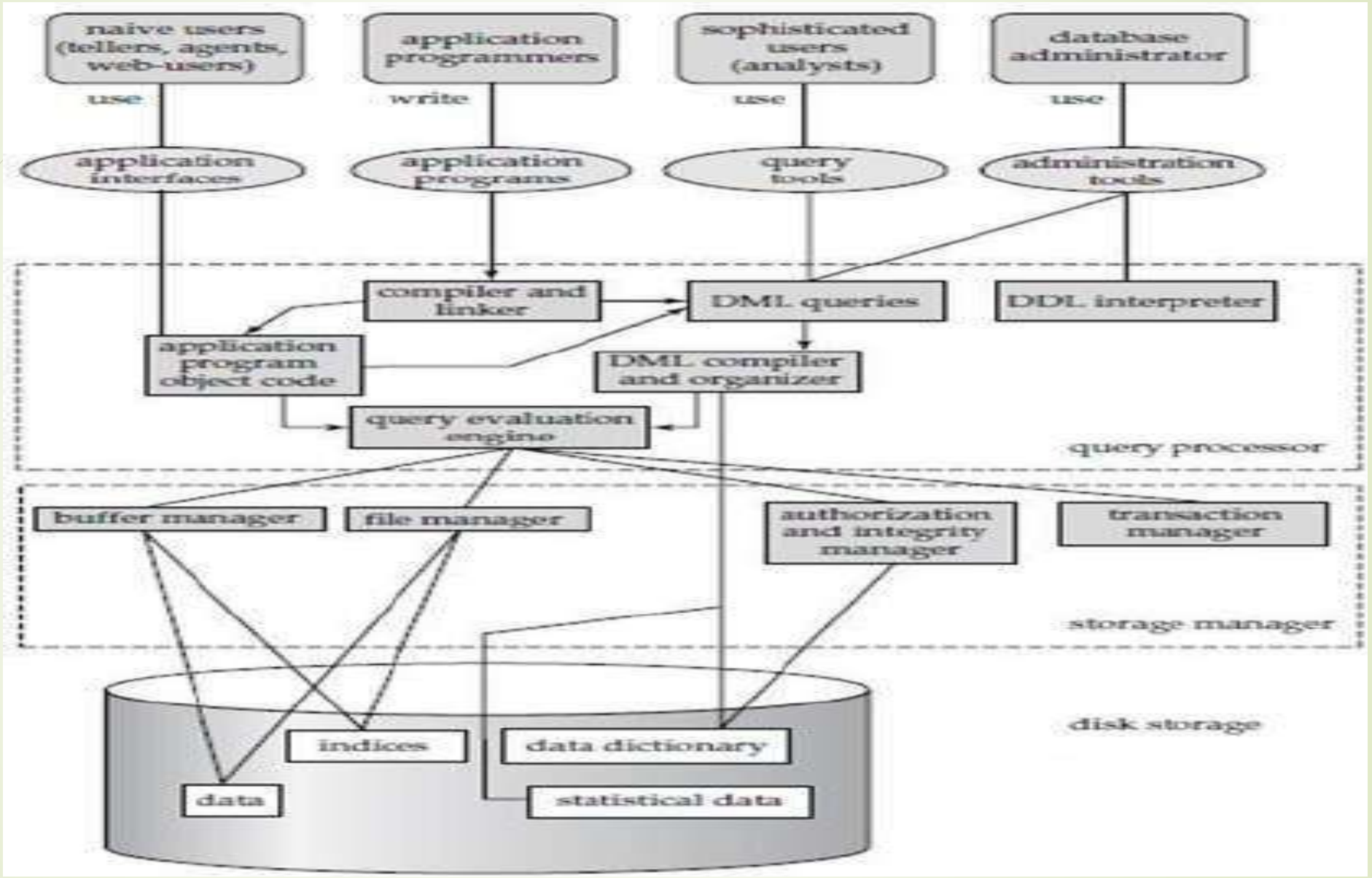
UNIT-1 DBMS

Database Architecture

- The architecture of a database system is greatly **influenced by the underlying computer system** on which the database system runs.
- **Database systems** can be **centralized**, or **client-server**, where one server machine executes work on behalf of multiple client machines. Database systems can also be designed to exploit parallel computer architectures. **Distributed databases** span multiple geographically separated machines.
- A database system is partitioned into modules that deal with each of the responsibilities of the overall system.


