



IT-212

Database Systems

Lecture-02

Dr. Muhammad Asif Khan
Assistant Professor

Department of Computer Science & IT
Sarhad University of Science and Information Technology, Peshawar

Last Lecture Summary

- Some common uses of database systems
- Characteristics of file-based systems
- Problems with file-based approach
- The Database (DB)
- Database Management System (DBMS)
- Typical functions of a DBMS
- Major components of the DBMS environment
- Personnel involved in the DBMS environment
- Advantages and disadvantages of DBMSs
- History of the development of DBMSs

Today's Topics

- Data Models and Their Categories
- History of Data Models
- Schemas, Instances, and States
- Structure of Relational Databases
- Database Schema
- ANSI-SPARC Three-Level Architecture
- Schemas (Three-Level Architecture)
- Mappings
- Data Independence

IT-212: Database Systems 3



Data Models

- **Data Model:**
 - A set of concepts to describe the *structure* of a database, – the *operations* for manipulating these structures, and – certain *constraints* that the database should obey.
- **Data Model Structure and Constraints:**
 - Constructs are used to define the database structure
 - Constructs typically include *elements* (and their *data types*) as well as groups of elements (e.g. *entity*, *record*, *table*), and *relationships* among such groups
 - Constraints specify some restrictions on valid data;
 - Constraints must be enforced at all times

IT-212: Database Systems 4



Categories of Data Models

- **Conceptual (high-level, semantic) data models:**
 - Provide concepts that are close to the way many users perceive data.
 - (Also called *entity-based* or *object-based* data models.)
- **Physical (low-level, internal) data models:**
 - Provide concepts that describe details of how data is stored in the computer.
- **Implementation (representational) data models:**
 - Provide concepts that fall between the above two, used by many commercial DBMS implementations (e.g. relational data models used in many commercial systems).

IT-212: Database Systems 5



Schemas and Instances

- **Database Schema:**
 - The *description* of a database.
 - Includes descriptions of the database structure, data types, and the constraints on the database.
- **Schema Diagram:**
 - An *illustrative* display of (most aspects of) a database schema.
- **Schema Construct:**
 - A *component* of the schema or an object (entity) within the schema, e.g., STUDENT, COURSE.
- **Instance:** The collection of data and information that the database stores at any particular moment.
 - The term *instance* is also applied to individual database components, e.g. *record instance*, *table instance*, *entity instance*

IT-212: Database Systems 6



Example of a Database Schema

STUDENT

Name	Student_number	Class	Major
------	----------------	-------	-------

COURSE

Course_name	Course_number	Credit_hours	Department
-------------	---------------	--------------	------------

PREREQUISITE

Course_number	Prerequisite_number
---------------	---------------------

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
--------------------	---------------	----------	------	------------

GRADE_REPORT

Student_number	Section_identifier	Grade
----------------	--------------------	-------

Book: Ramez Elmasri and Shamkant B. Navathe

Figure 2.1

Schema diagram for the database in Figure 1.2.

Schema: instructor (ID, name, dept_name, salary)

ID	name	dept_name	salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	King	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califeri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Book: Silberschatz, Korth and Suda

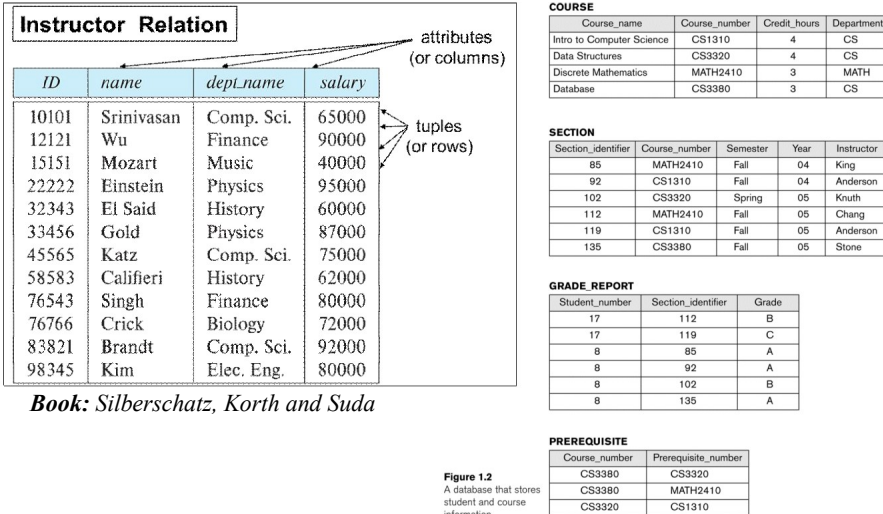
Database State, Distinction, Intension, Extension

