Cloud Databases: New Techniques, Challenges, and Opportunities

Guoliang Li
Tsinghua University
liguoliang@tsinghua.edu.cn

Haowen Dong
Tsinghua University
dhw21@mails.tsinghua.edu.cn

Chao Zhang
Tsinghua University
cycchao@mail.tsinghua.edu.cn

ABSTRACT

As database vendors are increasingly moving towards the cloud data service, i.e., databases as a service (DBaaS), cloud databases have become prevalent. Compared with the early cloud-hosted databases, the new generation of cloud databases, also known as cloud-native databases, seek for higher elasticity and lower cost by developing new techniques, e.g., compute-storage disaggregation and the log is the database. To better harness the power of these cloud databases, it is important to study and compare the pros and cons of their key techniques. In this tutorial, we offer a comprehensive survey of cloud-native databases. Based on various system architectures, we introduce a taxonomy for the state-of-the-art cloud-native OLTP databases and OLAP databases, respectively. We then take a deep dive into their key techniques regarding storage management, transaction processing, analytical processing, data replication, serverless computing, database recovery, and security. Finally, we discuss the research challenges and opportunities.

PVLDB Reference Format:
Guoliang Li, Haowen Dong, and Chao Zhang. Cloud Databases: New Techniques, Challenges, and Opportunities. PVLDB, 15(12): 3758 - 3761, 2022.
doi:10.14778/3554821.3554893

1 INTRODUCTION

Background. Nowadays, cloud databases [2, 3, 5–7, 22] are increasingly proliferating as the database vendors are moving toward cloud data services. A recent Gartner report predicts [19] that the revenue from the cloud DBMS will account for 50% of total DBMS market revenue by the end of 2022, indicating that cloud databases play a crucial role in the next generation of data management systems. In the early stage of cloud data services, customers can choose the offered data service by the cloud vendors (i.e., databases as a service (DBaaS)), then pay for the on-demand resource fee based on the service level agreement (SLA) [8, 15, 16, 27, 30]. However, those providers regard the deployed databases as a general kind of software without any underlying optimizations.

Cloud-Native Databases. To further improve the elasticity and save the resources, cloud data service providers propose the cloud-native databases. The foremost innovation is the *disaggregation of compute and storage* architecture [22, 24], which decouples the storage from the compute nodes, then connects the compute nodes to shared storage through a high-speed network. On the one hand, the disaggregation architecture enables to expand the compute and storage resources independently, thereby bringing more elasticity for the customers. On the other hand, providers can support better multi-tenancy and alleviate the write amplification problem by writing the data into a unified and disaggregated log storage (i.e., the log is the database). However, as the compute and storage are disaggregated, many techniques are proposed to improve the performance, e.g., local caching [25], shared memory pool [5, 29], computation pushdown [25, 26], etc.

Challenges. There are three main challenges. (C1) Since the storage is disaggregated, it is challenging to support efficient transaction processing using a remote redo log, especially for the case that logs have yet been replayed for the secondary read-only nodes. (C2) Cloud databases need to reduce the network traffic by designing caching strategies and computational pushdown on the storage side. However, it is challenging to develop an efficient and effective method due to the computation limitation and cost of the storage side. (C3) Several cloud databases have supported *serverless computing* that can dynamically schedule resources for users' workloads with pre-defined rules, but it is still challenging for them to adaptively schedule the resources for the workloads in a fine-grained granularity [20]. To better harness the power of cloud databases, it is important to study the pros and cons of their key techniques.

Tutorial Overview. We will provide a comprehensive tutorial on cloud-native databases. The intended length of the tutorial is 1.5 hours. The tutorial consists of six sections as follows.

(1) Introduction (5 min). This section starts with an introduction to the motivation and challenges of cloud databases.

(2) Cloud OLTP Architectures (20 min). This section introduces three types of cloud OLTP disaggregation architectures.

(3) Cloud OLTP Techniques (30 min). This section introduces cloud-native OLTP techniques, including transaction processing, data replication, database node recovery, and storage management.

(4) Cloud OLAP Architectures (10 min). This section introduces two types of disaggregation architectures of cloud OLAP databases.

