Database Environment

Objectives

- Purpose of three-level database architecture.
- Contents of external, conceptual, and internal levels.
- Purpose of external/conceptual and conceptual/internal mappings.
- Meaning of logical and physical data independence.
- Distinction between DDL and DML.
- A classification of data models.

Objectives

- Purpose/importance of conceptual modeling.
- Typical functions and services a DBMS should provide.
- Function and importance of system catalog.
- Software components of a DBMS.
- Meaning of client–server architecture and advantages of this type of architecture for a DBMS.
- Function and uses of Transaction Processing Monitors.

Objectives of Three-Level Architecture

- All users should be able to access same data.
- A user's view is immune to changes made in other views.
- Users should not need to know physical database storage details.

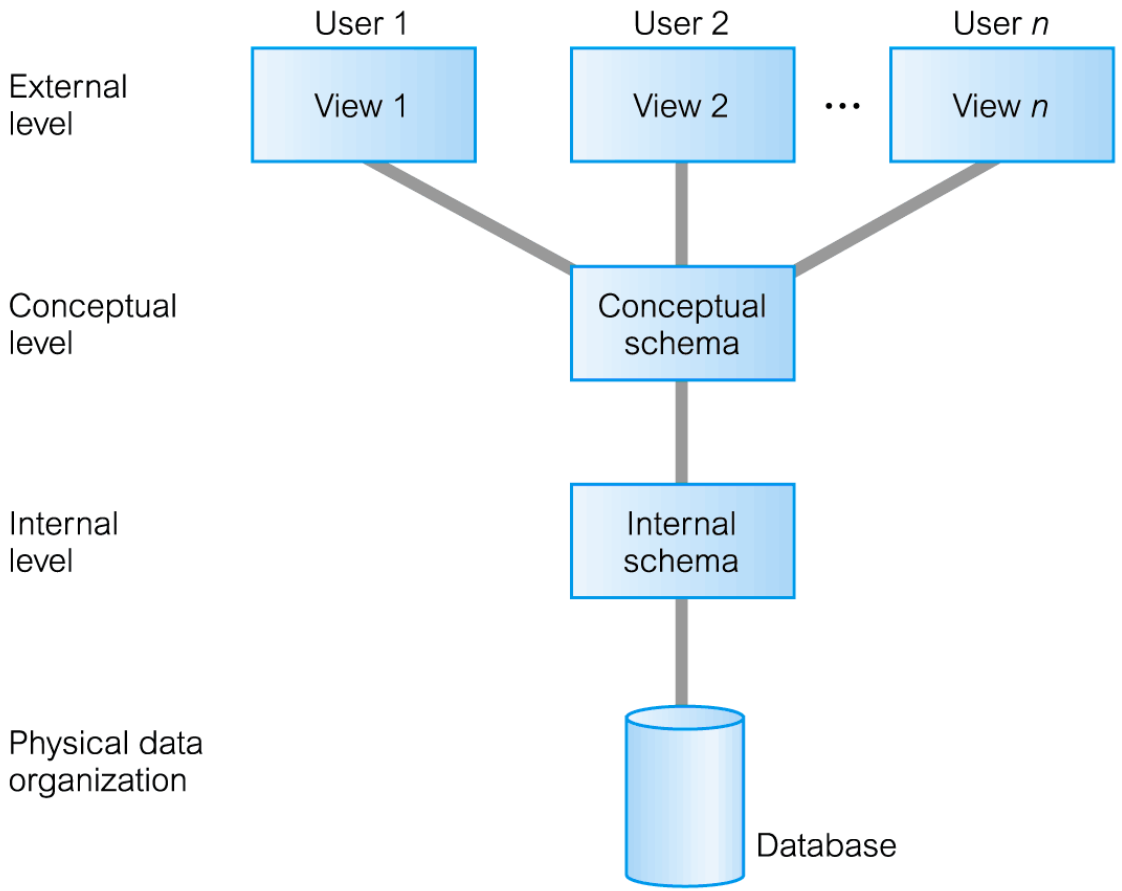
Objectives of Three-Level Architecture

- DBA should be able to change database storage structures without affecting the users' views.

- Internal structure of database should be unaffected by changes to physical aspects of storage.

- DBA should be able to change conceptual structure of database without affecting all users.

