CS 3306-01 Databases 2

Instructor: Professor David Nagus

Name: Anonymos

Written Assignment Unit 2

**Introduction**

In database systems, a deadlock occurs when multiple transactions are waiting for each other’s resources, resulting in an indefinite waiting state. This situation can halt transaction processing and significantly impact overall system performance. When a deadlock is detected, it must be resolved quickly. In this paper, I will explain the three actions that need to be taken when a deadlock is detected, using Figure 16.19 as a reference.

**Deadlock Detection**

Deadlock detection is typically performed using a wait-for graph. As shown in Figure 16.19, transactions T26, T27, and T28 are mutually waiting for resources held by each other, forming a cycle, and are thus in a deadlock state. In such graphs, the presence of a cycle indicates the occurrence of a deadlock. To resolve the deadlock, one or more transactions must be rolled back, allowing other transactions to proceed.

**Three Actions for Deadlock Recovery**

1. Choosing the Transaction to Roll Back

The first action to be taken after detecting a deadlock is to decide which transaction to roll back. In Figure 16.19, T26, T27, and T28 form a cycle and are deadlocked. In this case, the system selects the transaction with the lowest cost to roll back. Typically, younger transactions or transactions that have consumed fewer resources are selected. For instance, T27 may be chosen to be rolled back because it is a younger transaction compared to T26 and T28.

2. Rolling Back the Selected Transaction

The next step is to roll back the selected transaction. This process reverses all operations performed by the transaction, and the resources held by it are released. For example, if T27 is rolled back, the resources held by T27 are freed, allowing T28 to proceed. Rolling back a transaction is crucial for maintaining data consistency while restoring the system to a normal state.

3. Restarting the Other Transactions

After the rollback is complete, the other transactions that were part of the deadlock (in this case, T26 and T28) can be restarted. With T27's resources released, T26 and T28 can resume execution, and the system returns to normal processing. Restarting the transactions ensures that system performance recovers, and resources are managed efficiently.

**Conclusion**

Detecting and recovering from deadlocks is essential to maintaining the stability and performance of a database system. When a deadlock is detected, three actions must be performed: choosing the transaction to roll back, executing the rollback, and restarting the other transactions. By following these steps, the system can recover from the deadlock, and transactions can resume processing. Moving forward, it is also crucial to implement proper locking mechanisms during the design phase of transactions to prevent deadlocks from occurring.

Word Count: 438

References

1. Silberschatz, A., Korth, H.F., & Sudarshan, S. (2001). *Database System Concepts (4th ed.).* New York, NY: McGraw-Hill. Available at Database System Concepts 4th Edition By Silberschatz-Korth-Sudarshan.pdf