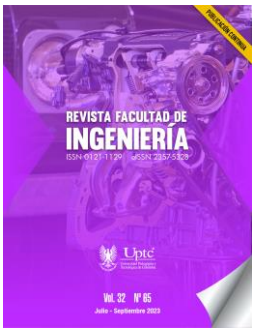
**Revista Facultad de Ingeniería**
Journal Homepage: https://revistas.uptc.edu.co/index.php/ingenieria




# NoSQL Database Modeling and Management: A Systematic Literature Review

Raúl Aguilar Vera[1]
Andrés Naal Jácome[2]
Julio Díaz Mendoza[3]
Omar Gómez Gómez[4]

**Received:** May 25, 2023 **Accepted:** September 21, 2023 **Published:** September 30, 2023

**Citation:** R. Aguilar Vera, A. Naal Jácome, J. Díaz Mendoza, O. Gómez Gómez, "NoSQL Database Modeling and Management: A Systematic Literature Review," *Revista Facultad de Ingeniería*, vol. 32, no. 65, e16519, 2023. https://doi.org/10.19053/01211129.v32.n65.2023.16519

## Abstract

The NoSQL databases that emerged this century were created to solve the limitations of relational database systems due to the different types of data that have appeared for information processing. In this paper, we present the results of a

[1] Ph. D. Universidad Autónoma de Yucatán (Mérida, México). avera@correo.uady,mx. ORCID: 0000-0002-1711-7016
[2] Universidad Autónoma de Yucatán (Mérida, México). a17016365@alumnos.uady.mx
[3] M. Sc. Universidad Autónoma de Yucatán (Mérida, México). julio.diaz@correo.uady.mx. ORCID: 0000-0002-3005-3432
[4] Ph. D. Escuela Superior Politécnica de Chimborazo (Riobamba, Ecuador). ogomez@espoch.edu.ec. ORCID: 0000-0002-2951-3833

secondary study carried out to find and synthesize the research made up to now on modeling processes, characteristics of the used types of data, and management tools for NoSQL Databases. Currently, four types are recognized and classified according to the data model they use: key-value, document-oriented, column-based, and graph-based. With this study, it was possible to identify that the most frequently type of NoSQL database model is that of documents because it offers greater flexibility and versatility compared to the other three models. Although it offers more complex search methods, in terms of data, column and document schemas are the ones that usually describe their characteristics. It was also possible to observe a trend in the use of the column-oriented model and the document-oriented model in the management tools, and, although they all comply with the basic functionalities, the differences lie in the way in which the information is stored and the way they can be accessed.

**Keywords:** database modeling; NoSQL; software engineering; systematic literature review.

## Modelado y gestión de bases de datos NoSQL: Revisión sistemática

### Resumen

Las bases de datos NoSQL que surgieron este siglo fueron creadas para resolver las limitaciones de los sistemas de bases de datos relacionales debido a los diferentes tipos de datos que han aparecido para el procesamiento de la información. En este artículo, presentamos los resultados de un estudio secundario realizado con el fin de encontrar y sintetizar la investigación realizada hasta ahora sobre procesos de modelado, características de los tipos de datos utilizados, y herramientas de gestión para bases de datos NoSQL. Actualmente, se reconocen y clasifican cuatro tipos según el modelo de datos que utilizan: clave-valor, orientada a documentos, basada en columnas y basada en gráficos. Con este estudio se identificó que el tipo de modelo de base de datos NoSQL más frecuente es el de documentos porque ofrece una mayor flexibilidad y versatilidad en comparación con los otros tres modelos. Aunque ofrecen métodos de búsqueda más complejos, en términos de datos, los esquemas de columnas y documentos son los que suelen

describir sus características. También se pudo observar una tendencia en el uso del modelo orientado a columnas y el modelo orientado a documentos en las herramientas de gestión, y, aunque todas cumplen con las funcionalidades básicas, las diferencias radican en la forma en que se almacena la información y la forma en que se puede acceder a ellas.

**Palabras clave:** ingeniería de software; modelado de bases de datos; NoSQL; revisión sistemática de la literatura.

## Modelagem e gerenciamento de bancos de dados NoSQL: Revisão sistemática

### Resumo

Os bancos de dados NoSQL que surgiram neste século foram criados para solucionar as limitações dos sistemas de bancos de dados relacionais devido aos diferentes tipos de dados que surgiram para o processamento de informações. Neste artigo apresentamos os resultados de um estudo secundário realizado com o objetivo de encontrar e sintetizar as pesquisas realizadas até o momento sobre modelagem de processos, características dos tipos de dados utilizados e ferramentas de gerenciamento para bancos de dados NoSQL. Atualmente, quatro tipos são reconhecidos e classificados de acordo com o modelo de dados que utilizam: valor-chave, orientado a documentos, baseado em colunas e baseado em gráficos. Este estudo identificou que o tipo mais comum de modelo de banco de dados NoSQL é o documento, pois oferece maior flexibilidade e versatilidade em comparação aos outros três modelos. Embora ofereçam métodos de pesquisa mais complexos, em termos de dados, esquemas de colunas e documentos são o que geralmente descrevem suas características. Também se pode observar uma tendência na utilização do modelo orientado a colunas e do modelo orientado a documentos nas ferramentas de gestão e, embora todos cumpram as funcionalidades básicas, as diferenças residem na forma como a informação é armazenada e no como eles podem ser acessados.

**Palavras-chave:** engenharia de software; modelagem de banco de dados; NoSQL; revisão sistemática da literatura.

## I. INTRODUCTION

For years, attempts have been made to solve the deficiencies presented by the file system through various models that allow better data management [1]. These models represent the thinking about the concept of database: What should they do? What kind of structures to use? and, What technology should be used to implement them? In the period from 1960 to 1970, information systems were developed using the file system model, which consisted of a set of managed records. During the 1960s, the hierarchical database model was developed to manage large amounts of data associated with manufacturing projects that were very complex, such as the Apollo rocket in 1969. Also, the network model was created to represent more complex data relationships more effectively than the hierarchical model, with the goal of improving database performance and establishing a standard. With the growth of information needs, more sophisticated databases were required, so the hierarchical model and the network model became a problem, since a change in the database structure could cause a strong impact on the applications that obtained the data. The disadvantages of these models were largely overcome by the relational model proposed by Codd [2], which has prevailed until now.

The relational model is based on a mathematical concept known as relation, whose physical representation can be perceived as a table. The integrity of the information that it allows to manage is fundamentally based on the concepts of primary key and foreign key.

However, with the new century, new applications arose that created the need for other types of structures for a database that was capable of handling large volumes of data; what we know today as NoSQL Databases (DB) comes from there. Google with Big Table and Amazon with Dynamo were the first two influences for the emergence of this new type of Databases; in fact, the term “NoSQL” that we know today was proposed in a meeting held in 2009, organized by Johan Oskarsson [3].

There is no global definition recognized by the scientific community of what NoSQL is, however, it is possible to identify some of its characteristics. The highlight is that they do not use SQL as a query language, they are perceived as open-source projects, most of them are based on transactions known as ACID (atomicity,

consistency, isolation, and durability). According to [4] they are used to guarantee the integrity of the data and ensure its consistency in any situation during management.

Another key feature of NoSQL systems is their “nothing shared" horizontal scalability, that is, to replicate and partition data across many servers, allowing for large numbers of both read and write operations [5]. The above feature takes advantage of low-cost processors that have their own RAM memory and storage; thus, the database can be stored on multiple processors and maintain high performance [6].

