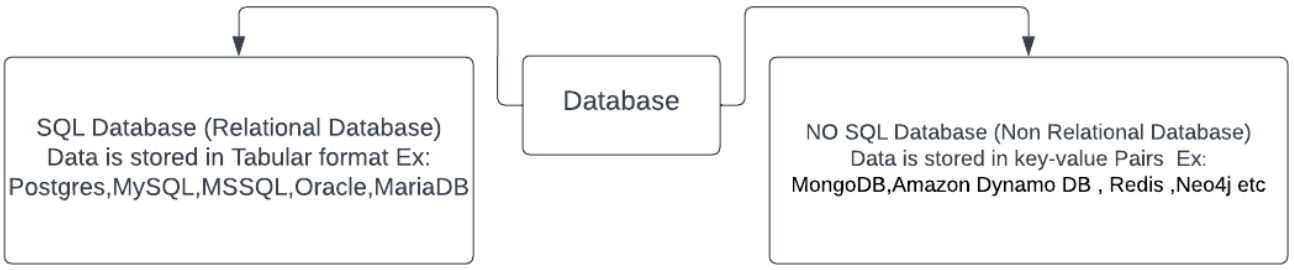
Database

There are mainly two types of Databases SQL Database and No-SQL Database



Centralized Database: A centralized database is a type of database architecture in which all data is stored in a single location, and all access to the data is controlled by a central server. In this architecture, the database server is responsible for managing and controlling access to the data, and clients access the data through the server.

Distributed Database: A distributed database is a type of database architecture in which data is distributed across multiple nodes or servers in a network. In a distributed database, each node stores a portion of the data, and the nodes work together to provide a unified view of the data to users and applications. It is used in Ecommerce, Social networks , Banking , Healthcare

Cloud Database: A cloud database is a type of database that is hosted on a cloud computing platform, such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform. In a cloud database, data is stored in the cloud rather than on a local server, and users can access the data from anywhere with an internet connection.

Object Oriented Database: An object-oriented database is a type of database that is based on the principles of object-oriented programming. In an object-oriented database, data is represented as objects, which are instances of classes that contain data and methods for manipulating that data.

Hierarchical Database: A hierarchical database is a type of database that organizes data into a tree-like structure, where each record has a parent-child relationship with other records in the database. In a hierarchical database, the data is organized into levels, with each level representing a different type of record.

SQL Database:

- SQL database is used in case of Structured Data
- SQL databases are optimized for transactions and provide ACID (Atomicity, Consistency, Isolation, and Durability) compliance
- SQL databases provide powerful querying capabilities that allow for ad-hoc queries and complex joins. This makes them a good choice for applications that require advanced data analytics and reporting.
- Scalability: SQL databases can scale vertically by adding more powerful hardware, making them a good choice for applications that require high performance and scalability.

When not to use SQL databases:

- Unstructured data: If your application deals with large volumes of unstructured data.
- High write throughput: SQL databases can be slower when it comes to high write throughput or real-time data processing or high concurrency.

- Horizontal scalability: SQL databases can be more difficult to scale horizontally by adding more nodes to a cluster. NoSQL databases may be more suitable for applications that require horizontal scalability and fault tolerance.
- Cost: SQL databases can be expensive to license and require significant hardware resources. NoSQL databases may be a more cost-effective option for applications with large amounts of data.

Ex:MySQL,Postgres,MSSQL,Oracle,MariaDB

Table: customers

customer_id	first_name	last_name	phone	country
1	John	Doe	817-646-8833	USA
2	Robert	Luna	412-862-0502	USA
3	David	Robinson	208-340-7906	UK
4	John	Reinhardt	307-242-6285	UK
5	Betty	Taylor	806-749-2958	UAE

NoSQL Database:

NoSQL databases are becoming increasingly popular due to their ability to handle large, complex, and unstructured data.

When to use NoSQL databases:

- Large amounts of unstructured data
- Scalability: NoSQL databases are highly scalable and can easily accommodate increasing data.
- High availability: provide high availability and fault tolerance, making them a good choice for applications that require continuous availability.
- Flexibility: NoSQL databases are schemaless, which means they can accommodate changes to data structure without requiring changes to the database schema.

