CIS 412
DATABASE MANAGEMENT SYSTEMS

Chapter 7

DBMS Functions

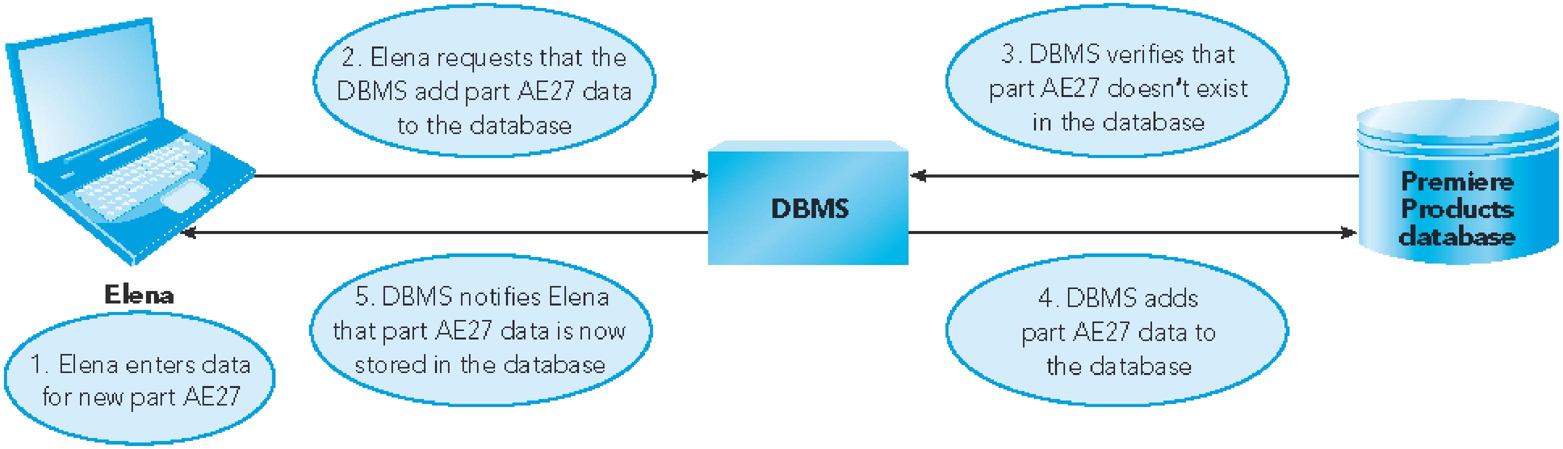
INTRODUCTION

- Functions of a DBMS
  - Update and retrieve data
  - Provide catalog services
  - Support concurrent update
  - Recover data
  - Provide security services
  - Provide data integrity features
  - Support data independence
  - Support data replication
  - Provide utility services

Update and Retrieve Data

- Fundamental capability of a DBMS
- Users don’t need to know how data is stored or manipulated
- Users add, change, and delete records during updates

Update and Retrieve Data (continued)



1. Elena enters data for new part AE27

2. Elena requests that the DBMS add part AE27 data to the database

3. DBMS verifies that part AE27 doesn't exist in the database

4. DBMS adds part AE27 data to the database

5. DBMS notifies Elena that part AE27 data is now stored in the database

FIGURE 7-1: Adding a new part to the Premiere Products database

Update and Retrieve Data (continued)

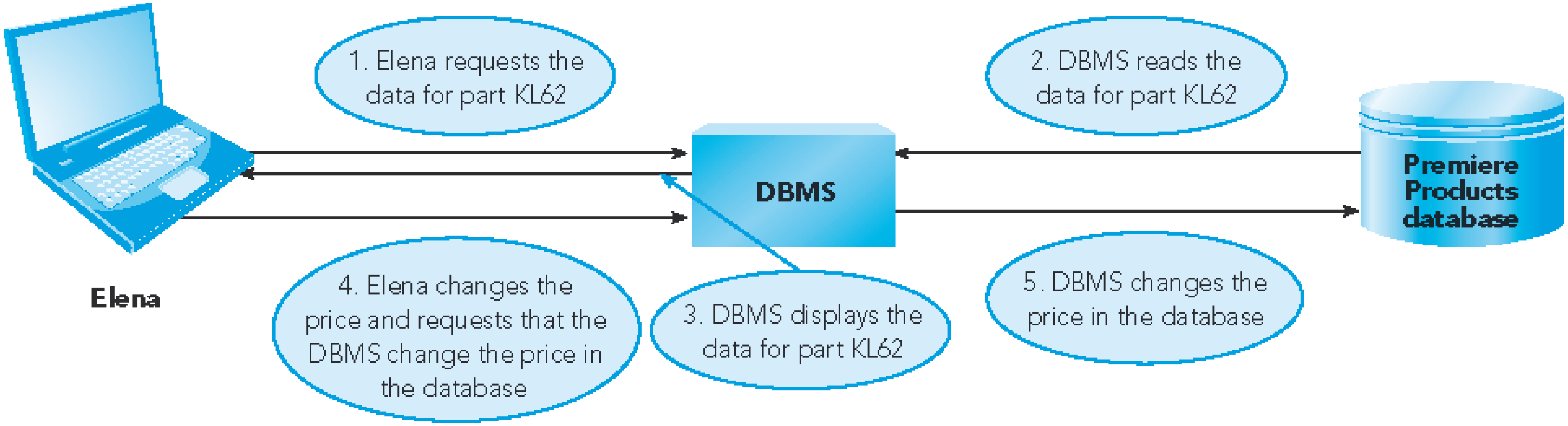

1. Elena requests the data for part KL62
2. DBMS reads the data for part KL62
DBMS
Premiere Products database
Elena
4. Elena changes the price and requests that the DBMS change the price in the database
3. DBMS displays the data for part KL62
5. DBMS changes the price in the database

FIGURE 7-2: Changing the price of a part in the Premiere Products database

Update and Retrieve Data (continued)

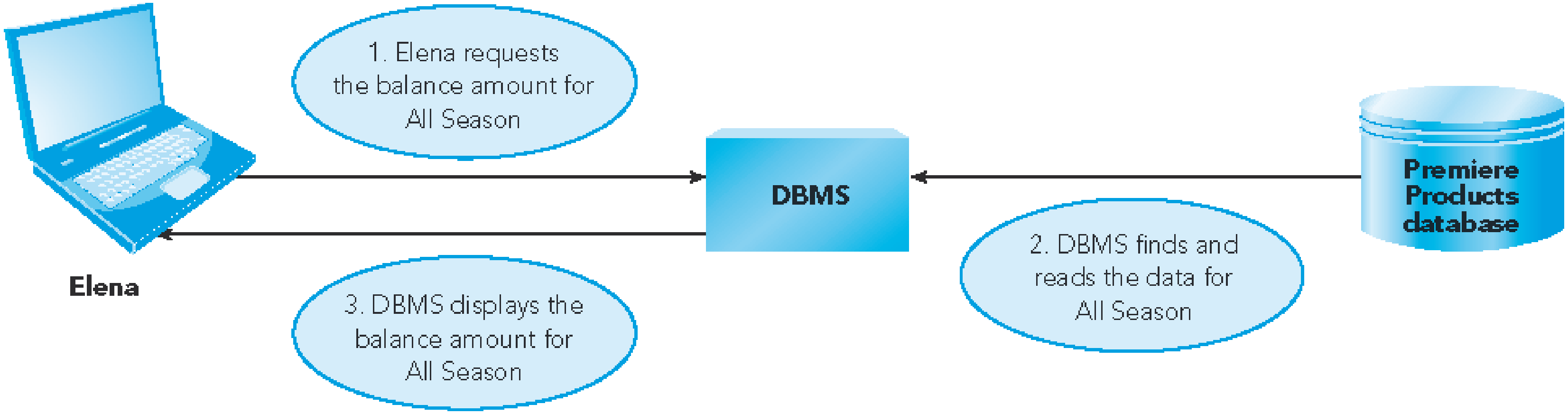

1. Elena requests the balance amount for All Season
DBMS
Premiere Products database
2. DBMS finds and reads the data for All Season
Elena
3. DBMS displays the balance amount for All Season

