BITS, PILANI – K. K. BIRLA GOA CAMPUS

Database Systems

(IS F243)

by

Mrs. Shubhangi Gawali

Dept. of CS and IS

TRANSACTION MANAGEMENT

Definition of Transcation

- Unit of program that accesses and updates various data items
- Set of instructions
- Sequential order
- ACID properties

Responsible Components

Property of transaction	**Component**
Atomicity	Transaction Management Component
Concurrency	Application Programmer
Isolation	Concurrency Control Component
Durability	Recovery Management Component

Types of failures

- **Computer Failure :** A H/W or S/W error in the computer during transaction execution.
- **Transaction Failure :** Failure caused by an operation in the transaction e.g. Division by zero.
- **Local errors :** Conditions that cause the cancellation of transaction e.g. data needed for transaction is not found.

Types of errors

- **Concurrency control enforcement :** Either of the transaction need to be aborted by CC methods.

  e.g withdrawal from ATM and Funds transfer.

- **Disk Failure :** Loss of data in disk blocks during a transaction due to say a disk read/write head.

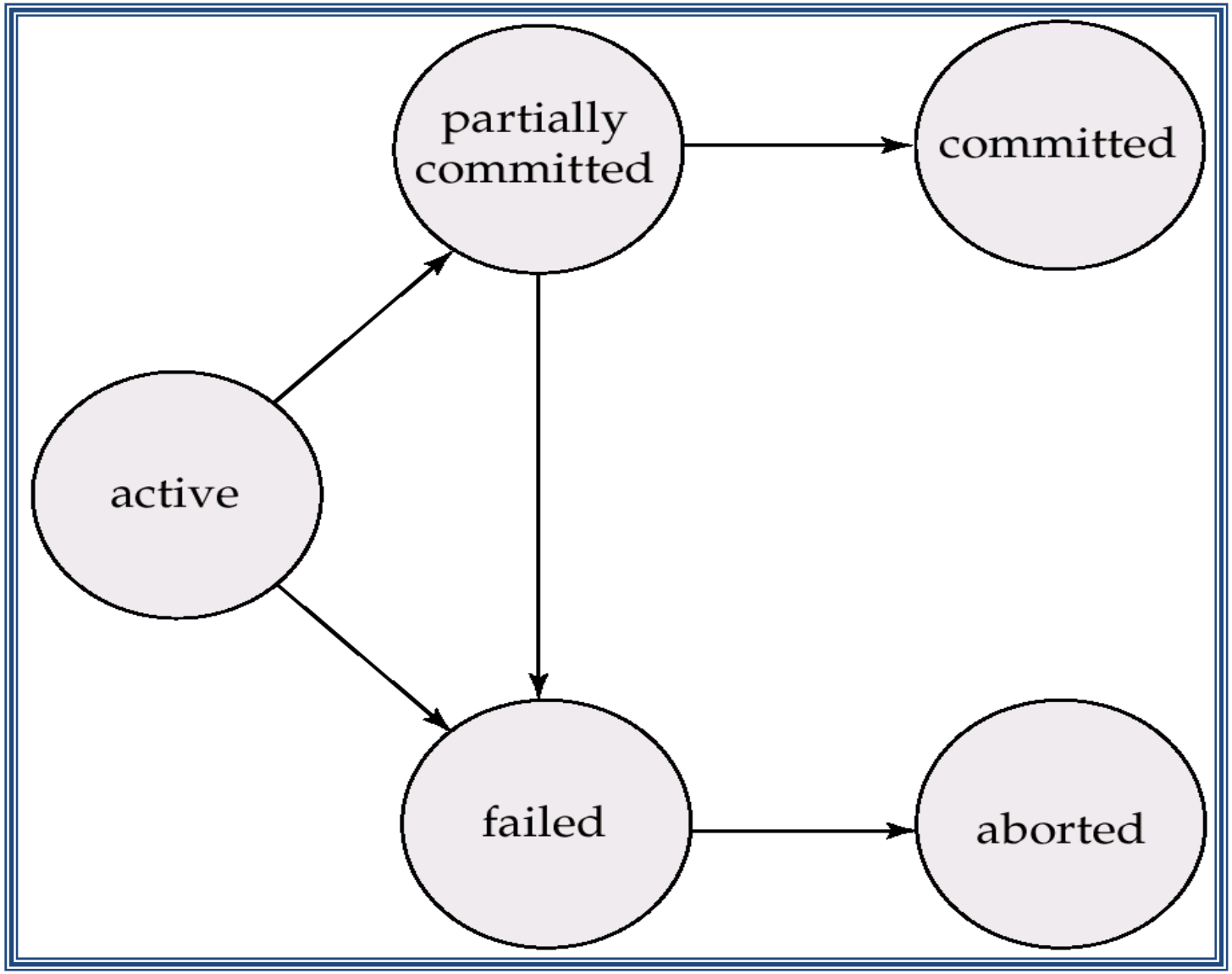
- **Physical problems & catastropes :** problems like power failure, earthquake i.e beyond human control.

