DATABASE MANAGEMENT SYSTEMS

Part 1

SYLLABUS

- **Introduction & Entity Relationship (ER) Model**
- Concept & Overview of Database Management Systems (DBMS). Characteristics of Database system, Database Users, structured, semi-structured and unstructured data.
- Data Models and Schema - Three Schema architecture.
- Database Languages, Database architectures and classification.
- ER model - Basic concepts, entity set & attributes, notations, Relationships and constraints, cardinality, participation, notations, weak entities, relationships of degree 3.

DATA, DATABASE & DBMS

- **Data**
  - Known facts that can be recorded and have implicit meaning
- **Database**
  - The collection of data
- **Database-management system (DBMS)**
  - It is a collection of programs that enable users to create and maintain a database.
- The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.
- DBMS is a general purpose software system that facilitates process of **defining**, **constructing**, **manipulating**, and **sharing** database.

DATA, DATABASE & DBMS

- Database systems are designed to manage large bodies of information.
- Management of data involves both storage of information and mechanisms for manipulation of information.
- The database system must ensure the safety of the information stored
- If data are to be shared among several users, the system must avoid possible anomalous results.

STRUCTURED, SEMI-STRUCTURED AND UNSTRUCTURED DATA

- **Structured data**
- Represented in a strict format
- It has been organized into a formatted repository that is typically a database.
- It concerns all data which can be stored in database SQL in a table with rows and columns
- Example: Relational data

STRUCTURED, SEMI-STRUCTURED AND UNSTRUCTURED DATA

- **Semi-Structured data**
- Information that does not reside in a relational database but that have some organizational properties that make it easier to analyze.
- With some process, you can store them in the relational database.
- Example: XML data

STRUCTURED, SEMI-STRUCTURED AND UNSTRUCTURED DATA

- **Unstructured data**
- Data which is not organized in a predefined manner or does not have a predefined data model.

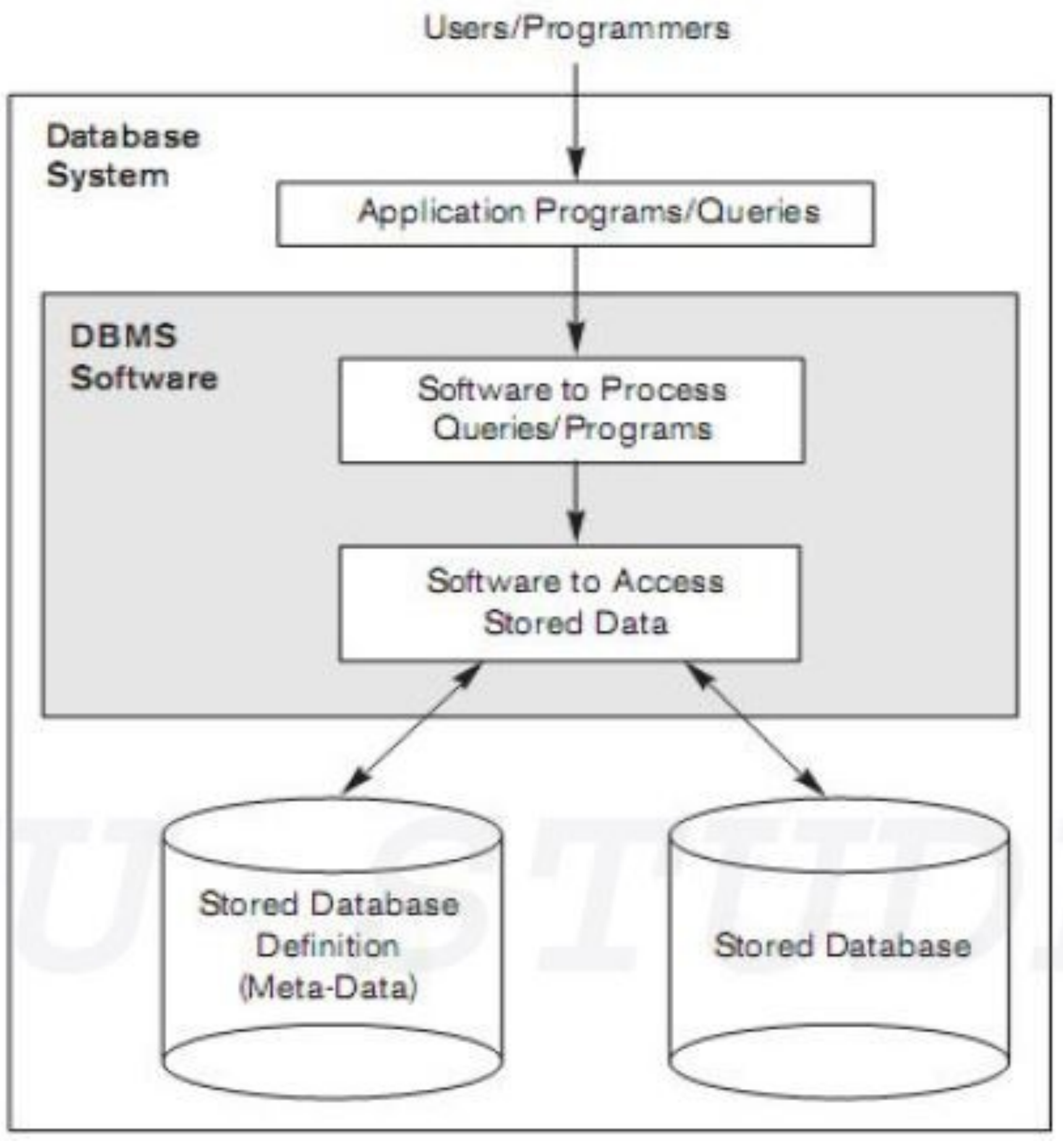
- It is not a good fit for a mainstream relational database.
- There are alternative platforms for storing and managing, it is increasingly prevalent in IT systems and is used by organizations in a variety of business intelligence and analytics applications.

- Example: Word, PDF, Text, Media logs.

DATABASE IMPLICIT PROPERTIES

- Universe of discourse(UoD) or Miniworld
  - Database represent some aspects of real world
  - Changes to the miniworld are reflected in the database
- A database is a logically coherent collection of data with some inherent meaning.
- A database is designed, built and populated with data for specific purpose.

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

CHARACTERISTICS OF THE DATABASE APPROACH

1) Self describing nature of the database system.
2) Insulation between programs and data.
3) Support of multiple views of the data.
4) Sharing of data & multiuser transaction processing.

1) Self describing nature of the database system.

- Database system contains not only the database itself but also a complete definition or description of the database structure and constraints.

- This definition is stored in the DBMS catalog.
- Information stored in the catalog is called meta- data and it describes the structure of the primary database.

2) Insulation between Programs and Data, and Data Abstraction

- The structure of data files is stored in the DBMS catalog separately from the access programs.
- This property is called program-data independence.
- An operation (also called a function or method) is specified in two parts.
- Interface
  - The interface (or signature) of an operation includes the operation name and the data types of its arguments (or parameters).
- Implementation
  - The implementation (or method) of the operation is specified separately and can be changed without affecting the interface.

