Chapter 7 The Potential and Capabilities of NoSQL Databases for ERP Systems

Gülay Ekren
Sinop University, Turkey

Alptekin Erkollar
ETCOP, Austria

ABSTRACT

Today, nearly all possible business activities or information systems in enterprises such as sales, marketing, accounting, finance, customer relations, and manufacturing are carried out through traditional relational database management systems. However, technological, social, and competitive pressures in enterprises coming together with the rapid change in technology and then the problems arising from traditional databases force enterprises to adopt new database technologies. This chapter aims to highlight the main differences between traditional relational databases and NoSQL databases and to present an overview of the concepts, features, potential, problems, benefits, and limitations of NoSQL databases for enterprise information management systems, for especially enterprise Resource Planning (ERP) systems, which have a significant role in digital transformation of enterprises.

DOI: 10.4018/978-1-5225-9550-2.ch007

INTRODUCTION

Recently, the development of the Internet and cloud computing are impacted database management systems in especially nonlinear ways. A large amount of data are streaming around the world in different formats (e.g. images, documents, texts, web logs) which cannot be easily modeled, processed or analyzed by traditional methods like relational database management systems (Kaur & Rani, 2013). Many organizations have already changed their single-CPU based relational database management systems to capture and analyze vast amounts of dynamic data. In other words, using traditional relational database management systems has started to be underserved to store and query streaming data. Therefore, it is required to meet the needs of reading and writing such data concurrently as well as accessing and storing them efficiently in higher scalability and availability with lower management and operational costs (Singh, 2015:363).

Nowadays, nearly all possible activities and actions in enterprises such as Customer Relationship Management, Enterprise Resource Planning, Supply Chain Management as well as sales, finance and production systems usually carried out through structural data managed by traditional relational database management systems. However, technological, social and competitive pressures come together with the rapid change in technology as well as the problems arising from the relational database systems built-in enterprise management systems force enterprises to update or transform their data warehousing systems. Moreover, relational databases are well-suited for traditional data analysis such as reporting, classical statistical analysis; however these databases do not fit with big data analytics. Additionally, the need for data storage increasing day by day makes traditional relational databases insufficient, since relational databases have problems in responding to multiple concurrent requests simultaneously. The data stores of the relational databases are usually clustered in more than one field (distributed systems), but in this case, data synchronization problems are encountered. Besides, too much joins and relationships between tables make data access more complex as well as more difficult to implement strict principles in relational databases. In order to eliminate these problems, NoSQL databases which structured as non-relational come into existence by changing some of the parameters of the existing databases including no joins or no complex relationships (Kanwar & Trivedi, 2013: 702-705; Ghasemi, 2013).

On the other hand, NoSQL solutions play a key role for enterprises. Most of organizations have already integrated their enterprise management systems with NoSQL databases to gain competitive advantage in today's marketplace by offering the possibility to work with both structured and unstructured data. Additionally, they can be used successfully to store huge amount of data for enterprise systems (Singh, 2015; Radulović et al., 2016). Since the increasing popularity of big data technology facilitates the integration of NoSQL database systems into business information management systems. Today, enterprises are investing these innovative technologies and then restructuring their decision making mechanisms based on big data. By force of NoSQL database systems as well as big data technologies, enterprise management systems are significantly improved by real time data processing, as the data are accessible in real time by various groups within the enterprise.

NoSQL database systems are considered a solution for the survival of enterprises in the future. They can be seen a new trend for the enterprises that wants to improve their managing capability and its resources. However, there is a need to discover the potentials and capabilities of NoSQL database systems for management of enterprise systems, and for especially ERP systems that play a crucial role on digital transformation. This chapter aims to highlight the main differences between traditional relational database management systems and NoSQL database systems by covering some of the concepts, features, benefits and limitations of these systems.

BACKGROUND

NoSQL was first introduced by Carlo Strozzi in 1998 as a name for his relational database without SQL interface. In 2009, Eric Evans has used it for the discussion of open source, distributed systems with non-relational databases (Kaur & Rani, 2013; Priyanka, 2016).

The term NoSQL is the abbreviation of ‘Not only SQL’ as NoSQL databases use their own SQL language as well as low-level SQL-like query languages (e.g. Cypher Query Language-Neo4j, Cassandra Query Language- Cassandra, MongoDB Query Language- MongoDB). NoSQL databases developed in the late 1990s provide higher scalability and more performance than traditional relational databases. NoSQL typically used for distributed data stores. They have no relational structures but include semi-structured as well as unstructured data especially suitable for big data storage systems. In order to talk about big data systems, it is necessary to collect at least 100 TB data a year, ready to grow at least 60% every year, and to create scale-out architectures for storing data (Pokorný, 2013).

In recent years, “the NoSQL databases” and “cloud computing technology” are two sides of the medallion. Since cloud computing provides a cheaper storage place for NoSQL databases. Besides, the broad distribution of IT resources of cloud computing systems provides advantages for enterprises such as the processing and analyzing of the high volume of stored data (Liu, Yang & Zhang, 2013). According to McCreary and Kelly (2014), NoSQL databases are not only for cloud computing, as they can also run in any corporate data center. Besides, they are not only open source; many commercial products use NoSQL concepts like other NoSQL based systems.

NoSQL based systems have better performance than the relational database management systems under a large volume of data (big data). In case big data is an issue for NoSQL approaches, not only volume (scalability) and velocity (speed) but also variability and agility are seen important. Since big data is a term used for large or complex datasets including diverse data types such as structured, semi-structured and unstructured data, and the relational database management systems are inefficient to handle these them (Priyanka, 2016). NoSQL databases can be defined as "*a set of concepts that allows the rapid and efficient processing of data sets with a focus on performance, reliability, and agility (p.4)*". In this definition, agility refers to how quickly an organization responds to business change (McCreary & Kelly, 2014). Most of the internet giants use NoSQL databases; such as Google- Bigtable, Facebook-Apache HBase, Twitter-MongoDB, Amazon-DynamoDB.

