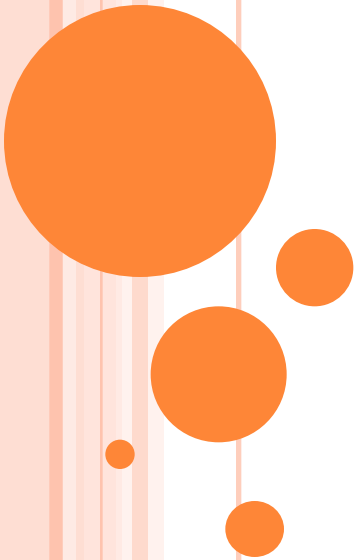


Database Administration

SIT103 Lecture 10
Transaction Processing, Locking and Deadlocks, Backups and Recovery, Performance Issues

Data Administration

- Manage the data needs of the entire organization so all data is available.
- Database Planning
- Data Analysis, Design and Implementation
- Data Maintenance and Protection
- Education and Training
- Support to end users
- High level management role

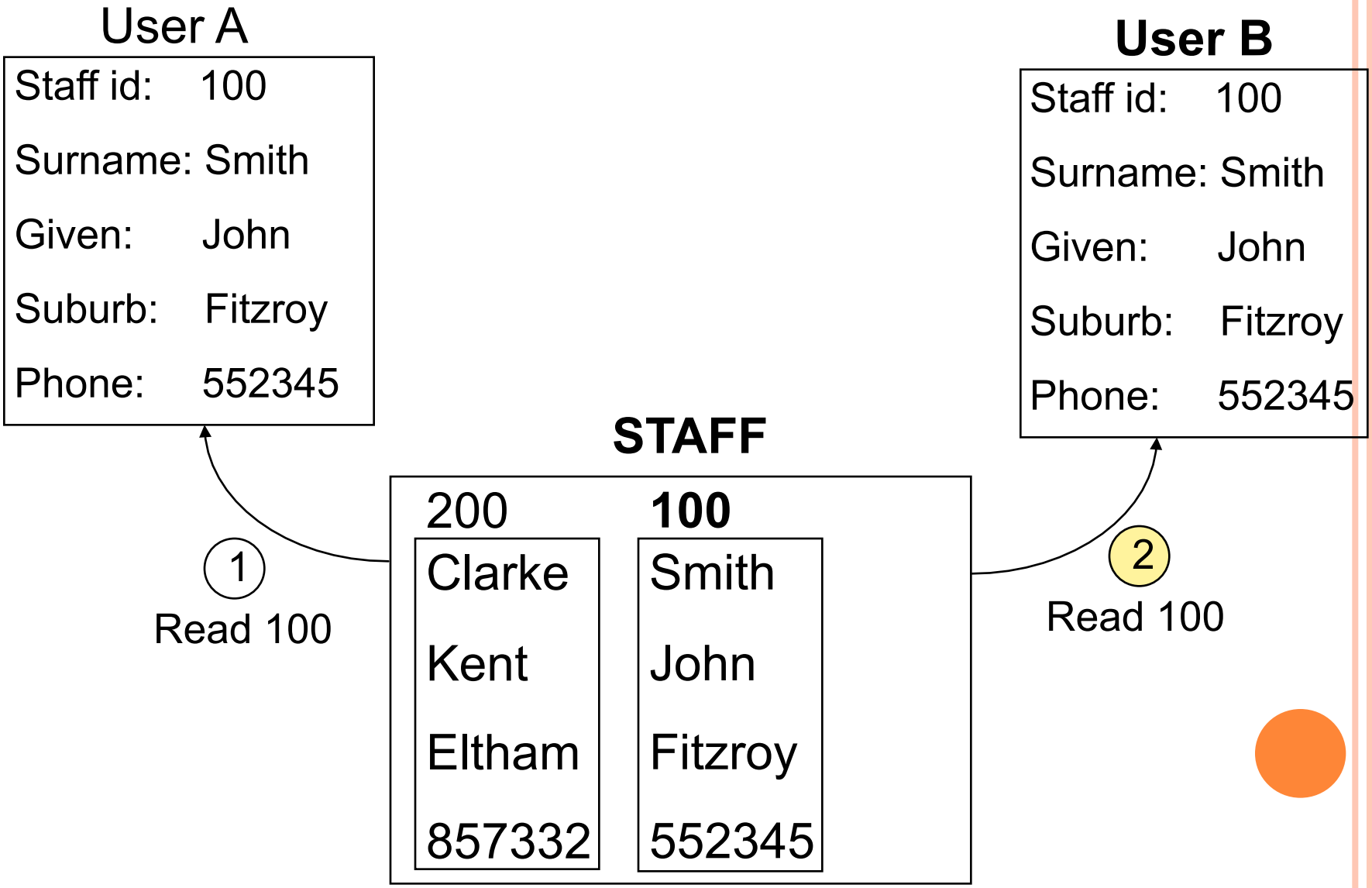
Database Administration

- Manage physical database servers.
- More of a technical, hands on role.
- Installation and configuration of DBMS
- Data Analysis, Design and Implementation
- Performance Tuning
- Managing Security and Policies
- Managing backups and recovery

