ISSN 2085-4579

Data Modeling for Big Data

M Misbachul Huda[1], Dian Rahma Latifa Hayun[1], Zhin Martun[1]
[1] Informatics Engineering, ITS, Surabaya, Indonesia
misbachul.h@gmail.com,

Accepted on 26 Januari 2015
Approved on 25 Februari 2015

***Abstract*—Today the rapid growth of the internet and the massive usage of the data have led to the increasing CPU requirement, velocity for recalling data, a schema for more complex data structure management, the reliability and the integrity of the available data. This kind of data is called as Large-scale Data or Big Data. Big Data demands high volume, high velocity, high veracity and high variety. Big Data has to deal with two key issues, the growing size of the datasets and the increasing of data complexity. To overcome these issues, today researches are devoted to kind of database management system that can be optimally used for big data management. There are two kinds of database management system, relational database management system and non-relational system that can be optimally used for big data management. There are two kinds of database management, Relational Database Management and Non-relational Database Management. This paper will give reviews about these two database management system, including description, vantage, structure and the application of each DBMS.**

***Index Terms*—Big Data, DBMS, Large-scale Data, Non-relational Database, Relational Database**

I. INTRODUCTION

The rapid growth of the internet and WWW has led to vast amount of information available online. The widespread use of information technology also has led to a dramatic increase in the data availability. More documents or data are produced and stored in cloud or some kind of database management system. These stored and produced have more complex data structure, need more storage, but also need to be executed and recalled fast. This kind of data, nowadays, is called as Large-scaled data or Big Data [1, 31].

Big data is high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making [2]. This definition challenges is twofold. The first is about cost-effective innovative forms of information processing. And the second is enhanced insight and decision making [3]. Big data is fundamentally about applying innovative and cost-effective techniques for solving existing and future business problems whose resource requirements (for data management space, computation resources, in memory representation needs) exceed the capabilities of traditional computing environments as currently configured within the enterprise. The problem happened in early 2000s when data volumes started skyrocketing, storage and CPU technologies were overwhelmed by the numerous terabytes of big data to the point that Information Technology faced a data scalability crisis. The Enterprise, then, went from being unable to afford or manage big data to lavishing budgets on its collection and analysis.

Because of this background, nowadays the scalable and distributed data management has been the vision of the database research community for more than three decades [4]. The research is devoted to kind of database management system that can be optimally used for big data management. There are two kinds of database management system, relational database management system and un-relational database management system. The example of relational database management system is MySQL, and the example of un-relational database management system is key-value, document, or graph database.

This paper will give a more general overview on the overall definition, description, vantage, data structure, and application of relational and non-relational database management system.

The remainder of this paper is organized as follows: The review of the definition and the characteristic of big data. Then, it will be the approach of big data analysis. The next is the modeling of big data, the application and the discussion of this topic. And as the final part is the conclusion of the overview.

II. BIG DATA CHARACTERISTIC

A. The Definition of Big Data

Most definitions of big data focus on the size of data in storage. Size matters, but there are other important attributes of big data, namely data variety and data velocity. The three Vs of big data (volume, variety, and velocity) constitute a comprehensive definition, and they bust the myth that big data is only about data volume. In addition, each of the three Vs has its own ramifications for analytics [5]. The simulation is shown at Figure 1.

It's obvious that data volume is the primary attribute of big data. With that in mind, most people define big data in terabytes—sometimes petabytes. Big data also can be described by its velocity or speed. You may prefer to think of it as the frequency of data generation or the frequency of data delivery.

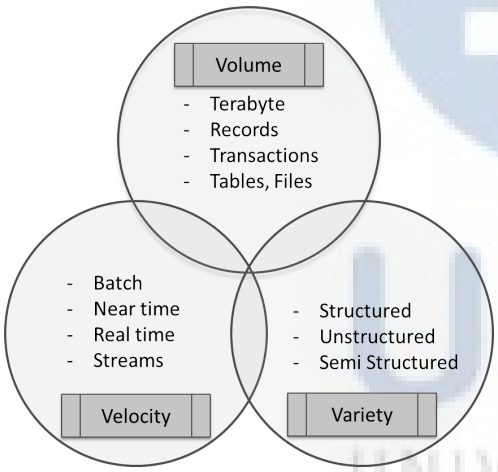

Fig 1. The 3 Vs of Big Data

B. Big Data Characteristic

Talking about Big Data is not only about the big size, but also about the stream and the type. So, it is important to define the characteristic of Big Data. The defined characteristic will be used to measure the quality of each database management system to tackle the Big Data challenge.

The characteristics are defined below.

1. Volume

According to the 2011 IDC Digital Universe Study, "In 2011, the amount of information created and replicated will surpass 1.8 zeta bytes, growing by a factor 9 in just five years [6]." The scale of this growth surpasses the reasonable capacity of traditional relational database management system, or even typical hardware configuration supporting file-based data access. The rapid acceleration of data growth also causes the increasing data volumes pushed into the network. These makes Big Data can be described by its volume or size of data [3].

