Database Systems

Introduction to the course

Venkateswara Rao Kagita

Text Books

- **Adhsakkdi Y Raghuram Krishnan and Johannes Gehrke, *Database Management Systems*, 3/e, TMH, 2007.**

- Elamsri, Navathe, Somayajulu and Gupta, *Database Concepts*, Pearson Edition, 2006.

- Introduction to Database Systems, CJ Date, Pearson
- The Database Systems – The Complete Book, HG Molina, J D Ullman, J Widom Pearson
- Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7th Edition.
-

Class Rules

- Assignments are to be completed individually
- Academic **honesty** taken seriously

Syllabus

Topic	#hours
• Overview of Database Systems	4
• Introduction to Database Design, ER-Model	4
• Relational model	4
• Relational algebra and Calculus	5
• SQL	6
• Functional Dependencies and normalization process	4
• Transaction Management	5
• Concurrency Control	4
• Crash recovery	3
• Security and Authorization	3
Total	42

Overview of Database Systems

What is Database System ?

- **Data:**
  - Known facts that can be recorded and have an implicit meaning.
- **Database:**
  - A collection of related data.
- **Database Management System (DBMS):**
  - Software for creating, storing, maintaining, and accessing database files
  - Makes using databases more efficient
  - There are many databases available like MySQL, Sybase, Oracle, MongoDB, Informix, PostgreSQL, SQL Server, etc.
- **Database System:**
  - The DBMS software together with the data itself. Sometimes, the applications are also included.

What is the need of DBMS?

- When dealing with huge amount of data, there are two things that require optimization:
  - **Data storage:** Efficient storage, no redundancy.
  - **Data retrieval:** Retrieve the data quickly when needed
- Purpose of Database Systems
  - The main purpose of database systems is to manage the data.
  - To manage the data we need to store this data somewhere where we can add new data, delete unused data, update outdated data, retrieve data.
  - To perform these operations on data we need a Database management system that allows us to store the data in such a way so that all these operations can be performed on the data efficiently.
- Why Use a DBMS ?
  - Large amount of data (Giga's, Tera's) drives many business processes
  - Data is very structured
  - Persistent and valuable data
  - Performance requirements
  - Concurrent and restricted access to the data

Simplified database system environment

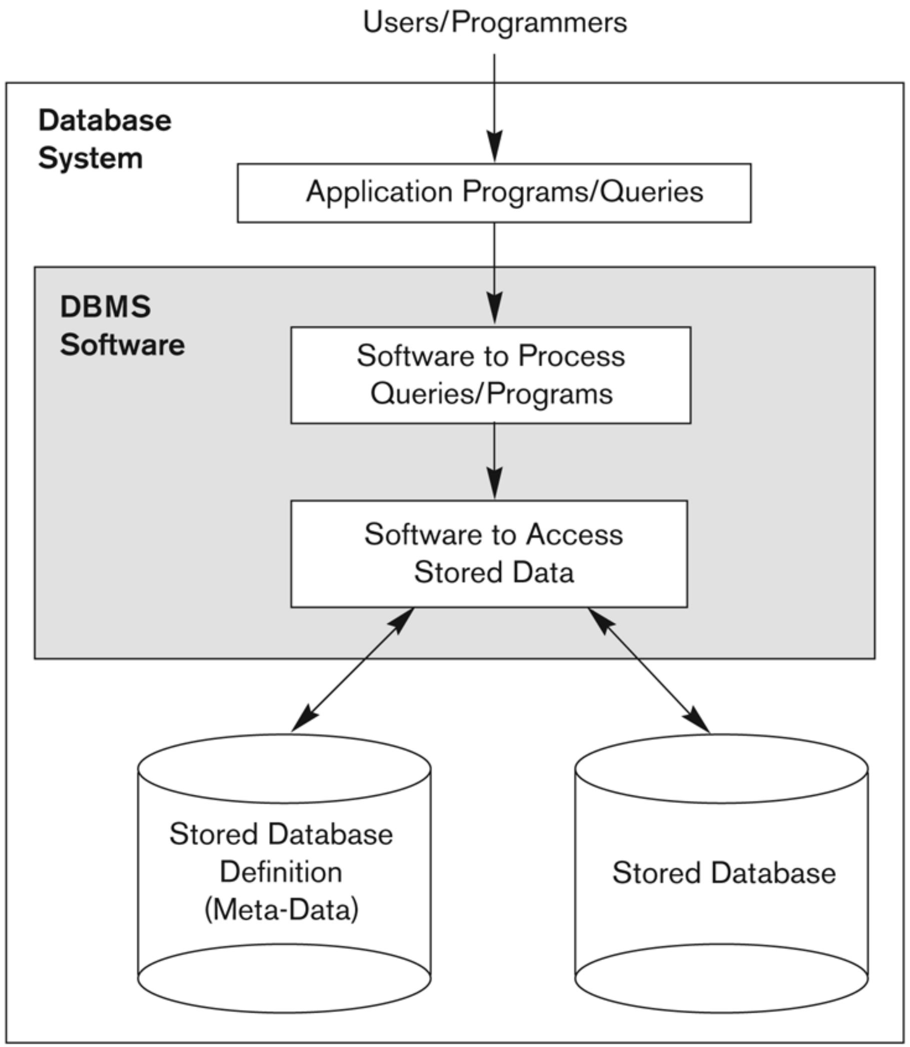
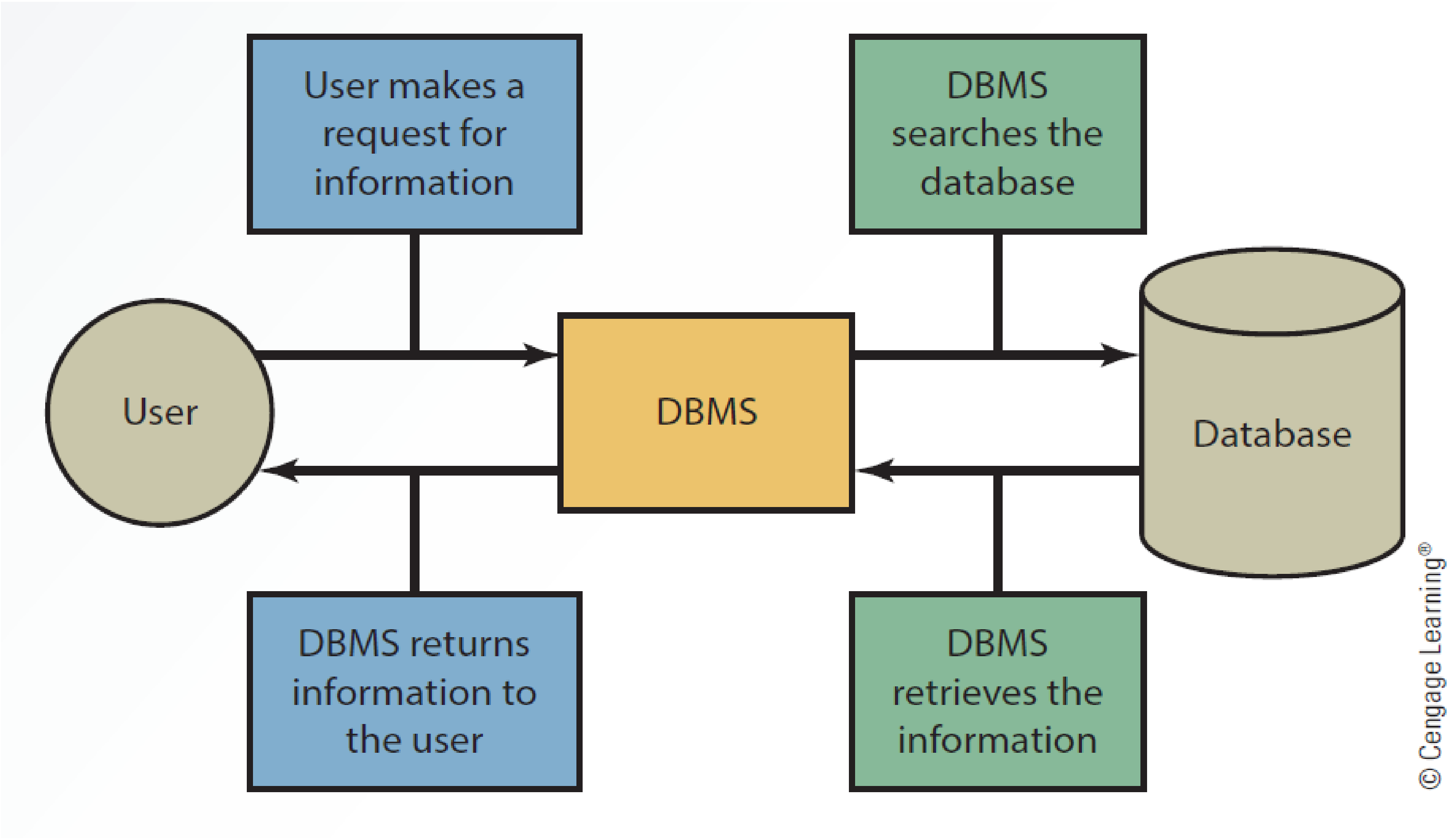


