

FLASHCARDS

GCP

212 tarjetas de repaso rápido

frente = pregunta, dorso = respuesta · numeradas · 12 subtemas



CHULETA = pregunta clave de examen

GCP - 212

Ordena su nivel abajo a la izquierda; el mazo viene ordenado de lo BÁSICO a lo AVANZADO.

CÓMO USARLO

1 · IMPRIMIR

A4 a doble cara, volteando por el BORDE LARGO, escala 100% (sin "ajustar a página"). Corta por las líneas discontinuas: cada pregunta queda con su respuesta al dorso, verificable por la numeración.

2 · CELULAR

Abre flashcards.html en el navegador (sin internet): toca para voltear, marca «La sé» o «Repasar» y el progreso queda guardado. Incluye búsqueda, filtros por subtema, modo aleatorio y lista completa.

3 · ANKI / QUIZLET

Importa cards.csv (Frente ; Reverso ; Categoría ; Subtema) para estudiar con repetición espaciada.

MUESTRA A ESCALA REDUCIDA

FRENTE (página impar)

REVERSO (página par, espejado)

GCP**G005**

Cloud Shell y Cloud Editor: qué te dan sin instalar nada

GCP**G005**

RESPUESTA

Terminal Linux en el navegador con gcloud/gsutil/kubectl preinstalados y 5GB de home persistente.
Editor web integrado; permite deploy de ejemplo en minutos.
Útil: emergencias desde cualquier equipo, prototipos de APIs, administrar cuando no tienes el SDK local.
Limitación: no para cargas largas (timeout de sesión).

Al imprimir a doble cara, cada respuesta queda exactamente detrás de su pregunta; la numeración lo verifica.

Contenido: 212 tarjetas + chuleta exprés al dorso de esta portada · 56 páginas · generado 12/09/2026

CHULETA EXPRÉS

Lo básico que siempre se pregunta: HTTP, seguridad, testing, datos, Docker/K8s y GCP.

- Verbos HTTP: GET leer · POST crear · PUT reemplaza · PATCH parcial · DELETE borra. Idempotentes: GET/PUT/DELETE. Seguro: GET.
- Códigos HTTP: 200 OK · 201 Created (+Location) · 204 sin body · 400 validación · 401 no autenticado · 403 sin permiso · 404 no existe · 409 conflicto · 429 rate limit · 500 bug · 503 caído.
- ACID: Atomicidad, Consistencia, Aislamiento, Durabilidad. CAP: ante partición (P) eliges consistencia (C) o disponibilidad (A).
- Escalera REST (Richardson): 0 RPC/POX · 1 recursos · 2 verbos + códigos HTTP · 3 HATEOAS.
- JWT: header.payload.signature; firma HS256 (simétrica) o RS256 (asimétrica). Claims clave: sub, iss, aud, exp, iat. Nunca guardar datos sensibles: es legible (base64).
- OAuth2: Authorization Code + PKCE para apps/SPA · Client Credentials para servicio-a-servicio · Refresh Token para renovar sin re-login.
- SQL rápido: índices B-tree aceleran WHERE/JOIN · EXPLAIN para ver el plan · N+1 se resuelve con JOIN FETCH / batch · paginar con keyset, no solo OFFSET.
- Concurrencia en BD: optimista = @Version + retry (mejor para poca contención) · pesimista = SELECT FOR UPDATE (contención alta).
- Pool de conexiones: HikariCP (default en Boot). Ajustar maximumPoolSize (10-20 suele bastar) y connectionTimeout.
- Caché HTTP: Cache-Control (max-age, no-cache) + ETag/If-None-Match → 304 Not Modified. Caché de app: Spring @Cacheable.
- JUnit 5: @Test, @BeforeEach, AssertJ assertThat. Mockito: @Mock + @InjectMocks. Tipos de test Spring: @WebMvcTest (capa web), @DataJpaTest (repositorios), @SpringBootTest (integración completa).
- TDD: rojo (test que falla) → verde (código mínimo) → refactor. Pirámide: muchos unit, algunos integration, pocos E2E.
- SOLID: S una razón de cambio por clase · O abierto a extensión, cerrado a modificación · L subtipos sustituibles · I interfaces pequeñas · D depender de abstracciones.
- GoF que salen siempre: Singleton, Factory Method, Builder, Strategy, Adapter, Observer, Decorator, Proxy (Spring AOP usa proxies dinámicos).
- Maven: coordenadas groupId:artifactId:version · scopes: compile, provided, runtime, test · BOM para alinear versiones · mvnw wrapper para versionar Maven.
- Docker: imagen inmutable por capas con cache · multi-stage build para imagen liviana · no correr como root · ENTRYPOINT vs CMD · healthcheck.
- Kubernetes: Pod (unidad mínima) · Deployment (réplicas + rolling update) · Service (IP estable/DNS) · Ingress (rutas HTTP) · HPA (autoescalado) · ConfigMap/Secret (config).
- Logging: niveles TRACE < DEBUG < INFO < WARN < ERROR. Estructurado (JSON) y con correlation ID. Nunca datos personales ni secretos en logs.
- CORS: el navegador lanza preflight OPTIONS; el servidor responde con Access-Control-Allow-Origin/Methods/Headers. Es un mecanismo del navegador, no del servidor.
- Git/entrega: feature branches + Pull Request · squash merge · Conventional Commits (feat/fix/chore) · SemVer MAYOR.MINOR.PATCH (mayor = rompe compatibilidad).
- Rate limiting: 429 Too Many Requests + header Retry-After. Algoritmos: token bucket, sliding window. Se aplica en gateway.
- Async en Spring: @Async + @EnableAsync (executor), @EventListener para eventos en proceso, CompletableFuture para composición.
- Programación reactiva (WebFlux): Mono (0..1) y Flux (0..N), backpressure, no bloquea hilos; útil con mucha I/O concurrente. MVC por hilos sigue siendo válido para CRUD normal.
- Microservicios: contrato primero (OpenAPI), consumer-driven contract tests (Pact), CI/CD por servicio, feature flags para desacoplar deploy de release.
- En la nube: 12-factor (config en entorno, procesos stateless, logs como stream), probes de salud, escalado horizontal primero, idempotencia en todo endpoint que escribe.
- K8s probes: liveness reinicia el pod, readiness lo saca del balanceo, startup da margen a arranques lentos. Requests/limits de CPU-memoria definen la calidad de servicio.
- JOINs SQL: INNER (solo coincidencias) · LEFT (toda la izquierda + coincidencias) · RIGHT · FULL. Subconsultas correlacionadas fila a fila = sospechar de N+1.
- HTTP/2: multiplexa streams en una conexión, comprime headers (HPACK) y permite prioridades; gRPC exige HTTP/2. HTTP/3 va sobre QUIC/UDP.
- Complejidad: HashMap get/put O(1) medio · búsqueda binaria O(log n) · ordenar O(n log n) · recorrer O(n). ArrayList contains O(n).
- Colecciones thread-safe: ConcurrentHashMap (no bloquea el mapa entero) · CopyOnWriteArrayList (muchas lecturas) · BlockingQueue (productor/consumidor).
- Bean Validation: @Valid en el parámetro + @NotNull/@Size/@Email/@Pattern en el DTO; si falla, MethodArgumentNotValidException → 400 con detalles.
- Clientes REST en Spring: RestClient (síncrono, moderno) · WebClient (reactivo) · OpenFeign (declarativo por interfaz). Configura siempre timeout y retry.
- Docker Compose levanta el stack local (app + BD + cola) con docker compose up; perfiles para servicios opcionales. En prod: Kubernetes o Cloud Run.
- GCP rápido: proyecto = unidad de facturación/IAM · región (multi-zona) vs zona · Cloud Run = contenedor serverless · Pub/Sub = at-least-once · IAM = least privilege · ADC para autenticar apps sin claves.

GCP

★

Fundamentos y organización

G001

Jerarquía de recursos en GCP

BASICO

GCP

★

Fundamentos y organización

G002

Presupuestos y optimización de costos en GCP

BASICO

GCP

Fundamentos y organización

G003

¿Qué es un 'proyecto' en GCP?

BASICO

GCP

Fundamentos y organización

G004

Región vs zona vs multi-región: decisiones de arquitectura

BASICO

GCP

Fundamentos y organización

G005

Cloud Shell y Cloud Editor: qué te dan sin instalar nada

BASICO

GCP

Fundamentos y organización

G006

gcloud: configuración, proyecciones y comandos esenciales

BASICO

GCP

Fundamentos y organización

G007

Billing export: cómo saber quién gasta qué

BASICO

GCP

Fundamentos y organización

G008

Committed Use Discounts vs Spot VMs vs Sustained Use

BASICO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP ★
RESPUESTA

Fundamentos y organización

G002

Budgets con alertas (email/Pub/Sub) · labels para atribuir costo por equipo/app · informes de Billing.

Ahorro: committed use discounts (1-3 años), VMs Spot para carga interrumpible, apagar dev de noche, right-sizing recommendations, Cloud Run escala a 0.

BASICO

GCP ★
RESPUESTA

Fundamentos y organización

G001

Organization → Folders → Projects → Resources.

Los proyectos aíslan facturación, cuotas, IAM y APIs habilitadas.

Las políticas (IAM y org policies) se HEREDAN hacia abajo.

Buena práctica: un proyecto por entorno (dev/stage/prod) o dominio, carpetas por equipo.

BASICO

GCP
RESPUESTA

Fundamentos y organización

G004

Zona: datacenter físico (latencia mínima, riesgo de caída).

Región: conjunto de zonas independientes (latencia baja entre zonas <2ms).

Multi-región: conjunto de regiones — máxima disponibilidad y cercanía global (Cloud Storage multi-region, Spanner).

Diseño: producción crítica = mínimo 2 zonas; global = multi-región para datos, LB global para tráfico.

BASICO

GCP
RESPUESTA

Fundamentos y organización

G003

Unidad base de organización: dentro viven todos los recursos;

define cuotas, facturación, IAM base y qué APIs están habilitadas.

Cada recurso pertenece exactamente a un proyecto.

Patrón habitual: proyectos separados por entorno (dev/stage/prod) bajo la misma organización.

BASICO

GCP
RESPUESTA

Fundamentos y organización

G006

gcloud init / config set project ID; config list.

Proyectos: gcloud projects create/list.

Compute: gcloud compute instances list/create; Run: gcloud run deploy svc --image ... --region ...

Formato: --format="value(name,zone)" o --format=json; --filter="status=RUNNING".

Componentes: gcloud components update.

BASICO

GCP
RESPUESTA

Fundamentos y organización

G005

Terminal Linux en el navegador con gcloud/gsuite/kubectl preinstalados y 5GB de home persistente.

Editor web integrado; permite deploy de ejemplo en minutos.

Útil: emergencias desde cualquier equipo, prototipos de APIs, administrar cuando no tienes el SDK local.

Limitación: no para cargas largas (timeout de sesión).

BASICO

GCP
RESPUESTA

Fundamentos y organización

G008

Committed (CUD): 1-3 años por uso estable → hasta 57% off (paga compromiso de vCPU/RAM).

Spot: hasta 91% off pero PUEDE ser evicted — solo cargas tolerantes a fallos (batch, workers con retry).

Sustained use: descuento automático por uso mensual alto (solo en algunas regiones/máquinas).

Estrategia: base en CUD, picos en estándar, batch en Spot.

BASICO

GCP
RESPUESTA

Fundamentos y organización

G007

BigQuery export detallado: Billing > Export → tabla de línea por SKUs con labels, proyecto, servicios.

Queries: costo por label team=api en el mes, tendencia día a día.

Budgets con alertas (50/90/100%) y notificación a Pub/Sub para acciones automáticas.

Recommender sugiere rightsizing y committed use.

BASICO

GCP

★

Fundamentos y organización

G009

Recommender y right-sizing: cómo bajar la factura sin romper nada

BASICO

GCP

Fundamentos y organización

G010

Infraestructura como código: por qué Terraform y no console

BASICO

GCP

Fundamentos y organización

G011

Cloud Resource Manager y jerarquía por API

BASICO

GCP

Fundamentos y organización

G012

Soporte GCP y roles de emergencia (break-glass)

BASICO

GCP

★

Fundamentos y organización

G013

¿Qué es la nube? IaaS vs PaaS vs SaaS (con ejemplos GCP)

BASICO

GCP

Fundamentos y organización

G014

CapEx vs OpEx: por qué las empresas van a la nube

BASICO

GCP

Fundamentos y organización

G015

Firebase vs Google Cloud: ¿qué es y cuándo usar cada uno?

BASICO

GCP

Fundamentos y organización

G016

¿Cuántos proyectos y cómo organizarlos? (patrón de referencia)

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

RESPUESTA

Fundamentos y organización

G010

Console manual = estado invisible, irreproducible, sin code review, drift.

Terraform: HCL declarativo, plan antes de apply, state como fuente de verdad, módulos reutilizables, CI/CD de infra.

Google provider: google_compute_instance, google_cloud_run_service...

Regla de oro: si no está en Terraform, no existe.

BASICO

GCP

RESPUESTA

Fundamentos y organización

G009

Active Assist Recommender: sugiere instancias sobredimensionadas (p95 de CPU/mem), imágenes huérfanas, IPs sin usar.

IDLE resources: discos sin attach, IPs estáticas sin usar (¡cobran!)

Aplicar con métricas a mano: cambiar machine type requiere stop/start (o live migration en algunos casos).

Reporte mensual de ahorros = credibilidad con finance.

BASICO

GCP

RESPUESTA

Fundamentos y organización

G012

Planos de soporte: Basic, Standard, Enhanced, Premium (SLA de respuesta).

Break-glass: cuenta admin de emergencia con credencial offline sellada, alertada al usarse, auditable (log de admin activity).

Runbooks con escalamiento y contactos de soporte versionados junto al código.

BASICO

GCP

RESPUESTA

Fundamentos y organización

G011

APIs: projects.create, folders.create, setIamPolicy — automatiza el bootstrap de entornos.

Patrón: repo de infraestructura crea proyecto + APIs + budget + SA con Terraform; equipos solo despliegan apps.

Límites: 100 etiquetas? no — recursos por proyecto varían por servicio; API rate limits a considerar en tooling masivo.

BASICO

GCP

RESPUESTA

Fundamentos y organización

G014

CapEx: comprar servidores (inversión grande adelantada, sobredimensionas para picos, te quedas corto igual).

OpEx: pagar por uso (cloud) — escala con la demanda, apagas lo que no usas, cero mantenimiento de hardware.

Otros motores: time-to-market (minutos vs semanas), regiones globales el mismo día, servicios gestionados (BD, ML) que un equipo chico no podría operar.

Contra: costo variable hay que GOBERNarlo (budgets, labels).

BASICO

GCP

RESPUESTA

Fundamentos y organización

G013

Alquilar cómputo gestionado por otro en lugar de comprar servidores.

IaaS (infraestructura): TÚ gestionas SO + apps — Compute Engine.

PaaS (plataforma): gestionas SOLO tu código — App Engine, Cloud Run.

SaaS (software): usas el producto terminado — Gmail, Workspace.

Regla mental: cuanto más 'aaS', menos infraestructura mantienes y más rápido entregas.

BASICO

GCP

RESPUESTA

Fundamentos y organización

G016

Por límite de cuota/aislamiento y por ciclo de vida: host proyecto de red (Shared VPC), un proyecto por entorno (dev/stage/prod) por servicio, proyectos de sistema (logging, billing, artifact registry).