FIGURE 7-3: Retrieving a balance amount from the Premiere Products database

Provide Catalog Services

- **Metadata**: data about data
- Stores metadata and makes it accessible to users
- Enterprise DBMSs often have a **data dictionary** (a super catalog)

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

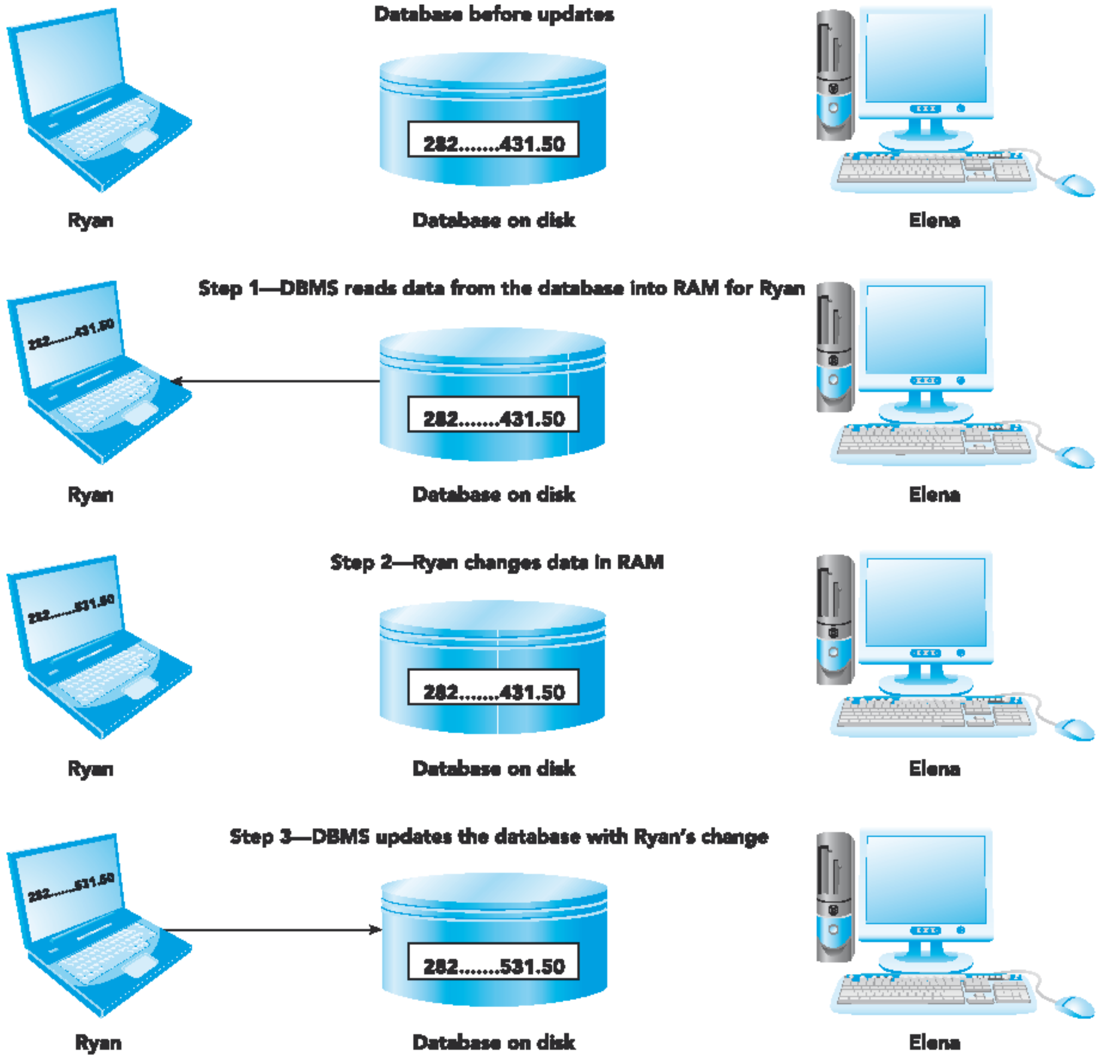


FIGURE 7-4: Ryan updates the database

The Concurrent Update Problem (continued)

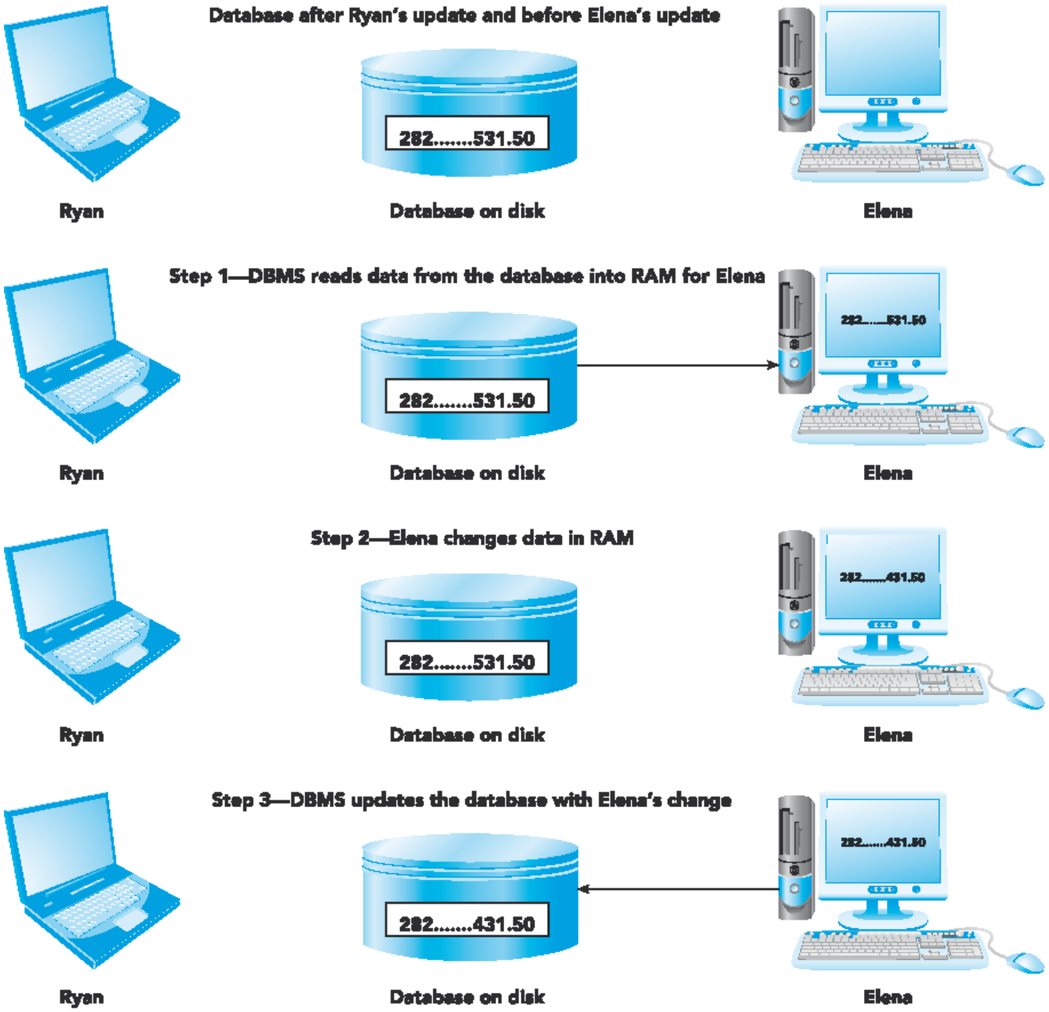


FIGURE 7-5: Elena updates the database

The Concurrent Update Problem (continued)

