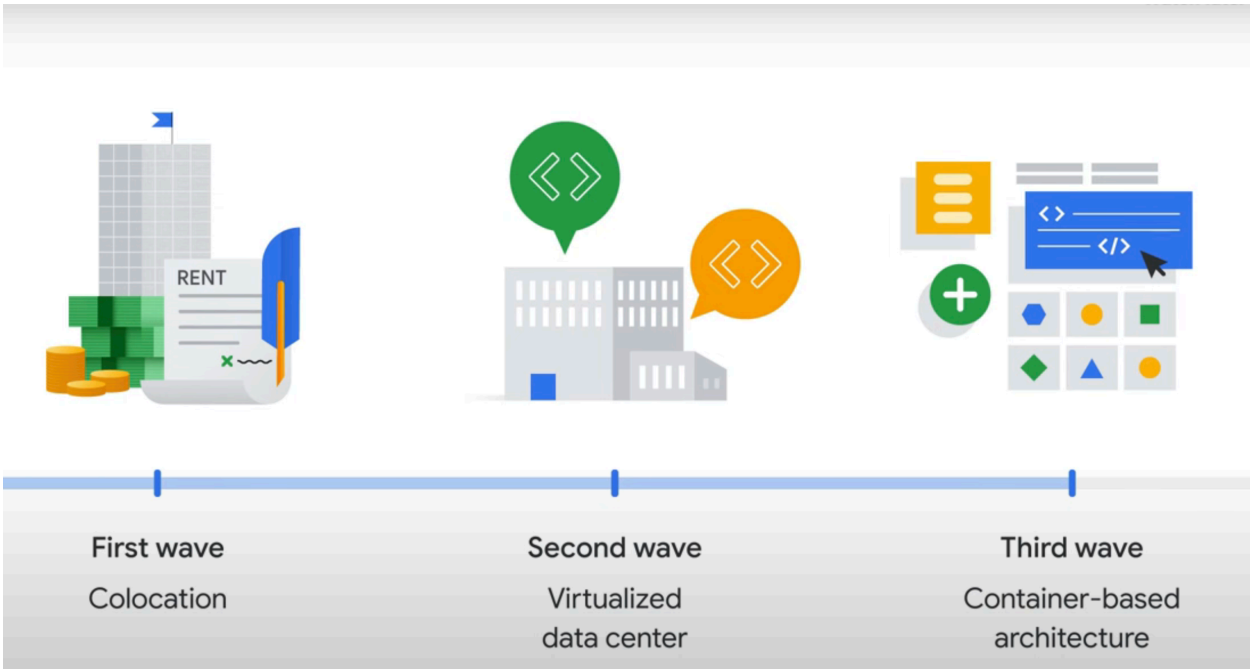
Google Cloud Fundamentals: Core Infrastructure

Course Link: https://partner.cloudskillsboost.google/course_templates/60

Cloud Computing:

- Cloud Computing is a way of using information technology, IT, that has these five equally important traits.
- Customers get computing resources that are on-demand and self-service.
- Customers get access to those resources over the internet from anywhere they have a connection.
- The cloud provider has a big pool of those resources and allocates them to users out of that pool.
- The resources are elastic–which means they're flexible, so customers can be.
- Customers pay only for what they use, or reserve as they go.

The Trend toward Cloud Computing:



IaaS (Infrastructure as a Service):

- IaaS delivers on-demand infrastructure resources via the cloud, such as: Raw Compute, Storage, and Network Capabilities.
- Compute Engine is an example of a Google Cloud IaaS service.
- In the IaaS model, customers pay for the resources they allocate ahead of time.

PaaS (Platform as a Service):

- PaaS binds code to libraries that provide access to the infrastructure application needs. This allows more resources to be focused on application logic.
- App Engine is an example of a Google Cloud PaaS service.
- In the PaaS model, customers pay for the resources they actually use.

Serverless:

- It allows developers to concentrate on their code, rather than on server configuration. No Infrastructure management needed.
- **Cloud Function:** Manages event-driven code as a pay-as-you-go service.
- **Cloud Run:** Allows customers to deploy their containerized microservices based application in a fully-managed environment.

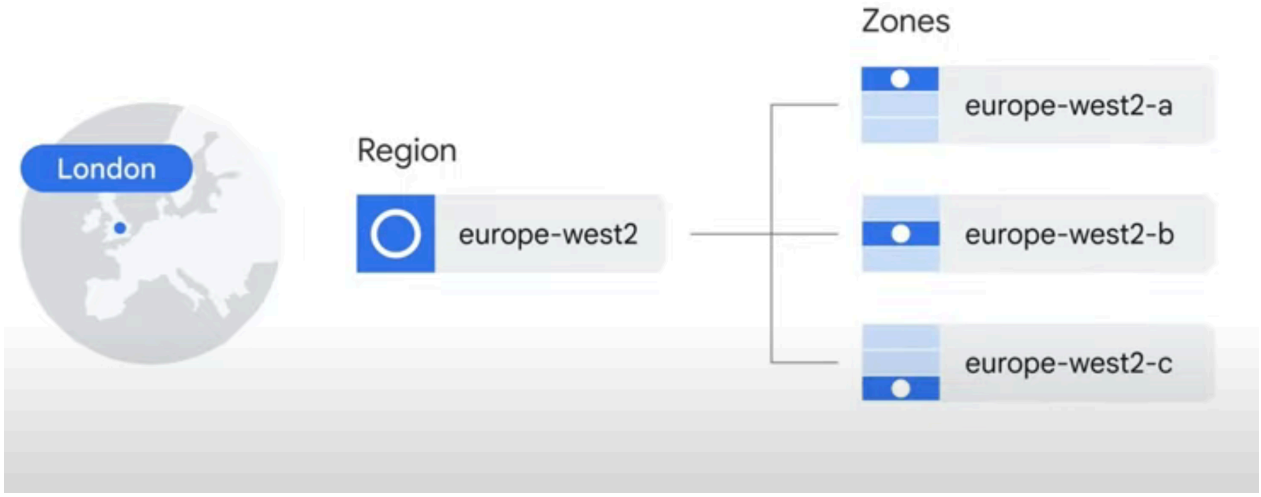
SaaS (Software as a Service):

- SaaS provides the entire application stack, delivering an entire cloud-based application that customers can access and use.
- SaaS applications are not installed on your local computer. Instead, they run in the cloud as a service and are consumed directly over the internet by end users Popular.
- Google applications such as Gmail, Docs, and Drive, that are a part of Google Workspace, are all examples of SaaS.

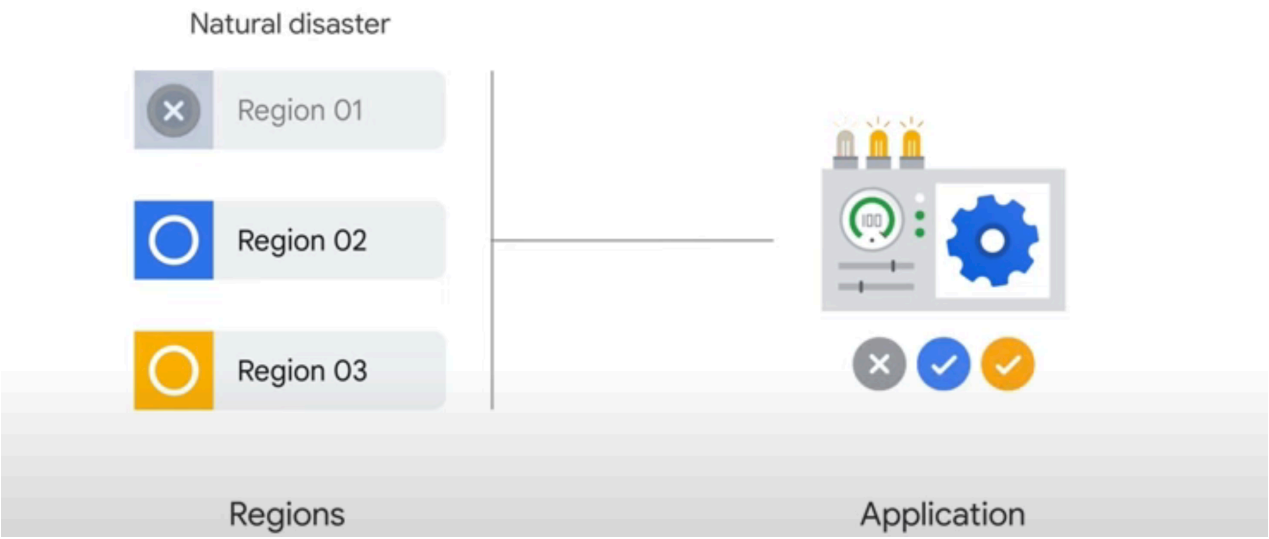
The Google Cloud Network:

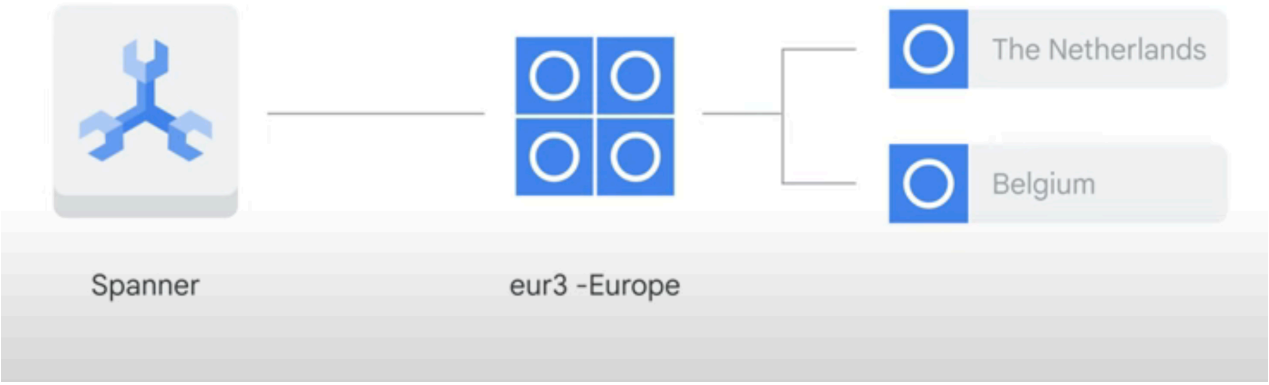
- This network is designed to give customers the highest possible throughput and lowest possible latencies for their applications by leveraging more than 100 content caching nodes worldwide.
- These are locations where high demand content is cached for quicker access, allowing applications to respond to user requests from the location that will provide the quickest response time.
- Google Cloud's infrastructure is based in five major geographic locations: North America, South America, Europe, Asia, and Australia.
- Having multiple service locations is important because choosing where to locate applications affects qualities like availability, durability, and latency.
- Each of these locations is divided into several different regions and zones.
- Regions represent independent geographic areas and are composed of zones. For example, London, or europe-west2, is a region that currently comprises three different zones.
- A zone is an area where Google Cloud resources are deployed. For example, if you launch a virtual machine using Compute Engine it will run in the zone that you specify to ensure resource redundancy.
- You can run resources in different regions. This is useful for bringing applications closer to users around the world, and also for protection in case there are issues with an entire region, say, due to a natural disaster.
- Some of Google Cloud's services support placing resources in what we call a multi-region. For example, Spanner multi-region configurations allow you to replicate the database's data not just in multiple zones, but in multiple zones across multiple regions, as defined by the instance configuration. These additional replicas enable you to read data with low latency from multiple locations close to or within the regions in the configuration, like The Netherlands, and Belgium.


Location
Regions
Zones


Zones
London
Region
europe-west2
europe-west2-a
europe-west2-b
europe-west2-c


europe-west2-a
europe-west2-b
europe-west2-c
Virtual Machine
Zones
Resource redundancy





Google Infrastructure Security:

- **Hardware Infrastructure Layer:**
  - Hardware Design and Provenance
  - Secure Boot Stack (BIOS & Kernel)
  - Premises Security
- **Service Deployment Layer:**
  - Encryption of inter-service communication (RPC)
- **User Identity Layer:**
  - User Identity (Login Form is more about than "Username" & "Password" like "U2F")
- **Storage Services Layer:**
  - Encryption at Rest

- **Internet Communication Layer:**
  - Google Frontend (GFE) (using Private and Public Keys & Certificate from CA)
  - Denial of Services (DoS) Protection
- **Operational Security Layer:**
  - Intrusion Detection
  - Reducing Insider Risk
  - Employee Universal Second Factor (U2F) Use

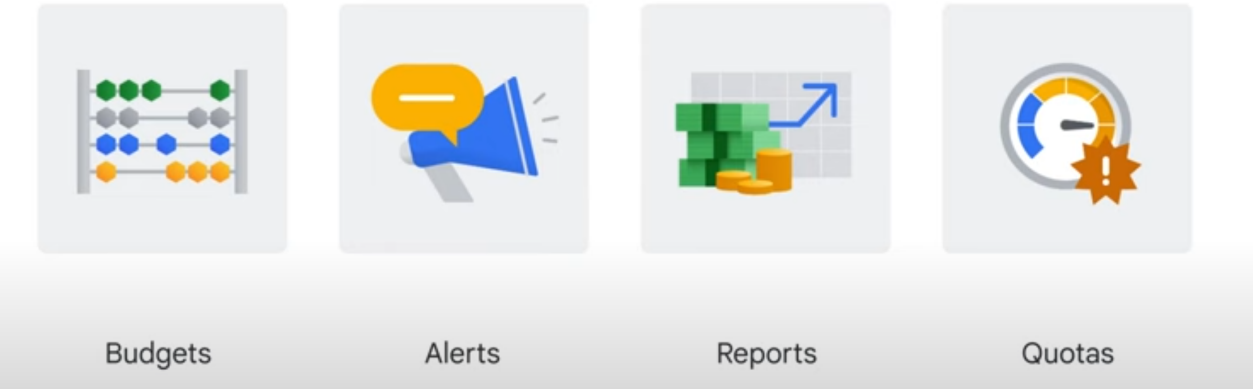
- Software Development Practices (Private Libraries to prevent from Bugs, 2-Party Review for New Code & Vulnerability Rewards Program)

Open Source Ecosystems:

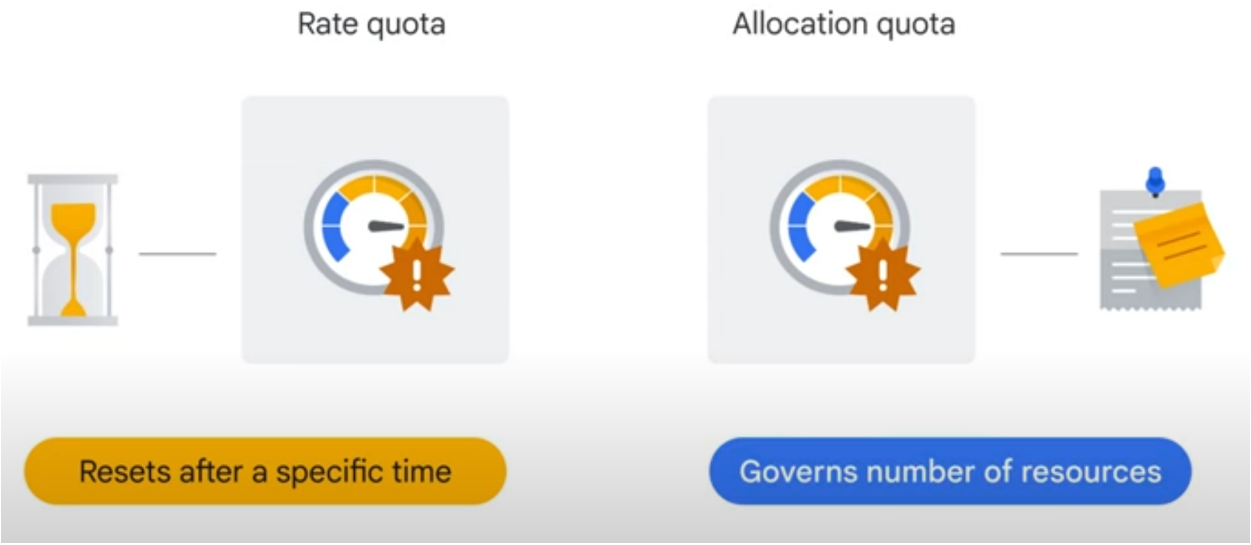
Kubernetes and Google Kubernetes Engine give customers the ability to mix and match microservices running across different clouds, while Google Cloud Observability lets customers monitor workloads across multiple cloud providers.