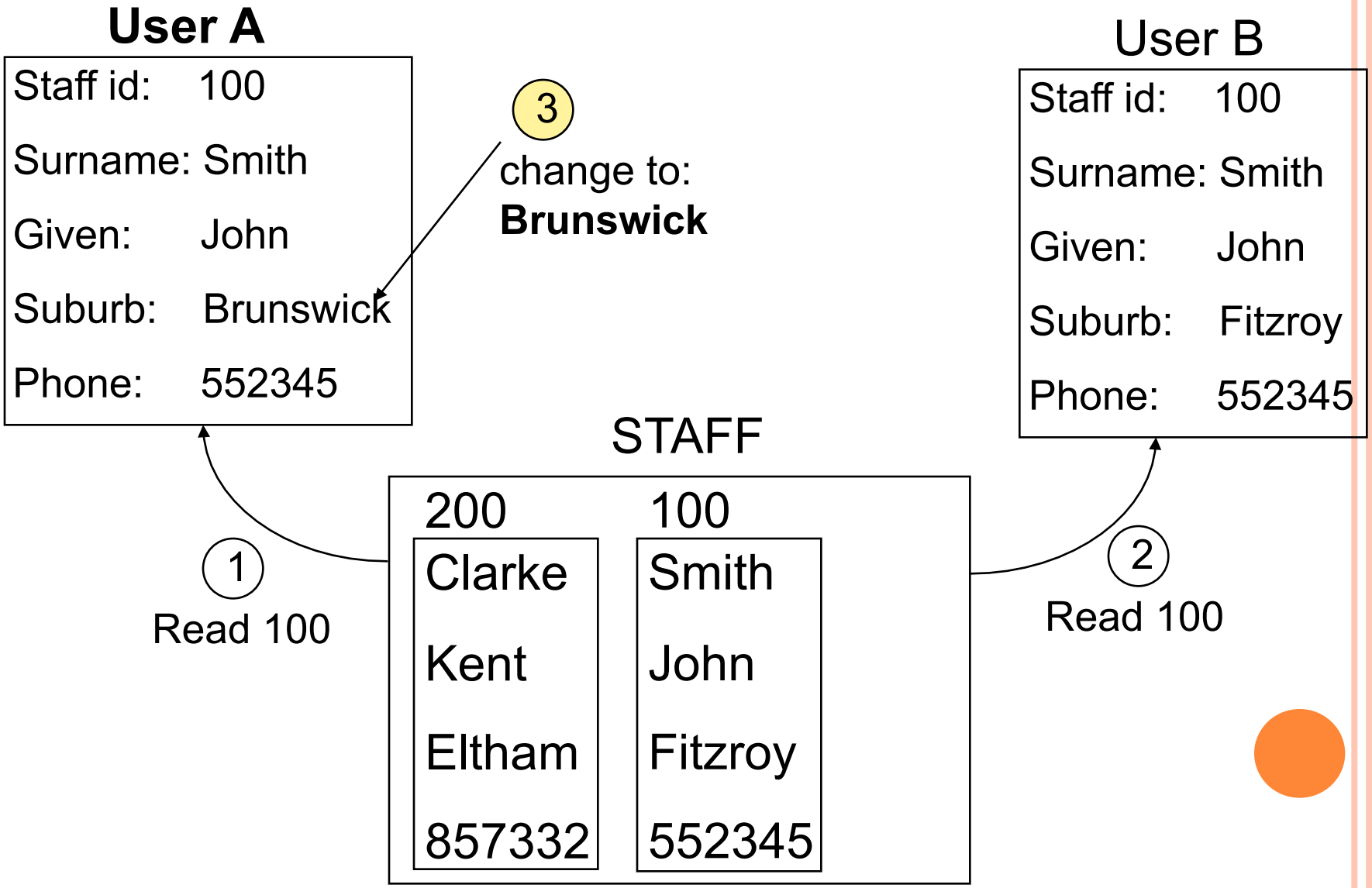
Changing Roles

- In larger companies, these two roles may be performed by distinct people or groups.
- In smaller organisations or departments, one person will often perform both roles.
- DBA roles are becoming specialized:
  - Product specific (eg: Oracle DBA)
  - Overlap with developers (server programming)
  - Database types (eg: data warehouses)
  - Specific App Software (eg: SAP or PeopleSoft)

