TOP NEWSQL DATABASES AND FEATURES CLASSIFICATION

Ahmed Almassabi[1], Omar Bawazeer and Salahadin Adam[2]
[1]Department of Computer Science, Najran University, Najran, Saudi Arabia

[2]Department of Information and Computer Science, King Fahad University of Petroleum and Mineral, Dhahran, Saudi Arabia

ABSTRACT

Versatility of NewSQL databases is to achieve low latency constrains as well as to reduce cost commodity nodes. Out work emphasize on how big data is addressed through top NewSQL databases considering their features. This NewSQL databases paper conveys some of the top NewSQL databases [54] features collection considering high demand and usage. First part, around 11 NewSQL databases have been investigated for eliciting, comparing and examining their features so that they might assist to observe high hierarchy of NewSQL databases and to reveal their similarities and their differences. Our taxonomy involves four types categories in terms of how NewSQL databases handle, and process big data considering technologies are offered or supported. Advantages and disadvantages are conveyed in this survey for each of NewSQL databases. At second part, we register our findings based on several categories and aspects: first, by our first taxonomy which sees features characteristics are either functional or non-functional. A second taxonomy moved into another aspect regarding data integrity and data manipulation; we found data features classified based on supervised, semi-supervised, or unsupervised. Third taxonomy was about how diverse each single NewSQL database can deal with different types of databases. Surprisingly, Not only do NewSQL databases process regular (raw) data, but also they are stringent enough to afford diverse type of data such as historical and vertical distributed system, real-time, streaming, and timestamp databases. Thereby we release NewSQL databases are significant enough to survive and associate with other technologies to support other database types such as NoSQL, traditional, distributed system, and semi-relationship to be as our fourth taxonomy-based. We strive to visualize our results for the former categories and the latter using chart graph. Eventually, NewSQL databases motivate us to analyze its big data throughput and we could classify them into good data or bad data. We conclude this paper with couple suggestions in how to manage big data using Predictable Analytics and other techniques.

KEYWORDS

NewSQL, NoSQL, RDBMs. FF, Non-FF, and Big data.

1. INTRODUCTION

The database has been developed through many stages during its development life cycle and it has been commenced with Relational Databases Management Systems Generation. Then NoSQL databases generation is introduced and thereby we moved to NewSQL generation recently.

Relational Databases Management Systems Generation: The revolution of database systems started in 1960s by IBM [47]. Throughout that time until now, many Relational Database Management Systems have been emerged and developed to meet the computing and trading requirements. RDBMSs depend on relation tables and provide JOIN functionality. In addition, they use normalization for reducing redundancy and are engineered for data integrity. Despite all of these properties, RDBMSs have been suffered in several aspects like dealing with real time data and web applications [47]. In the 2000s, the web applications arise and the need to connect many concurrent users online at the same time made a real problem to RDBMSs. So, many of them tried

to grow their DBMS in vertical scale by using more powerful machines [47]. However, this procedure will increase the performance. But it requires stopping the databases systems to move them to the powerful machine and that is not acceptable for the real-time and web applications. As a result of these problems and others, the NoSQL was appeared in the late mid of 2000s.

NOSQL Generation: In 1988, NoSQL was first appeared for the relational databases that are not including SQL interfaces [44]. However, in 2009, NoSQL was reintroduced for some types of web-scale databases [44]. NoSQL databases appeared to provide more suitable solutions for many applications while Relational Databases Management Systems (RDBMS) give less flexibility, performance, and processing speed [44].

NoSQL stands for Not Only Structured Query Language [45]. The major purposes behind endorsed NoSQL databases are the flexibility of their architecture, handling huge quantity of multimedia capability, social media, word processing, emails, and other unstructured databases [44]. The authors in [46] mentioned that there are many features of NoSQL like high scalability and reliability, very simple data model, very simple (primitive) query language, lack of mechanism for handling and managing data consistency and integrity constraints maintenance (e.g., foreign keys), and almost no support for security at the database level.

One of the main differences between NoSQL an RDBMS is that NoSQL does not advocate Atomicity, Consistency, Isolation, and Durability (ACID). It supports BASE (Basic Availability, Soft state, and Eventual consistency) [48]. In contrast to RDBMS, NoSQL is non-relational databases, so it cannot support the applications that are already written for the previous generation [48]. Since NoSQL does not support OLAP and to meet the real-time features and ACID feature in one DBMS, the NewSQL generation has emerged.

NEW SQL Generation: NewSQL is a third generation of databases systems, a modern class of RDBMS that combined the OLTP and the high performance of NoSQL [47] [48] [49][50][51]. In 2011, NewSQL term was used by Matthew Aslett paper that discussed the need for new database systems [52]. Many systems that require more scalability such as financial enterprise systems, it cannot use NoSQL because it does not support ACID so it has not strong consistency and transactional requirements as RDBMS [47][48][52][53]. So, this paper is prepared to discover the features of the newest databases systems, and according to that features we classify and categorize the NewSQL databases. To start our work, we select 13 of the top NewSQL databases [54] and collect 35 features of databases systems (see appendixes A and B) to study them through all of these NewSQL databases.

Functions	RDBMS	NoSQL	NewSQL
SQL	Supported	Not supported	Supported
Machine dependency	Singe machine	Multi- machine/Distributed	Multi- machine/Distributed
DBMS type	Relational	Non- relational	Relational
Schema	Table	Key-value, column store, document store	Both
Storage	On disk + cache	On disk + cache	On disk + cache
Properties support	ACID	CAP through BASE	ACID
Horizontal scalability	Not supported	Supported	Supported
Query Complexity	Low	High	Very High
Security concern	Very high	Low	Low
Big volume	Less performance	Fully supported	Fully supported
OLTP	Not fully supported	Supported	Fully supported
Cloud support	Not fully supported	Supported	Fully supported

Recently, several NewSQL and No SQL databases have been presented to address data with different manner regardless of tables and structured query language (SQL) statements of relational

database [13]. Getting involved to know the power of NewSQL is worthwhile for all technical or academia environments. Students need to enhance their knowledge and experience with NewSQL to keep up to data with most upcoming technologies or system applications. Having the benefits without radically changing your application is one of the interesting features offered by most NewSQL databases to the new generation of database structures [35].

Low cost of NewSQL databases offers lower maintenance costs. Optimize and economize on your scale-out without making any changes to your existing applications. However, big data is defined as foremost about data volume well-known as large data set. DBMS where traditional (structured) databases are manipulated and managed whereas VLDB, very large Database, where unstructured databases are processed, stored and controlled as well [16].

Literature review

2. NEWSQL DATABASE TAXONOMY

2.1. Functional and Non-Function Features

After profoundly looking for over than 30 features embedded in most top NewSQL databases, and investigating these features to recognize and oversee how NewSQL databases manipulate data types such as batched, sparse, stream, real time, or historical data, we came up with two main categories embodying and modeling numerous features categories. First, Functional features whose operation transaction influences the functionality of the data process. Functional features not only will ensure the smooth of data, but also it reveals its high performance. Second, non-functional features attempt to ensure reliability and availability of data while processing. Operation transaction of non-function has impact on transaction behaviour's proficiency and security.