Pricing and Billing:

- **How can I make sure I don't accidentally run up a big Google Cloud bill ?**

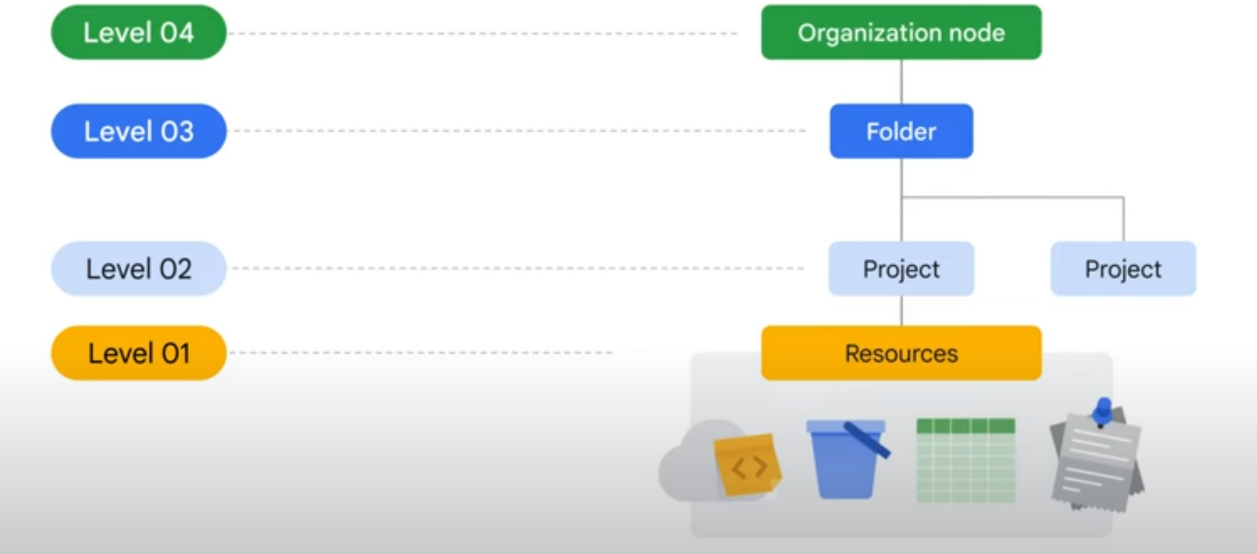


- **Types of Quotes:**



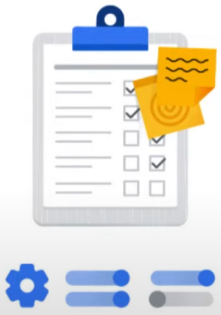
Google Cloud Resource Hierarchy:

- **Resources:** These represent virtual machines, Cloud Storage buckets, tables in BigQuery, or anything else in Google Cloud.
- **Policies:** Can be defined at the project, folder, and organization node levels, they are also inherited downward, this means that if you apply a policy to a folder, it will also apply to all of the projects within that folder.
- **Folders:** Let you assign policies to resources at a level of granularity that you choose. The resources in a folder inherit policies and permissions assigned to that folder. A folder can contain projects, other folders, or a combination of both. You can use folders to group projects under an organization in a hierarchy.
- If you have two different projects that are administered by the same team, you can put policies into a common folder so they have the same permissions. Doing it the other way--putting duplicate copies of those policies on both projects–could be tedious and error-prone. If you needed to change permissions on both resources, you would have to do that in two places instead of just one.

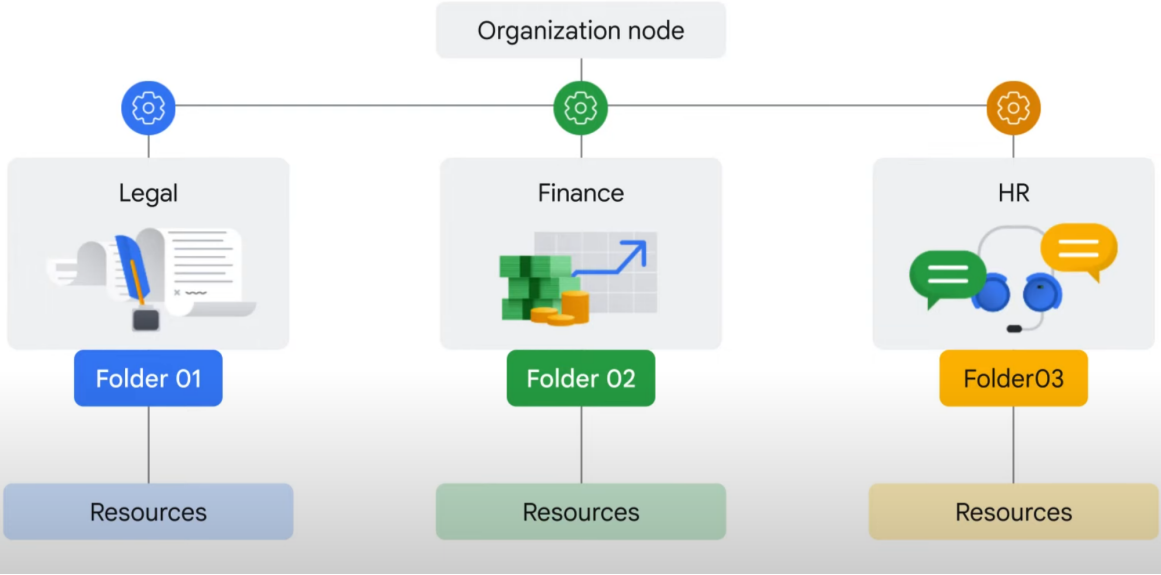


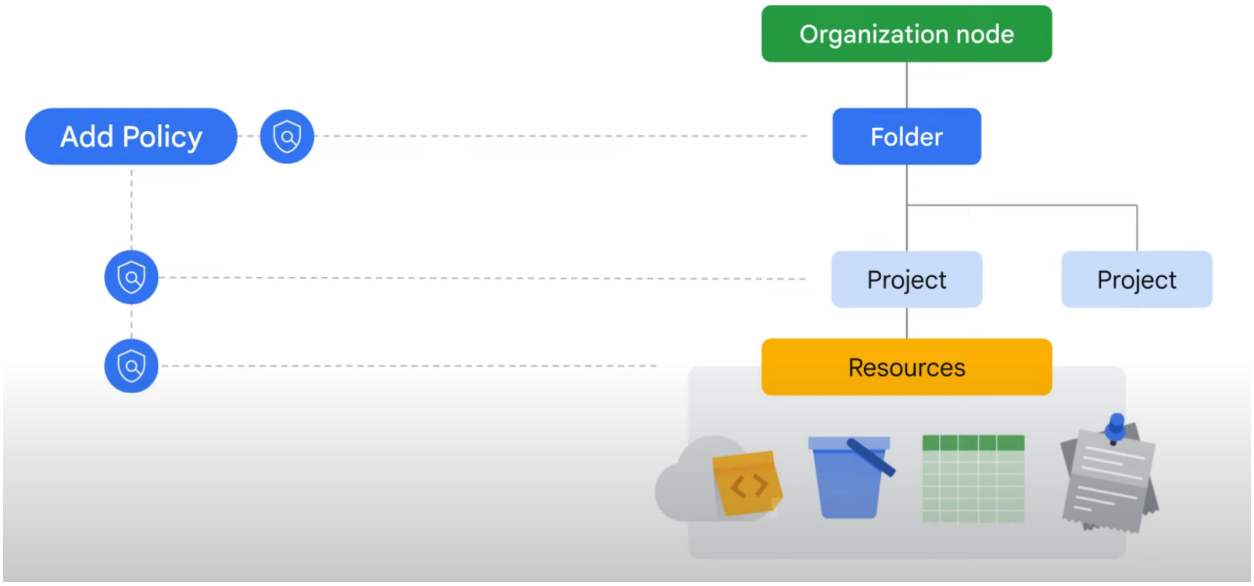
Project ID	Project name	Project number
Globally unique	Need not be unique	Globally unique
Assigned by Google Cloud but mutable during creation	Chosen by you	Assigned by Google Cloud
Immutable after creation	Mutable	Immutable

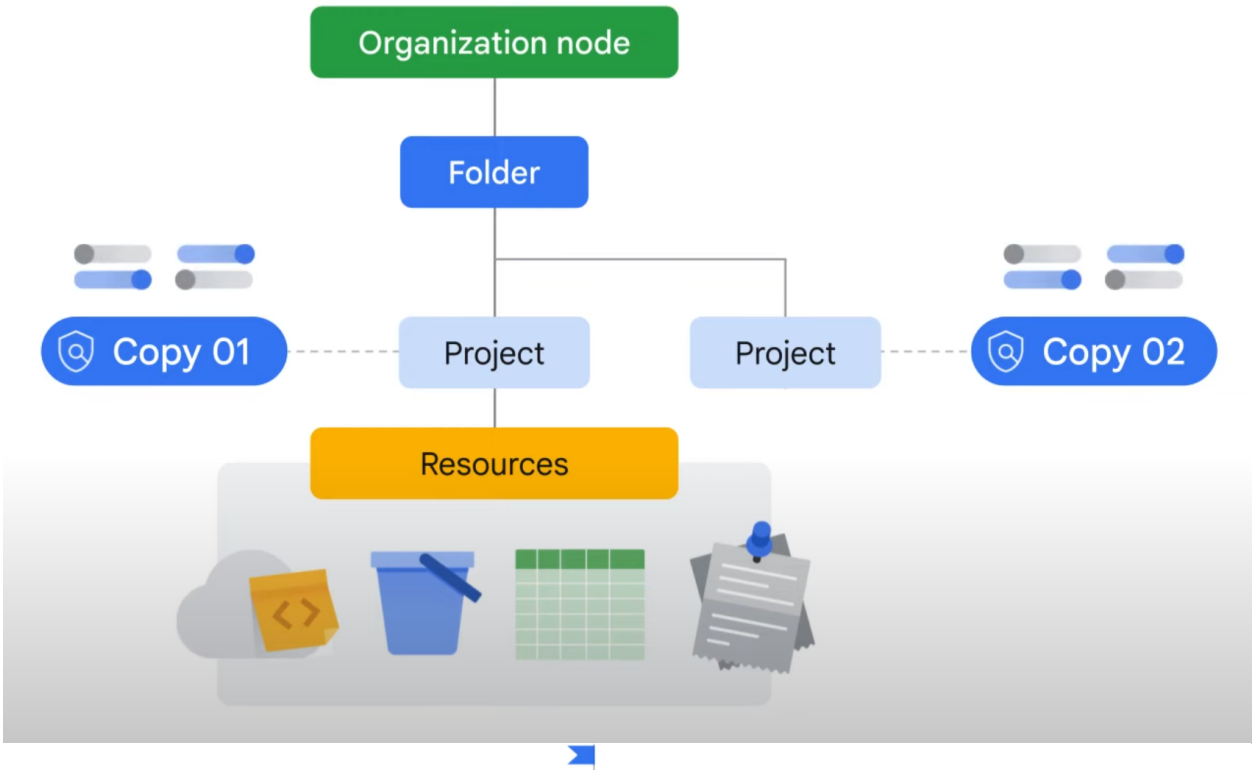
Resource Manager Tool

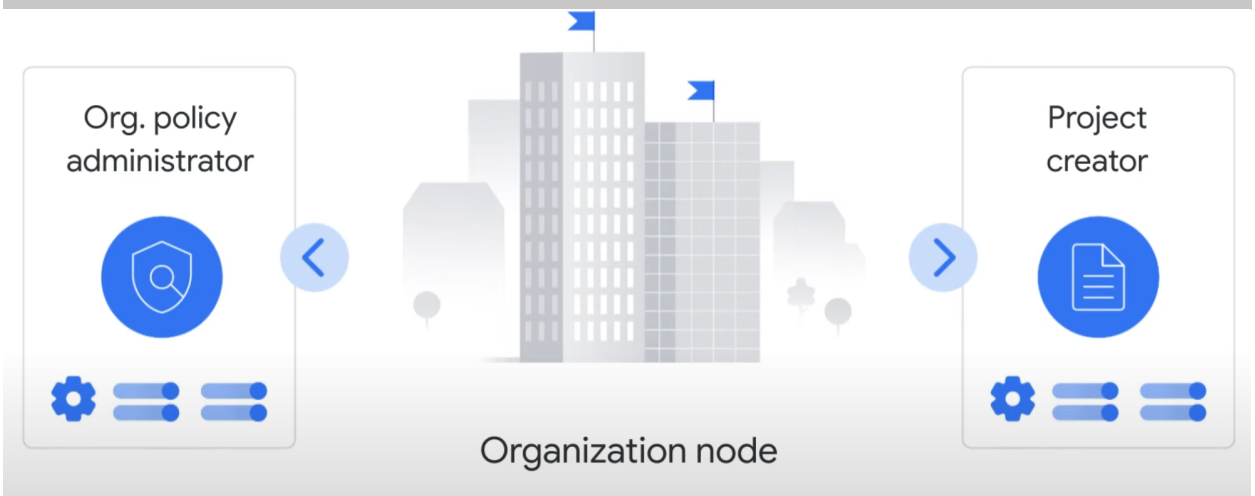


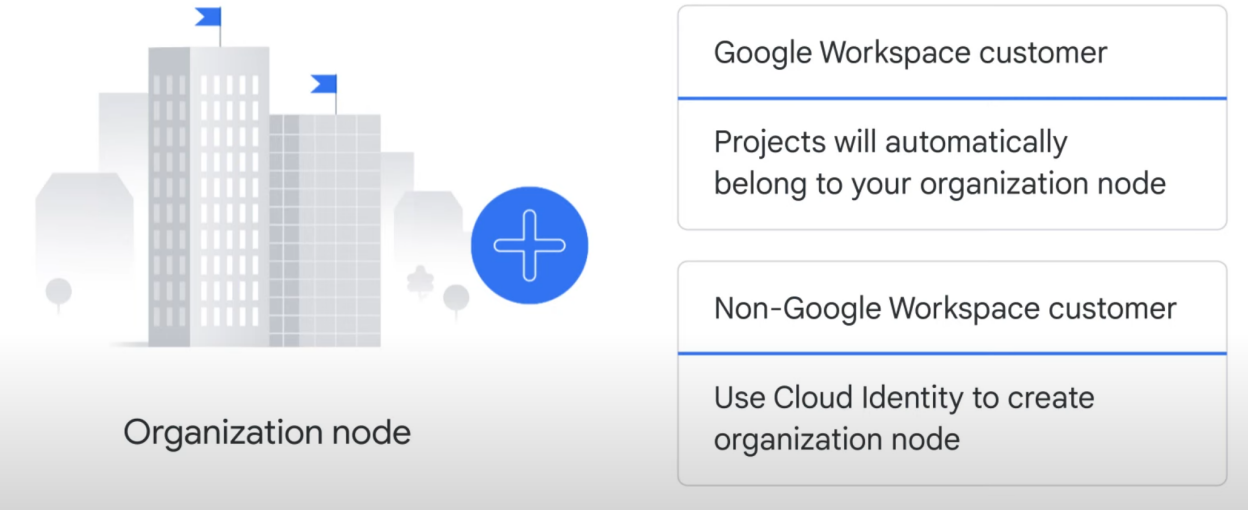
- Gather a list of projects
- Create new projects
- Update existing projects
- Delete projects
- Recover previously deleted projects
- Access through RPC API and REST API




