Comparative Analysis of Nosql Specimen with Relational Data Store for Big Data in Cloud

Sangeeta Gupta*

Abstract

The massive amount of data collected by various fields is a challenging aspect for analysis using the available storage technologies. Relational databases are a traditional approach of data storage more suitable for structured data formats and are constrained by ACID properties. As the modern world data in the form of word documents, pdf files, audio and video formats is unstructured, where tables and schema definition is not a major concern. Relational databases such as Mysql may not be suitable to serve such Bigdata. An alternate approach is to use the emerging Nosql databases. This paper presents a comparative analysis of Nosql types such as Hbase, Mongodb, Simple DB and Big Table with relational database like Mysql and specifies their limitations when applied to real world problems. It also proposes solution to overcome these limitations using an integrated data store which serve to be beneficial over the mentioned Nosql and Mysql stores in terms of efficiently implementing simple and complex queries yielding better performance.

Keywords: Mysql, Nosql, Big Data, HBase, MongoDB, Simple DB, Big Table, Integrated Store

Introduction

Cloud computing has evolved as a new computing paradigm, allowing end users to utilise the resources on a demand-driven basis, unlike grid and cluster computing which are the traditional approaches to access the resources. Enormous amounts of data are flooded across the internet and the storage capacities of the relational technologies have experienced inadequacy for the same. To store peta bytes of data, most of the organisations, particularly social networking sites and e-commerce sites are moving towards cloud to deploy their applications, but at increased security risks. This growing amounts of data which is too big and complex to capture, store, process, and interpret is referred to as Big Data (Venkat, 2014).It is characterised by 4 V's such as Volume, Velocity, Veracity and Variety. The storage and analysis of such data can be made effective using the Nosql databases.

The foremost benefit of cloud is to pay only for the resources which users utilise. If there is an unexpected set of users bombarding for the resources, they would just have to pay for what they have been using. This usage is termed as elasticity of the cloud. Cloud provides a variety of service models such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), Database as a Service (DaaS) and deployment models such as public, private, hybrid, and community clouds. An application to be hosted on a scalable environment can use either of these models in a cost efficient manner to reap their benefits. The other benefits provided by cloud can be utilised in terms of elasticity, scalability, efficiency, reusability (Thomas, 2014).

Most of the modern world data is projected in the form of word documents, pdf files, audio and video formats and relational databases may not be suitable to serve such data. Also, using them for scalable applications impose heavy costs making them less attractive for deploying large scale applications in cloud (Divyakanth, 2011). An alternate approach is to use the emerging Nosql databases, which are not ACID compliant and which provide support to structured, unstructured, and semi-structured storage of massive data in terms of peta bytes. Nosql databases don't rely on a fixed schema, there are no join operations

[*] Jawahar Lal Nehru Technological University, Kakinada, Andhra Pradesh, India. Email: ss4gupta13@gmail.com

and they rely on CAP (Consistency, Availability, and Partitioning) features in contrast to the ACID properties supported by traditional databases. Eventual consistency is supported by a few Nosql types, where the updates may not propagate immediately across all the nodes in a cluster.

This paper presents a comparative analysis of various Nosql types such as Hbase, Mongodb, SimpleDB, and BigTable with Mysql and highlights the limitations of Mysql and Nosql types thereby presenting suggestive mechanisms to overcome these limitations, which was not discussed in any of the previous works. It also presents a novel integrated Nosql, Cassandra which is advantageous and efficient over the mentioned Nosql species, amalgamating their benefits and crossing out their shortcomings in terms of performance and scalability over the estimated read and write operations on the database for execution of simple and complex queries.

The paper is organised as follows. The second section presents background study of the work including the scenarios where the mentioned databases are being used, third section presents a detailed comparative analysis with suggested solutions and integrated store developments with performance estimation on read and write operations on simple and complex queries, and fourth section concludes the work.

Related Work

In this part of the section, several related works on mysql and nosql types are discussed and their limitations are observed. Mysql has been used as a prominent relational database for storing data samples in a wide variety of applications. Fadzlina (2011) has used mysql to store finger prints data for biometric, with the help of a virtual server. Tables created were person identification number, real end-points data and real branch-points data, which employ structured data storage. If the amount of information collected is drastically increased, this would require large number of tables to accommodate the growing data and also if the data storage is in form of text or image format rather than pixel data as in (Fadzlina, 2011), then usage of mysql will become inappropriate.

