Research Article

Features and Conceptual Design of NoSQL Database Modeling

Richard E. Ochogwu[1], Matthew C. Okoronkwo[2], Colins Ifeanyi Osuji[3] and Francis Ekle Adoba[4]

[1,2,4]Department of Computer Science, University of Nigeria, Nsukka, Nigeria.
[3]Department of Computer Science, Federal University, Wukari, Nigeria.

Received 21 August 2024; Acceptance 5 September 2024; Published 11 September 2024.

Abstract

NoSQL databases has been growing in recognition and is expected to keep growing faster than that of the SQL market during the next few years, with the advent of big data technology, organizations and individuals are eager to understand why and how it fits into their developmental structures. Traditional relational databases have been the de facto choice since the 1970s, as they have been the sole option available for both developers and infrastructure teams. Traditional Relational Database Management Systems (RDBMS) use the ACID theorem for data consistency, whereas NoSQL Databases use a non-transactional approach called BASE. NoSQL, also known as not only SQL or non-relational databases, were specifically introduced to handle the rise in data types, data access, and data availability needs. This paper reviews the four types of NoSQL databases, namely, Document-oriented, Key-Value Pairs, Column-oriented and Graph. Results showed NoSQL data modeling for document-oriented databases which is similar to data modeling for traditional RDBMS during the conceptual and logical modeling phases. Findings also showed a database modeling operation for a car sales shop using the BASE theorem, and show concepts like denormalization, joins and aggregation.

***Keywords**:* NoSQL Databases, NoSQL Data Modeling, Denormalization.

Introduction

In the past 25 years, data has raised in a massive scale in diverse fields including software based medical rehabilitation system [1], and sports coaching [2]. According to the report of International Data Corporation (IDC), the over-all created data in the world will reach 44 ZB or trillion gigabytes during the time of 2013 to 2020 [3]. The increase of data volume (Big Data) during the last decade is attributed to a variety of data sources, such as social media, GPS data, sensor data, surveillance data, text documents, e-mails, etc. For example, the Internet of Things adds urgency for companies to be able to handle vast amounts of data [4].

Correspondence to: Richard E. Ochogwu, e-mail: Richard.emoche@gmail.com

How to Cite: Ochogwu et al. (2024). Features and Conceptual Design of NoSQL Database Modelling. *Annals of Industrial Technology*, **1(9)**. DOI: **10.5281/zenodo.13748137**

Data that was once considered too expensive to store, can now be captured, stored and processed. The business world is undergoing massive change as industry after industry shifts to the Digital Economy. It's an economy powered by the Internet and other 21st century technologies – the cloud, mobile, social media, and big data [5]. NoSQL was pioneered a decade ago by leading Internet companies – including Google, Amazon, Facebook, and LinkedIn – to overcome limitations of relational databases like Oracle, and MySQL for modern web applications [5]. With major enterprises now putting NoSQL solutions in key line of business applications that power major aspects of their company, many IT leaders not using NoSQL are interested in understanding why and how they can use the technology in their organizations [6]. NoSQL is a generic term used to refer to any data store or process that does not follow the traditional model of relational database management system [7]. There are 4 basic types of NoSQL databases includes key-value store, column store, document-based store, and graph-based store [8].
This paper provides a general implementation and modeling strategy for NoSQL by exploring the key reasons why enterprises are turning to NoSQL solutions and providing examples of how modern businesses can use the technology to model their databases.

Why NoSQL?

Although traditional Relational Database Management Systems (RDBMS) have existed for decades and are constantly being improved by the database vendors, RDBMS struggle to handle the large volumes of data [9]. However, a new category of database technology called NoSQL databases are able to support larger volumes of data by providing faster data access and cost savings. Basically, the cost savings and improved performance of NoSQL databases results from a physical architecture that includes the use of inexpensive commodity servers that leverage distributed processing. For example, according to [10], traditional RDBMS SAN storage costs average $30,000+ per terabyte, whereas storage for NoSQL databases average $1000 per terabyte. This dramatic reduction in storage cost has made it possible to store data that was previously considered too expensive. In addition to distributed processing and inexpensive hardware, NoSQL databases differ significantly on their approach to maintaining data integrity and consistency [11]. A more relaxed approach to data consistency helps NoSQL databases improve the performance of data storage. Because RDBMS highly value data integrity, they use the ACID theorem for data consistency which was presented in the early 1980's by Jim Grey. ACID is an acronym for Atomicity, Consistency, Isolation and Durability and supports the concept of a transaction.

