Part 2: Semantic Data Modeling

Phases of Database Design (Simplified)

Miniworld

DATABASE REQUIREMENTS COLLECTION AND ANALYSIS

Database Requirements

CONCEPTUAL DESIGN

Conceptual Schemas (In a high-level data model)

LOGICAL DESIGN/DATA MODEL MAPPING

Logical(Conceptual Schemas) (In the data model of a specific DBMS)

PHYSICAL DATABASE DESIGN

Internal Schema (For the same DBMS)
Data Model and Data Modeling

Data model:

- A data model is the main tool for providing the data abstraction.
- Data model = schema + operations + constraints

Data modeling:

- It is a procedure to represent data requirements in an abstract, correct, consistent, and stable format.
- It is a process to translate database requirements to the conceptual database schema in a high level data model.
- The conceptual database schema in a high level data model is dbms-independent.

Database Schema and Database Design

Database design:

- It is the process of data modeling and translation of database schema from a high level data model to a dbms-specific implementation data model.

Database schema:

- It is the description of a database, and it consists of a set of schemas or schema constructs.
- Each schema or schema construct describes a data entity or an interaction between data entities structurally.
Database Modeling and Design

Phases of database design:

Step 1: requirement collection and analysis.
Input: prospective database users’ requirements.
Output: a set of users’ requirements written in a concise way.

Step 2: conceptual database design
Input: output from step 1.
Output: creation of conceptual schema for the database via high-level conceptual data model.

Step 3: data model mapping
Input: output from step 2.
Output: conceptual database schema in the implementation data model.

Step 4: physical database design

Database Design Outline

Real World (Mini World)

Semantic Model (ER Model)

- Relational Schema
- Network Schema
- Hierarchical

Junping Sun
Database Systems
2-5

Page 3
Data Modeling

Entity-relationship (ER) model:

- It is designed to be closer to user’s perception of data and not meant to describe the way in which data will be stored in the computer.
- It is used mainly during the process of database design.
- ER approach for conceptual database modeling was first described in 1976 by Peter Chen.
- Schema diagram was formalized in 1960s by Bachman.
- It has been used by many industries and company to design initial database.
- It has been chosen as industry standard for database modeling.
- There are some CASE tools available to support ER modeling.

ER model representation of database model:

- Entity-relationship diagram
- Conceptual schema definition language

ER Model Concepts

Entity:
- It is a basic object that the ER model represents within independent existence.
- An entity could be an object with a physical existence
  - Person, car, house, or employee
- An entity could be an object with a conceptual existence
  - A company, a job, or a university

Attributes
- The particular properties to describe the entity.
- An employee entity may be described by the name, age, address, salary, and job.

Entity type: (similar to record type)
- The set of relevant attributes constructs the corresponding entity type.
- In ER diagram, it is represented by a rectangular box.

Entity instance or entity occurrence: (similar to record or record value)
- Values for the set of the corresponding relevant attributes construct the entity instance or entity occurrence.
### Entities and Attribute Values

- **Name**: John Smith
- **Address**: 2331 Kierby, Houston, Texas 77001
- **Age**: 55
- **Home Phone**: 713-749-2630

A hierarchy of composite attributes

### Entity Type and Entity Instance Components

**Attribute:**
- It can be classified into
  a. **Composite Attribute:**
     - An attribute that is composed of several more basic attributes.
  b. **Single or Atomic Attribute:**
     - An attribute is not divisible
  c. **Single and Multivalue Attribute:**
     - An attribute may have repeating values.
  d. **Derived Attribute:**
     - The value of an attribute can be derived from a stored

**Key Attribute:**
- An entity type usually has an attribute whose values are distinct for each individual entity.

**Value Set:**
- Each simple attribute of an entity type is associated with a value set (or domain), which specifies the set of values that may be assigned to that attribute for each individual entity.
Entity Types and Some of the Member Entities

