Conceptual Data Modeling and the Entity-Relationship Model

© Department of Computer Science
Northern Illinois University
September 2014
Data Models

• A means of describing the structure of the data
• A set of operations that manipulate the data (only in data models that are implemented)
• Types of data models
  – Conceptual data model
  – Logical data models - relational, network, hierarchical, inverted list, or object-oriented
Conceptual Data Model

- Shows the structure of the data including the relationships
- Communication tool
- Independent of commercial DBMSs
- Easy to learn and use
- Provides semantics
- Graphical representation of the data
- Entity-Relationship Model is most common one used in world
Logical Data Models

• Relational -
  - data stored in tables with no repeating groups allowed
  - based upon a mathematical model
  - first presented by E. F. Codd in early 1970s
  - Commercial relational data models
    • DB2, Oracle, Ingress, and Microsoft Access
Logical Data Models

• Network -
  – data stored in records and associations called sets
  – very complex model
  – based upon the CODASYL model
  – created by a committee in 1970’s
  – commercial DBMSs
    • IDMS and TOTAL
Logical Data Models

• Hierarchical -
  – data stored in tree structure with parent / child relationships
  – first commercial DBMS created by IBM in late 1960s
  – commercial DBMSs
    • IMS and System 2000
  – XML
Logical Data Models

• Inverted List -
  – tabular representation of the data using indicies to access the tables
  – first touted themselves as relational in early 1970’s when no real relational available
  – NOT relational because repeating groups are allowed
  – commercial DBMSs
    • ADABAS (out of Germany)
Logical Data Models

• Object-Oriented –
  – Data stored as objects which contain
    • Identifier
    • Name
    • Lifetime
    • Structure
  – Commercial object-oriented DBMSs
    • O2 (now called Ardent) and ObjectStore
Entity-Relationship Model

- First introduced in 1976 by Peter P. Chen
- Simple
- Readable
- Understood easily by both database designer and unsophisticated user
Basic ER Concepts

• Entities
  - principal objects about which information is kept
  - denote a noun such as person, place, thing, or event
  - shown as a rectangle with the name (singular) inside
Basic ER Concepts

• Relationships
  – associations among one or more entities
  – cannot exist without associated entities
  – represented as a diamond with name inside or just next to it
Basic ER Concepts

• Attributes
  – characteristics of entities or relationships
  – fields in COBOL vernacular
  – sometimes shown using oval attached to entity

Person \( \rightarrow \) Name
Basic ER Concepts

• Attributes

Identifier (key) • Stud-id

Descriptor • Name

Multivalued descriptor • Majors

Composite attribute • Address

Street • City • State • Zip
Basic ER Concepts

• Degree of a Relationship
  – the number of entities associated with a relationship
    • binary
    • ternary
Basic ER Concepts

• Degree of a Relationship
  - the number of entities associated with a relationship
    • no limit (n-ary)

Diagram:
- Sales person
- Contact person
- Contact Description
- Date of Contact
- Contacted
- Customer
Basic ER Concepts

• Connectivity of a Relationship
  – constraint on the mapping of associated entities
  – written as (min, max)
  – minimum - zero or one (usually)
  – maximum - one or many (usually)
  – actual number is called CARDINALITY
Basic ER Concepts

- Connectivity of a Relationship

**One-to-one**

Person  \( \rightarrow \) lives-at \( \rightarrow \) Address

**One-to-many**

Person  \( \rightarrow \) Receives-mail-at \( \rightarrow \) Address
Basic ER Concepts

• Connectivity of a Relationship

**mandatory many-to-many**

Student (1,m) \(\text{enrolled-in} \) (1,n) Section

**optional many-to-many**

Student (0,m) \(\text{enrolled-in} \) (0,n) Section
Basic ER Concepts

- Attributes of a Relationship
  - must be on a many-to-many relationship (NOT on a 1-m or 1-1 relationship)
  - intersection data
  - needs to know ALL associated entities to access attribute
Basic ER Concepts

Need to know BOTH Student AND Course to get to grade.
Basic ER Concepts

- **Weak Entity**

  - **Person**
    - **SSN**
    - **Strong Entity**
    - **Address**
      - **Type**

  - **Address**
    - **Discriminator**

  - **Identifier of Address = SSN || Type**
Basic ER Concepts

- **Recursive Relationship**: many-to-many (network)

A Person has many relatives. AND A Person is related to many other Persons.
Basic ER Concepts

- Recursive Relationship: 1 - many (tree)

A Department reports to One and only one Department. AND A Department may have 0, 1, or more reporting to it.
Basic ER Concepts

• Recursive Relationship: many-to-many (network)

A Person has many relatives. AND A Person is related to many other Persons.
Advanced ER Constructs

• Supertype / Subtype (isa relationship)
  – generalization
  – specialization
  – overlapping subtypes
  – disjoint subtypes
**Advanced ER Constructs**

- **Inheritance**
  - the attributes describing the supertype entity are inherited by the entities of the subtypes

- The identifier of the subtypes is the same as the supertype.

- NOTE: the notation used here is different than Toby Teorey’s book.
Advanced ER Constructs

- Generalization
  - Supertype is the UNION of all the subtypes.
  - An instance of the supertype CANNOT exist without being related to at least one instance of a subtype.