Taxonomy 1: illustrates main taxonomy of NewSQL databases in terms of features:

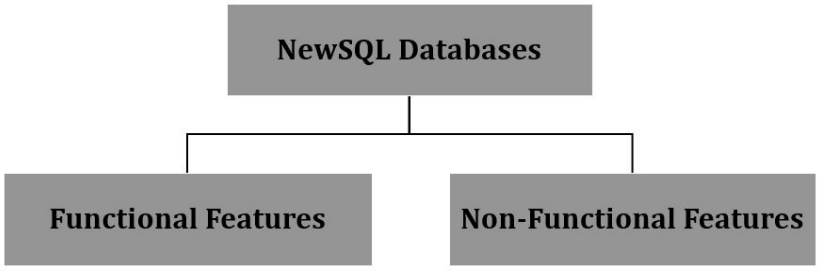
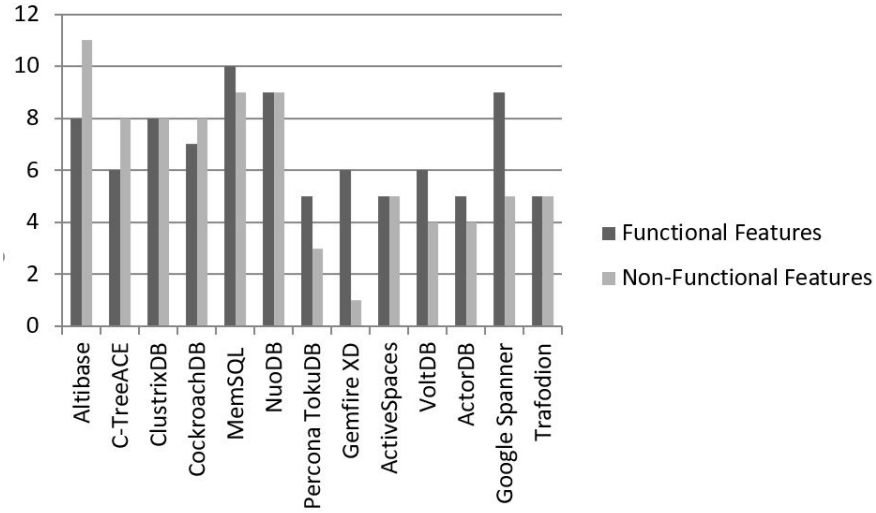


Figure 1: Taxonomy 1: displays features categories

In table 1 conveys how most non-functional features dominate most NewSQL database illustrating that how efficient the NewSQL databases to deal with big data with intensive care.

Table 2: shows features type's distribution.

NewSQL Databases	Functional Features	Non-Functional Features
Altibase	8	11
C-TreeACE	6	8
ClustrixDB	8	8
CockroachDB	7	8
MemSQL	10	9
NuoDB	9	9
Percona TokuDB	5	3
GemFire XD	6	1
ActiveSpaces	5	5
VoltDB	6	4
ActorDB	5	4
Google Spanner	9	5
Trafodion	5	5



In taxonomy 1, it discerns functional and not-functional features across selected NewSQL databases. The majority of NewSQL databases provides significant performance. High performance is required with compatibility for processing and storing big data. Performance accelerates the performance of a mission critical application by several times with in-memory processing [35]. For instance, Altibase' architecture permits utilization of memory table for high performance and disk table for economical storage [35]. Scalability is the database ability to boost significantly with respect of usability [13].

Altibase delivers high intensity of processed data through an in-memory database section and large storage capacity via one-disk database portion [35]. Altibase is flexible ANSI SQL compliant and ACID compliant. Moreover, Altibase mitigates cost through in-memory and on-disk DBMS. Experts are easily found to use it for. It provides efficient indexing, low latency, and high throughput designed and optimized in memory to set up a powerful engine. Reliability is supported for varied workloads, so critical data is accessed real time, real time access to historical data, and handles complex transaction through integrated data. Moreover, reliability assists easily bidirectional data movement between disk table and memory and joins between them. Deployment phase is flexible enough to have In-memory only, In-disk only, and Hybrid (both). A versatile

feature is multi durability level to address balance between persistence and performance. Productivity and Administration are outstanding features to assess technology to interoperate and migrate operations as well.

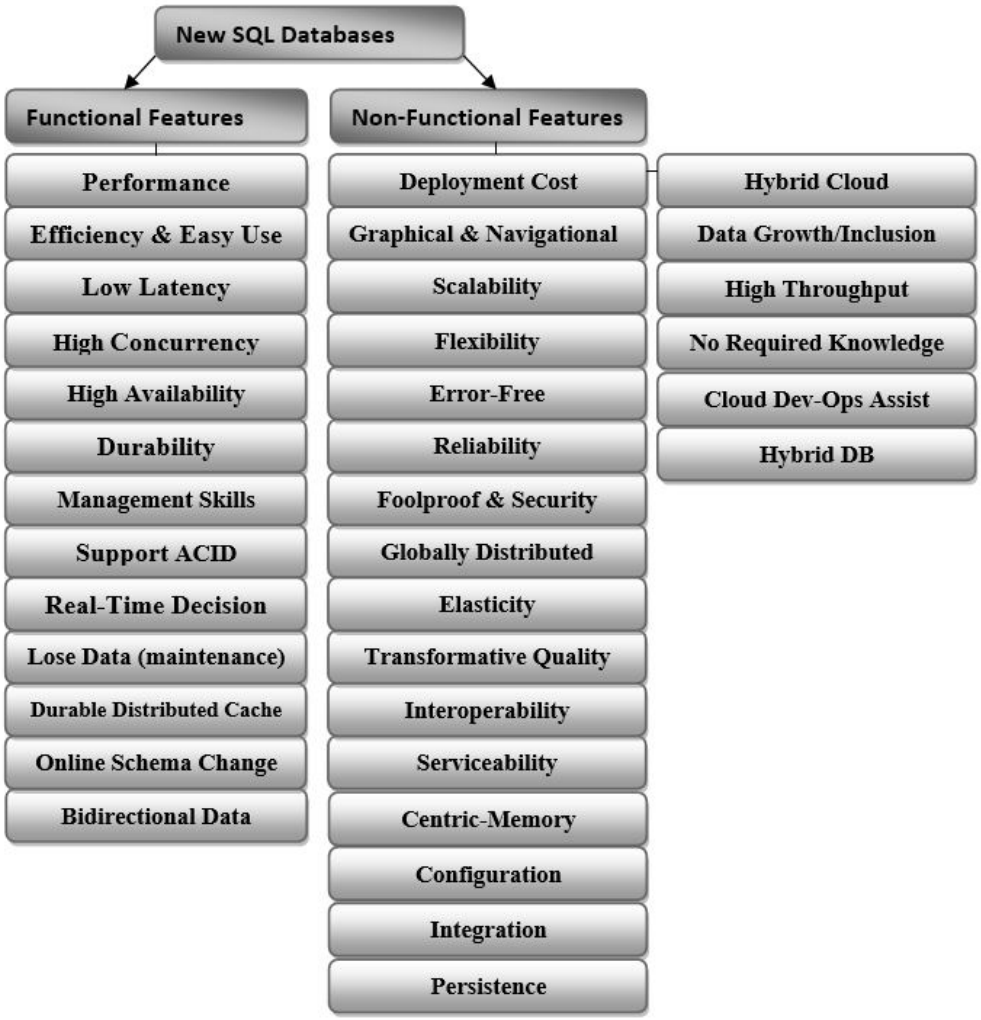


Figure 3: Result 1 conveys features distributions based on functional and non-functional features.

An in-memory cashing feature has been introduced 30 years ago to mitigate the need of I/O for either certain big data or relational databases (RDBMs) [4]. Cashing big data difficulty is low chosen patterns of big data to check data analysis strategies within rapid random access searching structure [4].

In taxonomy 1 reveals several functional features such as availability which offers a wide variety of features to make sure your application data is available when it needs. In contrast non-functional features are more such as serviceability ensuring a nearly zero-administration database, offering a complete set of tools to interact with the Database, and manage backups and service the database with minimal disruption to workflow. Another versatile non-functional feature is Cloud DevOps Assist which lets developers to optimize their databases in such a fashion that not only is the workload smoothly, but also the query cache for reuse and options to view data statistics providing customers aware of where, when and how concerning their data and how it is being

handled by third parties. Moreover, it aligns set of practices to reach high degree of quality and automated delivery [7].

Below table demonstrate percentage of features for each NewSQL database so that if anyone is interested to figure out which NewSQL database cares about certain features, he could recognize that by these functional features or non-functional features percentage. NewSQL Databases Functional Percentage Non-Functional Percentage

Table 3: depicts percentage of features for each NewSQL database.

