Outline

- Types of Databases and Database Applications
- Basic Definitions
- Typical DBMS Functionality
- Example of a Database (UNIVERSITY)
- Main Characteristics of the Database Approach
- Database Users
- Advantages of Using the Database Approach
- When Not to Use Databases

Types of Databases and Database Applications

- Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:
  - Multimedia Databases
  - Geographic Information Systems (GIS)
  - Data Warehouses
  - Real-time and Active Databases
  - Many other applications
- First part of book focuses on traditional applications
- *A number of recent applications are described later in the book (for example, Chapters 24,26,28,29,30)*

Basic Definitions

- **Database:**
  - A collection of related data.
- **Data:**
  - Known facts that can be recorded and have an implicit meaning.
- **Mini-world:**
  - Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.
- **Database Management System (DBMS):**
  - A software package/ system to facilitate the creation and maintenance of a computerized database.
- **Database System:**
  - The DBMS software together with the data itself. Sometimes, the applications are also included.

Simplified database system environment

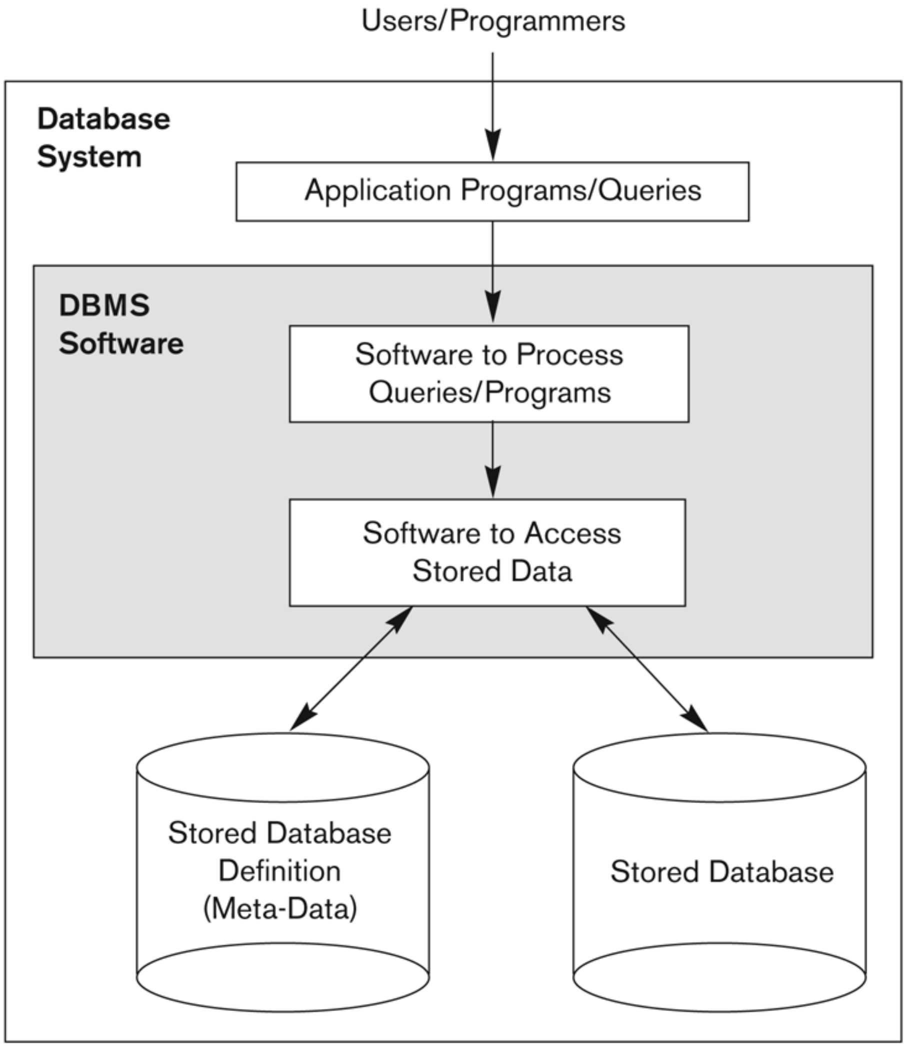
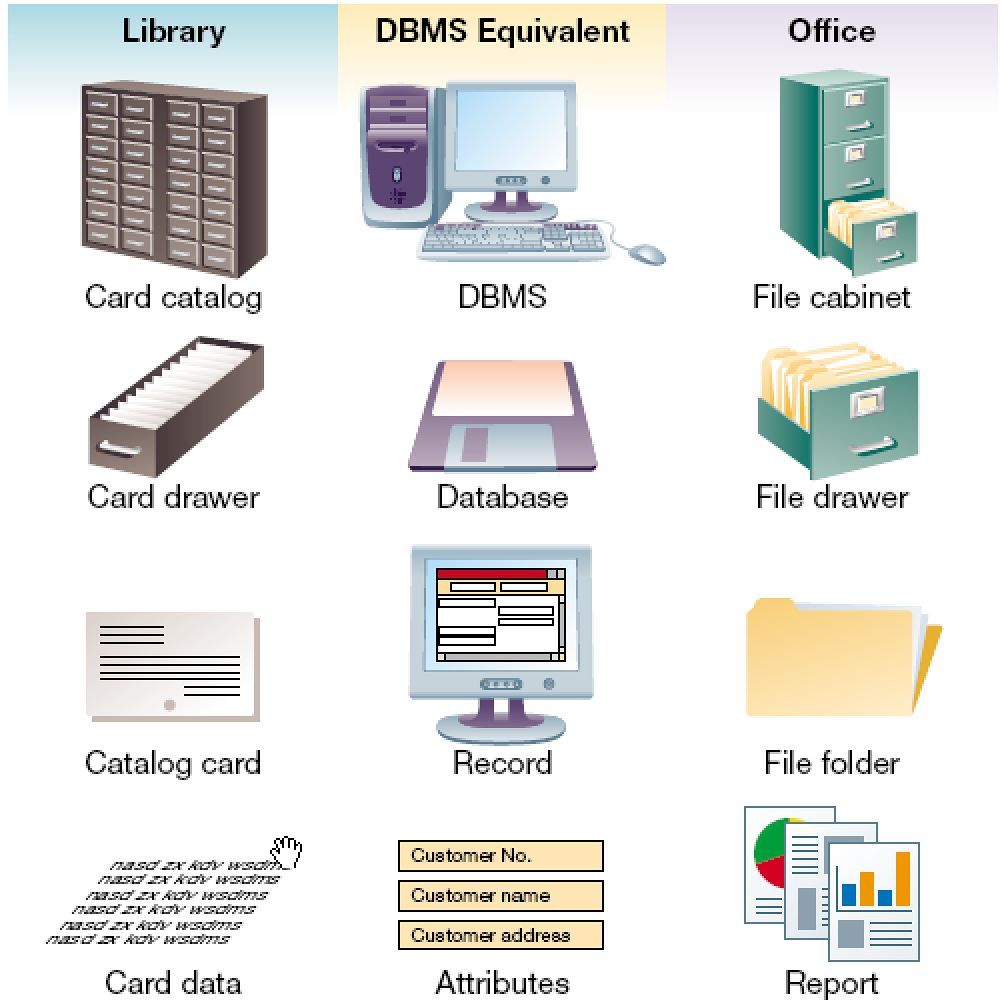

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Figure 1.1
A simplified database system environment.

Relationship of DBMS Concepts to Others?



Library	DBMS Equivalent	Office
Card catalog	DBMS	File cabinet
Card drawer	Database	File drawer
Catalog card	Record	File folder
Card data	Attributes	Report

Figure 3.1 ➥ Computers make the process of storing and managing data much easier.

