Selected Database Issues: Security

- **Security** refers to the protection of the database against unauthorized access, either intentional or accidental.

- **Access Control** includes:
  - Identification
  - Authentication
  - Authorization
  - Audit
  - Views
  - Encryption

Access Control Mechanisms

- **Identification** is how a user tells a system who he or she is (for example, by using a username).

- **Authentication** is the process of verifying a user's claimed identity (for example, by comparing an entered password to the password stored on a system for a given username).

- **Authorization** the granting of a right or privilege that enables a subject to have legitimate access to a system or a system’s object.

- **Accountability** uses such system components as *audit trails* and *logs* to associate a subject with its actions. Audit trails and logs are important for detecting security violations.

Views

A virtual or derived relation that is dynamically created from the underlying base relation(s) when required.

Pros:

- Improved security
- Data independence
- Convenience

Cons:

- Update restriction

Transaction Support

Transaction: An action, or series of actions, carried out by a single user or application program, which reads or updates the contents of the database.

Properties of a Transaction:

Atomicity:

- ‘All or nothing’ property. Either performed in its entirety or is not performed at all.
- Responsibility of recovery subsystem of the DBMS

Consistency:

- Transformation of database from one consistent state to another consistent state.
- Responsibility of DBMS and application developers.

Properties of a Transaction (Cont'd)

Isolation:

- Transactions should execute independently of one another.
- Partial effects of incomplete transactions should not be visible to other transactions.
- Responsibility of concurrency subsystem of the DBMS

Durability:

- Effects of successful transactions should store permanently in the database and must not be lost of a subsequent failure.
- Responsibility of recovery subsystem.

Concurrency Control

The process of managing simultaneous operations on the database without having them interfere with one another.

Motivation

- Data needs to be shared among many users in a DBMS environment

- If users are only reading the data, concurrent access is easy. However, if two users access the same data and at least one is updating data, there may be interference that results inconsistencies.

- Overlapping I/O and CPU activity can increase ‘system throughput’.

- Interleaving a short transaction with a long transaction allows short transaction to complete quickly

Schedule

A schedule is a list of actions (reading, writing, aborting, or committing) from a set of transactions.

Serial Schedule

If the actions of different transactions are not interleaved i.e., transactions are executed from start to finish, one by one.