Figure 1.1
A simplified database system environment.

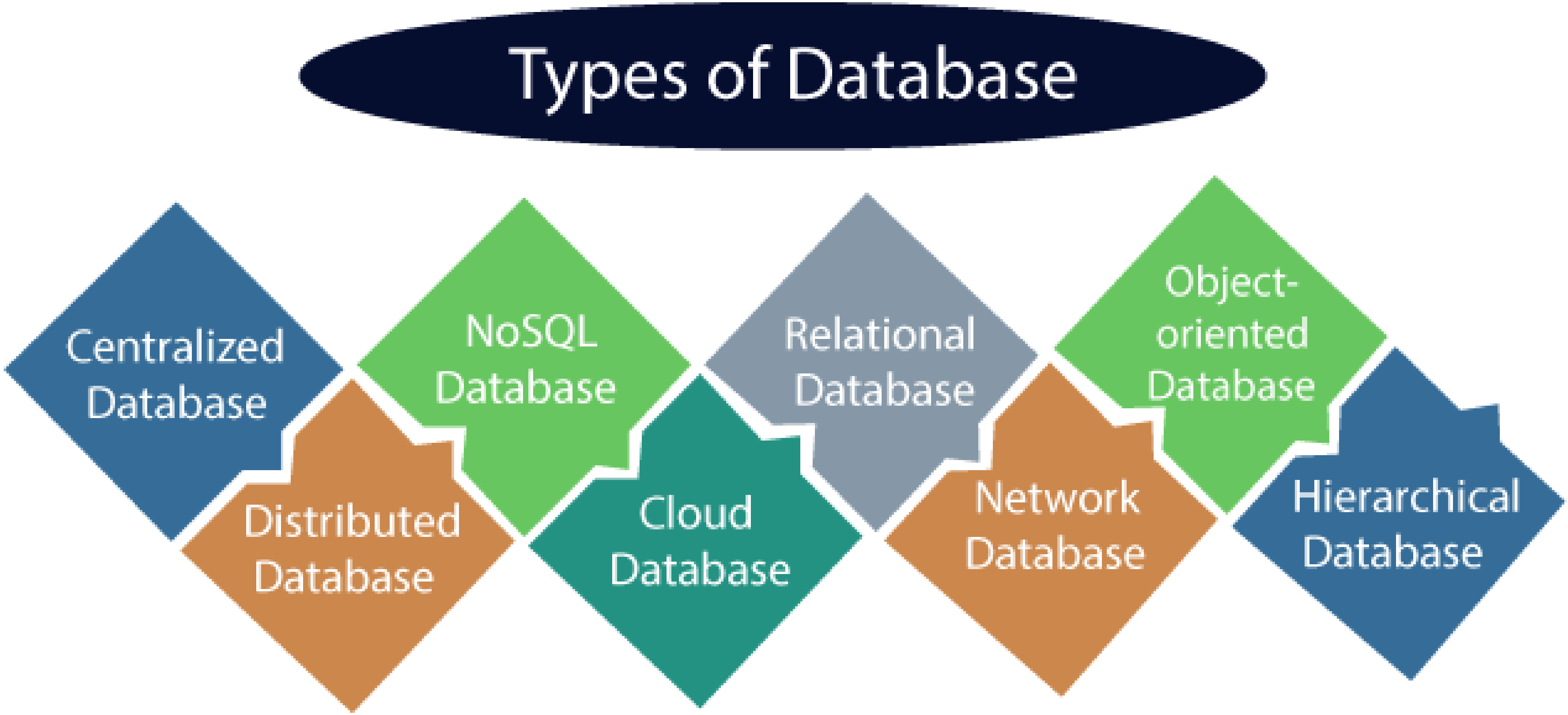
Interaction Between the User, DBMS and Database



A few of Applications

- **Telecom**: Calls made, network usage, customer details etc.
- **Industry**: Where it is a manufacturing unit, to keep the records of ins and outs.
- **Banking System**: For storing customer info, tracking day to day credit and debit transactions, generating bank statements etc.
- **Sales**: To store customer information, production information and invoice details.
- **Airlines**: Reservation information along with flight schedule is stored in database.
- **Education sector**: To store and retrieve the data regarding student details, staff details, course details, exam details, payroll data, attendance details, fees details etc.
- **Online shopping**: To store the product information, your addresses and preferences, credit details and provide you the relevant list of products based on your query.

Types of Database



https://www.javatpoint.com/types-of-databases

Why not use simple file to store the data ?

Drawbacks of File system

- **Data redundancy**
  - Duplication of data
  - Requires more storage than needed.
  - Higher storage costs and poor access time.
- **Data inconsistency:**
  - Data redundancy leads to data inconsistency
- **Data Isolation:** a property that determines when and how changes made by one operation become visible to other concurrent users and systems. Difficult to achieve in file processing system.
- **Dependency on application programs:** Changing files would lead to change in application programs.
- **Atomicity issues:** "All or nothing"
  - For example: Ram transfers 1000 INR to Krishna account.
    - Debt 1000 rupees from Ram account
    - Credit 1000 rupees to Krishna account
    - What if the system fails after first operation ?
    - In such case the rollback of operation should occur to maintain the atomicity of transaction.
  - It is **difficult to achieve atomicity in file processing systems**.
- **Data Security:**
  - Data should be secured from unauthorized access.
  - For example a student should not be able to access grades.
  - Such kind of security constraints are difficult to apply in file processing systems.

