Is Distributed Database Evaluation Cloud-ready?

Daniel Seybold and Jörg Domaschka

Ulm University, Institute of Information Resource Management, Ulm, Germany
{daniel.seybold, joerg.domaschka}@uni-ulm.de

Abstract. The database landscape has significantly evolved over the last decade as cloud computing enables to run distributed databases on virtually unlimited cloud resources. Hence, the already non-trivial task of selecting and deploying a distributed database system becomes more challenging. Database evaluation frameworks aim at easing this task by guiding the database selection and deployment decision. The evaluation of databases has evolved as well by moving the evaluation focus from performance to distribution aspects such as scalability and elasticity. This paper presents a cloud-centric analysis of distributed database evaluation frameworks based on evaluation tiers and framework requirements. It analysis eight well adopted evaluation frameworks. The results point out that the evaluation tiers performance, scalability, elasticity and consistency are well supported, in contrast to resource selection and availability. Further, the analysed frameworks do not support cloud-centric requirements but support classic evaluation requirements.

Keywords: NoSQL, distributed database, database evaluation, cloud

1 Introduction

Relational database management systems (RDBMS) have been the common choice for persisting data for many decades. Yet, the database landscape has changed over the last decade and a plethora of new database management systems (DBMS) have evolved, namely NoSQL [20] and NewSQL [15]. These are promising persistence solutions not only for Web applications, but also for new domains such as "BigData" and "IoT". While NewSQL database systems are inspired by the relational storage model, the storage models of NoSQL database system can be further classified into key-value stores, document-oriented stores, column-oriented stores and graph-oriented stores [20]. NoSQL and NewSQL DBMS are designed to satisfy requirements such as high performance or scalability by running on commodity hardware as a distributed database management system (DDBMS), providing a single DBMS, which is spread over multiple nodes. An element of the overall DDBMS is termed *database node*.

An enabler of the DBMS evolvement is cloud computing by providing fast access to commodity hardware via elastically, on-demand, self-service resource provisioning [17]. Infrastructure as a Service (IaaS) is the preferable way to

deploy a DDBMS, requiring a high degree of flexibility in compute, storage and network resources [17].

With the number of available NoSQL and NewSQL systems and cloud resource offerings, the database selection and deployment on the cloud is a challenging task. Hence, DBMS evaluation is a common approach to guide these decisions. With the evolvement of the DBMSs, the landscape of database evaluation frameworks (DB-EFs) has evolved as well: from single node evaluation, *e.g.* TPC-E [1] of the Transaction Processing Performance Council (TPC), to DDBMS evaluation, *e.g.* the Yahoo Cloud Serving Benchmark (YCSB) [7], adding new evaluation tiers such as scalability, elasticity or consistency. Yet, these DB-EFs aim at different evaluation tiers and differ towards common DB-EF requirements [14][6], especially with respect to cloud computing.

In order to facilitate the selection and deployment of DDBMS in the cloud, we present an analysis of DB-EF with the focus on exploiting cloud computing. Our contribution is threefold by (1) defining relevant evaluation tiers for DDBMS deployed in the cloud; (2) extending existing requirements towards DDBMS evaluation with cloud specific requirements; (3) analyse existing evaluation frameworks based on the evaluation tiers and evaluation requirements.

The remainder is structured as follows: Section 2 introduces the background on DBMS evaluation. Section 3 defines the evaluation tiers while Section 4 defines the requirements towards evaluation frameworks. Section 5 analysis and discusses existing frameworks. Section 6 concludes.

2 Background

Evaluating DBMS imposes challenges for the evaluation frameworks itself, which have been discussed over decades. A first, but still valid guideline for evaluating RDBMS defines the requirements such as relevance to an application domain, portability to allow benchmarking of different systems, scalability to support benchmarking large systems, and simplicity to ensure that the results are easy to understand [14]. A more recent guideline adds the DDBMS and the resulting challenges for supporting different deployment topologies and coordination of distributed experiments [6]. By adopting these challenges, several DB-EFs have been established over the years, which are analysed in Section 5.