A Person MUST be either a Faculty, Student, or Staff.
Advanced ER Constructs

• Specialization
  – The subtype entities specialize the supertype.
  – An instance of the supertype CAN exist without being related to any subtype.

A Person CAN be either a Faculty, Student, or Staff but DOES NOT have to be any of them.
Advanced ER Constructs

• Overlapping Subtype Entities
  – An instance of the supertype can be related to one or more of the subtypes.

A Person CAN be either a Faculty, Student, or Staff OR can be BOTH a Faculty and Student OR BOTH a Faculty and Staff OR BOTH a Student and Staff OR can be all three.
Advanced ER Constructs

- **Disjoint Subtype Entities**
  - the subtype entities are mutually exclusive

A Person CAN be ONLY ONE of either Faculty, Student, or Staff.
Advanced ER Constructs

Use combinations:
- G / O: generalization with overlapping subtypes
- G / D: generalization with disjoint subtypes
- S / O: specialization with overlapping subtypes
- S / D: specialization with disjoint subtypes

A Person MUST be ONE and ONLY ONE of either Faculty, Student, or Staff.
Advanced ER Constructs

- Aggregation (ispo relationship)
  - Is-part-of
  - Is-made-up-of

An Auto is-made-up-of one Hood, four Wheels, and two doors.
Relationship is represented by a table whose Primary Key is the concatenation of the IDs of the related entities.

<table>
<thead>
<tr>
<th>SSN</th>
<th>VIN #</th>
</tr>
</thead>
<tbody>
<tr>
<td>222</td>
<td>1111</td>
</tr>
<tr>
<td>333</td>
<td>5555</td>
</tr>
<tr>
<td>444</td>
<td>8888</td>
</tr>
</tbody>
</table>

Question?
What happens if a Person buys the SAME Car more than one time?
Interesting Construct

NO, because the insert would cause a duplicate instance. A duplicate instance is NOT allowed.
Interesting Construct

Now can we add to the table?

<table>
<thead>
<tr>
<th>SSN</th>
<th>VIN #</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>222</td>
<td>1111</td>
<td>2/4/98</td>
</tr>
<tr>
<td>333</td>
<td>5555</td>
<td>5/8/00</td>
</tr>
<tr>
<td>444</td>
<td>8888</td>
<td>11/3/77</td>
</tr>
<tr>
<td>222</td>
<td>1111</td>
<td>10/02/99</td>
</tr>
</tbody>
</table>

"SSN" connects to "Person" with (0,1) relationship, "buys" with (0,n) relationship, and "VIN #" with (1,n) relationship.
### Interesting Construct

The diagram illustrates a relationship between Person, Car, and Date attributes. Each Person can buy multiple Cars, and each Car can be bought by multiple Persons. The Date attribute is part of the key, as indicated by the YES annotation:

**Table:**

<table>
<thead>
<tr>
<th>SSN</th>
<th>VIN #</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>222</td>
<td>1111</td>
<td>2/4/98</td>
</tr>
<tr>
<td>333</td>
<td>5555</td>
<td>5/8/00</td>
</tr>
<tr>
<td>444</td>
<td>8888</td>
<td>11/3/77</td>
</tr>
<tr>
<td>222</td>
<td>1111</td>
<td>10/02/99</td>
</tr>
</tbody>
</table>

**Key Note:**

YES, because the Date attribute is part of the key.
Entity-Relationship Model

Person
- is-related-to (0,n)
- teaches (0,m)
- (1,1)

Faculty
- (0,1)

Staff
- (0,m)

Student
- (0,p)

Address
- has (1,m)

S,O

Semester
- completed (0,n)
- (0,p)
- (0,m)

Dept.
- reports-to (0,m)
- (1,1)

Course
- (1,1)

Grade
- within (0,m)

Section
- of (0,m)
- teaches (0,m)
- (0,n)

enrolled in (0,n)

credit-hours (0,m)

completed (0,n)
Entity-Relationship Model

- **Person**
  - SSN (Identifier)
  - Name
  - Birth-Date
  - Beginning Date

- **Address**
  - Type (discriminator)
  - Street
  - City
  - State
  - Zip

- **Faculty**
  - SSN (Identifier)
  - Contact hours
  - Tenure status

- **Staff**
  - SSN (Identifier)
  - Position

- **Student**
  - SSN (Identifier)
  - Overall GPA
  - Major
Entity-Relationship Model

• Dept.
  – Dept-Code (ID)
  – Dept-Name
  – Dept-Address
  – Dept-Chair

• Course
  – Crse-Code (ID)
  – Crse-Title
  – Crse-Max-Credit-Hours
  – Crse-Var-Hours-Code
  – Crse-Fee

• Section
  – Sect-Code (ID)
  – Sect-Credit-Hours
  – Sect-Meet-Time
  – Sect-Meet-Day

• Semester
  – Sem-Yr (ID)
  – Sem-Session (ID)
Entity-Relationship Model

RELATIONSHIPS with attributes

• Student enrolled-in Section
  – Credit-hours
    • In a variable credit section this attribute would be used to hold the credit hours for which a specific student is enrolled.

• Completed
  – Grade
    • A student is allowed to take a course more than once.