

BD Assignment by Wanzi Su

by Wanzi Su

FILE	17848_WANZI_SU_BD_ASSIGNMENT_BY_WANZI_SU_226876_852527973.DOCX (26.86K)		
TIME SUBMITTED	27-MAR-2020 11:03PM (UTC+0000)	WORD COUNT	3936
SUBMISSION ID	122998711	CHARACTER COUNT	20079

Contents

1. Introduction	1
2. Big Data	2
3. CAP Theorem.....	4
4. Relational Databases and NoSQL Databases.....	6
5. NoSQL Database for Zang Technologies	9
6. Conclusion	11
7. Reference	11

1. Introduction

This essay provides an overview of some basic terms like Big Data and CAP Theorem. Another purpose is to analyse some databases, e.g. relational databases, NoSQL databases, and MongoDB. To describe their characteristics, it is necessary to make a comparison and provide appropriate examples.

This essay has been divided into four parts. It first gives a brief overview of Big Data and the characteristics of it, which are also known as four V's. This is the first step before learning Big Data module so it is quite important.

It will then go on to CAP Theorem. This is a key theorem about the distributed database.

The next part is about relational databases and NoSQL databases. In addition to that, the acronyms ACID and BASE are specifically explained to describe the characteristics of those two databases. Database is a big topic in Big Data so it is really significant to understand different databases.

The last part is concerned with MongoDB, which is the chosen database for Zang Technologies. To explain why to chose MongoDB, a comparison between the benefits of MongoDB and the actuality of Zang Technologies is made. It is not enough to only gain knowledge of the database. To show that you understand it, you should know how to use it in reality.

2. Big Data

To explain the term “Big Data”, we must know first that “Big Data” is not actually data. It is not even a thing. It is a concept that describes a method or technology to store and manage a lot of data. “Big” means a massive number. How much data can be described as “Big”? There is no rule for that but the important thing is “Big Data” can deal with almost as much data as you wish. So that is the meaning of “Big”. Judith (2013) provides in-depth analysis of Big Data:

Big data is not a single market. Rather, it is a combination of data-management technologies that have evolved over time. Big data enables organizations to store, manage, and manipulate vast amounts of data at the right speed and at the right time to gain the right insights.[1]

Then what about the “Data”? What kind of data is going to be stored? The answer to this question would lead us to one of the four V’s: “Variety”. The data is structured data like SQL, semi-structured data like JSON, and unstructured data like video. There is even a formula for “Big Data” as the literature below.

Big data refers to massive volume of structured, semi-structured and unstructured data that is so large that it is very difficult to process using traditional data processing techniques. ... According to Connolly (2012), big data includes the following and can be formulized as follows:

Big Data = Transactions + Interactions + Observations

For example, transactions data are ERP data. Interactions data are tweets or facebook posts. Observations data come from Internet of Things (IOT) such as GPS, CCTV or monitoring data.[2]

In summary, “Big Data” is a technology that stores and manages a great deal of various data. The data is so much that it is hard for other techniques to operate.

As is mentioned above, the characteristics of “Big Data” are four V’s. In fact, according to the references, there are many different versions of V’s. The first version is three V’s: “Volume”, “Velocity”, and “Variety”. Some writers (e.g. Judith, 2013) holds the view that Big Data is broken down by three characteristics i.e three V’s.

As we note earlier in this chapter, big data is typically broken down by three characteristics:

Volume: How much data

Velocity: How fast that data is processed

Variety: The various types of data [1]

The second version is four V's. According to Delaney et al. (2017), there are four characteristics and an extra one. "Veracity" is the new one of the V's. In addition to that, "Variability" is the extra one and does not count in the four V's.

The predictable seeds for Big Data, which are characterized by data volume, variability, veracity, velocity, and variety, were planted by the "information explosion" (a term first used 70 years ago in 1941, Oxford English Dictionary).[3]

The third version is five V's. Other authors (see Joshi and Valentin, 2020) concluded that five V's are the characteristics of Big Data. "Value" replaces "Variability" and becomes one of the five V's.

