**20.05.2025 ASSIGNMENT**

**Differences between NoSQL and SQL**

**1. Structure**

SQL databases are table based, while NoSQL databases can be document-oriented, key-value pairs, or graph structures. In a NoSQL database, a document can contain key value pairs, which can then be ordered and nested.

**2. Scalability**

SQL databases scale vertically, usually on a single server, and require users to increase physical hardware to increase their storage capacities. In effect, while cloud-storage options are available, SQL databases can be prohibitively expensive for businesses when dealing with vast amounts of [big data](https://www.coursera.org/articles/what-is-big-data-a-laypersons-guide).

NoSQL databases offer horizontal scalability, meaning that more servers simply need to be added to increase their data load. This means that NoSQL databases are better for modern cloud-based infrastructures, which offer distributed resources.

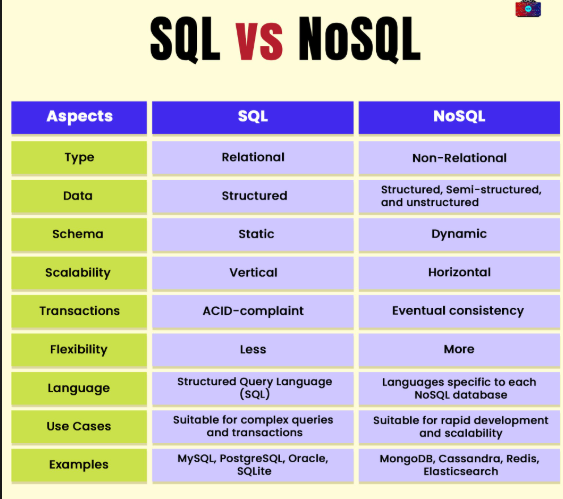
**3. Language**

SQL databases use SQL (Structured Query Language). NoSQL databases use JSON (JavaScript Object Notation), XML, YAML, or binary schema, facilitating unstructured data. SQL has a fixed-defined schema, while NoSQL databases are more flexible.

**4. Support**

SQL is a popular standard language that is well supported by many different database systems, while NoSQL has varying levels of support in various database systems.

Regarding support, you’ll generally find that more help is available for SQL databases than NoSQL. This is because SQL is a more established technology and thus has many more users and developers who can help you with your problems. In contrast, NoSQL is still relatively new, with less help available on forums or through the community. Your support options may be limited if you run into difficulties using it.



**Pros and cons of SQL**

**Pros of SQL:**

* SQL is widely understood and supported; most developers know it well.
* SQL is extremely useful for simple aggregations over large datasets, such as calculating averages.
* SQL is extremely useful for setting up simple ETL jobs, especially if the input and output formats are relational databases.
* SQL is well-documented and easy to learn.

**Cons of SQL:**

* The performance of SQL can be poor on substantial data sets because it requires multiple passes over the data to complete many operations (especially joins).
* Debugging SQL can be complicated because it doesn't provide informative error messages.
* The syntax of SQL tends to be verbose compared with programming languages like python or R, which makes it harder to write complex transformations as scripts or functions.

**Pros and cons of NoSQL**

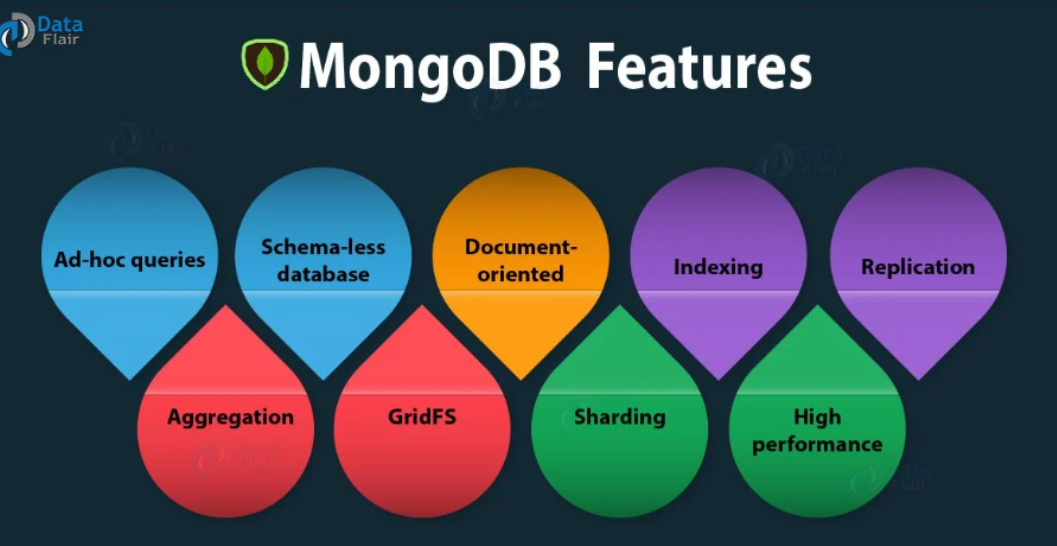
**Pros of NoSQL:**

* Flexible schema
* Usable on distributed infrastructure platforms
* Low-cost infrastructure
* High availability and throughput

**Cons of NoSQL:**

* Less mature technology and difficult to manage
* Limited query capabilities
* Data inconsistency and poor performance in some complex scenarios

**FEATURES OF MONGODB**



MongoDB is a popular **NoSQL** database known for its flexibility, scalability, and performance. Here are the key features of MongoDB:

**1. Document-Oriented Storage**

* Stores data in **BSON** (Binary JSON) format.
* Documents are similar to JSON objects with dynamic schemas (i.e., you don't need to predefine the structure).

**2. Schema-less (Flexible Schema)**

* Each document in a collection can have a different structure.
* Ideal for applications where the data structure changes frequently.

**3. High Performance**

* Fast read/write operations due to in-memory storage engine and efficient indexing.
* Supports embedded documents and arrays to reduce the need for joins.

**4. Horizontal Scalability**

* Built-in **sharding** support allows data to be distributed across multiple servers.
* Handles large volumes of data and high throughput.

**5. Rich Query Language**

* Supports powerful queries using JSON-like syntax.
* Queries can include filters, ranges, regular expressions, and aggregation pipelines.

**6. Indexing**

* Supports primary and secondary indexes.
* Indexes on any field improve query performance significantly.

**7. Aggregation Framework**

* Powerful aggregation pipeline similar to SQL’s GROUP BY, JOIN, etc.
* Allows transformation and computation on data within the database.

**8. Replication & High Availability**

* **Replica Sets** provide automatic failover and data redundancy.
* Ensures availability and disaster recovery.

**9. Geospatial Indexing**

* Built-in support for geospatial queries (e.g., finding nearby locations).

**10. Integration with Multiple Languages**

* Official drivers for many programming languages: Python, Java, Node.js, C#, PHP, etc.

**11. ACID Transactions (Since v4.0)**

* Supports multi-document ACID-compliant transactions for data integrity.

**12. Change Streams**

* Real-time data change tracking using **Change Streams**, useful for reactive applications.

**13. MongoDB Atlas**

* Fully managed cloud-based MongoDB service with automatic scaling, backups, and monitoring.