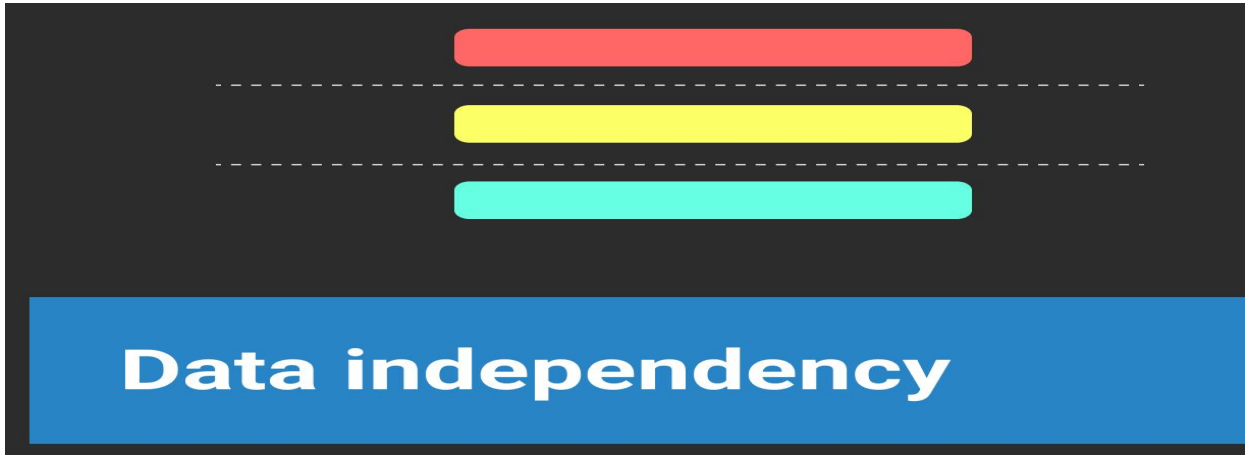
Database System Concepts and Architecture



Introduction and motivation, Data independence, Three schema architecture, Centralized and Client/Server architectures, Database components, Database users, Entity Types, Entity Sets, Attributes, Entity Type (Strong and Weak), Relationship Types, Relationship Sets, Roles, Structural Constraints, ER diagram construction.

Data Models, Schemas, and Instances

Data Models

- Data models refer to how the logical structure of a database is made.
- The data model acts as a conceptual tool that describes the data relationship, data semantics, and data constraints.
- The database uses a tier architecture so we need to design the database at physical, logical, and view level

Categories of data model

- Relational Data Model
- ER(Entity-Relationship) Data Model
- Object-based Data Model
- Semistructured Data Model

Data Models, Schemas, and Instances

Schema

- It is used to logically represent the structure of a database along with the relationship and organization between them.
- It is considered a blueprint of a database that shows the relationship between tables.

Types of Schema

Conceptual schemas

- It is used to gather the initial requirements of projects by offering a detailed view of the system.

Logical database schemas

- It implies the rules that govern the database and define schema with tables, view, relationship, and integrity constraints.

Physical database schemas

- Data gets stored within disk storage i.e. this schema refers to the actual storage of data.