ANSI-SPARC Three-Level Architecture



ANSI-SPARC Three-Level Architecture

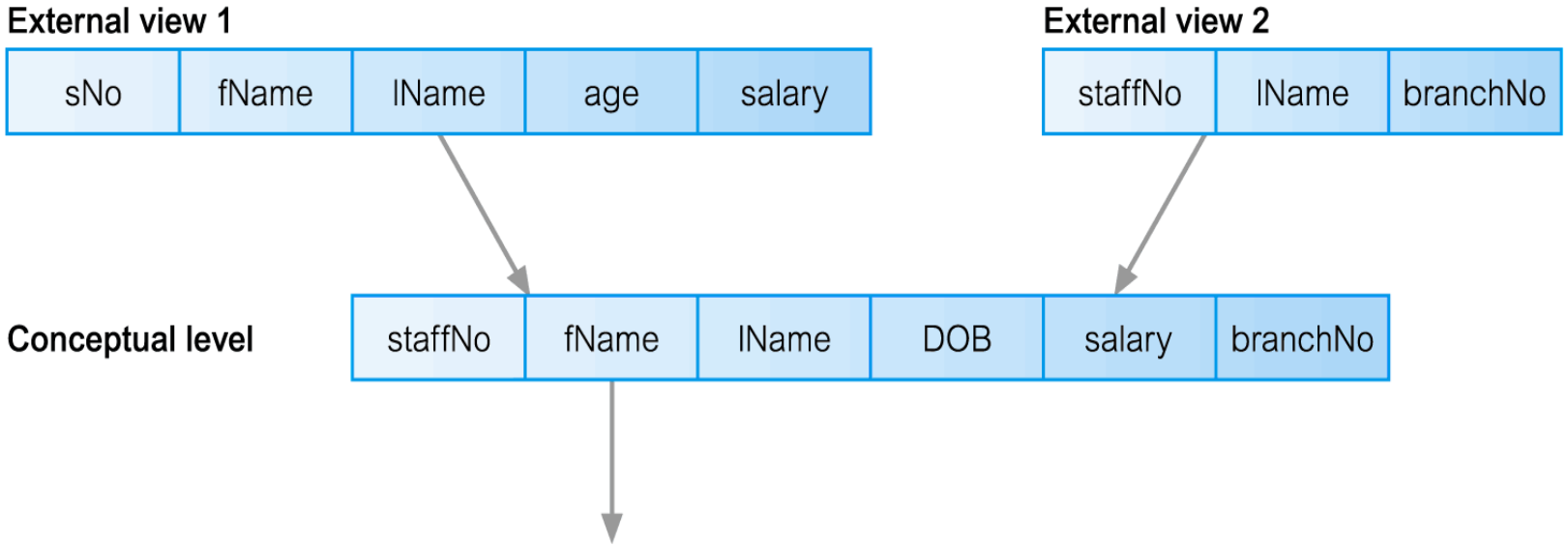
- **External Level**
  - Users' view of the database.
  - Describes that part of database that is relevant to a particular user.

- **Conceptual Level**
  - Community view of the database.
  - Describes what data is stored in database and relationships among the data.

ANSI-SPARC Three-Level Architecture

- **Internal Level**
  - Physical representation of the database on the computer.
  - Describes how the data is stored in the database.

Differences between Three Levels of ANSI-SPARC Architecture




```
struct STAFF {
      int staffNo;
      int branchNo;
      char fName [15];
      char lName [15];
      struct date dateOfBirth;
      float salary;
      struct STAFF *next;                 /* pointer to next Staff record */
};
index staffNo; index branchNo;            /* define indexes for staff */
```

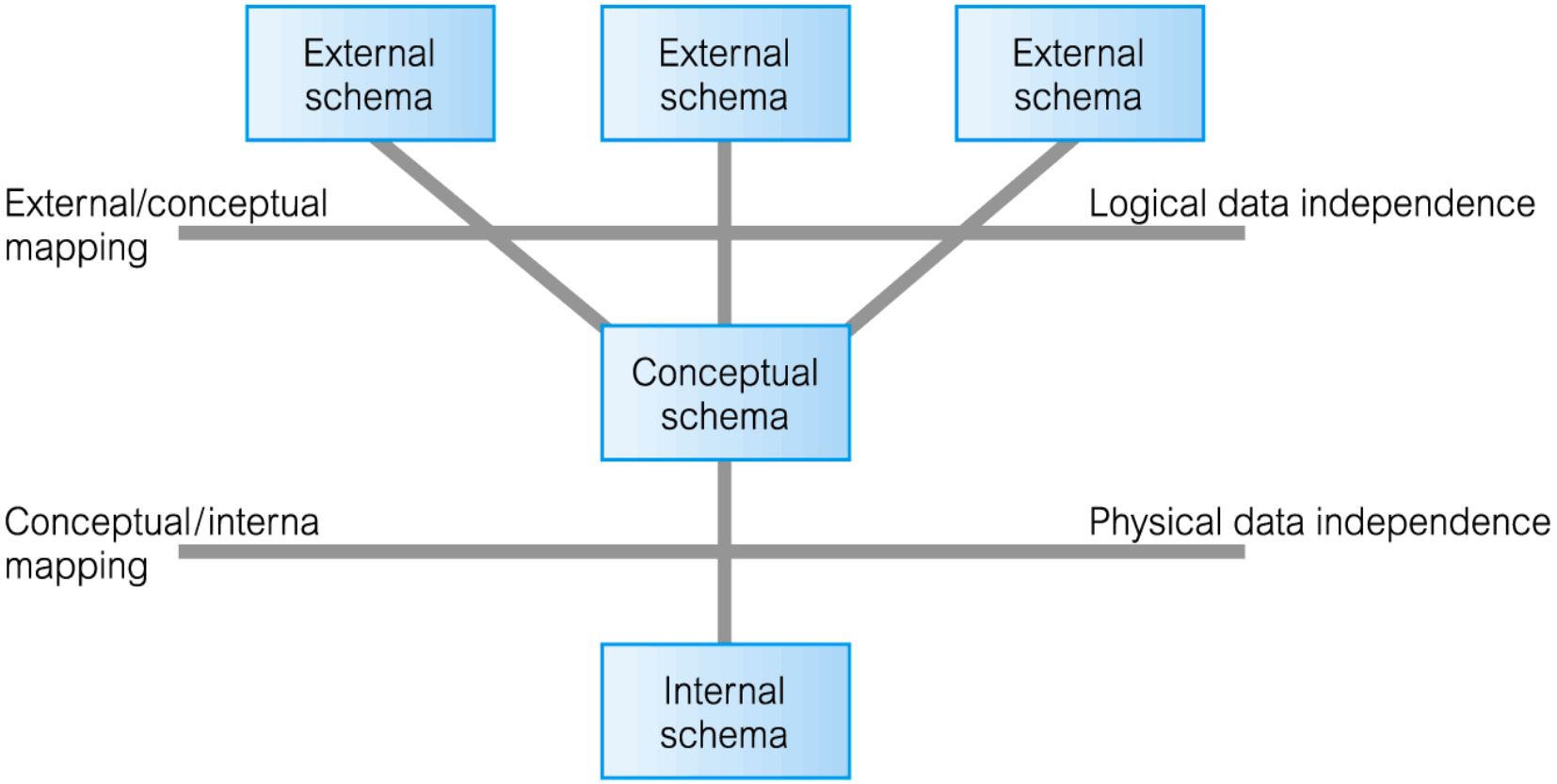
Data Independence

- **Logical Data Independence**
  - Refers to immunity of external schemas to changes in conceptual schema.
  - Conceptual schema changes (e.g. addition/removal of entities).
  - Should not require changes to external schema or rewrites of application programs.

