Data Modeling and NoSQL Databases - A Systematic Mapping Review

HARLEY VERA-OLIVERA, GUO RUIZHE, and MARISTELA HOLANDA, Department of Computer Science, University of Brasília, Brasil
RUBEN CRUZ HUACARPUMA, Software AG, Brasil
ANA PAULA BERNARDI DA SILVA, Master in Governance, Technologies and Innovation, Catholic University of Brasília, Brasil
ARI MELO MARIANO, Department of Production Engineering, University of Brasília, Brasil

Modeling is one of the most important steps in developing a database. In traditional databases, the **Entity Relationship (ER)** and **Unified Modeling Language (UML)** models are widely used. But how are NoSQL databases being modeled? We performed a systematic mapping review to answer three research questions to identify and analyze the levels of representation, models used, and contexts where the modeling process occurred in the main categories of NoSQL databases. We found 54 primary studies where we identified that conceptual and logical levels received more attention than the physical level of representation. The UML, ER, and new notation based on ER and UML were adapted to model NoSQL databases, in the same way, formats such as JSON, XML, and XMI were used to generate schemas through the three levels of representation. New contexts such as benchmark, evaluations, migration, and schema generation were identified, as well as new features to be considered for modeling NoSQL databases, such as the number of records by entities, CRUD operations, and system requirements (availability, consistency, or scalability). Additionally, a coupling and co-citation analysis was carried out to identify relevant works and researchers.

CCS Concepts: • **Information systems** → **Database design and models;**

Additional Key Words and Phrases: Data modeling, NoSQL databases, systematic mapping

ACM Reference format:
Harley Vera-Olivera, Guo Ruizhe, Maristela Holanda, Ruben Cruz Huacarpuma, Ana Paula Bernardi da Silva, and Ari Melo Mariano. 2021. Data Modeling and NoSQL Databases - A Systematic Mapping Review. *ACM Comput. Surv.* 54, 6, Article 116 (July 2021), 26 pages.
https://doi.org/10.1145/3457608

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Authors' addresses: H. Vera-Olivera (corresponding author), G. Ruizhe, and M. Holanda, Department of Computer Science, University of Brasília, Campus Universitário Darcy Ribeiro, Brasília, DF, Brasil; emails: harleyve@gmail.com, guo_ruizhe@163.com, maristela.holanda@gmail.com; R. C. Huacarpuma, Software AG, Brasília - DF, Brasil; email: ruben.cruz@softwareag.com; A. P. B. da Silva, Master in Governance, Technologies and Innovation, Catholic University of Brasília, Brasília, Brasil; email: anap.bernardi@gmail.com; A. M. Mariano, Department of Production Engineering, University of Brasília, Campus Universitário Darcy Ribeiro, Brasília, DF, Brasil; email: arimariano@unb.br.

0360-0300/2021/07-ART116 $15.00
https://doi.org/10.1145/3457608

1 INTRODUCTION

The advance and evolution of social networks, the Internet of Things, web technology, and mobile devices has resulted in the explosion of structured, semi-structured, and unstructured data generated by applications around the world. Individuals and organizations produce large quantities of data at a high-speed rate [7, 21]. This enormous data volume is usually called big data. However, as technology and big data applications have grown and new needs such as availability or flexibility have appeared, the traditional relational database has become less equipped to manage rapidly growing large data volumes and the complexities of data structures.

To store a large volume of data, NoSQL database systems are commonly used and there are many different NoSQL families [19]. However, NoSQL databases have some design characteristics in common [21]. First, they adopt more flexible data models that are mostly schemaless. Second, eventual consistency transactions are reached by relaxing ACID properties to scale-out while achieving high availability and low latency [25]. Third, query performance is not only achieved by co-locating data, but rather mainly via horizontal and elastic scalability [25]. Fourth, data can simply be replicated and horizontally partitioned over local and remote servers [25].

Before being stored, big data needs to be modeled; in fact, one of the most critical stages in the development of an information system is a data model for the database [60]. A data model is used to represent, abstract, and limit a problem in the real world by graphics and symbols. At high-level, concepts that are familiar with the way users note data are provided by conceptual models. At low-level, concepts that describe the aspects of how data are saved on the computer are provided by physical models. Within these two limits exists a class of representational models, which provide concepts that may be easily understood by end-users, although it is not extremely far removed from the form in which data are organized on the computer.

But modeling NoSQL databases is not an easy task because of heterogeneity. Currently, there are many NoSQL databases available with different consistency and durability guarantees, different data access APIs, and different characteristics of the data model. This vast heterogeneity has opened up new problems for designers. Moreover, a complete and fully standardized data modeling procedure for NoSQL databases does not exist yet. There are also challenges in data modeling for NoSQL databases, such as at how many levels of representation (conceptual, logical, or physical) should a NoSQL database be modeled, what models (ER, UML, etc.) should be used at each level of representation for each category of NoSQL database and so on. In our article, we will use the classification based on [21, 37]:

- Graph databases model data as a graph structure including nodes and edges describing the relationships between nodes. Each node may also include properties that describe data included within each object. On the other hand, edges may also have their properties.
- In column-oriented databases, a tabular format of rows and column families is used to represent data. The schema of a column-family is flexible and a column has a name and a value with a complex or simple structure.
- Key-value databases store data as an associative collection of entries. Every element in the database is put away as an attribute name (“key”), together with its value.
- Document-oriented databases are extended key-value stores in which the value is represented as a document encoded in conventional semistructured formats such as JSON, or Binary JSON (BSON). In document-oriented databases, a document can contain denormalized data. The choice of denormalizing data depends on the end-user.

Most NoSQL databases are schema-less; however Atzeni [10] argues that data modeling effects can be useful and illustrates two research directions. First, the diversity of systems and models could create difficulties for developers and their organizations so modeling-based approaches leverage the necessity for standardization and regular access. Second, data models can give a basis of a description of generic approaches to the logical and physical design. Likewise, Kaur and Rani [37], argue that there is yet a need to model NoSQL databases to adequately understand the storage of data. Additionally, Gomez et al. [27] and Reis et al. [50] show an experiment utilizing MongoDB with many alternative data models and a set of queries. They used these to demonstrate how structuring data in NoSQL databases has a large impact on data size, readability of the code, and query performance, all factors which influence maintainability and software debugging.

In this context, this systematic mapping aims to answer three research questions: (1) What levels of representation are presented in the methodologies for modeling NoSQL databases?; (2) What models are used at each level of representation for modeling NoSQL databases?; and (3) In what context does the modeling process occur? The analysis performed on 54 primary studies selected from 2008 to the first semester of 2019 showed that the UML, ER, and new notation based on ER and UML were adapted to model NoSQL databases. In the same way, formats such as JSON, XML, and XMI were used to generate schemas through the three levels of representation. Conceptual and logical levels received more attention than the physical level of representation and new contexts such as migration, evaluations, benchmark, and schema generation were identified. Moreover, new features to be considered for modeling NoSQL databases such as CRUD operations, number of records by entities, system requirements (availability, consistency, or scalability) were identified. Additionally, a coupling and co-citation analysis was carried out to identify relevant works and researchers in the area of data modeling of NoSQL databases.

The article is structured as follows: Section 2 shows the background to NoSQL databases and data modeling. Section 3 shows related works. The systematic mapping method is shown in Section 4, including the identification of the need for mapping, defining the research questions and defining the protocol to be used. Section 5 gives the result of our systematic mapping. A discussion of the results is presented in Section 6. In Section 7, the threats to validity are outlined. Finally, in Section 8, our conclusions are presented.

2 NOSQL DATABASES

The rapid growth of technologies such as the Internet of Things has resulted in the explosion of structured, semistructured, and unstructured data. In the past, relational databases were used to store data, but high data traffic has rendered them incapable of sustaining these new features [25]. New requirements also emerged to support the large amount of data, including horizontal scalability, high availability, fault tolerance, schema-less design and eventual consistency [21]. For traditional **Relational Database Management Systems (RDBMS),** it is very difficult to meet these requirements mainly for the following reasons [21]: First, the growth of data in these systems requires its transfer to new servers with upgraded hardware. Second, the complexity and cost of joining distributed normalized data are increased by scaling out the RDBMSs. Third, predefined schemas in relational databases make the evolution costly due to the complexity of data migration. These three requirements are complex and can be tackled by sacrificing the ones that are not required according to the requirements of applications. In this way, a new generation of non-relational databases, called NoSQL, emerged.

A strong consistency model is implemented by relational databases [25], where the transactions are immediately committed, and clients operate over correct data states. Reads over the same data will present an equal value to all client requests. Unlike relational databases, NoSQL databases can be scaled horizontally over thousands of servers but don't offer the same level of data consistency as relational databases. Although horizontal scalability may seem better, the CAP theorem [14] shows that when network partitions occur, designers have to choose between consistency and availability. Some NoSQL database designers choose higher availability over a strict consistency strategy, this approach is known as BASE (Basically Available, Soft-state, and Eventually consistent).

2.1 NoSQL Data Models

A data model is a representation of entities and their relationships of a problem in the real world. Entities, relationships, attributes, and cardinalities are the primary elements or constructions of a model. Although the constructions, defined in ER and UML, are widely used to model in relational databases, we wondered if those constructs work well for modeling NoSQL databases and big data and their new features.

In relational databases, three levels of representations are well known by the academic and business community: the conceptual, logical, and physical models. At a low level, the physical model depends on the specifications and semantics of the target **database management system (DBMS)**. Physical design is performed by converting logical design into table definitions, including pre-deployment design, table definitions, normalization, primary and foreign key relationships, and basic indexing. At a high-level, a problem in the real world is represented by the conceptual model. ER and UML are the most used models to represent a real problem using ER and class diagrams, respectively. Between these two models, a logical model defines how the system should be implemented regardless of the DBMS. In this level relational or dedicated models (with a metamodel using UML class diagram) are used frequently to represent the data model. On the other hand, it is not clear yet how many levels of representation should be used, and what models should be used for each level of representation in NoSQL databases.

Although NoSQL databases are schema-less and more flexible, there are more benefits if NoSQL databases are previously modeled [10, 37]. But at this point, some questions arise like: Which is the methodology or standard used to model NoSQL databases? Is there a standard widely accepted by the academic and business community? How many levels of representation should be used to model NoSQL databases? Obtaining answers to these questions is not a trivial task because of the heterogeneity of NoSQL databases. However, many researchers are proposing new methods to model NoSQL databases [9].

