# GCP Professional Cloud Architect Crash Course

## Get Fully Prepared to Crush the Exam

**Victor Dantas**

Author, Certified Cloud Architect

Pearson

# Segment 1: Intro and overview of GCP exam

Objectives

- Introduction to the course
- Overview of GCP exam and what to expect

# Day 1 Schedule

- **Segment 1: Intro and overview of GCP exam, what to expect**
- **Segment 2: Designing a solution infrastructure that meets business requirements**
  - Business use cases and strategies
  - Success measurements
  - Movement of data and external integrations
  - Cost optimization
  - Supporting the application design
  - Creating a migration plan
  - Example: designing a solution infrastructure that meets business requirements
  - Break (10min)
- **Segment 3: Designing a solution infrastructure that meets technical requirements**
  - Designing for high availability and failover
  - Elasticity of cloud resources, quotas and limits
  - Designing for scalability
  - Designing for performance
  - Example: designing a solution infrastructure that meets technical requirements
  - Break (10min)

# Day 1 Schedule

- **Segment 4: Designing network, storage, and compute**
  - Integration with on-premises
  - Multicloud environments
  - Designing VPC networks
  - Choosing appropriate storage types
  - Choosing data processing technologies
  - Choosing compute resources
  - Break (10min)
- **Day 1 wrap-up**

# Audience Poll Question

What is your experience level with GCP?  (Single response)

- Basic knowledge of GCP but experienced with public cloud (AWS/Azure)
- Basic knowledge of GCP but experienced with private cloud / IT
- Foundational knowledge of GCP (< 6 months), no prior experience
- 6-months to 1-year working with GCP
- 1-year to 3-years working with GCP
- More than 3 years working with GCP

# Exam Overview

- **Length**: 2 hours
- **Registration fee**: $200 (plus tax where applicable)
- **Languages**: English, Japanese
- **Exam format**: Multiple choice and multiple select, taken remotely or in person at a test center.
- **Prerequisites**: None
- **Recommended experience**: 3+ years of industry experience including 1+ years designing and managing solutions using Google Cloud
- **Certification Validity**: Two years from the date of certification. Candidates must recertify in order to maintain their certification status.

# Exam Overview

A Google Cloud Certified Professional Cloud Architect:

- **Designs**, **develops**, and **manages** solutions that drive business objectives
- Is proficient in all aspects of enterprise **cloud strategy** and **architectural best practices**
- Is experienced in **software development methodologies**
- Is experienced with solutions that include **distributed applications** which span **multicloud** or **hybrid** environments

Source: https://cloud.google.com/certification/guides/professional-cloud-architect

# Exam Overview

**Four Case Studies:**

- **EHR Healthcare**
- **Helicopter Racing League**
- **TeramEarth**
- **Mountkirk Games**

**Each Case Study:**

- Company overview
- Solution concept
- Existing technical environment
- Business requirements
- Technical requirements
- Executive statement

https://cloud.google.com/certification/guides/professional-cloud-architect

# Why Get Certified? Why Google Cloud?

- Google Cloud strengths:
  - Focus on digital transformation (vs IaaS)
  - Innovation from Google
  - Multicloud and open-source friendly
  - High growth and momentum
- Why get GCP certified?
  - **GCP Cloud Solutions Engineer / Site Reliability Engineer** higher salary than equivalent roles for AWS and Azure*
  - **Google Certified Professional Cloud Architect** is a top-paying IT certification (#1 according to some rankings)

***based on data from PayScale***

# Segment 2: Designing a solution infrastructure that meets business requirements

Objectives

- Business use cases and strategies
- Success measurements
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

# Segment 2: Designing a solution infrastructure that meets business requirements

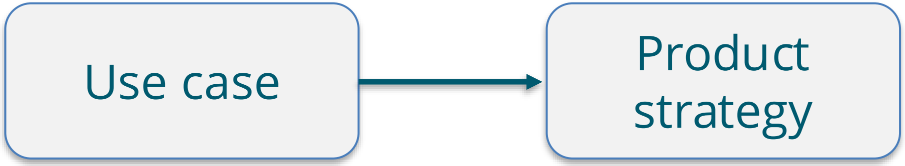
Objectives

- Business use cases and strategies
- Success measurements
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

# Business Use Cases and Strategies

A business **use case** identifies a customer need or a pain point

A **product strategy** translates a use case into a high-level plan (features and functionalities) of the product to achieve the business goal



# Business Use Cases and Strategies

A business **use case** identifies a customer need or a pain point

A **product strategy** translates a use case into a high-level plan (features and functionalities) of the product to achieve the business goal

Personalized product recommendations [to increase sales] → AI-powered recommendation system

**Key question**: What does success look like?

# Business Requirements

## Examples

| Requirement | What to think of |
|---|---|
| Reduce infrastructure administration costs | Managed services, serverless, automation, PaaS and SaaS |

# Business Requirements

## Examples

| Requirement | What to think of |
| --- | --- |
| Maintain high availability | Multi-AZ or Multi-region deployments. Managed services, serverless. Load balancers. |

# Business Requirements

## Examples

| Requirement | What to think of |
| --- | --- |
| Increase development agility | CI/CD, dev/prod environment parity, containers, infrastructure as code, blue/green and canary deployments. |

# Business Requirements

## Examples

| Requirement | What to think of |
| --- | --- |
| Increase ability to generate predictions and insights | Modern data warehousing, AI/ML, data pipelines, BI and data visualization tools. |

# Segment 2: Designing a solution infrastructure that meets business requirements

Objectives

- Business use cases and strategies
- Success measurements
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

# Success Measurements

**Business measurements**

- Total Cost of Ownership (TCO)
- Return on Investment (ROI)
- Agility (time from code to production)
- Key Performance Indicators (KPI)

**Technical measurements**

- Service availability
- Service response times
- Error rate
- Mean time to recovery (MTTR)

# Technical Metrics

**Relevant to users**

- Service availability
- Service response time
- Service error rate

**Irrelevant to users**

- Server availability
- Server CPU utilization
- Database replication lag

# Design Decision Trade-offs

**Almost every architecture decision is a trade-off**

Performance vs Cost

Security vs Flexibility

Reliability vs Agility

# Design Decision Trade-offs

**Some decisions are based on a priority.**

- Example: Your organization may place very high priority on security.

> The solution's design may sacrifice some agility in favor of security by e.g., introducing guardrails and policies.

# Design Decision Trade-offs

**Some decisions are just implicitly better.**

- Example: Your organization wants to set up a development environment that is expected to be used only during business days.

> A solution may include always-on VMs. Another solution may include VMs with a shutdown schedule. Both solutions meet the requirements, but one is cheaper.

# Segment 2: Designing a solution infrastructure that meets business requirements

Objectives

- Business use cases and strategies
- Success measurements
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

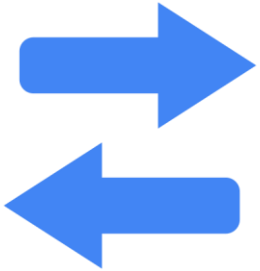
# Movement of Data: Options

- Storage Transfer Service
- Storage Transfer Appliance
- gsutil
- Database Migration Service
- BigQuery Data Transfer Service

# Movement of Data: Storage Transfer Service

Transfer data securely between **object** and **file** storage across Google Cloud, AWS, Azure, and on-premises.

- Encrypts data in transit and uses checksums for data integrity checks
- Does incremental transfer
- Can set up a repeating schedule for transferring data



**Scenario: less than 1TB from other clouds or on-premises**

# Movement of Data: Storage Transfer Appliance

- High-capacity, ruggedized, tamper-resistant storage device
- Ship your data to a Google upload facility
  - Data is uploaded to **Cloud Storage**
- Data is encrypted with AES 256 encryption

Request appliance → Upload data → Ship appliance back → Google uploads data

**Scenario: offline very large-scale transfer**

# Movement of Data: Transferring Small Datasets

- For small transfers (<1TB and enough bandwidth), can use **gsutil** tool
  - For multi-threaded transfers, use **gsutil –m**
  - For a single large file, use **Composite transfers**

**Scenario: small (<1TB) uploads**

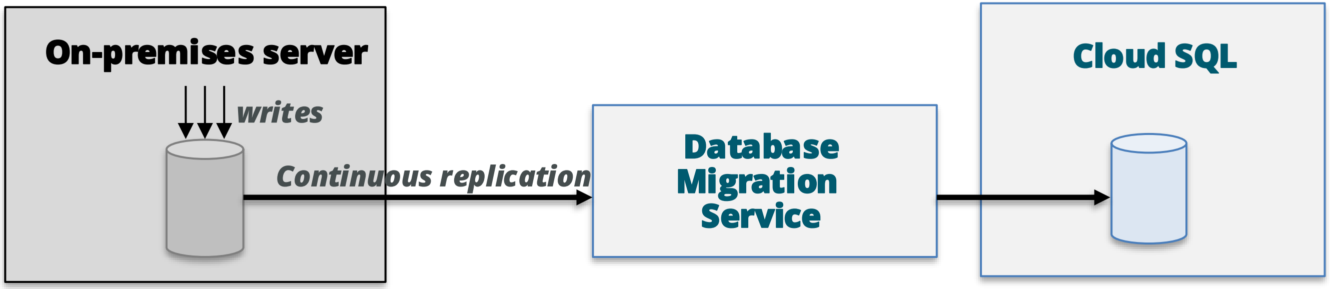
# Movement of Data: Database Migration Service

Migrate databases to Cloud SQL or AlloyDB

- Serverless and guided experience
- Can migrate from on-premises, GCE, and other clouds
- Available for MySQL, PostgreSQL, Oracle, and SQL Server

# Movement of Data: Database Migration Service

Replicate data continuously for minimal downtime migrations



# Movement of Data: BigQuery Data Transfer Service

Automates data movement into BigQuery on a schedule

- Currently can only be used to transfer data into BigQuery
- Supported sources:
  - Cloud Storage
  - Amazon S3
  - Teradata
  - Amazon Redshift
  - Google SaaS apps (Google Ads, Google Play, etc.)
  - Several third-party transfers available in Google Cloud Marketplace

## Segment 2: Designing a solution infrastructure that meets business requirements

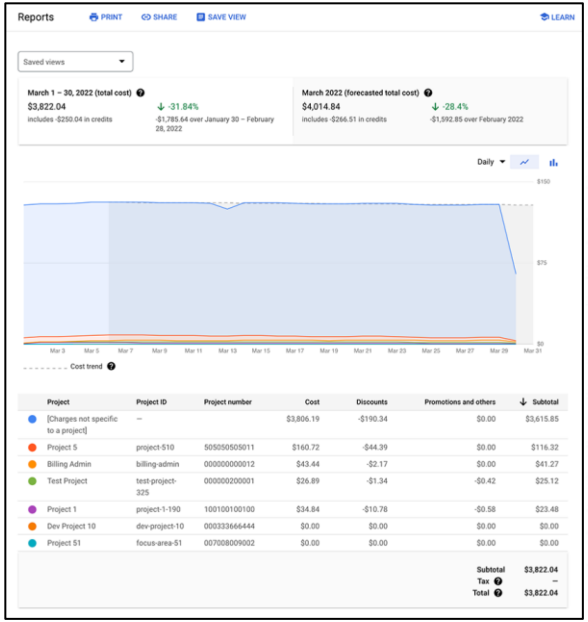
Objectives

- Business use cases and strategies
- Success measurements
- Buy vs build
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

# Cost Optimization Best Practices

## Leverage billing and cost management tools

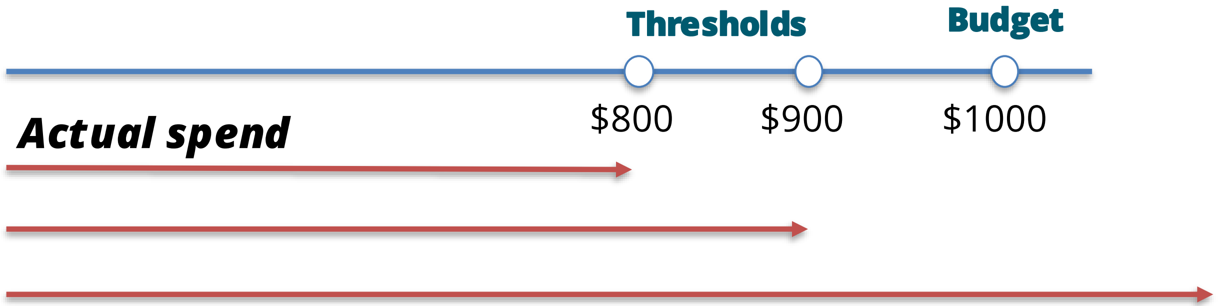
- Analyze billing reports
- Use tags to attribute costs back to departments/teams
- Export Cloud Billing data to BigQuery
- Use quotas, budgets, and alerts to closely monitor cost trends and forecast costs over time



# Budgets and Alerts: Example

**Budget**: $1000

**Budget alerts**: 80%, 90%




***! Notification***

***! Notification***

Pearson

# Cost Optimization Best Practices

- Don’t pay for resources you don’t use
  - Identify idle VMs (tip: use **Recommender** service and the Idle Resource Recommender)
  - Schedule VMs to auto start and stop (with **instance schedule**)
- Rightsize VMs
  - Leverage custom machine types
  - Apply machine type recommendations
- Leverage **Spot** VMs for fault-tolerant workloads
- Leverage committed use discounts
  - Ideal for workloads with predictable resource needs, available as 1- or 3-year term(s).

# Cost Optimization Best Practices

## Optimize Cloud Storage costs

- Leverage storage classes
- Leverage lifecycle policies

| Storage class | Minimum duration | Typical monthly availability |
|---|---|---|
| Standard Storage | None | >99.99% in multi-regions and dual-regions<br>99.99% in regions |
| Nearline Storage | 30 days | 99.95% in multi-regions and dual-regions<br>99.9% in regions |
| Coldline Storage | 90 days | 99.95% in multi-regions and dual-regions<br>99.9% in regions |
| Archive Storage | 365 days | 99.95% in multi-regions and dual-regions<br>99.9% in regions |

Retrieval Cost ↓

Storage Cost ↑

Pearson

# Cost Optimization Best Practices

- Optimize networking costs
  - Avoid cross-region data processing
  - Compress data output prior to egress
  - Use Cloud CDN to reduce traffic volume

# Cost Optimization Best Practices

Tune BigQuery

- Enforce controls to limit query costs
- Use partitioning and clustering
- Checking for unnecessary streaming inserts (use batch loading instead, it’s free)

# Segment 2: Designing a solution infrastructure that meets business requirements

Objectives

- Business use cases and strategies
- Success measurements
- Buy vs build
- Movement of data and external integrations
- Cost optimization
- Creating a migration plan

# Cloud Migration Strategies

Lift and Shift

Improve and Move

Remove and Replace

Time and effort →

Future ROI →

# Migration Phases

## Assess

Thorough discovery of existing environment

Identifying app dependencies and requirements

TCO Calculations

Performance benchmarks

## Plan

Foundational cloud infrastructure

Identity management

Organization and project structure

Networking

## Deploy

Implement and execute a deployment process

Move workloads

Refine cloud infrastructure

## Optimize

Adopt cloud-native Technologies

Scalability, DR

Cost optimization

Training

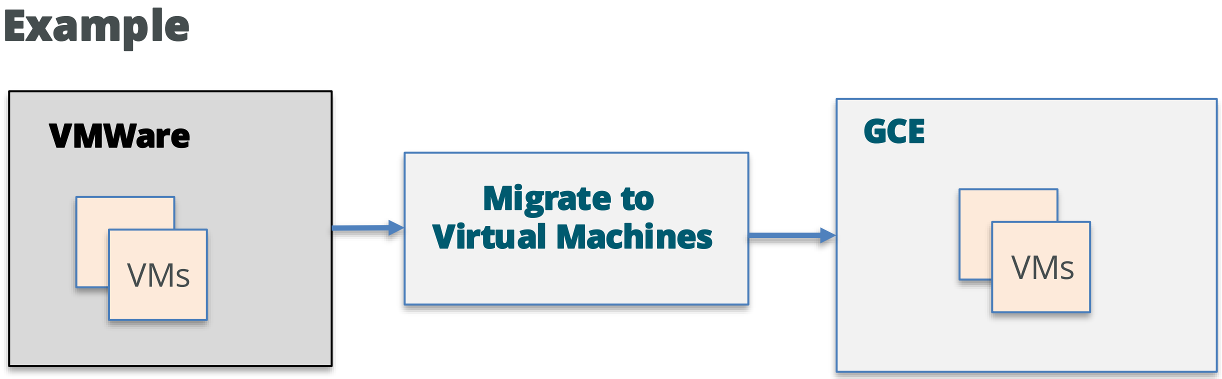
AI/ML and insights

# Good Candidates to Migrate First

- ✓ Not business critical
- ✓ Requires minimal app changes
- ✓ Doesn’t need large volume of data
- ✓ No strict compliance requirements
- ✓ Doesn't require third-party proprietary licenses

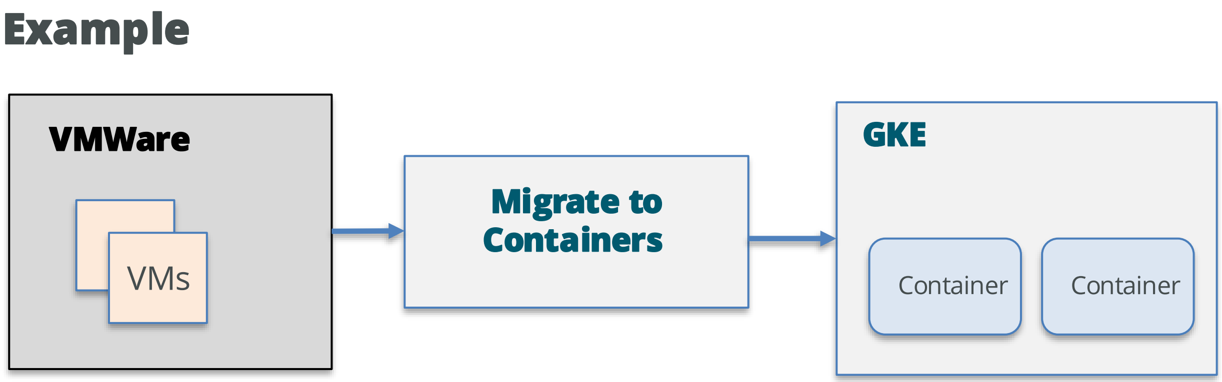
# Migrating Applications: Migrate to Virtual Machines