Atomicity – ensures that all tasks with a transaction are performed completely (all or nothing).
Consistency – ensures that a transaction must leave the database in a consistent state at the end of the transaction.
Isolation – ensures that transactions are isolated and do not interfere with other transactions.
Durability – ensures that once the transaction is complete, the data will persist permanently, even in the event of a system failure. In contrast, NoSQL databases use the CAP theorem (Consistency, Availability and Partition Tolerance) for data consistency which was presented in 2000 by Eric Brewer at an ACM symposium [11]. The CAP Theorem states that of the three possible combinations of CAP, only two are available at a given point in time. In another words, NoSQL databases can have partition tolerance (in a distributed environment) and consistency or partition tolerance and availability, but not all three factors at the same time according to [12]. Cap theorem has evolved into what is now called BASE (Basically Available, Soft state and Eventual consistency).
Basically Available – means that data will be available, however the response from the database can be a failure to retrieve the data or the data retrieved may be in an inconsistent state.
Soft state – means that the data can change over time as the database seeks consistency.

Eventual consistency – means that sooner or later, the database will eventually become consistent.

Types of NoSQL

NoSQL is simply the term used to describe a family of databases that are all non-relational. NoSQL term for the first time was used in 1998 by Carlo Strozzi, who designed the relational database system without SQL implementation and named it Strozzi NoSQL [13]. The term NoSQL can mean either "No SQL systems" or the more commonly accepted translation of "Not only SQL," to emphasize the fact some systems might support SQL-like query languages [14]. The different types are discussed below;

i. Key-Value Stores

Stores information in form of matched pairs with only two columns permitted - the key (hashed key) and the value [15]. The values can be simple text or complex data types such as sets of data. Data must be retrieved via an exact match on the key. As you can imagine, such a structure is good for high read access. Due to a lack of referential integrity, the data integrity has to be managed by the front- -end applications. The well-known examples of that database model are as follows: Redis, BerkleyDB, LevelDB [16]. An example is shown in table 1 below:

Table 1. Key-value database model

Key	**Value**
Book Title	Java: How to Program
Author(s)	Paul Deitel, Harvey Deitel
Publication Date	2012
Edition	Ninth Edition

ii. Column Stores

They have a format of data storage that is very similar to RDBMS [12]. Although RDBMS tend to have simple data types and a predefined schema (structure), Column-oriented NoSQL databases provide much more flexibility. They can support complex data types, unstructured text and graphics (e.g. jpeg, gif, bmp, etc.) [17]. Examples of column-oriented databases: Cassandra, HBase, Bigtable.

Table 2. Column database model

Row Key	**Column Name**	
	Key	Key
	Value	Value
	Column Name	
	Key	Key
	Value	Value

iii. **Document Stores**

Document databases store text, media, and JSON (Java Script Object Notation) or XML data. The value in a row is a blob of the aforementioned data and can be retrieved using a key. If search is needed through multiple documents for a specific string, a document database should be used. CoucheDB and MongoDB are the most well-known representatives of the document database model. [16].

```
Document1 {
        "Field1": value1,
        "Field2": value2,
        "Field3": value3
        }
```

iv. **Graph Stores**

Graph databases are databases which store data in the form of a graph. The graph consists of nodes and edges, where nodes act as the objects and edges act as the relationship between the objects. The graph also consists of properties related to nodes. It uses a technique called index free adjacency meaning every node consists of a direct pointer which points to the adjacent node. Millions of records can be traversed using this technique [18]. HyperGraph, InfoGrid, Neo4 are typical examples of such database models.

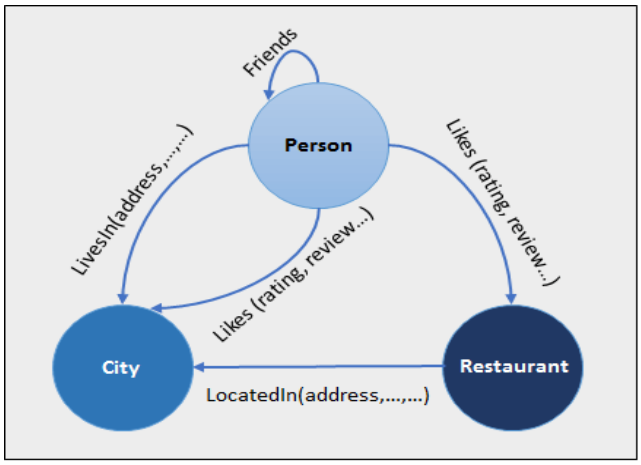

Figure 1. Graph database model

Ben Scofield rated different categories of NoSQL databases as follows [19]:

Table 3. Comparisons of different categories of NoSQL data models

Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–value store	high	high	high	none	variable (none)
Column-oriented store	high	high	moderate	low	minimal
Document-oriented store	high	variable (high)	high	low	variable (low)
Graph database	variable	variable	high	high	graph theory
Relational database	variable	variable	low	moderate	relational algebra

Benefits of NoSQL

NoSQL approach is a true standard in executing frameworks to deal with Big Data. Numerous organizations worldwide are at present applying high performance conveyed NoSQL arrangements [20]. When we are comparing to RDBMS and NoSQL databases than we found NoSQL are more scalable and provide high performance, and their data model addresses several issues that the RDBMS is not designed to address [12]. NoSQL database innovation offers advantages of versatility and accessibility through even scaling, replication, and improved information models, however particular usage must be picked right on time in the architecture configuration prepare [21]. A major difference from relational databases is the lack of explicit data scheme. NoSQL databases infer scheme from stored data, if it requires it at all, depending on which model was used [22]. The main benefit of using different data models is that they are very good at what they do. At the same time, don't force them to do something they aren't designed for. This means that it is of the upmost importance to understand and correctly use the data model when choosing NoSQL solutions [22].

Data Modeling Techniques for NoSQL

NoSQL data modeling often starts from the application-specific queries as opposed to relational modeling [23].

- Relational modeling is typically driven by the structure of available data. The main design theme is "What answers do I have?"
- NoSQL data modeling is typically driven by application-specific access patterns, i.e. the types of queries to be supported. The main design theme is "What questions do I have?" [23]

NoSQL data modeling often requires a deeper understanding of data structures and algorithms than relational database modeling does. Data modeling in NoSQL emphasizes more on data duplication and denormalization [23].

The following table shows a logical relationship of certain key definitions used in Logical, RDBMS and NoSQL data models.

Table 4. Logical relationship of key definitions

Conceptual / Logical	**RDBMS**	**NoSQL**
Entity	Table	Collection / Column Family
Entity instance	Row	Document / Row
Property	Column	Key / Column
Property of an entity instance	Cell Value	Field Value
Domain	Data type	Data type (Some NoSQL database no data type, all of value is plain text.)
Relationship	Constraint	Reference, Embedded, Additional table, row across multiple column families.
Key	Index	Index, Additional table, Reference
Unique identification	Primary Key	Row Key

Some Conceptual Techniques

In this section, we take a brief look at some basic principles of NoSQL data modeling.

1. **Denormalization** can be defined as the copying of the same data into multiple documents or tables in order to simplify/optimize query processing or to fit the user's data into a particular data model. Using denormalization, one can group all data that is needed to process a query in one place. This often means that for different query flows the same data will be accessed in different combinations. Hence, we need to duplicate data, which increases total data volume [24].

2. **Application Side Joins**
   Joins are rarely supported in NoSQL solutions. As a consequence of the "question-oriented" NoSQL nature, joins are often handled at design time as opposed to relational models where joins are handled at query execution time. Query time joins almost always mean a performance penalty, but in many cases, one can avoid joins using Denormalization and Aggregates, i.e. embedding nested entities [24].

3. **Aggregates**

Aggregates help minimize one-to-many relationships by means of nested entities and, consequently, reduction of joins [24]. Since NoSQL databases offer limited ACID support, using the aggregates technique, some of the ACID properties can be achieved [23].

