Chapter 1: Introduction to Database Management System

Adama Science and Technology University
School of Electrical Engineering and Computing
Department of CSE
CSEg 2208: Database Systems
Dr. Mesfin Abebe Haile (2022)

Outline

- What is database ?
- Database management system and its components
- Database design life cycle
- Roles in Database Design and Use
- database Architecture

What is Database ?

- A very *large integrated* collection of data.
- They are used to *maintain internal records*, to present data to *customers* and *clients* on the World-Wide-Web, and to support *many other commercial processes*.
- *Example*: Model Real world enterprise.

- Database management system (DBMS) – a powerful tool (software package) for *creating* and *managing large amounts of data efficiently* and allowing it to persist over long periods of time, safely.

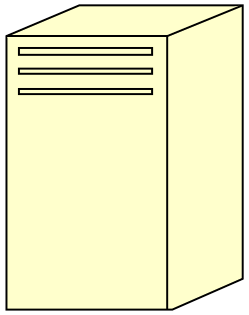
What is Database ?

- Data management involves *both definition* and the *manipulation* of the data.
- So the term database refers to a *collection of data* that is *managed by a DBMS*.
- Thus the Database Systems course is about:
  - How to *organize data*;
  - Supporting *multiple users*;
  - Efficient and effective *data retrieval*;
  - Secured and *reliable storage* of data;
  - Maintaining *consistent data*.

What is Database ?

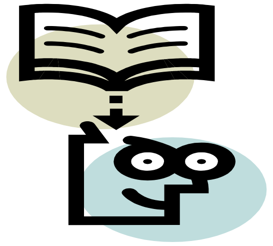
- What is a Database?- It is a collection of *related facts*.

Filing Cabinet



Hard disk full of data

Diary



Library

DBMS + Database = Database System

What is Database ?

- Data management passes through *different levels of development*.

- The common data management levels of development are three:
  - Manual Approach,
  - Traditional File Based Approach,
  - Database Approach.

Manual Approach

- Data storage and retrieval follows the *primitive* and *traditional* way of information handling *where cards* and *paper* are used for the purpose. Example:
  - Files for as many event and objects as the organization has are used to store information.
  - Each of the *files* containing *various kinds of information* is labelled and stored in *one ore more cabinets*.
  - The cabinets could be *kept in safe places* for security purpose based on the sensitivity of the information contained in it.
  - *Insertion* and *retrieval* is done by *searching first* for the right *cabinet* then for the right the *file* then the *information*.

Manual Approach

- Limitations of the Manual approach:
  - Prone to *error*,
  - Difficult to *update*, *retrieve*, *integrate*,
  - You have the data but it is difficult to *compile the information*,
  - Cross referencing is *difficult*.

Traditional File Based Approach

- Uses *computer* for data processing to the business community, via the *device for data storage* and *processing increase*.
  - File based systems were an *early attempt to computerize* the manual filing system.
  - This approach is the *decentralized computerized* data handling method.
  - Since *every application defines* and *manages* its own data, the system is subjected to serious *data duplication problem*.

Traditional File Based Approach

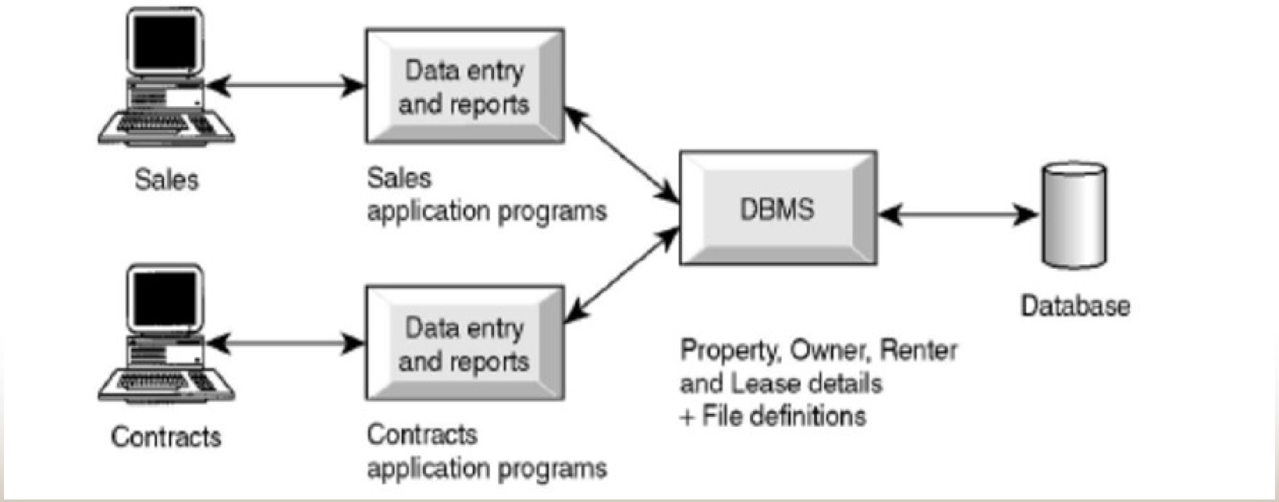
- Limitations of the Traditional File Based approach:
  - Limited data sharing - *every application maintains* its own data.
  - *Duplication* or *redundancy* of data (money and time cost and loss of data integrity)
  - The problem in "*update anomalies*"; three types of anomalies:
    - *Modification Anomalies*: a problem experienced when one ore more data value is modified on one application program but not on others containing the same data set.
    - *Deletion Anomalies*: a problem encountered where one record set is deleted from one application but remain untouched in other application programs.
    - *Insertion Anomalies*: a problem experienced when ever there is new data item to be recorded, and the recording is not made in all the applications.