- Enables "lift and shift" migrations, with minor automatic modifications
- Uses data replication technology for disk data

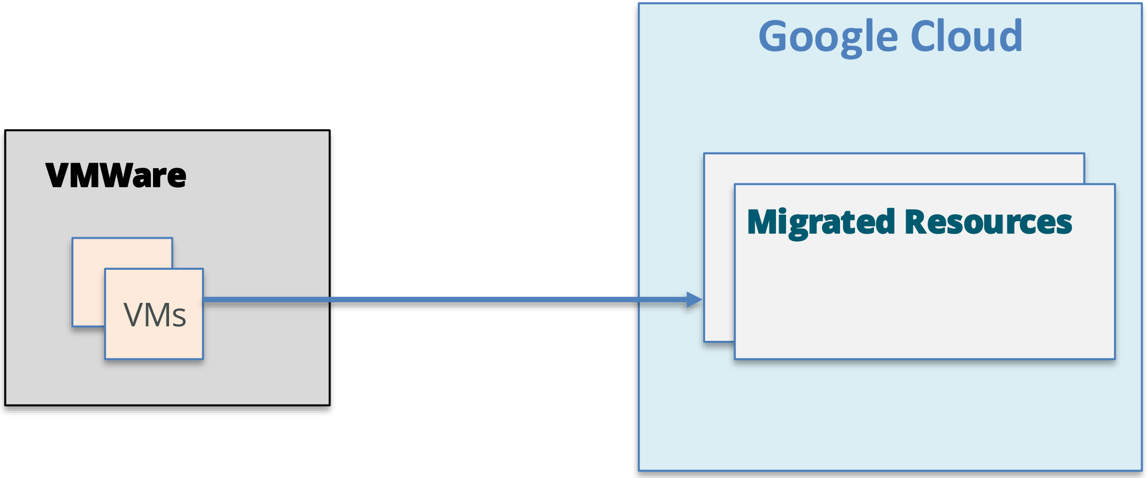


# Migrating Applications: Migrate to Containers

Use **Migrate to Containers** to convert VM-based workloads into containers in GKE or Cloud Run



# Authentication for On-Premises Workloads



# Authentication for On-Premises Workloads

## Service Account Keys

- RSA key pairs
- Lets you authenticate as the service account by having access to the private key
- User-managed key pairs are a security risk

## Workload Identity Federation

- Identity federation with AWS, Azure, or any identity provider that supports OpenID Connect / SAML 2.0.
- Use IAM to grant external identities IAM roles, including ability to impersonate service accounts
- No need to maintain service account keys

**PREFERRED**

# Example: Designing a Solution Infrastructure that Meets Business Requirements

## Business Requirements

i. Provisioning of cloud infrastructure resources must be auditable
ii. Changes to cloud infrastructure must go through a review process that minimizes impact to development velocity
iii. Data cannot traverse the public internet
iv. Data must be encrypted in transit and at rest
v. A failure in one part of the system should not bring down the entire system

## Technical Requirements

i. Provisioning must be done through Infrastructure-as-Code process
ii. Tools for code review and approvals must be in place for infrastructure provisioning.
iii. All API access should be private and use of external IP addresses restricted
iv. All API communications over HTTPS, enforced SHA-256 encryption on all data stores
v. Microservices architecture

# Questions Breakdown

You are responsible for planning a migration of your company's workloads to Google Cloud. What actions should you do first?

A. Run a thorough discovery and assessment of the current environment, identify app dependencies, and calculate total cost of ownership (TCO).

B. Set up the foundational infrastructure on GCP, decide which workload to migrate first, and run a proof-of-concept.

C. Run a performance benchmark for your existing applications, codify the target infrastructure, and deploy a landing zone.

D. Define the project structure on GCP, educate the team on cloud-native technologies, and modernize all applications ahead of the migration.

# Questions Breakdown

You are responsible for **planning a migration** of your company's workloads to Google Cloud. What actions should you do **first**?

A. Run a thorough discovery and assessment of the current environment, identify app dependencies, and calculate total cost of ownership (TCO).
B. Set up the foundational infrastructure on GCP, decide which workload to migrate first, and run a proof-of-concept.
C. Run a performance benchmark for your existing applications, codify the target infrastructure, and deploy a landing zone.
D. Define the project structure on GCP, educate the team on cloud-native technologies, and modernize all applications ahead of the migration.

# Questions Breakdown

Your company wants to migrate an on-premises MySQL deployment to a managed offering on Google Cloud. You need to minimize downtime and performance impact during the migration. Which approach should you recommend?

A. Use MySQL tools to create a dump of the existing server, then shut down the server, upload the dump file to Cloud Storage, and load it into a new Cloud SQL instance.

B. Provision a Google Compute Engine (GCE) virtual machine and install MySQL. Set up replication on the on-premises server until cut-over.

C. Provision a Cloud SQL for MySQL instance. Import data using the latest MySQL database backup.

D. Provision a Cloud SQL for MySQL instance. Use the Database Migration Service to set up continuous data replication until cut-over.

# Questions Breakdown

Your company wants to migrate an on-premises **MySQL** deployment to a **managed offering** on Google Cloud. You need to **minimize downtime** and performance impact during the migration. Which approach should you recommend?

A. Use MySQL tools to create a dump of the existing server, then shut down the server, upload the dump file to Cloud Storage, and load it into a new Cloud SQL instance.

B. Provision a Google Compute Engine (GCE) virtual machine and install MySQL. Set up replication on the on-premises server until cut-over.

C. Provision a Cloud SQL for MySQL instance. Import data using the latest MySQL database backup.

D. Provision a Cloud SQL for MySQL instance. Use the Database Migration Service to set up continuous data replication until cut-over.

# Questions Breakdown

You are responsible for migrating VMware-based workloads to Google Cloud. Your company's CTO decided to adopt container technologies and modernize applications away from VMs and into native containers. You need to plan a solution for migrating VMs to GCP while minimizing downtime. Which approach should you recommend?

A. Create a virtual machine image from each existing VM and use a third-party tool to convert them to Docker images. Deploy the Docker images into a Google Kubernetes Engine (GKE) cluster.

B. Upload all application source codes to Cloud Source Repositories. Use Cloud Build to create a continuous deployment pipeline that deploys the source code into Cloud Run.

C. Run a discovery exercise to identify existing workloads and assess migration readiness. Configure the deployment environment for the migrated containers, and use Anthos Migrate to Containers.

D. Use Migrate for Compute Engine to first migrate applications to virtual machines on GCP. Use Anthos Migrate to Containers to migrate all VMs to containers.

# Questions Breakdown

You are responsible for migrating **VMware-based workloads to Google Cloud**. Your company's CTO decided to **adopt container** technologies and modernize applications **away from VMs** and into native containers. You need to plan a solution for migrating VMs to GCP while **minimizing downtime**. Which approach should you recommend?

A. Create a virtual machine image from each existing VM and use a third-party tool to convert them to Docker images. Deploy the Docker images into a Google Kubernetes Engine (GKE) cluster.

B. Upload all application source codes to Cloud Source Repositories. Use Cloud Build to create a continuous deployment pipeline that deploys the source code into Cloud Run.

C. Run a discovery exercise to identify existing workloads and assess migration readiness. Configure the deployment environment for the migrated containers, and use Anthos Migrate to Containers.

D. Use Migrate for Compute Engine to first migrate applications to virtual machines on GCP. Use Anthos Migrate to Containers to migrate all VMs to containers.

# Segment 3: Designing a solution infrastructure that meets technical requirements

Objectives

- Designing for high availability and failover
- Elasticity of cloud resources, quotas and limits
- Designing for scalability
- Designing for performance

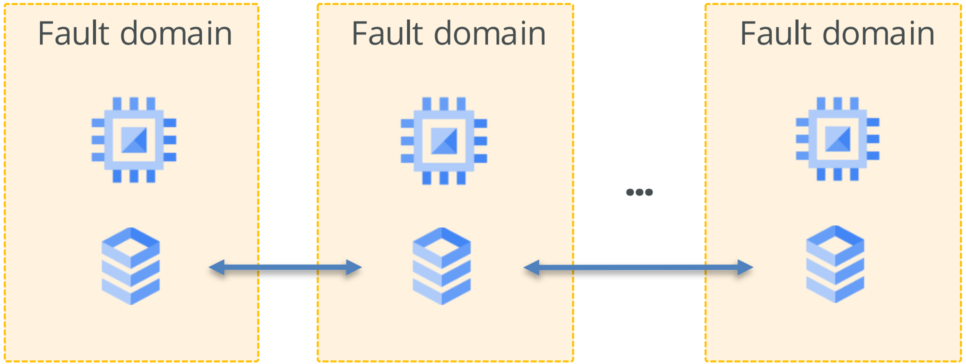
# Segment 3: Designing a solution infrastructure that meets technical requirements

Objectives

- Designing for high availability and failover
- Elasticity of cloud resources, quotas and limits
- Designing for scalability
- Designing for performance

# High availability (HA) Design Principles

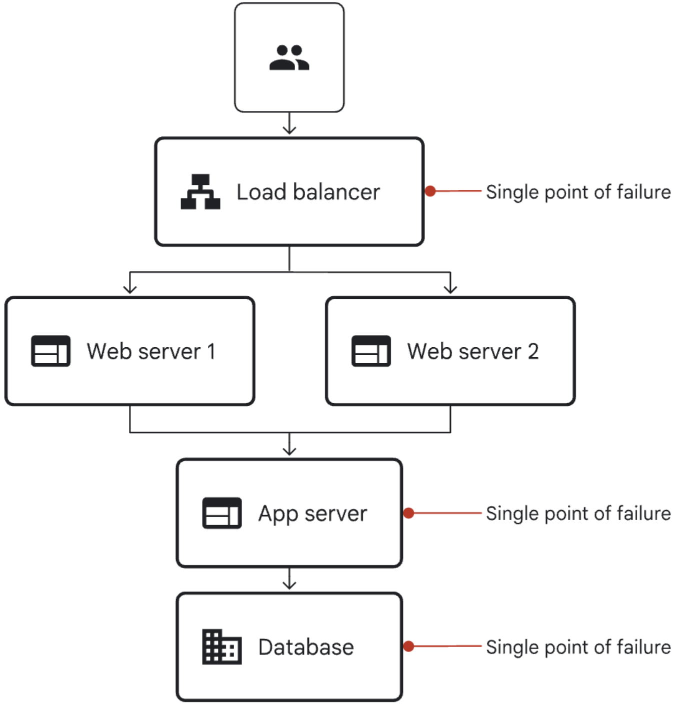
- Create redundancy and eliminate single points of failure
- Deploy resources across multiple fault domains
- Leverage load balancing and autoscaling




Pearson

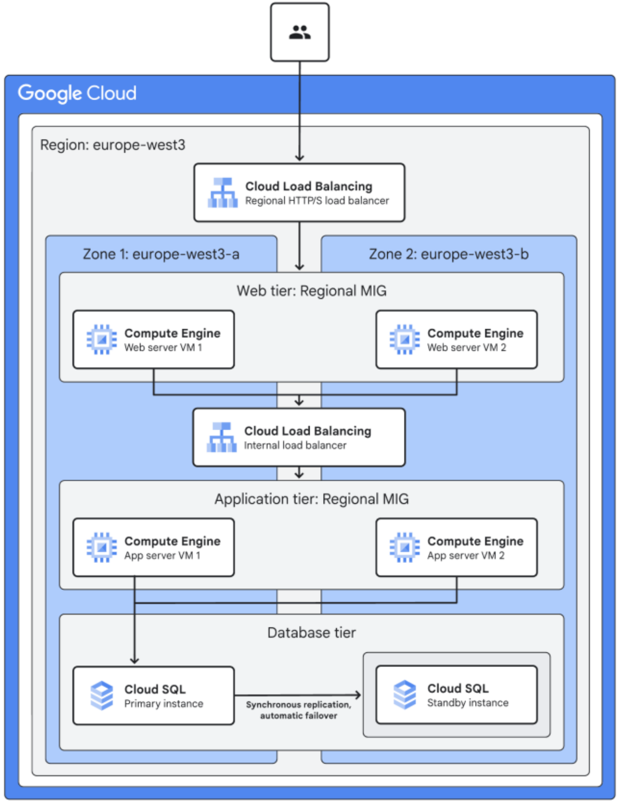
# High Availability

Avoid single point of failures!




Pearson

# Multi-zone HA Architecture



# Leverage Managed Services

- Most managed services are regional (i.e., resilient against zone outages)
- Some are global or multi-region (i.e., resilient against regional outages), for example:

Cloud Storage

Cloud Spanner

BigQuery

Cloud Firestore

HTTP(S) Load Balancer

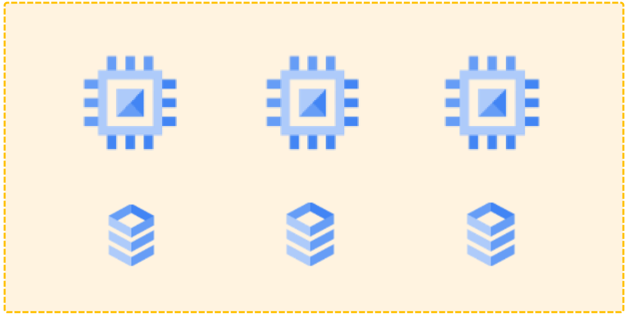
# Segment 3: Designing a solution infrastructure that meets technical requirements

Objectives

- Designing for high availability and failover
- Elasticity of cloud resources, quotas and limits
- Designing for scalability
- Designing for performance

# Elasticity

- Does the solution scale when busy, so that the service's availability and performance remain intact?
- Does the solution scale back when demand is low, so that infrastructure cost is reduced?



# Design Strategies for Elasticity

- Autoscaling and load balancing
- Statelessness
- Managed services (serverless)

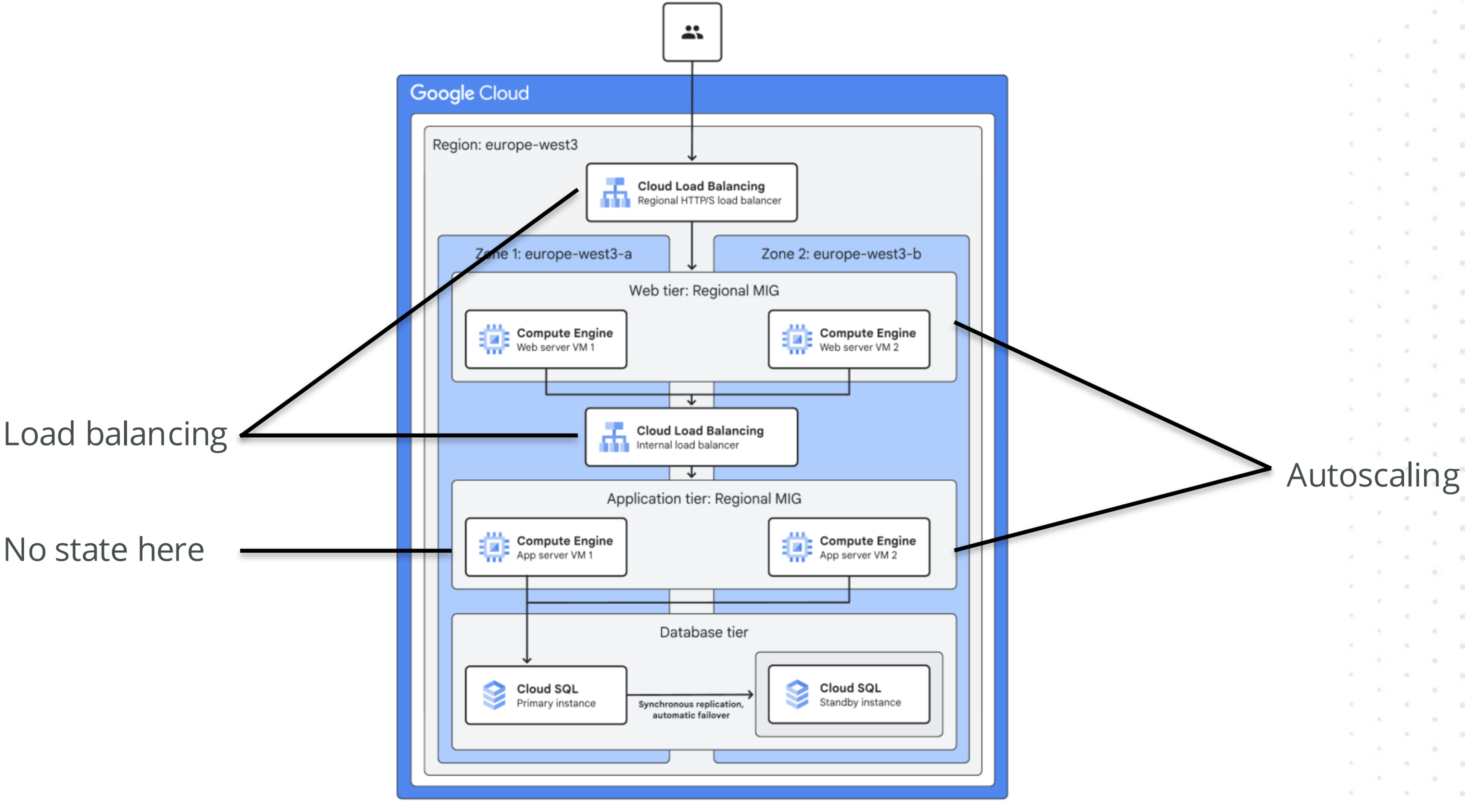
# Managed Instance Groups (MIGs)

- Deploy stateless identical instances based on **instance template**
- **Autoscaling**, autohealing, and rolling update capabilities
- Can be single zone or regional