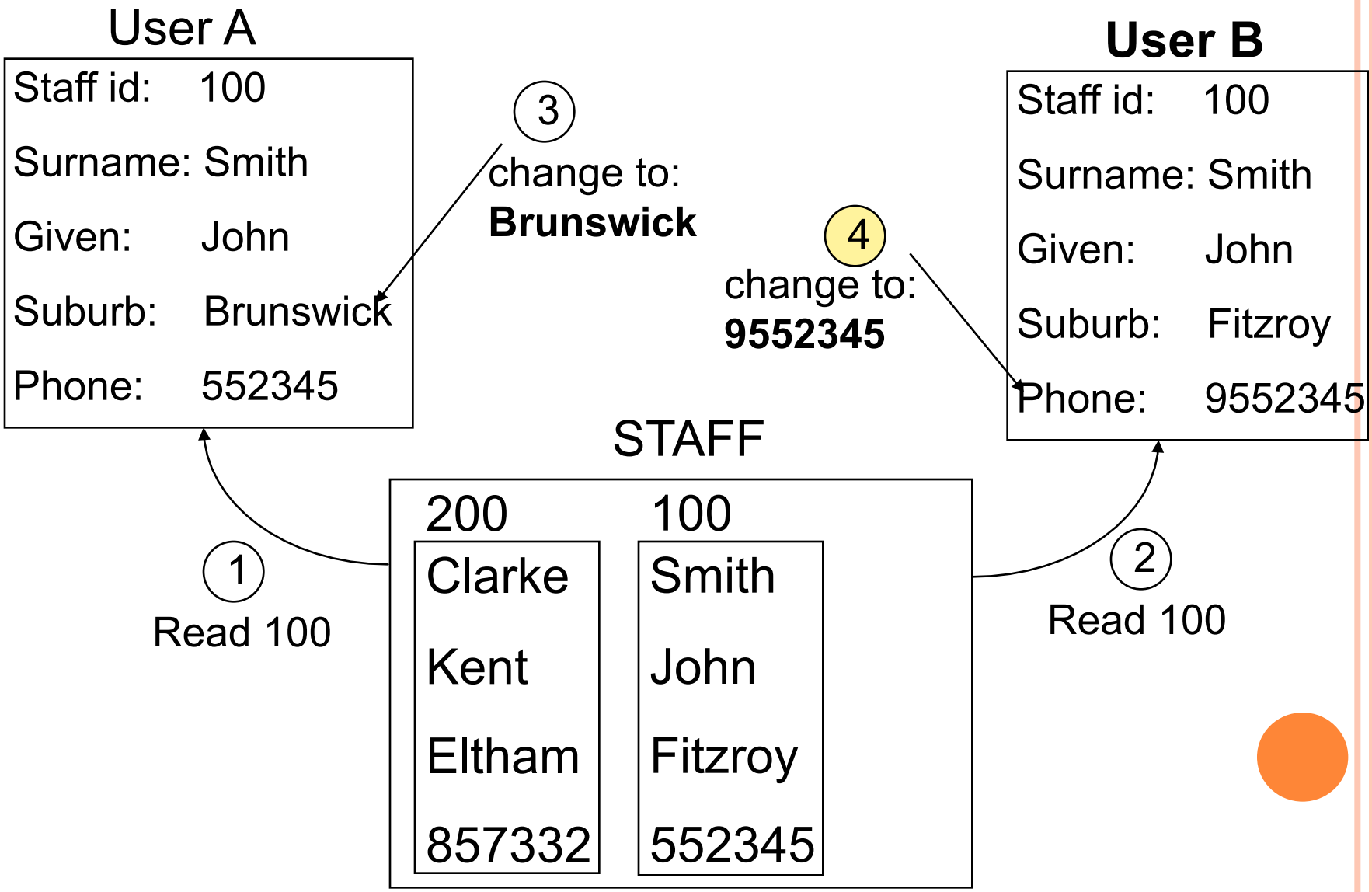
Transaction Processing - no locking



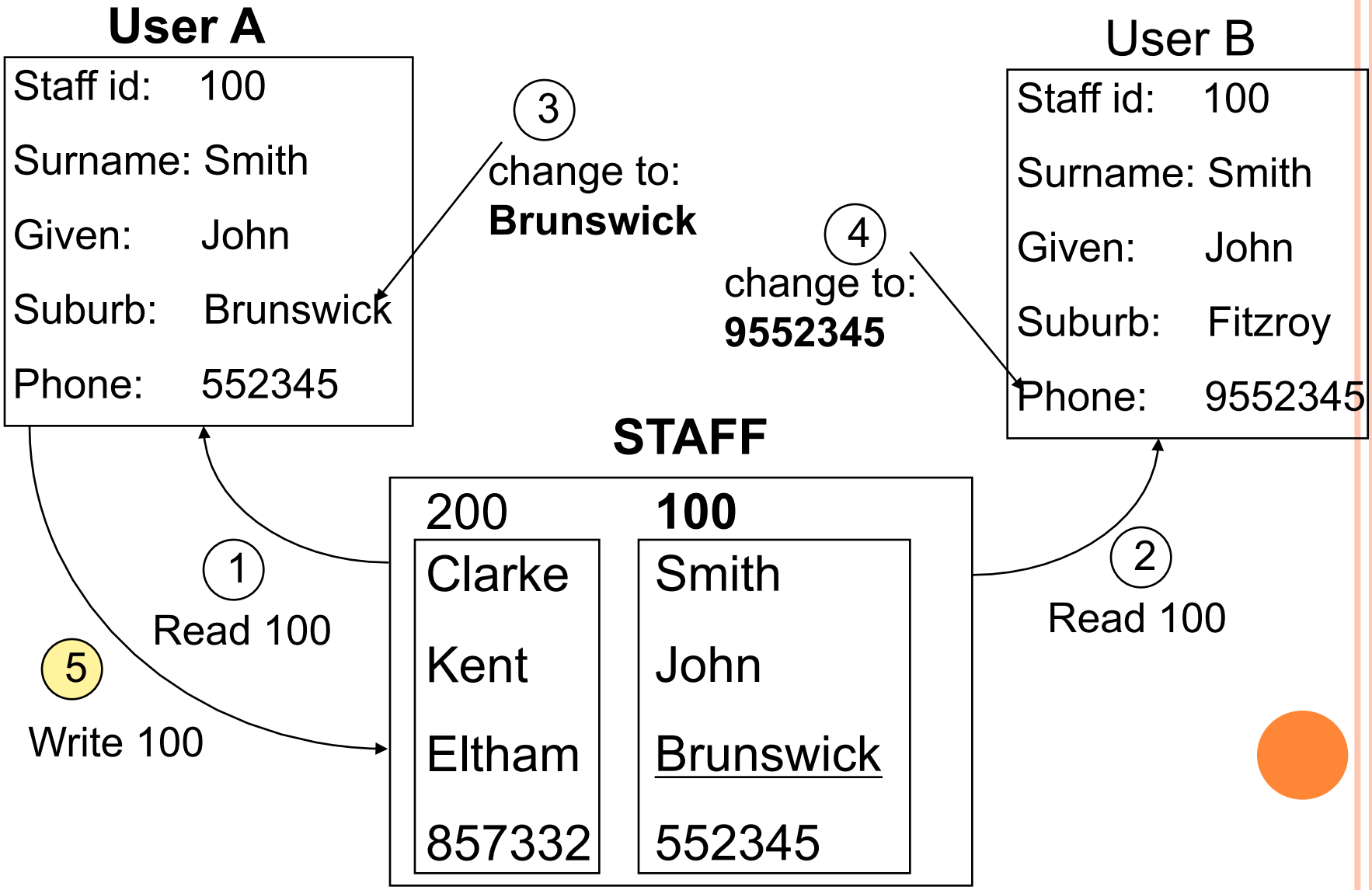
