International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V73I3P113

Volume 73 Issue 3, 103-111, March 2025

Original Article

How Database Development Has Evolved Over the Years in the Web Development Area?

Umit Mester

Endicott College, Boston, MA.

Corresponding Author : umester@mail.endicott.edu

Received: 20 January 2025 Revised: 26 February 2025 Accepted: 17 March 2025 Published: 30 March 2025

Abstract *- Database development in web development has significantly evolved, particularly over the past few decades, driven by the internet's growth and user needs. Initially rooted in the relational database model from the 1970s, advancements were influenced by major companies like Oracle and IBM. The complexity of databases has increased due to big data requirements, leading to the creation of NoSQL databases. Open-source systems like PostgreSQL, MySQL, and MongoDB have further accelerated web application development by offering flexible, cost-effective solutions. However, challenges like big data management and security remain prevalent. Future trends indicate that databases will become smarter through AI and machine learning, with technologies like blockchain potentially reshaping the landscape. This study seeks to explore these developments in depth.*

Keywords *- AI, Databases, MySQL, NoSQL.*

1. Introduction

Database development in web development has evolved dramatically over the years, reflecting the rapid change and advancement of the digital world. This trend has accelerated with the widespread use of the internet and online platforms. Web development relies heavily on databases, making advancements in this field crucial for improving user experience and data management (Jazayeri, 2007). In historical terms, database development in the field of web development began with the relational database model in the 1970s, along with the development of basic computer science concepts, and has continued. During this time, technology companies like Oracle, IBM, and Microsoft changed the direction of the industry by focusing on databases (Brown et al., 2014). Furthermore, database development in the field of web development has grown rapidly, particularly over the last 20 years. Databases have grown in complexity as the internet has evolved, necessitating more big data management (Amra et al., 2015).

As a result, NoSQL databases were created to address this need, which has been largely met. The rise of open-source database systems has significantly impacted database development in the field of web development. Popular open-source database systems like PostgreSQL, MySQL, and MongoDB have facilitated the rapid and efficient development of web applications by providing developers with a free and adaptable database solution (Meier & Kaufmann, 2019). Database development in the field of web development has numerous perspectives to consider, as well as positive and negative aspects. For example, the need for big data management and database security issues have posed significant challenges to developers (Grulke et al., 2019).

Database development in web development is expected to accelerate and become more complex in the future. With the spread of technologies such as artificial intelligence and machine learning, databases are expected to become smarter and more automated, providing developers with significant convenience. Furthermore, new technologies such as blockchain and distributed database systems are expected to impact the field's development significantly. This study aims to investigate database development processes in web development.

1.1. Early Stages of Database Development

Large and small businesses today require information to perform their operations, compete with their competitors, and effectively maintain their assets. As a result, information is now considered a manufacturing element alongside capital, raw materials, energy, and labor. Information must be supplied on time and efficiently to serve the intended goal (Otto et al., 2011).

To carry out their tasks organizationally and effectively, businesses must receive crucial information on time and transmit it to the appropriate people and units as needed. Information systems are systems that collect relevant data, conduct operations on it, transform it into meaningful information, and convey it to the necessary persons and units in a certain order. Although the methods, processes, and tactics used to acquire and handle data in businesses have changed over time, the value of information systems remains undeniable (Cleve et al., 2015). With advances in technology and information technology, today's information systems are made up of

computer hardware, software, data sources, network technology, and human components. Among these components, data sources are those that store and access the data that will be used to generate the information required by the business (Mohan, 2013).

The desire to access data on a subject quickly and readily has necessitated the need for data to be stored in an organized manner, resulting in the rise of the phenomena of databases being interwoven into our lives in some capacity, if not all. The tools, equipment, procedures, and approaches employed now differ from those used in the past. Previously, data was physically saved on paper, in notebooks and folders (Kholmakhamatovich, 2023). In this manner, communication was once sought only when there were home phones, but as the amount of data that needed to be structured and kept grew, physically storing, protecting, and swiftly accessing this data for an extended period became a challenge. One of the primary drivers of information technology development has been the desire and need to address these challenges (Sult et al., 2013).

In today's information technology era, storing and managing massive volumes of complicated data has gotten considerably simpler. As a result, when we talk about databases nowadays, we usually refer to databases developed in an IT context. When IT became a profitable option for private firms in the 1960s, the era of computer-based database creation began. Charles Bachman developed IDS (Integrated Data Warehouse), the first general-purpose database management system, in the early 1960s. In the late 1960s, IBM created IMS (Information Management). The first commercial DBMS (Data Management System) was developed under the name System Information Management System, and its structure served as the foundation for the hierarchical data model (Klein, 2012). In 1967, the CODASYL group (Conference on Data System Languages) established COBOL language guidelines for computer users and manufacturers, which ANSI recognized. Following the successful standardization, the CODASYL Group established the DTGG (Database Job Force) and assigned the group to develop database standards. DBTG defined key database components for network data models, including language functions, schemas, data definitions, and processing languages (Coughlan & Coughlan, 2014). Edgar F. Codd invented the relational database model in 1969, which changed people's perceptions about databases. In the paradigm, the schema or logical organization of the database is separated from the actual storage of information, which has become a basic database principle. 1970: Between 1974 and 1977, two key relational database system prototypes were developed. One of these was Ingres, developed at UBC, while the other was System R, developed at IBM San Jose. Ingres Corp. developed a query language known as QUEL (Gugerli, 2012).

