GCP Certification Enhanced Cheatsheet

Comprehensive Reference

Professional Certification Guide

✓ Complete exam coverage
✓ Structured learning path
✓ Real-world scenarios
✓ Quick reference tables
✓ Best practices & tips

Version 1.0
Generated: December 22, 2025

Table of Contents

GCP Developer + DevOps — Enhanced Cheat Sheet

Ultimate rapid refresher for both exams with decision tables, formulas, and exam tactics.

■ Quick Decision Tables

Compute Service Selection

Need	Cloud Run	App Engine	GKE	Functions
HTTP API	■ Best	■ Good	■ Good	■■ Limited
Background jobs	■ Jobs	■■ Tasks	■ CronJobs	■ Best
Scale to zero	■ Yes	■ No (Std)	■ No	■ Yes
Stateful	■ No	■■ Limited	■ StatefulSet	■ No
WebSockets	■ Yes	■ Yes	■ Yes	■ No
Custom runtime	■ Container	■■ Limited	■ Any	■■ Runtimes
Cost (idle)	$ Free	$$ Always-on	$$$ Always-on	$ Free

Database Selection

Requirement	Cloud SQL	Firestore	Spanner	Memorystore
Relational SQL	■ Postgres/MySQL	■ NoSQL	■ Yes	■ Cache
Global scale	■ Regional	■ Multi-region	■ Global	■ Regional
Strong consistency	■ Yes	■■ Optional	■ Yes	■ Yes
Transactions	■ Full	■■ Limited	■ Full	■■ Limited
Max size	64 TB	Unlimited	Unlimited	300 GB
Cost	$ Low	$$ Medium	$$$ High	$ Low
Best for	Legacy apps	Mobile/web	Finance/ERP	Cache/sessions

SLO Error Budget Quick Reference

SLO	Downtime/30d	Downtime/year	When to use
99%	7.2 hours	3.65 days	Internal tools
99.5%	3.6 hours	1.83 days	Standard services
99.9%	43.2 min	8.76 hours	Business-critical
99.95%	21.6 min	4.38 hours	High-value SaaS
99.99%	4.3 min	52.6 minutes	Financial systems

■ Essential Commands

CI/CD

```
# Build & push
gcloud artifacts repositories create repo --repository-format=docker --
location=$REGION
gcloud builds submit --tag $REGION-docker.pkg.dev/$PROJECT/repo/app:latest

# Deploy Cloud Run
gcloud run deploy app --image=$IMAGE --region=$REGION --allow-unauthenticated

# Traffic splitting (canary)
gcloud run services update-traffic app --to-revisions REV1=90,REV2=10 --
region=$REGION

# Rollback
gcloud run services update-traffic app --to-latest --region=$REGION

# Pub/Sub with DLQ
gcloud pubsub subscriptions create sub \
  --topic=topic --dead-letter-topic=dlq --max-delivery-attempts=5

# Secret Manager
echo "mypassword" | gcloud secrets create db-password --data-file=-
gcloud run deploy app --update-secrets=DB_PASS=db-password:latest
```

Monitoring & Debugging

```
# View logs
gcloud logging read 'resource.type="cloud_run_revision" severity>=ERROR' --limit=50

# List metrics
gcloud monitoring time-series list --filter='metric.type="run.googleapis.com/
request_latencies"'

# Create alert policy
gcloud alpha monitoring policies create --policy-from-file=alert.yaml

# Cloud Trace
gcloud trace list --limit=10
```

■ Security Essentials

IAM Best Practices

```
# Create dedicated service account
gcloud iam service-accounts create app-sa

# Grant minimal permissions
gcloud projects add-iam-policy-binding $PROJECT \
  --member="serviceAccount:app-sa@$PROJECT.iam.gserviceaccount.com" \
  --role="roles/cloudsql.client"

# Use in Cloud Run
gcloud run deploy app --service-account=app-sa@$PROJECT.iam.gserviceaccount.com
```

Key Principles

- ■ Use Secret Manager (never env variables for secrets)
- ■ Service account per service (not default)
- ■ Workload Identity for GKE (no keys)
- ■ CMEK for compliance
- ■ VPC Service Controls for perimeter

■ SRE Formulas & Calculations

Error Budget

```
Error Budget = 1 - SLO

Example: 99.9% SLO
Error Budget = 1 - 0.999 = 0.001 = 0.1%

30-day window:
Allowed downtime = 43,200 min × 0.001 = 43.2 minutes
```

Burn Rate

```
Burn Rate = (Current Error Rate) / (SLO Error Rate)

Example: 99.9% SLO (0.1% error budget)
Current error rate: 1.4%
Burn rate = 1.4% / 0.1% = 14x

Time to exhaust = 30 days / 14 = 2.14 days
```

Multi-Window Alerts

Window	Burn Rate	Error Rate	Action
5 minutes	14x (1.4%)	1.4%	Page
1 hour	4x (0.4%)	0.4%	Page
6 hours	2x (0.2%)	0.2%	Ticket
3 days	1x (0.1%)	0.1%	Monitor

Common Troubleshooting

Symptom → Root Cause → Quick Fix

503 Service Unavailable:

- Concurrency too high → Reduce concurrency
- Not enough instances → Increase max-instances
- Cold starts → Set min-instances

429 Too Many Requests:

- Rate limiting → Add backoff/retry
- Quota exceeded → Request increase

502 Bad Gateway:

- Upstream timeout → Increase timeout
- Backend unhealthy → Check health endpoint

Out of Memory:

- Memory leak → Profile with Cloud Profiler
- Undersized → Increase memory limit

Slow Queries:

- Missing index → Add index
- N+1 problem → Use joins/eager loading
- Large result set → Add pagination

Cloud Run Cold Starts:

- Set min-instances=1 for latency-critical
- Reduce image size (multi-stage builds)
- Keep warm with synthetic requests

Pub/Sub Backlog:

- Scale consumers horizontally
- Optimize processing time
- Check for slow dependencies
- Add partitions (increase parallelism)

■ Cost Optimization

Comparison (10K req/day API)

```
Cloud Run (scale-to-zero):
Monthly: ~$8-12 ■ Winner (saves 70-90%)

App Engine Standard (1 instance):
Monthly: ~$36

GKE (e2-medium node):
Monthly: ~$97 (node) + $73 (cluster) = $170
```

Cost Checklist

- [ ] Use scale-to-zero (Cloud Run, Functions)
- [ ] Set min-instances only where latency critical
- [ ] Enable Cloud CDN for public content
- [ ] Add Memorystore caching for hot data
- [ ] Lifecycle policies for storage (Nearline/Coldline)
- [ ] Committed use discounts (57% off for 1-year)
- [ ] Delete unused resources regularly
- [ ] Budget alerts at 50%, 90%, 100%

■ Exam Strategy

Question Keywords → Answer Hints

Keyword	Likely Answer
"Cost-effective"	Cloud Run > App Engine > GKE
"Minimal changes"	Lift-and-shift: App Engine or GKE
"Cloud-native"	Cloud Run, Functions, managed services
"Existing Kubernetes"	GKE (import manifests)
"Low latency"	Set min-instances, use Memorystore
"Secure"	Secret Manager, private endpoints, least privilege
"Compliance"	Organization policies, CMEK, VPC-SC
"Global scale"	Spanner, Firestore, multi-region GCS
"Real-time"	Pub/Sub, Firestore, streaming
"Batch processing"	Cloud Run Jobs, GKE CronJobs

Red Flags (Wrong Answers)

■ Over-engineering: "Use GKE for simple API"

■ Ignoring managed services: "Build your own X"

■ Security holes: "Store keys in code/env vars"

■ No error handling: "Just deploy and hope"

■ Ignoring cost: "Always use biggest machines"

■ No observability: "Check logs manually"

Decision Framework

```
1. Identify constraint:
   - Cost → Serverless (Run/Functions)
   - Control → GKE
   - Simplicity → Managed services
   - Legacy → App Engine or lift-and-shift

2. Match workload:
   - HTTP API → Cloud Run
   - Events → Functions
   - Batch → Jobs
   - Stateful → GKE

3. Apply best practices:
   - Security: Secret Manager + IAM
   - Reliability: Multi-zone + retries
   - Observability: Logging + Monitoring + Traces
   - Cost: Autoscaling + lifecycle policies
```

Time Management (2 hours, 50-60 questions)

- **0-40 min:** First pass - easy questions (~35-40)
- **40-80 min:** Second pass - difficult questions (~15-20)
- **80-120 min:** Review all, check for mistakes
- **2 min average per question**
- **Mark unsure questions for review**

Last-Minute Tips

■ Read ENTIRE question before answering

■ Eliminate obviously wrong answers first

■ Look for "MOST" (one best answer)

■ Trust first instinct (unless clear error)

■ No penalty for guessing - answer all

■ Watch double negatives

■ Pre-Exam Checklist

Day Before:

- [ ] Review this cheat sheet (30 min)
- [ ] Skim high-frequency topics
- [ ] Review sample questions
- [ ] Get 8 hours sleep

Morning Of:

- [ ] Eat breakfast
- [ ] Test internet/webcam (online)
- [ ] Have water nearby
- [ ] Clear desk (no notes)
- [ ] Arrive/log in 15 min early

During Exam:

- [ ] Read twice
- [ ] Eliminate wrong answers
- [ ] Mark difficult
- [ ] Manage time
- [ ] Review marked
- [ ] Stay calm!

■ Most Commonly Tested

1. **Service selection** - Requirements → compute service
2. **Database selection** - Data model + scale → database
3. **CI/CD pipeline** - Build/test/deploy flow
4. **Traffic splitting** - Canary or blue/green
5. **Secret management** - Secure credentials
6. **Pub/Sub patterns** - Idempotency, DLQ, ordering
7. **Error budgets** - Calculate downtime from SLO
8. **Burn-rate alerts** - Multi-window alerting
9. **Incident response** - Troubleshoot failures
10. **Cost optimization** - Reduce spend, meet SLOs

■ Quick Reference

Pub/Sub Guarantees

- **At-least-once delivery** (implement idempotency!)
- **Ordering keys** (same key → same partition → order preserved)
- **DLQ** (poison messages after max attempts)

Cloud Run Limits

- Max timeout: 60 min
- Max concurrency: 1000 (default 80)
- Max instances: 1000 (default 100)
- Max memory: 32 GB
- Max CPU: 8 vCPU

GKE Best Practices

- Use Workload Identity (not service account keys)
- Binary Authorization for image verification
- Network policies for pod-to-pod security
- HPA for pod autoscaling
- Cluster Autoscaler for node autoscaling
- Regional clusters for HA (3 zones)

Observability Stack

- **Logging:** Structured JSON logs, log-based metrics
- **Monitoring:** SLO/SLI/alerts, uptime checks
- **Trace:** Distributed tracing, latency analysis
- **Profiler:** CPU/memory hotspots
- **Debugger:** Live debugging, snapshots
- **Error Reporting:** Automatic error aggregation

Good luck! ■