big data and cognitive computing

MDPI

Systematic Review

SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review

Wisal Khan [1], Teerath Kumar [2], Cheng Zhang [1,*], Kislay Raj [2], Arunabha M. Roy [3] and Bin Luo [1,*]

[1] School of Computer and Technology, Anhui University, Hefei 230039, China
[2] School of Computing, Dublin City University, SFI for Research Training in Artificial Intelligence, D02 FX65 Dublin, Ireland
[3] Aerospace Engineering Department, University of Michigan, Ann Arbor, MI 48109, USA
* Correspondence: cheng.zhang@ahu.edu.cn (C.Z.); luobin@ahu.edu.cn (B.L.)

Abstract: The competent software architecture plays a crucial role in the difficult task of big data processing for SQL and NoSQL databases. SQL databases were created to organize data and allow for horizontal expansion. NoSQL databases, on the other hand, support horizontal scalability and can efficiently process large amounts of unstructured data. Organizational needs determine which paradigm is appropriate, yet selecting the best option is not always easy. Differences in database design are what set SQL and NoSQL databases apart. Each NoSQL database type also consistently employs a mixed-model approach. Therefore, it is challenging for cloud users to transfer their data among different cloud storage services (CSPs). There are several different paradigms being monitored by the various cloud platforms (IaaS, PaaS, SaaS, and DBaaS). The purpose of this SLR is to examine the articles that address cloud data portability and interoperability, as well as the software architectures of SQL and NoSQL databases. Numerous studies comparing the capabilities of SQL and NoSQL of databases, particularly Oracle RDBMS and NoSQL Document Database (MongoDB), in terms of scale, performance, availability, consistency, and sharding, were presented as part of the state of the art. Research indicates that NoSQL databases, with their specifically tailored structures, may be the best option for big data analytics, while SQL databases are best suited for online transaction processing (OLTP) purposes.

Keywords: big data; SQL and NoSQL databases; MapReduce; aggregation; ACID; BASE; DBaaS

Citation: Khan, W.; Kumar, T.; Zhang, C.; Raj, K.; Roy, A.M.; Luo, B. SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review. *Big Data Cogn. Comput.* **2023**, *7*, 97. https://doi.org/10.3390/bdcc7020097

Academic Editors: Min Chen and Carson K. Leung

Received: 13 January 2023
Revised: 6 May 2023
Accepted: 8 May 2023
Published: 12 May 2023

1. Introduction

The architecture of a particular software application addresses non-functional characteristics, such as dependability, usability, scalability, performance, interoperability, portability, adaptability, and data sharding. There are always trade-offs among the set of quality attributes, and a software architect faces the difficult task of balancing them. Big data systems [1] are intrinsically distributed. Data availability and consistency difficulties are produced by data sharding and replication within vast data systems. Due to the increasing expansion of data applications, database technologies have experienced substantial variations. Over the course of more than a decade, NoSQL databases have grown exponentially, although classic database automation has persisted. Traditional mock-up forces a rigid schema structure, which leads to scaling obscurity and inhibits data modification across clusters. In contrast, NoSQL databases support simple prototypes. Principal properties of NoSQL database designs include:

- Schema-less structure
- Permitting data representations to grow effectively and dynamically
- Scaling horizontally, by data replication collections and sharding, over massive clusters.

In recent years, numerous organizations have accumulated vast volumes of data, which relational databases cannot process efficiently. In the past four decades, there has been a huge rise in the use of relational databases. They adhere to the ACID (availability, consistency, isolation, and durability) attribute and are designed for structured data. While "Big Data" comprises tools and technology that manage massive amounts of data at any scale, these tools and technologies are scalable. Big data comprises the 5Vs (volume, velocity, variety, veracity, and value) and a massive amount of unstructured data with a diverse nature. Numerous frameworks, including Hadoop/MapReduce, Spark, Flink, and Samza, are utilized for large data processing [1].

The subject of SQL query performance and optimization for enterprise, production, parallel databases, and big data [2] has received increased attention in recent years. Ineffective and non-optimized queries may consume system and server resources, resulting in database locking and data loss issues. Information mining entails extracting the facts and logical correlation structure from the original data set, as opposed to the information itself. Query optimization refers to selecting the optimal query execution strategy with minimal cost and system resource consumption. The data mining algorithms conduct in-depth and extensive database queries to extract patterns and knowledge from comprehensive data [3]. Alternative methodologies, such as XML and object databases, have never achieved the same level of popularity as RDBMS technology.

Over the past decade, science and online vendors have questioned the "one-size-fits-all" nature of data shop technology. This line of thinking resulted in the development of a new alternative database system known as NoSQL, which stands for "Not only SQL." NoSQL describes web developers' usage of non-relational databases [4]. In 1998, the term NoSQL [5] was used for the first time, and the non-relational databases conference in San Francisco drew greater attention to it. Figure 1 describes the key aspects of NoSQL databases.

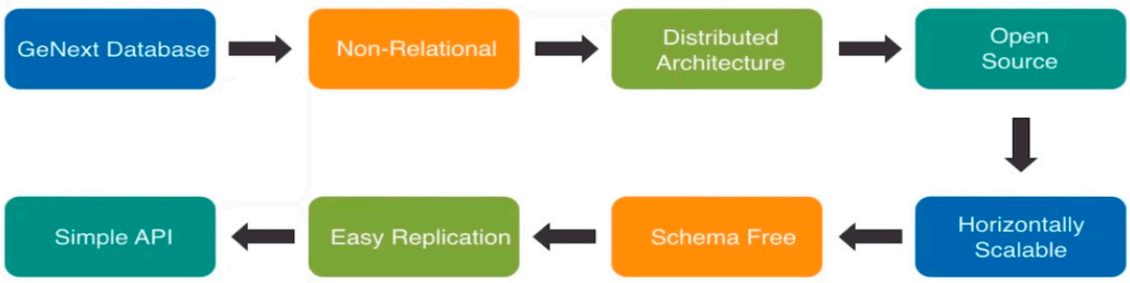

Figure 1. NoSQL—The essential features.

Eric Brewer presented the CAP (consistency, availability, and partition tolerance) theory [6,7]. The main characteristics of CAP theory are given in Table 1.

Table 1. CAP Theory.

Consistency	Availability	Partition Tolerance
• Consistency means that the data in the database remain consistent after the execution of an operation. • For example, after an update operation, all clients see the same data.	• Availability means that the system will not have downtime (100% service uptime guaranteed). • Every node (if not failed) always executes the query.	• Partition tolerance means that the system continues to function even when the communication among servers is unreliable. • The servers may be partitioned into multiple groups that cannot communicate with one another.

Theoretically, it is impossible to fulfil all three requirements. Therefore, CAP supplies the essential need for a distributed system to follow two of the three elements. Hence, all the current NoSQL databases support the various combinations of C, A, and P of the CAP theory, as described in Figure 2.

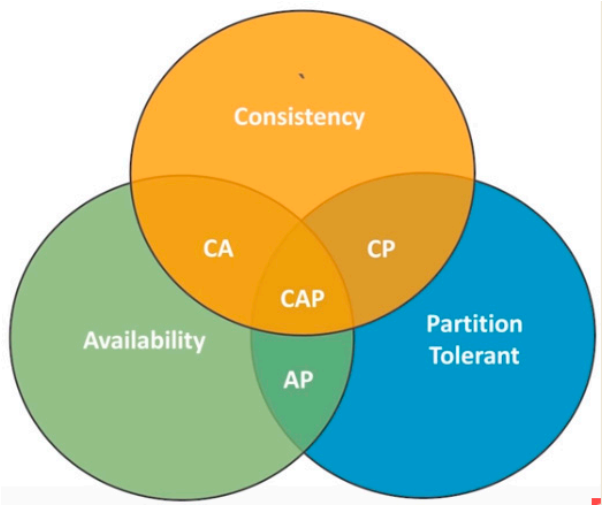

Figure 2. CAP Combinations.

The following are the advantages of NoSQL databases:

- Volume: Data at rest—Terabytes to exabytes of existing data to process.
- Velocity: Data in motion—Streaming data, milliseconds to seconds to respond.
- Variability: Data in many forms—structured, unstructured, text, etc.
- Veracity: Data in doubt—uncertainty due to latency, deception, ambiguities, etc.
- Not built on tables and does not employ SQL to manipulate data.
- Schema comprises key-value, document, columnar, graph, etc.
- Alternative to traditional relational databases.
- Database to handle unstructured, messy, and unpredictable data.
- Helpful for working with large sets of distributed data.

NoSQL databases differ from relational databases and have their own models and architectures. When storing NoSQL databases in the cloud, caution is necessary due to the diversity of these databases and the need for interoperability, portability, and security measures. Cloud service providers (CSP) [8–12] offer scalability, availability, and privacy, but it is important to encrypt sensitive user data before granting access to the CSP. This can be challenging due to the varied nature of NoSQL databases. Database as a Service (DBaaS) [13,14] is a cloud platform that converts traditional architectures into cloud architectures.

The main contributions of this SLR are the following:

- This SLR is related to the SQL and NoSQL database architecture assessments, scaling capabilities, and performance analysis, particularly Oracle RDBMS and NoSQL Document Database (MongoDB). In addition, data movement among various databases across multiple cloud platforms is explored.
- A total of 142 studies have been analyzed to accomplish the research goals mentioned earlier.
- This article identifies the research gaps in the associated architectures and their causes.

1.1. State of the Problem

The proliferation of big data has led to the need for scalable systems that can effectively process massive amounts of data. Relational databases, which are based on SQL,

can manage structured, semi-structured, and unstructured data, to a certain extent, but have limitations in terms of scalability. On the other hand, NoSQL databases, which follow the BASE property and can expand their storage capacity horizontally, are better suited to managing vast volumes of data and adapting to changes in data type and structure.

There are four types of NoSQL databases—key-value, document, column, and graph databases—each with their own distinct features and applications. For example, a NoSQL graph database stores information in nodes rather than tables and stores associations (joins) among nodes, which requires constant execution time. This gives it an advantage over SQL databases when handling highly interconnected data.

MongoDB is an example of a NoSQL database that has effectively managed huge data due to its excellent scalability characteristics. However, converting from OLTP databases to MongoDB can cause issues, such as unique indices, composite keys, data inconsistency, and data duplications due to the complexity of their schema.

Another major concern in cloud storage transfers is the protection of users' private information. Current cloud service provider (CSP) designs are developed without interoperability and portability issues being taken into consideration, making it difficult to offer and create a unified cloud solution for NoSQL models.

Finally, in this SLR, the state-of-the-art security techniques and policies for NoSQL databases are examined in detail.

1.2. Method

This SLR follows the PRISMA guidelines and aims to synthesize high-quality primary studies based on evidence associated with specific research questions. SLRs have been widely used in software engineering research, and our study focuses on SQL and NoSQL database architecture assessments and performance evaluations, as well as data portability and interoperability across multiple cloud platforms. The characteristics that differentiate our study from existing ones are its systemic data collection process, comprehensive list of covered studies, focused study scope, and its detailed classification and analysis of selected studies. The remainder of the paper is organized as follows: Section 2 outlines the research questions, search strings, inclusion/exclusion criteria, data extraction, and classification, while Section 3 presents the results of the selected papers. The discussion and research gaps are described in Section 4, and the conclusion and future work are summarized in Section 5

2. Objectives and Research Questions

The purpose, focus, and objectives of the current SLR:

1. Address the existing SQL and NoSQL document approaches and techniques by considering big data processing.
2. Perform a systematic literature review associated with SQL and NoSQL databases.
3. Review selected study subsets in depth.
4. Assess the strength and weaknesses of SQL and NoSQL databases on the basis of the evidence collected and analyzed from these studies.
5. Highlight the research gap in the area.
6. Identify future research directions.
7. Formulate the following research questions to achieve the main objective of our study:
    - Considering big data (structured and unstructured data): What is the need for NoSQL?
    - Why does the NoSQL database follow the BASE property instead of the SQL database ACID property?
    - Does DBaaS tackle data interoperability and portability efficiently in various NoSQL databases?

2.1. Search Criteria

The search criteria for primary studies involve identifying and collecting the literature that meets the inclusion and exclusion criteria. To accomplish this, various search techniques were used, such as an electronic database search, manual search, and snowballing, along with searching associated journals and conference proceedings. For our SLR, we followed a protocol proposed by [15,16] and the seven phases investigated in [17]. Figure 3 illustrates the selection of associated studies.

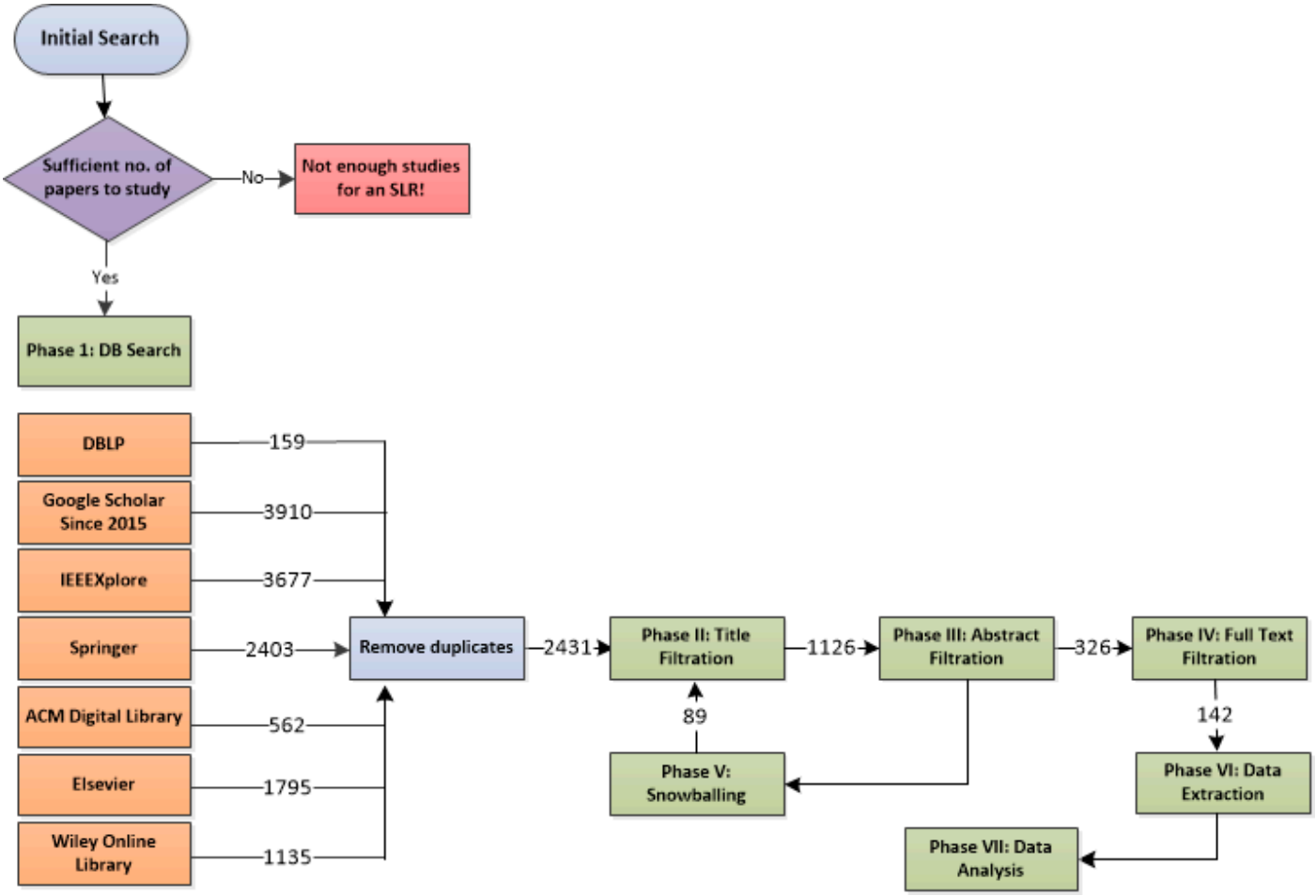

Figure 3. Process flow—Selected studies.

First, we performed the database search suggested by [18]. The search strings mentioned in the search strategy section were used for categorizing and classifying, i.e., tools, methods, and framework.

2.1.1. Search Resources

In Phase I, we derived a set of associated search strings. We used the derived set of strings to find the related papers. The largest databases were selected for finding the associated articles:

- DBLP
- IEEExplore Digital Library (ieeexplore.ieee.org)
- Google Scholar
- ACM Digital Library (dl.acm.org)
- Springer (Springerlink.com)
- Elsevier (sciencedirect.com)
- Wiley Online Library (onlinelibrary.wiley.com)

We found 13,000 papers during the search process using article titles, abstracts, and keywords. There were 2431 papers that met the inclusion and exclusion criteria after removing duplicates from Phase I. Backward snowballing in Phase V led to 89 articles being selected after repeatedly validating the inclusion/exclusion criteria from Phases II through IV.

2.1.2. Search Strategy

Various combinations of search strings were created. The devised search strings were run on the mentioned search resources to identify the associated literature:

- "SQL and NoSQL"
- "SQL or NoSQL"
- "Relational Database and Document Database"
- "Relational Database or Document Database"
- "Relational Database and NoSQL Document Database"
    - "Relational Databases and MongoDB"
- "Relational Databases or MongoDB"
- "Oracle and MongoDB Comparison"
- "SQL and NoSQL Database Comparisons"
- "Advantages of MongoDB over RDBMS"
- "Relational and Non-relational Databases"
- "Cloud Data Portability and Interoperability"

2.2. Selection Process and Criteria

We included all associated papers by applying inclusion/exclusion criteria. In Phase 1, we evaluated the quality and characteristics of the papers according to our research questions and finalized the selected list. Our research papers are organized by source as follows:

- Step1: Total number of documents based on:
    1. Papers Titles.
        i. Papers Abstract.
            a. Associated papers full reading
                (1) Check the quality and impact of related papers
                    ✓ Check the article in the catalogue to avoid repetition
                    ✓ Add item to the finalized papers catalogue
                (2) Manual search and snowballing
                (3) Repeat the entire process, go to Step1

Table 2 depicts the number of associated papers related to the points mentioned earlier after each phase filtration.

Table 2. Paper selection process and criteria.

Search Source	Based on Title (Phase II)	Based on Abstract (Phase III)	Based on Full Reading of Papers (Phase IV)
Total No. of Papers	1126	326	142

Table 2 shows that 1126 papers were initially selected on the basis of their titles. After reviewing their abstracts, the first author chose 326 articles, and 142 papers were selected for full reading. Of these, 89 were chosen via snowballing. Most of the papers were based on empirical studies, which include:

- Research purpose
- Associated literature and supported theories
- Hypothesis measurement
- Proposed method, design, approach, dimension, and data collection
- Data result analysis
- Conclusion

2.2.1. Inclusion Criteria

The following types of papers were included according to our research questions:

- IC1: related SLRs and survey papers
- IC2: new proposed techniques and approaches relevant to our proposed SLR
- IC3: effective research methods presented in the proposed study

To increase the reliability and efficiency of the SLR, the corresponding author reviewed and investigated the impact and methodology of the included papers.

