# ICT713
# Advanced Database Management

## Lecture 1 – Chapter 2
## Data models

Adopted from: Coronel, C. and Morris, S. (2022). *Database Systems: Design, Implementation, & Management,* 14th Edition., Cengage Learning

Compiled By: Divya Leekha

# Subject Learning Outcomes

| Subject Learning Outcomes |
| --- |
| a) Model and design databases based on given requirements using ERDs and EERDs |
| b) Optimise querying, storing and indexing of databases |
| c) Articulate the trade-offs in using distributed databases and data warehousing technologies |
| d) Employ emerging trends in the field of database design and development |

# Learning Objectives

- After completing this chapter, you will be able to:
  - Discuss data modeling and why data models are important
  - Describe the basic data-modeling building blocks
  - Define what business rules are and how they influence database design
  - Understand how the major data models evolved
  - List emerging alternative data models and the needs they fulfill
  - Explain how data models can be classified by their level of abstraction

# Data Modeling and Data Models

- Data modeling: creating a specific data model for a determined problem domain
  - Data model: simple representation of complex real-world data structures
    - Useful for supporting a specific problem domain
  - Model: abstraction of a more complex real-world object or event

# The Importance of Data Models

- The importance of data modeling cannot be overstated
  - Facilitates communication
  - Gives various views of the database
  - Organizes data for various users
  - Provides an abstraction for the creation of good a database

# Data Model Basic Building Blocks

- Entity: person, place, thing, or event about which data will be collected and stored
  - Attribute: characteristic of an entity
  - Relationship: association among entities
    - One-to-many (1:M OR 1..*)
    - Many-to-many (M:N or *..*)
    - One-to-one (1:1 OR 1..1)
  - Constraint: restriction placed on data
    - Ensures data integrity

# Business Rules

- Brief, precise, and unambiguous description of a policy, procedure, or principle
  - Create and enforce actions within that organization’s environment
  - Establish entities, relationships, and constraints

# Discovering Business Rules (1 of 2)

- Sources of business rules
  - Company managers
  - Policy makers
  - Department managers
  - Written documentation
  - Direct interviews with end users

# Discovering Business Rules (2 of 2)

- Reasons for identifying and documenting business rules
  - Standardize company's view of data
  - Facilitate communications tool between users and designers
  - Assist designers
    - Understand the nature, role, scope of data, and business processes
    - Develop appropriate relationship participation rules and constraints
    - Create an accurate data model

# Translating Business Rules into Data Model Components

- Business rules set the stage for the proper identification of entities, attributes, relationships, and constraints
  - Nouns translate into entities
  - Verbs translate into relationships among entities
- Relationships are bidirectional
  - Questions to identify the relationship type
    - How many instances of B are related to one instance of A?
    - How many instances of A are related to one instance of B?

# Naming Conventions

- Entity name requirements
  - Be descriptive of the objects in the business environment
  - Use terminology that is familiar to the users
- Attribute name
  - Required to be descriptive of the data represented by the attribute
- Proper naming
  - Facilitates communication between parties
  - Promotes self-documentation

# Hierarchical and Network Models (1 of 2)

- Hierarchical models: developed to manage large amounts of data for complex manufacturing projects
  - Represented by an upside-down tree which contains segments
    - Segments are the equivalent of a file system’s record type
  - Depicts a set of one-to-many (1:M) relationships

# Hierarchical and Network Models (2 of 2)

- Network models: created to represent complex data relationships effectively
  - Improved database performance and imposed a database standard
  - Allows a record to have more than one parent
- Standard database concepts that emerged with the network model are still used by modern data models
  - Schema and subschema
  - Data manipulation language (DML)
  - Data definition language (DDL)

