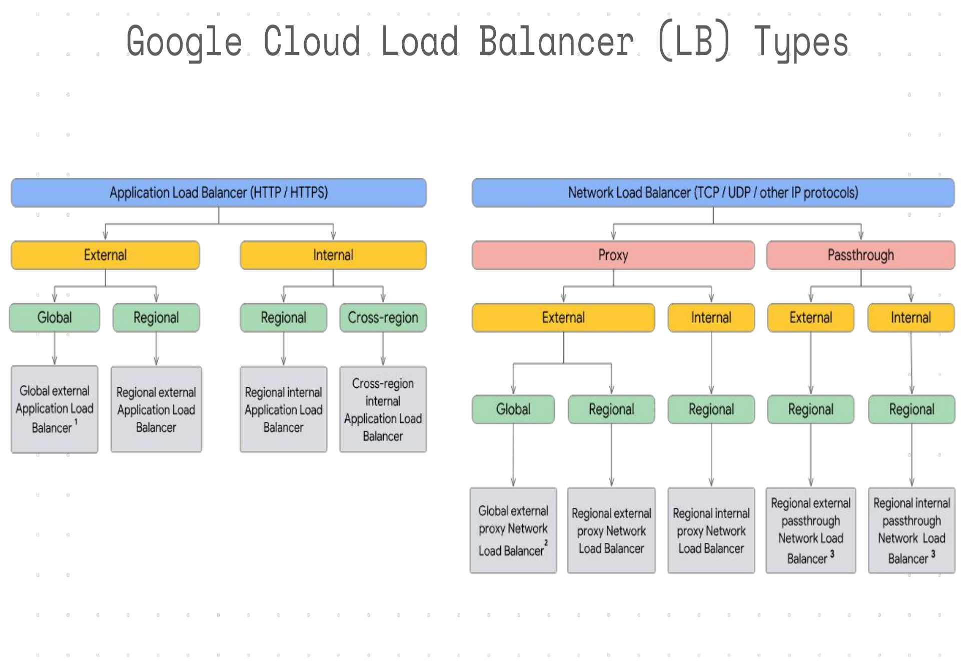

Google Cloud Load Balancer (LB) Types
Application Load Balancer (HTTP / HTTPS)
External
Internal
Global
Regional
Regional
Cross-region
Global external Application Load Balancer 1
Regional external Application Load Balancer
Regional internal Application Load Balancer
Cross-region internal Application Load Balancer
Network Load Balancer (TCP / UDP / other IP protocols)
Proxy
Passthrough
External
Internal
External
Internal
Global
Regional
Regional
Regional
Regional
Global external proxy Network Load Balancer 2
Regional external proxy Network Load Balancer
Regional internal proxy Network Load Balancer
Regional external passthrough Network Load Balancer 3
Regional internal passthrough Network Load Balancer 3

https://cloud.google.com/load-balancing/docs/choosing-load-balancer

1) Application Load Balancers

- are **proxy-based** Layer 7 load balancers that enable you to run and scale your services behind a single IP.
- can be deployed externally or internally depending on whether your app is internet-facing or internal.

2) Network Load Balancers

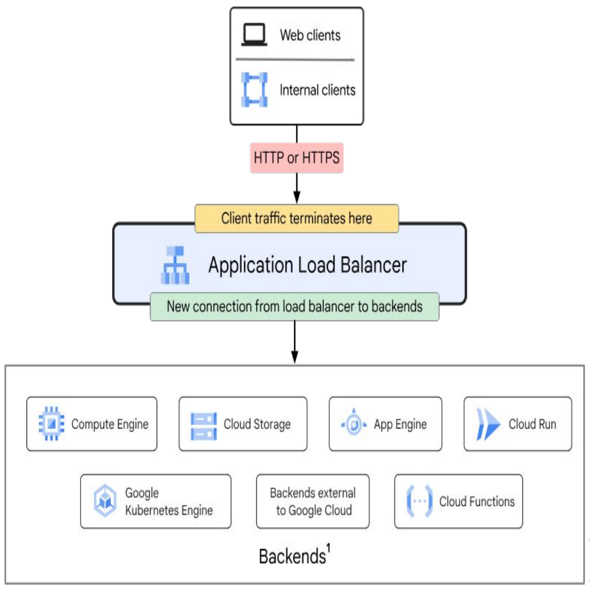
- are Layer 4 load balancers that can handle TCP, UDP, or other IP protocol traffic.
- are available as either:

