Database I

(60012301-1)

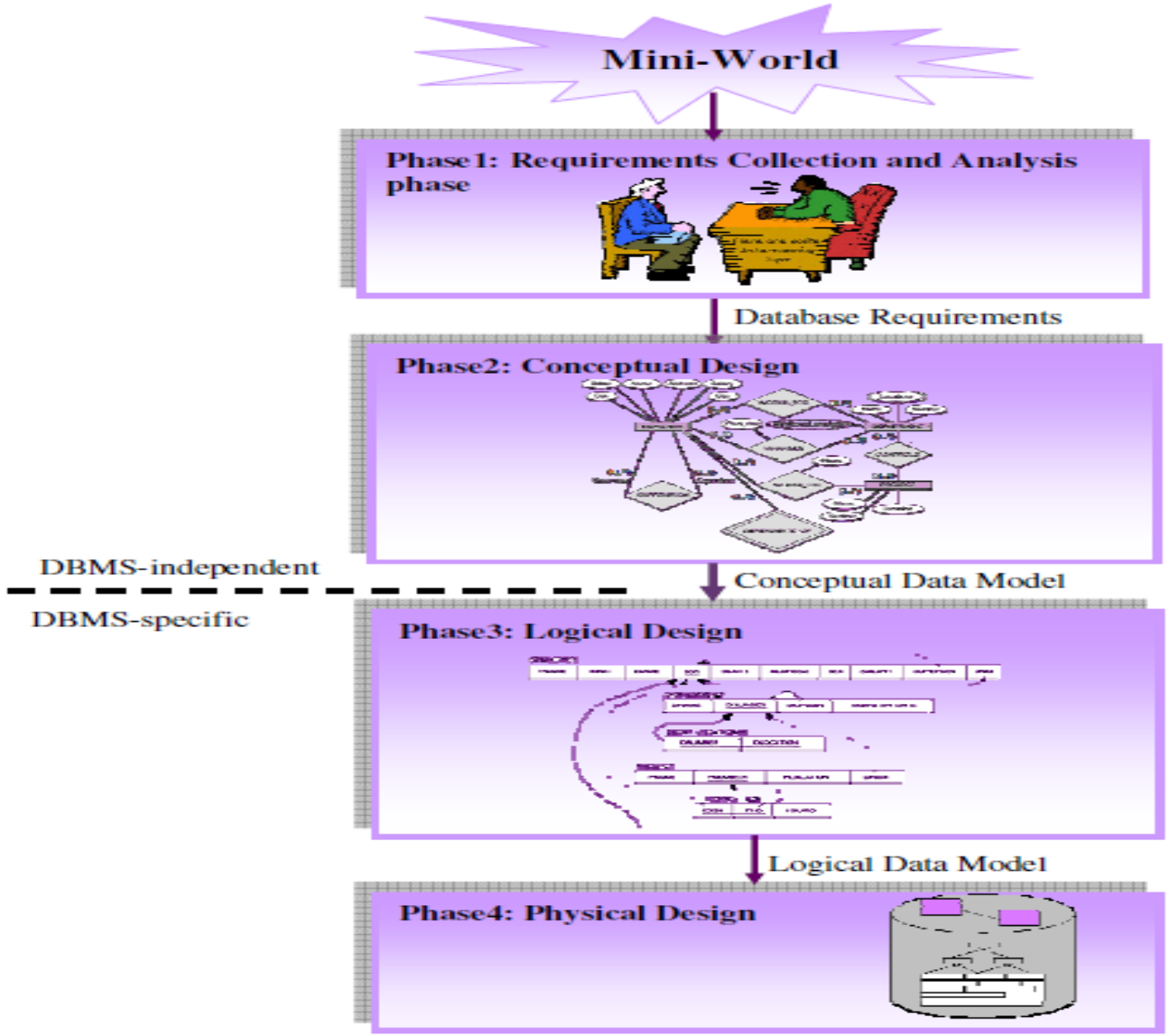
Lecture 5: ER to Relational Mapping

Dr. Obead Alhadreti

Outline

- Logical Data Model
- Relational Model
- Relational Model Constraints
- ER to Relational Mapping

Database Design Phases



Logical Data Model

- The **logical database design phase** maps the conceptual data model into a logical model, which is influenced by the data model for the target database (for example, the relational model).