An overview of existing DB-EF focuses on the tiers availability and consistency. Yet, general requirements for DB-EF are not introduced and cloud specific characteristics are not considered [11]. An overview of DB-EFs for NoSQL database systems is provided by [19]. Yet, the focus lies on evaluating the data model capabilities, without taking explicitly into accout DDBMS aspects and the usage of cloud resources. Evaluating the dimensions of consistency in DDBMS, is also analysed by [5], introducing client-centric and data-centric consistency metrics. Related DB-EF for consistency are presented and missing features for fine-grained consistency evaluation are outlined. An overview of DB-EF is included in a recommendation compendiums [16] for distributed database selection

[1] `http://www.tpc.org/tpce/`

based on functional and non-functional requirement Yet, the compendium considers only the performance evaluation.

3 Distributed Database Evaluation Tiers

With the evolving heterogeneity in DDBMSs their evaluation becomes even more challenging. DBMS evaluation is driven by **workload domains (WD)**. On a high level WDs can classified into *transactional* (TW) [9], *web-oriented* (WOW) [9], *Big Data* (BDW) [12] and *synthetic* workloads (SW) [7]. These WDs drive the need for considering various evaluation tiers, which are distilled out of database and cloud research.

Resource selection (RS) determines the best matching resources to run a DBMS. For traditional RDBMSs the focus lies on single node resources, (CPU, memory, storage). For DDBMSs network, locality and number of nodes became important factors. By using cloud resources for a DBMS, the cloud providers tend to offer more *heterogeneous resources* such as VMs with dedicated storage architectures[2], container based resources[3], or dedicated resource locations from data center to physical host level.

Performance (P) evaluates the behaviour of a DBMS against a specific kind of workload. Performance metrics are *throughput* and *latency*, which are measured by the evaluation framework.

Scalability (S) defines the capability to process arbitrary workload sizes by adapting the DBMS by scaling *vertically* (scale-up/down) or *horizontally* (scale-in/out) [1]. Scaling vertically changes the computing resources of a single node. Horizontal scaling adds nodes to a DDBMS cluster (scale-out) or removes nodes (scale-in) In the following the term scalability implies horizontal scalability Measuring scalability is performed by correlating throughput and latency for growing cluster sizes and workloads. A high scalability rating is represented by constant latency and proportionally growing throughput with respect to the number of nodes and the workload size [7].

Elasticity (E) defines the ability to cope with sudden workload fluctuations without service disruption [1]. Elasticity metrics are *speedup* and *scaleup* [7]. Speedup refers to the required time for a scaling action, *i.e.* adapting the cluster size, redistributing data and stabilising the cluster. Scaleup refers to the benefit of this action, *i.e.* the throughput/latency development with respect to the workload fluctuation.

Availability (A) represents the degree to which a DBMS is operational and accessible when required for use. The availability of a DBMS can be affected by *overload* (issuing more requests in parallel than theDBMS can handle or *failures on the resource layer* (a node failure). With respect to failures, DDBMSs apply replication of data to multiple database nodes. A common metric to measure availability with respect to node failures are the takeover time, and the performance impact.

[2] `https://aws.amazon.com/de/ec2/instance-types/`
[3] `https://wiki.openstack.org/wiki/Magnum`

Consistency (C) Distributed databases offer different consistency guarantees as there is trade-off between consistency, availability and partitioning, *i.e.* the CAP theorem [13]. Consistency can be evaluated *client-centric* (*i.e.* from the application developer perspective) and *data-centric* (*i.e.* from the database administrator perspective) [5] . Here, we only consider client-centric consistency that can be classified into *staleness* and *ordering* [5]. Staleness defines how much a replica lags behind its master. It is measured either in time or versions. Ordering defines all requests must be executed on all replicas in the same chronological order.

4 Evaluation Frameworks Requirements

Besides the evaluation tiers, DBMS evaluation imposes requirements towards the DB-EF itself. We briefly present established requirements [14],[6] as well as novel cloud-centric requirements.

Usability (U) eases the framework configuration, execution and extension by providing sufficient documentation and tools to run the evaluation. Hence, the evaluation process has to be transparent to provide objective results [14].

Distribution/Scalability (D/S) is provided by distributed workload generation, *i.e.* the framwork clients can be distributed across multiple nodes in order to increasing the workload by utilising an arbitrary amount of clients [6].

