COMPARISON OF SQL, NOSQL AND NEW SQL DATABASES IN LIGHT OF INTERNET OF THINGS – A SURVEY

[1]HALEEMUNNISA FATIMA, [2]KUMUD WASNIK

[1]M.Tech (Student), [2]Asst. Professor
Department of Computer Science and Technology, Usha Mittal Institute of Technology, Mumbai, India
E-mail: [1]fatima035@gmail.com, [2]kumudwasnik@gmail.com

Abstract- In this era of technological advancement the recent growth in the IT enabled industry has increased the amount of "Big Data" namely unpredictable amount of data as well as with the arrival of Internet-of-Things, the IoT is expected to generate huge amount of data from varied locations that is collected very quickly. Hence, with features like scalability, consistency and performance factors are coercing the traditional database users into adopting the NoSQL database management system. This new storing database technology is asserted to perform better than the classic SQL counterpart. NoSQL DBMSs although highly scalable compromises on ACID properties for its high performance. To overcome this integral deficiency a modern alternative class of database management system has emerged called the NewSQL. NewSQL extends the scalable performance of NoSQL system at the same time retaining the SQL and ACID behavior of the traditional database system. In this paper the challenges in the said technology which both NoSQL and NewSQL have and have to overcome are discussed. Various open source SQL, NoSQL and NewSQL databases are studied such as MySOL, MongoDB, Cassandra and VoltDB which further contains data models, characteristics and comparisons.

Keywords- SQL, NoSQL, NewSQL, Document Stores, Column Family, ACID, Big Data, Internet of Things.

I. INTRODUCTION

Database Management System is an exhaustive term that refers to a collection of different tools and programs that allow the user to store modify or retrieve data based on various parameters. Traditional database systems for storage have been based on the relational model. These are widely known as SQL databases named after the language they were queried by [1]. However, with the advent of internet and a surge in the number of applications available, has led to humongous amount of data that is hard to handle. This gave rise to a new class of database systems, namely non-relational, ordinarily known as NoSQL databases. Despite the fact that these NoSQL databases process unordered data they do not fully support the online transaction processing (OLTP) the way they should. To surpass these setbacks NewSQL came into being with the resolve of filling this gap and providing required scalability.

In last few years there has been a phenomenal growth in the amount of data that is being generated which is observed by the rapid innovation in the technology. Big data is a term, used to describe this enormous quantity of data, which is structured, semi-structured and unstructured. According to the Gartner group, Big Data can be defined by 3Vs: volume, velocity and variety [2]. Processing such vast amounts of data requires speed, flexible schemas and distributed databases [1].
There has been a dramatic increase in the usage of sensors enabled applications particularly automation of homes, traffic control, smart street lights et al. in consumer as well as industrial domain. Internet-of-Things, IoT, is an application domain that integrates different technological and social fields [3]. The term Internet of Things (IOT), also known as Internet of Objects refers to the networked interconnection of everyday objects, which is generally viewed as a self-configuring wireless network of sensors whose purpose would be to interconnect all things [4]. In the past, the IoT was primarily referred to RFID (Radio-Frequency Identification) objects elevating to WSN (Wireless Sensor Networks) and now including just about everything. Thus, IoT foresees an age where billions of sensors will be connected to the Internet, thereby expanding the growth in the amount of data that will be generated in due course.

Owing to these characteristics of Big Data and IoT data, the applications that are created need to be carefully designed and implemented on many storage layers more like a hybrid platform. That is to say the classic RDBMS systems cannot fully overcome the challenges in handling such large amounts of data. Nevertheless to achieve; scalability, high performance, security, availability, et al. the NoSQL and NewSQL data stores offer themselves as better data processing alternatives.