2. Velocity

Big data also can be described by its velocity or speed. There are two aspects to velocity, one representing the throughput of data and the other representing latency. Throughput represents the data moving in the pipes. Latency is a time interval between the stimulation or request or data recalled and the responds [7].

3. Complexity/Variety

Nowadays, Data Warehouse technology is rapidly introduced. The purpose is to create meta-models to represents all the data in one standard format. The data was compiled from a variety of sources and transformed using ETL (Extract, Transform, Load) or ELT (Extract the data and Load it in the warehouse, then Transform it inside the warehouse). The basic premise was narrow variety and structured content. Big Data has significantly expanded our horizons, enabled by new data integration and analytics technologies. A number of call center analytics solutions are seeking analysis of call center conversations and their correlation with emails, trouble tickets, and social media blogs. The source data includes unstructured text, sound, and video in addition to structured data. A number of applications are gathering data from emails, documents, or blogs.

4. Veracity

Most Big Data comes from sources outside our control and therefore suffers from significant correctness or accuracy problems. Veracity represents both, the credibility of the data source as well as the suitability of the data for the target audience [7].

For example, if a company wants to collect product information from third party and offer it to their contact center employees to support customer queries, the data would have to be screened for source accuracy and credibility. Otherwise, the contact centers could end up recommending competitive offers that might marginalize offerings and reduce revenue opportunities. Likewise, the suitability for the user or audience.

5. Reliability

The reliability in big data is about the accuracy and completeness of computer processed data, given the uses they are intended for. Those, in Big Data challenge, when there are a lot of data

that must be executed in some ways, the expected output is the closes intention.

6. Extensibility

The extensibility is a system design principle where the implementation takes future growth into consideration. Because of the rapid growth of the data, Big Data will lead to a new challenge to overcome. Therefore, to accomplice the current and the future goal of Big Data, the system must consider what is going to be happened in the future.

7. Interoperability

The available data in the cloud or in the Big Data environment is going to be used together, interchangeable, and interpreted. So, for a system to be interoperable, it must be able to exchange data and subsequently present that data such that it can be understood by a user [10]. In Big Data area, it is essential to take a global approach to interoperability and discoverability of information.

8. Scalability

Big Data can be considered as the tsunami of information which has been steadily growing and growing as result of the increasing of digital world. Nowadays, every single people movement or activity is captured and transformed to the digital data. At the end, Big Data is going to keep getting bigger, and more organization are going to be looking to find out more about what to do[9].

9. Integrity

Instrumentation of data requires a complete understanding of the data and the need to maintain consistency of processing (if the data set is broken into multiple pieces), the need to integrate multiple data sets through the processing cycles to maintain the integrity of the data, and the need for complete associated computations within the same processing cycle. The instrumentation of transactional data has been a challenge considering the discrete nature of the data, and the magnitude of the problem amplifies with the increase in the size of the data. This problem has been handled in multiple ways within the RDBMS-based ecosystem for online transaction processing (OLTP) and data warehousing, but the solutions cannot be extended to the Big Data situation. So, one of the points is how to deal with processing Big Data.

10. Flexibility

The data growth affects the flourish of data type spread in the universe. This makes another challenge to effectively and efficiently recalling the data. For some cases, SQL has insufficient expressive prowess. To perform more difficult or complex data structure and schema, the user must using very complex query. So, it is needed to leverage new programming language functionality to implement an object-relational mapping pattern. These programming environments allow developers to benefit from the robustness of DBMS technologies without the burden of writing complex SQL [8].

11. Fault tolerance

Managing large-scale data needs to concern about the performance. One of the performance points is handling the fault that occurs during the execution of computation. Such as the system has to deal with disk failures. Therefore, it is needed a fault handling scheme. If a unit of work fails, then the system must automatically restart the task on an alternate node, in order to do not waste the time by restarting the entire query [8].

III. APPROACH OF BIG DATA ANALYSIS

There are two kinds of approach for big data analysis, Map Reduce and parallel database management system.

A. Map Reduce

The Map Reduce programming model is designed to process large volumes of data in parallel by dividing the Job into a set of independent Tasks. The Job referred to here as a full Map Reduce program, which is the execution of a Mapper or Reducer across a set of data. A Task is an execution of a Mapper or Reducer on a slice of data. So the Map Reduce job usually splits the input data set into independent chunks, which are processed by the map tasks in a completely parallel manner [11]. The simulation of task partitioning is shown at Figure 2

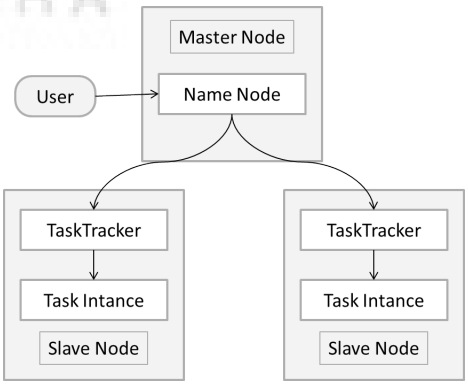

Fig 2. Task Partitioning in Map Reduce [11]

Map Reduce is being originally designed for a largely different application (unstructured text data processing). Map Reduce (or one of its publicly available incarnations such as open source Hadoop) can nonetheless be used to process structured data, and can do so at tremendous scale. For example, Hadoop is being used to manage Facebook's 2.5 petabyte data warehouse. Unfortunately, as pointed out by DeWitt and Stonebreaker [12], Map Reduce lacks many of the features that have proven invaluable for structured data analysis workloads and its immediate gratification paradigm precludes some of the long term benefits of first modeling and loading data before processing. These shortcomings can cause an order of magnitude slower performance than parallel databases.