Similar to…

AWS: Auto Scaling group
Azure: Virtual Machine Scale Set

# Elastic Architecture




Pearson

# Autoscaling and Load Balancing: GKE

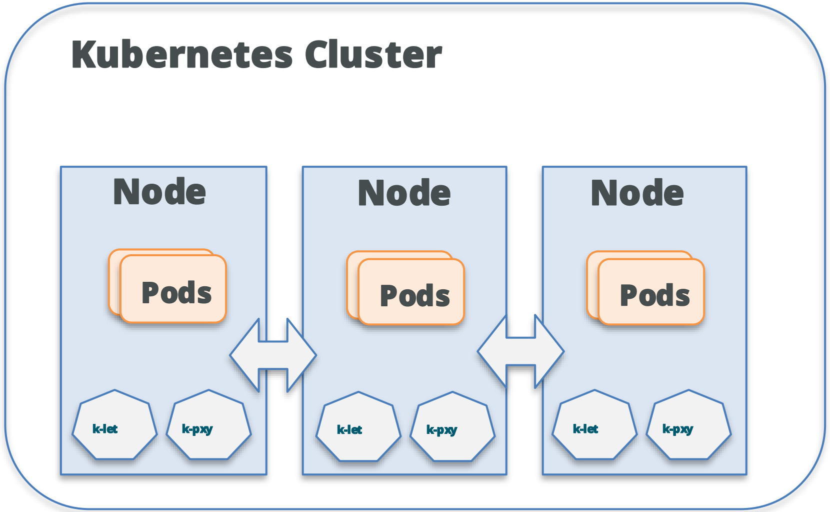
**Kubernetes Engine containers: GKE Autoscaling**

- Pod autoscaling
- Cluster autoscaling

# Autoscaling and Load Balancing

## GKE Autoscaling: Cluster autoscaling

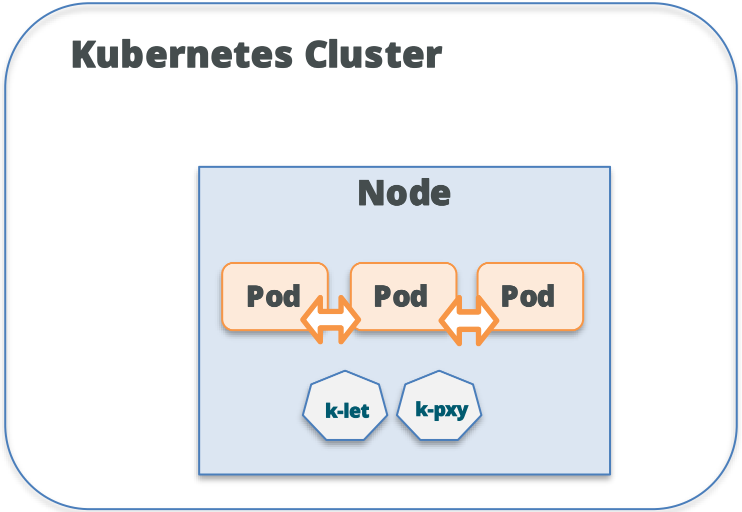
- Automatically resizes the number of nodes in a given node pool based on the demands of workloads.
- Automatically distributes nodes across zones



# Autoscaling and Load Balancing

## GKE Autoscaling: Pod autoscaling

- Automatically scale Pods
- Automatically distributes Pods across nodes




Will **NOT** improve resilience against zonal failures

# Quotas and Limits

**GCP-enforced quotas**

- Used to restrict how much of a particular shared Google Cloud resource you can use
- Enforced to protect community from unforeseen spikes in usage and overloaded services

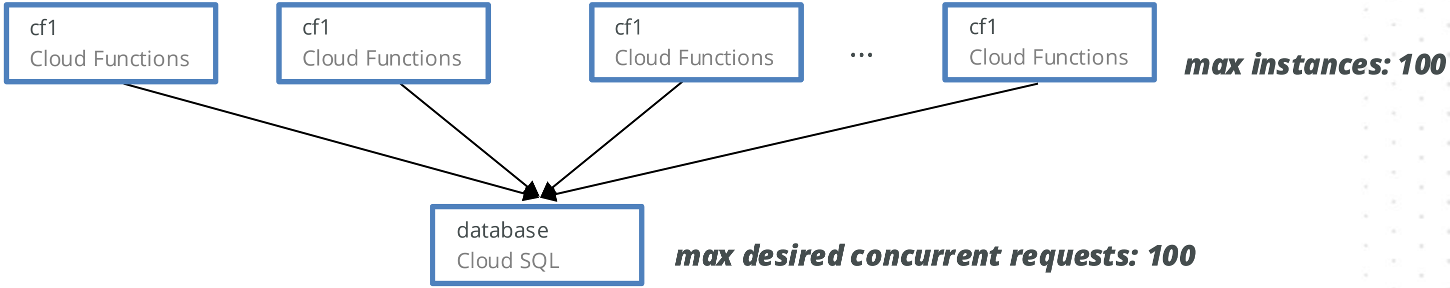
**User-enforced quotas**

- You can set your own limits on service usage to avoid unexpected bills

# Quotas and Limits

Scale without hitting limits:

- Example: Limit concurrent connections to Cloud SQL database
- With Cloud Run, use **max instances** setting to limit how many concurrent instances of the function are running and establishing database connections



# Segment 3: Designing a solution infrastructure that meets technical requirements

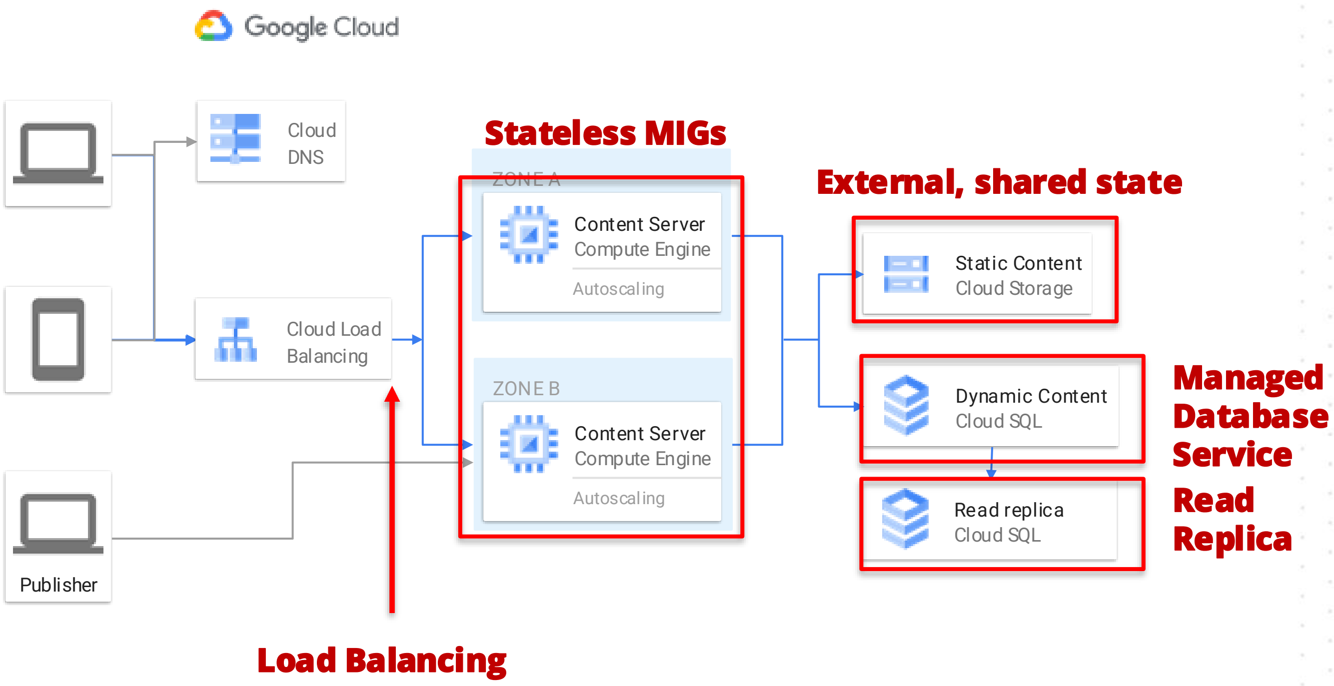
Objectives

- Designing for high availability and failover
- Elasticity of cloud resources, quotas and limits
- Designing for scalability
- Designing for performance

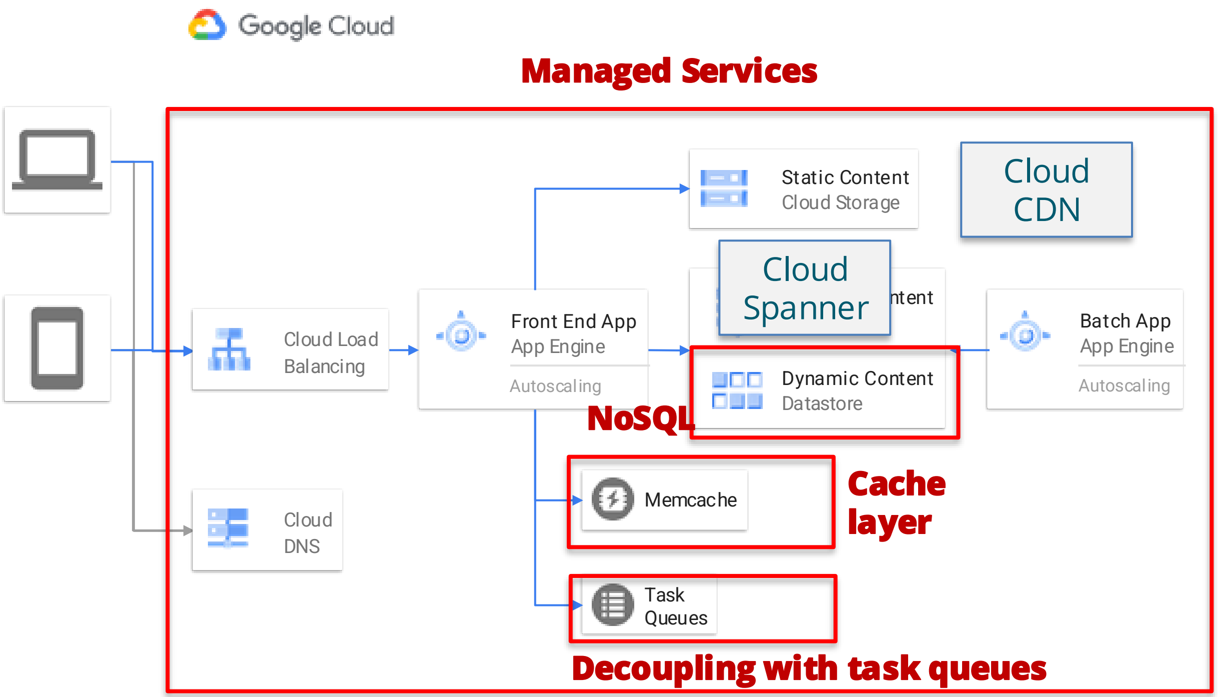
# Scalability for Growth

- More than just autoscaling to cover temporary demand fluctuations.
- Think about scalable, cloud-native design patterns:
  - Aim for **statelessness** and leverage **managed services**
  - **Decouple** architectures into modular components or tiers. Load-balance at each tier. Leverage **task queues**.
  - Leverage **NoSQL databases** for non-relational / non-transactional data
  - Leverage **read replicas** and **caching**
  - Automate and apply **continuous delivery** to make frequent, incremental updates

# Scalable Architecture Example #1



# Scalable Architecture Example #2




**More scaling opportunities?**

# Segment 3: Designing a solution infrastructure that meets technical requirements

Objectives

- Designing for high availability and failover
- Elasticity of cloud resources, quotas and limits
- Designing for scalability
- Designing for performance

# Performance Optimization Best Practices

- Monitor and analyze application performance
- Architect workloads for optimal resource placement
- Implement content caching
- Isolate read and write traffic

# Performance Optimization Design Patterns

- Autoscale resources
- Use GPUs/TPUs for AI/ML or graphics-intensive workloads
- Pick the appropriate machine family for the workload
- Cache frequently accessed data
- Consider SSD storage (vs HDD)
- Deploy application close to your users

# Example: Designing a Solution Infrastructure that Meets Technical Requirements

## Technical Requirements

i. Provisioning must be done through Infrastructure-as-Code process
ii. Tools for code review and approvals must be in place for infrastructure provisioning.
iii. All API access should be private and use of external IP addresses restricted
iv. All API communications over HTTPS, enforced SHA-256 encryption on all data stores
v. Microservices architecture

## Design and Implementation

i. Terraform and Cloud Build. Only Cloud Build pipelines can deploy resources.
ii. Cloud Source Repositories, one source repo per product/service, pull request mechanism enforced
iii. Private Google Access, Organization Service Policy constraints
iv. Cloud-first development leveraging Google’s default encryption at rest and in transit..
v. Kubernetes Enterprise (fka Anthos)

# Questions Breakdown

Your company has a web-based application hosted in a single data center. As the customer base grows, customers from distant locations often complain that the website is slow. Your company has decided to move the application to Google Cloud to benefit from its global footprint.

How should you design the solution to reduce latency for customers?

A. Deploy the application to a set of virtual machines and use DNS-based load balancing
B. Use zonal managed instance groups in different zones with a regional load balancer
C. Use regional managed instance groups in different regions with a global load balancer
D. Deploy the application to a regional App Engine instance with a global load balancer.

# Questions Breakdown

Your company has a web-based application hosted in a **single data center**. As the customer base grows, customers from distant locations often complain that the **website is slow**. Your company has decided to move the application to Google Cloud to benefit from its **global footprint**.

How should you design the solution to **reduce latency** for customers?

A. Deploy the application to a set of virtual machines and use DNS-based load balancing

B. Use zonal managed instance groups in different zones with a regional load balancer

C. Use regional managed instance groups in different regions with a global load balancer

D. Deploy the application to a regional App Engine instance with a global load balancer.

# Questions Breakdown

Your company has acquired another company that has a containerized web application running on-premises. You need to move the application to Google Cloud and redesign the solution to accommodate a larger number of users and scale automatically with usage. What should you do?

A. Host the application on Google Kubernetes Engine and enable Horizontal Pod Autoscaler and cluster autoscaling

B. Host the application on Google Kubernetes Engine and enable Vertical Pod Autoscaler and cluster autoscaling

C. Host the application on Compute Engine instances. Perform a load test and use Google Cloud’s machine type recommender to identify the most appropriate machine type

D. Host the application on a managed instance group with an autoscaling policy. Use Google Cloud’s managed instance group machine type recommender to identify the most appropriate machine type

# Questions Breakdown

Your company has acquired another company that has a **containerized** web application running on-premises. You need to move the application to Google Cloud and redesign the solution to accommodate a **larger number of users** and **scale automatically** with usage. What should you do?

A. Host the application on Google Kubernetes Engine and enable Horizontal Pod Autoscaler and cluster autoscaling
B. Host the application on Google Kubernetes Engine and enable Vertical Pod Autoscaler and cluster autoscaling
C. Host the application on Compute Engine instances. Perform a load test and use Google Cloud's machine type recommender to identify the most appropriate machine type
D. Host the application on a managed instance group with an autoscaling policy. Use Google Cloud's managed instance group machine type recommender to identify the most appropriate machine type

# Questions Breakdown

Your company is running a stateless web application on two Compute Engine instances in two availability zones. The application receives a lot of traffic during business hours and little traffic otherwise. During peak hours, several users are complaining that the application is slow and sometimes crashing. You need to redesign the solution to improve performance. What should you do?

A. Deploy the application to two more instances in a separate Google Cloud region

B. Deploy the application to an extra instance in a new availability zone. Configure startup and shutdown scripts so that the extra instance only runs during business hours.

C. Set up a Cloud Monitoring alert that triggers a Cloud Function to create a new instance if the average CPU utilization is high.

D. Create an instance template and deploy the application to a managed instance group with an autoscaling policy.

# Questions Breakdown

Your company is running a **stateless** web application on two Compute Engine instances in two availability zones. The application receives **a lot of traffic during business hours** and little traffic otherwise. **During peak hours**, several users are complaining that the **application is slow and sometimes crashing**. You need to redesign the solution to **improve performance**. What should you do?

A. Deploy the application to two more instances in a separate Google Cloud region

B. Deploy the application to an extra instance in a new availability zone. Configure startup and shutdown scripts so that the extra instance only runs during business hours.

C. Set up a Cloud Monitoring alert that triggers a Cloud Function to create a new instance if the average CPU utilization is high.

