symmetry

MDPI

Review

A Comprehensive Study of HBase Storage Architecture—A Systematic Literature Review

Muhammad Umair Hassan [1,*,†], Irfan Yaqoob [2,†], Sidra Zulfiqar [3,†] and Ibrahim A. Hameed [1,*]

[1] Department of ICT and Natural Sciences, Norwegian University of Science and Technology (NTNU), Larsgårdsvegen 2, 6009 Ålesund, Norway

[2] School of Information Science and Engineering, University of Jinan, Jinan 250022, China; 201824100007@mail.ujn.edu.cn

[3] Department of Computer Science, University of Okara, Okara 56300, Pakistan; ms160400838@vu.edu.pk

* Correspondence: muhammad.u.hassan@ntnu.no (M.U.H.); ibib@ntnu.no (I.A.H.)

† Authors contributed equally.

Abstract: According to research, generally, 2.5 quintillion bytes of data are produced every day. About 90% of the world's data has been produced in the last two years alone. The amount of data is increasing immensely. There is a fight to use and store this tremendous information effectively. HBase is the top option for storing huge data. HBase has been selected for several purposes, including its scalability, efficiency, strong consistency support, and the capacity to support a broad range of data models. This paper seeks to define, taxonomically classify, and systematically compare existing research on a broad range of storage technologies, methods, and data models based on HBase storage architecture's symmetry. We perform a systematic literature review on a number of published works proposed for HBase storage architecture. This research synthesis results in a knowledge base that helps understand which big data storage method is an effective one.

Keywords: HBase; big data; storage architecture; internet of things

check for updates

Citation: Hassan, M.U.; Yaqoob, I.; Zulfiqar, S.; Hameed, I.A. A Comprehensive Study of HBase Storage Architecture—A Systematic Literature Review. *Symmetry* **2021**, *13*, 109. https://doi.org/10.3390/sym13010109

Received: 10 December 2020
Accepted: 6 January 2021
Published: 9 January 2021

1. Introduction

One of the leading technical problems confronted by today's companies is to ensure that a massive amount of data is stored, manipulated, and recovered efficiently. Online services and social media are the prime cause of data creation nowadays. Facebook (https://www.facebook.com/) users generate 4 million likes every minute and upload 450 billion photos and videos per day [1]. Together, other social media apps, the internet of things (IoT), and geographic, vector space, electric power, wireless sensor, and power grids produce an immense amount of data, exceeding the scale of petabytes daily [1,2]. These data may include valuable information that current technologies sometimes fail to investigate properly. Some existing traditional systems extract those parts of data that the system can store and process [3,4]. Moreover, all the remaining data are wasted—most of the data stored in an unstructured manner using different languages and tools are not compatible.

Big data evolution is disclosing a severe systemic problem. A systemic problem is when the amount of data is increasing day by day, but they are not stored and processed efficiently. The reason is that the standard tools and techniques are not designed to manage and handle big data's complexity [5,6]. Conventional tools and techniques are unable to handle these complexities because data are growing in a huge volume and in terms of variety and substantial value, and includes different data types [7]. It is necessary to handle such a huge amount of data and its significant problems efficiently. It is undoubtedly a crucial challenge for a database designer to design a system according to big data's immense problems. Inevitably, designing a scheme according to big data issues is a crucial task for database designers.

Large companies in the transportation, weather forecasting, IoT and power sectors, and social websites, e.g., Facebook, Instagram (https://www.instagram.com/), Yahoo (https://www.yahoo.com/), etc., produce petabytes of data per day that include different types and formats [8,9]. These data are used for future decision making, and this decision immediately impacts the company's future. These data have a wide range of distinct data formats and types that traditional schemes may not store and process. Many problems are faced while refining a big dataset; the main problem is to guarantee adequate storage, rather than just processing or analyzing.

There are specific tools that developed over time to store and process big data accurately. Hadoop is one of the tools that can process structured, unstructured, and semi-structured colossal datasets. It has two main paradigms—the Hadoop distributed file system and Hadoop storage area—and map-reduce: a powerful processing tool that processes the stored data [8]. HBase is another big data refining tool developed on Hadoop and ran on top of the Hadoop Distributed File System (HDFS). Because the architecture of HDFS is rigid, it cannot change according to the dataset's parameters. The first time Facebook selected HBase to implement its new messaging platform was in November 2010 [10]. In 2010, Apache HBase began as a project by the company powerset out of a need to process the massive amount of natural language search data. Now the Apache HBase is a top-level project.

This study's core findings have provided a systematic literature review that illustrates the works directly associated with particular datasets. This paper seeks to define, taxonomically classify, and systematically compare existing research on a broad range of storage technologies, methods, and data models based on HBase storage architecture. We perform a systematic literature review on a number of published works proposed for HBase storage architecture's symmetry. This research synthesis results in a knowledge base that helps understand which big data storage method is an effective one.

1.1. Apache HBase

HBase is an open-source, distributed, multi-dimensional, and NoSQL database [10]. It is designed to achieve high throughput and low latency. It provides fast and random read/write functionality on substantial datasets. It provides the bloom filter data structure, which fulfills the requirement of fast and random read-write. The HBase runs on the top of HDFS and provides all capabilities of a large table to Hadoop. HBase stores files on HDFS. It has the capabilities to store and process a billion rows of data at a time. Due to the dynamic feature of HBase, its storage capacity can be increased at any time [11]. HBase leverages Hadoop infrastructure like HDFS and Zookeeper. Zookeeper co-ordinates with HMaster to handle the region server. A region server has a collection of regions. The region server is responsible for the handling, managing, and reading/writing functions for the region.

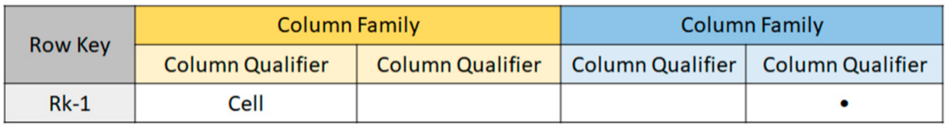
A region has data of all columns/qualifiers of column families. A HBase table can be divided into many regions where each region server handles a collection of regions. HBase became famous due to its features of fast and random read/write operation. Many companies adopted HBase because the current problem is handling big data with fast processing [11], and HBase is a good option.

The HBase framework offers several methods for extracting, manipulating, and storing big data [12]. This tool has progressed in recent years due to its dynamic architecture and promotes data integrity, scalability, fault-tolerance, and easy-to-use methods [12]. All of these variables have contributed to the popularity of Apache Hadoop, both in academia and in businesses. Most of the companies are using Apache HBase to store their dataset. They have changed the storage architecture according to the parameters of datasets.

1.2. Apache HBase Data Model

HBase is a NoSQL and column-oriented database. While it looks like a relational database that includes rows and columns, HBase is not a relational database. It is a column-oriented database, while the relational databases are row-oriented. Both databases store data differently on the hard disk [13]. Figure 1 is the illustration of the HBase table architecture while the components [12,13] of the HBase table are described below:

(1) **Table:** HBase tables are column-oriented, i.e., data are stored in column format.
(2) **Row Key:** It is the most crucial component of the HBase table. It is used for searching and retrieving data. It increases the speed of the searches.
(3) **Column Families:** The entire columns related to each other are combined and called column families, as shown in Figure 1.
(4) **Column Qualifiers:** Each column in the HBase table is known as the Column Qualifier.
(5) **Cell:** A cell is made up of row key, column family, and column qualifier. Actual data are stored in a cell. There are many versions of the cell.
(6) **Time Stamp:** A Time Stamp is made up of date and time. Whenever data are stored, they have a unique date and time. The timestamp is stored with the actual data, making it easy to search for a particular version of the data.



Row Key	Column Family		Column Family	
	Column Qualifier	Column Qualifier	Column Qualifier	Column Qualifier
Rk-1	Cell			•

Figure 1. HBase table architecture.

1.3. Contributions

Over the past years, the researchers proposed several methods to store different datasets on HBase in different ways [14–17]. Most of the research was published worldwide in journals and conferences. Some of the storage technique research studies are dataset specific, and some cover all types of datasets. Many publications make it difficult for companies to find an accurate and efficient technique to store datasets. Thus, there is a need to organize and classify all the proposed storage techniques based on HBase storage architecture. Table 1 below presents the research questions and objectives which we tried to investigate in this work.