Carpetas por equipo o por producto bajo la Organization.

Ventaja: blast radius pequeño, IAM granular, presupuestos por proyecto.

INTERMEDIO

GCP

RESPUESTA

Fundamentos y organización

G015

Firebase: plataforma para apps MÓVILES y web — Auth, Firestore, Realtime DB, FCM (push), Hosting, Analytics. Pensada para desarrolladores de app, sin tocar infraestructura.

Google Cloud: infraestructura y servicios backend para CUALQUIER carga (VMs, Kubernetes, BigQuery...).

Se combinan: frontend móvil con Firebase Auth + backend API en Cloud Run + datos en Cloud SQL.

Firebase comparte infraestructura GCP (mismo IAM/Firestore).

BASICO

GCP

Fundamentos y organización

G017

Cuotas y límites: qué son y cómo gestionarlos

INTERMEDIO

GCP

Fundamentos y organización

G018

Labels vs tags vs nombres: qué usar para qué

INTERMEDIO

GCP

Fundamentos y organización

G019

Organization policies vs IAM policies

INTERMEDIO

GCP

Fundamentos y organización

G020

SLAs reales de los servicios: qué esperara tu arquitectura

INTERMEDIO

GCP

Fundamentos y organización

G021

Regiones: cómo elegir entre latencia, costo y residencia de datos

INTERMEDIO

GCP

Fundamentos y organización

G022

¿Qué es el Well Architecture Framework de GCP? (5 pilares)

INTERMEDIO

GCP

Fundamentos y organización

G023

Cloud APIs: quotas, rate limits y manejo de errores

INTERMEDIO

GCP

Fundamentos y organización

G024

Cloud Marketplace y deployments terceros: riesgos

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Fundamentos y organización

G018

RESPUESTA

Labels: pares key-value en recursos FACTURABLES — cost attribution (team=api, env=prod); heredables a Billing export.
 Network tags: para FIREWALL rules de Compute (target-tags).
 Resource name: identificador inmutable del recurso.
 Gobernanza: org policy que exija labels obligatorios en instancias nuevas.

INTERMEDIO

GCP

Fundamentos y organización

G017

RESPUESTA

Techo por recurso/servicio por proyecto-región (CPUs, IPs, API QPS) — protege de errores de costo y abuso.
 Ver: gcloud compute project-info describe, console IAM > Quotas.
 Pedir aumento: console quota request (algunos instantáneos).
 Diseño: no dependas de cuotas altas sin alertas (403 QUOTA_EXCEEDED en producción duele).

INTERMEDIO

GCP

Fundamentos y organización

G020

RESPUESTA

Compute Engine 99.9% (instancia), multi-zona MIG 99.99%; Cloud Storage multi-region 99.95% (SLA de clase); Cloud Run 99.95% regional; Spanner 99.999% multi-región.
 Tu SLA compuesto = producto de los SLAs de los componentes del path crítico.
 Diseño por demanda: cada 9 extra cuesta complejidad — no todo necesita 99.99%.

INTERMEDIO

GCP

Fundamentos y organización

G019

RESPUESTA

IAM: QUIÉN puede hacer QUÉ (permisos, aditivo).
 Org policy: QUÉ se permite hacer EN GENERAL (restricciones): solo IPs externas permitidas, solo ubicaciones específicas (gcp.resourceLocations), deshabilitar service account key creation, VMs sin IP pública.
 Se heredan por la jerarquía y aplican a TODOS (incluso owners) — guardrails de compliance.

INTERMEDIO

GCP

Fundamentos y organización

G022

RESPUESTA

Operational excellence (IaC, observabilidad, automatización), Security (least privilege, encryption, defense in depth), Reliability (HA, DR, cuotas), Performance (escalar, caching, regiones), Cost optimization (CUDs, right-size, apagar).
 Usarlo como checklist de revisión de arquitectura antes de lanzar.

INTERMEDIO

GCP

Fundamentos y organización

G021

RESPUESTA

Prioridad: 1) requisitos legales de residencia (PII en país X), 2) latencia a usuarios, 3) costo (regiones caras vs baratas hasta 2x), 4) catálogo de servicios disponible.
 Herramientas: ping/latency de Cloud CDNs, pricing calculator, página de regiones por servicio.
 Multi-región para datos + routing global resuelve mayoría de casos.

INTERMEDIO

GCP

Fundamentos y organización

G024

RESPUESTA

Marketplace: soluciones empaquetadas (DBs partners, software comercial) desplegadas en tu proyecto con billing integrado.
 Riesgos: revisar qué Service Accounts y permisos pide, costos de licencia mensual, soporte del vendor.
 Para POCs acelera; para core business evaluar antes el managed nativo.

INTERMEDIO

GCP

Fundamentos y organización

G023

RESPUESTA

Cada API: límites de QPS por proyecto. 429/403 RESOURCE_EXHAUSTED → exponential backoff + retry con jitter (client libs lo hacen por defecto).
 Errores gRPC comunes: UNAVAILABLE (retry transitorio), PERMISSION_DENIED (IAM), FAILED_PRECONDITION (APIs no habilitadas).
 Cliente lib vs REST puro: libs manejan auth/retries/ batching.

INTERMEDIO

GCP

Fundamentos y organización

G025

Resource hierarchy anti-patrones

INTERMEDIO

GCP

Fundamentos y organización

G026

Terraform state: por qué importa y buenas prácticas

AVANZADO

GCP

Fundamentos y organización

G027

Disaster recovery: RTO/RPO y niveles de estrategia GCP

AVANZADO

GCP

Fundamentos y organización

G028

Migración de cuentas y organización enterprise: Landing Zone

AVANZADO

GCP★

IAM y seguridad

G029

¿Cómo funciona IAM en GCP?

BASICO

GCP

IAM y seguridad

G030

Service Accounts: ¿qué son y cómo se usan bien?

INTERMEDIO

GCP★

IAM y seguridad

G031

¿Para qué sirve Secret Manager?

INTERMEDIO

GCP

IAM y seguridad

G032

Autenticación de usuarios finales vs de servicios

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Fundamentos y organización

G026

RESPUESTA

State guarda el mapeo recursos-reales → código.
 Remoto + lockeado: bucket GCS con versioning (backend "gcs") — evita corruption y race conditions en equipo.
 Nunca editar a mano; import para recursos existentes (terraformer/import block).
 State contiene secretos → bucket privado + acceso controlado.

AVANZADO

GCP

Fundamentos y organización

G025

RESPUESTA

Un solo proyecto gigante para todo (cuotas, blast radius, permisos groseros).
 Un proyecto por VM/servicio mínimo (operación imposible).
 Carpetas por TIPO de recurso en vez de por equipo/producto (IAM no escala).
 Entornos mezclados en el mismo proyecto (dev junto a prod = desastre).
 Regla: cambiar algo de dev NO debe poder tocar jamás a prod.

INTERMEDIO

GCP

Fundamentos y organización

G028

RESPUESTA

Landing zone: la base lista antes de crear proyectos —
 Organization, carpetas, org policies base (no public IPs, no SA keys), jerarquía de billing, Shared VPC host, logging central, Terraform bootstrap (seed project + state bucket + CI).
 Blueprints: module terraform-google-project-factory.
 Sin landing zone, cada equipo inventa su seguridad — deuda permanente.

AVANZADO

GCP

Fundamentos y organización

G027

RESPUESTA

RTO: tiempo para recuperar; RPO: datos máximos a perder.
 Niveles: backup & restore (horas/días), pilot light (BD replicada, cómputo apagado — minutos), warm standby (réplica chica corriendo), hot/active-active (multi-región — segundos).
 Herramientas: snapshots cross-region de Cloud SQL, GCS dual/multi-region, Spanner multi-región, DNS failover.
 Probar el DR (gameday) o no existe.

AVANZADO

GCP

IAM y seguridad

G030

RESPUESTA

Identidad para CÓDIGO, no personas (email: sa@proyecto.iam.gserviceaccount.com).
 A cada workload (Cloud Run, GKE, Compute) le adjuntas una SA con roles mínimos; la app autentica con ADC sin claves.
 Evita descargar JSON keys; si son inevitables: rotar y guardar en Secret Manager.
 Impersonation (actAs) para tareas puntuales.

INTERMEDIO

GCP



IAM y seguridad

G029

RESPUESTA

Policy = QUIÉN (member) con qué ROL sobre qué RECURSO.
 Members: usuarios, grupos, service accounts.
 Roles: básicos (Owner/Editor/Viewer, muy amplios), predefinidos (granulares: roles/pubsub.publisher), custom.
 Se hereda por la jerarquía org → carpeta → proyecto.
 Principio: least privilege.

BASICO

GCP

IAM y seguridad

G032

RESPUESTA

IAM autentica administradores y service accounts.
 Para usuarios FINALES de tu app: Identity Platform / Firebase Auth (email, Google, OAuth federado) emite JWTs que tu backend valida (firma + audience).
 API Gateway / Cloud Endpoints pueden validar JWTs antes de llegar a tu servicio.

INTERMEDIO

GCP



IAM y seguridad

G031

RESPUESTA

Bóveda central de secretos (API keys, passwords, tokens) con versionado, IAM por secreto, rotación, auditoría y replicación.
 La app lee secretos en runtime con la client library usando ADC.
 Nada de secretos hardcodeados ni en variables de entorno planas.

INTERMEDIO

GCP

★

IAM y seguridad

G033

¿Qué es Application Default Credentials (ADC)?

INTERMEDIO

GCP

IAM y seguridad

G034

Custom roles: cuándo valen la pena

INTERMEDIO

GCP

IAM y seguridad

G035

Audit logs: qué registra GCP por ti

INTERMEDIO

GCP

IAM y seguridad

G036

Claves de Service Account: por qué evitarlas y qué hacer si son inevitables

INTERMEDIO

GCP

IAM y seguridad

G037

Cloud Data Loss Prevention (Sensitive Data Protection): para qué sirve

INTERMEDIO

GCP

IAM y seguridad

G038

Defensa en profundidad en GCP: las 6 capas

INTERMEDIO

GCP

IAM y seguridad

G039

Rotación de secretos sin downtime: patrón completo

INTERMEDIO

GCP

IAM y seguridad

G040

Anatomía de una IAM policy binding

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

IAM y seguridad

G034

RESPUESTA

Cuando un predefinido es muy amplio y necesitas solo un subconjunto: roles/custom.verOnlyRun = compute.instances.get + list...

Límite 300 custom por org; mantenerlos documentados y revisados.

Alternativa moderna: reusar predefinidos pequeños en vez de crear custom por equipo (gobernanza más simple).

INTERMEDIO

GCP



IAM y seguridad

G033

RESPUESTA

Mecanismo estándar de autenticación: las client libraries buscan credenciales en orden:

1. GOOGLE_APPLICATION_CREDENTIALS (archivo)
2. SA adjunta al entorno (metadata server en Cloud Run/GKE/Compute)
3. credenciales de gcloud en local.

Así el MISMO código corre en local y en GCP sin claves en el código.

INTERMEDIO

GCP

IAM y seguridad

G036

RESPUESTA

Riesgo: credencial larga que puede filtrarse (git, image, laptop), rotación manual, sin origen.

Alternativas: ADC/workload attached SA (GCP), Workload Identity Federation (externos), impersonation (CLI).

Si inevitable: org policy permite bloquear creación, Secret Manager para guardar, rotación <90 días, scope mínimo, detección de uso (SCC anomaly).

INTERMEDIO

GCP

IAM y seguridad

G035

RESPUESTA

Admin Activity (siempre, quién creó/modificó qué) + Data Access (lecturas de datos, opt-in por servicio) + System Event + Policy Denied.

Exportar a BigQuery para búsquedas tipo 'quién descargó el bucket X en marzo'.

Retención default 400 días (admin), configurable a más con bucket lock.

INTERMEDIO

GCP

IAM y seguridad

G038

RESPUESTA

- 1) Identidad: IAM mínimo, MFA, Federation.
- 2) Perímetro: VPC-SC, org policies, Armor.
- 3) Red: private IPs, firewall deny-by-default, NAT controlado.
- 4) Datos: cifrado (CMEK), DLP, retention.
- 5) Aplicación: secretos, image scanning, Binary Authorization.
- 6) Operación: SCC, audit logs, alertas.

Ninguna capa sola basta; el ataque se detiene en el anillo que no esperaba.

INTERMEDIO

GCP

IAM y seguridad

G037

RESPUESTA

Detectar y clasificar PII en texto/imágenes/tablas: emails, tarjetas, DNI, nombres.

De-identify: masking, tokenización, hashing (format-preserving).

Casos: escanear buckets/BigQuery en busca de datos sensibles expuestos, anonimizar logs antes de guardar, minimizar datos en analítica.

API por contenido o job scan programado.

INTERMEDIO

GCP

IAM y seguridad

G040

RESPUESTA

bindings: [{role: roles/run.invoker, members: [serviceAccount:sa@p.iam..., group:equipo@x.com, user:ana@x.com]}]

Aplica sobre un recurso (proyecto, bucket, servicio) y SE HEREDA hacia abajo.

Auditoría: getIamPolicy; modificar con etag (optimistic concurrency).

Deny policies (nuevo): reglas explícitas de denegación que SIEMPRE ganan sobre allow.

AVANZADO

GCP

IAM y seguridad

G039

RESPUESTA

1) Secret Manager guarda versión N; app cachea con TTL.

2) Rotación: crear versión N+1 (doble válido en BD si es password) → esperar TTL de cache → invalidar N.

3) Automatizar: Secret Manager rotation schedule + Cloud Function/Run que cambia la BD y publica evento.

Conexiones vivas viejas: drain con max lifetime de pool < período de rotación.

INTERMEDIO

GCP

IAM y seguridad

G041

Roles predefinidos más usados que debes reconocer

AVANZADO

GCP

IAM y seguridad

G042

IAM Conditions: permisos temporales o por recurso

AVANZADO

GCP

IAM y seguridad

G043

Workload Identity Federation: autenticar CI/CD SIN claves

AVANZADO

GCP

IAM y seguridad

G044

Service Account Hierarchy: user-managed vs default SAs

AVANZADO

GCP

IAM y seguridad

G045

actAs y impersonation: delegar sin dar tus permisos

AVANZADO

GCP

IAM y seguridad

G046

Cloud KMS y CMEK: claves de cifrado propias

AVANZADO

GCP

IAM y seguridad

G047

VPC Service Controls: el perímetro anti-exfiltración

AVANZADO

GCP

IAM y seguridad

G048

Cloud Armor: WAF y anti-DDoS en el edge

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

IAM y seguridad

G042

RESPUESTA

Condición CEL en el binding:
 resource.name.startsWith("projects/_/buckets/logs-prod"),
 request.time < timestamp("2025-12-31T00:00:00Z") (acceso temporal automático).
 Casos: acceso solo a un bucket, contractor con fecha de expiración, entornos por sufijo de nombre.
 Ojo: condiciones solo en algunos niveles (proyecto y recursos que soportan conditions).

AVANZADO

GCP

IAM y seguridad

G041

RESPUESTA

roles/viewer, roles/editor, roles/owner (básicos — evitar en prod).
 Por servicio: roles/run.invoker (invocar), roles/run.admin, roles/storage.objectViewer/objectAdmin, roles/cloudsql.client, roles/secretmanager.secretAccessor, roles/pubsub.publisher/subscriber, roles/logging.logWriter.
 Elije el mínimo que cumpla la tarea.