# The Relational Model (1 of 4)

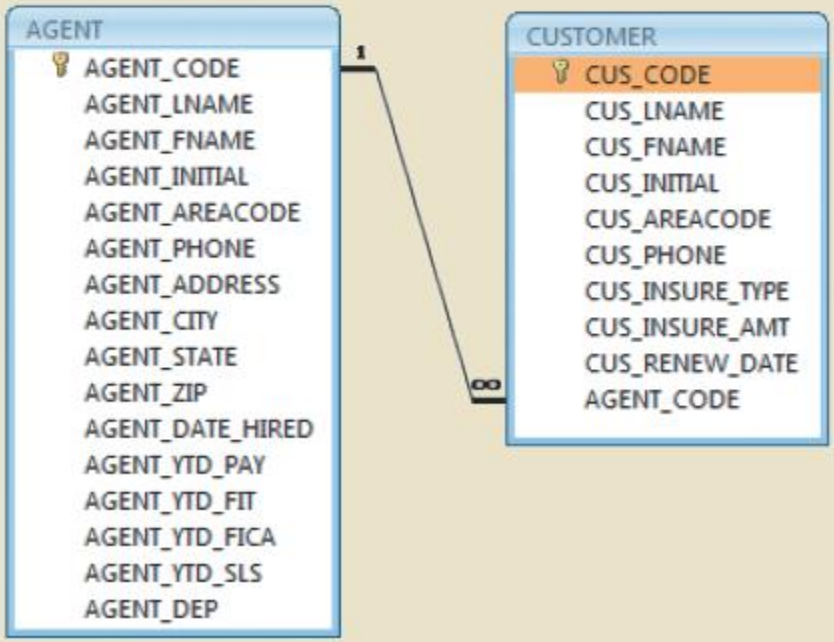
- Produced an automatic transmission database that replaced standard transmission databases
  - Based on a relation (i.e., table): matrix composed of intersecting tuples (rows) and attributes (columns)
- Describes a precise set of data manipulation constructs

# The Relational Model (2 of 4)

- Relational database management system (RDBMS)
  - Performs basic functions provided by the hierarchical and network DBMS systems
  - Makes the relational data model easier to understand and implement
  - Hides the complexities of the relational model from the user

# The Relational Model (3 of 4)

FIGURE 2.2 A RELATIONAL DIAGRAM



# The Relational Model (4 of 4)

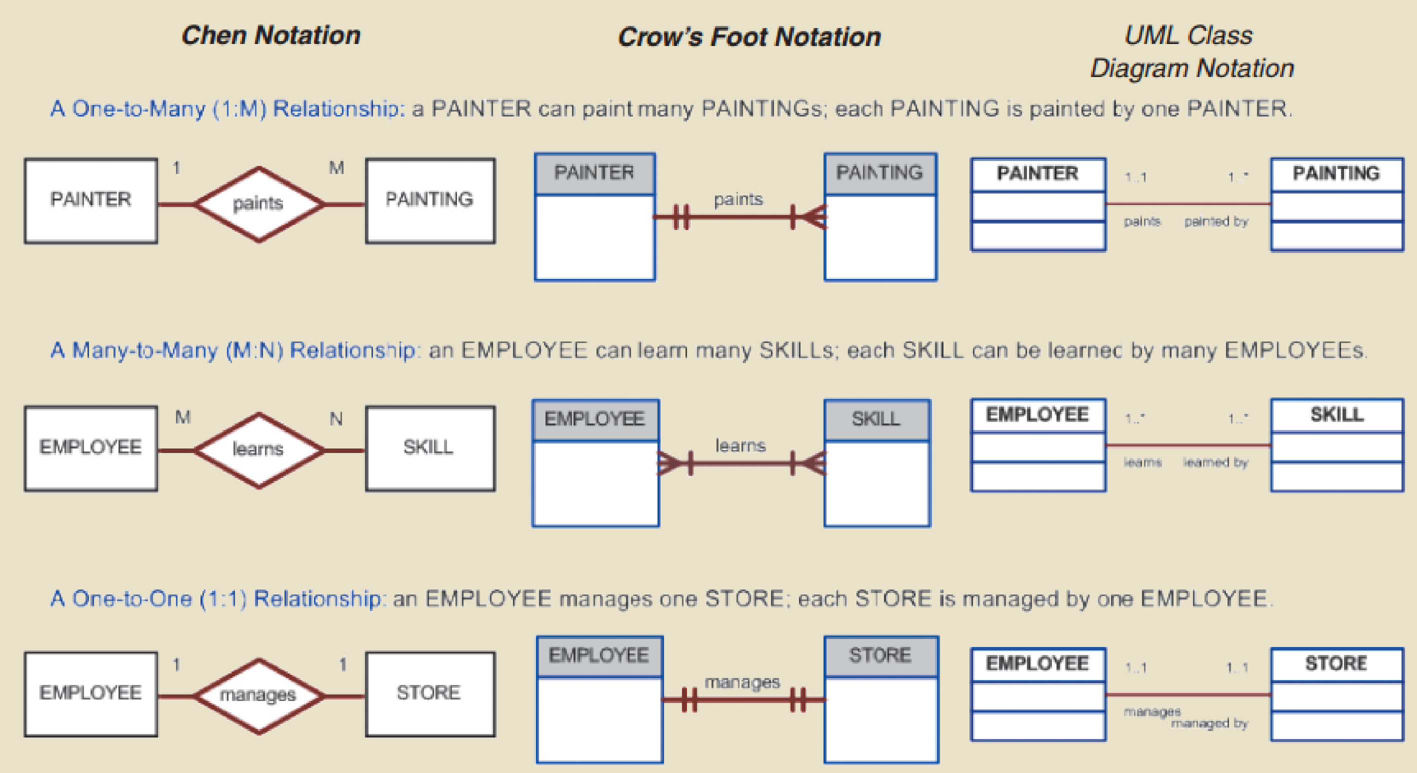
- SQL-based relational database application
  - End-user interface
    - Allows end user to interact with the data
  - Collection of tables stored in the database
    - Each table is independent from another
    - Rows in different tables are related based on common values in common attributes
  - SQL engine
    - Executes all queries