Table 1. Research questions and their objectives.

No.	Research Questions	Objectives
RQ1	What are the primary datasets and storage techniques used by the researchers?	The aim is to investigate the primary focus of the researcher in datasets and storage techniques.
RQ2	What are the primary factors of selecting HBase storage architecture in various domains?	The aim is to investigate the domain that is working on HBase storage architecture.
RQ3	Are the proposed approaches application/data-specific or generic?	The aim is to identify whether the proposed techniques are data-specific or generic.

In this paper, we present a comprehensive systematic literature review on the HBase storage framework. Thus, our main contributions are:

- We have provided a systematic literature review (SLR) that shows which studies are directly associated with particular datasets.
- We have indicated the main research areas in HBase storage architecture and that there is still a need for improvements.
- The present study's taxonomy will lead to a comprehensive analysis of the more active storage framework areas.

The rest of the paper is organized as follows. Research methodology, motivation, and research questions are described in Section 2. Section 3 provides the answers to our research questions. A comprehensive discussion about selected studies is available in Section 4. The concluding remarks are available in Section 5.

2. Materials and Methods

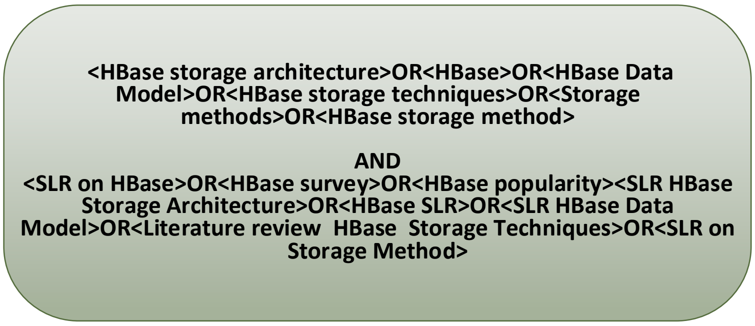
In this systematic literature review, we pursued the three-phase rules proposed by [14]. We conducted three tasks in the first phase: planning review, procedure review, and protocol review. The evaluation was performed in the second phase, as is the norm for research papers. In the third phase, we have reported the outcomes of the dissemination evaluation. All the phases are discussed in detail.

2.1. Search Criteria

After the research question was identified, search criteria were described based on the research question. There have always been two search criteria: (1) search string and (2) search sources.

2.1.1. Search String

This was described based on research questions. Since our primary focus was on reviewing the proposed storage techniques on HBase, our search string was based on the following criteria. Figure 2 shows the search string.



<HBase storage architecture>OR<HBase>OR<HBase Data Model>OR<HBase storage techniques>OR<Storage methods>OR<HBase storage method>

AND
<SLR on HBase>OR<HBase survey>OR<HBase popularity><SLR HBase Storage Architecture>OR<HBase SLR>OR<SLR HBase Data Model>OR<Literature review HBase Storage Techniques>OR<SLR on Storage Method>

Figure 2. Search string for searching required papers.

2.1.2. Search Sources

We selected ACM (Association for Computing Machinery society), ScienceDirect, and IEEEXplore (A digital research database by Institute of Electrical and Electronics Engineers society) for research papers, as shown in Table 2. These repositories provide the advanced level search and refine the search query by year, journal, and conference.

Table 2. List of selected research resources.

Resources	Hyperlinks
IEEEXplore	https://ieeexplore.ieee.org/
ACM	https://dl.acm.org/
ScienceDirect	https://www.sciencedirect.com/

2.2. Selection Strategy

We purely apply some variables (requirements) to separate the high and low-quality research papers in the selection strategy phase. These variables are based on inclusion and exclusion rules, as shown in Table 3. The inclusion rules were established based on study topics and questions introduced in the section on motivation and research questions. The information about paper selection stages is available in Table 4. These rules were applied to all the papers of stage 2 and papers were selected based on the inclusion rules. Table 5 shows all the stages according to the papers.

Table 3. Inclusion and exclusion rules.

	Inclusion Rules	Exclusion Rules
1	Those papers that address one research question at least.	Papers do not answer even a single defined research question.
2	Those papers have the ultimate goal of HBase storage architecture, data model, and storage techniques and do not concentrate on other HBase issues.	Papers do not use HBase for storage architecture; they only discuss the working of HBase architecture.
3	Those papers that are published in journals or conferences.	Papers are not published at conferences or in journals.
4		Papers do not have full text.
5		Short articles and discussions.

Table 4. Papers selection stages according to inclusion and exclusion rules.

Stages	Applied Inclusion and Exclusion Rules
Stage 1	Performed the defined research query on the selected sources.
Stage 2	Downloaded all the papers by just going through abstract, title, keywords, and those related to the research question.
Stage 3	Applied the inclusion and exclusion defined rules.
Stage 4	Deeply studied all the contents and finalized for systematic literature review (SLR).

Table 5. All papers according to stages.

Source	Stage 1	Stage 2	Stage 3	Stage 4
ScienceDirect	690	89	20	13
IEEEXplore	380	120	60	15
ACM	108	60	15	03
Total	1178	269	95	31

Search queries were conducted on 1 July 2019. All chosen documents are structured in excel by source, name, publication year, authors, and keywords and classified according to the investigated questions.

3. Results

In this section, we provide answers to all the research questions that we defined in Section 2. We intensely discuss and analyze these numerous selected research articles classified based on datasets such as transportation data, IoT, Internet of Vehicles (IoV), geographic spatial data, power grid, sensor data, and electric data. There are several features of the datasets that have different levels of scalability and implementation. Whether a specific work includes the indexing/row key, novel techniques and experimental results, we have provided all such main features for all 31 works discussed in this study.

- **RQ1: What are the primary datasets used by the research?**

RQ1 analyses several studies and storage techniques developed for specific datasets. Each storage method is based on specific characteristics of datasets, such as their volume environment, hot-spotting, load balancing, higher reading or writing processes, index, data integrity, scalability, and high availability. Most of the research has been carried out to store specific data on HBase effectively. Thus, these storage models are best for this specific dataset. We classify all the selected studies-based on datasets and relevant storage techniques in Table 6. RQ1 provides the sound details of storage techniques for several types of datasets. The discussions are also added for each dataset type. In Table 6, P refers to the paper number used for the systematic literature review conducted in this work (e.g., P1 refers to paper 1, P2 refers to paper 2, and so on.), while citation reference is available against each paper id. The dataset and storage technique columns include several state-of-the-art HBase architecture based datasets and storage methods, respectively.

Table 6. Numbers of studies based on storage technique and dataset.