(5) Cloud OLAP Techniques (20 min). This section takes a deep dive into the key techniques of cloud-native OLAP databases, including query processing, storage management, serverless computing, security, and machine learning.

(6) Challenges and Open Problems (5 mins). The final section concludes the tutorial and discusses the research challenges and open problems for cloud-native databases.

2 TUTORIAL OUTLINE

2.1 Cloud-Native OLTP Architectures

As shown in Figure 1 and Table 1, we classify the architectures of cloud-native OLTP databases into three categories as follows:

(1) Disaggregated Compute-Storage OLTP Architecture. This category of databases adopts a disaggregation architecture that separates the compute nodes and storage nodes. Particularly, the

Proceedings of the VLDB Endowment, Vol. 15, No. 12 ISSN 2150-8097.
doi:10.14778/3554821.3554893

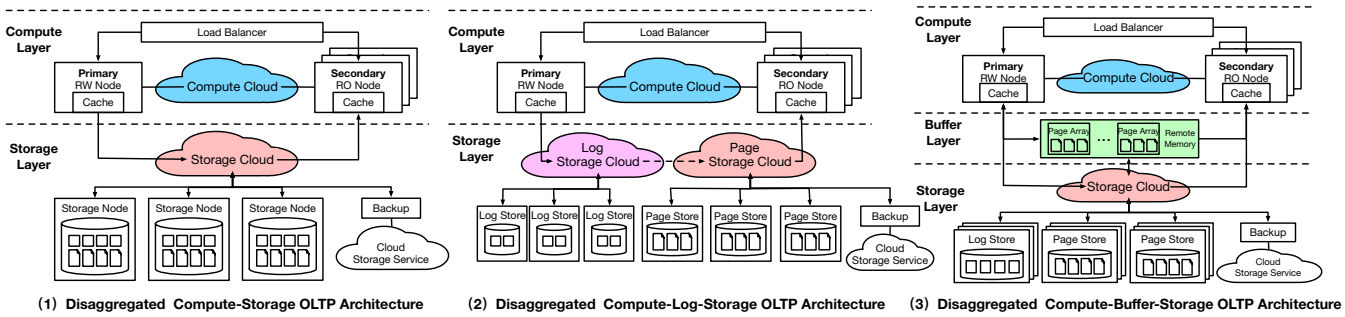

Figure 1: Architectures of Cloud-Native OLTP Databases

Table 1: A Classification of Cloud-Native OLTP Databases based on the Architecture

OLTP Architecture	Computation	Buffer	Storage
Disaggregated Compute-Storage	One RW Primary Node + Multiple RO Secondary Nodes	Local Cache for each Compute Node	Aggregated Log & Page Storage
Disaggregated Compute-Log-Storage	One RW Primary Node + Multiple RO Secondary Nodes	Local Cache for each Compute Node	Disaggregated Log & Page Storage
Disaggregated Compute-Buffer-Storage	One RW Primary Node + Multiple RO Secondary Nodes	Local Cache + Shared Remote Buffer	Disaggregated Log & Page Storage

Table 2: A Classification of Cloud-Native OLAP Architectures

OLAP Architecture	Computation	Storage
Disaggregated Compute-Storage	Multiple Clusters with Worker Nodes	Local Cache+ Cloud Storage
Disaggregated Compute-Memory-Storage	Multiple Worker Nodes with Shuffle Layer	Memory Pool+ Cloud Storage

logs are treated as the first-citizen storage to achieve high availability and consistency. A representative is Aurora [22], which is the first commercial cloud database that implements the disaggregated architecture. To reduce the I/O overhead caused by checkpointing, dirty page writing, and data synchronization, it offloads the redo processing to the storage tier.

(2) Disaggregated Compute-Log-Storage OLTP Architecture. This type of databases proposes a disaggregation architecture that separates the compute nodes, log service, and storage nodes. Compared with the first category, it achieves availability and durability by decoupling the log service from the storage. For instance, Socrates [2] adopts a four-tier abstraction of disaggregation architecture, which separates the storage into two parts: the `XLog Service` for log storage and `Page Servers` for page storage. The `XLog Service` persists write requests to logs, and the `Page Servers` are in charge of serving the read requests.

