TRANSACTION

- Transaction is a unit consist of logically related operation like(read, write)

For example, you are transferring money from your bank account to your friend's account, the set of operations would be like this:

1. Read your account balance
2. Deduct the amount from your balance
3. Write the remaining balance to your account
4. Read your friend's account balance
5. Add the amount to his account balance
6. Write the new updated balance to his account

This whole set of operations can be called a transaction.

In DBMS, we write the above 6 steps transaction like this:

Lets say your account is A and your friend's account is B, you are transferring 10000 from A to B, the steps of the transaction are:

```
1. R(A);
2. A = A - 10000;
3. W(A);
4. R(B);
5. B = B + 10000;
6. W(B);
```

In the above transaction **R** refers to the **Read operation** and **W** refers to the **write operation.**

Transaction failure in between the operations

The transaction may fail before finishing the all the operations in the set. This can happen due to power failure, system crash etc.

Assume that transaction fail after third operation (see the example above) then the amount would be deducted from your account but your friend will not receive it.

To solve this problem, we have the following two operations.

- **Commit:** If all the operations in a transaction are completed successfully then commit operation help to save those changes to the database permanently.
- **Rollback:** If any one of the operation fails then roll back operation help to restore the database into its original state (before applying transaction).

Even though these operations can help us avoiding several issues that may arise during transaction but they are not sufficient when two transactions are running concurrently. To handle those problems we need to understand database ACID properties.

Transaction property (ACID)

The transaction has the four properties. These are used to maintain consistency in a database, before and after transaction.

Property of Transaction

Atomicity

- The transaction cannot occur partially. Each transaction is treated as one unit. Atomicity ensure that the transaction either completed successfully or not executed at all.
- Transaction control manager component responsible for atomicity.

Atomicity involves the following two operations:

Commit: If a transaction success then all the changes are visible.

Roll back: If a transaction fails then all the changes are not visible.

Example: Let's assume that following transaction T consisting of T1 and T2. A consists of Rs 600 and B consists of Rs 300. Transfer Rs 100 from account A to account B.

T1	**T2**
Read(A)	Read(B)
A:= A-100	Y:= Y+100
Write(A)	Write(B)

After completion of the transaction, A consists of Rs 500 and B consists of Rs 400.

If the transaction T fails after the completion of transaction T1 but before completion of transaction T2, then the amount will be deducted from A but not added to B. This shows the inconsistent database state. In order to ensure correctness of database state, the transaction must be executed in entirety.

Consistency

- It ensure that if the system is consistent before and after transaction
- User /application programmer responsible for consistent state.

For example: The total amount must be maintained before or after the transaction.

1. Total before T occurs = 600+300=900
2. Total after T occurs= 500+400=900

Isolation

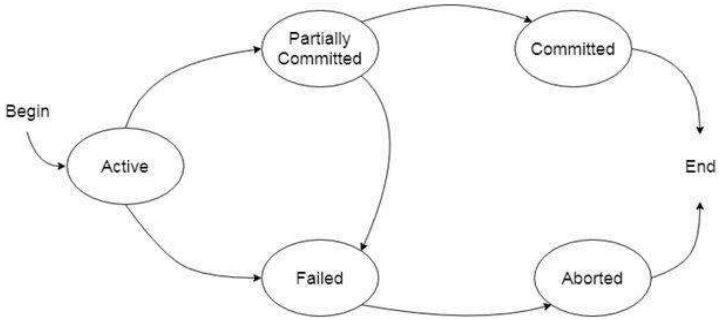
- It ensure that no transaction should be affected by other parallelly executed transaction.
- In isolation, if the transaction T1 is being executed and is using the data item X then that data item can't be accessed by any other transaction T2 until the transaction T1 ends.
- The concurrency control manager component responsible for isolation property.

Durability

- It ensure that all the updates made by transaction must become permanent in DB irrespective of hardware and software failure.
- The recovery manager component is responsibility for durability.

States of Transaction

In a database, the transaction can be in one of the following states -



Active state

- Active state is the first state of the every transaction. In this state, transaction is being executed.
- For example: Insertion or deletion or updating a record is done here. But all the records are still not saved to the database.

Partially committed

- In the partially committed state, a transaction executes its final operation but the data is still not saved to the database.

Committed

A transaction is said to be in a committed state if it executes all its operations successfully. In this state, all the effects are now permanently saved on the database system.

Failed state

- If any of the query made from the database system fails then the transaction is said to be in failed state.
- In the example of total mark calculation, if the database is not able to fire a query to fetch the marks then the transaction will fail to execute.

Aborted – If transaction reached failed state, then the recovery manager rolls back all its write operations on the database to bring the database back to its original state .

Schedule

Transactions are set of operations on database. If multiple transactions are running concurrently, we need to perform the only one operation on the database. This sequence of operations is known as Schedule.