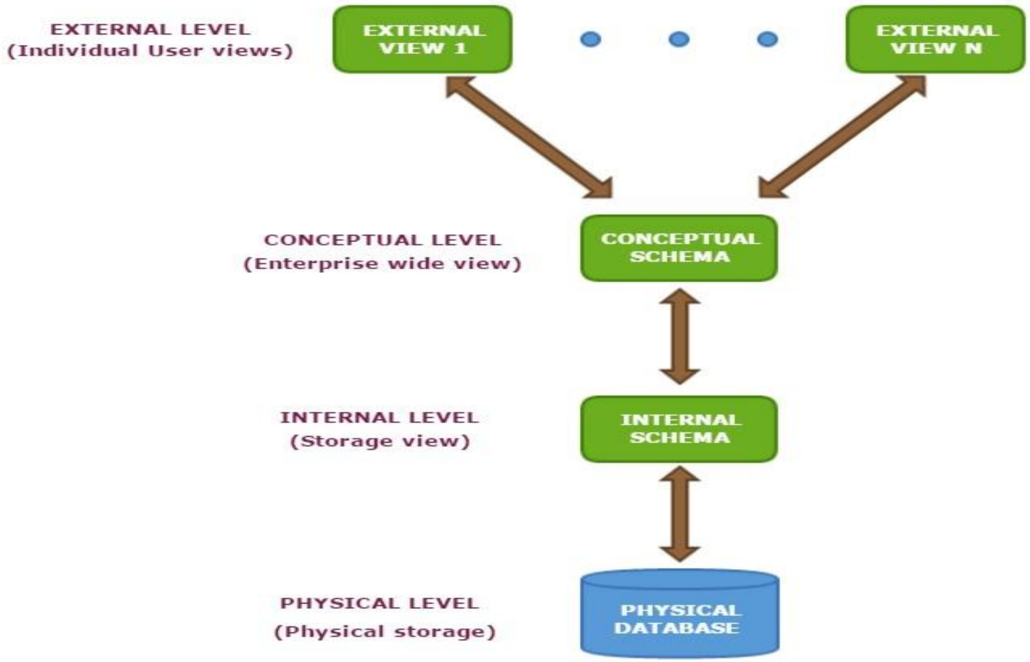
University of Bisha

Faculty Of Computing and Information Technology

Part (2)

Database Architecture & Design



2020

Outlines of Part 2 → use Chapter 2

- Introduction
- Database Architecture
- Data Abstraction
- Physical and Logical Data independency

1. Introduction

- **Data Model:**

Is a collection of concepts that can be used to describe the ***structure of a database.***

- **Structure of a database :** *means the data types, relationships*, and constraints that apply to the data.

- Most data models also include a set of basic **operations** for specifying *retrievals* and *updates* on the database.

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. by representing information such as record formats, record orderings, and access paths (**index**).

Categories of Data Models

- **Representational (implementation) data models:**

- Provide concepts that fall between the above two which provide concepts that may be easily understood by end users but that are not too far removed from the way data is organized in computer storage.

- (e.g. relational data models used in many commercial systems – Network & hierarchical models).

- Representational data models represent data by using record structures and hence are sometimes called **record-based data models.**

- **Database Schema:**
  - Is the ***description*** of a database.
  - Includes descriptions of the database structure, data types, and the constraints on the database.
- It is a set of relation schemes S = {R1, R2, . . ., Rn, } and a set on integrity constraints
- **Example of database schema:**

Employee (SSN, Name, Address, Salary, DepN)

Dept (Dep#, Name, MGRSSN)

DepLct (Dep#, DepLocation)

Prjct(Proj#, Name, ProjLoc, Dep#)

Works_on (SSN, Prj#, firs)

Schemas

Schema Diagram:

- An ***illustrative*** display of (most aspects of) a database schema.
- Figure 2.1 shows a schema diagram for the database shown in Figure 1.2

Figure 2.1
Schema diagram for the database in Figure 1.2.

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

Schema Construct:

- A ***component*** of the schema or an object within the schema, e.g., STUDENT, COURSE.

The DBMS stores the descriptions of the schema constructs and constraints—also called the **meta-data—in the DBMS catalog so** that DBMS software can refer to the schema whenever it needs to.

State

- **Database State** or **snapshot** :
  - The actual data stored in a database at a ***particular moment in time***. This includes the collection of all the data in the database.
  - Also called the **current set of instances (or occurrence).**
    - The term *instance* is also applied to individual database components, e.g. *record instance, table instance, entity instance.*

- **Initial Database State:**
  - Refers to the database state when it is initially loaded into the system.

