

1/39    7:31:22    \*\*\*

← →

# Data Modeling

•

11e

Database Systems  
Design, Implementation, and Management

Coronel | Morris

## Chapter 2 Data Models

## Learning Objectives

- In this chapter, you will learn:
  - About data modeling and why data models are important
  - About the basic data-modeling building blocks
  - What business rules are and how they influence database design

## Learning Objectives

- In this chapter, you will learn:
  - How the major data models evolved
  - About emerging alternative data models and the need they fulfill
  - How data models can be classified by their level of abstraction

## Data Modeling and Data Models

- **Data modeling:** Iterative and progressive process of creating a specific data model for a determined problem domain
- **Data models:** Simple representations of complex real-world data structures
- Useful for supporting a specific problem domain
- **Model** - Abstraction of a real-world object or event

•

## Importance of Data Models

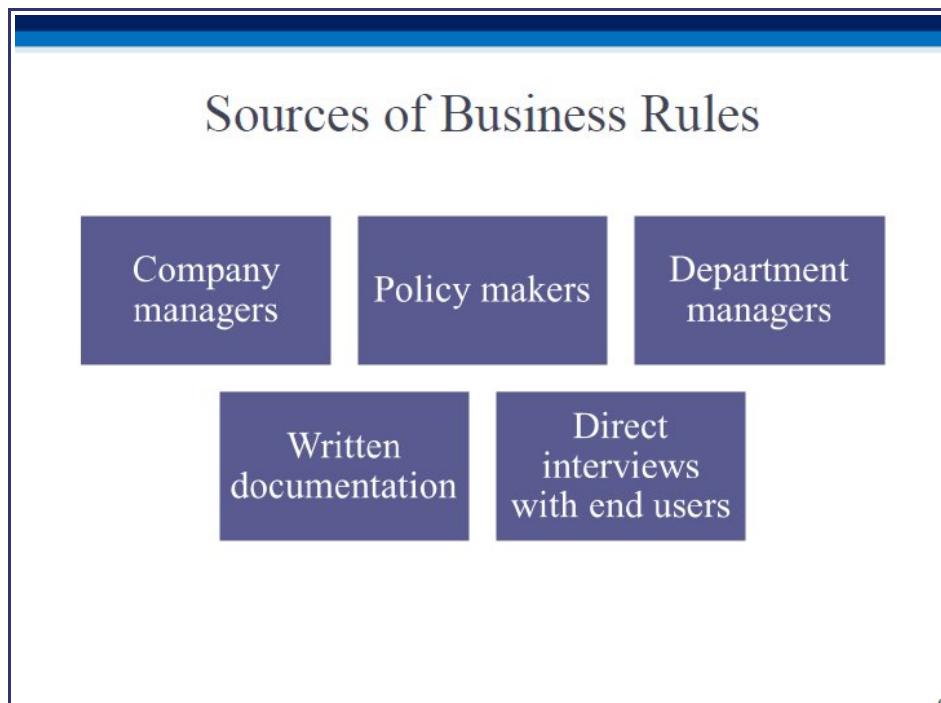
- Are a communication tool
- Give an overall view of the database
- Organize data for various users
- Are an abstraction for the creation of good database

- 

## Data Model Basic Building Blocks

- **Entity:** Unique and distinct object used to collect and store data
  - **Attribute:** Characteristic of an entity
- **Relationship:** Describes an association among entities
  - **One-to-many (1:M)**
  - **Many-to-many (M:N or M:M)**
  - **One-to-one (1:1)**
- **Constraint:** Set of rules to ensure data integrity

- - Business Rules
    - Brief, precise, and unambiguous description of a policy, procedure, or principle
    - Enable defining the basic building blocks
    - Describe main and distinguishing characteristics of the data



- 

## Reasons for Identifying and Documenting Business Rules

- Help standardize company's view of data
- Communications tool between users and designers
- Allow designer to:
  - Understand the nature, role, scope of data, and business processes
  - Develop appropriate relationship participation rules and constraints
  - Create an accurate data model

