Multi-Tenant Cloud Data Services: State-of-the-Art, Challenges and Opportunities

Vivek Narasayya
Microsoft Research

Surajit Chaudhuri
Microsoft Research

ABSTRACT

Enterprises are moving their business-critical workloads to public clouds at an accelerating pace. Multi-tenancy is a crucial tenet for cloud data service providers allowing them to provide services in cost-effective manner by sharing of resources among tenants of the service. In this tutorial we review architectures of today's cloud data services and identify trends and challenges that arise in multi-tenant cloud data services. We discuss techniques that have been developed for enabling elasticity, providing SLAs, ensuring performance isolation, and reducing cost. We conclude with open research problems in cloud data services.

CCS CONCEPTS

• **Information systems → DBMS engine architectures**; **Database query processing**; **Record and buffer management**; **Autonomous database administration**.

KEYWORDS

Cloud data service, multi-tenancy, disaggregation, serverless, resource management

ACM Reference Format:
Vivek Narasayya and Surajit Chaudhuri. 2022. Multi-Tenant Cloud Data Services: State-of-the-Art, Challenges and Opportunities. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3514221.3522566

1 INTRODUCTION

The worldwide public cloud database services market is growing rapidly. The wave of cloud adoption is being driven by digital transformation projects within enterprises that are migrating their applications to the cloud. Major cloud vendors that offer public cloud database services include Alibaba Cloud, Amazon Web Services (AWS), Google Cloud Platform, IBM Cloud, Microsoft Azure, and Oracle Cloud. Cloud database services offer improved manageability compared to on-premises database systems by taking away the burden of managing hardware, virtual machines (VMs), and storage. They handle tasks such as patching and upgrades of the operating system and DBMS, ensuring high availability and disaster recovery, automated backups, point-in-time recovery and even some facets of performance tuning [129].

An overview of the classes of database services available in the cloud are shown in Figure 1[1]. Relational online transaction processing (OLTP) services, e.g., Amazon Aurora [158], Azure SQL Hyperscale [17], and Google Cloud Spanner [32] enable enterprises to run their operational workloads. In addition to traditional SQL-based OLTP services, a new class of operational cloud data services with key-value, document and graph data models has become popular. Examples of such NoSQL services are Google BigTable [37], Amazon Dynamo DB [61], Azure Cosmos DB [1], MongoDB Atlas [127] and Apache Cassandra. Data analytics services (see [39] for a survey) help enterprises to transform and shape data from operational and external sources for use in subsequent analysis, as well as run batch and interactive SQL queries. Examples of cloud services that support Extract-Transform-Load (ETL), interactive and batch SQL queries include AWS Redshift [26], Azure Synapse [28], Google BigQuery [78], Snowflake [52] as well as Spark-based [175] services such as Databricks. Finally, another class of online analytic processing (OLAP) services support ad-hoc interactive analysis over multi-dimensional data, and stream processing services enable near real-time data analytics on operational data.

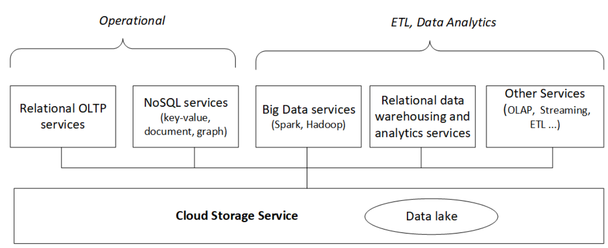


Figure 1: Classes of cloud data services

Three major factors are leading to the re-architecting and redesign of databases in the cloud: *disaggregation*, *multi-tenancy* and *serverless*. Disaggregation has been driven by the need to scale compute and storage independently of each other. Designing cloud databases with independent compute and storage tiers on modern data center architectures raises new challenges. Second, in the on-premises setting the customer owns all hardware resources (servers, networks, storage). However, in the cloud, multiple databases from different customers (a.k.a. *tenants*), share computing resources of the cloud provider including CPU, memory and disk (SSD) resources as well as the data center network. Multi-tenancy is crucial since dedicating hardware for each tenant is simply not cost effective, and sharing resources across tenants enables cloud providers to

[1]Figures in this tutorial have been adopted from [129].

ACM ISBN 978-1-4503-9249-5/22/06…$15.00
https://doi.org/10.1145/3514221.3522566

sharply lower the cost of providing the service. Finally, customers traditionally need to provision a database in the cloud by choosing a fixed set of resources up front; and they pay for those resources regardless of actual usage. Recently, *serverless* databases, which offer a pay-for-use consumption model have gained traction. Tenants no longer need to provision resources for peak usage ahead of time. Rather, the cloud provider acquires and releases resources in response to demands of the database workload, and tenants only pay for resources that their workloads actually use. Thus, today's cloud vendors not only face the challenges of ensuring the traditional demands of enterprise databases such as performance, high-availability, and durability, they must also meet new requirements demanded by serverless, while adhering to the foundational cloud tenets of disaggregation and multi-tenancy.

Scope and goals of the tutorial: In this tutorial we describe key architectures and techniques developed both in industry and research to address the above challenges, and present open problems in the area. We discuss virtualization technology for achieving multi-tenancy and its impact on isolation and consolidation, quality of service (i.e., SLOs and SLAs) and pricing models provided by cloud services, techniques for multi-tenant resource management and improving efficiency, and the new challenges raised by the serverless database paradigm. A detailed outline of the tutorial and the topics covered therein are described in Section 2. Since challenges and open problems are a major focus of the tutorial, in Section 3 we call out the open problems we plan to discuss. This tutorial is based on the survey article [129].

Target Audience and Duration: The intended audience is both academic researchers as well as engineers in the database industry. The tutorial will be largely self-contained for a database audience, and we will include introductory material for attendees not familiar with public cloud architectures.

2 TOPICS COVERED IN THE TUTORIAL

2.1 Architectures

The architectures of cloud data services have been deeply impacted by the design of data centers. Data centers are comprised of a large number of clusters of commodity machines connected to each other over fast, high-bandwidth networks [3], and organized into availability zones and regions. The need for independently scaling compute and storage has led to disaggregated architectures for all types of cloud data services: OLTP, data analytic and NoSQL. For OLTP services such as [2, 32, 158], we describe how their architectures have evolved to achieve high-availability, durability, performance and its impact on cost. We discuss and compare architectures of data analytic services [26, 28, 52, 78, 175] along key dimensions such as shared nothing vs. shared data, pre-loaded data vs. in-situ querying, interactive vs. batch querying capabilities, and the sophistication of query optimization. Finally, we review architectures of NoSQL services such as [1, 37, 61, 127] and compare their data models and consistency vs. performance trade-offs.

2.2 Multi-Tenancy

Multi-tenancy is a key tenet for public cloud data service providers since it allows consolidation of tenants (i.e. databases) by sharing resources, which reduces cost. In general, the greater the degree of consolidation – i.e. the number of databases that are hosted on a single server or cluster – the greater is the reduction in cost. In order to achieve good performance and security for tenants while achieving a high degree of consolidation, cloud service providers need to *virtualize* the available hardware resources. The strength of performance and security isolation and the degree of consolidation achieved depend primarily on the virtualization technology used, and are usually at odds with one another.

