GCP Certification Full course in Hindi

Associate Cloud Engineer

DEVOPS MADE SIMPLE

Youtube channel link- [Google Cloud Platform (GCP) Full Course 2025 - Hindi | Associate Cloud Engineer Certification - YouTube]

Google Cloud

Compute Engine

- Create, Start and Stop VMs
- Instance Templates
- Machine Images
- ✓ Startup Scripts
- Attaching GPU to VMs
- Instance Groups – Unmanaged, Managed (Stateless & Stateful)
- SSH Keys – login and SSH OS Login

Google Cloud

Persistent Disks

- Attaching Non-boot disk
- Resize disks
- Disk Images
- Disk Snapshots
- Using local SSDs for VMs
- Creating VMs from Disk Images or Disk Snapshots
- Encryption using KMS

IAM (Identity & Access Management)

- IAM Roles – Basic, Pre-defined & Custom
- IAM Policy
- Service Accounts
- IAM Service Account Keys

VPC (Virtual Private Cloud)

- VPC – auto mode and custom mode
- Networking
- NAT (Network Address Translation)
- Domains
- Private Google Access
- VPC Firewall Rules –
  - Ingress, Egress and Policies

Load Balancers

Google Cloud

- HTTP
- HTTPS
- TCP Proxy
- SSL Proxy

Cloud Storage

Google Cloud

Basics	Storage classes	Object Lifecycle Management
Versioning	Security ACLs	Security UBLA

GKE (Google K8s Engine)

Google Cloud

- Pods
- Deployments
- Replica Sets
- Stateful Sets
- Daemon Sets
- Services
  - Node port, Cluster IP, Ingress
- Cluster Management
- Auto- Scaling
  - Cluster Auto-scaler, Horizontal & Vertical Pod Auto-scaling
- Storage

Google Cloud

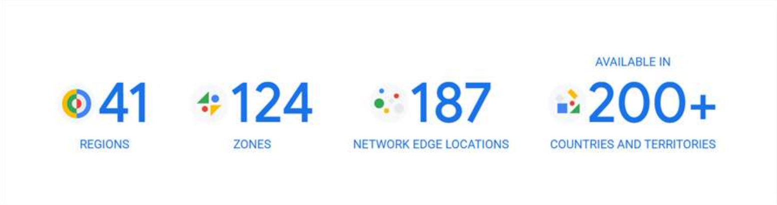
Others-

- Cloud Logging
- Cloud Monitoring
- Cloud Run
- Cloud Functions

Regions-

- independent geographic areas
  - Examples - us-central1, us-east1
- consists of three or more zones
- Regions matter for following reasons –
  - ▪ High-availability
  - ▪ Low latency
  - ▪ Compliance
  - ▪ Pricing
- Total – 41 Regions

Google Cloud



41 REGIONS

124 ZONES

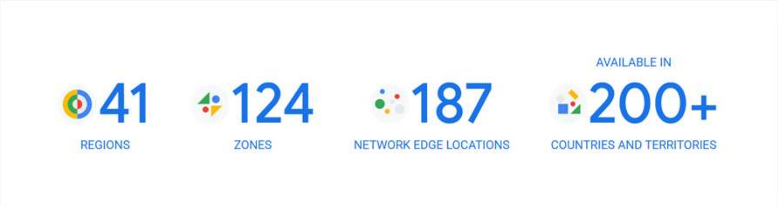
187 NETWORK EDGE LOCATIONS

AVAILABLE IN 200+ COUNTRIES AND TERRITORIES

Zones

- a deployment area within a region
  - Example - `asia-northeast1-a`, `asia-northeast1-b`
- fault-tolerance and high availability

Google Cloud



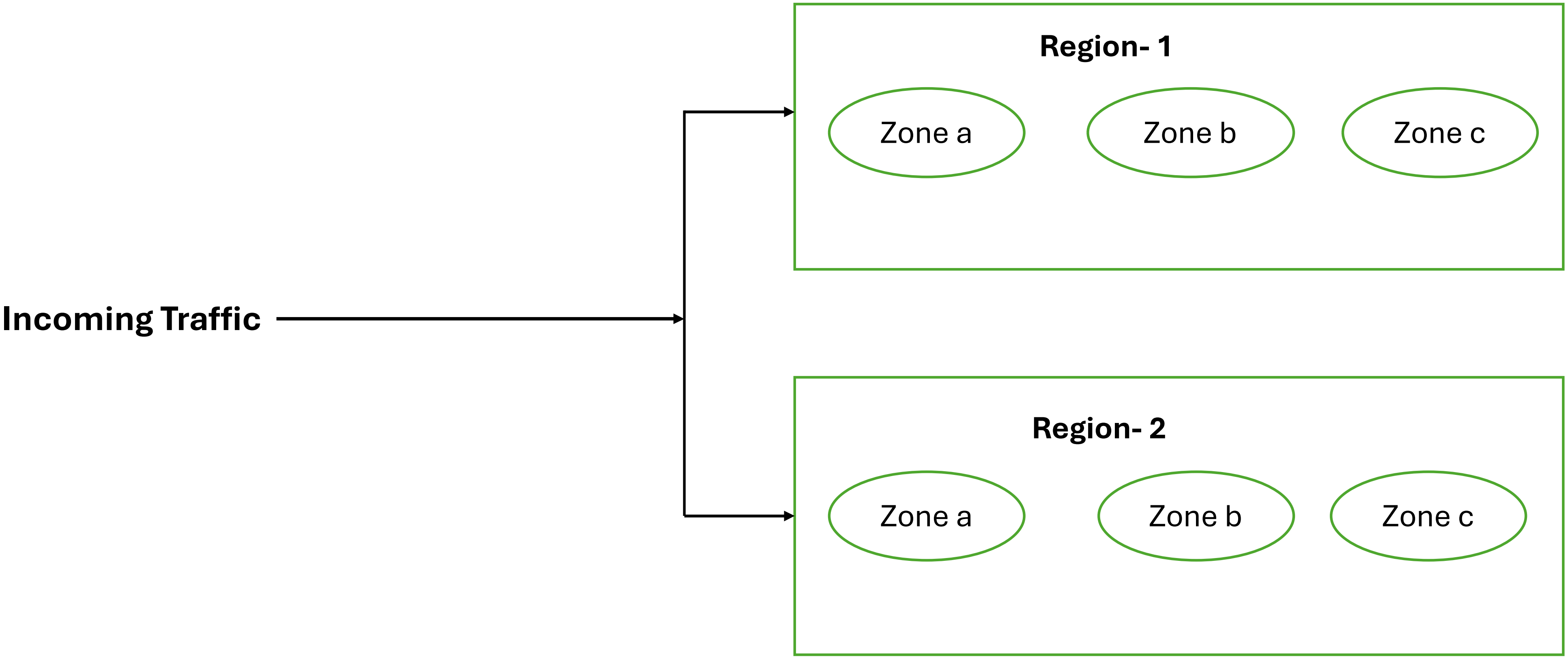
41 REGIONS

124 ZONES

187 NETWORK EDGE LOCATIONS

AVAILABLE IN 200+ COUNTRIES AND TERRITORIES

Google Cloud



Incoming Traffic

Region- 1

Zone a

Zone b

Zone c

Region- 2

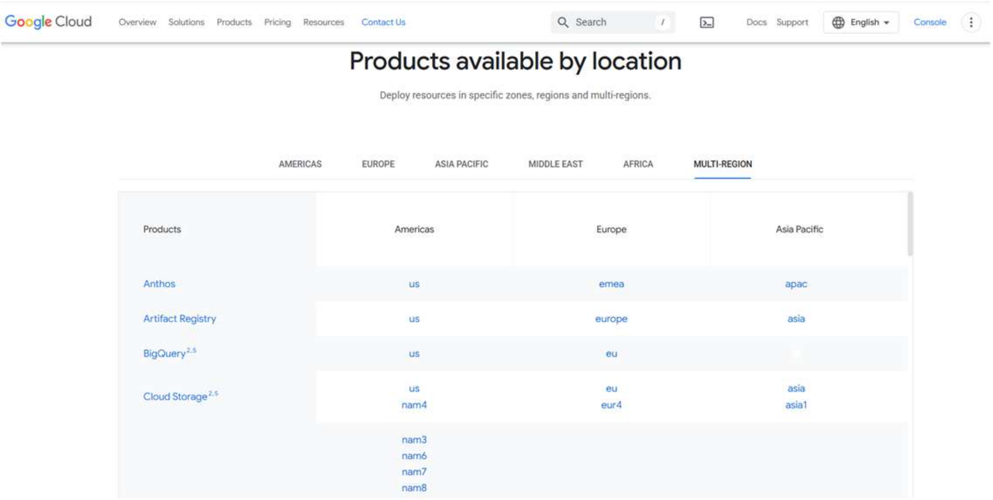
Zone a

Zone b

Zone c

Multi- region

- Some services in Google Cloud provide **multi-region configurations**.
- These services automatically replicate data across multiple regions by default, ensuring redundancy and efficient distribution across different regions or zones



Products available by location

Deploy resources in specific zones, regions and multi-regions.

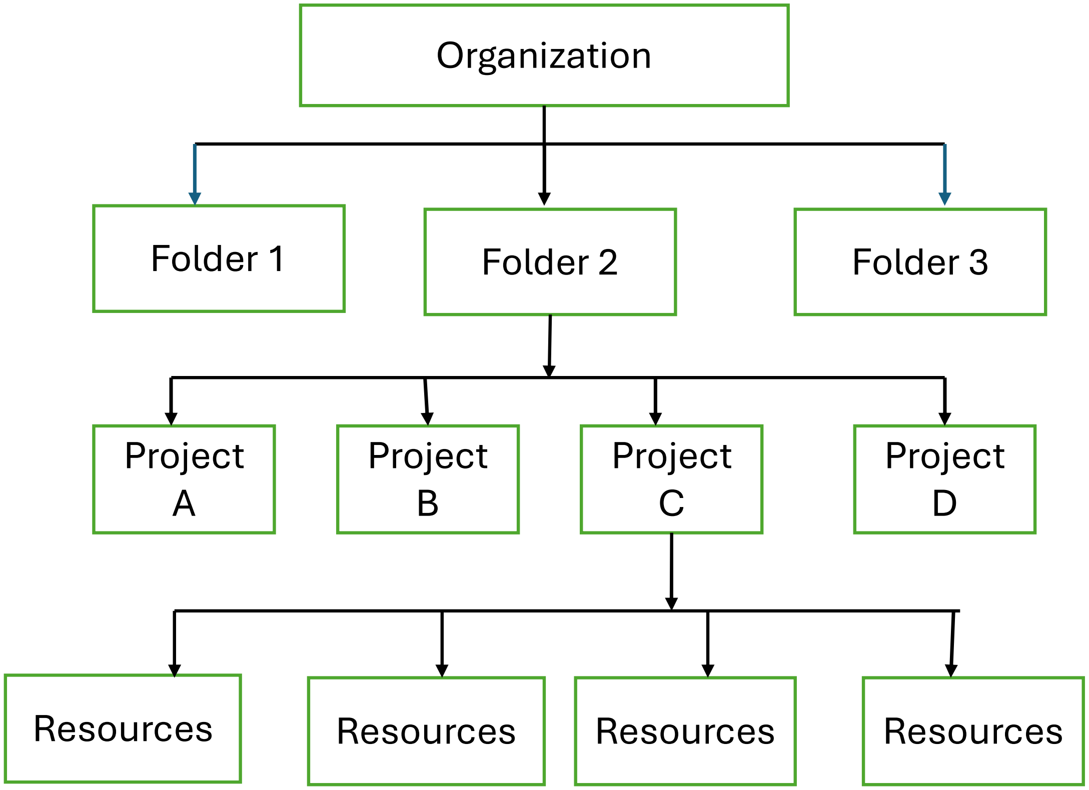
Products	Americas	Europe	Asia Pacific
Anthos	us	emea	apac
Artifact Registry	us	europe	asia
BigQuery[2,5]	us	eu	
Cloud Storage[2,5]	us nam4	eu eur4	asia asia1
	nam3 nam6 nam7 nam8		

Resource Hierarchy in Google Cloud

The purpose of the Google Cloud resource hierarchy is two-fold:

Provide a hierarchy of ownership, which binds the lifecycle of a resource to its immediate parent in the hierarchy.