No.	Dataset	Storage Technique
[P1] [11]	Remote sensing data	A distributed storage model for heterogeneous remote sensing data. Data were stored as a NetCDF (Network Common Data Form) file.
[P2] [15]	Wireless sensor data (IoT)	A two-layer distributed storage structure was designed: (1) distributed database to store metadata; (2) MySQL to store sensor data.
[P3] [16]	Wireless sensor data	Designed a real-time storage model: stored massive data quickly.
[P4] [17]	Power grid	A versatile event-driven data model for power grid multi-source data.
[P5] [18]	Internet of Vehicles (IoV)	A distributed storage and processing system on HBase.
[P6] [19]	Sensor data	Designed a data processing middleware for storing sensor data.
[P7] [20]	Remote sensor image data	The storage model of remote sensing image data was designed.
[P8] [21]	Geospatial data	Designed the HBase table to store Geospatial dataUsed the MapReduce to improve input/output (I/O) efficiency for geospatial data.
[P9] [22]	Geographic data/spatial-temporal	An efficient organization and storage proposed by adopting the strong expansibility for spatial-temporal information.
[P10] [23]	Spatial vector	Designed two models: (1) HBase storage schema with Z curve as row key; (2) HBase storage schema with geometry objects identifiers as the row key.
[P11] [24]	Sea wind and satellite image	Geographical information query system based on HBase storage model.
[P12] [25]	Spatial-temporal	Proposed an Automatic Identification System based on HBase and Spark (AISHS).
[P13] [26]	Location-based data	Proposed a multi-dimensional data storage model Index structure using quadtree over HBase.
[P14] [27]	Healthcare data	Designed a multi-table structure and three kinds of index tables.
[P15] [28]	Healthcare data	Designed an architecture for healthcare big data management and analysis. For personal health problem detection and vital real-time sign monitoring, a prototype system was constructed.
[P16] [29]	Equipment dataset	The data model of the equipment database was designed based on HBase. Equipment data's reading and writing processes were established.
[P17] [30]	Electricity power data	A distributed storage system on HBase for electric power data.
[P18] [31]	Electric consumption data	Storage model for electric power to manage Morocco school electricity consumption.
[P19] [32]	Sample data and alarm dataset in a power system	Designed a data storage structure for power data.
[P20] [33]	E-Commerce	Designed a non-primary key table; through this primary table query, the row key table is generated.
[P21] [34]	DNA	DNA sequence storage schemes based on HBase were designed.
[P22] [35]	E-Commerce	Devised a new distributed data storage framework for e-commerce data.
[P23] [36]	Marketing strategy	Designed a unique table schema on HBase to store business data.
[P24] [37]	Online data	Redesigned the HBase Architecture, used the thinner layer of a log-structured B+ tree to store actual data instead of HDFS.
[P25] [38]	Urban traffic data (Time-series)	Designed a distributed storage model for urban road traffic data.
[P26] [39]	Geography	Designed high-performance geographical database.
[P27] [40]	Document data	Designed an index framework to store versioned documents and processed them efficiently.
[P28] [41]	Bixi Data (Time-series dataset collected by the sensor)	Designed a Column Family Indexed Data Model.
[P29] [42]	Power grid	A secondary index on the event-driven storage model.
[P30] [43]	Traffic data (Time-series)	A scheme for monitoring data storage and processing; used the HBase to store timing monitoring data. Enhanced the field method to improve storage efficiency.
[P31] [44]	IoV	A row key structure design to store IoV data.

[P1] worked on remote sensing data to detect distant objects. Their work stored data in NetCDF file formats. Additionally, they used a metadata standard based on the ISO 19115-2:2009 metadata template, hence making a heterogenous file format. [P2] developed an HBase storage system for IoT. They constructed a two-layer distributed storage architecture that used the HBase and MySQL architecture to store sensor metadata. [P3] used the HBase architecture for wireless sensors' data to design a real-time storage model. The dynamics of HBase storage architecture are updated and optimized in their work. [P4] designed an event-driven approach for the HBase storage model. Their architecture can store multi-source data from power grids by using a join operation. They also proposed a scheme of virtual column families for resolving the multi-source data storage problems. [P5] advanced a distributed storage architecture for storing and processing the IoVs data. They integrated the HBase, Hadoop, and Spark to store and process the IoVs data.

[P6] proposed a massive sensor data oriented database (MSDB) architecture to solve the problems of data scattering, hotspot, and high concurrent transaction processing. They employed the MSDB strategy on the HBase medium. [P7] presented a scalable approach of big data technology, e.g., using the HBase storage database, they designed a remote sensing storage architecture, and then they combined a grid index with a Hibert curve to establish an indexed image data. [P8] proposed a geospatial data storage technique based on the MapReduce and HBase architecture. Their method's key components are: (1) a refinement method based on the MapReduce for accessing the geospatial data, and (2) designing HBase tables for storing and managing geospatial data. An efficient storage method was proposed by [P9], which adopted the HBase characteristics of reading and writing the data in real-time. They proposed encapsulated spatial-temporal meta-semantic object (MSO) information. [P10] developed two schemes of HBase storage architecture. The first is based on the Z-curve for row keys, and the other is based on geometric object identifiers.

A geography information query system was studied and implemented by [P11]. The system developed by them was based on HBase architecture, and their query systems can retrieve the aerial images and sea winds data using a graphical interface efficiently. [P12] presented a distributed system based on HBase and Spark architecture for storage and automatic identification system (AIS) data querying. They implemented a co-location based HBase region and Spark RDD (Resilient Distributed Dataset) partitions to ensure a spatial-temporal query system. [P13] presented a NoSQL based design for HBase architecture to store largescale data. Their approach is rather weak in this regard. This is because the data presented by them are available in column families, and they have stored them in relational database systems, which is a relatively old approach. A snowflake model proposed by [P14] was based on the HBase multi-table architecture. They also proposed a guideline for healthcare data processing. The test set was carried out on 750 million records to compare their proposed scheme with traditional tall-table HBase models. [P15] also worked on the healthcare datasets by proposing a big data management architecture. They constructed a prototype based on HBase, Spark MLLib, Hive, and Spark Streaming for real-time health problems detection.

A system to manage equipment data using HDFS file formats and HBase databases was designed by [P16]. They studied HBase's data management, reading and writing processes of HBase, and data modeling of equipment databases. [P17] proposed a distributed storage architecture for electric power data systems. Their system consists of HBase storage, status monitoring, data migrations models, and fragmentation technique. They also tuned HBase parameters for improving the efficiency of storage architectures. [P18] developed an interesting project for a Moroccan school to check the electricity consumption; they proposed a solution of big data selection, integration, and storage on an HBase model architecture. They selected several features, such as solar radiation, temperature, humidity, and electricity consumption, to propose a storage model. A distributed column-oriented approach was designed by [P19], based on a power application to store largescale real-time data. [P20] proposed a non-primary key table based on the fact that a secondary index will be developed, and using those secondary indexes, an efficient query was established for efficient storage retrieval. Their experiment was also based on HBase storage architecture.

The research on DNA sequence storage was proposed by [P21]. They used HBase as a storage model, and a pre-splitting strategy was applied to optimize DNA classification code. This helps in resolving the imbalanced dataset problems. [P22] designed a scalable storage framework by changing the read-write mode of hotspot data. They performed their experiments on S-HBase storage architecture instead of HBase architecture. A unique table scheme for storing the business data was proposed by [P23]. This results in efficient market data analysis, and by using the business indexes, it is easier to plot the floating population and store transactions. [P24] implemented an HBase-BDB (Berkley Database) that was operating over local volumes. Their proposed architecture uses a log-structure-based B+ tree scheme for key-value storing. [P25] adopted HBase of Hadoop for stirring largescale urban traffic data. They applied the framework of MapReduce in their proposed method for statistical analysis of traffic flow data.

A geographical database (G-HBase) was designed based on the HBase storage architecture by [P26]. The G-HBase indexes geographic data by using the Geohash function as the rowkey; G-HBase employs a Geohash rectangle partitioning for supporting spatial queries. [P27] worked on documents by proposing a new framework for indexing versioned documents. They also extended the keyword queries into general phrase queries. [P28] proposed a column family Indexed data model (CFIDM). In their proposed model, they convert the queried columns into multiple column families.

Many relational databases face the issues of malfunctioning when there are large-scale datasets. [P29] provided a refined HBase architecture that uses a secondary index scheme. Their proposed method saves storage space efficiently and also accelerates the query process. [P30] proposed a monitoring scheme for data storage and processing that was based on Hadoop architecture. Their proposed scheme uses the HBase database for storing timing monitoring data. They also worked on traffic data by using MapReduce distributed computing. Internet of Vehicles (IoV) is a unique concept being used in smart city applications. It provides meaningful information about city traffic data. [P31] proposed a massive traffic data querying technique using HBase architecture to analyze smart city data.

- **RQ2: What are the primary factors for selecting HBase storage architecture in various domains?**

We answer RQ2 by describing the storage techniques and methods of selected studies. We organize the selected studies based on sensor, IoT, IoV, Traffic, geoscience, healthcare, marketing, and document data, electric power data, and various other domains. Besides, the main features are presented in various tables.

3.1. Internet of Things (IoT)

The internet of things continually produces a large amount of data. There is a need to collect, process, store, analyze, and use these data. Humans produce big data while machines produce IoT-based data. IoT is the next stage of big data. Different NoSQL databases are used to manage such a tremendous amount of data. HBase is one such NoSQL database. The research proposed a broad range of storage models on HBase to store and process IoT, power, and sensor data. Table 7 shows the main features of selected studies in the IoT domain.