2.2.2. Exclusion Criteria

The research papers were excluded on the basis of the following criteria:

- EC1: papers not related to the mentioned domain
- EC2: irrelevant papers
- EC3: some papers based on the title and abstract
- EC4: non-peer-reviewed materials and papers
- EC4: articles not written in English and duplicated articles

To reduce the threat to the reliability of the SLR, the co-authors (2nd and 3rd) rechecked the excluded papers according to the checklist of exclusion criteria.

The latest version of the proposed SLR will be used when it is published in the journal or presented at a conference. The document quality was checked by the corresponding-authors and co-authors.

2.3. Data Collection and Extraction

After collecting the 142 associated studies, two reviewers reviewed them and extracted valuable data that satisfied the research questions [19]. The following data were obtained from each selected paper:

- Title of paper
- Abstract of paper
- Paper source (journal or conference)
- Publication year
- Paper classification (type, scope)
- Relatedness to the proposed SLR
- Proposed SLR objectives and research question issues
- Paper summary and method

We selected most of the empirical articles. The empirical studies consisted of the following categories of the papers' evaluations, discussion assessments, experiments, and reviews of existing techniques (SQL and NoSQL).

2.4. Data Analysis and Classification

Furthermore, information about each of the extracted data was arranged and tabulated in accordance with the research questions as follows:

1. Considering big data (structured and unstructured data): What is the need for NoSQL?
2. Why does the NoSQL database follow the BASE property instead of the SQL database ACID property?

3. Does DBaaS tackle data interoperability and portability efficiently in various NoSQL databases?

During analysis, we grouped all extracted strategies into categories and summarized our findings into three groups: research methods, research process phases, and evaluation.

2.5. Validity Threats and Evaluations

Threats to the SLR process, as suggested by [11], should be consistently evaluated. These threats were categorized into different groups, including descriptive validity, theoretical validity, interpretive validity, generalizability validity, and repeatability [20,21]. Professor Zhang Cheng of Anhui University analyzed the SLR validity checks, and we incorporated his suggested changes into the protocol.

3. Results

In this section, we summarize our chosen research by publication year, paper genre, and number of chosen studies from a particular digital library (presented in full in Table A1 (Appendix A). Based on the selection procedure and criteria, most of the empirical research articles were chosen. According to the research literature, the researcher utilized both types of databases for their recommended methodologies and studies. In addition to actual investigations, we also discovered survey articles concerning SQL and NoSQL databases. Following the associated review selection procedure, we categorized the selected studies and publications into three main categories. Figure 4 illustrates the category pie chart for studies.

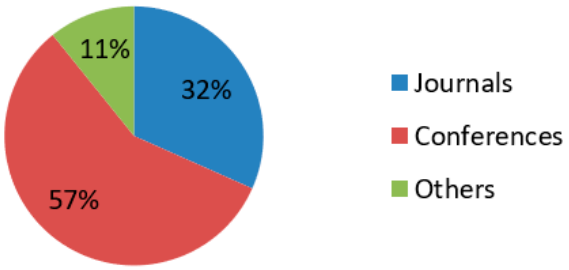

Figure 4. Categorization of the selected studies.

The publication years of the selected SQL- and NoSQL-database-related studies range between 2000 and 2022. Figure 5 depicts the number of papers published annually, showing that NoSQL databases gained greater interest following 2008.

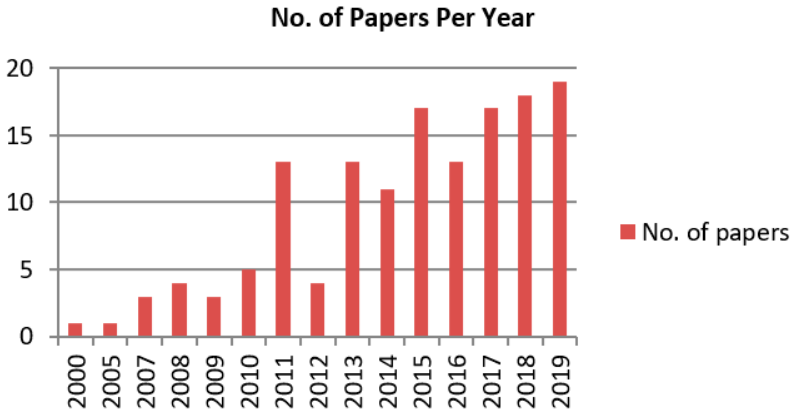

Figure 5. Number of papers per year.

Figure 6 depicts the number of papers connected to relational and NoSQL databases for each digital library. Most relevant articles are in the IEEE and Springer digital libraries. Other possibilities in Figure 6 include articles such as white papers, book chapters, and technical reports from a variety of publishers, including Oracle, MIT, Academic Journal, SciTePress, IJACSA, and IOPScience. Figure 7 depicts the paper category (journal/conference proceeding) of a particular digital library from Figure 6.

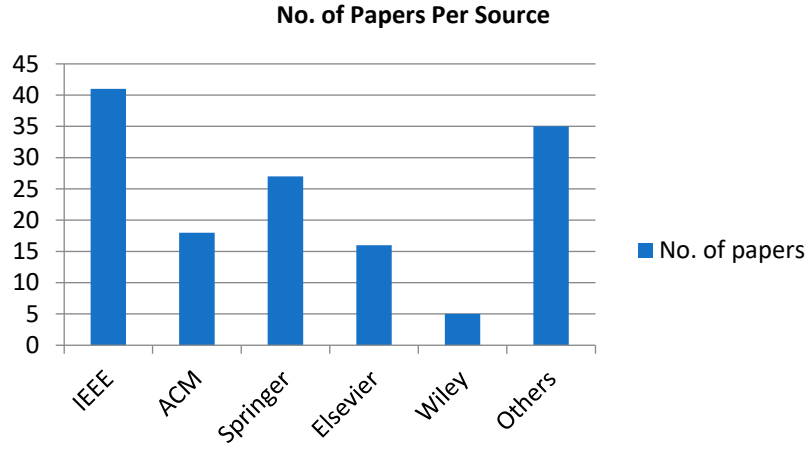

Figure 6. Number of papers per digital library.

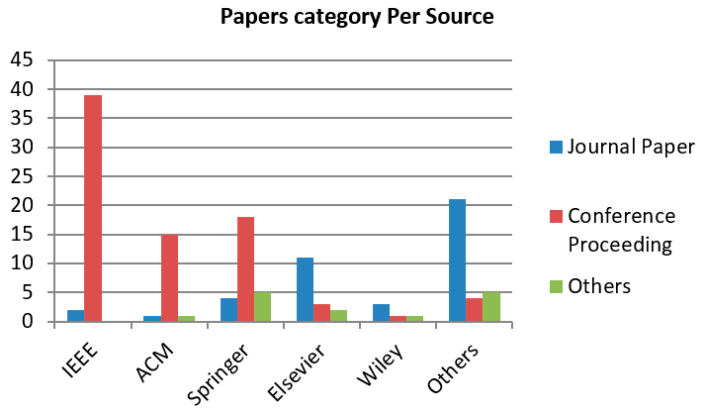

Figure 7. Number of papers category per source.

Many researchers compared the performance and properties of databases, such as MongoDB, MySQL, Cassandra, Couchbase, Oracle, SQL Server, PostgreSQL, Neo4j, and Hbase, throughout the study of the chosen studies. Figure 8 shows the variety of databases used to select the total number of research works.

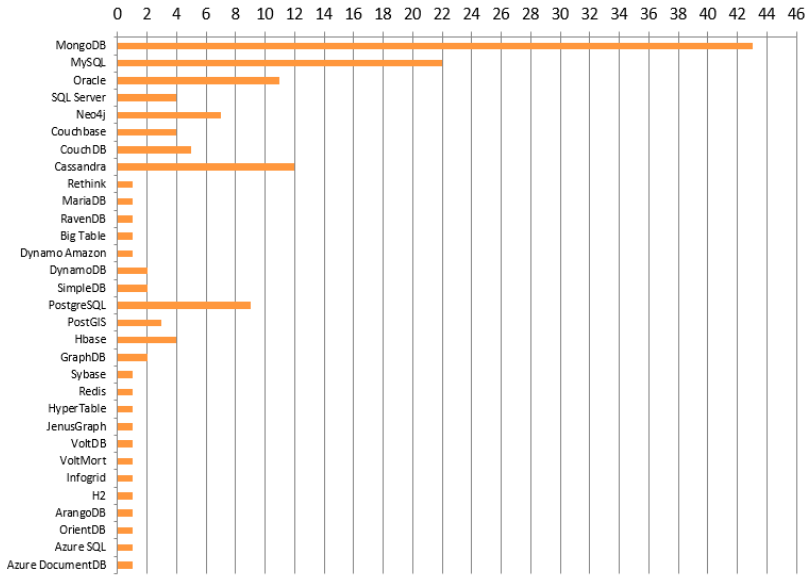

Figure 8. Databases used in selected studies.

As can be seen in Figure 8, most studies comparing and analyzing performance and attributes used one or more of the following databases: MongoDB, MySQL, Oracle, Cassandra, PostgreSQL, Neo4j, SQL server, or CouchDB.

3.1. Empirical Studies Analysis

Empirical research and articles fall into the categories of comparisons, evaluations, experiments, categorizations, dialogues, and surveys. Articles that we believe were relevant to our study topics were chosen from the literature and assessed in light of our hypotheses and selection criteria.

RQ1: Considering big data (structured and unstructured data): What is the need for NoSQL databases? RQ2: Why do NoSQL databases follow the BASE property instead of the SQL database ACID property?

Modeling a database helped us anticipate the types of data that will be stored in it and how they will be stored.

NoSQL, an acronym for "Not only SQL" [22], is an approach to database management that excels at handling massive amounts of unstructured data and big data [23] analytics [14]. All sorts of different query languages can be used with these databases, and they do not adhere to a strict, predetermined schema structure. However, over the past few decades, relational databases have used the industry standard SQL language. Document-oriented databases are a subset of NoSQL databases. Databases that focus on storing and retrieving documents include MongoDB and CouchDB. Databases of this type are utilized for the storage and administration of data that is primarily document-based. Complex data formats, such as JSON, BSON, XML, and PDF are used to store information in document-oriented databases. Both MongoDB and CouchDB are free and open source; however, MongoDB [24] is more suited to a distributed setting and to JSON. Researchers in [25] investigated several different NoSQL database features. One popular database that was built specifically with JSON in mind is MongoDB [22,26], which uses the C++ programming language. MongoDB uses dynamic schema [17] structures [1] instead of predefined, static documents. Data analysis and retrieval are quick and accurate because of improvements in query processing, indexing support, and in-memory aggregation. In addition to offering complete safety, it provides recovery and backup utilities. SQL and NoSQL databases are shown in Figure 9 along with their respective data storage structure.

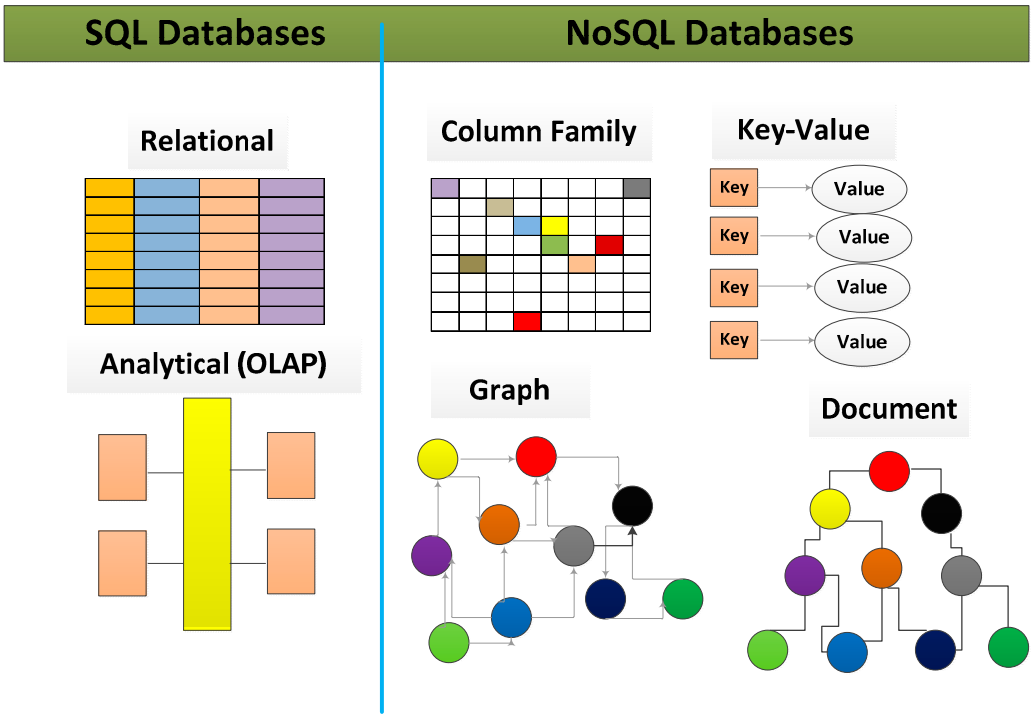

Figure 9. SQL and NoSQL database storage structures.

SQL (Oracle, MySQL, and SQL-Server) and NoSQL (MongoDB, Neo4j) databases have been the subject of numerous studies comparing their structure, design, and performance [24–50]. These studies used the proposed techniques for situations involving SQL and NoSQL databases and analyzed the outcomes. NoSQL databases have different features and applications. Because of their ability to scale horizontally, NoSQL databases cannot guarantee the ACID property. In this regard, the Neo4j graph database [51,52] is an excellent option. Graph databases [53–55] have been shown to effectively organize and store data with complex dependencies. Documents serve as MongoDB's primary focus. MongoDB uses the BSON format, which is an offshoot of JSON.

With huge data, MongoDB performs well and has shown itself repeatedly. Research in [56] compared the NoSQL database MongoDB with the relational database Postgre SQL for social network service and stream sensor data. In the scenarios where the two databases were compared; MongoDB came out on top [56]. Unlike relational database management systems (RDBMS), the NoSQL [57] database is well-suited for use in a cluster setting, where its flexible and powerful architecture can accommodate vast amounts of data in a wide variety of forms. MongoDB and MySQL were compared with the canonical database management system in [35]. MongoDB outperformed MySQL in data retrieval and data insertion. Based on the comparisons, they presented, it was clear that NoSQL MongoDB was the superior option to MySQL when handling large amounts of data. In-depth research of the prerequisites for effectively managing massive amounts of unstructured data was conducted for articles [43,58]. They preferred to use the NoSQL MongoDB database instead of employing MySQL RDBMS. The relational and non-relational database model was examined in detail [44]. Depending on their analysis, the NoSQL MongoDB database outperformed the MySQL database. They suggested that for a small dataset with basic queries, MySQL is efficient, while for large datasets with complicated queries, MongoDB is a more acceptable solution. The authors of [40] compared the performance of Oracle RDBMS with the NoSQL MongoDB database. As a result of their research, they concluded that NoSQL databases are not a viable alternative to SQL databases. They [40] claimed that companies can select a database suitable for their needs by considering the specifics of their company.

As for data scalability, high performance, and data availability, we discovered many publications [59–68] that discuss migrating from a relational database to a NoSQL database, as well as other papers [59,69–78] on the topic. Because of its scalability, integrity, dissemination, security [24,59,69–74], and customizable designs, MongoDB is well-suited for processing massive amounts of data. According to the cited works, switching from a relational database to the NoSQL database MongoDB is the way to go. MongoDB uses the Binary JSON Object Notation (BSON) format to store its information. Because it does not require junctions as do relational databases, MongoDB is able to store and retrieve large numbers of documents in a single collection efficiently. MongoDB is versatile enough to handle and process data in a wide variety of formats, including structured, semi-structured, and unstructured data. Relational databases were created with the express purpose of handling organized data, and as such, they can handle a certain amount of data that is vertical in nature. To be clear, NoSQL databases are not meant to replace relational databases, but their increased performance and utility mean that they have a place alongside relational databases in some contexts. According to the research [60,79], the relational database system does not scale well with massive amounts of data. Automatic mapping from MySQL RDBMS to a MongoDB NoSQL database utilizing metadata saved in MySQL RDBMS was proposed in article [61]. NoSQL databases do not promise to offer the same atomicity, consistency, isolation, and durability that are hallmarks of relational database management systems [80]. The BASE [81] feature is observed by NoSQL databases. NoSQL [60] is an umbrella term for a collection of technologies that share the goals of data availability, efficient data scaling, effective storage management, and improved performance. Research in [25] looks at the motivations for the shift from relational database management systems to nonrelational document-oriented databases. Meanwhile, the ACE (availability, consistency, and efficiency) aspects of the big data system have been explored in both databases [82,83]. Research into the topic of automatic schema transformation from SQL to NoSQL can be found in [84].

Articles [70,85] and [86] compared the efficiency of several NoSQL databases (including Couchbase, MongoDB, Re-thinkDB, and in-memory) when used with real-world data. While planning the trials they've proposed, the authors considered several factors, such as the response time of various databases. Comparative performance analyses between Oracle NoSQL and MongoDB were conducted in the study [71]. Databases were studied from multiple angles, including their storage model, scalability, concurrency, and replication. The authors drew the conclusion that, overall, MongoDB was more popular than Oracle's NoSQL offering. Based on DBEngine.com's rankings, MongoDB is the fifth best NoSQL database, whereas Oracle NoSQL is ranked as number 78. The paper [73] claimed that when compared with MongoDB's MapReduce, Oracle's RDBMS aggregation performed better. On the other hand, regarding the speed with which queries were answered, MongoDB bested Oracle RDBMS. According to [40], Oracle aggregation was superior to MongoDB aggregation for SUM, COUNT, and AVG workloads. The work in [40] used a dataset with approximately 500,000 records, which was insufficient for big data analytics. The process flow diagram of the SQL Select statement in Oracle 11g RDBMS is depicted in Figure 10.

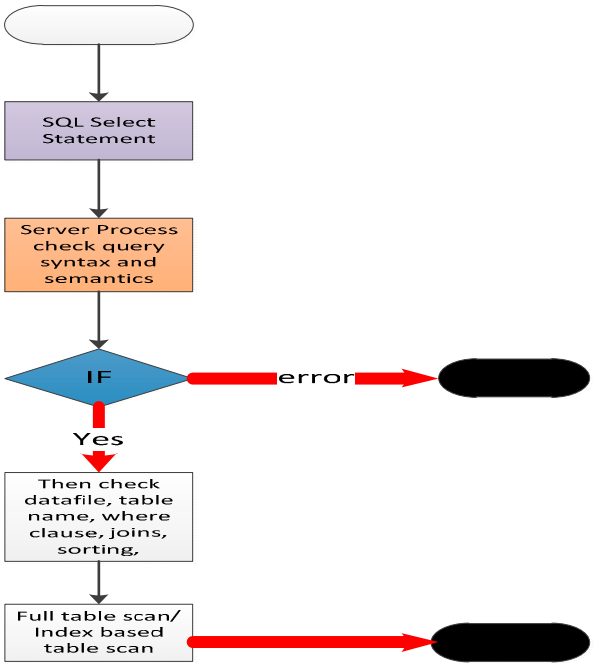

Figure 10. SQL Select statement process flow diagram.

MongoDB data retrieval is simple and easy because of the sub-document structure and does not require checking constraints or any clause, unlike the SQL Select statement of Oracle 11 g RDBMS. The SQL Insert statement process flow is shown in Figure 11.