NoSQL databases are essentially categorized into graph, document-oriented, key-value and column. A determined number of features characterize each of these NoSQL databases and make them adequate for a specific application scenario.

2.1.1 Key-Value. In this NoSQL database, data are represented as pairs (*key, value*) stored in key-based, high-scalable and efficient lookup structures such as Log-Structured Merge-trees and Distributed Hash Tables [21]. A *key* is simple or complex; it can be defined either by the application or by the key-value system. A *value* represents data that can be of different structure, type, or size duly represented by a unique indexed *key*. Querying and indexing based on *values* are not supported by the Key-value systems due to the schemaless structure of stored *values*. However, advanced key-value systems provide extra functionalities for querying and indexing the content of *values* of specific data types like document (Riak KV), tabular (IBM Spinnaker, HyperDex, and Yahoo Pnuts), or list data types (Redis and Aerospike) [21].

Key-value systems can be classified into three kinds, based on data persistence [21]: *in-memory key-value systems*, that provide notably fast access to data by keeping it in memory, such as Memcached; *persistence key-value systems*, that provide access to data by being stored in HDD/SSD, such as Riak KV; *hybrid key-value systems*, that keep data in memory and store them when some conditions are satisfied, such as Redis and Aerospike.

The most representative key-value systems are Redis, Memcached, Hazelcast, etcd, Ehcache, Aerospike, and Riak KV.

2.1.2 Column. In this NoSQL database, data are represented in a tabular format of *rows* and a determined number of *column-families* with names and values that are logically related to each other. This is the reason why data in column systems are physically stored in columns instead of rows. *Super-column-families* is a feature that some column systems such as Cassandra offer. This means that a *column-family* is nested by another *column-family* [21]. A *primary key,* composed by one or more columns, is used to identify each row in the table. The *primary keys* are used in column systems to provide row uniqueness and to assure that some sets of rows are stored in the same physical node, in this way these rows can be quickly retrieved together. For this purpose, some column systems like Cassandra and ScyllaDB [22] divide the *primary key* into the *partition key* and the *clustering key*. The *partition key* is used to spread rows over the same node, with the aim that all the rows with the same *partition key* are stored in the same node. The *clustering key*, which can be optional, is used to sort rows with the aim of being easily retrieved. The elements that share the same *partition key* value are sorted according to their *clustering key*. On the other hand, in systems such as HBase or BigTable, columns of a table are grouped into disjoint sets that are never retrieved together [22].

For performance purposes, some column systems (Cassandra) restrict query access to the set of rows connected with a single *partition key*, the retrieval of data from different partitions in a single query is not possible. Column systems do not provide support for joins between tables. On the other hand, the denormalization in column systems has a side effect, namely the expensive insertion of new data or updating existing ones [22].

Some representative column systems are Cassandra, Hbase, Datastax Enterprise, Microsoft Azure Table Storage, Accumulo, and Google Cloud Bigtable.

2.1.3 Document-Oriented. A document-oriented system is considered as an extension of key-value systems [21] where values are represented as *documents* and each document is represented by a unique key. A document can be encoded in standard formats such as XML, JSON, or BSON (Binary JSON). A *document* has a flexible schema, where attributes can be removed or added at runtime and can have more than one value. Search functionalities and indices based on their attribute names and values are supported by document-oriented databases.

Collection or *bucket* is an *aggregate* offered by systems such as MongoDB and Couchbase Server [21]. It consists of a set of *documents* describing the same kind of information. A *collection* is similar to a *table* in relational databases, where a *document* with a unique key, but not necessarily with the same schema as other *documents*, represents a *row* in a *table*. In this way, using *collections*, features like security, persistence, replication, and resources can be managed for each group of documents.

Reference or embedding documents are supported by document-oriented databases for model relationships between documents. To avoid requiring a join operation, designers can embed one or many relationships in a single document, so applications do not need to join data from documents, and a single request can retrieve all data. On the other hand, document-oriented databases such as ArangoDB and OrientDB use referenced documents to develop a hybrid graph/document data model [21].

The most commonly document-oriented systems in this category are MongoDB, Couchbase, Firebase Realtime Database, CouchDB, Realm, Google Cloud Firestore, RethinkDB, and RavenDB.

2.1.4 Graph. In these databases, entities are represented by *vertices* and relationships are represented by *edges*, it is based on the graph-theoretical foundation. Some graph structures are *undirected/directed graphs, labeled graphs, attributed graphs, multigraphs, hypergraphs, and nested graphs* [?]. Because these structures are not mutually exclusive, the property graph can be a combination of *directed*, *labeled*, and *attributed* structures, for example.

In an *undirected* graph, all relationships are bidirectional, but in a *directed* graph each edge from one vertex to another is a directed tuple. In *labeled* graphs, edges and vertices are tagged with scalar values; these labels may represent roles in different domains or attached data. To represent the properties of graphs, a list of attributes is attached to edges and vertices. It is usually applied in social networking applications with social interaction between users. A *multigraph* has multiple edges between two vertices even with the same label, and self-loops are allowed, too. A *hypergraph* is a generalization of a graph. In contrast with ordinary graphs where an edge connects exactly two vertices, a hyperedge can join any number of vertices. Finally, in a *nested* graph, each vertex can contain another graph.

Storage techniques are categorized into *native* and *non-native* graph stores [?]. Based on the features of the graph data model, the storage is adapted in a *native* graph store. To optimize the storage, three well-known techniques are used: *Compressed Sparse Row* (CSR), *adjacency list*, and *edge list*. The most representative graph NoSQL databases are Neo4j, JanusGraph, GraphDB, Dgraph, TigerGraph, and OrientDB.

This broad heterogeneity has opened up new problems and challenges for designers. Additionally, there is no standard to model NoSQL databases accepted by both the business and academic communities.

3 RELATED WORK

With the emergence of NoSQL databases and the need to model these databases, many academic works have been published to fill this knowledge gap. Table 1 presents academic works that have been produced as expert surveys or assessments but none of them as systematic mapping. Although data modeling is mentioned as a part of the work, in [30] and [31], the concept of data modeling is related to the structure, organization, and the method of storing data in NoSQL systems, but not how to model the data using levels of representation and specific models. In the same way, Gil and Song [26] introduced the term “conceptual modeling” for big data using techniques like ontology, semantics, RDF schemas, and SPARQL Language. This work discussed the strengths and weaknesses of the different benchmark design approaches.

In 2018, two academic works were published: the first one [21] a general discussion on four non-orthogonal model bases of distributed database systems: data partitioning, data models, the CAP theorem, and consistency model. Again, the data model in this academic work is just understanding in the sense of organization, structure and the way of storing data. The second one is the academic work [9], a survey precisely focused on the data model of NoSQL databases, and despite not being a systematic mapping, important information was found. The main focus of this academic work is the methodology applied by different works. For this, a classification of methodologies was proposed, first by works aiming to unify most NoSQL databases under a uniform design methodology and second, by categories of NoSQL databases.

In the next section, our systematic mapping planning, and systematic mapping implementation are presented.

4 SYSTEMATIC MAPPING METHOD

In this section, we will define and justify the research questions and the systematic mapping protocol to conduct the systematic mapping.

Table 1. Academic Works about Data Modeling in NoSQL Databases

Review Study	Year	Context
Hencht & Jablonski [31]	2011	NoSQL Databases
Han et al. [30]	2011	NoSQL Databases
Gil & Song [26]	2016	Big Data modeling and management
Assad & Baïna [9]	2018	NoSQL databases design methodology
Davoudian et al. [21]	2018	NoSQL Databases

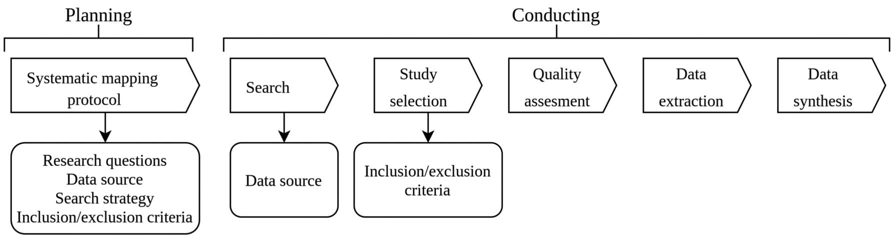

Fig. 1. Systematic mapping process adapted from [39].

4.1 Systematic Mapping Planning

For planning the systematic mapping review, two guidelines were considered: Petersen et al. [48] and Kitchenham et al. [39]. The first one is designed to give a summary of a research area through classification and counting contributions about the categories of that classification. The second one is a method for identifying, evaluating, and interpreting all possible research relevant to a particular topic area. We set our systematic mapping method based on [48] and our analysis on [39]. Figure 1 illustrates the adopted protocol.

4.1.1 Need for a Systematic Mapping. New ways of storing data started with the emergence NoSQL databases. Some of these new technologies for storing data, like key-value, document-oriented, column, graphs, and others have properties that allow more reliable performance when saving a large quantity of data. Nonetheless, there are no methodologies for data modeling widely admitted by the business community and the academic world in these new technologies. Though there are clearly three levels of representation, not all of them are used to model, nor has it been determined which model to use at each level of representation. New proposals to model data are emerging in NoSQL databases but due to the heterogeneity the number of levels of representation is not clear, nor are the models used in each level, or even the construction used for each model. Our interest is to conduct research in this area, and given that there are still no academic works that answer the research questions posed, we consider this to be a relevant undertaking.

A systematic mapping was chosen to carry out this mapping because of the methodological advantage, since its process minimizes the possibility of bias whereas it maximizes the coverage possibility, moreover, systematic mapping studies have not yet been conducted. Also, this will be a starting point for future research work.

Table 2. Generic Search String

((design OR model OR "data structure" OR "relational data") AND nosql) OR (modeling AND (key-value OR column OR document OR graph) AND (nosql OR database OR schema))

4.1.2 Research Questions. The purpose of this mapping research is to find out how the different models applied in traditional databases are being applied to NoSQL databases. For this proposal, three research questions (RQ) were defined.

RQ1. *What levels of representation are presented in the methodologies for modeling NoSQL databases?* The purpose of this question is to identify which and how many levels of representation (conceptual, logical, and physical) are being applied in the process of data modeling in NoSQL databases. The level of representation used to model a NoSQL database will influence the properties or specifications to be considered for the modeling process.