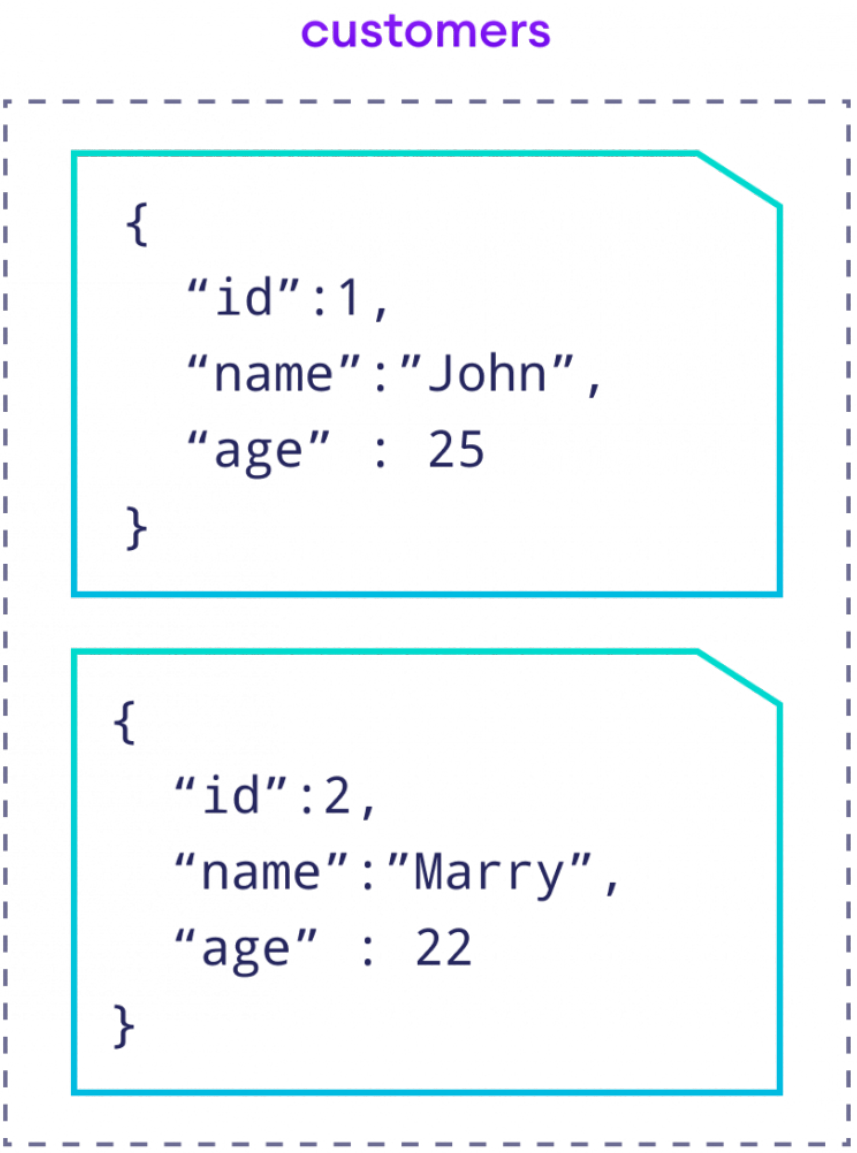
When not to use NoSQL databases:

- Transactions
- Complex data relationships: NoSQL databases do not support complex data relationships as well as traditional relational databases, so they may not be the best choice for applications with complex data models.
- Consistency: NoSQL databases do not provide the same level of data consistency as traditional relational databases
- Analytics and reporting: NoSQL databases are not always the best choice for analytics and reporting, particularly if the data is highly structured and requires complex queries.

In summary, NoSQL databases are a good choice for applications that deal with large amounts of unstructured data, require high scalability and availability, and are flexible enough to accommodate changes to the data structure. However, they may not be the best choice for applications that require strict data consistency, complex data relationships, transactions, or advanced analytics and reporting.