NewSQL Databases	Functional %	Non-Functional %
Altibase	42.11	57.89
C-TreeACE	42.86	57.14
ClustrixDB	50	50
CockroachDB	46.67	53.33
MemSQL	52.63	47.37
NuoDB	50	50
Percona TokuDB	62.5	37.5
Gem re XD	85.71	14.29
ActiveSpaces	50	50
VoltDB	60	40
ActorDB	55.56	44.44
Google Spanner	64.29	35.71
Trafodion	50	50

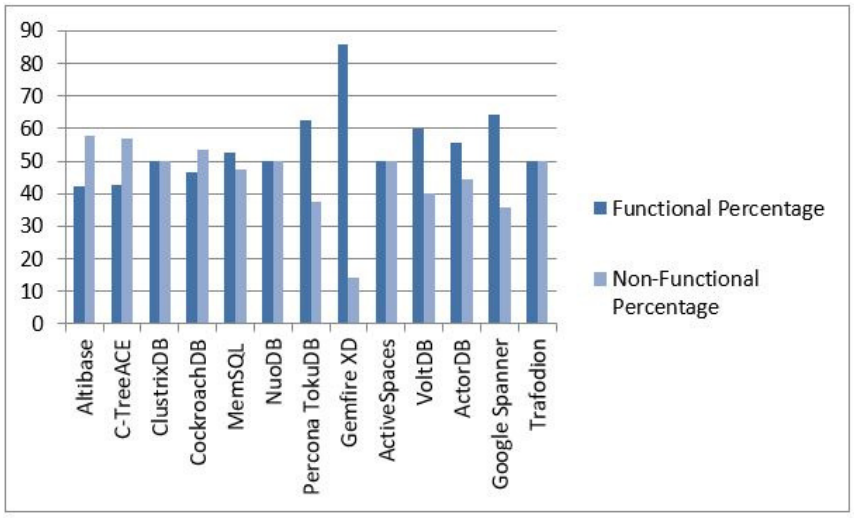


Figure 4: discerns percentage of features for each NewSQL database.

Above graph shows that how each NewSQL database is significant in certain features and minor in other features. Percentage of these functional and non-functional features will assist arbitrary usage to gain enough knowledge of which feature is standing out in each single NewSQL database. GemFire has the highest functional features, so it might lead us to see it as stringent enough to manipulate data whereas the lowest non-functional features exist in the same GemFire. Surprisingly, Frafodion has functional features as same as non-functional features.

2.2. Supervised, Semi Supervised and Unsupervised Features [12]

We group the target NewSQL databases into another classification. Therefore, we focus on how data processes are being addressed; we came up with three main categories. First, supervised feature whose transaction process result requires no a certain requirement to be performed considering its classification. Second, semi-supervised feature whose transaction process result might or not require a certain requirement to be performed with respect of its classification or cluster on its sparse data. Third, unsupervised feature whose transaction process result demands certain requirements to be performed considering its cluster responsibility.

Supervised features ensure high performance guarantee, accuracy, and advocating replication. In addition of that, supervised features verify good data from bad data. Unsupervised features found their endeavor is to be compatible with supervised features to validate and achieve data integrity. Semi-supervised features balance between the former features and the later as well.

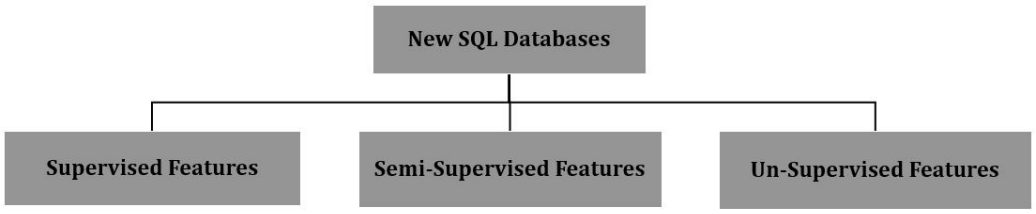


Figure 5: Taxonomy 2 displays features categories.

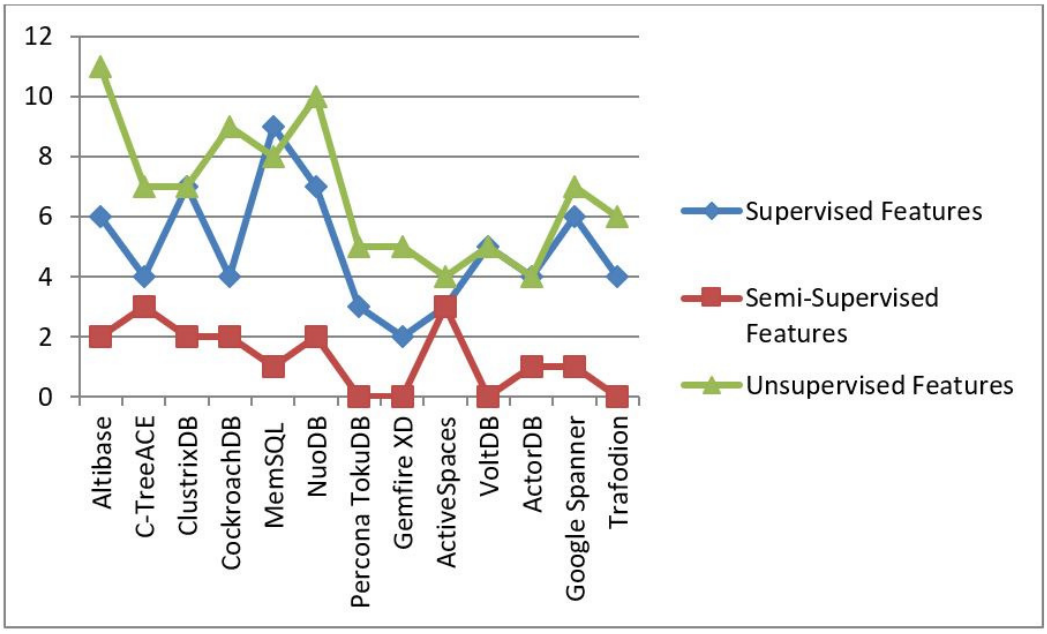


Table 4: shows features type's distribution

The Result:

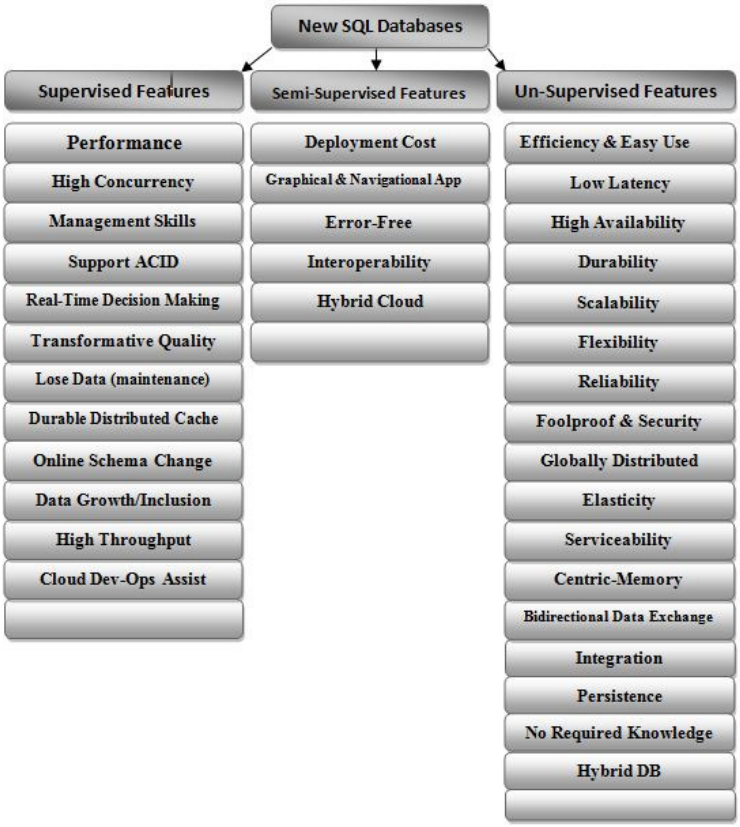


Figure 7: Result 2 discern features distributions based on supervised semi and unsupervised features.

