Chapter 7: Data Modeling Using the Entity-Relationship (ER) Model

Dr. Chris Irwin Davis

Email: cid021000@utdallas.edu
Phone: (972) 883-3574
Office: ECSS 4.705
Chapter 7 Outline

• Using High-Level Conceptual Data Models for Database Design
• A Sample Database Application
• Entity Types, Entity Sets, Attributes, and Keys
• Relationship Types, Relationship Sets, Roles, and Structural Constraints
• Weak Entity Types
• Refining the ER Design for the COMPANY Database
• ER Diagrams, Naming Conventions, and Design Issues
• Example of Other Notation: UML Class Diagrams
• Relationship Types of Degree Higher than Two
Data Modeling Using the Entity-Relationship (ER) Model

- **Entity-Relationship (ER) model**
  - Popular high-level conceptual data model

- **ER diagrams**
  - Diagrammatic notation associated with the ER model

- **Unified Modeling Language (UML)**
Using High-Level Conceptual Models

• Requirements collection and analysis
  ° Database designers interview prospective database users to understand and document data requirements. Result:
  ° Data requirements
  ° Functional requirements of the application
Using High-Level Conceptual Models

• Conceptual schema
  ° Conceptual design
  ° Description of data requirements
  ° Includes detailed descriptions of the entity types, relationships, and constraints
  ° Transformed from high-level data model into implementation data model
Logical and Physical Design

• **Logical design or data model mapping**
  ° Result is a database schema in implementation data model of DBMS

• **Physical design phase**
  ° Internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files specified
A Sample Database Application

- COMPANY
  - Employees, departments, and projects
  - Company is organized into departments
  - Department controls a number of projects
  - Employee: store each employee’s name, Social Security number, address, salary, sex (gender), and birth date
  - Keep track of the dependents of each employee
Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.
Entity Types, Entity Sets, Attributes, and Keys

• ER model describes data as:
  ° Entities
  ° Relationships
  ° Attributes
Entities and Attributes

• **Entity**
  ° Thing in real world with independent existence

• **Attributes**
  ° Particular properties that describe entity
  ° Types of attributes:
    • *Composite* versus *simple* (atomic) *attributes*
    • *Single-valued* versus *multivalued* attributes
    • *Stored* versus *derived* attributes
    • *NULL* values
    • *Complex* attributes
Entities and Attributes

Figure 7.3
Two entities, EMPLOYEE $e_1$, and COMPANY $c_1$, and their attributes.
Entity Types, Entity Sets, Keys, and Value Sets

- **Entity type**
  - Collection (or set) of entities that have the same attributes

![Diagram showing entity types and entity sets.](image-url)
Types of Attributes

• Several types of attributes occur in the ER model
  o **Simple** versus **composite**
  o **Single-valued** versus **multivalued**
  o **Stored** versus **derived**
Composite versus Simple (Atomic) Attributes

• **Composite attributes** can be divided into smaller subparts, which represent more basic attributes with independent meanings.

• Attributes that are not divisible are called simple or atomic attributes.

![Address Diagram]

- **Address**
  - **Street_address**
  - **Number**
  - **Street**
  - **Apartment_number**
  - **City**
  - **State**
  - **Zip**
Single-Valued versus Multivalued Attributes

• Most attributes have a single value for a particular entity; such attributes are called single-valued
  ○ For example, Age is a single-valued attribute of a person
• An attribute can have a set of values for the same entity
  ○ A multivalued attribute may have lower and upper bounds to constrain the number of values allowed for each individual entity
Stored versus Derived Attributes

- Two (or more) attribute values are related
  - e.g. AGE and BIRTH_DATE
- BIRTH_DATE may be a store attribute
- AGE can be derived from BIRTH_DATE
Entity Types, Entity Sets, Keys, and Value Sets

- **Key or uniqueness constraint**
  - Attributes whose values are distinct for each individual entity in entity set
  - **Key attribute**
    - Uniqueness property must hold for every entity set of the entity type

- **Value sets (or domain of values)**
  - Specifies set of values that may be assigned to that attribute for each individual entity
Initial Conceptual Design of the COMPANY Database
Relationship Types, Relationship Sets, Roles, and Structural Constraints

- **Relationship**
  - When an attribute of one entity type refers to another entity type
  - Represent references as relationships not attributes
Relationship Types, Sets, and Instances

- **Relationship type** $R$ among $n$ entity types $E_1, E_2, \ldots, E_n$
  - Defines a set of associations among entities from these entity types

- **Relationship instances** $r_i$
  - Each $r_i$ associates $n$ individual entities $(e_1, e_2, \ldots, e_n)$
  - Each entity $e_j$ in $r_i$ is a member of entity set $E_j$
Relationship Degree

- **Degree** of a relationship type
  - Number of participating entity types
  - **Binary, ternary**
- Relationships as attributes
  - Think of a binary relationship type in terms of attributes
"Supply" Relationship

Figure 7.10
Some relationship instances in the SUPPLY ternary relationship set.
Role Names and Recursive Relationships

• **Role names**
  ° Role name signifies role that a participating entity plays in each relationship instance

• **Recursive relationships**
  ° Same entity type participates more than once in a relationship type in different roles
  ° Must specify role name
Recursive Relationship SUPERVISION

Figure 7.11
A recursive relationship SUPERVISION between EMPLOYEE in the supervisor role (1) and EMPLOYEE in the subordinate role (2).
Recursive Relationship SUPERVISION
Constraints on Binary Relationship Types

- **Cardinality ratio** for a binary relationship
  - Specifies maximum number of relationship instances that entity can participate in

- **Participation constraint**
  - Specifies whether existence of entity depends on its being related to another entity
  - Types: *total* and *partial*
Attributes of Relationship Types

- Attributes of 1:1 or 1:N relationship types can be migrated to one entity type

- For a 1:N relationship type
  - Relationship attribute can be migrated only to entity type on N-side of relationship

- For M:N relationship types
  - Some attributes may be determined by combination of participating entities
  - Must be specified as relationship attributes
Weak Entity Types

• Do not have key attributes of their own
  ° Identified by being related to specific entities from another entity type. This does not mean they don’t have a key, but the “parent” entity’s key is part of it.