naive users (tellers, agents, web-users)
application programmers
sophisticated users (analysts)
database administrator
use
write
use
use
application interfaces
application programs
query tools
administration tools
compiler and linker
DML queries
DDL interpreter
application program object code
DML compiler and organizer
query evaluation engine
query processor
buffer manager
file manager
authorization and integrity manager
transaction manager
storage manager
indices
data dictionary
data
statistical data
disk storage

- The **functional components of a database system can be broadly divided into the storage manager and the query processor components.**
- The **storage manager** is important because databases typically require a large amount of **storage space**.
- The **query processor** is important because it helps the database system **simplify** and **facilitate access** to **data**.

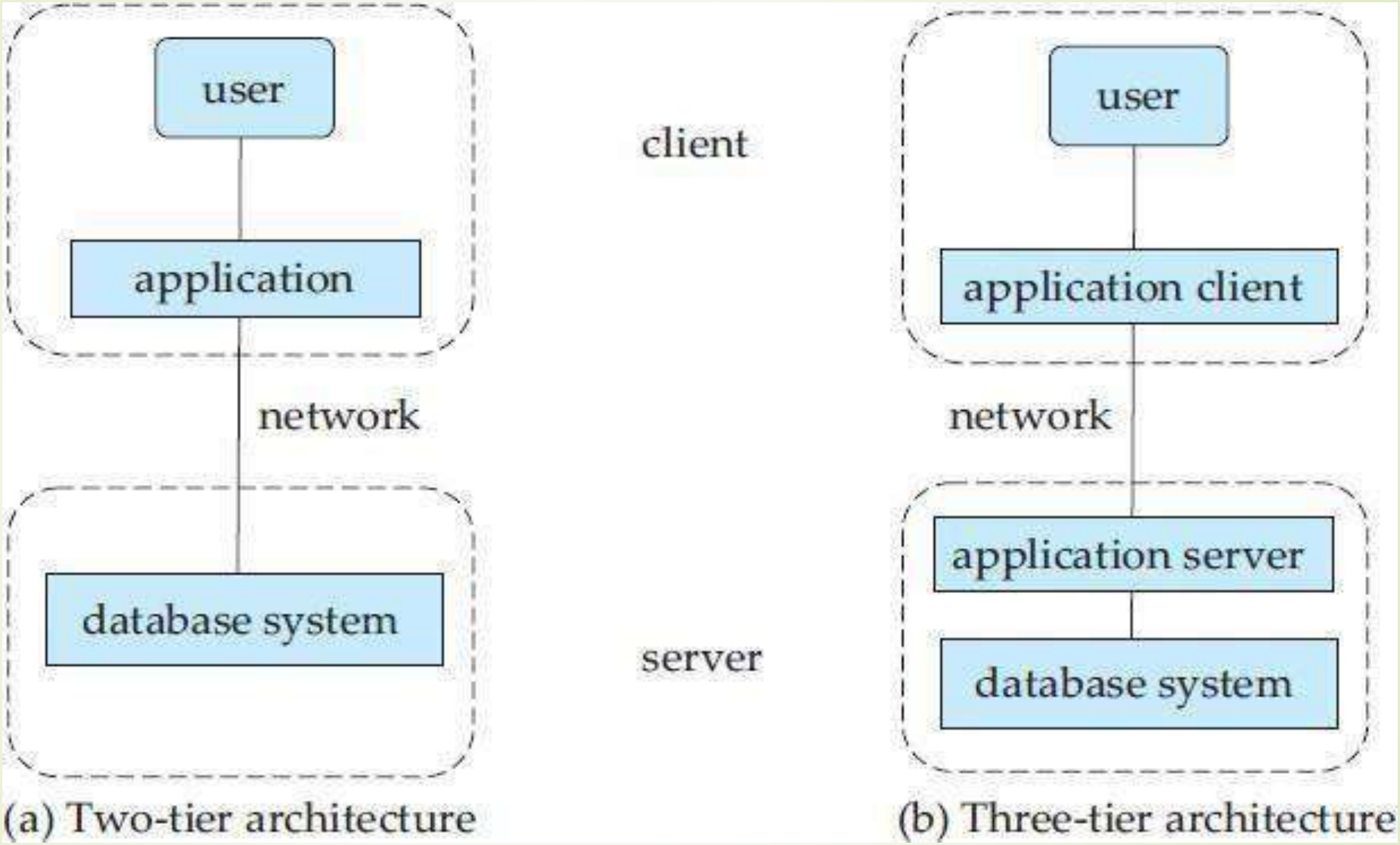
- **Database applications** are usually **partitioned into two or three parts**