At result 2 convey features are classified into three categories which are supervised, semi supervised and unsupervised feature. High concurrency, management, ACID, and loss data are standing out of supervised features. Interoperability, error-free are fell within semi-supervised. Scalability, elasticity, centric-memory, persistence are fell down under unsupervised.

Table 5: display percentage of supervised-semi-unsupervised features.

NewSQL Databases	Supervised %	Semi-Supervised %	Unsupervised %
Altibase	31.58	10.53	57.89
C-TreeACE	28.57	21.43	50
ClustrixDB	43.75	12.5	43.75
CockroachDB	26.67	13.33	60
MemSQL	50	5.56	44.44
NuoDB	36.84	10.53	52.63
Percona TokuDB	37.5	0	62.5
GemFire XD	28.57	0	71.43
ActiveSpaces	30	30	40
VoltDB	50	0	50
ActorDB	44.44	11.11	44.44
Google Spanner	42.83	7.14	50
Trafodion	40	0	60

This below table will display percentage of supervised, semi-supervised and unsupervised features so that if arbitrary usage by anyone would mitigate the effort to find which NewSQL database involves and offers certain serviceability.

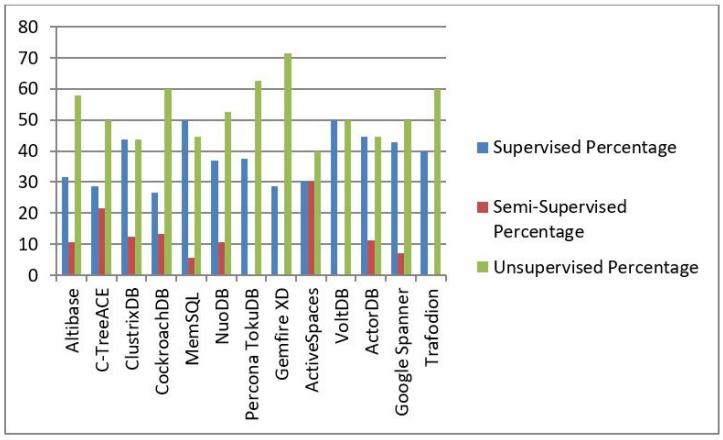


Figure 8: Graph 4 displays percentage of supervised-semi-unsupervised features

Above graph will demonstrate each NewSQL database how much percentage to provide supervised, semi-supervised, and unsupervised features. Interestingly, GemFire seems support the highest non-supervised features to sustain data integrity whereas the lowest is Active Spaces. Another versatility of supervised features is advocated by VoltDB as well as MemSQL which they are stringent enough to resist high performance. In terms of semi-supervised features, again Active Space stands out to defeat failure error strongly.

4.3. NewSQL databases process several types of big data

The versatility of NewSQL database is to ingest diverse types of databases at the same application. So most database types have been found are categorized into historical, real-time, streaming and timestamp databases.

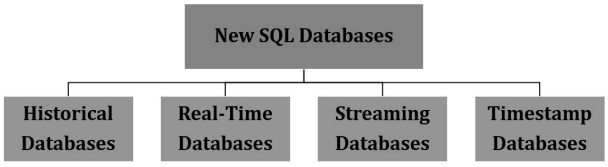


Figure 9: Taxonomy 3: shows diverse databases categories advocated in NewSQL.
Defining Bitmap:

Table 6: depicts NewSQL databases manipulating diverse data.

NewSQL Databases	Historical Database	Real-Time Database	Streaming Database	Timestamp Database
Altibase	1	1	0	0
C-TreeACE	1	0	0	0
ClustrixDB	0	1	1	0
CockroachDB	1	0	0	0
MemSQL	1	1	0	0
NuoDB	0	0	1	0
Percona TokuDB	0	0	0	0
GemFire XD	0	1	0	0
ActiveSpaces	1	0	0	0
VoltDB	0	1	0	0
ActorDB	0	0	0	0
Google Spanner	1	0	0	1
Trafodion	0	1	0	0

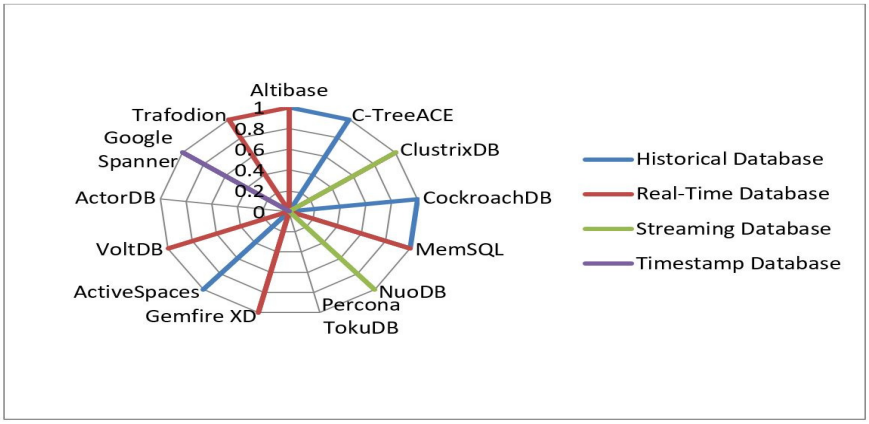


Figure 10: depicts NewSQL databases manipulating diverse data.

4.3.1. Historical Database

Historical data is stored in warehouse system in huge volume of data comparing to OLTP, Online Transaction Processing [8]. Historical data can be useful in helping to predict the future when conducting predictive analyses [43]. In our finding for instance Altibase has found a reliable support for varied workloads to permit real time access to time critical hot data, access to historical data, and complex transaction through integrated data, with easy bidirectional data movement between disk table and memory as well as joint between disk and memory tables [35].

In addition, C-TreeACE provides replication and hot backup as part of the whole package [33]. Another NewSQL database advocating historical data CockroachDB offers bounding data to center to provide transplant and ongoing serviceability [29]. When CockroachDB databases are placed in different geographical regions, they are able enough to support high elastic configuration as well as data access from the lowest latency sources. Therefore, consistency and speed are not required anymore to do so.

Not only does that, but also it could provision history data as well as records for reusability in future. The plus feature is the self-cleaning process for storage appliances and disks. Moreover, MemSQL is able enough to manipulate and access both real-time as well as historical data [27].

Not does MemSql render decision about picking out the latest data, but it eliminates the unnecessary queries. As well as ActiveSpaces EE reveals data to be conveyed as desired history data so preferable data will be available at real-time viewing of data and to keep users to view demanded data at later or at necessary time [38]. Therefore, the system here replaces the less used data with frequent used data to ensure all fall under the efficient data management process.

4.3.2. Real-Time Database

Real-time database involves processing data to address system workload and heavy traffic where changes are constantly updated [41]. Real time processing is to handle workloads whose state is constantly changing. This differs from traditional databases containing persistent data, mostly unaffected by time. . Banking and online shipping tracking systems are suitable for instance for real-time database. To illustrate for this database type, an aircraft should not land before certain real-time data have been retrieved such as fuel, altitude, and speed [40]. So delaying for these data information may lead and guide to devastating. Possibly, real-time transactions are involved individually or multiply into these types of systems, or even are mingled and associated with non-real-time transactions [41]. Real-time databases concern about time constrains which are characterized by deadline and reliability requirements, so critical transactions are classified into computation time and completion deadline [41].

