

Table of Contents

Abstract	<i>Error! Bookmark not defined.</i>
Table of contents:	<i>Error! Bookmark not defined.</i>
What is SQL:	1
History and development:	2
What is NoSQL:	2
Key features:.....	3
Schema Free	3
Key value pairs:	3
Examples of key value databases:	4
Document:.....	4
Examples of document databases:	4
Graph:	4
Examples of graph databases:	5
Column based database:.....	5
Examples of column family based:.....	5
Vertical scaling:	5
Horizontal scaling:	6
Key features:.....	6
Cost implications:.....	6
Use cases:	6
ACID:	6
BASE:.....	7
Appdendix:	7
References:	7

What is SQL:

Structured Query Language, also known as (SQL), is one of the most used languages for writing and querying database commands through relational database management based on tables. A developer or data analyst would rely on the process of writing queries and scripts through SQL, which is split into different sections i.e. Data Manipulation Language (DML), which is essentially the same as CRUD (Create, Read,

Update and Delete) that is used in Software Engineering and Databases, but in DML this would be equivalent to 'Insert' to add new records into a table, 'Select' to read a record or all records from a table, 'Update' to edit an existing table and 'Delete' which is used to remove records from a table.

When a table is created, the table would have a column and row, where each column would reflect the data type i.e a student table with student no, name, address, email, phone number, would then have rows that link to each column with the relevant information. There are three other categories, which do different things, i.e Data Definition Language (DDL) which would define and modify the structure of the database, Data Control Language (DCL), which set's the permissions on the database and finally Transaction Control Language (TCL) which handles the transactions of the database, ensuring the data is logical, accurate, consistent and reliable [1] [2]

History and development:

Relational databases soon became the standard language for writing and querying database commands in SQL, which had gained recognition between the seventies and early eighties [2]. Relational Database Management Systems (RDBMS) has existed for over three decades. After the year 2000s, relational databases had started to face constraints. When the internet had first started gaining popularity, the need for a database was needed, to handle media, files, documents etc. In the 2000's large companies such as Google, Facebook, Amazon were developed and required a way to manage and store data [3].

Relational databases which were once highly effective at scaling requirements and being able to manage data are no longer sufficient due to the rise of large corporations such as Google, Facebook and Amazon to name a few. This resulted in new databases that could handle huge data and requests more efficiently. These tech firms had paved the way for the modern use of databases that required exceptionally scalable DBMS, that could manage data rapidly, all while being available and delivered to hundreds of millions of users worldwide [3].

What is NoSQL:

NoSQL also known as (Not Only SQL) is a type of database where it allows users to create a database without explicitly defining the database schema. NoSQL has two types of meanings. The first is where Carlo Strozzi had invented a database that does not use SQL for writing queries. The second is linked to portraying a framework that works by not using a relational structure. A NoSQL database would be structured into a

tree-like structure by using a UNIX shell as the way to communicate with the database [3].

Across the use of different NoSQL databases, which can mean that each NoSQL model is unique in its own way. NoSQL databases organise data using four main categories i.e graphs, documents, column based and key-value. The flexibility around schemas allows records or documents to have different data types and amount of fields without the need to define all the fields when creating the table [3].

Key features:

The non-relational model implies that data would be organised differently rather the one used by relational models [3].

- NoSQL databases are better utilised as they are more productive and versatile in working with systems that operate on numerous servers with different configurations [3].
- Is Open-source, unlike majority of RDBMS. This allows highly complex software to be provided at a lower price. A professional edition is available for user's that require more features or assistance [3].
- Horizontally scalable means that databases can be split across many servers, which can result in it being less expensive, allowing the process to be simplified and deploy new servers into the network, while improving flexibility and lowering expenses [3].

Schema Free

A NoSQL database does not require a schema to be defined. This lets the user to hold different types of structures in the table of the database. This results in NoSQL being much easier and effective to work with and is used by big companies than handle big volumes heterogeneous data, which does not need a schema to be designed [4].