Organization node
Add Policy
Folder
Project
Project
Resources


Organization node
Folder
Copy 01
Project
Project
Copy 02
Resources

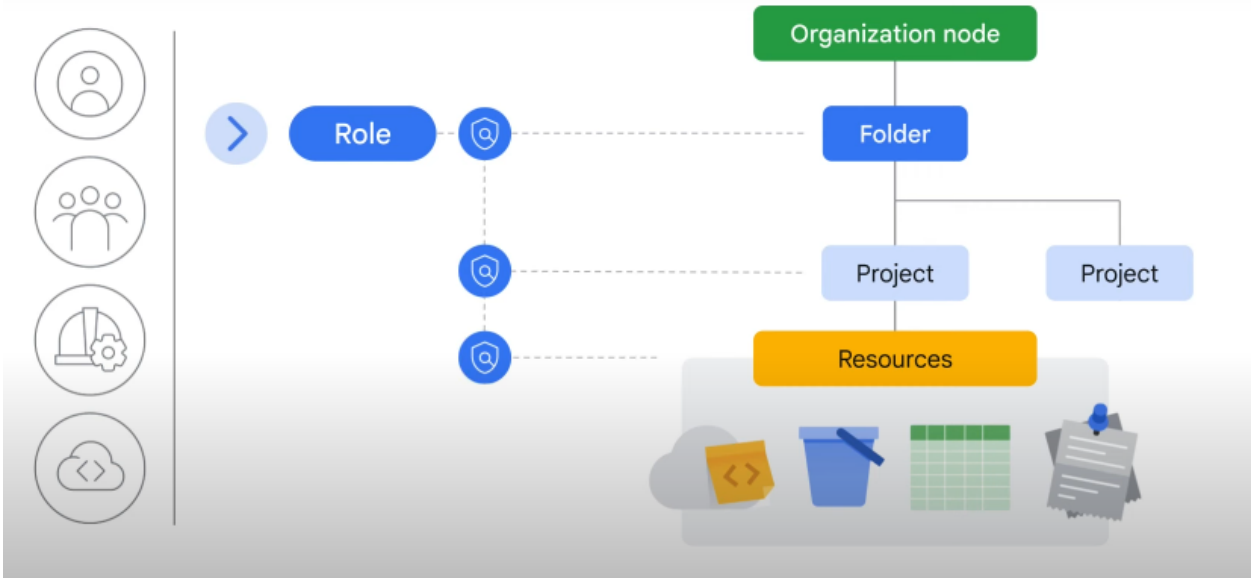

Org. policy administrator
Project creator
Organization node

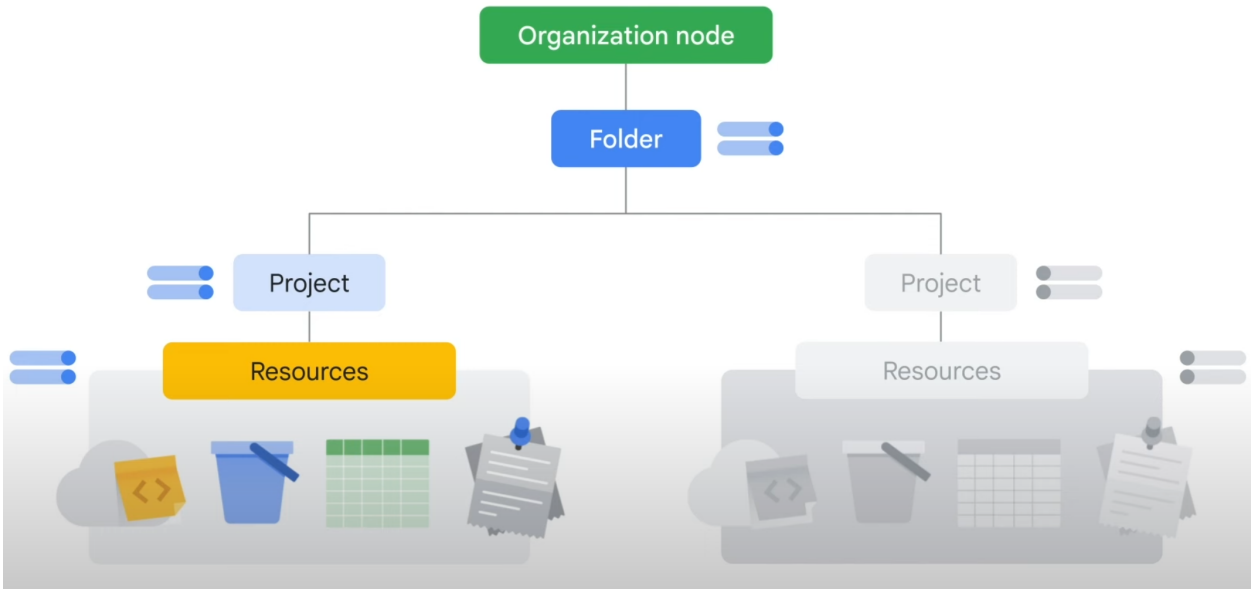


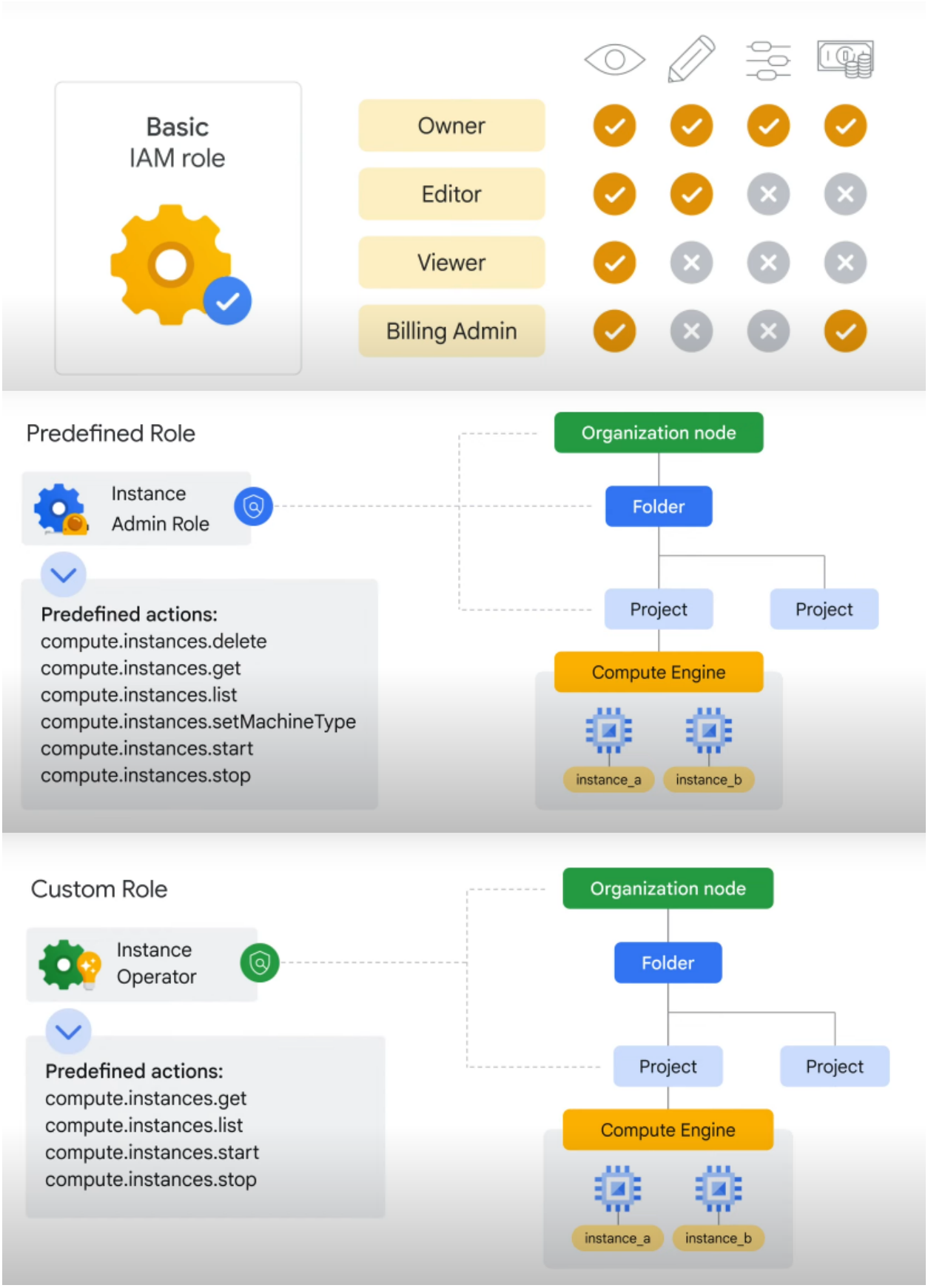
Identity and Access Management (IAM):

- With IAM, administrators can apply policies that define who can do what and on which resources.
- The "**who**" part of an IAM policy can be a Google account, a Google group, a service account, or a Cloud Identity domain. A "who" is also called a "principal". Each principal has its own identifier, usually an email address.
- The "**can do what**" part of an IAM policy is defined by a role. An IAM role is a collection of permissions. When you grant a role to a principal, you grant all the permissions that the role contains. For example, to manage virtual machine instances in a project, you must be able to create, delete, start, stop and change virtual machines.
- When a principal is given a role on a specific element of the resource hierarchy, the resulting policy applies to both the chosen element and all the elements below it in the hierarchy.
- You can define deny rules that prevent certain principals from using certain permissions, regardless of the roles they're granted. This is because IAM always checks relevant deny policies before checking relevant allow policies.
- **Three Kinds of Roles for IAM:**
  - Basic IAM Role
  - Predefined IAM Role
  - Custom IAM Role
- If several people are working together on a project that contains sensitive data, basic roles are probably too broad. So we use at this case **Predefined IAM Role**.
- We use **Custom IAM Role** when we need to assign a role that has even more specific permissions, For example, many companies use a "least-privilege" model in which each person in your organization is given the minimal amount of privilege needed to do their job. So, for example, maybe you want to define an "instanceOperator" role to allow some users to stop and start Compute Engine virtual machines, but not reconfigure them.
- In **Custom IAM Role,** you'll need to manage the permissions that define the custom role you've created, also it can only be applied to either the project level or organization level.


Project
Role

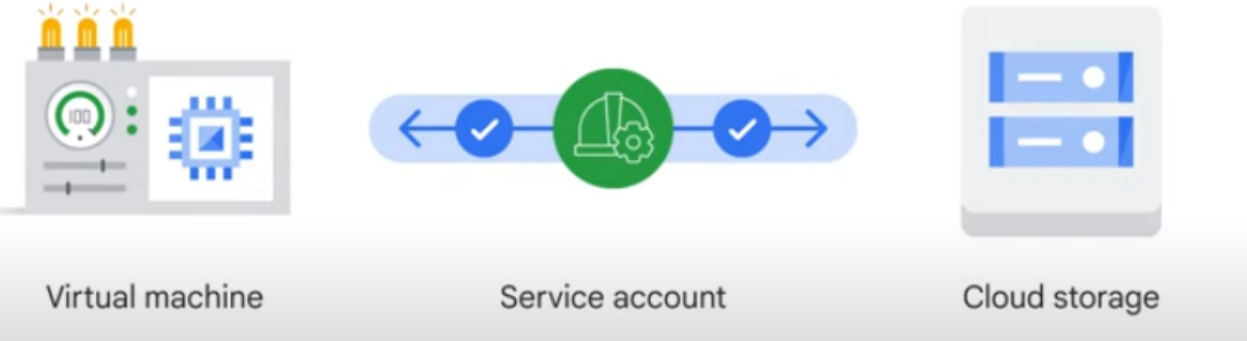

Organization node
Role
Folder
Project
Project
Resources


Organization node
Folder
Project
Project
Resources
Resources


Basic
IAM role
Owner
Editor
Viewer
Billing Admin
Predefined Role
Instance
Admin Role
Predefined actions:
compute.instances.delete
compute.instances.get
compute.instances.list
compute.instances.setMachineType
compute.instances.start
compute.instances.stop
Organization node
Folder
Project
Project
Compute Engine
instance_a
instance_b
Custom Role
Instance
Operator
Predefined actions:
compute.instances.get
compute.instances.list
compute.instances.start
compute.instances.stop
Organization node
Folder
Project
Project
Compute Engine
instance_a
instance_b

Service Accounts:

- What if you want to give permissions to a Compute Engine virtual machine, rather than to a person?, that's what service accounts are for.
- If you have an application running in a virtual machine that needs to store data in Cloud Storage, but you don't want anyone on the internet to have access to that data–just that particular virtual machine. You can create a service account to authenticate that VM to Cloud Storage.
- Service Accounts are named with an email address, but instead of passwords they use cryptographic keys to access resources.



Cloud Identity:

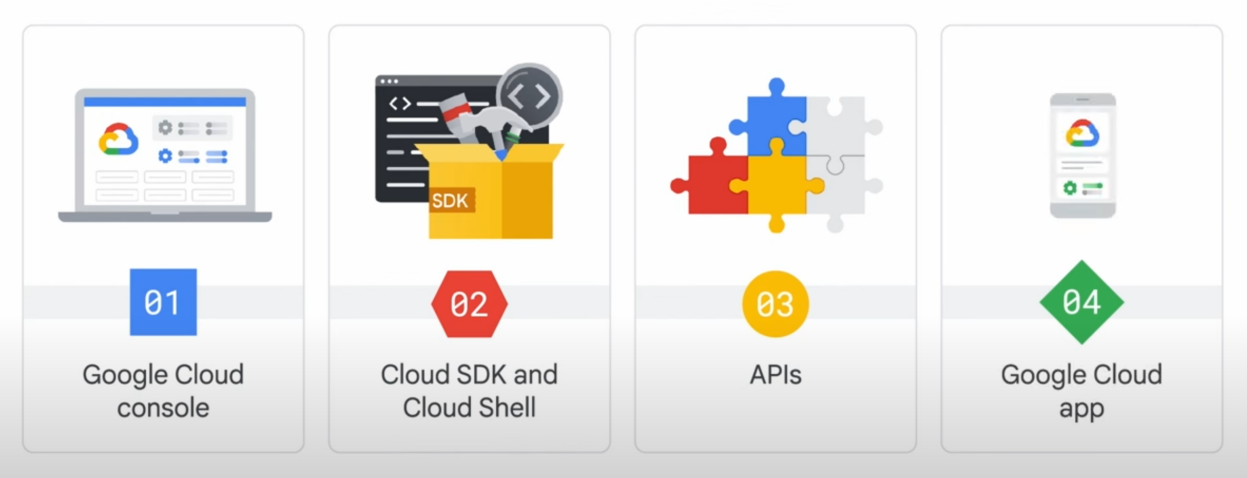
- When new Google Cloud customers start using the platform, it's common to log in to the Google Cloud Console with a Gmail account and then use Google Groups to collaborate with teammates who are in similar roles. Although this approach is easy to start with, it can present challenges later because the team's identities are not centrally managed. This can be problematic if, for example, someone leaves the organization. With this setup, there's no easy way to immediately remove a user's access to the team's cloud resources.
- By using Cloud Identity, organizations can define policies and manage their users and groups using the Google Admin Console and the administrator can use the Google Admin Console to disable their account and remove them from groups.

Interacting with Google Cloud:

- **Google Cloud Console:** Which is Google Cloud's graphical user interface, or GUI, that helps you deploy, scale, and diagnose production issues in a simple web-based interface. You can easily find your resources, check their health, have full management control over them, and set budgets to control how much you spend on them.
- **Cloud SDK:** Is a set of tools that you can use to manage resources and applications hosted on Google Cloud. These include the **Google Cloud CLI**, a command-line interface for Google Cloud products and services, and **bq**, a command-line tool for BigQuery.
- **APIs:** The Cloud Console includes a tool called the Google APIs Explorer that shows which APIs are available, and in which versions. You can try these APIs interactively, even those that require user authentication. Google provides Cloud Client libraries and Google API Client libraries in many popular languages to take a lot of the drudgery out of the task of

calling Google Cloud from your code. Languages currently represented in these libraries are Java, Python, PHP, C#, Go, Node.js, Ruby, and C++.