Microsoft SQL Server, Sybase, Wang's PACE, and Britton-Lee Pioneer In contrast, System R employed the SEQUEL query language and contributed to the development of SQL/DS, DB2, Allbase, Oracle, and Non-Stop SQL. The term relational database management system (RDBMS) also gained popularity during this time. In 1976, Peter Chen suggested a novel database paradigm known as entity relationship. This concept enabled designers to leverage data rather than logical table structures. It enabled them to focus on their apps (Ikromovna, 2023).

Structured Query Language (SQL) became the dominant query language in the 1980s, and relational database systems achieved commercial success as computer sales increased rapidly. This resulted in a considerable decrease in the popularity of network and hierarchical database models. DB2 reclaimed its position as IBM's most popular database during this period. It became a product. The launch of the IBM PC prompted the formation of numerous new database businesses and the creation of PARADOX, RBASE 5000, RIM, Dbase III and IV, and OS/2. This led to the creation of programs like Database Manager and Watcom SQL (Klonatos et al., 2014).

Because of the database crisis in the early 1990s, companies that remained in this industry were able to market complex database systems at high prices by capitalizing on the problem. Oracle Developer, PowerBuilder, and Visual Basic were among the first application development tools to hit the market. Furthermore, in the early 1990s, the Access program was introduced for Object Data Management Systems (ODBMS) (Mandapuram & Hosen, 2018). The introduction of the internet in the mid-1990s accelerated the growth of the database sector. Demand for web database connections such as front-end sites, active server sites, Java Servlets, Dream Weaver, ColdFusion, Enterprise Java Beans, and Oracle Developer 2000 increased significantly in the late 1990s, and has since returned with MySQL, Apache, and other technologies (Alzahrani, 2016).

Although the Internet industry declined in the early 2000s, database applications expanded. With the advancement and widespread usage of web-based activities, the importance of database management systems grows, resulting in the development of new applications. Microsoft, IBM, and Microsoft are currently regarded as the three leading providers of database management systems software (Cooper & James, 2009).

2. Relational Databases

Today, technology is continually advancing, and its power grows by the day. The number of locations where information is recorded increases in tandem with the computer's information storage memory. Computer-generated data has no value on its own because it only becomes meaningful when processed for a specific purpose. As a result, the ability to apply methodologies and approaches capable of handling massive amounts of data is critical (Batra & Batra, 2018).

Data mining is a technique for transforming raw data into information and meaning. When the historical history of data mining is reviewed, computerized records were first utilized in the 1950s, data warehouse procedures took their position in the world of technology in the 1960s, and the applications of relational database methods rose to prominence. In the 1970s, rule-based systems were developed, and primitive machine learning was used. Database management systems gained popularity in the 1980s and were used in a variety of fields including engineering (Lake et al., 2013). During these years, businesses established databases based on data about their products, customers, and competitors, and because these databases held a vast quantity of data, they could be accessed using SQL database query language or comparable languages. In the 1990s, problems regarding how to extract valuable information from fast-growing databases arose, and publications and studies on the subject began to appear (Meier & Kaufmann, 2019).

It is well known that advances in the IT business have numerous benefits that make them popular and desirable when they are produced. The necessity for an interface to access the database prompted the creation of the contemporary Structured Query Language (SQL). SQL is an interactive language that works with RDBMS and object-relational databases. Although Codd introduced the relational data model through his study in the IBM laboratory, the SQL language is closely related to the emergence of the relational database model. SQL's popularity is not without reason; it is the foundation of any DBMS system (Guagliardo & Libkin, 2019). Chamberlin and Boyce pioneered SQL development in the 1970s. SQL is the foundation of all DBMS systems. Because it was originally termed Structured English Query Language (SEQUEL) and was intended to process and retrieve data in IBM's prototype relational database system, later, another corporation took ownership of the abbreviation "SEQUEL" and the language. SQL quickly established itself as the primary language for RDBMS, becoming the standard language for their systems and one of their best-sellers. As SQL became more widespread, there was a greater demand for a standard everyone had to follow. In the 1980s, the American National Standards Institute (ANSI) and the International Standards Organization (ISO) worked together, and the first version of the SQL standard was developed under the name SQL-86 (SQL1) (Skoulis et al., 2015).