RQ2. *What models are used at each level of representation for modeling NoSQL databases?* The purpose of this question is to identify what models are used according to the number of levels of representation proposed in the different methodologies for modeling NoSQL databases.

RQ3. *In what context does the modeling process occur?* This question seeks to answer how the models were used in some particular contexts of methodologies to model NoSQL databases, for example, in design guidelines, schema recommendation, migration tools, case studies, or benchmarking.

4.1.3 Systematic Mapping Protocol. A protocol defines in detail the steps to perform mapping in a specific domain of knowledge [39]. In that sense, we define data sources, determine search strategy, select primary studies by following a selection strategy, extract information from primary studies by a method to extract and finally summarize the data. As data sources to search for candidates for primary studies, we chose ISI Web of Science (WoS) and Scopus. These databases have mainly conferences and journals in computer science. To define the search string, initially seed keywords determined from the three research questions were selected. The keywords were "model", "NoSQL database", "graph", "document", "column", and "key-value". To limit the search to NoSQL databases the "NoSQL database" keyword was used. To search academic works directly related to our research the keywords "graph model", "document model", "column model", "key-value model" and "model" were used. To search academic works related to the process or method of modeling the keyword "method" was used. Additionally, to identify more representative keywords and maximize the number of candidate academic works for primary studies, a deeper analysis was performed. For this purpose, we used the Scopus database and Vosviewer[1] software to find new keywords related to data modeling for NoSQL databases.

As a result of the additional analysis, many important new keywords were identified. Finally, the new set of keywords was defined as the generic search string shown in Table 2. The generic search string was determined with the aim of getting the largest number of candidates for primary studies. It was not possible to apply the generic search string to Scopus and Web of Science because of tool constraints, therefore we adjusted the generic search string shown in Table 2 to apply to both search engines chosen.

Inclusion and exclusion criteria were specified as a selection strategy for relevant primary studies, considering the purpose of mapping and research questions. The criteria were set to

[1]http://www.vosviewer.com/.

guarantee that only relevant studies and academic works related to the context of modeling of NoSQL databases should be selected.

The inclusion criteria defined are:

- *IC-1.* Any academic work that primarily addresses data modeling for NoSQL databases as an essential part of the study.
- *IC-2.* Any academic work that was published in conferences or journals.

The exclusion criteria are:

- *EC-1.* The academic work is not available on the web.
- *EC-2.* The academic work is not in English.
- *EC-3.* The academic work was published as a short paper.
- *EC-4.* The academic work is a review, such as a survey or a systematic mapping or systematic review.
- *EC-5.* The academic work is primarily related to an area other than information systems, computer science, and engineering.
- *EC-6.* Academic works presenting non-peer reviewed material.
- *EC-7.* Academic works outside the range 2008 to 2019.

To include a work into primary studies, both IC-1 and IC-2 criteria must be fulfilled. Any work excluded by EC-4 will be discussed.

4.2 Systematic Mapping

To conduct the mapping, the following steps were necessary: identifying academic works using the search string, selecting the primary studies, checking the quality of primary studies, extracting and synthesizing data. For this work, we had two students (Harley and Guo), one external collaborator (Ruben), and three supervisors (Ana, Ari, Maristela). Students, collaborator, and supervisors are all authors of this article.

4.2.1 Identifying Primary Studies. To identify the candidates for primary studies, first, we applied the generic search string (Table 2) to the Scopus and WoS search engines. As a result, we obtained 6,508 documents in total from the two sources. Moreover, a manual search (Snowballing technique) was carried out in Google Scholar with the aim of finding more academic works related to data modeling for NoSQL databases.

4.2.2 Selecting Primary Studies. In this step, we applied the exclusion and inclusion criteria through the process shown in Figure 2, the objective of this step is to filter academic works that are not relevant for our study. For this work, the two students together with the collaborator participated. First, we removed academic works before 2008 (EC-7), and as a result, 2,038 academic works remained. Second, the exclusion and inclusion criteria EC-1 to EC-6 and IC-1 were applied and, as a result of this step, 60 academic works remained. Third, the snowballing technique was applied adding 11 new academic works. Finally, the full text-reading step was applied to the academic works leaving 59 academic works.

4.2.3 Assessing the Relevance of Primary Studies. For this step, the students, collaborator, and supervisor participated. The main objective of this step was to eliminate those academic works with less relevance in data modeling for NoSQL databases. As a result of this step, five extra academic works were eliminated exclusively for relevance reasons. Finally, a complete list of academic works selected as primary studies is shown in Table 10. They are classified by the contexts where the process of data modeling occurred in NoSQL databases.

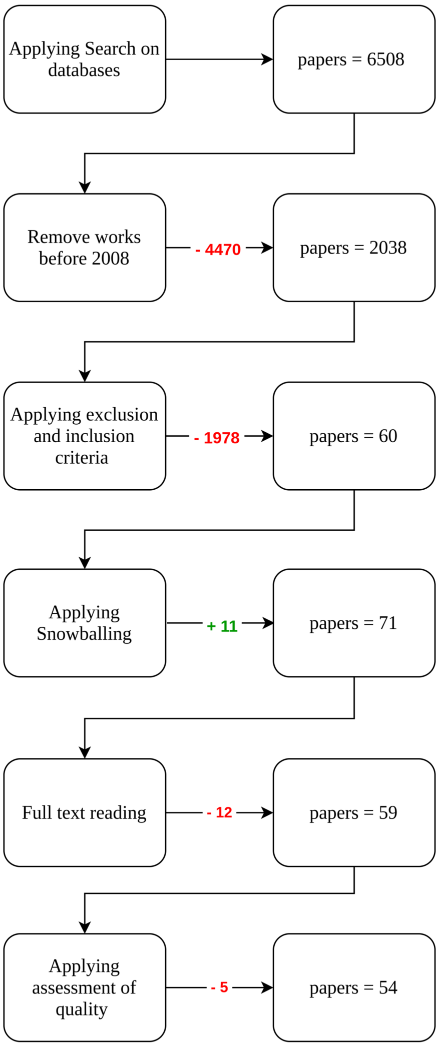

Fig. 2. The study selection process.

4.2.4 Extracting Data from Primary Studies. In this step, the students and collaborator analyzed in more detail the primary studies. We read the selected academic works by reviewing their titles, abstracts, keywords, and, when necessary, introductions and conclusions. It was performed through a detailed content analysis driven by identifying keywords and graphics representing methods, models, and levels of representation within the whole academic work. In this step, the two students were responsible for extracting the data, they completed the data extraction form and the collaborator checked the data, and confirmed that the extraction of data was correct. Whenever there were doubts about extracting data in some primary studies, supervisors were contacted to resolve the doubts. To extract data from primary studies, we used a form with generic and common attributes like title, abstract, and keywords. In addition to these attributes, we defined the attributes' levels of representation, type of NoSQL database, context of data modeling, and models.

Table 3. Form Items to Extract Data from Primary Studies

Data Item	Value	RQ
Title	Name of the article	RQ1, RQ2, RQ3
Abstract	Abstract of the article	RQ1, RQ2, RQ3
Keywords	Keywords of the article	RQ1, RQ2, RQ3
Levels of representation	Levels of representation for modeling NoSQL databases (conceptual, logical, or physical)	RQ1, RQ2
Type of NoSQL database	Document-oriented, graph, column or key-value	RQ1, RQ2
Context of data modeling	The context of data modeling, for example, guideline, benchmark, or another	RQ1, RQ3
Model	A specific model used to model in a level of representation, for example ER or UML	RQ2

For *RQ1* ("What levels of representation are presented in the methodologies for modeling NoSQL databases?"), six attributes (see column RQ in Table 3) were chosen for data extraction and to assess this research question. From this information, we identified the levels of representation proposed by primary studies selected based on the type of NoSQL databases.

For *RQ2* ("What models are used at each level of representation for modeling NoSQL databases?"), six attributes (see column RQ in Table 3) were chosen for data extraction and to assess this research question. The models used in each level of representation were identified along with the types of NoSQL databases.

For *RQ3* ("In what context does the modeling process occur?"), four attributes were selected (see column RQ in the Table 3). From this information, we identified the context of the methodologies proposed by primary studies selected, for example, guidelines, process transformation, benchmark, and so on.

5 RESULTS

In this section, a general description of primary studies and answers to research questions are presented. Section 5.1 presents a description type, authors, and origin of primary studies, as well as a coupling and co-citation analysis to identify relevant authors and works in the area. Section 5.2 answers the first research question (RQ1) on the levels of representation used for modeling NoSQL databases. Section 5.3 answers the second research question (RQ2) about the models used at each level of representation. Finally, Section 5.4 presents the contexts where the modeling process occurs.

5.1 General Description of Primary Studies

In this section, results are shown from the extraction and synthesis process according to our protocol (Section 4.1.3).

In general terms, Figure 3 shows the distribution of primary studies over the years of study (from 2008 to the first semester of 2019). The figure shows data by publication type (journal or conference). Symposiums and workshops are considered as conferences. The growth in the number of studies in NoSQL database modeling is constant over the years. Academic works for NoSQL database modeling appeared at conferences starting in 2011, while academic works published in journals have only appeared since 2016. Twenty three percent of academic works selected as primary studies were published in journals while 77% of academic works were published in conferences.

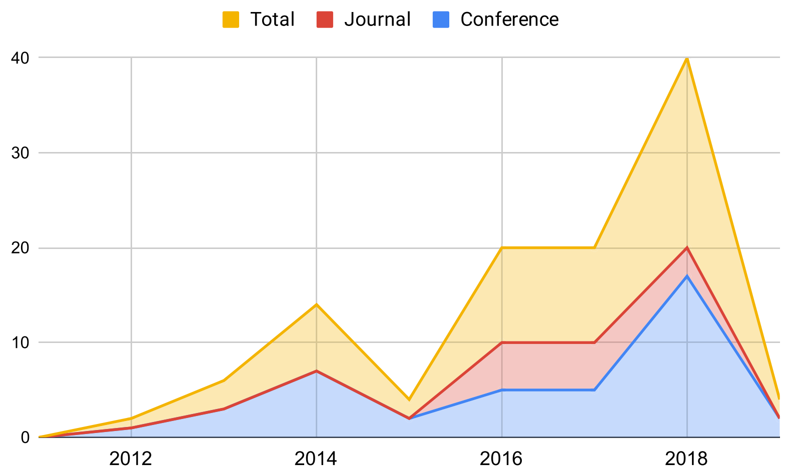

Fig. 3. Visual distribution of primary studies over the years.