Advantage of DBMS over file system

- **Data Independence**
  - Application programs should not, ideally, be exposed to details of data representation and storage.
- **No redundant data**:
  - Data normalization removes the data redundancy.
  - Saves storage and improves access time.
- **Data Consistency**
- **Data Integrity and Security**
  - data is always accessed through the DBMS, the DBMS can enforce integrity constraints.
  - Each user has a different set of access thus data is secured.
- **Privacy**: Limited access means privacy of data.
- **Easy access to data** – Data is easily accessible with fast response times.
- **Concurrent Access**
- **Easy recovery**:
  - Keeps the backup of data.
  - Easy to recover data in case of a failure.
- **Flexible**: Database systems are more flexible than file processing systems.

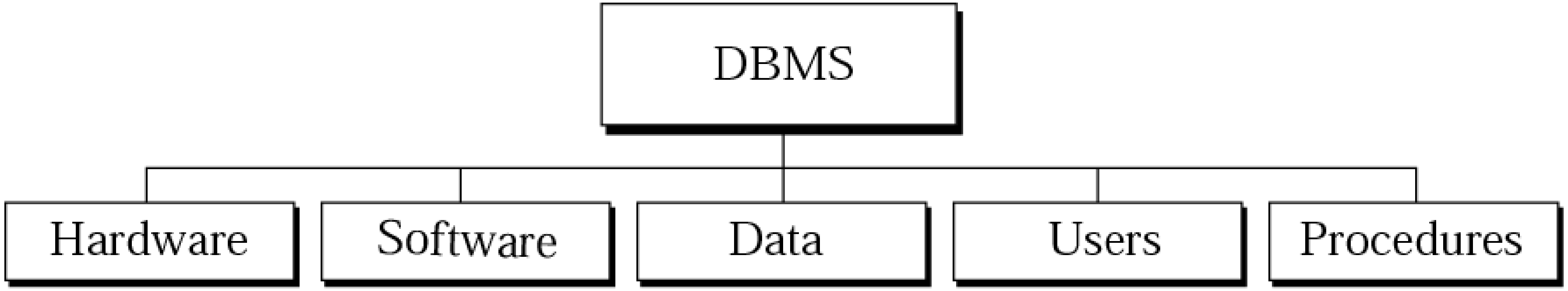
Disadvantages of DBMS:

- DBMS implementation cost is high compared to the file system
- Complexity: Database systems are complex to understand
- Performance: Database systems are generic, making them suitable for various applications. However this feature affect their performance for some applications.

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.
- When a DBMS may be unnecessary:
  - If the database and applications are simple, well defined, and not expected to change.
  - If there are stringent real-time requirements that may not be met because of DBMS overhead.
  - If access to data by multiple users is not required.
- When no DBMS may suffice:
  - If the database system is not able to handle the complexity of data because of modeling limitations
  - If the database users need special operations not supported by the DBMS.

DBMS components



- Hardware –
  the physical computer system that allows physical access to data.
- Software –
  the actual program that allows users to access, maintain, and update physical data.
- Data – stored physically on the storage devices
- Users –
  - database implementors, who build DBMS software,
  - End users - Normal user and DBA (Database Administrator)
  - Application programs, Application programmers
- Procedures –
  a set of rules that should be clearly defined and followed by the users.

Database Administrator Responsibilities

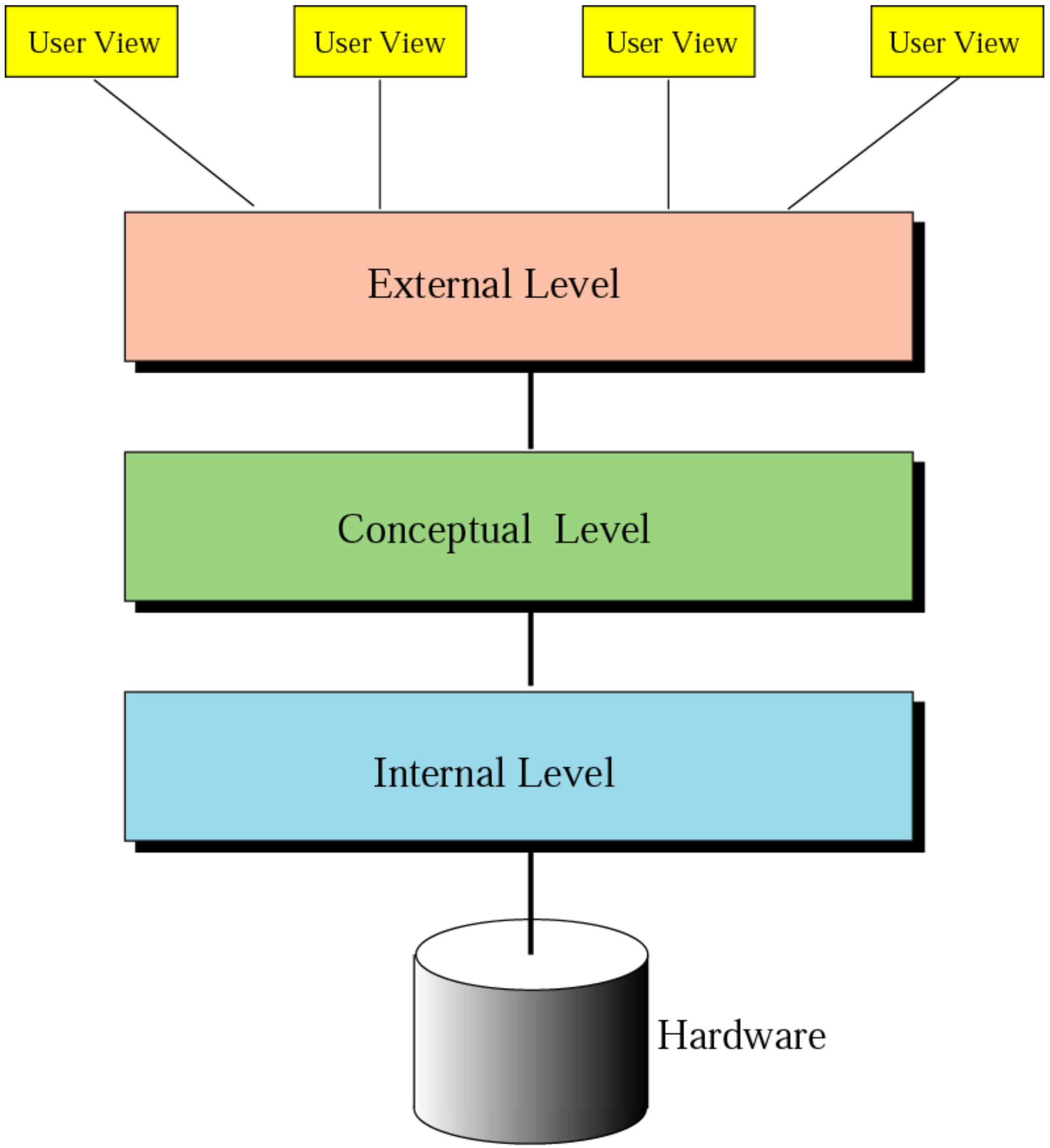
- Design of the Conceptual and Physical Schemas

- Security and Authorization

- Data Availability and Recovery from Failures.