Data Models, Schemas, and Instances

Schema Example

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

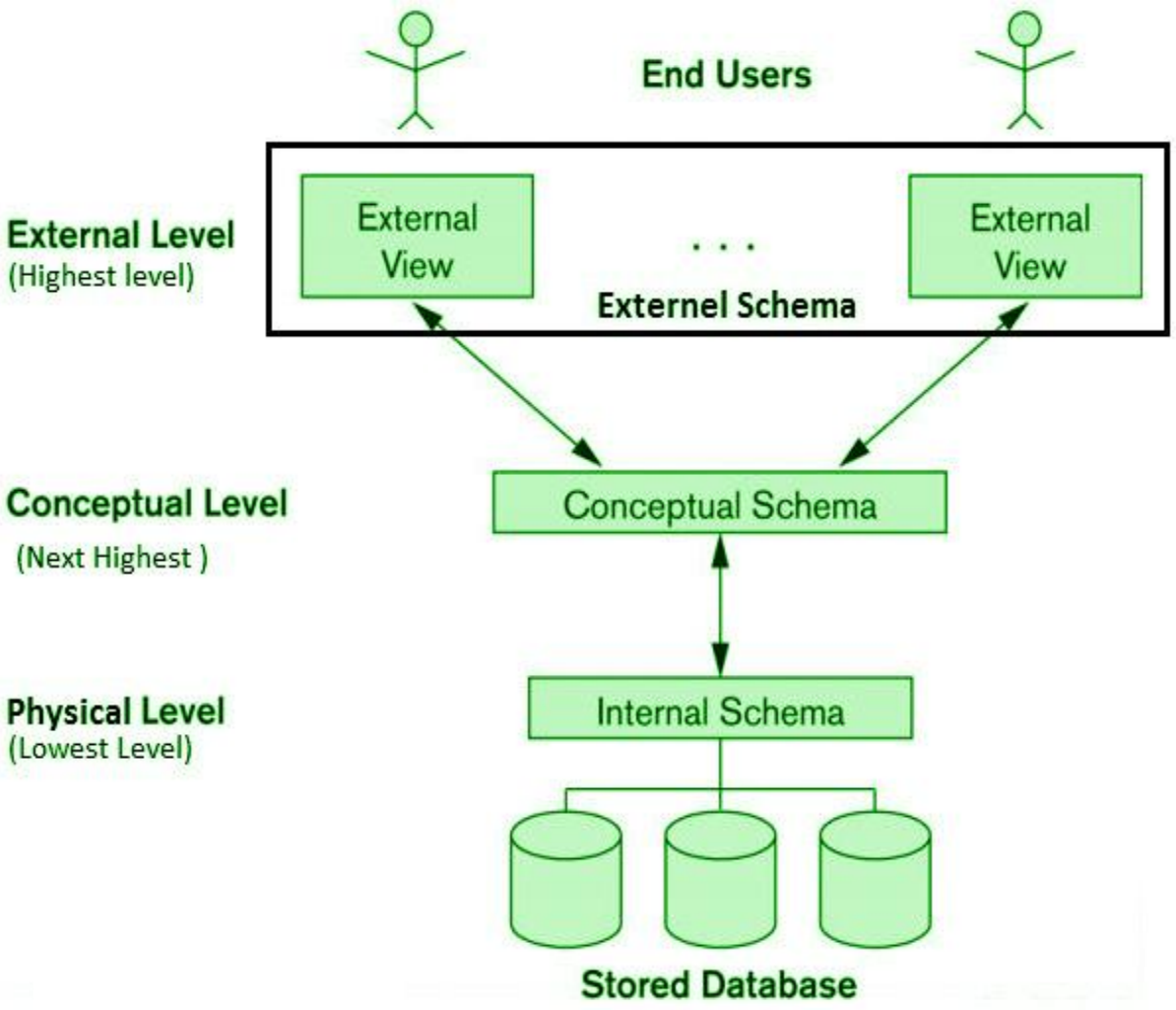
Student_number	Section_identifier	Grade

Data Models, Schemas, and Instances

Instance

- The collection of information stored in the database at a particular moment is called an instance.
- It contains a snapshot of the database.
- The data present in the database can be changed frequently.
- The database instance is equal to the value of the variable.

The Three-Schema Architecture



- The internal level has an internal schema, which describes the physical storage structure of the database.
- The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users.
- The external or view level includes some external schemas or user views.

Data Independence

- Data independence can be explained using the three-schema architecture.
- Data independence refers to the property of DBMS through which we can modify the schema definition at any level without changing the schema definition at any higher level.
- The main purpose of the three levels of data abstraction is to achieve data independence. As the database changes and expands over time, the changes in one level mustn't affect the data at other levels of the database. This would save time and cost required while changing the database.

Types of data independence

Two types of data independence:

1.Physical Data Independence

2.Logical Data Independence

Physical Data Independence

- Physical Data Independence refers to the characteristic of changing the physical level without affecting the logical level or conceptual level. Using this property we can easily change the storage device of the database without affecting the logical schema.