AVANZADO

GCP

IAM y seguridad

G044

RESPUESTA

Default SAs (compute@..., run-invoker): permisos amplios (editor) — riesgo.
 Práctica: una SA POR SERVICIO con permisos mínimos (api-sa@proj.iam), attach al deploy, y deshabilitar/restringir default SAs vía org policy.
 Nomenclatura: nombre corto + entorno (pedidos-prod@...).

AVANZADO

GCP

IAM y seguridad

G043

RESPUESTA

GitHub Actions/Azure/Okta asume una identidad externa → intercambio por token GCP temporal vía provider (pool + provider con mapping de claims).
 Cero JSON keys descargadas, sin rotación manual, audit trail.
 Config: gcloud iam workload-identity-pools create + add-azure/github provider + bind SA con roles/iam.workloadIdentityUser.

AVANZADO

GCP

IAM y seguridad

G046

RESPUESTA

Por default Google cifra at-rest con claves de Google (Google-managed).
 CMEK: TU keyring/clave en Cloud KMS protege Cloud SQL, GCS, BigQuery, Pub/Sub, disks... — se puede deshabilitar la clave y los datos quedan inaccesibles (kill switch).
 Rotación programada; HSM protection level para compliance alto.
 Costo por clave + versión.

AVANZADO

GCP

IAM y seguridad

G045

RESPUESTA

roles/iam.serviceAccountTokenCreator sobre una SA permite GENERAR tokens como ella (gcloud auth impersonate-credentials, SDK Option.withCredentials).
 Humanos con acceso a CLI operan 'como' la SA sin conocer sus claves.
 Todo queda auditado como el token impersonado + el actor.

AVANZADO

GCP

IAM y seguridad

G048

RESPUESTA

Security policies adjuntas al backend service del LB: reglas preconfiguradas (XSS, SQLi, LFI), rate-based bans (N req/min por IP), geo-restricciones, allow/deny/redirect.
 Adaptive Protection (ML) contra ataques volumétricos L3/L4 en Cloud Armor Enterprise.
 Reglas en yaml/terraform versionadas.

AVANZADO

GCP

IAM y seguridad

G047

RESPUESTA

Perímetro lógico alrededor de servicios (GCS, BigQuery, Secret Manager...): solo tráfico desde proyectos/redes del perímetro.
 Bloquea copiar data a un bucket externo aunque IAM lo permita.
 Dry-run primero (audit) para no romper, luego enforce.
 Concepto clave para compliance de datos sensibles.

AVANZADO

GCP IAM y seguridad G049 Identity-Aware Proxy (IAP): acceso administrativo sin VPN AVANZADO	GCP IAM y seguridad G050 Security Command Center: el dashboard de seguridad AVANZADO
GCP IAM y seguridad G051 Binary Authorization: solo imágenes firmadas se despliegan AVANZADO	GCP IAM y seguridad G052 Cloud Armor rate limiting y bot management AVANZADO
GCP★ Redes G053 VPC, subredes y firewall rules INTERMEDIO	GCP★ Redes G054 Cloud Load Balancing y Cloud CDN INTERMEDIO
GCP★ Redes G055 ¿Cómo conecto Cloud Run/GKE con recursos privados? INTERMEDIO	GCP★ Redes G056 VPC global vs subredes regionales: entendiendo el modelo GCP AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

IAM y seguridad

G050

RESPUESTA

SCC: inventario y hallazgos (findings) de vulnerabilidades (VM vulnerable, bucket público, SA con clave vieja), threat detection (anomalías), posture management (compliance CIS). Integración con Pub/Sub para automatizar remediación. Standard vs Premium/Enterprise (más detectores y SIA).

AVANZADO

GCP

IAM y seguridad

G049

RESPUESTA

IAP delante de apps internas/admin: el usuario pasa por Google login y el LB inyecta el header X-Goog-Authenticated-User — solo members con roles/iap.httpsResourceAccessor pasan. Para SSH/RDP a VMs: IAP Desktop Tunneling sin IP externa ni bastión. Zero-trust interno: identidad por request.

AVANZADO

GCP

IAM y seguridad

G052

RESPUESTA

Reglas rate-based: throttle por IP (N req/min), ban temporal al exceder; por cookie/header con Expression language. Bot management (Enterprise): reCAPTCHA Enterprise integrado — score por request, desafíos. Geo + ASN blocking para mercados definidos. Métricas de la policy → detectar ataques en curso antes del SLO burn.

AVANZADO

GCP

IAM y seguridad

G051

RESPUESTA

Policy que exige attestations (firma del pipeline de CI) antes de permitir deploy en GKE/Cloud Run. Chainguard/Sigstore cosign para firmar; provenance de Cloud Build como attestor. Bloquea supply chain attacks (imagen 'cualquiera' en producción). Dry-run mode para calibrar sin romper.

AVANZADO

GCP ★

Redes

G054

RESPUESTA

Balanceador global anycast HTTP(S) L7 (o L4): terminación SSL, URL maps que enrutan por path/host a backend services (VMs, serverless NEG, Storage). Cloud CDN cachea en edge POPs cerca del usuario. Cloud Armor añade WAF y protección anti-DDoS.

INTERMEDIO

GCP ★

Redes

G053

RESPUESTA

VPC: red privada GLOBAL; subredes REGIONALES con rangos CIDR. Firewall: reglas stateful allow/deny por prioridad, aplican a targets (tags de red o service accounts). Conectar VPCs: VPC Peering o Shared VPC. Private Google Access: llegar a las APIs de Google sin IP externa.

INTERMEDIO

GCP ★

Redes

G056

RESPUESTA

Una VPC es GLOBAL (abarca todas las regiones); las SUBREDES son regionales con CIDR propio. Dentro de la VPC, tráfico inter-región viaja por el backbone de Google (cifrado en tránsito). Comparado con AWS (VPC por región): menos piezas para conectar multi-región. Auto-mode VPC crea subredes default — para prod usa custom mode.

AVANZADO

GCP ★

Redes

G055

RESPUESTA

Serverless VPC Access connector (Cloud Run/Functions) para alcanzar la VPC privada: VMs, Memorystore, Cloud SQL con IP privada. GKE habla nativamente con la VPC. Cloud SQL Auth Proxy o conectores IAM para bases de datos. Private Service Connect: consumir servicios gestionados por IP privada.

INTERMEDIO

GCP

Redes

G057

Firewall rules: campos y orden de evaluación

AVANZADO

GCP

★

Redes

G058

Cloud NAT: por qué casi todo backend lo necesita

AVANZADO

GCP

Redes

G059

Shared VPC vs VPC Peering: cuándo cada uno

AVANZADO

GCP

Redes

G060

Cloud Load Balancing: mapa de tipos

AVANZADO

GCP

★

Redes

G061

Health checks del LB: por qué tu 200 vale menos de lo crees

AVANZADO

GCP

Redes

G062

Cloud DNS y dominios: gestión básica

AVANZADO

GCP

Redes

G063

Cloud Interconnect vs Cloud VPN: conectividad híbrida

AVANZADO

GCP

Redes

G064

Network tiers y egress: cómo se cobra la salida de datos

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP
RESPUESTA

Redes

G058

VMs/GKE con SOLO IP privada necesitan salir a internet (actualizar paquetes, llamar APIs externas): Cloud NAT provee egress NAT gestionado (IPs estáticas de salida).
Beneficios: sin IPs públicas (superficie de ataque), IP fija para allowlists del tercero.
Se asocia por router/subred; ports por VM ajustables para muchas conexiones.

AVANZADO

GCP
RESPUESTA

Redes

G057

Reglas stateful: direction (ingress/egress), source/dest (CIDR, tags, service accounts), protocolo/puerto, action allow/deny, PRIORITY (0-65535, MENOR gana), disabled.
Default: deny all ingress, allow all egress.
Targeting por tags de red o por service account (más preciso).
Logs de firewall activables por regla.

AVANZADO

GCP
RESPUESTA

Redes

G060

Global externo: HTTPS LB (L7, anycast, Cloud Armor/CDN), SSL Proxy/TCP Proxy (L4 no-HTTP), External passthrough Network LB (UDP/TCP directo).
Regional: internal HTTP(S), internal passthrough (entre servicios de la VPC).
Serverless NEG: LB delante de Cloud Run/Functions con dominio propio.
Componentes: forwarding rule → target proxy → URL map → backend service → NEG/MIG.

AVANZADO

GCP
RESPUESTA

Redes

G059

Shared VPC: UNA red central administrada por el equipo de plataforma (host project) donde los proyectos de servicio despliegan sin tocar red — control centralizado estricto.
Peering: conecta VPCs independientes 1:1 (transitividad NO incluida) — equipos autónomos.
Enterprise: Shared VPC para prod; peering para integraciones puntuales.

AVANZADO

GCP
RESPUESTA

Redes

G062

Zones públicas (tu dominio) y privadas (nombres internos svc.internal → IPs privadas).
DNS peering/forwarding para resolver hacia on-prem u otras VPCs.
DNSSEC activable; TTL bajo (60s) antes de migraciones.
Routing policies: geolocation/failover — DNS como primer nivel de DR.

AVANZADO

GCP
RESPUESTA

Redes

G061

El health check del backend service determina si la instancia recibe tráfico — debe chequear un endpoint REAL (ping a DB ligero ok, pero pesado = cascada de reinicios).
Firewall: permitir 35.191.0.0/16 y 130.211.0.0/22 (rangos del LB).
Peligro: health check que depende de una dependencia lenta → app sana marcada DOWN → outage por diseño.

AVANZADO

GCP
RESPUESTA

Redes

G064

Premium tier: sale por backbone de Google (óptimo global).
Standard tier: sale por ISP regional — más barato, latencia variable.
Egress es el costo invisible que crece: streaming de archivos/videos a usuarios lejanos = dinero.
CDN + caché reduce egress drásticamente; medir en Billing por SKU.

AVANZADO

GCP
RESPUESTA

Redes

G063

VPN (HA VPN): túneles IPsec sobre internet — días de setup, costo bajo, 99.99% con 2 interfaces; para inicio de híbrido.
Dedicated/Partner Interconnect: circuito físico privado (10/100Gbps o partner) — latencia estable, más caro, para workloads serios.
Router Cloud (BGP) intercambia rutas en ambos.
Patrón: empezar VPN, migrar a Interconnect al crecer.

AVANZADO

GCP

Redes

G065

Private Google Access y Private Service Connect: APIs sin internet

AVANZADO

GCP

Redes

G066

Serverless VPC Access connector: puente Cloud Run → VPC

AVANZADO

GCP

Redes

G067

Packet Mirroring y Flow Logs: depuración de red

AVANZADO

GCP

Redes

G068

Troubleshooting de conectividad GCP: el árbol de decisión

AVANZADO

GCP

Redes

G069

TLS interno entre servicios: mTLS con service mesh en GCP

AVANZADO

GCP

★

Compute y contenedores

G070

Compute Engine vs App Engine vs Cloud Run vs GKE

BASICO

GCP

★

Compute y contenedores

G071

¿Cuándo elegirías Cloud Run?

BASICO

GCP

Compute y contenedores

G072

Machine types: familias y cuál elegir

BASICO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Redes

G066

RESPUESTA

El connector (red por región, rango /28) permite que Cloud Run/Functions alcancen recursos de VPC privada: VMs, Memorystore, Cloud SQL con IP interna.
Costo por instancia del connector + throughput; dimensionar (2-10 instancias min/max según tráfico).
Direct VPC egress (nuevo, mejor): sin connector, salida directa por subred.

AVANZADO

GCP

Redes

G065

RESPUESTA

PGA: subred con ruta a 'private.googleapis.com' → las apps llaman APIs de Google (GCS, Secret Manager) por IP PRIVADA sin NAT ni internet.
PSC: endpoints privados para servicios específicos (incluye terceros en Marketplace y tu propio servicio detrás de un LB interno).
Estricto zero-internet: ambos + Cloud NAT solo si es indispensable.

AVANZADO

GCP

Redes

G068

RESPUESTA

- 1) ¿La VM/pod tiene IP esperada y rutas? (gcloud compute routes list)
- 2) Firewall: regla allow aplicando al target correcto (tags/SA) y source correcto.
- 3) Para servicios gestionados: Private Google Access / PSC / connector configurado?
- 4) DNS: zona privada resolviendo el FQDN interno?
- 5) Flow logs + firewall rules logging = evidencia del paquete denegado.

Connectivity Tests (Network Intelligence) simula el path y dice dónde se corta.

AVANZADO

GCP

Redes

G067

RESPUESTA

VPC Flow Logs: metadatos de conexiones (5-tuple, bytes, aceptado/denegado) exportables a BigQuery — responder '¿quién habla con quién y cuánto?'.
Firewall rules logging: qué regla bloqueó qué.
Packet Mirroring: copia completa de paquetes a un collector (IDS/IPS como Suricata) para análisis profundo.
Ante 'no conecta': flow logs primero, mirroring después.

AVANZADO

GCP ★

Compute y contenedores

G070

RESPUESTA

Compute Engine: VMs (IaaS), control total — lift & shift.
App Engine: PaaS clásico (standard/flexible).
Cloud Run: contenedor serverless, escala a 0, pagas por uso.
GKE: Kubernetes gestionado para orquestar muchos microservicios.
Serverless simple → Cloud Run; control de clúster → GKE.

BASICO

GCP

Redes

G069

RESPUESTA

Istio (Anthos Service Mesh) en GKE: sidecars/ambient manejan mTLS automático entre workloads con identidades SPIFFE — sin tocar código.
Autorización L7: AuthorizationPolicy (frontend puede llamar pedidos:GET, nada más).
Cloud Run/GKE sin mesh: TLS termina en LB; interno confía en red privada (menos fuerte).
Zero trust interno: mesh es la respuesta estándar.

AVANZADO

GCP

Compute y contenedores

G072

RESPUESTA

E2: costo mínimo (no garantiza hardware fijo) — dev, web ligera.
N2/N2D: propósito general balanceado (la default de prod).
C2/C3: compute optimizado (CPU puro). M2/M3: memoria enorme. A2/A3: GPU (ML).
Custom shapes: vCPU/RAM a la carta en N2.
Regla: empezar N2D estándar, medir, ajustar; Recommender sugiere.

BASICO

GCP ★

Compute y contenedores

G071

RESPUESTA

Cuando empaquetas la app en un contenedor Docker sin gestionar infraestructura: APIs HTTP, webhooks, microservicios, eventos.
Escala automático a 0 (no pagas en reposo) hasta N instancias; concurrencia por instancia configurable.
Debe ser stateless y escuchar en la variable \$PORT.

BASICO

GCP

Compute y contenedores

G073

Disks en Compute Engine: tipos y cuándo

INTERMEDIO

GCP

Compute y contenedores

G074

Managed Instance Groups: autoscaling y autohealing

INTERMEDIO

GCP

Compute y contenedores

G075

Instance template + startup script: boot reproducible

INTERMEDIO

GCP

Compute y contenedores

G076

Spot VMs: patrón de uso correcto

INTERMEDIO

GCP

Compute y contenedores

G077

OS Login y metadata server: acceso y datos de instancia

INTERMEDIO

GCP

★

Compute y contenedores

G078

GKE Autopilot vs Standard: decisión práctica

INTERMEDIO

GCP

Compute y contenedores

G079

Node pools y taints/tolerations: aislar cargas

INTERMEDIO

GCP

★

Compute y contenedores

G080

Workload Identity en GKE: pods con SA de Google sin claves

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Compute y contenedores

G074

RESPUESTA

MIG: grupo de VMs idénticas desde instance template.
 Autoscaling por CPU/utilización o métrica de LB; autohealing con health check (recrea VMs malas); rolling updates (surge/unavailable) y stateful MIGs.
 Zonal (una zona) vs Regional (3 zonas — HA por diseño).
 Es 'Kubernetes sin Kubernetes' para apps simples.

