SREference

SLO design cheatsheet

Write it down

```
SLO: [service], [user journey]
SLI: good = [e.g. non-5xx in under 300 ms]
     valid = [e.g. all requests except auth failures]
     measured at [load balancer / probe / client]
Target: [99.9]% of valid events over a rolling [28]
days
Why: [what users need; the data the target came from]
Alerts: multiwindow burn-rate rules; page [team]
Policy: [link to error budget policy]
Owners: [engineering lead], [product lead]
Review: [quarterly]; last reviewed [date]
```

SLI vs SLO vs SLA → sreference.com/sli-slo-sla/

Pick the SLI

```
SLI = good events ÷ valid events
```

TYPE	GOOD EVENT	FITS
Availability	Served without a server-side error	APIs, web
Latency	Served faster than a threshold	Interactive
Freshness	Data updated within a threshold	Pipelines, caches
Correctness	Output matches the expected result	Billing, search

- Express latency as the share of requests faster than X, not as a percentile.
- Combine instances by summing good and valid events, then dividing. Never average ratios or percentiles.
- Measure as close to users as you can afford: load balancer before application metrics, probes for critical journeys.
- **Few events?** One failure can page. Use a binomial-test threshold, a minimum request count, or synthetic traffic.

Latency percentiles → sreference.com/latency-percentiles/

Set the target

- **Never 100%.** It is unachievable and leaves no budget for change.
- **Start from history, then ask what users need.** Round four weeks of data down; history is a starting point, not the target. Start loose and tighten.
- **Check dependencies.** 3 hard dependencies at 99.95% cap you at about 99.85% if they fail independently, before your own failures. Redundancy or graceful degradation raises it.
- **Keep an SLA looser** than the SLO, so the SLO warns you first.
- **Just enough SLOs** to cover what users care about; one to three per journey is a starting point. Drop any whose miss would change no decision.

Production readiness review → sreference.com/production-readiness-review/

Budget per target

SLO	BUDGET	28 DAYS	30 DAYS	PER 1M
99%	1%	6h 43m 12s	7h 12m	10,000
99.5%	0.5%	3h 21m 36s	3h 36m	5,000
99.9%	0.1%	40m 19s	43m 12s	1,000
99.95%	0.05%	20m 10s	21m 36s	500
99.99%	0.01%	4m 2s	4m 19s	100

Times are full outage at steady traffic. “Per 1M” is failed requests allowed per million; a request-based SLO counts those, not minutes.

Availability table → sreference.com/availability-table/

Window and alerts

- **Rolling four weeks** (28 days) is a good default: rolling matches user experience; whole weeks keep weekends constant. Calendar months suit reporting.

```
burn rate = error rate ÷ (1 - SLO)
```

ACTION	SPENT	WINDOWS	BURN	30 D
Page	2%	1h / 5m	13.44×	14.4×
Page	5%	6h / 30m	5.6×	6×
Ticket	10%	3d / 6h	0.9333×	1×

- Burn for 28 days; “30 d” is the Workbook’s 30-day figure. Fire when both windows exceed burn × (1 − SLO).
- For a latency SLO, use its own budget: at 99%, a 13.44× page fires at 13.44% slow requests, not 1.344%.
- Below 92.55%, the 13.44× page can never fire; rely on the slower alerts.

Burn-rate alerting → sreference.com/burn-rate-alerting/

Policy and review

BUDGET SPENT	RESPONSE
50%	Review the biggest consumers; prefer low-risk changes
75%	Extra review for risky changes; reliability work moves up
100%	Freeze changes, except P0 and security fixes, until back within the SLO

- One incident spending over 20% of the budget gets a postmortem with at least one top-priority action item.
- **Review regularly.** SLO met but users unhappy: tighten it, or fix the SLI if it misses what users see. SLO missed but users happy: loosen it.

Error budget policy → sreference.com/error-budget-policy/

Sources: Google, Site Reliability Engineering, 1st ed., ch. 4 “Service Level Objectives” and 2nd ed., ch. 8; The Site Reliability Workbook, ch. 2 “Implementing SLOs”, ch. 5 “Alerting on SLOs” and appendix B. Alert thresholds and the graduated policy steps are starting points, not standards.