Example: Suppose you want to replace the storage device form hard disk to SSD or magnetic tape then it should not affect the data stored at the logical level.

The changes in the physical level may include changes like:

1.Using a new storage device like SSD, magnetic tape, hard disk, etc.

2.Using a new data structure for storage.

3.Using a different data access method or using an alternative file organization technique.

4.Changing the location(like changing the drive) of the database.

Logical Data Independence

- It refers to the characteristics of changing the logical level without affecting the external or view level. This also helps in separating the logical level from the view level.
- If we make any changes in the logical level then the user view of the data remains unaffected. The changes in the logical level are required whenever there is a change in the logical structure of the database.

The changes in the logical level may include:

- Changing the data definition.
- Adding, deleting, or updating any new attribute, entity or relationship in the database.

Examples: Students(sid: string, name: string, login: string, age: integer, gpa: real)
Faculty(fid: string, fname: string, sal: real)

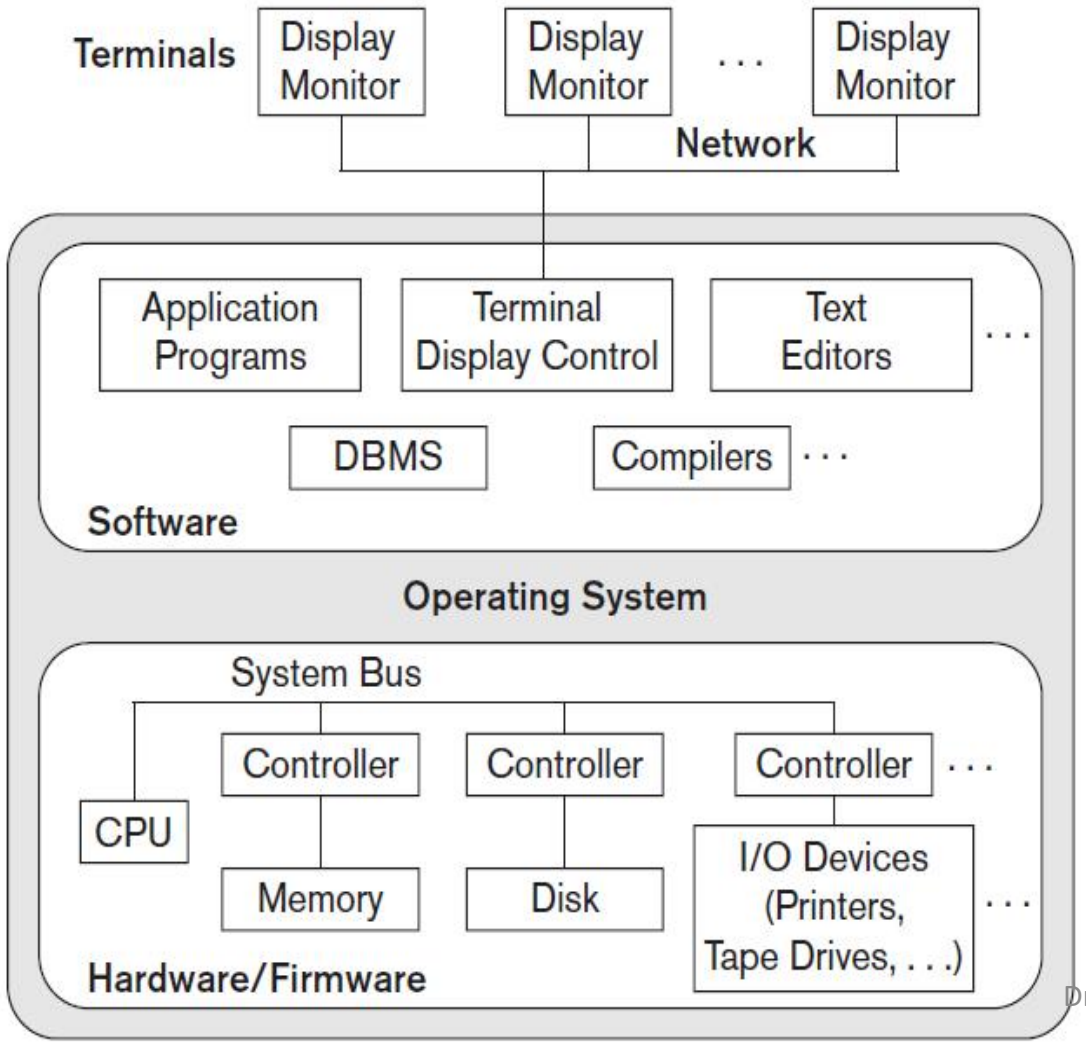
Centralized and Client-Server DBMS Architectures

