Paper Name: Database Management Systems

Topic: Concurrent Transaction

Paper Code: BCA CC410

By Ms. Manisha Prasad

Head, Assistant Professor,

Department of Computer Science

Patna Women's College

Concurrent Transactions

Transaction-processing systems usually allow multiple transactions to run simultaneously. Simultaneous execution of transactions means interleaving of operations of multiple transactions. The two good reasons for allowing concurrency are:

Improved throughput and resource utilization:

A transaction consists of many steps. Some involve I/O activity; others involve CPU activity. The CPU and the I/O devices in a computer system can operate in parallel. Therefore, I/O activity can be done in parallel with processing at the CPU. The parallelism of the CPU and the I/O system can therefore be exploited to run multiple transactions in parallel. While a read or write on behalf of one transaction is in progress on one disk, another transaction can be running in the CPU, while another disk may be executing a read or write on behalf of a third transaction. All of this increases the throughput of the system. **Throughput** is defined as the number of transactions executed in a given amount of time. Therefore, the processor and disk utilization also increases i.e. the processor and disk spend less time idle, or not performing any useful work.

Reduced waiting time:

There may be a mix of transactions running on a system, some short and some long. If transactions run serially(that means only one transaction can run at a time), a short transaction may have to wait for a preceding long transaction to complete, which can lead to unpredictable delays in running a transaction. If the transactions are operating on different parts of the database, it is better to let them run concurrently, sharing the CPU cycles and disk accesses among them. Concurrent execution reduces the unpredictable delays in running transactions. Moreover, it also reduces the **Average Response Time** i.e. the average time for a transaction to be completed after it has been submitted.

However, allowing multiple transactions to update data concurrently causes several complications with consistency of the data. Ensuring consistency in spite of concurrent execution of transactions requires extra work; it is far easier to insist that transactions run serially—that is, one at a time, each starting only after the previous one has completed. The database system must control the interaction among the concurrent transactions to prevent them from destroying the consistency of the database. It is achieved using Concurrency-control schemes.