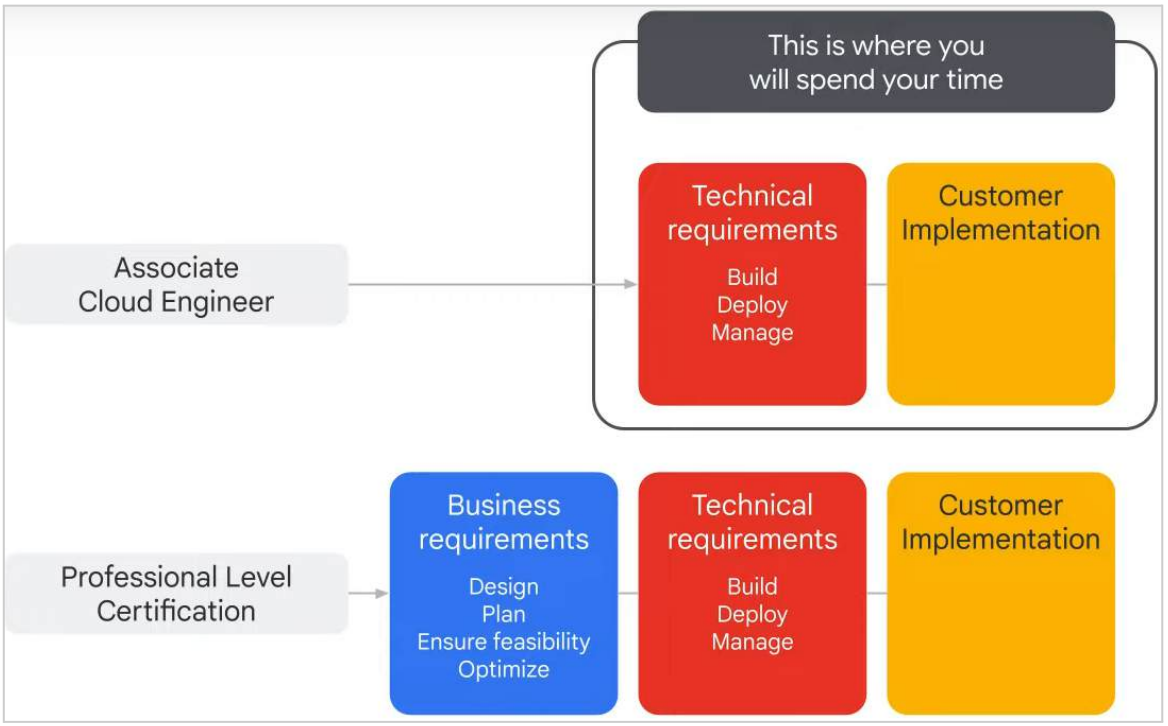
GCP Associate Cloud Engineer role



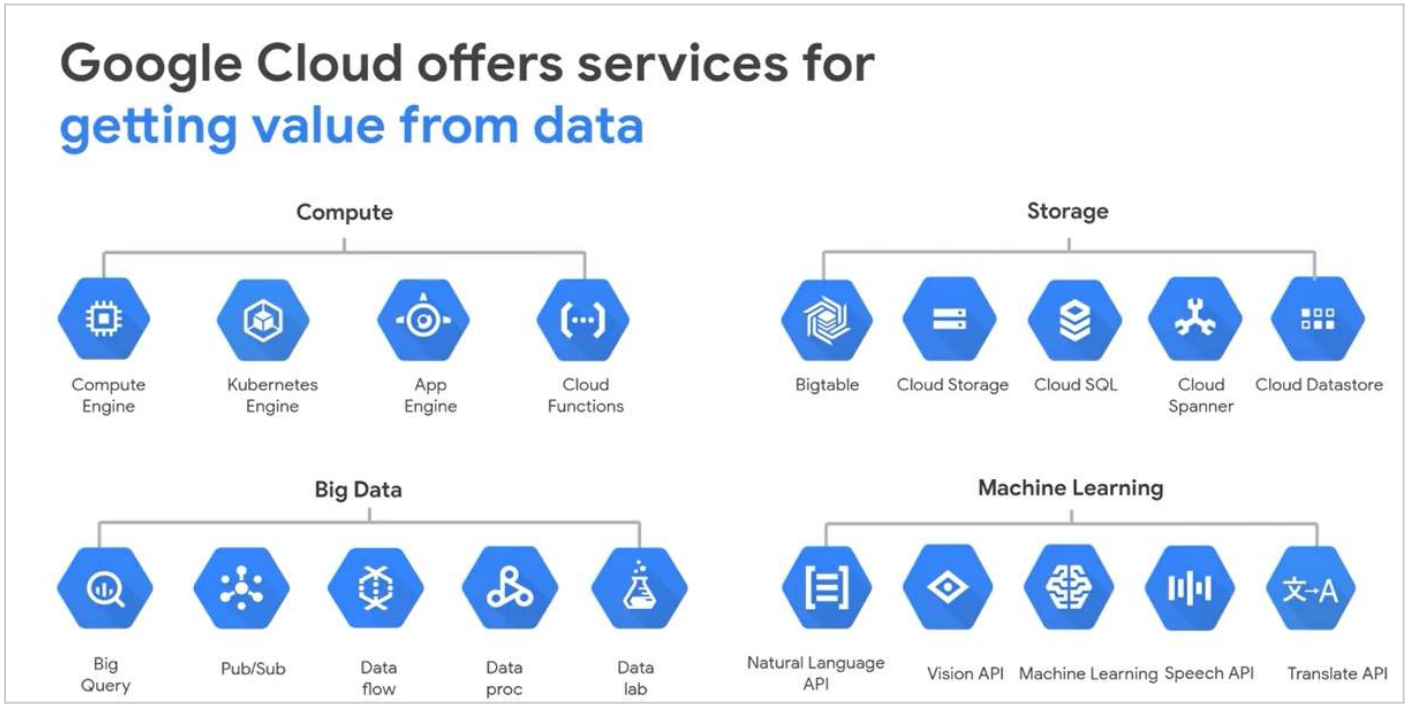
Google Cloud is a suite of cloud services hosted on Google's infrastructure. From computing and storage, to data analytics, machine learning, and networking, Google Cloud offers a wide variety of services and

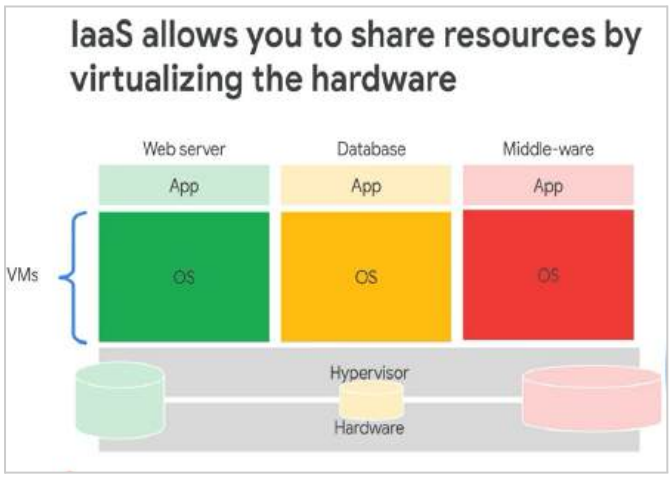
APIs that can be integrated with any cloud-computing application or project, from personal to enterprise-grade.

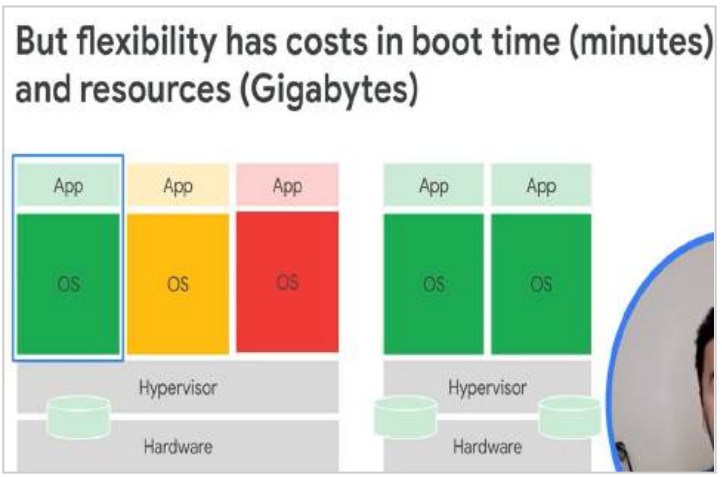
There are seven categories of Google Cloud services:

- Compute: A variety of machine types that support any type of workload. The different computing options let you decide how much control you want over operational details and infrastructure.
- Storage: Data storage and database options for structured or unstructured, relational or non relational data.
- Networking: Services that balance application traffic and provision security rules.
- Cloud Operations: A suite of cross-cloud logging, monitoring, trace, and other service reliability tools.
- Tools: Services that help developers manage deployments and application build pipelines.
- Big Data: Services that allow you to process and analyze large datasets.
- Artificial Intelligence: A suite of APIs that run specific artificial intelligence and machine learning tasks on Google Cloud.

Distributed Cloud Edge lets you run private Kubernetes clusters on hardware that is maintained by Google, including remote management by a dedicated Google team to enable low-latency or data residency use cases. Edge zones are a logical grouping of physical Distributed Cloud Edge hardware installed in the same location. GCP

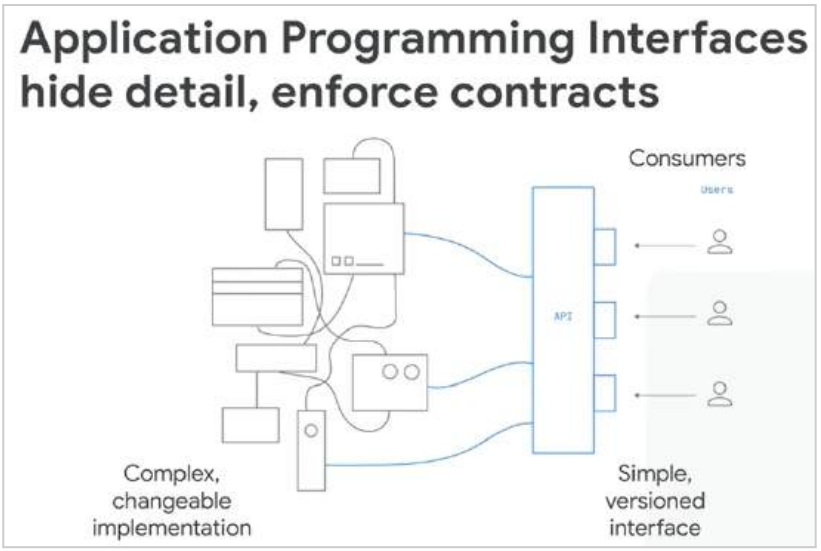






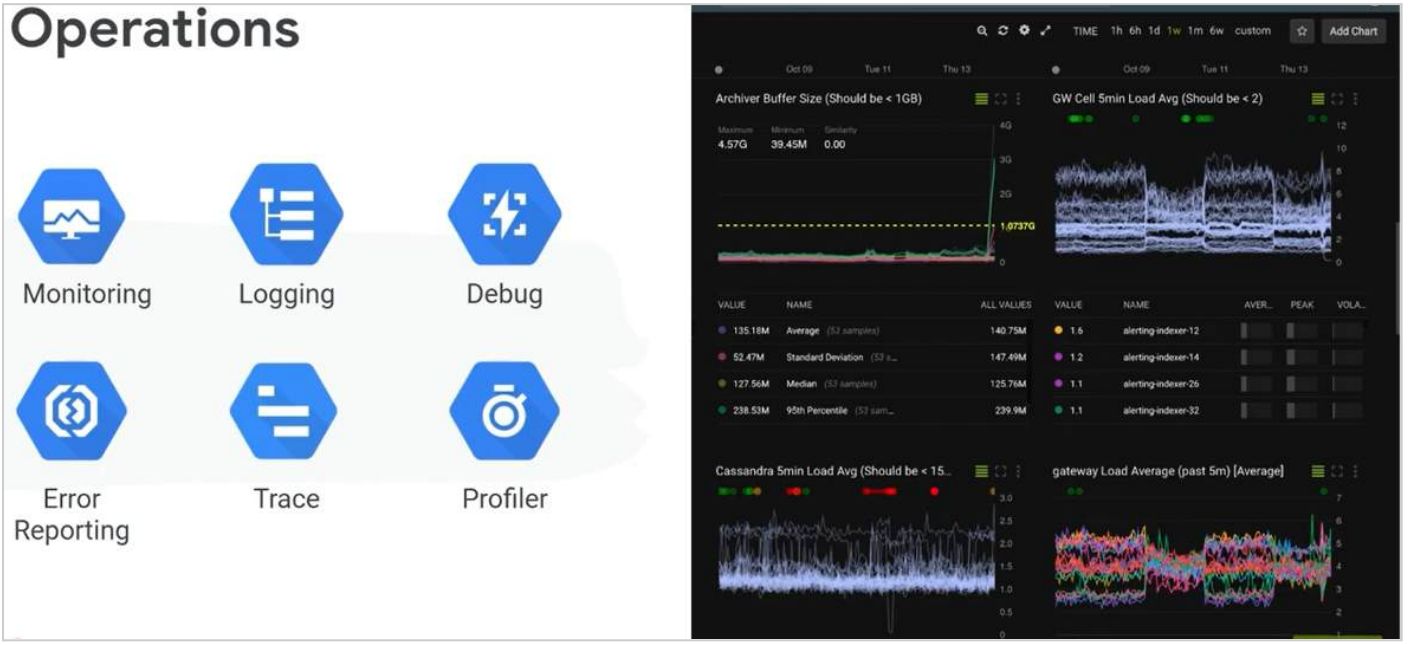
Cloud Source Repositories

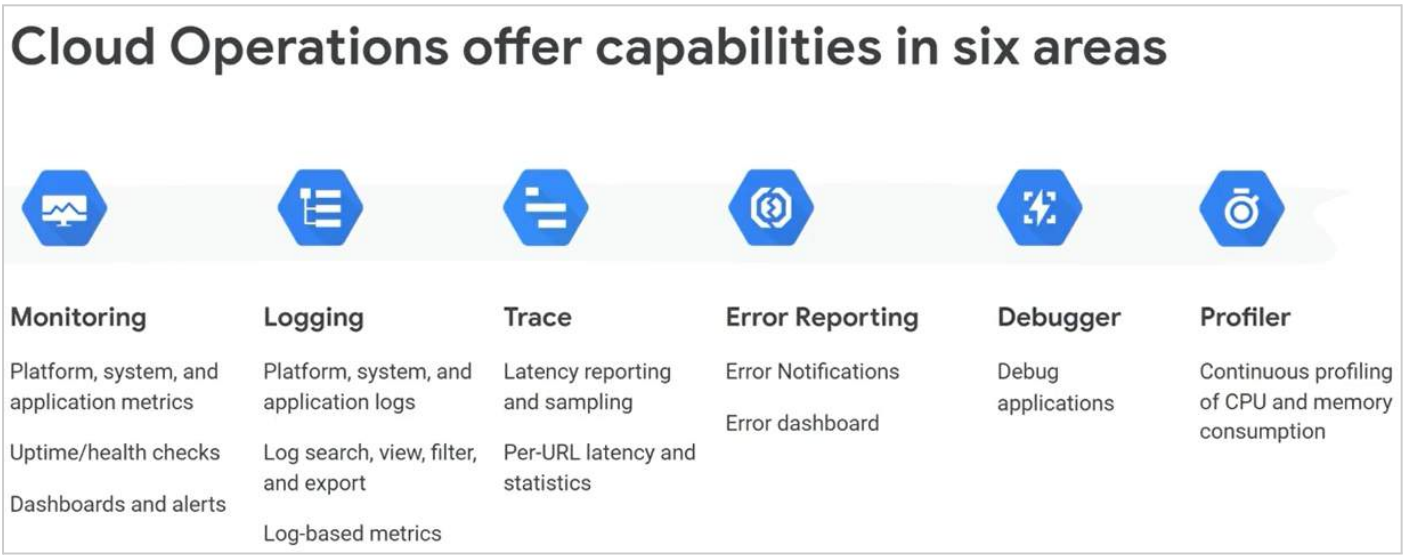
- Fully featured Git repositories hosted on Google Cloud Platform
- Supports collaborative development of cloud apps
- Includes integration with Cloud Debugger



Deployment Manager

- Infrastructure management service
- Create a .yaml template describing your environment and use Deployment Manager to create resources
- Provides repeatable deployments





When you configure the load balancing service, your virtual machine instances will receive packets that are destined for the static external IP address you configure. Instances made with a Compute Engine image are automatically configured to handle this IP address.

Setting the tags field lets you reference these instances all at once, such as with a firewall rule.

Managed instance groups (MIGs) let you operate apps on multiple identical VMs. You can make your workloads scalable and highly available by taking advantage of automated MIG services, including: autoscaling, autohealing, regional (multiple zone) deployment, and automatic updating.

Google Cloud provides health checking mechanisms that determine whether backend instances respond properly to traffic
URL map is a Google Cloud configuration resource used to route requests to backend services or backend buckets. For example, with an external HTTP(S) load balancer, you can use a single URL map to route requests to different destinations based on the rules configured in the URL map
A forwarding rule and its corresponding IP address represent the frontend configuration of a Google Cloud load balancer.

Regional persistent disks share replicas of the physical disks across two zones.
When you create a virtual machine instance in the console it uses balanced SSD, while when you create one via a gcloud command, it uses standard HDD. Balanced SSD gives you higher I/O than standard HDD, but less cost and I/O than fully capable SSD disks.
Local SSDs can be added to your instances based on machine type. They provide very high I/O since they are physically connected to the server your VM is running on. They are ephemeral, and thus go away when your VM is stopped or terminated. Data on your local SSD will survive a reboot. You are responsible for formatting and striping the local SSD per your requirements.
Opportunistic updates are not interactive.

The APIs Explorer uses its own API key whenever it makes a request. When you use the APIs Explorer to make a request, it displays the request syntax, which includes a placeholder labeled {YOUR_API_KEY}. If you want to make the same request in your application, you need to replace this placeholder with your own API key

(BETA) Beta versions of gcloud command

Associate Cloud Engineer : Cloud Engineer Learning Path

Section 1. Setting up a cloud solution environment

1.1 Setting up cloud projects and accounts. Activities include:

a. Creating a resource hierarchy

- The *organization* is the root node in the hierarchy.
- *Folders* are children of the organization.
- *Projects* are children of the organization, or of a folder.
- *Resources* for each service are descendants of projects.

Each resource has exactly one parent.The organization resource is closely associated with a Google Workspace or Cloud Identity account.organization resource ID, which is a unique identifier for an organization.The initial IAM policy for a newly created Organization resource grants the Project Creator and Billing Account Creator roles to the entire Google Workspace domain.

Folder resources allow delegation of administration rights, so for example, each head of a department can be granted full ownership of all Google Cloud resources that belong to their departments.

Project resources consist Two identifiers:

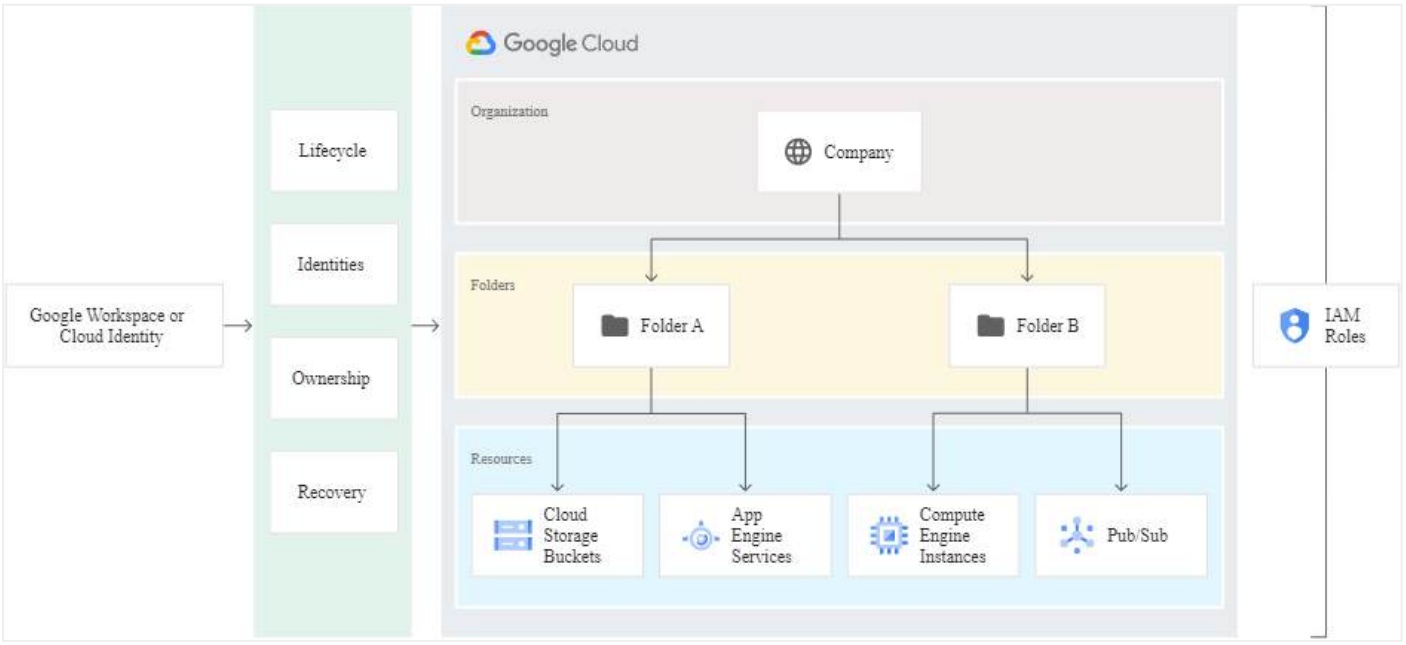
1. Project resource ID, which is a unique identifier for the project resource.
2. Project resource number, which is automatically assigned when you create the project. It is read-only.

A project resource ID is the customized name you chose when you created the project resource. The initial IAM policy for the newly created project resource grants the owner role to the creator of the project.

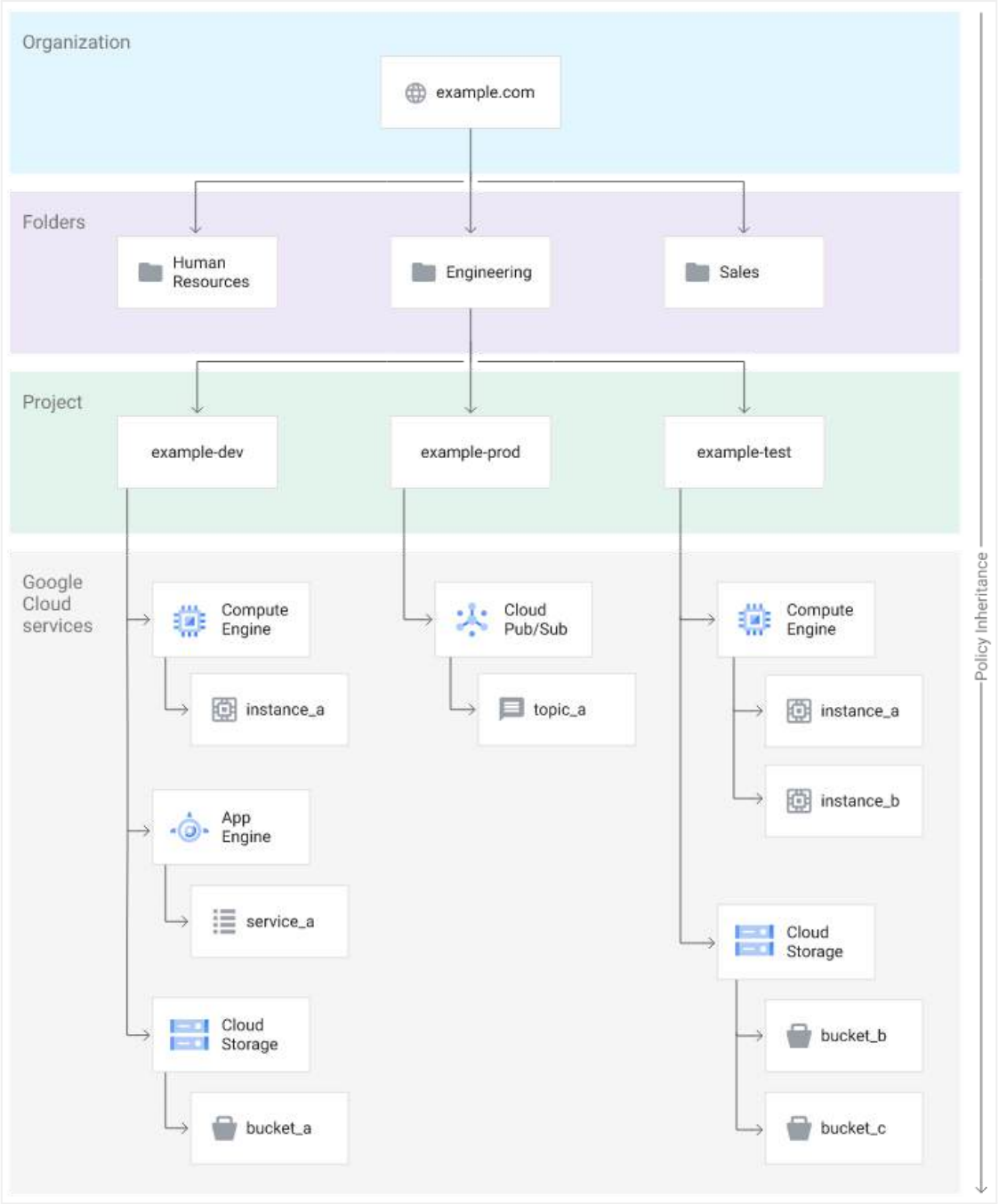
Project ID

A Google Cloud project is an organizing entity for your Google Cloud resources. It often contains resources and services; for example, it may hold a pool of virtual machines, a set of databases, and a network that connects them together. Projects also contain settings and permissions, which specify security rules and who has access to what resources.Your project has a name, ID, and number. These identifiers are frequently used when interacting with Google Cloud services.

A *Project ID* is a unique identifier that is used to link Google Cloud resources and APIs to your specific project. Project IDs are unique across Google Cloud: there can be only one qwiklabs-gcp-xxx...., which makes it globally identifiable.



The Google Workspace super admin is the individual responsible for domain ownership verification and the contact in cases of recovery. For this reason, the Google Workspace super admin is granted the ability to assign IAM roles by default. The Google Workspace super admin's main duty with respect to Google Cloud is to assign the Organization Administrator IAM role to appropriate users in their domain. This will create the separation between Google Workspace and Google Cloud administration responsibilities that users typically seek.

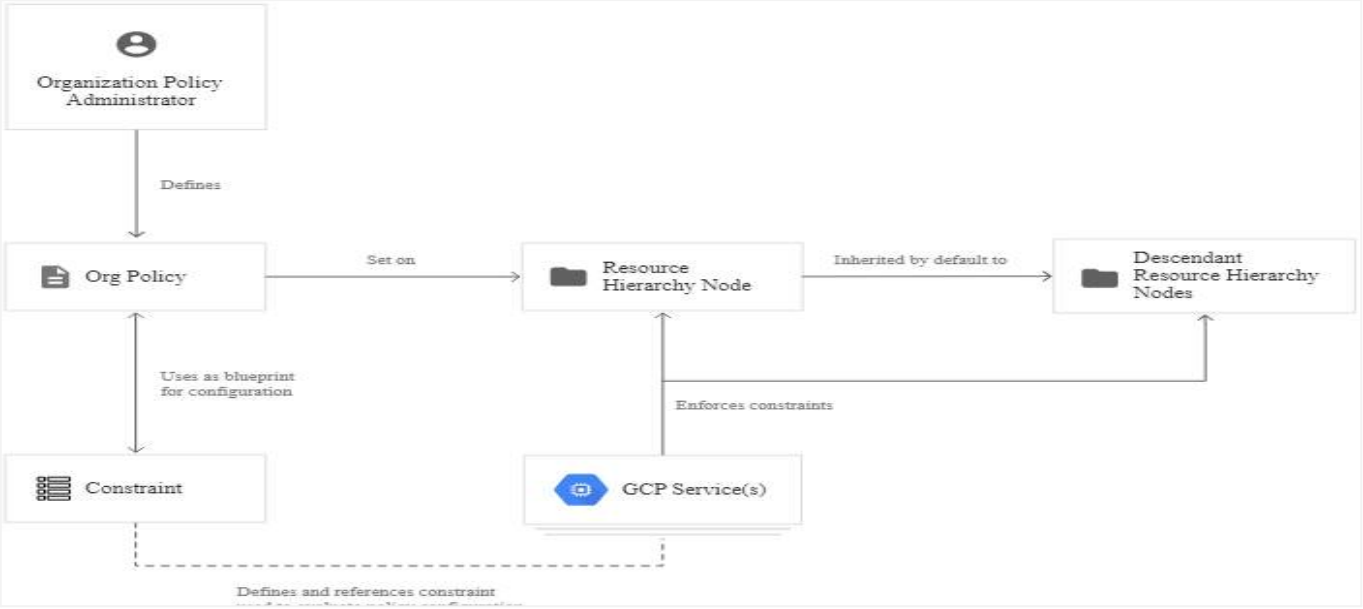


b. Applying organizational policies to the resource hierarchy

Organization Policy Service gives you centralized and programmatic control over your organization's cloud resources. A configuration of restrictions.The Identity and Access Management (IAM) role roles/orgpolicy.policyAdmin enables an administrator to manage organization policies.

- Limit resource sharing based on domain.
- Limit the usage of Identity and Access Management (IAM) service accounts.
- Restrict the physical location of newly created resources.

Identity and Access Management (IAM) focuses on **who,** Organization Policy focuses on **what**,



A constraint is a particular type of restriction against a Google Cloud service or a list of Google Cloud services. Think of the constraint as a blueprint that defines what behaviors are controlled. A constraint has a type, either list or boolean. List constraints evaluate the constraint with a list of allowed or denied values that you provide. Boolean constraints are either enforced or not enforced for a given resource, and govern a specific behavior.

If a new organization policy sets a restriction on an action or state that a service is already in, the policy is considered to be in violation, but the service will not stop its original behavior. You will need to address this violation manually. This prevents the risk of a new organization policy completely shutting down your business continuity.

To set the organization policy	gcloud org-policies set-policy POLICY_PATH

Where POLICY_PATH is the full path to your organization policy JSON file, Organization policy List constraints that are enabled using tags will merge with the existing organization policy, per normal rules of inheritance.Organization policy Boolean constraints that are enabled using tags will override the existing organization policy.

c. Granting members IAM roles within a project

In Identity and Access Management (IAM), access is granted through *allow policies*, also known as IAM policies. An allow policy is attached to a Google Cloud resource.Each allow policy contains a collection of *role bindings* that associate one or more principals, such as users or service accounts, with an IAM role.

Enable the Resource Manager API. Administrator to grant you the following IAM roles on the resource that you want to manage access for:

- To manage access to a project: Project IAM Admin (roles/resourcemanager.projectIamAdmin)
- To manage access to a folder: Folder Admin (roles/resourcemanager.folderAdmin)
- To manage access to projects, folders, and organizations: Organization Admin (roles/resourcemanager.organizationAdmin)
- To manage access to almost all Google Cloud resources: Security Admin (roles/iam.securityAdmin)

Role Name	Permissions
roles/viewer	Permissions for read-only actions that do not affect state, such as viewing (but not modifying) existing resources or data.

roles/editor	All viewer permissions, plus permissions for actions that modify state, such as changing existing resources.
roles/owner	All editor permissions and permissions for the following actions: manage roles and permissions for a project and all resources within the project; set up billing for a project.

To see who has access to your project	gcloud projects get-iam-policy my-project --format=json > ~/policy.json
To grant the Project Creator role to the user	gcloud projects add-iam-policy-binding my-project \ --member=user:my-user@example.com --role=roles/resourcemanager.projectCreator
To revoke the Project Creator role from the user	gcloud projects remove-iam-policy-binding my-project \ --member=user:my-user@example.com --role=roles/resourcemanager.projectCreator
list of all roles that can be applied to a given resource	gcloud iam list-grantable-roles full-resource-name

Note: In general, any policy changes take effect within 60 seconds. However, in some cases, it can take up to 7 minutes for changes to propagate across the system.

You can only grant roles related to activated API services.

Grant or revoke multiple roles

To make large-scale access changes that involve granting and revoking multiple roles, use the *read-modify-write* pattern to update the resource's allow policy:

1. Read the current allow policy by calling getIamPolicy().
2. Edit the allow policy, either by using a text editor or programmatically, to add or remove any principals or role bindings.
3. Write the updated allow policy by calling setIamPolicy()

To grant roles to your principals, edit the returned allow policy by adding the principal to an existing role binding. Revoke a role by editing the JSON or YAML allow policy returned by the get-iam-policy command. This change will not take effect until you set the updated allow policy.

To set the allow policy for the resource	gcloud projects set-iam-policy my-project ~/policy.json

d. Managing users and groups in Cloud Identity (manually and automated)

Cloud Identity is an Identity as a Service (IDaaS) and enterprise mobility management (EMM) product. It offers the identity services and endpoint administration that are available in Google Workspace as a stand-alone product. As an administrator, you can use Cloud Identity to manage your users, apps, and devices from a central location—the Google Admin console.

Manage user access
Cloud Identity automated provisioning

Cloud Identity has a catalog of automated provisioning connectors, which act as a bridge between Cloud Identity and third-party cloud apps.

Once you've set up SAML(Security Assertion Markup Language) for Single Sign-On (SSO), you can set up automated user provisioning to create, modify, or delete a user's identity across your cloud apps. Admins can authorize Cloud Identity to synchronize a subset of their Cloud Identity users to one or more supported apps.

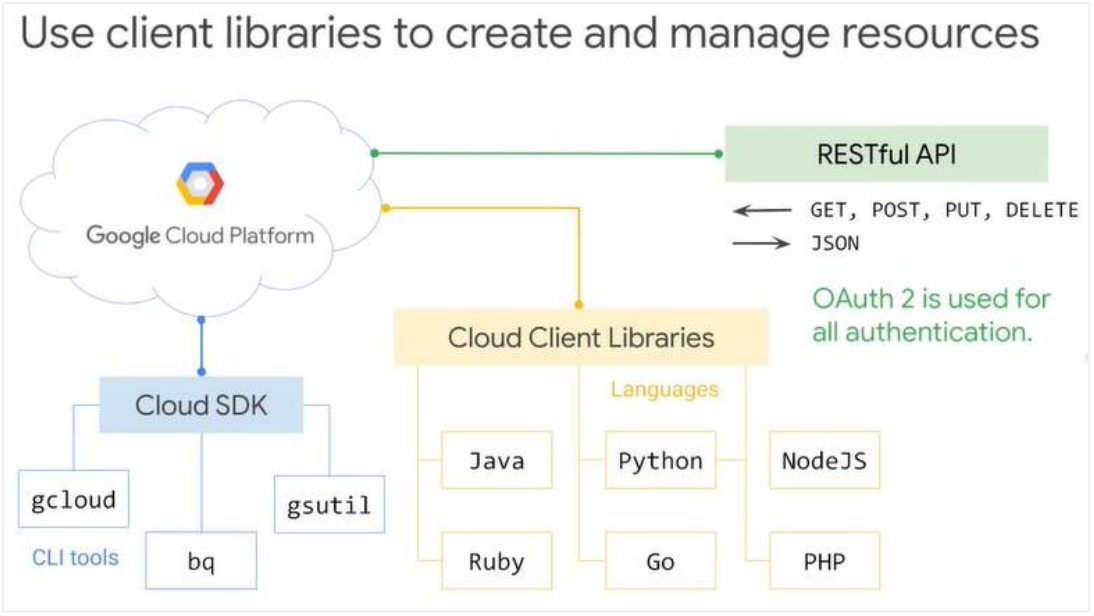
Advantages

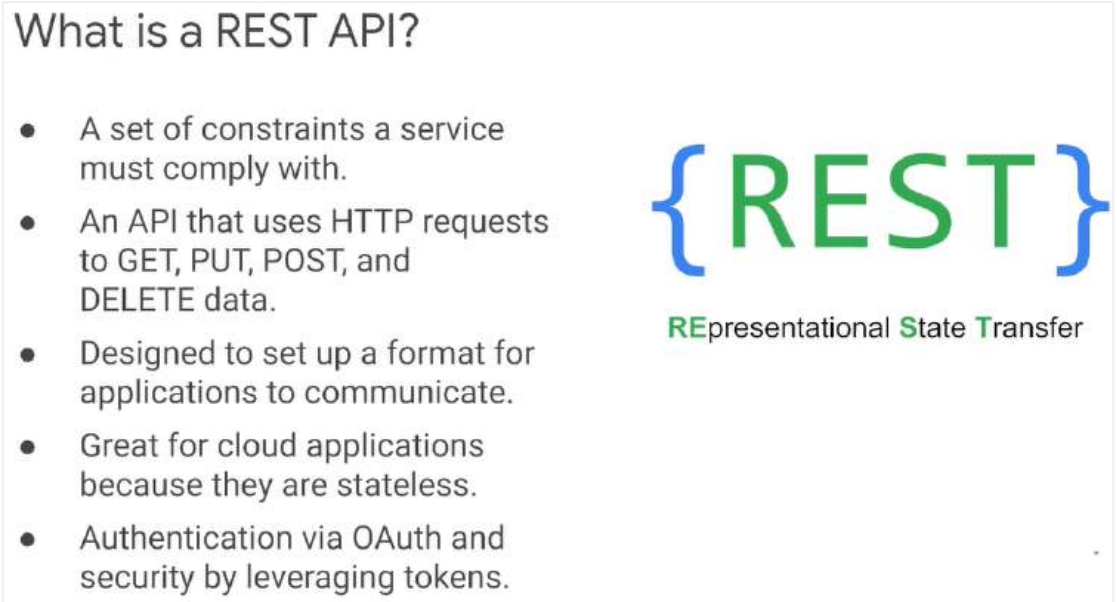
- Accommodates the full user lifecycle by creating, updating, removing, or suspending user profiles.
- Accommodates full app lifecycle management by enabling companies to add or remove applications from their organization in a central location.
- Provides a consistent user experience for all supported apps, including unified reporting, audit logs, and granular event tracking.

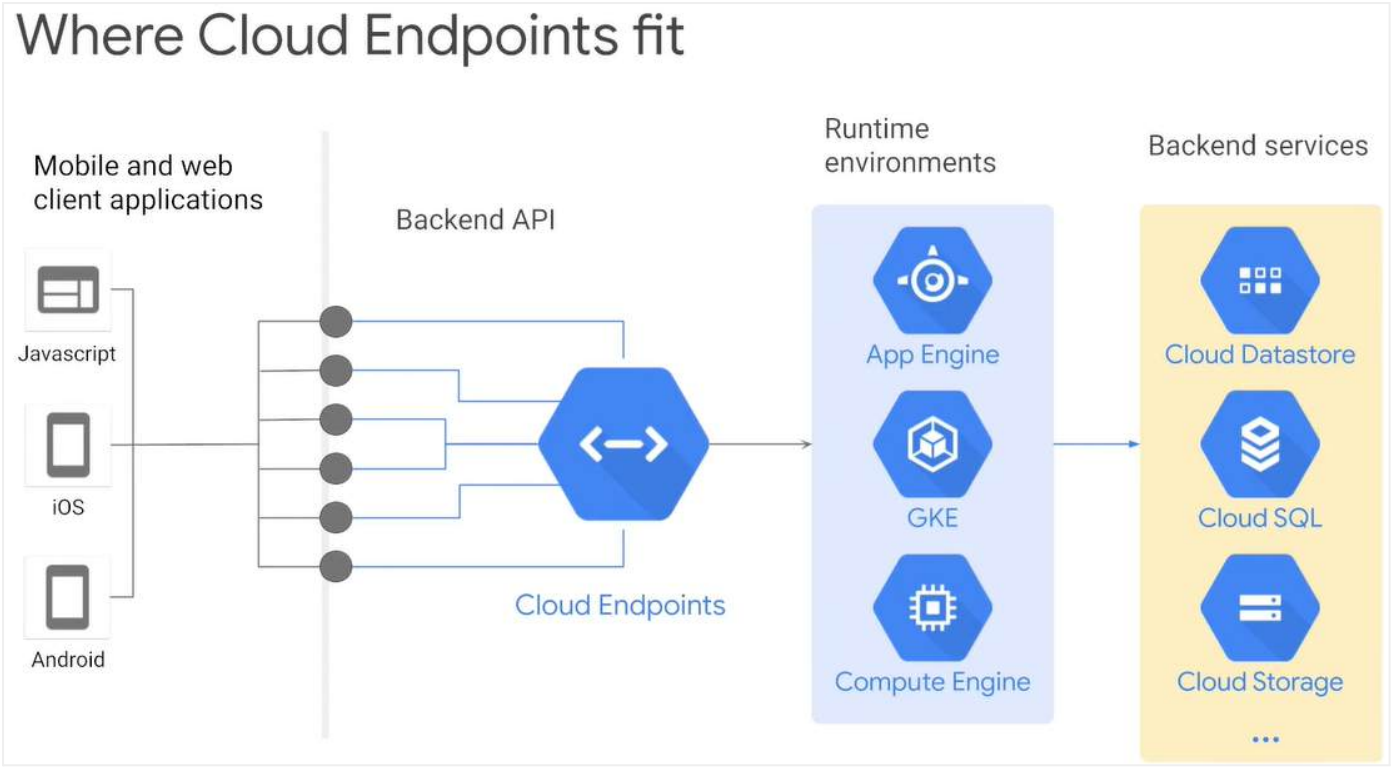
Identity provider (IdP) | Just-In-Time (JIT) provisioning usually only handles user profile creation and doesn't address profile updating, suspension, or deletion.

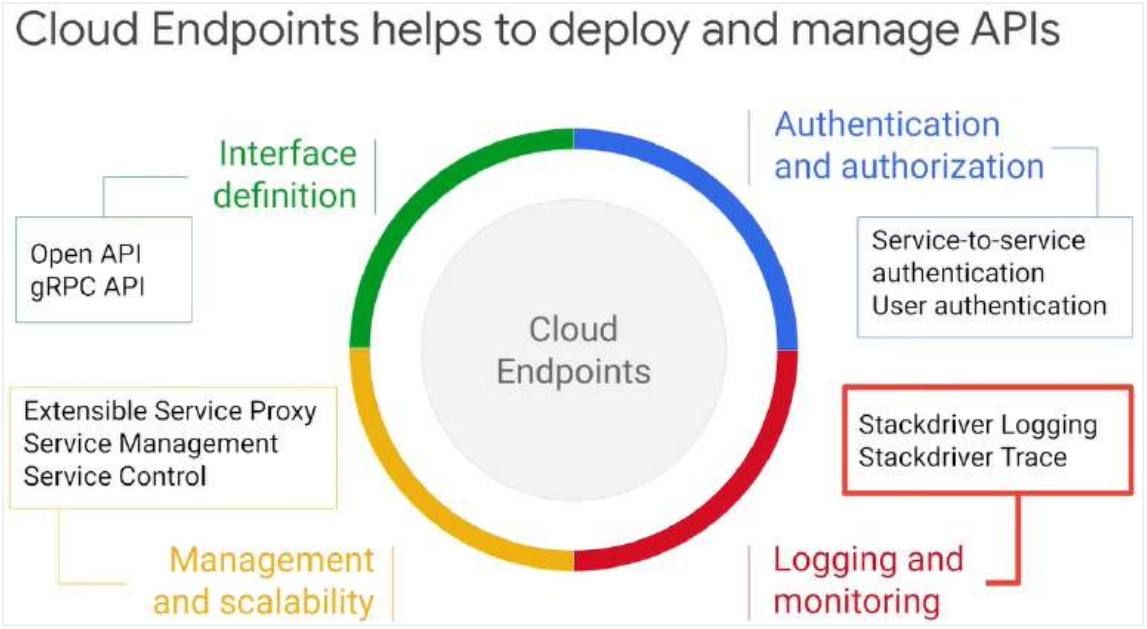
e. Enabling APIs within projects

Google Cloud provides two API management tools: Cloud Endpoints and Apigee Edge.



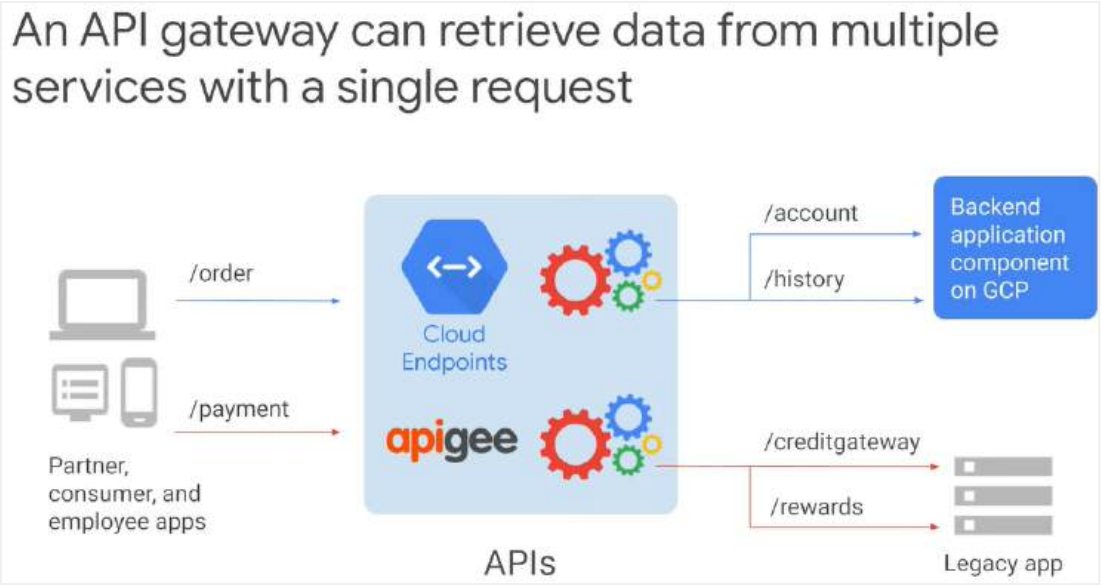






Cloud Endpoints helps you create and maintain APIs

- Distributed API management through an API console
- Expose your API using a RESTful interface
- Control access and validate calls with JSON Web Tokens and Google API keys
    - identify web, mobile users with Auth0 and Firebase Authentication
- Cloud Endpoints provides an API console, hosting, logging, monitoring, and other features to help you create, share, maintain, and secure your APIs.
- use Cloud Endpoints with any APIs that support the OpenAPI Specification.
- Cloud Endpoints supports applications running in App Engine, Google Kubernetes Engine, and Compute Engine.
- Generate client libraries : Clients include Android, iOS, and Javascript.



Apigee Edge helps you secure and monetize APIs

- A platform for making APIs available to your customers and partners
- Contains analytics, monetization, and a developer portal

When an API requires an API key and the API is associated with a Google Cloud project that you don't have access to, you have the following options to obtain an API key:

- **Option 1**: Ask a security admin to create an API key for you.
- **Option 2**: Ask a security admin to grant you access to the project so that you can create an API key in the same project that the API is associated with.
- **Option 3**: Ask a security admin to grant you access to enable the API in your own Google Cloud project so that you can create an API key.

To display the project IDs	gcloud projects list
Using the applicable project ID from the previous step, set the default project to the one in which you want to enable the API	gcloud config set project YOUR_PROJECT_ID
Get a list of services that you can enable in your project	gcloud services list --available

Using the applicable service name from the previous step, enable the service	gcloud services enable SERVICE_NAME
Enable the required Recommender APIs	gcloud services enable recommender.googleapis.com

If you don't see the API listed, that means you haven't been granted access to enable the API.

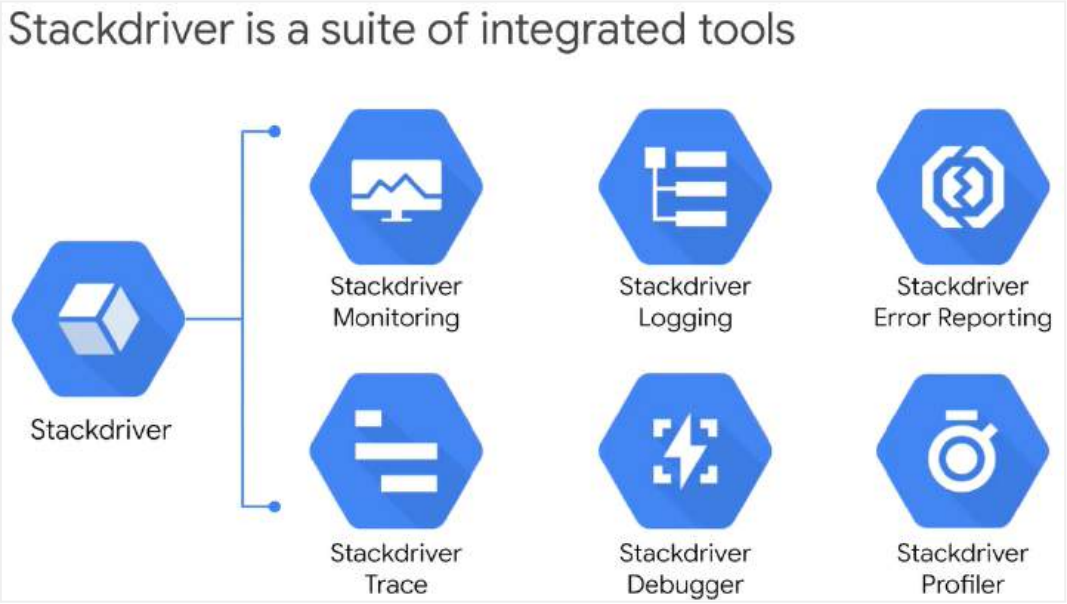
Summary
APIs are software structures that present easy to understand interfaces and remove needless detail.
REST APIs ensure apps can communicate.
Cloud Endpoints is a distributed API management system that helps you create and maintain APIs.
Apigee Edge is a platform for developing and managing API proxies.

f. Provisioning and setting up products in Google Cloud's operations suite

Google Cloud's operations suite (**formerly Stackdriver**) Integrated monitoring, logging, and trace managed services for applications and systems running on Google Cloud and beyond.

- Stackdriver Monitoring offers visibility into web applications and services running in a cloud environment.
- Stackdriver Logging allows you to store, search, analyze, monitor and alert on log data and events.
- Stackdriver Error Reporting counts, analyzes, and aggregates crashes in running cloud services in real time.
- Stackdriver Trace allows you to inspect latency information.
- Stackdriver Debugger allows you to inspect the state of a running application in real time.
- Stackdriver Profiler gathers and analyzes the performance of CPU and memory-intensive functions executed across applications.

Cloud Monitoring provides visibility into the performance, uptime, and overall health of cloud-powered applications. Cloud Monitoring collects metrics, events, and metadata from Google Cloud, Amazon Web Services, hosted uptime probes, application instrumentation, and a variety of common application components including Cassandra, Nginx, Apache Web Server, Elasticsearch, and many others. Cloud Monitoring ingests that data and generates insights via dashboards, charts, and alerts. Cloud Monitoring alerting helps you collaborate by integrating with Slack, PagerDuty, HipChat, Campfire, and more.
Uptime checks verify that a resource is always accessible.



Monitor your infrastructure : Cloud Logging and Cloud Monitoring provide your IT Ops/SRE/DevOps teams with out-of-the box observability needed to monitor your infrastructure and applications. Cloud Logging

automatically ingests Google Cloud audit and platform logs so that you can get started right away. Cloud Monitoring provides a view of all Google Cloud metrics at zero cost and integrates with a variety of providers for non Google Cloud monitoring.

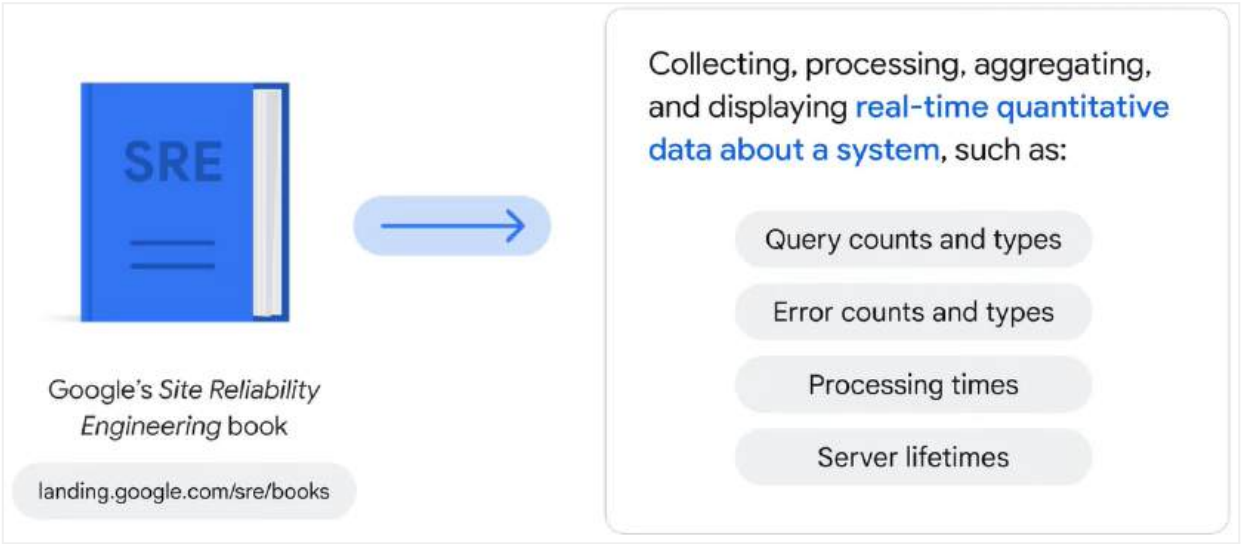
Log management: Log Router allows customers to control where logs are sent. All logs, including audit logs, platform logs, and user logs, are sent to the Cloud Logging API where they pass through the log router. The log router checks each log entry against existing rules to determine which log entries to discard, which to ingest, and which to include in exports.

Proactive monitoring: Cloud Monitoring allows you to create alerting policies to notify you when metrics, health check results, and uptime check results meet specified criteria. Integrated with a wide variety of notification channels, including Slack and PagerDuty.

Performance and cost management: Cloud Profiler provides continuous profiling of resource consumption in your production applications, helping you identify and eliminate potential performance issues. is a statistical, low-overhead profiler that continuously gathers CPU usage and memory-allocation information from your production applications

Monitoring

- Ensure continued system operations
- Uncover trend analyses over time
- Build dashboards
- Alert personnel when systems violate predefined SLOs
- Compare systems and systems changed
- Provide data for improved incident response



Cloud Monitoring

- Provides visibility into the performance, uptime, and overall health of cloud-powered applications
- Collects metrics, events, and metadata from projects, logs, services, systems, agents, custom code, and various common application components including: Cassandra, Nginx, Apache Web Server, Elasticsearch, and many others.
- Ingests that data and generates insights

Cloud Logging

1. Log analysis
    a. Uses Google Cloud's integrated Logs Explorer
    b. Entries can be exported to several destinations
    c. Pub/Sub messages can be analyzed in near-real time using custom code or stream processing
    d. BigQuery allows examination of logging data through SQL queries
    e. Archived log files in Cloud Storage can be analyzed with several tools and techniques
2. Log export
    a. Data can be exported as files to Cloud Storage
    b. Data can be exported as messages through Pub/Sub
    c. Data can be exported into BigQuery tables
    d. Log-based metrics can be created and integrated
3. Log retention
    a. Default log retention depends on the log type
    b. Data access logs are retained by default for 30 days and up to a maximum of 3,650 days
    c. Admin logs are stored by default for 400 days
    d. Logs can be exported to Cloud Storage or BigQuery to extend retention

Cloud Monitoring collects metrics, events, and metadata from Google Cloud, Amazon Web Services (AWS), hosted uptime probes, and application instrumentation. Using the BindPlane service, you can also collect this data from over 150 common application components, on-premise systems, and hybrid cloud systems. Google Cloud's operations suite ingests that data and generates insights via dashboards, charts, and alerts. BindPlane is included with your Google Cloud project at no additional cost.

To collect metrics data from your Compute Engine instances, create an Agent Policy that automatically installs and maintains the Google Cloud's operations suite agents across your fleet of VMs.

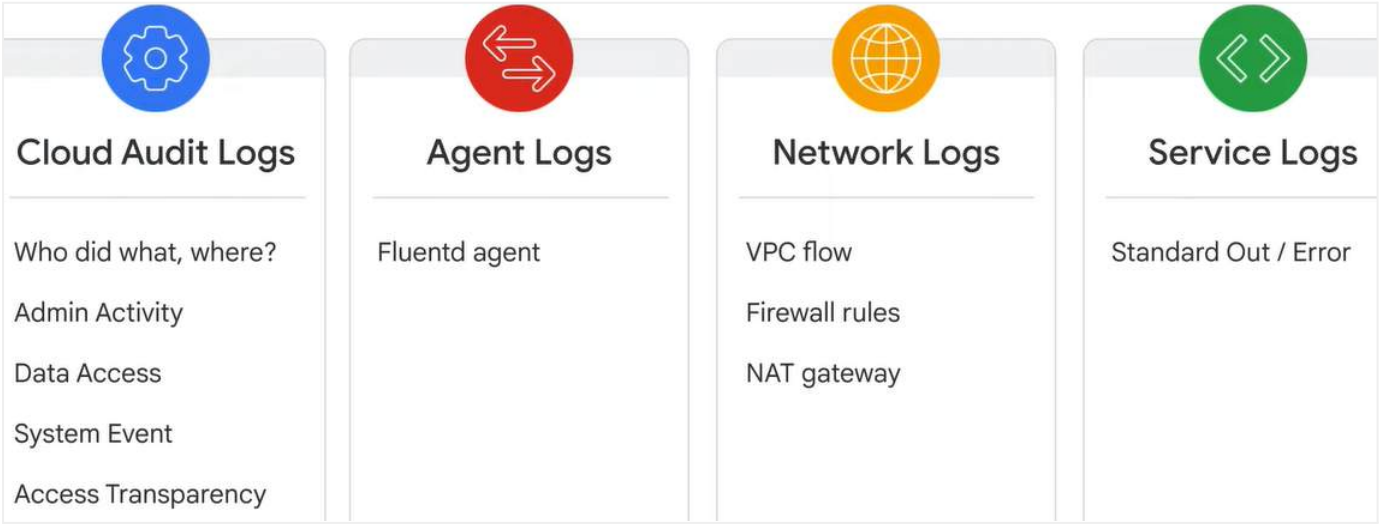
viewing metrics for multiple projects : A *scoping project* hosts a metrics scope. Because every Google Cloud project hosts a metrics scope, every project is also a scoping project. The scoping project stores information about its metrics scope. It also stores the alerts, uptime checks, dashboards, and monitoring groups that you configure for the metrics scope. You can identify the scoping project for a metrics scope as the project selected by the Google Cloud console project picker.

Log entries: The gcloud CLI has a group of commands, gcloud logging, that provide a command-line interface to the Cloud Logging API.

Write a log entry with unstructured data to the log my-test-log:	gcloud logging write my-test-log "A simple entry."
Write a log entry with structured data to the log my-test-log:	gcloud logging write --payload-type=json my-test-log '{ "message": "My second entry", "weather": "partly cloudy"}'

retrieve and display the log entries with a resource type of global	gcloud logging read "resource.type=global"
To delete the log entries you created	gcloud logging logs delete my-test-log

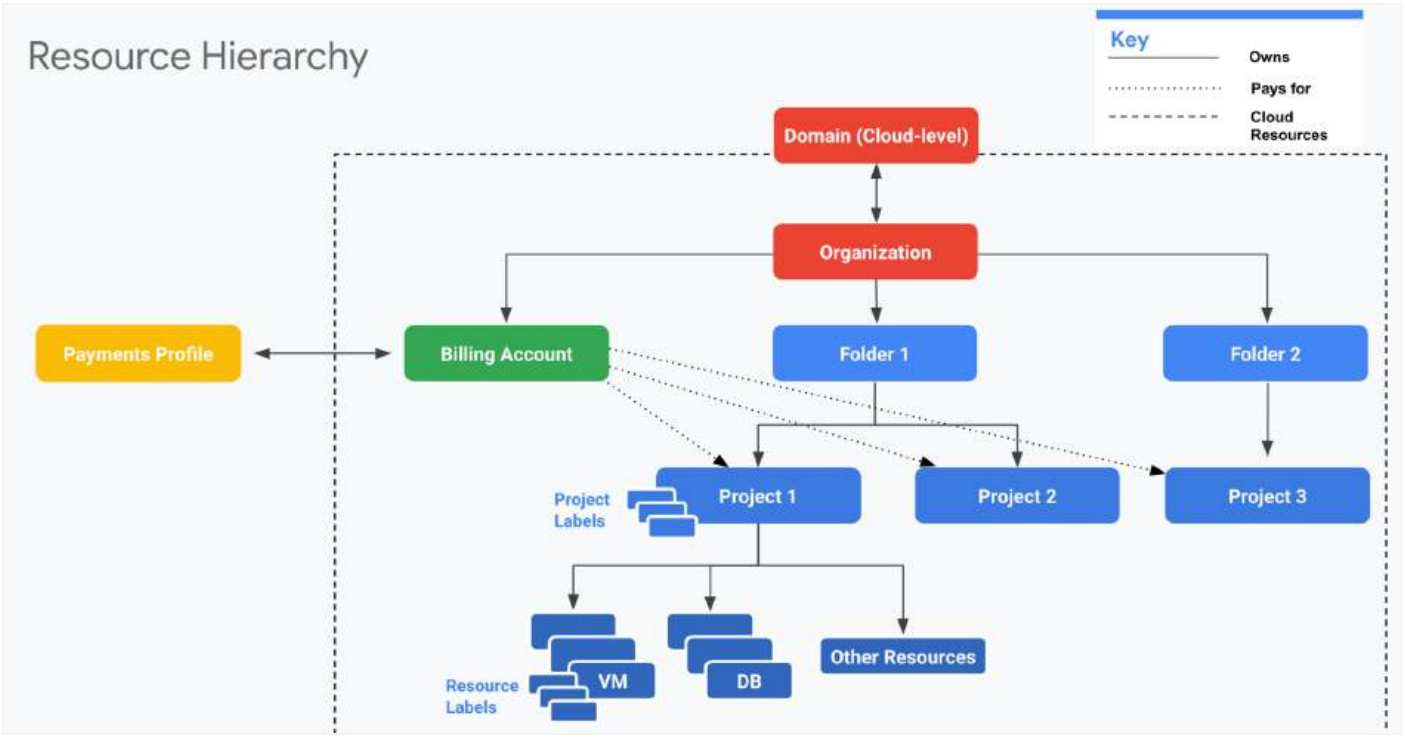
When you write a log entry with structured data, you must include --payload-type=json. If you omit this field, then Logging interprets the payload as unstructured data.



