**Transaction Concept**

A **transaction** is a *unit* of program execution that accesses and possibly updates various data items

A transaction is the DBMS’s abstract view of a user program: a sequence of reads and writes

A transaction must see a consistent database

During transaction execution the database may be temporarily inconsistent

A sequence of many actions which are considered to be one atomic unit of work

When the transaction completes successfully (is committed), the database must be consistent

After a transaction commits, the changes it has made to the database persist, even if there are system failures

Multiple transactions can execute in parallel

Two main issues to deal with:

Failures of various kinds, such as hardware failures and system crashes

Concurrent execution of multiple transactions

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-

**ACID Properties**

To preserve the integrity of data the database system transaction mechanism must ensure:

***Atomicity*.** Either all operations of the transaction are properly reflected in the database or none are

***Consistency*.** Execution of a transaction in isolation preserves the consistency of the database

***Isolation*.** Although multiple transactions may execute concurrently, each transaction must be unaware of other concurrently executing transactions. Intermediate transaction results must be hidden from other concurrently executed transactions

That is, for every pair of transactions *Ti* and *Tj,* it appears to *Ti* that either *Tj,* finished execution before *Ti* started, or *Tj* started execution after *Ti* finished

***Durability*.** After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures

--------------------------------------------