



L-7 Database Systems

(ACCA7002)

ASSIGNMENT 2

Madhusudan Ghumare

(2116948)

Abstract

This report presents a detailed overview, use cases, and comparison between NoSQL and relational databases. In NoSQL, the focus is on the document, graph, key-value, and wide-column databases. The practical segment demonstrates the implementation of relational databases in MySQL and document NoSQL databases in MongoDB.

Table of Contents

Abstract	2
Table of Contents	3
Comparison Between Different Types of NoSQL Databases.....	4
Document Databases	4
Graph Databases.....	5
Key-value Databases	5
Wide Column Databases	6
A choice between NoSQL and Relational Database.....	7
References.....	8

Comparison Between Different Types of NoSQL Databases

NoSQL databases are divided into four categories. They include document, graph, key-value, and wide-column databases.

Document Databases

Document databases store organized, unstructured, and semi-structured information as documents and collections [1]. The documents are comparable to relational database records. Each document stores data related to one object. A group of related documents is stored in a collection that is comparable to relational database data. The document format can be JSON, BSON, or XML [2]. Document databases include MongoDB, CouchDB, and RavenDB [3]. The documents in a document database can have a variable number of data fields. The data fields do not have to be structured in a specific manner. It allows extra data fields to be stored in documents with additional information. As such, documents do not conform to the same rigid table structure as relational databases. The missing data fields in a document are omitted, while extra data fields in other documents are effectively stored [2]. Every document is assigned a unique key value during storage. The key is indexed as a pointer, including the path to the document's storage location inside the database. The key facilitates faster retrieval of documents and other data manipulation operations. Document databases have a query language that retrieves documents according to the document's contents.

Document-based databases are suitable for storing and retrieving information related to single objects. As such, it is ideal for maintaining user profile information where the user is at liberty to provide or withhold any information of their choice without being confined to a particular data structure [2].

Graph Databases

Graph databases are NoSQL databases that store data and represent data in graphs. The data comprises nodes and edges [1]. The nodes serve as the objects of data, while edges describe the connections between the stored information. The nodes in graph databases are comparable to relational database tables, while the edges are similar to records. Every node has a unique identifier used to describe the role of a node in the graph database. The entities in graph databases do not rely on entity relationships, as is the case for relational databases. Instead, nodes are connected directly to each other using edges [3]. Just like nodes, the edges have unique values that show their respective nodes' start and endpoints. Graph database models can be of two types: Resource Description Framework (RDF) graphs and Property graphs [4]. RDF graphs are purposely designed to facilitate data integration. Data integration involves linking data together in knowledge graphs, such as in statistical companies and healthcare facilities. On the other hand, property graphs are descriptive and effectively used in data analysis [4].

Graph databases are suitable for storing information for social media. It can store all user data, analyze their level of engagement and relations with other users, and determine the level of liveliness depending on the volume of their activity. Graph databases include Neo4j, Microsoft Azure Cosmos DB, OrientDB, and ArangoDB.

Key-value Databases

Key-value databases are a form of NoSQL database that store data in key-value pairs. The keys uniquely identify the values that relate to the objects stored in the database [3]. The key-value pairs are stored independently of each other to form database records. Unlike relational databases with finite data fields of specific type and length, key-value databases do not conform to a particular structure. However, data and information in key-value databases

are organized in columns [1]. The keys attached to each key pair are used during data retrieval. Key value databases are considered the simplest type of NoSQL database. The implementation is simple and precise, and its data retrieval process does not rely on a querying database [5].

CouchDB, DynamoDB, and BerkeleyDB are the most common key-value databases. Key-value databases store information on user sessions from successful login authentication to logout point. It is also used to record shopping sessions on e-Commerce websites [5].

Wide Column Databases

Wide column databases are NoSQL databases that store data in columns instead of rows standard in relational databases. The columns contain rows, each with a key, and can have numerous segments [1]. The columns can have a variable number of rows containing different data types. Each column is separate from the other. The related columns are grouped into families, where independent families are stored separately from other column families. Each row in a column is identified with a key. The key is stored in the first column within a column family. Similarly, each column has a column key name used to identify rows. It is also used during querying columns in a wide columnar database. Tracking of data input and modifications is possible through the value and timestamp inserted after the column key [6]. Relational databases have each column with the same number of rows, unlike wide-column databases with a variable number of rows per their column. Particular columns with empty rows during data insertion are simply omitted rather than holding null or empty values like in relational databases.

Wide column databases are best suited for intensive data processing needs, such as in data warehouses [6]. They include Apache Cassandra and HBase [3].

A choice between NoSQL and Relational Database

NoSQL databases are best used for storing semi-structured and unstructured data. The data can include documents, random data for social platforms, and analytical data with huge disparities. The data stored in NoSQL databases can change in structure, type, and length without redesigning the respective database. On the other hand, data management systems build around a traditional relational design are suitable for relational databases. The data has to be ordered in a specific finite structure where the data entry mechanism has to conform to a specified order. The data is expected to remain rigid. The relational nature of data makes it easy to ascertain the integrity of the data stored. It enables data to be updated once through an update statement rather than updating multiple files, as is the case with distributed NoSQL databases [7].

NoSQL databases are best suited for storing and maintaining big data where fast availability is a top priority. It offers high performance and flexibility in managing scalable data. NoSQL can scale horizontally, where more hubs can be added to the database framework for extra data storage space and higher processing power [1]. This can be achieved through the addition of servers to the NoSQL database. Horizontal scalability is preferred for its inexpensive implementation that does not compromise performance, unlike vertical scalability.

NoSQL databases are best suited for maintaining flexible data. Their flexibility allows for adaptation to changing data structure that does not conform to predetermined characteristics. They are dynamic with unique agility to suit evolving and advancement in data stored. On the other hand, relational databases are less flexible, where changes to the data structure can be tedious and time-consuming. It is suitably used for data known to be static save for vertical growth that requires enhanced storage to accommodate increasing database data.

References

- [1] T. A. Taha, "The SQL vs NoSQL Differences and Similarities," *International Journal of Scientific & Engineering Research (IJSER)*, vol. X, no. 8, pp. 58-63, A 2019.
- [2] A. Williams, "NoSQL database types explained: Document-based databases," 3 November 2021. [Online]. Available: <https://www.techtarget.com/searchdatamanagement/tip/NoSQL-database-types-explained-Document-based-databases>.
- [3] V. Sumalatha and S. Pabboju, "Overview of NoSQL Databases and A Concise Description of MongoDB," *International Journal of Engineering Research & Technology (IJERT)*, vol. IX, no. 12, pp. 295-299, December 2020.
- [4] A. Williams, "NoSQL database types explained: Graph," 30 December 2021. [Online]. Available: <https://www.techtarget.com/searchdatamanagement/tip/NoSQL-database-types-explained-Graph>.
- [5] A. Williams, "NoSQL database types explained: Key-value store," 17 August 2021. [Online]. Available: <https://www.techtarget.com/searchdatamanagement/tip/NoSQL-database-types-explained-Key-value-store>.
- [6] A. Williams, "NoSQL database types explained: Column-oriented databases," 22 September 2021. [Online]. Available: <https://www.techtarget.com/searchdatamanagement/tip/NoSQL-database-types-explained-Column-oriented-databases>.
- [7] B. Anderson and B. Nicholson, "SQL vs. NoSQL Databases: What's the Difference?," 12 June 2022. [Online]. Available: <https://www.ibm.com/cloud/blog/sql-vs-nosql>.