- The logical Model is Intermediate between entity-relationship model and physical data model.

Relational Models

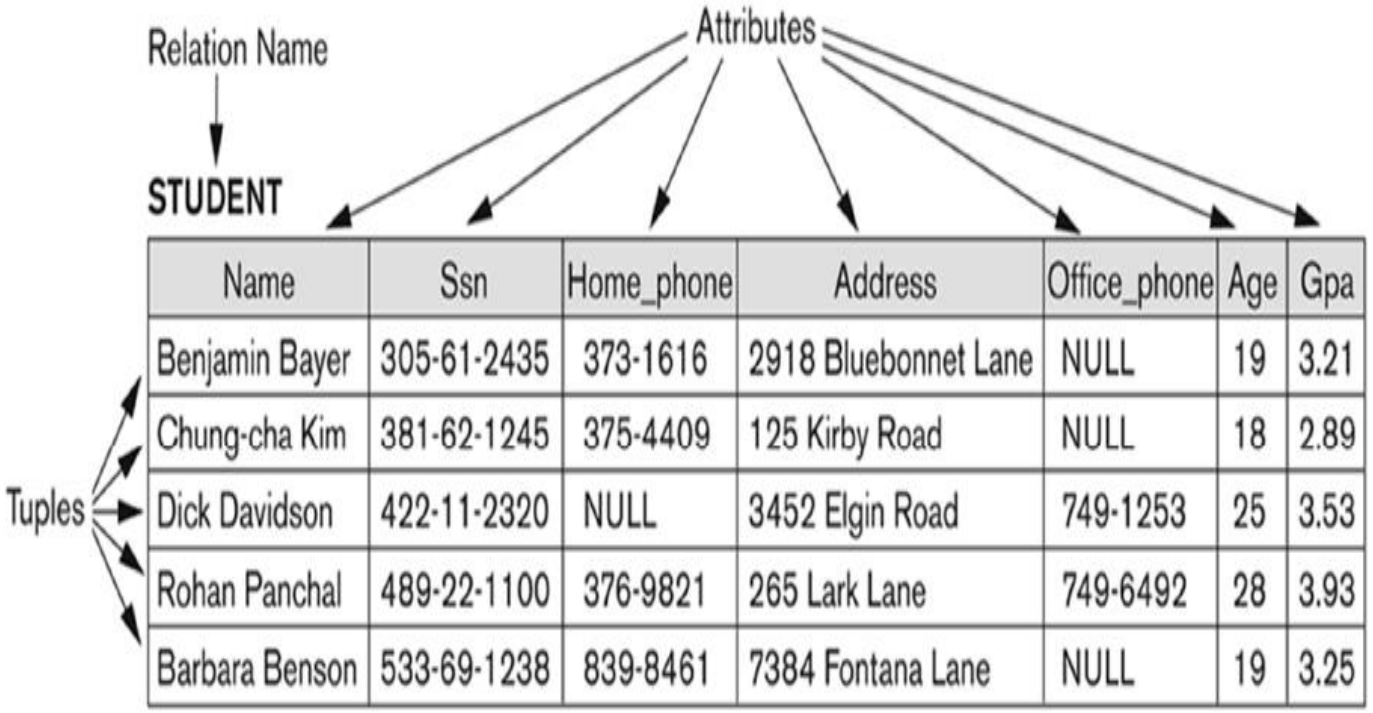
- The Relational Model of Data is based on the concept of a *Relation*.
- Relational Model was proposed by Edgar Codd in 1969 to model data in the form of relations or tables.
- It represents how data is stored in Relational Databases.

Relational Models Concepts

- **Relation** looks like a **table**.
- **Attribute** are the properties that define a relation. It is the column header that gives an indication of the meaning of the data items in that column.
- **Tuple:** Each row in the relation is known as tuple. The data elements in each **row** represent certain facts that correspond to a real-world **entity** or **relationship.**

Example of a Relation

- A relation STUDENT.