In the context of cloud databases, a wide variety of virtualization technology are used – see Figure 2. These include virtual machines (VMs) [141], operating system constructs, e.g. CGroups in Linux and Job Objects in Windows, containers (e.g. Docker) or even logical constructs implemented inside the application or database system (DBMS) for isolating tenants [133, 166]. The choice of virtualization technology is crucial since it influences the following key factors:

- Degree of consolidation: The higher up in the stack virtualization is supported, the greater the degree of consolidation achievable, leading to lower cost for the service provider.
- Degree of isolation: The lower down in the stack the virtualization logic is supported, the greater is the security and performance isolation achievable across tenants.
- Ease of provisioning: The time taken to create a new database or upsize/downsize resource limits for a database is important for many applications. Such provisioning is simpler in logical virtualization technology implemented higher up in the stack.
- Impact of failures: Cloud database services experience planned and unplanned failures. For example, DBMS, operating system and hypervisor software need to be patched and upgraded from time-to-time. Sometimes there are unplanned failures due to failed hardware (e.g. disk) or software (e.g. bug in DBMS code). Depending on the virtualization technology used, a single failure may affect only one tenant or many tenants.

We review each of these virtualization technologies and comment on their effectiveness for each of the above key factors.

2.3 Quality of Service and Pricing Models

In on-premises scenarios, customers provision their own hardware and have full control over the resources of their machines and enterprise network. This control enables them to provide certain quality of service (QoS) to users. These QoS guarantees are also referred to as Service level agreements (SLAs) with customers or internal Service Level Objectives (SLOs) [102] which are not made visible to customers. In contrast, in multi-tenant cloud data services, such QoS guarantees are not automatic since the cloud provider shares the resources of a machine and the data center network with multiple tenants.

Some cloud data services attempt to provide SLAs or SLOs. One common guarantee provided by most cloud data services is on *availability* of the service. End-to-end performance SLAs in terms of throughput and latency are typically not provided by cloud data

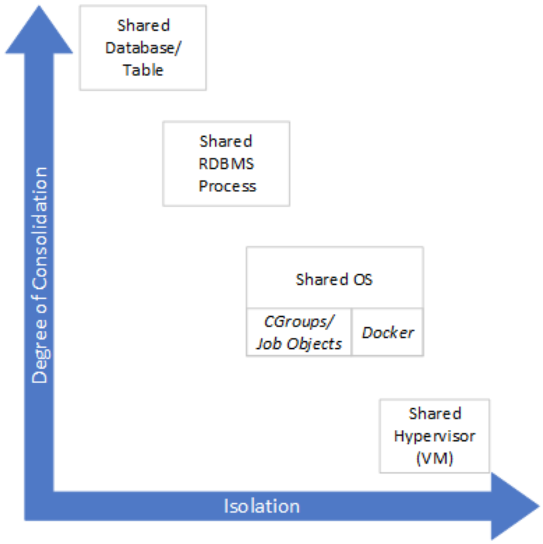
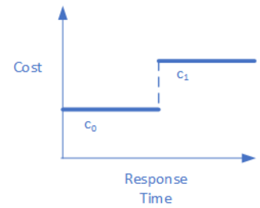
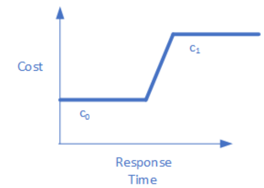


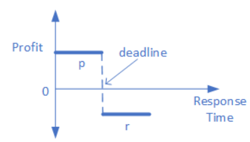
Figure 2: Virtualization technology in databases. Trade-offs in Consolidation vs. Isolation.



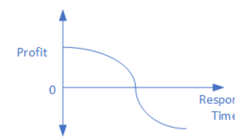
(a) Step function SLA



(b) Piece-wise linear SLA



(c) SLA that models deadlines



(d) SLA that models continuous drop in value over time

Figure 3: Performance SLAs

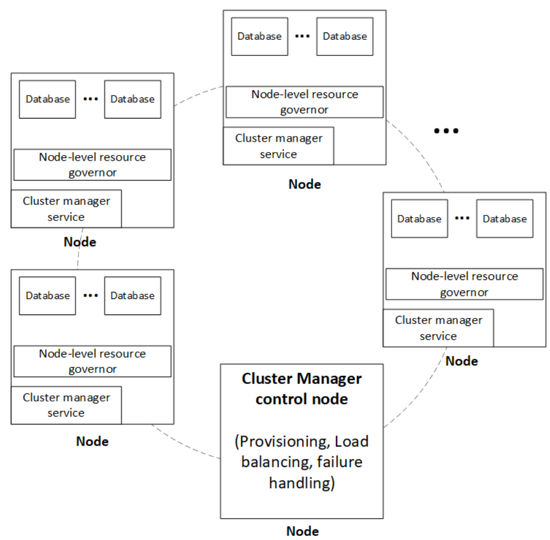


Figure 4: Resource management infrastructure

services. Key-value stores offer a much more limited API and do provide SLAs on latency and throughput. However, services with richer APIs such as SQL only provide guarantees of a minimum amount of resources available for the tenant, and not on latency or throughput.

In general, guaranteeing a higher quality of service requires the service provider to reserve more resources for the workload, and hence incurs a higher cost due to unutilized resources. Therefore, service providers use pricing models such as spot pricing and pre-emptible instances to incentivize customers to use cheaper services with weak or no quality of service guarantees, thereby allowing them to sell capacity that would otherwise go unutilized.

In this tutorial, we review some of the key factors that give rise to variations in quality of service and pricing. We also review models of SLAs and pricing in today's commercial cloud data services as well as other models proposed in the research literature (e.g., some examples of SLAs we will cover are shown in Figure 3 [43, 44, 56, 130, 170, 171]).

2.4 Multi-Tenant Resource Management

Effective management of resources plays a critical role in achieving high availability, good performance, scalability, and efficiency in a multi-tenant cloud data service. Servers in a data center are organized into *clusters*, which are a logical collection of nodes (i.e., machines). Figure 4 shows the typical infrastructure for resource management in a cloud database, where the tasks of managing resources is divided between two infrastructure components: the *cluster manager* and *node-level resource governor*.

The cluster manager (e.g., Kubernetes [89], Service Fabric [103]), is a distributed system infrastructure that is responsible for managing the collection of nodes in the cluster including key decisions such as placement of databases onto nodes, handling failures of nodes and databases, and balancing utilization across nodes in the cluster by moving databases while ensuring minimal impact on availability and performance e.g., [46, 57, 68, 163]. In data warehousing and Big Data analytic systems, when a query or job executes, decisions need to be made on how many nodes to execute the query on and with what amount of resources; and the necessary resources must then be acquired. We review such cluster-level resource management techniques for data analytics briefly [5, 49, 81].

On each node, when multiple databases execute and share resources such as CPU, I/O and memory on the node, a node-level resource governor ensures that databases do not suffer from *noisy neighbor* problems that can adversely affect SLOs. We will review mechanisms that have been developed for *performance isolation* in cloud data services [56, 85, 130, 131, 164].

In the techniques described above it is important to have good estimates of the resources required for a database or query/job.

We review the body of work on *resource estimation* for database workloads. These solutions leverage different techniques such as analytical models that use knowledge of database query processing [6, 41, 118], statistical techniques [168, 169, 177], machine learning, [7, 64, 72, 139], and combinations of the above [55, 115].

Finally, we review techniques that can help improve efficiency and thereby reduce cost both for cloud providers. In particular, we discuss techniques for more efficient use of resources in multi-tenant environments [33, 85, 131, 154, 162] which enables higher packing density of databases and reduced capital expenditure [56, 113, 130].