Google Cloud's operations suite provides the following agents for collecting logs from Linux and Windows VM instances. **Ops Agent** & **Legacy Logging agent**

1.2 Managing billing configuration. Activities include:

a. Creating one or more billing accounts



The Domain is linked to either a Google Workspace or Cloud Identity account.You manage the domain-level functionality using the **Google Admin Console (admin.google.com)**.

Organization, you can centrally manage your Google Cloud resources and your users' access to those resources. This includes:

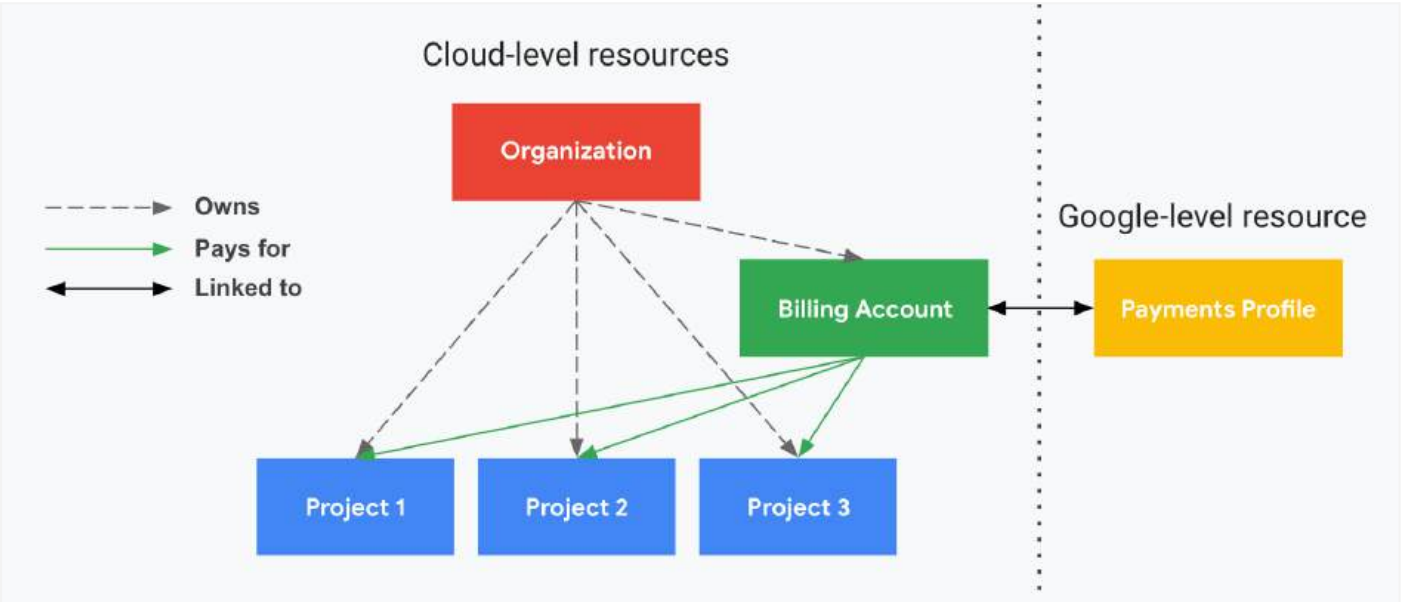
- Proactive management: reorganize resources as needed (for example, restructuring or spinning up a new division may require new projects and folders).
- Reactive management: an Organization resource provides a safety net to regain access to lost resources (for example, if one of your team members loses their access or leaves the company).

Note: Newly created labels can take up to a day to appear in Cloud Billing.

A **Cloud Billing account** is set up in Google Cloud and is used to define who pays for a given set of Google Cloud resources and Google Maps Platform APIs. Access control to a Cloud Billing account is established by IAM roles. A Cloud Billing account is connected to a **Google payments profile**. Your Google payments profile includes a payment instrument to which costs are charged.
Two types of Cloud Billing accounts:

- Self-serve (or Online) account
- Invoiced (or Offline) account

Two types of payments profiles: Individual & Business



Create a new Cloud Billing account : Sign in to the Manage billing accounts page in the Google Cloud console. By default, the person who creates the Cloud Billing account is a Billing Account Administrator for the Cloud Billing account.

b. Linking projects to a billing account

Typically, projects are linked to a billing account at the time that you create the project.

1. select your project.
2. Open the console **Navigation menu** (menu), and then select **Billing**.
3. To enable billing on the project, select **Link a billing account**. Note that you need adequate permissions to link a billing account. See enable billing for an existing project for more information.

View the projects linked to a billing account

1. In the Google Cloud console, go to the **Manage Billing Accounts** page.
2. Select the name of the billing account you want manage.
3. From the Billing navigation menu, click **Account management**.

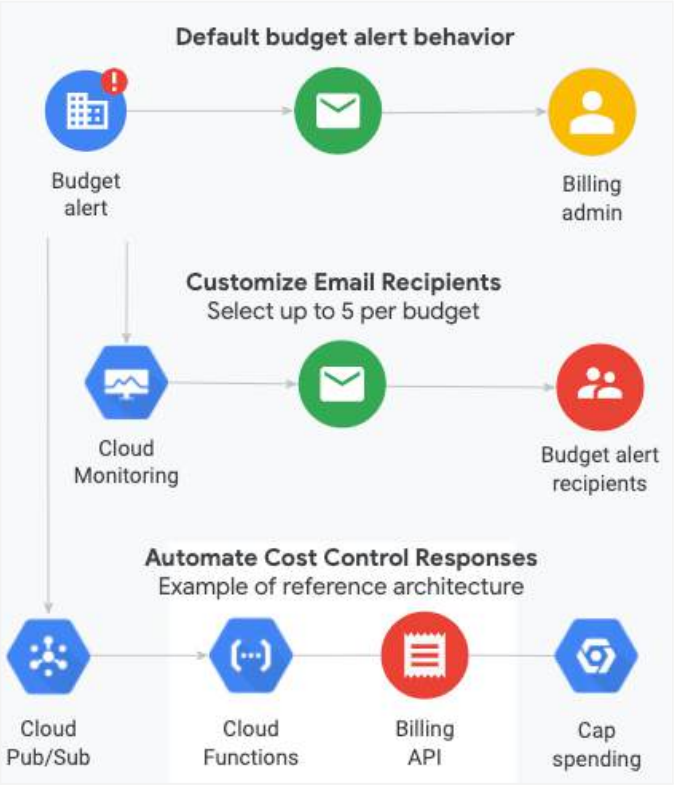
4. On the *Account management* page, linked projects are listed under **Projects linked to this billing account**.

To view the detailed billing information for a project	gcloud beta billing projects describe my-project
To see available IDs	gcloud alpha billing accounts list
To view the details for a Cloud Billing account ACCOUNT_ID is the ID of the billing account	gcloud beta billing accounts describe ACCOUNT_ID

billingEnabled: true and see billingEnabled: false, then the project is linked to an *inactive* Cloud Billing account.

c. Establishing billing budgets and alerts

Avoid surprises on your bill by creating Cloud Billing budgets to monitor all of your Google Cloud charges in one place. A budget enables you to track your actual Google Cloud spend against your planned spend. After you've set a budget amount, you set budget alert threshold rules that are used to trigger email notifications. Budget alert emails help you stay informed about how your spend is tracking against your budget. You can also use budgets to automate cost control responses.



- You can set threshold rules to trigger email alert notifications. When your costs (actual costs or forecasted costs) exceed a percentage of your budget (based on the rules you set), alert emails are sent to the recipients you specify.
- You can specify the recipients of email alerts in these ways:
  - Using the role-based option (default), you can send email alerts to billing admins and users on the Cloud Billing account.
  - Using Cloud Monitoring, you can specify other people in your organization (for example, project managers) to receive budget alert emails.

Tip: Setting a budget does ***not*** automatically cap Google Cloud or Google Maps Platform usage/spending.

- One option to automatically control spending is to use budget notifications to programmatically disable Cloud Billing on a project.
- If you prefer to set a cap on API usage to prevent incurring costs, see Capping API Usage.

ask your administrator to grant you one of the following Cloud Billing IAM roles on your Cloud Billing account:

- Billing Account Administrator
- Billing Account Costs Manager

d. Setting up billing exports

Cloud Billing export to BigQuery enables you to export detailed Google Cloud billing data (such as usage, cost estimates, and pricing data) automatically throughout the day to a BigQuery dataset that you specify.

Then you can access your Cloud Billing data from BigQuery for detailed analysis, or use a tool like Google Data Studio to visualize your data. You can also use this export method to export data to a JSON file.

Analyze billing data with BigQuery To export Cloud Billing data to BigQuery, take the following steps:

- Create a project where the Cloud Billing data will be stored, and enable billing on the project (if you have not already done so).
- Configure permissions on the project and on the Cloud Billing account.
- Enable the BigQuery Data Transfer Service API (required to export your pricing data).
- Create a BigQuery dataset in which to store the data.
- Enable Cloud Billing export of cost data and pricing data to be written into the dataset.

1.3 Installing and configuring the command line interface (CLI), specifically the Cloud SDK (e.g., setting the default project).

Cloud SDK (Libraries and tools for interacting with Google Cloud products and services.) provides language-specific Cloud Client Libraries supporting each language's natural conventions and styles. This makes it easier for you to interact with Google Cloud APIs in your language of choice. Client libraries also handle authentication, reduce the amount of necessary boilerplate code, and provide helper functions for pagination of large datasets and asynchronous handling of long-running operations.

To launch a **Cloud Shell** session from console, click **Activate Cloud Shell** in the Google Cloud console. **Cloud Shell** is a Debian-based virtual machine with a persistent 5-GB home directory, which makes it easy for you to manage your Google Cloud projects and resources. The gcloud command-line tool and other utilities you need are pre-installed in Cloud Shell, which allows you to get up and running quickly.
you will use your $HOME directory, which is used in persistent disk storage to store files across projects and between Cloud Shell sessions. Your $HOME directory is private to you and cannot be accessed by other users. Can launch a Cloud Shell session with shell.cloud.google.com.
Cloud Shell Editor, you can use ide.cloud.google.com.

ID	Name	Description
gcloud	Default gcloud CLI commands	Tool for interacting with Google Cloud. Only commands at the General Availability and Preview release levels are installed with this component. You must separately install the gcloud alpha Commands and/or gcloud beta Commands components if you want to use commands at other release levels.
bq	BigQuery command-line tool	Tool for working with data in BigQuery
gsutil	Cloud Storage command-line tool	Tool for performing tasks related to Cloud Storage.
core	gcloud CLI core libraries	Libraries used internally by the gcloud CLI tools.

Install the gcloud CLI : Download the Google Cloud CLI installer.
Alternatively, open a PowerShell terminal and run the following PowerShell commands:

```
(New-Object
Net.WebClient).DownloadFile("https://dl.google.com/dl/cloudsdk/channels/rapid/GoogleCloudSDKInstaller
.exe", "$env:Temp\GoogleCloudSDKInstaller.exe")

& $env:Temp\GoogleCloudSDKInstaller.exe
```

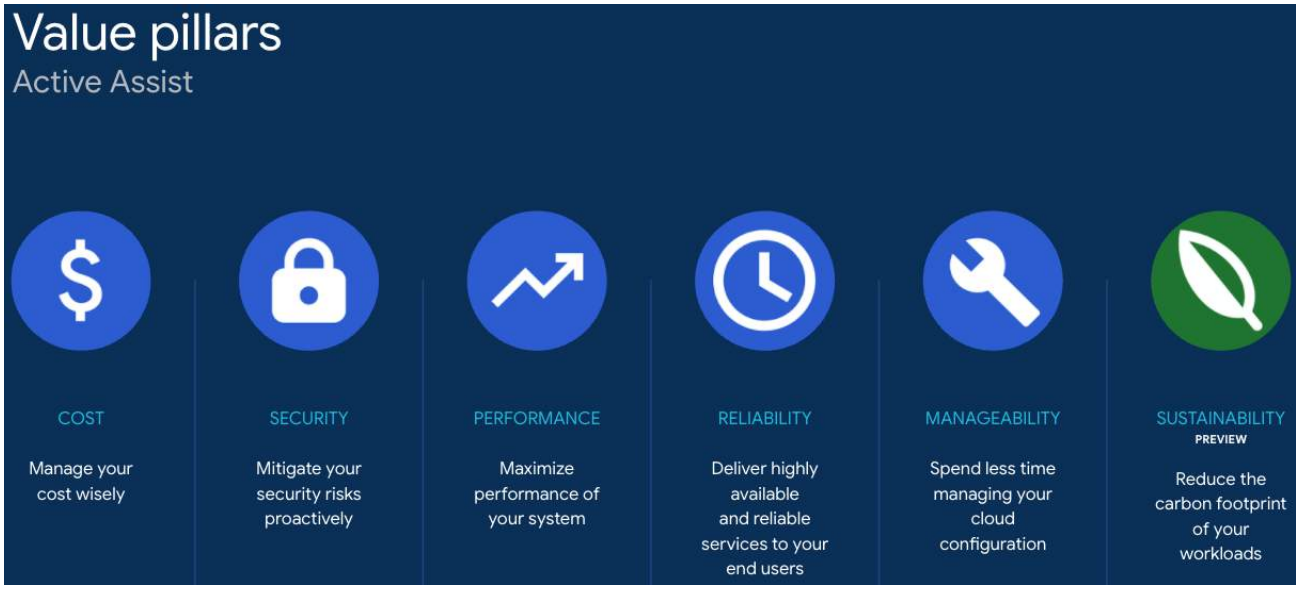
Set up default Google Cloud CLI settings for your project by using one of the following methods:

- Use gcloud init, if you want to be walked through setting project defaults.
- Use gcloud config, to individually set your project ID, zone, and region.

Section 2. Planning and configuring a cloud solution

2.1 Planning and estimating Google Cloud product use using the Pricing Calculator

Cost Management Tools for monitoring, controlling, and optimizing your costs.View intelligent recommendations for optimizing your costs and usage. Easily apply these changes for immediate cost savings and greater efficiency.

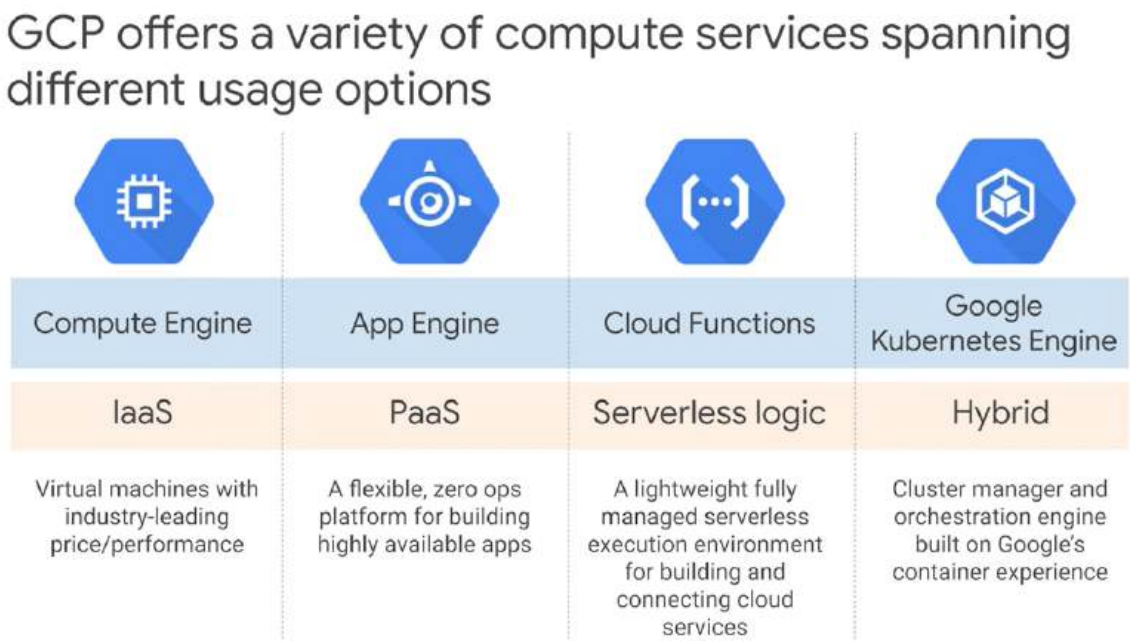


Active Assist refers to the portfolio of tools used in Google Cloud to generate insights and recommendations to help you optimize your cloud projects. This includes recommenders that generate recommendations and insights and analysis tools.

2.2 Planning and configuring compute resources. Considerations include:

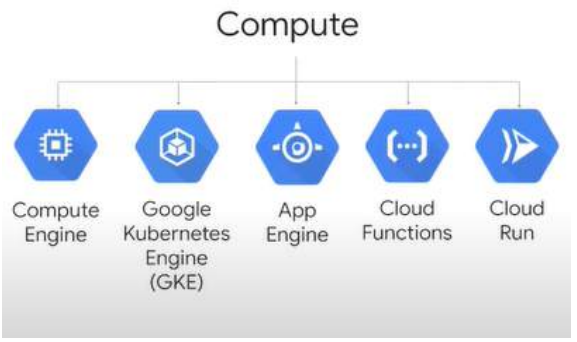
a. Selecting appropriate compute choices for a given workload (e.g., Compute Engine, Google Kubernetes Engine, Cloud Run, Cloud Functions)

Compute Engine lets you create and run virtual machines on Google infrastructure. Compute Engine offers scale, performance, and value that allows you to easily launch large compute clusters on Google's infrastructure.

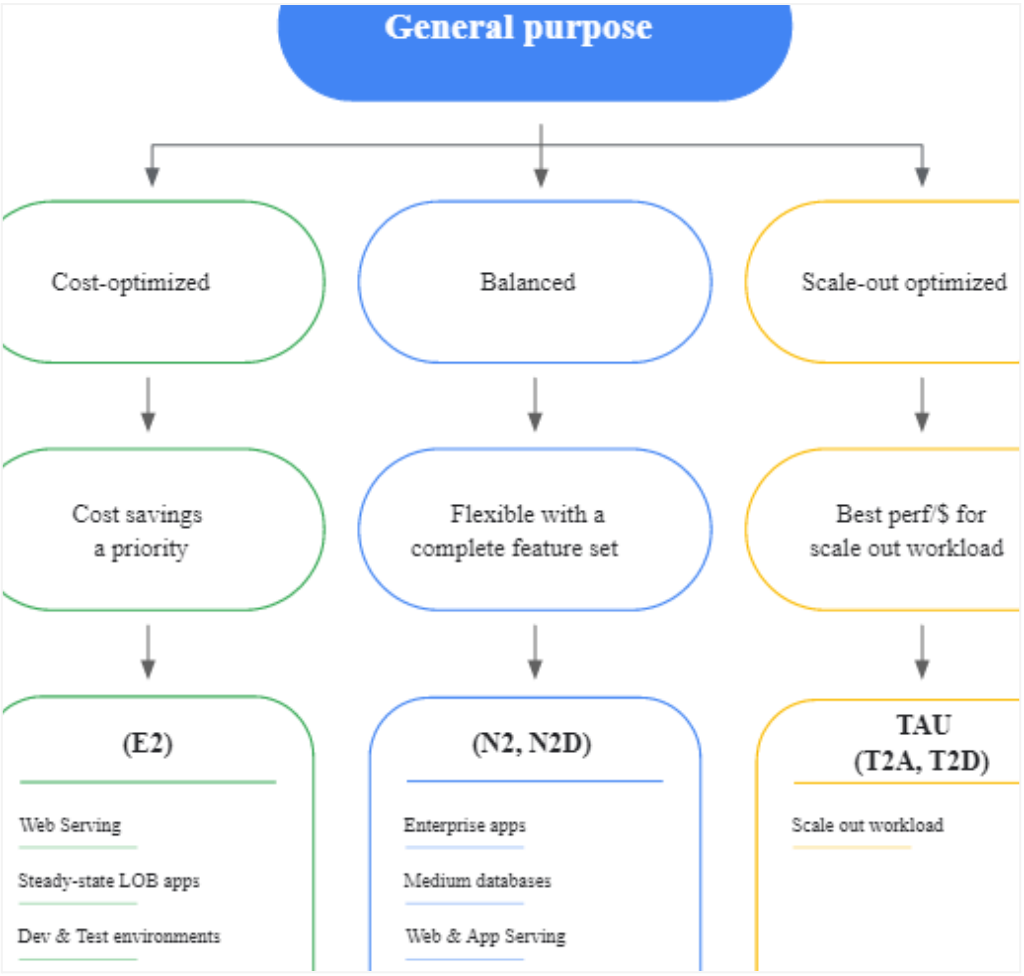


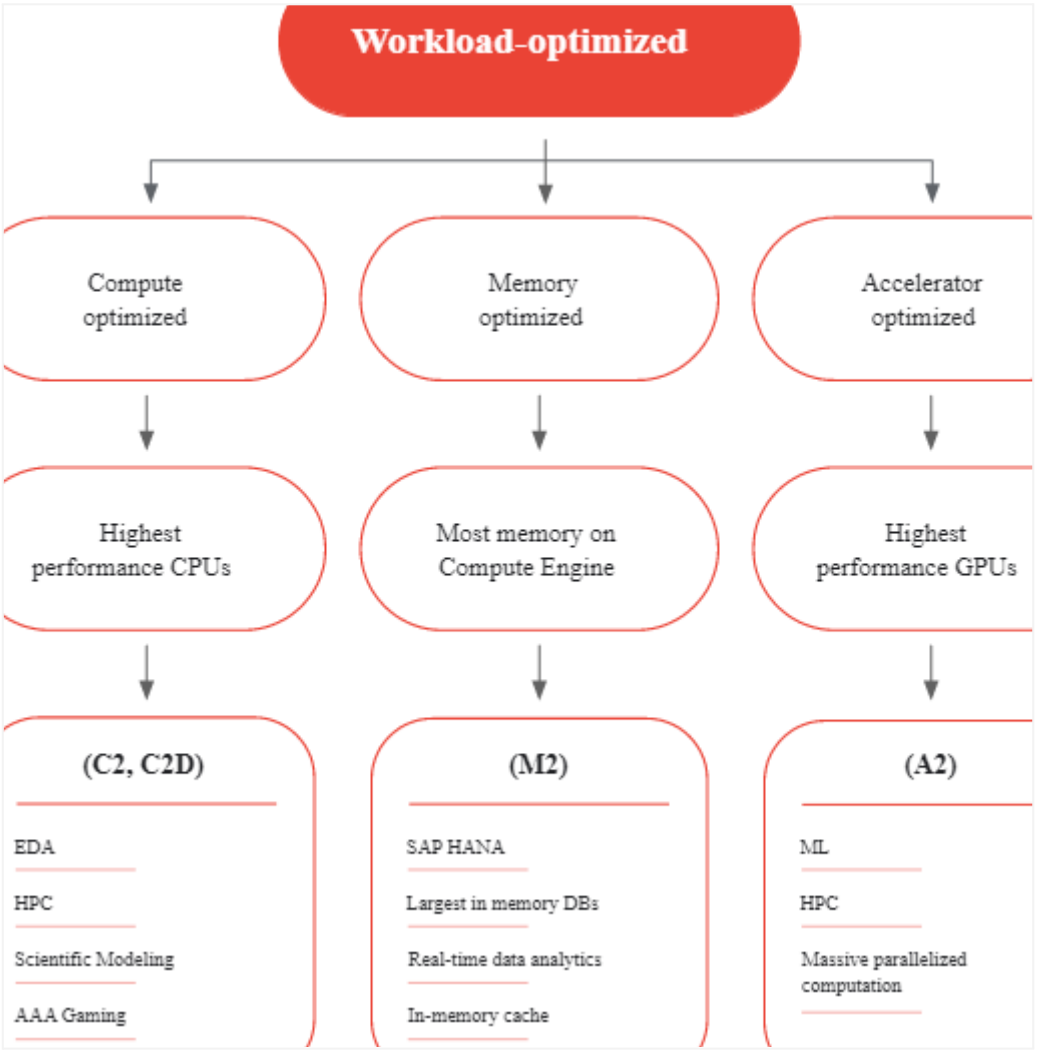
You can run your Windows applications on Compute Engine and take advantage of many benefits available to virtual machine instances, such as reliable storage options, the speed of the Google network, and Autoscaling.

Choosing the right virtual machine type : Whether you're new to cloud computing, or just getting started on Google Cloud, these recommendations can help you optimize your Compute Engine usage and benefits. The table provides machine type recommendations for different workloads.



Resources that live in a zone are referred to as zonal resources. Virtual machine instances and persistent disks live in a zone. If you want to attach a persistent disk to a virtual machine instance, both resources must be in the same zone. Similarly, if you want to assign a static IP address to an instance, the instance must be in the same region as the static IP address.


General purpose
Cost-optimized
Balanced
Scale-out optimized
Cost savings a priority
Flexible with a complete feature set
Best perf/$ for scale out workload
(E2)
Web Serving
Steady-state LOB apps
Dev & Test environments
(N2, N2D)
Enterprise apps
Medium databases
Web & App Serving
TAU (T2A, T2D)
Scale out workload


Workload-optimized
Compute optimized
Memory optimized
Accelerator optimized
Highest performance CPUs
Most memory on Compute Engine
Highest performance GPUs
(C2, C2D)
EDA
HPC
Scientific Modeling
AAA Gaming
(M2)
SAP HANA
Largest in memory DBs
Real-time data analytics
In-memory cache
(A2)
ML
HPC
Massive parallelized computation

Kubernetes:

- Open-source platform for managing containerized workloads and services
- Makes it easy to orchestrate many containers on many hosts, scale them as microservices, and deploy rollouts and rollbacks
- Is a set of APIs to deploy containers on a set of nodes called a cluster
- 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 figures how to make that happen

Advanced cluster management features include:

- Google Cloud load-balancing for Compute Engine instances
- Node pools to designate subsets of nodes within a cluster
- Automatic scaling of your cluster's node instance count
- Automatic upgrades for your cluster's node software
- Node auto-repair to maintain node health and availability
- Logging and monitoring with Google Cloud's operations suite

Google Kubernetes Engine The most automated and scalable managed **Kubernetes** platform

GKE now offers two modes of operations: **Autopilot** and Standard. Autopilot mode is a hands-off, fully managed solution that manages your entire cluster's infrastructure without worrying about configuring and monitoring, while still delivering a complete Kubernetes experience. And with per-pod billing, Autopilot ensures you pay only for your running pods, not system components, operating system overhead, or unallocated capacity. Standard mode is the experience we've been building since the launch of GKE, enabling additional customization options over the nodes with the ability to fine tune and run custom administrative workloads for when you need low level controls.

Pod and cluster autoscaling: GKE is the industry's first fully managed Kubernetes service that implements full Kubernetes API, 4-way autoscaling, release channels and multi-cluster support. Horizontal pod autoscaling can be based on CPU utilization or custom metrics. Autopilot automatically scales your cluster capacity based on the resource requirements in your Pod specs. In the Standard mode of operation, cluster autoscaling works on a per-node-pool basis to scale up your nodes on demand. Vertical pod autoscaling continuously analyzes the CPU and memory usage of pods, automatically adjusting CPU and memory requests.

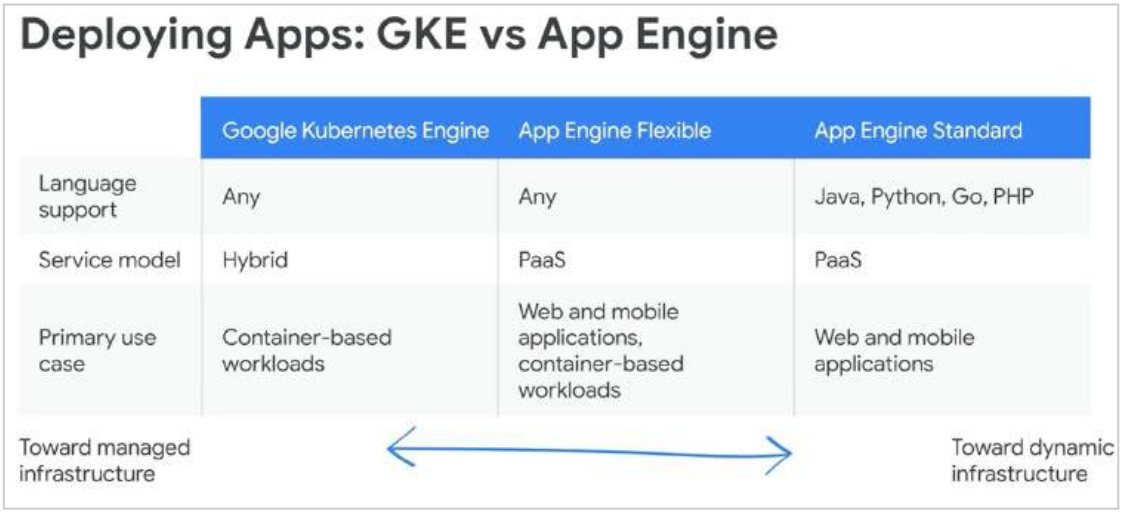
Cluster configuration choices

Cluster choices	Mode	
	Autopilot	**Standard**
Availability type	Regional	Regional or Zonal
Version	Release channel	Release channel, Default, or Specific
Network routing	VPC-native	VPC-native or Routes-based
Network isolation	Private or Public	Private or Public
Kubernetes features	Production	Production or Alpha

New features in Kubernetes are listed as **Alpha**, **Beta**, or **Stable**, depending upon their status in development. In most cases, Kubernetes features that are listed as **Beta** or **Stable** are included with GKE. Kubernetes **Alpha** features are available in special GKE alpha clusters.

Note: Use Migrate for GKE to containerize existing VM-based applications to run on GKE. Migrate for GKE provides a simple way to migrate your existing applications to containers without requiring access to your source code or rewriting the applications.

You can use Cloud Build to build container images (such as Docker) from a variety of source code repositories, and Artifact Registry or Container Registry to store and serve your container images.
Billing : Pay per Pod resource requests (CPU, memory, and ephemeral storage) for Autopilot while in Standard Mode Pay per node (CPU, memory, boot disk)
Google automatically upgrades the cluster and its nodes when an update is available in that release channel. The Rapid channel receives multiple updates a month, while the Stable channel only receives a few updates a year.
If you know that you need to use a specific supported version of Kubernetes for a given workload, you can specify it when creating the cluster.If you do not need to control the specific patch version you use, consider enrolling your cluster in a release channel instead of managing its version directly.
A cluster that uses Alias IPs is called a *VPC-native cluster*. A cluster that uses Google Cloud routes is called a *routes-based cluster*.VPC-native is the recommended network mode for new clusters. This is the default for clusters created in the Autopilot mode.



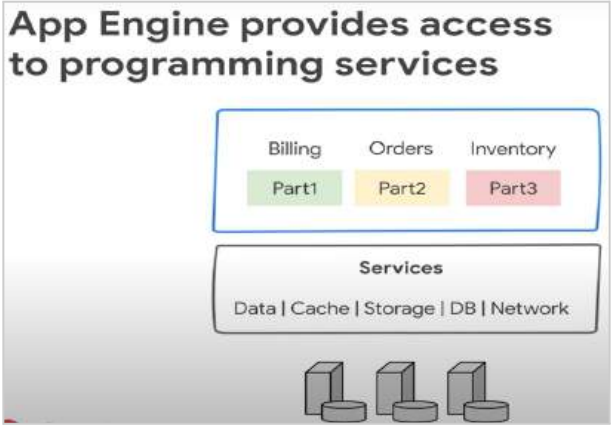
App Engine is a PaaS for building scalable applications

- App Engine makes deployment, maintenance, and scalability easy so you can focus on innovation
- Especially suited for building scalable web applications and mobile backends

App Engine

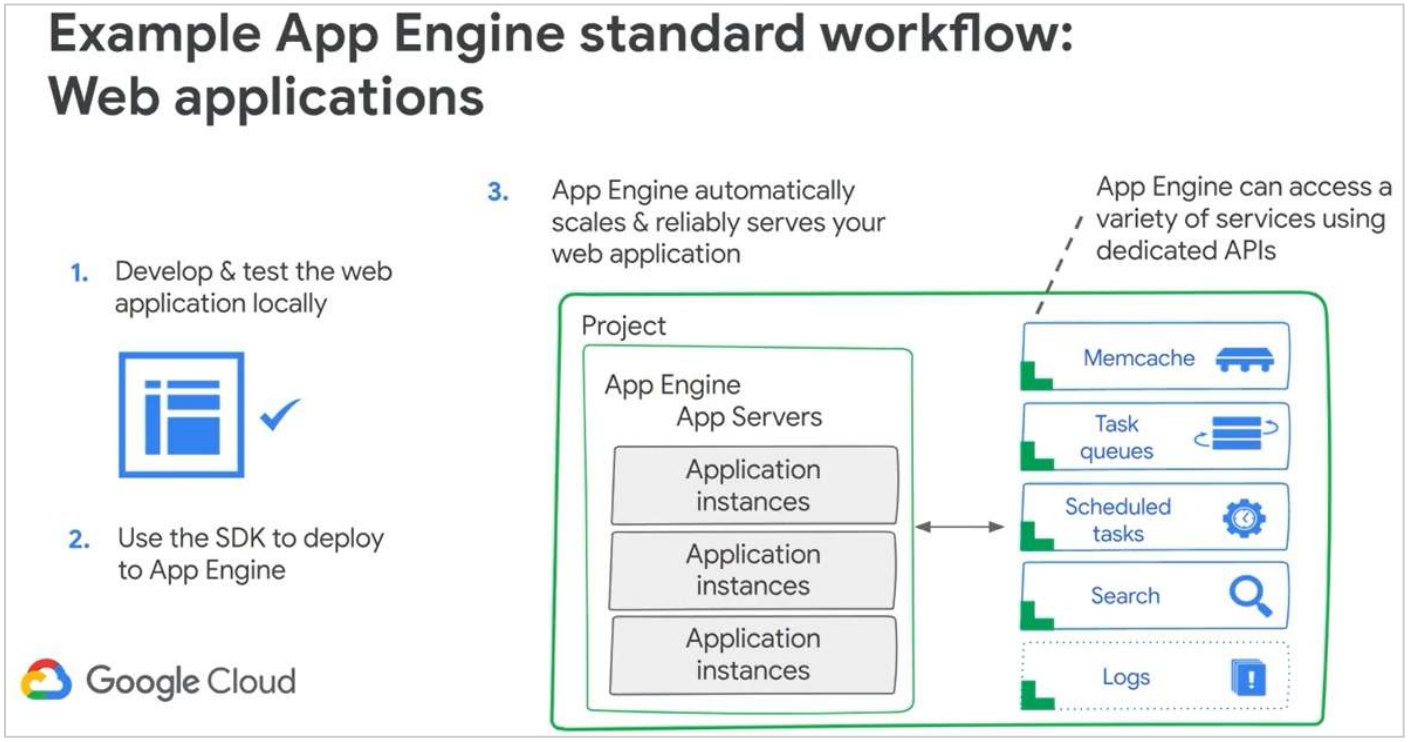
- Access to programming services to write your code
- Seamless, independent and rapid scaling
- No fine-tuning the underlying architecture to save cost

Another traditional serverless application manager available in Google Cloud is **App Engine**. App Engine has two management environments: standard and flexible.
In the standard environment, apps run in a sandbox using a specific language runtime. Standard environment is good for rapid scaling. It is limited to specific languages. It can scale to 0 when there is no incoming traffic. It starts up in seconds.
In the standard environment you are not allowed to make changes to the runtime. In contrast to the standard environment, App Engine flexible runs in Docker containers in Compute Engine VMs. Flexible supports more programming languages. It can use native code and you can access and manage the underlying Compute Engine resource base.
App Engine flexible does not scale to 0. Startup is in minutes. Deployment time is in minutes (longer than standard). It does allow you to modify the runtime environment.
Concurrency is how many users can connect to a particular instance. It does not directly affect connections to backend services.



App Engine standard environment

- Easily deploy your applications
- Autoscale workloads to meet demand
- Economical: Free daily quota, Usage based pricing
- SDKs for development, testing and deployment



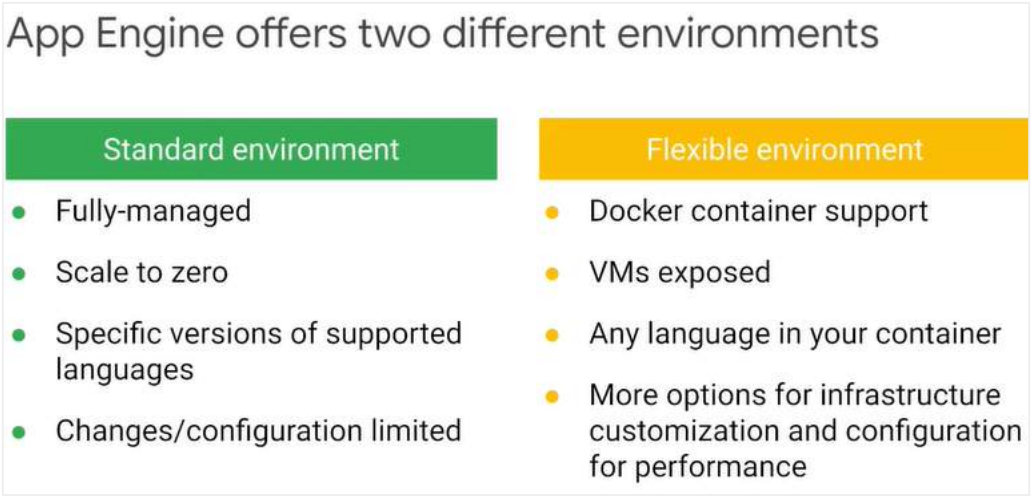
App Engine flexible environment

- Build and deploy containerized apps with a click
- No sandbox constraints
- Can access App Engine resources
- Standard runtimes: Python, Java, Go, Node.js
- Custom runtime support: Any language that supports HTTP requests
- Package your runtime as a Dockerfile

	Standard environment	Flexible environment
Instance startup	Seconds	Minutes
SSH access	No	Yes (although not by default)
Write to local disk	No (some runtimes have read and write access to the /tmp directory)	Yes, ephemeral (disk initialized on each VM startup)
Support for 3rd-party binaries	For certain languages	Yes
Network access	Via App Engine services	Yes
Pricing model	After free tier usage, pay per instance class, with automatic shutdown	Pay for resource allocation per hour; no automatic shutdown

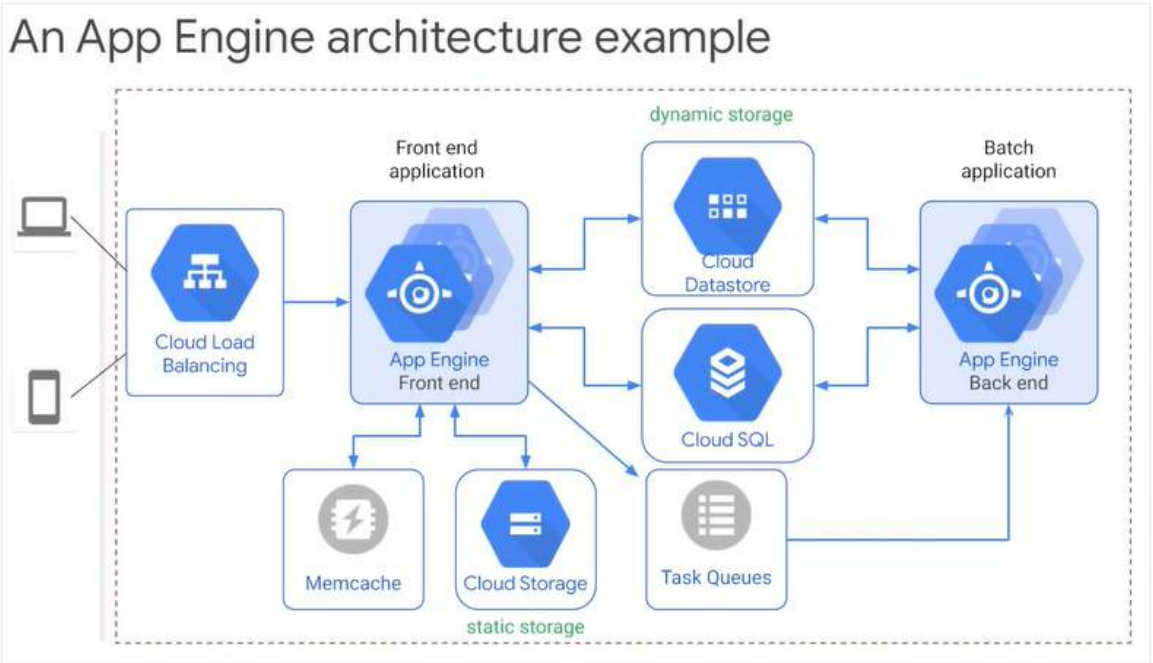
App Engine's standard environment is for people who want the service to take maximum control of their web and mobile applications deployment and scaling. Google Kubernetes Engine, however, gives the application owner the full flexibility of Kubernetes. App Engine's flexible environment is somewhere between the two.

App Engine is a fully managed, serverless platform for developing and hosting web applications at scale.
Features : Built-in services and APIs, NoSQL datastores, Health checks, Memcashe, Application logging, Load balancing, User authentication API
App Engine environments



Flexible environment :

- Instances are health-checked, healed, and co-located
- Critical, backward-compatible updates are automatically applied to the underlying operating system
- VM instances are automatically located by geographical region according to the settings in your project
- VM instances are restarted on a weekly basis



The App Engine standard environment is based on container instances running on Google's infrastructure. Containers are preconfigured with one of several available runtimes (Java 7, Java 8, Python 2.7, Go and PHP). Each runtime also includes libraries that support App Engine Standard APIs. For many applications, the standard environment runtimes and libraries might be all you need.
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.

Applications run in a secure, sandboxed environment, allowing App Engine standard environment to distribute requests across multiple servers, and scaling servers to meet traffic demands. Your application runs within its own secure, reliable environment that is independent of the hardware, operating system, or physical location of the server.

copy the Hello World sample app repository	git clone https://github.com/GoogleCloudPlatform/python-docs-samples.git
Go to the directory	cd python-docs-samples/appengine/standard_python3/hello_world
Start Google Cloud development server (dev_appserver.py)	dev_appserver.py app.yaml
To deploy your app to App Engine prompted to choose the region	gcloud app deploy
To launch your browser	gcloud app browse

Identity-Aware Proxy (IAP) is a Google Cloud service that intercepts web requests sent to your application, authenticates the user making the request using the Google Identity Service, and only lets the requests through if they come from a user you authorize. In addition, it can modify the request headers to include information about the authenticated user.

Restrict access with IAP

1. In the cloud console window, click the **Navigation menu** ☰ > **Security** > **Identity-Aware Proxy**.
2. Click **ENABLE API**.
3. Click **GO TO IDENTITY-AWARE PROXY**.
4. Click **CONFIGURE CONSENT SCREEN**.
5. Select **Internal** under User Type and click **Create**

Note: App Engine has its standard and flexible environments which are optimized for different application architectures. Currently, when enabling IAP for App Engine, if the Flex API is enabled, Google Cloud will look for a Flex Service Account. Your lab project comes with a multitude of APIs already enabled for the purpose of convenience. However, this creates a unique situation where the Flex API is enabled without a Service Account created.

CMD : gcloud services disable appengineflex.googleapis.com

If there is a risk of IAP being turned off or bypassed, your app can check to make sure the identity information it receives is valid. This uses a third web request header added by IAP, called `X-Goog-IAP-JWT-Assertion`. The value of the header is a cryptographically signed object that also contains the user identity data. Your application can verify the digital signature and use the data provided in this object to be certain that it was provided by IAP without alteration.

The `assertion` is the cryptographically signed data provided in the specified request header. The code uses a library to validate and decode that data. Validation uses the public keys that Google provides for checking data it signs, and knowing the audience that the data was prepared for (essentially, the Google Cloud project that is being protected). Helper functions `keys()` and `audience()` gather and return those values.

The signed object has two pieces of data we need: the verified email address, and the unique ID value (provided in the `sub`, for subscriber, standard field).

If IAP is turned off or bypassed, the verified data would either be missing, or invalid, since it cannot have a valid signature unless it was created by the holder of Google's private keys.

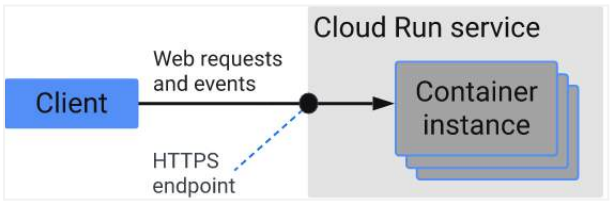
Cloud Run and Cloud Functions are Google's serverless approach to handling containers and functional code. In both of these technologies, you pay for resources based on how many requests are coming in. One big difference between the two of them is that Cloud Run is optimized for multiple concurrent connections to each instance, while Cloud Functions only lets you have only one connection per function instance.

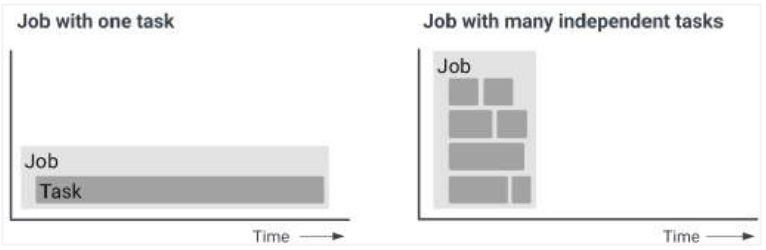
Cloud Run is a managed compute platform that lets you run containers directly on top of Google's scalable infrastructure.You don't have to create a cluster or manage infrastructure in order to be productive with Cloud Run.

- A managed compute platform that can run stateless containers
- Serverless, removing the need for infrastructure management
- Built on Knative, an open API and runtime environment built on Kubernetes
- Can automatically scale up and down from zero almost instantaneously, charging only for the resources used

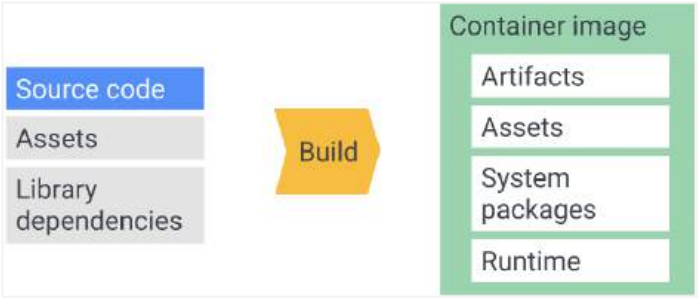
Cloud Run then starts your container on demand to handle requests, and ensures that all incoming requests are handled by dynamically adding and removing containers

- **Cloud Run services.** Used to run code that responds to web requests, or events.
- **Cloud Run jobs.** Used to run code that performs work (a job) and quits when the work is done. However, you can also start many identical, independent container instances in parallel, that is, an array job. Array jobs are a faster way to process jobs that can be split into multiple independent tasks





A container image is a package with everything your service needs to run.
If you're concerned about costs or overloading downstream systems, you can limit the maximum number of instances.To make sure your service doesn't scale to zero instances, you can configure Cloud Run to keep a minimum amount of container instances active.



1. **Trap termination signal (SIGTERM)** : Cloud Run sends a SIGTERM signal to your container instance before the container instance terminates.
2. **CPU allocation (services)** : By default, Cloud Run container instances are only allocated CPU during request processing, container startup and shutdown.

Cloud Run capabilities:

- Serverless Container management
- Based on a service resource
- A service exposes an endpoint: 1.Regional 2.Replicated across zones
- Scales based on incoming requests

1. Cloud Run provides a service to manage containers in a serverless way, which means you don't need to manage infrastructure when you deploy an application. The main resource container for Cloud Run is a service. A service is a regional resource that is replicated in multiple zones and exposed as

an endpoint. Underlying infrastructure scales automatically based on incoming requests. If there are no requests coming in it can scale to zero to save you money.

2. Changes to containers or environment settings create new revisions. Revisions can be rolled out in a way that supports canary testing by splitting traffic according to your specifications.
3. Cloud Run is built using an open source initiative called knative. It is billed to the nearest 100 MS as you deploy containers to it.
4. Cloud Run can use system libraries and tools made available to the container environment. It has a timeout of 60 minutes for longer running requests. Cloud Run can send multiple concurrent requests to each container instance, improving latency, and saving costs for large volumes of incoming traffic.

Google Cloud services that work well with Cloud Run Cloud Run uses a default runtime service account that has the *Project > Editor* role, which means it is able to call all Google Cloud APIs and have read and write access on all resources in your Google Cloud project.

Data Storage

Service	Description
Firestore	Fully managed NoSQL database.
Cloud Spanner	Fully managed, scalable, relational database.
Cloud SQL	Fully managed relational database. Refer to Connecting to Cloud SQL instances.
Cloud Storage	Object storage. Store objects and serve static content.
Memorystore	Fully managed in-memory data store service. Connect to your VPC network to access Memorystore instances. Refer to Connecting to a Redis instance from a Cloud Run service.
BigQuery	Fully managed cloud data warehouse for analytics.
Secret Manager	Create and access secrets.
Filestore	Fully managed NFS file servers on Google Cloud

Networking

Service	Description
Virtual Private Cloud	Managed networking functionality for your Google Cloud resources. Refer to Connecting to a VPC network.
Cloud Load Balancing	Use serverless NEGs to configure a Cloud Run backend for an external HTTP(S) load balancer.
Google Cloud Armor	Helps protect your applications and websites against denial of service and web attacks.
Cloud CDN	Cloud CDN is supported via HTTP(S) Load Balancing.
API Gateway	Fully managed API management including routing, authentication, API keys, rate limiting, and quota.

Orchestration

Service	Description
Pub/Sub	Push events to Cloud Run services. Refer to the Using Pub/Sub with Cloud Run Tutorial.
Cloud Scheduler	Trigger Cloud Run services on a schedule.

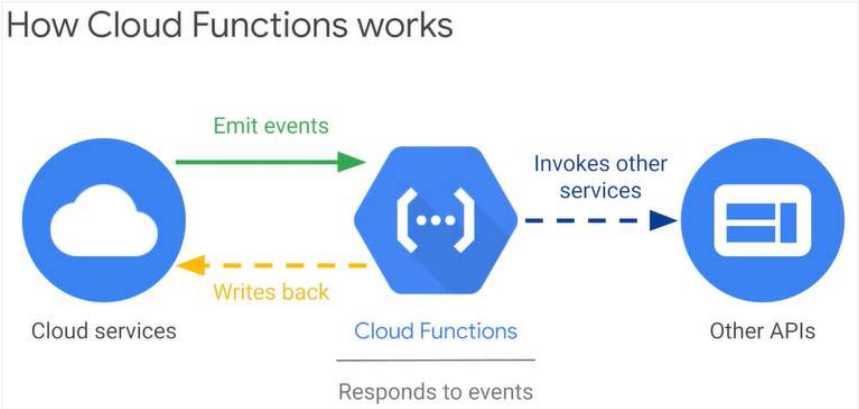
Cloud Tasks	Execute asynchronous tasks on Cloud Run. Refer to HTTP Target tasks with authentication tokens.
Workflows	Orchestrate and automate Cloud Run services.

Web-apps

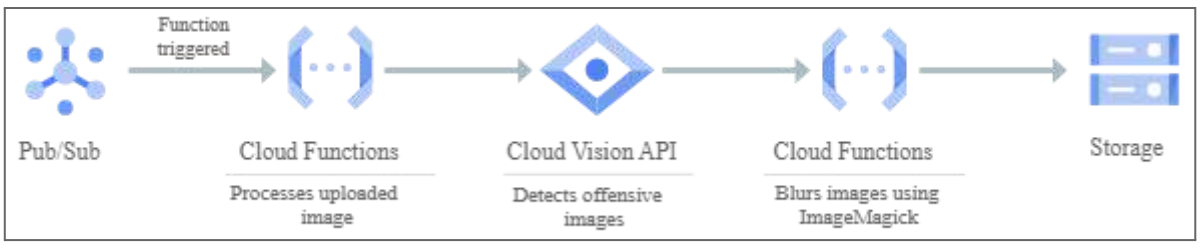
Service	Description
Identity Platform	Login your users.
Firebase Hosting	Fully managed hosting service for static and dynamic content with configurable CDN caching.

Tools

Service	Description
Cloud Build	Build container images, continuous integration and delivery.
Artifact Registry	Store container images.
Google Cloud's operations suite	Monitoring and logging of Cloud Run services.



Cloud Function Run your code in the cloud with no servers or containers to manage. **Cloud Functions** is a scalable, pay-as-you-go functions as a service (FaaS) product to help you build and connect event driven services with simple, single purpose code.
Cloud Functions is a serverless execution environment for building and connecting cloud services. With Cloud Functions you write simple, single-purpose functions that are attached to events emitted from your cloud infrastructure and services. Cloud events are *things* that happen in your cloud environment.
Your Cloud Function is triggered when an event being watched is fired. Your code executes in a fully managed environment. There is no need to provision any infrastructure or worry about managing any servers.



Cloud Functions provides a connective layer of logic that lets you write code to connect and extend cloud services.Cloud events are *things* that happen in your cloud environment. Events occur whether or not you choose to respond to them. You create a response to an event with a *trigger*. A trigger is a declaration that you are interested in a certain event or set of events. Binding a function to a trigger allows you to capture and act on events. When deploying a new function, you must specify `--trigger-topic`, `--trigger-bucket`, or `--trigger-http`.

A background function is one type of event-driven function. It is supported for Node.js, Go, Python, and Java.

to create a new cloud storage bucket	gsutil mb -p [PROJECT_ID] gs://[BUCKET_NAME]
Deploy the function to a pub/sub topic named **hello_world**	gcloud functions deploy helloWorld \ --stage-bucket [BUCKET_NAME] \ --trigger-topic hello_world \ --runtime nodejs8
status of the function	gcloud functions describe helloWorld
Check the logs to see your messages in the log history	gcloud functions logs read helloWorld

Use case	**Description**
Data processing / ETL	Listen and respond to Cloud Storage events such as when a file is created, changed, or removed. Process images, perform video transcoding, validate and transform data, and invoke any service on the internet from your Cloud Functions.
Webhooks	Via a simple HTTP trigger, respond to events originating from 3rd party systems like GitHub, Slack, Stripe, or from anywhere that can send HTTP requests.
Lightweight APIs	Compose applications from lightweight, loosely coupled bits of logic that are quick to build and that scale instantly. Your functions can be event-driven or invoked directly over HTTP/S.
Mobile backend	Use Google's mobile platform for app developers, Firebase, and write your mobile backend in Cloud Functions. Listen and respond to events from Firebase Analytics, Realtime Database, Authentication, and Storage.
IoT	Imagine tens or hundreds of thousands of devices streaming data into Pub/Sub, thereby launching Cloud Functions to process, transform and store data. Cloud Functions lets you do it in a way that's completely serverless.

To enable automatic management and scaling of your functions, functions must be *stateless*—one function. Each instance of a function handles only one concurrent request at a time.Because concurrent requests are processed by different function instances, they do not share variables or local memory. See Statelessness and Function instance lifespan for more information. Requests that include function instance startup, called *cold starts*, can be slower than requests routed to existing function instances. A function has access to its allocated resources (memory and CPU) only for the duration of function execution. Function execution is also subject to the timeout duration of the function.Each function has a certain amount of memory allocated for its use. Every deployed function is isolated from all other functions—even those deployed from the same source file. In particular, they do not share memory, global variables, file systems, or other state. To share data across deployed functions, you can use services such as Memorystore, Datastore, Firestore, or Cloud Storage.

Cloud Functions capabilities:

1. Serverless function execution
2. Event based
3. Functions trigger when an event occurs
4. Scales by number of events received

5. Functions are stateless - need to persist data if you need to share it outside the function

- Cloud Functions is Google Cloud's answer to serverless functions. It is a fully managed service based on events that happen across your cloud environment, including services and infrastructure. The functions you develop run in response to those events. There are no servers to manage or scaling to configure. The service provides the underlying resources required to execute your function.
- A trigger sends an https request to an endpoint the service is listening on. This endpoint then responds by deploying and executing the function and returning the results specified in your code. Pricing is based on the number of events, compute time, and memory required in network ingress/egress. If no requests are coming in, your function doesn't cost anything.
- Cloud Functions use cases include IoT processing and lightweight ETL.
- Depending on the programming language you choose, the Cloud Functions service provides a base image and runtime that will be updated and patched automatically. This helps keeps execution of your deployed functions secure.
- By design, functions you write for use by the Cloud Functions service are stateless. If you need to share and persist data across function runs, you should consider using Datastore or Cloud Storage. Each Cloud Function instance handles only one concurrent request at a time. If, while handling a request, another one comes in, Cloud Functions will ask for more instances to be created. This is another reason functions need to be stateless, because they can run on different instances. You can implement minimum instance limits to avoid latency associated with cold starts.

Cloud Functions

- Create single-purpose functions that respond to events without a server or runtime
    - Event examples: New instance created, file added to Cloud Storage
- Written in Javascript (Node.js), Python or Go; execute in managed Node.js environment on Google Cloud.

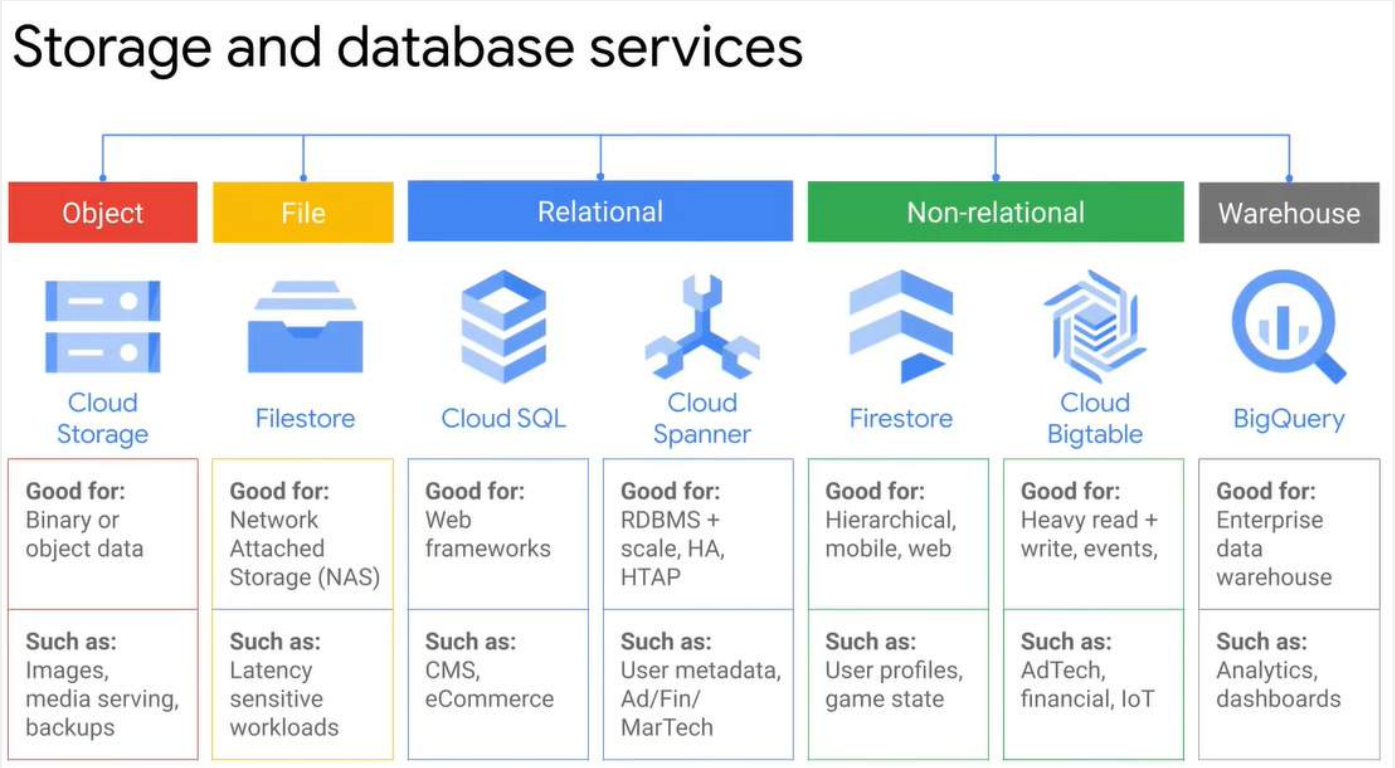
b. Using preemptible VMs and custom machine types as appropriate

Important: Spot VMs are the latest version of **preemptible VMs**. New and existing preemptible VMs continue to be supported, and preemptible VMs use the same pricing model as Spot VMs. However, Spot VMs provide new features that preemptible VMs do not support. For example, preemptible VMs can only run for up to 24 hours at a time, but Spot VMs do not have a maximum runtime.

