

Database Environment & Architecture

INT191 Relational Database Concepts and Design

Dr. Sunisa Sathapornvajana

What we will learn

- * The ANSI-SPARC Architecture
- * Database Design Concept
- * Functions of a DBMS
- * Multi-user DBMS Architectures
- * Components of a DBMS

ANSI-SPARC

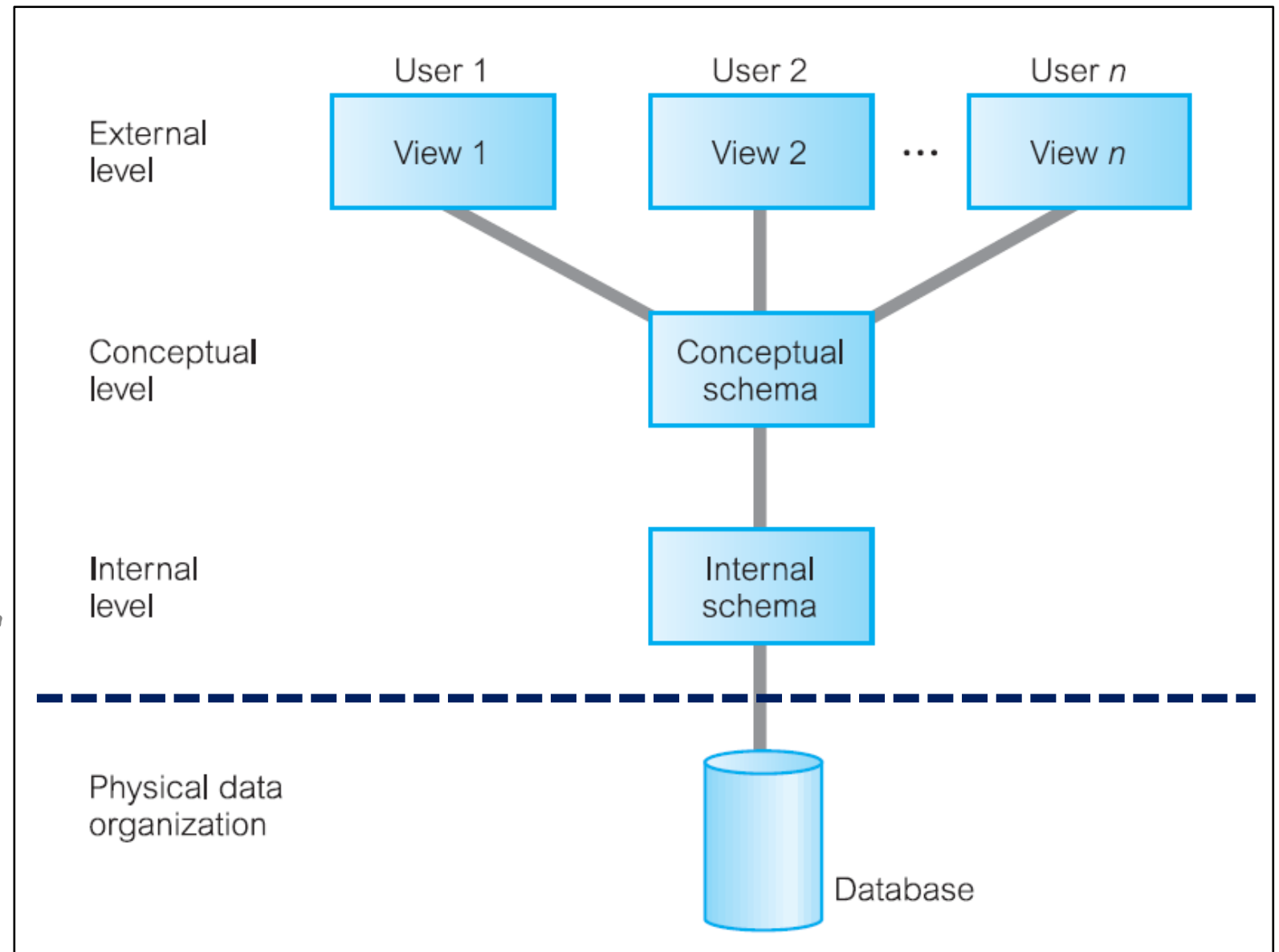
- The ANSI-SPARC Architecture is an abstract design standard for a Database Management System (DBMS).
- **ANSI** - American National Standards Institute
- **SPARC** - Standards Planning And Requirements Committee
- The ANSI-SPARC model, however, never became a formal standard.
- It provides a basis for understanding some of the functionality of a DBMS.

Objective of ANSI-SPARC

- * Users access the same data but have a different customized view.
- * Users can change the way they view the data with no effect to other users.
- * Users should not need to know physical database storage details.
- * DBAs can change conceptual structure of database or storage structures without affecting all users.
- * Internal structure of database should be unaffected by changes to physical aspects of storage.

ANSI-SPARC Architecture (Three-level architecture)

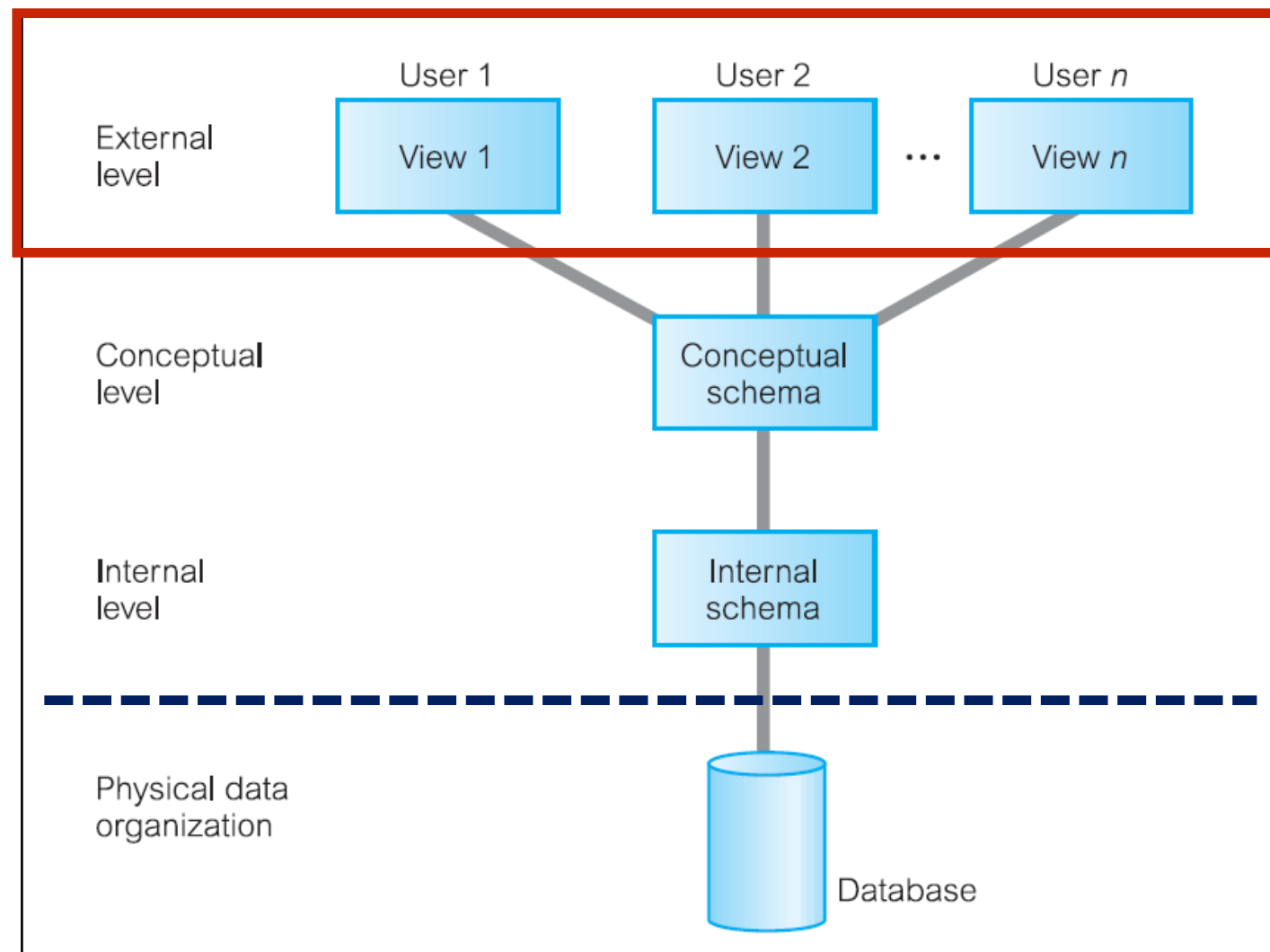
- * The ANSI-SPARC model of a database identifies **three distinct levels** at which data items can be described.
 - * An external level.
 - * A conceptual level.
 - * An internal level.



