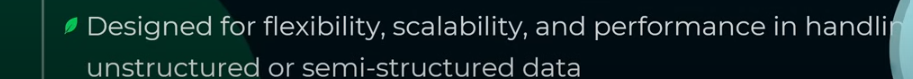
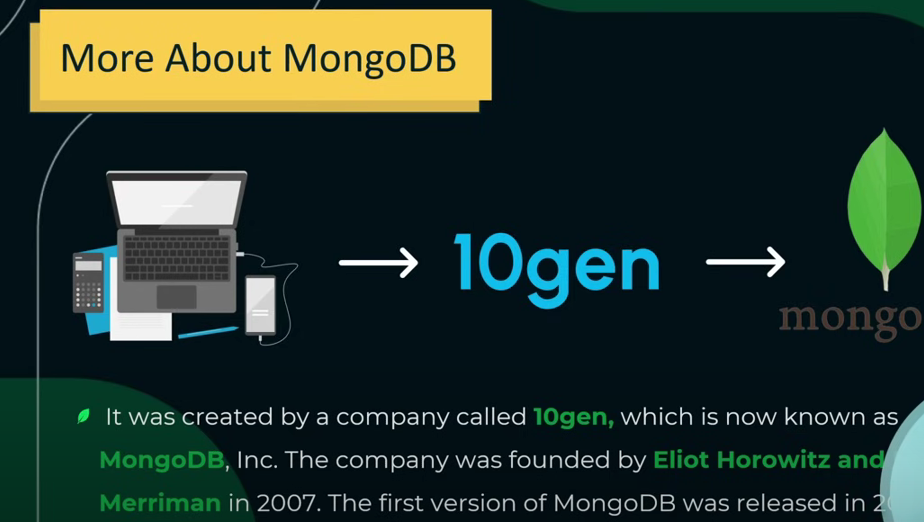
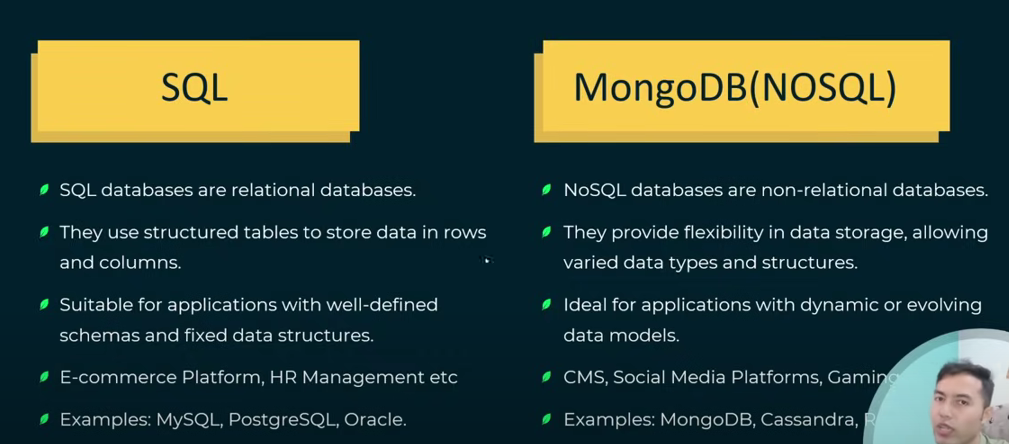
MongoDB