1. Introduction…..Cont
Example of a database State

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

1. Introduction…..Cont
Database Schema vs. Database State

- Distinction
  - The ***database schema*** changes very infrequently.
  - The ***database state*** changes every time the database is updated.

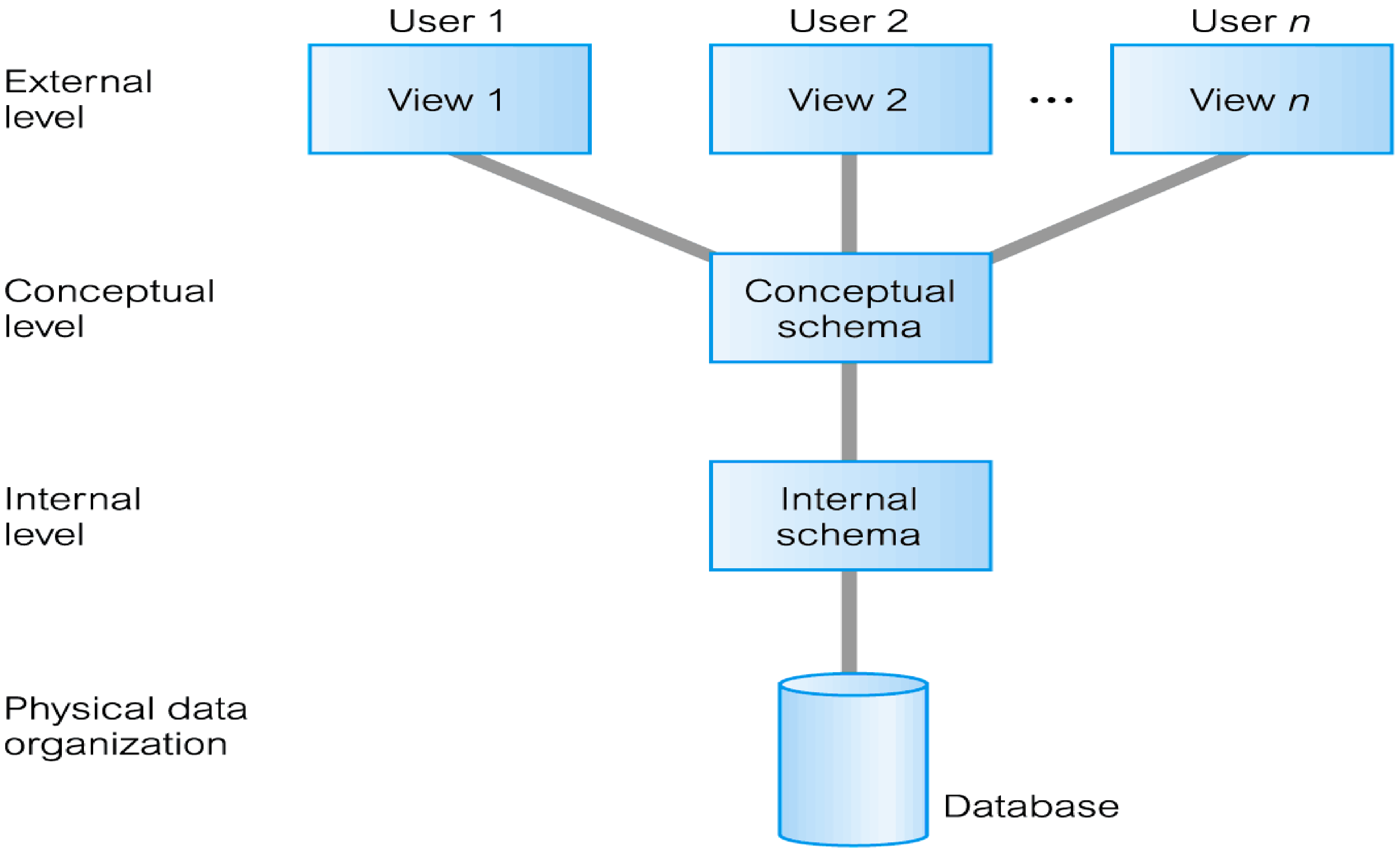
- **Schema** is also called **intension**.
- **State** is also called **extension.**

2. Database Architecture

- The American National Standards Institute (ANSI) produced a three level of database architecture (**external, a conceptual, and an internal level**) in 1975

- The **conceptual level provides both the mapping and the desired independence** between the external and internal levels.

Database Environment - ANSI-SPARC Three Schema-Level Architecture



Objectives of Three-Level Architecture

- All users should be able to access same data.
- Database storage structures can be changed without affecting the users' views.
- Conceptual structure of database can be changed without affecting all users.