TCL Commands

TCL : Transaction Control Language

- COMMIT : save work done to the server end
- SAVEPOINT : identify a point in a transaction to which we can later roll back
- ROLLBACK : Undo the transaction operations till last savepoint or till mentioned savepoint

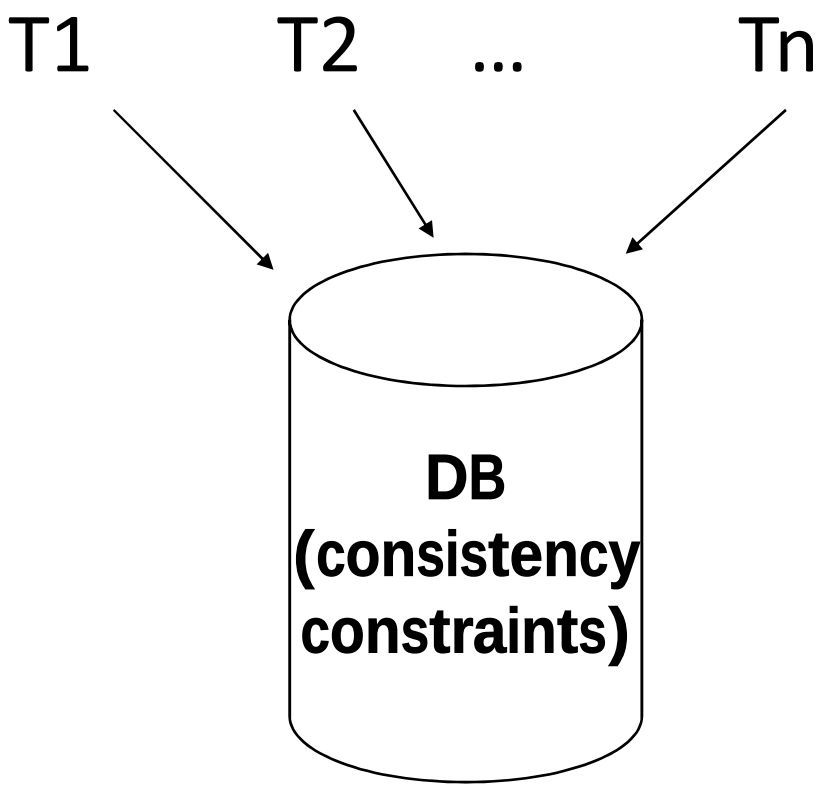
Transaction State Diagram



Transaction State Diagram

- **Active,** the initial state; the transaction stays in this state while it is executing
- **Partially committed,** before the final statement has been executed.
- **Failed,** after the discovery that normal execution can no longer proceed.
- **Aborted,** after the transaction has been rolled back and the database restored to its state prior to the start of the transaction. Two options after it has been aborted:
  - restart the transaction – only if no internal logical error
  - kill the transaction
- **Committed,** after successful completion.

Concurrent Executions



Multiple transactions are allowed to run concurrently in the system.

Concurrent Executions

- Advantages :
  - **increased processor and disk utilization**, leading to better transaction *throughput:* one transaction can be using the CPU while another is reading from or writing to the disk
  - **reduced average response time** for transactions: short transactions need not wait behind long ones.

Concurrent Executions

Concurrency control schemes – mechanisms to achieve isolation,

i.e., to control the interaction among the concurrent transactions in order to prevent them from destroying the consistency of the database

Schedules

- Schedules – sequences that indicate the order in which instructions of concurrent transactions are executed
  - a schedule for a set of transactions must consist of all instructions of those transactions
  - must preserve the order in which the instructions appear in each individual transaction.

Example

- Consider two transactions (transactions) :

```
T1: BEGIN   A=A+1500,   B=B -1500   END
T2: BEGIN   A=1.06*A,    B=1.06*B   END
```

- Intuitively, the first transaction is transferring Rs1500 from B's account to A's account. The second is crediting both accounts with a 6% interest payment.

- There is no guarantee that T1 will execute before T2 or vice-versa, if both are submitted together. However, the net effect *must* be equivalent to these two transactions running serially in some order.

Anomalies due to interleaved executions

- Reading Uncommitted data (WR Conflicts)
- Unrepeatable Reads (RW Conflicts)
- Overwriting Uncommitted Data (WW Conflicts)

Example (Contd.)

- Consider a possible interleaving (*schedule*):

```
T1:  A=A+1500,              B=B-1500
T2:             A=1.06*A,              B=1.06*B
```

- The above one is OK. But what about:

```
T1: A=A+1500,                          B=B-1500
T2:             A=1.06*A, B=1.06*B
```

- The DBMS's view of the second schedule:

```
T1: R(A), W(A),                          R(B), W(B)
T2:             R(A), W(A), R(B), W(B)
```