**BULE HORA UNIVERSITY**

**COLLEGE OF INFORMATICS**

**DEPARTMENT OF INFORMATION SYSTEM**

**COURSE: - OBJECT-ORIENTED PROGRAMMING**

**COURSE CODE: - SEng6022**

**GROUP PROJCT**

**OBJECT-ORIENTED PROGRAMMING PROJECT**

**GROUP**

**NAME ID.NO**

**1. WARITU SULTAN ------------------------------------------------RU**

***SUBMITTED TO: SUBMITTED DATE:***

**1. What types of problems can occur during transactions in Ethiopian commercial banks**

that may result in losses exceeding in a million birr within a two-day period? As a database

administrator, what measures can be taken to address and resolve these issues?

In Ethiopian commercial banks, some potential problems during transactions that could lead to significant losses within a short period include:

- Double Spending: Occurs when the same funds are spent more than once due to processing errors or fraudulent activities.

- Unauthorized Transactions: Unauthorized access to accounts leading to unauthorized transfers or withdrawals.

- Data Inconsistencies: Inconsistent data due to concurrent updates or failures in processing.

As a database administrator, some measures to address and resolve these issues could include:

- Implementing robust authentication and authorization mechanisms to prevent unauthorized access.

- Utilizing encryption and secure communication protocols to protect sensitive data.

- Implementing transaction management techniques like ACID properties to maintain data integrity.

- Deploying real-time transaction monitoring systems to detect and prevent fraudulent activities.

- Employing data redundancy and disaster recovery mechanisms to ensure business continuity in case of failures.

**2. Compare and contrast different concurrency control techniques (e.g., locking vs. optimistic .**

locking, different locking granularities).

Locking: Acquires locks on data to ensure exclusive access during write operations. Can cause delays if locks are held for long periods.

- Optimistic Locking: Assumes no conflicts will occur, checks for conflicts at the end of transactions. Less overhead but may require rollbacks.

- Locking Granularities: Fine-granular locking can allow more concurrency but may lead to more overhead due to lock management.

**3. Discuss the advantages and disadvantages of each technique in various scenarios (e.g., high-concurrency vs. low-concurrency environments, read-dominated vs. write-dominated workloads).**

Advantages and Disadvantages of Each Technique:

- High-Concurrency vs. Low-Concurrency Environments:

- Locking: Suitable for low-concurrency environments but may cause bottlenecks in high-concurrency scenarios.

- Optimistic Locking: More appropriate for high-concurrency environments with a lower chance of conflicts.

- Read-Dominated vs. Write-Dominated Workloads:

- Locking: Better for write-dominated workloads where data consistency is crucial but may impact read operations.

- Optimistic Locking: Can be better for read-dominated workloads with infrequent write conflicts, as it minimizes contention.

Understanding these concurrency control techniques and their applicability in different scenarios can help database administrators design efficient and effective transaction management systems tailored to the specific needs and characteristics of their environments.

**5. Choose a real-world application (e.g., online banking, e-commerce) that requires robust concurrency control.Real-World Application Requiring Robust Concurrency Control: Online Reservation Systems**

Online Reservation Systems for services such as hotel bookings, airline reservations, event ticketing, car rentals, and restaurant reservations necessitate robust concurrency control mechanisms to handle high transaction volumes and ensure data consistency. Here's why they require strong concurrency control:

1. High Transaction Volume: Online reservation systems often experience a significant number of concurrent transactions from multiple users attempting to book the same service, especially during peak hours or popular events. Without effective concurrency control, conflicts can occur, leading to double bookings or inconsistencies in reservation data.

2. Consistency and Integrity: Data consistency is crucial in reservation systems to prevent overbooking or conflicting reservations. Robust concurrency control ensures that transactions are executed in a manner that maintains data integrity, preventing scenarios where two users book the same resource simultaneously.

3. Real-Time Updates: Online reservation systems demand real-time updates to reflect the latest availability of services, pricing changes, and booking status. Effective concurrency control mechanisms enable quick and accurate updates to reservation data without delays or conflicts.

4. Handling Race Conditions: Race conditions can occur in online reservation systems when multiple users attempt to book the same resource concurrently. Strong concurrency control mechanisms are essential to manage race conditions and ensure that bookings are processed correctly and without conflicts.

5. Optimizing Resource Utilization: By implementing effective concurrency control, online reservation systems can optimize resource utilization by allowing multiple users to access and book services concurrently while preventing conflicts or data inconsistencies.

Generally online reservation systems are a prime example of real-world applications that require robust concurrency control to ensure data consistency, handle high transaction volumes, prevent conflicts, and deliver a seamless user experience. Implementing efficient concurrency control mechanisms is essential to provide reliable and accurate reservation services to customers in a competitive and dynamic online environment.

**9. Research recent advancements and ongoing research in the field of concurrency control techniques.**

* **Machine Learning for Concurrency Control:**

- Advancement: Researchers are exploring the application of machine learning techniques for dynamic concurrency control in database systems. This approach aims to predict conflicts and optimize concurrency management based on past transaction patterns.