- 

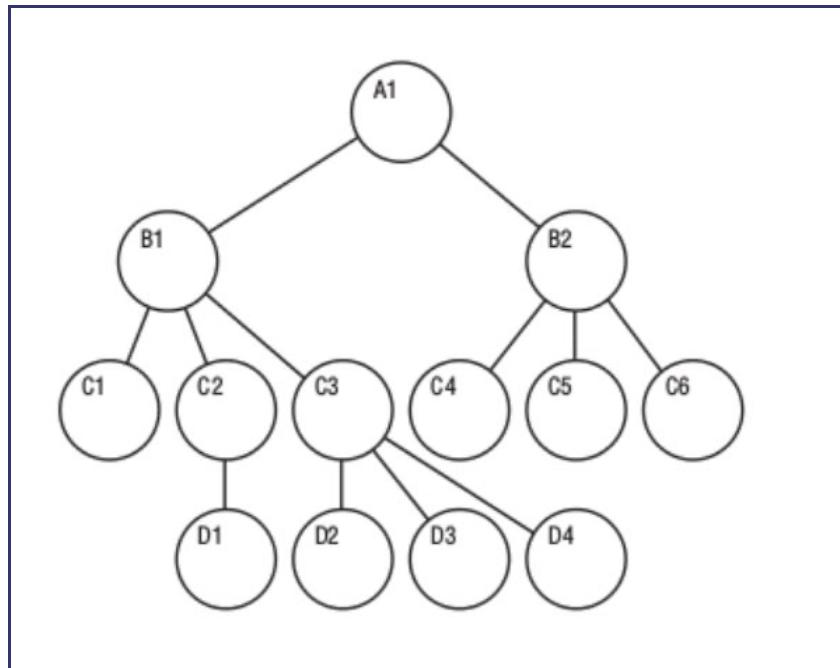
## Translating Business Rules into Data Model Components

- 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 names - Required to:
  - 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 modeling

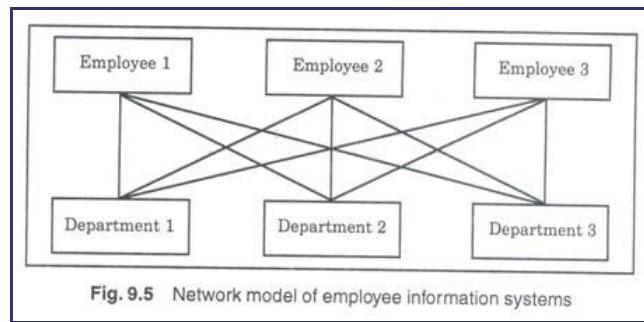


At first, data was stored in individual files (transitioned from paper). The next improvement was a 'hierarchical DB model', where data was structured in the form of a tree [similar to a modern filesystem].

Data, in the form of nodes, are linked in a tree-like fashion. To traverse the tree, we need to know the underlying format ('class hierarchy, to make an analogy with classes and objects), and the actual path [eg. to relate A1 and D2, we need to traverse A1->B1->C3>D2].

Hierarchies are good for '1:M' [tree], but not 'M:N' [graph or multiple inheritance].

# Network modeling



A network model is better than a hierarchical one, because it can capture M:N [in addition to the above, another example is 'products and orders'].

•

## Hierarchical and Network Models

### Hierarchical Models

- Manage large amounts of data for complex manufacturing projects
- Represented by an upside-down tree which contains segments
  - **Segments:** Equivalent of a file system's record type
- Depicts a set of one-to-many (1:M) relationships

### Network Models

- Represent complex data relationships
- Improve database performance and impose a database standard
- Depicts both one-to-many (1:M) and many-to-many (M:N) relationships

## 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

- 

## Standard Database Concepts

### Schema

- Conceptual organization of the entire database as viewed by the database administrator

### Subschema

- Portion of the database seen by the application programs that produce the desired information from the data within the database

# Data creation, querying

## Schema data definition language (DDL)

- Enables the database administrator to define the schema components

## Data manipulation language (DML)

- Environment in which data can be managed and is used to work with the data in the database

# Relational model

## The Relational Model

- Based on a relation
  - **Relation or table:** Matrix composed of intersecting tuple and attribute
    - **Tuple:** Rows
    - **Attribute:** Columns
  - Describes a precise set of data manipulation constructs

# Relational model

Relational Model	
Advantages	Disadvantages
<ul style="list-style-type: none"><li>▪ Structural independence is promoted using independent tables</li><li>▪ Tabular view improves conceptual simplicity</li><li>▪ Ad hoc query capability is based on SQL</li><li>▪ Isolates the end user from physical-level details</li><li>▪ Improves implementation and management simplicity</li></ul>	<ul style="list-style-type: none"><li>▪ Requires substantial hardware and system software overhead</li><li>▪ Conceptual simplicity gives untrained people the tools to use a good system poorly</li><li>▪ May promote information problems</li></ul>