Key value pairs:

Key-value pairs handle data as pairs, where it is stored as alphanumeric reference pairs and related to values in an isolated hash table. Therefore it is required that each key is unique and the values can be objects, integers, arrays as well as strings, which removes the constraint of having a predefined data structure. To search for a query, it is only done using the key and is restricted to exact matches, which cannot be done with the value. To accelerate the performance of running complicated queries in SQL that normally take a while, it is common for key-value pairs to be utilised because they handle data in memory. As a result, applications tend to use single key to reach data for e-commerce websites that use digital carts, storing user settings and web session records [3] [5].

Examples of key value databases:

- Redis
- MemcachedDB
- Riak

Document:

Document is another type of NoSQL database that is similarly to how key-value databases work, using XML or JavaScript Object Notation (JSON) documents. Documents provide restricted support for transactions in atomicity, consistency, isolation, and durability (ACID) and incorporate attributes with names fields and various values. Document uses a flexible layout, which lets users modify attributes during runtime without impacting the applications performance. Unlike key-value pairs, document allows data to be searched both by values and attributes and can support multiple different types of data. Additionally, an object will contain all the data required in one document, which conveys that an object will be unique, eliminating the need for object-relational mapping, which is beneficial for data integration [3] [5].

Certain documents incorporate an additional tier of aggregation, which is known as a bucket or collection which organises documents with similar data. Collections run similar to how tables would in relational databases, where each row is treated with a unique key identifier which allows for different document structures. This method enhances on the efficiency and optimises resources, persistence, security and replication at the collection level, rather than at the document level. Content management systems (CMS) i.e digital libraries, healthcare and online archives, are stored using XML document. In contrast, interactive web applications, such as social networks and blogging platforms use JSON stores, which are more popular for its quick nature [3] [5].

Examples of document databases:

- Apache
- Elasticsearch
- CouchDB

Graph:

Unlike relational databases and alternative types of NOSQL databases. Graph is appropriate for effective handling of relational data, which reduces the need for recursive and JOIN operations by leveraging the use of efficient graph methods. Graph databases store objects using key-value pairs or by documents within nodes, which are connected by relationships. Nodes and relationships can have properties. Nodes can be presented using entities such as individuals, businesses etc, just like in OOP (Object-Oriented Programming). Properties describe the way edges and nodes can be

connected from one node to another or to other properties. Graph use specific algorithms to support advanced queries, therefore graph becomes an important choice for scenarios where grasping the relationships between objects is as critical as to comprehending the object themselves [3] [5].

Examples of graph databases:

- Neo4j
- ArangoDB
- OrientDB

Column based database:

Column based family, often referred to as wide column stores, process and manage data by columns instead of rows. It indicates that data for each column is grouped together on the disk, unlike in relational databases where data for each row is organised together. This improves column stores for enhancing queries that use data aggregation, since required values are stored in the same disk block.

Columns can be formed and organised into families, which can enhance the organisation of data. To help improve consistency and improve performance, multiple releases of a value are stored and ordered chronologically by default.

Predefining column stores, makes this type of database less flexible and adaptable than key-value pairs and document stores combined. However, that being said, rows and columns can still be added dynamically during runtime.

Therefore, column stores are useful for business intelligence (BI) applications and analytics, that require analysing and aggregating large amounts of datasets. Typically, these applications concentrate on a few attributes but handle a vast number of rows. Particularly when data is distributed over numerous rows, but only a few columns are relevant [3] [5]

Examples of column family based:

- Apache
- Hadoop
- Hbase

Vertical scaling:

Vertical scaling is the process of improving the specifications of the virtual machines and power of the server, such as storage, processor and RAM. This can quickly rise costs and can be complex. It is also non-linear due to performance of servers [5].

Horizontal scaling:

Horizontal scaling refers to boosting performance within a distributed environment, used by multiple general-purpose machines. NoSQL is useful, as it helps to assist the workload and spread the data over multiple machines. Databases will allow for distribution of data automatically to new machines, when they join the cluster. Success is defined through assessing of the scalability and elasticity [6].