First NewSQL database supporting real-time database is ClusterixDB handles a massive transaction volume with its in-build management system; and real time analytics on customer's live operational data [31]. MemSQL is also stringent enough to process as well as access real time data to build decision for how to choose updated data and how to ignore replications or unimportant transactions [27]. Pivotal GemFire XD advocates on demand data for real-time data in order to keep high performance and management of good data grid [21]. The feature of real-time event notifies all subscribed real-time events so that all changes have been made will be applied into all event members. Trafodion performs data in run time for optimization and in high performance, so OLTP workload will be compiled efficiently [23].

4.3.3. Streaming Database

Streaming Data is data established constantly through numerous data sources considering its synchronization while sending data records in slight sizes. Streaming data involves a wide variety of data like log files a wide variety of data [5]. NuoDB has wide geography mapping to deliver mainstream data with no hindering as well as with a quality transmission [24]. Not how efficient NuoDB manage and store the data, but how effective it keeps them up to date to render them available for management and queries.

ClusterixDB's Cloud DevOps feature assists permitting to optimize the database in such way to handle streamed workload smoothly without any replacement of hardware or code changes [31]. Another NewSQL database supporting stream data is MemSQL which addresses stream data to control significant chunks of data which provides propriety to investigate the power of changing the datasets [27]. MemSQL permits data growth completely through its provisioning In-memory row & disk-based column store features versatilities in a unique database with significant accomplishment of low latency. [10] VoltDB has streamline architecture to deploy consistency aligned with reliability, so this feature will ensure all efficiency for professional usage.

4.3.4. Timestamp Database

Applications in Spanner reads, reveals and writes based on timestamp, application can read data at old timestamps. Spanner offers replication configuration, so application could manage how far its data from user to control read latency and how far replicas are from each others to satisfy write latency. By this manner which accessing globally database in timestamp, it make implementing

its database sound of difficult. However, these features let Spanner support backup, consistent Map Reduce execution, and atomic schema updates. Spanner assigns meaningful timestamp into distributed transactions to represent serialization order thereby serialization order ensure consistency (or linearizability) [10].

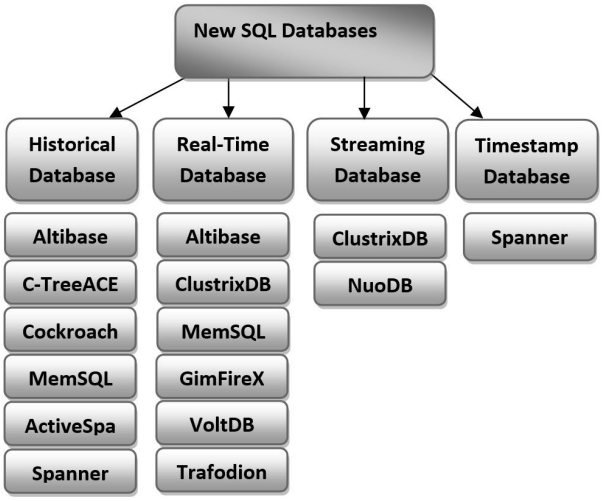


Figure 11: Result 3 convey versatility of NewSQL databases to ingest di erent other databases.

Our result 3 discerns which NewSQL databases ingest different databases associated concurrently with respect of low latency. Most NewSQL databases are conveyed as powerful ingestion of real-time, streaming, and historical databases whereas timestamp database is found rarely utilized within Spanner operated by Google. To illustrate, ClustrixDB offers high availability with self-management to involve database servers so that it could boost in growing capacity, throughput, and elimination of hardware failures [31]. ClustrixDB has self-management system manipulating all complicated operations as well as eliminating all chances of time delay due to failures in hardware.

Furthermore, CockroachDB provision scalability and consistency SQL database as well as an automated recovery once disaster has been taken place to keep data at safe mode [29]. Another database is Pivotal GemFire XD; its Cloud-Ready feature lets on-demand access ensure high performance, high availability, and event-driven data on leading cloud platform [21]. Not only does this feature work independently, but it ensures no hindering on ongoing data action, changes made to other data with self-maintained manipulation.

4.4. NewSQL database Generation

4.4.1. Horizontal and Vertical Scalability of Distributed System Generation

Data is managed through several ways to ensure data integrity and its consistency considering its fault tolerance, so several NewSQL databases provide horizontal and vertical scalability to ensure previous features and others. Horizontal scaling concerns of increasing the commodity nodes (hardware) whereas vertical scaling considers of the CPU and RAM power enhancement into current nodes/commodities [8].

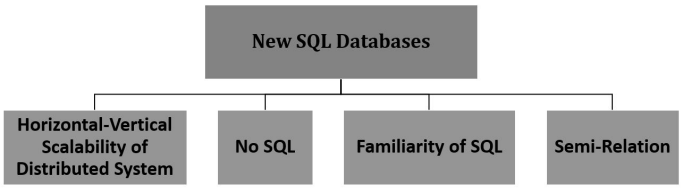


Figure 12: Taxonomy 4 reveals mechanism and technologies categories sup-ported in NewSQL. We convey several of NewSQL databases which support it. Defining Bitmap in order to reveal which NewSQL databases support certain technologies.

Table 7: depicts NewSQL databases supporting other data technologies.

NewSQL Databases	Horizontal-Vertical Scalability of Distributed System	Familiarity of SQL	NoSQL	Semi-Relational
Altibase	0	1	0	0
C-TreeACE	0	1	1	0
ClustrixDB	0	1	0	0
CockroachDB	0	1	0	0
MemSQL	1	1	0	0
NuoDB	0	1	0	0
Percona TokuDB	0	0	0	0
Gem re XD	1	0	0	0
ActiveSpaces	0	0	1	0
VoltDB	0	1	0	0
ActorDB	1	1	0	0
Google Spanner	0	1	0	1
Trafodion	0	1	0	0

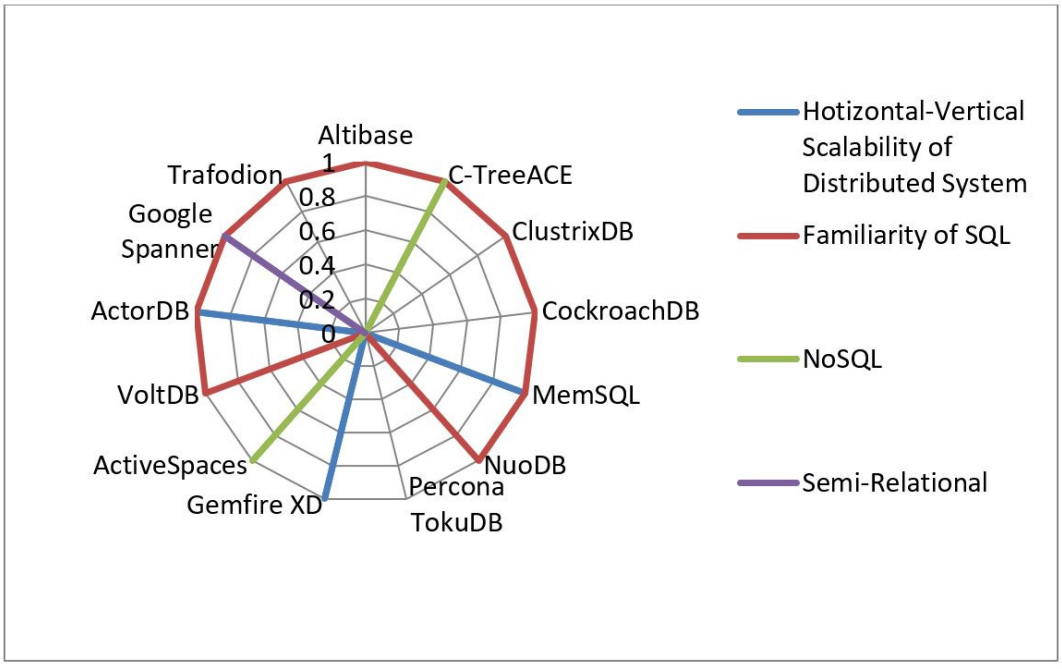


Figure 13: depicts NewSQL databases supporting other data technologies.

