big data and cognitive computing

MDPI

Review

Integrating OLAP with NoSQL Databases in Big Data Environments: Systematic Mapping

Diana Martinez-Mosquera [1,*], Rosa Navarrete [1], Sergio Luján-Mora [2], Lorena Recalde [1] and Andres Andrade-Cabrera [1]

[1] Department of Informatics and Computer Science, Escuela Politécnica Nacional, Quito 170525, Ecuador; rosa.navarrete@epn.edu.ec (R.N.); lorena.recalde@epn.edu.ec (L.R.); luis.andrade03@epn.edu.ec (A.A.-C.)
[2] Department of Software and Computing Systems, University of Alicante, 03690 Alicante, Spain; sergio.lujan@ua.es
* Correspondence: diana.martinez@epn.edu.ec

Abstract: The growing importance of data analytics is leading to a shift in data management strategy at many companies, moving away from simple data storage towards adopting Online Analytical Processing (OLAP) query analysis. Concurrently, NoSQL databases are gaining ground as the preferred choice for storing and querying analytical data. This article presents a comprehensive, systematic mapping, aiming to consolidate research efforts related to the integration of OLAP with NoSQL databases in Big Data environments. After identifying 1646 initial research studies from scientific digital repositories, a thorough examination of their content resulted in the acceptance of 22 studies. Utilizing the snowballing technique, an additional three studies were selected, culminating in a final corpus of twenty-five relevant articles. This review addresses the growing importance of leveraging NoSQL databases for OLAP query analysis in response to increasing data analytics demands. By identifying the most commonly used NoSQL databases with OLAP, such as column-oriented and document-oriented, prevalent OLAP modeling methods, such as Relational Online Analytical Processing (ROLAP) and Multidimensional Online Analytical Processing (MOLAP), and suggested models for batch and real-time processing, among other results, this research provides a roadmap for organizations navigating the integration of OLAP with NoSQL. Additionally, exploring computational resource requirements and performance benchmarks facilitates informed decision making and promotes advancements in Big Data analytics. The main findings of this review provide valuable insights and updated information regarding the integration of OLAP cubes with NoSQL databases to benefit future research, industry practitioners, and academia alike. This consolidation of research efforts not only promotes innovative solutions but also promises reduced operational costs compared to traditional database systems.

Keywords: Big Data; NoSQL; OLAP; systematic mapping

Citation: Martinez-Mosquera, D.; Navarrete, R.; Luján-Mora, S.; Recalde, L.; Andrade-Cabrera, A. Integrating OLAP with NoSQL Databases in Big Data Environments: Systematic Mapping. *Big Data Cogn. Comput.* **2024**, *8*, 64. https://doi.org/10.3390/bdcc8060064

Academic Editor: Domenico Ursino

Received: 20 March 2024
Revised: 16 April 2024
Accepted: 22 April 2024
Published: 5 June 2024

1. Introduction

Over the past several years, the exponential growth of data has prompted research into new paradigms and tools [1]. With data volumes escalating and NoSQL technology maturing, traditional relational data warehouses are undergoing a profound transformation, ceding ground to NoSQL-based counterparts equipped with Online Analytical Processing (OLAP) capabilities [2]. This transformation marks a departure from the conventional paradigm in which data warehouses adhered to distributed design principles like Atomicity, Consistency, Isolation, and Durability (ACID) to ensure integrity, scalability, and availability. Instead, NoSQL-based data warehouses embrace an entirely different philosophy.

The exponential increase in data volume, variety, and velocity has triggered the development of innovative data technologies in modern information systems. This evolution is driven by the need to address the challenges presented via massive data sets and the

dynamic nature of information in today's environment [3]. As organizations increasingly adopt advanced data storage and processing solutions, there is a growing demand for a deeper understanding of the methodologies, challenges, and advancements in this integration [4]. This study sought to explore these evolving dynamics, adding valuable insights to the ongoing discussion of integrating OLAP and NoSQL in the realm of Big Data analytics.

OLAP involves dealing with fact tables and dimensions, which are fundamental components of data warehousing (DW). Fact tables contain numerical data (facts) that represent business transactions or events, while dimension tables provide context and descriptive information about the data in the fact tables [5].

Integrating OLAP and NoSQL presents a significant challenge. Unlike relational databases, NoSQL databases do not adhere to a rigid, predefined schema, which complicates the implementation of fact tables and dimensions due to unexpected changes in the database structure. Another major obstacle is automating all stages of the Extract, Transform, Load (ETL) process because of the dynamic nature of the schema [5].

To address the increasing demand in this area and the lack of up-to-date systematic mapping on the subject, this study aims to consolidate the most relevant advancements made by different authors in the realm of OLAP and NoSQL. The primary objective is to present scientific evidence on methods proposed for constructing OLAP cubes from NoSQL databases. A notable challenge in this research was to identify studies that offered solutions pertinent to the characterization of Big Data, especially in terms of volume, variety, and velocity [6]. While numerous initiatives focus on deploying OLAP cubes over data warehouses, traditional methods are unsuitable for handling Big Data, presenting a prominent challenge in the research community.

This study collected articles from five digital scientific libraries, resulting in 1649 articles that were individually analyzed, with 25 selected studies that matched the research question. Seven dimensions were raised, focusing on the most used NoSQL databases with OLAP, types of OLAP systems, methods for processing data and building OLAP cubes, and the computational resources required to implement the proposed methods.

The primary objective of this research is to explore the recent advancements in integrating OLAP with various types of NoSQL databases within the context of Big Data environments. The study endeavored to offer a comprehensive roadmap for organizations navigating the integration of OLAP with NoSQL, with a specific focus on addressing the escalating demands of data analytics in modern business environments and understanding the associated operational costs.

This work holds significance as the scientific community grapples with challenges in implementing OLAP on Big Data, such as handling vast data volumes and dealing with complex, multidimensional data models [5]. By consolidating the most relevant contributions, this research facilitates academia and industry in addressing these challenges, and it guides future research efforts.

The remainder of this paper is structured as follows. The Theoretical Background section offers a concise explanation of the key concepts needed to understand the rest of this paper. The Related Work section summarizes previous research efforts in the field. The Materials and Methods section provides comprehensive details of the process followed to conduct the systematic mapping, adhering to the guidelines proposed by Kitchenham for a systematic literature review in software engineering [7]. The Results section highlights the main findings and addresses the research questions one by one. The paper concludes with a summary of the study's findings in the Discussion section, and it outlines possible future research directions in the Conclusion section.

2. Theoretical Background

2.1. Data Warehousing

DW refers to the process of collecting and storing data from various sources in a centralized repository. The data are structured and organized in a way that facilitates reporting,

analysis, and decision making. DW systems are designed to handle large volumes of data and provide historical and current insights into the operations of organizations [8].

2.2. NoSQL Databases

NoSQL databases, also known as "not only SQL", are non-tabular databases that store data differently from relational tables. They offer various types based on their data model, including column-oriented, document-oriented, graph, and key–value databases [6]:

- Column-oriented databases, such as Apache Cassandra and HBase, store data in columns, rather than rows, enabling high-speed read and write operations for analytical queries.
- Document-oriented databases, such as MongoDB, store data in flexible, JSON-like documents, making them suitable for handling semi-structured and unstructured data types.
- Key–value databases store data as key–value pairs, providing fast access to individual data items and efficient caching mechanisms.
- Graph databases, such as Neo4j, store data in graph structures with nodes, edges, and properties, facilitating the representation and analysis of complex relationships between data entities.

These databases provide flexible schemas, and they excel in scaling with large data sets and handling high user loads [9].

2.3. OLAP

OLAP is a decision-oriented, descriptive modeling approach used for data analysis. At the core of OLAP systems are multidimensional cubes, which organize data into multiple dimensions, enabling users to explore and analyze data from various perspectives. These cubes facilitate efficient data aggregation and summarization, making it possible to process and visualize vast amounts of data. OLAP empowers users to perform complex queries, data drill-downs, roll-ups, and pivots effortlessly, providing valuable insights for informed decision making and strategic planning in business intelligence and DW scenarios [10].

An OLAP cube is an abstract representation of a Relational Database Management System (RDBMS) that extends beyond the conventional two dimensions. For instance, it can be expressed as a three-dimensional function, D, denoted as follows:

$D = D1, D2, D3.$

In this context, the dimensions $D1$, $D2$, and $D3$ serve as the axes of the cube, while D represents the data or values stored in the individual cells of the cube. OLAP cubes enable users to perform multidimensional analysis, allowing for a comprehensive view of data from different perspectives and facilitating efficient data aggregation and summarization.

2.3.1. OLAP Systems

In this study, it is important to understand the different types of OLAP systems used for analysis. Commonly mentioned ones include the following:

- Relational OLAP (ROLAP) stores data in relational databases, leveraging SQL for query processing. Popular examples include Oracle OLAP and SAP BW [11].
- Multidimensional OLAP (MOLAP) stores data in pre-aggregated cubes, offering fast query performance. It is suitable for complex analyses with many dimensions and measures [11]. Popular tools are Essbase and SAP.
- Spatial OLAP (SOLAP) extends OLAP to spatial data, allowing users to analyze multidimensional data with a spatial component [12]. Popular SOLAP tools include GeoMondrian and Map4Decision.

This research findings explore additional proposals of OLAP variants within the integration of NoSQL databases, enhancing the understanding of the dynamic landscape of Big Data analytics. Among these proposals, OLAP for columnar databases, OLAP for Hadoop, real-time OLAP, and OLAP for graphs were found.

2.3.2. OLAP Schemas

In the realm of OLAP, diverse schema designs are key to shaping data warehouse architecture. Each schema provides unique benefits, catering to specific analytical requirements in the ever-evolving landscape of DW and business intelligence. Notable among them are the following [13]:

- Star schema: A central fact table is surrounded by dimension tables, forming a star-like structure. The fact table contains quantitative data (measurements), and each dimension table holds descriptive attributes.
- Snowflake schema: An extension of the star schema that normalizes dimension tables, breaking them into multiple related tables. This schema results in a more complex, snowflake-like structure.
- Galaxy schema: An extension of the star schema in which multiple fact tables share dimension tables. This interconnected structure resembles a galaxy with stars (fact tables) and shared planets (dimension tables). This schema presents enhanced analytical capabilities by allowing analysis across multiple fact tables.

2.4. Big Data

Big Data refers to the vast volume of structured, semi-structured, and unstructured data generated at high velocities, and it requires advanced technologies and analytics methods to process and extract meaningful insights. These data are characterized mainly by their volume, velocity, and variety. Big Data technologies include distributed computing frameworks like Hadoop, Spark, or NoSQL databases to process these massive data sets [6].

2.4.1. Map Reduce

MapReduce is a programming model and framework for processing large data sets in parallel. It is a powerful tool used in Big Data analytics to handle massive amounts of information [6].

2.4.2. Big Data Processing

Big Data processing refers to the set of techniques and tools used to analyze and extract information from large data sets. There are two main approaches to Big Data processing: batch and real-time. Batch processing involves processing historical data sets that have already been stored. Real-time processing, meanwhile, involves processing data as they are generated, without the need to store them beforehand [6].

2.5. TPC Benchmarks

TPC Benchmarks, or TPC performance tests, are a set of industry standards for measuring the performance of transaction processing (TP) systems and databases. They are developed and maintained by the Transaction Processing Performance Council (TPC), a non-profit organization founded in 1988 [14].

TPC Benchmarks are based on real-world use cases, and they simulate the workloads found in different types of enterprise applications. The test results are published on the TPC website, and they can be used to compare the performance of different systems.

There are several types of TPC Benchmarks, each of which focuses on a specific type of workload [14]:

- TPC-C is the best-known and most widely used benchmark for measuring the performance of On-Line Transaction Processing (OLTP) systems. It simulates the activity of an electronics wholesaler.
- TPC-H measures the performance of OLAP systems for complex queries on large data sets. It simulates the analysis of sales data for a company.
- TPC-DS is a more recent version of TPC-H that focuses on data analysis from large data warehouses.
- TPC-E measures the performance of e-commerce systems.

TPC Benchmarks are an important tool for evaluating the performance of transaction processing systems and databases. They help companies make informed decisions about hardware and software selection and optimize their systems' performance.