Codd offered a relational data model that is compatible with most data storage structure formats and then demonstrated how to interface with relational databases using the relational model. SQL seeks to attract customers who prefer to perform data retrieval operations in English rather than using mathematical symbols. One of the primary causes for the origin of SQL was the high cost of software development and the desire to enable regular users to interact with databases (Date, 2013). The software development process is not limited to the initial creation phase but rather requires constant development and maintenance of the software to meet the changing needs of the users. Because most software is used by users with little or no knowledge of computers and software, developers must simplify the processes for utilizing this software to meet the awareness level of these users. The advantages of RDBMS are that they support enterprise information systems built on low-cost hardware. RDBMS has long been one of the most essential information storage systems, and it is widely used in a variety of industrial and commercial settings. RDBMS grew in popularity and relevance until the advent of cloud computing (Garba & Abubakar, 2020).

The RDBMS landscape has shifted since the introduction of cloud computing to the software industry. RDBMS systems must provide higher scalability when dealing with massive amounts of data, yet adhering to ACID standards is a major impediment to achieving this goal. At the same time, the locking technique used by RDBMS to ensure data consistency impairs scalability, which is problematic in a distributed context. Web-based services generate more disturbance, making it challenging to achieve consistency, availability, and scalability simultaneously (Chittayasothorn, 2022). It is one of the most widely used database systems today. It comprises of tables with rows and columns. These tables are interrelated, and they are tables. As a result, to discuss a relationship in a database, at least two tables must exist, with the data in these two tables being related to one another. In this fashion, relational databases are made up of enormous files known as databases (Garba & Abubakar, 2020). The basic functionalities provided in classic relational database systems are as follows:

- Atomicity
- Consistency
- Isolation
- Durability

2. Client-Server Architecture

The 1990s saw a considerable expansion in developing and applying client-server paradigms in information technology. Client-server models are distributed computing architectures in which duties are split between clients requesting services or resources and servers offering those services or resources (Potapov et al., 2019). This concept transformed data storage, management, and access, resulting in networking, communication, and overall computing efficiency gains. Prior to the 1990s, mainframes were the dominant kind of computing, with a single centralized system processing all data and performing all operations. However, this paradigm had limitations in terms of scalability, flexibility, and speed, prompting the creation of client-server architectures. The development and popularization of these models have had far-reaching consequences for industries such as finance, healthcare, and e-commerce (Kumar, 2019). The decentralization of computer resources and enhanced flexibility and scalability in application management are two of the most significant advantages. During this time, client-server architectures

gained popularity with the release of Windows NT by Microsoft Corporation.

Furthermore, client-server models influenced the creation of peer-to-peer (P2P) networks and cloud computing as alternative models for more efficiently distributing computer resources in the field (Moltchanov, 2013). Adopting more distributed and decentralized computing architectures, such as microservices and serverless computing, has made it easier to manage complicated applications and workloads. More responsive, intelligent, and autonomous systems are expected to be built in the future as clients and servers integrate and link more effectively. The rise of AI and machine learning is also seen as a significant influence in optimizing client-server interactions and increasing the efficiency and efficacy of digital services (Wu et al., 2024).

Using server-side scripting languages is crucial for developing database-interactive web applications. The first scripting language was the PHP programming language, which was introduced in 1994. According to Huang (2019), PHP is a popular server-side scripting language for web application development due to its ease of use and interoperability with multiple databases. Another is that it revolutionized web development in 1995 with the JavaScript programming language. With this change, e-commerce websites, social media platforms, and online banking applications have taken on an entirely new dimension.

Google's search engine has changed the way information is accessed on the internet, resulting in the development of web applications that rely on server-side scripting to retrieve and display data (Britvin et al., 2022). Scripting languages provide dynamic and interactive online applications, providing users with a personalized experience. Security flaws such as SQL injection attacks and cross-site scripting have also produced significant issues with sensitive database data. With the future demand for smart and intuitive user experiences, developers intend to employ new technologies, such as human intelligence and machine learning, to produce smart and personalized web applications (Iskandar et al., 2020).

In addition, the proliferation of IoT devices necessitates the development of web apps that can connect with databases in order to process and analyze the data generated by these devices over time. Server-side scripting language is a computer language that allows a server to give dynamic information to users. Server-side scripting languages are executed on servers rather than clients. The server processes client requests and sends dynamically created responses (Serik et al., 2019). This mechanism supports dynamic activities, including database operations, user authentication, file operations, and other server-based tasks. Server-side scripting languages: PHP, Node.js, Python, Ruby, and ASP.NET. According to Wexler (2019), these languages are commonly used for server-based tasks such as database operations, session management, and email sending.

2.1. NoSQL Databases

NoSQL systems are more sophisticated than traditional databases but simpler than others. They are not based on a single model, such as an RDBMS's relational model, and each database uses a separate model based on its intended functions (Strauch et al., 2011).