Database Approach

- ❖ The database systems is presented for *user with a view of data organized as tables* called *relations*.
- ❖ It emphasizes the *integration* and *sharing of data* throughout the organization.
- ❖ Database is a *collection of logically related data* where these logically related data comprises *entities*, *attributes*, *relationships*, and *business rules* of an organization's information.
- ❖ Database is deigned once and used simultaneously by many users.

Database Approach

- Unlike the traditional file based approach ,in database there is the separation of the *data definition* from the *application*.
- Thus the *application is not affected* by changes made in the *data structure and file organization*.


Data entry and reports
Sales
Sales application programs
DBMS
Database
Data entry and reports
Contracts
Contracts application programs
Property, Owner, Renter and Lease details + File definitions

Benefits of the database approach

- *Data can be shared*: two or more users can access and use same data instead of storing data in redundant manner for each user.
- *Improved accessibility of data*: by using structured query languages, the users can easily access data without programming experience.
- *Quality data can be maintained*: the different integrity constraints in the database approach will maintain the quality leading to better decision making
- *Integrity can be maintained*: data at different applications will be integrated together with additional constraints to facilitate validity and consistency of shared data resource.

Benefits of the database approach

- *Security measures can be enforced*: the shared data can be secured by having different levels of clearance.
- *Speed*: data storage and retrieval is fast.

- *Centralized information control*: since relevant data in the organization will be stored at one repository, it can be controlled and managed at the central level.

Limitation of the database approach

- Limitations and risk of Database Approach:
  - *Complexity* in designing and managing data,
  - The *cost* and *risk* during conversion from the old to the new system,
  - High cost to be incurred to *develop* and *maintain* the system,
  - Complex *backup* and *recovery* services from the users perspective,
  - *Reduced performance* due to centralization and data independency,
  - *High impact* on the system when failure occurs to the central system.

Database Management System (DBMS)

- A *full scale DBMS* should at least have the *following services* to provide to the user:
  - Data *storage*, *retrieval* and *update* in the database.
  - A *user accessible* catalogue.
  - *Transaction support service*: ALL or NONE transaction, which minimize data inconsistency.
  - *Concurrency Control Services*: access and update on the database by different users simultaneously should be implemented correctly.
  - *Recovery Services*: a mechanism for recovering the database after a failure must be available.

Database Management System (DBMS)

- A full scale DBMS should at least have the following services to provide to the user:
  - *Authorization Services* (Security): must support the implementation of access and authorization service to database administrator and users.
  - *Integrity Services*: rules about data and the change that took place on the data, correctness and consistency of stored data, and quality of data based on business constraints.
  - Services to *promote data independency* between the data and the application.

DBMS - Database Languages

- DBMS should have facilities to *define the database*, *manipulate* the content of the database and *control* the database.
- It provides the *following facilities*:
  - Data Definition Language (DDL):
    - Language used *to define each data element* required by the organization.
    - Commands *for setting up schema* or *database*.
    - These commands are used to setup a *database*, *create*, *delete* and *alter* table with the facility of handling constraints.

DBMS - Database Languages

- It provides the *following facilities*:
  - Data Manipulation Language (DML):
    - Is a core command used by *end-users* and *programmers* to *store*, *retrieve*, and *access* the data in the database e.g. SQL
    - Since the required data or Query by the user will be extracted using this type of language, it is also called "*Query Language*“.