Provide attach points and inheritance for access control and organization policies



Organization-

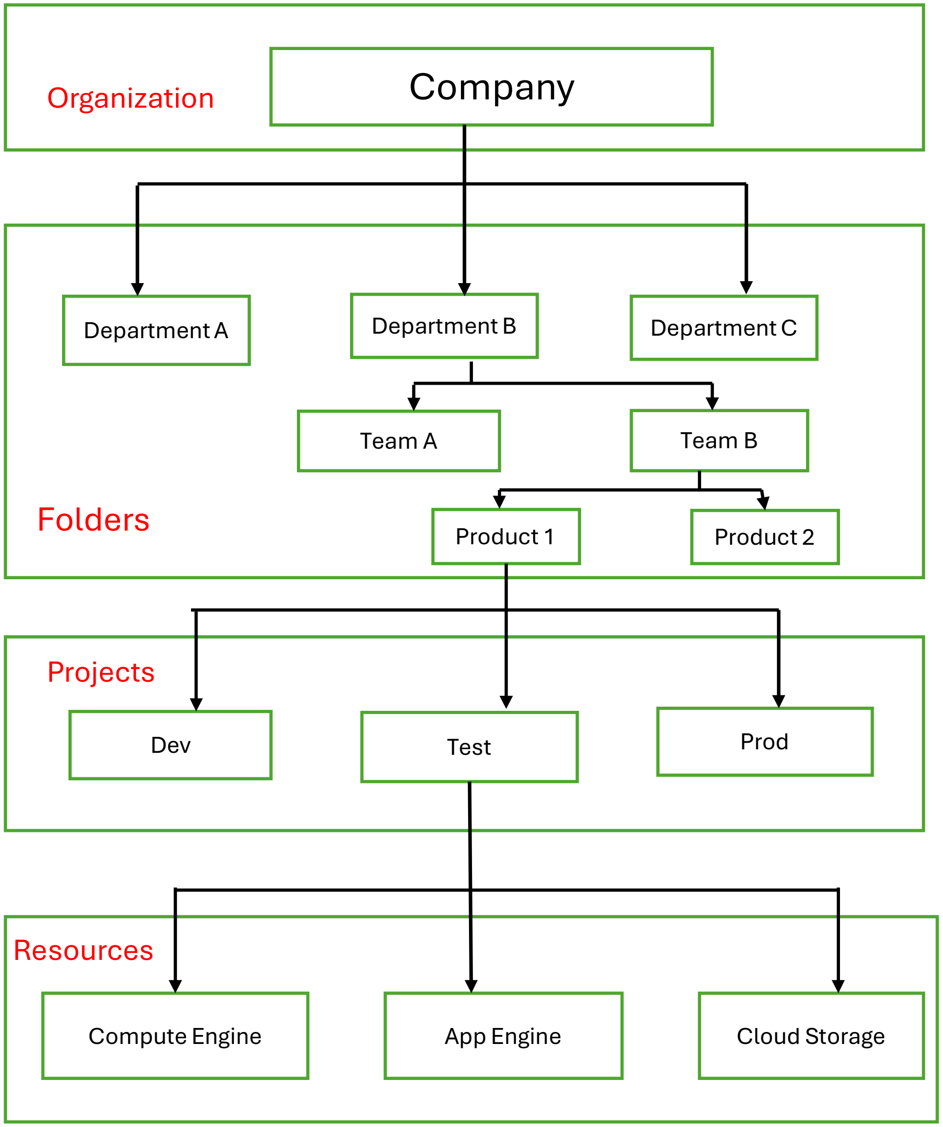
- root node in the Google Cloud
- hierarchical ancestor of folder and project resources
- policies applied on the organization apply throughout the hierarchy on all resources in the organization
- Organization Administrators have central control of all resources. They can view and manage all projects

Folders-

Folder resources optionally provide an additional grouping mechanism and isolation boundaries between projects

They can be seen as sub-organizations within the organization resource

IAM roles granted on a folder resource are automatically inherited by all project and folder resources included in that folder


Organization
Company
Department A
Department B
Department C
Team A
Team B
Folders
Product 1
Product 2
Projects
Dev
Test
Prod
Resources
Compute Engine
App Engine
Cloud Storage

Projects-

- The project resource is the base-level organizing entity
- Organization and folder resources may contain multiple projects.
- forms the basis for creating, enabling, and using all Google Cloud services, managing APIs, enabling billing, adding and removing collaborators, and managing permissions
- moving a project resource from one folder resource to another will change the inherited permissions
- All project resources consist two identifiers:
    - Project resource ID, which is a unique identifier for the project resource.
    - Project resource number, which is automatically assigned when you create the project. It is read-only.
- One mutable display name.

Compute Engine

Compute Engine is an infrastructure as a service (IaaS) product that offers self-managed virtual machine (VM) instances and bare metal instances.

Here are some of the benefits of using Compute Engine:

- **Extensibility:** Compute Engine integrates with Google Cloud technologies such as Cloud Storage, Google Kubernetes Engine, and BigQuery, to extend beyond the basic computational capability to create more complex and sophisticated applications.
- **Scalability:** Scale the number of compute resources as needed without having to manage your own infrastructure.
- **Reliability:** Google's infrastructure is highly reliable, with a 99.9% uptime guarantee.
- **Cost-effectiveness:** Compute Engine offers a variety of pricing options to fit your budget. Also, you only pay for the resources that you use, and there are no up-front costs.

Types of Machines –

- **Custom Machines** - Allows you to configure CPU and memory to achieve the optimal balance for your applications based on your needs.
- **Pre-defined Machines** –
  - they are pre-built
  - They are configured with specific amounts of vCPUs, Memory and GPUs
- All vCPUs and GB of memory are charged minimum of 1 minute, after that they are charged in 1 second increment.

Compute Engine terminology

- **Machine family**: A curated set of processor and hardware configurations optimized for specific workloads.
- **Machine series**: Machine families are further classified by series, generation, and processor type.
  - Each series focuses on a different aspect of computing power or performance. For example, the E series offers efficient VMs at a low cost, while the C series offer better performance.
- **Machine type**: Every machine series offers at least one machine type. Each machine type provides a set of resources for your compute instance, such as vCPUS, memory, disks, and GPUs.
  - For example, a c3-standard-22 machine type has 22 vCPUs, and as a standard machine type, it also ha 88 GB of memory.

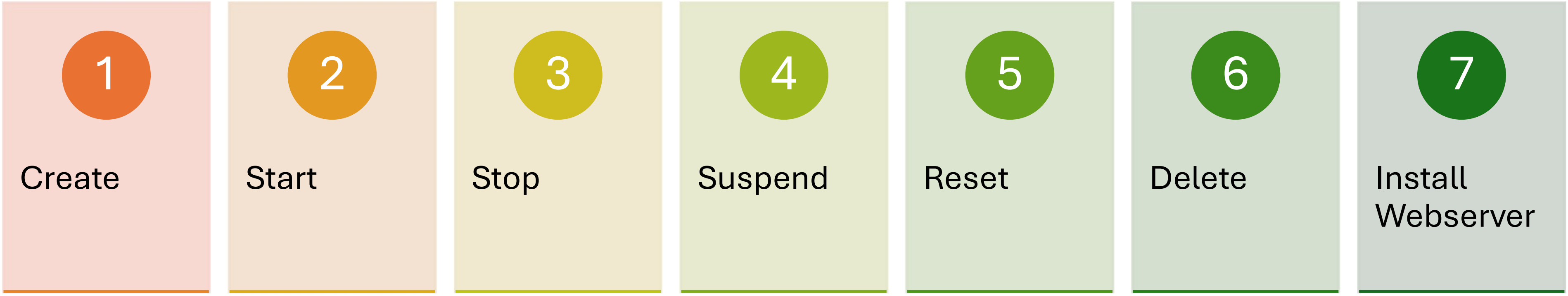
Machine families-

- **General-purpose** —best price-performance ratio for a variety of workloads.
- **Storage-optimized** —best for workloads that are low in core usage and high in storage density.
- **Compute-optimized** —highest performance per core on Compute Engine and optimized for compute-intensive workloads.
- **Memory-optimized** —ideal for memory-intensive workloads, offering more memory per core than other machine families, with up to 12 TB of memory.
- **Accelerator-optimized** —ideal for massively parallelized Compute Unified Device Architecture (CUDA) compute workloads, such as machine learning (ML) and high-performance computing (HPC). This family is the best option for workloads that require GPUs.

General-purpose workloads			
N4, N2, N2D, N1	**C4A, C4, C3, C3D**	**E2**	**Tau T2D, Tau T2A**
Balanced price/performance across a wide range of machine types	Consistently high performance for a variety of workloads	Day-to-day computing at a lower cost	Best per-core performance/cost for scale-out workloads
• Medium traffic web and app servers • Containerized microservices • Business intelligence apps • Virtual desktops • CRM applications • Development and test environments • Batch processing • Storage and archive	• High traffic web and app servers • Databases • In-memory caches • Ad servers • Game Servers • Data analytics • Media streaming and transcoding • CPU-based ML training and inference	• Low-traffic web servers • Back office apps • Containerized microservices • Microservices • Virtual desktops • Development and test environments	• Scale-out workloads • Web serving • Containerized microservices • Media transcoding • Large-scale Java applications

Optimized workloads			
Storage-optimized	**Compute-optimized**	**Memory-optimized**	**Accelerator-optimized**
Z3	**H3, C2, C2D**	**X4, M3, M2, M1**	**A3, A2, G2**
Highest block storage to compute ratios for storage-intensive workloads	Ultra high performance for compute-intensive workloads	Highest memory to compute ratios for memory-intensive workloads	Optimized for accelerated high performance computing workloads
● SQL, NoSQL, and vector databases ● Data analytics and data warehouses ● Search ● Media streaming ● Large distributed parallel file systems	● Compute-bound workloads ● High-performance web servers ● Game Servers ● High performance computing (HPC) ● Media transcoding ● Modeling and simulation workloads ● AI/ML	● Medium to extra-large SAP HANA in-memory databases ● In-memory data stores, such as Redis ● Simulation ● High Performance databases such as Microsoft SQL Server, MySQL ● Electronic design automation	● Generative AI models such as the following: ● Large Language Models (LLM) ● Diffusion Models ● Generative Adversarial Networks (GAN) ● CUDA-enabled ML training and inference ● High-performance computing (HPC) ● Massively parallelized computation ● BERT natural language processing ● Deep learning recommendation model (DLRM) ● Video transcoding ● Remote visualization workstation

Virtual Machines (VMs)-



1. Create
2. Start
3. Stop
4. Suspend
5. Reset
6. Delete
7. Install Webserver

Suspend or Resume a Compute Engine instance

- Suspending an instance preserves the instance and migrates the contents of the instance's memory to storage
- After resuming the instance, Compute Engine migrates the instance's memory from storage back to the instance, and the instance starts running again.
- Suspending an instance is **analogous to closing the lid of your laptop**, and it's useful in the following scenarios:
  - You want to stop paying for the core and memory costs of running an instance, and pay the comparatively cheaper cost of storage, to preserve the state of your instance instead.
  - You don't need the instance at this time, but you want to be able to bring it back up quickly with its OS and application state where you left it.
- An instance can be suspended for maximum 60 days, after that VM is automatically stopped.

Stopping an instance-

- Stopping an instance is useful when you no longer use it, or to modify its properties.
  - For example, to change its machine type, or remove any attached and mounted disks.
- When an instance is stopped, current External IP is lost.