Spot VMs are available at much lower price—60-91% discounts for machine types and GPUs as well as smaller discounts for local SSDs—compared to the on-demand price for standard VMs. Spot VMs are excess Compute Engine capacity, so their availability varies with usage. Spot VMs do not have a minimum or maximum runtime.You can simulate the preemption of a VM by stopping the VM or deleting the VM accordingly. You can access and recover data from any persistent disks that are attached to the VM, but those disks still incur storage charges until you delete them. As with standard VMs, persistent disks that are marked for auto-delete are deleted when you delete Spot VMs.

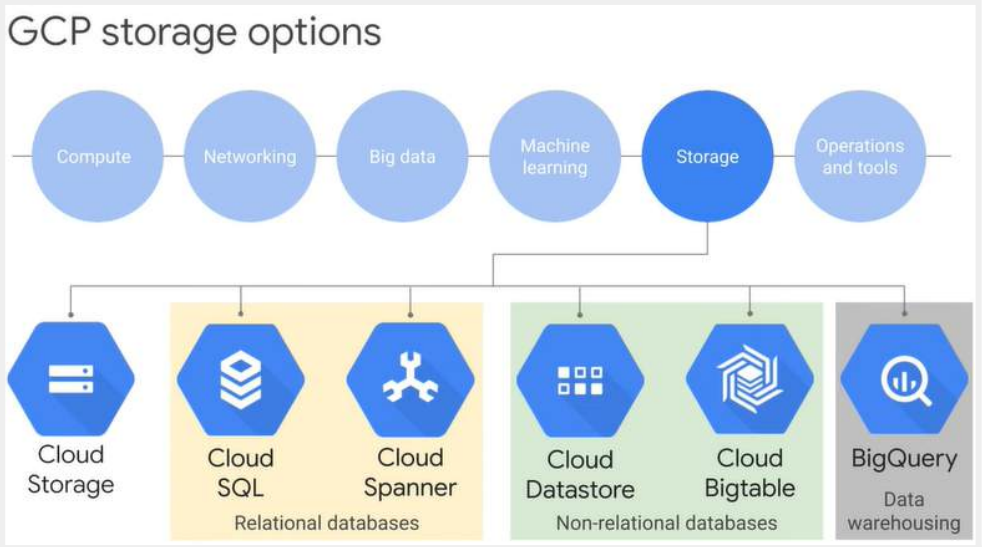
Preemption can happen when Spot VMs are in a RUNNING state; while in a TERMINATED state, Spot VMs are not considered for preemption. As a result, you can reset the preemption process by stopping then restarting Spot VMs, since stopping VMs leaves them in a TERMINATED state.Spot prices, the prices for Spot VMs, change over time, up to once every 30 days.

Managed instance groups recreate your instances if the vCPU, memory, and GPU resources are available. If you want a warning before your VMs are preempted, Compute Engine provides one hour advance notice before preemption. Spot VMs require available CPU quotas, also require disk quota and GPU quota respectively.

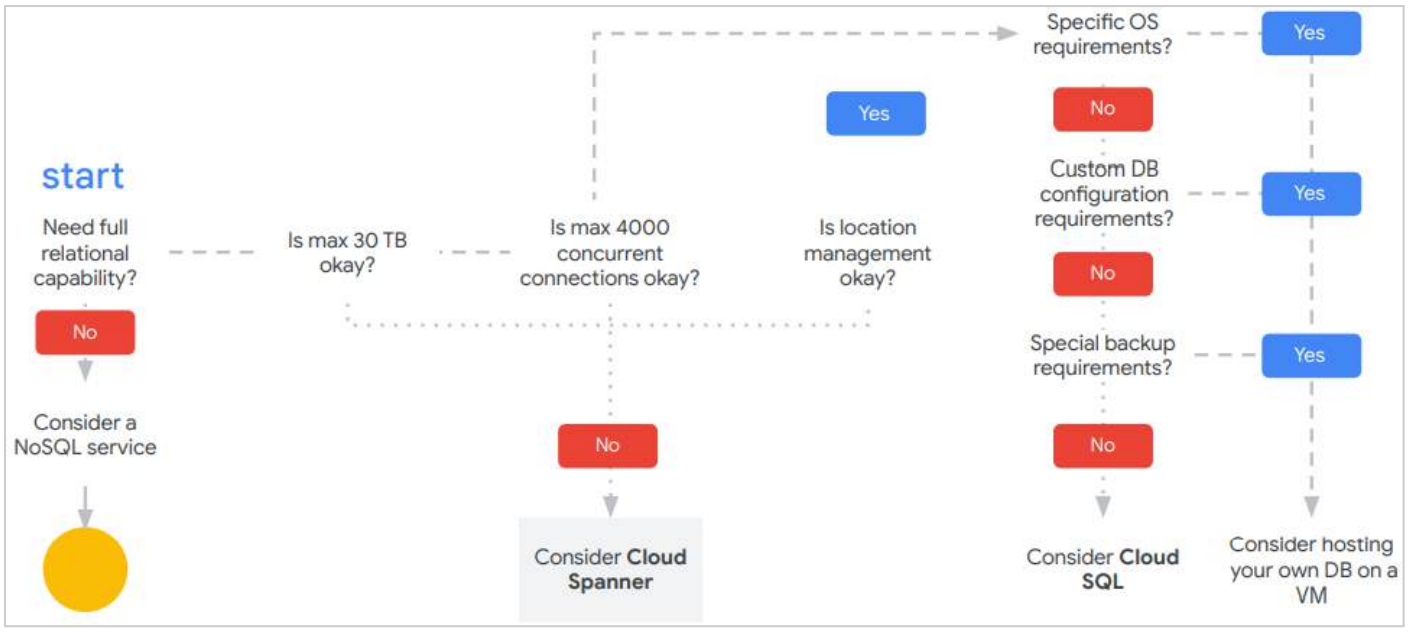


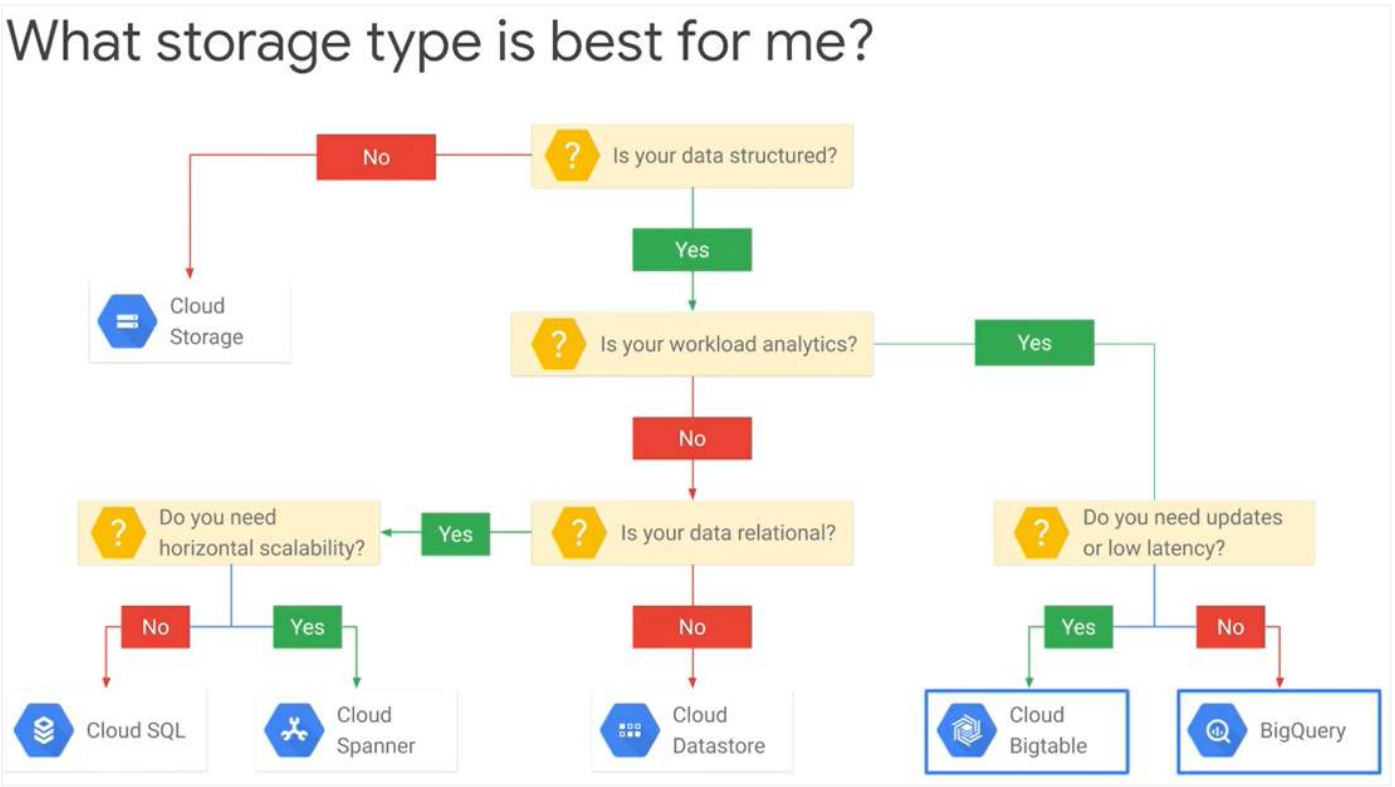
2.3 Planning and configuring data storage options. Considerations include:

a. Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)



Choosing Cloud SQL

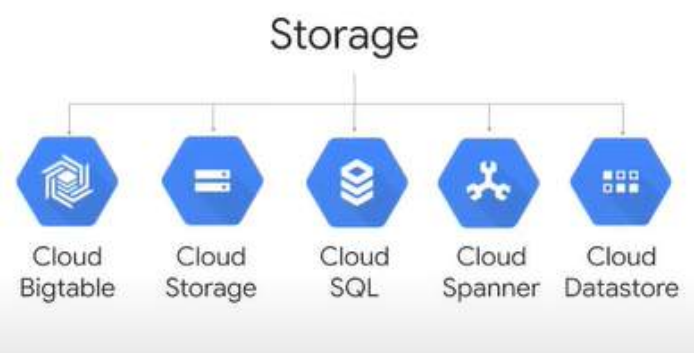


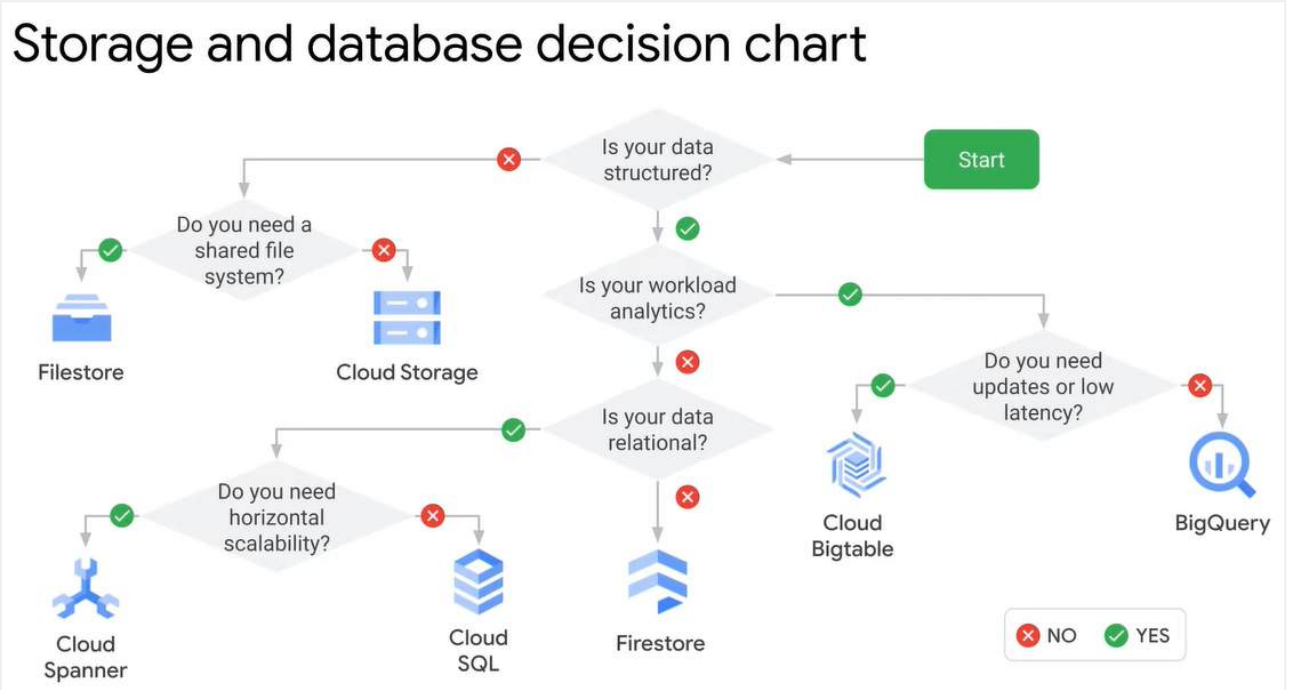


Cloud SQL is a Google Cloud service that manages a database instance for you. You are responsible for how you structure your data within it. Cloud SQL can handle common database tasks for you, such as automating backups, implementing high availability, handling data encryption, updating infrastructure and software, and providing logging and monitoring services. You can use it to deploy MySQL, PostgreSQL, or SQL Server databases to Google Cloud. It uses persistent disks attached to underlying Compute Engine instances to house your database, and implements a static IP address for you to connect to it.
Steps for setting up a Cloud SQL instance:

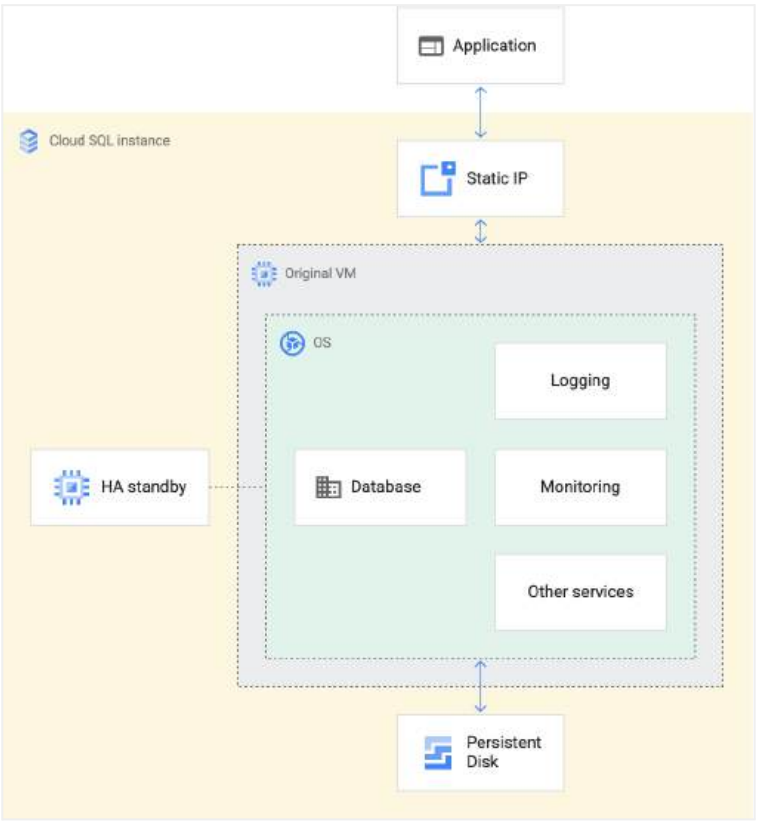
1. Create Instance.
2. Select database type.
3. Enter name: do not use sensitive or personally identifiable information. Your instance name can be publicly available.
4. Enter password for root user.
5. Select proper version: choose carefully, it cannot be edited.

6. Regional and zonal availability settings. Can't be modified. Pick a region where most people will access. You can also choose multi-region.
7. Select both primary and secondary zone. Both default to any with secondary being different than the primary.
8. Config settings include machine type, private or public ip, storage type, storage capacity, threshold for automated storage increase, as well as an increase setting to specify a limit on how big your database grows

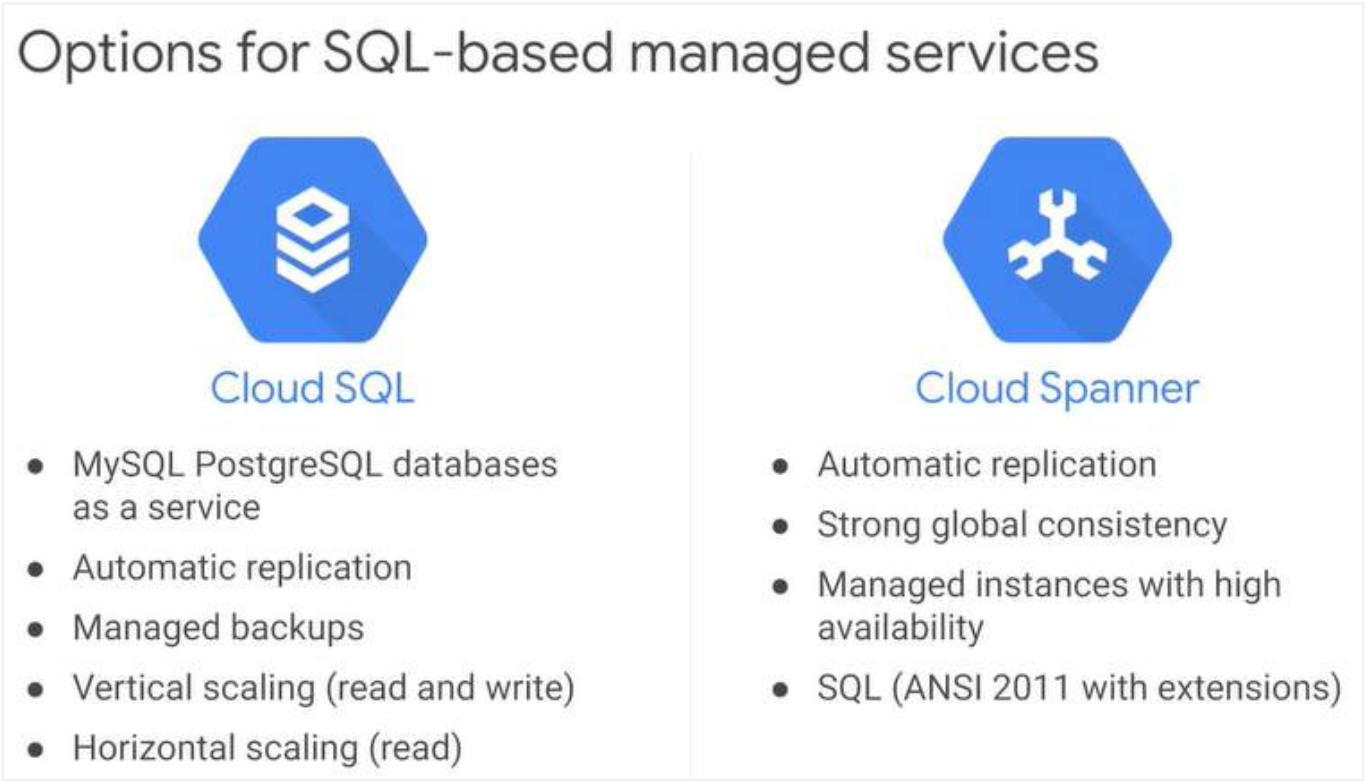




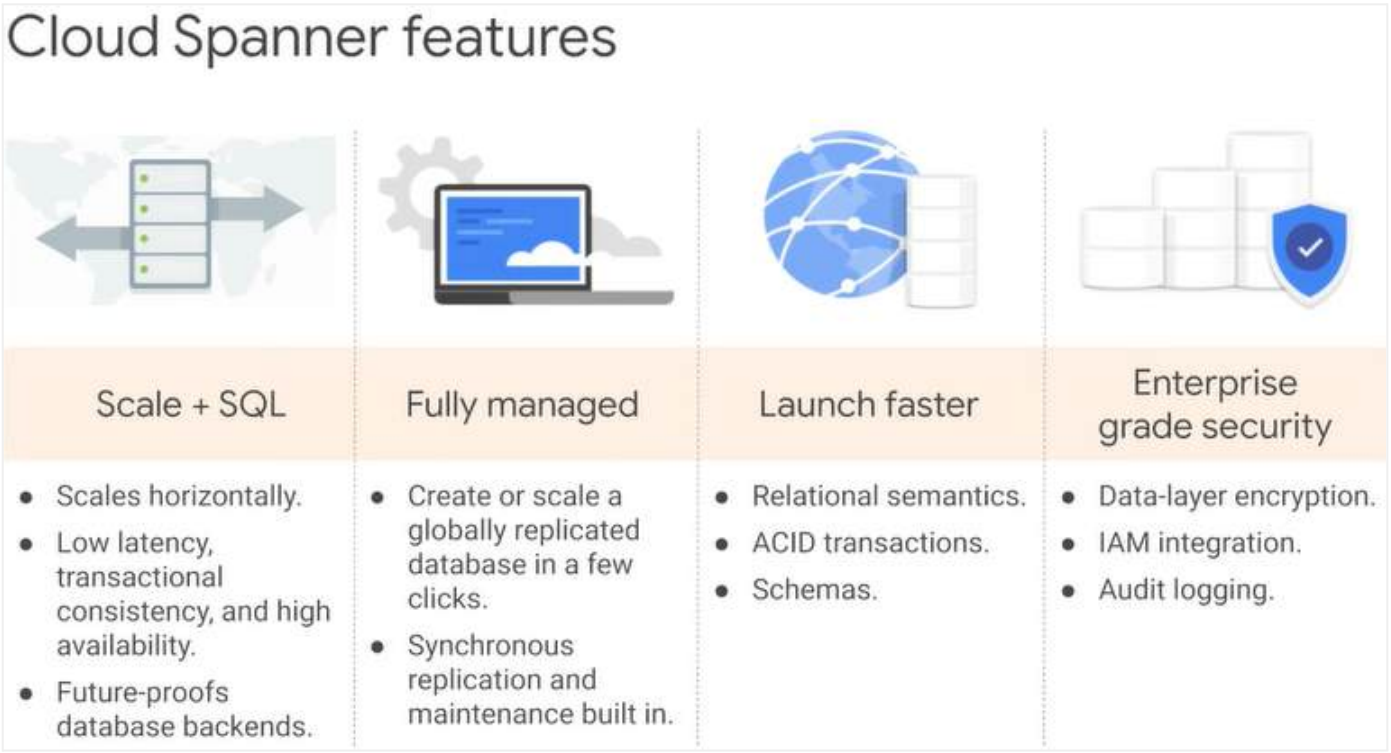
[Cloud SQL] Fully managed relational database service for MySQL, PostgreSQL, and SQL Server. Run the same relational databases you know with their rich extension collections, configuration flags and developer ecosystem, but without the hassle of self management. Each Cloud SQL instance is powered by a virtual machine (VM) running on a host Google Cloud server. Each VM operates the database program, such as MySQL Server, PostgreSQL, or SQL Server, and service agents that provide supporting services, such as logging and monitoring. The high availability option also provides a standby VM in another zone with a configuration that's identical to the primary VM. The database is stored on a scalable, durable network storage device called a persistent disk that attaches to the VM. A static IP address sits in front of each VM to ensure that the IP address an application connects to persists



throughout the lifetime of the Cloud SQL instance.

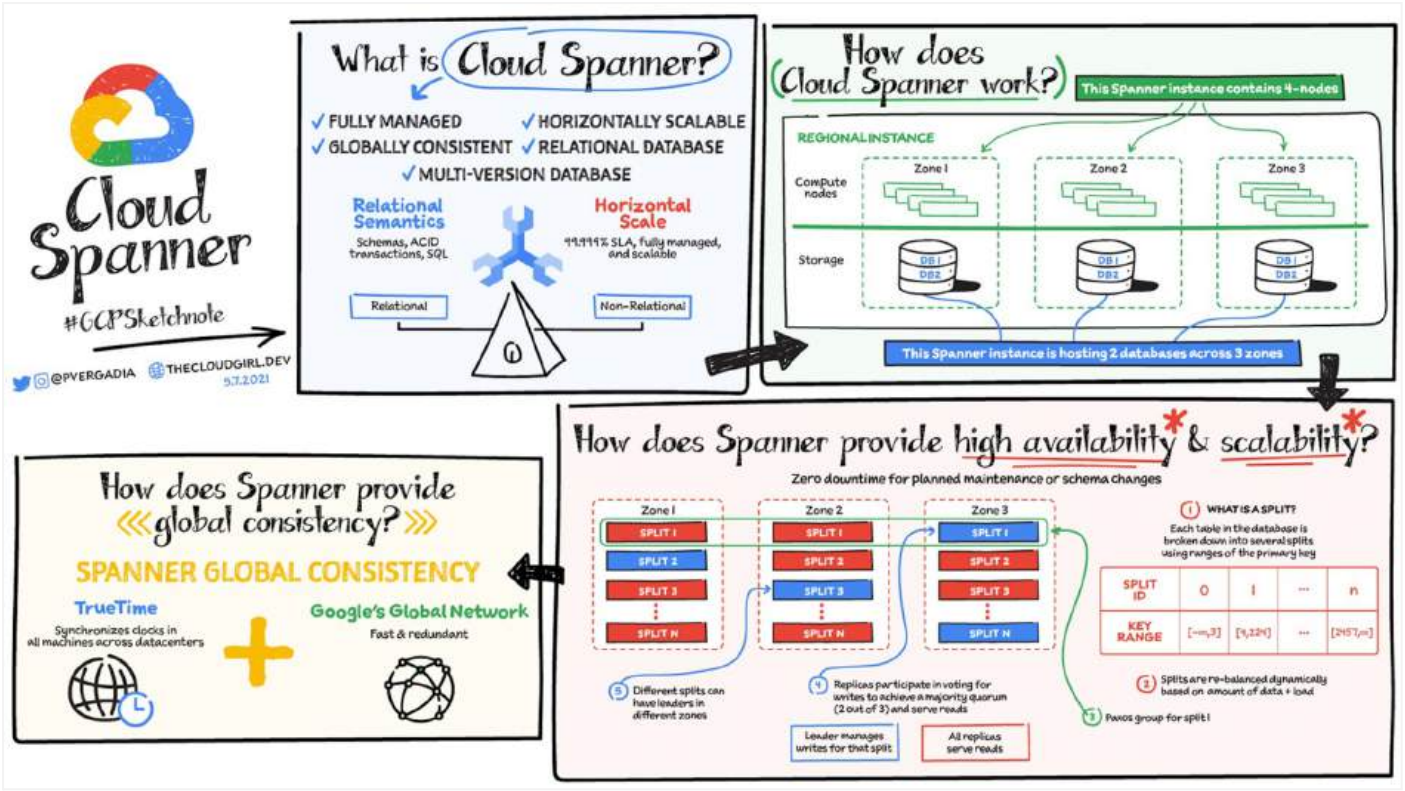


connect to your Cloud SQL instance	gcloud sql connect myinstance --user=root
Create a SQL database	CREATE DATABASE guestbook;



[Cloud Spanner](#) Fully managed relational database with unlimited scale, strong consistency, and up to 99.999% availability.Cloud Spanner is a fully managed, mission-critical, relational database service that offers transactional consistency at global scale, schemas, SQL (ANSI 2011 with extensions), and automatic, synchronous replication for high availability. It is a unique database that combines transactions,

SQL queries, and relational structure with the scalability that you typically associate with non-relational or NoSQL databases.

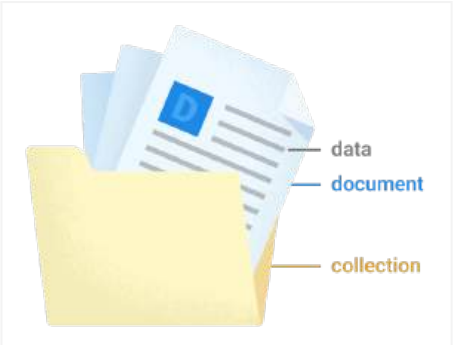


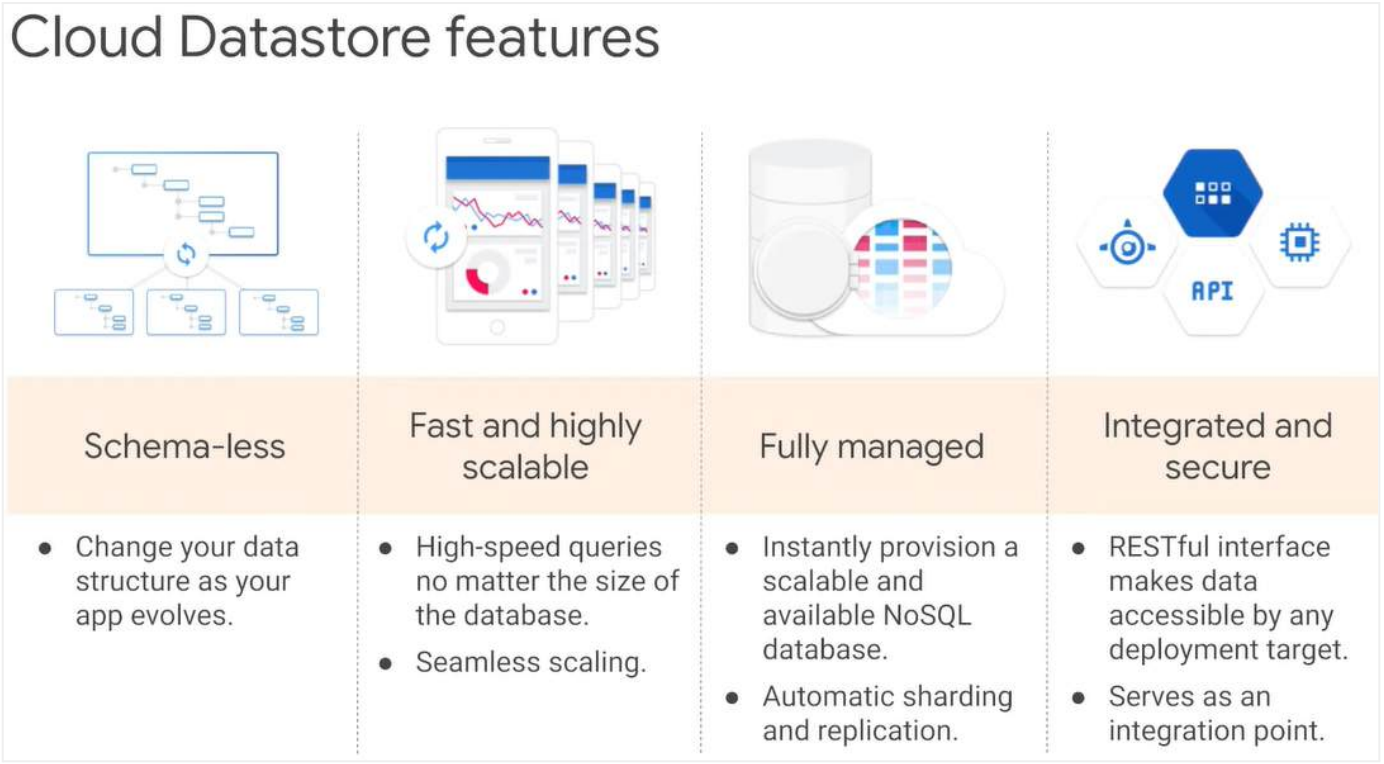
[Cloud Spanner](#) offers:

- **Strong consistency**, including strongly consistent secondary indexes.
- **SQL support**, with ALTER statements for schema changes.
- **Managed instances with high availability** through transparent, synchronous, built-in data replication.

Cloud Spanner offers regional and multi-region instance configurations.Cloud Spanner is ideal for relational, structured, and semi-structured data that requires high availability, strong consistency, and transactional reads and writes.As described in Query execution plans, Cloud Spanner's SQL compiler transforms a SQL statement into a query execution plan, which is used to obtain the results of the query.
A node is a measure of compute in Spanner.Node servers serve the read and write/commit transaction requests, but they don't store the data. Each node is replicated across three zones in the region. The database storage is also replicated across the three zones. Nodes in a zone are responsible for reading and writing to the storage in their zone. The data is stored in Google's underlying Colossus distributed replicated file system. TrueTime is a way to synchronize clocks in all machines across multiple datacenters.

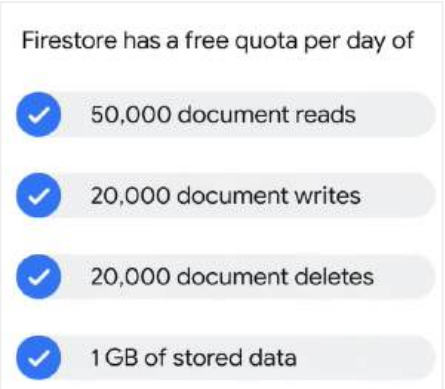
[Cloud Firestore](#) is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud. Like Firebase Realtime Database, it keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. [Cloud Firestore](#) also offers seamless integration with other Firebase and Google Cloud products, including Cloud Functions. Is flexible, scalable NoSQL cloud database to store and sync data for client- and server-side development.Store your data in documents, organized into collections. Horizontally scaling Cloud Firestore Data model [Firestore](#) automatically scales up and down based on demand. It requires no maintenance, and provides high availability of 99.99–99.999% achieved through strongly consistent data replication.





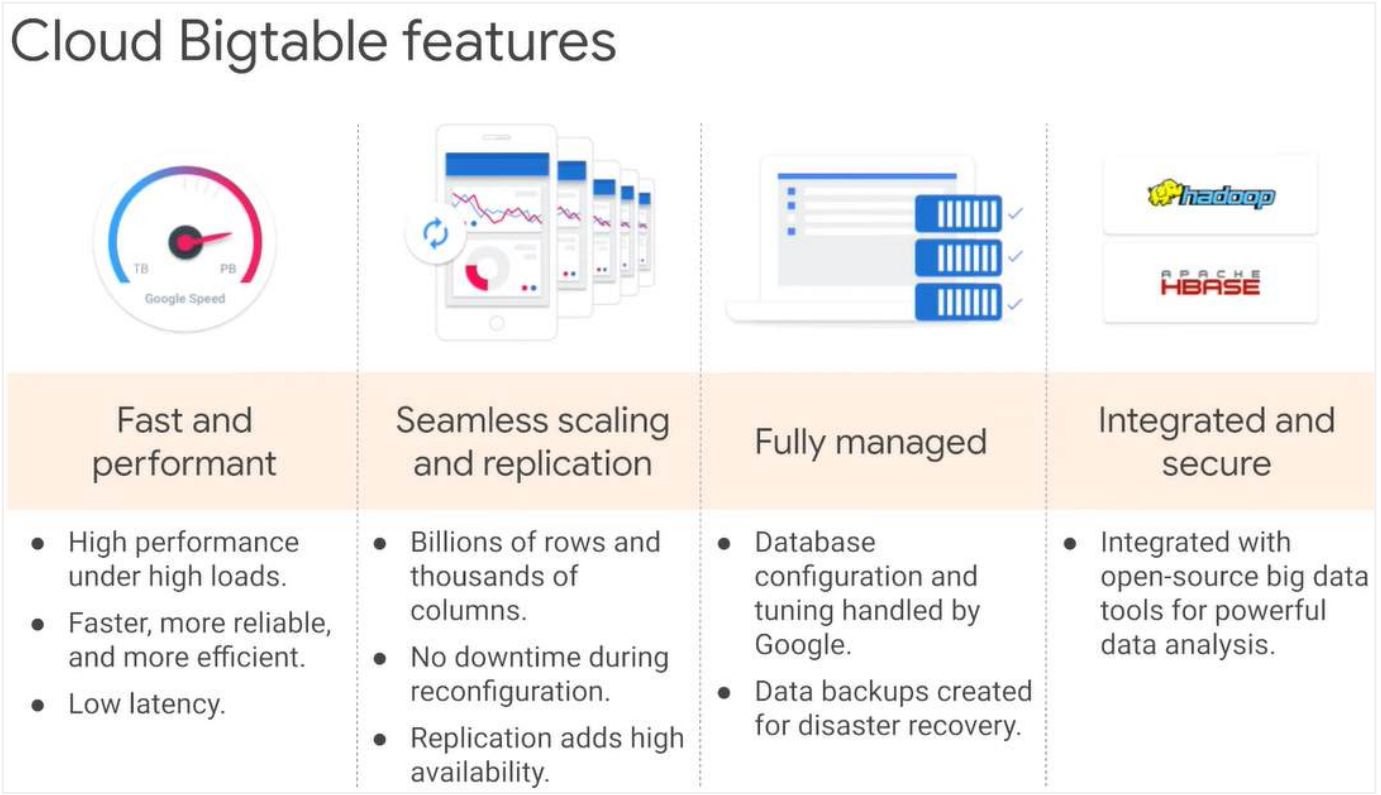
Firebase offers [two cloud-based, client-accessible database solutions] that support real time data syncing:

- **Cloud Firestore** is Firebase's newest database for mobile app development. It builds on the successes of the Realtime Database with a new, more intuitive data model. Cloud Firestore also features richer, faster queries and scales further than the Realtime Database.
- **Realtime Database** is Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in realtime.



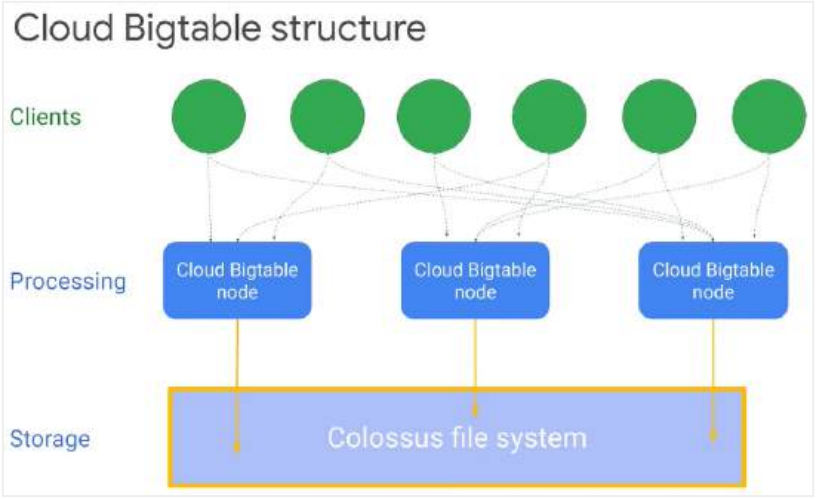
Realtime Database	Cloud Firestore
Stores data as one large JSON tree. • Simple data is very easy to store. • Complex, hierarchical data is harder to organize at scale. • Offline support for Apple Android clients. • Deep queries with limited sorting and filtering functionality. • Basic write and transaction operations. • Realtime Database is a regional solution. Learn more about the Realtime Database data model.	Stores data as collections of documents. • Simple data is easy to store in documents, which are very similar to JSON. • Complex, hierarchical data is easier to organize at scale, using subcollections within documents. • Requires less denormalization and data flattening. • Offline support for Apple, Android, and web clients. • Indexed queries with compound sorting and filtering. • Advanced write and transaction operations.

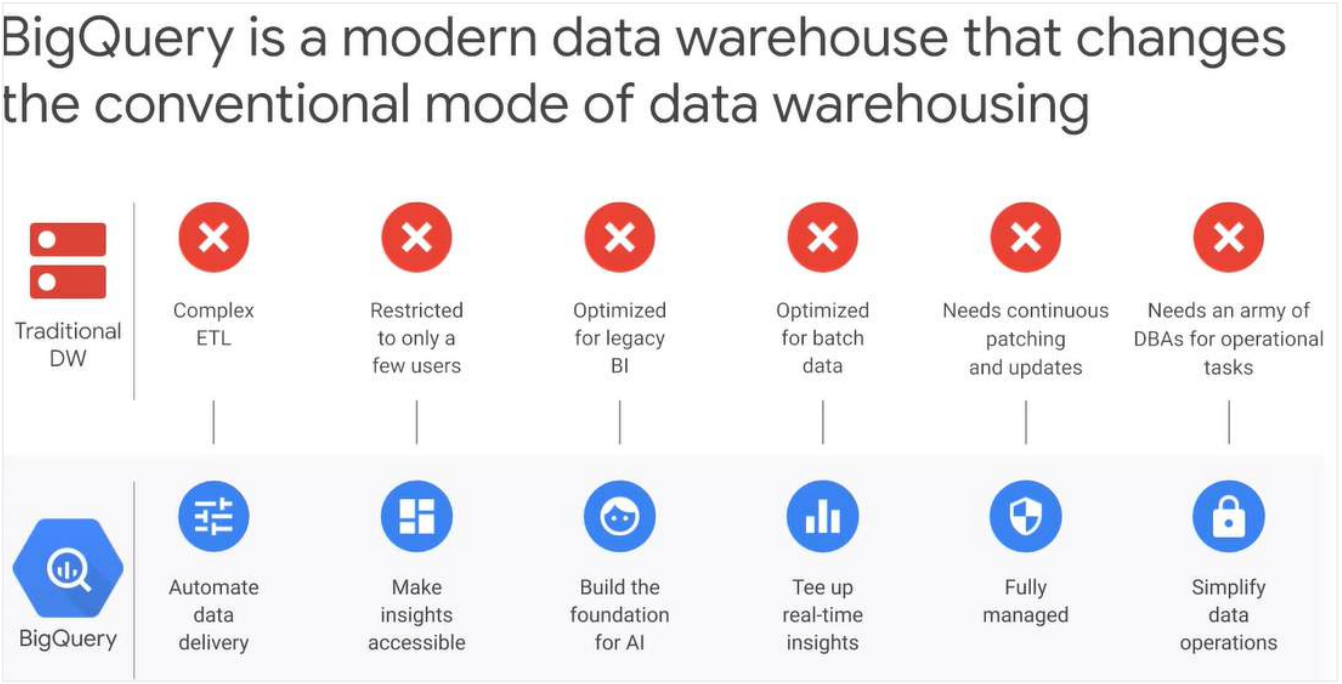
	• Cloud Firestore is a regional and multi-region solution that scales automatically. Learn more about the Cloud Firestore data model.



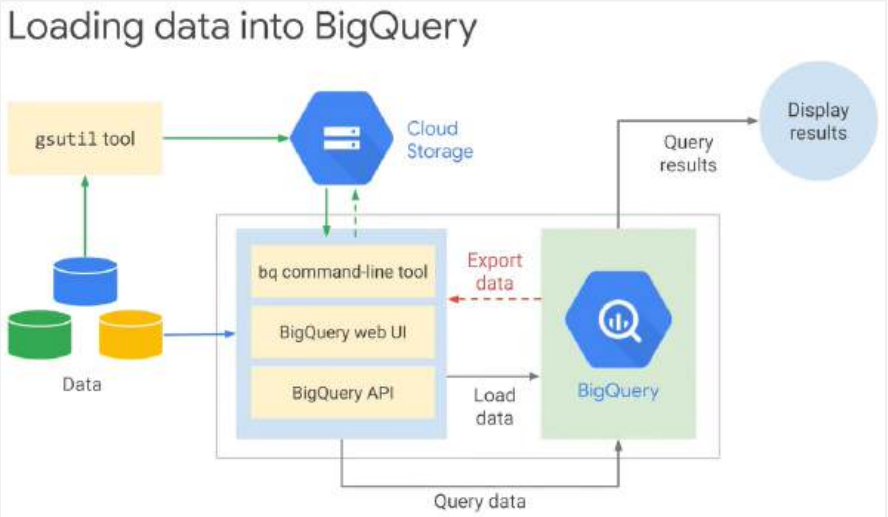
Cloud Bigtable A fully managed, scalable NoSQL database service for large analytical and operational workloads with up to 99.999% availability. **Bigtable** is ideal for storing very large amounts of data in a key-value store and supports high read and write throughput at low latency for fast access to large amounts of data. Throughput scales linearly—you can increase QPS (queries per second) by adding Bigtable nodes.

Cloud Bigtable is a sparsely populated table that can scale to billions of rows and thousands of columns, enabling you to store terabytes or even petabytes of data.Bigtable is ideal for applications that need high throughput and scalability for key/value data, where each value is typically no larger than 10 MB.
Columns can be unused in a row.Each cell in a given row and column has a unique timestamp (t).
In the original Bigtable paper, these nodes are called "tablet servers."





BigQuery Serverless, highly scalable, and cost-effective multicloud data warehouse designed for business agility. BigQuery is Google's fully managed, petabyte-scale, low-cost analytics data warehouse.
BigQuery is a fully managed enterprise data warehouse that helps you manage and analyze your data with built-in features like machine learning, geospatial analysis, and business intelligence. BigQuery's serverless architecture lets you use SQL queries to answer your organization's biggest questions with zero infrastructure management. BigQuery's scalable, distributed analysis engine lets you query terabytes in seconds and petabytes in minutes.BigQuery storage is automatically replicated across multiple locations to provide high availability.
Gain insights with real-time and predictive analytics: Query streaming data in real time and get up-to-date information on all your business processes. Predict business outcomes easily with built-in machine learning–without the need to move data.



BigQuery ML lets you create and execute machine learning models in BigQuery using standard SQL queries. Export **BigQuery ML** models for online prediction into Vertex AI or your own serving layer.
BigQuery Omni is a flexible, fully managed, multicloud analytics solution that allows you to cost-effectively and securely analyze data across clouds such as AWS and Azure. **BigQuery video tutorials**

To create a dataset	bq --location=US mk -d \ --default_table_expiration 3600 \ --description "This is my dataset." \ mydataset

Comparing storage options: technical details

	Cloud Firestore	Cloud Bigtable	Cloud Storage	Cloud SQL	Cloud Spanner	BigQuery
Type	NoSQL document	NoSQL wide column	Blobstore	Relational SQL for OLTP	Relational SQL for OLTP	Relational SQL for OLAP
Transactions	Yes	Single-row	No	Yes	Yes	No
Complex queries	No	No	No	Yes	Yes	Yes
Capacity	Terabytes+	Petabytes+	Petabytes+	Up to ~30 TB	Petabytes	Petabytes+
Unit size	1 MB/entity	~10 MB/cell ~100 MB/row	5 TB/object	Determined by DB engine	10,240 MiB/ row	10 MB/row

Option	Best for	Capacity
Cloud Storage	Storing immutable blobs larger than 10 MB	Petabytes Max unit size: 5TB per object
Cloud SQL	• Full SQL support for an online transaction processing system • Web frameworks and existing applications	Up to 30,720GB
Spanner	• Full SQL support for an online transaction processing system • Horizontal scalability • Large-scale database applications that are larger than 2 TB	Petabytes
Firestore	Massive scaling and predictability together with real time query results and offline query support	Terabytes Max unit size: 1MB per entity
Cloud 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: 10MB p/cell, 100MB p/row

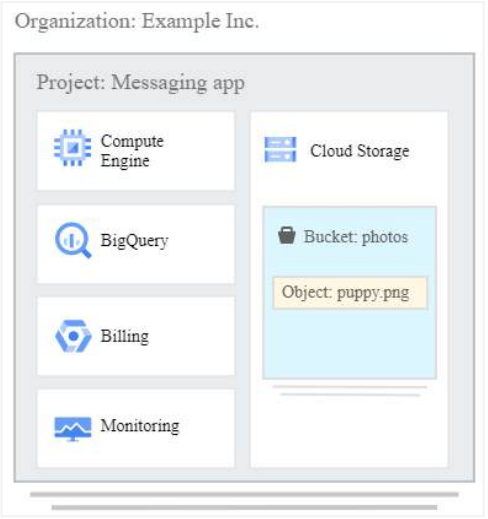
Sharding is a very important concept that helps the system to keep data in different resources according to the sharding process. The word “Shard” means “a small part of a whole“. Hence Sharding means dividing a larger part into smaller parts. In DBMS, Sharding is a type of DataBase partitioning in which a large DataBase is divided or partitioned into smaller data and different nodes. These shards are not only smaller, but also faster and hence easily manageable.

b. Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

Cloud Storage is a service for storing your *objects* in Google Cloud. An object is an immutable piece of data consisting of a file of any format. You store objects in containers called *buckets*. All buckets are associated with a *project*, and you can group your projects under an *organization*. Object Lifecycle Management (OLM)
For offline data transfer our Transfer Appliance is a shippable storage server that sits in your datacenter and then ships to an ingest location where the data is uploaded to Cloud Storage.
Storage classes for any workload

1. Standard Storage: Good for “hot” data that’s accessed frequently, including websites, streaming videos, and mobile apps.

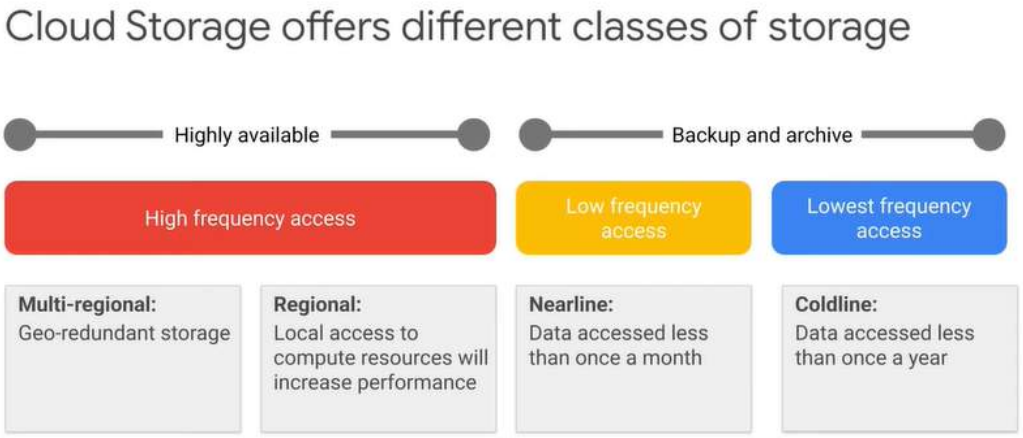
2. Nearline Storage: Low cost. Good for data that can be stored for at least 30 days, including data backup and long-tail multimedia content. storing infrequently accessed data.
3. Coldline Storage: Very low cost. Good for data that can be stored for at least 90 days, including disaster recovery.
4. Archive Storage: Lowest cost. Good for data that can be stored for at least 365 days, including regulatory archives. for data archiving, online backup, and disaster recovery.



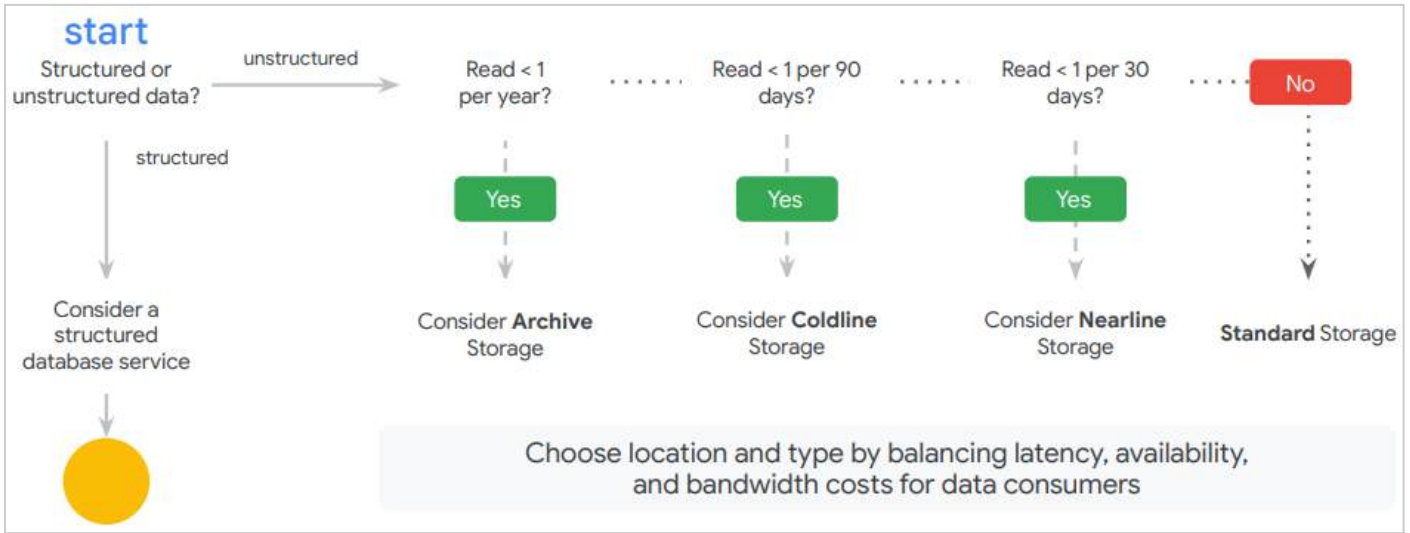
If you don't specify a default storage class when you create a bucket, that bucket's default storage class is set to Standard storage.DRA (Durable Reduced Availability)has higher pricing for operations. Define minimum retention periods that objects must be stored for before they're deletable. Place a hold on an object to prevent its deletion. When a live version of an object is replaced or deleted, it can be retained as a *noncurrent version* if you enable Object Versioning.
storage services:

- Cloud Storage for Firebase: Manage data for your mobile applications.
- Persistent Disk: Add block storage to your Compute Engine virtual machine.
- Storage Transfer Service: Quickly import online data into Cloud Storage or between Cloud Storage buckets.

- Filestore: Create a file-based workload.



Choosing a storage class



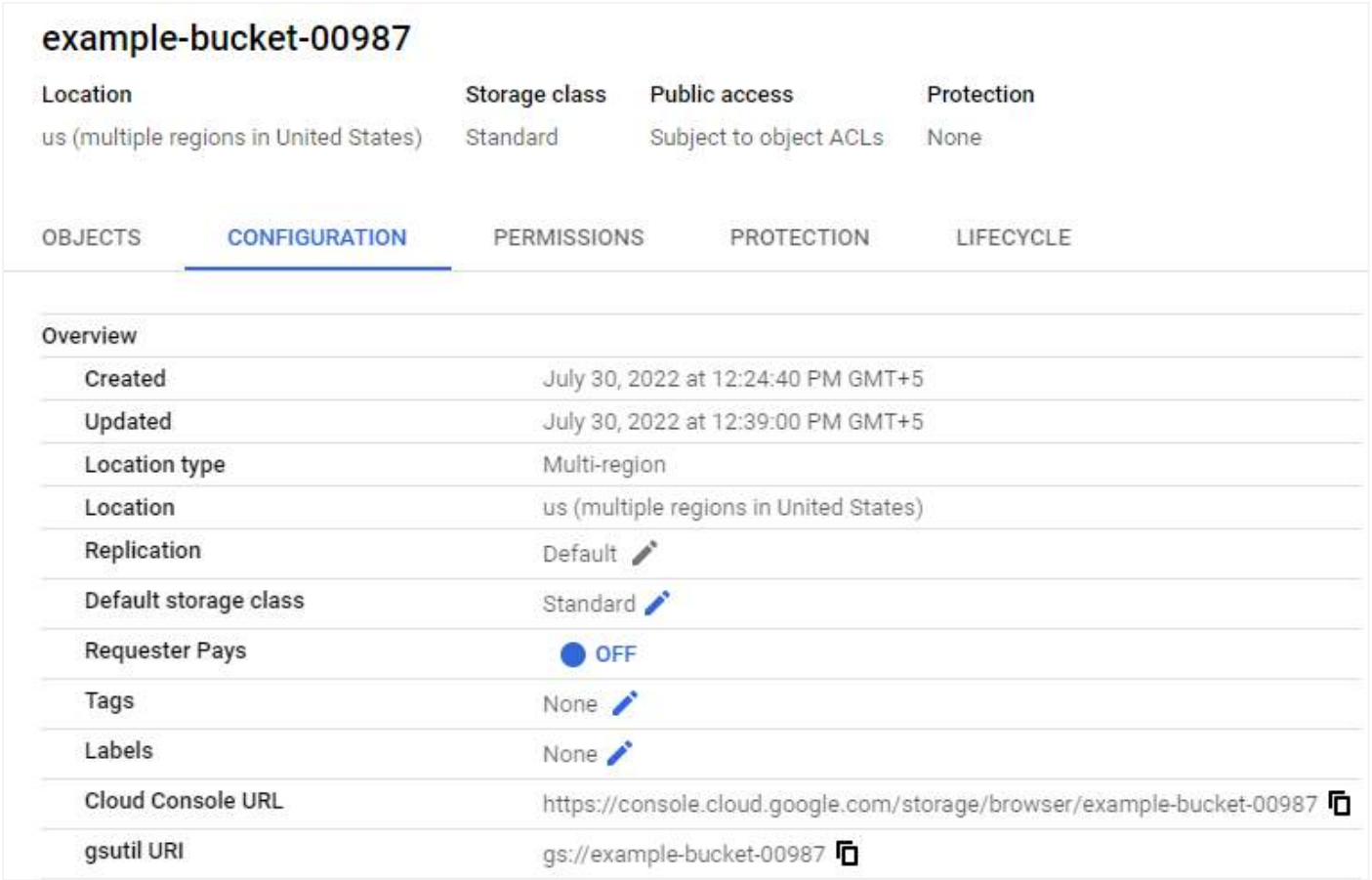
Cloud Storage helps you store binary objects in Google Cloud. It can house data in any format as an immutable object. In Cloud Storage, objects are stored in containers called buckets.

Buckets can be used to upload and download objects, and permissions can be assigned to specify who has access to them.
You can manage and interact with Cloud Storage via the console, via the command line and the gsutil command set, via client libraries, or through APIs. Steps for creating a cloud bucket include:

1. Naming your bucket - has to be globally unique, and not contain any sensitive information
2. Choose location type and option:
    a. Regional is a specific geographical area where a datacenter campus resides. It minimizes latency and network bandwidth for consumers grouped in a specific region
    b. Dual-region is a a specific pair or regions. It provides geo-redundancy
    c. Multi-region is a disbursed geographic area, such as the US or Europe. You can use it to serve content to consumers outside of Google and distributed over large areas.
3. Pick a default storage class for bucket. You can override this per object.
    a. Standard is for immediate access and has no minimum storage duration
    b. Nearline has a 30 day minimum duration and data retrieval charges
    c. Coldline has a 90 day min duration and data retrieval charges
    d. Archive has a 365 day min duration and data retrieval charges
4. Click Create or submit the command.

Storage options has unique price and performance characteristics:

- Zonal persistent disk: Efficient, reliable block storage.
- Regional persistent disk: Regional block storage replicated in two zones.
- Local SSD: High performance, transient, local block storage.
- Cloud Storage buckets: Affordable object storage.
- Filestore: High performance file storage for Google Cloud users.



gsutil is a command-line tool that allows you to interact with Cloud Storage through a terminal.

gsutil is a Python application that lets you access Cloud Storage from the command line. If you only want to access public data, follow the instructions in Accessing Public Data.When using gsutil with Windows, you cannot use Ctrl-C to cancel a command that is currently running.

To download this image	curl https://upload.wikimedia.org/wikipedia/commons/thumb/a/a4/Ada_Lovelace_portrait.jpg/800px-Ada_Lovelace_portrait.jpg --output ada.jpg
to upload the image	gsutil cp ada.jpg gs://YOUR-BUCKET-NAME
remove the downloaded image	rm ada.jpg
Download an object from your bucket	gsutil cp -r gs://YOUR-BUCKET-NAME/ada.jpg .
Copy an object to a folder in the bucket	gsutil cp gs://YOUR-BUCKET-NAME/ada.jpg gs://YOUR-BUCKET-NAME/image-folder/
List contents of a bucket or folder List details for an object	gsutil ls gs://YOUR-BUCKET-NAME gsutil ls -l gs://YOUR-BUCKET-NAME/ada.jpg
Make your object publicly accessible	gsutil acl ch -u AllUsers:R gs://YOUR-BUCKET-NAME/ada.jpg
Remove public access	gsutil acl ch -d AllUsers gs://YOUR-BUCKET-NAME/ada.jpg
To delete an object	gsutil rm gs://YOUR-BUCKET-NAME/ada.jpg

Persistent disks are durable network storage devices that your instances can access like physical disks in a desktop or a server.The data on each persistent disk is distributed across several physical disks.
Disk types:

- **Standard persistent disks** (pd-standard) are backed by standard hard disk drives (HDD).
- **Balanced persistent disks** (pd-balanced) are backed by solid-state drives (SSD). They are an alternative to SSD persistent disks that balance performance and cost.
- **SSD persistent disks** (pd-ssd) are backed by solid-state drives (SSD).
- **Extreme persistent disks** (pd-extreme) are backed by solid-state drives (SSD). With consistently high performance for both random access workloads and bulk throughput, extreme persistent disks are designed for high-end database workloads. Unlike other disk types, you can provision your desired IOPS. For more information, see Extreme persistent disks.

If you create a disk in the console, the default disk type is pd-balanced. If you create a disk using the gcloud CLI or the Compute Engine API, the default disk type is pd-standard.
Configure disks to meet performance requirements | Share persistent disks between VMs

Zonal persistent disks: Standard persistent disks are efficient and economical for handling sequential read/write operations, but they aren't optimized to handle high rates of random input/output operations per second (IOPS). If your apps require high rates of random IOPS, use SSD or extreme persistent disks. Persistent disk performance is based on the total persistent disk capacity attached to an instance and the number of vCPUs that the instance has.Each persistent disk remains encrypted either with system-defined keys or with customer-supplied keys.
Most instances can have up to 128 persistent disks and up to 257 TB of total persistent disk space attached. Total persistent disk space for an instance includes the size of the boot persistent disk.Shared-core machine types are limited to 16 persistent disks and 3 TB of total persistent disk space.