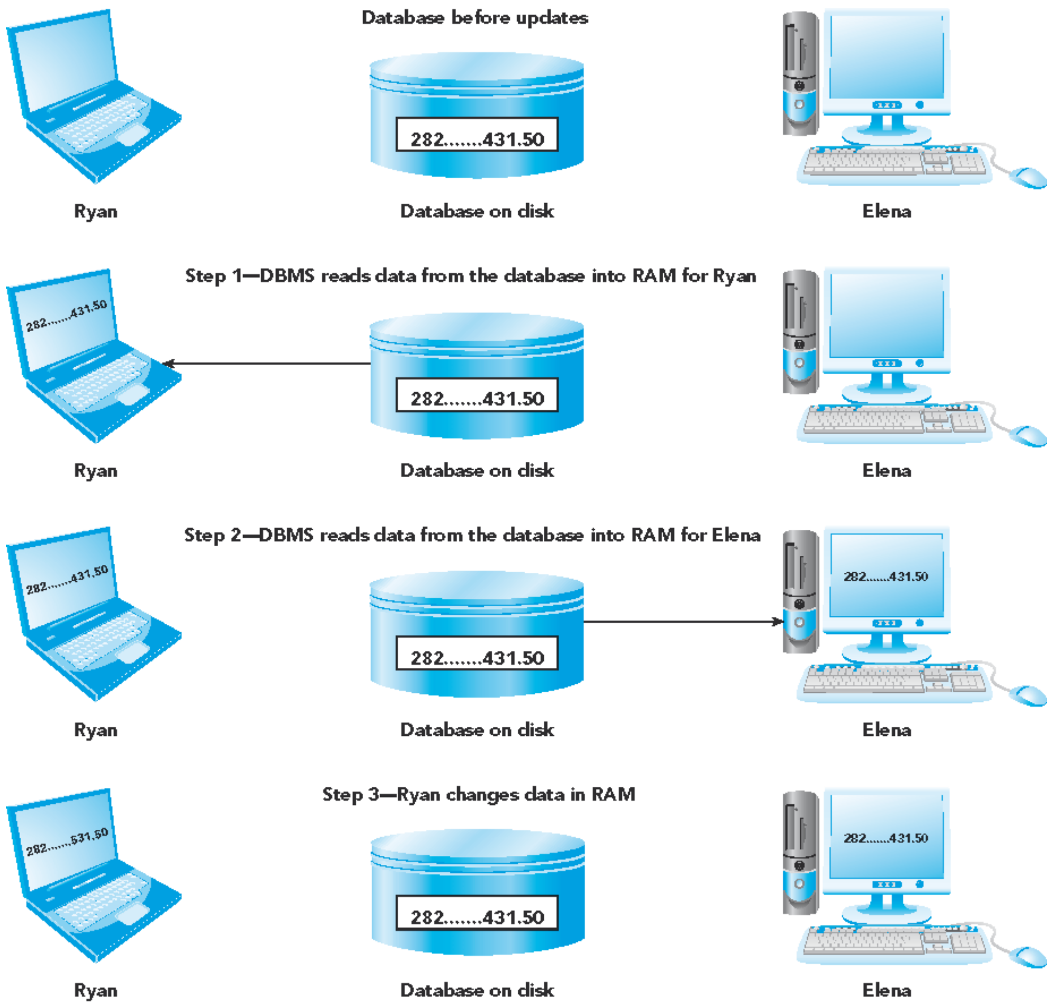


FIGURE 7-6: Ryan's and Elena's updates to the database result in a lost update

THE CONCURRENT UPDATE PROBLEM (CONTINUED)

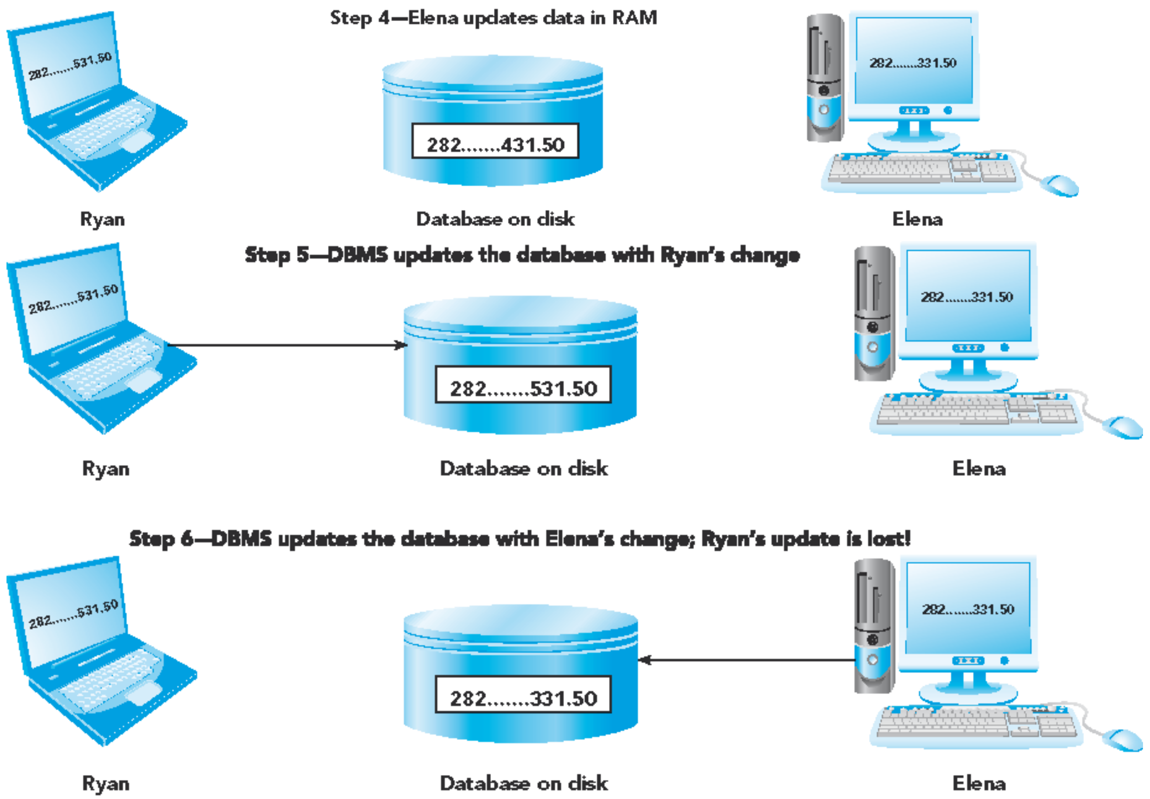
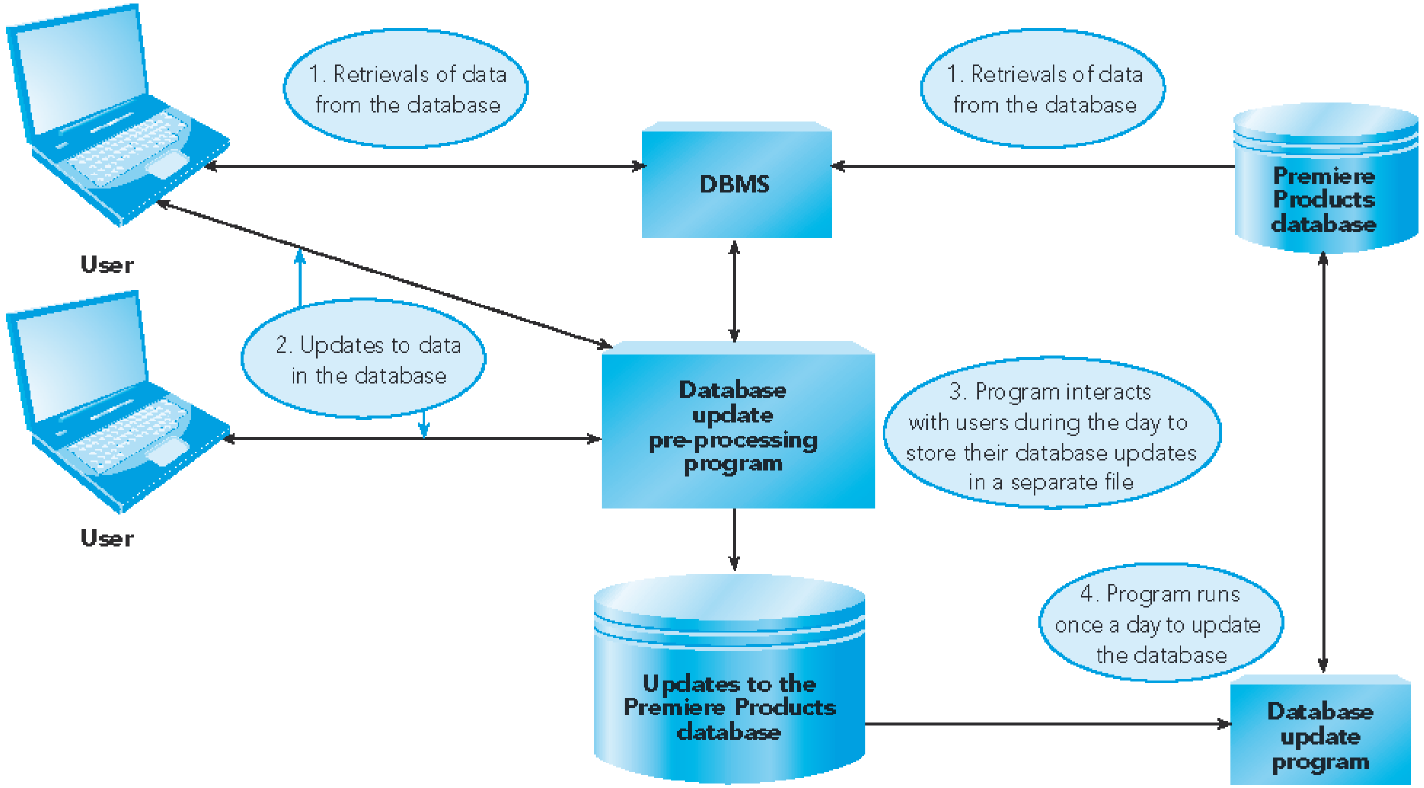


