

MAPPING
RELATIONAL MODEL – ER
LECTURE 5
DATABASE TECHNOLOGY
TDDB48

DESIGNING A RELATIONAL DATABASE SCHEMA

This is the logical database design step.

Many tools (CASE, etc) use ER diagrams or variations to develop the schema graphically, and then automatically convert it into a relational database schema in the DDL of a specific relation DBMS.

RELATIONAL CALCULUS

Two variations of the tuple calculus

- **The tuple relational calculus**
- **The domain relational calculus**

Much of the SQL query language is based on the tuple relational calculus.

The Relational Calculus

The relational calculus is a formal language, based on the branch of mathematical logic called predicate calculus.

- In tuple relational calculus, variables range over tuples
- In domain relational calculus, variables range over the domains (values) of attributes

QBE Language

We will also consider the QBE language – Query by Example.

This is a graphical user-friendly relational language based on domain relational calculus

ER TO RELATIONAL MAPPING

RELATIONAL DATABASE DESIGN USING ER TO RELATIONAL MAPPING

ER to Relational Mapping Algorithm

There are seven general steps in this procedure.

STEP ONE

For each strong (regular) entity type E in the ER schema:

- a. Create a relation R that includes all the simple attributes of E.
- b. 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.

EXAMPLE for STEP ONE

The relations EMPLOYEE, DEPARTMENT, and PROJECT in Fig 7.5 correspond to the strong entity types EMPLOYEE, DEPARTMENT, and PROJECT from figure 3.2.

Note that the foreign key and relationship attributes, if any, are not included yet, they will be added in subsequent steps.

STEP TWO

For each weak entity type W in the ERD with owner entity type E :

- a. Create a relation R , include all the atomic attributes of W as attributes of R .
- b. Include as foreign key attributes of R the primary key attributes that correspond to the owner entity types.

This is 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 for STEP TWO

- The relation DEPENDENT in this step corresponds 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.
- We rename it ESSN (though not necessary).
- The primary key of the DEPENDENT relation is the combination {ESSN, DEPENDENT_NAME} because DEPENDENT_NAME is the partial key of DEPENDENT.

It is common to choose the **CASCADE** option for the referential triggered action on the foreign key in the relation corresponding to the weak entity type, since a weak entity has an existence dependency on its owner entity.

This can be used for both **ON UPDATE** and **ON DELETE**.

STEP THREE:

For each binary 1:1 relationship type R in the ER schema:

- a. Identify the relations S and T that correspond to the entity types participating in R.
- b. 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 atomic attributes of the 1:1 relationship type R as attributes of S.

EXAMPLE for STEP THREE

Map the 1:1 relationship type MANAGES from fig 3.2 by choosing the participating entity type DEPARTMENT to serve in the role of S.

include the primary key of the EMPLOYEE relation as foreign key in the DEPARTMENT relation and rename it MGRSSN. We also include the simple attribute Starting-Date of the MANAGES relationship type in the DEPARTMENT relation and rename it MGSTARTDATE.

Notice that an alternative mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This is appropriate when both participations are total.

STEP FOUR

For each binary 1:N relationship type R:

- a. Identify the relation S that represents the participation entity type at the N-side of the relationship type.
- b. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.

Each entity instance on the N-side is related to at most one entity instance on the 1-side of the relationship type.

Any atomic attributes of the 1:N relationship type are attributes of S.

EXAMPLE for STEP FOUR

- We now map the 1:N relationship types **WORKS_FOR**, **CONTROLS**, and **SUPERVISION** from fig. 3.2 For **WORKS_FOR** we include the primary key **DNUMBER** of the **DEPARTMENT** relation as foreign key in the **EMPLOYEE** relation and call it **DNO**.
- For **SUPERVISION** we include the primary key of the **EMPLOYEE** relation as foreign key in the **EMPLOYEE** relation itself—because relationship is recursive—and call it **SUPERSSN**.
- The **CONTROLS** relationship is mapped to the foreign key attribute **DNUM** of **PROJECT**, which references the primary key **DNUMBER** of the **DEPARTMENT** relation.

STEP FIVE

For each binary M:N relationship type R:

- a. Create a new relation S to represent the participation entity types.
- b. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. This combination forms the primary key of S.
- c. Include any atomic attributes of the M:N relationship type as attributes of S.

We cannot represent an M:N relationship type by a single foreign key attribute in one of the participating relations, because of its cardinality ratio.

EXAMPLE for STEP FIVE

Map the M:N relationship type WORKS_ON from fig 3.2 by creating the relation WORKS_ON in fig 7.5.

Include the primary keys of the PROJECT and EMPLOYEE relations as foreign keys in WORKS_ON and rename them PNO and ESSN.

Include an attribute HOURS in WORKS_ON and to represent the Hours attribute of the relationship type.

The primary key of the WORKS_ON relation is the combination of the foreign key attributes {ESSN, PNO}.

The CASCADE option for the referential triggered action (fig 8.1) should be specified on the foreign keys in the relation corresponding to the relationship R, since each relationship instance has an existence dependency on each of the entities it relates. This can be used for both ON UPDATE and ON DELETE.

STEP SIX

For each multivalued attribute A:

- a. Create a new relation R.
- b. 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 of relationship type that has A as an attribute.
- c. The primary key of R is the combination of A and K.

We include all atomic components.

EXAMPLE for STEP SIX

- We create a relation DEPT_LOCATIONS.
- The attribute DLOCATION represents the multivalued attribute locations of DEPARTMENT, while DNUMBER—as foreign key—represents the primary key of the DEPARTMENT relation.
- The primary key of DEPT_LOCATION is the combination of {DNUMBER, DLOCATION}.
- A separate tuple will exist in DEPT_LOCATIONS for each location that a department has.

Again, the CASCADE option for the **referential triggered action** should be specified on the foreign key in the relation corresponding to the multivalued attribute for both ON UPDATE and ON DELETE.

STEP SEVEN

For each n-ary relationship type R, where $n > 2$:

- a. Create a new relation S to represent R.
- b. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
- c. Include any atomic attributes of the n-ary relationship type 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.

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 . This concludes the mapping procedure.

EXAMPLE for STEP SEVEN

Consider the relationship type SUPPLY of fig 4.13a. This can be mapped to the relation SUPPLY shown in fig 9.1, whose primary key is the combination of foreign keys {SNAME, PARTNO, PROJNAME}

The main points to note in a relational schema, in contrast to an ER schema, are

- a. Relationship types are not represented explicitly.
- b. They are represented by having two attributes A and B, one a primary key and the other a foreign key—over the same domain—included in two relations S and T.
- c. Two tuples in S and T are related when they have the same value for A and B.

- d. Using the EQUIJOIN (NATURAL JOIN) operation over S.A and T.B, we can combine all pairs of related tuples from S and T and materialize the relationship.
- e. When a binary 1:1 or 1:N relationship type is involved, a single join operation is usually needed.
- f. For a binary M:N relationship type, two join operations are needed, whereas for n-ary relationship types, n joins are needed.

SUMMARY of MAPPING for MODEL CONSTRUCTS and CONSTRAINTS