# Database Systems

Lecture 3 – 4

Instructor: Bilal Khalid Dar

# Agenda

- Data Model
  - Data Model Structure and Constraints
  - Data Model Operations
- Categories of Data Models
  - (Conceptual – Physical - Implementation )
- Database Schema VS Database State, Data Base Instance
- Relational DBMS
- Database Design /Database Model
- Database Development Stages

# Agenda

- Relational Model
  - What is Relational Model,
  - The Use Of Relational Model,
  - Parts of Relational Model (Structural, Manipulative, Integrity )
- Database Relations
- Properties of Relation
- Relational Keys
- Representing Relational DB Schema
- Integrity Constraints
- View, Purpose of Views, Updating View

# 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 group
- Constraints specify some restrictions on valid data; these constraints must be enforced at all times

# Data Models (continued)

- **Data Model Operations:**
- These operations are used for specifying database retrievals and updates by referring to the constructs of the data model.
- Operations on the data model may include basic model operations (e.g. generic insert, delete, update, remove) and user-defined operations (e.g. compute_student_gpa, update_inventory)

# 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.
- These are usually specified in an ad-hoc manner through DBMS design and administration manuals

- **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)

# Schemas versus Instances

- **Database Schema:**
- The description of a database
- Includes descriptions of the database structure, data types, and the constraints on the database

- **Schema Construct:**

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

# Schemas versus Instances

- **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.
- It is also called database instance (or occurrence or snapshot).
- The term instance is also applied to individual database components, e.g. record instance, table instance, entity instance.

# Schemas versus Instances

- **Database State:**

Refers to the content of a database at a moment in time.

- **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.

# Schemas versus Instances

**Distinction (Note)**

- The database schema changes very infrequently.
- The database state changes every time table is updated
- Schema is also called intension.
- State is also called extension.
- Schema – Structural description of relations in database
- Schema is a repository or structure to express the format and other different information about data and database,
- Actual place where these definitions and descriptions are performed is database schema.
- Instance – Actual contents at given point in time

