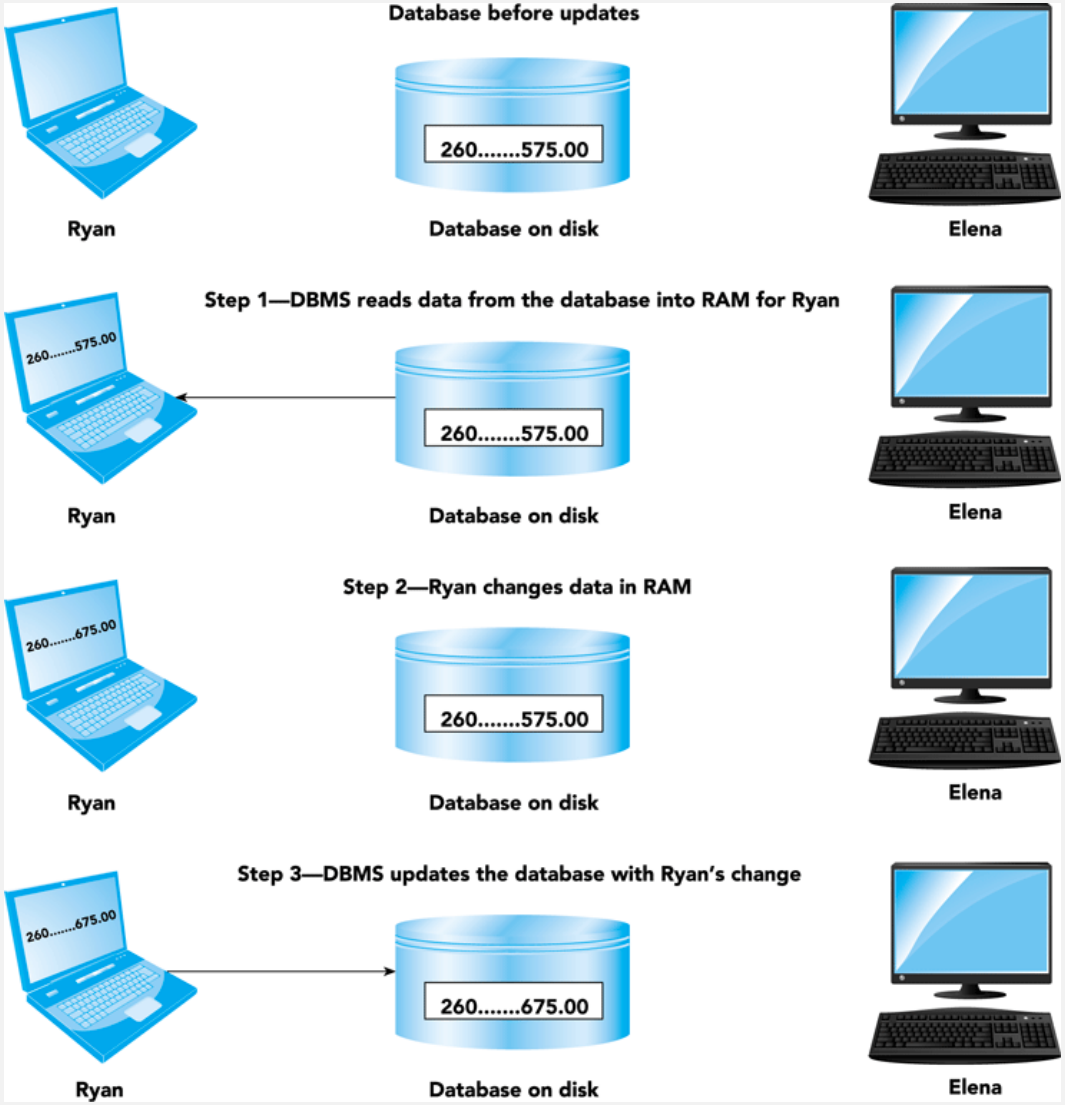
Concepts of Transactions and Locking

Support Concurrent Update

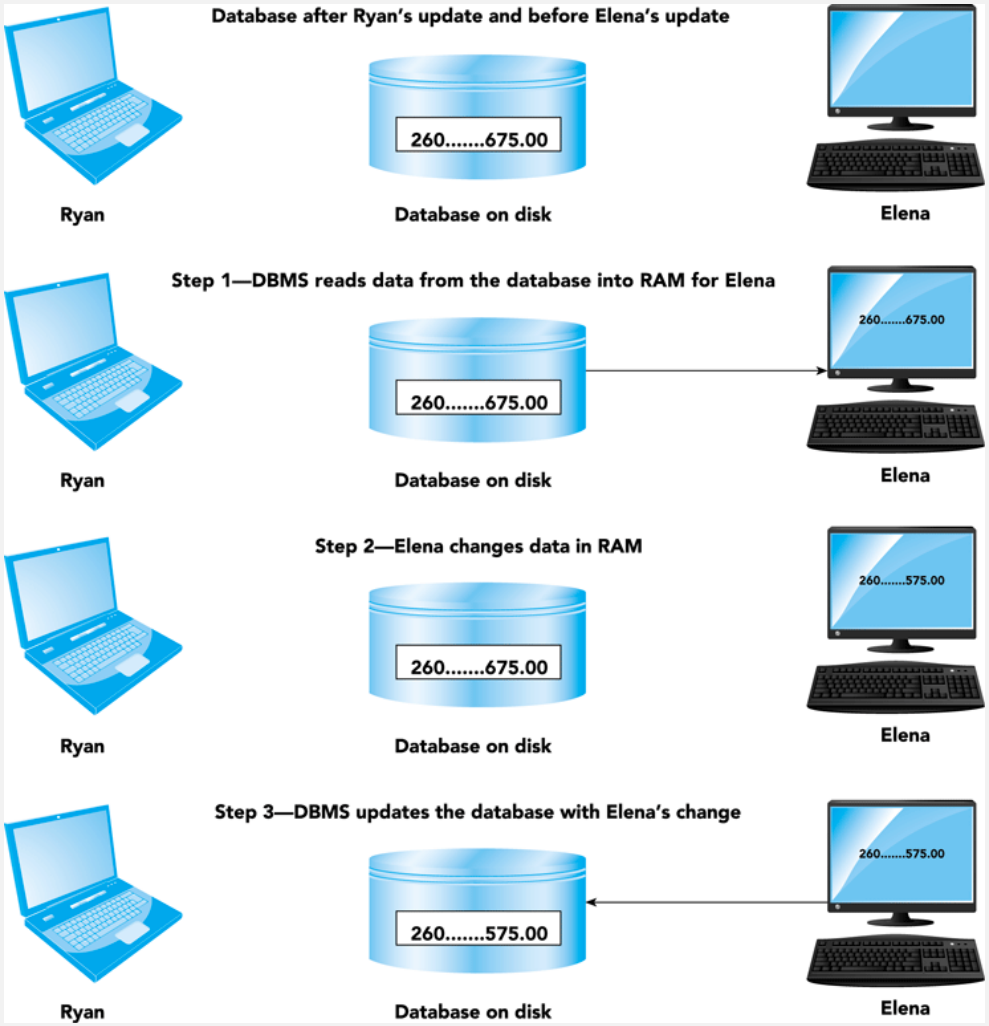
- Ensures accuracy when several users update database at the same time
- Manages complex scenarios for updates
- **Concurrent update**: multiple users make updates to the same database at the same time

