SESIÓN 12: Bases de Datos Distribuidas
Arquitectura, Fragmentación y Replicación

INTRODUCCIÓN: EL PROBLEMA REAL

Situación real
Imagina una **empresa global**:

- Usuarios en Europa, América y Asia
- Miles de accesos simultáneos
- Compras 24/7
- No puede permitirse caídas

Pregunta

- ¿Qué pasaría si toda la base de datos estuviera en un único servidor en Madrid?

Problemas inmediatos:

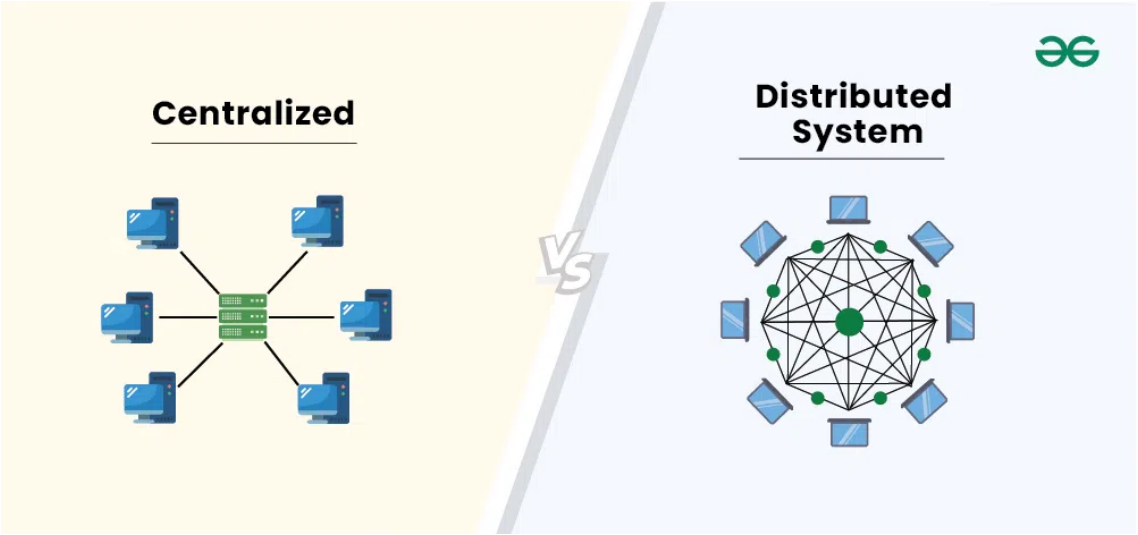
- Latencia alta para usuarios lejanos
- Punto único de fallo
- Saturación del servidor
- Difícil escalado

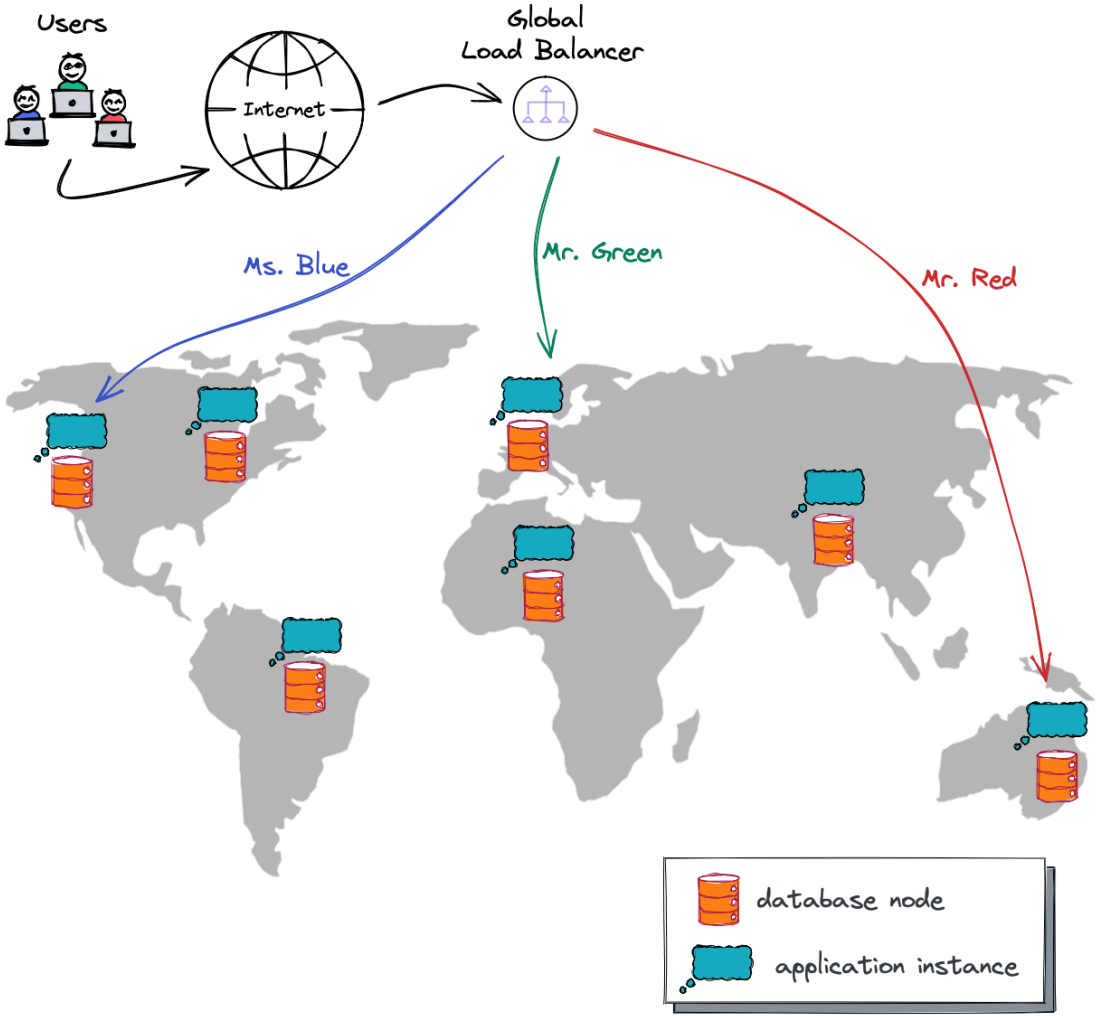
Solución: Bases de datos distribuidas

DEFINICIÓN FORMAL

Una **base de datos distribuida** es un conjunto de bases de datos:

- Lógicamente **únicas**
- Físicamente **repartidas** en distintos nodos
- Conectadas por red
- Gestionadas como si fueran una sola


Centralized
Distributed System
VS


Users
Internet
Global Load Balancer
Ms. Blue
Mr. Green
Mr. Red
database node
application instance

ARQUITECTURA DE BASES DE DATOS DISTRIBUIDAS

COMPONENTES FUNDAMENTALES

Nodos

- Servidores físicos o virtuales
- Cada nodo puede almacenar **datos completos o parciales**

Red

- Internet, red privada o cloud
- Introduce **latencia y fallos**

SGBD distribuido

- Decide dónde están los datos
- Gestiona consultas, transacciones y sincronización

Aplicaciones cliente

- No saben dónde están los datos realmente (**transparencia**)

TIPOS DE ARQUITECTURA

Cliente–Servidor Distribuido

- Clientes : múltiples servidores
- Muy común en aplicaciones empresariales

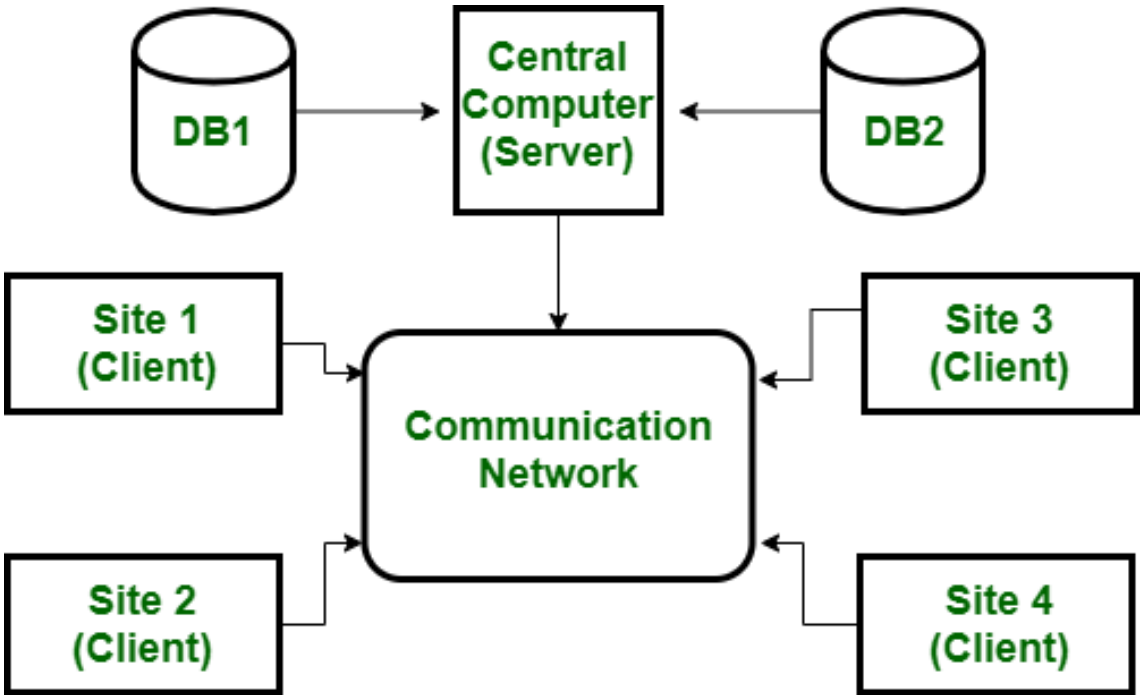
Ejemplo:
Cliente web → Servidor Europa
Cliente móvil → Servidor América