- **Database State:**
 - The actual data stored in a database at a *particular moment in time*. – This includes the collection of all the data in the database.
- **Initial Database State:**
 - Refers to the database state when it is initially loaded into the system.
- **Valid State:**
 - A state that satisfies the structure and constraints of the database.
- **Distinction**
 - The *database schema* changes very infrequently.
 - The *database state* changes every time the database is updated. •

Schema is also called **intension**.

- **State** is also called **extension**.

Example of a database state

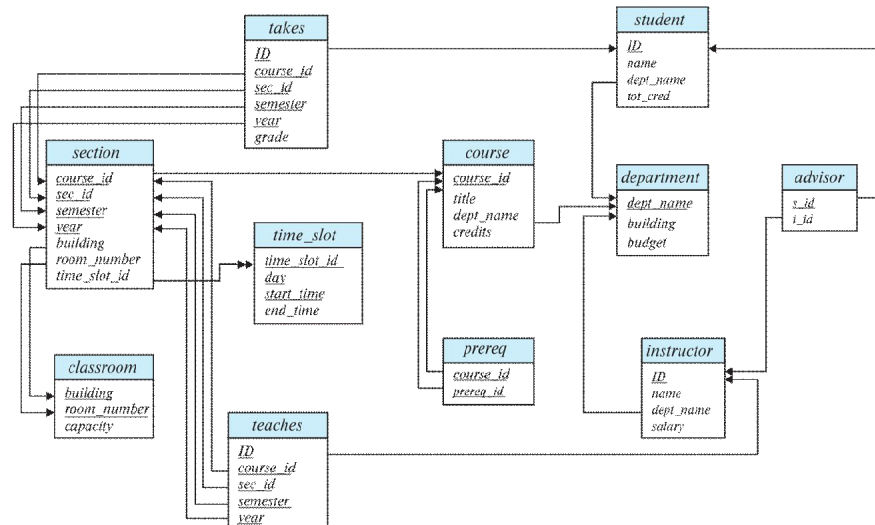


Book: Silberschatz, Korth and Suda

Book: Ramez Elmasri and Shamkant B. Navathe

IT-212: Database Systems 9

Schema Diagram for University Database



Book: Silberschatz, Korth and Suda

IT-212: Database Systems

10

Three-Level / Schema Architecture

Objectives:

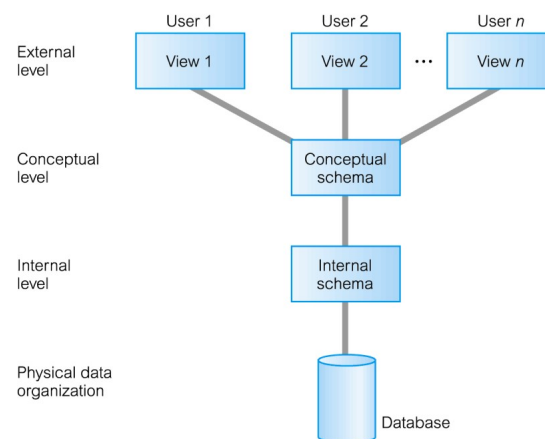
- All users should be able to access same data but have a different customized view.
- A user's view is protected from changes made in other views– Users should not need to know physical database storage details.
- DBA should be able to change database storage structures without affecting the users' 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