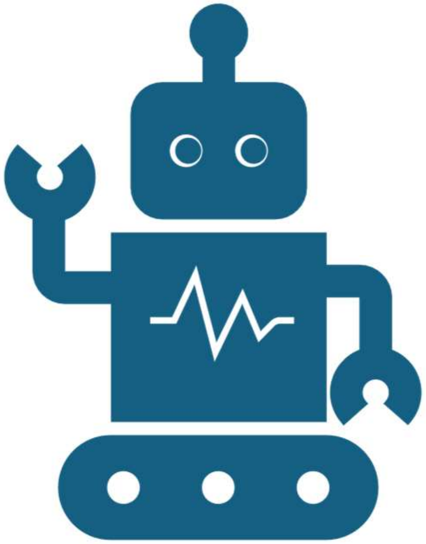
Reset a Compute Engine instance

- Resetting an instance can help ensure optimal performance and stability, or help resolve issues like a frozen, slow, or crashing guest operating system (OS)
- Resetting an instance is analogous to doing a reset of your computer, such as when you press a reset button or press and hold the power button.

•Re-initializes the instance to its initial boot state, without modifying metadata or disks.
•Wipes the contents of the instance's memory.
•Keeps the instance state to RUNNING throughout the reset operation.

Delete a Compute Engine instance

- If you no longer need an instance, then delete it to stop incurring charges for the instance and its attached resources.
- Compute Engine deletes the instance and all attached resources within a few seconds
- To prevent accidental deletion **Enable deletion protection in Advanced options** section



GPU (Graphics Processing Units)

- GPUs accelerate specific workloads on VMs- such as machine learning (ML) and data processing
- GPUs are supported for N1 general-purpose, and the accelerator-optimized (A3, A2, and G2) machine series.
- N1 general-purpose machine types, you attach the GPU to the VM during, or after VM creation.
- For A3, A2 or G2 machine types, the GPUs are automatically attached when you create the VM.

Startup Scripts

A startup script is a file that contains commands that run when a virtual machine (VM) instance boots.

Compute Engine provides support for running startup scripts on Linux VMs and Windows VMs.

Important notes

1. The startup script runs on *every boot*.
2. The startup script has default root privileges.

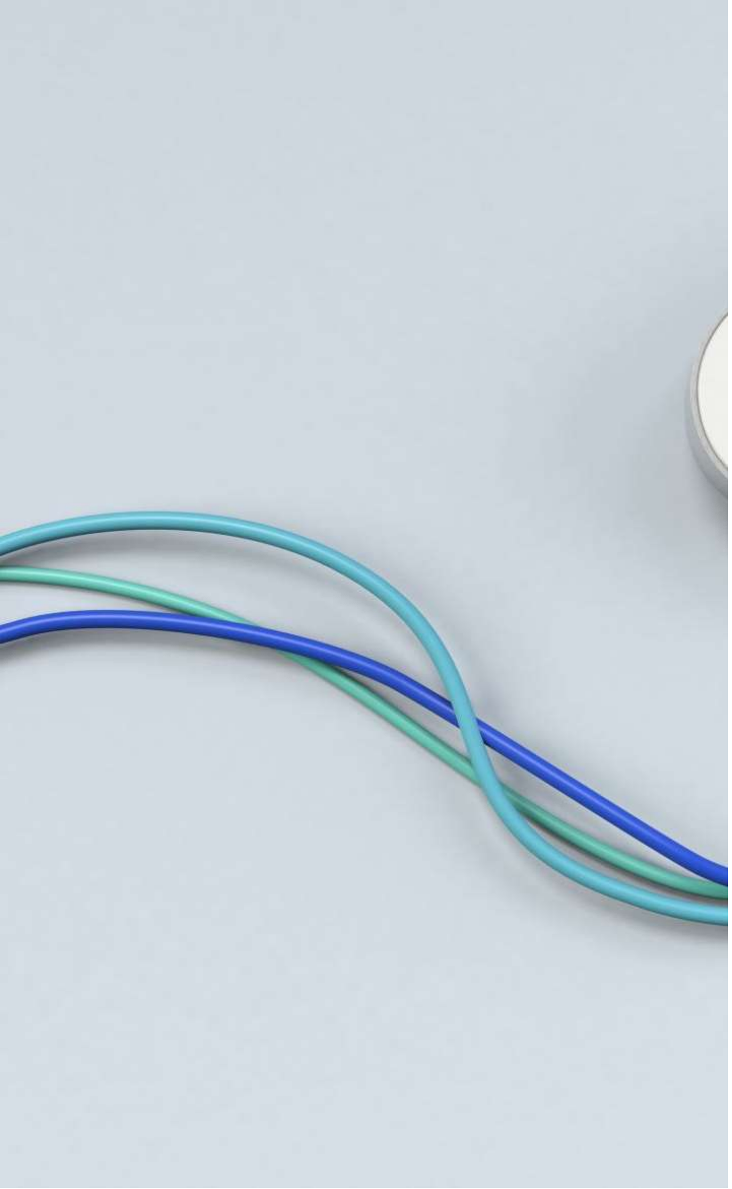
- When we specify a startup script, Compute Engine does the following:
  1. Copies the startup script to the VM
  2. Sets run permissions on the startup script
  3. Runs the startup script as the root user when the VM boots

- We can use multiple startup scripts
- **Order of execution of startup script:-**
  - First Preference – Startup script provided directly or locally
  - Second Preference – Startup Script stored in cloud storage

- We can configure Startup Script at both VM level and Project Level
- **Note :** VM level startup Script will override the Project level startup script
- For Linux level Start up script, we can use both bash or non-bash file. To use a non-bash file. designate the interpreter by adding a **#!** To the top of file.
  - For example – to use a Python3 startup script - #! /usr/bin/python3

Ways to specify Startup Script -

1. Providing Startup script directly-
   - in the automation box, when we create VM
   - In the Metadata
     - Key: startup-script
     - Value: copy your script
2. Provide a startup script from cloud storage. For this update the metadata section-
   - Key: startup-script-url
   - Value: URL
3. Provide a script from a local file using gcloud



Cloud Shell

- Cloud Shell provisions a Compute Engine virtual machine running a Debian-based Linux operating system for your temporary use.
- Cloud Shell provisions 5 GB of free persistent disk storage mounted as your $HOME directory on the virtual machine instance.
- All files you store in your home directory, including installed software, scripts and user configuration files persist between sessions. Your $HOME directory is private to you and can't be accessed by other users.
- If you do not need persistent storage, use Cloud Shell in ephemeral mode.
- The built-in code editor provides the convenience of viewing and editing files in the same environment where projects are built and deployed.
- you can access your Git repositories (or create a new one), view existing and staged changes, and merge changes.

gcloud-

The Google Cloud CLI is a set of tools to create and manage Google Cloud resources. You can use these tools to perform many common platform tasks from the command line or through scripts and other automation.

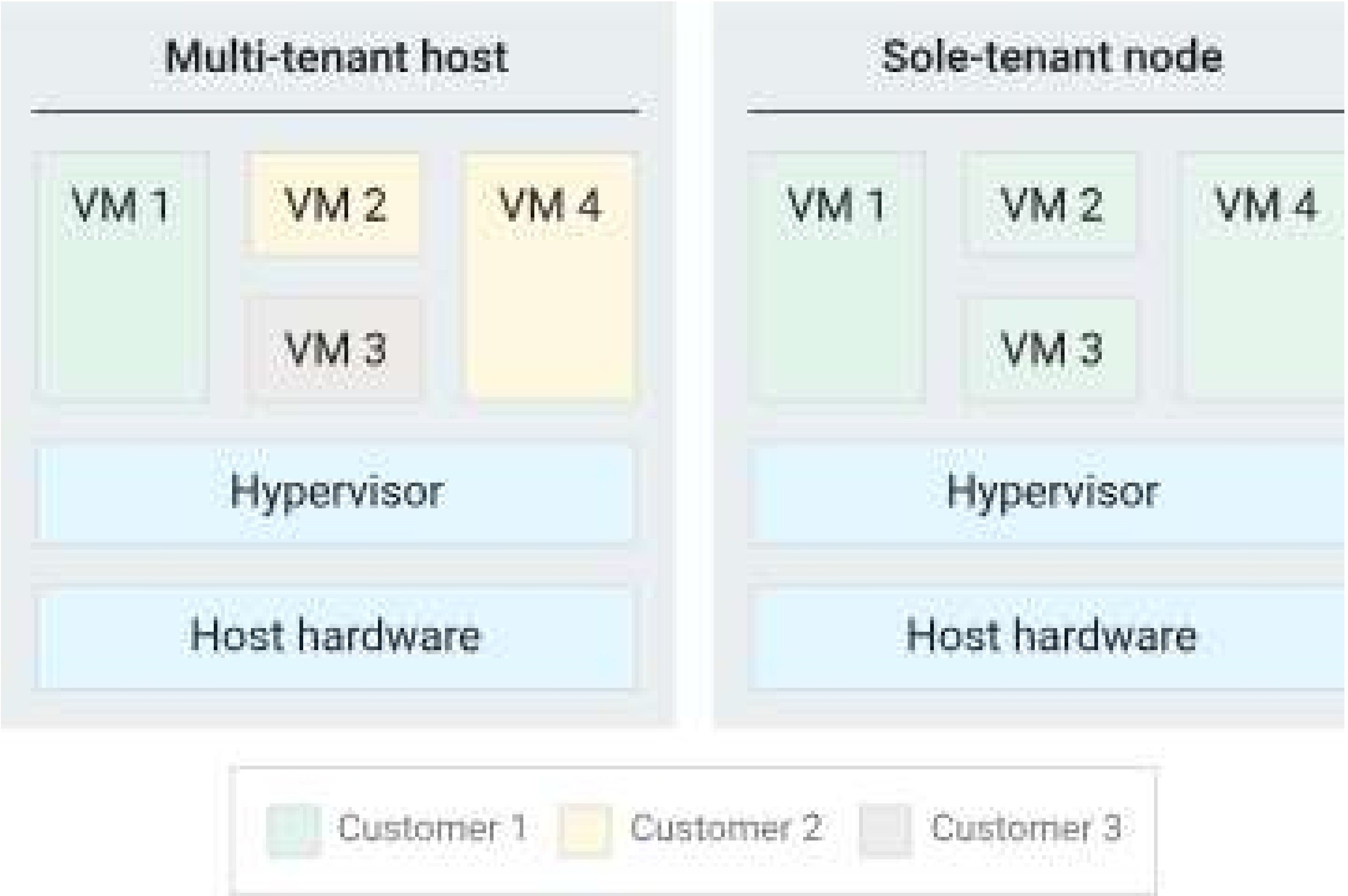
For example, you can use the gcloud CLI to create and manage the following:

- Compute Engine virtual machine instances and other resources
- Cloud SQL instances
- Google Kubernetes Engine clusters
- Dataproc clusters and jobs
- Cloud DNS managed zones and record sets
- Cloud Deployment Manager deployments

- gcloud can be installed on local
- Available to use by default on cloud shell

Sole Tenant nodes

- Sole-tenant node is a physical Compute Engine server that is dedicated to hosting only your project's VMs.
- Use sole-tenant nodes to keep your VMs physically separated from VMs in other projects, or to group your VMs together on the same host hardware.
- create a sole-tenant node group and specify whether you want to share it with other projects or with the entire organization.
- Sole-tenant nodes can help you meet dedicated hardware requirements for bring your own license (BYOL) scenarios that require per-core or per-processor licenses


Multi-tenant host
VM 1
VM 2
VM 4
VM 3
Hypervisor
Host hardware
Sole-tenant node
VM 1
VM 2
VM 4
VM 3
Hypervisor
Host hardware
Customer 1
Customer 2
Customer 3

Workload considerations

The following types of workloads might benefit from using sole-tenant nodes:

- Gaming workloads with performance requirements.
- Finance or healthcare workloads with security and compliance requirements.
- Windows workloads with licensing requirements.
- Machine learning, data processing, or image rendering workloads. For these workloads, consider reserving GPUs.
- Workloads requiring increased I/O operations per second (IOPS) and decreased latency, or workloads that use temporary storage in the form of caches, processing space, or low-value data.

Node templates

A node template is a regional resource that defines the properties of each node in a node group. When you create a node group from a node template, the properties of the node template are immutably copied to each node in the node group.