Not all New SQL databases advocate concrete technologies for instances C-TreeACE provides full SQL engine supporting with relational and non-relational APIs [33]. With its unique No SQL facilitates high performance No SQL and Standard SQL access within the same application over the same data. Moreover, MemSQL joins horizontal scalability of distributed system with the familiarity of SQL to let businesses process changing datasets so that it could ingest streams smoothly as well as manipulating transactions simultaneously in a unique database [27]. Another observation is ActorDB has distributed and horizontal scalability ensuring consistency and reliability on its database engine [22].

4.4.2. Familiarity of SQL Generation

Interestingly, it has been found the majority of NewSQL databases does advocate familiarity of SQL, for instance ClusterixDB takes pain of scaling by automating all of the complicated database operations traditionally performed to enhance throughput, capacity, and high availability [31]. Another example is Altibase which advocates basically traditional relation database, so no need for shortcuts for speed, and no learning curve for DBA [35]. \par Moreover, CockroachDB supports distributed SQL and transactions with extreme consistent to let developer considerate on what matters so that he could build robust application [29]. Letting a developer considerate on what matters is an interesting characteristic to be involved, so CockroachDB will provide SQL RDMBS with ACID transactions. Furthermore, MemSQL works with familiarity of SQL considering its horizontal scalability distributed system. Therefore, MemSQL reveals as easier ever in how to manage distributed system structure database [27].

NuoDB deals and acts to address considerable data in respect of the familiarity of SQL to keep consistency state for data [21]. Durable Distributed Cache, DDC, involve the power of RDBMs, ANSI SQL support, ACID transaction, backup, security and administration. At significant peak time of synchronized access, GemFire's horizontal architecture assists users to get rid of low latency due to in-memory data grid built in GemFire XD so that it could meet the highest degree of demanding peak of data utilities [21]. Not only does that, but also it sustains runtime cost down when users commence scaling back at peak usage. VoltDB governs distributed data with speed of scalability through its versatile architecture of transactional ACID with its traditional RDBMs [18].

Furthermore, it supports in wide range of SQL familiarity to enhance compatibility and decision making. ActorDB advocate completely relational database within queries and transactions across multiple of actors as well as ACID guarantee [22]. Trafodion involves a web scale SQL and it reuses SQL skills to enhance productivity [23].

4.4.3. No-SQL Generation

Several NewSQL databases involve No SQL technology to be associated and support diverse applications. Yet surprisingly, in our survey study, we found a few of NewSQL databases supporting No SQL such as C-Tree ACE, It's appropriate for a broad range of transactional applications with ACID advanced key-value supporting relational and non-relational APIs [33]. Not only has that, but also it reveals its versatility No SQL technology to achieve and accomplish high performance [33]. No SQL to access within the same applications over the same data. Zero Administration is supported in C-TreeACE. Replication, backup, and auto-recovery are involved features. TIBCO ActiveSpaces is capable to ensure full ACID-compliant NoSQL data grid so that consistency as well as concurrency control access are advocated across plenty and numerous of databases [38]. Across several nodes, fault tolerance and distributed persistence which are offered will ensure synchronous replication and durability as well [38].

4.4.4. Semi-Relationship Generation

This semi-relational has been observed in Spanner and how it works. In Google Spanner, Megastore is used by several Google applications due to their semi-relational data model and to advocate simultaneous replications regardless of their poor throughput [10]. Data schema is stored in semi-relational tables, so each version has certain time with its commit time. Spanner supports SQL-based queries [10].

The result:

In result 4 demonstrates our target of technologies or other mechanisms where NewSQL databases support and associate. Most significant mechanism or technology found is horizontal and vertical scalability of distributed system where MemSQL, GemFire XD, and Actor DB are advocated in such this mechanism. No SQL and Semi-Relation are found slightly associated in our databases target whereas the familiarity of SQL is provided by the majority of NewSQL databases.

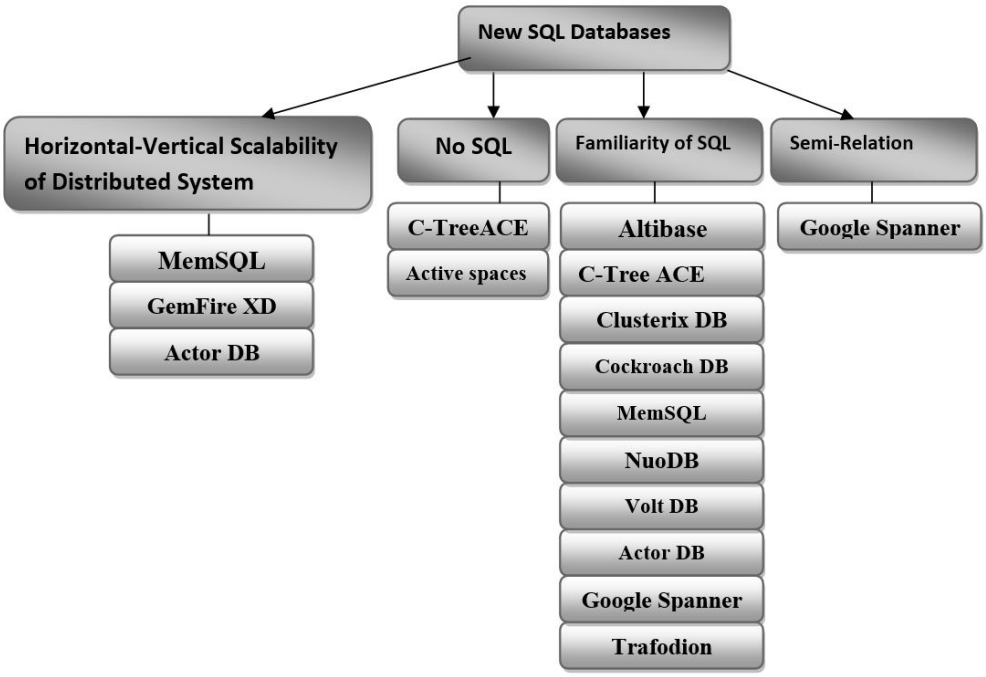


Figure 14: Result 4 discern how NewSQL databases advocate other technologies.

Result 4 depicts which NewSQL databases associated and providing other technologies; for instance In C-TreeACE, It compounds a unique NoSQL technology to boost a high performance. Configuration and its efficient are controlled by C-TreeACE to develop data application [33]. Security is applied to all diverse type of relation data transactions. Moreover, Percona TokuDB offers slightly deployment cost for big data application by restricting such specifications which are related to scaling effort and optimization [20].

Besides of that, NuoDB has a durable distributed cash feature, DDC, combines the power of RDBMs, ANSI SQL with ACID support, and administration to support security and scalability to provide constant availability across multiple data centers [24]. Centric Memory provides functional and available schema with to boost geographical distribution.

5. DETAILED TAXONOMY

Data volume growth appears in every single minute forming big data which convey that it comes from uncontrolled environment. Therefore, manipulation or processing and storing of big data in native form is one innovation way of NewSQL databases innovations. A single node used to process and analyze data in legacy system whereas multi-nodes execute its data separately based on relative task and the eventual output is aggregated [8].

Big data main characteristics are 3 up to 7 Vs which our concern is up to five: Volume, Variety, Velocity, Value, and Veracity [8]. Volume discerns significant volume of data; bid data velocity represents short time in data manipulation; whereas variety convey divers data type such as numeric, textual, videos, pictures. Usually data are batched once in big data unlike in relational database where data is restricted to follow a certain table structure. Big data will induce organizations to redesign their accountabilities and relationships between business and information technologies [14]. The traditional behavior of information technology forces concrete requirements and process into certain standard to control changes [14]. Another disruption method of big data against traditional role of business and IT is to reveal business data as though prior analysis of business [14]. Processing, storing and controlling big data is versatile mechanism of NewSQL databases.

