Journal of Computer Sciences and Applications, 2015, Vol. 3, No. 6, 162-165
Available online at http://pubs.sciepub.com/jcsa/3/6/10

DOI:10.12691/jcsa-3-6-10

Emergence of Big Data on Cloud Computing (An Insight to Current and Future)

Amar Nath Singh, Rakesh Kumar Rath*

GIET, Baniatangi, Bhubaneswar
*Corresponding author: rakesh.rath@gmail.com

Abstract In Database the prime objective of database is to arrange the data in a well designed manner so that the data should be stored and retrieved effectively as and when required. But in traditional database like DBMS the major fault was scalability. So in this database management systems (DBMS) the following problems are arise always such as, 1. Update intensive application workloads, and 2. Decision support systems for descriptive and deep analytics. These are a critical part of the cloud infrastructure and play an important role in ensuring the smooth transition of applications from the traditional enterprise infrastructures to next generation cloud infrastructures. So these scalability natures of the data base cause a lot of problem for data integration when the amounts of data were huge. This paper presents an organized picture of the challenges faced by application developers and DBMS designers in developing and deploying internet scale applications. We have designed this paper on the basis of the following aspects, such as: (i) For supporting update heavy applications, and (ii) For ad-hoc analytics and decision support. Here we also focus on providing an in-depth analysis of systems for supporting update intensive web-applications which are mostly now a day's used in the companies and provide a survey of the state-of-the art in this domain. We have tried to crystallize the design aspect of choices made by considering some large scale database management systems, analyze the application demands and access patterns, and enumerate the desiderata for a cloud-bound DBMS.

Keywords*: cloud, big data, crystallization, data base, scalable data*

Cite This Article: Amar Nath Singh, and Rakesh Kumar Rath, "Emergence of Big Data on Cloud Computing (An Insight to Current and Future)." *Journal of Computer Sciences and Applications*, vol. 3, no. 6 (2015): 162-165. doi: 10.12691/jcsa-3-6-10.

1. Introduction

Now a day's we are mainly goes for Cloud computing, because it is an extremely successful paradigm of service oriented computing, and has revolutionized the way computing infrastructure is abstracted and used In case of cloud computing three most popular cloud paradigms include:

1. Infrastructure as a Service (IaaS),
2. Platform as a Service (PaaS), and
3. Software as a Service (SaaS).

The above three concept however can also be extended to Database as a Service or In case of cloud computing three most popular cloud paradigms include:

1. Infrastructure as a Service (IaaS),
2. Platform as a Service (PaaS), and
3. Software as a Service (SaaS).

The above three concept however can also be extended to Database as a Service or management has been the vision of the database research community for more than three decades. The Initial designs include distributed databases [11] for update intensive workloads, and parallel database systems [10] for analytical workloads. On the other hand if we consider the case of Parallel databases, it mainly grew beyond prototype systems to large commercial systems, but distributed database systems were not very successful and were never commercialized.

In summary, the quest for conquering the challenges posed by management of big data has led to a plethora of systems. Furthermore, applications being deployed in the cloud have their own set of desideratum which opens up various possibilities in the design space.

2. Outline

Our tutorial is intended to span three hours, and is divided into four broad subsections. We now provide a high-level summary of the goals of each of these subsections where we provide a broad survey of the state-of- the-art as well as delve deeper into the design of some systems to crystallize some design choices and enumerate new challenges for data management in the cloud .

2.1. Background: Scalable Data Management

In the first part of the tutorial, we outline the features of the cloud that make it attractive for deploying new applications and provide the necessary background by analyzing the different scalable data management solutions. Our background study encompasses both classes of systems: (i) for supporting update heavy applications, and (ii) for ad-hoc analytics and decision

support. The goal is to provide an elaborate summary of the various approaches for scaling to the management of big data and for supporting both classes of applications and systems. We outline the interesting design choices and scope of the different systems. We begin by motivating the discussion of cloud data management and outline some of the reasons why cloud computing is relevant and successful. We also discuss the major enabling features that have led to its widespread popularity and success [3]. In particular, a system in the cloud must posses some features (often referred to as cloud features) to be able to effectively utilize the cloud economies. These cloud features include: scalability, elasticity, fault-tolerance, self-manageability, and ability to run on commodity hardware. Most traditional relational database systems were designed for enterprise infrastructures and hence were not designed to meet all these goals. This calls for novel data management systems for cloud infrastructures. Having set the stage for the need of scalable data management solutions for the cloud, we delve deeper to analyze the different scalable data management systems.