NoSQL systems can handle a considerably broader set of databases. The data model in NoSQL databases is typically more constrained than in SQL databases. Carlo Strozzi invented the term "NoSQL" in 1998 to describe a simple, open-source relational database that did not support SQL queries. This usage has remained relatively limited, although it is unrelated to the evolution of NoSQL as we know it today. The word NoSQL has existed since 2009 because the market needs a phrase to characterize new technologies, and Google's Big Table and Amazon's Dynamo were followed by a surge in new databases such as Cassandra, MongoDB, and CouchDB. This effectively means that the phrase evolved to denote the transformation of non-relational databases into marketing assets (Kalid et al., 2017). This effectively means that the phrase evolved to denote the transformation of non-relational databases into marketing assets.

All NoSQL solution vendors agree on one point: "NoSQL" is not perfect, but it is appealing. NoSQL rejects software and hardware architectures more than any other language, product, or technology. The key reasons for using NoSQL databases are flexibility, speed, scalability, and high availability, hence, NoSQL has a place in production. However, it is still immature and falls short of industry standards (Kausar and Nasar, 2021). Pig, Hive, SPARQL, Mongo Query Language, Cypher, and others can be used with any of the new databases. NoSQL systems fulfil the needs of businesses that handle terabytes of data. In 2000, a typical PC probably had 10 megabytes of storage, whereas Amazon, the world's largest retail store, handles more than 42 terabytes of data. However, Facebook is currently generating 500 gigabytes of fresh data per day. The explosion of smartphones and the data they generate and consume will result in billions of new and constantly updated data feeds containing environmental, location, and other information, including video (Corbellini et al., 2017). Most of the data is kept in a dispersed environment and does not require retrieving schemas or join procedures. No policy exists for primary or foreign keys in NoSQL databases because the data is not stored in tables. It is a limitation because NoSQL does not store data in tables and does not support the concepts of foreign and primary keys. This data eliminates link operations, does not require a fixed table schema, and does not attempt to give ACID features (Garba and Abubakar, 2020).

NoSQL enables high-performance access to database programming languages from the application layer. A NoSQL user can manage data and applications through the application layer. Direct access to the database from the

application layer allows for greater flexibility while running programs (Patel and Eltaieb, 2015).

In a relational database, the database can only be managed at the database layer, not the application layer. This feature allows NoSQL significant flexibility in managing apps in addition to databases. Normalization governs the data management process in relational databases. Normalization is the practice of managing data fields and database tables in such a way as to avoid data reliance and redundancy (Lahudkar et al., 2018). Relational database management systems (RDMS) can keep basic constants, making it easier to maintain consistent data by normalizing data across several tables. Normalizing data from a large dataset might cause performance concerns owing to requests for bulk information, which can be compounded by several JOIN-type queries. However, NoSQL does not have any JOIN query capabilities. NoSQL allows users to process massive volumes of parallel data using software patterns such as MapReduce (Khan et al., 2023).

NoSQL denormalizes the data, which reduces overhead while querying the database, allowing for a considerable improvement in query time to crash a database. NoSQL avoids the AGGREGATE and JOIN operations. Since data denormalization, there has been an increase in data inconsistency, such as redundant or duplicate data. The application layer should rely on data synchronization to avoid duplication, eliminating inconsistencies during the data copy process. Another distinction between relational and NoSQL databases is the use of schemas. When working with relational databases, a schema should be developed before adding any fields. When you insert a new record into the database, the schema is added (Stanescu et al., 2016; Al Jawarneh et al., 2020).

2.2. Cloud and Distributed Databases

When investigating the evolution of cloud computing and distributed databases, the first step is to look back to the 1990s. This period can be defined as a watershed moment in the evolution of client-server models in information technology, as well as the rapid development of network technology and the broad adoption of client-server architectures across industries (Lisdorf, 2021). First, throughout the late 1980s, computer and network technology advances created the groundwork for the rise of the client-server model. The invention of protocols like TCP/IP and the growth of the internet enabled communication between client devices and server systems. This relationship cleared the way for the allocation of computer resources and data processing jobs. Simultaneously, they determined that using the client-server architecture will improve system efficiency, scale, and flexibility (Yu & Sun, 2022). The client-server approach allows for the separation of presentation-application logic and data storage layers, simplifying software application development and management. The transition to distributed computing paved the way for modern cloud computing and web-based applications. The broad use of client-server models in the 1990s had far-reaching consequences for the IT sector (Anerousis et al., 2021). Organizations in a wide range of industries, including banking, healthcare, and education, have found it an efficient approach to upgrade their IT infrastructures and enhance business operations. Enterprise systems' performance, reliability, and scalability have improved as computing workloads can be distributed among several servers and clients. Client-server architectures have also democratized access to information and services by allowing users to connect to remote servers and retrieve data via the internet. This trend toward decentralized computing has accelerated the development of e-commerce, online banking, and social networking platforms, which have revolutionized how people connect and conduct business online (Pan et al., 2021).

