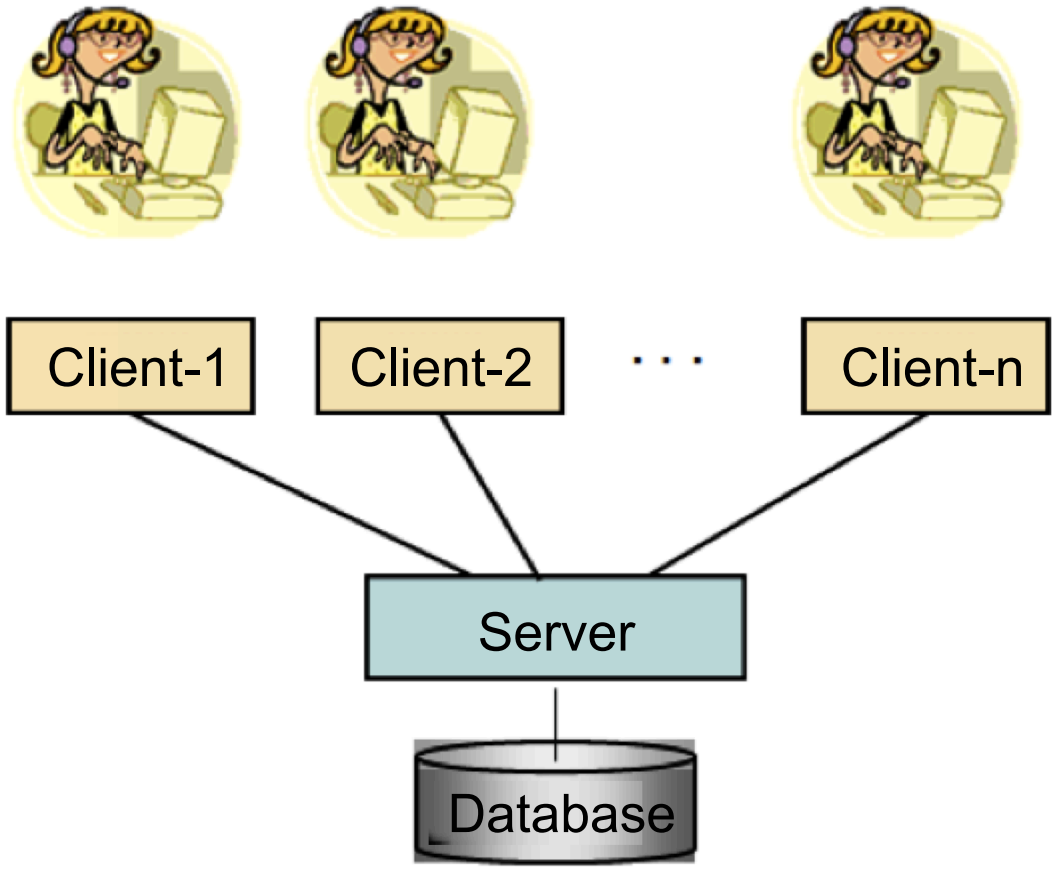
DBTech VET / DBTechNet
SQL Transactions*
Pilot Course Offering
Athens & Thessaloniki, Greece May 20 - June 13, 2013

Lecture No. 3

D.A. Dervos, C. Skourlas, M. Laiho, J.F. Aldana-Montes
www.dbtechnet.org

*

Concurrently executing transactions



Martti Laiho

IF

- The DBMS is lacking the support of basic concurrency control (CC) services, or
- The programmer is lacking the knowledge of how to make proper use of the DBMS supported CC services

IF ... THEN

IF

- The DBMS is lacking the support of basic concurrency control (CC) services, or
- The programmer is lacking the knowledge of how to make proper use of the DBMS supported CC services

THEN

Data update operations may end up corrupting the DB data content

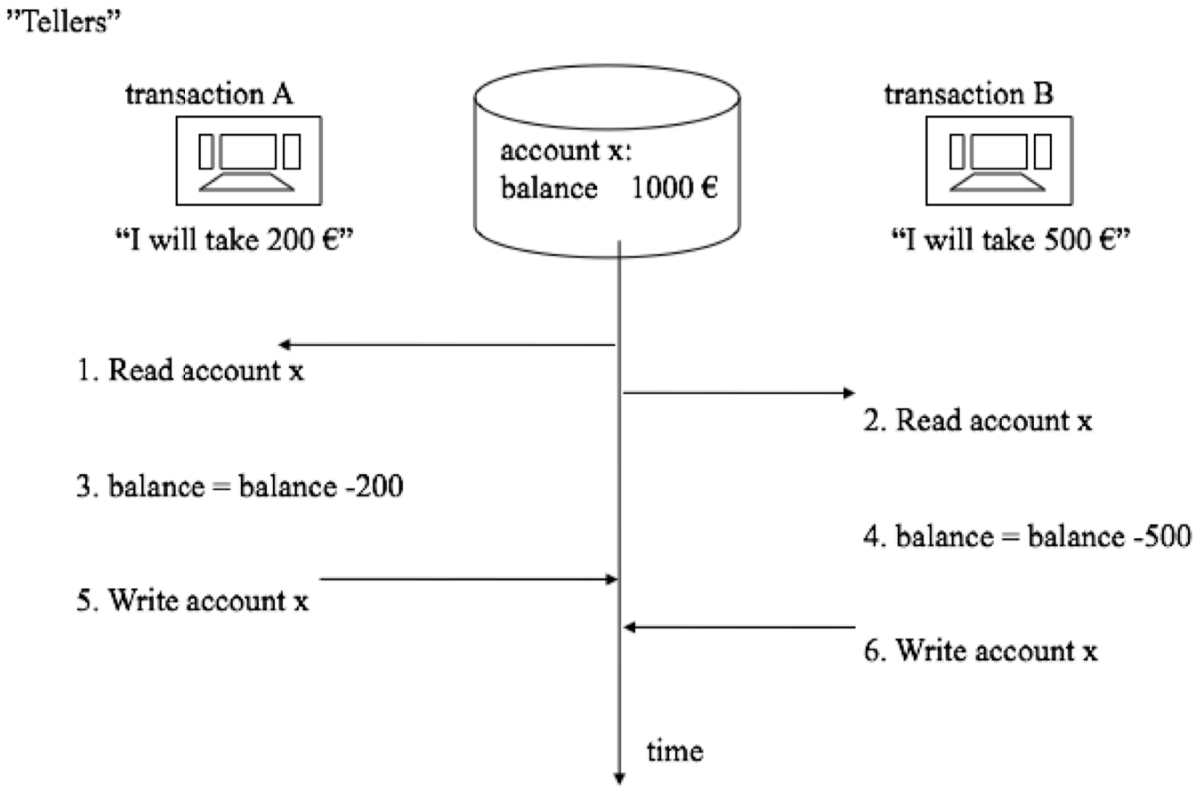
Concurrency problems (anomalies)

- Lost update
- Dirty read
- Non-repeatable read
- Phantom read

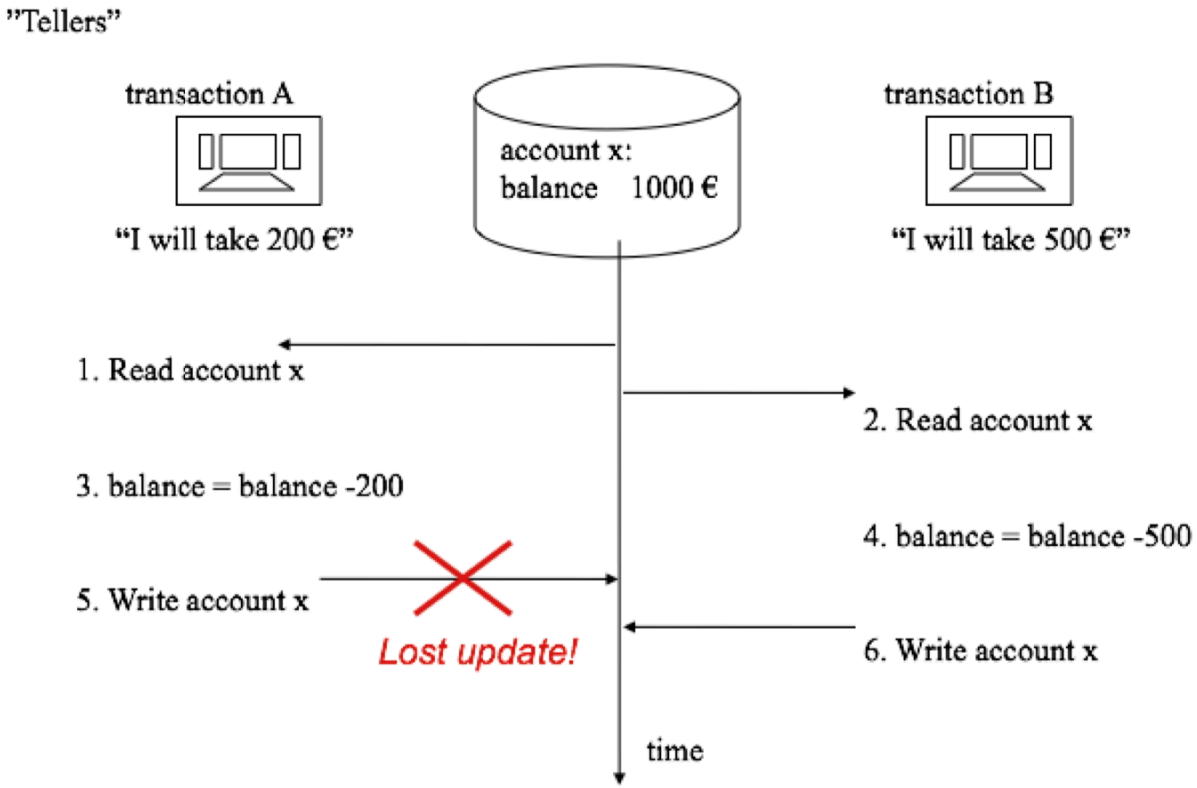
Concurrency problems (anomalies)

- Lost update
- Dirty read
- Non-repeatable read
- Phantom read

The lost update problem



The lost update problem

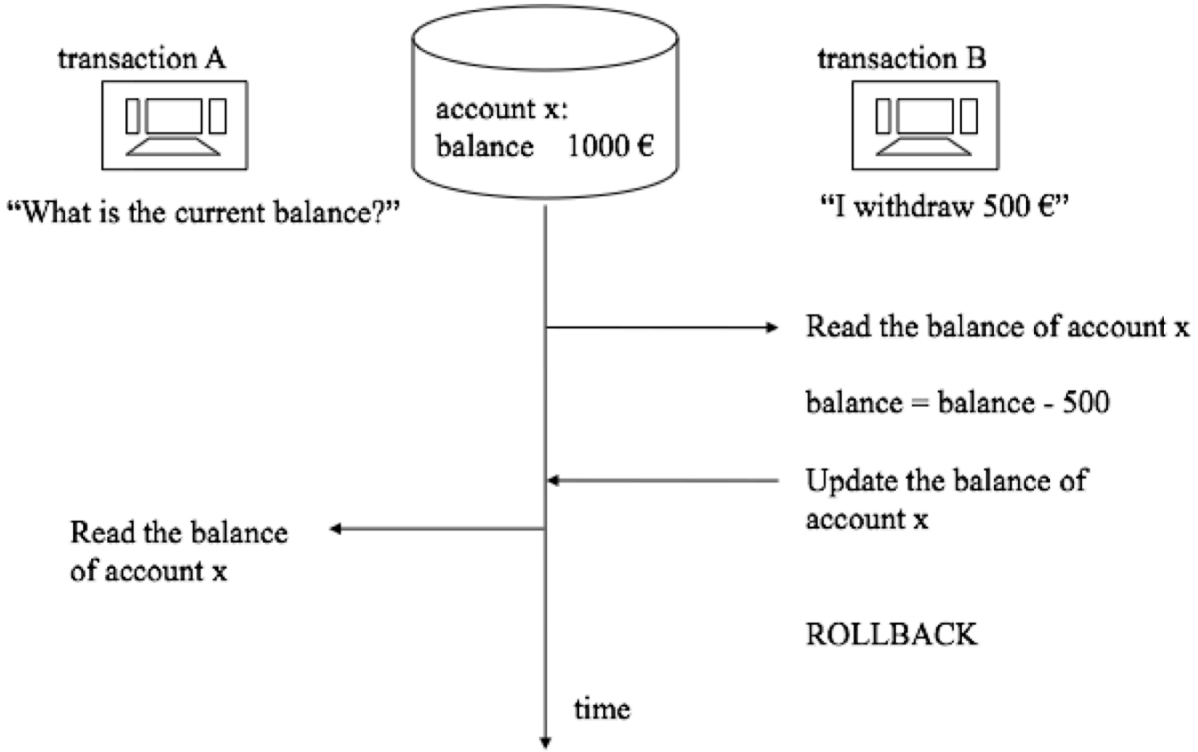


Concurrency problems (anomalies)

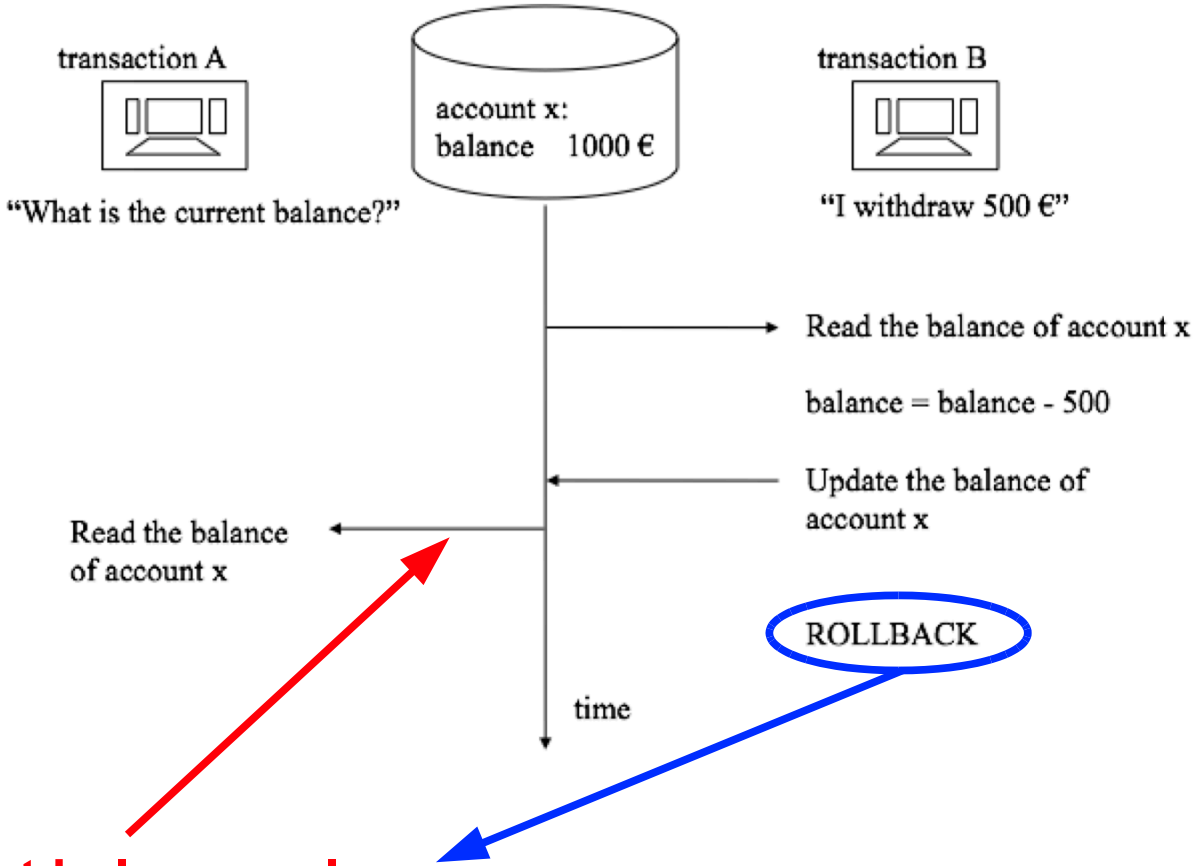
Lifelong Learning Programme

- Lost update
- Dirty read
- Non-repeatable read
- Phantom read

The dirty read problem



The dirty read problem



account balance value that never existed!

Lock-based concurrency control

Compatibility of S and X locks:

Lock of transaction A to object o

Lock request of transaction B to object o	Shared	eXclusive
Shared	**Grant**	**Wait !**
eXclusive	**Wait !**	**Wait !**

- S-lock grants read access to object
- X-lock grants write access to object
- X-lock request after getting S-lock is called as lock promotion

Lock-based concurrency control

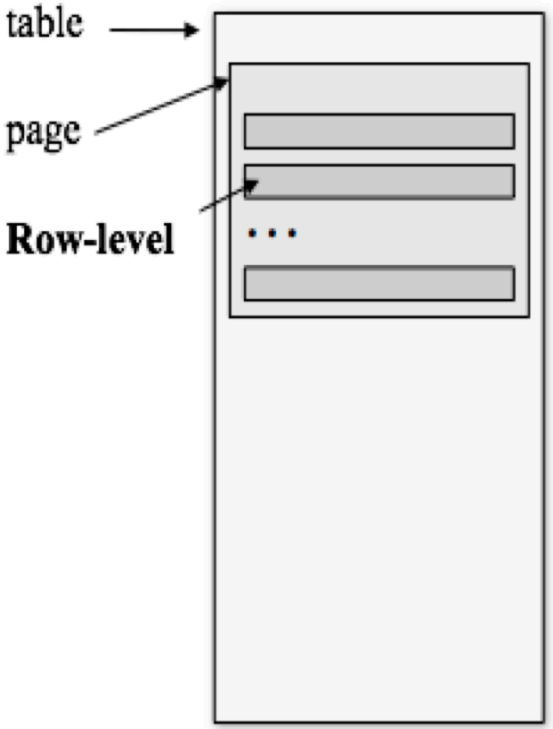
Compatibility of S and X locks:

Lock of transaction A to object o

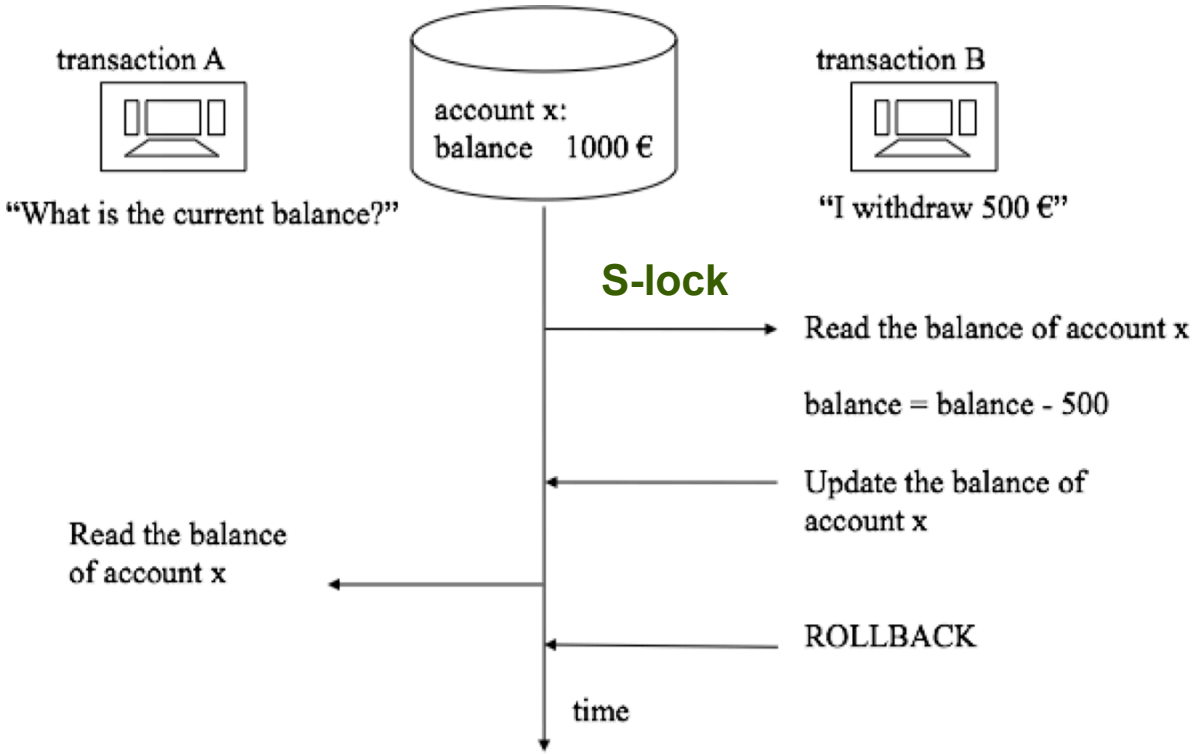
Lock request of transaction B to object o	Shared	eXclusive
Shared	**Grant**	**Wait !**
eXclusive	**Wait !**	**Wait !**

- S-lock grants read access to object
- X-lock grants write access to object
- X-lock request after getting S-lock is called as lock promotion

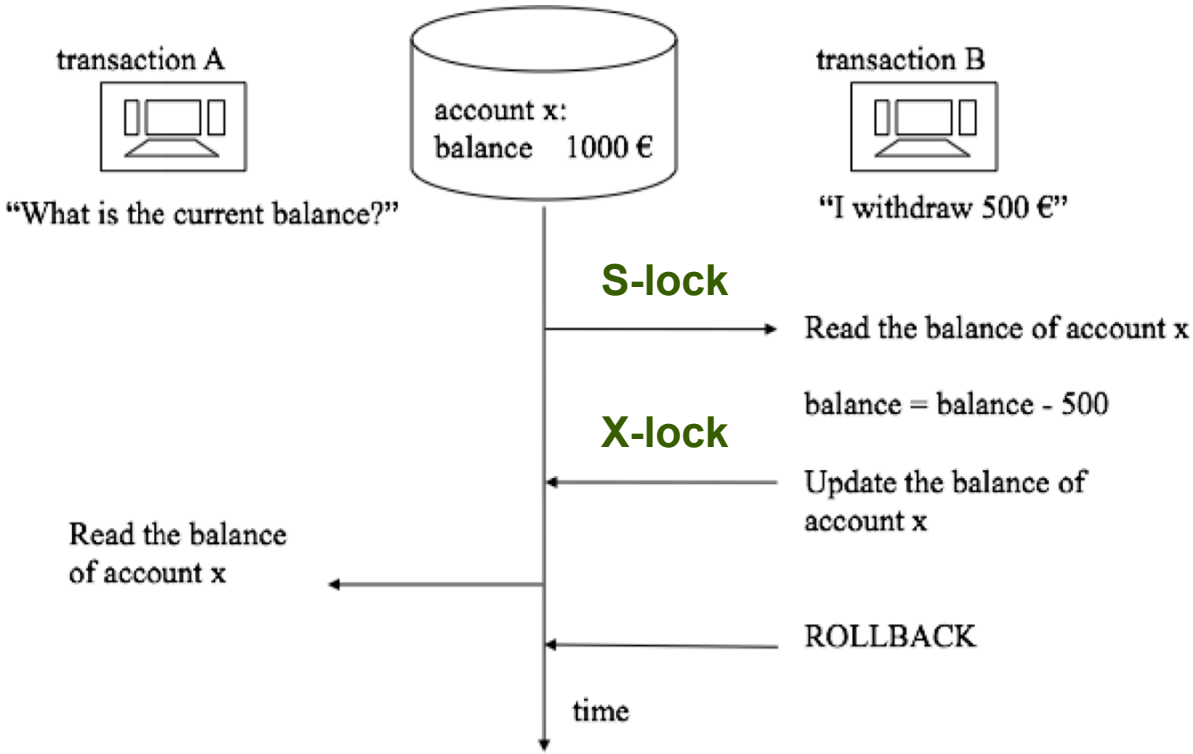
Locking granularity:



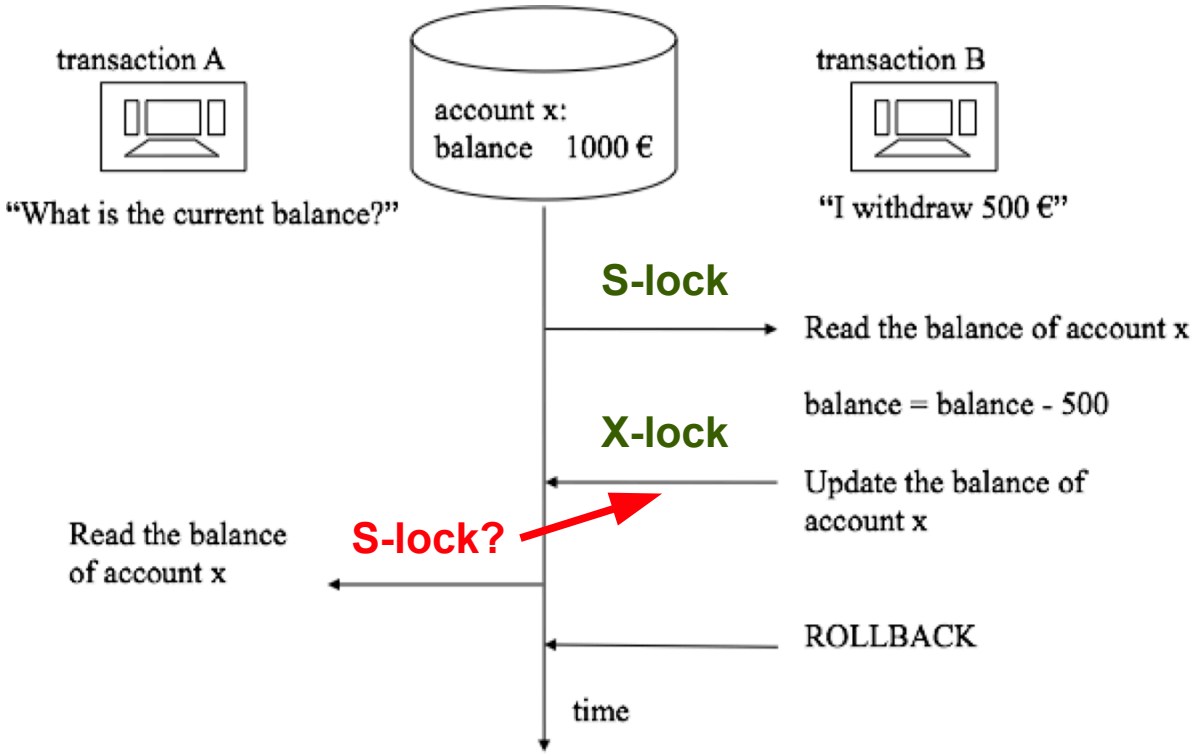
Dirty read with locks...



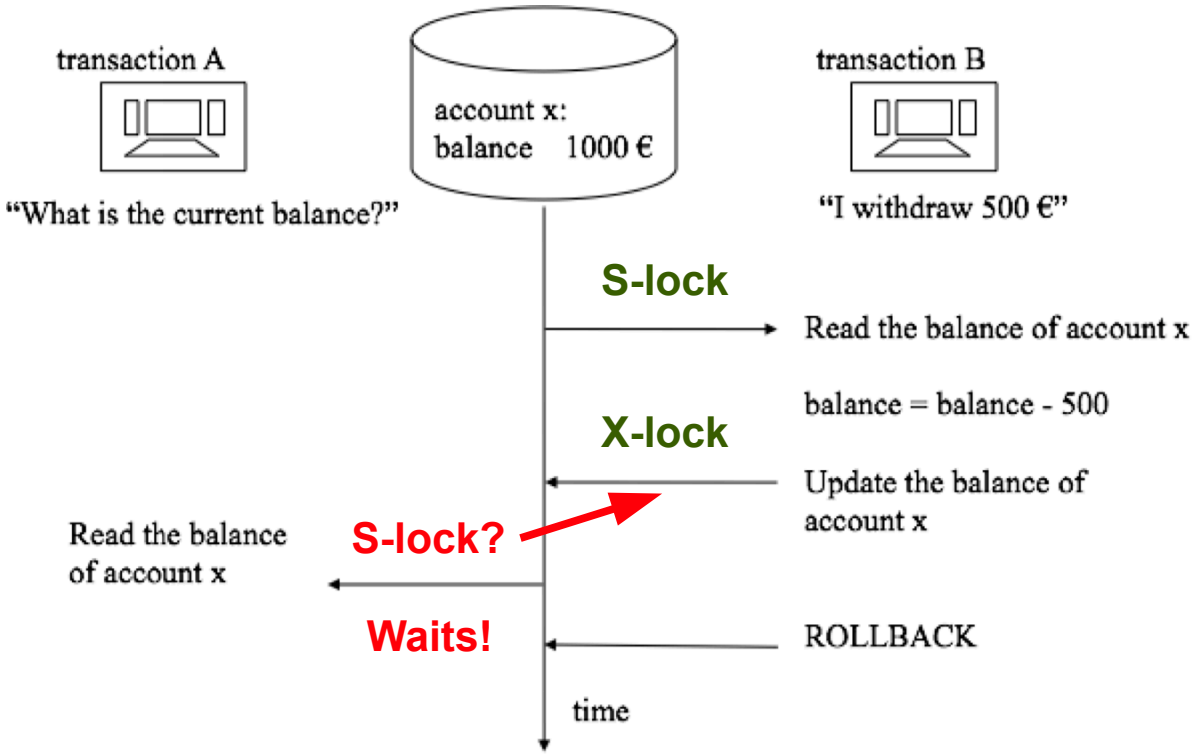
Dirty read with locks...



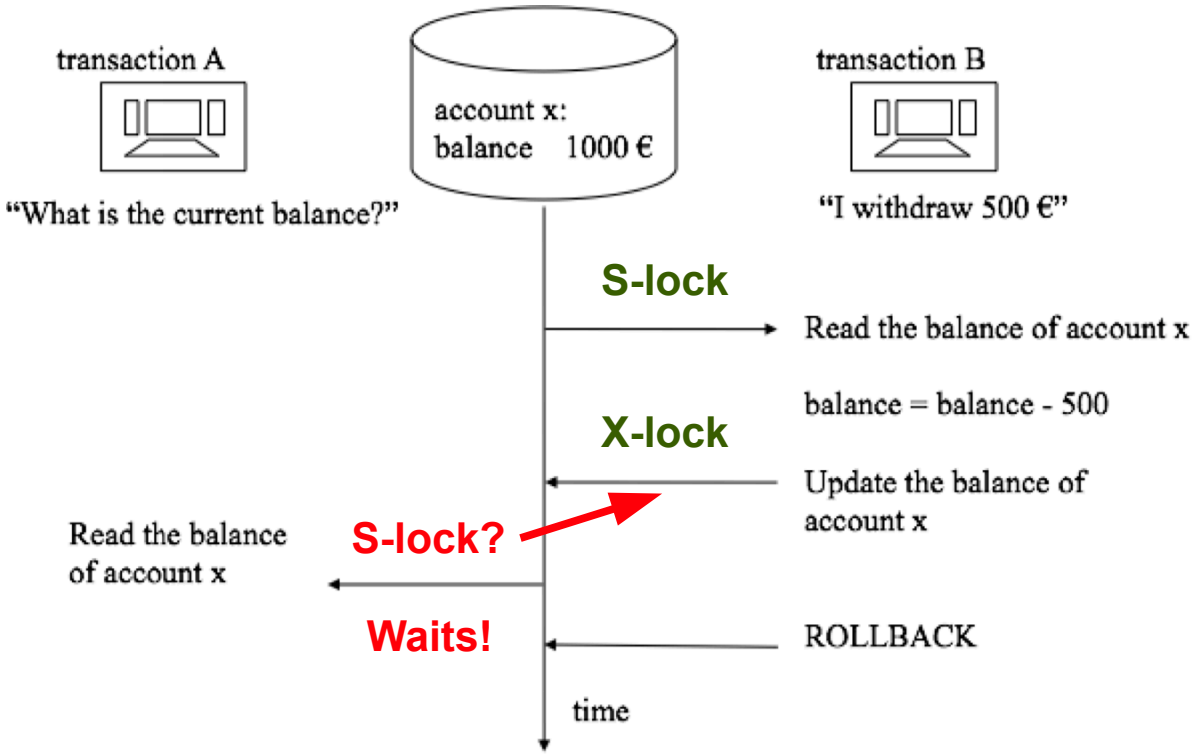
Dirty read with locks...



Dirty read with locks...

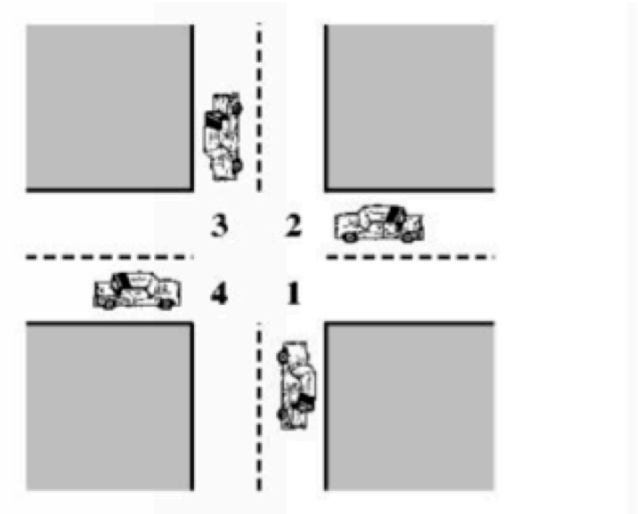


Dirty read with locks...



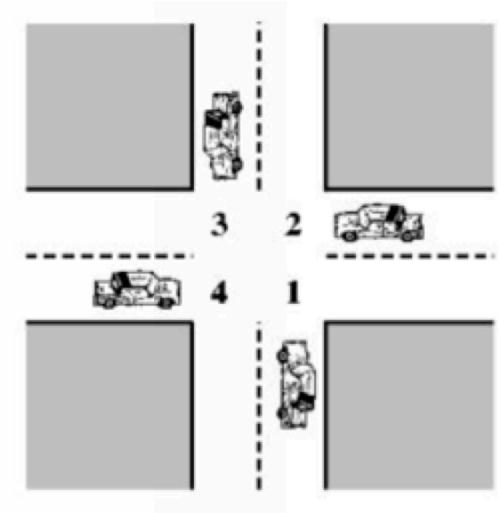
... problem resolved!

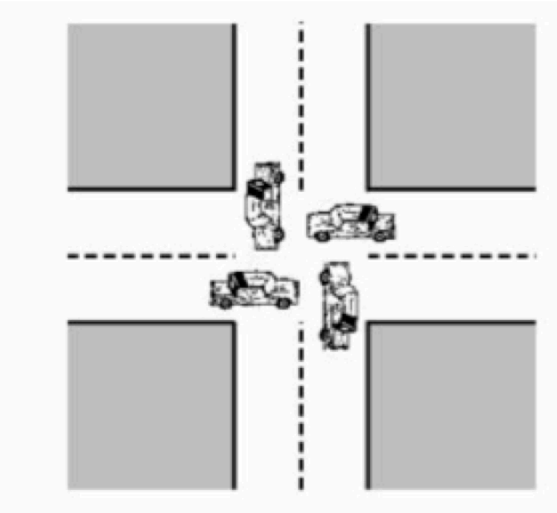
Deadlocks in general*



* From: http://cse.csusb.edu/tong/courses/cs460/notes/deadlck.php

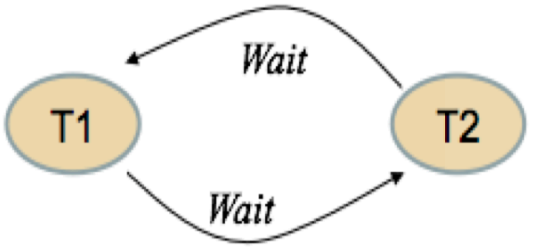
Deadlocks in general*





* From: http://cse.csusb.edu/tong/courses/cs460/notes/deadlck.php

DBMS deadlock



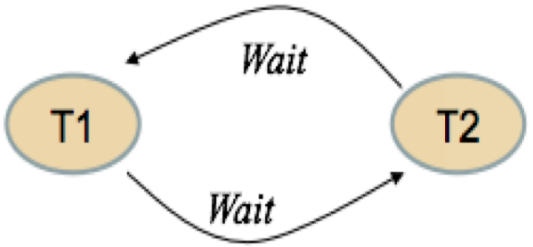
A Cycle of Lock Waits

Modern DBMS systems will detect the deadlock in some seconds (deadlock detection) and solve the waiting cycle

- *selecting the victim*
- *making automatic Rollback (not Oracle)*
- *send error message to the application*

=> *Application must react on the deadlock !*

DBMS deadlock

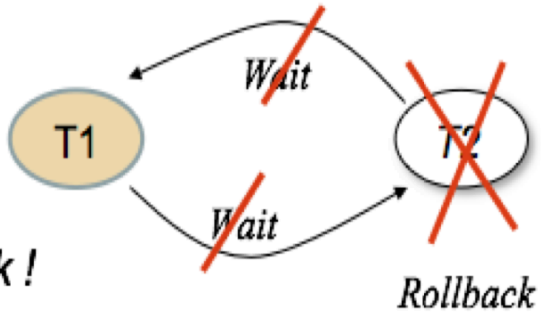


A Cycle of Lock Waits

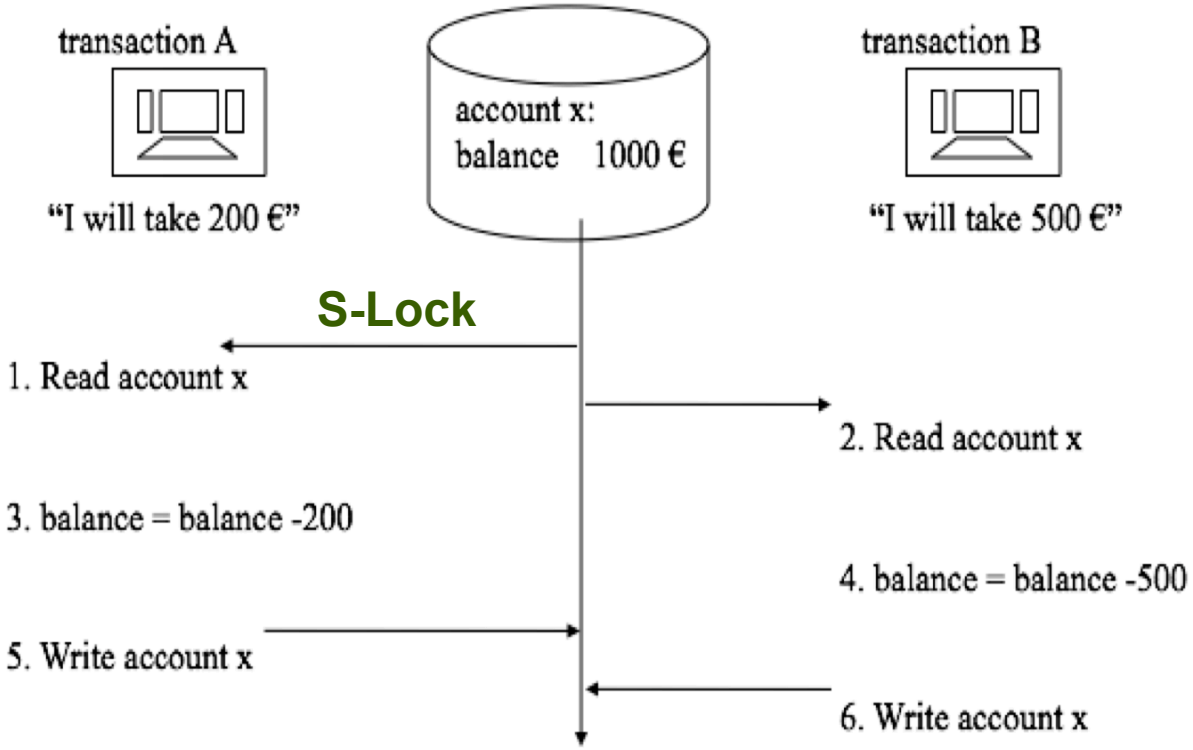
Modern DBMS systems will detect the deadlock in some seconds (deadlock detection) and solve the waiting cycle

- *selecting the victim*
- *making automatic Rollback (not Oracle)*
- *send error message to the application*

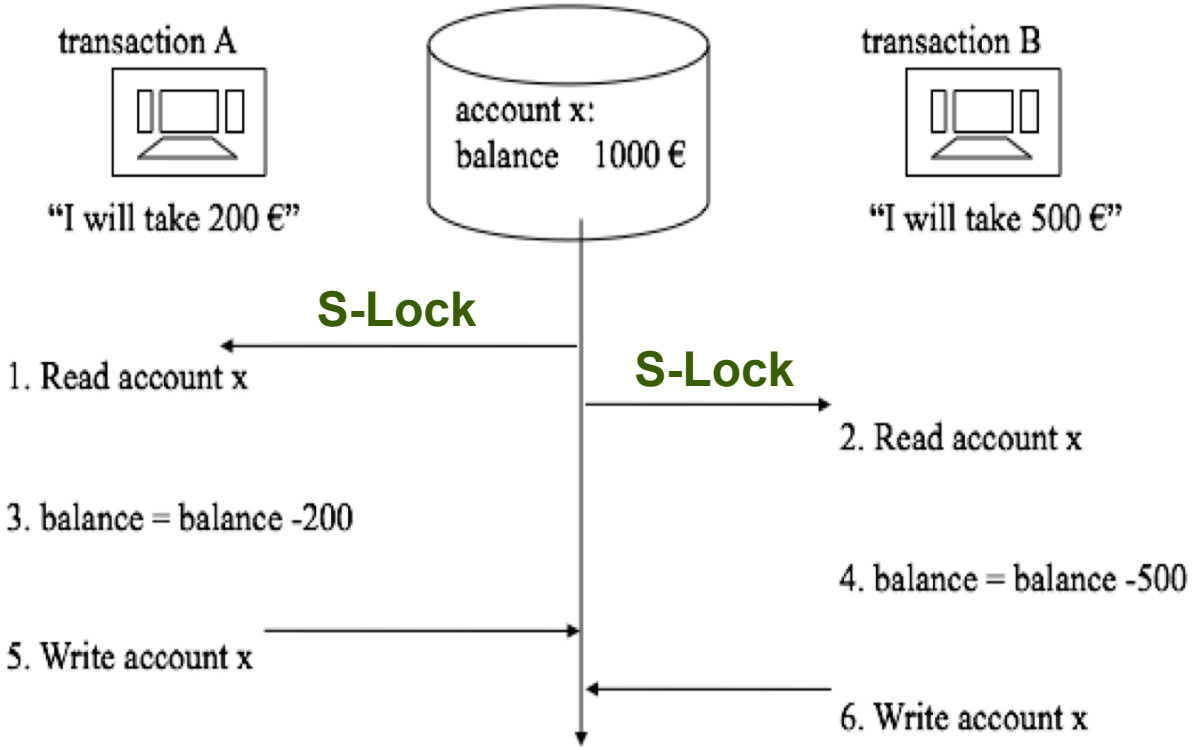
=> *Application must react on the deadlock !*



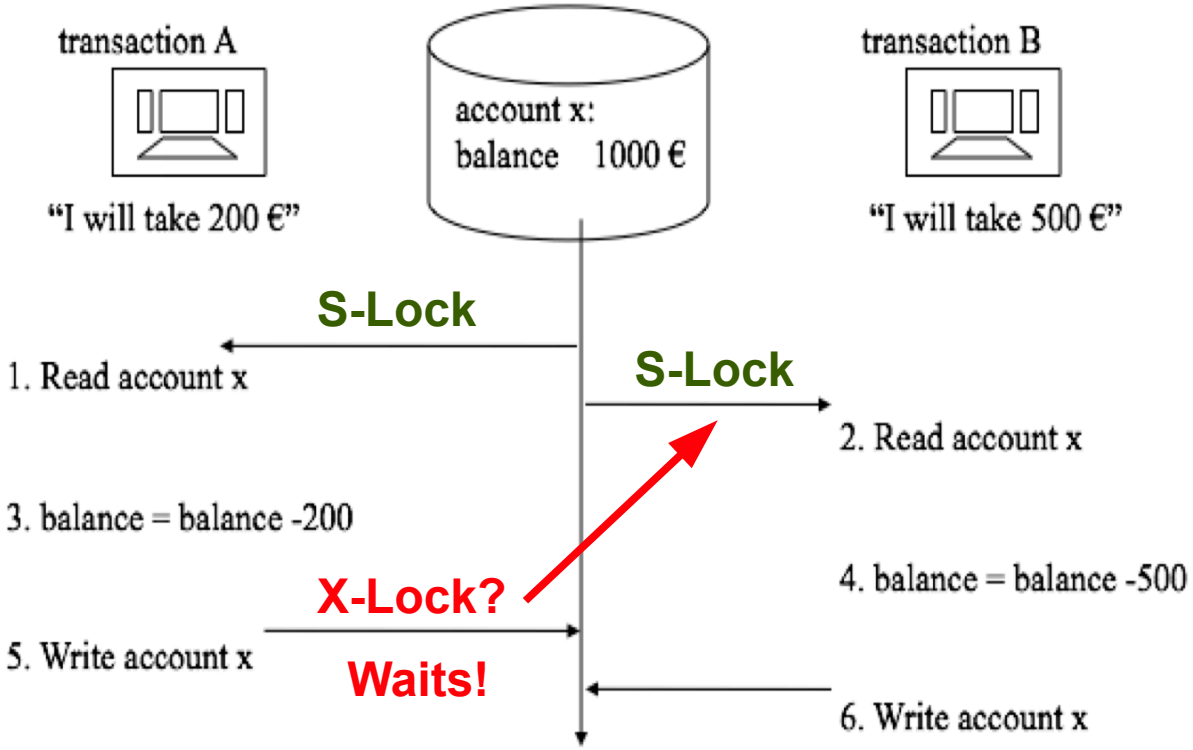
Lost update with locks



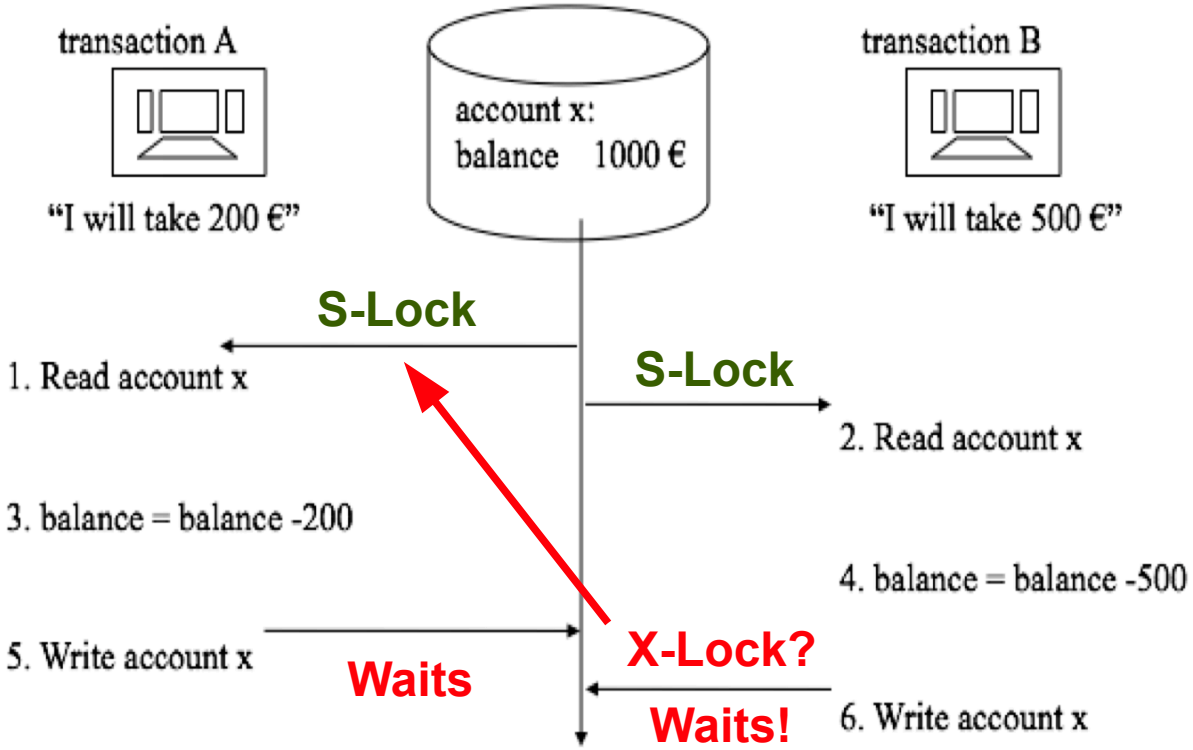
Lost update with locks



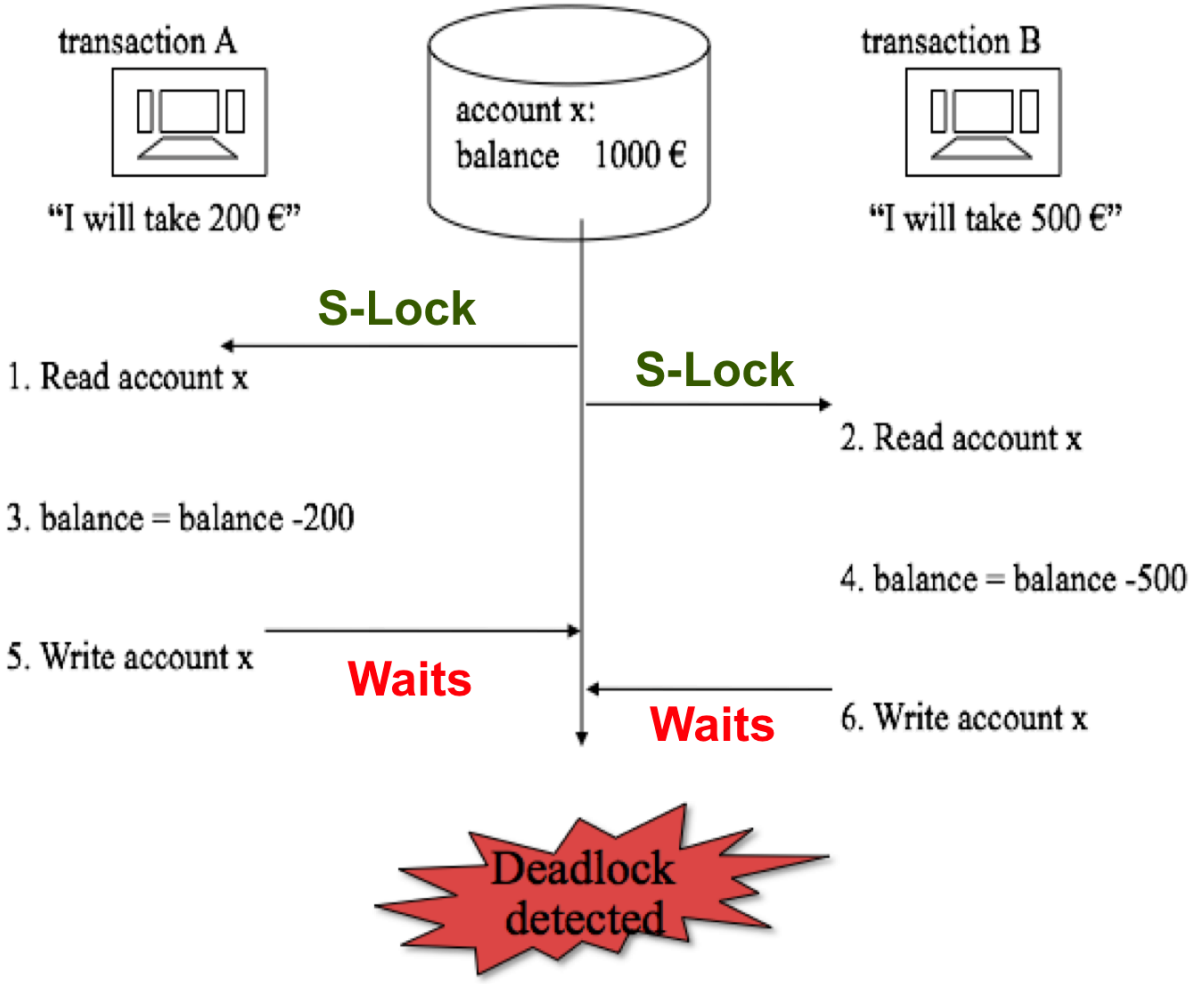
Lost update with locks



Lost update with locks



Lost update with locks



... one of (A,B) need be rolled back

Lost update with locks: sensitive updates

Using ANSI/ISO SQL, the “lost update” scenario is implemented with transactions A and B making use of (single) SQL UPDATE statements, instead of (separate) READ and WRITE operations, e.g.:

```
UPDATE Accounts SET balance = balance – 200
WHERE acctID = 100;
```

Lost update with locks: sensitive updates

Using ANSI/ISO SQL, the “lost update” scenario is implemented with transactions A and B making use of (single) SQL UPDATE statements, instead of (separate) READ and WRITE operations, e.g.:

```
UPDATE Accounts SET balance = balance – 200
WHERE acctID = 100;
```

Provided that lock-based CC protection is in effect,
the lost update problem is resolved in a deadlock-free manner.
This is the case with most of today's OLTP DBMS products...

Lost update with locks: sensitive updates

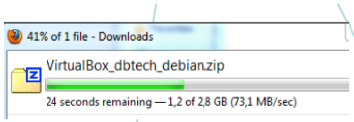
Using ANSI/ISO SQL, the “lost update” scenario is implemented with transactions A and B making use of (single) SQL UPDATE statements, instead of (separate) READ and WRITE operations, e.g.:

```
UPDATE Accounts SET balance = balance – 200
WHERE acctID = 100;
```

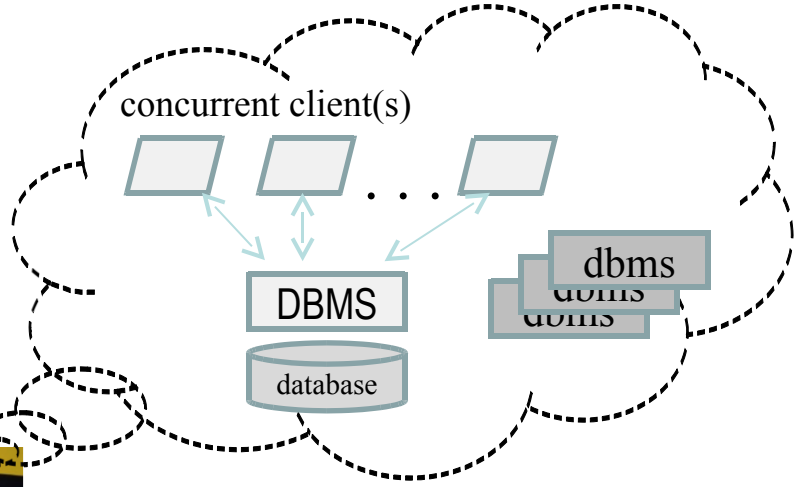
Provided that lock-based CC protection is in effect,
the lost update problem is resolved in a deadlock-free manner.
This is the case with most of today's OLTP DBMS products...

...is it, really?

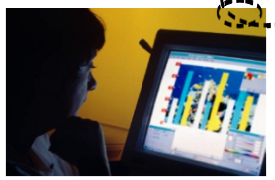
http://www.dbtechnet.org/download/DebianDBVM05.zip



.. just download, unzip, import into Oracle's VirtualBox, and run



for the classroom ...



... and/or for self-practicing

Martti Laiho

Hands-On Demo

Five short videos

Before we conclude

Lifelong Learning Programme

Any Questions?

Our next meeting: Lecture No. 4

Thursday 30 May, 7.00 – 8.00 p.m. EEST