# The Entity Relationship Model (1 of 2)

- Graphical representation of entities and their relationships in a database structure
  - Entity relationship diagram (ERD): uses graphic representations to model database components
  - Entity instance or entity occurrence: rows in the relational table
  - Attributes: describe particular characteristics
  - Connectivity: term used to label the relationship types

# The Entity Relationship Model (2 of 2)

FIGURE 2.3 THE ER MODEL NOTATIONS



# The Object-Oriented Data Model (1 of 3)

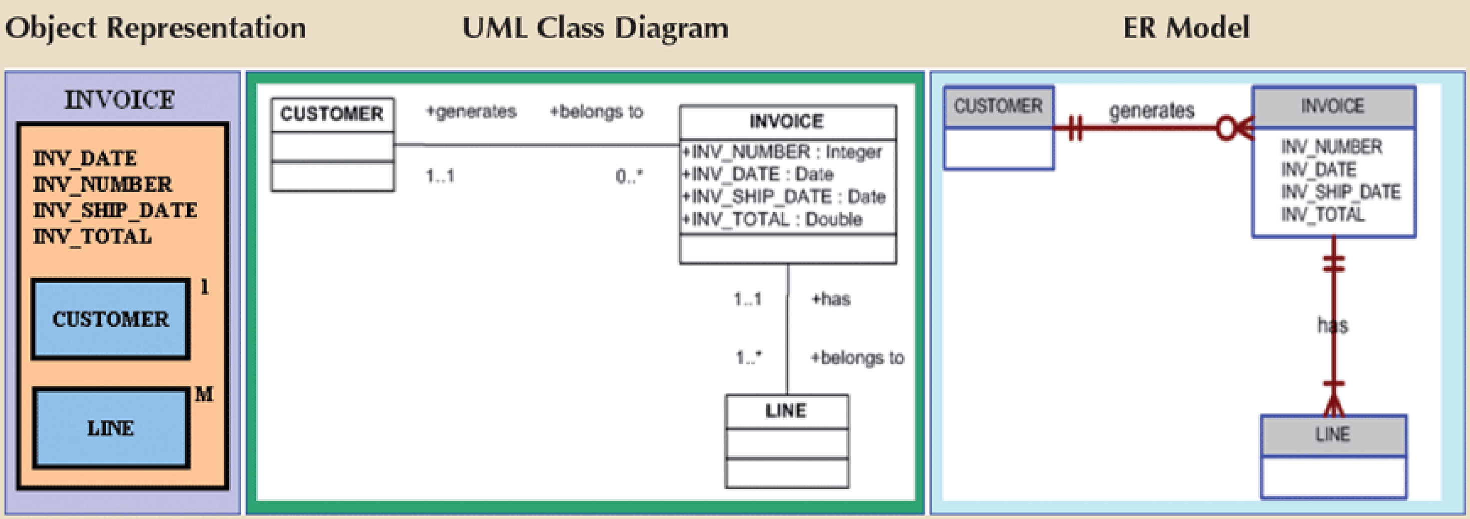
- Both data and its relationships are contained in a single structure known as an object
  - Object-oriented database management system(OODBMS): based on OODM
- Object: contains data and their relationships with operations that are performed on it
  - Basic building block for autonomous structures
  - Abstraction of real-world entity
- Attribute: describes the properties of an object

