Orange Coast College
Business Division
CS/CIS Department
Fall 2004
CIS 182
Introduction to Database Concepts

Instructor
Dr. Martha Malaty
Text & Original Presentations
Chapter 2

Data Models

In this chapter, you will learn:

• Why data models are important
• Basic data-modeling building blocks
• What business rules are and how they affect database design
• How the major data models evolved, and their advantages and disadvantages
• How data models can be classified by level of abstraction
Data Models

• Data are viewed in different ways by different people
• Model: FOLDOC
  – Webster’s Dictionary: “Description or analogy used to visualize something that cannot be directly observed”
  – Abstraction of more complex real-world object or event
• Data model
  – Relatively simple representation, usually graphical, of complex real-world data structures
  – Communications tool to facilitate interaction among the designer, the applications programmer, and the end user
  – Good database design uses an appropriate data model as its foundation
• Database Model:
  – A specific implementation of a data model
Importance of Data Modeling

• Data model organizes data for various users
• End-users have different views and needs for data
• Story time!

• Example: Company’s Data
  – Managers have enterprise-wide view
  – Different managers view data differently
    • Purchasing: Item costs, suppliers
    • Inventory: inventory levels
  – Application programmer: Data location, formatting, reports
Data Model Basic Building Blocks

• Entity
  – Anything about which data are to be collected and stored

• Attribute
  – Characteristic of an entity

• Relationship
  – Describes an association among (two or more) entities
  – Each relationship should have two directions

• Types of relationships
  – One-to-many (1:M) relationship
  – Many-to-many (M:N or M:M) relationship
  – One-to-one (1:1) relationship
Business Rules

• Brief, precise, and unambiguous description of a policy, procedure, or principle within a specific organization’s environment
• Apply to any organization that stores and uses data to generate information
• Describe characteristics of the data as viewed by the company
• Could be a term, fact, derivation or a constraint
• Sometimes are external to the organization
• Description of operations that help create and enforce actions within that organization’s environment
• Must be rendered in writing
• Must be kept up to date
• Must be easy to understand and widely disseminated
Business Rules Examples

- A customer may make many payments on account
- Each payment on account is credited to only one customer
- A machine operator may not work more than 10 hours in any 24 hour period
- A business trip destination must be at least 200 miles away for an airline ticket to be purchased
- A training session can’t be scheduled for fewer than 10 employees or for more than 30 employees
- A company aircraft must have an airframe inspection every 100 flight hours
- A lab can’t be scheduled for demonstration purposes more than one hour per day
- A customer may generate many invoices
- Each invoice is generated by one customer
Business Rules Exercise

• In-class group activity
  – Write the business rules (at least 20) for
    • Academic performance of a student
    • OCC registration office
    • Election process
    • A video rental store
    • Online bookstore
    • A Bank
    • FEMA (Federal Emergency Management Agency)
    • For the company that you work at, if any
Business Rules References

• Books:
  – http://www.kpiusa.com/BRBook/
  – ACM digital library:
    http://portal.acm.org/citation.cfm?id=779520&dl=ACM&coll=portal

• Websites:
  – !! Please print and read the following issues:
    • http://www.tdan.com/i027fe03.htm
      – According to the Business Rules Group,[1] business rules are each
        one of the following kinds:
        » Term -- The application of a single definition to a word or phrase
        » Fact -- The attribution of something to describe a thing: a role it plays, or some other descriptor.
        » Derivation -- An attribute that is derived from other attributes or system variables.
        » Constraint -- A condition that determines what values an attribute or relationship can or must have.
    • http://www.tdan.com/i028fe04.htm
    • http://www.tdan.com/i029ht02.htm
    • http://www.tdan.com/i030ht01.htm
    • http://www.tdan.com/i031hy03.htm
Sources of Business Rules

- Company managers
- Policy makers
- Department managers
- Written documentation
  - Procedures
  - Standards
  - Operations manuals
- Direct interviews with end users
- Exercise:
  - What other sources could be added?
Importance of Business Rules

- Promote creation of an accurate data model
- Standardize company’s view of data
- Constitute a communications tool between users and designers
- Allow designer to
  - understand the nature, role, and scope of data
  - understand business processes
  - develop appropriate relationship, participation rules, and constraints
Evolution of Data Models

• Hierarchical (App. H)
• Network (App. I)
• Relational
• Entity relationship
• Object oriented

• Some Figures of Data Models
Hierarchical Model Evolution

• Hierarchical structures were widely used in the first mainframe database management systems. However, due to their restrictions, they often cannot be used to relate structures that exist in the real world.