Data Independence

- **Physical Data Independence**
  - Refers to immunity of conceptual schema to changes in the internal schema.
  - Internal schema changes (e.g. using different file organizations, storage structures/devices).
  - Should not require change to conceptual or external schemas.

Data Independence and the ANSI-SPARC Three-Level Architecture



Database Languages

- **Data Definition Language (DDL)**
  - Allows the DBA or user to describe and name entities, attributes, and relationships required for the application
  - plus any associated integrity and security constraints.

Database Languages

- **Data Manipulation Language (DML)**
  - Provides basic data manipulation operations on data held in the database.
- **Procedural DML**
  - allows user to tell system exactly how to manipulate data.
- **Non-Procedural DML**
  - allows user to state what data is needed rather than how it is to be retrieved.
- **Fourth Generation Languages (4GLs)**
  - Smalltalk, SQL

Data Model

Integrated collection of concepts for describing data, relationships between data, and constraints on the data in an organization.

- **Data Model comprises:**
  - a structural part;
  - a manipulative part;
  - possibly a set of integrity rules.

Data Model

- **Purpose**
  - To represent data in an understandable way.

- **Categories of data models include:**
  - Object-based
  - Record-based
  - Physical.

Data Models

- **Object-Based Data Models**
  - Entity-Relationship
  - Semantic
  - Functional
  - Object-Oriented.
- **Record-Based Data Models**
  - Relational Data Model
  - Network Data Model
  - Hierarchical Data Model.
- **Physical Data Models**

Relational Data Model

Branch

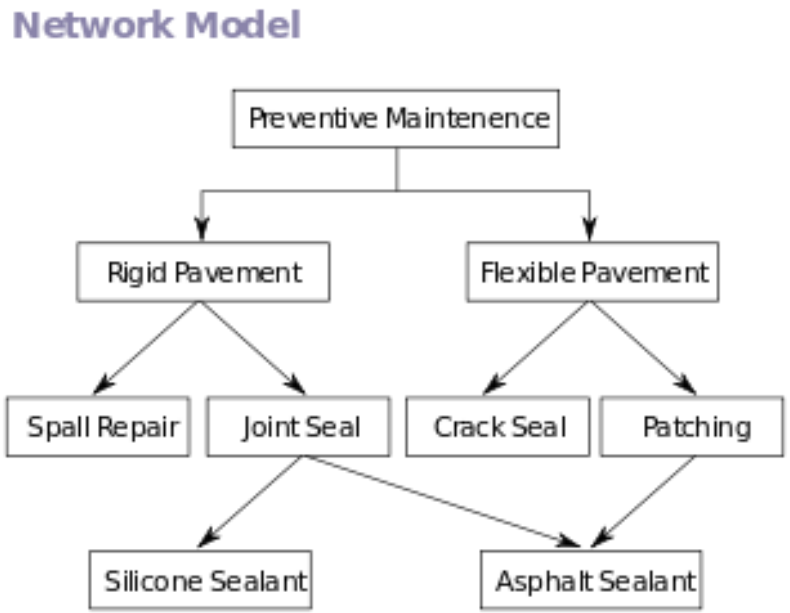
branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

