What Is Transaction ?

It is a unit of program execution by the user or application that can access and update different data points.
It is a logical unit of work that contains one or more SQL statements.
The result of all these statements in a transaction either gets completed successfully (all the changes made to the database are permanent) or if at any point any failure happens it gets reversed (all the changes being done are undone.)
Eg: Transfer 70 rupees from your account(X) to your friends account(Y):

Transaction:

1. READ(X)
2. X=X-70
3. WRITE(X)
4. READ(Y)
5. Y=Y+70
6. WRITE(Y)

A transaction can be a whole program, or even a single clause command.

Before we discuss further about the properties of transaction, let's have a quick look at the issues that we might face within a transaction:

- Hardware failures or system crashes.
- Concurrent execution of multiple transactions.

These issues can result in loss of data consistency. In order for that to not happen, We have certain properties of Transaction that we need to follow.

They are called **ACID** properties.

Transaction Properties:
ACID stands for:

- Atomicity
- Consistency
- Isolation
- Durability

Let's understand them briefly with the example of the transaction that we also performed above.

➔ Atomicity:

Now, suppose there are system failures(software or hardware) due to which transaction fails after step 3 and before step 6. The money that was being

transferred will be lost(i.e. It has been deducted from your account but hasn't been added to your friends account), this leads to the inconsistency in the database state.

Therefore, In layman terms, 'all or nothing' property. That is, **if a transaction is to happen either it will be performed entirely or will not be executed at all.**

We know a transaction can have a set of operations in it, but we consider one whole transaction as one unit.

Also it involves two operations:

- ➢ Abort: If we abort the transaction, the changes that might have happened due to the transaction to the database aren't visible.
- ➢ Commit: If we commit the transaction, the changes that might have happened due to the transaction to the database are visible.

➔ Consistency:

Now, the sum of money in the accounts before the transaction should be equal to the sum of money in the accounts after the transaction.

Therefore, we define consistency as the effect of integrity constraints on the database due to which data remains consistent before and after the transaction when it transfers the database from one state to another.

During the transaction, the database can be inconsistent.

➔ Isolation:

Now suppose while this transaction (mentioned above) is taking place and another transaction has started in between and it is accessing the partially updated database.

Now that transaction will encounter an inconsistent database.

Therefore, isolation ensures that the transactions are executing independently, i.e. when one transaction is being done, it won't be interrupted by the other one. Although multiple transactions can happen simultaneously, given that each transaction is unaware of the other concurrently executing transactions.

➔ Durability:

Now suppose, you received a message on your mobile phone stating that Rs. 70 have been debited from your account to transfer to this account.
But your friend states that he hasn't received the money yet.
Now this could happen due to the software or hardware crash, but the updates to the database by the transaction has been made and they won't be lost even after the system broke down.

Therefore, This property ensures all the changes or updates to the database have been recorded and have been stored and will be never lost even if the system crashes.