a. **Proxy Network Load Balancer**. Used if we want to configure a reverse proxy LB with support for **1)** advanced traffic controls and **2)** backends on-prem or in other cloud environments.
   - It distribute TCP traffic to VMs.
   - Traffic is terminated at the load balancing layer and then forwarded to the closest available backend.
   - They are deployed in two modes:
     1. **External proxy Network Load Balancers** that distribute traffic that comes from the internet to backends in your VPC network, on-prem, or in other cloud providers. They can be deployed in global mode or regional mode.
     2. **Internal proxy Network Load Balancers**

b. **Passthrough Network Load Balancer**. Choose it if **1)** you want to preserve the source IP of the client packets, **2)** you prefer direct server return for responses, **or 3)** you want to handle a variety of IP protocols such as TCP, UDP, ICMP/ICMPv6, Encapsulating Security Payload (ESP), and Generic Routing Encapsulation (GRE).
   - They distribute traffic among backends in the same region as the load balancer.

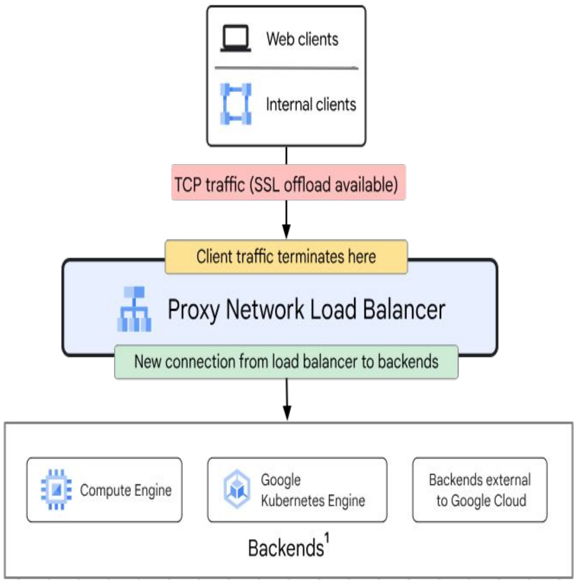
- They are not proxies.
- Load-balanced connections are terminated at the backends.
- Responses from the backend VMs go directly to the clients, not back through the load balancer. The industry term for this is Direct Server Return (DSR).
- They are deployed in two modes:
    1. **External passthrough Network Load Balancers**
    2. **Internal passthrough Network Load Balancers**

Google Cloud Application LB Architecture



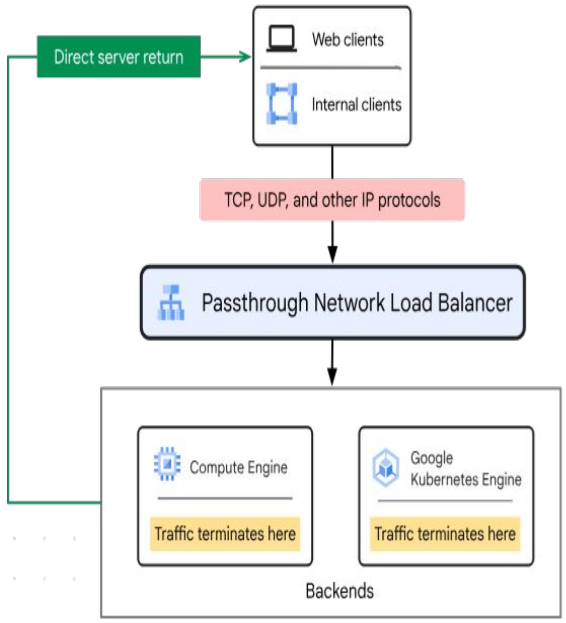
[1] Backend support differs depending on the deployment mode of the load balancer (internal or external, global or regional).