Transaction Processing - no locking



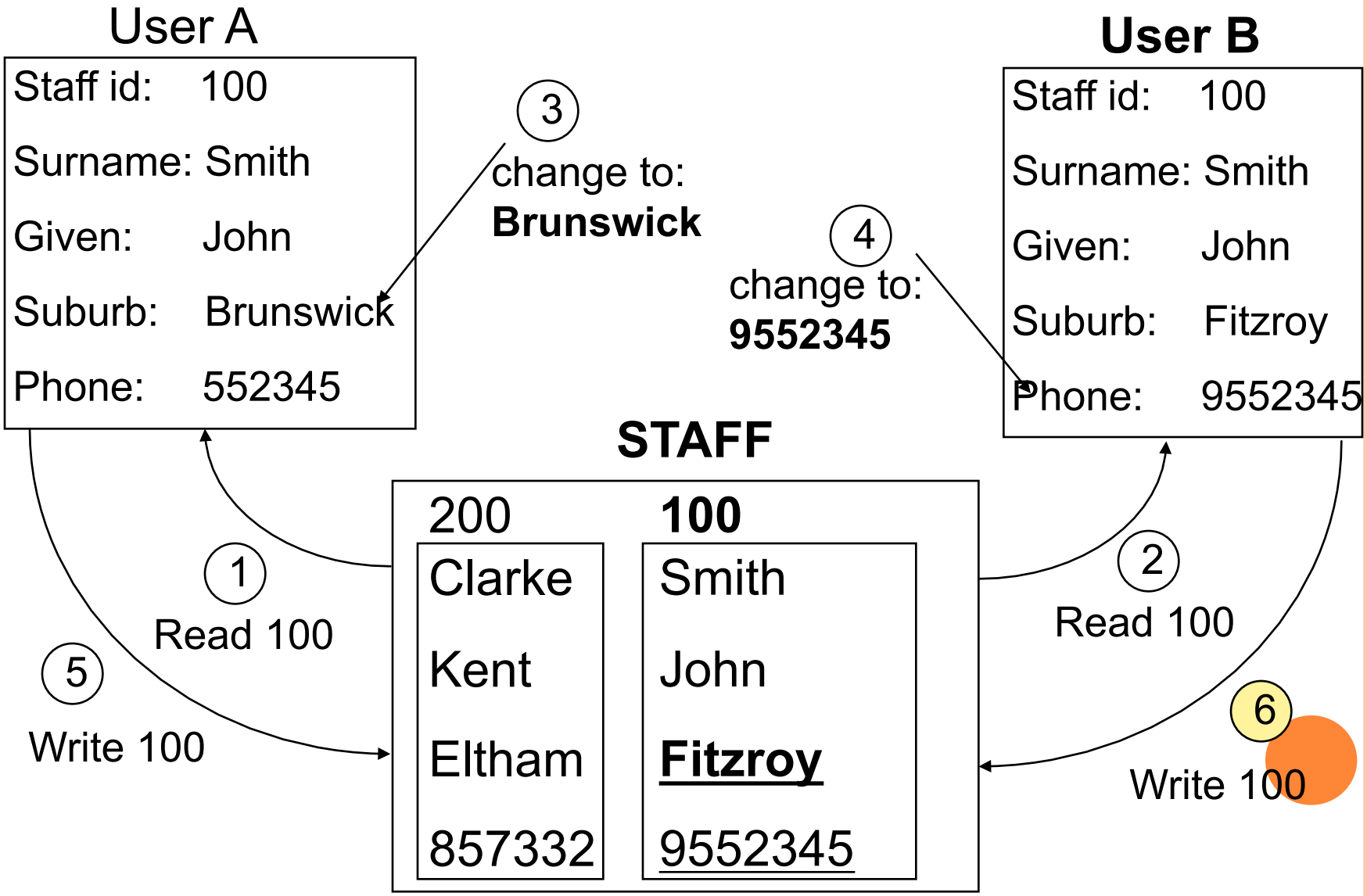
Transaction Processing - no locking