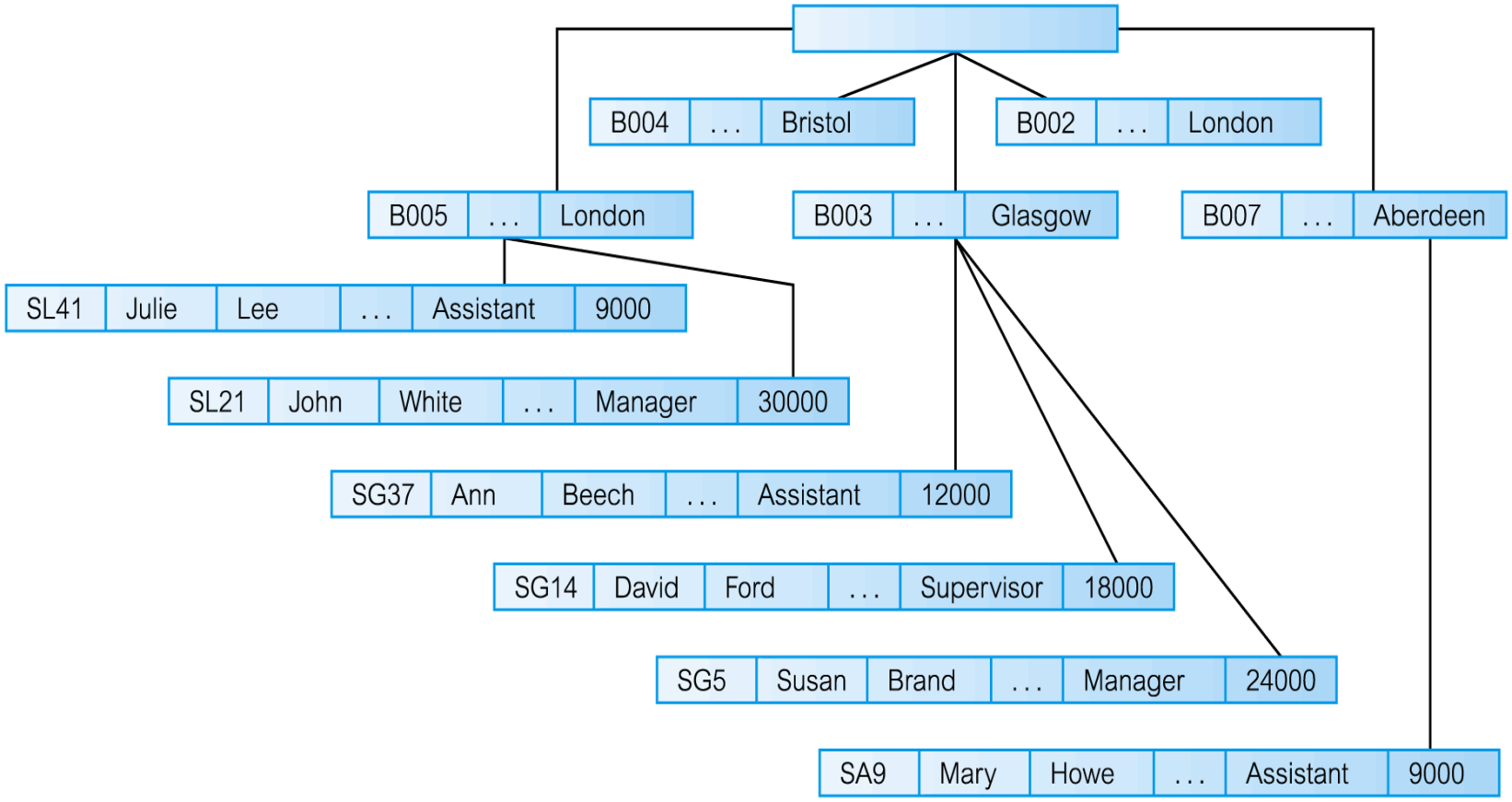
staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Network Data Model

- Its distinguishing feature is that the schema, viewed as a graph in which object types are nodes and relationship types are arcs, is not restricted to being a hierarchy or lattice.



Hierarchical Data Model



Conceptual Modeling

- Conceptual schema is the core of a system supporting all user views.
- Should be complete and accurate representation of an organization's data requirements.
- Conceptual modelling is process of developing a model of information use that is independent of implementation details.
- Result is a conceptual data model.

Functions of a DBMS

- Data Storage, Retrieval, and Update.
- A User-Accessible Catalog.
- Transaction Support.
- Concurrency Control Services.
- Recovery Services.