• **Identifying relationship**
  ° Relates a weak entity type to its owner

• Always has a total participation constraint
Weak Entity Types

- If the “parent” entity is deleted, all related weak entities are deleted too
Refining the ER Design for the COMPANY Database

• Change attributes that represent relationships into relationship types
• Determine cardinality ratio and participation constraint of each relationship type
Summary of ER Diagram Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity</td>
<td>Entity</td>
</tr>
<tr>
<td>Weak Entity</td>
<td>Weak Entity</td>
</tr>
<tr>
<td>Relationship</td>
<td>Relationship</td>
</tr>
<tr>
<td>Indentifying Relationship</td>
<td>Indentifying Relationship</td>
</tr>
<tr>
<td>Attribute</td>
<td>Attribute</td>
</tr>
<tr>
<td>Key Attribute</td>
<td>Key Attribute</td>
</tr>
<tr>
<td>Multivalued Attribute</td>
<td>Multivalued Attribute</td>
</tr>
<tr>
<td>Composite Attribute</td>
<td>Composite Attribute</td>
</tr>
<tr>
<td>Derived Attribute</td>
<td>Derived Attribute</td>
</tr>
<tr>
<td>Total Participation of E2 in R</td>
<td>Total Participation of E2 in R</td>
</tr>
<tr>
<td>Cardinality Ratio 1: N for E1:E2 in R</td>
<td>Cardinality Ratio 1: N for E1:E2 in R</td>
</tr>
<tr>
<td>Structural Constraint (min, max) on Participation of E in R</td>
<td>Structural Constraint (min, max) on Participation of E in R</td>
</tr>
</tbody>
</table>
Proper Naming of Schema Constructs

- Choose names that convey meanings attached to different constructs in schema
- Nouns give rise to entity type names
- Verbs indicate names of relationship types
- Choose binary relationship names to make ER diagram readable from left to right and from top to bottom
Design Choices for ER Conceptual Design

• Model concept first as an attribute
  ° Refined into a relationship if attribute is a reference to another entity type

• Attribute that exists in several entity types may be elevated to an independent entity type
  ° Can also be applied in the inverse
Alternative Notations for ER Diagrams

- Specify structural constraints on relationships
  - Replaces cardinality ratio (1:1, 1:N, M:N) and single/double line notation for participation constraints
  - Associate a pair of integer numbers (min, max) with each participation of an entity type $E$ in a relationship type $R$, where $0 \leq \text{min} \leq \text{max}$ and $\text{max} \geq 1$
Figure 7.15: ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.
Example of Other Notation: UML Class Diagrams

- UML methodology
  - Used extensively in software design
  - Many types of diagrams for various software design purposes
- UML class diagrams
  - Entity in ER corresponds to an object in UML
Figure 7.16
The COMPANY conceptual schema in UML class diagram notation.

EMPLOYEE
Name: Name.dom
  Fname
  Minit
  Lname
Ssn
Bdate: Date
Sex: {M,F}
Address
Salary
age
change_department
change_projects
...
Dependent_name

DEPENDENT
Sex: {M,F}
Birth_date: Date
Relationship
...

DEPARTMENT
Name
Number
add_employee
number_of_employees
change_manager
...
Start_date
MANAGES

WORKS_FOR
4.*
1..1
0..1

DEPENDENT
supervisee
0..1
supervisor

PROJECT
Name
Number
add_employee
add_project
change_manager
...
Hours
WORKS_ON

LOCATION
Name
1..1

CONTROLS
0..*
1..1

Multiplicity
Notation in OMT:

OMT Notation in UML:
Whole Part
Example of Other Notation:
UML Class Diagrams

- **Class** includes three sections:
  - Top section gives the class name
  - Middle section includes the attributes;
  - Last section includes operations that can be applied to individual objects
Example of Other Notation: UML Class Diagrams

- **Associations**: relationship types
- **Relationship instances**: links
- Binary association
  - Represented as a line connecting participating classes
  - May optionally have a name
- Link attribute
  - Placed in a box connected to the association’s line by a dashed line
Example of Other Notation:
UML Class Diagrams

- **Multiplicities**: min..max, asterisk (*) indicates no maximum limit on participation
- Types of relationships: **association** and **aggregation**
- Distinguish between **unidirectional** and **bidirectional** associations
- Model weak entities using **qualified association**
Relationship Types of Degree Higher than Two

• **Degree** of a relationship type
  ° Number of participating entity types

• **Binary**
  ° Relationship type of degree two

• **Ternary**
  ° Relationship type of degree three
Choosing between Binary and Ternary (or Higher-Degree) Relationships

- Some database design tools permit only binary relationships
  - Ternary relationship must be represented as a weak entity type
  - No partial key and three identifying relationships
- Represent ternary relationship as a regular entity type
  - By introducing an artificial or surrogate key
Figure 7.17
Ternary relationship types. (a) The SUPPLY relationship. (b) Three binary relationships not equivalent to SUPPLY. (c) SUPPLY represented as a weak entity type.
Constraints on Ternary (or Higher-Degree) Relationships

- Notations for specifying structural constraints on $n$-ary relationships
  - Should both be used if it is important to fully specify structural constraints