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

Relational Models Concepts

- **Relation Schema:** A relation schema represents name of the relation with its attributes. If a schema has more than 1 relation, it is called Relational Schema.

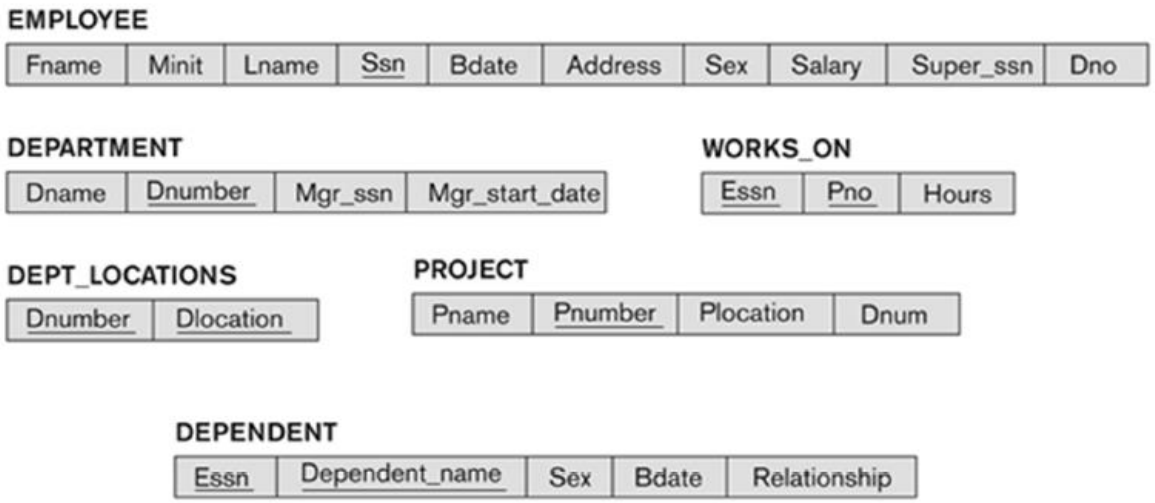


Fig 4.5. Schema diagram for the COMPANY relational database schema

Relational Models Concepts

- **Relation Instance:** The set of tuples of a relation at a particular instance of time is called as relation instance.
- **Degree:** The number of attributes in the relation is known as degree of the relation. The **STUDENT** relation defined in the previous slide has degree 7.
- **Cardinality:** The number of tuples in a relation is known as cardinality. The **STUDENT** relation defined above has cardinality 5.

Relational Models Concepts

- **Column:** Column represents the set of values for a particular attribute. The column **Gpa** is extracted from relation STUDENT.

Gpa
3.21
2.89
3.53
3.93
3.25

- **Relation key:** Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely. The attribute Ssn is the relation key for the relation STUDENT.

Relational Integrity Constraints

- Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**. There are three main integrity constraints:
  1. Key constraints
  2. Domain constraints
  3. Referential integrity constraints

- These constraints are checked before performing any operation (insertion, deletion and updation) in database. If there is a violation in any of constrains, operation will fail.

Key constraints

- Every relation in the database should have at least one set of attributes which defines a tuple uniquely. Those set of attributes is called key.

- If there are more than one such minimal subsets, these are called ***candidate keys***.

- A key has two properties:
  1. It should be unique for all tuples.
  2. It can't have NULL values.

Key constraints

- If a relation has several **candidate keys**, one is chosen to be the **primary key**.
- The **primary key** value is used to *uniquely identify* each tuple in a relation.
- Also used to *reference* the tuple from another tuple.
- Choose as primary key the smallest of the candidate keys (in terms of size).
- The primary key attributes are underlined.

Domain Constraints

- Attributes have specific values in real-world scenario.
- An attribute can only take values which lie inside the domain range.
- For example, age can only be a positive integer.

Referential integrity Constraints

- Referential integrity constraint states that if a relation refers to a key attribute of a different or *same* relation, then that key element must exist.

- Used to specify a **relationship** among tuples in two relations: The **referencing relation** and the **referenced relation**.