D. Create an instance template and deploy the application to a managed instance group with an autoscaling policy.

# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

# Segment 4: Designing network, storage, and compute

Objectives

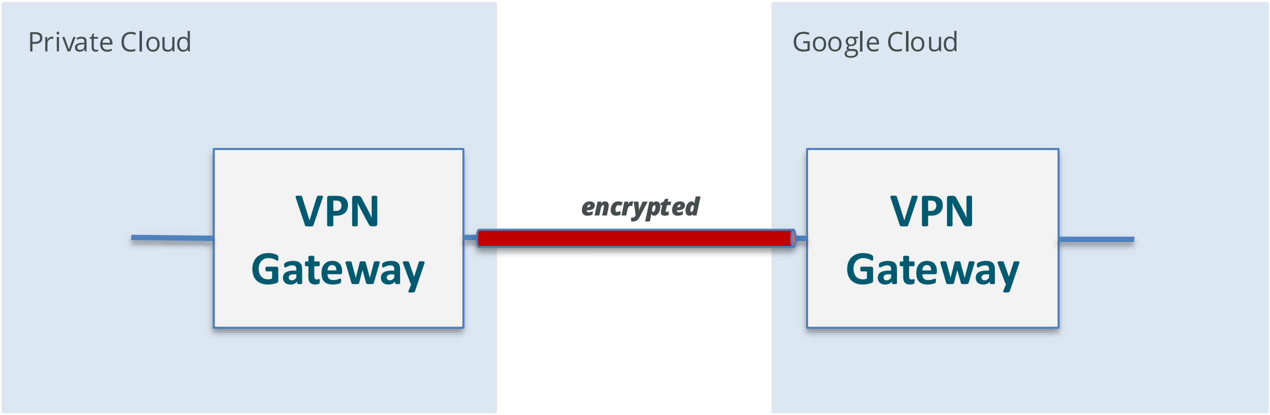
- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

# Hybrid Networking Services

| | | |
|---|---|---|
| Cloud VPN | Cloud Interconnect | Network Connectivity Center |

# Hybrid Networking Services: Cloud VPN

- IPSec VPN tunnel
- Traffic travels over the public internet, encrypted by one VPN gateway, then decrypted by another VPN gateway



# Integration with On-Premises: Cloud VPN

Two types of Cloud VPN:

- HA VPN: High availability VPN (99.99% SLA) solution
- Classic VPN: 99.9% availability SLA (partially deprecated)

# Integration with On-Premises: Cloud VPN

Two types of Cloud VPN:

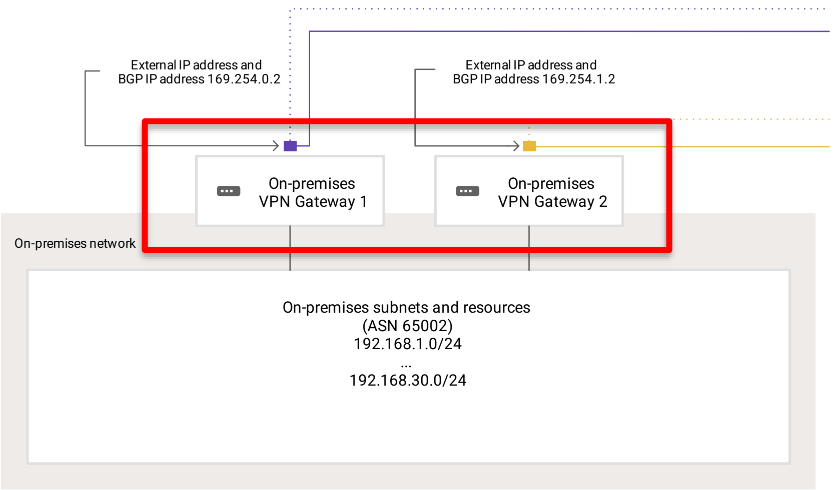
- HA VPN: High availability VPN (99.99% SLA) solution
- Classic VPN: 99.9% availability SLA (partially deprecated)

**Recommended configuration**

Two interfaces, two external IPV4 addresses

# High Availability VPN Requirements

- 99.99% SLA is guaranteed on Google Cloud side only
- For end-to-end 99.99% availability:




VPN device configured with adequate redundancy

Configure two tunnels

# Cloud Interconnect

## Dedicated Interconnect

- Direct connection to Google's network with end-to-end SLA
- Must be able to physically meet Google's network
- **10-Gbps** or **100-Gbps** circuits with flexible VLAN attachment capacities from 50 Mbps to 50 Gbps.
- Maximum of 8x10Gbps (**80Gbps** aggregate bandwidth) or 2x100Gbps (**200Gbps** aggregate bandwidth) circuits

## Partner Interconnect

- Traffic passes through service provider's network, but **not** the public internet
- More points of connectivity through one of the supported service providers
- Flexible VLAN attachment capacities from **50Mbps** to **50Gbps**
- Google provides SLA for Google-Partner connection

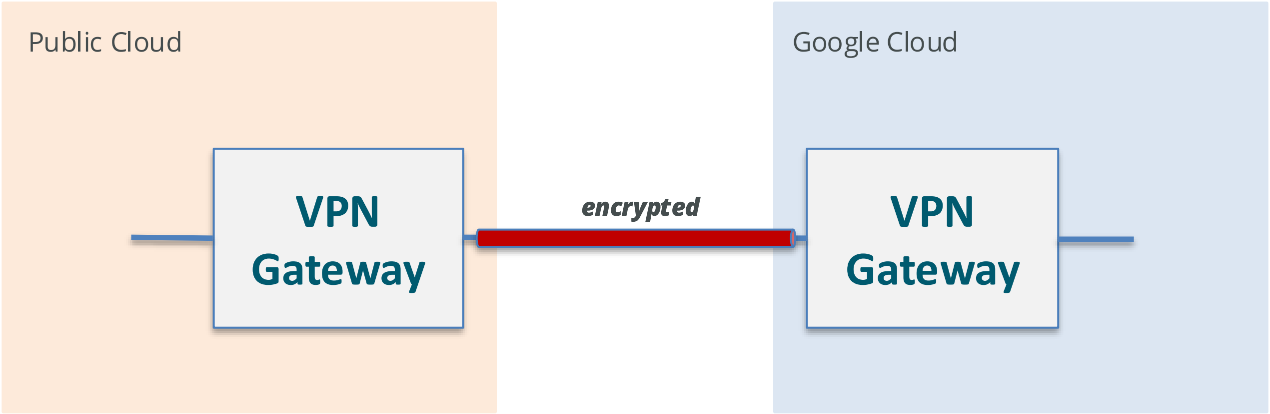
# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

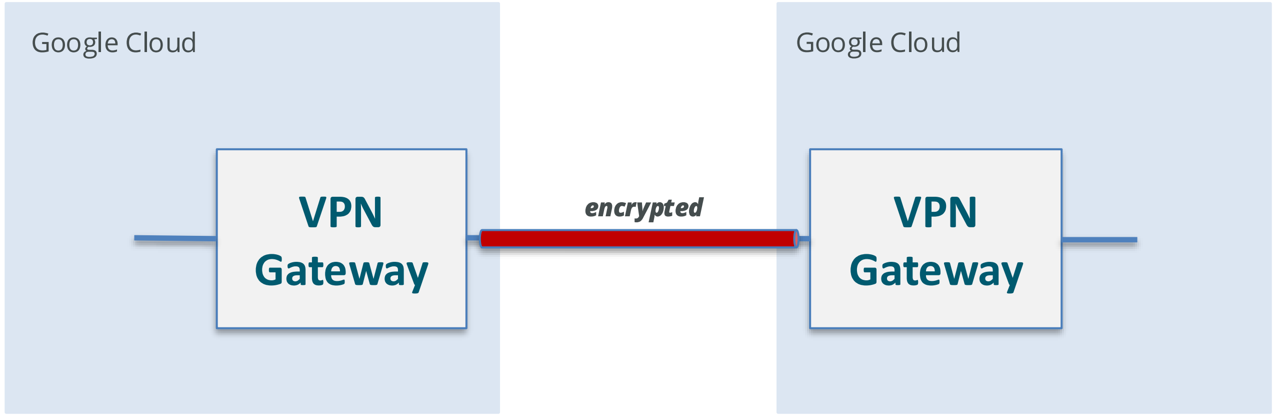
# Multicloud Networking

- You can connect a Google Cloud VPC network to another cloud provider's network (AWS, Azure, etc.)



# Multicloud Networking

- You can connect a Google Cloud VPC network to another cloud provider's network (AWS, Azure, etc.)
- You can also connect two Google Cloud VPC networks, whether they belong to the same organization or not

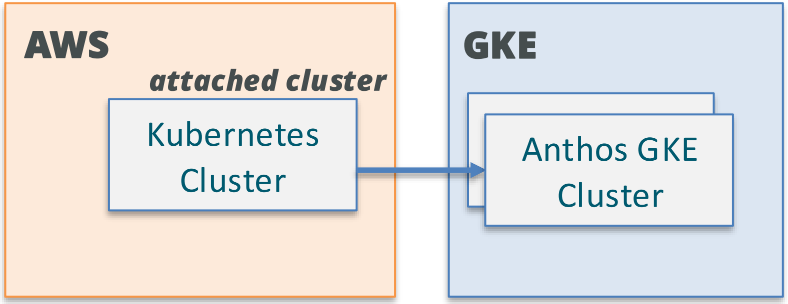



Pearson

# Multicloud Solutions: GKE Multi-cloud (Anthos)

GKE Multi-cloud (fka Anthos) enables you to manage **GKE clusters** and workloads running on **virtual machines** across environments.

**Consistent managed Kubernetes experience** with upgrades validated by Google.



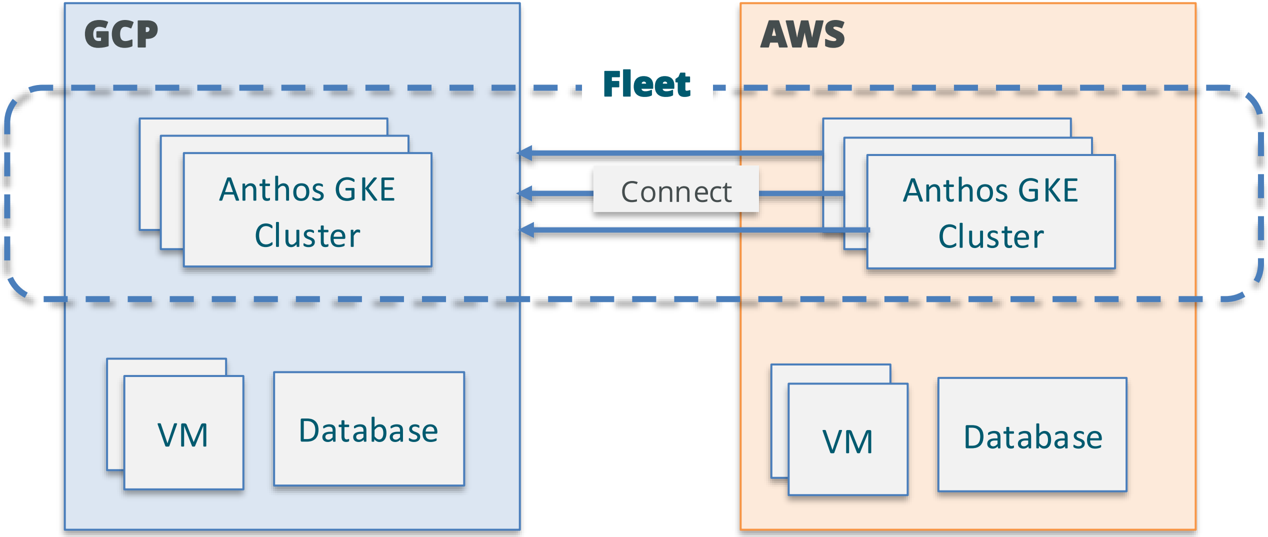

Pearson

# Anthos Fleets and Connect

- A **fleet** is a logical grouping of Kubernetes cluster and other resources that can be managed together
- When you register a cluster outside of GCP, Anthos uses a Kubernetes Deployment called the **Connect Agent**
- **Connect** establishes a long-lived, encrypted connection between the cluster's Kubernetes API server and Google Cloud

# Anthos Fleets and Connect

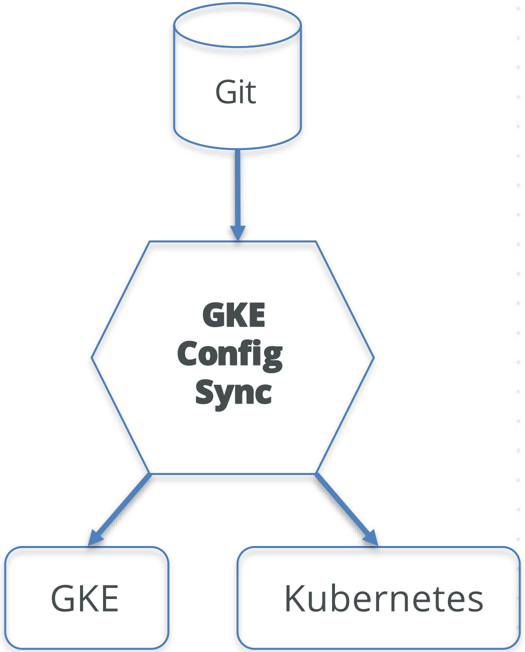
## Example




Unified management (control plane) and user interface for all your clusters

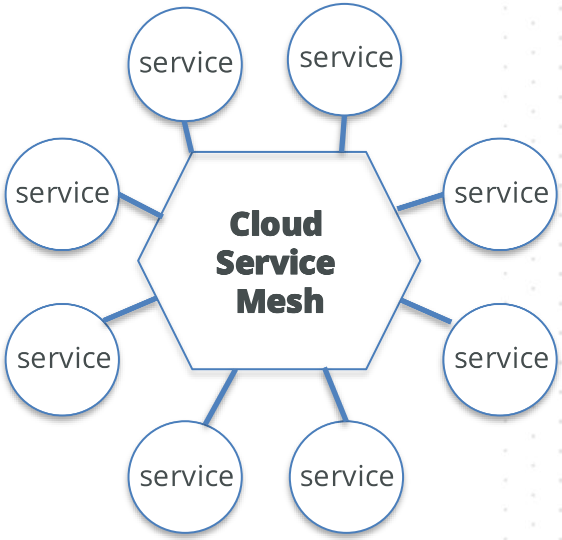
# Config Sync (fka Anthos Config Management)

- You create a common configuration across all your infrastructure
- Once you declare a new desired state, it continuously checks for changes that go against state
- Changes are rolled out to all clusters to reflect the desired state



# Cloud Service Mesh (fka Anthos Service Mesh)

- Fully managed service mesh based on **Istio**
- Out-of-the-box telemetry with all traffic monitored through a proxy
- Enforce policies across VMs and containers
- Can be used for traffic management (canary, location-based routing)



# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

Pearson

# Cloud-native Networking: VPC

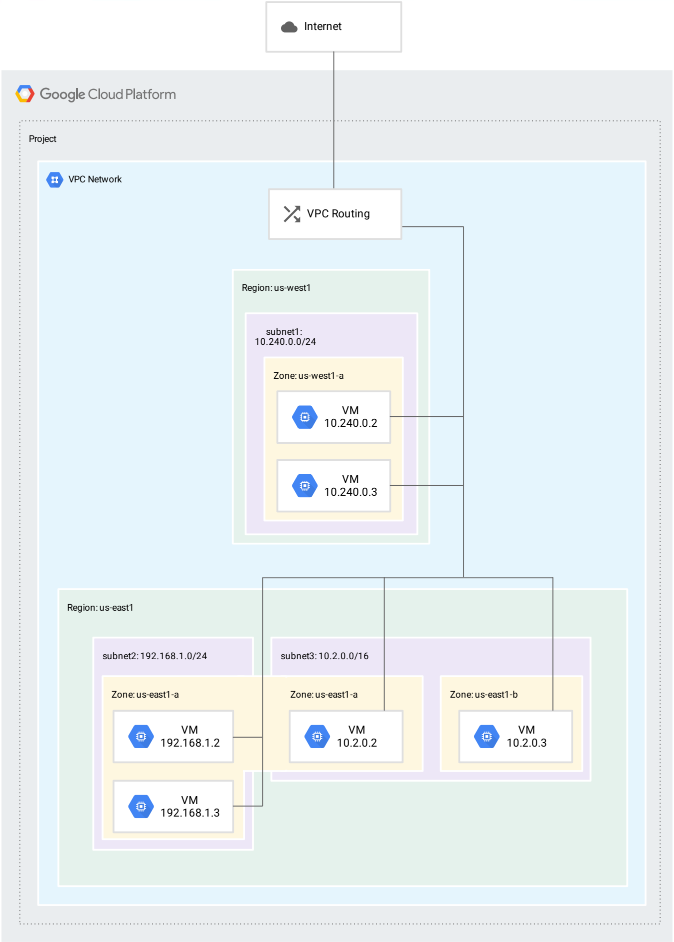
- Global resource, logically isolated
- Consisting of a list of regional subnetworks (subnets)

### Auto mode

- One subnet per region
- Regional IP allocation
- Fixed /20 subnetwork per region
- Expandable up to /16

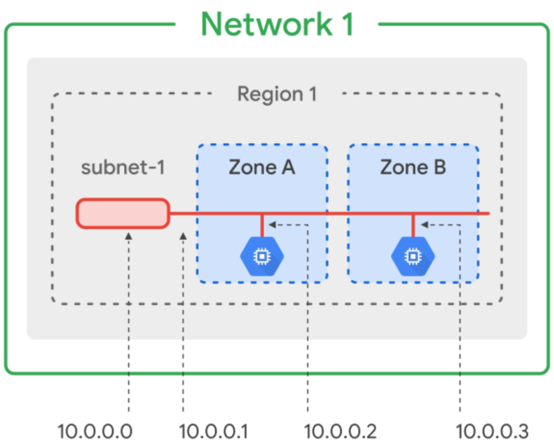
### Custom mode

- No default subnets
- Full control of IP ranges
- Regional IP allocation
- Expandable to any RFC 1918 size




Pearson

# Cloud-native Networking: Subnets

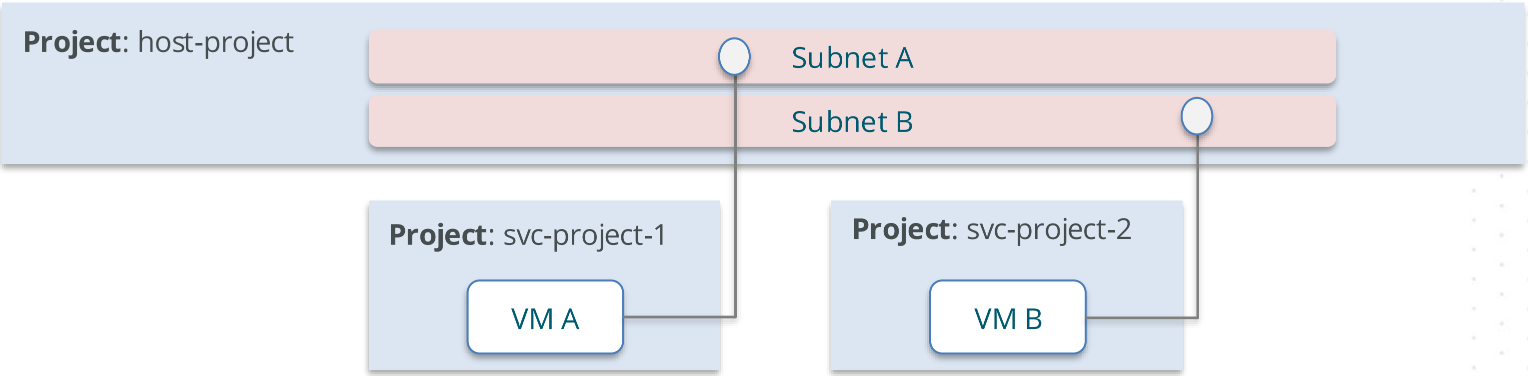



## Subnets cross zones

- VMs can be on the same subnet but different zones
- A single firewall rule can apply to both VMs
- Subnets cannot overlap with each other

