The Relational Model and Relational Database Design

Modified from Chapter 5 and 9 by Ramez Elmasri and Shamkant B. Navathe, Fundamentals of Database Systems, Sixth Edition,

Pearson Education, 2017

Chapter Outline

- Relational Model Concepts
- Relational Model Constraints and Relational Database Schemas
- Insert, Delete, and Update Operations and Dealing with Constraint Violations
- Relational Database Design by Relational Mapping

Relational Model Concepts

The relational model was first proposed by Dr. E.F. Codd of IBM in 1970 in the following paper: "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970.

The relational model of data is based on the concept of a **Relation**.

A Relation is a mathematical concept based on the ideas of sets.

INFORMAL DEFINITIONS

- Informally, a **relation** looks like a **table** of values.
- A relation typically contains a **set of rows**.
- The data elements in each **row** represent certain facts that correspond to a real-world **entity** or **relationship**
- Each **column** has a column header that gives an indication of the meaning of the data items in that column

INFORMAL DEFINITIONS (cont.)

- **Key of a Relation:**
  - Each row has a value of a data item (or set of items) that **uniquely identifies** that row in the table
    - **Called the *key***
  - In the STUDENT table, SSN is the key
  - Sometimes row-ids or sequential numbers are assigned as keys to identify the rows in a table

Example

Relation name

Column

Tuples

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	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
	Charles Cooper	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

FORMAL DEFINITIONS

- R (A1, A2, ...An)
- **R** is called the ***name of this relation***.
- Relation schema **R** is defined over **attributes A1, A2, ..., An**
- Each **attribute** Ai is the name of a role played by some **Domain D** in the relational schema R
- D is called the **domain** of Ai, and is denoted by **dom(Ai)**
- The **degree** (or **arity**) of a relation is the **number of attributes *n* of its relation schema**.

FORMAL DEFINITIONS (cont.)

A relation may be regarded as a set of tuples (rows).

A **tuple** is an ordered set of values

Columns in a table are also called **attributes** of the relation.

STUDENT(Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)

Relation name

Attributes

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	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
	Charles Cooper	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

Tuple

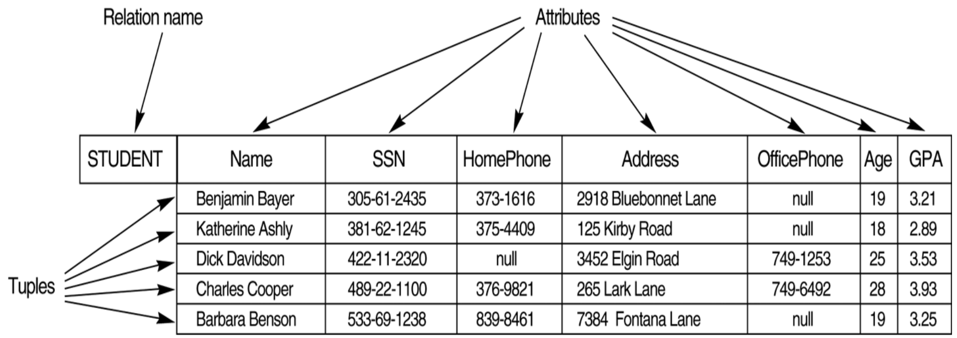
~~Row~~

Examples of Domains

- **Social_security_numbers (SSN):**
  - The set of valid nine-digit Social Security numbers.
  - (This is a unique identifier assigned to each person in the United States for employment, tax, and benefits purposes.)
- **Names:**
  - The set of character strings that represent names of persons.
- **GPA:**
  - Possible values of computed grade point averages;
  - each must be a real (floating-point) number between 0 and 4.
- **Age:**
  - Possible ages of student; each must be an integer value between 15 and 40.

FORMAL DEFINITIONS (cont.)

- Formally,
  - Given R(A1, A2, ..., An)
  - r(R) $\subset$ dom (A1) X dom (A2) X ....X dom(An)
- R(A1, A2, ..., An) is the **schema** of the relation
- **R** is the **name** of the relation
- A1, A2, ..., An are the **attributes** of the relation
- r(R): a specific **state** (or "value" or "population") of relation R – this is a *set of tuples* (rows)
  - r(R) = {t1, t2, ..., tn} where each ti is an n-tuple
  - ti = <v_1, v_2 , ..., v_n > where each v_i *element-of* dom(A_i)



STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	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
	Charles Cooper	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

FORMAL DEFINITIONS (cont.)

- Let R(A1, A2) be a relation schema:
  - Let dom(A1) = {0,1}
  - Let dom(A2) = {a,b,c}
- Then: dom(A1) X dom(A2) is all possible combinations:
  {<0,a> , <0,b> , <0,c>, <1,a>, <1,b>, <1,c> }

- The relation state r(R) ⊂ dom(A1) X dom(A2)
- For example: r(R) could be {<0,a> , <0,b> , <1,c> }
  - this is one possible state (or "population" or "extension") r of the relation R, defined over A1 and A2.
  - **It has three 3-tuples: <0,a> , <0,b> , <1,c>**

DEFINITION SUMMARY

Informal Terms	Formal Terms
Table	Relation
Column Header	Attribute
All possible Column Values	Domain
Row	Tuple
Table Definition	Schema of a Relation
Populated Table	State of the Relation

Characteristics of Relations

- **Ordering of tuples** in a relation r(R):
  - **The tuples are not considered to be ordered,** even though they appear to be in the tabular form.
- **Ordering of attributes** in a relation schema R:
  - **We will consider the attributes in R(A1, A2, ..., An) and the values in t = <v1, v2, ..., vn> to be ordered.**
  - However, a more general alternative definition of relation does not require this ordering.
- **Values in a tuple**:
  - **All values are considered atomic (indivisible).**
  - A special **null** value is used to represent values that are **unknown** or **inapplicable** to certain tuples.

Characteristics of Relations (cont.)

- Notation:
  - **R(A1,A2, ..., An)** is a relational schema R of degree n
  - We refer to **component values** of a tuple t by:
    - $t[A_i]$ or $t.A_i$
    - This is the value vi of attribute Ai for tuple t
  - Similarly, t[Au, Av, ..., Aw] refers to the subtuple of t containing the values of attributes Au, Av, ..., Aw, respectively in t

Characteristics of Relations (cont.)