In 2000s, developing community needed to manipulate increasingly growing data generated by web applications. As querying these large amounts of data started to force the designers of relational databases through more clusters of commodity processors, as well as more processors on a single chip. On the other hand, faster processors frequently moved organizations to data management problems since many organizations considered to move away from the relational database systems to the NoSQL systems due to the such reasons (McCreary & Kelly, 2014): (1) a single processor system based on the relational databases have the ability to read and write data rapidly but not handle real-time inserts and online queries at an equal rate, (2) The relational databases have a rigid database schema structures as adding a few columns or rows to a large database mostly required extra time and cost as well as normally impact the availability of the whole system, (3) The applications based on relational databases usually have nested and repeated subgroups of data structures using complex object-relational mapping systems. Adding new applications or updating these existing applications are usually a big handicap when a rapid change needed.

RELATIONAL DATABASES VS NoSQL DATABASES

Relational database systems invented by Edgar Codd in 1970s (Codd, 1970:61). These systems are used as a primary data storage system for years (Enaya, 2016). Before the 2000s, nearly all the organizations used the traditional relational database management system to store real-time data (e.g. event logs). Real-time data usually has a set of activities associated with business processes ordered by timestamp (Gupta, 2015). However; the characteristics of relational database systems are inadequate to deal with large amounts of traditional enterprise data, machine-generated sensor data or social networking data. On the other side, using NoSQL database systems is more convenient as well as more flexible to expand the data storages and to custom system architecture (Liu, Yang & Zhang, 2013).

Additionally, the relational database systems are typically different from NoSQL databases in various aspects with regards to patterns, architectures, and methods (See Table 1).

The key differences between relational and NoSQL databases in performing operations can be summarized as follows:

Investment

The relational databases use expensive proprietary servers and storage systems; however, NoSQL databases use clusters of cheap commodity servers to manage the exploding data and transaction volumes (Sareen & Kumar, 2015). Besides, storing a big volume of data usually required commercial solutions on relational database management systems (e.g. MSSQL Server, Oracle), however NoSQL databases exactly open source (Győrödi, Győrödi & Sotoc, 2015) and their hardware infrastructures tend to have a high cost. In this case cloud solutions can be available to reduce cost, because of this skilled experts required for network configurations as well as for the management of the systems (Tole, 2013)

Table 1. The main differences between relational and NoSQL databases

Properties	Relational Databases	NoSQL Databases
investment	Need expensive proprietary servers	Using clusters of cheap commodity servers
management	Need highly trained database administrators	Required less management
scaling	Vertical scaling (scale-up)	Horizontal scaling (scale-out)
data modeling	Minor changes are more complicated	Allow changes easily
design theme	Focused on questions	Focused on answers
schema characterization	Have a rigid schema	No specific schema
transaction control	Using ACID principles	Using BASE principles
diverse data	Handle structured data	Handle structured, semi-structured and unstructured data.
standardization	Structured Query Language	Different query languages

Management

The relational databases required maintaining only with the assistance of expensive, highly trained database administrators; however NoSQL databases require less management through automatic repair, data distribution, and simpler data models (Sareen & Kumar, 2015). Additionally, relational databases are in use for years and they are stable, but NoSQL databases are still evolving. Therefore, it is difficult to find skilled manpower for the management of NoSQL databases.

Scaling

The relational databases have vertical scaling, thus required more powerful servers for database load increases. They have designed to run on a centralized environment as well as on the distributed environment. However; NoSQL databases easily scale out on commodity clusters and distributed hardware well-suited for horizontal scaling (Kaur & Rani, 2013; Priyanka, 2016; Enaya, 2016). Different from relational databases, NoSQL databases share their workload among distributed nodes without having a rigid schema (Kanwar & Trivedi, 2013; Priyanka, 2016; Gupta, 2015).

Data Modelling

Relational data models are generally used by the relational database systems. This means the relational databases consist of tables and their relationships through key fields. They generally used to store structured data. Doing minor changes in the data model are more complicated when databases have much more interconnected tables. On the other hand, data modeling is still mature for NoSQL databases; however, these databases allow changes to be created without too much fuss (Kaur & Rani, 2013; Vera et al., 2015; Singh, 2015).

Design Theme/ Schema Characterization

Design theme focused on questions in the relational database management systems; but the design theme focused on answers in NoSQL databases (Kaur & Rani, 2013). Kanwar and Trivedi (2013) stated that "*NoSQL works according to the designer instead of the designer working according to the database (p. 701)*". Additionally, NoSQL databases do not have a specific schema characterization. Therefore, there are difficulties in the integration of structured datasets with unstructured or semi-structured datasets (Pokorný, 2013).

Transaction Control

Transaction control is crucial for distributed systems with respect to performance and consistency. The most used transaction control models are as follows (McCreary & Kelly, 2014; Kanwar & Trivedi, 2013): (1) **ACID Model:** ACID model can be used in relational databases to ensure the principles- Atomicity, Consistency, Isolation, and Durability to maintain transaction control. Relational databases utilize foreign keys of structured data for data integrity and data consistency, especially for transaction controls. Therefore, the maintenance of relational database systems can be quite complex when the data has no structure. In this model; atomicity preserves competition of business process. For example, a transaction takes place completely or never occurs. Likewise, isolation means that the transactions can be carried out simultaneously. Thus, it is required a locking mechanism for uncommitted database modifications. Additionally, in the event of any interruption or failure in the system, the ongoing transactions must not be adversely affected (Mohmmed & Osman, 2017). However; this model is well-fit with banking systems to provide data integrity, (2) **BASE Model:** BASE model can be found mostly in NoSQL systems typically to maintain transaction control and to ensure the principles- Basically, Available, Soft state and Eventual consistent. Basically Available uses replication and sharing to increase the availability of data. Soft State means that data can be inconsistent in NoSQL databases. Eventual Consistency means that consistency can only be guaranteed by NoSQL databases in some unidentified cases (Berndt, Lasa & McCart, 2012). The BASE model can be preferred by e-commerce systems.