Big Data also have five Vs characteristics, which are listed below: [2]

Volume (magnitude of data in multiple terabytes and petabytes) [2]

Variety (structural heterogeneity in a dataset) [2]

Velocity (the rate at which data are generated and the speed at which it should be analyzed and acted upon) [2]

Value (attribute of big data - low value density) [2]

Veracity (unreliability inherent in some sources of data) [2]

While there are different definitions of what is 'big data', depending on the context and perspectives, five key and interrelated 'Vs' are typically distinguished there: (i) Variety, (ii) Volume, (iii) Veracity, (iv) Velocity, (v) Value. Covering all these 'Vs' in one paper is of course infeasible.[4]

Here, the second version, four V's, is considered as the standard version. In this case, the four V's are "Volume", "Variety", "Velocity", and "Veracity".

“Volume” means how much data can be stored or managed. As technology develops, the volume of Big Data has changed from gigabytes or terabytes to petabytes or even beyond. For instance, a record “Tom, drinks, water” is 14 bytes so there can be a huge number of records like this in the Big Data.

“Variety” means how many kinds of data is in Big Data. Big Data cannot be called “Big Data” if the data is a single data. For example, if this data is about a single person or is submitted by a single person, it is not “Big Data”. Commonly, there are three types of data as is discussed above. Structured data can be presented in the same structure by numbers or characters. Semi-structured data is between the structured data and the unstructured data. Its structure and content are mixed. Unstructured data like videos or images is some data that cannot completely be presented in a structure.



“Velocity” means how fast can the data be stored or managed. The data is generated very quickly because of Internet transmission. But the difference is the speed at which the data is processed. As a result, a lot of resources in the server are used to analyse the data. If a company can make real-time data analysis, it will have much more benefits than those who can only make near real-time data analysis.

“Veracity” means how unreliable is this data. Many factors can have a bad influence on the veracity of the data. For instance, the data might be outdated or the data is not fully completed so it could not reflect the truth.

3. CAP Theorem

To describe CAP Theorem, it is necessary to understand the three capital characters ‘C’, ‘A’, and ‘P’. ‘C’ is consistency and it means all the nodes will read the same value at the same time. One thing to mention is that if a node does read operation and returns value A, meanwhile another node does write operation and changes value A to B so the first node read the wrong value. This is against consistency. To solve this problem, while a node is doing write operation, other nodes cannot do read operation until the write operation is completed. Martin (2015) gave us a specific explanation:

In the case of linearizability, the situation that may not occur is a stale read: stated informally, once a write operation has completed or some read operation has returned a new value, all

following read operations must return the new value, until it is overwritten by another write operation. Gilbert and Lynch observe that if the write and read operations occur on different nodes, and those nodes cannot communicate during the time when those operations are being executed, then the safety property cannot be satisfied, because the read operation cannot know about the value written.[5]

'A' is availability and it means every non-failing node in the system must respond to the request to show that it is working. In the solution above, it is clear that while a node is doing write operation to node A, non-failing node A cannot respond to the read request from another node B. This is against availability. But this does not mean that consistency and availability cannot both exist all the time. The reason is in the next part about 'P'. Here, Martin (2015) quote the opinion of Gilbert and Lynch to show his conclusion:

On the other hand, Gilbert and Lynch write: "For a distributed system to be continuously available, every request received by a non-failing node in the system to be in different partitions in the distributed system. Partition membership must result in a response".[5]

'P' is partition tolerance and it means if a node cannot communicate with another node within a prescriptive time, these two nodes are in different partitions. Martin (2015) points out:

A network partition has long been defined as a communication failure in which the network is split into disjoint sub-networks, with no communication possible across sub-networks.[5]

That makes consistency and availability cannot be both obeyed. To have both consistency and availability, two nodes must be in the same partition so that they can communicate with each other within a prescriptive time. This prescriptive time is the time limit in consistency. Then if node A does read operation again and returns value v1, node B does write operation and changes value v1 to v2 at the same time. In this case, node B would send a message to node A and tells node A that the value was changed to v2. Now you have both consistency and availability but partition tolerance is lost because it is not possible to ensure that every node is in the same partition. Nodes in different partitions must be tolerated so that is the meaning of partition tolerance.