The increase of systems compatible with ACID qualities has become a problem since conflicts have appeared between high availability aspects in distributed systems that cannot be solved entirely. This is known as the CAP theorem [4], which states that only a maximum of two of the following characteristics can be met: consistency, availability, and tolerance to partitions, systematic requirements for this type of environment [7]. Currently, four main types of NoSQL databases are recognized, classified according to the data model they use: (1) key-value, (2) document-oriented, (3) column-based, and (4) graph-based [8].

NoSQL DBs are not intended to replace relational DBs, rather, they provide certain advantages such as many models to choose from, they are easier to scale, do not require administrators to use them, some are programmed to manage hardware failures, are faster, more efficient, and more flexible. Although these databases have been evolving fast, they still have disadvantages: they are somewhat immature, and there is still no standardized language or interface for their use [9].

Some previous secondary studies have addressed design methods in the new era of databases [10], i.e., the existing challenges in NoSQL data storage in a distributed way [11], the new perspectives of NoSQL database design [12], as well as the data models for the Big Data problem [13]. However, no studies were found that synthesize, according to the type of database, the modeling procedures of these databases, the characteristics that said data must meet, and the characteristics of the management tools of those systems.

## II. METHODOLOGY

The Systematic Literature Review (SLR) is a methodology that makes it possible to identify, evaluate, and interpret the research available in the literature that is relevant to a research question, thematic area, or phenomenon of interest. According to Genero et al. [14] individual studies that contribute to a systematic review are called primary studies, so a study based on a systematic review is considered secondary.

To carry out this SLR, the guide proposed by Kitchenham in [15] was used, and to increase the number of selected primary studies (SPS) that could be relevant, the authors decided to use the forward snowball method [16] in a complementary way to identify new articles in the first set of SPS.

The purpose of this study is to collect, analyze, extract, and synthesize the research carried out so far on NoSQL database modeling. The intention is to describe the main processes that DB designers follow to build and obtain a non-relational schema, the characteristics that the data must meet, and the tools used to manage them.

### *A. Research Questions*

Derived from this research opportunity, three research questions were formulated that will serve to conduct the present study:

***RQ-1*** *What are NoSQL database modeling procedures, based on their type?*

***RQ-2*** *What characteristics must the data used in NoSQL databases meet, depending on its type?*

***RQ-3.*** *What characteristics should NoSQL database management tools have to implement data models, based on their type?*

### *B. Selection of Sources and Search Strategies*

Once the questions were established, the characteristics of various databases and repositories were analyzed, and the following four databases were selected: ACM, IEEE, Science Direct, and Springer Link. They will serve as a source of article consultation, since they are the most used for this type of study, as reported in previous works.

### *C. Creating the Search String*

For the automated search process within the databases, the second task of the strategy consisted of generating a search string using a set of keywords related to the subject and linked by logical connectors. It will be used to carry out the searches: *(NoSQL OR Non-relational) AND (Design OR Model) AND (Process OR Technique OR Approach OR Practice OR Procedure OR Method).* However, this string must be configured based on the queried database.

### *D. Inclusion and Exclusion Criteria*

The inclusion and exclusion criteria allow us to shorten the number of results to focus on the primary studies of our interest.

Inclusion criteria:

- Empirical research article published in a specialized journal in English
- Studies published from 2014 to 2021
- Subject of the study related to the modeling of NoSQL databases

Exclusion criteria:

- Secondary and tertiary studies
- Duplicate studies
- Articles whose content is not accessible
- “Forward” Snowball Method

It was agreed to use the forward snowball method, which consists of identifying the articles that have cited any of the studies belonging to the base set, and from them, examining and selecting those that are important for the investigation. In this study, the ones that meet the same inclusion and exclusion criteria applied to the first set. To identify new studies, Google Scholar was used because it allows us to find, with some relevance, the articles that have cited a particular selected study.

To select the relevant primary studies, the generic string was first adapted to the search engine characteristics in each DB. Table 1 shows the string used for each.

**Table 1**. Strings Configured by Database.

| Database | Configured String |
|---|---|
| IEEE Xplore | (("Abstract": NoSQL OR "Abstract": Non-relational) AND ("Abstract": Model OR "Abstract": Design) AND ("Abstract": Process OR "Abstract": Approach OR "Abstract": Technique OR "Abstract": Procedure OR "Abstract": Method)) |
| ACM-DL | Abstract: ((NoSQL OR “Non-relational”) AND (Design OR Model*) AND (Techni* OR Process* OR Practic* OR Approach OR Procedure OR Metho*)) |
| ScienceDirect | (NoSQL OR “Non-relational”) AND (Design OR Model) AND (Technique OR Process OR Method OR Approach OR Practice) |
| SpringerLink | (NoSQL OR 'Non-relational') AND (Design* OR Model*) AND (Techni* OR Process* OR Practic* OR Approach OR Procedure OR Metho*) |

After completing the search process with the strings and filters in each database and obtaining the list of primary studies, the process of locating complete articles from various sources began. Table II shows the number of studies selected by database, in each of the three stages of the process.

**Table 2**. Results of the Search.

| Database | Results | Inclusion criteria | Exclusion criteria |
|---|---|---|---|
| IEEE Xplore | 379 | 12 | 12 |
| ACM-DL | 160 | 0 | 0 |
| ScienceDirect | 2,197 | 15 | 15 |
| SpringerLink | 10,037 | 8 | 8 |

With the aim of increasing the number of selected primary studies, the forward snowball method was used to incorporate articles that could be relevant for the systematic review. The initial group consisted of thirty-five primary studies drawn from the four databases selected as the initial set. Twenty-two new studies were added to the systematic review. In the same way, the number of citations was extracted for later use in the Quality Assessment.

### *E. Primary Studies Quality Assessment*

Quality assessment is an important process in Systematic Literature Reviews since it allows us to guarantee the validity and reliability of a study's findings. Unfortunately, given the diversity of types of primary study, there is no specific or standardized instrument to carry it out, which is why the instrument proposed in [17] was adapted, considering the following criteria:

- Report quality: It allows to evaluate the clarity of the objectives, the context, and the justification of a study.
- Credibility: Evaluates that the study methods guarantee the validity and significance of the findings. Additionally, the evaluation must ensure that the findings and/or conclusions are effective [18].
- Relevance: Focuses on evaluating the importance and value of research for the software industry and the scientific community. According to [19], there are two types of relevance: Academic, which is shown through the ability to publish articles and quotes from other researchers; and Industrial, which refers to the value of the assessment for professionals looking to adopt technologies.
- Rigor: The characteristics of the research methods are considered to declare the validity of the data collection tools and analysis methods, therefore, the reliability of the findings.

To make a quantitative assessment of the first three criteria, three levels or degrees of compliance were considered (0, 0.5, 1), for which a rubric was designed. In the case of rigor, it was considered appropriate to verify compliance using a two-level checklist (0, 1). Each SPS will obtain a quality indicator with values between 0 and 10. With the purpose of classifying the SPS, three categories were defined: first, High Quality, if i >= 8, all sections of the article will be carefully reviewed, and the study findings will be included in the synthesis; second, Medium Quality, if 5.5 <= i <= 7.5, all sections will be reviewed; finally, Moderate Quality, if i <= 5, only the main sections will be reviewed.