[P1] proposed a distributed data management architecture. This architecture handles remote sensing data in a heterogeneous environment. There are different file formats that HBase supports. Remote sensing data have a different characteristic, based on these characteristics; data are generated in different data types and formats. The authors selected a NetCDF file format to store the remote sensing in the proposed architecture. First, they converted all types of data in a NetCDF format then loaded it into the proposed architecture of HBase. An elastic search-based secondary index was built on HBase that provided the ability to search for data based on metadata content. This architecture is mainly divided into three components: the data integration component, data organization and storage component, and data index and search component. The first component extracts, transforms and loads remote sensing data automatically. The second component organizes the data remote sensing data in HBase, while the third component indexes the data and prepares it for search queries. [P2] introduced a lexicographical order for solving hotspot data center issues and high concurrent data transactions. A two-layer distributed storage structure was designed that includes two systems such as (1) a distributed database to store metadata and (2) MySQL to store sensor data. However, they have not mentioned the experimental evaluations in their work. [P3] worked on IoT-based sensors data. They proposed the HBase architecture for wireless sensors' data to design a real-time storage model. The dynamics of HBase storage architecture are updated and optimized in their work.

Table 7. Main features of selected studies for IoT domain.

No.	Storage Efficient	Scalable	Novel Technique	Indexing/Row Key	Improved Reading/Writing	Main Features	Experimental	Implementation
[P1]	✔	✔	✔	Elastic based secondary	✔	Managed heterogeneous remote sensing data.	✔	✔
[P2]	✔	✔	✔	Implement row key in lexicographic order	✔	Solved the issue of hotspot data scatter and high concurrent transactions.	Not mentioned (NM)	✔
[P3]	✔	✔		✖	✖	Quickly stored massive sensor data.	✔	✔
[P4]	✔	✔	✔	Event type and event time as the row key	✔	Novel virtual column family improved the join operation.	✔	✔
[P5]	✔	✔	✔	Single row key, complexed row key	✔	Overcame the single point of failure. Reliable data storage. Met the real-time computing requirements.	✔	✔
[P6]	✔	✔	✔	✖	✔	Redesigned the HBase table, solved the problem of the hotspot.	✔	✔
[P7]	✔	✔	✔	Grid index and the Hibert curve	✔	Stored imaged data. Effectively improved the data writing and query speed, and showed good scalability.	✖	✔
[P25]	✔	✔	✖	✔	✔	Stored the traffic data.	✔	✔
[P30]	✔	✔	✔	Hash prefix with a timestamp	✔	Improved the HBase table for time series data.	✖	✔
[P31]	✔	✔	✖	✔	✔	Improved query performance of time series data.	✔	✔

[P4] proposed a versatile Event-Driven data model on HBase for multi typed data of the power grid. Event-driven models defined each row as a unique record in the power grid and stored each single row in the HBase table structure. A novel virtual column family is used to resolve the compatibility issue between various data formats. HBase does not support the join operation. The join operation is integrated with the proposed model that improved the reading performance. A unique row key is designed to improve query performance. The row key is a combination of a timestamp and a novel virtual column. Because all the data are stored in a single column, there is no need to use the join operation.

[P5] proposed a storage system on HBase for handling the IoT. IoT based system architecture consists of three primary levels: the gateway layer, HBase managing layer, and the last one is global managing layer. The proposed system has two layers. One is the distributed database HBase to store the metadata of sensor data, and the second is MySQL to store the sensor data.

[P6] proposed a novel HBase data storage for wireless sensor networks. The storage model is designed for efficient real-time data processing. The main reason behind this storage model is that a real-time processing system can only fulfill the need for different users' queries one at a time. Ethnics primitives were used to integrate heterogeneous datasets stored in the HBase database. A real-time flow of data, stored in the cluster database, is used to satisfy the users' multiple needs that required data storage performance. Besides, text data of the historical stream is also migrated in a proposed storage model. A line keyword is used as a unique row key to store the data in HBase. In the last section, a discussion is conducted on when the storage space is insufficient.

The power grid usually generates enormous amounts of data from multi-sources with hundreds of complicated data types. Different complex data types are needed to store accurate results efficiently.

Sensor technologies generate a tremendous amount of data. How to efficiently store and process this sensor data is a hot research topic. [P6] proposed a data processing middleware to tackle the issue related to sensor data. This middleware was named massive sensor data processing (MSDB). The hotspot is a big issue while managing big data; the authors used a pre-splitting technique to solve the hotspot issue. To store sensor data, they redesigned the HBase table according to the characteristics of sensor data.

[P7] proposed an improved distributed storage and query for remote sensing data. The proposed system works in three steps. First, the storage model, according to the characteristics of remote sensing data, redesigned the HBase table. Second, a grid index and Hibert curve were combined to establish the index for the image data. The third is the last step in which MapReduce parallel processing was used to read and write remote sensing data.

The work [P25] used the MapReduce framework component of Hadoop for processing and statistical analysis of urban traffic data. The monitoring system generates a massive amount of time series monitoring data in real-time. It is an issue to store and process data efficiently. To tackle the issue of time series data in the cloud computing environment, [P30] proposed a scheme on Hadoop. This schema used HBase as a database to store the time-series data and MapReduce for improving the processing efficiency. The authors changed the HBase table structure according to the need for traffic data and their requirements. Furthermore, the authors designed a framework for storing or processing the traffic data. This program has two main modules: the HBase storage module and the MapReduce processing module.

The concept of the Internet-of-vehicles is derived from the Internet of Things. Primarily, the IoV's problem is to collect and process enormous amounts of massive information. [P5] originally designed a distributed HBase IoV data storage system. The proposed system has three main layers: the application layer, which provides the interface; the data processing layer, which processes the data according to the design format of HBase; and the third layer is the data storage layer, which stores the actual data. Moreover, three types of row keys are designed, such as single row key, complex row key, and secondary row key. This system ensured to solve the main three issues of IoV data. First, it can overcome a single point of failure; secondly, it ensures the real-time computing requirement; third, it ensures reliable data storage. In the same way, [P31] proposed a massive traffic data querying solution using HBase. The authors designed a schema for the HBase table, and, in particular, a row key structure was designed to store data effectively.

3.2. GeoScience

A traditional database is a good option for transactional operations but not suitable for large scale data analysis and processing. For large scale data analysis and processing, like geospatial data, an efficient storage model is required. [P8] proposed a storage model based on HBase and MapReduce to tackle geospatial data storage and processing. There are two important parts of the proposed storage model. First, to improve the efficiency of I/O, an advanced method for geospatial data was proposed on MapReduce. Secondly, the authors redesigned the table structure of HBase according to the requirement of Geospatial data. A unique row key was designed based on the Geohash string to access the stored geospatial data efficiently. Table 8 shows that the main feature of selected studies of the GeoScience domain.

Table 8. Main features of selected-studies of GeoScience domain.

No.	Storage Efficient	Scalable	Novel Technique	Indexing/Row Key	Improved Reading/Writing	Main Features	Experimental	Implementation
[P8]	✔	✔	✔	Geo Hash	✔	Improved I/O efficiency.	✔	✔
[P9]	✔	✔	✔	✔	✔	Improved query processing time.	✔	✖
[P10]	✔	✔	✔	Z curve or Geometry object identifier	✔	Improved query performance.	✔	✖
[P11]	✔	✔	✔	✔	✔	Improved write operation, and a log layer was used instead of HDFS.	✔	✔
[P12]	✔	✔	✔	Combination of B-order value and keyword	✔	Improved query efficiency and reduced the time consumption of spatial queries.	✔	✔
[P13]	✔	✔	✔	Index structure using quadtree over HBase	✔	Multi-dimensional data storage. Thousands of location updates per second	✖	✔
[P26]	✔	✖	✔	Geohash	✔	Decreased the spatial query time.	✔	✔

[P9] stores spatial-temporal information in HBase and mainly focuses on the classification and abstraction of it. The HBase storage model is built by constructing a spatial-temporal data table and designing the meta semantic object (MSO). MSO is an abstraction of geographic space. To support spatial queries for geographical databases, Hong Van Le et al. [P26] proposed a novel partitioning method.