Systems for Update heavy Workloads.

Internet facing applications typically generate large amounts of data from regular usage. Systems capable of handling this large volume of updates are important to sustain this class of applications. In this part of the tutorial, we focus on a survey of scalable data management systems targeting this problem, and present a high-level overview of these systems with the intent of laying the ground for a detailed analysis to be covered in the latter half of the tutorial. We focus our discussion on the new generation of distributed Key-Value data stores [8,10] which have been extremely successful and widely adopted. We highlight the application level changes [13] and the system design level changes [3] that contributed to the success of these systems.

Systems for Data Analysis. Analyzing large amounts of data generated by different applications is critical for gaining a competitive advantage and improving customer experience. Historically, parallel database systems, such as Gamma [9], were designed to provide "descriptive" analysis of large amounts of data. The success of these prototype systems and the business incentives associated with data analysis resulted in the blossoming of a multi-billion dollar data warehousing industry with tens of commercial players. We focus our analysis on some of the design choices of these analytical systems which have contributed to their widespread success, and the impact of the cloud on such systems. We also analyze another evolving paradigm for large data analysis pioneered by MapReduce [11] and its open source counterpart Hadoop [2]. The Hadoop framework has been very successful and has seen widespread adoption in industry and academia alike. Hadoop is also being integrated to traditional data warehousing software as well as enterprises are building custom solutions for Hadoop based analytics [12]. We also survey some improvements to the Hadoop framework suggested through research prototypes such as HadoopDB [1] etc. Recently, we have also seen a raging debate on MapReduce vs. Parallel DBMS [11]. We aim to reconcile this debate by presenting an analysis of the advantages and disadvantages of the different approaches. We conclude this discussion on analytical systems by highlighting a new direction in data analysis referred to as "deep analytics" [6,9]. This new class of data analysis applications are driven by the application of complex statistical analysis and machine learning techniques on huge amounts of data to garner intelligence from the data. Having laid the foundation for scalable data management, we move our focus to the class of systems that are designed to support update heavy web-applications deployed in the cloud. We subdivide this class into two sub-classes: one where the goal of the system is to support a single large application with large amounts of data (scalable single tenant DBMS); and another where the goal of the system is to support a large number of applications each with a small data footprint (large multitenant DBMS).

2.2. Data Management for Large Applications

In this part of the tutorial, we focus on the design issues in building a DBMS for dealing with applications with single large databases. We refer to this as a large single tenant system. Figure 1 provides a schematic representation of the design goals this section of the tutorial concentrates. Many applications often start small, but with growing popularity, their data footprint continues to grow, and at one point grows beyond the limits of a traditional relational database. This has been observed specifically in the modern era of innovative application ideas whose deployment has been made feasible by the cloud economics, and whose popularity often has wide fluctuations. The application servers can easily scale out, but the data management infrastructure often becomes a bottleneck. The lack of cloud features in open source relational DBMSs.

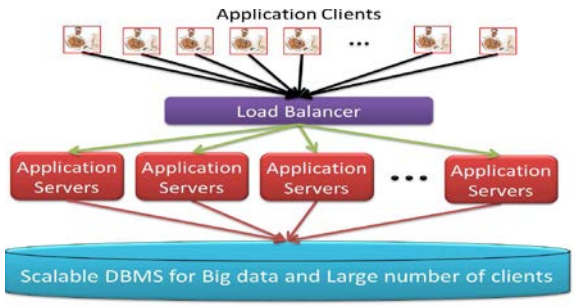

Figure 1. Scalable database management systems to support large applications with lots of data and supporting hundreds of thousands of clients

(RDBMSs) and the hefty cost associated with enterprise solutions make RDBMSs less attractive for the deployment of large scale applications in the cloud. This has resulted in the popularity of Key-Value stores: examples include Bigtable [8], PNUTS [10], Dynamo [1,8] and their open-source counterparts HBase, Cassandra, Voldemort etc. These systems have been extensively deployed in various private as well as public and commercial cloud infrastructures. We provide a survey of the popular Key-Value stores and crystallize the features and design choices made by these system that have allowed them to scale out to petabytes of data and thousands of concurrent requests – a scale which distributed databases have failed to achieve. Popularly referred to as NoSQL stores, we crystallize the design principles of these systems [3], and how these principles can be extended beyond Key-Value stores for designing

scalable systems with a richer set of functionality compared to these Key-Value stores [14]. We also provide a survey of some of the current research projects which aim to infuse the loud features in relational databases. These systems include Elas- TraS [5,12,13], DBonS3 [3,7], Project Deuteronomy [4], Relational Cloud [11], epiC [9], Hyder [6] to name a few.