Regional persistent disks: If you are designing robust systems or high availability services on Compute Engine, use regional persistent disks combined with other best practices such as backing up your data

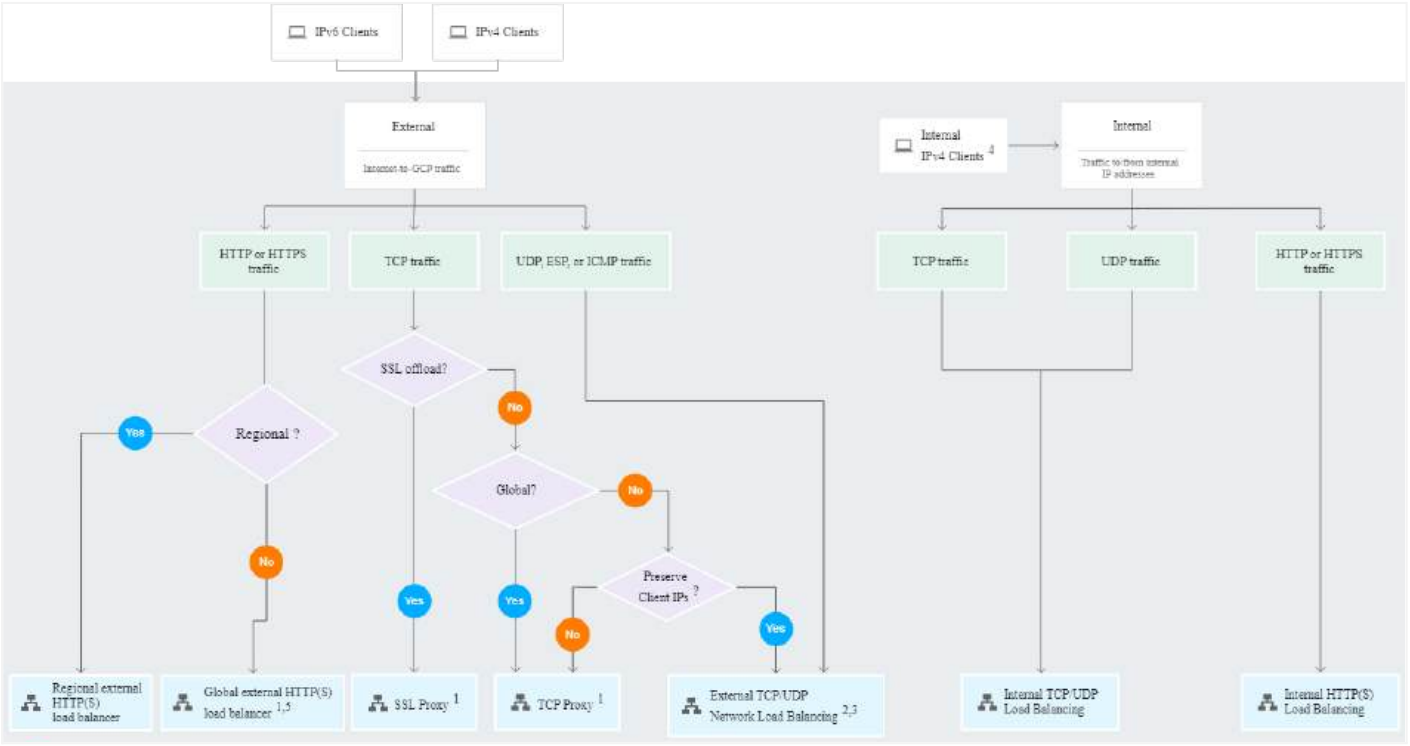
using snapshots. Regional persistent disks are also designed to work with regional managed instance groups. The --force-attach flag lets you attach the regional persistent disk to a standby VM instance even if the disk can't be detached from the original VM due to its unavailability. You cannot force attach a zonal persistent disk to an instance.You can attach a regional balanced persistent disk to at most 10 VM instances in read-only mode.
Compute Engine automatically encrypts your data before it travels outside of your instance
Each local SSD is 375 GB in size, but you can attach a maximum of 24 local SSD partitions for a total of 9 TB per instance.Local SSDs are designed to offer very high IOPS and low latency.
Use the gsutil tool to create buckets, write data to buckets, and read data from those buckets.
Cloud Storage pricing is based on the following components:

- Data storage: the amount of data stored in your buckets. Storage rates vary depending on the storage class of your data and location of your buckets.
- Data processing: the processing done by Cloud Storage, which includes operations charges, any applicable retrieval fees, and inter-region replication.
- Network usage: the amount of data read from or moved between your buckets.

2.4 Planning and configuring network resources. Tasks include:

a. Differentiating load balancing options

Cloud Load Balancing is a fully distributed, software-defined, managed service for all your traffic. Apply Cloud Load Balancing to all of your traffic: HTTP(S), TCP/SSL, and UDP.Google Cloud Armor security policies enable you to rate-limit or redirect requests to your HTTP(S) Load Balancing, TCP Proxy Load Balancing, or SSL Proxy Load Balancing at the Google Cloud edge, as close as possiblDNSe to the source of incoming traffic.
Load balancer decision tree



1. IPv6 termination is available if you configure the load balancer in Premium Tier.
2. Choose external TCP/UDP network load balancing if you need to ensure that the load balancer is located in a particular region, or if you want to configure IPv6 load balancing (with dual-stack backends). The latter is only available for backend service-based network load balancers.
3. External TCP/UDP network load balancers use regional external IP addresses that are accessible by clients anywhere.
4. Clients in a VPC network or in a network connected to a VPC network.

5. The global external HTTP(S) load balancer (classic) can be configured to be effectively regional in Standard Tier.

HTTP(S) load balancing can balance HTTP and HTTPS traffic across multiple backend instances, across multiple regions. Your entire app is available via a single global IP address, resulting in a simplified DNS setup.Enable Cloud CDN for HTTP(S) load balancing for optimizing application delivery for your users with a single checkbox.
TCP load balancing can spread TCP traffic over a pool of instances within a Compute Engine region. It is scalable, does not require pre-warming, and health checks help ensure only healthy instances receive traffic.
UDP load balancing can spread UDP traffic over a pool of instances within a Compute Engine region.

Internal or external IP address	Regional or global	Premium or Standard network tier	Proxy or pass-through	Traffic type	Load balancer type
Internal	Regional	Premium only	Pass-through	TCP or UDP	Internal TCP/UDP load balancer
			Proxy	HTTP or HTTPS	Internal HTTP(S) load balancer
External	Global	Premium only	Proxy	HTTP or HTTPS	Global external HTTP(S) load balancer
	Global in Premium Tier Effectively regional1 in Standard Tier	Premium or Standard	Proxy	HTTP or HTTPS	Global external HTTP(S) load balancer (classic)
				SSL	External SSL proxy load balancer
				TCP	External TCP proxy load balancer
	Regional	Premium or Standard	Pass-through	TCP, UDP, ESP, GRE, ICMP, and ICMPv6	External TCP/UDP Network load balancer
		Standard only	Proxy	HTTP or HTTPS	Regional external HTTP(S) load balancer

External versus internal load balancing

- **External load balancers** distribute traffic coming from the internet to your Google Cloud Virtual Private Cloud (VPC) network. Global load balancing requires that you use the Premium Tier of Network Service Tiers. For regional load balancing, you can use Standard Tier.
- **Internal load balancers** distribute traffic to instances inside of Google Cloud.

Global versus regional load balancing

1. Use **global load balancing** when your backends are distributed across multiple regions, your users need access to the same applications and content, and you want to provide access by using a single anycast IP address. Global load balancing can also provide IPv6 termination.
2. Use **regional load balancing** when your backends are in one region, you only require IPv4 termination, or when you have jurisdictional compliance requirements for traffic to stay in a particular region.

The global external HTTP(S) load balancers and external SSL proxy load balancers terminate Transport Layer Security (TLS) in locations that are distributed globally, so as to minimize latency between clients and the load balancer.

Network Service Tiers lets you optimize connectivity between systems on the internet and your Google Cloud instances. Premium Tier delivers traffic on Google's premium backbone, while Standard Tier uses regular ISP networks.

- Use **Premium Tier** for high performance and low latency. If you do not specify a network tier, your load balancer defaults to using Premium Tier.
- Use **Standard Tier** as a low-cost alternative for applications that don't have strict requirements for latency or performance. As noted in this table, not all load balancers can be deployed in the Standard Tier.

Proxy load balancers terminate incoming client connections and open new connections from the load balancer to the backends.The global external HTTP(S) load balancers terminate client connections using Google Front End (GFE) proxies worldwide.

Pass-through load balancers do not terminate client connections.Use a pass-through load balancer when you need to preserve the client packet information.

Use load balancing to distribute user requests among sets of instances

Global	HTTP(S) load balancing	Distributes HTTP(S) traffic among groups of instances based on: • Proximity to the user • Requested URL • Both	External
	SSL Proxy load balancing	Distributes SSL traffic among groups of instances based on proximity to the user.	
	TCP Proxy load balancing	Distributes TCP traffic among groups of instances based on proximity to the user.	
Regional	Network load balancing	• Distributes traffic among a pool of instances within a region. • Can balance any kind of TCP/UDP traffic.	
	Internal load balancing	Distributes traffic from GCP virtual machine instances to a group of instances in the same region.	Internal

Cloud Armor IP allowlist/denylist enable you to restrict or allow access to your HTTP(S) load balancer at the edge of the Google Cloud, as close as possible to the user and to malicious traffic. This prevents malicious users or traffic from consuming resources or entering your virtual private cloud (VPC) networks.
The host and path rules determine how your traffic will be directed. Health checks determine which instances of a load balancer can receive new connections. This HTTP health check polls instances every 5 seconds, waits up to 5 seconds for a response and treats 2 successful or 2 failed attempts as healthy or unhealthy, respectively.

Network Tags enable you to make firewall rules and routes applicable to specific VM instances.

Managed instance groups offer **autoscaling** capabilities that allow you to automatically add or remove instances from a managed instance group based on increases or decreases in load. Autoscaling helps your applications gracefully handle increases in traffic and reduces cost when the need for resources is lower. You just define the autoscaling policy and the autoscaler performs automatic scaling based on the measured load.

In the Cloud Console, navigate to **Navigation menu > Network Security > Cloud Armor.** Cloud Armor security policies create logs that can be explored to determine when traffic is denied and when it is allowed, along with the source of the traffic.

Google Cloud offers Internal Load Balancing for your TCP/UDP-based traffic. Internal Load Balancing enables you to run and scale your services behind a private load balancing IP address that is accessible only to your internal virtual machine instances. Each Google Cloud project starts with the **default** network.

Note: The **curl** commands demonstrate that each VM instance lists the Client IP and its own name and location. This will be useful when verifying that the Internal Load Balancer sends traffic to both backends.

Note: Choosing **Only between my VMs** makes this Load Balancer internal. This choice requires the backends to be in a single region (us-central1) and does not allow offloading TCP processing to the Load Balancer.

b. Identifying resource locations in a network for availability

Resource locations supported services Google Cloud networking products

c. Configuring Cloud DNS

Cloud DNS is a high-performance, resilient, global Domain Name System (DNS) service that publishes your domain names to the global DNS in a cost-effective way. DNS is a hierarchical distributed database that lets you store IP addresses and other data, and look them up by name. Cloud DNS lets you publish your zones and records in DNS without the burden of managing your own DNS servers and software. Google Cloud offers inbound and outbound DNS forwarding for private zones. You can configure DNS forwarding by creating a forwarding zone or a Cloud DNS server policy.
Note: Cloud DNS does *not* support DNS forwarding for public zones. Cloud DNS public zones must be authoritative.
Cloud DNS uses anycast to serve your managed zones from multiple locations around the world for high availability. Requests are automatically routed to the nearest location, reducing latency and improving authoritative name lookup performance for your users. The time-to-live (TTL) value that you set for your records, which is specified in seconds, controls the DNS resolver's cache. For example, if you set a TTL value of 86400 (the number of seconds in 24 hours), the DNS resolvers are instructed to cache the records for 24 hours.

Set up a domain by using Cloud DNS

- Register a domain name using Google Domains or Cloud Domains
- Create a virtual machine (VM) instance
- Run a basic Apache web server
- Set up your domain using Cloud DNS
- Update name servers
- Verify your setup

Section 3. Deploying and implementing a cloud solution

3.1 Deploying and implementing Compute Engine resources. Tasks include:

a. Launching a compute instance using Cloud Console and Cloud SDK (gcloud) (e.g., assign disks, availability policy, SSH keys)

Add a persistent disk to your VM:

1. On the **VM instance details** page, click **Edit**.
2. Under **Additional disks**, click **Add new disk**.
3. Specify a name for the disk, configure the disk's properties, and select **Blank** as the **Source type**.
4. Click **Done** to complete the disk's configuration.

To create the zonal persistent disk	gcloud compute disks create DISK_NAME \ --size DISK_SIZE \ --type DISK_TYPE
Attach it to any running or stopped instance	gcloud compute instances attach-disk

	INSTANCE_NAME \ --disk DISK_NAME

DISK_TYPE: full or partial URL for the type of the persistent disk.

Set VM host maintenance policy On the VM details page, complete the following steps:

1. Click the **Edit** button at the top of the page.
2. Under **Availability policies**, update the policy as needed. From the **Availability policies** section, you can set the **On host maintenance** and **Automatic restart** options.
3. Click **Save**.

```
gcloud compute instances create VM_NAME \
  --maintenance-policy=MAINTENANCE_POLICY \
  --host-error-timeout-seconds=NUMBER_OF_SECONDS \
  [--no-restart-on-failure]
```

```
gcloud compute instances set-scheduling VM_NAME \
   --maintenance-policy=MAINTENANCE_POLICY \
   --host-error-timeout-seconds=NUMBER_OF_SECONDS \
   [--no-restart-on-failure | --restart-on-failure]
```

- MAINTENANCE_POLICY: the policy for this VM, either TERMINATE or MIGRATE
- NUMBER_OF_SECONDS: the number of seconds Compute Engine waits before restarting an unresponsive VM, between 90 and 330, in increments of 30. This option is available in Preview. To use this option use the gcloud beta compute instances create command.

Add SSH keys to VMs : When you add SSH keys to your Google account, Compute Engine generates a username for you by combining the username and domain from the email associated with your Google Account. For example, if your email address is cloudysanfrancisco@gmail.com, your username is cloudysanfrancisco_gmail_com. If you add an SSH key in a project that is outside of your organization, your username is prefixed with ext_, for example, ext_cloudysanfrancisco_gmail_com.

KEY_FILE_PATH: the path to the public SSH key on your workstation. The key must use the public-openssh format
PROJECT: Optional: a project where you intend to use your SSH key. Specify this field to use your SSH key in a project outside of your organization, or you are not a member of a Cloud Identity organization
EXPIRE_TIME: Optional: the expiration time for the SSH key
For example, if you specify 30m the SSH key expires after 30 minutes.This flag uses the following units:
○ s for seconds
○ m for minutes
○ h for hours
○ d for day
	gcloud compute os-login ssh-keys add \
	  --key-file=KEY_FILE_PATH \
	  --project=PROJECT \
	  --ttl=EXPIRE_TIME

No	Task	CMD
1	Set the region to us-central1	gcloud config set compute/region us-central1
2	view the project region setting	gcloud config get-value compute/region
3	Set the zone to us-central1-b	gcloud config set compute/zone us-central1-b

4	view the project zone setting	gcloud config get-value compute/zone
5	view the project id	gcloud config get-value project
6	view details about the project	gcloud compute project-info describe --project $(gcloud config get-value project)
7	Environment variable to store your Project ID	export PROJECT_ID=$(gcloud config get-value project)
8	Environment variable to store your Zone	export ZONE=$(gcloud config get-value compute/zone)
9	verify that your variables were set properly	echo -e "PROJECT ID: $PROJECT_ID\nZONE: $ZONE"
10	To create your VM	gcloud compute instances create gcelab2 --machine-type n1-standard-2 --zone $ZONE
11	open help for the create command	gcloud compute instances create --help
12	simple usage guidelines that are available by adding the -h flag	gcloud -h, gcloud config --help gcloud help config
13	View the list of configurations	gcloud config list
14	To see all properties and their settings	gcloud config list --all
15	List your components	gcloud components list
16	List the compute instance available in the project	gcloud compute instances list
17	List the gcelab2 virtual machine	gcloud compute instances list --filter="name=('gcelab2')"
18	List the Firewall rules	gcloud compute firewall-rules list
19	List the Firewall rules for the default network	gcloud compute firewall-rules list --filter="network='default'"
20	List the Firewall rules for the default network where the allow rule matches an ICMP rule	gcloud compute firewall-rules list --filter="NETWORK:'default' AND ALLOW:'icmp'"
21	connect to your VM with SSH	gcloud compute ssh gcelab2 --zone $ZONE
22	Install nginx web server on to virtual machine	sudo apt install -y nginx
23	Add a tag to the virtual machine	gcloud compute instances add-tags gcelab2 --tags http-server,https-server
24	Update the firewall rule to allow	gcloud compute firewall-rules create default-allow-http --direction=INGRESS --priority=1000 --network=default --action=ALLOW --rules=tcp:80 --source-ranges=0.0.0.0/0 --target-tags=http-server
25	List the firewall rules	gcloud compute firewall-rules list --filter=ALLOW:'80'

26	Verify communication is possible for http to the virtual machine	curl http://$(gcloud compute instances list --filter=name:gcelab2 --format='value(EXTERNAL_IP)')
27	View the available logs on the system	gcloud logging logs list
28	View the logs that relate to compute resources	gcloud logging logs list --filter="compute"
29	Read the logs related to the resource type	gcloud logging read "resource.type=gce_instance" --limit 5
30	Read the logs for a specific virtual machine	gcloud logging read "resource.type=gce_instance AND labels.instance_name='gcelab2'" --limit 5

b. Creating an autoscaled managed instance group using an instance template

An instance template is a resource that you can use to create virtual machine (VM) instances and managed instance groups (MIGs).Instance templates are a convenient way to save a VM instance's configuration so you can use it later to create VMs or groups of VMs.
Instance templates are designed to create instances with identical configurations. So you cannot update an existing instance template or change an instance template after you create it.
Shared VPC on interfaces other than nic0 for instance templates is supported in gcloud CLI and the API, but not in Google Cloud console.

create an instance template	gcloud compute instance-templates create INSTANCE_TEMPLATE_NAME gcloud compute instance-templates create example-template-custom \ --machine-type=e2-standard-4 \ --image-family=debian-10 \ --image-project=debian-cloud \ --boot-disk-size=250GB

If you do not provide explicit template settings, gcloud compute uses the following default values:

- Machine type: the machine type—for example, n1-standard-1
- Image: the latest Debian image
- Boot disk: a new standard boot disk named after the VM
- Network: the default VPC network
- IP address: an ephemeral external IP address

create an instance template based on an existing instance	gcloud compute instance-templates create INSTANCE_TEMPLATE_NAME \ --source-instance=SOURCE_INSTANCE \ --source-instance-zone=SOURCE_INSTANCE_ZONE \

For example, the following command creates an instance template based on **my-source-instance**, with the option to use the original image from **data-disk-a**, but set auto-delete to **true** and replace **data-disk-b** with a custom image

```
gcloud compute instance-templates create my-instance-template  \
    --source-instance=my-source-instance \
    --configure-disk=device-name=data-disk-a,instantiate-from=source-image, \
      auto-delete=true
    --configure-disk=device-name=data-disk-b,instantiate-from=custom-image, \
```

```
    custom-image=projects/cps-cloud/global/images/cos-89-16108-403-15
```

AUTO_DELETE specifies whether the disk is auto-deleted when the instance is deleted.

For example, the following command creates a new instance template named nginx-vm. A VM instance created from this template launches and runs the container image, gcr.io/cloud-marketplace/google/nginx1:TAG, when the VM starts:

```
gcloud compute instance-templates create-with-container INSTANCE_TEMPLATE_NAME \
    --container-image=CONTAINER_IMAGE
```

```
gcloud compute instance-templates create-with-container nginx-vm \
    --container-image=gcr.io/cloud-marketplace/google/nginx1:TAG
```

```
gcloud compute instance-templates create template-qa \
  --region=us-central1 \
  --subnet=subnet-us-qa
```

Replace TAG with the tag defined for a specific version of NGINX container image available on Google Cloud Marketplace.
Use the --subnet flag to place instances that are created from the template into the subnet of your choice. The --subnet flag requires the --region flag.
Note: If the image you want to use belongs to a different project, you can still use that image in your instance template, provided that the owner of that project grants the MIG access to the images. For more information, see Granting a MIG access to images.

Use deterministic instance templates to ensure identical VMs

```
gcloud compute instance-templates create example-template-with-startup \
    --image-family debian-9 \
    --image-project debian-cloud \
    --metadata startup-script='#! /bin/bash
    sudo apt install -y apache2
    scp myuser@108.59.87.185:index.php /var/www/'
```

There are two potential issues with this startup script:

- The script does not explicitly define which version of apache2 to install, and relies on the current version available in the apt repository.
- The script relies on a file hosted on a third-party that isn't versioned and could have been changed since the last time the instance template was used.

If you use an autoscaler, a non-deterministic instance template can cause your autoscaler to add new instances to a managed instance group with a different configuration, such as a different version of apache2.
If you do use startup scripts, consider updating your scripts to be deterministic. For example, create a new version of the previous template, and specify a deterministic startup script as follows:

```
gcloud compute instance-templates create example-template-with-startup-2-1-3 \
  --image-family debian-9 \
  --image-project debian-cloud \
  --metadata startup-script='#! /bin/bash
  sudo apt install -y apache2=2.2.20-1ubuntu1
  scp myuser@108.59.87.185:version_2_1_3/index.php /var/www/'
```

where "version_2_1_3" is a subdirectory containing PHP scripts for the version 2.1.3 of your service.

c. Generating/uploading a custom SSH key for instances

Open a terminal and use the ssh-keygen command with the -C flag to create a new SSH key pair.	ssh-keygen -t rsa -f ~/.ssh/KEY_FILENAME -C USERNAME -b 2048

KEY_FILENAME: the name for your SSH key file USERNAME: your username on the VM

d. Installing and configuring the Cloud Monitoring and Logging Agent

The Cloud Monitoring agent gathers system and application metrics from your VM instances and sends them to Monitoring.
The Logging agent streams logs from your VM instances and from selected third-party software packages to Cloud Logging. If your VMs are running in **Google Kubernetes Engine** or **App Engine**, the agent is already included in the VM image. Configure the Logging agent

e. Assessing compute quotas and requesting increases

Google Cloud uses **quotas** to restrict how much of a particular shared Google Cloud resource that you can use. Each quota represents a specific countable resource, such as API calls to a particular service, the number of load balancers used concurrently by your project, or the number of projects that you can create.

To view a project's quota for a particular service	`gcloud alpha services quota list \`
	`  --service=SERVICE_NAME.googleapis.com \`
	`  --consumer=projects/PROJECT_ID`

service's quota details for an organization	`gcloud alpha services quota list \`
	`  --service=SERVICE_NAME.googleapis.com \`
	`  --consumer=organizations/ORG_ID`

Requesting a higher quota limit

1. On the **Quotas** page, find the quota you want to increase in the **Quota** column.
   You can use the **Filter** search box to search for your quota.
2. Select the checkbox to the left of your quota.
3. Click create **EDIT QUOTAS**. The **Quota changes** form displays.
4. In the **Quota changes** form, enter the increased quota that you want for your project in the **New limit** field.
5. Complete any additional fields in the form, and then click **DONE**.
6. Click **SUBMIT REQUEST**.

3.2 Deploying and implementing Google Kubernetes Engine resources. Tasks include:

a. Installing and configuring the command line interface (CLI) for Kubernetes (kubectl)

Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure. The Kubernetes Engine environment consists of multiple machines (specifically Compute Engine instances) grouped to form a container cluster.
Google Kubernetes Engine (GKE) clusters are powered by the Kubernetes open source cluster management system. Kubernetes provides the mechanisms through which you interact with your container cluster. You use Kubernetes commands and resources to deploy and manage your applications, perform administrative tasks, set policies, and monitor the health of your deployed workloads.

Kubernetes draws on the same design principles that run popular Google services and provides the same benefits: automatic management, monitoring and liveness probes for application containers, automatic scaling, rolling updates, and more.

When you run a GKE cluster, you also gain the benefit of advanced cluster management features that Google Cloud provides. These include:

- Load balancing for Compute Engine instances
- Node pools to designate subsets of nodes within a cluster for additional flexibility
- Automatic scaling of your cluster's node instance count
- Automatic upgrades for your cluster's node software
- Node auto-repair to maintain node health and availability
- Logging and Monitoring with Cloud Monitoring for visibility into your cluster

Your compute zone is an approximate regional location in which your clusters and their resources live.
A cluster consists of at least one **cluster master** machine and multiple worker machines called **nodes**.
Nodes are Compute Engine virtual machine (VM) instances that run the Kubernetes processes necessary to make them part of the cluster.
After creating your cluster, you need authentication credentials to interact with it.

kubectl is a command-line tool that you can use to interact with your GKE clusters. gcloud CLI will start to require that the gke-gcloud-auth-plugin binary is installed

1. Check the gke-gcloud-auth-plugin binary version: gke-gcloud-auth-plugin --version
2. Install the kubectl component: gcloud components install kubectl
3. Verify that kubectl is installed : kubectl version

GKE uses Kubernetes objects to create and manage your cluster's resources. Kubernetes provides the Deployment object for deploying stateless applications like web servers. Service objects define rules and load balancing for accessing your application from the internet.
This Kubernetes command creates a Deployment object that represents `hello-server`. In this case, `--image` specifies a container image to deploy. The command pulls the example image from a Container Registry bucket. `gcr.io/google-samples/hello-app:1.0` indicates the specific image version to pull. If a version is not specified, the latest version is used.
To **create a Kubernetes Service**, which is a Kubernetes resource that lets you expose your application to external traffic

b. Deploying a Google Kubernetes Engine cluster with different configurations including AutoPilot, regional clusters, private clusters, etc.

To create a regional cluster in the Standard mode Creating a regional cluster.
To create a regional cluster in the Autopilot mode Creating an Autopilot cluster.

1. Go to the **Google Kubernetes Engine** page in the console.Go to Google Kubernetes Engine
2. Click add_box **Create**.
3. In the **Autopilot** section, click **Configure**.
4. Enter the **Name** for your cluster.
5. Select a region for your cluster.
6. Choose a public or private cluster.
7. (Optional) Expand **Networking Options** to specify network settings
8. (Optional) Expand **Advanced options** to specify more settings
9. Click **Create**

```
gcloud container clusters create-auto CLUSTER_NAME \
   --region REGION \
   --project=PROJECT_ID
```

By default, Autopilot clusters are public. must configure Cloud NAT. Cloud NAT lets private clusters send outbound packets to the internet and receive any corresponding established inbound response packets. Perform the following tasks to create a NAT configuration on a Cloud Router.

Create a Cloud Router	`gcloud compute routers create NAT_ROUTER \`
	`  --network NETWORK \`
	`  --region REGION \`
	`  --project=PROJECT_ID`

To create a private cluster Creating a private cluster.
Autopilot clusters, run the following command:

```
gcloud container clusters create-auto private-cluster-0 \
   --create-subnetwork name=my-subnet-0 \
   --enable-master-authorized-networks \
   --enable-private-nodes \
   --enable-private-endpoint
```

c. Deploying a containerized application to Google Kubernetes Engine

Deploy an app to a GKE cluster

Set the default project
Set the default zone
To **set your default compute zone**
	`gcloud config set project PROJECT_ID`
	`gcloud config set compute/zone COMPUTE_ZONE`
	`gcloud config set compute/zone us-central1-a`

To **create a cluster** replacing [CLUSTER-NAME] example:my-cluster)	`gcloud container clusters create [CLUSTER-NAME]`
To **authenticate the cluster**	`gcloud container clusters get-credentials [CLUSTER-NAME]`
To **create a new Deployment**	`kubectl create deployment hello-server \` `--image=us-docker.pkg.dev/google-samples/containers/gke/hello-app:1.0`
To **create a Kubernetes Service**	`kubectl expose deployment hello-server --type=LoadBalancer --port 8080`
To **inspect** the hello-server Service	`kubectl get service`
To **delete** the cluster	`gcloud container clusters delete [CLUSTER-NAME]`

d. Configuring Google Kubernetes Engine monitoring and logging

When you create a GKE cluster running on Google Cloud, Cloud Logging and Cloud Monitoring are enabled by default and provide observability specifically tailored for Kubernetes.
Note: For GKE Autopilot clusters, you cannot disable the Cloud Monitoring and Cloud Logging integration.
Cloud Operations for GKE is designed to monitor GKE clusters. It manages Monitoring and Logging services together and features a Cloud Operations for GKE dashboard

Configuring Cloud Operations for GKE

--logging=NONE : collect no logs
--logging=SYSTEM,WORKLOAD collect both system and workload
--logging=SYSTEM only system logs
	`gcloud container clusters create [CLUSTER_NAME] \`
	`  --zone=[ZONE] \`
	`  --project=[PROJECT_ID] \`
	`  --logging=SYSTEM`

--monitoring=NONE :collect no metrics
--monitoring=SYSTEM : collect system metrics
	`gcloud container clusters create [CLUSTER_NAME] \`
	`  --zone=[ZONE] \`
	`  --project=[PROJECT_ID] \`
	`  --monitoring=SYSTEM`

enable Managed Service for Prometheus by using	`gcloud container clusters create [CLUSTER_NAME] \`
	`  --zone=[ZONE] \`
	`  --project=[PROJECT_ID] \`
	`  --enable-managed-prometheus`

Configuring monitoring and logging for an existing cluster
 	
--enable-managed-prometheus or --disable-managed-prometheus flags.
	`gcloud container clusters update [CLUSTER_NAME] \`
	`  --zone=[ZONE] \`
	`  --logging=NONE`
	`  --monitoring=NONE`
	`gcloud container clusters update [CLUSTER_NAME] \`
	`  --zone=[ZONE] \`
	`  --enable-managed-prometheus`

3.3 Deploying and implementing Cloud Run and Cloud Functions resources. Tasks include, where applicable:

a. Deploying an application and updating scaling configuration, versions, and traffic splitting

Container instance autoscaling

Rollbacks, gradual rollouts, and traffic migration

Split traffic between multiple revisions : gcloud run services update-traffic SERVICE --to-revisions LIST

How-to guides

b. Deploying an application that receives Google Cloud events (e.g., Pub/Sub events, Cloud Storage object change notification events)

Eventarc allows you to build event-driven architectures without having to implement, customize, or maintain the underlying infrastructure. Eventarc offers a standardized solution to manage the flow of state changes, called *events*, between decoupled microservices.

Events are things that happen within your cloud environment that you might want to take action on. When an event triggers the execution of your Cloud Function, data associated with the event is passed via the function's parameters.The type of event determines the parameters passed to your function.Event-driven functions come in two formats: background functions and CloudEvent functions.

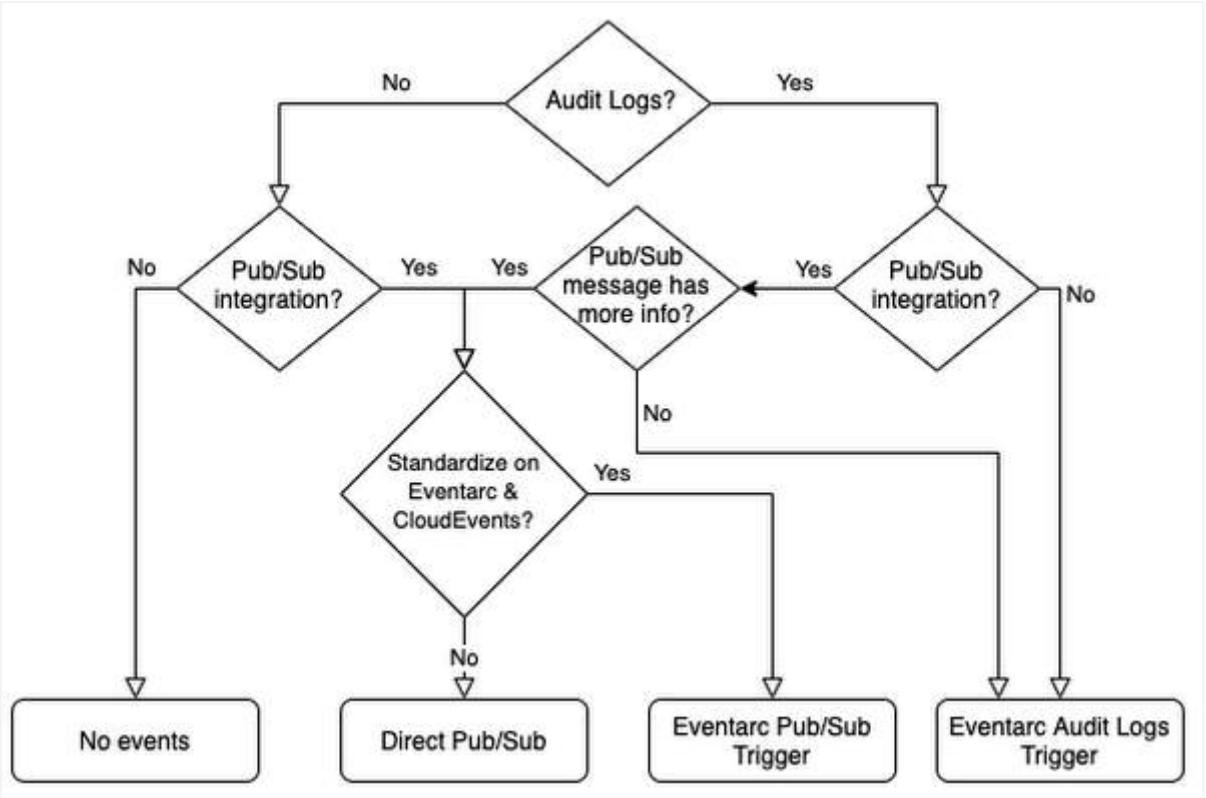
Creating a response to an event is done with a Trigger. A trigger is a declaration that you are interested in a certain event or set of events. Binding a function to a trigger allows you to capture and act on events.

1. Calling Pub/Sub Trigger Functions : **Cloud Pub/Sub Tutorial**
2. Calling Cloud Storage Trigger Functions : **Cloud Storage Tutorial**
3. Create a trigger for Cloud Run
4. Receive direct events from Cloud Storage (console)
5. Receive events using Pub/Sub messages (gcloud CLI)

The three main ways of receiving events in Cloud Run are:

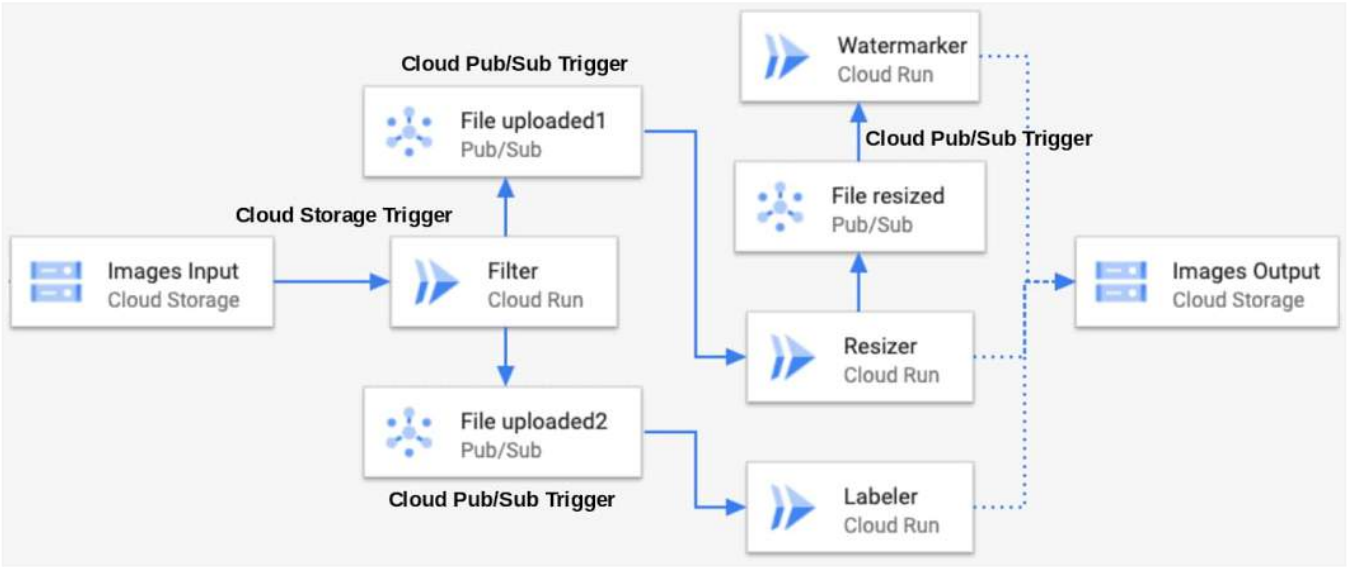
1. Audit Logs via Eventarc

2. Pub/Sub via Eventarc
3. Pub/Sub direct



Eventarc [enables any number of use cases] for applications running on Cloud Run, among them:

- Use a **Cloud Storage event (via Cloud Audit Logs)** to trigger a data processing pipeline
- Use a **BigQuery event (via Cloud Audit Logs)** to initiate downstream processing in Cloud Run each time a job is completed
- Use an event from **custom sources (publishing to Cloud Pub/Sub)** to signal between microservices, leveraging the same standardized infrastructure for any asynchronous coordination of services



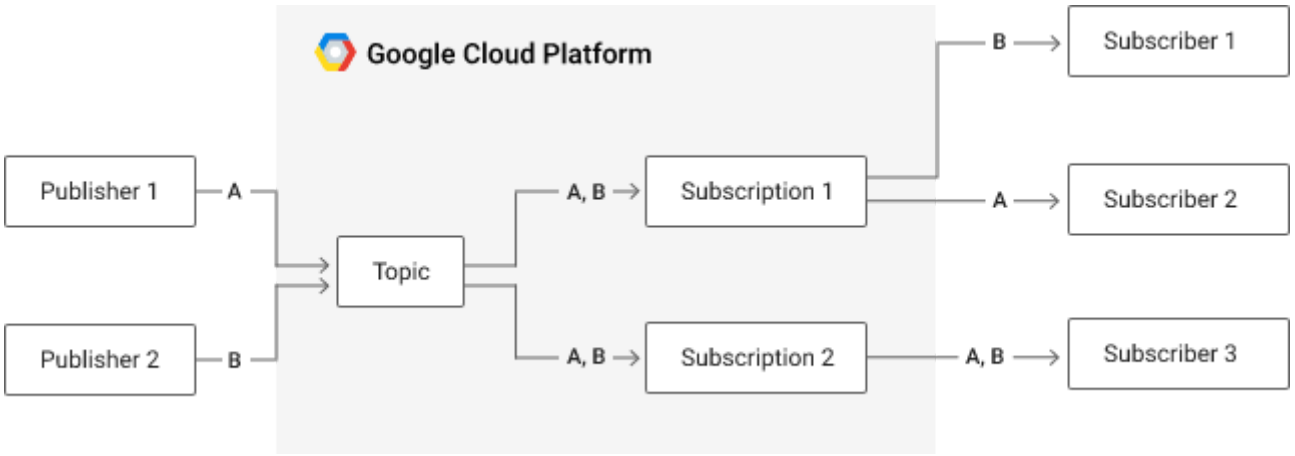
3.4 Deploying and implementing data solutions. Tasks include:

a. Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)

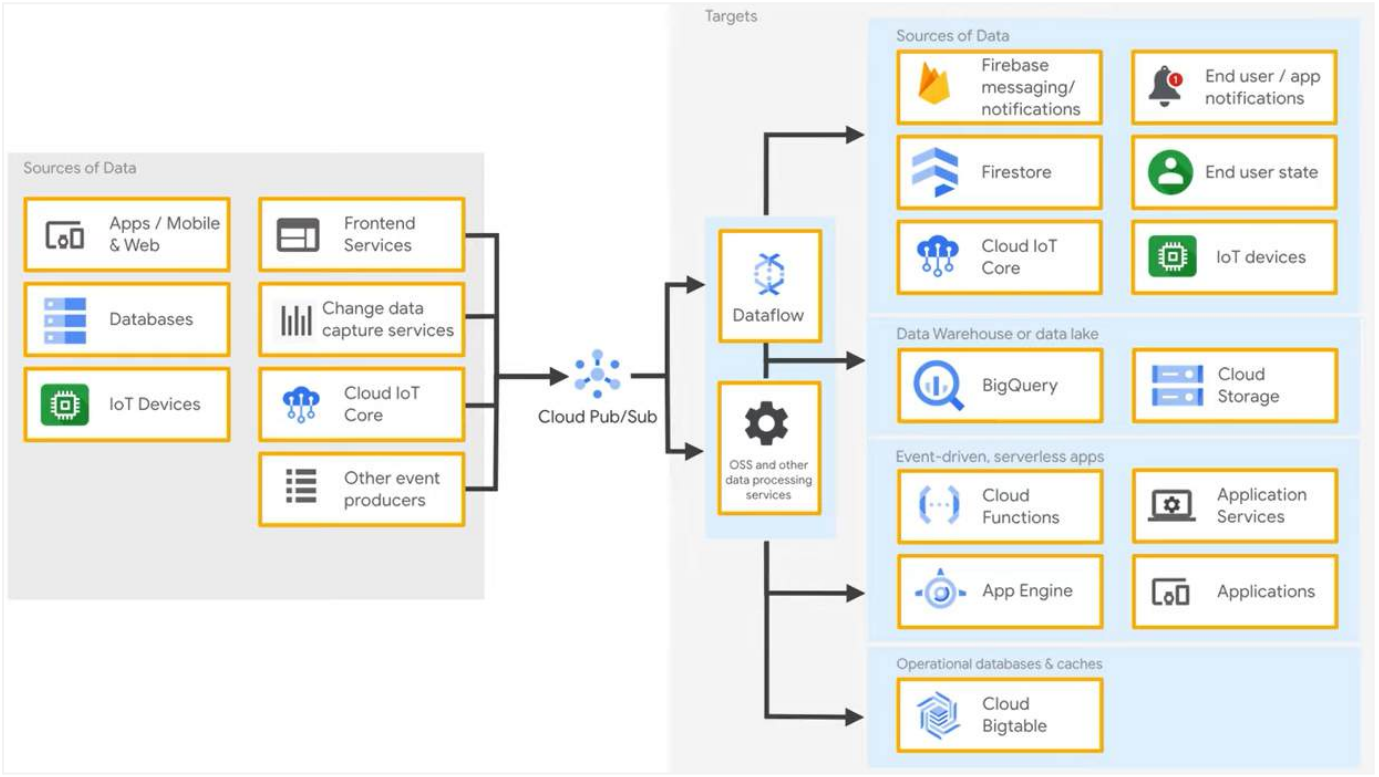
[Pub/Sub] is an asynchronous messaging service designed to be highly reliable and scalable.

Pub/Sub is a *publish/subscribe* (*Pub/Sub*) *service*: a messaging service where the senders of messages are decoupled from the receivers of messages. There are several key concepts in a Pub/Sub service:

- *Message*: the data that moves through the service.
- *Topic*: a named entity that represents a feed of messages.
- *Subscription*: a named entity that represents an interest in receiving messages on a particular topic.
- *Publisher* (also called a producer): creates messages and sends (publishes) them to the messaging service on a specified topic.
- *Subscriber* (also called a consumer): receives messages on a specified subscription.

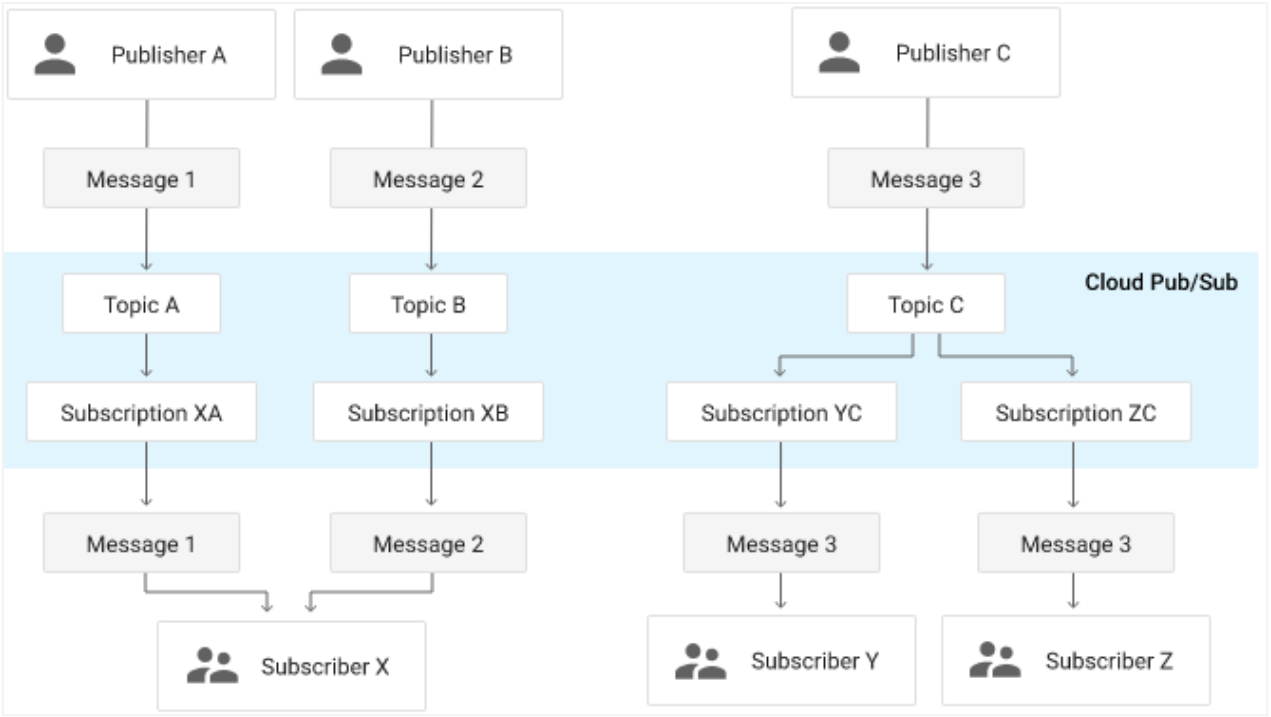


Pub/Sub allows services to communicate asynchronously, with latencies on the order of 100 milliseconds.Pub/Sub is used for streaming analytics and data integration pipelines to ingest and distribute data. Pub/Sub enables you to create systems of event producers and consumers, called **publishers** and **subscribers**. Publishers communicate with subscribers asynchronously by broadcasting events, rather than by synchronous remote procedure calls (RPCs).



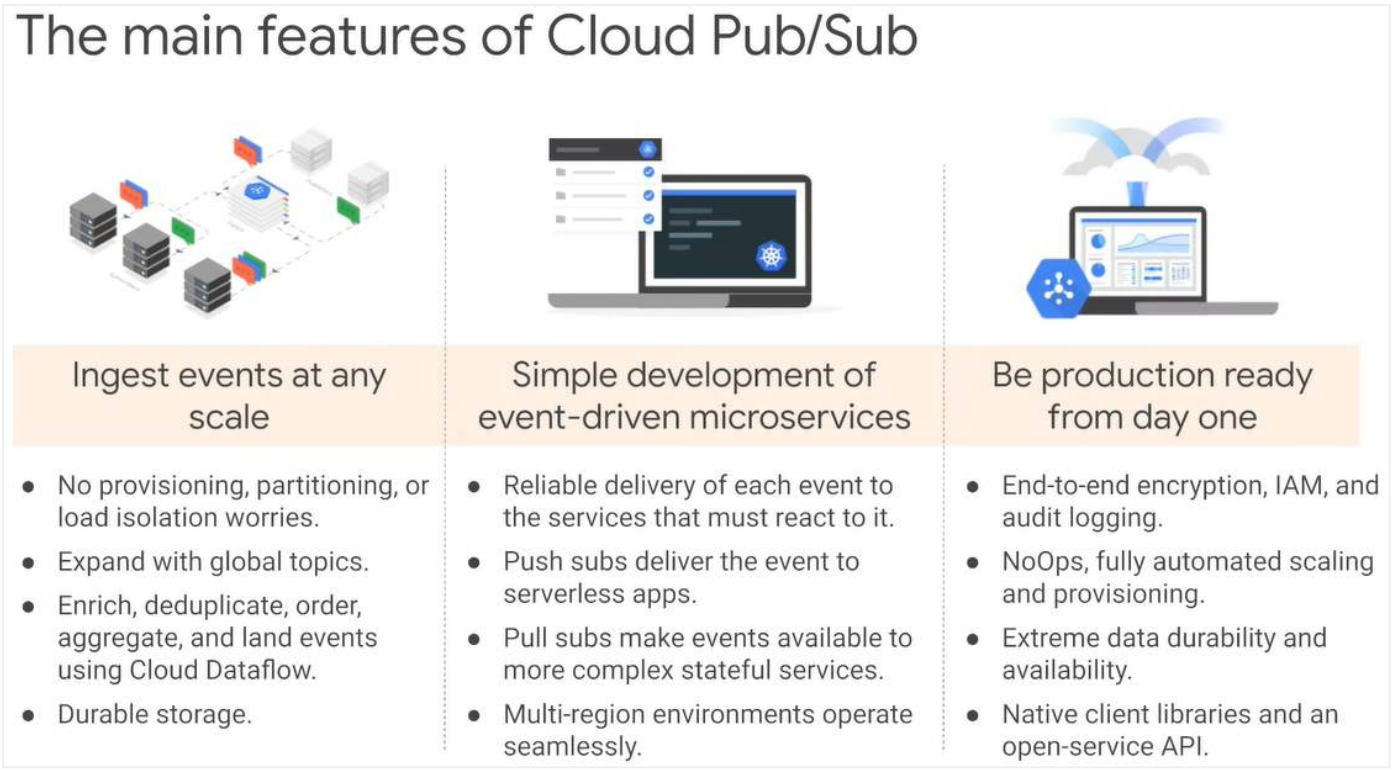
Publishers send events to the Pub/Sub service, without regard to how or when these events are to be processed. Pub/Sub then delivers events to all the services that react to them.
Pub/Sub consists of two services: 1] **Pub/Sub service.** 2] **Pub/Sub Lite service.**
Pub/Sub is intended for service-to-service communication rather than communication with end-user or IoT clients. Pub/Sub and Pub/Sub Lite are both horizontally scalable and managed messaging services.

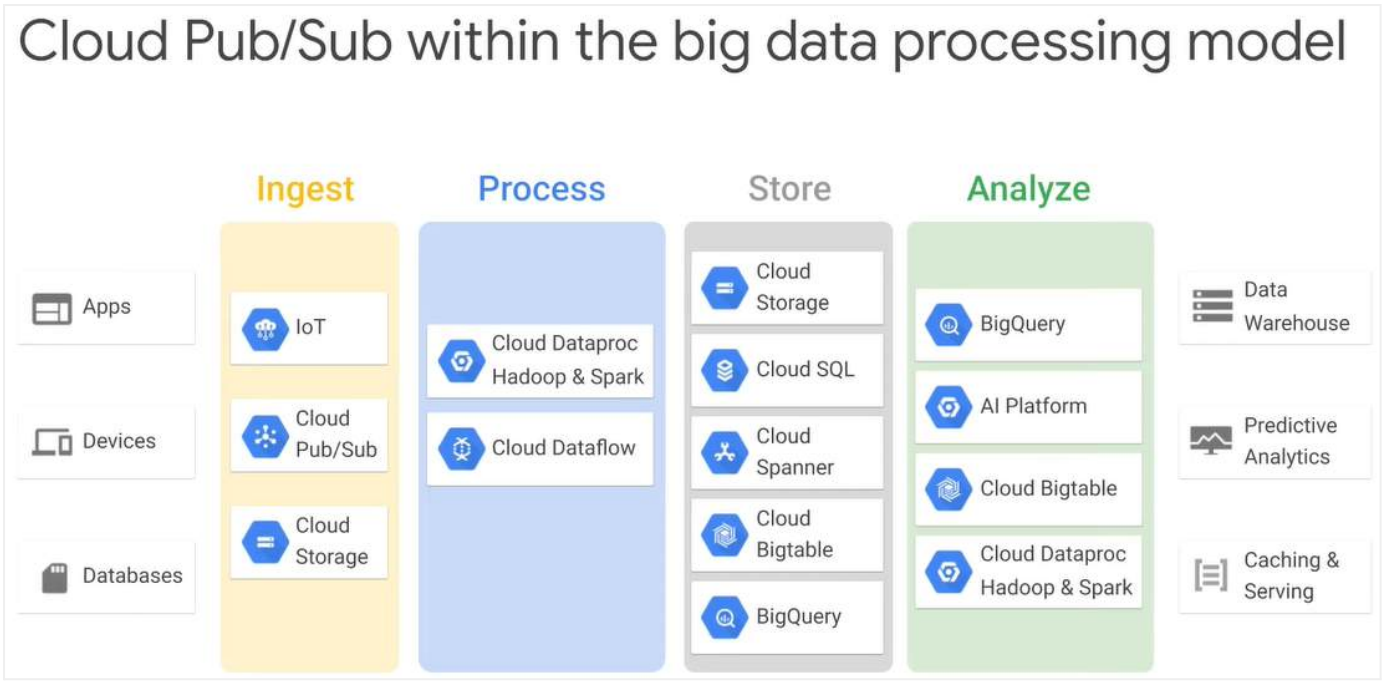
Note: A partner-managed Apache Kafka service, Confluent Cloud is available. If you're considering a migration from Kafka to Pub/Sub, consult this migration guide.Partition-based parallelism is used by Pub/Sub Lite, but not Pub/Sub.
Publisher-subscriber relationships can be one-to-many (fan-out), many-to-one (fan-in), and many-to-many, as shown in the following diagram:



A pub/sub is a messaging service where the senders of messages are decoupled from the receivers of messages. When a message is sent or posted, a subscription is required for a receiver to be alerted and receive the message.
Pub/Sub: A Google-Scale Messaging Service : The system is designed to be *horizontally scalable*, where an increase in the number of topics, subscriptions, or messages can be handled by increasing the number of instances of running servers. Pub/Sub is divided into two primary parts: the *data plane*, which handles moving messages between publishers and subscribers, and the *control plane*, which handles the assignment of publishers and subscribers to servers on the data plane. The servers in the data plane are called *forwarders*, and the servers in the control plane are called *routers*.

The Google Cloud Pub/Sub service allows applications to exchange messages reliably, quickly, and asynchronously. To accomplish this, a data producer publishes messages to a Cloud Pub/Sub topic. A subscriber client then creates a subscription to that topic and consumes messages from the subscription.
Cloud Pub/Sub persists messages that could not be delivered reliably for up to seven days.
Google Cloud Pub/Sub is an asynchronous global messaging service. There are three terms in Pub/Sub that appear often: *topics*, *publishing*, and *subscribing*.
A topic is a shared string that allows applications to connect with one another through a common thread. Publishers push (or publish) a message to a Cloud Pub/Sub topic. Subscribers will then make a *subscription* to that thread, where they will either pull messages from the topic or configure webhooks for push subscriptions. Every subscriber must acknowledge each message within a configurable window of time.
In sum, a publisher creates and sends messages to a topic and a subscriber creates a subscription to a topic to receive messages from it.
Pub/Sub comes preinstalled in the Cloud Shell, so there are no installations or configurations required to get started with this service.

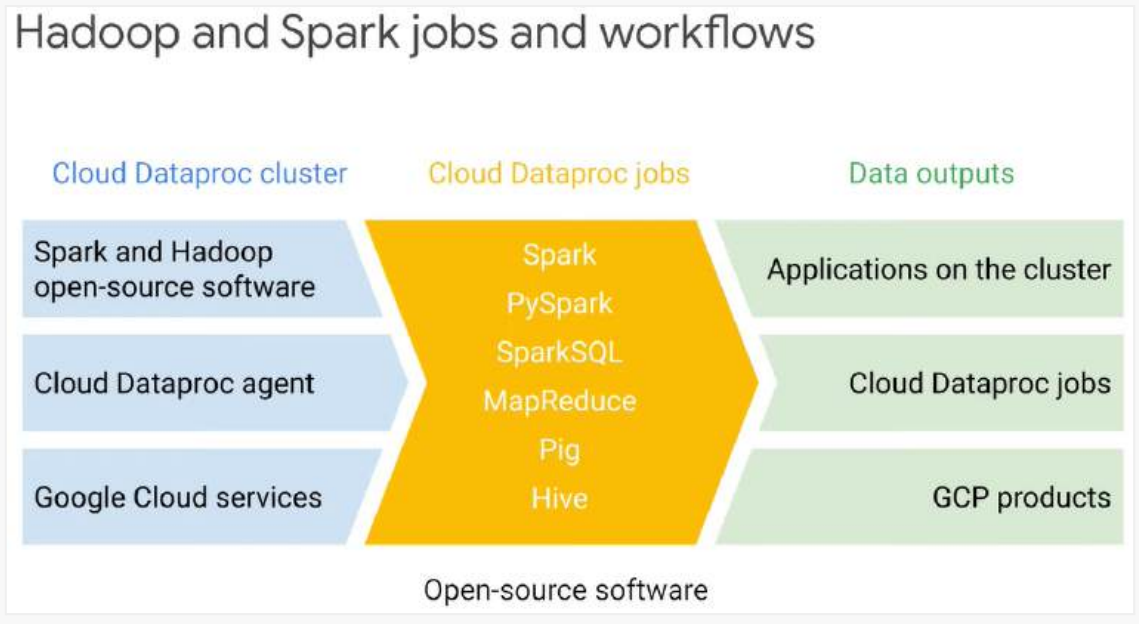




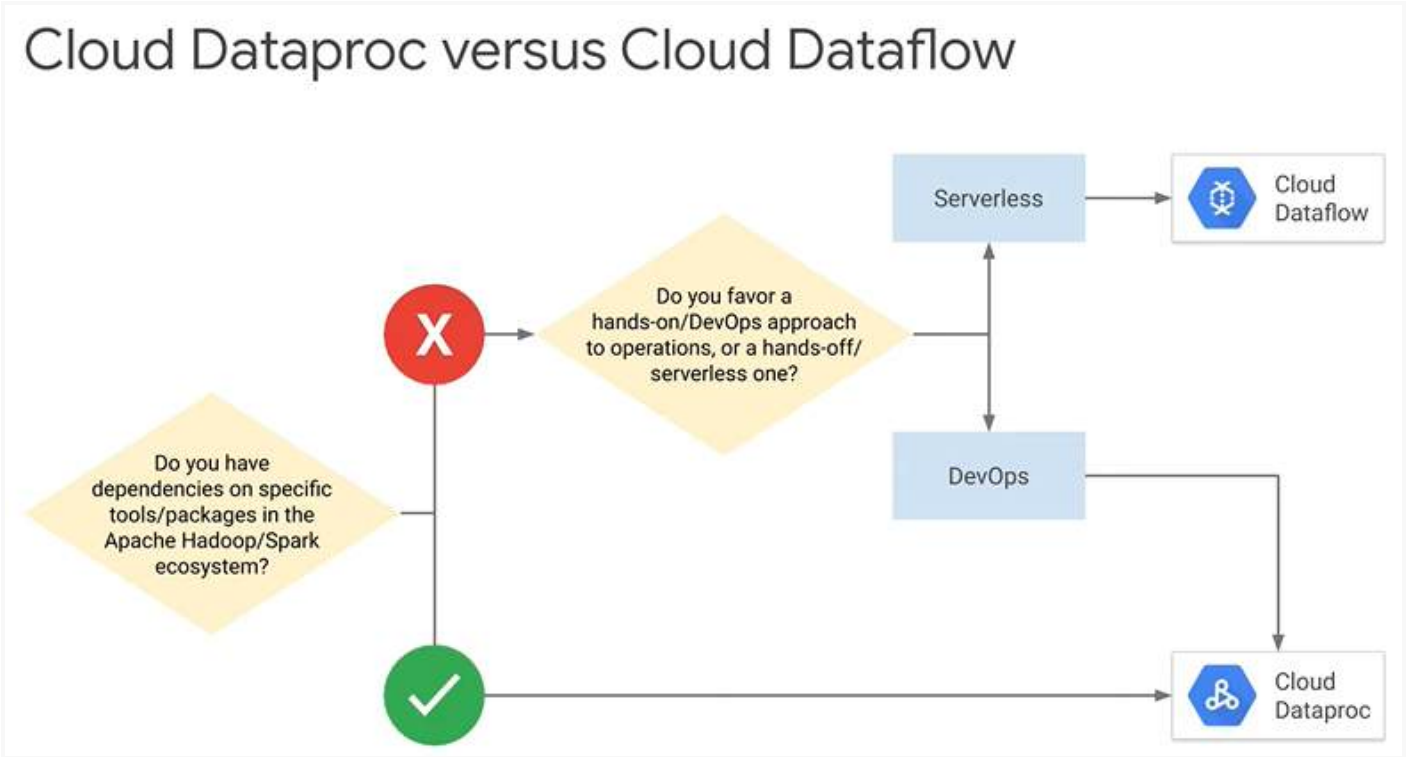
A managed messaging system ingests, transforms, and analyzes massive amounts of data. It also orchestrates complex business processes.
Cloud Pub/Sub passes messages between data gathering and processing systems.

Cloud Dataprep by Trifacta is an intelligent data service for visually exploring, cleaning, and preparing data for analysis. Cloud Dataprep is serverless and works at any scale. There is no infrastructure to deploy or manage. Easy data preparation with clicks and no code! Cloud Dataprep uses a `flow` workspace to access and manipulate datasets.

Cloud Dataproc is a cloud-based managed Spark and Hadoop service.
Cloud Dataflow allows you to perform extract, transform, and load operations.



To set the Region	gcloud config set dataproc/region us-east1
To create a cluster	gcloud dataproc clusters create example-cluster --worker-boot-disk-size 500
To submit a sample Spark job	gcloud dataproc jobs submit spark --cluster example-cluster \ --class org.apache.spark.examples.SparkPi \ --jars file:///usr/lib/spark/examples/jars/spark-examples.jar -- 1000
To change the number of workers in the cluster	gcloud dataproc clusters update example-cluster --num-workers 4



Dataproc is a managed Spark and Hadoop service that lets you take advantage of open source data tools for batch processing, querying, streaming, and machine learning. Dataproc automation helps you create clusters quickly, manage them easily, and save money by turning clusters off when you don't need them.

