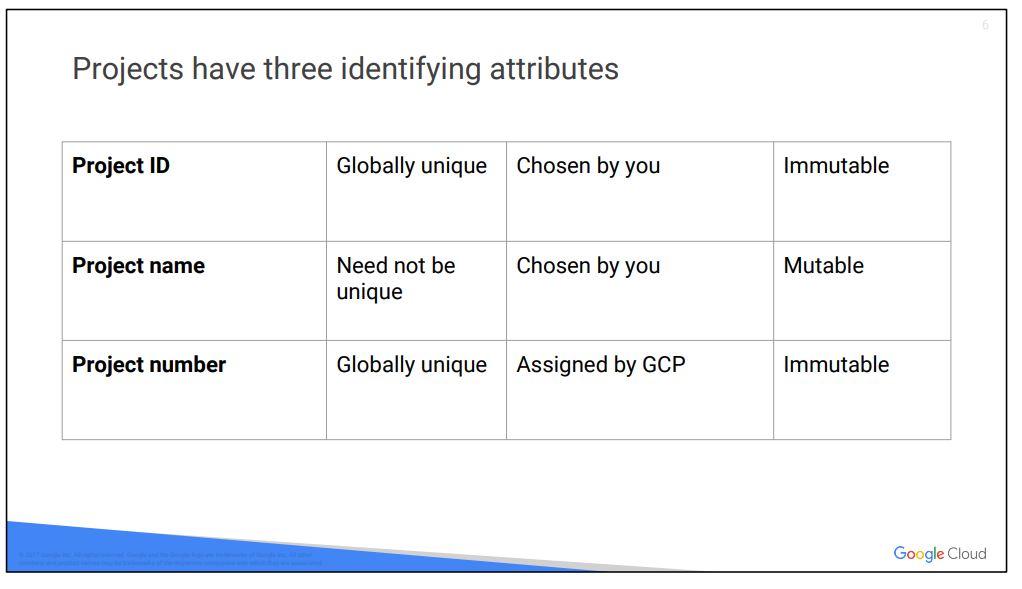
**2.**

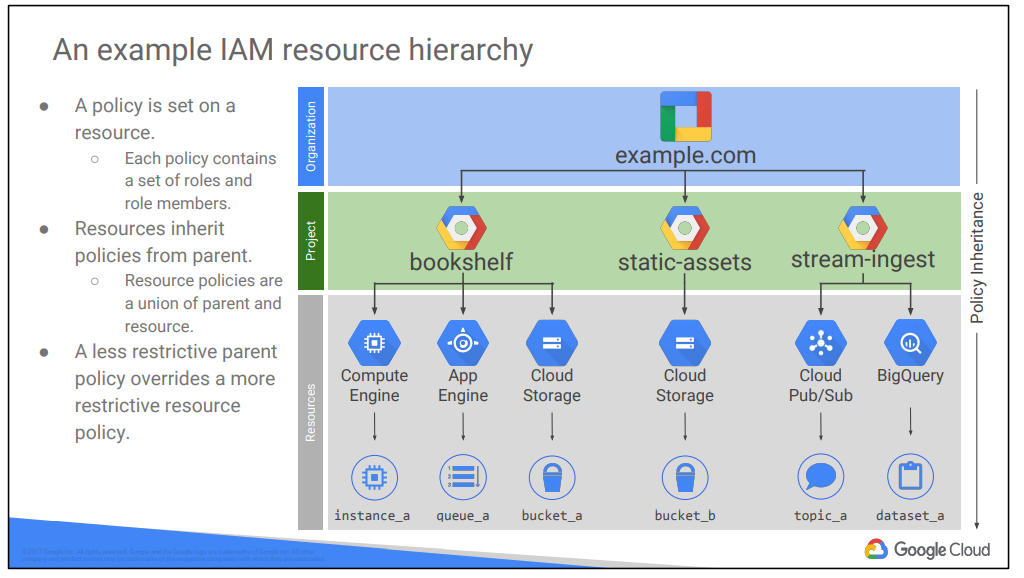




Policies are inherited downwards in the hierarchy.



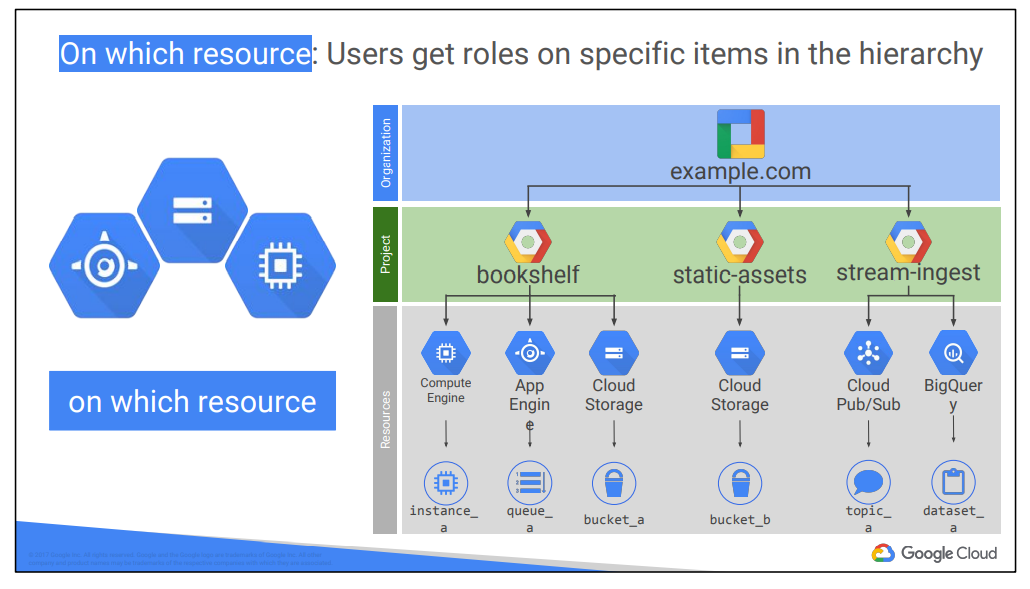
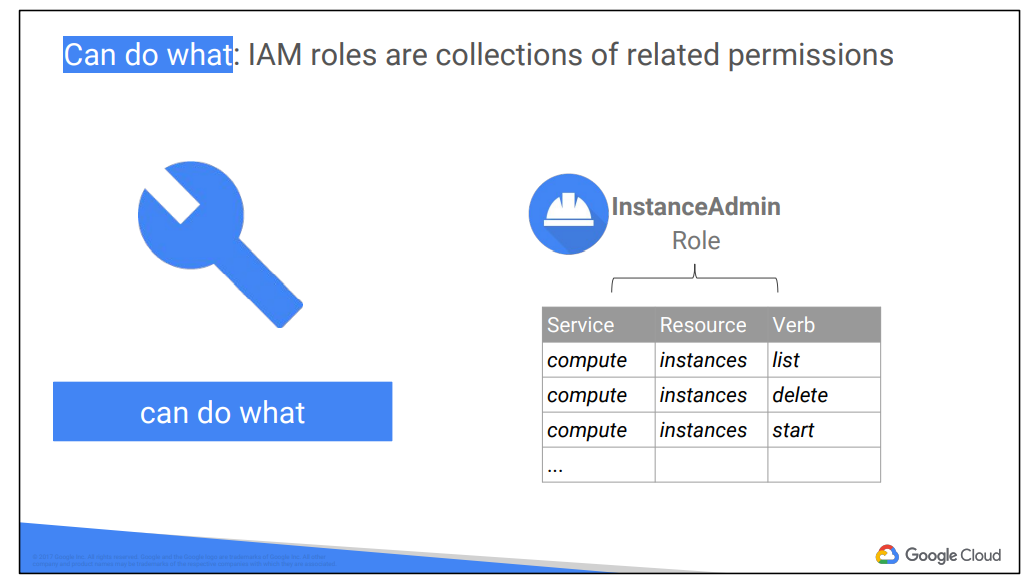
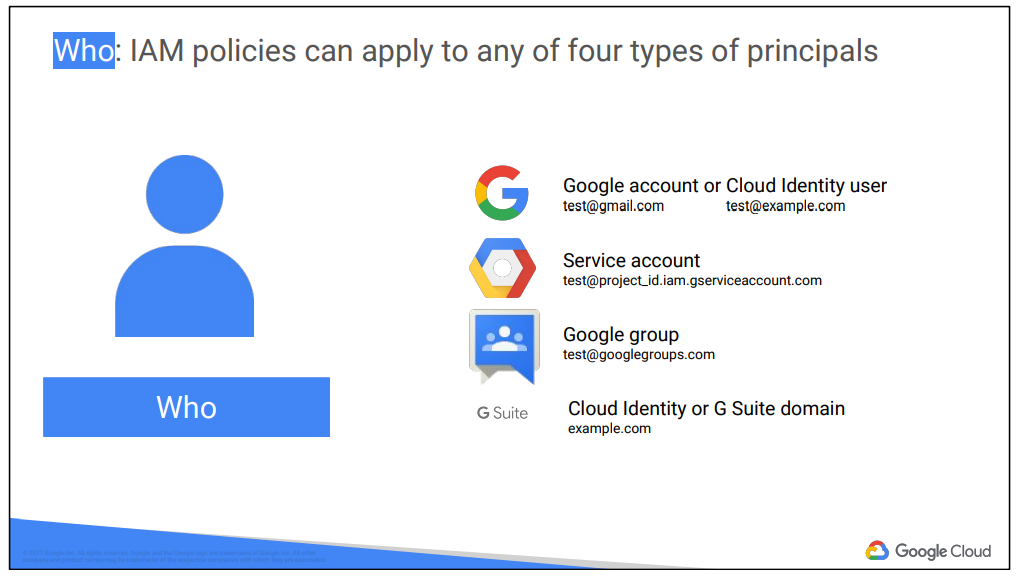