Centralized DBMSs Architecture

- A centralized database 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.
- It is maintained and modified from that location only and usually accessed using an internet connection such as a LAN or WAN. The centralized database is used by organizations such as colleges, companies, banks etc.

Centralized and Client-Server DBMS Architectures

Centralized DBMSs Architecture



Advantages:

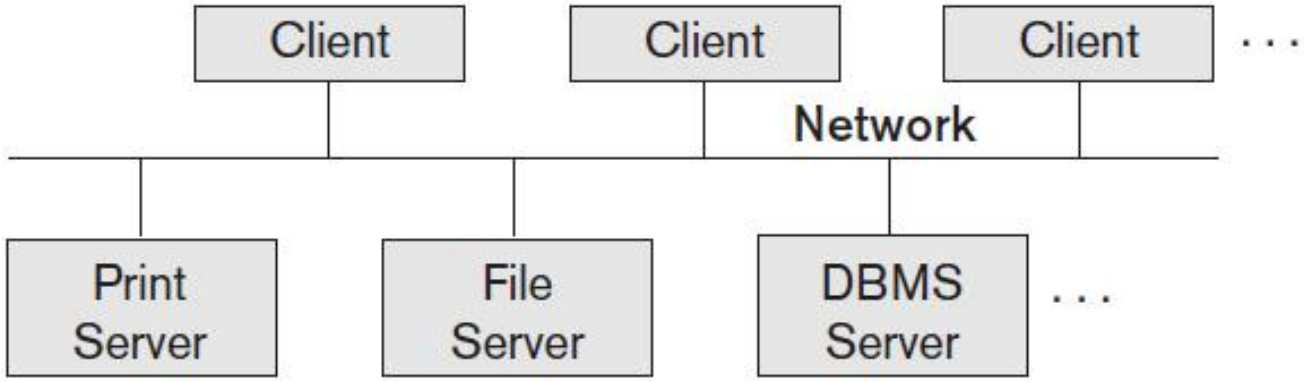
- Since all data is stored at a single location only thus it is easier to access and co-ordinate data.
- The centralized database has very minimal data redundancy since all data is stored in a single place.
- It is cheaper in comparison to all other databases available.

Disadvantages:

- The data traffic in the case of a centralized database is more.
- If any kind of system failure occurs at a centralized system then entire data will be destroyed.

Centralized and Client-Server DBMS Architectures

Basic Client/Server Architectures



Logical two-tier client/server architecture.