The Concurrent Update Problem



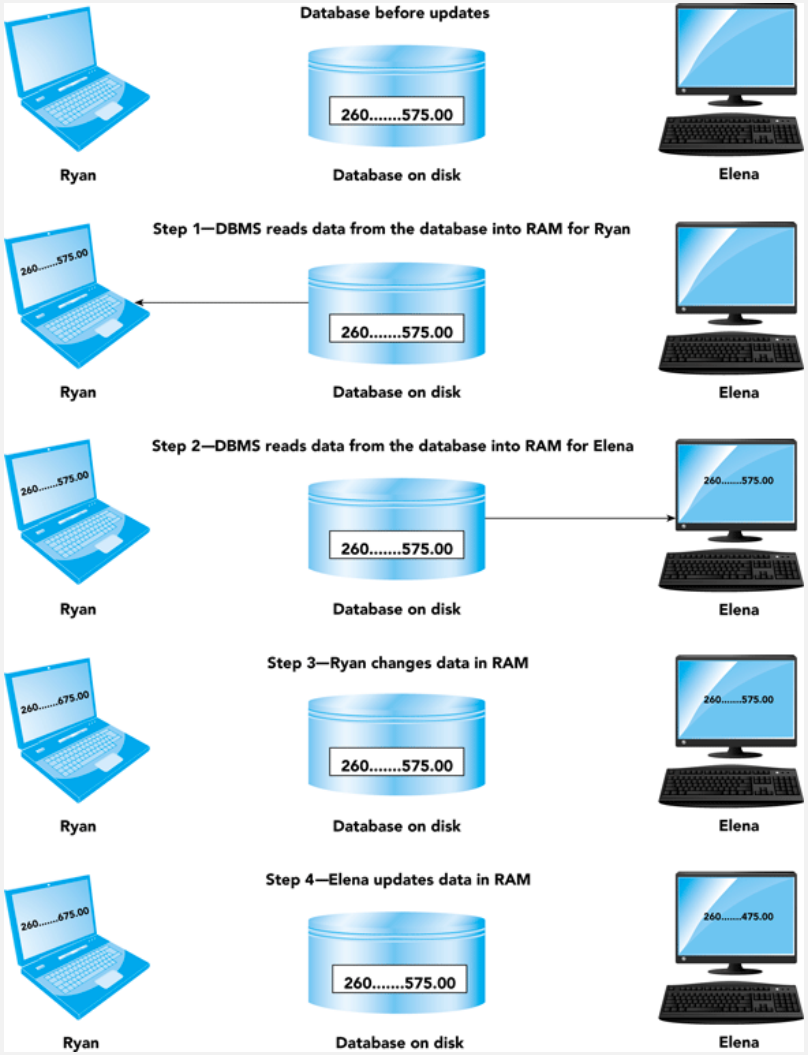
Ryan updates the database

The Concurrent Update Problem (continued)