External Level

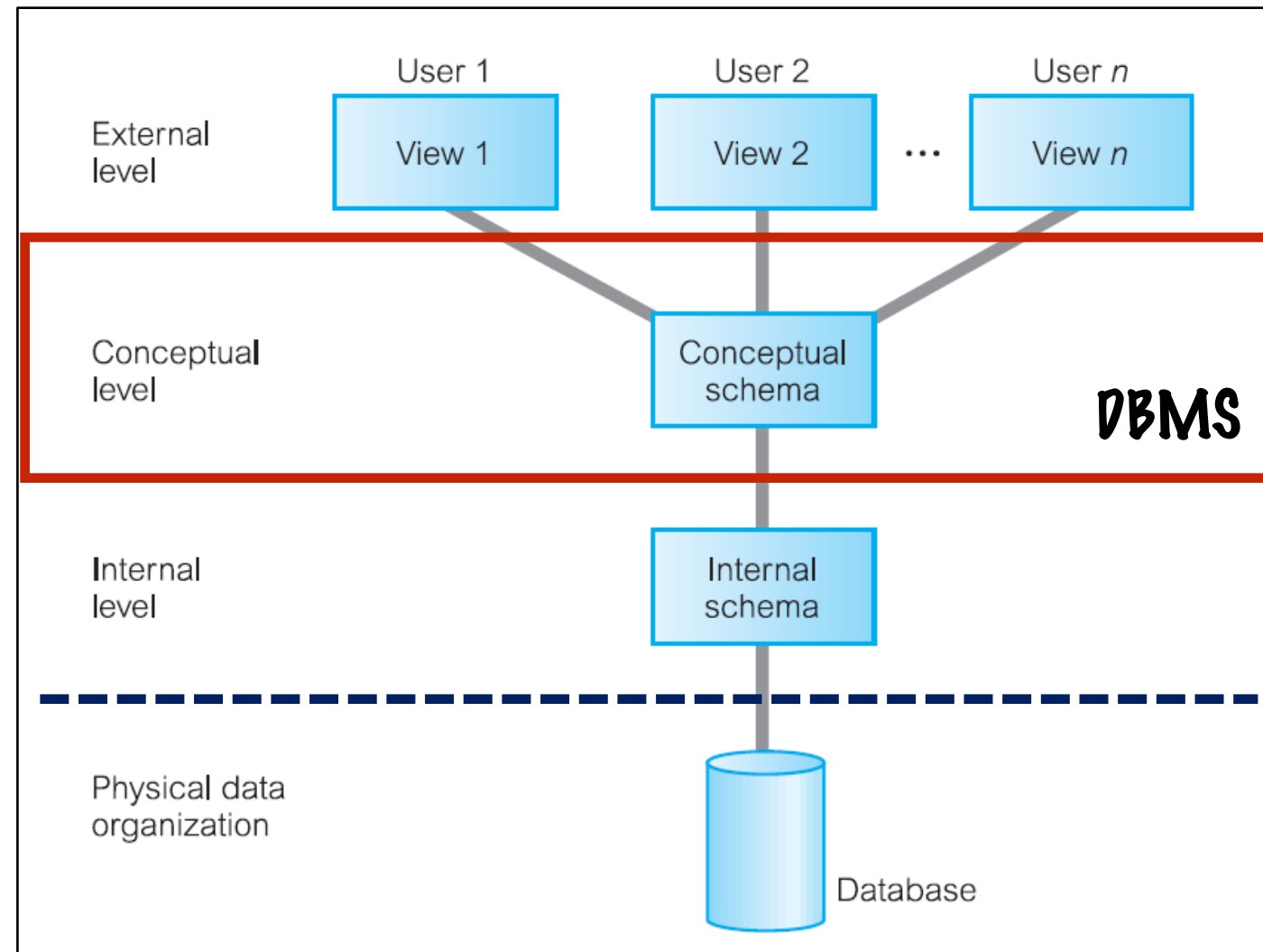
End Users/Application Programs

- * Each **group of users** has a separate external view
- * The external view **includes only data that the user is interested in.**
- * Other data that are not of interest are in the database but the user will not aware of them
- * e.g. forms, reports, applications, views



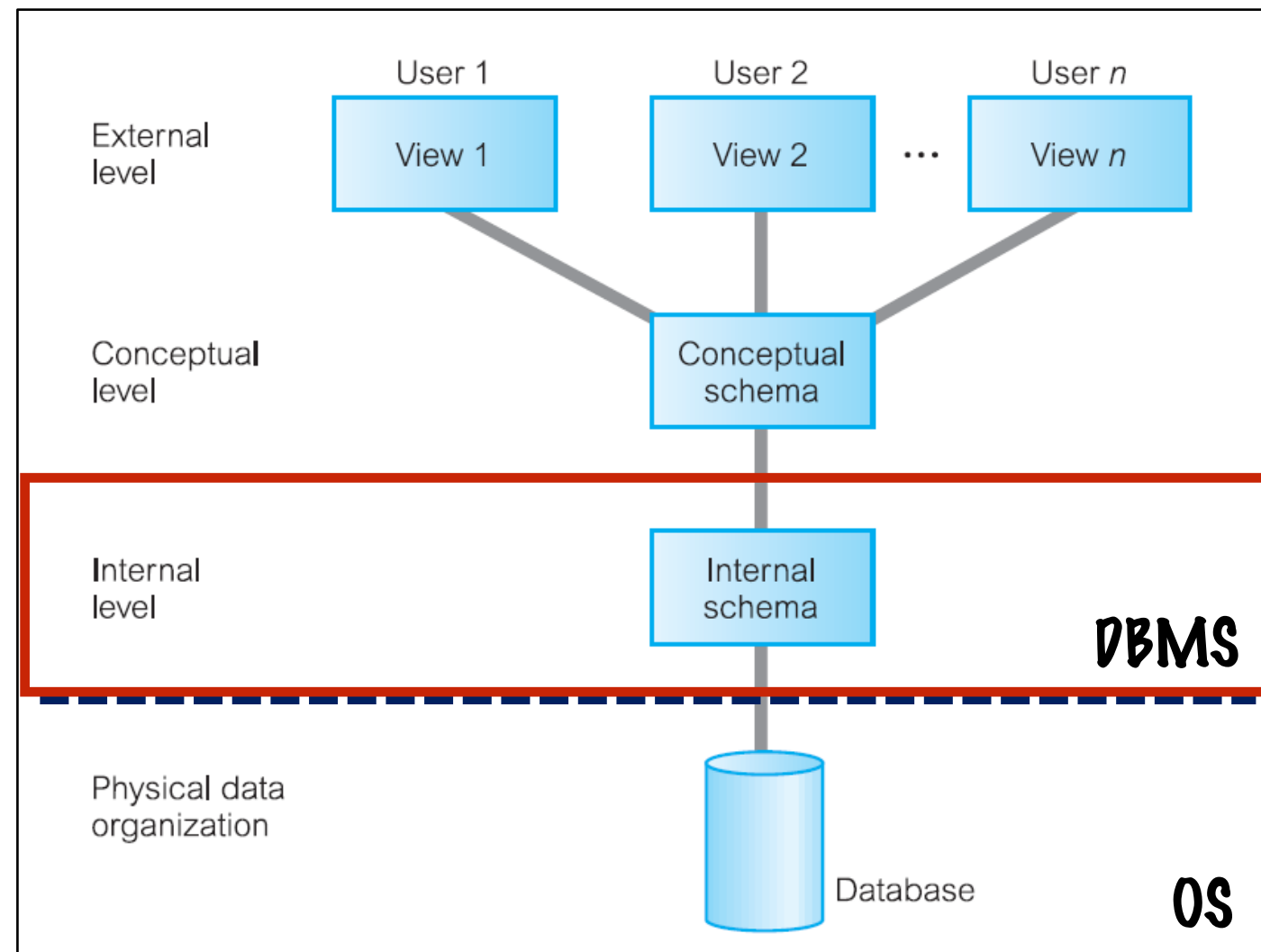
Conceptual Level

- * It contains **the logical structure of the entire database.**
- * It represent what data (entity) is stored in the database and the relationships among the data
- * It is **a complete view of data requirements of the organization**
- * e.g. customer table, product table, and relationships