Node types

The sole-tenant node type, specifies the total amount of vCPU cores and memory for nodes created in node groups that use that template. For example, the n2-node-80-640 node type has 80 vCPUs and 640 GB of memory.

Node groups

Sole-tenant node templates define the properties of a node group, and you must create a node template before creating a node group in a Google Cloud zone. When you create a group, specify the host maintenance policy for VMs on the node group, the number of nodes for the node group

Maintenance windows

If you are managing workloads—for example—finely tuned databases, you can determine when maintenance begins on a sole-tenant node group by specifying a maintenance window when you create the node group. You can't modify the maintenance window after you create the node group.

gcloud command to create sole tenant node-

```
gcloud compute sole-tenancy node-groups create GROUP_NAME \
 --node-template=TEMPLATE_NAME \
 --target-size=TARGET_SIZE \
[--zone=ZONE \]
 [--maintenance-policy=MAINTENANCE_POLICY \]
[--maintenance-window-start-time=START_TIME \]
[--autoscaler-mode=AUTOSCALER_MODE: \
 --min-nodes=MIN_NODES \
--max-nodes=MAX_NODES]
```

Instance Templates

- An instance template is a convenient way to save a virtual machine (VM) instance's configuration that includes machine type, boot disk image, labels, startup script, and other VM properties.
- Use an instance template to do the following:
  - Create individual VMs.
  - Create VMs in a managed instance group (MIG).
  - Create reservations for VMs.
  - Create future reservations for VMs.

When to use instance templates

- Use instance templates any time you want to quickly create VMs or reservations for VMs based off of a pre-existing VM property. If you want to create a group of identical VMs (a MIG), then you must create the MIG using an instance template.

How to update instance templates

- Instance templates are designed to create VMs with identical configurations. You can't update instance templates after they're created. Instead, do one of the following:
- Create a new instance template as follows:
  - Create an instance template based on an existing VM.
  - Create an instance template based on an existing template.
- Create VMs while overriding the instance template's properties.

Use deterministic instance templates to ensure identical VMs

- Deterministic instance templates make explicitly clear the type of third-party services or apps to install on your VMs. This helps to ensure that your instance template always creates VMs with an identical configuration. For example, if your template has a startup script that fetches an app, you can specify the version of the app that you want in your template's startup script.

Regional and global instance templates

- Instance templates are available both as regional and global resources.
- Unless you need to reuse an instance template across multiple regions, Google recommends using regional instance templates over global instance templates.

	Regional instance template	**Global instance template**
Scope	You can use the template only in the template's region.	You can use the template in any region.
Reliability	Hardware errors are isolated to the template's region.	Hardware errors can impact any region where the template is used.
Use case	•Reduce cross-region dependency. •Achieve data residency in a region. For example, to meet compliance requirements on the physical location of the data.	Reuse your global instance template to create VMs, MIGs, and reservations across multiple regions.

Use of zonal or regional resources in instance templates

- In an instance template, you might specify zonal resources, which restricts the use of that template to the zone where that resource resides. Similarly, if you specify a regional resource in a global instance template, the template is restricted to that region.
- **For example**, if you include a read-only Persistent Disk from **us-central1-a** in your instance template, you can't use that template in any other zone because that specific Persistent Disk exists only in zone us-central1-a.

- **gcloud command to create instance template from scratch-**

```
gcloud compute instance-templates create gcp-test \
  --machine-type=e2-micro \
  --instance-template-region=us-central1 \
  --network-interface=subnet=default \
  --tags=http-server \
  --metadata-from-file=startup-script=startup-script.sh
```

- **gcloud command to create a VM instance from an instance template-**

```
gcloud compute instances create test-vm \
 --source-instance-template projects/learn-gcp-451419/regions/us-central1/instanceTemplates/gcp-test
```

- **gcloud command to create instance template based on an existing instance-**

```
 gcloud compute instance-templates create gcp-test-2 \
--source-instance=test-vm \
--source-instance-zone=us-central1-a \
--instance-template-region=us-central1
```

Machine Images-

- A machine image is a Compute Engine resource that stores all the configuration, metadata, permissions, and data from multiple disks of a virtual machine (VM) instance.
- You can use a machine image in many system maintenance, backup and recovery, and instance cloning scenarios.

- Machine images can be stored in a Cloud Storage multi-region, such as asia or a Cloud Storage region, such as asia-south1.
- By default, when creating a machine image from an instance, the machine image is stored in either the Cloud Storage multi-region bucket that contains the source instance, or the geographically closest Cloud Storage multi-region bucket to the source instance.

For example, if your source instance is stored in us-central1 your machine image is stored in the us multi-region by default. However, a default location like australia-southeast1 is outside of a multi-region. The closest multi-region is asia.

- **A machine image collects the following information from the source instance:**

- VM instance configuration. Each VM configuration includes the following properties:
  - Description
  - Machine type
  - Instance metadata
  - Labels
  - Network tags
  - Maintenance policy
- The volume mapping used to create disks for the source instance.
- Data stored on disks at consistent points in time across the disks.

- **The following information from the source instance is not collected by a machine image:**

- Data in memory.
- Data stored in attached Local SSD disks. However, a machine image captures the device mapping of the Local SSD disks.
- Attributes that are specific to the source instance, such as the name or IP address.

- **Note: -** You can't override any properties of the attached disk other than the name of the disk while creating an instance from the machine image.

- **gcloud command to list Machine images-**

```
gcloud compute machine-images list
```

- **gcloud command to create Machine image from VM instance –**

```
gcloud compute machine-images create test-mi \
--source-instance=mi-vm \
--source-instance-zone=us-central1-a \
--storage-location=us
```

- **gcloud command to create VM instance from Machine image –**

```
gcloud compute instances create test-mi-vm \
--zone=us-central1-a \
--source-machine-image=test-mi
```

- **gcloud command to delete Machine image –**

```
gcloud compute machine-images delete test-mi
```

Spot VMs-

- Spot VMs have significant discounts, but Compute Engine might preemptively stop or delete (preempt) Spot VMs to reclaim the capacity at any time.
- Spot VMs are available at much lower prices—60-91% discounts for most machine types and GPUs as well as smaller discounts for some other resources—compared to the on-demand price for standard VMs
- Spot VMs are excess Compute Engine capacity, so their availability varies based on Compute Engine usage.

Spot VMs limitations

- Compute Engine might preempt Spot VMs to reclaim the resources at any time.
- Spot VMs are finite Compute Engine resources, so they might not always be available.
- Spot VMs can't live migrate to become standard VMs while they are running or be set to automatically restart when there is a host event.
- Spot VMs are not covered by any Service Level Agreement
- The Google Cloud Free Tier credits for Compute Engine do not apply to Spot VMs.
- Spot VMs are only available for supported machine types.

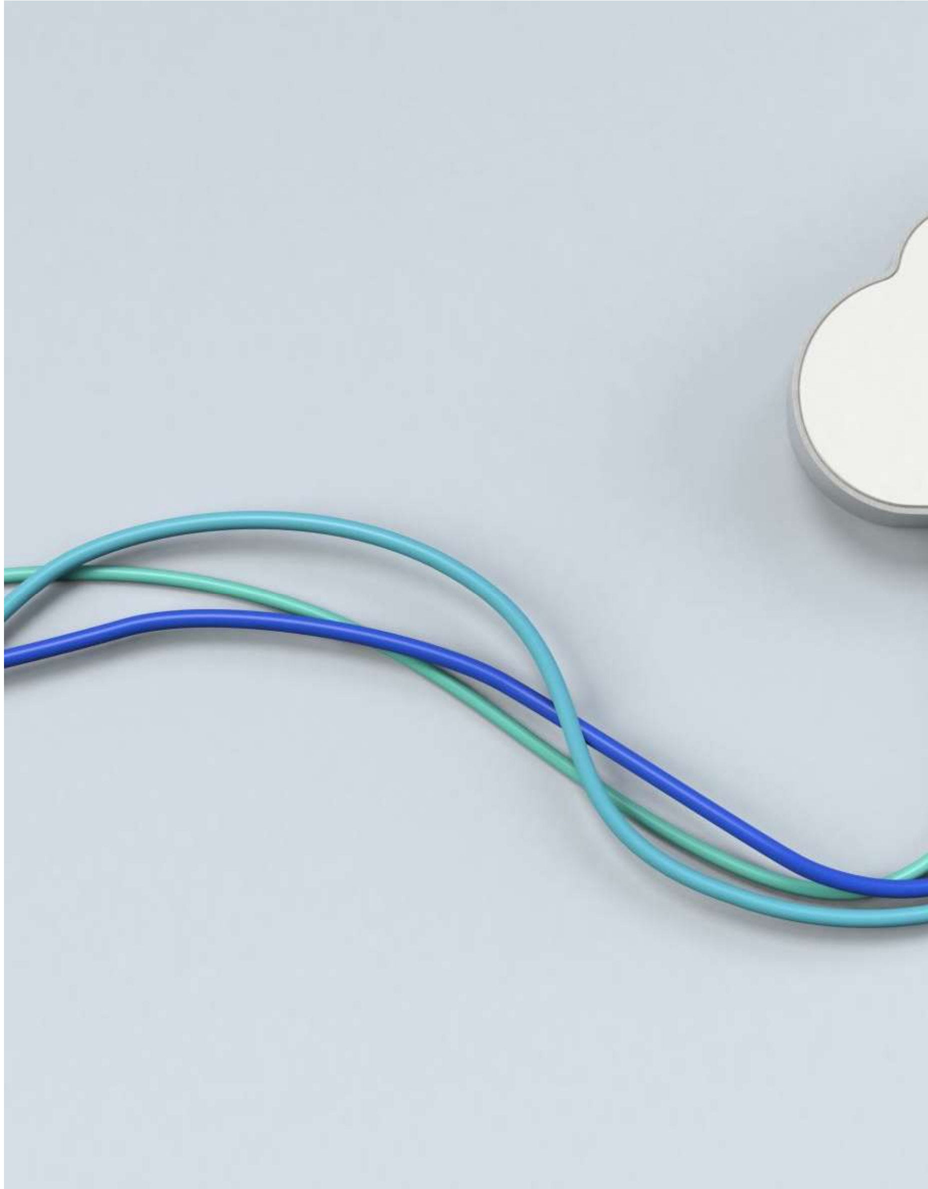
  For example - All machine series support Spot VMs with the exception of the M2, M3, and X4 machine series, and C3 bare metal machine types.

gcloud command to create Spot VM-

```
gcloud compute instances create VM_NAME \
--provisioning-model=SPOT \
--instance-termination-action=TERMINATION_ACTION \  (STOP or DELETE)
--zone=us-central1-a \
--machine-type=e2-micro \
--network-interface=subnet=default \
```

Autoscaling-

- Autoscaling works by adding more VMs to your MIG when there is more load (scaling out) and deleting VMs when the need for VMs is lowered (scaling in).
- Target utilization metrics-
  - Average CPU utilization
  - HTTP load balancing serving capacity
  - Cloud Monitoring metrics



Schedules

You can use schedule-based autoscaling to allocate capacity for anticipated loads. You can have up to 128 scaling schedules per instance group. For each scaling schedule, specify the following:

- **Capacity**: minimum required VM instances
- **Schedule**: start time, duration, and recurrence (for example, once, daily, weekly, or monthly)

Each scaling schedule is active from its start time and for the configured duration. During this time, autoscaler scales the group to have at least as many instances as defined by the scaling schedule.

Predictive autoscaling

If you enable predictive autoscaling to optimize your MIG for availability, the autoscaler forecasts future load based on historical data and scales out a MIG in advance of predicted load, so that new instances are ready to serve when the load arrives.

Predictive autoscaling works best if your workload meets the following criteria:

•Your application takes a long time to initialize—for example, if you configure a initialization period of more than 2 minutes.

•Your workload varies predictably with daily or weekly cycles.