There are several database contributions that offer viable solutions and adaptable data models for both existing and future applications depending on what results are to be yielded based on the DBMS system used. Numerous survey papers have thus been published (Hecht, R., & Jablonski, S. [5] or Tudorica, B. G., & Bucur, C. [6] or Moniruzzaman, A. B. M. [7]) to remark and give feedback in response to the queries that arise.
In this paper, we have discussed the basics of traditional DBMS systems namely MySQL, its features and constraints considering it is open source and also widely used. Next we discuss and review the

prominent open source NoSQL and NewSQL data models and stores, specifically MongoDB, Cassandra and VoltDB respectively discussing the key features and the recent advancements and challenges in data management. As enormous amount of data is a vital feature of Internet-of-Things.

II. LITERATURE REVIEW

This section reviews the existing data models and related surveys done in the domain of database management systems. Every database that exists today is based on a particular model. These models then logically categorize the database based on different approaches of how the data is managed. Broadly there are three main data models viz. the Traditional or Relational model, The Non-relational model and Modern Relational model. In spite of the fact that relational databases are efficient, easy to use, ACID oriented and pliant there have been many instances where they fail to yield optimal solutions. This led to users migrating to other newer platforms that were more compliant when it came to solving the aforementioned issues.

A. The Relational (Traditional) Model

The relational model (RM) for database management is an approach to managing data using a structure and language consistent with first-order predicate logic, first described in 1969 by Edgar F. Codd [8]. It is an elementary data model that is used for storing and processing the data effectively. The primary practice of storing data in this model is in the form of tables that comprises of rows and columns.

The databases belonging to the traditional class handle the concurrency in transactions by the way of conservation of ACID properties. ACID properties are basically atomicity, consistency, isolation and durability hence, a transaction must either be complete wholly or not at all. Despite the fact that this model is rigid in the way of handling data and is schema oriented, it is by far an efficient and reliable standard model.

B. The Non-Relational Model

The Non-Relational model otherwise known as the NoSQL approach is a way of managing data in other forms in ways other than the tabular way prescribed by the relational model for storing and extracting data. The term NoSQL was first used in 1998 by Carlos Strozzi [9] as a name of a database he was developing, although it was simply a RDBMS adhering database just without a SQL interface. In 2009, Johan Oskarsson, in a conference used it to describe an “open-source, distributed, non-relational databases” [10]. NoSQL databases are occasionally referred to as Not-only-SQL databases thereby not conforming to the popular practice of SQL-style querying. They are schema free or have a formable schema design that can handle a variety of data. Most of the non-relational databases available today try to fit two of the three requirements of the CAP theorem described by Brewer [11]. The ongoing existent data models can be branched into four sub-categories: namely key-value stores, document stores, column-family stores, and graph databases.

1) Key-Value Stores

Key-Value stores are the simplest kind of NoSQL databases. As the name conveys, they are similar to maps or dictionaries where data is addressed by a unique key [3]. The key uniquely identifies the value and extracts the data stored, where value is nothing but a block of data which has no fixed structure or type. Another leading feature is that they are exceptionally fast when it comes to reading and writing data, if the key is accessible. Contrary to this, if there are multiple updates they are invariably slow.

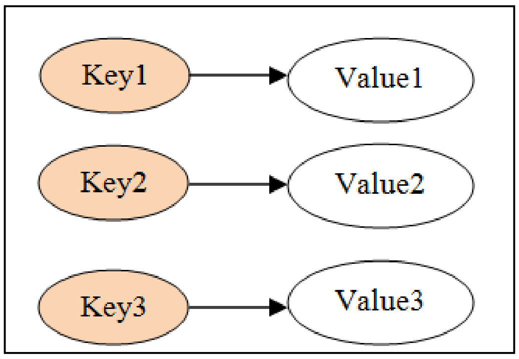

Fig. 1: Key-Value Store

Although key-value stores are simple they are not competent for complex applications. Amazon’s DynamoDB [12], Redis [13] and Memcached [14] are some popular available key-stores.