Transaction Processing - no locking



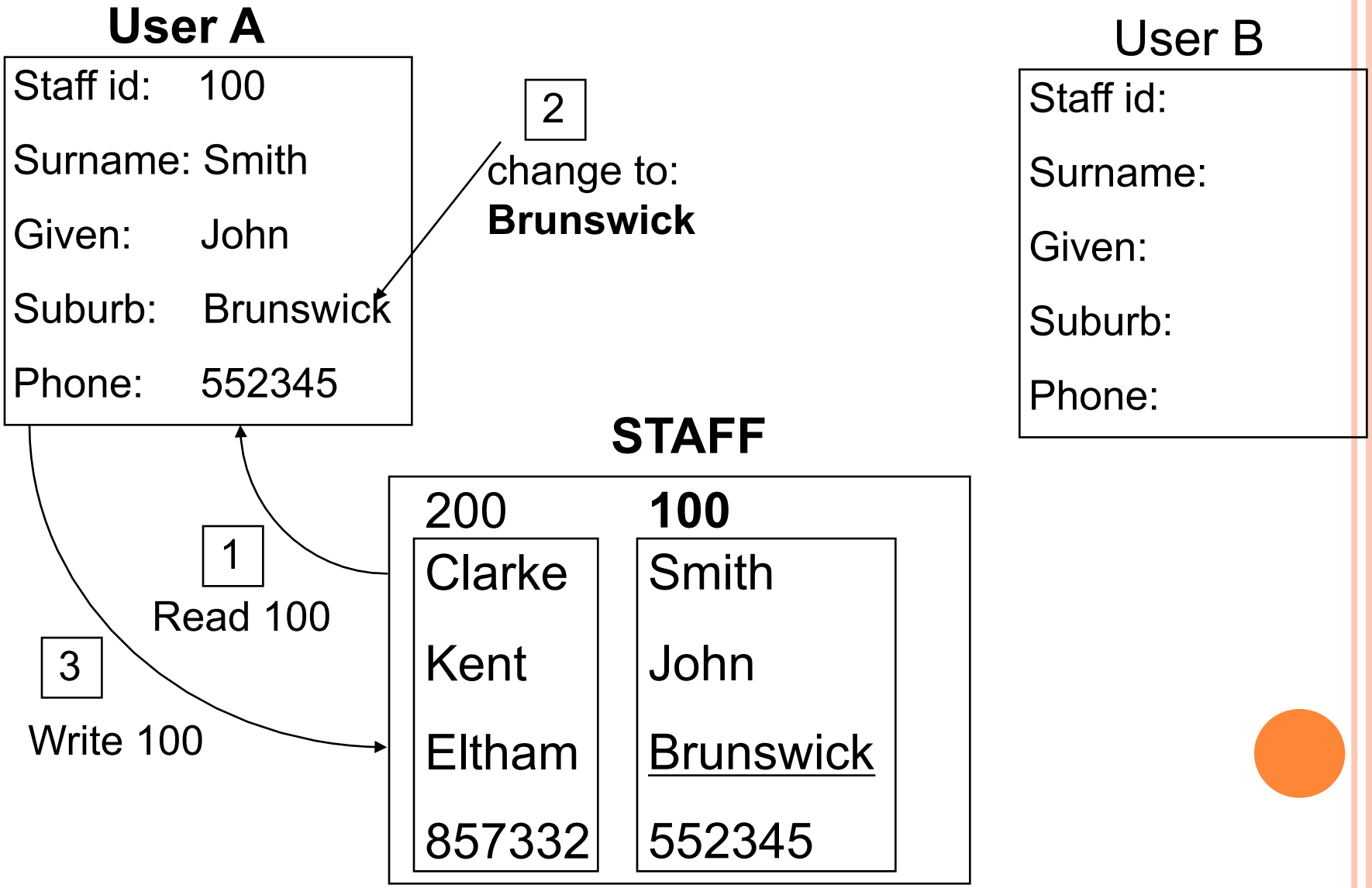
"known as the LOST UPDATE PROBLEM"



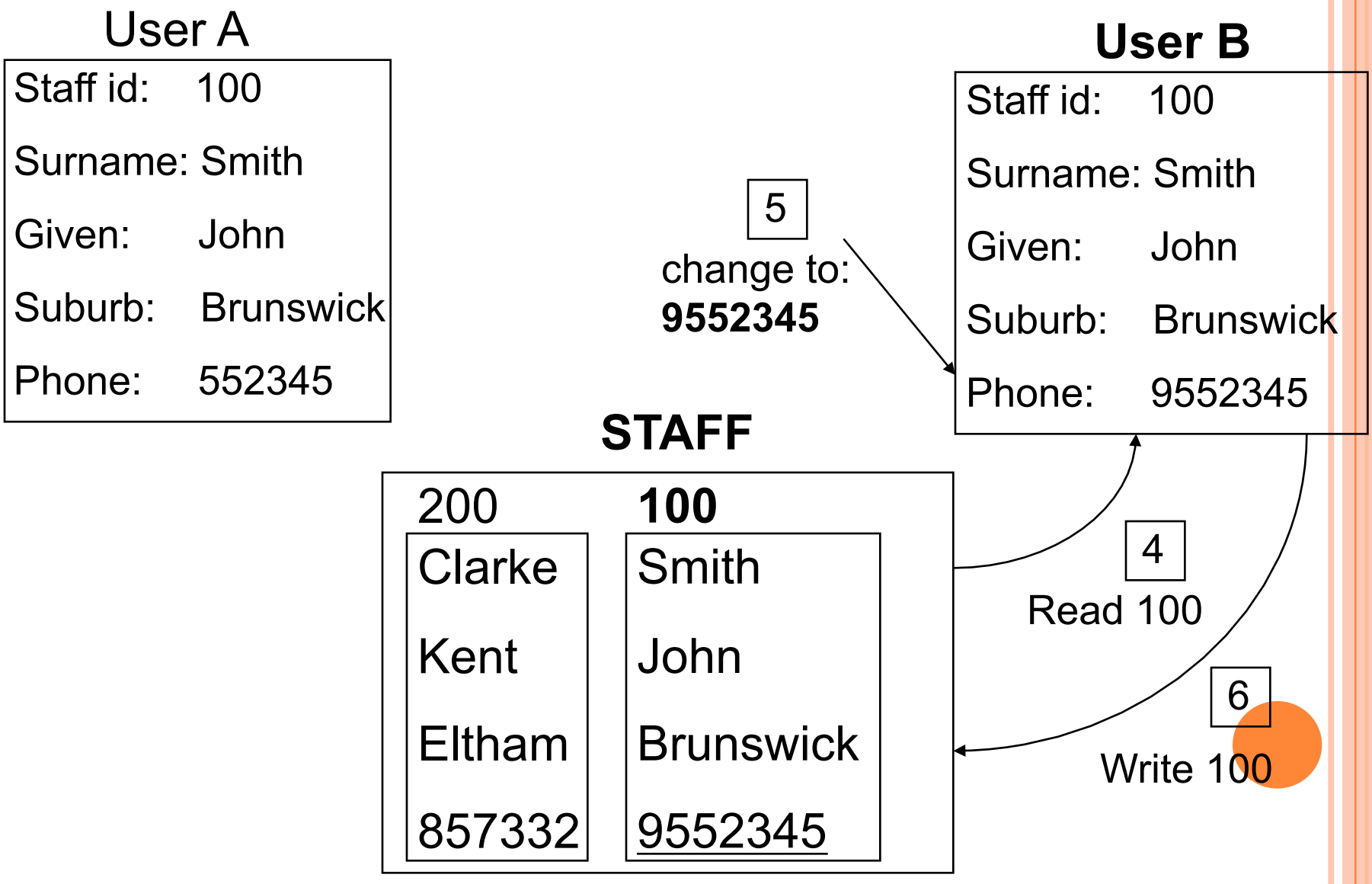
Locking

- How do we solve this problem?
  - When a user/process wants to use a resource, LOCK IT.
  - No other user/process may change it until it is UNLOCKED.
- if used LOCKING in previous example order of execution would change, lets us see what happens.....

Transaction Processing - with locking



Transaction Processing - with locking



Level Of Locking

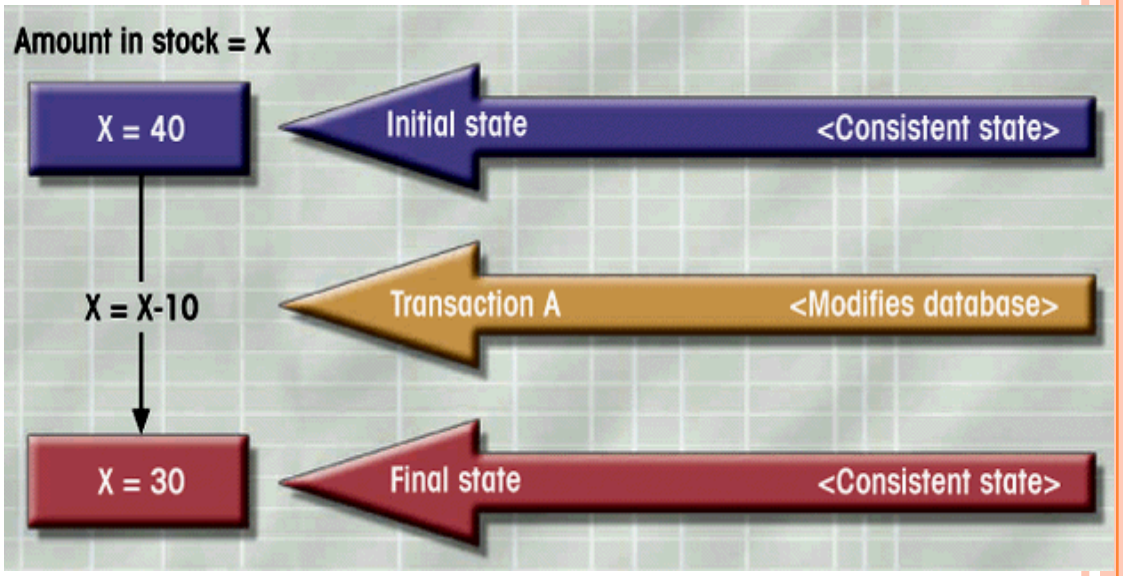
1. **Database** - backup
2. **Table** - batch work (updates)
3. **Block/Page** - most common
4. **Record** - 1 record, most commonly required

Lock Types

- Shared Locks
  - other users/processes can read but not update; i.e. allowing querying;
- Exclusive Locks
  - other users can do NOTHING.

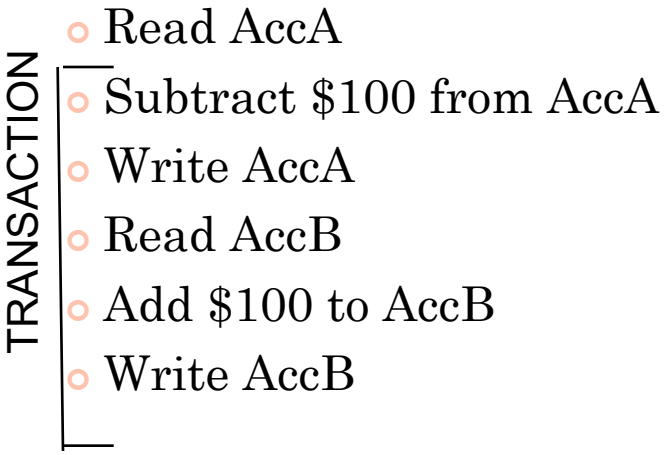
What is a transaction?

- Logical unit of work
  - Must be entirely completed or aborted
  - No intermediate states



Transaction

- Is a single business operation.
- A whole unit of work MUST be applied or NONE will be applied to database.
- eg. Transfer $100 from AccA to AccB.



Desirable properties of transactions

- Atomicity
  - All transaction operations must be completed
  - Incomplete transactions aborted
- Consistency
  - Maps one consistent database state to another
- Isolation
  - Transaction data cannot be reused until its execution complete
- Durability
  - Permanence of consistent DB state once a transaction commits

Example

- Assume the following SQL
  - Consistent state after transaction is executed
  - No changes made to database

```
SELECT ACC_NUM, ACC_BALANCE
FROM CHECKACC
WHERE ACC_NUM = '0908110638';
```

Example (cont'd)

- Register credit sale of 100 units of product X to customer Y for $500
  - State is consistent if both transactions complete
  - DBMS does not guarantee this

applied

```
UPDATE PRODUCT
SET PROD_QOH = PROD_QOH - 100
WHERE PROD_CODE = 'X';

UPDATE ACCT_RECEIVABLE
SET ACCT_BALANCE = ACCT_BALANCE + 500
WHERE ACCT_NUM = 'Y';
```

Transaction management in SQL

- Transaction support
  - COMMIT
  - ROLLBACK
- A transaction sequence must continue until:
  - COMMIT statement is reached
  - ROLLBACK statement is reached
  - End of a program reached
  - Program reaches abnormal termination

Deadlocks

- Deadlocks arise from more than one process holding a resource and requesting other resources which are in turn held by other processes, for example:

User A
transfer $100 from X to Y
Read X {X is locked}
Subtract $100 from X
Read Y
- Y is locked by user A, waiting

User B
transfer $100 from Y to X
Read Y {Y is locked}
Subtract $100 from Y
Read X
- X is locked by user A, waiting

Deadlock Resolution

- Pessimistic Locking Scheme
  - every process must lock ALL resources required first.
  - if cannot, unlock everything and have another go later
  - performance slow due to overhead of locking everything.

Deadlock Resolution

- Deadlock Watchdog
  - Let processes do what they want/please
  - program keeping an eye on locks
  - If DEADLOCK recognized then kill off processes until resolved.

Deadlock Resolution

- Optimistic Locking Scheme (versioning)
  - read and use records without concern
  - before writing at end of transaction
    - re-read record(s)
    - if not changed since 1st read the WRITE Ok
    - if has changed then abort whole transaction!
  - efficient - performance high; if over 90%, no change then QUICK1

Backups

- If your computer room has a fire and everything is lost, how long would it take to replace:
  - Hardware
  - Operating System
  - DBMS (eg: Oracle)
  - Data in the database
- Which is the most valuable ??

Database Recovery - Backup

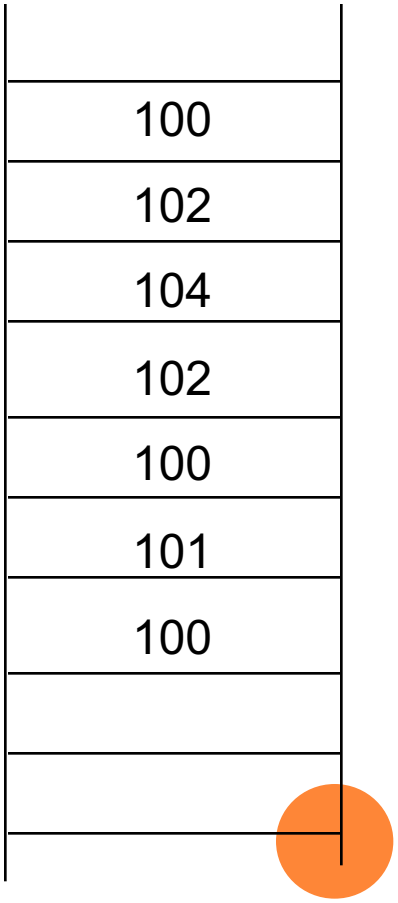
- Full Backup
  - everything is backed up on some sort of magnetic media.
- Partial Backup
  - backup changes made since last FULL Backup was made.