Cloud Dataproc is a fast, easy-to-use, fully-managed cloud service for running Apache Spark and Apache Hadoop clusters in a simpler, more cost-efficient way.
Cloud Pub/Sub to BigQuery template, which reads messages written in JSON from a Pub/Sub topic and pushes them to a BigQuery table. You can find the documentation for this template in the Get started with Google-provided templates Guide.

To create a dataset	bq mk taxirides
To instantiate a BigQuery table.	bq mk \ --time_partitioning_field timestamp \ --schema ride_id:string,point_idx:integer,latitude:float,longitude:float,\ timestamp:timestamp,meter_reading:float,meter_increment:float,ride_status:string,\ passenger_count:integer -t taxirides.realtime
create a bucket	export BUCKET_NAME=<your-unique-name> gsutil mb gs://$BUCKET_NAME/

Dataflow is a managed service for executing a wide variety of data processing patterns.The Apache Beam SDK is an open source programming model that enables you to develop both batch and streaming pipelines. You create your pipelines with an Apache Beam program and then run them on the Dataflow service. Cloud Dataflow is a distribution of Apache Beam.

Dataflow Technical guidance

Cloud SQL	Firestore	BigQuery	Cloud Spanner	Cloud Bigtable	Cloud Storage

b. Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

An uploaded object consists of the data you want to store along with any associated metadata.

```
gcloud alpha storage cp OBJECT_LOCATION gs://DESTINATION_BUCKET_NAME/
gcloud alpha storage cp gs://BUCKET_NAME/OBJECT_NAME SAVE_TO_LOCATION
gsutil cp OBJECT_LOCATION gs://DESTINATION_BUCKET_NAME/
gsutil cp gs://BUCKET_NAME/OBJECT_NAME SAVE_TO_LOCATION
```

- OBJECT_LOCATION is the local path to your object. For example, Desktop/dog.png.
- DESTINATION_BUCKET_NAME is the name of the bucket to which you are uploading your object. For example, my-bucket.
- SAVE_TO_LOCATION is the local path where you are saving your object. example, Desktop/Images.

Storage Transfer Service: Transfer data quickly and securely between object and file storage across Google Cloud, Amazon, Azure, on-premises, and more.
The BigQuery Data Transfer Service automates data movement into BigQuery on a scheduled, managed basis. Your analytics team can lay the foundation for a BigQuery data warehouse without writing a single line of code. Access the BigQuery Data Transfer Service using the:

- console
- bq command-line tool : is a Python-based command-line tool for BigQuery.
- BigQuery Data Transfer Service API

Loading CSV data from Cloud Storage : When you load CSV data from Cloud Storage, you can load the data into a new table or partition, or you can append to or overwrite an existing table or partition. When your data is loaded into BigQuery, it is converted into columnar format for Capacitor (BigQuery's storage format).the dataset that contains the table must be in the same regional or multi- regional location as the Cloud Storage bucket.

Google Cloud APIs are programmatic interfaces to Google Cloud Platform services. call the APIs using curl commands.When an application calls a Cloud API, the project that owns the application credentials is called the *client project*, and the project that owns the target resource is called the *resource project*.oauth2l can accept any application credentials and use them to make calls to Google Cloud APIs using the curl command.

```
$ gcloud services enable pubsub.googleapis.com
$ gcloud services disable pubsub.googleapis.com
oauth2l curl --credentials ./creds.json --scope cloud-platform --url
https://pubsub.googleapis.com/v1/projects/my-project-id/topics -- -v
```

Streaming with Pub/Sub : Dataflow compliments Pub/Sub's scalable, at-least-once delivery model with message deduplication, exactly-once processing, and generation of a data watermark from timestamped events. To use Dataflow, write your pipeline using the Apache Beam SDK and then execute the pipeline code on the Dataflow service.
stream of data into a Pub/Sub topic: Cloud Storage Text to Pub/Sub (Batch) >> creates a batch pipeline that reads records from text files stored in Cloud Storage and publishes them to a Pub/Sub topic.

```
gcloud dataflow jobs run JOB_NAME \
   --gcs-location gs://dataflow-templates/VERSION/GCS_Text_to_Cloud_PubSub \
   --region REGION_NAME \
   --parameters \
inputFilePattern=gs://BUCKET_NAME/files/*.json,\
outputTopic=projects/PROJECT_ID/topics/TOPIC_NAME
```

- VERSION: the version of the template, **latest** to use the latest version of the template
- TOPIC_NAME: your Pub/Sub topic name
- BUCKET_NAME: the name of your Cloud Storage bucket

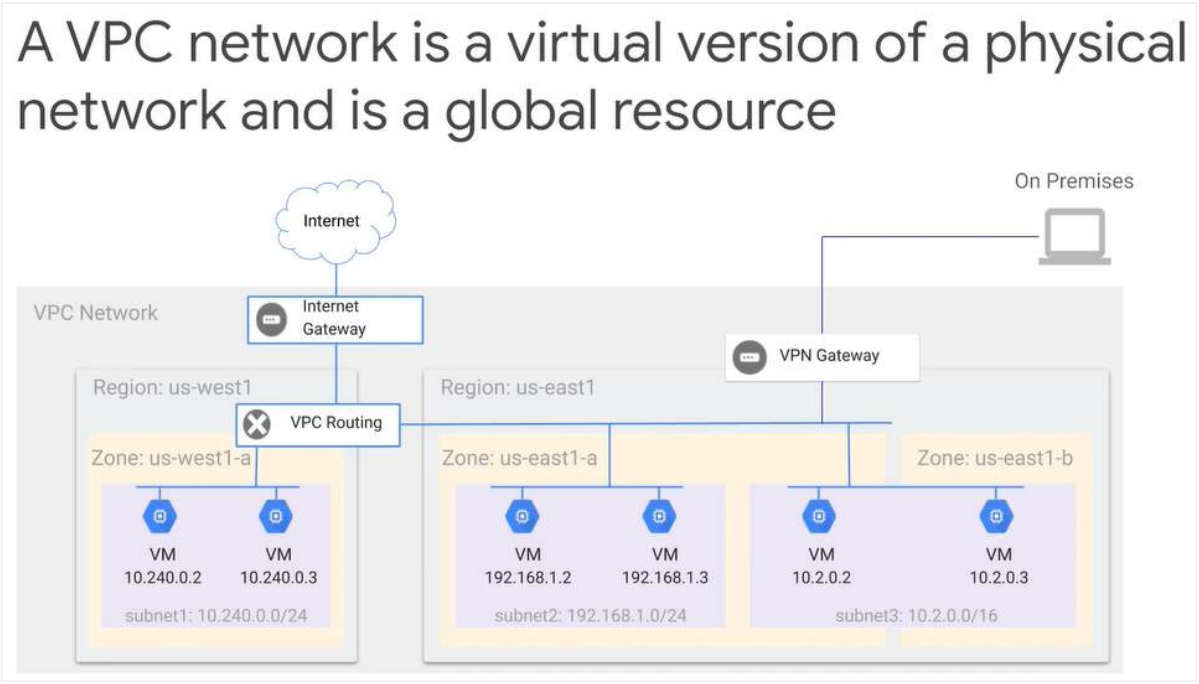
Stream messages from Pub/Sub by using Dataflow

3.5 Deploying and implementing networking resources. Tasks include:
a. Creating a VPC with subnets (e.g., custom-mode VPC, shared VPC)

A Virtual Private Cloud (VPC) network is a virtual version of a physical network, implemented inside of Google's production network. Each subnet defines a range of IPv4 addresses. Subnets in custom mode VPC networks can also have a range of IPv6 addresses. VPC networks can be connected to other VPC networks in different projects or organizations by using VPC Network Peering. VPC networks can be securely connected in hybrid environments by using Cloud VPN or Cloud Interconnect. VPC networks support IPv4 and IPv6 unicast addresses. VPC networks do **not** support broadcast or multicast addresses *within* the network.
When an *auto mode* VPC network is created, one subnet from each region is automatically created within it.
When a *custom mode* VPC network is created, no subnets are automatically created.

Subnetting is the process of stealing bits from the HOST part of an IP address in order to divide the larger network into smaller sub-networks called subnets.

A VPC network is a virtual version of a physical network and is a global resource



- VPC networks connect Google Cloud resources to each other and to the internet
- Segmenting networks
- Using firewall rules to restrict access to instances
- Creating static routes to forward traffic to specific destinations
- Google VPC networks are global and can have subnets in any Google Cloud region worldwide

A /16 range provides 65,536 IP addresses

CIDR IP address totals						
/16	/17	/18	/19	/20	/21	/22
65,536	**32,768**	**16,384**	**8,192**	**4,096**	**2,048**	**1,024**
/23	/24	/25	/26	/27	/28	
512	**256**	**128**	**64**	**32**	**16**	

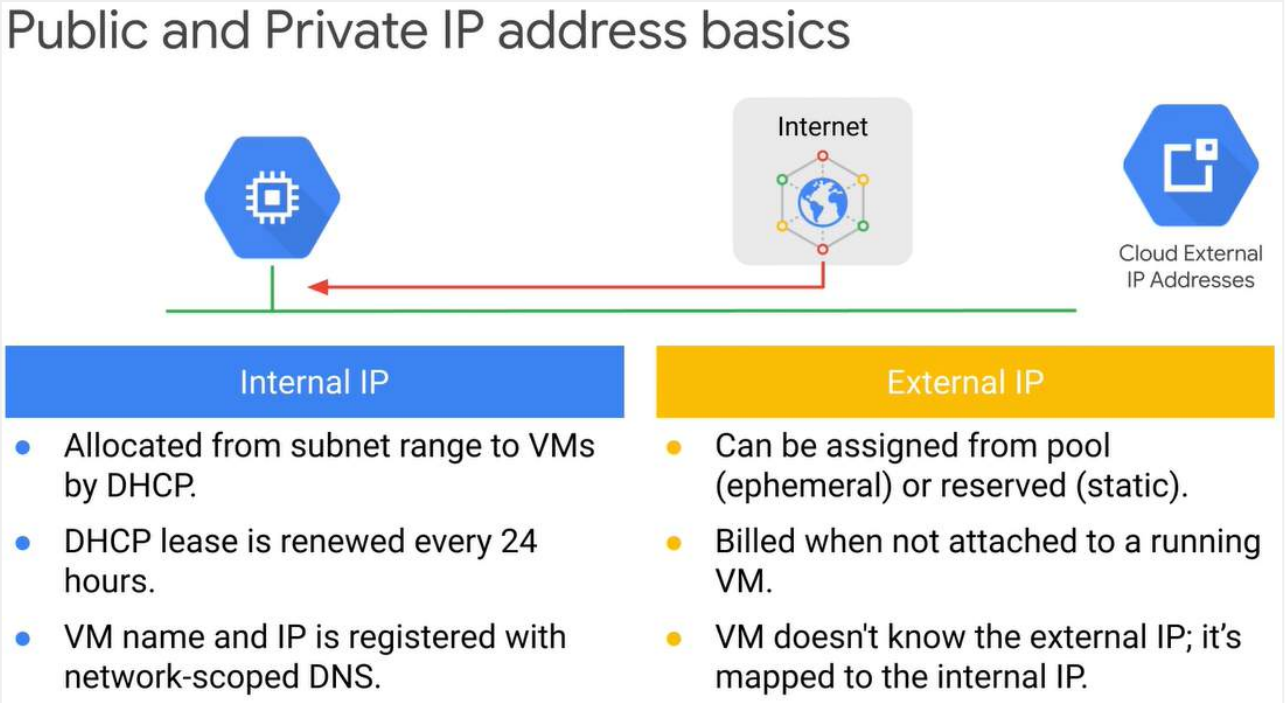
Auto versus custom networks

Auto subnet mode

- One subnet from each region is automatically created.
- Set of predefined IP ranges
- Comes with default firewall rules
- Expandable up to /16 only
- Good for isolated use cases (Proof of concepts (PoCs), testing, etc.)

Custom subnet mode

- No subnets are automatically created
- Subnets and IP ranges are defined
- No default firewalls rules
- Expandable to any RFC 1918 size
- Recommended for Production environments



Each Google Cloud project has a default network with subnets, routes, and firewall rules. Firewall rules allow you to control which packets are allowed to travel to which destinations. Every VPC network has two implied firewall rules that block all incoming connections and allow all outgoing connections.

Create a custom mode VPC network with only IPv4 subnets

```
gcloud compute networks create NETWORK \
  --subnet-mode=custom \
  --bgp-routing-mode=DYNAMIC_ROUTING_MODE \
  --mtu=MTU
```

- DYNAMIC_ROUTING_MODE can be either global or regional to control the route advertisement behavior of Cloud Routers in the network. For more information, refer to dynamic routing mode.
- MTU is the maximum transmission unit of the network. MTU can be set to anything from 1300 through 8896 (default: 1460). Review the maximum transmission unit overview before setting the MTU to higher than 1460. Next, add subnets to your network.

```
gcloud compute networks subnets create SUBNET \
  --network=NETWORK \
  --range=PRIMARY_RANGE \
  --region=REGION
```

- PRIMARY_RANGE is the primary IPv4 range for the new subnet, in CIDR notation. For more information, see IPv4 subnet ranges. Dual-stack subnets have both IPv4 and IPv6 address ranges. --stack-type=IPV4_IPV6 \

List the networks in your project	gcloud compute networks list
List all subnets in all VPC networks particular VPC network Describe the subnet	gcloud compute networks subnets list gcloud compute networks subnets list \ --network=NETWORK gcloud compute networks subnets describe SUBNET \ --region=REGION
Change a subnet's stack type to	gcloud compute networks subnets update SUBNET \

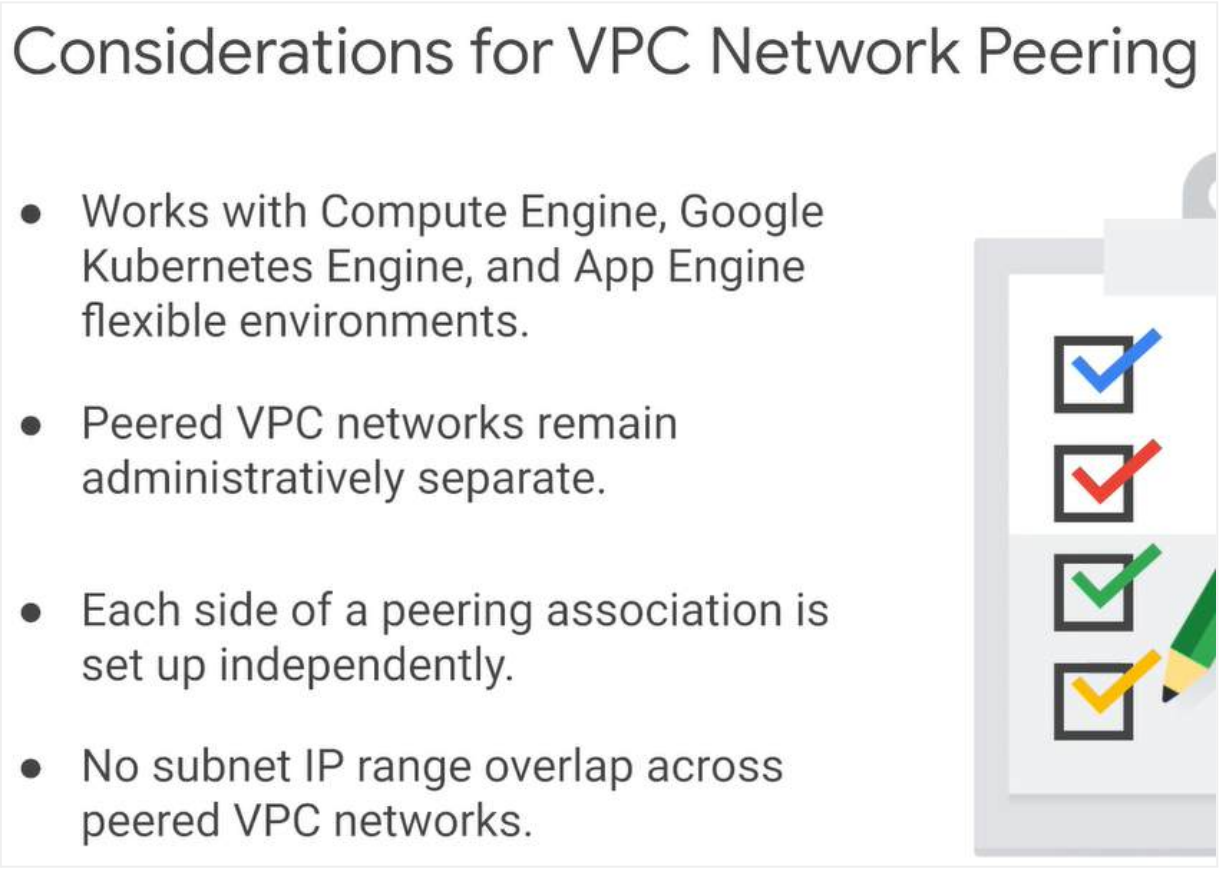
dual-stack	--stack-type=IPV4_IPV6 \ --ipv6-access-type=IPv6_ACCESS_TYPE \ --region=REGION

- IPv6_ACCESS_TYPE is the IPv6 access type of the subnet. It can be EXTERNAL or INTERNAL.

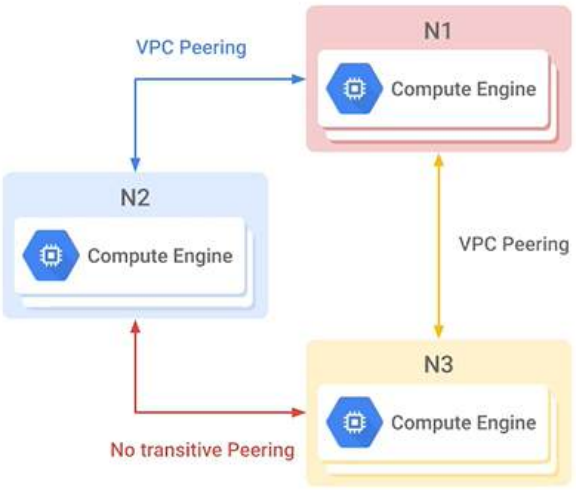
Shared VPC allows an organization to connect resources from multiple projects to a common Virtual Private Cloud (VPC) network, so that they can communicate with each other securely and efficiently using internal IPs from that network. Shared VPC is also referred to as "XPN" in the API. A project that participates in Shared VPC is either a *host project* or a *service project*. *If a user has the Shared VPC Admin role at the folder level, they also need the* ***Compute Network Viewer*** *role or another role that has the compute.subnetworks.list permission in order to manage Shared VPC from the Google Cloud console.*
Setting up Shared VPC : You need to attach service projects to the host project to share selected networks and subnets with the service projects.

Authenticate to gcloud as a Shared VPC Admin	gcloud auth login SHARED_VPC_ADMIN
Enable Shared VPC for the project that you need to become a host project If you have Shared VPC Admin role at the organization level / at the folder level (2)	gcloud compute shared-vpc enable HOST_PROJECT_ID gcloud beta compute shared-vpc enable HOST_PROJECT_ID
	gcloud compute shared-vpc organizations list-host-projects ORG_ID
	gcloud auth revoke SHARED_VPC_ADMIN

SHARED_VPC_ADMIN is name of the Shared VPC Admin. Replace HOST_PROJECT_ID with the ID of the project. ORG_ID with your organization ID (determined by gcloud organizations list).



Directly peered networks can communicate



Shared VPC versus VPC peering

Consideration	Shared VPC	VPC Network Peering
Across organizations	No	Yes
Within project	No	Yes
Network administration	Centralized	Decentralized

to create the **privatenet** network	gcloud compute networks create privatenet --subnet-mode=custom
to create the **privatesubnet-us** subnet	gcloud compute networks subnets create privatesubnet-us --network=privatenet --region=us-east1 --range=172.16.0.0/24
create the **privatesubnet-eu** subnet	gcloud compute networks subnets create privatesubnet-eu --network=privatenet --region=europe-west4 --range=172.20.0.0/20
o list the available VPC subnets	gcloud compute networks subnets list --sort-by=NETWORK
To create firewall rule	gcloud compute firewall-rules create privatenet-allow-icmp-ssh-rdp --direction=INGRESS --priority=1000 --network=privatenet --action=ALLOW --rules=icmp,tcp:22,tcp:3389 --source-ranges=0.0.0.0/0

Note: You are able to ping the external IP address of all VM instances, even though they are either in a different zone or VPC network. This confirms public access to those instances is only controlled by the **ICMP** firewall rules that you established earlier.

Note: You are able to ping the internal IP address of **mynet-eu-vm** because it is on the same VPC network as the source of the ping (**mynet-us-vm**), even though both VM instances are in separate zones, regions and continents!

VPC networks are by default isolated private networking domains. However, no internal IP address communication is allowed between networks, unless you set up mechanisms such as VPC peering or VPN. Every instance in a VPC network has a default network interface. You can create additional network interfaces attached to your VMs. Multiple network interfaces enable you to create configurations in which an instance connects directly to several VPC networks (up to 8 interfaces, depending on the instance's type).

Note: The number of interfaces allowed in an instance is dependent on the instance's machine type and the number of vCPUs. The e2-standard-4 allows up to 4 network interfaces. Refer to the Maximum number of network interfaces section of the Google Cloud Guide

VPC networks have an internal DNS service that allows you to address instances by their DNS names rather than their internal IP addresses.

Note: The **default-allow-internal** firewall rule allows traffic on all protocols/ports within the **default** network.

- **Network Admin**: Permissions to create, modify, and delete networking resources, except for firewall rules and SSL certificates. As expected, the **Network Admin** role has permissions to list but not modify/delete firewall rules.
- **Security Admin**: Permissions to create, modify, and delete firewall rules and SSL certificates.

Note: The **Compute Engine default service account** does not have the right permissions to allow you to list or delete firewall rules.
Shared VPC allows an organization to connect resources from multiple projects to a common VPC network.
Direct Peering provides a direct connection whilst Carrier Peering provides connectivity through a supported partner.
Load balancing can be used to take advantage of an augmented infrastructure.

b. Launching a Compute Engine instance with custom network configuration (e.g., internal-only IP address, Google private access, static external and private IP address, network tags)

Configuring network tags To assign new tags to an instance	gcloud compute instances add-tags INSTANCE_NAME \ --zone ZONE \ --tags TAGS

Create a VM instance with a specific internal IP address	gcloud compute instances create VM_NAME --private-network-ip IP_ADDRESS
list all static IP address, including external IP address and internal IP address.	gcloud compute addresses list

Configuring VMs for networking use cases :

Set up one instance with an external (static or ephemeral) IP address.	gcloud compute instances create gateway-instance --project=project_id --zone=zone --network-interface=network-tier=PREMIUM,subnet=default --scopes=https://www.googleapis.com/auth/cloud-platform
Set up one or more instances without external IP addresses	gcloud compute instances create hidden-instance --project=project_id --zone=zone --network-interface=network-tier=PREMIUM,subnet=default,no-address --scopes=https://www.googleapis.com/auth/cloud-platform

Configuring Private Google Access : Private Google Access also allows access to the external IP addresses used by App Engine, including third-party App Engine-based services. Private Google Access is enabled on a per-subnet basis.

To list the subnets for a particular network	gcloud compute networks subnets list --filter=NETWORK_NAME
enable Private Google Access	gcloud compute networks subnets update SUBNET_NAME \ --region=REGION \ --enable-private-ip-google-access
Verify that Private Google Access is enabled	gcloud compute networks subnets describe SUBNET_NAME \ --region=REGION \ --format="get(privateIpGoogleAccess)"

When creating a new subnet, use the --enable-private-ip-google-access flag to enable Private Google Access	gcloud compute networks subnets create SUBNET_NAME \
	  --region=REGION \
	  --network=NETWORK_NAME \
	  --range=PRIMARY_IP_RANGE \
	  [--stack-type=STACK_TYPE] \
	  [--ipv6-access-type=IPv6_ACCESS_TYPE] \
	  --enable-private-ip-google-access

disable Private Google Access	gcloud compute networks subnets update SUBNET_NAME \
	  --region=REGION \
	  --no-enable-private-ip-google-access

- NETWORK_NAME: the name of the VPC network that contains the subnet
- PRIMARY_IP_RANGE: the subnet's primary IP address range
- STACK_TYPE is the stack type for the subnet: IPV4_ONLY or IPV4_IPV6.
- IPv6_ACCESS_TYPE is the IPv6 access type: EXTERNAL or INTERNAL. Only specify the IPv6 access type if you have also specified --stack-type=IPV4_IPV6.

c. Creating ingress and egress firewall rules for a VPC (e.g., IP subnets, network tags, service accounts)

VPC Service Controls use ingress and egress rules to allow access to and from the resources and clients protected by service perimeters. The ingress and egress rule blocks specify the direction of allowed access to and from different identities and resources. Rules allow you to privately and efficiently exchange data within and across organizations using Google Cloud service APIs.Grant read-only access to external datasets and images that are **not** managed by you.
Ingress: Refers to any access by an API client from outside the service perimeter to resources within a service perimeter.
Egress Refers to any access that involves an API client or resources within the service perimeter and resources outside a service perimeter.

Firewalls protect VM instances

- VPC network functions as a distributed firewall.
- Firewall rules are applied to the network as a whole.
- Connections are allowed or denied at the instance level.
- Firewall rules are stateful.
- Implied deny all ingress and allow all egress.

Firewall rules

- Direction of the rule
- Source or destination of the connection
- Protocol and port of the connection
- Action of the rule
- Priority of the rule
- Rule assignment

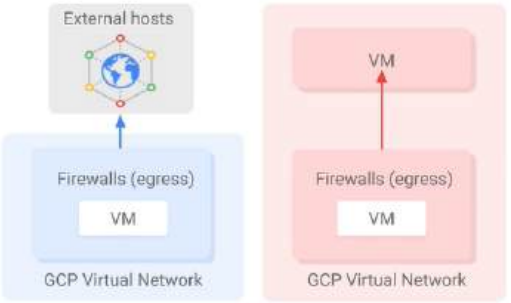
GCP firewall use case: Egress

Action:

- Allow: permit the matching egress connection
- Deny: block the matching egress connection

Conditions:

- Destination CIDR ranges
- Protocols
- Ports



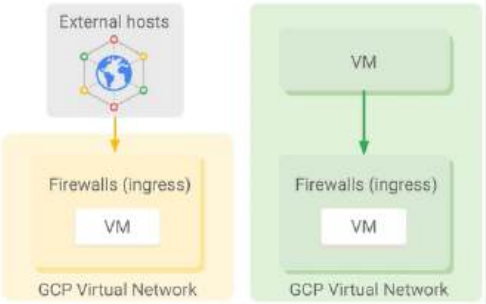
GCP firewall use case: Ingress

Action:

- Allow: permit the matching ingress connection
- Deny: block the matching ingress connection

Conditions:

- Source CIDR ranges
- Protocols
- Ports



[Configuring ingress and egress policies] :

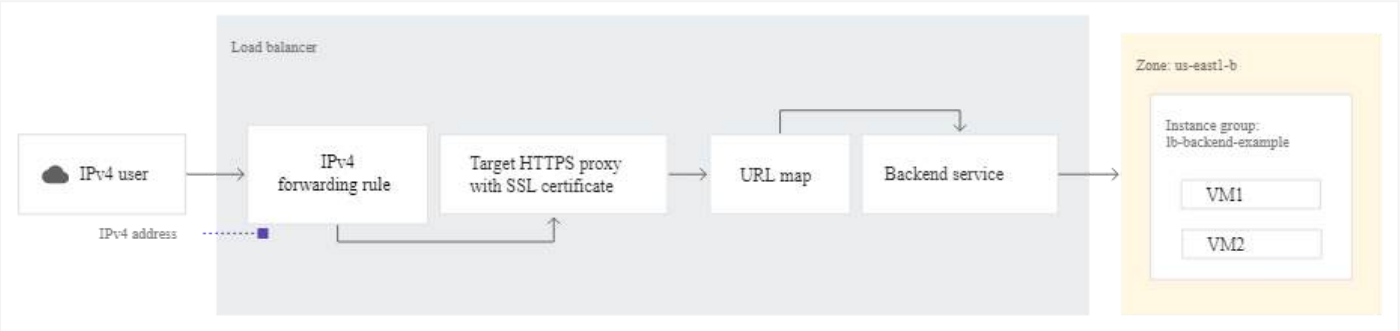
Updating ingress and egress policies for a service perimeter	gcloud beta access-context-manager perimeters update PERIMETER_NAME --set-ingress-policies=INGRESS-FILENAME.yaml gcloud beta access-context-manager perimeters update PERIMETER_NAME

	--set-egress-policies=EGRESS-FILENAME.yaml
Setting ingress and egress policies during perimeter creation	gcloud beta access-context-manager perimeters create PERIMETER_NAME --title=TITLE --ingress-policies=INGRESS-FILENAME.yaml --restricted-services=SERVICE --resources="projects/PROJECT" gcloud beta access-context-manager perimeters create PERIMETER_NAME --title=TITLE --egress-policies=-EGRESS-FILENAME.yaml --restricted-services=SERVICE --resources="projects/PROJECT"

d. Creating a VPN between a Google VPC and an external network using Cloud VPN

e. Creating a load balancer to distribute application network traffic to an application (e.g., Global HTTP(S) load balancer, Global SSL Proxy load balancer, Global TCP Proxy load balancer, regional network load balancer, regional internal load balancer)

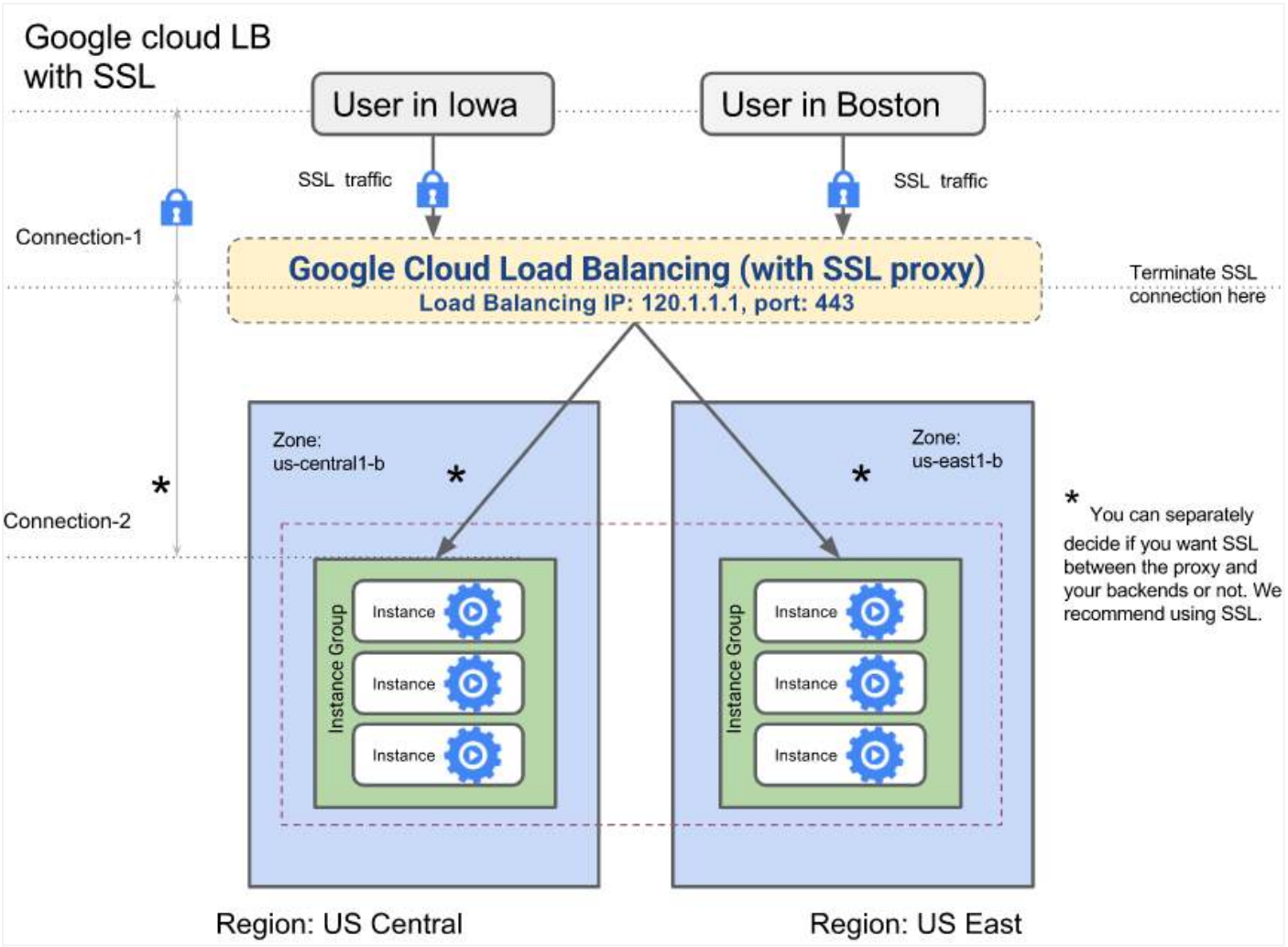
Setting up a global external HTTP(S) load balancer with a Compute Engine backend



Setting up the load balancer : For HTTPS, you need one or more SSL certificate resources to configure the proxy. We recommend using a Google-managed certificate.
Setting up External TCP Proxy Load Balancing : External TCP Proxy Load Balancing is a reverse proxy load balancer that distributes TCP traffic coming from the internet to virtual machine (VM) instances in your Google Cloud VPC network. When using External TCP Proxy Load Balancing, traffic coming over a TCP connection is terminated at the load balancing layer, and then forwarded to the closest available backend using TCP or SSL.Google Cloud External TCP Proxy Load Balancing allows you to use a single IP address for all users around the world. External TCP Proxy Load Balancing automatically routes traffic to the instances that are closest to the user. **configure the following:**

1. A health check for verifying instance health
2. A backend service, which monitors the instances and prevents them from exceeding configured usage
3. The target TCP proxy
4. An external static IPv4 address and forwarding rule that sends user traffic to the proxy
5. An external static IPv6 address and forwarding rule that sends user traffic to the proxy
6. A firewall rule that allows traffic from the load balancer and health checker to reach the instances

Setting up External SSL Proxy Load Balancing: With the Premium Tier, External SSL Proxy Load Balancing can be configured as a global load balancing service. With Standard Tier, the external SSL proxy load balancer handles load balancing regionally.



[Setting up Internal HTTP(S) Load Balancing for Compute Engine VMs]
[Setting up Internal TCP/UDP Load Balancing]

3.6 Deploying a solution using Cloud Marketplace. Tasks include:

a. Browsing the Cloud Marketplace catalog and viewing solution details

Google Cloud Marketplace lets you quickly deploy functional software packages that run on Google Cloud.You can select and deploy software packages from the Cloud Marketplace page of the console. use Cloud Deployment Manager to add resources to a deployment, or remove software deployments that you no longer need. Cloud Deployment Manager uses a declarative approach

Cloud Deployment Manager

- Repeatable deployment process
  - Achieve consistent results.
  - Deploy many resources at once, in Parallel.
  - One reference definition can reference another.
  - Easy to update.
- Declarative language
  - The system figures out the steps to take.
  - Specify the resources needed using YAML.
- Template-driven
  - Creates sets of resources that are typically deployed together.
  - Python and Jinja2 templates control what gets deployed programmatically.
  - Pass variables and get output values in return.

b. Deploying a Cloud Marketplace solution: Integrating your product with Google Cloud Marketplace

3.7 Implementing resources via infrastructure as code. Tasks include:
a. Building infrastructure via Cloud Foundation Toolkit templates and implementing best practices

The Cloud Foundation Toolkit provides a series of reference templates for Deployment Manager and Terraform which reflect Google Cloud best practices. with infrastructure as code (IaC), you can easily update the foundation as your needs change. A blueprint is a package of deployable, reusable configuration and policy that implements and documents a specific opinionated solution. For more details see, blueprints overview. A list of blueprints/modules that are packaged as Terraform modules and can be used for creating resources for Google Cloud.

b. Installing and configuring Config Connector in Google Kubernetes Engine to create, update, delete, and secure resourceskubectl

Anthos : Migrate directly from VMs, Build, deploy, and optimize apps on GKE, Anthos serverless landing zones and VMs anywhere-simply, flexibly, and securely. Anthos is a modern application management platform that provides a consistent development and operations experience for cloud and on-premises environments.

- A hybrid and multi-cloud solution
- Framework rests on Kubernetes and GKE On-Prem
- Provides a rich set of tools for monitoring and maintenance

Anthos Config Management : Automate policy and security at scale for Kubernetes clusters on-premises, on GKE, and on other public clouds. Continuously monitors environments to ensure their desired configuration is in place and no violations of governance controls are present. Config Controller is a hosted service which brings you Config Connector, Config Sync, and Policy Controller for you.
Config Connector is an open source Kubernetes addon that allows you to manage Google Cloud resources through Kubernetes. Config Connector provides a collection of Kubernetes Custom Resource Definitions (CRDs) and controllers.
Install Config Connector :

Creating a new cluster with the Config Connector add-on enabled	gcloud container clusters create CLUSTER_NAME \ --release-channel CHANNEL \ --addons ConfigConnector \ --workload-pool=PROJECT_ID.svc.id.goog \ --logging=SYSTEM \ --monitoring=SYSTEM
To enable Workload Identity for a node pool	gcloud container node-pools update NODE_POOL \ --workload-metadata=GKE_METADATA \ --cluster CLUSTER_NAME
enable the Config Connector add-on on an existing GKE cluster	gcloud container clusters update CLUSTER_NAME \ --update-addons ConfigConnector=ENABLED

- CHANNEL with a GKE release channel, rapid and regular are supported.

Note: When you enable the Config Connector add-on, Config Connector creates a namespace called **configconnector-operator-system**. Do not make any changes or add any resources to this namespace.

Configuring Config Connector : To complete the installation, create a configuration file for the ConfigConnector CustomResource, then apply it using the kubectl apply command. The Config Connector Operator installs Google Cloud Resource CRDs and Config Connector components in your cluster.
Apply the configuration to your cluster: kubectl apply -f configconnector.yaml

Section 4. Ensuring successful operation of a cloud solution

4.1 Managing Compute Engine resources. Tasks include:

a. Managing a single VM instance (e.g., start, stop, edit configuration, or delete an instance)

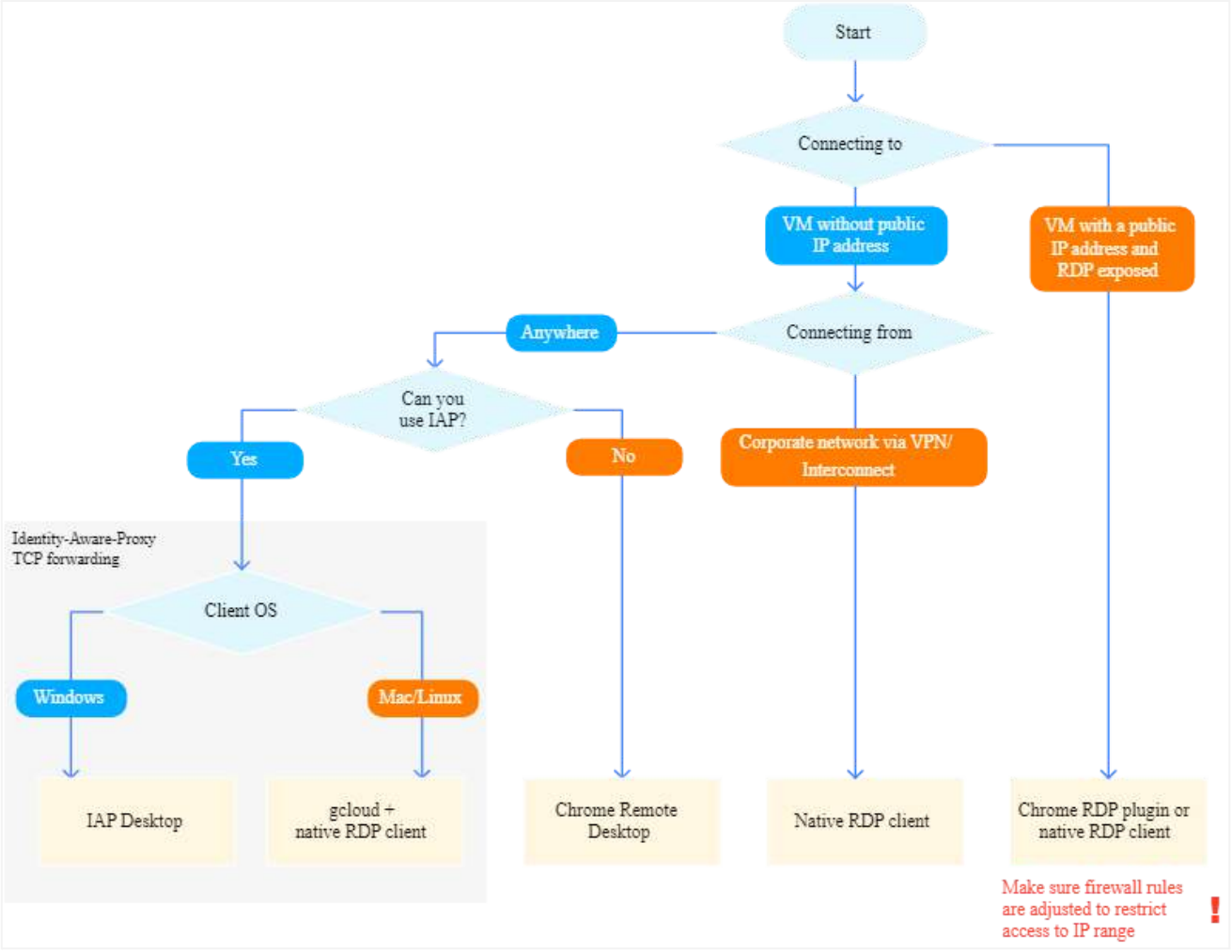
A stopped VM retains its persistent disks, its internal IPs, and its MAC addresses. However, the VM shuts down the guest OS and loses its application state. If you need to retain the guest OS and application state, suspend the VM instead.

To stop a VM	gcloud compute instances stop VM_NAME
To start a VM	gcloud compute instances start VM_NAME
Update instance properties export the existing instance properties	gcloud compute instances export INSTANCE_NAME \ --project PROJECT_ID \ --zone ZONE \ --destination=FILE_PATH
to run a test update of the target instance. Specify the --most-disruptive-allowed-action=NO_EFFECT flag. The response identifies misconfigured properties and indicates whether a RESTART or REFRESH action is required to apply the update.	gcloud compute instances update-from-file INSTANCE_NAME \ --project PROJECT_ID \ --zone ZONE \ --source=FILE_PATH \ --most-disruptive-allowed-action NO_EFFECT
To update the target instance.	gcloud compute instances update-from-file INSTANCE_NAME \ --project PROJECT_ID \ --zone ZONE \ --source=FILE_PATH \ --most-disruptive-allowed-action ALLOWED_ACTION
To delete an instance	gcloud compute instances delete example-instance [example-instance-2 example-instance-3..]

b. Remotely connecting to the instance

If you have set default properties for the Google Cloud CLI	gcloud compute ssh VM_NAME
To connect to a VM using SSH from PuTTY	gcloud compute ssh --project=PROJECT_ID --zone=ZONE VM_NAME

Compute Engine supports multiple ways to connect to your Windows instances.



To connect with Microsoft Windows Remote Desktop, do the following:

1. Create a Windows account and password if you do not have one yet.
2. To connect over the internet, use the *external* IP address. To connect by using Cloud VPN or Cloud Interconnect, use the *internal* IP address.
3. Identify the external and internal IP addresses of your Windows instance by completing one of the following steps:

IAP TCP forwarding allows you to establish an encrypted tunnel over which you can forward SSH, RDP, and other traffic to VM instances. port **22** for SSH and port **3389** for RDP.

c. Attaching a GPU to a new instance and installing necessary dependencies

Add or remove a GPU from an existing VM is as follows:

1. Check that your VM has a boot disk size of at least 40 GB.
2. Prepare your VM for the modification.
3. Stop the VM then Add or remove the GPU.
4. If you are adding a GPU, you need to complete the following steps:
    - Modify the host maintenance setting for the VM. VMs with GPUs cannot live migrate because they are assigned to specific hardware devices. GPU restrictions.
    - Change the machine type. GPUs are only supported on select machine types.
    - Install a GPU driver on your VM, so that your system can use the device.

`regions describe` ensure you have sufficient GPU quota	gcloud compute regions describe REGION
To prevent the network interface from persisting, run the following command on your VM	rm /etc/udev/rules.d/70-persistent-net.rules

d. Viewing current running VM inventory (instance IDs, details)

Get a list of VMs : **`gcloud compute instances list command`**

e. Working with snapshots (e.g., create a snapshot from a VM, view snapshots, delete a snapshot)

Create and manage disk snapshots **:** You can create snapshots from disks even while they are attached to running instances.

To create a snapshot of a persistent disk in the default storage location	gcloud compute snapshots create SNAPSHOT_NAME \
	  --source-disk SOURCE_DISK \
	  --source-disk-zone SOURCE_DISK_ZONE

Alternatively, to create a snapshot in a custom storage location, use the --storage-location flag to indicate where to store your snapshot	gcloud compute snapshots create SNAPSHOT_NAME \
	  --source-disk SOURCE_DISK \
	  --source-disk-zone SOURCE_DISK_ZONE \
	  --storage-location STORAGE_LOCATION

To create a snapshot of a regional persistent disk in the default storage location	gcloud compute snapshots create SNAPSHOT_NAME \
	  --source-disk SOURCE_DISK \
	  --source-disk-region=SOURCE_DISK_REGION

To delete a snapshot	gcloud compute snapshots delete SNAPSHOT_NAME
To delete the snapshots, use a combination command with xargs	gcloud compute snapshots list --filter="creationTimestamp>'2021-01-01'" --uri | xargs gcloud compute snapshots delete
To see a list of snapshots available to you in a particular project	gcloud compute snapshots list --project PROJECT_ID
To list information about a particular snapshot, such as the creation time, size, and source disk	gcloud compute snapshots describe SNAPSHOT_NAME

xargs - build and execute command lines from standard input
xargs [*options*] [*command* [*initial-arguments*]]

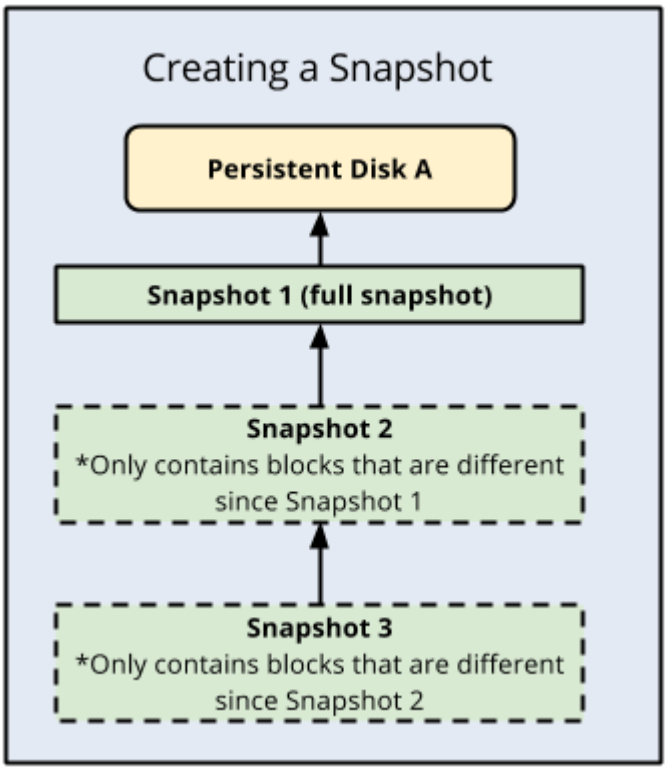
Share snapshot data across projects: To move data from a disk in one project to a disk in a different project, use the following process

Create a disk snapshot in the destination project	gcloud compute snapshots create SNAPSHOT_NAME \
	  --source-disk
	https://www.googleapis.com/compute/v1/projects/SOURCE_PROJECT_ID/zones/ZONE/disks/SOURCE_DISK_NAME \
	  --project DESTINATION_PROJECT_ID

In the destination project, create a zonal or regional disk that's based on the snapshot	gcloud compute disks create DISK_NAME \
	  --source-snapshot SNAPSHOT_NAME \
	  --project DESTINATION_PROJECT_ID

Snapshots are incremental and automatically compressed, so you can create regular snapshots on a persistent disk faster and at a lower cost than if you regularly created a full image of the disk.
You can only create them for persistent disks. They are stored in a Cloud Storage bucket managed by the snapshot service and are automatically compressed.

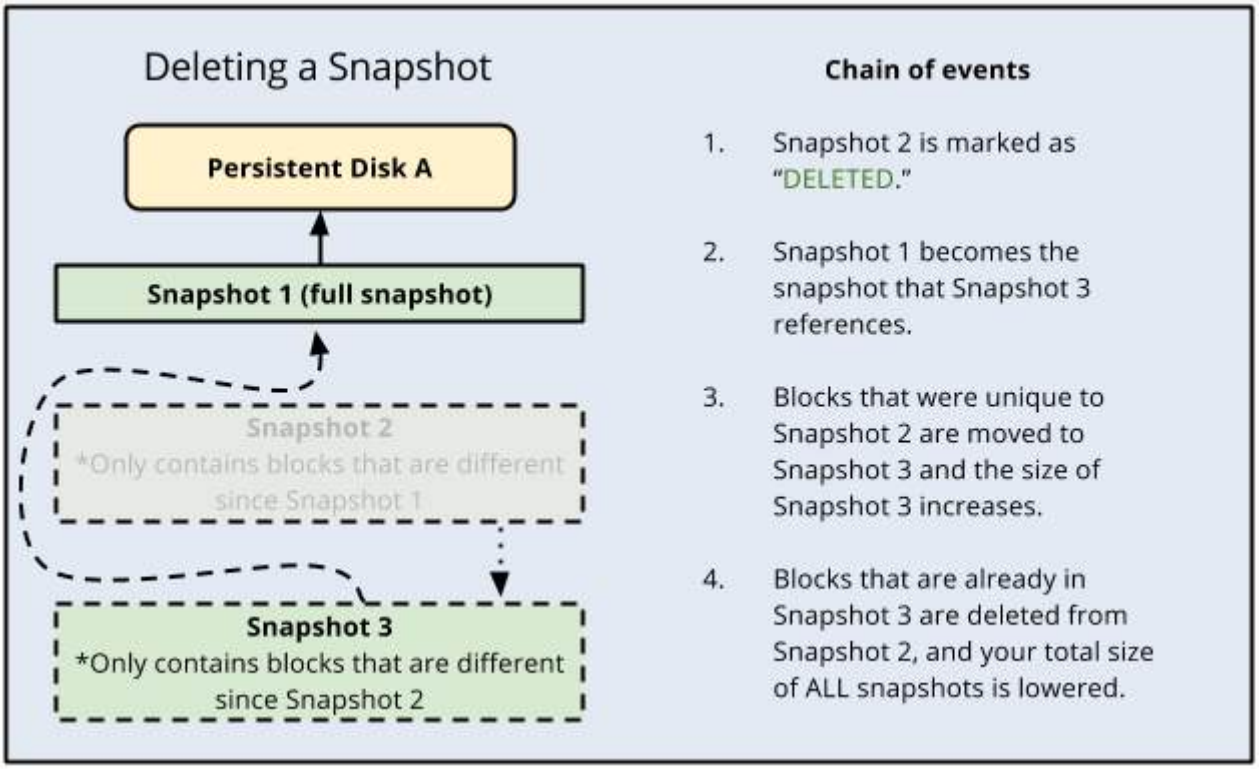
Compute Engine uses incremental snapshots so that each snapshot contains only the data that has changed since the previous snapshot. For unchanged data, snapshots reference the data in previous snapshots. Storage costs for persistent disk snapshots charge only for the total size of the snapshot.



When you delete a snapshot, Compute Engine immediately marks the snapshot as DELETED in the system. If the snapshot has no dependent snapshots, it is deleted outright. However, if the snapshot does have dependent snapshots:

1. Any data that is required for restoring other snapshots is moved into the next snapshot, increasing its size.
2. Any data that is not required for restoring other snapshots is deleted. This lowers the total size of all your snapshots.
3. The next snapshot no longer references the snapshot marked for deletion, and instead references the snapshot before it.

If your disk has a snapshot schedule, you must detach the snapshot schedule from the disk before you can delete the schedule. Removing the snapshot schedule from the disk prevents further snapshot activity from occurring. You cannot delete a schedule that is attached to a disk. You have the option to manually delete snapshots at any time.



sometimes deleted files are not discarded. If this happens, you might see that the size of your snapshot is larger than the used space on the disk reported by the file system. To avoid this, it is a best practice to enable the discard option or run fstrim on your disk.

If you do not specify a storage location for a snapshot, Google Cloud uses the default location, which stores your snapshot in a Cloud Storage multi-regional location closest to the region of the source disk. If you need to choose regional storage, or if you need to specify a different multi-regional location, store your snapshot in a custom location.

Note: You cannot change the storage location of an existing snapshot. If you need to move a snapshot from one region or multi-regional location to another, you must create a new snapshot and specify its location, and delete the previous snapshot. When you create a snapshot, you can only specify one multi-regional location or one regional location. If you need to store a snapshot in more than one location, you must create a snapshot in each location.

Operation status	Snapshot resource status
PENDING	No snapshot resource exists yet.
RUNNING	CREATING or UPLOADING CREATING: Snapshot creation is not yet complete. UPLOADING: Snapshot has been created but is not yet saved to Cloud Storage.
DONE	FAILED or READY.

Selecting your snapshot storage location is vital to minimizing network costs. If you store your snapshot in the same region as your source disk, there is no network charge when you access that snapshot from the same region. If you access the snapshot from a different region, there is a network cost.
If your source disk's geographic storage location is the same as its multi-region, there is no network charge.

f. Working with images (e.g., create an image from a VM or a snapshot, view images, delete an image)

Create, delete, and deprecate custom images : Before creating an image from a disk, disable auto-delete to ensure that the disk is not automatically deleted when you delete the VM. The **VM instance details** page Click **Edit** in the **Boot disk** section, for the **Deletion rule**, ensure that the **Keep disk** option is selected.
The `--force` flag is an optional flag that lets you create the image from a running instance. By default, you cannot create images from running instances.

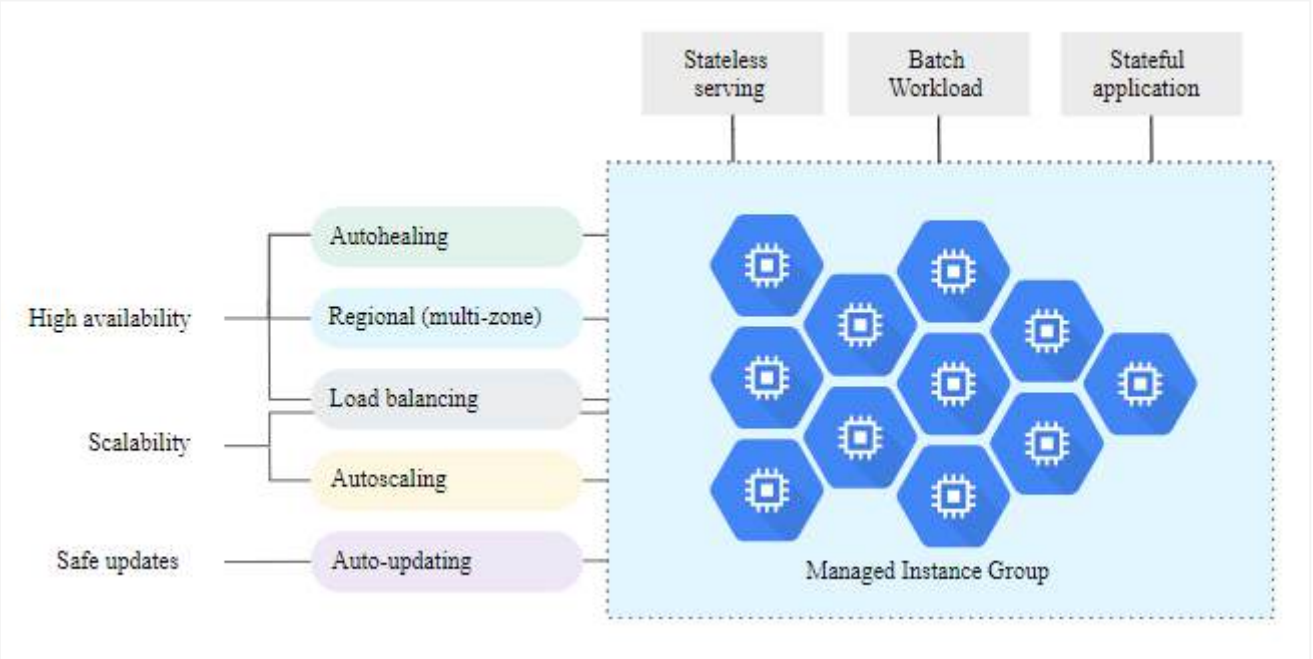
to disable auto-delete for the disk.	gcloud compute instances set-disk-auto-delete VM_NAME \ --no-auto-delete \ --disk=SOURCE_DISK
Create an image from a source disk	gcloud compute images create IMAGE_NAME \ --source-disk=SOURCE_DISK \ --source-disk-zone=ZONE \ [--family=IMAGE_FAMILY] \ [--storage-location=LOCATION] \ [--force]
Create an image from a snapshot	gcloud compute images create IMAGE_NAME \ --source-snapshot=SOURCE_SNAPSHOT \ [--storage-location=LOCATION]
to set the deprecation status of an image	gcloud compute images deprecate IMAGE_NAME \ --state STATE \ --replacement REPLACEMENT
to delete an image	gcloud compute images delete IMAGE_NAME

- SOURCE_DISK: the disk from which you want to create the image
- ZONE: the zone where the disk is located
- IMAGE_FAMILY: Optional: a flag that specifies which image family this image belongs to

- LOCATION: Optional: a flag that lets you designate the region or multi-region where your image is stored.
- SOURCE_SNAPSHOT: the snapshot from which you want to create the image
- STATE: the deprecation state : `ACTIVE DEPRECATED OBSOLETE DELETED`

g. Working with instance groups (e.g., set autoscaling parameters, assign instance template, create an instance template, remove instance group)

An instance group is a collection of virtual machine (VM) instances that you can manage as a single entity.



Implementing an instance group :

1. Create instance template e.g. identify machine type and boot disk
2. Configure your instance group e.g. number of instances and autoscaling settings

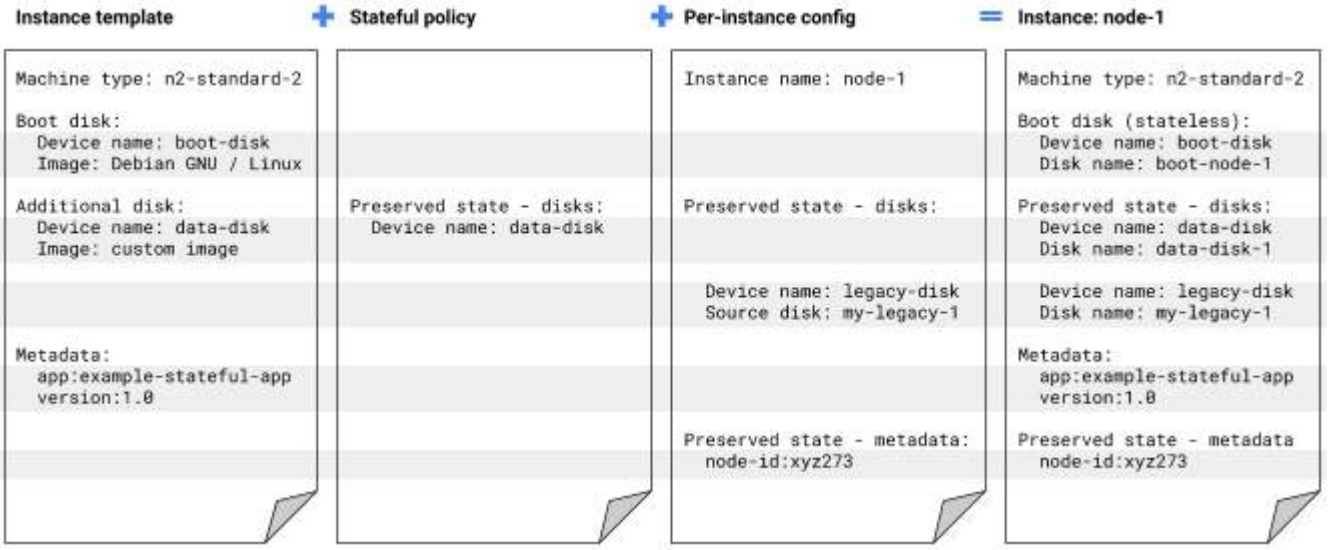
Managed instance groups: A managed instance group (MIG) is a group of virtual machine (VM) instances that you treat as a single entity. Each VM in a MIG is based on an instance template.

- Deploy identical instances based on instance template
- Instance group can be resized
- Manager ensures all instances are RUNNING
- Typically used with autoscaler
- Can be single zone or regional
- MIGs preserve instance names on recreation and on autohealing by default.
- MIG recreates the stateless items and preserves the stateful items
- For stateless MIGs, **Autoscaling** is enabled by default. With autoscaling, your group automatically adds or removes instances based on its utilization.

Limitations:

- By default, you can create up to 1,000 VMs in a zonal MIG
- You cannot create a MIG with multiple subnets. Once created, you cannot change the network or subnetwork in a MIG.
- multiple zones—a *regional MIG,* By default, you can create up to 2,000 VMs.
- If you use an HTTP(S) load balancing scheme with a regional MIG, you must choose the maxRatePerInstance or maxUtilization balancing mode.
- For stateful regional MIGs, you must disable proactive redistribution (set the redistribution type to NONE) to prevent deletion of stateful instances by automatic cross-zone redistribution

A stateful managed instance group (MIG) takes its instance configuration from a combination of the instance template, stateful policy, and per-instance configurations that you set. After you apply the instance template, stateful policy, and/or per-instance configurations to your group, the MIG uses that configuration when creating, recreating, autohealing, or auto-updating its VM instances.



