Chapter 9: Relational DB Design by ER/EER to Relational Mapping

- Relational Database Design Using ER-to-Relational Mapping
- Mapping EER Model Constructs to Relations
Relational Database Design by ER- and EER-to-Relational Mapping

- Design a relational database schema
  - Based on a conceptual schema design
- Seven-step algorithm to convert the basic ER model constructs into relations
- Additional steps for EER model
Relational Database Design Using ER-to-Relational Mapping
Figure 9.2
Result of mapping the COMPANY ER schema into a relational database schema.
ER-to-Relational Mapping Algorithm

- COMPANY database example
  - Assume that the mapping will create tables with simple single-valued attributes
- Step 1: Mapping of Regular Entity Types
  - For each regular entity type, create a relation $R$ that includes all the simple attributes of $E$
  - Called entity relations
    - Each tuple represents an entity instance
ER-to-Relational Mapping Algorithm (cont’d.)

- **Step 2: Mapping of Weak Entity Types**
  - For each weak entity type, create a relation $R$ and include all simple attributes of the entity type as attributes of $R$
  - Include primary key attribute of owner as foreign key attributes of $R$
ER-to-Relational Mapping Algorithm (cont’d.)

Figure 9.3
Illustration of some mapping steps.

(a) EMPLOYEE
   - Fname
   - Minit
   - Lname
   - Ssn
   - Bdate
   - Address
   - Sex
   - Salary

   DEPARTMENT
   - Dname
   - Dnumber

   PROJECT
   - Pname
   - Pnumber
   - Plocation

(b) DEPENDENT
   - Essn
   - Dependent_name
   - Sex
   - Bdate
   - Relationship

(c) WORKS_ON
   - Essn
   - Pno
   - Hours

(d) DEPT_LOCATIONS
   - Dnumber
   - Dlocation
Step 3: Mapping of Binary 1:1 Relationship Types

- For each binary 1:1 relationship type
  - Identify relations that correspond to entity types participating in $R$

Possible approaches:
- Foreign key approach
- Merged relationship approach
- Crossreference or relationship relation approach
ER-to-Relational Mapping Algorithm (cont’d.)

- **Step 4: Mapping of Binary 1:N Relationship Types**
  - For each regular binary 1:N relationship type
    - Identify relation that represents participating entity type at N-side of relationship type
    - Include primary key of other entity type as foreign key in S
    - Include simple attributes of 1:N relationship type as attributes of S
ER-to-Relational Mapping Algorithm (cont’d.)

- Alternative approach
  - Use the *relationship relation* (cross-reference) option as in the third option for binary 1:1 relationships
Step 5: Mapping of Binary $M:N$ Relationship Types

- For each binary $M:N$ relationship type
  - Create a new relation $S$
  - Include primary key of participating entity types as foreign key attributes in $S$
  - Include any simple attributes of $M:N$ relationship type
ER-to-Relational Mapping Algorithm (cont’d.)

- **Step 6: Mapping of Multivalued Attributes**
  - For each multivalued attribute
    - Create a new relation
    - Primary key of $R$ is the combination of $A$ and $K$
    - If the multivalued attribute is composite, include its simple components
ER-to-Relational Mapping Algorithm (cont’d.)

- Step 7: Mapping of $N$-ary Relationship Types
  - For each $n$-ary relationship type $R$
    - Create a new relation $S$ to represent $R$
    - Include primary keys of participating entity types as foreign keys
    - Include any simple attributes as attributes
Discussion and Summary of Mapping for ER Model Constructs