Autoscaling mode

- If you need to investigate or configure your group without interference from autoscaler operations, you can temporarily turn off or restrict autoscaling activities.
- The autoscaler's configuration persists while it is turned off or restricted, and all autoscaling activities resume when you turn it on again or lift the restriction.

Instance groups

- An instance group is a collection of virtual machine (VM) instances that you can manage as a single entity.
- Compute Engine offers two kinds of VM instance groups, managed and unmanaged:

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.

Unmanaged instance groups let you load balance across a fleet of VMs that you manage yourself.

Regional or Zonal groups

Two types of Managed Instance Groups:

- **Zonal MIG -** deploys instances to a single zone.
- **Regional MIG -** deploys instances to multiple zones across the same region.
- Regional MIGs add higher availability by spreading application load across multiple zones, which protects your workload against zonal failure, and regional MIGs offer more capacity.
- By default, you can create up to 2,000 VMs in a regional MIG and 1,000 VMs in a zonal MIG.
- If you need more VMs, you can increase the size limit of your MIG

Unmanaged instance groups

- Heterogeneous instances (non-identical VMs)
- Do not offer autoscaling, autohealing, multi-zone support, or the use of instance templates
- Not a good fit for deploying highly available and scalable workloads
- Supports Load balancing

Automatic repair and autohealing

- You might want to repair instances when an application freezes, crashes, or runs out of memory. Application-based autohealing improves application availability by relying on a health checking signal that detects application-specific issues such as freezing, crashing, or overloading.
- If a health check determines that an application has failed on a VM, the group automatically recreates that VM instance.

- **gcloud command to create Unmanaged Instance Groups**

```
gcloud compute instance-groups unmanaged create unmanaged-test \
--description=creating-unmanaged-instance-group \
--zone=us-central1-a
```

- **gcloud command to set ports -**

```
gcloud compute instance-groups unmanaged set-named-ports unmanaged-test \
--zone=us-central1-a \
--named-ports=http-port=80
```

- **add VMs to Unmanaged Instance Groups-**

```
gcloud compute instance-groups unmanaged add-instances unmanaged-test \
--zone=us-central1-a \
--instances=test1,test2
```

- **gcloud command to create health checks**

```
gcloud compute health-checks create http health-test \
--port=80 \
--request-path=index.html \
--response=DevOps \
--region=us-central1 \
--no-enable-logging \
--check-interval=5 \
--timeout=5 \
--unhealthy-threshold=2 \
--healthy-threshold=2
```

- **gcloud command to set Firewall rules -**

```
gcloud compute firewall-rules create health-test \
--allow tcp:80 \
--source-ranges 130.211.0.0/22,35.191.0.0/16 \
--network default
```

Managed Instance Groups -Stateless

- Automatically repairing failed VMs
- Application-based autohealing
- Regional (multiple zone) coverage
- Load balancing
- Scalability
- Automated updates
- Instance template is required

MIG in a single zone - Putting all your MIG's VMs in a single zone helps to minimize latency, which is useful for certain workloads

- **gcloud command to create zonal MIG-**

```
gcloud compute instance-groups managed create INSTANCE_GROUP_NAME \
  --size SIZE \
  --template INSTANCE_TEMPLATE \
  --zone ZONE
```

MIG with VMs in multiple zones in a region (Regional MIG)

its VMs spread across multiple zones in a region. Spreading your application load across multiple zones protects your workload against zonal failures.

- **gcloud command to create regional MIG**

```
gcloud compute instance-groups managed create example-rmig \
  --template example-template \
  --size 30 \
  --zones us-east1-b,us-east1-c
```

gcloud command to set autoscaling-

```
gcloud compute instance-groups managed set-autoscaling INSTANCE_GROUP_NAME \
--zone=us-central1-a \
--cool-down-period=120 \
--max-num-replicas=4 \
--min-num-replicas=2 \
--mode=on \
--target-cpu-utilization =0.5 \
--scale-in-control=max-scaled-in-replicas=1, time-window=300
```

gcloud command to set autoscaling schedules-

```
gcloud compute instance-groups managed update-autoscaling INSTANCE_GROUP_NAME \
--zone=us-central1-a \
--schedule-cron='0 4 * * Mon' \
--schedule-duration-sec= 360 \
--schedule-time-zone='Asia\Kolkata' \
--schedule-min-required-replicas= 3 \
--schedule-description='testing-schedule'
```

Managed Instance Groups -Stateful

- A stateful managed instance group (stateful MIG) preserves the unique state of each virtual machine (VM) instance—including VM name, attached persistent disks, IP addresses, and/or metadata—on machine restart, recreation, auto-healing, or update.

- A stateful managed instance group (MIG) takes its instance configuration from a combination of the instance template, optional all-instances configuration, optional stateful policy, and optional per-instance configurations that you set.

You can make the following changes to a stateful policy:

- Configure disks to become stateful by adding them to the stateful policy.
- Configure disks to become stateless by removing them from the stateful policy.
- Specify that IP addresses must be stateful by adding network interface configuration to the stateful policy.
- Specify that IP addresses must be treated as stateless by removing the configuration from stateful policy.

MIG Stateful Features

Application-based autohealing

Regional (multiple zone) coverage

Load balancing

Automated updates

Instance template is required

Auto-scaling not supported

Live migration

- To keep an instance running during a host event, Compute Engine performs a *live migration* of the instance to another host server in the same zone
- Live migration lets Google Cloud perform maintenance without interrupting a workload, rebooting an instance, or modifying any of the instance's properties, such as IP addresses, metadata, block storage data, application state, or network settings.

Live migration keeps instances running during the following situations:

- **Infrastructure maintenance.** Infrastructure maintenance includes host hardware, network and power grids in data centers, and host operating system (OS) and BIOS.
- **Security-related updates and system configuration changes.** These include events such as installing security patches and changing the size of the host root partition for storage of the host OS image and packages.
- **Hardware failures.** This includes failures in memory, CPUs, network interface cards, and disks. If the failure is detected before there is a complete server failure, then Compute Engine performs a preventative live migration of the instance to a new host server. If the hardware fails completely or otherwise prevents live migration, then the instance terminates and restarts automatically.

- During live migration, Google Cloud ensures a minimum disruption time, which is typically much less than 1 second.

- If a VM is not set to live migrate, Compute Engine terminates the VM during host maintenance. VMs that are set to terminate during a host event stop and (optionally) restart.

Limitations

Live migration is not supported for the following VM types:

•**Bare metal instances**. C3 and X4 bare metal instances don't support live migration. The maintenance behavior for these instances is set to TERMINATE and RESTART, respectively.

- **VMs with GPUs attached**. VM instances with GPUs attached must be set to stop and optionally restart. Compute Engine offers a 60-minute notice before a VM instance with a GPU attached is stopped.

•**Cloud TPUs**. Cloud TPUs don't support live migration.

•**Storage-optimized VMs**. Z3 VMs don't support live migration. The maintenance behavior for Z3 VMs is set to TERMINATE.

gcloud command to set host maintenance policy of an instance

```
gcloud compute instances create INSTANCE_NAME \
--zone=ZONE \
--maintenance-policy=MAINTENANCE_BEHAVIOR \  (Terminate or Migrate)
--RESTART_ON_FAILURE_BEHAVIOR \  (restart-on-failure or no-restart-on-failure)
```

Ops Agent

Combines the collection of logs, metrics, and traces into a single process.

the Ops Agent uses **Fluent Bit** for logs, which supports high-throughput logging,

the **OpenTelemetry Collector** for metrics and traces.

Ops Agent features

Overall features include:

•Single download and installation/upgrade process.

•Simple, unified, YAML-based configuration.

•Support for standard Linux and Windows distros.

Logging features

Logging features include:

- **Improved performance compared to the legacy Logging Agent:**
    - High throughput capability, taking full advantage of multi-core architecture.
    - Efficient resource (e.g. memory, CPU) management.
- **Collecting logs from various sources:**
    - Standard system logs (/var/log/syslog and /var/log/messages for Linux, Windows Event Log)
    - File-based logs with customizable paths and refresh interval.
    - Journald daemon / systemd logs.
    - Logs over TCP protocol.
    - Logs over Forward protocol (used by Fluent Bit and Fluentd).
- **Flexible processing:**
    - Parse text logs into structured logs: JSON-based and regular-expression-based parsing.
    - Modify log entries by removing, renaming, or setting fields.
    - Exclude logs based on labels and regular expressions.
    - Detect and concatenate multiline language-exception logs from Java, Python, and Golang.
- **Third-party application support**
    - Curated third-party application log integration that recognizes common app log file paths and formats.

Monitoring features

Monitoring features include:

- System metrics collected with no configuration. Metrics collected include:
  - cpu metrics, disk metrics, interface metrics, gpu metrics (Linux only), memory metrics, mssql metrics (Windows only), network metrics, processes metrics, agent self metrics
- Third-party application support
  - Curated integrations for third-party application metrics, which collect common app metrics and offer sample dashboards and alert policies.
- Collection of Prometheus metrics from applications running on Compute Engine.
- Collection of NVIDIA Data Center GPU Manager (DCGM) metrics.

Ops Agent Installation Methods-

- Install the Ops Agent during VM creation
- Install the Ops Agent by using automation tools
  - Example – Ansible, Chef, Puppet
- Installing the Ops Agent on individual VMs
- Installing the Ops Agent with Google Cloud CLI or Terraform

How to install Ops-Agent on VM-

gcloud command to create VM instance-

```
gcloud compute instances create ops-agent-test \
--zone=us-central1-a \
--machine-type=e2-micro \
--network-interface=subnet=default
```

download Ops-Agent

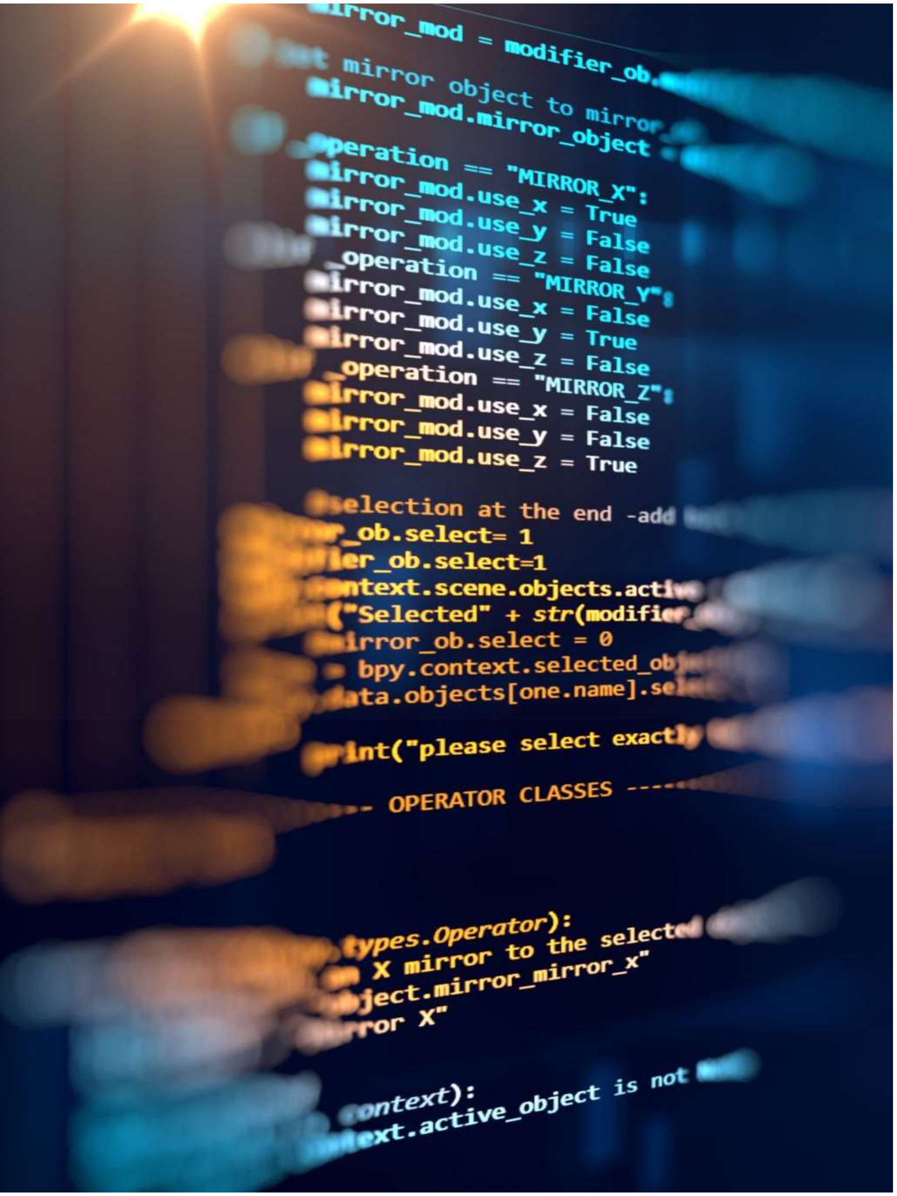
```
curl -sSO https://dl.google.com/cloudagents/add-google-cloud-ops-agent-repo.sh
```