INTERMEDIO

GCP

Compute y contenedores

G073

RESPUESTA

Standard PD: barato, secuencial (backups). Balanced PD: default razonable. SSD PD: IOPS/throughput alto (DBs). Hyperdisk: máximo performance desacoplada del tamaño.
 Snapshots: incrementales multi-region — backups programados (snapshot schedule).
 Regional PD: replicación sincrónica entre zonas para BDs auto-alojadas HA.

INTERMEDIO

GCP

Compute y contenedores

G076

RESPUESTA

Hasta 91% de descuento; Google puede EVICTAR (preemption notice de 30s).
 Ideal: workers de colas (Pub/Sub redeliver), batch, CI runners, render — todo con checkpoint/retry.
 NO para: API crítica, BDs, estado sin replicar.
 Combina con MIG (mezcla estándar + spot) para absorber picos baratos.

INTERMEDIO

GCP

Compute y contenedores

G075

RESPUESTA

Template: imagen, machine type, network, SA, metadata — immutable; cambios = template nuevo + MIG rolling.
 Startup script (metadata startup-script): bash al boot — instalar paquetes, tocar configs.
 Mejor: imagen dorada pre-baked (Packer/Cloud Build) + startup script mínimo = arranque rápido y determinista.

INTERMEDIO

GCP



Compute y contenedores

G078

RESPUESTA

Autopilot: Google gestiona nodos (pagas por pods) — menos ops, seguridad por default (no root, hardened), SLA de pods; costo ligeramente mayor y menos control (taints, kernel).
 Standard: control total de node pools, DaemonSets, costos más bajos a gran escala — requiere equipo k8s.
 Default moderno: Autopilot salvo necesidad específica.

INTERMEDIO

GCP

Compute y contenedores

G077

RESPUESTA

OS Login: SSH por permisos IAM (roles/compute.osLogin) — sin gestionar keys por VM, 2FA heredado, audit por usuario.
 Metadata server (169.254.169.254): la instancia lee su SA token, zona, custom metadata — así funcionan ADC sin claves.
 Bloquear metadata con headers? No aplica (Google usa 'Metadata-Flavor').

INTERMEDIO

GCP



Compute y contenedores

G080

RESPUESTA

- 1) KA SA de Google + binding roles/iam.workloadIdentityUser al K8s SA.
- 2) K8s SA anotada con iam.gke.io/gcp-service-account.
- 3) Pod usa el K8s SA → metadata server entrega tokens de la Google SA.

Reemplaza key.json montados — el estándar de acceso a GCP desde GKE.

INTERMEDIO

GCP

Compute y contenedores

G079

RESPUESTA

Node pool por tipo de carga: pool GPU (taint nvidia.com/gpu), pool spot, pool con máquina de memoria.
 Taint en el pool + toleration en el pod = solo ciertos pods aterrizan ahí.
 nodeSelector/affinity para ubicación (zona, tipo).
 Auto-upgrade por release channel (regular/stable) con ventanas de mantenimiento.

INTERMEDIO

GCP

Compute y contenedores

G081

HPA y VPA: autoscaling de pods y nodos

INTERMEDIO

GCP

Compute y contenedores

G082

Container Registry → Artifact Registry: el switch obligatorio

INTERMEDIO

GCP

Compute y contenedores

G083

GKE networking: VPC-native (alias IPs) y por qué importa

INTERMEDIO

GCP

Compute y contenedores

G084

Release channels y upgrades de GKE sin sustos

INTERMEDIO

GCP

Compute y contenedores

G085

Cloud SQL como servicio en GKE: proxy sidecar vs connector

INTERMEDIO

GCP

Compute y contenedores

G086

Ingress vs Gateway API vs Service de tipo LoadBalancer en GKE

AVANZADO

GCP

Compute y contenedores

G087

Network Policy en GKE: microsegmentación

AVANZADO

GCP

Compute y contenedores

G088

Dataplane V2 y Network Policies: rendimiento + seguridad

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Compute y contenedores

G082

RESPUESTA

Container Registry (gcr.io) deprecado; Artifact Registry (REGION-docker.pkg.dev/PROJECT/REPO/IMAGE).
 Repos tipados: docker, npm, maven, python, apt... por región (más cerca = pull más rápido y sin egress).
 Remote repos (proxy a Docker Hub) y vulnerabilidad scanning integrado.
 gcloud auth configure-docker region-docker.pkg.dev.

INTERMEDIO

GCP

Compute y contenedores

G081

RESPUESTA

HPA: replica PODS por CPU/mem o métricas custom (QPS de Prometheus) — la primera línea.
 VPA: recomienda/ajusta requests-limits por pod (right-sizing; modo recomendación en prod).
 Cluster Autoscaler: añade/elimina NODOS cuando no caben pods o sobran.
 KEDA para scale por longitud de cola (event-driven).

INTERMEDIO

GCP

Compute y contenedores

G084

RESPUESTA

Channels: rapid/regular/stable — Google parchea nodos y control plane automáticamente.
 Estrategia: clúster de stage en rapid para ver el futuro; prod en regular.
 Maintenance windows y exclusions para no parchear en picos.
 PodDisruptionBudgets + surge upgrades: la app sobrevive al drenaje de nodos.

INTERMEDIO

GCP

Compute y contenedores

G083

RESPUESTA

VPC-native: pods reciben IPs DE LA SUBRED (secondary range) → visibles/ruteables en la VPC, firewall por pod, mejor para service mesh y politiques.
 Routes-based (legado): NAT de IP de nodo — rompe firewalls por pod.
 Requisito para Shared VPC, Cloud NAT por pod, Network Policies con Dataplane V2.
 Pools nuevos ya son VPC-native por default.

INTERMEDIO

GCP

Compute y contenedores

G086

RESPUESTA

Service LoadBalancer: IP externa por servicio (caro, uno por svc).
 Ingress (GCE o nginx controller): rutas HTTP/HTTPS por host/path a Services — el clásico.
 Gateway API (el sucesor): roles separados (Gateway vs HTTPRoute), portable entre vendors, traffic split nativo.
 Nuevo diseño: Gateway API; Ingress sigue soportado.

AVANZADO

GCP

Compute y contenedores

G085

RESPUESTA

Sidecar del Auth Proxy (contenedor junto a tu app): conexiones locales por socket unix, IAM auth, cifrado — el patrón estándar.
 En Autopilot: workload identity + proxy con recursos asignados.
 Direct private IP + gcp cloud-sql-proxy como DaemonSet para muchas apps.
 Pool sizing: cada instancia de app abre sus conexiones — cuenta el total vs max_connections de la BD.

INTERMEDIO

GCP

Compute y contenedores

G088

RESPUESTA

Dataplane V2 (eBPF, Cilium): NetworkPolicies más eficientes, logging/observabilidad de flujos (kube API, flow export) — habilitarlo en la creación del cluster.
 Sin DPv2: Calico alternativo.
 Policy audit mode: loguea sin bloquear para calibrar antes de enforce.
 Reglas ejemplo: frontend→backend:8080 only, backend→db:5432 only.

AVANZADO

GCP

Compute y contenedores

G087

RESPUESTA

K8s NetworkPolicy (Calico/Dataplane V2): default deny entre namespaces y permitir solo lo necesario (frontend→backend:8080, backend→db:5432).
 Sin políticas, cualquier pod habla con todo (lateral movement fácil si uno se compromete).
 Aplicar por fases: audit con logging primero, luego deny.

AVANZADO

GCP

Compute y contenedores

G089

StatefulSet, PVs y discos en GKE

AVANZADO

GCP

Compute y contenedores

G090

Node auto-provisioning y bin packing:
optimizar costo de clúster

AVANZADO

GCP★

Serverless

G091

Cloud Functions vs Cloud Run

BASICO

GCP★

Serverless

G092

Cloud Run: revisiones y traffic split (canary fácil)

BASICO

GCP★

Serverless

G093

Cloud Run: concurrencia y CPU allocation

BASICO

GCP★

Serverless

G094

Min instances y cold starts en Cloud Run

BASICO

GCP

Serverless

G095

Cloud Run jobs y sidecars: capacidades menos conocidas

BASICO

GCP

Serverless

G096

Cloud Functions 2ª gen: qué hereda de Cloud Run

BASICO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Compute y contenedores

G090

RESPUESTA

NAP: crea node pools automáticamente según los pods pendientes (por tipo de máquina/GPU) — no precisas pools. Combinar con: cluster autoscaler scale-down (vaciar nodos), pod anti-affinity razonable (no desperdiciar nodos), requests REALES de recursos (over-request = dinero tirado). Ver nodos zombis: `kubectl get nodes + utilization dashboard`.

AVANZADO

GCP

Compute y contenedores

G089

RESPUESTA

StatefulSet: identidad estable (nombre ordinal), PVCs por réplica (volumeClaimTemplates) — para BDs autoalojadas. PD CSI driver: persistent disks/zonal, Filestore para NFS, Cloud SQL proxy en vez de correr DB en k8s (recomendado). Backup: Velero a GCS. Regla de oro: en GKE el state EXTERNO (managed DBs), el clúster Stateless.

AVANZADO

GCP



Serverless

G092

RESPUESTA

Cada deploy crea una REVISIÓN inmutable (imagen + env + recursos).
`gcloud run services update-traffic svc --to-revisions svc-00002=10,svc-00001=90` — canary sin Kubernetes.
 Tag una revisión (--tag blue) para URL de preview antes de servir tráfico.
 Rollback = redirigir 100% a la revisión anterior (instantáneo).

BASICO

GCP



Serverless

G091

RESPUESTA

Cloud Functions: FaaS — función suelta disparada por evento (HTTP, Pub/Sub, Storage); la 2ª generación corre sobre Cloud Run.
 Cloud Run: contenedor completo (cualquier runtime/framework), más control, alta concurrencia por instancia.
 Regla: una función pequeña → Functions; una app/servicio → Run.

BASICO

GCP



Serverless

G094

RESPUESTA

Frío: nueva instancia = pull imagen + arranque JVM (1-3s+).
 Min instances = N instancias siempre calientes (costo fijo pero latencia estable) — 1-2 para APIs críticas.
 Reducir frío: imagen distroless pequeña, JVM con CDS/AppCDS, arranque lazy de beans, startup CPU boost activado.

BASICO

GCP



Serverless

G093

RESPUESTA

Concurrency (default 80, max 1000): requests simultáneos por instancia — Node/async lo usa bien; Java/thread-per-request bajo (4-16) con más instancias.
 CPU allocation: solo-durante-request (barato, latencias frías al volver) vs always-allocated (background threads, gRPC servers, manteniendo cachés calientes).
 Ajuste = costo vs latencia.

BASICO

GCP

Serverless

G096

RESPUESTA

Gen2 ES Cloud Run por debajo: cualquier runtime con Functions Framework, timeouts hasta 60 min, más memoria/CPU, concurrency, Eventarc triggers nativos (60+ fuentes de eventos), tráfico por revisiones.
 Gen1: legado (timeouts 9 min, triggers limitados).
 Nuevo código: siempre gen2 o Cloud Run directo.

BASICO

GCP

Serverless

G095

RESPUESTA

Jobs: tareas batch one-shot o programadas (migraciones, ETL) con paralelismo por task index — no atado a HTTP.
 Sidecars (gen2): contenedores auxiliares en la misma instancia (proxy de service mesh, collector de logs) con lifecycle compartido.
 Ambos mantienen el modelo serverless sin Kubernetes.

BASICO

GCP

Serverless

G097

Eventarc: el bus de eventos estándar de GCP

BASICO

GCP

Serverless

G098

App Engine en 2024+: ¿todavía tiene sentido?

BASICO

GCP★

Serverless

G099

Elegir ambiente de ejecución: matriz de decisión rápida

BASICO

GCP

Serverless

G100

Cloud Run y websockets/SSE: limitaciones reales

BASICO

GCP

Serverless

G101

Cold start de JVM en Cloud Run: checklist completo

BASICO

GCP★

Almacenamiento

G102

Cloud Storage: clases y lifecycle

BASICO

GCP

Almacenamiento

G103

GCS: estructura (buckets/objetos) y límites clave

BASICO

GCP★

Almacenamiento

G104

Signed URLs: dar acceso temporal sin credenciales

BASICO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Serverless

G098

RESPUESTA

Standard: runtimes gestionados muy simples, scale to zero histórico, pero restricciones de runtime.
 Flexible: VMs gestionadas (menos serverless de verdad).
 Nuevos proyectos: Cloud Run cubre el 95% (contenedor portable, concurrencia, revisiones).
 App Engine sigue operado para apps existentes; no es la recomendación de partida hoy.

BASICO

GCP

Serverless

G097

RESPUESTA

Rutea eventos (Cloud Audit Logs, Pub/Sub, GCS, Firebase, custom) a Cloud Run/Functions/GKE con filtros por tipo/atributos.
 Ejemplo: 'cuando un objeto se suba a gs://facturas con prefijo xe/', invoca Cloud Run procesar.
 Reemplaza triggers ad-hoc de gen1; entrega at-least-once con retry configurable.

BASICO

GCP

Serverless

G100

RESPUESTA

SSE/streaming HTTP: soportado bien (response streaming) — dashboards push simples.
 WebSockets: soportados (gen2) PERO con concurrencia por instancia: cada instancia mantiene sus conexiones; scaling por conexiones requiere métricas custom.
 Always-allocated CPU obligatorio para mantener conexiones vivas entre requests.
 Timeouts: 60 min máx por request; reconexión del cliente debe estar diseñada.

BASICO

GCP



Serverless

G099

RESPUESTA

Contenedor HTTP/API → Cloud Run.
 Función pequeña evento-driven → Cloud Functions gen2.
 Tarea batch → Cloud Run Jobs (o Batch service para HPC).
 Muchos microservicios con malla/ecosistema k8s → GKE.
 VM con software legado/licencias → Compute Engine.
 Sin servidores y con BD gestionada → App Engine solo si ya vives ahí.

BASICO

GCP



Almacenamiento

G102

RESPUESTA

Standard: acceso frecuente.
 Nearline: ~1 acceso/mes. Coldline: ~1 cada 90 días. Archive: retención larga (lo más barato).
 Multi-region para HA; region para menor costo.
 Lifecycle rules automáticas: degradar clase, borrar versiones viejas. Versioning y retención.

BASICO

GCP

Serverless

G101

RESPUESTA

- 1) Startup CPU boost ON (2 vCPU al arrancar).
- 2) Imagen distroless + JVM headless; AppCDS/CDS training para compartir clases.
- 3) Spring: lazy initialization selectivo, excluir autoconfigs de arranque, iniciar listeners tras ready (readiness).
- 4) Min instances = 1 para rutas críticas.
- 5) Medir: container startup time en métricas de Run — objetivo <2s con boost.

BASICO

GCP



Almacenamiento

G104

RESPUESTA

URL firmada con expiración (V4): permite al cliente SUBIR o DESCARGAR directo del bucket con tu firma — tu app solo genera la URL.
 Casos: uploads de archivos grandes sin atravesar tu API, links de descarga seguros temporales.
 Signed POST policies para restringir tamaño/CONTENT-type en upload.