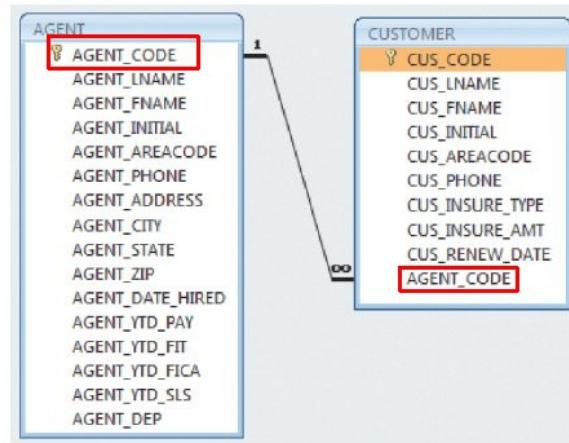
# Relational DBMS

## 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

# Relation - BETWEEN entities

Figure 2.2 - A Relational Diagram



Note: this relation is NOT what relational modeling is about!! Here, we relate two entities, via a common attribute (AGENT\_CODE, in our example).

# SQL + RDBMS

## 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

# E-R

## The Entity Relationship Model

- 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
- **Connectivity:** Term used to label the relationship types

# E-R

## Entity Relationship Model

### Advantages

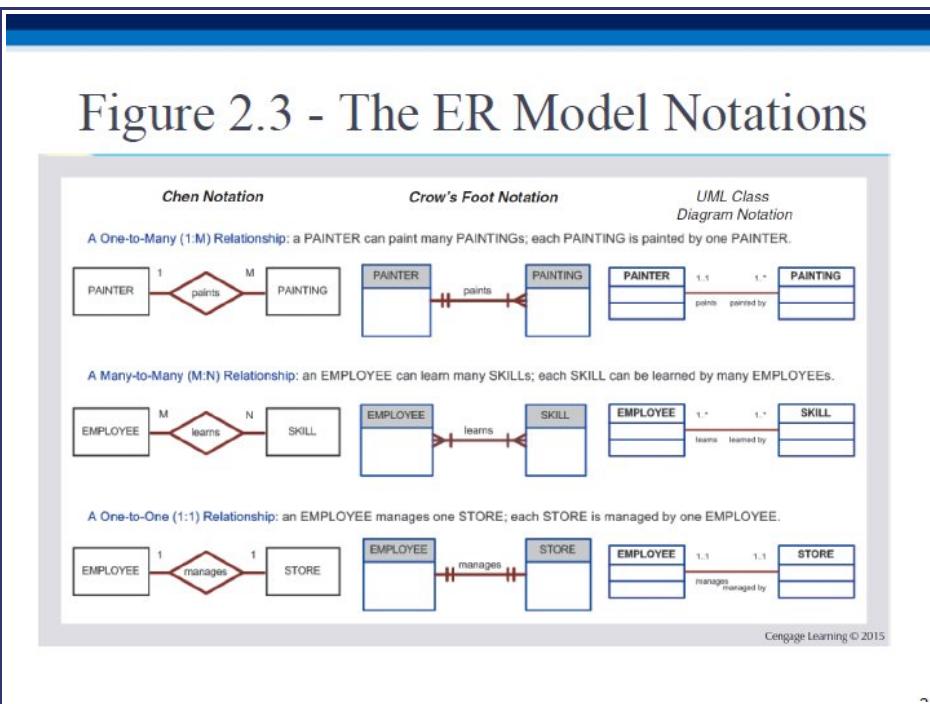
- Visual modeling yields conceptual simplicity
- Visual representation makes it an effective communication tool
- Is integrated with the dominant relational model

### 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

# Notations

Figure 2.3 - The ER Model Notations



# Notations - more..

Additional reading: [here](#) is information on, and comparison between, four ER notations: Chen, Crow, Rein85, IDEFIX.

# O-O databases

Also called 'object stores', these dbs offer(ed) a way to store ("persist") objects on disk. The objects (entity instances) are instanced from classes (entities), like with standard OO programming practice.

Advantages:

- 'cleaner' design - objects mimic real-world counterparts
- inheritance and encapsulation possible
- richer datatypes (attributes) available
- good for CAD, multimedia..

Drawbacks:

- harder to query (compared to relational DBs) - no straightforward way to build and traverse relations between objects
- relations are simpler in certain situations

The RDBMS community collectively ignored this development..

# O-R databases

These are a compromise between RDBs and OODBs - they feature an O-O front-end over a relational architecture. Interfacing applications do so in an O-O way, and queries/modifications are translated to/from relational form ("ORM").

