Topic 2

Database System Architecture

From

Chapters 1 and 2 of Fundamentals of Database Systems, Authors: Elmasri and Navathe

Publisher: Addison Wesley - Pearson

By: Abdolreza Abhari

CPS510

Ryerson University

Topics in this Section

- Three levels of architecture
- Mappings
- Database Administrator (DBA)
- Database Management System (DBMS)
- Database Communications
- Client/Server Architecture
- Utilities
- Distributed Processing

Data Modeling: Schemas and Instances

- Before start to talk about database architecture note that in any data model, it is important to distinguish between
  - » description of the database (database ***schema***)
  - » database itself (***instance*** of a database)
- Database schema
  - * Describes the database
  - * Specified during the database design phase
    - » Not expected to change frequently
  - * Most data models have a notation for graphical representation of schema

Data Modeling: Schemas and Instances

Example schema: SUPPLIER-PARTS database

SUPPLIER

supplier#	supplier_name	city

PART

part#	part_name	weight

PARTS_SUPPLIER

part#	supplier#	quantity

Data Modeling: Schemas and Instances

- Database instance
  - » Refers to the data in the database at a particular moment in time
  - » Many database instances can correspond to a particular schema
  - » Every time we insert, delete, update the value of a data item, we change one instance of database to another
  - » DBMS is partially responsible for ensuring that every instance satisfies
    - – Structure and constraints specified in the database schema
  - » See the example instance of SUPPLIER-PARTS database shown before

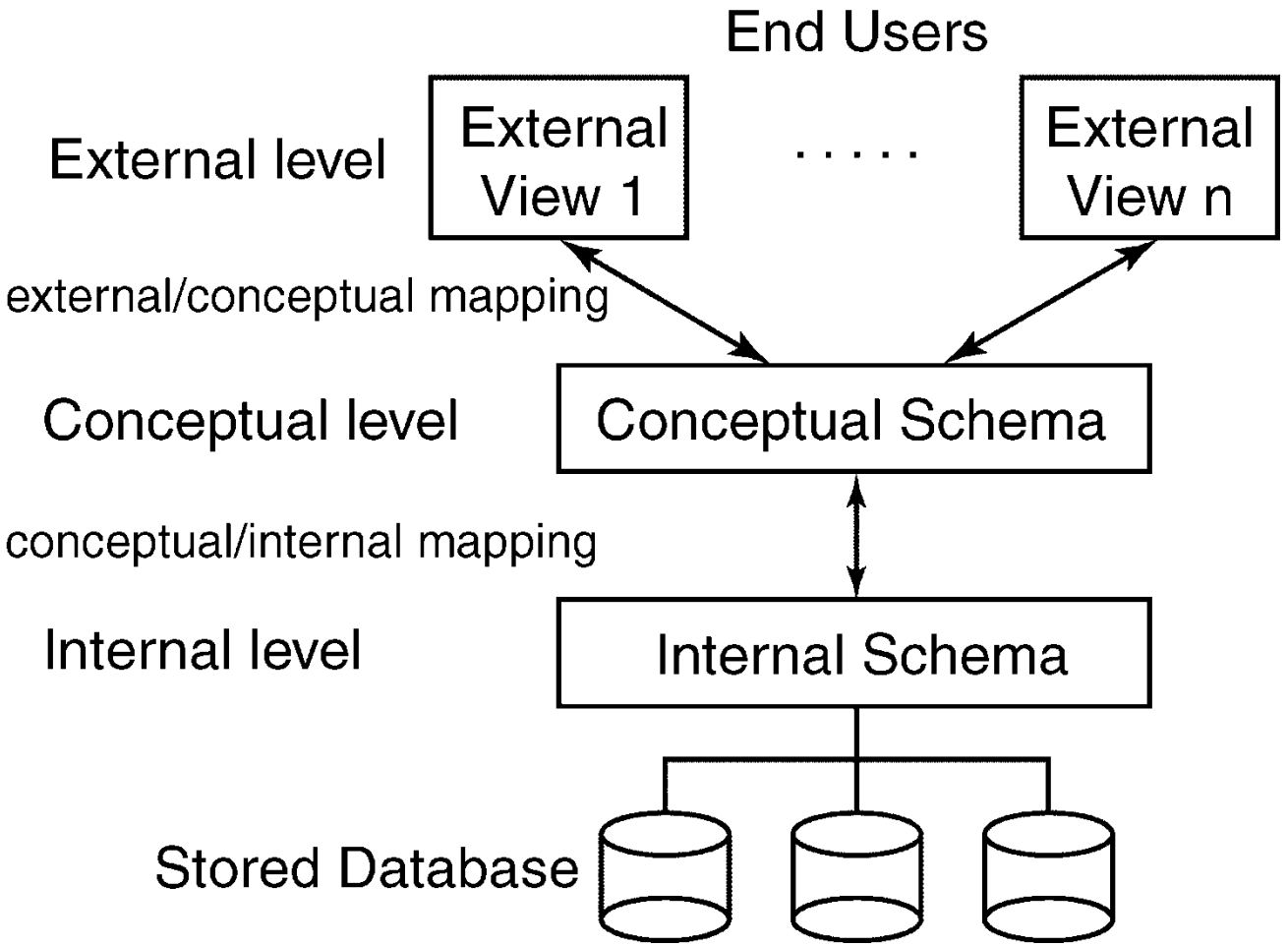
Three Levels of Architecture

- Three level architecture is also called ANSI/SPARC architecture or three schema architecture
- This framework is used for describing the structure of specific database systems (small systems may not support all aspects of the architecture)
- In this architecture the database schemas can be defined at three levels explained in next slide

