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
Sources of Business Rules

- Company managers
- Policy makers
- Department managers
- Written documentation
- Direct interviews with end users
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 and Network Models

<table>
<thead>
<tr>
<th>Hierarchical Models</th>
<th>Network Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage large amounts of data for complex manufacturing projects</td>
<td>Represent complex data relationships</td>
</tr>
<tr>
<td>Represented by an upside-down tree which contains segments</td>
<td>Improve database performance and impose a database</td>
</tr>
<tr>
<td>▪ Segments: Equivalent of a file system’s record type</td>
<td>standard</td>
</tr>
<tr>
<td>▪ Depicts a set of one-to-many (1:M) relationships</td>
<td>▪ Depicts both one-to-many (1:M) and many-to-many</td>
</tr>
<tr>
<td></td>
<td>(M:N) relationships</td>
</tr>
</tbody>
</table>
## Hierarchical Model

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

Data manipulation language (DML)

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

Schema data definition language (DDL)

- Enables the database administrator to define the schema components
The Relational Model

- Produced an automatic transmission database that replaced standard transmission databases
- 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

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

©2015 Cengage Learning. All Rights Reserved. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.
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
Figure 2.2 - A Relational Diagram
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

- 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
## Entity Relationship Model

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Visual modeling yields conceptual simplicity</td>
<td>- Limited constraint representation</td>
</tr>
<tr>
<td>- Visual representation makes it an effective communication tool</td>
<td>- Limited relationship representation</td>
</tr>
<tr>
<td>- Is integrated with the dominant relational model</td>
<td>- No data manipulation language</td>
</tr>
<tr>
<td></td>
<td>- Loss of information content occurs when attributes are removed from entities to avoid crowded displays</td>
</tr>
</tbody>
</table>
Figure 2.3 - The ER Model Notations

Chen Notation

A One-to-Many (1:M) Relationship: a PAINTER can paint many PAINTINGS; each PAINTING is painted by one PAINTER.

A Many-to-Many (M:N) Relationship: an EMPLOYEE can learn many SKILLS; each SKILL can be learned by many EMPLOYEES.

A One-to-One (1:1) Relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.
The Object-Oriented Data Model (OODM) or Semantic Data Model

- **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

- **Attributes**: Describe the properties of an object
The Object-Oriented Data Model (OODM)

- **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 parent class
- **Unified Modeling Language (UML)**
  - Describes sets of diagrams and symbols to graphically model a system
## Object-Oriented Model

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic content is added</td>
<td>Slow development of standards caused vendors to supply their own enhancements</td>
</tr>
<tr>
<td>Visual representation includes semantic content</td>
<td>Compromised widely accepted standard</td>
</tr>
<tr>
<td>Inheritance promotes data integrity</td>
<td>Complex navigational system</td>
</tr>
<tr>
<td></td>
<td>Learning curve is steep</td>
</tr>
<tr>
<td></td>
<td>High system overhead slows transactions</td>
</tr>
</tbody>
</table>

©2015 Cengage Learning. All Rights Reserved. May not be scanned, copied or duplicated, or posted to a publicly accessible website, in whole or in part.
Figure 2.4 - A Comparison of OO, UML, and ER Models
Object/Relational and XML

- Extended relational data model (ERDM)
  - Supports OO features and complex data representation
  
- Object/Relational Database Management System (O/R DBMS)
  - Based on ERDM, focuses on better data management

- Extensible Markup Language (XML)
  - Manages unstructured data for efficient and effective exchange of all data types
Big Data

- Aims to:
  - Find new and better ways to manage large amounts of web and sensor-generated data
  - Provide high performance and scalability at a reasonable cost

- Characteristics
  - Volume
  - Velocity
  - Variety
Big Data Challenges

- Volume does not allow the usage of conventional structures
- Expensive
- OLAP tools proved inconsistent dealing with unstructured data
Big Data New Technologies

- Hadoop
- Hadoop Distributed File System (HDFS)
- MapReduce
- NoSQL
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
- Store data in key-value stores
NoSQL

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>• High scalability, availability, and fault tolerance are provided</td>
<td>• Complex programming is required</td>
</tr>
<tr>
<td>• Uses low-cost commodity hardware</td>
<td>• There is no relationship support</td>
</tr>
<tr>
<td>• Supports Big Data</td>
<td>• There is no transaction integrity support</td>
</tr>
<tr>
<td>• 4. Key-value model improves storage efficiency</td>
<td>• In terms of data consistency, it provides an eventually consistent model</td>
</tr>
</tbody>
</table>
pleease check the line marked in red. i didnt understand why its an disadvantage, given same in pdf
Tejas Iyer, 1/29/2014
Figure 2.5 - A Simple Key-value Representation

In the relational model:
- Each row represents one entity instance.
- Each column represents one attribute of the entity.
- The values in a column are of the same data type.

In the key-value model:
- Each row represents one attribute/value of one entity instance.
- The “key” column could represent any entity’s attribute.
- The values in the “value” column could be of any data type and therefore it is generally assigned a long string data type.

