Transaction Management

Transaction Concepts

A Banking Example

- Ada plans to transfer $1000 from her saving account to her chequing account:
  - Debit *$1000* from the **saving** account;
  - Credit *$1000* to the **chequing** account;
  - Ada's total balance should remain the same.
- Implementation in a database:

  UPDATE *account* SET *balance=balance-1000*
  WHERE *user="Ada"* AND *type="saving"*
  UPDATE *account* SET *balance=balance+1000*
  WHERE *user="Ada"* AND *type="chequing"*

What If An Error Happens?

UPDATE *account* SET *balance=balance-1000*
WHERE *user=“Ada”* AND *type=“saving”*

What if a system failure happens here?

UPDATE *account* SET *balance=balance+1000*
WHERE *user=“Ada”* AND *type=“chequing”*

- Ada’s total balance is *$1000* less.
  - Ada should not lose *$1000* due to the system failure!

Transaction

- The two updates should be bounded into **a unit**: either both operations succeed, or none of them take effect.
  - Ada's total balance should be maintained consistently.
- **Transaction:** a unit of program execution that accesses and possibly updates various data.

```
UPDATE account SET balance=balance-1000
WHERE user="Ada" AND type="saving"
UPDATE account SET balance=balance+1000
WHERE user="Ada" AND type="chequing"
```

Consistency of Transactions

- Consistency requirement: constraints (business rules) that should be respected by transactions.
  - Example: if an amount is transferred from one account to another, the total balance of the two accounts should not change.
- Execution of a transaction independently preserves the consistency of the database.

Atomicity of Transactions

- In a transaction, either all operations of the transaction are reflected properly in the database, or none are.
  - In a fund transfer transaction, either the fund is transferred (i.e., both accounts are adjusted properly) or not at all (i.e., both accounts are not updated).
- A database may have an inconsistent state at some point, but an inconsistent state should not be visible to users.

Ensuring Atomicity

- What if a system failure happens during fund transfer?
  - Recover the balances of accounts before the transaction.
- A database system keeps track of the old consistent values and restores those values if a transaction fails.

Durability of Transactions

- If a fund transfer transaction succeeds, even if a system failure later should not lead to the loss of the transaction effect.
- Once a transaction completes successfully, all the updates that it carried out on the database persist, even if there is a system failure after the transaction completes execution.

Ensuring Durability

- Using a reliable data storage (e.g., hard disk or tape)
- The updates carried out by the transaction have been written to disk before the transaction completes.
- Information about the updates carried out by the transaction and written to disk is sufficient to enable the database to reconstruct the updates when the database system is restarted after failure.