Key features:

Sharding divides data among multiple database instances, whereas replication creates duplicated copies of different nodes. Abadi discusses different methods of sharding, such as hash-based, range based and direct-sharding. Each come with their own challenges in terms of query routing and load balancing [7].

NoSQL uses databases such as MongoDB and Cassandra. These support horizontal scaling, which are designed to maintain consistency, provide fault tolerance across multiple nodes and handle data distribution [7].

Horizontal scaling can provide substantial improvements in terms of read and write throughput and is useful for large applications. Although, this can cause challenges in terms of data consistency and maintaining the consistency of data and managing transactions that are distributed [7].

While horizontal scaling offers great advantages. This can incur additional costs [7].

Cost implications:

- Distributed database fixes, may result in licensing fees [7].
- Network bandwidth and transfers in the cloud [7].
- Overhead in management and increased operations [7].

Use cases:

Horizontal scaling is advantageous for the following reasons:

Web applications operating at a high scale with high write/read speeds [7].

Ability to process big data and analytical tasks [7].

Social networks such as Facebook use MySQL databases to use horizontal scaling to handle billions of transactions and users across global distributed data centres [7].

ACID:

Atomicity: Each action within a transaction should be achieved effectively or none at all [8].

Consistency: Every transaction, when executed individually, ensures the database upholds its consistency and integrity [8].

Isolation: When a transaction is run, it's kept separate from all other transactions and temporary changes made are not shown to other transactions, until they are fully finalised [8].

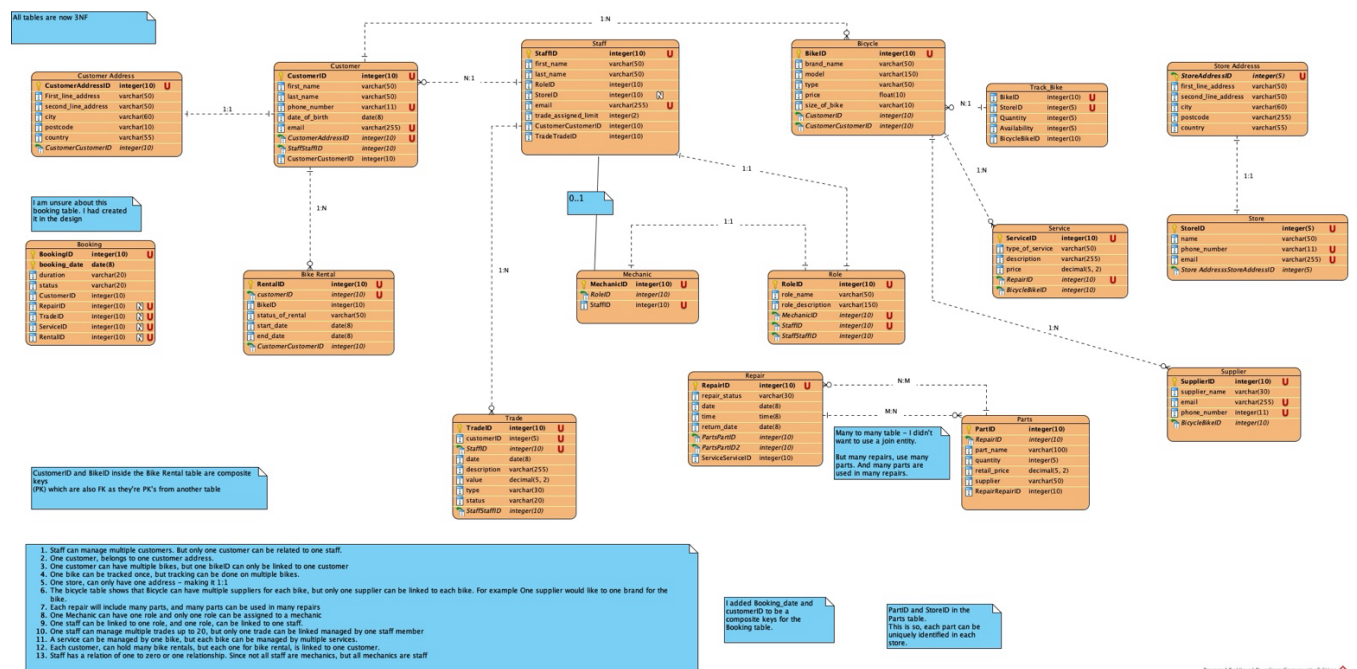
Durability: When a transaction is committed to the database, all changes are irreversible, regardless of any system failures that come up [8].