2.5 Serverless Databases

In the past few years, there has been increasing interest from industry and academia in *serverless computing*. The promise of serverless computing is compelling for cloud database customers. First, it reduces the burden of provisioning resources, since a serverless database can leverage the elasticity of the cloud to handle variable loads by automatically acquiring and releasing the required resources on-demand. Second, unlike in the traditional provisioned model, in a serverless consumption model, the customer only pays for resources that they use. This model is similar to a utility such as electricity where customers are billed only for power they actually consume. An example of such a pricing model in Microsoft Azure SQL Serverless Database is shown in Figure 5. When the database is in an active state, customers pay for a (small) minimum amount of resources or the actual resource usage, whichever is higher, (shown by the green shaded area). Customers pay nothing for compute when the database is in a paused state.

Serverless databases are particularly well suited for use cases where the resources needed to run the workload can vary substantially over time or cannot be accurately estimated in advance. One example is a line-of-business OLTP application in an enterprise with sporadic and bursty workload over the course of a week. Another scenario for data analytics is a resource-intensive query that would be too slow to execute with the fixed amount of resources that were provisioned up front.

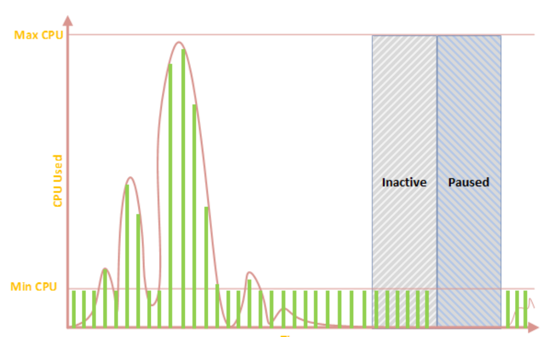


Figure 5: Example pricing model in a serverless database

We will highlight unique challenges for serverless databases, review examples of two commercial database services (one OLTP service and one data analytic service) and describe techniques developed in industry [5, 10, 11, 30, 78] and research [87, 101, 149] for serverless databases.

3 CHALLENGES AND OPEN PROBLEMS

Describing challenges and open problems facing multi-tenant cloud data services is a major focus of the tutorial. Below, we provide some examples of problem areas as well as specific problems we highlight in the tutorial.

3.1 Disaggregation

Separation of compute and storage has been a major driver of architectural changes in cloud data services. One area of active research in the systems and database communities is whether *memory disaggregation* can bring similar benefits by independently scaling CPU and memory. We also see *functional* disaggregation by components. For example, some of today's cloud data analytic services (e.g., Snowflake) use a multi-tenant metadata service that is separate from the query processing service. Similarly, cloud OLTP services (e.g. Azure SQL Hyperscale) employ caching and logging services that are separate from the compute and storage. Studying the benefits of such functional disaggregation for cloud data services has the potential to influence future architectures.

3.2 Caching

Services in all three workload categories: OLTP, Data analytics and NoSQL have developed architectures where the compute tier and the storage tier are separated. While such separation makes it much easier to scale compute resources independently, in response to workload demand, and thereby reduce cost, it simultaneously degrades performance. Hence cloud data services resort to aggressive caching strategies in the compute tier (or resort to adding intermediate caching tiers) to regain performance. Such caching, in turn, raises the cost of the service. Therefore, the one major challenge in caching is to effectively address the performance-cost trade-off. Furthermore, traditional caching techniques need to be made multi-tenant aware in order to be cost-effective.

The lack of our ability to move critical database state – particularly caches – efficiently across machines in a data center hurts user experience, for example, resulting in cold restarts. If database state can be migrated quickly and automatically, it makes it possible for cloud data services to improve performance, increase efficiency and lower cost. Hence improvements in this technology, including cache migration, and database live migration are important areas of work.

3.3 Optimizing Use of Cloud Services

We are seeing a proliferation of choices available to customers who want to run their applications on the cloud. There are different classes of services with different performance, availability and pricing characteristics. For example, provisioned vs. serverless databases vs. spot instances. Therefore it becomes important to help enterprises make decisions as to which offering to choose for their workloads. Software-as-a-service offerings for CRM and ERP (e.g. Salesforce, Microsoft Dynamics) are some of the largest users of cloud database services. The benefits of optimizing usage of cloud databases by choosing the appropriate class of database services for their workloads are therefore amplified for SaaS application vendors.

There are also opportunities for intermediary platforms that layer on top of cloud data services and expose value-added functionality. One example is the ability to use a collection of spot/preemtible compute to create a reliable enough but cheaper compute for batch data analytic jobs (e.g., [92]).

3.4 Auto Tuning

While auto-tuning databases is crucial in on-premises and cloud databases, there are potential opportunities that are uniquely enabled in the cloud. For example, cloud data service vendors have access to telemetry of many different databases and workloads of their customers. There is an opportunity to leverage this rich history, without compromising data privacy and compliance constraints, to aid in auto-tuning their services. An instance of such a problem is calibrating the cost model used by the query optimizer based on the specific machine (SKU) that the database runs on. The availability of large scale telemetry of optimizer estimates and actual execution costs from multiple databases running on various hardware SKUs in the data center can be a valuable asset that is not available in a traditional on-premises setting. Another aspect of the cloud is the relatively easy access to extra resources (at a certain cost). In the auto-tuning context, it is possible to leverage such resources to perform controlled A-B testing for a new database configuration, e.g, to evaluate if the new configuration indeed translates to better performance – without affecting performance of the production database.

3.5 Impact of new hardware

Data centers these days already have new hardware such as network based offload technology (e.g., RDMA), storage technologies such as SSD (now mainstream), FPGAs, GPUs, and non-volatile memory (NVRAM). These technologies have the potential for significant disruption and acceleration in the software stack. For instance, RDMA holds the promise of low-latency access to remote memory in a cluster, thereby accelerating query processing and improving efficiency by sharing resources across nodes more easily.

Due to improvements in data center networking, the ratio of latency of I/O to remote storage or caching tiers vs. local SSDs has been reducing, thereby favoring architectures where compute and storage tiers are separated. However, if new hardware technologies such as NVRAM (which are significantly lower latency compared to SSDs) gain traction for use in databases, then this ratio might once again increase. Understanding the impact of such technology on cloud data services architecture is an open area of study.

3.6 Resource Estimation

Despite a large body of research work on resource estimation, adoption in practice has been limited, in part due to lack of robustness of the techniques. Cloud database providers can collect telemetry on resource usage of many databases and even individual queries. Developing robust models for resource estimation of queries and databases by leveraging such telemetry, particularly for ad-hoc queries and fast-changing or unseen datasets, will be crucial for effective resource management by cloud providers.

4 BIOGRAPHICAL SKETCHES

Vivek Narasayya is a Partner Research Manager in the Data Systems group at Microsoft Research. His areas of interest include infrastructure for cloud data services, physical database design, query optimization and data preparation. He is the recipient of a VLDB 10 year Best paper award. In collaboration with his colleagues at Microsoft Research, he has helped incorporate resource management technologies into Azure Data, and index tuning and data cleaning capabilities into Microsoft SQL Server. Vivek received his Ph.D. from the University of Washington in 2000.

Surajit Chaudhuri is a Distinguished Scientist at Microsoft Research and leads the Data Systems group. His current areas of interest are enterprise data analytics, data discovery, self-manageability and cloud database services. Working with his colleagues in Microsoft Research, he helped incorporate the Index Tuning Wizard (and subsequently Database Engine Tuning Advisor) and data cleaning technology into Microsoft SQL Server. Surajit is an ACM Fellow, a recipient of the ACM SIGMOD Edgar F. Codd Innovations Award, ACM SIGMOD Contributions Award, a VLDB 10 year Best Paper Award, and an IEEE Data Engineering Influential Paper Award. Surajit received his Ph.D. from Stanford University in 1992.