- Database Tuning.

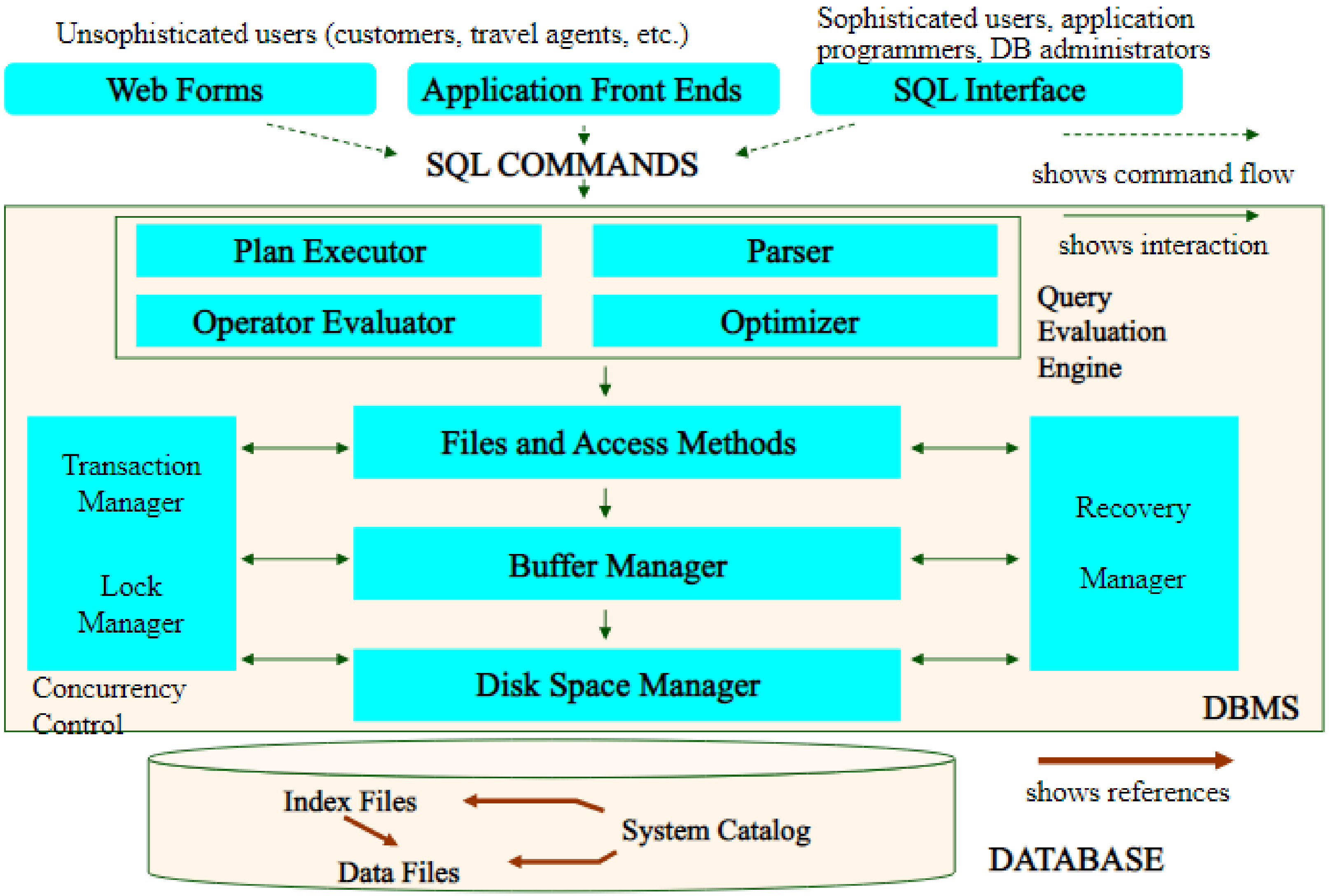
Database Architecture



Architecture

- Internal level –
  - Determines where data are actually stored on the storage device.
  - Low-level access method
- Conceptual level –
  - Defines the logical view of the data
  - The main functions of DBMS are in this level.
- External level –
  - Interacts directly with the user.
  - Change the data coming from the conceptual level to a format and view that are familiar to the users.

