Dieses Dokument ist eine Zweitveröffentlichung (Postprint) /

This is a self-archiving document (accepted version):

Kai-Uwe Sattler, Wolfgang Lehner

Database as a service (DBaaS)

Erstveröffentlichung in / First published in:

IEEE 26th International Conference on Data Engineering (ICDE 2010). Long Beach, 01.03-06.03.2010. IEEE, S. 1216-1217. ISBN 978-1-4244-5445-7

DOI: https://doi.org/10.1109/ICDE.2010.5447723

Diese Version ist verfügbar / This version is available on:

https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa2-813613

SLUB Wir führen Wissen.

TECHNISCHE UNIVERSITÄT DRESDEN

QUCOSA Quality Content of Saxony

Final edited form was published in "IEEE 26th International Conference on Data Engineering. Long Beach 2010", S. 1216-1217, ISBN 978-1-4244-5445-7
https://doi.org/10.1109/ICDE.2010.5447723

Database as a Service (DBaaS)

Wolfgang Lehner [1], Kai-Uwe Sattler [2]

[1]*Database Technology Group, Dresden University of Technology, 01062 Dresden, Germany*
[1]wolfgang.lehner@tu-dresden.de

[2]*Database Systems Research Group, Ilmenau University of Technology, 98693 Ilmenau, Germany*
[2]kus@tu-ilmenau.de

***Abstract*— Modern Web or "Eternal-Beta" applications necessitate a flexible and easy-to-use data management platform that allows the evolutionary development of databases and applications. The classical approach of relational database systems following strictly the ACID properties has to be extended by an extensible and easy-to-use persistency layer with specialized DB features. Using the underlying concept of Software as a Service (SaaS) also enables an economic advantage based on the "economy of the scale", where application and system environments only need to be provided once but can be used by thousands of users. Within this tutorial, we are looking at the current state-of-the-art from different perspectives. We outline foundations and techniques to build database services based on the SaaS-paradigm. We discuss requirements from a programming perspective, show different dimensions in the context of consistency and reliability, and also describe different non-functional properties under the umbrella of Service-Level agreements (SLA).**

I. Motivation

The efficient management of a consistent and integrated database is a central task in modern IT and highly relevant for science and industry. Hardly any critical enterprise solution comes without any functionality for managing data in its different forms. Advantages of database systems (DBS) include the possibility to store data persistently while guaranteeing physical data independence, the ability to process declarative queries, and their general usability independent of specific applications.

The database research field has seen a variety of approaches for different domains. The first systems to include the ACID properties were relational database systems with their simple structure and easy operability. Soon, object-oriented concepts, triggers, or XML processing complemented their functionality. However, such systems are expensive, monolithic server systems that require immense maintenance efforts and an understanding of complex query languages and schema design.

Even though these stable, scalable systems will not become obsolete, modern Web or "Eternal-Beta" applications necessitate a more flexible and easy-to-use data management platform that allows the evolutionary development of databases and applications. We are currently also faced with a trend to move away from the closed world perspective of classic database system, towards an open world principle as found, for example, in recent Internet trends. Instead of modeling the complete business environment, we increasingly follow an approach to model the core entities of a real world scenario that can then be adjusted and expanded continuously.

Additionally, the success of relational database systems has led to a huge number of systems within an organization with heterogeneous schemas, with systems coming from different vendors, and with multiple places and different database admins running such systems. The result is a number of "fixes" such as federated db architectures, database-middleware systems, and data warehouse systems. However, all these attempts address problems from the perspective of classic databases and show the need for a new wave of data management [3].

II. From SaaS to DBaaS

The concept of Software as a Service (SaaS) means that providers allow applications to consume certain applications via Internet. The central economic advantage based on the "economy of the scale" principle is obvious: application and system environment only need to be provided once but can be used by thousands of users. Enterprises may focus on their core competencies instead of having to maintain complex IT landscapes. Recent developments in terms of enterprise solutions obey this trend just like social network applications, such as Flickr, YouTube or Facebook (all of which are easy to use and customizable). This makes new business models and marketplaces for user-generated solutions possible (see SalesForce.com as an example).

All these developments call for an extensible and easy-to-use persistency layer with classic DB features. First-generation SaaS systems still build on relational DB technology but extend it with application logic to meet functional requirements (e.g., multi-tenancy implemented on top of the classic DBS. As a consequence, such complex infrastructures require heavy maintenance.

To face these issues, large providers (Yahoo!, Google, Amazon) have begun to implement platforms for DB services supporting the idea of data outsourcing. Providing data management in this way as a reliable service obviates the need for customers to invest in expensive software and hardware and to employ administration staff.

Current Cloud DBS (e.g., PNUTS, HBase, SimpleDB, Google BigTable) allow to submit queries to DBs with generic schemas. Basically, such systems consist of simple container-like data management services with put/get semantics for data blocks of unknown structure. Additional functionality (transactional features, quality guarantees) are difficult to incorporate.