**SCHEMA:** Employee Company

**(INTENSION)** Name, Age, Salary Name, Headquarters, President

**INSTANCES:** (EXTENSION)

- \( e_1 \) • (John Smith, 55, 80k)
  - (Sunco Oil, Houston, John Smith)
- \( e_2 \) • (Fred Brown, 40, 30k)
  - (Fast Computer, Dallas, Bob King)
- \( e_3 \) • (Judy Clark, 25, 20k)

Composite and Multivalued Attributes

**CAR**
Registration(RegistrationNumber, State), VehicleID, Make, Model, Year, {Color}

- Car 1 • ((ABC 123, Texas), TK629, Ford Mustang, Convertible, 1986, {red, black})
- Car 2 • ((ABC 123, New York), WP9872, Nissan Sentra, 2-door, 1985, {blue})
- Car 3 • ((VSY 72, Texas), TD729, Chrysler LeBaron, 4-door, 1987, {white, blue})

Page 6
Example of Requirements Collections

- Suppose that after the requirements collection and analysis phase, the database designers stated the following description of the "miniworld", the part of the company to be represented in the database.

1. The company is organized into departments. Each department has a name, a number, and an employee who manages the department. We keep track of the start date when that employee started managing the department. A department may have several locations.

2. A department controls a number of projects, each of which has a name, a number, a single location.

3. We store each employee's name, social security number, address, salary, sex, and birth date. An employee is assigned to one department but may work on several projects, which are not necessarily controlled by the same department. We keep track of the number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee.

4. We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's name, sex, birth date, and relationship to the employee.

Initial Conceptual Design of The Company Database

Identifying Entity Types:

1. An entity type DEPARTMENT with attributes Name, Number, Locations, Manager, and ManagerStartDate. 
   - Locations attribute is the only multivalued attribute.
   - Name and Number is a key attribute.

2. An entity type PROJECT with attributes Name, Number, Location, and ControllingDepartment.
   - Name and Number is a key attribute.

3. An entity type Employee with attributes Name, SSN, Sex, Address, Salary, Birthdate, Department and Supervisor.
   - Both Name and Address may be composite attributes, this was not specified in the requirements. Sometimes we should go back to users.

4. An entity type DEPENDENT with attributes Employee, DependentName, Sex, BirthDate, and Relationship.
Alternatives to Represent the Facts

Example:
Suppose we have the requirement 3, which specifies

a. An employee can work on several projects.
b. Number of hours per week an employee works on each project.

Option1:
To represent it by a multivalued composite attribute of EMPLOYEE called WorksOn and call its simple components (Projects, Hours).

Option2:
To represent it by a multivalued composite attribute of PROJECT called Workers with simple components (Employee, Hours).

Preliminary Design of Entity Type

DEPARTMENT
Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT
Name, Number, Location, ControllingDepartment

EMPLOYEE
Name(FName, Minit,LName), SSN, Sex, Address, Salary, BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

DEPENDENT
Employee, DependentName, Sex, BirthDate, Relationship
Relationship Types and Relationship Instances

Relationship Type:
- A relationship type $R$ among $n$ entity type $E_1, E_2, ..., E_n$ is a set of associations among entities from these types.
- In ER diagram, it is represented by a diamond-shaped box.

Relationship Instance:
- Each relationship instance is an association of entity instances, where the association includes exactly one entity instance from each participating entity type.

Degree of a Relationship:
- It is the number of participating entity types.

Relationships as Attributes:
- Relationships (1:1, 1:N) can be represented by an attribute in the entity.
- M:N relationships can not be represented by an attribute in the implementation of relational database.

Binary Relationship WORKS_FOR

![Diagram of the WORKS_FOR relationship between EMPLOYEE and DEPARTMENT entities. The diagram shows several entity instances and relationship instances connecting them.]
Ternary Relationship SUPPLY

SUPPLIER

s₁
s₂
...