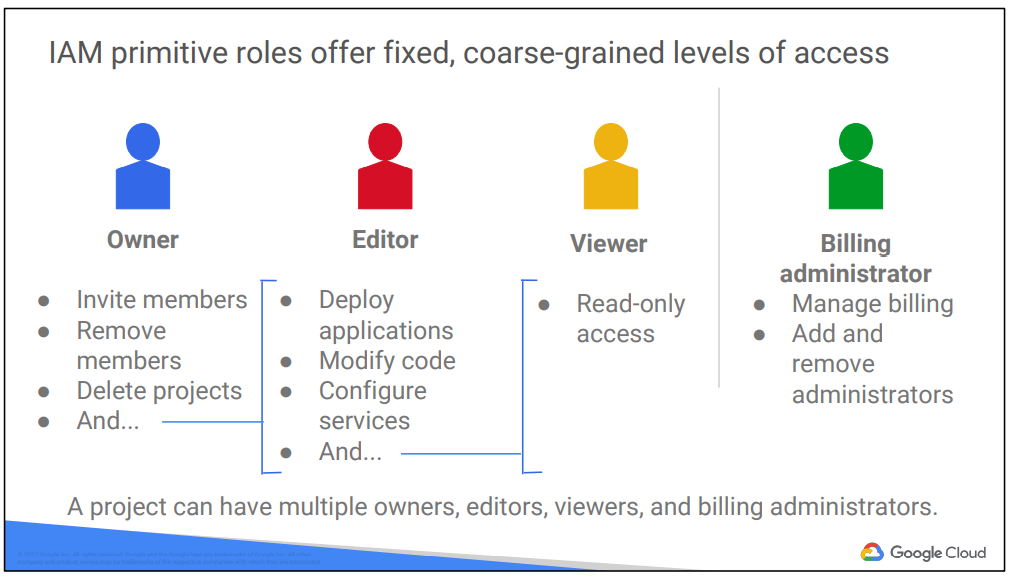
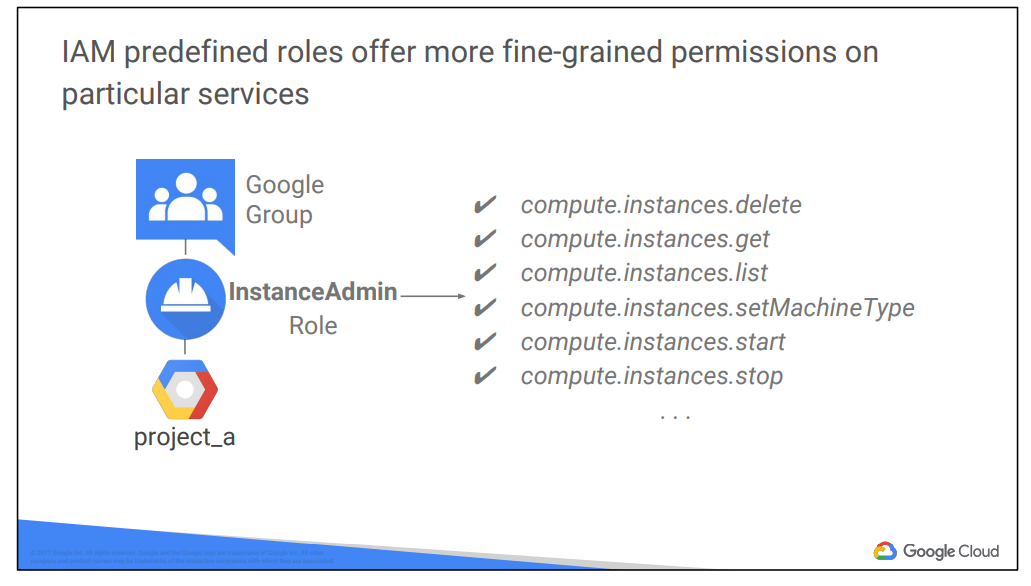
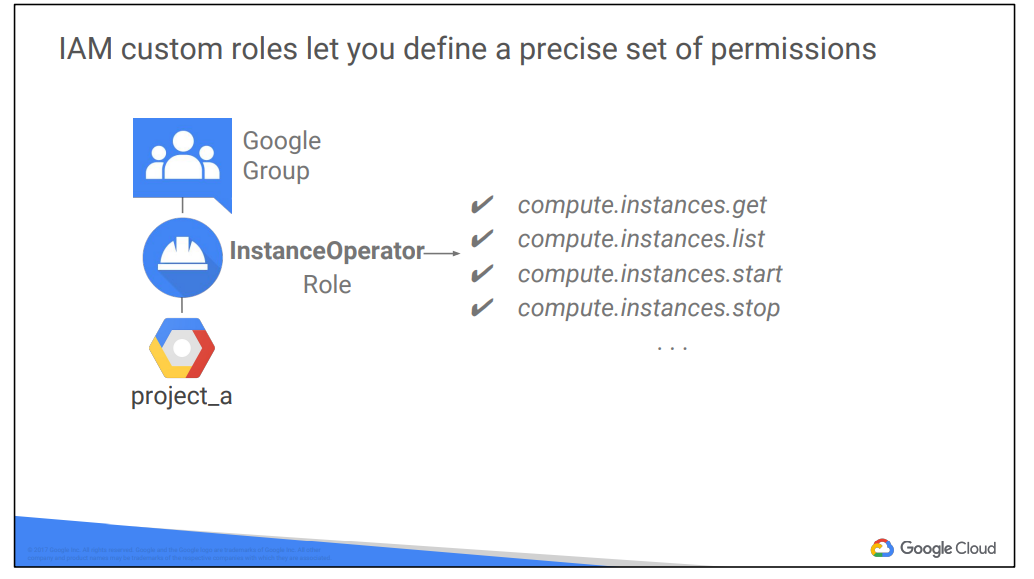
Resources inherit the policies of their parent resource. For instance, if you set a policy at the organization level, it is automatically inherited by all its children projects. And this inheritance is transitive, which means that all the resources in those projects inherit the policy too. The policies implemented at a higher level in this hierarchy can’t take away access that’s granted at lower level.

**IAM :**

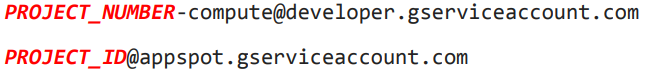
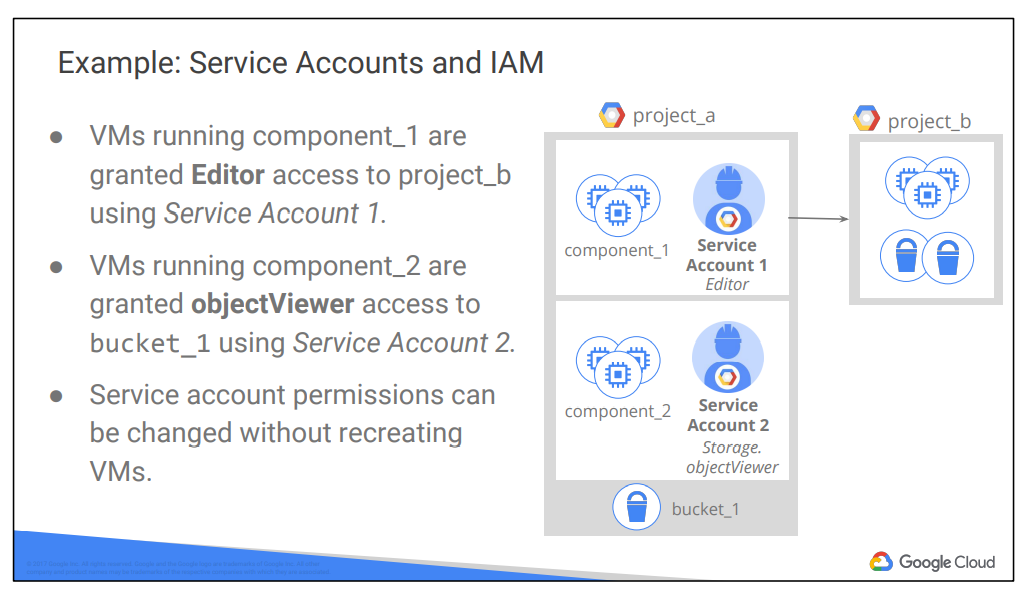
IAM lets administrators authorize who can take action on specific resources. An IAM policy has a “who” part, a “can do what” part, and an “on which resource” part.

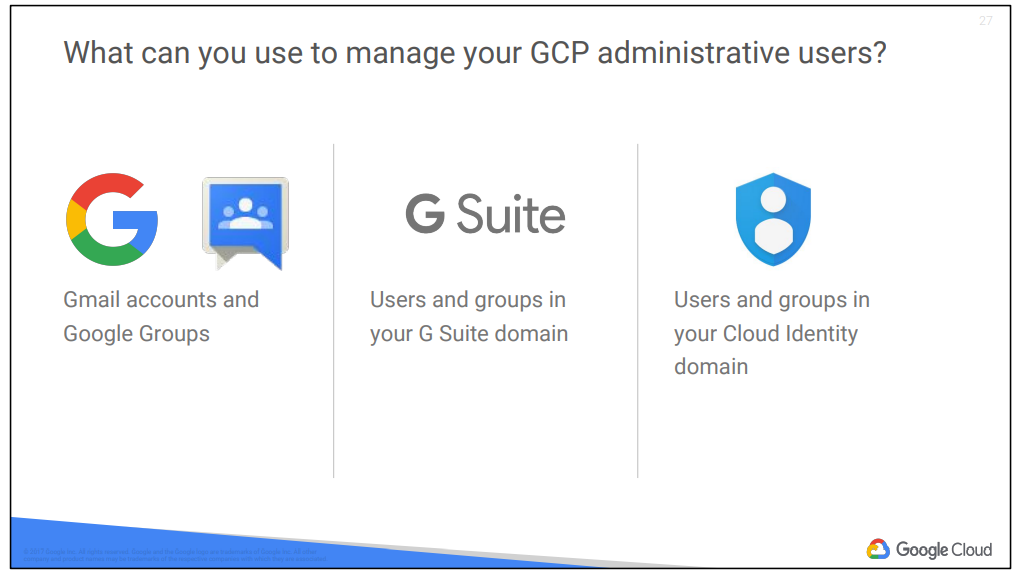


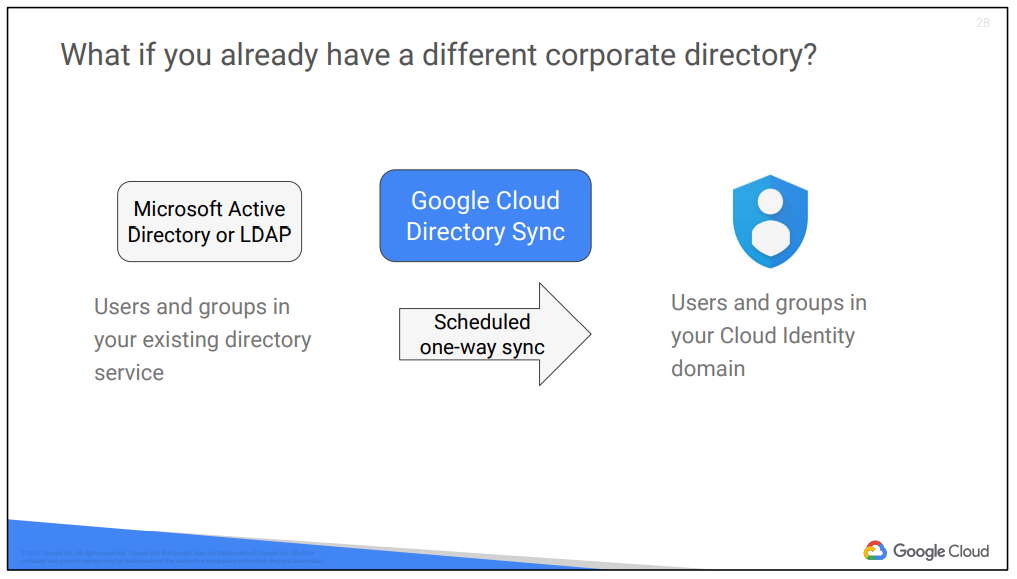
There are three types of IAM roles :