Sudhanshu (2014) has compared the performance of mysql with DB4o database on sample hospital dataset and showed that object-oriented databases such as DB4o are always better as compared to relational databases such as mysql in terms of time taken to persist the data in the events of huge amounts of growing records. Though object-database deals well with respect to the huge data, they occupy large storage space.

The column-oriented data store HBase is a distributed database developed on top of Hadoop Distributed File System (HDFS), which adopts master-slave architecture with Name Node acting as a Master and Data Nodes acting as slaves. Mehul (2011) used the Nosql database HBase to perform random reads and writes on very large datasets in the form of image files and the results were proved to be better than using mysql on such data. Though the performance of HBase was shown to be better than mysql, Sudhanshu (2014) presented the model to be appropriate to perform write-once read-many operations on the attributes, but not suitable to support multiple write operations i.e., the files in HDFS were accessible efficiently in read mode but does not support multiple writes. Also, another limitation observed was, in order to use HBase, an in-depth understanding about Hadoop framework, Map-Reduce Programming model is required.

Gansen (2013) presented a comparison of Mongo DB, a document-based Nosql store with Mysql, highlighting the exceptional features of mongo DB like support for dynamic schemas, faster data integration, support for adhoc queries, load balancing and automatic sharding and also depicted the support of mongo DB at relational calculus, achieving better performance than mysql. But the limitation of this approach is that there is no expertise in this area and no specialist tools are available to analyze data efficiently.

Shalini (2011) performs a comparison of the databases: Amazon's Simple DB and Google's BigTable and motivates researchers to pick an appropriate one, highlighting their advantages and limitations. Simple DB, a hosted cloud based web service provides core functionality for storing and querying data in cloud. It does not rely on a predefined schema, like in relational databases and provides support to structured databases, but is inefficient for running complex queries including multiple join operations. It also does not assure the data integrity aspect of security providence. Google's BigTable on the other hand, stores structured data in a distributed fashion unlike simple DB which organises data in a centralised way. It is scalable without imposing the requirement for the nodes to be reconfigured, but it is not an open source and does not support any query language.

Table 1: Comparison of Relational with Nosql Databases

Database	*Supporting Features*			
	Consistency	*Scalability*	*Data Format*	*Joins*
Relational (mysql)	Strict	Vertical	Structured	Complex tasks require joins
Nosql(Hbase, Mongodb, Simpledb,cassandra)	Strict/eventual	Horizontal	Structured, semi-Structured, unstructured	No joins are to be performed

In this section, apart from identifying various areas of application of both Relational and Nosql databases, their limitations are brought into consideration and solutions are presented in Section III to overcome these limitations.

Big Data in Cloud using Nosql

Big Data is a term used to refer to massive and complex datasets made up of a variety of data structures including structured, semi-structured and unstructured data characterised by volume, variety, velocity aspects. The set of requirements imposed by cloud environment scale linearly in relation with the Big Data strategies and Nosql databases.

Though relational databases have occupied a higher position with respect to the data storage, their usage led to certain limitations like slow reading and writing, limited capacity and expansion difficulty. To solve these difficulties, a variety of new databases in form of Nosql types emerged. The main advantages of Nosql types are they support mass storage and they are easy to expand. The various categories of Nosql databases are column-oriented store like Hbase, key-value store like Redis, Document store like MongoDB and Graph-based like Neo4j (Jing, 2011). All these are based on Brewer's CAP Theorem (Consistency, Availability, Partitioning), which states that any Nosql data store will possess any two of these properties at a time, but not all three. All the Nosql types are used to analyze Big Data with most of them as open-sources that use cloud's DaaS to acquire supporting infrastructure.

Comparison of Nosql with Mysql

Relational databases are confined to ACID properties in contrast to Nosql's CAP properties. Nosql databases support both strict and eventual consistency, in which changes need not propagate instantly across all the nodes in a cluster as compared to the relational databases, which provide support only to strict consistency. Nosql serves horizontal scalability aspect than the vertical scalability of relational model as in mysql (Katarina, 2013). In all the Nosql data stores, a most prominent factor related to the scalability aspect is replication, which implies storing same data on multiple servers serving provision of fault-tolerance withstanding the failure of one or more servers. Replication also improves system reliability. There are two main approaches to replicate data. They are: master-slave and multi-master replication. In master-slave replication, a single node is designated as a master and it is the only node that processes write requests. Changes are propagated from master to the slave nodes. Example data stores that use this kind of replication are: HBase and MongoDB .In multi-master replication, multiple nodes can process write requests, which are then propagated to the remaining nodes. CouchDB is an example of multi-master replication. Table 1 depicts differences between relational and Nosql databases in terms of features like consistency, scalability, join operations, and data formats.