Levels of a DBMS

Level	Term	Term Definitions
Lowest	Field	Individual characteristics about an ENTITY. Fields are also called attributes or columns depending on the type of DBMS
	Record	A group of fields or attributes to describe a single instance of an ENTITY. These are also called rows depending on the DBMS
	File	A collection of records or instances for a given ENTITY. These are also called tables depending on the DBMS
Highest	Database	A collection of files or entities containing information to support a given system or a particular topic area

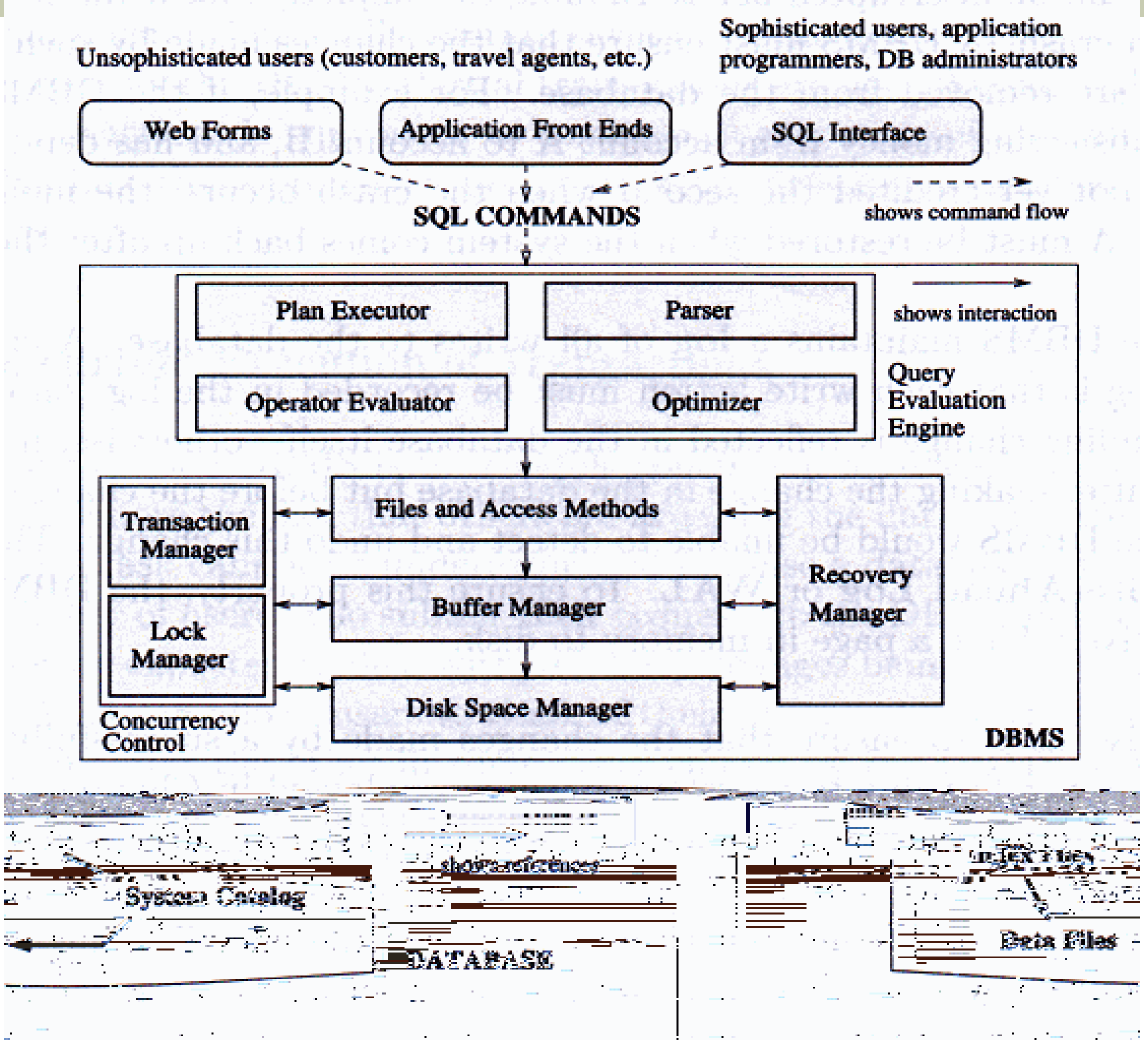
Typical DBMS Functionality

- *Define* a particular database in terms of its data types, structures, and constraints
- *Construct* or Load the initial database contents on a secondary storage medium
- *Manipulating* the database:
  - Retrieval: Querying, generating reports
  - Modification: Insertions, deletions and updates to its content
  - Accessing the database through Web applications
- *Processing* and *Sharing* by a set of concurrent users and application programs – yet, keeping all data valid and consistent