* **Conflict Detection and Resolution:**

- Advancement: Efforts are being made to enhance conflict detection and resolution strategies. Techniques such as timestamp-based protocols, optimistic concurrency control with conflict avoidance, and hybrid concurrency control mechanisms are being developed to improve data consistency and reduce contention.

* **Blockchain and Distributed Concurrency Control:**

- Advancement: With the rise of blockchain technology, research is focusing on developing concurrency control mechanisms for distributed databases and blockchain systems. Consensus algorithms like Proof of Work and Proof of Stake are being adapted to manage concurrent transactions and ensure data integrity across decentralized networks.

* **Adaptive Concurrency Control Policies:**

- Advancement: Adaptive concurrency control policies are being explored to dynamically adjust isolation levels and locking strategies based on workload and contention levels. This approach aims to optimize performance and scalability in varying workload conditions.

* **Optimistic Concurrency Control for In-Memory Databases:**

- Advancement: Researchers are investigating the application of optimistic concurrency control techniques in in-memory databases to boost performance and reduce overhead. Novel algorithms and protocols are being developed to enhance data consistency and enable efficient parallel processing of transactions.

* **Efficient Lock-Free Data Structures:**

- Advancement: Ongoing research is focused on designing lock-free data structures that eliminate the need for traditional locks in concurrent data access scenarios. Concurrent data structures like skip lists, lock-free queues, and non-blocking hash tables are being optimized for efficient multi-threaded access.

* **Transaction Scheduling and Parallel Query Processing:**

- Advancement: Research is exploring enhanced transaction scheduling algorithms and parallel query processing techniques to improve throughput and scalability in highly concurrent database environments. Approaches such as speculative execution and transaction batching are being investigated for efficient resource utilization.

* **Multi-Version Concurrency Control (MVCC) Optimization:**

- Advancement: Ongoing research is focusing on optimizing Multi-Version Concurrency Control mechanisms for improved performance and reduced storage overhead. Advanced MVCC techniques with better garbage collection strategies and version storage management are under development.

As the demand for efficient and scalable concurrency control mechanisms continues to grow with the increasing volume and complexity of data transactions, ongoing research efforts are exploring innovative approaches to enhance data consistency, optimize performance, and address the challenges of concurrency management in modern database systems.

1. **Identify a specific limitation or challenge in existing concurrency control**

**techniques.**

Limitation in Existing Concurrency Control Techniques: High Overheads and Scalability Challenges with Granular Locking

**1. Granular Locking Overhead:**

- Issue: Granular locking, such as object-level locking, can lead to high overhead due to the management of numerous fine-grained locks. This overhead is primarily associated with lock acquisition, release, and conflict resolution.

- Challenge: Managing a large number of locks at a granular level can result in increased lock contention, lock memory consumption, and synchronization costs. This overhead can impact system performance, scalability, and resource utilization.

**2. Lock Hierarchy and Deadlock Risk:**

- Issue: When implementing granular locking, maintaining a complex hierarchy of locks can increase the risk of deadlocks. Deadlocks occur when transactions hold locks on different resources and are waiting for each other to release locks, causing a cyclic wait scenario.

- Challenge: Managing a hierarchical lock structure to prevent deadlocks can be cumbersome and resource-intensive. The need for deadlock detection mechanisms and resolution strategies adds complexity to concurrency control implementations.

**3. Scalability Challenges:**

- Issue: Granular locking may hinder scalability, especially in systems with high contention and concurrent access patterns. As the number of transactions and data entities increases, the granularity of locks can lead to bottlenecks and limit the system's ability to accommodate a growing workload.

- Challenge: Supporting a large number of fine-grained locks in a highly scalable system poses challenges in maintaining performance and ensuring efficient resource utilization. Balancing the need for data consistency with the scalability requirements becomes a critical consideration.

**4. Contention and Performance Impact:**

**- Issue:** Granular locking can escalate contention among transactions competing for fine-grained locks, potentially leading to increased wait times, reduced throughput, and degraded system performance.

**- Challenge:** Minimizing lock contention while ensuring data consistency is a delicate balance. The design and implementation of efficient concurrency control mechanisms to mitigate contention without compromising performance remain a significant challenge.

**5. Complexity and Maintenance Overhead:**

- Issue: Managing and maintaining a large number of fine-grained locks introduce complexity in concurrency control implementations. The intricate lock management process requires careful coordination and oversight, adding to system maintenance overhead.

- Challenge: Balancing the need for granularity in locking to ensure data consistency with the complexity and maintenance overhead associated with the management of numerous locks is a persistent challenge. Simplifying lock management without sacrificing the benefits of granular control poses a significant hurdle.

Addressing these limitations in granular locking mechanisms requires innovative approaches to enhance concurrency control efficiency, reduce overhead, improve scalability, and mitigate challenges related to contention and performance degradation. Research efforts are ongoing to develop optimized concurrency control techniques that strike a balance between the granularity of locking and the associated complexities and limitations.

**14. Explain the difference between serializability and isolation in the context of concurrency**

**control.**