2) Column Family Stores

It is extensively considered that Google’s Bigtable [15] was the first of NoSQL databases and most of the column-family stores are derived from it. It is based on three keys: first one is called row key, second is called the column key, while the third is the timestamp [1]. Also, the dataset consists of several rows, each of which is addressed by a unique row key, known as the primary key.

The column-family stores a particular column family in different rows can contain different column. Column-family stores also share the feature of being schema-free, faster in processing and querying data. Some of the widely known column-family stores are Cassandra [16] and Hadoop HBase [17].

3) Document Stores

A document database is a schema-free derivative of the key-value store model where the database is defined as a collection of *documents*. Within these documents, keys have to be unique [3]. Thus, they are also known as document-oriented database. These documents are primarily JSON or XML format (BSON for MongoDB). Document stores allow the user to manipulate the data and store it as needed. As the database is schema-free the documents can contain any type of data with no restraints to structure et al. Indexing documents as well as the contents within these documents is implemented by the document stores based on the primary key.

The document stores often have fast writing and good querying based on the indexing along with flexibility and nested structure of records. Popular examples of document stores databases are MongoDB [18] and CouchDB [19].

4) *Graph Databases*

The main focus of the graph oriented databases is relationships with heavy interconnected data. Graph databases store data in the form of Nodes and Relationships as traversal becomes easier. These databases are good at finding pattern and determining the results. Use cases for graph databases are location based services, knowledge representation and path finding problems raised in navigation systems, which involve complex relationships [3]. Neo4j [20] is an example of graph databases.

C. The Modern Relational Data Model

The NewSQL approach is used to analyze a set of solutions that aim to support the relational model at the same time providing them with the benefit of scalability supported by the NoSQL models. The first use of the term can be traced back to 2011by 451 Group analyst Mathew Aslett [21]. Michael Stonebraker, the founder of many databases defines NewSQL as, System that preserves SQL and offers high performance and scalability, while preserving the traditional ACID concept for transactions [22].

Generally these modern approach data stores support the relational data model with SQL as the primary interface. Google Spanner [23] and VoltDB [24] are some of the most sought after representatives of this league. They are particularly designed to achieve high transaction rates combined with higher performance rates for OLTP oriented queries.

II. COMPARATIVE STUDY

Recently there has been a surge in the technological advancements with a number of databases. Many surveys have been done in comparing SQL databases with NoSQL ones or amongst NoSQL themselves [1][3][4]. In this section, we compare the popular open source SQL and NoSQL databases with their latest counterpart NewSQL.

Table 1: Comparison of different Databases

Database	**MySQL**	**MongoDB**	**Cassandra**	**VoltDB**
Category	SQL-Relational	NOSQL-Document Stores	NOSQL-Column Family	NEWSQL-Relational
Implementation Language	C and C++	C++	Java	Java
Schema	Schema oriented-Tables	Schema-less – Collections	Schema-less	Schema oriented
Map Reduce	Yes	Yes	No	No
Partitioning	Horizontal partitioning and sharding using MySOL Cluster.	Auto Sharding. Range partitioning based on shard key.	Order preserving partitioning and Consistent hashing.	Consistent hashing with user defines where the stored procedure runs.
Database	**MySQL**	**MongoDB**	**Cassandra**	**VoltDB**
OLTP	Yes	No	No	Yes
Transaction	ACID complaint focus on integrity.	Non-ACID complaint.	Non-ACID complaint.	ACID complaint focus on integrity.
Replication	Master-Slave and Master-Master approach.	Master-Slave, asynchronous	Master-less, asynchronous	Update executes on all servers concurrently.
Querying	SQL	Internal API, MapReduce and compiler Query support.	Internal API, CQL (Cassandra Query Language)	SQL, CLI and API. JDBC support.

