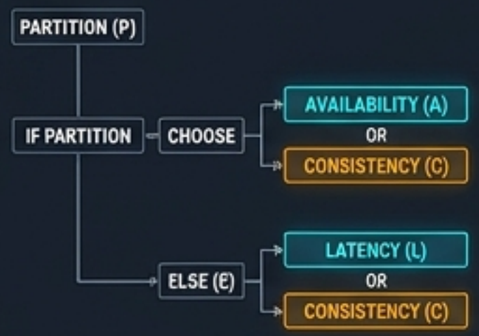
Modern Data Stores — S05E06

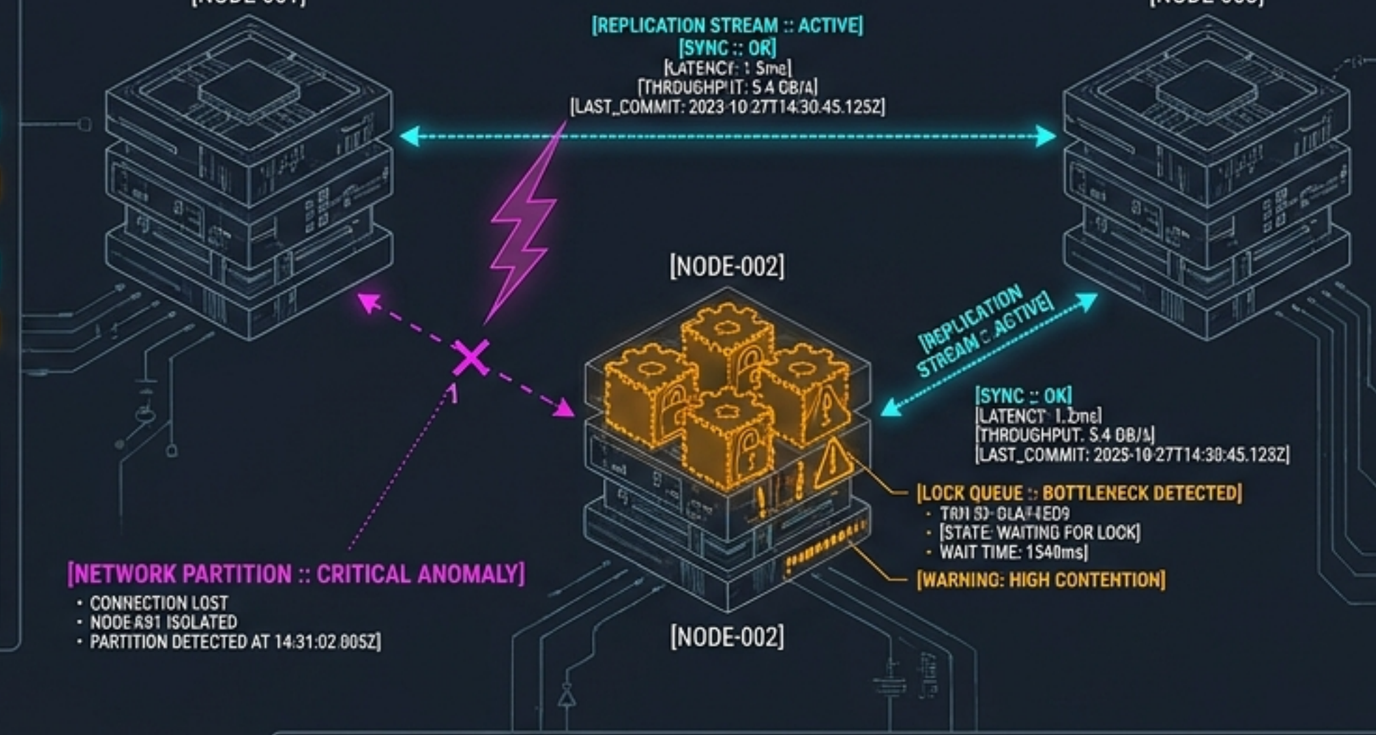
Consistency (PACELC & Jepsen), Failover Fencing, PITR Backups, Capacity & Zero-Downtime Migrations.

CONSISTENCY (PACELC & JEPSEN)



JEPSEN TEST RESULTS

TEST_RUN	J-2023-Q4
LINEARIZABILITY	PASS
SERIALIZABILITY	PASS
SNAPSHOT ISOLATION	PASS
EVENTUAL CONSISTENCY	WARN - LATENCY SPIKE
PARTITION TOLERANCE	FAIL - SPLIT BRAIN SCENARIO



FAILOVER FENCING & RECOVERY



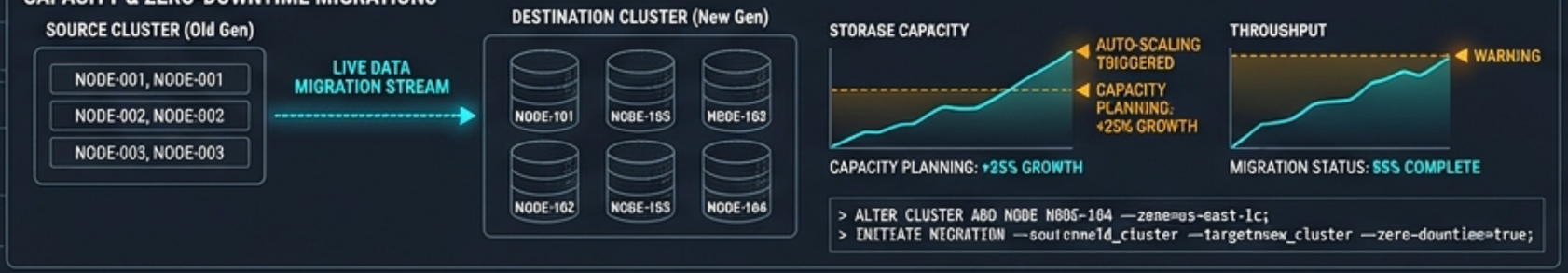
- STONITH (Shoot The Other Node In The Head) ENABLED
- LEASE EXPIRED
- NEW LEASE ACQUIRED BY NODE-003]

