International Journal of Allied Practice, Research and Review

Website: www.ijaprr.com (ISSN 2350-1294)

Relational and Non-Relational Database Constraints for Big Data Management

Behjat U Nisa[1] and Er. Kiran Gupta[2]
[1]MTech Scholar, Department of CSE, Swami Devi Dyal Group of Professional Institutes,Haryana, India.
[2]Assistant Professor, Department of CSE, Swami Devi Dyal Group of Professional Institutes, Haryana, India.

Abstract-Databases are used for storing and managing large amounts of data. Relational model is useful when it comes to reliability but when it comes to the modern applications dealing with large amounts of data and the data is unstructured; non-relational models are usable. NoSQL databases are used to store large amounts of data. NoSQL databases are non-relational, distributed, open source and are horizontally scalable. This paper provides the a data model which uses both relational and NoSQL database systems in combination for the storage and management of extremely voluminous data of diverse components known as big data.

Keywords: SQL, NoSQL, ACID, CAP, Big Data.

I. Introduction

Relational model is useful when it comes to reliability but when it comes to modern applications dealing with large amounts of data and the data is unstructured; non-relational models are usable. A huge challenge these days is the management and analysis of Big Data. Large amounts of data generated by many organizations are rendered useless due to the inefficiency of handling big data. However, data is a great resource. This data generated is very useful for the purpose of research. In order to handle the new data management requirements thus NOSQL databases which are non-relational databases are used. Some core features of database system such as ACID have been compromised in NOSQL databases.

This work proposes the use of both relational and NoSQL database systems in combination for the storage and management of extremely voluminous data of diverse components known as big data. The two models are integrated in one system to eliminate the limitations of the individual systems. The system is implemented in MongoDB which is a NoSQL database and SQL. The results obtained, revealed that having these two databases in one system can enhance storage and management of big data bridging the gap between relational and NoSQL storage approach.

II. Relational Database and NoSQL Database Storage Approach

Relational Databases store data in rows and columns. In relational model, data is represented in tables or relations. A table as shown below is primarily a collection of related data entries and it is made up of several columns and rows referred to as fields and records respectively. Almost all databases built on relational model guarantee Atomicity Consistency Isolation and Durability transactions.

ID	Name	Age	Semester
1001	Jane	20	First
1002	Thomas	25	Third
1003	Catherine	22	First
1004	Joe	23	Third

NoSQL is a non-relational database. NoSQL databases are horizontally scalable databases. These databases differ from relational databases in a number of ways. No SQL databases can handle unstructured data such as documents, e-mail, multimedia and social media unlike the relational databases. These databases do not use tables as storage structures. They do not ensure the ACID properties instead they guarantee the BASE properties.

NoSQL databases can be classified into five types:

1. Key-value store databases
2. Column oriented databases
3. Document store databases
4. Graph databases
5. Object oriented databases

Each database is individually good at dealing with size and complexities.

III. Choosing an Appropriate Data Solution

There are a lot of NoSQL databases exists in the market today. Users sometimes get confused to choose a system for their application. Computer scientist Eric Brewer in his CAP theorem that is stated above provides a perfect solution for this problem. The choice of the database selection depends on the characteristics of the application. From the figure, it can be seen that if a system needs availability and consistency then it should use relational databases. NoSQL databases like MongoDB, Scalaris, BerkeleyDB and Rediscon firm consistency and partition tolerance. On the other hand, Cassandra, CouchDB, Dynamo etc. provide availability and partition tolerance.