Nowadays, NoSQL databases have adopted by CAP theorem stands for Consistency, Availability, and Partition Tolerance. CAP theorem is firstly proposed by Eric Brewer in the Principles of Distributed Computing of ACM Symposium in 2000 and then proved by Seth Gilbert and Nancy Lynch. Berndt, Lasa, and McCart (2017) defined consistency, availability and partition tolerance as "*Consistency means if one value is written or updated to one node then another node should be updated automatically. Availability means if there is any kind of failure in the system then the value can still be retrieved from the system. Partition tolerance means if partitioning is done then there will be no impact on data that is data should be able to retrieve and write to multiple places*" (p. 5). Consistency and availability principles are important especially for distributed systems and available in both relational and NoSQL databases. However, partition tolerance is not available in relational databases (Singh, 2015), additionally some of NoSQL databases such as BigTable, MongoDB, MemcacheDB ensure only "availability and partition tolerance" when some of the others such as Cassandra, CouchDB, Dynamo ensure "consistency and partition tolerance" (Choi, Jeon & Yoon, 2014).

Handling Diverse Data

The relational database management systems cannot handle a huge amount of diverse data, as they usually used for only structured data. However, NoSQL databases can handle a huge amount of diverse data, not only for structured but also for semi-structured and unstructured data (McCreary & Kelly, 2014; Priyanka, 2016). Therefore, NoSQL databases are suitable for big data applications and show better performance than relational databases with a growing amount of data coming quickly (Kaur & Rani, 2013).

Standardization

Relational databases use Structured Query Language (SQL) to access data, thus it has high reliability as well as high portability by means of the American National Standards Institute (ANSI) standard. On the other hand, there is no portability or standard for NoSQL because of vendor-dependent APIs (Choi, Jeon & Yoon, 2014; Enaya, 2016). SQL is a standard query language for storing, retrieving and manipulating data in relational databases. However, there is no standard query language for NoSQL databases. Since NoSQL databases are data-model specific, therefore each database comes with its own query language (Kaur & Rani, 2013). NoSQL database does not use SQL queries to retrieve data with tabular form, as relational databases do (Priyanka, 2016).

NoSQL DATA STORES

Currently, many business organizations have a big volume of data to be processed and required a new type of thinking for managing that data by moving away from traditional methods. Since they need to increase the performance of their databases as well as they need to accept orders or requests quickly and budget-friendly. Relational databases were started to be inefficient to handle a huge amount of diverse data generated by real-time web applications as well as by enterprise management systems. Since Google was one of the first business drivers that lead NoSQL data stores by introducing BigTable in 2006 named as column-based data stores, followed by Amazon's Dynamo in 2007 named as key-value based data stores. Then, document-based and graph-based data stores have emerged. NoSQL databases can be categorized based on their data models as follows (See Figure 1):

1. *Column-family data models* can be stored columns of data together, instead of rows. Each data item has a key (row) and a set of attributes. Attributes stored

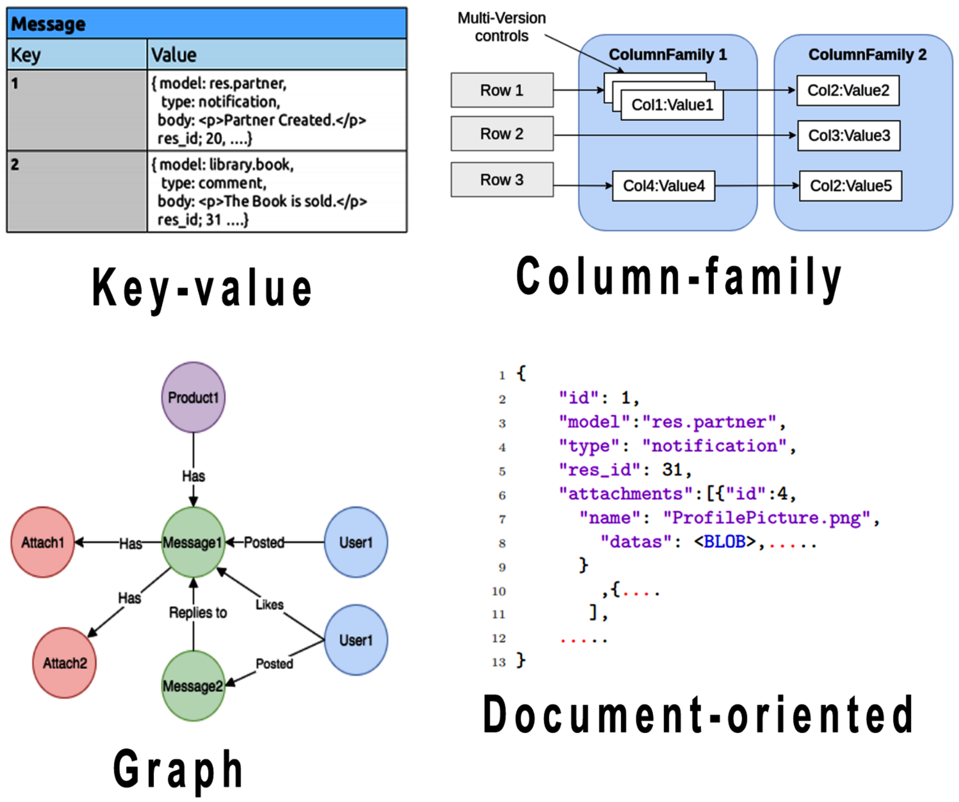
as key-value pairs, the key can be called a column, and a set of related columns forms a column-family. The tables involve one or more column-families (Enaya, 2016). As it is easy to manage distributed and replicated data by means of this model. Thus, this model is usually efficient to solve big data problems and to show web crawler results. Some popular examples of these data stores; Big Table (Google), Cassandra (Facebook), HBase (Apache), DynamoDB (Amazon), PNUTS (Yahoo), Hypertable, Druid, Vertica, Accumulo.