REFERENCES

[1] A Technical Overview of Azure Cosmos DB. 2020. https://azure.microsoft.com/en-us/blog/a-technical-overview-of-azure-cosmos-db/. Accessed 20 January 2021.

[2] Daniel Abadi. 2012. Consistency tradeoffs in modern distributed database system design: CAP is only part of the story. *Computer* 45, 2 (2012), 37–42.

[3] Daniel Abadi, Anastasia Ailamaki, David Andersen, Peter Bailis, Magdalena Balazinska, Philip Bernstein, Peter Boncz, Surajit Chaudhuri, Alvin Cheung, AnHai Doan, et al. 2020. The Seattle Report on Database Research. *ACM SIGMOD Record* 48, 4 (2020), 44–53.

[4] Sanjay Agrawal, Surajit Chaudhuri, and Vivek R Narasayya. 2000. Automated selection of materialized views and indexes in SQL databases. In *VLDB*, Vol. 2000. 496–505.

[5] Josep Aguilar-Saborit, Raghu Ramakrishnan, Krish Srinivasan, Kevin Bocksrocker, Yannis Ioalagia, Mahadevan Sankara, and Moe Shafiei. 20120. POLARIS: The Distributed SQL Engine in Azure Synapse. *Proceedings of the VLDB Endowment* (20120).

[6] Mumtaz Ahmad, Ashraf Aboulnaga, Shivnath Babu, and Kamesh Munagala. 2008. Modeling and exploiting query interactions in database systems. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 183–192.

[7] Mert Akdere, Ugur Çetintemel, Matteo Riondato, Eli Upfal, and Stanley B Zdonik. 2012. Learning-based query performance modeling and prediction. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 390–401.

[8] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. 2008. A scalable, commodity data center network architecture. *ACM SIGCOMM computer communication review* 38, 4 (2008), 63–74.

[9] Mohammad Alizadeh and Tom Edsall. 2013. On the data path performance of leaf-spine datacenter fabrics. In *2013 IEEE 21st annual symposium on high-performance interconnects*. IEEE, 71–74.

[10] Amazon Athena. 2020. https://aws.amazon.com/athena/. Accessed 20 January 2021.

[11] Amazon Aurora Serverless. 2020. https://aws.amazon.com/rds/aurora/serverless/. Accessed 20 January 2021.

[12] Amazon Dynamo DB On-Demand. 2020. https://aws.amazon.com/about-aws/whats-new/2018/11/announcing-amazon-dynamodb-on-demand/. Accessed 20 January 2021.

[13] Amazon Firecracker. 2020. https://aws.amazon.com/about-aws/whats-new/2018/11/firecracker-lightweight-virtualization-for-serverless-computing/. Accessed 20 January 2021.

[14] Amazon RDS Multi-AZ. 2020. https://aws.amazon.com/rds/features/multi-az/. Accessed 20 January 2021.

[15] An Updated Performance Comparison of Virtual Machines and Linux Containers. 2014. https://dominoweb.draco.res.ibm.com/reports/rc25482.pdf. Accessed 20 January 2021.

[16] Ganesh Ananthanarayanan, Christopher Douglas, Raghu Ramakrishnan, Sriram Rao, and Ion Stoica. 2012. True elasticity in multi-tenant data-intensive compute clusters. In *Proceedings of the Third ACM Symposium on Cloud Computing*. ACM, 24.

[17] Panagiotis Antonopoulos, Alex Budovski, Cristian Diaconu, Alejandro Hernandez Saenz, Jack Hu, Hanuma Kodavalla, Donald Kossmann, Sandeep Lingam, Umar Farooq Minhas, Naveen Prakash, et al. 2019. Socrates: The New SQL Server in the Cloud. In *Proceedings of the 2019 International Conference on Management of Data*. ACM, 1743–1756.

[18] Apache Hadoop. 2020. http://hadoop.apache.org. Accessed 20 January 2021.

[19] Raja Appuswamy, Goetz Graefe, Renata Borovica-Gajic, and Anastasia Ailamaki. 2019. The five-minute rule 30 years later and its impact on the storage hierarchy. *Commun. ACM* 62, 11 (2019), 114–120.

[20] Arvind Arasu, Spyros Blanas, Ken Eguro, Raghav Kaushik, Donald Kossmann, Ravishankar Ramamurthy, and Ramarathnam Venkatesan. 2013. Orthogonal Security with Cipherbase.. In *CIDR*.

[21] Michael Armbrust, Tathagata Das, Liwen Sun, Burak Yavuz, Shixiong Zhu, Mukul Murthy, Joseph Torres, Herman van Hovell, Adrian Ionescu, Alicja Łuszczak, et al. 2020. Delta lake: high-performance ACID table storage over cloud object stores. *Proceedings of the VLDB Endowment* 13, 12 (2020), 3411–3424.

[22] Michael Armbrust, Reynold S Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K Bradley, Xiangrui Meng, Tomer Kaftan, Michael J Franklin, Ali Ghodsi, et al. 2015. Spark sql: Relational data processing in spark. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*. 1383–1394.

[23] AtScale. 2020. https://www.atScale.com/. Accessed 20 January 2021.

[24] Automatic Plan Correction. 2020. https://docs.microsoft.com/en-us/sql/relational-databases/automatic-tuning/automatic-tuning. Accessed 20 January 2021.

[25] AWS Nitro System. 2020. https://aws.amazon.com/ec2/nitro/. Accessed 20 January 2021.

[26] AWS Redshift. 2020. https://aws.amazon.com/redshift/. Accessed 20 January 2021.

[27] Azure Cosmos DB Serverless. 2020. https://azure.microsoft.com/en-us/blog/build-apps-of-any-size-or-scale-with-azure-cosmos-db/. Accessed 20 January 2021.

[28] Azure SQL Data Warehouse. 2020. https://azure.microsoft.com/en-us/services/sql-data-warehouse/. Accessed 20 January 2021.

[29] Azure SQL DB Automatic Tuning. 2020. https://docs.microsoft.com/en-us/sql/relational-databases/automatic-tuning/automatic-tuning. Accessed 20 January 2021.

[30] Azure SQL DB Serverless. 2020. https://docs.microsoft.com/en-us/azure/sql-database/sql-database-serverless. Accessed 20 January 2021.

[31] Azure Synapse Analytics. 2020. https://docs.microsoft.com/en-us/azure/synapse-analytics/overview-what-is. Accessed 20 January 2021.

[32] David F Bacon, Nathan Bales, Nico Bruno, Brian F Cooper, Adam Dickinson, Andrew Fikes, Campbell Fraser, Andrey Gubarev, Milind Joshi, Eugene Kogan, et al. 2017. Spanner: Becoming a sql system. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 331–343.

[33] Ishan Banerjee, Fei Guo, Kiran Tati, and Rajesh Venkatasubramanian. 2013. Memory overcommitment in the ESX server. *VMware Technical Journal* 2, 1 (2013), 2–12.

[34] Paul Barham, Rebecca Isaacs, Richard Mortier, and Dushyanth Narayanan. 2003. Magpie: Online Modelling and Performance-aware Systems.. In *HotOS*. 85–90.

[35] R. Bringhurst. 2012. *The Elements of Typographic Style* (fourth ed.). Hartley & Marks: Vancouver, BC.

[36] Brendan Burns, Brian Grant, David Oppenheimer, Eric Brewer, and John Wilkes. 2016. Borg, Omega, and Kubernetes. *Queue* 14, 1 (2016), 10.

[37] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. 2008. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)* 26, 2 (2008), 1–26.

[38] Surajit Chaudhuri. 1998. An overview of query optimization in relational systems. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*. ACM, 34–43.