# The Object-Oriented Data Model (2 of 3)

- Class: collection of similar objects with shared structure and behavior organized in a class hierarchy
- Class hierarchy: resembles an upside-down tree in which each class has only one parent
- Inheritance: object inherits methods and attributes of classes above it
- Unified Modeling Language (UML): describes sets of diagrams and symbols to graphically model a system

# The Object-Oriented Data Model (3 of 3)

FIGURE 2.4 A COMPARISON OF THE OO, UML, AND ER MODELS



# Object/Relational and XML

- Extended relational data model (ERDM)
  - Supports OO features, extensible data types based on classes, and inheritance
    - Object/relational database management system (O/R DBMS): based on ERDM
- Extensible Markup Language (XML)
  - Manages unstructured data for efficient and effective exchange of structured, semistructured, and unstructured data

# Emerging Data Models: Big Data and NoSQL (1 of 3)

- Goals of Big Data
  - Find new and better ways to manage large amounts of web and sensor-generated data
  - Provide high performance at a reasonable cost
- Characteristics of Big Data
  - Volume
  - Velocity
  - Variety

# Emerging Data Models: Big Data and NoSQL (2 of 3)

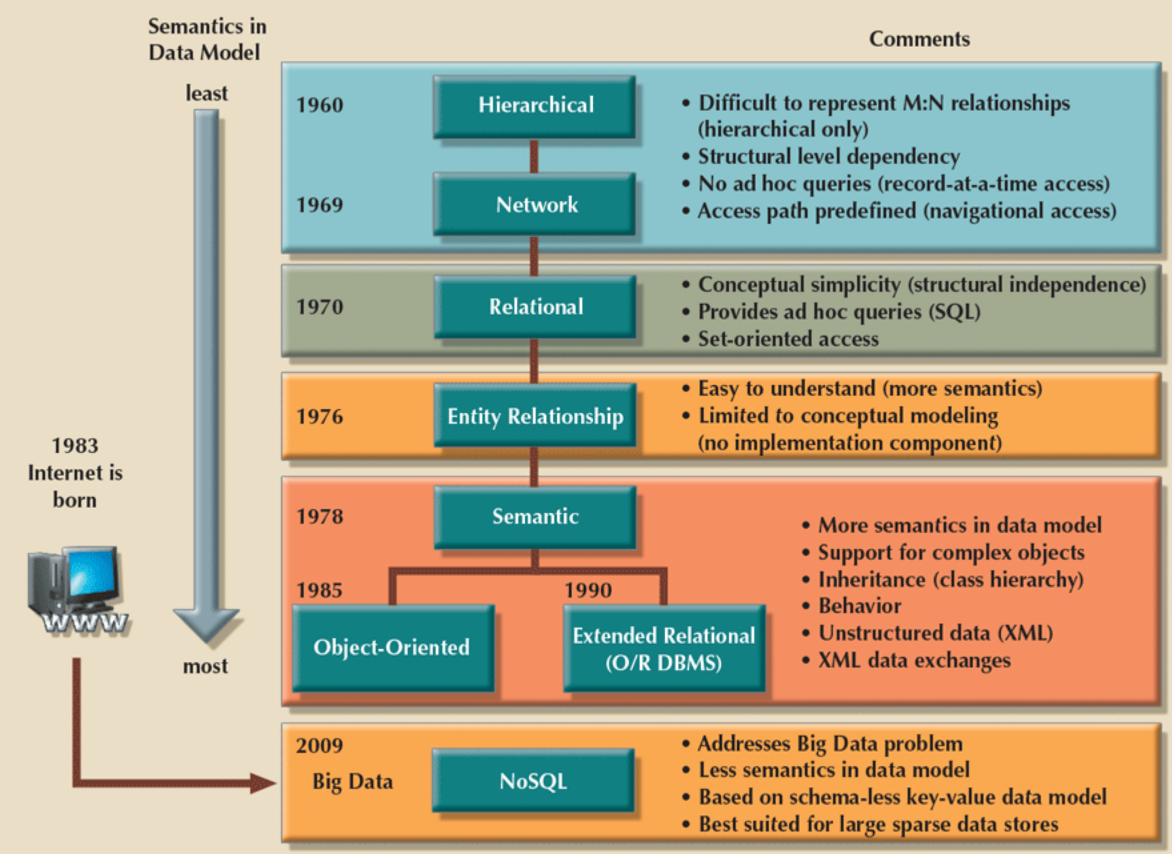
- Challenges of Big Data
  - Volume doesn’t allow usage of conventional structures
  - Expensive
  - OLAP tools proved inconsistent dealing with unstructured data
- New technologies of Big Data
  - Hadoop
  - Hadoop Distributed File System (HDFS)
  - MapReduce
  - NoSQL