# Example of a Database Schema

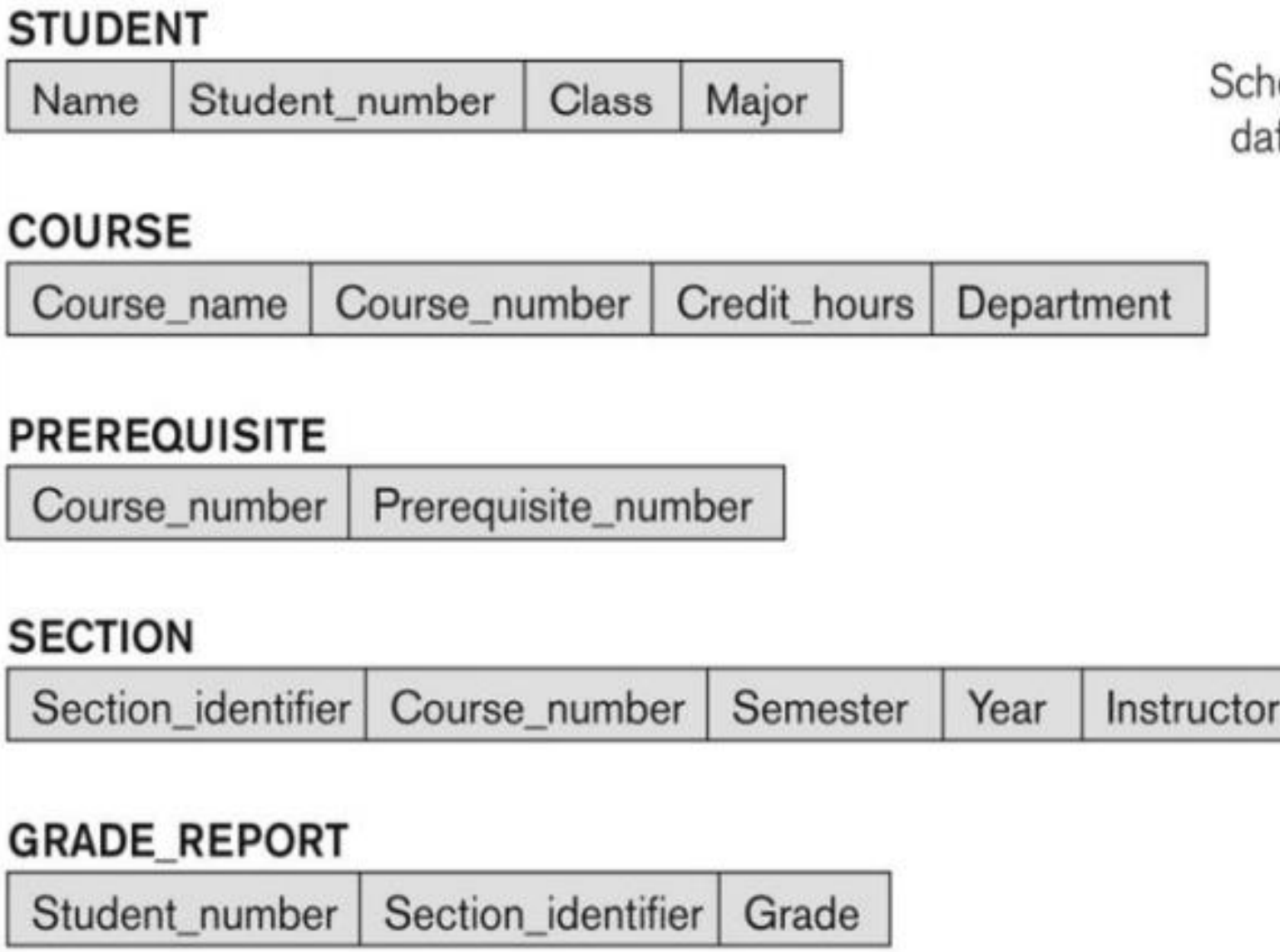



**Figure 2.1**
Schema diagram for the database in Figure 1.2.

# 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.

# Relational DBMS - (RDBMS)

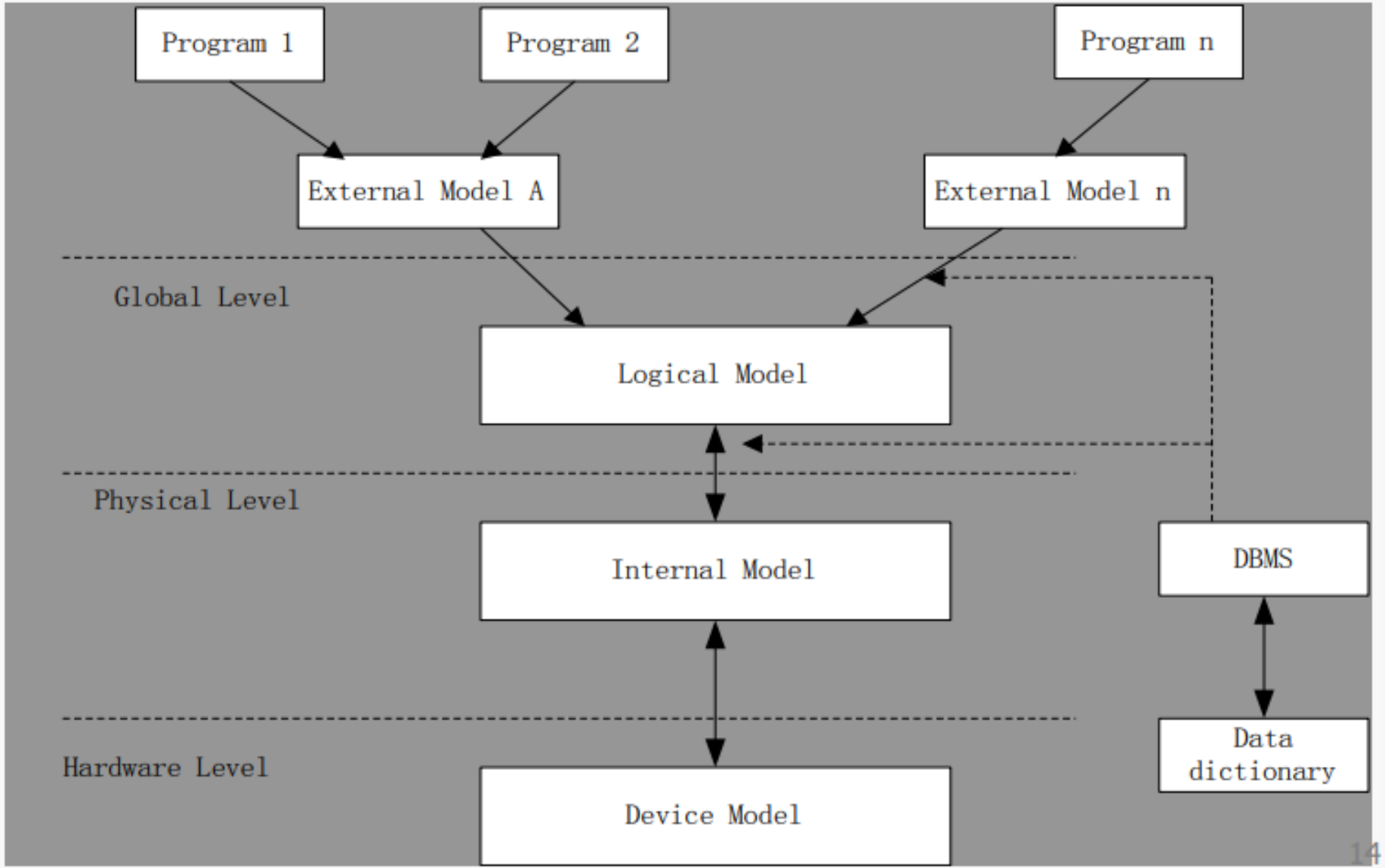
- **RDBMS** is one of the most widely used and easiest to operate database systems.

