University of Central Florida
CGS 2545
Database Concepts

DEPARTMENT OF ELECTRICAL ENGINEERING & COMPUTER SCIENCE
COMPUTER SCIENCE DIVISION

Transaction

- A transaction can be defined as a group of tasks.
- A single task is the minimum processing unit which cannot be divided further.
- Example of a simple transaction.
  - Suppose a bank employee transfers Rs 500 from A's account to B's account.
  - This very simple and small transaction involves several low-level tasks.

Transaction

- Example of a simple transaction.

A's Account

```
Open_Account(A)
Old_Balance = A.balance
New_Balance = Old_Balance - 500
A.balance = New_Balance
Close_Account(A)
```

B's Account

```
Open_Account(B)
Old_Balance = B.balance
New_Balance = Old_Balance + 500
B.balance = New_Balance
Close_Account(B)
```

Transactions

- Properties of Transactions
  - Transactions have the following four standard properties, usually referred to by the acronym **ACID**.
    - **Atomicity**
      - ensures that all operations within the work unit are completed successfully.
      - Otherwise, the transaction is aborted at the point of failure and all the previous operations are rolled back to their former state.

Transactions

- Properties of Transactions
  - Transactions have the following four standard properties, usually referred to by the acronym **ACID**.
    - **Consistency**
      - ensures that the database properly changes states upon a successfully committed transaction.
    - **Isolation**
      - enables transactions to operate independently of and transparent to each other.
    - **Durability**
      - ensures that the result or effect of a committed transaction persists in case of a system failure.

Transaction

- Serializability
  - When multiple transactions are being executed by the operating system in a multiprogramming environment, there are possibilities that instructions of one transactions are interleaved with some other transaction.
    - Schedule
    - Serial Schedule

Transaction

- Serializability
  - **Schedule**
    - A chronological execution sequence of a transaction is called a schedule.
    - A schedule can have many transactions in it, each comprising of a number of instructions/tasks.

Transaction

- Serializability
  - **Serial Schedule**
    - It is a schedule in which transactions are aligned in such a way that one transaction is executed first.
    - When the first transaction completes its cycle, then the next transaction is executed.
    - Transactions are ordered one after the other.
    - This type of schedule is called a serial schedule, as transactions are executed in a serial manner.

Transaction

- Serializability
  - In a multi-transaction environment, serial schedules are considered as a benchmark.
  - The execution sequence of an instruction in a transaction cannot be changed, but two transactions can have their instructions executed in a random fashion.
  - This execution does no harm if two transactions are mutually independent and working on different segments of data; but in case these two transactions are working on the same data, then the results may vary.
  - This ever-varying result may bring the database to an inconsistent state.

Transaction

- Serializability
  - To resolve this problem, we allow parallel execution of a transaction schedule, if its transactions are either serializable or have some equivalence relation among them.

Transaction

- Equivalence Schedules
  - An equivalence schedule can be of the following types
    - Result equivalence
    - View equivalence
    - Conflict equivalence

Transaction

- Equivalence Schedules
  - Result Equivalence
    - If two schedules produce the same result after execution, they are said to be result equivalent.
    - They may yield the same result for some value and different results for another set of values.
    - That's why this equivalence is not generally considered significant.