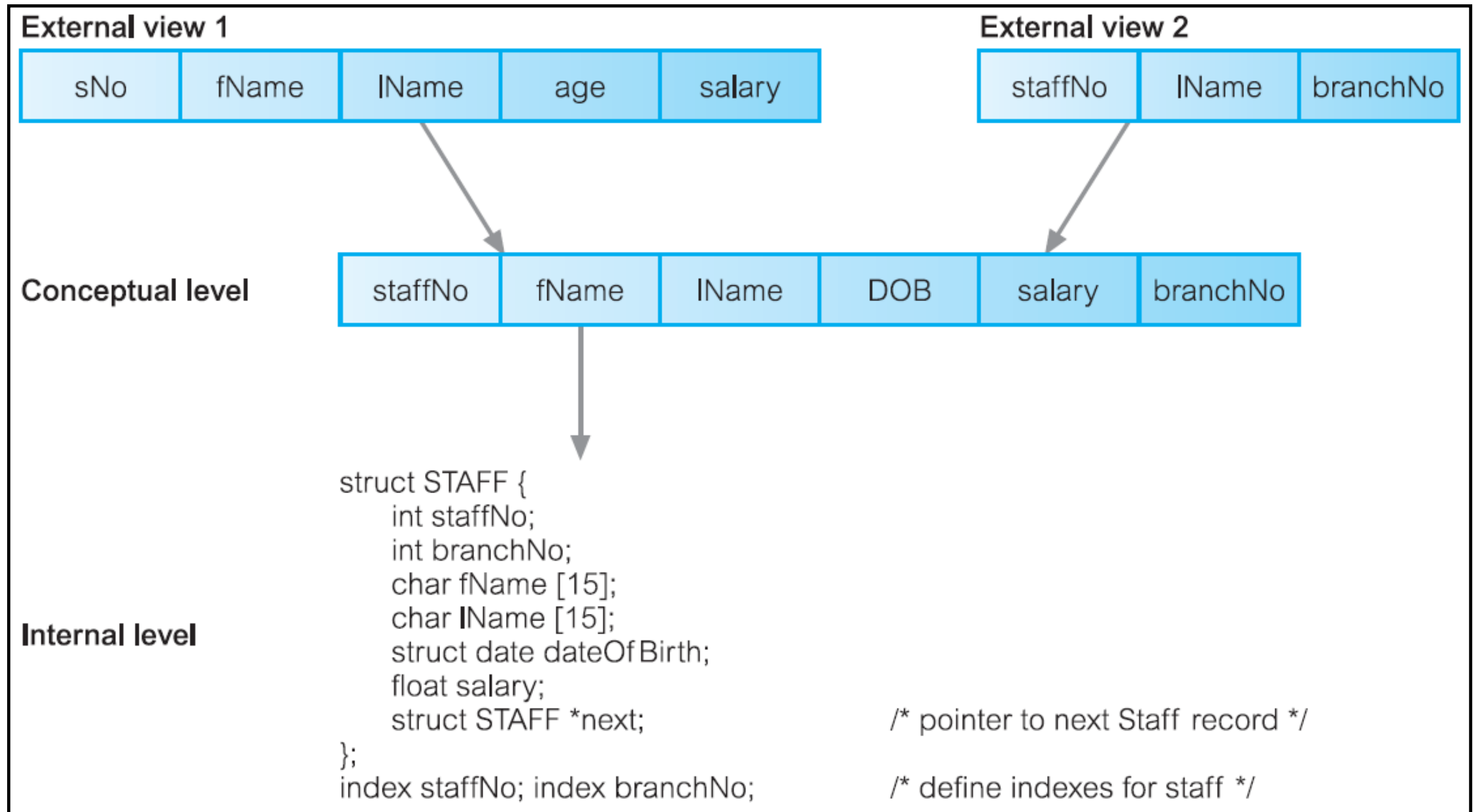


Internal Level

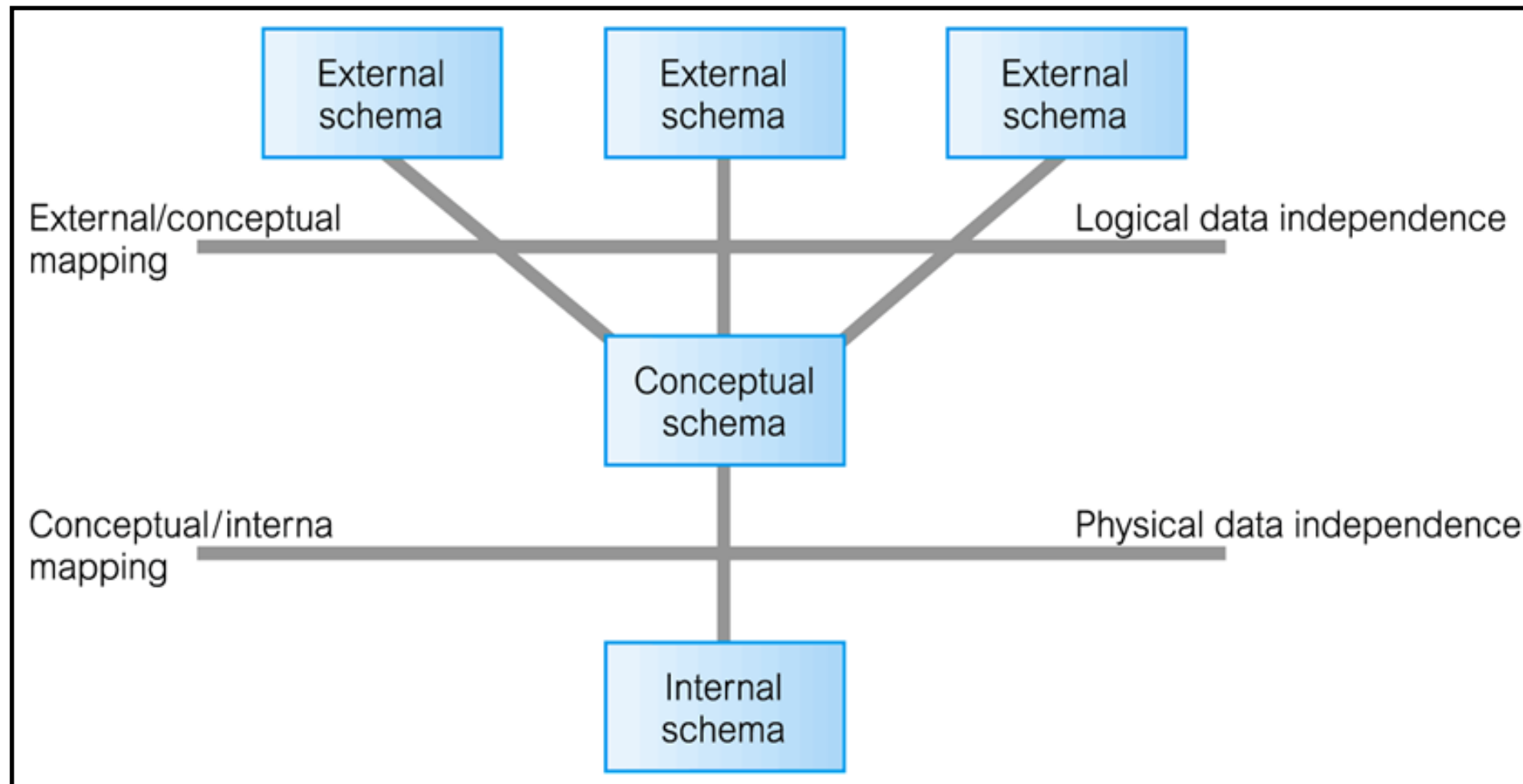
- * It is **the physical structures of the database** on the computer
- * It describes how the data is stored in the database
- * It covers **the physical implementation of the database:**
- * To achieve optimal runtime performance and storage space utilization.
- * e.g. database files



Mapping between levels



Data Independence



“Schema in upper levels are unaffected by changes to schema in lower levels”

Home Design



Source: <http://www.azhomeplan.com/home-layouts-plans>

Home layout plan



Source: www.download3dhouse.com

Home model

Database Design

Requirement Collection and Analysis



1. Conceptual database design (Conceptual data model)

Entity Relationship Diagram

Independent of implementation details
(applications, programming languages, target DBMS)



2. Logical database design (Logical data model)

Tables and Constraints

Dependent of implementation details
(applications, programming languages, target DBMS)



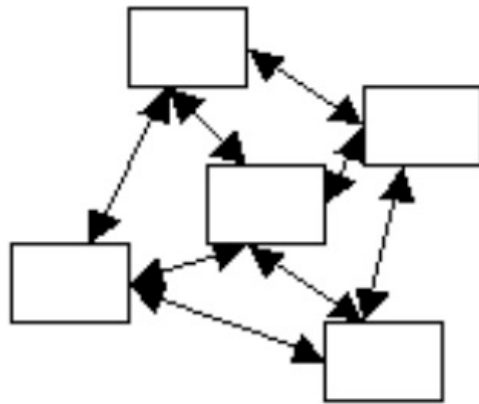
3. Physical database design (Physical data model)

