Chapter (9)
ER and EER-to-Relational Mapping, and other Relational Languages

Objectives
- How a relational database schema can be created from a conceptual schema developed using the ER models.

We will look at the Relational Database Design Using ER-to-Relational Mapping. We outline an algorithm that can map an ER schema into the corresponding relational database schema.

ER-Relational Mapping Algorithm

We will use the COMPANY database to walk through the algorithm.

Step 1:
For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
- Include only the simple component attributes of a composite attribute.
- Choose one of the key attributes of E as primary key for R. If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

Before we got to the next step, let’s look at one definition, Foreign Key. The conditions for a foreign key specify the constraints between two relations and is used to maintain the consistency among tuples (rows) of the two relation schemas R₁ and R₂. This constrain is called referential integrity constraints. Now we have to answer this question.

What are the conditions for a foreign key?
Foreign Key

A set of attributes (FK) in relation schema R₁ is a foreign key of R₁ that reference relation R₂ if it satisfies the following two rules:

1. The attributes in FK have the same domain(s) as the primary key attributes PK of R₂; the attributes FK are said to reference or refer to the relation R₂.
2. A value of FK in a tuple t₁ of the current state r₁(R₁) either occurs as a value of PK for some tuple t₂ in the current state r₂(R₂) or is null. In the former case, we have t₁[FK] = t₂[PK], and we say that the tuple t₁ references or refers to the tuple t₂. R₁ is called the referencing relation and R₂ is the referenced relation.

Example: The department that an employee work is not an attribute in the EMPLOYEE entity. The department number will come from the DEPARTMENT entity. Thus, on every tuples of a relation the Department Number (DNO) is a foreign key to the EMPLOYEE relation. Thus, all DNO (on all tuples) on EMPLOYEE relation must much a value of the primary key of DEPARTMENT. This value can be null only if the Employee does not belong to a department.

For more info, see 7.2.4.

Step 2:

•For each weak entity type W in the ER schema with owner entity type E, create a relation R, and include all simple attributes (or simple components of composite attributes) of W as attributes of R.

•In addition, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of the identifying relationship type of W. The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.

•Example: We created the relation DEPENDENT in this step to correspond to the weak entity type DEPENDENT. We include the primary key SSN of the EMPLOYEE relation which corresponds to the owner entity type – as a foreign key attribute of DEPENDENT.
STEP 3:
• For each binary 1:1 relationship type $R$ in the ER schema, identify the relations $S$ and $T$ that correspond to the entity types participating in $R$.
• Choose one of the relations—$S$, say—and include as foreign key in $S$ the primary key of $T$. It is better to choose an entity type with total participation in $R$ in the role of $S$. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type $R$ as attributes of $S$.

Example: we map the 1:1 relationship type MANGES by choosing the participating entity type DEPARTMENT to serve in the role of $S$, because its participation in the MANGES relationship type is total.

STEP 4:
For each regular binary 1:N relationship type $R$, identify the relation $S$ that represents the participating entity type at the $N$-side of the relationship type. Include as foreign key in $S$ the primary key of the relation $T$ that represents the other entity type participating in $R$; this is because each entity instance on the $N$-side is related to at most one entity instance on the 1-side of the relationship type. Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attributes of $S$.

Example: we map the 1:N relationship types WORKS_FOR, CONTROLS, and SUPERVISION. WORKS_FOR has DNUMBER as primary key (foreign), SUPERVISION has a SSN as primary key (foreign), and the CONTROLS relationship has DNUM of PROJECT as its key (also foreign).
STEP 5:
For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S. Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S. Notice that we cannot represent an M:N relationship type by a single foreign key attribute in one of the participating relations—as we did for 1:1 or 1:N relationship types—because of the M:N cardinality ratio.

Example: we map the M:N relationship type WORKS_ON by creating the relation WORKS_ON as given on Figure 7.5. The primary keys of the PROJECT and EMPLOYEE relations are included as foreign keys in WORKS_ON with the new names PNO and ESSN respectively. The HOURS is a new attribute that represents the number of hours.
**STEP 6:**
For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in R—of the relation that represents the entity type or relationship type that has A as an attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

Example: A relation DEPT_LOCATIONS which includes the DLOCATION that represents the multivalued attribute Locations of DEPARTMENT. It also includes DNUMBER as a foreign key that represents the primary key of the DEPARTMENT relation.

Note: The primary key of the DEPT_LOCATION is .............

**STEP 7:**
For each n-ary relationship type R, where n > 2, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S. The primary key of S is usually a combination of all the foreign keys that reference the relations representing the participating entity types. However, if the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of S should not include the foreign key attribute that references the relation E’ corresponding to E (see section 4.7). This concludes the mapping procedure.

Example: On Figure 9.1, one can see that the relation SUPPLY with primary key {SNAME, PARTNO, PROJECTNAME}. 
Summary of Mapping for Model Constructs and Constraints

Table 9.1  Correspondence between ER and Relational Models

<table>
<thead>
<tr>
<th>ER Model</th>
<th>Relational Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity type</td>
<td>“Entity” relation</td>
</tr>
<tr>
<td>1:1 or 1:N relationship type</td>
<td>Foreign key (or “relationship” relation)</td>
</tr>
<tr>
<td>M:N relationship type</td>
<td>“Relationship” relation and two foreign keys</td>
</tr>
<tr>
<td>n-ary relationship type</td>
<td>“Relationship” relation and n foreign keys</td>
</tr>
<tr>
<td>Simple attribute</td>
<td>Attribute</td>
</tr>
<tr>
<td>Composite attribute</td>
<td>Set of simple component attributes</td>
</tr>
<tr>
<td>Multivalued attribute</td>
<td>Relation and foreign key</td>
</tr>
<tr>
<td>Value set</td>
<td>Domain</td>
</tr>
<tr>
<td>Key attribute</td>
<td>Primary (or secondary) key</td>
</tr>
</tbody>
</table>