ANSI-SPARC Three-Level Architecture

Three-Schema Architecture defines DBMS schemas at ***three*** levels **:**

- **Internal schema** at the internal level to describe physical storage structures of the database.

  - It describes the complete details of data storage (how the data is stored in the database -tree) and access paths for the database (e.g indexes).

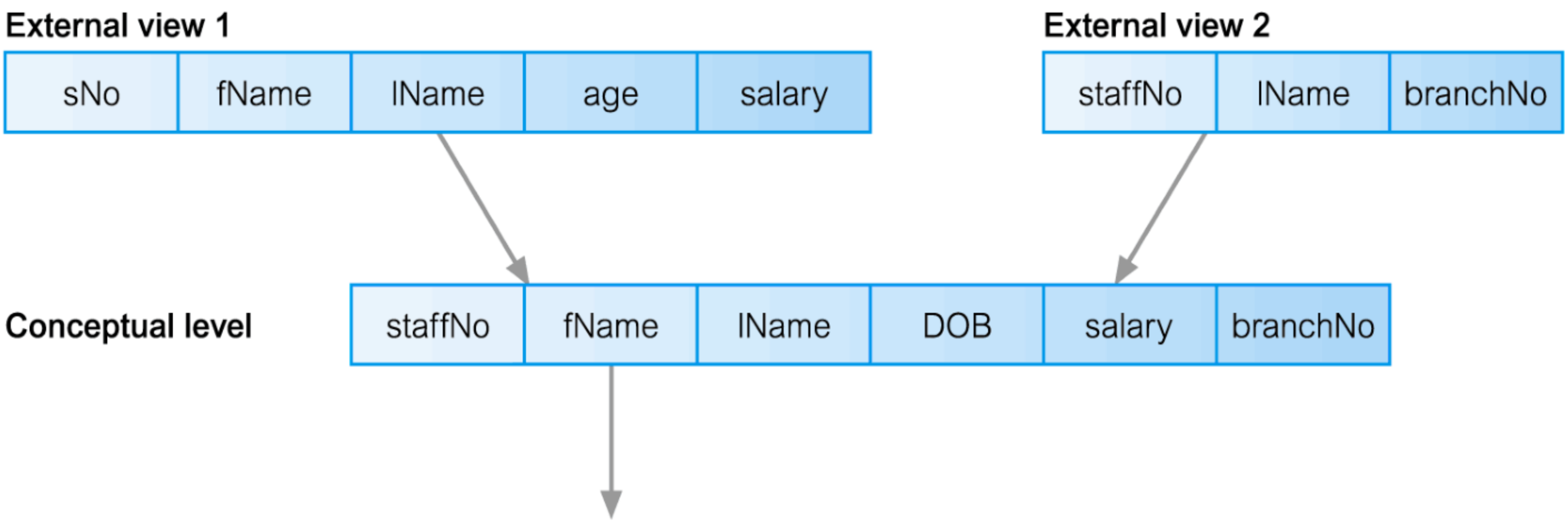
  - Typically uses a **physical** data model.

ANSI-SPARC Three-Level Architecture

- **Conceptual schema** at the conceptual level to describe the structure (all entities, their attributes, and their relationships) and constraints for the whole database for a community of users.
  - Uses a **conceptual** or an **implementation** data model.

- **External schemas** at the external level to describe the various user views.
  - Usually uses the same data model as the conceptual schema.

Differences between Three Levels of ANSI-SPARC Architecture



Internal level

```
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 */
```

3- Data Abstraction

- One fundamental characteristic of the database approach is that it provides some level of data abstraction.

- **Data abstraction:** generally refers to the suppression of details of data organization and storage, and the highlighting of the essential features for an improved understanding of data.

- data abstraction enables different users to perceive data at their preferred level of detail.(ex. DBA & users).

4- Physical and Logical Data independency

- A major objective for the three-level architecture is to provide **data independence,** which means that upper levels are unaffected by changes to lower levels.

- There are two kinds of data independence: ***physical*** **and** ***logical*.**

4- Physical and Logical Data independency

- **Physical Data Independence:**
  - The capacity to change the internal schema without having to change the conceptual schema.
- Changes to the internal schema, such as using different file organizations or storage structures, using different storage devices, modifying indexes, should be possible without having to change the conceptual or external schemas.
- From the users' point of view, the only effect that may be noticed is a change in performance.
- In fact, weakening in performance is the most common reason for internal schema changes.

4- Physical and Logical Data independency

- **Logical Data Independence:**

  The capacity to change the conceptual schema without having to change the external schemas and their associated application programs.