DBMS Architecture: Revisited

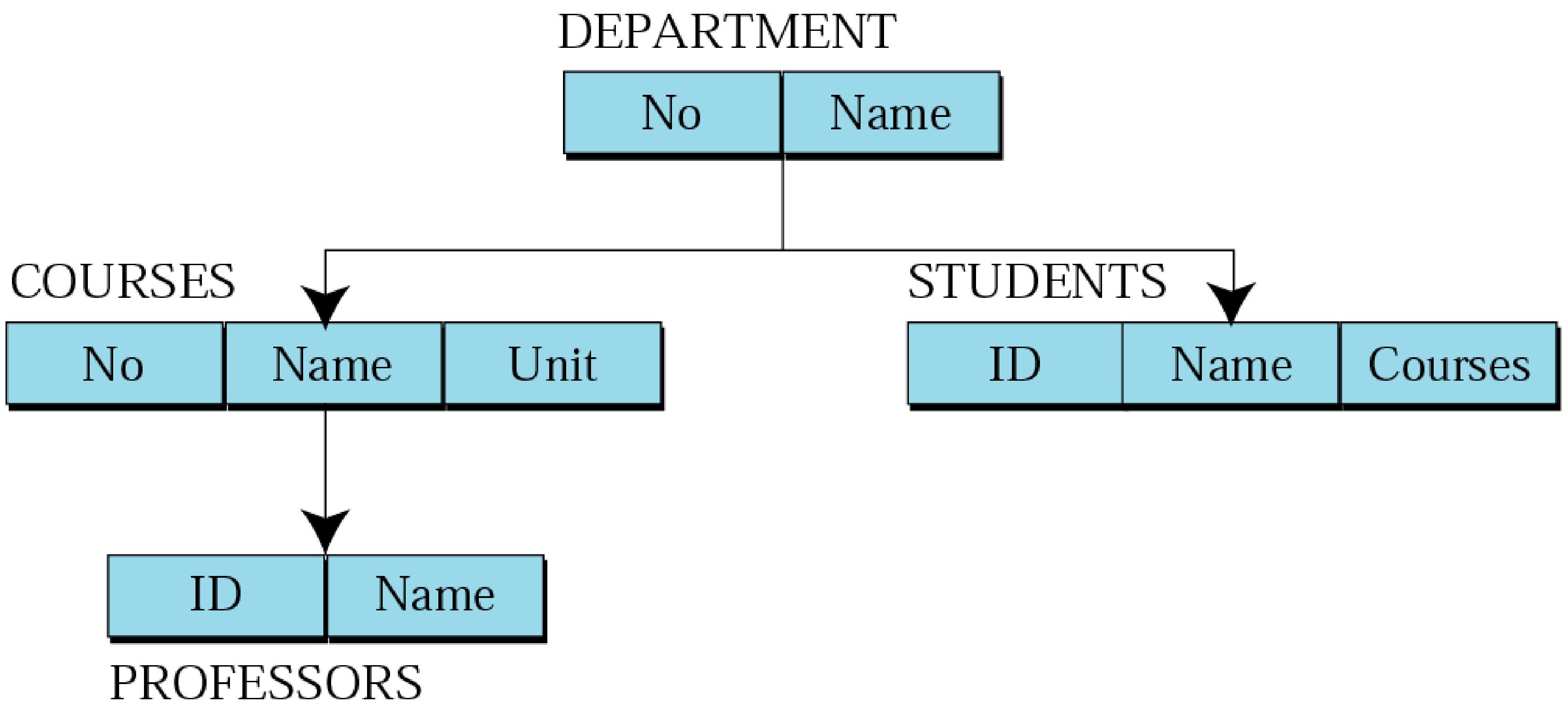


Database models

- A **data model** is a collection of high-level data description constructs that hide many low level storage details
- Defines the logical design of data.
- Describes the relationships between different parts of data.
- Different models
  - Hierarchical model (e.g., used in IBM's IMS DBMS)
  - Network model (e.g., used in IDS and IDMS)
  - Relational model (e.g. Oracle, Sybase, Microsoft's Access, etc.)
  - The object-oriented model (e.g., Objectstore and Versant)
  - The object-relational model (e.g., used in DBMS products from IBM, Informix, ObjectStore, Oracle, Versant, and others).
  - …
- While many databases use the hierarchical and network models and systems based on the object-oriented and object-relational models are gaining acceptance in the marketplace, the dominant model is the relational model.

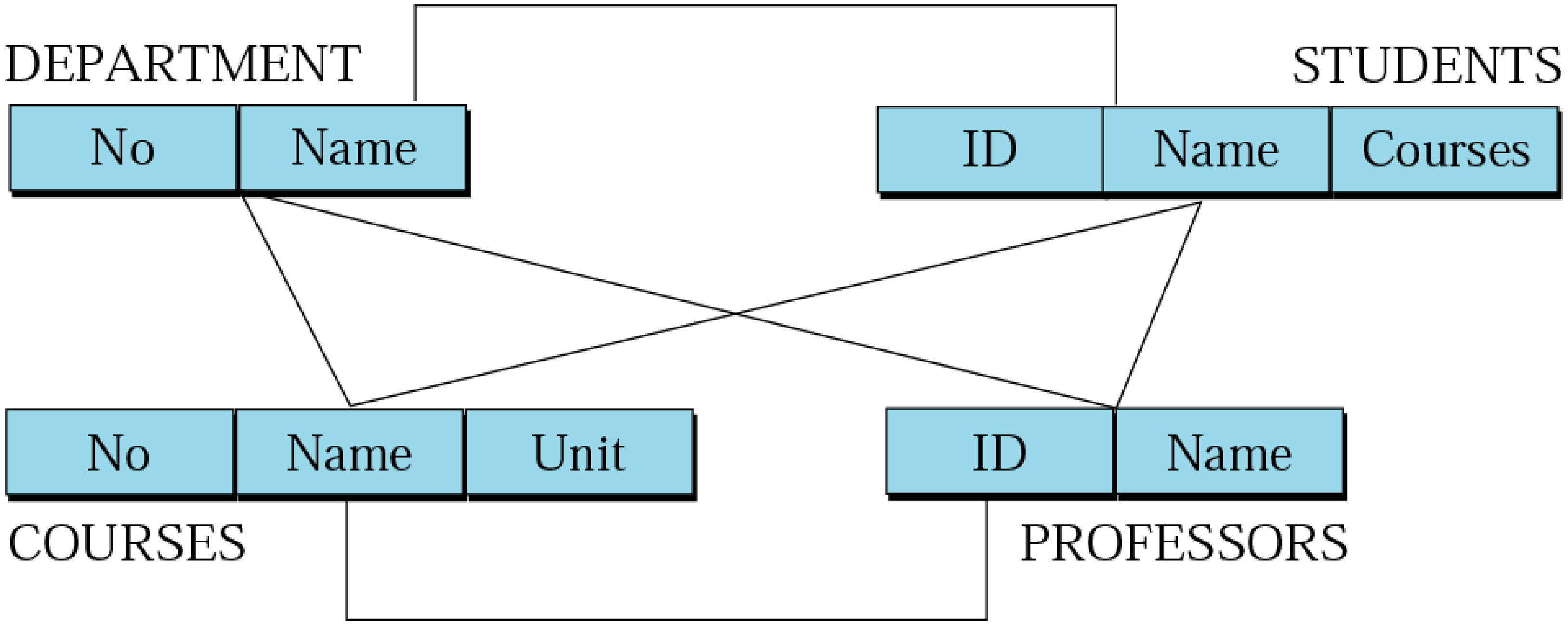
Hierarchical model

- Data are organized as an upside down tree.
- Each entity has only one parent but can have several children.