The comparison between relational and Nosql databases can also be discussed with respect to the complex query formulation and execution (Zhiyun, 2014). In relational model, operations on Cartesian products generate large amounts of valueless intermediate results, leading to low query efficiency. This feature makes them inefficient to analyze voluminous amounts of Big Data. Oracle released a solution in form of Mysql Cluster to address performance issues in scenarios of concurrent access to data, but this cluster stores data in RAM, disabling the provision of complete horizontal scalability at persistent storage level. To overcome these problems with relational models like mysql, Big Data uses Nosql databases provides support to various challenging aspects like frequent needs to changes in the data model in a transparent way, ensuring efficient data collection for faster performance, ensuring elastic behaviour of hardware infrastructure based on load, support to high read and write throughputs. All these features can be implemented with low cost commodity hardware. Key challenges in adoption of Big Data technologies for addressing Enterprise Data Management requirements are interoperability, manageability, security,

Table 2: Limitations of Mysql and Nosql Databases and Proposed Solutions

Database	*Application Where Used*	*Limitations Observed from Related Work*	*Suggested Solutions to Overcome the Mentioned Limitations*
Mysql	Used to store Finger prints data, and is used in integration with PHP in several applications for effective result generation.	Unsuitable for unstructured/semi-structured data storage, and inability to share workloads across multiple mysql servers.	Changes to the custom code or database architecture should be avoided to overcome limitations with scalability.
HBase	Used on any sample dataset to perform random reads and writes in an efficient way.	Not suitable for multiple write operations and requires in depth knowledge about hadoop framework and map reduce, this is too time consuming.	Data analysis tools like pig and hive can be used which automatically generate map-reduce code and provides an efficient way of analyzing data for a beginner.
MongoDB	Used to Implement relational calculus, textbook management database.	Does not perform well with respect to aggregate queries and there is no expertise to sort the same.	Performance of aggregated queries can be improved by using Map-Reduce on a sharded database.
Simple DB	Used for efficiently storing the requirements of new web-based applications.	Does not support unstructured data and complex query development using join operations. Also does not assure data integrity.	Multiversion timestamps may be designed to support complex queries with join operations.
BigTable	It was designed to support applications aiming at the maintenance of chronological queries and faster response times.	Inadequate access control and complex schema definition.	Design of a query language like SQL will enable efficient data storage and retrieval, along with access control facility.

maturity, development scalability, and maintainability (Subhankar, 2014). Though the future of Big Data looks very promising as it analyzes all kinds of unstructured data, its adoption by enterprise is still at infancy.

Table 2 summarizes the limitations of the papers taken in related work section and presents suggestions to overcome those limitations as a part of proposed work.

Cassandra-An Integrated Data Store

Considering the limitations and suggested solutions as in Table 2, a novel integrated Nosql, Cassandra is used which aims at providing support to any kind of data (structured, semi-structured or unstructured) as emerging from the real-world social networking websites such as facebook, and e-commerce sites such as eBay concatenating the scalability aspects of Big Data, leading to eventual consistency and providing an efficient way to solve complex queries by avoiding join operations. Cassandra is used to overcome the limitations possessed by the mentioned data stores, integrates the benefits of Mysql, Hbase, MongoDB and Simple DB data stores. Cassandra uses peer-to-peer architecture, where all nodes are given equal priority in a cluster and the nodes are said to communicate with each other through gossip protocol. There is no single point of failure in Cassandra; hence there is no down time for running an application. The query language used to perform operation with the database is CQL (Cassandra Query Language).

Querying Differences

Relational databases like Mysql, Oracle etc., use SQL for storing, retrieving and manipulating data, whereas in nosql types, there is no single standard query language to meet varying user's requirements. Querying data stored in nosql databases is specific to the data model. So, each nosql comes with its own query language like SimpleDB has SQL (SimpleDB Query Language), Cassandra has CQL, HBase has HQL.