Table 4. Main Conferences

Initials	Conference name
ER	International Conference on Conceptual Modelling
WorldCIST	World Conference on Information Systems and Technologies
TENCON	IEEE Region 10 Annual International Conference
BIGDATA CONGRESS	IEEE International Congress on Big Data
IIWAS	International Conference on Information Integration and Web-based Applications & Services
IDEAS	International Database Engineering and Applications Symposium

Table 4 shows the most relevant conferences found in our primary studies in terms of the number of publications. One of the most notable conferences in the area of data modeling is the "International Conference on Conceptual Modelling - ER". In this conference, four papers related to data modeling for NoSQL databases were published. The articles published in journals were spread over 13 journals. Table 5 shows the more relevant journals, in this table the relevance of journals is shown with the H-Index and SJR metrics. The 54 academic works selected were produced by 185 authors, from 91 research institutions public and private, located in 30 countries. Such numbers show how diffuse the research in data modeling for NoSQL databases is. Figure 4 shows all countries that contributed to the production of the 54 academic works selected related to data modeling for NoSQL databases. The most productive countries, sorted by the number of academic works produced, are France followed by Spain, Brazil, India, and China.

The most productive institutions in data modeling of NoSQL databases are shown in Table 6 and it is notable that Europe and Asia are the most productive continents in this research area. In particular, France has been working with a strong collaboration between private companies (TRIMANE), research institutes (CEDRIC-CNAM) and universities. Table 7 shows the top 10 most productive researchers in data modeling of NoSQL databases, once again the participation of French researchers is noticeable. Although Table 7 shows the most productive authors based on the number of publications, it does not mean that they are the most influential within the area of NoSQL

Table 5. More Relevant Journals

H-Index	SJR	Journal title	Publisher
148	1.14	IEEE Transactions on Knowledge and Data Engineering	IEEE Computer Society
17	1.12	Journal of Big Data	Springer Open
189	0.72	Communications of the ACM	Association for Computing Machinery (ACM)
56	0.7	Knowledge and Information Systems	Springer London
42	0.65	Enterprise Information Systems	Taylor and Francis Ltd.
19	0.2	International Journal of Data Warehousing and Mining	IGI Publishing

Table 6. The top Nine Most Productive Research Institution in Data Modeling for NoSQL Databases from 2008 to the First Semester of 2019

Pos	Research Institution	Country	No Studies
1	TRIMANE	France	6
2	Centre d'études et de recherche en informatique et communications (CEDRIC-CNAM)	France	5
3	Toulouse Capitole University	France	4
4	National Institute of Technology	India	4
5	Ben-Gurion University of the Negev	Israel	3
6	Ahmadu Bello University	Nigeria	3
7	Babes-Bolyai University	Romania	3
8	University of Brasilia	Brazil	3
9	Universita Roma TRE	Italy	3

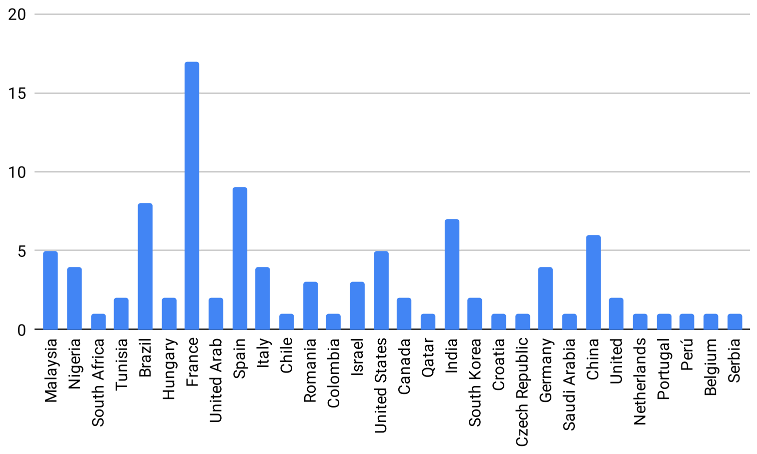

Fig. 4. Visual distribution of production of academic works by countries.

Table 7. The Top 10 Most Productive Researchers in Data Modeling for NoSQL Databases from 2008 to first Semester of 2019

Pos	Researcher	Research Institution	Country	# Studies
1	Abdelhedi F.	TRIMANE and IRIT (Institut de Recherche en Informatique de Toulouse)	France	5
2	Ait-Brahim A.	Toulouse Capitole University & IRIT (Institut de Recherche en Informatique de Toulouse)	France	5
3	Zurfluh G.	Toulouse Capitole University	France	5
4	Atigui F.	Centre d'études et de recherche en informatique et communications	France	5
5	Gonzalez Aparicio M. T.	University of Oviedo Gijon	Spain	4
6	Imam A.A	Universiti Teknologi PETRONAS	Malaysia	4
7	Basri S.	Universiti Teknologi PETRONAS	Malaysia	4
8	Ahmad R.	Universiti Teknologi PETRONAS	Malaysia	4
9	Sarkar A.	National Institute of Technology	India	3
10	Banerjee S.	National Institute of Technology	India	3

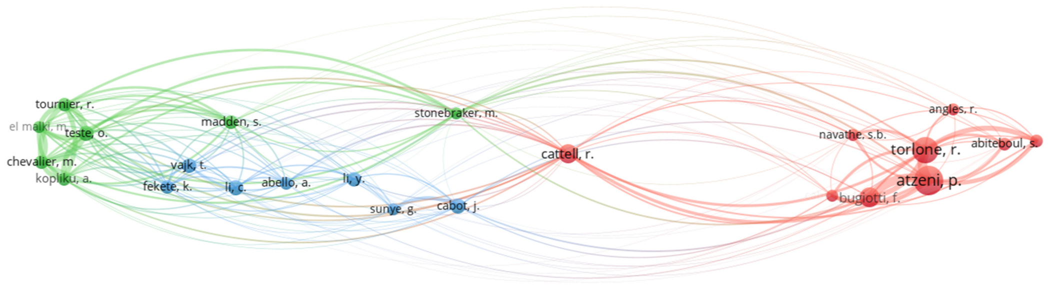

Fig. 5. Co-citation network map of authors.

database modeling. For this reason, a cocitation analysis was carried out using VOSviewer software, Figure 5 shows a network map of the cocitation analysis between 2008 and the first semester of 2019. The closer the authors, the greater the similarity of the works and the larger their name, the more cited he or she was. From Figure 5, three important authors can be identified as Riccardo Torlone, Paolo Atzeni, and Francesca Bugiotti.

Additionally, a coupling analysis was carried out in the 54 primary studies selected to find those works that are at the forefront of research in the area of data modeling of NoSQL databases. Figure 6 shows a coupling analysis. According to the network map, three authors stand out: Kaur and Rani [37] with the academic work "Modeling and Querying Data in NoSQL Databases", Bugiotti et al. [15] with the academic work "Database Design for NoSQL Systems," and Chebotko et al. [16] with the academic work "A Big Data Modeling Methodology for Apache Cassandra".

5.2 Levels of Representation for Modeling NoSQL Databases

After the extraction of items shown in Table 3, we answered this research question (RQ1) according to each category of NoSQL databases.

Figure 7 shows the levels of representation by each category. In the case of document-oriented NoSQL databases, academic works focus on modeling in the logical level of representation, while in two levels of representation works focus on the logical-physical levels of representation. No work with three levels of representation was found. For column NoSQL databases, the literature

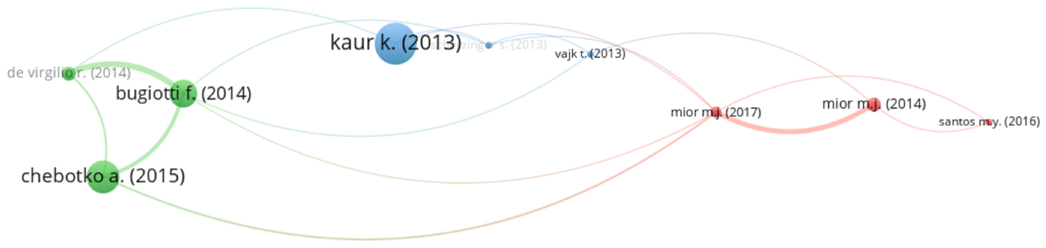

Fig. 6. Network map of Bibliographic Coupling.

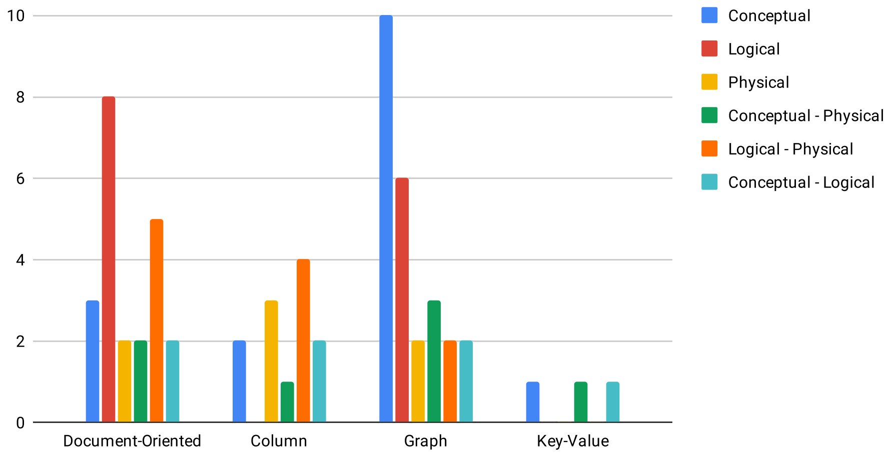

Fig. 7. Levels of representation by categories of NoSQL databases.

focuses on the physical level of representation and the logical-physical levels of representation. No work with three levels of representation was found, and a single logical level was not considered in this category of NoSQL databases. In graph databases, it is notable that the works focus on modeling graph databases using one level of representation; in this case, the conceptual level of representation. The second level that was used most frequently in this category was the logical. Finally, the key-value NoSQL databases are the least explored by researchers. the only three works found were in conceptual, conceptual-logical, and conceptual-physical levels of representations.