DBMS

- Taking a DBMS as a system, to design and use a database, there will be the interaction or integration of *Hardware*, *Software*, *Data*, *Procedure* and *People*.
  - *Hardware*: These components are comprised of various types of personal computers, mainframe or any server computers to be used in multi-user system, network infrastructure, and other peripherals required in the system.
  - *Software*: are collection of commands and programs used to manipulate the hardware to perform a function. Like the DBMS software, application programs, operating systems, network software, language software and other relevant software.

DBMS

- Taking a DBMS as a system, to design and use a database, there will be the interaction or integration of *Hardware*, *Software*, *Data*, *Procedure* and *People*.
  - *Procedure*: this is the rules and regulations on how to design and use a database. It includes procedures like how to log on to the DBMS, how to use facilities, how to start and stop transaction, how to make backup, how to treat hardware and software failure.
  - *Data*: Data is the *most important component* to the user of the database. There are two categories of data in any database system: that is *Operational* and *Metadata*.

DBMS

- *Operational data* is the data actually stored in the system to be used by the user.
- *Metadata* is the data that is used to store information about the database itself.

- The structure of the data in the database is called the *schema*, which is composed of the *Entities*, *Properties of entities*, and *relationship between entities* and *business constraints*.

- *People*: this component is composed of the people in the organization that are responsible or play a role in *designing*, *implementing*, *managing*, *administering* and *using* the resources in the database. This component includes group of people who are experts or user of DB.

Database Development Life Cycle (DDLC)

- Major steps in database design are:
  - *Planning*: that is identifying information gap in an organization and propose a database solution to solve the problem.
  - *Analysis*: that concentrates more on fact finding about the problem or the opportunity. Feasibility analysis, requirement determination and structuring, and selection of best design method are also performed at this phase.
  - *Design*: The phase is further divided into three sub-phases.
    - a. Conceptual Design: concise description of the data, data type, relationship between data and constraints on the data.

Database Development Life Cycle (DDLC)

- Major steps in database design are:
  - *Design*: The phase is further divided into three sub-phases.
    - b. *Logical Design*: a higher level conceptual abstraction with selected specific data model to implement the data structure.
    - – It is particular DBMS **independent** and with no other physical considerations.
    - c. *Physical Design*: physical implementation of the logical design of the database with respect to internal storage and file structure of the database for the selected DBMS.
    - – To develop all technology and organizational specification.

Database Development Life Cycle (DDLC)

- Major steps in database design are:
  - *Implementation*: the testing and deployment of the designed database for use.
  - *Operation and Support*: administering and maintaining the operation of the database system and providing support to users.

Roles in Database Design and Use

- There are *group of roles* played by different stakeholders of the *designing* and *operation* of a database system. These are:
- Database Administrator (DBA):
  - Responsible to *oversee*, *control* and *manage* the database resources
  - *Authorizing access* to the database,
  - *Coordinating* and *monitoring* the use of the database,
  - Responsible for *determining* and *acquiring* hardware and software resources,
  - *Accountable* for problems like poor security, poor performance of the system,
  - Involves in *all steps* of database development.

Roles in Database Design and Use

- There are *group of roles* played by different stakeholders of the *designing* and *operation* of a database system. These are:
- Database Designer (DBD)
  - Identifies the *data to be stored* and *choose the appropriate structures* to represent and store the data.
  - Should understand the *user requirement* and should *choose how the user views* the database.
  - Involve on the *design phase* before the implementation of the database system.

Roles in Database Design and Use

- There are *group of roles* played by different stakeholders of the *designing* and *operation* of a database system. These are:
- Application Programmer and Systems Analyst
  - System analyst *determines the user requirement* and *how the user wants to view* the database.
  - The application programmer implements these specifications as programs; *code*, *test*, *debug*, *document* and *maintain* the application program.
  - The application programmer *determines the interface* on how to *retrieve*, *insert*, *update* and *delete data* in the database.
  - The application could use any high *level programming language* according to the availability, the facility and the required service..

Roles in Database Design and Use

- There are *group of roles* played by different stakeholders of the *designing* and *operation* of a database system. These are:
- End Users
  - Workers, whose job requires *accessing the database frequently* for various purposes, there are different group of users in this category. Like *Naïve Users* and *Sophisticated Users*.