[39] Surajit Chaudhuri, Umeshwar Dayal, and Vivek Narasayya. 2011. An overview of business intelligence technology. *Commun. ACM* 54, 8 (2011), 88–98.

[40] Surajit Chaudhuri and Vivek Narasayya. 2007. Self-tuning database systems: a decade of progress. In *Proceedings of the 33rd international conference on Very large data bases*. VLDB Endowment, 3–14.

[41] Surajit Chaudhuri, Vivek Narasayya, and Ravishankar Ramamurthy. 2004. Estimating progress of execution for SQL queries. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. ACM, 803–814.

[42] Surajit Chaudhuri and Vivek R Narasayya. 1997. An efficient, cost-driven index selection tool for Microsoft SQL server. In *VLDB*, Vol. 97. Citeseer, 146–155.

[43] Yun Chi, Hyun Jin Moon, and Hakan Hacigümüş. 2011. iCBS: incremental cost-based scheduling under piecewise linear SLAs. *Proceedings of the VLDB Endowment* 4, 9 (2011), 563–574.

[44] Yun Chi, Hyun Jin Moon, Hakan Hacigümüş, and Junichi Tatemura. 2011. SLA-tree: a framework for efficiently supporting SLA-based decisions in cloud computing. In *Proceedings of the 14th International Conference on Extending Database Technology*. ACM, 129–140.

[45] Kristina Chodorow. 2013. *MongoDB: the definitive guide: powerful and scalable data storage.* " O'Reilly Media, Inc.".

[46] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield. 2005. Live migration of virtual machines. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 273–286.

[47] Neil Conway, William R Marczak, Peter Alvaro, Joseph M Hellerstein, and David Maier. 2012. Logic and lattices for distributed programming. In *Proceedings of the Third ACM Symposium on Cloud Computing*. 1–14.

[48] James C Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, Jeffrey John Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, et al. 2013. Spanner: Google's globally distributed database. *ACM Transactions on Computer Systems (TOCS)* 31, 3 (2013), 8.

[49] Carlo Curino, Djellel E Difallah, Chris Douglas, Subru Krishnan, Raghu Ramakrishnan, and Sriram Rao. 2014. Reservation-based scheduling: If you're late don't blame us!. In *Proceedings of the ACM Symposium on Cloud Computing*. ACM, 1–14.

[50] Carlo Curino, Evan PC Jones, Samuel Madden, and Hari Balakrishnan. 2011. Workload-aware database monitoring and consolidation. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 313–324.

[51] Carlo Curino, Evan PC Jones, Raluca Ada Popa, Nirmesh Malviya, Eugene Wu, Sam Madden, Hari Balakrishnan, and Nickolai Zeldovich. 2011. Relational cloud: A database-as-a-service for the cloud. (2011).

[52] Benoit Dageville, Thierry Cruanes, Marcin Zukowski, Vadim Antonov, Artin Avanes, Jon Bock, Jonathan Claybaugh, Daniel Engovatov, Martin Hentschel, Jiansheng Huang, et al. 2016. The snowflake elastic data warehouse. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 215–226.

[53] William James Dally and Brian Patrick Towles. 2004. *Principles and practices of interconnection networks*. Elsevier.

[54] Sudipto Das, Miroslav Grbic, Igor Ilic, Isidora Jovandic, Andrija Jovanovic, Vivek R Narasayya, Miodrag Radulovic, Maja Stikic, Gaoxiang Xu, and Surajit Chaudhuri. 2019. Automatically indexing millions of databases in microsoft azure sql database. In *Proceedings of the 2019 International Conference on Management of Data*. ACM, 666–679.

[55] Sudipto Das, Feng Li, Vivek R Narasayya, and Arnd Christian Konig. 2016. Automated demand-driven resource scaling in relational database-as-a-service. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 1923–1934.

[56] Sudipto Das, Vivek R Narasayya, Feng Li, and Manoj Syamala. 2013. CPU sharing techniques for performance isolation in multi-tenant relational database-as-a-service. *Proceedings of the VLDB Endowment* 7, 1 (2013), 37–48.

[57] Sudipto Das, Shoji Nishimura, Divyakant Agrawal, and Amr El Abbadi. 2010. Live database migration for elasticity in a multitenant database for cloud platforms. *CS, UCSB, Santa Barbara, CA, USA, Tech. Rep* 9 (2010), 2010.

[58] Sudipto Das, Shoji Nishimura, Divyakant Agrawal, and Amr El Abbadi. 2011. Albatross: lightweight elasticity in shared storage databases for the cloud using live data migration. *Proceedings of the VLDB Endowment* 4, 8 (2011), 494–505.

[59] Jeffrey Dean and Luiz André Barroso. 2013. The tail at scale. *Commun. ACM* 56, 2 (2013), 74–80.

[60] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (2008), 107–113.

[61] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: amazon's highly available key-value store. In *ACM SIGOPS operating systems review*, Vol. 41. ACM, 205–220.

[62] Bailu Ding, Sudipto Das, Ryan Marcus, Wentao Wu, Surajit Chaudhuri, and Vivek R Narasayya. 2019. Ai meets ai: Leveraging query executions to improve index recommendations. In *Proceedings of the 2019 International Conference on Management of Data*. 1241–1258.

[63] Songyun Duan, Shivnath Babu, and Kamesh Munagala. 2009. Fa: A system for automating failure diagnosis. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 1012–1023.

[64] Jennie Duggan, Ugur Cetintemel, Olga Papaemmanouil, and Eli Upfal. 2011. Performance prediction for concurrent database workloads. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 337–348.

[65] Anshuman Dutt, Chi Wang, Vivek Narasayya, and Surajit Chaudhuri. 2020. Efficiently approximating selectivity functions using low overhead regression

models. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2215–2228.
[66] Anshuman Dutt, Chi Wang, Azade Nazi, Srikanth Kandula, Vivek Narasayya, and Surajit Chaudhuri. 2019. Selectivity estimation for range predicates using lightweight models. *Proceedings of the VLDB Endowment* 12, 9 (2019), 1044–1057.
[67] Elastic Pools in Azure SQL Database. 2020. https://docs.microsoft.com/en-us/azure/sql-database/sql-database-elastic-pool. Accessed 20 January 2021.
[68] Aaron J Elmore, Sudipto Das, Divyakant Agrawal, and Amr El Abbadi. 2011. Zephyr: live migration in shared nothing databases for elastic cloud platforms. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 301–312.
[69] Extended Events Overview. 2019. https://docs.microsoft.com/en-us/sql/relational-databases/extended-events/. Accessed 20 January 2021.
[70] Fauna DB. 2020. https://fauna.com. Accessed 20 January 2021.
[71] Rodrigo Fonseca, George Porter, Randy H Katz, and Scott Shenker. 2007. X-trace: A pervasive network tracing framework. In *4th {USENIX} Symposium on Networked Systems Design & Implementation ({NSDI} 07)*.
[72] Archana Ganapathi, Harumi Kuno, Umeshwar Dayal, Janet L Wiener, Armando Fox, Michael Jordan, and David Patterson. 2009. Predicting multiple metrics for queries: Better decisions enabled by machine learning. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 592–603.
[73] Anshul Gandhi, Mor Harchol-Balter, Ram Raghunathan, and Michael A Kozuch. 2012. Autoscale: Dynamic, robust capacity management for multi-tier data centers. *ACM Transactions on Computer Systems (TOCS)* 30, 4 (2012), 14.
[74] Anshul Gandhi, Sidhartha Thota, Parijat Dube, Andrzej Kochut, and Li Zhang. 2016. Autoscaling for hadoop clusters. In *2016 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 109–118.
[75] Gartner DBMS Future. 2019. https://www.gartner.com/document/3941821. Accessed 20 January 2021.
[76] Rahul Ghosh and Vijay K Naik. 2012. Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE, 25–32.
[77] Zhenhuan Gong, Xiaohui Gu, and John Wilkes. 2010. Press: Predictive elastic resource scaling for cloud systems. In *Network and Service Management (CNSM), 2010 International Conference on*. Ieee, 9–16.
[78] Google BigQuery. 2020. https://cloud.google.com/bigquery. Accessed 20 January 2021.
[79] Google Persistent Disk. 2020. https://cloud.google.com/persistent-disk/. Accessed 20 January 2021.
[80] Abel Gordon, Michael Hines, Dilma Da Silva, Muli Ben-Yehuda, Marcio Silva, and Gabriel Lizarraga. 2011. Ginkgo: Automated, application-driven memory overcommitment for cloud computing. *Proc. RESoLVE* (2011).
[81] Robert Grandl, Ganesh Ananthanarayanan, Srikanth Kandula, Sriram Rao, and Aditya Akella. 2015. Multi-resource packing for cluster schedulers. *ACM SIGCOMM Computer Communication Review* 44, 4 (2015), 455–466.
[82] Robert Grandl, Srikanth Kandula, Sriram Rao, Aditya Akella, and Janardhan Kulkarni. 2016. G: Packing and Dependency-aware Scheduling for Data-Parallel Clusters. In *Proceedings of OSDI'16: 12th USENIX Symposium on Operating Systems Design and Implementation*. 81.
[83] Jim Gray and Franco Putzolu. 1987. The 5 minute rule for trading memory for disc accesses and the 10 byte rule for trading memory for CPU time. In *Proceedings of the 1987 ACM SIGMOD international conference on Management of data*. 395–398.
[84] Albert Greenberg, James R Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A Maltz, Parveen Patel, and Sudipta Sengupta. 2009. VL2: a scalable and flexible data center network. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*. 51–62.
[85] Ajay Gulati, Arif Merchant, and Peter J Varman. 2010. mClock: handling throughput variability for hypervisor IO scheduling. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*. USENIX Association, 437–450.
[86] Hakan Hacigümüş, Bala Iyer, Chen Li, and Sharad Mehrotra. 2002. Executing SQL over encrypted data in the database-service-provider model. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*. ACM, 216–227.
[87] Joseph M Hellerstein, Jose Faleiro, Joseph E Gonzalez, Johann Schleier-Smith, Vikram Sreekanti, Alexey Tumanov, and Chenggang Wu. 2018. Serverless computing: One step forward, two steps back. *arXiv preprint arXiv:1812.03651* (2018).
[88] Herodotos Herodotou, Harold Lim, Gang Luo, Nedyalko Borisov, Liang Dong, Fatma Bilgen Cetin, and Shivnath Babu. 2011. Starfish: A Self-tuning System for Big Data Analytics.. In *Cidr*, Vol. 11. 261–272.
[89] Kelsey Hightower, Brendan Burns, and Joe Beda. 2017. *Kubernetes: up and running: dive into the future of infrastructure*. " O'Reilly Media, Inc.".
[90] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D Joseph, Randy H Katz, Scott Shenker, and Ion Stoica. 2011. Mesos: A platform for fine-grained resource sharing in the data center.. In *NSDI*, Vol. 11. 22–22.
[91] Botong Huang, Matthias Boehm, Yuanyuan Tian, Berthold Reinwald, Shirish Tatikonda, and Frederick R Reiss. 2015. Resource elasticity for large-scale machine learning. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 137–152.
[92] Botong Huang, Nicholas WD Jarrett, Shivnath Babu, Sayan Mukherjee, and Jun Yang. 2015. Cümülön: Matrix-based data analytics in the cloud with spot instances. *Proceedings of the VLDB Endowment* 9, 3 (2015), 156–167.
[93] Jiamin Huang, Barzan Mozafari, Grant Schoenebeck, and Thomas F Wenisch. 2017. A top-down approach to achieving performance predictability in database systems. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 745–758.
[94] Jiamin Huang, Barzan Mozafari, and Thomas F Wenisch. 2017. Statistical analysis of latency through semantic profiling. In *Proceedings of the Twelfth European Conference on Computer Systems*. ACM, 64–79.
[95] Hyperspace: An indexing subsystem for Apache Spark. 2020. https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-performance-hyperspace?pivots=programming-language-csharp. Accessed 20 January 2021.
[96] IBM Netezza. 2020. "https://www.ibmbigdatahub.com/sites/default/files/document/redguide_2011.pdf". Accessed 20 January 2021.
[97] Calin Iorgulescu, Reza Azimi, Youngjin Kwon, Sameh Elnikety, Manoj Syamala, Vivek Narasayya, Herodotos Herodotou, Paulo Tomita, Alex Chen, Jack Zhang, et al. [n. d.]. PerfIso: Performance Isolation for Commercial Latency-Sensitive Services. In *2018 USENIX Annual Technical Conference USENIX ATC 18), pages=519–532, year=2018, organization=USENIX Association*.
[98] Navendu Jain, Ishai Menache, and Ohad Shamir. 2014. On-demand, spot, or both: Dynamic resource allocation for executing batch jobs in the cloud. (2014).
[99] Virajith Jalaparti, Chris Douglas, Mainak Ghosh, Ashvin Agrawal, Avrilia Floratou, Srikanth Kandula, Ishai Menache, Joseph Seffi Naor, and Sriram Rao. 2018. Netco: Cache and I/O Management for Analytics over Disaggregated Stores. In *Proceedings of the ACM Symposium on Cloud Computing*. ACM, 186–198.
[100] Alekh Jindal, Konstantinos Karanasos, Sriram Rao, and Hiren Patel. 2018. Selecting subexpressions to materialize at datacenter scale. *Proceedings of the VLDB Endowment* 11, 7 (2018), 800–812.
[101] Eric Jonas, Johann Schleier-Smith, Vikram Sreekanti, Chia-Che Tsai, Anurag Khandelwal, Qifan Pu, Vaishaal Shankar, Joao Carreira, Karl Krauth, Neeraja Yadwadkar, et al. 2019. Cloud programming simplified: A berkeley view on serverless computing. *arXiv preprint arXiv:1902.03383* (2019).
[102] Chris Jones and John Wilkes. 2016. Service Level Objectives. Site Reliability Engineering: How Google Runs Production Systems.
[103] Gopal Kakivaya, Lu Xun, Richard Hasha, Shegufta Bakht Ahsan, Todd Pfleiger, Rishi Sinha, Anurag Gupta, Mihail Tarta, Mark Fussell, Vipul Modi, et al. 2018. Service fabric: a distributed platform for building microservices in the cloud. In *Proceedings of the Thirteenth EuroSys Conference*. ACM, 33.
[104] Konstantinos Karanasos, Sriram Rao, Carlo Curino, Chris Douglas, Kishore Chaliparambil, Giovanni Matteo Fumarola, Solom Heddaya, Raghu Ramakrishnan, and Sarvesh Sakalanaga. 2015. Mercury: Hybrid centralized and distributed scheduling in large shared clusters. In *2015 {USENIX} Annual Technical Conference ({USENIX}{ATC} 15)*. 485–497.
[105] David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin. 1997. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. 654–663.
[106] Nodira Khoussainova, Magdalena Balazinska, and Dan Suciu. 2012. Perfxplain: debugging mapreduce job performance. *Proceedings of the VLDB Endowment* 5, 7 (2012), 598–609.
[107] Andreas Kipf, Thomas Kipf, Bernhard Radke, Viktor Leis, Peter Boncz, and Alfons Kemper. 2018. Learned cardinalities: Estimating correlated joins with deep learning. *arXiv preprint arXiv:1809.00677* (2018).
[108] D. E. Knuth and D. Bibby. 1986. *The TEXbook*. Addison-Wesley: Reading, MA.
[109] Paraschos Koutris, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. 2013. Toward practical query pricing with QueryMarket. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 613–624.
[110] Paraschos Koutris, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. 2015. Query-based data pricing. *Journal of the ACM (JACM)* 62, 5 (2015), 43.
[111] Kubernetes Documentation. 2020. https://kubernetes.io/docs. Accessed 20 January 2021.
[112] L. Lamport. 1994. *LATEX: A Document Preparation System* (second ed.). Addison-Wesley.
[113] Willis Lang, Karthik Ramachandra, David J DeWitt, Shize Xu, Qun Guo, Ajay Kalhan, and Peter Carlin. 2016. Not for the Timid: On the Impact of Aggressive Over-booking in the Cloud. *Proceedings of the VLDB Endowment* 9, 13 (2016), 1245–1256.