2. *Document-oriented data models* can be stored data or records as documents in formats such as a binary JSON, XML or Yet Another Markup Language (YAML). This model can store messages and their related attachments with different attributes. It is also well-fit for high-variability data, searching documents, integration of hubs, web content management and publishing. Some examples of these data stores; MongoDB, Couchbase, Apache CouchDB, Lotus Notes, OrientDB, Clusterpoint.
3. *Graph data models* can be stored data about networks such as social connections. Therefore, this model is well-fit with social networking applications (e.g. Facebook, Twitter) as well as fraud detection. In this model data items represented as graphs (nodes) and the relations between data, items are more important than the data item itself (Enaya, 2016). Some examples of these data stores; OrientDB, FlockDB, Neo4J, HyperGraphDB, AllegroGraph, Virtuoso, MarkLogic, InfiniteGraph.
4. *Key-value data models* can be stored data as an attribute name or key together with its value. They are well-fit for image stores and key-based file systems. This model also the base for all NoSQL databases, but it can not search the message on different attributes (Enaya, 2016). Some examples of these data stores; DynamoDB (Amazon), Redis, Riak, Voldemort (LinkedIn), BerkeleyDB (Oracle), MemcacheDB, FoundationDB, HyperDex, SimpleDB.
5. *Multi-model data models* support more than one of data models such as document, graph, relational, and key-value. Some examples of these data stores; ArangoDB, CosmosDB, Couchbase, EnterpriseDB, OrientDB, SAP HANA, Oracle Database.

NoSQL FOR INFORMATION SYSTEMS

NoSQL databases are suitable not only for scientific data (astronomy, physics, chemistry, medicine, statistics, etc.) but also for sensor data, social networking data, digital data streams (media, video, image, sound, text, etc.), as well as for traditional corporate data in information systems such as customer information from CRM

Figure 1. Different data models for NoSQL databases



Key	Value
1	{model: res.partner, type: notification, body: <p>Partner Created.</p> res_id; 20, ...}
2	{model: library.book, type: comment, body: <p>The Book is sold.</p> res_id; 31 ....}

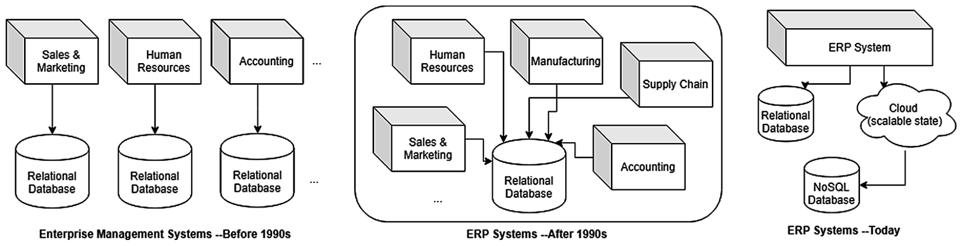
```
{
    "id": 1,
    "model":"res.partner",
    "type": "notification",
    "res_id": 31,
    "attachments":[{"id":4,
      "name": "ProfilePicture.png",
        "datas": <BLOB>,.....
      }
        ,{....
      ],
    .....
}
```

systems, transactional data from ERP systems (Liu, Yang & Zhang, 2013), financial data in accounting systems, sales and marketing data in e-commerce systems.

Enterprise information systems (e.g. ERP systems) usually initiated with data obtained from commercial transactions (See Figure 2). This data describes the completed operations but does not define data created before and after transactions. Moreover, this data are standard and structured as well as semantically homogeneous. In ERP systems, other structured data come from CRM systems through customer interactions before and after transactions. These interactions are not a commercial transaction but increase the amount of data called pre-transaction and post-transaction data. Besides, unstructured elements which are semantically homogeneous such as Web records, sensor data, and social networking data increase the amount of data (Schmidt & Möhring, 2013).

Nowadays, as the data flow on the internet increases, the systems face scaling problems. In order to solve this problem, new problems such as increasing costs and limit deficiency, updating of clusters have arisen, and then different solutions were

Figure 2. Growing data in enterprise information systems
(Schmidt & Möhring, 2013)



produced such as purchasing large servers or creating distributed systems (Berndt, Lasa & McCart, 2012).

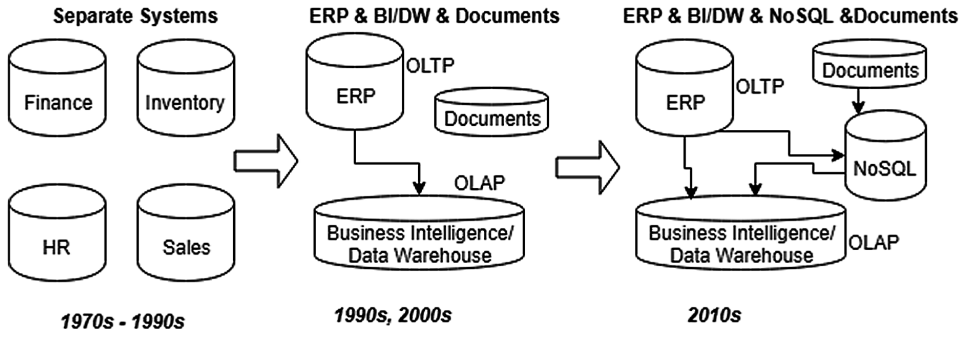
On the other hand, the findings show that the information system based on NoSQL database system reveals higher performance than the one which has a traditional relational database, for instance, the throughput of the system using NoSQL was found 20% higher than the one using relational database (Choi, Jeon & Yoon, 2014). In a similar study, Gupta (2015) conducted a study to investigate the performance of relational and NoSQL databases in a Process-Aware Information System (PAIS) and to compare which database system is more efficiently store massive event logs (real-time data) and analyze it in seconds to discover a process model. PAIS is an information system which manages and supports business processes and generated data like ERP and CRM systems. In this study, HBase as a NoSQL column-oriented data store is used in a real-time messaging system and compared its performance against MySQL as a relational database system. The results show that HBase performs better (on an average is 29 times faster) than MySQL in terms of storing data, performing query as well as loading large datasets.

