Introductory Concepts of DBMS

Database Management System (DBMS)

- DBMS contains information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is both *convenient* and *efficient* to use
- Database systems are used to manage collections of data that are:
  - Highly valuable
  - Relatively large
  - Accessed by multiple users and applications, often at the same time.
- A modern database system is a complex software system whose task is to manage a large, complex collection of data.
- Databases touch all aspects of our lives

DBMS Applications

- Enterprise Information
  - Sales: customers, products, purchases
  - Accounting: payments, receipts, assets
  - Human Resources: Information about employees, salaries, payroll taxes.
- Manufacturing: management of production, inventory, orders, supply chain.
- Banking and finance
  - customer information, accounts, loans, and banking transactions.
  - Credit card transactions
  - Finance: sales and purchases of financial instruments (e.g., stocks and bonds; storing real-time market data
- Universities: registration, grades

DBMS Applications

- Airlines: reservations, schedules
- Telecommunication: records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards
- Web-based services
  - Online retailers: order tracking, customized recommendations
  - Online advertisements
- Document databases
- Navigation systems: For maintaining the locations of varies places of interest along with the exact routes of roads, train systems, buses, etc.

Purpose of Database Systems

In the early days, database applications were built directly on top of file systems, which leads to:

- Data redundancy and inconsistency: data is stored in multiple file formats resulting induplication of information in different files
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become “buried” in program code rather than being stated explicitly
  - Hard to add new constraints or change existing ones

Purpose of Database Systems

- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - Example: Transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies
  - Ex: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
- Security problems
  - Hard to provide user access to some, but not all, data

Database systems offer solutions to all the above problems

University Database Example

- In this text we will be using a university database to illustrate all the concepts
- Data consists of information about:
  - Students
  - Instructors
  - Classes
- Application program examples:
  - Add new students, instructors, and courses
  - Register students for courses, and generate class rosters
  - Assign grades to students, compute grade point averages (GPA) and generate transcripts

Instances and Schemas

Similar to types and variables in programming languages

Logical Schema – the overall logical structure of the database

Example: The database consists of information about a set of customers and accounts in a bank and the relationship between them

Physical schema– the overall physical structure of the database

Instance – the actual content of the database at a particular point in time. E.g. value of a variable

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 Independence

- Data independece is defined as a property of DBMS that helps you to change the database schema at one level of a database system without requiring to change the schema at next higher level.
- In DBMS there are two types of data independence

1. **Logical Data independence :-**
   - Unable to change the conceptual schema without having to change the external schema.
   - It is used to separate the external level from the conceptual view.
   - It occurs at the user interface level.
2. **Physical Data independence :-**
   - It can be defined as the capacity to change the internal schema without having to change the conceptual schema.
   - If we do any changes in the storage size of the database system server, then the conceptual structure of the database will not be affected.
   - It is used to separate the conceptual levels from the internal levels.
   - It occurs at the logical interface level.