2.3. Large Multitenant Databases

Another important domain for data management in the cloud is the need to support large number of applications, each of which has a small data footprint. This is referred to as a large multitenant system. Database multitenancy is traditionally considered only in the case of SaaS with Salesforce.com being a canonical example [1,3] where different tenants share the same database tables. But different models of multitenancy are relevant in the context of the different cloud paradigms. For instance, a PaaS provider, dealing with a large number of applications with very different schemas, might require a different form of sharing in contrast to the shared table approach used in traditional designs [1,3,5]. Yang et al. [2,3], for example, suggest a different model of multitenancy where every tenant has its own independent database instance. In summary, different multitenancy models are uitable for different cloud paradigms. Figure 2 provides an illustration of some of the different forms of multitenancy and the ifferent trade-offs associated with different forms of sharing [2,6]. The different models share resources at different levels of abstraction and provide different isolation guarantees. The goal of this section of the tutorial is to survey the different approaches to ultitenancy in a database, and garner the understanding of the requirements and applicability of the different multitenancy models or infusing cloud features into such a system. We also analyze the different challenges involved indesigning multitenant systems or serving hundreds of thousands of clients, and the impact on the choice of the multitenancy models on these designs.

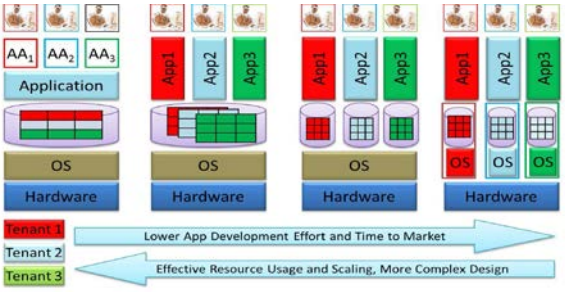

Figure 2. Different models of multitenancy. From left to right, they correspond to shared table, shared database, shared OS, and shared hardware

2.4. Major Open Problems

In this concluding part of the tutorial, we identify some of the major open problems that must be addressed to ensure the success of data management systems in the cloud. In summary, a single perfect data management solution for the cloud is yet to be esigned. Different systems target different aspects in the design space, and multiple open problems still remain. With respect to Key-Value stores, though these systems are popular, they only support very simple functionality. Providing support for ad-hoc

querying on top of a Key-Value store [4] or providing consistency guarantees at different access granularities [14] are some research efforts targeted towards enriching the functionality supported by Key-Value stores. Further research however is needed to generalize these proposals to different classes of applications and different Key-Value stores. Similarly, extending the Key- alue stores for supporting richer set of applications is also an important research challenge. On the other hand, in the domain of relational database management, an important open problem is how to make the systems elastic for effectively utilizing the available resources and minimizing the cost of operation. Furthermore, characterizing the different consistency semantics that can be provided at different scales, and effective techniques for load balancing are also critical aspects of the system. Designing scalable, elastic, and autonomic multitenant database systems is another important challenge that must also be addressed. In addition, ensuring the security and privacy of the data outsourced to the cloud is also an important problem for ensuring the success of data management systems in the cloud.

3. Goals of the Tutorial

3.1. Learning Outcomes

Following are the learning outcomes from this tutorial:

•State-of-the-art in scalable data management for traditional and cloud computing infrastructures for both update heavy as well as analytical workloads. Summary of current research projects and

future research directions.

• Design choices that have led to the success of the scalable systems, and the errors that limited the success of some other systems.

• Design principles that should be carried over in designing the next generation of data management systems for the cloud.

•Understanding the design space for DBMS targeted to supporting update intensive workloads for supporting large single tenant systems and large multitenant systems.

•Understanding the different forms of multitenancy in the database

layer.

• A list of open research challenges in cloud data management that must be addressed to ensure the continued success of DBMS.

3.2. Intended Audience

This tutorial is intended to benefit researchers and system designers in the broad area of scalable data management for traditional as well as cloud data platforms. Our tutorial would benefit both designers of the systems as well as users of the systems since a survey of the current systems and an in-depth understanding will is essential for choosing the appropriate system as well as designing an effective system. This tutorial does not require any knowledge on scalable data management systems.

3.3. Earlier Presentation and Changes

An earlier version of this tutorial was presented at the VLDB 2010 Conference [2]. This version of the tutorial will augment the previous version with new techniques and results presented at recent conferences. We will also add more discussion on run-time system monitoring and performance monitoring, an important aspect of a successful cloud infrastructure.