Netscape Communications Corporation was one of the businesses that pioneered client-server architecture and popularized the concept of web browsing. Furthermore, as is well known, Microsoft Corporation's NT server operating system and SQL Server database management system were fundamental components of many corporate IT systems, facilitating the expansion of client-server architectures in the corporate sector. The client-server approach also had a negative impact (Rittinghouse & Ransome, 2015). Although the move to distributed computing accelerated innovation in software development, network technology, and internet services, it also raised security, privacy, and data integrity issues. Nonetheless, the rise of cloud-based services and virtual platforms can be attributed to enterprises' use of client-server applications in the digital age (Robison, 2009). This has had an ongoing impact on the IT industry and society as a whole. In the 2000s, as technology advanced, so did the need for current database services. As businesses and organizations began to generate massive volumes of data that needed to be stored, managed, and retrieved efficiently, corporations were able to manage large amounts of data while providing continuous services to their users (Collen & Kulikowski, 2015). Scalability refers to the system's ability to manage workloads without compromising performance, whereas high availability assures that the system remains operational and useable even after a failure. Because of the competitive business environment, organizations began to adopt databases to store and manage data (Zielonka, 2022). First and foremost, traditional databases were built to function on a single server, limiting scalability and availability. As data volumes increase, organizations need faster data access, necessitating the development of replication-non-replication systems capable of being distributed across several servers. During this time, the company advocated for the development of the most scalable systems available to support Amazon's rapidly growing e-commerce business, and businesses that used this technology in collaboration with Amazon improved their ability to manage large amounts of data and support the growth of the user base (Yrjönkoski et al., 2019).

The introduction of cloud computing systems such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform makes a substantial contribution. The platform offers managed storage services that are scalable and highly available, allowing businesses to focus on developing apps rather than worrying about infrastructure. Cloud-based database services are now very common. Amazon.com's work has influenced how firms might create and run scalable cloud applications (Sampathkumar, 2015). As corporations seek new ways to secure data integrity and availability in distributed systems, emerging technologies such as blockchain and decentralized storage are gaining traction. Serverless computing solutions like AWS Lambda and Google Cloud Functions make installing and managing database services easy, allowing businesses to expand and construct highly scalable applications without concern for the underlying structure (Targowski, 2016). Database services also play an important part in allowing businesses to handle enormous amounts of data, expand their user base, and deliver reliable client services. These services have had a big impact on the industry because they enable businesses to create strong and scalable applications that can fulfill the demands of the new digital company (Pan et al., 2021).

3. Modern Trends in Database Development

The microservices architecture and the concept of multilingual persistence, which uses multiple data sources to satisfy individual requirements, have altered the way software is developed and utilized. This method enables organizations to construct smaller, more focused services that can be designed, executed, and assessed separately, making services simpler, faster, and more repeatable. In truth, the concept of microservices architecture originated in the early 2000s, when organizations such as Amazon, Netflix, and Google began to shift away from monolithic systems and toward a more modular and decentralized approach. Baškarada et al. (2020) cited the need to address the limits of traditional software development, including long release timelines, high maintenance costs, and scalability concerns.

Microservices architecture leads to considerable speed, time to market, and scalability advantages. Breaking complicated systems into smaller, more manageable components allows teams to operate independently and release updates more frequently without affecting the overall application. As technology advances, the future of microservices architecture and multilingual persistence appears to be jeopardized (Villamizar et al., 2017). There are projections that the need for these concepts will grow as firms prioritize power, flexibility, and dependability. Future development priorities include increasing service integration, maintaining and monitoring the paradigm, and simplifying its deployment and management (Luntovskyy & Shubyn, 2020). While there are certain hurdles and trade-offs to consider, many companies find that the benefits outweigh the drawbacks. Serverless storage, which provides predictable and cost-effective storage, has emerged in the new digital age, and it is ideal for high-level applications and multi-application systems. Major cloud providers, like Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure, provide a variety of services centered on browserless and automated repositories (Guo et al., 2016). This technology seeks to improve the flexibility and cost of standard products while increasing data storage and control. In this context, serverless and automatically curated databases appeal to a diverse spectrum of users, from huge corporations to small and medium-sized organizations.

