Some repeated questions

Created by	Ⓑ Borhan
Last edited time	@December 9, 2024 9:48 PM
Tag	

A transaction can be defined as a group of tasks. A transaction is a single collection of operations that forms a single logical unit of work.

- **Atomicity** − This property states that a transaction must be treated as an **atomic unit**, that is, either all of its operations ar**e executed or none.** There must be no state in a database where a transaction is left partially completed. States should be defined either before the execution of the transaction or after the execution/abortion/failure of the transaction.
- **Consistency** − The database must r**emain in a consistent state after any transaction**. No transaction should have any **adverse effect** on the data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.
- **Durability** − The database should be **durable enough to hold all its latest updates** even if the **system fails** or restarts. If a transaction updates a chunk of data in a database and commits, then the database wil**l hold the modified data.** If a transaction commits but the **system fails** before the data could be written on to the disk, then that data **will be updated once the system springs** back into action.
- **Isolation** − In a database system where more than one transaction are being **executed simultaneously** and in **parallel**, the property of isolation states that all the transactions will be carried out and executed as if it is the **only transaction in the system**. N**o transaction will affect the existence of any other 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.

A concurrent schedule is serializable if its outcome is the same as **some serial schedule of the same transactions**.

Key Concepts of Serializability

1. **Serial Schedule**: A schedule where **transactions are executed sequentially**, one after another, without interleaving. For example:
    - Transaction T1 executes entirely before Transaction T2 begins.
2. **Concurrent Schedule**: A schedule where the **operations of transactions are interleaved**. This can **improve performance** but may lead to **conflicts** or inconsistencies.

Result Equivalence

If two schedules produce the same result after execution, they are said to be result equivalent.

Conflict Equivalence

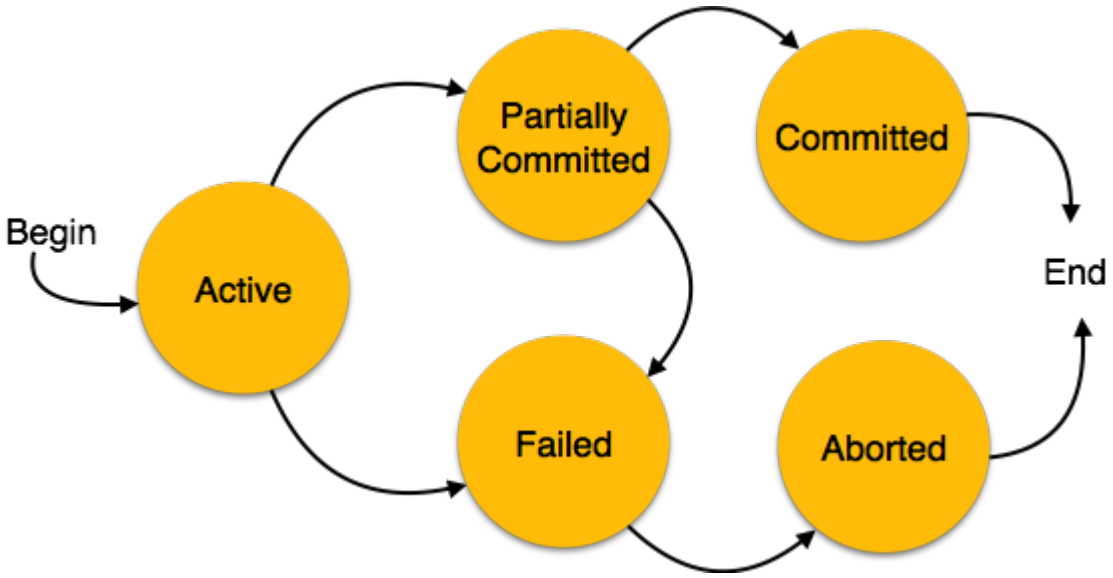
Two schedules would be **conflicting** if they have the following properties –

- 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.