* **Primitive :** Primitive roles are broad. You apply them to a GCP project, and they affect all resources in that project.
  + 
* **Predefined :** These roles are applied to a particular GCP service in a project.
* **Custom :** It lets to define a precise set of permissions.
  + 

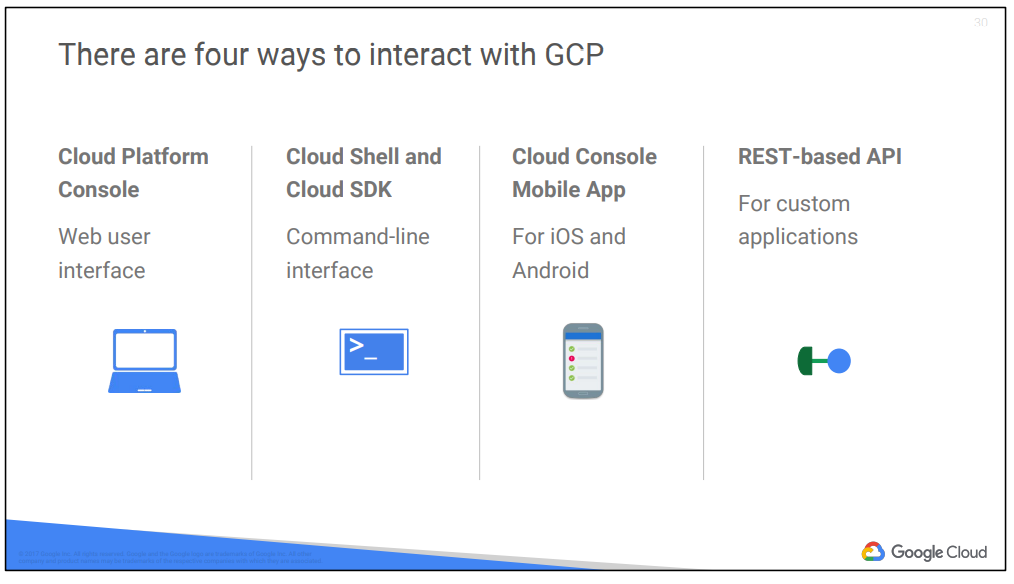
**Service accounts control server to server interactions**

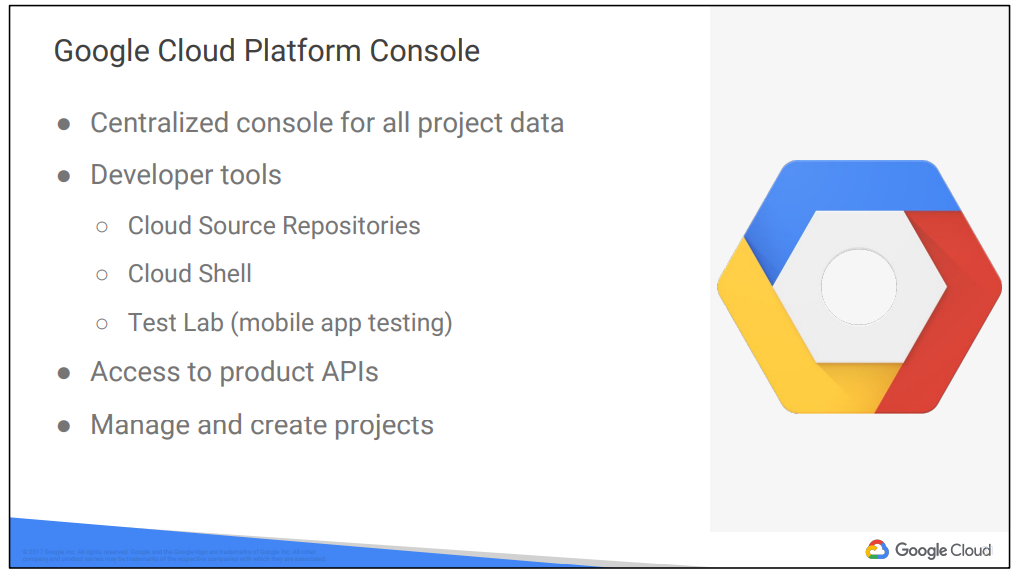
* Provides an identity to carrying out server to server interactions in a project
* Used to authenticate from one server to another
* Used to control privileges control by resources so that applications can perform actions on behalf of authenticated end users
* Identified with an email address
  + 
* It authenticates using keys, it manages keys for compute engine and app engine.
* We can assign predefined or custom IAM role to the service account.
* 