# Emerging Data Models: Big Data and NoSQL (3 of 3)

- NoSQL databases
  - Not based on the relational model
  - Support distributed database architectures
  - Provide high scalability, high availability, and fault tolerance
  - Support large amounts of sparse data
  - Geared toward performance rather than transaction consistency
  - Provides a broad umbrella for data storage and manipulation

# Data Models: A Summary

FIGURE 2.5 THE EVOLUTION OF DATA MODELS



# Hierarchical Model

- Advantages
  - Promotes data sharing
  - Parent/child relationship promotes conceptual simplicity and data integrity
  - Database security is provided and enforced by DBMS
  - Efficient with 1:M relationships
- Disadvantages
  - Requires knowledge of physical data storage characteristics
  - Navigational system requires knowledge of hierarchical path
  - Changes in structure require changes in all application programs
  - Implementation limitations
  - No data definition
  - Lack of standards

# Network Model

- Advantages
  - Conceptual simplicity
  - Handles more relationship types
  - Data access is flexible
  - Data owner/member relationship promotes data integrity
  - Conformance to standards
  - Includes data definition language (DDL) and data manipulation language (DML)
- Disadvantages
  - System complexity limits efficiency
  - Navigational system yields complex implementation, application development, and management
  - Structural changes require changes in all application programs

# Relational Model

- Advantages
  - Structural independence is promoted using independent tables
  - Tabular view improves conceptual simplicity
  - Ad hoc query capability is based on SQL
  - Isolates the end user from physical-level details
  - Improves implementation and management simplicity
- Disadvantages
  - Requires substantial hardware and system software overhead
  - Conceptual simplicity gives untrained people the tools to use a good system poorly
  - May promote information problems

# Entity Relationship Model

- Advantages
  - Visual modeling yields conceptual simplicity
  - Visual representation makes it an effective communication tool
  - Is integrated with the dominant relational model
- Disadvantages
  - Limited constraint representation
  - Limited relationship representation
  - No data manipulation language
  - Loss of information content occurs when attributes are removed from entities to avoid crowded displays

# Object-Oriented Model

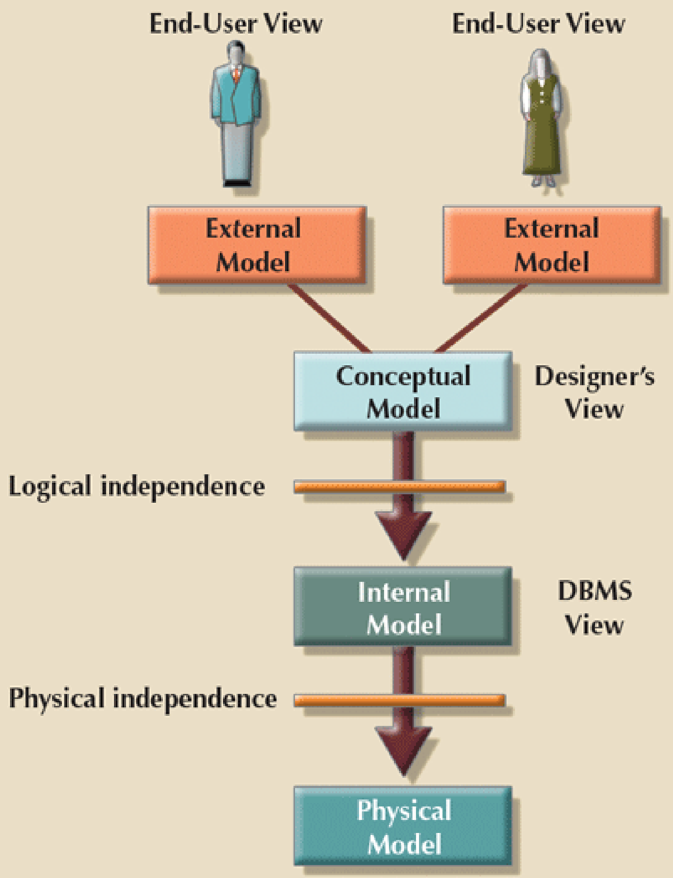
- Advantages
  - Semantic content is added
  - Visual representation includes semantic content
  - Inheritance promotes data integrity
- Disadvantages
  - Slow development of standards caused vendors to supply their own enhancements
  - Complex navigational system
  - Learning curve is steep
  - High system overhead slows transactions