Analyzing Figure 7, we can also see the levels of representation applied through the categories of NoSQL databases. For the document-oriented NoSQL databases, the most used level of representation was the logical level, for column NoSQL databases the physical level was the most used, the most used level of representation in graph and key-value NoSQL databases was the conceptual level. Finally, overall, we can see that most of the efforts were applied to the logical and conceptual levels.

5.3 Models Used at Each Level of Representation for Modeling NoSQL Databases

To answer RQ2, attributes were selected from primary studies in accordance with Table 3. The models used were classified according to the NoSQL database type and the levels of representation (conceptual, logical, and physical). Table 8 shows the different models used, along with the three levels of representation in the four categories of NoSQL databases.

From Table 8, we can see that, in general, the models widely used in the process of modeling traditional databases, ER, EER, and UML, were used in the first level of representation for NoSQL data modeling as a conceptual model. However, new notations, such as new cardinalities notations [33],

Table 8. Models and Formats Used in Document-Oriented, Graph, Key-Value and Column NoSQL Databases by Levels of Representation

	Conceptual	Logical	Physical
Document-Oriented	New notation based on ER and UML, UML, OWL, FCA	JSON, XMI, New notation based on UML, Relational Model	JSON, Generic Model, MongoDB model
Graphs	EER, UML, Generic Model, RDF O-ER, New notation based on ER, ER, OWL	JSON, Generic Model, XMI, New notation based on ER, EER, and UML	JSON, XML, Neo4j model, Generic Model, OrientDB model
Key-Value	New notation based on ER, OWL, UML	JSON	Oracle NoSQL model
column	UML, New notation based on ER, ER, OWL	JSON, new notation based on UML, Generic Model, XML, Relational Model, XMI	JSON, Generic Model, XML Cassandra model, Hbase model Amazon DynamoDB model

based on ER and UML, were proposed according to the new features such as, the number of records by entities, CRUD operations, or system requirements (availability, consistency, or scalability) of specific NoSQL databases. These new notations, in some cases, mean the creation of new graphics or the adaption of the model to represent specific new features of a NoSQL database. The W3C **Web Ontology Language (OWL)** is a semantic web language designed to represent rich and complex knowledge about objects and their relationships. Because of this feature, OWL was used to model in the conceptual level of representation in the four NoSQL databases. In the primary studies selected, the **RDF (Resource Description Framework)** standard is used to establish a conceptual model for graph databases [18]. **FCA (Formal Concept Analysis)** is a way of deriving the formal ontology of a group of objects and their properties. FCA is used to define a model for document-oriented NoSQL databases [64].

Although UML, ER, and EER were defined for the conceptual model, some authors also used them at the logical level of representation, adding new features and properties to generate new notations based on ER, UML, and EER. Some examples of these adaptations are shown in [29], [43], and [58]. ER and EER were adapted at this level only for graph databases because of the easy representation of graph properties, while the UML diagram class was adapted for document-oriented and column databases. New schemas were defined at this level also, based on formats such as **JSON (JavaScript Object Notation)**, **XMI (XML Metadata Interchange)**, and **CQL (Cassandra Query Language)** to bring the data model closer to a specific NoSQL database.

Finally, in the last level of representation (physical), the specific NoSQL database to use is known. Therefore, to define the schema generally, each NoSQL database offers its schema model according to the specific features used in the NoSQL database selected. For example, if the NoSQL database selected is MongoDB, this tool offers a MongoDB schema to define the schema in the last level of representation; similarly, we have a Neo4j schema or a Cassandra schema, and so on. However, in many cases, JSON and **XML (extensible markup language)** formats are used as a basis for modeling data in this level of representation. Additionally, a more detailed table showing the academic works and models used by the level of representation can be seen in Table 9.

5.4 Context where the Modeling Process Occurs

We extracted the context where the process of data modeling occurred in NoSQL databases. After analyzing the academic works selected, we classified the contexts as *guidelines, process transform, query oriented, schema generation, evaluation, ontology, benchmark*, and *migration*. Table 10 shows in detail the academic works selected in our study, classified by the context used to model NoSQL

Table 9. Academic Works Detailed by Models Used on Levels of Representation

Academic work	NoSQL Type	Conceptual	Logical	Physical
Imam et al. [34]	document-oriented	—	—	JSON
Akintoye et al. [6]	document-oriented, graph	—	—	JSON
Martins de Sousa and del Val Cura [43]	graph	new based on ER	new based on ER	—
Vágner [68]	graph	EER	—	Neo4j model
Abdelhedi et al. [2]	column	—	—	Cassandra and Hbase models
Nogueira et al. [46]	document-oriented	—	—	JSON
Hamouda and Zainol [29]	document-oriented	—	new based on UML	—
Abdelhedi et al. [5]	document-oriented, graph, column	—	generic model	generic model
Imam et al. [36]	document-oriented	—	new based on UML	—
Suárez-Otero et al. [61]	column	ERD	new based on UML	—
Van Erven et al. [63]	graph	UML	—	—
Angles [8]	graph	generic model	—	—
Chiş-Raţiu and Buchmann [18]	graph	RDF	—	—
Villa et al. [67]	graph	O-ER	—	—
Imam et al. [33]	document-oriented	new based on UML	—	—
Roy-Hubara et al. [52]	graph	UML	—	—
Zhang [70]	graph	new notation	—	—
Mior et al. [45]	column	UML	—	Cassandra model
Banerjee and Sarkar [12]	document-oriented, graph, column, key-value	new based on ER	JSON	—
Shin et al. [58]	document-oriented	UML	new based on UML	—
Chillón et al. [17]	—	UML	—	—
Orel et al. [47]	graph	—	—	Neo4j model
Pokorný [49]	graph	ER	—	—
Bermbach et al. [13]	column	ER	—	—
Lima and Mello [42]	document-oriented	—	new based on UML	—
Banerjee and Sarkar [11]	document-oriented, graph, column, key-value	OWL	—	—
Abdelhedi et al. [3]	column	—	XML	—
Varga et al. [64]	document-oriented	FCA	—	—
Zhao et al. [72]	graph	new notation	—	—
De Lima and Dos Santos Mello [23]	document-oriented	—	new based on UML	—
Vera et al. [66]	document-oriented	new based on UML	—	—
Li et al. [40]	—	—	new based on UML	JSON
Mior [44]	column	—	—	Cassandra and Hbase models
Li et al. [41]	column	—	—	XML
Sedlmeier and Gogolla [57]	graph	—	new based on UML	—
Yoo et al. [69]	graph	new notation	—	—
Zhao et al. [71]	document-oriented, graph	—	relational model	—
Kaur and Rani [37]	document-oriented	—	new based on UML	—
Vajk et al. [62]	column	—	—	Amazon DynamoDB model
Schram and Anderson [56]	column	—	OWN	—
Santos and Costa [55]	column	—	relational model	—
Hewasinghage et al. [32]	document-oriented	—	HeRM model	—
Santisteban and Ticona-Herrera [54]	graph	—	new based on ER	—
De Virgilio et al. [24]	graph	O-ER	—	—
Daniel et al. [20]	graph	—	—	Neo4j or OrientDB models
Abdelhedi et al. [4]	document-oriented, graph column	—	UML	MongoDB, Neo4j, and Cassandra models
Abdelhedi et al. [1]	document-oriented, graph column	—	XMI	MongoDB, Neo4j, and Cassandra models
Imam et al. [35]	document-oriented	—	—	MongoDB models
Shoval [59]	graph	—	UML	—
Roy-Hubara et al. [53]	graph	—	UML	—
Reniers et al. [51]	document-oriented	—	new notation	—
de la Vega et al. [22]	document-oriented, column	—	UML	JSON
Varga et al. [65]	graph	ERD	—	XML
Bugiotti et al. [15]	key-value	UML	—	Oracle NoSQL model

databases. Most of the works proposed data modeling in the context of guidelines and process transformation.

The purpose of the *benchmark* context is to evaluate the performance of a data model based on query response time. In the *evaluation* context, a new proposal of data modeling for graph

Table 10. List of Primary Studies Selected, Classified by the Contexts where the Process of Data Modeling Occurred in NoSQL Databases

Context	Documents
Benchmark	[46]
Evaluation	[53]
Guidelines	[6], [8], [11], [13], [15], [17], [18], [24], [32], [33], [36], [37], [49], [54], [58], [61], [63], [64], [65], [66], [68], [71], [72]
Migration	[29]
Ontology	[11]
Process Transform	[1], [2], [3], [4], [5], [12], [20], [22], [23], [41], [42], [43], [44], [47], [51], [52], [55], [56], [59], [62], [67], [69], [70]
Query Oriented	[40]
Schema Generation	[34], [35], [45]

databases [53] based on ER was evaluated; the evaluation of this method is by a controlled experiment. A *guideline* context is the process of generating a data model from scratch, it could start on a conceptual or logical level and go on to the physical level. In the *migration* context, a migration process from relational databases to a specific NoSQL database is proposed (this includes data migration), and for that it is necessary to define a data model in a NoSQL database. The only academic work found on the process of *migration* (see Table 10) started in the conceptual level of representation in a relational database (ER diagram) and went on to the logical level of representation in a document-oriented NoSQL database. In the context of *ontologies*, a driven meta-model is proposed to conceptualize data representation independently of any NoSQL database, proposing a common conceptual level abstraction for NoSQL databases to generate logical or physical schemas. For this purpose, a formal vocabulary is implemented using the *Protégé* tool based on the OWL format. Unlike the context of migration (migration of model and data), a *process transformation* in the context of relational and NoSQL databases, transforms a model expressed in most cases in ER or UML, to a model in a NoSQL database (see Table 10). A model is required as input and, as output, a model will be obtained in the conceptual, logical, or physical level in NoSQL databases. In some cases, the process transformation occurs within NoSQL databases, from conceptual to logical or from logical to physical level, for example. The academic work identified in the context of *query-oriented* generates a data model and data schema based on the stored data and query requirements. The *schema generation* context generates schemas in the physical model. Three academic works were found in this context that generate schema for document-oriented and column NoSQL databases (see Table 10). For this, a list of inputs is required, like system requirement (availability, consistency, or scalability), CRUD operations, entities, and a number of records.

A distribution of NoSQL databases types along with the contexts where the models were used is shown in Figure 8. It is immediately clear that the guidelines and process transform contexts were the most used. In guidelines context, document-oriented and graph NoSQL databases were the most analysed, while in process transform graphs, column and document-oriented databases were the most analyzed.