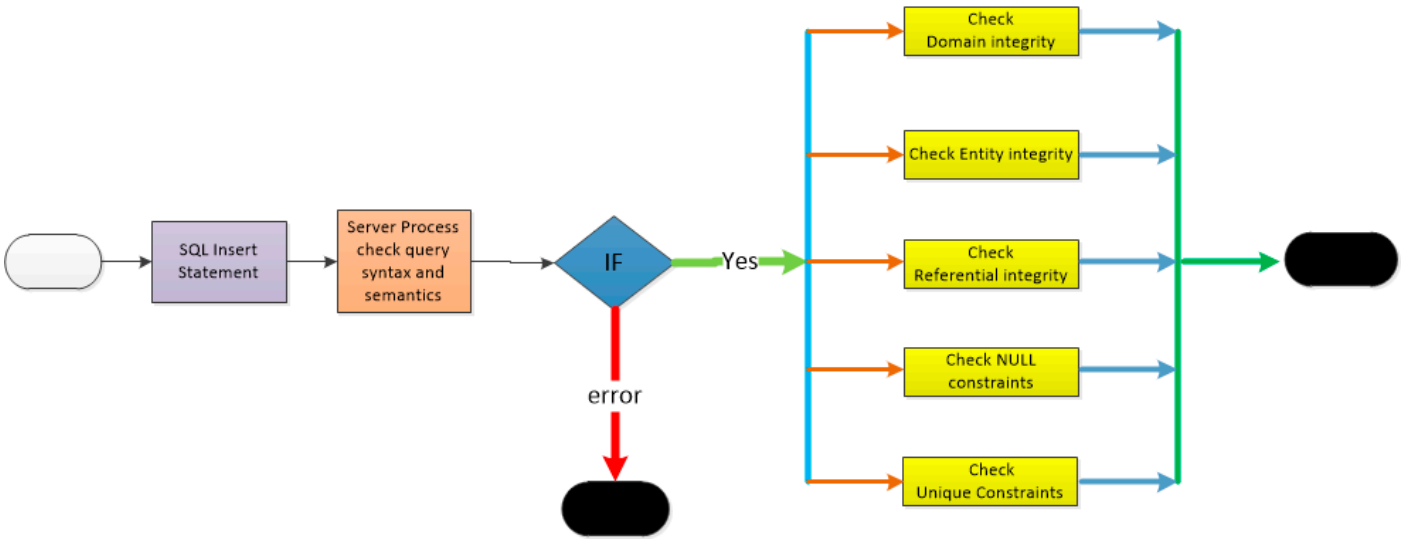

Figure 11. SQL Insert statement process flow diagram.

In comparison with Oracle RDBMS, MongoDB insertion is faster since it does not have to verify the steps shown in Figure 11 of the SQL Insert statement.

In contrast, MapReduce is the programming module [87] that performs well in distributing scenarios and is more ideal for big data [88] analysis compared with basic aggregation [73] in cluster (multiple server) contexts. Before commencing the map stage, MapReduce operations can conduct any arbitrary sorting, and they limit the documents in a single collection that they utilize as their input. The two phases of MapReduce, "Map" and "Reduce", are required for the MapReduce algorithm to work; MongoDB uses the map phase for each document that is fed into MapReduce. A pair of critical values is

returned from the map function. MongoDB employs a process called "reduction" to gather and compress the aggregated data, and then MongoDB stores the results in a collection. For example, in the Chicago crime dataset, the map function calculates all the crimes against each day, and then the reduction function takes the day as the key and extracts the appropriate values (Key: values). The MapReduce–Merge framework was developed by the authors of [89] by including the merge operation into the MapReduce architecture. The performance of MapReduce was increased and was able to calculate relational algebra due to the merge operation and was able to process the data in the cluster. In order to maximize MapReduce productivity and enhance the cluster's query performance, others proposed the MRShare framework [90]. Therefore, Oracle RDBMS aggregation outperformed MongoDB in MapReduce aggregation.

Over the past decade, Oracle RDBMS has been widely adopted by businesses of all sizes. Due to their single-image abstract system architecture, SQL RDBMSs did not fare well with regard to processing unstructured data.

The study [72] investigated several features and classifications of NoSQL databases operating on top of Hadoop [91–93]. The study [72,94] found that NoSQL databases could be broken down into subcategories according to factors such as scalability and data type (connected vs. document). Table 3 compares Oracle RDBMS with MongoDB and lists their primary features.

Table 3. MongoDB and Oracle RDBMS Characteristics.

SQL	MongoDB
Rigid schema	Flexible schema
Table	Collection
Row	Document
Column	Field
Multiple joins are required to obtain the complete detail of a single person, student, or customer, which causes slow data accessibility.	Data accessibility is very fast, as all data are stored in one single document
Vertical scalability	Horizontal scalability
Complex data sharding	Data sharding becomes simpler
Considers data storage efficiency	Considers the speed, performance, developer time
Performs well in aggregation	Performs better in a retrieval
Does not perform read/write very quickly in big data analytics	Performs read/write very quickly because of memory mapping function in big data analytics.
Supports relational algebra [60]	Supports relational algebra [60]

The key strengths of Oracle relational database and MongoDB NoSQL database are described in Table A3 (Appendix A), while Table A2 (Appendix A) compares NoSQL databases (MongoDB, Neo4j) with relational databases (Oracle, MySQL, and SQL-Server).

According to the cited works [95–97], there has been a notable increase in the quantity of geospatial and geolocated data. Many applications were developed to handle and use geospatial data [98–103], and this is true across many different fields (emergency management, archaeology, IoT, and smart cities). In order to process the massive amounts of geographical data effectively [27], a powerful database management system is strictly required.

When handling massive amounts of data, NoSQL databases, rather than SQL databases, are the best option for web applications [57,104–107]. Geospatial and location data were also managed using NoSQL databases [108,109]. It was suggested in multiple reports [110–113] that NoSQL databases were able to effectively process vast amounts of unstructured data. The study [114] reported that researchers first became aware of spatial data in Geographic Information Systems foundational writing. Two problems with effective geospatial query data processing were investigated in the research [115,116]. The first was

that conventional optimization methods were inadequate for solving geographical queries. When taking big analytics into account, it became clear that every method and approach to geographic queries tried thus far was inadequate for the massive amounts of data involved. In contrast to traditional textual data, which contain only a limited set of qualities, geospatial data offer a wide range of additional features. Extensive geospatial data processing, according to several studies [113,117–126], necessitates well-established methods for processing the enormous quantity of geo-spatial data at hand. The article [127] demonstrated that well-known RDBMS systems encountered numerous difficulties when attempting to analyze geographical data. Using a performance comparison analysis, the authors of [108] looked at how NoSQL document database MongoDB stacked up against the relational database management system PostGIS. As their tests showed, MongoDB performed better than PostGIS when handling geospatial data.

The oracle spatial storage data model [128] had two basic components: location and form. Storage models, such as index Engine and Geometry Engine, employed by query analysis, made use of the SDO_GEOMETRY data type. For location services, a geocoder was utilized to translate an address into SDO_GEOMETRY information. Oracle Maps and Map Viewer were used for visualization.

SQL Server 2016 (https://www.microsoft.com/en-us/cloud-platform/sql-server (accessed on 07 January 2020)) and Azure SQL database (https://azure.microsoft.com/en-us/services/sql-Database (accessed on 09 January 2020)) and its cloud version both include various geo-functions for geospatial data analytics, as does SQL Oracle spatial database, while NoSQL databases, e.g., Azure DocumentDB (https://azure.microsoft.com/en-us/services/documentdb (accessed on 10 January 2020)) and MongoDB (https://www.mongodb.com (accessed on 15 January 2020)), enable geographical capabilities. The Database as a Service (DBaaS) paradigm utilized by the Azure SQL database also features the Microsoft SQL server's same functionalities and provides cloud services. PostgreSQL (https://www.postgresql.org (accessed on 17 January 2020)) is a free and open relational database system. Comparatively, PostGIS is an extension of PostgreSQL that serves as a spatial database (http://postgis.net (accessed on 18 January 2020)) for working with geospatial information. The NoSQL database Azure DocumentDB is built by Microsoft and provides the same geographic operations and functionalities as MongoDB. MongoDB supports the GeoJSON standard data format. The authors of [129] undertook a performance analysis of geographical data. According to their research, the Azure DocumentDB was faster than the Azure SQL database but less scalable than the Azure SQL database. Popular SQL and NoSQL databases' primary geospatial properties are outlined in Table 4.

Table 4. SQL and NoSQL databases geospatial characteristics.

Database	Oracle	PostGIS	Azure SQL	MongoDB	DocumentDB
Geometry Objects Supported	Point, LineString, Polygon, MultiPoint, MultiLinePoint, MultiPolygon, GeometryCollection	Point, LineString, Polygon, MultiPoint, MultiLinePoint, MultiPolygon, GeometryCollection	Point, LineString, Polygon, MultiPoint, MultiLinePoint, MultiPolygon, GeometryCollection	Point, LineString, Polygon, MultiPoint, MultiLinePoint, MultiPolygon, GeometryCollection	Point, LineString, Polygon, MultiPoint, MultiLinePoint, MultiPolygon, GeometryCollection
Geometry Functionalities Supported	For geometry instances, Oracle has the support of Open Geospatial Consortium (OGC)	For geometry instances, PostGIS has the support of Open Geospatial Consortium (OGC)	For geometry instances, Azure SQL has the support of Open Geospatial Consortium (OGC)	Inclusion, Intersection, Distance/Proximity	Inclusion, Distance/Proximity
Spatial Indexes Supported	B-Trees, Parallel index builds for spatial R-tree indexes	GiST index, R-Tree index, B-Tree index	B-Trees, 2D plane index	2D index, 2D sphere index	2D plane index, quadtree

GeoServer Compatibility	Yes	Yes	Yes	Yes	Yes
DBaaS	Yes	No	Yes (Cloud Computing Platform)	Yes	Yes
Horizontal Scalability	No	No	No	Yes	Yes

NoSQL MongoDB Data Modeling

Figure 12 depicts the shard nodes, configuration servers, and routing servers (or mongos) that make up the MongoDB architecture, as described in [42,130,131].

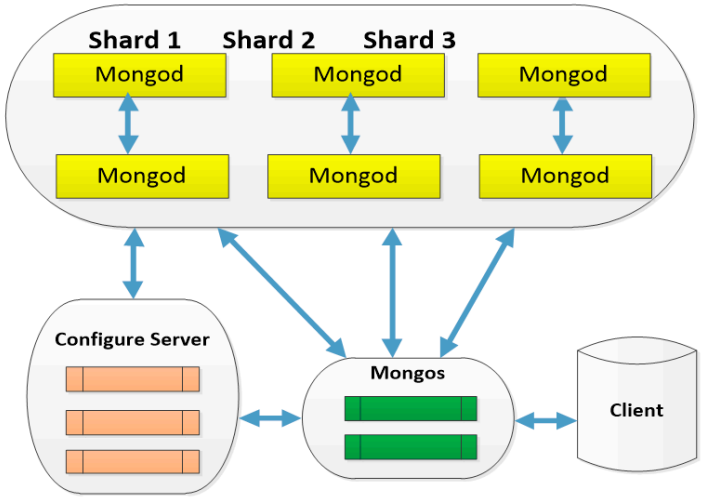

Figure 12. MongoDB Architecture.

The data are stored in a shard, and a MongoDB cluster cannot be constructed without one or more shards. When a node fails, the information for that shard is held by its replicas. Data transactions (read/write) determine which shard is used. One or more servers are used by the replicated node, and the secondary node is modeled like the original server. In case the primary server goes down, one of the backup servers will take over. The server is the hub around which all operations (read/write) revolve. Eventually, the cluster will be in sync with all of the distributed read transactions. A set of configuration servers in the cluster is responsible for storing metadata. Data shards are identified by these servers, which also relay which data chunk belongs to which shard. Customer service requests that either the routing servers or MongoDB carry out the action. Prior to receiving confirmation from the client, MongoDB assigns each user task to the appropriate shard on the basis of the task type, then combines the resulting data. Since Mongos [23] are stateless, they can be used in a distributed setting.

Memory-mapped files are used by MongoDB to make the most of available memory, which, in turn, boosts performance. Indexing in the MongoDB [132] database uses B-trees. In a MongoDB cluster [133], a user can claim ownership of a certain partition collection by using a shard key.

RQ3: Does DBaaS tackle data interoperability and portability efficiently in various NoSQL databases?

In-depth research on the literature of DBaaS architecture was conducted for this work. The data analysis and extraction show that the cloud DBaaS approach developed for relational databases is not optimal for NoSQL databases. Expert time and effort can be reduced, and security can be improved if scientists and researchers can find a unified [134–138] DBaaS [139] solution for both SQL and NoSQL databases. There is also no need to re-engineer programs for use with different CSPs thanks to standard APIs (APIs). Data portability and interoperability among different cloud providers are the primary obstacles to overcome. Interoperability

is defined differently by each of the three paradigms [136]: IaaS, PaaS, and SaaS. Open standards [137] help mitigate the interoperability issue; however, our solution is focused on the IaaS layer specifically. When transferring information among different cloud providers, unified APIs are typically necessary [138]. There is not a single data storage model utilized by all cloud services. When a developer migrates from one CSP to another, the data are transferred according to the high-level architecture depicted in Figure 13.

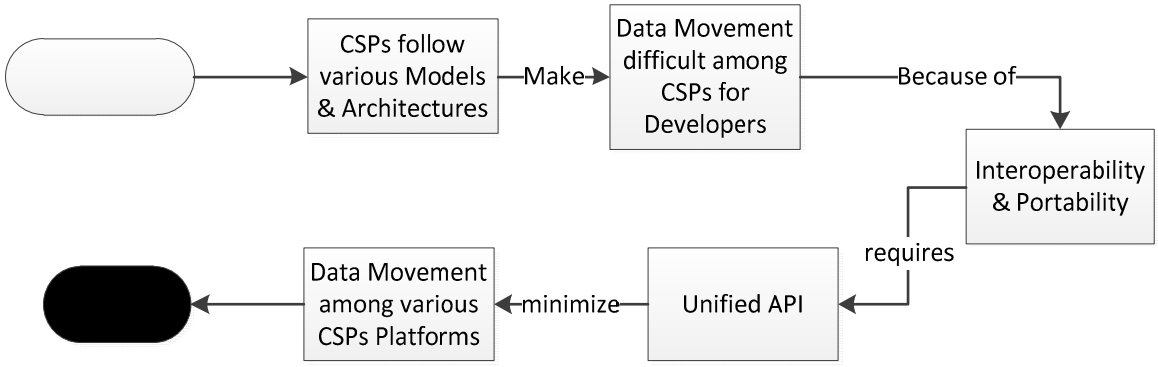

Figure 13. Data Movement inside CSPs.

Databases [140] that are both consistent and connected, as well as management [141] that is both effective and efficient, are key and critical concerns in the modern information technology era. The fundamental features of a database system are the permanent data store's assurance of the data's independence from the underlying physical storage medium, as well as the query processing capabilities enabled by declarative queries (DBS). In the database sector, a wide variety of approaches to the various domains have been observed. Multiple DMSs, including ACID, OOM, XML, and data warehousing, are based on the relational data model. However, the BASE [81] attribute is supported in NoSQL databases, which are primarily intended for processing large amounts of data.

New cloud services [142] and capabilities are being offered by cloud service providers to customers at a low cost and with high efficiency. However, multiple cloud service providers offer the same features using varying implementations and user interfaces [143], which inevitably causes problems with interoperability [144], incompatibility [145], and portability. These are the difficult problems for cloud service providers while embracing and facilitating cloud technology [146]. IaaS, PaaS, and SaaS interoperability are all different terms with different meanings and applications within the realm of cloud services [136]. Cloud users should be moved from one CSP to another for the following reasons [137]: downtime or failure higher rate, contract termination, corporate plan changes, better alternatives with low cost, and legal difficulties. Customers using multiple cloud providers cannot easily port their data among them. The cloud service models strive to control the customer's capabilities since their key architectures lack interoperability. This issue is addressed by resorting to vendor lock-in, which poses a significant security concern for cloud-based models [147]. Data portability [148] will improve as increasingly more cloud service providers (CSPs) adopt the open standard to solve interoperability problems. As a result, it can be challenging for a developer to ensure that data and applications are consistent across different cloud services.

Open Virtualization Format (OVF) (https://www.dmtf.org/standards/ovf accessed on 3 March 2020), Cloud Infrastructure Management Interface (CIMI) (https://www.iso.org/standard/66296.html, accessed on 15 March 2020)Open Cloud Computing Interface (OCCI)(https://www.analyticssteps.com/blogs/what-open-cloud-computing-interface-occi, accessed on 18 March 2020) and Cloud Data Management Interface (CDMI) (https://www.iso.org/standard/60617.html, accessed on 20 March 2020) have all been explored as potential solutions for managing interoperability across CSPs. These standards support only the IaaS layer. Similarly, many initiatives, such as MOSAIC [4,149], MODACLOUDS [5,150], and the Clous4SOA [151], have been established to address the

semantic interoperability challenge at the PaaS layer. Having multiple APIs for each PaaS means that the initiatives cannot solve the interoperability problem on their own.

As expected, the CIMI standard is compatible with the IaaS API and helps reduce the degree of interoperability required by cloud users and their infrastructure service providers. By using OCCI, the interoperability between CSPs can be reduced while still producing adequate results. Because these standards do not incorporate interoperability features into their underlying architectures, they are not widely used by a variety of CSPs. In addition, there is a problem with the availability of standardized interfaces and APIs.

Design patterns, cloud middle infrastructures, the Service Delivery Cloud Platform (SDCP), and migration tools have all been developed by other academics [137,152,153] to facilitate the movement of data among different clouds. Despite saving users time, these methods do not solve the portability problem among different CSPs that arises from having to learn and implement several APIs. Amazon Web Service (AWS), Microsoft Azure, and Google App Engine (GAE) are examples of cloud service providers that help consumers build and launch cloud-based apps. In addition, they provide the Database as a Service (DBaaS) cloud platform to support the database developers. Data porting in DBaaS, which can occur between SQL [154] and NoSQL [155] databases as well as inside each type of NoSQL database, can potentially be problematic due to the interoperability issue. Many types of NoSQL databases adhere to incompatible storage formats and data paradigms. There is a need for standardized APIs that can regulate data movement among various cloud storage platforms.

To host their data and applications, software developers can take advantage of DBaaS [139], a highly scalable and available backend cloud service. As a result, DBaaS is the most alluring option for cloud customers right now [154]. Database as a Service (DBaaS) [156] is a cloud service offered by cloud providers (CSPs) that facilitates the transition from traditional database architecture to cloud database architecture. SaaS facilitates remote access to computer programs and their features and functionality. PNUTS, HBase, SimpleDB, and Google BigTable are some other cloud-based database services. Potentially, the DBaaS framework can do a good job of handling traditional databases. However, DBaaS performance declines in areas such as data consistency, confidentiality, integrity, availability, and lack of security due to the different approaches followed by the many databases. If data privacy and security were guaranteed, the outsourcing model known as DBaaS may be a financial success. An expert virtualization consultant for private table database clouds was proposed in the research [157]. Cloud computing allows for the integration of numerous applications into the framework of the service itself. The scalability and adaptability of cloud computing comes at a lower price than ever before.