3. Related Work

Various digital scientific repositories were examined to discern the imperative of this study. Regrettably, the exhaustive search yielded no up-to-date systematic mapping studies. However, during the exploration, several analogous works were identified that aimed to consolidate knowledge about OLAP, Big Data, and NoSQL from different perspectives.

Cuzzocrea [5] presents a comprehensive overview of the open issues and future challenges of using OLAP over Big Data. This survey study, conducted in 2015, emphasizes the relevance of the convergence between OLAP and Big Data in both academic and industrial fields. Among the primary research problems identified in the study are the sizes of fact tables, the complexity of building OLAP data cubes, computing resource requirements, and considerations related to quality, usability, and visualization. The study also suggests exploring innovative methodologies to design data cubes, aggregations, high-performance architectures, and applications. As this article is dated from 2015, it is important to provide an up-to-date revision of this work. Additionally, this systematic mapping offers an in-depth examination of NoSQL and OLAP, providing a comprehensive guide to current approaches and future directions.

Chevalier et al. [15] conducted a comparison of NoSQL multidimensional solutions for DW in which they identified various benchmarks for OLAP on NoSQL and proposed a new benchmark. Their study primarily focused on the proposal itself, rather than examining existing solutions. In contrast, this work provides comprehensive systematic mapping, contributing valuable insights to new proposals in the field.

Aftab and Farooq [16] conducted a survey in 2018, focusing on data warehouses for Big Data. Although their study did not conduct systematic mapping, it addressed various aspects, including data warehouse challenges, quality factors, data preprocessing, and analysis. Additionally, they acknowledged the research challenge of implementing OLAP in Big Data environments. It is important to note that their survey was performed in 2018, while this systematic mapping was up to date as of December 2023. Furthermore, this study covers specific criteria that were not considered in their research.

El Malki et al. [17] proposed a novel method for benchmark functionalities such as support to NoSQL systems, snowflake, star, and flat schemas, variety, and velocity characteristics. They covered multidimensional data warehouses, and NoSQL with the support of column and document-oriented models. However, this study focused on explaining a new approach, and it scarcely mentioned the existing research.

While other works have addressed the topic of OLAP and Big Data, none of them have achieved the level of detail and precision planned in this research on NoSQL. In Table 1, a summary of each related work that has been identified is provided, highlighting the distinctions between these approaches and this study. This observation led to the recognition of the necessity to conduct a systematic mapping study. In the following subsection, the objectives and research questions of this study are delved into.

Table 1. Distinctions between related work and this study.

Related Work	This Study
Cuzzocrea presented an overview of the open issues and future challenges in using OLAP over Big Data [5].	Conducted a systematic mapping study in order to provide a comprehensive guide to current approaches and future directions in integrating OLAP with NoSQL databases.
Chevalier et al. focused on identifying various benchmarks for OLAP on NoSQL, and they proposed a new benchmark [15].	Focused primarily on examining existing solutions to determine gaps and trends.

Table 1. *Cont.*

Related Work	This Study
Aftab and Farooq conducted a survey in 2018, focusing on data warehouses for Big Data [16].	Conducted systematic mapping in 2023, focusing on OLAP for NoSQL.
El Malki et al. covered multidimensional data warehouses and NoSQL with the support of column and document-oriented models [17].	Covered OLAP in column-oriented, document-oriented, graph, and key–value NoSQL databases.

4. Materials and Methods

Following Kitchenham's guidelines to perform a systematic literature review in software engineering [7], this study is structured into three phases: planning, conducting, and reporting. The specific details of each phase are expounded upon in the subsequent sections.

4.1. Planning

The planning phase was subdivided into two parts: the identification of the need and the development of the review protocol, which serves as the foundational framework for conducting a comprehensive and structured analysis.

4.1.1. Identification of the Need

The thorough review conducted in Section 3, Related Work, of other works related to OLAP, NoSQL, and Big Data reveals that researchers have focused their efforts on analyzing new, unstructured data types using OLAP techniques, driven by the interest of companies in expanding analyses based on these data types and the potential of NoSQL databases. Moreover, none of the studies attained the level of detail and precision intended in this research concerning NoSQL. This observation underscores the imperative to undertake an up-to-date, systematic mapping study.

4.1.2. Review Protocol

A well-defined review protocol is crucial to minimize potential biases from researchers, and it should be established before conducting systematic mapping. Furthermore, a review protocol enables other researchers to reproduce and validate research. During this stage, the applied method was developed by first outlining specific development goals. Subsequently, the main research question and seven dimensions aimed at summarizing existing evidence regarding OLAP and NoSQL in Big Data environments were formulated. Finally, an effective strategy to carry out this systematic mapping study was devised.

Objectives

The main objective of this research was to provide up-to-date and comprehensive insights into the most relevant research about the models proposed for integrating OLAP with NoSQL databases within a Big Data environment. As part of the efforts, the following specific objectives were identified:

- Identify the various approaches and techniques utilized in constructing OLAP cubes to effectively handle the complexities of Big Data with NoSQL.
- Gain insights into the prevalent NoSQL databases and their compatibility with OLAP, enabling efficient data storage and analysis in the context of Big Data.
- Examine the computational resources required to implement the proposed methods in designing multidimensional data structures over NoSQL.

Research Question

In the context of this research, three actors within the population were identified: (1) researchers, (2) industry practitioners, and (3) academics, who research, document, and implement solutions for DW with OLAP and NoSQL. The main goal was to summarize the

findings and identify trends and gaps in the studied topic. The research question raised in this study was the following: What approaches and techniques are commonly used to construct OLAP cubes with NoSQL databases to handle Big Data complexities, and how have these methods been implemented?

Based on this main research question, this work aimed to obtain the following seven results (R).

R1: Types of OLAP systems proposed to integrate OLAP with NoSQL databases in Big Data environments.

R2: The most common types of NoSQL databases used with OLAP.

R3: The most prevalent methods for modeling OLAP data cubes.

R4: The proposed structures of OLAP data cubes.

R5: The models suggested for batch and near-real-time processing.

R6: The computational resources required to implement the proposed models.

R7: The performance of the models in terms of query execution times.

Strategy

The strategy devised for this research on OLAP and NoSQL in Big Data environments encompassed five key actions:

1. Initial studies were meticulously sourced from reputable scientific digital libraries, focusing on criteria such as indexed research documents, database update frequency, and relevance to the topic of interest.
2. Stringent exclusion and inclusion criteria were implemented to filter studies related to OLAP, NoSQL, and Big Data, using specific search terms across titles, abstracts, and keywords within the chosen digital libraries to select primary studies.
3. A full content review was carried out on the content of the selected primary studies to select relevant papers.
4. The snowballing technique was employed to expand the search and identify additional pertinent articles by examining references from within the already-reviewed studies.
5. Finally, the relevant studies were meticulously reviewed to obtain insights that address the research question and the expected results.

4.2. Conducting

As planned in the strategy for this systematic mapping, initially, research studies were identified from scientific digital libraries, and they were selected based on their relevance, which was primarily determined according to the number of publications [18,19]. These libraries include the following:

- IEEE Xplore
- Google Scholar
- Scopus
- Springer
- Web of Science (WoS)

The inclusion criteria applied to the initial studies encompassed those with the common search syntax ("*Big Data*" or "*NoSQL*") and "*OLAP*" in the title, abstract, and keywords. In this work, the proposals related to the implementation of OLAP on NoSQL databases were identified. To cover the largest portion of studies, the search was expanded with the term "Big Data", obtaining the largest number of articles related to this research question. Additionally, only studies from journals and conferences written in English were selected. The search was not limited by a time frame, and it was conducted in December 2023. As a result, 1646 studies were identified, of which 249 were duplicates; this represents approximately 15.1% of the initially identified studies.

The computational tool used to consolidate the identified research studies and develop the selection of primary and relevant studies was State of the Art through Systematic Review (StArt) [20].

A very interesting result was the unexpected proposals to create OLAP cubes in environments like Hadoop using the MapReduce paradigm. This approach challenges the initial belief that these technologies were limited to Big Data processing and reporting. In the majority of cases, the proposals include the use of the Hive database, which is not considered a NoSQL database [21]. For this reason, those works were discarded. Hence, this work focuses only on those that present results on NoSQL databases.

In the selection phase, after performing a quick review of the titles, abstracts, and keywords of the 1397 articles, only 182 were selected as primary studies, representing approximately 13% of the initial articles. Finally, after thoroughly scanning the entire content of these studies, 22 were accepted as relevant studies, representing approximately 12% of the primary selected 182 studies. With the snowballing technique, 3 more were selected, resulting in a final corpus of 25 articles. In conclusion, out of the total 1646 articles initially found, only 1.5% focused on addressing the research question. A summary of the final corpus selection process is provided in Figure 1.

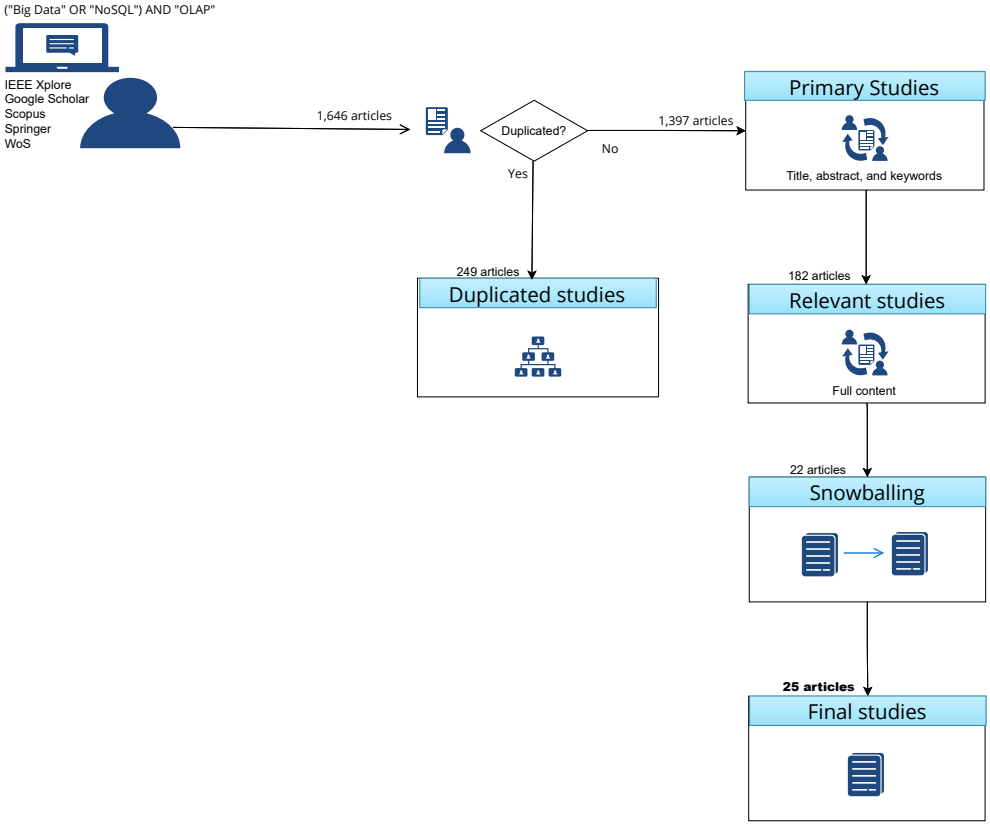

Figure 1. Conduction stage for systematic mapping.

4.3. Reporting

This stage was crucial, as it aimed to address the research question raised in the review protocol. As a result, Section 5, Results, provides a detailed presentation of the findings derived from the final selected studies. Additionally, Table 2 summarizes the concept matrix derived from the features identified in the mapping process applied to the 25 selected relevant papers.

Table 2. Concept matrix.