BASE:

Base analysis is where a section of the database fails, the rest of the system will still continue operating. In the event of a node failure, the system will replicate data across the remaining nodes to ensure functionality and accessibility is available. This technique is better as users can access the system will some parts are not in sync. In a soft state, data can fluctuate based on factors engagement. This can lead to data decreasing overtime; therefore it is important to retrieve and update data to maintain the system. Eventual consistency implies that data will not be immediate consistently across the system and gradually it will be consistent overtime [9]

Appendix:

My ERD design for this assessment.



References:

1. T. Capris, P. Melo, N. M. Garcia, I. M. Pires and E. Zdravevski, "Comparison of SQL and NoSQL databases with different workloads: MongoDB vs MySQL evaluation," 2022 International Conference on Data Analytics for Business and Industry (ICDABI), Sakhir, Bahrain, 2022, pp. 214-218, doi: 10.1109/ICDABI56818.2022.10041513.
2. Malik, A. Burney, and F. Ahmed, "A Comparative Study of Unstructured Data with SQL and NO-SQL Database Management Systems," Journal of Computer and Communications, vol. 08, no. 04, pp. 59–71, 2020, doi: <https://doi.org/10.4236/jcc.2020.84005>.
3. G. Mihai, "Comparison between Relational and NoSQL Databases," Annals of Dunarea de Jos University of Galati. Fascicle I. Economics and Applied Informatics, vol. 26, no. 3, pp. 38–42, Dec. 2020, doi: <https://doi.org/10.35219/eai15840409134>.
4. G. K. Kaur, S. Singla and V. Khawas, "Database Management System: A Study of Increasing Impact of NoSQL Databases," 2023 International Conference on Advanced Computing & Communication Technologies (ICACCTech), Banur, India, 2023, pp. 370-374, doi: 10.1109/ICACCTech61146.2023.00067.
5. V. F. de Oliveira, M. A. de O. Pessoa, F. Junqueira, and P. E. Miyagi, "SQL and NoSQL Databases in the Context of Industry 4.0," Machines, vol. 10, no. 1, p. 20, Dec. 2021, doi: <https://doi.org/10.3390/machines10010020>.
6. B. Sethi, S. Mishra, and P. Patnaik, "A Study of NoSQL Database," Apr. 2014. Available: <https://www.ijert.org/research/a-study-of-nosql-database-IJERTV3IS041265.pdf>
7. U. Kumar, "Article ID: IJCET_15_05_048 Cite this Article: Uday Kumar Manne, Horizontal vs. Vertical Scaling in Modern Database Systems: A Comparative Analysis of Performance and Cost Trade-Offs," International Journal of Computer Engineering and Technology (IJCET), vol. 15, no. 5, pp. 514–524, 2024, doi: <https://doi.org/10.5281/zenodo.13851050>.
8. S. Chittayasothorn, "The Misconception of Relational Database and the ACID Properties," 2022 8th International Conference on Engineering, Applied Sciences, and Technology (ICEAST), Chiang Mai, Thailand, 2022, pp. 30-33, doi: 10.1109/ICEAST55249.2022.9826318.
9. W. Khan, T. Kumar, C. Zhang, K. Raj, A. M. Roy, and B. Luo, "SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review," Big Data and Cognitive Computing, vol. 7, no. 2, p. 97, Jun. 2023, doi: <https://doi.org/10.3390/bdcc7020097>.