MongoDB Interview Q&As (Part 2)

🔹 Section 7: Advanced Aggregation

Q31. What is $unwind in MongoDB?
Answer: $unwind deconstructs an array field → creates one document per array element.

```
db.orders.aggregate([{ $unwind: "$items" }])
```

💡 **Tip:** Combine with $group to analyze array data (e.g., item sales).

Q32. What is $facet used for?
Answer: Runs multiple pipelines in a single aggregation. Example → get paginated results + counts at once.
💡 **Tip:** Useful for dashboards and reports.

Q33. What is $project in aggregation?
Answer: Reshapes documents, includes/excludes fields, creates computed values.

```
db.users.aggregate([{ $project: { name: 1, yearOfBirth: { $year: "$dob" } } }])
```

💡 **Tip:** Think of $project = SQL's SELECT column AS alias.

Q34. What is $match vs $group?
Answer:

- $match → filters documents (like WHERE).
- $group → groups documents (like GROUP BY).
  💡 **Tip:** Always use $match before $group for performance.

Q35. What is a pipeline in MongoDB?
Answer: A series of aggregation stages where output of one stage becomes input of the next.
💡 **Tip:** Analogy = SQL query execution plan.

🔹 Section 8: Schema Design & Modeling

Q36. What is the difference between Embedded and Referenced documents?
Answer:

- Embedded → Store related data in the same document.
- Referenced → Store in separate collections using ObjectId.
  💡 **Tip:** Embed for bounded small data, reference for unbounded/large data.

Q37. What are Capped Collections?
Answer: Fixed-size collections that overwrite oldest documents when full.
💡 **Tip:** Perfect for logs, caches, IoT time-series.

Q38. What is GridFS in MongoDB?
Answer: A specification for storing large files (>16 MB) by splitting into smaller chunks.
💡 **Tip:** Mention media storage (images, videos) as use case.

Q39. What is the 16MB document limit in MongoDB?
Answer: MongoDB restricts max document size to 16MB. For larger → use GridFS.
💡 **Tip:** Common trick question — always mention GridFS.

Q40. What is the difference between normalized and denormalized schema in MongoDB?
Answer:

- Normalized → references (less duplication, more joins).
- Denormalized → embedded/duplicated (fast reads, more storage).
  💡 **Tip:** MongoDB often prefers denormalization for performance.

🔹 Section 9: Replication & Sharding

Q41. What is a Replica Set in MongoDB?
Answer: Group of mongod instances with one primary (writes) and multiple secondaries (reads/failover).
💡 **Tip:** Interviewers love: *"Replication = high availability."*

Q42. How does MongoDB handle failover?
Answer: If primary fails, secondaries hold an election to choose a new primary.
💡 **Tip:** Always mention automatic failover.

Q43. What is Sharding in MongoDB?
Answer: Horizontal scaling method → splits data across multiple shards using a shard key.
💡 **Tip:** Good for big data apps like e-commerce, logs.

Q44. How do you choose a Shard Key?
Answer: Pick a field with:

- High cardinality
- Even distribution
- Matches query patterns
  💡 **Tip:** Bad shard key = hotspots + uneven load.

Q45. What is the difference between Replication and Sharding?
Answer:

- Replication → copies of same data across servers (HA).
- Sharding → distributes data across servers (scalability).
  💡 **Tip:** Many candidates confuse both — highlight difference.

🔹 Section 10: Advanced Indexing & Performance

Q46. What are Multikey Indexes?
Answer: Indexes built on array fields, creating an index entry for each

element.
💡 **Tip:** Useful for tags, categories.

Q47. What is a Text Index in MongoDB?
Answer: A **Text Index** is a special type of index in MongoDB that allows you to **search text inside string fields** (like full-text search).

```
db.articles.createIndex({ content: "text" })
```

```
db.articles.find({ $text: { $search: "MongoDB" } })
```

💡 **Tip:** Good for blogs, search bars.

Q48. What is a Geospatial Index?
Answer: Geospatial indexes (2d, 2dsphere) allow MongoDB to efficiently query **location-based data** (latitude/longitude).

Used for queries like *find nearby restaurants*.

Supports $near, $geoWithin.
💡 **Tip:** Mention Google Maps-like queries.

Q49. What is a Covered Query?
Answer: A query satisfied entirely by the index → no need to fetch documents.
💡 **Tip:** Saves I/O, improves performance.

Q50. How do you use explain() in MongoDB?
Answer: The .explain("executionStats") method shows **how MongoDB executes a query**.

Helps you check **index usage** and query efficiency.

```
db.users.find({ age: 25 }).explain("executionStats")
```

Shows query plan, index usage, scanned docs.

Output (important fields):

- **nReturned** → number of documents returned.
- **totalDocsExamined** → how many docs scanned.

- **indexUsed** → which index (if any) was used.
  💡 **Tip:** Check nReturned vs totalDocsExamined.

🔹 Section 11: Security & Transactions

Q51. How do you secure MongoDB in production?
Answer: Enable authentication, TLS, RBAC, encryption at rest, IP whitelisting.
💡 **Tip:** Mention “Never expose MongoDB to internet without auth.”

Q52. What is Role-Based Access Control (RBAC)?
Answer: MongoDB assigns roles (read, readWrite, dbAdmin) to users.
💡 **Tip:** Common enterprise feature.

Q53. What are Transactions in MongoDB?
Answer: Multi-document ACID transactions supported since v4.0.
💡 **Tip:** Mention important for financial apps.

Q54. What are Write Concerns in MongoDB?
Answer: Define acknowledgment level for writes (w:0,1,majority).
💡 **Tip:** Higher write concern = safer, slower.

Q55. What are Read Concerns in MongoDB?
Answer: Define consistency level for reads (local, majority, linearizable).
💡 **Tip:** Important in replicated environments.