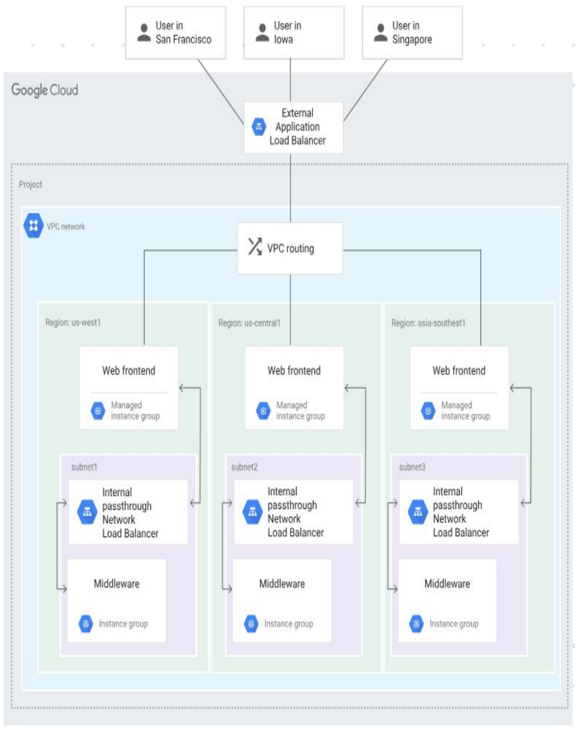
Google Cloud Proxy Network LB Architecture


Web clients
Internal clients
TCP traffic (SSL offload available)
Client traffic terminates here
Proxy Network Load Balancer
New connection from load balancer to backends
Compute Engine
Google Kubernetes Engine
Backends external to Google Cloud
Backends[1]

[1] Backend support differs depending on the deployment mode of the load balancer (internal or external, global or regional).