- Tuples in the **referencing relation** R1 have attributes called **foreign key** attributes (FK) that reference the primary key attributes PK of the **referenced relation** R2.

- A **foreign key** is a key attribute of a relation that can be referred in other relation. It is followed by #

Example

STUDENT

ROLL_NO	NAME	ADDRESS	PHONE	AGE	BRANCH_CODE
1	RAM	DELHI	9455123451	18	CS
2	RAMESH	GURGAON	9652431543	18	CS
3	SUJIT	ROHTAK	9156253131	20	ECE
4	SURESH	DELHI		18	IT

BRANCH

BRANCH_CODE	BRANCH_NAME
CS	COMPUTER SCIENCE
IT	INFORMATION TECHNOLOGY
ECE	ELECTRONICS AND COMMUNICATION ENGINEERING
CV	CIVIL ENGINEERING

BRANCH_CODE of STUDENT can only take the values which are present in BRANCH_CODE of BRANCH which is called referential integrity constraint. The relation which is referencing to other relation is called REFERENCING RELATION (STUDENT in this case) and the relation to which other relations refer is called REFERENCED RELATION (BRANCH in this case).

ER to Relational Mapping

Mapping from ER to Relational Model

- Data Model Mapping refer to the process of converting Entity-relationship schema to Relational schema which can directly be implemented by any Relational database management system (DBMS) like Oracle, MySQL etc.

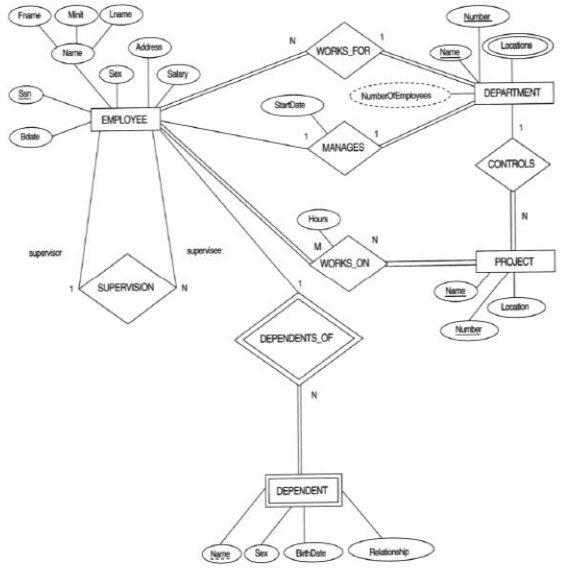


Figure 5.1: The ER conceptual schema diagram for the COMPANY database.

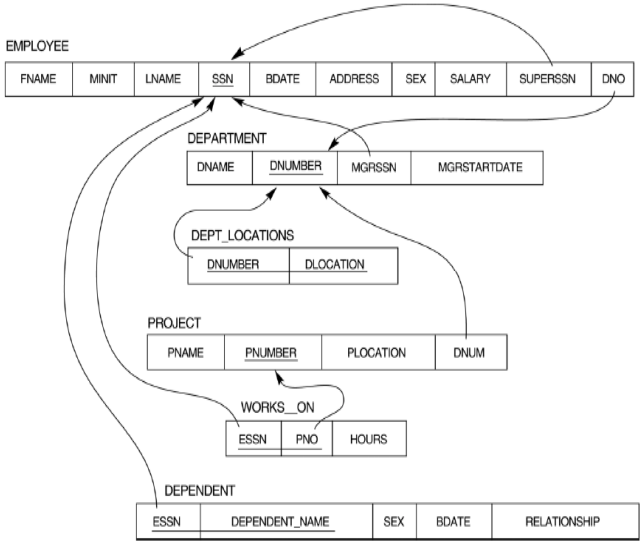


Figure 5.2: Result of mapping the COMPANY ER schema into a relational schema.