2) Insulation between Programs and Data, and Data Abstraction

- User can operate on the data by invoking these operations – Program operation independence.
- The characteristic that allows program-data independence and program operation independence is called data abstraction.

3) Support of Multiple Views of the Data

- A database has many users, each user may require a different perspective or view of the database.
- A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

4) Sharing of Data and Multiuser Transaction Processing

- DBMS must include concurrency control software
  - to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct
- DBMS must enforce several transaction properties
  - Atomicity
  - Consistency
  - Isolation
  - Durability

1) Atomicity

- Either the entire transaction takes place at once or doesn't happen at all. There is no midway.
- ie; Transactions do not occur partially.
- 'All or Nothing' rule.
- Eg: Transfer of 100 from account A to account B.
- 2 events:
  - □ Debiting 100 from A's balance
  - □ Creating 100 to B's balance.

Before	A:500	B:200
	Read(A)	Read(B)
	A = A-100	B = B+100
	Write(A)	Write(B)
After	A:400	B:300

2) Consistency

- Database is consistent before and after a transaction
- It refers to correctness of a database.
- In the previous eg., the total amount in both the account before and after the transaction must be maintained.

3) Isolation

- One transaction should start execution only when the other finished execution.

```
Read(X)          Read(X)
X = X*100        Read(Y)
Write(X)         Z = X+Y

            Write(Z)
```

4) Durability

- Ensures that once the transaction has completed execution, the updates should be permanent & they persist even if system failure occurs.

ACTORS ON THE SCENE

- People whose jobs involve the day-to-day use of a database.
- For a small personal DB, one person typically Defines, Constructs & Manipulates the db.
- In large organization, many people are involved.

1) Database Administrators (DBA)
2) Database Designers
3) End Users
4) System Analysts & Application Programmers/Software Engineers

1. Database Administrators

- Administrating the resources in a database environment
- Primary resource – Database
- Secondary resource – DBMS & the related software.

a) Authorizing access

b) Coordinating & monitoring its use

c) Acquiring software & hardware resources as needed.

2. Database Designers

- Identify the data to be stored in the DB.
- Choose appropriate structures to represent this data.
- Communicate with all db users to understand their requirements.
- Develop views for each user.
- Integrate all the views to develop the final database design.

3. End Users

- People who access the DB for querying, updating & generating reports.
- Database primarily exist for their use.

a) Casual end users

b) Naive/ Parametric end users

c) Sophisticated end users

d) Standalone users

a) Casual end users

- Occasionally access the DB.
- They may need different information each time.
- Eg: Occasional browsers.

b) Naive/ Parametric end users

- They make up a sizable portion of DB end users.
- They constantly query & update the DB using **canned transactions –** carefully programmed & tested.
- Eg: Bank Tellers – check account balance & post deposits and withdrawals

c) Sophisticated end users

- They thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirement.
- Eg: Engineers, scientists, business analysts, etc.

d) Standalone users

- Maintain personal databases by using ready made program packages that provide easy to use menu based or graphics based interfaces.

4. System Analysts & Application Programmers

- System Analysts determines the requirements of naive & parametric end users & develop specifications for standard canned transactions.
- Application programmers implement these specifications as programs, then they test it, debug, document & maintain these canned transactions.
- They are also called as Software Developers or Software Engineers.

WORKERS BEHINDTHE SCENE

- People who work to maintain the database software environment.

- They are not interested in the DB contents.

1) Database System Designers & Implementers

- DBMS modules & interfaces as a Software package.

2) Tool Developers

- Design & Implement tools to improve performance.

3) Operators & Maintenance Personnel/ System Administration Personnel

- Responsible for actual running & maintenance of the hardware & software environments.

ADVANTAGES OF USING THE DBMS

1) Controlling Redundancy
2) Restricting unauthorized access
3) Providing storage structures for efficient query processing
4) Providing backup & recovery
5) Providing multiple user interfaces
6) Representing complex relationships among data
7) Enforcing integrity constraints
8) Permitting inferencing & actions using rules

1. Controlling Redundancy

- Storing same data multiple times leads to duplication of effort & wastage of storage space.

2. Restricting Unauthorized Access

- When multiple users share a large DB, it is likely that most users will not be authorized to access all information.
- DBMS provides security & authorization subsystem, which the DBA uses to create accounts & to specify account restrictions.

3. Providing storage structures & search techniques

- For efficient query processing.
- DBMS provides specialized data structures & search techniques to speed up disk search.
- It has a buffering/ caching module that does data buffering.
- Query processing & Optimization module is responsible for choosing an efficient query execution plan.

4. Providing Backup & Recovery

- The backup & recovery subsystem of the DBMS is responsible for recovery from hardware & software failures.
- For eg., if the computer system fails in the middle of a transaction, the recovery subsystem is responsible for making sure that the DB is restored to the state it was in before the transaction started executing.
- Disk backup is also necessary in case of any disk failure.

5. Providing Multiple User Interfaces

- Because many types of users with varying levels of technical knowledge use a DB, a DBMS should provide a variety of UI’s.
- Programming language interfaces for application programmers.
- Forms & Command codes for parametric users
- Menu driven & natural language interface for standalone users
- Both form-based & menu driven interfaces are commonly known as Graphical User Interfaces (GUI).

6. Representing Complex Relationships among data

- A DB may include numerous varieties of data that are interrelated in many ways.

- A DBMS must have the following capabilities:
- to represent a variety of complex relationships among the data

- to define new relationships as they arise

- to retrieve & update related data easily & efficiently.

7. Enforcing Integrity Constraints

- Specifying datatype for each data item – to ensure accuracy & consistency of data in DB.

8. Permitting Inferencing & Actions using Rules

- Inferencing new information from the stored database facts.

- Associate triggers with tables.

DISADVANTAGES OF USING THE DBMS

1) Cost of Hardware & Software
2) Cost of Data Conversion
3) Cost of Staff Training
4) Appointing Technical Staff
5) Database Damage

DATA MODELS

- Collection of concepts that can be used to describe the structure of a DB.
- Most data models also include a set of basic operations for specifying retrievals and updates on the database.
- Different types of data models are:

1) High-level (or conceptual) data models
2) Representational (or implementation) data models
3) Low-level (or physical) data models

1. High-level or conceptual data models

- Provide concepts that are close to the way many users perceive data.

- Use concepts such as entities, attributes, and relationships.
- An entity represents a real-world object. Eg: an employee, student, teacher, etc.

- An attribute represents some property of interest that further describes an entity. Eg: name, salary, etc.

- A relationship among two or more entities represents an association among them. Eg: works-on, teaches.