But despite of that, because Map Reduce is designed to perform unstructured data analysis, unlike a DBMS, Map Reduce systems do not require users to define a schema for their data [13].

B. Parallel Database Management System

The background of having the parallel database management system is the widespread adoption of the relational database [14]. A parallel DBMS can be defined as a DBMS implemented on a multiprocessor computer. This includes many alternatives ranging from the straightforward porting of an existing DBMS, which may require only rewriting the operating system interface routines, to a sophisticated combination of parallel processing and database system functions into a new hardware/software architecture. As always, we have the traditional trade-off between portability (to several platforms) and efficiency. The sophisticated approach is better able to fully exploit the opportunities offered by a multiprocessor at the expense of portability [15]. The parallel DBMS is shown at Figure 3.

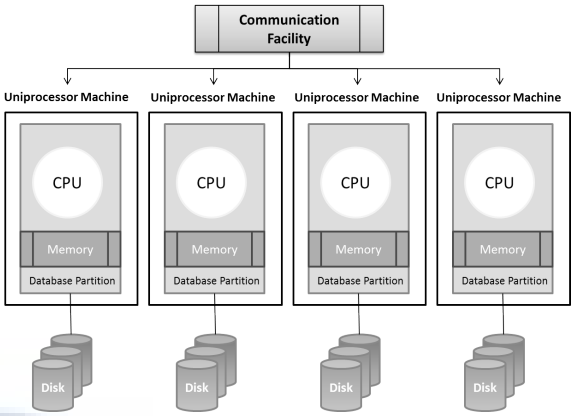

Fig 3. Architecture Parallel DBMS

Ideally, a parallel DBMS (and to a lesser degree a distributed DBMS) should demonstrate two advantages, linear scale up and linear speedup. Linear scale up refers to a sustained performance for a linear increase in both database size and processing and storage power. Linear speedup refers to a linear increase in performance for a constant database size, and a linear increase in processing and storage power [15].

C. The Differences between Parallel Database Management System and Map Reduce

At glance, Parallel DBMS and Map Reduce have many common elements. But actually, there are some basic differences. Parallel DBMSs require data to fit into the relational paradigm of rows and columns. In contrast, the MR model does not require that data files adhere to a schema defined using the relational data model. That is, the MR programmer is free to structure their data in any manner or even to have no structure at all.

All modern DBMSs use hash or B-tree indexes to accelerate access to data. If one is looking for a subset of records, then using a proper index reduces the scope of the search dramatically. Most database systems also support multiple indexes per table. Thus, the query optimizer can decide which index to use for each query or whether to simply perform a brute-force sequential search. Because the Map Reduce model is so simple. Map Reduce frameworks do not provide built-in indexes. To speed up accessing to the data inside the application, any indexes must be implemented. This is not easily accomplished, as the framework's data fetching mechanisms must also be instrumented to use these indexes when pushing data to running Map instances. Once more this is an acceptable strategy if the indexes do not need to be shared between multiple programmers, despite requiring every Map Reduce programmer re-implement the same basic functionality.

IV. Data Modeling for Big Data

Database model is a theory or specification describing how a database is structured and used. Several such models have been suggested such as hierarchical, network, relational and non-relational [20]. Nowadays, relational database models are the dominant persistent storage technology. The relational database model has been dominating since 80s [16], with implementation like Oracle databases [17], MySQL [18], and Microsoft SQL Server [19].

A. Relational Database Model

A relational database is a collection of data items organized in formally-described tables from which data can be accessed or reassembled in many different ways. Relational Database is a set of tables referred to as relation with data category described in columns similar to spreadsheets. Each row contains a unique instance of data for the corresponding data category. While creating a relational database domain of possible values along with constrains are applied to the data. It is the relation between the tables that makes it a 'relation' table. They require few assumptions about how data will be extracted from the database. As a result, the same database can be viewed in many different ways. Mostly all the relational databases use Structured Query Language (SQL) to access and modify the data stored in the database. Originally it was based upon relational calculus and relational algebra and is subdivided into elements such as clauses, predicates, queries and statements [21].

The advantages of Relational Database Model are as follow.

- The data in relational database model are mostly stored in database, not in application.
- The database is structured in a tabular form with highly-related tables.
- It is quite simple to make change in the database schema.