PITR BACKUPS (POINT-IN-TIME RECOVERY)



- RESTORE TO TIMESTAMP: 2023-10-27T14:00:00Z]
- DATA RECOVERY IN PROGRESS
- TARGET RPO: <5m
- TARGET RTO: <15m

CAPACITY & ZERO-DOWNTIME MIGRATIONS



Gemini Notebook

THE DAY-2 STATEFUL REALITY

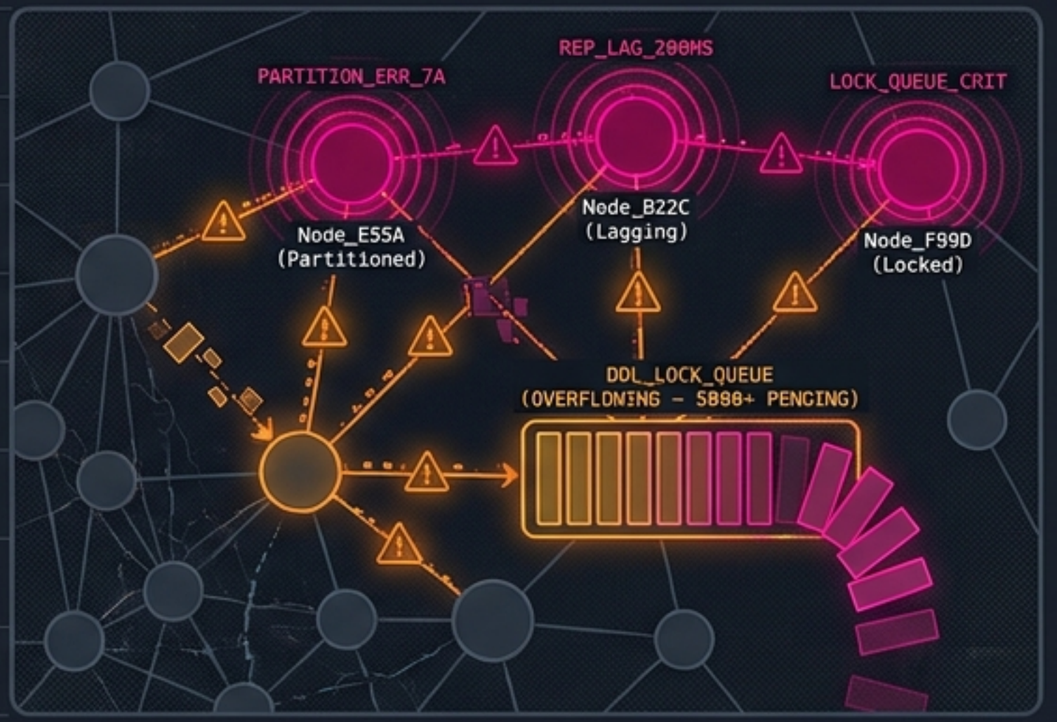
Production readiness isn’t about avoiding failure; it’s about bounding the blast radius when failure inevitably occurs.

IDEALIZED FAILURE MODEL

```
SELECT *
FROM users
WHERE active_status = 'TRUE'
AND subscription_type = 'premium'
ORDER BY signup_date DESC
LIMIT 100;
```

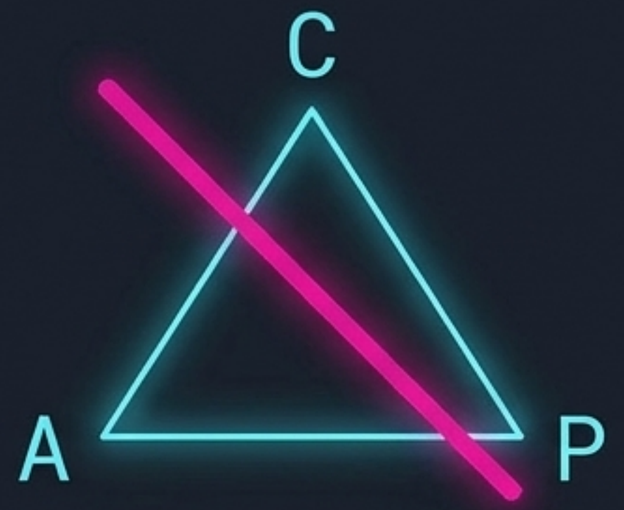
What we think causes outages:
Bad SQL syntax.

REALISTIC FAILURE MODEL



What actually causes outages: Network partitions, replication lag, connection saturation, untested restores, and DDL lock queues.

Gemini Notebook



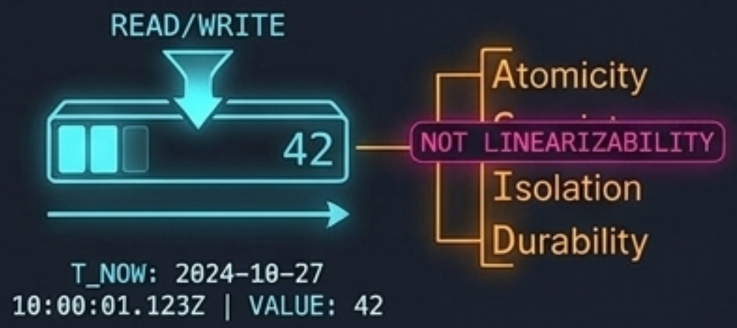
Why the ‘Pick 2 of 3’ CAP Theorem Misleads Engineers

Stop categorizing databases as “CP” or “AP”. Most modern datastores fit neither strict definition.