PART

p₁
p₂
p₃
...

SUPPLY

r₁
r₂
r₃
r₄
r₅
r₆
r₇

PROJECT

j₁
j₂
j₃
...

Role Names and Recursive Relationships

Role Name:
- It signifies the role that a participating entity from the entity type plays in each relationship instance.
- Each entity type that participates in a relationship type plays a particular role in the relationship.

Example:
In the WORKS_FOR relationship type, EMPLOYEE plays the role of employee or worker, and DEPARTMENT plays the role of department or employer.

Recursive Relationship:
- A relationship that involves the same entity type participates more than once in the relationship type in different roles.
- In such case, the role name becomes essential and critical for distinguishing the meaning of each participation.

Example:
The EMPLOYEE entity type participates twice in SUPERVISION relationship
1. once in the role of supervisor (or boss)
2. once in the role of supervisee (or subordinate)
EMPLOYEE plays two roles of supervisor (1) and supervisee (2).

Constraints on Relationship Types

- Relationship types usually have certain constraints that limit the possible combinations of entities participating in relationship instances.
- These constraints are determined from the miniworld situation that the relationships represent.

Example:
Each employee works for exactly one department.

Cardinality Ratio Constraint:
- It specifies the number of relationship instances that an entity can participate in.

Example:
The WORKS_FOR binary relationship type
DEPARTMENT:EMPLOYEE is of the cardinality ratio 1:N
It means that each DEPARTMENT entity can be related to many EMPLOYEE entities, but one EMPLOYEE entity can be only related to one DEPARTMENT entity.
- Common cardinality ratios for binary relationship types are 1:1, 1:N, and M:N.
Many-to-Many Relationship WORKS_ON

Participation Constraints

The participation constraints

- It specifies whether the existence of an entity depends on its being related to another entity via the relationship type.

There are two types of participation constraints, total and partial.

Total participation constraints

- Every entity in "the total set" must be related to another entity via relationship type.

Example:

- If a company states that every employee must work for a department, then an employee entity can exist only if it participates in a WORKS_FOR relationship instance.

Partial participation constraints

- Not every entity in "the total set" is related to another entity in a relationship type.

Structural Constraints

Cardinality Ratio Constraint + Participation Constraint
### Relationships

- **One-to-one**
- **One-to-many**
- **Many-to-one**
- **Many-to-many**

**1:1 Relationship MANAGES**

- Partial participation of EMPLOYEE
- Total participation of DEPARTMENT

Diagram showing the relationship between EMPLOYEE, MANAGES, and DEPARTMENT.
Attributes of Relationship Types

- In 1:1 relationship type, the attributes can be included as attributes of one of the participating entity types.

Example:
The StartDate attribute for the MANAGES relationship can be an attribute of either EMPLOYEE or DEPARTMENT.

This is because MANAGES is a 1:1 relationship, so every DEPARTMENT or EMPLOYEE entity participates in at most one relationship instance.

Hence the value of StartDate attribute can be determined separately by either the participating EMPLOYEE entity or participating DEPARTMENT.

Attributes of Relationship Types

- In 1:N relationship type, the attributes can be only included as an attribute of the entity type at the N side of the relationship.

Example:
if the WORKS_FOR relationship also has an attribute StartDate that indicates when an employee started working for a department, this attribute could be included as an attribute of EMPLOYEE.

This is because the relationship is 1:N so each employee entity instance participates in at most one relationship instance in WORKS FOR.

Hence, the participating EMPLOYEE entity alone can determine the value of StartDate.
Weak Entity Types

Weak Entity Type:
- The entity types may not have any key attributes of their own.
- We may not be able to distinguish between some entities because the combinations of values of their attributes can be identical.

How to identify an entity from weak entity type:
- Entities belong to a weak entity type are identified by being related to specific entities from another entity in combination with some of their attribute values.

Identifying Owner:
- The entity used to identify the weak entity.