BASICO

GCP

Almacenamiento

G103

RESPUESTA

Namespace global de buckets; dentro, objetos planos con nombre (no hay 'carpetas' reales — son prefijos).
 Objetos inmutables por versión; hasta 5TB por objeto; metadata + ACLs/IAM.
 Ops por HTTP/XML/JSON API, gsutil o client libs con resumable upload para grandes.
 Diseño de nombres: prefijos tipo fecha (logs/2025/09/12/) para listar/particionar.

BASICO

GCP

★

Almacenamiento

G105

Lifecycle rules: automatizar costo y retención

BASICO

GCP

Almacenamiento

G106

Versioning, retention y bucket lock: inmutabilidad

BASICO

GCP

Almacenamiento

G107

Public objects y uniform access: seguridad de buckets

BASICO

GCP

Almacenamiento

G108

Storage Transfer Service: migraciones y sincronización

BASICO

GCP

Almacenamiento

G109

Sitio estático + CDN en GCS: la arquitectura más barata

BASICO

GCP

Almacenamiento

G110

Diseño de buckets multi-entorno sin filtros de datos cruzados

BASICO

GCP

Bases de datos

G111

¿Para qué sirve Memorystore?

BASICO

GCP

★

Bases de datos

G112

Cloud SQL: HA, réplicas y backups de verdad

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP
RESPUESTA

Almacenamiento

G106

Versioning: sobrescribir crea versión antigua accesible (restore/soft-delete accidental).
Retention policy: objetos no borrables antes de N días (compliance). Bucket Lock con retención LOCKED = WORM inmutable (auditoría legal).
Soft delete (nuevo): ventana de recuperación de objetos borrados por defecto (7 días).

BASICO

GCP ★
RESPUESTA

Almacenamiento

G105

Reglas por prefijo/edad/tiempo desde última modificación: cambiar a Nearline (30d), Coldline (90d), Archive (365d), borrar versiones viejas, eliminar objetos temporales.
Ejemplo: logs → 90 días Standard, luego Archive, borrar a 3 años (compliance).
Se aplican de fondo; verificar con lifecycle log/dry runs.

BASICO

GCP
RESPUESTA

Almacenamiento

G108

Transfiere: de S3/Azure/on-prem a GCS, entre buckets, o respaldo recurrente con filtros (prefijo, fecha) y schedules.
Verificación de integridad (checksums), bandwidth limit, delete objects que no estén en el origen (sync).
Transfer Appliance: hardware físico para TB/PB sin red suficiente.

BASICO

GCP
RESPUESTA

Almacenamiento

G107

Uniform bucket-level access (obligatorio en nuevos): SOLO IAM, sin ACLs por objeto — gobernanza simple.
allUsers/allAuthenticatedDomains = público: detectar con SCC y bloquear con org policy (publicAccessPrevention).
Compartir: signed URLs o IAM por grupo, nunca ACLs ad-hoc.

BASICO

GCP
RESPUESTA

Almacenamiento

G110

Separación por proyecto (dev/stage/prod) — cada entorno su bucket, su IAM.
Dentro: prefijos por dominio y fecha (invoices/2025/09/). Evitar bucket único 'todo' con ACLs finas.
Permisos por prefijo solo si IAM uniforme está OFF (anti-patrón) — mejor buckets separados.
Nombre del bucket = identidad global: usar sufijo random/hashed para evitar colisiones.

BASICO

GCP
RESPUESTA

Almacenamiento

G109

Bucket con index.html + LB backend bucket + Cloud CDN → sitio global HTTPS con cert gestionado.
Cache-Control por tipo (html no-cache, assets 1 año immutable con hash en nombre).
Alternativa serverless: Firebase Hosting (deploy en 1 comando).
Costo: centavos para tráfico moderado.

BASICO

GCP ★
RESPUESTA

Bases de datos

G112

HA: regional con standby sincrónico en otra zona — failover automático (~60s, misma IP via proxy).
Read replicas: lecturas escaladas / analítica ligera; cross-region replica para DR (promotable).
Backups automáticos + PITR con write-ahead logs (restaurar a un minuto concreto).
Maintenance window configurable (patches).

INTERMEDIO

GCP
RESPUESTA

Bases de datos

G111

Redis / Memcached gestionado en memoria:

- caché de resultados y sesiones
- rate limiting
- colas ligeras

Reduce latencia (sub-milisegundos) y carga sobre Cloud SQL/Firestore.
Tiers: Basic (sin HA) vs Standard (con réplica). Configurar eviction (allkeys-lru) y TTLs.

BASICO

GCP

★

Bases de datos

G113

Conectarte a Cloud SQL: proxy vs IP privada

INTERMEDIO

GCP

Bases de datos

G114

Cloud SQL vs AlloyDB: cuando Postgres crece

INTERMEDIO

GCP

Bases de datos

G115

Firestore: modelo de datos y reglas

INTERMEDIO

GCP

Bases de datos

G116

Firestore: límites que definen si te sirve

INTERMEDIO

GCP

★

Bases de datos

G117

Matriz de decisión de bases de datos GCP (entrevista clásica)

INTERMEDIO

GCP

Bases de datos

G118

Memorystore para Redis: HA, tamaño y patrones de uso

INTERMEDIO

GCP

★

Bases de datos

G119

Caching pattern: cache-aside implementado bien

INTERMEDIO

GCP

Bases de datos

G120

Cloud SQL performance: los 6 knobs que mueven la aguja

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Bases de datos

G114

RESPUESTA

Cloud SQL: Postgres/MySQL gestionado clásico — HA, réplicas, precio contenido.
 AlloyDB: Postgres compatible pero con capa analítica (columnar engine 100x para queries analíticas), ~2x más rápido OLTP, escáner de caché ultra rápido, escala de réplicas de lectura sin lag.
 Candidato cuando Cloud SQL se queda corto en performance y el presupuesto existe.

INTERMEDIO

GCP



Bases de datos

G113

RESPUESTA

Cloud SQL Auth Proxy: sidecar/proceso local que cifra y autentica por IAM (sin SSL certs manuales ni IP pública) — recomendado para Cloud Run/develop.
 Private IP: la BD solo en la VPC (Serverless VPC connector / GKE nativo) — topología segura de prod.
 Jamás IP pública + password solo (combinable pero mínima).

INTERMEDIO

GCP

Bases de datos

G116

RESPUESTA

1MB/doc, 500 writes/seg por documento (hotspot), sin queries OR amplios ni 'NOT IN' complejos, consistencia eventual entre documentos (tx solo sobre doc/grupo).
 Diseño: denormalizar, arrays acotados (array de 10k = antipatrón), distribuir writes con sharded counters.
 Analítica pesada → exportar a BigQuery.

INTERMEDIO

GCP

Bases de datos

G115

RESPUESTA

Documentos JSON (~1MB) en colecciones; subcolecciones anidadas; documentos con IDs.
 Consultas con índices (compuestos automáticos/manuales); SIN joins ni agregados complejos (count/sum básicos).
 Realtime listeners y offline sync (móvil/web) — su superpoder.
 Security Rules: permisos por documento evaluados en el cliente-borde (para apps directas).

INTERMEDIO

GCP

Bases de datos

G118

RESPUESTA

Tiers: Basic (sin réplica, barato) vs Standard (réplica + failover); cluster mode para >5GB o sharding.
 Uso: caché de resultados, sesiones, rate limiting (INCR), leaderboards (sorted sets), locks con SET NX PX.
 Eviction policy: allkeys-lru típico de caché; volatile-ttl si mezclas.
 Persistencia (RDB) opcional — Redis como caché no la necesita.

INTERMEDIO

GCP



Bases de datos

G117

RESPUESTA

Relacional simple gestionado → Cloud SQL. Postgres a escala/analítica → AlloyDB. Relacional global ultra-consistente → Spanner. Documentos + realtime/offline móvil → Firestore. Series temporales/IoT masivo → Bigtable. Caché/sesiones → Memorystore. Analítica OLAP → BigQuery. Blobs/archivos → Cloud Storage.
 Pregunta clave: patrón de acceso + consistencia + escala + presupuesto.

INTERMEDIO

GCP

Bases de datos

G120

RESPUESTA

- 1) Tamaño de instancia correcto (CPU/RAM p95).
- 2) Índices con EXPLAIN ANALYZE (missing index = sec scan).
- 3) Connection pooling: Hikari + max_connections coherentes; PgBouncer si apps múltiples.
- 4) flags: innodb_buffer_pool_size / shared_buffers ~70% RAM.
- 5) read replica para analítica.
- 6) SSD PD y maintenance window fuera de picos.

INTERMEDIO

GCP



Bases de datos

G119

RESPUESTA

Lectura: 1) buscas en Redis, 2) miss → BD, 3) guardas en Redis con TTL + jitter.
 Escritura: actualiza BD → invalida/actualiza caché (delete, no write-through, para evitar race).
 Claves con namespaces: prod:user:42.
 Serialize DTOs compactos (JSON/protobuf); nunca entidades JPA con lazy.

INTERMEDIO

GCP

Bases de datos

G121

Secrets y datos sensibles en BDs: cifrado y acceso

INTERMEDIO

GCP

Bases de datos

G122

Cloud SQL flags y mantenimiento: los detalles que evitan sustos

INTERMEDIO

GCP

★

Bases de datos

G123

¿Qué base de datos GCP elijo? (SQL vs Firestore vs Spanner vs Bigtable vs BigQuery)

AVANZADO

GCP

Bases de datos

G124

Spanner: qué lo hace único (TrueTime y global)

AVANZADO

GCP

Bases de datos

G125

Bigtable: diseño de row key que todo el mundo hace mal

AVANZADO

GCP

Bases de datos

G126

Database Migration Service: migrar con mínimo downtime

AVANZADO

GCP

Bases de datos

G127

Cross-region disaster recovery para Cloud SQL: pasos

AVANZADO

GCP

Bases de datos

G128

Vector databases en GCP: opciones actuales

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Bases de datos

G122

RESPUESTA

Flags (database flags): `slow_query_log=ON`, `max_connections`, `timezone`, `character_set`.

Maintenance window (día/hora) + maintenance timing (earlier/later) — Google patchea ahí; HA hace failover real (prueba el rebalanceo).

Rejections en producción: siempre el warning de 'patching imminent' por email/monitoring.

Instance restart por flag = planificar ventana.

INTERMEDIO

GCP

Bases de datos

G121

RESPUESTA

At-rest: cifrado por defecto; CMEK si compliance exige.

En tránsito: SSL obligatorio (force SSL flag) + IAM-auth (Cloud SQL IAM authentication: usuarios = identidades Google).

Passwords de app: Secret Manager, nunca en properties.

Auditoría: Cloud SQL audit logs para SELECT/UPDATE de tablas sensibles (PII).

INTERMEDIO

GCP

Bases de datos

G124

RESPUESTA

SQL relacional DISTRIBUIDO con consistencia EXTERNA (strong) global — TrueTime (relojes atómicos GPS) permite transacciones serializables a escala planetaria.

Escalas horizontalmente sin sharding manual (interleave/tablas hijas para localidad).

Costo alto (nodos + storage): para fin-tech, inventario global, SaaS multi-tenant serio.

99.999% multi-región.

AVANZADO

GCP



Bases de datos

G123

RESPUESTA

Cloud SQL: relacional gestionado (MySQL/Postgres/SQL Server) — apps transaccionales.

Firestore: NoSQL documentos, tiempo real (móvil/web).

Spanner: SQL distribuido GLOBAL con consistencia fuerte.

Bigtable: NoSQL wide-column, time-series/IoT, throughput masivo.

BigQuery: analítica OLAP sobre petabytes (serverless).

AVANZADO

GCP

Bases de datos

G126

RESPUESTA

DMS: réplica continua desde MySQL/Postgres/SQL Server (on-prem u otro cloud) a Cloud SQL/AlloyDB con CDC (binlog/WAL).

Cutover: cortar escrituras origen → esperar lag 0 → promover destino → redirigir app.

Homogeneous principalmente; heterogéneo vía tools adicionales (PgBouncer/assessment).

Prueba: run pre-migration assessment.

AVANZADO

GCP

Bases de datos

G125

RESPUESTA

Rendimiento = distribución de filas por row key. MAL: timestamp puro al inicio (todo cae en un nodo — hotspot). BIEN: prefijo reversado/distribuido + timestamp reversado para 'lo más nuevo primero': `user#reversed_ts`.

Wide-column: miles de versiones por celda, GC policies por edad/versión.

Casos: series temporales IoT, telemetría, fin personalizado de HBase.

AVANZADO

GCP

Bases de datos

G128

RESPUESTA

1) Vertex AI Vector Search (Managed Index, ANN a escala — el más potente).

2) AlloyDB con extensiones pgvector + fast path — Postgres friendly.

3) Cloud SQL pgvector para escala moderada.

4) Firestore vector search para apps móviles.

Patrón RAG: documentos → embeddings (Vertex) → índice → query embedding → top-k → LLM con contexto.

AVANZADO

GCP

Bases de datos

G127

RESPUESTA

1) Read replica cross-region continua (asincrónica) en la región DR.

2) Backups también en la región secundaria (backup replication).

3) Drill: promover la réplica (promote) y apuntar la app vía connection name por región/flag.

4) RTO estimado = promoción (~min) + redirect; RPO = lag de réplica (segundos-min).

Automatizar el failover con Workflows + alerta de región caída.

AVANZADO

GCP

★

Mensajería y eventos

G129

Pub/Sub: modelo y garantías de entrega

INTERMEDIO

GCP

Mensajería y eventos

G130

Pub/Sub vs Kafka vs RabbitMQ: mapa conceptual

INTERMEDIO

GCP

★

Mensajería y eventos

G131

Push vs Pull en Pub/Sub: cuándo cada uno

INTERMEDIO

GCP

Mensajería y eventos

G132

Pub/Sub schemas y filtrado de subscriptions

INTERMEDIO

GCP

★

Mensajería y eventos

G133

Cloud Tasks vs Pub/Sub: la pregunta que confunde a todos

INTERMEDIO

GCP

Mensajería y eventos

G134

Event-driven en GCP: la arquitectura canónica

INTERMEDIO

GCP

Mensajería y eventos

G135

Cloud Scheduler + Workflows: orquestación sin servidores

INTERMEDIO

GCP

Mensajería y eventos

G136

Idempotencia de consumidores: implementación concreta

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Mensajería y eventos

G130

RESPUESTA

Pub/Sub: managed global, push/pull, at-least-once, sin claves de partición complejas — el default GCP.
 Kafka: log distribuido con particiones y consumer groups, replay nativo, exactly-once con transacciones — streaming pesado y ecosistema (self-managed o Dataproc/Confluent).
 RabbitMQ: colas inteligentes con routing (exchanges) — patrones de cola enterprise clásicos.

INTERMEDIO

GCP



Mensajería y eventos

G129

RESPUESTA

Publisher → topic → subscription → subscriber (pull o push).
 Entrega AT-LEAST-ONCE: puede duplicar → el consumidor debe ser idempotente.
 Ack deadline: si no confirmas, se reenvía.
 Dead-letter topic tras N intentos fallidos. Ordering keys para orden por clave.