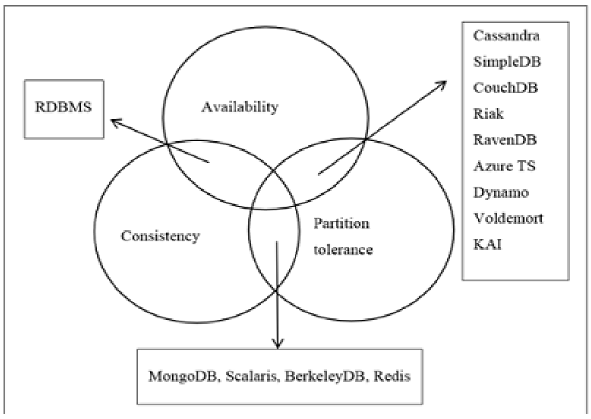


Fig 1: CAP Theorem

IV. Big Data

The term **"Big Data"** was first used by the NASA researchers Michael Cox and David Ellsworth (Friedman, 2012). Big data is the amount of data which is too big to process in a single machine. Such data is created very fast and it comes from different sources with different formats. Big Data is the large and complex data that is difficult to use the traditional tools to store, manage, and analyze in an acceptable duration. The universe is full with smart devices that generate a huge volume of data and enterprises are dealing with massive amount of data. The data generated by these devices is very essential for an enterprise to compete with other business vendors. Big data does not only refer to collection of voluminous data, but extremely large volume of structured and unstructured data subject to very high rate of change, derived from divers avenues which may include email, social media, phone calls etc. There are several challenges that enterprises face while dealing with Big Data like:

- The security of Big Data which contains very sensitive information, personal information and intellectual property.
- There may be delay in speed and availability requirements of Big Data when security mechanisms are operated.
- How Big Data is always safe and secure irrespective of where it is stored.

CHARACTERISTICS OF BIG DATA:

- **Volume:** in present world the data being stored is exploding. Every action in present era generates data means the whole world is full with data generated from so many sources. Organizations face massive amount of data and sometimes happen to a serious problem for enterprises if not know how to manage this huge quantity of data but there are best technological options to manage and analyze this data to get a better understanding of business.
- **Variety:** Present world is full with smart devices and sensors that results in data complexity that organizations face because data is not only the structured but also semi-structured and un-structured. The future of organization is based on analysis of data but whole including structured, semi-structured and un-structured but as for survey we are targeting only structured that is only the 20 percent that is formatted very well and fits nicely in schemas and 80 percent of data in the world is semi-structured and un-structured. For future it is necessary for enterprises to analyze data of all types both relational as well as non-relational.

- **Velocity:** Velocity is not only the growth rate of data repositories but also applicable to the concept of data in motion means the speed at which data is stored and retrieved. Enterprises are dealing with huge amount of data even in petabytes because of deploying different types of sensors and smart devices that is very difficult to manage by traditional systems.

- **Veracity:** Big Data Veracity refers to the biases, noise and abnormality in data. Refers to the truthfulness of the data. It deals with the relevance of the data (processed or analyzed) to the task at hand. Veracity is the biggest challenge when compared to volume and variety. It reveals the need to avoid accumulation of dirty data.

- **Volatility:** Big Data volatility refers to how long is data valid and how long should it be stored. In this world of real time data we need to determine at what point is data no longer relevant to the current analysis. In other words we can say that it deals with the reasonable life span of stored data in the world of real time data processing. It investigates the validity of stored data to the current analysis.

V. Proposed Solution

The aim of this work is to deploy relational database and NOSQL database in combination for handling and management of Big Data. The combinational database system hence proposed uses MongoDB and Microsoft's SQL Server. The data is categorized into structured and unstructured before storing it. Structured data is then stored in the SQL Server database which is our relational database. This structured data is also managed by SQL Server. The unstructured data on the other hand is stored and managed by the MongoDB database. We can make the data storage either together (i.e. structured in SQL Server and unstructured in MongoDB) or separately (i.e. store data structured data in SQL Server and ignore the unstructured data or store both structured and unstructured data in MongoDB).

DESIGN:

There are two components namely the relational component (SQL Server) and the Non-Relational Component (MongoDB).

- Relational Component: is used to store the structured data.

- Non-Relational Component: is used to store and manage the unstructured data.

ALGORITHM:

The step-by-step procedure for data processing in our proposed system for Big Data management is given as;

1. Load the Data.

2. Initialize our interface

3. Test the data: If Data is structured then store it in SQL Server else if the data is unstructured we store it in MongoDB.