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21

- **t3.Age = 28**
- **t1[Age] = 25**
- **t5[SSN, Name, GPA] = 305-61-2435, Benjamin Bayer, 3.21**

Relational Database Constraints

Inherent model-based constraints

- Constraints that are inherent in the model
- **For example, no duplication tuples**

Schema-based constraints

- expressed in the schemas of the data model (specified by DDL)

Application-based constraints

- expressed and enforced by the application programs

Schema-based Constraints

- Domain constraints
- Key constraints
- Entity integrity constraints
- Referential integrity constraints
- Other types of constraints

Domain Constraints

Within each tuple, the value of each attribute A must be an atomic value from the domain dom(A)

data types for each attribute such as integers, real numbers, characters, booleans, date, time, time-stamp, etc.

a subrange of values, an enumerated type

Key Constraints

A relation is defined as a set of tuples

All element of a set are distinct

- **no 2 tuples can have the same combination of values for all their attributes**

Usually, there are other subset of attributes of R with the property that no two tuples in any relation state r of R should have the same combination of values for these attributes

denote such the subset by SK (Superkey)

A **Superkey** is a set of one or more attributes in a relational database that uniquely identifies a record in a table.

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
t1	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
t2	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25
t3	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
t4	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
t5	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21

t1[Name,SSN,Age] = Dick Davidson, 422112320, 25
t2[Name,SSN,Age] = Barbara Benson, 533691238, 19

t1[SK] is not equal to t2[SK]

Key Constraints (cont.)

- Every relation has at least one **default superkey** - the **set of all its attribute**
- **Key** of R:
  - A "minimal" superkey
  - That is, a key is a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey (does not possess the superkey uniqueness property)
  - **Key** คือ **Superkey** ที่มีคุณสมบัติพิเศษคือ ไม่สามารถลบ attribute ใดๆ ออกจากมันแล้วจะยังคงความเป็น Superkey

Key Constraints (cont.)

- Example: The CAR relation schema:

  CAR(State, RegNo, SerialNo, Make, Model, Year)

  has two keys

    - Key1 = {State, RegNo}, Key2 = {SerialNo}, which are also superkeys.

  - {SerialNo, Make} is a superkey but not a key.

- **Hence, a key is a superkey but not vice versa.**
  - **A superkey may be a key (if it is minimal) or may not be a key (if it is not minimal).**
- If a relation has several candidate keys, one is chosen arbitrarily to be the primary key. The **primary key** attributes are **underlined**.

The CAR relation, with two candidate keys: {State, RegNo} and {SerialNo}.

CAR	State	RegNo	SerialNo	Make	Model	Year
	Texas	ABC-739	A69352	Ford	Mustang	96
	Florida	TVP-347	B43696	Oldsmobile	Cutlass	99
	New York	MPO-22	X83554	Oldsmobile	Delta	95
	California	432-TFY	C43742	Mercedes	190-D	93
	California	RSK-629	Y82935	Toyota	Camry	98
	Texas	RSK-629	U028365	Jaguar	XJS	98

Entity Integrity

- Relational Database Schema: A set S of relation schemas that belong to the same database. S is the name of the database.
- S = {R1, R2, ..., Rn}
- Entity Integrity:
  - **The primary key attributes PK of each relation schema R in S cannot have null values in any tuple of r(R).**
  - This is because primary key values are used to identify the individual tuples.
- t[PK] is not null for any tuple t in r(R)
- Note: Other attributes of R may be similarly constrained to disallow null values, even though they are not members of the primary key.

Schema diagram for the COMPANY relational database schema.

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

One possible database state for the COMPANY database schema.

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPT_LOCATIONS	DNUMBER	DLOCATION
	1	Houston
	4	Stafford
	5	Bellaire
	5	Sugarland
	5	Houston

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

WORKS_ON	ESSN	PNO	HOURS
	123456789	1	32.5
	123456789	2	7.5
	666884444	3	40.0
	453453453	1	20.0
	453453453	2	20.0
	333445555	2	10.0
	333445555	3	10.0
	333445555	10	10.0
	333445555	20	10.0
	999887777	30	30.0
	999887777	10	10.0
	987987987	10	35.0
	987987987	30	5.0
	987654321	30	20.0
	987654321	20	15.0
	888665555	20	null

PROJECT	PNAME	PNUMBER	PLOCATION	DNUM
	ProductX	1	Bellaire	5
	ProductY	2	Sugarland	5
	ProductZ	3	Houston	5
	Computerization	10	Stafford	4
	Reorganization	20	Houston	1
	Newbenefits	30	Stafford	4