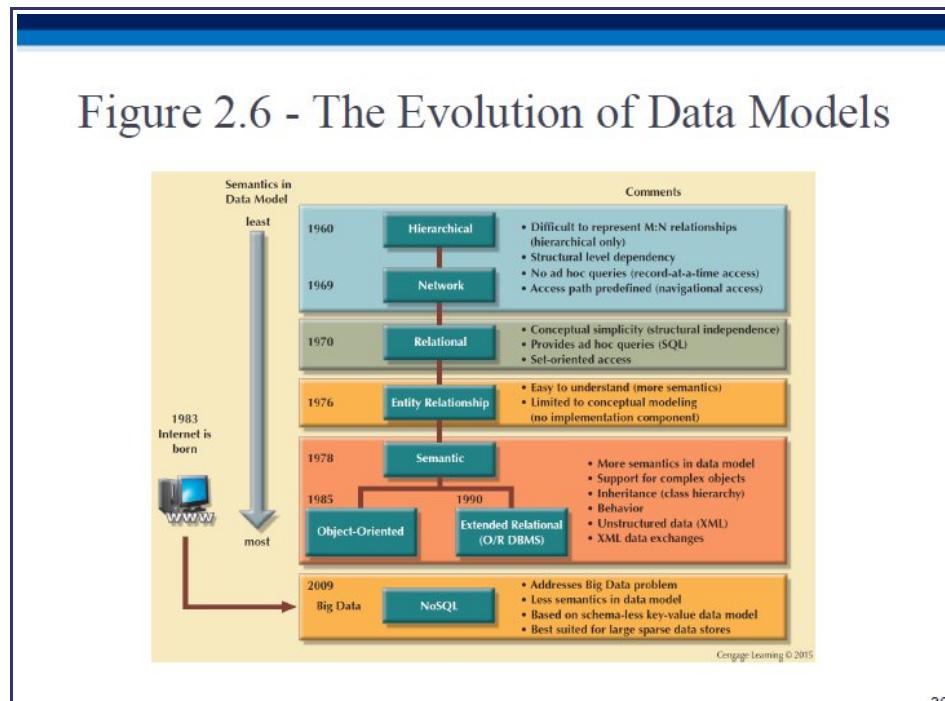
Benefits:

- easy to access the data from an O-O application
- queries can be simpler (can use objects' structure)

Drawback:

- performance can be poor on account of the two-way translation

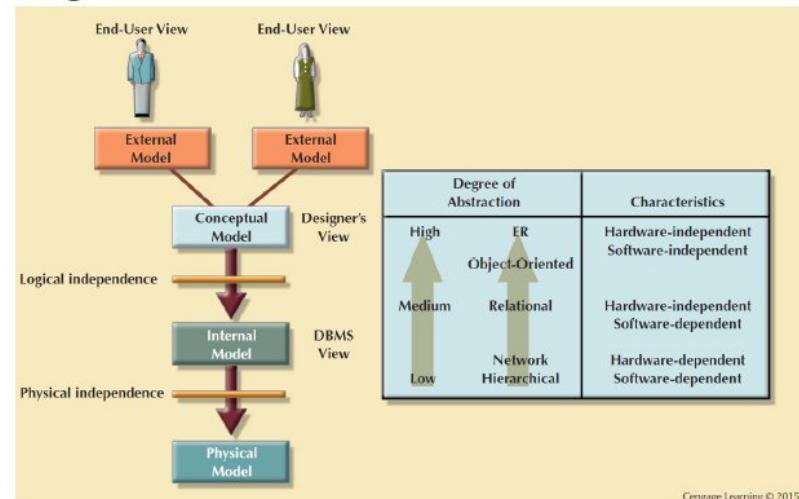
# Data models: hierarchical => => NoSQL



Data models have evolved - from 'hierarchical' (very rigid) to 'NoSQL' (VERY flexible).