To provide more data interoperability, portability, and security during data porting, the paper [134,135] investigated two unified API frameworks, CDPort and Se-cloudDB, for SQL and NoSQL databases. Before handing it over to a third party, the API that they offered ensured the privacy of users' critical information (CSP). As part of their planned MCTool, requests are transformed into their corresponding models and then communicated to the appropriate database, considering the models that it can handle. Metadata encryption/decryption keys ensure that only authorized users and the DBA have access to and can make changes to the data stored in the various clouds. Encryption and decryption are supported by their proposed framework.

4. Discussion and Classification

The SQL vs. NoSQL debate is not about relational versus non-relational databases, despite the names. Both databases' transaction models are analyzed and contrasted. When an application is executed, the database performs a series of operations known as "transactions". All transactions in a SQL database depend on ACID properties. In this case, ACID refers to the principles of atomism, consistency, isolation, and durability. However, NoSQL database designers concluded that the ACID property was a useless roadblock to securing huge data. Consequently, in the early 2000s, Professor Eric Brewer proposed a new theory. The CAP principles are consistency, availability, and partition tolerance. The

theorem states that all these qualities can be obtained concurrently by designers working in a distributed environment. As a trade-off for guaranteed consistency and availability, developers can implement a partition-tolerant database using the CA model. If the developer of a database places more value on availability and partition tolerance than consistency, then the database cannot be said to be AP-based. To achieve consistency and partition tolerance without compromising availability, a CP-based database is deployed. Table 5 describes the main features supported by both flavors of databases.

Table 5. DBMS's Features.

DBMSs	Features								
	Data	**Schema**	**Scalability**	**Compliance**	**Architecture**	**Consistency**	**Query Language**	**Performance**	**Best Suited For**
RDBMS	S	Fixed	Vertical	ACID	Centralized	Strict	SQL	Slow	BFT
NoSQL	SUSm	Dynamic	Horizontal	BASE	Distributed	Eventual	OO API, SQL Like	Fast	LSWA, SD

S: Structured; SUSm: Structured, Unstructured, and Semi-Structured; LSWA: Large-Scale Warehouse Applications, Sensor Data; BFT: Banking, Financial Transaction.

Standardization on how to classify DBMSs has led to several sub-categories. The foundation of any database management system is the data model upon which it is built. Many contemporary commercial DBMSs rely heavily on the relational data model. The object data model has seen little adoption, even though it has seen use in a small number of commercial systems. Many legacy applications continue to run on database systems based on hierarchical and network data models. We can classify the NoSQL business model as one of the following, as defined in Table 6.

Table 6. Database Classification.

Challenges with reliability and availability (CA): This database design places a priority on data consistency and accessibility using a replication approach. Parts of the database do not care about partition tolerance. If the nodes are partitioned, the data will go out of sync. Vertica, Greenplum, and relational database management systems are examples of this type of database.
There are problems with consistency and partition tolerance (CP) that must be fixed. The primary objective of such a database management system is to ensure the integrity of the data it stores. However, high availability is no longer supported. The data are stored in the various nodes, and when a node crashes, it causes the data that ensures consistency between them to become unavailable. It maintains partition tolerance by blocking data resynchronization. Hypertable, BigTable, and HBase are only a few examples of the database systems that are CP-aware.
In this type of database, providing data availability and partition tolerance (AP) is a top priority. If there is a communication breakdown among nodes, it does not affect the status of any individual node. After a partition is resolved, data are resynchronized but consistency is not ensured. These principles are followed by databases such as Riak, CouchDB, and KAI.
When one element of a database goes down but the others keep running, that section of the database is said to be "basically available." In the event of a node failure, the operation will continue by replicating the data among the remaining nodes.
In a soft state, data are subject to change over time depending on factors such as the level of participation of the user. The usefulness of such information may also deteriorate after a certain period has passed. Therefore, it is necessary to either update or obtain the data for the information to be useful in the system.
According to eventual consistency, after every data modification, the data do not instantly become consistent throughout the entire system, but they will become consistent eventually. The data, it is said, will continue to be accurate into the foreseeable future.

NoSQL database systems have been given a variety of indications, which can be attributed to the fact that the implementation structure does not include conventional SQL. Several distinct types of NoSQL databases each make use of a unique set of querying procedures. Software developers are often responsible for appropriately designing query accomplishment rather than depending on a declarative query language that combines questions and delivers fast query execution plans. This is because a declarative query language can combine questions. Users of the system are responsible for performing extra tasks, such as validating and interpreting the information that has been gathered. In addition, the responsibility for ensuring data consistency, data replication, and availability during concurrent changes in shared and replicated databases is shifting more and more to the hands of developers. The combination of massive data systems with NoSQL databases can have several different effects on software designs and architectures.

Brewer formulated a set of requirements for distributed databases by utilizing his CAP method [7]. These requirements serve as a baseline. In the case where there is a network partition (P: in the cluster, when the information is lost randomly between the nodes), consistency (C: present the identical data to all users) must take precedence over availability (available to users) (A: the acknowledgement must be received by every client as either success or failure). If there is not a partition (P), a framework will favor latency (L) over consistency (C), as shown by Abadi's PARCELS; nevertheless, when there is a partition (P), the framework will emphasize availability (A) over consistency (C).

Today, it is not enough to rely solely on structured data, as the unstructured nature of astronomical data necessitates fast data analytics and information extraction methods. SQL database capabilities are limited in their ability to effectively process various forms of large data. The capabilities of NoSQL databases allow for the efficient processing of massive data sets. NoSQL databases excel in the areas of storage capacity expansion, schema flexibility, scalability, and real-time access. The BASE attribute is adhered to by NoSQL databases. Instead of prioritizing data consistency and security, NoSQL databases place an emphasis on improving data read/write efficiency. The application is responsible for ensuring data consistency. For this reason, it is an excellent option for handling large datasets. In addition, NoSQL databases do not apply restrictions at the data, column, or table levels as used in structured database.

This research analyzed approximately 140 previous studies that compared the usefulness, productivity, and dependability of SQL databases with NoSQL databases. This research considered vast amounts of data. According to our investigations, NoSQL databases offer a greater capacity for expansion [23,37,68,158,159]. SQL databases are better suited for transactional systems and use more resources to ensure data integrity and consistency, whereas NoSQL databases are better equipped to handle huge and diverse datasets and use fewer resources to ensure data integrity and consistency. On the other hand, NoSQL databases [24,95] are distinct in the sense that they place a larger priority on the accessibility of data. The results of our testing indicate that relational databases cannot be effectively replaced by NoSQL databases. Because both databases have their advantages and disadvantages, the database that an organization chooses to use will depend on the requirements that are unique to that business. To give just one illustration, NoSQL databases, which make it possible to use the MapReduce programming module, are better suited to parallel computing [124,160] when they are implemented within a cluster environment.

NoSQL databases are constructed on a schema that is more fluid and dynamic, in contrast to relational databases, which are strongly dependent on a preset data structure that is referred to as a "schema" (tabular form) [24,62,68,81,85,158,161]. For example, to keep track of student data, one should use the StdRegNo, StdName, and StdAddress fields. When working with relational databases, the first thing that needs to occur is the construction of a schema that satisfies all the necessary domain and integrity requirements. After that, one will be able to save the relevant student data and comply with the necessary restrictions. When considering extending the scope of the current database by

incorporating two new columns, one should give this some consideration. Those who are going to be responsible for making modifications to the existing schema will also be responsible for migrating data from the previous schema to the new one. When dealing with large data sets, the procedure might become impractically slow and drawn out. The fact that NoSQL databases do not automatically update themselves whenever new changes are made to the schema presents the most significant challenge for agile software development. [25]. When working with NoSQL databases, it is not necessary to have a predetermined schema to make any kind of change. Table A4 of Appendix A contains a listing of the several databases that were accessed during the research that was selected.

Data normalization to control outliers, relational schema (attributes), domain constraints, check constraints, unique constraints, and Not NULL constraints, among other things, all aid safe data integration in relational databases [24,26,37] as opposed to NoSQL databases. [24,26,26,37] Relational databases provide well-established security and authentication systems for users. SQL databases are superior to NoSQL databases in terms of both performance and endurance. SQL is the standard interface for all relational databases; however, NoSQL databases do not use SQL. Instead, they use a different interface.

SQL and NoSQL databases follow a wide variety of models [60,68,126]. In addition, there is a difference in the underlying data model used by each NoSQL database [72,76] type. Given the prevalence of multiple CSPs, it is difficult to accomplish data portability [134,138]. The rapid expansion of commercial databases presents another difficulty for cloud storage. Instead, AWS DBaaS [1] offers the storage expansion options for Azure cloud databases are somewhat restricted. Product categories, architectures, database types, and languages are outlined in Table 7.

Table 7. Database product category, architecture type, and languages.

Database Name	Database Category	Database Architecture	Database Type	Written in
MongoDB	NoSQL-Document Store	Distributed Multi-Model	Open Source	C++, Go, JavaScript, Python
MySQL	SQL		Open Source	C, C++
Oracle	SQL		Not	Assembly language, C, C++
SQL Server	SQL		Not	C, C++
Neo4j	NoSQL-Graph Family		Open Source	Java
Couchbase	NoSQL-Document Store	Distributed Multi-Model	Open Source	C++, Erlang, C, Go
CouchDB	NoSQL-Document Store	Distributed Multi-Model	Open Source	Erlang, JavaScript, C, C++
Cassandra	NoSQL-Column Based	Distributed Multi-Model	Open Source	Java
Rethink	NoSQL-Document Store	Distributed Multi-Model	Open Source	C++, Python, Java, JavaScript, Bash
MariaDB	SQL		Open Source	C, C++, Perl, Bash
RavenDB	NoSQL-Document Store		Open Source	C#
BigTable	NoSQL-Column Based		Not	C++ (core), Java, Python, Go, Ruby
DynamoDB	NoSQL-Key/Value Store		Not	Java
SimpleDB	NoSQL-Key/Value Store	Distributed Database	Not	Erlang
PostgreSQL	SQL		Open Source	C
PostGIS	SQL		Open Source	C

Hbase	NoSQL-Column Based	Distributed Multi-Model	Open Source	Java
GraphDB	NoSQL-Graph Family	Distributed Database		
Sybase	SQL		Not	SQL
Redis	NoSQL- Key/Value Store		Open Source	ANSI C
HyperTable	NoSQL-Column Based		Open Source	C++
JenusGraph	NoSQL-Graph Family		Open Source	Java
VoltDB	in-memory DBMS		Open Source	Java, C++
VoldeMort	NoSQL-Key/Value Store	Distributed datastore	Open Source	Java
Infogrid	NoSQL-Graph Family		Open Source	Java
H2	SQL		Open Source	Java
ArangoDB	NoSQLDB		Open Source	C++, JavaScript
OrientDB	NoSQLDB		Open Source	Java
Azure SQL	SQL		Open Source	C, C++
Azure DocumentDB	NoSQL-Document Store		Open Source	SQL (Core) API

4.1. Research Gap

The implementation of complicated distribution systems is required to achieve significant levels of both availability and scalability. In addition, sharding and partitioning take place on the foundational layers of the software architecture, which are referred to as the application tier, the caching tier, and the back-end storage tier, respectively. To achieve great scalability, however, can be challenging because of the atomic abstraction image that is presented by a typical framework that uses SQL as the gold-standard non-procedural language. In addition, the software needs to be intelligent to tackle issues that arise with data replicas and inconsistencies that are brought about by collisions between changes to replicas that are happening at the same time. When regard to quality criteria, such as scalability, consistency, performance, and durability, each variety of NoSQL database has its own unique set of drawbacks and limitations. In addition, the architect is fundamentally required to conduct research on the characteristics of the nominee database to fulfill the requirements imposed by the vendor while choosing the appropriate database. The gaps of each flavor of NoSQL databases are described in Table 8.

Table 8. Gaps among NoSQL databases.

If your applications simply need to store and retrieve data items that are transparent to the DBMS and can be identified by a key, a key-value store may be the best option for you. In contrast, the software crashes if it tries to perform a database query based on a value for an attribute other than the key. Furthermore, it is impossible to update or obtain just one field from a document's key-value store.
When applications require granular control over which records to obtain, which fields within a record to change, and which fields to retrieve based on criteria other than the primary key, document databases are an excellent option. Document data stores provide more query flexibility than key-value stores.
When applications need to store data with hundreds or thousands of fields but need to access only a subset of these fields in most queries, column-family data stores are an efficient solution. Such data repositories are well suited for massive data sets.
Graph databases are ideally suited for use cases that include storing and analyz-ing data on entities with complex relationships to one another. In a graph data-base, the importance of entities and their connections is treated equally.

The term "big data" [88,158,162] is frequently used to refer to data obtained from large-scale sensor networks or user-generated information from social networks [88,158,162]. However, because many people do not have the necessary skills, they do not know how to handle data to accomplish a desired outcome for their businesses. This is because a significant number of the technologies, techniques, and disciplines concerning big data are relatively recent developments. Relational data stores, othe other hand, are unable to process such a large data collection, which necessitates the creation of new storage system capabilities. Data stored in a NoSQL database do not require a centralized schema. Because of their greater scalability, data availability, fault tolerance, and the rapid processing of massive amounts of unstructured data, NoSQL databases are suitable for handling big data. There are a lot of questions that have not been solved yet because of the differences between relational databases and NoSQL databases.

The paper concludes that NoSQL is not a suitable alternative to relational databases. In addition to this, NoSQL is an excellent option for heterogeneous big data. Each of these fields has numerous unfilled research voids. The simplicity, scalability, and performance of NoSQL databases' designs are areas with significant room for study. Different from the relational database's strict schema data structure, the NoSQL databases use a flat-file or key/value data model (tabular form). NoSQL databases, due to their horizontal scalability, are well-suited to deal with the vast quantities and varieties of data currently in use. However, this is not the case for SQL databases, which can scale vertically. When comparing NoSQL with relational databases, scalability and performance are the two most important factors. There is also the unanswered research question of how to integrate the features [49] of relational databases with those of nonrelational databases. For NoSQL databases, there is also a lack of study into the problem of simple schema design [163–165].

The development of flexible data migration frameworks [50,54,62,65,76] from a relational database to a NoSQL data storage is another area of research that has received increased interest in recent years. There is a lot at stake regarding the success of the SQL-to-NoSQL data transfer. It is crucial since any company needs to extract useful insights from its own data, such as system resource use and employee performance reviews.

Additionally, NoSQL is the superior option to relational databases on the cloud because of the cloud platform's distributed nature. In order to reduce cloud consumer effort during data movement across many cloud platforms and to exert control over data interoperability and portability concerns, the state of the art requires unified API [134,135,138] frameworks.

4.2. Prediction and Occurrences of DBMSs against a Particular DBMS

Our data set-up is based on Table A4. First, we created a pair of (x, y) data. We then used a combination formula to create a pair of (x, y) for a line with more than two database names, thus constructing 301 pairs. We used a label encoder to encode x and y to numbers from string for pre-processing and used Gaussian Naïve Bayes to train our encoded data. Once it was prepared, we obtained a unique database name from x. We encoded those unique names and then tested them on individual encoded data. Then, we obtained the probability of all classes corresponding to individual encoded data and represented the result as a unique x encoded corresponding to the rest of the different database probabilities. Thus, the table size is n by n, where n is a unique database.

Analysis of the collected data: On the collected data, we trained the Gaussian Naïve Bayes model. Nevertheless, the main issue is that it predicted the MongoDB for all database names except itself. MongoDB in label set (y) are many as compared to the others. Pie Chart 1 and Table A6 of Appendix A show that the MongoDB percentage contains more than 22% of all data. Thus, the model is data-biased. To cater to this issue of data imbalance, we generated data.

Analysis of the collected and generated data: On the basis of the analysis presented in Figure A1, we concluded that the databases "MongoDB, Cassandra, MySQL, and HBase" significantly contributed to a collected dataset. Thus, we adopted the strategy of

ignoring these database names when creating a combination of other database names. The percentage of these database names can then be decreased, and another database name can be increased, thus causing the dataset to be balanced. As we can see from Figure A2 and Table A7 of Appendix A, the dataset seems balanced, and the results differ from what we obtained from the collected dataset.

5. Conclusions

The findings indicate that there is no need to transition from relational databases to NoSQL databases. As there are advantages and disadvantages to both types of databases, a company can choose the DBMS that best meets its requirements. An organization can utilize an SQL database if it places a priority on data standardization and consistency. NoSQL should be utilized when a business has a large quantity of unstructured data and data availability is a high requirement. A relational database may be preferred over a NoSQL database for the aggregation of small datasets, and vice versa for big data analytics. MapReduce is most suited for use in clusters due to its distributed nature. Although MapReduce is slow during the process of aggregation, it is more effective in parallel computing and was developed to handle massive amounts of unstructured data. NoSQL databases, due to their dispersed and highly scalable nature, are well-suited for applications that generate voluminous data with diverse features. When geospatial data are involved, the scalability of relational databases is superior to that of NoSQL databases. NoSQL databases have a faster data response time than relational databases, particularly when handling huge amounts of geospatial data.

NoSQL data stores offer an alternative to conventional RDBMS, but it is probable that organizations may not immediately decide which one to use. The optimal strategy for choosing the proper NoSQL database is to determine what the application requires that a relational database management system cannot deliver. If a Relational Database Management System (RDBMS) can effectively manage the data, a NoSQL storage system may not be required.

Moreover, it is generally agreed that the NoSQL database is a newer development in the database space. However, they are being developed using generally recognized theories. Despite the benefits, NoSQL-based systems are not without their flaws, such as lack of widely accepted standards or a well-known query language for use with NoSQL databases. Each database has its own characteristics and ways of working. In contrast, these data sets are new and dynamic. There is no assurance that all data will be written to the data store correctly because NoSQL databases do not offer strict ACID characteristics.

Furthermore, NoSQL databases make rapid development simple because of their dynamic/flexible schema. Different from the rigid structure of relational databases, the models and architectures of NoSQL databases are more adaptable.

Changing from one storage model to another is challenging for developers because of the different models used by NoSQL databases. Different CSPs use incompatible protocols and interfaces, making them incompatible with one another. Because each CSP has developed their own APIs for their specific services, it becomes difficult for data to be shared between them. To effectively manage data portability and interoperability, NoSQL databases require a single, standardized cloud solution.

Future directions for work regarding structured data include adopting a denormalized strategy for SQL RDBMS and comparing the outcomes of inserting, updating, and retrieving data in MongoDB with those in another system. Instead, we can compare the performance of MongoDB and SQL RDBMS in a specific scenario by keeping an eye on the normalized approach taken by the former and then evaluating the outcomes of data insertion, update, and retrieval. The parallel geospatial approaches used by NoSQL databases demand more attention if they are to serve large numbers of users efficiently. Computer vision [166], object detection [167], signal classification [168], and various other deep learning applications [169] are just some of the deep learning-based applications that can benefit from the expansion of big data [170] methodologies.

