Difference Between SQL and NoSQL Databases

Aspect	SQL (Relational Database)	NoSQL (Non-Relational Database)
Full Form	Structured Query Language	Not Only SQL
Data Model	Tables with rows and columns	Documents, key-value pairs, graphs, or wide-column stores
Schema	Fixed and predefined schema	Dynamic or flexible schema
Scalability	Vertically scalable (by increasing server power)	Horizontally scalable (by adding more servers)
Consistency Model	Follows ACID (Atomicity, Consistency, Isolation, Durability)	Often follows BASE (Basically Available, Soft state, Eventual consistency)
Relationships	Supports complex joins and relationships between tables	Typically handles relationships through embedding or linking at the application level
Query Language	Uses SQL for defining and manipulating data	Uses database-specific query languages (e.g., MongoDB Query Language, Cassandra CQL)
Data Type Supported	Best for structured data	Best for unstructured, semi-structured, or rapidly changing data
Examples	MySQL, PostgreSQL, Oracle, SQL Server	MongoDB, Cassandra, Redis, CouchDB, DynamoDB

Aspect	SQL (Relational Database)	NoSQL (Non-Relational Database)
Transactions	Strong transactional support for data integrity	May have limited transactional support depending on the system
Performance	Slower for large, unstructured data sets due to normalization and joins	Faster for unstructured or high-volume workloads due to denormalized design
Use Cases	Financial systems, ERP, CRM — where consistency and reliability are crucial	Real-time analytics, IoT, big data, social media platforms — where scalability and flexibility are key

---

Quick Summary

- SQL Databases → Best for structured, consistent, and transactional systems.
- NoSQL Databases → Best for scalable, flexible, and high-performance systems handling unstructured data.