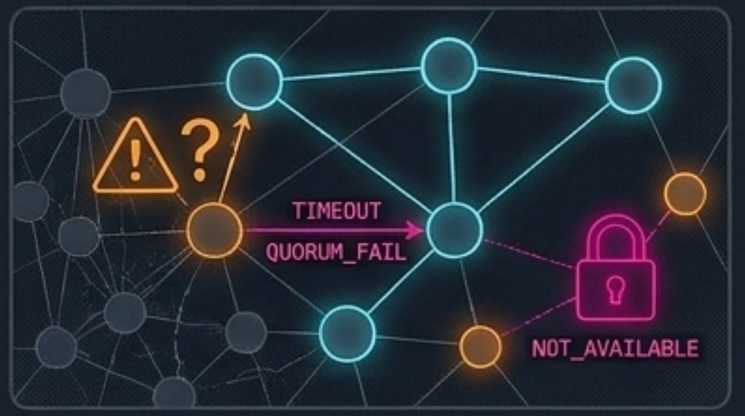
[P] Partitions: A reality of asynchronous networks, not a choice. (Networks delay/drop messages).



[C] Consistency: Strictly means Linearizability (single-register, real-time order). It has nothing to do with the ‘C’ in ACID.



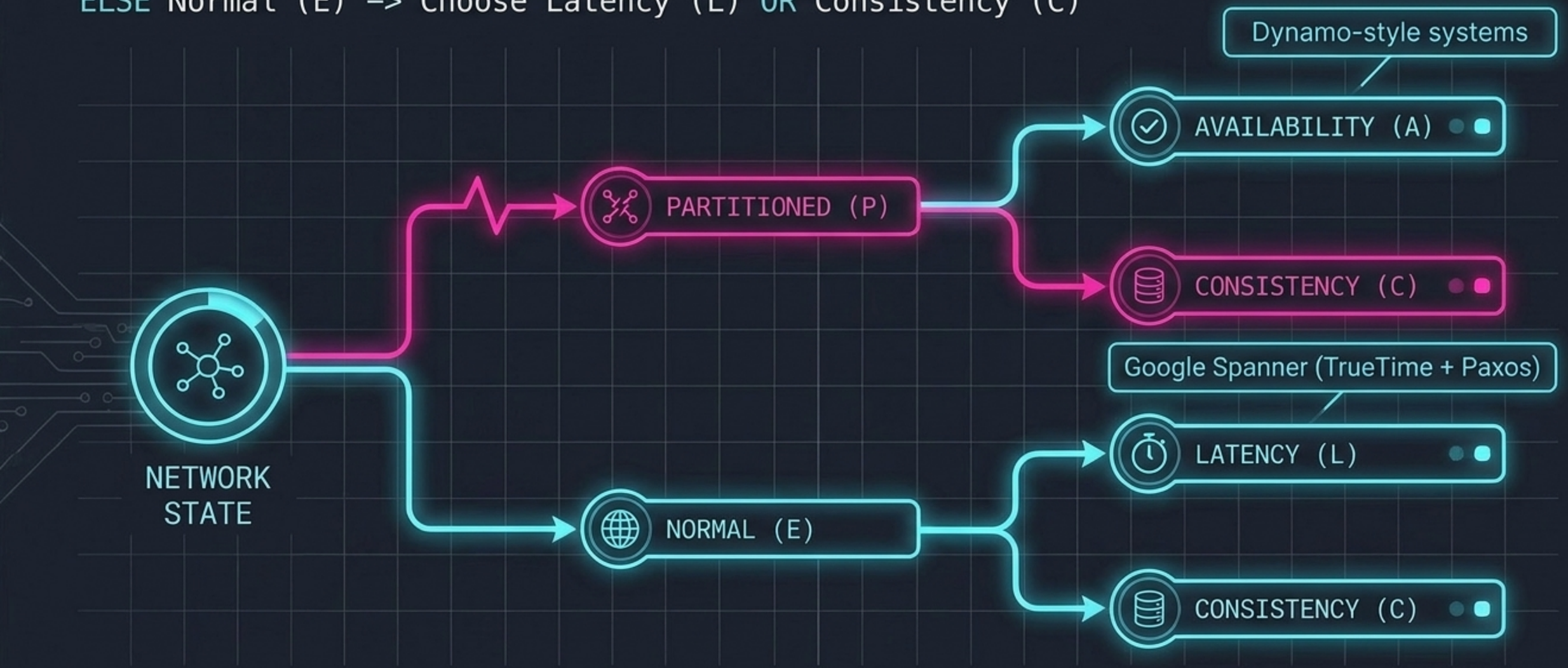
[A] Availability: Every non-failing node must respond. Quorum systems (where a minority partition cannot reach consensus) violate this.



Gemini Notebook

THE OPERATIONAL REALITY: THE PACELC FRAMEWORK

IF Partition (P) -> Choose Availability (A) OR Consistency (C)
ELSE Normal (E) -> Choose Latency (L) OR Consistency (C)



Gemini Notebook

Jepsen Models: The Consistency Topography

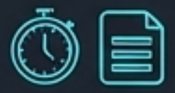
Single-Object / Real-Time

Multi-Object / Transactional

Y-Axis
Order (Transactional Total Order vs.
vs. Real-Time Total Order)

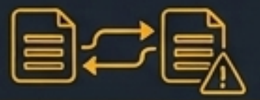
Linearizability

(Every operation takes place atomically, consistent with real-time clocks)



Serializability

(Operations act atomically across multiple objects, but no real-time guarantee. Reordering is legally allowed)



Snapshot Isolation

(No total order; allows interleaving sub-operations)



Strict Serializability

(The holy grail:
Spanner / TrueTime + Paxos)