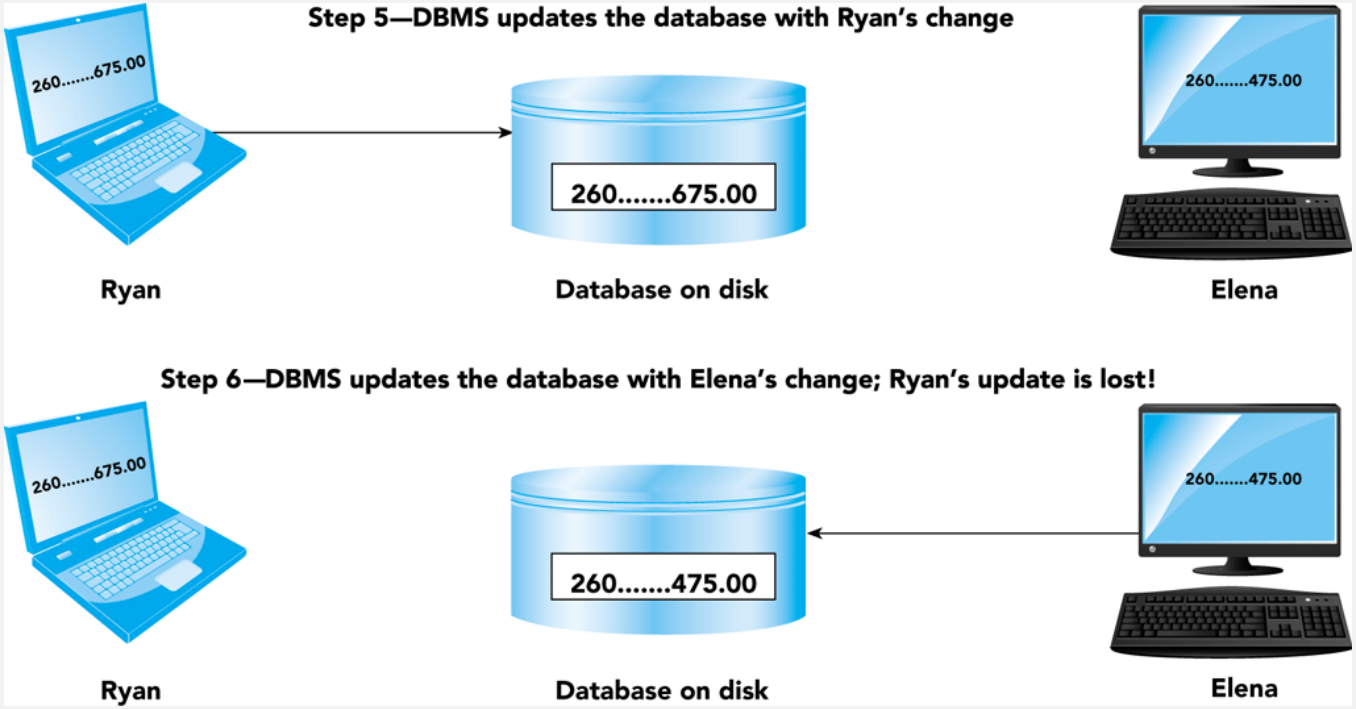
Elena updates the database

The Concurrent Update Problem (continued)



Ryan's and Elena's updates to the database result in a lost update

The Concurrent Update Problem (continued)