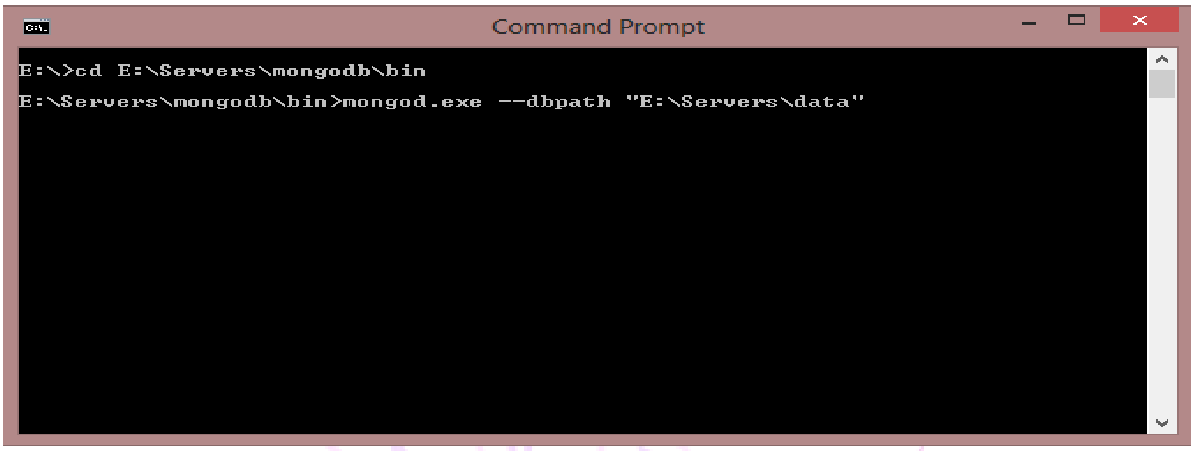
4. Display Acknowledgements.

5. View, Delete, Update, Exit

VI. Experimental Setup and Results

The proposed system is implemented in JAVA using Net Beans Integrated Development Environment. MongoDB and SQL Server were used for data storage.

The MongoDB server would first be started. By default, MongoDB server will start at port 27017. Here we open the command prompt. Change the directory to the one which holds our MongoDB server. Give the path to the bin folder. Then enter the command.**mongod.exe –dbpath "path to data folder".**



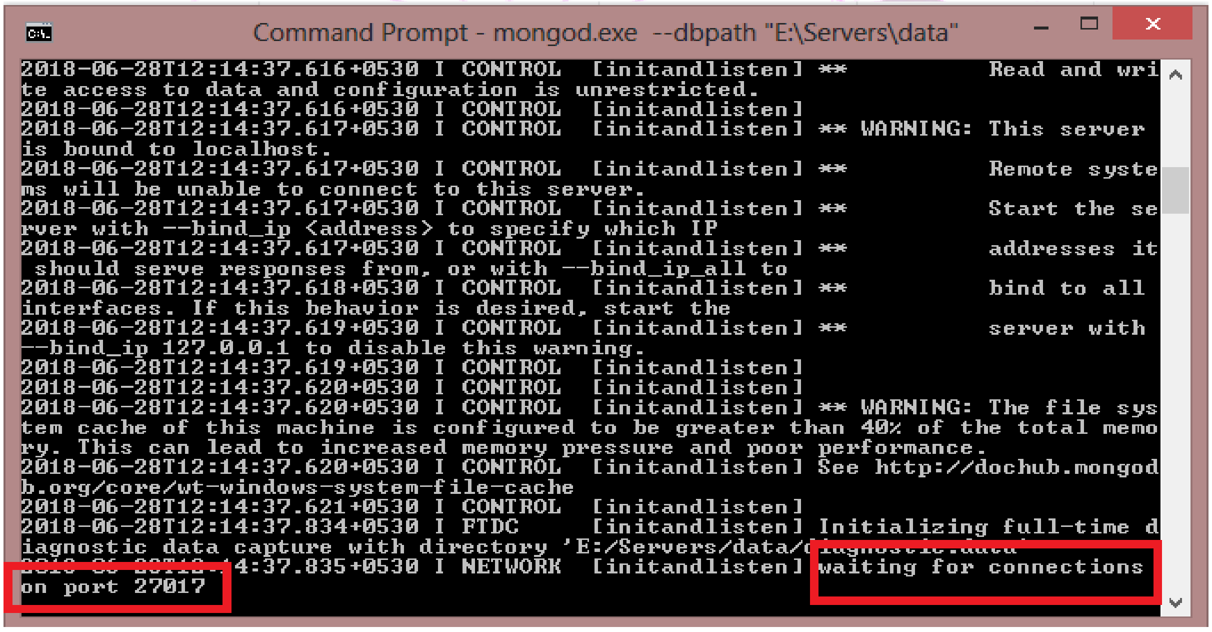
```
Command Prompt

E:\>cd E:\Servers\mongodb\bin

E:\Servers\mongodb\bin>mongod.exe --dbpath "E:\Servers\data"
```