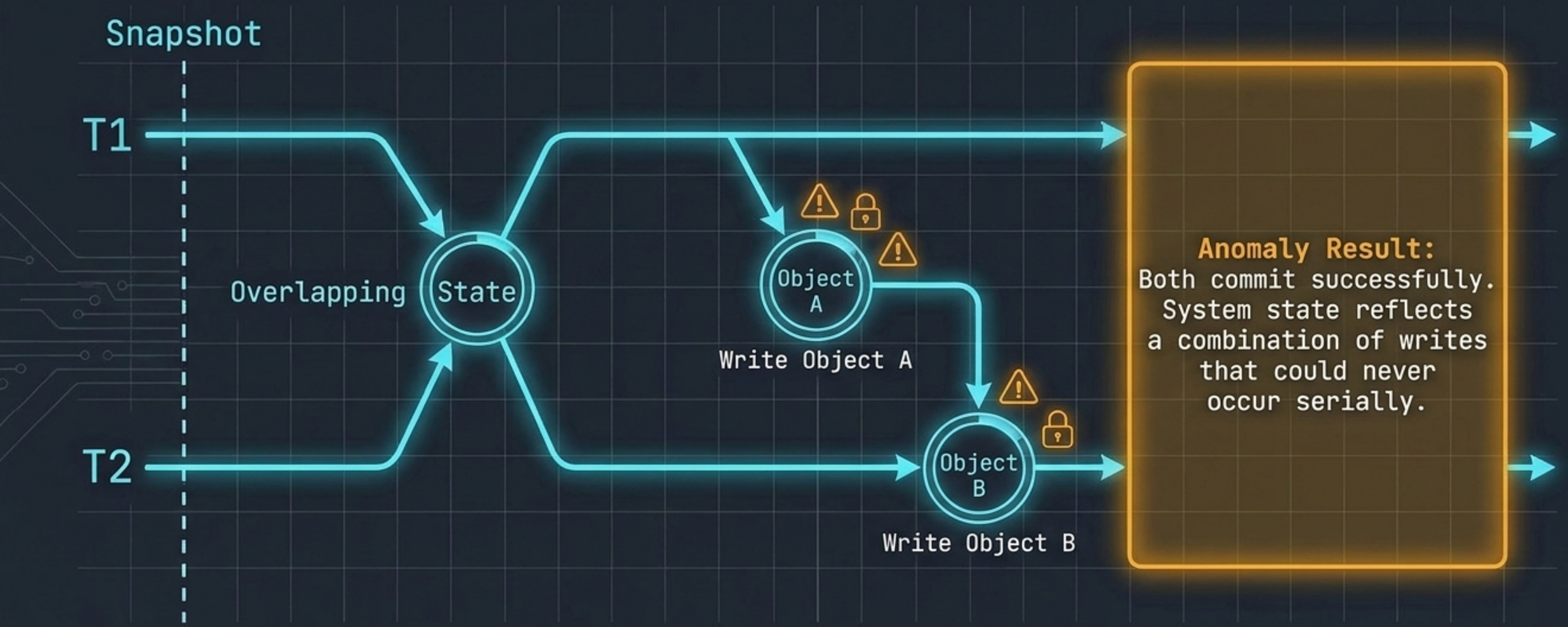


X-Axis
Scope (Single-Object vs. Multi-Object)

Gemini Notebook

Snapshot Isolation & The Write-Skew Anomaly (G2-item)

PostgreSQL's REPEATABLE READ is actually Snapshot Isolation. Historically, its SERIALIZABLE mode allowed G2-item anomalies due to a conflict detection bug.



Gemini Notebook

Replication & Quorum Trade-Off Matrix

Configuration	Write RTT	Stale Reads?	RPO (Data Loss Window)
PG remote_write	1 RTT (OS buffer)	Yes	OS crash window
PG remote_apply	1 RTT + Replay	No (Causal consistency)	Zero
Raft/Paxos (Spanner)	Quorum RTT	No (Strong Leader)	Zero (Leader completeness property)
Dynamo Sloppy Quorum	Configurable (W=1 vs Quorum)	Yes	Windowed data loss

Gemini Notebook

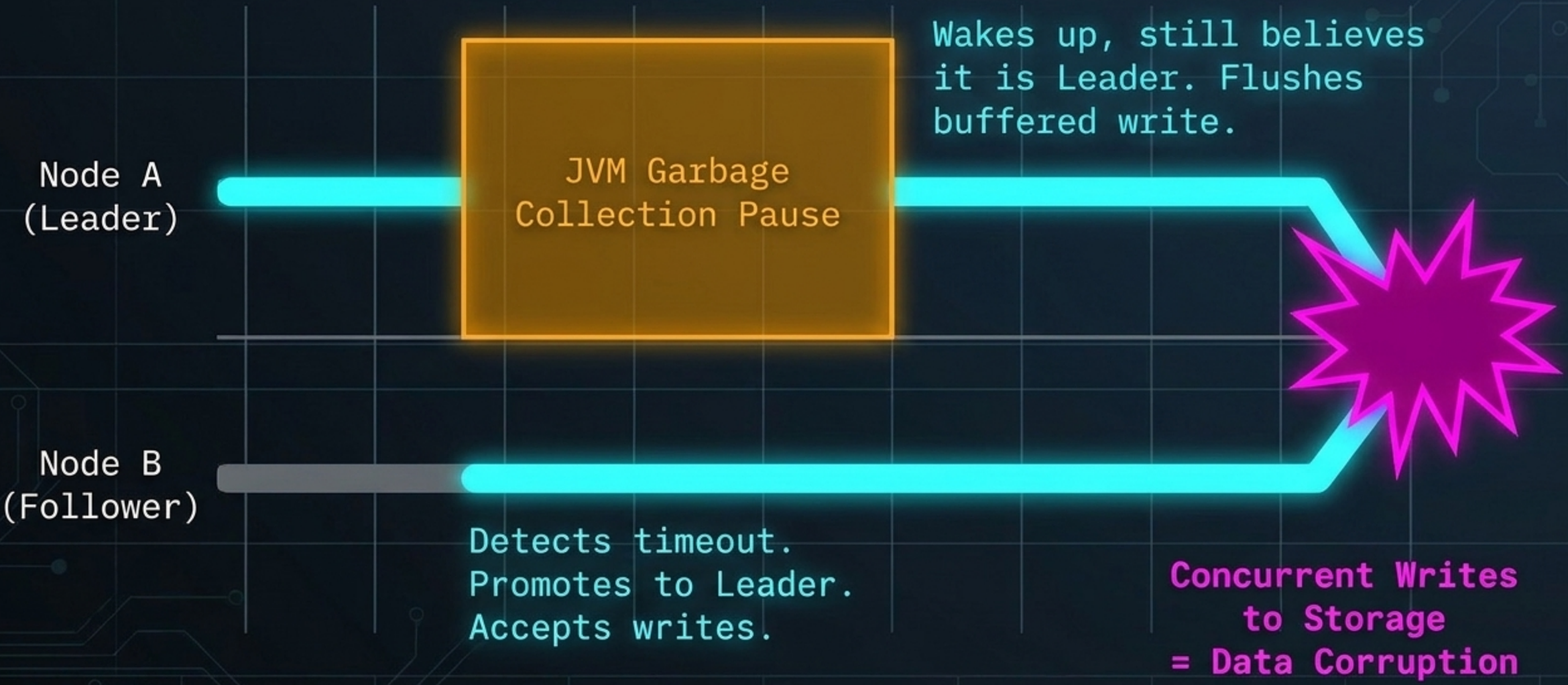
Backup & Restore Matrices (The PITR Chimera)