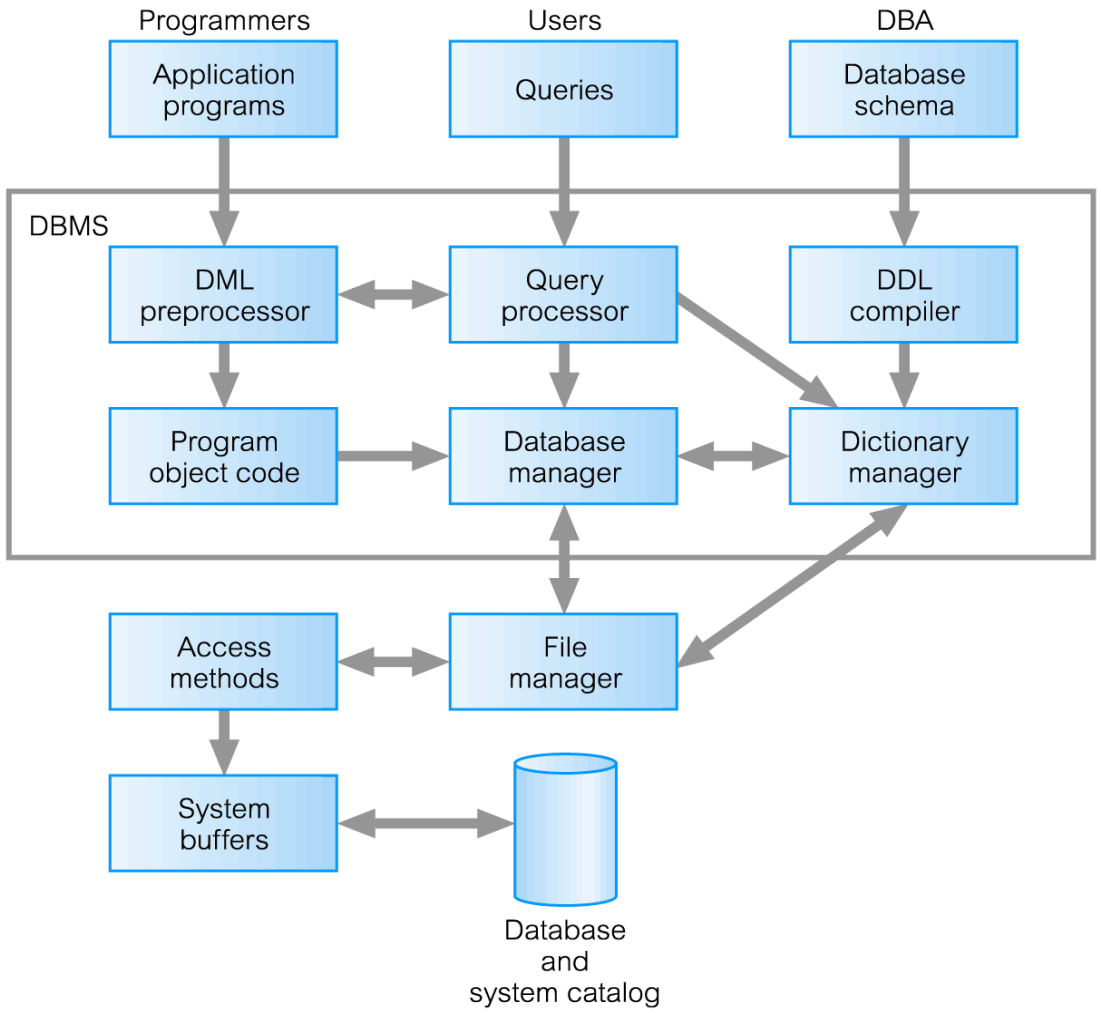
Functions of a DBMS

- Authorization Services.
- Support for Data Communication.
- Integrity Services.
- Services to Promote Data Independence.
- Utility Services.

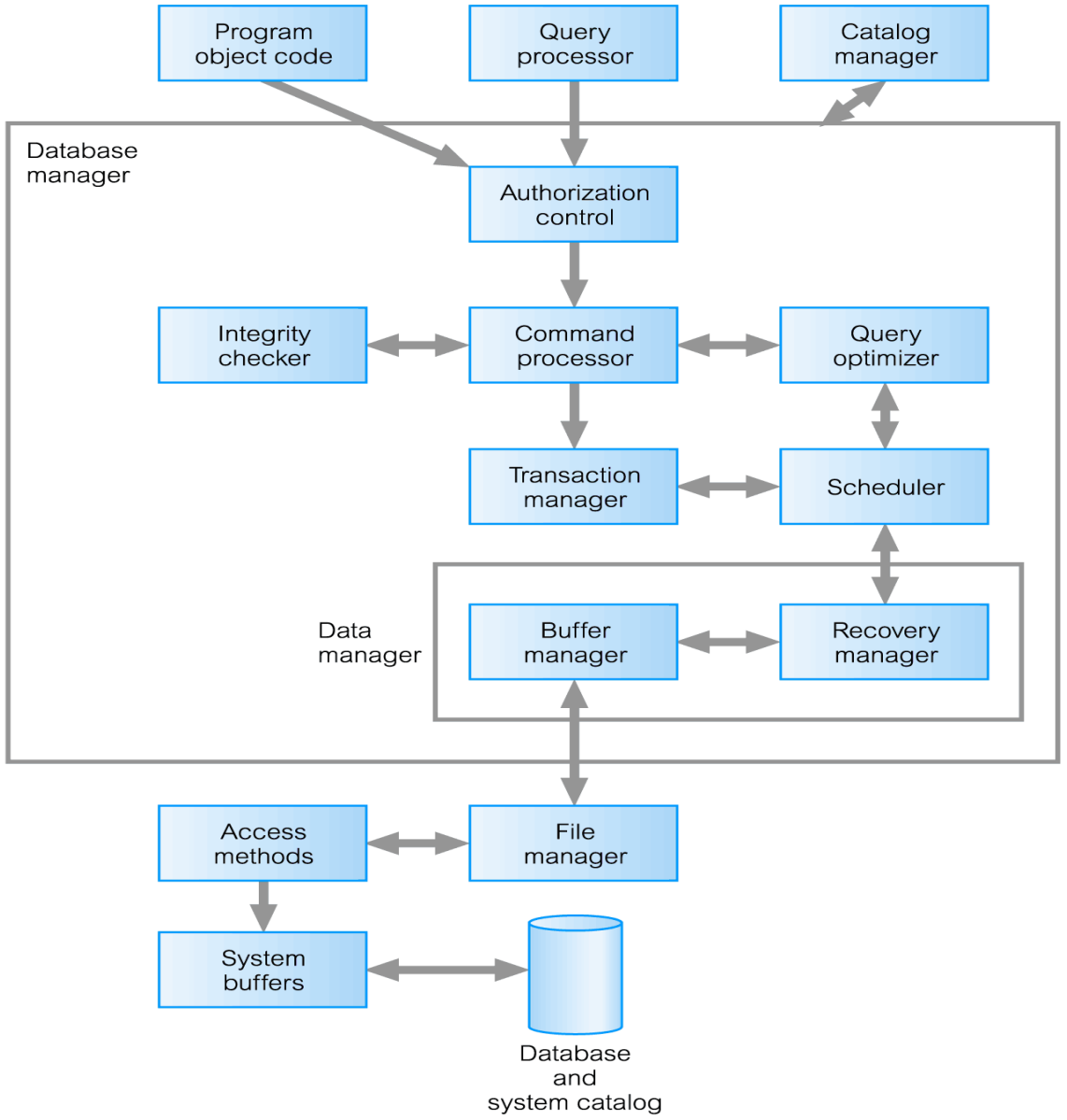
System Catalog

- **Repository of information (metadata) describing the data in the database.**
- **One of the fundamental components of DBMS.**
- **Typically stores:**
  - names, types, and sizes of data items;
  - constraints on the data;
  - names of authorized users;
  - data items accessible by a user and the type of access;
  - usage statistics.