• FOLDOC
  – A kind of database management system that links records together like a family tree such that each record type has only one owner
  – File system
  – Can lead to data redundancies
Hierarchical Model Evolution

- GUAM (Generalized Update Access Method)
  - Developed by North American Rockwell
  - Alternative strategy for managing huge data while avoiding data redundancies
  - Allowed introduction of complex pointer systems
  - Based on the recognition that the many smaller parts would come together as components of still larger components

- Information Management System (IMS)
  - Result of joint effort of Rockwell & IBM
  - World’s leading mainframe hierarchical database system in the 1970s and early 1980s
Hierarchical Structure Example

- Manufacturing process

**FIGURE 2.1 A HIERARCHICAL STRUCTURE**
Hierarchical Structure Characteristics

- **Parent-child combinations**
  - 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

- **Example:**
  - Final assembly → Component A → Assembly A → Part A → Part B → Component B → Component C → Part C → Part D
Hierarchical Model Advantages

• Conceptual simplicity
  – Simple Relationship between layers
• Database security
  – Enforced uniformly throughout the system by the DBMS, not by application programmer
• Data integrity
  – Parent/child relationship always links between child & parent segments
• Data independence
  – A change in a data type will automatically cascade throughout the DB
• Efficiency
  – 1:M relationships work more efficiently in hierarchical model
Hierarchical Model Disadvantages

• Complex implementation
  – A navigation structure (Pre-order traversal algorithm) needed to access data segments
  – Designer/programmer needs to know details of physical data storage characteristics
  – Organizing children might affect efficiency

• Difficult to manage & lack of standards
  – Any DB structure changes needs changes in all application programs accessing the DB
  – Deletion of a segment (node) leads to involuntary deletes all its children
    • Needs extreme caution to avoid errors

• Lacks structural independence
  – Data access requires using physical path to navigate to the required data segment (To reach any child, we need to trace the path from the root)
    • Any changes in the structure might lead to errors in navigation
Hierarchical Model Disadvantages

• Applications programming and use complexity
  – To access data, application programmers/end users need to know precisely how data are physically distributed and to use complex pointer system

• Implementation limitations
  – M:N relationships are difficult to implement
  – Two-parent condition can’t be implemented easily in a hierarchical environment

• Lack of standards
  – Lacked DDL to manipulate DB components & DML manipulate DB contents
  – Less portable
Hierarchical Model Technical Details

- Not covered any further in this class
- See Appendix H
The Network Model

- **Network database**
- Created to
  - Represent complex data relationships more effectively
  - Improve database performance
  - Impose a database standard
- **CODASYL**
  - Conference on Data Systems Languages
  - Originally formed by users & computer manufacturers to create a COBOL standard
- **American National Standards Institute (ANSI)**
  - Accepted the recommendation of CODASYL
  - Created ANSI COBOL
The Network Model

• Database Task Group (**DBTG**)
  – Group created to define standards for an environment to facilitate database creation
  – Report contained specification of crucial DB components
    • Schema
    • Subschema
    • DML (Data management/manipulation language)
Network Model Components

• The Network Schema
  – Conceptual organization of entire database as viewed by the database administrator
  – Includes database name, record type for each record, & the components that make up the records

• The Network Subschema
  – Defines database portion “seen” by the application programs that actually produce the desired information from data contained within the database
  – Allows all application programs to simply invoke the subschema required to access the appropriate files of the database

• Data Management Language (DML)
  – Define data characteristics and data structure in order to manipulate the data
DML Components

- **Schema Data Definition Language (DDL)**
  - Enables database administrator to define schema components
- **Subschema DDL**
  - Allows application programs to define database components that will be used
- **DML**
  - Manipulates database contents
Network Model Basic Structure

- Resembles hierarchical model
- Collection of records in 1:M relationships
- Set
  - Relationship
  - Composed of at least two record types
    - Owner
      - Equivalent to the hierarchical model’s parent
    - Member
      - Equivalent to the hierarchical model’s child
Network Data Model Example