1. High-level or conceptual data models

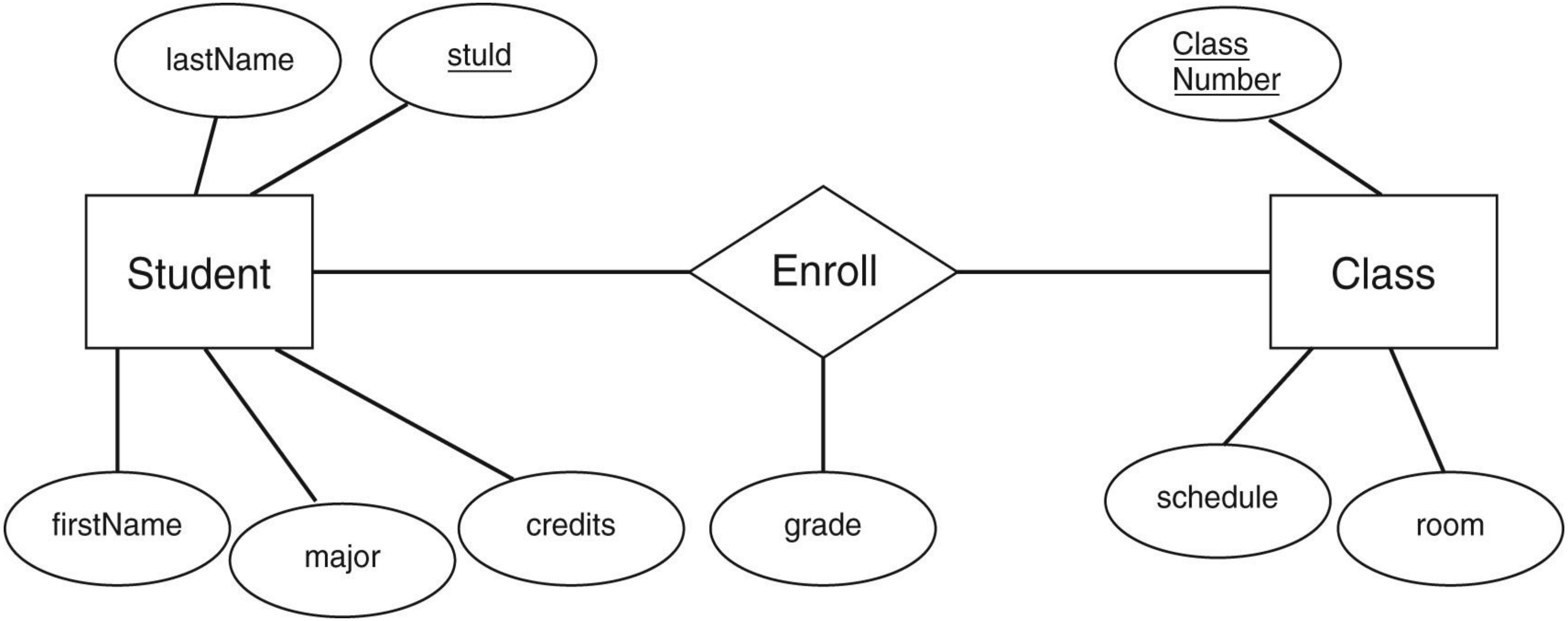
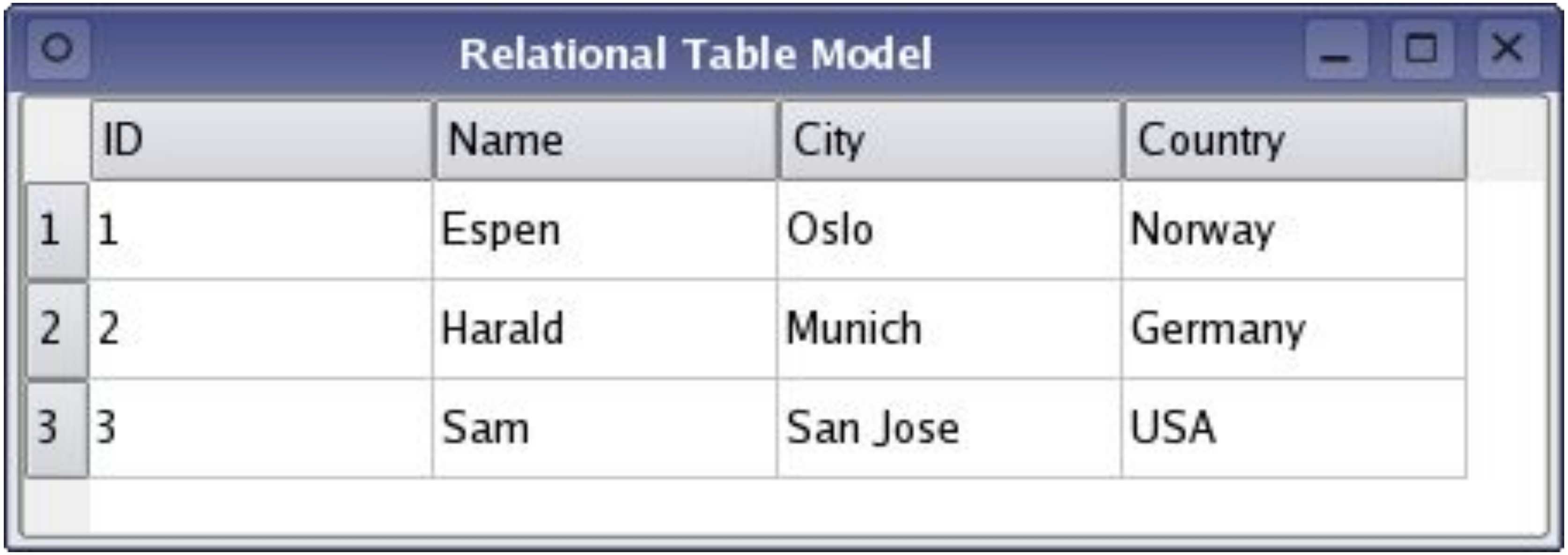


Fig: ER Model

2. Representational data models

- Most frequently used data models.
- Provide concepts that may be easily understood by end users.
- 3 types: Relational, Network & Hierarchical