Google Cloud Passthrough Network LB Architecture

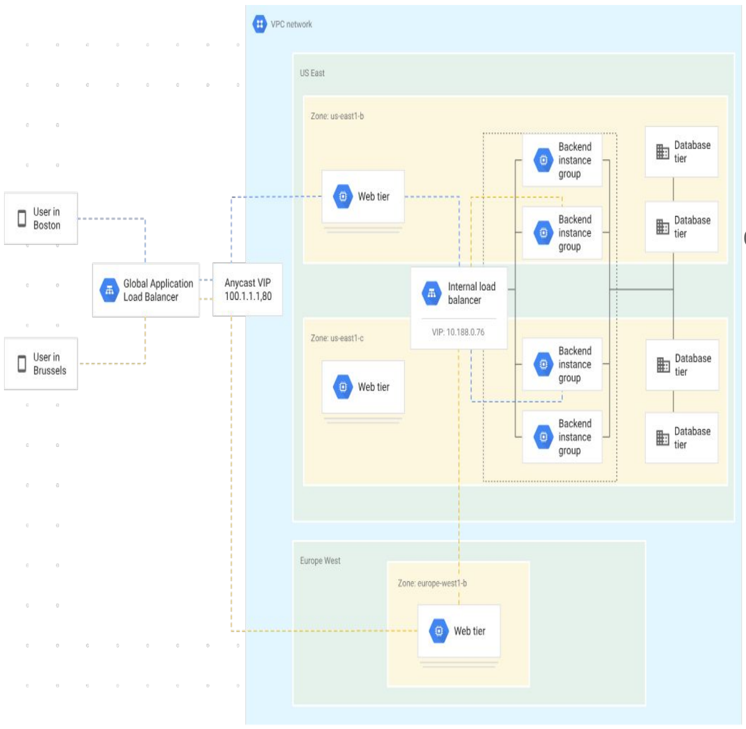




Three-tier web app from Google Cloud with

1. Global External Application LB
2. **Three** Internal Passthrough Network LB (each in each region)

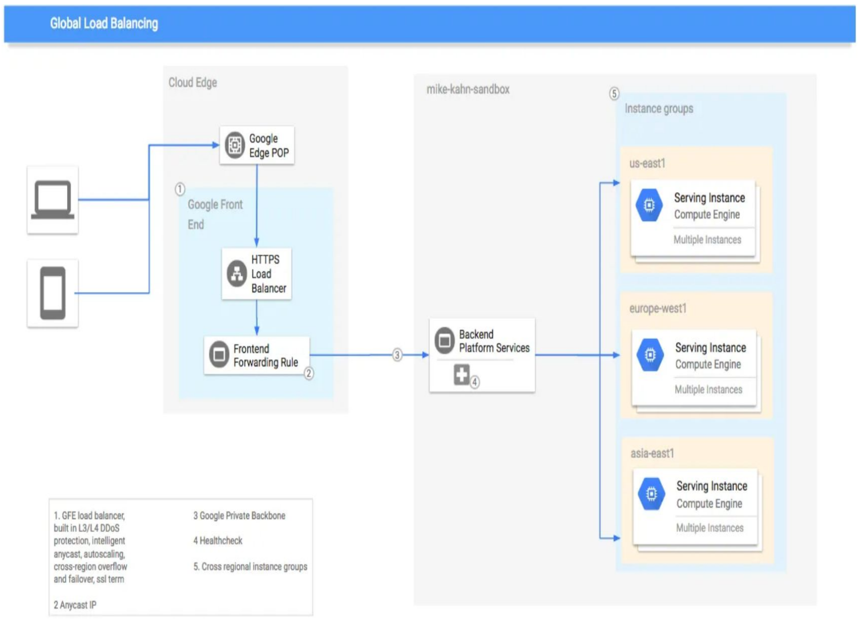
Note: the Database Tier is not shown



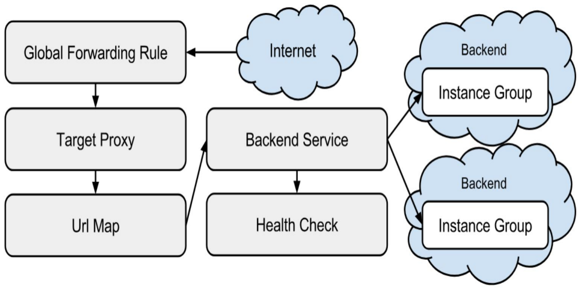
Three-tier web app from Google Cloud with

1. Global External Application LB
2. Internal Passthrough Network LB

Google Cloud Load Balancer: Backend Service Example



Google Cloud Load Balancer: URL Map & Target Proxy



Cloud Load Balancing supports the following resource combinations:

1. Forwarding rule > target HTTPS proxy > URL map > backend services
2. Forwarding rule > target HTTP proxy > URL map > backend services
3. Forwarding rule > target TCP proxy > **one** backend service
4. Forwarding rule > target SSL proxy > **one** backend service

Health checks and backends are not shown in the preceding list.

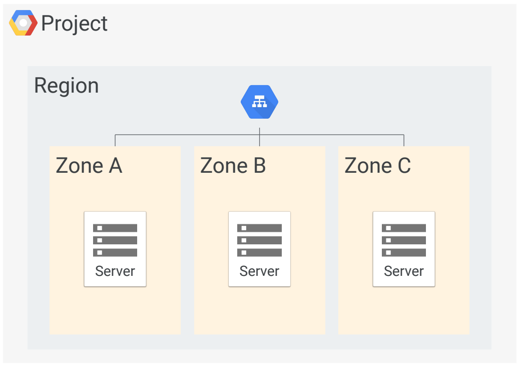
Target Proxy

- When you create an **Application LB** or a **proxy Network LB**, one of the resources that you configure is the target proxy.
- **Target proxies** terminate incoming connections from clients and create new connections from the load balancer to the backends.
- At a high-level, traffic is handled according to the following 4-step process:
  - A client makes a connection to the IP and port of the LB's forwarding rule
  - A **target proxy** is referenced by one or more forwarding rules. The **target proxy** listens on the IP and port specified by the load balancer's forwarding rule.