FIGURE 2.3 A NETWORK DATA MODEL

Diagram showing the relationships between SALESPER, CUSTOMER, PRODUCT, INVOICE, and PAYMENT.
Network Data Model Advantages

- Conceptual simplicity
  - Simple DB conceptual view that promotes design simplicity
- Handles more relationship types
  - M:N relationships are easier to implement
- Data access flexibility
  - An application can access owner & all related records within a set
  - Can move directly from one owner to another. No need for pre-order traversal
- Promotes database integrity
  - A member can’t exist without an owner. User must first define the owner and then the member
- Data independence
  - Changes in data characteristics don’t require changes in data access portions of application programs
- Conformance to standards
  - DDL & DML facilitate DB administration & portability
Network Data Model Disadvantages

• System complexity
  – DBA’s, programmers/end users must be very familiar with the internal DB structure to access it

• Lack of structural independence
  – Difficult to make structural changes and some changes are impossible to make

• More cumbersome
  – As information, implementation & applications needs grew due to structural complexity,
Network Data Model Technical Details

- Not covered any further in this class
- See Appendix H
The Relational Model

• Developed by E.F. Codd (of IBM) in 1970’s
• Considered ingenious but impractical in 1970
• Conceptually simple
• Major breakthrough for both users & designers
• Initially, computers lacked the power to implement relational model
  – Size
  – Efficiency
  – Cost
• Today, microcomputers can run sophisticated relational database software
E.F. Codd

- E.F. Codd Died 2003 at age 79
  - [http://www.theregister.co.uk/content/4/30378.html](http://www.theregister.co.uk/content/4/30378.html)

- Codd's 12 Rules:
  - [http://www.databaseanswers.org/codds_rules.htm](http://www.databaseanswers.org/codds_rules.htm)
Codd’s 12 Rules Summary

1. Information rule:
   – Data are represented only in one way: as values within columns within rows.
   – Simple, consistent and versatile.

2. Guaranteed access rule:
   – Every value can be accessed by providing table name, column name and key.
   – All data are uniquely identified and accessible via this identity.

3. Systematic treatment of null values
   – Separate handling of missing and/or non applicable data.
   – Codd would further like several types of null to be handled.

4. Relational online catalog
   – Catalog (data dictionary) can be queried by authorized users as part of the database.
   – The catalog is part of the database.

5. Comprehensive data sublanguage
   – Today means: must support SQL.

6. View updating rule
   – All theoretically possible view updates should be possible.
   – Views are virtual tables. They appear to behave as conventional tables except that they are built dynamically when the query is run. This means that a view is always up to date.
Codd’s 12 Rules Summary

7 High-level insert, update and delete
   – User should be able to modify several tables by modifying the view to which they act as base tables.

8 Physical data independence
   – Physical layer of the architecture is mapped onto the logical layer.
   – Users and programs are not dependent on the physical structure of the database.

9 Logical data independence
   – Users and programs are independent of the logical structure of the database.
   – The logical structure of the data can evolve with minimal impact on the programs.

10 Integrity independence
    – Integrity constraints are to be stored in the catalog not the programs.
    – This simplifies the programs. It is not always possible to do this.

11 Distribution independence
    – Applications should still work in a distributed database (DDB).

12 Nonsubversion rule
    – If there is a record at a time interface (eg via 3GL), security and integrity of the database must not be violated.
    – There should be no backdoor to bypass the security imposed by the DBMS.
Relational DBMS Examples

- Informix
- Oracle
- Ingress
- DB2
- MySQL
- Sybase
- PostgreSQL
Relational Model Basic Structure

• Relational Database Management System (RDBMS)
• Performs same basic functions provided by hierarchical and network DBMS systems, plus other functions
• Most important advantage of the RDBMS is its ability to let the user/designer operate in a human logical environment
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
  – This property became the source of a real database revolution
• Also called a relation
• Each table is a matrix consisting of rows/column intersections
• Tables are related to each other sharing a common entity characteristic
• Stores a collection of related entities
  – Resembles a file
## Linking Relational Tables