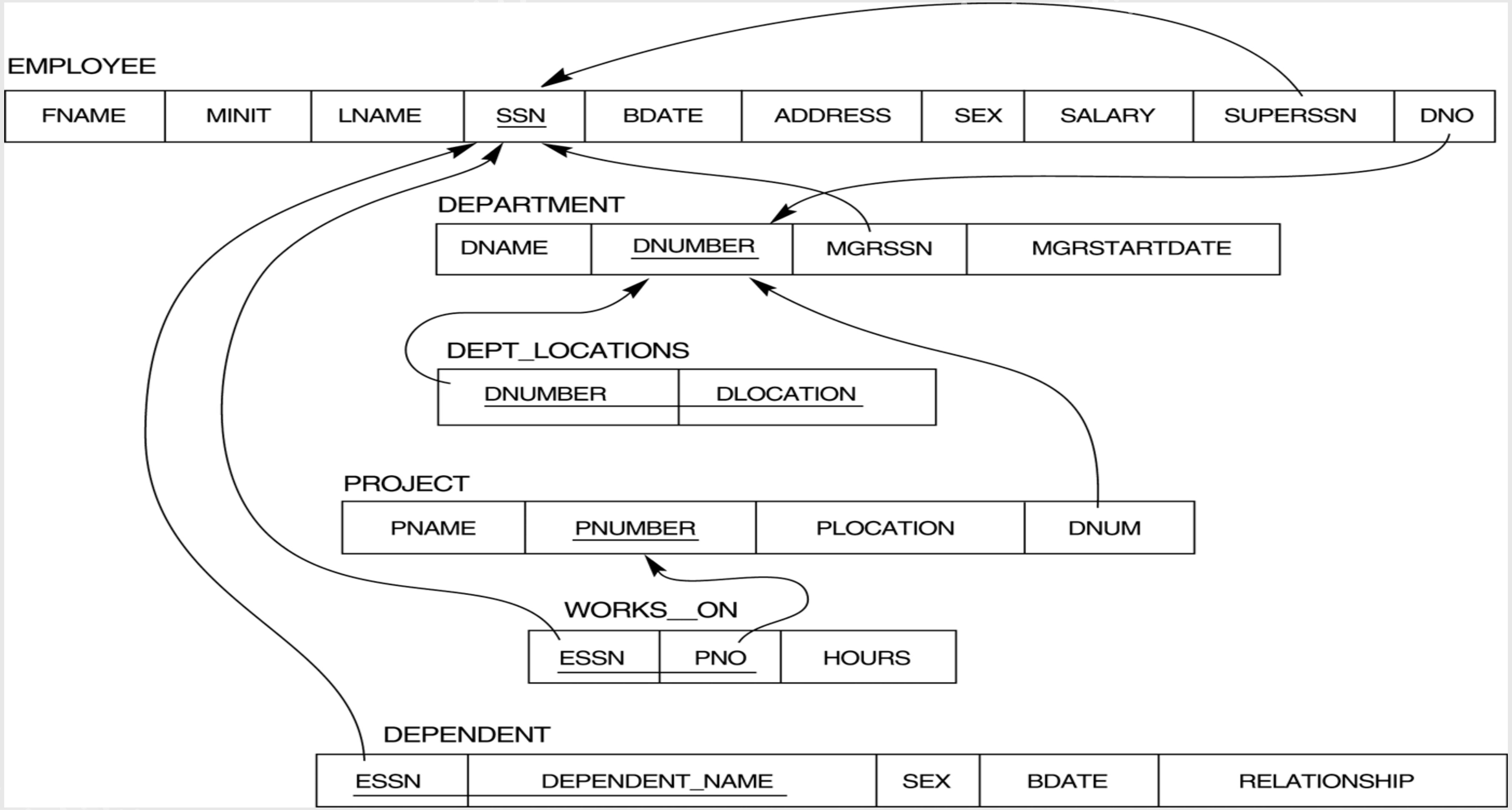
DEPENDENT	ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP
	333445555	Alice	F	1986-04-05	DAUGHTER
	333445555	Theodore	M	1983-10-25	SON
	333445555	Joy	F	1958-05-03	SPOUSE
	987654321	Abner	M	1942-02-28	SPOUSE
	123456789	Michael	M	1988-01-04	SON

Referential Integrity Constraint

- **A constraint involving two relations.**
- 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 FK (called foreign key attributes) that reference the primary key attributes PK of the referenced relation R2.
- A tuple t1 in R1 is said to reference a tuple t2 in R2

  if t1[FK] = t2[PK].
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R1.FK to R2.PK

Referential integrity constraints displayed on the COMPANY relational database schema.



Referential Integrity Constraint

- Statement of the constraint
  - The value in the foreign key column (or columns) FK of the the referencing relation R1 can be either:

    (1) a value of an existing primary key value of the corresponding primary key PK in the referenced relation R2, or..

    (2) a null.

  - **In case (2), the FK in R1 should not be a part of its own primary key.**

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

Other Types of Constraints

Semantic Integrity Constraints:

- based on application semantics and cannot be expressed by the model
- E.g., “the max. no. of hours per employee for all projects he or she works on is 56 hrs per week”
- Mechanisms called triggers and assertions can be used

Functional Dependency Constraints:

- establish a functional relationship among two sets of attributes X and Y (eg. Value of X determines the value of Y denoted by X -> Y)

Update Operations on Relations

INSERT	• insert a new tuple or tuples in a relation
DELETE	• delete a tuple or tuples in a relation
UPDATE or MODIFY	• change the values of some attributes in existing tuples

Integrity constraints should not be violated by these operations.

The INSERT Operation

- Provide a list of attribute values for a new tuple *t* that is to be inserted into a relation R
- Insert can violate any constraints discussed
  - **domain constraints**: if an attribute value violates the corresponding constraint
  - **key constraints**: if a key value in the new tuple **t** already exists in another tuple in the relation r(R)
  - **entity integrity**: if the PK of the new tuple **t** is null
  - **Referential integrity**: if the value of any foreign key in **t** refers to a tuple that does not exist in the referenced relation

The DELETE Operation

Delete can be violated only referential integrity

- if the tuple being deleted is referenced by the foreign keys from other tuples in the database

In case that the delete violation occurs, several options can be occurred:

- **reject the deletion**
- **attempt to cascade (propagate)** the deletion by deleting tuples that reference the tuple that is being deleted
- **modify the referencing attribute values** that cause the violation (set to be null or change to another valid tuple)
- **Execute a user-specified error-correction routine**
- Note: if the referencing attribute is a part of PK, it can't be set to be a null value

The UPDATE Operation

- Update can **violate domain constraints, primary key, entity integrity, and referential constraints.**