But the Relational Database Model do not support high scalability, until a certain point better hardware can be employed using parallel distributed management system. When the amount of the data become huge, the database has to be partitioned across multiple servers. The other disadvantage is because of the structure of relational database model, gives rise to high complexity in case data cannot be easily encapsulated in a table [21].

B. Relational Database Model

Nowadays, relational database models are the dominant persistent storage technology. It has many shortcomings which can hinder performance levels. As more and more applications are launched in environments that have massive workloads such as cloud and web services, their scalability requirements change very quickly and also grow very large. It is difficult to manage with a relational database sitting on a single in-house server.

To solve all these matters, vendors can optimize for non-relational database models. Non-Relational databases enjoy schema-free architecture and possess the power to manage highly unstructured data. They can be easily deployed to multi-core or multi-server clusters serving modularization, scalability and incremental replication. Non-relational databases being extremely scalable, offer high availability and reliability, even while running on hardware that is typically prone to failure, thereby challenging relational database, where consistency, data integrity, uptime and performance are of prime importance [20,21,33].

Non-relational database model is unlike Relational database model. It does not guarantee the ACID properties [32]. Non-relational databases may primarily be classified on the basis of way of organizing data as follows.

1. Key Value Store

Key value store allows us to store schema-less data. This data consists of a key which is represented by a string and the actual data which is the value in key-value pair. The data can be any primitive of programming language, which may be a string, an integer or an array or it can be an object. Thus it loosens the requirement of formatted data for storage, eliminating the need for fixed data model [21].

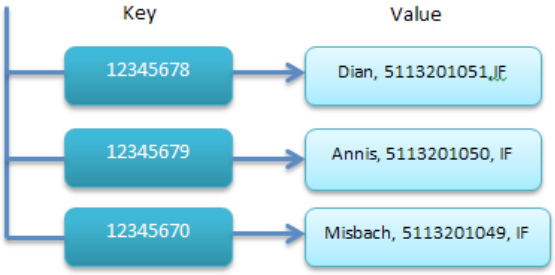

Fig 4. Key Value Store Structure

2. Document Store

Document Store supports more complex data than the key-value stores. The meaning of document here is not like a document in Microsoft Word file or such kind. But it refers to any kind of pointer less object. This kind of non-relational database supports secondary indexes and multiple types of object.

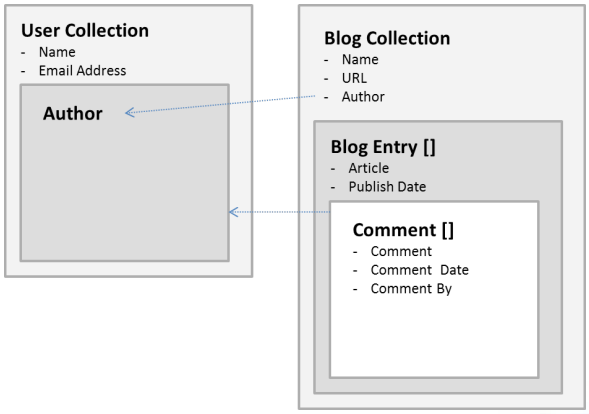

Fig 5. Document Store Structure

The Document Store are schema-less. It provides a mechanism to query collections based on multiple attribute value constraints [22]. Document Store is good for storing and managing kind of text documents, email messages, and XML documents. It also good for storing semi-structured data [23].

3. Graph Store

The other approach for storing data is to model the database directly and entirely as a graph. Big Data has to deal with two key issues, the growing size of the datasets and the increasing of data complexity. Therefore, the alternative database models such as graph databases are more and more used to address this second problem [26].

A graph model is one whose single underlying data structure is a labeled directed graph. The Graph Store consists of single digraph [24]. A database schema in this model is a directed graph, where leaves represent data and internal nodes represent connections between the data. Directed labeled graphs are used as the formalism to specify and represent database schemes, instances, and rules [25].

Fig 6. Graph Store Structure

The reason why using the graph database is the requirement of the application itself. In Graph Store, the interconnectivity or the topology of the data is more important than or at least as important as the data itself. The advantages of Graph Store usage are, it leads to a more natural modeling, because graph structure is visible to the user. Graph can keep all the information about an entity in a single node and show related information by arcs connected to it. The queries can refer directly to this graph structure. Explicit graphs and graph operations allow a user to express a query at a very high level [25]. As far as the implementation is concerned, Graph Store may provide special storage graph structures for the representation of graphs and the most efficient graph algorithms available for realizing specific operations [27]. Although the data may have some structure, the structure is not as rigid, regular or complete as traditional DBMS. The illustration of Graph Store is shown at Figure 6.

4. Column-oriented Database(COD)

A column-oriented database stores data in column order and not in row order as in traditional relational database [28]. Regarding for join algorithm, COD id better than relational DB. With a column store architecture, a DBMS need only read the values of columns required for processing a given query, and can avoid bringing into memory irrelevant attributes for some query it's better than row store[29].