"A backup and recovery solution that provides PITR… while meeting strict uptime, latency, and cost goals is a chimera today!" – Google SRE

Mechanism	RTO (Downtime)	RPO (Data Loss)	Survives Human Error (DROP TABLE)
Replicas	Seconds	Zero	 Fails - replicates the DROP instantly
Snapshots (EBS)	Minutes	Hours	 Survives, but loses hours of data
pg_dump (Logical)	Hours	Days	 Survives, but extremely slow to restore
Base Backup + WAL (PITR)	Variable	Sub-minute	 Rewind to exactly 5:14 PM before the drop

Gemini Notebook

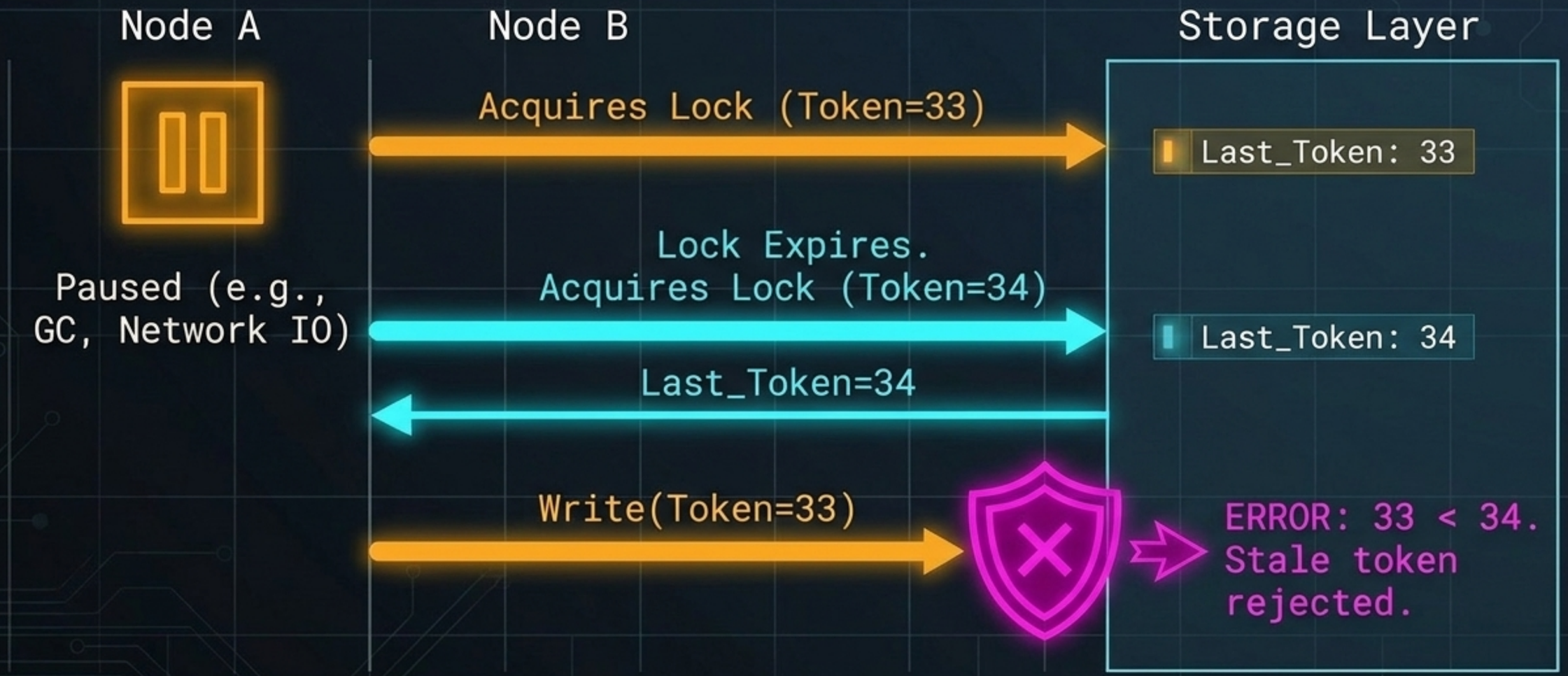
The Anatomy of a Split-Brain



Gemini Notebook

Monotonic Fencing Tokens (Kleppmann)