Reference	Authors	Year	Title	Journal/Conference	R1	R2	R3	R4	R5	R6	R7
[22]	Jiao et al.	2012	CDDTA-JOIN: One-Pass OLAP Algorithm for Column-Oriented Databases	Web Technologies and Applications	-	Column-oriented	Snow-flake	-	Batch	YES	-
[23]	Dehdouh et al.	2014	Columnar NoSQL CUBE: Aggregation operator for columnar NoSQL data warehouse	International Conference on Systems, Man, and Cybernetics	MOLAP	Column-oriented	-	CN-Cube	Batch	YES	YES
[24]	Zhao et al.	2014	A multidimensional OLAP engine implementation in key–value database systems	Advancing Big Data Benchmarks: Proceedings of the 2013 Workshop Series on Big Data Benchmarking	MOLAP	Key–value	Star	-	Batch	YES	YES
[25]	Li et al.	2014	R-Store: A scalable distributed system for supporting real-time analytics	IEEE International Conference on Data Engineering	RTOLAP	Column-oriented	-	-	Real	YES	-
[26]	Chevalier et al.	2015	How can we implement a multidimensional data warehouse using NoSQL?	Enterprise Information Systems International Conference	ROLAP	Document-oriented	Snow-flake	-	Batch	YES	YES
[27]	Cuzzocrea et al.	2015	Taming Size and Cardinality of OLAP Data Cubes over Big Data	Data Analytics: International Conference on Databases	ROLAP	Column-oriented	Snow-flake	-	Batch	YES	-
[28]	Lee et al.	2015	Efficient level-based top-down data cube computation using MapReduce	Transactions on Large-Scale Data- and Knowledge-Centered Systems XXI	ROLAP	-	-	MRLevel, MRPipeLevel	-	YES	-
[29]	Song et al.	2015	Ha-OLAP: A Hadoop based OLAP system for Big Data	Journal of Systems and Software	HaOLAP	-	-	-	Batch	YES	-
[30]	Chevalier et al.	2015	Implementing Multidimensional Data Warehouses into NoSQL	International Conference on Enterprise Information Systems	ROLAP	Column-oriented, document-oriented	Star	-	Batch	YES	-

Table 2. *Cont.*

Reference	Authors	Year	Title	Journal/Conference	R1	R2	R3	R4	R5	R6	R7
[31]	Dehdouh et al.	2015	Using the column-oriented NoSQL model for implementing Big Data warehouses	International Conference on Parallel and Distributed Processing Techniques and Applications	-	Column-oriented	Star	-	Batch	YES	-
[32]	Chevalier et al.	2016	Document-oriented data warehouses: Models and extended cuboids, extended cuboids in oriented document	International Conference on Research Challenges in Information Science	ROLAP	Document-oriented	Star	-	Batch	YES	-
[33]	Scabora et al.	2016	Physical data warehouse design on NoSQL databases-OLAP query processing over Hbase	International Conference on Enterprise Information Systems	-	Column-oriented	Star	-	Batch	YES	-
[34]	Chen et al.	2018	An optimized distributed OLAP system for Big Data	IEEE International Conference on Computational Intelligence and Applications	ROLAP, MOLAP	Column-oriented	-	-	Batch	YES	-
[35]	El Malki et al.	2018	Benchmarking Big Data OLAP nosql databases	In Ubiquitous Networking: 4th International Symposium	MOLAP	Column-oriented, Document-oriented	Snow-flake, star, flat	-	Batch	YES	YES
[36]	Guminska and Zawadzka	2018	Ev-OLAP graph–evolution and OLAP-aware graph data model	In Beyond Databases, Architectures and Structures	Ev-OLAP	Graph	Star	-	Batch	YES	-
[37]	Ferro et al.	2019	Document-oriented geospatial data warehouse: An experimental evaluation of SOLAP queries	IEEE 21st Conference on Business Informatics	SOLAP	Document-oriented	-	-	Batch	YES	-
[38]	Dehdouh et al.	2020	Big Data Warehouse: Building columnar NoSQL OLAP cubes	International Journal of Decision Support System Technology	MOLAP	Column-oriented	Star	MC-Cube	Batch	YES	-
[39]	Gómez et al.	2020	Online analytical processing on graph data	Intelligent Data Analysis	-	Graph	Star	Graphoid	Batch	YES	-

Table 2. *Cont.*

Reference	Authors	Year	Title	Journal/Conference	R1	R2	R3	R4	R5	R6	R7
[40]	Jianmin et al.	2020	An improved join-free Snow-flake schema for ETL and OLAP of data warehouse	Concurrency and Computation: Practice and Experience	ROLAP	Column-Oriented	Snow-flake	-	Batch	YES	-
[41]	Khalil et al.	2020	Key–value data warehouse: Models and OLAP analysis	International Conference on Electronics, Control, Optimization and Computer Science	ROLAP, MOLAP	Key–value	Star	KV-CUBE	Batch	-	-
[42]	Yue et al.	2020	Geocube: Towards the Multi-Source Geospatial Data Cube in Big Data Era	IEEE International Geoscience and Remote Sensing Symposium	SOLAP	Column-oriented	Geo-Cube	-	Batch	-	-
[43]	Akid et al.	2022	Performance of NoSQL Graph Implementations of Star vs. Snow-flake Schemas	IEEE Access	MOLAP	Graph	Star, Snow-flake	-	Batch	YES	YES
[44]	Khalil et al.	2023	An Approach for Implementing Online Analytical Processing Systems under Column-Family Databases	International Journal of Applied Mathematics	C-OLAP	Column-oriented	Star	MRC-Cube, SC-Cube	Batch	-	YES
[45]	Khalil et al.	2023	A Data Placement Strategy for Distributed Document-oriented Data Warehouse	International Journal of Computer Science	-	Document-oriented	Star	MRC-Cube, SC-Cube	Batch	YES	YES
[46]	Labzioui et al.	2023	New Approach based on Association Rules for Building and Optimizing OLAP Cubes on Graphs	International Journal of Advanced Computer Science and Applications	-	Graph	Galaxy	-	Batch	YES	-

5. Results

Researchers have dedicated their efforts to delving into the scientific realm of analyzing novel, unstructured data types using OLAP methodologies. This scientific pursuit is spurred by two fundamental factors: the keen interest of corporations in broadening their analytical scope to encompass these emerging data types, particularly unstructured data, and the vast potential offered via NoSQL databases. A crucial aspect driving this exploration is the imperative need for systems capable of executing both real-time and batch analyses, ultimately enhancing the decision-making process [26,36].

Through empirical experimentation, it has been demonstrated that employing OLAP on NoSQL platforms can yield superior performance in terms of query execution times compared to traditional DW solutions. Furthermore, NoSQL presents a simpler configuration approach to managing vast data sets. For instance, a notable study [40] underscores that, while conventional data warehouses necessitate a model rebuild for new queries, NoSQL platforms require only the generation of concise code. Building upon these scientific insights, this section elucidates the seven pivotal discoveries that directly address the core research question of this study.

5.1. R1—Types of OLAP Systems Proposed to Integrate OLAP with NoSQL Databases in Big Data Environments

The systematic mapping conducted in this study revealed a comprehensive spectrum of OLAP proposals, showcasing the diversity in conceptualizations. The distribution of seven types of OLAP systems, along with the respective number of proposals identified, is presented in Figure 2 and detailed below.

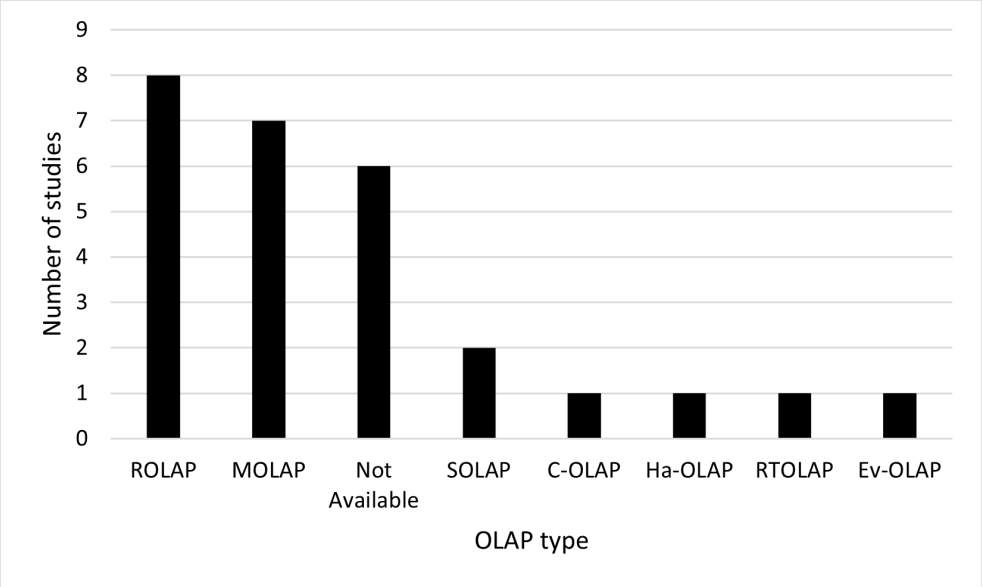

Figure 2. Types of OLAP systems used with NoSQL.

ROLAP: Eight proposals were identified, representing approximately 32% of the studies, that involved mapping data from NoSQL databases into a relational structure that can be queried using traditional SQL-based OLAP techniques [26–28,30,32,34,40,41]. The process involves creating a virtual layer or schema on top of the NoSQL data and transforming it into a relational format. This allows analysts and applications to perform complex queries, aggregations, and analyses using SQL-like expressions, similar to how they would with traditional relational databases. ROLAP provides a bridge between the flexible and scalable storage capabilities of NoSQL databases and the analytical requirements of OLAP.

MOLAP: Seven proposals, representing 28% of the total studies, adapted a multidimensional data model to fit the flexible schema of NoSQL systems [23,24,34,35,38,41,43]. Initially, a multidimensional data model was defined to capture relevant dimensions, hier-

archies, and measures for analytical purposes. The data model was then adapted to align with the flexible schema of NoSQL databases, accommodating diverse data structures. Subsequently, data were loaded into the NoSQL database, and multidimensional cubes were constructed to pre-aggregate and summarize data along different dimensions.

SOLAP: Two proposals, which account for 8% of the total studies, indicated a niche focus on incorporating spatial elements into OLAP systems, allowing for multidimensional analysis with spatial considerations [37,42]. Firstly, a spatial data model was defined, encompassing geographical dimensions, measures, and hierarchies relevant to analytical needs. The spatial data model was adapted to align with the schema-less nature of NoSQL databases, which can handle diverse spatial data types. Data are then loaded into the NoSQL database, ensuring compatibility with the spatial data model. Spatial indexing techniques within NoSQL are employed to optimize spatial queries and analyses. Multidimensional cubes, which incorporate spatial dimensions, are constructed, facilitating efficient OLAP operations.

Columnar OLAP (C-OLAP): One proposal [44], accounting for 4% of the 25 proposals, introduced the concept of C-OLAP, suggesting a novel approach to implementing OLAP using columnar databases. C-OLAP involves transforming the multidimensional conceptual model used as the basis of data warehouses and OLAP applications to a target columnar logical model. This transformation allows for the efficient storage and retrieval of data for OLAP operations.

Furthermore, C-OLAP introduces specific OLAP operators [44], such as Map-Reduce Columnar Cube (MRC-Cube) and Spark Columnar Cube (SC-Cube), which leverage technologies like Hadoop MapReduce and Apache Spark to compute OLAP cubes. The MRC-Cube operator works in multiple stages to compute the lattice of cuboids sequentially. It involves the extraction of data from the column family data warehouse and the performance of multiple joins between the fact and its dimensions. The operator leverages the MapReduce paradigm to efficiently process and aggregate data for OLAP cube construction.

The SC-Cube operator works in multiple stages to compute the OLAP cube [44]. SC-Cube involves reading input data from the columnar database and converting each row to a key–value pair Resilient Distributed Dataset (RDD) in which the key is the row key and the value is a nested map structure that associates a given column family and column name to a value. The operator then applies transformations to fetch only the columns that compose the cube, and it generates a new pair RDD in which the key is a combination of all the dimensions involved in the cube and the value is the measure to be aggregated.

Hadoop OLAP (Ha-OLAP): One proposal [29], accounting for 4% of the total studies, introduced Ha-OLAP, which adopts a simplified, multidimensional model to map dimensions and measures, and it uses dimension coding and traversing algorithms to achieve roll-up operations over dimension hierarchies. Ha-OLAP also employs partition and linearization algorithms to store data and chunk selection strategies in order to filter data. The system architecture of Ha-OLAP includes a Hadoop cluster, a metadata server, a job node, an OLAP service facade, and an OLAP client.

Real-Time OLAP (RTOLAP): One proposal [25], representing 4% of the total studies, addressed Real-Time OLAP. In contrast to the other findings, this represents the sole proposal centered specifically on real-time data processing.