FIGURE 7-6: Ryan's and Elena's updates to the database result in a lost update (continued)

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

Avoiding the Lost Update Problem (continued)



1. Retrievals of data from the database
1. Retrievals of data from the database
DBMS
Premiere Products database
User
2. Updates to data in the database
Database update pre-processing program
3. Program interacts with users during the day to store their database updates in a separate file
User
4. Program runs once a day to update the database
Updates to the Premiere Products database
Database update program

FIGURE 7-7: Delaying updates to the Premiere Products database to avoid the lost update problem

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)

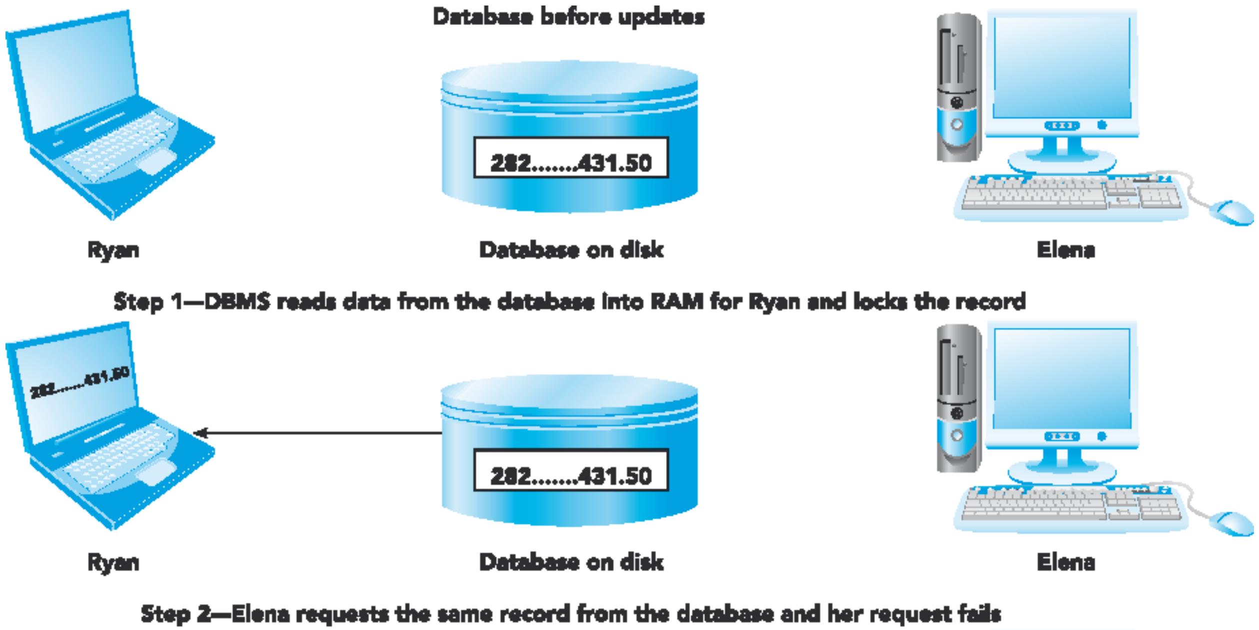
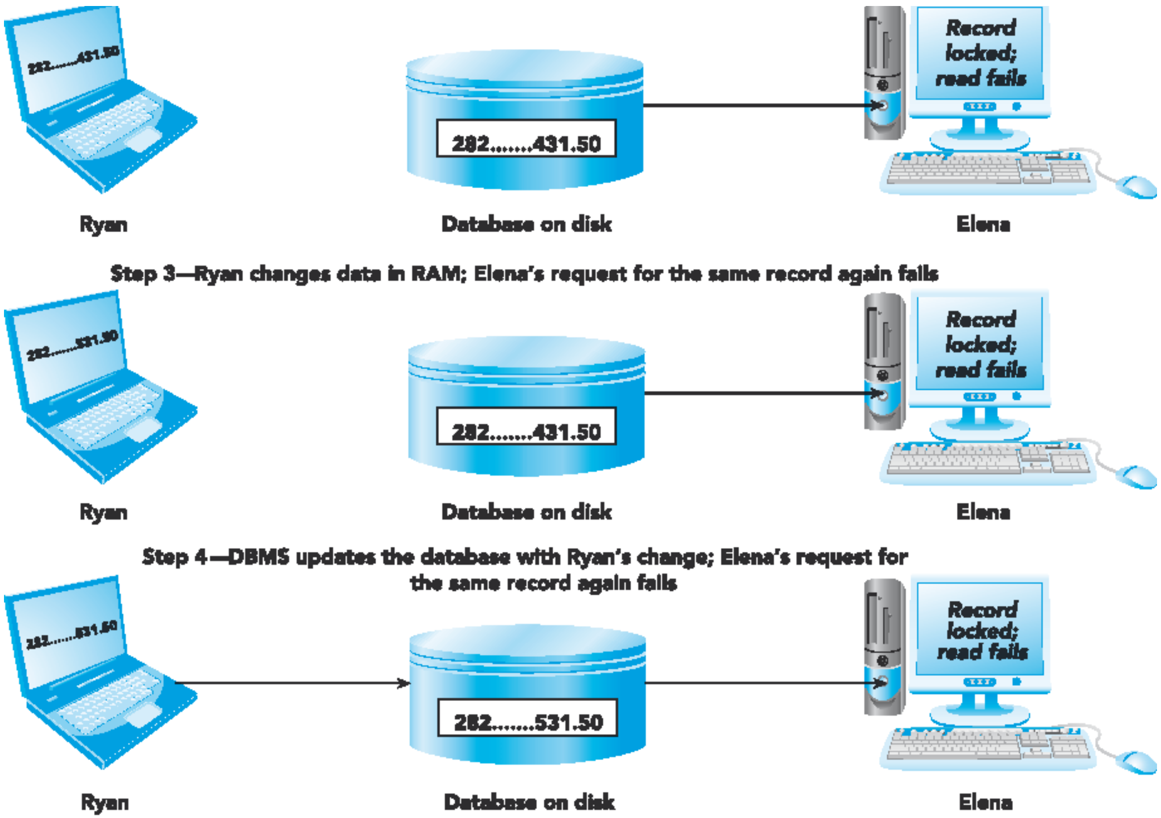

Database before updates
282.......431.50
Ryan
Database on disk
Elena
Step 1—DBMS reads data from the database into RAM for Ryan and locks the record
282.......431.50
282.......431.50
Ryan
Database on disk
Elena
Step 2—Elena requests the same record from the database and her request fails

FIGURE 7-8: The DBMS uses a locking scheme to apply Ryan's and Elena's updates to the database

TWO-PHASE LOCKING (CONTINUED)



Step 3—Ryan changes data in RAM; Elena's request for the same record again fails

Step 4—DBMS updates the database with Ryan's change; Elena's request for the same record again fails

FIGURE 7-8: The DBMS uses a locking scheme to apply Ryan's and Elena's updates to the database (continued)

Two-Phase Locking (continued)

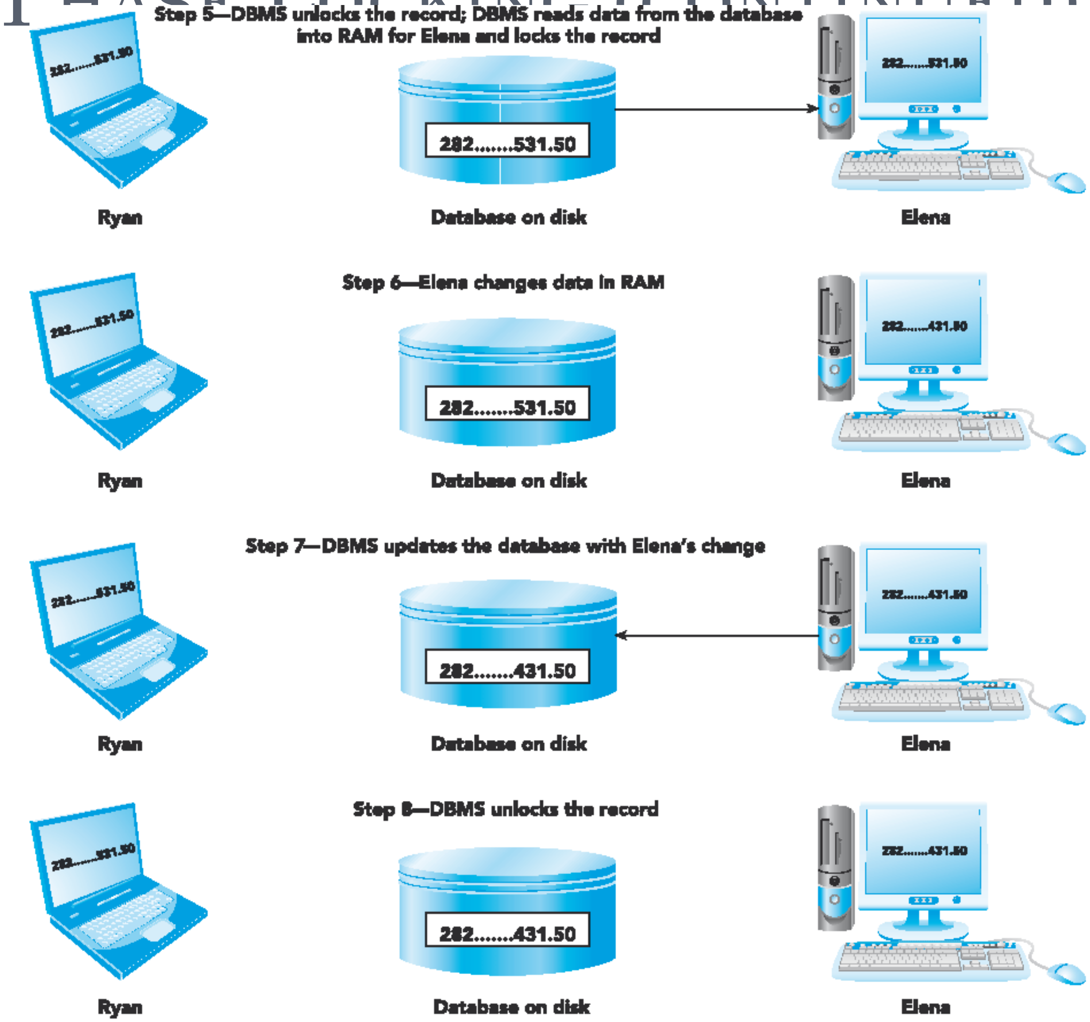


FIGURE 7-8: The DBMS uses a locking scheme to apply Ryan's and Elena's updates to the database (continued)

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)

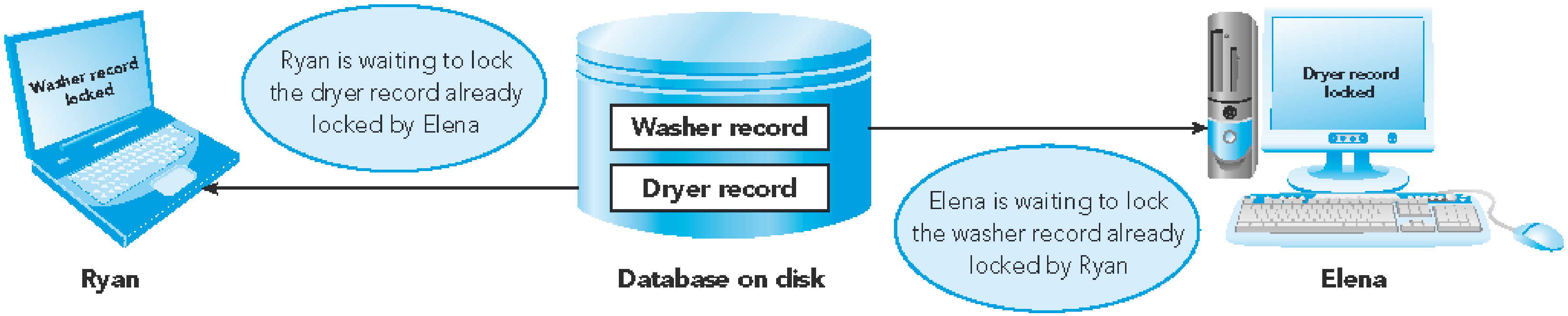

Washer record locked
Ryan is waiting to lock the dryer record already locked by Elena
Washer record
Dryer record
Dryer record locked
Elena is waiting to lock the washer record already locked by Ryan
Ryan
Database on disk
Elena

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

Timestamping

- DBMS assigns each database update a unique time (**timestamp**) when the update started
- Advantages
    - Avoids need to lock rows
    - Eliminates processing time needed to apply and release locks and to detect and resolve deadlocks
- Disadvantages
    - Additional disk and memory space
    - Extra processing time

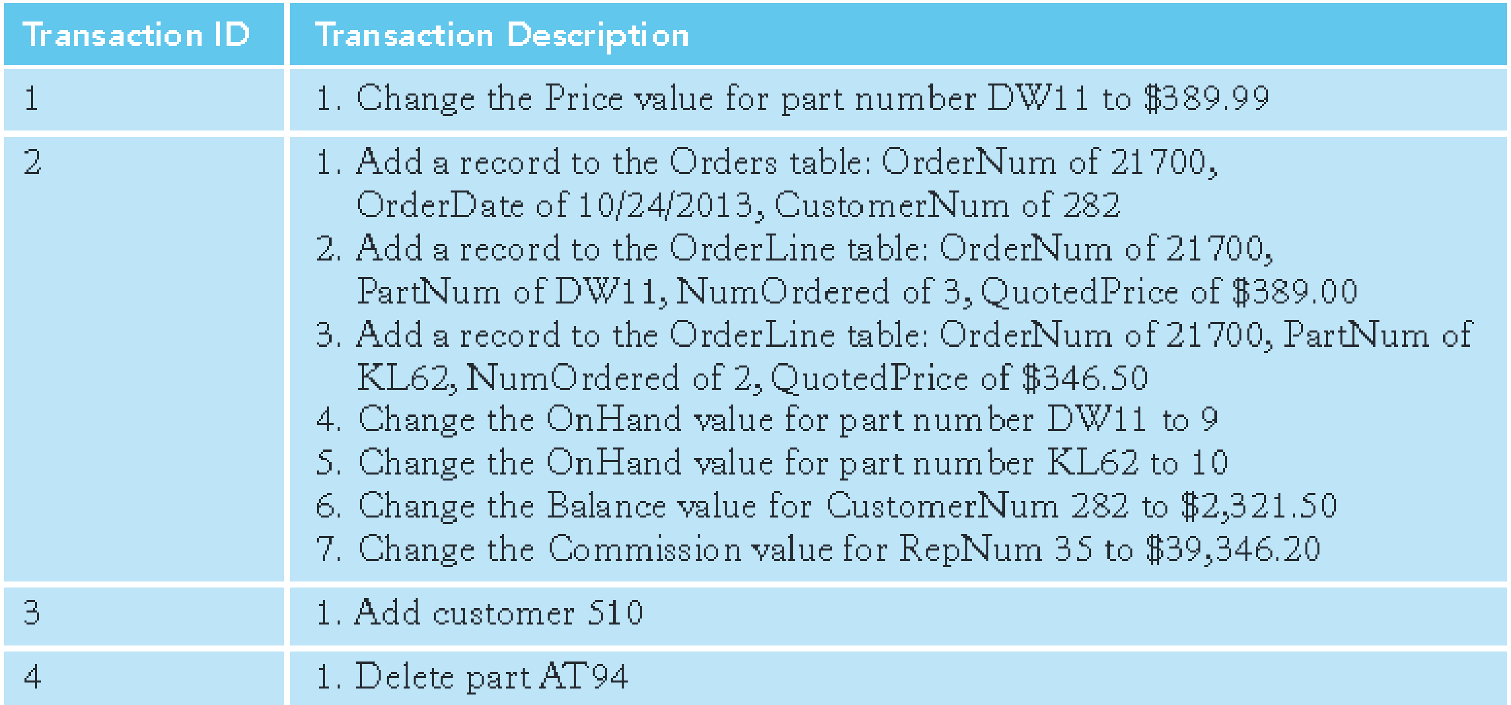
Recover Data

- **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

JOURNALING (CONTINUED)



Transaction ID	Transaction Description
1	1. Change the Price value for part number DW11 to $389.99
2	1. Add a record to the Orders table: OrderNum of 21700, OrderDate of 10/24/2013, CustomerNum of 282 2. Add a record to the OrderLine table: OrderNum of 21700, PartNum of DW11, NumOrdered of 3, QuotedPrice of $389.00 3. Add a record to the OrderLine table: OrderNum of 21700, PartNum of KL62, NumOrdered of 2, QuotedPrice of $346.50 4. Change the OnHand value for part number DW11 to 9 5. Change the OnHand value for part number KL62 to 10 6. Change the Balance value for CustomerNum 282 to $2,321.50 7. Change the Commission value for RepNum 35 to $39,346.20
3	1. Add customer 510
4	1. Delete part AT94

FIGURE 7-10: Four sample transactions

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

Forward Recovery (continued)

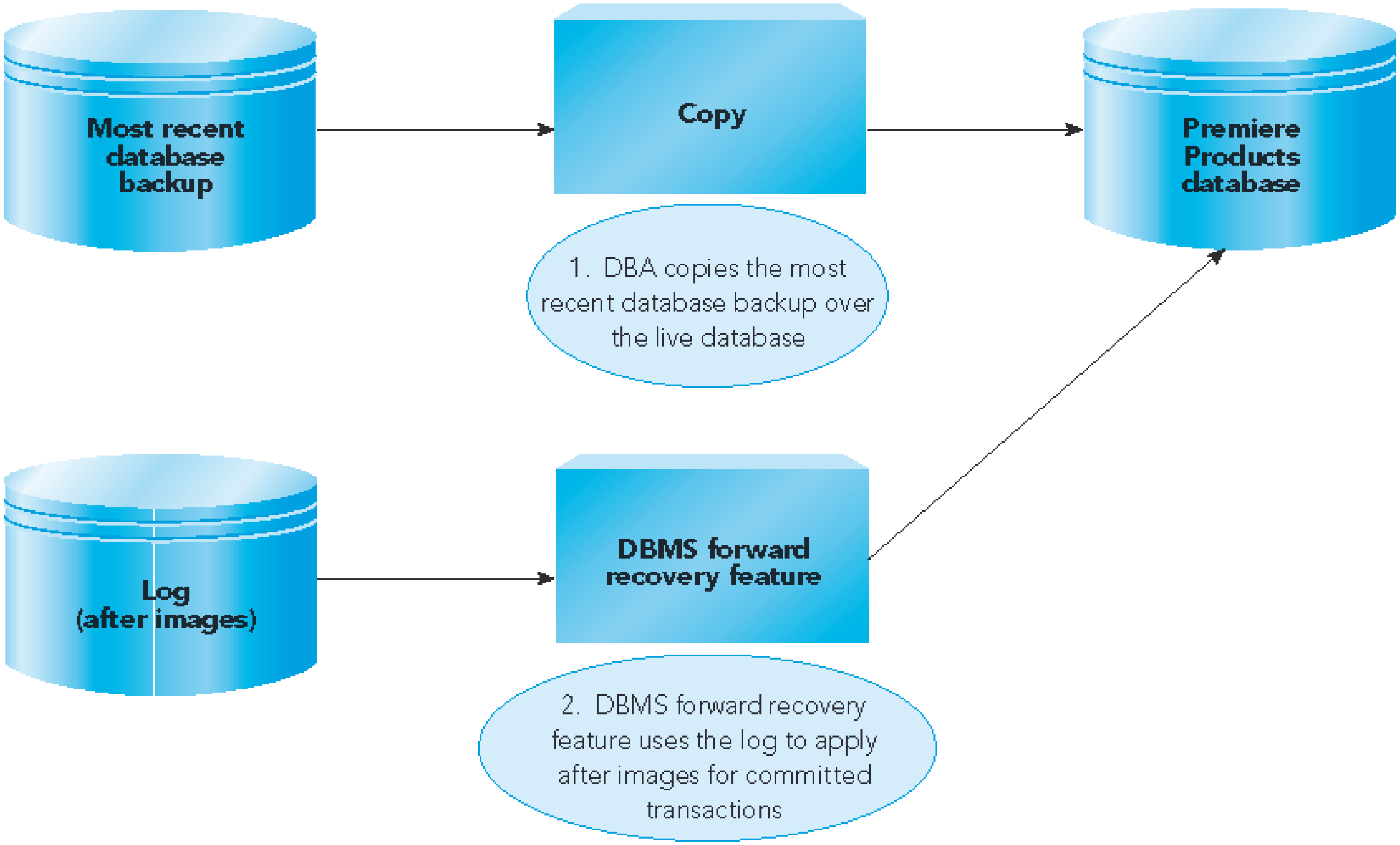

Most recent database backup
Copy
Premiere Products database
1. DBA copies the most recent database backup over the live database
Log (after images)
DBMS forward recovery feature
2. DBMS forward recovery feature uses the log to apply after images for committed transactions

FIGURE 7-12: Forward recovery

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

Backward Recovery (continued)