- **Google Cloud App:** Used to start, stop, and use SSH to connect to Compute Engine instances and see logs from each instance. It also lets you stop and start Cloud SQL instances. You can administer applications deployed on App Engine by viewing errors, rolling back deployments, and changing traffic splitting. It provides up-to-date billing information for your projects and billing alerts for projects that are going over budget. You can set up customizable graphs showing key metrics such as CPU usage, network usage, requests per second, and server errors.



Compute Engine:

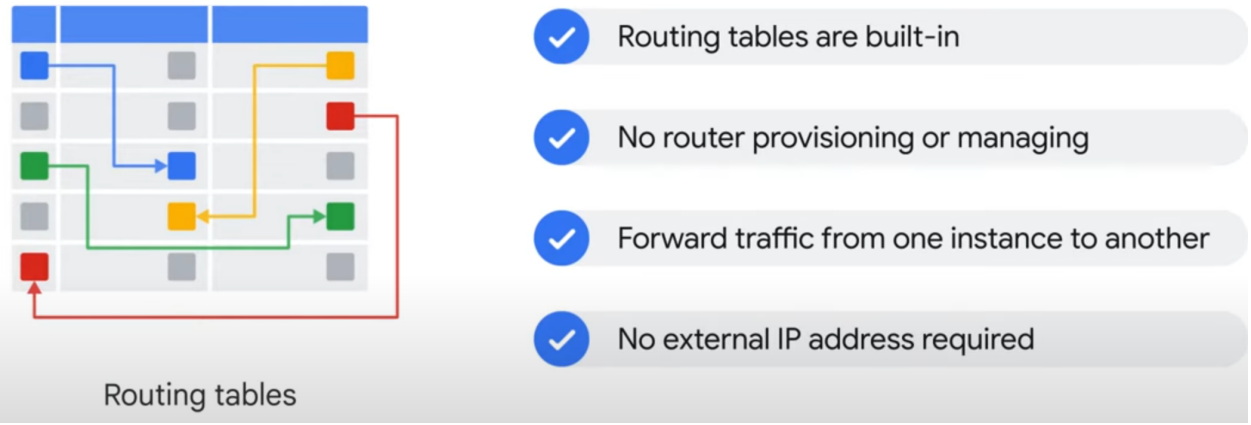
- Can create and run virtual machines on Google infrastructure.
- No upfront investments, and thousands of virtual CPUs can run on a system that's designed to be fast and to offer consistent performance.
- Each virtual machine contains the power and functionality of a full-fledged operating system, this means a virtual machine can be configured much like a physical server.

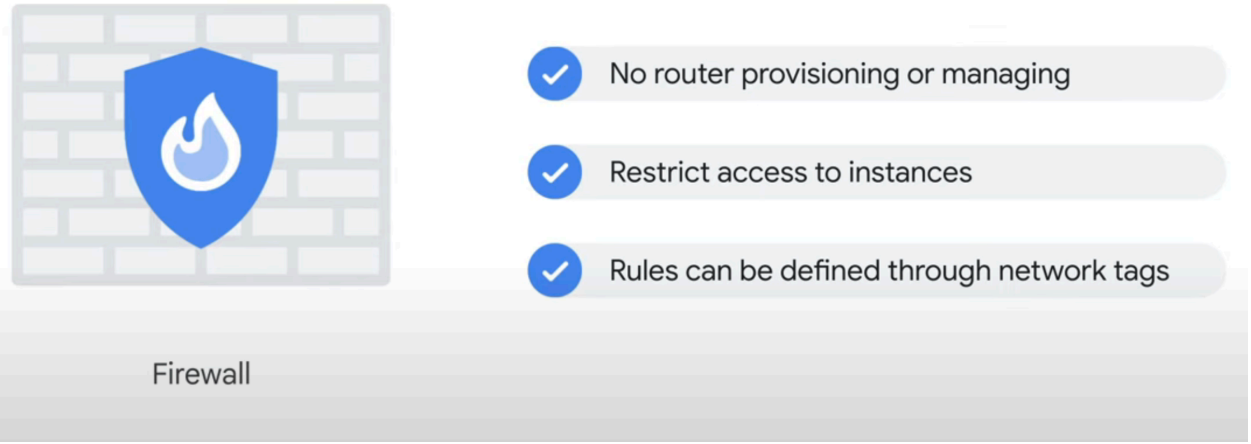
Virtual Machine:

- Can be created via the Google Cloud console, the Google Cloud CLI, or the Compute Engine API.
- Can run Linux and Windows Server images provided by Google or any customized versions of these images.
- Can build and run images of other operating systems and flexibly reconfigure virtual machines.

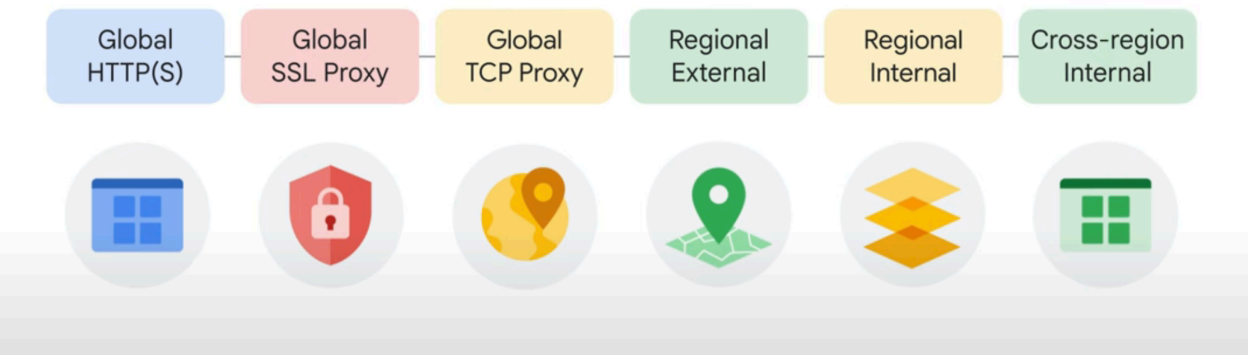
VPC Compatibility Features:

- Routing Tables
- Firewall
- VPC Peering
- Shared VPC



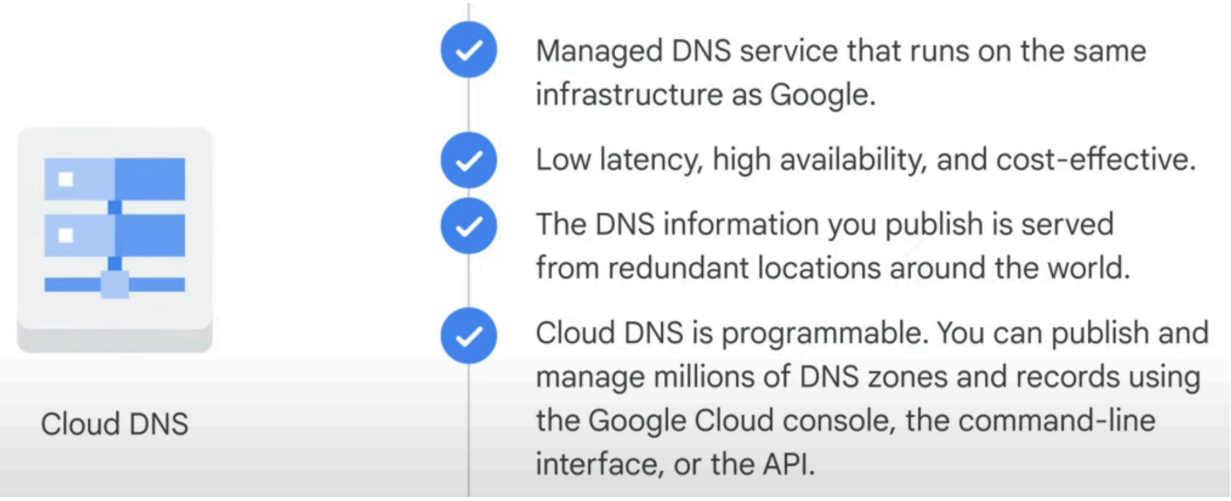


Cloud Load Balancers:



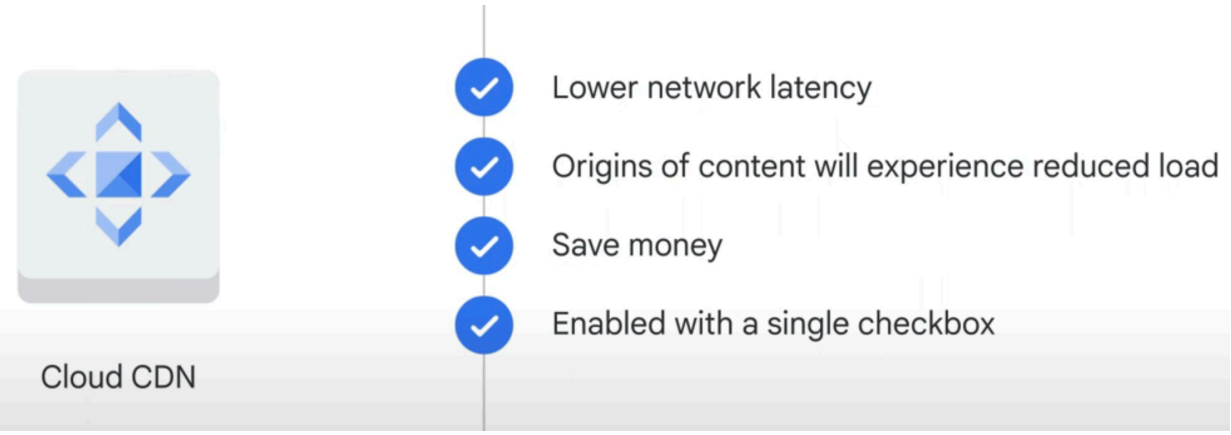
Cloud DNS:

DNS is what translates internet hostnames to addresses, and as you might imagine, Google has a highly developed DNS infrastructure.

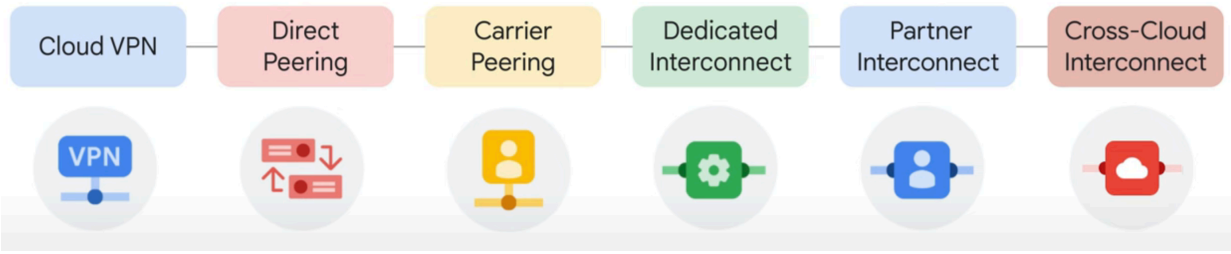


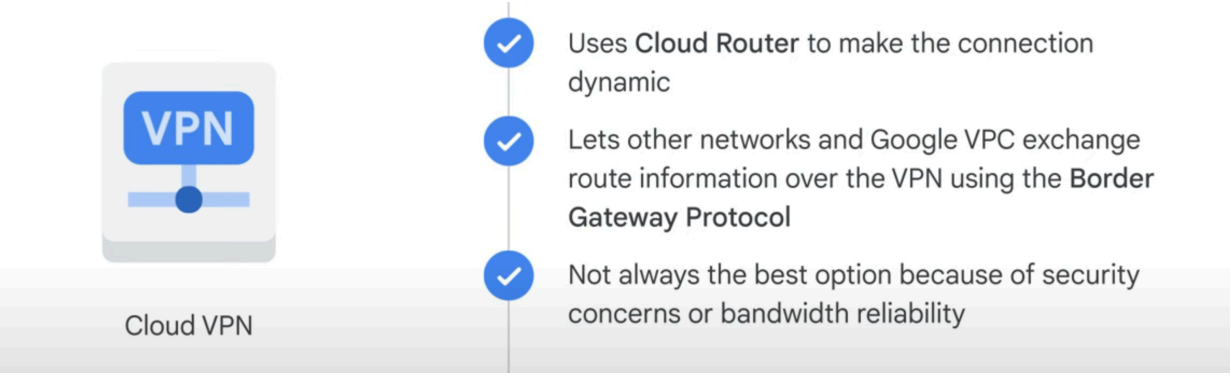
Cloud CDN:

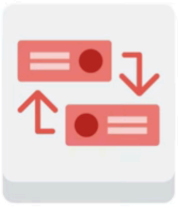
You can use CDN to accelerate content delivery in your application.



Connection Networks to Google VPC:







Direct Peering

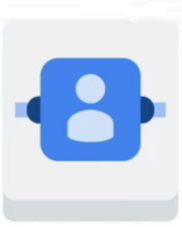
- Puts a router in the same public datacenter as a Google point of presence (PoP)
- Uses a router to exchange traffic between networks
- More than 100 Google points of presence around the world

Carrier Peering

- Gives direct access from an on-premises network through a service provider's network
- Not covered by a Google Service Level Agreement

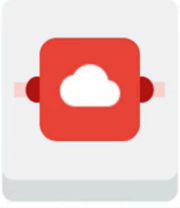
Dedicated Interconnect

- Allows for one or more direct, private connections to Google
- Can be covered by up to a 99.99% SLA
- Connections can be backed up by a VPN



Partner Interconnect

- Useful if a data center is in a physical location that can't reach a Dedicated Interconnect colocation facility
- Useful if the data needs don’t warrant an entire 10 GigaBytes per second connection
- Can be configured to support mission-critical services or applications that can tolerate some downtime
- Can be covered by up to a 99.99% SLA



Cross-Cloud Interconnect

- Establish high-bandwidth dedicated connectivity between Google Cloud and another cloud service provider
- Supports the adoption of an integrated multicloud strategy
- Two connection sizes: 10 Gbps or 100 Gbps

Google Cloud Storage Options:

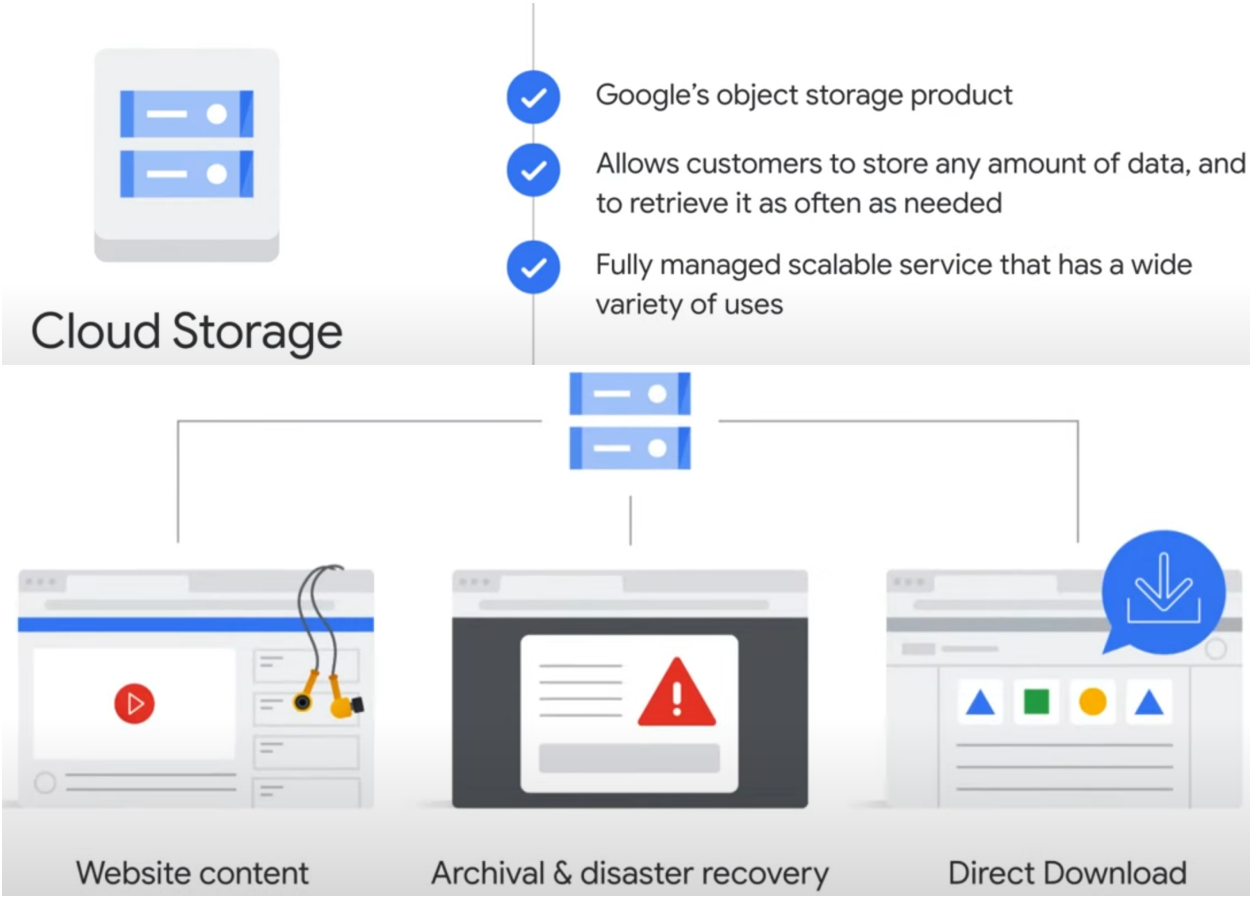
Google Cloud has storage options for structured, unstructured, transactional, and relational data.

