Chapter 2

Objectives: to understand

- Data modeling and why data models are important
- The basic data-modeling building blocks
- What business rules are and how they influence database design
- How the major data models evolved historically
- How data models can be classified by level of abstraction

Introduction to Data Modeling

- Data modeling reduces complexities of database design
- Designers, programmers, and end users see data in different ways
- Different views of same data lead to designs that do not reflect organization's operation
- Various degrees of data abstraction help reconcile varying views of same data

Data Modeling and Data Models

- Model: an abstraction of a real-world object or event
  - Useful in understanding complexities of the real-world environment
- Data models
  - Relatively simple representations of complex real-world data structures
    - Often graphical
- Creating a Data model is iterative and progressive

The Importance of Data Models

- Facilitate interaction among the designer, the applications programmer, and the end user
- End users have different views and needs for data
- Data model organizes data for various users
- Data model is a conceptual model - an abstraction
- It's a graphical collection of logical constructs representing the data structure and relationships within the database.
  - Cannot draw required data out of the data model
  - An implementation model would represent how the data are represented in the database.

Data Model Basic Building Blocks Terminology

- Entity: anything about which data are to be collected and stored
- Attribute: a characteristic of an entity
- Relationship: describes an association among entities
  - One-to-many (1:M) relationship
  - Many-to-many (M:N or M:M) relationship
  - One-to-one (1:1) relationship
- Constraint: a restriction placed on the data

Business Rules

- Descriptions of policies or principles within an organization
- Description of operations or procedures, to create/enforce actions within an organization's environment
  - Must be in writing and kept up to date
  - Must be easy to understand and widely disseminated
  - Sometimes externally defined, i.e. government regulations.
- These describe characteristics of data as viewed by the company

Discovering Business Rules

- Sources of business rules:
  - Company managers
  - Policy makers
  - Department managers
  - Written documentation
    - Procedures
    - Standards
    - Operations manuals
  - Direct interviews with end users
- Always verify sources of information

Importance of Business Rules

- Standardize company's view of data
- Useful as a communications tool between users and designers
- Allows the designer to
  - understand the nature, role, and scope of data
  - understand business processes
  - develop appropriate relationship participation rules and constraints
- Promotes the creation of an accurate data model

Translating Business Rules into Data Model Components

- Generally, nouns translate into entities
- Verbs translate into relationships among entities
- Relationships are bidirectional
- Two 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

- Naming occurs during translation of business rules to data model components
- Names should make the object unique and distinguishable from other objects
- Names should also be descriptive of objects in the environment and be familiar to users
- Proper naming:
  - Facilitates communication between parties
  - Promotes self-documentation

Evolution of Data Implementation Models

- Hierarchical
  - Logically represented by an upside down tree
    - Each parent can have many children
    - Each child has only one parent
- Network
- Relational
- Object oriented
- Hybrid, XML

TABLE 2.1 Evolution of Major Data Models

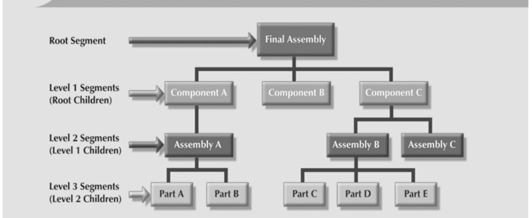
GENERATION	TIME	DATA MODEL	EXAMPLES	COMMENTS
First	1960s–1970s	File system	VMS/VSAM	Used mainly on IBM mainframe systems Managed records, not relationships
Second	1970s	Hierarchical and network	IMS ADABAS IDS-II	Early database systems Navigational access
Third	Mid-1970s to present	Relational	DB2 Oracle MS SQL-Server MySQL	Conceptual simplicity Entity relationship (ER) modeling and support for relational data modeling
Fourth	Mid-1980s to present	Object-oriented Object/relational (O/R)	Versant Objectivity/DB DB/2 UDB Oracle 11g	Object/relational supports object data types Star Schema support for data warehousing Web databases become common
Next generation	Present to future	XML Hybrid DBMS	dbXML Tamino DB2 UDB Oracle 11g MS SQL Server	Unstructured data support O/R model supports XML documents Hybrid DBMS adds an object front end to relational databases

The Hierarchical Model

- The hierarchical model was developed in the 1960s to manage large amounts of data for manufacturing projects
- Basic logical structure is represented by an upside-down "tree"
- Hierarchical structure contains levels or segments
  - Segment analogous to a record type
  - Set of one-to-many relationships between segments
- Example – manufacturing a car from components (a,b,or c), each made of subassemblies (1,2,or3), each having parts (x,y,&z) ....(tree structure)

