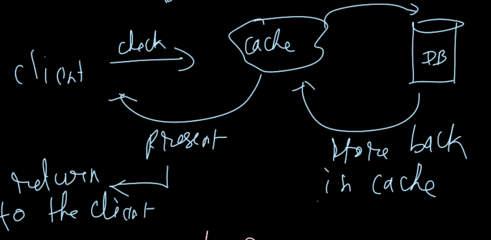
Cache → Redis

Use-case	why Redis
caching	super fast
Queues/pub-sub	High throughput
Distributed locks	Safe concurrency

#caching patterns & Expiry

① cache - Aside

the application first check the cache, and only if missing, it fetches from db and store in cache.


client
check
cache
DB
present
store back in cache
return to the client

why its used ?

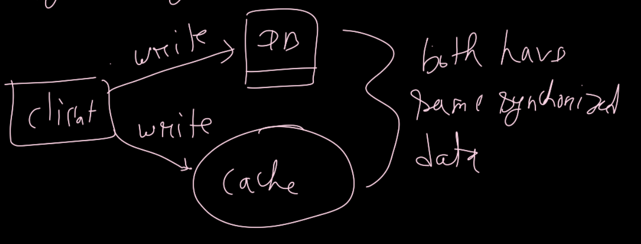
* DB and cache consistency is controlled by the application

* No unnecessary caching (only frequently accessed data)

② write-through - strong consistency

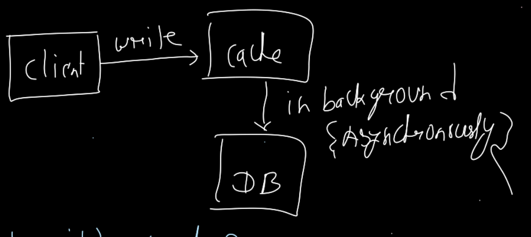
↳ every write operation goes to both DB and cache at same time

↳ Data is always updated everywhere synchronously


client
write
DB
write
cache
both have same synchronized data

③ write-Behind (write-back) :- high performance

↳ write only to cache first, and write to DB asynchronously


client
write
cache
in background {Asynchronously}
DB

why it's used ?

* super fast writes

* writes can be batched → Huge DB savings

* very good for high-write workloads

problems:

problem	why	Fix
Data loss risk	cache may crash before flush	AOF persistency/ replication
Harder implementation	Need async job processor	use Redis streams kafka, etc.
DB lag	writes delayed	monitoring + max delayed timeout

*→ where's it used ?

*→ logging systems

*→ metrics/analytics dashboard

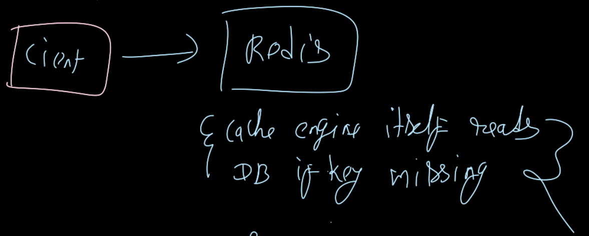
*→ Game score updates

impor

④ Read-through - cache load data automatically

↳ Application never calls DB directly.

↳ If Data is missing, cache provider fetches from DB automatically and returns the result.


Client
Redis
{cache engine itself reads DB if key missing}

where's its used ?

* CDN and managed caching systems

* AWS Elasticache / memcached auto-loader

* most enterprise platforms