Components of a DBMS



Components of Database Manager

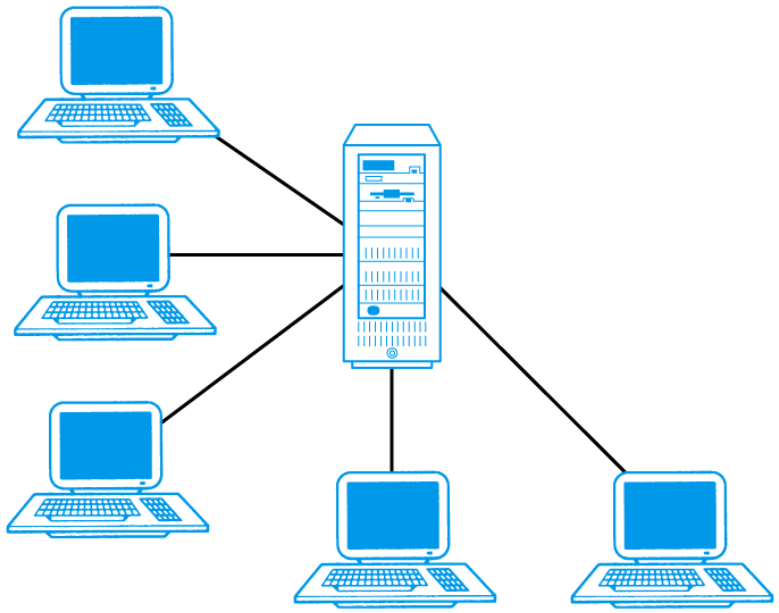


Multi-User DBMS Architectures

- Teleprocessing
- File-server
- Client-server

Teleprocessing

- Traditional architecture.
- Single mainframe with a number of terminals attached.
- Trend is now towards downsizing.

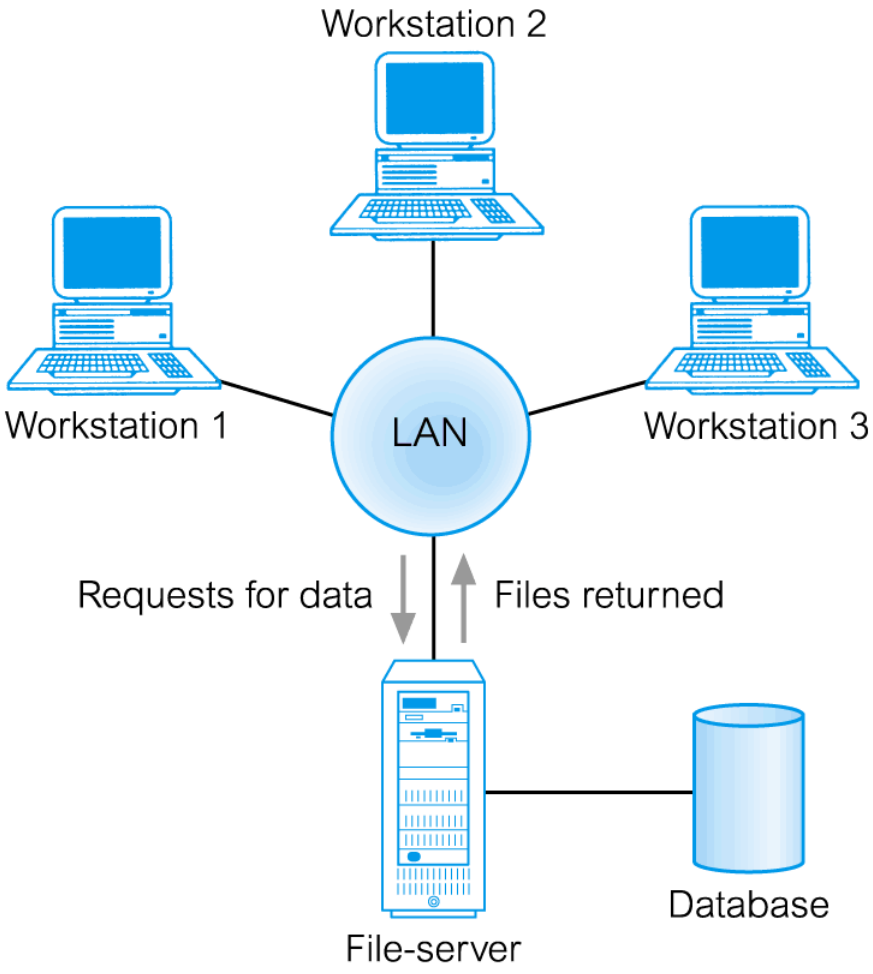


File-Server

- File-server is connected to several workstations across a network.
- Database resides on file-server.
- DBMS and applications run on each workstation.
- Disadvantages include:
  - Significant network traffic.
  - Copy of DBMS on each workstation.
  - Concurrency, recovery and integrity control more complex.

