## ***SQL vs NoSQL***

### 

### ***When to use SQL ;***

**SQL is a good choice when working with related data. Relational databases are efficient, flexible and easily accessed by any application. A benefit of a relational database is that when one user updates a specific record, every instance of the database automatically refreshes, and that information is provided in real-time.**

**SQL and a relational database make it easy to handle a great deal of information, scale as necessary and allow flexible access to data — only needing to update data once instead of changing multiple files, for instance. It’s also best for assessing data integrity. Since each piece of information is stored in a single place, there’s no problem with former versions confusing the picture.**

**Most of the big tech companies use SQL, including Uber, Netflix and Airbnb. Even major companies like Google, Facebook and Amazon, which build their own database systems, use SQL to query and analyze data.**

### 

### **When to use NoSQL ;**

**While SQL is valued for ensuring data validity, NoSQL is good when it’s more important that the availability of big data is fast. It’s also a good choice when a company will need to scale because of changing requirements. NoSQL is easy-to-use, flexible and offers high performance.**

**NoSQL is also a good choice when there are large amounts of (or ever-changing) data sets or when working with flexible data models or needs that don't fit into a relational model. When working with large amounts of unstructured data, document databases (e.g. MongoDB, and Amazon DocumentDB) are a good fit. For quick access to a key-value store without strong integrity guarantees, Redis may be the best choice. When a complex or flexible search across a lot of data is needed, Elastic Search is a good choice.**

**Scalability is a significant benefit of NoSQL databases. Unlike with SQL, their built-in sharding and high availability requirements allow horizontal scaling. Furthermore, NoSQL databases like Cassandra, developed by Facebook, handle massive amounts of data spread across many servers, having no single points of failure and providing maximum availability.**

## **MongoDB vs SQL Server**

**°The key differences between MongoDB and SQL Server:**

### **Database Model**

MongoDB is a Non-Relational Database based on a document-oriented structure that internally is based on a key-value structure.

SQL Server is a Relational Database based on relatable structured tables with various rows and columns.

### **Implementation Language**

MongoDB is implemented in JavaScript, Python, Java, PHP, C++, C, Ruby, Perl, C, C++ whereas SQL Server is implemented in C and C++.

### **License**

MongoDB is an Open-Source DBMS whereas a SQL Server license is required for commercial purposes.

### **Data Schema**

MongoDB has a flexible dynamic schema that can easily be changed with the evolution of data, application, or business.

SQL Server has a fixed Schema that is predefined before inserting any data. Fixed schema means schema can’t be changed as data, application, or business requirement evolves.

### **Query Language**

MongoDB uses MongoDB Query Language to query unstructured data from the database.

SQL Server uses SQL Query Language to Create Tables, Insert, Fetch or Update data in the database.

### **Scalability**

MongoDB supports Horizontal Scaling in which data is distributed across clusters. The Sharding process is used to implement horizontal scaling which results in an always-up server.

SQL Server supports vertical scaling which is the traditional approach. In vertical scaling, physical or virtual resources are added to the hosting server of the database.

### **Map Reduce**

MongoDB supports a data processing algorithm for concentrating large volumes of data into aggregated results. MongoDB provides the MapReduce database command to perform map-reduce operations.

SQL Server does not support the Map-Reduce method.