IT-212: Database Systems

11

ANSI-SPARC Three-Level Architecture (1)

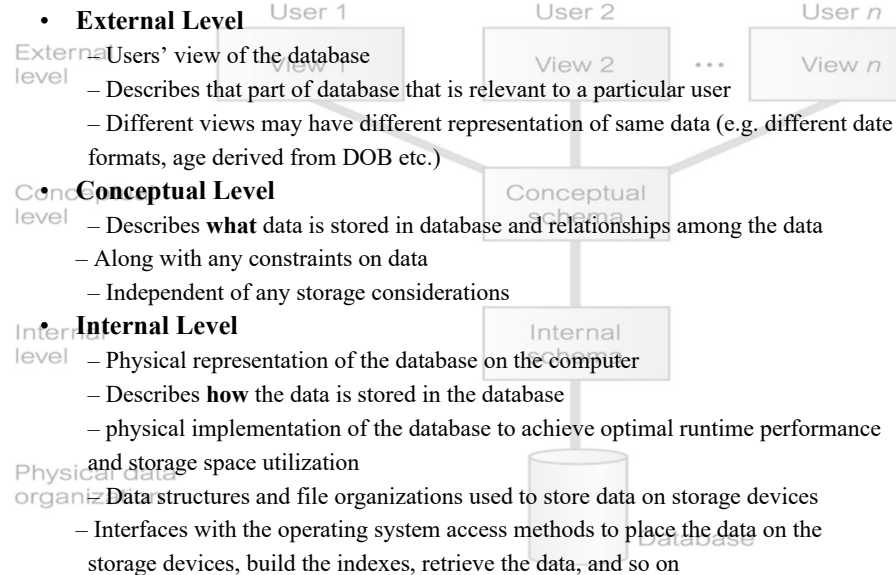
- **ANSI-SPARC:** American National Standards Institute, Standards Planning And Requirements Committee
 - An abstract design standard for a DBMS, first proposed in 1975.



IT-212: Database Systems

12

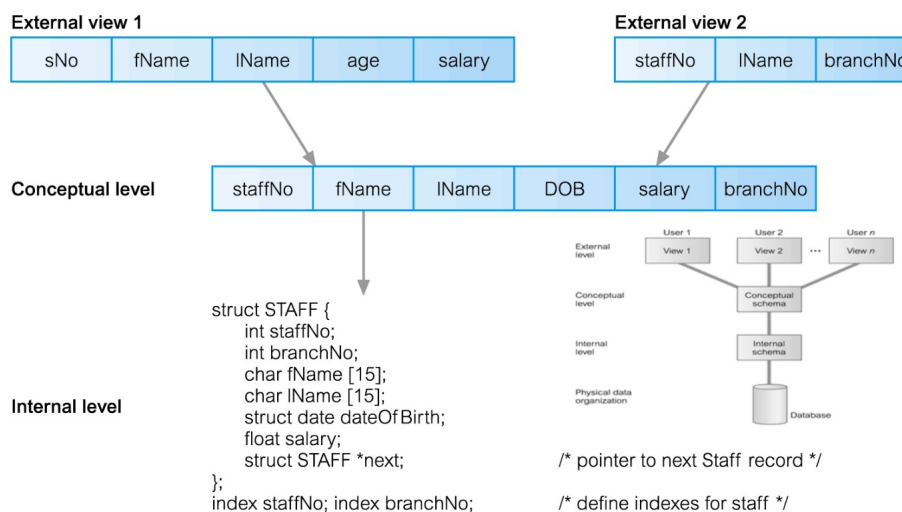
ANSI-SPARC Three-Level Architecture (2)



IT-212: Database Systems

(13)

ANSI-SPARC Three-Level Architecture (3)



IT-212: Database Systems

(14)

Schemas (Three-Level Architecture)