Column-stores have the advantage that dictionary entries may encode multiple values at once [13]. Data stored in columns is more compressible than data stored in rows. Compression algorithms perform better on data with low information entropy [30].

For example, a database containing information about students that have attributes name, registration number, address, and department. Storing that data in column allows all of the name to be stored together, all of the registration number. Further, if the data is sorted by one of the columns, that column will be super-compressible (for example, runs of the same value can be run-length encoded). But of course, the above observation only immediately affects compression ratio. It will lead to get cheap disk space.

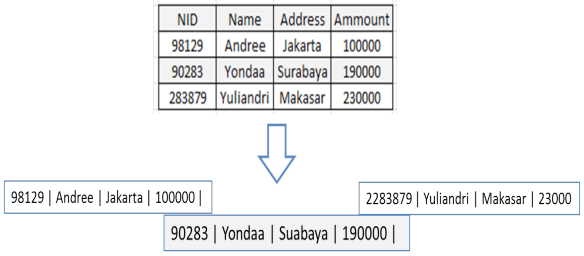

Fig 7. Column-oriented Database Structure

V. COMPARISON OF RELATIONAL AND NON RELATIONAL DATABASE MODEL

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Figure 8 shows the Database Engine survey done at January 2013 to November 2013. From Figure 8, the points that can be taken are as follow relational database has 90.8% presentation value and non-relational database has 9.2% presentation value.

From the data given, shows that relational database is more popular than no relational database. The user is more familiar to the relational database, so that they use it more that non-relational database model. But as new comer the non-relational database having a good presentation.

From the 9.2% of presentations value of non-relational database, it can be divide into four specific part. It is shown at Figure 9. That four parts are:

1. Document Store 39.13%
2. Key-Value 22%
3. Wide Column/column oriented 17.39%
4. Graph and other database 21.74%

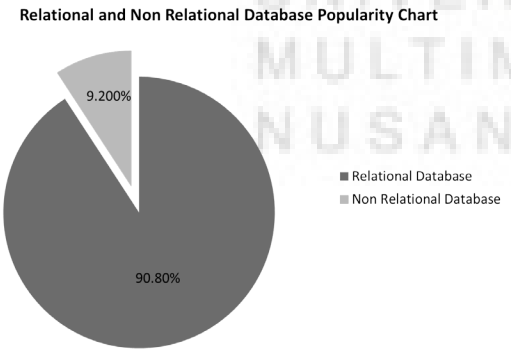

Fig 8. Relational and Non-Relational Database Popularity Ranking [34]

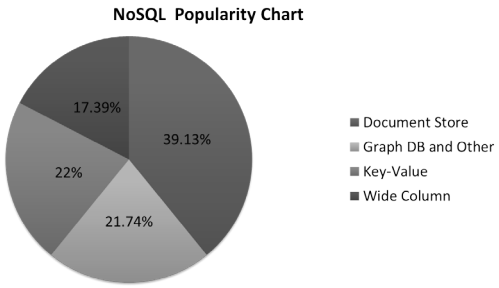

Fig 9. Non-Relational Database Popularity Chart [34]

After comparing the number of database user between users of relational database a non-relational database, it will be discussed about the comparison of relational and non-relational databases. We used 32 parameters for comparison to define how good databases are. The parameters used in the comparison are as follows.

1. Database Model

Database model is model data type represents logical structure from a database. This model is used to store, organize, and manipulate the data. The examples of database model are relational, document, key-value and graph.

2. Integrity model

Integrity model is a computer security rule describing a set of access control rule that is designed to ensure the data integrity. The integrity has 3 purposes. The first is preventing data modification by unauthorized user. The second is preventing illegal data changing by authorized user. And the last is to keep the data consistency [35].

3. Embeddable

Embeddable is a capability for being able to be embedded at some devices. Some databases have the ability to be embedded at specific hardware and software. This can increase the amount of the available resources to expand the database.

4. Query language

Query language is an interface to communicate to the database. This interface can be an application to database, or database to database, and remote database through a media like internet.

5. Isolation

A transaction in the database cannot be discovered by another transaction. The isolation is a feature to tackle this problem. The database has the isolation function in its data model for having high data integrity.

6. Horizontal Scalable

Horizontal scalable is an ability for resource growth and development. The development and the growth of the resource are used to work up the performance of a big data application. Horizontal scalable is devoted to arisen amount of resource usage.

7. Replication

Replication is reduplication process and database object maintenance in databases at distributed database system. The replication is used to create a secondary data or data backup, so that the user can take the data from distributed database system by spending less cost.

8. Replication Mode

Replication is a part of node client definition indicates whether the node client is managed to receive or to send data replication. Besides that, the replication mode is also used to indicate data synchronization at first replication process.

9. Sharding

Sharding is splitting the collection into smaller part that is called as chunks. Chunks, then spread to cluster servers called Shard. Every shard is responsible to the stored subset data.

10. Shared Nothing Architecture

Shared nothing architecture is a distributed computation architecture that every node is independent to another node. Between the nodes, there are no shared memory or disk storage. Every node has memory and disk storage needed.