Author Contributions: Conceptualization, W.K. and C.Z.; methodology, W.K. and C.Z.; software, W.K., C.Z. and T.K.; validation, W.K., Z.C, and T.K.; formal analysis, W.K. and C.Z.; investigation, W.K., C.Z. and A.M.R.; resources, W.K., C.Z. and K.R.; data curation, W.K., C.Z., T.K., K.R. and A.M.R.; writing—original draft preparation, W.K.; writing—review and editing, W.K., T.K., K.R. and A.M.R.; visualization, W.K. and T.K.; supervision, C.Z. and B.L.; project administration, C.Z. and B.L.; funding acquisition, C.Z., B.L. and A.M.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the National High Technology Research and Development Program of China (863 programs) under Grant 2014AA012204, the NSFC under Grant 61671018, and the Chinese Government Scholarship (CSC) for International Scholars.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: The data associated with this study can be found at: https://www.kaggle.com/datasets/wisalkhan5/db-all-combinations; https://www.kaggle.com/datasets/wisalkhan5/db-data (accessed on 23 April 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

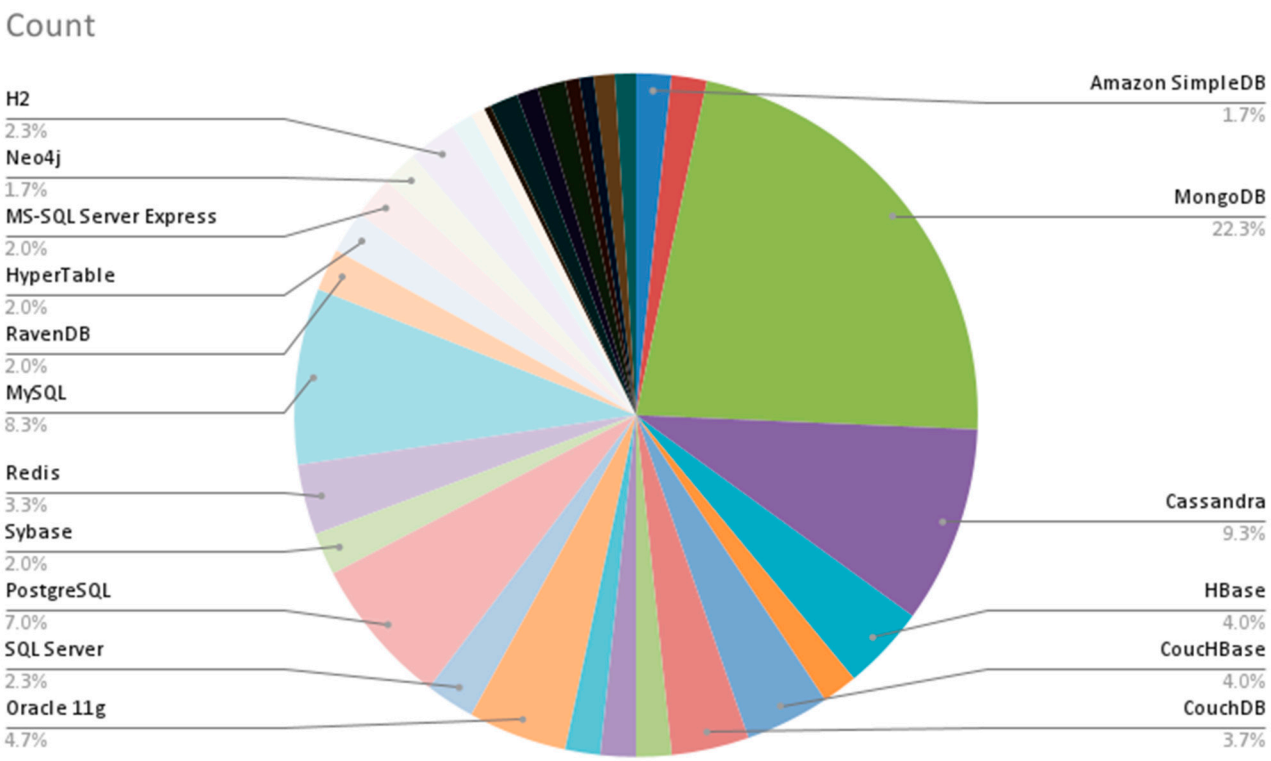

Figure A1. Pie Chart 1: Collected Data.

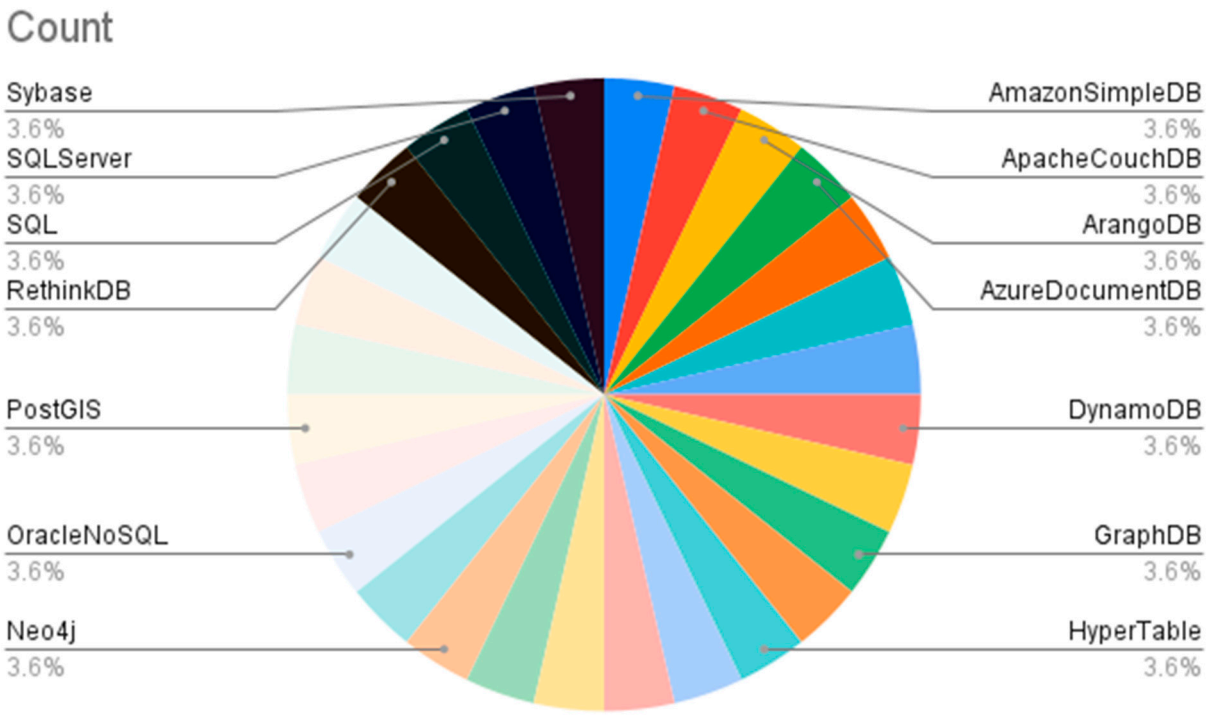

Figure A2. Pie Chart 2: Updated Collected Data.

Table A1. Selected Primary studies.

Title	Publication Year	Journal/Conference Name	Category
Optimization of linear recursive queries in SQ	2009	IEEE Transaction on Knowledge and Data Engineering	Journal
Building scalable databases: Denormalization, the NoSQL movement and Digg	2009	NA	Other
NoSQL–NOT ONLY SQL	2013	International Journal of Enterprise Computing and Business Systems	Journal
Towards robust distributed systems	2000	ACM- Principal on Distributed Computing	Conference
A study on data storage security issues in cloud computing	2016	2nd International Conference on Intelligent Computing, Communication & Convergence	Conference
CloudDBGuard: A framework for encrypted data storage in NoSQL wide column stores	2019	Data & Knowledge Engineering	Journal
Survey on NoSQL database	2011	6th international conference on pervasive computing and applications	Conference
RDBMS to NoSQL: reviewing some next-generation non-relational database's	2011	INTERNATIONAL JOURNAL OF ADVANCED ENGINEERING SCIENCES AND TECHNOLOGIES	Journal
SQL databases v. NoSQL databases	2010	Communication of the ACM	Journal
The transition from rdbms to nosql. a comparative analysis of three popular	2014	Database Systems Journal	Journal

non-relational solutions: Cassandra, mongodb and couchbase			
The battle between NoSQL Databases and RDBMS	2019	NA	Other
Ten years of critical review on database forensics research	2019	Digital Investigation	Journal
Big data processing tools: an experimental performance evaluation	2019	Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery	Journal
MongoDB NoSQL injection analysis and detection	2016	3rd International Conference on Cyber Security and Cloud Computing (CSCloud)	Conference
A comparative study: MongoDB vs. MySQL	2015	13th International Conference on Engineering of Modern Electric Systems (EMES)	Conference
Comparing nosql mongodb to an sql db	2013	In Proceedings of the 51st ACM Southeast Conference	Conference
Using MongoDB to implement textbook management system instead of MySQL	2011	IEEE 3rd International Conference on Communication Software and Networks	Conference
MongoDB vs Oracle--database comparison	2012	third international conference on emerging intelligent data and web technologies	Conference
A performance comparison of SQL and NoSQL databases	2013	IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)	Conference
SQL database with physical database tuning technique and NoSQL graph database comparisons	2019	In Proceedings of 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC	Conference
Predictive Performance Comparison Analysis of Relational & NoSQL Graph Databases	2017	IJACSA	Journal
Comparative study of relational and non-relations database performances using Oracle and MongoDB systems	2014	Journal Impact Factor	Journal
A comparison of a graph database and a relational database: a data provenance perspective	2010	In Proceedings of the 48th annual Southeast regional conference	Conference
SQL support over MongoDB using metadata	2013	International Journal of Scientific and Research Publications	Journal
Comparative analysis of nosql (mongodb) with mysql database	2015	International Journal of Modern Trends in Engineering and Research	Journal
A comprehensive comparison of SQL and MongoDB databases	2015	International Journal of Scientific and Research Publications	Journal
Performance comparison of in-memory and disk-based databases using transaction processing performance council (TPC) benchmarking	2018	Journal of Internet and Information. Systems	Journal
ANALYSIS AND COMPARISON OF DOCUMENT-BASED DATABASES WITH SQL RELATIONAL DATABASES: MONGODB VS MYSQL	2018	Proceedings of the International Conference on Information Technologies	Conference

Performance Analysis of RDBMS and No SQL Databases: PostgreSQL, MongoDB and Neo4	2018	3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)	Conference
Comparison of query performance in relational a non-relation databases	2019	13th International Scientific Conference on Sustainable, Modern and Safe Transport (TRANSCOM)	Conference
Closing the functional and performance gap between SQL and NoSQL	2016	In Proceedings of the 2016 International Conference on Management of Data	Conference
Migration from rdbms to column-oriented nosql: Lessons learned and open problems	2018	In Proceedings of the 7th International Conference on Emerging Databases	Conference
A performance evaluation of open source graph databases	2014	In Proceedings of the first workshop on Parallel programming for analytics applications	Conference
Labeled Property Graphs: SQL or NoSQL?	2019	Ivannikov Memorial Workshop (IVMEM)	Conference
Graph Schema Storage in SQL Object-Relational Database and NoSQL Document-Oriented Database: A Comparative Study,	2019	in International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning	Conference
Graph-Based Denormalization for Migrating Big Data from SQL Database to NoSQL Database	2019	In Intelligent Communication Technologies and Virtual Mobile Networks	Conference
The use of a graph ‐ based system to improve bibliographic information retrieval: System design, implementation, and evaluation	2017	Journal of the Association for Information Science and Technology	Journal
A study on data input and output performance comparison of MongoDB and PostgreSQL in the big data environment	2015	In 2015 8th International Conference on Database Theory and Application (DTA)	Conference
Comparison of SQL, NoSQL and NewSQL databases for internet of things	2016	IEEE Bombay Section Symposium (IBSS)	Conference
Data Migration from Relational Database to MongoDB	2019	Global Journal of Computer Science and Technology	Journal
Modeling MongoDB with relational model	2013	In 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies	Conference
Automatic mapping of MySQL databases to NoSQL MongoDB	2016	in 2016 Federated Conference on Computer Science and Information Systems (FedCSIS)	Conference
Migrating from SQL to NOSQL Database: Practices and Analysis	2018	in 2018 International Conference on Innovations in Information Technology (IIT)	Conference
Data adapter for querying and transformation between SQL and NoSQL database	2016	Future Generation Computer Systems.	Journal
Migration of healthcare relational database to NoSQL cloud database for healthcare analytics and management	2019	Healthcare Data Analytics and Management,	Other
A framework for migrating relational datasets to NoSQL	2015	International Conference On Computational Science	Conference

Transformation of SQL system to NoSQL system and performing data analytics using SVM	2017	In 2017 International Conference on Trends in Electronics and Informatics (ICEI)	Conference
Correlation Aware Technique for SQL to NoSQL Transformation	2014	7th International Conference on Ubi-Media Computing and Workshops	Conference
SQL to NoSQL transformation system using data adapter and analytics	2017	IEEE International Conference on Technological Innovations in Communication, Control and Automation (TICCA)	Conference
Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB	2014	International Conference on Computational Science and Computational Intelligence	Conference
NoSQL real-time database performance comparison	2018	International Journal of Parallel, Emergent and Distributed Systems	Journal
MongoDB and Oracle NoSQL: A technical critique for design decisions	2016	International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS)	Conference
Nosql database: New era of databases for big data analytics-classification, characteristics and comparison	2013	ARVIX	NA
Query Response Time Comparison NOSQLDB MONGODB with SQLDB Oracle	2015	Jurnal Ilmiah Teknologi Informasi	Journal
Relative scalability of NoSQL databases for genotype data manipulation.	2018	Revista de Informática Teórica e Aplicada - RITA	Journal
Scalable SQL and NoSQL data stores	2011	ACM Sigmod Record	Other
SQL-to-NoSQL schema denormalization and migration: a study on content management systems	2015	IEEE International Conference on Systems, Man, and Cybernetics	Conference
SQL & NoSQL Databases	2019	Other	Other
Integration of Relational and NoSQL Databases	2018	In Asian Conference on Intelligent Information and Database Systems.	Other
Literature Review on Database Design Testing Techniques	2019	Advances in Intelligent Systems and Computing	Other
Database engines: Evolution of greenness	2018	Journal of Software: Evolution and Process	Conference
BASE analysis of NoSQL database	2015	Future Generation Computer Systems	Journal
Evaluation of ACE properties of traditional SQL and NoSQL big data systems	2019	In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing	Conference
Adaptive trade‐off between consistency and performance in data replication	2017	Software: Practice and Experience	Other
Automatic SQL-to-NoSQL schema transformation over the MySQL and HBase databases	2015	IEEE International Conference on Consumer Electronics-Taiwan	Conference
Analyzing and Comparison of NoSQL DBMS	2018	International Scientific-Practical Conference Problems of Infocommunications	Conference
A performance evaluation of in-memory databases	2017	Journal of King Saud University Computer and Information Science	Journal

MapReduce: simplified data processing on large clusters	2008	Communications of the ACM	Journal
Database technologies in the world of big data	2015	In Proceedings of the 16th International Conference on Computer Systems and Technologies	Conference
Map-reduce-merge: simplified relational data processing on large clusters	2007	In Proceedings of the 2007 ACM SIGMOD international conference on Management of data	Conference
MRShare: sharing across multiple queries in MapReduce	2010	Proceedings of the VLDB Endowment	Conference
A Comparison of NoSQL and SQL Databases over the Hadoop and Spark Cloud Platforms using Machine Learning Algorithms	2018	IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)	Conference
Performance Analysis of Hadoop-Based SQL and NoSQL for Processing Log Data	2015	International Conference on Database Systems for Advanced Applications	Conference
The impact of columnar file formats on SQL - on - hadoop engine performance: A study on ORC and Parquet	2019	Concurrency and Computation: Practice and Experience	Journal
Working with NoSQL Alternatives	2018	In Cloud Data Design, Orchestration, and Management Using Microsoft Azure	Conference
Evaluation of relational and NoSQL approaches for patient cohort identification from heterogeneous data sources	2017	IEEE International Conference on Bioinformatics and Biomedicine (BIBM)	Conference
Comparison of NoSQL Database and Traditional Database-An emphatic analysis	2018	International Journal on Informatics Visualization	Journal
Performance Comparison of Two Database Management Systems MySQL vs MongoDB	2018	Other	Other
The Comparison of Processing Efficiency of Spatial Data for PostGIS and MongoDB Databases	2019	In International Conference: Beyond Databases, Architectures and Structures	Conference
Geospatial big data: challenges and opportunities	2015	Big Data Research	Journal
Considerations on geospatial big data	2016	IOP Conf. Series: Earth and Environmental Science	Conference
Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale	2017	in Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining	Conference
Report on the Seventh International Workshop on Location and the Web (LocWeb 2017)	2017	IEEE International Conference on Big Data	Conference
Internet of things as a methodological concept	2013	Fourth International Conference on Computing for Geospatial Research and Application	Conference
Twitter under crisis: Can we trust what we RT?	2010	in Proceedings of the first workshop on social media analytics,	Conference

Speeding up the clock in remote sensing: identifying the 'black spots' in exposure dynamics by capitalizing on the full spectrum of joint high spatial and temporal resolution	2017	Natural Hazards	Journal
Geospatial Big Data and archaeology: Prospects and problems too great to ignore	2017	Journal of Archaeological Science	Journal
How Poor Is the 'Poor Man's Search Engine'?	2018	in International Conference: Beyond Databases, Architectures and Structures	Conference
Performance aspects of migrating a web application from a relational to a NoSQL database	2015	In International Conference: Beyond Databases, Architectures and Structures	Conference
MySQL and NoSQL database comparison for IoT application	2016	IEEE International Conference on Advances in Computer Applications (ICACA),	Conference
A proposed performance evaluation of NoSQL databases in the field of IoT	2018	8th International Conference on Computer Science and Information Technology (CSIT)	Conference
SQL or NoSQL? Contrasting approaches to the storage, manipulation and analysis of spatio-temporal online social network data	2014	International Conference on Computational Science and Its Applications	Conference
Comparative analysis of relational and non-relational databases in the context of performance in web applications	2017	International Conference: Beyond Databases, Architectures and Structures	Conference
Evaluation of XPath queries over XML documents using SparkSQL framework	2017	International Conference: Beyond Databases, Architectures and Structures, 2017	Conference
The multi-model databases–a review	2017	International Conference: Beyond Databases, Architectures and Structures	Conference
1.06 GIS Databases and NoSQL Databases	2017	Comprehensive Geographic Information Systems	Other
Geographic information systems and science	2005	Other	Other
Computational model for efficient processing of geofield queries	2009	Man-Machine Interactions, Springer	Other
A data model for heterogeneous data integration architecture	2014	International Conference: Beyond Databases, Architectures and Structures	Conference
Efficient storage of big-data for real-time gps applications	2014	Fourth International Conference on Big Data and Cloud Computing	Conference
An attempt to automate the simplification of building objects in multiresolution databases	2015	International Conference: Beyond Databases, Architectures and Structures	Conference
The extended structure of multi-resolution database,	2014	International Conference: Beyond Databases, Architectures and Structures	Conference
GISB: a benchmark for geographic map information extraction	2015	International Conference: Beyond Databases, Architectures and Structures	Conference

