Neo4j - Practice Sheet

10 conceptual Q&A Databases

Q1. What type of database is Neo4j and what are its core strengths?

A: Neo4j is a relational, document, key-value, graph, or time-series store. Its strengths such as ACID guarantees, horizontal scale, or flexible schema guide the workloads it is best suited for.

Q2. How does Neo4j guarantee consistency and handle transactions?

A: Neo4j provides either full ACID transactions or eventual consistency depending on its CAP theorem positioning. Transaction support varies from none (simple K/V stores) to serializable isolation (relational DBs).

Q3. What index types does Neo4j support and when do you use each?

A: Neo4j offers B-tree, hash, full-text, spatial, or composite indexes depending on the engine. Choose based on query patterns: B-tree for range queries, hash for equality lookups, full-text for search.

Q4. How do you design a schema or data model in Neo4j?

A: Schema design in Neo4j involves normalizing relations (relational DB) or denormalizing for access patterns (document/key-value). Understanding query needs upfront prevents costly migrations later.

Q5. How does Neo4j scale horizontally?

A: Neo4j scales via replication (read replicas) for read-heavy workloads and sharding (partitioning data by key) for write-heavy ones. Some Neo4j implementations handle this automatically; others require manual configuration.

Q6. How do you monitor and tune slow queries in Neo4j?

A: Neo4j provides a slow query log, explain/explain plan, or profiling tools to surface inefficient queries. Common fixes include adding indexes, rewriting queries, or increasing cache size.

Q7. What backup and restore strategies does Neo4j support?

A: Neo4j supports logical backups (export SQL or BSON), physical backups (filesystem snapshot), and continuous replication to a standby. Point-in-time recovery requires WAL/oplog archiving on top of backups.

Q8. How do you handle schema migrations in Neo4j?

A: Schema migrations in Neo4j are managed with versioned migration files applied in order by a migration tool. Migrations should be idempotent and tested in staging before production to avoid downtime.

Q9. What security features does Neo4j provide?

A: Neo4j supports role-based access control, encrypted connections (TLS), encryption at rest, and audit logging. Always restrict user privileges to the minimum needed and disable anonymous access in production.

Q10. What are common anti-patterns when using Neo4j in production?

A: Common mistakes include selecting all columns instead of needed fields, missing indexes on frequently queried columns, running migrations without backups, and storing large blobs directly in the database instead of object storage.