11. Data types

Data type is a kind of data that can be stored in database. In this attribute, data type is data type in storage. Orin other words is the most primitive data in a database.

12. Graph Support

Graph is set of entity connected to the amount of references. Graph may cause the cyclic occurrence to the used reference. Graph support is the ability of the database handling the cyclic reference.

13. Map and reduce

Map reduce is a parallel programming model to execute a large number of Meta data in computer cluster. Map reduce is based on scale-out principal. It involves a large of computer grouping. The main point using Map Reduce is to move the computation to the data nodes, rather than bringing the data to the computation nodes, and thus fully utilize the advantage of data locality. The code that divides the work, it provides control, and incorporate output in Map Reduce completely hidden from the user application within the framework. In fact, most applications can be implemented in the Map Reduce parallel during synchronized and shared global state is not required

14. TTL for Entries

TTL for entries is the ability for limiting time process in data changing. If the limitation is exceeded, the transaction will be cancelled. TTL for entries can clear up the long transaction and keep the DBMS condition to be not so busy.

15. Secondary Indexes

The secondary index represents a set of attribute in an entity. The secondary index can be used for query from some attributes to work out the performance. Every entity can have 0, 1, 2, or more secondary index according to the query needed.

16. Composite Keys

Composite Key is a key that consists of 2 or more attributes that identify an entity. This key is built for describing more specific entity from some attributes.

17. Geospatial Indexes

Spatial data type is difference from the common data type. Geospatial data type is an encoding from data spatial vector. Indexing ability for geospatial will make the database have more value for GIS application.

18. Query cache

Query cache is used to increase the data fetching for former query. With this ability, every query and every query result will be stored until a certain duration. Every redundant query will be taken from cache data as long as there is no any changing of the database.

19. Data Storage

Data in database will be stored in a storage media. The storage media can be a memory or a file system. The data store is a storage media in a database. Every database has different data store.

20. Conditional Entry Update

This feature has 'where' and 'when' clause. The ability to change the data in a certain condition.

21. Unicode

Unicode is an industry standard designed to allow text and symbol from all writing system in the world to be shown and manipulated consistently in a computer. Unicode guarantees the data consistency to be treated in another

platform or application.

22. Compression

Compression is a database ability used to simplify the data in a smaller size. The advantage is minimizing disk size. A better compression will reduce disk usage.

23. Atomicity

The atomicity is a simplest atomic activity that must be done in the right time or cancelled. The atomic activity cannot be executed partially.

24. Consistency

Consistency is a resource retrieval condition of one consistent part with the other parts. Consistency preserves data integrity.

25. Durability

The durability is the ability to save the data, despite of system failure. This ability is concerned to data stored rather than the active system that handle the data.

26. Transactions

The transaction is an activity or group of activities used by user or application where they access and modify the data in database. The activities will not be stored before commit command. The revocation is also can be done by using rollback command. The database do not have the ability to change the data in database when the user or application is modifying the data.

27. Referential Integrity

The referential integrity is a way used to keep the consistency between the correlated data model. With this ability the correlated data can be ensured for having consistency. At the common, this ability is had by the relational database and graph database.

28. Revision Control

The revision control is technique used to save the system from the backdoor. Revision control system is a system used to store configuration in database. So, every single changing will be noted. All the configuration changes will be stored in a directory to be observed.

29. Locking Model

Locking model is ability to lock the model when changing data. This locking will be opened when the commit and roll back command is given. Locking model is very useful for transaction proses in a long database.

30. Full Text Search

Full Text Search is searching document tube done by the computer by browsing the entire part of a document. The way is by searching the document that has user query word. The query result will be arranged according to the level of words and commonly the frequency will be sorted from high to low. The document browsing will only use basic operation string matching without any other algorithm operation [36, 37].

31. Active

Active database is the ability of the database used to perform computation. Active database clears the perception that a database is only a place to store data. An example of a database can be called as active database is the store procedure database.

32. Value Size Max

Maximum data size handled by database.

At Table 1 shows some examples of the database represent each of database model. These databases will be compared using the explained attribute. The attributes have been classified based on the characteristic of big data. The classification is made to show the quality of a database to the Big Data challenges.

Table 1. Case Study Application Database

No	Model	Database Name	Ranking
1	Relational Database	Oracle	1
2		Postgre	4
3	Document Store	CouchDB	19
4		MongoDB	6
5	Key Value Store	Hbase	17
6		Cassandra	10
7	Wide Column Store	Oracle Coherence	51
8		Redis	13
9	Graph Store	Neo4j	24
10		Titan	74

As shown at Table 2 is the comparison table between the relational database and the non-relational database by using the explained attribute. From the table, it can be concluded that non-relational database have more complex attribute than relational database. For example, as can be seen in the Table 2 some of non-relational database have more attributes than relational database

Such as the Horizontal scalable attribute. All of the non-relational database given in the table have this attribute. But one of relational database,

object relational, has no this attribute. Another prove is that the relational database has no Map reduce attribute that can accelerate the computation process. Unlike the non-relational database that almost all of them having this attribute.