III. CONCLUSION

Non-Relational and Modern-Relational data models coupled with IoT data is the most upcoming technology today. Considering this problem, we analyzed the existing data stores and learned that the C in CAP Theorem followed by most NoSQL is not equivalent to the C in ACID. Also since IoT data is huge there will be the case of many transactions, therefore transaction processing is an issue when dealing with classic SQL data management systems. Every system has its own pros and cons with performance being of supreme priority. VoltDB though not purely OLAP oriented has consistent performance combined with low-latency.

REFERENCES

[1] Li, Yishan, and Sathiamoorthy Manoharan. "A performance comparison of sql and nosql databases." Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on. IEEE, 2013.

[2] Beyer, Mark A., and Douglas Laney. "The importance of 'big data': a definition." Stamford, CT: Gartner (2012).

[3] Minerva, Roberto, and Abiy Biru. "Towards a Definition of the Internet of Things." IEEE IoT Initiative white paper.

[4] Conner, Margery (May 27 2010). Sensors empower the "Internet of Things" pp. 32–38. ISSN 0012-7515

[5] Hecht, Robin, and Stefan Jablonski. "NoSQL evaluation: A use case oriented survey." (2011): 336-341.

[6] Tudorica, Bogdan George, and Cristian Bucur. "A comparison between several NoSQL databases with comments and notes." Roedunet International Conference (RoEduNet), 2011 10th. IEEE, 2011.

[7] Moniruzzaman, A. B. M. "NewSQL: Towards Next-Generation Scalable RDBMS for Online Transaction Processing (OLTP) for Big Data Management." arXiv preprint arXiv:1411.7343 (2014).

[8] Codd, Edgar F. "A relational model of data for large shared data banks." Communications of the ACM 13.6 (1970): 377-387.

[9] Strozzi, Carlo. "NoSQL–A relational database management system. 2007–2010." WWW page of the article: http://www. strozzi. it/cgi-bin/CSA/tw7/I/en_US/nosql/Home% 20Page.

[10] NOSQL meetup. Eventbrite, San Francisco. http://nosql.eventbrite.com/. Accessed 29 Sep 2013

[11] https://en.wikipedia.org/wiki/CAP_theorem

[12] DeCandia, Giuseppe, et al. "Dynamo: amazon's highly available key-value store." ACM SIGOPS Operating Systems Review. Vol. 41. No. 6. ACM, 2007.

[13] Redis. http://redis.io/.

[14] Memcached. http://memcached.org/

[15] F. Chang, *et al.*, "Bigtable: A Distributed Storage System for Structured Data," *ACM Transactions on Computer Systems*, vol. 26, pp. 1-26, 2008

[16] Lakshman, Avinash, and Prashant Malik. "Cassandra: a decentralized structured storage system." ACM SIGOPS Operating Systems Review 44.2 (2010): 35-40

[17] Apache HBase. http://hbase.apache.org/

[18] The mongodb manual. http://docs.mongodb.org/manual

[19] Couchdb, a database for the web. http://couchdb.apache.org/

[20] "Neo4j." Internet: http://neo4j.org

[21] Aslett M (2011) How will the database incumbents respond to NoSQL and NewSQL? https://451research.com/report-short?entityId=66963

[22] Stonebraker, Michael. "NewSQL: An Alternative to NoSQL and Old SQL for New OLTP Apps." Communications of the ACM. Retrieved (2012): 07-06.

[23] Corbett JC, Dean J, Epstein M, Fikes A, Frost C, Furman JJ, Ghemawat S, Gubarev A, Heiser C, Hochschild P, Hsieh W, Kanthak S, Kogan E, Li H, Lloyd A, Melnik S, Mwaura D, Nagle D, Quinlan S, Rao R, Rolig L, Saito Y, Szymaniak M, Taylor C, Wang R, Woodford D (2012) Spanner: Google's globally-distributed database.

[24] VoltDB Inc (2013) VoltDB Technical Overview. 1–4. http://voltdb.com/downloads/datasheets_collateral/technical_overview.pdf