Database Recovery - Backup

- Recommended backup philosophy
  - FULL Backup each week
  - PARTIAL backups each day
  - ROTATING the media used
  - OFFSITE media storage

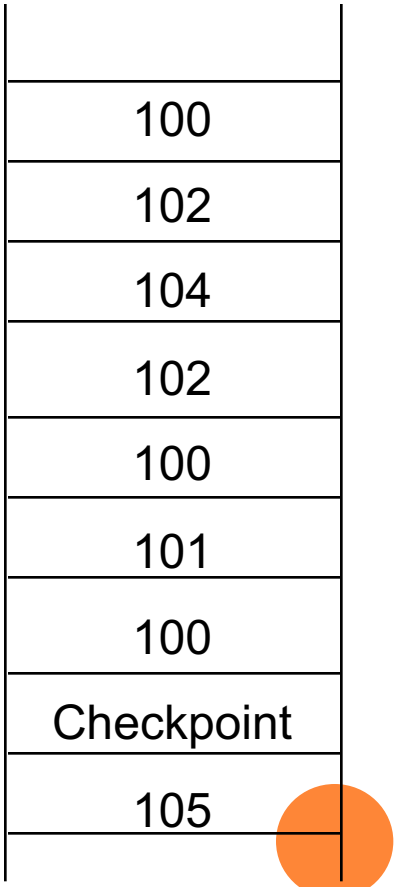
Database Recovery - Logs

- Journals/Transaction Logs
- Log enough information about a change to allow it to be applied again, if required.
  - before and after images of record changed; and/or
  - details about each transaction;
  - log start and finish of transaction.



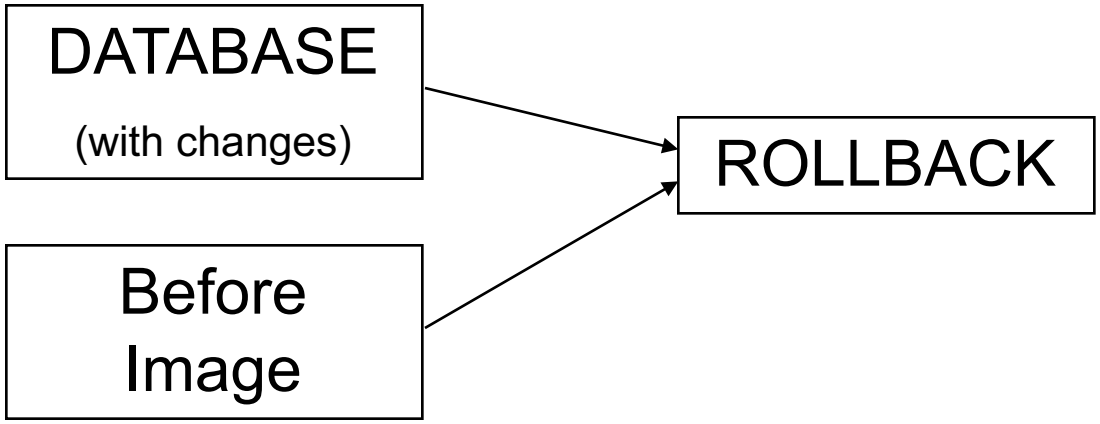
Database Recovery - Checkpoints

- Allows all transactions to finish, at a particular time.
- Write a checkpoint, as a log entry. This gives us a point at which no transaction are unfinished.
- Useful for recovery, if a problem occurs with the database.



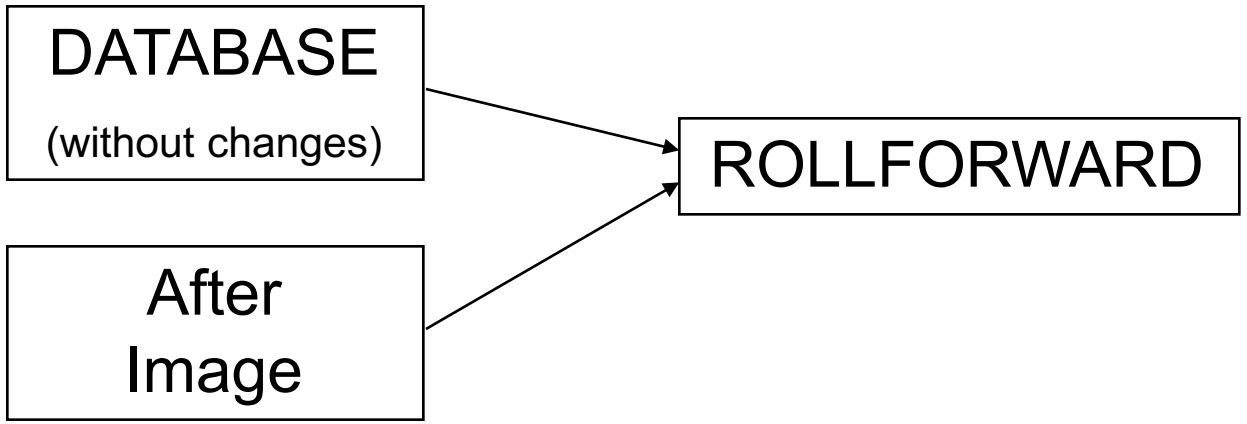
Database Recovery - Rollback

- If database intact
  - can undo changes until checkpoint because OK point.
  - undo unfinished transaction back to a safe point (usually checkpoint)



Database Recovery - RollForward

- database retrieved from backup
- re-apply transactions up to latest checkpoint.



Database - Security

- more sensitive data being stored
- more people given access to database

- security is defined as protection of the database against accidental or intentional loss, destruction or misuse.

Database - Security

- Data management software usually provides the following security features:
  - creation of views to restrict user access;
  - authorization rules;
  - programs further constraining or limiting database access;
  - encryption of data;
  - identify user attempting any database access (biometric, smartcard).