Two-tier architecture

- In a **two-tier architecture**, the **application resides at the client machine**, where it invokes database system functionality at the server machine through query language statements.
- Application program interface standards like ODBC and JDBC are used for interaction between the client and the server.

Three-tier architecture

- In a **three-tier architecture**, the **client machine acts as merely a front end** and **does not contain any direct database calls**.
- Instead, the **client** end **communicates with an application server**, usually through a forms interface.
- The **application server in turn communicates with a database system to access data**.
- The **business logic** of the application, which says what actions to carry out under what conditions, **is embedded in the application server**, instead of being distributed across multiple clients.
- Three-tier applications are more appropriate for large applications, and for applications that run on the WorldWideWeb.


user
application
network
database system
client
server
(a) Two-tier architecture
user
application client
network
application server
database system
(b) Three-tier architecture

Query Processor

- The query processor components include
- DDL interpreter, which **interprets DDL statements and records the definitions in the data dictionary**.
- DML compiler, which **translates DML statements** in a query language into an evaluation plan consisting of **low-level instructions** that the query evaluation engine understands.
- A query can usually be translated into **any of a number of alternative evaluation plans** that all give the same result.
- The **DML compiler also performs query optimization**, that is, it **picks the lowest cost evaluation** plan from among the alternatives.
- **Query evaluation engine**, which **executes low-level instructions** generated by the DML compiler.

Storage Manager

- A ***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 for the interaction with the file manager**.
- The raw data are stored on the disk using the file system, which is usually provided by a conventional operating system.
- The **storage manager is responsible for storing, retrieving, and updating data in the database.**

- The storage manager component includes:
- **Authorization and integrity manager**, which tests for the satisfaction of **integrity constraints** and **checks the authority** of users to access data.
- **Transaction manager**, which ensures that the **database remains in a consistent** (correct) **state** despite system failures, and that concurrent transaction executions proceed without conflicting.
- **File manager**, which manages the **allocation of space on disk storage and the data structure**s used to represent information stored on disk.
- **Buffer manager**, which is responsible for **fetching data from disk storage into main memory**, and deciding what data to cache in main memory. The buffer manager is a critical part of the database system, since it **enables the database to handle data sizes that are much larger than the size of main memory.**

Transaction Manager

- A transaction is a **collection of operations that performs a single logical function** in a database application.
- **Each transaction is a unit of both atomicity and consistency**.
- Thus, we require that **transactions do not violate any database-consistency constraints.**
- That is, if the **database was consistent when a transaction started**, the **database must be consistent when the transaction successfully terminates**.
- Transaction - manager **ensures that the database remains in a consistent** (correct) **state despite system failures** (e.g., power failures and operating system crashes) and **transaction failures**.

DATA INDEPENDENCE

- The **ability to modify a schema definition in one level without affecting a schema definition in the next higher level** is called data independence.

  OR

- Data and programs are independent of each other, so change is one has no or minimum effect on the other. Data and its structure are stored in the database where as application programs manipulating this data are stored separately, the change in one does not necessarily affect the other.

- There are of 2 types:
- **Logical data independence**: The **ability to modify the conceptual schema without changing the external schema** or application programs to be rewritten.
- Example: The addition or removal of new entities attributes or relationships to the conceptual schema should be possible without having to change existing external schemas or having to rewrite existing application programs.
- **Physical data independence:** The **ability to modify the internal schema** without having to **change the conceptual schemas**.
- Modifications at this level are usually to improve performance. Changes in the physical schema may include:
  - Using new storage device
  - Using different data structures
  - Switching from one access method to another, modifying indexes etc.