Typical DBMS Functionality

- Other features:
  - Protection or Security measures to prevent unauthorized access
  - “Active” processing to take internal actions on data
  - Presentation and Visualization of data
  - Maintaining the database and associated programs over the lifetime of the database application
    - Called database, software, and system maintenance

Architecture of a DBMS



Example of a Database (with a Conceptual Data Model)

- **Mini-world for the example:**
  - Part of a UNIVERSITY environment.
- **Some mini-world *entities*:**
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - (academic) DEPARTMENTs
  - INSTRUCTORs

Example of a Database (with a Conceptual Data Model)

- **Some mini-world *relationships*:**
  - SECTIONs *are of specific* COURSEs
  - STUDENTs *take* SECTIONs
  - COURSEs *have prerequisite* COURSEs
  - INSTRUCTORs *teach* SECTIONs
  - COURSEs *are offered by* DEPARTMENTs
  - STUDENTs *major in* DEPARTMENTs

- Note: The above entities and relationships are typically expressed in a conceptual data model, such as the ENTITY-RELATIONSHIP data model (see Chapters 3, 4)

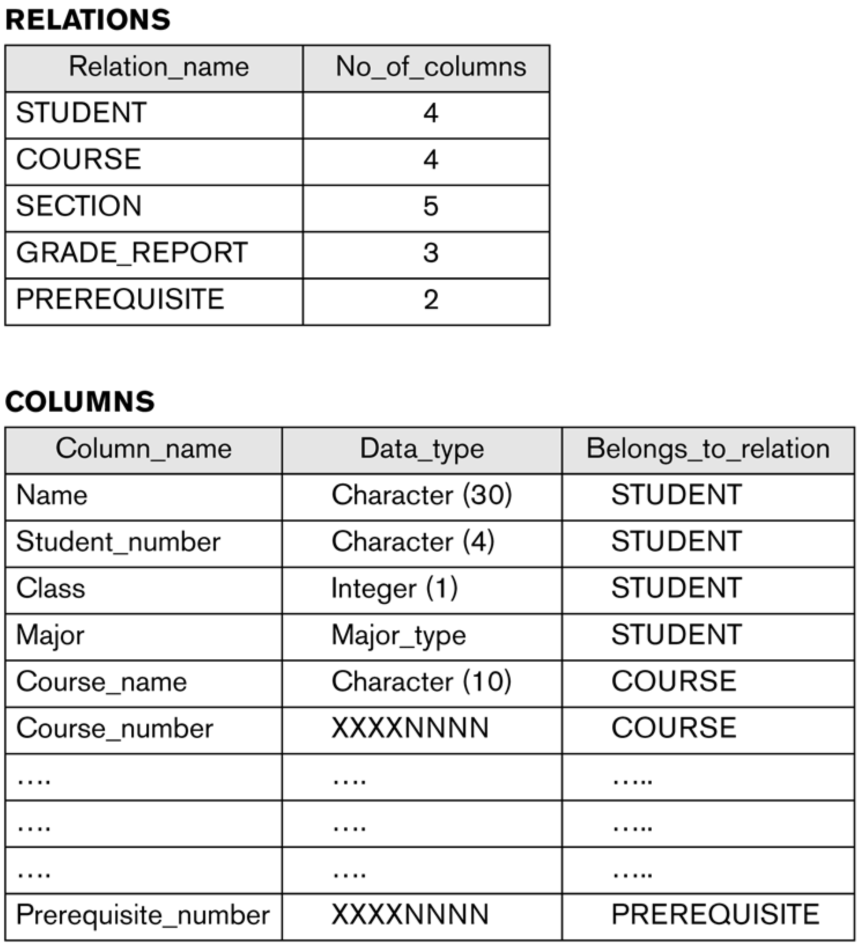
Example of a simple database

The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Main Characteristics of the Database Approach

- **Self-describing nature of a database system:**
  - A DBMS **catalog** stores the description of a particular database (e.g. data structures, types, and constraints)
  - The description is called **meta-data**.
  - This allows the DBMS software to work with different database applications.
- **Insulation between programs and data:**
  - Called **program-data independence**.
  - Allows changing data structures and storage organization without having to change the DBMS access programs.