The spatial vector data of the geographic information system (GIS) is increasing dramatically. Nowadays, storing and processing spatial vector data is a key step for GIS. [P10] proposed a NoSQL storage schema for spatial vector data on HBase. Two storage schemas were designed: one, Z schema, is a row key-based Z curve, and the second, ID schema, is a row key based on geometry object schema. In the Z curve, the schema row key is based on the Z curve. First, it divides the regions into two dimensions, then fills the gap by Z curve. Now, this Z curve is used as a row key. In the second schema, the polygon ID is used as a row key. In this schema, each column family has only two column qualifiers. The first column qualifier is used to store the spatial index, and the other column is used to store the co-ordinate information of the polygon. A series of experiments were performed to evaluate the proposed schema design.

In the advancement of cloud computing, its dynamic nature, powerful storage capacity, and other features create incentives to migrate the traditional system to cloud computing. This leads to the opening of a new research gate for researchers to solve the deployment issues from a traditional system to the cloud computing environment. [P11] designed a geographical information query system based on HBase. The query system made users retrieve the information of sea wind and satellite images by the graphical interface. This system has three layers: the operation and view layer, data processing layer, and storage layer. Data processing is improved by designing a row key as combined by the B-order value and keyword. The system was implemented on the National Geographic Public Welfare project. The scale of maritime traffic data is rapidly increasing. The AIS has emerged as a safe means of navigation for ships. [P12] proposed AIS Data management based on HBase and Spark. This distributed system provides efficient storage and near real-time queries to massive AIS data.

AIS HBase and spark store all data of a single ship in one region of HBase. This work is done with the help of HMaster. This way, it improves query performance and data access time. A secondary index was constructed on spatial-temporal attributes of massive AIS data, and co-location was created between HBase regions and Spark RDD to ensure efficient spatial-temporal query. The AISHS consists of main three-components. First, the data storage module; this module stores massive AIS data on HBase. It is also responsible for storing single ship data in a single region of the region server of HBase. The second component is the secondary index module; it creates the secondary index based on AIS's spatial-temporal attributes. The third module is a query processing module responsible for ensuring efficient query processing with a low cost and a high performance.

[P13] presented a multi-dimensional data storage model for location-based applications on HBase. The authors introduced a new index structured on HBase using the quadtree. The multi-dimensional data were based on the location that was inserted first and nearest neighbor queries were performed. In the end, response time was compared with the traditional database management systems.

3.3. Healthcare

The growth of healthcare data is increasing day by day with different data types and formats. [P14] proposed a multi-table structure on HBase for storing and processing healthcare data. On top of the multi-table structure, three types of index tables were designed. Each table has a single column family with various column qualifiers for the single medical field, as multiple tables were designed for different medical fields but with the same structure of the table. An author called this model a snowflake model architecture. It enables a more flexible scheme for designing three index tables. The proposed model mainly composed of a fact table and a set of dimension tables. The fact table has the actual data of the patient and the health status of the patient. At the same time, the dimensional table is the split part of the fact table. The use of dimension tables eliminated the extra cost of filter operation. Because only a part of the table can be accessed in the query, there is no need to query or scan the big full table. Results were evaluated in a real-time environment.

The volume of healthcare data will increase dramatically to 35 Zeta Bytes, as estimated by [28]. [P15] presented an architecture for healthcare, big data management, and analysis. The proposed architecture consists of five layers: data layer, data aggregation layer, analytics layer, information and exploration layer, and data governance layer. The data layer is concerned with different data formats such as unstructured data, semi-structured data, and structured data. Moreover, it provides the facility for data collection. The primary responsibilities of the data aggregation layer are (1) data extraction, (2) transformation, and (3) loading data into the storage architecture. Second, the analytics layer includes the main functionalities, e.g., (1) streaming, (2) data processing, (3) optimization, and (4) indexing. Thirdly, the primary tasks of the information exploration layer are (1) real-time monitoring, (2) reporting, and (3) clinical decision support. Finally, the data governance layer is responsible for metadata management.

Organizing the data is a crucial factor for any organization. [P16] organized the equipment data into three dimensions: equipment type, data format, and data type. They then classified the equipment into four methods: equipment type, equipment style, equipment class, and equipment. Table 9 shows the main features of selected studies of the healthcare domain.

Table 9. Main features of selected studies of the healthcare domain.

No.	Storage Efficient	Scalable	Novel Technique	Indexing/Row Key	Improved Reading/Writing	Main Features	Experimental	Implementation
[P14]	✔	✔	✔	Personal Identification Number (PIN)	Improved reading/writing by three index table	Multi-tables structure, simple or complex row key.	✔	✔
[P15]	✔	✔	✔	✔	NM	Detected potential health problems.	✔	✔
[P16]	✔	✔	✔	Equipment code is used as the row key	✔	Improved the reading/writing operation. Organized the equipment data into three dimensions. Supported fast retrieval of massive data.	NM	✔

3.4. Electrical Power

Electric power generates millions or billions of statuses, reading, debugging, and on and off conditions' data on a single dataset. It is real-time decision-making data at a specific time without a single error. [P17] proposed a distributed storage system on HBase to fulfill the electric power data. This storage system consists of various modules like the client, HBase database, status monitors, data migration modules, and data fragmentation modules. A common information model (CIM) is used to exchange data between modules and heterogeneous environments. CIM is a file format and is designed for more efficient data exchange. A little change is made in the table structure of HBase to store electric data with a unique row key. This model was implemented in a real-time environment and the result was evaluated. Table 10 shows the main features of selected studies of the electric power domain. Table 11 shows the main features of selected studies of different datasets, document data, and marketing data.

Table 10. Main features of selected studies of electric power domain.

No.	Storage Efficient	Scalable	Novel Technique	Indexing/Row Key	Improved Reading/Writing	Main Features	Experimental	Implementation
[P17]	✔	✔	✔	Combination of region key, name, id, and timestamp	✔	Improved writing operation	✔	✔
[P18]	✔	✔	✔	Combination of column and timestamp value	Decrease the cost of the expense of electrical consumption	School electric data storage	✔	✔
[P19]	✔	✔	✔	Sequential key	✖	Stored the massive power data quickly	✖	✔
[P28]	✔	✔	✔	✔	✔	Improved the multi-dimension query time	✔	✔
[P29]	✔	✔	✔	Event time used as the row key	✔	Improved the reading operation	✔	✔

Table 11. Main features of selected studies on document, marketing and other domains.

No.	Storage Efficient	Scalable	Novel Technique	Indexing/Row Key	Improved Reading/Writing	Main Features	Experimental	Implementation
Document Domain								
[P27]	✖		✔	✔	✔	Indexing technique on versioned document	✔	✔
Marketing Domain								
[P20]	✔	✔	✔	✔	✔	Improved the data processing by adding the index table in each region	✖	✔
[P22]	✔	✔	✔	✖	✔	Solved the issue of hotspot data. Improved storage balance.	✔	✔
[P23]	✔	✔	✔	✔	✔	An optimized table schema for suggesting marking strategy operations.	✖	✔
Different Domain								
[P21]	✔	✔	✔	DNA classification code used as the row key	✔	Solved the issue of hotspot data.	✖	✔
[P24]	✔	✔	✔	✔	✔	Changed the layered architecture of HBase.	✔	✔

The electrical consumption rate is different in various fields. It also depends on the meter. There are two types of meter: home field meter and industrial field meter. Various sectors expose electrical consumption expenses. The educational sector is at the top of electricity consumption due to their new practices and activities, e.g., electrical equipment usage, implementing complex scientific experiments, and organizing various events in different domains. [P18] proposed a storage system for electrical consumption forecasting in a Moroccan Engineering School. An actual storage model on HBase is proposed to manage the electricity cost; this helped to manage or decrease the electricity expenses. All the paraments like solar radiation, humidity, wind speed, electrical equipment, and temperature directly impact electrical consumption. The proposed system has three main components: data collection, data integration, and data storage. High fault tolerance and scalability are the main features of this system.

[P19] designed a massive data storage structure of power data. The designed architecture has three main layers: application layer, service layer, and storage layer. The application layer's concern is to provide an interface, and the service layer is a source of communication between application layers and the database layer. The storage layer holds the actual power data. Experiments have been performed in a real-time environment. [P28] stated that HBase is inefficient if users carry out multi-dimensional queries. Some existing approaches incur additional costs in writing performance or consistency maintenance, while others are restricted to specific applications. The authors proposed a column family indexed data model (CFIDM) to improve the presence of both known row key choir and composite row key queries. The queried column is converted into several families of columns. Values are partitioned in the specific column. Each partition is shown by a column family, which transforms the column family into an index without additional costs.