Current research tries to use these approaches and integrate application semantics (for example: "good" units of paral-

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Final edited form was published in "IEEE 26th International Conference on Data Engineering. Long Beach 2010", S. 1216-1217, ISBN 978-1-4244-5445-7
https://doi.org/10.1109/ICDE.2010.5447723

lelization) into queries. A key role is taken by the map-reduce paradigm with its simplicity and ability to represent complex queries in a semi-procedural manner. From a programming perspective, languages such as LINQ, PigLatin, Sawzall, Space or SCOPE are a first step into this general direction.

Finally, database services have to consider the semantic perspective. For example, Cloud DBS show only limited support of multiple tenants with slightly modified schemas for a heterogeneous user base. Also, model management aspects (e.g., semantic merging of isolated schemas) are often neglected. Hence, one of the goals of ongoing database research is to close the gap between classic database systems with all their restrictions and simple but highly scalable data management platforms. Only then will next-generation data-centric SaaS applications be able to support high scalability, extensibility and non-functional guarantees at the same time.

The proposed tutorial addresses fundamental challenges posed by the need and desire to provide database functionality in the context of the Database as a Service (DBaaS) paradigm for database outsourcing.

III. Scope of the Tutorial

We expect that attendees of this tutorial will take home an understanding of the problems and benefits of database services, basic knowledge of techniques for implementing such services particularly in cloud environments as well as of appropriate programming and query models. The tutorial will also cover a survey of existing techniques addressing the special requirements of database services with respect to non-functional properties such as consistency, reliability, security, and trustworthiness. Finally, the tutorial tries to give an insight into future trends and challenges in database services and data clouds.

We will start our tutorial with a brief motivation of the new paradigm of cloud computing and its impact to data outsourcing and service-oriented computing in data-intensive applications. The second part of the tutorial will outline foundations and architectures of DBaaS solutions by reviewing the most prominent aspects of the concepts of Software as a Service and Cloud Computing Infrastructures. We will look at these aspects from two directions. On the one hand, we factor out the key characteristics of these concepts and discuss the potential impact for a DBaaS solution. On the other hand, we will review and characterize the most relevant solutions of the large providers. This review will paint the global architectures of the solutions starting from the base services up to database-related capabilities. The different setup will be compared and discussed with respect to the capability of serving as a DBaaS platform. The main part of this tutorial will give a survey on techniques addressing the special requirements for building database services. We focus on techniques from the following classes:

- **Programming models:** Though, declarative query languages like SQL and XQuery are the dominant languages for building database applications, recent developments in mapping and Web frameworks, language-integrated query interfaces as well as data-flow languages illustrate, that the complexity of a full-fledged query language is not always acceptable. On the other hand, new paradigms such as Google's map- reduce framework gain much attention in applications where huge data sets have to be processed on large clusters. Within this block, we will compare these different approaches and particularly discuss the mapping and integration of declarative query language and map-reduce.
- **Consistency & reliability:** Consistency and reliability are important properties of database services not only in mission-critical applications. Whereas reliability can be achieved by redundancy (replication) and fail-over techniques, data consistency in large distributed systems is a big challenge because classical techniques are typically not scalable. Thus, in this block we will discuss Brewer's CAP theorem and its consequences. We will present alternative consistency models such as eventual consistency and variations and discuss techniques to implement these models [2].
- **Security & Trustworthiness:** DBaaS represents basically a data outsourcing paradigm which will be commercially successful only if security and confidentially of the outsourced database is ensured. We discuss different approaches addressing this problem, ranging from techniques to support the direct execution of queries on encrypted data to dispersal-based approaches where data is split into n pieces which are then stored at different places [1].
- **Predictability & QoS:** When using data management services in distributed environments, users give up control over their data and employed resources. Thus, we need to give guarantees for non-functional properties. We explain how to adjust Service-Level Agreements (SLA) to the requirements posed by data management services and simultaneously exploit the options given by virtualized infrastructures, provisioning, and capacity planning. We will look at techniques in the context of query workload management to discover changes in the usage behavior of database services. In detail, we will outline proposed concepts to represent workload data in order to estimate and guarantee query response times be scheduling.

We conclude the tutorial by reviewing the most prominient issues and list the major challenges and trends, we see in this context.

References

[1] D. Agrawal, A. El Abbadi, F. Emekci, and A. Metwally. Database Management as a Service: Challenges and Opportunities. In *Proc. 25th Intl. Conf. on Data Engineering (ICDE 2009), Shanghai, China*, pages 1709–1716, 2009.

[2] S. Finkelstein, D. Jacobs, and R. Brendle. Principles for Inconsistency. In *Proc. 4th Biennial Conf. on Innovative Data Systems Research (CIDR 2009), Asilomar, CA*, 2009.

[3] M. Stonebraker and U. Çetintemel. One Size Fits All: An Idea Whose Time Has Come and Gone (Abstract). In *Proc. 21st Intl. Conf. on Data Engineering (ICDE 2005), Tokyo, Japan*, pages 2–11, 2005.

Provided by Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden