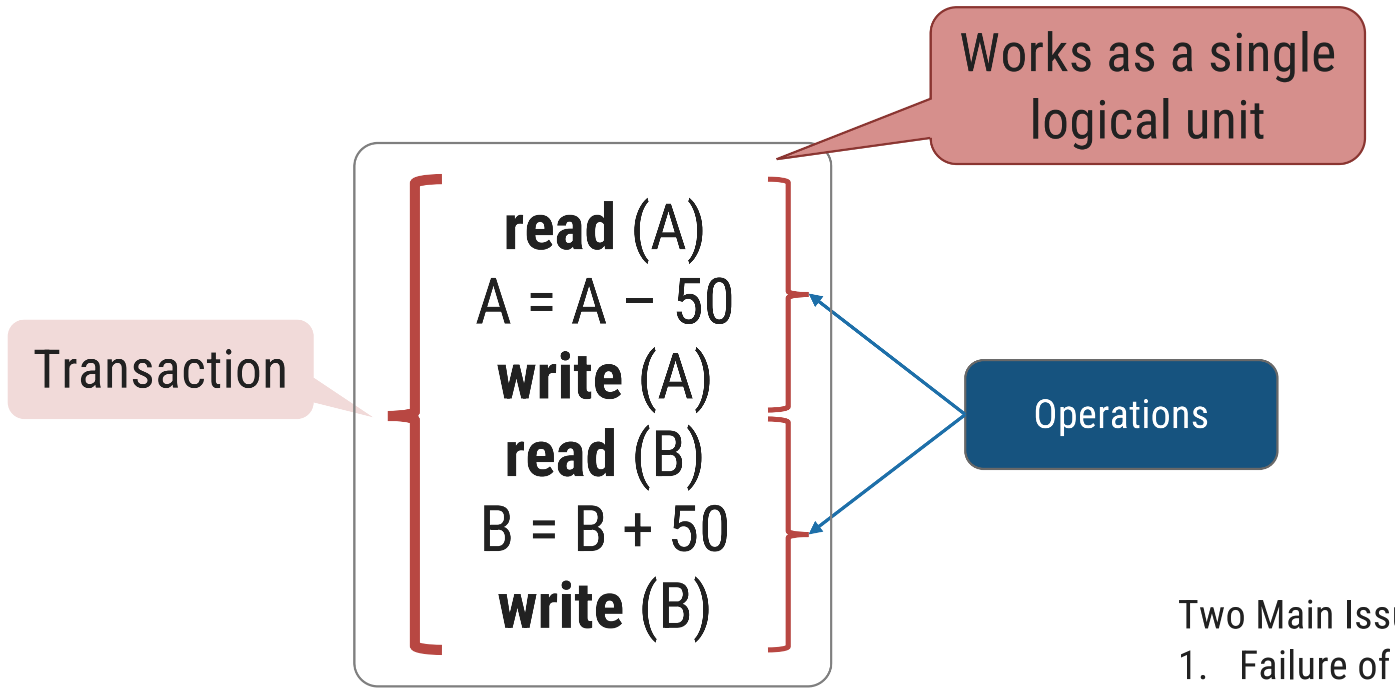
Transaction Processing

What is transaction?

What is transaction?

- A transaction is a **sequence of operations performed as a single logical unit of work**.
- A transaction is a **logical unit of work that contains one or more SQL statements**.
- Example of transaction: Want to transfer Rs. 50 from Account-A to Account-B