(3) Disaggregated Compute-Buffer-Storage OLTP Architecture. This category of databases [5, 29] develops a disaggregation architecture that separates the compute nodes, storage nodes, and the buffer. A shared remote buffer pool is used to serve as a unified data interface, thus compute nodes can read the pages from the same buffer area, which will reduce the duplicate data read from the same requests of different compute nodes and achieve high read throughput. For instance, PolarDB Serverless [5] adopts a shared buffer of all compute nodes. It utilizes various techniques, e.g., RDMA technology, cache invalidation, global latches, and read views to ensure data consistency and query performance of shared buffer area in the cloud.

2.2 Key Techniques of Cloud OLTP Databases

Table 3 summarizes four types of cloud-native techniques.

(1) Transaction Processing. We present three types of OLTP techniques. The first type is (i) Write Log, read from redo log [22] which uses logs for OLTP. The second type is (ii) Write log, read from page server[7] which uses logs and pages for OLTP. The third type is (iii) Write log, read from the shared memory[2, 5] which supports OLTP based on a disaggregated compute-memory-storage architecture.

(2) Data Replication We introduce three types of techniques for data replication in the cloud. Namely, (i) Quorum-based log replication [22, 23]; (ii) Paxos-based log replication [5]; and (iii) Log-page-separated replication [7].

(3) Database Node Recovery. We study two types of cloud-based techniques for node recovery: (i) ARIES-based recovery methods and (ii) Non-Redo recovery methods, which skips the Redo phase based on the mechanism of *the log is the database*.

(4) Storage Management. We introduce three types of cloud storage management for OLTP. Namely, (i) Coupled log-page storage [22], (ii) Decoupled log-page storage [7], and (iii) Decoupled log-buffer-page storage [2, 5].

2.3 Cloud-Native OLAP Architectures

As shown in Table 2 and Figure 2, we classify the architectures of cloud-native OLAP databases into two categories as follows:

(1) Disaggregated Compute-Storage OLAP Architecture. This category of databases adopt a disaggregated compute-storage architecture. The entire data is stored in shared storage, and hot

Table 3: An Overview of Key Techniques of Cloud-Native OLTP Databases

Technique Type	Main Approaches	Cloud Databases	Read Perf.	Write Perf.	Elasticity	Latency	Cost
Transaction Processing	Write Log, Read from Redo Log	Aurora[22]	Medium	High	High	Low	Medium
	Write Log, Read from Page Server	Taurus[7]	Medium	High	Excellent	Medium	Medium
	Write Log, Read from Shared Memory	PolarDB[5]	Excellent	Excellent	Excellent	Low	High
Data Replications	Quorum-based Log Replication	Aurora[22, 23]	High	Medium	High	Medium	Medium
	Paxos-based Log Replication	PolarDB[5]	High	High	High	High	High
	Log-Page-Separated Replication	Taurus[7]	Medium	High	High	Medium	Medium
Database Node Recovery	ARIES-based Recovery	Socrates[2]	-	-	High	Low	Medium
	Non-Redo Recovery	Aurora[22]	-	-	High	Medium	High
Storage Management	Coupled Log-Page Storage	Aurora[22]	Medium	-	High	Low	Medium
	Decoupled Log-Page Storage	Taurus[7]	Medium	-	Excellent	Medium	Medium
	Decoupled Log-Buffer-Page Storage	Socrates[2]	High	-	Excellent	Low	High