For the Quality Assessment of each of the 57 primary studies, the instrument described in the previous section was used. The distribution was the following: 20 studies were of High Quality, 35 of Medium Quality, and only 2 were classified as Moderate Quality.

## III. Results

This section presents the answers to the research questions in this Systematic Literature Review.

### *A. RQ-1. What are NoSQL Database Modeling Procedures, Based on Their Type?*

Regarding the type of key-value database, eight studies addressed modeling processes; four of them introduced a process based on a set of rules. In [20], an approach is proposed in which an engine is used to convert a relational model to a key-value model, mapping the concepts of each model to generate transformation rules that said engine uses to carry out the process, while the three remaining studies present formal definitions to build the models. In [21], the authors developed a set of rules to map the key-value model to a unified model. Moreover, in [22], an extension of the JSON schema directed to the representation of geographic data is presented. This consists of mapping the key-value database type to a JSON format, using a set of rules presented in the formal grammar that governs the schema definitions of spatial data types. Finally, in [23], a unifying data model that combines the different concepts from existing data sources is proposed. They presented a formal definition of the unifying model using these concepts and introduced a set of correspondence rules to express the possible unions between sets of entities.

Furthermore, three studies presented other ways of representing a data set with a key-value model. In [24], the authors explain that to perform this representation a key-value pair can be used for each entry in the data set. The key is made up of two parts, the major key, made up of the collection name, and the block key. The second part is made up of the minor key, thus being an appropriate encoding of the input key. Additionally, [25] presents two solutions for a database to manage bi-temporal properties in the data. One of these is to add two attributes to each key-value pair, one for the initial valid time and one for the final valid time, while the other solution is to split the key into two parts to form a composite key. This would be made up of one part that serves to identify the row, and the other to represent the time interval. Moreover, in [26], a scheme made up of two persistent data stores is proposed, in which the key-value pairs are classified into two different groups so that each pair is directed to one of the two stores, depending on the size of its data. Just as importantly, in [27], the authors propose a method to select the most appropriate

database model for polyglot persistence applications based on user requirements related to data management.

Regarding column-oriented databases, twenty-one studies addressed this kind of model. In six of these studies, models designed at the discretion of the authors were proposed in accordance with the objective they had. In [28], the authors proposed a model for the storage and management of geospatial data. In [24], an independent model of the target system is proposed. Along with it, in [29], a logical change (delta) data model is proposed. In [25], a solution is presented to include temporal data in the column-oriented model through the manipulation of row keys. In [30], the authors proposed a model based on the Z-curve for spatial vector data. Finally, in [31], the authors designed a database to store seismic information and developed a Kafka production-consumption model to process the data in real-time.

In relation to the rule-based approach, the authors of [21] created a representation of column-oriented databases as a unified model through a mapping process. On their own, [32] present the development of a column-oriented data store through a set of definitions. In [22], an extension of JSON to represent geographic data is presented, which consists of mapping the column-oriented database type to said format. In [33], the authors extracted a set of rules for the transformation of relational databases to column-oriented databases, through the knowledge obtained by running two experiments. Then, in [34], the process of creating an XML data model is described through a series of definitions.

In addition, four studies presented design processes that consist of following a series of defined steps. In [35], a control cycle is developed for tuning schemes automatically based on the workload. In [36], the authors develop a NoSQL schema recommendation system given a conceptual model and application workload. Additionally, two studies focused on a process for various NoSQL technologies, including the column-oriented model. Furthermore, in [37], a model-based process is proposed, which begins with the creation of a conceptual model; later, it is translated into a logical NoSQL model by applying a set of transformation rules. The obtained model is then used to generate a deployment script for the selected NoSQL technology. In [38], a design method is proposed for systems that require multiple

data models, through a process of three main steps, consisting of conceptual design, logical design, and physical design. Another approach consists of creating algorithms that were implemented by two studies. In the first [39], the authors describe a smart grid data generator with a machine learning algorithm. The second [40] presents a model for data related to the health sector, which consists of a set of algorithms developed to store and process said data. Three studies provided methods for modeling the data. In [41], techniques and guides are provided for migrating from a relational database to a column-oriented model. In [42], a method of data partitioning and selective replication for social media is proposed; it uses information from previous workloads to predict future query patterns. Finally, in [43], the authors present a model for radio spectrum data provided by monitoring sensors. It was created following the simple design pattern of using the device identifier as a partition key and a timestamp as a grouping key.

Regarding the document schema, twenty-nine studies addressed modeling processes. One way of executing such a process is through a set of established steps, which was used in eight studies. Authors [44], define a model-driven solution, which consists of a two-step model transformation chain, in which an intermediate model is created to facilitate code generation from inferred schemes. In addition, [45] presents an approach to the design process, which consists of three main steps. The first is generated with a set of schemas from a UML data model given by a user. The second evaluates a set of metrics on the created schemes. The third consists of analyzing user priorities and system requirements to select the best option.

In [46], a relational database conversion approach is proposed to define, evaluate, and compare schema alternatives in relation to access patterns, relying on query metrics and scores to choose the best option. In [47], the authors improved the coding of CityJSON, a format with city models, through its decomposition, resulting in a scheme to store the information in the database. For their part, in [48], the authors present an approach using reverse engineering to introduce security into existing NoSQL schemes. By automatically analyzing the structure of the database and the data using a domain ontology, the problems found are listed, and based on them, it proposes a model with improved security. In [37], the authors develop a

model-based design process, which starts with the creation of a conceptual model. It is then translated into a logical NoSQL model through the application of a set of transformation rules, and finally, the resulting model is used to generate a deployment script. In [49], an ontology-driven metamodel is proposed to conceptualize data representations, which provides an abstraction based on semantically enriched formal vocabularies for NoSQL and SQL databases. Subsequently, said conceptualization is mapped to the database. In [50], a design method for systems that require multiple data models is presented, based on the traditional database design process. It consists of the conceptual design, and the logical design, particularly where the conceptual model is converted to the desired NoSQL model, and the physical design.

Conversely, seven studies used algorithms for the schema design process. In [51], the authors propose a model that is responsible for suggesting schemes at the beginning of development based on user requirements. The architecture of this model is made up of four phases, and an algorithm is used to generate the schemes. In [52], a strategy to discover inheritance relationships between entities using an algorithm is presented; it starts from an already inferred scheme, resulting in a final model that includes the detected subtypes. The authors of [53] created a data partitioning method based on dimensionality reduction, using a schema fragmentation algorithm. An algorithm that transforms data relationally to a NoSQL model is proposed, which starts with the conversion of the structure to a graph. Subsequently, adjustments are made to the graph to improve the design, and finally the document-oriented schema is generated based on the modified graph [54]. An intelligent network data generator with a machine learning algorithm is proposed, and one of its implementations was carried out in a document-oriented database [39]. Authors [55], present an approach to model time series data. Indicating that through an algorithm, the model can evolve when there are new data integrations, and other [56] proposes a model that, through algorithms, generates schema proposals using the relational database of an existing system as a starting point.

Likewise, in five studies, the modeling was carried out according to criteria. In [57], the authors propose a methodology to store fuzzy geospatial data by adding a

control, which consists of an automatic revision system for the consistent storage of information, using a JSON-based schema proposed by the authors. In [24], the authors present an independent model of the target system, explaining that their implementation can be based on a MongoDB collection for each collection of blocks, and a single main document for each block. In [58], a system to store geospatial data is proposed, whose database is generated according to the characteristics of the data set and the data attributes, based on a predetermined design.