Data Independence

Levels of Database / Data Abstraction

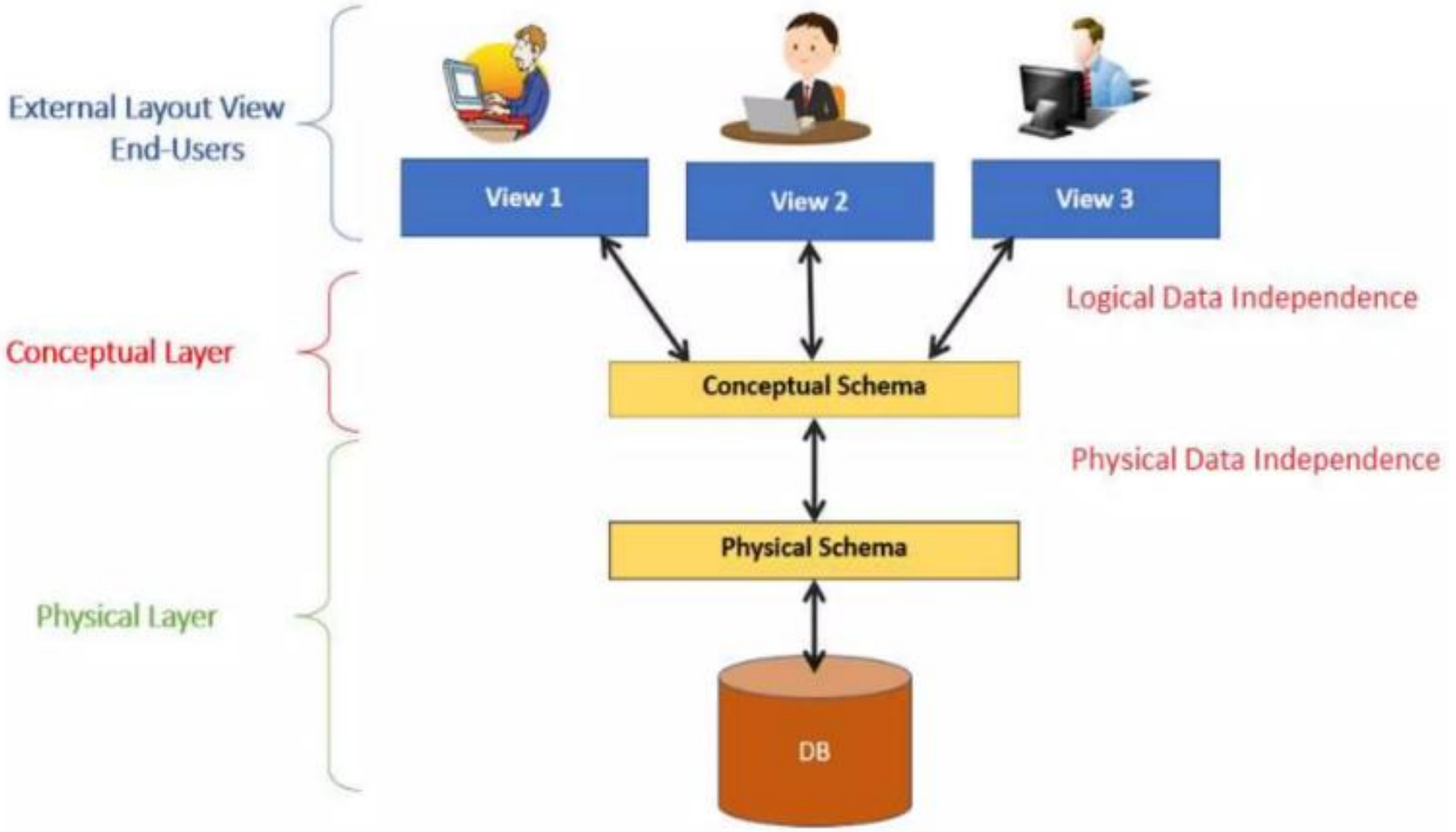
1. **View / External Level:-** Application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

2. **Conceptual / Logical Level:-** Describes data stored in database, and the relationships among the data.

```
type instructor = record
        ID : string;
        name : string;
        dept_name : string;
        salary : integer;
            end;
```

3. **Physical / Internal Level:-** Describes how a record (e.g., instructor) is stored.

