



Basic Database Concepts

Hafiz Tayyeb Javed

Assistant Professor(CS)

Department of Computer Science

FAST-NU, CFD Campus



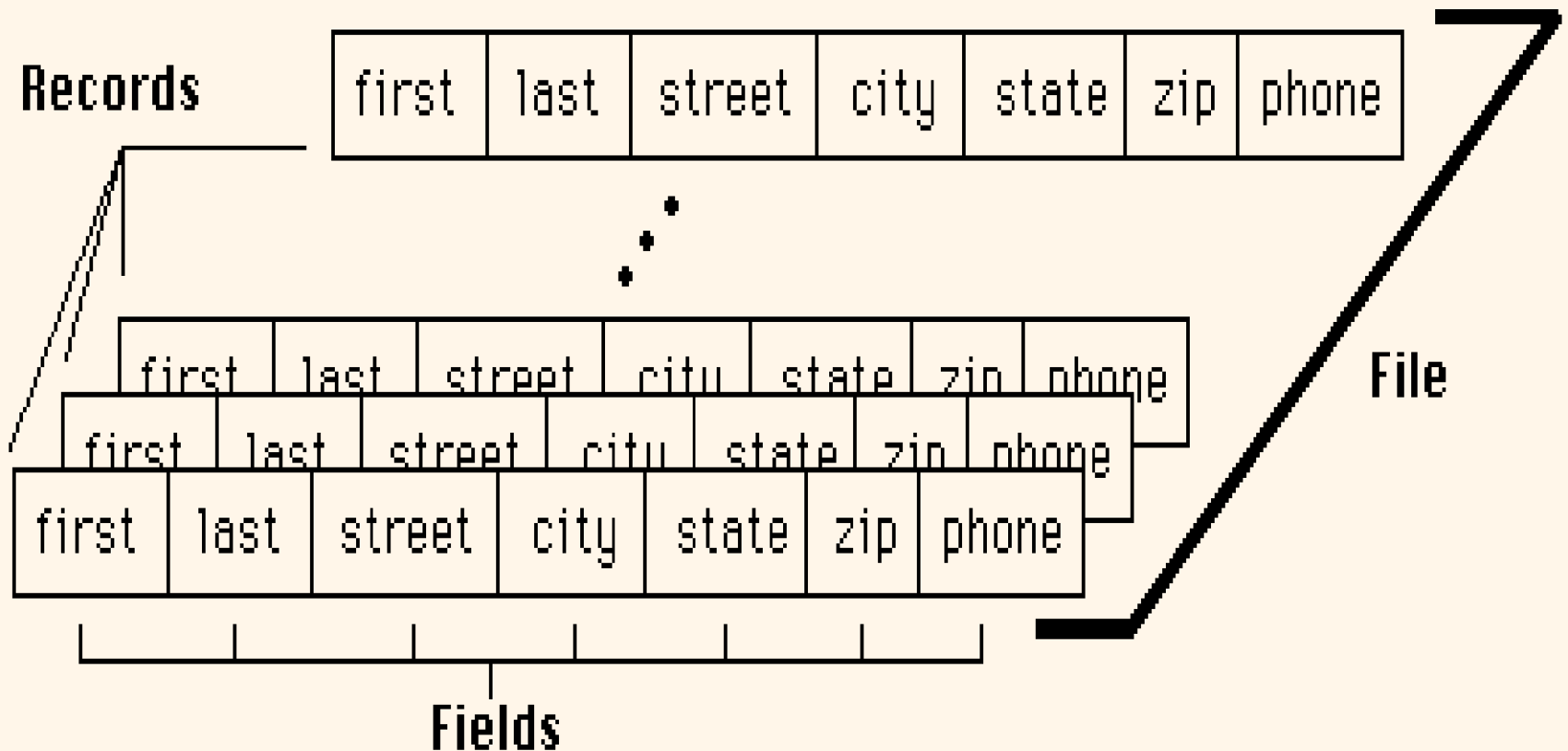


Objectives



- Basic Concepts & Architecture
- History
- DBMS Benefits
- DBMS Abstraction
- Database Schema and State

Files, Records, and Fields



A light blue diamond shape with a thin black border.