### Figure 2.4 Linking Relational Tables

**Database name: Ch02_InsureCo**

**Table name: AGENT (first six attributes)**

<table>
<thead>
<tr>
<th>AGENT_CODE</th>
<th>AGENT_LNAME</th>
<th>AGENT_FNAME</th>
<th>AGENT_INITIAL</th>
<th>AGENT_AREACODE</th>
<th>AGENT_PHONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>501</td>
<td>Alby</td>
<td>Alex</td>
<td>B</td>
<td>713</td>
<td>226-1249</td>
</tr>
<tr>
<td>502</td>
<td>Hahn</td>
<td>Leah</td>
<td>F</td>
<td>615</td>
<td>882-1244</td>
</tr>
<tr>
<td>503</td>
<td>Okon</td>
<td>John</td>
<td>T</td>
<td>615</td>
<td>123-5589</td>
</tr>
</tbody>
</table>

**Link through AGENT_CODE**

**Table name: CUSTOMER**

<table>
<thead>
<tr>
<th>CUS_CODE</th>
<th>CUS_LNAME</th>
<th>CUS_FNAME</th>
<th>CUS_INITIAL</th>
<th>CUS_AREACODE</th>
<th>CUS_PHONE</th>
<th>CUS_RENEW_DATE</th>
<th>AGENT_CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10010</td>
<td>Ramas</td>
<td>Alfred</td>
<td>A</td>
<td>615</td>
<td>844-2573</td>
<td>05-Apr-2004</td>
<td>502</td>
</tr>
<tr>
<td>10011</td>
<td>Durne</td>
<td>Leona</td>
<td>K</td>
<td>713</td>
<td>894-1238</td>
<td>16-Jun-2004</td>
<td>501</td>
</tr>
<tr>
<td>10012</td>
<td>Smith</td>
<td>Kathy</td>
<td>W</td>
<td>615</td>
<td>894-2285</td>
<td>29-Jan-2005</td>
<td>502</td>
</tr>
<tr>
<td>10013</td>
<td>Olowsk</td>
<td>Paul</td>
<td>F</td>
<td>615</td>
<td>894-2180</td>
<td>14-Oct-2004</td>
<td>502</td>
</tr>
<tr>
<td>10014</td>
<td>Orlando</td>
<td>Myron</td>
<td></td>
<td>615</td>
<td>222-1672</td>
<td>28-Dec-2004</td>
<td>501</td>
</tr>
<tr>
<td>10015</td>
<td>O'Brien</td>
<td>Amy</td>
<td>B</td>
<td>713</td>
<td>442-3381</td>
<td>22-Sep-2004</td>
<td>503</td>
</tr>
<tr>
<td>10016</td>
<td>Brown</td>
<td>James</td>
<td>G</td>
<td>615</td>
<td>297-1228</td>
<td>25-Mar-2004</td>
<td>502</td>
</tr>
<tr>
<td>10017</td>
<td>Williams</td>
<td>George</td>
<td></td>
<td>615</td>
<td>290-2556</td>
<td>17-Jul-2004</td>
<td>503</td>
</tr>
<tr>
<td>10018</td>
<td>Farriss</td>
<td>Anne</td>
<td>G</td>
<td>713</td>
<td>382-7185</td>
<td>03-Dec-2004</td>
<td>501</td>
</tr>
<tr>
<td>10019</td>
<td>Smith</td>
<td>Olette</td>
<td>K</td>
<td>615</td>
<td>297-3809</td>
<td>14-Mar-2004</td>
<td>503</td>
</tr>
</tbody>
</table>
A Relational Schema

• Visual representation of the relational database’s entities, attributes, & relationships between entities

FIGURE 2.5 A RELATIONAL SCHEMA
Relational Model Advantages

- **Structural independence**
  - Changes in structure doesn’t affect DBMS’s data access abilities, since it doesn’t use navigational data access system

- **Improved conceptual simplicity**
  - Since the user doesn’t need to know the physical data storage characteristics, he can concentrate on logical view of the DB

- **Easier database design, implementation, management, and use**
  - Due to data & structural independence, it is easier to design the DB & manage its contents

- **Ad hoc query capability**
  - With structured query language SQL (4th generation language)
  - User specifies what must be done without having to specify how it must be done
  - Requires less programming than any other DB / file environment

- **Powerful database management system**
  - Makes it possible to hide system’s physical complexity from both DB designer & end user
Relational Model Disadvantages

