DataBase Management System

-Archana Vyas
Assistant Professor,
Information Technology,
FOT, DDU

Overview

Overview

- Chapter 11 - Storage and File Structure
- Chapter 12 - Indexing and Hashing
- Chapter 13 - Query Processing
- Chapter 14 - Query Optimization
- Chapter 15 - Transactions
- Chapter 16 - Concurrency Control
- Chapter 17 - Recovery System
- Chapter 18 - Database System Architectures X
- Chapter 19 - Distributed Databases

Recommended Text Books

- Data Base System Concepts by: Henry F. Korth and A. Silberschatz. 4th Ed., Publisher: McGraw-Hill 1991.

Reference Books

- Fundamentals of Database Systems, by: Shamkant Navathe, Publisher: Pearson

Chapter -1 Introduction

- Data - The raw facts are called as data. The word "raw" indicates that they have not been processed. For example 89 is the data.

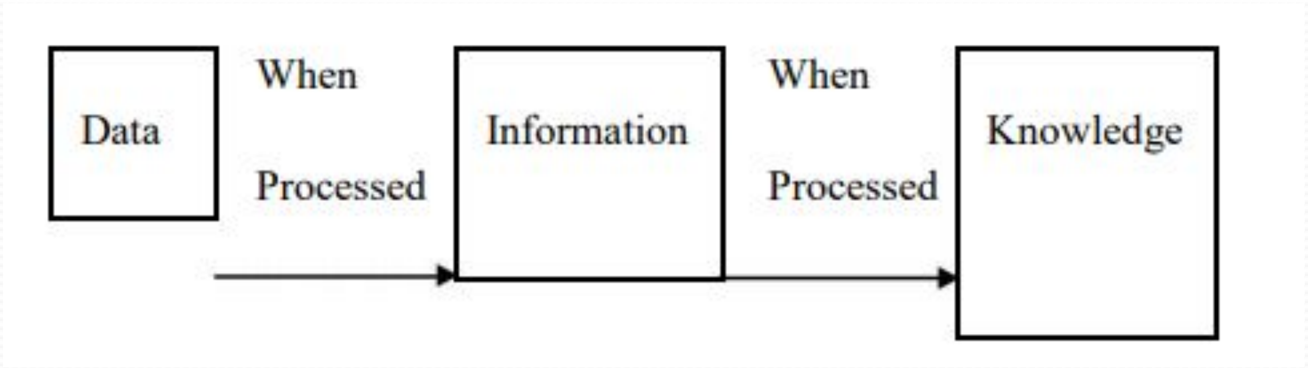
  Ex:- Text, Audio, Video, Image, Map, etc
- Information - The processed data is known as information. Ex:Marks:89; then it becomes information.
- Knowledge - Knowledge refers to the practical use of information.

  Knowledge necessarily involves a personal experience.

Chapter -1 Introduction

- DATA / INFORMATION PROCESSING -

  The process of converting the data (raw facts) into meaningful information is called as data/information processing.



- Note: In business processing knowledge is more useful to make decisions for any organization.

Chapter -1 Introduction

- Database –
  - Collection of similar / related data
  - Set of programs to access the data
  - DBMS contains information about a particular enterprise
  - DBMS provides an environment that is both *convenient* and *efficient* to use.
- Ex: Songs.pk

Chapter -1 Introduction

- Database Applications:
  - Banking: all transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions

FILE ORIENTED APPROACH

- The earliest business computer systems were used to process business records and produce information.
- They were generally faster and more accurate than equivalent manual systems.
- These systems stored groups of records in separate files, and so they were called **file processing systems.**

Disadvantages of File System

1. Data Redundancy
2. Data Inconsistency
3. Difficulty in accessing data
4. Data Isolation
5. Security problem
6. Atomicity problem
7. Concurrent access anomalies
8. Integrity problem

Data Redundancy and Inconsistency:

- Since files and application programs are created by different programmers over a long period of time, the files are likely to be having different formats and the programs may be written in several programming languages.
- Moreover, the same piece of information may be duplicated in several places.
- This redundancy leads to higher storage and access cost. In addition, it may lead to data inconsistency.

Difficulty in Accessing Data:

- The conventional file processing environments do not allow needed data to be retrieved in a convenient and efficient manner.
- Better data retrieval system must be developed for general use.
- Need to write a new program to carry out each new task

Data Isolation:

- Since data is scattered in various files, and files may be in different formats, it is difficult to write new application programs to retrieve the appropriate data.

Concurrent Access Anomalies:

- In order to improve the overall performance of the system and obtain a faster response time, many systems allow multiple users to update the data simultaneously.
- In such an environment, interaction of concurrent updates may result in inconsistent data.
- E.g. two people reading a balance and updating it at the same time.

Security Problems:

- Not every user of the database system should be able to access all the data. For example, in banking system, payroll personnel need only that part of the database that has information about various bank employees.
- They do not need access to information about customer accounts. It is difficult to enforce such security constraints.

Integrity Problems:

- The data values stored in the database must satisfy certain types of consistency constraints.
- For example, the balance of a bank account may never fall below a prescribed amount (< 1000) or no. of transactions <= 5 / month
- These constraints are enforced in the system by adding appropriate code in the various application programs.
- When new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items for different files.

Atomicity Problem:

- A computer system like any other mechanical or electrical device is subject to failure.
- In many applications, it is crucial to ensure that once a failure has occurred and has been detected, the data are restored to the consistent state existed prior to the failure

Lecture 2

History of Database Systems

History of Database Systems:

- 1950s and early 1960s:
  - Magnetic tapes were developed for data storage .
  - Data processing tasks such as payroll were automated, with data stored on tapes.
  - Data could also be input from punched card decks, and output to printers.
- Late 1960s and 1970s:
  - The use of hard disks in the late 1960s changed the scenario for data processing greatly, since hard disks allowed direct access to data.

History of Database Systems:

- With disks, network and hierarchical databases could be created that allowed data structures such as lists and trees to be stored on disk. Programmers could construct and manipulate these data structures.
- In the 1970’s the EF CODD defined the Relational Model.

History of Database Systems:

- In the 1980's:
  - Initial commercial relational database systems, such as IBM DB2, Oracle, Ingress, and DEC Rdb, played a major role in advancing techniques for efficient processing of declarative queries.
  - In the early 1980s, relational databases had become competitive with network and hierarchical database systems even in the area of performance.
  - The 1980s also saw much research on **parallel and distributed databases**, as well as initial work on object-oriented databases.

History of Database Systems:

- Early 1990s:
  - The SQL language was designed primarily in the 1990's.
  - And this is used for the transaction processing applications.
  - Decision support and querying re-emerged as a major application area for databases.
  - Database vendors also began to add object-relational support to their databases.

History of Database Systems:

- Late 1990s:
  - The major event was the explosive growth of the World Wide Web.
  - Databases were deployed much more extensively than ever before.
  - Database systems now had to support very high transaction processing rates, as well as very high reliability and 24 * 7 availability (availability 24 hours a day, 7 days a week, meaning no downtime for scheduled maintenance activities).
  - Database systems also had to support Web interfaces to data.

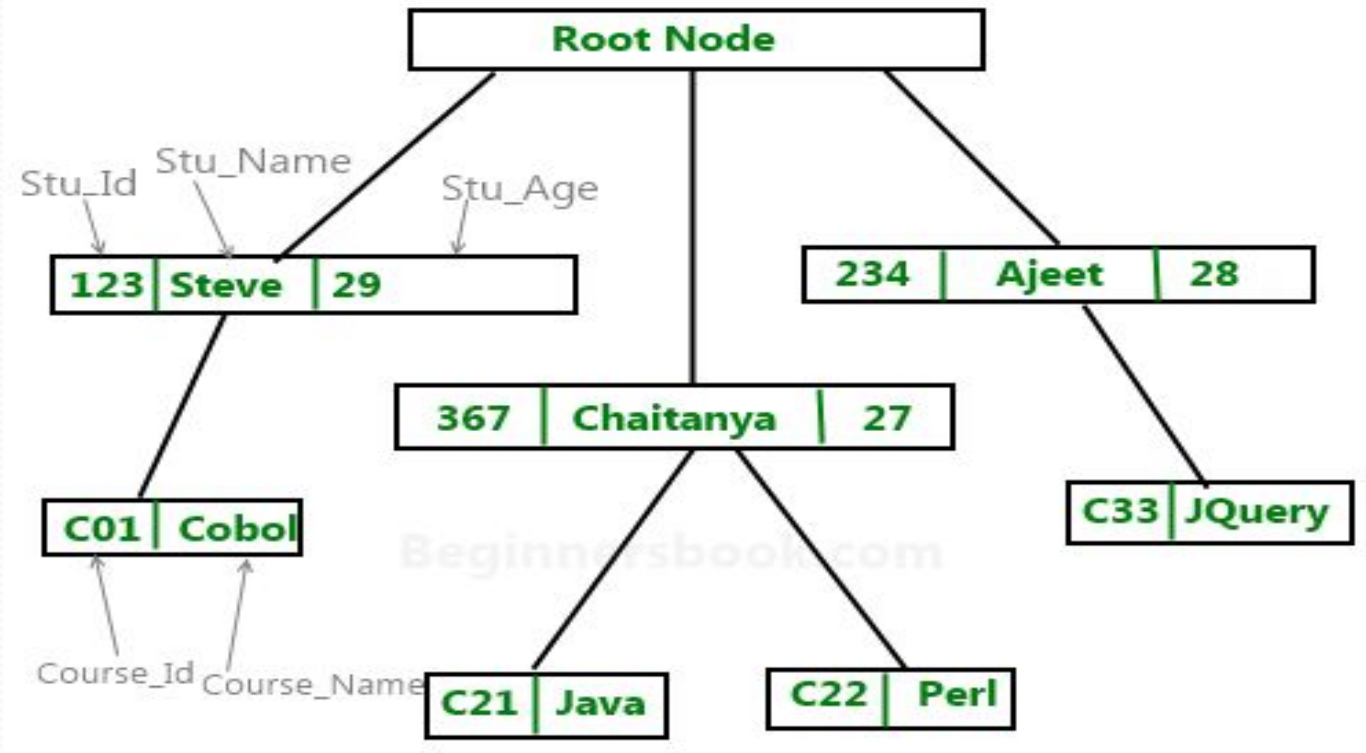
History of Database Systems:

- The Evolution of Database systems are as follows:
  1. File Management System
  2. Hierarchical database System
  3. Network Database System
  4. Relational Database System

File Management System:

- The file management system also called as FMS in short is one in which all data is stored on a single large file.
- The main disadvantage in this system is searching a record or data takes a long time. This lead to the introduction of the concept, of indexing in this system.
- Then also the FMS system had lot of drawbacks to name a few like updating or modifications to the data cannot be handled easily, sorting the records took long time and so on.
- All these drawbacks led to the introduction of the Hierarchical Database System.

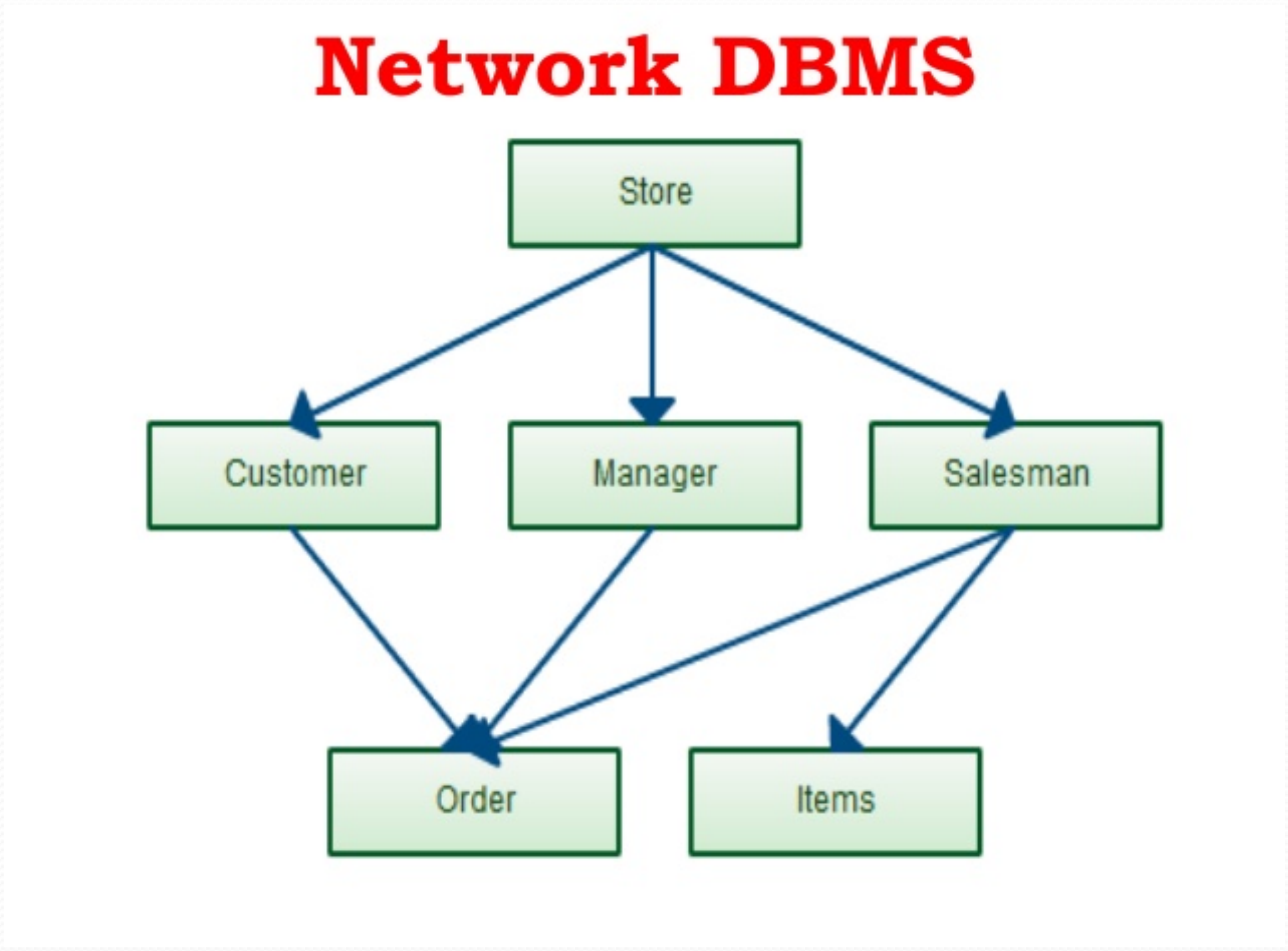
Hierarchical Database System:



Hierarchical Database System:

- The previous system FMS drawback of accessing records and sorting records which took a long time was removed in this by the introduction of parent-child relationship between records in database.
- The origin of the data is called the root from which several branches have data at different levels and the last level is called the leaf.
- The main drawback in this was if there is any modification or addition made to the structure then the whole structure needed alteration which made the task a tedious one.
- In order to avoid this next system took its origin which is called as the Network Database System.

Network Database System:



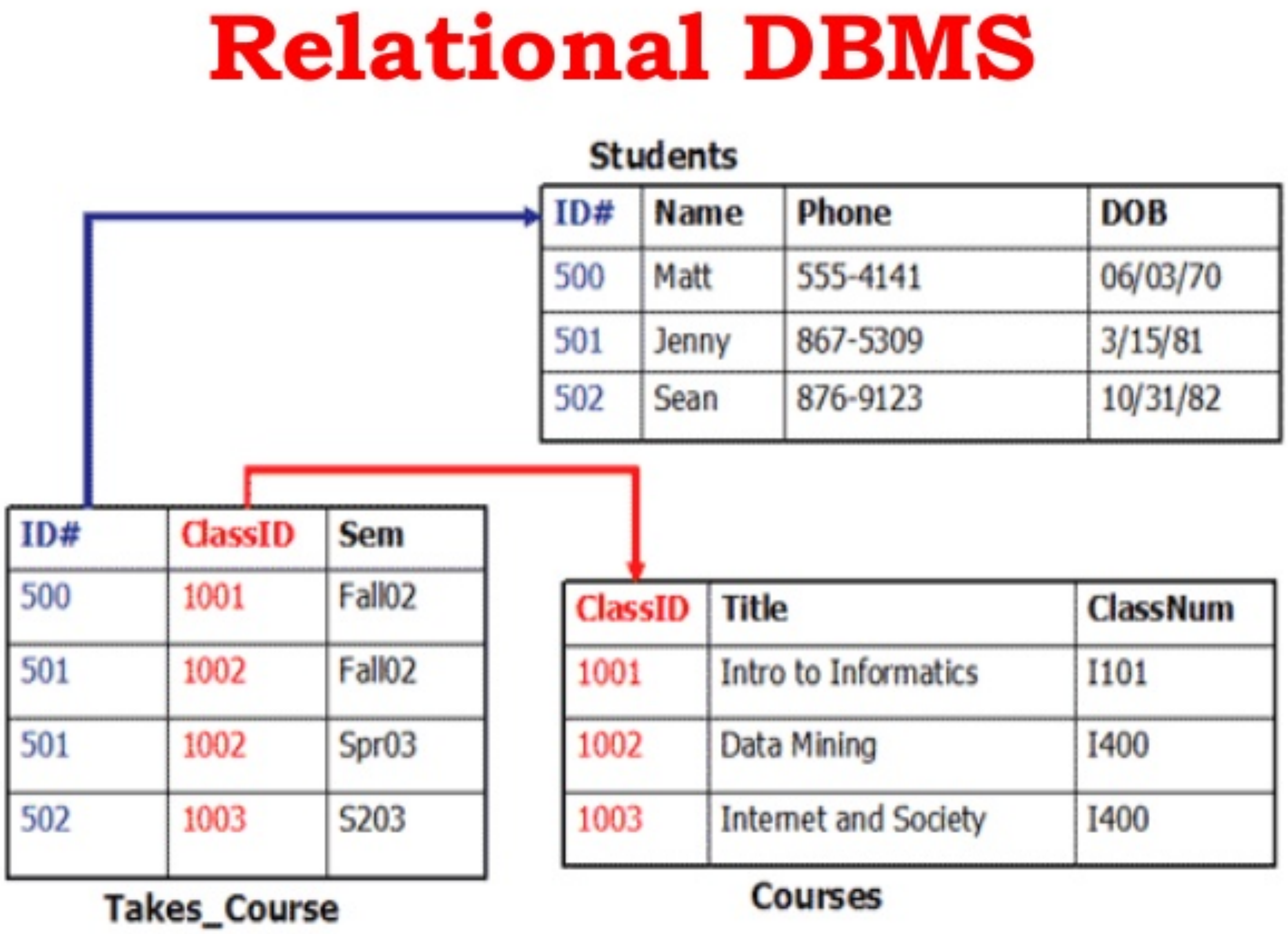
Network Database System:

- In this the main concept of many-many relationships got introduced.
- Network DBMSs are based on a network data model that allows each record to have multiple parents and multiple child records.
- A network database allows flexible relationship model between entities. The result is a faster data access, search, and navigation.
- The disadvantages of the network database model are the structure is **difficult to change**, this type of system is very **complex**, and there is a **lack of structural independence**

Relational Database System:

- In order to overcome all the drawbacks of the previous systems, the Relational Database System got introduced in which data get organized as **tables** and each record forms a row with many fields or attributes in it.
- Relationships between tables are also formed in this system.
- A relational database refers to a database that stores data in a **structured** format, using rows and columns.

Relational Database System:



Students

ID#	Name	Phone	DOB
500	Matt	555-4141	06/03/70
501	Jenny	867-5309	3/15/81
502	Sean	876-9123	10/31/82

ID#	ClassID	Sem
500	1001	Fall02
501	1002	Fall02
501	1002	Spr03
502	1003	S203

Takes_Course

ClassID	Title	ClassNum
1001	Intro to Informatics	I101
1002	Data Mining	I400
1003	Internet and Society	I400

Courses

CHARACTERISTICS OF DATABASE:

1. <u>Structured and Described Data:</u>
   - Fundamental feature of the database approach is that the database system does not only contain the data but also the complete definition and description of these data.
   - These descriptions are basically details about the extent, the structure, the type and the format of all data and, additionally, the relationship between the data.
   - This kind of stored data is called metadata ("data about data").
2. <u>Separation of Data and Applications:</u>
   - Application software does not need any knowledge about the physical data storage like encoding, format, storage place, etc.

CHARACTERISTICS OF DATABASE:

- It only communicates with the management system of a database (DBMS) via a standardized interface with the help of a standardized language like SQL.
- The access to the data and the metadata is entirely done by the DBMS. In this way all the applications can be totally separated from the data.

4. <u>Data Integrity:</u>

- Data integrity is a byword for the quality and the reliability of the data of a database system.
- In a broader sense data integrity includes also the protection of the database from unauthorized access (confidentiality) and unauthorized changes.
- Data reflect facts of the real world.

CHARACTERISTICS OF DATABASE:

4. Transactions:
   - A transaction is a bundle of actions which are done within a database to bring it from one consistent state to a new consistent state.
   - In between the data are inevitable inconsistent.
   - A transaction is atomic what means that it cannot be divided up any further.
   - Within a transaction all or none of the actions need to be carried out. Doing only a part of the actions would lead to an inconsistent database state.

CHARACTERISTICS OF DATABASE:

5. Data Persistence:

- Data persistence means that in a DBMS all data is maintained as long as it is not deleted explicitly.
- The life span of data needs to be determined directly or indirectly by the user and must not be dependent on system features.
- Additionally data once stored in a database must not be lost. Changes of a database which are done by a transaction are persistent.
- When a transaction is finished even a system crash cannot put the data in danger

Lecture 3

DataBase Management System

TYPES OF DATABASES:

- Database can be classified according to the following factors. They are:
  1. Number of Users
  2. Database Location
  3. Expected type
  4. Extent of use

TYPES OF DATABASES:

1. <u>Based on number of Users:</u>

- According to the number of users the databases can be classified into following types.
- They are :

  a). Single user b). Multiuser user

a) Single User Database
   - Single user database supports only one user at a time.
   - Desktop or personal computer database is an example for single user database.

b) Multiuser Database
   - Multi user database supports multiple users at the same time.
   - Workgroup database and enterprise databases are examples for multiuser database.

TYPES OF DATABASES:

- Workgroup database:
  - If the multiuser database supports relatively small number of users (fewer than 50) within an organization is called as Workgroup database.
- Enterprise database:
  - If the database is used by the entire organization and supports multiple users (more than 50) across many departments is called as Enterprise database.

2. <u>Based on Location:</u>
   - According to the location of database the databases can be classified into following types. They are:

     a). Centralized Database

     b). Distributed Database

TYPES OF DATABASES:

- Centralized Database:
  - It is a database that is located, stored, and maintained in a single location. This location is most often a central computer or database system, for example a desktop or server CPU, or a mainframe computer.
  - In most cases, a centralized database would be used by an organization (e.g. a business company) or an institution (e.g. a university.)
- Distributed Database:
  - A distributed database is a database in which storage devices are not all attached to a common CPU.
  - It may be stored in multiple computers located in the same physical location, or may be dispersed over a network of interconnected computers

Levels of Abstraction:

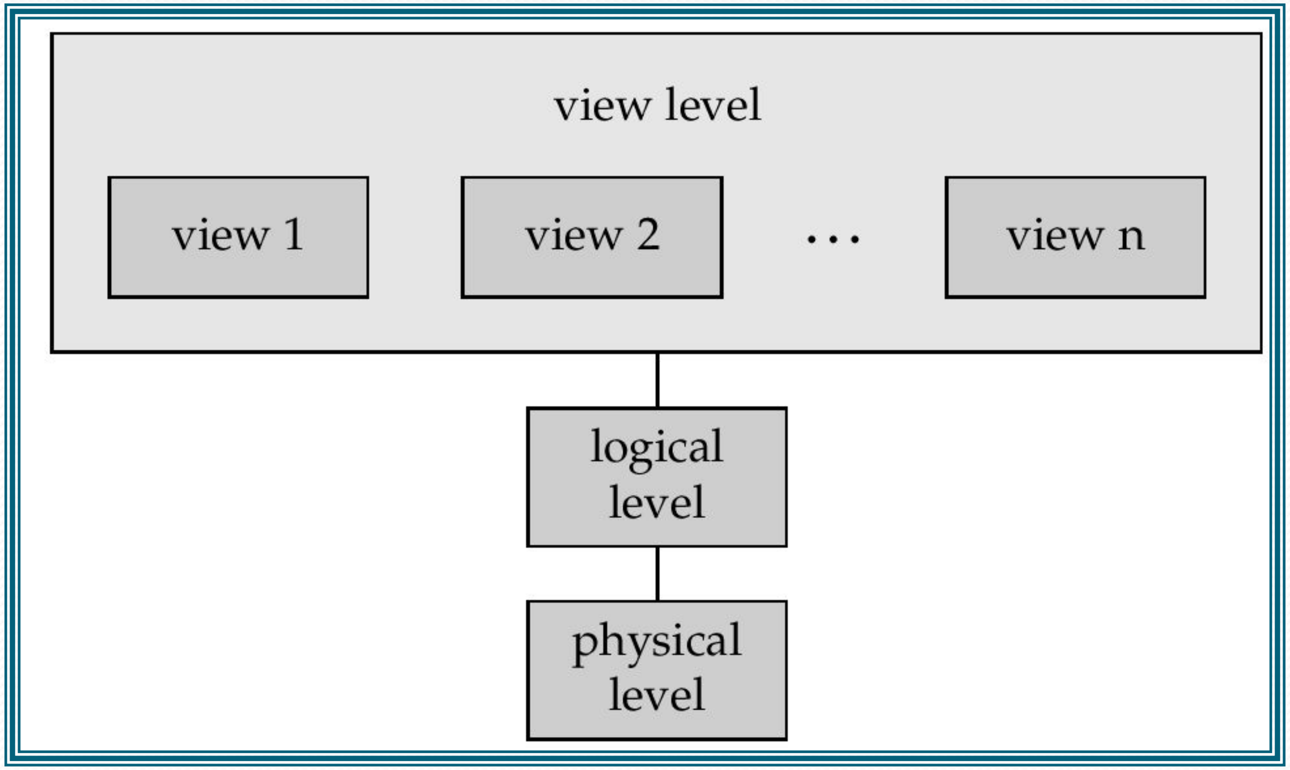
- Physical level: describes how a record (e.g., customer) is stored. It describes complex low-level data structures in detail.
- Logical level: describes what data are stored in database, and the relationships among the data. DBA decides what information to keep in the database, use logical level of abstraction

```
type customer = record
    name : string;
    street : string;
    pincode : integer;
end;
```

- View level: application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes

View of Data:

- An architecture for a database system



Instances and Schemas

- Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them)
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable

Instances and Schemas

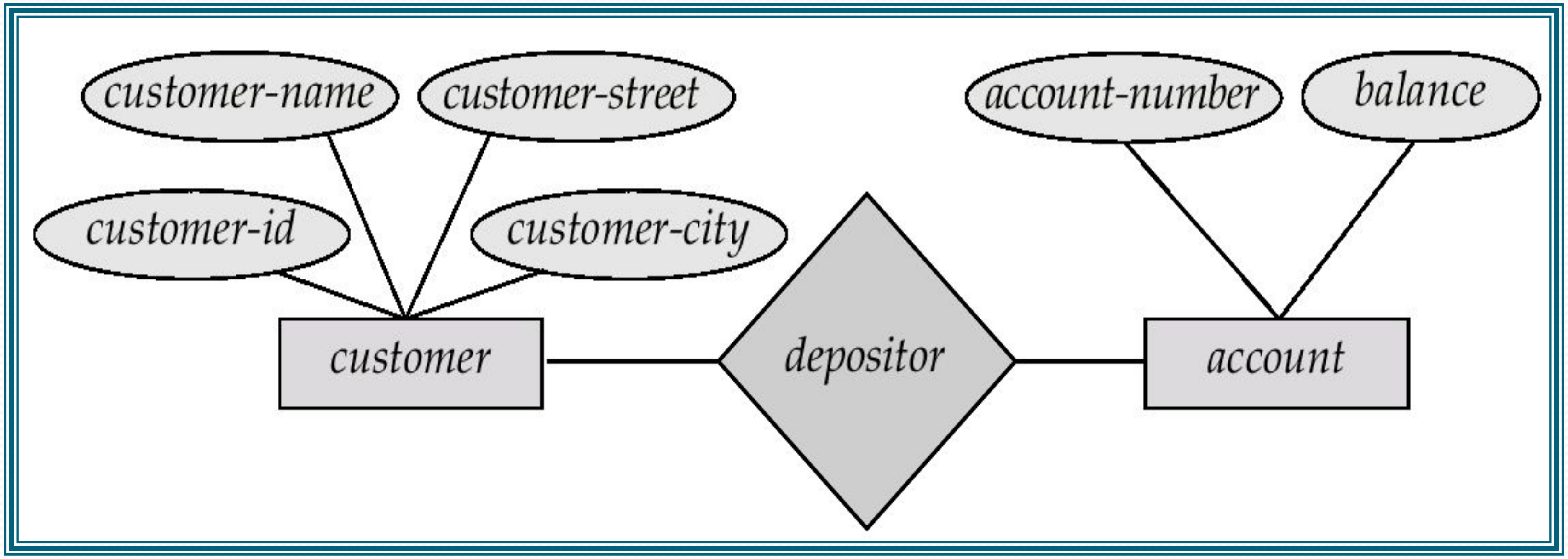
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Data Models