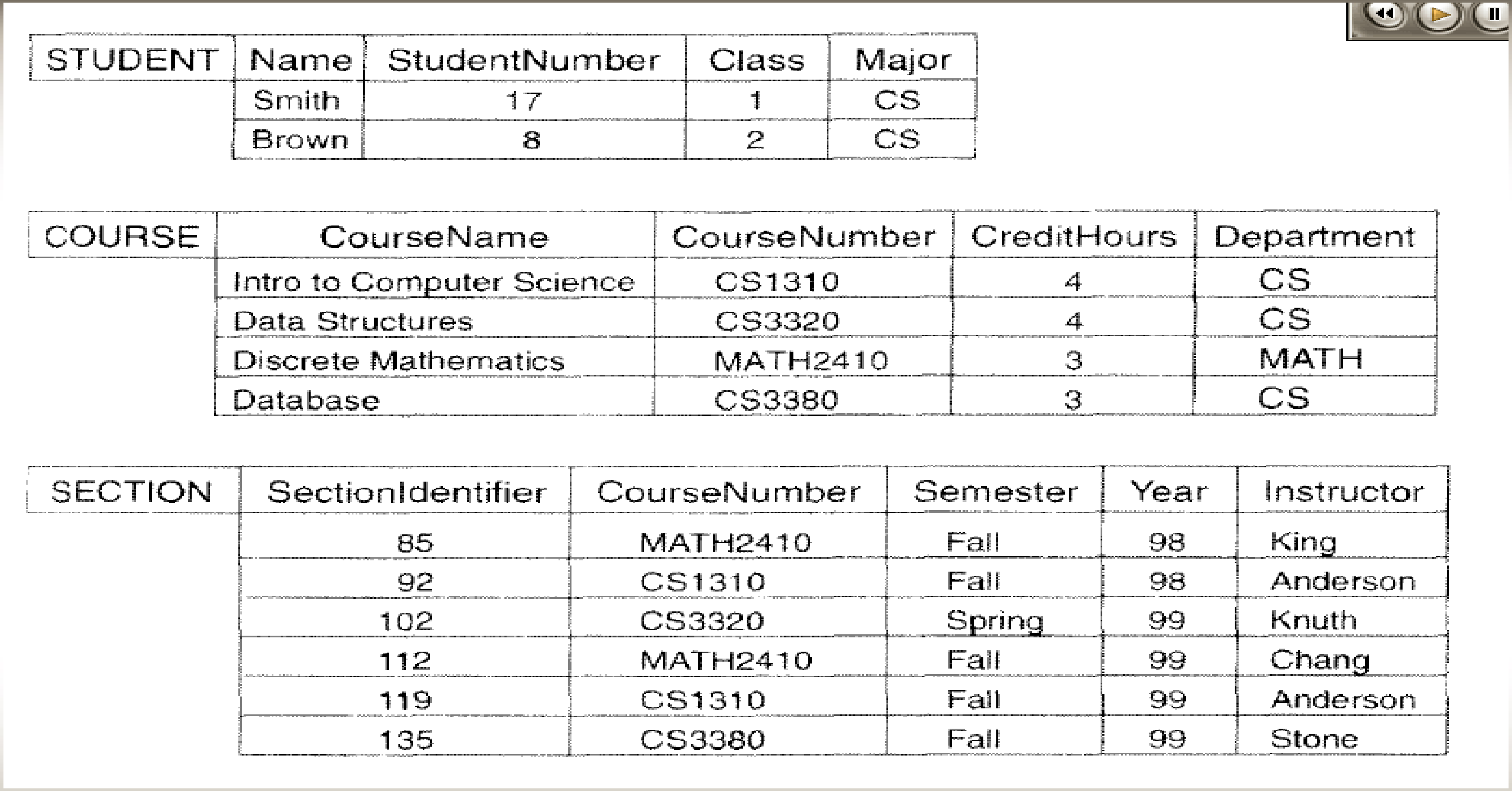
Example of a Database

- Part of a UNIVERSITY environment; Information concerning *students*, *courses*, and *grades* in a university environment.

- Some *mini-world entities*:
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - DEPARTMENTs
  - INSTRUCTORs