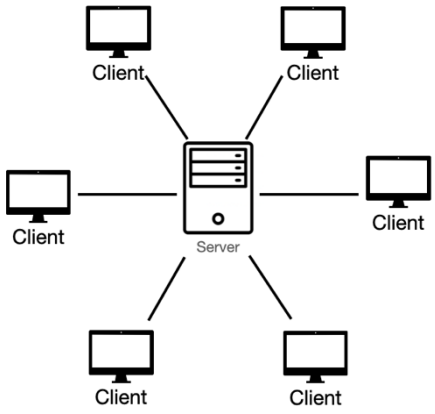
Arquitectura Peer-to-Peer

- Todos los nodos son iguales
- Muy tolerante a fallos
- Más compleja

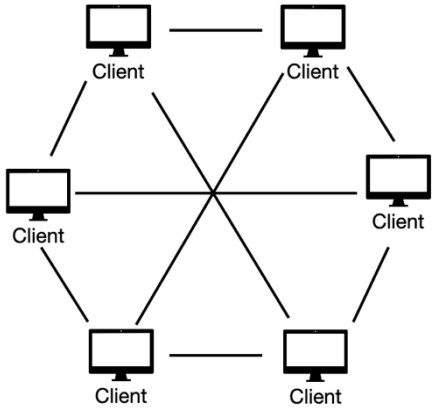
Arquitectura Cloud Distribuida

- Nodos en distintas regiones
- Escalado automático
- Replicación integrada

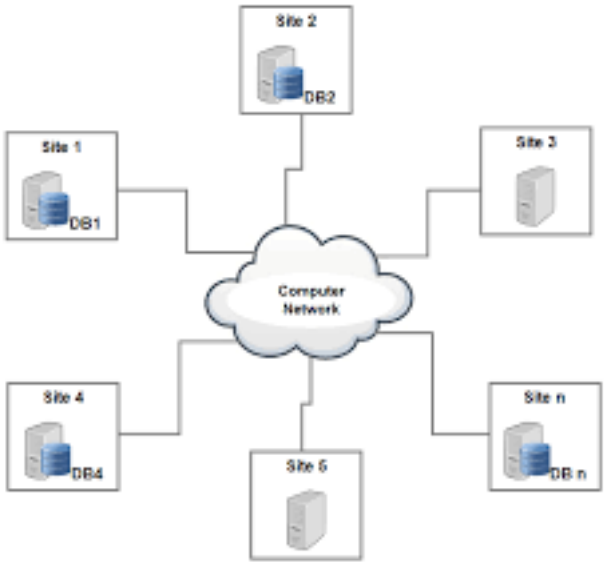




Client Server Architecture



P2P Architecture



FRAGMENTACIÓN DE DATO

La **fragmentación** consiste en **dividir los datos** y almacenarlos en distintos nodos.

El usuario NO ve la fragmentación → **Transparencia**

FRAGMENTACIÓN HORIZONTAL (FILAS)

Se dividen los **registros** de una tabla.

Tabla original
CLIENTES(id, nombre, país)

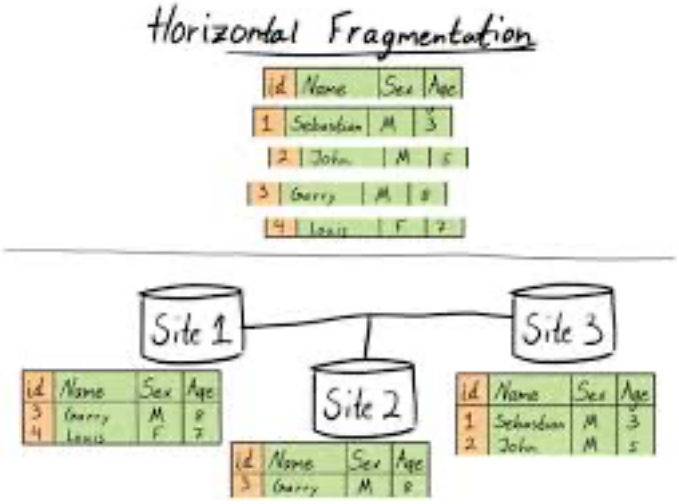
Fragmentación
CLIENTES_ES → país = 'España'
CLIENTES_FR → país = 'Francia'
CLIENTES_DE → país = 'Alemania'

Ejemplo práctico

- Usuarios españoles acceden más rápido a CLIENTES_ES
- Menor tráfico de red
- Consultas locales más eficientes

Uso típico:

- Aplicaciones internacionales
- Datos geográficos



Range-Based Sharding

Product	Price
A	$15
B	$49
C	$78
D	$131

Shard 1

Product	Name
A	$15
B	$49

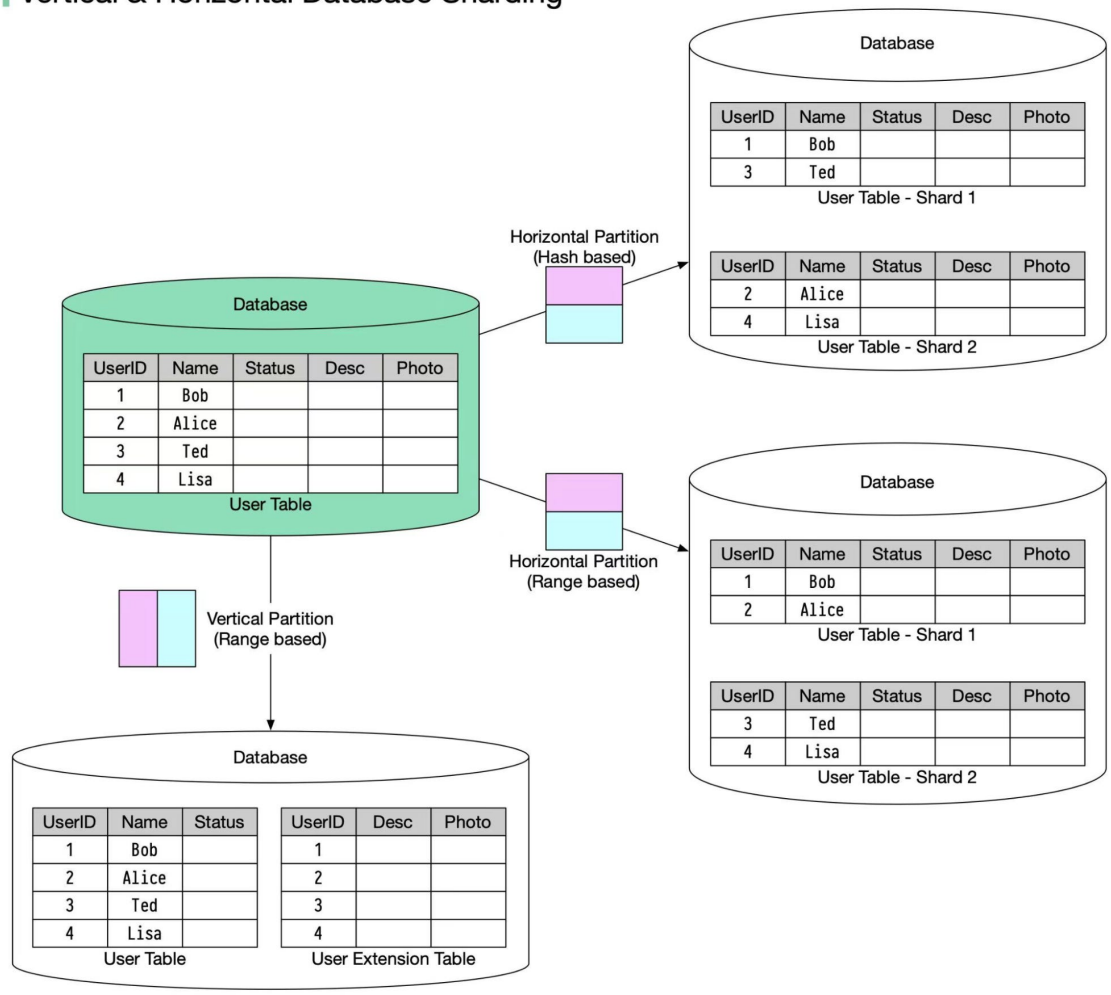
$0 - $75

Shard 2

Product	Name
C	$78
D	$131

$76-150

Vertical & Horizontal Database Sharding



FRAGMENTACIÓN VERTICAL (COLUMNAS)

