1. How do you design a database for 1B+ records?

Estimate growth, use partitioning, optimize indexing strategy, archive cold data, use read replicas, and avoid cross-shard joins.

---

2. How do you decide between normalization and denormalization?

Normalization improves consistency; denormalization improves read performance. Decision depends on workload and query patterns.

---

3. What are partitioning strategies?

Range, Hash, List, and Composite partitioning based on access patterns.

---

4. Difference between sharding and partitioning?

Partitioning occurs within one DB; sharding distributes data across multiple DB servers.

---

5. How do you choose a shard key?

High cardinality, even distribution, frequently used in queries.

---

6. Why are queries slow even with indexes?

Low selectivity, functions in WHERE clause, wrong join order, outdated stats.

7. What is a covering index?

An index containing all columns needed for a query, avoiding table lookups.

8. What is index selectivity?

Ratio of unique values to total rows. Higher selectivity improves performance.

9. How do you troubleshoot slow queries?

Analyze execution plan, check scans, verify indexes, examine locks and resource usage.

10. Common indexing mistakes?

Over-indexing, indexing low-cardinality columns, missing foreign key indexes.

11. Impact of isolation levels?

Higher isolation increases locking and reduces concurrency.

12. How to prevent deadlocks?

Keep transactions short, consistent locking order, retry logic.

13. What is write amplification?

One logical write triggers multiple physical writes (data, index, WAL, replication).

14. How to design for high read throughput?

Use read replicas, caching, covering indexes, denormalization.

15. What is replication lag?

Delay between primary and replica due to heavy writes or network latency.

16. How to delete large datasets safely?

Delete in batches or drop partitions instead of full delete.

17. What is a materialized view?

Stored precomputed query result used for reporting and analytics.

18. How to design audit logging at scale?

Append-only tables, partitioning by date, archive old logs.

---

19. Explain CAP theorem in SQL systems.

Distributed systems can guarantee only two of Consistency, Availability, Partition tolerance.

---

20. How to design multi-tenant databases?

Shared schema, separate schema, or separate DB per tenant depending on isolation needs.

---

21. How to prevent hot partitions?

Avoid sequential keys, use hashing or random distribution.

---

22. OLTP vs OLAP difference?

OLTP handles transactions; OLAP handles analytics queries.

---

23. Backup and disaster recovery strategy?

Full/incremental backups, automated restore tests, multi-region replicas.

---

24. Safe schema migrations?

Backward-compatible changes, gradual deployment, backfill before switch.

---

25. How to reduce lock contention?

Short transactions, proper indexing, partitioning.

---

26. How to monitor database health?

Track latency, IOPS, lock waits, deadlocks, replication lag.

---

27. What is connection pooling?

Reusing database connections to improve performance and avoid exhaustion.

---

28. When to choose NoSQL over SQL?

Flexible schema, massive horizontal scale, event logging use cases.

---

29. How to design highly available SQL architecture?

Primary-replica setup, auto-failover, load balancing, backups.

---

30. Design database for large-scale e-commerce.

Partition orders, cache products, separate read/write DBs, async processing.