# NoSQL

- Advantages
  - High scalability, availability, and fault tolerance are provided
  - Uses low-cost commodity hardware
  - Supports Big Data
  - Key-value model improves storage efficiency
- Disadvantages
  - Complex programming is required
  - There is no relationship support
  - There is no transaction integrity support
  - In terms of data consistency, it provides an eventually consistent model

# Degrees of Data Abstraction

FIGURE 2.6 DATA ABSTRACTION LEVELS



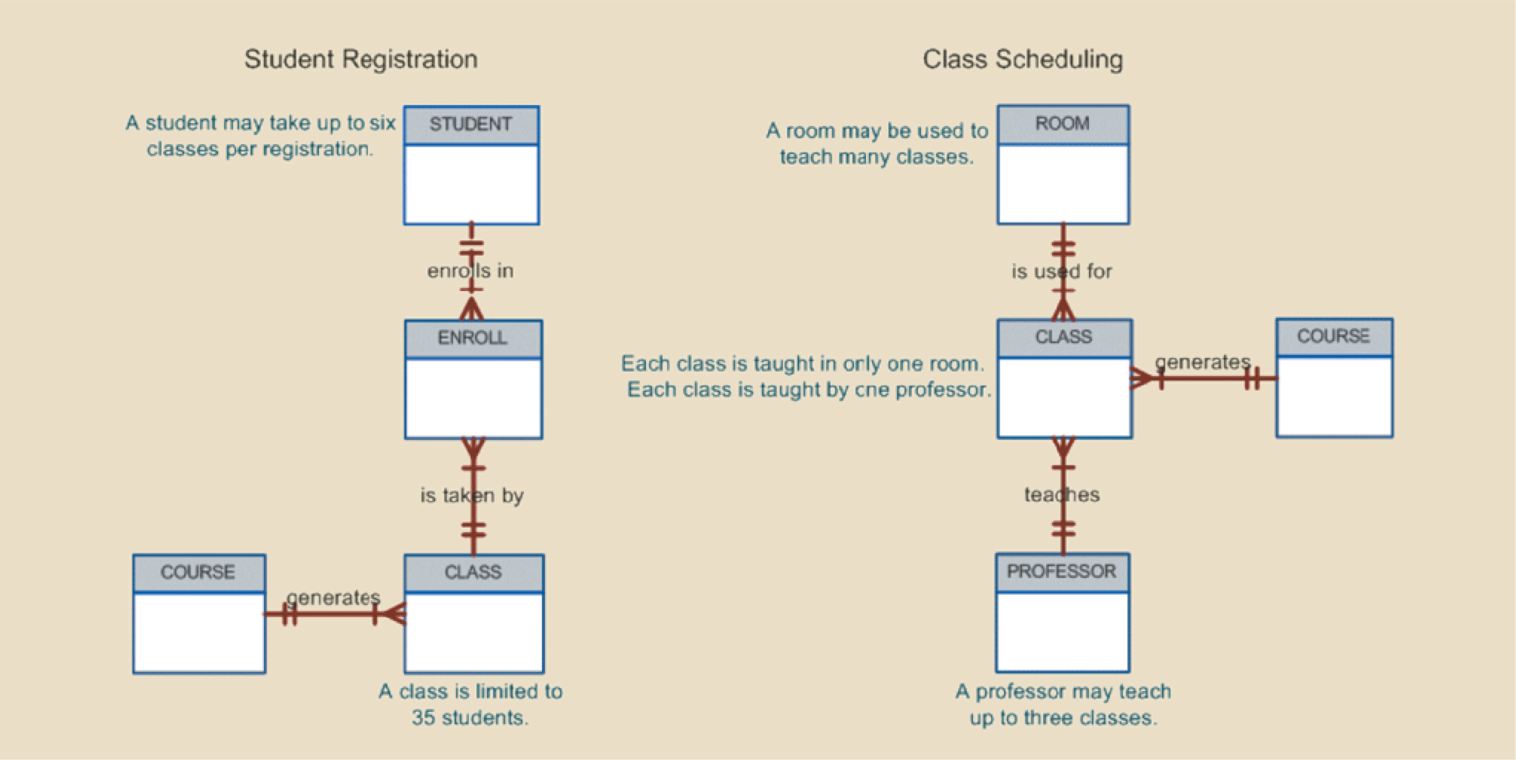

| Degree of Abstraction | | Characteristics |
|---|---|---|
| High | ER | Hardware-independent Software-independent |
| | Object-Oriented | |
| Medium | Relational | Hardware-independent Software-dependent |
| Low | Network Hierarchical | Hardware-dependent Software-dependent |