RT-OLAP refers to the capability of performing OLAP queries in real time with column-oriented databases. In the context of the R-Store system, RT-OLAP involves accessing the latest value preceding the submission time of the query for each key, and it aims to provide real-time analytics to make effective and timely decisions.

R-Store is a scalable, distributed system designed to support real-time OLAP queries by extending the MapReduce framework and utilizing HBase as the underlying storage system. It maintains a real-time data cube and implements incremental scanning to efficiently process real-time queries, ensuring the freshness of answers and low processing latency.

Evolutionary OLAP (Ev-OLAP): One proposal, which accounted for 4% of the total studies, introduced Evolutionary OLAP, suggesting an approach that adapts OLAP systems

to graph databases [36]. Ev-OLAP was defined by the authors as a single graph with a set of nodes, a set of edges, and a set of labels of all nodes and edges. The model implements Ian Robinson's approach of separating the structure from the state, allowing the independent versioning of the graph topology and the state. The graph distinguishes between identity nodes, structural edges, state nodes, and state edges. Ev-OLAP introduces hypernodes, representing meta-nodes containing identity nodes with all connected state nodes. Hierarchical edges associate the levels of an abstracted hierarchy as identity nodes.

The model's mechanics enable the addition and modification of a graph structure without deletion. For added entities, new state nodes that represent changes over time are introduced. Deletion is treated similarly to modification, with the new state of the entity stored in the graph. The evolution-awareness feature facilitates the analysis of changes in both the state of the graph and its structure over time.

In terms of OLAP awareness, the paper presents core data warehouse terms for Ev-OLAP. Dimensions are represented by node labels, and measures are divided into informational (numeric metrics in attributes or relationships) and topological (graph structure analysis metrics) categories. Facts in Ev-OLAP can be concealed from the entire graph as events modeled as either nodes or relationships. Hierarchies are created using a special type of relationship called hierarchical edges, indicating the start of a hierarchy path and the direction of an increasing detail level. The proposed model handles analytical queries, addressing issues like slowly changing dimensions in traditional data warehouses. Ev-OLAP preserves historical data, allowing for both old and new data to coexist with information on validity periods.

Additionally, there were six instances in which the type of OLAP was not explicitly categorized, labeled as Not Available (NA) in Figure 2. This detailed breakdown provides a nuanced understanding of the diverse OLAP approaches identified in the literature.

Figure 3 provides a comprehensive overview of how studies are distributed in the realm of OLAP analysis across various types of NoSQL databases. It is highlighted that, in terms of popularity, ROLAP is the most studied in column-oriented and document-oriented databases, but it has also been studied in key–value databases. ROLAP is widely researched, as is MOLAP for column-oriented databases. On the other hand, in key–value databases, MOLAP emerges as the most proposed approach, followed by ROLAP, indicating the versatility of these analysis models across different data storage types. MOLAP has also been proposed in graph databases.

It is interesting to note that SOLAP, RTOLAP, and C-OLAP have a more limited representation compared to ROLAP and MOLAP. However, implementations of SOLAP have been proposed in both column and document-oriented databases, suggesting a growing interest in spatial analysis in these contexts. On the other hand, RTOLAP and C-OLAP are more related to column-oriented databases, while Ev-OLAP stands out in the realm of graph databases.

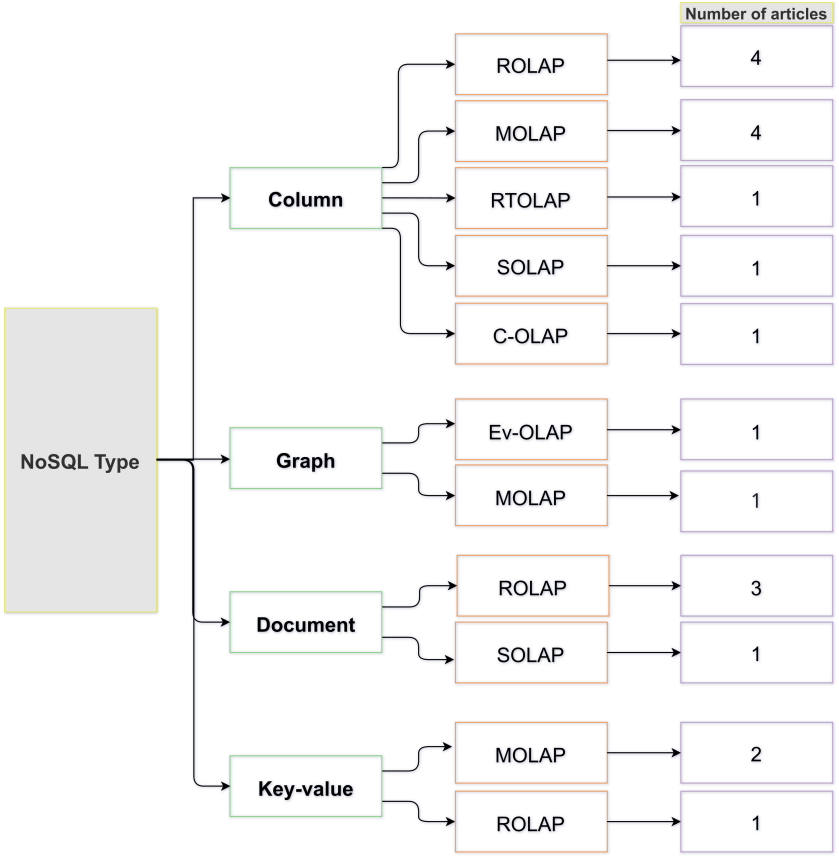

Figure 3. Types of OLAP system by NoSQL types.

5.2. R2—The Most Common Types of NoSQL Databases Used with OLAP

The results presented in Figure 4 indicate that column-oriented databases are the most commonly proposed, with 13 articles, 52%, presenting OLAP integration proposals. Document-oriented and graph databases follow closely, with five, 20%, and four articles, 16%, respectively, outlining approaches to combining OLAP functionality with their respective NoSQL types. Key–value stores and a few instances categorized as NA were also explored, with two articles, 8%, each. These findings highlight the diverse landscape of NoSQL databases utilized in conjunction with OLAP, showcasing the adaptability of OLAP across different NoSQL data models.

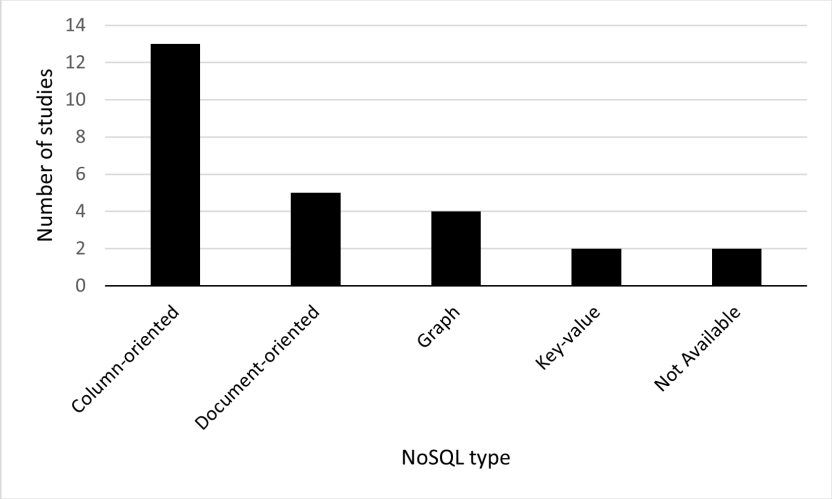

Figure 4. NoSQL types proposals with OLAP.

The analysis of the most commonly used NoSQL databases in conjunction with OLAP reveals specific Database Management Systems (DBMSs) associated with each NoSQL type, as shown in Figure 5. In the column-oriented category, HBase stands out as the predominant choice, featured in eleven articles [22,23,25,30,31,33,34,38,40,42,44], 44%, while Cassandra [35] and NA [27] were mentioned in one article, 4%, each. Document-oriented databases were notably represented by MongoDB in four articles [26,30,35,37], 16%, with one article marked as NA [45]. For graph databases, Neo4J emerged as a prominent choice, mentioned in all four relevant articles [36,39,43,46], which constituted 28% of the total. Key–value stores exhibited two proposals, 8%, with HBase [24] and Oracle NoSQL Database [41]. Although HBase was not typically categorized as a key–value NoSQL database, the authors presented their proposals within this framework as a key–value solution.

These detailed insights into the specific DBMS associated with each NoSQL type provide a comprehensive understanding of the diverse platforms integrated with OLAP for various analytical purposes. This information can help select the best combination according to the context of use.

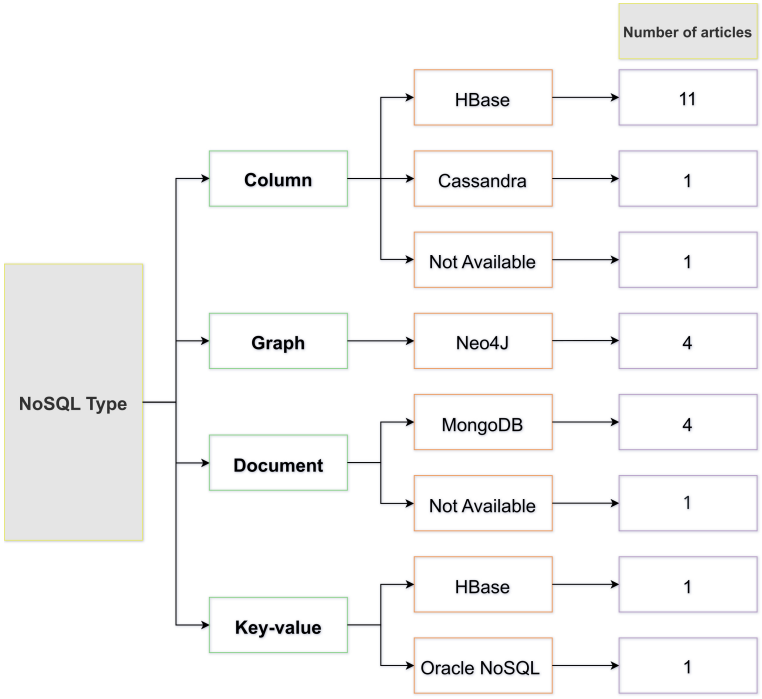

Figure 5. Database management systems commonly used.

5.3. R3—The Most Prevalent Methods for Modeling OLAP Data Cubes

The results of the study, summarized in Figure 6, reveal that the star method is the most prevalently reported, with 13 occurrences, 52% [24,30–33,35,36,38,39,41,43–45]. Following behind is the snowflake method, with six instances, 24%, identified [22,26,27,35,40,43]. Additionally, the flat method, characterized by denormalized data storage in a single table, was found in two instances [32,35], representing 8%. Less commonly encountered were the galaxy [46] and geo-cube [42] methods, each appearing once in the data set, which constituted 4% of the total. NA was classified in six articles [23,25,28,29,34,37], 24%, indicating either a lack of information or an inability to categorize according to the predefined methods.

The analysis of Figure 7 reveals that the star method emerged as the most commonly employed approach across all data types, with six instances, 24%, reported in the Column category, three instances, 12%, in the Graph category, and four instances, 16%, in the Document category. This suggests a widespread adoption of the star method to model OLAP data cubes across different types of data. Conversely, the Snowflake method is less prevalent overall, with four occurrences, 16%, in the Column category,

one in the Graph category, accounting for 4%, and two in the Document category, representing 8%. This indicates a lower frequency of using the Snowflake method compared to the Star method across the different data types analyzed. The Flat method was observed in only one instance, 4%, within the Column category and two instances, 8%, within the Document category, suggesting a less common but still present usage for certain types of data. Additionally, the geo-cube and galaxy methods exhibited minimal usage, each appearing only once and accounting for 4% in the Column and Graph categories, respectively.