To explain about the differences among Mysql, SimpleDB, Hbase, Cassandra, we have considered sample tables titled journal and conference. The syntax's used by various data stores for the tables vary as shown in Table 3 to perform insert, update and delete operations. We have also taken simple and complex queries to analyse these differences for data retrieval operation (select) as in Table 4 and Table 5.

Table 3: Querying Differences between Mysql and Nosql with Insert, Update and Delete Operations

Database	Insert Operation	Update (Write) Operation	Delete Operation
Mysql	Insert into journal values('ieee',1234, 'openaccess);	Update journal set jid=1234 where jid=1345;	Delete from journal where jname='mnuoo';
HBase	Put 'journal','row1','jid:a','ieee';	Same as insert	Disable 'journal';
MongoDB	Db.journal.insert({jname:"ieee",jid:1234,accesstype:"openaccess"})	Db.journal.update({},{'$set':'jid':'jid'}});	Db.journal.remove();
SimpleDB	?action=PutAttributes &DomainName=Journal&attribute.1.name=jname&attribute.1.value=ieee&attribute.2.name=jid&attribute.2.value=1234&attribute.3.name=accesstype&attribute.3.value=openaccess	Same as in insert	action=DeleteAttributes &DomainName=Journal&attribute.1.name=jname&attribute.1.value=abcd
Cassandra	Insert into journal values('ieee',1234, 'openaccess);	Update journal set jid=1234 where jid=1345;	Update journal set jid=1234 where jid=1345

Table 4: Simple Query: Find the Name of Journal with ID 1234

Database	Retrieval Operation
Mysql	Select j.jname from journal j where j.jid=1234;
HBase	Get 'journal','jname';
MongoDB	Db.journal.find({},{"jname":1,"jid":0,"jtype":0});
SimpleDB	Select jname from journal where jid=1234;
Cassandra	Select jname from journal where jid=1234;

In the above example, the syntax for retrieving (reading) data from Mysql, SimpleDB and Cassandra are similar, while in MongoDB find is used to retrieve the data, and in HBase, get is used for the same.

Table 5: Complex Query(Joins/Nested): Find the Journal Names Whose IDS Match with that of the IDS in Conference Table

Database	Retrieval Operation
Mysql	Select j.jname from journal j where jid in(select c.jid from conference c).Here jid in conference table is a foreign key.
HBase	Get command can't be used to run the same query as in mysql, but if integrated with mapreduce code, the query will be executed.
MongoDB	As in HBase, MongoDB also requires mapreduce command integration.
SimpleDB	Does not support such operations but enables an attribute to possess multiple values to eliminate the use of join operation.
Cassandra	Super column families and data denormalisation can be done to execute complex queries in an efficient way.

To perform complex joins or nested queries, Mysql requires foreign keys to be created performing joins across multiple tables. But, this method may lead to increase in execution time in order to retrieve data from multiple tables, there by degrading the overall performance. In HBase, complex joins are supported by integrating Hbase code with map reduce code using nested loops, which is again a time consuming process. MongoDB also uses mapreduce command to process such data. SimpleDB avoids joins by enabling an attribute to hold multiple values, and due to this, the results retrieved may not yield correct set of answers to the query.

In Cassandra, performing complex joins or nested queries requires denormalisation of data into partitions, leading to efficient querying from a single replica node, rather than gathering the data from across the entire cluster. Thus, it provides an efficient mechanism to retrieve data in a simpler way, and also the speed of query execution is much better than in Mysql, SimpleDB, MongoDB and Hbase.

Indexes are created on the tables to speed up the performance. The process of index creation required a third-party module in Hbase while in MongoDB, SimpleDB and Cssandra, it is a built-in feature.

CONCLUSION

Most of the organisations rely on structures databases like Mysql, which do not harness the requirements of scalability and availability of real-world data. The available set of Nosql databases support various aspects to meet the upcoming trends in growing data like support for eventual consistency, scalability, availability, and fault-tolerance. In this paper, Nosql databases Hbase, MongoDB and SimpleDB are discussed and apart from mentioning their advantages as compared to Mysql, their limitations are also presented and solutions are suggested to overcome the limitations, heading towards the adoption of Integrated Nosql database-Cassandra. Modern world requirements in form of Big Data can be efficiently analysed and interpreted using this integrated Nosql database with respect to query analysation. The Future work can be taken up to conduct more experiments including insert, update and delete operations and comparing the performance with benchmarks.