Stateful MIGs are intended for applications with stateful data or configuration, such as:

- **Databases**. For example: Cassandra, ElasticSearch, mongoDB, and ZooKeeper. Before deciding on stateful MIGs, consider using fully managed services, for example, MySQL and PostgreSQL are available in Cloud SQL, to focus on your applications and not have to deal with VMs.
- **Data processing** applications. For example: Kafka and Flink. Before deciding on stateful MIGs, consider using fully managed services, for example, Dataflow or Dataproc, to focus on your data processing tasks and not have to deal with VMs.
- **Other stateful** applications. For example: TeamCity, Jenkins, Bamboo, DNS servers with stateful IP address, and custom stateful workloads.
- **Legacy monolith** applications. These applications store application state on a boot disk or additional persistent disks, or they rely on stateful configuration, such as specific VM instance names, IP addresses, or metadata key values.
- **Batch workloads with checkpointing**. With this configuration, you can preserve checkpointed results of long-running computation in anticipation of workload or VM failure or instance preemption. Stateful MIGs can recreate a failed machine, while preserving its data disk, so that your computation can continue from the last checkpoint.

Create a MIG in a single zone	gcloud compute instance-groups managed create INSTANCE_GROUP_NAME \ --size SIZE \ --template INSTANCE_TEMPLATE \ --zone ZONE
For example, the following command creates an instance group named example-group, with base VM name test. The group contains three instances	gcloud compute instance-groups managed create example-group \ --base-instance-name test \ --size 3 \ --template an-instance-template
Create a MIG with VMs in multiple zones in a region	gcloud compute instance-templates create example-template gcloud compute instance-groups managed create example-rmig \ --template example-template \ --size 30 \

	--zones us-east1-b,us-east1-c (--region us-east1) creates a regional MIG
Scaling based on CPU utilization	gcloud compute instance-groups managed set-autoscaling example-managed-instance-group \ --max-num-replicas 20 \ --target-cpu-utilization 0.60 \ --cool-down-period 90

Note: Use separate health checks for load balancing and for autohealing. Health checks for load balancing can and should be more aggressive because these health checks determine whether a VM receives user traffic. You want to catch non-responsive VMs quickly, so you can redirect traffic if necessary. In contrast, health checking for autohealing causes Compute Engine to proactively replace failing VMs, so this health check should be more conservative than a load balancing health check.
You can set a maximum of one autohealing policy per MIG. You can apply a single health check to a maximum of 50 MIGs. If you have more than 50 groups, create multiple health checks.

Create instance templates	gcloud compute instance-templates create example-template-custom \ --machine-type=e2-standard-4 \ --image-family=debian-10 \ --image-project=debian-cloud \ --boot-disk-size=250GB
Create a VM instance from an instance template	gcloud compute instances create VM_NAME --source-instance-template INSTANCE_TEMPLATE_NAME
Delete instances from a managed instance group	gcloud compute instance-groups managed delete-instances INSTANCE_GROUP_NAME \ --instances example-i3n2,example-z2x9 \ --zone ZONE
Delete a managed instance group	gcloud compute instance-groups managed delete INSTANCE_GROUP_NAME \ --zone ZONE

Note: You cannot delete an instance group if it is being used by a load balancer's backend service. Remove the backend service before deleting the instance group.
Autoscaling groups of instances : scaling based on target utilization metrics, see the following pages:

- Scaling based on CPU utilization
- Scaling based on the serving capacity of an external HTTP(S) load balancer
- Scaling based on Cloud Monitoring metrics

h. Working with management interfaces (e.g., Cloud Console, Cloud Shell, Cloud SDK)

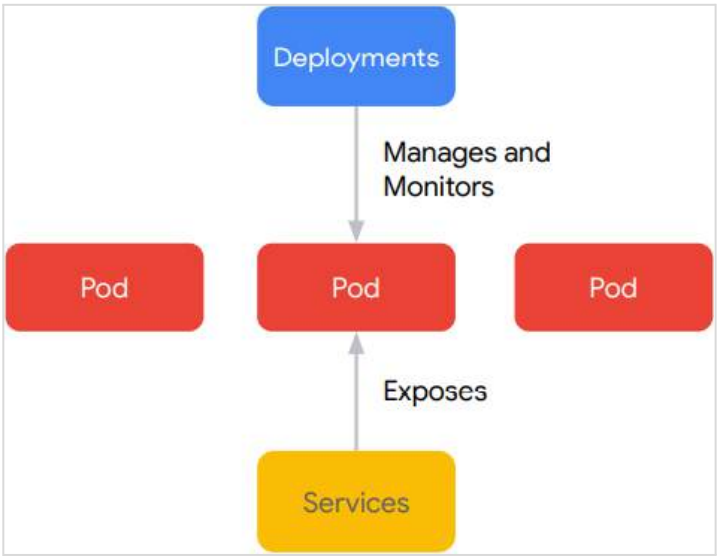
Cloud SDK : Libraries and tools for interacting with Google Cloud products and services.

4.2 Managing Google Kubernetes Engine resources. Tasks include:

a. Viewing current running cluster inventory (nodes, pods, services)

Kubernetes objects
A cluster typically has one or more nodes, which are the worker machines that run your containerized applications and other workloads. Each node is managed from the control plane, which receives updates on

each node's self-reported status.A node runs the services necessary to support the containers that make up your cluster's workloads.



A Pod Is the smallest deployable object in Kubernetes. It is a single instance of a running process that contains one or more docker containers. Pods provide networking and storage to containers, and contain dependencies the container needs to run and communicate.

A Deployment manages a set of multiple identical pods. It uses a replica set to define the number of pods. A deployment monitors pods in a replica set and replaces unhealthy instances to ensure your application remains available. A deployment uses a pod template, which provides a spec of what each deployed pod should look like. When you update the pod template in a deployment, it starts a rolling upgrade of the pods in the deployment.

A Service is a group of pod endpoints that you can configure access for. You use selectors to define which pods are included in a service. A service gives you a stable IP that belongs to the service. Pods have internal IP addresses but they can change as pods get restarted and replaced. A service can be configured to implement load balancing.

The `kubectl` command-line tool supports several different ways to create and manage Kubernetes objects. Read the Kubectl book for details of managing objects by Kubectl.
Imperative : Overwrite existing state | Operate on single object| Specifies how
Imperative commands such as run, create, replace, delete act on a live object or single config file and overwrite any state changes that have occurred on an existing object.

Declarative: - apply : Works on a directory of config files | Specifies what
Declarative commands use a config stored in a directory to deploy and apply changes to your app objects.

Mode - GKE has to modes to choose from: autopilot mode and standard mode. Autopilot is fully-provisioned and managed. You are charged according to the resources pods use as you deploy and replicate them based on the pod spec. Standard mode provides you flexibility to define and manage the cluster structure yourself.

b. Browsing Docker images and viewing their details in the Artifact Registry

Artifact Registry provides a single location for managing private packages and Docker container images. Docker requires privileged access to interact with registries.

c. Working with node pools (e.g., add, edit, or remove a node pool)

Manually upgrading a cluster or node pool : By default, a cluster's nodes have auto-upgrade enabled, and it is recommended that you do not disable it.While a node is being upgraded, GKE stops scheduling new Pods onto it, and attempts to schedule its running Pods onto other nodes. This is similar to other events that re-create the node, such as enabling or disabling a feature on the node pool.
Alpha clusters cannot be upgraded.

Manually upgrade a node pool	gcloud container clusters upgrade CLUSTER_NAME \ --node-pool=NODE_POOL_NAME \

	--cluster-version VERSION
Checking node pool upgrade status	gcloud container operations list
Checking node pool upgrade settings	gcloud container node-pools describe NODE_POOL_NAME \ --cluster=CLUSTER_NAME

VERSION: the Kubernetes version to which the nodes are upgraded. For example, --cluster-version=1.7.2 or cluster-version=latest

Add and manage node pools :

Add a node pool / To create a node pool	gcloud container node-pools create POOL_NAME \ --cluster CLUSTER_NAME \ --service-account SERVICE_ACCOUNT
To check the status of all node pools	gcloud container node-pools list --cluster CLUSTER_NAME
To resize a cluster's node pools	gcloud container clusters resize CLUSTER_NAME \ --node-pool POOL_NAME \ --num-nodes NUM_NODES
To delete a node pool	gcloud container node-pools delete POOL_NAME \ --cluster CLUSTER_NAME

- POOL_NAME: the name of the new node pool.
- CLUSTER_NAME: the name of your existing cluster.
- SERVICE_ACCOUNT: the name of the IAM service account for your nodes to use. If omitted, the node pool uses the Compute Engine default service account.
- NUM_NODES: the number of nodes in the pool in a zonal cluster. If you use multi-zonal or regional clusters, NUM_NODES is the number of nodes for each zone the node pool is in.

d. Working with pods (e.g., add, edit, or remove pods)

Kubernetes best practices: Resource requests and limits

e. Working with services (e.g., add, edit, or remove a service)

The idea of a Service is to group a set of Pod endpoints into a single resource.
Why use a Kubernetes Service? In a Kubernetes cluster, each Pod has an internal IP address. But the Pods in a Deployment come and go, and their IP addresses change. So it doesn't make sense to use Pod IP addresses directly. With a Service, you get a stable IP address that lasts for the life of the Service, even as the IP addresses of the member Pods change.
A Service also provides load balancing. Clients call a single, stable IP address, and their requests are balanced across the Pods that are members of the Service.
five types of Services:

- **ClusterIP (default):** creates a stable IP address that is accessible from nodes in the cluster.
- **NodePort:** Clients send requests to the IP address of a node on one or more nodePort values that are specified by the Service.Service is accessible by using the IP address of any node along with the nodePort value.
- **LoadBalancer:** Clients send requests to the IP address of a network load balancer.
- **ExternalName:** Internal clients use the DNS name of a Service as an alias for an external DNS name.
- **Headless:** You can use a headless service when you want a Pod grouping, but don't need a stable IP address.

When you create a Service, Kubernetes creates an Endpoints object that has the same name as your Service. Kubernetes uses the Endpoints object to keep track of which Pods are members of the Service.

Exposing applications using services :

1. Creating a Service of type ClusterIP
2. Creating a Service of type NodePort
3. Creating a Service of type LoadBalancer
4. Creating a Service of type ExternalName
5. Using kubectl expose to create a Service
6. Steps to remove resources : kubectl delete services my-cip-service my-np-service my-lb-service

f. Working with stateful applications (e.g. persistent volumes, stateful sets)

Stateful applications save data to persistent disk storage for use by the server, by clients, and by other applications. Kubernetes uses the StatefulSet controller to deploy stateful applications as StatefulSet objects. Pods in StatefulSets are not interchangeable: each Pod has a unique identifier that is maintained no matter where it is scheduled. Stateful applications are different from stateless applications, in which client data is not saved to the server between sessions.StatefulSet objects include a volumeClaimTemplates array, which automatically generates the PersistentVolumeClaim objects. Each StatefulSet replica gets its own PersistentVolumeClaim object.
Creating a StatefulSet : To create a StatefulSet resource, use the kubectl apply command.

To manually scale a StatefulSet	kubectl scale statefulset STATEFULSET_NAME --replicas NUMBER_OF_REPLICAS
To delete a StatefulSet	kubectl delete statefulset STATEFULSET_NAME

g. Managing Horizontal and Vertical autoscaling configurations

The Horizontal Pod Autoscaler changes the shape of your Kubernetes workload by automatically increasing or decreasing the number of Pods in response to the workload's CPU or memory consumption, or in response to custom metrics reported from within Kubernetes or external metrics from sources outside of your cluster.
Vertical Pod autoscaling provides recommendations for resource usage over time. For sudden increases in resource usage, use the Horizontal Pod Autoscaler.Do not use the Horizontal Pod Autoscaler together with the Vertical Pod Autoscaler on CPU or memory. You can use the Horizontal Pod Autoscaler with the Vertical Pod Autoscaler for other metrics.

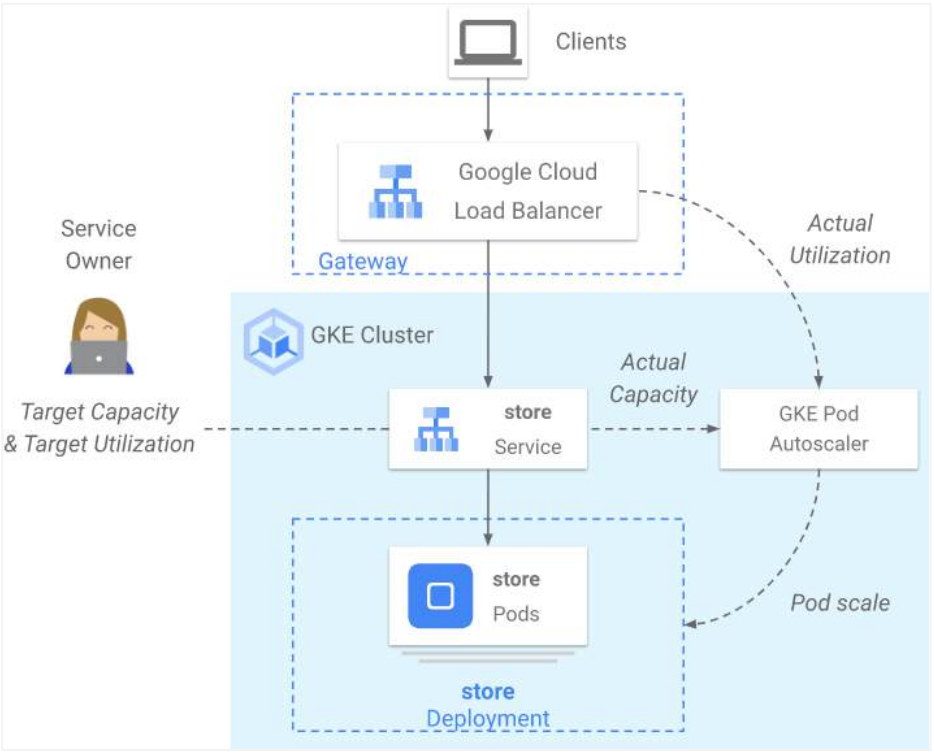
Horizontal Pod autoscaling cannot be used for workloads that cannot be scaled, such as DaemonSets.The Horizontal Pod Autoscaler can automatically scale the number of Pods in your workload based on one or more metrics of the following types: 1] **Actual resource usage** 2] **Custom metrics** 3] **External metrics**
Note: To use resource utilization percentage targets with horizontal Pod autoscaling, you must configure requests for that resource for each container running in each Pod in the workload. Otherwise, the Horizontal Pod Autoscaler cannot perform the calculations it needs to, and takes no action related to that metric
The controller uses the average or raw value for a reported metric to produce a ratio, and uses that ration to autoscale the workload.Each configured Horizontal Pod Autoscaler operates using a control loop. If one or more of the metrics are unavailable for some reason, the Horizontal Pod Autoscaler still scales **up** based on the largest size calculated, but does not scale **down**.
Each Horizontal Pod Autoscaler exists in the cluster as a `HorizontalPodAutoscaler object`. You can use commands like `kubectl get hpa` or `kubectl describe hpa HPA_NAME` to interact with these objects.

Configuring horizontal Pod autoscaling :

API versions for HorizontalPodAutoscaler objects : Google Cloud console, HorizontalPodAutoscaler objects are created using the `autoscaling/v2beta2` API.
`apiVersion: autoscaling/v1` is the default, and allows you to autoscale based only on CPU utilization.
`apiVersion: autoscaling/v2beta2` allows you to autoscale based on multiple metrics, including custom or external metrics.
Autoscaling based on resources utilization :
Console > Workloads page > Deployment Name > click Action > Autoscale : Specify the following values : Minimum number of replicas: 1 Maximum number of replicas: 10 Autoscaling metric: CPU Target: 50 Unit: % > click done > click Autoscale.

	kubectl autoscale deployment nginx --cpu-percent=50 --min=1 --max=10
delete the HPA	kubectl delete hpa nginx
To manually scale the Deployment back to 3 Pods	kubectl scale deployment nginx --replicas=3

Autoscaling based on load balancer traffic is only available for Gateway workloads.Traffic-based autoscaling is enabled by the Gateway controller and its global traffic management capabilities.
The Service capacity is a critical element when using traffic-based autoscaling because it determines the amount of per-Pod traffic that triggers an autoscaling event.
To deploy traffic-based autoscaling, Install the Gateway API CRDs in your cluster. Check using kubectl get gatewayclass
Note: The ratios between Gateway, Horizontal Pod Autoscaler, Deployment, and Service must be 1:1:1:1.
Requests (queries) per second (RPS, QPS) At 30 RPS, the Deployment is scaled to 5 replicas so that each replica ideally receives 6 RPS of traffic, which would be 60% utilization per Pod.



Autoscaling based on multiple metrics : a custom metric named `packets_per_second`.
`unable to fetch pod metrics for pod` However, if you continue to see the warnings and you notice that Pods are not scaling for your workload, please ensure you have specified resource requests for each container in your workload.

Configure vertical Pod autoscaling to provide recommended values for CPU and memory requests and limits that you can use to manually update your Pods, or you can configure vertical Pod autoscaling to automatically update the values.Vertical Pod autoscaling is enabled by default in Autopilot clusters.

Notifies the cluster autoscaler to adjust cluster capacity. Vertical Pod autoscaling supports a maximum of 500 VerticalPodAutoscaler objects per cluster.Vertical Pod autoscaling works best with long-running homogenous workloads.

Enable vertical Pod autoscaling in your cluster : The Vertical Pod Autoscaler automatically analyzes your containers and provides suggested resource requests.To view suggested resource requests in the console, you must have an existing workload deployed that is at least 24 hours old.
Analyze resource requests :
Console > Workloads page > click the name of the workload you want to scale > Click list Actions > Scale > Edit resource requests

Autoscaling Cluster : creates an autoscaling multi-zonal cluster with six nodes across three zones initially, with a minimum of one node per zone and a maximum of four nodes per zone	gcloud container clusters create example-cluster \
	  --num-nodes=2 \
	  --zone=us-central1-a \
	
	--node-locations=us-central1-a,us-central1-b,us-central1-f \
	  --enable-autoscaling --min-nodes=1 --max-nodes=4

Enable vertical Pod autoscaling for your cluster	gcloud container clusters update CLUSTER_NAME --enable-vertical-pod-autoscaling

Set Pod resource requests manually
Set Pod resource requests automatically : Vertical Pod autoscaling uses the `VerticalPodAutoscaler` object to automatically set resource requests on Pods when the `updateMode` is `"Auto"`. The `updateMode` value of `Auto` means that the Vertical Pod Autoscaler controller can delete a Pod, adjust the CPU and memory requests, and then start a new Pod.
The Vertical Pod Autoscaler uses the `lowerBound` and `upperBound` attributes to decide whether to delete a Pod and replace it with a new Pod. If a Pod has requests less than the lower bound or greater than the upper bound, the Vertical Pod Autoscaler deletes the Pod and replaces it with a Pod that meets the target attribute.
Opt out specific containers : In this exercise you create a `VerticalPodAutoscaler` object that has a specific container opted out. Then you create a Deployment that has one Pod with two containers. When the Pod is created, the Vertical Pod Autoscaler creates and applies a recommendation only for single container, ignoring the one that was opted out.

Disable vertical Pod autoscaling in your cluster	gcloud container clusters update CLUSTER_NAME --no-enable-vertical-pod-autoscaling

Configuring multidimensional Pod autoscaling : With multidimensional Pod autoscaling, you can use horizontal scaling based on CPU and vertical scaling based on memory at the same time. A `MultidimPodAutoscaler` object modifies memory requests and adds replicas so that the average CPU utilization of each replica matches your target utilization. `MultidimPodAutoscaler` manifest automatically adjusts the number of replicas and memory requests based on the values you specify.

View all MultidimPodAutoscaler objects	kubectl get mpa

h. Working with management interfaces (e.g., Cloud Console, Cloud Shell, Cloud SDK, kubectl)

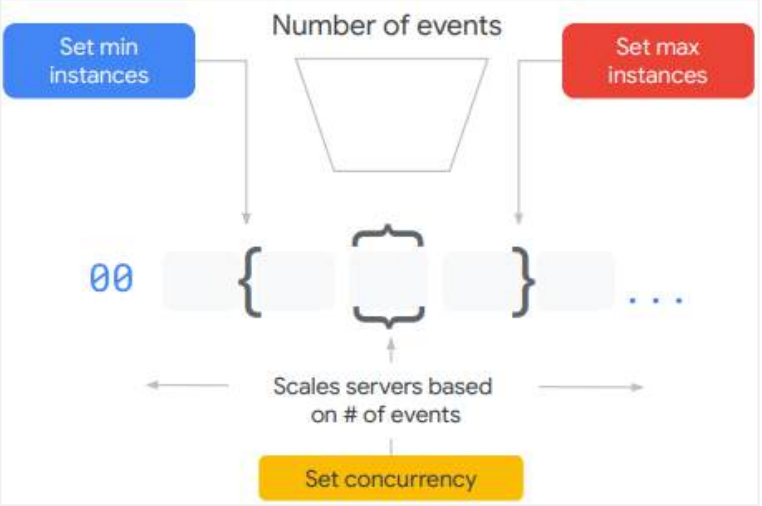
4.3 Managing Cloud Run resources. Tasks include:
a. Adjusting application traffic-splitting parameters

Cloud Run allows you to specify which revisions should receive traffic and to specify traffic percentages that are received by a revision. This feature allows you to rollback to a previous revision, gradually roll out a revision, and split traffic between multiple revisions.
Split traffic between multiple revisions : Replace LIST with a comma delimited list of revisions and percentages: REVISION1=PERCENTAGE1,REVISION2=PERCENTAGE2,REVISIONn=PERCENTAGEx
for example, hello2-00005-red=25,hello2-00001-bod=25,hello2-00002-nan=50.

To split traffic between two or more revisions	gcloud run services update-traffic SERVICE --to-revisions LIST
To send all traffic to the most recently deployed revision	gcloud run services update-traffic SERVICE --to-latest

b. Setting scaling parameters for autoscaling instances

Cloud Run autoscaling :
Cloud Run automatically scales the number of container instances required for each deployed revision. When no traffic is received, the deployment automatically scales to zero.



Other ways you can affect Cloud Run autoscaling:

1. CPU utilization, with a default 60% utilization.
2. Concurrency settings, with a default of 80 concurrent requests. You can increase it to 1000. You can also lower it if you need to.
3. Max number of instances limits total number of instances. It can help you control costs and limit connections to a backing service. Defaults to 1000. Quota increase required if you want more.
4. Min number of instances keeps a certain number of instances up. You will incur cost even when no instances are handling requests.

Instances that are started might remain idle for up to 15 minutes to reduce latency associated with cold starts. You don't get charged for these idle instances. You set a min and max on the container tab in the advanced settings dialog.

Cloud Run provides a *maximum concurrent requests per instance* setting that specifies the maximum number of requests that can be processed simultaneously by a given container instance.

Setting maximum concurrent requests per instance (services) :

To set maximum concurrent requests per instance	gcloud run services update SERVICE --concurrency CONCURRENCY
To revert to the default `(80)`	gcloud run services update SERVICE --concurrency default
	gcloud run services update SERVICE --concurrency 1

Setting and updating maximum instances

	gcloud run services update SERVICE --max-instances MAX-VALUE

instances during deployment	gcloud run deploy --image IMAGE_URL --max-instances MAX-VALUE

Setting and updating minimum instances : Minimum instances will be kept running at idle, ready to serve requests. By default, container instances have `min-instances` turned off, with a setting of `0`.

SERVICE is the name of service	gcloud run services update SERVICE --min-instances MIN-VALUE
IMAGE_URL reference to container image	gcloud run deploy --image IMAGE_URL --min-instances MIN-VALUE

c. Determining whether to run Cloud Run (fully managed) or Cloud Run for Anthos

Cloud Run : The Cloud Run platform allows you to deploy stateless containers without having to worry about the underlying infrastructure. Your workloads are automatically scaled out or in to zero depending on the traffic to your app. Cloud Run has pay-per-use pricing, rounded up to the nearest 100 millisecond.

Cloud Run for Anthos : Cloud Run for Anthos abstracts away the complexity of Kubernetes, making it easy to build and deploy apps across hybrid and multi-cloud environments. Cloud Run for Anthos is Google's managed and fully supported Knative offering, an open source project that enables serverless workloads on Kubernetes.

	Cloud Run	Cloud Run for Anthos
Pricing	Pay-per-use	Included as part of Anthos.
Machine types	Limited CPU and Memory.	Standard or custom machine types on Anthos, including GPUs.
Autoscaling	Up to 1,000 container instances by default, can be increased via a Quota increase.	Limited by the capacity of your Anthos cluster.
Identity and policy	Manage the identities that are allowed to invoke each service (or allow unauthenticated invocations).	Publish services to the internet or make them available to cluster or VPC network only.
Networking	Access to VPC / Compute Engine network via Serverless VPC Access. Services are not part of the Istio service mesh.	Access to VPC / Compute Engine network. Services participate in the Anthos Service Mesh.
URLs	Automatic service URLs and SSL certificates.	Custom domains only with manual SSL certificates.
Container isolation	Strict container isolation based on gVisor sandbox.	Default Kubernetes container isolation.
Execution environments	Fully managed on Google infrastructure.	Anthos clusters.

Setting up Cloud Run for Anthos : Deploy an application to Cloud Run for Anthos

4.4 Managing storage and database solutions. Tasks include:

a. Managing and securing objects in and between Cloud Storage buckets

Object Holds : an object has a hold placed on it, the object cannot be deleted or replaced. types of holds:

- **Event-based** holds
- **Temporary** holds

When an object is stored in a bucket without a retention policy, both hold types behave exactly the same. When an object is stored in a bucket with a retention policy, the hold types have different effects on the object when the hold is released:

- An event-based hold resets the object's time in the bucket for the purposes of the retention period.

- A temporary hold does not affect the object's time in the bucket for the purposes of the retention period.

Set Object Versioning on a bucket : Once Object Versioning is enabled, each time a live object version is replaced or deleted, that version becomes a *noncurrent* version.
Note: Object Versioning cannot be enabled on a bucket that currently has a retention policy.

	gsutil versioning set STATE gs://BUCKET_NAME
check whether Object Versioning is enabled	gsutil versioning get gs://BUCKET_NAME

- STATE is either `on` to enable Object Versioning or `off` to disable Object Versioning.

b. Setting object life cycle management policies for Cloud Storage buckets

In order to use Object Lifecycle Management, you define a lifecycle configuration, which must be set on a bucket. The configuration contains a set of rules which apply to current and future objects in the bucket. When an object meets the criteria of one of the rules, Cloud Storage automatically performs a specified action on the object. Each lifecycle management configuration contains a set of rules. Each rule contains one action and one or more conditions. Changes to a bucket's lifecycle configuration can take up to 24 hours to go into effect, and Object Lifecycle Management might still perform actions based on the old configuration during this time. A lifecycle rule specifies exactly one of the following actions:

- Delete
- SetStorageClass
- AbortIncompleteMultipartUpload

If the expiration time is not available for an object, the object is charged for storage until the time it is deleted.To track the lifecycle management actions that Cloud Storage takes, use one of the following options:

- Use Cloud Storage usage logs. This feature logs both the action and who performed the action. A value of GCS Lifecycle Management in the cs_user_agent field of the log entry indicates the action was taken by Cloud Storage in accordance with a lifecycle configuration.
- Enable Pub/Sub Notifications for Cloud Storage for your bucket. This feature sends notifications to a Pub/Sub topic of your choice when specified actions occur. Note that this feature does not record who performed the actions.

Set the lifecycle configuration for a bucket : Create a JSON file with the lifecycle configuration rules you would like to apply. See configuration examples for sample JSON files.

To apply the configuration	gsutil lifecycle set LIFECYCLE_CONFIG_FILE gs://BUCKET_NAME
Check the lifecycle configuration for a bucket	gsutil lifecycle get gs://BUCKET_NAME

LIFECYCLE_CONFIG_FILE is the path for the JSON file that you created
Configuration examples for Object Lifecycle Management

c. Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)

BigQuery supports querying Cloud Storage data : The Cloud Storage URI comprises your bucket name and your object (filename). For example, if the Cloud Storage bucket is named `mybucket` and the data file is named `myfile.csv`, the bucket URI would be `gs://mybucket/myfile.csv`.

A federated query is a way to send a query statement to an external database and get the result back as a temporary table. Federated queries use the BigQuery Connection API to establish a connection with the external database.
Cloud SQL federated queries : A service account is automatically created when you enable the BigQuery Connection API. BigQuery Cloud SQL federation only supports Cloud SQL instances with public IP connectivity. Configure public IP connectivity for your Cloud SQL instance.

Setting up Cloud SQL database connections :

```
bq mk --connection --display_name='friendly name' --connection_type=TYPE \
    --properties=PROPERTIES --connection_credential=CREDENTIALS \
    --project_id=PROJECT_ID --location=LOCATION \
    CONNECTION_ID
```

```
bq mk --connection --display_name='friendly name' --connection_type='CLOUD_SQL' \
 --properties='{"instanceId":"federation-test:us-central1:mytestsql","database":"mydatabase","type":"MYSQL"}' \
 --connection_credential='{"username":"myusername", "password":"mypassword"}' \
 --project_id=federation-test --location=us my_connection_id
```

- TYPE: the type of the external data source.
- PROPERTIES: the parameters for the created connection in JSON format. For example: `--properties='{"param":"param_value"}'`. For creating a connection resource, you must supply the `instanceID`, `database`, and `type` parameters.
- CREDENTIALS: the parameters `username` and `password`.
- PROJECT_ID: your project ID.
- LOCATION: the region your Cloud SQL instance is located in.
- CONNECTION_ID: the connection identifier.

Setting up Spanner database connections :

```
bq mk --connection \
 --connection_type=CLOUD_SPANNER \
 --properties='PROPERTIES' \
 --location=LOCATION \
 --display_name='FRIENDLY_NAME' \
 --description 'DESCRIPTION' \
 CONNECTION_ID
```

```
bq mk --connection \
 --connection_type='CLOUD_SPANNER' \
 --properties='{"database":"projects/my_project/instances/my_instance/databases/database1"}' \
 --project_id=federation-test \
 --location=us \
 my_connection_id
```

You can use the `EXTERNAL_QUERY()` function to query information_schema tables to access database metadata, such as list all tables in the database or show table schema.

d. Estimating costs of data storage resources

Cloud Storage pricing is based on the following components:

- Data storage: the amount of data stored in your buckets. Storage rates vary depending on the storage class of your data and location of your buckets.
- Data processing: the processing done by Cloud Storage, which includes operations charges, any applicable retrieval fees, and inter-region replication.

- Network usage: the amount of data read from or moved between your buckets.

e. Backing up and restoring database instances (e.g., Cloud SQL, Datastore)

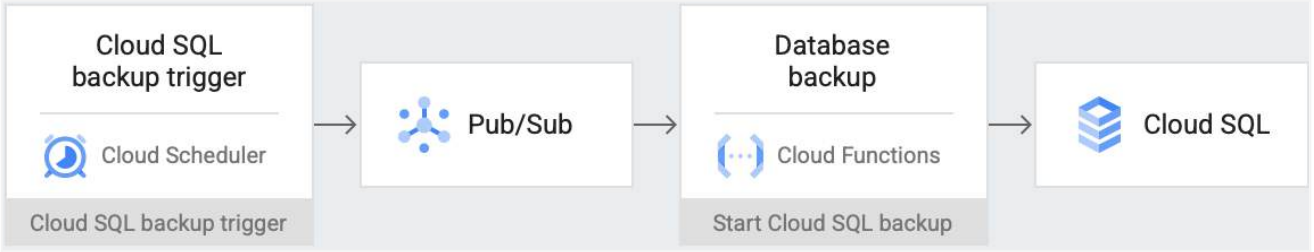
Cloud SQL backups : Backups help you restore lost data to your Cloud SQL instance.Backups protect your data from loss or damage.By default, for each instance, Cloud SQL retains seven automated backups, in addition to all on-demand backups.
Backup and restore operations can't be used to upgrade a database to a later version. You can only restore from a backup to an instance with the same database version.To upgrade to a later version, consider using the Database Migration Service or exporting and then importing your database to a new Cloud SQL instance.Cloud SQL backups are incremental. They contain only data that changed after the previous backup was taken. Cloud SQL performs two types of backups:

- On-demand backups create a backup at any time
- Automated backups backups are taken daily, within a 4-hour backup window

Automated backups can be retained for up to a year by configuring the retention period whereas on-demand backups persist until you delete the backups or until your instance is deleted. No, you can't export a backup. You can only export instance data. You are allowed a maximum of five backup operations every 50 minutes per instance per project.

Create and manage on-demand and automatic backups :

To create an on-demand backup	gcloud sql backups create \ --async \ --instance=INSTANCE_NAME
To create a backup in a custom location	gcloud sql backups create \ --async \ --instance=INSTANCE_NAME \ --location=BACKUP_LOCATION
To schedule automated backups for an instance	gcloud sql instances patch INSTANCE_NAME --backup-start-time=HH:MM
To update an instance with a custom location	gcloud sql instances patch INSTANCE_NAME \ --backup-location=BACKUP_LOCATION
List the backups of the instance View the details of a backup using ID	gcloud sql backups list \ --instance INSTANCE_NAME gcloud sql backups describe BACKUP_ID \ --instance INSTANCE_NAME
Delete a backup of a Cloud SQL instance	gcloud beta sql backups delete BACKUP_ID \ --instance INSTANCE_NAME
To disable automated backups for an instance	gcloud sql instances patch INSTANCE_NAME \ --no-backup



Restore an instance : When you restore an instance, the following data from the primary instance are restored to the new instance : Database and Users. Point-in-time recovery (PITR) helps you recover an

instance to a specific point in time. Point-in-time recovery is enabled by default when you create a new Cloud SQL instance and always creates a new instance.

Restore to the same instance

1. Describe the instance 2. Delete all replicas : Repeat for all replicas 3. List the backups for the instance: Find the backup you want to use and record its ID value **Note:** Select a backup that is marked **SUCCESSFUL**. 4. Restore the instance from the specified backup	gcloud sql instances describe INSTANCE_NAME gcloud sql instances delete REPLICA_NAME gcloud sql backups list --instance INSTANCE_NAME gcloud sql backups restore BACKUP_ID \ --restore-instance=INSTANCE_NAME
Restore to a different instance	gcloud sql backups restore BACKUP_ID \ --restore-instance=TARGET_INSTANCE_NAME \ --backup-instance=SOURCE_INSTANCE_NAME

Datastore is a NoSQL document database built for automatic scaling, high performance, and ease of application development. **Note:** Firestore, the newest version of Datastore, makes all queries strongly consistent.

Exporting and Importing Entities : With the managed export and import service, you can recover from accidental deletion of data and export data for offline processing. You can export all entities or just specific kinds of entities. Likewise, you can import all data from an export or only specific kinds.

To export all entities in your database	gcloud datastore export gs://bucket-name --async
To list long-running operations	gcloud datastore operations list
to show the status of a long-running operation	gcloud datastore operations describe operation-name

f. Reviewing job status in Dataproc, Dataflow, or BigQuery

Life of a Dataproc Job : If a job fails, you can access job logs in Logging. Dataproc job and cluster logs can be viewed, searched, filtered, and archived in Cloud Logging.

To examine a running job's status	gcloud dataproc jobs describe job-id \ --region=region

Using the Dataflow monitoring interface : monitoring interface lets you see and interact with your Dataflow jobs. A list of all currently running Dataflow jobs and previously run jobs within the last 30 days. The Dataflow service automatically chooses the number of worker instances required to run your autoscaling job. The number of worker instances can change over time according to the job requirements.
Obtain information about your Dataflow job by using the Dataflow command-line interface :
Note: For a complete list of all available Dataflow commands and associated documentation, see the Dataflow command-line reference documentation for Google Cloud CLI.

To see the list of available Dataflow commands	gcloud dataflow --help
To get a list of all the Dataflow jobs in your project	gcloud dataflow jobs list
To display more information about a job	gcloud dataflow jobs describe JOB_ID

Managing jobs : you can view job details, list jobs, cancel a job, repeat a job, or delete job metadata.

To view job details Example myproject	bq --location=LOCATION show --job=true JOB_ID bq show --job=true myproject:US.bquijob_123x456_123y123z123c
To cancel a job	bq --location=LOCATION --nosync cancel JOB_ID bq --nosync cancel my-project-1234:US.bquijob_123x456_123y123z123c

4.5 Managing networking resources. Tasks include:

a. Adding a subnet to an existing VPC

Work with subnets : The primary IPv4 range for the subnet can be expanded, but not replaced or shrunk, after the subnet has been created.The minimum primary or secondary range size is eight IPv4 addresses. In other words, the longest subnet mask you can use is /29.

Add an IPv4 only subnet	gcloud compute networks subnets create SUBNET \ --network=NETWORK \ --range=PRIMARY_RANGE \ --region=REGION
Add a dual-stack subnet	gcloud compute networks subnets create SUBNET \ --network=NETWORK \ --range=PRIMARY_IPv4_RANGE \ --stack-type=IPV4_IPV6 \ --ipv6-access-type=IPv6_ACCESS_TYPE \ --region=REGION

- NETWORK is the name of the VPC network that will contain the new subnet.
- PRIMARY_IPv4_RANGE is the primary IPv4 range for the new subnet, in CIDR notation. For more information, see IPv4 subnet ranges.
- IPv6_ACCESS_TYPE is the IPv6 access type. It can be `EXTERNAL` or `INTERNAL`.
- REGION is the Google Cloud region in which the new subnet will be created.

b. Expanding a subnet to have more IP addresses

Expand a primary IPv4 range : You can expand the primary IPv4 range of an existing subnet by modifying its subnet mask, setting the prefix length to a *smaller* number.Any prefix broader than `/16` would conflict with the primary IPv4 ranges of the other automatically created subnets.

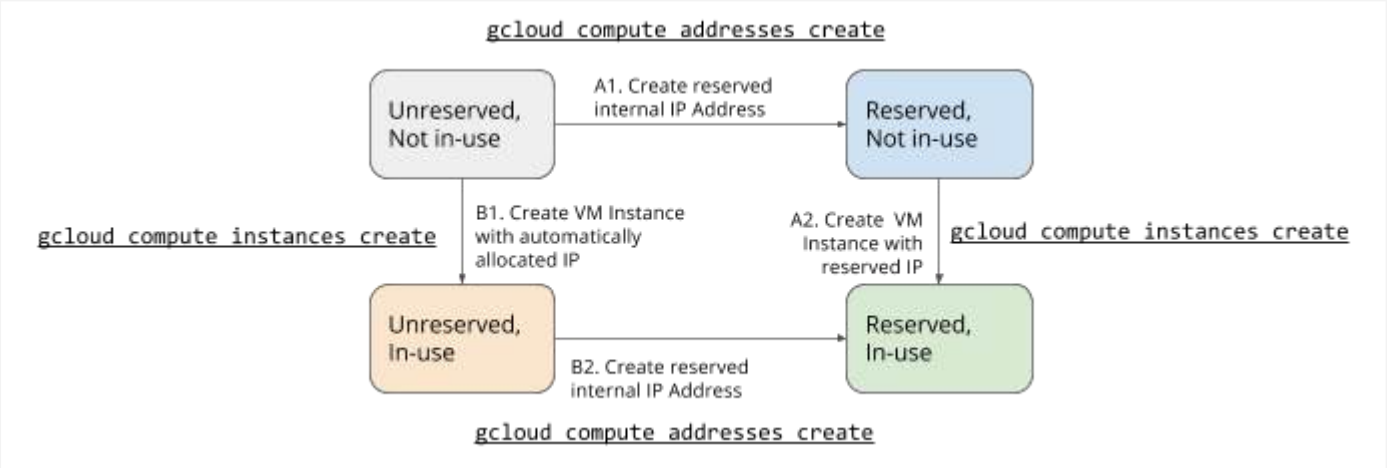
PREFIX_LENGTH is a subnet mask size in bits. If the primary IPv4 range is 10.1.2.0/24, you can supply 20 to reduce the subnet mask to 20 bits, which changes the primary IPv4 range to 10.1.2.0/20. For valid ranges, see IPv4 subnet ranges.	gcloud compute networks subnets expand-ip-range SUBNET \ --region=REGION \ --prefix-length=PREFIX_LENGTH

c. Reserving static external or internal IP addresses

Reserving a static external IP address : go to the **External IP addresses** page. Use the compute addresses list sub-command to see a list of static external IP addresses available to the project.

For global IP address, use the `--global` and `--ip-version` fields specify either `IPV4` or `IPV6`. Global IPv6 addresses can only be used with global load balancers.	gcloud compute addresses create ADDRESS_NAME \ --global \ --ip-version [IPV4 | IPV6]

reserve a regional IP address, use the --region field	gcloud compute addresses create ADDRESS_NAME \ --region=REGION
To view the result	gcloud compute addresses describe ADDRESS_NAME
Assign a static external IP address to a new VM instance	gcloud compute instances create VM_NAME --address=IP_ADDRESS



Reserving a static internal IP address : Go to the **IP addresses** page

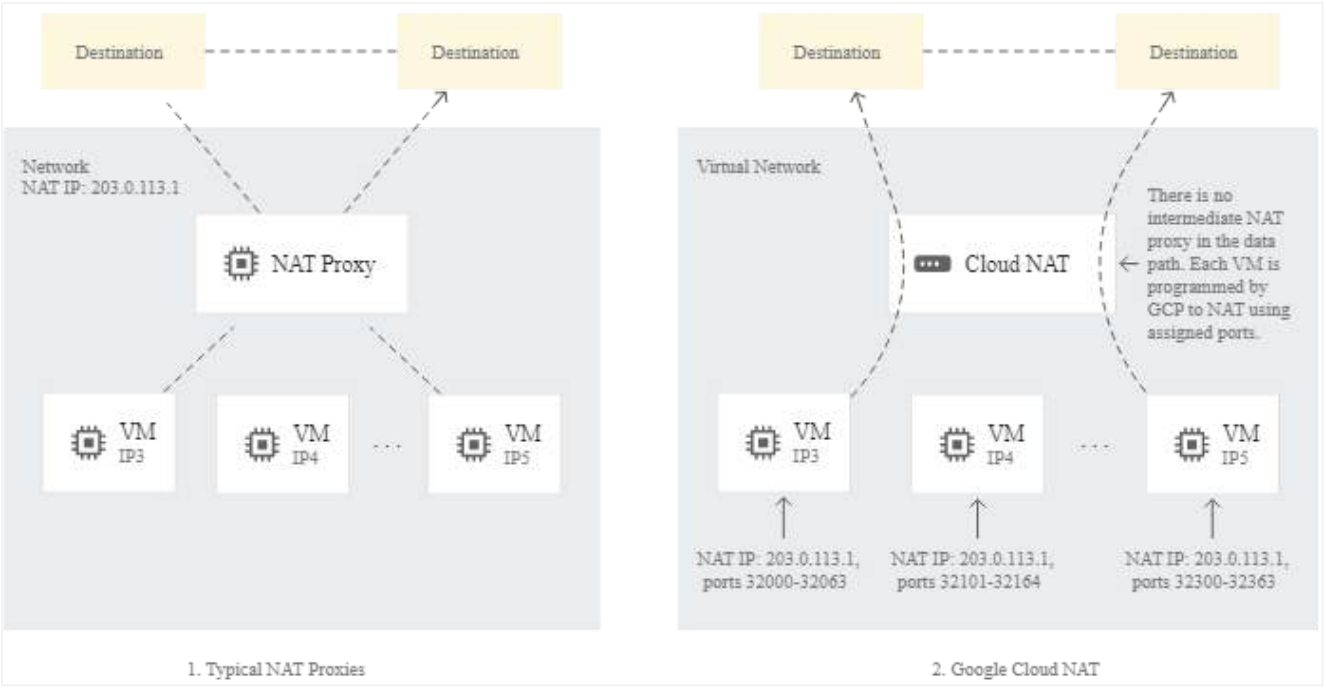
Reserve a new static internal IP address	gcloud compute addresses create ADDRESS_NAME [ADDRESS_NAME..] \
	  --region REGION --subnet SUBNETWORK \
	  --addresses IP_ADDRESS

To reserve an automatically allocated internal IP address from a subnet	gcloud compute addresses create example-address-1 \ --region us-central1 --subnet subnet-1
To reserve a specific internal IP address from a subnet	gcloud compute addresses create example-address-1 \ --region us-central1 --subnet subnet-1 --addresses 10.128.0.12
Create a VM instance with a specific internal IP address	gcloud compute instances create VM_NAME --private-network-ip IP_ADDRESS

d. Working with CloudDNS, CloudNAT, Load Balancers and firewall rules

Cloud DNS : for creating, updating, listing, and deleting Cloud DNS managed zones.

CloudNAT : Google Cloud-managed high-performance network address translation. Cloud NAT is a distributed, software-defined managed service. It's not based on proxy VMs or appliances. Cloud NAT configures the Andromeda software that powers your Virtual Private Cloud (VPC) network so that it provides *source network address translation (source NAT or SNAT)* for VMs without external IP addresses. Cloud NAT also provides *destination network address translation (destination NAT or DNAT)* for established inbound response packets.



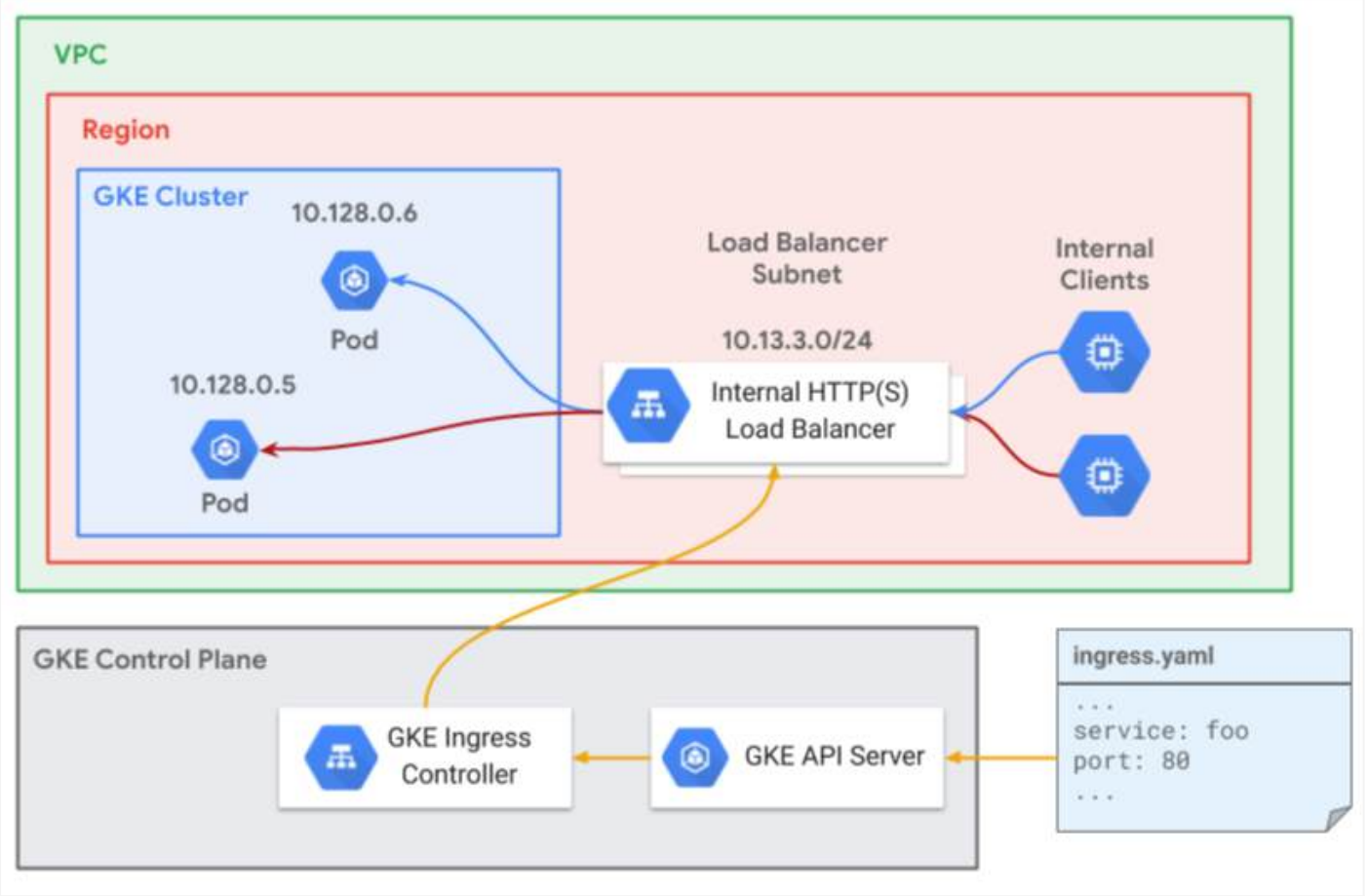
A *NAT IP address* is a regional external IP address, routable on the internet.
Set up Cloud NAT with Compute Engine Create NAT

Availability - in a GKE cluster, availability deals with both the control plane and the distribution of your nodes. A zonal cluster has a single control plane in a single zone. You can distribute the nodes of a zonal cluster across multiple zones, providing node availability in case of a node outage. A regional cluster, on the other hand, has multiple replicas of the control plane in multiple zones with a given region. Nodes in a regional cluster are replicated across three zones, though you can change this behavior as you add new node pools.
Network routing: Routing between pods in GKE can be accomplished using alias IPs or Google Cloud Routes. The first option is also known as a VPC-native cluster, and the second one is called a routes-based cluster.
Network Isolation: Public GKE networks let you set up routing from public networks to your cluster. Private networks use internal addresses for pods and nodes and are isolated from public networks.
Containers in GKE are based on images which are shared via the Google Container registry.

Ingress for Internal HTTP(S) Load Balancing
In GKE, the internal HTTP(S) load balancer is a proxy-based, regional, Layer 7 load balancer that enables you to run and scale your services behind an internal load balancing IP address. GKE Ingress objects support the internal HTTP(S) load balancer natively through the creation of Ingress objects on GKE clusters.

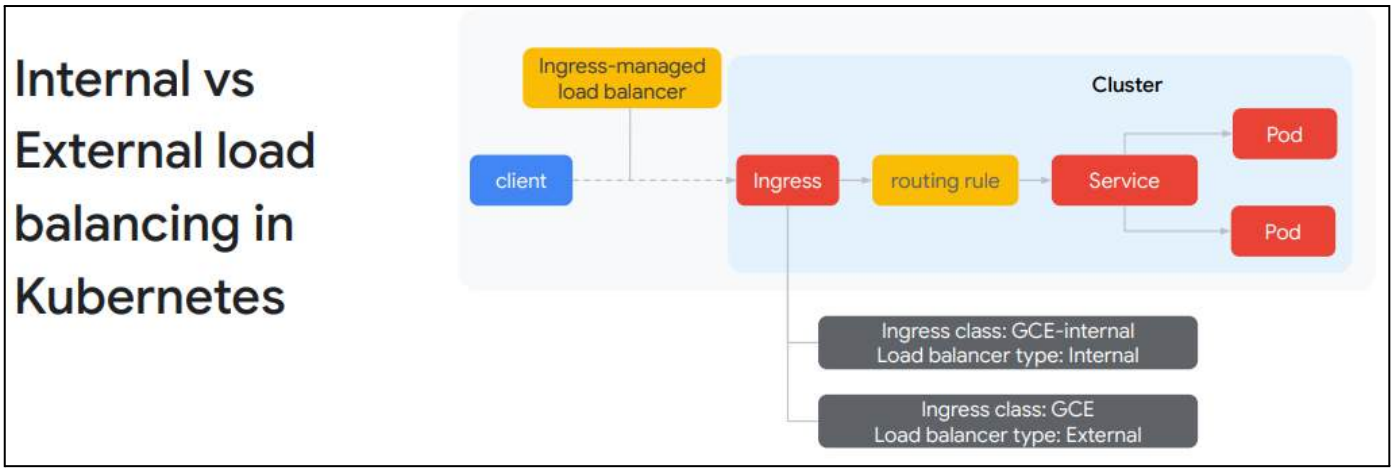
- Container-native load balancing with Network Endpoint Groups (NEG).
- Self-managed SSL Certificates using Google Cloud. Only regional certificates are supported for this feature.
- Self-managed SSL Certificates using Kubernetes Secrets.

Important: Ingress for Internal HTTP(S) Load Balancing requires you to use NEGs as backends. It does not support Instance Groups as backends.
The following diagram provides an overview of the traffic flow for internal HTTP(S) load balancer, as described in the preceding paragraph.



The GKE Ingress Controller manages deployment of firewall rules, so you do not need to manually deploy them.
To configure the protocol used between the load balancer and your application, use the cloud.google.com/app-protocols annotation in your Service manifest.
The following Service manifest specifies two ports. The annotation specifies that an internal HTTP(S) load balancer should use HTTP when it targets port 80 of the Service, And use HTTPS when it targets port 443 of the Service.
You must use the port's name field in the annotation. Do not use a different field such as targetPort.
learn how to set up and use Ingress for Internal HTTP(S) Load Balancing.

Ingress for External HTTP(S) Load Balancing
Google Cloud's external HTTP(S) load balancer is a globally distributed load balancer for exposing applications publicly on the internet. It's deployed across Google Points of Presence (PoPs) globally providing low latency HTTP(S) connections to users. Anycast routing is used for the load balancer IPs, allowing internet routing to determine the lowest cost path to its closest Google Load Balancer.
GKE Ingress deploys the external HTTP(S) load balancer to provide global load balancing natively for Pods as backends.
Configuring Ingress for external load balancing
This page shows you how to configure an external HTTP(S) load balancer by creating a Kubernetes Ingress object. An Ingress object must be associated with one or more Service objects, each of which is associated with a set of Pods.
A Service object has one or more servicePort structures. Each servicePort that is targeted by an Ingress is associated with a Google Cloud backend service resource.



VPC firewall rules overview : If you want to apply firewall rules to multiple VPC networks in an organization, see Firewall Policies.VPC firewall rules let you allow or deny connections to or from your virtual machine (VM) instances based on a configuration that you specify. Enabled VPC firewall rules are always enforced, protecting your instances regardless of their configuration and operating system, even if they have not started up.
Using firewall rules : Firewall rules are defined at the network level, and only apply to the network where they are created. A firewall rule can contain either IPv4 or IPv6 ranges, but not both. When you create a firewall rule, you can choose to enable Firewall Rules Logging. If you enable logging, you can omit metadata fields to save storage costs. For more information, see Using Firewall Rules Logging.

4.6 Monitoring and logging. Tasks include:

a. Creating Cloud Monitoring alerts based on resource metrics

Cloud Monitoring : Gain visibility into the performance, availability, and health of your applications and infrastructure.It also supports monitoring of hybrid and multi cloud environments.
Alerting gives timely awareness to problems in your cloud applications so you can resolve the problems quickly.In Cloud Monitoring, an *alerting policy* describes the circumstances under which you want to be alerted and how you want to be notified.
Alerting policies that are used to track metric data collected by Cloud Monitoring are called *metric-based* alerting policies.Monitoring continuously monitors the conditions of that policy. When the conditions of that policy are met, Monitoring creates an incident and sends a notification about the incident creation. This notification includes summary information about the incident, a link to the **Policy details** page so that you can investigate the incident, and any documentation that you specified. If an incident is open and Monitoring determines that the conditions of the metric-based policy are no longer met, then Monitoring automatically closes the incident and sends a notification about the closure.
Cloud Monitoring supports Cloud Mobile App and Pub/Sub in addition to common notification channels.
Notification options

1. Creating alerting policies by using the console
2. Monitor a Compute Engine virtual machine
3. Creating alerting policies by using the Cloud Monitoring API or Google Cloud CLI.
4. Creating alerting policies by using Monitoring Query Language (MQL).

When you use the console, you can enable a recommended alert or you can create an alert by starting from the **Alerts** page of Cloud Monitoring. Example Pub/Sub Lite **Topics** page links to alerts that are configured to notify you when you're reaching a quota limit. Similarly, the **VM Instances** page from within Monitoring links to alerting policies that are configured to monitor the memory utilization and network latency of those instances.

When you use the Cloud Monitoring API, you can specify the ratio by using Monitoring Query Language (MQL) or by using Monitoring filters see Metric ratio.Required console IAM roles to create an alerting policy : 1] Monitoring Editor 2] Monitoring Admin 3] Project Owner

A *metric-absence condition* triggers when a monitored time series has no data for a specific duration window.
A *metric-threshold condition* triggers when the values of a metric are more than, or less than, the threshold for a specific duration window.

Condition type	JSON example	console
Metric threshold	View	Instructions
Rate of change	View	Instructions
Group aggregate	View	Instructions
Uptime check	View	Instructions
Process health	View	Instructions
Metric ratio	View	Instructions

The alignment period (console : **Rolling window** and **Rolling window function**)is a look-back interval from a particular point in time. For example, when the alignment period is five minutes, at 1:00 PM, the alignment period contains the samples received between 12:55 PM and 1:00 PM. At 1:01 PM, the alignment period slides one minute and contains the samples received between 12:56 PM and 1:01 PM.
Set the duration window to be long enough to minimize false positives, but short enough to ensure that incidents are opened in a timely manner.configure the duration window by using the **Retest window** field in the **Configure trigger** step.
The alignment period determines how many data samples are combined with the aligner. To combine many samples, choose a long period. To restrict the interval to as few as one sample, choose a short period. In contrast, the duration window specifies how long the aligned values must be greater than the threshold before the condition is met. To let the condition be met when a single aligned value is greater than the threshold, set the duration window to zero.

To configure multiple conditions are combined in the **Multi-condition trigger** step.
Note: The alerting policy uses the label values to identify the resource that caused a condition to be met.
This table lists the settings in the console, the equivalent value in the Cloud Monitoring API, and a description of each setting:

console **Policy triggers value**	**Cloud Monitoring API** **combiner value**	**Meaning**
Any condition is met	OR	An incident is opened if any resource causes any of the conditions to be met.
All conditions are met even for different resources for each condition (default)	AND	An incident is opened if *each* condition is met, even if a *different* resource causes those conditions to be met.
All conditions are met	AND_WITH_MATCHING_RESOURCE	An incident is opened if the *same* resource causes *each* condition to be met. This setting is the most stringent combining choice.

Delays in data arriving from third-party cloud providers can be as high as 30 minutes, with 5-15 minute delays being the most common.There are two configurable fields that specify how Monitoring evaluates metric-threshold conditions when data stops arriving:

- To configure how Monitoring determines the replacement value for missing data, use the **Evaluation missing data** field which you set in the **Condition trigger** step. This field is disabled when the retest window is set to **No retest**.

- To configure how long Monitoring waits before closing an open incident after data stops arriving, use the **Incident autoclose duration** field. You set the auto-close duration in the **Notification** step. The default auto-close duration is seven days.

number of incidents that a single policy can open simultaneously is limited to 5000.

Note: You can't configure Cloud Monitoring to create a single incident and send a single notification when the policy contains multiple conditions.

Alerting policies created by using the Cloud Monitoring API notify you when the condition is met and when the condition stops being met. By default, alerting policies created by using the Google Cloud console notify you when an incident is opened. They don't notify you when an incident is closed. You can enable notifications on incident closure.

The following describes the different options for the missing data field:

console "Evaluation missing data" field	Summary	Details
Missing data empty	Open incidents stay open. New incidents aren't opened.	For conditions that are met, the condition continues to be met when data stops arriving. If an incident is open for this condition, then the incident stays open. When an incident is open and no data arrives, the auto-close timer starts after a delay of at least 15 minutes. If the timer expires, then the incident is closed. For conditions that aren't met, the condition continues to not be met when data stops arriving.
Missing data points treated as values that violate the policy condition	Open incidents stay open. New incidents can be opened.	For conditions that are met, the condition continues to be met when data stops arriving. If an incident is open for this condition, then the incident stays open. When an incident is open and no data arrives for the auto-close duration plus 24 hours, the incident is closed. For conditions that aren't met, this setting causes the metric-threshold condition to behave like a metric-absence condition. If data doesn't arrive in the time specified by the retest window, then the condition is evaluated as met. For an alerting policy with one condition, the condition being met results in an incident being opened.
Missing data points treated as values that don't violate the policy condition	Open incidents are closed. New incidents aren't opened.	For conditions that are met, the condition stops being met when data stops arriving. If an incident is open for this condition, then the incident is closed. For conditions that aren't met, the condition continues to not be met when data stops arriving.

Disabling an alerting policy prevents the policy from triggering or closing incidents, but it doesn't stop Cloud Monitoring from evaluating the policy conditions and recording the results. To close open issues after you disable an alerting policy, silence the corresponding incidents. For information on that process, see Silencing incidents.

Notification latency is the delay from the time a problem first starts until the time a policy is triggered.

Labels are key-value pairs that are associated with time series, alerting policies, and incidents. There are metric labels, resource labels, and user-defined labels. Metric and resource labels contain specific information about the metric being collected or the resource against which the metric is written. In contrast, user-defined labels are those labels that you create and that record information specific to your needs. When time-series data is written, labels are attached to the data to record information about that data. For example, the labels on a time series might identify a virtual machine (VM), a zone, a Google Cloud project, and a device type.

Report severity level with labels : use the value of the severity label to determine which incident to investigate first.The time-series data generated by the policy handler is the input to the *incident manager*, which determines when incidents are created and closed. To determine when to close an incident, the incident manager uses the values of the duration, evaluationMissingData, and autoClose fields.

If you don't use the recommended value for the evaluationMissingData field, then when data stops arriving, open incidents aren't closed immediately. The result is that you might see multiple open incidents for the same input time series.
Create metric-based alert policies :

1. In the console, select **Monitoring**
2. In the navigation pane, select notifications **Alerting** and then click **Create policy**.
3. Select the time series to be monitored then Click **Select a metric**

Create a notification channel : Managing notification channels by API

b. Creating and ingesting Cloud Monitoring custom metrics (e.g., from applications or logs)

Cloud Monitoring collects measurements to help you understand how your applications and system services are performing. A collection of these measurements is generically called a *metric*. The applications and system services being monitored are called *monitored resources*.