History of Database Systems

- **First-generation**
 - Hierarchical and Network
- **Second generation**
 - Relational
- **Third generation**
 - Object Relational
 - Object-Oriented

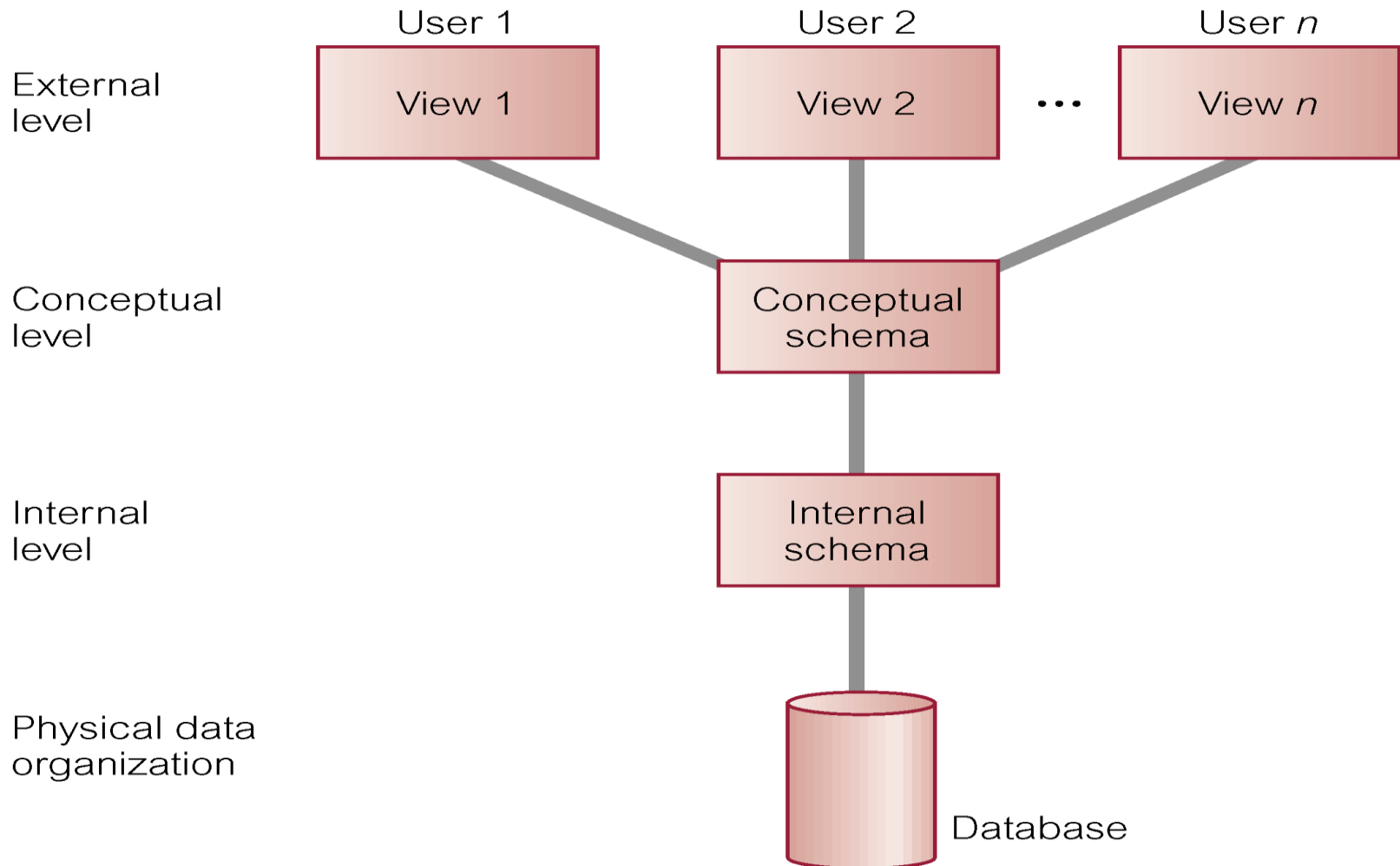


Terms and Concepts

Data Independence

- Physical representation and location of data and the use of that data are separated
 - The application doesn't need to know how or where the database has stored the data, but just how to ask for it
 - Moving a database from one DBMS to another should not have a material effect on application program
 - Recording, adding fields, etc. in the database should not affect applications

ANSI-SPARC Three-Level Architecture





Objectives of Three-Level Architecture



- All users should be able to access same data
- The user's view is immune to change made in other views
- User 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 user's 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