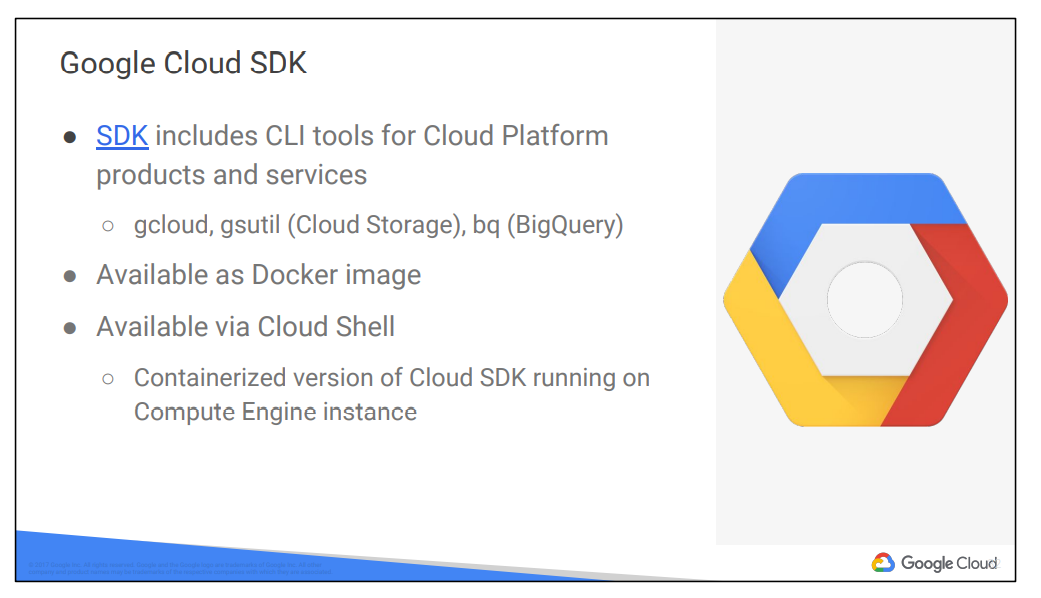


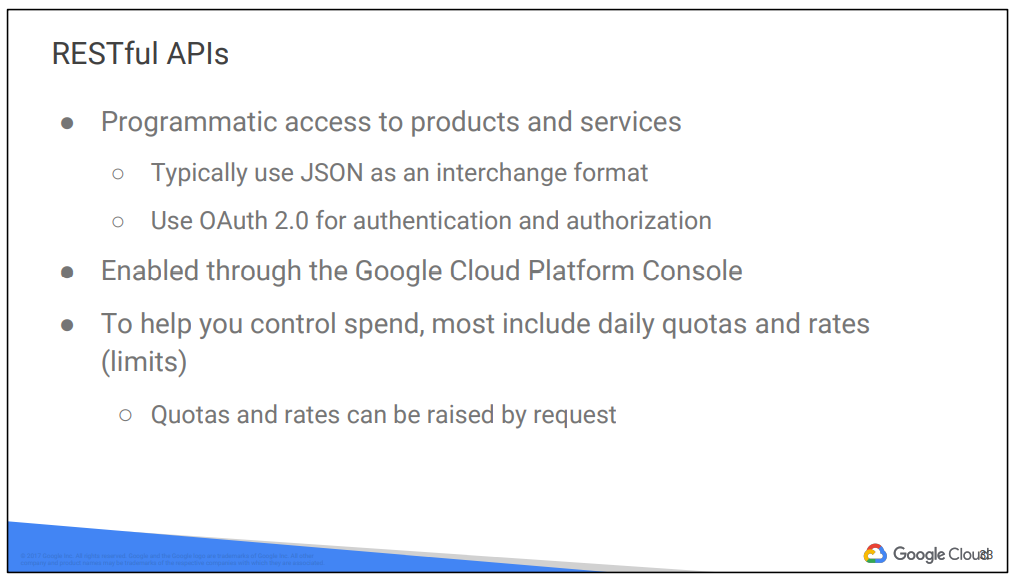


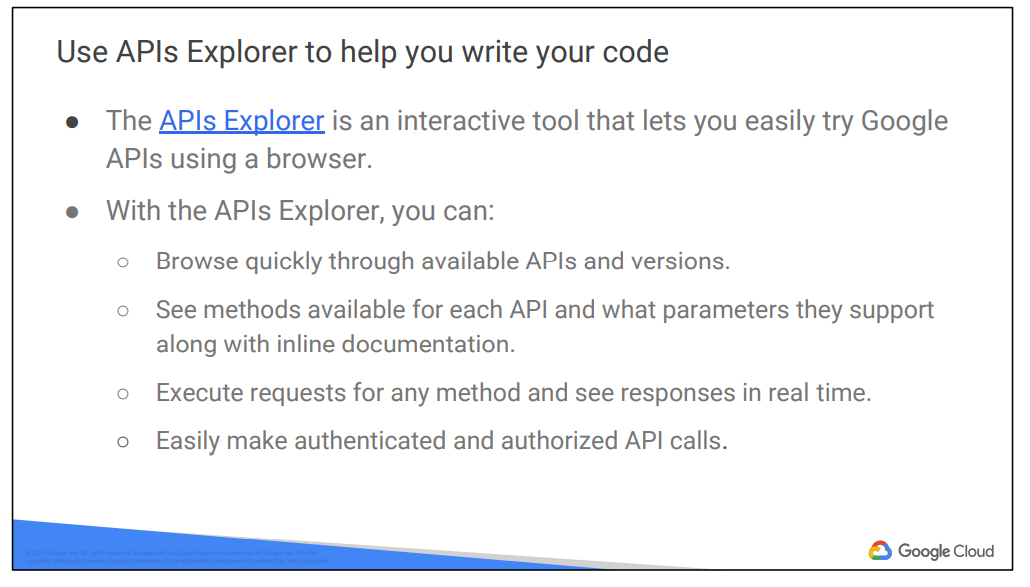
**Interactions with GCP**











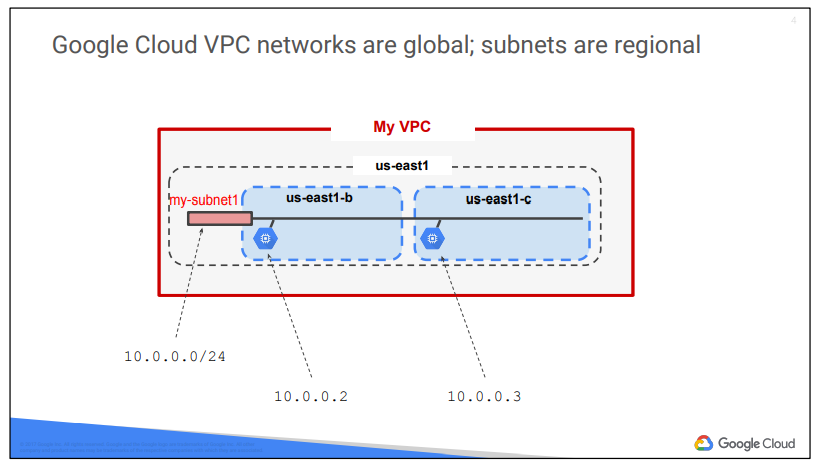


**3.**

**Virtual Private Cloud (VPC)**

Each VPC is contained in a GCP Project. We can provision cloud platform resources, connect them to each other and isolate them from one another. We can segment our networks , use firewall rules to restrict access to instances and create static routes to forward traffic to specific destinations.

VPC has global scope but they can have subnets in any GCP region worldwide.



**Compute Engine :**

Compute Engine offers managed virtual machines.

* High CPU, high memory, standard and shared core machine types
* Persistent disk
  + Standard, SSD , local SSD
  + Snapshots
* Resize disks with no downtime
* Instance metadata and startup scripts
* It offers customer friendly pricing
  + Per second billing, sustained use discounts, committed use discounts
  + Preemptible instances
  + High throughput to storage with no extra cost
  + Custom machine types : only pay for the hardware you need

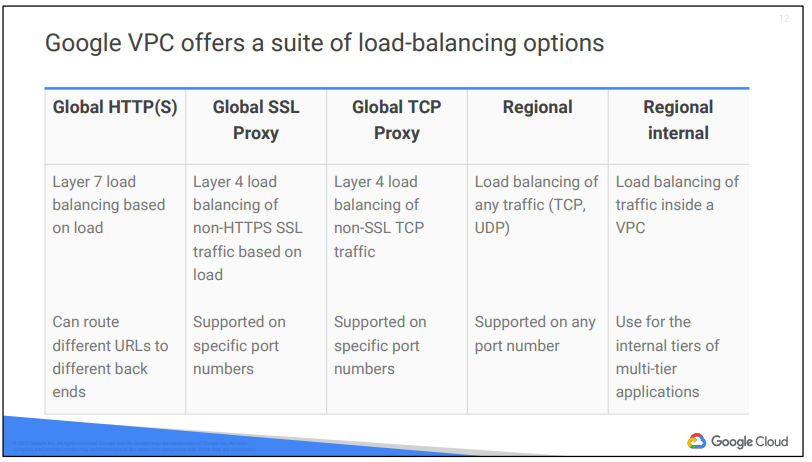
**Important VPC Capabilities**

We control the topology of VPC network :

* Use its route table to forward traffic within network, even across subnets
* Use its firewall to control what traffic network is allowed.
* Use Shared VPC to share a network or individual subnets- with other GCP Projects
* Use VPC Peering to interconnect networks in GCP Projects

With global cloud load balancing, the application presents a single frontend to the world

* Users gets a single , anycast IP address
* Traffic goes over the global backbone from the closest point of presence to the user
* Backends are selected based on load.
* Only healthy backends receive traffic.
* No prewarning is required.



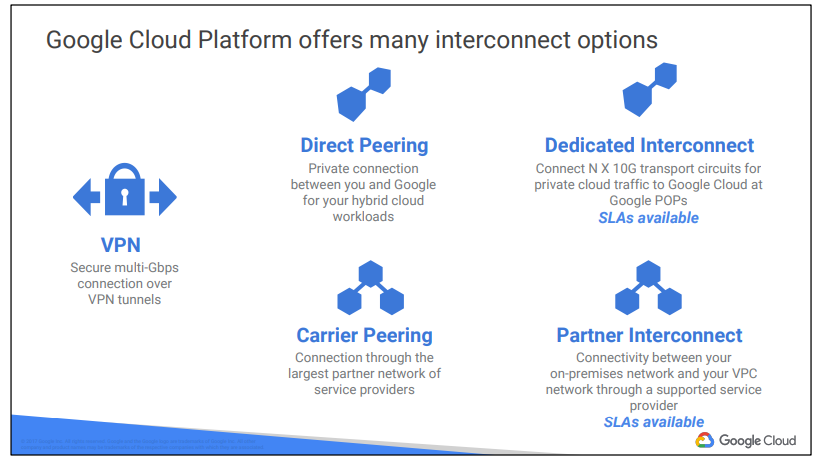
If you need cross-regional load balancing for a Web application, use HTTP(S) load balancing. For Secure Sockets Layer traffic that is not HTTP, use the Global SSL Proxy load balancer. If it’s other TCP traffic that does not use Secure Sockets Layer, use the Global TCP Proxy load balancer. Those two proxy services only work for specific port numbers, and they only work for TCP. If you want to load balance UDP traffic, or traffic on any port number, you can still load balance across a GCP region with the Regional load balancer. But what if you want to load balance traffic inside your project, say, between the presentation layer and the business layer of your application? For that, use the Internal load balancer. It accepts traffic on a GCP internal IP address and load balances it across Compute Engine VMs.

**Cloud DNS :**

GCP offers Cloud DNS to help the world find them. It’s a managed DNS service running on the same infrastructure as Google. It has low latency and high availability, and it’s a cost-effective way to make your applications and services available to your users. The DNS information you publish is served from redundant locations around the world. Cloud DNS is also programmable. You can publish and manage millions of DNS zones and records using the GCP Console, the command-line interface, or the API.

**Cloud CDN (Content Delivery Network)**

Google has a global system of edge caches. You can use this system to accelerate content delivery in your application using Google Cloud CDN. Your customers will experience lower network latency, the origins of your content will experience reduced load, and you can save money too. Once you've set up HTTP(S) Load Balancing, simply enable Cloud CDN with a single checkbox.



**Direct Peering :**

Direct Peering enables you to establish a direct [peering](https://www.wikipedia.org/wiki/Peering) connection between your business network and Google's edge network and exchange high-throughput cloud traffic.

When established, Direct Peering provides a direct path from your on-premises network to Google services, including Google Cloud products that can be exposed through one or more public IP addresses. Traffic from Google's network to your on-premises network also takes that direct path, including traffic from VPC networks in your projects.

**Career Peering :**

Carrier Peering enables you to access Google applications, such as Google Workspace, by using a [service provider](https://cloud.google.com/network-connectivity/docs/carrier-peering#service-providers) to obtain enterprise-grade network services that connect your infrastructure to Google.

When connecting to Google through a service provider, you can get connections with higher availability and lower latency, using one or more links. Work with your service provider to get the connection that you need.

**Dedicated Interconnect :**

Dedicated Interconnect provides direct physical connections between your on-premises network and Google's network. Dedicated Interconnect enables you to transfer large amounts of data between networks, which can be more cost-effective than purchasing additional bandwidth over the public internet.

**Partner Interconnect :**

Partner Interconnect provides connectivity between your on-premises network and your Virtual Private Cloud (VPC) network through a supported service provider. A Partner Interconnect connection is useful if your data center is in a physical location that can't reach a Dedicated Interconnect colocation facility, or your data needs don't warrant an entire 10-Gbps connection.

**4.**

**Cloud Storage (like S3 in AWS)**

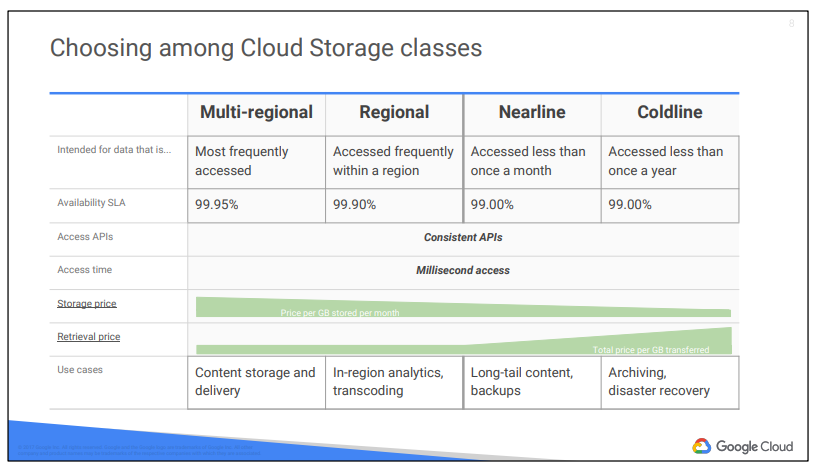
Cloud storage is binary large object storage

* High performance, internet scale
* Simple administration i.e doesn’t require capacity management
* Data encryption at rest
* Data encryption in transit be default from Google to endpoint
* Online and offline import services are available

**Object Storage :**

It’s not the same as file storage, in which you manage your data as a hierarchy of folders. It’s not the same as block storage, in which your operating system manages your data as chunks of disk. Instead, object storage means this: you say to your storage, “Here, keep this arbitrary sequence of bytes,,” and the storage lets you address it with a unique key. In Google Cloud Storage and in other systems, these unique keys are in the form of URLs, which means object storage interacts well with web technologies. Google Cloud Storage always encrypts your data on the server side, before it is written to disk, at no additional charge. Data traveling between a customer’s device and Google is encrypted by default using HTTPS/TLS (Transport Layer Security). In fact, Google was the first major cloud provider to enable HTTPS/TLS by default.

