Transection

A transection is an action or series of action which is perform a single user or application on database. Which read, write and update the content of the database? A transection can be defined as a logical unit of works on database.

A transection must follow the ***ACID*** property:--

Atomicity

Atomicity state that a transection must be consider as an atomic unit. Either all of its operation execute at once or none. The operation of a transection cannot be done partially.

Consistency

The data base must remain in consistent state before and after any transection. If a transaction is completed successfully then the database must goes from one stable state to another stable state.

Isolation

Isolation state that all transection must execute independently. Multiple transection can occur without effect one another.it says that data used by one transection at the time of execution cannot be used by other transection until the first transection is commit and write to the memory permanently.

Durability

Durability ensure that the database remain in consistent state always and any transection committed successfully must be write on the database the database should be capable of holding all of its latest update and changes if the system fails. If a transaction is committing but the change cannot write to the memory due to the system fail then once the system is back on the data must be writing on the system. The consistent state of the database cannot be lost if the system failure occurs. If a transaction is committed then that change must write on the database at any cost.

States of a Transection

Active State

This is the initial state of a transaction. In this state the transaction is being executed

Insert, delete and update of any data but the records or change are not saving or write to the database.

Partially committed

In this state the transection executes its final operation but still the data is not saving to the database.

System gives an alert of final submission or gives a chance to overview of input data before final submission.

Committed

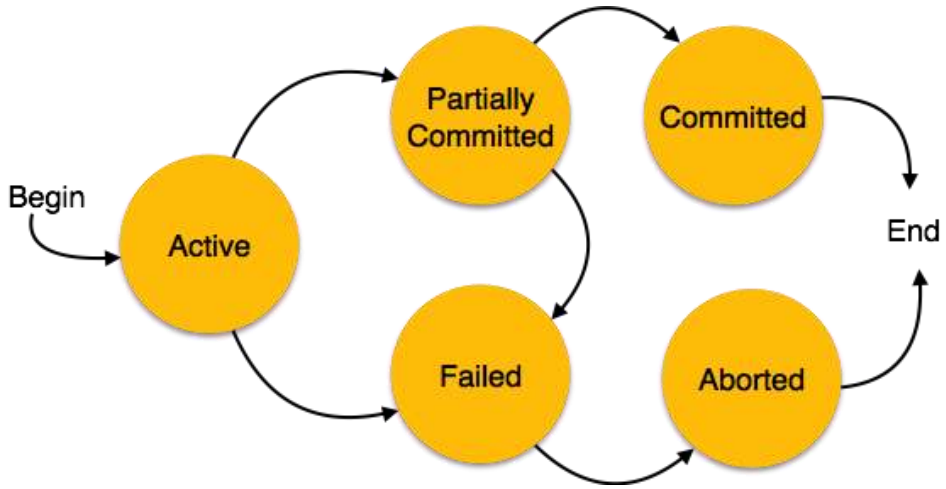
A transection is said to be committed if all of its operation are execute at once and all the effect are permanently save on the database.

Fail state

A transection is said to be fail if any of its checks made by the database recovery system is fail.

Aborted

If the transaction is in fail sate then it must be abort. It the transection fails, if the system goes fail or something wrong happens middle of a transection then the recovery system manager will roll back all its write operation from the database and bring the database in its original state or in consistent state.



Operations of a transection

The main operations of a transaction are

1. Read operation

   In this operation read a particular value from the database and store the buffer in main memory until the transaction is committed or roll back to its initial state?

2. Write Operation

   Write operation is used to write the data on database from buffer after the transaction is committee successfully.

```
R(X);
X = X - 500;
W(X);
```

Let's assume the value of X before starting of the transaction is 4000.

- The first operation reads X's value from database and stores it in a buffer.
- The second operation will decrease the value of X by 500. So buffer will contain 3500.
- The third operation will write the buffer's value to the database. So X's final value will be 3500.