- 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

Example of a Database

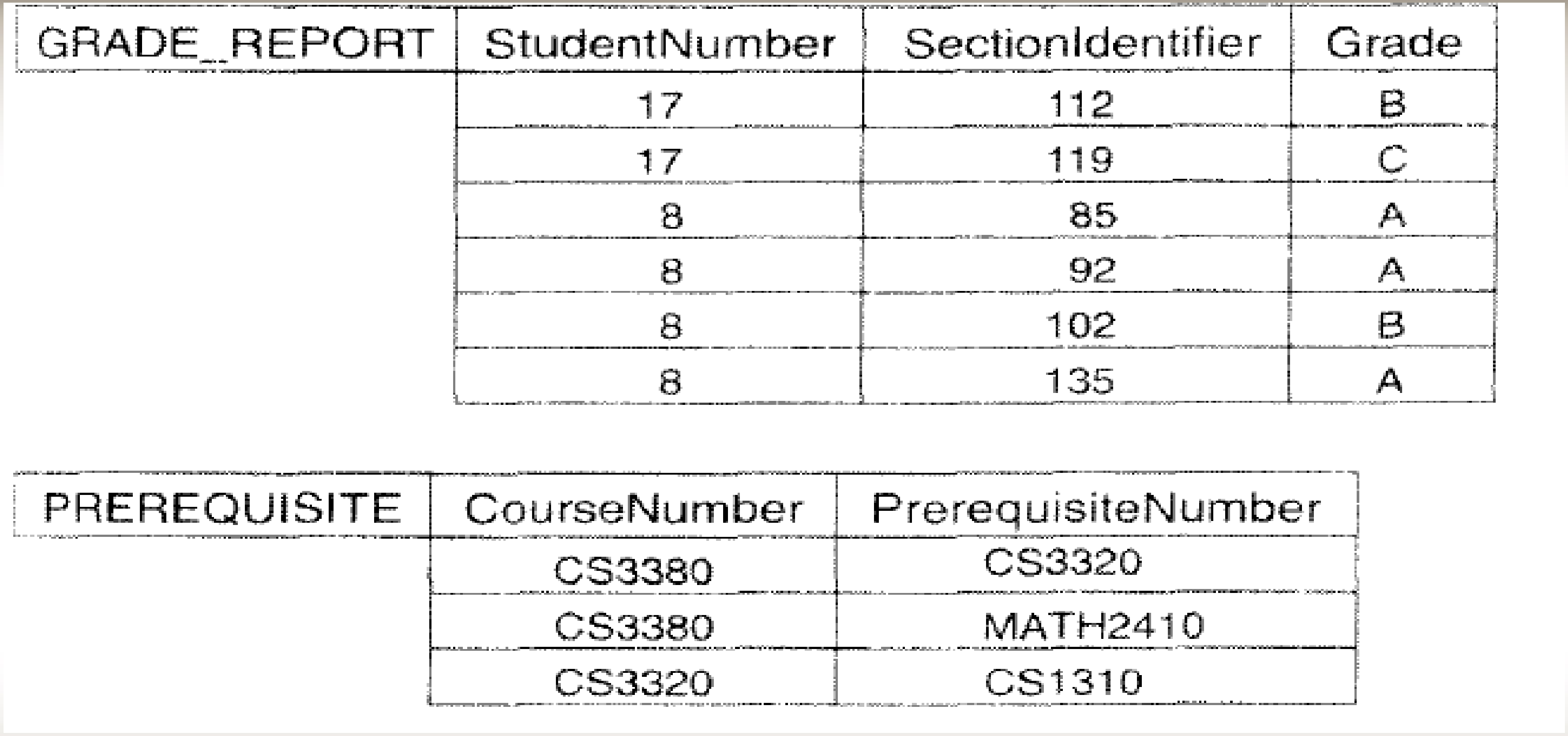


STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	CS1310	Fall	98	Anderson
	102	CS3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	CS1310	Fall	99	Anderson
	135	CS3380	Fall	99	Stone

Example of a Database



GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	CS3380	CS3320
	CS3380	MATH2410
	CS3320	CS1310

Example of a Database

- Construct *UNIVERSITY* database:
  - Store data to represent each *student*, *course*, *section*, *grade report*, and *prerequisite* as a record in appropriate file.
- *Relationships* among the records.
- Manipulation involves *querying* and *updating*.
- Examples of *queries*:
  - Retrieve the *transcript*,
  - List the *names of students* who took the section of the 'Database' course offered in fall 2013 and their grades in that section.
  - List the *prerequisites* of the 'Fundaments of Database' course.

