CSCI 5408

DATA MANAGEMENT AND WAREHOUSING

ASSIGNMENT – 1

Problem 1: Systematic literature review and critical analysis

Banner ID: B00948977

Git Assignment Link :
https://git.cs.dal.ca/sukumaran/csci5408_f23_b00948977_balaji_sukumaran/-/tree/main/Assignment1

Table of contents

Transaction recovery in federated distributed database systems

Summary

The paper discusses a more efficient method of performing transaction recovery and maintaining concurrency control in an environment prone to failures, specifically within a collection of cooperating databases, i.e., a federated database system. This paper also examines algorithms used in multi-database and federated systems, designs a recovery algorithm that uses a sync coordinator for federated distributed databases, and then implements and tests the newly designed algorithm [1, p.116].

The main problem addressed in this paper is the challenge of transaction recovery in federated database systems, due to issues such as database failures, heterogeneity, serializability, and the complexities of synchronous and asynchronous distributed databases [1, Section III]. The paper claims that these recovery mechanisms can be improved through enhancements in the system's architecture.

The literature review in the paper provides an overview of the benefits and challenges associated with distributed database systems. The federated state of a DBMS comprises several autonomous nodes, resulting in multiple levels of heterogeneity and autonomy. In business organizations [1, Section II], various distributed systems are implemented, allowing the creation of localized information systems to establish local control over data. Global queries are often executed on-demand on local databases. Data fragmentation, similar to that in domicile organizations, improves network reliability and enhances accessibility in a more economical manner. Although data replication across the federated database system benefits data recovery mechanisms, it also complicates the data synchronization process, as it requires undergoing several restoration processes across all replicated data.

The research designs and implements a more efficient transaction recovery algorithm for a federated database system, consisting of a sync coordinator, local transaction manager, and global transaction manager to synchronize partitioned global and local databases upon commit. To coordinate the transactions, a local transaction manager uses a concurrency control agent and a two-phase commit protocol agent. Similarly, the global transaction manager coordinates transactions across multiple resources. Synchronization of the local and global databases, to maintain consistency and avoid dirty reads, is handled by the sync coordinator [1, Section V]. This new architecture was tested in a bank transfer transaction scenario, demonstrating the applicability of the concept in a real-world environment.

Although the research was successful, there were some shortcomings. For instance, the paper primarily focuses on the sync coordinator [1, Section VI], with less emphasis on other components of the federated database system [2] that might also play a role in transaction recovery.

The authors suggest that further research should be conducted in the sync coordinator, specifically in relation to concurrency issues when using the sync coordinator at the wrapper level. This indicates that there are still aspects of the proposed architecture and algorithm that could be optimized and refined to better address the challenges of transaction recovery in federated database systems.

Critical Analysis:

The paper provides an overview of distributed databases, federated databases, and related concepts. But the concepts like heterogeneity and locking mechanism are not clearly explained. More examples and details could be added to help the readers who aren't familiar with these concepts.

The paper discusses about the limitations FDBS Recovery Algorithms [1, Section III] but it doesn't compare the proposed algorithm with existing algorithms. A comparison would have helped in highlighting the advantages and limitations of the proposed algorithm.

The explanation of the global transaction manager and Sync Coordinator algorithm is not so detailed. More steps and explanations could be added to make it clearer for readers how exactly these algorithms work.

The effectiveness of the proposed algorithm is not established properly, as the paper does not include any performance evaluations or experimental results. Including statistical data or case studies would have strengthened the paper's claims.

It would have been beneficial to include the intuition of the algorithm, more detailed steps, or challenges faced during the implementation to provide a comprehensive view of the process.

Overall, the paper provides valuable information on the design and implementation of recovery algorithms in federated distributed database systems. However, it could be improved by adding more technical details, statistical details, and case studies.

Key Technologies of Distributed Transactional Database Storage Engine

Summary:

The paper introduces data sharding as the most important function in a distributed transactional database [3, Section I] and it also provides a brief description [3, Section III] of the design and implementation of a distributed database storage engine's sharding scheme, including database distributed design, distributed storage design, data node layer design, data source layer design, data space layer design, and storage engine sharding solution implementation. Additionally, it also explains the design of a dynamic cluster [3, Section III] that can scale by adding or removing data nodes, and the implementation of data rebalancing to distribute data load evenly across the cluster. Management node retrieves information of the targeted data node and uses the decoupling module to remove it from the cluster. The data node status detection module then verifies the node's removal, and once confirmed, fully eliminates the node from the cluster.

Critical Analysis:

The paper provides a clear overview of the design and implementation of a distributed transactional database storage engine.

The three-tier architecture model for data sharding is well documented and provides a clear structure for the implementation of data sharding.

The inclusion of dynamic cluster scaling and data rebalancing is an interesting point, as it addresses the challenges of managing data load and cluster resources in a distributed database system.

To provide the context for the readers. The paper could have contained a literature review section that discusses previous work in the field of distributed transactional database storage engines.

While the authors discuss the importance of transaction consistency, there is a lack of detailed analysis on how their system ensures transaction consistency and handles conflicting transactions.

The paper didn’t discuss about the security measures implemented in the system to protect data integrity and confidentiality.

REFERENCES

[1] D. Damoah, J. B. Hayfron-Acquah, S. Sebastian, E. Ansong, B. Agyemang and R. Villafane, "Transaction recovery in federated distributed database systems," Proceedings of IEEE International Conference on Computer Communication and Systems ICCCS14, Chennai, India, 2014, pp. 116-123, doi: 10.1109/ICCCS.2014.7068178.

[2] “Components of a Database Management System,” Data entry source, [Online], December 18, 2022. Available: https://www.dataentryoutsourced.com/blog/components-of-a-database-management-system/ [Accessed: October 28, 2023]

[3] H. Xu, G. Chen, C. Zhang, J. Zhou, H. Wei and X. Gao, "Key Technologies of Distributed Transactional Database Storage Engine," 2020 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI), Harbin, China, 2020, pp. 106-112, doi: 10.1109/IAAI51705.2020.9332810.