Two Main Issues

1. Failure of various kind
2. Concurrent execution of multiple transactions

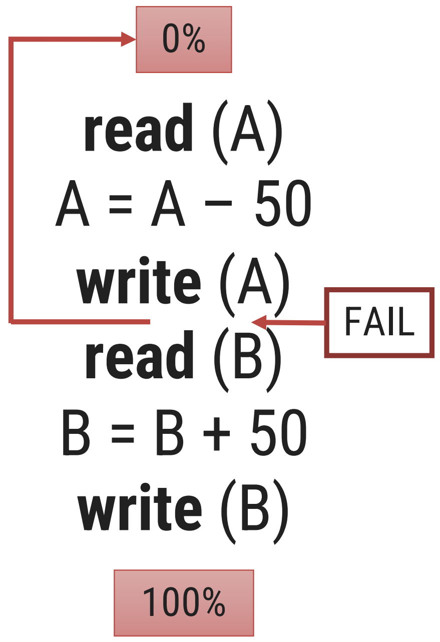
ACID properties of transaction

ACID properties of transaction

- **A**tomicity (**Either transaction execute 0% or 100%**)
- **C**onsistency (**Database must remain in a consistent state after any transaction**)
- **I**solation (**Intermediate transaction results must be hidden from other concurrently executed transactions**)
- **D**urability (**Once a transaction completed successfully, the changes it has made into the database should be permanent**)

ACID properties of transaction (Atomicity)

- This property states that a **transaction must be treated as an atomic unit**, that is, **either all of its operations are executed or none**.
- **Either transaction execute 0% or 100%**.
- For example, consider a transaction to transfer Rs. 50 from account A to account B.
- In this transaction, if Rs. 50 is deducted from account A then it must be added to account B.



ACID properties of transaction (Consistency)

- The **database must remain in a consistent state** after any transaction.
- 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.
- In our example, total of A and B must remain same before and after the execution of transaction.

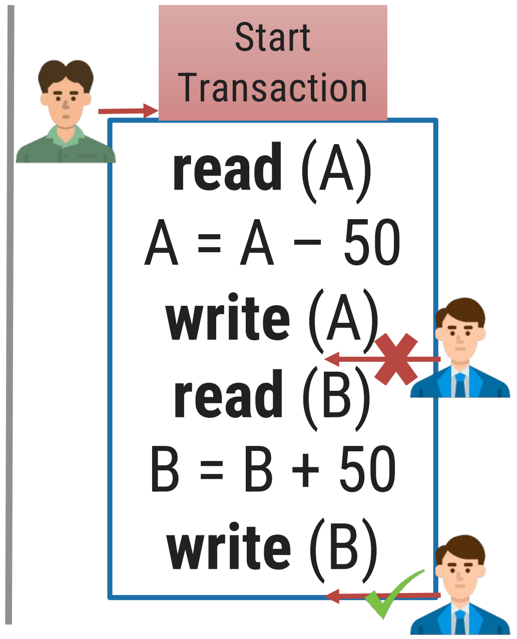
A=500, B=500
A+B=1000

read (A)
A = A – 50
write (A)
read (B)
B = B + 50
write (B)

A=450, B=550
A+B=1000

ACID properties of transaction (Isolation)

- **Changes occurring in a particular transaction will not be visible to any other transaction until it has been committed.**
- **Intermediate transaction results must be hidden** from other concurrently executed transactions.
- In our example once our transaction starts from first step (step 1) its result should not be access by any other transaction until last step (step 6) is completed.



ACID properties of transaction (Durability)

- After a transaction completes successfully, the **changes it has made to the database persist (permanent)**, even if there are system failures.
- Once our transaction completed up to last step (step 6) its result must be stored permanently. It should not be removed if system fails.

A=500, B=500

read (A)
A = A – 50
write (A)
read (B)
B = B + 50
write (B)