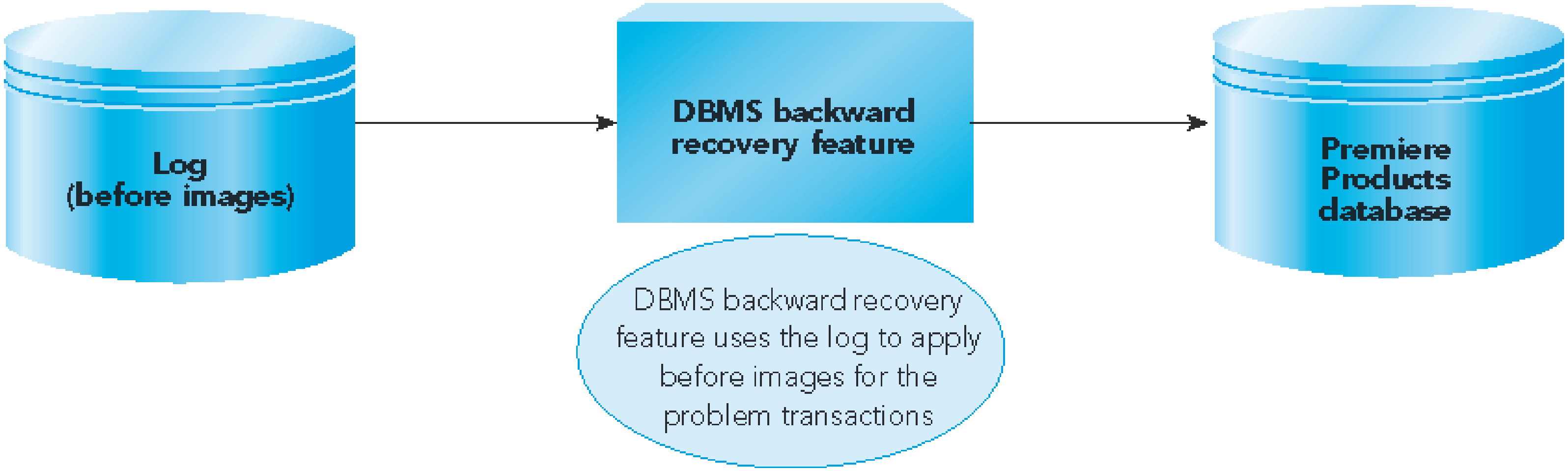

Log
(before images)
DBMS backward recovery feature
Premiere Products database
DBMS backward recovery feature uses the log to apply before images for the problem transactions

FIGURE 7-13: Backward recovery

Recovery on PC-Based DBMSs

- Sophisticated recovery features not available on PC-based DBMSs
- Regularly make backup copies using DBMS
  - Use most recent backup for recovery
- Systems with large number of updates between backups
  - Recovery features not supplied by DBMS need to be included in application programs

Provide Security Services

- Security: prevention of unauthorized access, either intentional or accidental, to a database
- Most common security features used by DBMSs:
  - Encryption
  - Authentication
  - Authorizations
  - Views

Encryption

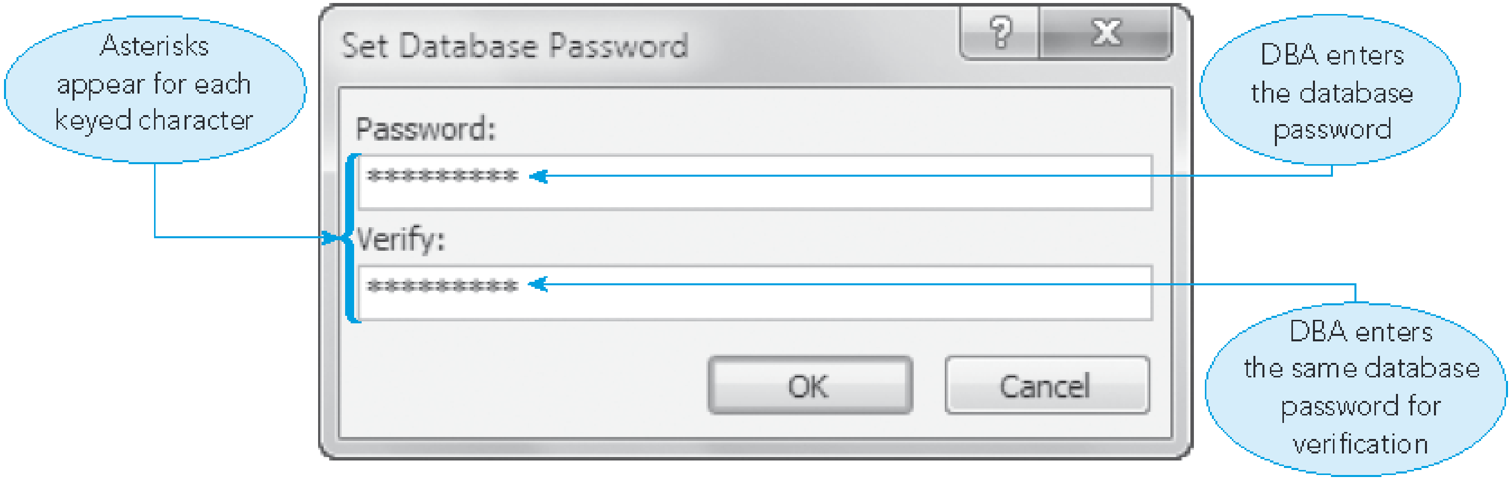
- **Encryption**: converts data to a format indecipherable to another program and stores it in an encrypted format
- Encryption process is transparent to a legitimate user
- **Decrypting**: reversing the encryption
- In Access, encrypt a database with a password

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

AUTHENTICATION (CONTINUED)

- **Database password**: string of characters assigned to database that users must enter for accessing the database



Set Database Password

Password: **********

Verify: **********

Asterisks appear for each keyed character

DBA enters the database password

DBA enters the same database password for verification

FIGURE 7-14: Assigning a database password to the Premiere Products database

Authorizations

- DBA can use **authorization rules** to specify which users have what type of access to which data
- **Permissions**: specify what kind of access the user has to objects in the database
- **Workgroups**: groups of users

Views

- **View**: snapshot of certain data in the database at a given moment in time
- Can be used for security purposes

Privacy

- **Privacy**: right of individuals to have certain information about them kept confidential
- Laws and regulations dictate some privacy rules
- Companies institute additional privacy rules