[P29] describes that indexing improves query processing speed. Many plans have been developed to improve HBase query performance, such as Hive and Index. Nevertheless, these indexing techniques are on columns. The author suggested a secondary indexing technique. It was implemented on the statistical basis of an event-driven data storage model on HBase. The author modifies the event-driven storage model's table structure to increase efficiency. In the HBase table design, a new layer of the virtual column family has been added in columns. The mechanism provides a minimum speed of 5.584× and a maximum speed of 571.360×, whereas speedup without index provides a minimum speed of 1.797× and a maximum speed of 8.581×.

3.5. Documents

The major problem in the big databases is to manage versioned documents [P27]. The authors of [P27] concentrate on the unique case of versioned indexing documents, in which those documents can improve over time, and each consists of several variants. The objective is to achieve a feasible compressed index for collecting versioned documents with limited redundancy while supporting phrase search queries with sensitive time limits for output [P27].

3.6. Marketing

With the advancement of e-commerce and e-Business, these areas require a scalable and consistent database. NoSQL provides features to address the issues related to cloud computing. However, hot spot data and unbalanced data distribution between nodes are still a severe bottleneck. [P22] devised a new database storage framework to solve the hotspot issue, with fair distribution of data between nodes on HBase. The authors distributed data in different nodes to address the hot spot issue. Moreover, it has improved the reading and writing operation.

It has become difficult to respond to fast-changing and uncertain trends for small business entrepreneurs. To store and process the business data, [P23] proposed a unique table schema to store business data for marketing strategy analysis.

[P20] describe that HBase supports the query based on the row key only if you do not know the row key; you will require a full scan table to obtain the requested data. That leads to cost and low efficient operation. The authors suggested the secondary indexes in a non-primary key table, and a query will be made on the row key by this secondary index table. This has enhanced query performance with high latency. The suggested secondary indexed table and the primary table are added to the same region server. Each region server will have the same indexed table and the primary table. If the user establishes an index or query, it is possible to complete the particular connection to the Region server. This approach helps in load balance, and at a time, only one Region Server is responsible for the query result.

3.7. Different Datasets

The demand for research and analysis of biological sequences, such as DNA sequences and protein sequences, is evolving. More efficient storage and biological sequence data applications are new challenges currently faced by developers and researchers. [P21] proposed a novel storage architecture for the DNA sequence of biomolecules. Based on the current biological sequence file's storage format characteristics, a new storage system was designed using the pre-splitting row key design strategy based on DNA division and the file indexing mechanism.

As for the process, there is a higher layer that can help in the process to store the data. The layer system has many issues due to the issue of integration and consistency between layers. [P24] proposed an elastic key-value store. It used the B+ log file structure for storage instead of HDFS, which can help solve the HBase layered system's issues. The proposed system was deployed by using the yahoo cloud serving Benchmark.

We have provided a brief introduction of different domains that are currently using the HBase storage architecture. Table 12 includes all those papers published during 2015–2019, while different domains used for such works are also available in Table 12. The sensor is the domain that was studied the most, while documents are the least studied domain. Geospatial comes next to the sensor; also, electric power, healthcare, and marketing domains were discussed in the above sections. The sensors domain works are the most discussed ones (~32%) that used HBase storage architecture as a basis for the storage model. It includes the NetCDF file formats, MySQL, real-time storage models, event-driven models for power grids, IoVs data, data processing middleware frameworks, and real-time computing environments. The selected studies based on IoT data also include remote sensing data, wireless sensors data, and urban traffic data.

Table 12. Groups based on domain.

Years	Worked	Domain
(2015 to 2019)	[P1–P7], [P25], [P30], [P31]	Sensor
	[P8–P13], [P26]	Geospatial
	[P14–P16]	Healthcare
	[P17–P19], P28], [P29]	Electric power
	[P21], [P24]	Different datasets
	[P27]	Documents
	[P20], [P22], [P23]	Marketing

The geospatial domain includes the works (~23%) that used geographic/geospatial datasets in the primary implementation. It mainly consists of spatial vectors, sea winds and satellite images, spatial-temporal and location-based data, and high-performance geography databases. The electric power domain is also dominant (~16%) over several discussed domains. It mainly includes electricity power data, electricity consumption data, alarm systems, Bixi-data (time-series data), and power grid data. The healthcare domain consists of largescale healthcare data (~10%) and equipment datasets. The data models are designed based on HBase model architectures. The marketing domain includes e-commerce and marketing strategic data (~10%) devised for distributed data storage frameworks by marketing experts. Documents (~3%), DNA sequencing, and online data storage frameworks based on HBase architecture (~6%) are also discussed in Table 12. An analysis of different datasets was also provided in Table 11. Figure 3 shows the selected studies based on the domain in the group.

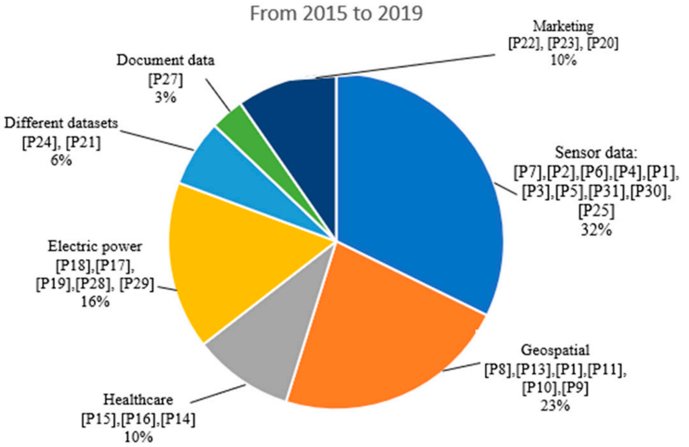


Figure 3. The number of works in a specific domain.

- **RQ3: Are the proposed approaches application/data-specific or generic?**

In order to answer RQ3, we categorize the preferred techniques in a graph, whether the techniques are particular to the application, data, or more generic. Figure 4 shows that 77% of data models from 2015 to 2019 are domain-specific; however, only 23% are generic models. A significant number of works used the specific models in their architectures, while a few architectures are based on the generic models.

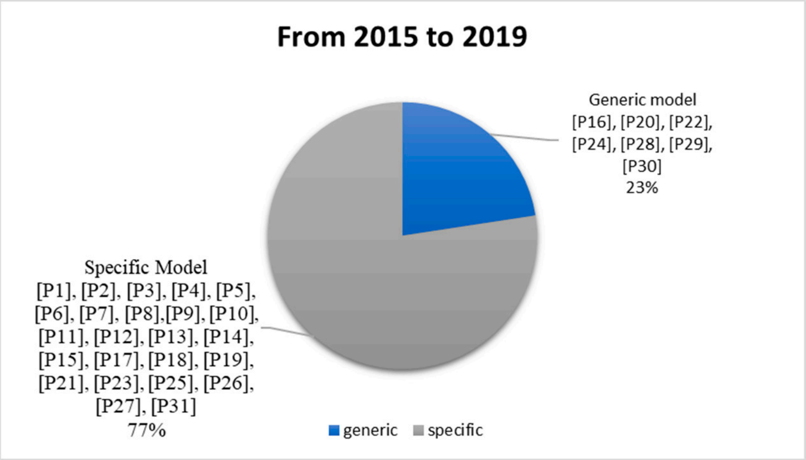


Figure 4. Generic and specific data model.

Approximately 77% of the works are application/data-specific, as mentioned in our work. [P1] used an elastic-based secondary key to manage heterogeneous remote sensing data, [P2] implemented a row key to solve hotspot data scattering issues. [P3] also used the sensor data without using any indexed row key, [P4] improved the virtual column family join operation, and [P5] is based on the single and complex row key HBase architecture.