The importance of contextual topology in the process of harmonization of the spatial databases on example BDOT500	2016	Baltic Geodetic Congress (BGC Geomatics)	Conference
A Big Data processing strategy for hybrid interpretation of flood embankment multisensor data	2016	Geology, Geophysics and Environment	Journal
Evaluation of relational and NoSQL database architectures to manage genomic annotations	2016	Journal of Biomedical Informatics	Journal
SQL or NoSQL? Which Is the Best Choice for Storing Big Spatio-Temporal Climate Data?	2018	International Conference on Conceptual Modeling	Conference
Mysql spatial and postgis–implementations of spatial data standards	2011	Electronic Journal of Polish Agricultural Universities (EJPAU)	Journal
Pro oracle spatial for oracle database 11	2008	Dreamtech Press	Other
SQL versus NoSQL databases for geospatial applications	2017	IEEE International Conference on Big Data (Big Data)	Conference
Exploring the Design Needs for the New Database Era	2018	Enterprise, Business-Process and Information Systems Modeling	Other
Forensic investigation framework for the document store NoSQL DBMS: MongoDB as a case study	2016	Digital Investigation	Journal
Performance Analysis of Not Only SQL Semi-Stream Join Using MongoDB for Real-Time Data Warehousing	2019	IEEE Acces	Journal
Security issues in nosql databases	2011	IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications	Conference
Cdport: A portability framework for nosql datastores	2015	Arabian Journal for Science and Engineering	Journal
SeCloudDB: A Unified API for Secure SQL and NoSQL Cloud Databases	2019	in Proceedings of the 2019 3rd International Conference on Cloud and Big Data Computing,	Conference
A Survey on Approaches for Interoperability and Portability of Cloud Computing Services	2014	the proceedings of the 4th International Conference on Cloud Computing and Services Science (CLOSER 2014)	Conference
Design patterns to enable data portability between clouds' databases	2012	12th International Conference on Computational Science and Its Applications	Conference
Cdport: A framework of data portability in cloud platforms	2014	Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services	Conference
Internet of things data storage infrastructure in the cloud using NoSQL databases	2017	EEE Latin America Transactions	Journal
Data management in cloud environments: NoSQL and NewSQL data stores	2013	Journal of Cloud Computing: Advances, Systems and Applications	Journal
Cloud computing—The business perspective	2011	Decision Support System	Journal

The cloudmdsql multistore system	2016	Proceedings of the 2016 International Conference on Management of Data	Conference
A semantic interoperability framework for cloud platform as a service	2011	IEEE Third International Conference on Cloud Computing Technology and Science	Conference
Cloud Computing Interoperability Approaches-Possibilities and Challenges	2012	Local Proceedings of the Fifth Balkan Conference in Informatics	Conference
Cloud computing interoperability: the state of play	2011	IEEE Third International Conference on Cloud Computing Technology and Science	Conference
UML model of a standard API for cloud computing application development	2019	9th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)	Conference
Experiences in building a mOSAIC of clouds	2013	Journal of Cloud Computing, Advances, Systems and Applications	Journal
Supporting the development and operation of multi-cloud applications: The modaclouds approach	2013	15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing	Conference
Portability and interoperability between clouds: challenges and case study	2011	European Conference on a Service-Based Internet	Conference
Simplifying MapReduce data processing	2013	International Journal of Computational Science and Engineering	Journal
A common API for delivering services over multi-vendor cloud resources	2013	Journal of Systems and Software	Journal
A survey of large scale data management approaches in cloud environments	2011	IEEE Communications Surveys and Tutorials	Conference
Relational cloud: A database-as-a-service for the cloud	2011	MIT	Journal
Database as a service (DBaaS)	2010	IEEE 26th International Conference on Data Engineering (ICDE 2010)	Conference
Private table database virtualization for dbaas	2011	Fourth IEEE International Conference on Utility and Cloud Computing	Conference

Table A2. SQL and NoSQL Databases—Comparison Studies.

Performance Comparisons	NoSQL Databases
Relational Databases	[3,5–10,12–14,16,17,20,25,27,28,30,37,38,47,56,108]

Table A3. Main strength of Oracle relational database and MongoDB NoSQL database.

Properties	Oracle RDBMS	MongoDB
ACID	X	
BASE		X
Large Data Scalability		X
Data Sharding	X	X
Partitioning	X	X
Replication	X	X
Distributed	X	X
Vertical/Horizontal	Vertical	Horizontal
Schema	Rigid Schema	Schema-less/dynamic schema
Full SQL	X	

Indexing	X	X
Uni-Code Characters	X	X
Built-in MapReduce		X
Maximum Value Size	4 KB	16 MB
Sharing Support		X
Open Source/Licensed	Licensed	Open Source

Table A4. Databases used in each of the selected studies:.

PID	Discussion/Performance Evaluations/Comparisons/Characteristics of SQL & NoSQL databases
[11]	Google Big Table, Amazon SimpleDB, Apache CouchDB, MongoDB, Cassandra, Hbase
[13]	Cassandra, MongoDB, Couchbase
[14]	SQL, Cassandra, CouchDB, DynamoDB, MongoDB, GraphDB
[15]	MySQL, Oracle, SQL Server, PostgreSQL, Sybase, MongoDB, Redis
[23]	MongoDB
[24]	MySQL, MongoDB
[25]	SQL Server, MongoDB
[26]	MongoDB, MySQL
[27]	MongoDB, Oracle
[28]	MongoDB, RavenDB, CouchDB, Cassandra, HyperTable, CouchBase, MS-SQL Server Express
[29]	Oracle, Neo4j
[30]	Neo4j, Oracle
[31]	Oracle, MongoDB
[32]	MySQL, Neo4j
[34]	MongoDB, MySQL
[36]	SQL Server, In-memory TPC databases via HammerDB
[37]	MongoDB, MySQL
[38]	PostgreSQL, MongoDB, Neo4j
[39]	SQL and NoSQL databases
[40]	Oracle 12c, JSON, BSON, OSON
[42]	Open Source Graph Databases
[43]	PostgreSQL, H2 (Open Source lightweight Java RDMS), HBase, JanusGraph
[44]	Oracle 11g, MongoDB
[46]	Neo4j, MySQL
[47]	MongoDB, PostgreSQL
[48]	MongoDB, MySQL, VoltDB for IoT data used in sensor
[49]	MySQL, MongoDB
[50]	SQL to NoSQL MongoDB Migration
[51]	MySQL, MongoDB
[52]	MySQL, MongoDB
[54]	MySQL, MongoDB
[55]	MySQL, MongoDB
[56]	MySQL to MongoDB transformation
[58]	MySQL (JDBC driver), Cassandra (Simba's Cassandra JDBC and ODBC)
[59]	MySQL, MongoDB
[60]	CouchBase, RethinkDB, MongoDB
[61]	MongoDB and Oracle NoSQL
[62]	Dynamo (Amazon), Voldmart (LinkedIn), Redis, BerkeleyDB, Riak, MongoDB, CouchDB, SimpleDB (Amazon), DynamoDB, Neo4j, InfoGrid, Sones GraphDB, Infinite Graph
[63]	MongoDB, Oracle

[64]	CAP, ACID, BASE
[65]	SQL and NoSQL databases characteristics
[67]	CAP, ACID, BASE, NoSQL database categories discussions
[69]	Literature Review on Database Design Testing Techniques (SQL & NoSQL databases)
[71]	ACID Model Databases
[72]	NoSQL BASE Analysis
[73]	SQL & NoSQL Availability, Consistency and Efficiency properties
[74]	SQL ACID & NoSQL BASE properties are discussed
[75]	MySQL, Hbase databases
[76]	NoSQL DBMSs, CAP, Aerospike, Cassandra, CouchDB, MongoDB
[77]	In memory databases: MongoDB, Redis, Memcached, Cassandra, H2
[82]	SQL to NoSQL databases over Hadoop and spark cloud
[83]	PostgreSQL, MongoDB, MariaDB, Hbase Hadoop based analysis
[84]	SQL on Hadoop, Columnar file format, Hive, SparkSQL
[86]	MySQL, MongoDB, Cassandra, 8 de-identified patients datasets
[96]	MySQL, MongoDB, Cassandra
[97]	SQL and NoSQL databases characteristics, IoT, MySQL & MongoDB comparisons
[98]	BASE, IoT, RDBMS, MongoDB, Cassandra
[99]	PostGIS and MongoDB comparisons for spatial data
[100]	PostgreSQL, Oracle, MongoDB in cloud platform for spatial data
[101]	PostgreSQL, MongoDB, Cassandra for web applications
[103]	ArangoDB, OrientDB, Couchbase server characteristics & comparisons, ACID, BASE
[104]	Various databases models for geospatial data
[105]	Heterogeneous data integration models and architectures have been investigated
[108]	Efficient storage data model for GPS application
[110]	Spatial databases, MRDB, Topographic database and WGS have been discussed
[113]	GISB: Geo-information extraction framework
[114]	Spatial databases inconsistencies
[115]	Big geospatial data processing strategies
[116]	MySQL, PostgreSQL, MongoDB, DbSNP database for genomic annotations.
[117]	Investigated general data management platform for high-dimensional spatio-temporal data
[118]	Spatial data standards: OGC OpenGIS and SQL/MM – PostgreSQL +PostGIS & MySQL Spatial
[119]	Oracle 11g database for spatial data
[120]	Azure SQL database, PostGIS, MongoDB, Azure DocumentDB, DBaaS for spatial data
[121]	ACID, BASE, Database modeling & Design, SQL & NoSQL databases characteristics
[122]	NoSQL MongoDB Case study
[123]	Synthetic dataset, NoSQL MongoDB (semi-structured & structured data)
[124]	Security features of MongoDB and Cassandra
[125]	Cloud data portability framework (Unified APIs) for various NoSQL databases

Table A5. Each DBMS with its DBMSID.

DBMSID	DBMS-Name
0	AmazonSimpleDB
1	ApacheCouchDB
2	ArangoDB
3	AzureDocumentDB
4	AzureSQLdatabase
5	Cassandra
6	CoucHBase

7	CouchDB
8	DynamoDB
9	GoogleBigTable
10	GraphDB
11	H2
12	HBase
13	HyperTable
14	JanusGraph
15	MS-SQLServerExpress
16	MariaDB
17	Memcached
18	MongoDB
19	MySQL
20	Neo4j
21	Oracle11g
22	OracleNoSQL
23	OrientDB
24	PostGIS
25	PostgreSQL
26	RavenDB
27	Redis
28	RethinkDB
29	SQL
30	SQLServer
31	Sybase

Table A6. Occurrences of a particular DBMS against other DBMSs based on Table A4 (Unbalanced data).

DBMSID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	Predicted Result
0	0.05	0.05	0.01	0.01	0.02	0.10	0.05	0.05	0.03	0.08	0.03	0.01	0.07	0.04	0.00	0.04	0.00	0.02	0.21	0.01	0.00	0.00	0.00	0.03	0.03	0.01	MongoDB
1	0.05	0.05	0.01	0.01	0.01	0.10	0.05	0.05	0.03	0.07	0.03	0.01	0.07	0.03	0.00	0.04	0.00	0.02	0.21	0.01	0.00	0.00	0.00	0.05	0.03	0.01	MongoDB
2	0.05	0.05	0.01	0.01	0.01	0.10	0.05	0.05	0.03	0.06	0.03	0.01	0.06	0.03	0.01	0.03	0.00	0.02	0.20	0.01	0.00	0.00	0.00	0.07	0.03	0.01	MongoDB
3	0.05	0.05	0.01	0.01	0.01	0.10	0.05	0.05	0.02	0.06	0.03	0.01	0.06	0.03	0.01	0.03	0.00	0.02	0.20	0.01	0.00	0.00	0.00	0.08	0.03	0.01	MongoDB
4	0.05	0.05	0.01	0.01	0.01	0.10	0.05	0.05	0.02	0.05	0.02	0.01	0.06	0.03	0.01	0.03	0.00	0.01	0.20	0.01	0.00	0.00	0.00	0.08	0.03	0.01	MongoDB
5	0.04	0.04	0.01	0.01	0.01	0.11	0.05	0.05	0.02	0.05	0.02	0.02	0.06	0.03	0.01	0.03	0.00	0.02	0.20	0.02	0.00	0.00	0.00	0.06	0.03	0.02	MongoDB
6	0.04	0.04	0.01	0.01	0.01	0.11	0.05	0.05	0.02	0.04	0.02	0.02	0.06	0.03	0.01	0.03	0.00	0.02	0.21	0.02	0.00	0.00	0.00	0.04	0.03	0.02	MongoDB
7	0.04	0.04	0.01	0.01	0.01	0.11	0.05	0.05	0.02	0.04	0.02	0.02	0.06	0.03	0.01	0.03	0.00	0.02	0.21	0.03	0.00	0.00	0.00	0.02	0.03	0.03	MongoDB
8	0.04	0.04	0.01	0.01	0.01	0.12	0.05	0.05	0.02	0.03	0.02	0.02	0.05	0.03	0.01	0.03	0.00	0.02	0.22	0.04	0.00	0.00	0.00	0.01	0.02	0.04	MongoDB
9	0.04	0.03	0.01	0.01	0.01	0.12	0.05	0.05	0.02	0.03	0.02	0.02	0.05	0.03	0.01	0.03	0.01	0.02	0.22	0.04	0.00	0.00	0.00	0.00	0.02	0.04	MongoDB
10	0.03	0.03	0.01	0.01	0.01	0.12	0.05	0.05	0.02	0.03	0.02	0.02	0.05	0.03	0.01	0.03	0.01	0.02	0.22	0.05	0.00	0.00	0.00	0.00	0.02	0.05	MongoDB
11	0.03	0.03	0.01	0.01	0.01	0.12	0.05	0.05	0.02	0.02	0.02	0.03	0.05	0.03	0.01	0.03	0.01	0.02	0.22	0.06	0.00	0.01	0.00	0.00	0.02	0.06	MongoDB
12	0.03	0.02	0.01	0.01	0.01	0.12	0.05	0.04	0.02	0.02	0.02	0.03	0.05	0.03	0.01	0.03	0.01	0.02	0.23	0.07	0.00	0.01	0.00	0.00	0.02	0.07	MongoDB
13	0.02	0.02	0.01	0.01	0.01	0.12	0.04	0.04	0.02	0.02	0.02	0.03	0.05	0.02	0.01	0.03	0.01	0.02	0.23	0.08	0.00	0.02	0.00	0.00	0.02	0.07	MongoDB
14	0.02	0.02	0.01	0.01	0.01	0.11	0.04	0.04	0.02	0.01	0.02	0.03	0.04	0.02	0.01	0.02	0.01	0.01	0.23	0.09	0.00	0.02	0.00	0.00	0.02	0.08	MongoDB
15	0.02	0.01	0.01	0.01	0.01	0.11	0.04	0.04	0.02	0.01	0.02	0.03	0.04	0.02	0.01	0.02	0.01	0.01	0.23	0.10	0.00	0.03	0.00	0.00	0.01	0.09	MongoDB
16	0.01	0.01	0.01	0.01	0.01	0.11	0.04	0.04	0.02	0.01	0.02	0.03	0.04	0.02	0.01	0.02	0.01	0.01	0.22	0.11	0.01	0.04	0.00	0.00	0.01	0.09	MongoDB
17	0.01	0.01	0.01	0.01	0.01	0.10	0.04	0.04	0.02	0.01	0.01	0.03	0.04	0.02	0.01	0.02	0.02	0.01	0.22	0.11	0.02	0.05	0.00	0.00	0.01	0.10	MongoDB
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.99	0.00	0.00	0.00	OracleNoSQL
19	0.01	0.01	0.01	0.01	0.01	0.09	0.04	0.03	0.01	0.01	0.01	0.03	0.03	0.02	0.01	0.02	0.02	0.01	0.21	0.12	0.04	0.07	0.00	0.00	0.01	0.10	MongoDB
20	0.00	0.00	0.01	0.01	0.01	0.08	0.03	0.03	0.01	0.00	0.01	0.03	0.03	0.01	0.01	0.01	0.02	0.01	0.21	0.12	0.06	0.08	0.00	0.00	0.01	0.10	MongoDB
21	0.00	0.00	0.01	0.01	0.01	0.08	0.03	0.03	0.01	0.00	0.01	0.03	0.03	0.01	0.01	0.01	0.02	0.01	0.20	0.12	0.06	0.09	0.00	0.00	0.01	0.10	MongoDB
22	0.00	0.00	0.01	0.01	0.01	0.07	0.03	0.03	0.01	0.00	0.01	0.02	0.03	0.01	0.01	0.01	0.01	0.01	0.21	0.12	0.06	0.10	0.00	0.00	0.01	0.09	MongoDB
23	0.00	0.00	0.01	0.01	0.01	0.07	0.03	0.03	0.01	0.00	0.01	0.02	0.03	0.01	0.01	0.01	0.01	0.01	0.21	0.12	0.05	0.10	0.00	0.00	0.01	0.09	MongoDB
24	0.00	0.00	0.01	0.01	0.01	0.07	0.03	0.03	0.01	0.00	0.01	0.02	0.03	0.01	0.01	0.01	0.01	0.01	0.22	0.12	0.03	0.11	0.00	0.00	0.00	0.09	MongoDB
25	0.00	0.00	0.01	0.01	0.01	0.07	0.03	0.02	0.01	0.00	0.01	0.02	0.03	0.01	0.01	0.01	0.01	0.01	0.22	0.11	0.02	0.11	0.00	0.00	0.00	0.09	MongoDB

Table A7. Occurrences of a particular DBMS against other DBMSs based on Table A4 data and generated data (Balanced data).