6 DISCUSSION OF RESULTS

In this section, we will discuss the results obtained in Section 5. For this, general results are shown and analyzed first, then each research question is analyzed.

As shown in Figure 3, the research in this area started with only a few academic works in 2012, and from there, many proposals appeared over time until the first semester of 2019. Research into

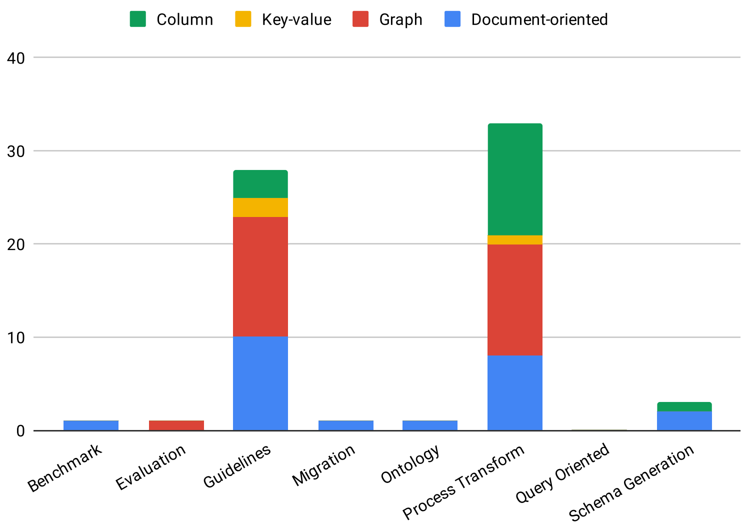

Fig. 8. Contexts of data modeling by NoSQL databases types.

data modeling for NoSQL continues to increase. Despite an exhaustive search for data modeling for NoSQL databases in journals, it was found that more academic works have been published in conferences than in journals. One of the most important conferences on data modeling is the *"International Conference on Conceptual Modelling - ER"*, while one of the most important journals for data modeling is "IEEE Transactions on Knowledge and Data Engineering". Geographically, most of the research and researchers working in data modeling of NoSQL databases are located in Europe and Asia (Figure 4), France being the most productive country in researching this area. According to Tables 6 and 7, France works in collaboration with private companies, research institutions, and universities. An analysis of co-citation (Figure 5) and bibliographic coupling (Figure 6) was carried out to identify the most influential authors in data modeling for NoSQL databases.

6.1 Levels of Representation by Methodologies for Modeling NoSQL Databases

Three levels of representation for data modeling were considered to analyze the works, conceptual, logical and physical. These three levels are widely used in relational databases but their use depends on the methodology used in NoSQL databases. When the methodology applied is the guideline, the conceptual model in most cases was applied with the aim of generalizing the model so that it could be applied to any NoSQL database [11] or for better representation of the problem to be modeled, as in [33], for example. For this reason, no academic works were found which applied to the physical model independently. On the other hand, some academic works preferred to start the model directly in the logical model. From our point of view, it is better if the problem to be modeled is identified adequately along with the three levels of representation. To understand a real problem to be modeled, entities, attributes, and relationships should be defined in a conceptual model. Additionally, features such as system requirements (availability, consistency, or scalability), number of records by entities, and CRUD operations should be identified. In the logical level, the category of NoSQL to be used, and how to implement the database should be defined regardless of the NoSQL system, for example, if the document-oriented category is selected, embedded and/or reference relationships should be defined between entities; or primary key and partition key should be defined if column category is selected. On the physical level, and according to the specific NoSQL system selected, data storage on computers is defined using the specific tools and formats of the NoSQL system.

In this way, if a problem is modeled at an initial level of representation as a conceptual model, it will be easier to get a specific schema for any NoSQL database. When you model at the logical or physical representation levels, more specific details should be considered for generating the schema, such as those identified in the conceptual model.

6.2 Models Used at Each Level of Representation for Modeling NoSQL Databases

Table 8 shows the different models used in our primary studies selected in each level of representation for four categories of the NoSQL databases. According to Shin et al. [58], the models that can be used at the conceptual level are ER, UML, **ORM (Object role Modeling)**, and **FCO-IM (Fully Communication Oriented Information Modeling).** According to our mapping study, additional models were identified, such as OWL, FCA, RDF, OER, and new notations based on ER and UML. In the logical level of representation, Shin et al. [58] consider as logical models the process to develop a logical schema that describes data structured and managed in NoSQL databases. It is dependent on the category used in the NoSQL database but independent of the NoSQL system (software). Additionally these models can make use of more specific formats in every NoSQL database identified by our mapping study, such as JSON, and XML, as well as relational models, ER, EER, UML and new notation based on UML and ER. In the last level of representation (physical representation) Shin et al. [58] consider as models, physical schemas generated by a specific NoSQL system using formats such as JSON, XML, or generic models. Each NoSQL system has specific features to be applied in the process of generating the schema to improve availability and performance. Some NoSQL systems have a specific language to define schemas at the physical level, such as Cassandra and CQL.

The key-value NoSQL database is the least explored, based on data modeling according to Figure 7. However, modeling is possible at the first level of representation using the models presented in Table 8. In the logical level of representation,the only work that presented a proposed for modeling was [12] but is not consolidated yet. In the physical level of the representation, no model is identified in our mapping study because the database is implemented using the specific data structure of a target data store, as shown in [15].

On the other hand, in document-oriented NoSQL databases, many proposals were made along the three levels of representations; therefore, it is possible to model in the three levels. ER, UML, and new notations based on ER and UML were used to generate schemas. Schemas based on JSON and XMI formats were proposed also to generate schemas along the three levels of representation. In the physical level, the schema generated will depend on the technology selected and the format used to implement the NoSQL database. This is because of the different ways of representing data to store on computers.

Similarly, in column family NoSQL databases, the conceptual level is used to understand and represent a problem. For this purpose, UML, ER, OWL and new notations based on ER were used to generate the conceptual schema. In the logical level, UML, the generic model, and relational model were used to generate the logical schema. Additionally, CQL language, JSON, XML and XMI formats were used to generate logical and physical schemas.

Finally, in graph databases, the most used model in the conceptual level is ER which can be used independently of the tool selected for the next levels. At the logical level, it is possible to use UML and ER to represent a more detailed schema regardless of any specific tool, but at the physical level, the schema will depend on the tools selected, for example, Neo4j, infinity Graph, Orient DB, and so on.

As one moves from a high-level model to a low-level one, the modeling process becomes more difficult due to the heterogeneity of the NoSQL databases and it will depend on the tool selected for modeling. In addition, operational factors must be taken into account, such as atomicity, sharding, indexes, number of records and so on. The number of levels of representation used to model NoSQL databases will also depend on the methodology selected, in the case of guideline, for example, it could be recommended to model with the three levels but it will be different if a process transformation methodology is used.

6.3 Context where the Modeling Process Occurs

Researchers are focusing on two main contexts of modeling NoSQL databases, both guideline for modeling a NoSQL database from scratch, and process transformation, for transforming relational database models or a list of requirements into NoSQL data models. Therefore, there are two strong lines of enquiry, one is trying to understand the problem to be modeled from scratch by considering new features of specific NoSQL databases and starting modeling in conceptual models. The other is either transforming models from relational databases to the logical and physical level of representation (process transform context), or trying to obtain models for the physical level of representation (schema generation context) from a list of requirements.

For our mapping study, we considered the methods identified in [9] as contexts where the process of modeling had occurred. Therefore, new contexts in our mapping study were identified as we can see in Table 10, they are benchmark, evaluation, migration and schema generation. Additionally, more academic works were found in addition to the different contexts of modeling already identified in [9].

6.4 Data Modeling Efforts in the Industrial Area

Due to the strict mapping protocol, when we applied our systematic mapping we only considered academic works. However, contributions from the industrial area have also been made, mainly in the development of software for data modeling, for example, Hackolade and KDM [16].

Hackolade is multi-platform visualization software for the data modeling of NoSQL databases. It allows graphical schema design and physical data modeling, forward and reverse engineering, schema model comparison, data model documentation, JSON nested objects, polymorphism, JSON schema editing and persistence [28]. Hackolade is based on entity-relationship, Information Engineering notation and JSON schema standards. Currently, the databases and target support are MongoDB, Couchbase, CouchDB, Elasticsearch, Google Real time Firebase, Google Cloud Firestore in document-oriented databases; Neo4j and TinkerPop in graph databases; DynamoDB in key-value databases and Apache Cassandra, Apache HBase and ScyllaDB in column databases.

The Kashliev Data Modeler (KDM) is a Big Data modeling tool for Apache Cassandra [38]. It follows the Chetboko data modeling methodology for Cassandra and ensures logical schema. It applies a query-driven approach, different from Hackolade, KDM offers the entire cycle starting with a conceptual model and access patterns, in the first level of representation, and ends with a physical data model. KDM automates time-consuming data modeling tasks, such as conceptual to logical mapping, logical to physical mapping, physical optimization, and CQL script generation.

6.5 Additional Results

As additional results, we found that the most used tools in the academic works selected were as shown in Figure 9. In the document-oriented NoSQL databases, one of the most used tools is MongoDB, one of the fastest-growing and most popular NoSQL databases with documentation and the support of a large community. For graph NoSQL databases, Neo4j is one of the most used tools, Neo4j is a high performance read and write scalability tool and, like MongoDB, has documentation and the support of a large community. For column NoSQL databases, Cassandra and HBase were the most used tools in the primary studies. Finally, for key-value NoSQL databases, Redis is one of the most used tools in the academic works selected.

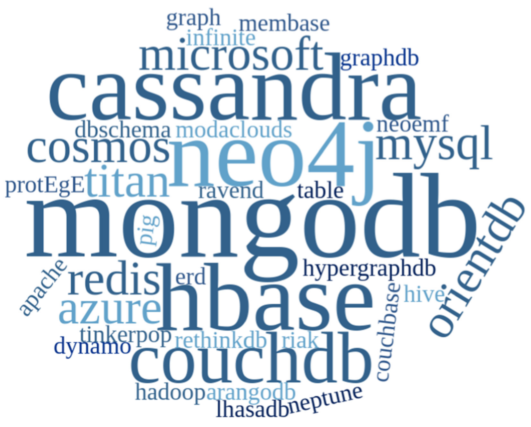

Fig. 9. Tools used in primary studies selected.