[114] Pedro Las-Casas, Giorgi Papakerashvili, Vaastav Anand, and Jonathan Mace. 2019. Sifter: Scalable Sampling for Distributed Traces, without Feature Engineering. In *Proceedings of the ACM Symposium on Cloud Computing*. 312–324.

[115] Jiexing Li, Arnd Christian König, Vivek Narasayya, and Surajit Chaudhuri. 2012. Robust estimation of resource consumption for sql queries using statistical techniques. *Proceedings of the VLDB Endowment* 5, 11 (2012), 1555–1566.

[116] David Lo, Liqun Cheng, Rama Govindaraju, Parthasarathy Ranganathan, and Christos Kozyrakis. 2015. Heracles: improving resource efficiency at scale. In *ACM SIGARCH Computer Architecture News*, Vol. 43. ACM, 450–462.

[117] Jiaheng Lu, Yuxing Chen, Herodotos Herodotou, and Shivnath Babu. 2019. Speedup Your Analytics: Automatic Parameter Tuning for Databases and Big Data Systems. *Proceedings of the VLDB Endowment* 12, 12 (2019), 1970–1973.

[118] Gang Luo, Jeffrey F Naughton, Curt J Ellmann, and Michael W Watzke. 2004. Toward a progress indicator for database queries. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. ACM, 791–802.

[119] Lin Ma, Bailu Ding, Sudipto Das, and Adith Swaminathan. 2020. Active learning for ML enhanced database systems. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 175–191.

[120] Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. 2019. Neo: A learned query optimizer. *arXiv preprint arXiv:1904.03711* (2019).

[121] Ryan Marcus and Olga Papaemmanouil. 2018. Deep reinforcement learning for join order enumeration. In *Proceedings of the First International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. ACM, 3.

[122] MarketResearch. 2019. https://www.marketsandmarkets.com/Market-Reports/cloud-database-as-a-service-dbaas-market-1112.html. Accessed 20 January 2021.

[123] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. 2010. Dremel: interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment* 3, 1-2 (2010), 330–339.

[124] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, Theo Vassilakis, Hossein Ahmadi, Dan Delorey, Slava Min, et al. 2020. Dremel: a decade of interactive SQL analysis at web scale. *Proceedings of the VLDB Endowment* 13, 12 (2020), 3461–3472.

[125] Microsoft Power BI . 2020. https://powerbi.microsoft.com. Accessed 20 January 2021.

[126] Pulkit A Misra, Íñigo Goiri, Jason Kace, and Ricardo Bianchini. [n. d.]. Scaling distributed file systems in resource-harvesting datacenters. In *2017 USENIX Annual Technical Conference USENIX ATC 17), pages=799–811, year=2017, organization=USENIX Association*.

[127] MongoDB Atlas. 2020. https://www.mongodb.com/. Accessed 20 January 2021.

[128] Hyun Jin Moon, Yun Chi, and Hakan Hacigümüs. 2010. SLA-aware profit optimization in cloud services via resource scheduling. In *Services (SERVICES-1), 2010 6th World Congress on*. IEEE, 152–153.

[129] Vivek Narasayya and Surajit Chaudhuri. 2021. Cloud Data Services: Workloads, Architectures and Multi-Tenancy. *Foundations and Trends® in Databases* 10, 1 (2021), 1–107. https://doi.org/10.1561/1900000060

[130] Vivek Narasayya, Sudipto Das, Manoj Syamala, Badrish Chandramouli, and Surajit Chaudhuri. 2013. Sqlvm: Performance isolation in multi-tenant relational database-as-a-service. (2013).

[131] Vivek Narasayya, Ishai Menache, Mohit Singh, Feng Li, Manoj Syamala, and Surajit Chaudhuri. 2015. Sharing buffer pool memory in multi-tenant relational database-as-a-service. *Proceedings of the VLDB Endowment* 8, 7 (2015), 726–737.

[132] Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. 2016. Oblivious Multi-Party Machine Learning on Trusted Processors.. In *USENIX Security Symposium*. 619–636.

[133] Oracle MultiTenant. 2013. https://www.oracle.com/technetwork/database/multitenant-wp-12c-1949736.pdf. Accessed 20 January 2021.

[134] Oracle SQL Trace. 2020. https://docs.oracle.com/database/121/TGSQL/tgsql_trace.htm. Accessed 20 January 2021.

[135] Jennifer Ortiz, Victor Teixeira De Almeida, and Magdalena Balazinska. 2015. Changing the Face of Database Cloud Services with Personalized Service Level Agreements.. In *CIDR*.

[136] Andrew Pavlo, Gustavo Angulo, Joy Arulraj, Haibin Lin, Jiexi Lin, Lin Ma, Prashanth Menon, Todd C Mowry, Matthew Perron, Ian Quah, et al. 2017. Self-Driving Database Management Systems.. In *CIDR*.

[137] Matthew Perron, Raul Castro Fernandez, David DeWitt, and Samuel Madden. 2020. Starling: A Scalable Query Engine on Cloud Functions. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 131–141.

[138] Raluca Ada Popa, Catherine Redfield, Nickolai Zeldovich, and Hari Balakrishnan. 2011. CryptDB: protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. ACM, 85–100.

[139] Adrian Daniel Popescu, Andrey Balmin, Vuk Ercegovac, and Anastasia Ailamaki. 2013. PREDIcT: towards predicting the runtime of large scale iterative analytics. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1678–1689.

[140] Jeff Rasley, Konstantinos Karanasos, Srikanth Kandula, Rodrigo Fonseca, Milan Vojnovic, and Sriram Rao. 2016. Efficient queue management for cluster scheduling. In *Proceedings of the Eleventh European Conference on Computer Systems*. ACM, 36.

[141] Mendel Rosenblum and Tal Garfinkel. 2005. Virtual machine monitors: Current technology and future trends. *Computer* 38, 5 (2005), 39–47.

[142] Sudip Roy, Arnd Christian König, Igor Dvorkin, and Manish Kumar. 2015. Perfaugur: Robust diagnostics for performance anomalies in cloud services. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE, 1167–1178.

[143] Felix Schuster, Manuel Costa, Cédric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich. 2015. VC3: Trustworthy data analytics in the cloud using SGX. In *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE, 38–54.

[144] Pat Selinger. 2017. Optimizer Challenges in a Multi-Tenant World. In *High Performance Transaction Systems, HPTS 2017*.