Fig 2: Connecting to MongoDB Server.

After this we hit enter. And it shows waiting for connections which is shown in the figure that follows.



```
Command Prompt - mongod.exe  --dbpath "E:\Servers\data"
2018-06-28T12:14:37.616+0530 I CONTROL  [initandlisten] **          Read and wri
te access to data and configuration is unrestricted.
2018-06-28T12:14:37.616+0530 I CONTROL  [initandlisten]
2018-06-28T12:14:37.617+0530 I CONTROL  [initandlisten] ** WARNING: This server
is bound to localhost.
2018-06-28T12:14:37.617+0530 I CONTROL  [initandlisten] **          Remote syste
ms will be unable to connect to this server.
2018-06-28T12:14:37.617+0530 I CONTROL  [initandlisten] **          Start the se
rver with --bind_ip <address> to specify which IP
2018-06-28T12:14:37.617+0530 I CONTROL  [initandlisten] **          addresses it
 should serve responses from, or with --bind_ip_all to
2018-06-28T12:14:37.618+0530 I CONTROL  [initandlisten] **          bind to all
interfaces. If this behavior is desired, start the
2018-06-28T12:14:37.619+0530 I CONTROL  [initandlisten] **          server with
--bind_ip 127.0.0.1 to disable this warning.
2018-06-28T12:14:37.619+0530 I CONTROL  [initandlisten]
2018-06-28T12:14:37.620+0530 I CONTROL  [initandlisten]
2018-06-28T12:14:37.620+0530 I CONTROL  [initandlisten] ** WARNING: The file sys
tem cache of this machine is configured to be greater than 40% of the total memo
ry. This can lead to increased memory pressure and poor performance.
2018-06-28T12:14:37.620+0530 I CONTROL  [initandlisten] See http://dochub.mongod
b.org/core/wt-windows-system-file-cache
2018-06-28T12:14:37.621+0530 I CONTROL  [initandlisten]
2018-06-28T12:14:37.834+0530 I FTDC     [initandlisten] Initializing full-time d
iagnostic data capture with directory 'E:/Servers/data/diagnostic.data'
2018-06-28T12:14:37.835+0530 I NETWORK  [initandlisten] waiting for connections
on port 27017
```

Fig 3: Waiting For Connections.

To start the client shell, the command mongo.exe is executed on a separate command prompt window.

To execute our model, the input data which is made up of structured and unstructured data is saved in the system's local disk. When big data is loaded in the system, the database used for storage is determined by the mode in which the application runs in. Unstructured data is channeled to the MongoDB database (except when the application is in MongoDB mode in which case MongoDB stores both structured and unstructured data) while SQL Server database is used to store and manage structured data. The architecture of the model and the interface and insert operation are shown in the figures that follow.