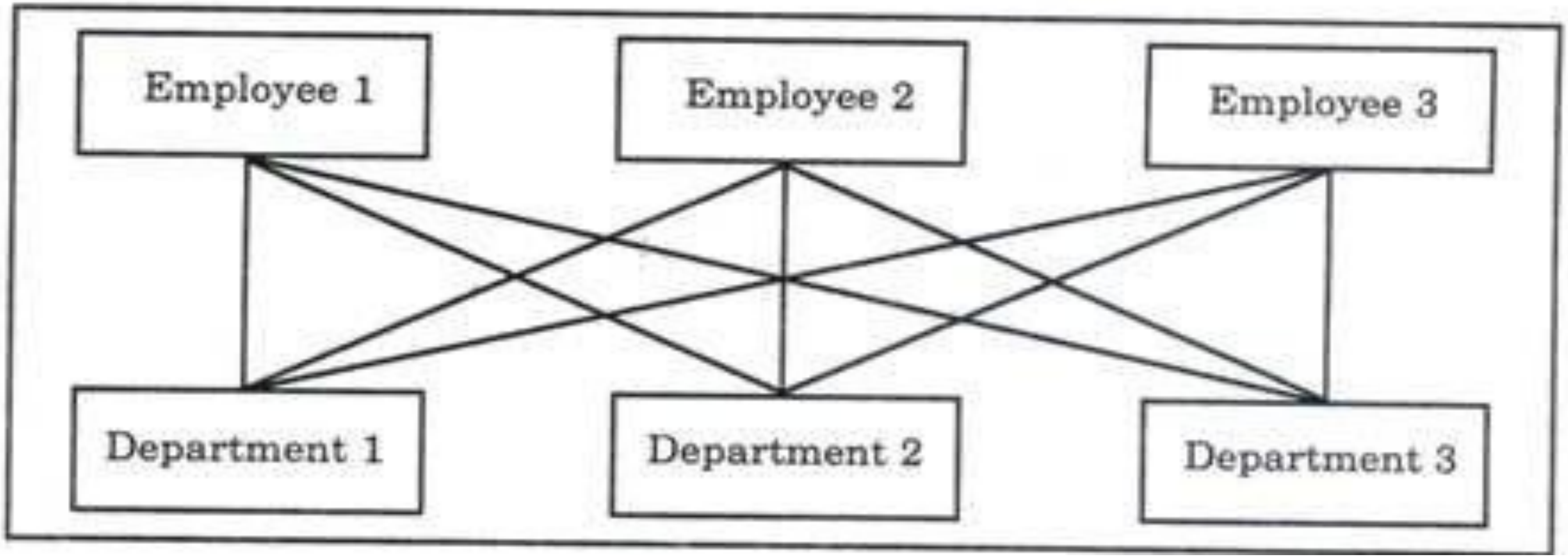
a) Relational data model



	ID	Name	City	Country
1	1	Espen	Oslo	Norway
2	2	Harald	Munich	Germany
3	3	Sam	San Jose	USA

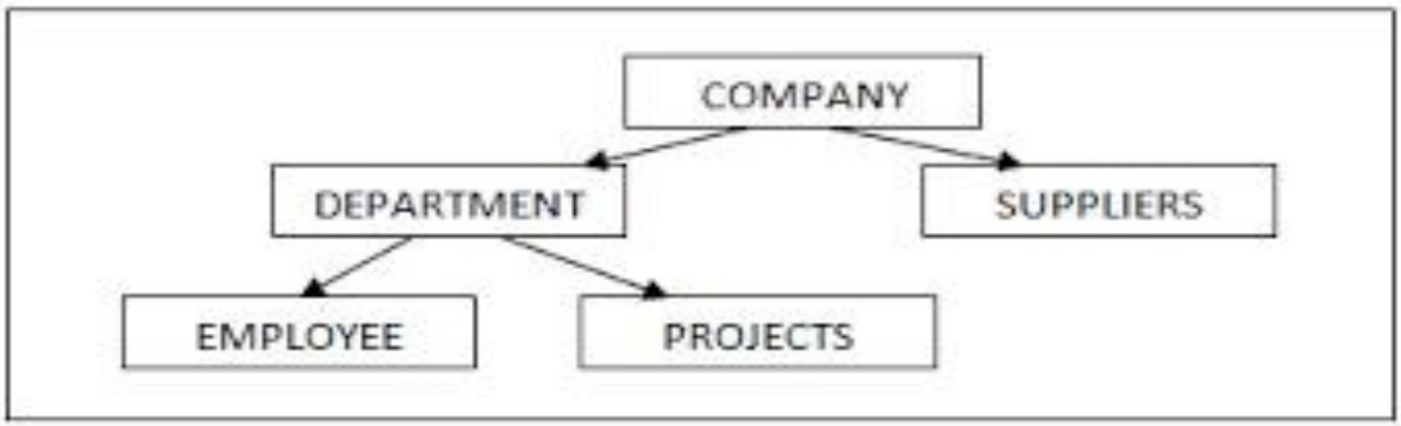
b) Network data model

- In the network model of database, there are no levels and a record can have any number of owners and also can have ownership of several records.



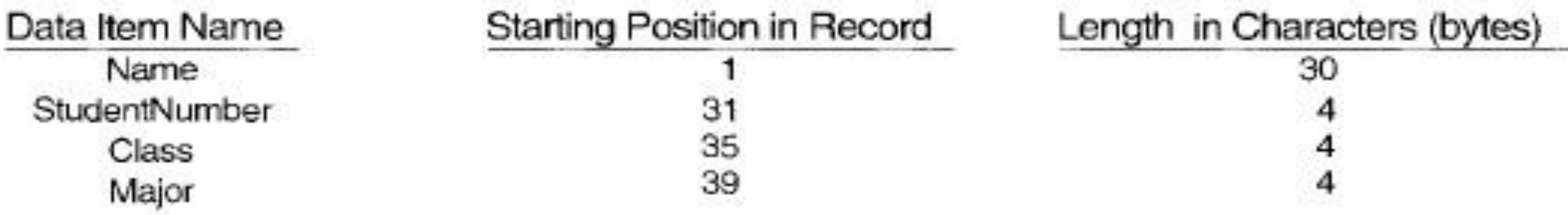
c) Hierarchical data model

- This model presents data to users in a hierarchy of data elements that can be represented like a tree.



3. Low level or Physical data models

- Provide concepts that describe the details of how data is stored.
- Meant for computer specialists, not for end users.
- Eg. Record Format, access path, etc.

