Anthony Pichardo

Comp 440

Professor Mushkatblat

11 May 2017

Hadoop VS SQL Relational Databases

**Abstract:**

As various organizations and groups continue to collect more and more data, industries need to find new ways to manage that data. In this paper, we will be discussing utilizing distributed file systems like Hadoop as opposed to traditional relational databases like SQL. As the amount of data continues to grow, typical data storage units cannot scale at the same rate and the connections between them get more and more complicated. With Hadoop and similar tools, we can scale up much easier and process our data quickly in ways that were previously difficult. Both applications handle ACID and CAP properties differently and make different tradeoffs. There are pros and cons to both HDFS and Relational Databases, but I believe that as the amount of data continues to grow we will see more systems switch to HDFS or a new technology.

**Paper Summarization:**

A paper published by Seth Gilbert and Nancy Lynch, discussed the CAP theorem and the trade offs that make it impossible to achieve consistency, availability, and partition tolerance within distributed computing. Consistency means that every response sent to the client is the correct response. Availability is the idea that every request will receive a response. Partition tolerance refers to the ability to function when the network is split and are unable to communicate with one another. The paper then dives into the proof of these three principles. For traditional distributed systems it is impossible to be both consistent and partition tolerant because neither principle remains true if there is attempt to have both. They define three constraints that will allow for synchrony and deliver CAP, they are: all processes need a clock and all clocks are synchronized; every message is delivered within a fixed and known amount of time; and all processes take a fixed and known rate. Doing this we can allow the system to progress in rounds where processes will send and receive messages that were sent during that round. If this is done there is an increase in the time complexity of operations. The paper then goes into practical applications of CAP and most importantly to our discussion the idea of scalability. The authors believe there is an inherent trade off between scalability and consistency in database systems.1

Brahim Medjahed, Mourad Ouzzani, and Ahmed K. Elmagarmid of Purdue University published a paper discussing the generalization of ACID properties. The ACID properties guarantee the reliability of database transactions and they are Atomicity, Consistency, Isolation, and Durability. The generalization of these properties are Recovery, Consistency, Visibility, and Permanence. The generalization happened because it allows the properties to be applied to systems other than traditional business-oriented applications. Atomicity states that transactions must either fully or occur or no changes occur at all. Consistency states that transactions must always keep its integrity constraints. Isolation ensures that transactions are invisible to other transactions until it is completed. Lastly, durability states that once a transaction has been completed the effects will persist. There are limitations inherent to the original properties in advanced database applications. The generalized principles are defined as follows: Recovery is the ability of the database to revert to a correct state after failure; consistency is the correctness of the state that a transaction produces; visibility requires that a transaction be able to see the results of another running transaction; lastly permanence states that a transaction must be able to record its results in the database. The generalized properties are much more flexible for a variety of applications. The paper concludes by discussing the Flex Transaction Model to further generalize the ACID properties for multidatabase systems and enable users to control isolation via sub-transactions.2

The final paper for this assignment discussed the difference between structured and unstructured data which is a large factor when determining whether or not to use tools like Hadoop or SQL. This paper was published by the data science department of UC Berkeley. Structured data is data that has been sorted in columns and rows like most traditional databases. Unstructured data, in contrast, does not conform to specific data models, it’s typically human-generated; an example of this would be text messages or voicemails. They then go on to discuss Hadoop’s viability in handling unstructured data in ways that relational databases cannot--Hadoop is part of a series of tools known as NoSQL. Unstructured data is valuable to businesses because it allows company to perform analytics in terms of public sentiment. Unstructured data still needs to be organized in some form for it to be useful though. ETL techniques are used on NoSQL databases in order to make the data usable in business ready forms that tie the information to users.3

**NoSQL or Relational Databases and Conclusion:**

The final paper summarized brings up a great point. When should we use NoSQL data stores as opposed to Relational Databases? My opinion on this follows from the articles--NoSQL data stores should be used in the case that the data being collected is unstructured or the distributed compute clusters meet your scalability requirements and if your data is structured and not too large in volume then your needs can most likely be met by relational databases. Both databases are able to meet 2 of the 3 CAP properties and the generalized ACID properties so that need not necessarily be a concern unless you need to handle them in a specific manner. Unstructured data allows you to perform analysis that would otherwise be difficult because of its ability to provide semantic analysis. That information is very valuable to companies because it gives them insights to their products for free through the use of social media monitoring. Data is power and unstructured data multiplies the amount of data available for consumption 10 fold. Another benefit of NoSQL data stores like Hadoop is their ability to scale with ease through use of the cloud and multiple compute nodes. Relational Databases do not scale as easily because of the need to add new systems when the current ones are full.

As we continue to gather more and more data relational databases may lose their viability even for small companies--distributed file systems are the future of data storage and tools like Hadoop are currently a great option.

**References:**

[1] “CAP for Networks”. Aurojit Panda, Colin Scott, Ali Ghodsi, Teemu Koponen, Scott Shenker. UC Berkeley. N.D. Web. 10 May 2017.

[2] “Generalization of ACID Properties”. Brahim Medjahed, Morad Ouzzani, Ahmed K. Elmagarmid, Purdue University, N.D. Web. 11 May 2017.

[3] "Bridging the Divide between Unstructured and Structured Data." *Datascience@berkeley*.edu, N.p., 26 Feb. 2014. Web. 11 May 2017.