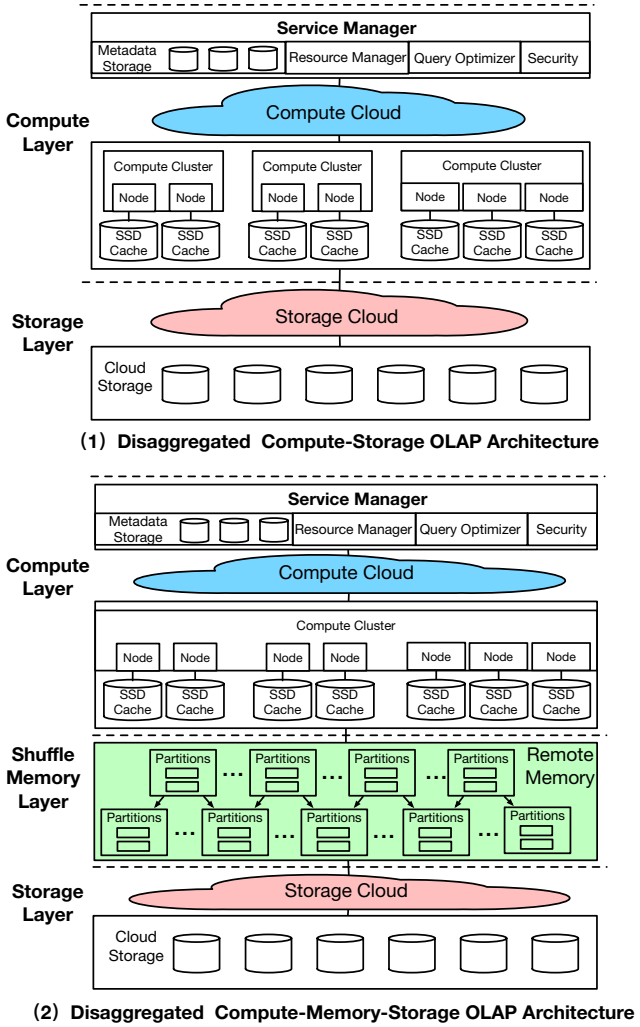

Figure 2: Architectures of Cloud-Native OLAP Databases

data is cached in the compute nodes on local SSDs. For instance, Snowflake [6, 24] owns an architecture with three layers: (1) the cloud service layer is in charge of the management of metadata, Virtual Warehouses (VWs), queries, transactions, and security. (2) the middle layer leverages VWs with EC2 instances to process the queries; (3) the storage layer uses the Amazon S3 to persist data.

(2) Disaggregated Compute-Memory-Storage OLAP Architecture. This category of databases adopts a disaggregated compute-memory-storage architecture with high elasticity. A representative is BigQuery [14] which is built on the Dremel query engine. It introduces a shared memory tier to accelerate the shuffle processing of the distributed joins, which significantly reduces the latency by avoid writing and reading the intermediate results from disks.

2.4 Key Techniques of Cloud OLAP Databases

Table 4 summarizes five types of OLAP-oriented techniques.

(1) Query Processing. We introduce three types of query processing. The first type is (i) Columnar scan with shuffle memory pool [14]. The second type is (ii) Columnar scan with pushdown [17, 24, 26], which aims to push the computation into the storage side, e.g., Amazon S3. The third type is (iii) Columnar scan with caching and pushdown [25].

(2) Storage Management. There are two types of storage management for cloud-native OLAP databases. The first type is (i) Local caching with a shared storage service [6, 17], and the second type is (ii) Unified memory pool with a storage system [14]. We will also introduce the semi-structured data management [6, 14, 28].

(3) Serverless Computing. We introduce two types of serverless computing in cloud databases. The first type is (i) Serverless with functions as a service [18], where queries are adaptively executed based on the cloud function services. The second type is (ii) Serverless with the elastic query engine [4], which enables to perform the queries by dynamically provisioning the query engine.

(4) Security. We present two types of data protection techniques: (i) Software-based data protection [6] and (ii) Hardware-based data protection, e.g., enclave in Intel SGX [1].

(5) Machine Learning. We will look at emerging cloud database techniques for machine learning, such as Sagemaker [13]. Moreover, we will introduce how cloud databases can benefit from machine learning techniques [10, 11, 21].

2.5 Challenges and Open Problems

Multiple Write Architecture. Existing cloud databases only support one write and multiple reads. Thus, it calls for cloud-native multiple write techniques that can scale out write capabilities.