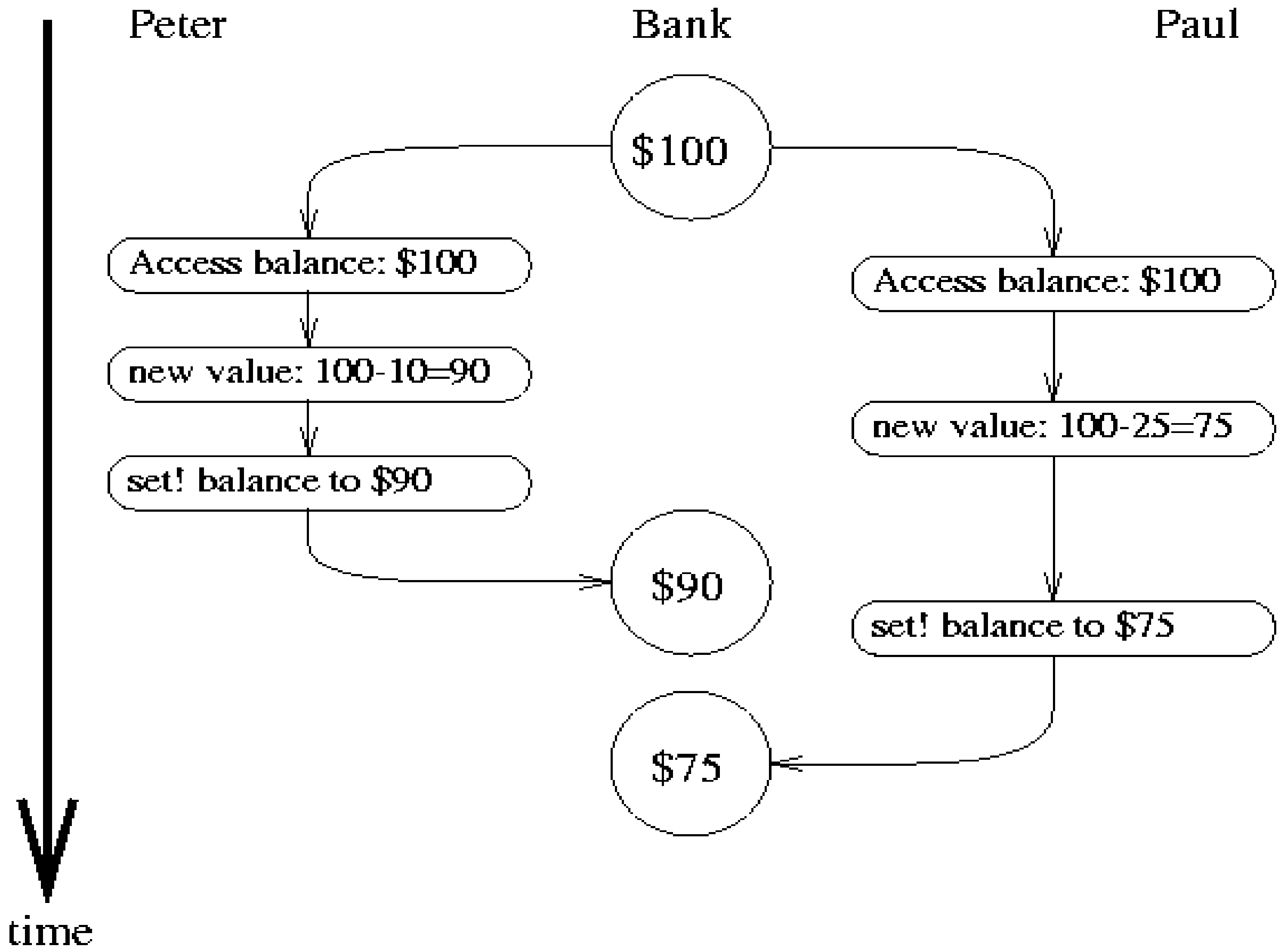
Non Serial Schedule

A schedule where the operations from a set of concurrent transactions are interleaved.

Serializability

If a set of transactions executes concurrently and the non-serial schedule produces the same result as some serial execution.

Timing diagram showing how interleaving the order of events in two banking withdrawals can lead to an incorrect final balance.



Anomalies Due to Interleaved Execution

- **Lost Update Problem**

An apparently successful operation by one user is overwritten by another user.

- **Uncommitted Dependency Problem**

One transaction is allowed to see the intermediate results of another transaction before it has committed.

- **Inconsistent Analysis Problem**

A transaction reads several values from the database but a second transaction updates some of them during the execution of the first transaction.

The Lost Update Problem (Overwriting Uncommitted Data)

Time	T1	T2	bal_x
t1		begin_transaction	100
t2	begin_transaction	read (bal_x)	100
t3	read (bal_x)	bal_x=bal_x+100	100
t4	bal_x=bal_x-10	write(bal_x)	200
t5	write(bal_x)	commit	90
t6	commit		90

If these transactions are executed serially, one after another without interleaving of operations, the final balance would be $190 no matter which transaction is performed first.

The Uncommitted Dependency / Dirty Read Problem

Time	T3	T4	bal_x
t1		begin transaction	100
t2		read (bal_x)	100
t3		bal_x=bal_x+100	100
t4	begin_ transaction	write(bal_x)	200
t5	read(bal_x)	..	200
t6	bal_x =bal_x-10	rollback	100
t7	write(bal_x)		190
t8	commit		190

T4 updates bal to $200 but then rollbacks to $100. In the mean time T3 has read the new value of $200 and decrements it by $10 giving a wrong bal of $190 instead of $90.