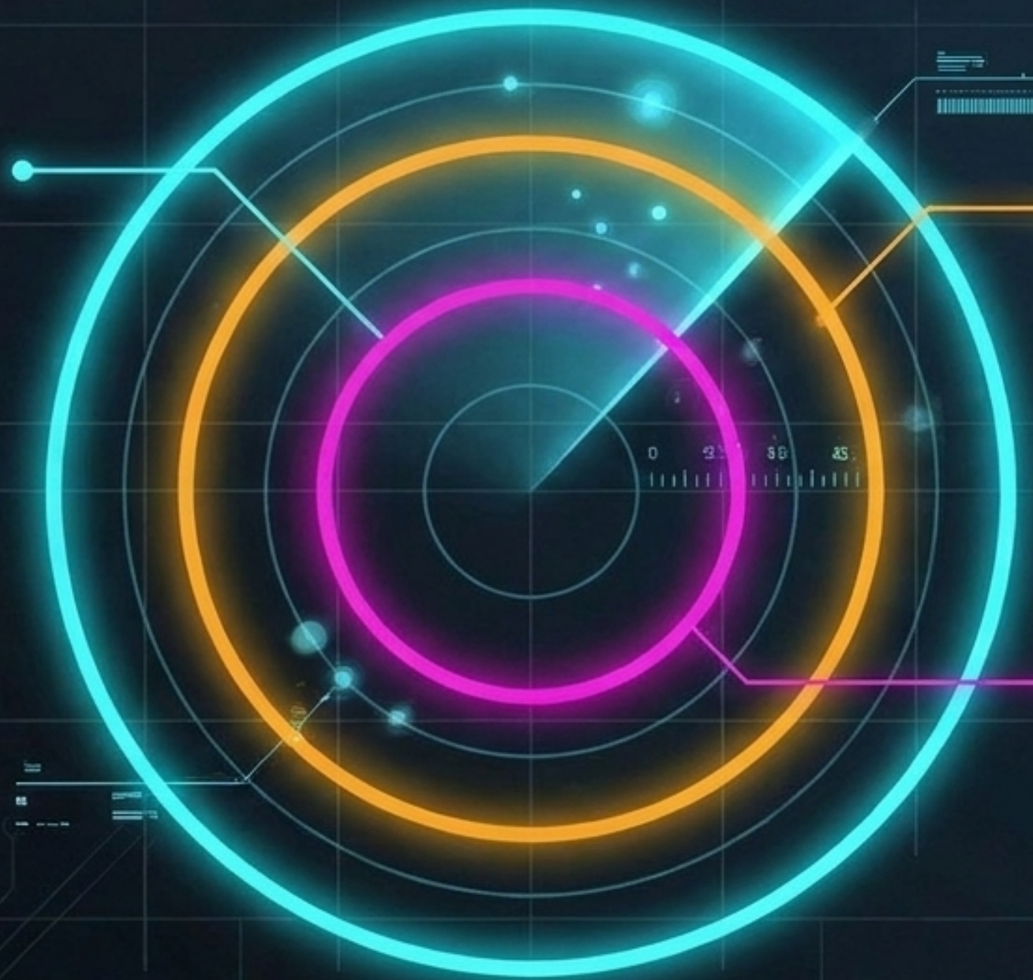
"Fencing must happen at the storage layer, not just the application layer. Consensus algorithms ensure tokens monotonically increase."



Gemini Notebook

Data Integrity Defense-in-Depth

"A backup system without automated, tested restores is just a trash can with a receipt."



Soft Deletes
(Application logic masking records rather than dropping rows)

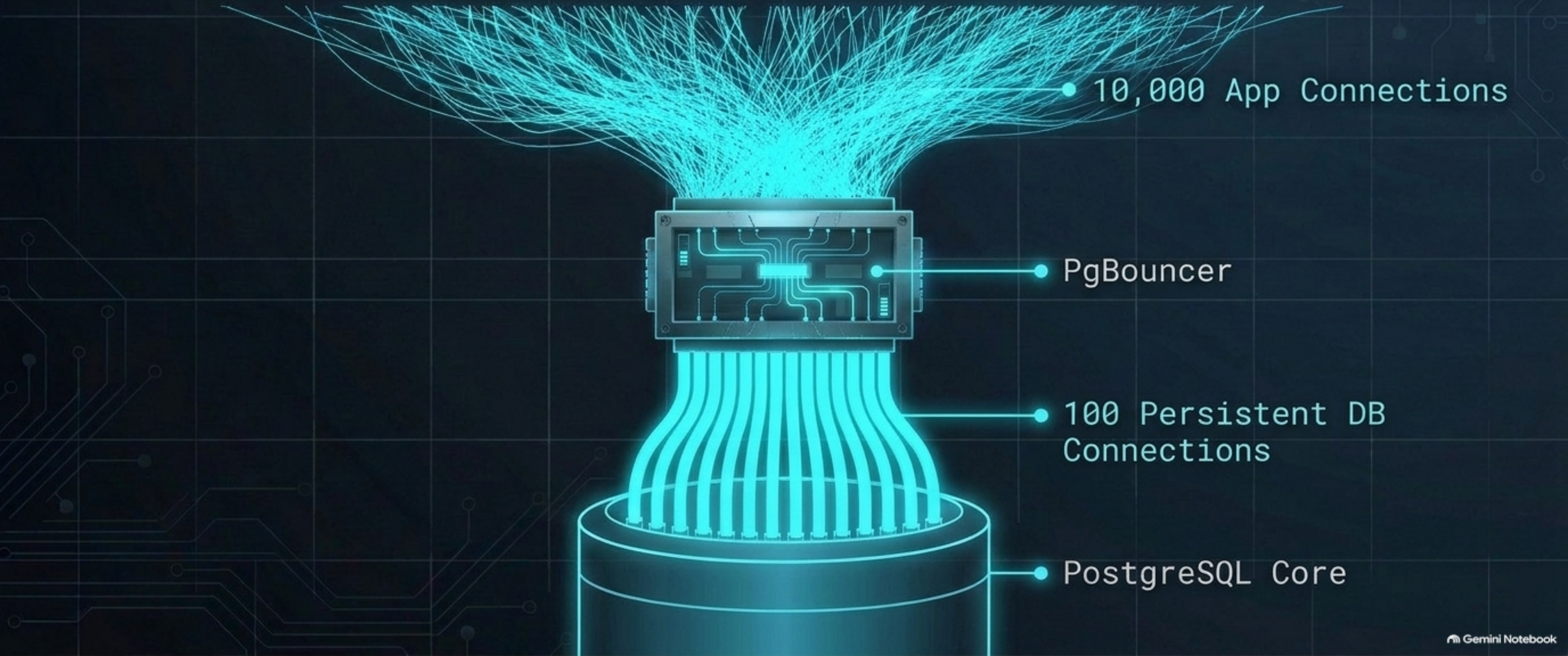
Block Checksums
(Storage layer detection of bit-rot and disk corruption)

Automated Restore Verification
(Continuous PITR drills validating recovery into isolated staging environments)

Gemini Notebook

Multiplexing & Connection Pooling

“Postgres processes are memory-heavy. Multiplexing turns thousands of idle app connections into a few high-throughput pipelines.”