Example of a simplified database catalog



RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major_type is defined as an enumerared type with all known majors. XXXXNNNN is used to define a type with four alpha characters followed by four digits

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

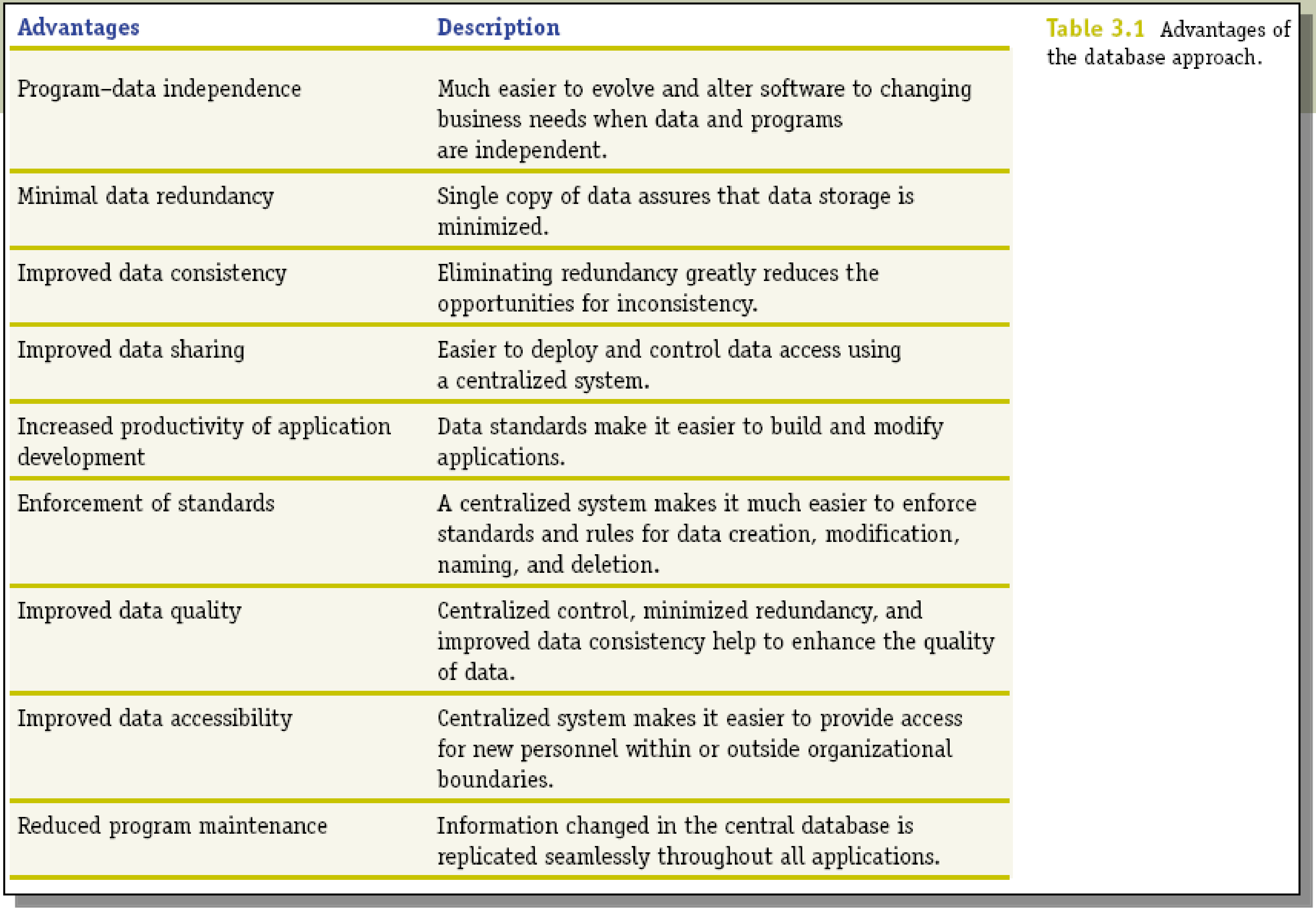
Main Characteristics of the Database Approach (continued)