Install Ops-Agent -

```
sudo bash add-google-cloud-ops-agent-repo.sh --also-install
```

To verify that the agent is working as expected, run:

```
sudo systemctl status google-cloud-ops-agent"*"
```



SSH connections

1. Metadata-managed SSH connections
    - Google Cloud Console or gcloud
    - Generating SSH keys and adding Public SSH key to metadata

    (Project level or Instance level)

2. OS login
    - Setting key-value in metadata (Project level and Instance level)
    - Generating SSH keys and adding Public SSH key

Metadata Managed- Steps to create SSH keys and adding Public SSH key to Metadata

- Generate SSH keys

ssh-keygen –t rsa –f metadata-key –C user1

- Give proper permissions to Private SSH key

chmod 400 metadata-key

- Add Public SSH key to Project level or Instance level metadata

cat metadata-key.pub

- Access the VM instance using Private SSH key

ssh –i metadata-key user1@VM_External_IP

OS Login managed Setting the key-value pair in metadata (Project level or instance level)

- Set the following key-value pair in metadata section-

Key- *enable-oslogin*

Value- *TRUE*

- Now you can access the VM instance using Google Cloud console or gcloud command

OS Login Managed Steps to add SSH Public key

- Generate SSH keys
- Add Public SSH key

```
gcloud compute os-login ssh-keys add \
--key-file=os-login.pub \
--ttl=0
```

- Describe profile

```
gcloud compute os-login describe-profile
```

- Verify the connectivity

```
ssh –i oslogin username@external-ip
```

Block Storage

- The block storage, commonly referred to as *disks* or *volumes*, can be used for boot and data volumes for all compute instances, including virtual machines (VMs), containers, and bare metal instances.
- Google Cloud offers two types of block storage—temporary and durable block storage.
- You can combine these block storage types in a single compute instance.
- One Block Storage can be connected to one VM instance in read-write mode.
- We can attach read-only block storage devices to multiple VM instances.
- One VM instance can be connected to multiple block storage devices.

Temporary block storage

Temporary, or ephemeral, block storage offers the fastest performance among all block storage types, with the tradeoff that the *stored data is lost if the VM stops for any reason*.

Data is lost if you stop, suspend, or restart the VM, or if the VM crashes or fails.

Durable block storage

- Durable, or persistent, block storage is for data that you want to preserve after you stop, suspend or delete the VM, or even if the VM crashes or fails.

- Hyperdisk and Persistent Disk are the durable block storage offerings in Google Cloud, but Persistent Disk isn't available with the latest machine series.

- Google recommends using Hyperdisk for the highest performance and advanced features.

Hyperdisk and Persistent Disk volumes have the following characteristics:

- **Function as physical disks**: you can use a Hyperdisk or Persistent Disk volume with compute instance as if it was a physical disk attached to the instance.

- **Portability**: Hyperdisk or Persistent Disk volumes are independent of the compute instances that you attach them to. This characteristic means you can attach a volume to a running instance without downtime. You can also detach the volume to keep your data even after you delete the instance.

- **Security**: by default, data is encrypted at rest and in transit. You can also customize the encryption with your own keys.

- **High availability options**: protect your data from zonal failures by replicating the volume across two zones.

Differences between Hyperdisk and Persistent Disk

Hyperdisk offers the following key advantages over Persistent Disk:

- Customizable performance: You can independently configure the performance and size of each Hyperdisk volume. Hyperdisk performance is independent of provisioned capacity. This feature means that you can increase or decrease a Hyperdisk volume's performance without changing its size.

- Unlike Hyperdisk, Persistent Disk performance depends on provisioned capacity. Consequently, to improve a Persistent Disk volume's performance, you must increase its size.

- Better overall performance: Hyperdisk has higher IOPS and throughput limits than Persistent Disk.

Persistent Disk

- Persistent Disk volumes are durable network storage devices that your instances can access like physical disks in a desktop or a server.
- When you read or write from a Persistent Disk, data is transmitted over the network.
- You can detach or move the volumes to keep your data even after you delete your instances.
- Persistent Disk performance increases with size, so you can resize your existing Persistent Disk volumes or add more Persistent Disk volumes to a VM to meet your performance and storage space requirements.

Persistent Disk types

- **Balanced Persistent Disk -** solid-state drives (SSD)
  - This disk type offers performance levels suitable for most general-purpose applications at a price point between that of standard and performance (pd-ssd) Persistent Disk
- **Performance (SSD) Persistent Disk -** solid-state drives (SSD)
  - Suitable for enterprise applications and high-performance databases that require lower latency and more IOPS than standard Persistent Disk provides.
- **Standard Persistent Disk** - standard hard disk drives (HDD)
  - Suitable for large data processing workloads that primarily use sequential I/Os.
- **Extreme Persistent Disk** - solid-state drives (SSD)
  - Offers consistently high performance for both random access workloads and bulk throughput.
  - Designed for high-end database workloads.
  - Lets you provision the target IOPS.
  - you must attach your extreme persistent disks to virtual machine (VM) instances that are large machine types, including M2, M3, or N2-64 and larger machine types.
  - Extreme persistent disks are zonal only. You cannot create regional extreme persistent disks.
  - You cannot attach multiple VM instances in read-only mode to an extreme persistent disk.
  - You cannot create an image or machine image from an extreme persistent disk.
  - You can resize an Extreme Persistent Disk only once in a 6 hour period.

Zonal Persistent Disk

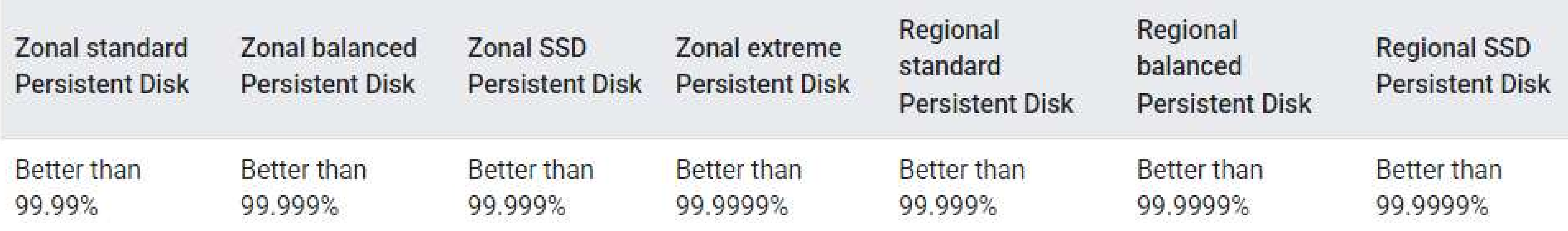
- A zonal Persistent Disk is a Persistent Disk that's accessible only within one specific zone

Regional Persistent Disk

- Persistent Disk replicated across two zones in the same region are called Regional Persistent Disk.
- One of the zones must be the same zone that the VM is running in.

Limitations for Regional Persistent Disk-

- You can attach regional Persistent Disk only to VMs that use E2, N1, N2, and N2D machine types.
- You cannot create a regional Persistent Disk from an image, or from a disk that was created from an image.
- When using read-only mode, you can attach a regional balanced Persistent Disk to a maximum of 10 VM instances.
- The minimum size of a regional standard Persistent Disk is 200 GiB.



Zonal standard Persistent Disk	Zonal balanced Persistent Disk	Zonal SSD Persistent Disk	Zonal extreme Persistent Disk	Regional standard Persistent Disk	Regional balanced Persistent Disk	Regional SSD Persistent Disk
Better than 99.99%	Better than 99.999%	Better than 99.999%	Better than 99.9999%	Better than 99.999%	Better than 99.9999%	Better than 99.9999%

Restrictions

- You can't attach a Persistent Disk volume to an VM in another project.
- You can attach a balanced Persistent Disk to a maximum of 10 VMs in read-only mode.
- For custom machine types or predefined machine types with a minimum of 1 vCPU, you can attach up to 128 Persistent Disk volumes.
- Each Persistent Disk volume can be up to 64 TiB in size.
- Most VMs can have up to 128 Persistent Disk volumes and up to 257 TiB of total disk space attached. Total disk space for a VM includes the size of the boot disk.
- Shared-core machine types are limited to 16 Persistent Disk volumes and 3 TiB of total Persistent Disk space.

- **gcloud command to create Zonal Persistent Disk Volume-**

```
gcloud compute disks create test-disk \
 --size=20GB \
 --type=pd-balanced \
--zone=us-central1-a
```

- **gcloud command to create Regional Persistent Disk Volume-**

```
gcloud compute disks create test-disk \
--size=20GB \
--type=pd-balanced \
--region=us-central1 \
--replica-zones=projects/learn-gcp/zones/us-central1-a,projects/learn-gcp/zones/us-central1-b
```

- **gcloud command to attach Persistent Disk to any running or stopped VM-**

```
gcloud compute instances attach-disk test-disk-vm \
--disk=test-disk
```

Format and mount a non-boot disk on a Linux VM

1.In the terminal, use the symlink created for your attached disk to determine which device to format.

```
ls -l /dev/disk/by-id/google-*
```

The device name for the new disk is **sdb**.

2.Format the disk device using the mkfs tool

```
sudo mkfs.ext4 -m 0 -E lazy_itable_init=0,lazy_journal_init=0,discard /dev/sdb
```

3.Create a directory that serves as the mount point for the new disk on the VM.

```
sudo mkdir -p /mnt/disks/test-dir
```

4.Use the mount tool to mount the disk to the instance, and enable the discard option

```
sudo mount -o discard,defaults /dev/sdb /mnt/disks/test-dir
```

5.Configure read and write permissions on the disk

```
sudo chmod a+w /mnt/disks/test-dir
```

6.Create file on Mounted Disk

```
echo "This is test file" >> /mnt/disks/test-dir/disk.txt
cat /mnt/disks/test-dir/disk.txt
```

Configure automatic mounting on VM restart

Add the disk to your ***/etc/fstab*** file, so that the disk automatically mounts again when the VM restarts. On Linux operating systems, the device name can change with each reboot, but the device UUID always points to the same volume, even when you move disks between systems.