A schedule is called conflict serializability if after s**wapping of non-conflicting operations,** it can transform into a **serial schedule**.



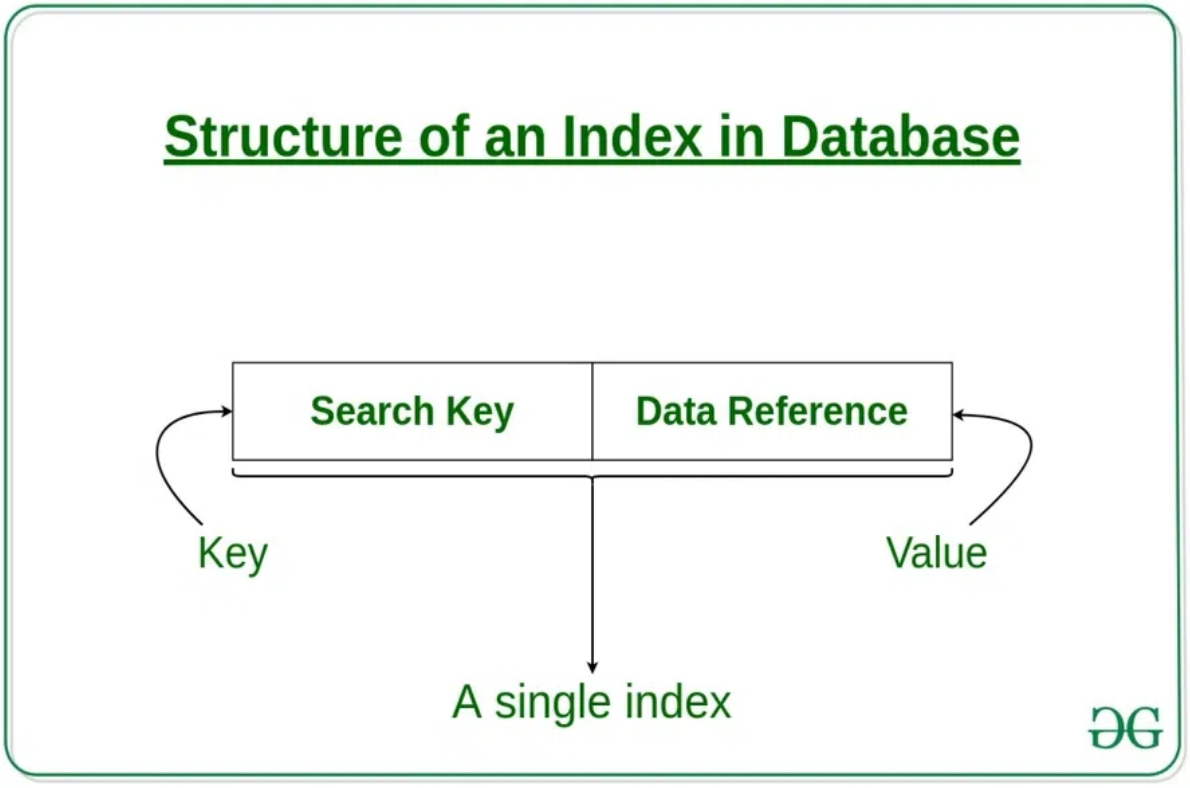
- **Active** – In this state, **the transaction is being executed**. This is the **initial state** of every transaction.
- **Partially Committed** – When a transaction **executes its final operation**, it is said to be in a partially committed state.
- **Failed** – A transaction is said to be in a failed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

Indexing

Indexing is technique for **improving database performance** by **reducing the number** of disk accesses necessary when a **query** is run.

Why Indexing is needed-

1. Speed up access
2. Minimizing number of disk access
3. Less I/O Operation



Deadlock

The Deadlock is a condition in a **multi-user database** environment where transactions are unable to the complete because they are **each waiting for the resources held by other transactions.**