The Inconsistent Analysis Problem

Time	T5	T6	bal_x	bal_y	bal_z	sum
t1		begin_ transaction	100	50	25	
t2	begin_ transaction	sum=0	100	50	25	0
t3	read(bal_x)	read(bal_x)	100	50	25	0
t4	bal_x=bal_x-10	sum=sum+bal_x	100	50	25	100
t5	write(bal_x)	read(bal_y)	90	50	25	100
t6	read(bal_z)	sum=sum+bal_y	90	50	25	150
t7	bal_z=bal_z+10		90	50	25	150
t8	write(bal_z)		90	50	35	150
t9	commit	read(bal_z)	90	50	35	150
t10		sum=sum+bal_z	90	50	35	185
t11		commit	90	50	35	185

Nonrepeatable (Fuzzy) Read
When a transaction T rereads a data item it has previously Read, but in between, another transaction has modified it. Thus, T receives two different values for the same data item.

Phantom Read
If a transaction T executes a query that retrieves a set of tuples from a relation satisfying a predicate, re-executes the query at a later time but finds that the retrieved set contains an additional (phantom) tuple that has been inserted by another transaction in the meantime.

Concurrency Control Techniques

Pessimistic Method

- Locking

Optimistic Method

- Timestamping

Locking Method

A procedure used to control concurrent access to data. When one transaction is accessing the database, a lock may deny access to other transactions to prevent incorrect results.

It is the most widely used approach to ensure serializability of concurrent transactions.

Types of Locks

Shared Lock

If a transaction has a shared lock on a data item, it can read the item but not update it.

Exclusive Lock

If a transaction has an exclusive lock on a data item, it can both read and update the item.

Two-Phase Locking (2PL)

A transaction follows the two-phase locking protocol if all locking operations precede the first unlock operation in the transaction.

Every transaction can be divided into two phases:

- Growing phase
- Shrinking phase

Rules:

1. A transaction must acquire a lock on an item before operating on the item. The lock may be read or write, depending on the type of access method.

2. A transaction cannot request additional locks once it releases any locks.

Preventing the Lost Update Problem Using 2PL

Time	T1	T2	bal_x
T1		begin_ transaction	100
T2	begin_ transaction	write_lock(bal_x)	100
T3	write_lock(bal_x)	read(bal_x)	100
T4	wait	bal_x=bal_x+100	100
T5	wait	write(bal_x)	200
T6	wait	commit/ unlock(bal_x)	200
T7	read(bal_x)		200
T8	bal_x=bal_x-10		200
T9	write(bal_x)		190
T10	commit/unlock(bal_x)		190

Preventing the Uncommitted Dependency Problem Using 2PL

Time	T3	T4	balx
T1		begin_ transaction	100
T2		write_lock(bal_x)	100
T3		read(bal_x)	100
T4	begin_ transaction	bal_x= bal_x+100	100
T5	write_lock(bal_x)	write(bal_x)	200
T6	wait	commit/unlock(bal_x)	100
T7	read(bal_x)		100
T8	bal_x= bal_x-10		100
T9	write(bal_x)		90
T10	commit/unlock(bal_x)		90

Preventing the Inconsistent Analysis Problem Using 2PL

Time	T5	T6	bal$_x$	bal$_y$	bal$_z$	sum
T1		begin_ transaction	100	50	25	
T2	begin_ transaction	sum=0	100	50	25	0
T3	write_lock(bal$_x$)		100	50	25	0
T4	read(bal$_x$)	read_lock(bal$_x$)	100	50	25	0
T5	bal$_x$= bal$_x$-10	wait	100	50	25	0
T6	write(bal$_x$)	wait	90	50	25	0
T7	write_lock(bal$_z$)	wait	90	50	25	0
T8	read(bal$_z$)	wait	90	50	25	0
T9	bal$_z$= bal$_z$+10	wait	90	50	25	0
T10	write(bal$_z$)	wait	90	50	35	0
T11	commit/unlock(bal$_x$)	wait	90	50	35	0

Time	T5	T6	bal_x	bal_y	bal_z	sum
T12		read(bal_x)	90	50	35	0
T13		sum= sum+bal_x	90	50	35	90
T14		read_lock(bal_y)	90	50	35	90
T15		read(bal_y)	90	50	35	90
T16		sum= sum+bal_y	90	50	35	140
T17		read_lock(bal_z)	90	50	35	140
T18		read(bal_z)	90	50	35	140
T19		sum= sum+bal_z	90	50	35	175
T20		commit/unlock(bal_x,bal_y,bal_z)	90	50	35	175