DBMSID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	Predicted Result
0	0.04	0.04	0.03	0.03	0.03	0.01	0.04	0.04	0.04	0.04	0.04	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.02	0.00	0.03	0.03	0.03	0.04	0.04	0.03	GoogleBigTable
1	0.04	0.04	0.03	0.03	0.03	0.01	0.04	0.04	0.04	0.04	0.04	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.04	0.04	0.03	GoogleBigTable
2	0.04	0.04	0.03	0.03	0.03	0.01	0.04	0.04	0.04	0.04	0.04	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.04	0.04	0.03	CouchDB
3	0.04	0.04	0.03	0.03	0.03	0.01	0.04	0.04	0.03	0.04	0.04	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.04	0.04	CouchDB
4	0.04	0.04	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.04	0.03	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.03	0.04	CouchDB
5	0.04	0.04	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.04	0.03	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.03	0.04	CouchDB
6	0.04	0.04	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.04	0.03	0.03	0.01	0.04	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.03	0.04	PostgreSQL
7	0.04	0.04	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.04	0.03	0.03	0.01	0.03	0.03	0.04	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.03	0.04	PostgreSQL
8	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	PostgreSQL
9	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.01	0.03	0.03	0.03	0.03	0.03	0.04	PostgreSQL
10	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.01	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
11	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.01	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
12	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.01	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
13	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.01	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
14	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
15	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
16	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
17	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
18	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
19	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
20	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
21	0.03	0.03	0.03	0.03	0.03	0.02	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
22	0.03	0.03	0.03	0.03	0.03	0.01	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
23	0.03	0.03	0.03	0.03	0.03	0.01	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
24	0.03	0.03	0.03	0.03	0.03	0.01	0.04	0.04	0.03	0.03	0.03	0.03	0.01	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL
25	0.03	0.03	0.03	0.03	0.03	0.01	0.04	0.04	0.03	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.03	0.03	0.04	0.02	0.03	0.04	0.03	0.03	0.03	0.04	PostgreSQL

References

1. Siddiqa, A.; Hashem, I.A.T.; Yaqoob, I.; Marjani, M.; Shamshirband, S.; Gani, A.; Nasaruddin, F. A survey of big data management: Taxonomy and state-of-the-art. *J. Netw. Comput. Appl.* **2016**, *71*, 151–166.
2. Kong, X.; Shi, Y.; Yu, S.; Liu, J.; Xia, F. Academic social networks: Modeling, analysis, mining and applications. *J. Netw. Comput. Appl.* **2019**, *132*, 86–103.
3. Ordonez, C. Optimization of Linear Recursive Queries in SQL. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 264–277.
4. Obasanjo, D. Building scalable Databases: Denormalization, the NoSQL movement and Digg. 2009.
5. Strozzi, C. NoSQL–A relational Database management system. 2007–2010. Available online: http//www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/nosql/Home% 20Page (accessed on 13 November 2019).
6. George, S. NoSQL–NOT ONLY SQL. *Int. J. Enterp. Comput. Bus. Syst.* **2013**, *2*.
7. Brewer, E.A. Towards robust distributed systems. In *PODC*; Inktomi: Foster City, CA, USA, 2000; Volume 7.
8. Díaz, M.; Martín, C.; Rubio, B. State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing. *J. Netw. Comput. Appl.* **2016**, *67*, 99–117.
9. Rao, B.T. A Study on Data Storage Security Issues in Cloud Computing. *Procedia Comput. Sci.* **2016**, *92*, 128–135.
10. Mansouri, Y.; Prokhorenko, V.; Babar, M.A. An automated implementation of hybrid cloud for performance evaluation of distributed databases. *J. Netw. Comput. Appl.* **2020**, *167*, 102740. https://doi.org/10.1016/j.jnca.2020.102740.
11. Ravi, K.; Khandelwal, Y.; Krishna, B.S.; Ravi, V. Analytics in/for cloud-an interdependence: A review. *J. Netw. Comput. Appl.* **2018**, *102*, 17–37.
12. Wiese, L.; Waage, T.; Brenner, M. CloudDBGuard: A framework for encrypted data storage in NoSQL wide column stores. *Data Knowl. Eng.* **2019**, *126*, 101732.
13. Ribas, M.; Furtado, C.; de Souza, J.N.; Barroso, G.C.; Moura, A.; Lima, A.S.; Sousa, F.R. A Petri net-based decision-making framework for assessing cloud services adoption: The use of spot instances for cost reduction. *J. Netw. Comput. Appl.* **2015**, *57*, 102–118.
14. Kumari, A.; Tanwar, S.; Tyagi, S.; Kumar, N.; Parizi, R.M.; Choo, K.-K.R. Fog data analytics: A taxonomy and process model. *J. Netw. Comput. Appl.* **2019**, *128*, 90–104.
15. Kitchenham, B.; Charters, S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*; Keele University: Keele, UK, 2007.
16. Dyba, T.; Kitchenham, B.A.; Jorgensen, M. Evidence-based software engineering for practitioners. *IEEE Softw.* **2005**, *22*, 58–65.
17. Hosseinzadeh, S.; Rauti, S.; Laurén, S.; Mäkelä, J.-M.; Holvitie, J.; Hyrynsalmi, S.; Leppänen, V. Diversification and obfuscation techniques for software security: A systematic literature review. *Inf. Softw. Technol.* **2018**, *104*, 72–93.
18. Petersen, K.; Vakkalanka, S.; Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Inf. Softw. Technol.* **2015**, *64*, 1–18.
19. Badampudi, D.; Wohlin, C.; Petersen, K. Experiences from using snowballing and Database searches in systematic literature studies. In Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering, Nanjing, China, 27–29 April 2015; p. 17.
20. Petersen, K.; Gencel, C. Worldviews, research methods, and their relationship to validity in empirical software engineering research. In Proceedings of the 2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement, Ankara, Turkey, 23–26 October 2013; pp. 81–89.
21. Maxwell, J. Understanding and validity in qualitative research. *Harv. Educ. Rev.* **1992**, *62*, 279–301.
22. Alsolai, H.; Roper, M. A systematic literature review of machine learning techniques for software maintainability prediction. *Inf. Softw. Technol.* **2020**, *119*, 106214.
23. Rodrigues, M.; Santos, M.Y.; Bernardino, J. Big data processing tools: An experimental performance evaluation. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2019**, *9*, e1297.
24. Hou, B.; Qian, K.; Li, L.; Shi, Y.; Tao, L.; Liu, J. MongoDB NoSQL injection analysis and detection. In Proceedings of the 2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud), Beijing, China, 25–27 June 2016; pp. 75–78.
25. Padhy, R.P.; Patra, M.R.; Satapathy, S.C. RDBMS to NoSQL: Reviewing some next-generation non-relational Database's. *Int. J. Adv. Eng. Sci. Technol.* **2011**, *11*, 15–30.
26. Győrödi, C.; Győrödi, R.; Pecherle, G.; Olah, A. A comparative study: MongoDB vs. MySQL. In Proceedings of the 2015 13th International Conference on Engineering of Modern Electric Systems (EMES), Oradea, Romania, 11–12 June 2015; pp. 1–6.
27. Băzăr, C.; Iosif, C.S. The transition from rdbms to nosql. a comparative analysis of three popular non-relational solutions: Cassandra, mongodb and couchbase. *Database Syst. J.* **2014**, *5*, 49–59.
28. Mukherjee, S. *The Battle between NoSQL Databases and RDBMS*; University of the Cumberlands: Williamsburg, KY, USA, 2019.
29. Chopade, R.; Pachghare, V.K. Ten years of critical review on database forensics research. *Digit. Investig.* **2019**, *29*, 180–197.
30. Kitchenham, B.; Brereton, P. A systematic review of systematic review process research in software engineering. *Inf. Softw. Technol.* **2013**, *55*, 2049–2075.
31. Imam, A.A.; Basri, S.; Ahmad, R.; González-Aparicio, M.T. Literature Review on Database Design Testing Techniques. In Proceedings of the Computer Science Online Conference, Faro, Portugal, 2019; pp. 1–13.

32. Han, J.; Haihong, E.; Le, G.; Du, J. Survey on NoSQL Database. In Proceedings of the 2011 6th International Conference on Pervasive Computing and Applications, Port Elizabeth, South Africa, 26–28 October 2011, 363–366.
33. Stonebraker, M. SQL Databases v. NoSQL Databases. *Commun. ACM* **2012**, *53*, 10–11.
34. Parker, Z.; Poe, S.; Vrbsky, S.V. Comparing nosql mongodb to an sql db. In Proceedings of the 51st ACM Southeast Conference, Savannah, GA, USA, 4–6 April 2013; p. 5.
35. Wei-Ping, Z.; Ming-Xin, L.I.; Huan, C. Using MongoDB to implement textbook management system instead of MySQL. In Proceedings of the 2011 IEEE 3rd International Conference on Communication Software and Networks, Xian, China, 27–29 May 2011; pp. 303–305.
36. Boicea, A.; Radulescu, F.; Agapin, L.I. MongoDB vs. Oracle-Database comparison. In 2012 Third International Conference on Emerging Intelligent Data and Web Technologies, Bucharest, Romania, 19–21 September 2012; pp. 330–335.
37. Li, Y.; Manoharan, S. A performance comparison of SQL and NoSQL Databases. In Proceedings of the 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), Victoria, BC, Canada, 27–29 August 2013; pp. 15–19.
38. Khan, W.; Ahmad, W.; Luo, B.; Ahmed, E. SQL Database with physical Database tuning technique and NoSQL graph Database comparisons. In Proceedings of the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 15–17 March 2019. https://doi.org/10.1109/ITNEC.2019.8729264.
39. Khan, W.; Shahzad, W. Predictive Performance Comparison Analysis of Relational & NoSQL Graph Databases. *Int. J. Adv. Comput. Sci. Appl.* **2017**, *8*, 523–530.
40. Faraj, A.; Rashid, B.; Shareef, T. Comparative study of relational and non-relations Database performances using Oracle and MongoDB systems. *J. Impact Factor* **2014**, *5*, 11–22.
41. Vicknair, C.; Macias, M.; Zhao, Z.; Nan, X.; Chen, Y.; Wilkins, D. A comparison of a graph Database and a relational Database: A data provenance perspective. In Proceedings of the 48th annual Southeast regional conference, Oxfrod, MS, USA, 15–17 April 2010; p. 42.
42. Khan, S.; Mane, V. SQL support over MongoDB using metadata. *Int. J. Sci. Res. Publ.* **2013**, *3*, 1–5.
43. Kumar, L.; Rajawat, S.; Joshi, K. Comparative analysis of nosql (mongodb) with mysql Database. *Int. J. Mod. Trends Eng. Res.* **2015**, *2*, 120–127.
44. Aghi, R.; Mehta, S.; Chauhan, R.; Chaudhary, S.; Bohra, N. A comprehensive comparison of SQL and MongoDB Databases. *Int. J. Sci. Res. Publ.* **2015**, *5*, 1–3.
45. Ayub, M.B.; Ali, N. Performance comparison of in-memory and disk-based Databases using transaction processing performance council (TPC) benchmarking. *J. Internet Inf. Syst.* **2018**, *8*, 1–8.
46. Deari, R.; Zenuni, X.; Ajdari, J.; Ismaili, F.; Raufi, B. Analysis and Comparison of Document-Based Databases with Sql Relational Databases: Mongodb vs. Mysql. In Proceedings of the International Conference on Information Technologies (InfoTech-2018), Varna, Bulgaria, 20–21 September 2018.
47. Sharma, M.; Sharma, V.D.; Bundele, M.M. Performance Analysis of RDBMS and No SQL Databases: PostgreSQL, MongoDB and Neo4j. In Proceedings of the 2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE), Rajasthan, India, 22–25 November 2018; pp. 1–5.
48. Čerešňák, R.; Kvet, M. Comparison of query performance in relational a non-relation Databases. *Transp. Res. Procedia* **2019**, *40*, 170–177.
49. Liu, Z.H.; Hammerschmidt, B.; McMahon, D.; Liu, Y.; Chang, H.J. Closing the functional and performance gap between SQL and NoSQL. In Proceedings of the 2016 International Conference on Management of Data, San Francisco, CA, USA, 26 June-1 July 2016; pp. 227–238.
50. Kim, H.-J.; Ko, E.-J.; Jeon, Y.-H.; Lee, K.-H. Migration from rdbms to column-oriented nosql: Lessons learned and open problems. In Proceedings of the 7th International Conference on Emerging Databases, Panevezys, Lithuania, 26–27 April 2018; pp. 25–33.
51. McColl, R.C.; Ediger, D.; Poovey, J.; Campbell, D.; Bader, D.A. A performance evaluation of open source graph Databases. In Proceedings of the First Workshop on Parallel Programming for Analytics Applications, Orlando, FL, USA, 16 February 2014; pp. 11–18.
52. Anikin, D.; Borisenko, O.; Nedumov, Y. Labeled Property Graphs: SQL or NoSQL? In Proceedings of the 2019 Ivannikov Memorial Workshop (IVMEM), Velikiy Novgorod, Russia, 13–14 September 2019; pp. 7–13.
53. El Mouden, Z.A.; Jakimi, A.; Hajar, M.; Boutahar, M. Graph Schema Storage in SQL Object-Relational Database and NoSQL Document-Oriented Database: A Comparative Study. In Proceedings of the International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning, Marrakech, Morocco, 21–23 November 2019; pp. 176–183.
54. Rathika, V. Graph-Based Denormalization for Migrating Big Data from SQL Database to NoSQL Database. In Proceedings of the Intelligent Communication Technologies and Virtual Mobile Networks, Tirunelveli, India, 14–15 February 2019; pp. 546–556.
55. Zhu, Y.; Yan, E.; Song, I. The use of a graph-based system to improve bibliographic information retrieval: System design, implementation, and evaluation. *J. Assoc. Inf. Sci. Technol.* **2017**, *68*, 480–490.

56. Jung, M.-G.; Youn, S.-A.; Bae, J.; Choi, Y.-L. A study on data input and output performance comparison of MongoDB and PostgreSQL in the big data environment. In Proceedings of the 2015 8th International Conference on Database Theory and Application (DTA), Jeju Island, Republic of Korea, 28–25 November 2015; pp. 14–17.
57. Fatima, H.; Wasnik, K. Comparison of SQL, NoSQL and NewSQL Databases for internet of things. In Proceedings of the 2016 IEEE Bombay Section Symposium (IBSS), Maharashtra, India, 21–22 December 2016; pp. 1–6.
58. Ray, P.P.; Dash, D.; De, D. Edge computing for Internet of Things: A survey, e-healthcare case study and future direction. *J. Netw. Comput. Appl.* **2019**, *140*, 1–22.
59. Singh, A. Data Migration from Relational Database to MongoDB. *Glob. J. Comput. Sci. Technol.* **2019**.
60. Zhao, G.; Huang, W.; Liang, S.; Tang, Y. "Modeling MongoDB with relational model. In Proceedings of the 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies, Washington, DC, USA, 9–11 September 2013; pp. 115–121.
61. Stanescu, L.; Brezovan, M.; Burdescu, D.D. Automatic mapping of MySQL Databases to NoSQL MongoDB. In Proceedings of the 2016 Federated Conference on Computer Science and Information Systems (FedCSIS), Gdansk, Poland, 11–14 September 2016; pp. 837–840.
62. Yassine, F.; Awad, M.A. Migrating from SQL to NOSQL Database: Practices and Analysis. In Proceedings of the 2018 International Conference on Innovations in Information Technology (IIT), Al Ain, United Arab Emirates, 18–19 November 2018; pp. 58–62.
63. Liao, Y.-T.; Zhou, J.; Lu, C.-H.; Chen, S.-C.; Hsu, C.-H.; Chen, W.; Jiang, M.-F.; Chung, Y.-C. Data adapter for querying and transformation between SQL and NoSQL database. *Futur. Gener. Comput. Syst.* **2016**, *65*, 111–121.
64. Tomar, D.; Bhati, J.P.; Tomar, P.; Kaur, G. Migration of healthcare relational Database to NoSQL cloud Database for healthcare analytics and management. In *Healthcare Data Analytics and Management*; Elsevier: Amsterdam, The Netherlands, 2019; pp. 59–87.
65. Rocha, L.; Vale, F.; Cirilo, E.; Barbosa, D.; Mourão, F. A Framework for Migrating Relational Datasets to NoSQL 1. *Procedia Comput. Sci.* **2015**, *51*, 2593–2602.
66. Ghule, S.; Vadali, R. Transformation of SQL system to NoSQL system and performing data analytics using SVM. In Proceedings of the 2017 International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli, India, 11–12 May 2017; pp. 883–887.
67. Hsu, J.C.; Hsu, C.H.; Chen, S.C.; Chung, Y.C. Correlation Aware Technique for SQL to NoSQL Transformation. In Proceedings of the 2014 7th International Conference on Ubi-Media Computing and Workshops, Washington, DC, USA, 12–14 July 2014; pp. 43–46.
68. Solanke, G.B.; Rajeswari, K. SQL to NoSQL transformation system using data adapter and analytics. In Proceedings of the 2017 IEEE International Conference on Technological Innovations in Communication, Control and Automation (TICCA), Chennai, India, 6 April 2017; pp. 59–63.
69. Lawrence, R. Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB. In Proceedings of the 2014 International Conference on Computational Science and Computational Intelligence, Kunming, China, 15–16 November 2014; Volume 1, pp. 285–290.
70. Pereira, D.A.; de Morais, W.O.; Pignaton de Freitas, E. NoSQL real-time Database performance comparison. *Int. J. Parallel Emergent Distrib. Syst.* **2018**, *33*, 144–156.
71. Anand, V.; Rao, C.M. MongoDB and Oracle NoSQL: A technical critique for design decisions. In Proceedings of the 2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS), Pudukkottai, India, 24–16 April 2016; pp. 1–4.
72. Moniruzzaman, A.B.M.; Hossain, S.A. Nosql Database: New era of Databases for big data analytics-classification, characteristics and comparison. *arXiv* **2013**, arXiv:1307.0191.
73. Simanjuntak, H.T.A.; Simanjuntak, L.; Situmorang, G.; Saragih, A. Query Response Time Comparison NOSQLDB MONGODB with SQLDB Oracle. *JUTI J. Ilm. Teknol. Inf.* **2015**, *13*, 95–105.
74. Almeida, A.L.; Schettino, V.J.; Barbosa, T.J.R.; Freitas, P.F.; Guimarães, P.G.S.; Arbex, W.A. Relative scalability of NoSQL Databases for genotype data manipulation. *Embrapa Gado Leite-Artig. Periódico Indexado* **2018**, *25*, 93–100.
75. Cattell, R. Scalable SQL and NoSQL data stores. *ACM SIGMOD Rec.* **2011**, *39*, 12–27.
76. Lee, C.-H.; Zheng, Y.-L. SQL-to-NoSQL schema denormalization and migration: A study on content management systems. In Proceedings of the 2015 IEEE International Conference on Systems, Man, and Cybernetics, Hong Kong, China, 9–12 October 2015; pp. 2022–2026.
77. Meier, A.; Kaufmann, M.; Meier, A.; Kaufmann, M. Nosql databases. In *SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management*; Springer Vieweg: Wiesbaden, Germany, 2019; pp. 201–218.
78. Pokorný, J. Integration of Relational and NoSQL Databases. In Proceedings of the Asian Conference on Intelligent Information and Database Systems, Dong Hoi, Vietnam, 19–21 March 2018; pp. 35–45.
79. Miranskyy, A.V.; Al-zanbouri, Z.; Godwin, D.; Bener, A.B. Database engines: Evolution of greenness. *J. Softw. Evol. Process.* **2017**, *30*, e1915.
80. Chapple, M. The Acid Model. Available online: http//Databases.about.com/od/specificproducts/a/acid.htm (accessed on 26 February 2020).
81. Chandra, D.G. BASE analysis of NoSQL database. *Futur. Gener. Comput. Syst.* **2015**, *52*, 13–21.