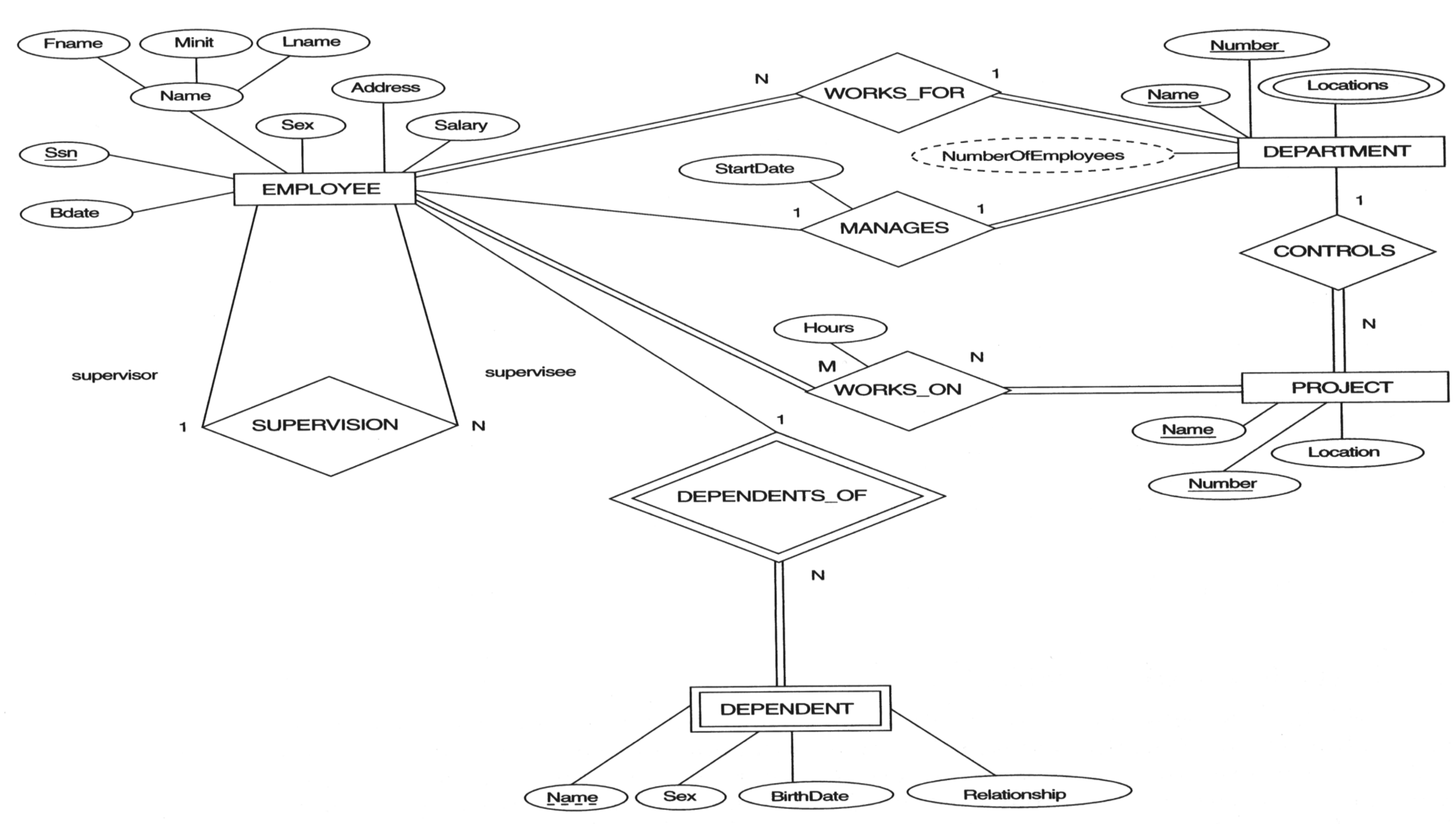
Relational Database Design by Relational Mapping

- ER-to-Relational Mapping Algorithm
  - Step 1: Mapping of Regular Entity Types
  - Step 2: Mapping of Weak Entity Types
  - Step 3: Mapping of Binary 1:1 Relation Types
  - Step 4: Mapping of Binary 1:N Relationship Types
  - Step 5: Mapping of Binary M:N Relationship Types
  - Step 6: Mapping of Multivalued attributes
  - Step 7: Mapping of N-ary Relationship Types
  - Step 8: Options for Mapping Specialization or Generalization
  - Step 9: Mapping of Union Types (Categories)

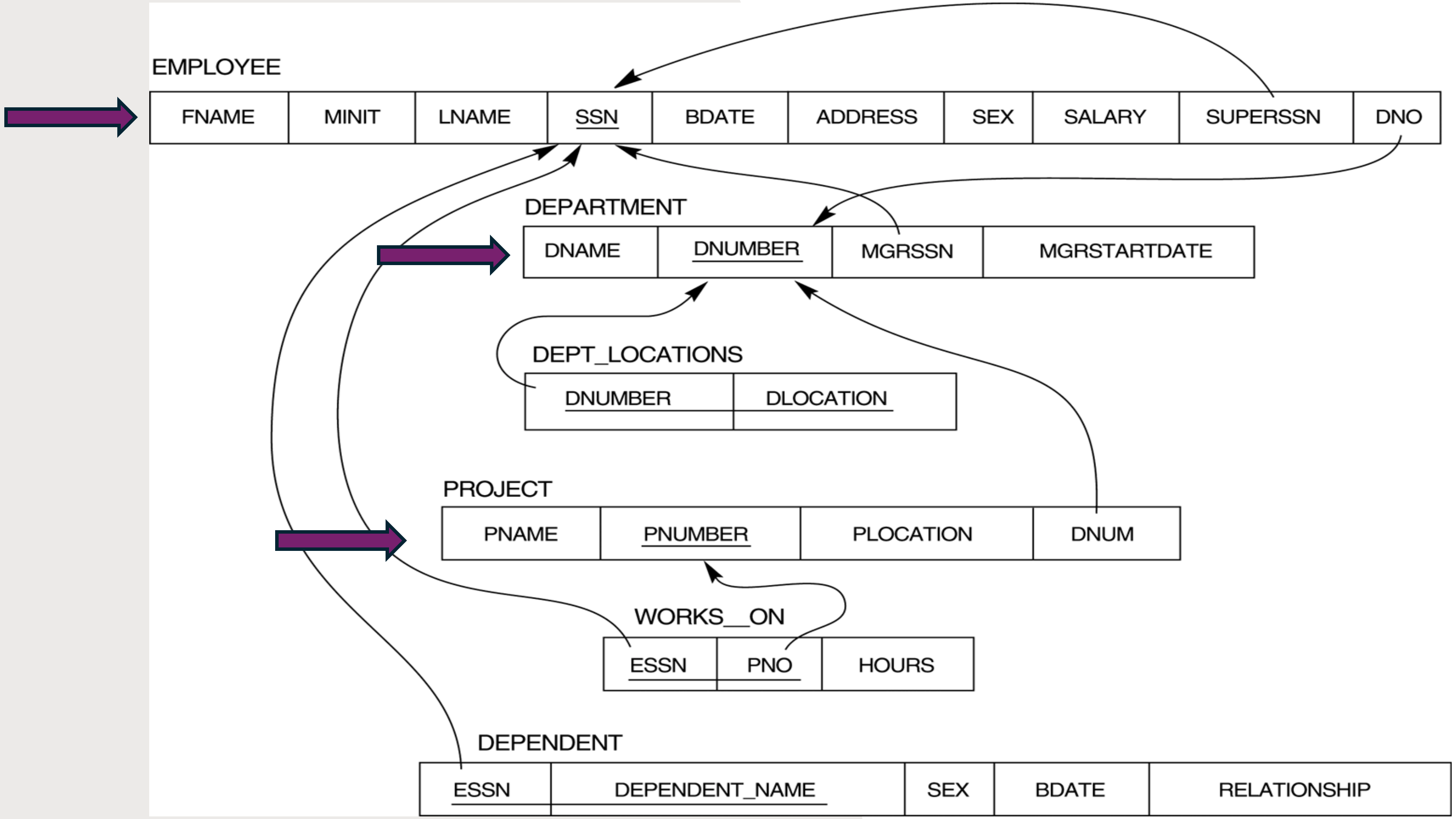
ER-to-Relational Mapping Algorithm

- **Step 1: Mapping of Regular Entity Types.**
  - For each regular (strong) entity type E in the ER schema, create **a relation R that includes all the simple attributes of E**.
  - **Choose one of the key attributes of E as the 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: 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.