- **Relational data model (RDM) –** Theoretical basis of RDBMS.

- **Logical data structure – Table (relation)**

A C. Each row (tuple) contains one value for the associated column.

# Relational DBMS - (RDBMS)



# Database Design /Database Model

- These terms can be used interchangeably for the logical structure of the database.
- The Database design/model stores the structure of the data and the links/relationships between Data that should be stored to meet the users' requirements.
- Database design is stored in the database schema, which is in turn stored in the data dictionary

# Database Modeling

- **The process of creating the logical structure of the database is called database modeling.**

- It is a very important process because the designing of the application provides us the basis for running our database system. If the database is not designed properly the implementation of the system can not be done properly

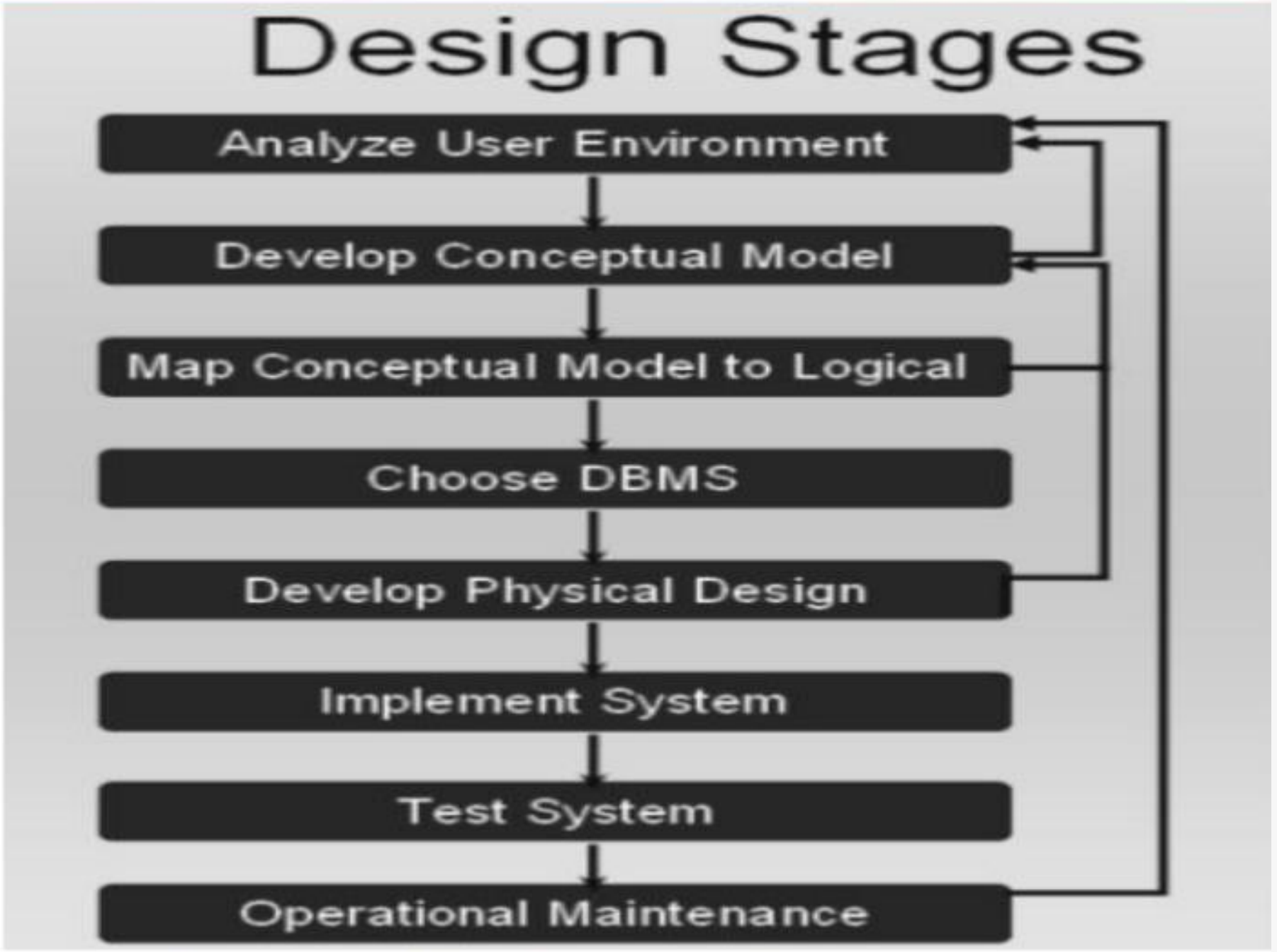



Fig: Database Development Stages.