Table 4: An Overview of Key Techniques of Cloud-Native OLAP Databases

Technique Type	Main Approaches	Cloud Databases	Throughput	Elasticity	Scalability	Cost
Query Processing	Columnar Scan with Pushdown	Snowflake[24], Redshift[17]	High	High	High	Medium
	Columnar Scan with Caching and Pushdown	FlexPushdownDB[25]	High	Medium	Medium	Medium
	Columnar Scan with Shuffle Memory Tier	BigQuery[14]	Excellent	Excellent	High	High
Storage Management	Local SSD Caching with Cloud Storage	Snowflake[6], Redshift[17]	High	High	Excellent	Medium
	Unified Memory Pool with Cloud Storage	BigQuery[14]	Excellent	Excellent	High	High
Serverless Computing	Serverless with Functions as a Service	Starling[18]	High	Excellent	Excellent	Medium
	Serverless with Elastic Query Engine	Athena[4]	Excellent	High	High	High
Security	Software-based Data Protection	Snowflake[24]	High	High	High	Low
	Hardware Data Protection (High Security)	Azure[1]	Low	Low	Low	High
Machine Learning	SQL-based ML Pipeline in the Cloud	Sagemaker [13]	High	High	High	High

Fine-grained Serverless. Existing elastic databases only support provisioning the resources for a query with coarse-grained serverless (e.g., query engine) but they suffer from the high latency of elastic scaling. It is challenging to support fine-grained serverless to efficiently schedule the resources for the incoming queries.

Cloud-Native HTAP Database. Existing cloud-native databases are either OLTP-oriented systems or OLAP-oriented systems, and there are no cloud-native HTAP systems [9, 12]. The main challenge is how to judiciously schedule the resources for OLTP and OLAP workloads with SLA-aware optimization.

Multi-Cloud Database. To provide cloud data services with high availability, e.g., zero downtime, it calls for multi-cloud databases that can harness the capability of multi-cloud deployment. It is challenging to effectively determine the strategy of data placement and to efficiently migrate the data among the multiple clouds.

3 BIOGRAPHY

Guoliang Li is a full professor at the Department of Computer Science, Tsinghua University. His research interests include database systems, large-scale data cleaning and integration. He got VLDB 2017 early research contribution award and TCDE 2014 early career award. He will present cloud OLTP databases and open problems.

Chao Zhang is a postdoctoral researcher at Tsinghua University. He will present cloud OLAP databases and the key techniques.

Haowen Dong is a PhD candidate at Tsinghua University. He will present the key techniques of cloud OLTP databases.

ACKNOWLEDGMENTS

This paper was supported by NSF of China (61925205), Huawei, TAL education, and Beijing National Research Center for Information Science and Technology (BNRist).