• Substantial HW & system SW overhead
  – RDBMS needs powerful computers to perform the assigned tasks
  – Tends to be slower than other DB systems

• Easy to make poor design/implementation
  – Easy to use such that untrained users/designers tend to generate report / queries without giving much thought to proper design
  – Poor design tends to slow down the system & produce data anomalies

• May promote “islands of information” problems
  – Too many people find it easier to create their own DB subsets / applications
  – This might produce “islands of information” similar to file systems
Entity Relationship Model/Diagram (ERD)

- Abbreviated as **ERD**
- Widely accepted and adapted graphical tool for data modeling
- Introduced by [Peter Chen](http://en.wikipedia.org/wiki/Peter_Chen) in 1976
- Graphical representation of entities and their relationships in a database structure
- **Examples from Google searches**
ERD Basic Structure

• Entity relationship diagram (ERD)
  – Uses graphic representations to model database components
  – Entity is mapped to a relational table

• Components
  – Entities
  – Entity Instance (occurrence)
  – Entity set
  – Attributes
  – Relationship
  – Connectivity
Entities

• Represented by a rectangle, sometimes with rounded edges
• Described by a set of attributes
• When applying the ER diagram to relational model, entity is mapped to a relational table
• Name is a singular, noun, written in uppercase in the center of the triangle
• Examples:
  – PAINTER, EMPLOYEE,…
• Entity Instance (occurrence):
  – A row in the relational table
• Entity set:
  – A collection of similar entities
Attributes

• Describe a particular characteristic of an entity
• Represented as ovals/ellipse
• Examples:
  – EMPLOYEE’s Name, address, Social Security Number, …etc
Relationships

• Describe association between entities
• Represented as a diamond
• Name is active or passive verb
  – Usually written inside or above/under the diamond
Connectivity

• Relationship classification
  – 1:1 relationship
  – 1:M relationship
  – M:N relationship
• Written next to each box (or inside the diamond) or using Crow’s foot representation
**Relationships: Basic Chen ERD**

**Figure 2.6 Relationships: The Basic Chen ERD**

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

![Diagram showing a One-to-Many relationship between PAINTER and PAINTING]

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

![Diagram showing a Many-to-Many relationship between EMPLOYEE and SKILL]

A One-to-One (1:1) Relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE

![Diagram showing a One-to-One relationship between EMPLOYEE and STORE]
**Figure 2.7** Relationships: The Basic Crow’s Foot ERD

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.
ER Model Advantages

• Exceptional conceptual simplicity
  – Conceptual view of DB main entities & relationships are easy to view & understand

• Visual representation
  – Gives DB designer, programmer, & end user an easy view of data & their relationships

• Effective communication tool
  – Integrates different data views into a common framework

• Integrated with the relational database model
  – Helps make relational DB design a very structured process
ER Model Disadvantages

• Limited constraint representation
  – Many important data constraints can’t be modeled & must be handled at the application level
    • Examples: Student GPA lies between 0.0 & 4.0, or Pilot may not be scheduled more than 10 consecutive hours of duty time
• Limited relationship representation
  – Relationships between attributes within entities can’t be represented
    • Example: No way to display student’s classification & his/her completed credit hours
• No data manipulation language
  – No data manipulation commands in ER model
• Loss of information content
  – Designers tend to omit attributes When the model tend to be crowded thus decreasing informational contents
Object Oriented (OO) Model

• Semantic data model (SDM)
  – Developed by Hammer and McLeod in 1981
  – Modeled both data and their relationships in a single structure known as an **object**
  – Basis of object oriented data model (OODM)

• OODM
  – Basis for the object oriented database management system (OODBMS)

• [Wikipedia](https://en.wikipedia.org)
• [FOLDOC](http://www.folDOC.com)
• [Google](https://www.google.com)
Object Oriented Model

• Object is described by its factual content
  – Like relational model’s entity
• Includes information about relationships between facts within object and relationships with other objects
• Subsequent OODM development allowed an object to also contain operations
• Object becomes basic building block for autonomous structures
Why OODM’s Became So Popular?