Custom metrics let you capture application-specific data or client-side system data.To keep your metric data beyond the retention period, you must manually copy the data to another location, such as Cloud Storage or BigQuery. **Note:** Custom metrics are a chargeable feature of Cloud Monitoring and there could be costs for your custom metrics.
A monitored-resource type that represents where your data comes from, and then use that to describe the specific origin. A common practice is to use the monitored resource objects that represent the physical resources where your application code is running.
Monitoring your logs Create *log-based* alerting policies, which notify you when a particular message appears in your logs.

Create custom metrics with OpenCensus : OpenCensus is a set of libraries for various languages that allow you to collect application metrics and distributed traces, then transfer the data to a backend of your choice in real time. This data can be analyzed by developers and admins to understand the health of the application and debug problems.
The Stackdriver exporter exports the metrics that OpenCensus collects to your Google Cloud project. You can then use Cloud Monitoring to chart or monitor those metrics. the component of OpenCensus that sends metric data to Cloud Monitoring is called the “stats exporter for Stackdrdiver”

- Write custom metrics with OpenCensus
- Read OpenCensus metrics in Cloud Monitoring

Create custom metrics with the API : To create a custom metric, you either define a MetricDescriptor object that specifies various information about the metric, or you write metric data. When you write metric data, Monitoring creates the metric descriptor for you based on the structure of the data you provide. To write your data, use the timeSeries.create method. To delete a custom metric, delete its metric descriptor. You can't delete the time-series data stored in your Google Cloud project; however, deleting the metric descriptor renders the data inaccessible. The data expires and is deleted according to the data retention policy.
To modify a custom metric, you must update the MetricDescriptor object that defines the custom metric.

Log-based metrics : apply only to a single Google Cloud project. 2 kinds of log-based metrics

1. System-defined log-based metrics are calculated only from logs that have been ingested by Logging.metrics are calculated from included logs only.
2. User-defined log-based metrics, created by you to track things in your Google Cloud project that are of particular interest to you. metrics are calculated from both included and excluded logs.

The system log-based metrics have predefined labels. You can define the labels for user-defined metrics.

Counter metrics count the number of log entries matching a given filter. *Distribution metrics* accumulate numeric data from log entries matching a filter.

Logging and Monitoring in the Cloud

1. Error Reporting
    a. Counts, analyzes, and aggregates the crashes in your running cloud services.
    b. Management interface displays the results with sorting and filtering capabilities.
    c. A dedicated view shows the error details: time chart, occurrences, affected user count, first- and last-seen dates, and a cleaned exception stack trace.
    d. Create alerts to receive notifications on new errors.
2. Debugger
    a. Debugs applications while running in production to examine code's function and performance under actual production conditions.
    b. Easy collaboration with other team members by sharing debug sessions with a Console URL.
    c. An application's state in production can be captured at a specific line location with snapshots.
    d. Easily integrates into existing developer workflows.
    e. Integrates with version control systems, such as Cloud Source Repositories, GitHub, Bitbucket, or GitLab.
3. Cloud Trace
    a. Collects latency data from distributed applications and displays it in the Google Cloud Console.
    b. Captures traces from applications deployed on App Engine, Compute Engine VMs, and Kubernetes Engine containers.
    c. Performance insights are provided in near-real time.
    d. Automatically analyzes all of your application's traces to generate in-depth latency reports to surface performance degradation's.
    e. Continuously gathers and analyzes trace data to automatically identify recent changes to application performance.
4. Cloud Profiler
    a. Uses statistical techniques and extremely low-impact instrumentation that runs across all production application instances to provide a complete CPU and heap picture of an application.
    b. Allows developers to analyze applications running anywhere, including Google Cloud, other cloud platforms, or on-premises, with support for Java, Go, Python, and Node.js.
    c. Presents the call hierarchy and resource consumption of the relevant function in an interactive flame graph.

c. Configuring log sinks to export logs to external systems (e.g., on-premises or BigQuery)

Sinks control how Cloud Logging routes logs. Using sinks, you can route some or all of your logs to supported destinations. You can create up to 200 sinks per Cloud project. You can create up to 50 exclusion filters per sink. Note that the length of a filter can't exceed 20,000 characters. Sinks to Cloud Storage are processed hourly while other destination types are processed in real time.
Cloud Logging provides two predefined sinks for each Cloud project, billing account, folder, and organization: _Required and _Default. All logs that are generated in a resource are automatically processed through these two sinks and then are stored either in the correspondingly named _Required or _Default buckets.want to combine and route logs from the resources contained by a Google Cloud organization or folder, you can create aggregated sinks.

To create a sink	gcloud logging sinks create SINK_NAME SINK_DESTINATION OPTIONAL_FLAGS
sink destination BigQuery data	bigquery.googleapis.com/projects/PROJECT_ID/datasets/DATASET_ID
_Default sink is now disabled To delete a sink	gcloud logging sinks list gcloud logging sinks describe SINK_NAME gcloud logging sinks update _Default --disabled gcloud logging sinks delete SINK_NAME
To create an aggregated sink for your folder	gcloud logging sinks create SINK_NAME SINK_DESTINATION --include-children \ --folder=FOLDER_ID filter
destination is a BigQuery dataset	gcloud logging sinks create SINK_NAME \ bigquery.googleapis.com/projects/PROJECT_ID/datasets/DATASET_ID --include-children \ --folder=FOLDER_ID --log-filter="logName:activity"

View logs in sink destinations

d. Configuring log routers

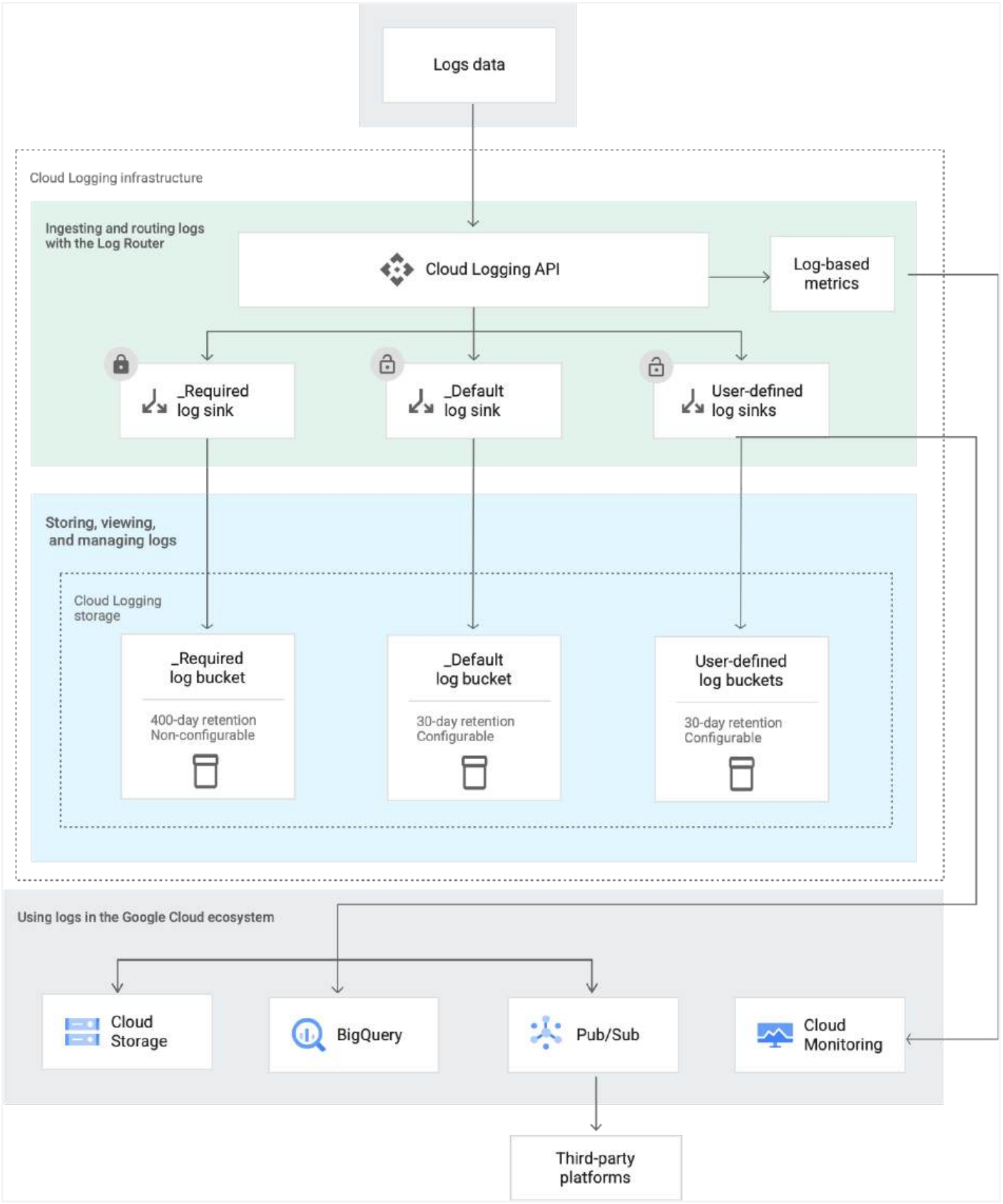
All logs, including audit logs, platform logs, and user-written logs, are sent to the Cloud Logging API where they pass through the *Log Router*. The Log Router checks each log entry against existing rules to determine which log entries to ingest (store) into log buckets, which log entries to route to a destination, and which log entries to exclude (discard). The sinks in the Log Router check each log entry against the existing inclusion filter and exclusion filters that determine which destinations, including Cloud Logging buckets, that the log entry should be sent to.To reliably route logs, the Log Router also stores the logs temporarily

Cloud Logging uses log buckets to store and organize your logs data.Logging automatically creates two log buckets: _Required and _Default. Logging automatically creates sinks named _Required and _Default that, in the default configuration, route logs to the correspondingly named buckets.Cloud Logging retains the logs in this bucket for 400 days; you can't change this retention period.You can't modify or delete the _Required bucket. You can't disable the _Required sink, which routes logs to the _Required bucket.

Neither ingestion pricing nor storage pricing applies to the logs data stored in the _Required log bucket.

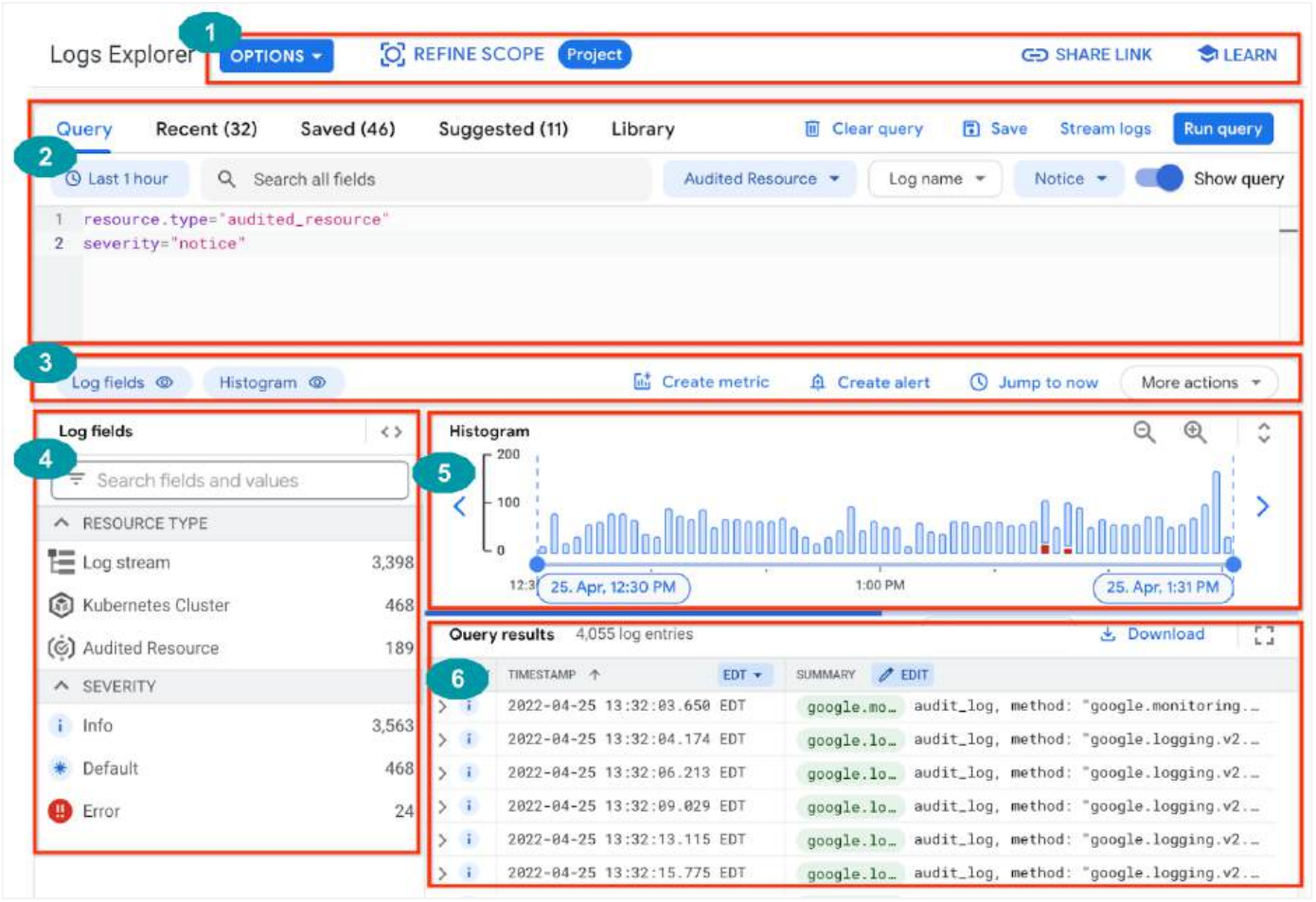
Logs held in the _Default bucket are retained for 30 days, unless you configure custom retention for the bucket.Cloud Logging pricing applies to the logs data held in the _Default bucket. Log buckets are regional resources.
In the console, select **Logging** (Operations) then Logs Router and create a sink. The **Logs Storage** page in the console tracks the volume of logs data ingested by log buckets.A deleted bucket stays in this pending state for 7 days, and Logging continues to route logs to the bucket during that time.

create a log bucket	gcloud logging buckets create BUCKET_ID --location=LOCATION OPTIONAL_FLAGS

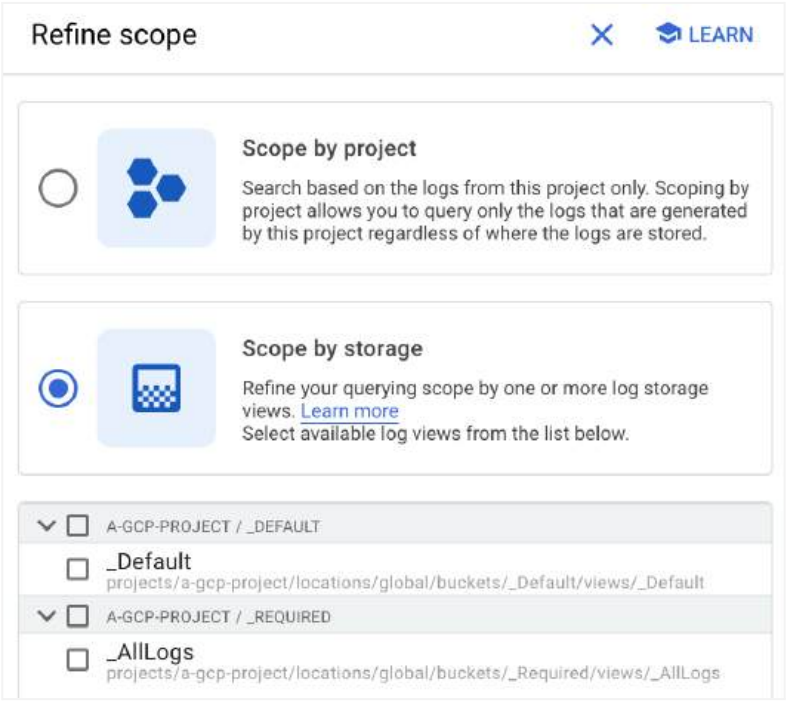


e. Viewing and filtering logs in Cloud Logging

Cloud Logging has introduced a new version of its interface for analyzing logs data, the Logs Explorer.



1. **Action** toolbar
2. **Query** pane
3. **Results** toolbar
4. **Log fields** pane
5. **Histogram**
6. **Query results** pane



You can refine the scope of the logs displayed in the Logs Explorer through the **Refine scope** option.

Streaming logs lets you view log entries in real time and is available in the Logs Explorer.

Live tailing lets you view your log entries in real time as Cloud Logging writes them, by using either the Google Cloud CLI or the Cloud Logging API.

You can view logs for your service in a couple of ways:

- Use the Cloud Run page in the console
- Use Cloud Logging Logs Explorer in the console.

Cloud Run has two types of logs, and these are automatically sent to Cloud Logging:

- Request logs: logs of requests sent to Cloud Run services. These logs are created automatically.
- Container logs: logs emitted from the container instances, typically from your own code, written to supported locations as described in Writing container logs.

Viewing logs using the command line in a format optimized ("prettified")

```
gcloud logging read "resource.type=cloud_run_revision AND resource.labels.service_name=SERVICE"
--project PROJECT-ID --limit 10
```

Configure log-based alerts to notify you whenever a specific message appears in your included logs.

- Create a log-based alert (Logs Explorer)
- Manage log-based alerts in Monitoring
- Create a log-based alert (Monitoring API)

Note: 10,000 is the maximum number of logs you can download.

f. Viewing specific log message details in Cloud Logging

g. Using cloud diagnostics to research an application issue (e.g., viewing Cloud Trace data, using Cloud Debug to view an application point-in-time)

Cloud Trace is a distributed tracing system that collects latency data from your applications and displays it in the Google Cloud Console.Cloud Trace automatically analyzes all of your application's traces to generate in-depth latency reports to surface performance degradations, and can capture traces from all of your VMs, containers, or App Engine projects.

In the Google Cloud console, the Cloud Trace **Overview** page displays a summary of the latency data for your application and lets you investigate individual traces. The **Insights** pane displays a list of performance insights for your application.The **Insights** pane highlights common problems in your application and can help you reduce latency for more information, see Viewing trace details. The **Chargeable Trace Spans** pane displays information related to your costs

Cloud Debugger is a feature of Google Cloud Platform that lets you inspect the state of an application, at any code location, without stopping or slowing down the running app. Cloud Debugger makes it easier to view the application state without adding logging statements.

h. Viewing Google Cloud status

Incidents and the Google Cloud Service Health Dashboard : The Google Cloud Service Health (CSH) Dashboard provides status information of the Google Cloud services organized by region and global locale. During a major incident, the status of the issue is communicated through the Google Workspace Status Dashboard and the Google Cloud Service Health Dashboard.

Different target platforms:

- Scenarios for routing Cloud Logging data: Splunk
- Routing Cloud Logging data to Elastic Cloud

Section 5. Configuring access and security

5.1 Managing Identity and Access Management (IAM). Tasks include:

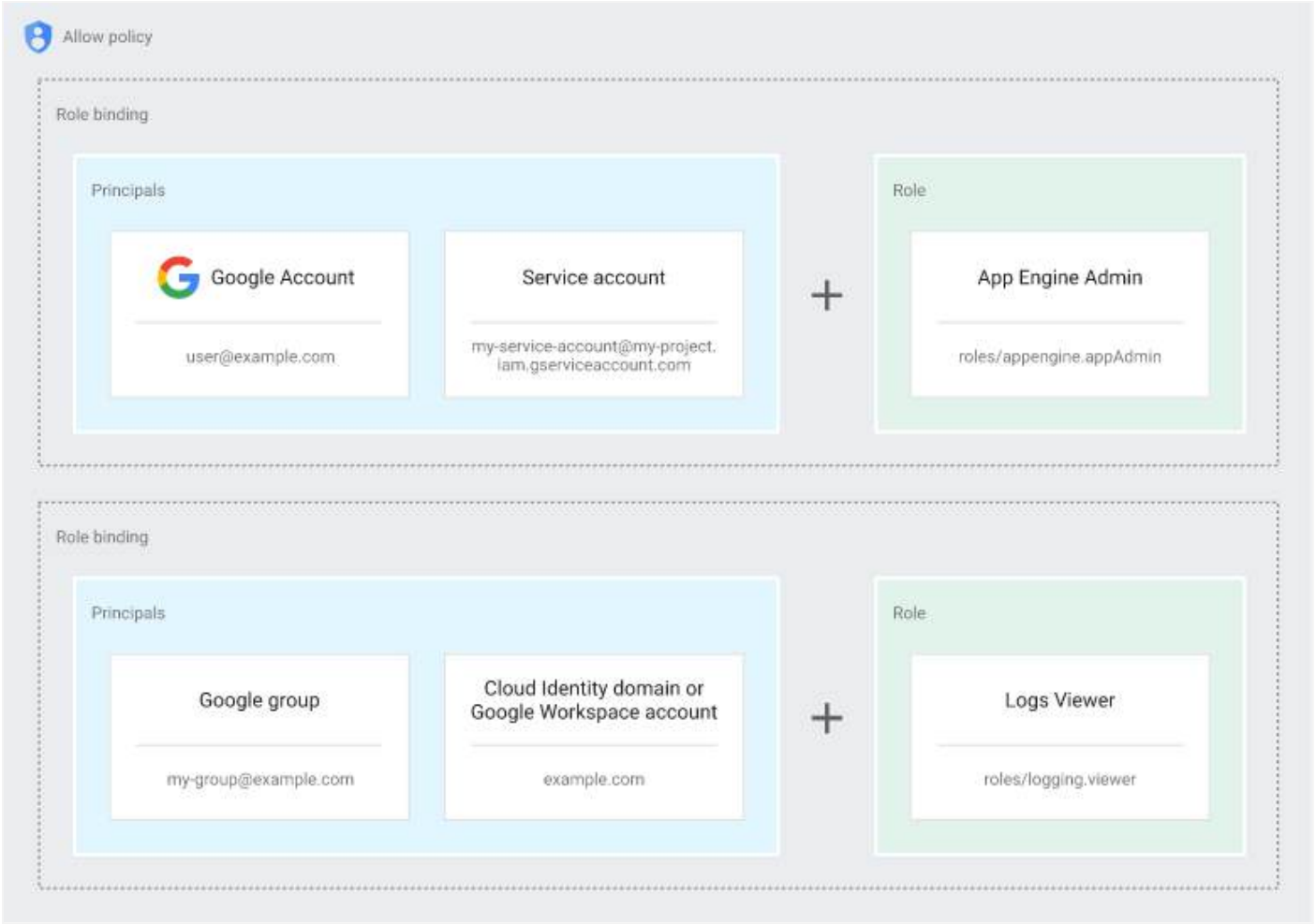
a. Viewing IAM policies

IAM lets you grant granular access to specific Google Cloud resources and helps prevent access to other resources. IAM lets you adopt the security principle of least privilege, which states that nobody should have more permissions than they actually need.

With IAM, you manage access control by defining *who* (identity) has *what access* (role) for *which resource*.In IAM, permission to access a resource isn't granted *directly* to the end user. Instead, permissions are grouped into *roles*, and roles are granted to authenticated *principals*.

An *allow policy*, also known as an *IAM policy*, defines and enforces what roles are granted to which principals. Each allow policy is attached to a resource. When an authenticated principal attempts to access a resource, IAM checks the resource's allow policy to determine whether the action is permitted.

The organization is divided into folders, and the folders into projects. Identity and Access Management lets your users and groups access Google Cloud resources.

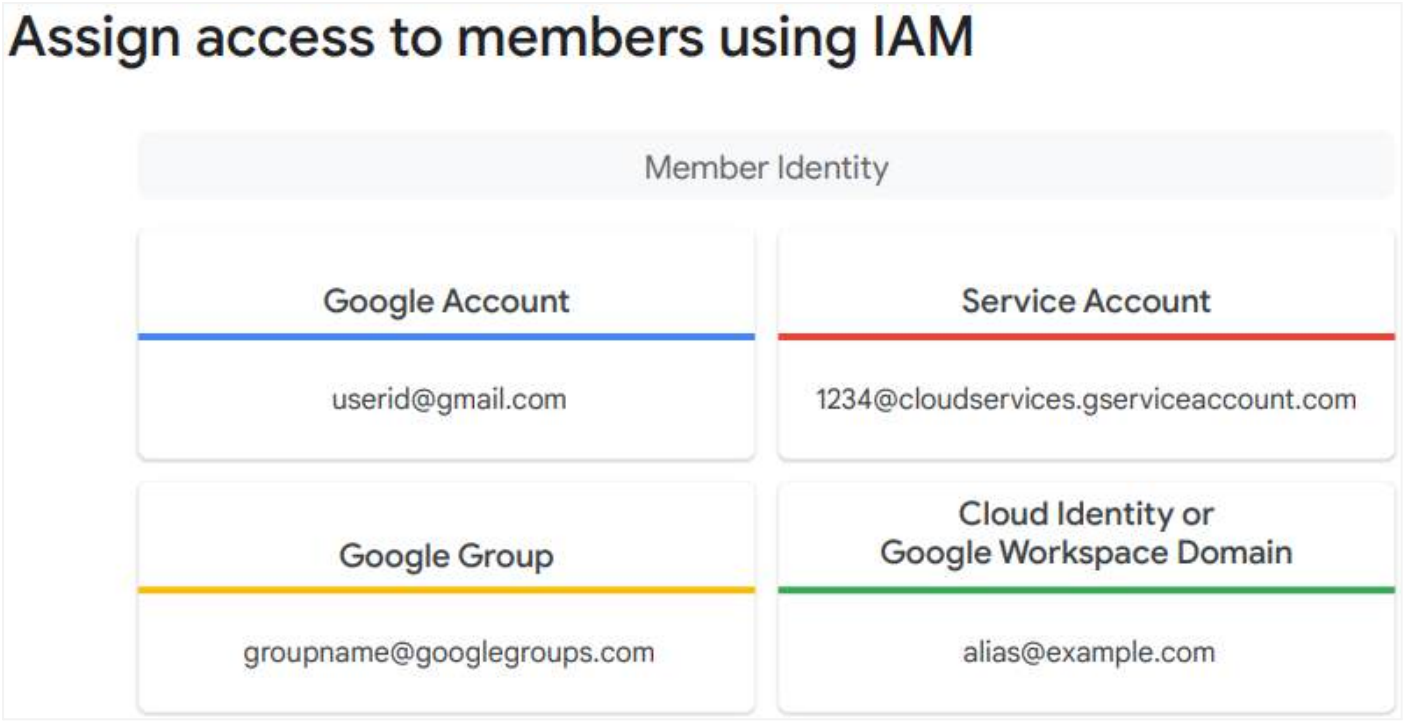


This model for access management has three main parts:

- **Principal**. A *principal* can be a Google Account (for end users), a service account (for applications and compute workloads), a Google group, or a Google Workspace account or Cloud Identity domain that can access a resource. The identity of a principal is an email address associated with a user, service account, or Google group; or a domain name associated with a Google Workspace account or a Cloud Identity domain.
- **Role**. A *role* is a collection of permissions. Permissions determine what operations are allowed on a resource. When you grant a role to a principal, you grant all the permissions that the role contains.
- **Policy**. The *allow policy* is a collection of role bindings that bind one or more principals to individual roles. When you want to define who (principal) has what type of access (role) on a resource, you create an allow policy and attach it to the resource.

A Google group is a named collection of Google Accounts and service accounts. Every Google group has a unique email address that's associated with the group.Google Groups are a convenient way to apply access controls to a collection of users.Google Groups don't have login credentials, and you cannot use Google Groups to establish identity to make a request to access a resource.
A Google Workspace account represents a virtual group of all of the Google Accounts that it contains. Google Workspace accounts are associated with your organization's internet domain name, such as example.com.
A Cloud Identity domain is like a Google Workspace account, because it represents a virtual group of all Google Accounts in an organization. However, Cloud Identity domain users don't have access to Google Workspace applications and features.

Permissions determine what operations are allowed on a resource. Can grant IAM permissions at the project level. The permissions are then inherited by all resources within that project.You don't grant permissions to users directly. Instead, you identify *roles* that contain the appropriate permissions, and then grant those roles to the user.

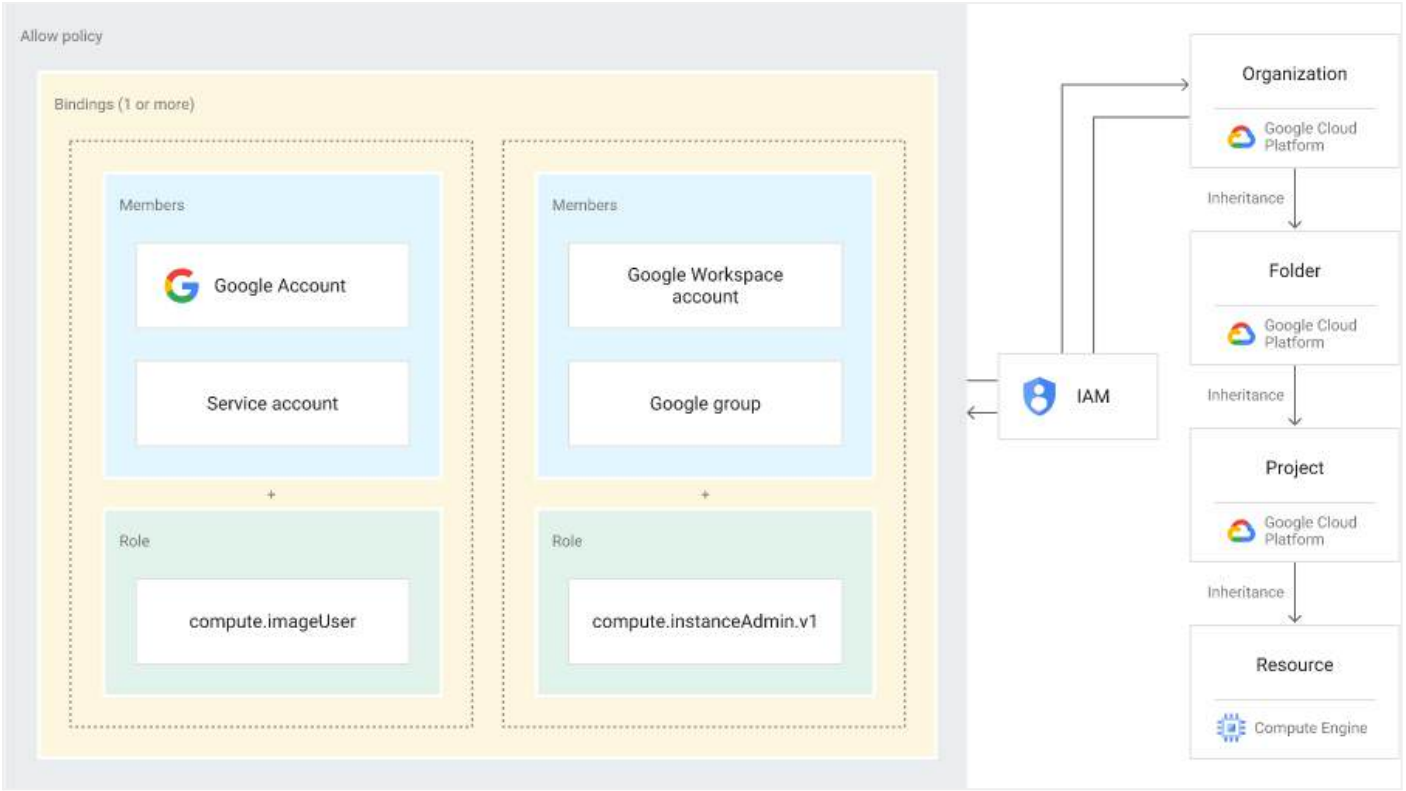


A Google account represents anyone who interacts with Google Cloud. When signing up for a Google account you will be asked to provide an email address that is associated with the account. The email does not have to come from the gmail domain.
A service account is how applications and resources authenticate and access services in Google Cloud. Since apps cannot sign in interactively with a username and password, service accounts use keys to authenticate.
Google groups are collections of identity principals that can be referenced by the email address assigned to the group. You can apply access policies to a group. Each member of the group will receive the permissions you specify in the group policy as they authenticate.
Google Workspace and Cloud Identity domains give you the ability to manage users based on the way your organization interacts with Google. Each method gives you a virtual group representing all the registered users in your organization and the ability to add, modify, and delete users and groups.

There are several kinds of roles in IAM:

- **Basic roles**: These roles are Owner, Editor, and Viewer.
- **Predefined roles**: Roles that give finer-grained access control than the basic roles. For example, the predefined role Pub/Sub Publisher (roles/pubsub.publisher) provides access to *only* publish messages to a Pub/Sub topic.
- **Custom roles**: Roles that you create to tailor permissions to the needs of your organization when predefined roles don't meet your needs.

You can grant roles to users by creating an *allow policy*, which is a collection of statements that define who has what type of access. An allow policy is attached to a resource and is used to enforce access control whenever that resource is accessed.



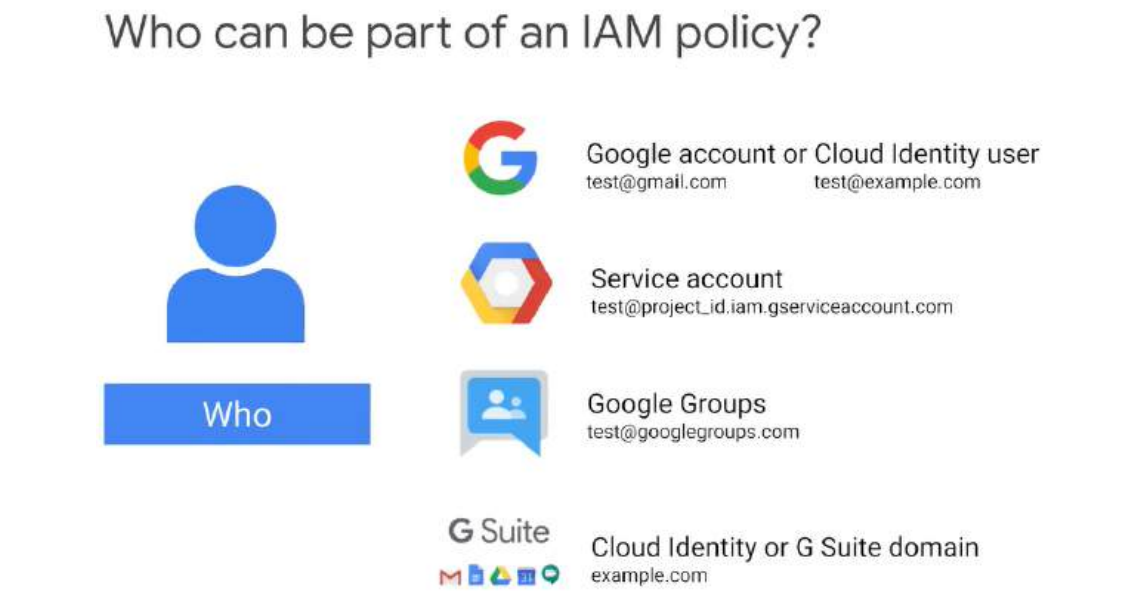
members: A list of one or more principals as described in the Concepts related to identity section in this document. Each principal type is identified with a prefix, such as a Google Account (user:), service account (serviceAccount:), Google group (group:), or a Google Workspace account or Cloud Identity domain (domain:).

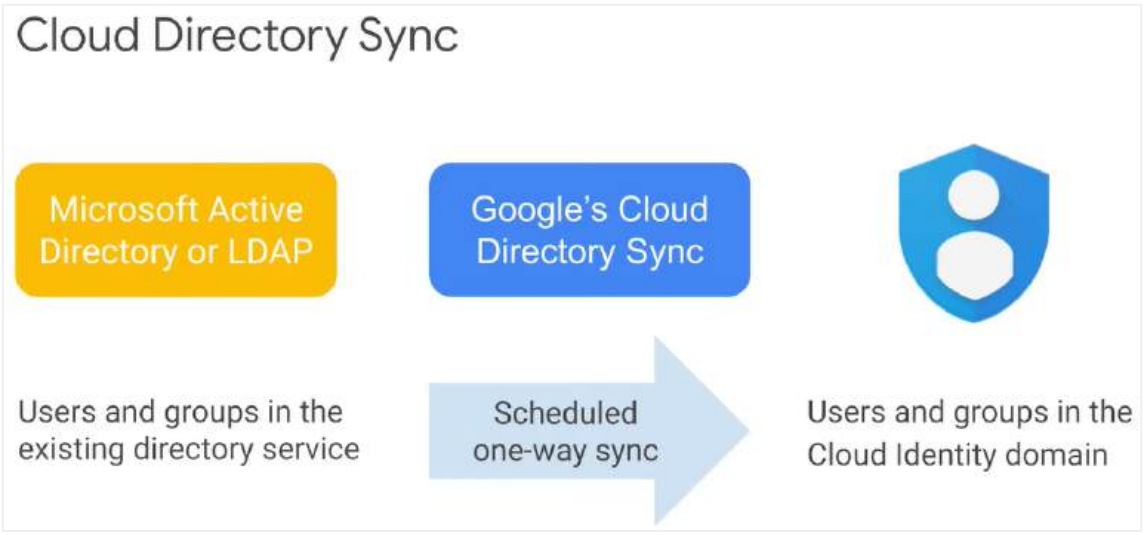
The IAM methods are:

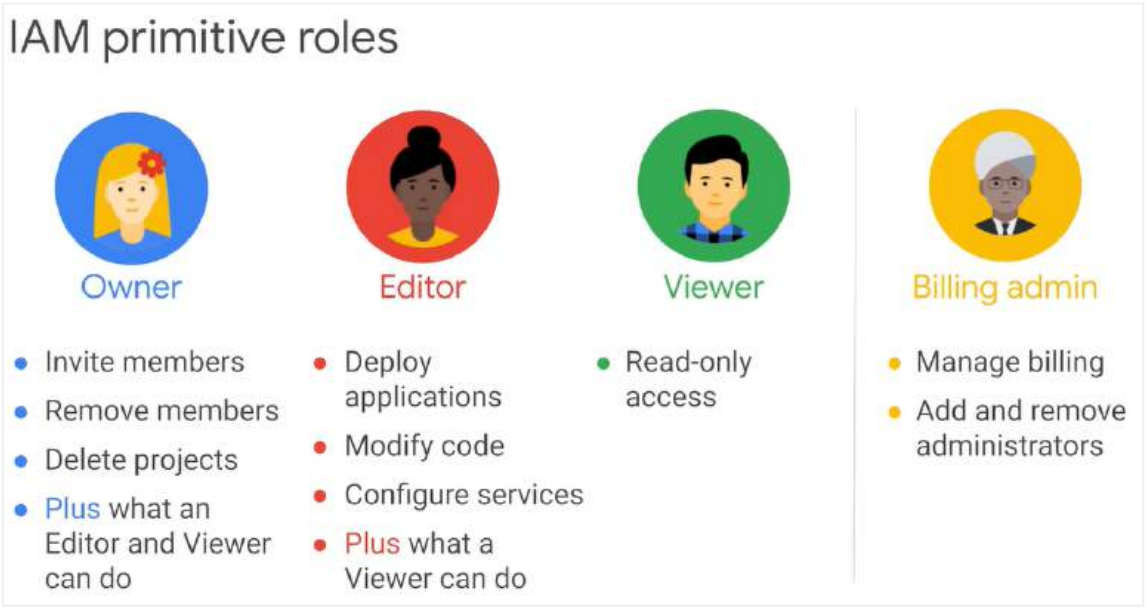
- setIamPolicy(): Sets allow policies on your resources.
- getIamPolicy(): Gets an allow policy that was previously set.
- testIamPermissions(): Tests whether the caller has the specified permissions for a resource.

These are the steps to assign roles in the IAM interface.

1. Go to the IAM page
2. Select project, folder, or organization
3. Show info panel if it is not available
4. Click permissions
5. Select or add a principal to add a role to
    a. If the principal already exists click on Add another role
    b. For a new principal click add and enter the principals email address
6. Select a role to grant
7. Add a condition
8. Click Save







b. Creating IAM policies

Modify the allow policy : Programmatically or using a text editor, modify the local copy of your resource's allow policy to reflect the roles that you want to grant or revoke.

To ensure that you do not overwrite other changes, do not edit or remove the allow policy's `etag` field. The `etag` field identifies the current state of the allow policy. When you set the updated allow policy, IAM

compares the etag value in the request with the existing etag, and only writes the allow policy if the values match.To grant roles to your principals, modify the role bindings in the allow **policy**.
Note: You can also use deny policies to prevent principals from using specific IAM permissions. For more information, see Deny policies.

To get the allow policy for resource	gcloud RESOURCE_TYPE get-iam-policy RESOURCE_ID --format=FORMAT > PATH
example	gcloud projects get-iam-policy my-project --format=json > ~/policy.json
list grantable roles	gcloud iam list-grantable-roles full-resource-name

c. Managing the various role types and defining custom IAM roles (e.g., primitive, predefined and custom)

There are three types of roles in IAM:

- **Basic roles**, which include the Owner, Editor, and Viewer roles that existed prior to the introduction of IAM. known as "primitive roles."
- **Predefined roles**, which provide granular access for a specific service and are managed by Google Cloud.
- **Custom roles**, which provide granular access according to a user-specified list of permissions.
- Run the gcloud iam roles describe command to list the permissions in the role.
- Call the roles.get() REST API method to list the permissions in the role.

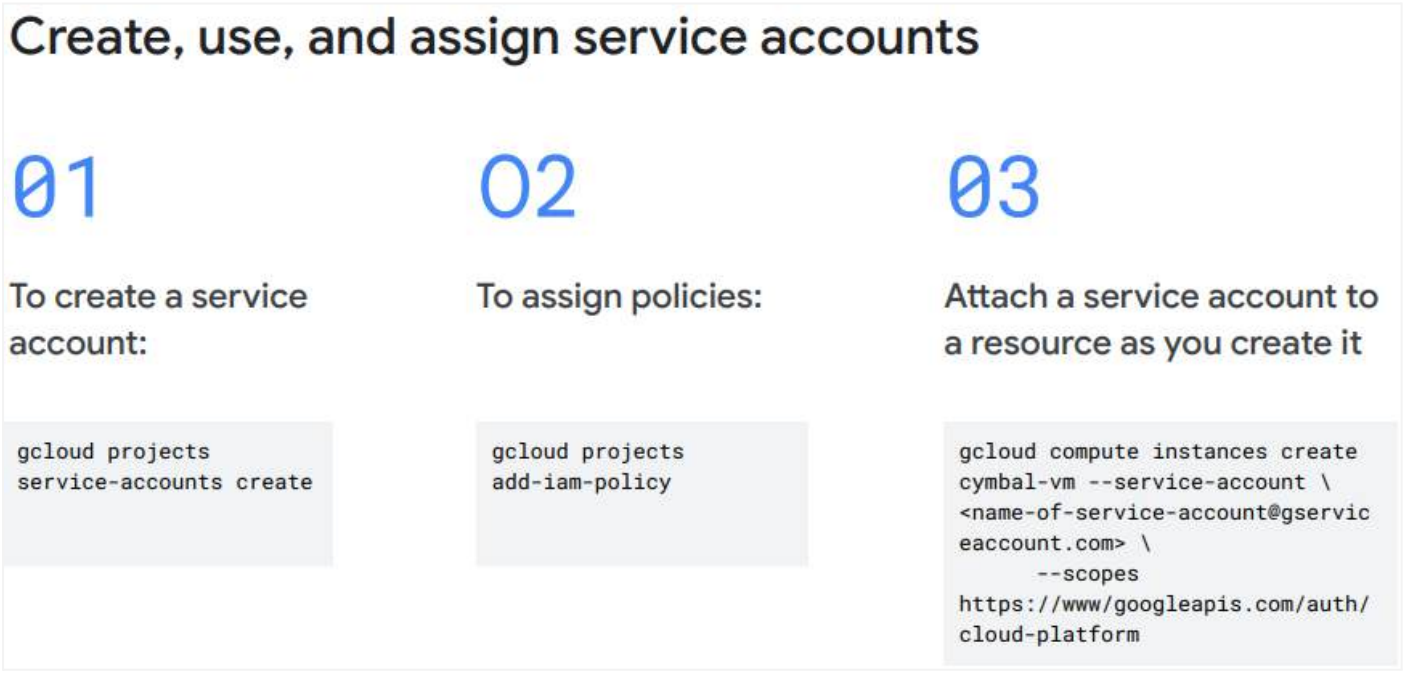
Note: You cannot define custom roles at the folder level. If you need to use a custom role within a folder, define the custom role at the organization level. Each custom role can contain up to 3,000 permissions. Also, the maximum total size of the title, description, and permission names for a custom role is 64 KB.
The first thing you need to do when creating custom permissions is be familiar with the permissions and roles that are available in your project or organization.
The gcloud command you need to run is : **gcloud iam list-testable-permissions <full-resource-name>**
To make sure there isn't already another role that will fill your needs, you can also look at the permissions assigned to a specific role by looking at the role metadata.The role metadata includes the role ID and the permissions associated with that role.
Custom roles can be created at the project or organizational level.
You need to have the Iam.roles.create permission. You have to be the owner of the group or project, or have an organization administrator role or the IAM Role Administrator role.
You can create roles from individual permissions, or you can select and pick permissions from predefined roles.
To update an existing role, you run roles.get(), update the role locally, and then run roles.patch().

5.2 Managing service accounts. Tasks include:

a. Creating service accounts

Service account A service account is an account for an application or compute workload instead of an individual end user. When you run code that's hosted on Google Cloud, the code runs as the account you

specify. You can create as many service accounts as needed to represent the different logical components of your application.
Each service account also has a permanent, unique numeric ID, which is generated automatically. The numeric ID is a 21-digit number, such as 123456789012345678901, that uniquely identifies the service account.



To create service account	gcloud iam service-accounts create SERVICE_ACCOUNT_ID \ --description="DESCRIPTION" \ --display-name="DISPLAY_NAME"
To grant your service account an IAM role on your project	gcloud projects add-iam-policy-binding PROJECT_ID \ --member="serviceAccount:SERVICE_ACCOUNT_ID@PROJECT_ID.iam.gserviceaccount.com" \ --role="ROLE_NAME"
To allow users to impersonate the service account	gcloud iam service-accounts add-iam-policy-binding \ SERVICE_ACCOUNT_ID@PROJECT_ID.iam.gserviceaccount.com \ --member="user:USER_EMAIL" \ --role="roles/iam.serviceAccountUser"
list service accounts	gcloud iam service-accounts list

b. Using service accounts in IAM policies with minimum permissions

To add a policy to a service account run the “gcloud projects add-iam-policy-binding” command. The “--member” argument should be a string starting with “serviceAccount:” and containing your service account id with an email address suffix of “@project_id.iam.gserviceaccount.com.” A “--role” argument contains the role you want to assign to the service account.

View current Access	gcloud iam service-accounts get-iam-policy SA_ID --format=FORMAT > PATH
Grant a single role to a principal	gcloud iam service-accounts add-iam-policy-binding SA_ID \ --member=PRINCIPAL --role=ROLE_ID \ --condition=CONDITION
Revoke a single role from a principal	gcloud iam service-accounts remove-iam-policy-binding SA_ID \ --member=PRINCIPAL --role=ROLE_ID

To get the allow policy	gcloud iam service-accounts get-iam-policy SA_ID --format=FORMAT > PATH
To set the allow policy	gcloud iam service-accounts set-iam-policy SA_ID PATH

- FORMAT: The desired format for the policy. Use json or yaml.
- PATH: The path to a new output file for the policy.

c. Assigning service accounts to resources

Resources in Google Cloud can be assigned a service account that acts as the resource's default identity. This process is known as attaching a service account to a resource. The resource, or apps running on the resource, impersonate the attached service account to access Google Cloud APIs.
Creating and enabling service accounts for instances using
Multiple virtual machine instances can use the same service account, but a virtual machine can only have one service account identity. Service account changes will affect all virtual machine instances using the service account. You can allow access via a cloud-platform scope that allows access to most cloud api's and then grant the service account the relevant IAM roles.
In gcloud you identify the service account you want to use by using the "--service-account" argument.

Two types of keys are available for authentication of a service account: user managed keys and Google managed keys. You create and manage user managed keys yourself. Google only stores the public key. With Google managed keys Google stores both the public and private portion of the keys. Google has APIs you can use to sign requests with the private key.

Attaching the service account to the new resource : After you configure the user-managed service account, you can create a new resource and attach the service account to that resource.
In most cases, you must attach a service account to a resource when you create that resource. After the resource is created, you cannot change which service account is attached to the resource. Compute Engine instances are an exception to this rule; you can change which service account is attached to an instance as needed.
Go to the Service Accounts page, select a project, check new service account and make note of the service account email.

Creating a service account	gcloud iam service-accounts create SERVICE_ACCOUNT_ID \ --description="DESCRIPTION" \ --display-name="DISPLAY_NAME"
service account's email derived from service account ID	[SERVICE-ACCOUNT-NAME]@[PROJECT_ID].iam.gserviceaccount.com
Setting up a new instance to run as a service account	gcloud compute instances create [INSTANCE_NAME] \ --service-account [SERVICE_ACCOUNT_EMAIL] \ --scopes [SCOPES,...]
example	gcloud compute instances create example-vm \ --service-account 123-my-sa@my-project-123.iam.gserviceaccount.com \ --scopes https://www.googleapis.com/auth/cloud-platform

Configure the service account for a resource in a different project.
Changing the service account and access scopes for an instance
Using OAuth 2.0 for Server to Server Applications

d. Managing IAM of a service account

Granting minimum permissions to service accounts : When granting permissions to users to access a service account, keep in mind that the user can access all the resources for which the service account has permissions. Therefore it's important to configure permissions of your service accounts carefully; that is, be strict about who on your team can act as (or impersonate) a service account.
Manage access to projects, folders, and organizations

e. Managing service account impersonation

Allow principals and resources to *impersonate*, or act as, an Identity and Access Management (IAM) service account.
several predefined roles that allow a principal to impersonate a service account:

- **Service Account User** (roles/iam.serviceAccountUser): Includes the permission iam.serviceAccounts.actAs, which allows principals to indirectly access all the resources that the service account can access. For example, if a principal has the Service Account User role on a service account, and the service account has the Cloud SQL Admin role (roles/cloudsql.admin) on the project, then the principal can impersonate the service account to create a Cloud SQL instance. This role also allows principals to attach a service account to a resource, as explained on this page. This role does *not* allow principals to create short-lived credentials for service accounts, or to use the --impersonate-service-account flag for the Google Cloud CLI. To complete these tasks, you also need the Service Account Token Creator role.
- **Service Account Token Creator** (roles/iam.serviceAccountTokenCreator): Allows principals to impersonate service accounts to create OAuth 2.0 access tokens, which you can use to authenticate with Google APIs. Also allows principals to sign JSON Web Tokens (JWTs) and sign binary blobs, so that they can be used for authentication. See Creating short-lived service account credentials for details.
- **Workload Identity User** (roles/iam.workloadIdentityUser): Allows principals to impersonate service accounts from GKE workloads.

Allowing a principal to impersonate multiple service accounts
Allowing a principal to impersonate a single service account
In a text editor, modify the bindings array in the **policy.json** file to grant the appropriate roles to your principals. To grant a role, do one of the following:

- If a binding for the role does not exist, add a new object to the bindings array that defines the role you want to grant and the principal you want to grant it to.
- If a binding already exists for the role, add the new principal to the list of existing principals. If your allow policy includes conditional role bindings, also ensure that the binding has the appropriate conditions before adding the principal.

To grant expirable access to a project resource
To grant access to a project resource for only certain days or hours of the week on a recurring basis:

f. Creating and managing short-lived service account credentials

Create short-lived credentials for a service account : Short-lived credentials have a limited lifetime, with durations of just a few hours or shorter. Short-lived service account credentials are useful for scenarios where you need to grant limited access to resources for trusted service accounts. They also create less risk than long-lived credentials, such as service account keys.
An OAuth 2.0 access token is useful for authenticating access from a service account to Google Cloud APIs.An OIDC ID token is useful for authenticating the identity of a service account to services that accept OpenID Connect.

When you create short-lived credentials for a service account using a direct request flow, the caller makes a direct request to create short-lived credentials. Two identities are involved in this flow: the caller, and the service account for which the credential is created.
To enable CALLER_ACCOUNT to create short-lived credentials for PRIV_SA, you grant CALLER_ACCOUNT the Service Account Token Creator role (roles/iam.serviceAccountTokenCreator) on PRIV_SA.

User account to generate short-lived tokens grants a role on a service account.	gcloud iam service-accounts add-iam-policy-binding PRIV_SA --member=user:CALLER_ACCOUNT --role=roles/iam.serviceAccountTokenCreator
calling application uses a service account as its identity	gcloud iam service-accounts add-iam-policy-binding PRIV_SA --member=serviceAccount:CALLER_SA --role=roles/iam.serviceAccountTokenCreator

- PRIV_SA: The email address of the privilege-bearing service account for which the token is generated.
- CALLER_ACCOUNT: The email address of the user account being used to request the short-lived token.
- Caller service account (CALLER_SA) represents the calling application, which issues the request for the short-lived credentials.
- Privilege-bearing service account (PRIV_SA) is granted the IAM roles needed for the short-lived token. This is the service account for which the short-lived token is created.

Credential types are supported:
- OAuth 2.0 access tokens
- OpenID Connect ID tokens
- Self-signed JSON Web Tokens (JWTs)
- Self-signed binary objects (blobs)

	gcloud auth login CALLER_ACCOUNT
generates an OAuth 2.0 access token for a service account.	gcloud auth print-access-token --impersonate-service-account=PRIV_SA

5.3 Viewing audit logs

Google Cloud services write audit logs to help you answer the questions, "Who did what, where, and when?" within your Google Cloud resources. Google Cloud services write audit logs that record administrative activities and accesses within your Google Cloud resources.
Cloud Audit Logs provides the following audit logs for each Cloud project, folder, and organization:
- Admin Activity audit logs : You can't disable Admin Activity audit logs.
- Data Access audit logs : Data Access audit logs are disabled by default and aren't written unless explicitly enabled (the exception is Data Access audit logs for BigQuery, which can't be disabled).
- System Event audit logs :
- Policy Denied audit logs

View logs : To query for audit logs, you need to know the audit log name, which includes the resource identifier of the Cloud project, folder, billing account, or organization for which you want to view audit logging information.In your query, you can further specify other indexed LogEntry fields, such as resource.type.

```
gcloud logging read "logName : projects/PROJECT_ID/logs/cloudaudit.googleapis.com" \
   --project=PROJECT_ID
```

```
gcloud logging read "logName : folders/FOLDER_ID/logs/cloudaudit.googleapis.com" \
```

```
  --folder=FOLDER_ID
```

```
gcloud logging read "logName : organizations/ORGANIZATION_ID/logs/cloudaudit.googleapis.com" \
  --organization=ORGANIZATION_ID
```

```
gcloud logging read "logName : billingAccounts/BILLING_ACCOUNT_ID/logs/cloudaudit.googleapis.com" \
  --billing-account=BILLING_ACCOUNT_ID
```

Go to the **Logging> Logs Explorer** page then Select an existing Cloud project, folder, or organization, In the **Query builder** pane, do the following:

- In **Resource type**, select the Google Cloud resource whose audit logs you want to see.
- In **Log name**, select the audit log type that you want to see:
  - For Admin Activity audit logs, select **activity**.
  - For Data Access audit logs, select **data_access**.
  - For System Event audit logs, select **system_event**.
  - For Policy Denied audit logs, select **policy**.