Describe how data is stored in order to access
data efficiently

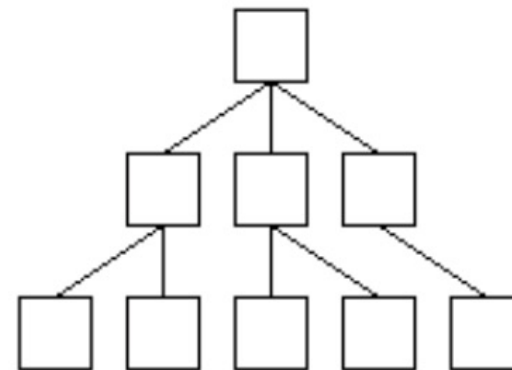


Data files, Indexes, Partitions, Data block sizes

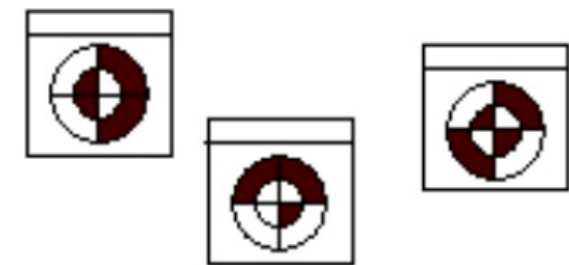
Data Models



Network model







Hierarchical model



Object-oriented model

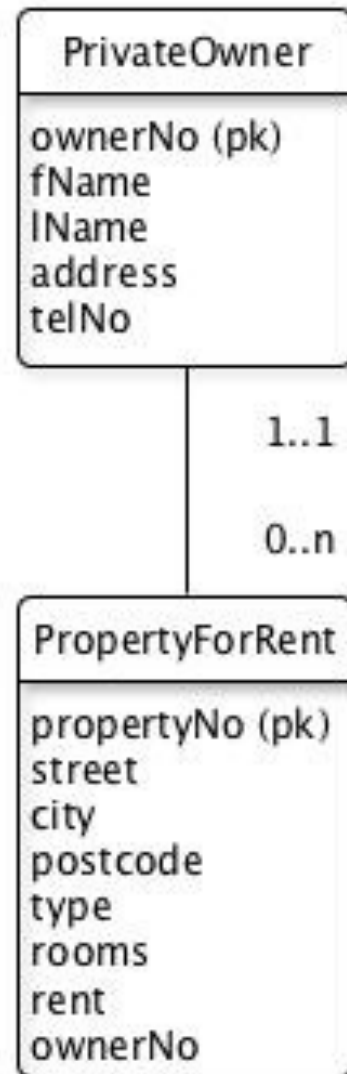
P-ID	Name	Prenome	City
1	Müller	Hans	Oberdorf
2	Meier	Jakob	Hinterwil
3	Keiser	Josef	Unterdorf
...

Relational model

ID	XY	DF	ER
56		XXX	
45		YYY	
...

Object-relational model

Data Model



Entity-Relationship Diagram

PrivateOwner

ownerNo	fName	lName	address	telNo
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93

Real Data

Data Model

“A collection of concepts and rules for the description of the structure of the database”

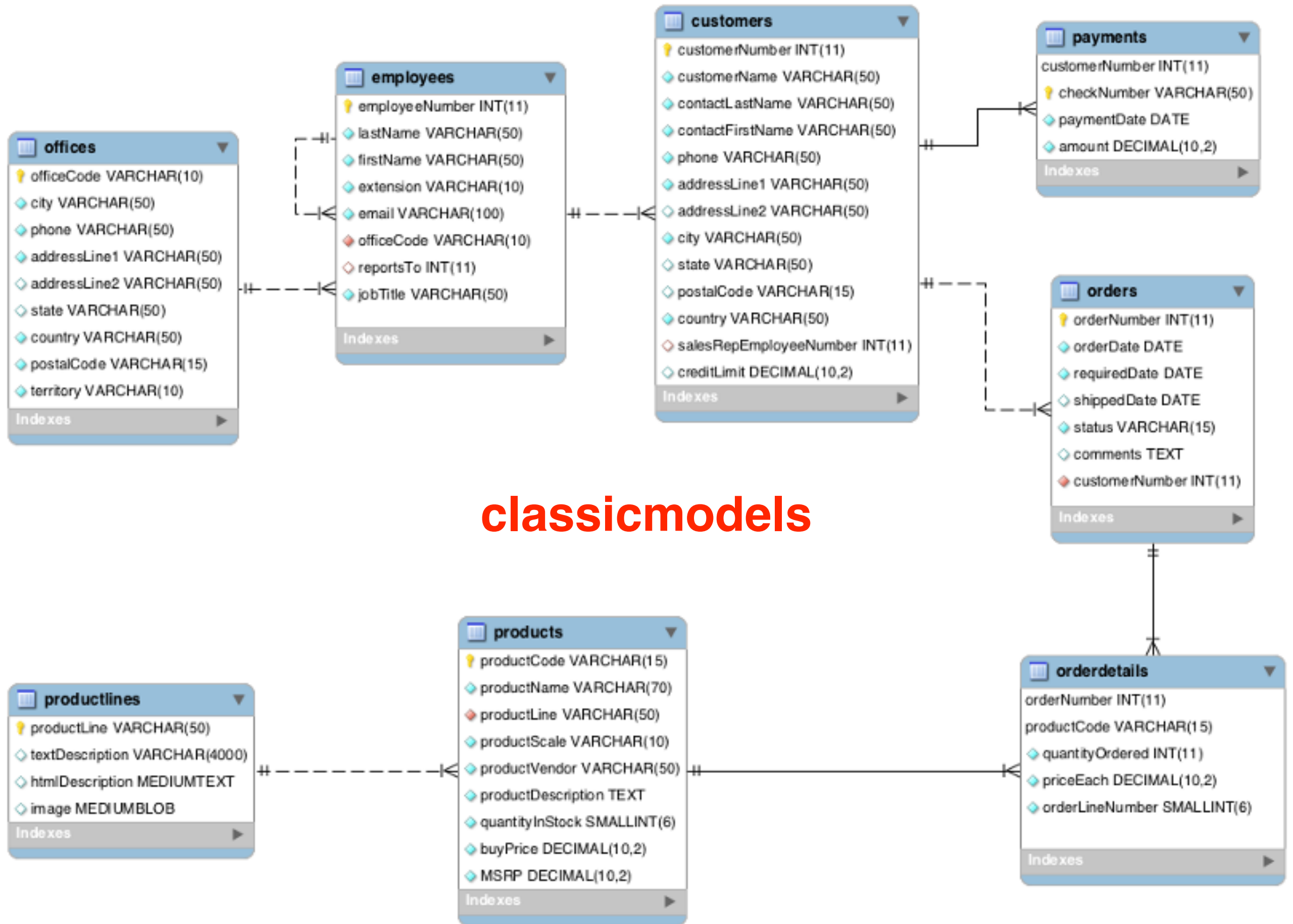
Structural part

Manipulative part

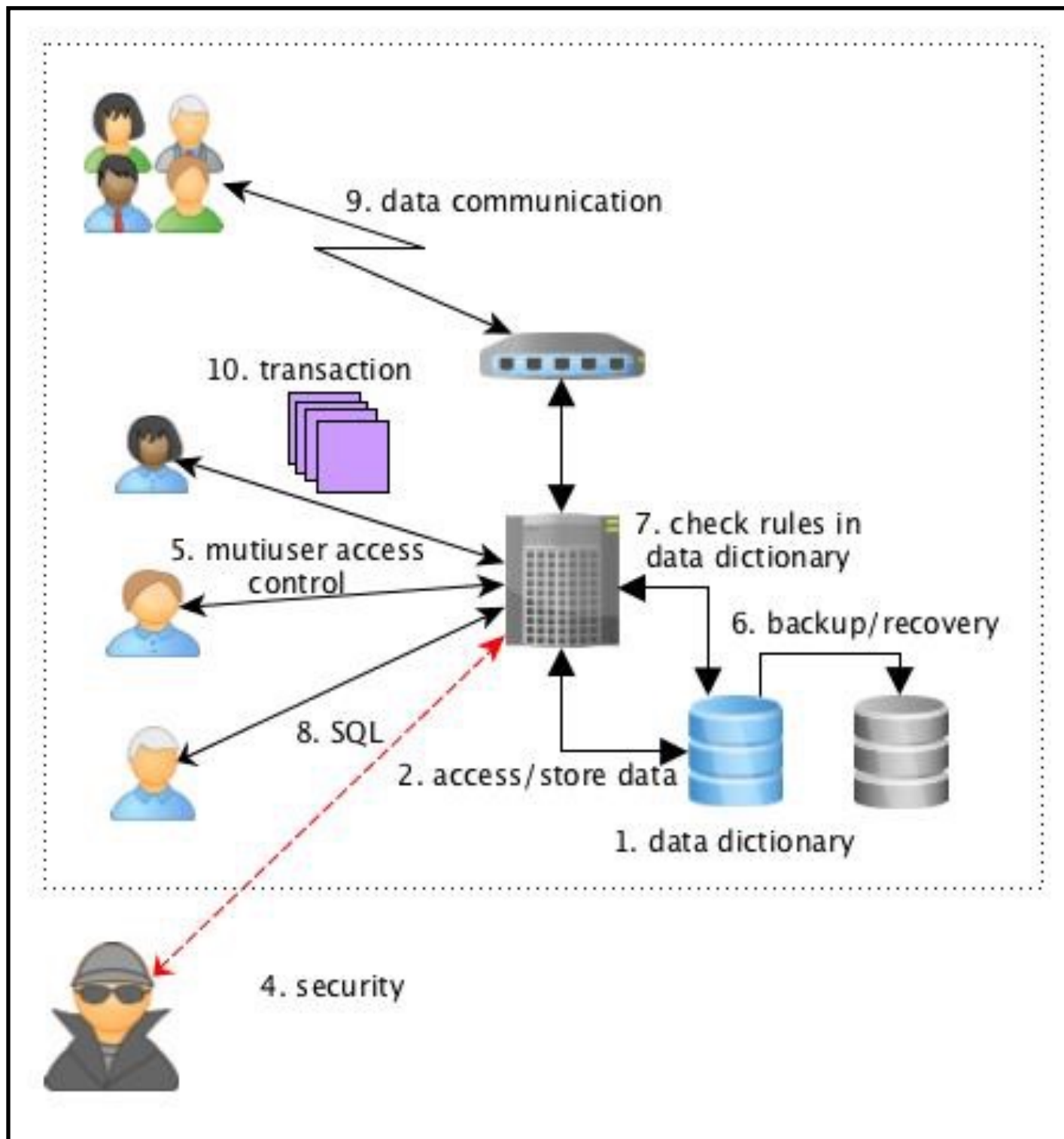
PropertyForRent	
propertyNo (pk)	Number(6)
street	Varchar2(50)
city (nn)	Varchar2(30)
postcode	Varchar2(6)
type	Varchar2(6)
rooms	Number(3)
rent	Number(8,2)
ownerNo (fk)	Number(6)