• Growing costs put a premium on code reusability
  – Programs written in self-contained modules that perform specific functions
• Complex data types and system requirements became difficult to manage with a traditional RDBMS
  – Graphics, video, & sound
• Became possible to support increasingly sophisticated transaction & information requirements
• Ever-increasing computing power made it possible to support the large computing overhead required
OODM Basic Structure

• Object:
  – Abstraction of a real-world entity

• Attributes:
  – Describe the properties of an object

• Classes:
  – Objects that share similar characteristics are grouped in classes

• Class Hierarchy:
  – Classes are organized in a class hierarchy

• Inheritance:
  – The ability of an object within the class hierarchy to inherit the attributes and methods of classes above it
OO Model Example

- **Object:**
  - Instance of PERSON

- **Attributes:**
  - Name, DOB, ...

- **Class:**
  - All objects in PERSON or CUSTOMER

- **Methods:**
  - Finding or changing a PERSON’s name
  - Printing a PERSON’s address

- **Class hierarchy:**
  - CUSTOMER & EMPLOYEE share the parent class PERSON

- **Inheritance:**
  - CUSTOMER & EMPLOYEE inherit all attributes & methods from PERSON

```
PERSON
  NAME
  DOB
  ADDRESS
  SSN

Get Name
Print Address

EMPLOYEE

CUSTOMER
```
OO Model vs. ER Model

**FIGURE 2.8 A COMPARISON OF THE OO MODEL AND THE ER Model**

![Diagram comparing an OO model and an ER model](image)
OO Model Advantages

• Adds semantic content
  – Gives data greater meaning in addition to the attributes

• Visual presentation includes semantic content
  – Makes it easier to visualize much more complex relationships within & between objects

• Database integrity
  – Uses inheritance to protect DB integrity

• Both structural and data independence
OO Model Disadvantages

• Slow pace of OODM standards
  – Different vendors support different & incompatible data access methods

• Complex navigational data access
  – Data access style resembles those of hierarchical & network model

• Steep learning curve
  – Objects ability to contain too much semantics makes them difficult to design & implement

• High system overhead slows transactions
  – OODM’s implementation requires substantial HW & OS overhead and slows don the transactions

• Lack of market penetration
Other Models

• Extended Relational Data Model (ERDM)
  – Semantic data model developed in response to increasing complexity of applications
  – DBMS based on the ERDM often described as an object/relational database management system (O/RDBMS)
  – Primarily geared to business applications

• Date’s (Bio) objections to ERDM label
  – Given proper support for domains, relational data models are quite capable of handling complex data
  – O/RDM label is not accurate because the relational data model’s domain is not an object model structure
Data Models: A Summary

• Each new data model capitalized on the shortcomings of previous models
• Common characteristics:
  – Conceptual simplicity without compromising the semantic completeness of the database
  – Represent the real world as closely as possible
  – Representation of real-world transformations (behavior) must be in compliance with consistency and integrity characteristics of any data model
The Development of Data Models

**Figure 2.9 The Development of Data Models**

<table>
<thead>
<tr>
<th>Semantics in Data Model</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hierarchical</strong></td>
<td>• Difficult to represent M:N relationships (Hierarchical only)</td>
</tr>
<tr>
<td></td>
<td>• Physical level dependency</td>
</tr>
<tr>
<td></td>
<td>• No ad hoc queries</td>
</tr>
<tr>
<td><strong>Network</strong></td>
<td>• Access path predefined (navigational access)</td>
</tr>
<tr>
<td><strong>Relational</strong></td>
<td>• Provides ad hoc queries</td>
</tr>
<tr>
<td></td>
<td>• Set-oriented access</td>
</tr>
<tr>
<td></td>
<td>• Weak semantic content</td>
</tr>
<tr>
<td><strong>Entity Relationship</strong></td>
<td>• Easy to understand</td>
</tr>
<tr>
<td></td>
<td>• Incorporates more semantics</td>
</tr>
<tr>
<td><strong>Semantic</strong></td>
<td>• More semantics in data model</td>
</tr>
<tr>
<td><strong>Object-Oriented</strong></td>
<td>• Support for complex objects</td>
</tr>
<tr>
<td><strong>Extended Relational (Object/Relational)</strong></td>
<td>• Inheritance</td>
</tr>
<tr>
<td></td>
<td>• Behavior</td>
</tr>
</tbody>
</table>
Degrees of Data Abstraction