# Relational Model

•The relational model is the conceptual basis of relational databases. Proposed by E.F. Codd in 1969, it is a method of structuring data using relations, which are grid-like mathematical structures consisting of columns and rows. Codd (1970) introduced normalized relations in the relational model .

- Very simple model
- Efficient implementations
- Used by all major commercial database systems
- Query with high-level languages: simple yet expressive

# Relational Model - The Use – Why a relational model?

**To represent data in an understandable way**

It is basis for designing relational RDBMSs.

1) The designer to reason about the database systems without actually having build them;
2) 2) An RDBMS implements the structures, operations and rules of the relational model.

# Relational Model

- Three parts of a relational model:

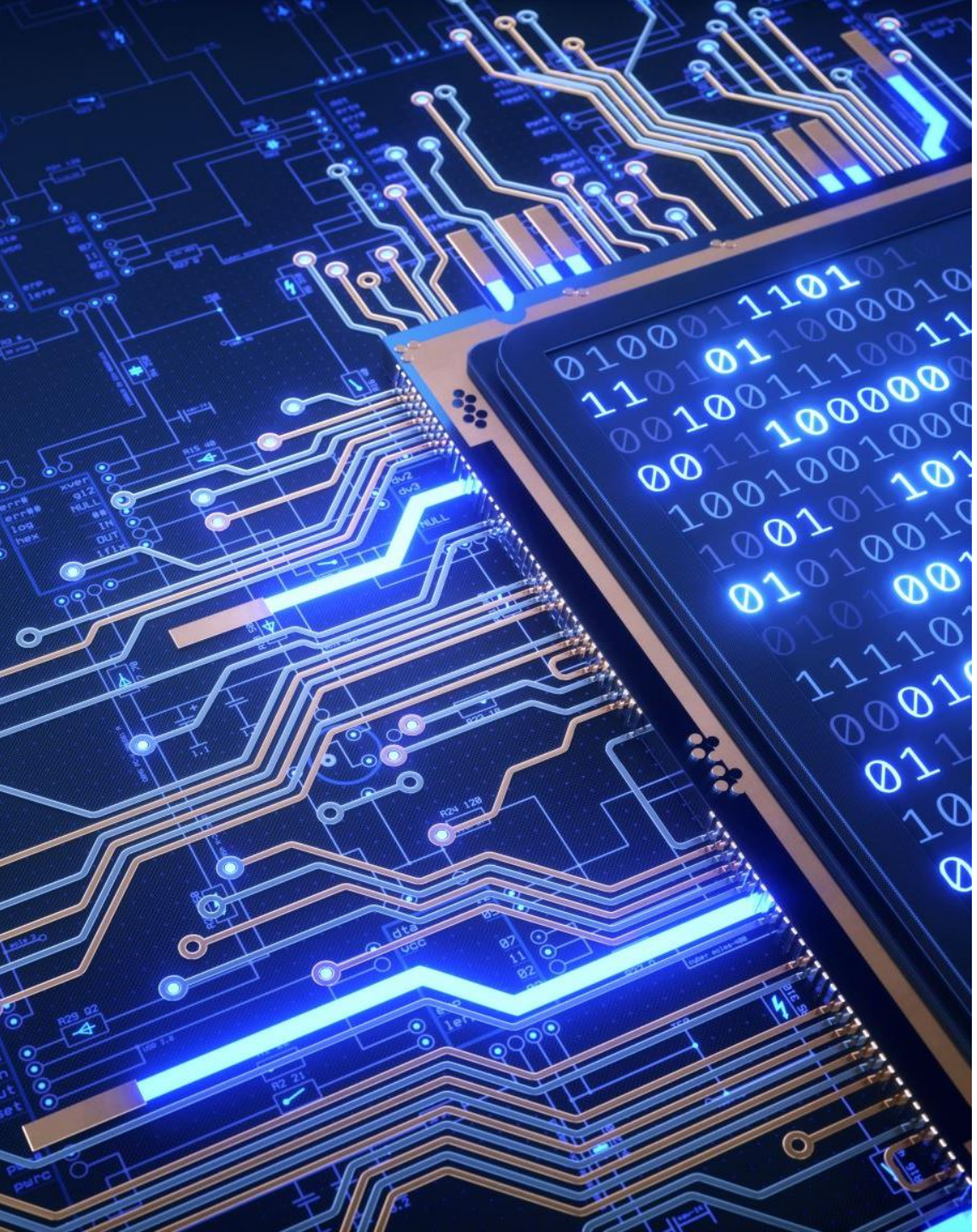
1) **Structural part**, Description of the **database elements** of and the structure of the database system.