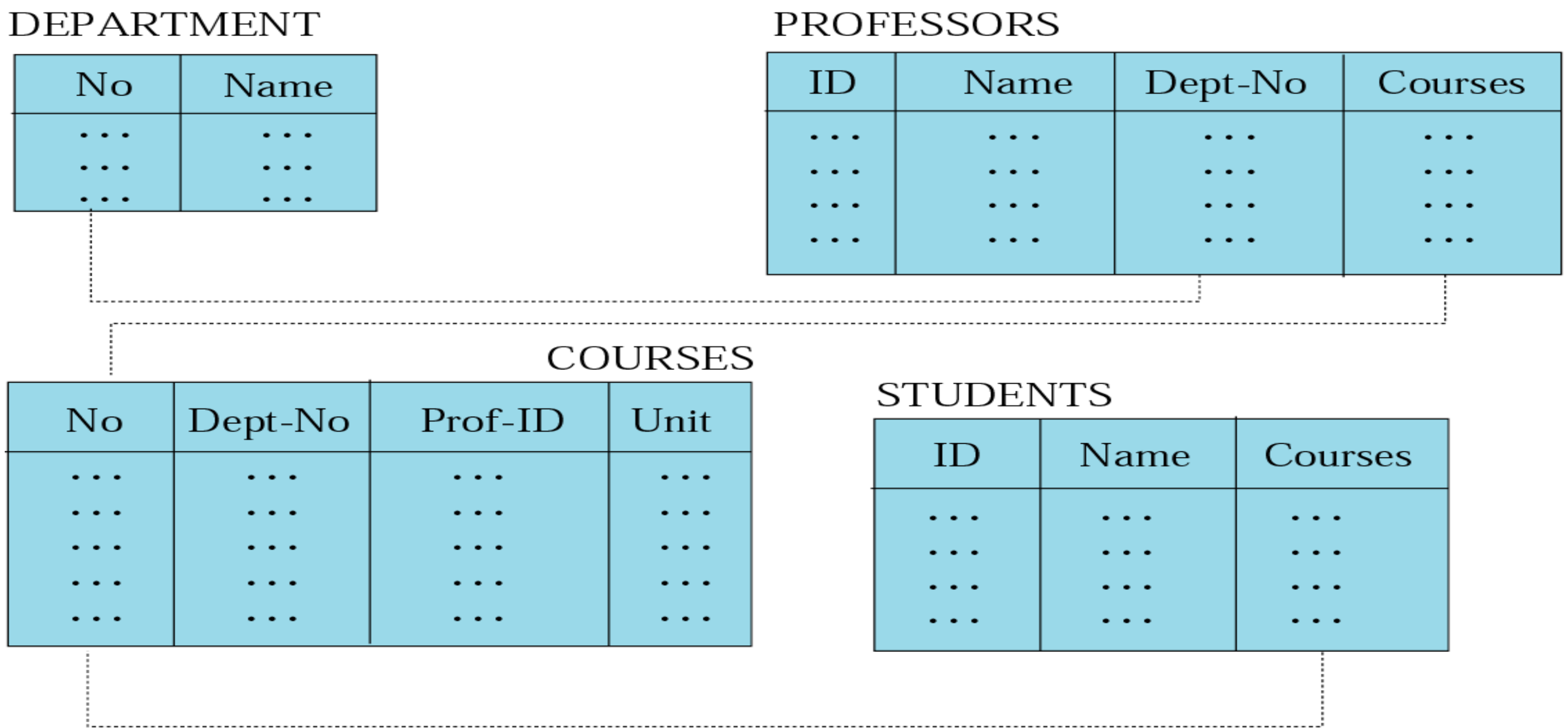
Network model

- The entities are organized in a graph.
- Some entities can be accessed through several paths.



Relational model

- Data are organized in two-dimensional tables called relations.
- The tables are related to each other.
- The most popular model. We mainly focus on relational model in this course



Relational model

- RDBMS (Relational Database Management System)
- external view
  - The data are represented as a set of relations.
  - A relation is a two-dimensional table.
- This doesn't mean that data are stored as tables; the physical storage of the data is independent of the way the data are logically organized.

Relation

- **Name** – each relation in a relational database should have a name that is unique among other relations.
- **Attribute** – each column in a relation.
  - The degree of the relation – the total number of attributes for a relation.
- **Tuple** – each row in a relation.
  - The cardinality of the relation – the total number of rows in a relation.

Example:

Fields (Attributes, Columns)

Field names

sid	Name	login	age	Gross
1111	Dave	dave@cs	19	1.2
2222	Jones	Jones@cs	18	2.3
333	Smith	smith@ee	18	3.4
4444	Smith	smith@math	19	4.5

Tuples (Records, Rows)

Students

Operations on relations

- In a relational database, we can define several operations to <u>create new relations</u> out of the existing ones.
- Basic operations:
  - Insert
  - Delete
  - Update
  - Select
  - Project
  - Join
  - Union
  - Intersection
  - Difference

Insert operation

- An unary operation.
- Insert a new tuple into the relation.

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5

Insert

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	*TCP/IP Protocols*	*6*

Delete operation

- An unary operation.
- Delete a tuple defined by a criterion from the relation.

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

Delete

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

Update operation

- An unary operation.
- Changes the value of some attributes of a tuple.

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

Update

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	**6**
CIS52	TCP/IP Protocols	6

Select operation

- An unary operation.
- It is applied to one single relation and creates another relation.
- The tuples in the resulting relation are a subset of the tuples in the original relation.
- Use some criteria to select

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

→ **Select** →

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS51	Networking	5

Project operation

- An unary operation.
- It is applied to one single relation and creates another relation.
- The attributes in the resulting relation are a subset of the attributes in the original relation.

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

Project

No	Unit
CIS15	5
CIS17	5
CIS19	4
CIS51	5
CIS52	6

Join operation

- A binary operation.
- Combines two relations based on common attributes.

COURSES

No	Course-Name	Unit
CIS15	Intro to C	5
CIS17	Intro to Java	5
CIS19	UNIX	4
CIS51	Networking	5
CIS52	TCP/IP Protocols	6

TAUGHT-BY

No	Professor
CIS15	Lee
CIS17	Lu
CIS19	Walter
CIS51	Lu
CIS52	Lee

Join

No	Course-Name	Unit	Professor
CIS15	Intro to C	5	Lee
CIS17	Intro to Java	5	Lu
CIS19	UNIX	4	Walter
CIS51	Networking	5	Lu
CIS52	TCP/IP Protocols	6	Lee

Union operation

- A binary operation.
- Creates a new relation in which each tuple is either in the first relation, in the second, or in both.
- The two relations must have the same attributes.

CIS15-Roster

Student-ID	F-Name	L-Name
145-67-6754	John	Brown
232-56-5690	George	Yellow
345-89-6580	Anne	Green
459-98-6789	Ted	Purple

CIS52-Roster

Student-ID	F-Name	L-Name
342-88-9999	Rich	White
145-67-6754	John	Brown
232-56-5690	George	Yellow

Union

Student-ID	F-Name	L-Name
145-67-6754	John	Brown
232-56-5690	George	Yellow
345-89-6580	Anne	Green
459-98-6789	Ted	Purple
342-88-9999	Rich	White

Intersection operation

- A binary operation.
- Creates a new relation in which each tuple is a member in both relations.
- The two relations must have the same attributes.

CIS15-Roster

Student-ID	F-Name	L-Name
145-67-6754	John	Brown
232-56-5690	George	Yellow
345-89-6580	Anne	Green
459-98-6789	Ted	Purple

CIS52-Roster

Student-ID	F-Name	L-Name
342-88-9999	Rich	White
145-67-6754	John	Brown
232-56-5690	George	Yellow

Intersection

Student-ID	F-Name	L-Name
145-67-6754	John	Brown
232-56-5690	George	Yellow

Difference operation

- A binary operation.
- Creates a new relation in which each tuple is in the first relation but not the second.
- The two relations must have the same attributes.