Identifying Relationship of the Weak Entity Type:
- The relationship type relates a weak entity type to its owner entity.
- A weak entity type always has a total participation constraints (existence dependency) with respect to its identifying relationship.
- It may not be able to identify a weak entity without an owner entity.
- Not every existence dependency (total participation) results in a weak entity type.

Example

Suppose there are two entities from the DEPENDENT entity type:
- DEPENDENT(DependentName, BirthDate, Sex, Relationship)
- It is related to EMPLOYEE entity, which is used to keep track of the dependents of each employee via a 1:N relationship.
- Two dependents of distinct employees may have the same values.
- They are identified as distinct dependent entities only after determining the employee entity to which is related. (or determined by the owner entity)

Partial Key:
- It is a set of attributes in the weak entity type that can uniquely identify weak entities related to the same owner entity.

Example:
- If assume that not two dependents of the same employee ever have the same name, then the attribute DependentName of DEPENDENT is the partial key.

Representation of Weak Entity Type:
- Weak entity type can be represented as a composite, multivalued attributes in the owner entity.
- If weak entity type have many attributes, then the weak entity type itself can be a entity type with an identifying attribute from owner entity type.
Refining the ER Design for the Company Database

- Change the attributes that represent relationships into relationship types.
- Determine the cardinality ratio and participation constraint of each relationship type.

The following relationships are specified based on requirements.

1. MANAGES: a 1:1 relationship type between EMPLOYEE and DEPARTMENT. EMPLOYEE participation is partial, DEPARTMENT participation is total.
2. WORKS_FOR: a 1:N relationship type between DEPARTMENT and EMPLOYEE. Both participations are total.
3. CONTROLS: a 1:N relationship type between DEPARTMENT and PROJECT. Participation of PROJECT is total, DEPARTMENT is partial.
4. SUPERVISION: a 1:N relationship type between EMPLOYEE (in the supervisor role) and EMPLOYEE (in the supervisee role).
   Both participations are partial. Since not every employee is a supervisor and not every employee has a supervisor.
5. WORKS_On: an M:N relationship type with attribute Hours, both participations are total.
6. DEPENDENTS_OF: an 1:N relationship type between EMPLOYEE and DEPENDENT, which is also a identifying relationship for the weak entity type DEPENDENT, participation of EMPLOYEE is partial, and DEPENDENT is total.
### Summary of ER Diagram Notation-1

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>![ENTITY TYPE]</td>
<td>ENTITY TYPE</td>
</tr>
<tr>
<td>![WEAK ENTITY TYPE]</td>
<td>WEAK ENTITY TYPE</td>
</tr>
<tr>
<td>![RELATIONSHIP TYPE]</td>
<td>RELATIONSHIP TYPE</td>
</tr>
<tr>
<td>![IDENTIFYING RELATIONSHIP TYPE]</td>
<td>IDENTIFYING RELATIONSHIP TYPE</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>
</tbody>
</table>
Summary of ER Diagram Notation-2

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COMPOSITE ATTRIBUTE</td>
</tr>
<tr>
<td></td>
<td>DERIVED ATTRIBUTE</td>
</tr>
<tr>
<td>E₁ ⊔ R ⊔ E₂</td>
<td>TOTAL PARTICIPATION OF E₂ IN R</td>
</tr>
<tr>
<td>E₁ ⊔ 1 ⊔ R ⊔ N ⊔ E₂</td>
<td>CARDINALITY RATIO 1:N FOR E₁:E₂ IN R</td>
</tr>
<tr>
<td>R ⊔ (MIN,MAX) ⊔ E₂</td>
<td>STRUCTURAL (MIN,MAX) ON PARTICIPATION OF E IN R</td>
</tr>
</tbody>
</table>