The ER conceptual schema diagram for the COMPANY database.



Result of mapping the COMPANY ER schema into a relational schema.



ER-to-Relational Mapping Algorithm (cont.)

- **Step 2: Mapping of Weak Entity Types**
  - 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).
  - 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

ER-to-Relational Mapping Algorithm (cont.)

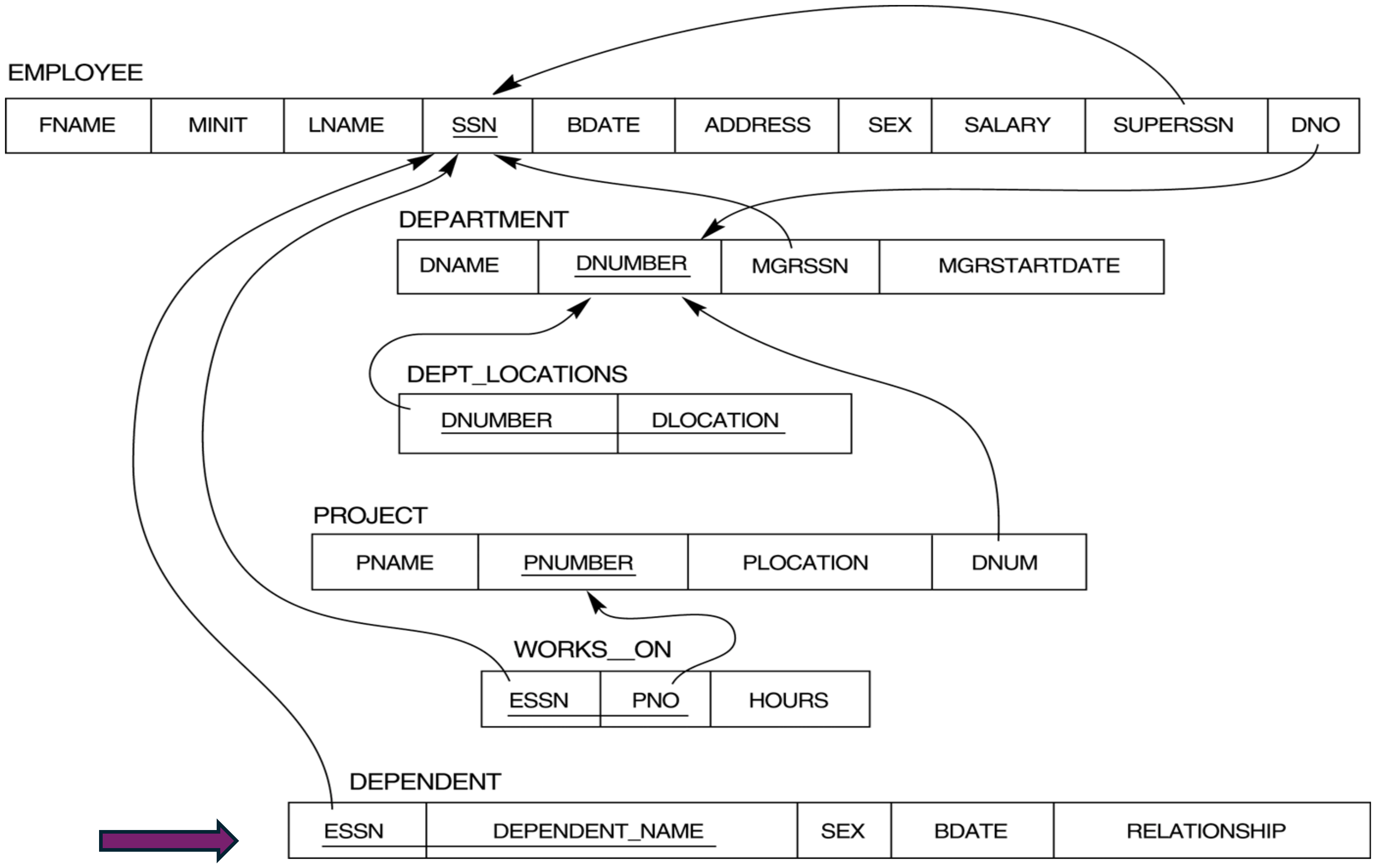
Example:

- Create the relation **DEPENDENT** corresponding 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.

Result of mapping the COMPANY ER schema into a relational schema.