Table 2. Comparison some of Relational and Non-Relational Database with basic few attributes-scalability, variety, velocity, veracity, volume.

Characteristic	Attribute	Relational Database				Non Relational Database					
		Relational	Object Relational	Document-Stored		Wide-Column Store		Key-Value Stored		Graph-Oriented	
	Database Name	Oracle	Postgre	CouchDB	MongoDB	Hbase	Cassandra	Oracle Coherence	Redis	Neo4j	Titan
	Database Model	Relational	Object-Relational	Document-Stored	Document-Stored	Column-oriented	Column-oriented	Key-value	Key-value	Graph-Oriented	Graph-Oriented
Scalability	Query language	SQL, HTTP, SparQL, Xquery, Xpath, API calls, Java API	SQL	JavaScript, REST, Erlang	API calls, JavaScript, REST	API calls, REST, XML Thrift	API calls, CQL Thrift	API calls, CohQL	API calls, Lua	API calls, REST, SparQL, Cypher, Tinkerpop, remlin	Tinkerpop, Gremlin, REST, API calls
	Horisontal Scalable	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Replication Mode	Master-Slave Replication Multi-master replication	Master-Slave Replication	Master-Master Replication	Master-Slave-Replica Replication	Master-Slave Replication	Master-Master Replication	Master-Slave-Replica Replication Master-Master Replication	Master-Slave Replication	Master-Slave Replication	Symmetric Replication
	Sharding	No	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes
	Shared Nothing Architecture	No	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes
Variety	Data types	Binnary	Binnary	JSON	BSON,	Binnary	Binnary	Binnary	Binnary	Binnary	Binnary
	Graph Support	Yes	Yes	No	Yes	No	No	Yes	No	Yes	Yes
Velocity	Map and reduce	No	No	Yes	Yes	Yes	Yes	Yes	No	No	No
	Replication	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	TTL for entries	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No
	Secondary Indexes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes
	Composite keys	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
	Geospatial Indexes	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes
	Query Cache	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes
Veracity	Data Storage	ASM File System	File System	File System	Volatile memory File System	HDFS	File System	Volatile Memory	Volatile memory File System	File System Volatile memory	Berkeley DB Cassandra Hadoop
	Conditional entry updates	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	Isolation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Unicode	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Compression	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
	Atomicity	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Consistency	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Durability (data storage)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Transactions	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes
	Referential integrity	Yes	Yes	No	No	No	No	Yes	No	Yes	Yes
	Revision control	No	No	Yes	No	Yes	Yes	Yes	No	No	No
	Locking model	Lock on Model	MVCC	No	Optimistic Locking Lock on write	MVCC	MVCC	Explicit locking	Lock Free Model	Lock on write	Distributed Locking
	Full Text Search	Yes	Yes	No	Yes	No	No	Yes	No	Yes	Yes
	Integrity model	ACID	ACID	MVCC ACID	BASE	Log Replication	BASE	ACID	ACID	ACID Log Replication	Eventual consistency ACID
Volume	Value size max.	4GB	1GB	4GB	500000GB	2000GB	2GB	64000GB	0.5GB	4GB	64GB

VI. CONCLUSION

Big data be real in the present case. Volume, velocity, variety, veracity, and scalability is a challenge that must be resolved by the database. A variety of modeling approaches, both relational and non-relational been used to try to overcome the problems Big data. From the data shown in section V can be concluded that the relational database has high popularity. But in the case studies of big data, non-relational database has better attributes to satisfy criteria of big data.

In this paper it has been shown that the non-relational databases have attributes more appropriate to resolve the problem big data.

REFERENCES

[1] Cubrid. Database Technology for Large Scale Data. Accesed 29th December 2013. Available: http://www.cubrid.org/blog/dev-platform/database-technology-for-large-scale-data/

[2] Gartner's IT Glossary. Accessed 29th December 2013. Available: http://www.gartner.com/it-glossary/big-data/

[3] D. Loshin, Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL and Graph. USA: Elsevier, 2013.

[4] D. Agrawal, S. Das, and A. El Abadi . Big Data and Cloud Computing: Current State and Future Opportunities. ACM, March 2011.

[5] P. Russom. Big Data Analytics. IBM: The Data Warehousing Institute. 2011.

[6] IDC Digital Universe Study, Study: Extracting Value from Chaos. Accessed 29th December 2013. Available: http://www.emc.com/collateral/demos/microsites/emc-digital-universe-2011/index.htm.

[7] A. Sathi. Bid Data Analytics. USA: MC Press Online. IBM, October 2012.

[8] A. Pavlo, E. Paulson, and A. Rasin. A Comparison of Approaches to Large-Scale Analyisis. ACM, June 2009.