INTERMEDIO

GCP

Mensajería y eventos

G132

RESPUESTA

Schema registry (Avro/Protobuf): valida el payload al publicar — contrato versionado con compatibilidad (backward/forward).
 Filter de subscription: atributos (type: "order.created" && region: "eu") → cada servicio solo recibe lo suyo sin fan-out manual.
 BigQuery subscription: escribe directo a tabla (ETL sin consumer).
 Retention de mensajes (hasta 31 días) para replay.

INTERMEDIO

GCP



Mensajería y eventos

G131

RESPUESTA

Pull: el consumidor pide mensajes (control total de ritmo, ideal para workers y batch).
 Push: Pub/Sub llama a tu endpoint HTTPS (webhooks, Cloud Run) — simple, escala automática, pero control limitado y el receptor debe responder 2xx rápido.
 Push con OIDC token para autenticar el caller.
 Streaming pull: alta tasa con cliente gestionando más eficientemente.

INTERMEDIO

GCP

Mensajería y eventos

G134

RESPUESTA

Origen: app cambia estado → publica a Pub/Sub (o Eventarc escucha Audit Logs de GCS/BQ).
 Eventos fluyen: subscriptions filtradas por tipo → Cloud Run consumers (escalado por mensaje).
 Efectos: BD actualizada, notificaciones, analítica (BigQuery subscription).
 Reglas: eventos inmutables en pasado ('PedidoCreado'), payload con id + datos mínimos o claim-check.

INTERMEDIO

GCP



Mensajería y eventos

G133

RESPUESTA

Pub/Sub: eventos FAN-OUT a muchos suscriptores, throughput masivo, sin control fino de timing.
 Cloud Tasks: comandos punto a punto UNO A UNO con control fino: tasa de dispatch (rate limits), schedule en el futuro, retry backoff explícito, token OIDC.
 Casos Tasks: emails con rate limit de proveedor, callbacks programados, throttling a API externa.

INTERMEDIO

GCP

Mensajería y eventos

G136

RESPUESTA

Clave = eventId/messageId del evento.
 Tabla dedup: INSERT INTO processed(event_id) — si constraint violation, ya procesado → ack y skip (en la MISMA tx del negocio = atómico).
 TTL/purga de la tabla. Redis SETNX con expiración como caché de dedup rápida.
 Operaciones naturales idempotentes: upsert, set estado por versión.

INTERMEDIO

GCP



Mensajería y eventos

G135

RESPUESTA

Scheduler: cron gestionado que llama HTTP/PubSub/AppEngine (UI de cron, timezone).
 Workflows: orquestador YAML/JSON con pasos, condiciones, retries, llamadas a APIs — el 'paso a paso' serverless (sustituye scripts glue).
 Ejemplo: Scheduler diario → Workflow: extraer datos, llamar Cloud Run, si falla retry, notificar.
 Barato y sin infraestructura.

INTERMEDIO

GCP

Mensajería y eventos

G137

Poison messages y DLQ handling operativo

INTERMEDIO

GCP

Mensajería y eventos

G138

Ack deadline, max delivery y dead letter: el ciclo de vida del mensaje

AVANZADO

GCP

Mensajería y eventos

G139

Ordering keys: orden garantizado y su costo

AVANZADO

GCP

Mensajería y eventos

G140

Pub/Sub Lite vs Pub/Sub: el caso especial

AVANZADO

GCP

Mensajería y eventos

G141

Ordering + exactly-once: recipe final con Pub/Sub

AVANZADO

GCP

Analítica de datos

G142

BigQuery: modelo de costos on-demand vs slots

INTERMEDIO

GCP★

Analítica de datos

G143

Partitioning y clustering en BigQuery: el 80% de la performance

INTERMEDIO

GCP

Analítica de datos

G144

Cargar datos a BigQuery: las 5 rutas

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Mensajería y eventos

G138

RESPUESTA

Mensaje entregado queda 'in flight' hasta ack o expiración del ackDeadline (10-600s) → modAck para extender en procesamiento largo.

Tras N intentos (delivery attempt/dead letter policy en la subscription) → topic dead-letter con header de causa.

Consumidor idempotente OBLIGATORIO (at-least-once).

Seek permite re-entregar (snapshot/fecha) para re-procesar.

AVANZADO

GCP

Mensajería y eventos

G137

RESPUESTA

Mensaje que SIEMPRE falla (payload inválido): dead-letter policy tras N intentos evita bloquear la cola.

Operación: monitorear profundidad del DLQ (alerta), inspeccionar causa (header stack/reason), fix del bug, re-publicar corregidos (seek o republish tool).

Sin DLQ: un solo mensaje malo puede parar todo el pipeline.

INTERMEDIO

GCP

Mensajería y eventos

G140

RESPUESTA

Lite: particiones/zonas fijas, costo por capacidad reservada MUCHO menor para streaming continuo masivo (telemetría), modelo Kafka-like (offsets).

Pub/Sub: global, dinámico, features ricas (schemas, filters, push).

Lite recomendado para volúmenes predecibles enormes con presupuesto apretado; Pub/Sub para todo lo demás.

(Nota: está en sunset gradual hacia managed Kafka — evaluar BigQuery subscription/direct Kafka).

AVANZADO

GCP

Mensajería y eventos

G139

RESPUESTA

messageOrdering=true en publisher + orderingKey (ej. orderId) → mensajes de la MISMA clave llegan en orden al mismo suscriptor.

Costo: throughput reducido (una clave = una secuencia); si un mensaje falla, se BLOQUEA la clave hasta ack/expiración (entrega la detiene).

Exactly-once delivery (región con feature) refuerza orden + no duplicados.

Alternativa: procesar orden en el consumidor por versiones/sequence.

AVANZADO

GCP

Análítica de datos

G142

RESPUESTA

On-demand: pagas por bytes ESCANEADOS (~\$6/TB) — el WHERE por columna (columnar) importa; particionar reduce el scan.

Editions/slots: capacidad reservada (slots por segundo/hora/mensual) — predecible y mejor a escala (sin límites de bytes).

Controles: custom quotas por usuario/proyecto, maximum bytes billed por query, dry run antes de correr.

INTERMEDIO

GCP

Mensajería y eventos

G141

RESPUESTA

Ordering key por entidad + enable_message_ordering en subscription + endpoint regional (mismo región del topic).

Exactly-once delivery (regiones soportadas): no duplicados con retry del mismo mensaje.

Aun así: consumidor idempotente (dedup table) — la defensa final.

Monitorear 'oldest unacked message' por ordering key para bloqueos.

AVANZADO

GCP

Análítica de datos

G144

RESPUESTA

1) Batch load (CSV/Parquet/AVRO desde GCS) — gratis, ideal para cargas grandes puntuales.

2) Streaming insert (write API) — latencia segundos, costo por GB.

3) Storage Write API (nuevo, exactly-once, transactions) — el estándar para pipelines.

4) BigQuery subscription desde Pub/Sub — directo desde mensajería.

5) Federated queries (tabla externa sobre GCS/Cloud SQL) — sin mover datos.

INTERMEDIO

GCP



Análítica de datos

G143

RESPUESTA

Partition por DATE/TIMESTAMP (o integer range): cada query con WHERE fecha = 'X' escanea UNA partición (no toda la tabla).

Clustering por columnas frecuentes de filtro (user_id, pais): ordena bloques dentro de la partición → skips.

Result: terabytes evitados = dinero y velocidad.

Re-particionar tabla grande: create table partition by... as select.

INTERMEDIO

GCP

Analítica de datos

G145

Looker Studio / Looker: el extremo BI

INTERMEDIO

GCP

Analítica de datos

G146

Dataflow (Apache Beam): batch y streaming unificados

AVANZADO

GCP

Analítica de datos

G147

Lakehouse en GCP: GCS + BigQuery + Dataplex

AVANZADO

GCP

Analítica de datos

G148

Composer (Airflow gestionado): DAGs como código

AVANZADO

GCP

Analítica de datos

G149

Streaming pipeline canónico: Pub/Sub → Dataflow → BigQuery

AVANZADO

GCP

Analítica de datos

G150

BigQuery ML: modelos sin salir del almacén

AVANZADO

GCP

Analítica de datos

G151

Vertex AI: qué ofrece end-to-end

AVANZADO

GCP

★ DevOps y CI/CD

G152

CI/CD en GCP: Cloud Build + Artifact Registry

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Analítica de datos

G146

RESPUESTA

Pipeline en Java/Python → corre en managed Dataflow: auto-scaling de workers, exactly-once en streaming.
 Casos: ETL por lotes, procesamiento de streams (ventanas, agregaciones), correcciones tardías (watermarks).
 Costo por vCPU/RAM de workers — tuning con autoscaling y streaming engine.
 Alternativa light: Dataproc (Spark) si el equipo ya tiene Spark.

AVANZADO

GCP

Analítica de datos

G145

RESPUESTA

Looker Studio (gratis): dashboards conectados a BigQuery/Sheets — reportes operativos rápidos.
 Looker (enterprise): capa semántica (LookML) — métricas definidas UNA vez, gobernadas, self-service real.
 Patrón de entrega de datos: BigQuery mart → dashboard ejecutivo.
 Embed: dashboards embebidos para clientes SaaS.

INTERMEDIO

GCP

Analítica de datos

G148

RESPUESTA

Composer = Apache Airflow gestionado: DAGs en Python versionados en GCS, schedule/orquestación con dependencias entre tareas.
 Casos: pipelines ETL multi-step (extraer Cloud SQL → Dataflow → BigQuery → Looker), ML retraining.
 Operación: entorno en su propio GKE (costo fijo mensual notable), monitoring de tasks fallidas.
 Alternativa serverless: Workflows para orquestaciones simples.

AVANZADO

GCP

Analítica de datos

G147

RESPUESTA

Data lake: GCS (Parquet) como almacenamiento barato por zonas (raw/curated/consumed).
 BigQuery consulta lakehouse: tablas externas / BigLake (metadatos, seguridad a nivel fila/columna, caché).
 Dataplex: catálogo, calidad y gobernanza por zonas.
 ETL con Dataflow/Dataproc; BI con Looker/Looker Studio.

AVANZADO

GCP

Analítica de datos

G150

RESPUESTA

CREATE MODEL ... OPTIONS(model_type='logistic_reg') AS SELECT ... — entrena en SQL.
 Predicción: ML.PREDICT(MODEL m, TABLE t); evaluación ML.EVALUATE.
 Tipos: regresión, clasificación, clustering (kmeans), time series (ARIMA), matrices.
 Casos: churn simple, forecasting de demanda — sin mover datos a otra plataforma.
 Para ML serio: Vertex AI, pero BQML cubre el 80% analítico.

AVANZADO

GCP

Analítica de datos

G149

RESPUESTA

Eventos (clicks, sensores, transacciones) → Pub/Sub topic → Dataflow streaming: parse/validate/enriquecer, ventanas por minuto → BigQuery tablas particionadas.
 Exactly-once con Storage Write API sink; late data con watermarks.
 Monitoreo: system lag de Pub/Sub, watermarks de Dataflow, frescura de BQ.
 Misma topología sirve para agregados en tiempo real (Firestore/Memorystore sink).

AVANZADO

GCP



DevOps y CI/CD

G152

RESPUESTA

Cloud Build: ejecuta pipelines definidos en cloudbuild.yaml (build, test, push, deploy) con triggers desde Git.
 Artifact Registry: almacén de imágenes Docker y paquetes (npm, Maven, Python) con IAM y análisis de vulnerabilidades.
 Flujo típico: build → push imagen → gcloud run deploy.

INTERMEDIO

GCP

Analítica de datos

G151

RESPUESTA

Datasets + entrenamiento (AutoML / custom containers), Feature Store, Model Registry, endpoints de serving (online/batch), Pipelines (Kubeflow) para MLOps, model monitoring de drift.
 Generative: Vertex AI Studio, Gemini API, grounding con tus datos (RAG), embeddings + Vector Search.
 Costo: cómputo por hora de entrenamiento + serving por nodo-hora.
 Es el 'AWS SageMaker' de GCP.

AVANZADO

GCP

★

DevOps y CI/CD

G153

Cloud Build: pipeline básico en cloudbuild.yaml

INTERMEDIO

GCP

★

DevOps y CI/CD

G154

Triggers de Cloud Build: por branch, tag, PR y manual

INTERMEDIO

GCP

★

DevOps y CI/CD

G155

Secretos en Cloud Build (sin exponerlos en logs)

INTERMEDIO

GCP

DevOps y CI/CD

G156

CD a Cloud Run: build → deploy → smoke test

INTERMEDIO

GCP

DevOps y CI/CD

G157

Estrategia de branching y deploy por entornos en GCP

INTERMEDIO

GCP

DevOps y CI/CD

G158

Terraform en GCP: estructura de proyecto típica

INTERMEDIO

GCP

DevOps y CI/CD

G159

Cloud Deploy: progresión gestionada dev→prod

INTERMEDIO

GCP

DevOps y CI/CD

G160

Pipelines de imágenes seguras: scan → sign → deploy

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP ★
RESPUESTA

DevOps y CI/CD

G154

Push a a branch (main→deploy stage), push a tag (v*→prod), Pull Request (solo build/test), manual (deploy one-off).
Filtros por ruta incluidos/excluded files (solo build si cambia /api).
Service account del build con permisos mínimos (deploy a Run, nada más).
Aprobaciones (approvals) antes de desplegar a prod.

INTERMEDIO

GCP ★
RESPUESTA

DevOps y CI/CD

G153

steps: [{name: gcr.io/cloud-builders/docker, args: [build, -t, \$LOCATION-docker.pkg.dev/\$PROJECT_ID/repo/app:\$SHORT_SHA, .]} , {name: docker, args: [push, ...]}]
images, substitutions (\$PROJECT_ID, \$SHORT_SHA, \$_VARS custom), availableSecrets para tokens.
gcloud builds submit --config cloudbuild.yaml .
Trigger desde GitHub: push a main → build → deploy.

INTERMEDIO

GCP
RESPUESTA

DevOps y CI/CD

G156

Step 1 docker build/push; step 2 gcloud run deploy svc --image ... --region --quiet; step 3 curl smoke test al URL de la revisión tagged; si falla → rollback traffic a revisión anterior.
Alternativa declarativa: Cloud Deploy (pipeline dev→stage→prod con approvals y rollbacks unificados para Run/GKE).
Todo como código en el repo del servicio.

INTERMEDIO

GCP ★
RESPUESTA

DevOps y CI/CD

G155

availableSecrets: secretManager: [versionName: projects/p/secrets/npm-token/versions/latest] + step env: NPM_TOKEN=secretManager:secretManager:secretName.
El valor no aparece en logs (enmascarado).
Credenciales de deploy: la SA del build con roles mínimos, impersonation en lugar de keys.
Nunca COPY secrets en la imagen.

INTERMEDIO

GCP
RESPUESTA

DevOps y CI/CD

G158

modules: network (VPC/subnets/NAT), project-factory (proyecto + APIs + budgets), run-service (deploy app), data (SQL + backup).
Environments: dev.tfvars / prod.tfvars sobre el mismo módulo.
Remote state en GCS + lock. CI: plan en PR (comentario del plan), apply manual o auto en merge para dev.
Version pinning de providers.

INTERMEDIO

GCP
RESPUESTA

DevOps y CI/CD

G157

Trunk-based + feature flags: main siempre desplegable → dev automático; stage con tag/merge; prod con release tag + aprobación.
Entornos = proyectos GCP distintos (misma receta Terraform) — nunca carpetas en el mismo proyecto.
Promoción de la MISMA imagen entre entornos (solo cambia config) — no re-builds por entorno.