# Cloud-native Networking: Shared VPC

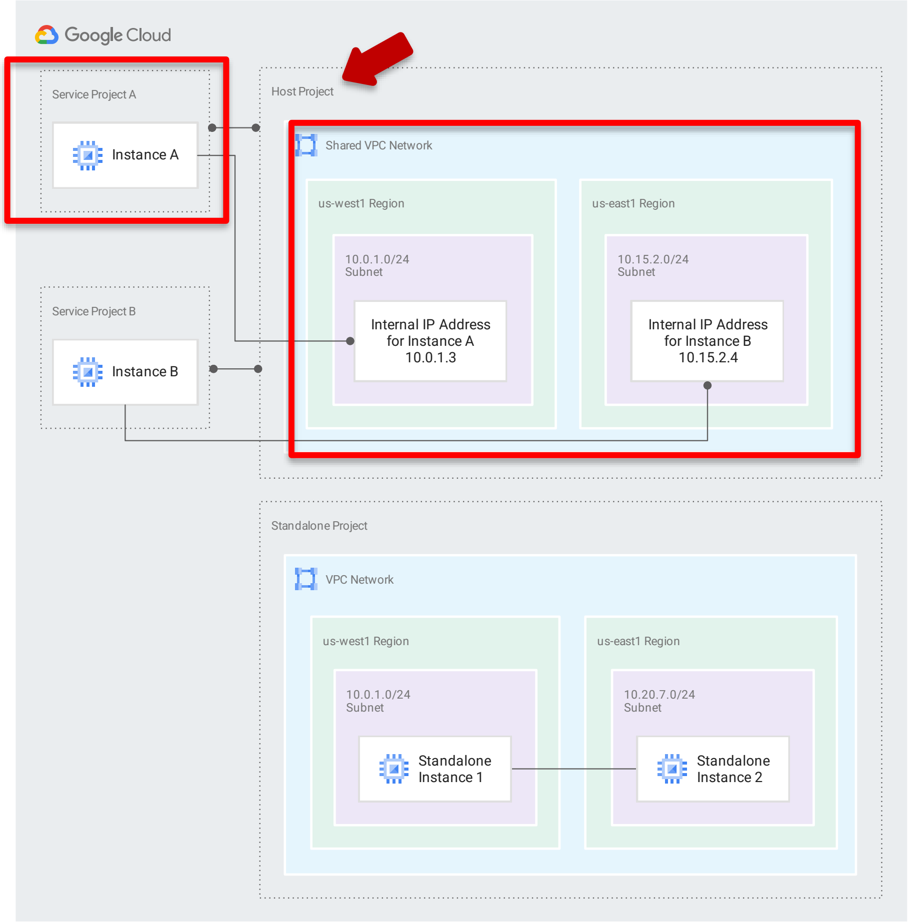
- You can share a VPC network from one project (**host** project) to other projects (**service** projects)
- Benefits:
  - Separation of concerns
  - Enforce consistent security policies for multiple (service) projects
  - Help separate budgeting and internal cost allocation




Pearson

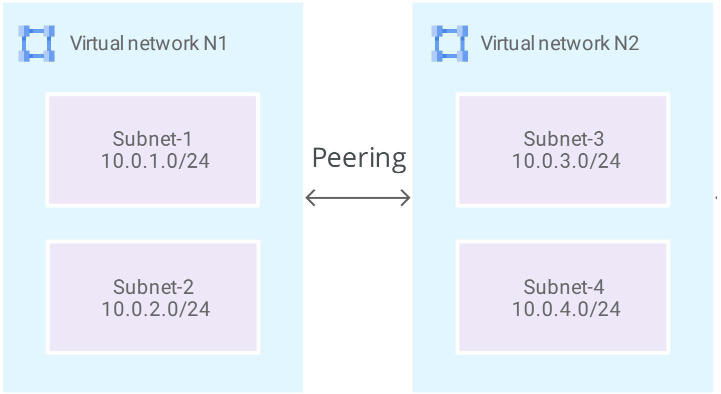
# Cloud-native Networking: Shared VPC

## Shared VPC



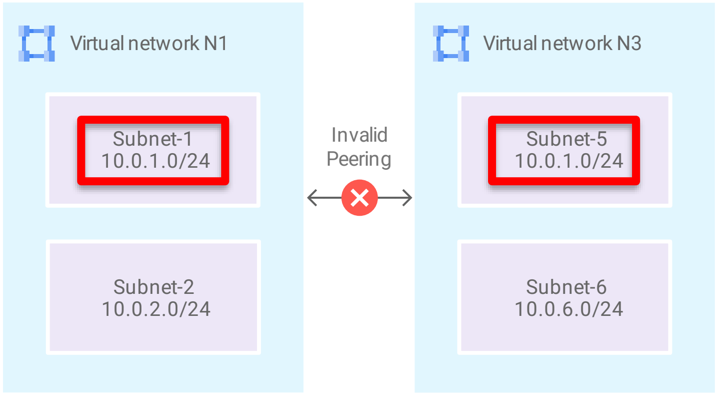
# Cloud-Native Networking: VPC Network Peering

- Internal IP address connectivity across VPCs
- Workloads on peered VPCs become privately accessible



# Cloud-Native Networking: VPC Network Peering

- Internal IP address connectivity across VPCs
- Workloads on peered VPCs become privately accessible

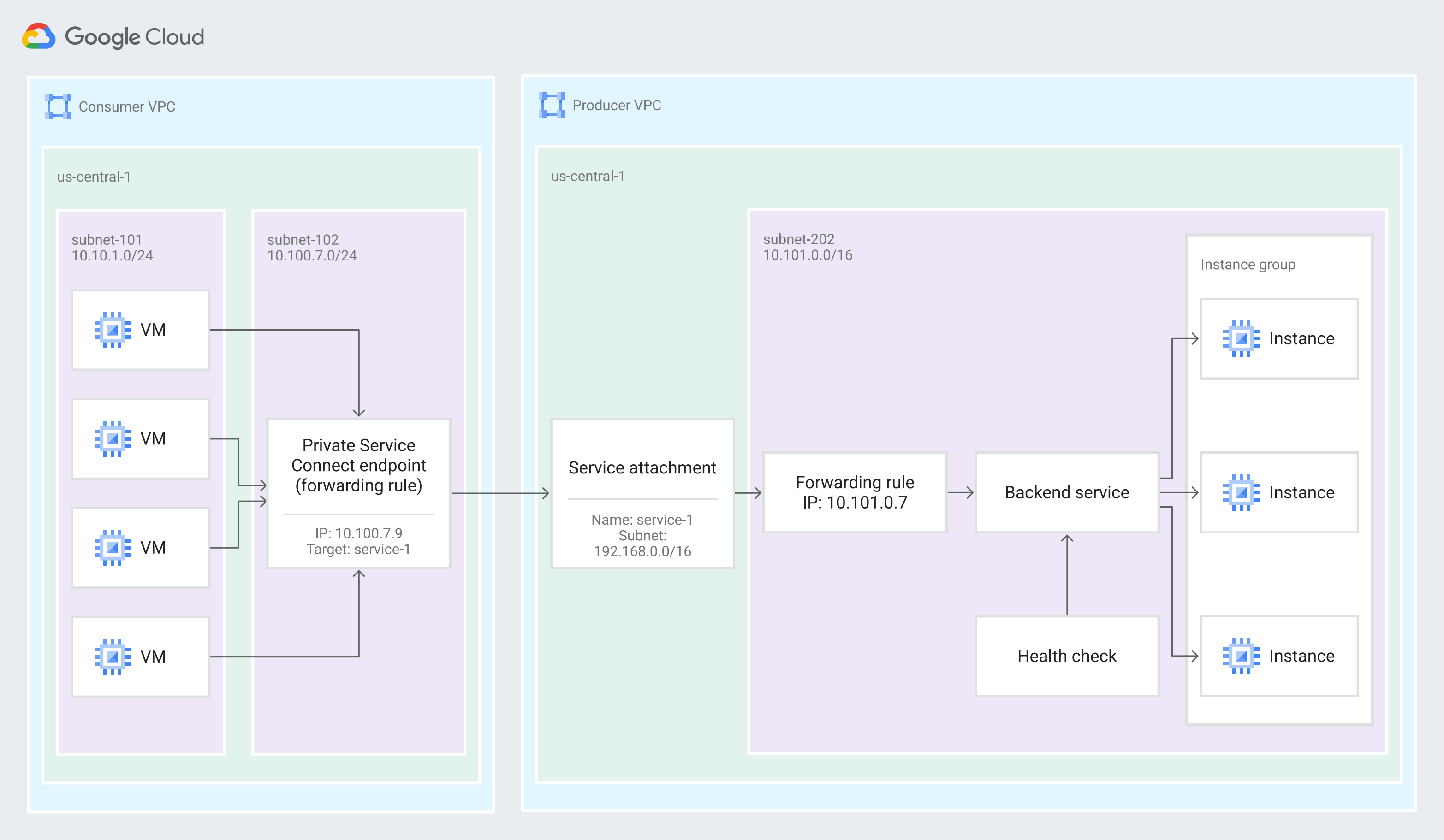


# Cloud-native Networking: Private Service Connect

- Allows private consumption of services across VPC networks
- Can be used to access:
  - Supported Google APIs and services
  - Third-party managed services in another VPC network (e.g. Snowflake, MongoDB)
  - A self-hosted service

Pearson

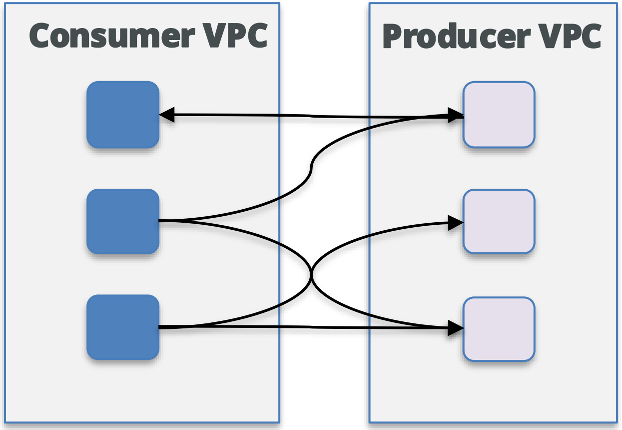
# Cloud-native Networking: Private Service Connect




Pearson

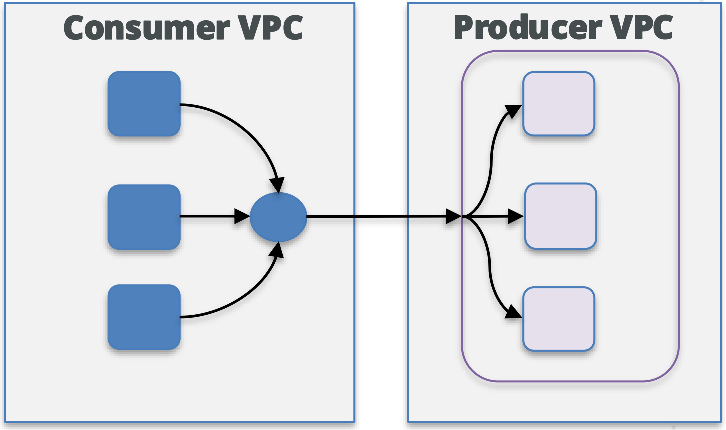
# VPC Peering vs Private Service Connect

## VPC Peering




Network-to-network, many-to-many communication
IP address coordination

## Private Service Connect




Service-oriented model
Granular and uni-directional (client -> service)
No IP coordination required

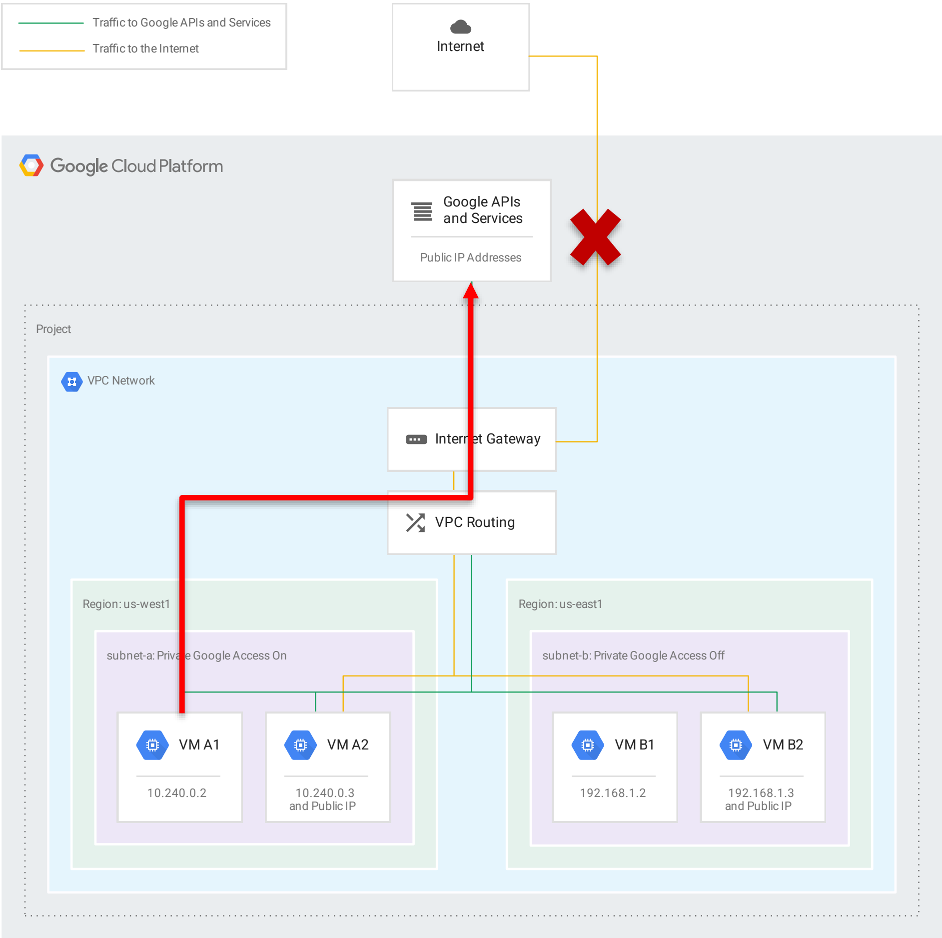
# Cloud-native Networking: Private Google Access

- Allows VMs without public IP address to reach Google APIs and services
- Enabled on a subnet
- **Private Google Access for on-premises hosts** allows on-premises hosts to reach Google APIs and services through Cloud VPN or Cloud Interconnect

# Cloud-native Networking: Private Google Access

# Private Google Access

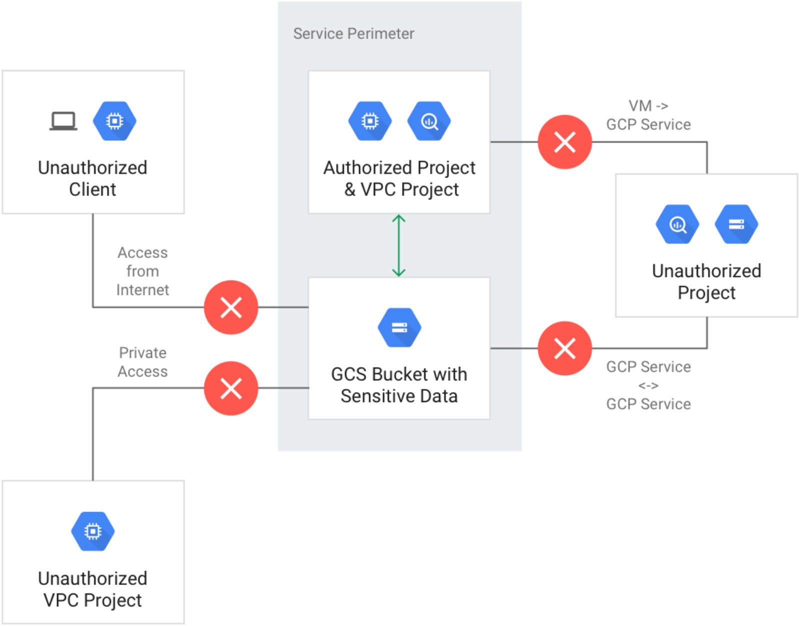
Private Google Access has **no effect** on instances that have external IP addresses