[145] Raghav Sethi, Martin Traverso, Dain Sundstrom, David Phillips, Wenlei Xie, Yutian Sun, Nezih Yegitbasi, Haozhun Jin, Eric Hwang, Nileema Shingte, et al. 2019. Presto: SQL on Everything. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1802–1813.

[146] Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, and John Wilkes. 2011. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 5.

[147] Benjamin H Sigelman, Luiz Andre Barroso, Mike Burrows, Pat Stephenson, Manoj Plakal, Donald Beaver, Saul Jaspan, and Chandan Shanbhag. 2010. Dapper, a large-scale distributed systems tracing infrastructure. (2010).

[148] Rohit Sinha, Manuel Costa, Akash Lal, Nuno P Lopes, Sriram Rajamani, Sanjit A Seshia, and Kapil Vaswani. 2016. A design and verification methodology for secure isolated regions. In *ACM SIGPLAN Notices*, Vol. 51. ACM, 665–681.

[149] Vikram Sreekanti, Chenggang Wu Xiayue Charles Lin, Jose M Faleiro, Joseph E Gonzalez, Joseph M Hellerstein, and Alexey Tumanov. 2020. Cloudburst: Stateful Functions-as-a-Service. *arXiv preprint arXiv:2001.04592* (2020).

[150] Tableau Online. 2020. https://www.tableau.com/products/cloud-bi. Accessed 20 January 2021.

[151] Alexander Thomson, Thaddeus Diamond, Shu-Chun Weng, Kun Ren, Philip Shao, and Daniel J Abadi. 2012. Calvin: fast distributed transactions for partitioned database systems. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 1–12.

[152] Stephen Tu, M Frans Kaashoek, Samuel Madden, and Nickolai Zeldovich. 2013. Processing analytical queries over encrypted data. In *Proceedings of the VLDB Endowment*, Vol. 6. VLDB Endowment, 289–300.

[153] Prasang Upadhyaya, Magdalena Balazinska, and Dan Suciu. 2016. Price-optimal querying with data apis. *Proceedings of the VLDB Endowment* 9, 14 (2016), 1695–1706.

[154] Bhuvan Urgaonkar, Prashant Shenoy, and Timothy Roscoe. 2009. Resource overbooking and application profiling in a shared internet hosting platform. *ACM Transactions on Internet Technology (TOIT)* 9, 1 (2009), 1.

[155] Leslie G Valiant and Gordon J Brebner. 1981. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*. 263–277.

[156] Dana Van Aken, Andrew Pavlo, Geoffrey J Gordon, and Bohan Zhang. 2017. Automatic database management system tuning through large-scale machine learning. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 1009–1024.

[157] Vinod Kumar Vavilapalli, Arun C Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, et al. 2013. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 5.

[158] Alexandre Verbitski, Anurag Gupta, Debanjan Saha, Murali Brahmadesam, Kamal Gupta, Raman Mittal, Sailesh Krishnamurthy, Sandor Maurice, Tengiz Kharatishvili, and Xiaofeng Bao. 2017. Amazon aurora: Design considerations for high throughput cloud-native relational databases. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 1041–1052.

[159] Alexandre Verbitski, Anurag Gupta, Debanjan Saha, James Corey, Kamal Gupta, Murali Brahmadesam, Raman Mittal, Sailesh Krishnamurthy, Sandor Maurice, Tengiz Kharatishvilli, et al. 2018. Amazon aurora: On avoiding distributed consensus for i/os, commits, and membership changes. In *Proceedings of the 2018 International Conference on Management of Data*. 789–796.

[160] Ben Verghese, Anoop Gupta, and Mendel Rosenblum. 1998. Performance isolation: sharing and isolation in shared-memory multiprocessors. In *ACM SIGPLAN Notices*, Vol. 33. ACM, 181–192.

[161] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. 2015. Large-scale cluster management at Google with Borg. In *Proceedings of the Tenth European Conference on Computer Systems*. ACM, 18.

[162] ESX VMware. [n. d.]. Understanding Memory Resource Management in VMware ESX 4.1. ([n. d.]).

[163] William Voorsluys, James Broberg, Srikumar Venugopal, and Rajkumar Buyya. 2009. Cost of virtual machine live migration in clouds: A performance evaluation.

In *IEEE International Conference on Cloud Computing*. Springer, 254–265.
[164] Carl A Waldspurger. 2002. Memory resource management in VMware ESX server. *ACM SIGOPS Operating Systems Review* 36, SI (2002), 181–194.
[165] Gerhard Weikum, Axel Moenkeberg, Christof Hasse, and Peter Zabback. 2002. Self-tuning database technology and information services: from wishful thinking to viable engineering. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 20–31.
[166] Craig D Weissman and Steve Bobrowski. 2009. The design of the force. com multitenant internet application development platform. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 889–896.
[167] Chenggang Wu, Jose Faleiro, Yihan Lin, and Joseph Hellerstein. 2019. Anna: A kvs for any scale. *IEEE Transactions on Knowledge and Data Engineering* (2019).
[168] Wentao Wu, Yun Chi, Hakan Hacigümüş, and Jeffrey F Naughton. 2013. Towards predicting query execution time for concurrent and dynamic database workloads. *Proceedings of the VLDB Endowment* 6, 10 (2013), 925–936.
[169] Wentao Wu, Xi Wu, Hakan Hacigümüş, and Jeffrey F Naughton. 2014. Uncertainty aware query execution time prediction. *Proceedings of the VLDB Endowment* 7, 14 (2014), 1857–1868.
[170] Pengcheng Xiong, Yun Chi, Shenghuo Zhu, Hyun Jin Moon, Calton Pu, and Hakan Hacgümüş. 2015. SmartSLA: Cost-sensitive management of virtualized resources for CPU-bound database services. *IEEE Transactions on Parallel and Distributed Systems* 26, 5 (2015), 1441–1451.
[171] Pengcheng Xiong, Yun Chi, Shenghuo Zhu, Junichi Tatemura, Calton Pu, and Hakan HacigümüŞ. 2011. ActiveSLA: a profit-oriented admission control framework for database-as-a-service providers. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 15.
[172] Yin Yang, Dimitris Papadias, Stavros Papadopoulos, and Panos Kalnis. 2009. Authenticated join processing in outsourced databases. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. ACM, 5–18.
[173] Dong Young Yoon, Ning Niu, and Barzan Mozafari. 2016. DBSherlock: A performance diagnostic tool for transactional databases. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 1599–1614.
[174] Heechul Yun, Gang Yao, Rodolfo Pellizzoni, Marco Caccamo, and Lui Sha. 2013. Memguard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2013 IEEE 19th*. IEEE, 55–64.
[175] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster computing with working sets. *HotCloud* 10, 10-10 (2010), 95.
[176] Zhi-Hui Zhan, Xiao-Fang Liu, Yue-Jiao Gong, Jun Zhang, Henry Shu-Hung Chung, and Yun Li. 2015. Cloud computing resource scheduling and a survey of its evolutionary approaches. *ACM Computing Surveys (CSUR)* 47, 4 (2015), 1–33.
[177] Ning Zhang, Peter J Haas, Vanja Josifovski, Guy M Lohman, and Chun Zhang. 2005. Statistical learning techniques for costing XML queries. In *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 289–300.
[178] Yunqi Zhang, George Prekas, Giovanni Matteo Fumarola, Marcus Fontoura, Íñigo Goiri, and Ricardo Bianchini. 2016. History-based harvesting of spare cycles and storage in large-scale datacenters. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*. 755–770.