File-Server Architecture

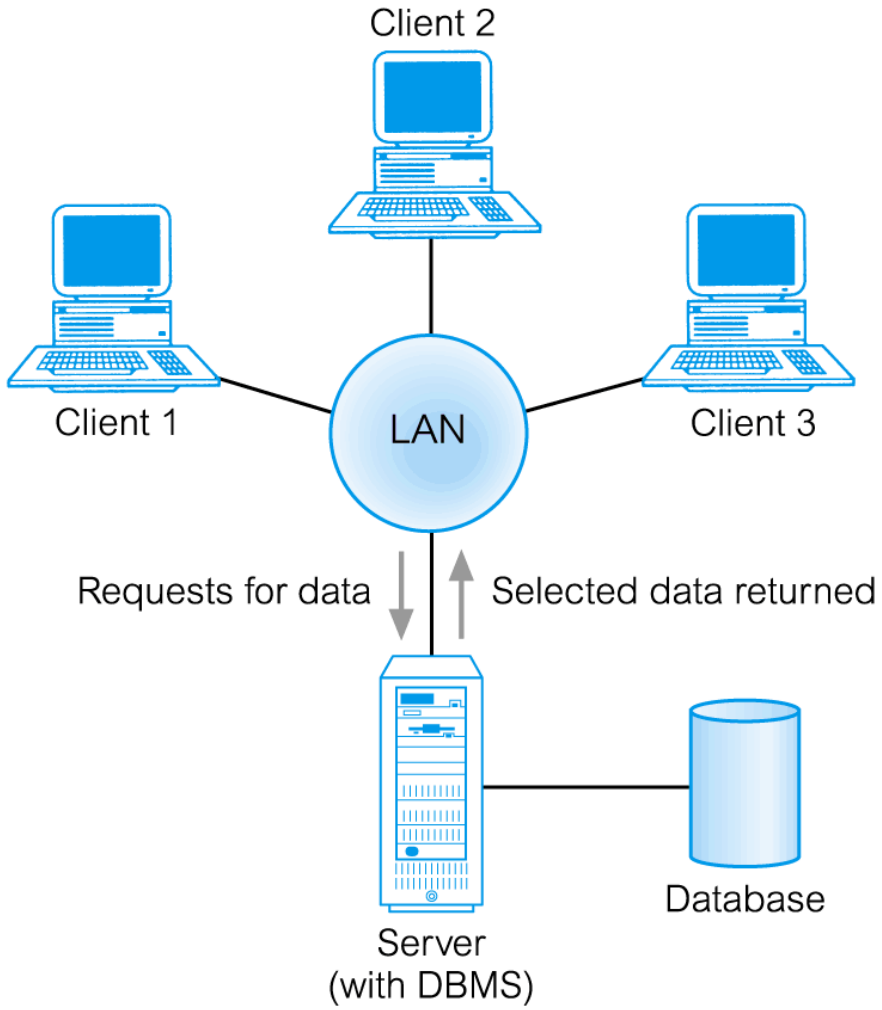
- Usually, a file server does not perform computational tasks, and does not run programs on behalf of its clients.
- It is designed primarily to enable the storage and retrieval of data while the computation is carried out by the workstations.



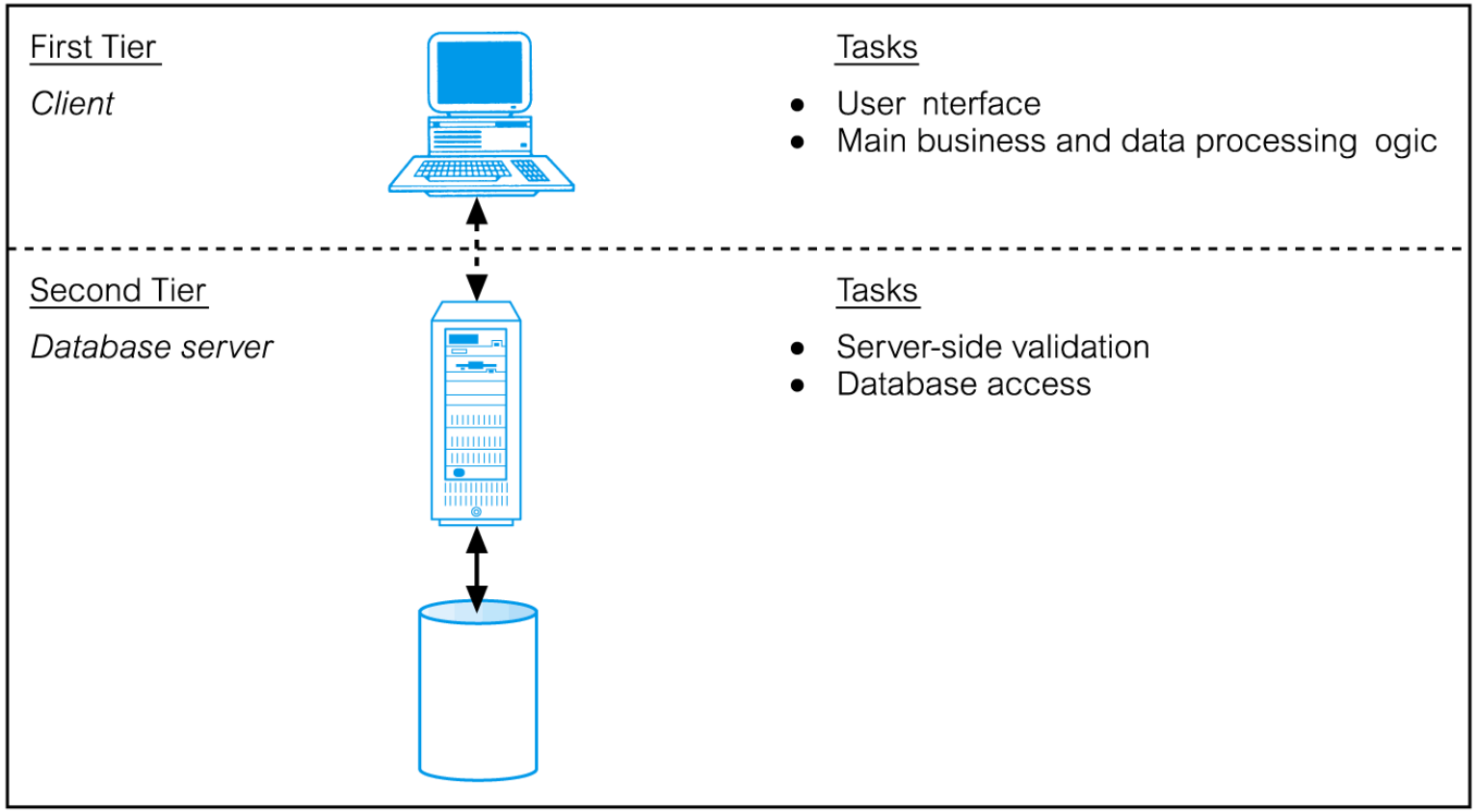
Traditional Two-Tier Client-Server

- **Client (tier 1) manages user interface and runs applications.**
- **Server (tier 2) holds database and DBMS.**
- **Advantages include:**
  - wider access to existing databases;
  - increased performance;
  - possible reduction in hardware costs;
  - reduction in communication costs;
  - increased consistency.