Dead Lock && Roll Back

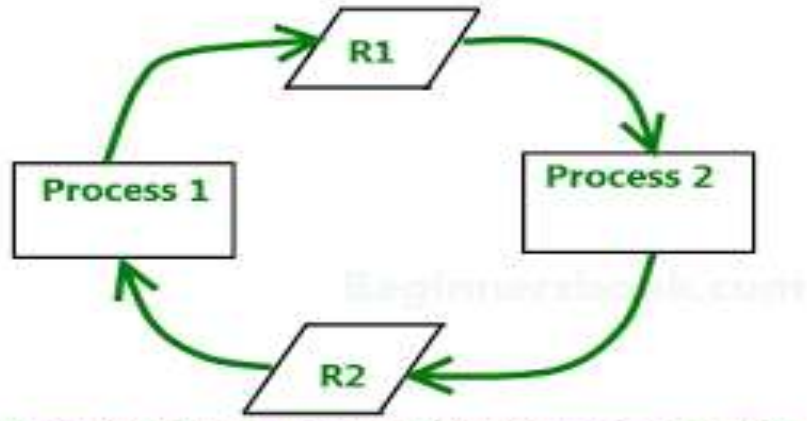
Sometimes due to hardware or software and database failure a transection cannot commit or may fail. Beside that dead lock is one of the main reasons to fail a transaction. And to solve this problem we do Roll back.

Dead lock

Deadlock is a situation in which two or more transaction want to access the same resources and. or two or more transaction waiting for one another to give up locks.

Suppose transection A might hold some rows on account table and need to update some rows on order table to commit the transaction. But transaction B holds the same rows of order table that A require and need to update some rows on account table that A holds.

The transection A cannot complete because B has a lock on order table that A require. And transection B cannot complete because A has a lock on account table that B require. so all the activity comes to a halt state and stop for forever. This situation is call dead lock.



Deadlock Handling Mechanism

There are three mechanisms to handle a deadlock situation:

1. Deadlock avoidance
2. Deadlock detection
3. Deadlock prevention

Deadlock avoidance:

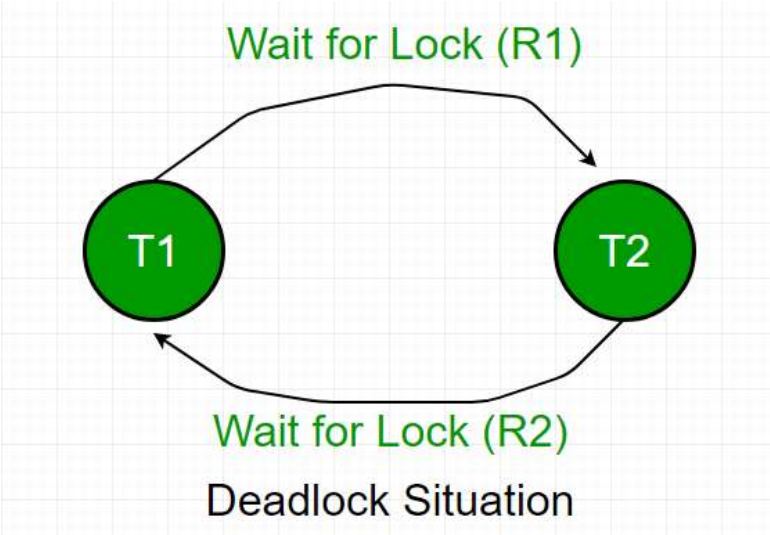
Deadlock avoidance mechanism handles a deadlock situation before it occurs. Design the system in such a way that it is consistence and avoid deadlock. In this system there is an algorithm which always checks for the data which is requested by a transection to lock on. If the requested data is lock on by some other transection then lock manager will decide whether the transection need to put on wait stage or abort it so that deadlock is not occurs.

This approach is useful for small database.

Deadlock detection:

When a transection waits for a long time for a lock on some resources the database management system must checks whether the transaction involves on deadlock situation or not.

One way to detect deadlock is using **Wait-for-graph**. In this method a graph is grown based on transections and their lock on resources. If the graph became a close loop or cycle then there is a deadlock.



Useful for small database

Deadlock prevention

Is deadlock situation happen we can prevent it using two methods

1. Wait- die scheme
2. Wound Wait scheme

Wait – die scheme

If a transection request for lock on some resources which is already locked by some other transection then DBMS quickly checks the timestamp of both the transection and put the older transection in a wait state until the new transection is committed and resources is released.

Wound Wait Scheme

If a older transection request lock on some resources which is already held by a younger transection then the older transection force the younger transection to kill the transection and restart it after a millisecond.