Integrity constraint part

classicmodels



Functions of a DBMS

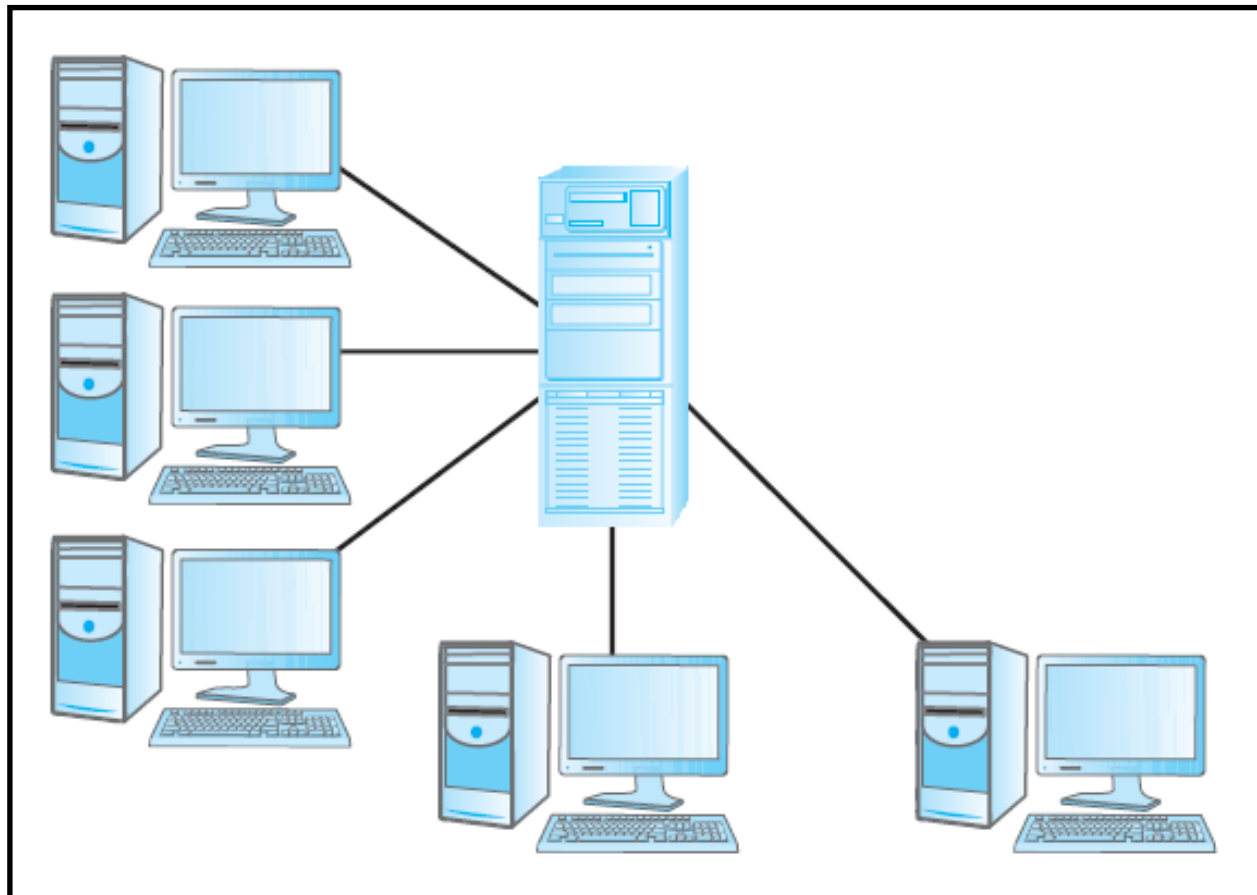


1. Data Dictionary Management
2. Data Storage Management
3. Data Transformation and Presentation
4. Security Management (Authorization)
5. Multiuser Access Control (Concurrency control)
6. Backup and Recovery Management
7. Data Integrity Management
8. Database Access Language and Application
9. Database Communication Interfaces
10. Transaction Management

Multi-user DBMS Architectures

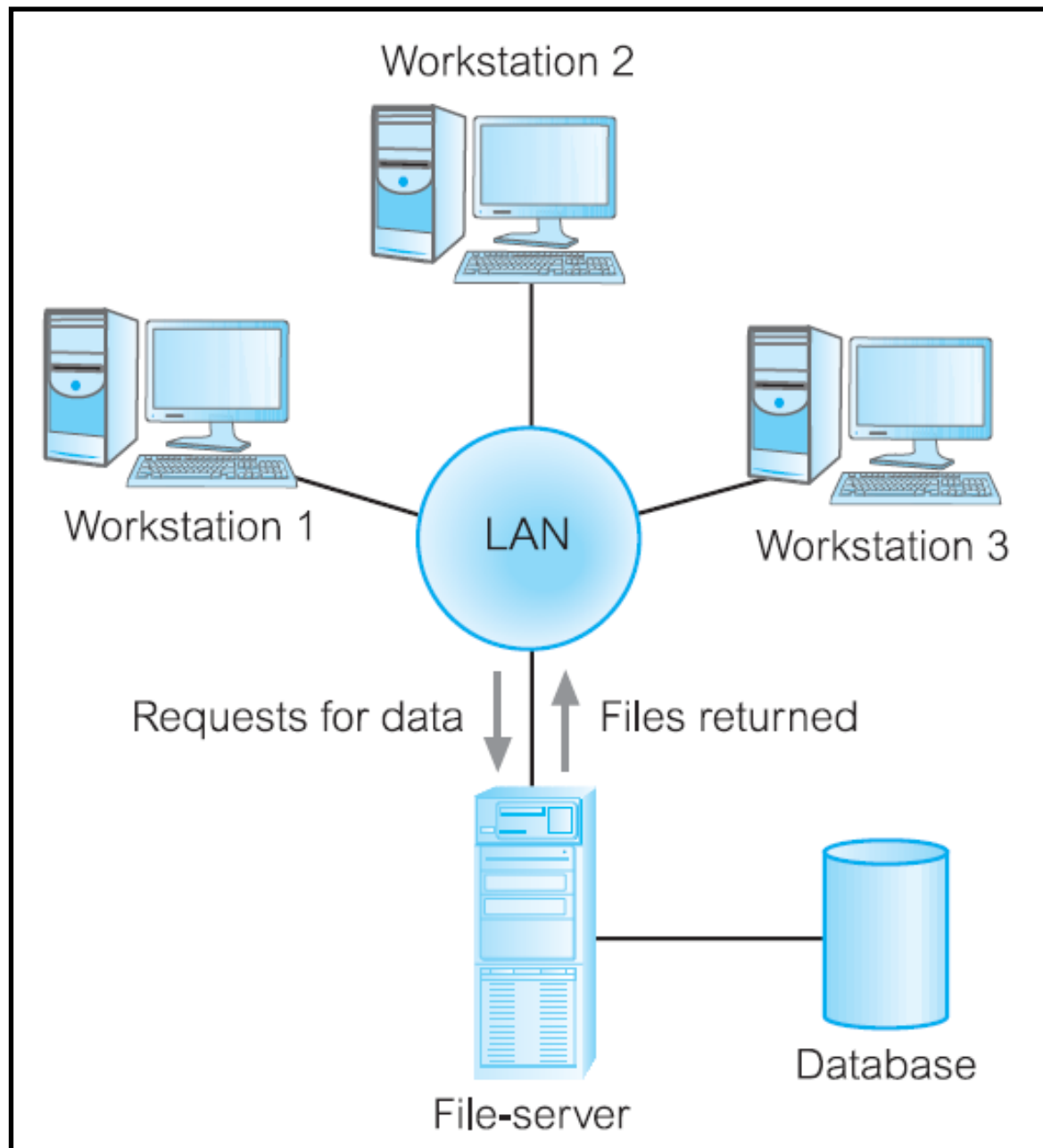
- *The common architectures that are used to implement multi-user database management systems:
 - *Teleprocessing
 - *File-Server Architecture
 - *Two-Tier Client-Server Architecture
 - *Three-Tier Client-Server Architecture