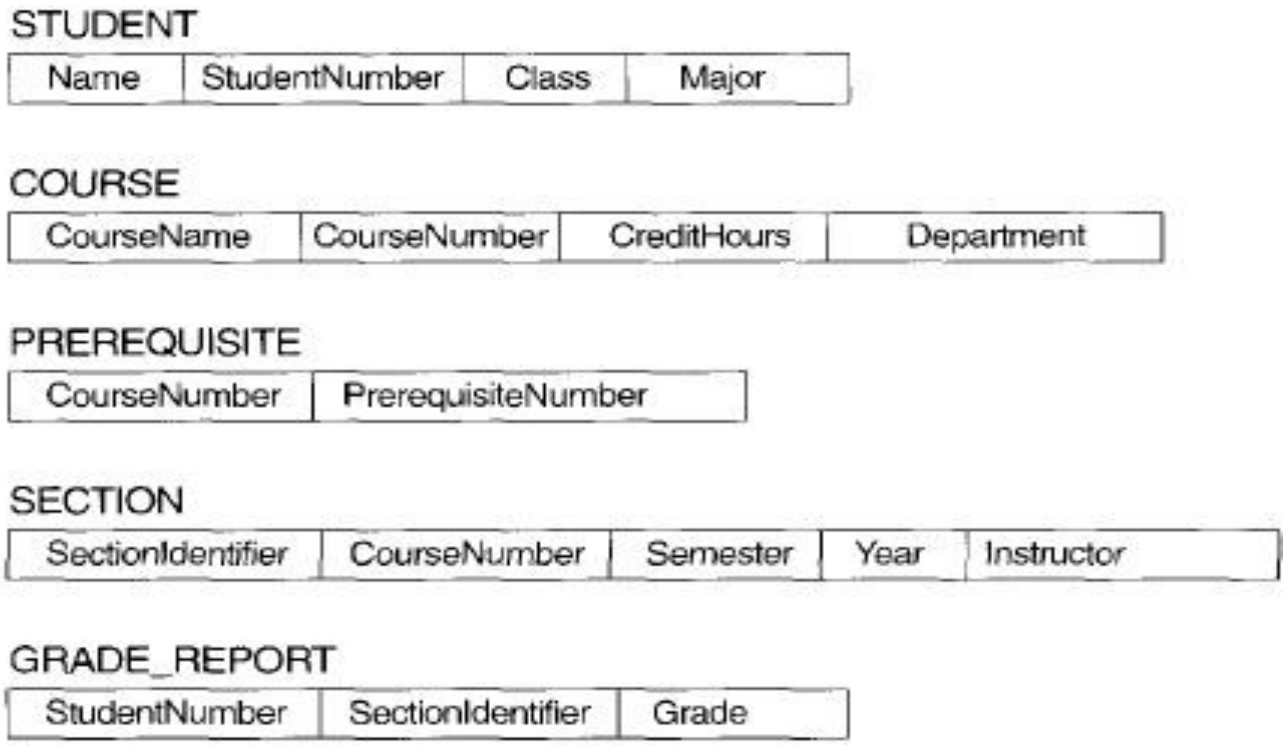


Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
StudentNumber	31	4
Class	35	4
Major	39	4

SCHEMAS, INSTANCES & DATABASE STATE

- The description of a DB is called the database schema, which is specified during database design and is not expected to change frequently.

- A displayed schema is called a schema diagram.



STUDENT

Name	StudentNumber	Class	Major

COURSE

CourseName	CourseNumber	CreditHours	Department

PREREQUISITE

CourseNumber	PrerequisiteNumber

SECTION

SectionIdentifier	CourseNumber	Semester	Year	Instructor

GRADE_REPORT

StudentNumber	SectionIdentifier	Grade

- A schema diagram shows only some aspects of a schema, such as name.

- It neither shows the data types nor the relationships.

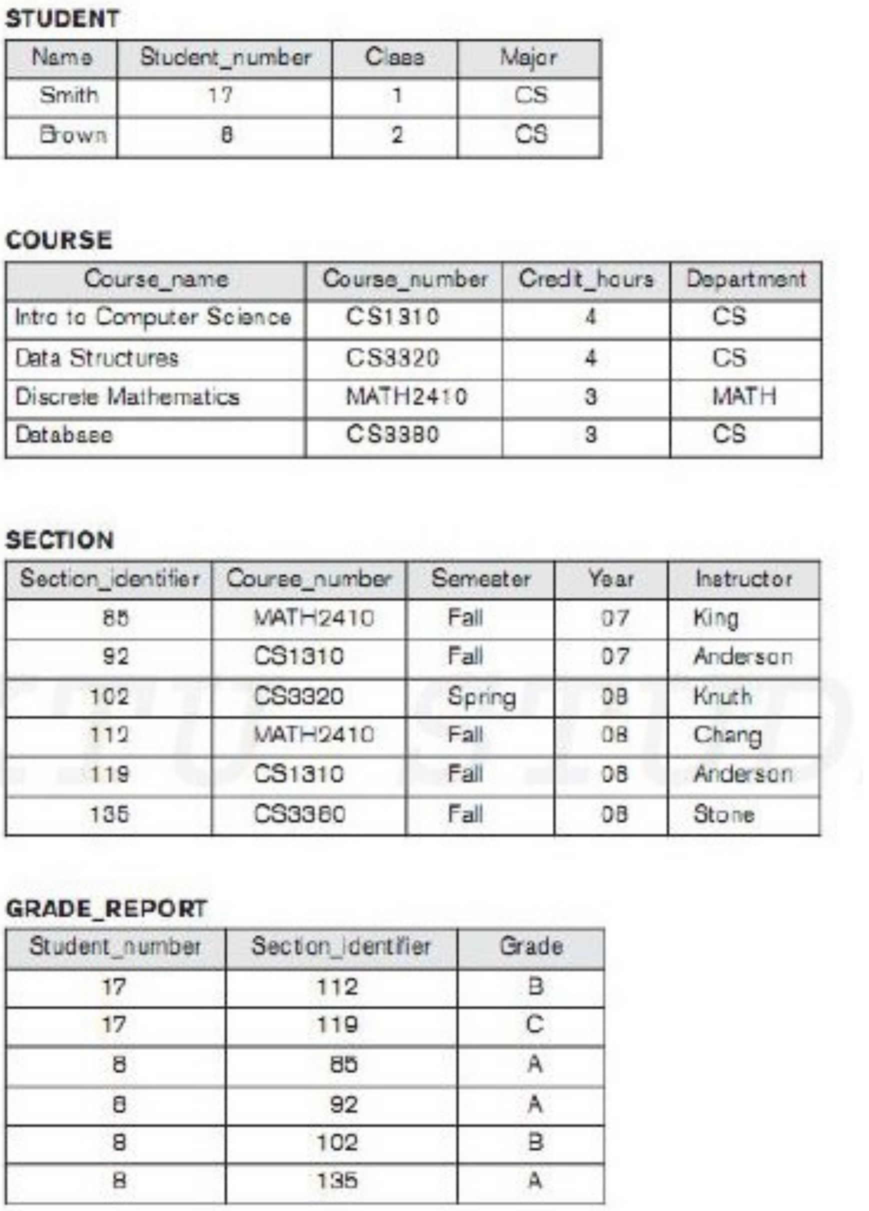
- The actual data in the DB changes frequently.
- The data in the DB at a particular moment in time is called the database state/ snapshot.

- It is also called the current set of occurrences/ instances in the DB.

- When we define a new DB, we specify its DB schema. At this point, the DB state is empty state with no data.

- We get the initial state of the DB when the DB is first populated with the initial data.
- From then on, every time an update operation is applied to the DB, we get another DB state.
- At any point of time, the DB has a current state.
- DBMS is responsible for ensuring that every state of the DB is a valid state.
- The schema is not supposed to change frequently, but as application requirements changes, schema will also be changed. This is known as Schema Evolution.