# Layered data abstraction

Figure 2.7 - Data Abstraction Levels



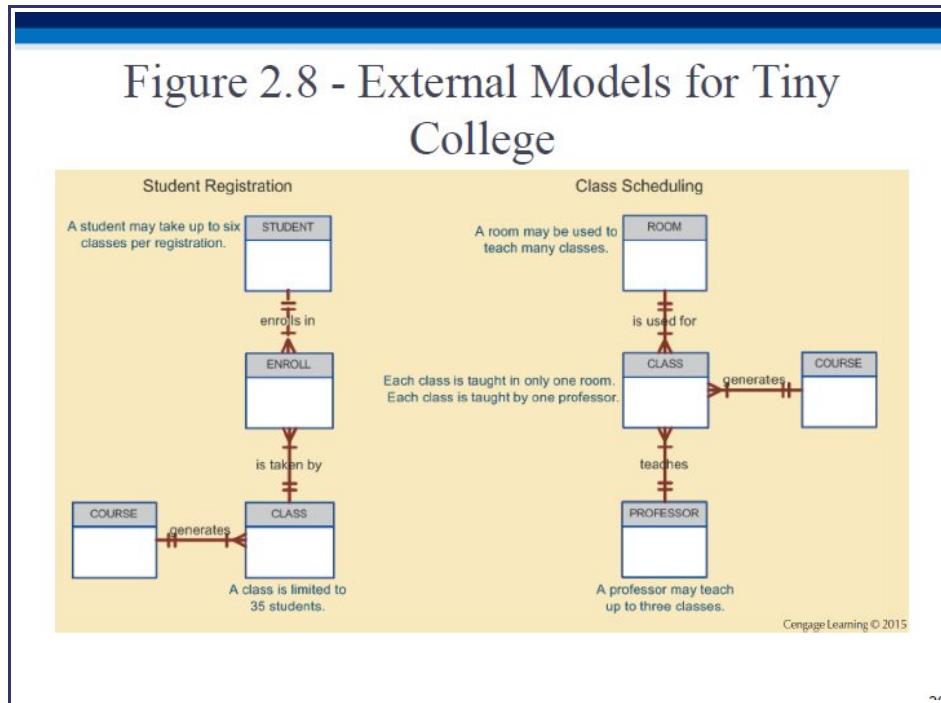
# External model

## The External Model

- End users' view of the data environment
- ER diagrams are used to represent the external views
- **External schema:** Specific representation of an external view

An external model is a collection of 'fragmented', 'from the stakeholders' POV', modeling of a database.

# External model



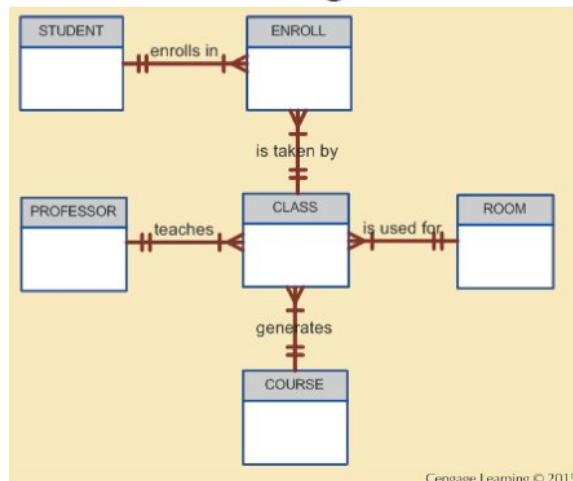
# Conceptual model

## The Conceptual Model

- 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
- Has a macro-level view of data environment
- Is software and hardware independent
- **Logical design:** Task of creating a conceptual data model

# Conceptual model

Figure 2.9 - Conceptual Model for Tiny College



A conceptual model unifies the external views into a cohesive one.

# Internal model

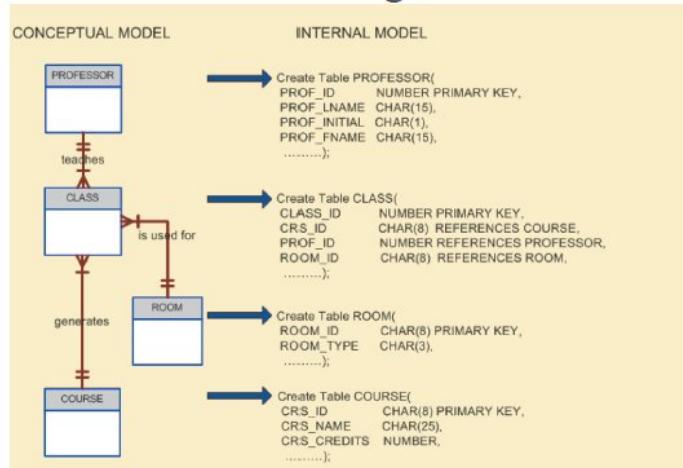
## The Internal Model

- Representing database as seen by the DBMS mapping conceptual model to the DBMS
- **Internal schema:** Specific representation of an internal model
  - Uses the database constructs supported by the chosen database
- Is software dependent and hardware independent
- **Logical independence:** Changing internal model without affecting the conceptual model

An internal model specifies what type of modeling (eg. relational, NoSQL...) to use for storing the data.

# Internal model

Figure 2.10 - Internal Model for Tiny College



# Physical model

## The Physical Model

- Operates at lowest level of abstraction
- Describes the way data are saved on storage media such as disks or tapes
- Requires the definition of physical storage and data access methods
- 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 specifies actual data storage specifics (file format, APIs...).