RELATIONAL DATABASE DESIGN VIA ER MODELLING

CHAPTER 9 (6/E)
CHAPTER 7 (5/E)
LECTURE OUTLINE

- Relational Database Design Using ER-to-Relational Mapping
  - Algorithm to convert the basic ER model constructs into relations
- Mapping EER Model Constructs to Relations
  - Additional steps for EER model
Figure 9.1
The ER conceptual schema diagram for the COMPANY database.
Figure 9.2
Result of mapping the COMPANY ER schema into a relational database schema.
STEP 1: MAP REGULAR ENTITY TYPES

- For each regular entity type, create a relation schema \( R \) that includes all the single-valued attributes of \( E \)
  - “Flatten” composite attributes
  - Example renames some attributes (e.g., Dname), but not needed
  - Pick one of the keys as “primary key” and declare the rest to be unique
- Called entity relations
- Each tuple represents an entity instance

(a) 

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fname</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>DEPARTMENT</td>
</tr>
<tr>
<td>Dname</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>PROJECT</td>
</tr>
<tr>
<td>Pname</td>
</tr>
</tbody>
</table>
STEP 2: MAP WEAK ENTITY TYPES

- For each weak entity type, create a relation schema $R$ and include all single-valued attributes of the weak entity type and of the identifying relationship as attributes of $R$
  - Include primary key attribute of identifying entity as foreign key attribute of $R$
  - Primary key of $R$ is primary key of identifying entity together with partial key from $R$
- Omit the identifying relationship when subsequently translating (other) relationship types to relation schemas

(b) DEPENDENT

<table>
<thead>
<tr>
<th>Essn</th>
<th>Dependent_name</th>
<th>Sex</th>
<th>Bdate</th>
<th>Relationship</th>
</tr>
</thead>
</table>

STEP 3: MAP BINARY 1:1 RELATIONSHIP TYPES

- For each binary 1:1 relationship type $R$, identify relation schemas that correspond to entity types participating in $R$
  - Apply one of three possible approaches:
    - **Foreign key approach**
      - Add primary key of one participating relation as foreign key attribute of the other, which will also represent $R$
        - If only one side is total, choose it to represent $R$ *(why?)*
      - Declare foreign key attribute as unique
    - **Merged relationship approach**
      - Combine the two relation schemas into one, which will also represent $R$
      - Make one of the primary keys “unique” instead
    - **Cross-reference or relationship relation approach**
      - Create new relation schema for $R$ with two foreign key attributes being copies of both primary keys
      - Declare one of the attributes as primary key and the other one as unique
  - Add single-valued attributes of relationship type as attributes of $R$
STEP 4: MAP BINARY 1:N RELATIONSHIP TYPES

- **Foreign key approach**
  - Identify relation schema $S$ that represents participating entity type at $N$-side of 1:N relationship type
  - Include primary key of other entity type (1-side) as foreign key in $S$

- **Relationship relation approach**
  - Create new relation schema for $S$ with two foreign key attributes being copies of both primary keys
  - Declare the foreign key attribute for the relation schema corresponding to the participating entity type on the $N$-side as primary key
  - Include single-valued attributes of relationship type as attributes of $S$
STEP 5: MAP BINARY $M:N$ AND HIGHER ORDER RELATIONSHIP TYPES

- For each binary $M:N$ relationship type or ternary or higher order relationship type, create a new relation $S$
  - Include primary key of participating entity types as foreign key attributes in $S$
  - Make all these attributes primary key of $S$
  - Include any simple attributes of relationship type in $S$
STEP 6: MAP MULTIVALUED ATTRIBUTES

- For each multivalued attribute
  - Create new relation $R$ with attribute to hold multivalued attribute values
    - If multivalued attribute is composite, include its simple components
  - Add attribute(s) for primary key of relation schema for entity or relationship type to be foreign key for $R$
  - Primary key of $R$ is the combination of all its attributes

(d) DEPT_LOCATIONS

| Dnumber | Dlocation |
OPTIONS FOR MAPPING SPECIALIZATION OR GENERALIZATION

- For *any* specialization (total or partial, disjoint or overlapping)
  - Separate relation per superclass and subclasses
  - Single relation with at least one attribute per subclass
    - Introduce a Boolean attribute if none specific for subclass
SPECIALIZATION OPTIONS (CONT’D)

- For total specializations (and generalizations) only
  - Separate relation per subclass relations only
    - Overlapping subclasses will result in multiple tuples per entity

- For disjoint specializations only
  - Single relation with one type attribute
    - **Type** or **discriminating attribute** indicates subclass of tuple
    - Might require many NULL values if several specific attributes exist in subclasses
MAPPING UNION TYPES

- Create relation schema to represent union type (generalization)
- Specify a new key attribute
  - **Surrogate key**
- Example: Owner and Registered Vehicle
LEARNING SUMMARY

- Algorithm for ER-to-relational mapping

<table>
<thead>
<tr>
<th>Table 9.1 Correspondence between ER and Relational Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ER MODEL</strong></td>
</tr>
<tr>
<td>Entity type</td>
</tr>
<tr>
<td>1:1 or 1:N relationship type</td>
</tr>
<tr>
<td>M:N relationship type</td>
</tr>
<tr>
<td><em>n</em>-ary relationship type</td>
</tr>
<tr>
<td>Simple attribute</td>
</tr>
<tr>
<td>Composite attribute</td>
</tr>
<tr>
<td>Multivalued attribute</td>
</tr>
<tr>
<td>Value set</td>
</tr>
<tr>
<td>Key attribute</td>
</tr>
</tbody>
</table>

- Extensions for mapping constructs from EER model into relational model
EXERCISE

Translate the following ER Diagram into a relational database schema.
What ER Diagram might produce the following relational database schema?