- The **target proxy** receives the client request.
  - It then compares the request's destination IP and port to the IP and port configured in each forwarding rule that references the target proxy.
  - If a match is found, the **target proxy** terminates the client's network connection.
- The **target proxy** establishes a new connection to the appropriate backend VM or endpoint, as determined by:
  - the load balancer **URL map** (applicable only to Application Load Balancers) and
  - backend service configuration.

High availability is achieved by deploying infrastructure across more than one zone in a region

- In the cloud, a zone is considered a fault boundary:
  - Applications deployed to multiple zones continue to work even if a zone fails.
- Route traffic to application instances through a load balancer:
  - Load balancers are regional (*or sometimes global*) resources.
  - Load balancers would survive a zonal outage.



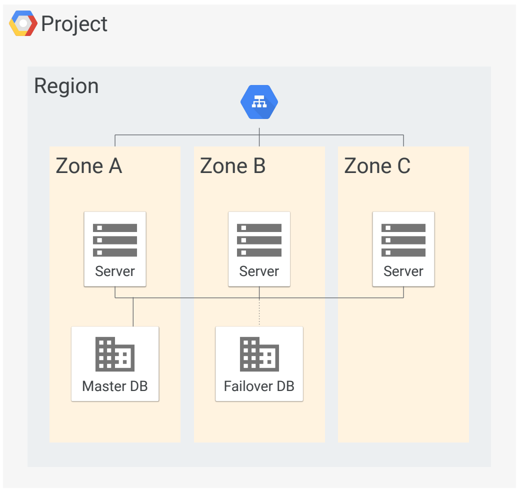
Google Cloud

High availability is achieved by deploying infrastructure across more than one zone in a region. In Google Cloud, each region is divided into three zones. Think of a zone as a fault boundary. No single thing that can fail would cause resources deployed in different zones to be unavailable.

If you're deploying an application like a web app or service, create instances in multiple zones, and then create a load balancer that routes traffic to those backend instances. The load balancers will monitor the health of the instances and only send traffic to healthy ones. The load balancers are services provided by Google and are also fault-tolerant. Load balancers can be either regional or global. In either case, they will survive a zonal outage.