Pearson

# Cloud-native Networking: VPC Service Controls

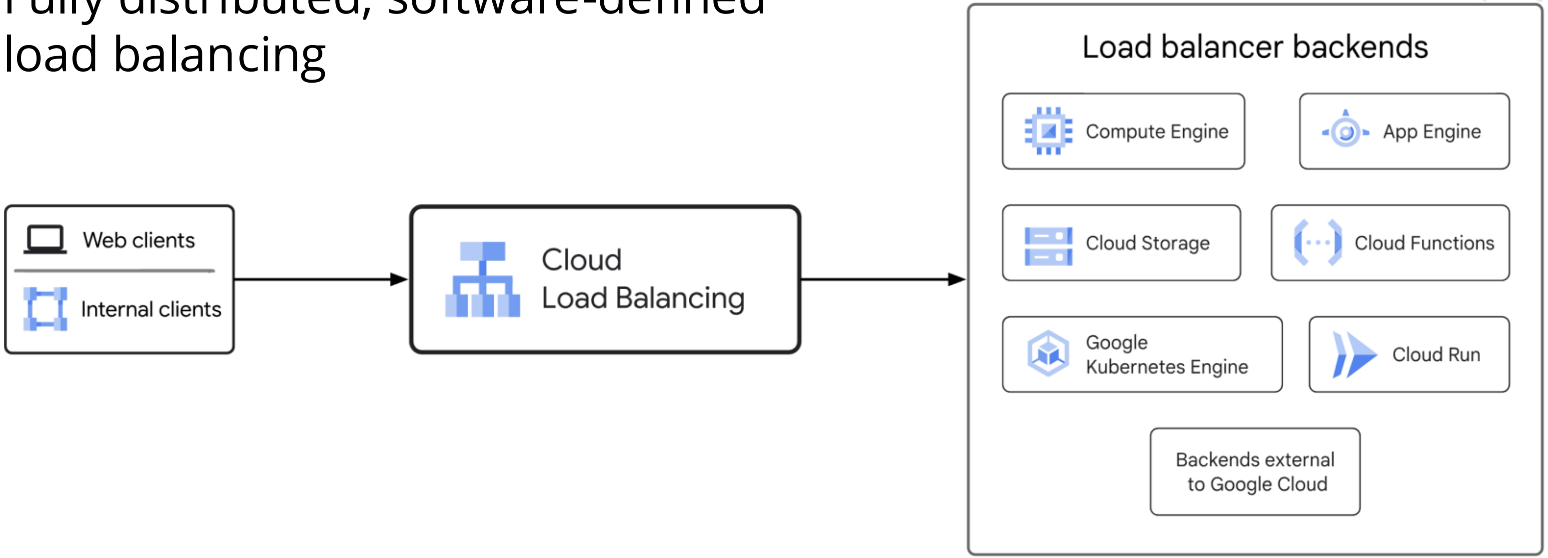
- Create a **perimeter** that protects resources and data
- Clients within perimeter do not have access to (unauthorized) resources outside the perimeter
- Unauthorized clients outside the perimeter don't have access to resources inside the perimeter




Pearson

# Cloud-native Networking: Load Balancing

Fully distributed, software-defined load balancing



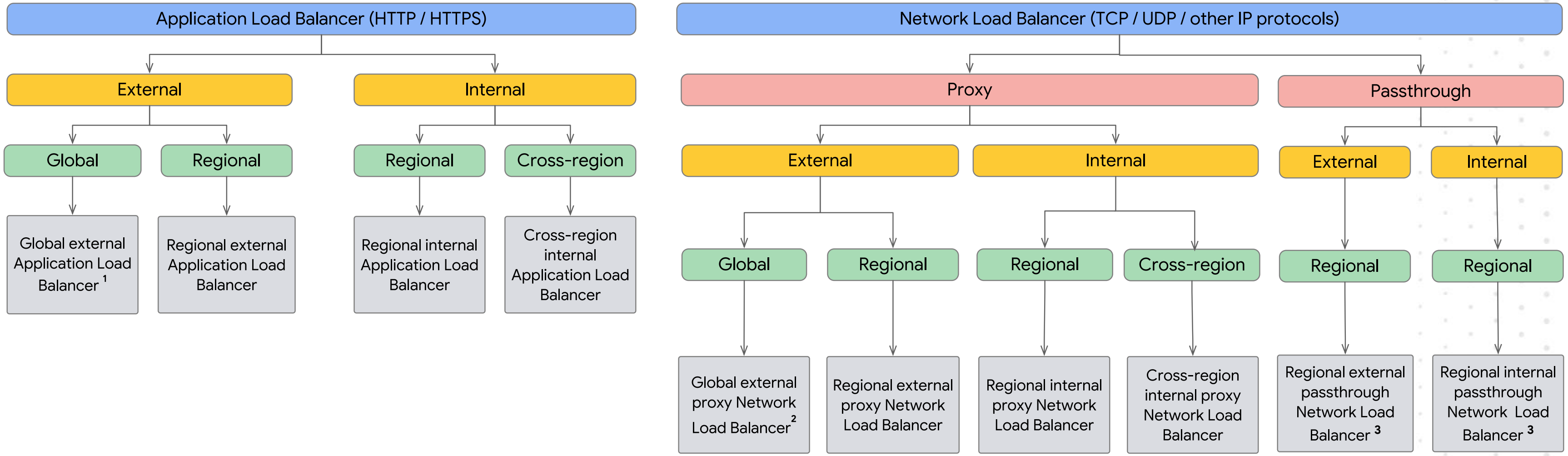

Pearson

# Cloud-native Networking: Load Balancing

<table>
<tr><th>Cloud Load Balancing</th><th>External<br>(Accepts internet traffic)</th><th>Internal<br>(Accepts only RFC 1918 traffic)</th></tr>
<tr><td>Application Load Balancers<br><br>HTTPS<br>Layer 7 load balancing</td><td>• global external<br>• regional external<br>• classic</td><td>• cross-region internal<br>• regional internal</td></tr>
<tr><td rowspan="4">Network Load Balancers<br><br>TCP/SSL/Other<br>Layer 4 load balancing</td><td colspan="2">Proxy Network Load Balancers</td></tr>
<tr><td>• global external<br>• regional external<br>• classic</td><td>• cross-region internal<br>• regional internal</td></tr>
<tr><td colspan="2">Passthrough Network Load Balancers</td></tr>
<tr><td>• regional external</td><td>• regional internal</td></tr>
</table>

# Cloud-native Networking: Load Balancing

## Load balancer deployment modes tree




Source: https://cloud.google.com/load-balancing/docs/load-balancing-overview#choose-lb

Pearson

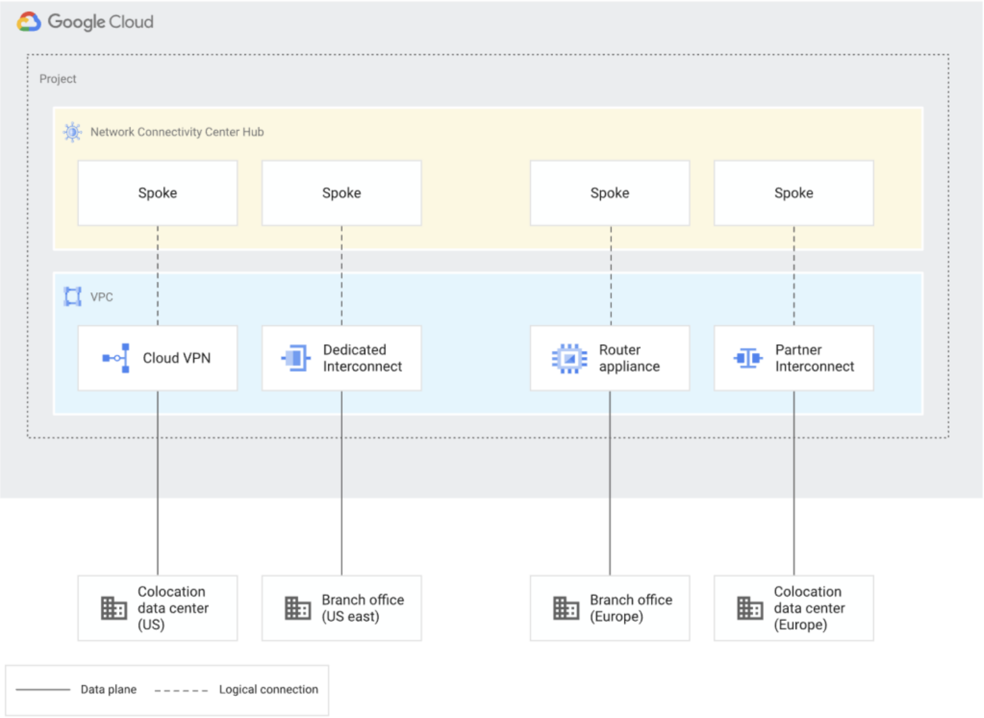
# Network Connectivity Center

Simplifies network connectivity

Spokes are connected to a central Management resource (hub)

Connect multiple VPC networks to one another or to on-premise/other cloud networks

Use Router appliance VMs to manage connectivity between your VPC networks.



# Cloud-native Networking: Other Services

- Cloud DNS
- Cloud CDN
- Cloud Armor
- Cloud IDS

# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

# Storing Data

- Relational vs. non-relational
- Transactional vs. non-transactional
- Structured vs. unstructured (vs. semi-structured)

# Relational and Structured Storage

## Cloud SQL

MySQL, PostgreSQL, and SQL Server

ty SLA of **99.95%**

**Amazon RDS / Aurora**

**Azure SQL / Database for MySQL**

## Cloud Spanner

Standard SQL and PostgreSQL

tic, synchronous replication

ity SLA of **99.99%** (regional instance) or **99.999%** (multi-regional instance)

**Amazon Aurora**

**Azure CosmosDB**

# Relational and Structured (Non-transactional)

## BigQuery

- Serverless petabyte-scale data warehouse
- Storage separate from compute
- For analytics and business intelligence (BI) workloads
- Availability SLA of **99.99%**

**Amazon Redshift**

**Azure Synapse Analytics**

# Non-Relational and (Semi-)structured Storage

## Cloud Firestore

- Automatic scaling and availability SLA of **99.999%** (multi-regional)
- Real-time updates, direct database connectivity for le, web, and IoT apps
- g consistency, ACID support (document-level)

**AWS DynamoDB**

**Azure CosmosDB**

## Cloud Memorystore

- Fully-managed Redis and Memcached
- Up to 5 Read replicas (Redis)
- rizontally scale for reads and writes (Memcached)
- tically scale up to 300GB (Redis) and up to 256GB node (Memcached)
- ilability SLA of **99.9%**

**Amazon Elasticache**

**Azure Cache for Redis**

Pearson

# Non-Relational and (Semi-)structured Storage

## Cloud Bigtable

- Fully managed, NoSQL database for large analytical and operational workloads
- Consistent sub-10ms latency
- Handles millions of requests per second
- Storage scales seamlessly with demand
- Easily connect to Apache ecosystem or BigQuery
- **Up to 99.999%** availability

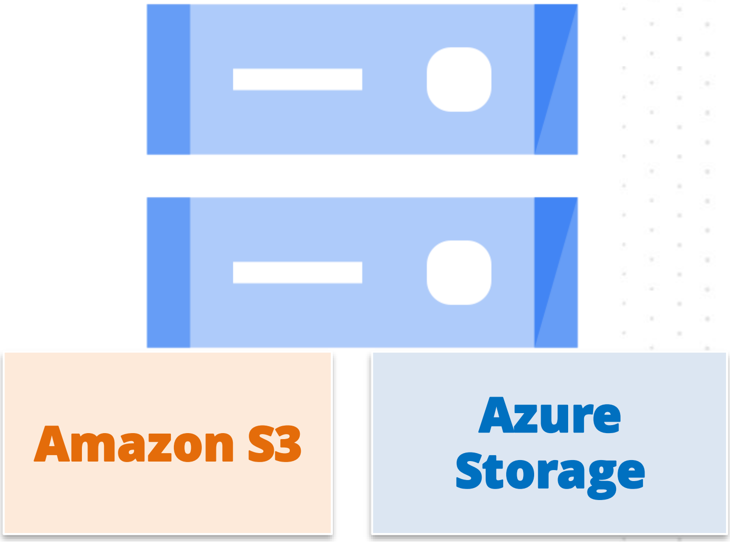
**AWS DynamoDB**

**Azure CosmosDB**

# Unstructured Storage ("Object" Storage)

## Cloud Storage

- Object storage for any amount of data
- 99.999999999% annual storage durability
- Different storage classes for cost saving opportunities
- Object Lifecycle Management (OLM) features
- Archival storage
- Availability SLA of **99.9%** (regional), **99.95%** (dual-, multi-region)



# File Storage

## Cloud Filestore

- Fully managed service for file migration and storage
- Mount file shares on Compute Engine VMs
- Can automatically scale up or down based on demand
- Regional availability SLA of **99.99%**

**Amazon EFS / FSX**

**Azure Files**

# Block Storage

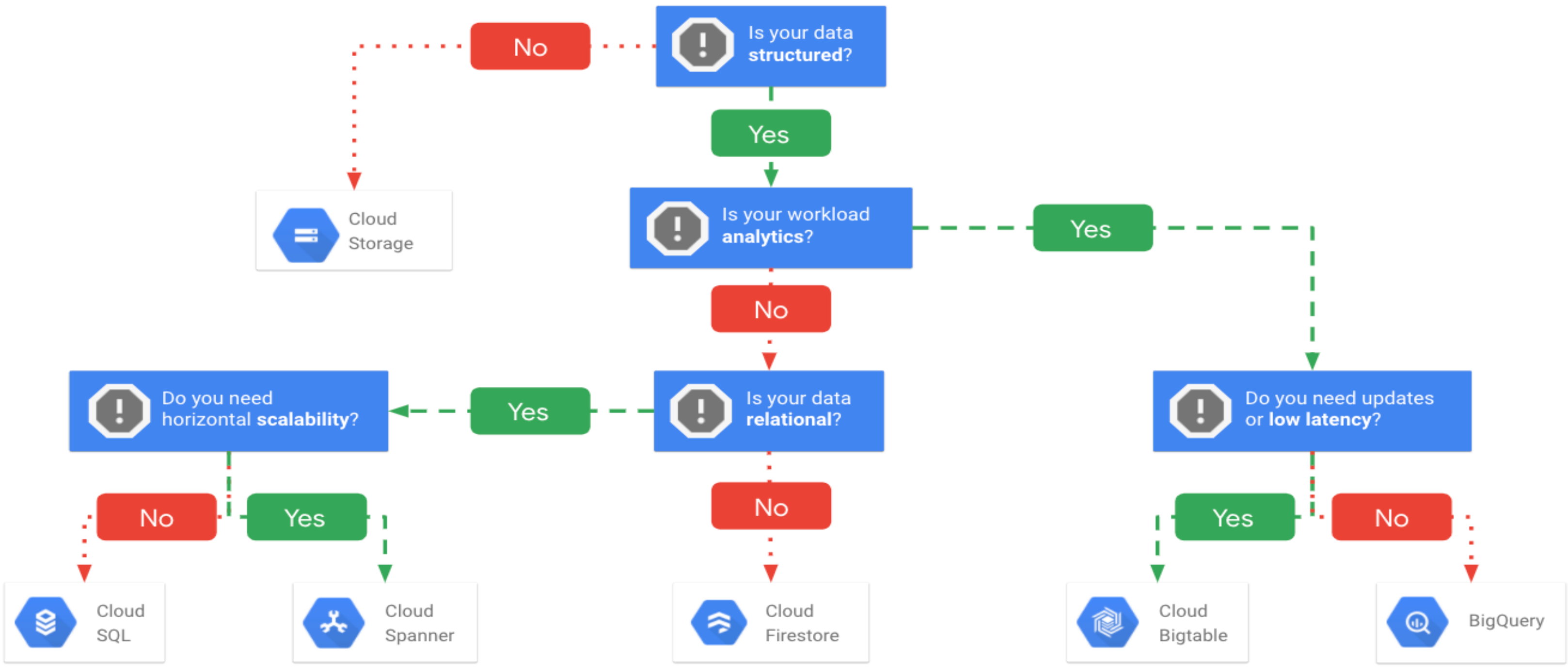
## Persistent Disks

- Durable, high-performance block storage for virtual machines
- Performance scales with the size of the disk and with the number of vCPUs on the VM
- Data stored redundantly

## Local SSD

- High-performance block storage for virtual machines
- Physically attached to the server
- Higher throughput and lower latency than persistent disks
- **Data persists only until instance is stopped or deleted**
- Each local SSD is 375GB

# Choosing Storage and Database Types




Pearson

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
|---|---|
| Data lakes<br>Videos, images, and web assets<br>Backups<br>Media archives | Cloud Storage |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
|---|---|
| Disks for virtual machines<br>Storage for databases<br>Sharing read-only data across multiple VMs<br>VM disk backups | Persistent Disk |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
| --- | --- |
| Flash-optimized databases<br>Hot caching layer for analytics<br>Application scratch disk | **Local SSD** |

Pearson

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
|---|---|
| Rendering and media processing<br>Filesystem migrations<br>Web content management | **Filestore** |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
| --- | --- |
| Mobile apps<br>User-generated content<br>Robust uploads over mobile networks | **Cloud Storage for Firebase** |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
|---|---|
| Time-series data<br>Big data<br>IoT<br>Adtech<br>Personalization | **Bigtable** |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
| --- | --- |
| Big data analytics<br>Business intelligence<br>Data warehousing<br>Machine learning | BigQuery |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
| --- | --- |
| User profiles<br>User session management<br>Real-time capabilities<br>Cross-device data synchronization<br>Collaborative multi-user mobile apps, Gaming | **Cloud Firestore** |

# Choosing Appropriate Storage Types

| Use cases | Service to think of |
|---|---|
| Application caching<br>Gaming<br>Stream processing<br>Very low-latency data access | **Cloud Memorystore** |

# Data Growth Planning: Capacity

| | Cloud Firestore | Cloud SQL | Cloud Storage | Bigtable | Cloud Spanner | BigQuery |
|---|---|---|---|---|---|---|
| **Capacity** | Terabytes+ | Terabytes | Petabytes+ | Petabytes+ | Petabytes | Petabytes+ |

Pearson

# Data Growth Planning: Cost Considerations

## Cost-effective for small (and large) data

- Cloud Storage
- BigQuery
- Cloud Firestore
- Cloud SQL

## Cost-effective only for large data

- Bigtable ***(>1TB)***
- Cloud Spanner ***(>2TB)***

# Data Growth Planning: Going Global

- Consider using **Cloud CDN** for static web assets
- Consider **Spanner** instead of Cloud SQL
- Leverage **multi-region** locations for **Cloud Storage** buckets
- Leverage **multi-region** locations for **Cloud Firestore**
- Leverage **Bigtable cross-region** replication

# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

# Data Processing Design Considerations

- Streaming vs. batch
- Real-time analytics vs. storing for later use
- Data volume

# Design Data Pipelines: Cloud Dataflow

## Cloud Dataflow

- Unified streaming and batch
- Apache Beam SDK
- Processing resources are provisioned automatically
- Horizontal autoscaling
- Consistent exactly-once processing