The authors of [59] present a method for normalizing and structuring subway data and modeling it in a NoSQL schema, which was done at the discretion of the authors, based on the design direction of the data. Then, in [60], a system is developed using NoSQL to train, validate and implement potato late blight risk modeling using a grid data format, whose model was created considering the characteristics of the data.

Four studies used formal definitions to carry out the process. The authors in [61] present a formal model to assist in database design considering query costs and memory usage. In [21], a set of formal rules is presented to map document databases to a unified model, which is a schema formed by a collection of types, either entities or relationships. In [22], the authors created an extension to the JSON schema for the representation of geographic data. It consists of converting a document-oriented model to a JSON format through a set of rules presented in formal grammar. Lastly, in [23], a unifying data model is proposed for the main existing models, including the document model, with a formal definition using concepts from existing data stores. They also introduced a set of correspondence rules to express possible relationships between entities. Another way is to follow a method to support the design, as shown in three studies. In [62], a storage method to improve the management of geochemical data is proposed; it consists of two levels, JSONification, where the data is converted using an existing library, and storage in the cloud. In [51], the authors developed a model whose architecture is made up of seven layers, which generates the scheme according to the user and system requirements, with the aim of managing data asymmetry. In [27], a method to select the most appropriate database model for polyglot persistence applications was developed, considering user requirements related to data handling.

Another way consisted of the use of a set of rules. In [63] the authors created a NoSQL store for maritime data, whose model was created from a UML diagram and using simple transformation rules. Authors [43], present a model for the radioelectric spectrum data provided by monitoring sensors, created by applying a grouping pattern to include multiple records from the same sensor in a single document.

Finally, seven studies addressed the modeling of graph databases. Three of these studies presented formal definitions to build the models. In [64], the authors introduced a metamodel that allows to impose structural constraints on the data in a flexible way. An extension of the JSON schema is used to represent geographic data by converting the graph model to the format. In [65], a formal design consisting of a set of transformation rules for three types of databases is presented. The goal is to manage the metadata of a polyglot system by converting them to a graph [22].

Likewise, two studies used a set of rules to perform the modeling. In [21], the authors created a representation of the graph databases to a unified model. Then, in [66], the authors present the redesign of a relational database to a hybrid model through the creation of an ontology based on a series of steps. They used the database model, loaded in an RDF triple store, which is a type of graph store. In addition, the authors in [67] developed an algorithm to solve the graph partition problem with cascade recognition. In [68], a model based on graphs is proposed to manage data generated by networks of multimedia sensors, which was built based on the topology of these networks.

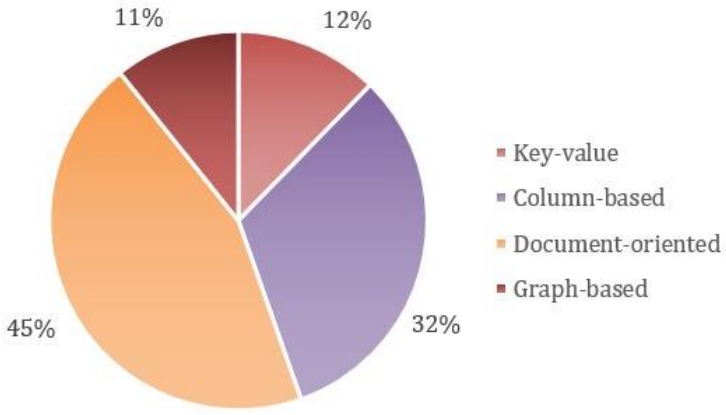



**Fig. 1**. Classification of the SPS that address modeling processes according to their type.

As can be seen in Figure 1, the forty-eight studies responding to the QI-1 dealt with the four models on a total of sixty-five occasions. The document schema is the most used by the authors who approached NoSQL database modeling processes, it was used 29 times. We consider that this is due to the flexibility it offers compared to the other models; instead, the key-value and graph schemes are the least used, with 8 and 7 times, respectively.

### *B. RQ-2. What Characteristics must the Data Used in NoSQL Databases Meet, Depending on its Type?*

Regarding the key-value model, only one study described the characteristics of the data, the authors worked with geographic data, and categorized a set of definitions into five classes, each with different characteristics [22]:

- *Direct Position* is an array of two or more decimal numbers representing longitude, latitude, and optionally, altitude,
- *Primitive Geometry* must have two mandatory elements, being the type and the coordinates,
- *Geometry Collection* contains an array of geometries,
- *Feature* must have a geometric element that defines the spatial component, whose value must be one of the spatial data types or null; likewise, it must contain properties that define the attributes.
- *Feature Collection* is defined as an array of features, keeping the Features as items.

Afterward, seven studies, which cover the column-oriented scheme, mention the characteristics of the data. In [69], when handling photovoltaic data, the authors describe that a triple consisting of a row key, a column key and the input value is stored. Because they deal with time series data, they find it intuitive to use timestamps as a unique identification key for each value. They also mention that using the location of the test sites as a column family name facilitates the search of the data and reduces the need to use aggregations. Furthermore, each time series data file is separated by time, multiple nodes processing on the same file in parallel reduce processing time. In [70], the authors, who deal with agricultural data, mention

the characteristics of seasonality, region, comprehensiveness, opportunity, multiple hierarchy, and innovation. Additionally, they have a diversified source and there are different types, so there is not much similarity when dealing with other data sets. In [30], the authors focused on spatial polygonal data, working with geometric objects. In [22], geographic type data is addressed, whose characteristics previously described in the key-value type also apply to the model in question. In [31], the authors worked with seismic wave data, whose main characteristic is that they are obtained and processed in real time. Besides, large amounts of information are constantly generated. In [40], the authors deal with health information and describe the considerations they have regarding the data. They assume that all health centers use the same type of identifier for patients. Furthermore, before storing the captured information, it is necessary to ensure the transmission of the data and check its integrity. Finally, in [42], a method of partitioning data from social networks is proposed, which is characterized by having information on users and the interactions between them.

Regarding the document model, five studies addressed data characteristics. In [62], the authors deal with geochemical data, which amounts increase rapidly with the improvement of test methods and precision. These are in a format called a shape file, which consists of data vectors. In [57], the authors, working with fuzzy geospatial data, describe that these are naturally imprecise due to the distribution of city centers, whose location is difficult to identify due to the constant growth of these cities; also, due to the complexity of determining where a forest starts. They also indicate that the semantic characteristics must be validated so that the JSON format, the fuzzy structure, and the semantic restrictions must be respected to store the data. In [71], a structure that adapts to medical image packages is developed; it can contain one or several of them. In [22], the authors conducted their study on geographic data; the characteristics described in the key-value model also apply to this scheme. In [59], the study focused on subway data, detecting that due to the duplication of information between the files, it is difficult to maintain consistency. It is also difficult to manage them when they are separated into many files, in addition, the data is not normalized, and the amount of information increases exponentially.

Regarding the graph model, one study addressed the characteristics of the data. [22] was directed to geographic data, whose characteristics described in the key-value model also apply to this scheme.

Figure 2 indicates that, overall, few studies described the characteristics that the data have or should meet. Eleven studies covering the QI-2 deal with the subject fourteen times. The column model is the most used —7 times. On the contrary, the number of times that the studies identify characteristics of the data using the key- and graphs value model turns out to be the least, with only one mention each.