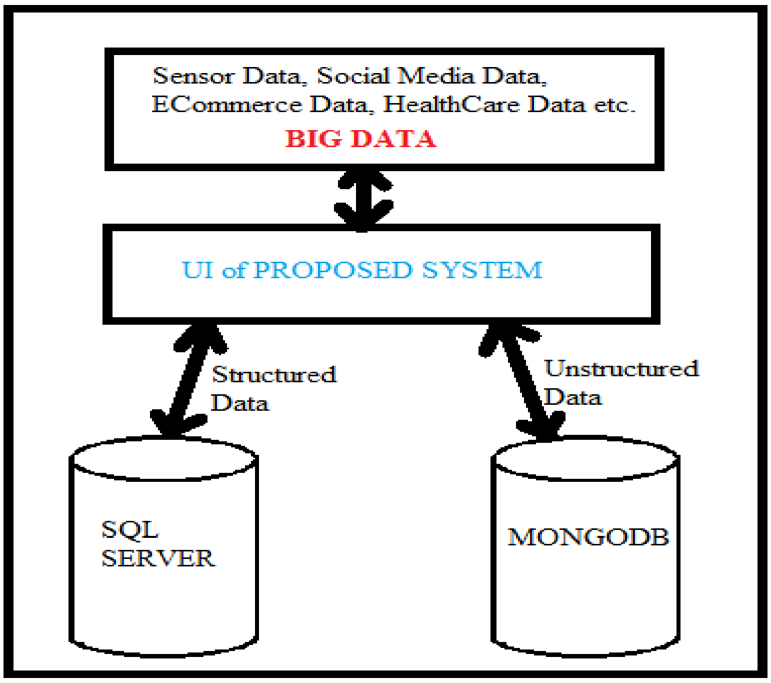


Fig 4: Architectural Design.

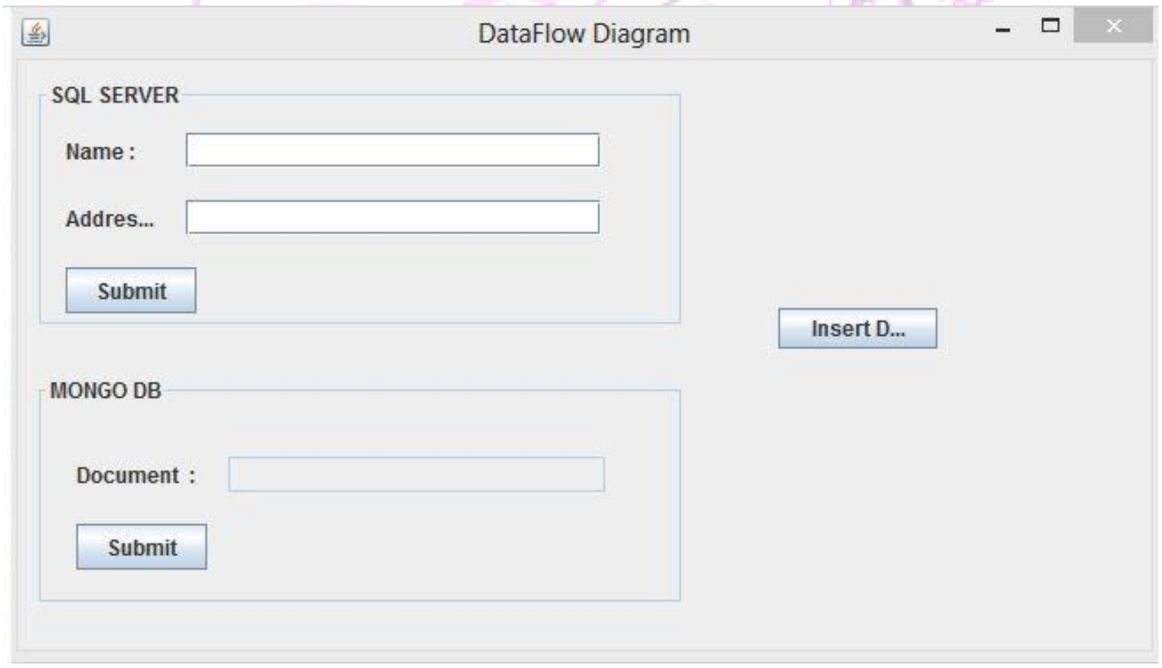
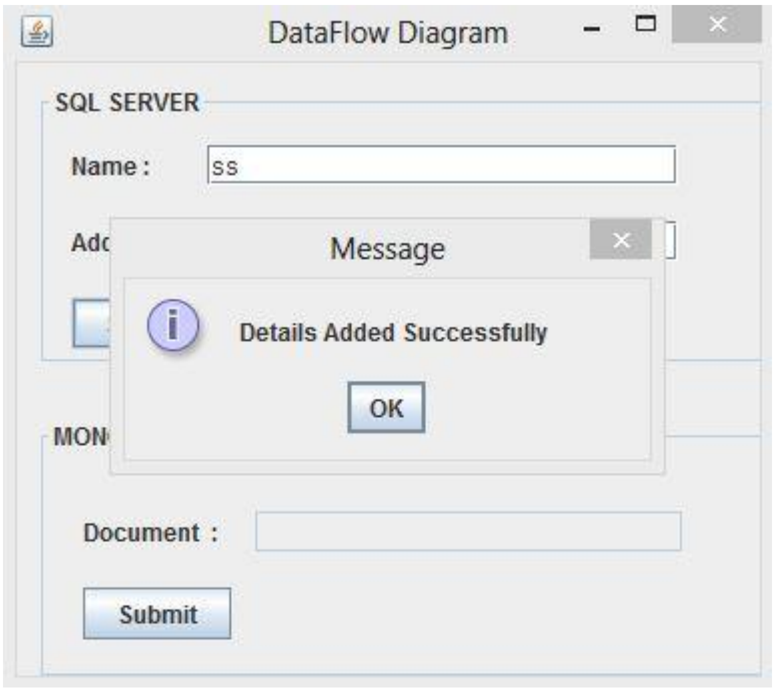