[9] Sand. Simple Scalability Key Big Data. Accessed 29th December 2013. Available: ttp://www.sand.com/simple-scalability-key-big-data/

[10] Himss. What is Interoperability. Accessed 28th December 2013. Available:http://www.himss.org/library/interoperability-standards/what-is

[11] L. Wang, J. Tao, R. Ranjan, H. Marten, A. Streit,

J. Chen, and D. Chen. G-Hadoop: MapReduce Across Distributed Data Centers for Data Intensive Computing. ACM, New York, USA, NY, USA, 2012, pp. 739-750.

[12] M. Stonebreaker, D. Abadi, D.J. Dewitt, S. Madden, E. Paulson, A. Pavlo, And A. Rasin. MapReduce and Parallel DBMS: Friends or Foes?. 2010.

[13] S. Harizopoulos, D. Abadi, and P. Boncz. Column-Oriented Database System. 2009. Available: www.cs.yale.edu/homes/dna/talks/Column_Store_Tutorial_VLDB09.pdf

[14] D. J. Dewitt, J. Gray. Parallel Database Systems: The Future of High Performance Database processing. June, 1992.

[15] [15] M.T. Ozsu, P.. Distributed and Parallel Database Systems. -. Available: www.cs.uoi.gr/~pitoura/courses/ddbs03/paper-to-translate.pdf

[16] [16] A. B. M. Moniruzzaman, S. A. Hossain. NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. International Journal of Database Theory and Application. 2013.

[17] Oracle. Oracle Databases. Accessed: 29th December 2013. Available: Oracle Databases from web: http://www.oracle.com/us/products/database/overview/index.html.

[18] MySQL. MySQL Database. Accessed 29th December 2013. Avalable: web: http://www.mysql.com.

[19] Microsoft. Microsoft SQL Server Databases. Accessed: 29th December 2013. Available: http://www.microsoft.com/en-us/sqlserver/default.aspx.

[20] U. Bhat, S. Jadhav. Moving towards Non-Relational Databases. International Journal of Computer Applications, 2010.

[21] N. Jatana, S. Puri, M. Ahuja, I. Kathuria, and D. Gosain. A Survey and Comparison of Relational and Non-Relational Database. International Journal of Engineering Research & Technology, August 2012.

[22] R. Cattell. Scalable SQL and NoSQL Datastore. 2011.

[23] K. Orend, (2010) "Analysis and Classification of NoSQL Databases and Evaluation of their Ability to Replace an Object-relational Persistence Layer," Master Thesis, Technical University of Munich, Munich.

[24] M. Levene and G. Loizou. A Graph-Based Data Model and its Ramifications. IEEE Transactions on Knowledge and Data Engineering (TKDE), 7(5):809–823, 1995.

[25] R. Angles, C. Gutierrez. Survey of Graph Database Models. Technical Report Number TR/DCC-2005-10, Computer Science Department: Universidad de Chile. 2005.

[26] S.Jouili, V. Vansteenberghe. An Empirical Comparison of Graph Databases.

[27] R. H. G¨uting. GraphDB: Modeling and Querying Graphs in Databases. In Proc. of 20th Int. Conf. on Very Large Data Bases (VLDB), pages 297–308. Morgan Kaufmann, September 1994.

[28] K. C. Kim, C. S. Kim. Parallel Processing of Sensor Network Data using Column-Oriented Databases. AASRI Conference on Parallel and Distributed Computing Systems, pp. 2-8. 2013.

[29] M. Stonebraker, D.J. Abadi, A. Batkin, X. Chen, M. Cherniack, M Ferreira, E. Lau, A.Lin, S. Madden, E. O'Neil, P. O'Neil, A. Rasin, N. Tran, and S. Zdonik. C-Store: A Column-oriented DBMS. Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005.

[30] D.J. Abadi, S. R. Madden, and N. Hachem. Column-Store vs Row-Store. SIGMOD'08, Vancouver, BC, Canada. June, 2008.

[31] L. Wang, M. Kunze, J. Tao, G. von Laszewski, Towards building a cloud for scientific applications, Advances in Engineering Software 42 (9), pp. 714–722. 2011.

[32] T. A. M. C Thantriwatte, C. I. Keppetiyagama. NoSQL Query Processing System for Wireless Ad-hoc and Sensor Networks. In Advances in ICT for Emerging Regions (ICTer), 2011 International Conference on (pp. 78-82). IEEE. September, 2011.

[33] J. Han, E. Haihong, Guan Le, and Jian Du. Survey on NoSQL Database. Pervasive Computing and Applications (ICPCA), 6th International Conference, pp.363-366. October, 2011.

[34] A. Paul, RDBMS dominate the database market, but NoSQL systems are catching up, Accessed 28th December 2013. Available: http://db-engines.com/en/blog_post/23

[35] Z. Belal, A. Essam. The Constraints of Object-Oriented Databases, Int. J. Open Problems Compt. Math., Vol. 1, No. 1, June 2008.

[36] Beall, J.: The Weaknesses of Full-Text Searching. The Journal of Academic Librarianship (September 2008)

[37] Yates, R.B., Neto, B.R.: Modern Information Re-trieval. Addison Weasley Longman Limited (1999)

[38] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.