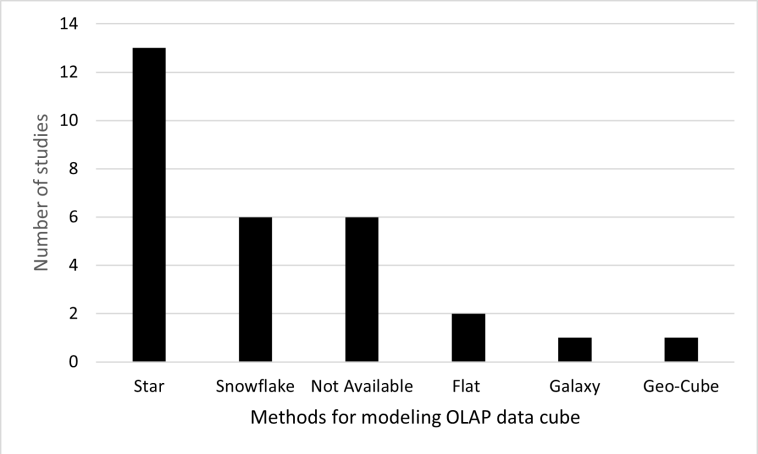

Figure 6. Methods of modeling OLAP data cubes.

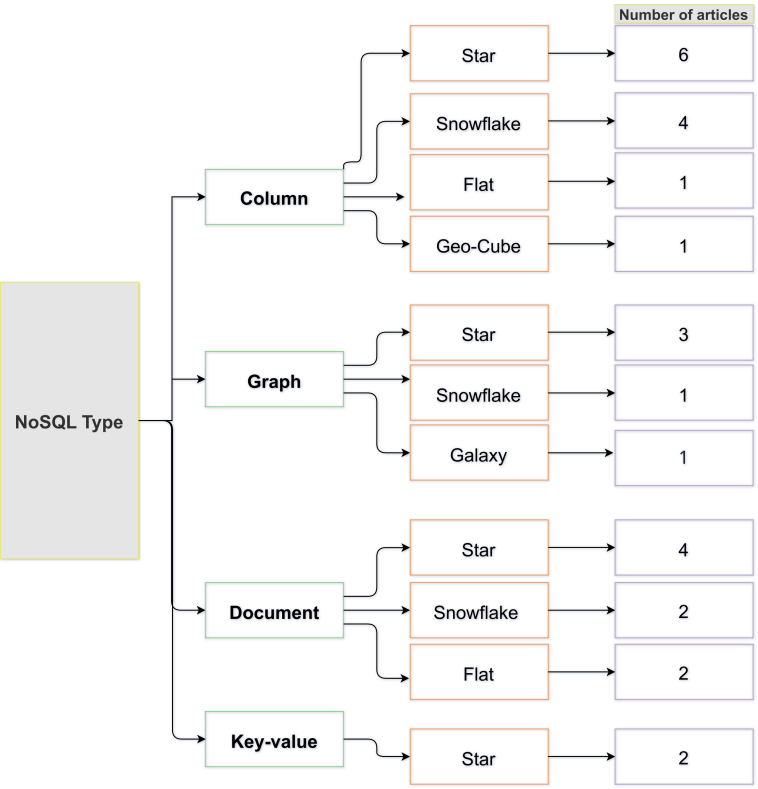

Figure 7. Data modeling methods across different NoSQL types.

5.4. R4—The Proposed Structures of OLAP Data Cubes

In the proposals, the structure of OLAP data cubes follows a typical multidimensional format, consisting of dimensions, measures, and cells. Dimensions serve as axes and encompass various attributes, while each dimension is organized into hierarchies, facilitating

granularity levels. Measures represent the numerical data under analysis or aggregation. Cells, located at the intersection points of dimensions and measures within the data cube, hold the aggregated values. Furthermore, OLAP data cubes are often structured using fact tables and dimension tables to efficiently organize and store data, with fact tables containing detailed transactional data and dimension tables providing descriptive attributes for analysis.

Another significant discovery from the review of selected studies is the novelty proposal, such as cube operators, including Columnar NoSQL Cube (CN-Cube), Key–Value Cube (KV-Cube), a graphoid model, Map Reduce Cube (MR-Cube), Spark Cube (SC-Cube), Map Reduce Columnar Cube (MRC-Cube), and Map Reduce Columnar Cube (MC-Cube). These last two abbreviations, MRC-Cube and MC-Cube, were used in different works to refer to the same meaning, which is the Map Reduce Columnar Cube. However, the proposed methods differ in each case. These operators showcase a wide range of approaches and methodologies, highlighting the novelty within the realm of cube operations in OLAP systems over NoSQL. Furthermore, algorithms such as MRLevel and MRPipe Level have been proposed for the efficient computation of level-based, top-down data cubes.

CN-Cube [23] is an aggregation operator designed for column-oriented NoSQL database management systems. It allows OLAP cubes to be computed using column-oriented NoSQL data warehouses with a view based on the attributes (dimensions and measures) needed to compute the OLAP cube. The operator uses value positions and hash tables to take into account all dimension combinations and extract data that satisfy the predicates of the query, thus producing the cells and measures needed for OLAP cube computation. It has been implemented using the SQL Phoenix interface of HBase DBMSs, and it has been shown to have OLAP cube computation times very suitable for NoSQL warehouses.

CN-Cube follows a series of steps, starting with data extraction and grouping, in which a query is initiated to extract data meeting specific conditions and is then grouped based on dimensions. These grouped data form an intermediate result relation denoted as R, which contains dimensions and measures for aggregation. Each dimension in R is hashed to create position lists indicating a presence or an absence, and the logical AND function is used to find the intersection of these lists, providing sets of positions that represent combined dimension values for aggregation. These steps efficiently create the data cube, enabling total and partial aggregations at various levels of granularity.

The experimental phase of the proposed model involved two main experiments to assess the performance of the CN-Cube operator in OLAP cube computation times. The first experiment evaluated cubes with two to five dimensions using a data set of 60 million records. Four OLAP cube computation queries were performed, showing that the CN-Cube operator performed better than the traditional Cube operator as the number of dimensions increased, resulting in faster computation times.

The second experiment focused on evaluating the scalability of the CN-Cube operator within a multi-node cluster. It assessed execution times for a three-dimensional OLAP cube across various configurations and data sample sizes (100 GB, 500 GB, and 1 TB), including single-node, five-node, ten-node, and fifteen-node clusters. The results showed that increasing the number of nodes led to decreased computation times for OLAP cubes of different warehouse sizes, particularly for larger data warehouses. This scaling effect highlighted significant reductions in computation times with more cluster nodes, offering valuable insights into the efficiency of the operator CN-Cube in computing OLAP cubes within column-oriented NoSQL data warehouses.

KV-Cube [41] is structured with dimensions, cells, and measurements. The structure involves the use of the Bit-Encoded Sparse Storage (BESS) technique to store dimensions and measurements, allowing for the efficient computation of OLAP cubes. Additionally, KV-Cube is designed to support basic OLAP operations, and it is implemented using a key–value data model.

BESS assigns a binary index to each dimension member, minimizing bits for optimal storage, and it concatenates these binary representations to form cuboid indexes that store corresponding values. Retrieving data involves using bit mask operations to extract dimension indexes, enabling quick access to desired information. This integration ensures that KV-Cube represents multidimensional data structures with minimal storage, which is ideal for large-scale OLAP operations and supporting rapid data retrieval, the seamless execution of basic OLAP operations, and effective multidimensional data analysis and decision-making processes.

During the experimental analysis of the proposed model, a comparison between KV-Cube and traditional Oracle Cubes was conducted, focusing on storage space consumption for the Embedded Logical Model (ELM) and Hierarchical Logical Model (HLM) across different scale factors. The evaluation also included measuring the response times of OLAP queries by incrementally increasing the number of dimensions in the queries. Utilizing the TPC-H benchmark, consisting of eight separate tables with defined relationships, the experiments were conducted using the Oracle NoSQL database and Oracle 11g Release 2 as containers within a DevOps approach using Docker.

The comparison was based on the elapsed time in milliseconds to execute OLAP queries with two to four dimensions and a scale factor equivalent to 5.2 million line items. The results indicated that KV-Cube outperformed Oracle Cubes, showing up to three times faster query response times. This advantage was attributed to the efficient data structure of KV-Cube, utilizing BESS for dimension storage, which enabled rapid data extraction once dimension combinations were established. The integrated caching feature further bolstered the data retrieval speed, showcasing KV-Cube as a promising solution for efficient OLAP operations within key–value data models.

A graphoid [39] is not an OLAP cube. Instead, a graphoid is a node- and edge-labeled directed multi-hypergraph that serves as a basic data structure for modeling OLAP on graph data. It represents information on the application domain at a certain level of granularity and can be defined at several different levels of granularity using associated dimensions. The paper proposes a formal multidimensional model for graph analysis, and it demonstrates that the typical OLAP operations on cubes can be expressed over the graphoid model. It also shows that the classic data cube model is a particular case of the graphoid data model. The paper presents a formal definition of the graphoid model for OLAP and proves that the classic OLAP queries remain competitive when using the graphoid model.

In the experimental analysis of the proposed model, a case study involving group calls between phone lines was used to analyze the data. The data set comprised calls in which a line could not call itself, requiring the identification of the initiating line. This study aimed to assess the hypergraph model against the traditional relational OLAP approach, particularly focusing on analyzing vast amounts of call data with variable dimensions due to varying participant counts. The analysis encompassed standard OLAP operations on fact measures and the aggregation of graph elements using graph measures such as shortest paths and centrality.

The experiment compared the performance of the graphoid model in two scenarios: the classic OLAP scenario using the relational model and the graph OLAP scenario involving the aggregation of graph metrics. The hypothesis tested was that, while the relational OLAP approach is effective for fixed dimensionality scenarios, the graphoid model is competitive when dealing with variable dimensions, and it outperforms in scenarios requiring graph metrics aggregation. The results provided valuable insights into the model of the graphoid effectiveness for modern data analysis needs, demonstrating its capability to deliver superior performance for critical queries.

MR-Cube [25,29] refers to the use of the MapReduce framework to efficiently compute and maintain data cubes in a large-scale distributed environment. In one proposal [25], the data cube consisted of a lattice of cuboids in which each cuboid represents a combination of dimensions. The cuboids are used to organize the data cube, and the map and reduce

functions are used to compute the aggregation value for each cell of each cuboid. The map output key is the combination of the dimension attributes for the cuboid, and the map output value is the numeric value. The reduce function is invoked to compute the new value of each cell based on the old cell value, the change in the cell, and the aggregation function. In one work [29], the cube was divided into chunks, and each chunk contained cells, which were the logical partitions of the cube. The cells contained measurements, and the cube was organized based on dimensions, which were used to represent the different aspects of the data being analyzed.

In one work [29], the analysis compared MR-cubes with several cloud data warehouse systems like Hive, HadoopDB, Olap4Cloud, and HBaseLattice in terms of loading, dice, roll-up, and storage performance. Ha-OLAP excelled in loading performance, especially compared to Olap4Cloud and HBaseLattice, due to its simplified data model and lack of index structure generation during loading. Queries consistently showed superior performance, except against HBaseLattice, in terms of time consumption. Roll-up operations across data sets revealed reduced result sizes due to aggregation, with time consumption compared across systems. Storage experiments highlighted the low storage cost of Ha-OLAP even with high-dimensional data sets, showcasing its efficiency for Big Data analytics tasks compared to other cloud data warehouse systems.

SC-Cube [44,45] is an OLAP cube operator that uses the Apache Spark framework to compute OLAP cubes. It processes data in memory using RDDs to speed up data flow between iterations, thus overcoming the I/O cost associated with processing data on a disk. SC-Cube performs cube computation in five stages. The first stage involves reading input data, while the second stage focuses on building the lowest level of granularity. The third stage computes higher granularity levels, and the fourth stage performs aggregation with dimension attributes. Finally, the fifth stage involves materializing the cube. SC-Cube is designed to take full advantage of in-memory processing, and it has been shown to outperform the MapReduce paradigm in terms of performance.

The experimental analysis in one work [44] evaluated the performance, scalability, and efficiency of OLAP cube operators, including the proposed SC-Cube, through three key experiments. The first experiment compared SC-Cube and MRC-Cube with a traditional relational approach as the data volume increased, revealing consistent performance for SC-Cube and MRC-Cube due to NoSQL databases and parallel processing, while the relational approach slowed significantly with larger data sets. The second experiment focused on building full OLAP cubes using the SC-Cube Spark-Cube component, MR-Cube, and Apache Hive, showcasing Spark-Cube's faster execution times due to in-memory processing. The third experiment assessed the query response times for all operators, highlighting the superior performance of Spark-Cube for complex queries, thanks to in-memory processing and optimized join operations.