Measurements Processing (MP) defines that measurements are gathered not only in an aggregated but also in a fine-grained manner for further processing [6]. As the amount of measurements can grow rapidly for multiple or long running evaluation runs, file-based persistence might not be sufficient. Hence, advanced persistence options such as time series databases (TSDBS), will ease the dedicated processing and visualisation.

Monitoring (MO) data improves the significance of evaluation results. Hence, monitoring of the involved resources, clients ,and DBMSs should be supported by the evaluation framework to provide the basis of a thorough analysis. Again an advanced persistence solution is beneficial.

Database Abstraction (DA) enables the support of multiple DBMSs by abstracting database driver implementations. Yet, the abstraction degree needs to be carefully chosen as a too high abstraction might limit specific DBMS features and distort results. Therefore, the abstraction interface should be aligned with the specified workload scenarios [6].

Client Orchestration (CO) enables automated evaluation runs. Therefore, the framework should provide tools that orchestrate evaluations, *i.e.* provision (cloud) resources, create, execute and collect the results and clean-up the clients. Hence, CO eases the creation of arbitrary load patterns and the simulation of multi-tenant workload patterns.

Database Orchestration (DO) enables the management of the DDBMSs to facilitate repetitive evaluation for different resources, configurations and the adaptation of the DDBMS based on predefined conditions. Hence, the evaluation framework should provide tools to automatically orchestrate DDBMSs, *i.e.* provision resources, setup, configure and adapt generic DDBMSs.

Evaluation Framework	WD	Evaluation Tier RS	P	S	E	A	C
TPC-E	TW	(✓)	✓	✓	✗	✗	✗
YCSB [7]	SW	✗	✓	✓	✓	✗	✗
YCSB++ [18]	TW, BDW, SW	✗	✓	✓	✓	✗	✓
BG [3]	WOW	✗	✓	✓	✗	✗	✓
BigBench [12]	TW, BDW	✗	✓	✗	✗	✗	✗
OLTP-bench [9]	WOW, SW	✗	✓	✓	✗	✗	✗
YCSB-T [8]	TW, SW	✗	✓	✓	✓	✗	✓
LinkBench [2]	WOW	✗	✓	✗	✗	✗	✗

Table 1. Distributed database evaluation tiers

Multi-phase Workloads (MpW) define the support of multiple workloads that run in parallel This is crucial to execute advanced evaluation scenarios. Further, the specification of the load development over a certain time frame per workload is required to simulate real world scenarios.

Extensibility (E) defines the need to provide an architecture, which eases the extension of the framework capabilities, *e.g.* by adding support for additional DBMSs or workload types.

5 Analysis of Evaluation Frameworks

In this section we analyse DB-EFs, which focus on DDBMSs. Hereby, we consider only DB-EFs, which have been published within the evolvement of DDBMSs, *i.e.* from 2007 on. In addition, we only consider the original evaluation frameworks and no minor extensions or evaluations based on these frameworks.

First, we analyse each framework based on the workload domain and supported evaluation tiers. The results are shown in Table 1. Second, we analyse each frameworks capabilities against the presented DB-EF requirements from Section 4. The results are shown in Table 2. The analysis applies ✗*= not supported*, *(*✓*)= partially supported*, ✓*= supported*. A detailed analysis can be found in an accompanying technical report [21].

The first insight of our analysis is that performance, scalability, elasticity and consistency tiers are well covered, but the resource selection and availability tier lack support (*cf.* Section 5.2). The second insight points out that the traditional DB-EF requirements [14] such as usability, distribution and extensibility are well supported, while monitoring and cloud orchestration are not (*cf.* Section 5.2).