**Amazon Kinesis**

**Azure Databricks**

# Design Data Pipelines: Cloud Pub/Sub

## Cloud Pub/Sub

- No-ops streaming ingestion
- Scalable messaging or queueing system
- At-least-once message delivery
- In-order or any-order
- Push and Pull modes

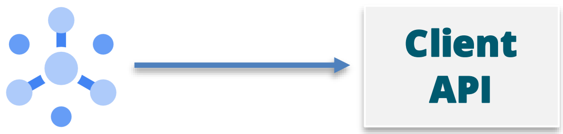
**Amazon SQS / MSK**

**Azure Stream Analytics / HDInsight Kafka**

# Design Data Pipelines: Cloud Pub/Sub

## Cloud Pub/Sub: Push mode

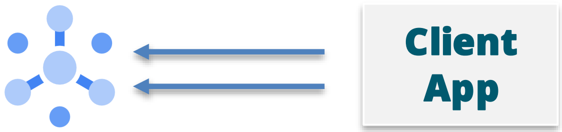
- Pub/Sub server initiates request to deliver messages
- One response is returned in each request
- Automatic flow control based on acknowledgement rate




Use cases: multiple topics, one webhook, App Engine (Standard) and Cloud Functions subscribers

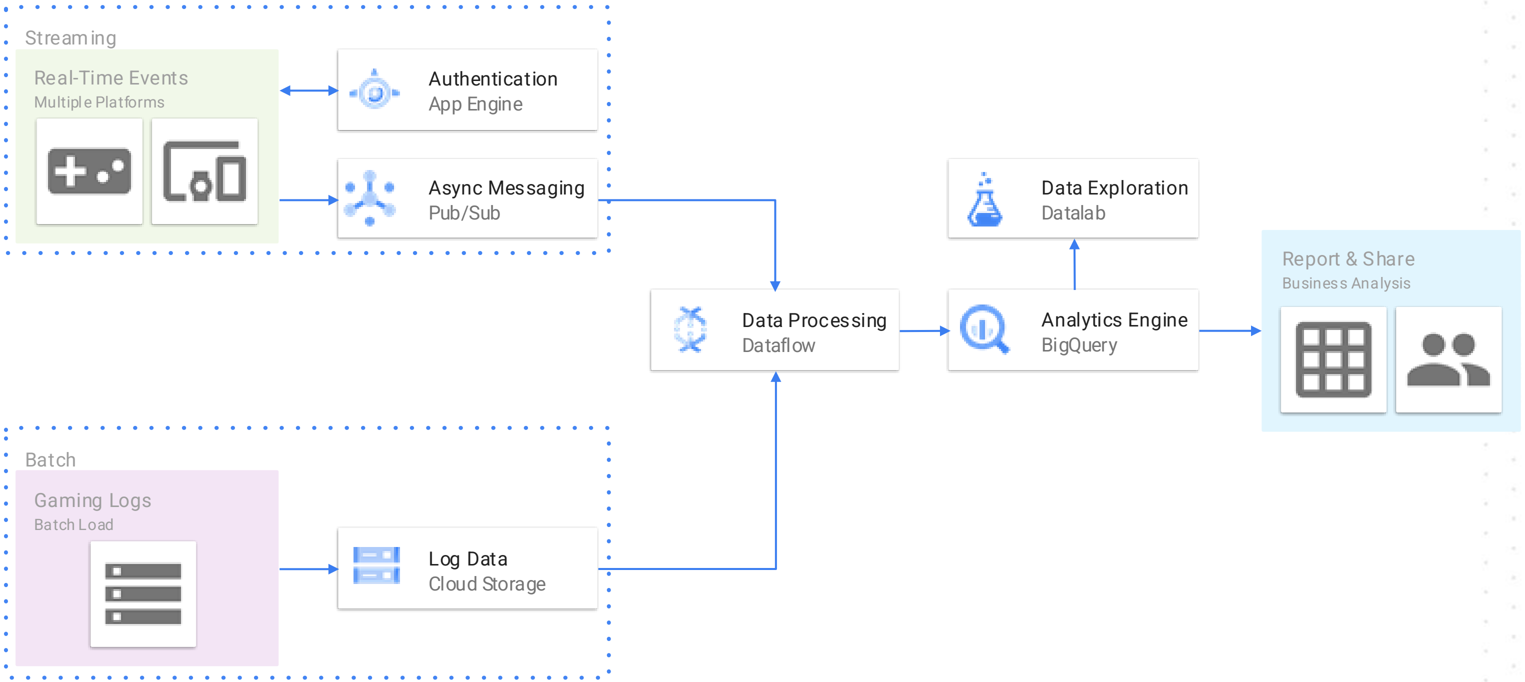
## Cloud Pub/Sub: Pull mode

- Subscriber client initiates requests to retrieve messages
- Can have multiple responses returned
- The subscriber client controls the rate of delivery (flow control)



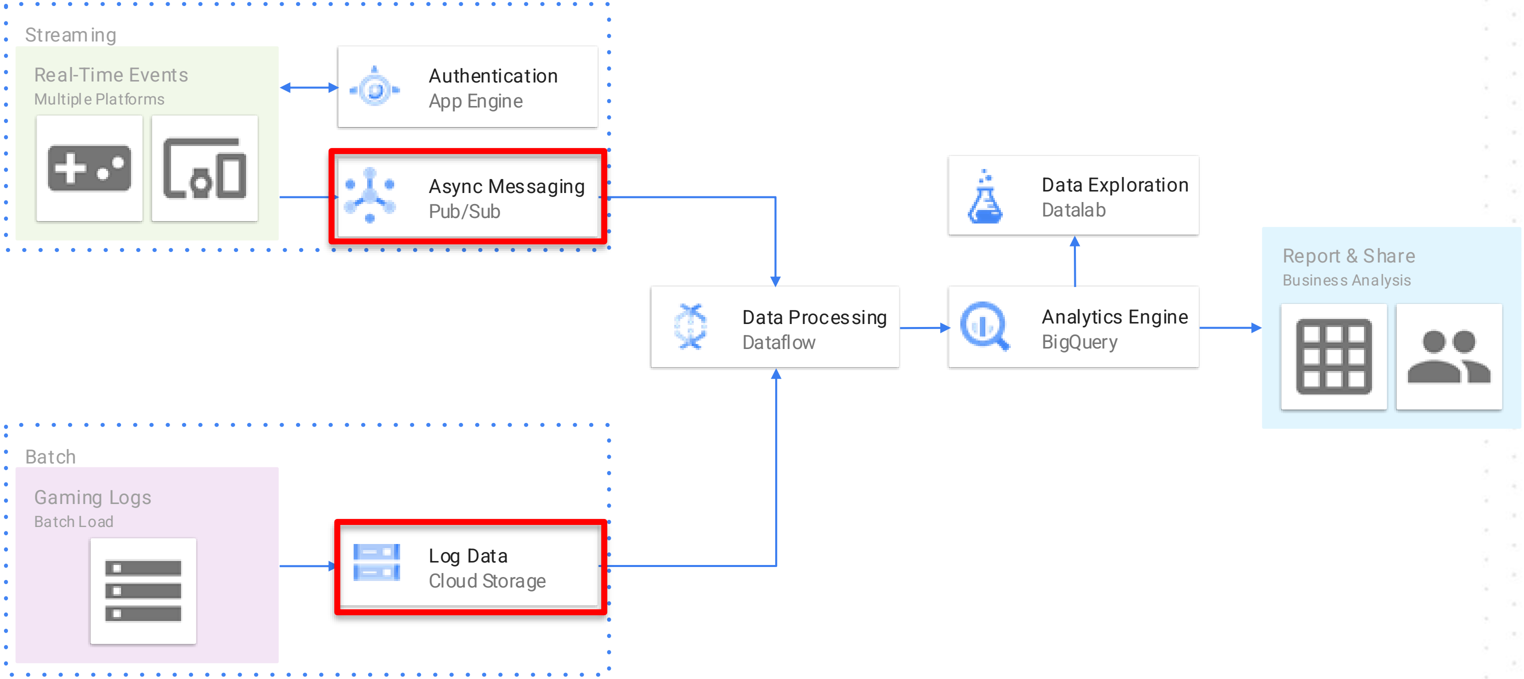

Use cases: large volume of messages, throughput is critical

# Streaming and Batch Data Pipeline Example

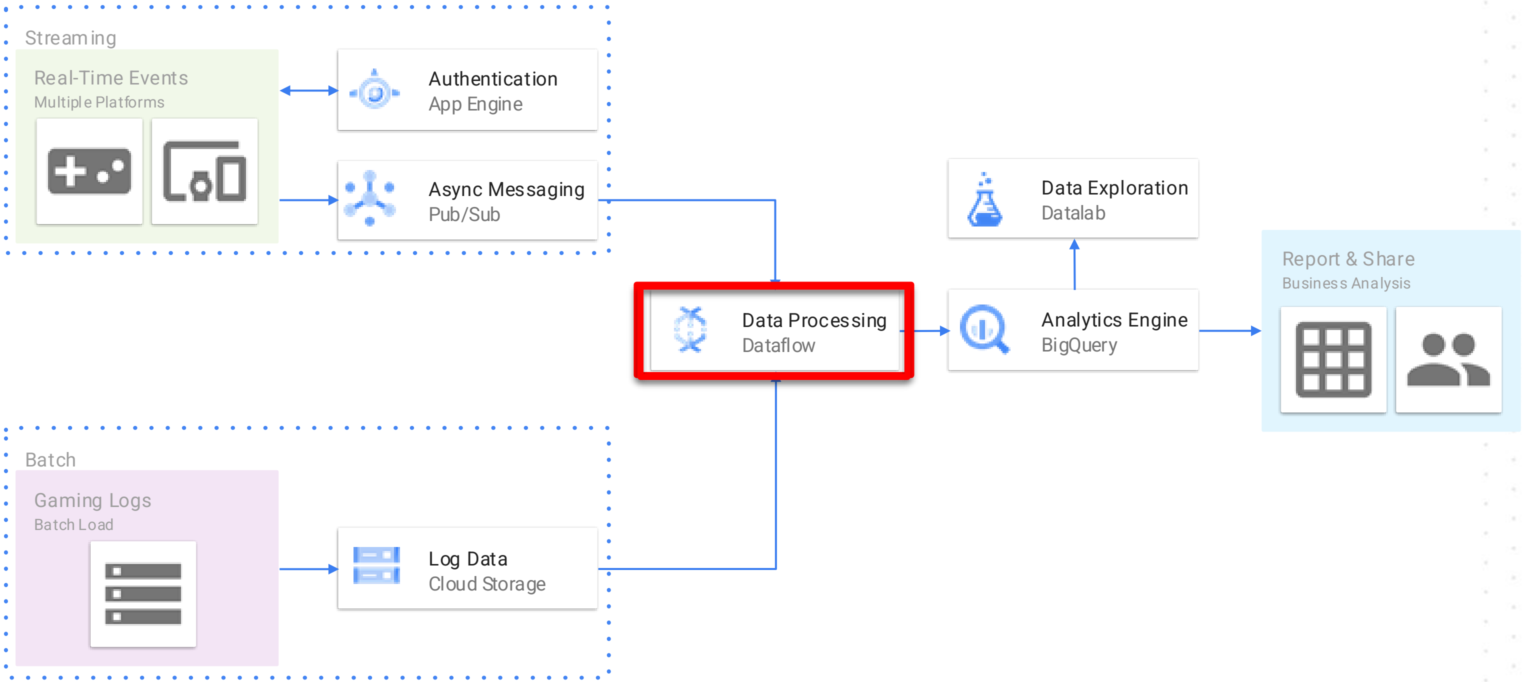



Pearson

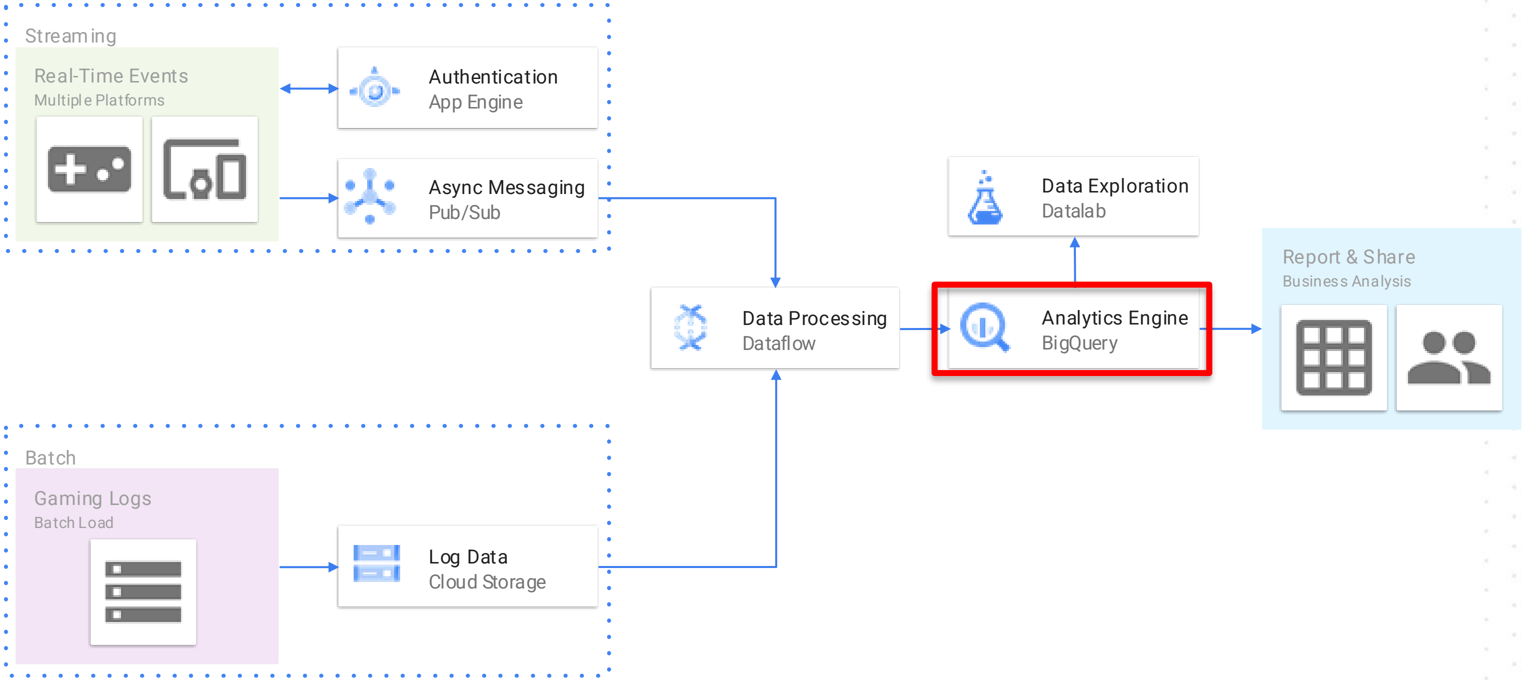
# Streaming and Batch Data Pipeline Example



# Streaming and Batch Data Pipeline Example



# Streaming and Batch Data Pipeline Example



# Big Data Processing: Cloud Dataproc

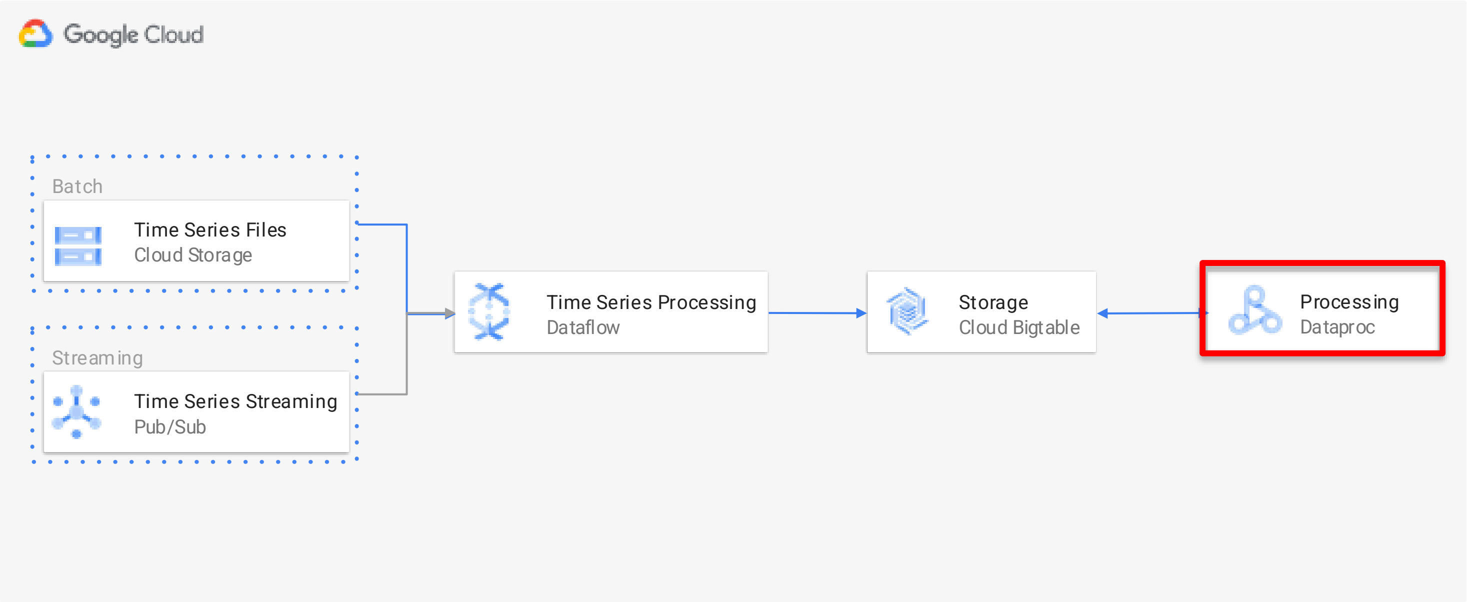
## Cloud Dataproc

- Fully managed **Apache Spark** and **Hadoop** service
- Apache Flink, Presto and 30+ other OSS frameworks
- For large-scale batch processing, querying, streaming, and machine learning
- Can run serverless, or on Kubernetes/VMs
- Built-in integration with Cloud Storage, BigQuery and Bigtable

**Amazon EMR**

**Azure Synapse Analytics / HDInsight**

# Bid Data Processing: Time Series Analysis



# Other Data Processing Services

## Cloud Dataprep

- Integrated partner service operated by Trifacta
- Visual data wrangling tool to explore, combine, and transform data
- A set of transformation steps is called a “recipe”

## Cloud Data Fusion

- Powered by the open source project CDAP (with pipeline portability)
- Visual data wrangling and data integration/pipelining tool (code-free ETL)

# Segment 4: Designing network, storage, and compute

Objectives

- Integrations with on-premises
- Multicloud environments
- Designing VPC networks
- Choosing appropriate storage types
- Choosing data processing technologies
- Choosing compute resources

# Compute Strategies

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)

# Compute Strategies

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)

Application

Data

Runtime

Middleware

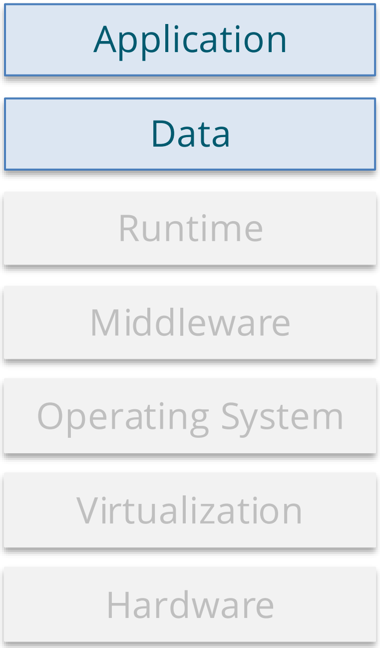
Operating System

Virtualization

Hardware

# Compute Strategies

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)



# IaaS: Compute Engine instances

When to choose Compute Engine instances?