REFERENCES

[1] Panagiotis Antonopoulos, Arvind Arasu, Kunal D. Singh, et al. 2020. Azure SQL Database Always Encrypted. In *SIGMOD*. ACM, 1511–1525.
[2] Panagiotis Antonopoulos, Alex Budovski, Cristian Diaconu, et al. 2019. Socrates: The New SQL Server in the Cloud. In *SIGMOD*. ACM, 1743–1756.
[3] Nikos Armenatzoglou, Sanuj Basu, Naga Bhanoori, et al. 2022. Amazon Redshift Re-invented. In *SIGMOD*. ACM, 2205–2217.
[4] AWS. 2022. Severless Interactive Query Service. https://aws.amazon.com/athena/
[5] Wei Cao, Yingqiang Zhang, Xinjun Yang, et al. 2021. PolarDB Serverless: A Cloud Native Database for Disaggregated Data Centers. In *SIGMOD*. 2477–2489.
[6] Benoît Dageville, Thierry Cruanes, Marcin Zukowski, et al. 2016. The Snowflake Elastic Data Warehouse. In *SIGMOD*. ACM, 215–226.
[7] Alex Depoutovitch, Chong Chen, Jin Chen, et al. 2020. Taurus Database: How to be Fast, Available, and Frugal in the Cloud. In *SIGMOD*. ACM, 1463–1478.
[8] Hai Lan, Zhifeng Bao, and Yuwei Peng. 2021. A Survey on Advancing the DBMS Query Optimizer: Cardinality Estimation, Cost Model, and Plan Enumeration. *Data Science and Engineering* 6, 1 (2021), 86–101.
[9] Guoliang Li and Chao Zhang. 2022. HTAP Databases: What is New and What is Next. In *SIGMOD*. ACM, 2483–2488.
[10] Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. AI Meets Database: AI4DB and DB4AI. In *SIGMOD*. ACM, 2859–2866.
[11] Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. Machine Learning for Databases. *VLDB* 14, 12 (2021), 3190–3193.
[12] Guoliang Li, Xuanhe Zhou, Ji Sun, Xiang Yu, Yue Han, Lianyuan Jin, Wenbo Li, Tianqing Wang, and Shifu Li. 2021. openGauss: An Autonomous Database System. *VLDB* 14, 12 (2021), 3028–3041.
[13] Edo Liberty, Zohar S. Karnin, Bing Xiang, et al. 2020. Elastic Machine Learning Algorithms in Amazon SageMaker. In *SIGMOD*. ACM, 731–737.
[14] Sergey Melnik, Andrey Gubarev, Jing Jing Long, et al. 2020. Dremel: A Decade of Interactive SQL Analysis at Web Scale. *VLDB* 13, 12 (2020), 3461–3472.
[15] Vivek R. Narasayya and Surajit Chaudhuri. 2021. Cloud Data Services: Workloads, Architectures and Multi-Tenancy. *Foundations and Trends in Databases* 10, 1 (2021), 1–107.
[16] Vivek R. Narasayya, Ishai Menache, Mohit Singh, et al. 2015. Sharing Buffer Pool Memory in Multi-Tenant Relational Database-as-a-Service. *VLDB* 8, 7 (2015), 726–737.
[17] Ippokratis Pandis. 2021. The Evolution of Amazon Redshift. *VLDB* 14, 12 (2021), 3162–3163.
[18] Matthew Perron, Raul Castro Fernandez, David J. DeWitt, and Samuel Madden. 2020. Starling: A Scalable Query Engine on Cloud Functions. In *SIGMOD*. ACM, 131–141.
[19] Massimo Pezzini, Donald Feinberg, Nigel Rayner, and Roxane Edjlali. 2021. Magic Quadrant for Cloud Database Management Systems. *Gartner (2021, December 13)* (2021), 1–37.
[20] Johann Schleier-Smith. 2019. Serverless Foundations for Elastic Database Systems.. In *CIDR*.
[21] Ji Sun, Jintao Zhang, Zhaoyan Sun, Guoliang Li, and Nan Tang. 2021. Learned Cardinality Estimation: A Design Space Exploration and A Comparative Evaluation. *VLDB* 15, 1 (2021), 85–97.
[22] Alexandre Verbitski, Anurag Gupta, Debanjan Saha, et al. 2017. Amazon Aurora: Design Considerations for High Throughput Cloud-Native Relational Databases. In *SIGMOD*. ACM, 1041–1052.
[23] Alexandre Verbitski, Anurag Gupta, Debanjan Saha, et al. 2018. Amazon Aurora: On Avoiding Distributed Consensus for I/Os, Commits, and Membership Changes. In *SIGMOD*. ACM, 789–796.
[24] Midhul Vuppalapati, Justin Miron, Rachit Agarwal, et al. 2020. Building An Elastic Query Engine on Disaggregated Storage. In *NSDI*. 449–462.
[25] Yifei Yang, Matt Youill, Matthew E. Woicik, et al. 2021. FlexPushdownDB: Hybrid Pushdown and Caching in a Cloud DBMS. *VLDB* 14, 11 (2021), 2101–2113.
[26] Xiangyao Yu, Matt Youill, Matthew E. Woicik, et al. 2020. PushdownDB: Accelerating a DBMS Using S3 Computation. In *ICDE*. IEEE, 1802–1805.
[27] Haitao Yuan and Guoliang Li. 2021. A Survey of Traffic Prediction: from Spatio-Temporal Data to Intelligent Transportation. *Data Science and Engineering* 6, 1 (2021), 63–85.
[28] Chao Zhang, Jiaheng Lu, Pengfei Xu, and Yuxing Chen. 2018. UniBench: A Benchmark for Multi-model Database Management Systems. In *TPCTC*, Vol. 11135. Springer, 7–23.
[29] Yingqiang Zhang, Chaoyi Ruan, Cheng Li, et al. 2021. Towards Cost-Effective and Elastic Cloud Database Deployment via Memory Disaggregation. *VLDB* 14, 10 (2021), 1900–1912.
[30] Xuanhe Zhou, Chengliang Chai, Guoliang Li, and Ji Sun. 2022. Database Meets Artificial Intelligence: A Survey. *IEEE Transaction Knowledge Data Engineering* 34, 3 (2022), 1096–1116.