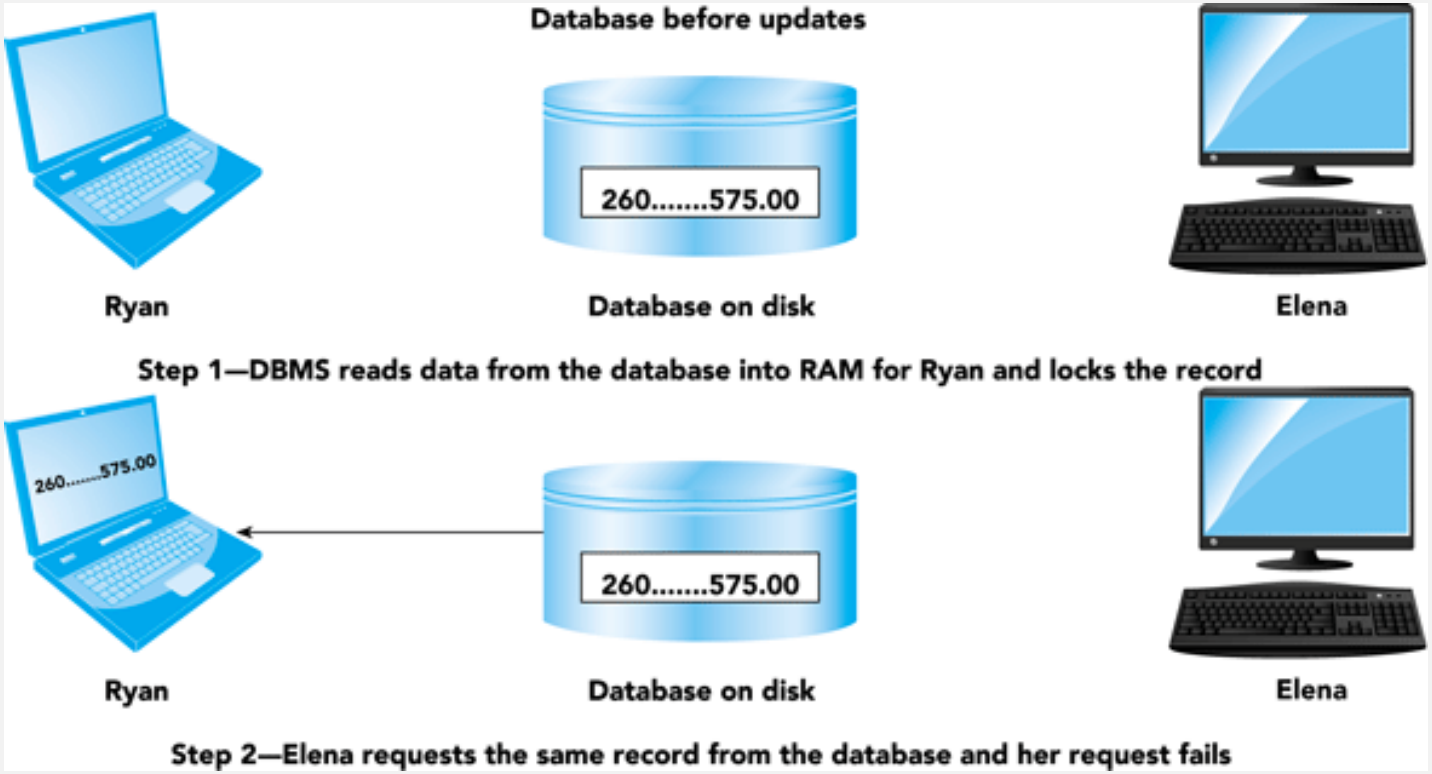
Avoiding the Lost Update Problem

- **Batch processing**
  - All updates done through a special program
  - Problem: data becomes out of date
  - Does not work in situations that require data to be current

Two-Phase Locking

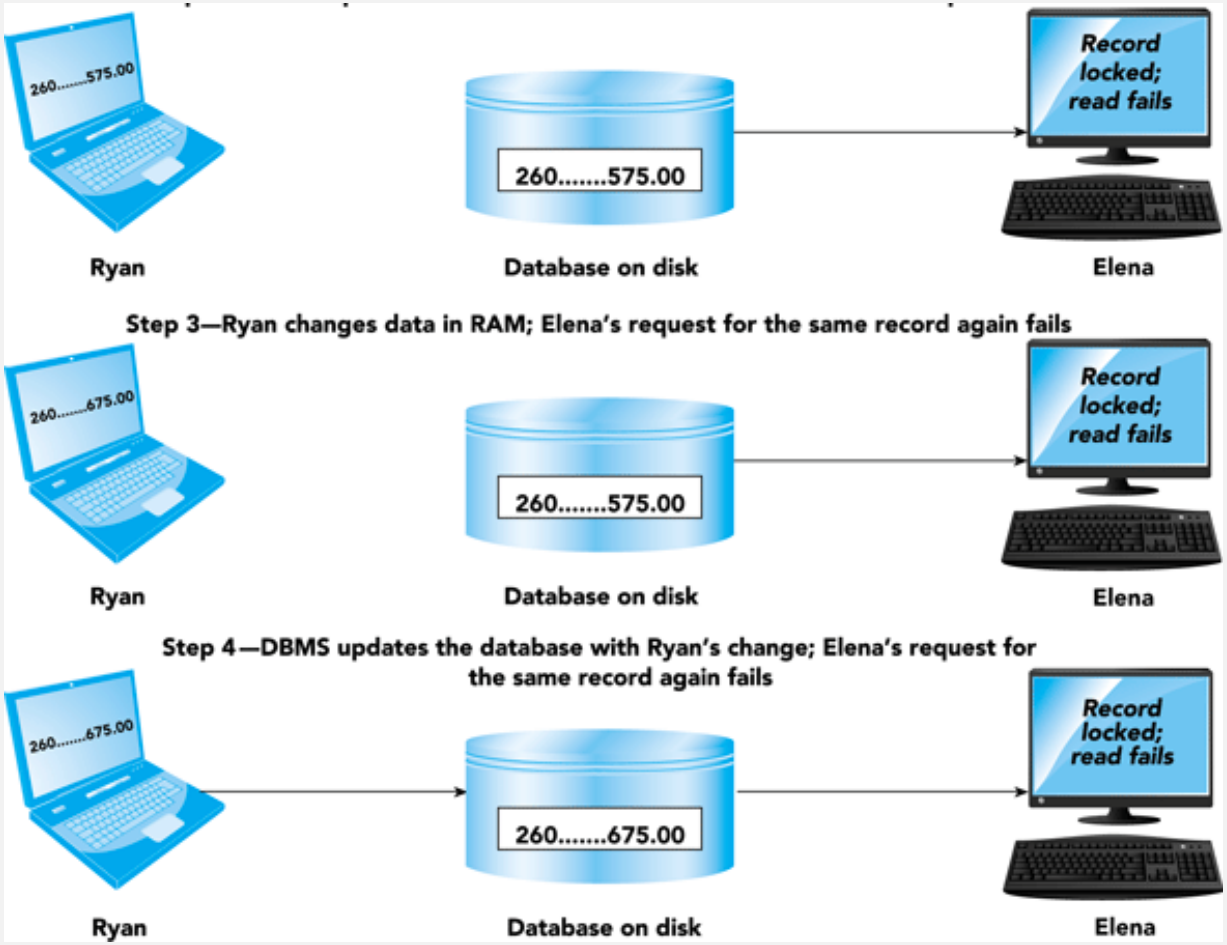
- **Locking**: deny other users access to data while one user's updates are being processed
- **Transaction**: set of steps completed by a DBMS to accomplish a single user task
- **Two-phase locking** solves lost update problem
  - **Growing phase**: DBMS locks more rows and releases none of the locks
  - **Shrinking phase**: DBMS releases all the locks and acquires no new locks