Gemini Notebook

Diagnosing the DB via pg_stat_statements

"Shared-buffer misses force disk reads. This is the root cause of tail latency and cloud IOPS billing spikes."

```
user@database:~$
SELECT query, calls, total_exec_time, rows,
       100.0 * shared_blks_hit / nullif(shared_blks_hit + shared_blks_read, 0) AS hit_percent
  FROM pg_stat_statements
  ORDER BY total_exec_time DESC LIMIT 5;

                  query                    | calls | total_exec_time | rows | hit_percent
-------------------------------------------+-------+-----------------+------+-------------
  SELECT * FROM huge_table WHERE id > 5000... |  700 |      56789.123 | 100000 | 82.413201
  SELECT * FROM huge_table WHERE id > 5000... |  200 |      56789.123 | 100000 | 82.413201
  SELECT * FROM huge_table WHERE id > 5000... | 1200 |      56789.123 | 100000 | 82.413201
  SELECT * FROM huge_table                    |   20 |      36789.123 |   5000 | 10.413201
  SELECT * FROM huge_table                    |    7 |       5789.123 |     10 |  3.373201
  SELECT * FROM huge_table                    |    1 |        189.123 |      0 |  3.567201
(1 rows)

user@database:~$
```

WARNING: < 99% means query is thrashing the disk cache, causing significant I/O.

Gemini Notebook

Surviving Retry Storms & Cold Caches

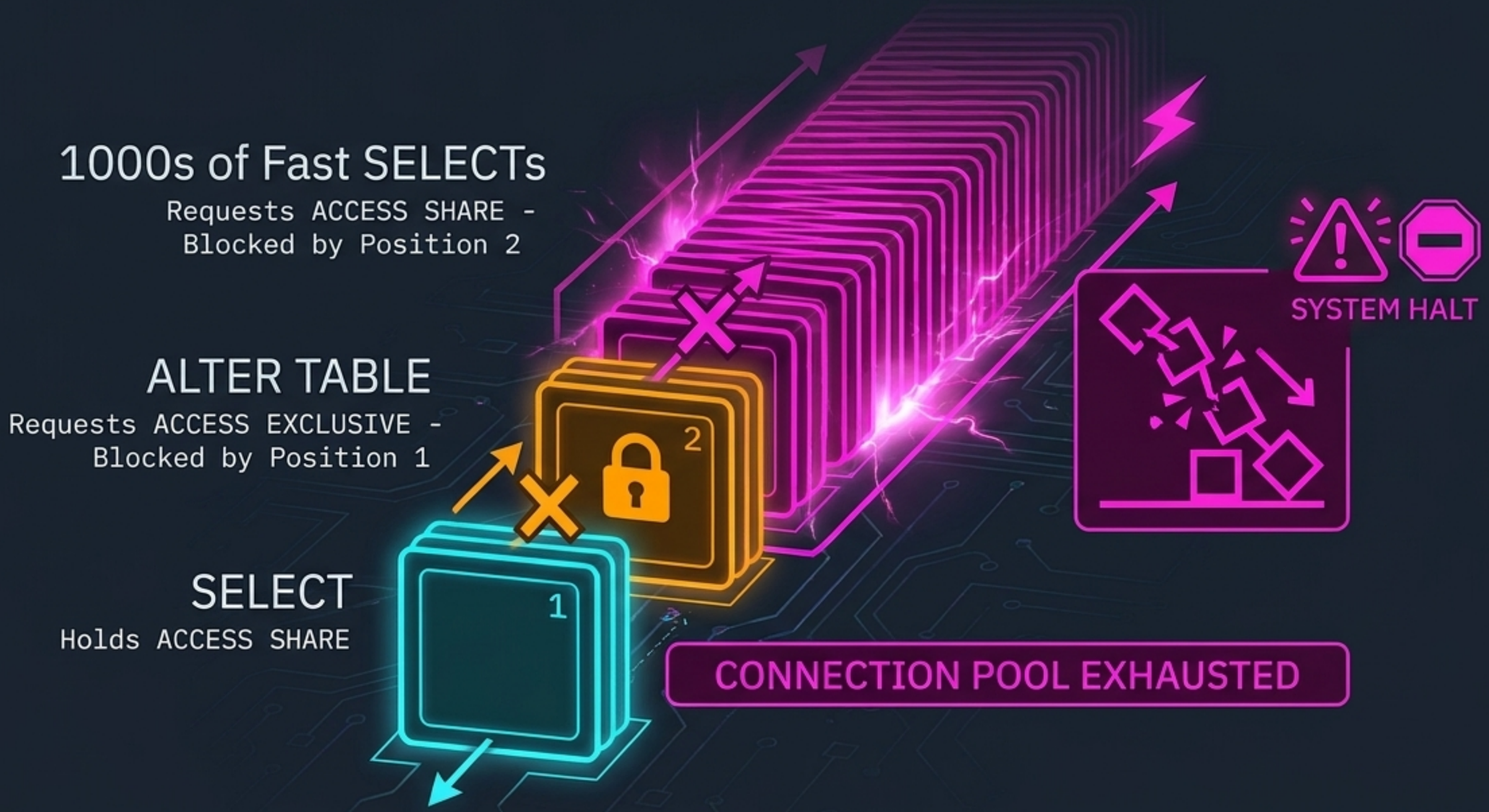




Gemini Notebook

The ACCESS EXCLUSIVE Queue of Death

The lock queue backs up instantly, exhausting the entire connection pool.



Gemini Notebook

Safe Migrations Playbook

Never execute DDL in production without a lock_timeout.