Fig 5: Overview of Interface


DataFlow Diagram

SQL SERVER

Name : ss

Message

Details Added Successfully

Fig 6: Insert Operation

Data was loaded in our system in three modes as discussed:

First the data is stored in SQL Server. This system discards data which is unstructured. Second the data is stored and managed using MongoDB. This works on both structured and unstructured data. Then we follow our algorithm using which we are able to store our data in both the components that is the structured data in SQL Server and unstructured data in MongoDB.

VII. Conclusion

In conclusion, we proposed a method which combines SQL Server database which belongs to the relational group of database systems and MongoDB being a NoSQL database to store and manage big data. With the result obtained it is understandable that our system can be used for storage and management of big data eliminating the weaknesses in both databases.

VIII. References

1. **Eric A. Brewer, "Towards Robust Distributed Systems" Portland, Oregon, July 2000. – Keynote at the ACM Symposium on Principles of Distributed Computing (PODC) http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf**

2. **Chris Snijders, Uwe Matzat and Ulf-Dietrich Reips, "Big Data: Big Gaps of Knowledge in the Field of Internet Science", International Journal of Internet Science, 2012.**

3. **Sultana Kalid, Ali Syed, Azeem Mohammad and Malka N. Halgamuge, "Big-Data NoSQL databases: A Comparison and Analysis of Big-Table, DynamoDB, and Cassandra", IEEE 2nd International Conference, 2017.**

4. **Shawn Graham, Ian Milligan and Scott B. Weingart "Exploring Big Historical Data - The Historian's Macroscope", London: Imperial College Press, 2016.**

5. **Tony Marston, "The Relational Data Model, Normalization and Effective Database Design", Internet: http://www.tonymarston.net/php-mysql/database-design.html, August 2005.**

6. **J. Porkony, "NoSQL databases: A step to database scalability in web environment." International Journal of Web Information Systems, Volume 9, 2013.**

7. **V. Abramova, J. Bernardino, P. Furtado, "Experimental Evaluation of NoSQL Databases" International Journal of Database Management System, Volume 6, 2014.**

8. **L. Wu, L. Yuan, J. You, "BASIC: An alternative to BASE for Large Scale Data Management Systems", https://webdocs.cs.ualberta.ca/~yuan/papers/ieee_bigdata.pdf 2014.**