After understanding the meaning of those three capital characters, it is very easy to explain the CAP Theorem as the discussion above. CAP Theorem describes necessary compromises that

distributed databases cannot have Consistency, Availability, and Partition Tolerance all at one time. Those three characteristics are against each other in some cases so distributed databases can have two of them but not three. This view is supported by Francesc et al. (2008) who write that:

Scalable distributed services try to maintain their service continuity in all situations. When they are geo-replicated, a trade-off exists among three properties: replica consistency (C), service availability (A) and network partition tolerance (P). Only two of those three properties can be simultaneously guaranteed. [6]

A reality example can be used to demonstrate this theorem. Mobile phone applications are designed to serve more users. But in fact, when users are using those applications, there is only one partition for each user: user and application. So mobile phone applications can have both consistency and availability at that time.

Now if the developer wants to update this application, the system will have partition tolerance because this update is for every user and they cannot communicate with each other. That is when the availability has to be sacrificed. Users cannot access the application until the update is done. During this process, you have consistency and partition tolerance.

Of course, you can choose not to update. Then your application is different from other users. In this case, you have availability and partition tolerance but not consistency.

4. Relational Databases and NoSQL Databases

ACID and BASE are the characteristics of relational databases and NoSQL databases. It is much easier to compare and contrast these two databases after introducing their characteristics.

There are four characteristics suited for the transactions of relational databases. They are "Atomicity", "Consistency", "Isolation", and "Durability". Their acronym is ACID.

"Atomicity" means all parts of the transactions are all completed at once or all failed at once. As a result, this transaction is considered as an atom that cannot be divided. If any middle part of the

transaction is failed, the transaction rolls back, which means all previous parts are invalid. The whole database goes back to the state where this transaction begins.

“Consistency” means transactions will not cause the data state error in databases. For example, if a transaction moves a number from A to B, it will not cause the error that this number is removed from A but B is still the same.

“Isolation” means transactions will not influence each other. For instance, when transaction A uses value V1 and it is not completed, transaction B changes value V1 to V2. In this case, the value that transaction A used will not be affected.

“Durability” means once a transaction is committed, the change it made will be permanently saved in the database. Even if a system failure or a machine crash happens, this change will not be lost.

Previous research has established the origin of ACID:

In 1983, Andreas Reuter and Theo Harder published the paper “Principles of Transaction-Oriented Database Recovery” and officially coined the term ACID, which some two decades later has come to mean:

Atomicity: Either the task (or all tasks) within a transaction are performed or none of them are. This is the all-or-none principle. If one element of a transaction fails the entire transaction fails.

Consistency: The transaction must meet all protocols or rules defined by the system at all times. The transaction does not violate those protocols and the database must remain in a consistent state at the beginning and end of a transaction; there are never any half-completed transactions.

Isolation: No transaction has access to any other transaction that is in an intermediate or unfinished state. Thus, each transaction is independent unto itself. This is required for both performance and consistency of transactions within a database.

Durability: Once the transaction is complete, it will persist as complete and cannot be undone; it will survive system failure, power loss and other types of system breakdowns.[7]

BASE is the acronym for the three characteristics of NoSQL databases. They are “Basic Available”, “Soft state”, and “Eventual consistency”.

“Basically Available” means the database allows to sacrifice some availability if the system has an unpredictable fault. For example, normally it should take 0.5 second to respond but now due to some error it will take 1 to 2 seconds or the response is “failure”.

“Soft state” means the database is not strict on consistency. Even if no data is changed, the data state can be changed due to eventual consistency.

“Eventual consistency” means the database must become consistent after some time. The data is allowed to not follow consistency but eventually, it will reach a state following consistency.