The experimental analysis in another work [45] evaluated the performance and efficiency of SC-Cube compared to other OLAP cube operators. The study involved comparing SC-Cubes with Apache Hive across various analytical queries and cube-building tasks, measuring key metrics like execution time and storage space under different scales and scenarios. The analysis aimed to assess the ability of SC-Cube to handle large data volumes and varying scalability demands. Additionally, the study compared the execution time for building full cubes using MR-Cube and Spark-Cube from the proposed model of the data warehouse to the default model of the Apache Hive. The study also evaluated the response time of SC-Cube for processing analytical queries with varying dimensions in grouping clauses to gauge its effectiveness in handling complex queries.

MRC-Cube [44,45] utilizes the MapReduce processing technique to build the cube in multiple stages. The first stage involves extracting the data that forms the cube from the column family, followed by a reduced side join operation. The second stage focuses on building the first level of the cube corresponding to each dimension combination. The third stage uses the output of the second stage to calculate the second level of granularity, representing different dimension combinations.

The experiments described in one work [44] focused on evaluating the performance and scalability of MR-Cube compared to traditional relational OLAP implementations, specifically Oracle OLAP. These experiments aimed to demonstrate the benefits of using NoSQL technology and columnar databases for OLAP cube construction and analysis. The first experiment assessed the storage efficiency of OLAP cubes built with MR-Cube compared to the default star schema model, providing insights into storage optimization achieved through the column–family architecture. The second experiment measured the execution time of building full OLAP cubes using MR-Cube from a data warehouse based on the proposed model, with Apache Hive serving as a benchmark competitor against the default model. This comparison highlighted the efficiency and performance gains of MRC-Cube over traditional OLAP implementations. Finally, Experiment 3 evaluated the scalability of MRC-Cube as the data warehouse size expands, showing consistent performance and scalability in handling large data sets compared to relational OLAP implementations.

The experiments in another work [45] primarily focused on evaluating the response time of analytical queries with varying dimension numbers in grouping clauses, specifically analyzing the performance of the implemented OLAP system using MRC-Cube and SC-Cube operators. These experiments provided insights into the efficiency of OLAP cube operators in managing analytical queries with diverse dimension combinations, and they highlighted the impact of dimension numbers on query processing time and system performance. Additionally, the experiments assessed the response time of queries with variations in the scale factor, showcasing the performance disparities between Spark-Cube, MR-Cube, and Hive when scaling up the data volume. The results emphasized the advantages of memory-based computation in Spark over disk-based operations in MapReduce, contributing valuable insights into query performance and scalability in OLAP systems using MRC-Cube and SC-Cube operators.

MC-Cube [38] is also an aggregation operator designed to build OLAP cubes using a column-oriented NoSQL model like MRC-Cube. In this proposal, MC-Cube is structured to perform a cube in five phases. In the first phase, the study identifies the data that satisfy all the predicates and allow the aggregation according to all columns representing dimensions to be produced. Then, it implements the invisible join in a distributed environment to perform the join between tables and achieve aggregation computing. MC-Cube uses the MapReduce paradigm to optimize the processing of massive data, and it executes MapReduce jobs to achieve the five phases of building an OLAP cube.

The data analysis phase of the research involved recording and analyzing the time required to compute queries and construct OLAP cubes. This analysis aimed to evaluate the performance of the system across various query types and dimensions, using computation time as a key metric. Subsequently, the results of these experiments, including computation and cube construction times, were reported to assess the efficiency and effectiveness of the proposed MC-Cube operator. The analysis of experimental data yielded valuable insights into the performance of the operator in efficiently building OLAP cubes and conducting data analysis tasks. Through experiments with diverse queries and dimensions, the ability of the proposed model to handle large data sets and perform tasks effectively was evaluated, shedding light on its performance and scalability in real-world scenarios.

MRLevel [28] utilizes the MapReduce framework to efficiently compute level-based top-down data cubes, aiming to parallelize cube computation and reduce the number of data scans by level. It operates by processing levels in a top-down manner and storing cuboid results with the cuboid size from the cube lattice structure.

Moreover, **MRPipeLevel** [28] integrates the MRLevel algorithm with the PipeSort algorithm, known for its efficiency in top-down ROLAP cube computation. PipeSort generates a minimum-cost sort plan tree from a cube lattice and computes cuboids sharing the same sort order to minimize the computation time and data scans. On the other hand, **MRPipeLevel** [28] enhances its performance by incorporating a distributed parallel processing strategy for PipeSort in the MapReduce framework, maximizing parallelism,

and minimizing MapReduce phases and data scans. It includes pipeline and multi-pipeline aggregation methods that aim to reduce the computational costs for Big Data and high-dimensional cubes.

In the experimental phase of the proposed model, the performance of the MRPipeLevel algorithm was thoroughly evaluated through a series of experiments. The analysis of data from the proposed model involved several key steps. Firstly, the MRPipeLevel algorithm was implemented to efficiently compute data cubes using MapReduce in a distributed parallel processing environment. Various experiments were conducted to assess the performance of the algorithm across different scenarios, including low- and high-dimensional data sets. Comparative experiments were executed with other MapReduce data cube algorithms to gauge the effectiveness of the MRPipeLevel approach. Secondly, the experiments encompassed variations in data size, dimensions, and cluster numbers to analyze the adaptability of the algorithm under diverse conditions. The elapsed time for sorting trees and pipelines was meticulously measured and analyzed to comprehend the efficiency of the MRPipeLevel algorithm in processing data cubes. Through these systematic experiments and thorough analysis, the researchers successfully demonstrated the efficiency and effectiveness of the MRPipeLevel algorithm in computing data cubes within a distributed and parallel processing environment using MapReduce.

5.5. R5—The Models Suggested for Batch and Near Real-Time Processing

Out of the 25 studies comprising the final corpus, 23 presented their proposals with batch data analysis, which accounted for 92% of the total. Only one study [25], 4%, focused on real-time data, while one [28] did not specify its data processing method. In batch analysis, the data typically used for testing are generated using warehouse benchmarks, which are widely used to create data for decision support systems. These benchmarks are populated with data samples, and they enable the generation of data sets of various sizes by specifying the scalability factor (SF), which refers to a parameter that determines the size or scale of the generated data sets.

The study [25], which presented its proposal in real time, focused on column-oriented databases. The proposed solution to creating RTOLAP cubes involved the development of R-Store, a scalable distributed system that extends the MapReduce framework and uses HBase as the underlying storage system. The system architecture includes a distributed key–value store, a streaming system to maintain the real-time data cube, a MapReduce system for processing large-scale OLAP queries, and a MetaStore for storing global variables and configurations. The solution efficiently scans real-time data, maintains the data cube, and processes real-time queries based on an adaptive algorithm and cost model. It also includes techniques for caching the data cube result and integrating streaming MapReduce for faster data cube updates.

5.6. R6—The Computational Resources Required to Implement the Proposed Models

The hardware configurations varied across studies within each NoSQL type category, reflecting differences in the number of nodes, RAM per node, CPU specifications, disk capacities, and network speeds. Table 3 presents consolidated information regarding the hardware and software utilized for each proposal categorized by NoSQL type. When considering the variety of operating systems utilized across different types of NoSQL databases, the following points are notable:

- **CentOS** was used the most in several studies of the column-oriented and document-oriented NoSQL types.
- **Ubuntu** was also observed in multiple studies, mainly in the column-oriented, graph, and key–value NoSQL types.
- **Windows** was mentioned in two specific studies of the column-oriented and graph NoSQL types.
- **Debian** appeared in a study of the document-oriented NoSQL type.

Table 3. Hardware and software specifications by NoSQL type.

NoSQL Type	Study	Nodes	RAM per Node	CPU per Node	Disk	Network	Operating System	OLAP System
Column	[40]	3	1 GB	Intel Core i5	20 GB	1 Gbps	CentOS 6.5	ROLAP
	[34]	5	24 GB	Intel (R) Xeon E3-1220	120 GB	NA	CentOS, CDH 5.5.1	ROLAP MOLAP
	[27]	NA	32 GB	Intel Xeon E5520, 2.27 GHz, 2 CPUs, 4 cores	NA	NA	Squeeze-x64-xen-1.3	ROLAP
	[23]	15	4 GB	Intel-Core TMi3-3220 CPU 3.30 GHz	NA	100 Mbps	Ubuntu-12.10	MOLAP
	[38]	15	4 GB	Intel-Core TMi3-3220 CPU 3.30 GHz	NA	1 Gbps	Ubuntu-14.04	MOLAP
	[35]	3	8 GB	Intel Core i5-4670 p4-core CPU, 3.4 GHz	2 TB	1 Gbps	CentOS	MOLAP
	[42]	NA	NA	NA	NA	NA	NA	SOLAP
	[44]	NA	NA	NA	NA	NA	NA	C-OLAP
	[25]	144	8 GB	Intel X3430 2.4 GHz	2 × 500 GB	1 Gbps	CentOS 5.5	RTOLAP
	[22]	1	8 GB	Intel Core i3-2312M CPU 2.10 GHz	500 GB	NA	Windows 7	NA
	[33]	4	16 GB	Intel-Core i5-3330 quad-core CPU at 3.0 GHz	1 TB	1 Gbps	NA	NA
	[30]	3	8 GB	Intel-Core i5 x 4	2 TB	1 Gbps	NA	ROLAP
	[31]	25	8 GB	Intel-Core TMi5-3220M CPU 3.30 GHz	NA	NA	NA	NA
Document	[26]	3	8 GB	4 Intel-Core i5	2 TB	1 Gbps	NA	ROLAP
	[32]	3	8 GB	Intel-Core i5 x 4 CPU	2 TB	1 Gbps	CentOS	ROLAP
	[35]	3	8 GB	Intel-Core i5-4670 x 44-core CPU, 3.4 GHz	2 TB	1 Gbps	CentOS	MOLAP
	[37]	4	4 GB/8 GB	Intel-Core i3	500 GB/1 TB	1 Gbps	CentOS 7.1	SOLAP
	[30]	3	8 GB	Intel-Core i5 × 4	2 TB	1 Gbps	NA	ROLAP
	[45]	6	NA	Intel-Core i7-6600U 2.60GHz	NA	100 Mbps	Debian-10.10	NA
Graph	[36]	NA	8 GB	Intel Core i5-3210M	NA	NA	Windows 10	Ev-OLAP
	[43]	1	32 GB	NA	8 TB	NA	Ubuntu-18.04.01 LTS	MOLAP
	[39]	1	12 GB	i7-6700	250 GB	NA	NA	NA
	[46]	NA	16 GB	Intel-Core i7	1 TB	NA	NA	NA
Key–value	[41]	NA	NA	NA	NA	NA	NA	ROLAP MOLAP
	[24]	6	256 GB	Intel Xeon CPU E5-2640 2.50 GHz	NA	10 GBps	Ubuntu Server 12.04	MOLAP

Furthermore, it is important to note that the utilization of tools like Apache Kylin 1.6.0 and Saiku 3.7 in the implementation of ROLAP and MOLAP was explicitly mentioned in a single study [34], which proposed a solution for columnar databases. Apache Kylin [47] is an open-source, distributed in-memory analysis platform that allows the definition of multidimensional data models and offers high-performance OLAP analysis capabilities. On the other hand, Saiku [48], developed by Meteorite BI, is a platform that enables the efficient and visual implementation of ROLAP and MOLAP, providing an intuitive interface for exploring and analyzing multidimensional data. The study [34] highlights the importance of having specialized platforms for multidimensional data analysis in Big Data environments.

Table 3 also facilitates the identification of computational resources utilized with different OLAP system types. There were more details for ROLAP and MOLAP, but also, data were available for additional proposals, such as C-OLAP, Ev-OLAP, and RTOLAP.

Table 3 reveals that the majority of studies utilize three nodes in the cluster for testing, with one node being the minimum and 144 the maximum. Regarding RAM, the minimum usage per node was 1 GB, while the maximum reached 32 GB. Intel CPUs were the most commonly utilized. Disk capacities ranged from 120 GB to 8 TB per node, and network speeds ranged from 100 Mbps to 10 Gbps.