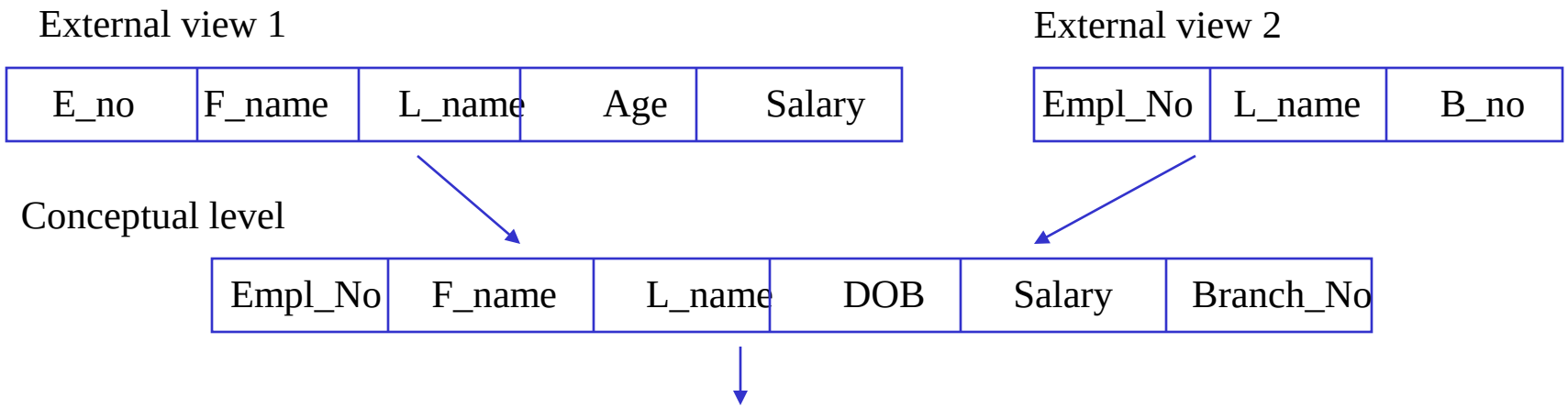
Three Levels of Architecture

- Internal level: Shows how data are stored inside the system. It is the closest level to the physical storage. This level talks about database implementation and describes such things as file organization and access paths. Note that relational model has **nothing** explicit to say regarding the internal level
- Conceptual level: Deals with the modeling of the whole database. The conceptual schema of database is defined in this level
- External level: This level models a user oriented description of part of the database. The views for individual users are defined by means of external schemas in this level

Three Levels of Architecture



Three Levels of Architecture-Example



Internal level

```
struct EMPLOYEE {
    int Empl_No;
    int Branch_No;
    char F_name [15];
    char L_name [15];
    struct date Date_of_Birth;
    float Salary;
    struct EMPLOYEE *next;       //pointer to next employee record
    };   index Empl_No; index Branch_No;      //define indexes for  employees
```

Mapping

- Mapping is the key for providing data independence. Here is more explanation on data independence based on three-level architecture.
- Data independence is the capacity to change the schema at one level without having to change the schema at the next higher level
- Two types of data independence are
  - Logical data independence
  - Physical data independence

Mapping - Data Independence

- Logical data independence (provided by external/ conceptual mapping)
  - Ability to modify conceptual schema without changing
    - External views
    - Application programs
  - Changes to conceptual schema may be necessary
    - Whenever the logical structure of the database changes
      - ✈Due to changed objectives
  - Examples
    - » Adding a data item to schema
      - Adding price of a part to PART table
    - » Adding PROJECT table to the SUPPLIER-PARTS database

Mapping - Data Independence

- Physical data independence (provided by conceptual/internal mapping)
  - Ability to modify internal or physical schema without changing
    - Conceptual or view level schema
    - Application programs
  - Changes to physical schema may be necessary to
    - Improve performance of retrieval or update
      - Example: Adding a new index structure on ***`city`***
- Achieving logical data independence is more difficult than physical data independence
  - Because application programs heavily rely on the logical structure of the data they access

Database Administrator

- Participates in conceptual database design
- Determines how to implement conceptual schema
- Teach users, and help them report
- Implement security and integrity
- Implement unload/reload utilities
- Monitor and tune database performance

DBMS (Languages)

- Users interact with database with data sublanguage (embedded within a host language) which consists of at least two types of languages
  - DDL: To define the database
    - Used to define the database
      - ✈ For defining schemas at various levels
    - Required in building databases
    - DBA and database designers are typical users
    - Commonly referred to as ***data definition language***
  - DML: To manipulate data
    - Used to construct and use the database
      - ✈ Facilitates retrieval, insertion, deletion and updates
    - Typical users are “End Users”
    - Referred to as ***data manipulation language***

Database Management System

- DDL processor / compiler
- DML processor / compiler
- Handle scheduled and *ad hoc* queries
- Optimizer and run-time manager
- Security and integrity
- Recovery and concurrency
- Data dictionary :The **data dictionary** is a system database that contains "data about the data“. That is *definitions* of other objects in the system, also known as *metadata*
- Performance tuning utilities

Support for System Processes

- Data Communications interface
- Client Server Architecture
- External tool support: query, reports, graphics, spreadsheets, statistics
- Utilities: unload/reload, stats, re-org
- Distributed processing