1.Create a backup of your current /etc/fstab file.

```
sudo cp /etc/fstab /etc/fstab.backup
```

2.Use the blkid command to list the UUID for the disk.

```
sudo blkid /dev/DEVICE_NAME
```

3.Open the /etc/fstab file in a text editor and create an entry that includes the UUID.

```
UUID=UUID_VALUE /mnt/disks/test-dir ext4 discard,defaults,nofail 0 2
```

nofail mount option - lets the system boot even if the disk is unavailable

4.Use the cat command to verify that your /etc/fstab entries are correct

```
cat /etc/fstab
```

Increase the size of a persistent disk

1.gcloud to resize the disk-

```
gcloud compute disks resize test-dir \
--size 20GB \
--zone=us-central1-a
```

2. Use the df and the lsblk commands to list the size of the file system and to find the device names for your disks.

```
sudo df -Th
```

3. Extend the file system using the resize2fs command:

```
sudo resize2fs /dev/sdb
```

4. Use the df command to verify that the file system is extended

```
sudo df -h /dev/sdb
```

5. *sudo lsblk*

Local SSD disks

- Consider using Local solid-state drive (Local SSD) disks, if your workloads need high performance, low latency, temporary storage.
- Local SSD disks offer superior I/O operations per second (IOPS), and very low latency compared to the persistent storage provided by Hyperdisk and Persistent Disk.
- Local SSD disks are physically attached to the server that hosts your instance

Local SSD disks are ideal when you need storage for any of the following use cases:

- Caches or storage for transient data
- Scratch processing space for high performance computing or data analytics
- Temporary data storage

- Local SSD performance depends on several factors, including the number of attached Local SSD disks, the selected disk interface (NVMe or SCSI), and the instance's machine type.
- The available performance increases as you attach more Local SSD disks to your instance

- Compute Engine preserves the data on Local SSD disks in certain scenarios, and in other cases, Compute Engine does not guarantee Local SSD data persistence.

Scenarios where Compute Engine persists Local SSD data

- Data on Local SSD disks persist only through the following events:
- If you reboot the guest operating system.
- If you configure your instance for live migration and the instance goes through a host maintenance event.
- If you opt to preserve the Local SSD data when you stop or suspend the instance. This feature is in Preview.

Local SSD encryption

- Compute Engine automatically encrypts your data when it is written to Local SSD disks. You can't use customer-supplied encryption keys with Local SSD disks.

Local SSD has the following limitations:

- You can't use Local SSD disks with VMs with shared-core machine types.
- You can't attach Local SSD disks to instances that use N4, H3, M2 E2, or Tau T2A machine types.
- You can't use customer-supplied encryption keys or customer-managed encryption keys with Local SSD disks. Compute Engine automatically encrypts your data when it's written to Local SSD storage.
- You can't back up Local SSD disks with snapshots, clones, machine images, or images. Store important data on Hyperdisk or Persistent Disk volumes

gcloud command to create a VM with Local SSD disk storage

```
gcloud compute instances create test-instance \
  --zone=us-central1-a \
  --machine-type=n2d-standard-2 \
  --network-interface=subnet=default \
  --local-ssd=interface=NVME
```

Format and mount a Local SSD on a Linux VM

1.In the terminal, identify the Local SSD that you want to mount.

```
lsblk
```

2.Format the Local SSD with an ext4 file system. This command deletes all existing data from the Local SSD.

```
sudo mkfs.ext4 -F /dev/disk/by-id/google-local-nvme-ssd-0
```

3.Create a directory that serves as the mount point for the local ssd on the VM.

```
   sudo mkdir -p /mnt/disks/test-dir
```

4.Use the mount tool to mount the disk to the instance, and enable the discard option

```
   sudo mount /dev/disk/by-id/google-local-nvme-ssd-0 /mnt/disks/test-dir
```

5.Configure read and write permissions on the disk

```
   sudo chmod a+w /mnt/disks/test-dir
```

6.Create file on Mounted Disk

```
   echo "This is test file" >> /mnt/disks/test-dir/disk.txt
   cat /mnt/disks/test-dir/disk.txt
```

Adding the Local SSD to the /etc/fstab file so that the device automatically mounts again when the instance restarts

1.Create a backup of your current /etc/fstab file.

```
sudo cp /etc/fstab /etc/fstab.backup
```

2.Use the blkid command to list the UUID for the disk.

```
sudo blkid -s UUID
```

3.Open the /etc/fstab file in a text editor and create an entry that includes the UUID.

```
echo UUID= /mnt/disks/test-dir ext4 discard,defaults,nofail 0 2 | sudo tee -a /etc/fstab
```

nofail mount option - lets the system boot even if the disk is unavailable

4.Use the cat command to verify that your /etc/fstab entries are correct

```
cat /etc/fstab
```

Hyperdisk

Google Cloud Hyperdisk, the newest generation of network block storage service, is designed for the most demanding mission-critical applications

Hyperdisk offers a scalable, high-performance storage service with a comprehensive suite of data persistence and management capabilities.

Hyperdisk storage capacity is partitioned and made available to virtual machine (VM) instances as individual volumes.

Hyperdisk volumes are decoupled from VMs enabling you to attach, detach, and move volumes between VMs.

Data stored on Hyperdisk volumes are persistent over VM reboots and deletions.

Hyperdisk volumes have the following features:

- A Hyperdisk volume is mounted as a disk on a VM using an NVMe or SCSI interface, depending on the machine type of the VM.
- Hyperdisk volumes feature substantially better performance than Persistent Disk. With Hyperdisk, you get dedicated IOPS and throughput with each volume, as compared to Persistent Disk where performance is shared between volumes of the same type.
- Hyperdisk lets you scale storage performance and capacity to match your workload needs.
- To access static data from multiple VMs, you can attach the same disk in read-only mode to hundreds of VMs.
- The maximum total Hyperdisk capacity is 512 TiB for VMs with 32 or more vCPUs, and 257 TiB for VMs with 1 to 31 vCPUs.
- Synchronous replication is available with Hyperdisk Balanced High Availability, which synchronously replicates data between two zones in the same region. This ensures high availability (HA) for Hyperdisk Balanced High Availability disk data for up to one zonal failure.
- Hyperdisk Extreme, Hyperdisk Balanced, Hyperdisk ML are designed for sub-millisecond latencies.

Hyperdisk types

1. Hyperdisk Balanced -

- good fit for LOB applications, web applications, and medium-tier databases
- Use it for applications where multiple VMs in the same zone simultaneously require write access to the same disk

2. Hyperdisk Balanced High Availability

- synchronously replicate the disk data across two zones in the same region
- use it for applications where multiple VMs in the same regions simultaneously require write access to the same disk.

3. Hyperdisk ML

- highest throughput available and the fastest data load times
- For large inference and training workloads, you can attach a single Hyperdisk ML volume to multiple VMs in read-only mode.

4. Hyperdisk Extreme

- feature higher maximum IOPS and throughput along with low sub-millisecond latencies, and offer high performance for the most demanding workloads, such as high-performance databases.
- For performance-critical applications, use Hyperdisk Extreme if Extreme Persistent Disk isn't supported or doesn't provide enough performance.

5. Hyperdisk Throughput

- flexibly provision capacity and throughput as needed
- For scale-out analytics workloads like Hadoop and Kafka, cold storage, and data drives for cost sensitive apps

How Hyperdisk storage works

- Hyperdisk volumes are durable network storage devices that your VMs can access, similar to Persistent Disk volumes.
- The data on each Hyperdisk is distributed across several physical disks
- Hyperdisk volumes are located independently from your VMs, so you can detach or move Hyperdisk volumes to keep your data, even after you delete your VMs.
- Hyperdisk performance is decoupled from size, so you can update the performance, resize your existing Hyperdisk volumes or add more Hyperdisk volumes to a VM to meet your performance and storage space requirements.
- Hyperdisk volumes use the NVMe or SCSI storage interface, depending on the VM machine type.

Machine type restrictions

- To use Hyperdisk Balanced with A3 VMs, the VM must have at least 8 GPUs.
- For Hyperdisk Extreme, the following restrictions apply:
  - A3 machine types require at least 4 GPUs.
  - C3 machine type require at least 88 vCPUs.
  - C3D machine types require at least 60 vCPUs.
  - C4 machine types require at least 96 vCPUs.
  - M1 machine types require at least 80 vCPUs.
  - C4A and M3 machine types require at least 64 vCPUs.
  - N2 requires 80 or more vCPUs; Custom N2 machine types aren't supported.
- You can't use Hyperdisk Throughput with C3-metal machine types.

Share Hyperdisk volumes between VMs

You can share a Hyperdisk volume between multiple VMs by simultaneously attaching the same volume to multiple VMs.

The following scenarios are supported:

- Concurrent read-write access to a single volume from multiple VMs. Recommended for clustered file systems and highly available workloads like SQL Server Failover Cluster Infrastructure. Supported for Hyperdisk Balanced and Hyperdisk Balanced High Availability volumes.
- Concurrent read-only access to a single volume from multiple VMs. This is more cost effective than having multiple disks with the same data. Recommended for accelerator-optimized machine learning workloads. Supported for Hyperdisk ML volumes.
- You can't attach a Hyperdisk Throughput or Hyperdisk Extreme volume to more than one VM.

Encryption for Hyperdisk volumes

By default, Compute Engine protects your Hyperdisk volumes with Google-owned and Google-managed encryption keys. You can also encrypt your Hyperdisk volumes with customer-managed encryption keys (CMEK).

Limitations for Hyperdisk

- You can't create a machine image from a Hyperdisk volume.
- You can't back up a disk in multi-writer mode with snapshots or images. You must disable multi-writer mode first.
- You can't create an image from a Hyperdisk Extreme, Hyperdisk Throughput, or Hyperdisk Balanced High Availability volume.
- You can't create an instant snapshot from a Hyperdisk volume.
- You can't attach Hyperdisk Throughput or Hyperdisk Extreme volumes to more than one VM.
- Hyperdisk Extreme, Hyperdisk ML and Hyperdisk Throughput volumes can't be used as boot disks.
- You can attach a Hyperdisk ML volume to up to 100 VMs at most once every 30 seconds.
- You can't create a Hyperdisk ML disk in read-write mode from a snapshot or a disk image. You must create the disk in read-only mode.
- If you enable read-only mode for a Hyperdisk ML volume, you can't re-enable read-write mode.

Create a new Hyperdisk volume

- Create a blank, non-boot, and zonal Hyperdisk volume and attach it to your instance either during or after instance creation.
- Format and mount the volume to provide access to a data or file system.
- For Hyperdisk Balanced volumes, you can also create boot disks as well as data disks.

gcloud command to create the Hyperdisk volume.

```
gcloud compute disks create DISK_NAME \
  --zone=ZONE \
  --size=DISK_SIZE \
  --type=DISK_TYPE \
  --provisioned-iops=IOPS_LIMIT
  --provisioned-throughput=THROUGHPUT_LIMIT
  --access-mode=DISK_ACCESS_MODE
```

DISK_ACCESS_MODE: Optional: How compute instances can access the data on the disk. Supported values are:

READ_WRITE_SINGLE, for read-write access from one instance. This is the default.
READ_WRITE_MANY, (Hyperdisk Balanced and Hyperdisk Balanced High Availability only) for concurrent read-write access from multiple instances.
READ_ONLY_MANY, (Hyperdisk ML only) for concurrent read-only access from multiple instances.

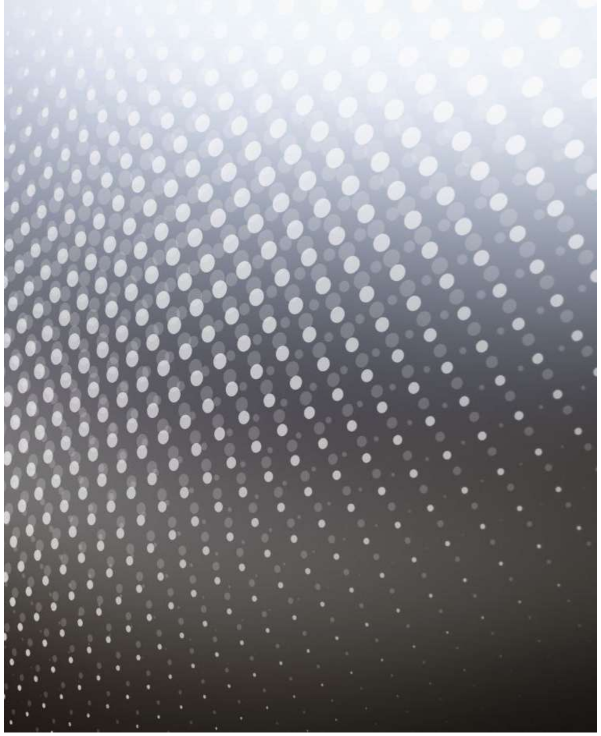
Hyperdisk Storage Pools

Hyperdisk Storage Pool is a pre-purchased collection of capacity, throughput, and IOPS which you can then provision to your applications as needed.