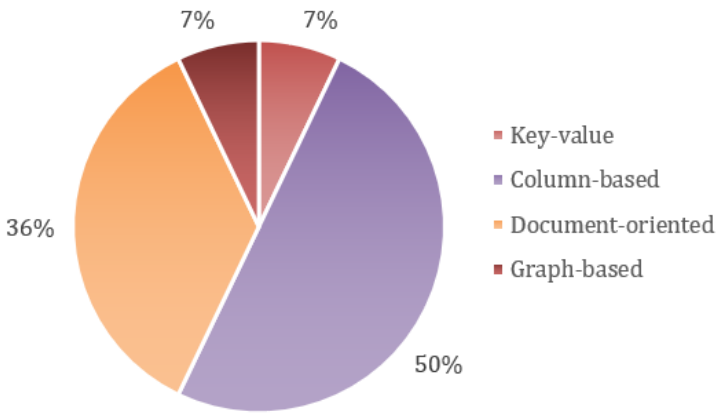



**Fig. 2**. Classification of the SPS that address data characteristics according to their type.

### *C. RQ-3. What Characteristics should NoSQL Database Management Tools have to Implement Data Models, Based on their Type?*

Regarding the type of key-value model, two studies mentioned the characteristics of the tool used. In [25], they comment that Redis allows managing temporary data because it is compatible with several types of structure data such as strings associated with a key, lists, sets, ordered sets, and hashes. This demonstrates its flexibility. In addition, it has a set of functions that help fulfill this purpose because it does not support them natively. In [26], the authors propose their own store, whose architecture consists of two stores: one for light data and one for heavy data. These are classified into one, depending on their size, in addition to including a functionality that converts the data into compressed hashes to reduce search latency.

Regarding the column-oriented model, fourteen studies address the characteristics of management tools. Six of them implemented their proposals. In [35], they highlight their fault tolerance, linear scalability, high availability, and ability to simulate a joint operation through denormalization. [36] Comment that it only supports put, get, and delete operations on column families, in addition, it provides a limited form of secondary indexing, which allows selecting records in other ways. In [39], authors tested different tools with data from smart networks. Cassandra turned out to be the best in read and write operations, particularly for its key structure and its ordering of data according to time, thus resulting in a shorter response time and low latency [42]. In [25], they presented a solution to implement temporal data and also point out that although Cassandra cannot be classified as a temporary database and does not have an interval data type, it does provide capabilities to handle them, as it allows time to be defined by adding a column with the valid time interval. Lastly [43], ran into two problems when implementing their radio spectrum data model, since sensors collect large amounts of information, and the tool does not allow too large partitions. Therefore, they added a column to the model although the queries became more complex, since the tool does not allow range conditions on the partition key. Likewise, six studies used Hbase [36]. Highlights that it only supports put, get, and delete operations on column families, and that it is possible to obtain information for a range of partition keys, since the records are ordered according to it. In [34], the authors decided to use it due to its ability to store large tables because it is distributed and scalable. In addition, it supports data compression, operations in memory, Bloom filters based on each column, which allows complex queries with XML, and uses HDFS. This allows better performance of the queries in any system. Additionally, focused on seismic data [31]; they highlight reliable and high performance. This file system uses HDFS to expand the system linearly by adding nodes. It provides a large-scale concurrency capability for data processing and supports the storage of time series in the form of records, which can have multiple versions. Its version number is assigned automatically, providing operability and traceability for its modification. In [30], authors noted that when handling tree storage schemas, it is necessary to visit the data table twice, since it does not use foreign

keys. In addition, they indicate that they resorted to the redundancy policy for their design, due to its excellent scalability to store large amounts of redundant data. In [33], they do not use relationships in their design, so a joint operation is not possible. They also identified that, due to their distributed nature, column families are not always stored on the same server, which can increase access time. In [69], they highlight that its characteristics are appropriate for the analysis of scalable photovoltaic data. In addition, they use its key-value pairs scheme, which allows time series, I-V curve, and spectral data to be inserted into the same database. This is useful for analyzing multiple types of data and even allows the addition of new power plants with new variables to existing data tables. Two studios proposed their own stores. [72] Equipped it with a resource reservation functionality to prevent performance issues in a multi-tenant environment, specifically with cache and disk space reservation. The one developed in [32] implements a key-cube data model, which supports high-dimensional data, and partitions into regions, improving query performance by reducing search space. Finally, the author proposes a way to store geospatial data using the NoSQL model due to its high input/output capacity and high search speed [28].

Twelve studies have identified characteristics of the tools with the document model, eleven of these focused on MongoDB for various reasons. [62] Point out that it can process large amounts of data efficiently. It supports embedded document objects and arrays, has an automatic chunking mechanism, and has remarkable performance for storing structured data. In [61], authors highlight that it can store data in a binary JSON format, supports queries like field, range regular expressions, aggregation, and secondary indexes. In addition, its architecture allows the user to choose which storage engine to use [46]. Used it since it includes a rich and expressive query language, as well as a set of operators that allowed them to standardize the operations required to retrieve documents. Selected it due to its ability to handle large amounts of information, besides, its document format adapts to the data structure of the domain in which it was focused [59]. In [63], they implemented it with the tool in question because it is the fastest growing database. It allows the division of data into collections, provides a rich document structure,

allows dynamic queries, has an aggregation engine, and integrates a manipulation engine for geospatial data. Highlight its functionalities to store files and easily retrieve them [73]. In [43], authors highlight their ability to use embedded documents; however, they mention that they had to adapt their model to the limit size of the documents, which is 16MB, a feature that is also pointed out in [60]. Additionally, in [53], they point to the lack of support it has in spatial analysis for proximity and containment search. However, they selected it because of its support for secondary indexes and native support for clustering and aggregation, even so, they highlight that it has many limitations for handling geospatial data. Additionally, [39] conducted experiments with smart grid data and found that MongoDB had the worst performance for managing this data due to its high latency. However, compared to MySQL, comment that MongoDB's Document Object Mappers provide more capabilities to facilitate the development of database applications, for example, index specifications, validators, and reference management [44]. In contrast, in [74], the authors used a different tool, Couchbase Server, which is designed with a distributed architecture for performance, scalability, and availability, and allows applications to be developed quickly due to the flexibility it provides thanks to its support for JSON. Finally, regarding the graph scheme, only one study mentions the characteristics of the tools. In [68], the authors implemented their model in Orient DB due to its qualities: ACID transactions, unique constraints, full text support, spatial support, record level security, Java Hooks support, user and role security, custom data types, indexes on multiple properties, different schema modes, sharing, server-side functions, unrestricted embedding, and streams. However, the most important ones were multi-master replication, SQL, and elastic scalability.