4. Biographical Sketches

Divyakant Agrawal is currently a Professor of the Department of Computer Science at the University of California, Santa Barbara. He was a visiting researcher at IBM Almaden Research Center, Visiting Senior Research Scientist at NEC Computing and Communication Research Laboratories, VP of Data Solutions and Advertising Systems at ASK.com, and recently a visiting researcher at NUS. He has served in the program committees of many leading conferences and was the PC Chair of SIGMOD 2010, and on the editorial boards of the VLDB journal and the Proceedings of the VLDB. His research expertise is in the areas of database systems, distributed computing, data warehousing, and large-scale information systems. He is an ACM distinguished scientist. Sudipto Das is currently a PhD candidate in the Department of Computer Science at University of California, Santa Barbara. His research interests lie in the area of scalable data management in cloud computing infrastructures, specifically on elastic and transactional database systems for cloud platforms. He is the recipient of UCSB Computer Science Outstanding Teaching Assistant award for Fall 2008 and the UCSB graduate division Dissertation Fellowship for 2011. He is a student member of ACM and IEEE. Amr El Abbadi is currently a Professor and Chair of the Department of Computer Science at the University of California, Santa Barbara. He has held visiting professor positions at the University of Campinas in Brazil, IBM Almaden Research Center, the Swedish Institute of Computer Science in Stockholm, and at IRISA at the University of Rennes in France. He was Vice Chair of ICDCS 1999, Vice Chair for ICDE 2002, the Americas Program Chair for VLDB 2000, and is the Program Chair for SIGSPATIAL 2010. He served as a board member of the VLDB Endowment from 2002 to 2008. In 2007, Prof. El Abbadi received the UCSB Senate Oustanding Mentorship Award for his excellence in mentoring graduate students. His research interests lie in the broad area of scalable database and distributed systems. He is a fellow of the ACM.

References

[1] A. Abouzeid, K. B. Pawlikowski, D. J. Abadi, A. Rasin, and A. Silberschatz. HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads. PVLDB, 2(1): 922-933, 2009.

[2] D. Agrawal, S. Das, and A. E. Abbadi. Big data and cloud computing: New wine or just new bottles? PVLDB, 3(2):1647-1648, 2010.

[3] D. Agrawal, A. El Abbadi, S. Antony, and S. Das. Data Management Challenges in Cloud Computing Infrastructures. In DNIS, pages 1-10, 2010.

[4] P. Agrawal, A. Silberstein, B. F. Cooper, U. Srivastava, and R. Ramakrishnan. Asynchronous view maintenance for vlsd databases. In SIGMOD Conference, pages 179-192, 2009.

[5] S. Aulbach, D. Jacobs, A. Kemper, and M. Seibold. A comparison of flexible schemas for software as a service. In SIGMOD, pages 881-888, 2009.

[6] P. Bernstein, C. Rein, and S. Das. Hyder – A Transactional Record Manager for Shared Flash. In CIDR, 2011.

[7] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska. Building a database on S3. In SIGMOD, pages 251-264, 2008.

[8] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A Distributed Storage System for Structured Data. In OSDI, pages 205-218, 2006.

[9] J. Cohen, B. Dolan, M. Dunlap, J. M. Hellerstein, and C. Welton. Mad skills: New analysis practices for big data. PVLDB, 2(2):1481-1492, 2009.

[10] B. F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H.-A. Jacobsen, N. Puz, D. Weaver, and R. Yerneni. PNUTS: Yahoo!'s hosted data serving platform. Proc. VLDB Endow., 1(2):1277-1288, 2008.

[11] C. Curino, E. Jones, Y. Zhang, E. Wu, and S. Madden. Relational Cloud: The Case for a Database Service. Technical Report 2010-14, CSAIL, MIT, 2010. http://hdl.handle.net/1721.1/52606.

[12] S. Das, S. Agarwal, D. Agrawal, and A. El Abbadi. ElasTraS: An Elastic, Scalable, and Self Managing Transactional Database for the Cloud. Technical Report 2010-04, CS, UCSB, 2010.

[13] S. Das, D. Agrawal, and A. El Abbadi. ElasTraS: An Elastic Transactional Data Store in the Cloud. In USENIX HotCloud, 2009.

[14] S. Das, D. Agrawal, and A. El Abbadi. G-Store: A Scalable Data Store for Transactional Multi key Access in the Cloud. In ACM SOCC, 2010.