View of Data

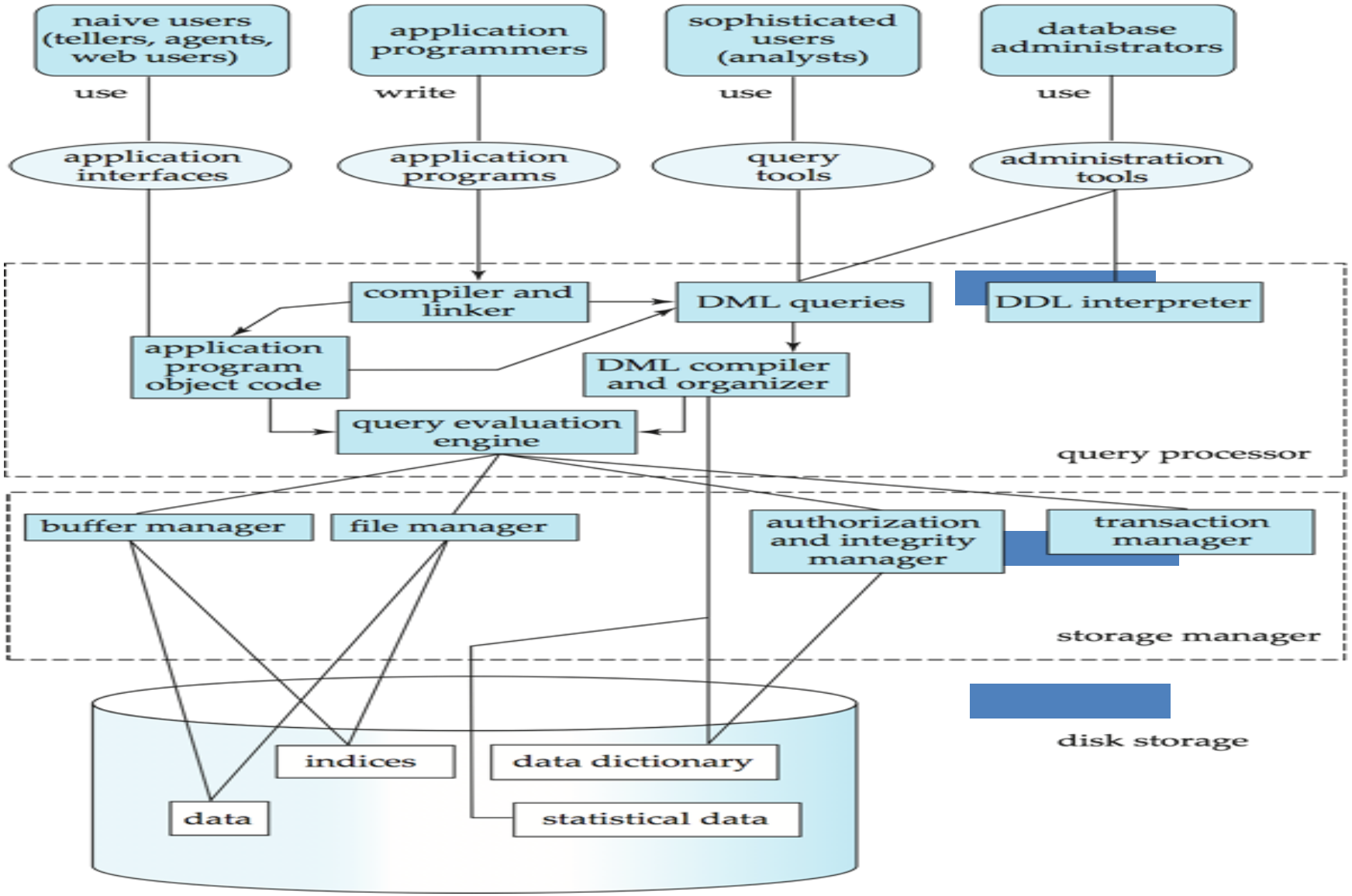

External Layout View
End-Users
View 1
View 2
View 3
Logical Data Independence
Conceptual Layer
Conceptual Schema
Physical Data Independence
Physical Schema
Physical Layer
DB

Database Architecture

The architecture of a database systems is greatly influenced by the underlying computer system on which the database is running:

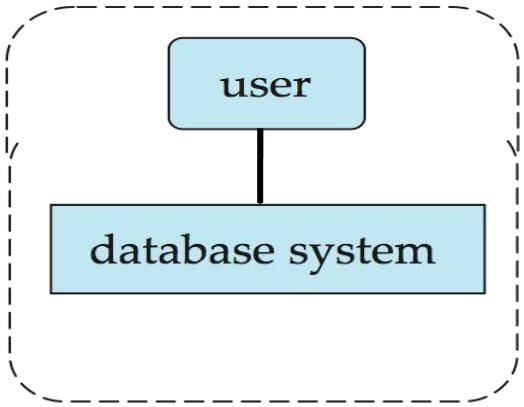
- Centralized
  - All the DBMS functionality, application program execution and user interface processing of dumb terminals were carried out on a single centralized machine like mainframe machine
- Client-server
  - A client is typically a user machine not dumb terminal that provides user interface capabilities and local processing
  - A server is a system containing both hardware and software that can provide services to the client machines
- Parallel (multi-processor)
- Distributed

Database System Internals



Client Server Architecture

- Single tier or Multi tier
- n-tier architecture divides the whole system into related but independent n modules, which can be independently modified, altered, changed or replaced
- 1-tier architecture
  - DBMS is the only entity where user directly sits on DBMS and uses it
  - Any changes done here will directly be done on DBMS itself
  - Does not provide handy tools for end users and preferably database designer and programmers use single tier architecture