Hierarchical Structure

FIGURE 2.1 A hierarchical structure



Hierarchical Structure

- Each parent can have many children
- Each child has only one parent
- Tree is defined by path that traces parent segments to child segments, beginning from the left
- Hierarchical path
  - Ordered sequencing of segments tracing hierarchical structure
- Preorder traversal or hierarchic sequence
  - "Left-list" path

The Hierarchical Model

- GUAM (Generalized Update Access Method)
  - Based on the recognition that the many smaller parts would come together as components of still larger components
- Information Management System (IMS)
  - World's leading mainframe hierarchical database system in the 1970s and early 1980s
- TCDMS/ADABAS – jointly developed by IBM and Lane County

The Hierarchical Model

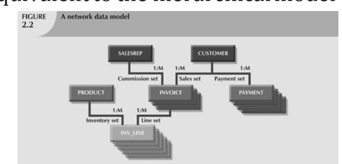
- Advantages
  - Conceptual simplicity
  - Database security
  - Data independence
  - Database integrity
  - Efficiency
- Disadvantages
  - Complex implementation
  - Difficult to manage
  - Lacks structural independence
  - Complex applications programming and use
  - Implementation limitations
  - Lack of standards

The Network Model

- The network model was created to represent complex data relationships more effectively than the hierarchical model
  - Improves database performance
  - Imposes a database standard
  - Represent complex data relationships more effectively – such as child w/ multiple parents
- Conference on Data Systems Languages (CODASYL)
- American National Standards Institute (ANSI)
- Database Task Group (DBTG)

The Network Model

- Collection of records in 1:M relationships
- A Set is a relationship and composed of two record types:
  - Owner: Equialent to the hierarchical model's parent
  - Member: Equivalent to the hierarchical model's child



The Network Model Components

- Concepts still used today:
  - ***Schema:*** Conceptual organization of entire database as viewed by the database administrator
  - ***Subschema***: Database portion "seen" by the application programs
  - ***Data management language*** (DML): Defines the environment in which data can be managed
  - ***Data definition language*** (DDL): Enables the administrator to define the schema components

The Network Model

- Advantages:
  - Conformance to standards
  - Handled more relationship types
  - Data access flexibility
- Disadvantages of the network model:
  - System complexity
  - Lack of ad hoc query capability placed burden on programmers to generate code for reports
  - Structural change in the database could produce havoc in all application programs

The Relational Model

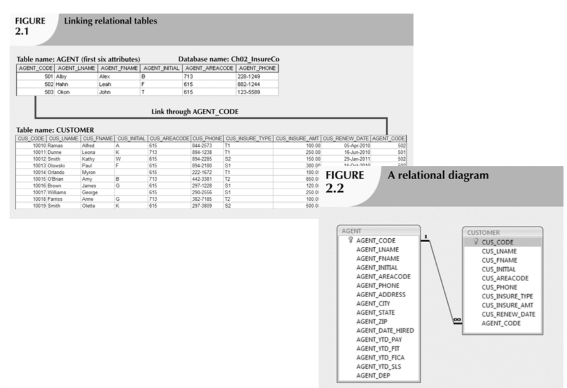
- Developed by E.F. Codd (IBM) in 1970
- Relational models were considered impractical in the 1970's.
- Model was conceptually simple at expense of computer overhead
- Relational table is purely logical structure
  - How data are physically stored in the database is of no concern to the user or the designer
  - *This concept is the source of a real database revolution*

Relational Table

- A Relational table is a purely logical structure
  - How data are physically stored in the database is of no concern to the user or the designer.
- Stores a collection of related entities
  - Resembles a file
- Table (relations)
  - Matrix consisting of a series of row/column intersections
  - Each row in a relation is called a tuple
  - Related to each other by sharing a common entity characteristic

The Relational Model Components

- Relational data management system (RDBMS)
  - Performs same functions provided by hierarchical model, but hides complexity from the user
- Relational schema/diagram
  - Visual representation of relational database's entities, attributes within those entities, and relationships between those entities
- Relational diagram
  - Representation of entities, attributes, and relationships
- Relational table stores collection of related entities.



