Objectives

- About data modeling (資料塑模) and why data models (資料模型) are important
- About the basic data-modeling building blocks
- What business rules (企業規則) are and how they influence database design
- How the major data models evolved
- How data models can be classified by level of abstraction

2.1 Introduction

- 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
- Data modeling reduces complexities of database design
- Various degrees of data abstraction help reconcile varying views of same data

Data Modeling and Data Models

- Data models
  - Relatively simple representations of complex real-world data structures
    - Often graphical
- Model: an abstraction of a real-world object or event
  - Useful in understanding complexities of the real-world environment
- Data modeling is iterative (反覆進行) and progressive (循序漸進)
2.2 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 an abstraction
  - Cannot draw required data out of the data model

2.3 Data Model Basic Building Blocks

- **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

2.4 Business Rules

- Descriptions of policies, procedures, or principles within a specific organization
  - Apply to any organization that stores and uses data to generate information
- Description of operations 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
- 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
Discovering Business Rules (continued)

- Standardize company's view of data
- Communications tool between users and designers
- Allow designer to understand the nature, role, and scope of data
- Allow designer to understand business processes (企業流程)
- Allow designer to develop appropriate relationship participation rules and constraints

Translating Business Rules into Data Model Components

- Generally, nouns (名詞) translate into entities
- Verbs (動詞) translate into relationships among entities
- Relationships are bidirectional (雙向)
- 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?

2.5 The Evolution of Data Models

<table>
<thead>
<tr>
<th>GENERATION</th>
<th>TIME</th>
<th>MODEL</th>
<th>EXAMPLES</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>1960s−1970s</td>
<td>File System</td>
<td>VM/SMS/VSAM</td>
<td>Used mainly on IBM mainframe system Managed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>records, not relationships</td>
</tr>
<tr>
<td>Second</td>
<td>1970s</td>
<td>Hierarchical and</td>
<td>IMS, ADABAS, IDS-II</td>
<td>Early database systems Navigational access</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Network Data Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Third</td>
<td>Mid-1970s to present</td>
<td>Relational Data Model</td>
<td>DB2, Oracle, MS SQL-Server, MySQL</td>
<td>Conceptual simplicity Entity-Relationship (ER) modeling and support for relational database modeling</td>
</tr>
<tr>
<td>Fourth</td>
<td>Mid-1980s to present</td>
<td>Object-Oriented</td>
<td>Versa, FastObjects.Net, Objectivity, DB2, UDB, Oracle 10g</td>
<td>Support complex data Extended relational products support objects and data warehousing Web databases become common</td>
</tr>
<tr>
<td>Next Generation</td>
<td>Present to future</td>
<td>XML</td>
<td>dXML, Tamino, DB2, UDB, Oracle 10g, MS SQL-Server</td>
<td>Organization and management of unstructured data Relational and object models add support for XML documents</td>
</tr>
</tbody>
</table>

2.5.3 The Relational Model

- Developed by E. F. Codd (IBM) in 1970
- Table (relations)
  - Matrix consisting of row/column intersections
  - Each row in a relation called a tuple (元組, 值組)
- Relational models considered impractical in 1970
- Model conceptually simple at expense of computer overhead
The Relational Model (continued)

• Relational data management system (RDBMS)
  – Performs same functions provided by hierarchical model
  – Hides complexity from the user
• Relational diagram
  – Representation of entities, attributes, and relationships
• Relational table stores collection of related entities

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 related based on common values in common attributes
  – SQL “engine” (引擎)
    • Executes all queries
2.5.4 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, 實體關係圖)
  - Uses graphic representations to model database components
  - Entity is mapped to a relational table

The Entity Relationship Model (continued)

- Entity instance (實例，或 occurrence，發生值) is row (列) in table
- Entity set is collection of like entities
- Connectivity (連結性) labels types of relationships
- Relationships expressed using Chen notation
  - Relationships represented by a diamond (菱形)
  - Relationship name written inside the diamond
- Crow’s Foot (鴉腳) notation used as design standard in this book

2.5.5 The Object-Oriented (OO) Model

- Data and relationships contained in single structure known as an object
- OODM (object-oriented data model) is the basis for OODBMS
  - Semantic (語意) data model
- Objects contain operations
- Object is self-contained: a basic building-block for autonomous structures
- Object is an abstraction of a real-world entity
The Object-Oriented (OO) Model (continued)

- Attributes describe the properties of an object
- Objects that share similar characteristics are grouped in classes
- Classes are organized in a class hierarchy
- Inheritance (繼承): object inherits methods and attributes of parent class
- UML (Unified Modeling Language, 統一模型語言) based on OO concepts that describe diagrams and symbols
  - Used to graphically model a system

2.5.6 The Convergence of Data Models

- Extended (擴充型) 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

2.5.7 Database Models and the Internet

- Internet drastically changed role and scope of database market
- Focus on Internet makes underlying data model less important
- Dominance of Web has resulted in growing need to manage unstructured/semi-structured information
- Current databases support XML
  - XML: the standard protocol for data exchange among systems and Internet services
Data Models: A Summary

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

---

**Advantages and Disadvantages of Data Models**

<table>
<thead>
<tr>
<th>Data Model</th>
<th>Data Indep.</th>
<th>Structural Indep.</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational</td>
<td>Yes</td>
<td>Yes</td>
<td>1. Structural Independence by independent tables 2. Tabular view improves conceptual simplicity 3. Ad hoc query based on SQL 4. Powerful RDBMS</td>
<td>1. hardware and system software overhead 2. gives untrained people to use a good system poorly 3. may produce &quot;islands of info&quot; problem</td>
</tr>
<tr>
<td>Entity Relationship</td>
<td>Yes</td>
<td>Yes</td>
<td>1. visual modeling yields conceptual simplicity 2. effective communication tool 3. integrated with dominant relational model</td>
<td>1. limited constraint specification 2. limited relationship representation 3. no data manipulation language 4. loss of information when attributes are removed</td>
</tr>
</tbody>
</table>
2.6 Degrees of Data Abstraction (抽象)

- 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):
    - External (外部模型)
    - Conceptual (概念模型)
    - Internal (內部模型)

The External Model

- End users’ view of the data environment
- ER diagrams represent external views
- External schema (外部結構描述): specific representation of an external view
  - Entities
  - Relationships
  - Processes
  - Constraints

The External Model (continued)

- Easy to identify specific data required 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
The Conceptual Model

- Represents global view of the entire database
- All external views integrated into single global view: conceptual schema
- ER model most widely used
- ERD graphically represents the conceptual schema

The Conceptual Model (continued)

- 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 do not affect database design at the conceptual level
The 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

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
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 (continued)

• 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 (continued)

• 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 abstraction levels: external, conceptual, internal