Danger

```
-- Blocks all reads/writes

CREATE INDEX
idx_users_email
ON users(email);
```

Safe

```
-- Fails fast if blocked,
builds in background

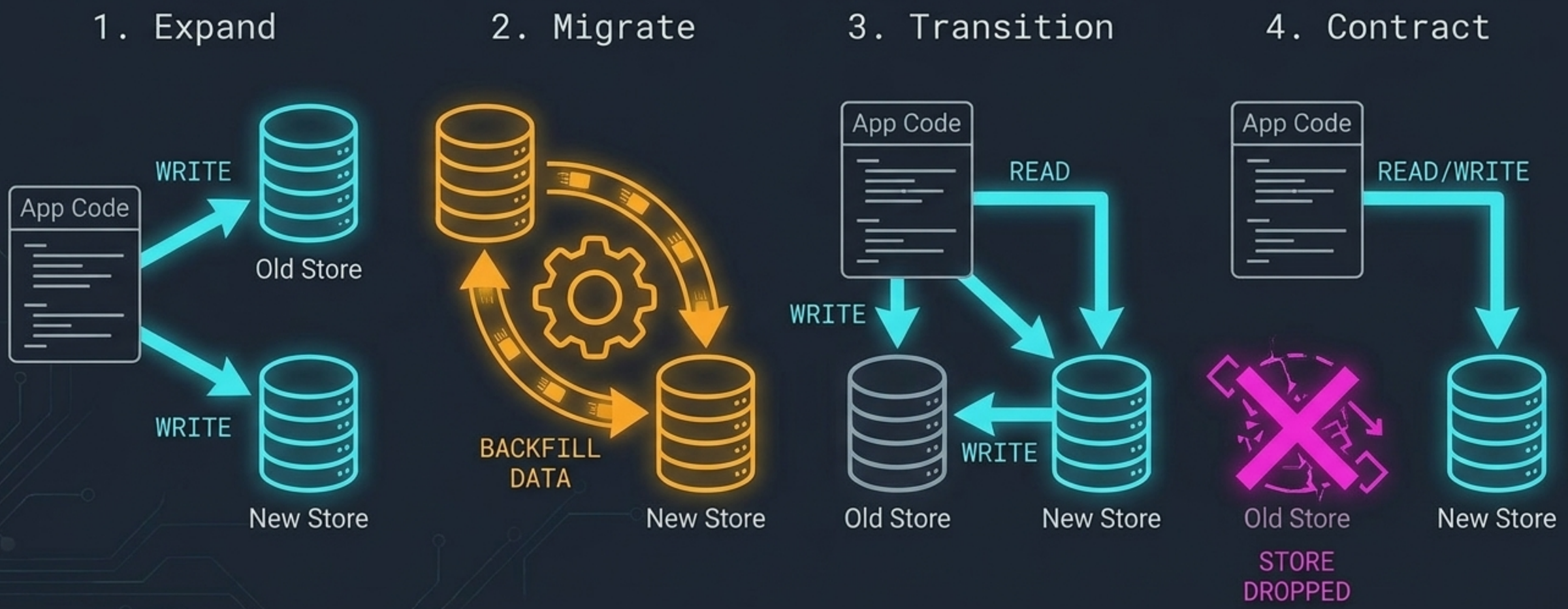
SET lock_timeout = '2s';

CREATE INDEX CONCURRENTLY
idx_users_email
ON users(email);
```

Gemini Notebook

Evolutionary DB Design: Expand and Contract

The 4-phase pattern for migrating columns or entire data stores live, with zero downtime.



Gemini Notebook

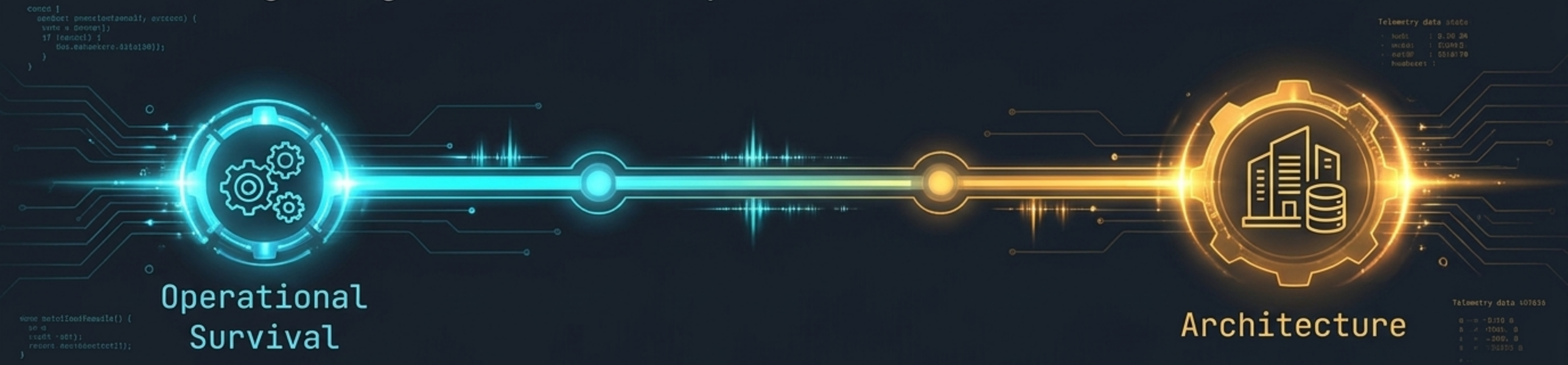
The Pre-Launch Stateful Checklist

- [ON] **RPO/RTO Defined:** Tested via automated PITR restore scripts.
- [ON] **Connection Multiplexing:** PgBouncer deployed with hard maximum limits.
- [ON] **Defensive DDL:** lock_timeout strictly enforced globally in all CI/CD migration runners.
- [ON] **Telemetry Active:** pg_stat_statements hit ratios explicitly monitored (>99% SLA).
- [ON] **Edge Protection:** Load shedding and out-of-memory (OOM) limits configured at the API gateway.

Gemini Notebook

Conclusion & Next Up

Consistency guarantees and robust failovers matter little if a basic schema change brings down the site. Operational survival dictates architecture.



Next Up: S05E07 Finale – Choosing a Data Store. Putting everything together to pick the right tool for the job.

Gemini Notebook