Teleprocessing



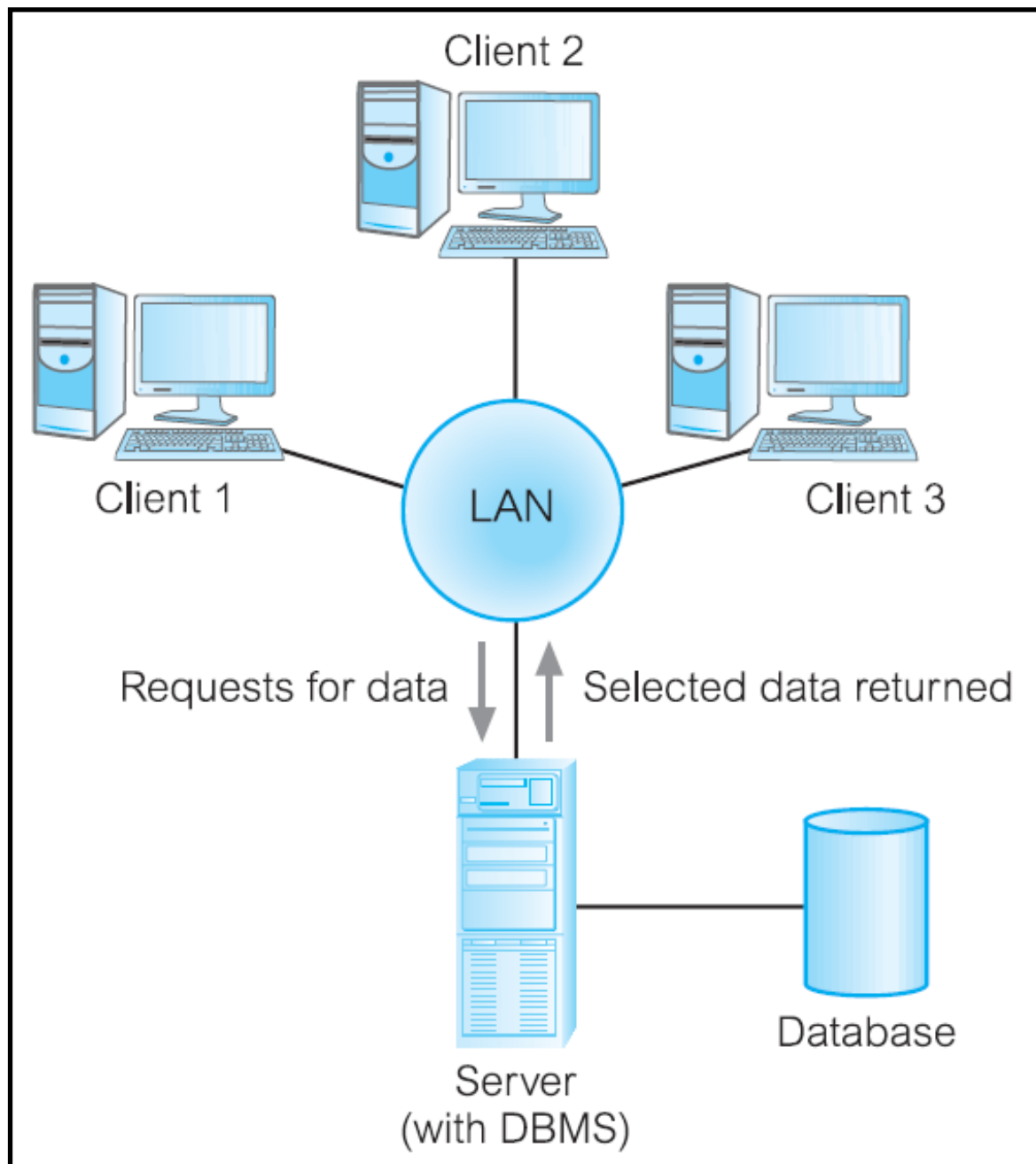
- * The traditional architecture with a **central high-performance computer** and several terminals.
- * User terminals are typically “**dumb**” ones, incapable of functioning on their own.
- * The terminals send messages via the communication channel.
- * This architecture placed a tremendous burden on the central computer.

File-Server



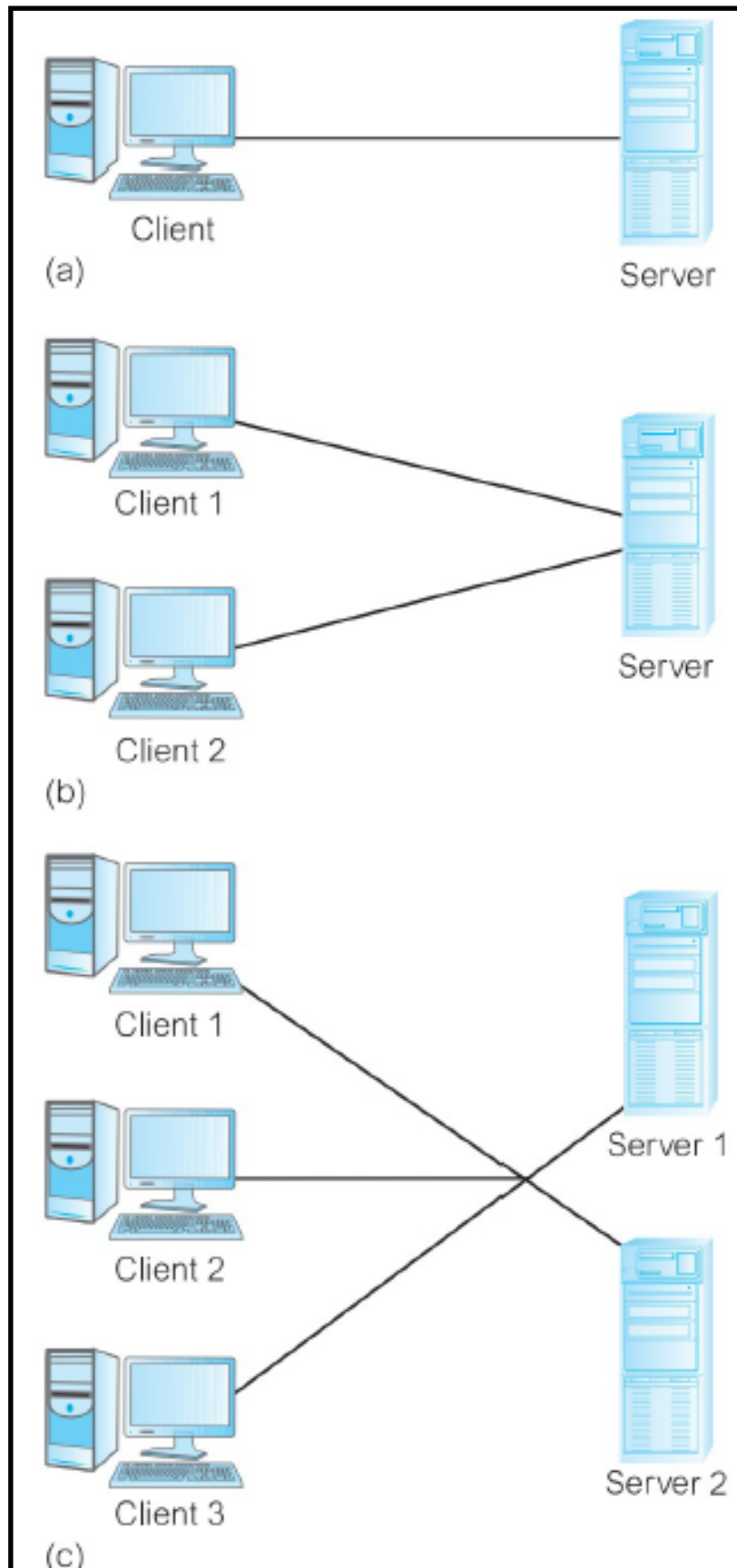
- * A computer attaches to a network and provides shared storage for computer files.
- * The applications and the DBMS are distributed on each workstation in the network.
- * **The file-server handles the file requests only.**
- * Disadvantages
 - * A full copy of the DBMS is required on each workstation.
 - * There is a large amount of network traffic.
 - * Concurrency, recovery, and integrity control are more complex, because there can be multiple DBMSs accessing the same files.