EXAMPLE STUDENT DATABASE



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

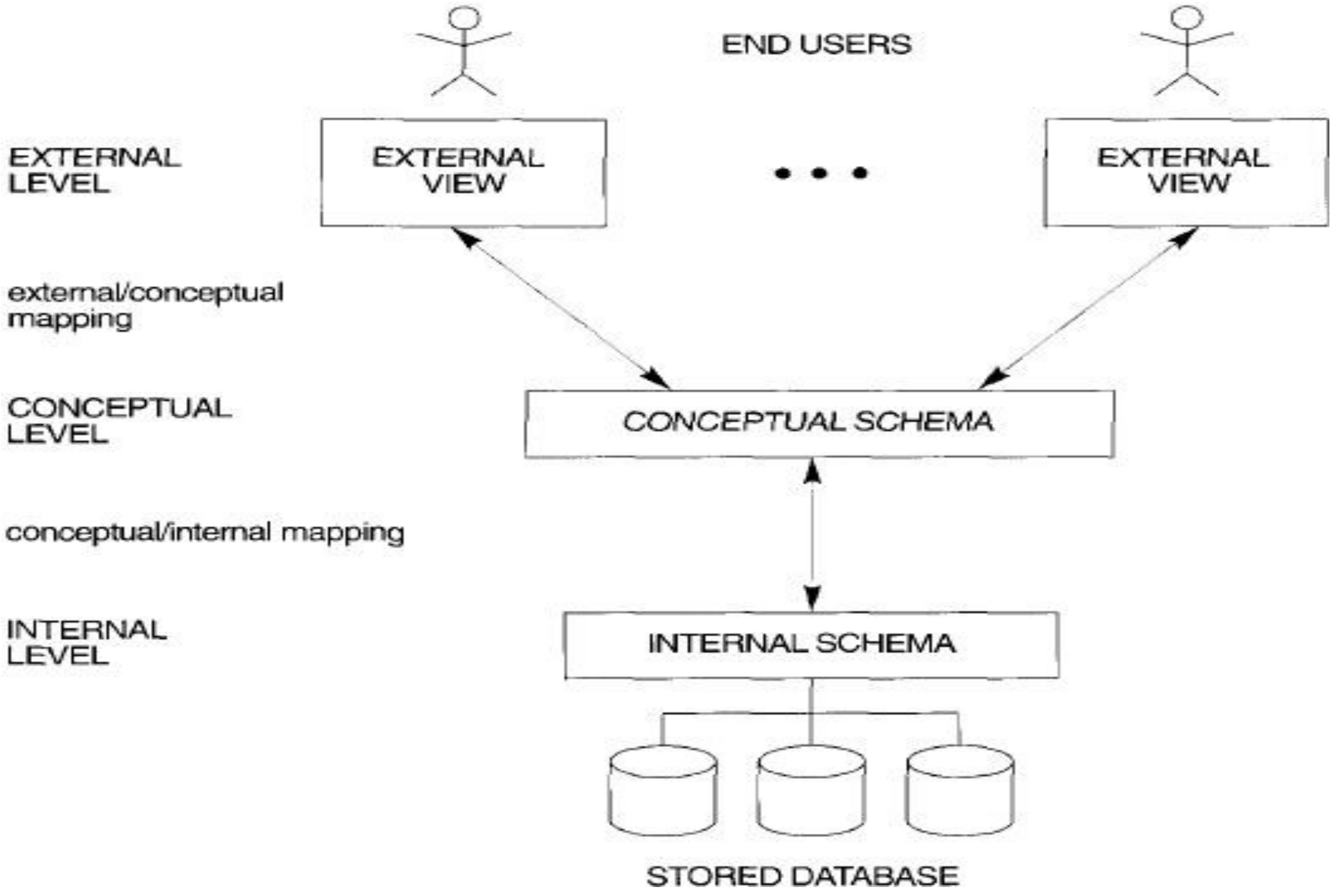
SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

THREE-SCHEMA ARCHITECTURE OF DBMS



- The main goal is to separate the user application from the physical DB.
- In this architecture, schemas can be defined at the following 3 levels: Internal, Conceptual & External.

1) Internal level

- Has an internal schema, which describes the physical storage structure of the DB.
- It is implemented using a physical data model.

2) Conceptual level

- It has a conceptual schema, which describes the entities, data types, relationships, etc.
- It uses a high level data model or a representational data model.

3) External/ View level

- It includes a number of external schemas.
- Each external schema describes the part of the DB that a particular user group is interested in & hides the rest of the DB from that user group.
- It also uses a representational data model.

- Working

1) Each user group refers to its own external schema.
2) DBMS must transform a request specified on an external schema into a request against the conceptual schema.
3) This is then transformed into a request on the internal schema for processing over the stored DB.
4) The data extracted from the stored DB must be reformatted to match the users external view.

- The process of transforming requests & results between levels are called mappings.

DATA INDEPENDENCE

- Defined as the capacity to change the schema at one level of a DB system without having to change the schema at the next higher level.

- Two types of data independence:

1) Logical data independence

- Capacity to change the conceptual schema without having to change the external schema.

2) Physical data independence

- Capacity to change the internal schema without having to change the conceptual/ external schema.

DATABASE LANGUAGES

- Different types of database languages are:

1) Data Definition Language (DDL)
2) Storage Definition Language (SDL)
3) View Definition Language (VDL)
4) Data Manipulation Language (DML)
5) Data Control Language (DCL)
6) Transaction Control Language (TCL)

1) Data Definition Language (DDL)

- Statements are used to define the database structure or schema.

  CREATE - to create objects in the database

  ALTER - alters the structure of the database

  DROP - delete objects from the database

  COMMENT - add comments to the data dictionary

  RENAME - rename an object

2) Storage Definition Language (SDL)

- In some DBMSs, the DDL is used to specify the conceptual schema only.
- SDL is used to specify the internal schema.

3) View Definition Language (VDL)

- VDL is used to specify the external schema.
- In most DBMS, DDL is used to define all the 3 schemas.

4) Data Manipulation Language (DML)

- Statements are used for managing data within schema objects.

  SELECT - Retrieve data from the database

  INSERT - Insert data into a table

  UPDATE - Updates existing data within a table

  DELETE - deletes all records from a table

5) Data Control Language (DCL)

GRANT - gives user access privileges to database

REVOKE - withdraw access privileges given with the GRANT command.

6) Transaction Control Language (TCL)

COMMIT - save work done

ROLLBACK - restore database to original since the last COMMIT.