The Relational DBMS Application

- SQL-based relational database application involves three parts:
  - User interface
    - Allows end user to interact with the data
  - Set 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 Relational Implementation Model

- Advantages
  - Structural independence
  - Improved conceptual simplicity
  - Easier database design, implementation, management, and use
  - Ad hoc query capability (SQL)
  - Powerful database management system
- Disadvantages
  - Substantial hardware and system software overhead
  - Can facilitate poor design and implementation
  - May promote "islands of information" problems

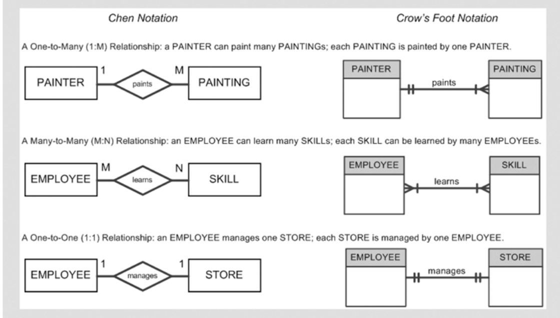
Logical/Conceptual Model The Entity Relationship Model

- Widely accepted standard for data modeling
- Introduced by Chen in 1976
- Graphical representation of entities and their relationships in a database structure
- Entity relationship diagram (ERD)
  - Uses graphic representations to model database components
  - Entity is mapped to a relational table

The Entity Relationship Model

- Entity instance (or occurrence) is row in table
- Entity set is collection of like entities
- Connectivity labels types of relationships
- Relationships are expressed using Chen notation
  - Relationships are represented by a diamond
  - Relationship name is written inside the diamond
- Crow's Foot notation used as design standard in this book

FIGURE 2.3 The Chen and Crow's Foot notations



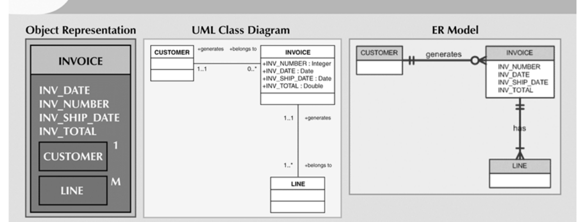
Logical/Conceptual Model
The Object-Oriented (OO) Model

- Models both data and relationships contained in a single structure known as an object
- OODM (object-oriented data model) is the basis for OO-DBMS (Semantic data model)
- An object is described by its factual content:
  - Are self-contained: a basic building-block for autonomous structures
  - Is an abstraction of a real-world entity
  - Contains information about relationships between facts within the object and with other objects.

The Object-Oriented (OO) Model

- An Object is the logical abstraction or basic building block for autonomous structures
  - Attributes describe the properties of an object
  - Objects that share similar characteristics are grouped in classes
  - Classes are organized in a class hierarchy
  - Inheritance: an object inherits methods and attributes of parent class
  - UML - Unified Modeling Language is used to graphically model a system
    - based on OO concepts that describe diagrams and symbols

FIGURE 2.4 A comparison of OO, UML, and ER models



Logical Models: Object Oriented Model

- Advantages
  - Adds semantic content
  - Visual presentation includes semantic content
  - Database integrity
  - Both structural and data independence
- Disadvantages
  - Slow pace of OODM standards development
  - Complex navigational data access
  - Steep learning curve
  - High system overhead slows transactions
  - Lack of market penetration

Newer Data Models: Object/Relational

- Extended relational data model (ERDM)
  - Semantic data model developed in response to increasing complexity of applications
  - Includes many of OO model's best features
  - Often described as an object/relational database management system (O/RDBMS)
  - Primarily geared to business applications

Newer Data Models: XML

- The Internet revolution created the potential to exchange critical business information
- Dominance of Web has resulted in growing need to manage unstructured information
- In this environment, Extensible Markup Language (XML) emerged as the de facto standard
- Current databases support XML
  - XML: the standard protocol for data exchange among systems and Internet services

The Future of Data Models

- Hybrid DBMSs
  - Retain advantages of relational model
  - Provide object-oriented view of the underlying data
- SQL data services – 'Cloud Computing'
  - Store data remotely without incurring expensive hardware, software, and personnel costs
  - Companies operate on a "pay-as-you-go" system

The Development of Data Models

FIGURE 2.5 The evolution of data models

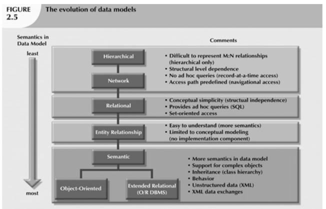


TABLE 2.3 Data Model Basic Terminology Comparison

REAL WORLD	EXAMPLE	FILE PROCESSING	HIERARCHICAL MODEL	NETWORK MODEL	RELATIONAL MODEL	ER MODEL	OO MODEL
A group of vendors	Vendor file cabinet	File	Segment type	Record type	Table	Entity Set	Class
A single vendor	Global supplies	Record	Segment occurrence	Current record	Row (tuple)	Entity occurrence	Object instance
The contact name	Johnny Ventura	Field	Segment field	Record field	Table attribute	Entity Attribute	Object attribute
The vendor identifier	G12987	Index	Sequence field	Record key	Key	Entity Identifier	Object identifier

Note: For additional information about the terms used in this table, please consult the corresponding chapters and online appendixes accompanying this book. For example, if you want to know more about the OO model, refer to Appendix G, Object-Oriented Databases.