---

https(s) is an application protocol, so it lives at layer 7 of the TCP stack.
gcloud auth list
gcloud config list project
gcloud compute instances get-serial-port-output instance-1
gcloud compute reset-windows-password instance-1 --zone us-central1-a --user admin

Set up a Network and HTTP load balancer

Update gcloud CLI	gcloud components update
Set the default zone	gcloud config set compute/zone us-central1-a
Set the default region	gcloud config set compute/region us-central1

For this load balancing scenario, create three Compute Engine VM instances and install Apache	gcloud compute instances create www1 \
	  --image-family debian-9 \
	  --image-project debian-cloud \
	  --zone us-central1-a \
	  --tags network-lb-tag \
	  --metadata startup-script="#! /bin/bash
	  sudo apt-get update
	  sudo apt-get install apache2 -y
	  sudo service apache2 restart
	  echo '<!doctype html><html><body> www1 </body></html>' | tee /var/www/html/index.html"

	gcloud compute instances create www2 \
	  --image-family debian-9 \
	  --image-project debian-cloud \
	  --zone us-central1-a \
	  --tags network-lb-tag \
	  --metadata startup-script="#! /bin/bash
	  sudo apt-get update
	  sudo apt-get install apache2 -y
	  sudo service apache2 restart
	  echo '<!doctype html><html><body> www2 </body></html>' | tee

	/var/www/html/index.html"
	gcloud compute instances create www3 \ --image-family debian-9 \ --image-project debian-cloud \ --zone us-central1-a \ --tags network-lb-tag \ --metadata startup-script="#! /bin/bash sudo apt-get update sudo apt-get install apache2 -y sudo service apache2 restart echo '<!doctype html><html><body> www3 </body></html>' | tee /var/www/html/index.html"
Create a firewall rule to allow external traffic to the VM instances	gcloud compute firewall-rules create www-firewall-network-lb \ --target-tags network-lb-tag --allow tcp:80
Run cmd to list your instances	gcloud compute instances list
Verify that each instance is running with `curl`, replacing **[IP_ADDRESS]** with the IP address for each of your VMs	curl http://[IP_ADDRESS]
Create a static external IP address for your load balancer	gcloud compute addresses create network-lb-ip-1 \ --region us-central1
Add a legacy HTTP health check resource	gcloud compute http-health-checks create basic-check
Add a target pool in the same region as your instances	gcloud compute target-pools create www-pool \ --region us-central1 --http-health-check basic-check
Add the instances to the pool	gcloud compute target-pools add-instances www-pool \ --instances www1,www2,www3
Add a forwarding rule	gcloud compute forwarding-rules create www-rule \ --region us-central1 \ --ports 80 \ --address network-lb-ip-1 \ --target-pool www-pool
command to view the external IP address	gcloud compute forwarding-rules describe www-rule --region us-central1
Use `curl` command to access the external IP address	while true; do curl -m1 IP_ADDRESS; done

Create the load balancer template	gcloud compute instance-templates create lb-backend-template \ --region=us-central1 \ --network=default \ --subnet=default \ --tags=allow-health-check \ --image-family=debian-9 \ --image-project=debian-cloud \ --metadata=startup-script='#! /bin/bash apt-get update apt-get install apache2 -y a2ensite default-ssl a2enmod ssl vm_hostname="$(curl -H "Metadata-Flavor:Google" \

	http://169.254.169.254/computeMetadata/v1/instance/name)
	"
	  echo "Page served from: $vm_hostname" | \
	  tee /var/www/html/index.html
	  systemctl restart apache2'

Create a managed instance group based on the template	gcloud compute instance-groups managed create lb-backend-group \ --template=lb-backend-template --size=2 --zone=us-central1-a

Create the `fw-allow-health-check` firewall rule. This is an ingress rule that allows traffic from the Google Cloud health checking systems (`130.211.0.0/22` and `35.191.0.0/16`). This lab uses the target tag `allow-health-check` to identify the VMs.	gcloud compute firewall-rules create fw-allow-health-check \
	  --network=default \
	  --action=allow \
	  --direction=ingress \
	  --source-ranges=130.211.0.0/22,35.191.0.0/16 \
	  --target-tags=allow-health-check \
	  --rules=tcp:80

set up a global static external IP address that your customers use to reach your load balance	gcloud compute addresses create lb-ipv4-1 \
	  --ip-version=IPV4 \
	  --global

Note the IPv4 address that was reserved	gcloud compute addresses describe lb-ipv4-1 \
	  --format="get(address)" \
	  --global

Create a health check for the load balancer	gcloud compute health-checks create http http-basic-check \ --port 80

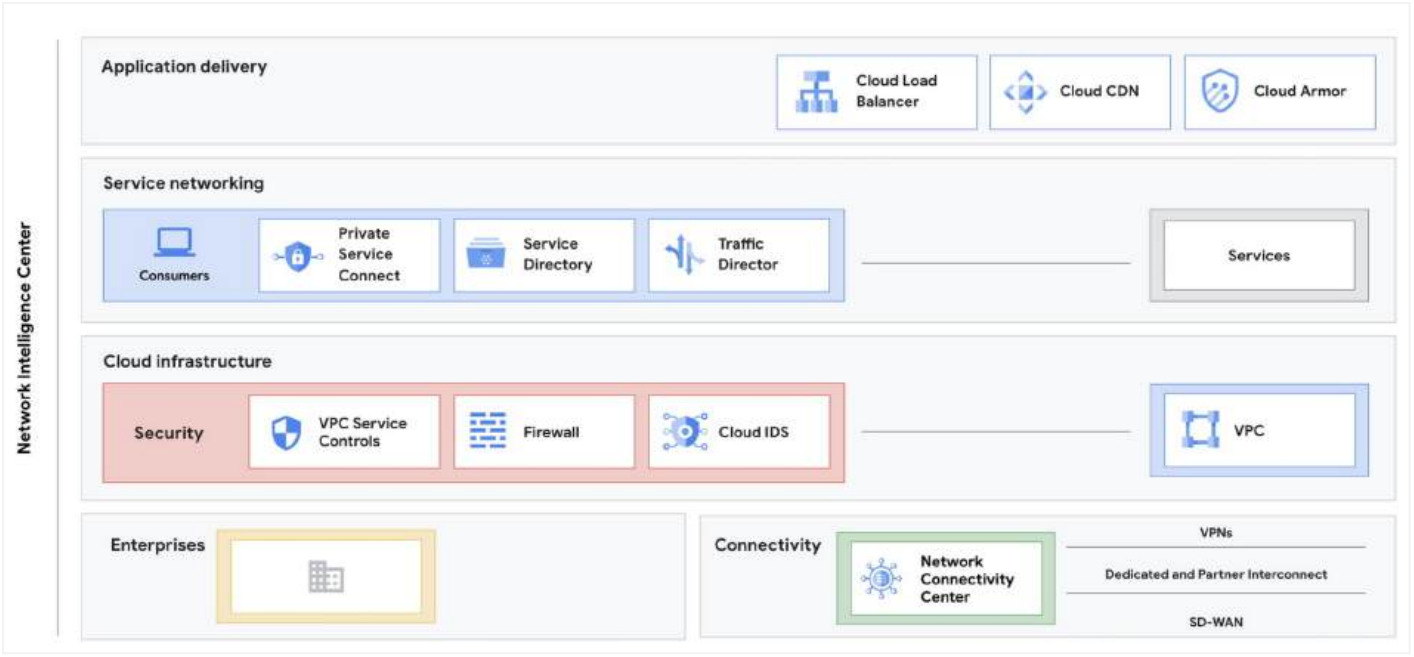
Create a backend service	gcloud compute backend-services create web-backend-service \
	  --protocol=HTTP \
	  --port-name=http \
	  --health-checks=http-basic-check \
	  --global

Add your instance group as the backend to the backend service	gcloud compute backend-services add-backend web-backend-service \
	  --instance-group=lb-backend-group \
	  --instance-group-zone=us-central1-a \
	  --global

Create a URL map to route the incoming requests to the default backend service	gcloud compute url-maps create web-map-http \ --default-service web-backend-service
Create a target HTTP proxy to route requests to your URL map	gcloud compute target-http-proxies create http-lb-proxy \ --url-map web-map-http

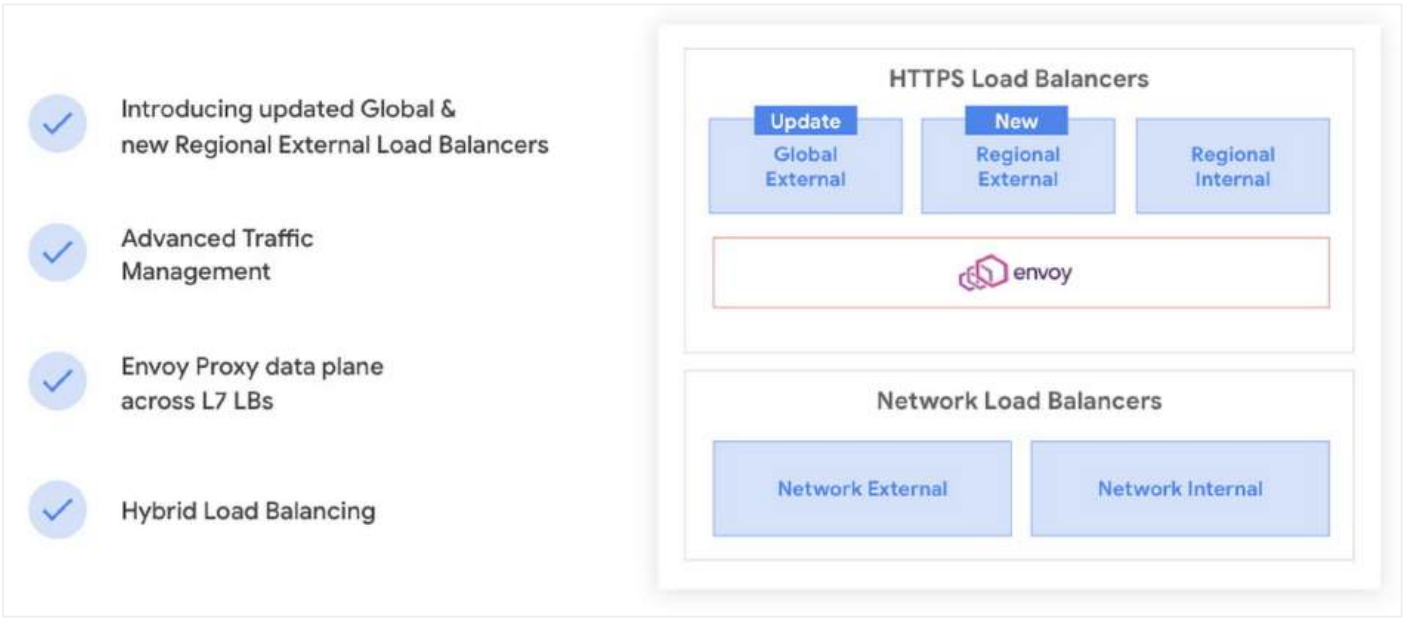
Create a global forwarding rule to route incoming requests to the proxy	gcloud compute forwarding-rules create http-content-rule \
	  --address=lb-ipv4-1\
	  --global \
	  --target-http-proxy=http-lb-proxy \
	  --ports=80

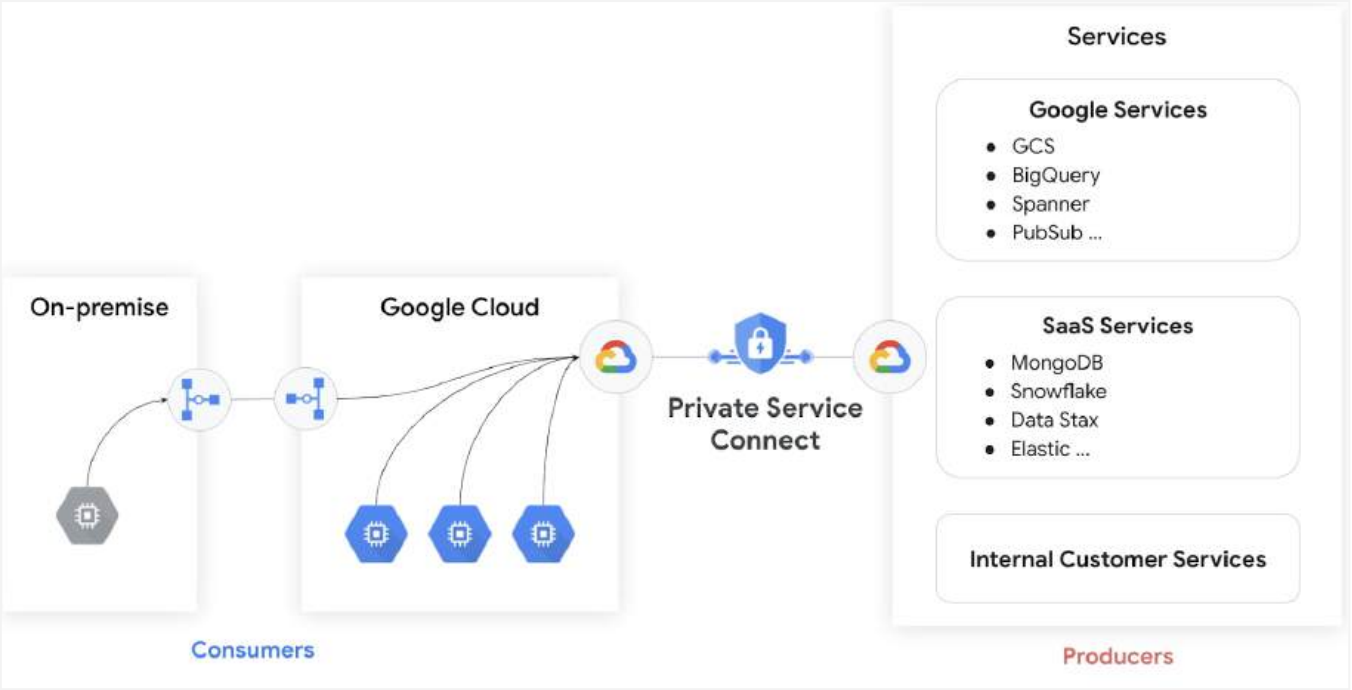
[The Google Cloud networking stack]



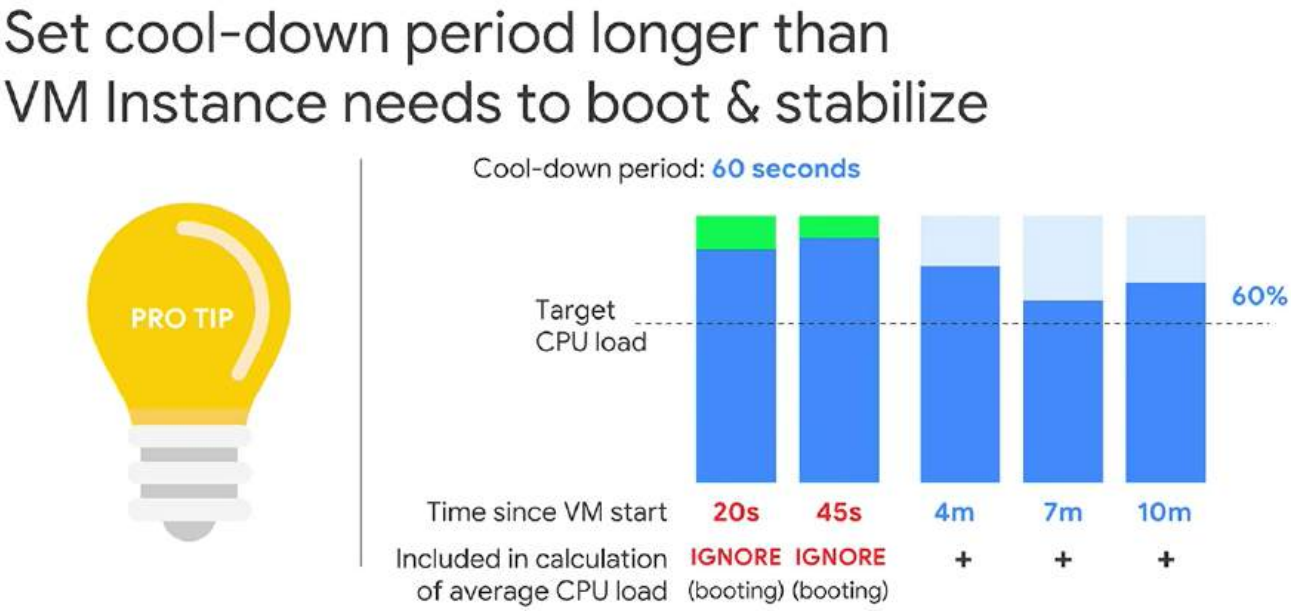
Together, these solutions are enable three primary Google Cloud Networking capabilities, including:

- Universal advanced traffic management with Cloud Load Balancing and the Envoy Proxy
- Connecting services across multicloud and on-premises hybrid network deployments
- Simplifying and securing service connectivity with Private Service Connect





[Autoscaling a cluster](#) gcloud container cluster create k1



Configuring an update

New template 50%

Max unavailable 2

Max surge 50%

Min wait time 120 sec

Action Recreate | Restart

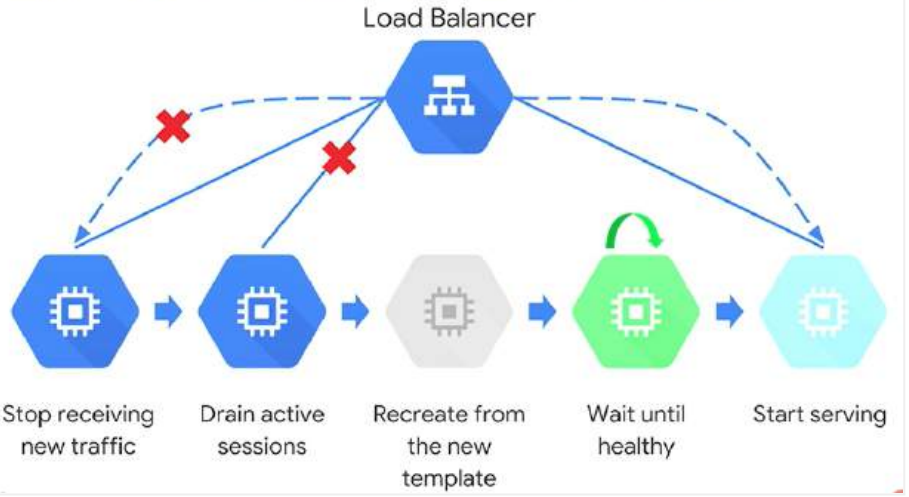
Mode Proactive | Opportunistic

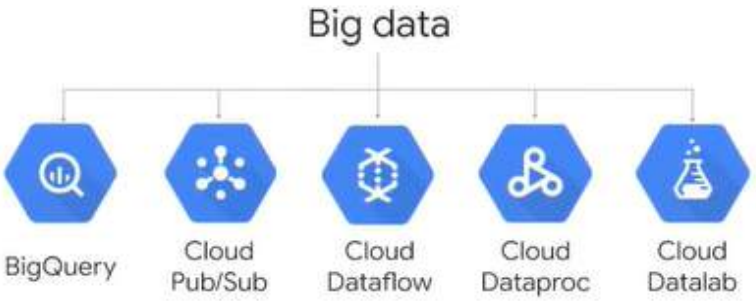
Updater & Load Balancer

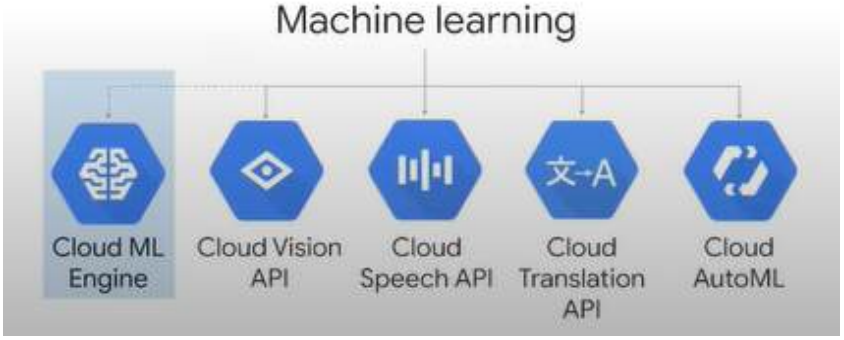
Worried about interrupting live user sessions?

Before an update, an instance:

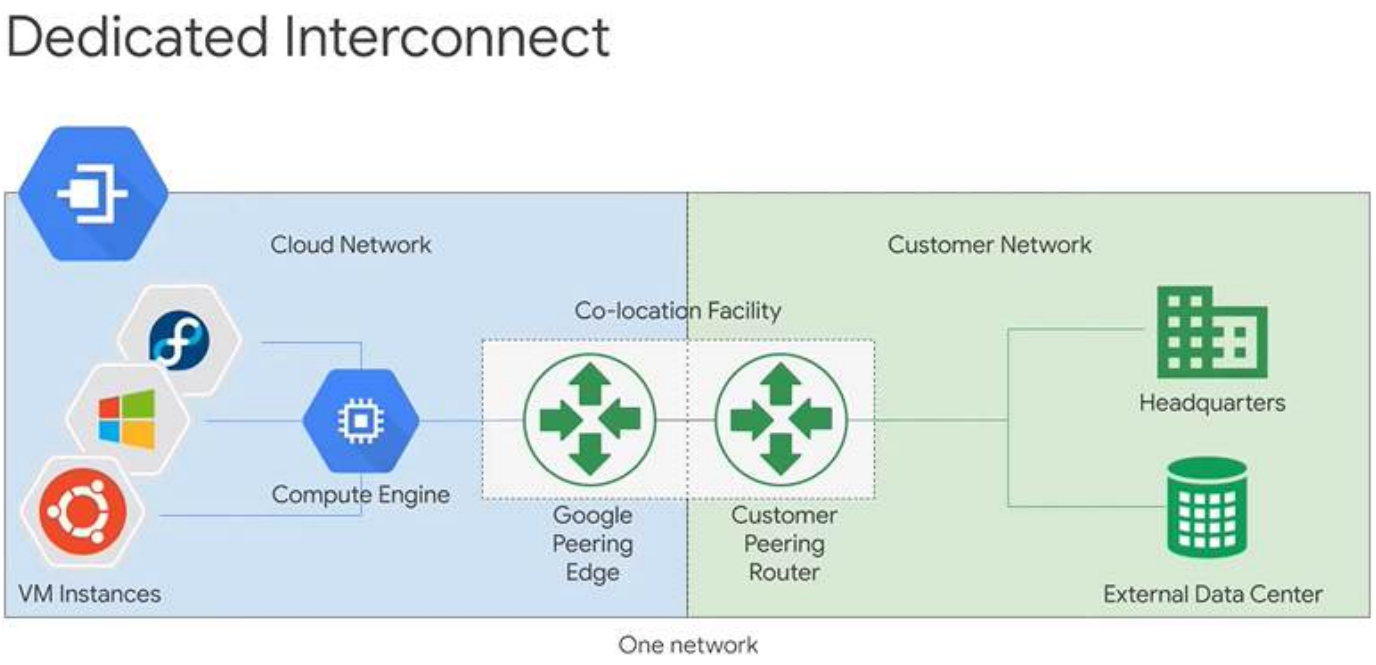
- stops receiving new traffic
- lets active sessions complete by observing connection draining timeout



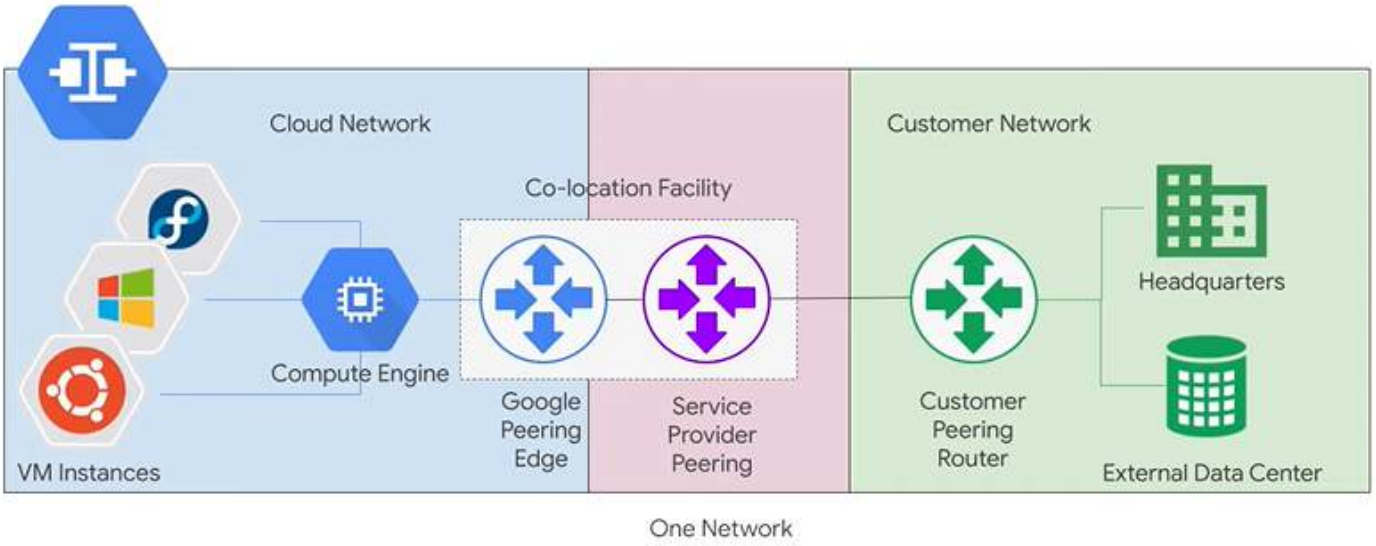




Google Cloud's Identity and Access Management (IAM) service lets you create and manage permissions for Google Cloud resources. Cloud IAM unifies access control for Google Cloud services into a single system and provides a consistent set of operations. Primitive roles set project-level permissions and unless otherwise specified, they control access and management to all Google Cloud services.



Partner Interconnect



One Network

A comparison of interconnect options

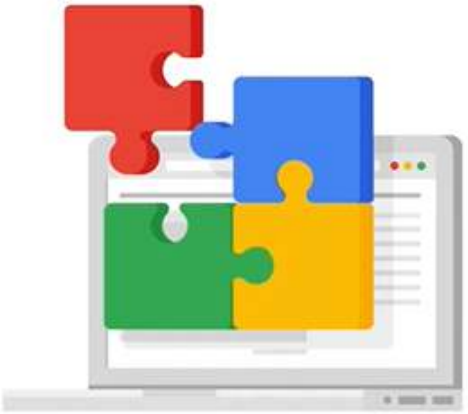
Connection	Provides	Capacity	Requirements	Access type
IPsec VPN tunnels	Encrypted tunnel to VPC networks through the public internet	1.5 - 3.0 Gbps per tunnel	On-premises VPN gateway	Internal IP addresses
Dedicated Interconnect	Dedicated, direct connection to VPC networks	10 Gbps per link	Connection in a colocation facility	Internal IP addresses
Partner Interconnect	Dedicated bandwidth, connection to VPC network through a service provider	50 Mbps – 10 Gbps per connection	Service provider	Internal IP addresses

Direct Peering provides a direct connection between a business network and Google's

- Leverage Google's broad-reaching edge network locations.
- Exchange BGP routes between Google and the peering entity.
- Reach all of Google's services.
- No SLA applies.
- Peering requirements must be satisfied.

Carrier Peering provides connectivity through a supported partner

- Leverage a provider's enterprise-grade network services to access Google applications.
- Get connections with higher availability and lower latency.
- No SLA offered by Google but may be offered by the provider.



A comparison of peering options

Connection	Provides	Capacity	Requirements	Access type
Direct Peering	Dedicated, direct connection to Google's network	10 Gbps per link	Connection in GCP PoPs	Public IP addresses
Carrier Peering	Peering through a service provider to Google's public network	Varies based on partner offering	Service provider	Public IP addresses

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

With VPC Peering, a relationship between two VPCs can be established to exchange traffic.
Alternatively, to use the full power of Identity Access Management (IAM) to control who and what in one project can interact with a VPC in another, you can configure a Shared VPC.

Define your required infrastructure as code

- Infrastructure requirements are defined as code in a template that's human-readable and machine consumable.
- Use templates to automate building, modifying, and deleting infrastructure.
- Templates can be stored, versioned, and shared.
- Templates can be used to rebuild an infrastructure after a crash.

Comparing the App Engine environments

	Standard Environment	Flexible Environment
Instance startup	Milliseconds	Minutes
SSH access	No	Yes (although not by default)
Write to local disk	Depends on language	Yes (but writes are ephemeral)
Support for 3rd-party binaries	No	Yes
Network access	Via App Engine services	Yes
Pricing model	After free daily use, pay per instance class, with automatic shutdown	Pay for resource allocation per hour; no automatic shutdown

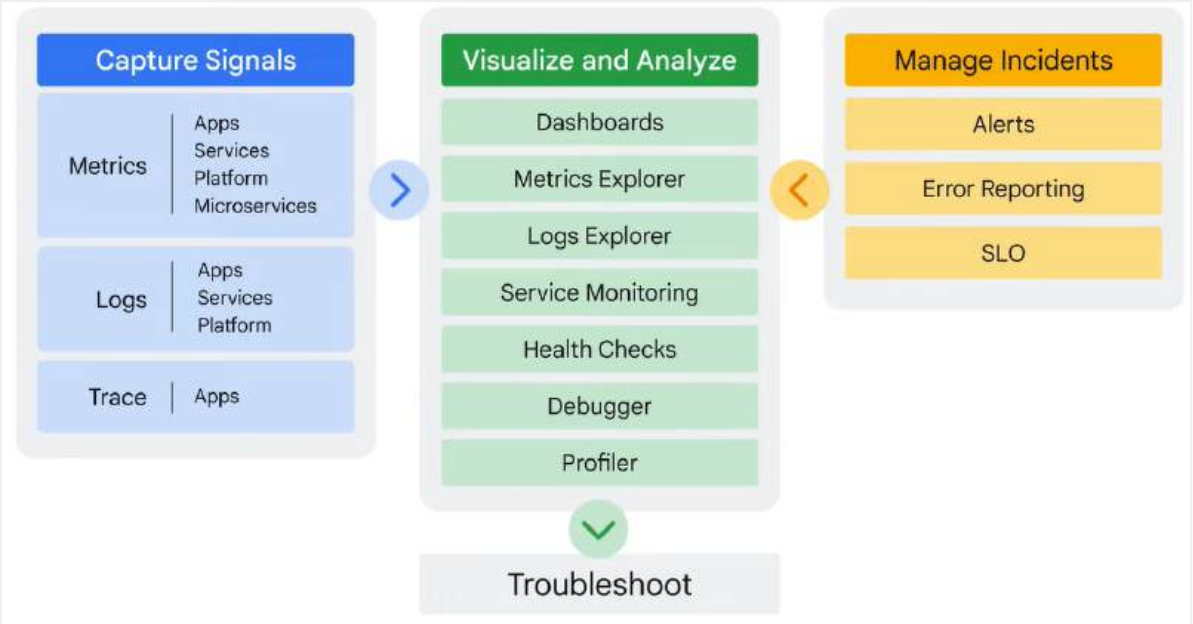
Command -->	**Action**	**.**	**Command -->**	**Action**
mkdir (*make directory*)	create a new folder	.	**cd** (*change directory*)	change location to another folder
ls (*list*)	list files and folders in the directory	.	**cat** (*concatenate*)	read contents of a file without using an editor
apt-get update	update package manager library	.	**ping**	signal to test reachability of a host
mv (*move*)	moves a file	.	**cp** (*copy*)	makes a file copy

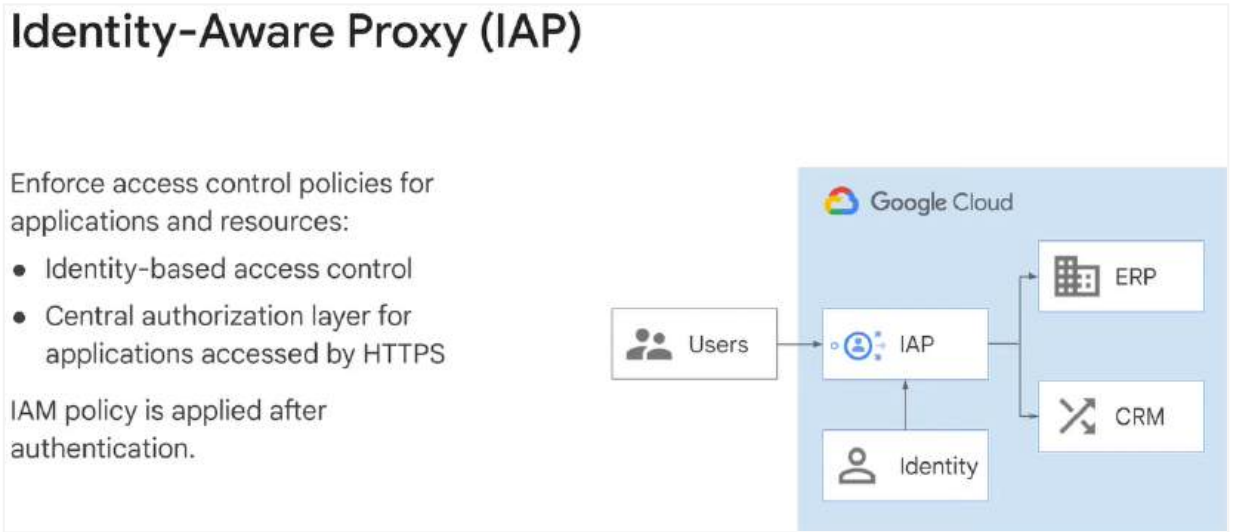
pwd (*present working directory*)	returns your current location	.	**sudo** (*super user do*)	gives higher administration privileges

Cloud Build is a service that executes your builds on GCP. It executes a series of build steps, where each build step is run in a Docker container to produce your application container (or other artifacts) and push it to Cloud Registry, all in one command.

Monitoring:
To improve service reliability, all parts of the business must agree that these are an accurate measure of user experience and must agree to use them as a primary driver for decision making

Observability Tools





Accessing Compute Engine as a different SSH user : You can specify a different user using the --ssh-key-file PRIVATE_KEY_FILE flag when running the gcloud compute ssh command.

To prevent idle connections, do the following:

- Set operating system TCP keep-alive parameters to a time frame of *less than* 10 minutes. This ensures that at least one packet is sent within the time frame.
- Ensure applications that open TCP connections do so with the SO_KEEPALIVE option enabled.

Troubleshooting using the serial console : A virtual machine instance has four virtual serial ports. Typically, these system-level entities use the first serial port (port 1) and serial port 1 is often referred to as the serial console.

To be prepared for the extreme case where one zone fails or an entire group of instances stops responding, Google strongly recommends overprovisioning your MIG.

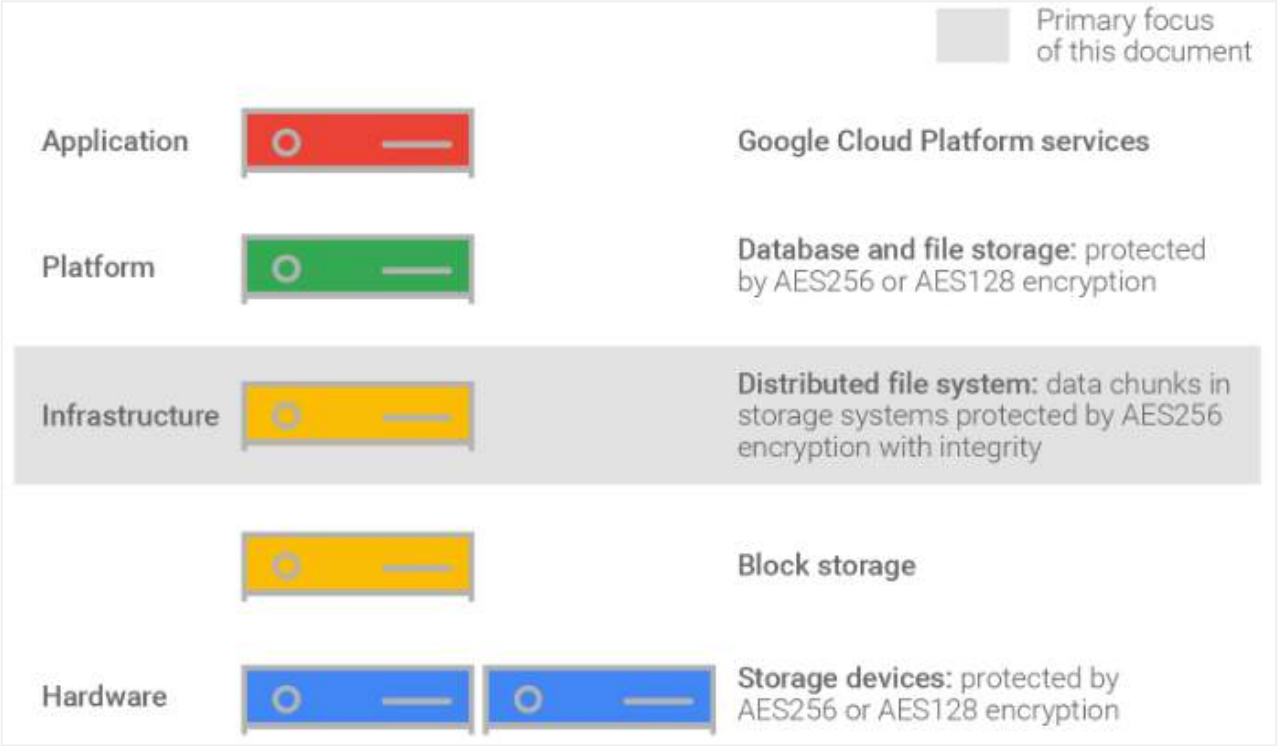
Note: The following recommendations for group size assume that your regional MIG uses proactive instance redistribution, which is enabled by default.

If you disable proactive instance redistribution, the instances in your group might not be evenly distributed across zones. In case of a zonal failure, your group might not have sufficient capacity in remaining zones to handle failover traffic.

Use the following table to determine the minimum recommended size for your group:

Number of zones	Additional VM instances	Recommended total VM instances
2	+100%	200%
3	+50%	150%
4	+33%	133%

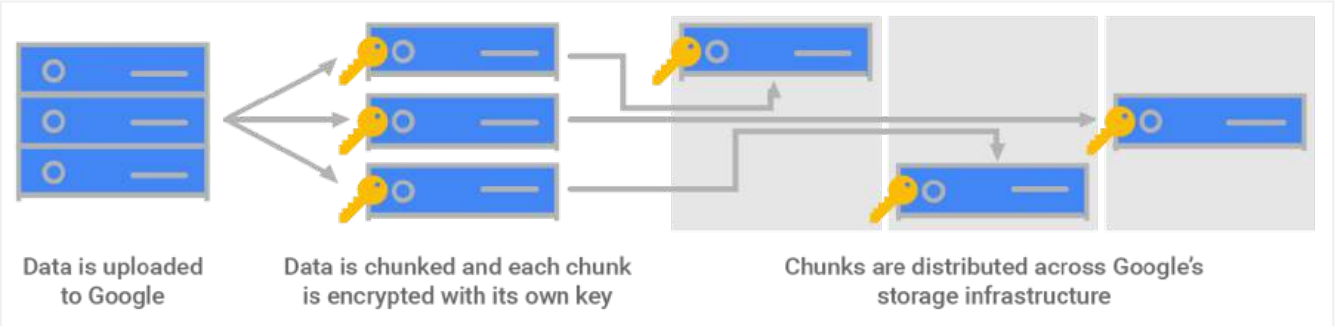
Using License Mobility with Microsoft server applications : Compute Engine lets you import Microsoft images, apply existing application licenses to the imported images, and then deploy Microsoft server applications on Google Cloud without paying additional licensing fees to Microsoft for those applications.



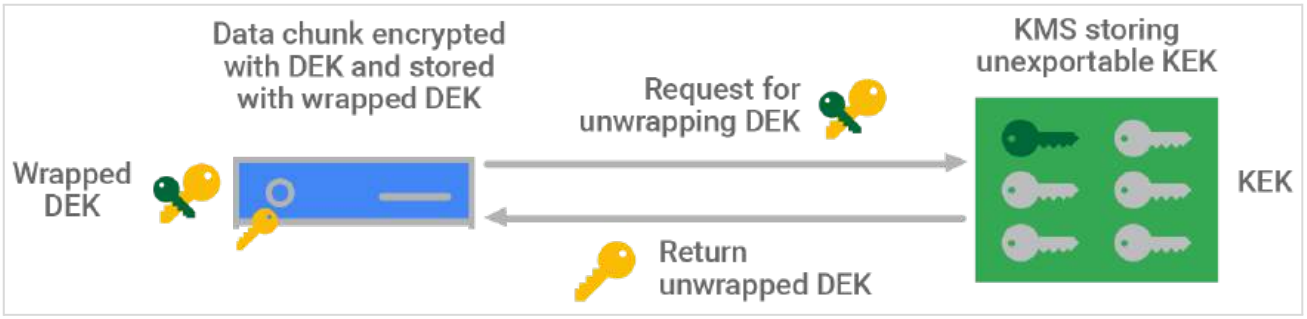
Encryption at rest in Google Cloud : By *encryption at rest*, we mean encryption used to protect data that is stored on a disk (including solid-state drives) or backup media.

- Google Cloud encrypts all customer content stored at rest, without any action required from the customer, using one or more encryption mechanisms.
- Data for storage is split into chunks, and each chunk is encrypted with a unique data encryption key. These data encryption keys are stored with the data, encrypted with ("wrapped" by) key encryption keys that are exclusively stored and used inside Google's central Key Management Service. Google's Key Management Service is redundant and globally distributed.

- All data stored in Google Cloud is encrypted at the storage level using (Advanced Encryption Standard)AES256, with the exception of a small number of Persistent Disks created before 2015 that use AES128.



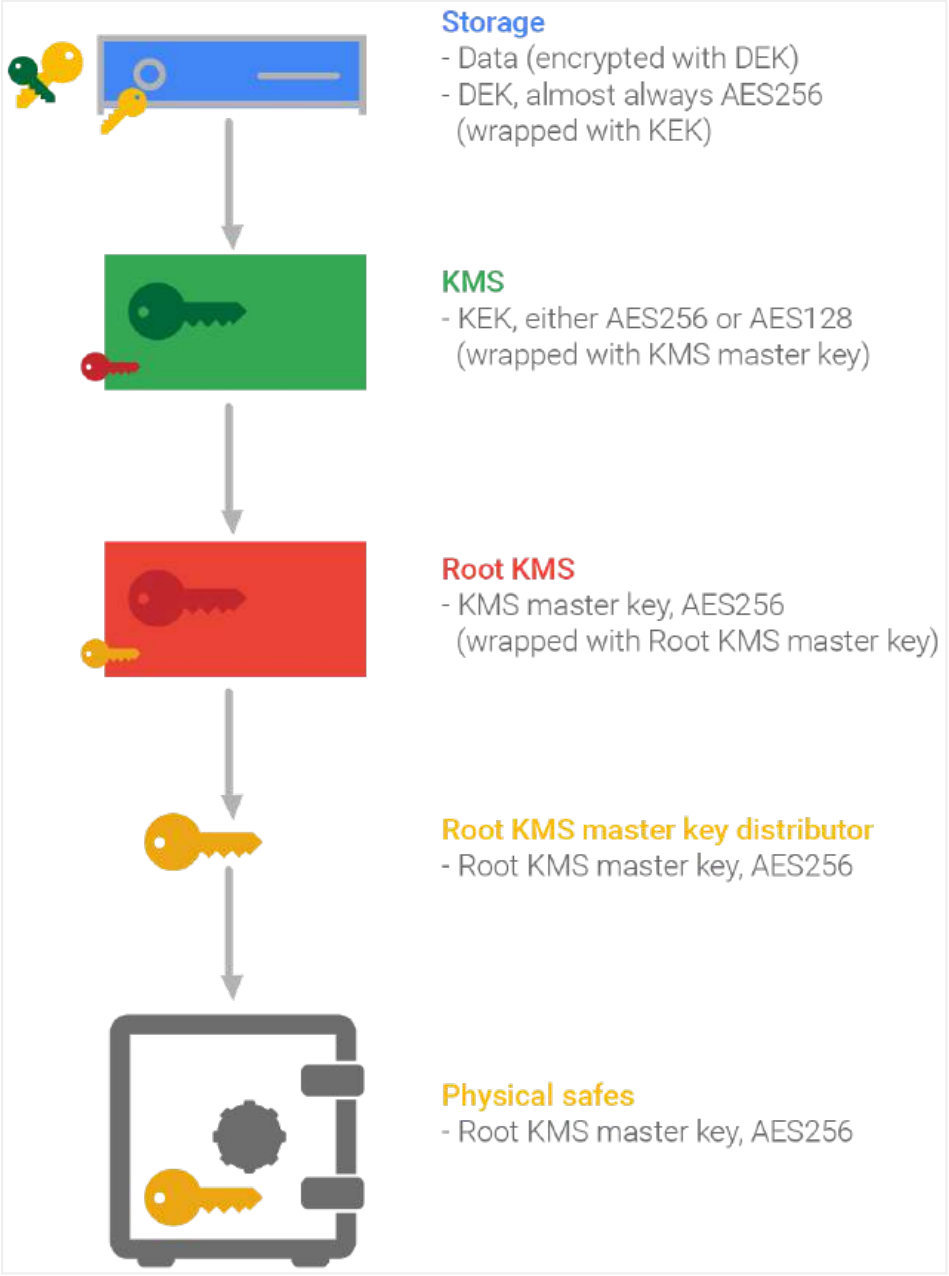
The key used to encrypt the data in a chunk is called a *data encryption key (DEK)*.The DEKs are encrypted with (or "wrapped" by) a *key encryption key (KEK)*. Cloud Storage specifically rotates its KEKs every 90 days, and can store up to 20 versions, requiring re-encryption of data at least once every five years (though in practice, data re-encryption is much more frequent). KMS-held keys are backed up for disaster recovery purposes, and they are indefinitely recoverable.



To decrypt a data chunk, the storage service calls Google's Key Management Service (KMS) to retrieve the unwrapped data encryption key (DEK) for that data chunk.
Google's KMS is protected by a root key called the *KMS master key*, which wraps all the KEKs in KMS.

To summarize:

- Data is chunked and encrypted with DEKs.
- DEKs are encrypted with KEKs.
- KEKs are stored in KMS.
- KMS is run on multiple machines in data centers globally.
    - KMS keys are wrapped with the KMS master key, which is stored in Root KMS.
- Root KMS is much smaller than KMS and runs only on dedicated machines in each data center.
    - Root KMS keys are wrapped with the root KMS master key, which is stored in the root KMS master key distributor.
- The root KMS master key distributor is a peer-to-peer infrastructure running concurrently in RAM globally on dedicated machines; each gets its key material from other running instances.
    - If all instances of the distributor were to go down (total shutdown), a master key is stored in (different) secure hardware in (physical) safes in limited Google locations.
    - The root KMS master key distributor is currently being phased in, to replace a system that operated in a similar manner but was not peer to peer.



Customers can use existing encryption keys that they manage with the Google Cloud, using the customer-supplied encryption keys feature. This feature is available for Cloud Storage and for Compute Engine for storage layer encryption.
Customers can also manage their own encryption keys on Google Cloud using Cloud Key Management Service. This product allows for application-layer encryption managed by the customer.

For Denial of Service (DoS) attack, Google Cloud Platform provides several of these mechanisms automatically and you can follow the best practices detailed below on your end to help secure your GCP deployment:

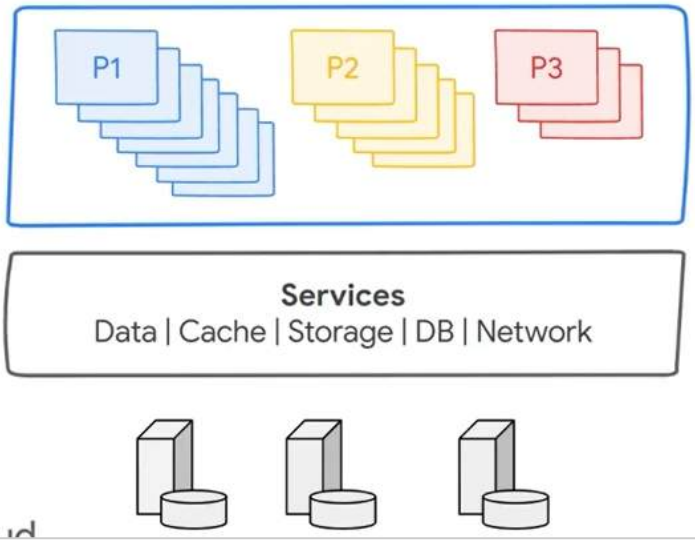
1. Reduce the attack surface for your GCE deployment
    a. GCP provides anti-spoofing protection for the private network (IP addresses) by default. GCP automatically provides isolation between virtual networks.
2. Isolate your internal traffic from the external world
    a. Set up a NAT gateway or SSH bastion to limit the number of instances that are exposed to the internet.
3. DDoS Protection by enabling Proxy-based Load Balancing

    a. enable HTTP(S) Load Balancing or SSL proxy Load Balancing, Google infrastructure mitigates and absorbs many Layer 4 and below attacks, such as SYN floods, IP fragment floods, port exhaustion, etc.
4. Scale to absorb the attack
    a. Protection by Google Frontend infrastructure
    b. Anycast-based Load Balancing
    c. Autoscaling
5. Protection with CDN Offloading
6. Deploy third-party DDoS protection solutions
7. App Engine deployment
    a. App Engine is designed to be a fully multi-tenant system and implements a number of safeguards intended to ensure that a single bad application will not impact the performance or availability of other applications on the platform.
    b. App Engine sits behind the Google Front End which mitigates and absorbs many Layer 4 and below attacks, such as SYN floods, IP fragment floods, port exhaustion, etc.
    c. You can also specify a set of IPs/IP networks via a dos.yaml file to block them from accessing your application(s).
8. Google Cloud Storage
    a. If you do not want to require your users to have a Google account in order to be able to access your Google Cloud Storage resources, you can control access using signed URLs.
9. API rate-limiting
    a. API rate limits define the number of requests that can be made to the Google Compute Engine API. API rate limits apply on a per-project basis.
    b. Currently, projects are limited to an API rate limit of 20 requests/second.
10. Resource Quotas

To deploy Autoscaler, decide which of the following topologies is best to fulfill your technical and operational needs:

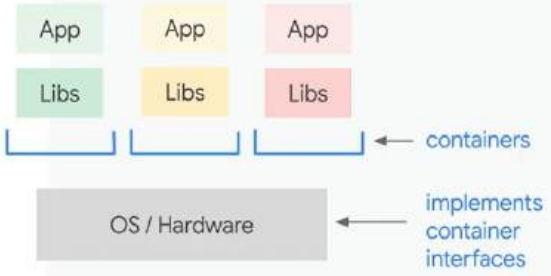
a) Per-project topology: The Autoscaler infrastructure is deployed in the same project as Cloud Spanner that needs to be autoscaled.
b) Centralized topology: Autoscaler is deployed in one project and manages one or more Cloud Spanner instances in different projects.
c) Distributed topology : Most of the Autoscaler infrastructure is deployed in one project, but some infrastructure components are deployed with the Cloud Spanner instances being autoscaled in different projects.

As demand for your app increases, the platform scales your app rapidly and independently by workload and infrastructure



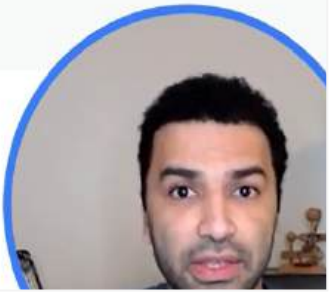
Google Cloud

Containers offer IaaS flexibility and PaaS scalability



Containers provide:

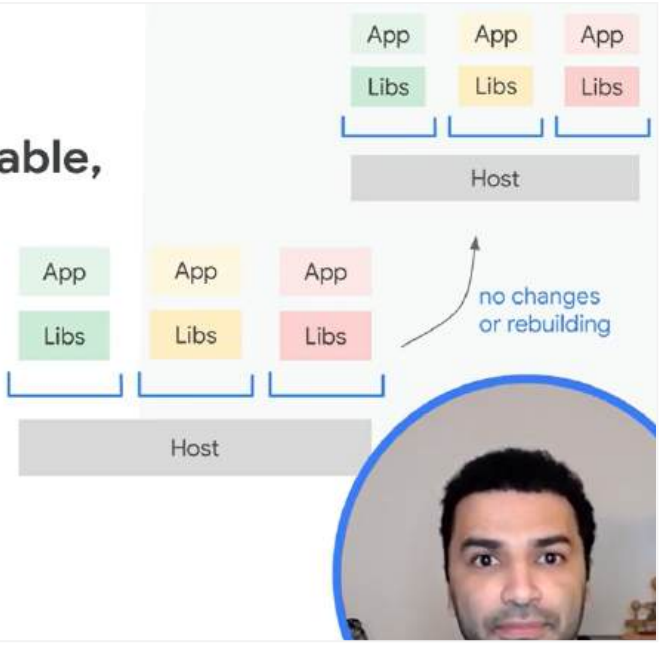
- An abstraction layer of the hardware and OS
- An invisible box with configurable access to isolated partitions of file system, RAM, and networking
- Fast startup (only a few system calls)



Containers are configurable, self-contained, and ultra-portable

With containers, you can:

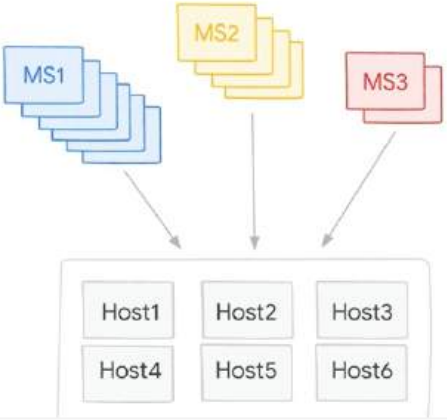
- Define your own hardware, OS, and software stack configurations
- Treat the OS and hardware as a black box and go from dev, to staging, to production, or your laptop to the cloud, without changing or rebuilding anything



With a common host configuration, you can deploy containers on a group of servers called a cluster

With a cluster, you can:

- Connect containers using network connections
- Build code modularly
- Deploy it easily
- Scale containers and hosts independently for maximum efficiency and savings



How do you get this app into Kubernetes?

You use a Dockerfile to specify such things as:

- A requirements.txt file for Flask dependencies
- Your OS image and version of Python
- How to install Python
- How to run your app

requirements.txt

```
Flask==0.12
uwsgi==2.0.15
```

Dockerfile

```
FROM ubuntu:18.10
RUN apt-get update -y && \
    apt-get install -y python3-pip python3-dev
COPY requirements.txt /app/requirements.txt
WORKDIR /app
RUN pip3 install -r requirements.txt
COPY . /app
ENDPOINT ["python3", "app.py"]
```

Google Cloud

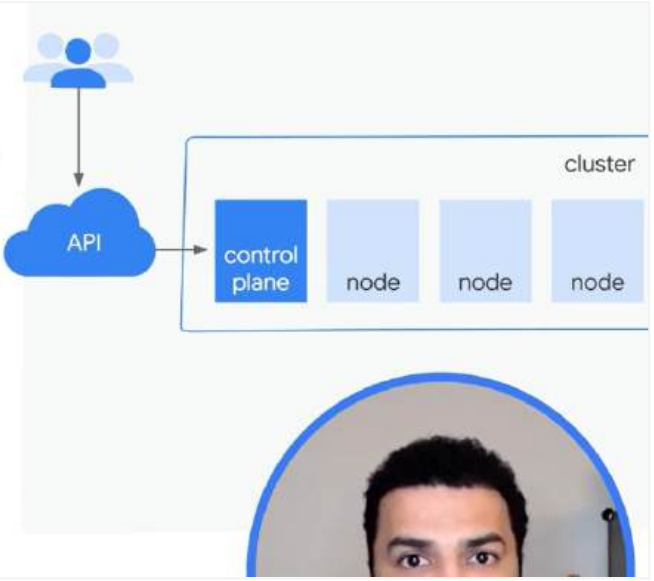
Then you build and run the container as an image

- **`docker build`** builds a container and stores it locally as a runnable image
- You can upload images to a registry service (like Google Container Registry) for sharing
- **`docker run`** starts the container image

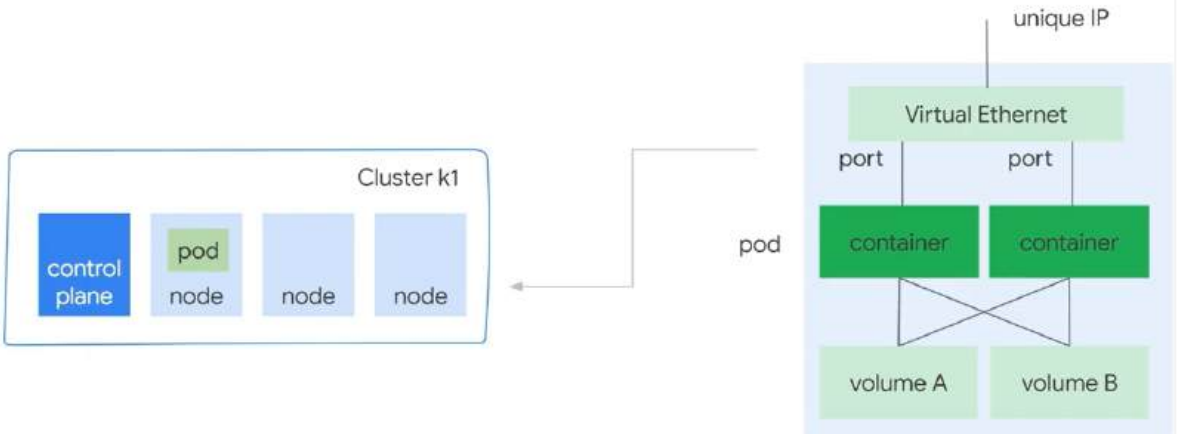
```
$> docker build -t py-server .
$> docker run -d py-server
```

You use Kubernetes APIs to deploy containers on a set of nodes called a cluster

- Masters run the control plane
- Nodes run containers
- Nodes are VMs (in GKE they're GCE instances)
- You describe the apps, Kubernetes figures out how to make that happen

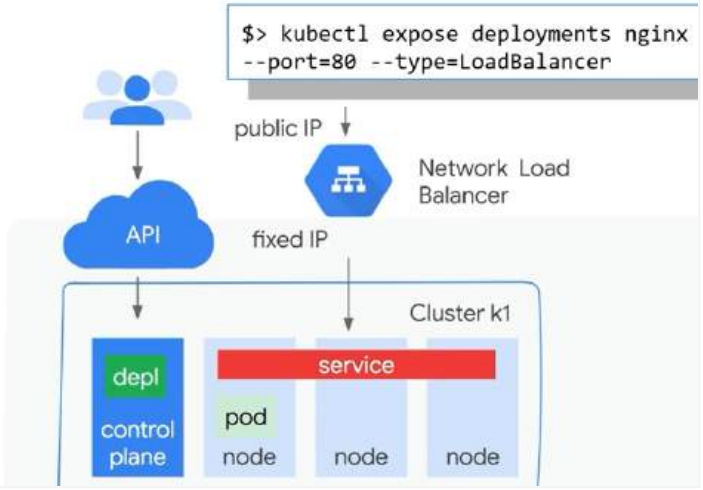


When you deploy containers on nodes you use a wrapper called a Pod



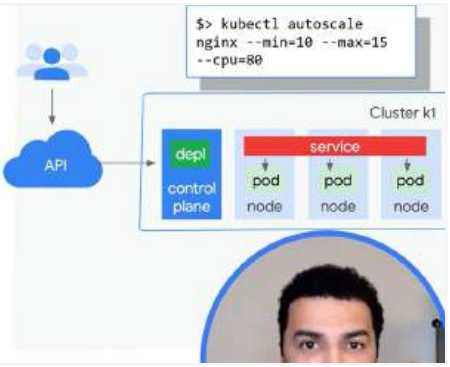
By default, Pods are only available inside a cluster and they get ephemeral IPs

- To make them publicly available with a fixed IP, you can connect a load balancer to your Deployment running **`kubectl expose`**
- Kubernetes creates a Service with a fixed IP for your Pods, and a controller says "I need to attach an external load balancer with a public IP address"

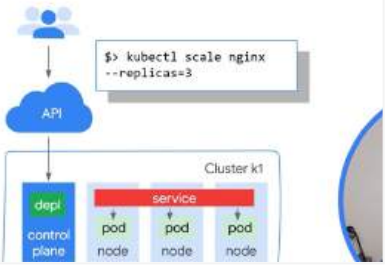


Google Cloud

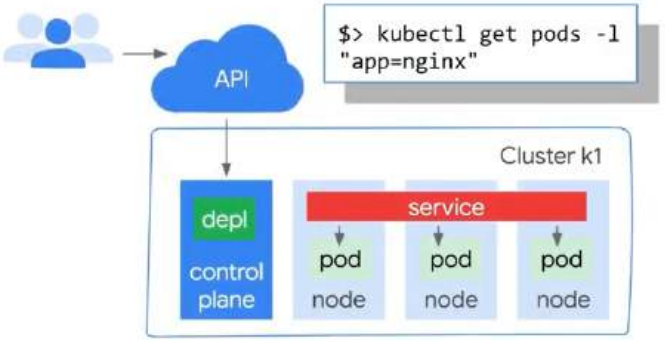
You can also run autoscaling with all kinds of parameters or put it behind programming logic for intelligent management



To scale a Deployment, run `kubectl scale`

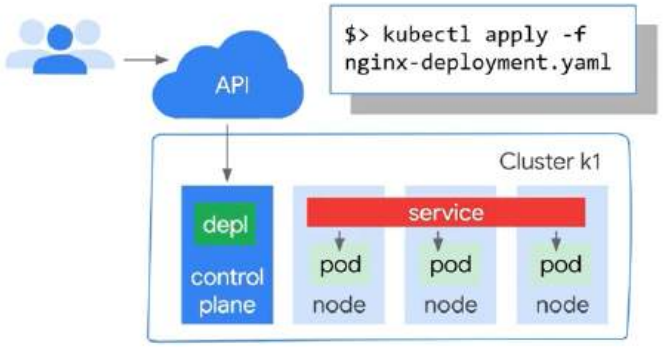


The real strength of Kubernetes comes when you work in a declarative way (here's how to get a config file)




```yaml
apiVersion: v1
kind: Deployment
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.15.7
        ports:
        - containerPort: 80
```

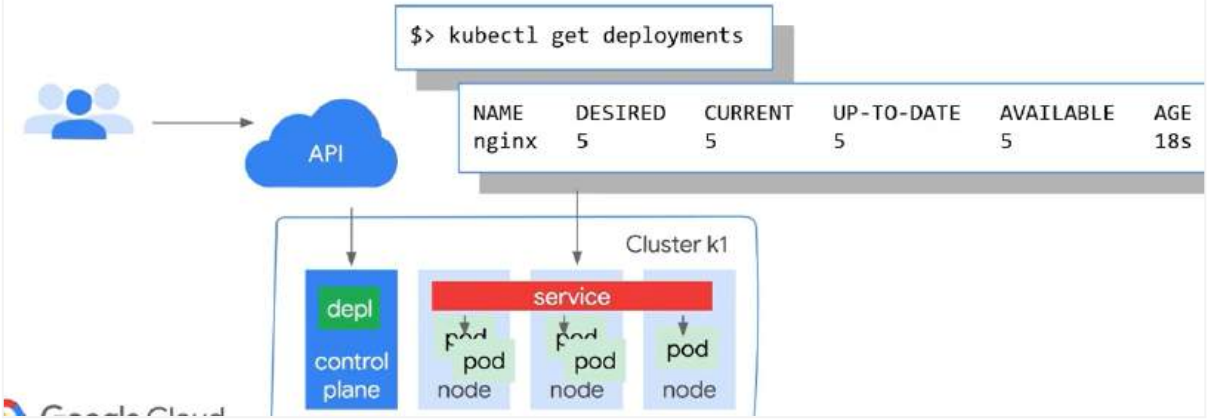
To declaratively apply changes run `kubectl apply -f`

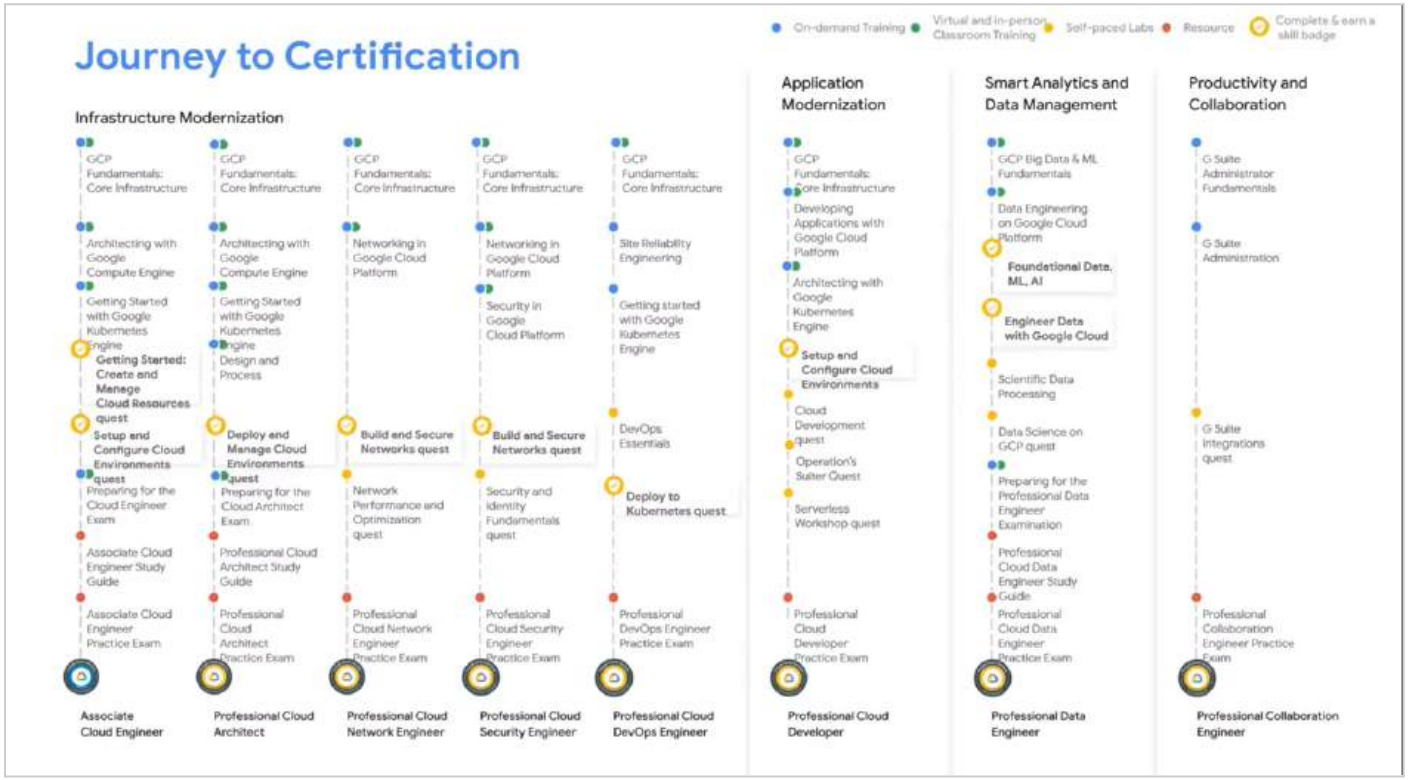



```yaml
apiVersion: v1
kind: Deployment
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  replicas: 5
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.10.0
        ports:
        - containerPort: 80
```

Google Cloud

Run `kubectl get deployments` or describe deployments to ensure the proper number of replicas are running





Skills Challenge tracks and badges

g.co/cloud/skillschallenge