The works by [P6–11] are data-dependent and have the applications of storing several data types. These works effectively improved data writing, query speed, and scalability. For example, [P6] worked on HBase table performance, [P7] developed a grid index and Hibert curve-based scalable method for HBase storage, [P8] improved I/O efficiency, [P9] adopted query processing improving strategy, [P10] also improved query performance, and [P11] improved the read/write operation of databases. [P12] introduced an efficient time cost-saving method for spatial queries. A multi-dimensional data storage method was introduced by [P13]. A significant improvement in multi-tables structure and potential threats detection in healthcare databases was developed by [P14] and [P15]. [P17–19] worked on upgrading the electric power domain by improving storage architectures and database writing operations.

The hotspot data issues were resolved by [P21], and an optimized table scheme for marketing data was suggested by [23]. [P25] worked in traffic sensor data, [P26] decreased the spatial query time of Geohash index key and [P27] introduced an indexing technique for versioned documents. An improvement in the query performance of time-series data was introduced by [P31].

Mainly, for generic approaches, the research was performed on equipment, e-commerce, online, Bixi data and traffic datasets. The generic approaches investigated the phenomenon of HBase storage architecture for different domains without being data specific. [P16] used the equipment code as the row key index, [P20] also used an indexing key to improve data processing, [P22] improved the storage imbalance problem without using any row key, [P24] used an indexing key and changed the architecture of HBase, and [P28–30] used event-based row keys and improved the multi-query operations. The works mentioned by [P28–30] used time-series data for improving query performance.

4. Discussion

We provided a significant number of reviews about state-of-the-art HBase architecture-based works used in different domains. The systematic literature review includes the paper from sensors, geospatial, healthcare, electric power, marketing, and different document and online database domains. We provided an investigation about HBase storage-based benchmarks which can be useful for many tech companies, and the provided research can help them establish their business models. The present work has also set a benchmark in the research of HBase storage architecture, and to the best of our knowledge, we are the first to provide a comprehensive review of HBase storage.

The features selected for comparing the selected works are storage efficiency, whether the work being studied is novel, indexing/row key, improved reading/writing capability, and whether it has experimental evaluation and implementation. We provided detailed results about all described features in different tables. Almost all the works discussed in the results section are efficient and scalable except [P26], which is not scalable and [P27] lacks in storage efficiency. Only [P25] and [P31] cannot provide novel techniques out of the 31 discussed works. The experimental results are not available for a some of the presented studies, e.g., [P7], [P13], [P16], [P19–P21], and [P23] have not provided experimental results. The implementation details are missing for works [P9] and [P10] only. All the discussed studies have several distinct features, which make this systematic literature review a sound benchmark for researchers to find any paper within the seven discussed domains.

Some works provided a literature review about big data processing and databases; for example, Habeeb et al. [45] provided a survey of real-time big data processing in anomaly detection applications. Neilson et al. [46] developed a survey about using big data applications in the transportation domain. They offered insights into the big data analytics for potential advanced transportation systems. Furthermore, there are still no reviews available on the selected research resources (available in Table 2), which are needed to provide a thorough insight into the HBase storage architecture. Hence, based on the presented study's provided literature, we can propose that a comprehensive survey on the HBase storage architecture is established in this work.

5. Conclusions

Big data is an emerging technology nowadays. Different tools and technologies have been used to store and manage big data based on different characteristics such as volume, velocity, veracity, value, etc. This study's core findings have provided a systematic literature review that illustrates the works directly associated with particular datasets. We have provided the taxonomical analysis of several state-of-the-art works regarding the HBase storage architecture. Our main objective in this investigation is to deeply analyze published research work on HBase storage architecture. This paper seeks to define, taxonomically classify, and systematically compare existing research on a broad range of storage technologies, methods, and data models on HBase. We performed a systematic literature review on a number of published works proposed for HBase storage architecture. This research synthesis results in a knowledge base that helps understand which big data storage method is an effective one. Simultaneously, the database designers can find the limitations of several methods while working on HBase storage architecture. They will also be able to identify that the proposed techniques are data-specific or generic.

In this paper, we conducted a systematic literature review of 31 papers published from 2015 to 2019. We selected these published papers based on inclusion and exclusion rules. Our primary focus was to analyze the proposed storage technique on HBase storage architecture. Additionally, we discussed the main features of this proposed storage architecture. Moreover, we grouped the storage architecture based on dataset and domain. According to the selected studies, 77% of the proposed techniques are dataset specific, while only 23% are generic techniques. The present work is a contribution to the database designers, researchers, and developers. It can be used as a study to find the literature specific to the HBase architecture, and businesses can assess the content based on our presented study.

The presented work also has some limitations, which will be considered in future work. We did not perform experiments for each study to evaluate its performance; instead, we only provided the literature based on the published content. We have only included seven domains in this existing literature review; however, several other domains can be studied in the future, e.g., transportation, manufacturing, and air quality monitoring sensor data, digital twins-driven smart monitoring data, as well as oil and gas, industry 4.0, and multimodal big data analytics. We only selected three major research resources for data collection; additionally, future works can consider using other libraries such as MDPI, Springer Link, and other journals/conferences.

In the future, we will focus on the execution time of several storage techniques, and we will also include the investigation of different data formats for storage techniques. We will try to add other domain challenges in our future work as well. The present study only focuses on the models studied based on the published materials, and the presented data were stored with a default HBase setting. Additionally, future work will also include the experimental evaluations of all discussed models.

Author Contributions: Conceptualization, M.U.H. and S.Z.; methodology, M.U.H. and S.Z.; formal analysis, I.Y. and S.Z.; writing—original draft preparation, M.U.H. and S.Z.; writing—review and editing, M.U.H.; visualization, I.Y.; supervision, I.A.H.; project administration, S.Z.; funding acquisition, M.U.H. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by Norwegian University of Science and Technology (NTNU) open access funding, Norway.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Acknowledgments: We are thankful to the anonymous reviewers for their valuable comments and suggestions in improving our manuscript. We are also thankful to the Norwegian University of Science and Technology (NTNU), Norway, to support open access.

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

References

1. Coughlin, T. 2019. Available online: https://www.seagate.com/in/en/our-story/data-age-2025/ (accessed on 13 February 2020).
2. Morris, T. 2019. Available online: https://www.business2community.com/big-data/19-data-and-analytics-predictions-through-2025-02178668 (accessed on 21 March 2020).
3. Zheng, K.; Fu, Y. Research on vector spatial data storage schema based on Hadoop platform. *Int. J. Database Theory Appl.* **2013**, *6*, 85–94. [CrossRef]
4. Um, J.H.; Lee, S.; Kim, T.H.; Jeong, C.H.; Song, S.K.; Jung, H. Distributed RDF store for efficient searching billions of triples based on Hadoop. *J. Supercomput.* **2016**, *72*, 1825–1840. [CrossRef]
5. Zhang, J.; Wu, G.; Hu, X.; Wu, X. A distributed cache for hadoop distributed file system in real-time cloud services. In Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing, Beijing, China, 20–23 September 2012; pp. 12–21.
6. Li, M.; Zhu, Z.; Chen, G. A scalable and high-efficiency discovery service using a new storage. In Proceedings of the 2013 IEEE 37th Annual Computer Software and Applications Conference, Kyoto, Japan, 22–26 July 2013; pp. 754–759.
7. Kim, M.; Choi, J.; Yoon, J. Development of the big data management system on national virtual power plant. In Proceedings of the 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Krakow, Poland, 4–6 November 2015; pp. 100–107.
8. Rathore, M.M.; Son, H.; Ahmad, A.; Paul, A.; Jeon, G. Real-time big data stream processing using GPU with spark over hadoop ecosystem. *Int. J. Parallel Program.* **2018**, *46*, 630–646. [CrossRef]
9. Smith, K. 2018. Available online: https://www.brandwatch.com/blog/facebook-statistics/ (accessed on 20 December 2019).
10. George, L. *HBase: The Definitive Guide: Random Access to Your Planet-Size Data*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2011.
11. Huang, X.; Wang, L.; Yan, J.; Deng, Z.; Wang, S.; Ma, Y. Towards Building a Distributed Data Management Architecture to Integrate Multi-Sources Remote Sensing Big Data. In Proceedings of the 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, UK, 28–30 June 2018; pp. 83–90.
12. Taylor, R.C. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinform.* **2010**, *11*, S1. [CrossRef] [PubMed]
13. Sinha, S. HBase Tutorial: HBase Introduction and FaceBook Case Study. 2018. Available online: https://www.edureka.co/blog/hbase-tutorial (accessed on 28 September 2020).
14. Okoli, C.; Schabram, K. *A Guide to Conducting a Systematic Literature Review of Information Systems Research.* 5 May 2010. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1954824 (accessed on 10 November 2020).
15. Zheng, Y.; Liu, C. HBase based storage system for the internet of things. In Proceedings of the 2016 4th International Conference on Machinery, Materials and Computing Technology, Hangzhou, China, 23–24 January 2016; pp. 484–487.