References

Agrawal, D., Das, S., & El Abbadi, A. (2011). *Big data and cloud computing: Current state and future opportunities*. Proceedings of the 14th International Conference on Extending Database Technology.

Dhar, S., & Mazumdar, S. (2014). *Challenges and best practices for enterprise adoption of big data technologies*. IEEE Technology Management Conference.

Grolinger, K., Higashino, W. A., Tiwari, A., & Capretz, M. A. M. (2013). Data management in cloud environments: NoSQL and NewSQL data stores. *Journal of Cloud Computing: Advances Systems and Applications*, 22(2), 1-41.

Gudivada, V. N., Rao, D., & Raghavan, V. V. (2014). *NoSQL systems for big data management*. In IEEE Computer Conference on Big Data: Promises and Problems, 48(3), 190-197.

Han, J., Haihong, E., Le, G., & Du, J. (2011). *Survey on NoSQL database*. 6th International Conference on Pervasive Computing and Application, (pp. 363-366).

Kulshreshta, S., & Sachdeva, S. (2014). *Performance comparison for data storage-db4o and mysql databases*. 7th International Conference on Contemporary Computing.

Naim, N. F., Yassin, A. I. M., Zamri, W. M. A. W., Sarnin, S. S. (2011). *My SQL Database for storage of finger print data*. 13th International Conference on Computer Modelling and Stimulation, (pp. 293-298).

Ramanathan, S., Goel, S., & Alagumlai, S. (2011). Comparison of cloud database: Amazon's simple db and Google's big table. *International Journal of Computer Science Issues*, 8(6), (pp. 243-246).

Sandholm, T., & Lee, D. (2014). Notes on cloud computing principles. *Journal of Cloud Computing: Advances, Systems and Applications*, 21(3), 1-10.

Vora, M. N. (2011). *Hadoop-HBASE for large-scale data*. International Conference on Computer Science and Network Technology, (pp. 601-605).

Zhao, G., Huang, W., Liang, S., & Tang, Y. (2013). *Modelling mongo DB with relational model*. 4th International Conference on Emerging Intelligent Data and Web Technologies, (pp. 115-121).

Zheng, Z., Du, Z., Li, L., & Guo, Y. (2014). *Big data oriented open scalable relational data model*. In IEEE International Congress on Big Data Congress (Big Data Congress), 398-405.

Venkat, N. (2014). What's the future of the data center? The big list of thought leadership perspective. Silicon Angle.

Thomas S., Dongman, L. (2014). Notes on Cloud Computing Principles. *Journal of Cloud Computing: Advances, Systems and Applications, Springer*.

Agarwal, D., Das, S., & EI Abbadi, A. (2011). Big data and Cloud Computing: Current State and Future opportunities.

Grolinger, K., Higashino, W. A., Tiwari, A., & Capretz, M. A. M. (2013). Data management in cloud environments: NoSQL and NewSQL data stores. *Journal of Cloud Computing*, 2(1), 22.

Naim, F. & Yassin, I. M. (2011). *MySQL Database for Storage of Fingerprint Data*. Proceedings of the 13th UK Sim-AMSS International Conference on Computer Modelling and Simulation, Cambridge University, Emmanuel College, Cambridge, UK

Kulshreshta, S., & Sachdeva, S. (2014). Performance Comparison for Data Storage-DB4o and Mysql Databases.

Vora, M. N. (2011). Hadoop-H-Base for Large Scale Data. IEEE 2011, (pp. 601-605).

Zhao, G., Huang, W., Liang, S., & Tang, Y. (2013). Modelling Mongo DB with Relational Model. IEEE, (pp. 115-121).

Ramanathan, S., Goel, S., & Alagumlai, S. (2011). *Comparison of Cloud Database: Amazon's Simple DB and Googles BigTable*. IEEE 2011 and International Journal of Computer Science Issues (IJCSI), November, 8(6).

Han, J., Hong, H. E., Le, G., & Du, J. (2011). Survey on NoSQL Databases. IEEE 2011, (pp. 363-366).

Zheng, Z., Du, Z., Li, L., & Guo, Y. (2014). *Big Data Oriented Open Scalable Relational Data Model*. IEEE Conference on Big Data Congress,)pp. 398-405).

Dhar, S., & Mazumdar, S. (2014). *Challenges and best practices for Enterprise Adoption of Big Data Technologies*. IEEE International Conference on Technology Management, (pp. 1-4).