1. What makes MongoDB the best?
Document-oriented (DO)
High performance (HP)
High availability (HA)
Easy scalability
Rich query language

2.If you remove an object attribute, is it deleted from the database?
Yes, it is deleted. Hence, it is better to eliminate the attribute and then save the object again.

3. **Explain the situation when an index does not fit into RAM.**
When an index is too huge to fit into RAM, then MongoDB reads the index, which is faster than reading RAM because the indexes easily fit into RAM if the server has got RAM for indexes, along with the remaining set.

4. **How does MongoDB provide consistency?**
MongoDB uses the **reader–writer locks**, allowing simultaneous readers to access any supply like a database or a collection but always offering private access to single writes.

5. **What is the use of Journaling in MongoDB?**
Journaling is used for safe backups in MongoDB.

6. **What is the use of Profiler?**
Profiler is used to show the performance characteristics of every operation against the database.

7. What is Vertical Scaling?
Vertical scaling adds more CPU and storage resources to increase capacity.

8.Define Horizontal Scaling.
Horizontal scaling divides the dataset and distributes data over multiple servers, or shards.

9. What are the components of the Sharded cluster?
The sharded cluster has the following components:

- Shards
- Query routers
- Config servers

10. What is the use of the pretty() method?
The pretty() method is used to show the results in a formatted way.

11. What is the use of the dot notation in MongoDB?

MongoDB uses the dot notation to access the elements of an array and the fields of an embedded document.

12. What is Splitting in MongoDB?

Splitting is a background process that is used to keep chunks from growing too large.

13. What is the difference between MongoDB and MySQL?

Although both MongoDB and MySQL are free and open-source databases, there is a lot of difference between them in terms of data representation, relationships, transaction, querying data, schema design and definition, performance speed, normalization, and many more. To compare MySQL with MongoDB is like a comparison between relational and non-relational databases.

14. Explain the structure of ObjectID in MongoDB.

ObjectIds are small, likely unique, fast to generate, and ordered. ObjectId values consist of 12 bytes, where the first four bytes are a timestamp that reflect the ObjectId's creation.

15. Why MongoDB is not preferred over a 32-bit system?

When running a 32-bit build of MongoDB, the total storage size for the server, including data and indexes, is 2 gigabytes. For this reason, do not deploy MongoDB to production on 32-bit machines. If you're running a 64-bit build of MongoDB, there's virtually no limit to storage size.

16. Does MongoDB support ACID transaction management and locking functionalities?

ACID stands that any update is:
Atomic: it either fully completes or it does not
Consistent: no reader will see a "partially applied" update
Isolated: no reader will see a "dirty" read
Durable: (with the appropriate write concern)

Historically MongoDB does not support default multi-document ACID transactions (multiple-document updates that can be rolled back and are ACID-compliant). However, MongoDB provides atomic operation on a single document. MongoDB 4.0 will add support for multi-document transactions, making it the only database to combine the speed, flexibility, and power of the document model with ACID data integrity guarantees.

17. Should I normalize my data before storing it in MongoDB?

It depends from your goals. Normalization will provide an update efficient data representation. Denormalization will make data reading efficient.

In general, use embedded data models (denormalization) when:
you have "contains" relationships between entities.
you have one-to-many relationships between entities. In these relationships the "many" or child documents always appear with or are viewed in the context of the "one" or parent documents.

In general, use normalized data models:
when embedding would result in duplication of data but would not provide sufficient read performance advantages to outweigh the implications of the duplication.

to represent more complex many-to-many relationships.

to model large hierarchical data sets.

Also normalizing your data like you would with a relational database is usually not a good idea in MongoDB. Normalization in relational databases is only feasible under the premise that JOINs between tables are relatively cheap. The $lookup aggregation operator provides some limited JOIN functionality, but it doesn't work with sharded collections. So joins often need to be emulated by the application through multiple subsequent database queries, which is very slow

18. What happens if an index does not fit into RAM? What happens if an index does not fit into RAM?

If the indexes do not fit into RAM, MongoDB reads data from disk which is relatively very much slower than reading from RAM.

19. What are Primary and Secondary Replica sets?

Primary and master nodes are the nodes that can accept writes. MongoDB's replication is 'single-master:' only one node can accept write operations at a time.

Secondary and slave nodes are read-only nodes that replicate from the primary.

20. How does MongoDB provide concurrency?

MongoDB uses reader-writer locks that allow concurrent readers shared access to a resource, such as a database or collection, but give exclusive access to a single write operation.

21.