Furthermore, because of its low cost, firms can boost their competitiveness and engagement (Srinivasan et al., 2018). However, there are issues about data security and the possibility of data loss when employing this technology. Furthermore, some of the current trends include mixing new technologies like AI and machine learning with unsupervised databases. However, due to the intricacy of this technology, it must be installed and managed correctly (Kelley et al., 2020). Modern query languages, such as GraphQL, are a significant advancement in web development. These technologies attempt to make data retrieval and communication more efficient and effective. Initially, data transmission was done using RESTful APIs and Soap, however, these techniques had issues, including duplicate data points and inaccurate data returns (Gözneli, 2020). This made data queries by developers more complicated. The introduction of GraphQL provides a solution to these concerns. Facebook created GraphQL in 2012 as a language for data analysis and communication. GraphQL enables clients to access data in a single format. Understanding GraphQL and other query languages is crucial for current web development, particularly among influential individuals (Niswwar et al., 2024). GraphQL and other new query languages have a significant impact on web development. These technologies improve data communication's efficiency and effectiveness. GraphQL enables clients to query data in a more structured manner. GraphQL and comparable query languages provide developers with both power and flexibility while reducing duplicate data retrieval and improving application speed. Some academics think RESTful APIs are easier to use (Ylisiurunen, 2019).

4. Conclusion

Database development in web development has significantly evolved, primarily due to the internet's growth, enhancing user experience and data management. Originating with the relational database model in the 1970s, technological giants like Oracle and IBM shaped the industry. In recent decades, the complexity of databases has increased, leading to the creation of NoSQL databases and the rise of open-source systems like PostgreSQL and MongoDB, which support rapid web application development. Challenges remain, particularly in big data management and security. Future advancements, driven by AI, machine learning, and blockchain technologies, are expected to enhance database capabilities and automation further. In today's business environment, information is essential for operations, competition, and asset management, positioning it as a key manufacturing

element alongside capital and labor. Timely and efficient information delivery is crucial, necessitating effective information systems that collect, process, and convey data to relevant parties. Although the methods for handling data have evolved, the importance of these systems remains. Modern information systems comprise hardware, software, data sources, networks, and human components, with organized data storage becoming increasingly vital as businesses seek quick access to information.

References

[1] Isam Mashhour Al Jawarneh et al., "Efficient QoS-Aware Spatial Join Processing for Scalable NoSQL Storage Frameworks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2437-2449, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Hibatullah Alzahrani, "Evolution of Object-oriented Database Systems," *Global Journal of Computer Science and Technology*, vol. 16, no. 3, pp. 1-5, 2016. [Google Scholar] [Publisher Link]

[3] Nadir K. Amra et al., *Modernizing IBM i Applications from the Database up to the User Interface and Everything in Between*, IBM Redbooks, pp. 1-720, 2015. [Google Scholar] [Publisher Link]

[4] Nikos Anerousis et al., "The Origin and Evolution of Open Programmable Networks and SDN," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1956-1971, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Sasa Baškarada, Vivian Nguyen, and Andy Koronios, "Architecting Microservices: Practical Opportunities and Challenges," *Journal of Computer Information Systems,* vol. 60, no. 5, pp. 428-436, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[6] Rahul Batra, "A History of SQL and Relational Databases," *SQL Primer*, pp. 183-187, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[7] A. Britvin, J. Alrawashdeh, and R. Tkachuk, "Client-server System for Parsing Data from Web Pages," *Advances in Cyber-Physical Systems*, vol. 7, no. 1, pp. 8-14, 2022. [Google Scholar] [Publisher Link]

[8] Kyle Brown et al., *Modern Web Development with IBM WebSphere: Developing, Deploying, and Managing Mobile and Multi-Platform Apps*, Pearson Education, pp. 1-384, 2014. [Google Scholar] [Publisher Link]

[9] Suphamit Chittayasothorn, "The Misconception of Relational Database and the ACID Properties," *2022 8th International Conference on Engineering, Applied Sciences, and Technology (ICEAST)*, Chiang Mai, Thailand, pp. 30-33, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[10] Anthony Cleve et al., "Understanding Database Schema Evolution: A Case Study," *Science of Computer Programming*, vol. 97, no. 1, pp. 113-121, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] Morris F. Collen, and Casimir A. Kulikowski, "The Development of Digital Computers," *The History of Medical Informatics in the United States*, pp. 3-73, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[12] Joshua Cooper, and Anne James, "Challenges for Database Management in the İnternet of Things," *IETE Technical Review*, vol. 26, no. 5, pp. 320-329, 2009. [Google Scholar] [Publisher Link]

[13] Alejandro Corbellini et al., "Persisting Big-data: The NoSQL Landscape," *Information Systems*, vol. 63, pp. 1-23, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[14] Michael Coughlan et al., *Introduction to COBOL*, Beginning COBOL for Programmers, pp. 1-15, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[15] C.J. Date, *Relational Theory for Computer Professionals: What Relational Databases Are Really All About*, O'Reilly Media, pp. 1-284, 2013. [Google Scholar] [Publisher Link]

[16] Musa Garba, and H. Abubakar, "A Comparison of NoSQL and Relational Database Management Systems (RDBMS)," *Kasu Journal of Mathematical Sciences*, vol. 1, no. 2, pp. 61-69, 2020. [Google Scholar]

[17] Berke Gözneli, "*Identification and Evaluation of a Process for Transitioning from REST APIs to GraphQL APIs in the Context of Microservices Architecture*," Master's Thesis, Technische Universität München, München, 2020. [Google Scholar]