Data Models: A Summary

- Each new data model capitalized on the shortcomings of previous models
- Common characteristics:
  - Conceptual simplicity with semantic completeness
  - Represent the real world as closely as possible
  - Real-world transformations (behavior) must comply with consistency and integrity characteristics
- Some models better suited for some tasks

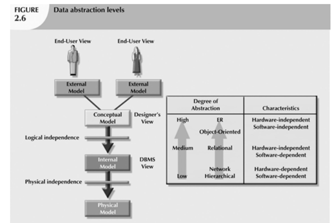
SPARC Framework : Degrees of Data Abstraction

- Database designer starts with abstracted view, then adds details
- ANSI Standards Planning and Requirements Committee (SPARC)
  - Defined a framework for data modeling based on degrees of data abstraction (1970s):
    1. External
    2. Conceptual
    3. Internal

The SPARC External Model

- Represents the End users' view of the data environment
- ER diagrams represent external views
- External schema: specific representation of an external view
  - Entities
  - Relationships
  - Processes
  - Constraints

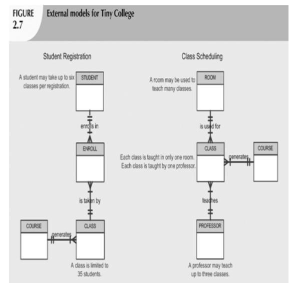
FIGURE 2.6 Data abstraction levels



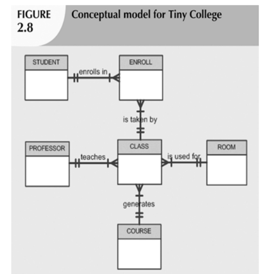
The External Model

- End users' view of the data environment
- Requires that the modeler subdivide set of requirements and constraints into functional modules that can be examined within the framework of their external models
- Advantages:
  - Easy to identify specific requirements to support each business unit's operations
  - Facilitates designer's job by providing feedback about the model's adequacy
  - Ensures security constraints in database design
  - Simplifies application program development

External Models showing two different Users



Conceptual Model



The SPARC Conceptual Model

- Represents global view of the entire database
  - All external views integrated into single global view: conceptual schema
- Representation of data as viewed by high-level managers
- ER Diagram graphically represents the conceptual schema
  - ER model most widely used conceptual model
- Basis for identification and description of main data objects, avoiding details

The Conceptual Model

Advantages

- Provides a relatively easily understood macro level view of data environment
- Independent of both software and hardware
  - Does not depend on the DBMS software used to implement the model
  - Does not depend on the hardware used in the implementation of the model
  - Changes in hardware or software or do not affect database design at the conceptual level

The SPARC Internal Model

- Representation of the database as “seen” by the DBMS
  - Maps the Conceptual model to the DBMS
- Internal schema depicts a specific representation of an internal model
- Depends on specific database software
  - Change in DBMS software requires internal model be changed
- Logical independence: change internal model without affecting conceptual model

FIGURE 2.9 Internal model for Tiny College

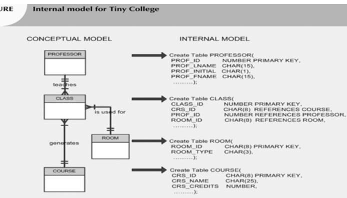


TABLE 2.4 Levels of Data Abstraction

MODEL	DEGREE OF ABSTRACTION	FOCUS	INDEPENDENT OF
External	High	End-user views	Hardware and software
Conceptual	↕	Global view of data (database model–independent)	Hardware and software
Internal		Specific database model	Hardware
Physical	Low	Storage and access methods	Neither hardware nor software

The Physical Model

- Operates at lowest level of abstraction
  - Describes the way data are saved on storage media such as disks or tapes
  - Software and hardware dependent
- 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

Summary

- A data model is an abstraction of a complex real-world data environment
- Basic data modeling components:
  - Entities
  - Attributes
  - Relationships
  - Constraints
- Business rules identify and define basic modeling components

Summary

- Hierarchical model
  - Set of one-to-many (1:M) relationships between a parent and its children segments
- Network data model
  - Uses sets to represent 1:M relationships between record types
- Relational model
  - Current database implementation standard
  - ER model is a tool for data modeling
    - Complements relational model

Summary

- Object-oriented data model: object is basic modeling structure
- Relational model adopted object-oriented extensions: extended relational data model (ERDM)
- OO data models depicted using UML
- Data-modeling requirements are a function of different data views and abstraction levels
  - Three SPARC abstraction levels: external, conceptual, internal