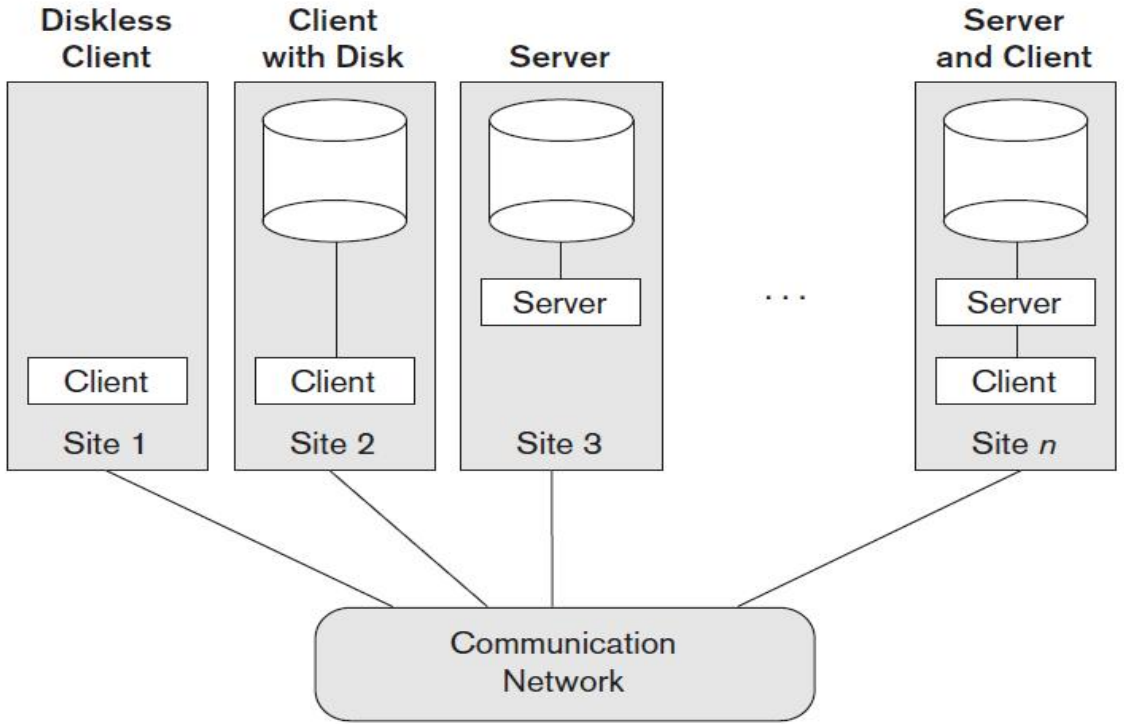
- The client/server architecture is based on the hardware and software components that interact to form a system.
- The system includes three main components: Clients, Servers and Communication Middleware

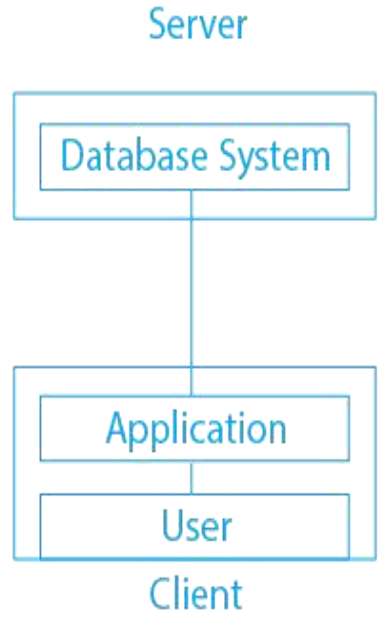
Two main types of basic DBMS architectures were created on this underlying client/server framework:

two-tier and **three-tier**

Centralized and Client-Server DBMS Architectures

Physical two-tier client/server architecture.