16. Li, X.; Li, Z.; Ma, X.; Liu, C. A novel HBase data storage in wireless sensor networks. *Eurasip J. Wirel. Commun. Netw.* **2017**, *2017*, 1–10. [CrossRef]
17. Liu, B.; Zhu, Y.; Wang, C.; Chen, Y.; Huang, T.; Shi, W.; Mao, Y. A versatile event-driven data model in hbase database for multi-source data of power grid. In Proceedings of the 2016 IEEE International Conference on Smart Cloud (SmartCloud), New York, NY, USA, 18–20 November 2016.
18. Chen, Z.; Chen, S.; Feng, X. A design of distributed storage and processing system for internet of vehicles. In Proceedings of the 2016 8th International Conference on Wireless Communications & Signal Processing (WCSP), Yangzhou, China, 13–15 October 2016; pp. 1–5.
19. Liu, B.; Huang, R.; Huang, T.; Yan, Y. MSDB: A massive sensor data processing middleware for HBase. In Proceedings of the 2017 IEEE Second International Conference on Data Science in Cyberspace (DSC), Shenzhen, China, 26–29 June 2017; pp. 450–456.
20. Jing, W.; Tian, D. An improved distributed storage and query for remote sensing data. *Procedia Comput. Sci.* **2018**, *129*, 238–247.
21. Gao, F.; Yue, P.; Wu, Z.; Zhang, M. Geospatial data storage based on HBase and MapReduce. In Proceedings of the 2017 6th International Conference on Agro-Geoinformatics, Fairfax, VA, USA, 7–10 August 2017; pp. 1–4.
22. Wang, K.; Liu, G.; Zhai, M.; Wang, Z.; Zhou, C. Building an efficient storage model of spatial-temporal information based on HBase. *J. Spat. Sci.* **2019**, *64*, 301–317. [CrossRef]
23. Wang, Y.; Li, C.; Li, M.; Liu, Z. HBase storage schemas for massive spatial vector data. *Clust. Comput.* **2017**, *20*, 3657–3666. [CrossRef]
24. Qian, L.; Yu, J.; Zhu, G.; Pang, H.; Mei, F.; Lu, W.; Mei, Z. Research and Implementation of Geography Information Query System Based on HBase. In *IOP Conference Series: Earth and Environmental Science*; IOP Publishing: Dalian, China, 2019; Volume 384, p. 012168.
25. Qin, J.; Ma, L.; Niu, J. Massive AIS Data Management Based on HBase and Spark. In Proceedings of the 2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Singapore, 21–23 July 2018; pp. 112–117.
26. Nitnaware, C.; Khan, A. A multi-dimensional data storage model for location based application on Hbase. In Proceedings of the 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, 19–20 March 2015; pp. 1–5.
27. Zhang, L.; Li, Q.; Li, Y.; Cai, Y. A Distributed Storage Model for Healthcare Big Data Designed on HBase. In Proceedings of the 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Honolulu, HI, USA, 18–21 July 2018; pp. 4101–4105.
28. Gui, H.; Zheng, R.; Ma, C.; Fan, H.; Xu, L. An architecture for healthcare big data management and analysis. In *International Conference on Health Information Science*; Springer: Cham, Switzerland, 2016; pp. 154–160.
29. Lei, R.A.O.; Fan-de, Y.A.N.G.; Xin-ming, L.I.; Dong, L.I.U. A Storage Model of Equipment Data Based on HBase. *Appl. Mech. Mater.* **2014**, *713–715*, 2418–2422.
30. Jin, J.; Song, A.; Gong, H.; Xue, Y.; Du, M.; Dong, F.; Luo, J. Distributed storage system for electric power data based on hbase. *Big Data Min. Anal.* **2018**, *1*, 324–334.
31. Daki, H.; El Hannani, A.; Ouahmane, H. HBase-based storage system for electrical consumption forecasting in a Moroccan engineering school. In Proceedings of the 2018 4th International Conference on Optimization and Applications (ICOA), Mohammedia, Morocco, 26–27 April 2018; pp. 1–6.
32. Yan, Z.J.; Sun, P.; Liu, X.M. An HBase-based platform for massive power data storage in power system. In *Advanced Materials Research*; Trans Tech Publications Ltd.: Stafa-Zurich, Switzerland, 2015; Volume 1070, pp. 739–744.
33. Zhengjun, P.; Lianfen, Z. Application and research of massive big data storage system based on HBase. In Proceedings of the 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, China, 20–22 April 2018; pp. 219–223.
34. Wen, S. Efficient DNA Sequences Storage Scheme based on HBase. In Proceedings of the 2018 International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE 2018), Qingdao, China, 30–31 March 2018; pp. 686–689.
35. Zhuang, H.; Lu, K.; Li, C.; Sun, M.; Chen, H.; Zhou, X. Design of a more scalable database system. In Proceedings of the 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, China, 4–7 May 2015; pp. 1213–1216.
36. Hong, S.; Cho, M.; Shin, S.; Um, J.H.; Seon, C.N.; Song, S.K. Optimizing hbase table scheme for marketing strategy suggestion. In Proceedings of the 2016 8th International Conference on Knowledge and Smart Technology (KST), Chiangmai, Thailand, 3–6 February 2016; pp. 313–316.
37. Saloustros, G.; Magoutis, K. Rethinking HBase: Design and implementation of an elastic key-value store over log-structured local volumes. In Proceedings of the 2015 14th International Symposium on Parallel and Distributed Computing, Limassol, Cyprus, 29 June–2 July 2015; pp. 225–234.
38. Zhu, L.; Li, Y. Distributed storage and analysis of massive urban road traffic flow data based on Hadoop. In Proceedings of the 2015 12th Web Information System and Application Conference (WISA), Jinan, China, 11–13 September 2015; pp. 75–78.
39. Van Le, H.; Takasu, A. G-hbase: A high performance geographical database based on hbase. *IEICE Trans. Inf. Syst.* **2018**, *101*, 1053–1065.
40. Kuo, C.T.; Hon, W.K. Practical index framework for efficient time-travel phrase queries on versioned documents. In Proceedings of the 2016 Data Compression Conference (DCC), Snowbird, UT, USA, 30 March–1 April 2016; pp. 556–565.

41. Cao, C.; Wang, W.; Zhang, Y.; Ma, X. Leveraging column family to improve multi-dimensional query performance in HBase. In Proceedings of the 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), Honolulu, CA, USA, 25–30 June 2017; pp. 106–113.
42. Wu, H.; Zhu, Y.; Wang, C.; Hou, J.; Li, M.; Xue, Q.; Mao, K. A performance-improved and storage-efficient secondary index for big data processing. In Proceedings of the 2017 IEEE International Conference on Smart Cloud (SmartCloud), New York, NY, USA, 3–5 November 2017; pp. 161–167.
43. Chi, Y.; Yang, Y.; Xu, P.; Li, G.; Li, S. Design and implementation of monitoring data storage and processing scheme based on distributed computing. In Proceedings of the 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA), Shanghai, China, 9–12 March 2018; pp. 206–211.
44. Xu, Y.; Zou, Q.; Feng, X. Efficient and Timely Querying of Massive Trajectory Data in Internet of Vehicles. In Proceedings of the 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Chengdu, China, 15–18 December 2016; pp. 293–298.
45. Habeeb, R.A.A.; Nasaruddin, F.; Gani, A.; Hashem, I.A.T.; Ahmed, E.; Imran, M. Real-time big data processing for anomaly detection: A survey. *Int. J. Inf. Manag.* **2019**, *45*, 289–307. [CrossRef]
46. Neilson, A.; Daniel, B.; Tjandra, S. Systematic review of the literature on big data in the transportation domain: Concepts and applications. *Big Data Res.* **2019**, *17*, 35–44. [CrossRef]