Two-Phase Locking (continued)

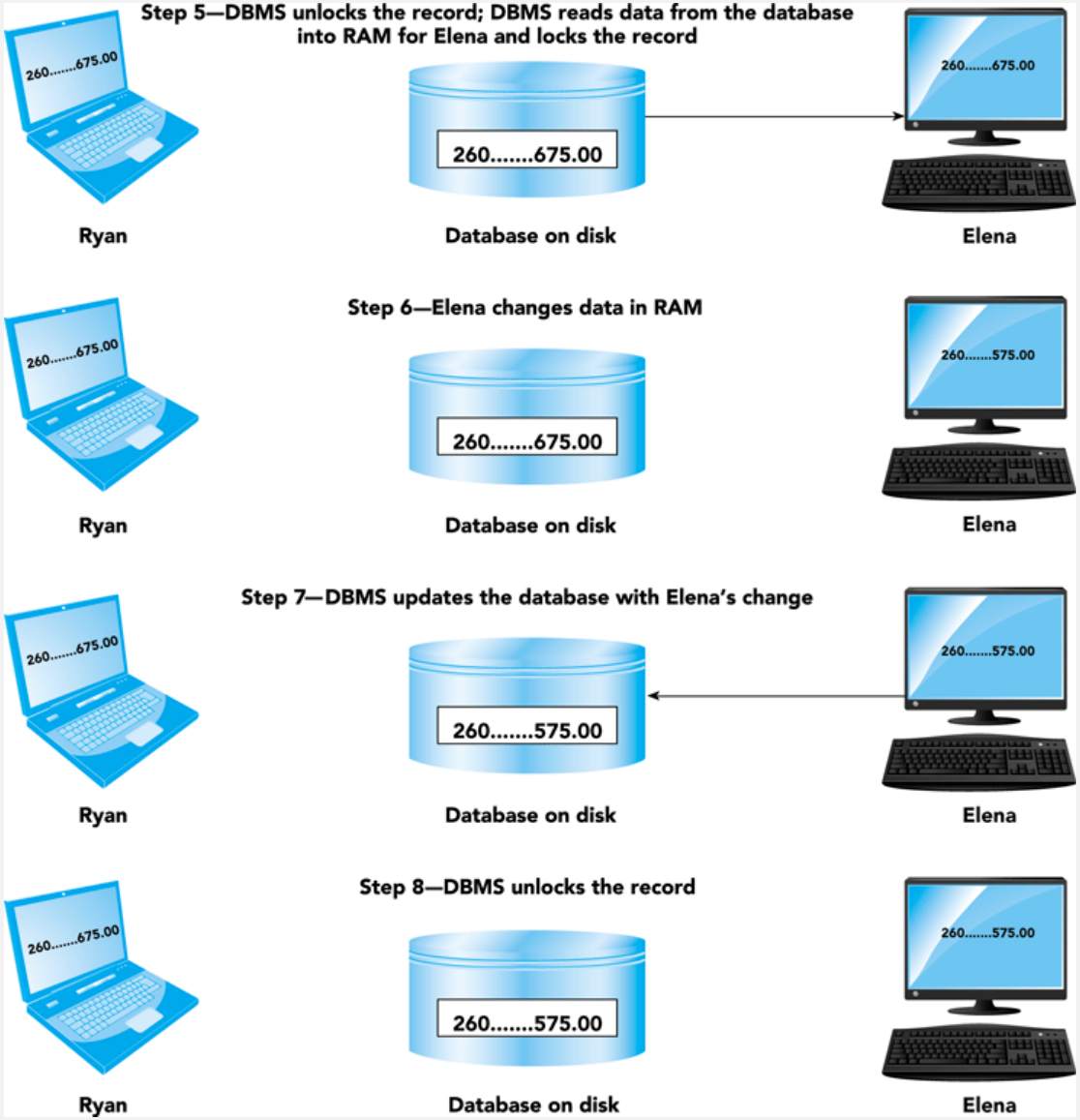


The DBMS uses a locking scheme to apply Ryan's and Elena's updates to the database

Two-Phase Locking (continued)



Two-Phase Locking (continued)



Deadlock

- **Deadlock** or **deadly embrace**
  - Two users hold a lock and require a lock on the resource that the other already has
  - To minimize occurrence, make sure all programs lock records in the same order whenever possible
- Managing deadlocks
  - DBMS detects and breaks any deadlock
  - DBMS chooses one user to be the **victim**

Deadlock (continued)

FIGURE 7-9: Two users experiencing deadlock

Locking on PC-Based DBMSs

- Usually more limited than locking facilities on enterprise DBMSs
- Programs can lock an entire table or an individual row within a table, but only one or the other
- Programs can release any or all of the locks they currently hold
- Programs can inquire whether a given row or table is locked

Recover Data

- **Recovery**: returning database to a correct state from an incorrect state
- Simplest recovery involves using backups
  - **Backup** or **save**: copy of database

Journaling

- **Journaling**: maintaining a **journal** or **log** of all updates
  - Log is available even if database is destroyed
- Information kept in log for each transaction:
  - Transaction ID
  - Date and time of each update
  - **Before image**
  - **After image**
  - Start of a transaction
  - Successful completion (**commit**) of a transaction

Forward Recovery

- DBA executes a DBMS recovery program
- Recovery program applies after images of committed transactions from log to database
- Improving performance of the recovery program
  - Apply the last after image of a record

Backward Recovery

- Database not in a valid state
  - Transactions stopped in midstream
  - Incorrect transactions
- **Backward recovery** or **rollback**
  - Undo problem transactions
  - Apply before images from log to undo their updates

Authentication

- **Authentication**: techniques for identifying the person attempting to access the DBMS
- **Password**: string of characters assigned by DBA to a user that must be entered for access
- **Biometrics**: identify users by physical characteristics such as fingerprints, voiceprints, handwritten signatures, and facial characteristics
- **Smart cards**: small plastic cards with built-in circuits containing processing logic to identify the cardholder