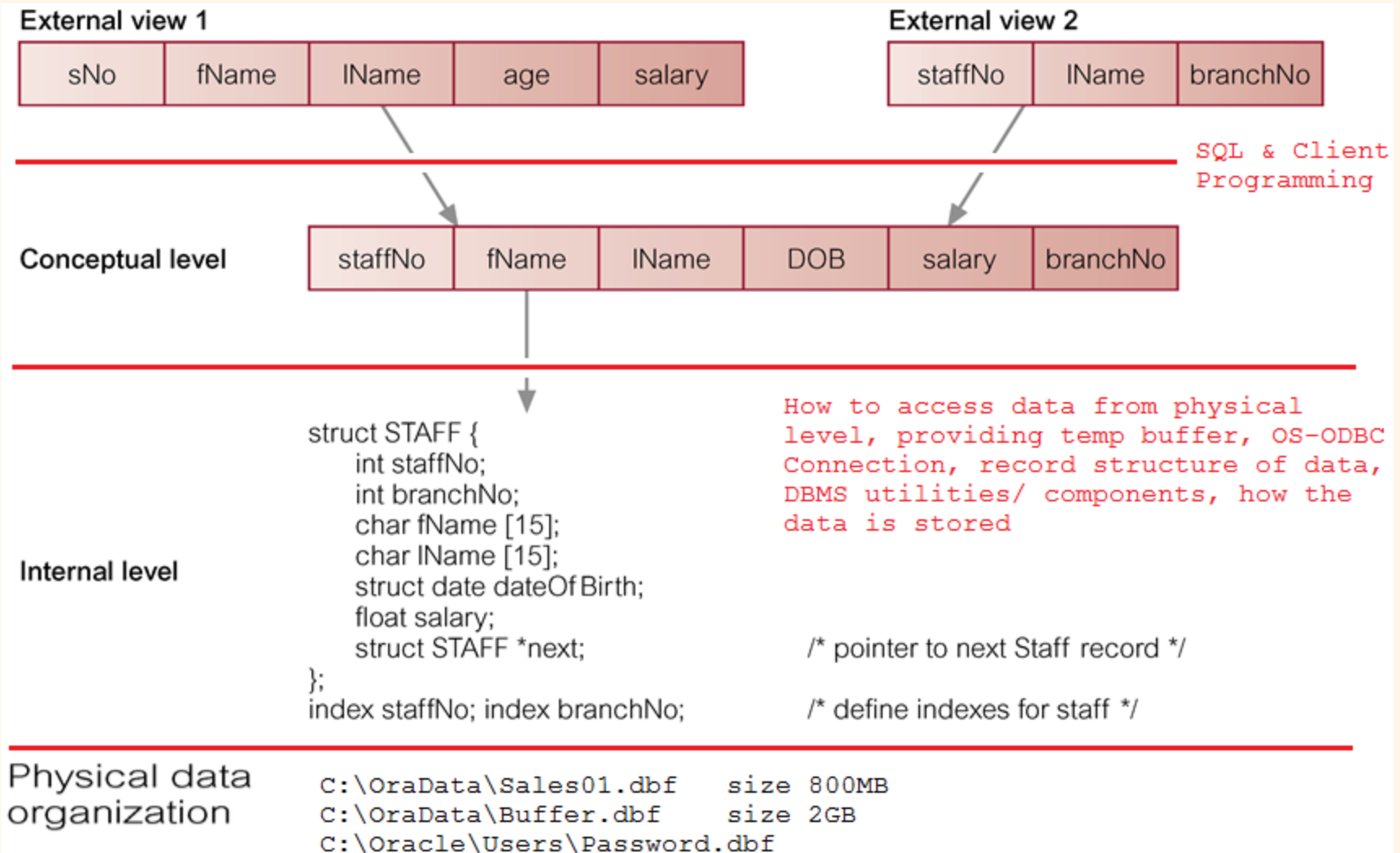


ANSI-SPARC Three-Level Architecture



- **External Level**
 - User's 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
- **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 - Example