Trucks-R-Us

Data stored using traditional relational model

<table>
<thead>
<tr>
<th>DID</th>
<th>CERT1</th>
<th>CERT2</th>
<th>CERT3</th>
<th>DOB</th>
<th>LCTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2732</td>
<td>80</td>
<td>95</td>
<td>1/24/1962</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>2946</td>
<td>92</td>
<td></td>
<td>4/11/1970</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3650</td>
<td>86</td>
<td></td>
<td>11/27/1983</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data stored using key-value model

<table>
<thead>
<tr>
<th>DID</th>
<th>KEY</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2732</td>
<td>CERT1</td>
<td>80</td>
</tr>
<tr>
<td>2732</td>
<td>CERT3</td>
<td>95</td>
</tr>
<tr>
<td>2732</td>
<td>DOB</td>
<td>1/24/1962</td>
</tr>
<tr>
<td>2732</td>
<td>LCTYPE</td>
<td>P</td>
</tr>
<tr>
<td>2946</td>
<td>CERT2</td>
<td>92</td>
</tr>
<tr>
<td>2946</td>
<td>DOB</td>
<td>4/11/1970</td>
</tr>
<tr>
<td>3650</td>
<td>CERT1</td>
<td>86</td>
</tr>
<tr>
<td>3650</td>
<td>DOB</td>
<td>11/27/1983</td>
</tr>
<tr>
<td>3650</td>
<td>LCTYPE</td>
<td>R</td>
</tr>
</tbody>
</table>
Figure 2.6 - The Evolution of Data Models

<table>
<thead>
<tr>
<th>Year</th>
<th>Data Model</th>
<th>Semantics in Data Model</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1960</td>
<td>Hierarchical</td>
<td>least</td>
<td>Difficult to represent M:N relationships (hierarchical only)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Structural level dependency</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No ad hoc queries (record-at-a-time access)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Access path predefined (navigational access)</td>
</tr>
<tr>
<td>1969</td>
<td>Network</td>
<td>least</td>
<td></td>
</tr>
<tr>
<td>1970</td>
<td>Relational</td>
<td>least</td>
<td>Conceptual simplicity (structural independence)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Provides ad hoc queries (SQL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Set-oriented access</td>
</tr>
<tr>
<td>1976</td>
<td>Entity Relationship</td>
<td>least</td>
<td>Easy to understand (more semantics)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Limited to conceptual modeling (no implementation component)</td>
</tr>
<tr>
<td>1978</td>
<td>Semantic</td>
<td>most</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>More semantics in data model</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Support for complex objects</td>
</tr>
<tr>
<td>1985</td>
<td>Object-Oriented</td>
<td>must</td>
<td>Inheritance (class hierarchy)</td>
</tr>
<tr>
<td>1990</td>
<td>Extended Relational</td>
<td>must</td>
<td>Behavior</td>
</tr>
<tr>
<td></td>
<td>(O/R DBMS)</td>
<td></td>
<td>Unstructured data (XML)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>XML data exchanges</td>
</tr>
<tr>
<td>2009</td>
<td>Big Data</td>
<td>must</td>
<td>Addresses Big Data problem</td>
</tr>
<tr>
<td></td>
<td>NoSQL</td>
<td>most</td>
<td>Less semantics in data model</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Based on schema-less key-value data model</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Best suited for large sparse data stores</td>
</tr>
</tbody>
</table>

Internet born in 1983
Table 2.3 - Data Model Basic Terminology Comparison

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

Note: For additional information about the terms used in this table, consult the corresponding chapters and online appendices that accompany this book. For example, if you want to know more about the OO model, refer to Appendix G, Object-Oriented Databases.
Figure 2.7 - Data Abstraction Levels

- End-User View
  - External Model
  - Conceptual Model
  - Internal Model
  - Physical Model

- Designer’s View
  - DBMS View

- Degree of Abstraction
  - High
  - Medium
  - Low

- Characteristics
  - Hardware-independent
  - Software-independent
  - Hardware-dependent
  - Software-dependent

- ER Object-Oriented
- Network Hierarchical
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
Figure 2.8 - External Models for Tiny College

Student Registration
A student may take up to six classes per registration.

Class Scheduling
A room may be used to teach many classes.

Each class is taught in only one room. Each class is taught by one professor.

A professor may teach up to three classes.
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
Figure 2.9 - Conceptual Model for Tiny College
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
Figure 2.10 - Internal Model for Tiny College

CONCEPTUAL 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 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
Table 2.4 - Levels of Data Abstraction

<table>
<thead>
<tr>
<th>MODEL</th>
<th>DEGREE OF ABSTRACTION</th>
<th>FOCUS</th>
<th>INDEPENDENT OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>High</td>
<td>End-user views</td>
<td>Hardware and software</td>
</tr>
<tr>
<td>Conceptual</td>
<td></td>
<td>Global view of data</td>
<td>Hardware and software</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(database model independent)</td>
<td></td>
</tr>
<tr>
<td>Internal</td>
<td>Low</td>
<td>Specific database model</td>
<td>Hardware</td>
</tr>
<tr>
<td>Physical</td>
<td></td>
<td>Storage and access methods</td>
<td>Neither hardware nor software</td>
</tr>
</tbody>
</table>