ER to Relational Mapping Algorithm

- To convert an Entity-Relationship model to a relational database schema, a procedure of seven steps should be followed:

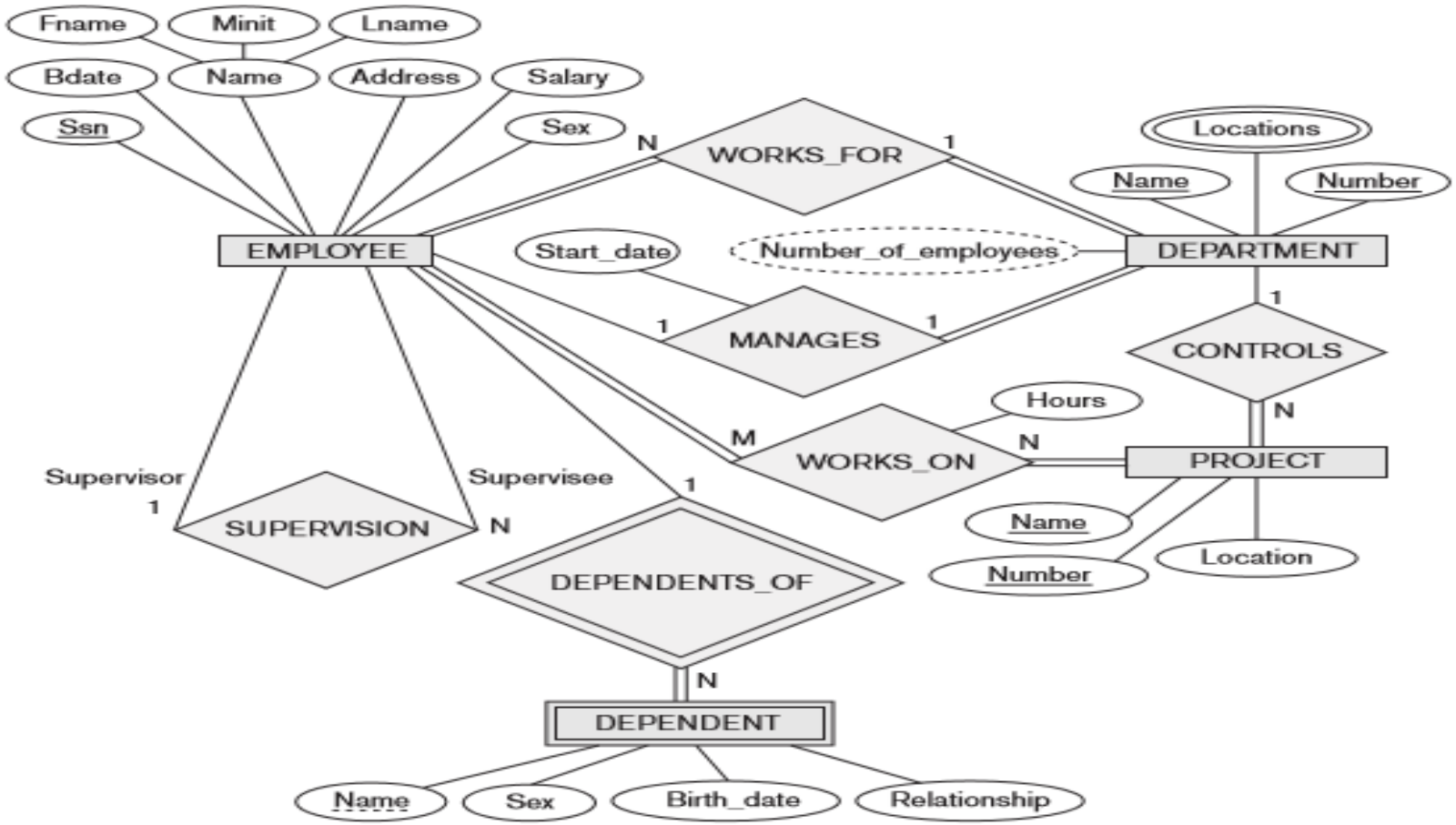
  Step 1: Mapping of Regular Entity Sets.
  Step 2: Mapping of Weak Entity Sets
  Step 3: Mapping of Binary 1:1 Relationship Sets
  Step 4: Mapping of Binary 1:N Relationship Sets.
  Step 5: Mapping of Binary M:N Relationship Sets.
  Step 6: Mapping of Multivalued attributes.
  Step 7: Mapping of N-ary Relationship Sets.