- **Data Abstraction:**
  - A **data model** is used to hide storage details and present the users with a conceptual view of the database.
  - Programs refer to the data model constructs rather than data storage details
- **Support of multiple views of the data:**
  - Each user may see a different view of the database, which describes **only** the data of interest to that user.

Main Characteristics of the Database Approach (continued)

- **Sharing of data and multi-user transaction processing:**
  - Allowing a set of **concurrent users** to retrieve from and to update the database.
  - *Concurrency control* within the DBMS guarantees that each **transaction** is correctly executed or aborted
  - *Recovery* subsystem ensures each completed transaction has its effect permanently recorded in the database
  - **OLTP** (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

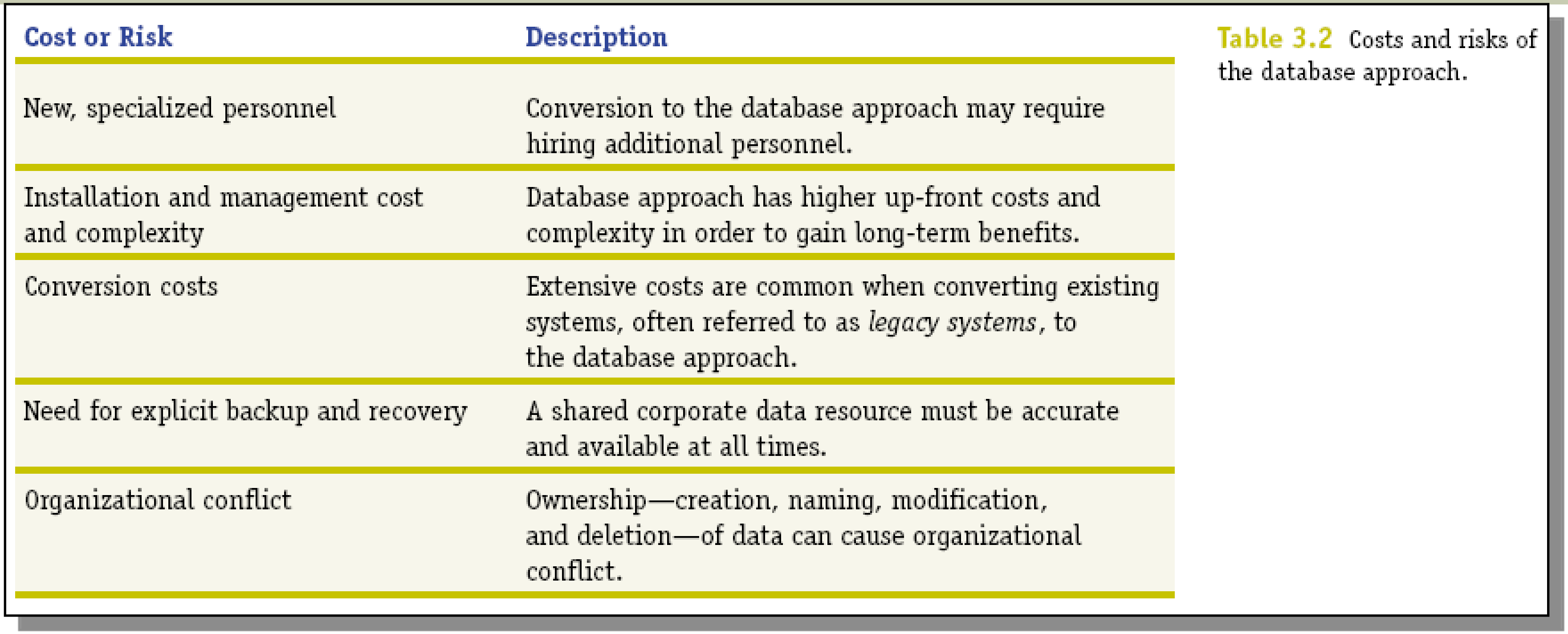
Advantages of the Database Approach



Advantages	Description
Program-data independence	Much easier to evolve and alter software to changing business needs when data and programs are independent.
Minimal data redundancy	Single copy of data assures that data storage is minimized.
Improved data consistency	Eliminating redundancy greatly reduces the opportunities for inconsistency.
Improved data sharing	Easier to deploy and control data access using a centralized system.
Increased productivity of application development	Data standards make it easier to build and modify applications.