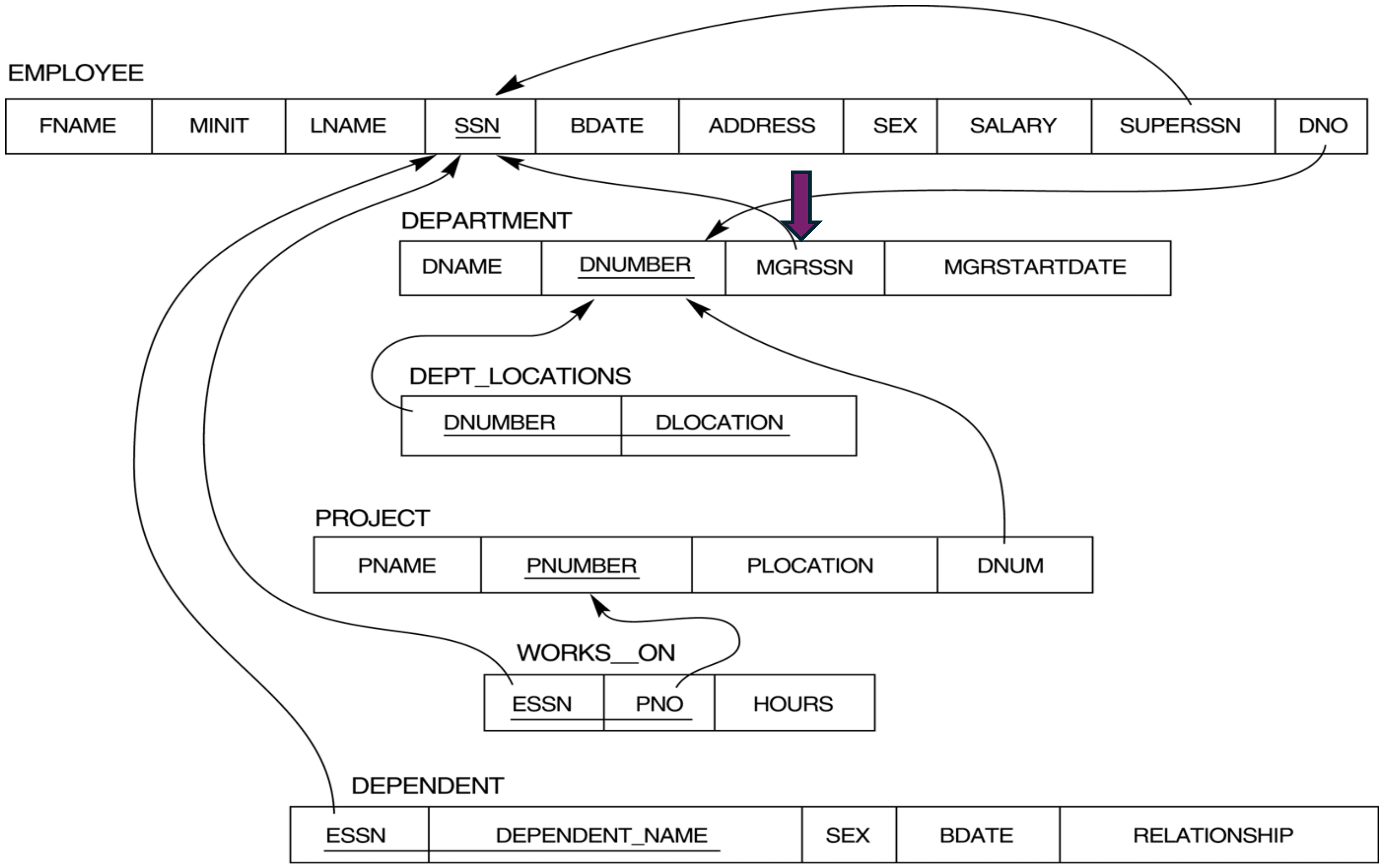
ER-to-Relational Mapping Algorithm (cont.)

- **Step 3: Mapping of Binary 1:1 Relation Types**
  - 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. There are three possible approaches:
    - ***(1) Foreign Key approach:*** Choose one of the relations-S, say-and include a 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.
      - Example: 1:1 relation MANAGES is mapped by choosing the participating entity type DEPARTMENT to serve in the role of S, because its participation in the MANAGES relationship type is total.

ER-to-Relational Mapping Algorithm (cont.)

- ***(2) Merged relation option:*** An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when **both participations are total.**
- ***(3) Cross-reference or relationship relation option:*** The third alternative is to set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

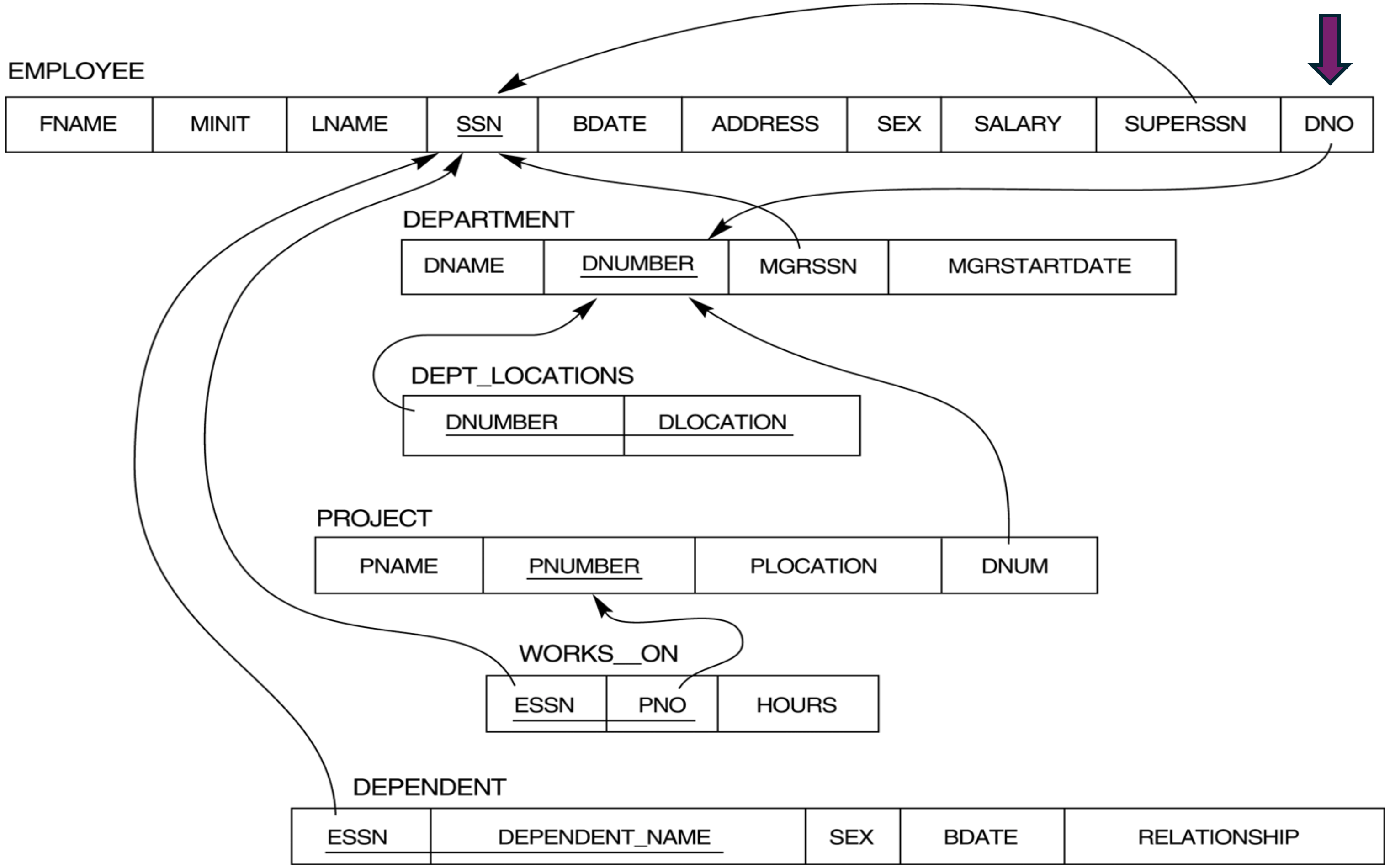
Result of mapping the COMPANY ER schema into a relational schema.