[18] Christopher M. Grulke et al., "EPA's DSSTox Database: History of Development of a Curated Chemistry Resource Supporting Computational Toxicology Research," *Computational Toxicology*, vol. 12, pp. 1-15, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Paolo Guagliardo, and Leonid Libkin, "On the Codd Semantics of SQL Nulls," *Information Systems*, vol. 86, pp. 46-60, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[20] David Gugerli, "The World as Database: On the Relation of Software Development, Query Methods, and İnterpretative İndependence," *Information & Culture*, vol. 47, no. 3, pp. 288-311, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[21] Dong Guo et al., "Microservices Architecture based Cloudware Deployment Platform for Service Computing," *2016 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ching-Yu Huang, "Integrated Curriculum of Multi-tier Client/server Web-based Database Applications," *International Journal of Information and Education Technology*, vol. 9, no. 5, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[23] Akhmedova Zulhumor Ikromovna, "SQL (Structured Query Language) Capabılıtıes Of The Statıstıcal Database Language," *Multidisciplinary Journal of Science and Technology*, vol. 3, no. 5, pp. 274-280, 2023. [Google Scholar] [Publisher Link]

[24] Taufan Fadhilah Iskandar et al., "Comparison between Client-side and Server-side Rendering in the Web Development," *IOP Conference Series: Materials Science and Engineering*, vol. 801, pp. 1-7, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[25] Mehdi Jazayeri, "Some Trends in Web Application Development," *Future of Software Engineering (FOSE'07)*, pp. 199-213, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[26] Sultana Kalid et al., "Big-data NoSQL Databases: A Comparison and Analysis of "Big-Table", "DynamoDB", and "Cassandra"," *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pp. 89-93, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[27] Mohammad A. Kausar, and Mohammad Nasar, "SQL Versus NoSQL Databases to Assess Their Appropriateness for Big Data Application," *Recent Advances in Computer Science and Communications*, vol. 14, no. 4, pp. 1098-1108, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[28] Robert Kelley et al., "Choosing the Right Compute Resources in the Cloud: An Analysis of the Compute Services Offered by Amazon, Microsoft and Google," *2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, Chongqing, China, pp. 214-223, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[29] Muhammad Zohaib Khan et al., "Comparative Case Study: An Evaluation of Performance Computation Between SQL And NoSQL Database," *Journal of Software Engineering*, vol. 1, no. 2, pp. 14-23, 2023. [Google Scholar]

[30] Ikromov Khusan Kholmakhamatovich, "Historıcal Context Of Development Of Informatıon Systems and Database Management," *International Journal of Pedagogics*, vol. 3, no. 11, pp. 119-123, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[31] Barbara Klein, *An Introduction to IMS: Your Complete Guide to IBM Information Management System*, IBM Press, 2012. [Google Scholar] [Publisher Link]

[32] Ioannis Klonatos et al., "Building Efficient Query Engines in a High-level Language," *Proceedings of the VLDB Endowment*, vol. 7, no. 10, pp. 853-864, 2014. [Google Scholar] [Publisher Link]

[33] Santosh Kumar, "A Review on Client-Server based Applications and Research Opportunity," *International Journal of Recent Scientific Research*, vol. 10, no. 7, pp. 33857-3386, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[34] Pratik Lahudkar et al., "NoSQL Database-Google's Firebase: A Review," *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 7, no. 3, pp. 1-8, 2018. [Google Scholar] [Publisher Link]

[35] P. Lake, and P. Crowther, *A History of Databases. Concise Guide to Databases: A Practical Introduction*, Springer-Verlag London, pp. 1-316, 2013. [Google Scholar]

[36] Anders Lisdorf, *Cloud Computing Basics*, Apress, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[37] Andriy Luntovskyy, and Bohdan Shubyn, "Highly-distributed Systems based on Micro-services and Their Construction Paradigms," *2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[38] Mounika Mandapuram, and Faruk Hosen, "The Object-oriented Database Management System Versus the Relational Database Management System: A Comparison," *Global Disclosure of Economics and Business*, vol. 7, no. 2, pp. 89-96, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[39] Andreas Meier, and Michael Kaufmann, *SQL & NoSQL Databases*, Springer, 2019. [Google Scholar] [Publisher Link]

[40] C. Mohan, "History Repeats İtself: Sensible and NonsenSQL Aspects of the NoSQL Hoopla," *Proceedings of the 16th International Conference on Extending Database Technology*, pp. 11-16, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[41] Dmitri Moltchanov, "Client/server and Peer-to-peer Models: Basic Concepts," *Department of Communications Engineering, Tampere University of Technology*, pp. 1-41, 2013. [Google Scholar] [Publisher Link]