Example of a Database

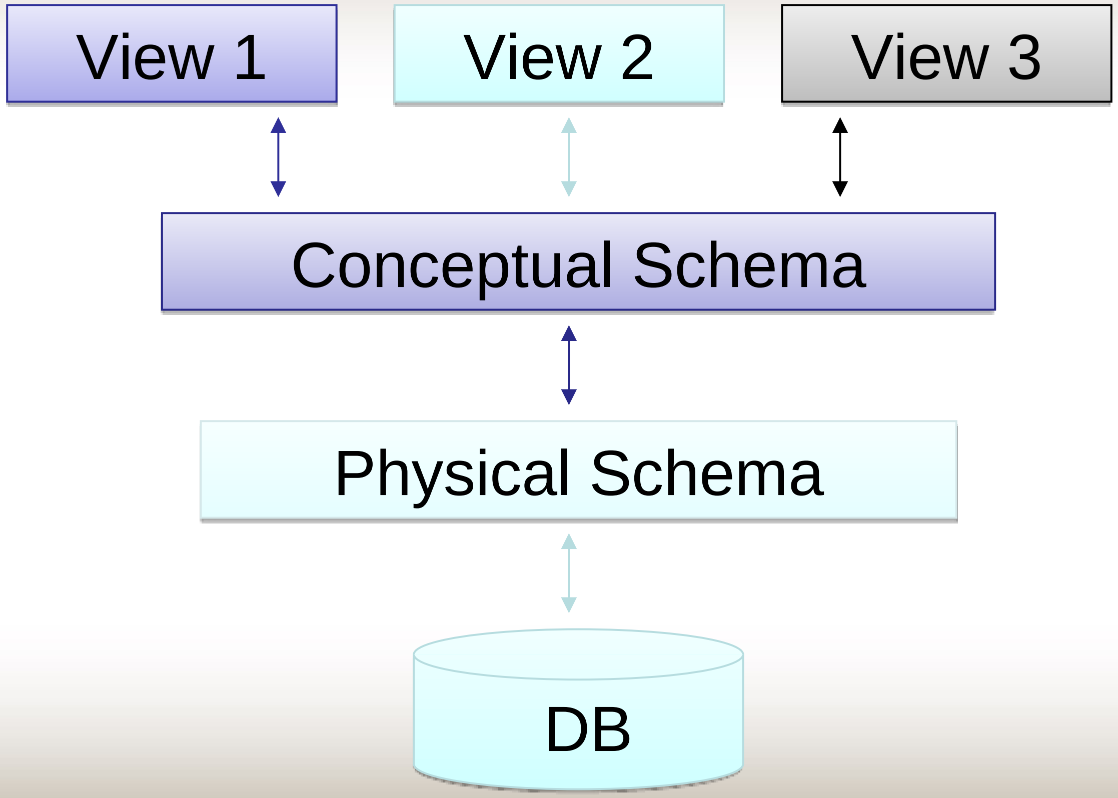
- Manipulation involves *querying* and *updating*.
- Examples of *updates*:
  - *Change* the class of ‘Lemma’ to sophomore.
  - Create a *new section* for the ‘Database’ course for this semester.
  - *Enter* a grade of ‘A’ for ‘Lemma’ in the ‘Database’ section of last semester.

Data Abstraction (Categories of data models)

