

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

A: Firebase Firestore 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 Firebase Firestore guarantee consistency and handle transactions?

A: Firebase Firestore 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 Firebase Firestore support and when do you use each?

A: Firebase Firestore 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 Firebase Firestore?

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

Q5. How does Firebase Firestore scale horizontally?

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

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

A: Firebase Firestore 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 Firebase Firestore support?

A: Firebase Firestore 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 Firebase Firestore?

A: Schema migrations in Firebase Firestore 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 Firebase Firestore provide?

A: Firebase Firestore 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 Firebase Firestore 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.