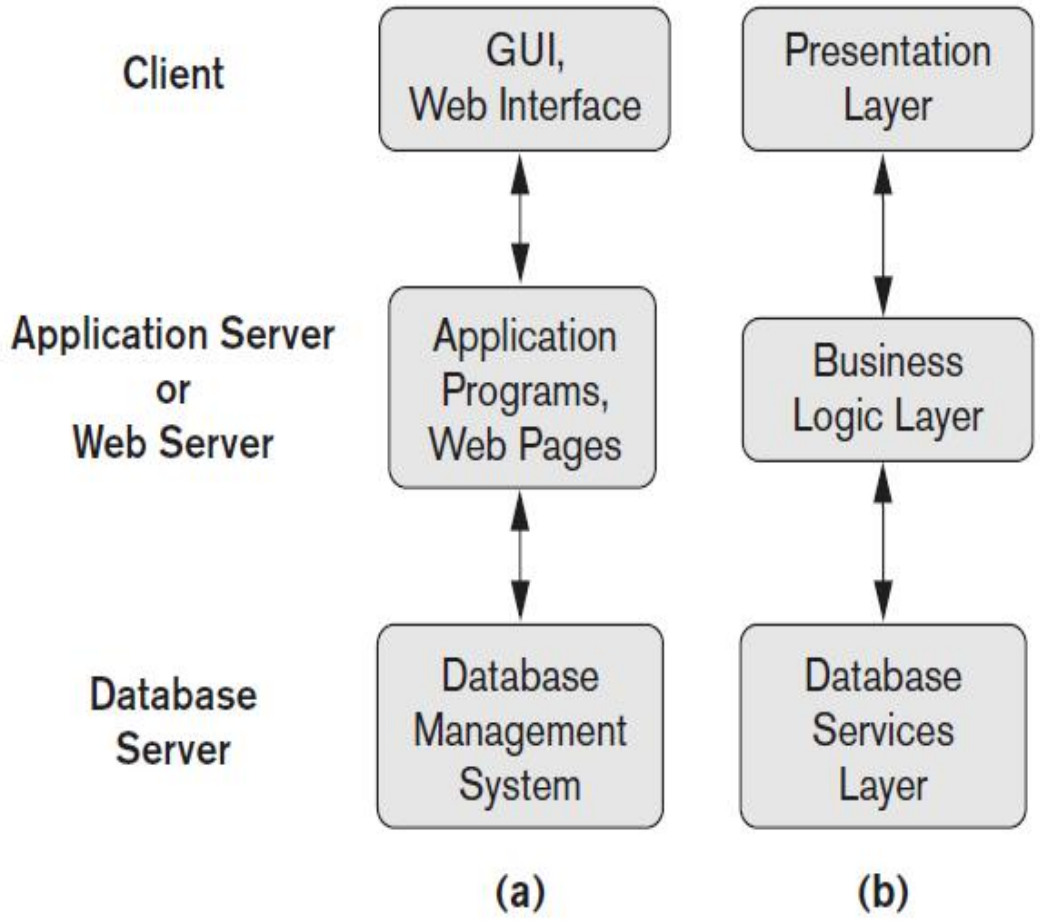
In a 2- tier architecture, client connects directly to Database

- In a Two-tier Architecture, the server manages the database functionality. It makes it possible for the clients to use the Database through APIs (Application Programming Interfaces) over a direct internet connection.

ODBC (Open Database Connectivity) and JDBC(Java Database Connectivity).