5.7. R7—Performance of the Models in Terms of Query Execution Times

Given that the studies vary greatly regarding the computational resources used, the software, the type of data set used for testing, and the complexity of the queries, among other factors, this question cannot be answered objectively. However, the most relevant data from the analyzed studies were collected, which are presented in Table 4.

Table 4. Summary of computational resources and processing time.

NOSQL Type	OLAP System	Study	Data Set	Number of Nodes	RAM per Node	CPU per Node	Time[s] for 1 GB	Time[s] for 10 GB	Time[s] for 100 GB	Time[s] for 1000 GB
Column	MOLAP	[23]	-	15	4 GB	Intel-Core TMi3-3220 CPU 3.30 GHz	-	-	9188	-
	MOLAP	[38]	-	15	4 GB	Intel-Core TMi3-3220 CPU 3.30 GHz	-	-	-	200
	MOLAP	[35]	-	3	8 GB	Intel Core i5-4670 p4-core CPU, 3.4 GHz	672	6643	69,025	-
	C-OLAP	[44]	TCP-H	-	-	-	-	2000	-	-
Document	ROLAP	[26]	-	3	8 GB	4 Intel-Core i5	540	6540	7920	-
	-	[45]	TCP-H	6	-	Intel-Core i7-6600U 2.60 GHz	135	258	-	-
Graph	MOLAP	[43]	TPC-DS	1	32 GB	-	50	-	-	-
Key–value	MOLAP	[24]	TPC-DS	6	256 GB	Intel Xeon CPU E5-2640 2.50 GHz	1000	4000	30,000	-

One notable observation from the selected studies is that multiple studies utilized the following data set for benchmarking:

- Transaction Processing Performance Council Decision Support (TPC-DS)
- Transaction Processing Performance Council Huge (TPC-H)
- Star Schema Benchmark (SSB) [32]

The graph model, combined with MOLAP, exhibited a relatively shorter query execution time of 50 s when utilizing 1 GB of data with TPC-DS. However, for a larger data set of 10 GB, the document-oriented model outperformed others, particularly when using TCP-H, although the specific OLAP system was not specified in the data. Moving to even larger data sets, such as 100 GB, document-oriented with ROLAP models demonstrated optimal performance. Interestingly, only one data point was available for the 1000 GB data set, which pertained to the column-oriented model with MOLAP, showcasing the shortest query time. It is crucial to note, though, that this single result represents the entire data available for this size.

6. Discussion

NoSQL databases are emerging as promising contenders for building agile and performant data warehouses. Their ability to handle semi-structured and unstructured data simplifies schema management and accommodates evolving data models. Additionally, horizontal scaling capabilities enable the efficient handling of massive data sets, which makes them ideal for Big Data scenarios where traditional relational databases may struggle.

This study identified a comprehensive overview of the open issues and future challenges of using OLAP over NoSQL, offering valuable insights for further research and development.

The findings in R1 illustrate how NoSQL, similar to RDBMSs, empowers users to interactively explore data across diverse dimensions and levels of detail through OLAP. Additionally, similar to its prevalence in RDBMS-based setups, ROLAP emerges as the most prevalent approach in NoSQL environments. MOLAP also enjoys considerable usage in the studies reviewed, and as indicated in R7, its implementation in column-oriented databases exhibits superior performance in terms of query execution times.

In this research, other proposals for types of OLAP systems for NoSQL, such as Ev-OLAP, C-OLAP, Ha-OLAP, and RTOLAP, were also found; they leverage the capabilities of

NoSQL to analyze large and complex data sets. ROLAP represented approximately 32% of the studies. MOLAP represented 28% of the total. SOLAP accounted for 8%, focusing on incorporating spatial elements into OLAP systems for multidimensional analysis with spatial considerations. C-OLAP, accounting for 4%, introduces a novel approach using column-oriented databases, enhancing OLAP operations with specific operators like MRC-Cube and SC-Cube for efficient data processing and aggregation. Ha-OLAP, accounting for 4%, adopts a simplified, multidimensional model, employing spatial data processing techniques for optimized OLAP operations. RTOLAP centers real-time data processing in column-oriented databases, ensuring timely analytics for decision making. Ev-OLAP, accounting for 4% of the total, proposes an evolutionary approach, adapting OLAP systems over graph databases and facilitating versioning and hierarchical analysis within a graph-based structure. The breakdown of these OLAP approaches provides a comprehensive view of their functionalities and applications across different database types.

Furthermore, ROLAP is the most popular approach in column-oriented and document-oriented databases, and it has also been studied in key–value databases. MOLAP is extensively researched in columnar databases, and it emerges as a preferred approach for key–value databases, followed by ROLAP. The versatility of these analysis models is evident across various data storage types, including graph databases for which MOLAP has also been proposed. However, SOLAP, RTOLAP, and C-OLAP have less representation compared to ROLAP and MOLAP, although SOLAP implementations are growing in column and document-oriented databases, reflecting increasing interest in spatial analysis. RTOLAP and C-OLAP are more associated with column-oriented databases, while Ev-OLAP stands out in graph databases.

According to R2, most of the identified proposals are centered on solutions for column-oriented NoSQL databases, followed by those tailored to document-oriented databases, giving us insight into trends and research opportunities in the graph and key–value types. Additionally, it was found that HBase is the most studied database, possibly because it is the default database in the Hadoop ecosystem. This suggests a preference for using frameworks that include a default DBMS, such as HBase in Hadoop. Furthermore, MongoDB and Neo4j emerged as the most researched DBMSs in the document-oriented and graph-oriented categories, respectively, which aligns with studies indicating their popularity in their respective categories.

The results of this study, depicted in R3, revealed that traditional OLAP schemas commonly used in RDBMS are also prevalent in NoSQL environments. The star method emerged as the most common across the four types of NoSQL databases considered in the collected studies, with 13 occurrences, accounting for 52% of the total instances. Following closely was the snowflake method, identified in six instances, making up 24% of the occurrences. The flat method, characterized by denormalized data storage, was observed in two instances, representing 8% of the total. The galaxy and geo-cube methods were less frequent, each appearing once and constituting 4% of the total instances. This finding suggests that traditional OLAP methods seamlessly adapt to new solutions, with the star and snowflake methods dominating the scene, while the flat, geo-cube, and galaxy methods are used minimally.

R4 indicates that the methodologies employed to structure OLAP cubes exhibit similarities to those utilized in RDBMSs. The analysis of OLAP data cubes in the NoSQL reveals a structured format consisting of dimensions, measures, and cells. Nevertheless, innovative proposals concerning cube operators, including CN-Cube, KV-Cube, MR-Cube, SC-Cube, MRC-Cube, and MC-Cube, among others, have emerged. Algorithms like MRLevel and MRPipeLevel have also been proposed for efficient computation of level-based, top-down data cubes. Some studies have indicated that building cubes in NoSQL takes more time compared to RDBMS, possibly due to the adaptation of solutions originally designed for RDBMS. However, this delay seems to be offset due to a reduction in query time execution.

CN-Cube was designed for column-oriented NoSQL databases, employing value positions and hash tables for OLAP cube computation. KV-Cube utilizes the BESS technique

for efficient cube computation with key–value data models. The graphoid method, although not an OLAP cube, serves as a data structure for graph modeling in OLAP systems. MR-Cube and SC-Cube use MapReduce and Apache Spark, respectively, for efficient OLAP cube computation, with SC-Cube leveraging in-memory processing to enhance performance.

MRC-Cube and MC-Cube, although similar in name, differ in their approaches to building OLAP cubes using MapReduce techniques. MRLevel focuses on level-based cube computation, while MRPipeLevel enhances performance using a distributed parallel-processing strategy. These advancements aim to optimize computation time and data scans, particularly for Big Data and high-dimensional cubes, showcasing ongoing innovations in OLAP operations over NoSQL.

The prominent finding in R5 among the 25 studies analyzed is the overwhelming focus on batch data analysis, with 92% of the proposals centered on this method. This indicates a strong preference within the research community for employing batch processing techniques in OLAP over NoSQL databases. Additionally, the study's observation of only one proposal, 4%, addressing real-time data analysis highlights a notable gap in research focus, suggesting a potential area for future exploration and development in the field.

The results from R6 shed light on the hardware configurations across different NoSQL types, underscoring significant variations in node counts, RAM, CPU specifications, disk capacities, and network speeds. Notably, CentOS emerged as the predominant choice across column-oriented and document-oriented NoSQL types, while Ubuntu is also widely utilized, particularly in column-oriented, graph, and key–value databases. Windows and Debian are less commonly mentioned but still present in specific studies.

Moreover, one study [34] highlighted the explicit mention of tools like Apache Kylin and Saiku in a single study focusing on columnar databases. These tools, known for their high-performance OLAP analysis capabilities, emphasize the importance of specialized platforms for effective, multidimensional data analysis in Big Data environments.

The computational characteristics vary significantly, making it challenging to conduct a direct comparison. Nonetheless, several noteworthy observations emerged. Firstly, the number of nodes ranged from 3 to 144, with single-node solutions being disregarded, as they did not constitute a cluster. Node counts of 3, 4, 5, 6, 15, 25, and 144 were observed. Regarding RAM, there was a wide spectrum, ranging from 1 GB to 256 GB per node. CPU specifications exhibited diversity, with the majority being Intel-based. Additionally, disk storage varied from 120 GB to 8 TB.

When R7 tried to compare the query execution times of OLAP cubes built in NoSQL, different data sets, queries, and, as mentioned earlier, hardware were noticed. The data sets used include TPC and SSB, which were originally designed for traditional relational databases but have been adapted to NoSQL systems. Studies demonstrated that lower query execution times were achieved with solutions implemented in NoSQL compared to RDBMSs.

The findings of this study underscore a dynamic landscape of performance across OLAP models and data set sizes. Initially, the graph model paired with MOLAP impressed, with a remarkably short query execution time of just 50 s, showcasing efficiency with smaller data sets like 1 GB in the TPC-DS context. However, as data scale up to 10 GB, the document-oriented model takes the lead, especially evident with TCP-H, despite the lack of clarity on the specific OLAP system used. This trend continues with larger data sets, notably 100 GB, for which document-oriented models paired with ROLAP exhibit top-tier performance.

Interestingly, the 1000 GB data set presents a unique scenario, with only one data point available, spotlighting the column-oriented model with MOLAP as boasting the shortest query time. Nonetheless, it is essential to note that this singular result encompasses the entirety of data accessible for this size, highlighting the need for broader data sets to draw comprehensive conclusions. Hence, it is advisable to carefully choose one of these proposals for implementation, considering factors such as the type of NoSQL database, the available hardware resources, and the unique characteristics of the data being handled.

The findings of this comprehensive analysis underscore the growing prominence of NoSQL databases as robust solutions for building agile and high-performance data warehouses. Notably, the study also highlights the need for careful consideration in selecting a suitable proposal for implementation, factoring in the type of NoSQL database, available hardware resources, and unique data characteristics involved, thereby paving the way for informed decision making and the efficient deployment of OLAP systems over NoSQL platforms.

7. Conclusions

This work has exhibited comprehensive, systematic mapping that sheds light on the evolving landscape of OLAP integration with NoSQL databases in Big Data environments. Seven key results were obtained, and they provide valuable insights into the state-of-the-art methodologies, trends, and challenges in this domain.

The study's findings underscore the growing significance of leveraging NoSQL databases for OLAP query analysis in response to the escalating demands of data analytics. By identifying the most commonly used NoSQL databases, prevalent OLAP modeling methods, and suggested models for batch and real-time processing, this research offers a roadmap for organizations navigating the integration of OLAP with NoSQL. Additionally, the exploration of computational resource requirements and performance benchmarks facilitates informed decision making and promotes advancements in Big Data analytics.

Moreover, the scientific research conducted in this area has focused on experimenting with various NoSQL databases and OLAP systems to compare their performance with RDBMS databases. The findings consistently show superior results with NoSQL databases, showcasing their scalability and high availability advantages. Notably, systems like ROLAP and MOLAP, as well as schema designs like Star and Snowflake, have demonstrated their applicability in these new paradigms. This highlights the potential to integrate the benefits of NoSQL, such as scalability and high availability, with well-established OLAP techniques, opening up new avenues for efficient data processing and analysis.