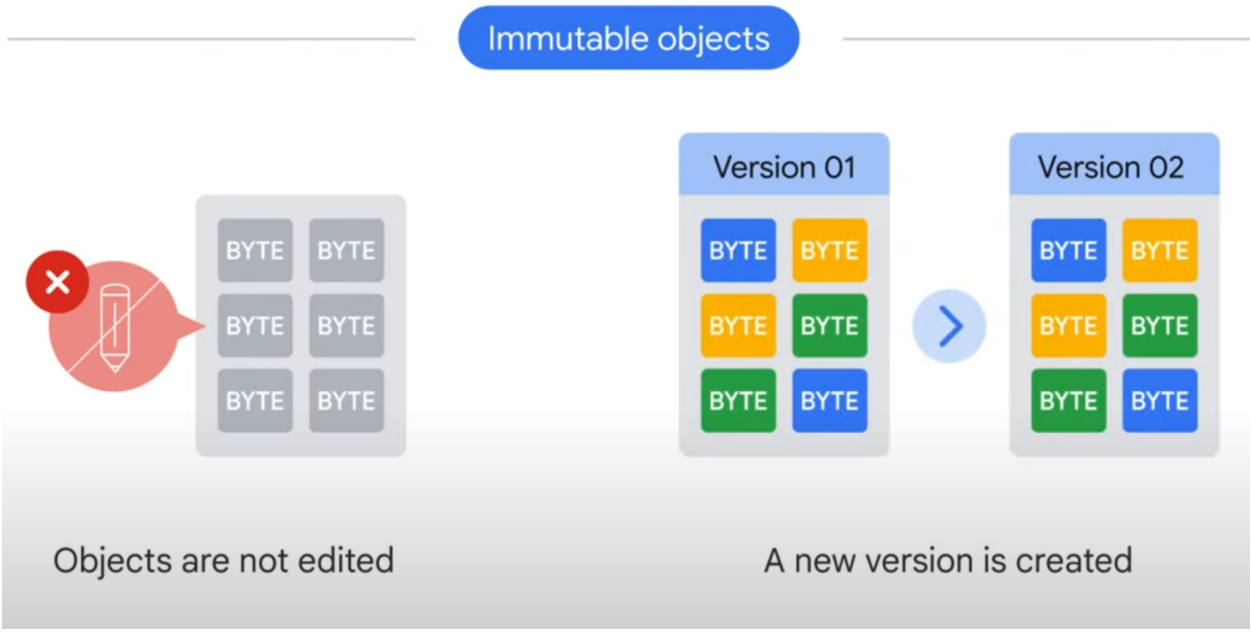
- Cloud Storage
- Cloud SQL
- Spanner
- Firestore
- Bigtable

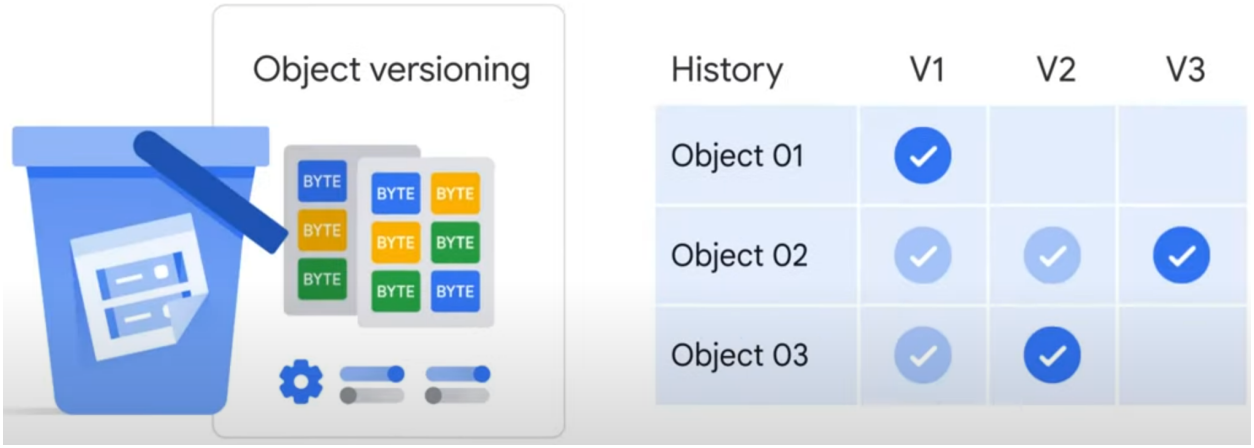
Cloud Storage:

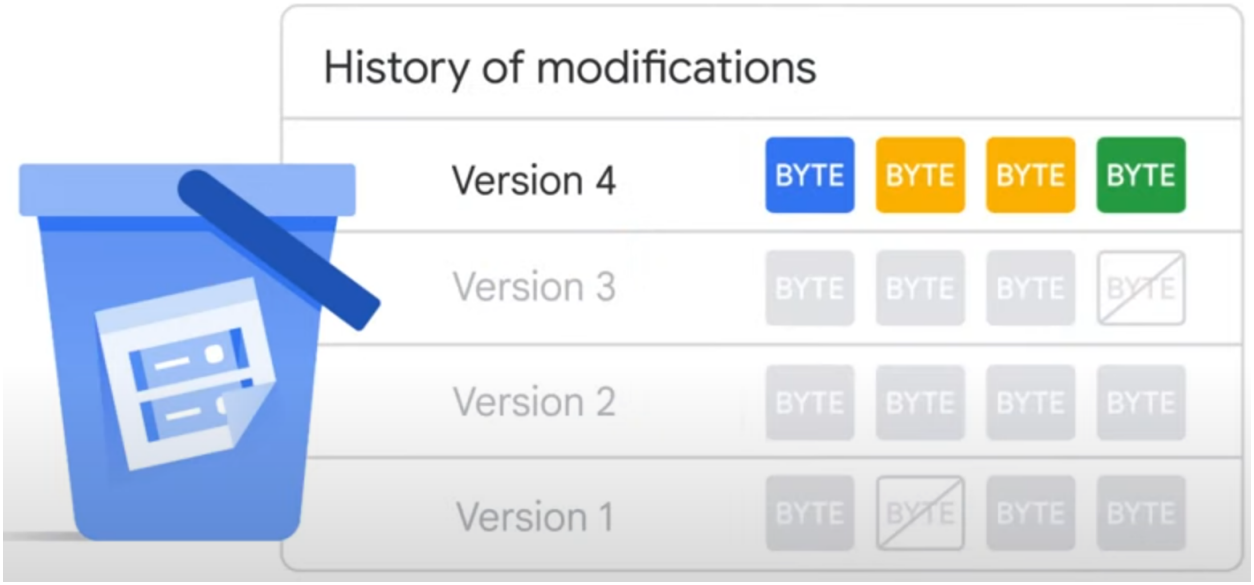
- Cloud Storage is a service that offers developers and IT organizations durable and highly available object storage. It is Google's object storage product.
- Object storage is a computer data storage architecture that manages data as “objects” and not as a file and folder hierarchy (file storage), or as chunks of a disk (block storage).
- These objects are stored in a packaged format which contains the binary form of the actual data itself, as well as relevant associated meta-data (such as date created, author, resource type, and permissions), and a globally unique identifier. Data commonly stored as objects include video, pictures, and audio recordings.
- A few examples of Cloud Storage include serving website content, storing data for archival and disaster recovery, and distributing large data objects to end users via Direct Download.
- Cloud Storage’s primary use is whenever binary large-object storage (also known as a “BLOB”) is needed for online content such as videos and photos, for backup and archived data and for storage of intermediate results in processing workflows.
- Cloud Storage files are organized into buckets. A bucket needs a globally unique name and a specific geographic location for where it should be stored, and an ideal location for a bucket is where latency is minimized. For example, if most of your users are in Europe, you probably want to pick a European location, so either a specific Google Cloud region in Europe, or else the EU multi-region.
- The storage objects offered by Cloud Storage are immutable, which means that you do not edit them, but instead a new version is created with every change made. Administrators have the option to either allow each new version to completely overwrite the older one, or to keep track of each change made to a particular object by enabling “versioning” within a bucket. If you choose to use versioning, Cloud Storage will keep a detailed history of modifications -- that is, overwrites or deletes -- of all objects contained in that bucket. If you don’t turn on object versioning, by default new versions will always overwrite older versions. With object versioning enabled, you can list the archived versions of an object, restore an object to an older state, or permanently delete a version of an object, as needed.
- Using IAM roles and, where needed, access control lists (ACLs), organizations can conform to security best practices, which require each user to have access and permissions to only the resources they need to do their jobs, and no more than that. There are a couple of options to control user access to objects and buckets. For most purposes, IAM is sufficient. Roles are inherited from project to bucket to object. If you need finer control, you can create access control lists. Each access control list consists of two pieces of information. The first is a scope, which defines who can access and perform an action. This can be a specific user or group of users. The second is a permission, which defines what actions can be performed, like read or write.
- Because storing and retrieving large amounts of object data can quickly become expensive, Cloud Storage also offers lifecycle management policies. For example, you could tell Cloud Storage to delete objects older than 365 days; or to delete objects created before January 1,

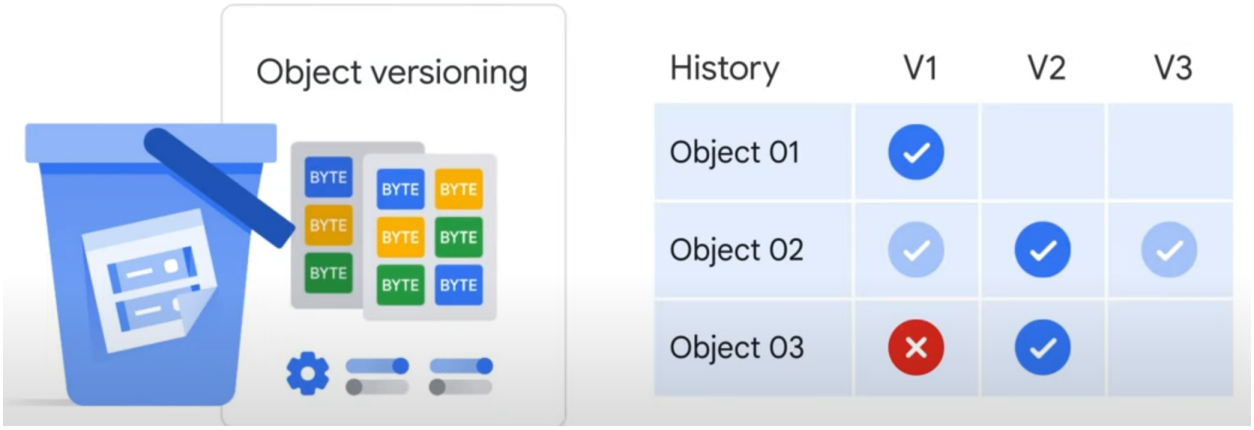
2013; or to keep only the 3 most recent versions of each object in a bucket that has versioning enabled. Having this control ensures that you're not paying for more than you actually need.









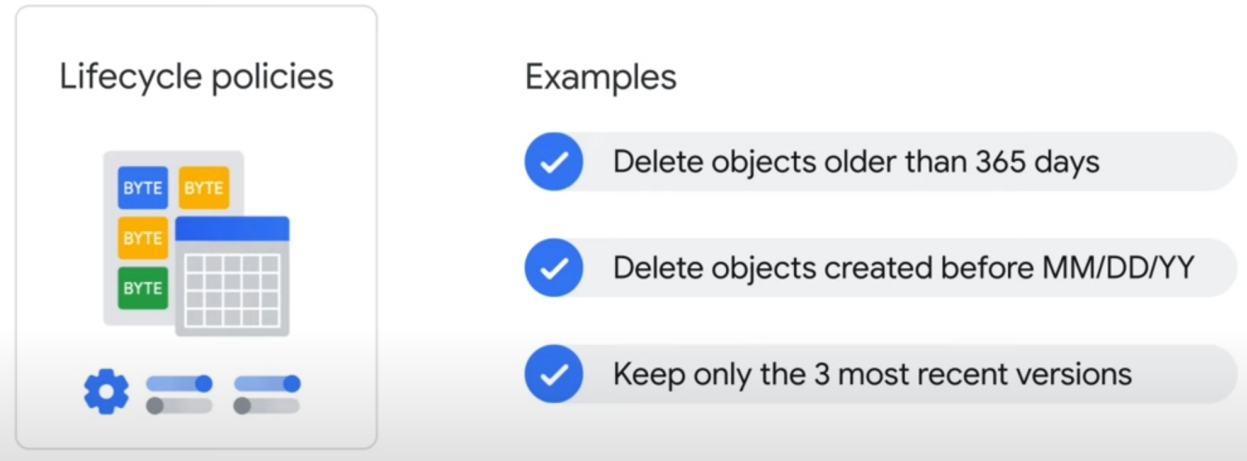


01 For most purposes, Cloud IAM is sufficient

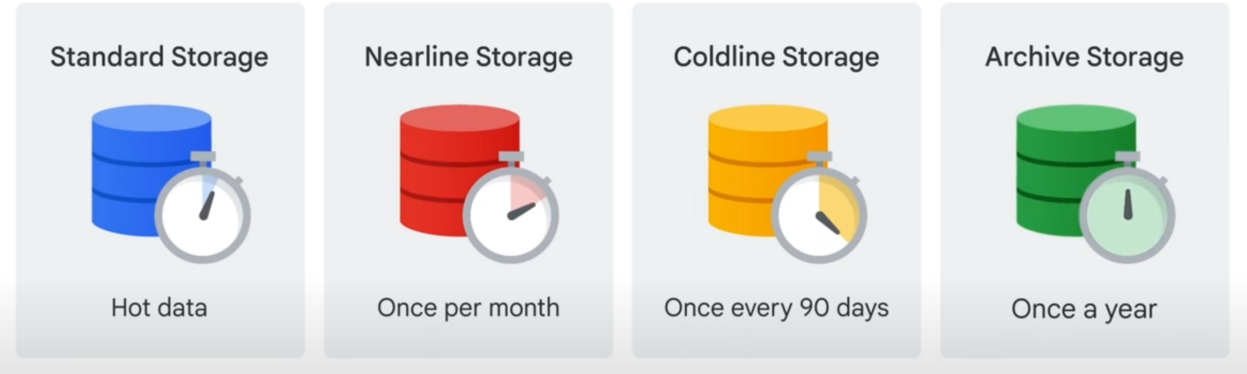
Inherited from project to bucket to object