NoSQL FOR ERP SYSTEMS

ERP systems have a significant role for the digital transformation of enterprises, not only in decision-making processes but also in reducing downtime and costs, quick, online and instant access to the streaming data, as well as efficient use of resources, real-time profit-loss and cost analysis. These systems integrate the transactional business data and business processes into a single system with a single database (See Figure 3).

ERP systems have a three-tier architecture including database layer, application layer, and user layer. Nowadays, businesses have started to move the database and application layer of their ERP systems to the cloud (Elragal, 2014).

Figure 3. Enterprise management systems over the years



Evolution of ERP Systems

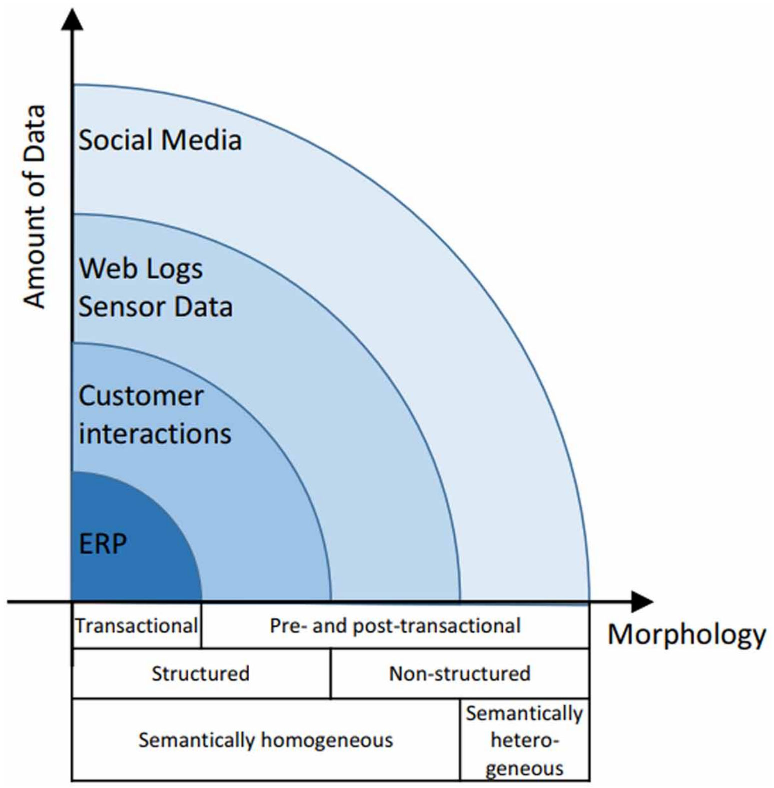
The foundation of ERP systems is based on pre-accounting and inventory systems in the 1960s. These systems were developed in the 1970s in the form of material requirement planning (MRP) systems. In the 1980s, manufacturing resource planning systems (MRPII) came into existence. MRPII has emerged as an extended and more comprehensive MRP version covering all business processes in manufacturing companies. In the 1990s, ERP systems were introduced as an extension of previous MRP versions. These systems aim to integrate all business processes of the entire organization into a single platform. In other words, it focuses not only on manufacturing processes but also on all business processes. In addition, these systems provide a centralized data storage and integration center across all departments within the organization (Elragal & Haddara, 2012).

Over the years, there have been three phases of ERP systems related to database systems (See Figure 4). Between the 1970s and 1990s, the enterprises had control on business units such as human resources, finance, and accounting, inventory, sales and marketing separately together with their own database systems for each unit. Then, from 1990s to 2000s, Enterprise Resource Planning (ERP) systems come into exist, as one of the information systems that consist of "*a set of business applications or modules, which links various business units of an organization such as financial, accounting, manufacturing, and human resources into a tightly integrated single system with a common platform to ensure the flow of information across the entire business* (Beheshti, 2006: 184)". ERP systems are integrated enterprise-wide systems which replace existing legacy systems or software applications with a single internal enterprise system. The system also provides to connect overall customers and suppliers to the common systems. Thus, eliminating the time and space limit by linking the entire organization from one point to the other, ERP systems ensure the joint and efficient use of production and distribution resources such as materials, labor, machinery-equipment, information (Beheshti, 2006).

ERP systems are one of the best innovations of IT-supported business applications. In recent years, the centralized relational databases in ERP systems cannot adopt the demand of processing a large amount of unstructured data, thus NoSQL databases started to be using with ERP systems. A study conducted by Enaya (2016) examines whether NoSQL databases improve the performance of Odoo which is an open source ERP system, or not? Odoo has two special modules for mail messages and attachments, and these modules can be used by almost all other modules. The growing amounts of stored data produced by these modules reduce Odoo performance. To overcome this problem, Odoo was modified to communicate with a NoSQL database system, since Hadoop Hbase selected as secondary data storage, and its performance compared with the performance of regular Odoo based on relational database system (PostgreSQL). The findings show that NoSQL system has better performance to handle huge volumes of data (e.g. the response time increase almost 45% between four and five million records – mail messages), increasing the number of requests and generating reports. PostgreSQL has better performance than HBase to handle a smaller amount of data and shows less adaption with increasing data size its performance is reduced greatly with a huge volume of data. HBase starts to perform better than PostgreSQL especially after two millions of records. The results also show that NoSQL systems have scalability and to increase their scalability, more nodes can be added to the cluster. On the other hand, more powerful hardware systems are needed to increase the performance and scalability of relational database systems. In a similar study; Bhaswara (2017) compared the NoSQL database (MongoDB) with relational database (MySQL) in terms of performance, flexibility, and scalability through an ERP system (a case study of Retail module). According to the results, MongoDB proved to have superior speed rather than MySQL in Create/Read/Update/Delete (CRUD) applications. It has also a flexible structure for data storage (scalability) as the model of data in the form of Shared-Nothing Architecture (Binary JSON). Findings show that NoSQL databases are better to apply ERP systems.