Language Definition of a Conceptual Schema

Schema COMPANY

<table>
<thead>
<tr>
<th>Entity</th>
<th>Attributes</th>
<th>Entity</th>
<th>Attribute</th>
<th>Identifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMPLOYEE</td>
<td>FName: text(10)</td>
<td>PROJECT</td>
<td>PNAME: text(10)</td>
<td>PNAME, PNUMBER</td>
</tr>
<tr>
<td></td>
<td>MINIT: text(10)</td>
<td></td>
<td>PNUMBER: integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LNAME: text(10)</td>
<td></td>
<td>PLOCATION: text(15)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SSN: text(12)</td>
<td></td>
<td>DNO: integer1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BDATE: text(10)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADDRESS: text(50)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SEX: text(1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SALARY: integer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SUPERSSN: text(12)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DNO: integer1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SSN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Identifiers: SSN
Language Definition of a Conceptual Schema

Relationship: WORKS_ON
Connected Entities: (1,n) EMPLOYEE  (1,n) PROJECT
Attributes: HOURS: float

Grammar of Conceptual Schema Definition Language

SCHEMA → Schema: SCHEMA RELATIONSHIP_SECTION
ENTITY_SECTION → (ENTITY_DECL)
ENTITYDECL → Entity: ENTITY_NAME
ATTRIBUTE_SECTION → Attributes: (ATTRIBUTE_DECL)
ATTRIBUTEDECL → ATTRIBUTE-NAME [:TYPE_DEL]
IDENTIFIER DECL → Identifiers: (IDENTIFIER_DECL)
IDENTIFIER LIST → IDENTIFIER_LIST (IDENTIFIER)
RELATIONSHIP_SECTION → (RELATIONSHIP_DECL)
RELATIONSHIPDECL → Relationship: RELATIONSHIP_NAME
CONN_NETDECL → Connected entities: (CONN_ENT_DECL)
ATTRIBUTEDECL → Attributes: (ATTRIBUTE_DECL)
CONN_ENTDECL → [(MIN_CARD, MAX_CARD)] ENTITY_NAME
Relationship Types of Degree Higher Than Two

![Relationship Types of Degree Higher Than Two](image1)

Multiple Binary Relationships

![Multiple Binary Relationships](image2)
ER Model to Relational Model Mapping

Step 1: Mapping Regular Entity

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

Step 2: Mapping Weak Entity

- For each weak entity type W in the ER schema with the owner entity type E, create a relation R and include all simple attributes of W as attributes of R.
- Include as foreign key attributes of R the primary key attributes of the relation that corresponds to the owner entity type E. This take care of the identifying relationship type of W.
- The primary key of R is the combination of the primary key of the owner entity and the partial key of the weak entity.
ER Model to Relational Model Mapping

Step 3: Mapping 1:1 relationship

- 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, 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 of the 1:1 relationship type R as attributes of S.

* 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 particular appropriate when both participations are total and when the entity types do not participate in any other relationship types.

Step 4: Mapping 1:N relationship

- For each regular (nonweak) 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 a 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.

Step 5: Mapping M:N relationship

- 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 combinations will form the primary key of S.

- Include any simple attributes of the M:N relationship type as attributes of S.

* An M:N relationship type can not be represented by a single foreign key attribute in one of the participating relations.
ER Model to Relational Model Mapping

Step 6: Mapping multivalued attribute

- For each multivalued attribute A, create a new relation R that includes an attribute corresponding to A plus the primary key attribute K of the relation that represents the entity type or relationship type that has A as an attribute.
- The primary key of R is then the combination of A and K. If the multivalued attribute is composite, include its simple attributes.

Step 7: Mapping n-ary relationship

- For each n-ary relationship type R, n > 2, create a new relationship S to represent R.
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
- Include any simple 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.