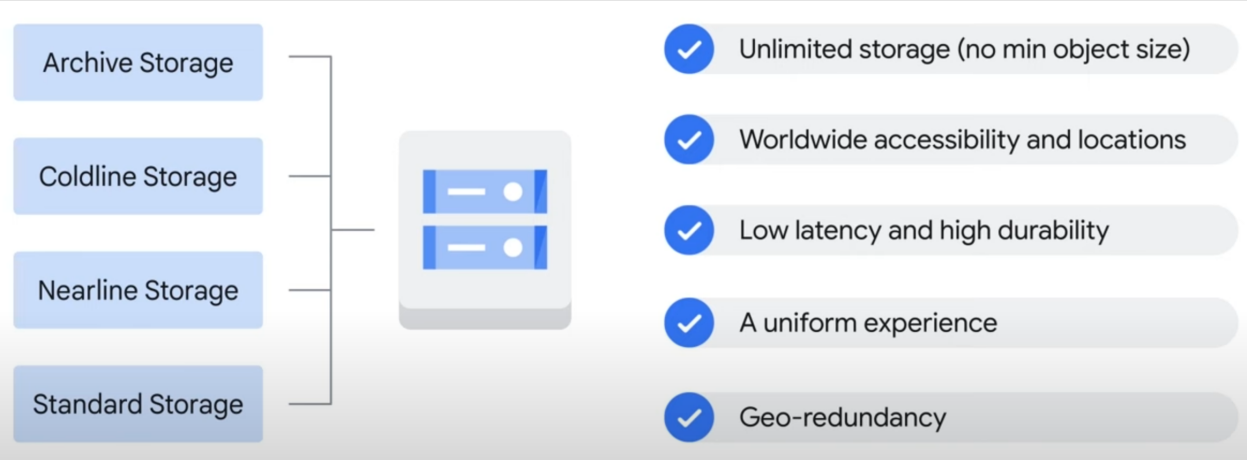
02 If you need finer control, you can create access control lists (ACLs)

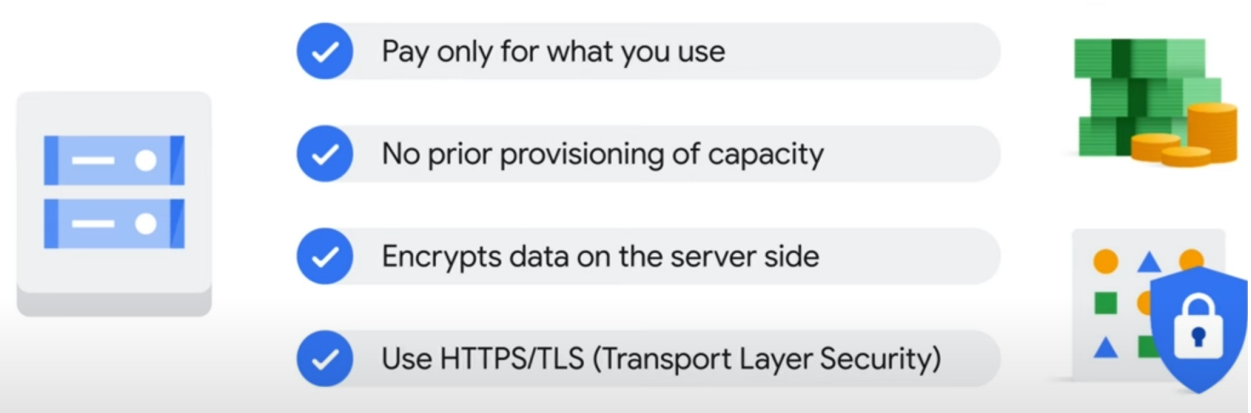
Scope + Permission



Cloud Storage: Storage Classes:

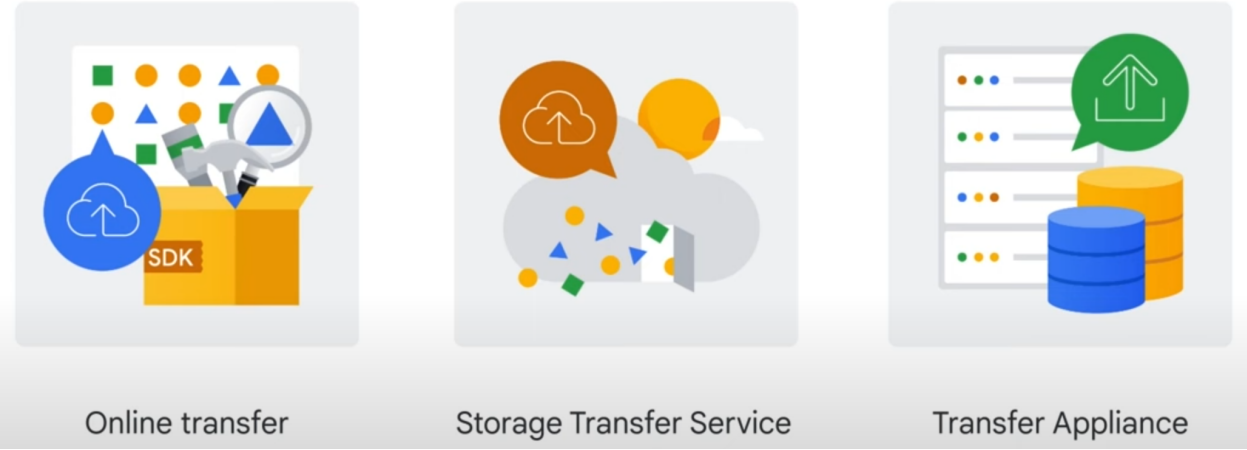






Cloud Storage: Data Transfer:

Regardless of which storage class you choose, there are several ways to bring data into Cloud Storage. Many customers simply carry out their own online transfer using gcloud storage, which is the Cloud Storage command from the Cloud SDK. Data can also be moved in by using a drag and drop option in the Cloud Console, if accessed through the Google Chrome web browser. There is also Storage Transfer Service that enables you to import large amounts of online data into Cloud Storage quickly and cost-effectively. 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. And then there is the Transfer Appliance, which is a rackable, high-capacity storage server that you lease from Google Cloud. You 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. You can transfer up to a petabyte of data on a single appliance.



Cloud SQL:

- Cloud SQL offers fully managed relational databases, including MySQL, PostgreSQL, and SQL Server as a service. It's designed to hand off mundane, but necessary and often time-consuming, tasks to Google—like applying patches and updates managing backups, and configuring replications.
- Cloud SQL doesn't require any software installation or maintenance.
- It can scale up to 128 processor cores, 864 GB of RAM, and 64 TB of storage.
- It supports automatic replication scenarios, such as from a Cloud SQL primary instance, an external primary instance, and external MySQL instances.
- Cloud SQL supports managed backups, so backed-up data is securely stored and accessible if a restore is required. The cost of an instance covers seven backups.

- Cloud SQL encrypts customer data when on Google's internal networks and when stored in database tables, temporary files, and backups.
- It includes a network firewall, which controls network access to each database instance.

Spanner:

Spanner is a fully managed relational database service that scales horizontally, is strongly consistent, and speaks SQL. Spanner is especially suited for applications that require a SQL relational database management system with joins and secondary indexes, built-in high availability, strong global consistency, and high numbers of input and output operations per second.

Firestore:

- Firestore is a flexible, horizontally scalable, NoSQL cloud database for mobile, web, and server development. With Firestore, data is stored in documents and then organized into collections. Documents can contain complex nested objects in addition to subcollections. Each document contains a set of key-value pairs. For example, a document to represent a user has the keys for the firstname and lastname with the associated values.
- Firestore's NoSQL queries can then be used to retrieve individual, specific documents or to retrieve all the documents in a collection that match your query parameters.
- Queries can include multiple, chained filters and combine filtering and sorting options. They're also indexed by default, so query performance is proportional to the size of the result set, not the dataset.
- Firestore uses data synchronization to update data on any connected device. However, it's also designed to make simple, one-time fetch queries efficiently. It caches data that an app is actively using, so the app can write, read, listen to, and query data even if the device is offline. When the device comes back online, Firestore synchronizes any local changes back to Firestore.

Bigtable:

- 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.
- Bigtable is designed to handle massive workloads at consistent low latency and high throughput, so it's a great choice for both operational and analytical applications, including Internet of Things, user analytics, and financial data analysis.
- **When deciding which storage option is best, customers often choose Bigtable if:**
    - They're working with more than 1TB of semi-structured or structured data.
    - Data is fast with high throughput, or it's rapidly changing.
    - They're working with NoSQL data.
    - Data is a time-series or has natural semantic ordering.
    - They're working with big data, running asynchronous batch or synchronous real-time processing on the data.
    - They're running machine learning algorithms on the data.

Comparing Storage Options:

Option	Best for	Capacity
Cloud Storage	Storing immutable blobs larger than 10 MB	Petabytes Max unit size: 5 TB per object
Cloud SQL	• Full SQL support for an online transaction processing system • Web frameworks and existing applications	Up to 64 TB
Spanner	• Full SQL support for an online transaction processing system • Horizontal scalability	Petabytes
Firestore	Massive scaling and predictability together with real time query results and offline query support	Terabytes Max unit size: 1 MB per entity
Bigtable	• Storing large amount of structured objects • Does not support SQL queries and multi-row transactions • Analytical data with heavy read and write events	Petabytes Max unit size: 10 MB p/cell, 100 MB p/row

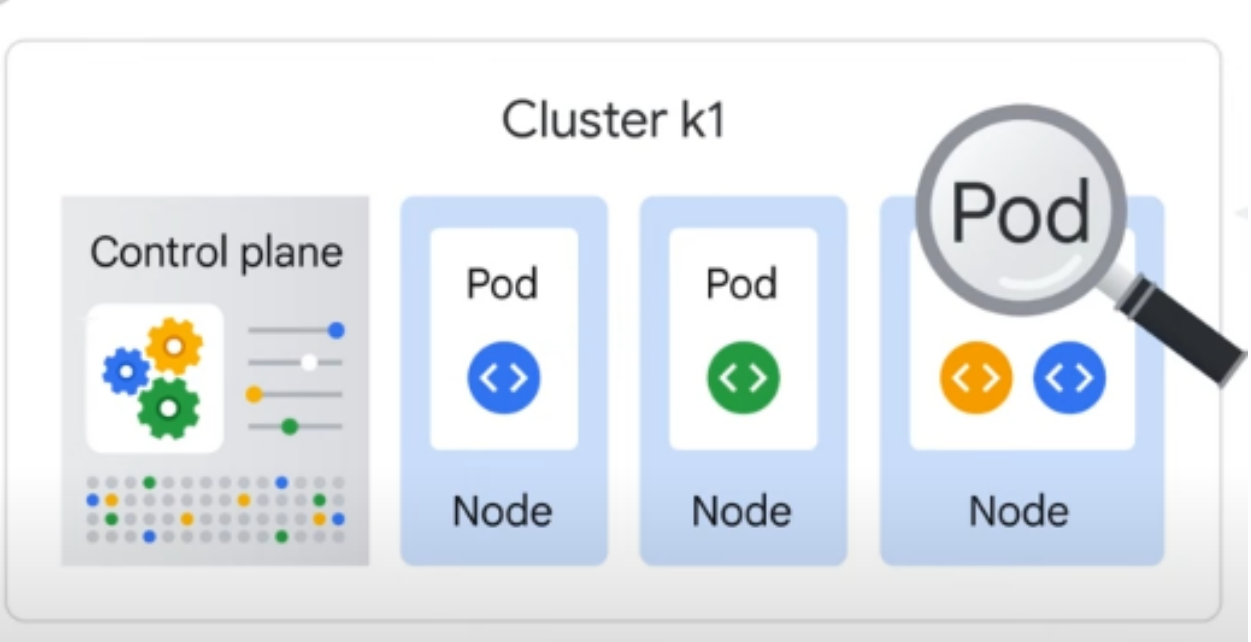
Containers:

- The idea of a container is to give the independent scalability of workloads in PaaS and an abstraction layer of the OS and hardware in IaaS. A configurable system lets you install your favorite runtime, web server, database, or middleware, configure the underlying system resources, such as disk space, disk I/O, or networking, and build as you like.
- A container is an invisible box around your code and its dependencies with limited access to its own partition of the file system and hardware. It only requires a few system calls to create and it starts as quickly as a process. It only requires a few system calls to create and it starts as quickly as a process. All that's needed on each host is an OS kernel that supports containers and a container runtime. It scales like PaaS but gives you nearly the same flexibility as IaaS. This makes code ultra portable, and the OS and hardware can be treated as a black box. So you can go from development, to staging, to production, or from your laptop to the cloud, without changing or rebuilding anything.

Kubernetes:

- Kubernetes is an open-source platform for managing containerized workloads and services. It helps manage and scale containerized applications.
- Kubernetes makes it easy to orchestrate many containers on many hosts, scale them as microservices, and easily deploy rollouts and rollbacks.
- Kubernetes is a set of APIs that you can use to deploy containers on a set of nodes called a cluster.
- Kubernetes Divided into a set of primary components that run as the control plane and a set of nodes that run containers.
- You can describe a set of applications and how they should interact with each other, and Kubernetes determines how to make that happen.

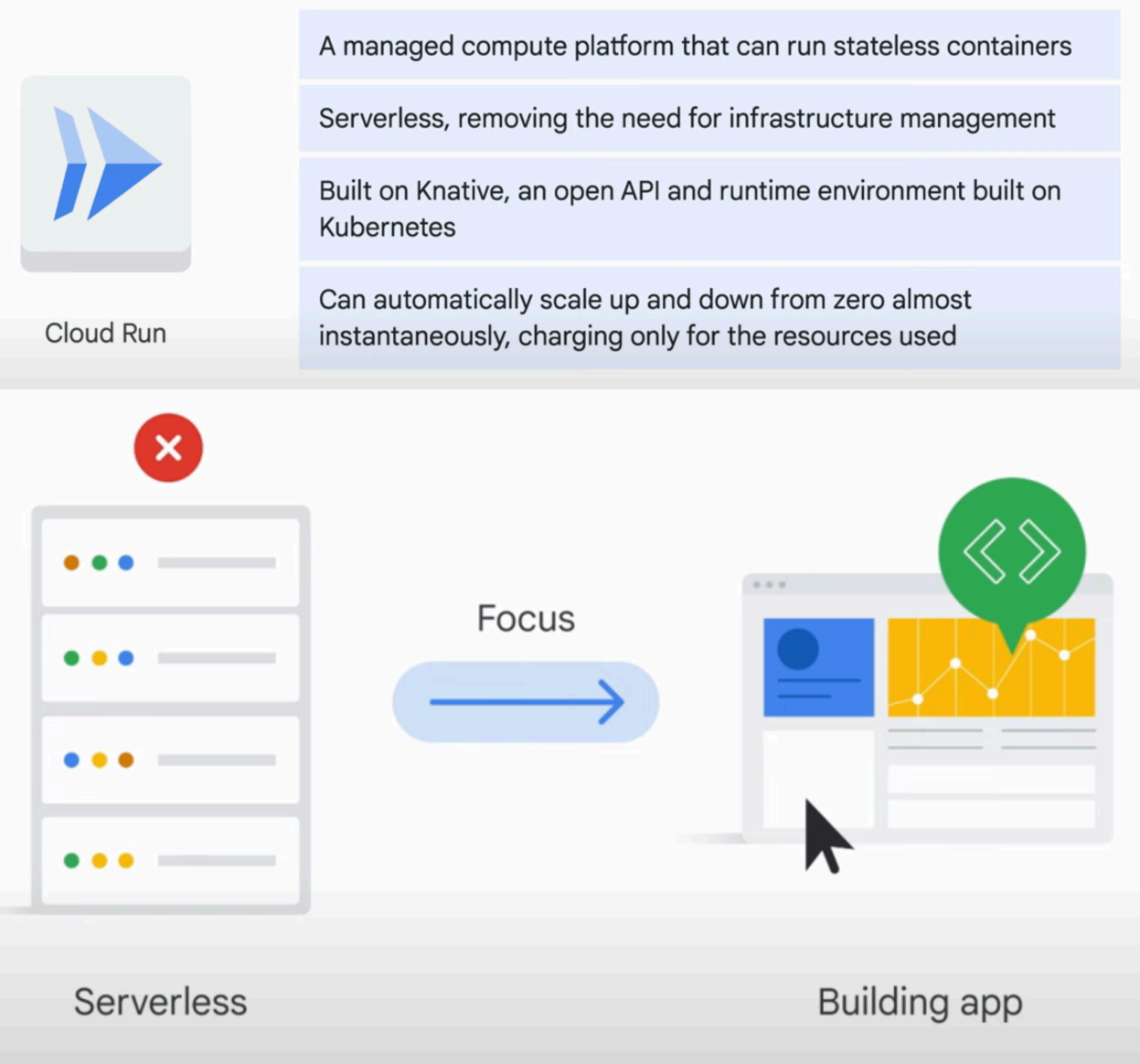
- Deploying containers on nodes by using a wrapper around one or more containers is what defines a Pod.
- A Pod is the smallest unit in Kubernetes that you can create or deploy. It represents a running process on your cluster as either a component of your application or an entire app. Generally, you only have one container per Pod, but if you have multiple containers with a hard dependency, you can package them into a single Pod and share networking and storage resources between them.
- The Pod provides a unique network IP and set of ports for your containers and configurable options that govern how your containers should run.