On the other hand, traditional relational databases are designed to efficiently add, update, search, and retrieving small amounts of transactional data in a large database. A user can log in to an e-commerce site, search for products, and buy products over these databases. However, these business processes are replicated across the world in countless ways over the days, months, years, and traditional database management systems are inadequate when someone wants to learn from the large amounts of accumulated data (Jacobs, 2009). Additionally, these systems oriented toward processing large blocks of replicated, disk-based data at a time, so latency problems come into existence in the system (Madden, 2012).

Figure 4. The evolution of ERP systems
(McCreary & Kelly, 2014)



ERP systems are a kind of Online Transaction Processing (OLTP) system designed to provide high-speed transaction recording. Moreover, ERP systems have the capability to drive business intelligence systems and data warehouse systems through Online Analytical Processing (OLAP) tools to reveal the analytical capabilities of enterprises such as accessing reports at a great speed, creating balanced scoreboards, displaying data from different points on a single interface. OLAP tools have no adverse impact on ERP performance. OLAP tools based on NoSQL database systems can be added to enterprise management systems to eliminate problems arising from relational databases, as most of the tasks are not suited for relational database systems. Besides, relational databases do not support document integration. So, it can be said that NoSQL systems are well suited for ERP systems, business intelligence systems, data warehouse systems (Figure 3), Process-Aware Information Systems, as well as not only for document-oriented database systems but also graph-based, key-value based and column-oriented database systems. However; a NoSQL system does not replace ERP or business intelligence or data warehouse systems, but it can complement these information systems to support the integration of structured and unstructured data (McCreary & Kelly, 2014)

NoSQL FOR BI/WAREHOUSE SYSTEMS

The use of data warehousing in business intelligence applications is generally seen as an alternative solution to traditional relational database problems. A data warehouse can be defined as a copy of process data specifically configured for query and analysis. In other words, the transactional data is bulk-extracted from one or more databases and then converted to analytical queries in a different database according to business rules. However, in the case of big data, it may not always be enough to create a data warehouse (Jacobs, 2009).

Business Intelligence (BI) is a system based on the intelligent use, reuse and management of enterprise data sources in order to add value as well as to increase the operational efficiency and to provide new business opportunities to the enterprises. BI systems are usually working together with big amounts of transactional data as an extension of the methods used in data warehouse systems. However, data warehouse systems generally focus on structured data, so they are not systems that fit the rich variability and main characteristics of big data. For the analytical processing of big data in business intelligence applications, new database architectures, as well as new methods, are required for analyzing data such as NoSQL databases, Hadoop MapReduce technologies. Hadoop is a framework that can be used with NoSQL databases as a highly scalable MapReduce platform (Pokorný, 2013).

Most of the traditional relational databases focus on OLTP applications but not have the capability to deal with efficiently on OLAP applications. However; NoSQL databases do not support OLTP and focus on OLAP applications, as they have been shown better performance on analytical queries using CRUD (Create/Read/Update/Delete) operations, MapReduce jobs or client API's - application programming interfaces (Gupta, 2015). MapReduce can deal with large amounts of structured, semi-structured and unstructured data by using a large number of cheap servers as it does not need high data consistency (Liu, Yang & Zhang, 2013).

BIG DATA AND ERP SYSTEMS

ERP systems and new applications should be updated to deal with new trends in technology such as cloud computing, social networking, crowdsourcing, and service-oriented architectures. However, these technologies discussed more in the IS (Information Systems) literature, but not usually discussed in the context of ERP systems (Elragal & Haddara, 2012). According to Cadersaib et al. (2018), big data is an issue that needs to be discussed and developed in the context of ERP systems.

A typical big data processing is performed as follows; (1) First of all, the data is collected coming from different sources such as traditional ERP transaction data, streaming data from various social media platforms, sensor data from Internet of Things (IOTs) based devices, data from business process, log files, tag data created by mobile phones etc. (2) Then, this data is transformed according to quality standards, as data with an acceptable level of quality is stored, (3) Next, the stored data processed, aggregated and analyzed with various algorithms such as MapReduce, (4) Finally, the analyzed data visualized and made ready for use by decision-making mechanisms (Schmidt & Möhring, 2013; Windmann et al., 2015). Data collected from ERP systems usually manually entered by staff with low efficiency, easy to have mistaken (unreliable), and poor in real-time performance, so the acquisition of big data must be realized by intelligent data collectors, classified and stored to facilitate query, and update in real time (Liu et al., 2018).

On the other hand, the combination of ERP systems and big data technologies brings together a number of problems and needs for enterprises. Elragal (2014) summarized some of them as follows; (1) Structured data in ERP systems contain certain quality restrictions and must comply with internal quality standards and procedures. However, when these data combined with data from external sources, there would be biases and inconsistencies related to the system, (2) Combining structured data with unstructured data or semi-structured data in ERP systems requires the development of new analysis methods, models and frameworks, (3) Big data outsourcing to ERP systems may require restructuring of ERP lifecycle or implementation processes. Most of the ERP vendors have their own implementation methodologies such as SAP ERP's ASAP methodology, Oracle ERP's AIM methodology. With the emergence of social networks and cloud computing, it is likely that there will be a change in the ERP implementation life cycle. As it is known that cloud computing shortens and changes the life-cycle of ERP systems (Elragal & Haddara, 2012). Babu and Sastry (2014) introduced a new implementation framework that uses large data to automate operational decision making and forecasting operations in SAP ERP systems. According to the Babu and Sastry, when designing decision-making systems for ERP systems, they should be designed in agile, analytical and adaptive with existing systems in a way that includes people and processes. ERP systems such as SAP, Oracle have already generated their system based on NoSQL to take advantages of sensor data, social networking data, customer data and data from other resources (Tole, 2013).