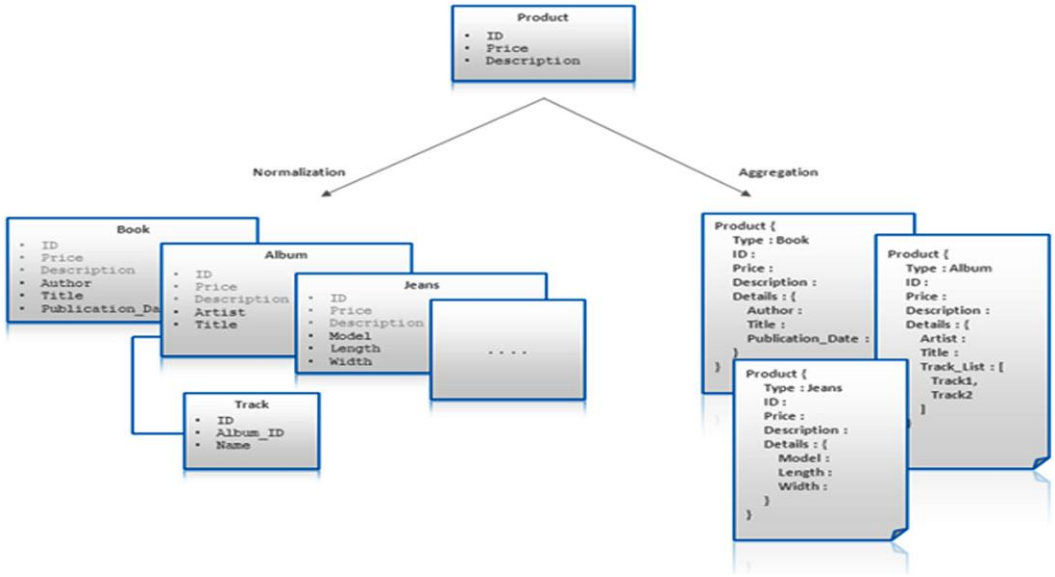

Figure 2. Diagram showing aggregation in NoSQL databases

4. Hierarchy modeling

Trees or even arbitrary graphs (with the aid of denormalization) can be modeled as a single record or document. This technique is efficient when the tree is accessed at once (for example, an entire tree of blog comments is fetched to show a page with a post).

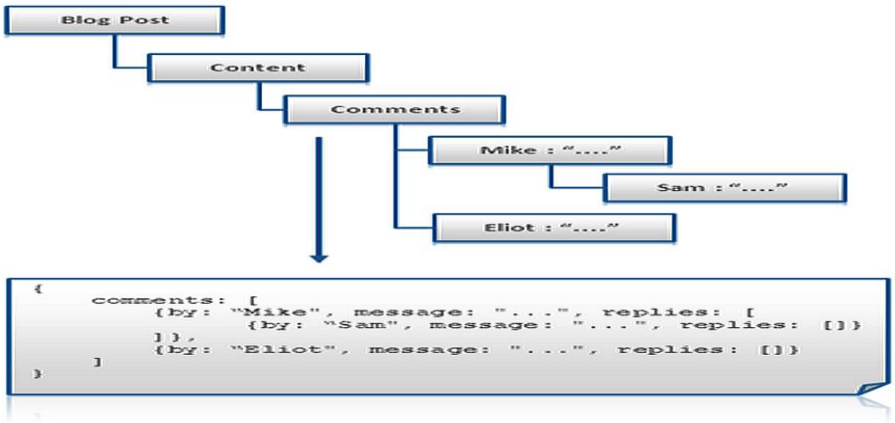

Figure 3. Diagram showing tree hierarchy model

5. Adjacency List

Adjacency Lists are a straightforward way of graph modeling – each node is modeled as an independent record that contains arrays of direct ancestors or descendants. It allows one to search for nodes by identifiers of their parents or children and, of course, to traverse a graph by doing one hop per query. This approach is usually inefficient for getting an entire subtree for a given node, for deep or wide traversals

Modeling Case Study: Car Sales Shop Database

In the following section, let us explore some of the modeling concepts described above for a Hospital Database example. For our case study, we shall consider modeling a Document database type NoSQL database like MongoDB which uses JSON documents as its internal storage.
The following diagram depicts the Entity Relationship (ER) model of our Car Sales Shop case study.

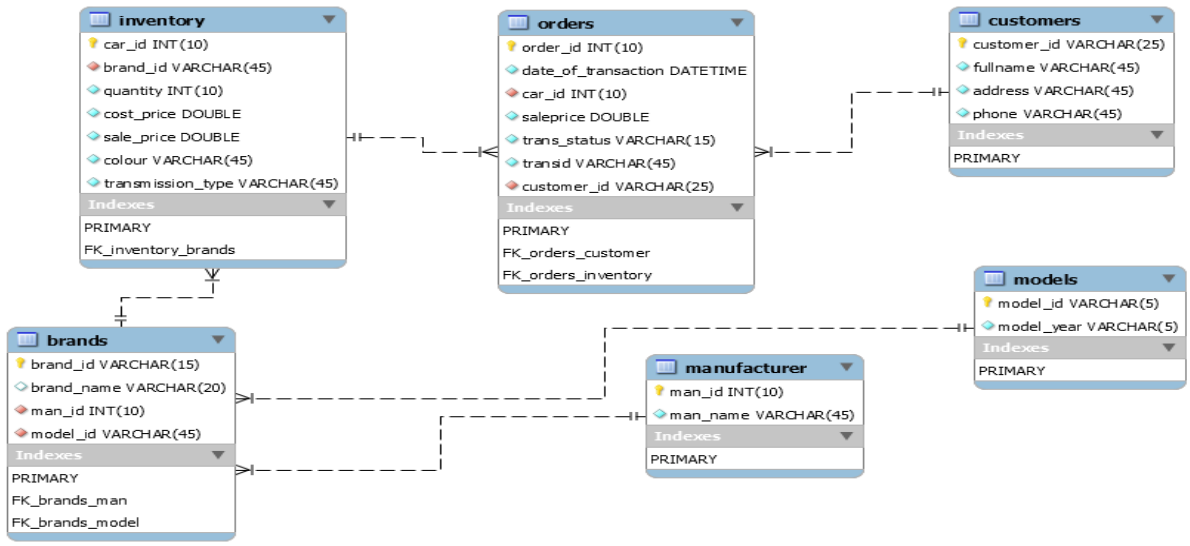

Figure 4. E-R model for our Car sales shop.

Denormalization

Some of the one-to-many relationships in the ER diagram could be denormalized into single structures.

- Models > Brands
- Manufacturer > Brands
- Brands > Inventory

The denormalized JSON model is shown below:

```
{
  "inventory": {
    "car_id": "",
    "brand_name": "",
    "quantity": "",
    "cost_price": "",
    "sale_price": "",
    "colour": "",
    "transmission_type": "",
    "man_name": "",
    "model_name": ""
  }
}
```

```
{
  "orders": {
    "order_id": "",
    "date_of_transaction": "",
    "car_id": "",
    "sale_price": "",
    "trans_status": "",
    "transid": "",
    "customer_id": ""
  }
}
```

```
{
  "customers": {
    "customer_id": "",
    "fullname": "",
    "address": "",
    "phone": ""
  }
}
```

Application Side Joins

Unlike traditional RDBMS, joins are rarely supported in NoSQL solutions. Application developers however overcome this with application side joins.
The transformed model is shown in the following diagram.

```
{
  "inventory": {
    "car_id": "",
    "brand_name": "",
    "quantity": "",
    "cost_price": "",
    "sale_price": "",
    "colour": "",
    "transmission_type": "",
    "man_name": "",
    "model_name": ""
  }, "orders": {
    "order_id": "",
    "date_of_transaction": "",
    "car_id": "",
    "sale_price": "",
    "trans_status": "",
    "transid": "",
    "customer_id": ""
  }, "customers": {
    "customer_id": "",
    "fullname": "",
    "address": "",
    "phone": ""
  }
}
```


Aggregates

One-to-many relationships in the ER model could be redesigned into aggregates.

• Inventory > Brands

• Brands > Models

• Brands > Manufacturer

The transformed model is shown below:

```
{
  "inventory": {
    "car_id":"",
    "quantity":"",
    "cost_price":"",
    "sale_price":"",
    "colour":"",
    "transmission_type":"",
    "man_name":"",
    "model_name":"",
    "brands":{
     "brand_id":"",
     "brand_name":""
    },
    "models":{
    "model_id":"",
    "model_year":""
   },
   "manufacturer":{
    "man_id":"",
    "man_year":""
   }
}
```

Conclusions

In this paper, we have discussed NoSQL database technologies. We categorized different NoSQL database models, we also talked about the CAP and BASE theorems and then we discussed key conceptual design principles with particular interest in document-based databases like MongoDB. NoSQL data modeling could prove an important cornerstone in software and database development strategies. Understanding the different data modeling techniques available and when to use which will help businesses and developers add greater depth to their programs and to set up a strong foundation for future initiatives.

References

[1] N. U. Ahamed, K. Sundaraj, R. B. Ahmad, M. Rahman, and A. Ali, A Framework for the Development of Measurement and Quality Assurance in Software-Based Medical Rehabilitation Systems, Procedia Engineering. 2012; 41: 53–60. https://doi.org/10.1016/j.pro-eng.2012.07.142.

[2] N. M. Khan, Realtime coaching system, U. S Patent No. 8,279,051, 2012.

[3] A. A. Safaei, Real-time processing of streaming big data, Real-Time System. 2017 53(1): 1–44. https://doi.org/10.1007/s11241-016-9257-0

[4] Devlin, B. (2014). Business un-intelligence: The marriage of BI and Big Data. ACM Webinar on June 17, 2014.

[5] Ravi Mayuram. (2016). Database Infrastructure for digital economy. Retrieved from https://www.itproportal.com/2016/07/03/database-infrastructure-for-the-digital-economy/

[6] http://www.odbms.org/wp-content/uploads/2014/03/Implementing_a_NoSQL_Strategy.pdf

[7] Tech Republic. 10 things you should know about NoSQL databases. http://www.techrepublic.com/blog/10-things/10-things-you-should-know-about-nosql-databases/. Accessed August 8, 2017.

[8] J. Bhogal and I. Choksi, Handling Big Data Using NoSQL. in Proceedings - IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2015, 393–398: IEEE.

[9] Mohan, C. (2013). History repeats itself: Sensible and Nonsen SQL aspects of the NoSQL hoopla. EDBT/ICDT 2013 Joint Conference, March 18–22, 2013, Genoa, Italy. ISBN: 978-1-4503-1597-5.

[10] Cloudera. (2014). Cloudera Big Data and Hadoop Sessions, May 21, 2014. Location: Hyatt Hotel, Denver, CO.

[11] Roe, C. (2012). ACID vs. BASE: The shifting pH of database transaction processing. Retrieved from http://www.dataversity.net/acid-vs-base-the-shifting-ph-of-database-transactionprocessing/

[12] Abramova, V., Bernardino, J., & Furtado, P. (2014). Which NoSQL database? A performance overview. Open Journal of Databases (OJDB), 1(2), 17-24.

[13] Bajpeyee R., Sinha S.P., Kumar V. (2015), Big Data: A Brief Investigation on NoSQL Databases, International Journal of Innovations & Advancement in Computer Science", Vol. 4, Issue 1, January, pp. 28-35.

[14] Keith D. F. (2018). A brief history of Non- Relational Databases. Retrieved from https://www.dataversity.net/a-brief-history-of-non-relational-databases/

[15] Moniruzzaman, A. B., & Hossain, S. A. (2013). NoSQL database: New era of databases for big data analytics - Classification, characteristics and comparison. International Journal of Database Theory and Application, 6(4), 1-14.

[16] Sadalage P., Fowler M. (2013), NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Pearson Education – Addison Wesley, Boston.

[17] Mason, R. T. (2015). NoSQL databases and data modeling techniques for a document-oriented NoSQL database. Proceedings of Informing Science & IT Education Conference (InSITE) 2015, 259-268. Retrieved from http://Proceedings.InformingScience.org/InSITE2015/InSITE15p259-268Mason1569.pdf

[18] Ameya N., Anil P., Dikshay P. (2013). Types of NoSQL databases and its comparisons with relational databases. International Journal of Applied Information Systems (IJAIS). ISSN: 2249-0868 Foundation of Computer Science FCS, New York, USA Volume 5 No.4, March 2013. www.ijais.org.

[19] NoSQL: Death to Relational Databases(?). https://www.slideshare.net/bscofield/nosql-codemash-2010. Retrieved 17 January 2020.

[20] Haseeb A., Pattun G. (2017). A review of NoSQL: Applications and Challenges. International Journal of Advanced Research in Computer Science. ISSN: 0976-5697. Volume 8 No.1, January - February 2017 203-207.

[21] Klein, J., Gorton, I., Ernst, N., Donohoe, P., Pham, K., & Matser, C. (2015, February).Performance evaluation of nosql databases: A case study. In Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems (pp. 5-10). ACM.

[22] Benefits of NoSQL. https://www.devbridge.com/articles/benefits-of-nosql/ retrieved on 08, January, 2020.
[23] NoSQL Data modeling Techniques. https://www.synechron.com/sites/default/files/whitepaper/NoSQL%20Data%20Modelling%20Techniques.pdf retrieved on 08, January, 2020.
[24] NoSql Data modeling Techniques. https://highlyscalable.wordpress.com/2012/03/01/nosql-data-modeling-techniques/ retrieved on 12, January, 2020.