INTERMEDIO

GCP
RESPUESTA

DevOps y CI/CD

G160

Cloud Build → Artifact Registry con on-push vulnerability scanning (Critical/High bloquea) → attestación firmada (Binary Authorization) → solo imágenes firmadas despliegan en GKE/Run.
SBOM generado (syft) adjunto; provenance SLSA de la build.
Base images minimal (distroless/chainguard) reducen CVEs de raíz.

INTERMEDIO

GCP
RESPUESTA

DevOps y CI/CD

G159

Pipeline declarativo: targets (dev, stage, prod), strategy (standard o canary con % por fases), approvals en prod, rollbacks de 1 click.
Renderiza scaffold profiles (config por target) y despliega a Run/GKE.
Historial de releases y estados centralizado — para equipos sin plataforma propia de CD.

INTERMEDIO

GCP

DevOps y CI/CD

G161

**Feature flags en pipeline de GCP:
decoupling deploy/release**

INTERMEDIO

GCP

DevOps y CI/CD

G162

**Monorepo vs multirepo para microservicios
+ CI**

INTERMEDIO

GCP

DevOps y CI/CD

G163

**Migraciones de schema seguras en CI/CD
(expand-contract)**

INTERMEDIO

GCP

DevOps y CI/CD

G164

GitOps con GKE: ArgoCD/Config Sync

AVANZADO

GCP ★

DevOps y CI/CD

G165

**Private pools de Cloud Build: builds en tu
VPC**

AVANZADO

GCP ★

Observabilidad y SRE

G166

**Cloud Operations: Logging, Monitoring,
Trace**

INTERMEDIO

GCP

Observabilidad y SRE

G167

**Cloud Logging: consultas y sinks que debes
dominar**

INTERMEDIO

GCP

Observabilidad y SRE

G168

**Cloud Monitoring: uptime, alertas y
políticas**

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

DevOps y CI/CD

G162

RESPUESTA

Monorepo: atomicidad de cambios cruzados, refactors globales; CI con filtros por carpeta (solo build lo tocado); governance única.

Multirepo: ownership y releases independientes, permisos simples; versiones y sync manual entre repos.

GCP: Cloud Build triggers con included files para monorepo; un trigger por servicio en multirepo.

Ambos válidos; el equipo manda.

INTERMEDIO

GCP

DevOps y CI/CD

G161

RESPUESTA

Deploy frecuente + flag apagado = código en prod invisible al usuario.

Gestión: Unleash self-hosted en GKE / LaunchDarkly / Firebase Remote Config (móvil) / properties en Firestore.

Kill switch para apagar feature problemática sin rollback.

Higiene: dueño + fecha de expiración por flag (los flags eternos son deuda).

INTERMEDIO

GCP

DevOps y CI/CD

G164

RESPUESTA

Git = fuente de verdad del estado del clúster: ArgoCD sincroniza manifiestos/Helm desde repo, diff visible, rollback = git revert.

Config Sync (Anthos): sincroniza configs/namespaces/policies a nivel org.

Separación: repo de app (código) vs repo de despliegue (manifiestos) — CI actualiza el segundo, CD sincroniza.

Beneficio: auditoría y recuperación total del clúster desde git.

AVANZADO

GCP

DevOps y CI/CD

G163

RESPUESTA

Deploy N: migración ADDITIVA (añadir columna nullable + backfill) — compatible con versión vieja.

Deploy código N (usa ambas). Deploy código N+1 (solo nueva).

Deploy N+2: migración contractiva (drop viejo).

Regla: cada migración debe correr contra el código de la versión PREVIA — nunca renombrar/dropear en el mismo release.

Flyway/Liquibase versionados por servicio.

INTERMEDIO

GCP ★

Observabilidad y SRE

G166

RESPUESTA

Logging: logs centralizados de todos los servicios, consultas, sinks a BigQuery/Storage, log-based metrics.

Monitoring: dashboards, uptime checks, alert policies y SLOs.

Cloud Trace: distributed tracing de latencias entre servicios.

Profiler: uso continuo de CPU/memoria.

INTERMEDIO

GCP ★

DevOps y CI/CD

G165

RESPUESTA

Worker pool dedicado dentro de tu VPC: acceso a recursos privados (Cloud SQL de test, Memorystore, repos internos) sin exponerlos.

Control de máquina (tipo, disco) y de egress.

Costo por tiempo de worker; usar solo para builds que requieren red privada — el resto en el pool compartido.

AVANZADO

GCP

Observabilidad y SRE

G168

RESPUESTA

Uptime checks: sonda externa cada N min desde varias regiones (URL, grupos).

Alert policies: condición sobre métrica (error rate > 5% en 5m) → notificación (email, SMS, Slack, Pub/Sub, PagerDuty).

Throttling/aggregation: ventanas, alineadores (mean, count), 'for X minutes' para evitar ruido.

Incidentes con timeline y ack.

INTERMEDIO

GCP

Observabilidad y SRE

G167

RESPUESTA

Queries: resource.type="cloud_run_revision" severity>=ERROR textPayload:"timeout"; guardar queries.

Sinks: enrutamiento a BigQuery (analítica), GCS (archivo barato), Pub/Sub (alertas custom/SIEM).

Exclusion sinks: bajar costo descartando logs ruidosos en el origen.

Log-based metrics: contar ocurrencias de un patrón → métrica + alerta.

INTERMEDIO

GCP

Observabilidad y SRE

G169

Cloud Trace y Profiler: diagnóstico de latencia y CPU

INTERMEDIO

GCP

Observabilidad y SRE

G170

Error Reporting: el dashboard de excepciones

INTERMEDIO

GCP

Observabilidad y SRE

G171

Alertas que no despiertan a nadie en vano: higiene

INTERMEDIO

GCP

★

Observabilidad y SRE

G172

Golden signals: qué monitorear en TODO servicio

INTERMEDIO

GCP

Observabilidad y SRE

G173

SLOs en Cloud Monitoring: del concepto al setup

AVANZADO

GCP

Observabilidad y SRE

G174

Managed Service for Prometheus y Grafana en GCP

AVANZADO

GCP

Observabilidad y SRE

G175

Log analytics y queries sobre logs en BigQuery

AVANZADO

GCP

gcloud y herramientas

G176

Console: atajos y funcionalidades que la gente no conoce

BASICO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Observabilidad y SRE

G170

RESPUESTA

Agrupar stack traces idénticos de logs (Cloud Run/GKE/Functions escriben con formato estándar) → conteo, usuarios afectados, primera/última vez.

Alerta sobre nuevo error group (fade-in).

Para Java: loggear Throwable con severity=ERROR y el agente lo agrupa.

Rutina: revisar grupos nuevos en cada deploy.

INTERMEDIO

GCP

Observabilidad y SRE

G169

RESPUESTA

Trace: waterfall de una request entre servicios (spans por llamada/SQL) — localizar el 300ms escondido; sampling configurable.

Profiler (continuous): CPU/heap muestreado en producción sin overhead — flame graphs reales de hotspots.

Ambos casi gratis de instrumentar con Cloud Ops agents/client libs (Java: auto-instrumentación Micrometer Tracing).

INTERMEDIO

GCP

Observabilidad y SRE

G172

RESPUESTA

Latency (p50/p95/p99 por endpoint), Traffic (req/s), Errors (4xx/5xx rate), Saturation (CPU, memoria, conexiones pool, profundidad de colas).

Por servicio: 4 dashboards que responden 'está sano?'.

Dependencias: latencia y tasa de error de llamadas externas (son tus cuellos más frecuentes).

INTERMEDIO

GCP

Observabilidad y SRE

G171

RESPUESTA

Alertar sobre SÍNTOMAS que afectan usuarios (SLO burn, error rate), no causas triviales (CPU alta transitoria).

Ventanas de estabilización (for 5 min), thresholds basados en p95, escalamiento jerárquico, runbook link en cada alerta.

Revisar alertas disparadas al mes: <50% accionables = refinar.

El objetivo: cada página = acción clara o se elimina.

INTERMEDIO

GCP

Observabilidad y SRE

G174

RESPUESTA

Managed Prometheus: ingesta de métricas de Prometeo gestionada (pagas por samples), sin operar servers — GKE lo scrapea automático.

Cloud Monitoring soporta PromQL.

Grafana: hosted por Google en Cloud Monitoring (dashboards de PromQL) o self-hosted en GKE contra el datasource managed.

Migración típica: Prometheus self-hosted → managed para quitar ops.

AVANZADO

GCP

Observabilidad y SRE

G173

RESPUESTA

SLI: métrica de servicio percibida (disponibilidad = % requests 5xx-free; latencia = % requests < 300ms).

SLO: objetivo en ventana (99.9% / 30 días rolling). Error budget = 100% - SLO.

Monitoring: crear SLO sobre service → burn rate alerts (rápido 14.4x/1h y lento 6x/6h).

Aterrizaje: pausa de releases cuando el budget está agotado (política de equipo).

AVANZADO

GCP

gcloud y herramientas

G176

RESPUESTA

Search global (barra superior) encuentra recursos por nombre/ID.

Active Assist: recommendations centralizadas.

Network Analyzer / Connectivity Tests.

Cloud Console mobile app para alertas/uptime on-call.

Keyboard shortcuts (?) y filtros guardados en tablas.

Recent projects: cambiar rápido entre entornos.

BASICO

GCP

Observabilidad y SRE

G175

RESPUESTA

Log Analytics (bucket de logs con Linked dataset): consultar logs en BigQuery con SQL — tendencias de errores, latencia por endpoint desde LOGS (no métricas).

Ejemplo: `SELECT httpRequest.requestUrl, COUNT(*) FROM logs WHERE severity='ERROR' GROUP BY 1 ORDER BY 2 DESC.`

Retención extendida barata vs default 30 días.

Base de análisis de incidentes post-mortem.

AVANZADO

GCP

gcloud y herramientas

G177

Comandos gcloud del día a día (chuleta)

INTERMEDIO

GCP

gcloud y herramientas

G178

gsutil vs gcloud storage: la sucesión

INTERMEDIO

GCP

gcloud y herramientas

G179

Logs y debugging desde CLI

INTERMEDIO

GCP

gcloud y herramientas

G180

IAM desde CLI: inspección y simulación

INTERMEDIO

GCP

gcloud y herramientas

G181

Cloud Code / Cloud Workstations: desarrollo en el cloud

INTERMEDIO

GCP

gcloud y herramientas

G182

Costos por CLI y presupuesto programático

INTERMEDIO

GCP

gcloud y herramientas

G183

Cloud Asset Inventory: saber qué tienes corriendo

INTERMEDIO

GCP

Patrones de arquitectura

G184

Arquitectura canónica: app web moderna en GCP

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

gcloud y herramientas

G178

RESPUESTA

gsutil (legado) → gcloud storage (reemplazo moderno): `cp -r, rsync (delta), ls -l gs://bucket/prefijo**, mv, rm -a` (todas las versiones), `sign-url generate`.

Patrones con wildcards y `--recursive`; `--billing-project` para requests en nombre de otro proyecto.

Para pipelines: client libs o Storage Transfer, no shell.

INTERMEDIO

GCP

gcloud y herramientas

G177

RESPUESTA

`gcloud config set project/set compute/region`.

`run: gcloud run deploy svc --image img --region us-central1; run services update-traffic`.

`compute: instances create/list, ssh NAME --tunnel-through-iap`.

`sql: connect INSTANCE --user=root; sql backups create`.

`iam: auth list, auth login --update-adc; projects get-iam-policy`.

`--format/--filter/--quiet` para scripts.

INTERMEDIO

GCP

gcloud y herramientas

G180

RESPUESTA

`gcloud projects get-iam-policy PROJECT --format=json`.

`get-ancestors-iam-policy` (quién me da este permiso).

Policy Troubleshooter (console o API): '¿por qué no puedo X?' — evalúa el binding real (condiciones incluidas).

Recommender IAM: permisos no usados en 90 días → sugerencia de roles más ajustados.

INTERMEDIO

GCP

gcloud y herramientas

G179

RESPUESTA

`gcloud logging read 'resource.type=cloud_run_revision AND severity>=ERROR' --limit 20 --format json`.

`gcloud run services logs read svc --region ... (tail -f)`.

`gcloud compute instances get-serial-port-output` (boot failures).

Streaming en console (Log Router live tail) para deploys en caliente.

INTERMEDIO

GCP

gcloud y herramientas

G182

RESPUESTA

`gcloud billing accounts list; budgets create` con `--billing-account --budget-amount --threshold-rule`.

BigQuery billing export queries para reportes custom por label.

API Cloud Billing Catalog para estimar precios en tooling.

Alerta 100% + acción: función que apaga entornos dev de noche (scheduler + script) — ahorro automático.

INTERMEDIO

GCP

gcloud y herramientas

G181

RESPUESTA

Cloud Code (extensión IDE): deploy/debug de Cloud Run y GKE desde VSCode/IntelliJ, skaffold integrado.

Cloud Workstations: entornos dev gestionados en la nube (CPU/RAM configurables, preinstalado, efímeros) — onboarding en minutos, código nunca en laptops.

Emulators locales: pubsub-emulator, datastore, bigquery (bq beta emulators) para tests sin GCP real.

INTERMEDIO

GCP

Patrones de arquitectura

G184

RESPUESTA

Cloud DNS + Cloud LB (global HTTPS) → Cloud Run (API) + GCS/CDN (frontend) → Cloud SQL (HA) + Memorystore (caché) → Secret Manager/IAM.

Observabilidad: Cloud Ops. CI/CD: Cloud Build + Artifact Registry.

IaC: Terraform.

Escala: Run auto, SQL read replica si hace falta. DR: backup cross-region.

Es el punto de partida del 80% de proyectos.

INTERMEDIO

GCP

gcloud y herramientas

G183

RESPUESTA

Inventario de TODOS los recursos de la org en un punto del tiempo + histórico de cambios (feed a BigQuery/PubSub).

Queries tipo: 'todas las VMs con IP pública', 'buckets sin CMEK', 'SAs con keys'.

Base de compliance continua y de limpieza de recursos zombies.

`gcloud asset search-all-resources --asset-types=...`

INTERMEDIO

GCP

Patrones de arquitectura

G185

Backend for a mobile app: qué cambia en el diseño

INTERMEDIO

GCP

Patrones de arquitectura

G186

Multi-tenant SaaS en GCP: 3 modelos de aislamiento

INTERMEDIO

GCP

Patrones de arquitectura

G187

API management: API Gateway vs Apigee vs Endpoints

INTERMEDIO

GCP

Patrones de arquitectura

G188

Exponer Cloud Run con dominio propio: opciones

INTERMEDIO

GCP

Patrones de arquitectura

G189

Webhooks entrantes (pagos, mensajería): diseño robusto

INTERMEDIO

GCP

Patrones de arquitectura

G190

Jobs nocturnos y batch sin servidores: opciones y criterio

INTERMEDIO

GCP

Escenarios de entrevista

G191

«Tu Cloud Run API se cae a las 3 a.m.» — runbook de diagnóstico

INTERMEDIO

GCP

Escenarios de entrevista

G192

«Migrar una app on-prem con BD a GCP» — camino realista

INTERMEDIO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Patrones de arquitectura

G186

RESPUESTA

Pool (todos los tenants en la misma BD con tenant_id + row-level security) — barato.
 Silos (BD/proyecto por tenant) — máximo aislamiento, caro de operar a escala.
 Bridge (BD compartida, esquema o servicio por tenant grande).
 Híbrido realista: pool para pequeños, silo para enterprise (cada silo es un proyecto con Terraform).