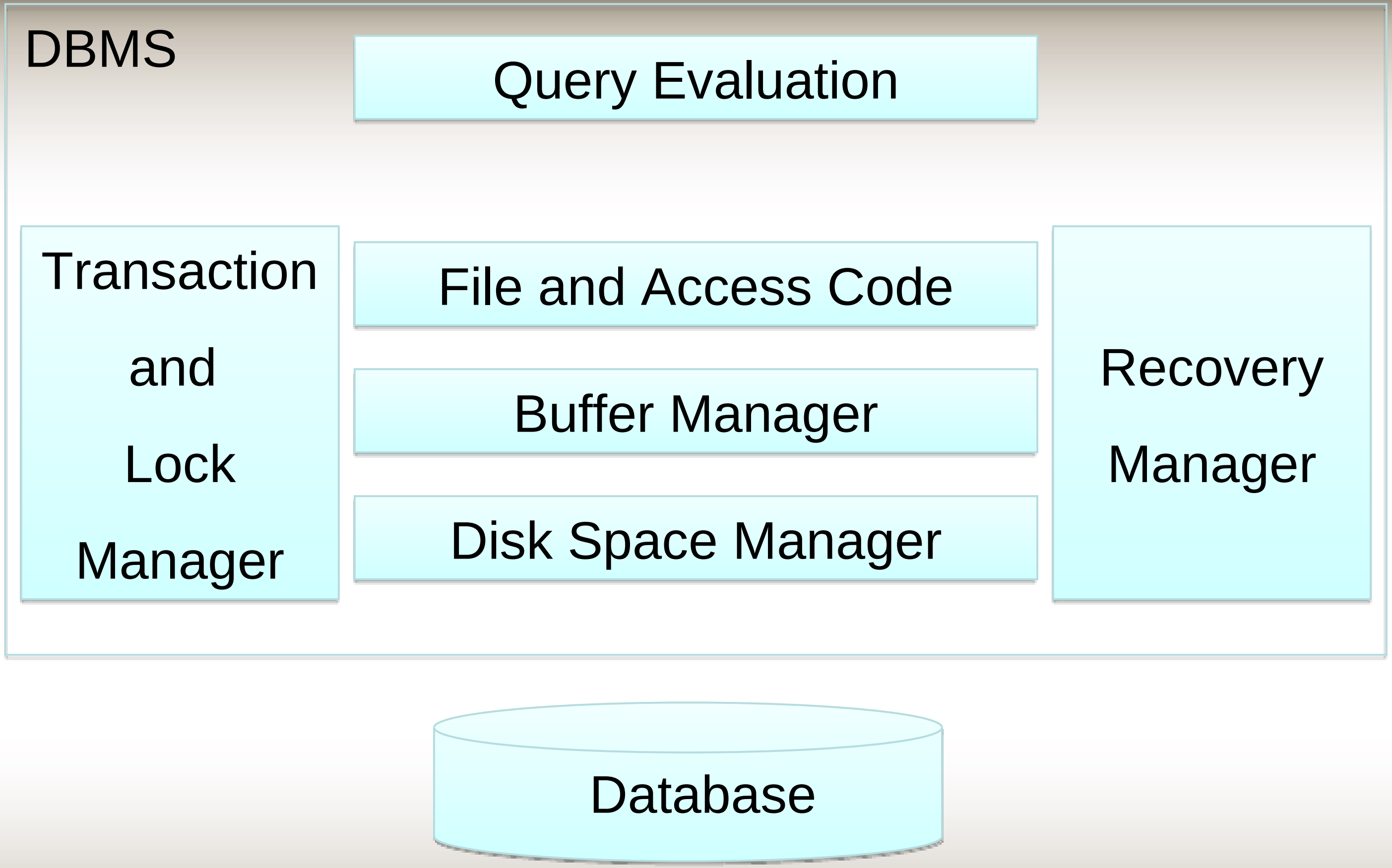
- Data can be described at *three levels of abstraction*.
- Physical schema:
  - The *lowest level* schema.
  - Describes *how data are stored* and *indexed*.
- Conceptual (or logical) schema:
  - What (not how) *data are stored*.
  - Describes *data* in terms of the *data model*.
- External (or view) schema:
  - The *highest level* schema.
  - Describes *how some users access* the data.
  - There can be *many different views*.

Levels of Abstraction

❖ Data can be described at *three levels of abstraction*.



Typical DBMS Structure/Database System



Database Components (Structure)

- *Disk space (storage) manager* – responsible for interaction with the *OS file system*.
  - Allows other levels of the DBMS to consider the data as a collection of pages.
- *Buffer manager* – responsible for bringing pages into *main memory* from disk.
  - Including the management of a replacement policy when main memory is full.
- *File and access code* - allows the query evaluation system to *request data from lower levels*.

Database Components (Structure)

- *Query evaluation* – most modern DBMSs will optimize queries.
  - There are often multiple equivalent queries.
  - The query optimizer determines an efficient execution plan for a query.

- *Transaction lock manager* – responsible for allowing *concurrent access*.
  - While maintaining *data integrity*.
- *Recovery manager* – responsible for *maintaining a log* and *restoring the system* after a crash.

Database System Architecture

- i. Centralized and Client-Server Architectures:
  - *Centralized DBMS*: combines everything into single system including- DBMS software, hardware, application programs and user interface processing software.

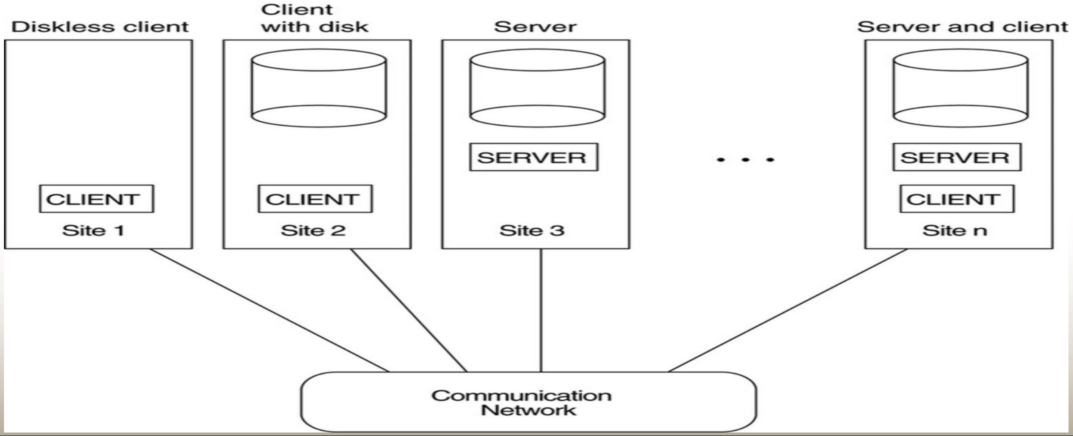
  - It includes the following *three things*:
    - A, Specialized Servers with Specialized functions:
      - File Servers
      - Printer Servers
      - Web Servers
      - E-mail Servers