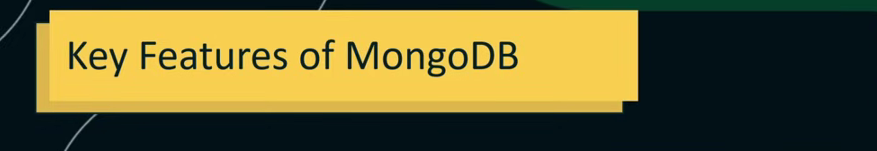


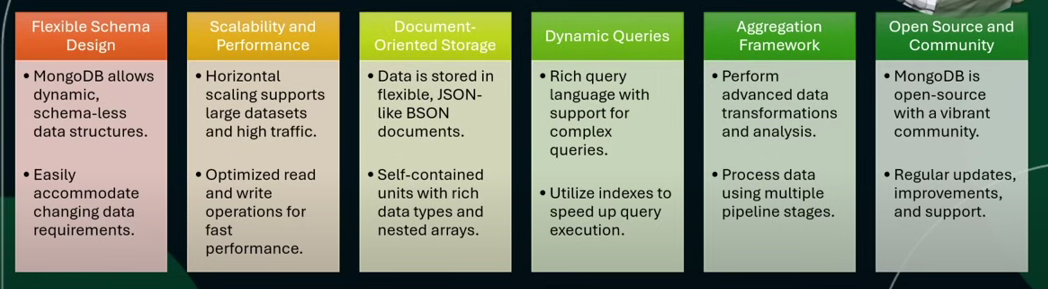
A yellow sign with black text

Description automatically generated

A computer screen shot of a computer

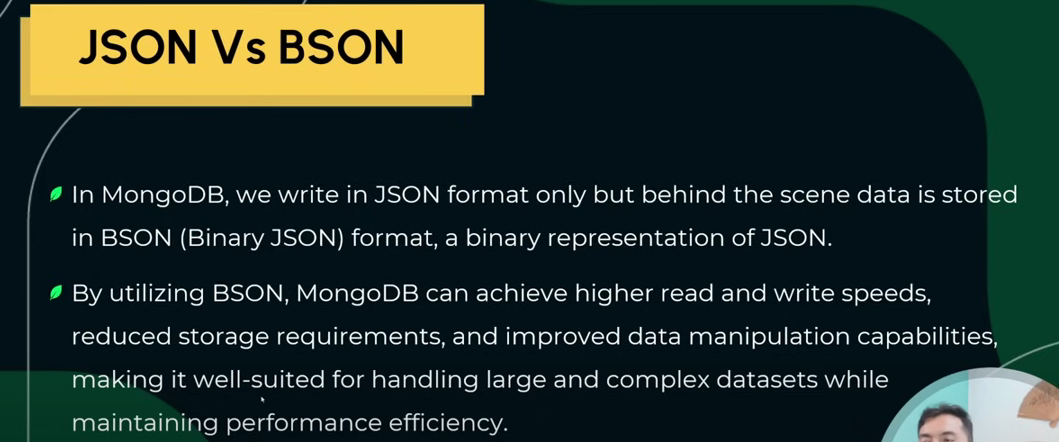
Description automatically generated





A screenshot of a computer

Description automatically generated





**Basic Level Questions**

**1. What is MongoDB and how is it different from SQL databases?**

MongoDB is a **NoSQL, document-oriented database**. Unlike SQL databases that store data in tables and rows, MongoDB stores data in **JSON-like documents** (BSON format), which allows for flexible and dynamic schemas.

**2. What are documents and collections in MongoDB?**

* A **document** is a single record in MongoDB, stored in BSON format.
* A **collection** is a group of MongoDB documents, similar to a table in relational databases.

**3. What is BSON? How is it different from JSON?**

**BSON** (Binary JSON) is a binary-encoded serialization of JSON-like documents. It supports more data types (like Date, Binary, ObjectId) and is optimized for speed and space.

**4. How do you insert, update, and delete documents in MongoDB?**

* **Insert**: db.collection.insertOne({ name: "Alice" })
* **Update**: db.collection.updateOne({ name: "Alice" }, { $set: { age: 25 } })
* **Delete**: db.collection.deleteOne({ name: "Alice" })

**5. What are indexes in MongoDB and why are they important?**

Indexes improve the **performance of queries** by allowing MongoDB to find data without scanning every document. Common types include:

* Single field index
* Compound index
* Text index
* Geospatial index

**🔹 Intermediate Level Questions**

**6. Explain the aggregation framework in MongoDB.**

The aggregation framework processes data records and returns computed results. It uses a **pipeline** of stages like:

* $match: filter documents
* $group: group by fields
* $sort: sort documents
* $project: reshape documents

**7. What is the difference between find() and findOne()?**

* find() returns a **cursor** to all matching documents.
* findOne() returns **only the first matching document**.

**8. How does MongoDB handle relationships between data?**

* **Embedding**: Store related data in the same document (good for one-to-few).
* **Referencing**: Use ObjectIds to reference documents in other collections (good for one-to-many or many-to-many).

**9. What are the different types of indexes in MongoDB?**

* **Single Field Index**
* **Compound Index**
* **Multikey Index** (for arrays)
* **Text Index**
* **Hashed Index**
* **Geospatial Index**

**10. How does MongoDB ensure data consistency and durability?**

* **Write concerns** define the level of acknowledgment requested from MongoDB.
* **Journaling** ensures durability by writing operations to a journal before applying them to the database.

**🔹 Advanced Level Questions**

**11. What is a replica set in MongoDB?**

A **replica set** is a group of MongoDB servers that maintain the same data set. It provides **high availability** and **automatic failover**. One node is primary, others are secondaries.

**12. Explain sharding in MongoDB.**

Sharding is MongoDB’s method for **horizontal scaling**. It splits data across multiple machines (shards) using a **shard key**. A **mongos** router directs queries to the appropriate shard.

**13. How does MongoDB handle concurrency and locking?**

MongoDB uses **multi-granularity locking**:

* Document-level locks (from version 3.0+)
* Readers and writers do not block each other on different documents.

**14. What are some common performance tuning techniques in MongoDB?**

* Use proper **indexes**
* Avoid large documents and deep nesting
* Use **projection** to return only needed fields
* Monitor with **MongoDB Atlas** or mongostat, mongotop

**15. How do you handle schema validation in MongoDB?**

MongoDB supports **JSON Schema validation** using the validator option in createCollection() or collMod.

**🔹 Scenario-Based Questions**

**16. How would you model a one-to-many relationship in MongoDB?**

* **Embed** if the "many" side is small and accessed with the "one".
* **Reference** if the "many" side is large or accessed independently.

**17. You have a collection with millions of documents. How would you optimize queries?**

* Use **indexes** on frequently queried fields.
* Use **pagination** with limit() and skip().
* Use **covered queries** (queries that can be answered using only the index).

**18. How would you migrate data from a relational database to MongoDB?**

* Use tools like **MongoDB Compass**, **mongoimport**, or **ETL pipelines**.
* Map relational tables to documents, considering embedding vs referencing.

**19. What would you do if a MongoDB query is running slow?**

* Use explain() to analyze the query plan.
* Check for missing indexes.
* Optimize schema and avoid large documents.

**20. How do you back up and restore a MongoDB database?**

* Use mongodump and mongorestore for binary backups.
* Use **MongoDB Atlas** for automated backups in the cloud.