You can use Hyperdisk Storage Pools to create and manage disks in pools and use the disks across multiple workloads.

By managing disks in aggregate, you can save costs while achieving expected capacity and performance growth.

By using only the storage you need in Hyperdisk Storage Pools, you reduce the complexity of forecasting capacity and reduce management toil by going from managing hundreds of disks to managing a single storage pool



Storage pools include the following benefits:

Lower Total Cost of Ownership (TCO)—Hyperdisk Storage Pools use thin provisioning and data reduction to help you store your data efficiently and achieve best in-class TCO.

Higher efficiency—Hyperdisk Storage Pools can take advantage of thin provisioning and data reduction to help you achieve higher resource utilization and lower TCO.

Less management overhead via Higher Flexibility—Disks in Hyperdisk Storage Pools can be provisioned to larger sizes and only use what they need, freeing workload owners from tedious capacity and performance forecasting, and from downtime related to rescaling.

Transparent to workloads—There is no change to how individual workloads use Hyperdisk volumes when using storage pools. There is no need for downtime or any other impact to workloads.

When to use storage pools

Difficulty planning the resource requirements when migrating workloads from on-premise workloads to Google Cloud.

Underutilization of resources

Complex management of the block storage used by your workloads

Hyperdisk Storage Pool features

- **Capacity and performance thin provisioning**: Capacity and performance are allocated as needed instead of allocating all the resources in advance. This helps to avoid low utilization rates of storage resources, where large amounts of disk space or performance are allocated, but not used.
- **Data reduction**: Storage pools use a variety of data reduction technologies to increase storage efficiency. Data reduction depends greatly on the type of data stored. Data that is already compressed or encrypted before it is stored in a disk in a Hyperdisk Storage Pool won't yield additional reduction.

How Hyperdisk Storage Pools work

- You create a storage pool with the aggregate capacity and performance that your workloads will later then create disks in the storage pool.
- You can then attach the disks to your VMs.
- When you create the disks, you can create them with a much larger size or provisioned performance limit than is needed.
- This simplifies planning and provides room for growth later, without necessitating changing the disk's provisioned size or performance at a later date.
- If your workloads grow and your disks need more capacity or performance, you can increase the provisioned capacity and performance of the storage pool.

Provisioning types for Hyperdisk Storage Pools

Standard capacity storage pools

- With Standard capacity provisioning, you create disks in the storage pool until the total provisioned capacity of all disks in the storage pool reaches the storage pool's provisioned capacity.

Advanced capacity storage pools

- Disks in an Advanced capacity storage pools consume capacity based only on the number of bytes written to your disks after data reduction
- The used capacity of the storage pool is defined by the amount of data written and not by the amount of provisioned disk capacity.
- You can fill disks in an Advanced capacity storage pool up to their provisioned size as long as the data written to all disks in the storage pool doesn't exceed the storage pool capacity.
- If the storage pool utilization reaches 80% of the pool provisioned capacity, the auto-grow feature attempts to automatically add capacity to the storage pool.

Types of Hyperdisk Storage Pools

- **Hyperdisk Throughput Storage Pool**: When creating the storage pool, you specify the capacity and throughput to provision for the storage pool. Each Hyperdisk Throughput disk you create in the storage pool uses some of the provisioned capacity and throughput.

- **Hyperdisk Balanced Storage Pool**: When creating the storage pool, you specify the capacity, throughput, and IOPS to provision for the storage pool. Each Hyperdisk Balanced disk that you create in the storage pool with provisioned capacity and performance above the baseline values uses some of the storage pool provisioned capacity and performance.

Limitations of storage pools

Resource limits:

- You can create a Hyperdisk Storage Pool with up to 1 PiB of provisioned capacity.
- You can create at most 10 storage pools per project.
- You can't change the provisioning model for a pool; you can't change a Standard capacity storage pool to an Advanced capacity storage pool or an Advanced performance storage pool to a Standard performance storage pool.
- Storage pools are a zonal resource.
- You can create up to 1,000 disks in a storage pool.
- You can use Hyperdisk Storage Pools with only Compute Engine. Cloud SQL instances cannot use Hyperdisk Storage Pools.

Limits for disks in a storage pool:

- Only new disks in the same project and zone can be created in a storage pool.
- Moving disks in or out of a storage pool is not permitted. To move a disk in or out of a storage pool, you have to recreate the disk from a snapshot
- To create boot disks in a storage pool, you must use a Hyperdisk Balanced Storage Pool.
- Storage pools don't support regional disks.
- You can't clone, create instant snapshots of, or configure Persistent Disk Asynchronous Replication for disks in a storage pool.
- Hyperdisk Balanced disks in a storage pool can't be attached to multiple compute instances.

OS images

- Operating System (OS) images used to create boot disks for your virtual machine (VM) instances

OS image types:

- **Public OS images** are provided and maintained by Google, open source communities, and third-party vendors. By default, all Google Cloud projects have access to these OS images and can use them to create VM instances.
- **Custom OS images** are available only to your Google Cloud project. You can create a custom OS image from boot disks and other images. Then, use the custom OS image to create VM instances.

Image families

- Image families help you manage images in your project by grouping related images together
- You can roll forward and roll back between specific image versions
- An image family always points to the latest version of an OS image that is not deprecated
- For example - the debian-11 image family in the debian-cloud project always points to the most recent Debian 11 image.

Custom image families

- If you regularly update your custom OS images with newer configurations and software, you can group those images into a custom image family.

Deprecation states

- **ACTIVE:** the image is active and can be used as normal. Image families point to the most recent and active image in a family.

- **DEPRECATED:** the image is marked deprecated but can still be used to create a VM. New links to this image are allowed. Image families no longer point to this image even if it is the most recent image in the family.

If you create a VM with a deprecated image using the Google Cloud CLI, the request succeeds with a warning.

- **OBSOLETE:** the image is marked obsolete and is no longer available for use. An error message is returned if you try to use this image in a request. Existing links to this image are still allowed.

- **DELETED:** this image is deleted. An error message is returned if you try to use a deleted image.

You can revert a deprecation (make an image active again), by changing the deprecation state to ACTIVE.

Create the image

- You can create disk images from the following sources:
- A persistent disk, even while that disk is attached to a VM
- A snapshot of a persistent disk
- Another image in your project
- An image that is shared from another project

gcloud command to create custom image

```
gcloud compute images create IMAGE_NAME \
   --source-disk=SOURCE_DISK \
   --source-disk-zone=ZONE \
   --family=IMAGE_FAMILY \
```

Snapshots

- Standard snapshots incrementally back up data on a disk.
- You can create snapshots from disks even while they are attached to running instances.
- Snapshots are global resources, so you can use them to restore data to a new disk or VM within the same project.
- You can also share snapshots across projects.
- The lifecycle of a snapshot created from a disk attached to a running VM instance is independent of the lifecycle of the VM instance.

Snapshot types

1. Standard
2. Instant
3. Archive

Retention after source disk deletion

- An instant snapshot of a disk only exists until the source disk is deleted.
- Standard and archive snapshots aren't deleted with the source disk. Therefore, if you want to retain a backup of a disk after you delete the disk itself, use archive or standard snapshots

Data recovery time

- The data recovery time is the length of time needed to create a new disk from a snapshot and varies by snapshot type.

- Instant snapshots offer the lowest and best recovery times.
- Standard snapshots have faster data recovery times than archive snapshots.
- Archive snapshots have the longest data recovery times, but offer the most cost efficient storage.

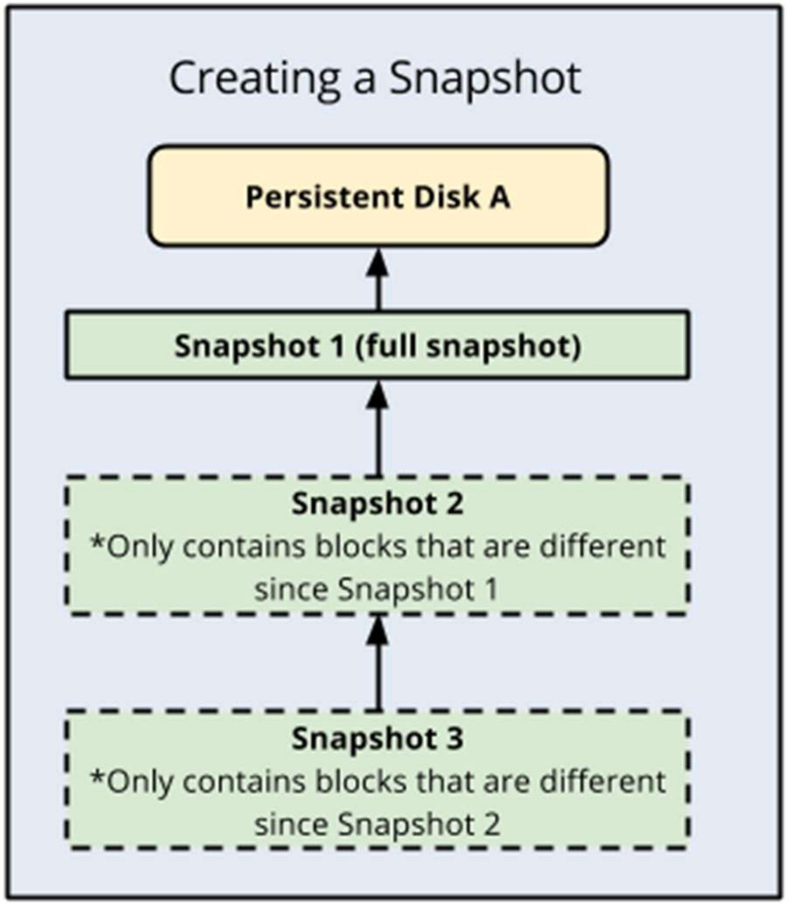
Storage location by snapshot type

- The storage location is the zone or region where Compute Engine stores the snapshot.

- Instant snapshots are local disk backups that are stored in the same zone or region as the source disk.
- Archive and standard snapshots are remote backups of disk data stored separately from the source disk.
- Copies of archive and standard snapshots are stored across multiple locations

Incremental snapshots work as follows:

- The first successful snapshot of a disk is a full snapshot that contains all the data on the disk.
- The second snapshot only contains any new data or modified data since the first snapshot. Data that hasn't changed since Snapshot 1 isn't included. Instead, Snapshot 2 contains references to Snapshot 1 for any unchanged data.
- Snapshot 3 contains any new or changed data since Snapshot 2 but won't contain any unchanged data from Snapshot 1 or 2. Instead, Snapshot 3 contains references to blocks in Snapshot 1 and Snapshot 2 for any unchanged data.


Creating a Snapshot
Persistent Disk A
Snapshot 1 (full snapshot)
Snapshot 2
*Only contains blocks that are different since Snapshot 1
Snapshot 3
*Only contains blocks that are different since Snapshot 2

Limitations

You can't create a snapshot of a Hyperdisk volume that's in multi-writer mode. Disable multi-writer mode for the disk, then create the snapshot.

You can't change the storage location of an existing standard snapshot.

You can snapshot a specific disk at most 6 times every 60 minutes.

You can't edit the data stored in a snapshot.

You can't recover deleted snapshots.

You can create an unlimited number of standard snapshots of a given disk.

gcloud command to create snapshot from disk

```
gcloud compute snapshots create SNAPSHOT_NAME \
   --source-disk-zone=SOURCE_ZONE \
   --source-disk=SOURCE_DISK_NAME \
   --snapshot-type=SNAPSHOT_TYPE \
   --storage-location=STORAGE_LOCATION
```

Key Management Service (KMS)

It allows you to create, import, and manage cryptographic keys and perform cryptographic operations in a single centralized cloud service

Types of Encryption:

- **Symmetric Key Encryption** – The same key is used for both encryption and decryption.
- **Asymmetric Key Encryption** – A pair of keys (public and private) is generated. The **public key** encrypts the data, while the **private key** decrypts it.