Concurrent Transactions

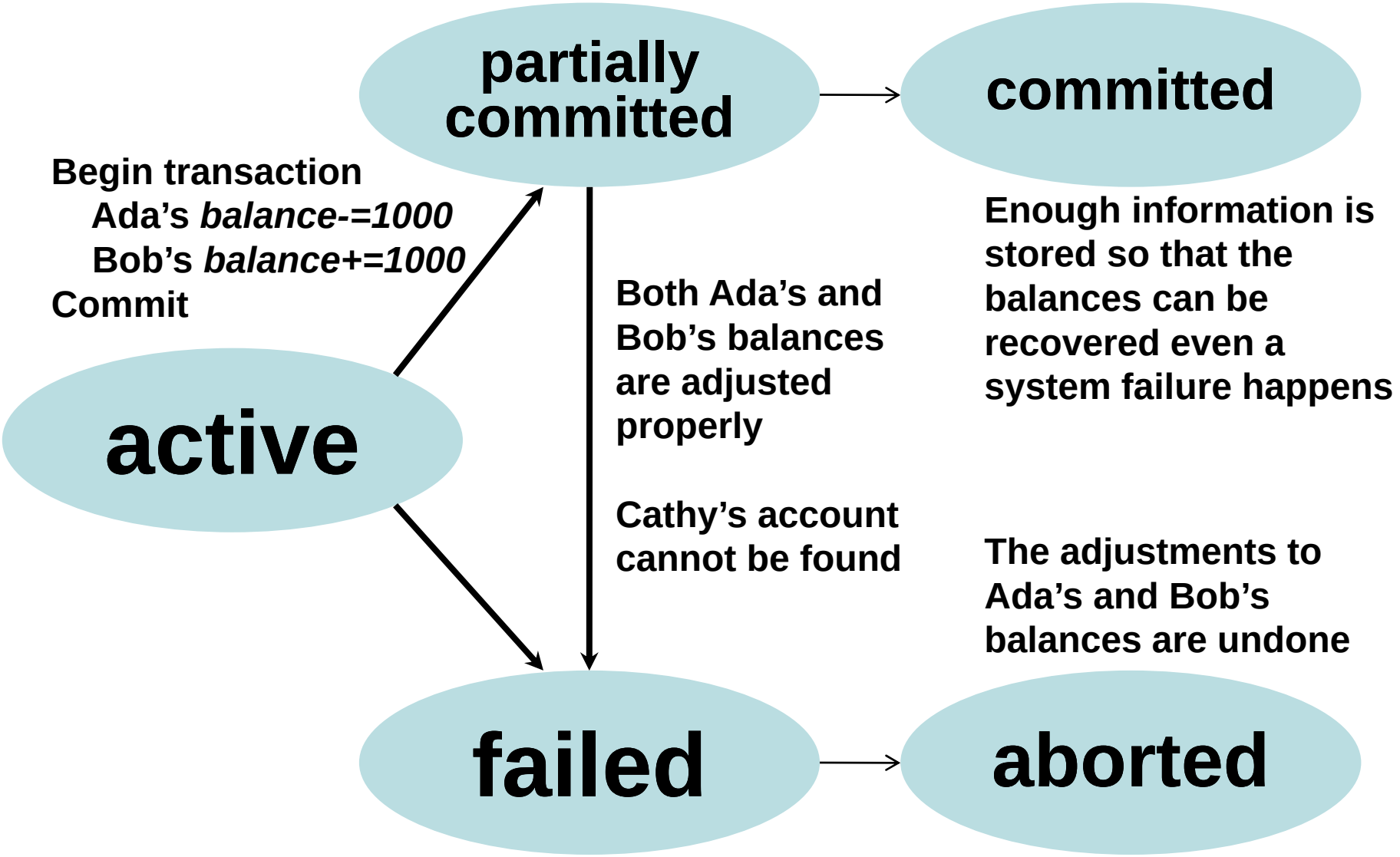
- Ada transfers *$1000* to Bob:
  - Let *v* be the balance of Bob's account
  - Set *v=v+1000*
- Cathy transfers *$500* to Bob
- What if the two transactions interleave?

Ada's transfer	**Cathy's transfer**
Get *v* (*v=800*)	
	Get *v* (*v=800*)
Set *v=v+1000=1800*	
	Set *v=v+500=1300*

Isolation of Transactions

- The concurrent execution of transactions results in a system state that is equivalent to a state that could have been obtained had these transactions executed one at a time in some order.
- One naïve solution: executing transactions serially – one transaction at a time
  - Concurrency is preferred due to performance concern.

State Diagram of a Transaction



Transaction States

- **Active:** the initial state; the transaction stays in this state while it is executing
- **Partially committed:** after the final statement has been executed
- **Failed:** after the discovery that normal execution can no longer proceed
- **Aborted:** after the transaction has been rolled back and the database restored to its state prior to the start of the transaction
- **Committed:** after successful completion

Handing Failed Transactions

- Restart the transaction: can be done only if no internal logical error
  - Example: the disk storing Cathy's account fails, the transaction can be restarted after the disk is recovered
- Kill the transaction: some internal logical error (the application program needs to be revised), input was bad, the desired data were not found
  - Example: Bob does not have an account at all

To-Do-List

- Do a research on the ACID (that is, atomicity, consistency, isolation, durability) properties of transactions.
- Given an example of transactions other than the ones we see in the course notes. Illustrate the ACID properties of transactions and the states of transactions using the example.