* Your Cloud Storage files are organized into buckets. When you create a bucket: you give it a globally-unique name; you specify a geographic location where the bucket and its contents are stored; and you choose a default storage class. Pick a location that minimizes latency for your users. For example, if most of your users are in Europe, you probably want to pick a European location: a GCP region in Europe, or else the EU multi-region.
* There are several ways to control users’ access to your objects and buckets. For most purposes, Cloud IAM is sufficient. Roles are inherited from project to bucket to object. If you need finer control, you can create access control lists (“ACLs”) that offer finer control, ACLs define who has access to your buckets and objects, as well as what level of access they have. Each ACL consists of two pieces of information: A scope, which defines who can perform the specified actions (for example, a specific user or group of users). And a permission, which defines what actions can be performed (for example, read or write).
* Remember that Cloud Storage objects are immutable. You can turn on object versioning on your buckets if you want. If you do, Cloud Storage keeps a history of modifications--that is, overwrites or deletes--of all objects in the bucket. You can list the archived versions of an object, restore an object to an older state, or permanently delete a version, as needed. If you don’t turn on object versioning, new always overwrites old.



**Regional Storage** lets you store your data in a specific GCP region, us-central1, europe-west1 or asia-east1. It’s cheaper than multi-regional storage, but it offers less redundancy.

**Multi-Regional Storage** costs a bit more, but it’s geo-redundant. That means you pick a broad geographical location, like United States, the European Union, or Asia, and Cloud Storage stores your data in at least two geographic locations separated by at least 160 kilometers.

Multi-Regional Storage is appropriate for storing frequently accessed storing data: website content, interactive workloads, or data that’s part of mobile and gaming applications.

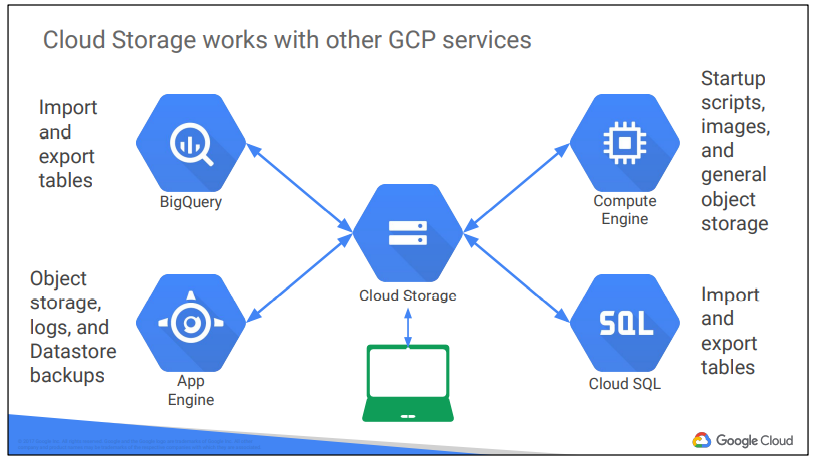
**Nearline storage** is a low-cost, highly durable storage service for storing infrequently accessed data. This storage class is a better choice than Multi-Regional Storage or Regional Storage in scenarios where you plan to read or modify your data on average once a month or less.

**Coldline Storage** is a very-low-cost, highly durable storage service for data archiving, online backup, and disaster recovery. Coldline Storage is the best choice for data that you plan to access at most once a year, due to its slightly lower availability, 90-day minimum storage duration, costs for data access, and higher per-operation costs.

The availability of these storage classes varies, with multi-regional having the highest availability of 99.95%, followed by regional with 99.9% and nearline and coldline with 99.0%.

**Data Transfer into Cloud Storage**

* Many customers simply use gsutil, which is the Cloud Storage command from the Cloud SDK. You can also move data in with a drag and drop in the GCP Console, if you use the Google Chrome browser.
* **The Storage Transfer Service** lets you schedule and manage batch transfers to Cloud Storage from another cloud provider, from a different Cloud Storage region, or from an HTTP(S) endpoint.
* **The Transfer Appliance** is a rackable, high-capacity storage server that you lease from Google Cloud. You simply connect it to your network, load it with data, and then ship it to an upload facility where the data is uploaded to Cloud Storage.



**Cloud BigTable**

Cloud Bigtable is Google's NoSQL big data database service. It's the same database that powers many core Google services, including Search, Analytics, Maps, and Gmail.

Cloud Bigtable is offered as a fully managed service, which means that you spend your time developing valuable applications instead of configuring and tuning your database for performance and scalability. In addition, Google’s own Bigtable operations team monitors the service to ensure that issues are addressed quickly.

You can use Cloud Bigtable to store and query all of the following types of data:

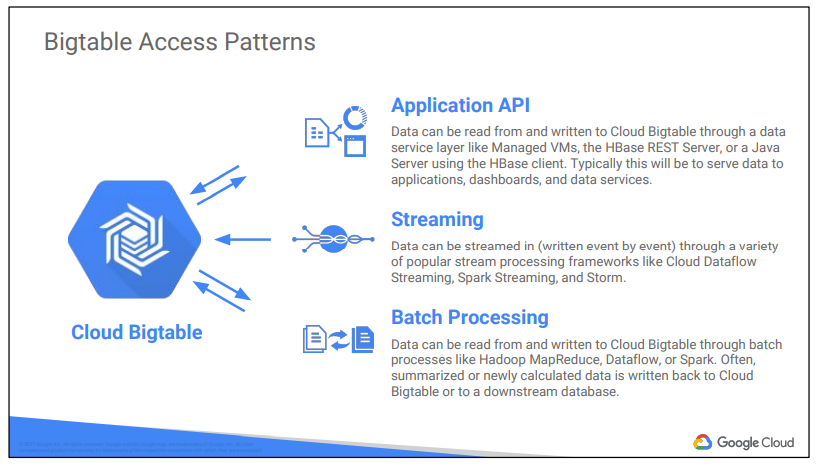
● Marketing data, such as purchase histories and customer preferences

● Financial data, such as transaction histories, stock prices, and currency exchange rates

● Internet of Things data, such as usage reports from energy meters and home appliances

● Time-series data, such as CPU and memory usage over time for multiple servers

Cloud Bigtable is offered through the same open source API as HBase, the native Hadoop database. This enables portability of applications between HBase and Bigtable.

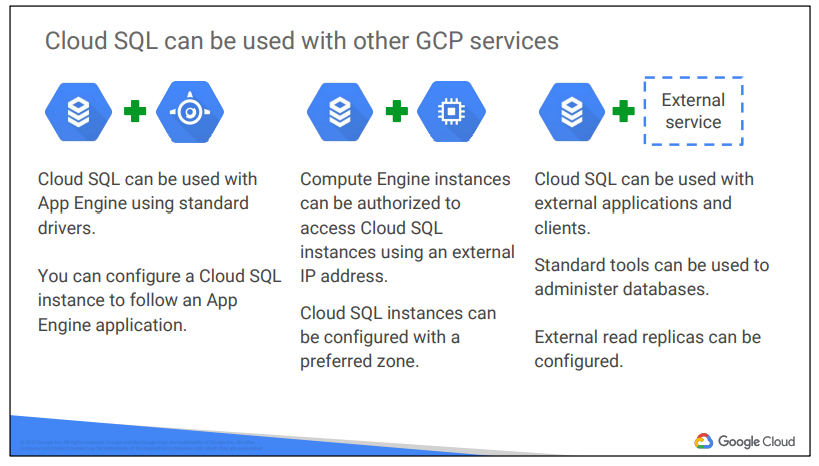


**Cloud SQL and Cloud Spanner**

Cloud SQL is a managed RDBMS, it offers MySQL and Postgress databases as a service.

Features :

* Automatic Replication :
  + Cloud SQL instances replicating from a Cloud SQL master instance Replicas are other instances in the same project and location as the master instance.
  + Cloud SQL instances replicating from an external master instance The master instance is external to Google Cloud SQL. can be outside the Google network or in a Google Compute Engine instance.
  + External MySQL instances replicating from a Cloud SQL master instance External replicas are in hosting environments, outside of Cloud SQL
* Managed Backups :
  + Cloud SQL takes care of securely storing your backed-up data and makes it easy for you to restore from a backup and perform a point-in-time recovery to a specific state of an instance. Cloud SQL retains up to 7 backups for each instance, which is included in the cost of your instance.



Cloud Spanner is a horizontally scalable RDBMS.

It supports :

* Automatic Replication
* Strong global consistency
* Managed instances with high availability
* SQL (ANSI 2011 with extensions)

Cloud Spanner is especially suited for applications requiring:

● A SQL RDBMS, with joins and secondary indexes

● Built-in high availability

● Strong global consistency

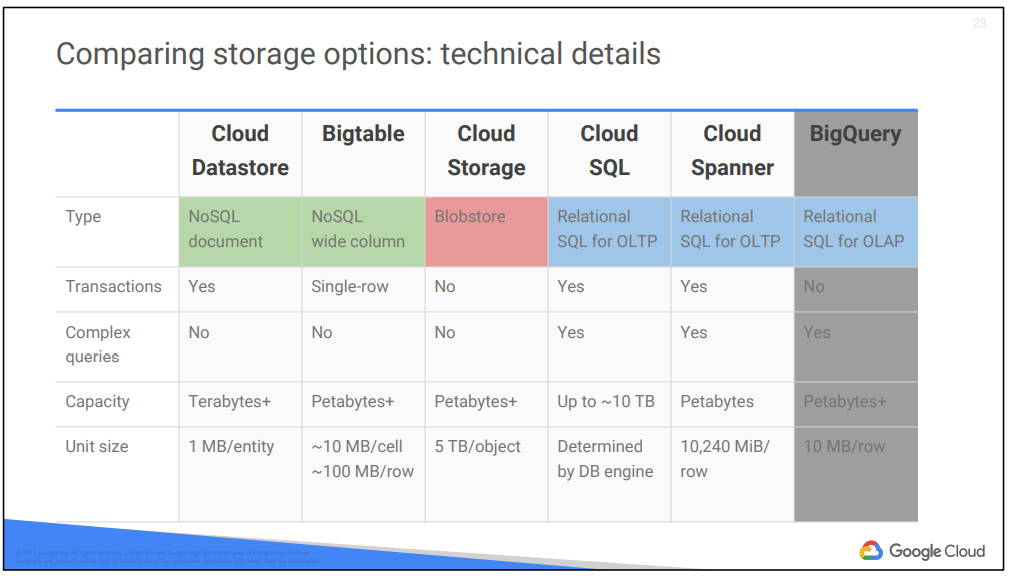
● Database sizes exceeding ~2 TB

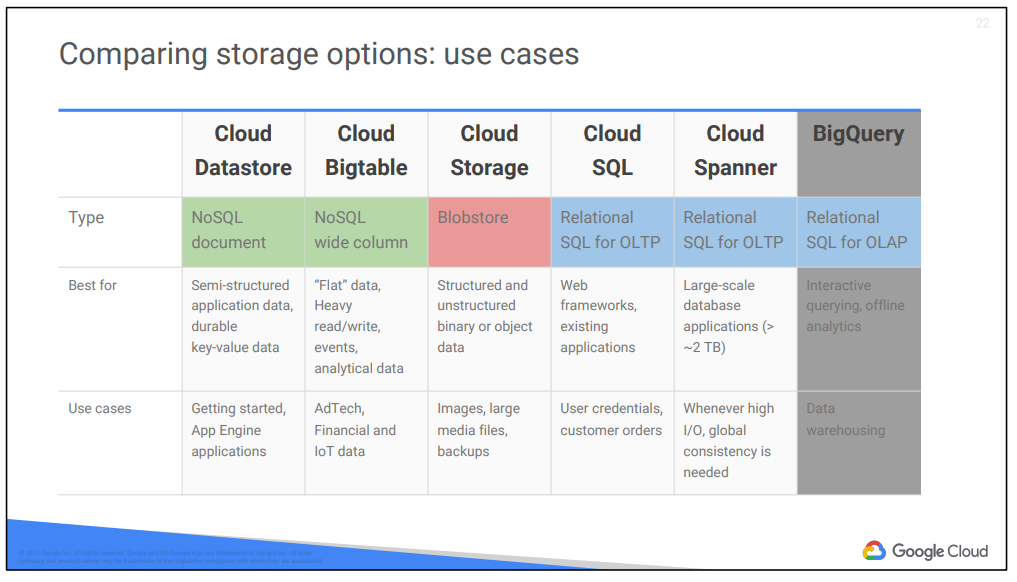
● Many IOPS (Tens of thousands of reads/writes per second or more)

**Cloud Datastore**

* NoSQL designed for application backends
* Fully Managed , uses a distributed architecture to automatically manage scaling
* Built in redundancy
* Supports ACID Transactions
* Schemaless Access i.e no need to think about underlying data structure
* Local Development tools
* Includes a free daily quota
* Access from anywhere through a RESTFUL interface

**Comparing storage options :**





**5.**

Compute Engine : GCP IAAS

App Engine : GCP PAAS

Containers and Kubernetes Engine : Hybrid between both



**6.**

**Google App Engine**

App Engine is a platform for building scalable web applications and mobile backends. It allows you to concentrate on innovating your applications by managing the application infrastructure for you. For example, App Engine manages the hardware and networking infrastructure required to run your code.

App Engine provides you with built-in services and APIs such as NoSQL datastores, memcache, load balancing, health checks, application logging, and a user authentication API, common to most applications.

App Engine will scale your application automatically in response to the amount of traffic it receives so you only pay for the resources you use. Just upload your code and Google will manage your app's availability. There are no servers for you to provision or maintain.

App Engine works with popular development tools such as Eclipse, IntelliJ, Maven, Git, Jenkins, and PyCharm.

**Google App Engine Standard Environment**

The App Engine standard environment makes it easy to build and deploy an application that runs reliably even under heavy load and with large amounts of data.

It includes the following features:

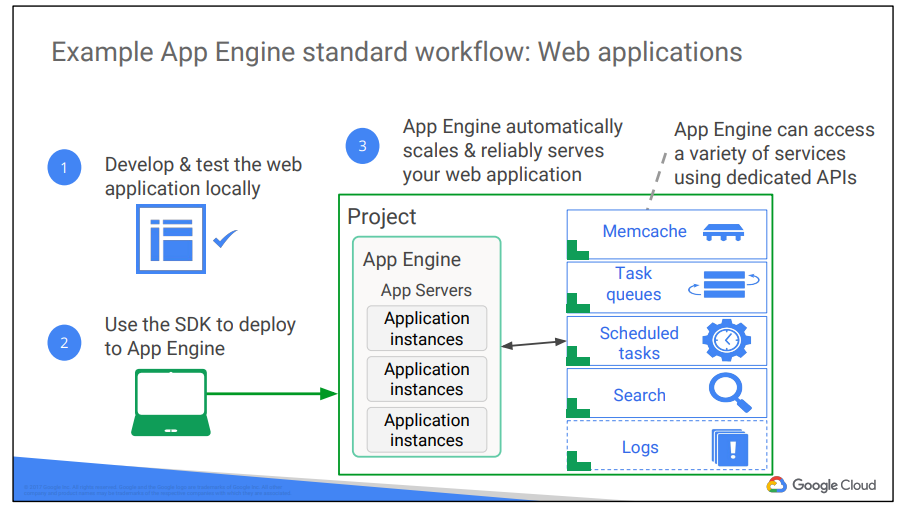
● Persistent storage with queries, sorting, and transactions

● Automatic scaling and load balancing

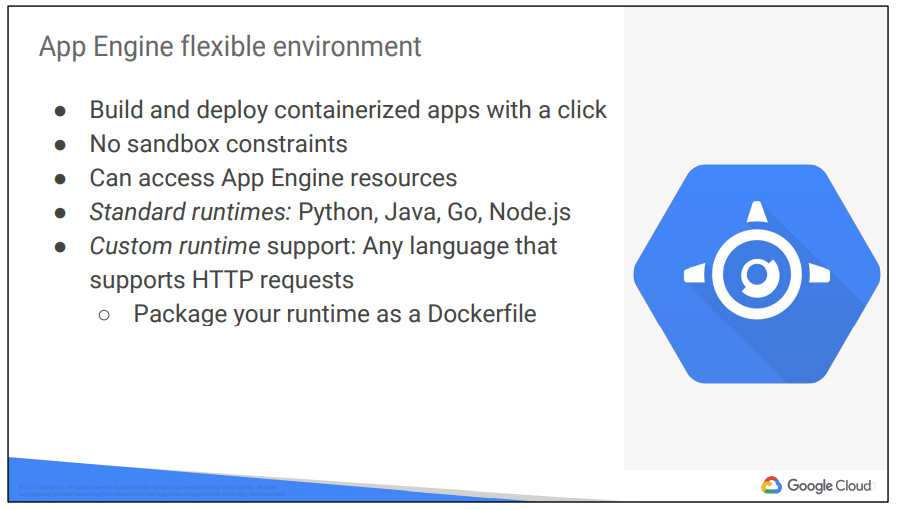
● Asynchronous task queues for performing work outside the scope of a request

● Scheduled tasks for triggering events at specified times or regular intervals

● Integration with other Google cloud services and APIs

* Specific version of JAVA , Python, PHP and GO are supported
* Your application must conform to sandbox constraints
  + No writing to local file system
  + All request time out to 60 seconds
  + Third party software installations are limited

**Google App Engine Flexible Environment**



Your application runs inside Docker containers on Google Compute Engine virtual machines (VMs). App Engine manages these Compute Engine machines for you. They’re health-checked, healed as necessary, and you get to choose what geographical region they run in. And critical, backward-compatible updates to their operating systems are automatically applied. All this so that you can just focus on your code.

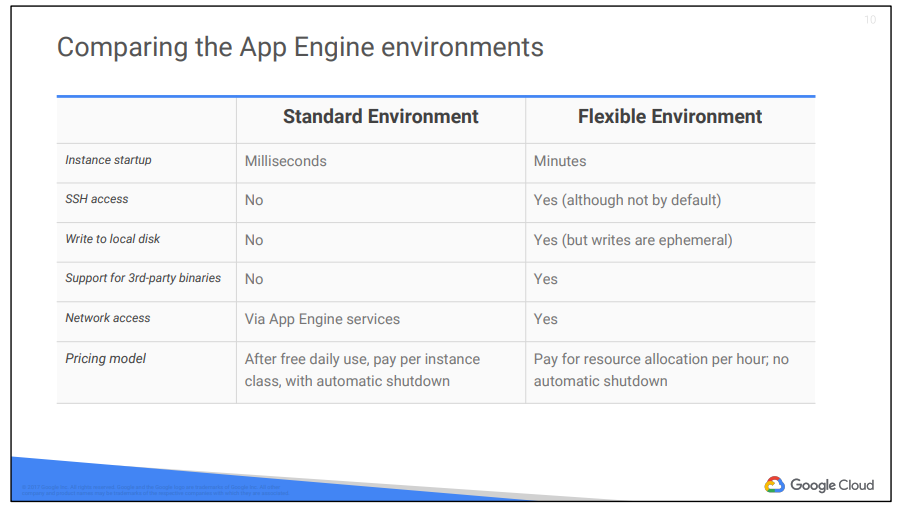
Microservices, authorization, SQL and noSQL databases, traffic splitting, logging, search, versioning, security scanning, memcache, and content delivery networks are all supported natively. In addition, the App Engine flexible environment allows you to customize your runtime and even the operating system of your virtual machine using Dockerfiles

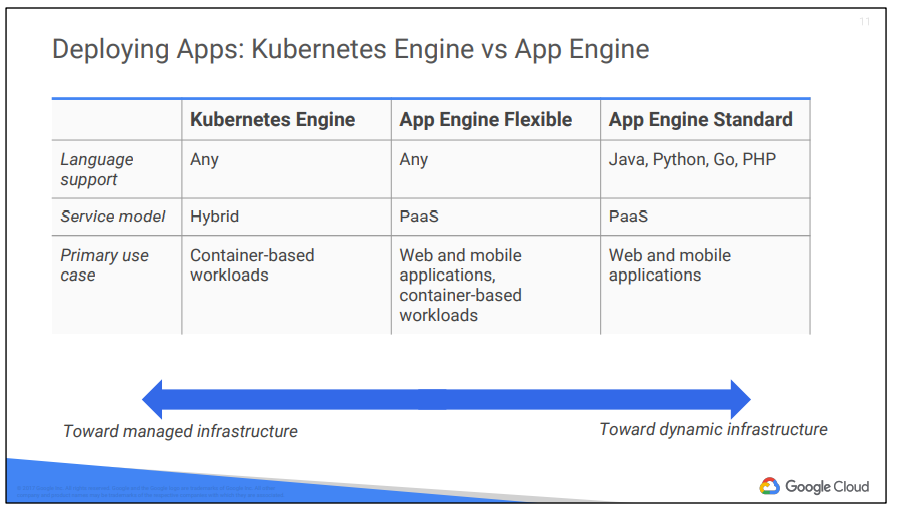
● Runtimes: The flexible environment includes native support for Java 8/Servlet 3.1/Jetty 9, Python 2.7 and Python 3.4, Node.js, and Go. Developers can customize these runtimes or provide their own runtime, such as Ruby or PHP, by supplying a custom Docker image or

● Dockerfile from the open source community.

