After completing this chapter, you will be able to:

- Learn about transaction.
- Gain the knowledge about properties of transaction.
- Understand the protocols used in transaction.
- Implement transaction practically.

**INTRODUCTION TO TRANSACTION**

This chapter covers very important concept of transaction. Transaction management is vital in the real time applications. In daily life we deal with transaction processing directly or indirectly.

A transaction is a logical unit of work. In other words it is group of one or more operations and all operations work as unit. If one operation fails entire transaction fails. For example when you purchase the item online by placing the order a transaction occurs. Let us understand how this transaction works. You place order of number of items and then you pay the money online banking system through the payment gateway. The bill amount is credited to the
vendor’s bank account and same amount is debited from your account. So this transaction involves two operations, credit and debit.

Now imagine that from these two operations only debit operation is executed, but credit is not executed. This means that the bill amount is debited from your bank account and not credited in the vendor’s account. You will not be happy! Likewise, if credit operation is executed and debit operation is not executed, you will be happy. These two situations are annoying from any of the related entities. There must be a solution to overcome this problem.

The solution to this problem is the use of the transaction mechanism in such situations. Put both debit and credit operations in the transaction. Commit the transaction if all the operations are executed successfully. Undo all the operations if any one operation fails to execute. The process of undoing the operations is known as rolling back the transaction. Now the both entities in the online purchase example are happy as there is no inconsistency because of the transaction management.

A transaction contains following ACID properties:

- **Atomic**: The operations of a transaction should work like an atom. This means you cannot separate the operations either all the operations should be executed or none of the operations should be executed. In other words, either transaction should be completed 100% or 0% there is no intermediate status such as 50%. Here 100% transaction should be committed and 0% means transaction should be rolled back.

- **Consistent**: The operations of a transaction should put a data base in consistent. For example, if Account 1 has a balance of amount 12000 and Account2 has a balance of amount 8000. Now if we perform a balance transfer of amount 1000 from account 1 to account 2, at the end of transaction Account 1 should have a balance of 11000 and Account 2 should have a balance of 9000. This involves two operations: One is the debit operation which deducts amount of 1000 from the balance of Account 1 and another is credit operation which adds amount of 1000 to the balance of Account 2. If either of the operations fails, then all the operations should be rolled back and the Account 1 and Account 2 should have an amount which was there prior to initiating the transaction. That is Account 1 should have an amount of 12000 and Account 2 should have an amount of 8000. In summary, if the transaction is committed, then Account 1 should have a balance of 11000 and Account 2 should have a balance of 9000 and if the transaction is rolled back, then Account 1 should have a balance of 12000 and Account 2 should have a balance of 8000.
Data Hiding Schemes Based on Singular Value Decomposition
www.igi-global.com/chapter/data-hiding-schemes-based-on-singular-value-decomposition/127161?camid=4v1a

Automatic Intrusion Detection and Secret Multi Agent Preservation Using Authentication Measurement Network Threat
www.igi-global.com/chapter/automatic-intrusion-detection-and-secret-multi-agent-preservation-using-authentication-measurement-network-threat/127152?camid=4v1a