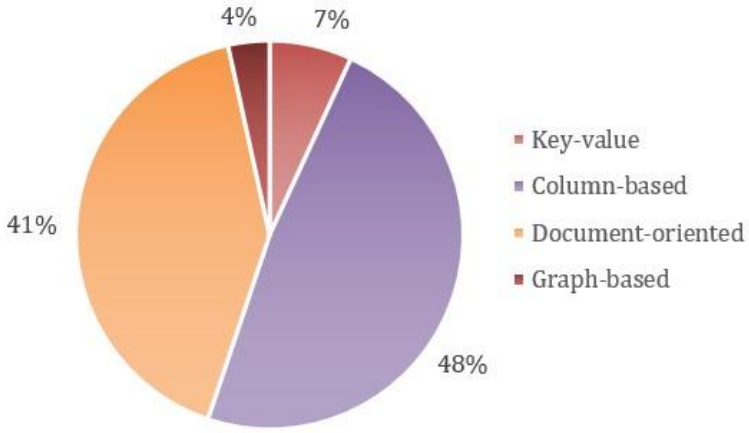



**Fig. 3**. Classification of the SPS that address characteristics of database management tools according to their type.

As can be seen in Figure 3, twenty-six studies related to QI-3 identify characteristics that management tools should have by twenty-nine times. The most frequently used are the column-oriented model and the document model, with 14 and 12 studies, respectively. In contrast, the studies that use key-value models, graphs and address these characteristics turned out to be the minority, 2 and 1 studies, respectively.

## IV. Conclusions

By analyzing the SPS through this SLR, it was possible to identify that the document model is the NoSQL model most taken into consideration to propose or perform database modeling processes. From our perspective, this is because it offers greater flexibility and versatility compared to key-value, column, and graph models. In addition, it offers more complex search methods, making it more relevant in the industry.

It was possible to identify that the most used modeling process with the key-value scheme is considering a set of specific rules that instruct the database designer to build the structure that will store the data. The second way to model the scheme is through a particular proposal to represent the data sets, depending on the needs or objectives. Likewise, regarding column-oriented modeling, most of the studies focused on a design of the authors' preference. According to the objective, while the second most used process is based on abiding by a set of rules to build the schema,

other approaches were also used, like following a series of steps or using algorithms, although to a lesser extent. Moreover, the creation of the design through a set of steps turns out to be the most used process with the document outline, while the second most frequent way is using algorithms. Regarding the graph model, the most widely used technique is the application of formal definitions, while following a set of rules is the second most applied.

Regarding the studies that explain or describe the characteristics of the NoSQL database management tools, a tendency towards the column-oriented model and the document-oriented model could be observed. The choice is based mainly on the type of project to develop. Likewise, although they all comply with the basic functionalities of scalability, search speed, flexibility, and management of large amounts of data, their differences lie in the way in which the information is stored and the way data is accessed.

As can be seen in the results of the review, key-value and graph models are the least reported in the literature, since they have the lowest percentage of use with respect to the research questions posed. Therefore, as a follow-up to this study, it would be interesting to conduct a primary study to determine the reasons that influence database designers to choose the other schemes. Likewise, it is proposed to develop new modeling processes for the schemes, seeking a greater variety of ways to implement them in a standardized and simple way for the developer. Finally, it is recommended to investigate different areas or specific domains in which both models could be applied more frequently, so that they improve data management with respect to the document and column models, which were the most used by the authors.

## AUTHORS' CONTRIBUTION

**Raúl Aguilar Vera:** conceptualization, investigation, supervision, writing-review & editing.

**Andrés Naal Jácome:** investigation, formal analysis, visualization, writing-original draft.

**Julio Díaz Mendoza:** investigation, supervision, validation, writing-review & editing.

**Omar Gómez Gómez:** investigation, validation, resources, writing-review & editing.

## REFERENCES


[1] C. Coronel, S. Morris, P. Rob, *Base de datos: diseño, implementación y administración*, Cengage Learning Editores, 2011.

[2] E. Codd, "A Relational Model of Data for Large Shared Data Banks", *Communications of the ACM*, vol. 13, no. 6, pp. 377-387, 1970. https://doi.org/10.1145/357980.358007

[3] P.J. Sadalage, M. Fowler, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*, Addison-Wesley Professional, 2012.

[4] P. Neubauer, *NOSQL and Neo4j*. https://www.scitepress.org/Papers/2017/63560/63560.pdf

[5] R. Cattell, "Scalable SQL and NoSQL data stores," *ACM Sigmod Record*, vol. 39, no. 4, pp. 12-27, 2011. https://doi.org/10.1145/1978915.1978919

[6] D. McCreary, A. Kelly, *Making sense of NoSQL: A guide for managers and the rest of us*, Manning, 2013.

[7] J. Browne, *Brewer's CAP Theorem*, 2009. http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

[8] L. George, *HBase-The Definitive Guide: Random Access to Your Planet-Size Data*, O'Reilly Media, 2011.

[9] A. Nayak, A. Poriya, D. Poojary, "Type of NOSQL databases and its comparison with relational databases," *International Journal of Applied Information Systems*, vol. 5, no. 4, pp. 16-19, 2013.

[10] N. Roy-Hubara, A. Sturm, "Design methods for the new database era: a systematic literature review," *Software and Systems Modeling*, vol. 19, pp. 297-312, 2019. https://doi.org/10.1007/s10270-019-00739-8

[11] S. Ramzan, I. Bajwa, R. Kazmi, "Challenges in NoSQL-Based Distributed Data Storage: A Systematic Literature Review," *Electronics*, vol. 8, no. 5, e488, 2019. https://doi.org/10.3390/electronics8050488

[12] C. Zdepski, A. Bini, S. Matos, "New Perspectives for NoSQL Database Design: A Systematic Review," *American Academic Scientific Research Journal for Engineering, Technology, and Sciences*, vol. 68, no. 1, pp. 50-62, 2020.

[13] F. Mostajabi, A. Safaei. A. Sahafi, "A Systematic Review of Data Models for the Big Data Problem," *IEEE Access*, vol. 9, pp. 128889-128904, 2021. https://doi.org/10.1109/ACCESS.2021.3112880

[14] M. Genero, J. Cruz, M. Piattini, *Métodos de Investigación en Ingeniería de Software*, Editorial Ra-Ma, 2014.

[15] B. Kitchenham, "Procedures for performing systematic reviews," *Keele*, vol. 33, p. 28, 2004.

[16] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014. https://doi.org/10.1145/2601248.2601268

[17] T. Dybå, T. Dingsøyr, "Strength of evidence in systematic reviews in software engineering," in *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, 2008. https://doi.org/10.1145/1414004.1414034

[18] L. Yang, H. Zhang, H. Shen, X. Huang, X. Zhou, G. Rong, D. Shao, "Quality Assessment in Systematic Literature Reviews: A Software Engineering Perspective," *Information and Software Technology*, vol. 130, e106397, 2021. https://doi.org/10.1016/j.infsof.2020.106397

[19] M. Ivarsson, T. Gorschek, "A method for evaluating rigor and industrial relevance of technology evaluations," *Empirical Software Engineering*, vol. 16, pp. 365-395, 2020. https://doi.org/10.1007/s10664-010-9146-4

[20] S. Ramzan, I. Bajwa, B. Ramzan, W. Anwar, "Intelligent Data Engineering for Migration to NoSQL Based Secure Environments," *IEEE Access*, vol. 7, pp. 69042-69057, 2019. https://doi.org/10.1109/ACCESS.2019.2916912

[21] C. Fernández, D. Sevilla, J. García-Molina, "A Unified Metamodel for NoSQL and Relational Databases," *Information Systems*, vol. 104, e101898, 2022. https://doi.org/10.1016/j.is.2021.101898

[22] A. Frozza, R. Mello, R. "JS4Geo: a canonical JSON Schema for geographic data suitable to NoSQL databases," *GeoInformatica*, vol. 24, no. 4, pp. 1-33, 2020. https://doi.org/10.1007/s10707-020-00415-w