Se dividen los **atributos** de una tabla.

Tabla original
USUARIOS(id, nombre, email, contraseña, tarjeta)

Fragmentación
USUARIOS_PUBLICO(id, nombre, email)
USUARIOS_PRIVADO(id, contraseña, tarjeta)

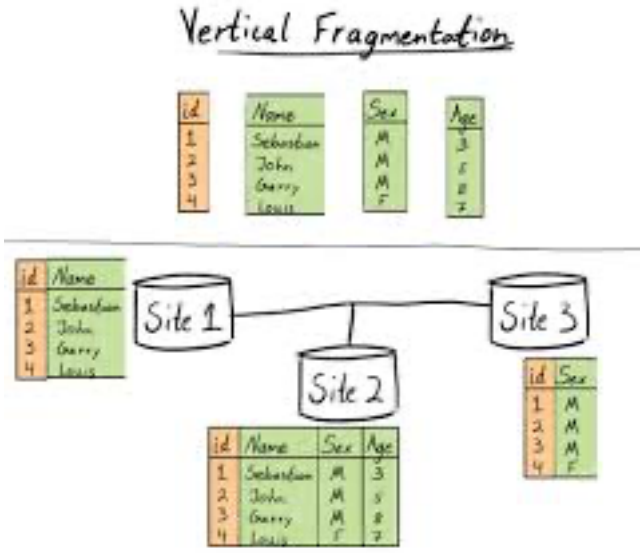
Clave primaria **SIEMPRE presente**.

Ventajas

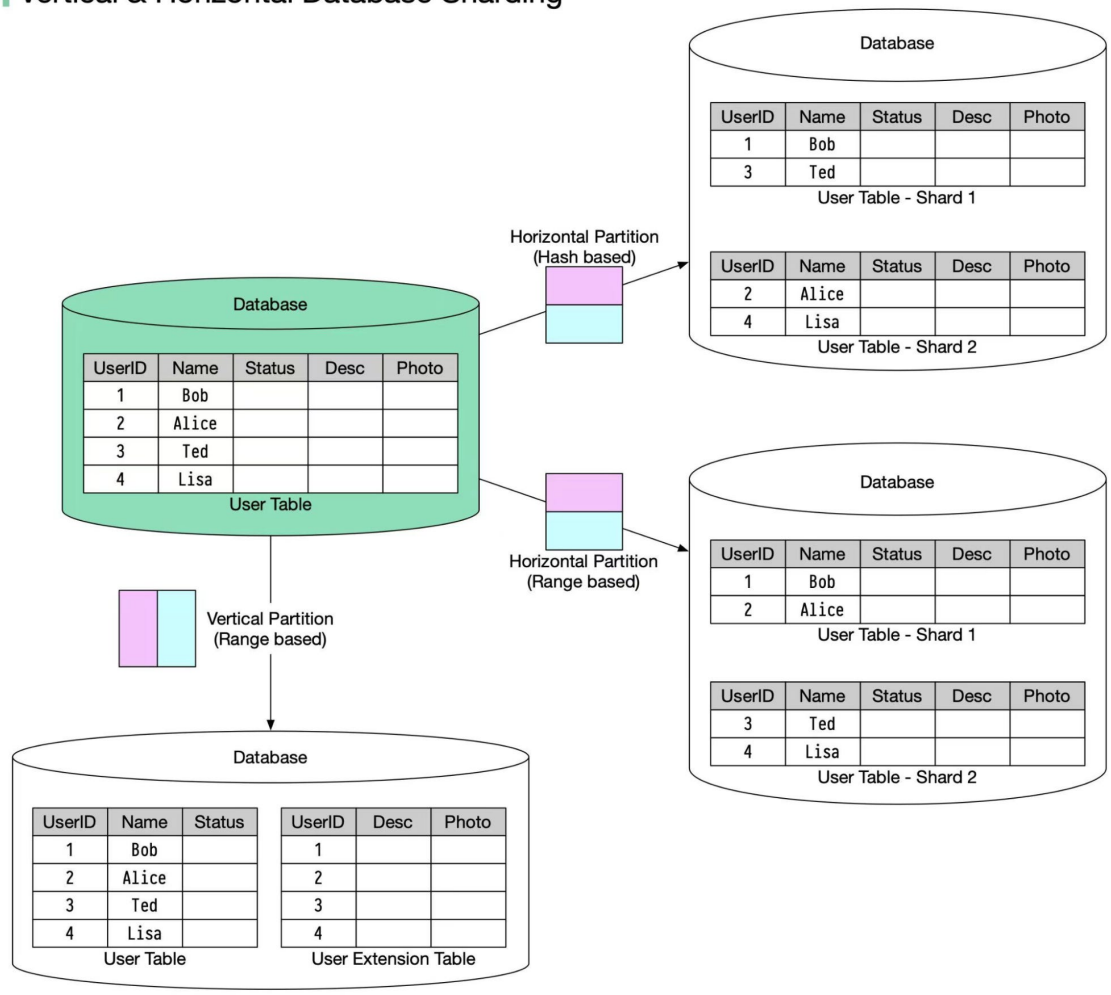
- Mayor seguridad
- Mejor rendimiento (menos datos por consulta)