● Infrastructure customization: Because VM instances in the flexible environment are Compute Engine virtual machines, you can use SSH to connect to every single VM and Docker container for debugging purposes and further customization.

● Performance: Take advantage of a wide array of CPU and memory configurations. You can specify how much CPU and memory each instance of your application needs, and the flexible environment will provision the necessary infrastructure for you





**Cloud Endpoints :**

Cloud Endpoints is a distributed API management system. It provides an API console, hosting, logging, monitoring, and other features to help you create, share, maintain, and secure your APIs. You can use Cloud Endpoints with any APIs that support the OpenAPI Specification, formerly known as the Swagger spec.

Cloud Endpoints uses the distributed Extensible Service Proxy to provide low latency and high performance for serving even the most demanding APIs. Extensible Service Proxy is a service proxy based on NGINX. It runs in its own Docker container for better isolation and scalability. The proxy is containerized and distributed in the Container Registry and Docker registry, and can be used with App Engine, Kubernetes Engine, Compute Engine or Kubernetes.

Cloud Endpoints features

User authentication

● JSON Web Token validation and a streamlined developer experience for Firebase Auth, Google Auth and Auth0.

Automated deployment

● With App Engine, the proxy is deployed automatically with your application. On Kubernetes Engine or Compute Engine, use Google’s containerized ESP for simple deployment.

Logging and monitoring

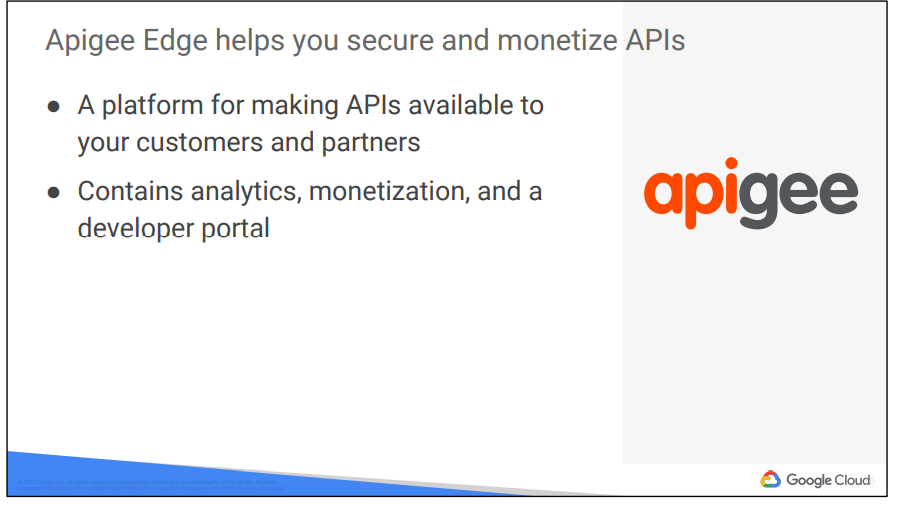
● Monitor traffic, error rates and latency, and review logs in Cloud Logging. Use Cloud Trace to dive into performance and BigQuery for analysis.

API keys

● Generate API keys in Google Cloud Platform Console and validate on every API call. Share your API with other developers to allow them to generate their own keys.

Easy integration

● Get started quickly by using one of Google’s Cloud Endpoints Frameworks or by simply adding an Open API specification to your deployment.



**7.**

**Cloud Source Repositories :**

* Fully featured git repositories hosted on google cloud platform
* Supports collaborative development of google cloud apps
* Includes integration with stackdriver debugger
  + If you are using the Stackdriver Debugger, you can use Cloud Source Repositories and related tools to view debugging information alongside your code during application runtime.

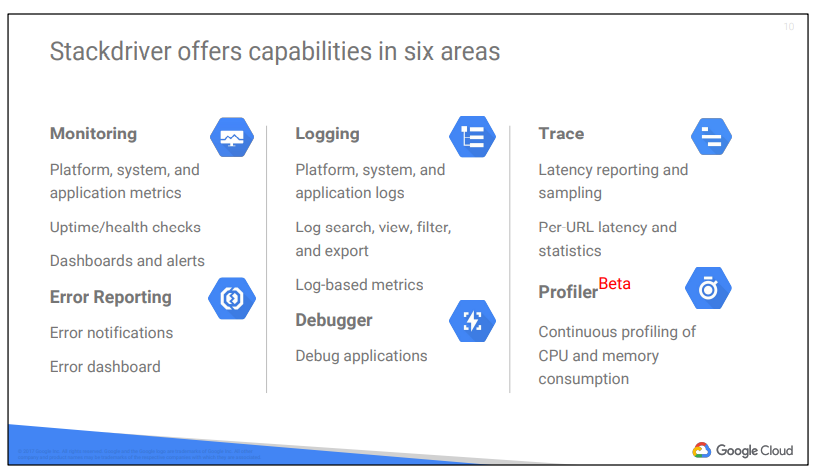
**Cloud Functions**

* Create single purpose functions that responds to events without a server or runtime
* Written in javascript, executed in managed Node.js environment on Google Cloud Platform

**Deployment Manager**

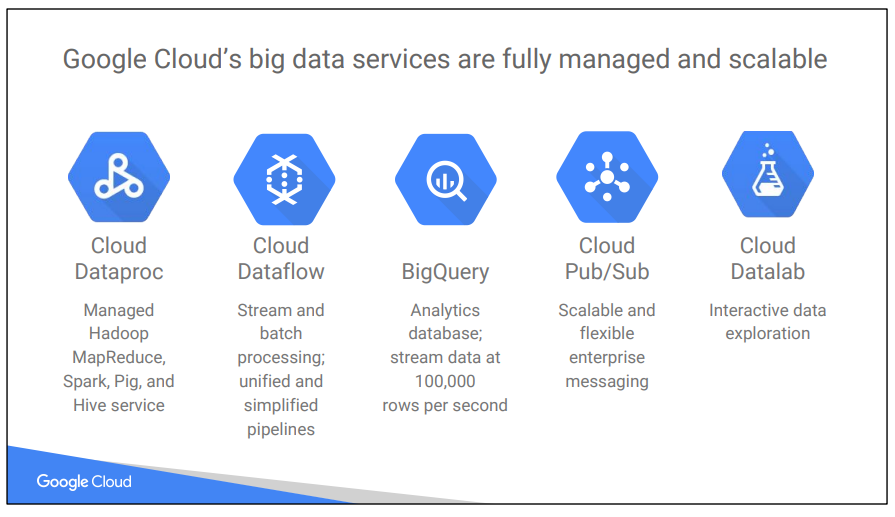
* Infrastructure Management Service
* Create a .yaml template describing your environment and use deployment manager to create resources
* Provides repeatable deployments

**Monitoring**



**8.**

**Data Analysis**



Google Cloud Big Data solutions are designed to help you transform your business and user experiences with meaningful data insights. It is an integrated, serverless platform. “Serverless” means you don’t have to provision compute instances to run your jobs. The services are fully managed, and you pay only for the resources you consume. The platform is “integrated” so GCP data services work together to help you create custom solutions.

**Cloud Dataproc**

Apache Hadoop is an open-source framework for big data. It is based on the MapReduce programming model, which Google invented and published. *The MapReduce model, at its simplest, means that one function -- traditionally called the “map” function -- runs in parallel across a massive dataset to produce intermediate results; and another function -- traditionally called the “reduce” function -- builds a final result set based on all those intermediate results*. The term “Hadoop” is often used informally to encompass Apache Hadoop itself and related projects, such as Apache Spark, Apache Pig, and Apache Hive.

* Cloud Dataproc is a managed Hadoop
* It is fast, easy, managed way to run Hadoop and Spark/Hive/Pig on GCP
* Creates cluster in 90 seconds or less on average
* Scales cluster up and down even when jobs are running

Why Cloud Dataproc :

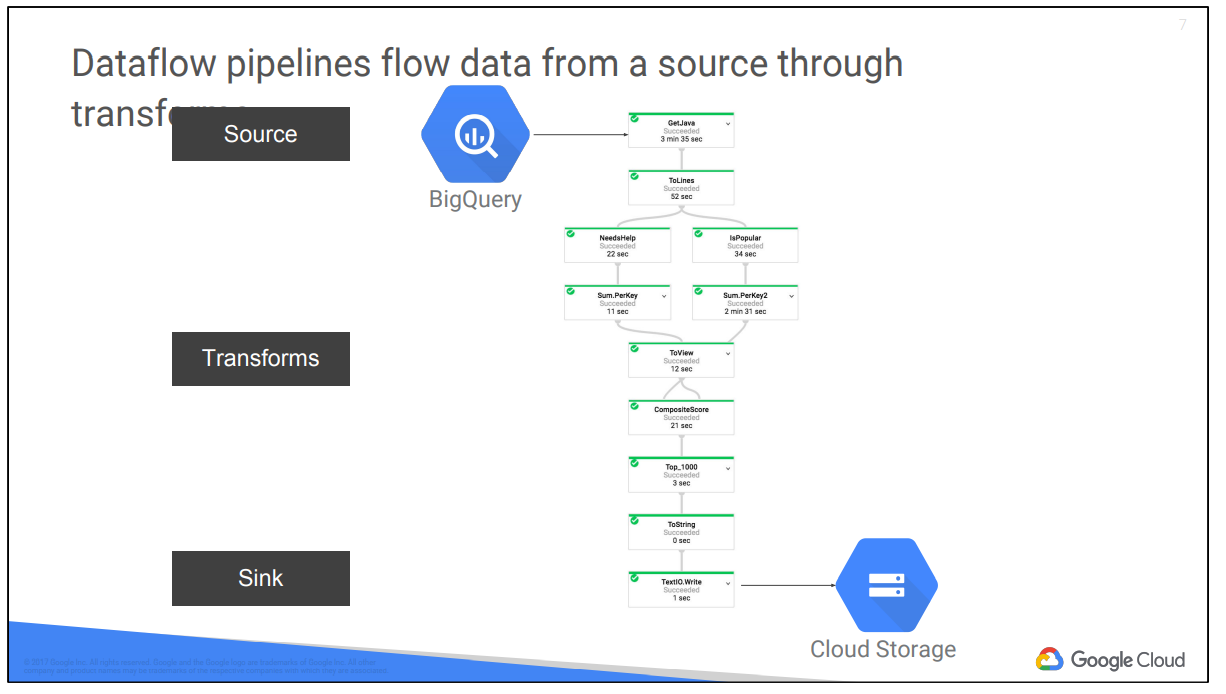
* Easily migrate on-premise Hadoop jobs to the cloud
* Quickly analyze data like log data stored in cloud storage, create a cluster in 90 seconds or less on average and then delete it immediately.
* Use Spark/Spark SQL to quickly perform data mining and analysis.
* Use Spark Machine Learning Libraries(MLib) to run classification algorithms.