Step 1: Mapping of Regular Entity Sets

- For each regular entity type in the ER diagram, create a relation.
- Simple attributes are represented as fields:
  - multi-valued attributes represented as a relation
  - derived attributes are not usually stored
  - Composite attributes are usually split into several fields
- Choose one of the key attributes of entity type as the primary key for the relation.
- If the chosen key of entity type is composite, the set of simple attributes that form it will together form the primary key of the relation.

Example

Figure 9.1
The ER conceptual schema diagram for the COMPANY database.



Example

- We create the relations **EMPLOYEE**, **DEPARTMENT**, and **PROJECT** in the relational schema corresponding to the regular entities in the ER diagram. **SSN**, **DNUMBER**, and **PNUMBER** are the primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT as shown.

Step 1 shows in fig 9.3 (a)

Figure 9.3
Illustration of some mapping steps.
(a) *Entity* relations after step 1.
(b) Additional *weak entity* relation after step 2.
(c) *Relationship* relations after step 5.
(d) Relation representing multivalued attribute after step 6.

(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 2: Mapping of Weak Entity Sets

- For each **weak** entity type, create a relation and include all simple and components of composite attributes of the entity type as attributes of the relation.

- Include primary key attribute of strong entity as ***foreign key attributes*** of the relation.

- The ***primary key*** of relation is the combination of the ***primary key*** of the strong entity and ***the partial key*** of the weak entity type, if any.

Example

- Create the relation **DEPENDENT.** In this step to correspond to the weak entity type **DEPENDENT**. Include the primary key **SSN** of the **EMPLOYEE** relation as a foreign key attribute of **DEPENDENT** (renamed to **ESSN**). The primary key of the **DEPENDENT** relation is the combination {**ESSN**, **DEPENDENT_NAME**} because **DEPENDENT_NAME** is the partial key of **DEPENDENT**. as shown.

Example

Figure 9.3
Illustration of some mapping steps.
(a) *Entity* relations after step 1.
(b) Additional *weak entity* relation after step 2.
(c) *Relationship* relations after step 5.
(d) Relation representing multivalued attribute after step 6.

Step 2 represents in (b)

(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

Mapping binary relationships

- Identify one entity as "parent"
- other entity as "child"
- As general rule,
  PK of parent is added to child as FK
- Any attributes of the relationship
  - are added to ***child*** relation

Step 3: Mapping of Binary 1:1 Relationship Sets

- For each binary 1:1 relationship type, identify relations that correspond to entity types participating in the relation.

- **Use the foreign key approach:** Identify one of the relations (child) and include in it the primary key of other relation as a foreign key. It is better to choose an entity type with total participation in relationship in the role of child. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type as attributes of child relation.

Step 3: Mapping of Binary 1:1 Relationship Sets

- Example: 1:1 relation **MANAGES** is mapped by choosing the participating entity type **DEPARTMENT** to serve in the role of child, because its participation in the **MANAGES** relationship type is total. (every department has a manager).

- We include the primary key of the EMPLOYEE relation as foreign key in the DEPARTMENT relation and rename it to Mgr_ssn. We also include the simple attribute Start_date of the MANAGES relationship type in the DEPARTMENT relation and rename it Mgr_start_date.

Step 4: Mapping of Binary 1:N Relationship Sets

- For each regular binary 1:N relationship type:
  - Identify relation (child) that represents participating entity type at N-side of relationship type
  - Include primary key pf other entity type as foreign key in child entity.
  - Include simple attributes of 1:N relationship type as attributes of child entity.

- Example: 1:N relationship types **WORKS_FOR**, **CONTROLS**, and **SUPERVISION** in the figure. For **WORKS_FOR** we include the primary key **DNUMBER** of the **DEPARTMENT** relation as foreign key in the **EMPLOYEE** relation and call it **DNO**.

Step 5: Mapping of Binary M:N Relationship Sets

- For each binary M:N relationship type:
  - Create a new relation.
  - Add primary key of participating entity types as foreign key attributes in the new relation.
  - Include any simple attributes of M:N relationship type.