Transaction

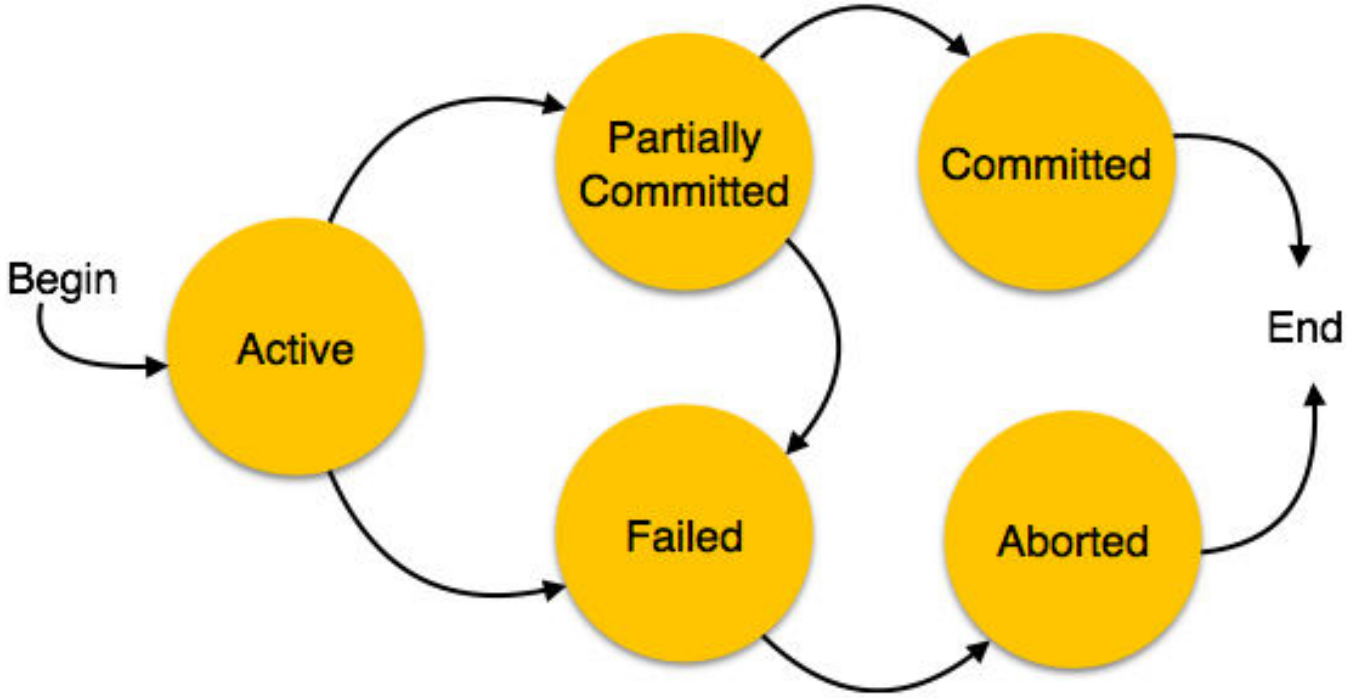
- Equivalence Schedules
  - View Equivalence
    - Two schedules would be view equivalence if the transactions in both the schedules perform similar actions in a similar manner.
    - For example
      - If T reads the initial data in S1, then it also reads the initial data in S2.
      - If T reads the value written by J in S1, then it also reads the value written by J in S2.
      - If T performs the final write on the data value in S1, then it also performs the final write on the data value in S2.

Transaction

- Equivalence Schedules
  - Conflict Equivalence
    - Two schedules would be conflicting if they have the following properties
      - Both belong to separate transactions.
      - Both accesses the same data item.
      - At least one of them is "write" operation.
    - Two schedules having multiple transactions with conflicting operations are said to be conflict equivalent if and only if −
      - Both the schedules contain the same set of Transactions.
      - The order of conflicting pairs of operation is maintained in both the schedules.
  - **Note** − View equivalent schedules are view serializable and conflict equivalent schedules are conflict serializable. All conflict serializable schedules are view serializable too.

Transaction

- States of Transactions
  - A transaction in a database can be in one of the following states



Transaction

- States of Transactions
  - **Active**
    - In this state, the transaction is being executed. This is the initial state of every transaction.
  - **Partially Committed**
    - When a transaction executes its final operation, it is said to be in a partially committed state.
  - **Failed**
    - A transaction is said to be in a failed state if any of the checks made by the database recovery system fails.
    - A failed transaction can no longer proceed further.

Transaction

- States of Transactions
  - **Aborted**
    - If any of the checks fails and the transaction has reached a failed state, then the recovery manager rolls back all its write operations on the database to bring the database back to its original state where it was prior to the execution of the transaction.
    - Transactions in this state are called aborted.
    - The database recovery module can select one of the two operations after a transaction aborts
      - Re-start the transaction
      - Kill the transaction

Transaction

- States of Transactions
  - **Committed**
    - If a transaction executes all its operations successfully, it is said to be committed.
    - All its effects are now permanently established on the database system.