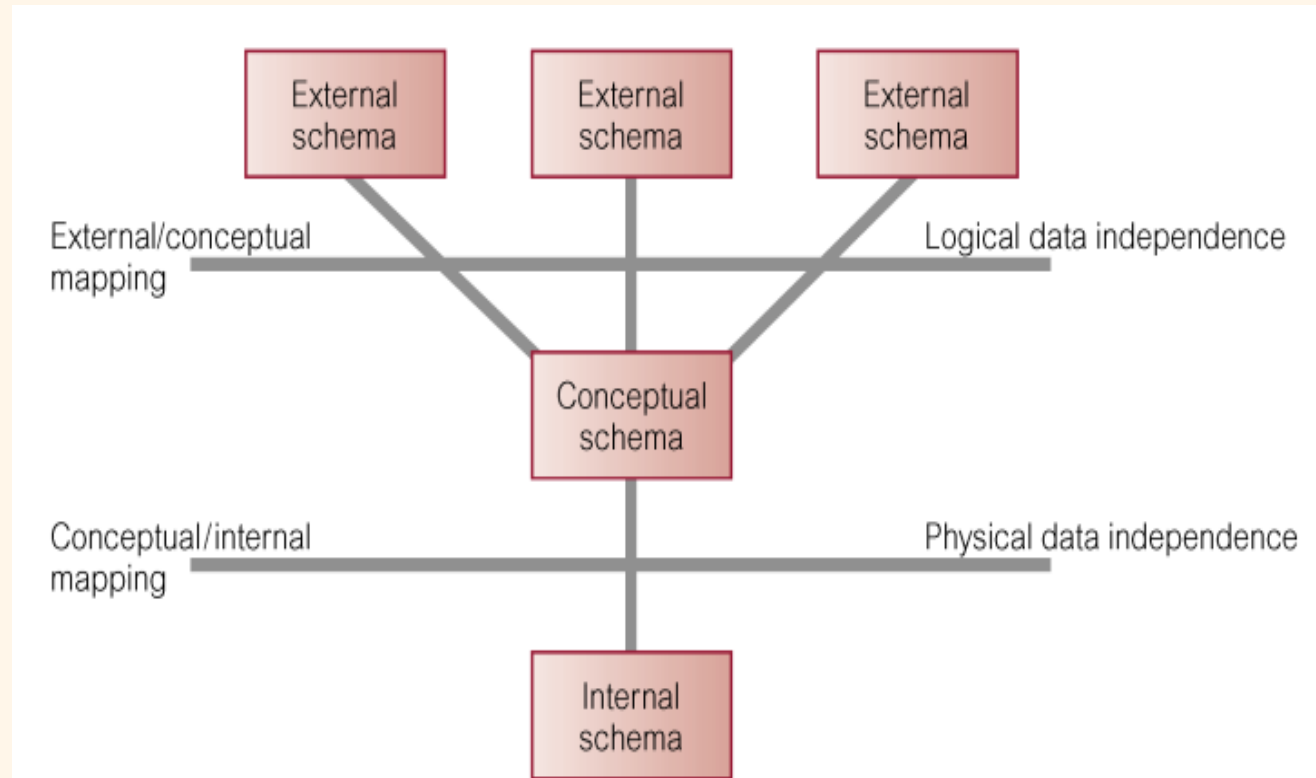



Data Independence



- **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 requires changes to external schema or rewrites of application programs

Data Independence and the ANSI-SPARC Three-Level Architecture





Schemas and Instances



Similar to types and variables in programming languages

- **Schema** – the logical structure of the database
E.g., the database consists of information about a set of customers and accounts and the relationship between them
Analogue to type information of a variable in a program
Physical schema: database design at the physical level
Logical schema: database design at the logical level
Sub-Schema: there may be several of these at the view level, each a different view of database
- **Instance** – the actual content of the database at a particular point in time
Analogous to the value of the variable

Hello



Database Schema



- Is the description of a database
- It is specified during the database design and is not expected to change frequently.
- It is represented as a diagram called **schema diagram**.
 - A schema diagram displays the structure of each record type but not the actual instance of a record.
- Each object in a schema is called a **schema construct**.



Example of a Database Schema



STUDENT	Name	StudentNumber	Class	Major
---------	------	---------------	-------	-------

COURSE	CourseName	CourseNumber	CreditHours	Department
--------	------------	--------------	-------------	------------

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
---------	-------------------	--------------	----------	------	------------

GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
--------------	---------------	-------------------	-------

PREREQUISITE	CourseNumber	PrerequisiteNumber
--------------	--------------	--------------------



Database State



- A Database state or **instance** is the data in the database at a particular moment of time.
- Every update operation changes the database from one state to another.
- The Schema is sometimes called the **intension**, and the database state an **extension** of the schema.

Example of a Database State



STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	CS1310	Fall	98	Anderson
	102	CS3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	CS1310	Fall	99	Anderson
	135	CS3380	Fall	99	Stone

GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	CS3380	CS3320
	CS3380	MATH2410
	CS3320	CS1310



Database Languages

- **Data Definition Language (DDL)**
 - Allows the DBA or users 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 of 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