# The External Model (1 of 2)

- End users' view of the data environment
  - People who use the application programs to manipulate the data and generate information
- ER diagrams are used to represent the external views
  - External schema: specific representation of an external view

# The External Model (2 of 2)

FIGURE 2.7 EXTERNAL MODELS FOR TINY COLLEGE



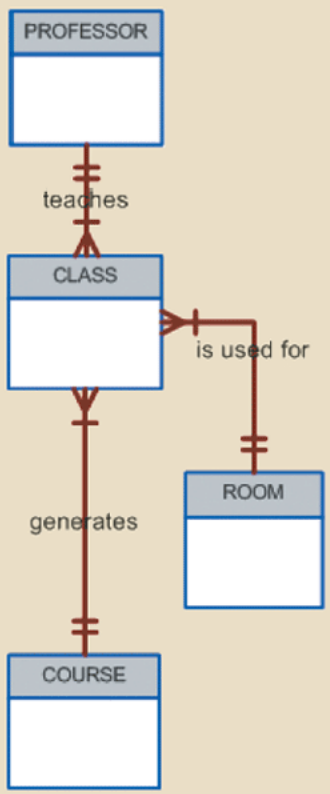
# The Conceptual Model (1 of 2)

- Represents a global view of the entire database by the entire organization
  - Conceptual schema: basis for the identification and high-level description of the main data objects
  - Logical design: task of creating a conceptual data model
- Conceptual model advantages
  - Macro-level view of data environment
  - Software and hardware independent

# The Conceptual Model (2 of 2)

FIGURE 2.9 INTERNAL MODEL FOR TINY COLLEGE

CONCEPTUAL MODEL




INTERNAL MODEL

```
Create Table PROFESSOR(
PROF_ID        NUMBER PRIMARY KEY,
PROF_LNAME     CHAR(15),
PROF_INITIAL   CHAR(1),
PROF_FNAME     CHAR(15),
.........);

Create Table CLASS(
CLASS_ID       NUMBER PRIMARY KEY,
CRS_ID         CHAR(8)  REFERENCES COURSE,
PROF_ID        NUMBER REFERENCES PROFESSOR,
ROOM_ID        CHAR(8)  REFERENCES ROOM,
.........);

Create Table ROOM(
ROOM_ID        CHAR(8) PRIMARY KEY,
ROOM_TYPE      CHAR(3),
.........);

Create Table COURSE(
CRS_ID         CHAR(8) PRIMARY KEY,
CRS_NAME       CHAR(25),
CRS_CREDITS    NUMBER,
.........);
```

# The Internal Model

- Representing database as seen by the DBMS mapping conceptual model to the DBMS
  - Internal schema: specific representation of an internal model, using the database constructs supported by the chosen database
  - Logical independence: changing internal model without affecting the conceptual model
  - Hardware independent: unaffected by the type of computer on which the software is installed

# The Physical Model (1 of 2)

- Operates at lowest level of abstraction
  - Describes the way data are saved on storage media such as magnetic, solid state, or optical media
- Requires the definition of physical storage and data access methods
  - Software and hardware dependent
- Relational model aimed at logical level
  - Does not require physical-level details
- Physical independence: changes in physical model do not affect internal model

# The Physical Model (2 of 2)

| **Table 2.4 Levels of Data Abstraction** | | | |
|---|---|---|---|
| Model | Degree of Abstraction | Focus | Independent of |
| External | High | End-user views | Hardware and software |
| Conceptual | Medium-High | Global view of data (database model independent) | Hardware and software |
| Internal | Medium-Low | Specific database model | Hardware |
| Physical | Low | Storage and access methods | Neither hardware nor software |

# Summary

- A data model is an abstraction of a complex real-world data environment
- There are many types of data models (e.g., hierarchical, network, relational, object-oriented, extended relational data model, etc.)
- Data-modeling requirements are a function of different data views (global versus local) and the level of data abstraction