CIS15-Roster

Student-ID	F-Name	L-Name
145-67-6754	John	Brown
232-56-5690	George	Yellow
345-89-6580	Anne	Green
459-98-6789	Ted	Purple

CIS52-Roster

Student-ID	F-Name	L-Name
342-88-9999	Rich	White
145-67-6754	John	Brown
232-56-5690	George	Yellow

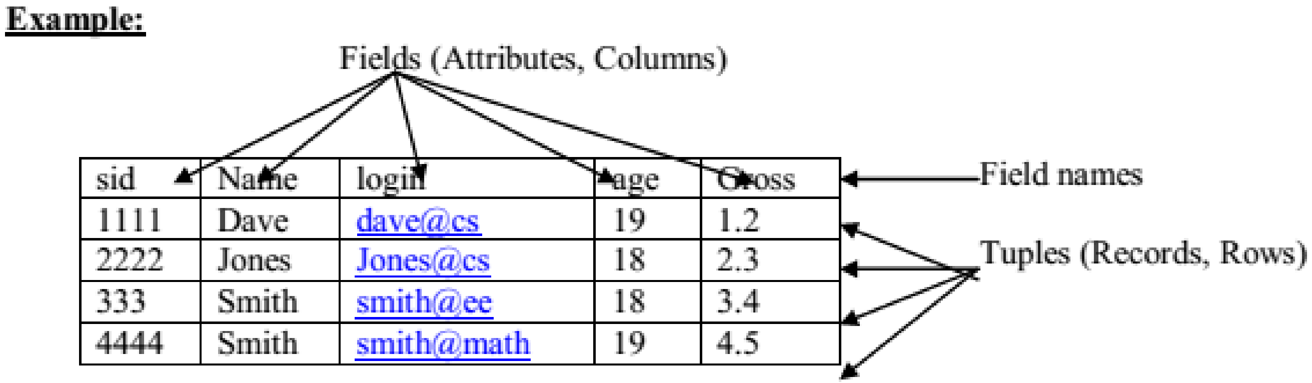
Difference

Student-ID	F-Name	L-Name
345-89-6580	Anne	Green
459-98-6789	Ted	Purple

Schema

- A description of data in terms of a data model is called a schema.
- In the relational model, the schema for a relation specifies
  - Relation name
  - The name of each field (or attribute or column)
  - The type of each field.

- Example: Student information in a university
  - Schema: Students(*sid:* string, *name:* string, *login:* string, *age:* integer, *gpa:* real)



sid	Name	login	age	Gross
1111	Dave	dave@cs	19	1.2
2222	Jones	Jones@cs	18	2.3
333	Smith	smith@ee	18	3.4
4444	Smith	smith@math	19	4.5

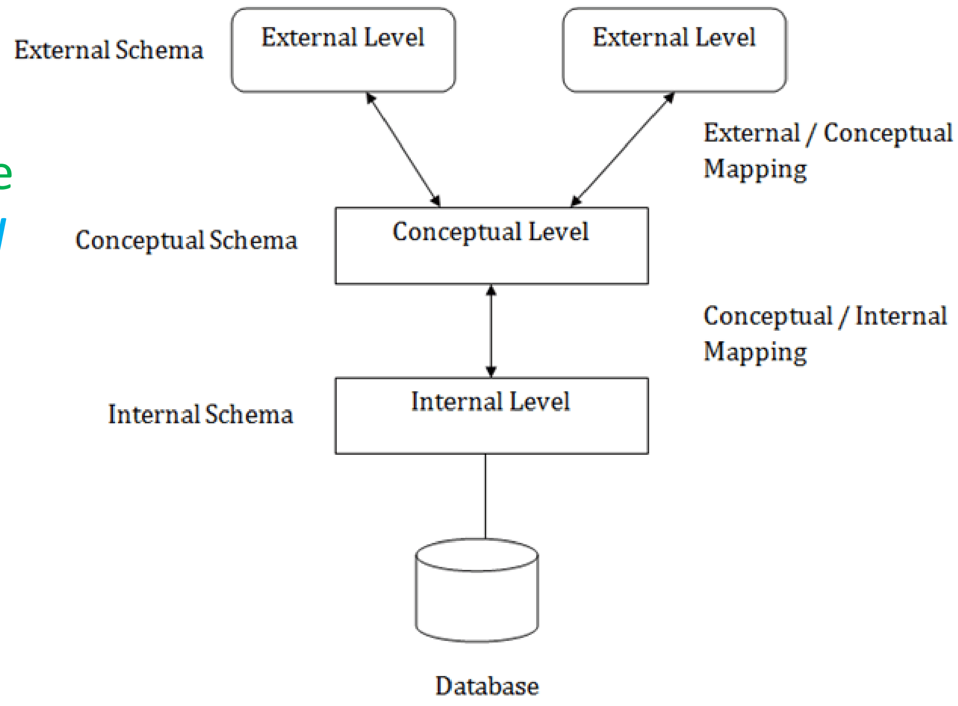
Levels of Abstraction in a DBMS

- The data in a DBMS is described at three levels of abstraction
  - *External schema: How users view the data*
    - *Eg: View number of enrollements in each course*

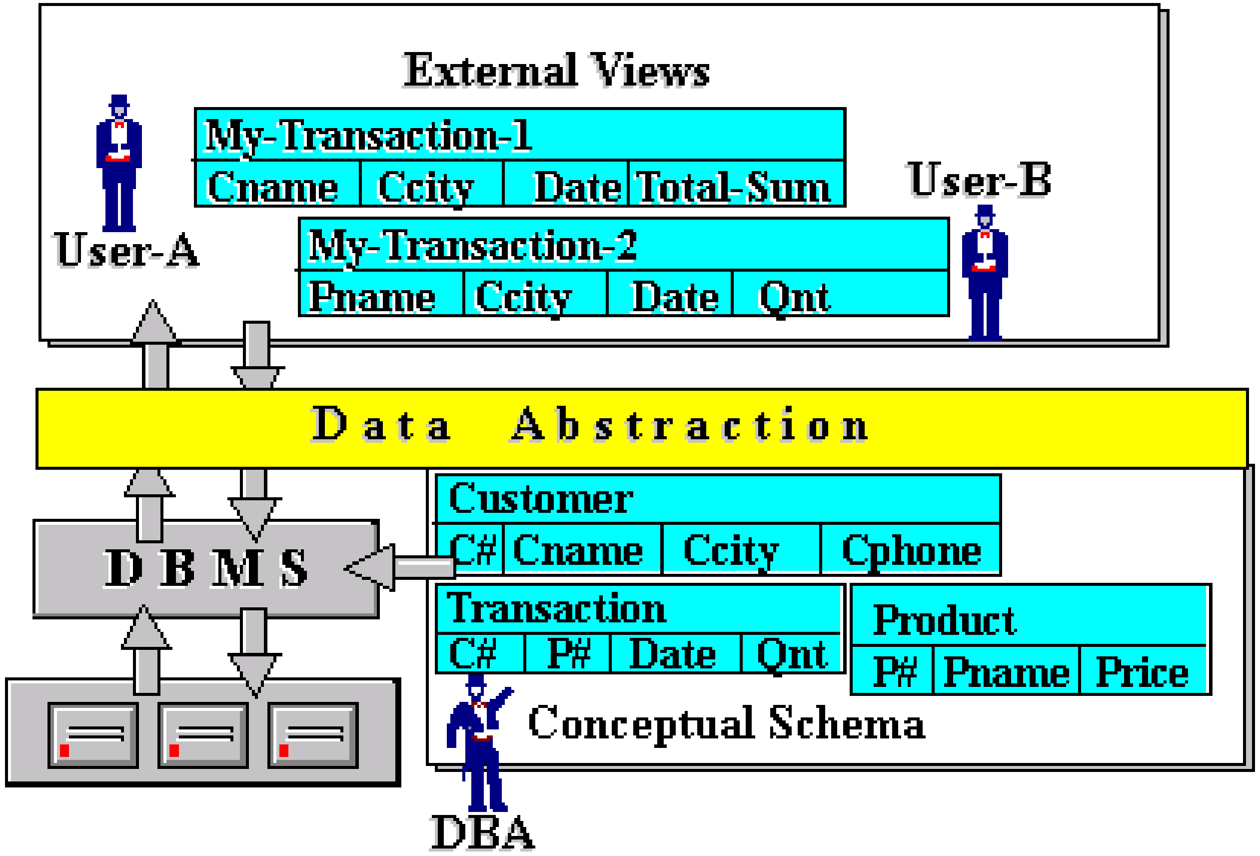
    Courseinfo(*cid:* string, *cname:* string, *enrollment:* intege
  - *Conceptual schema: Logical structure, tables and their attributes*
    - *Eg: Students (sid: string, name: string, …, gpa:real)*