- Full access to OS settings and/or underlying filesystem is required
- Speed up and/or derisk a data center migration (**lift-and-shift**)
- Legacy applications without a suitable platform product
- Workload requires GPU (note: also supported in GKE)

# Managed Instance Groups (MIGs)

- Deploy stateless identical instances based on **instance template**
- Autoscaling, autohealing, and rolling update capabilities
- Can be single zone or regional


Pearson

# IaaS-PaaS: Google Kubernetes Engine (GKE)

## Google Kubernetes Engine (GKE)

- Managed Kubernetes platform
- High-availability control plane
- Cluster autoscaling
- Two modes of operations: **Standard** and **Autopilot**

**Amazon EKS**

**Azure AKS**

Pearson

# IaaS-PaaS: Google Kubernetes Engine (GKE)

You're responsible for:

- Creating and maintaing container images
- Installing and maintaining application libraries and runtime
- Configuring some aspects of networking, storage, and observability
- Architecting for fault-tolerance and scalability (easier to do)

# When to Choose GKE

- Containerized workloads of sufficient complexity
- Stateful microservices
- Scaling and modernizing applications
- Hybrid- and multi-cloud workloads

# PaaS products: App Engine

## App Engine

- Fully-managed, serverless platform for developing and hosting web applications
- Can choose from several languages, libraries, and frameworks
- Scales automatically
- Availability SLA of 99.95%
- Two environments: Standard and Flexible

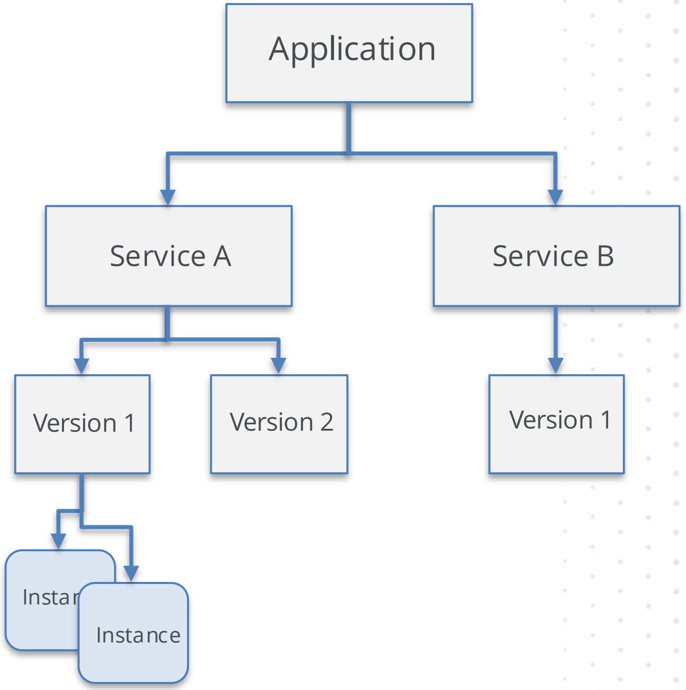
**AWS Elastic Beanstalk**

**Azure App Service**

# PaaS products: App Engine

## App Engine

- Fully-managed, serverless platform for developing and hosting web applications
- Can choose from several languages, libraries, and frameworks
- Scales automatically
- Availability SLA of 99.95%
- Two environments: Standard and Flexible



# PaaS products: App Engine

### App Engine (Standard)

- Containers are preconfigured with one of several available runtimes.
- The instance class determines the amount of memory and CPU available
- Can scale to zero if no traffic
- Cannot SSH or use custom libraries
- Supported languages: Python, Java, Node.js, PHP, Ruby, and Go

### App Engine (Flexible)

- Uses Compute Engine instances (managed by App Engine)
- Greater CPU and memory instance types
- You can take advantage of custom libraries, use SSH for debugging
- You can deploy your own Docker containers
- Can access resources in the same network
- Does **not** scale to zero

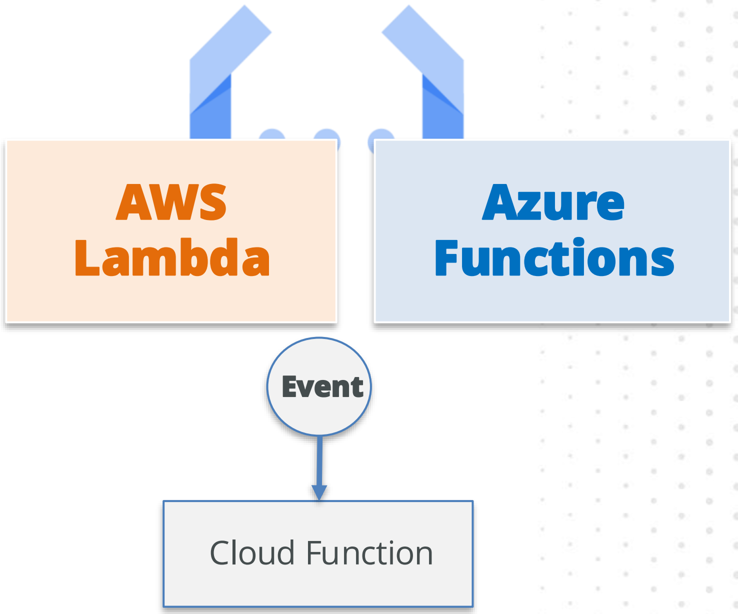
# PaaS products: Cloud (Run) Functions

## Cloud (Run) Functions

- Serverless lightweight compute service
- For single-purpose, standalone functions that respond to events
- Can be written using JavaScript, Python 3, Go, or Java runtimes

Example events that can trigger functions:

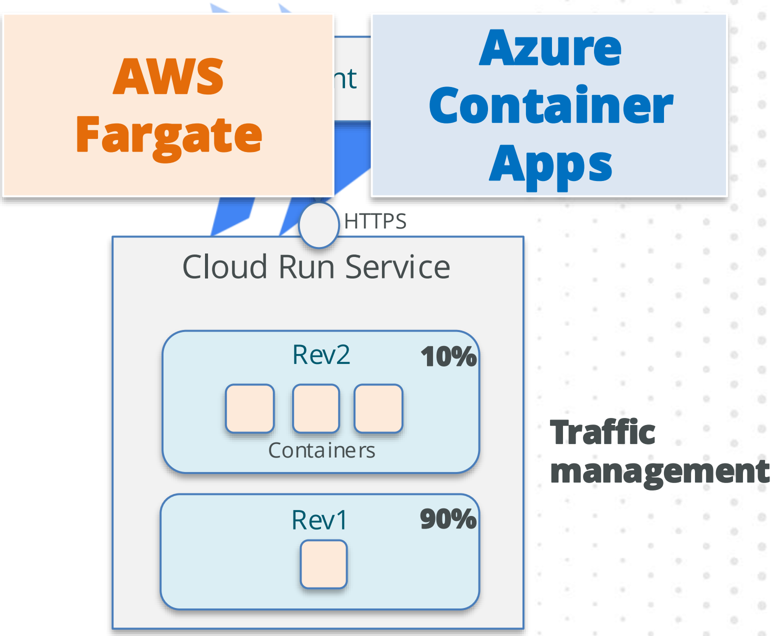
- Changes to data in database
- Files added to a storage system
- New VM created



# PaaS products: Cloud Run

## Cloud Run

- Serverless container platform
- For highly scalable stateless containerized web applications
- No need to build your own container if using **Go**, **Node.js**, **Python**, **Java**, **.NET Core**, or **Ruby**.
- Request-based auto scaling and scale to zero
- Built-in traffic management




Pearson

# Mapping Compute Needs to Platform Products

| Scenarios / needs | Compute platform |
|---|---|
| ETL when a file is created, changed, or removed in Cloud Storage<br><br>Webhooks for events from 3rd party systems<br><br>Lightweight APIs<br><br>Mobile backend that listens and responds to events<br><br>IoT ETL | **Cloud Function** |

# Mapping Compute Needs to Platform Products

| Scenarios / needs | Compute platform |
|---|---|
| Web server<br><br>Django app<br><br>Web and mobile backend | **App Engine (or Cloud Run)** |

# Mapping Compute Needs to Platform Products

| Scenarios / needs | Compute platform |
| --- | --- |
| Container-based stateless web applications<br>Processing streaming data from Pub/Sub<br>Websockets applications | **Cloud Run** |

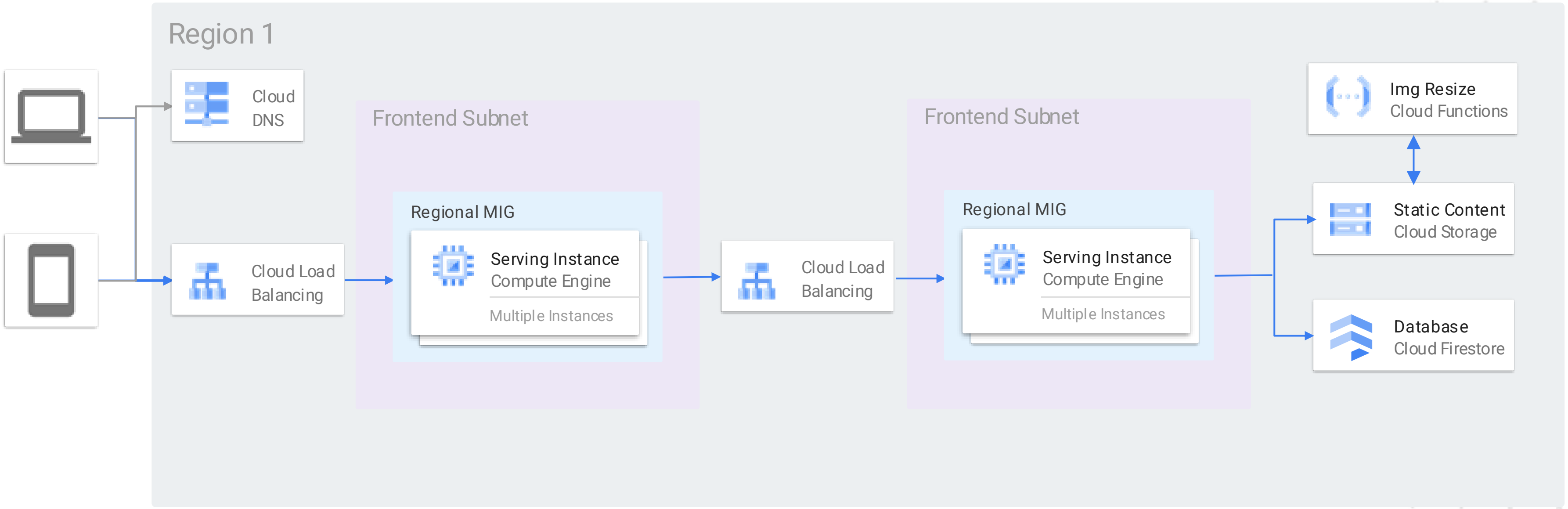
Pearson

# Example: Designing Network, Storage, and Compute

**Requirements**

- Regional 3-tier web app
- IaaS Compute with Debian Linux
- A document database
- Object storage for images
- Event-driven processing for new images (image resizing)

# Example: Designing Network, Storage, and Compute




Pearson

# Firebase

- Mobile development platform
- Firebase and Google Cloud share common infrastructure
  - Cloud Filestore
  - Cloud Storage
  - Cloud Functions
- Firebase also uses GCP's Projects, Billing, and access control

**Use cases:**
Mobile, mobile, mobile

# Questions Breakdown

Following an acquisition of another company, you are tasked with simplifying the management of Kubernetes clusters that are deployed on AWS and consolidate them under the GKE control plane.

What solution in Google Cloud can you leverage to meet this requirement?

A. Anthos with Fleets and Connect

B. Anthos Migrate to Containers

C. Google Kubernetes Engine

D. Kubernetes Multicluster Ingress

# Questions Breakdown

Following an acquisition of another company, you are tasked with simplifying the management of Kubernetes clusters that are deployed on **AWS** and consolidate them **under the GKE control plane**.

What solution in Google Cloud can you leverage to meet this requirement?

A. Anthos with Fleets and Connect

B. Anthos Migrate to Containers

C. Google Kubernetes Engine

D. Kubernetes Multicluster Ingress

# Questions Breakdown

Your company has decided to migrate from an on-premises infrastructure to Google Cloud. The workloads to migrate include a web server on Kubernetes, Apache Hadoop jobs, and a Network-Attached Storage (NAS).

What combination of services would you use on GCP?

A. Migrate Kubernetes application to Google Kubernetes Engine (GKE), Apache Hadoop to Cloud Dataproc, and NAS to Cloud Filestore.
B. Migrate Kubernetes application to Cloud Run, Apache Hadoop to Cloud Dataproc, and NAS to Cloud Storage.
C. Migrate Kubernetes application to Compute Engine VMs, Apache Hadoop to Cloud Dataflow, and NAS to Cloud Filestore.
D. Migrate Kubernetes application to Anthos, Apache Hadoop to Cloud Dataproc, and NAS to Cloud Storage.

# Questions Breakdown

Your company has decided to migrate from an on-premises infrastructure to Google Cloud. The workloads to migrate include a web server on **Kubernetes**, **Apache Hadoop** jobs, and a **Network-Attached Storage** (NAS).

What combination of services would you use on GCP?

A. Migrate Kubernetes application to Google Kubernetes Engine (GKE), Apache Hadoop to Cloud Dataproc, and NAS to Cloud Filestore.
B. Migrate Kubernetes application to Cloud Run, Apache Hadoop to Cloud Dataproc, and NAS to Cloud Storage.
C. Migrate Kubernetes application to Compute Engine VMs, Apache Hadoop to Cloud Dataflow, and NAS to Cloud Filestore.
D. Migrate Kubernetes application to Anthos, Apache Hadoop to Cloud Dataproc, and NAS to Cloud Storage.

# Questions Breakdown

You need to store data generated by multiple sensors installed across hundreds of your company's physical locations. Sensors report data in JSON format every second and this data will be accessed by analysts.

Which database type would you use on GCP?

A. NoSQL.
B. Relational.
C. Blob storage.
D. Filesystem.

Pearson

# Questions Breakdown

You need to store data generated by **multiple sensors** installed across hundreds of your company's physical locations. Sensors report data in **JSON** format **every second** and this data will be accessed by analysts.

Which database type would you use on GCP?

A. NoSQL.

B. Relational.

C. Blob storage.

D. Filesystem.

# Questions Breakdown

A new workload will require a mix of batch and stream processing of data on GCP. The data are to be sourced from Cloud Storage and Cloud Pub/Sub and need to be transformed before being ingested in BigQuery for analysis. Which technology should you choose?

A. Cloud Dataproc.

B. Cloud Dataflow.

C. BigQuery Data Transfer Service.

D. Cloud SQL.

# Questions Breakdown

A new workload will require a **mix of batch and stream** processing of data on GCP. The data are to be sourced from Cloud Storage and Cloud Pub/Sub and need to be transformed before being ingested in BigQuery for analysis. Which technology should you choose?

A. Cloud Dataproc.

B. Cloud Dataflow.

C. BigQuery Data Transfer Service.

D. Cloud SQL.

# Questions Breakdown

A product owner is building a new application and has asked you to identify the best cloud technology to host it. The application will be containerized and have a microservices architecture. The application's platform must be based on open-source technologies for portability and support dynamic scaling based on demand.

Which technology should you choose?

A. Google App Engine.

B. Google Compute Engine.

C. Google Kubernetes Engine.

D. Cloud Function.

# Questions Breakdown

A product owner is building a new application and has asked you to identify the best cloud technology to host it. The application will be **containerized** and have a **microservices** architecture. The application's platform must be based on **open-source** technologies for portability and **support dynamic scaling** based on demand.

Which technology should you choose?

A. Google App Engine.
B. Google Compute Engine.
C. Google Kubernetes Engine.
D. Cloud Function.

# Day 1 Wrap-Up

**Tomorrow's Agenda:**

- Configuring network, storage, and compute
- Designing for observability, security, and compliance
- Deployment and operational best practices
- Case studies solutioning