- External Schemas
 - Also called subschemas
 - Multiple schemas per database
 - Corresponds to different views of data
- Conceptual Schema
 - Describes all the entities, attributes, and relationships together with integrity constraints
 - Only one schema per database
- Internal Schema
 - A complete description of the internal model, containing the definitions of stored records, the methods of representation, the data fields, and the indexes and storage structures used
 - Only one schema per database

IT-212: Database Systems

15

Mappings (1)

- The DBMS is responsible for mapping between these three types of schema:
 - The DBMS must check that each external schema is derivable from the conceptual schema, and it must use the information in the conceptual schema to map between each external schema and the internal schema
- Types of mappings
 - Conceptual / Internal mapping
 - External / Conceptual mapping

IT-212: Database Systems

16

Mapping (2)

Conceptual / Internal Mapping

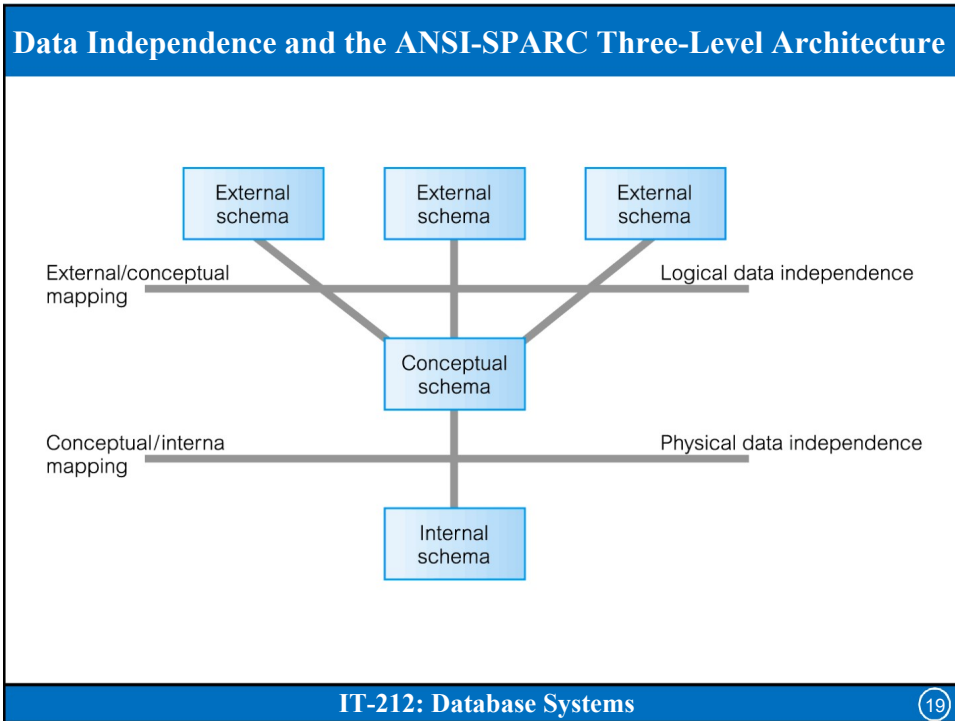
- Enables the DBMS to
 - Find the actual record or combination of records in physical storage that constitute a logical record in the conceptual schema,
 - Together with any constraints to be enforced on the operations for that logical record
 - It also allows any differences in entity names, attribute names, attribute order, data types, and so on, to be resolved

External / Conceptual Mapping

- Enables the DBMS to
 - Map names in the user's view on to the relevant part of the conceptual schema

Data Independence

- Logical Data Independence
 - Refers to immunity (freedom) 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
- 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, storage devices etc.)
 - Should not require change to conceptual or external schemas



Summary

- Data Models and Their Categories
- History of Data Models
- Schemas, Instances, and States
- Structure of Relational Databases
- Database Schema
- ANSI-SPARC Three-Level Architecture
- Schemas (Three-Level Architecture)
- Mappings
- Data Independence

IT-212: Database Systems (20)

THANK YOU