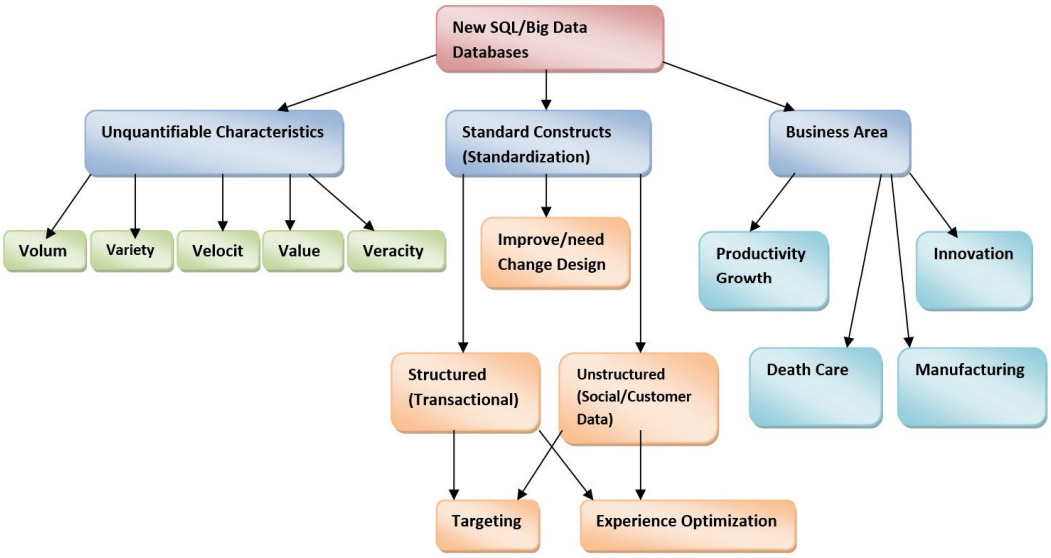


Figure 15: Taxonomy 5: reveals detailed NewSQL databases classification.

6. CONTRIBUTION

In this paper, we focus on NewSQL databases classification in certain methods which have not done formerly. Investigation of NewSQL databases features was our objective in this paper to recognize how versatile NewSQL databases are to provide divers databases associated with each other considering their significant engine to resist joining nodes and leaving others simultaneously with low cost of commodity hardware. Then we discussed four taxonomies of NewSQL databases with their results of our NewSQL databases target. First, we were concern about how these features were associated and compatible with NewSQL databases, so we found they might be classified into functional and non-functional features.

Second, versatility of NewSQL databases is proved by its offering numerous and different databases concurrently achieving low latency, so we released four different databases are ingested powerfully by NewSQL databases which ingest historical, real-time, streaming, and timestamp databases. We moved then into another category recognize what NewSQL databases can offer certain technologies and mechanism besides of its stableness. Based on supporting technologies, we discover the majority of NewSQL databases support horizontal and vertical of distributed system, familiarity of SQL whereas the minority of NewSQL databases are supporting NoSQL, Semi-Relation databases. At the end of this paper, we propose a detailed taxonomy of NewSQL databases.

7. WHAT IS NEXT FOR BIG DATA?

Smart data:

"Smart data is the way in which different data sources including big data are brought together, correlated, analyzed, etc, to be able to feed decision-making and action processes" [15]. Smart data considers as a significant candidate of what is next for big data. Moreover, smart data is arise from BI, Business Intelligence, to provide all values of technologies, processes, and structured data to been demonstrated as though an entire set [15].

Big data and data management are what big data management is about as well as how these two main factors associated to accomplish business and IT goals. In layered architecture of visualization and data management, data management is in charge of data flow to control storing or retrieval into or out of either main memory, or disk locally or remotely [17]. Therefore, managing big data demands such slight mechanism considering low latency and low commodity nodes [16]. We attempt to find an opportunity to architect new features into NewSQL databases so that they might assist organizations to be proactive, and making best future intelligent decision.

Depending on predictive analytics, we believe utilizing certain mechanisms data mining, modeling, machine learning, AI, and statistics to build future intelligent data prediction from diverse patterns and relationships considering both structured and unstructured data would assist to manage and maintain big data perfectly [39]. Moreover, raw data inform what the story of environment and customer operations is. So finding a way to sustain these data in safe mode is mandatory.

8. FUTURE WORK

Big data should enter a certain mechanism to ensure its durability so that it verifies data persistence as well as data resilience. Building data network to maintain data security during peak time or disaster or catastrophe failure. After data has been manipulated it could be into two classifications as good data which assist us to make decision and bad data which hinders to read, reveal, or retrieve data smoothly. Interestingly, New SQL databases process both new (upcoming) data and old data.

Another suggestion here is we believe that big data should deal and behave based on stringent and secure processing and store, so white data indicate to data has consistency of processing and storing big data whereas black data states these data have lacking of secure processing and insecure store. Another data type we suggest green data which refers to robust and versatile processing and storing big data.

CONCLUSION

To sum up, big data considers significant hurdle to process at concrete time with fault tolerance. Therefore, finding particular mechanism involving very intelligent features from what have been covered at this paper might accelerate not only how to process such big data, but also how to store them in a practical manner. Joining certain part of data on diverse nodes placed considers mandatory, yet that how big data must be treated, so by this method the response time will be boosted [8]. Hence, efficient architecture should have powerful features to defeat delaying in response time, joining or leaving nodes or processing time.

Almost all NewSQL databases are stringent and robust enough to ingest big data in a smart technique within certain time and with commodity nodes. This is the versatility of NewSQL databases at achieving low latency as well as failure recovery automatically. Most our work focus on NewSQL databases and other technologies offered and advocated. Another thing we discussed in this paper is how NewSQL databases associate with other databases. Moreover, we propose our own taxonomy in terms of functional and non-functional features. Then we observe these features could be classified based on supervised, semi-supervised, and unsupervised features. Interestingly, Google Spanner came up with concrete way in terms of how to manipulate (process) big data. Perhaps it has concrete applications it uses. MemSQL, GemFire XD and ActiveSpace were powerful in most our taxonomy achieving best result regarding other databases association, and other technologies supported.

REFERENCES

[1] Ismail, M., Gebremeskel, E., Kakantousis, T., Berthou, G., Dowling, J. (2017, June). Hopsworks: Improving User Experience and Development on Hadoop with Scalable, Strongly Consistent Metadata. In Distributed Comput-ing Systems (ICDCS), 2017 IEEE 37th International Conference on(pp. 2525-2528). IEEE.

[2] Sangtani, M., D'souza, S. M., Harsh, M., Chander, S., Vijaya, P. IN-TERNATIONAL JOURNAL OF ENGINEERING SCIENCES RESEARCH TECHNOLOGY IMPLEMENTATION CHALLENGES INVOLVED IN BIG DATA ANALYTICS.

[3] Kobielus, J. (2012). Hadoop: Nucleus of the next-generation big data ware-house. IBM Data Management Magazine.

[4] Lightstone, S., Ohanian, R., Haide, M., Cho, J., Springgay, M., Steinbach, T. (2017, April). Making Big Data Simple with dashDB Local. In Data Engi-neering (ICDE), 2017 IEEE 33rd International Conference on (pp. 1195-1205). IEEE.

[5] Santos, M. Y., Costa, C., Galv~ao, J., Andrade, C., Martinho, B. A., Lima, F. V., Costa, E. (2017, July). Evaluating SQL-on-hadoop for big data warehous-ing on not-so-good hardware. In Proceedings of the 21st International Database Engineering Applications Symposium (pp. 242-252). ACM.