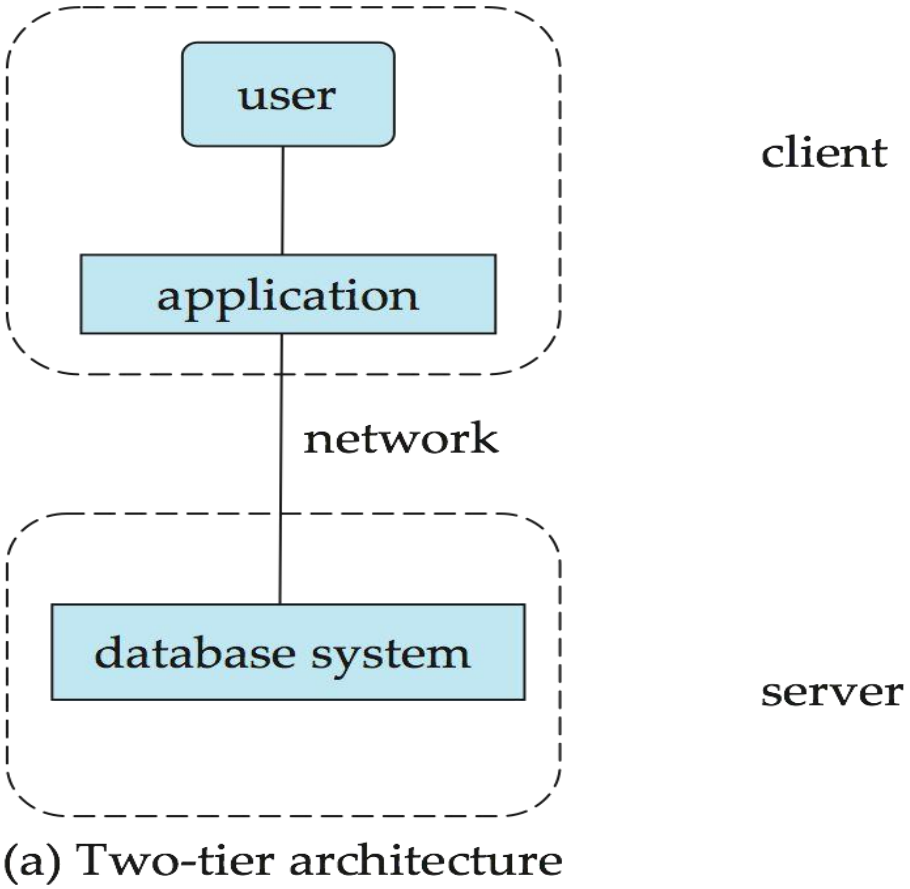


User with editor to access the database

Only database along with its query processing, all relations and their constraints

Client Server Architecture

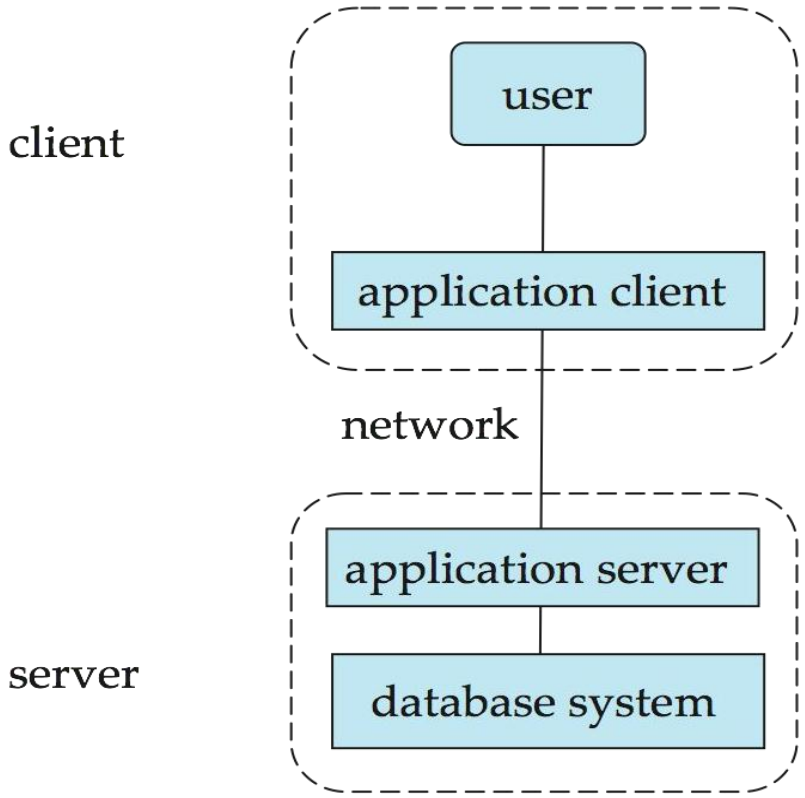
- Two-tier architecture
  - Must have some application, which uses the DBMS
  - Programmers use 2-tier architecture where they access DBMS by means of application
  - Application tier is entirely independent of database in term of operation, design and programming