Example

- Example: The M:N relationship type **WORKS_ON** from the ER diagram is mapped by creating a relation **WORKS_ON** in the relational database schema. The primary keys of the **PROJECT** and **EMPLOYEE** relations are included as foreign keys in **WORKS_ON** and renamed **PNO** and **ESSN**, respectively.

- Attribute **HOURS** in **WORKS_ON** represents the **HOURS** attribute of the relation type. The primary key of the **WORKS_ON** relation is the combination of the foreign key attributes {**ESSN, PNO**}.

Example

Figure 9.3
Illustration of some mapping steps.
(a) *Entity* relations after step 1.
(b) Additional *weak entity* relation after step 2.
(c) *Relationship* relations after step 5.
(d) Relation representing multivalued attribute after step 6.

(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

Step 5 represents in (c)

(c) **WORKS_ON**

Essn	Pno	Hours

(d) **DEPT_LOCATIONS**

Dnumber	Dlocation

Step 6: Mapping of Multivalued Attributes

- For each multivalued attribute create a new relation that contain:
  - The attribute itself.
  - The primary key of the "parent" entity as foreign key
  - The primary key of the new relation is usually made up of all its attributes.
  - If the multivalued attribute is composite, include its simple components.

Example

- The relation **DEPT_LOCATIONS** is created. 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 R is the combination of {**DNUMBER, DLOCATION**}

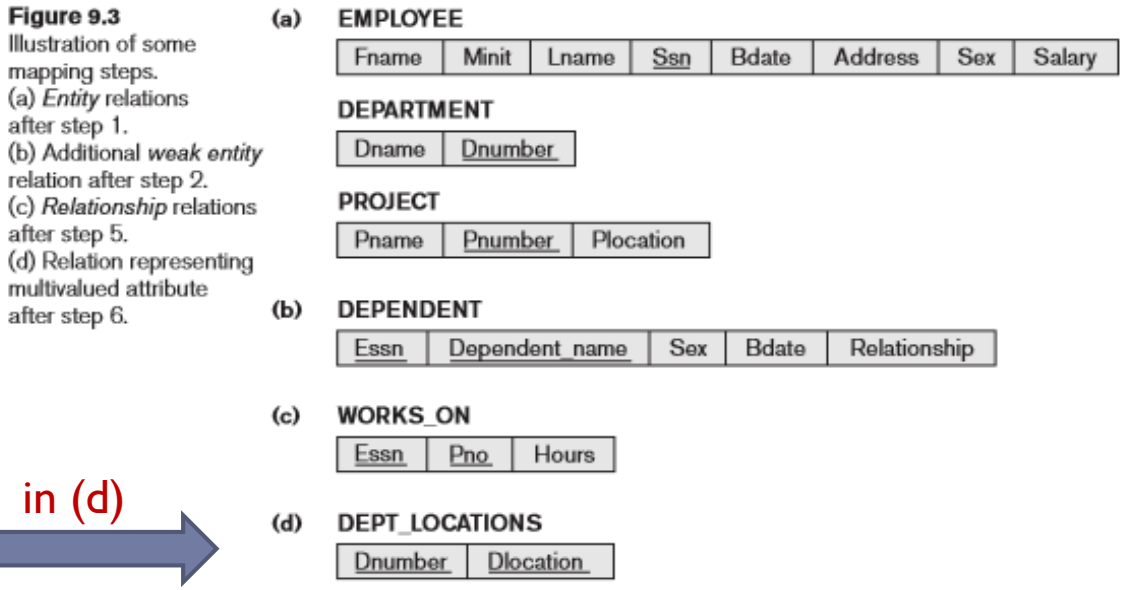


Figure 9.3
Illustration of some mapping steps.
(a) *Entity* relations after step 1.
(b) Additional *weak entity* relation after step 2.
(c) *Relationship* relations after step 5.
(d) Relation representing multivalued attribute after step 6.