[42] Muhammad Niswar et al., "Performance Evaluation of Microservices Communication with REST, GraphQL, and gRPC," *International Journal of Electronics and Telecommunication*, vol. 70, no. 2, pp. 1-8, 2024. [Google Scholar] [Publisher Link]

[43] Boris Otto, Yang W. Lee, and Ismael Caballero, "Information and Data Quality in Business Networking: A Key Concept for Enterprises in İts Early Stages of Development," *Electronic Markets*, vol. 21, pp. 83-97, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[44] Jianping Pan et al., "Network for AI and AI for Network: Challenges and Opportunities for Learning-oriented Networks," *IEEE Network*, vol. 35, no. 6, pp. 270-277, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[45] Tejal Patel, and Tarik Eltaieb, "Relational Database vs NoSQL," *Journal of Multidisciplinary Engineering Science and Technology*, vol. 2, no. 4, pp. 691-695, 2015. [Google Scholar] [Publisher Link]

[46] V.I. Potapov et al., "Reliability in the Model of an İnformation System with Client-server Architecture," *Journal of Physics: Conference Series*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[47] John W. Rittinghouse, and James F. Ransome, *Cloud Computing: History and Evolution*, Encyclopedia of Information Systems and Technology-Two Volume Set, pp. 1-15, 2015. [Google Scholar] [Publisher Link]

[48] William Jeremy Robison, "Free at What Cost: Cloud Computing Privacy Under the Stored Communications Act," *Georgetown Law Journal*, vol. 98, no. 4, pp. 1-45, 2010. [Google Scholar] [Publisher Link]

[49] Rajakumar Sampathkumar, *Disruptive Cloud Computing and IT: Cloud Computing SIMPLIFIED for Every IT Professional*, Xlibris Corporation, pp. 1-412, 2015. [Google Scholar] [Publisher Link]

[50] Meruert Serik, Meiramgul Mukhambetova, and Alibek Yeskermessuly, "Improving the Content of a Client-server Technology Training Course: Set Up and Collaborative İmplementation of Local and Cloud-based Remote Servers," *International Journal of Emerging Technologies in Learning*, vol. 14, no. 21, pp. 191-204, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[51] Ioannis Skoulis, Panos Vassiliadis, and Apostolos V. Zarras, “Growing up with Stability: How Open-source Relational Databases Evolve,” *Information Systems*, vol. 53, pp. 363-385, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[52] Vitthal Srinivasan, Janani Ravi, and Judy Raj, *Google Cloud Platform for Architects: Design and Manage Powerful Cloud Solutions*, Packt Publishing Ltd, pp. 1-372, 2018. [Google Scholar] [Publisher Link]

[53] Liana Stanescu, Marius Brezovan, and Dumitru Dan Burdescu, “Automatic Mapping of MySQL Databases to NoSQL MongoDB,” *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Gdansk, Poland, pp. 837-840, 2016. [Google Scholar] [Publisher Link]

[54] Christof Strauch, Ultra-Large Scale Sites, and Walter Kriha, “NoSQL Databases,” *Lecture Notes, Stuttgart Media University*, vol. 20, no. 24, 2011. [Google Scholar]

[55] Leslie Sult et al., “A New Approach to Online Database İnstruction: Developing the Guide on the Side,” *Reference Services Review*, vol. 41, no. 1, pp. 125-133, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[56] Andrew Targowski, *The History, Present State, and Future of Information Technology*, Informing Science Press, pp. 1-399, 2016. [Google Scholar] [Publisher Link]

[57] Mario Villamizar et al., “Cost Comparison of Running Web Applications in the Cloud using Monolithic, Microservice, and AWS Lambda Architectures,” *Service Oriented Computing and Applications*, vol. 11, pp. 233-247, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[58] Jonathan Wexler, *Get Programming with Node. js*, Manning, pp. 1-480, 2019. [Google Scholar] [Publisher Link]

[59] Liupengfei Wu, Weisheng Lu, and Chen Chen, “Strengths and Weaknesses of Client-server and Peer-to-peer Network models in Construction Projects,” *International Journal of Construction Management*, vol. 24, no. 12, pp. 1349-1363, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[60] Markus Ylisiurunen, “*GraphQL Query Language's Feasibility in a Microservice Architecture*,” Bachelor's Thesis, pp. 1-29, 2019. [Publisher Link]

[61] Katarina Yrjönkoski et al., “Software Business: A Short History and Trends for the Future,” *Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications*, 2019. [Google Scholar] [Publisher Link]

[62] Xiaofei Yu, and Fengyou Sun, “A Study on Telecommunication Network, Internet, and Internet of Things,” *2022 10th International Conference on Information Systems and Computing Technology (ISCTech)*, pp. 673-680, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[63] Ryan Zielonka, *Technology Innovation and Digital Revolution: Adoption and Diffusion of Digital Network Platforms, 1995-2001, with Implications for Future Development*, University of Washington Libraries, pp. 1-452, 2022. [Google Scholar] [Publisher Link]