**Basic Comparison of Relational Vs. NoSQL Databases**

In relational databases, relationships are an integral part of how they function and describe how tables are connected to each other through keys, allowing data in one table to relate to data in another. Two common types of relationships are one-to-many and many-to-many, each of which contains its own distinct pros and cons. A one-to-many relationship happens when one record in a table is linked to multiple records in another table, like a single customer having multiple orders. While a many-to-many relationship occurs when multiple records in one table relate to multiple records in another, such as students enrolled in multiple courses and each course having multiple students. These relationships help organize efficiently and maintain data integrity (Gillis et al., 2024).

Relational databases offer several advantages that make them popular for structured data management. Having this structure helps keep information organized and accurate, even when working with an abundance of it. In addition, asking detailed questions using SQL makes it easier to find, change, or organize the data as you want, leading to more personalization. Relational databases also enforce relationships between tables, reducing repeated data and supporting data storage. These features make them ideal for systems that require ownership and strict organization of information (Gillis et al., 2024).

NoSQL databases provide a different set of advantages more suited for larger-scale applications. They can handle vast amounts of unstructured or semi-structured data, which makes them useful for these massive data scenarios. NoSQL systems are designed for flexibility, often using dynamic schemas that adjust as data continues to change. Additionally, their architecture supports distributed environments, which allows for top performance for real-time applications (Foote, 2022).

Despite their benefits, both database types come with disadvantages. Relational databases lack flexibility due to their fixed schemas, which can be a challenge when handling evolving data types. They also scale vertically, meaning performance can suffer as data grows or traffic increases. Complex joins across multiple tables can slow down query performance in large systems. These issues can make relational databases less ideal for agile or high-speed environments (Gillis et al., 2024).

NoSQL databases, while dynamic, often lose consistency in favor of availability. They typically do not follow strict ACID properties, which can lead to challenges in maintaining data accuracy across distributed systems. The absence of a universal query language ends up making data access and manipulation a harder task to accomplish. Developers may also need to write more custom code as a result. This added complexity can increase development time and severely dampen the ability to perform maintenance. (Foote, 2022).

MySQL is a popular relational database that uses structured tables with fixed schemas to ensure consistent data organization. It supports complex queries through SQL, making it well-suited for applications that rely on structured and interrelated data. One of its major strengths is ACID compliance, which ensures reliable and consistent transactions by maintaining atomicity, consistency, isolation, and durability. This is essential for applications like banking or inventory systems, where data accuracy is critical. MySQL is also widely supported and integrated into many development stacks, making it accessible to a broad range of users (MongoDB, n.d.).

MongoDB is a popular NoSQL database that uses JSON documents to store data, producing a dynamic environment. This flexibility enables developers to store changing data without altering a static table structure. MongoDB also supports horizontal scaling through sharding, which splits data across multiple servers to improve performance and capacity. This makes it highly effective for handling massive volumes of data in real time, such as for social media platforms or analytics tools. This is also a tactic that is present in MMORPGs like World of Warcraft to allow the game not to overload with individual players in highly populated areas. Its document model aligns closely with how data is used in modern web applications, speeding up development and deployment (MongoDB, n.d).

**Reference**

Foote, K. (2022, November 17). *NoSQL Databases Advantages and Disadvantages.* Dataversity. <https://www.dataversity.net/nosql-databases-advantages-and-disadvantages/>

Gillis, A., & Lutkevich, B., & Biscobing, J. (2024, June 21). *Relational database.* TechTarget. <https://www.techtarget.com/searchdatamanagement/definition/relational-database>

Grammarly Inc. (n.d.). *Grammarly* [Grammar and writing enhancement tool]. <https://www.grammarly.com/>

MongoDB. (n.d.). *MongoDB vs. MySQL*. <https://www.mongodb.com/resources/compare/mongodb-mysql>

OpenAI. (2025, May 31). *ChatGPT (May 31 version) [Large language model]*. <https://chat.openai.com/>