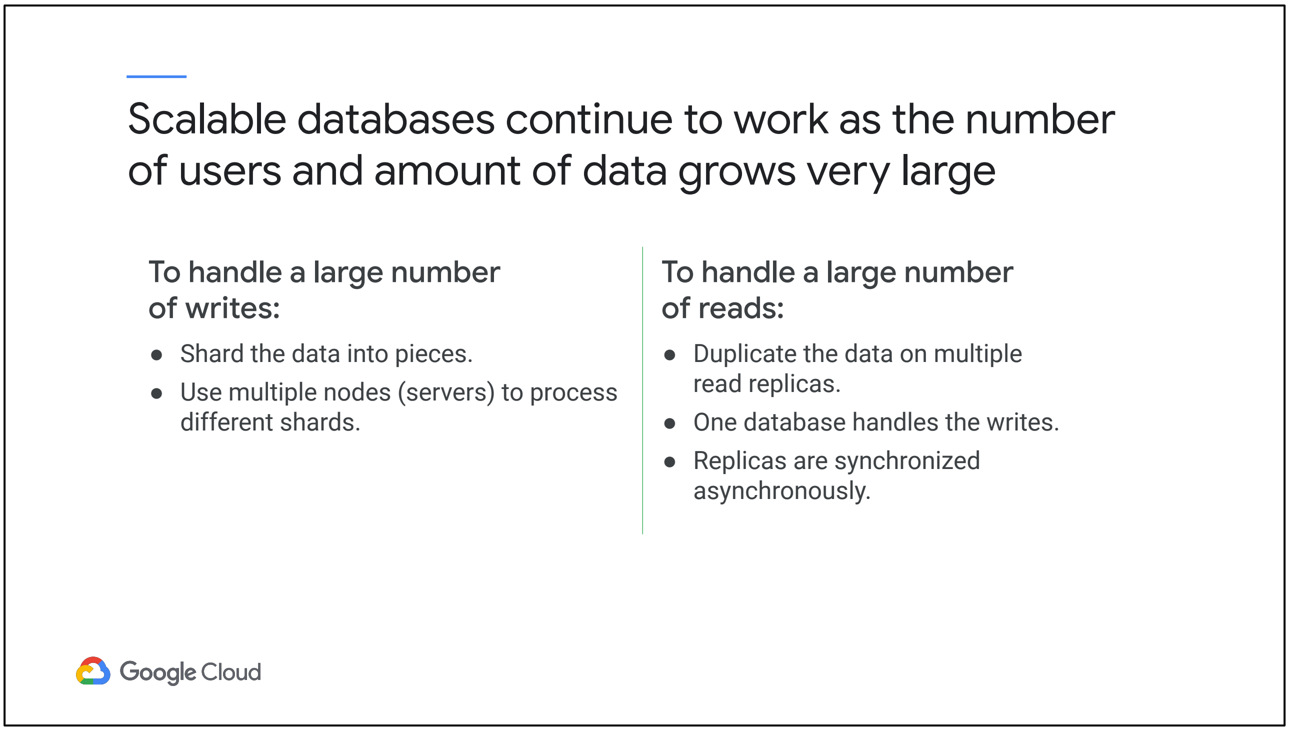
Failover replicas are used for database high availability

- For an application to be highly available, its database must also be highly available.
- Deploy two databases in different zones:
    - All access is routed to the master.
    - The master synchronizes its data with the failover replica.
    - In the event of an outage, the failover becomes the master.



Google Cloud

For your applications to be highly available, your databases must be too. Failover replicas are used to ensure database reliability. Deploy two databases in different zones. One is the master and one is the failover. Requests go to the master. All data is synchronized with the failover. If the master ever goes down, the failover takes requests until the master is back up and running.



Scalable databases continue to work as the number of users and amount of data grows very large

To handle a large number of writes:

- Shard the data into pieces.
- Use multiple nodes (servers) to process different shards.

To handle a large number of reads:

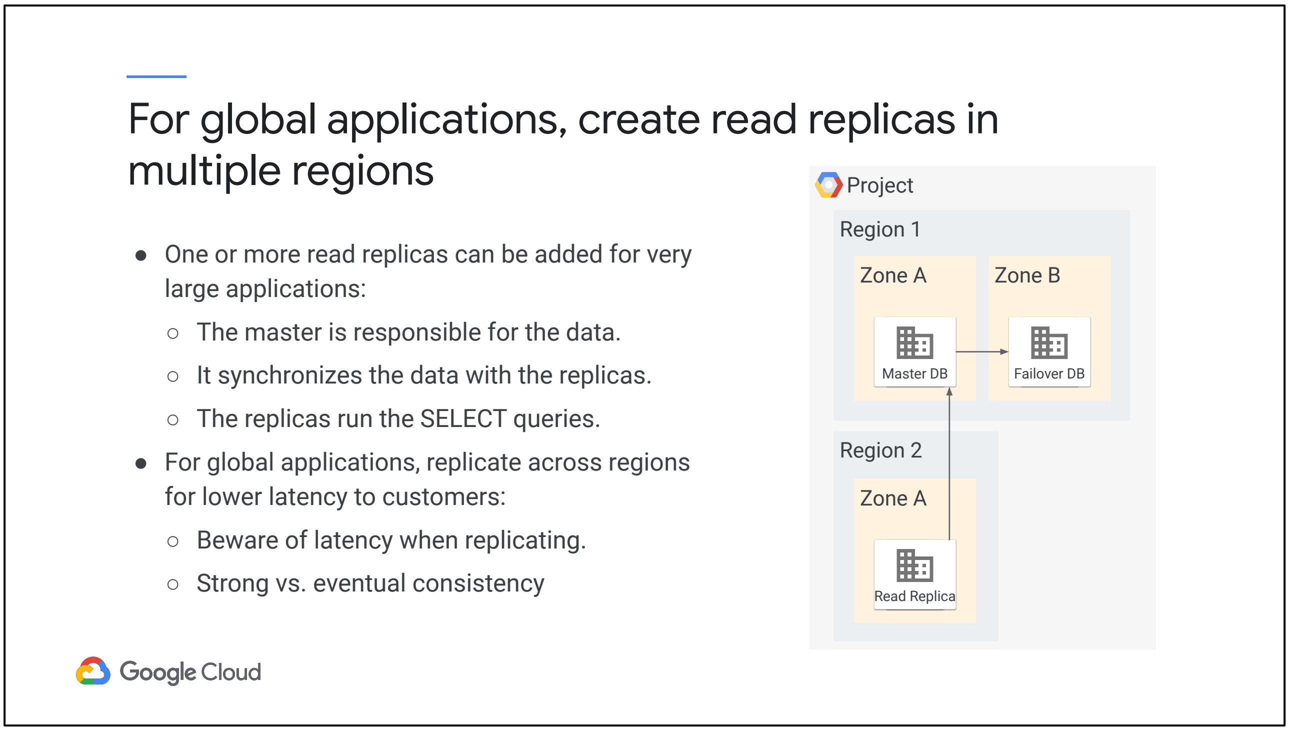
- Duplicate the data on multiple read replicas.
- One database handles the writes.
- Replicas are synchronized asynchronously.

Google Cloud

Scalable databases continue to work as the number of users and amount of data grow very large.

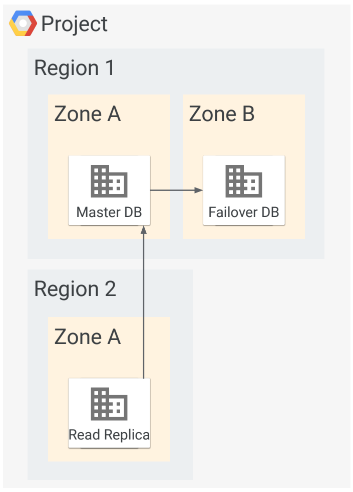
- To handle a **large volume of writes**, you split the database into pieces called shards. Then use multiple nodes, or servers, to process different shards.

- You can process **high volumes of reads** by creating multiple copies of the database called *replicas*. The read replicas handle analytics and reporting uses cases, while the master is left to handle the writes. The master ensures that the data is synchronized with the read replicas when changes are made.



For global applications, create read replicas in multiple regions

- One or more read replicas can be added for very large applications:
  - The master is responsible for the data.
  - It synchronizes the data with the replicas.
  - The replicas run the SELECT queries.
- For global applications, replicate across regions for lower latency to customers:
  - Beware of latency when replicating.
  - Strong vs. eventual consistency



Google Cloud

Customers with users all over the world can create read replicas in multiple regions. Requests from users are routed to the region geographically closest to them. This can be automated using Google's global load balancers.

A master will still handle the writes and synchronize those changes to the replicas. When synchronizing across regions, you just have to be aware of the increased latency around the world. If you are using asynchronous replication across regions, the replicas can have stale data for a short period of time. This is known as *eventual consistency,* as opposed to *immediate consistency*.