2) **Manipulative part**, Operations working on databases to implement the functions of systems;

3) **Integrity** part, The rules that databases must obey – to make sure the system work properly.

# Relational Model

## 1- The Structural Part



# Overview of Database Design Process

- Two main activities:
  - Database design
  - Applications design
- Focus in this chapter on database design
  - To design the conceptual schema for a database application
- Applications design focuses on the programs and interfaces that access the database
  - Generally considered part of software engineering

# Example COMPANY Database

- We need to create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
  - The company is organized into DEPARTMENTs. Each department has a name, number and an employee who *manages* the department. We keep track of the start date of the department manager. A department may have several locations.
  - Each department *controls* a number of PROJECTs. Each project has a unique name, unique number and is located at a single location.

# Example COMPANY Database (Contd.)

- We store each EMPLOYEE's social security number, address, salary, gender, and birthdate.
  - Each employee *works for* one department but may *work on* several projects.
  - We keep track of the number of hours per week that an employee currently works on each project.
  - We also keep track of the *direct supervisor* of each employee.
- Each employee may *have* a number of DEPENDENTs.
  - For each dependent, we keep track of their name, gender, birthdate, and relationship to the employee.

# ER Model Concepts

- Entities and Attributes
  - Entities are specific objects or things in the mini-world that are represented in the database.
    - For example the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT
  - Attributes are properties used to describe an entity.
    - For example an EMPLOYEE entity may have the attributes Name, SSN, Address, Gender, BirthDate
  - A specific entity will have a value for each of its attributes.
    - For example a specific employee entity may have Name='John Smith', SSN='123456789', Address ='731, Fondren, Houston, TX', Gender='M', BirthDate='09-JAN-55'
  - Each attribute has a *value set* (or data type) associated with it – e.g. integer, string, subrange, enumerated type, …

# Types of Attributes (1)

## Simple

- Each entity has a single atomic value for the attribute. For example, SSN or Gender.

## Composite

- The attribute may be composed of several components. For example:
  - Address(Apt#, House#, Street, City, State, ZipCode, Country), or
  - Name(FirstName, MiddleName, LastName).
  - Composition may form a hierarchy where some components are themselves composite.

## Multi-valued

- An entity may have multiple values for that attribute. For example, Color of a CAR or PreviousDegrees of a STUDENT.
  - Denoted as {Color} or {PreviousDegrees}.


Thank
You