82. Gonzalez-Aparicio, M.T.; Younas, M.; Tuya, J.; Casado, R. Evaluation of ACE properties of traditional SQL and NoSQL big data systems. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 8–12 April 2019; pp. 1988–1995.
83. Sun, H.; Xiao, B.; Wang, X.; Liu, X. Adaptive trade-off between consistency and performance in data replication. *Softw. Pract. Exp.* **2017**, *47*, 891–906.
84. Lee, C.-H.; Zheng, Y.-L. Automatic SQL-to-NoSQL schema transformation over the MySQL and HBase Databases. In Proceedings of the 2015 IEEE International Conference on Consumer Electronics-Taiwan, Taipei, Taiwan, 6–8 June 2015; pp. 426–427.
85. Kuzochkina, A.; Shirokopetleva, M.; Dudar, Z. Analyzing and Comparison of NoSQL DBMS. In Proceedings of the 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, Ukraine, 9–12 October 2018; pp. 560–564.
86. Kabakus, A.T.; Kara, R. A performance evaluation of in-memory databases. *J. King Saud Univ. Inf. Sci.* **2017**, *29*, 520–525.
87. Dean, J.; Ghemawat, S. MapReduce: Simplified data processing on large clusters. *Commun. ACM* **2008**, *51*, 107–113.
88. Pokorný, J. Database technologies in the world of big data. In Proceedings of the 16th International Conference on Computer Systems and Technologies, Dublin, Ireland, 25–26 June 2015; pp. 1–12.
89. Yang, H.; Dasdan, A.; Hsiao, R.-L.; Parker, D.S. Map-reduce-merge: Simplified relational data processing on large clusters. In Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2007; pp. 1029–1040.
90. Nykiel, T.; Potamias, M.; Mishra, C.; Kollios, G.; Koudas, N. MRShare: Sharing across multiple queries in MapReduce. *Proc. VLDB Endow.* **2010**, *3*, 494–505.
91. Lee, C.-H.; Shih, Z.-W. A Comparison of NoSQL and SQL Databases over the Hadoop and Spark Cloud Platforms using Machine Learning Algorithms. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Taiwan, 19–21 May 2018; pp. 1–2.
92. Son, S.; Gil, M.-S.; Moon, Y.-S.; Won, H.-S. Performance Analysis of Hadoop-Based SQL and NoSQL for Processing Log Data. In Proceedings of the International Conference on Database Systems for Advanced Applications, Hanoi, Vietnam, 20–23 April 2015; pp. 293–299.
93. Ivanov, T.; Pergolesi, M. The impact of columnar file formats on SQL-on-hadoop engine performance: A study on ORC and Parquet. *Concurr. Comput. Pract. Exp.* **2019**, *32*, e5523.
94. Diaz, F.; Freato, R. Working with NoSQL Alternatives. In *Cloud Data Design, Orchestration, and Management Using Microsoft Azure*; Springer: Cham, Switzerland, 2018; pp. 169–262.
95. Zeng, N.; Zhang, G.-Q.; Li, X.; Cui, L. Evaluation of relational and NoSQL approaches for patient cohort identification from heterogeneous data sources. In Proceedings of the 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Kansas City, MO, USA, 13–16 November 2017; pp. 1135–1140.
96. Lee, J.-G.; Kang, M. Geospatial Big Data: Challenges and Opportunities. *Big Data Res.* **2015**, *2*, 74–81.
97. Liu, Z.; Guo, H.; Wang, C. Considerations on Geospatial Big Data. *IOP Conf. Series: Earth Environ. Sci.* **2016**, *46*, 012058.
98. Albert, A.; Kaur, J.; Gonzalez, M.C. Using Convolutional Networks and Satellite Imagery to Identify Patterns in Urban Environments at a Large Scale. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 1357–1366.
99. Ahlers, D.; Wilde, E. Report on the Seventh International Workshop on Location and the Web (LocWeb 2017). *ACM SIGIR Forum* **2017**, *51*, 52–57.
100. Bari, N.; Mani, G.; Berkovich, S. Internet of things as a methodological concept. In Proceedings of the 2013 Fourth International Conference on Computing for Geospatial Research and Application, San Jose, CA, USA, 22–24 July 2013; pp. 48–55.
101. Mendoza, M.; Poblete, B.; Castillo, C. Twitter under crisis: Can we trust what we RT? In Proceedings of the First Workshop on Social Media Analytics, Washington, DC, USA, 25 July 2010; pp. 71–79.
102. Aubrecht, C.; Meier, P.; Taubenböck, H. Speeding up the clock in remote sensing: Identifying the 'black spots' in exposure dynamics by capitalizing on the full spectrum of joint high spatial and temporal resolution. *Nat. Hazards* **2017**, *86*, 177–182.
103. McCoy, M.D. Geospatial Big Data and archaeology: Prospects and problems too great to ignore. *J. Archaeol. Sci.* **2017**, *84*, 74–94.
104. Burzańska, M.; Wiśniewski, P. How Poor Is the 'Poor Man's Search Engine'? In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Poznań, Poland, 18–20 September 2018; pp. 294–305.
105. Harezlak, K.; Skowron, R. Performance aspects of migrating a web application from a relational to a NoSQL Database. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustroń, Poland, 26–29 May 2015; pp. 107–115.
106. Rautmare, S.; Bhalerao, D.M. MySQL and NoSQL Database comparison for IoT application. In Proceedings of the 2016 IEEE International Conference on Advances in Computer Applications (ICACA), Coimbatore, India, 24 October 2016; pp. 235–238.
107. Aya, A.-S.; Qattous, H.; Hijjawi, M. A proposed performance evaluation of NoSQL Databases in the field of IoT. In Proceedings of the 2018 8th International Conference on Computer Science and Information Technology (CSIT), Amman, Jordan, 11–12 July 2018; pp. 32–37.
108. Bartoszewski, D.; Piorkowski, A.; Lupa, M. The comparison of processing efficiency of spatial data for PostGIS and MongoDB databases. In Proceedings of the Beyond Databases, Architectures and Structures. Paving the Road to Smart Data Processing and Analysis: 15th International Conference, BDAS 2019, Ustroń, Poland, 28–31 May 2019; pp. 291–302.

109. Tear, A. SQL or NoSQL? Contrasting approaches to the storage, manipulation and analysis of spatio-temporal online social network data. In Proceedings of the International Conference on Computational Science and Its Applications, Guimaraes, Portugal, 30 June–3 July 2014; pp. 221–236.
110. Fraczek, K.; Plechawska-Wojcik, M. Comparative analysis of relational and non-relational Databases in the context of performance in web applications. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustroń, Poland, 30 May–2 June 2017; pp. 153–164.
111. Hricov, R.; Šenk, A.; Kroha, P.; Valenta, M. Evaluation of XPath queries over XML documents using SparkSQL framework. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustroń, Poland, 30 May–2 June 2017; pp. 28–41.
112. Płuciennik, E.; Zgorzałek, K. The multi-model Databases–A review. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustroń, Poland, 30 May–2 June 2017; pp. 141–152.
113. Yue, P.; Tan, Z. 1.06 GIS Databases and NoSQL Databases. *Compr. Geogr. Inf. Syst.* **2017**, *50*.
114. Longley, P.A.; Goodchild, M.F.; Maguire, D.J.; Rhind, D.W. *Geographic Information Systems and Science*; John Wiley & Sons: Hoboken, NJ, USA, 2005.
115. Bajerski, P.; Kozielski, S. Computational model for efficient processing of geofield queries. In *Man-Machine Interactions*; Springer: Cham, Switzerland, 2009; pp. 573–583.
116. Chromiak, M.; Stencel, K. A data model for heterogeneous data integration architecture. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, 2014, Ustron, Poland, 27–30 May 2014; pp. 547–556.
117. Akulakrishna, P.K.; Lakshmi, J.; Nandy, S.K. Efficient storage of big-data for real-time gps applications. In Proceedings of the 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, Sydney, Australia, 3–5 December 2014; pp. 1–8.
118. Lupa, M.; Kozioł, K.; Leśniak, A. An attempt to automate the simplification of building objects in multiresolution Databases. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustroń, Poland, 26–29 May 2015; pp. 448–459.
119. Kozioł, K.; Lupa, M.; Krawczyk, A. The extended structure of multi-resolution Database. In Proceedings of the International Conference: Beyond Databases, Architectures and Structures, Ustron, Poland, 27–30 May 2014; pp. 435–443.
120. Wyszomirski, M. Przegląd możliwości zastosowania wybranych baz danych NoSQL do zarządzania danymi przestrzennymi. *Rocz. Geomatyki-Ann. Geomat.* **2018**, *16*, 55–69.
121. Czerepicki, A. Perspektywy zastosowania baz danych NoSQL w inteligentnych systemach transportowych. *Pr. Nauk. Politech. Warsz. Transp.* **2013**, *92*, 29–38.
122. Martins, P.; Cecílio, J.; Abbasi, M.; Furtado, P. GISB: A benchmark for geographic map information extraction. In *Beyond Databases, Architectures and Structures. Advanced Technologies for Data Mining and Knowledge Discovery*; Springer: Cham, Switzerland, 2015; pp. 600–609.
123. Inglot, A.; Koziol, K. The importance of contextual topology in the process of harmonization of the spatial Databases on example BDOT500. In Proceedings of the 2016 Baltic Geodetic Congress (BGC Geomatics), Gdansk, Poland, 2–4 June 2016; pp. 251–256.
124. Chuchro, M.; Franczyk, A.; Dwornik, M.; Leśniak, A. A Big Data processing strategy for hybrid interpretation of flood embankment multisensor data. *Geol. Geophys. Environ.* **2016**, *42*, 269–277.
125. Schulz, W.L.; Nelson, B.G.; Felker, D.K.; Durant, T.J.S.; Torres, R. Evaluation of relational and NoSQL Database architectures to manage genomic annotations. *J. Biomed. Inform.* **2016**, *64*, 288–295.
126. Lian, J.; Miao, S.; McGuire, M.; Tang, Z. SQL or NoSQL? Which Is the Best Choice for Storing Big Spatio-Temporal Climate Data? In Proceedings of the International Conference on Conceptual Modeling, Xi'an, China, 22–25 October 2018; pp. 275–284.
127. Piórkowski, A. Mysql spatial and postgis–implementations of spatial data standards. *EJPAU* **2011**, *14*, 3.
128. Kothuri, R.; Godfrind, A.; Beinat, E. *Pro Oracle Spatial for Oracle Database 11g*; Dreamtech Press: New Delhi, India, 2008.
129. Baralis, E.; Dalla Valle, A.; Garza, P.; Rossi, C.; Scullino, F. SQL versus NoSQL Databases for geospatial applications. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; pp. 3388–3397.
130. Roy-Hubara, N.; Sturm, A. Exploring the Design Needs for the New Database Era. In *Enterprise, Business-Process and Information Systems Modeling*; Springer: Cham, Switzerland, 2018; pp. 276–290.
131. Yoon, J.; Jeong, D.; Kang, C.-H.; Lee, S. Forensic investigation framework for the document store NoSQL DBMS: MongoDB as a case study. *Digit. Investig.* **2016**, *17*, 53–65.
132. Mehmood, E.; Anees. T. Performance Analysis of Not Only SQL Semi-Stream Join Using MongoDB for Real-Time Data Warehousing. *IEEE Access* **2019**, *7*, 134215–134225.
133. Okman, L.; Gal-Oz, N.; Gonen, Y.; Gudes, E.; Abramov, J. Security issues in nosql databases. In Proceedings of the 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, Changsha, China, 16 November 201; pp. 541–547. IEEE.
134. Alomari, E.; Barnawi, A.; Sakr, S. CDPort: A Portability Framework for NoSQL Datastores. *Arab. J. Sci. Eng.* **2015**, *40*, 2531–2553.
135. Alomari, E.; Noaman, A. SeCloudDB: A Unified API for Secure SQL and NoSQL Cloud Databases. In Proceedings of the 2019 3rd International Conference on Cloud and Big Data Computing, Oxford, UK, 28–30 August 2019; pp. 38–42.
136. Stravoskoufos, K.; Preventis, A.; Sotiriadis, S.; Petrakis, E.G.M. A Survey on Approaches for Interoperability and Portability of Cloud Computing Services. In *CLOSER*; Technical University of Crete: Chania, Greece, 2014; pp. 112–117.

137. Shirazi, M.N.; Kuan, H.C.; Dolatabadi, H. Design patterns to enable data portability between clouds' Databases. In Proceedings of the 2012 12th International Conference on Computational Science and Its Applications, Salvador, Bahia, 18–21 June 2012; pp. 117–120.
138. Alomari, E.; Barnawi, A.; Sakr, S. Cdport: A framework of data portability in cloud platforms. In Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services, Singapore, 28–30 November 2014; pp. 126–133.
139. Indu, I.; PM, R.A.; Bhaskar, V. Encrypted token based authentication with adapted SAML technology for cloud web services. *J. Netw. Comput. Appl.* **2017**, *99*, 131–145.
140. Vanelli, B.; da Silva, M.P.; Manerichi, G.; Pinto, A.S.R.; Dantas, M.A.R.; Ferrandin, M.; Boava, A. Internet of Things Data Storage Infrastructure in the Cloud Using NoSQL Databases. *IEEE Lat. Am. Trans.* **2017**, *15*, 737–743.
141. Grolinger, K.; Higashino, W.A.; Tiwari, A.; Capretz, M.A. Data management in cloud environments: NoSQL and NewSQL data stores. *J. Cloud Comput. Adv. Syst. Appl.* **2013**, *2*, 22.
142. Marston, S.; Li, Z.; Bandyopadhyay, S.; Zhang, J.; Ghalsasi, A. Cloud computing—The business perspective. *Decis. Support Syst.* **2011**, *51*, 176–189.
143. Kolev, B.; Bondiombouy, C.; Valduriez, P.; Jiménez-Peris, R.; Pau, R.; Pereira, J. The cloudmdsql multistore system. In Proceedings of the 2016 International Conference on Management of Data, San Francisco, CA, USA, 26 June-1 July 2016; pp. 2113–2116.
144. Loutas, N.; Kamateri, E.; Tarabanis, K. A semantic Interoperability framework for cloud platform as a service. In Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science, Washington, DC, USA, 29 November–1 December 2011; pp. 280–287.
145. Zhou, L.; Fu, A.; Yu, S.; Su, M.; Kuang, B. Data integrity verification of the outsourced big data in the cloud environment: A survey. *J. Netw. Comput. Appl.* **2018**, *122*, 1–15.
146. Kostoska, M.; Gusev, M.; Ristov, S.; Kiroski, K. Cloud Computing Interoperability Approaches-Possibilities and Challenges. In *BCI*; Ss. Cyril and Methodius University: Skopje, Macedonia, 2012; pp. 30–34.
147. Loutas, N.; Kamateri, E.; Bosi, F.; Tarabanis, K. Cloud computing Interoperability: The state of play. In Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science, Washington, DC, USA, 29 November–1 December 2011; pp. 752–757.
148. Escalera, M.F.P.; Chavez, M.A.L. UML model of a standard API for cloud computing application development. In Proceedings of the 2012 9th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico, 26–28 September 2012; pp. 1–8.
149. Petcu, D.; Martino, B.D.; Venticinque, S.; Rak, M.; Máhr, T.; Lopez, G.; Brito, F.; Cossu, R.; Stopar, M.; Šperka, S.; et al. Experiences in building a mOSAIC of clouds. *J. Cloud Comput. Adv. Syst. Appl.* **2013**, *2*, 12.
150. Di Nitto, E.; da Silva, M.A.A.; Ardagna, D.; Casale, G.; Craciun, C.D.; Ferry, N.; Muntes, V.; Solberg, A. Supporting the development and operation of multi-cloud applications: The modaclouds approach. In Proceedings of the 2013 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, Timisoara, Romania, 23–26 September 2013; pp. 417–423.
151. Petcu, D. Portability and Interoperability between clouds: Challenges and case study. In Proceedings of the European Conference on a Service-Based Internet, Poznan, Poland, 26–28 October 2011; pp. 62–74.
152. Liao, C.-S.; Shih, J.-M.; Chang, R.-S. Simplifying MapReduce data processing. *Int. J. Comput. Sci. Eng.* **2013**, *8*, 219–226.
153. Silva, L.A.B.; Costa, C.; Oliveira, J.L. A common API for delivering services over multi-vendor cloud resources. *J. Syst. Softw.* **2013**, *86*, 2309–2317.
154. Sakr, S.; Liu, A.; Batista, D.M.; Alomari, M. A Survey of Large Scale Data Management Approaches in Cloud Environments. *IEEE Commun. Surv. Tutorials* **2011**, *13*, 311–336.
155. Curino, C.; Jones, E.P.; Popa, R.A.; Malviya, N.; Wu, E.; Madden, S.; Balakrishnan, H.; Zeldovich, N. *Relational Cloud: A Database-as-a-Service for the Cloud*; MIT Libraries: Boston, MA, USA, 2011.
156. Lehner, W.; Sattler, K.-U. Database as a service (DBaaS). In Proceedings of the 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010), Long Beach, CA, USA, 1–6 March 2010; pp. 1216–1217.
157. Kiefer, T.; Lehner, W. Private table Database virtualization for dbaas. In Proceedings of the 2011 Fourth IEEE International Conference on Utility and Cloud Computing, Melbourne, Austalia, 5–8 December 2011; pp. 328–329.
158. Zafar, R.; Yafi, E.; Zuhairi, M.F.; Dao, H. Big data: The NoSQL and RDBMS review. In Proceedings of the 2016 International Conference on Information and Communication Technology (ICICTM), Kuala Lumpur, Malaysia, 16–17 May 2016; pp. 120–126.
159. Sahatqija, K.; Ajdari, J.; Zenuni, X.; Raufi, B.; Ismaili, F. Comparison between relational and NOSQL Databases. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 216–221.
160. Alias, N.; Suhari, N.N.; Saipol, H.F.; Dahawi, A.A.; Saidi, M.M.; Hamlan, H.A.; Teh, C.R. Parallel computing of numerical schemes and big data analytic for solving real life applications. *Jurnal Teknologi.* **2016**, *78*. https://doi.org/10.11113/jt.v78.9552.
161. Chang, M.-L.E.; Chua, H.N. SQL and NoSQL Database Comparison. In Proceedings of the Future of Information and Communication Conference, Vienna, Austria, 4–6 December 2018; pp. 294–310.
162. Frizzo-Barker, J.; Chow-White, P.A.; Mozafari, M.; Ha, D. An empirical study of the rise of big data in business scholarship. *Int. J. Inf. Manag.* **2016**, *36*, 403–413.

163. Chickerur, S.; Goudar, A.; Kinnerkar, A. Comparison of relational Database with document-oriented Database (mongodb) for big data applications. In Proceedings of the 2015 8th International Conference on Advanced Software Engineering & Its Applications (ASEA), Jeju Isalnd, Republic of Korea, 25–28 November 2015; pp. 41–47.
164. Kumar, M.S. Comparison of NoSQL Database and Traditional Database-An emphatic analysis. *JOIV Int. J. Informatics Vis.* **2018**, *2*, 51–55.
165. Ansari, H. *Performance Comparison of Two Database Management Systems MySQL* vs. *MongoDB;* Umeå University: Umeå, Sweden, 2018.
166. Khan, W.; Raj, K.; Kumar, T.; Roy, A.M.; Luo, B. Introducing Urdu Digits Dataset with Demonstration of an Efficient and Robust Noisy Decoder-Based Pseudo Example Generator. *Symmetry* **2022**, *14*, 1976.
167. Roy, A.M.; Bhaduri, J. A Deep Learning Enabled Multi-Class Plant Disease Detection Model Based on Computer Vision. *AI* **2021**, *2*, 413–428. https://doi.org/10.3390/ai2030026.
168. Roy, A.M. An efficient multi-scale CNN model with intrinsic feature integration for motor imagery EEG subject classification in brain-machine interfaces. *Biomed. Signal Process. Control.* **2022**, *74*, 103496.
169. Singh, A.; Raj, K.; Kumar, T.; Verma, S.; Roy, A.M. Deep Learning-Based Cost-Effective and Responsive Robot for Autism Treatment. *Drones* **2023**, *7*, 81.
170. Dhasade, S.D. Nosql Database. Available online: https://www.irjmets.com/uploadedfiles/paper/issue_10_october_2022/30598/final/fin_irjmets1665589950.pdf (accessed on 11 November 2022).