Google Kubernetes Engine (GKE):

- To save time and effort when scaling applications and workloads, Kubernetes can be bootstrapped using Google Kubernetes Engine (GKE).
- GKE is a Google-hosted managed Kubernetes service in the cloud.
- The GKE environment consists of multiple machines, specifically Compute Engine instances, grouped together to form a cluster.
- GKE manages all the control plane components for us. It still exposes an IP address to which we send all of our Kubernetes API requests, but GKE takes responsibility for provisioning and managing all the control plane infrastructure behind it.
- With the Autopilot mode, which is recommended, GKE manages the underlying infrastructure such as node configuration, autoscaling, auto-upgrades, baseline security configurations, and baseline networking configuration.
- With the Standard mode, you manage the underlying infrastructure, including configuring the individual nodes.
- The GKE Standard mode has the same functionality as Autopilot, but you're responsible for the configuration, management, and optimization of the cluster. Unless you require the specific level of configuration control offered by GKE standard, it's recommended that you use Autopilot mode.

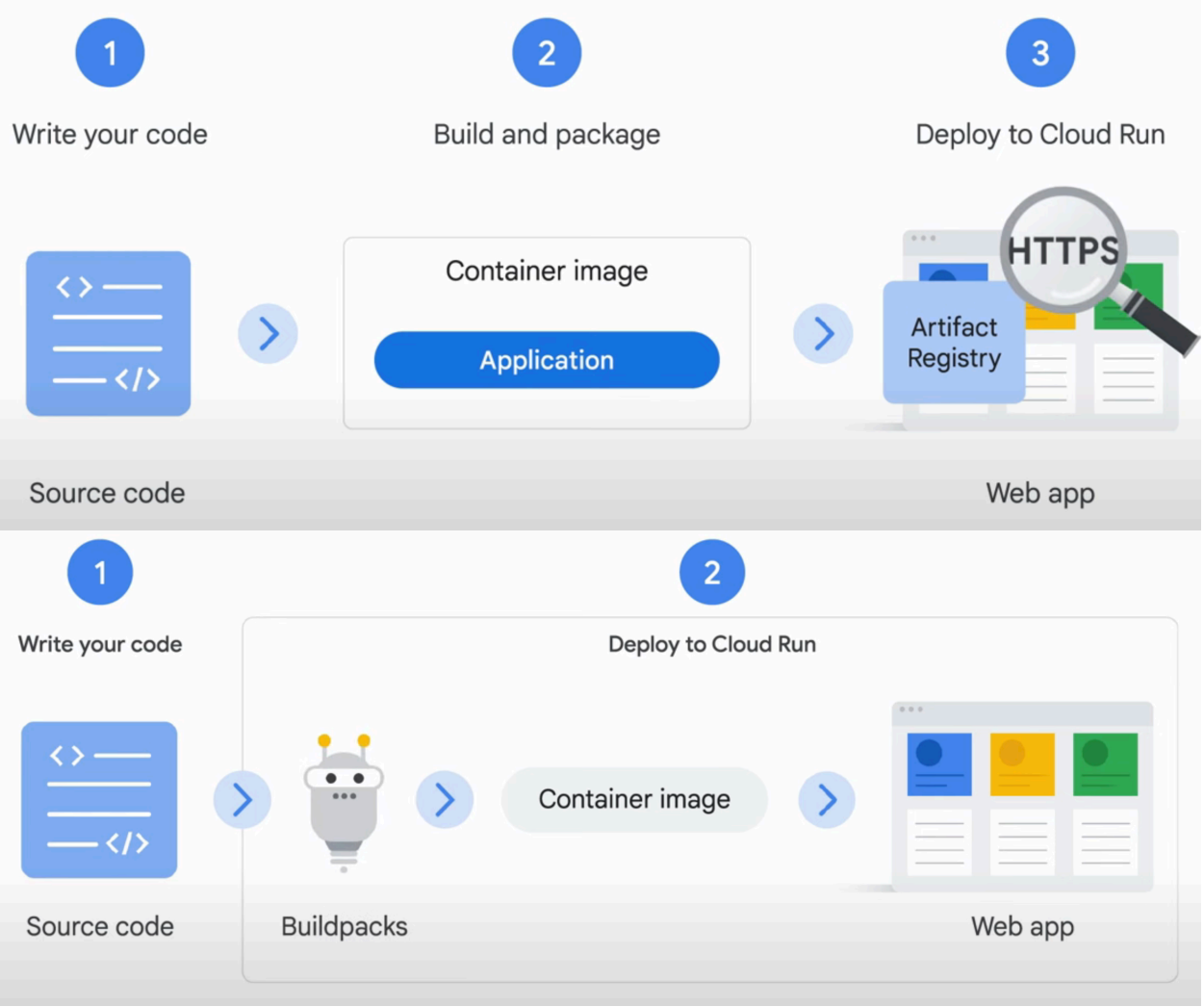
Cloud Run:



Cloud Run Workflows:

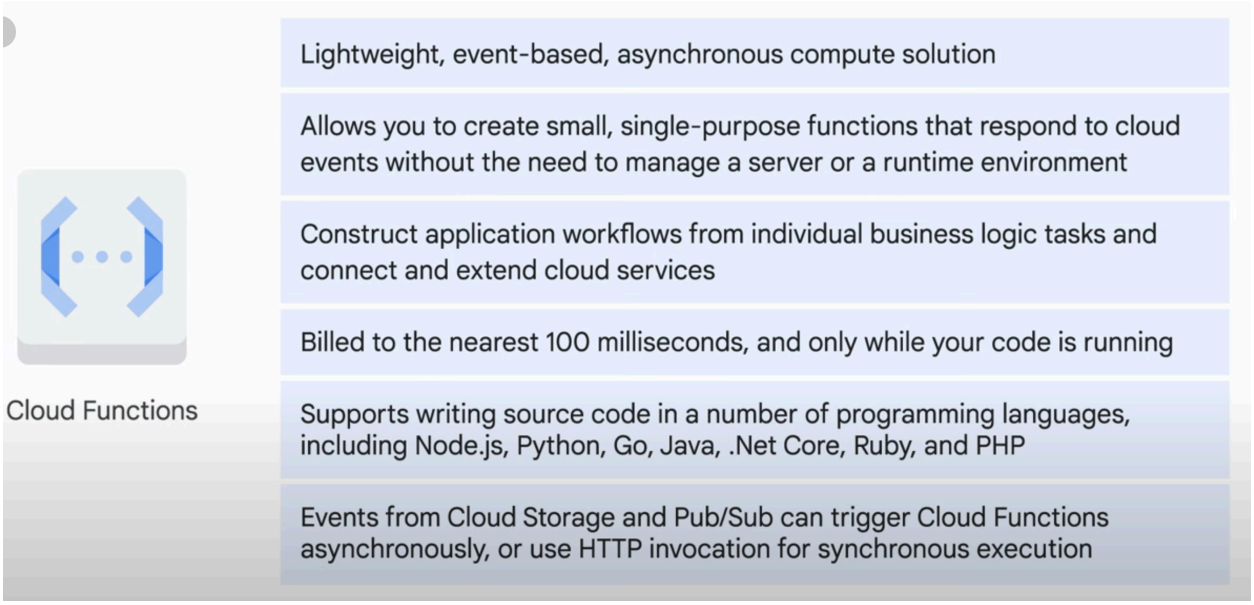
- The Cloud Run developer workflow is a straightforward three-step process. First, you write your application using your favorite programming language. This application should start a server that listens for web requests. Second, you build and package your application into a container image. And third, the container image is pushed to Artifact Registry, where Cloud Run will deploy it. Once you've deployed your container image, you'll get a unique HTTPS URL back. Cloud Run then starts your container on demand to handle requests, and ensures that all incoming requests are handled by dynamically adding and removing containers. A container-based workflow is great, because it gives you a great amount of transparency and flexibility.
- Sometimes, you're just looking for a way to turn source code into an HTTPS endpoint, and you want your vendor to make sure your container image is secure, well-configured and built

in a consistent way. With Cloud Run, you can do both. You can use a container-based workflow, as well as a source-based workflow. The source-based approach will deploy source code instead of a container image. Cloud Run then builds the source and packages the application into a container image. Cloud Run does this using Buildpacks - an open source project. Cloud Run handles HTTPS serving for you. That means you only have to worry about handling web requests, and you can let Cloud Run take care of adding the encryption.

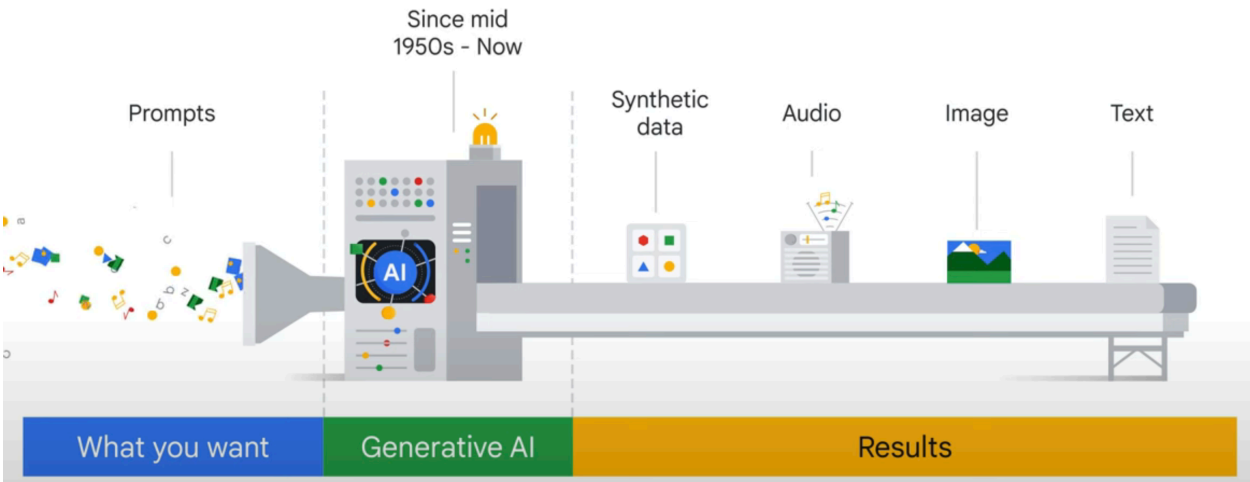


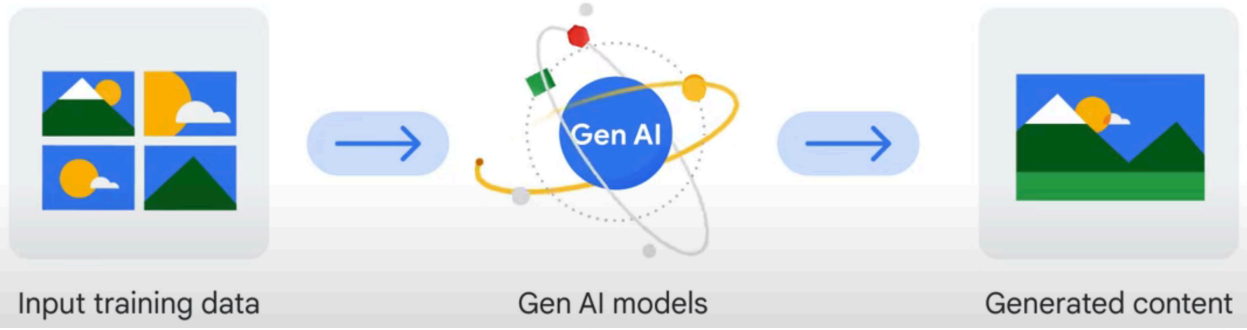
Cloud Functions:

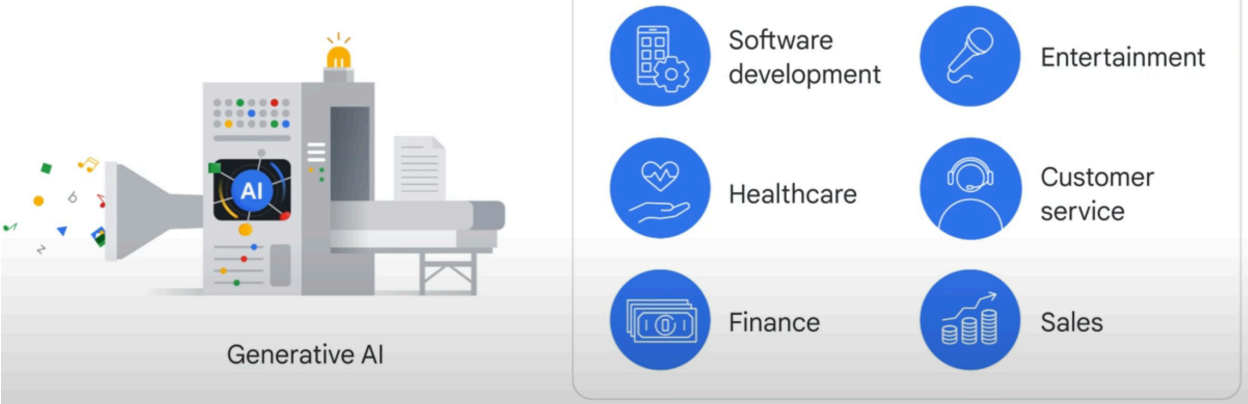
Many applications contain event-driven parts. For example, an application that lets users upload images. When that event takes place, the image might need to be processed in a few different ways, like converting the image to a standard format, converting a thumbnail into different sizes, and storing each new file in a repository. This function could be integrated into the application, but then you'd have to provide compute resources for it–whether it happens once a millisecond or once a day. With Cloud Functions, you write a single-purpose function that completes the necessary image manipulations and then arrange for it to automatically run whenever a new image is uploaded.