Step 6 represents in (d)

Step 7: Mapping of N-ary Relationship Sets

- For each n-ary relationship type
  - Create a new relation.
  - Made up of the primary keys from the n participating relations, as foreign keys.
  - plus any attributes of the relationship.

Summary of Mapping constructs and constraints

Table 9.1 Correspondence between ER and Relational Models

ER MODEL	RELATIONAL MODEL
Entity type	*Entity* relation
1:1 or 1:N relationship type	Foreign key (or *relationship* relation)
M:N relationship type	*Relationship* relation and *two* foreign keys
n-ary relationship type	*Relationship* relation and *n* foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Value set	Domain
Key attribute	Primary (or secondary) key

The Relational Database Schema Resulted

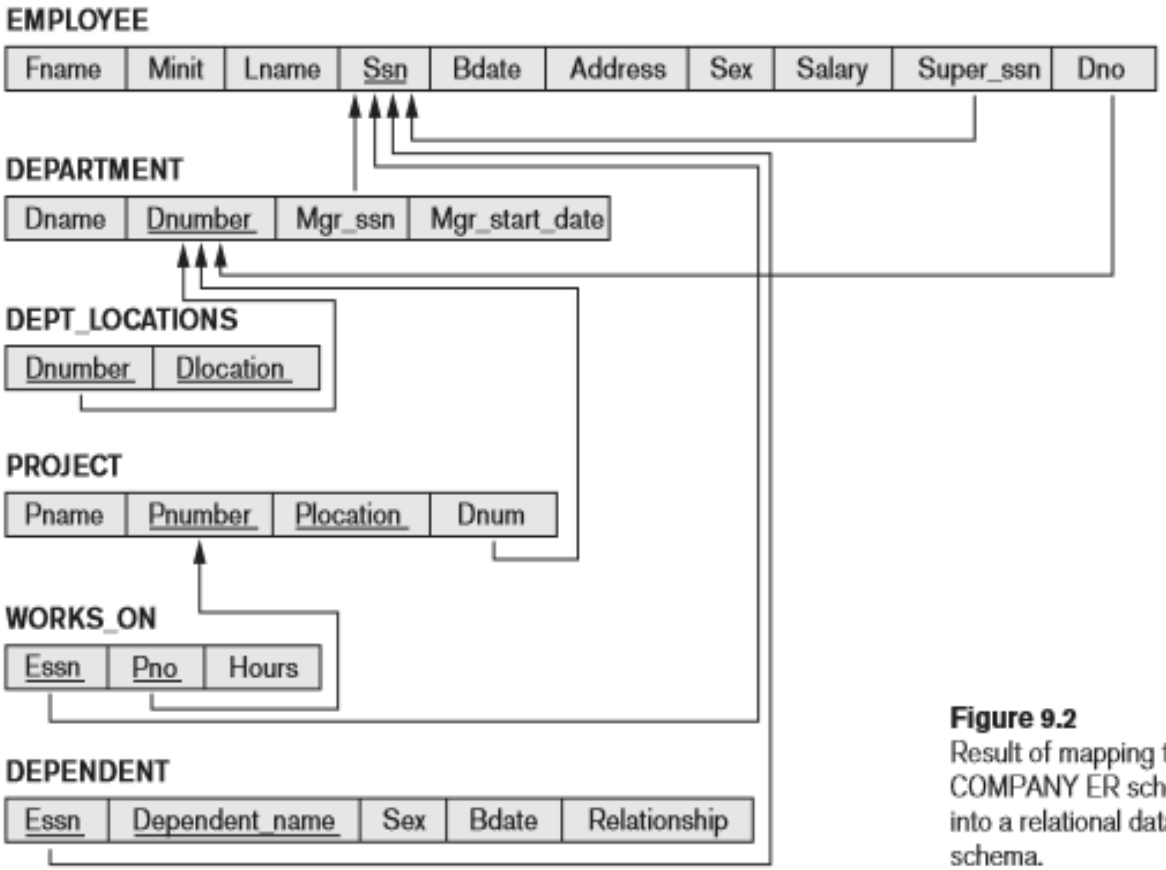


Figure 9.2
Result of mapping the COMPANY ER schema into a relational database schema.