THE DATABASE SYSTEM ENVIRONMENT

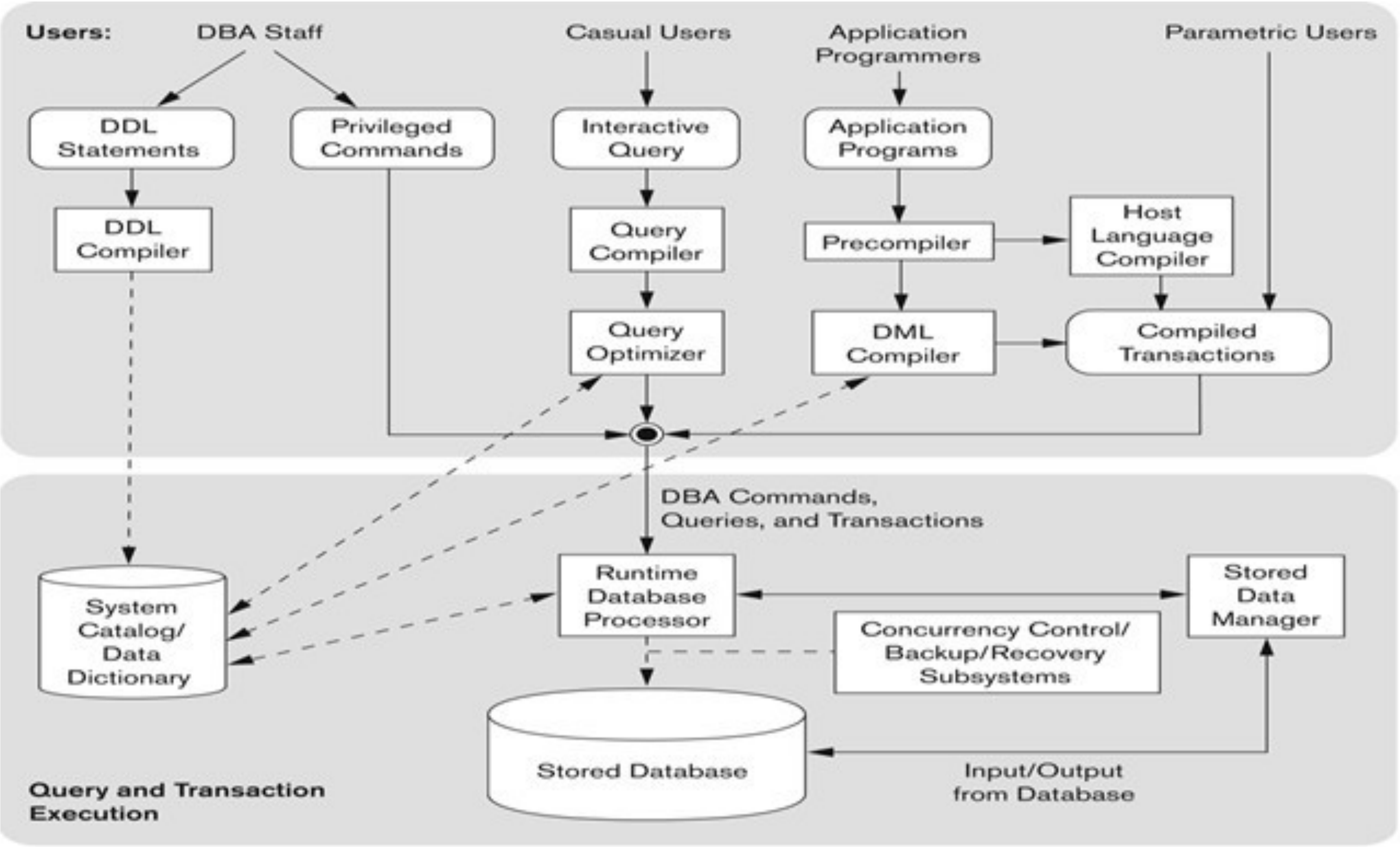


Figure 2.3
Component modules of a DBMS and their interactions.

DBMS Component Modules

- The top part refers to the various users of the DB environment & their interfaces.
- The lower part shows the internals of the DBMS responsible for storage of data & processing of transactions.
- DBA staff defines the DB by using DDL & other privileged commands.
- DDL compiler processes the DDL statements & stores the result in the DBMS catalog.

- Casual users interact using interactive query interface. Query compiler compiles them into an internal form. Query optimizer does optimization & elimination of redundancies.

- Application programmers write programs using host programming languages (C, C++, Java) that are submitted to a precompiler, which extracts DML commands & are passed to a DML compiler.

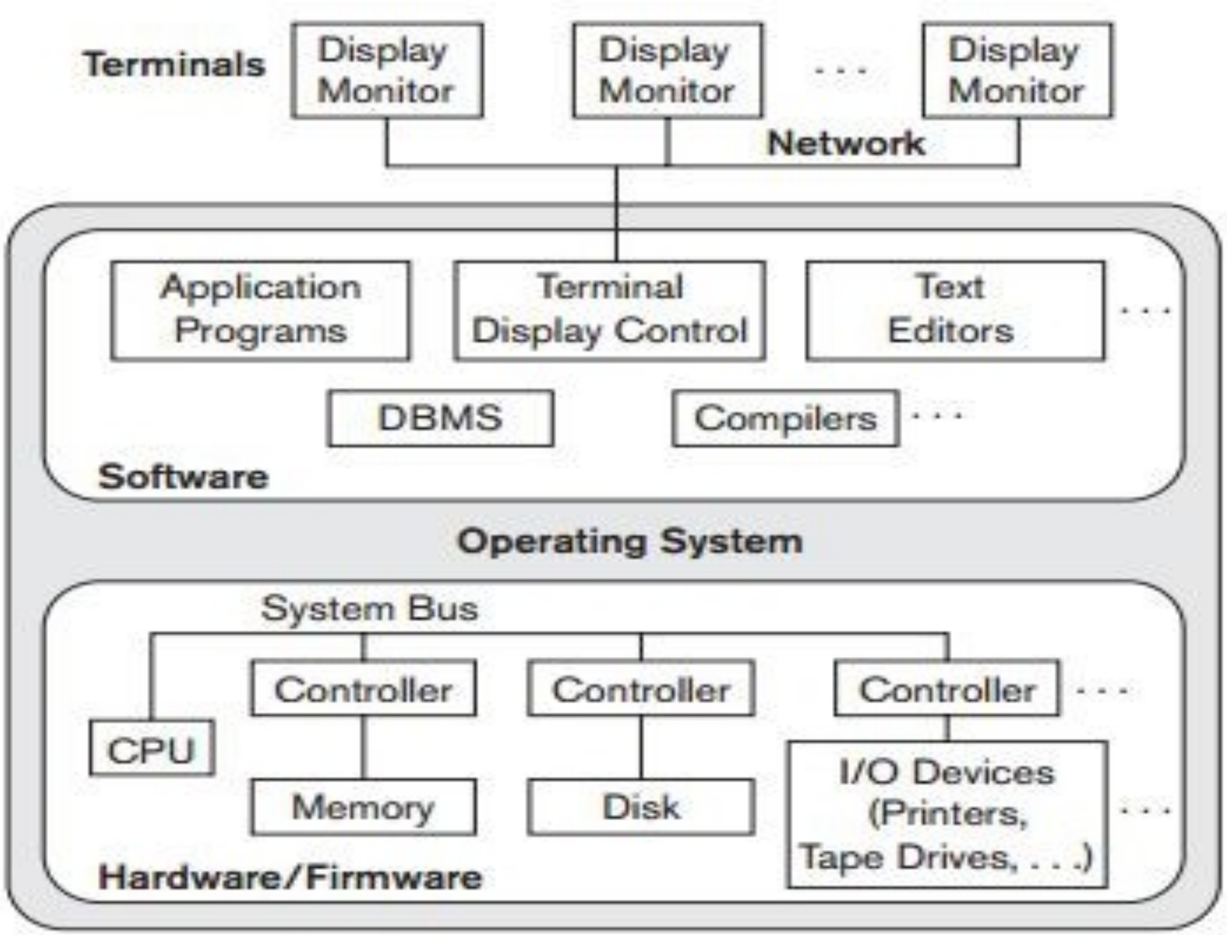
- Parametric users use these canned transactions.