5.1 Results for Evaluation Tiers

The resulting table (*cf.* Table 1) shows that the early DB-EFs focus on performance, scalability and elasticity, while newer frameworks focus as well on consistency. Hereby, only the performance tier has established common rating

Evaluation Framework	Evaluation Framework Requirement								
	U	**D/S**	**MP**	**MO**	**DA**	**CO**	**DO**	**MpW**	**E**
TPC-E	✓	✓	(✓)	✗	(✓)	✗	✗	✓	✓
YCSB [7]	✓	✓	(✓)	✗	✓	✗	✗	✗	✓
YCSB++ [18]	(✓)	✓	✓	✓	(✓)	✓	✗	✓	✗
BG [3]	✓	✓	✓	✓	✓	✗	✗	✓	✓
BigBench [12]	✓	✓	✗	✗	(✓)	✗	✗	✗	✓
OLTP-bench [9]	(✓)	✓	(✓)	✓	(✓)	✓	✗	✓	✓
YCSB+T [8]	(✓)	✓	(✓)	✗	✓	✗	✗	✗	✓
LinkBench [2]	✓	(✓)	(✓)	✓	(✓)	✗	✗	✓	✓

Table 2. Distributed database evaluation tiers

indices such as throughput, latency or SLA based rating [3]. While multiple frameworks target the scalability and elasticity tier, a common methodology and rating index has not yet been established. Yet, the need for a common evaluation methodology [22] and rating index [10] is already carved out.

Currently not supported evaluation tiers are resource selection and availability. While resource selection is partially supported by TPC-E, which considers physical hardware configurations, cloud-centric resource selection is not in the scope of any of the frameworks. With the increasing heterogeneity of cloud resources from diverse virtual machines offering to even container based resources, the consideration of cloud-centric resource selection needs to move into the focus of novel DB-EFs. Yet, existing DB-EFs can be applied to evaluate DDBMS running on heterogeneous cloud resources, but the DB-EFs do not offer an explicit integration with cloud resource offerings. Hence, manual resource management, monitoring and client/DDBMS orchestration hinders cloud-centric evaluations.

As availability is a major feature of DDBMS it is surprising that it is not considered by the analysed DB-EFs. Especially, as cloud resources do fail, availability concepts for applications running on cloud resources is widely discussed topic. Again, the support of DDBMSs orchestration can enable database specific availability evaluations.

5.2 Results for Evaluation Framework Requirements

The analysis of the evaluation framework requirements (*cf.* Table 2) shows that usability, scalability, database abstraction and extensibility are covered by all frameworks. Measurement processing is covered as well but only a few frameworks support advanced features such as visualisation and none of the frameworks supports advanced storage solutions such as TSDBs. Multi-phase workloads are partially covered by the frameworks, especially by the frameworks from the TW and WOW domains. The monitoring of client resources is partially covered, but only OLTP-bench considers resource monitoring. While all frameworks support the distributed execution of evaluations, only two support the orchestration of clients, which complicates the distributed evaluation runs. Further,

none of the frameworks supports DBMS orchestration. This fact leads to high complexity only for setting up the evaluation environment, especially when it comes to heterogeneous cloud resources. Further, dynamic DDBMS transitions for evaluating tiers such as elasticity or availability, always require custom implementations, which impedes the comparability and validity of the results.

6 Conclusion and Future Work

In the last decade the landscape of distributed database systems has evolved and NoSQL and NewSQL database systems appeared. In parallel, cloud computing enabled novel deployment option for database systems. Yet, these evolvements raise the complexity in selecting and deploying an appropriate database system.

In order to ease such decisions, several evaluation frameworks for distributed databases have been developed. In this paper, we presented an analysis of distributed database evaluation frameworks based on evaluation tiers and requirements towards the frameworks itself. The analysis is applied to eight evaluation frameworks and provides a thorough analysis of their evaluation tiers and capabilities. The results of this analysis shows that the performance, scalability, elasticity, and consistency tiers are well covered, while resource selection and availability are not considered by existing evaluation frameworks. With respect to the framework requirements, traditional requirements are covered [14], while cloud-centric requirements such as orchestration are only partially supported.

The analysis shows, that existing frameworks can be applied to evaluate distributed databases in the cloud, but there are still unresolved issues on the evaluation tier side, *i.e.* the support for resource selection and availability evaluation, and on the framework requirement side, *i.e.* the orchestration of clients and databases and exploitation of advanced storage solutions. This hinders repeatability [14] of evaluations on heterogeneous cloud resources as well as dynamic transition in the cluster. Yet, cloud computing research already offers approaches to enable automated resource provisioning and application orchestration in the cloud based on *Cloud Orchestration Tools (COTs)* [4]. Integrating COT into evaluation frameworks can be an option to ease the distributed execution of evaluation runs as well as orchestrating database clusters across different cloud resources. As COTs provide monitoring and adaptation capabilities, they can ease the evaluation of dynamic cluster transitions by defining advanced evaluation scenarios with dynamic database cluster adaptations.