Traditional Two-Tier Client-Server



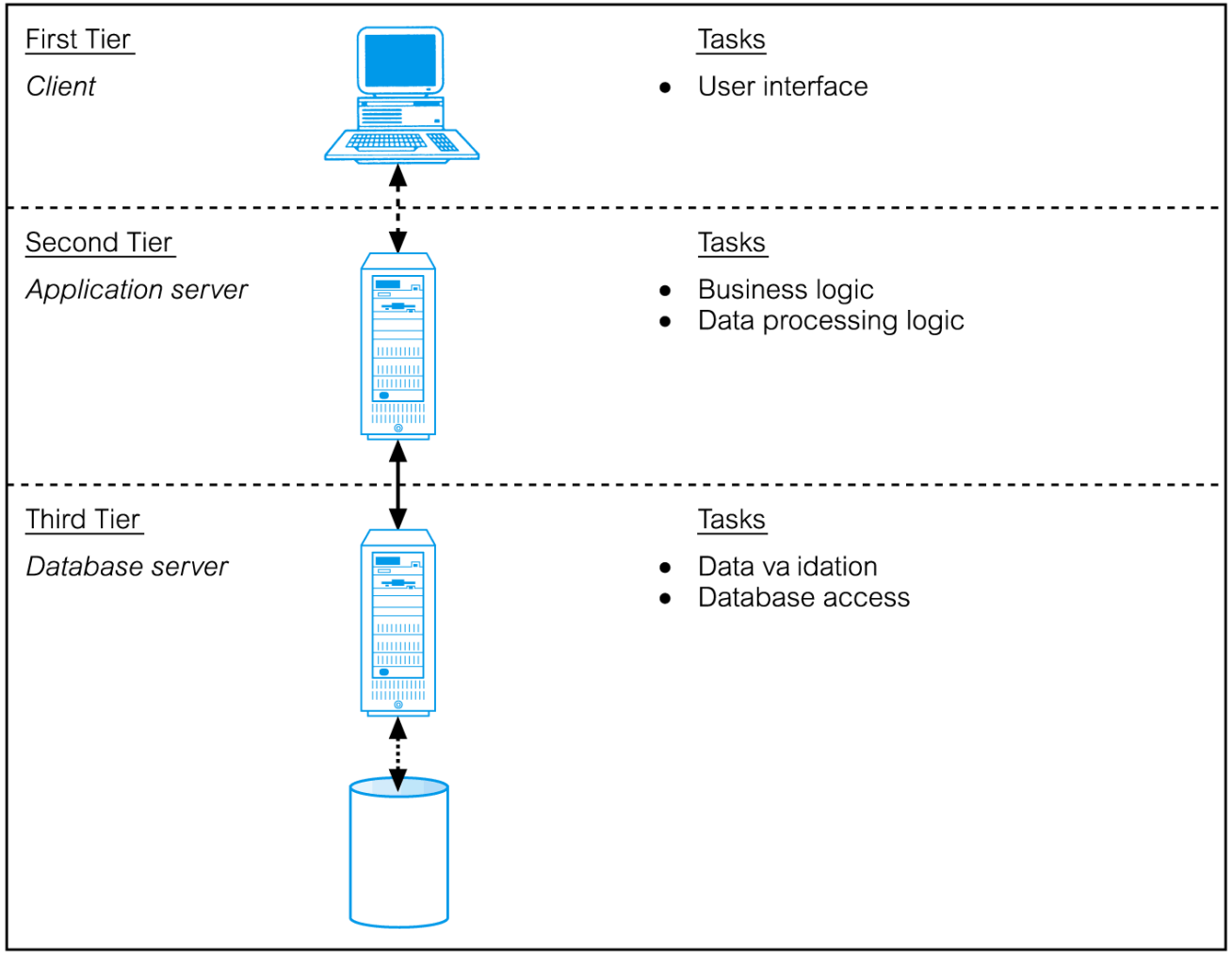
Traditional Two-Tier Client-Server



Three-Tier Client-Server

- **In two-tier client side presented two problems preventing true scalability:**
  - "Fat" client, requiring considerable resources on client's computer to run effectively.
  - Significant client side administration overhead.
- **By 1995, three layers proposed, each potentially running on a different platform.**

Three-Tier Client-Server



Three-Tier Client-Server

- **Advantages:**
  - ‘Thin’ client, requiring less expensive hardware.
  - Application maintenance centralized.
  - Easier to modify or replace one tier without affecting others.
  - Separating business logic from database functions makes it easier to implement load balancing.
  - Maps quite naturally to Web environment.

Transaction Processing Monitors (TPM)

- **Program that controls data transfer between clients and servers in order to provide a consistent environment.**
  - It monitors transactions from one stage to the next, ensuring that each one completes successfully;
  - if not, or if an error occurs, the TM Monitor takes the appropriate action.
  - A TPM main purpose/objective is to allow resource sharing and assure optimal use of the resources by applications.

TPM as middle tier of 3-tier client-server