Running on-premises Hadoop jobs requires a hardware investment. On the other hand, running these jobs in Cloud Dataproc allows you to pay only for hardware resources during the life of the ephemeral customer you create. You can further save money using preemptible instances for batch processing

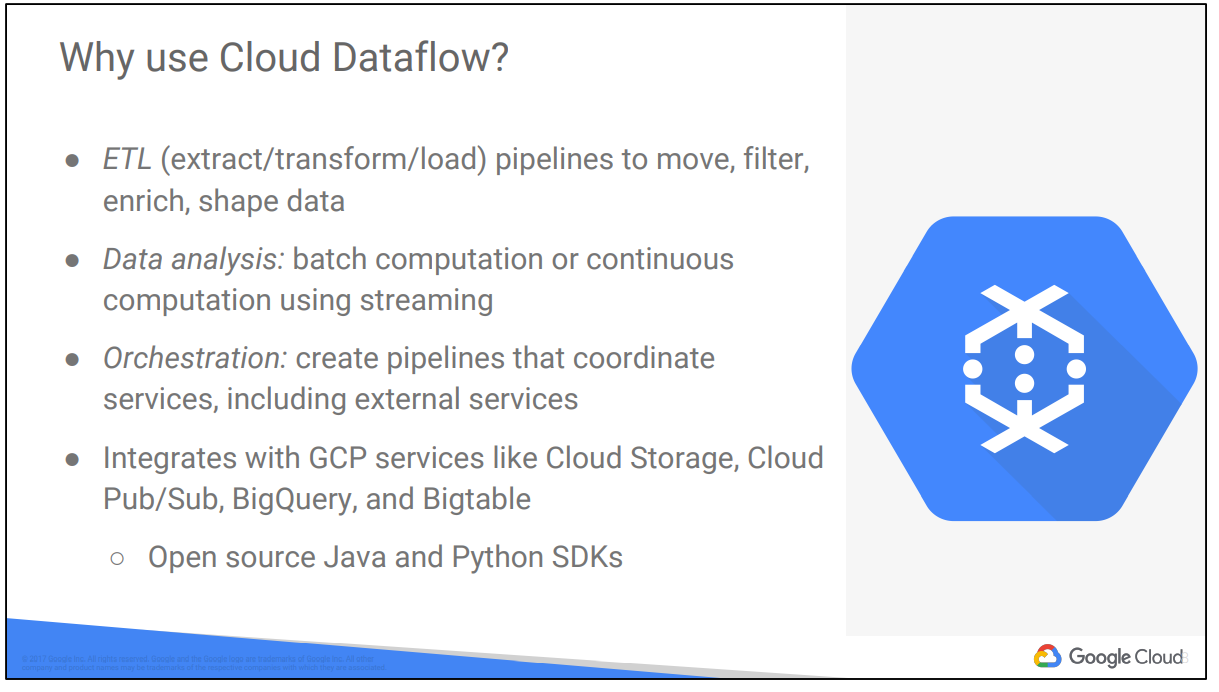
*Cloud Dataproc is great when you have a dataset of known size, or when you want to manage your cluster size yourself, not for realtime data.*

**Cloud Dataflow**

It’s both a unified programming model and a managed service, and it lets you develop and execute a big range of data processing patterns: extract-transform-and-load, batch computation, and continuous computation. You use Dataflow to build data pipelines, and the same pipelines work for both batch and streaming data. 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.

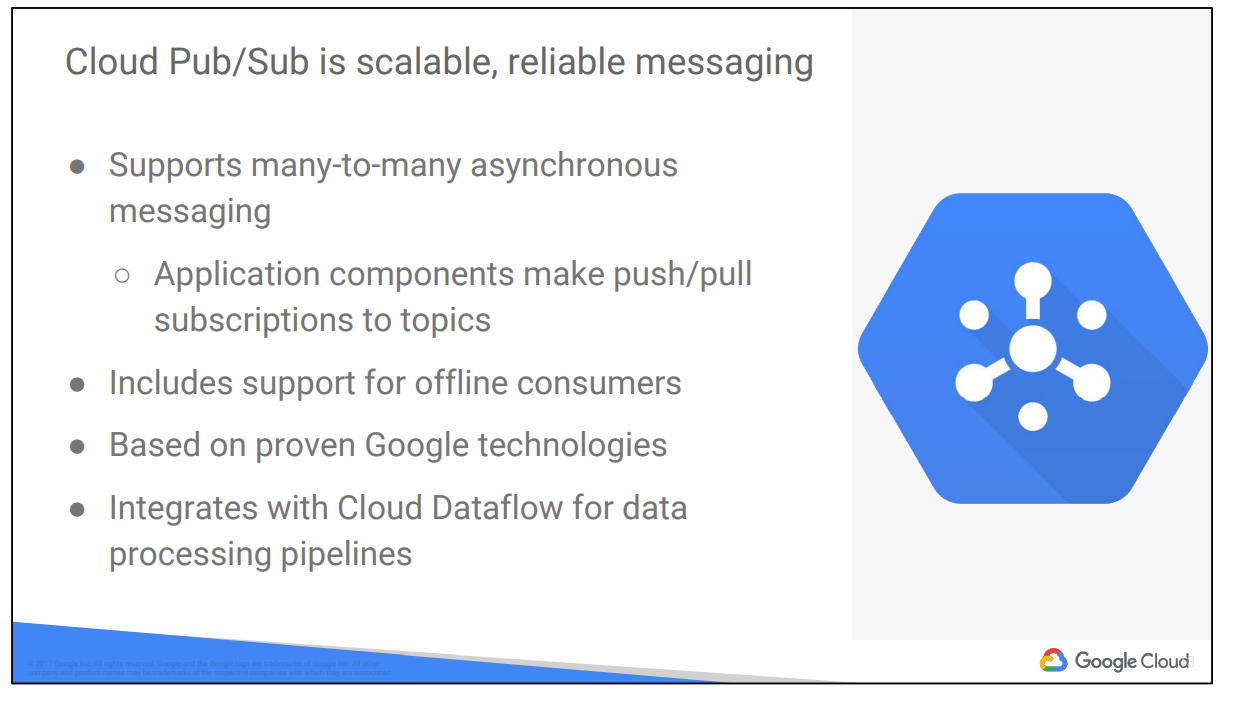


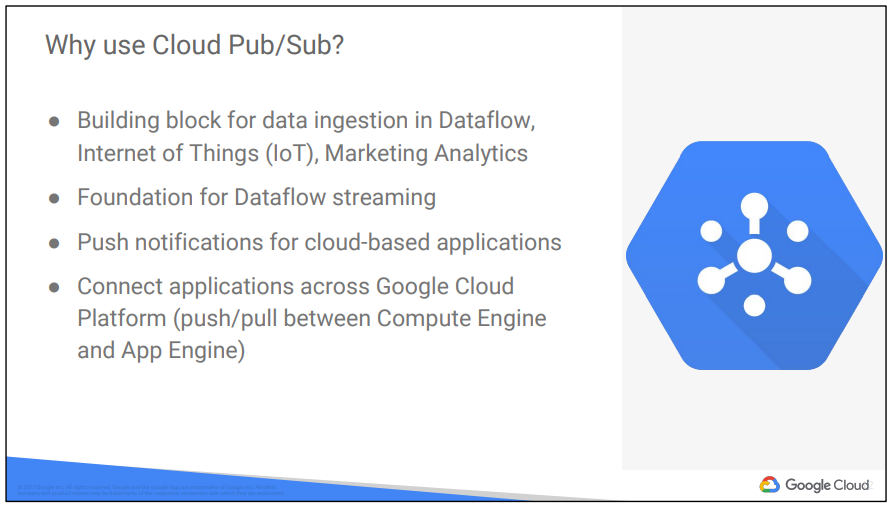
This example Dataflow pipeline reads data from a BigQuery table (the “source”), processes it in various ways (the “transforms”), and writes its output to Cloud Storage (the “sink”).



**BigQuery**

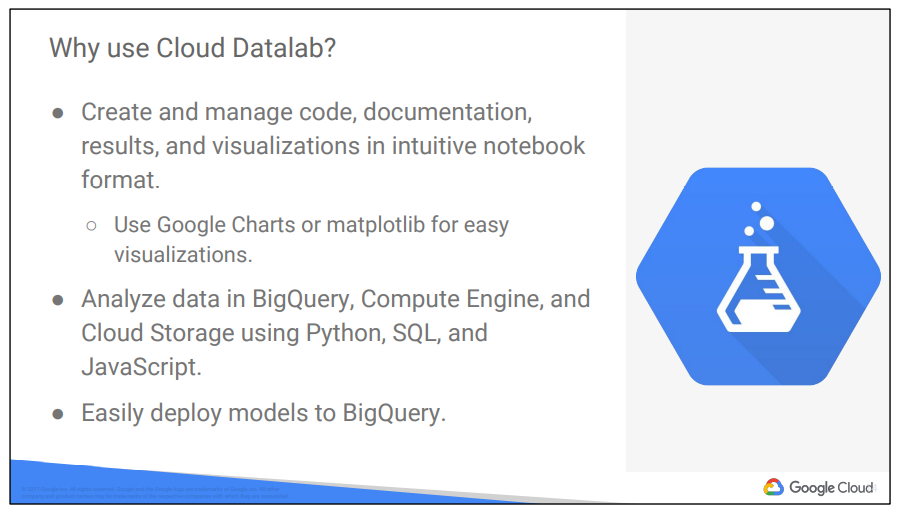
If, instead of a dynamic pipeline, you want to do ad-hoc SQL queries on a massive dataset, that is what BigQuery is for. BigQuery is Google's fully managed, petabyte scale, low cost analytics data warehouse. BigQuery is Google's fully managed, petabyte scale, low cost analytics data warehouse. BigQuery is NoOps: there is no infrastructure to manage and you don't need a database administrator, so you can focus on analyzing data to find meaningful insights, use familiar SQL, and take advantage of our pay-as-you-go model.





**Cloud Datalab**

* Iteractive tool for large scale data interaction
* Integrated, open source
  + Based on Jupyter



**Cloud Machine Learning Platform**