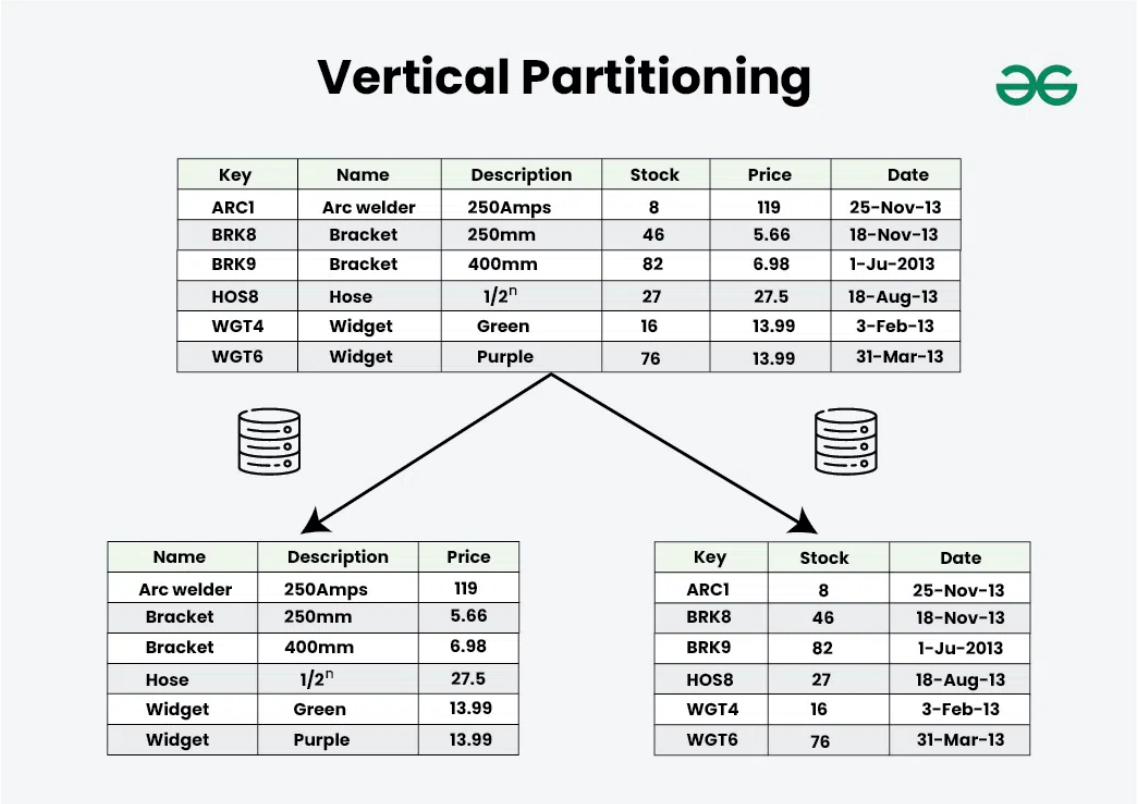
Uso típico:

- Datos sensibles
- Sistemas con distintos niveles de acceso



Vertical & Horizontal Database Sharding





FRAGMENTACIÓN MIXTA

Combinación de ambas.

Ejemplo real:

1. Fragmentación horizontal por país
2. Fragmentación vertical para separar datos sensibles

Muy usada en **grandes plataformas cloud**

PROBLEMAS DE LA FRAGMENTACIÓN

- Consultas complejas (JOIN entre nodos)
- Mayor lógica de reconstrucción
- Planificación clave del diseño

REPLICACIÓN DE DATOS

La **replicación** consiste en **copiar los mismos datos** en varios nodos.

Objetivo:

- Alta disponibilidad
- Tolerancia a fallos
- Mejor rendimiento en lectura

TIPOS DE REPLICACIÓN

Replicación Completa

- Toda la base de datos en todos los nodos
- Mucha redundancia
- Coste elevado

Replicación Parcial

- Solo ciertos fragmentos se replican
- Más eficiente

SEGÚN SINCRONIZACIÓN

Replicación SÍNCRONA

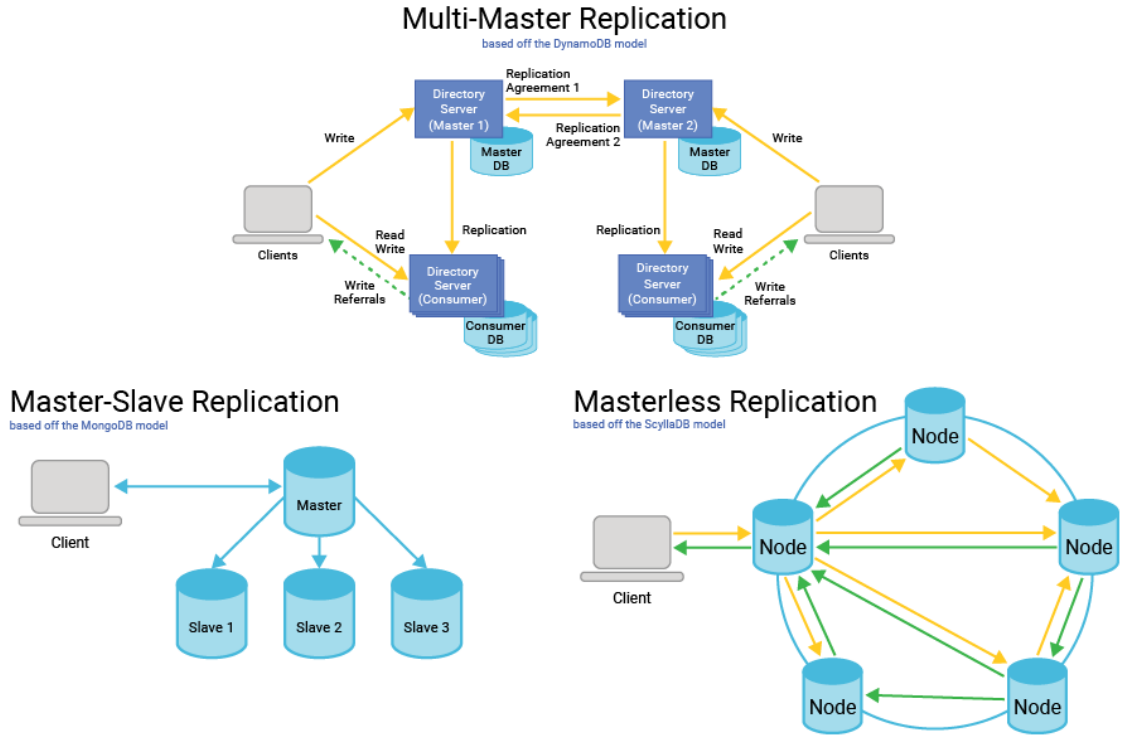
- Escritura en todos los nodos a la vez
- Máxima consistencia
- Más lenta