- Runtime Database Processor executes the privileged commands, executable query and the canned transactions.
- Stored data manager module controls access to the DBMS information that is stored on disk.
- Concurrency control, Backup & recovery subsystems are integrated for transaction management.

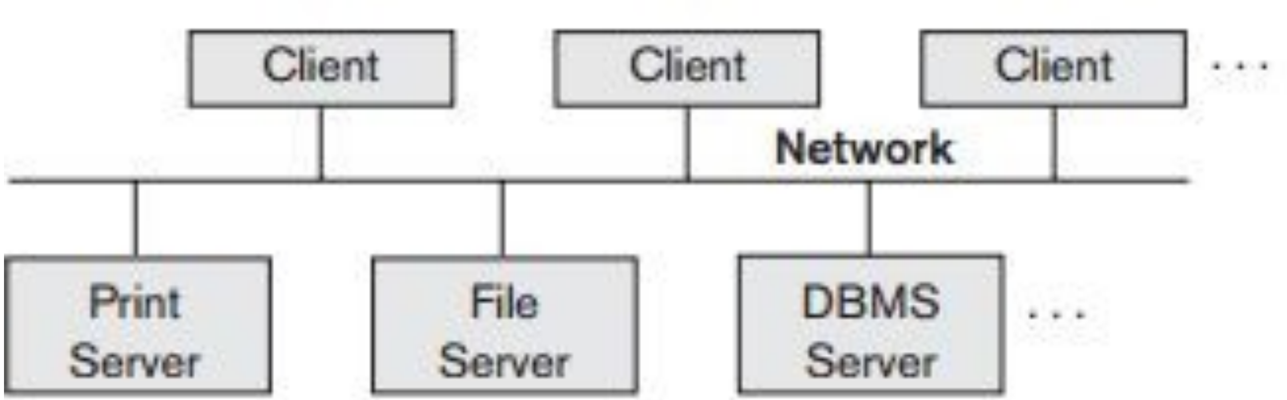
Database Architectures

- Centralized DBMSs Architecture
- Basic Client/Server Architectures
- Two-Tier Client/Server Architectures
- Three-Tier and n-Tier Architectures for Web Applications

Centralized DBMS Architecture



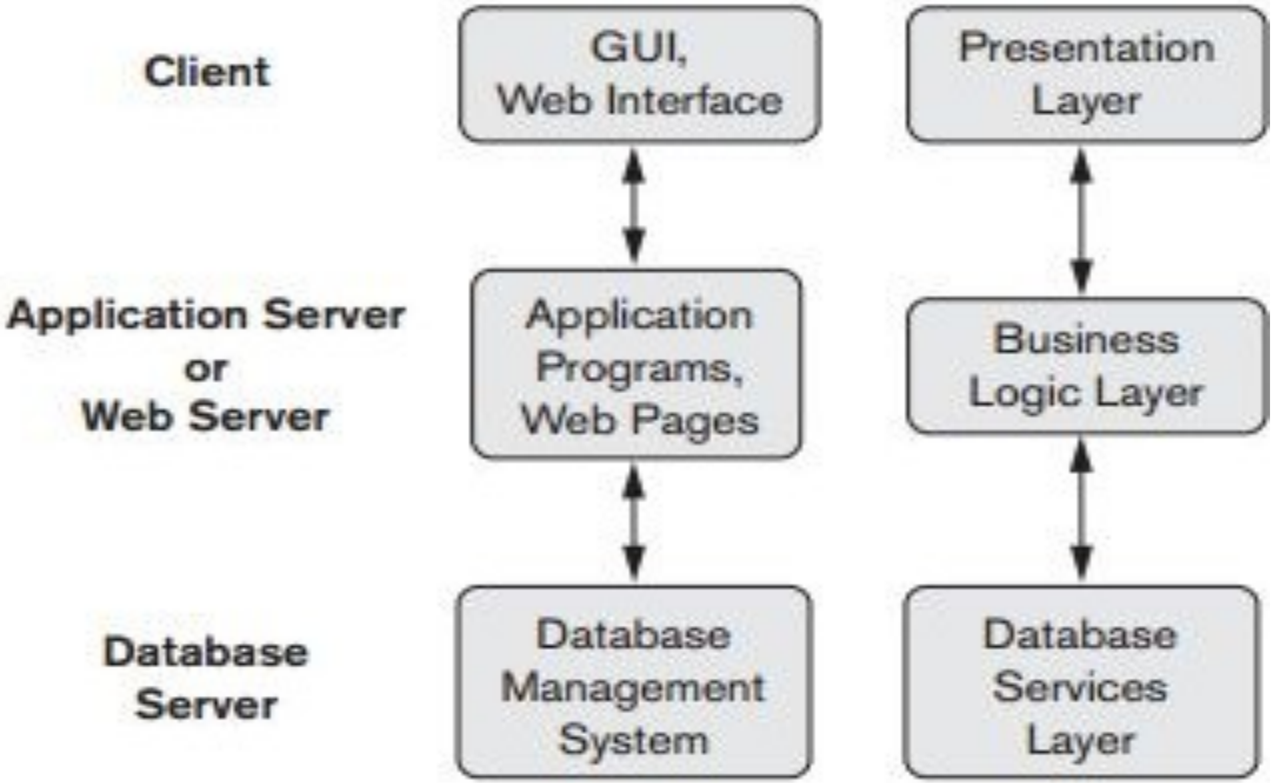
Basic Client/Server Architectures



Two-Tier Client/Server Architectures

- There exists a logical dividing point between the client and the server
- Server is called query server or transaction server
- User interface programs and application programs run on client side
- When DBMS access is required ,the program establishes a connection to the DBMS.
- Done with the help of Open database connectivity(ODBC) which provides an Application programming interface(API)

Three-Tier and n-Tier Architectures for Web Applications



Client

GUI, Web Interface

Presentation Layer

Application Server or Web Server

Application Programs, Web Pages

Business Logic Layer

Database Server

Database Management System

Database Services Layer

Classifications of DBMS

Based on Data Model

- Hierarchical data model
- Relational data model
- Object data model
- Network data model

These are called legacy database systems.

- Object Relational data model
- Native XML DBMSs

Based on number of users:

- Single User System
- Multi User System

Number of sites over which the database is distributed:

- Centralized
- Distributed

Distributed DB is again divided into based on the DBMS software used:

- Homogenous
- Heterogeneous

Middleware Software

Middleware is software that lies between an operating system and the applications running on it

Federated Database

A federated database system (FDBS) is a type of meta-database management system (DBMS), which transparently maps multiple autonomous database systems into a single federated database.

Based on purpose:

- General purpose
- Special purpose