INTERMEDIO

GCP

Patrones de arquitectura

G185

RESPUESTA

Auth de usuarios finales: Identity Platform/Firebase Auth (JWT en headers).
 APIs optimizadas para móvil: payloads pequeños, paginación por cursor, batching en BFF.
 Push: FCM. Offline: Firestore local cache.
 CDN para assets, Cloud Run para API — el tráfico móvil es spiky: serverless absorbe picos sin pagos fijos.

INTERMEDIO

GCP

Patrones de arquitectura

G188

RESPUESTA

1) Domain mapping directo de Cloud Run (simple, sin LB).
 2) LB global + serverless NEG → Cloud Armor + CDN + certificado gestionado (el estándar para prod serio).
 3) Firebase Hosting rewrites a Run (frontend + API juntos).
 DNS: CNAME/A al endpoint. TTL bajo durante cutover.

INTERMEDIO

GCP

Patrones de arquitectura

G187

RESPUESTA

API Gateway (serverless, barato): auth JWT, routing a Cloud Run/Functions, quotas — para APIs simples.
 Apigee (enterprise): portal de desarrolladores, monetización, analítica por producto de API, políticas avanzadas — para exponer APIs COMERCIALES.
 Endpoints (legado, quizá evitar en nuevo).
 Puerta de entrada + WAF (Cloud Armor) completa el borde.

INTERMEDIO

GCP

Patrones de arquitectura

G190

RESPUESTA

Cloud Run Jobs: contenedor batch, paralelismo por índice, retry por tarea — el default moderno.
 Cloud Scheduler + Tasks: encadenar pasos con rate control.
 Batch service: para cómputo HPC masivo (miles de VMs spot).
 Dataflow: si es procesamiento de datos continuo/paralelo real.
 Criterio: duración, paralelismo, dependencias.

INTERMEDIO

GCP

Patrones de arquitectura

G189

RESPUESTA

Endpoint idempotente por event id (dedup table) → aceptar 200 RÁPIDO y encolar el procesamiento (Pub/Sub/Tasks) → worker procesa con reintentos.
 Verificación de firma del proveedor (HMAC) ANTES de aceptar.
 Replay safety: proveedores re-envían; tu sistema debe tolerarlo.
 Monitoreo de latencia y tasa de fallo por proveedor.

INTERMEDIO

GCP

Escenarios de entrevista

G192

RESPUESTA

Fase 0: inventario y assessment (compatibilidad, tamaño, latencia).
 Lift & shift: Compute Engine + Cloud VPN/Interconnect + DMS para la BD (CDC, cutover controlado).
 Después (no en paralelo): contenerizar → Cloud Run/GKE, reemplazar componentes por managed (Secrets, Storage).
 Cutover por dominios con rollback planeado. Nunca modernizar y migrar a la vez.

INTERMEDIO

GCP

Escenarios de entrevista

G191

RESPUESTA

1) Dashboards: error rate, latencia, instancias, saturación.
 2) Logs con traceId de requests fallidos → excepción dominante.
 3) Dependencias: estado de Cloud SQL (failover?), límites de cuota (429?), circuit breakers abiertos.
 4) Deploy reciente? → rollback de revisión (1 comando) mientras se investiga.
 5) Postmortem sin culpa + alerta que debió dispararse antes.

INTERMEDIO

GCP

Escenarios de entrevista

G193

«Otro equipo debe consumir nuestros datos» — cómo exponerlos bien

INTERMEDIO

GCP

Escenarios de entrevista

G194

«¿Compute Engine o Cloud Run para un microservicio Java?» — respuesta estructurada

INTERMEDIO

GCP

Escenarios de entrevista

G195

«Auditoría: quién vio/exports datos de clientes» — implementación

INTERMEDIO

GCP

Patrones de arquitectura

G196

Procesamiento de archivos que suben usuarios: patrón completo

AVANZADO

GCP

Patrones de arquitectura

G197

Aplicación multi-región activa-activa: checklist

AVANZADO

GCP

Patrones de arquitectura

G198

Hybrid y multi-cloud: Anthos y el sentido real

AVANZADO

GCP

Patrones de arquitectura

G199

CQRS aplicado: lectura escalada con BigQuery/Firestore

AVANZADO

GCP

Patrones de arquitectura

G200

Anti-patrones GCP frecuentes (y cómo se ven en la factura)

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Escenarios de entrevista

G194

RESPUESTA

Preguntar primero: estado, tráfico (spiky?), equipo, licencias.
Default Cloud Run: contenedor, autoscaling, revisiones/canary gratis, sin parcheo de SO.

Compute Engine si: JVM tuning profundo (heap grande, GC específico), licencias por core/nodo, red especial (multicast), sidecars del sistema.

GKE si ya hay plataforma k8s y muchos servicios con mesh.

Justifica con costo operativo, no con preferencia.

INTERMEDIO

GCP

Escenarios de entrevista

G193

RESPUESTA

Opciones por acoplamiento: API REST/gRPC directa (sync, acopla), Pub/Sub topic con schema (eventos, desacopla), vista/materialized en BigQuery con data sharing (analítica), Data Share/Analytics Hub para datasets gobernados.

Contrato: schema versionado + SLA + auth por SA del consumidor con rol mínimo (publisher/invoke).

Nunca: acceso directo a TU base de datos (acoplamiento total).

INTERMEDIO

GCP

Patrones de arquitectura

G196

RESPUESTA

1) Backend firma V4 signed URL → cliente sube directo a GCS (no pasa por la API).

2) Notificación: Eventarc (GCS finalize) o Pub/Sub → Cloud Run procesa (valida, transformar, guarda metadata en SQL/Firestore).

3) Estado y errores: tabla de jobs + Cloud Tasks retry.

Ventajas: API ligera, escala automática, resumable uploads nativos.

AVANZADO

GCP

Escenarios de entrevista

G195

RESPUESTA

Data Access audit logs (ADMIN_READ/DATA_READ) habilitados para los servicios de datos sensibles → sink a BigQuery.

Queries: por objeto accedido, por usuario, por método (get/download).

Alertas: log-based metric para accesos masivos anómalos (exfil detection).

Aplicación: tabla de acceso a PII con propósito (por qué lo vio) si el compliance lo exige.

INTERMEDIO

GCP

Patrones de arquitectura

G198

RESPUESTA

Anthos/GKE Enterprise: plano de gestión para GKE on-cloud, on-prem y otros clouds — políticas, config sync y service mesh unificados.

Casos reales: migración gradual, cumplimiento de datos on-prem, evitar vendor lock-in de plataforma.

Costo: licencias + complejidad operativa significativa — justificar con requisito concreto, no con moda.

AVANZADO

GCP

Patrones de arquitectura

G197

RESPUESTA

Datos: Spanner multi-región (o SQL cross-region replicas manual), GCS multi-region, Memorystore regional por región.

Tráfico: LB global anycast → Cloud Run en 2+ regiones (health checks regionales).

Estado de sesión: stateless + JWT; caché por región.

DNS/routing: LB decide por latencia; failover automático.

Pruebas: caos regional simulado cada trimestre.

AVANZADO

GCP

Patrones de arquitectura

G200

RESPUESTA

VMs sin apagar en dev (24/7 por hábito) → CUD de cosas que no necesitas.

Logs ruidosos sin exclusiones → Cloud Logging caro.

Egress directo de GCS a usuarios sin CDN → transfer cost alto.

NAT de 1 instancia pero min-instances altas en Run.

BigQuery sin particionar → scans de TB por dashboard.

Cada uno tiene fix mecánico — auditar mensualmente.

AVANZADO

GCP

Patrones de arquitectura

G199

RESPUESTA

Escrituras en Cloud SQL (transaccional, normalizado) →

CDC/Dataflow → vistas denormalizadas en Firestore (consultas de app) o BigQuery (analítica).

El servicio de escritura no compete con lecturas masivas; cada lado optimiza su patrón.

Lag eventual de segundos: aceptable para feeds/dashboards, no para saldo bancario.

AVANZADO

GCP

Escenarios de entrevista

G201

«Diseñe un sistema de notificaciones para 10M usuarios» — boceto GCP

AVANZADO

GCP

Escenarios de entrevista

G202

«Necesito garantizar que facturas NUNCA se pierden» — patrón

AVANZADO

GCP

Escenarios de entrevista

G203

«Reduzca el 40% del costo sin tocar features» — plan de ataque

AVANZADO

GCP

Escenarios de entrevista

G204

«La app es lenta solo para usuarios de otro continente» — diagnóstico

AVANZADO

GCP

Escenarios de entrevista

G205

«Cumpla GDPR/PII en GCP» — controles concretos

AVANZADO

GCP

Escenarios de entrevista

G206

«Necesito un chat en tiempo real» — componentes GCP

AVANZADO

GCP

Escenarios de entrevista

G207

«Buscador sobre millones de documentos» — dónde entra cada servicio

AVANZADO

GCP

Escenarios de entrevista

G208

«Zero downtime deploy» — opciones por plataforma GCP

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

Escenarios de entrevista

G202

RESPUESTA

Escritura transaccional a Cloud SQL (factura + outbox event en la MISMA tx).

Publisher outbox (poller/CDC Debezium) → Pub/Sub con retention + DLQ.

Consumidor idempotente persiste en destino; reconciliación diaria compara conteos origen/destino.

Backups PITR + retention de GCS = durabilidad de datos; el outbox resuelve la entrega.

AVANZADO

GCP

Escenarios de entrevista

G201

RESPUESTA

API recibe evento → Pub/Sub (topic notificaciones) → Dataflow/Cloud Run workers por canal: FCM (push), SendGrid (email), SMS — cada canal con rate limiting propio (Cloud Tasks). Preferencias de usuario en Firestore (por usuario, lectura rápida). Dedup + throttling por usuario.

Métricas: entregados/fallidos por canal; DLQ para fallos; reintentos.

Escala: fan-out por shards (region/hash) para no hotspotear.

AVANZADO

GCP

Escenarios de entrevista

G204

RESPUESTA

Medir: ¿latencia de red (TTFB) o de aplicación (server time)? RUM/ping desde esa región.

Red: LB global anycast + CDN para estáticos (edge caching) — el fix más común.

Datos: BD en una sola región = RTT por query → read replica regional o caché local (Memorystore) de datos calientes.

Verificar handshake TLS y keep-alive (conexiones nuevas por request matan latencia).

AVANZADO

GCP

Escenarios de entrevista

G203

RESPUESTA

1) Billing export: top SKUs y por label/equipo — dónde está el dinero REAL.

2) Quick wins: dev apagado de noche (scheduler), IPs estáticas huérfanas, snapshots viejos, logs con exclusion.

3) Compute: right-sizing con Recommender, spot en batch, CUDs sobre la base estable.

4) Datos: storage classes por edad, particionado BQ, CDN para egress.

5) Medir y repetir; presupuesto + alertas para sostener.

AVANZADO

GCP

Escenarios de entrevista

G206

RESPUESTA

Transporte: WebSocket con STOMP (GKE) o SSE + polling (Cloud Run) — mantener conexiones requiere always-allocated CPU en Run o websockets en GKE.

Fan-out: Pub/Sub entre instancias/réplicas (todos los pods reciben el mensaje del room).

Persistencia: Firestore (realtime listeners para clientes web/móvil gratis).

Presencia y rate limiting: Memorystore.

Firebase Realtime/Firestore directo simplifica si el equipo es pequeño.

AVANZADO

GCP

Escenarios de entrevista

G205

RESPUESTA

Residencia: org policy resourceLocations + regiones de datos de cada servicio.

Acceso: IAM mínimo + VPC-SC (perímetro) + CMEK si aplica.

Ciclo de vida: retención automática (GCS lifecycle, BQ table expiration), derecho al olvido = jobs de borrado trazables.

Detección: Sensitive Data Protection escaneando buckets/tablas.

Auditoría: data access logs habilitados y exportados.

AVANZADO

GCP

Escenarios de entrevista

G208

RESPUESTA

Cloud Run: revisiones + traffic split (canary 5% → 100%), rollback instantáneo.

GKE: rolling update (maxSurge/maxUnavailable) o canary con Gateway API/Istio.

MIG: rolling updaters con health checks.

Base de datos: migraciones compatibles (expand → deploy → contract) para que N y N+1 convivan.

La parte dura casi siempre es el SCHEMA, no el deploy.

AVANZADO

GCP

Escenarios de entrevista

G207

RESPUESTA

Ingesta: GCS + Eventarc → Dataflow (parse/enrich) → índice.

Búsqueda full-text gestionada: Elastic Cloud en Marketplace,

Vertex AI Search (antes Enterprise Search), o pg_trgm/tsvector en Cloud SQL para escala chica.

Embeddings: Vertex AI text-embedding + vector search (Matching Engine) para semántica.

Queries: API en Cloud Run con caché de resultados calientes.

AVANZADO

FRENTE — imprime a DOBLE CARA volteando por el BORDE LARGO · escala 100%

GCP

Escenarios de entrevista

G209

«Rate limiting global para una API multi-región» — diseño

AVANZADO

GCP

Patrones de arquitectura

G210

Patrón claim check: payloads grandes por mensajería

AVANZADO

GCP

Patrones de arquitectura

G211

Compensación y reconciliación diaria: la red de seguridad

AVANZADO

GCP

Escenarios de entrevista

G212

«Justifica tu arquitectura ante un CFO» — lenguaje de costo

AVANZADO

REVERSO — cada respuesta queda al dorso de su pregunta (misma numeración)

GCP

RESPUESTA

Patrones de arquitectura

G210

Problema: mensajes >10MB (imágenes, PDFs) no van bien en Pub/Sub.

Solución: guardar payload en GCS → publicar mensaje solo con la REFERENCIA (bucket, objeto, metadata) → consumidor descarga y procesa.

Beneficios: mensajería liviana y rápida, retry barato, lifecycle del payload separado del mensaje.

Nombre clásico: Claim Check (Enterprise Integration Patterns).

AVANZADO

GCP

RESPUESTA

Escenarios de entrevista

G209

Límite por API key/usuario: contador distribuido en Memorystore (Redis) — pero Redis por región ≠ global consistente.

Opciones: Redis global vía replicas + read local (aproximado), contador en Spanner (consistente, costo), o rate limit por-región (suficiente en la práctica).

En el borde: Cloud Armor rate-based por IP (grueso) + fino por identidad en la app.

Documentar el trade-off de precisión.

AVANZADO

GCP

RESPUESTA

Escenarios de entrevista

G212

Traduce a dinero: 'Cloud Run escala a 0 → pagamos uso real vs \$X/mes de VMs siempre encendidas; ahorro estimado Y%'.

TCA: infraestructura + operación (horas de DevOps ahorradas por managed) + riesgo (SLA/fallas evitadas).

Costo por unidad de negocio: \$ por pedido procesado/por usuario activo — la métrica que entiende todos.

CUDs y compromisos = descuento contra estabilidad de demanda.

AVANZADO

GCP

RESPUESTA

Patrones de arquitectura

G211

Aun con outbox/idempotencia, los sistemas divergen (bugs, mensajes perdidos).

Job de reconciliación: compara conteos/sumas entre sistemas (pedidos vs pagos del día), reporta diferencias, genera work-queue de reparación.

Métrica: % de records divergentes por día — tendiente a 0.

Es humilde pero es lo que mantiene la confianza en event-driven.

AVANZADO