Future work will comprise the analysis of COTs with respect to their exploitation in database evaluation frameworks. In addition, the design and implementation of a cloud-centric database evaluation framework is ongoing.

Acknowledgements The research leading to these results has received funding from the EC's Framework Programme HORIZON 2020 under grant agreement number 644690 (CloudSocket) and 731664 (MELODIC). We thank Moritz Keppler and the Daimler TSS for their valuable and constructive discussions.

References

1. Agrawal, D., El Abbadi, A., Das, S., Elmore, A.J.: Database scalability, elasticity, and autonomy in the cloud. In: DASFAA
2. Armstrong, T.G., Ponnekanti, V., Borthakur, D., Callaghan, M.: Linkbench: a database benchmark based on the facebook social graph. In: SIGMOD
3. Barahmand, S., Ghandeharizadeh, S.: Bg: A benchmark to evaluate interactive social networking actions. In: CIDR
4. Baur, D., Seybold, D., Griesinger, F., Tsitsipas, A., Hauser, C.B., Domaschka, J.: Cloud orchestration features: Are tools fit for purpose? In: UCC
5. Bermbach, D., Kuhlenkamp, J.: Consistency in distributed storage systems. In: Networked Systems
6. Bermbach, D., Kuhlenkamp, J., Dey, A., Sakr, S., Nambiar, R.: Towards an extensible middleware for database benchmarking. In: TPCTC
7. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with ycsb. In: SoCC
8. Dey, A., Fekete, A., Nambiar, R., Rohm, U.: Ycsb+t: Benchmarking web-scale transactional databases. In: ICDEW
9. Difallah, D.E., Pavlo, A., Curino, C., Cudre-Mauroux, P.: Oltp-bench: An extensible testbed for benchmarking relational databases. VLDB
10. Dory, T., Mejias, B., Roy, P., Tran, N.L.: Measuring elasticity for cloud databases. In: CLOUD COMPUTING
11. Friedrich, S., Wingerath, W., Gessert, F., Ritter, N., Pldereder, E., Grunske, L., Schneider, E., Ull, D.: Nosql oltp benchmarking: A survey. In: GI-Jahrestagung
12. Ghazal, A., Rabl, T., Hu, M., Raab, F., Poess, M., Crolotte, A., Jacobsen, H.A.: Bigbench: towards an industry standard benchmark for big data analytics. In: SIGMOD
13. Gilbert, S., Lynch, N.: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. Acm Sigact News
14. Gray, J.: Benchmark handbook: for database and transaction processing systems. Morgan Kaufmann Publishers Inc.
15. Grolinger, K., Higashino, W.A., Tiwari, A., Capretz, M.A.: Data management in cloud environments: Nosql and newsql data stores. JoCCASA
16. Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., Gaikwad, P., Litoiu, M.: How do i choose the right nosql solution? a comprehensive theoretical and experimental survey. BDIA
17. Mell, P., Grance, T.: The nist definition of cloud computing. Tech. rep., National Institute of Standards & Technology (2011)
18. Patil, S., Polte, M., Ren, K., Tantisiriroj, W., Xiao, L., López, J., Gibson, G., Fuchs, A., Rinaldi, B.: Ycsb++: benchmarking and performance debugging advanced features in scalable table stores. In: SoCC
19. Reniers, V., Van Landuyt, D., Rafique, A., Joosen, W.: On the state of nosql benchmarks. In: ICPE
20. Sadalage, P.J., Fowler, M.: NoSQL distilled: a brief guide to the emerging world of polyglot persistence. Pearson Education (2012)
21. Seybold, D., Domaschka, J.: A cloud-centric survey on distributed database evaluation. Tech. rep., Ulm University
22. Seybold, D., Wagner, N., Erb, B., Domaschka, J.: Is elasticity of scalable databases a myth? In: IEEE Big Data