[6] Ismail, M., Gebremeskel, E., Kakantousis, T., Berthou, G., Dowling, J. (2017, June). Hopsworks: Improving User Experience and Development on Hadoop with Scalable, Strongly Consistent Metadata. In Distributed Com-puting Systems (ICDCS), 2017 IEEE 37th International Conference on (pp. 2525-2528). IEEE.

[7] Thusoo, A., Shao, Z., Anthony, S., Borthakur, D., Jain, N., Sen Sarma, J., Liu, H. (2010, June). Data warehousing and analytics infrastructure at face-book. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (pp. 1013-1020). ACM.

[8] Barkhordari, M., Niamanesh, M. (2017). Atrak: a MapReduce-based data warehouse for big data. The Journal of Supercomputing, 1-15.

[9] Tankard, C. (2012). Big data security. Network security, 2012(7), 5-8.

[10] Corbett, J. C., Dean, J., Epstein, M., Fikes, A., Frost, C., Furman, J. J., Hsieh, W. (2013). Spanner: Google's globally distributed database. ACM Transactions on Computer Systems (TOCS), 31(3), 8.

[11] Song, L., Smola, A., Gretton, A., Borgwardt, K. M., Bedo, J. (2007, June). Supervised feature selection via dependence estimation. In Proceedings of the 24th international conference on Machine learning (pp. 823-830). ACM.

[12] Huang, S. H. (2015). Supervised feature selection: A tutorial. Arti cial Intelligence Research, 4(2), 22.

[13] Erturk, E., Jyoti, K. (2015). Perspectives on a Big Data Application: What Database Engineers and IT Students Need to Know. Engineering, Technology Applied Science Research, 5(5), pp-850.
[14] Davenport, T. H., Barth, P., Bean, R. (2012). How big data is di erent. MIT Sloan Management Review, 54(1), 43.
[15] [Book] Iafrate, F. (2015). From big data to smart data (Vol. 1). John Wiley Sons.
[16] Russom, P. (2013). Managing big data. TDWI Best Practices Report, TDWI Research, 1-40.
[17] Cox, M., Ellsworth, D. (1997, August). Managing big data for scienti c visualization. In ACM Siggraph (Vol. 97, pp. 21-38).
[18] https://www.predictiveanalyticstoday.com/voltdb/
[19] https://www.predictiveanalyticstoday.com/tibco-activespaces/
[20] https://www.predictiveanalyticstoday.com/percona-tokudb/
[21] https://www.predictiveanalyticstoday.com/pivotal-gem re-xd/
[22] http://www.actordb.com
[23] http://trafodion.apache.org/
[24] https://www.predictiveanalyticstoday.com/nuodb/
[25] https://www.nuodb.com/product/durable-distributed-cache
[26] https://www.whoishostingthis.com/resources/ansi-sql-standards/
[27] https://www.predictiveanalyticstoday.com/memsql/
[28] https://www.memsql.com/product/
[29] https://www.predictiveanalyticstoday.com/cockroachdb/
[30] https://www.cockroachlabs.com/
[31] https://www.predictiveanalyticstoday.com/clustrixdb/
[32] https://www.clustrix.com
[33] https://www.predictiveanalyticstoday.com/c-treeace/
[34] https://www.faircom.com/products/c-treeace
[35] https://www.predictiveanalyticstoday.com/altibase/
[36] http://altibase.com
[37] http://epubs.siam.org/doi/abs/10.1137/1.9781611972771.75
[38] https://www.tibco.com/products/tibco-activespaces
[39] https://www.predictiveanalyticstoday.com/what-is-predictive-analytics/
[40] http://databasemanagement.wikia.com/wiki/Real-timeDatabase
[41] https://en.wikipedia.org/wiki/Real-time_database
[42] https://aws.amazon.com/streaming-data/
[43] https://businessintelligence.com/dictionary/historical-data/
[44] Zahid, A., Masood, R., & Shibli, M. A. (2014, June). Security of sharded NoSQL databases: A comparative analysis. In Information Assurance and Cyber Security (CIACS), 2014 Conference on (pp. 1-8). IEEE.
[45] P. Kadebu, I. Mapanga, A security requirements perspective towards a secured nosql database environment.
[46] databases2, in: 2011 IEEE Workshop on Electronics, Computer and Applications, Changsha, 2011.
[47] Pavlo, M. Aslett, What's really new with newsql?, ACM Sigmod Record 30 45 (2) (2016) 45–55.
[48] Moniruzzaman, A. B. M. (2014). Newsql: towards next-generation scalable rdbms for online transaction processing (oltp) for big data management. arXiv preprint arXiv:1411.7343.
[49] Aslett, M. (2011). How will the database incumbents respond to NoSQL and NewSQL.San Francisco, The, 451, 1-5.
[50] Stonebraker, M. (2012). Newsql: An alternative to nosql and old sql for new oltp apps. Communications of the ACM. Retrieved, 07-06.
[51] Hoff, T. (2012). Google Spanner's Most Surprising Revelation: NoSQL is Out and NewSQL is In. Retrieved 2012-10-07.
[52] Aslett, M. (2010). What we talk about when we talk about NewSQL.
[53] Jain, H. T. S. C. V. Rise of NewSQL.
[54] https://www.predictiveanalyticstoday.com/newsql-database

11. APPENDICES

Table 8: Appendix A - Functional / Non-Functional Features

No.	Features	Functional	Non-Functional
1	High Performance/Fast Storage Engine	*	
2	Operational Efficiency & Easy Use	*	
3	Deployment Cost		*
4	Achieving Low Latency	*	
5	High Concurrency	*	
6	High Availability	*	
7	Durability	*	
8	Management Skills Tasks	*	
9	Graphical & Navigational App		*
10	Support ACID	*	
11	Scalability		*
12	Flexibility		*
13	Error-Free		*
14	Reliability		*
15	Real-Time Personalization & Decision Making	*	
16	Foolproof & Security		*
17	Globally Distributed		*
18	Elasticity		*
19	Transformative Quality		*
20	Interoperability		*
21	Serviceability		*
22	Lose Data Re-replication & Redistribution (maintenance)	*	
23	Durable Distributed Cache (DDC)	*	
24	Centric-Memory		*
25	Online Schema Change	*	
26	Configuration		*
27	Integration		*
28	Persistence		*
29	Hybrid Cloud		*
30	Bidirectional Data Exchange	*	
31	Data Growth/Inclusion		*
32	High Throughput		*
33	No Required Knowledge		*
34	Cloud Dev-Ops Assist		*
35	Hybrid DB		*

Table 9: Appendix B Supervised - Semi Supervised - Unsupervised Features

No.	Features	Supervised	Semi-Supervised	Unsupervised
1	High Performance/Fast Storage Engine	*		
2	Operational Efficiency & Easy Use			*
3	Deployment Cost		*	
4	Achieving Low Latency			*
5	High Concurrency	*		
6	High Availability			*
7	Durability			*
8	Management Skills Tasks	*		
9	Graphical & Navigational App		*	
10	Support ACID	*		
11	Scalability			*
12	Flexibility			*
13	Error-Free		*	
14	Reliability			*
15	Real-Time Personalization & Decision Making	*		
16	Foolproof & Security			*
17	Globally Distributed			*
18	Elasticity			*
19	Transformative Quality	*		
20	Interoperability		*	
21	Serviceability			*
22	Lose Data Re-replication & Redistribution (maintenance)	*		
23	Durable Distributed Cache (DDC)	*		
24	Centric-Memory			*
25	Online Schema Change	*		
26	Configuration			*
27	Integration			*
28	Persistence			*
29	Hybrid Cloud		*	
30	Bidirectional Data Exchange			*
31	Data Growth/Inclusion	*		
32	High Throughput	*		
33	No Required Knowledge			*
34	Cloud Dev-Ops Assist	*		
35	Hybrid DB			*