<table>
<thead>
<tr>
<th>ER MODEL</th>
<th>RELATIONAL MODEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity type</td>
<td><em>Entity</em> relation</td>
</tr>
<tr>
<td>1:1 or 1:N relationship type</td>
<td><em>Foreign key</em> (or <em>relationship</em> relation)</td>
</tr>
<tr>
<td>M:N relationship type</td>
<td><em>Relationship</em> relation and <em>two</em> foreign keys</td>
</tr>
<tr>
<td>n-ary relationship type</td>
<td><em>Relationship</em> relation and <em>n</em> foreign keys</td>
</tr>
<tr>
<td>Simple attribute</td>
<td><em>Attribute</em></td>
</tr>
<tr>
<td>Composite attribute</td>
<td><em>Set of simple component attributes</em></td>
</tr>
<tr>
<td>Multivalued attribute</td>
<td><em>Relation and foreign key</em></td>
</tr>
<tr>
<td>Value set</td>
<td><em>Domain</em></td>
</tr>
<tr>
<td>Key attribute</td>
<td><em>Primary (or secondary) key</em></td>
</tr>
</tbody>
</table>
Discussion and Summary of Mapping for ER Model Constructs (cont’d.)

- In a relational schema relationship, types are not represented explicitly
  - Represented by having two attributes $A$ and $B$: one a primary key and the other a foreign key
Mapping EER Model Constructs to Relations

- Extending ER-to-relational mapping algorithm
Mapping of Specialization or Generalization

- Step 8: Options for Mapping Specialization or Generalization (see pages 294-295)
  - Option 8A: Multiple relations—superclass and subclasses
    - For any specialization (total or partial, disjoint or overlapping)
  - Option 8B: Multiple relations—subclass relations only
    - Subclasses are total
    - Specialization has disjointedness constraint
Mapping of Specialization or Generalization (cont’d.)

- **Option 8C: Single relation with one type attribute**
  - Type or discriminating attribute indicates subclass of tuple
  - Subclasses are disjoint
    - Potential for generating many NULL values if many specific attributes exist in the subclasses

- **Option 8D: Single relation with multiple type attributes**
  - Subclasses are overlapping
  - Will also work for a disjoint specialization
Mapping of Shared Subclasses (Multiple Inheritance)

- Apply any of the options discussed in step 8 to a shared subclass

Figure 9.6
Mapping the EER specialization lattice in Figure 8.8 using multiple options.
Mapping of Categories (Union Types)

- Step 9: Mapping of Union Types (Categories)
  - Defining superclasses have different keys
  - Specify a new key attribute
    - Surrogate key
Figure 9.7
Mapping the EER categories (union types) in Figure 8.8 to relations.
Summary

- Map conceptual schema design in the ER model to a relational database schema
  - Algorithm for ER-to-relational mapping
  - Illustrated by examples from the COMPANY database
- Include additional steps in the algorithm for mapping constructs from EER model into relational model
FIGURE 8.7
A specialization lattice with multiple inheritance for a UNIVERSITY database.
FIGURE 9.6
Mapping the EER specialization lattice in Figure 8.7 using multiple options.

<table>
<thead>
<tr>
<th>PERSON</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>Name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>Salary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ALUMNUS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>ALUMNUS_DEGREES</td>
</tr>
<tr>
<td></td>
<td>SSN</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STUDENT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>MajorDept</td>
</tr>
</tbody>
</table>
FIGURE 8.8
Two categories (union types): OWNER and REGISTERED_VEHICLE.
FIGURE 9.7
Mapping the EER categories (union types) in Figure 8.8 to relations.

<table>
<thead>
<tr>
<th>PERSON</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>DriverLicenseNo</td>
<td>Name</td>
<td>Address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BANK</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BName</td>
<td>BAddress</td>
<td>OwnerId</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMPANY</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CName</td>
<td>CAddress</td>
<td>OwnerId</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OWNER</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REGISTERED_VEHICLE</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>LicensePlateNumber</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CAR</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>CStyle</td>
<td>CMake</td>
<td>CModel</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TRUCK</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>TMake</td>
<td>TModel</td>
<td>Tonnage</td>
<td>TYear</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OWNS</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>VehicleId</td>
<td>PurchaseDate</td>
<td>LienOrRegular</td>
<td></td>
</tr>
</tbody>
</table>