7 THREATS TO VALIDITY

This section addresses threats to validity that arise in the process of the research and actions to minimize those risks. There are four categories: construct validity, internal validity, external validity, and conclusions validity.

Construct Validity. A common threat in systematic reviews is that they do not ensure the inclusion of all relevant academic works in the field. This risk might arise for different reasons, such as the keywords defined in titles and abstracts of academic works do not agree with the keywords defined in 4.1.3 or contributions in the field of data modeling of NoSQL databases are not indexed in the selected data sources. Accordingly, to minimize the construct validity threats, the "NoSQL database" and "data model" keywords were established to determine a reliable search string. Likewise, these initial keywords were combined with "key-value", "document-oriented", "column" and "graph" to cover the largest number of academic works. Additionally, two reputable and well-known data sources were chosen: ISI Web of Science and Scopus. Although only two databases were used, the two data sources maximize the number of candidates, since both data sources index most the of academic works in the most relevant digital libraries, such as IEEE Xplore, ACM Digital Library, SpringerLink, and ScienceDirect. We included the snowballing technique, ensuring the inclusion of context articles that have had the biggest impact on our mapping study.

Internal Validity. Due to a huge number of definitions for the same concept, as well as the absence of clear descriptions or appropriate objectives and results in some academic works, there is a clear risk of not covering relevant studies or including unrelated academic works. Such limitations make the application of the exclusion and inclusion criteria difficult, as well as the extraction of information, objectively and impartially. To minimize mistakes caused by subjective analyses, we had several meetings to discuss conflicts and define a proper treatment. For example, when there were doubts about whether or not to include an article (IC-1), an initial meeting between two authors was held, and the final decision was made with the advisor in another meeting.

External Validity. We consider that industrial, scientific, and academic communities in the data modeling of NoSQL databases can benefit from this mapping study.

Reliability Validity. To maximize reliability, we defined our systematic mapping method, including the data source, search string for data sources and systematic mapping implementation, taking into account that the method must be replicable by any researcher in an objective way. A source of reliability threat is related to the first exclusion criteria applied according to our selection process (Figure 2), because the EC-2, EC-5 and EC-6 criteria of exclusion were applied using the filtering tools of Scopus and Web of Science. Another source of reliability threat concerns the number of academic works returned in the period of 2008 to the first semester of 2019, our study was done in early 2019 (March, April). Therefore, some academic works from 2019 were probably not recovered in our search.

8 CONCLUSIONS

The systematic study performed allows us to map important information and characteristics of data modeling of NoSQL databases. Although some academic works have been published about data models for NoSQL databases, no previous systematic reviews had been performed to analyze academic works published in this area.

As a result of this mapping, 54 primary studies about data modeling for NoSQL databases were selected from 2008 to the first semester of 2019, including only full academic works published in conferences and journals. The number of documents found shows that this topic is still growing slowly but constantly. Our mapping was guided by three research questions to analyze these 54 primary studies. Levels of representation by methodologies for modeling NoSQL databases, models used at each level of representation for modeling NoSQL databases and contexts where the process of modeling occurred.

One of the main goals in this mapping review was to ascertain the current trend in the modeling of the NoSQL databases. Our main conclusion in this regard is that there is not yet a consensus on the adoption of a standard to model NoSQL databases in a general and specific (document-oriented, graphs, column and key-value) way. There is no single way or criterion for choosing the most suitable conceptual model for each NoSQL database, or, similarly, the most suitable logical and physical models for each NoSQL database. There is no consensus on the use of terms like “model” or “method” and “phases of representations” or “level of representation”. New features to model NoSQL databases are not being considered, such as sharding, system requirements (availability, consistency or scalability), number of records by entities approximately or CRUD operations. These features should be considered for generating models.

Models used in relational databases, like UML and ER and the three levels of representations (conceptual, logical and physical), are being adapted to model NoSQL databases mainly in the first level of representation, but the number of combinations between levels of representation and models to model a NoSQL database are generating many new proposals without considering the real importance of modeling a database.

As for future works, it is recommended to work in a new way to get requirements that capture new features of NoSQL databases. Capturing these new features can help modelers recognize the correct needs for database modeling in NoSQL databases. Alternatively, since many proposals were made about data modeling for NoSQL databases, finding the best one in a specific category of NoSQL database could be interesting. For this, comparative works should be done according to some metrics.

REFERENCES

[1] F. Abdelhedi, A. Ait Brahim, F. Atigui, and G. Zurfluh. 2017. MDA-based approach for NoSQL databases modelling. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10440 LNCS (2017), 88–102. https://doi.org/10.1007/978-3-319-64283-3_7

[2] F. Abdelhedi, A. Ait Brahim, and G. Zurfluh. 2018. Formalizing the mapping of UML conceptual schemas to column-oriented databases. *International Journal of Data Warehousing and Mining* 14, 3 (2018), 44–68. https://doi.org/10.4018/IJDWM.2018070103

[3] F. Abdelhedi, A. A. Brahim, F. Atigui, and G. Zurfluh. 2016. Big data and knowledge management: How to implement conceptual models in NoSQL systems'. *IC3K 2016 - Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* 3 (2016), 235–240.

[4] F. Abdelhedi, A. A. Brahim, F. Atigui, and G. Zurfluh. 2017. Logical unified modeling for NoSQL databases. *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems* 1 (2017), 249–256.

[5] F. Abdelhedi, A. A. Brahim, F. Atigui, and G. Zurfluh. 2018. UMLtoNoSQL: Automatic transformation of conceptual schema to NoSQL databases. *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA* 2017-October (2018), 272–279. https://doi.org/10.1109/AICCSA.2017.76

[6] S. B. Akintoye, A. B. Bagula, O. E. Isafiade, Y. Djemaiel, and N. Boudriga. 2019. Data model for cloud computing environment. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST* 275 (2019), 199–215. https://doi.org/10.1007/978-3-030-16042-5_19

[7] Jacky Akoka, Isabelle Comyn-Wattiau, and Nabil Laoufi. 2017. Research on Big Data–a systematic mapping study. *Computer Standards & Interfaces* 54 (2017), 105–115.

[8] R. Angles. 2018. The property graph database model. *CEUR Workshop Proceedings* 2100 (2018).

[9] Chaimae Asaad and Karim Baïna. 2018. NoSQL Databases–Seek for a Design Methodology. In *International Conference on Model and Data Engineering*. Springer, 25–40.

[10] Paolo Atzeni. 2016. Data Modelling in the NoSQL world: A contradiction?. In *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*. ACM, 1–4.

[11] S. Banerjee and A. Sarkar. 2016. Ontology Driven Meta-Modeling for NoSQL Databases: A Conceptual Perspective. *International Journal of Software Engineering and its Applications* 10, 12 (2016), 41–64. https://doi.org/10.14257/ijseia.2016.10.12.05

[12] S. Banerjee and A. Sarkar. 2017. Logical level design of NoSQL databases. *IEEE Region 10 Annual International Conference, Proceedings/TENCON* (2017), 2360–2365. https://doi.org/10.1109/TENCON.2016.7848452

[13] D. Bermbach, S. Müller, J. Eberhardt, and S. Tai. 2016. Informed Schema Design for Column Store-Based Database Services. *Proceedings - 2015 IEEE 8th International Conference on Service-Oriented Computing and Applications, SOCA 2015* (2016), 163–172. https://doi.org/10.1109/SOCA.2015.29

[14] Eric Brewer. 2010. A certain freedom: thoughts on the CAP theorem. In *Proceedings of the 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*. 335.

[15] Francesca Bugiotti, Luca Cabibbo, Paolo Atzeni, and Riccardo Torlone. 2014. Database design for NoSQL systems. In *International Conference on Conceptual Modeling*. Springer. 223–231.

[16] A. Chebotko, A. Kashlev, and S. Lu. 2015. A Big Data Modeling Methodology for Apache Cassandra. *Proceedings - 2015 IEEE International Congress on Big Data, BigData Congress* 2015 (2015), 238–245.

[17] A. H. Chillón, S. F. Morales, D. S. Ruiz, and J. G. Molina. 2017. Exploring the visualization of schemas for aggregate-oriented nosql databases? *CEUR Workshop Proceedings* 1979 (2017), 72–85.

[18] A. Chiş-Raţiu and R. A. Buchmann. 2018. Design and implementation of a diagrammatic tool for creating RDF graphs. *CEUR Workshop Proceedings* 2238 (2018), 37–48.

[19] Alejandro Corbellini, Cristian Mateos, Alejandro Zunino, Daniela Godoy, and Silvia Schiaffino. 2017. Persisting big-data: The NoSQL landscape. *Information Systems* 63 (2017), 1–23.

[20] G. Daniel, G. Sunyé, and J. Cabot. 2016. UMLtographDB: Mapping conceptual schemas to graph databases. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9974 LNCS (2016), 430–444. https://doi.org/10.1007/978-3-319-46397-1_33

[21] Ali Davoudian, Liu Chen, and Mengchi Liu. 2018. A survey on NoSQL stores. *ACM Computing Surveys (CSUR)* 51, 2 (2018), 40.

[22] A. de la Vega, D. García-Saiz, C. Blanco, M. Zorrilla, and P. Sánchez. 2018. Mortadelo: A model-driven framework for NoSQL database design. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11163 LNCS (2018), 41–57. https://doi.org/10.1007/978-3-030-00856-7_3

[23] C. De Lima and R. Dos Santos Mello. 2015. A workload-driven logical design approach for NoSQL document databases. In *Proceedings of 17th International Conference on Information Integration and Web-Based Applications and Services (iiWAS'15)*. https://doi.org/10.1145/2837185.2837218

[24] R. De Virgilio, A. Maccioni, and R. Torlone. 2014. Model-driven design of graph databases. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8824 (2014), 172–185.

[25] Miguel Diogo, Bruno Cabral, and Jorge Bernardino. 2019. Consistency Models of NoSQL Databases. *Future Internet* 11, 2 (2019), 43.

[26] David Gil and Il-Yeol Song. 2016. Modeling and management of big data: challenges and opportunities.

[27] Paola Gómez, Rubby Casallas, and Claudia Roncancio. 2016. Data schema does matter, even in NoSQL systems!. In *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*. IEEE, 1–6.

[28] Hackolade. 2020. Data modeling tool for NoSQL databases. https://hackolade.com/. (Accessed on 10/10/2020).