- A collection of tools for describing
  - data
  - data relationships
  - data semantics
  - data constraints
- Entity-Relationship model
- Relational model
- Other models:
  - object-oriented model
  - semi-structured data models
  - Older models: network model and hierarchical model

ER - Model

Example of schema in the entity-relationship model



Entity Relationship Model

- E-R model of real world
  - Entities (objects)
    - E.g. customers, accounts, bank branch
  - Relationships between entities
    - E.g. Account A-101 is held by customer Johnson
    - Relationship set *depositor* associates customers with accounts
- Widely used for database design
  - Database design in E-R model usually converted to design in the relational model which is used for storage and processing

Relational Model

- Example of tabular data in the relational model

Attributes

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
	Smith			

A Sample Relational Database

customer-id	*customer-name*	*customer-street*	*customer-city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
019-28-3746	Smith	4 North St.	Rye
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

account-number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer-id	*account-number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

Data Definition Language (DDL)

- Specification notation for defining the database schema
  - E.g.

```
create table account (
        account-number    char(10),
        balance           integer)
```

- DDL compiler generates a set of tables stored in a *data dictionary*
- Data dictionary contains metadata (i.e., data about data)
  - database schema
  - Data *storage and definition* language
    - language in which the storage structure and access methods used by the database system are specified
    - Usually an extension of the data definition language

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language
- Two classes of languages
  - Procedural – user specifies what data is required and how to get those data
  - Nonprocedural – user specifies what data is required without specifying how to get those data
- SQL is the most widely used query language

Database Users

- Users are differentiated by the way they expect to interact with the system
  - Application programmers – interact with system through DML calls
  - Sophisticated users – form requests in a database query language
  - Specialized users – write specialized database applications that do not fit into the traditional data processing framework
  - Naïve users – invoke one of the permanent application programs that have been written previously
    - E.g. people accessing database over the web, bank tellers, clerical staff

Database Administrator

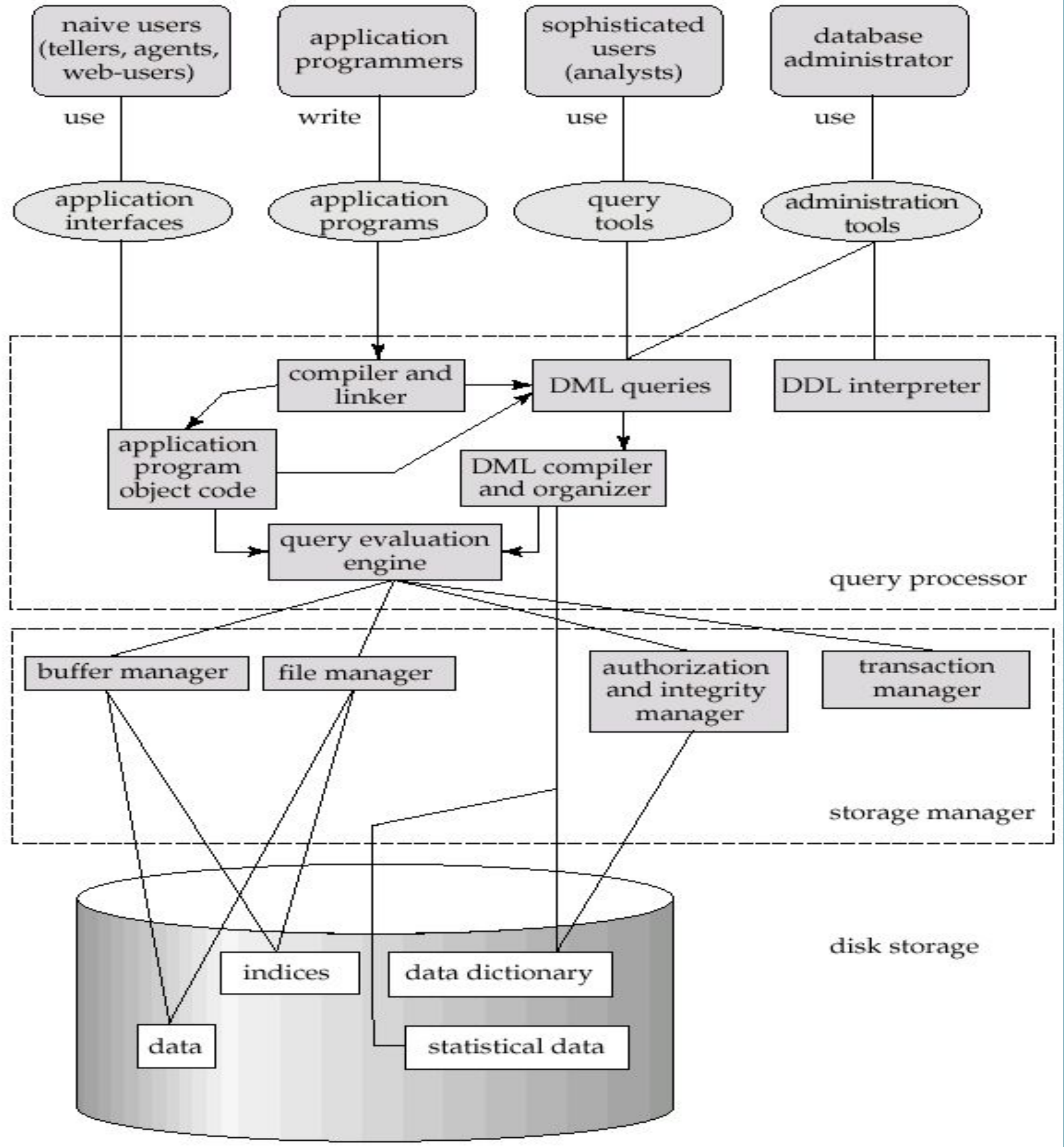
- Coordinates all the activities of the database system; the database administrator has a good understanding of the enterprise's information resources and needs.
- Database administrator's duties include:
  - Schema definition
  - Storage structure and access method definition
  - Schema and physical organization modification
  - Granting user authority to access the database
  - Specifying integrity constraints
  - Monitoring performance and responding to changes in requirements