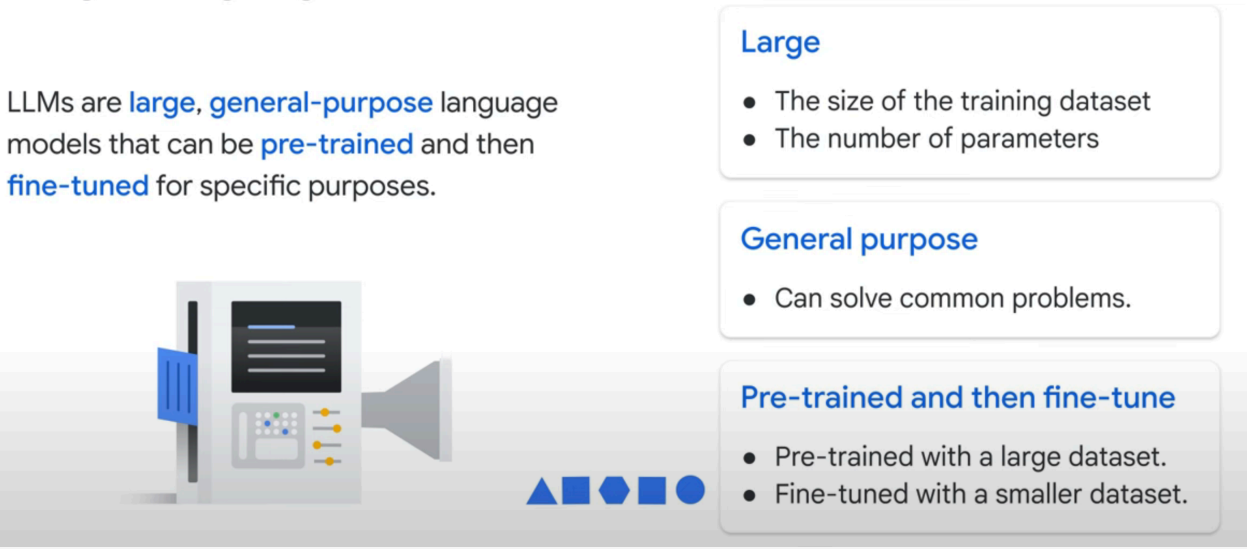
Generative AI:



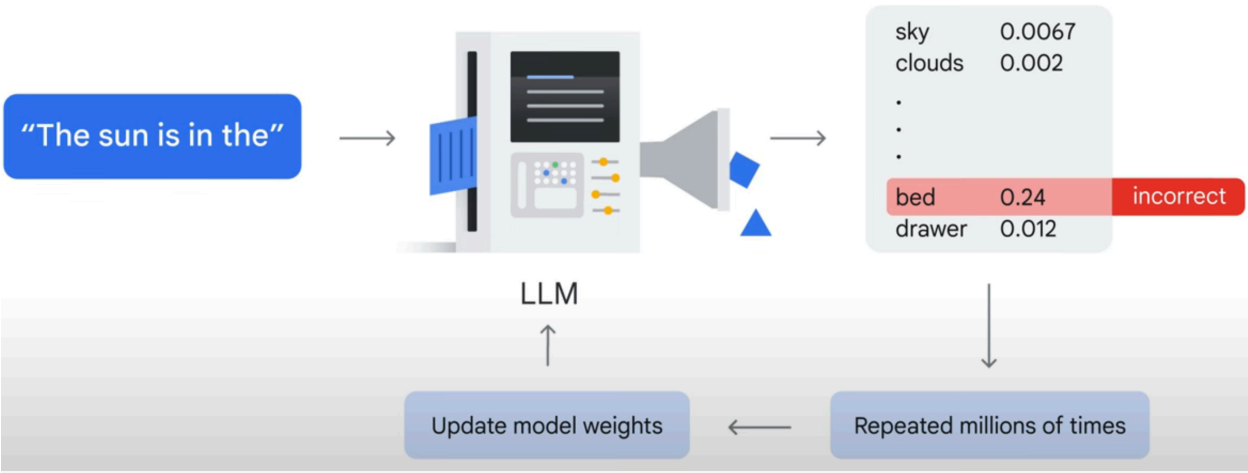




Large Language Models (LLM):



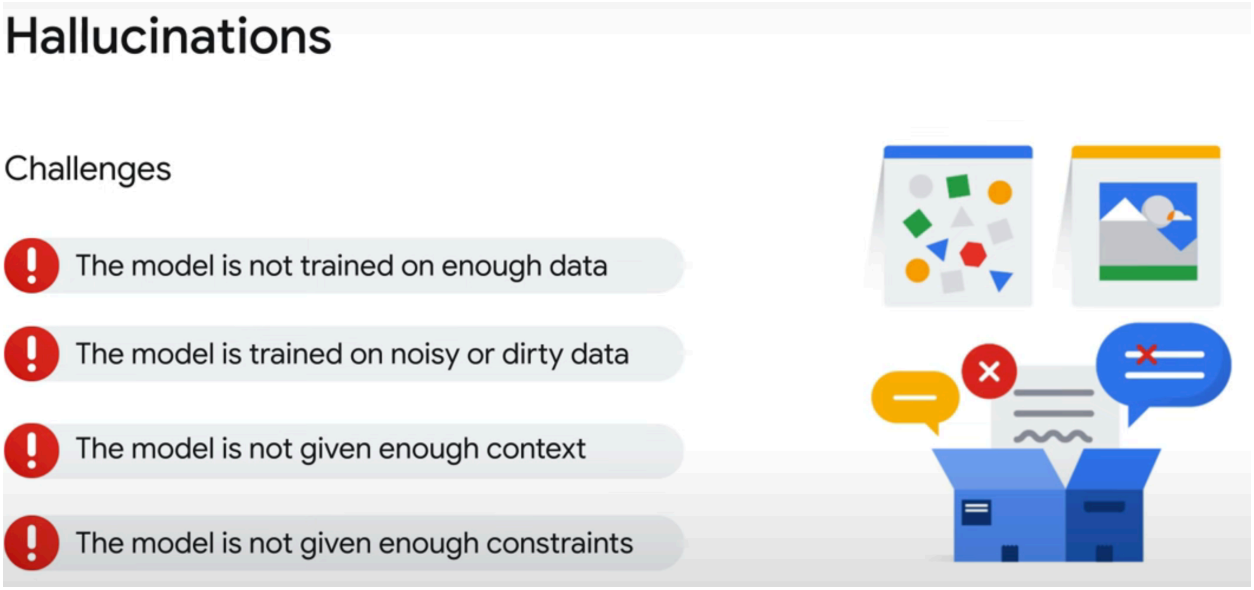
How are LLMs trained?



Language model pre-training

Hallucinations:

Hallucinations are words or phrases that are generated by the model that are often nonsensical or grammatically incorrect. This happens because LLMs can only understand the information they were trained on. This means that they might not be aware of your business's proprietary or domain-specific data. Also, they do not have access to real-time information. To make matters worse, LLMs only understand the information that is explicitly given to them in the prompt. In other words, they often assume that the prompt is true. They also do not have the ability to ask for more context information. Ultimately, an LLM does not know anything outside of what it was trained on, and it cannot truly know if that information is accurate.



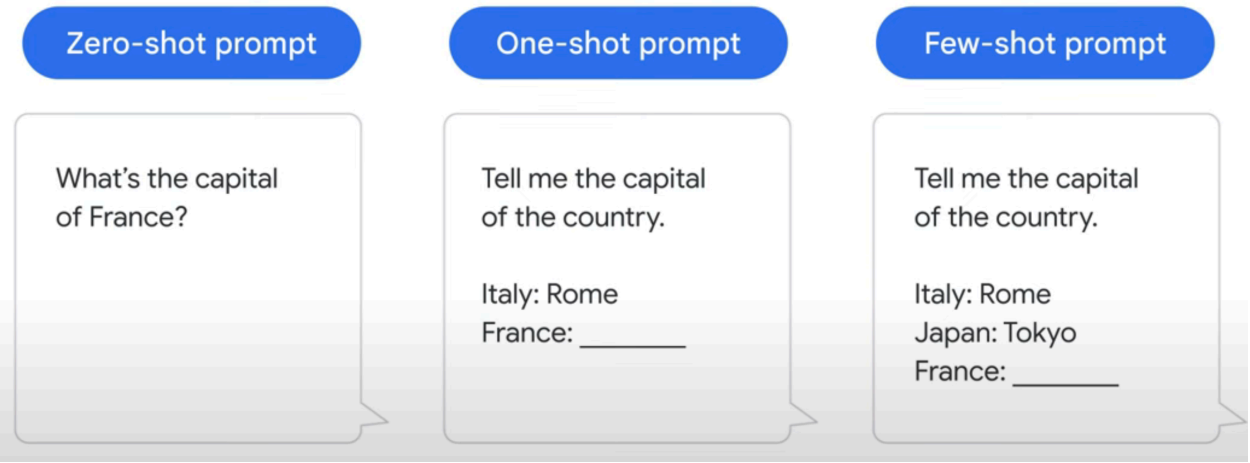
Gemini:

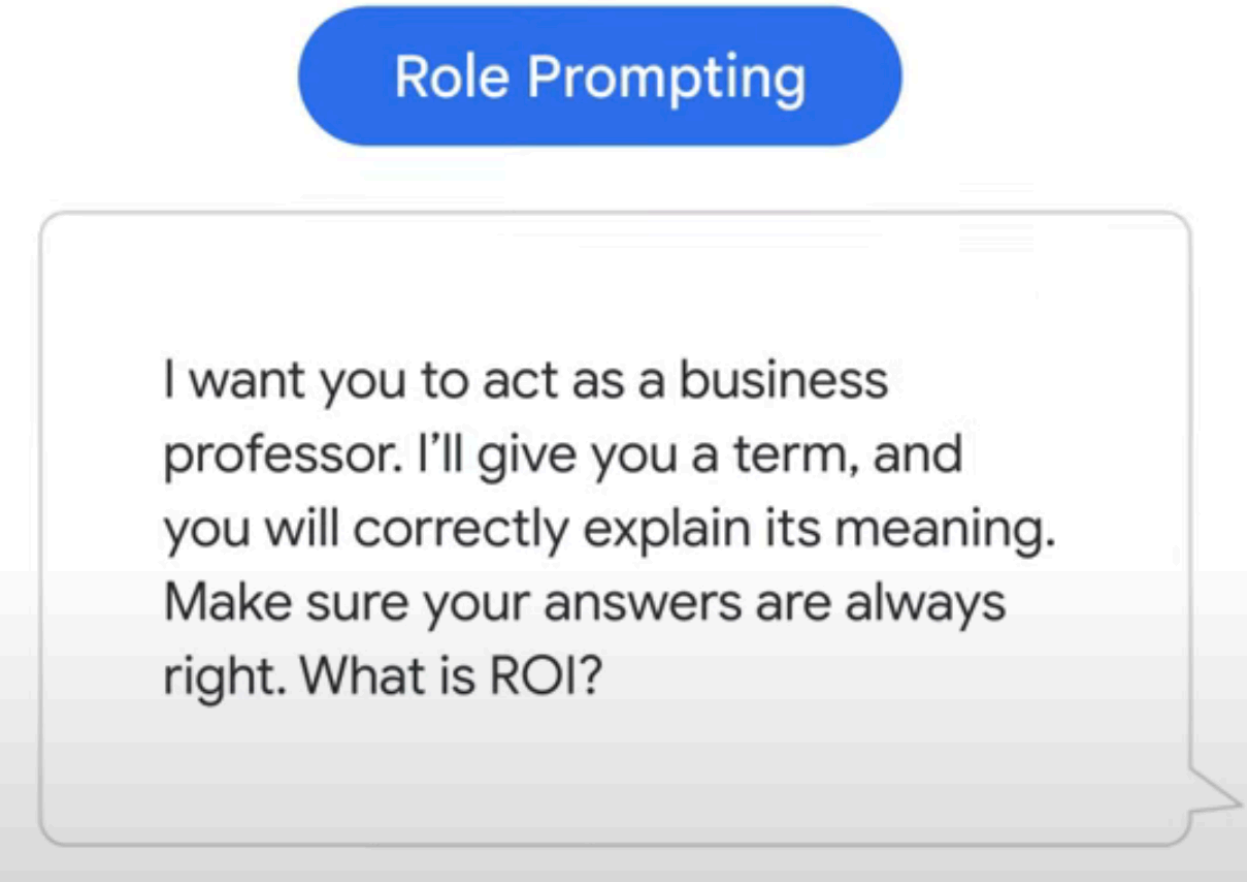
Gemini is a generative AI-powered assistant can help a wide range of Google Cloud users, including developers, data scientists, and operators. To provide an integrated assistance experience, Gemini is embedded in many Google Cloud products. Gemini has access to a massive range of data, including Google Cloud documentation, tutorials, and samples. Gemini can even create detailed gcloud commands and insert them into Cloud Shell.

Prompt Engineering:

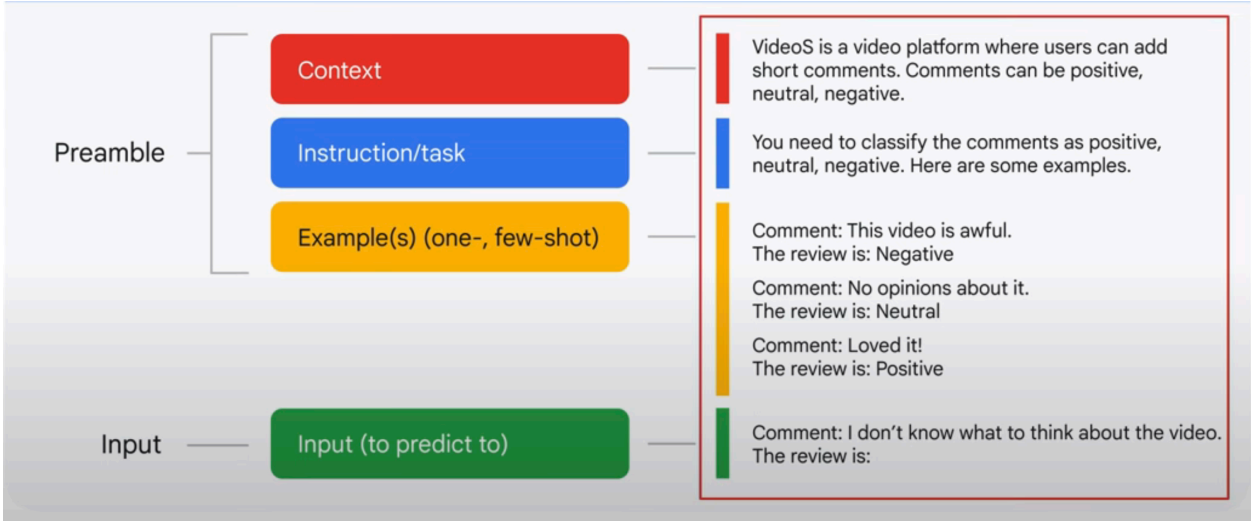
A prompt is the text that you feed to the model, and prompt engineering is a way of articulating your prompts to get the best response from the model. The better structured a prompt is, the better the output from the model will be.

Types of Prompts:





The Elements of Prompt:



Prompt Engineering Best Practices:

Write detailed and explicit instructions

Weak prompt

Summarize the meeting notes.

Better prompt

Summarize the meeting notes in a single paragraph. Then write a markdown list of the speakers and each of their key points. Finally, list the next steps or action items suggested by the speakers, if any.

Define boundaries for the prompt

Weak prompt

The following is an agent that recommends movies to a customer.

Do not ask for interests. Do not ask for personal information.

Customer: Please recommend a movie based on my interests.

Agent:

Better prompt

The following is an agent that recommends movies to a customer.

The agent is responsible for recommending a movie from the top global trending movies. It should refrain from asking users for their preferences and avoid asking for personal information.

If the agent doesn't have a movie to recommend, it should respond "Sorry, couldn't find a movie to recommend today."

Customer: Please recommend a movie based on my interests.

Agent:

Adopt a persona for your input

Weak prompt

What is the most reliable Google Cloud multi-region network architecture?

Better prompt

You're a cloud architect. You want to build a Google Cloud VPC network that can be centrally managed but also connect to other VPC networks in your company's other regions, so you don't have many different sets of firewall policies to maintain. What sort of network architecture would you recommend?

Keep each sentence concise

Good prompt

You're a cloud architect. You want to build a Google Cloud VPC network that can be centrally managed but also connect to other VPC networks in your company's other regions, so you don't have many different sets of firewall policies to maintain. What sort of network architecture would you recommend?

Better prompt

You're a cloud architect. You want to build a Google Cloud VPC network that can be centrally managed. You also connect to other VPC networks in your company's other regions. You don't want to have many different sets of firewall policies to maintain. What sort of network architecture would you recommend?