MAT JOURNALS
Journal of Image Processing and Artificial Intelligence
Volume 1 Issue 3

Designee of a Scalable Database Management Systems (DBMS)

Md. Masud Parvez
Department of CSE, Southeast University, Bangladesh
E-mail: masud.magura@gmail.com

Abstract

Scalable database management systems (DBMS)-both for update intensive application workloads as well as decision support systems for descriptive and deep analytics-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. Though scalable data management has been a vision for more than three decades and much research has focused on large scale data management in traditional enterprise setting, cloud computing brings its own set of novel challenges that must be addressed to ensure the success of data management solutions in the cloud environment. This tutorial presents an organized picture of the challenges faced by application developers and DBMS designers in developing and deploying internet scale applications. Our background study encompasses both classes of systems: (I) for supporting update heavy applications and (II) for ad-hoc analytics and decision support. We then focus on providing an in-depth analysis of systems for supporting update intensive web-applications and provide a survey of the state-of-the-art in this domain. We crystallize the design choices made by some successful systems large scale database management systems, analyze the application demands and access patterns, and enumerate the desiderata for a cloud-bound DBMS.

Keywords: *DBMS, management, cloud infrastructure, applications, design*

INTRODUCTION

Cloud computing is an extremely successful paradigm of ser-vice oriented computing and has revolutionized the way computing infrastructure is abstracted and used. Three most popular cloud paradigms include: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). The concept however, can also be extended to Database as a Service or Storage as a Service. Elasticity, pay-per-use, low upfront investment, low

time to market, and transfer of risks are some of the major enabling features that make cloud computing a ubiquitous paradigm for deploying novel applications which were not economically feasible in a traditional enterprise infrastructure settings. This has seen a proliferation in the number of applications which leverage various cloud platforms, resulting in a tremendous increase in the scale of the data generated as well as consumed by such applications. Scalable database management systems (DBMS)-both for update intensive application workloads, as well as decision support systems are thus a critical part of the cloud infrastructure.

Scalable and distributed data management has been the vision of the database research community for more than three decades. Much research has focused on designing scalable systems for both update intensive workloads as well as ad-hoc analysis workloads. Initial designs include distributed databases for update intensive workloads, and parallel database systems for analytical workloads [1, 2]. Parallel databases grew beyond prototype systems to large commercial systems, but distributed database systems were not very successful and were never commercialized-rather various ad-hoc approaches to scaling were used. Changes in the data access patterns of applications and the need to scale out to thousands of commodity machines led to the birth of a new class of systems referred to as Key-Value stores which are now being widely adopted by various enterprises [3–5]. In the domain of data analysis, the Map Reduce paradigm and its open-source implementation Hadoop has also seen widespread adoption in industry and academia alike [6, 7]. Solutions have also been proposed to improve Hadoop based systems in terms of usability and performance [8, 9]. 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. This has often resulted in discussions relating to the appropriate systems for a specific set of application requirements, the research challenges in data management for the cloud, and what is novel in the cloud for database researchers? This tutorial aims to clarify some of these questions. We also aim to address one basic question: whether the cloud computing poses new challenges in data management or it is just a reincarnation of old problems?

We provide a comprehensive background study of the state-of-the-art systems for scalable data management and analysis. We also identify the critical aspects in the design of different systems and the applicability and scope of these systems. We further focus on a set of systems which are designed to handle update heavy workloads for supporting internet facing applications. We identify some of the design challenges which application and system designers face in developing and deploying new applications and systems, and expand on some of the major challenges that need to be addressed to ensure the smooth transition of applications from traditional enterprise infrastructure to the next generation cloud infrastructure. A thorough understanding of current solutions and a precise characterization of the design space are essential for clearing the "cloudy skies of data management" and ensuring the success of DBMSs in the cloud, thus emulating the success enjoyed by relational databases in traditional enterprise settings.

BACKGROUND: SCALABLE DATA MANAGEMENT

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. The 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. I outline the interesting design choices and scope of the different systems. I 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 [10]. In particular, a system in the cloud must possess 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 us-age. 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 which have been extremely successful and widely adopted [3 –5]. We highlight the application level changes and the system design level changes that contributed to the success of these systems [5, 10–12].