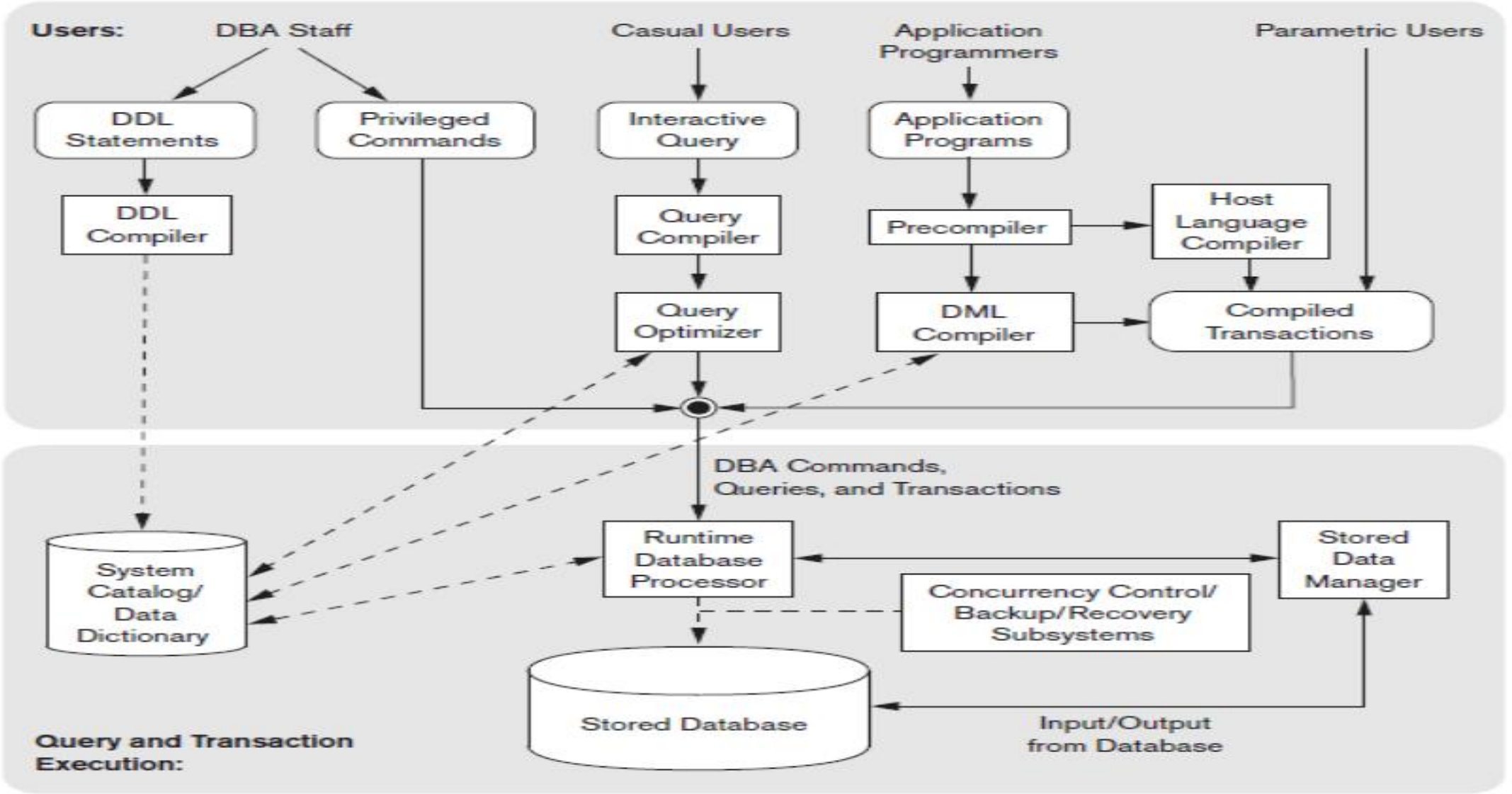
Centralized and Client-Server DBMS Architectures

Three-Tier and n-Tier Architectures for Web Applications.



- The client cannot communicate directly with the server with this design.
- On the client side, the program communicates with an application server, which then communicates with the database system.

DBMS Component Modules



Component modules of a DBMS and their interactions (or) system Architecture

DBMS Component Modules

A DBMS Component Modules is divided into two modules:

- Query Processor
- Storage Manager

Query Processor

- The query Processor components in DBMS accepts SQL commands generated from a variety of user interfaces, produce query evaluation plans, and execute these.

DML compiler: This translates DML statements in a query language to low-level instructions that the query evaluation engine understands.

Embedded DML compiler: SQL commands can be embedded in host-language application programs, ex: JAVA or COBOL programs.

DDL interpreter: which interprets DDL statements and records them in a set of tables containing metadata.

Query Evaluation Engine: which executes low-level instructions generated by the DML compiler.

Application programs object code: which is the low-level instructions of the programs written by naïve users, which the query evaluation engine understands and executes.

DBMS Component Modules

Storage Manager

- The storage manager components in DBMS provide the interface between the physical level stored data in the database and the programs/queries requesting the stored data from the database.

Authorization and integrity manager: which tests for the satisfaction of integrity constraints and checks the user's to access data.

Transaction manager: This ensures that the data remains in a consistent state despite of the system failures, and the concurrent transaction executions proceed without conflict.

File manager: which manages the allocation of space on disk storage and the data structures used to represent the stored information on disk.

DBMS Component Modules

Storage Manager

Buffer manager: which is responsible for fetching data from disk storage into main memory, and deciding what data to cache in memory.

Lock manager: which keeps track of requests for locks and grants locks on the data in the database.

Recovery manager: this is responsible for maintaining a log and restoring the system to consistent state after a crash.

The disk storage components are

- **Data files** : which store the database itself.
- **Data Dictionary**: which stores metadata about the structure of the database.
- **Indices**: which provide fast access to data items that hold particular values.
- **Statistical Data:** which stores statistical information about the data in the database.