Next Generation Relational Databases: NewSQL

One of the biggest disadvantages of NoSQL database systems is that they have no support earlier applications based on relational databases. In other words, they have no adaption to expand existing applications for real-time data. This means implementing scale-out architecture required NoSQL systems, but can not support all ACID properties. However, alternative database systems called NewSQL can be used to develop new applications in traditional OLTP systems. NewSQL is seen as next-generation relational databases for OLTP systems that provide high-level linear scalability, delivers all the properties of ACID transactions, handles complex data as well as supports standard SQL language (Moniruzzaman, 2014). Some of the NewSQL database systems are as follows:

- VoltDB (https://www.voltdb.com/),
- ClustrixDB (https://www.clustrix.com/),
- NuoDB (http://www.nuodb.com/),
- MemSQL (https://www.memsql.com/),
- Altibase (http://altibase.com/),
- c-treeACE (https://www.faircom.com/),
- CockroachDB (https://www.cockroachlabs.com/),
- Apache Trafodian (https://trafodion.apache.org/),
- ActorDB (http://www.actordb.com/).

NewSQL systems are significantly different in some aspects from NoSQL systems. The comparison of characteristics of both systems is shown in Table 2.

FUTURE RESEARCH DIRECTIONS

Database management is the heart of business information systems. Database systems are open to change and development in line with the needs of business information systems. Emerging technologies such as Apache Hadoop, NoSQL, and big data affect the way of using business data. On the other hand, companies wants to take advantages of the emerging technologies to better understand their customers and to design more useful as well as more preferred products and services. For example; business data collected in the last five years help us to predict what can be done in the next five years by force of solutions such as big data analytics. The information management systems need to adopt the NoSQL databases to understand where and when it makes sense.

Table 2. The main characteristics of NoSQL and NewSQL systems

Properties	NoSQL	NewSQL
scaling	Store big data and having more scalability, ability to scale horizontally	Store big data and having more scalability, ability to scale horizontally
data modeling	Not support ACID transactions, or OLAP/OLTP tools	Support ACID transactions, OLAP/OLTP tools
design theme	Platform-specific query language	SQL as the primary query language
schema characterization	Support flexible non-relational data models, not require fixed table schemas	May be support rigid relational data models
transaction control	Provide high availability, capable of running on a large number of nodes without suffering bottlenecks	Provide high availability, support a non-locking concurrency control mechanism
diverse data	Handle structured, semi-structured and unstructured data.	Appropriate for diverse data (from sensors, mobile phones, network access points) but inappropriate for volumes exceeding a few terabytes
standardization	Support various query languages	Familiar SQL and standard tooling, but offers only partial access to the traditional SQL systems

Further research are needed which focuses on NewSQL technology that is one of the few trends affecting business information systems nowadays. More research can be done by comparing the performance of NoSQL and NewSQL systems on unstructured data which are semantically homogeneous such as Web records, sensor data, and social networking data as well as by generating a variety of architectures or applications on OLTP systems. Additionally, further extensive studies can be done on the security concern of NoSQL or NewSQL systems, as it is known the reliability of traditional database management systems is rather high, but there are bias, fears, mistrusts on other database systems.

CONCLUSION

Enterprise information systems typically contain massive amounts of real-time data at very high volumes, variations, and speeds. It is difficult to manage them effectively with traditional data management tools. Business information systems are generally compatible with traditional relational database management systems. They are inherently having difficulties in processing big data. Therefore, high volume of data, diversity of data, as well as performance and scalability issues of data required alternative solutions for data management. For this reason, alternative database technologies such as NoSQL and NewSQL have emerged in recent years. Emerging systems are significantly different from traditional data management. For example, their data stores have generally scaled horizontally and are not required joining operations, fixed table schemas.

The need for alternative database technologies for enterprise information systems is increasing day by day, in which data is stored in a distributed manner but the access/read/write/update of data can be made quickly and smoothly by existing applications. Also, the data should be analyzed and used effectively by decision-making mechanisms.

For this reason, the integration of new technologies by information management systems is seen insufficient. However, the turning point for businesses to be ready for these alternative technologies are painful. The wished outcome of this chapter is to have brought awareness among researchers, scholars, business managers, decision makers, business analysts, ERP project managers to support the use of these technologies, which would be extensively used by enterprise information systems in a few years.

REFERENCES

Beheshti, H. M. (2006). What managers should know about ERP/ERP II. *Management Research News*, *29*(4), 184–193. doi:10.1108/01409170610665040

Berndt, D., Lasa, R., & McCart, J. (2017). SiteWit Corporation: SQL or NoSQL that is the Question. *Journal of Information Technology Education: Discussion Cases.*, *6*(4), 1–22.

Bhaswara, F. A. (2017). *Rancang Bangun Non-Relational Database pada Enterprise Resource Planning Retail yang Berorientasikan Multitenancy dengan Shared Distributed Database* (Undergraduate thesis). Institut Teknologi Sepuluh Nopember.

Cadersaib, B. Z., Sta, H. B., & Rahimbux, B. A. G. (2018, October). Making an Interoperability Approach between ERP and Big Data Context. In *2018 Sixth International Conference on Enterprise Systems (ES)* (pp. 146-153). IEEE. 10.1109/ES.2018.00030

Choi, Y. L., Jeon, W. S., & Yoon, S. H. (2014). Improving Database System Performance by Applying NoSQL. *JIPS*, *10*(3), 355–364.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13*(6), 377–387. doi:10.1145/362384.362685

Elragal, A. (2014). ERP and big data: The inept couple. *Procedia Technology*, *16*, 242–249. doi:10.1016/j.protcy.2014.10.089

Elragal, A., & Haddara, M. (2012). The Future of ERP Systems: Look backward before moving forward. *Procedia Technology*, *5*, 21–30. doi:10.1016/j.protcy.2012.09.003

Enaya, M. F. (2016). *An Experimental Performance Comparison of NoSQL and RDBMS Data Storage Systems in the ERP System Odoo* (Doctoral dissertation). University of Magdeburg.