In future research, our focus will be on maintaining up-to-date, systematic mapping concerning methodologies for structuring OLAP cubes in NoSQL databases. Additionally, we aim to evaluate the proposed methodologies in order to provide readers with more precise comparisons, optimize hardware configurations, and conduct comprehensive performance evaluations across diverse data sets. By addressing these aspects, we can provide specific guidelines to harness the potential of NoSQL databases in order to revolutionize DW and analytics.

Author Contributions: Conceptualization, D.M.-M. and R.N.; methodology, D.M.-M.; validation, R.N.; formal analysis, A.A.-C.; investigation, A.A.-C.; writing—original draft preparation, D.M.-M. and R.N.; writing—review and editing, S.L.-M. and L.R.; supervision, D.M.-M. and L.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: We would like to express our sincere gratitude to the Vicerectorate of Research, Innovation, and Linkage of the Escuela Politécnica Nacional for supporting this research.

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

References

1. Agrawal, D.; Das, S.; El Abbadi, A. Big Data and cloud computing: Current state and future opportunities. In Proceedings of the International Conference on Extending Database Technology, Uppsala, Sweden, 21–24 March 2011; pp. 530–533.
2. Hai, B.; Quix, C.; Jarke, M. Data lake concept and systems: A survey. *arXiv* **2021**, arXiv:2106.09592.
3. Ghazali, D.P.S.; Latip, R.; Hussin, M.; Abd Wahab, M.H. A review data cube analysis method in big data environment. *ARPN J. Eng. Appl. Sci.* **2015**, *10*, 8525–8532.
4. Golfarelli, M.; Rizzi, S. *From Star Schemas to Big Data: 20 Years of Data Warehouse Research—A Comprehensive Guide through the Italian Database Research over the Last 25 Years*; Springer: Cham, Switzerland, 2017; pp. 93–107.

5. Cuzzocrea, A. Data Warehousing and OLAP over Big Data: A Survey of the State-of-the-art, Open Problems and Future Challenges. *Int. J. Bus. Process Integr. Manag.* **2015**, *7*, 372–377. [CrossRef]
6. Martinez-Mosquera, D.; Navarrete, R.; Lujan-Mora, S. Modeling and Management Big Data in Databases—A Systematic Literature Review. *Sustainability* **2020**, *12*, 634. [CrossRef]
7. Kitchenham, B. *Procedures for Performing Systematic Review*; Keele University: Newcastle, UK, 2004; Volume 33, pp. 1–26.
8. Chaudhuri, S.; Umeshwar, D. An overview of data warehousing and OLAP technology. *ACM Sigmod Rec.* **1997**, *26*, 65–74. [CrossRef]
9. Mongo, D.B. What Is NoSQL? Available online: https://www.mongodb.com/nosql-explained (accessed on 27 December 2023).
10. Thomsen, E. *Building Multidimensional Information Systems*; OLAP Solutions, Ed.; John Wiley & Sons Inc.: Hoboken, NJ, USA, 2002; pp. 1–688.
11. Chaudhuri, S.; Umeshwar, D. An overview of OLAP systems. In *Data Mining and Knowledge Discovery*; Springer: Boston, MA, USA, 1997; pp. 3–43.
12. Bimonte, S.; Tchounikine, A.; Miquel, M. Towards a spatial multidimensional model. In Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP, Bremen, Germany, 4–5 November 2005; pp. 39–46.
13. Inmon, W.H. *Building the Data Warehouse*; John Wiley & Sons Inc.: Hoboken, NJ, USA, 2005.
14. TPC Benchmarks. Available online: https://www.tpc.org/information/benchmarks5.asp (accessed on 8 April 2024).
15. Chavalier, M.; El Malki, M.; Kopliku, A.; Teste, O.; Tournier, R. Benchmark for OLAP on NoSQL Technologies Comparing NoSQL Multidimensional Data Warehousing Solutions. In Proceedings of the International Conference on Research Challenges in Information Science, Athens, Greece, 3–15 May 2015; pp. 1–6.
16. Aftab, U.; Farooq, G. Big Data Augmentation with Data Warehouse: A Survey. In Proceedings of the International Conference on Big Data, Seattle, DC, USA, 10–13 December 2018; pp. 2775–2784.
17. El Marlki, M.; Kopliku, A.; Sabir, E.; Teste, O. Benchmarking Big Data OLAP NoSQL Databases. Open Archive Toulouse Archive Ouverte. Available online: http://oatao.univ-toulouse.fr/24706 (accessed on 10 February 2022).
18. Pastor-Ramón, E.; Herrera-Peco, I.; Agirre, O.; García-Puente, M.; Morán, J.M. Improving the Reliability of Literature Reviews: Detection of Retracted Articles through Academic Search Engines. *Eur. J. Investig. Health Psychol. Educ.* **2022**, *12*, 458–464. [CrossRef] [PubMed]
19. Gusenbauer, M.; Haddaway, N.R. Which academic search systems are suitable for systematic reviews or meta-analyses? Evaluating retrieval qualities of Google Scholar, PubMed, and 26 other resources. *Res. Synth. Methods* **2020**, *11*, 181–217. [CrossRef] [PubMed]
20. State of the Art through Systematic Review. Available online: https://www.lapes.ufscar.br/resources/tools-1/start-1 (accessed on 8 April 2024).
21. Apache Software Foundation. Hive: A Warehouse Infrastructure for Hadoop. 2023. Available online: https://hive.apache.org (accessed on 4 January 2024).
22. Jiao, M.; Zhang, Y.; Sun, Y.; Wang, S.; Zhou, X. CDDTA-JOIN: One-Pass OLAP Algorithm for Column-Oriented Databases. In Proceedings of the Web Technologies and Applications, Kunming, China, 11–13 April 2012; pp. 448–459.
23. Dehdouh, K.; Bentayeb, F.; Boussaid, O.; Kabachi, N. Columnar NoSQL CUBE: Agregation operator for columnar NoSQL data warehouse. In Proceedings of the International Conference on Systems, Man, and Cybernetics, San Diego, CA, USA, 5–8 October 2014; pp. 3828–3833.
24. Zhao, H.; Xiaojun, Y. A multidimensional OLAP engine implementation in key-value database systems. In Proceedings of the Advancing Big Data Benchmarks: Proceedings of the 2013 Workshop Series on Big Data Benchmarking, Xi'an, China, 16–17 July 2013; pp. 155–170.
25. Li, F.; Ozsu, M.T.; Chen, G.; Ooi, B.C. R-Store: A scalable distributed system for supporting real-time analytics. In Proceedings of the IEEE International Conference on Data Engineering, Chicago, IL, USA, 31 March–4 April 2014; pp. 40–51.
26. Chavalier, M.; El Malki, M.; Kopliku, A.; Teste, O.; Tournier, R. How can we implement a multidimensional data warehouse using NoSQL? In Proceedings of the Enterprise Information Systems International Conference, Barcelona, Spain, 27–30 April 2015; pp. 108–130.
27. Cuzzocrea, A.; Moussa, R.; Laabidi, A. Taming Size and Cardinality of OLAP Data Cubes over Big Data. In Proceedings of the Data Analytics: International Conference on Databases, London, UK, 10–12 July 2017; pp. 113–125.
28. Lee, S.; Kim, J.; Moon, Y.S.; Lee, W. Efficient level-based top-down data cube computation using MapReduce. In Proceedings of the Transactions on Large-Scale Data- and Knowledge-Centered Systems XXI, Valencia, Spain, 1–4 September 2015; pp. 1–19.
29. Song, J.; Guo, C.; Wang, Z.; Zhang, Y.; Yu, G.; Pierson, J.M. HaOLAP: A Hadoop based OLAP system for big data. *J. Syst. Softw.* **2015**, *102*, 167–181. [CrossRef]
30. Chavalier, M.; El Malki, M.; Kopliku, A.; Teste, O.; Tournier, R. Implementing multidimensional data warehouses into NoSQL. In Proceedings of the International Conference on Enterprise Information System, Barcelona, Spain, 27–30 April 2015; pp. 172–183.
31. Dehdouh, K.; Bentayeb, F.; Boussaid, O.; Kabachi, N. Using the column oriented NoSQL model for implementing big data warehouses. In Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, Beijing, China, 1–4 December 2015; pp. 469–479.

32. Chavalier, M.; El Malki, M.; Kopliku, A.; Teste, O.; Tournier, R. Document-oriented data warehouses: Models and extended cuboids, extended cuboids in oriented document. In Proceedings of the International Conference on Research Challenges in Information Science, Grenoble, France, 1–3 June 2016; pp. 1–11.
33. Scabora, L.C.; Brito, J.J.; Ciferri, R.R.; de Aguiar Ciferri, C.D. Physical data warehouse design on NoSQL databases-OLAP query processing over HBase. In Proceedings of the 18th International Conference on Enterprise Information System, Rome, Italy, 25–28 April 2016; Volume 2, pp. 111–118.
34. Chen, W.; Wang, H.; Zhang, X.; Lin, Q. An optimized distributed OLAP system for big data. In Proceedings of the IEEE International Conference on Computational Intelligence and Applications, Beijing, China, 8–11 September 2017; pp. 36–40.
35. El Malki, M.; Kopliku, A.; Sabir, E.; Teste, O. Benchmarking big data OLAP nosql databases. In Proceedings of the Ubiquitous Networking: 4th International Symposium, Hammamet, Tunisia, 2–5 May 2018; pp. 82–94.
36. Guminska, E.; Zawadzka, T. EvOLAP graph–evolution and OLAP-aware graph data model. In *Beyond Databases, Architectures and Structures. International Conference Facing the Challenges of Data Proliferation and Growing Variety, Proceedings of the 14th International Conference, BDAS 2018, Poznan, Poland, 18–20 September 2018*; Springer: Cham, Swizerland, 2018; pp. 75–89.
37. Ferro, M.; Fragoso, R.; Fidalgo, R. Document-oriented geospatial data warehouse: An experimental evaluation of SOLAP queries. In Proceedings of the IEEE 21st Conference on Business Informatics, Moscow, Russia, 5–17 July 2019; pp. 47–56.
38. Dehdouh, K.; Boussaid, O.; Bentayeb, F. Big data warehouse: Building columnar nosql OLAP cubes. *Int. J. Decis. Support Syst. Technol.* **2020**, *12*, 1–24. [CrossRef]
39. Gómez, L.; Kuijpers, B.; Vaisman, A. Online analytical processsing on graph data. *Intell. Data Anal.* **2020**, *24*, 515–541. [CrossRef]
40. Jianmin, W.; Wenbin, Z.; Tongrang, F.; Shilong, Y.; Hongwei, L. An improved join-free snowflake schema for ETL and OLAP of data warehouse. *Concurr. Comput. Pract. Exp.* **2020**, *32*, e5519. [CrossRef]
41. Khalil, A.; Belaissaoui, M. Key-value data warehouse: Models and OLAP analysis. In Proceedings of the International Conference on Electronics, Control, Optimization and Computer Science, Las Vegas, NV, USA, 16–18 December 2020; pp. 1–6.
42. Yue, P.; Shangguan, B.; Zhang, M.; Gao, F.; Cao, Z.; Jiang, L.; Fang, Z. Geocube: Towards the Multi-Source Geospatial Data Cube in Big Data Era. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Waikoloa, HI, USA, 26 September–2 October 2020; pp. 3127–3130.
43. Akid, H.; Frey, G.; Ayed, M.B.; Lachiche, N. Performance of NoSQL Graph Implementations of Star vs. Snowflake Schemas. *IEEE Access* **2022**, *10*, 48603–48614. [CrossRef]
44. Khalil, A.; Mustapha, B. An Approach for Implementing Online Analytical Processing Systems under Column-Family Databases. *IAENG Int. J. Appl. Math.* **2023**, *53*, 1–9.
45. Khalil, A.; Mustapha, B.; Fouad, T. A Data Placement Strategy for Distributed Document-oriented Data Warehouse. *Int. J. Comput. Sci.* **2023**, *50*, 1–9.
46. Labzioui, R.; Khadija, L.; Mohammed, R. New Approach based on Association Rules for Building and Optimizing OLAP Cubes on Graphs. *Int. J. Adv. Comput. Sci. Appl.* **2023**, *14*, 997–1008. [CrossRef]
47. Kylin, K. Extreme OLAP Engine for Big Data. Available online: https://kylin.apache.org/ (accessed on 8 April 2024).
48. Saiku Big Data. Available online: https://www.meteorite.bi/products/saiku-big-data/ (accessed on 8 April 2024).