[29] S. Hamouda and Z. Zainol. 2018. Document-Oriented Data Schema for Relational Database Migration to NoSQL. *Proceedings - 2017 International Conference on Big Data Innovations and Applications, Innovate-Data 2017* 2018-January (2018), 43–50. https://doi.org/10.1109/Innovate-Data.2017.13

[30] Jing Han, E Haihong, Guan Le, and Jian Du. 2011. Survey on NoSQL database. In *2011 6th international conference on pervasive computing and applications*. IEEE, 363–366.

[31] Robin Hecht and Stefan Jablonski. 2011. NoSQL evaluation: A use case oriented survey. In *2011 International Conference on Cloud and Service Computing*. IEEE, 336–341.

[32] M. Hewasinghage, N.B. Seghouani, and F. Bugiotti. 2018. Modeling strategies for storing data in distributed heterogeneous NoSQL databases. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11157 LNCS (2018), 488–496. https://doi.org/10.1007/978-3-030-00847-5_35

[33] A. A. Imam, S. Basri, R. Ahmad, N. Aziz, and M. T. Gonzalez-Aparicio. 2017. New cardinality notations and styles for modeling NoSQL document-store databases. *IEEE Region 10 Annual International Conference, Proceedings/TENCON* 2017-December (2017), 2765–2770. https://doi.org/10.1109/TENCON.2017.8228332

[34] A. A. Imam, S. Basri, R. Ahmad, and María Teresa González Aparicio. 2019. Schema proposition model for NoSQL applications. *Recent Trends in Data Science and Soft Computing* (2019).

[35] A. A. Imam, S. Basri, R. Ahmad, J. Watada, and M. T. González-Aparicio. 2018. Automatic schema suggestion model for NoSQL document-stores databases. *Journal of Big Data* 5, 1 (2018). https://doi.org/10.1186/s40537-018-0156-1

[36] Abdullahi Abubakar Imam, Shuib Basri, Rohiza Ahmad, Junzo Watada, María Teresa González Aparicio, and Malek Ahmad Almomani. 2018. Data modeling guidelines for NoSQL document-store databases. *International Journal of Advanced Computer Science and Applications*, 9 (2018).

[37] K. Kaur and R. Rani. 2013. Modeling and querying data in NoSQL databases. *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013* (2013), 1–7. https://doi.org/10.1109/BigData.2013.6691765

[38] KDM. 2020. The Kashliev Data Modeler. https://www.datafluent.org/. [Online; accessed 10/10/2020].

[39] Barbara Kitchenham, Rialette Pretorius, David Budgen, O. Pearl Brereton, Mark Turner, Mahmood Niazi, and Stephen Linkman. 2010. Systematic literature reviews in software engineering–a tertiary study. *Information and software technology* 52, 8 (2010), 792–805.

[40] X. Li, Z. Ma, and H. Chen. 2014. QODM: A query-oriented data modeling approach for NoSQL databases. *Proceedings - 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications, WARTIA 2014* (2014), 338–345. https://doi.org/10.1109/WARTIA.2014.6976265

[41] Yan Li, Ping Gu, and Chao Zhang. 2014. Transforming UML class diagrams into HBase based on meta-model. In *2014 International Conference on Information Science, Electronics and Electrical Engineering*, Vol. 2. IEEE, 720–724.

[42] C. Lima and R. S. Mello. 2016. On proposing and evaluating a NoSQL document database logical approach. *International Journal of Web Information Systems* 12, 4 (2016), 398–417. https://doi.org/10.1108/IJWIS-04-2016-0018

[43] V. Martins de Sousa and L. M. del Val Cura. 2018. Logical design of graph databases from an entity-relationship conceptual model. *ACM International Conference Proceeding Series* (2018), 183–189. https://doi.org/10.1145/3282373.3282375

[44] M. J. Mior. 2014. Automated schema design for NoSQL databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2014), 41–45. https://doi.org/10.1145/2602622.2602624

[45] Michael Joseph Mior, Kenneth Salem, Ashraf Aboulnaga, and Rui Liu. 2017. NoSE: Schema design for NoSQL applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 10 (2017), 2275–2289.

[46] I. D. Nogueira, M. Romdhane, and J. Darmont. 2018. Modeling data lake metadata with a data vault. *ACM International Conference Proceeding Series* (2018), 253–261. https://doi.org/10.1145/3216122.3216130

[47] O. Orel, S. Zakošek, and M. Baranović. 2017. Property oriented relational-to-graph database conversion [Konverzija relacijskih u grafovske baze podataka orijentirana na svojstva]. *Automatika* 57, 3 (2017), 836–845. https://doi.org/10.7305/automatika.2017.02.1581

[48] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology* 64 (2015), 1–18.

[49] J. Pokorný. 2016. Conceptual and database modelling of graph databases. *ACM International Conference Proceeding Series* 11-13-July-2016 (2016), 370–377. https://doi.org/10.1145/2938503.2938547

[50] Debora G. Reis, Fabio S. Gasparoni, Maristela Holanda, Marcio Victorino, Marcelo Ladeira, and Edward O Ribeiro. 2018. An evaluation of data model for NoSQL document-based databases. In *World Conference on Information Systems and Technologies*. Springer, 616–625.

[51] V. Reniers, D. Van Landuyt, A. Rafique, and W. Joosen. 2018. Schema design support for semi-structured data: Finding the sweet spot between NF and De-NF. *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017* 2018-January (2018), 2921–2930. https://doi.org/10.1109/BigData.2017.8258261

[52] N. Roy-Hubara, L. Rokach, B. Shapira, and P. Shoval. 2017. Modeling Graph Database Schema. *IT Professional* 19, 6 (2017), 34–43. https://doi.org/10.1109/MITP.2017.4241458

[53] N. Roy-Hubara, L. Rokach, B. Shapira, and P. Shoval. 2018. Evaluation of a design method for graph database. *Lecture Notes in Business Information Processing* 318 (2018), 291–303. https://doi.org/10.1007/978-3-319-91704-7_19

[54] J. Santisteban and R. Ticona-Herrera. 2018. Modeling a persistent graph. *Proceedings of a Special Session - 16th Mexican International Conference on Artificial Intelligence: Advances in Artificial Intelligence, MICAI 2017* (2018), 15–22. https://doi.org/10.1109/MICAI-2017.2017.00011

[55] M. Y. Santos and C. Costa. 2016. Data models in NoSQL databases for big data contexts. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9714 LNCS (2016), 475–485. https://doi.org/10.1007/978-3-319-40973-3_48

[56] A. Schram and K. M. Anderson. 2012. MySQL to NoSQL data modeling challenges in supporting scalability. *SPLASH'12 - Proceedings of the 2012 ACM Conference on Systems, Programming, and Applications: Software for Humanity* (2012), 191–202. https://doi.org/10.1145/2384716.2384773

[57] M. Sedlmeier and M. Gogolla. 2014. Design and prototypical implementation of an integrated graph-based conceptual data model. *Frontiers in Artificial Intelligence and Applications* 272 (2014), 376–395. https://doi.org/10.3233/978-1-61499-472-5-376

[58] K. Shin, C. Hwang, and H. Jung. 2017. NoSQL database design using UML conceptual data model based on peter chen's framework. *International Journal of Applied Engineering Research* 12, 5 (2017), 632–636.

[59] P. Shoval. 2018. A method for modeling a schema for graph databases. *Digital Presentation and Preservation of Cultural and Scientific Heritage* 8 (2018), 99–104.

[60] Graeme Simsion and Graham Witt. 2004. *Data Modeling Essentials*. Elsevier.

[61] P. Suárez-Otero, M. J. Suárez-Cabal, and J. Tuya. 2018. Leveraging conceptual data models for keeping cassandra database integrity. In *Proceedings of the 14th International Conference on Web Information Systems and Technologies (WEBIST'18)*. 398–403.

[62] Tamás Vajk, László Deák, Krisztián Fekete, and Gergely Mezei. 2013. Automatic NoSQL schema development: A case study. In *Artificial Intelligence and Applications*. Actapress, 656–663.

[63] G. Van Erven, W. Silva, R. Carvalho, and M. Holanda. 2018. GRAPHED: A graph description diagram for graph databases. *Advances in Intelligent Systems and Computing* 745 (2018), 1141–1151. https://doi.org/10.1007/978-3-319-77703-0_111

[64] V. Varga, K. T. Jánosi-Rancz, and B. Kálmán. 2016. Conceptual design of document NoSQL database with formal concept analysis. *Acta Polytechnica Hungarica* 13, 2 (2016), 229–248.

[65] V. Varga, C. Săcărea, and A. E. Molnar. 2018. Conceptual Graphs Based Modeling of Semi-structured Data. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10872 LNAI (2018), 167–175. https://doi.org/10.1007/978-3-319-91379-7_13

[66] H. Vera, W. Boaventura, M. Holanda, V. Guimarães, and F. Hondo. 2015. Data modeling for NoSQL document-oriented databases. *CEUR Workshop Proceedings* 1478 (2015), 129–135.

[67] Fernán Villa, Francisco Moreno, and Jaime Guzmán. 2018. An Analysis of a Methodology that Transforms the Entity-Relationship Model into a Conceptual Model for a Graph Database. In *International Conference for Emerging Technologies in Computing*. Springer, 70–83.

[68] A. Vágner. 2018. Store and visualize EeR in Neo4j. *ACM International Conference Proceeding Series* (2018). https://doi.org/10.1145/3284557.3284694

[69] K. M. Yoo, S. Park, and S.-G. Lee. 2014. RDB2Graph: A generic framework for modeling relational databases as graphs. *CEUR Workshop Proceedings* 1312 (2014), 148–151.

[70] Z. J. Zhang. 2017. Graph Databases for Knowledge Management. *IT Professional* 19, 6 (2017), 26–32. https://doi.org/10.1109/MITP.2017.4241463

[71] Gansen Zhao, Weichai Huang, Shunlin Liang, and Yong Tang. 2013. Modeling MongoDB with relational model. In *2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies*. IEEE, 115–121.

[72] M. Zhao, Y. Liu, and P. Zhou. 2016. Towards a systematic approach to graph data modeling: Scenario-based design and experiences. *Proceedings of the International Conference on Software Engineering and Knowledge Engineering*, SEKE 2016-January (2016), 634–637. https://doi.org/10.18293/SEKE2016-119

Received December 2019; revised January 2021; accepted March 2021