Types of NoSQL Database

1. Document Databases:These databases store data in a document format, such as JSON or BSON. Each document can have its own unique structure, allowing for more flexibility in data modeling. well-suited for applications that require high-speed access to large volumes of data They are also highly scalable, allowing them to easily accommodate increases in data volume and traffic. Popular document databases include MongoDB, Couchbase, and RavenDB.
2. Time series Databases: Time series databases are designed to handle large volumes of time series data such as sensor data, stock market data, or website traffic data, with efficient storage and querying of data points based on their timestamps. They often provide specialized functions for analyzing and processing time series data, such as aggregation, interpolation, and anomaly detection.Time series databases typically use a columnar or compressed storage format to optimize storage and retrieval of time-stamped data.Popular time series databases include InfluxDB, TimescaleDB, and OpenTSDB. They are often used in combination with other types of databases, such as graph databases and document databases, to create a complete data management solution.
3. Column Databases:These databases store data in columns instead of rows, which allows for high scalability and fast querying of large data sets. Column-family stores are commonly used for big data applications and data warehousing.Useful for handling large volumes of structured data that require fast read and write performance. They are designed to handle horizontal scaling, and can support high levels of concurrency. Popular column-family stores include Apache Cassandra, HBase, and Google Bigtable.
4. Realtime Databases:Real-time databases are optimized for fast read and write performance, and are typically used in applications where data needs to be processed and analyzed in real-time, such as in monitoring and control systems, financial trading platforms, and online gaming platforms,chat applications and messaging platforms.Real-time databases support features such as low latency, high availability, and data synchronization across multiple devices and locations.They often use specialized data structures and algorithms to optimize data processing and analysis, and support features such as event-driven triggers and real-time data streaming.Popular real-time databases include Firebase Realtime Database, Apache Kafka, and Redis. They are often used in combination with other types of databases, such as document databases and key-value databases.
5. Key-value Databases:These databases store data as key-value pairs, where each value is associated with a unique key. Key-value stores are known for their simplicity and high performance.Highly scalable and performant, making them well-suited for applications that require high-speed access to large volumes of data. They are often used as a caching layer for web applications, where they can store frequently accessed data in memory for faster retrieval. Key-value databases can also be used as a persistent data store, where they can handle high write throughput and support large datasets. Popular key-value stores include Redis, Amazon DynamoDB, and Riak.They are often used in combination with other types of NoSQL databases, such as document databases and column-family stores, to create a complete data management solution.
6. Graph Databases:These databases store data as nodes and edges in a graph, allowing for easy querying of complex relationships between data points. Graph databases are often used for social networks, recommendation engines, and fraud detection. Useful for handling complex, highly interconnected data, such as social networks, recommendation engines, and network topology data. They are designed to support graph queries, which allow users to traverse and analyze the relationships between entities in the graph.They are designed to handle horizontal scaling.Popular graph databases include Neo4j, OrientDB, and ArangoDB.



Some factors to consider when deciding between SQL and NoSQL databases include:

- Data structure: SQL databases are designed for structured data with predefined schemas, while NoSQL databases are designed for unstructured or semi-structured data.
- Data access patterns: SQL databases are well-suited for complex queries and support for joins, while NoSQL databases are designed for high throughput and low latency, and often use simple key-value access patterns.
- Scalability: NoSQL databases are designed for horizontal scalability, with the ability to distribute data across multiple nodes, while SQL databases are often more limited in their scalability options.
- Performance: NoSQL databases are often faster and more efficient than SQL databases for certain types of applications and data access patterns.
- Consistency: SQL databases are designed for strong consistency and transactional integrity, while NoSQL databases often sacrifice consistency for scalability and performance.
- Data integrity: SQL databases have built-in support for enforcing data integrity constraints, such as primary key and foreign key constraints, while NoSQL databases often rely on application-level logic for data validation and consistency.
- Data storage: SQL databases store data in tables with fixed columns and data types, while NoSQL databases can store data in a variety of formats, including key-value pairs, documents, and graphs.