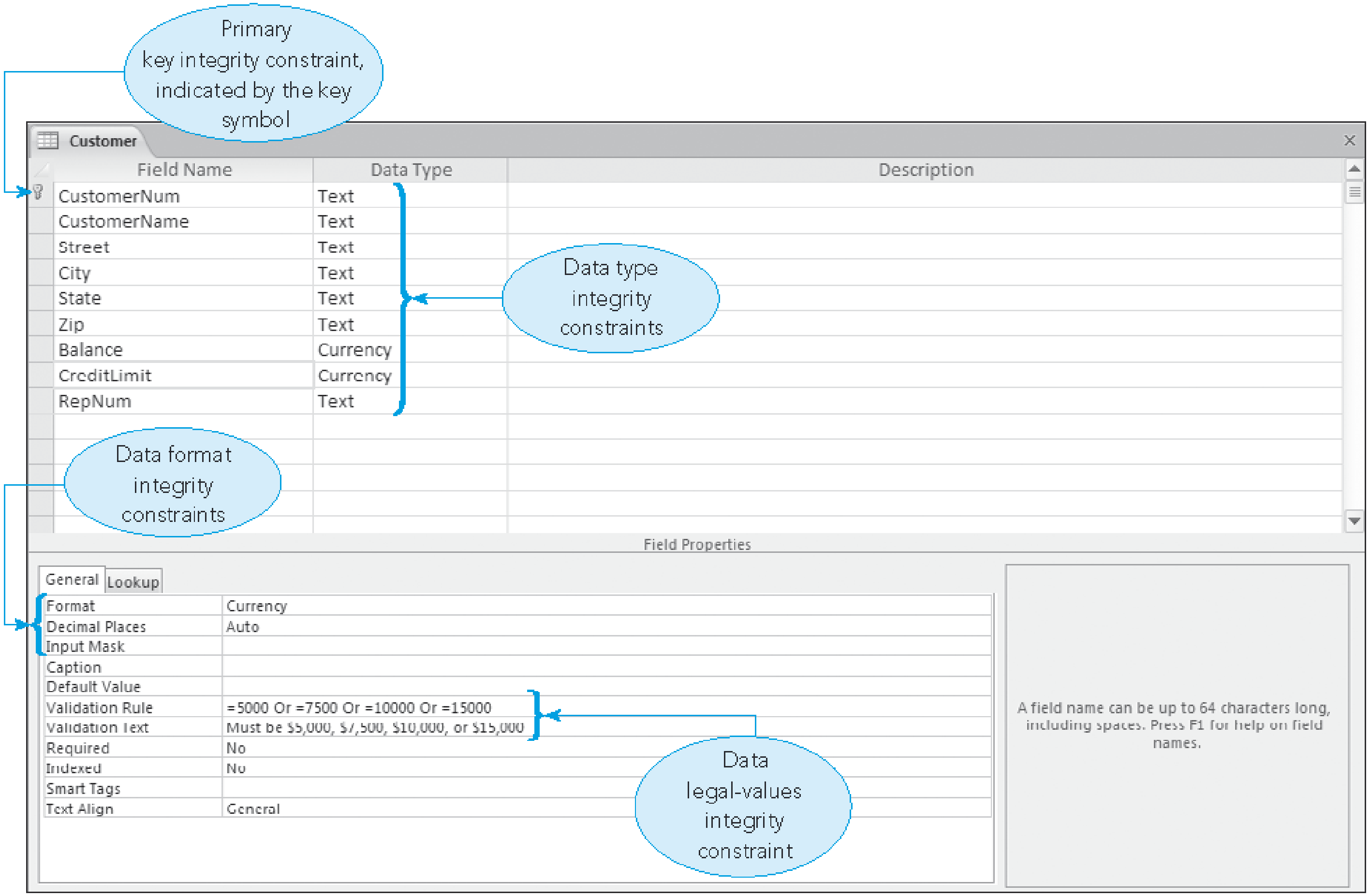
Provide Data Integrity Features

- Rules followed to ensure data is accurately and consistently updated
- Key integrity
  - Foreign key and primary key constraints
- Data integrity
  - Data type
  - Legal values
  - Format

Provide Data Integrity Features (continued)

- Four ways of handling integrity constraints:
  1. Constraint is ignored
  2. Responsibility for constraint enforcement placed on users
  3. Responsibility for constraint enforcement placed on programmers
  4. Responsibility for constraint enforcement placed on DBMS

PROVIDE DATA INTEGRITY FEATURES (CONTINUED)



Primary key integrity constraint, indicated by the key symbol
Data type integrity constraints
Data format integrity constraints
Data legal-values integrity constraint
Customer
Field Properties
General
Lookup

Field Name	Data Type	Description
CustomerNum	Text	
CustomerName	Text	
Street	Text	
City	Text	
State	Text	
Zip	Text	
Balance	Currency	
CreditLimit	Currency	
RepNum	Text	

Format	Currency
Decimal Places	Auto
Input Mask	
Caption	
Default Value	
Validation Rule	=5000 Or =7500 Or =10000 Or =15000
Validation Text	Must be $5,000, $7,500, $10,000, or $15,000
Required	No
Indexed	No
Smart Tags	
Text Align	General

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

FIGURE 7-16: Example of integrity constraints in Access

Support Data Independence

- **Data independence**: can change database structure without needing to change programs that access the database
- Types of changes:
  - Adding a field
  - Changing a field property (such as length)
  - Creating an index
  - Adding or changing a relationship

Adding a Field

- Don’t need to change any program except those programs using the new field
- SQL SELECT * FROM command will present an extra field
  - Solution: list the required fields in an SQL SELECT command instead of using *

Changing the Length of a Field

- Generally, don’t need to change programs
- Need to change the program if:
  - Certain portion of screen or report is set aside for the field and the space cannot fit the new length

Creating an Index

- To create an index, enter a simple SQL command or select a few options
- Most DBMSs use the new index automatically
- For some DBMSs, need to make minor changes in already existing programs

Adding or Changing a Relationship

- Trickiest of all
- May need to restructure database

Support Data Replication

- **Replicated**: duplicated
- Manage multiple copies of same data in multiple locations
- Maintained for performance or other reasons
- Ease of access and portability
- **Replicas**: copies
- **Synchronization**: DBMS exchanges all updated data between master database and a replica

Support Data Replication (continued)

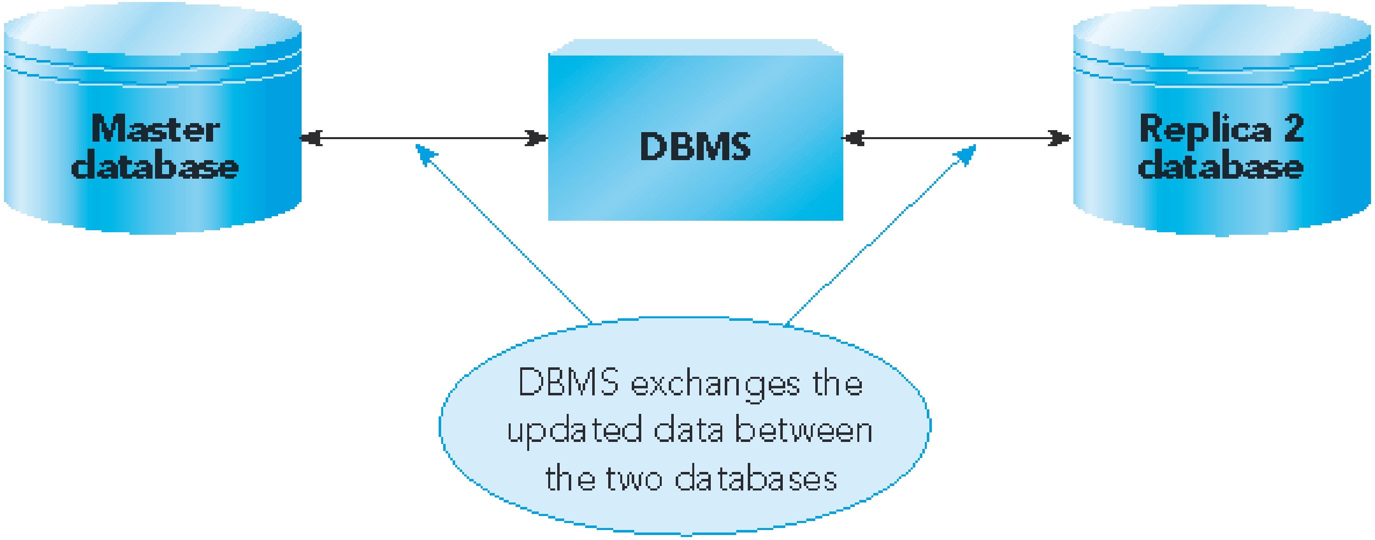


FIGURE 7-18: DBMS synchronizes two databases in a replica set

Provide Utility Services

- **Utility services** assist in general database maintenance
- Change database structure
- Add new indexes and delete indexes
- Use services available from operating system
- Export and import data
- Support for easy-to-use edit and query capabilities, screen generators, report generators, etc.

Provide Utility Services (continued)

- Support for procedural and nonprocedural languages
  - **Procedural language**: must tell computer precisely how a given task is to be accomplished
  - **Nonprocedural language**: describe task you want computer to accomplish
- Easy-to-use menu-driven or switchboard-driven interface

Summary

- DBMS allows users to update and retrieve data in a database without needing to know how data is structured on disk or manipulated
- DBMS must store metadata (data about the data) and make this data accessible to users
- DBMS must support concurrent update
- Locking denies access by other users to data while DBMS processes one user’s updates
- During *deadlock* and *deadly embrace*, two or more users are waiting for the other user to release a lock before they can proceed

SUMMARY (CONTINUED)

- In timestamping, DBMS processes updates to a database in timestamp order
- DBMS must provide methods to recover a database in the event the database is damaged
- DBMSs provide facilities for periodically making a backup copy of the database
- Enterprise DBMSs maintain a log or journal of all database updates since the last backup; log is used in recovery process

Summary (continued)

- DBMSs provide security features (encryption, authentication, authorizations, and views) to prevent unauthorized access to a database
- DBMS must follow rules or integrity constraints (key integrity constraints and data integrity constraints) so that it updates data accurately and consistently
- DBMS must support data independence
- DBMS must have facility to handle data replication
- DBMS must provide utility services that assist in general maintenance of a database