Relational Database Schema
Corresponding to the ER Schema

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th>f.k.</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNAME</td>
<td>MINIT</td>
<td>LNAME</td>
</tr>
<tr>
<td>SSN</td>
<td>BDATE</td>
<td>ADDRESS</td>
</tr>
<tr>
<td>SEX</td>
<td>SALARY</td>
<td>SUPERSSN</td>
</tr>
<tr>
<td>DNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DEPARTMENT</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNAME</td>
<td>DNUMBER</td>
</tr>
<tr>
<td>MGRSSN</td>
<td>MGRSTARTDATE</td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DEPT_LOCATIONS</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNUMBER</td>
<td>DLOCATION</td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WORKS_ON</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESSN</td>
<td>PNO</td>
</tr>
<tr>
<td>HOURS</td>
<td></td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROJECT</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNAME</td>
<td>PNUMBER</td>
</tr>
<tr>
<td>PLOCATION</td>
<td>DNO</td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DEPENDENT</th>
<th>f.k.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESSN</td>
<td>DEPENDENT_NAME</td>
</tr>
<tr>
<td>SEX</td>
<td>BDATE</td>
</tr>
<tr>
<td>p.k.</td>
<td></td>
</tr>
</tbody>
</table>
Comparing Relation Schema with ER Schema

1. Relationship type are not represented explicitly in a relational schema, they are represented by having two attributes A and B, one is primary key, and the other a foreign key - over the same domain - included in two relations S and T.
   - Two tuples in S and T are related when they have the same value for A and B.
   - By using EQUIJOIN operation over S.A and T.B, we can combine all pairs of related tuples from S and T and materialize the relationship.
   - When a binary 1:1 or 1:N relationship type is involved, a single join operation is usually needed.
   - For a binary M:N relationship type, two join operations needed, whereas for n-ary relationship type, n joins will be needed.

2. A separate relation is created for each multivalued attribute.
   - For a particular entity with a set of values for the multivalued attribute, the key attribute value of the entity is repeated once for each value of the multivalued attribute in a separate tuple.
   - This is because the basic relational model does not allow multituple values (or a set of values) for an attribute in a single tuple.

Join Conditions for Materializing the Relationship Types of the Company ER Schema

<table>
<thead>
<tr>
<th>ER Relationship</th>
<th>Participating Relations</th>
<th>Join Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKS_FOR</td>
<td>EMPLOYEE, DEPARTMENT</td>
<td>EMPLOYEE.DNO = DEPARTMENT.DNUMBER</td>
</tr>
<tr>
<td>MANAGES</td>
<td>EMPLOYEE, DEPARTMENT</td>
<td>EMPLOYEE.SSN = DEPARTMENT.MGRSSN</td>
</tr>
<tr>
<td>SUPERVISION</td>
<td>EMPLOYEE(E), EMPLOYEE(S)</td>
<td>EMPLOYEE(E).SUPERSSN = EMPLOYEE(S).SSN</td>
</tr>
<tr>
<td>WORKS_ON</td>
<td>EMPLOYEE, WORKS_ON PROJECT</td>
<td>EMPLOYEE.SSN = WORKS_ON.ESSN AND PROJECT.PNUMBER = WORKS_ON.PNO</td>
</tr>
<tr>
<td>CONTROLS</td>
<td>DEPARTMENT, PROJECT</td>
<td>DEPARTMENT.DNUMBER = PROJECT.DNO</td>
</tr>
<tr>
<td>DEPENDENTS_OF</td>
<td>EMPLOYEE, DEPENDENT</td>
<td>EMPLOYEE.SSN = DEPENDENT.ESSN</td>
</tr>
</tbody>
</table>
### 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 attribute</td>
</tr>
<tr>
<td>simple attribute</td>
<td>set of simple component attributes</td>
</tr>
<tr>
<td>composite</td>
<td>relation and foreign key</td>
</tr>
<tr>
<td>multivalued attribute</td>
<td>domain</td>
</tr>
<tr>
<td>value set</td>
<td>primary (or alternative) key</td>
</tr>
<tr>
<td>key attribute</td>
<td></td>
</tr>
</tbody>
</table>