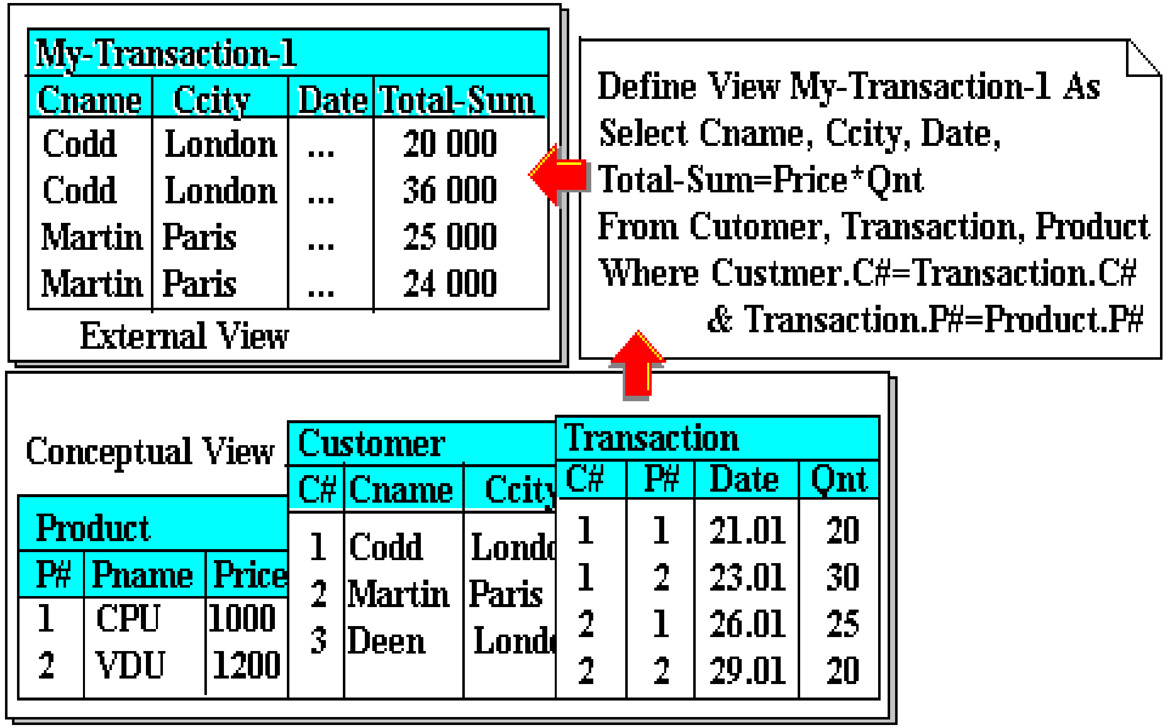
      Courses (cid: string, cname: string, credits: integer)
      Enrolled(sid: string, cid: string, grade: string)
  - *Physical schema: Storage methods and access methods*
    - *Relations stored as unordered files*
    - *Index on first column of students*
- A data definition language (DDL) is used to define the external and conceptual schemas.

https://www.javatpoint.com/dbms-three-schema-architecture



Example





http://coronet.iicm.edu/Dbase1/scripts/rdbh12.htm

Data Independence

- Change the Database schema at one level of a database system without requiring to change the schema at the next higher level.
  - **Physical Data Independence**: With Physical independence, you can easily change the physical storage structures or devices with no effect on the conceptual schema.
    - Any change done would be absorbed by the mapping between the conceptual and internal levels
    - Examples of internal level changes: Due to Physical independence, any these changes will not affect the conceptual layer.

      - Using a new storage device like Hard Drive or Magnetic Tapes
      - Modifying the file organization technique in the Database
      - Switching to different data structures.
      - Changing the access method, Modifying indexes.
      - Changes to compression techniques or hashing algorithms.
      - Change of Location of Database from say C drive to D Drive

  - **Logical Data Independence:** Logical Data Independence is the ability to change the conceptual scheme without changing External views or External programs.
    - Any change made will be absorbed by the mapping between external and conceptual level.
    - Examples of Conceptual level changes: Due to Logical independence, any of these changes will not affect the external layer.
      - Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs
      - Merging two records into one, - Breaking an existing record into two or more records

Transaction Management

- Scenario 1: Multiple customers accessing the available seat information in Airlines and trying to book.

- Scenario 2: Two transactions

T_1: Read(x)
$x \leftarrow x+100$
Write(x)
Commit

T_2: Read(x)
$x \leftarrow x-100$
Write(x)
Commit

Possible execution sequences:

T_1:	Read(x)
T_1:	$x \leftarrow x+100$
T_1:	Write(x)
T_1:	Commit
T_2:	Read(x)
T_2:	$x \leftarrow x-100$
T_2:	Write(x)
T_2:	Commit

T_1:	Read(x)	500
T_2:	Read(x)	500
T_1:	$x \leftarrow x+100$	600
T_1:	Write(x)	600
T_2:	$x \leftarrow x-100$	400
T_2:	Write(x)	400
T_1:	Commit	600
T_2:	Commit	400

Principles of Transaction

ATOMICITY

- all or nothing

CONSISTENCY

- no violation of integrity constraints

ISOLATION

- concurrent changes invisible $\Rightarrow$ serializable

DURABILITY

- committed updates persist

Before: X : 500	Y: 200
Transaction T	
T1	**T2**
Read (X) X: = X – 100 Write (X)	Read (Y) Y: = Y + 100 Write (Y)
After: X : 400	Y : 300

T	T''
Read (X) X: = X*100 Write (X) Read (Y) Y: = Y – 50 Write	Read (X) Read (Y) Z: = X + Y Write (Z)

Queries in a DBMS

- Here are some questions a user might ask from a university database system.
  - What is the name of the student with student ID 1234567
  - What is the average salary of professors who teach course CS5647
  - How many students are enrolled in CS5647
  - What fraction of students in CS564 received a grade better than B7
  - Is any student with a CPA less than 3.0 enrolled in CS5647
- Such questions involving the data stored in a DBMS are called queries.
  - A DBMS provides a specialized language, called the query language, in which queries can be posed. Eg: SQL (structured query language)
- A DBMS enables users to create, modify, and query data through a data manipulation language (DML).

References

- Adhsakkdi Y Raghuram Krishnan and Johannes Gehrke, Database Management Systems, 3/e, TMH, 2007.
- Referred various online sources for examples and images.