Ghasemi, P. (2013). *Migration of Legacy Web Application Using NoSQL Databases* (Thesis). Faculty of the Department of Department of Computer Science, East Carolina University.

Gupta, K. (2015). *Pragamana: performance comparison and programming-miner algorithm in relational database query language and NoSQL column-oriented using apache phoenix* (Doctoral dissertation).

Győrödi, C., Győrödi, R., & Sotoc, R. (2015). A Comparative Study of Relational and Non-Relational Database Models in a Web-Based Application. *International Journal of Advanced Computer Science and Applications*, *6*(11), 78–83. doi:10.14569/IJACSA.2015.061111

Jacobs, A. (2009). The pathologies of big data. *Communications of the ACM*, *52*(8), 36–44. doi:10.1145/1536616.1536632

Kanwar, R., & Trivedi, P. (2013). Reincarnating Traditional Relational Database through NoSQL. *International Journal of Information and Computation Technology*, *3*(7).

Kaur, K., & Rani, R. (2013, October). Modeling and querying data in NoSQL databases. In *2013 IEEE International Conference on Big Data* (pp. 1-7). IEEE. 10.1109/BigData.2013.6691765

Liu, Z., Yang, P., & Zhang, L. (2013, September). A sketch of big data technologies. In *2013 seventh international conference on internet computing for engineering and science* (pp. 26-29). IEEE. 10.1109/ICICSE.2013.13

Madden, S. (2012). From databases to big data. *IEEE Internet Computing*, *16*(3), 4–6. doi:10.1109/MIC.2012.50

McCreary, D., & Kelly, A. (2014). *Making sense of NoSQL* (pp. 19–20). Shelter Island: Manning. Retrieved from https://www.manning.com/books/making-sense-of-nosql

Mohmmed, A. G. M., & Osman, S. E. F. (2017). Study on SQL vs. NoSQL vs. NewSQL. *Journal of Multidisciplinary Engineering Science Studies*, *3*(6), 1821–1823.

Moniruzzaman, A. B. M. (2014). *Newsql: Towards next-generation scalable rdbms for online transaction processing (oltp) for big data management.* arXiv preprint arXiv:1411.7343

Pokorný, J. (2013). New database architectures: Steps towards big data processing. In *Proceedings of the IADIS International Conference Intelligent Systems and Agents* (pp. 3-10). Academic Press.

Priyanka, A. (2016). A review of nosql databases, types and comparison with relational database. *International Journal of Engineering Science*, 4963.

Radulović, B., Radosav, D., & Malić, M. (2016). The Application of NoSQL MongoDB in Developing the EPR System for Managing Human Resources. *Int'l Journal of Computing Communications & Instrumentation Engg.*, *3*(1), 181–185.

Schmidt, R., & Möhring, M. (2013, September). Strategic alignment of cloud-based architectures for big data. In *2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops* (pp. 136-143). IEEE. 10.1109/EDOCW.2013.22

Singh, K. (2015). *Survey of NoSQL Database Engines for Big Data* (Master's Thesis). Aalto University School of Science.

Windmann, S., Maier, A., Niggemann, O., Frey, C., Bernardi, A., Gu, Y., ... Kraus, R. (2015). Big data analysis of manufacturing processes. *Journal of Physics: Conference Series*, *659*(1), 12055. doi:10.1088/1742-6596/659/1/012055

ADDITIONAL READING

Babu, M. P., & Sastry, S. H. (2014, June). Big data and predictive analytics in ERP systems for automating decision making process. In *2014 IEEE 5th International Conference on Software Engineering and Service Science* (pp. 259-262). IEEE. 10.1109/ICSESS.2014.6933558

Bizer, C., Boncz, P., Brodie, M. L., & Erling, O. (2012). The meaningful use of big data: Four perspectives--four challenges. *SIGMOD Record*, *40*(4), 56–60. doi:10.1145/2094114.2094129

Dutta, D., & Bose, I. (2015). Managing a big data project: The case of ramco cements limited. *International Journal of Production Economics*, *165*, 293–306. doi:10.1016/j.ijpe.2014.12.032

Griffin, P. A., & Wright, A. M. (2015). Commentaries on Big Data's importance for accounting and auditing. *Accounting Horizons*, *29*(2), 377–379. doi:10.2308/acch-51066

Gudivada, V., Rao, D., & Raghavan, V. (2014). Renaissance in data management systems: SQL, NoSQL, and NewSQL. *Computer*.

Shi, Z., & Wang, G. (2018). Integration of big-data ERP and business analytics (BA). *The Journal of High Technology Management Research*, *29*(2), 141–150. doi:10.1016/j.hitech.2018.09.004

KEY TERMS AND DEFINITIONS

Big Data: A collection of huge amounts of real time data in very high volume, variety, and velocity in nature that cannot be managed effectively with traditional database management systems.

Binary JavaScript Object Notation (BSON): A format for binary-coded serialization of JSON documents.

Diverse Data: It refers to different types of storage of data. The data can be stored in three ways; (1) structured data that hold in tables on traditional relational databases, (2) unstructured data that hold in like xml files, csv files, (3) semi-structured data that hold in like doc, pdf, email, and post messages on social platforms.

Horizontal Scaling: Adding new nodes (servers) to the system to increase the application workload without making any changes to the application.

JavaScript Object Notation (JSON): A text-based open standard format for exchanging data between applications.

MapReduce: A programming model that process large amounts of data stored in commodity machines for processing massive datasets in parallel.

NoSQL Systems: Non-relational, distributed database systems designed for big data storage and have capability to process data across a large number of commodity servers.

Vertical Scaling: Increasing memory, number of CPUs and cores of a computer to provide more data processing power.