• Way of classifying data models
• Many processes begin at high level of abstraction and proceed to an ever-increasing level of detail
• Designing a usable database follows the same basic process
• American National Standards Institute/Standards Planning and Requirements Committee (ANSI/SPARC)
  – Classified data models according to their degree of abstraction (1970s):
    • Conceptual
    • External
    • Internal
    • Physical
• Definitions might be different in other books:
  – http://www.developer.com/db/article.php/719041
## Advantages/Disadvantages of DB Models

<table>
<thead>
<tr>
<th>DATA MODEL</th>
<th>DATA INDEPENDENCE</th>
<th>STRUCTURAL INDEPENDENCE</th>
<th>ADVANTAGES</th>
<th>DISADVANTAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>Yes</td>
<td>No</td>
<td>1. Promotes data sharing</td>
<td>1. Navigational systems yield complex design, implementation, application development, use, and management</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Parent/Child relationship promotes conceptual simplicity</td>
<td>2. Implementation limitations (No multiparent or M:N relationships)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Parent/Child relationship promotes data integrity</td>
<td>3. No data definition or data manipulation language in the DBMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. Efficient with 1:M relationships</td>
<td>4. Lack of standards</td>
</tr>
<tr>
<td>Network</td>
<td>Yes</td>
<td>No</td>
<td>1. Conceptual simplicity at least equal to that of the hierarchical model</td>
<td>1. System complexity limits efficiency—still a navigational system</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Handles more relationship types such as M:N and multiparent</td>
<td>2. Navigational systems yield complex design, implementation, application development, use, and management</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Owner/Member relationship promotes data integrity</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. Conformance to standards</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5. Includes data definition language (DDL) and data manipulation language (DML) in DBMS</td>
<td></td>
</tr>
<tr>
<td>Relational</td>
<td>Yes</td>
<td>Yes</td>
<td>1. Tabular view substantially improves conceptual simplicity, thereby promoting easier database design, implementation, management, and use</td>
<td>1. The RDBMS that makes the system easy to use and manage requires substantial hardware and software overhead</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Ad hoc query capability based on SQL</td>
<td>2. Conceptual simplicity gives relatively untrained people the tools to use a good system poorly</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Powerful RDBMS improves implementation and management simplicity</td>
<td>3. May promote “islands of information” problems as individuals and departments find it easy to develop their own applications</td>
</tr>
<tr>
<td>Entity Relationship</td>
<td>Yes</td>
<td>Yes</td>
<td>1. Visual modeling yields exceptional conceptual simplicity</td>
<td>1. Limited constraint representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Visual representation makes it an effective communication tool</td>
<td>2. Limited relationship representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Integrated with dominant relational model</td>
<td>3. No data manipulation language</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4. Loss of information content when attributes are removed from entities to avoid crowded displays (This limitation has been addressed in subsequent graphic versions)</td>
</tr>
<tr>
<td>Object Oriented</td>
<td>Yes</td>
<td>Yes</td>
<td>1. Adds semantic content</td>
<td>1. Slow development of standards caused vendors to supply their own enhancements, thus eliminating a widely accepted standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Visual representation includes semantic content</td>
<td>2. Complex navigational system</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Inheritance promotes data integrity</td>
<td>3. Steep learning curve</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4. High system overhead slows transactions</td>
</tr>
</tbody>
</table>

Note: All databases assume the use of a common data pool within the database. Therefore, all database models promote data sharing, thus at least potentially eliminating the “islands of information” problems as individuals and departments find it easy to develop their own applications.
Data Abstraction Levels

FIGURE 2.10  DATA ABSTRACTION LEVELS

Conceptual Model

Internal Model

Physical Model

External Model

External Model

Degree of Abstraction

High

Medium

Low

Characteristics

Hardware-independent
Software-independent

Hardware-independent
Software-dependent

Hardware-dependent
Software-dependent

DBMS

Relational

Hierarchical & Network

Relational

Hierarchical & Network
The Conceptual Model

• Represents global view of the database
• Enterprise-wide representation of data as viewed by high-level managers
• Basis for identification and description of main data objects, avoiding details
• Most widely used conceptual model is the entity relationship (ER) model
Tiny College Entities