- Changes to the conceptual schema, such as the addition or removal of new entities, attributes, or relationships.

- the users for whom the changes have been made need to be aware of them, but what is important is that other users should not be.

4- Physical and Logical Data independency …Cont

- When a schema at a lower level is changed, only the mappings between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence.

- The higher-level schemas themselves are unchanged.
  - Hence, the application programs need not be changed since they refer to the external schemas.

- Figure 2.3 illustrates where each type of data independence occurs in relation to the three-level architecture.

4- Physical and Logical Data independency …Cont

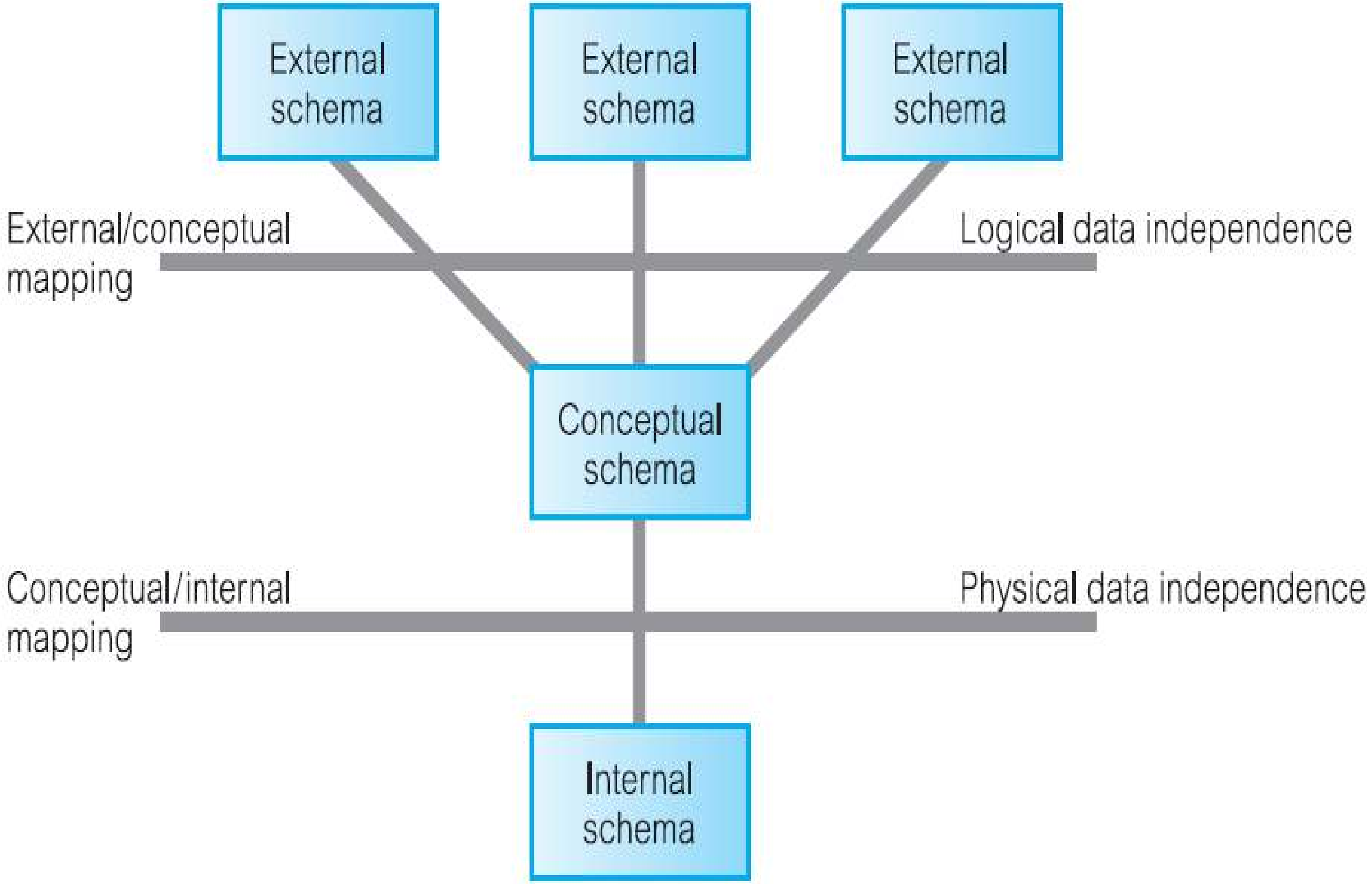


Figure 2.3 Data independence and the ANSI-SPARC three-level architecture.

Thank you