(a) Two-tier architecture

DBMS Application Architecture

- Three-tier Architecture
  - Most widely used architecture
  - Separates it tier from each other on basis of users



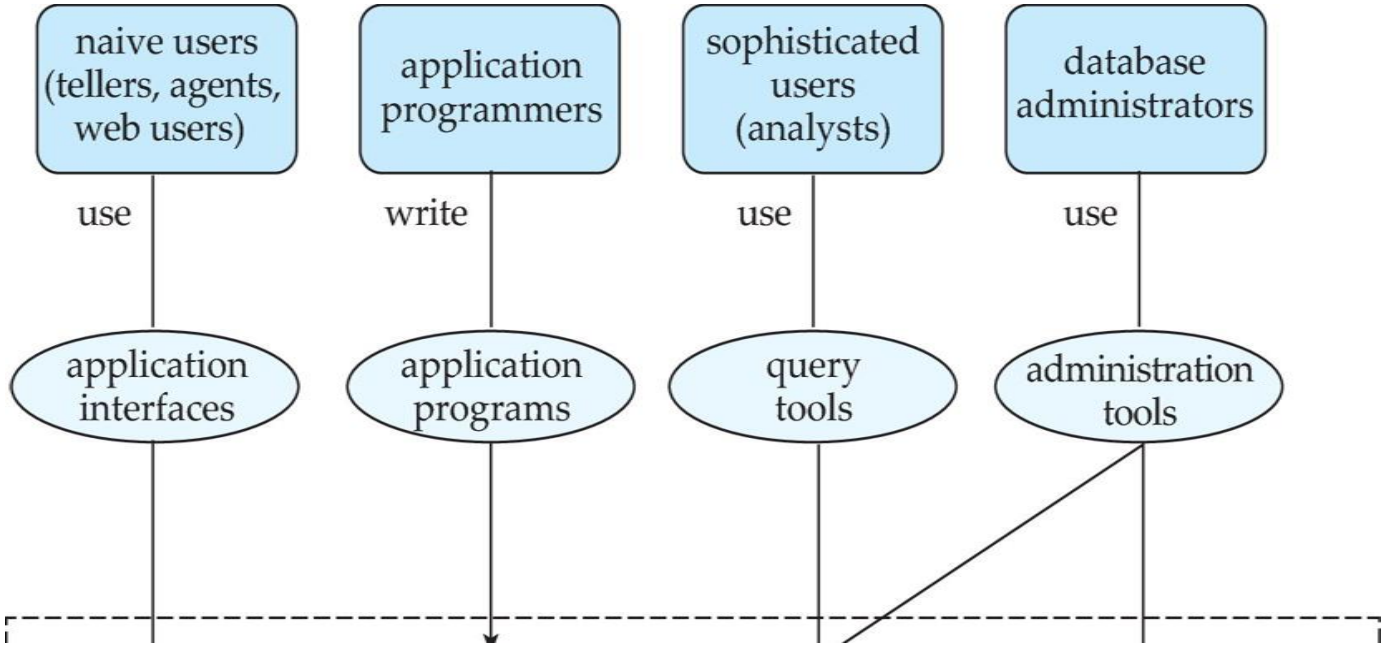
(b) Three-tier architecture

This tier is everything for user. He/she doesn't know about any existence/form of database beyond layer. Multiple views of database can be provided by application

Application server and program which access database. For a user this application tier works as abstracted view of database

Only database along with its query processing, all relations and their constraints

Database Users and Administrators



Database Administrator - Duties

- Schema Definition
  - Creates the original database schema using DDL statements
- Storage structure and access method definition
- Schema and physical organization modification
- Granting of authorization for data access
  - Which parts of the database various users can access
  - This information is consulted for granting information whenever someone attempts to access data in the system
- Routine maintenance
  - Periodically backup of database onto tapes, remote servers
    - To prevent data loss in case of disasters such as flooding
  - Ensures enough free disk space is available for normal operations and upgrading disk space as required
  - Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users