Transaction Management

- A *transaction* is a collection of operations that performs a single logical function in a database application
- Transaction-management component ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- Concurrency-control manager controls the interaction among the concurrent transactions, to ensure the consistency of the database.

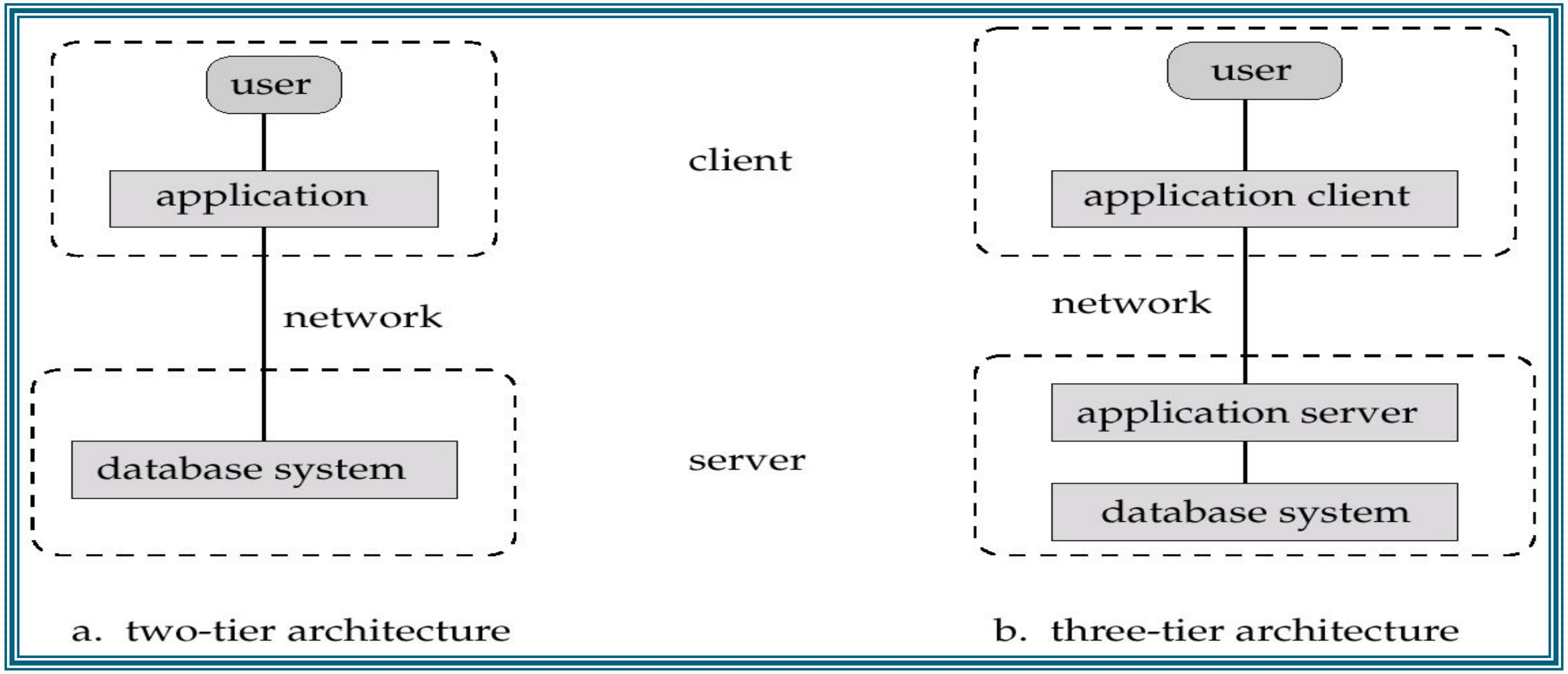
Storage Management

- Storage manager is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - interaction with the file manager
  - efficient storing, retrieving and updating of data



- Overall System Structure

Application Architectures



- **Two-tier architecture**: E.g. client programs using ODBC/JDBC to communicate with a database
- **Three-tier architecture**: E.g. web-based applications, and applications built using “middleware”