Enforcement of standards	A centralized system makes it much easier to enforce standards and rules for data creation, modification, naming, and deletion.
Improved data quality	Centralized control, minimized redundancy, and improved data consistency help to enhance the quality of data.
Improved data accessibility	Centralized system makes it easier to provide access for new personnel within or outside organizational boundaries.
Reduced program maintenance	Information changed in the central database is replicated seamlessly throughout all applications.

Table 3.1 Advantages of the database approach.

Costs or Risks of the Database Approach



Cost or Risk	Description
New, specialized personnel	Conversion to the database approach may require hiring additional personnel.
Installation and management cost and complexity	Database approach has higher up-front costs and complexity in order to gain long-term benefits.
Conversion costs	Extensive costs are common when converting existing systems, often referred to as *legacy systems*, to the database approach.
Need for explicit backup and recovery	A shared corporate data resource must be accurate and available at all times.
Organizational conflict	Ownership—creation, naming, modification, and deletion—of data can cause organizational conflict.

Table 3.2 Costs and risks of the database approach.

File Processing vs Database Approach

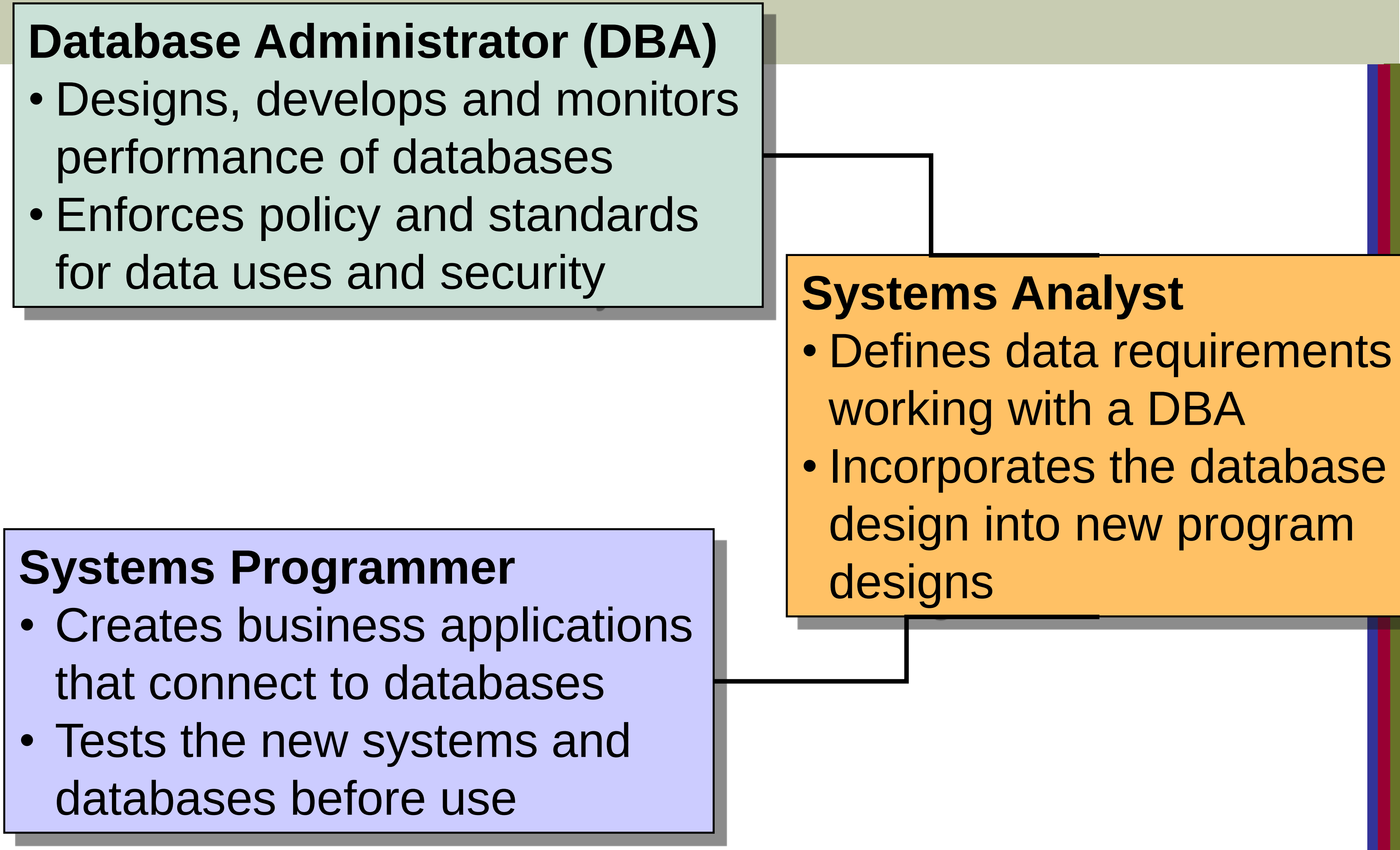
File Processing Approach (Old School)