[23] R. Sellami, S. Bhiri, B. Defude, "Supporting Multi Data Stores Applications in Cloud," *IEEE Transactions on Services Computing*, vol. 9, pp. 59-71, 2016. https://doi.org/10.1109/TSC.2015.2441703

[24] P. Atzeni, F. Bugiotti, L. Cabibbo, R. Torlone, "Data modeling in the NoSQL world," *Computer Standards & Interfaces*, vol. 67, e003, 2020. https://doi.org/10.1016/j.csi.2016.10.003

[25] M. Eshtay, A. Sleit, M. Aldwairi, "Implementing Bi-Temporal Properties into Various NoSQL Database Categories," *International Journal of Computing*, vol. 18, no. 1, pp. 45-52, 2019. https://doi.org/10.47839/ijc.18.1.1272

[26] H. Shim, "PHash: A memory-efficient, high-performance key-value store for large-scale data-intensive applications," *Journal of Systems and Software*, vol. 123, pp. 33-44, 2017. https://doi.org/10.1016/j.jss.2016.09.047

[27] N. Roy-Hubara, P. Shoval, A. Sturm, "Selecting databases for Polyglot Persistence applications," *Data & Knowledge Engineering*, vol 137, e101950, 2021. https://doi.org/10.1016/j.datak.2021.101950

[28] Z. Lv, X. Li, H. Lv, W. Xiu, "BIM Big Data Storage in WebVRGIS," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2566-2573, 2019. https://doi.org/10.1109/TII.2019.2916689

[29] H. Yong, S. Dessloch, "Extracting deltas from column-oriented NoSQL databases for different incremental applications and diverse data targets," *Data & Knowledge Engineering*, vol. 93, pp. 42-59, 2014. https://doi.org/10.1016/j.datak.2014.07.002

[30] D. Zhang, Y. Wang, Z. Liu, S. Dai, "Improving NoSQL Storage Schema Based on Z-Curve for Spatial Vector Data," *IEEE Access*, vol. 7, pp. 78817-78829, 2019. https://doi.org/10.1109/ACCESS.2019.2922693

[31] X. Chai, Q. Wang, W. Chen, W. Wang, D. Wang, Y. Li, "Research on a Distributed Processing Model Based on Kafka for Large-Scale Seismic Waveform Data," *IEEE Access*, vol. 8, pp. 39971-39981, 2020. https://doi.org/10.1109/ACCESS.2020.2976660

[32] J. Song, H. He, R. Thomas, Y. Bao, G. Yu, "Haery: A Hadoop Based Query System on Accumulative and High-Dimensional Data Model for Big Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, pp. 1362-1377, 2020. https://doi.org/10.1109/TKDE.2019.2904056

[33] R. Ouanouki, A. April, A. Abran, A. Gomez, J. Desharnais, "Toward building RDB to HBase conversion rules," *Journal of Big Data*, vol. 4, no. 1, pp. 1-21, 2017. https://doi.org/10.1186/s40537-017-0071-x

[34] L. Bao, J. Yang, C.Q. Wu, H. Qi, X. Zhang, S. Cai, "XML2HBase: Storing and querying large collections of XML documents using a NoSQL database system," *Journal of Parallel and Distributed Computing*, vol. 161, pp. 83-99, 2022. https://doi.org/10.1016/j.jpdc.2021.11.003

[35] M. Mozaffari, E. Nazemi, A. Eftekhari-Moghadam, "CONST: Continuous online NoSQL schema tuning," *Software: Practice and Experience*, vol. 51, no. 5, pp. 1147-1169, 2021. https://doi.org/10.1002/spe.2945

[36] M. Mior, K. Salem, A. Aboulnaga, R. Liu, "NoSE: Schema Design for NoSQL Applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, pp. 2275-2289, 2017. https://doi.org/10.1109/TKDE.2017.2722412

[37] A. De la Vega, D. García-Saiz, C. Blanco, M. Zorrilla. P. Sánchez, "Mortadelo: Automatic generation of NoSQL stores from platform-independent data models," *Future Generation Computer Systems*, vol. 105, pp. 455-474, 2020. https://doi.org/10.1016/j.future.2019.11.032

[38] C. Zdepski, A. Bini, S. Matos, "PDDM: A Database Design Method for Polyglot Persistence," *American Academic Scientific Research Journal for Engineering, Technology, and Sciences*, vol. 71, pp. 136-152, 2020.

[39] M. Ansari, V. Vakili, B. Bahrak, "Evaluation of big data frameworks for analysis of smart grids." *Journal of Big Data*, vol. 6, pp. 1-14, 2019. https://doi.org/10.1186/s40537-019-0270-8

[40] S. Sengupta, S., Bhunia, "Secure Data Management in Cloudlet Assisted IoT Enabled e-Health Framework in Smart City," *IEEE Sensors Journal*, vol. 20, pp. 9581-9588, 2020. https://doi.org/10.1109/JSEN.2020.2988723

[41] H. Kim, E. Ko, Y. Jeon, K. Lee, "Techniques and guidelines for effective migration from RDBMS to NoSQL," *The Journal of Supercomputing*, vol. 76, no. 10, pp. 7936-7950, 2018. https://doi.org/10.1007/s11227-018-2361-2

[42] A. Turk, R. Selvitopi, H. Ferhatosmanoğlu, C. Aykanat, "Temporal Workload-Aware Replicated Partitioning for Social Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 11, pp. 2832-2845, 2014. https://doi.org/10.1109/TKDE.2014.2302291

[43] G. Baruffa, M. Femminella, M. Pergoles, G. Reali, "Comparison of MongoDB and Cassandra Databases for Spectrum Monitoring As-a-Service," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 346-360, 2019. https://doi.org/10.1109/TNSM.2019.2942475

[44] A. Hernández, D. Sevilla, J. García, S. Feliciano, "A Model-Driven Approach to Generate Schemas for Object-Document Mappers," *IEEE Access*, vol 7, pp. 59126-59142, 2019. https://doi.org/10.1109/ACCESS.2019.2915201

[45] P. Gómez, C. Roncancio, R. Casallas, "Analysis and evaluation of document-oriented structures," *Data & Knowledge Engineering*, vol. 134, e101893, 2021. https://doi.org/10.1016/j.datak.2021.101893

[46] E. Kuszera, L. Peres, M. Fabro, "Exploring data structure alternatives in the RDB to NoSQL document store conversion process," *Information Systems*, vol. 105, e101941, 2021. https://doi.org/10.1016/j.is.2021.101941

[47] G. Nys, R. Billen, "From consistency to flexibility: A simplified database schema for the management of CityJSON 3D city models," *Transactions in GIS*, vol. 25, no. 6, pp. 3048-3066, 2021. https://doi.org/10.1111/tgis.12807

[48] A. Maté, J. Peral, J. Trujillo, C. Blanco, D. García-Saiz, E. Fernández-Molina, " Improving security in NoSQL document databases through model-driven modernization," *Knowledge and Information Systems*, vol. 63, no. 8, pp. 2209-2230, 2021. https://doi.org/10.1007/s10115-021-01589-x

[49] S. Banerjee, A. Sarkar, "Ontology Driven Meta-Modeling for NoSQL Databases: A Conceptual Perspective," *International Journal of Software Engineering and its Applications*, vol. 10, no. 12, pp. 41-64, 2016. https://doi.org/10.14257/ijseia.2016.10.12.05

[50] C. Zdepski, A. Bini, S. Matos, "PDDM: A Database Design Method for Polyglot Persistence," *American Academic Scientific Research Journal for Engineering, Technology, and Sciences*, vol. 71, no. 1, pp. 136-152, 2020.