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, were de

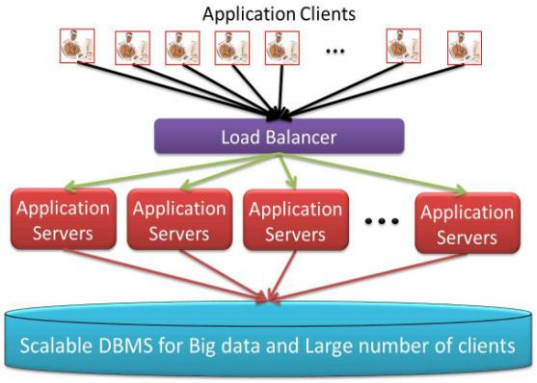

Fig. 1: *Scalable Database Management Systems to Support Large Applications with Lots of Data and Supporting Hundreds of Thousands of Clients.*

signed to provide "descriptive" analysis of large amounts of data [2]. 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 Map Reduce and its open source counterpart Hadoop [6, 7]. 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 [9]. We also survey some improvements to the Hadoop framework suggested through research prototypes such as HadoopDB etc. [8]. Recently, we have also seen a raging debate on Map Reduce vs. Parallel DBMS [13].

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" [14, 15].

This new class of data analysis applications is 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 sub-divide 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).

DATA MANAGEMENT FOR LARGE APPLICATIONS

In this part, i focus on the design issues in building a DBMS for dealing with applications with single large databases. i 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 bottle-neck. The lack of cloud features in open source relational DBMSs

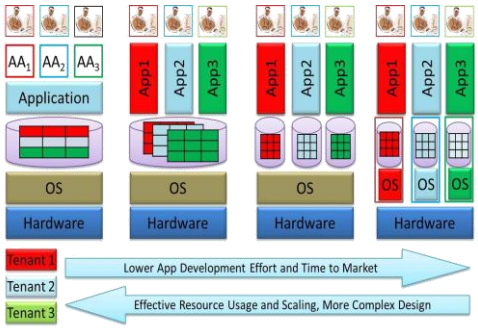

Fig. 2: *Different Models of Multitenancy. From Left to Right, they Correspond to Shared Table, Shared Database, Shared OS and Shared Hardware.*

(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 Big table, PNUTS, Dynamo and their open-source counterparts HBase, Cassandra, Voldemort etc. [3–5]. 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 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 [10, 16]. We also provide a survey of some of the current research projects which aim to infuse the cloud features in relational databases. These systems include Elas-TraS, DBonS3, Project Deuteronomy, Relational Cloud, epiC, Hyder to name a few [17–25].

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 multi-tenant system. Database multitenancy is traditionally considered only in the case of SaaS with Salesforce.com being a canonical example where different tenants share the same database tables [26]. 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 [26, 27]. Yang *et al.* for example, suggest a different model of multitenancy

where every tenant has its own independent database instance. In summary, different multitenancy models are suitable for different cloud paradigms [28]. Figure 2 provide an illustration of some of the different forms of multitenancy and the different trade-offs associated with different forms of sharing [29, 30]. 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 multitenancy in a database, and garner the understanding of the requirements and applicability of the different multitenancy models for infusing cloud features into such a system. We also analyze the different challenges involved in designing multitenant systems for serving hundreds of thousands of clients and the impact on the choice of the multitenancy models on these designs.

MAJOR OPEN PROBLEMS

In this concluding part, I 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 designed. Different systems target different aspects in the de-sign 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 or providing consistency guarantees at different access granularities are some research efforts targeted towards enriching the functionality supported by Key-Value stores [16, 31]. Further research, however, is needed to generalize these proposals to different classes of applications and different Key-Value stores. Similarly, extending the Key-Value 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.

OUTCOMES

Following are the learning outcomes:

- 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 DBMSs.

CONCLUSION

This manuscript is intended to benefit researchers and system designers in the broad area of scalable data management for traditional as well as cloud data platforms. It would benefit both de-signers 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.

REFERENCES

1. J. B. Rothnie Jr., P. A. Bernstein, S. Fox.
2. Introduction to a System for Distributed Databases (SDD-1). *ACM Trans. Database Syst.* 1980; 5(1): 1–17p.
3. D. J. Dewitt, S. Ghandeharizadeh, D. A. Schneider et al. The gamma database machine project. *IEEE Trans. on Knowl. and Data Eng.* 1990; 2(1): 44–62p.
4. F. Chang, J. Dean, S. Ghemawat et al.
5. Bigtable: A distributed storage system for structured data. *In OSDI*. 2006; 205–218p.
6. B. F. Cooper, R. Ramakrishnan, U. Srivastava et al.

a. PNUTS: Yahoo!'s hosted data serving platform. *Proc. VLDB Endow.* 2008; 1(2): 1277–1288p.