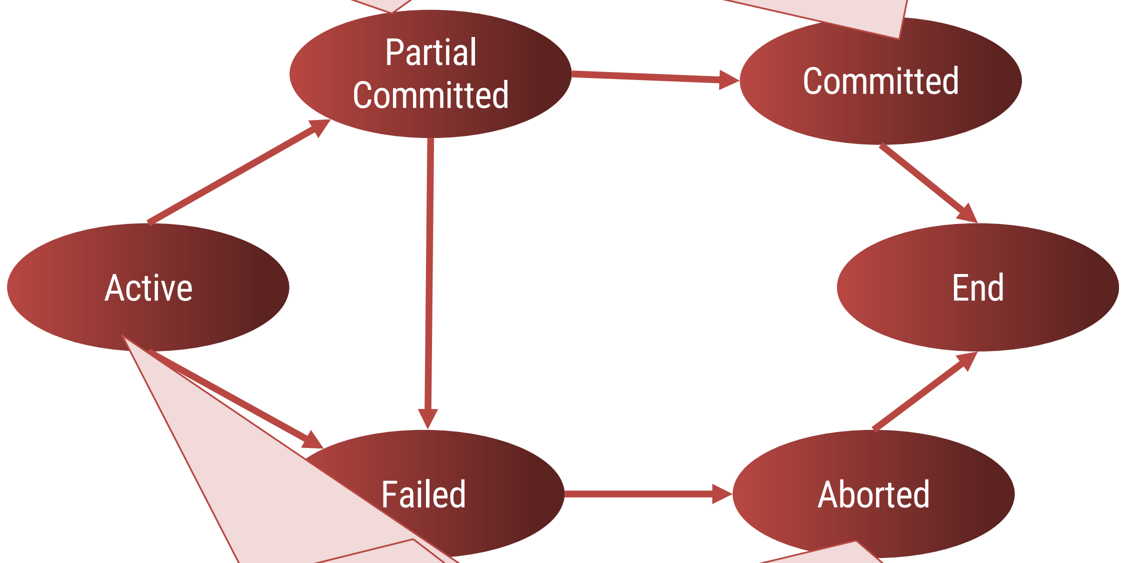
A=450, B=550

These values must be stored permanently in the database

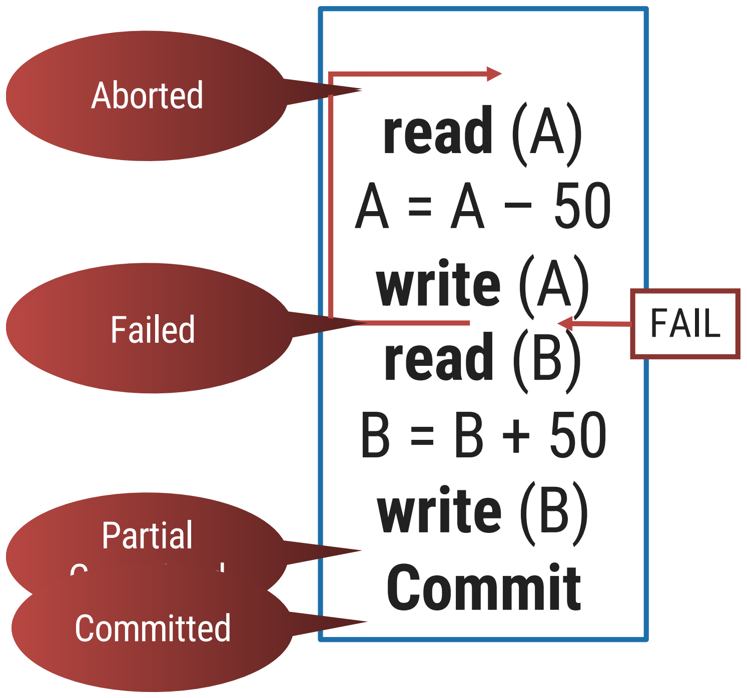
Transaction State Diagram \ State Transition Diagram

Transaction State Diagram \ State Transition Diagram

The transaction enters in this state after successful completion of the transaction.
We cannot abort or rollback a committed transaction.



The state after the transaction has been rolled back and the database has been restored to its state prior to the start of the transaction.



```
read (A)
A = A − 50
write (A)
read (B)
B = B + 50
write (B)
Commit
```

Transaction State Diagram \ State Transition Diagram

- Active
  - This is the **initial state**.
  - The transaction **stays in this state while it is executing**.
- Partial Committed
  - When a transaction **executes its final operation/ instruction**, it is said to be in a partially committed state.
- Failed
  - Discover that **normal execution can no longer proceed**.
  - Once a transaction **cannot be completed**, any **changes that it made must be undone rolling it back**.
- Committed
  - The transaction enters in this state **after successful completion of the transaction** (after committing transaction).
  - We **cannot abort or rollback a committed transaction**.
- Aborted
  - The state after the **transaction has been rolled back** and the **database has been restored to its state prior to the start of the transaction**.