[51] A. Imam, S. Basri, R. Ahmad, A. Wahab, M. González-Aparicio, L. Capretz, A. Alazzawi, A. Balogun, "DSP: Schema Design for Non-Relational Applications," *Symmetry*, vol. 12, no. 11, e1799, 2020. https://doi.org/10.3390/sym12111799

[52] A. Hernández, J. Hoyos, J. García, D. Sevilla, "Discovering entity inheritance relationships in document stores," *Knowledge-Based Systems*, vol. 230, e107394, 2021. https://doi.org/10.1016/j.knosys.2021.107394

[53] I. Al Jawarneh, P. Bllavista, A. Corradi, L. Foschini, R. Montanari, "Efficient QoS-Aware Spatial Join Processing for Scalable NoSQL Storge Frameworks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2437-2449, 2020. https://doi.org/10.1109/TNSM.2020.3034150

[54] S. Sutedi, N. Setiawan, T. Adji, "Enhanced Graph Transforming V2 Algorithm for Non-Simple Graph in Big Data Pre-Processing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 1, pp. 67-77, 2020. https://doi.org/10.1109/TKDE.2018.2880971

[55] N. Mehmood, R. Culmone, L. Mostarda, "Modeling temporal aspects of sensor data for MongoDB NoSQL database," *Journal of Big Data*, vol. 4, no. 1, pp. 1-35, 2017. https://doi.org/10.1186/s40537-017-0068-5

[56] B. Namdeo, U. Suman, "Schema design advisor model for RDBMS to NoSQL database migration," *International Journal of Information Technology*, vol. 13, no. 1, pp. 277-286, 2020. https://doi.org/10.1007/s41870-020-00515-8

[57] B. Khalfi, C. De Runz, S. Faiz, H. Akdag, "A New Methodology for Storing Consistent Fuzzy Geospatial Data in Big Data Environment," *IEEE Transactions on Big Data*, vol. 7, no. 2, pp. 468-482, 2021. https://doi.org/10.1109/TBDATA.2017.2725904

[58] A. Sveen, "Efficient storage of heterogeneous geospatial data in spatial databases," *Journal of Big Data*, vol. 6, no. 1, pp. 1-14, 2019. https://doi.org/10.1186/s40537-019-0262-8

[59] M. Min, "Modeling and Implementation of Public Open Data in NoSQL Database," *International Journal of Internet, Broadcasting and Communication*, vol. 10, no. 3, pp. 51-58, 2018. https://doi.org/10.7236/IJIBC.2018.10.3.51

[60] K. Baker, P. Roehsner, T. Lake, D. Rivet, S. Benston, B. Bommersbach, W. Kirk, "Point-trained models in a grid environment: Transforming a potato late blight risk forecast for use with the National Digital Forecast Database," *Computers and Electronics in Agriculture*, vol. 105, pp. 1-8, 2014. https://doi.org/10.1016/j.compag.2014.04.002

[61] M. Hewasinghage, A. Abelló, J. Varga, E. Zimányi, "A cost model for random access queries in document stores," *The VLDB Journal*, vol. 30, no. 4, pp. 559-578, 2021. https://doi.org/10.1007/s00778-021-00660-x

[62] Y. Cheng, K. Zhou, J. Wang, P. D. Maeyer, T. V. Voorde, J. Yan, S. Cui, "A Comprehensive Study of Geochemical Data Storage Performance Based on Different Management Methods," *Remote Sensing*, vol. 13, no. 6, e3208, 2021. https://doi.org/10.3390/rs13163208

[63] A. Charef, B. Abdelkader, "Towards NoSQL-based Data Warehouse Solution integrating ECDIS for Maritime Navigation Decision Support System," *Informatica*, vol. 45, no. 3, e3204, 2021. https://doi.org/10.31449/inf.v45i3.3204

[64] E. Damiani, B. Oliboni, E. Quintarelli, L. Tanca, L. "A graph-based meta-model for heterogeneous data management," *Knowledge and Information Systems*, vol. 6, no. 1, pp. 107-136, 2018. https://doi.org/10.1007/s10115-018-1305-8

[65] M. Hewasinghage, J. Varga, A. Abelló, E. Zimányi, "Managing Polyglot Systems Metadata with Hypergraphs," *Data & Knowledge Engineering*, vol. 134, e101896, 2021. https://doi.org/10.1016/j.datak.2021.101896

[66] M. Sokolova, F. Gómez, L. Borisoglebskaya, "Migration from an SQL to a hybrid SQL/NoSQL data model," *Journal of Management Analytics*, vol. 7, no. 1, pp. 1-11, 2020. https://doi.org/10.1080/23270012.2019.1700401

[67] G. Demirci, H. Ferhatosmanoğlu, C. Aykanat, "Cascade-aware partitioning of large graph databases," *The VLDB Journal*, vol. 28, no. 3, pp. 329-350, 2019. https://doi.org/10.1007/s00778-018-0531-8

[68] C. Küçükkeçeci, A. Yazıcı, "Big Data Model Simulation on a Graph Database for Surveillance in Wireless Multimedia Sensor Networks," *Big Data Research*, vol. 11, pp. 33-43, 2018. https://doi.org/10.1016/j.bdr.2017.09.003

[69] Y. Hu, V. Gunapati, P. Zhao, D. Gordon, N. Wheeler, M. Hossain, T.LJ. Peshek, L. Bruckman, G. Zhang, R. French, "A Nonrelational Data Warehouse for the Analysis of Field and Laboratory Data from Multiple Heterogeneous Photovoltaic Test Sites," *IEEE Journal of Photovoltaics*, vol. 7, no. 1, pp. 230-236, 2017. https://doi.org/10.1109/JPHOTOV.2016.2626919

[70] C. Li, Q. Zhang, P. He, Z. Wang, L. Chen, "An agricultural data storage mechanism based on HBase," *International Journal of Information and Communication Technology*, vol. 14, no. 4, pp. 456-469, 2019. https://doi.org/10.1504/IJICT.2019.101864

[71] K. Santhiya, V. Bhuvaneswari, "An Automated MapReduce Framework for Crime Classification of News Articles Using MongoDB, " *International Journal of Applied Engineering Research*, vol. 13, no. 1, pp. 131-136, 2018.

[72] J. Zeng, B. Plale, "Argus: A Multi-tenancy NoSQL store with workload-aware resource reservation," *Parallel Computing*, vol. 58, pp. 76-89, 2016. https://doi.org/10.1016/j.parco.2016.06.003

[73] J. Yoon, D. Jeong, C. Kang, S. Lee, "Forensic investigation framework for the document store NoSQL DBMS: MongoDB as a case study," *Digital Investigation*, vol. 17, pp. 53-65, 2016. https://doi.org/10.1016/j.diin.2016.03.003

[74] H. Asri, H. Mousannif,, H. Moatassime, "Reality mining and predictive analytics for building smart applications," *Journal of Big Data*, vol. 6, pp. 1-25, 2019. https://doi.org/10.1186/s40537-019-0227-y