- **Storage Media:** Sequential tapes or files
- **Data:** stored in long sequential files
- **Organization:** redundant data in multiple files
- **Efficiency:** data embedded to support processing
- **Updates:** requires multiple updates in many files
- **Processing:** slower query/faster processing

Data Base Approach (New School-TODAY)

- **Storage Media:** Direct Access Storage Device (DASD)
- **Data:** stored in related tables
- **Organization:** redundant data minimized/eliminated
- **Efficiency:** data only stored only in tables
- **Updates:** requires few or one update for a data field
- **Processing:** faster query/slower processing

Roles in Database Development and Use



Database Administrator (DBA)

- Designs, develops and monitors performance of databases
- Enforces policy and standards for data uses and security

Systems Analyst

- Defines data requirements working with a DBA
- Incorporates the database design into new program designs

Systems Programmer

- Creates business applications that connect to databases
- Tests the new systems and databases before use

Database Users

- Users may be divided into
  - Those who actually use and control the database content, and those who design, develop and maintain database applications (called “Actors on the Scene”), and
  - Those who design and develop the DBMS software and related tools, and the computer systems operators (called “Workers Behind the Scene”).

Database Users

- Actors on the scene
  - **Database administrators:**
    - Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.
  - **Database Designers:**
    - Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Categories of End-users

- Actors on the scene (continued)
  - **End-users:** They use the data for queries, reports and some of them update the database content. End-users can be categorized into:
    - **Casual**: access database occasionally when needed
    - **Naïve** or Parametric: they make up a large section of the end-user population.
      - They use previously well-defined functions in the form of “canned transactions” against the database.
      - Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.

Categories of End-users (continued)

- **Sophisticated:**
  - These include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities.
  - Many use tools in the form of software packages that work closely with the stored database.
- **Stand-alone:**
  - Mostly maintain personal databases using ready-to-use packaged applications.
  - An example is a tax program user that creates its own internal database.
  - Another example is a user that maintains an address book

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.
- When a DBMS may be unnecessary:
  - If the database and applications are simple, well defined, and not expected to change.
  - If there are stringent real-time requirements that may not be met because of DBMS overhead.
  - If access to data by multiple users is not required.

When not to use a DBMS

- When no DBMS may suffice:
  - If the database system is not able to handle the complexity of data because of modeling limitations
  - If the database users need special operations not supported by the DBMS.

Summary

- Types of Databases and Database Applications
- Basic Definitions
- Typical DBMS Functionality
- Example of a Database (UNIVERSITY)
- Main Characteristics of the Database Approach
- Database Users
- Advantages of Using the Database Approach
- When Not to Use Databases