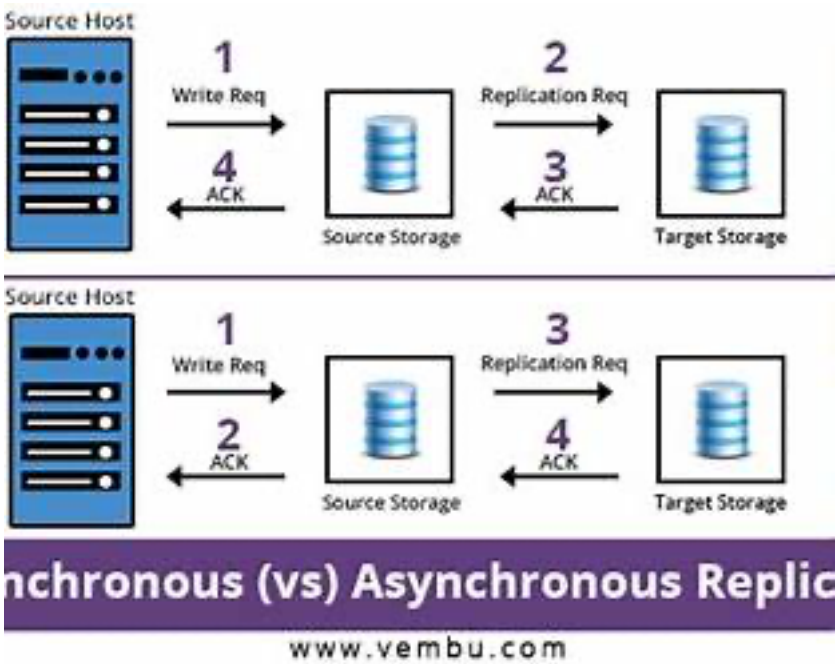
Ejemplo:
INSERT → Nodo A ✓ → Nodo B ✓ → OK

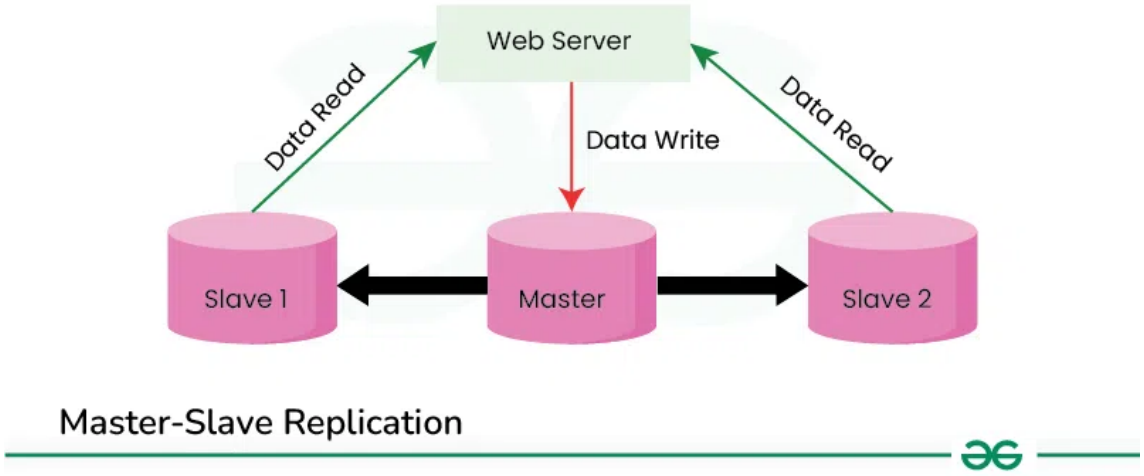
Replicación ASÍNCRONA

- Escritura inmediata
- Sincronización posterior
- Riesgo temporal de inconsistencia

Ejemplo:
INSERT → Nodo A ✓ → OK
↓
Nodo B (después)







PROBLEMA CLAVE: CONSISTENCIA

¿Qué pasa si dos usuarios leen datos distintos?

Aquí aparece el **compromiso** entre:

- Consistencia
- Disponibilidad
- Tolerancia a fallos

No se pueden maximizar las tres a la vez.