ER-to-Relational Mapping Algorithm (cont.)

- **Step 4: Mapping of Binary 1:N Relationship Types.**
  - 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.
  - Include any simple attributes of the 1:N relation type as attributes of S.
  - 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.

Result of mapping the COMPANY ER schema into a relational schema.



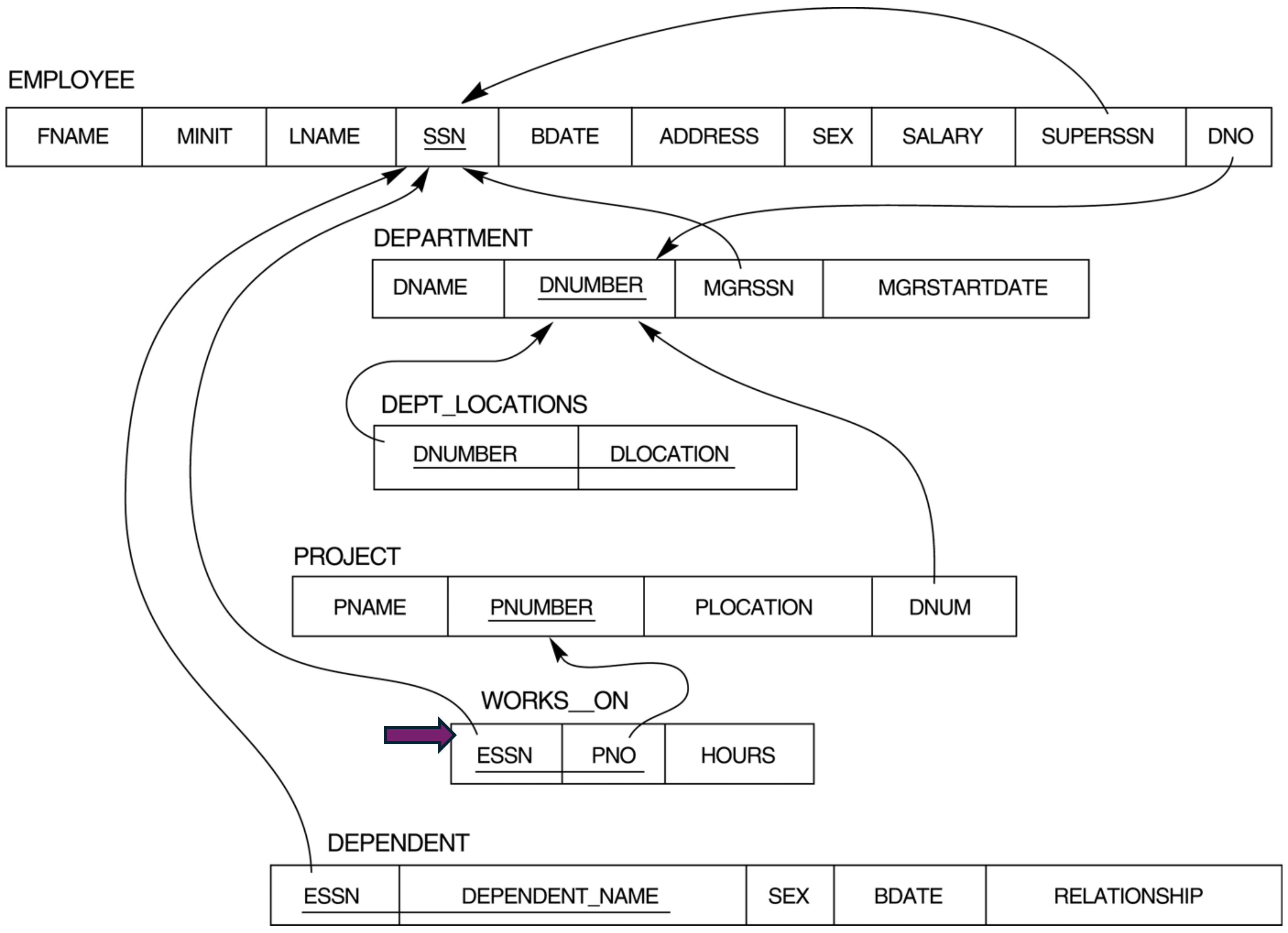
ER-to-Relational Mapping Algorithm (cont.)

- **Step 5: Mapping of Binary M:N Relationship Types.**
  - **For each regular 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.

ER-to-Relational Mapping Algorithm (cont.)

- 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).