Database System Architecture

- i. Centralized and Client-Server Architectures:
  - It includes the following *three things*:
    - B. Clients:
      - Provide *appropriate interfaces* and a client-version of the system to *access* and *utilize the server resources*.
      - Clients maybe *diskless machines* or *PCs* or *Workstations* with disks with only the client software installed.
      - *Connected* to the *servers* via some form of a network. (LAN: local area network, wireless network, etc.)
    - C. DBMS Server:
      - Provides *database query* and *transaction services* to the clients.
      - Sometimes called *query* and *transaction servers*.

Database System Architecture

- ii. Two Tier Client-Server Architecture:
  - *User Interface Programs* and *Application Programs* run on the client side.
  - Interface called ODBC (Open Database Connectivity) provides an *Application program interface* (API) allow client side programs to call the DBMS.
  - A *client program* may connect to several DBMSs.



Database System Architecture

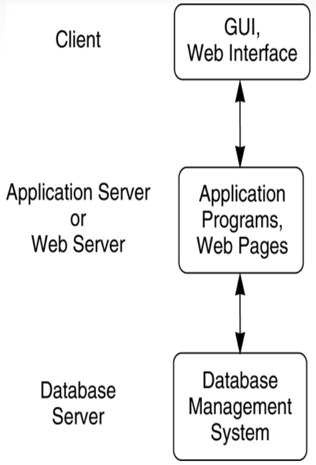
- ii. Two Tier Client-Server Architecture:
  - *User Interface Programs* and *Application Programs* run on the client side.
  - Interface called ODBC (Open Database Connectivity) provides an *Application program interface* (API) allow client side programs to call the DBMS.
  - A *client program* may connect to several DBMSs.

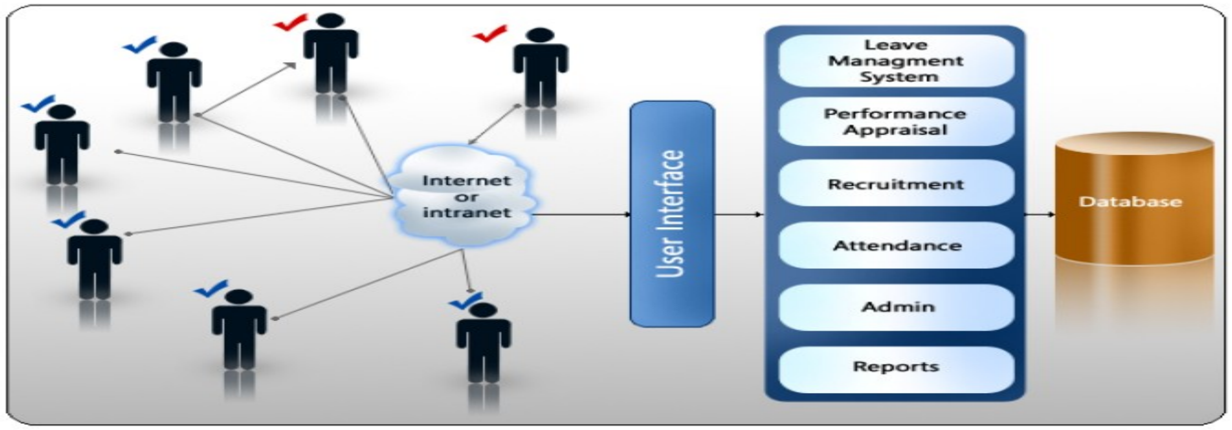
Database System Architecture

- iii. Three Tier Client-Server Architecture:
  - Common for *Web applications*.
  - Intermediate Layer called *Application Server* or *Web Server*:
    - Stores the web connectivity software and the rules and business logic (constraints) part of the application used to access the right amount of data from the database server
    - Acts like a conduit for sending partially processed data between the database server and the client.
  - Additional Features- *Security*:
    - Encrypt the data at the server before transmission.
    - Decrypt data at the client.

Database System Architecture

❖ iii. Three Tier Client-Server Architecture:


Client
GUI, Web Interface
Application Server or Web Server
Application Programs, Web Pages
Database Server
Database Management System



Question & Answer

Thank You !!!