This view is demonstrated by previous studies:

In an ACID system, the data fizzes and bubbles and is perpetually active; in a BASE system, the bubbles are still there much like bath water, popping, gurgling, and spinning, but not with the same vigor required from ACID. Here is why:

Basically Available: This constraint states that the system does guarantee the availability of the data as regards CAP Theorem; there will be a response to any request. But, that response could still be ‘failure’ to obtain the requested data or the data may be in an inconsistent or changing state, much like waiting for a check to clear in your bank account.

Soft state: The state of the system could change over time, so even during times without input there may be changes going on due to ‘eventual consistency,’ thus the state of the system is always ‘soft.’

Eventual consistency: The system will eventually become consistent once it stops receiving input. The data will propagate to everywhere it should sooner or later, but the system will continue to receive input and is not checking the consistency of every transaction before it moves onto the next one.[7]

As is mentioned above, the benefits of relational databases are quite clear. First, the relational databases are highly consistent. Unlike the eventual consistency of NoSQL databases, relational databases keep consistency in every moment. As a result, users will get consistent response whenever they query it. Second, relational databases have great reliability. Compared to NoSQL databases, all the data in the relational databases are always correct because they are undergoing constant checks to support the consistency.

On the other hand, NoSQL databases have their own merits. First, NoSQL databases are more available. Any request will get its response though the response might be slower or even a “failure” response. Second, most NoSQL databases are free. Third, NoSQL databases have a more flexible structure. Fourth, NoSQL databases have better performance. All users can get a response while the data are changing by other users. For instance, when you want to buy something online, you might find it still available but the response is that this product is sold out. This is because other people have bought it after you see it, which also means other people checking the product would not stop you from buying it.  5

We can summary the drawbacks of those two databases from the benefits of each other. Relational databases are expensive and inflexible. Moreover, relational databases have low availability and bad performance because it takes too many resources to constantly check the data. NoSQL databases are not as consistent as relational databases for most of the time and, as the above instance has shown, the data might not be reliable.

To demonstrate the strength of relational databases, ICBC (Industrial And Commercial Bank of China) mobile client is a good choice. It is using relational databases like Oracle. The most significant thing for a bank is security and any mistake is unacceptable. Your data must be reliable otherwise the customers would not trust you and go for another bank.

Amazon is the perfect example of NoSQL databases. Its customers are more than 300 million so it is impossible to constantly check the data consistency. Besides, availability is more important for Amazon than consistency. For customers, they must receive a quick response or they will get annoyed. They do not care much about the accurate amount of stock as long as they can know whether they can purchase it or not in the end. This is when eventual consistency and better performance play an important role.

5. NoSQL Database for Zang Technologies

NoSQL database can be explained as Not Only SQL database. Nowadays, there are so many NoSQL databases, which can be divided into four types. They are Key Value Pair Database, Document Database, Column Family Store Database, and Graph Database. As one of the Document Database, MongoDB is chosen for Zang Technologies in this scenario.

One of the best advantages of MongoDB is scalability. The data can be well distributed by MongoDB across multiple machines. As a result, it is much cheaper to store a large amount of data because you can just add more machines instead of purchasing a super large machine. The literature from Kristina (2013) support this view:

MongoDB was designed to scale out. Its document-oriented data model makes it easier for it to split up data across multiple servers. MongoDB automatically takes care of balancing data and load across a cluster, redistributing documents automatically and routing user requests to the correct machines.[8]

Another significant benefit is that MongoDB has high performance. MongoDB will automatically choose the correct indexes, which makes queries become faster. This is also a good news for massive data because it takes less time to deal with a large dataset. Kristina (2013) explained why MongoDB has incredible performance:

Incredible performance is a major goal for MongoDB and has shaped much of its design. MongoDB adds dynamic padding to documents and preallocates data files to trade extra space usage for consistent performance. It uses as much of RAM as it can as its cache and attempts to automatically choose the correct indexes for queries. In short, almost every aspect of MongoDB was designed to maintain high performance.[8]

Scalability and high performance are really suitable for a company like Zang Technologies that contains a large dataset and experiences high write and read loads. Scalability means the company can save some money and do not have to worry that the large dataset will cause a shortage of storage. Besides, the MongoDB can balance data and load. High performance means faster queries into a large dataset and higher speed of writing or reading.