Two-Tier Client-Server



- * The way that software components interact to form **a system consisting of a client process requesting a resource, and a server providing the resource.**
- * The client and server reside on the different machines.
 - * **The client** (tier 1) is primarily responsible for the presentation of data to the user.
 - * **The server** (tier 2) is primarily responsible for supplying data services to the client,

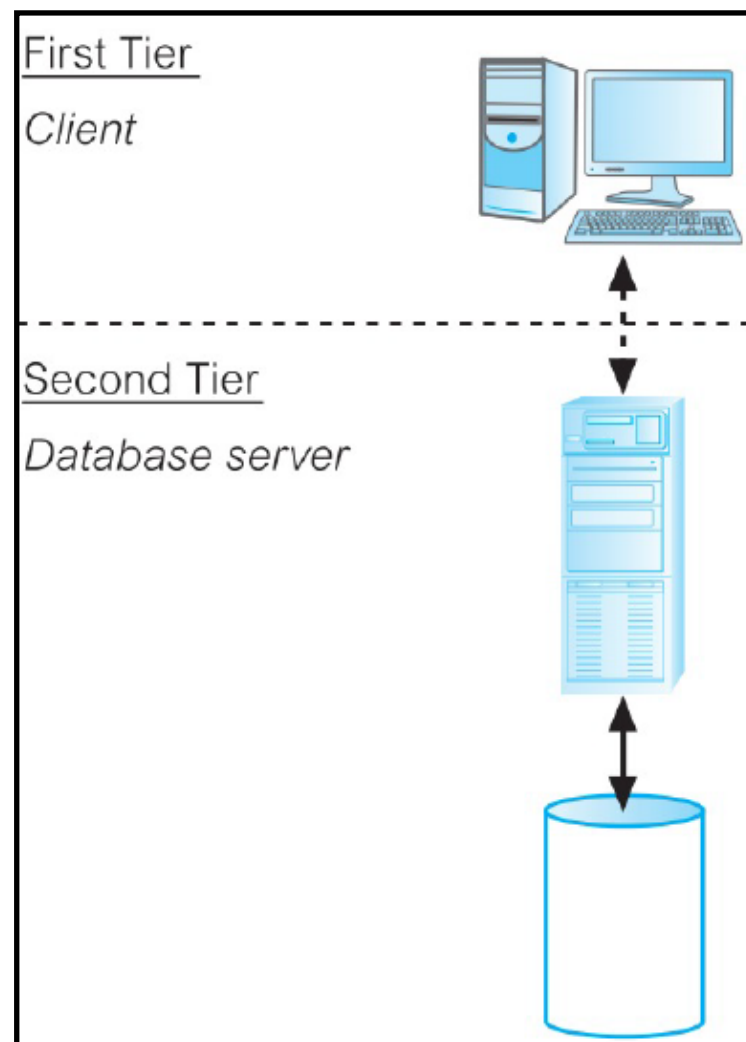
Two-Tier Client-Server



***The possible combinations of the client-server topology.**

- * Single client, single server**
- * Multiple clients, single server**
- * Multiple clients, multiple servers**

Two-Tier Client-Server



Client	Server
Manages the user interface	Accepts and processes database requests from clients
Accepts and checks syntax of user input	Checks authorization
Processes application logic	Ensures integrity constraints not violated
Generates database requests and transmits to server	Performs query/update processing and transmits response to client
Passes response back to user	Maintains system catalog Provides concurrent database access Provides recovery control

Two-Tier Client-Server

*Advantages

- * Increased performance: The clients and server reside on different computers; thus different CPUs can be processing applications in parallel.
- * Hardware costs are reduced: It is only the server that requires storage and processing power sufficient to store and manage the database.
- * Communication costs are reduced: Applications carry out part of the operations on the client and send only requests for database access across the network, resulting in less data being sent across the network.
- * Increased consistency: The server can handle integrity checks, so that constraints need be defined and validated only in the one place, rather than having each application program perform its own checking.

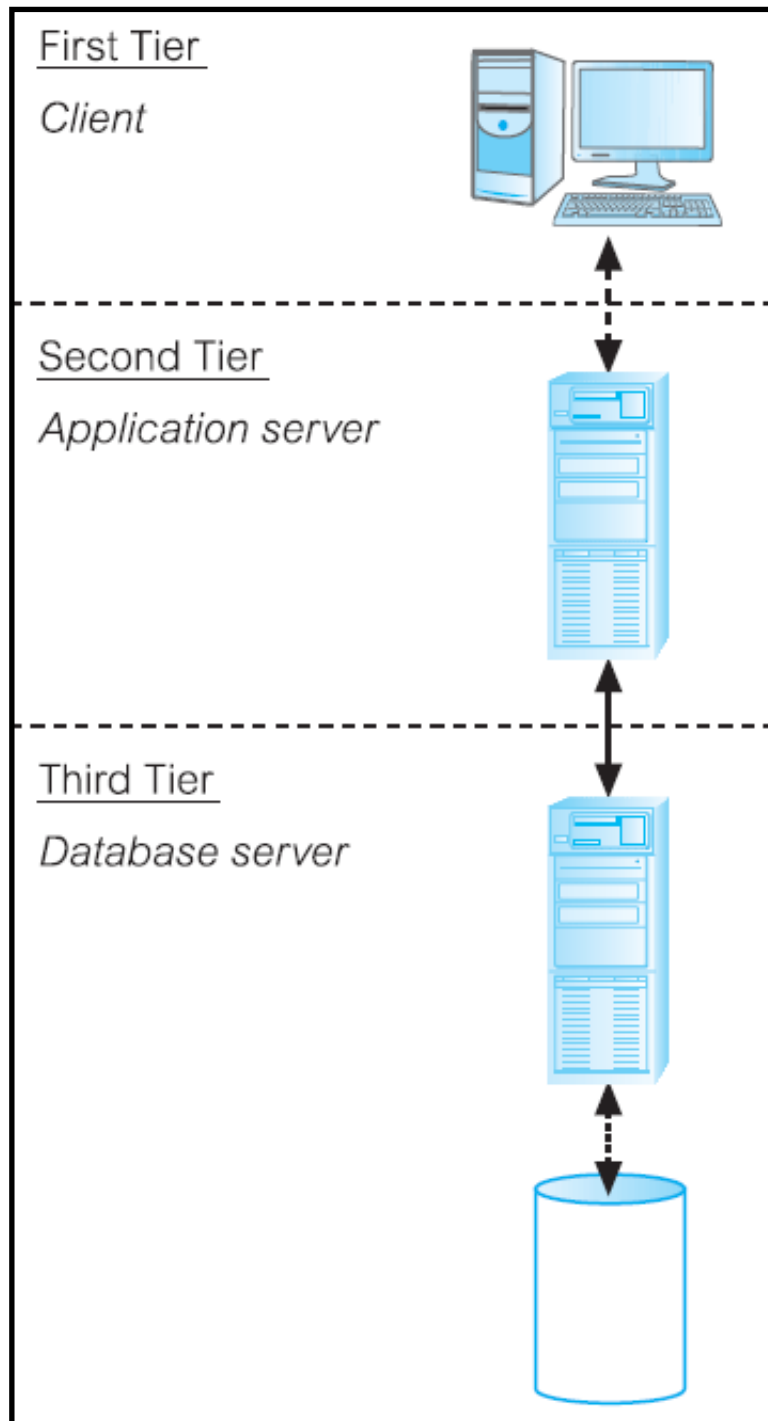
*Disadvantages

- * Fat client compares to teleprocessing; thus it needs more resources.
- * Significant client-side administration overhead.

Comparison

Issue	Teleprocessing	File-server	Two-tier Client-Server
Processing	Mainframe	Workstation	Client/Server
Application Location			
Data Location			
Network traffic	Low	High	Medium
Equipment Cost	High	High on workstation	High on Server
Performance/ Cost	Low	Medium	High

Three-Tier Client-Server



*First tier

- * It runs on the end-user's computer (thin client).
- * It is an application's user interface.
- * It performs some simple logic processing, e.g. input validation.

*Second tier

- * It runs on a server
- * It performs business logic and data processing.
- * One server is designed to serve multiple clients.

*Third tier

- * This tier run on a separate server called the database server.
- * A DBMS stores the data.
- * Transfer data over a LAN or wide area network (WAN).

Three-Tier Client-Server

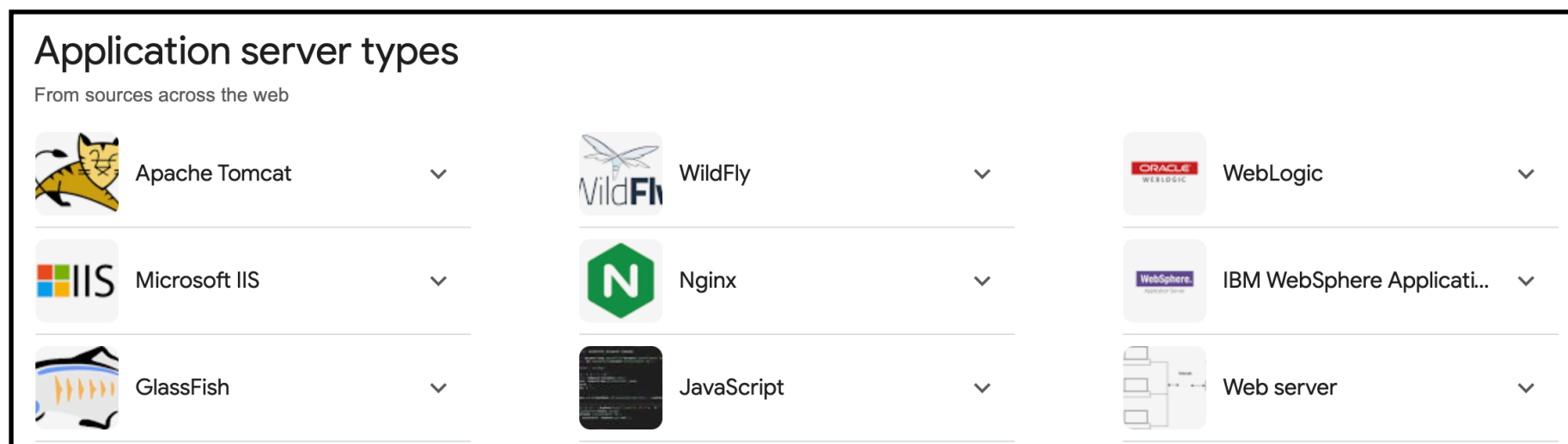
- *The need for less expensive hardware because **the client is thin**.
- ***Application maintenance is centralized** with the transfer of the business logic for many end-users into a single application server.
- *This eliminates the concerns of software distribution that are problematic in the traditional two-tier client-server model.
- *The added modularity makes it easier to modify or replace one tier without affecting the other tiers.
- *Load balancing is easier with the separation of the core business logic from the database functions.