FIGURE 2.11  TINY COLLEGE ENTITIES

Entities

STUDENT
PROFESSOR
COURSE
CLASS
ROOM

STUDENT attributes

Student number
Social Security number
First name
Last name
Middle initial
Sex
Date of birth
Home address
Home phone
College address
College phone
Major
Hours completed
Grade point average
A Conceptual Model for Tiny College

**Figure 2.12** A Conceptual Model for Tiny College

![Diagram of conceptual model](image)

- **COURSE**: 1
- **PROFESSOR**: 1
  - teaches: M
- **CLASS**: M
  - contains: M
  - is used for: M
- **STUDENT**: N
- **ROOM**: 1 generates

This diagram shows the relationships between courses, professors, classes, students, and rooms in a conceptual model for a tiny college.
Advantages of Conceptual Model

• 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 either the hardware or the DBMS software have no effect on the database design at the conceptual level
The Internal Model

- Representation of the database as “seen” by the DBMS
- Adapts the conceptual model to the DBMS
- Software dependent
  - Changes in DBMNS needs change of the internal model to find characteristics & requirements of DBMS
- Hardware independent
- Important for hierarchical / network DB designer
- Not as important for relational DBMS
  - Needed for mainframe environment (e.g. DB2 data storage group)
The Internal Model
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
- DBMS dependent
- Hardware independent
- Good design should:
  - Consider such relationships between views
  - Provide programmers with a set of restrictions that govern common entities
The External Model
Internal Model into External Models

**Figure 2.13** A Division of an Internal Model into External Models

```
PROFESSOR 1 teaches M CLASS M
  generates M
  is used for M
  is found in 1 ROOM

STUDENT 1
  enrolls in M

COURSE 1

ENROLL
```

Programmer Jim

```
STUDENT 1
  enrolls in M
  is found in 1 CLASS

ENROLL M
```

Programmer Anne

```
ROOM 1
  is used for M

PROFESSOR 1
  teaches M

CLASS
```
External Models Advantages

• Use of database subsets makes application program development much simpler
  – Facilitates designer’s task by making it easier to identify specific data required to support each business unit’s operations
  – Provides feedback about the conceptual model’s adequacy

• Creation of external models helps to ensure security constraints in the database design
The External Models for Tiny College

**Figure 2.14 The External Models for Tiny College**

### Student Registration
- **STUDENT**
- **ENROLL**
- **CLASS**

- A student may take up to six classes per registration.
- A class is limited to 35 students.

### Class Scheduling
- **ROOM**
- **CLASS**
- **PROFESSOR**

- 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 Physical Model

• Operates at lowest level of abstraction, describing the way data are saved on storage media such as disks or tapes
• Software and hardware dependent
• Requires that database designers have a detailed knowledge of the hardware and software used to implement database design
# Levels of Data Abstraction

<table>
<thead>
<tr>
<th>MODEL</th>
<th>DEGREE OF ABSTRACTION</th>
<th>DATA MODEL</th>
<th>FOCUS</th>
<th>INDEPENDENT OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptual</td>
<td>High</td>
<td>Entity</td>
<td>Global</td>
<td>Hardware and software</td>
</tr>
<tr>
<td>External</td>
<td></td>
<td>ER components</td>
<td>Subset</td>
<td>Hardware</td>
</tr>
<tr>
<td>Internal</td>
<td></td>
<td>Relational and others</td>
<td>Global</td>
<td>Hardware</td>
</tr>
<tr>
<td>Physical</td>
<td>Low</td>
<td>Physical storage methods</td>
<td>N/A</td>
<td>Neither hardware nor software</td>
</tr>
</tbody>
</table>
Summary

• A good DBMS will perform poorly with a poorly designed database
• A data model is a (relatively) simple abstraction of a complex real-world data-gathering environment
• Basic data modeling components are:
  – Entities
  – Attributes
  – Relationships
• Hierarchical model
  – Based on a tree structure composed of a root segment, parent segments, and child segments
  – Depicts a set of one-to-many (1:M) relationships between a parent and its children
  – Does not include ad hoc querying capability
Summary (continued)

• Network model:
  – Attempts to deal with many of the hierarchical model’s limitations

• Relational model:
  – Current database implementation standard
  – Much simpler than hierarchical or network design

• Object is basic modeling structure of object oriented model

• Data modeling requirements are a function of different data views (global vs. local) and level of data abstraction