One thing to mention is that MongoDB is a Document Database so its record is a document, which is a data structure composed of field and value pairs. Documents are BSON, which is a binary format similar to JSON objects. This is perfect for those “event” records received from a number of partner companies in the form of JSON. In this scenario, the “event” records can be well stored as documents and it is much easier to operate them because you do not have to change the format.



6. Conclusion

In part 2, I explained what is Big Data and its characteristics i.e four V's. Big Data is a technology that stores and manages a great amount of data. Four V's are "Volume", "Variety", "Velocity", and "Veracity", which respectively mean how much data, how many kinds of data, how fast, and how unreliable.

In part 3, I described CAP Theorem and gave an example to demonstrate it. CAP Theorem is that distributed databases cannot have Consistency, Availability, and Partition Tolerance all at one time.

In part 4, I discussed the characteristics of relational databases and NoSQL databases. Their acronyms are ACID and BASE. ACID represents "Atomicity", "Consistency", "Isolation", and "Durability". BASE represents "Basic Available", "Soft state", and "Eventual consistency". Then I compared and contrasted relational databases to NoSQL databases. The benefits of relational databases are high consistency and reliability. As for NoSQL databases, they have high performance, low cost, flexibility, and availability. I provided two applications, ICBC mobile client and Amazon, to highlight their benefits.

In part 5, I chose a NoSQL Database for Zang Technologies. It is MongoDB, a Document Database, and I explained why it is useful for Zang Technologies. The reasons are scalability, high performance, and similar format.

7. Reference

- [1]. Judith Hurwitz, Alan Nugent, Dr. Fern Halper And Marcia Kaufman, "Big Data For Dummies", March 2013.
- [2]. Joshi, Prem Lal, & Govindan Marthandan. "The Hype of Big Data Analytics and Auditors." EMAJ: Emerging Markets Journal [Online], 8.2 (2018): 1-4. Web. 28 Feb. 2020
- [3]. Delaney, C. W. and Westra, B. (2017) "Big Data: Data Science in Nursing", Western Journal of Nursing Research, 39(1), pp. 3-4. doi: 10.1177/0193945916671687.
- [4]. Valentin Zelenyuk, "Aggregation of inputs and outputs prior to Data Envelopment Analysis under big data", European Journal of Operational Research, Volume 282, Issue 1, 2020, Pages 172-187,

[5]. Martin Kleppmann, "A Critique of the CAP Theorem", 18 Sep 2015

[6]. Francesc D Muñoz-Escóí, Rubén de Juan-Marín, José-Ramón García-Escrivá, J R González de Mendivil, José M Bernabéu-Aubán, "CAP Theorem: Revision of Its Related Consistency Models", The Computer Journal, Volume 62, Issue 6, June 2019, Pages 943–960[6]. "A Dictionary of Computing (6 ed.)". John Daintith and Edmund Wright. 2008

[7]. N. Banothu, S. Bhukya and K. V. Sharma, "Big-data: Acid versus base for database transactions", 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, 2016, pp. 3704-3709.

[8]. Kristina Chodorow, 2013. "MongoDB: The Definitive Guide". Second edition. edn. O'Reilly Media.

BD Assignment by Wanzi Su

GRADEMARK REPORT

FINAL GRADE

77/100

GENERAL COMMENTS

Instructor

a (20): 17

b (20): 12

c (20): 15

di (20): 18

dii (20): 15

Disucssion:

CAP theory discussion was quite reasonable, but the mobile phone example as a demo of the theorem is not a good scenario.

Some nice points raised in SQL vs NoSQL discussions. I wanted a more technical example rather than naming organisations that use NoSQL.

Selection of MongoDB and some good reasoning for choice was provided. You elude to semi-structured data but you could have made this a little more clearer.

Some nice code here, it is fairly straightforward but well presented.



Comment 1

references



Comment 2

Hmm, never seen CAP applied at the application level since it implies a network of machines



Comment 3

?



Comment 4

response



Comment 5

reference



Comment 6

semi-structured data?