BASES DE DATOS DISTRIBUIDAS EN LA NUBE

REALIDAD CLOUD

En cloud:

- Todo está distribuido
- La fragmentación y replicación son automáticas
- El desarrollador **configura**, no implementa
-

Proveedores habituales

- Amazon Web Services
- Google Cloud Platform
- Microsoft Azure

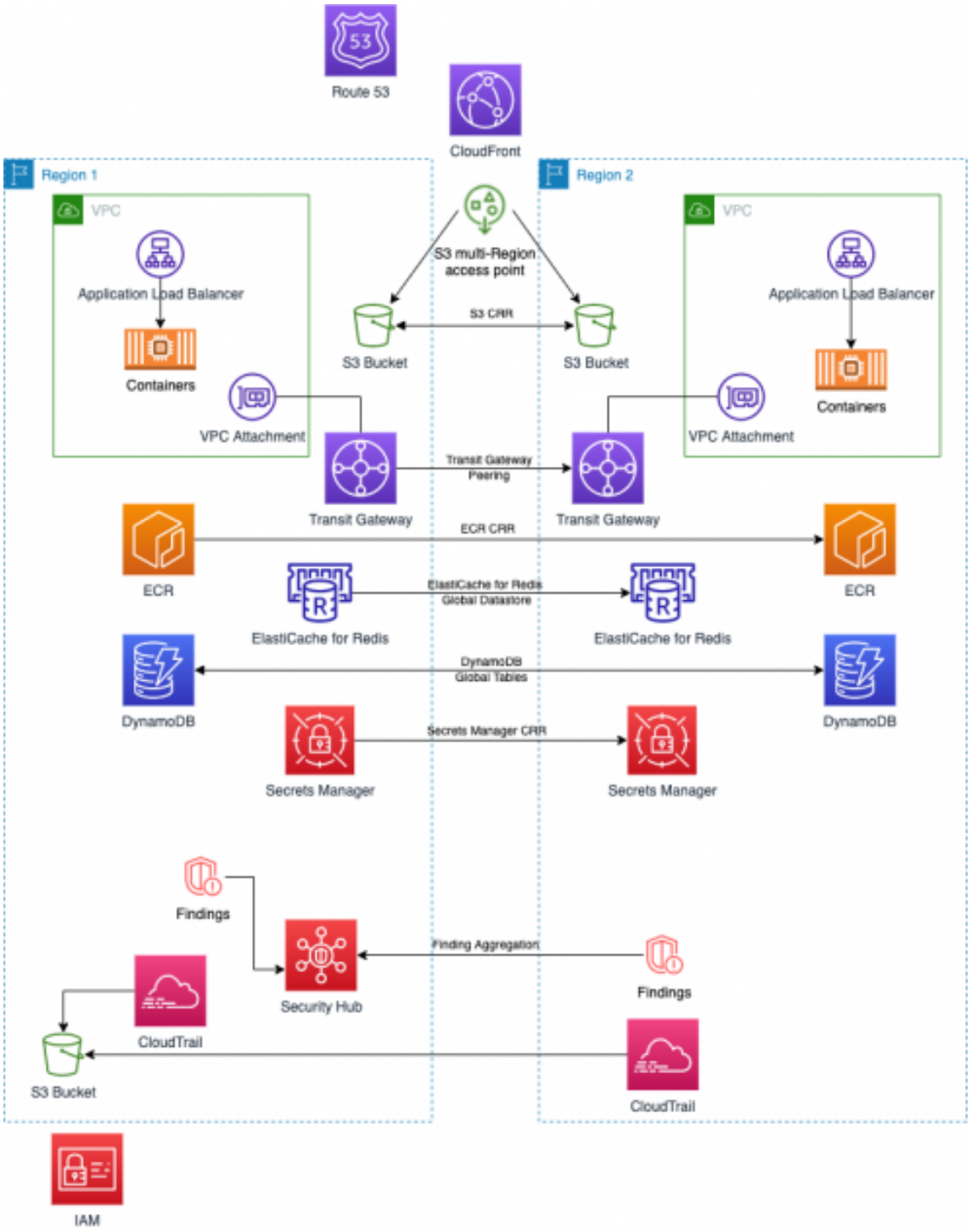
EJEMPLO REAL

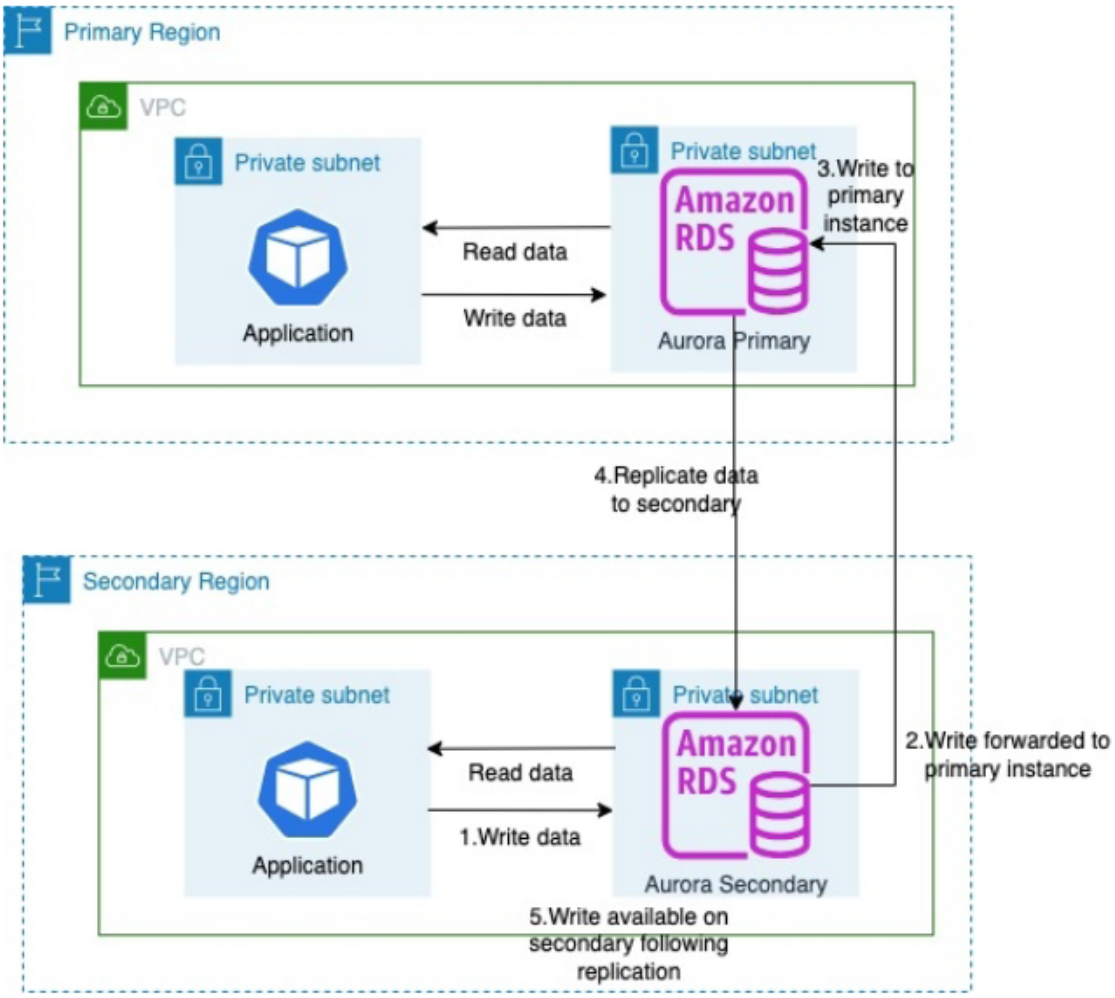
Una base de datos con:

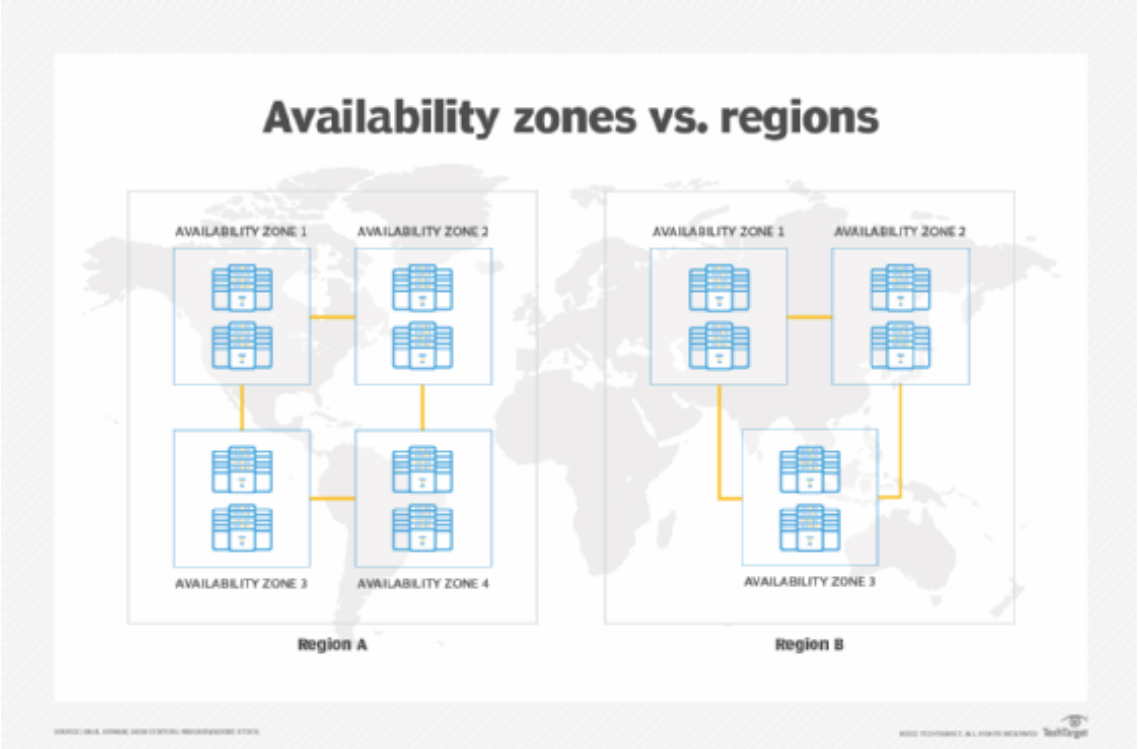
- Región Europa
- Región América
- Región Asia

Datos:

- Usuarios europeos → Europa
- Replicación en América como respaldo
- Replicación asíncrona global


Route 53
CloudFront
Region 1
VPC
Application Load Balancer
Containers
VPC Attachment
S3 multi-Region access point
S3 CRR
S3 Bucket
S3 Bucket
Region 2
VPC
Application Load Balancer
Containers
VPC Attachment
Transit Gateway Peering
Transit Gateway
Transit Gateway
ECR CRR
ECR
ECR
ElastiCache for Redis Global Datastore
ElastiCache for Redis
ElastiCache for Redis
DynamoDB Global Tables
DynamoDB
DynamoDB
Secrets Manager CRR
Secrets Manager
Secrets Manager
Findings
Finding Aggregation
Findings
Security Hub
CloudTrail
S3 Bucket
CloudTrail
IAM


Primary Region
VPC
Private subnet
Application
Read data
Write data
Private subnet
Amazon RDS
Aurora Primary
3.Write to primary instance
4.Replicate data to secondary
Secondary Region
VPC
Private subnet
Application
Read data
1.Write data
Private subnet
Amazon RDS
Aurora Secondary
2.Write forwarded to primary instance
5.Write available on secondary following replication


Availability zones vs. regions
AVAILABILITY ZONE 1
AVAILABILITY ZONE 2
AVAILABILITY ZONE 3
AVAILABILITY ZONE 4
Region A
AVAILABILITY ZONE 1
AVAILABILITY ZONE 2
AVAILABILITY ZONE 3
Region B

RESUMEN

- Distribuir = rendimiento + disponibilidad
- Fragmentar = dividir
- Replicar = copiar
- Cloud = distribución por defecto

Pregunta final
¿Qué es peor: perder rendimiento o perder datos?