Comparison

Issue	two-tier	Three-tier
Cost of client hardware	High	Low
Spec of client hardware		
Client application maintenance		
Modularity of each tier	Low	High
Load balance	Low	High

Application Server

- * Hosts an application programming interface (API) to expose business logic and business processes for use by other applications.
- * A software framework that provides both facilities to create web applications and a server environment to run them
- * **Example of application servers** : Java Platform, Enterprise Edition (J2EE), WebLogic, OC4J (Oracle), JBoss (Red Hat), WebSphere (IBM), Glassfish (Open Source), .NET Framework (Microsoft)



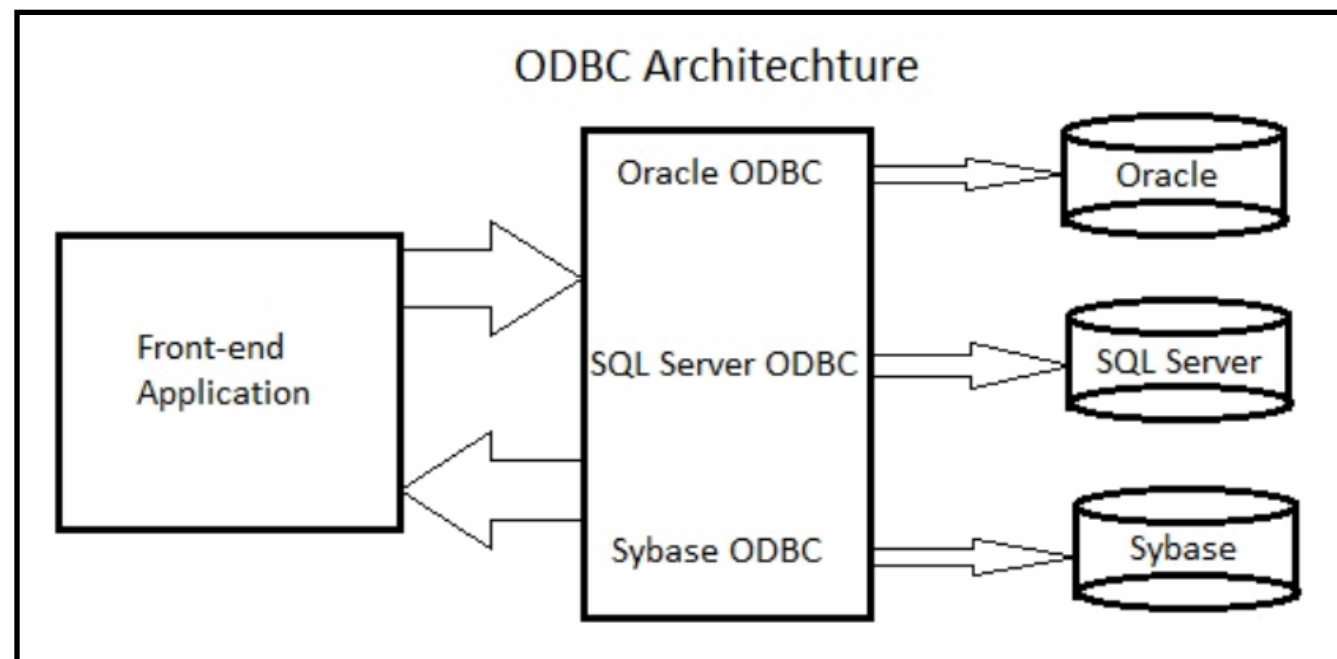
Application Server

- * An application server must handle a number of complex issues:
 - * concurrency;
 - * network connection management;
 - * providing access to all the database servers;
 - * database connection pooling;
 - * legacy database support;
 - * clustering support;
 - * load balancing;
 - * failover.

Middleware

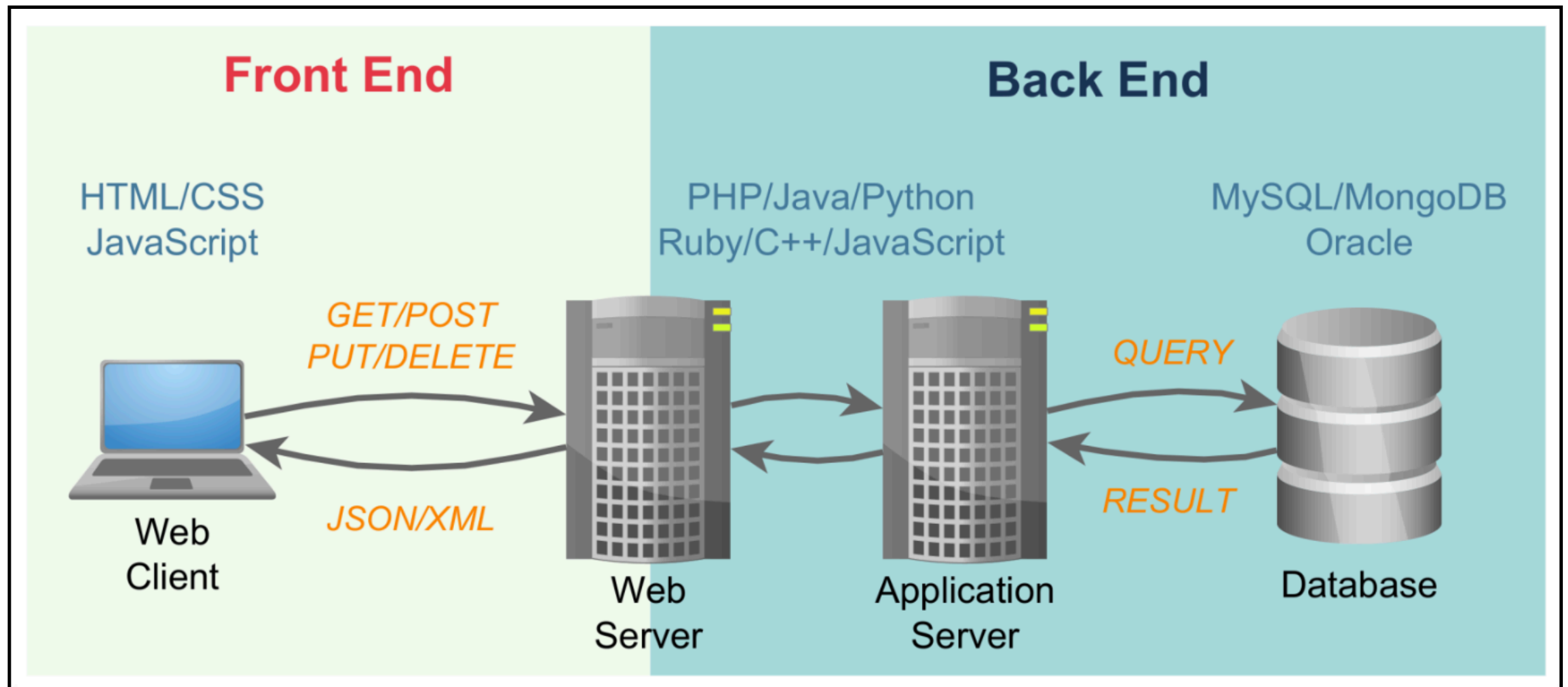
- *Middleware is a computer software that connects software components or applications.
- *It mediates with other software and allows for communication between disparate applications in a heterogeneous system.
- *The need for middleware arises when distributed systems become too complex to manage efficiently without a common interface.
- ***Advantages**
 - *make heterogeneous systems work efficiently across a network.
 - *be flexible enough to incorporate frequent modifications.
 - *hide the underlying complexity of distributed systems.

Example of Middleware



- * Database-oriented middleware connects applications to any type of database (not necessarily a relational DBMS through SQL).
- * It connects applications with databases across the network.
- * It acts as mediators in distributed DBMSs to translate one database language into to another language (for example, Oracle SQL into IBM's DB2 SQL).
- * Examples
 - * Microsoft's ODBC (Open Database Connectivity) API exposes a single interface to facilitate access to a database and uses drivers to accommodate differences between databases
 - * The JDBC API uses a single set of Java methods to facilitate access to multiple databases.

Full-stack Web Development



<https://circuitcellar.com/research-design-hub/basics-of-design/backend-web-development-for-mcu-clients/>