7. G. DeCandia, D. Hastorun, M. Jampani et al. Dynamo: Amazon's

highly available key-value store. *In SOSP*. 2007; 205–220p.

8. J. Dean, S. Ghemawat. MapReduce: simplified data proces sing on large clusters. *In OSDI*. 2004; 137–150p.
9. The Apache Hadoop Project. http://hadoop.apache.or g/core/, 2009.
10. A. Abouzeid, K. B. Pawlikowski, D. J. Abadi et al. HadoopDB: An architectural hybrid of mapreduce and DBMS technologies for analytical workloads. *PVLDB*. 2009; 2(1): 922–933p.
11. A. Thusoo, J. S. Sarma, N. Jain et al. Hive-A Warehousing Solution Over a Map-Reduce Framework. *PVLDB*. 2009; 2(2): 1626–1629p.
12. D. Agrawal, A. El Abbadi, S. Antony et al. Data Management Challenges in Cloud Computing Infrastructures. In *DNIS*, pages 1–10, 2010.
13. W. Vogels. Data access patterns in the amazon.com technology platform. *In VLDB*. 2007; 1p. VLDB Endowment.
14. P. Helland. Life beyond distributed transactions: an apostate's opinion. *In CIDR*. 2007; 132–141p.
15. A. Pavlo, E. Paulson, A. Rasin et al. A comparison of approaches to large-scale data analysis. *In SIGMOD*. 2009; 165–178p.
16. J. Cohen, B. Dolan, M. Dunlap et al. Mad skills: New analysis practices for big data. *PVLDB*. 2009: 2(2): 1481–1492p.
17. S. Das, Y. Sismanis, K. Beyer et al.
18. Ricardo: integrating r and hadoop. *In SIGMOD*; 2010.
19. S. Das, D. Agrawal, A. El Abbadi. G-Store: A scalable data store for transactional multi key access in the cloud. *In ACM SOCC*; 2010.
20. S. Das, S. Agarwal, D. Agrawal et al. ElasTraS: An elastic, scalable, and self managing transactional database for the cloud. *Technical Report*; 2010.
21. S. Das, D. Agrawal, A. El Abbadi. ElasTraS: An elastic transactional data store in the cloud. *In USENIX Hot Cloud*; 2009.
22. S. Das, S. Nishimura, D. Agrawal et al. Live database migration for elasticity in a multitenant database for cloud platforms. Technical Report; 2010.
23. M. Brantner, D. Florescu, D. Graf et al. Building a database on S3. *In SIGMOD*. 2008; 251–264p.
24. T. Kraska, M. Hentschel, G. Alonso et al. Consistency rationing in the cloud: Pay only when it matters. *PVLDB*. 2009; 2(1): 253–264p.

25. D. B. Lomet, A. Fekete, G. Weikum et al. Unbundling transaction services in the cloud. *In CIDR Perspectives*; 2009.
26. C. Curino, E. Jones, Y. Zhang et al. Relational cloud: the case for a database service. technical report. CSAIL; 2010-14. http://hdl.handle.net/1721.1/52606.
27. H. T. Vo, C. Chen, B. C. Ooi. Towards elastic transactional cloud storage with range query support. *PVLDB*. 2010; 3(1): 506–517p.
28. P. Bernstein, C. Rein, S. Das. Hyder- A transactional record manager for shared flash. *In CIDR*. 2011.
29. C. D. Weissman, S. Bobrowski. The design of the force.com multitenant internet application development platform. *In SIGMOD*. 2009; 889–896p.
30. S. Aulbach, D. Jacobs, A. Kemper et al. A comparison of flexible schemas for software as a service. *In SIGMOD*. 2009; 881–888p.
31. F. Yang, J. Shanmugasundaram, R. Yerneni. A scalable data platform for a large number of small applications. *In CIDR*. 2009.
32. D. Jacobs, S. Aulbach. Ruminations on multi-tenant databases. *In BTW*. 2007; 514–521p.
33. B. Reinwald. Database support for multi-tenant applications. *In IEEE Workshop on Information and Software as Services*. 2010.
34. P. Agrawal, A. Silberstein, B. F. Cooper et al. Asynchronous view maintenance for vlsd databases. *In SIGMOD Conference*. 2009; 179–192p.