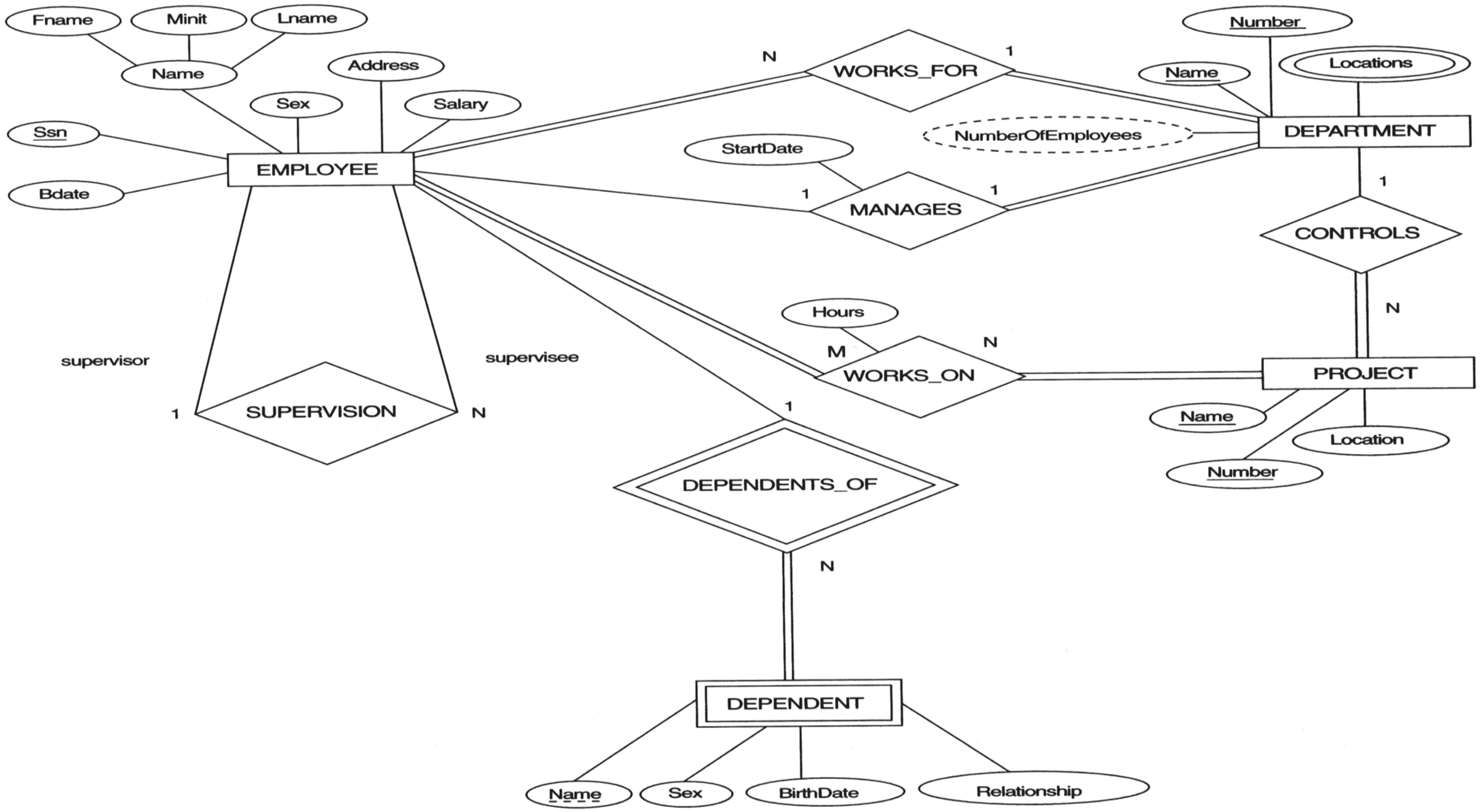
Result of mapping the COMPANY ER schema into a relational schema.



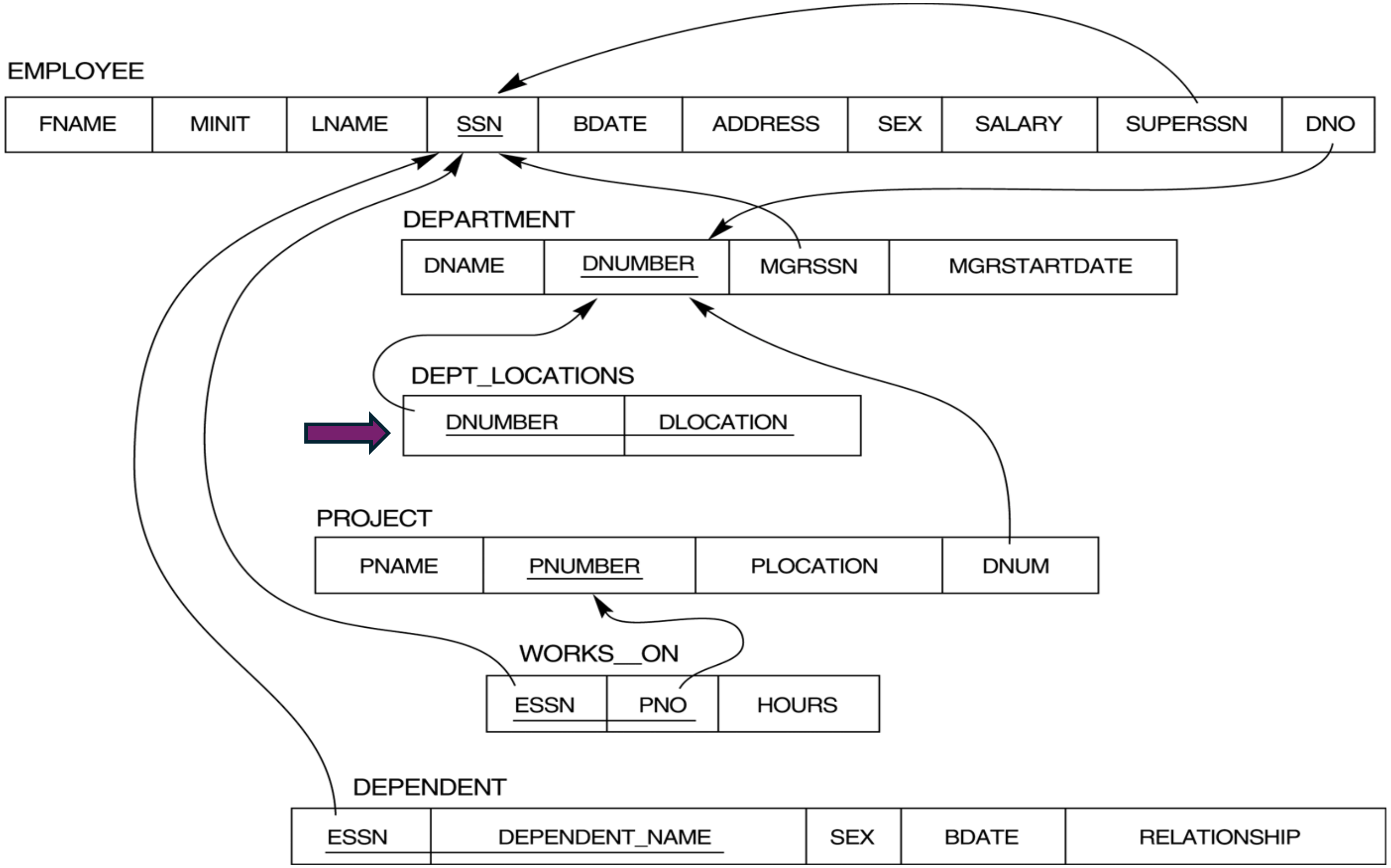
ER-to-Relational Mapping Algorithm (cont.)

- **Step 6: Mapping of Multivalued attributes.**
  - 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 of 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: 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).

The ER conceptual schema diagram for the COMPANY database.



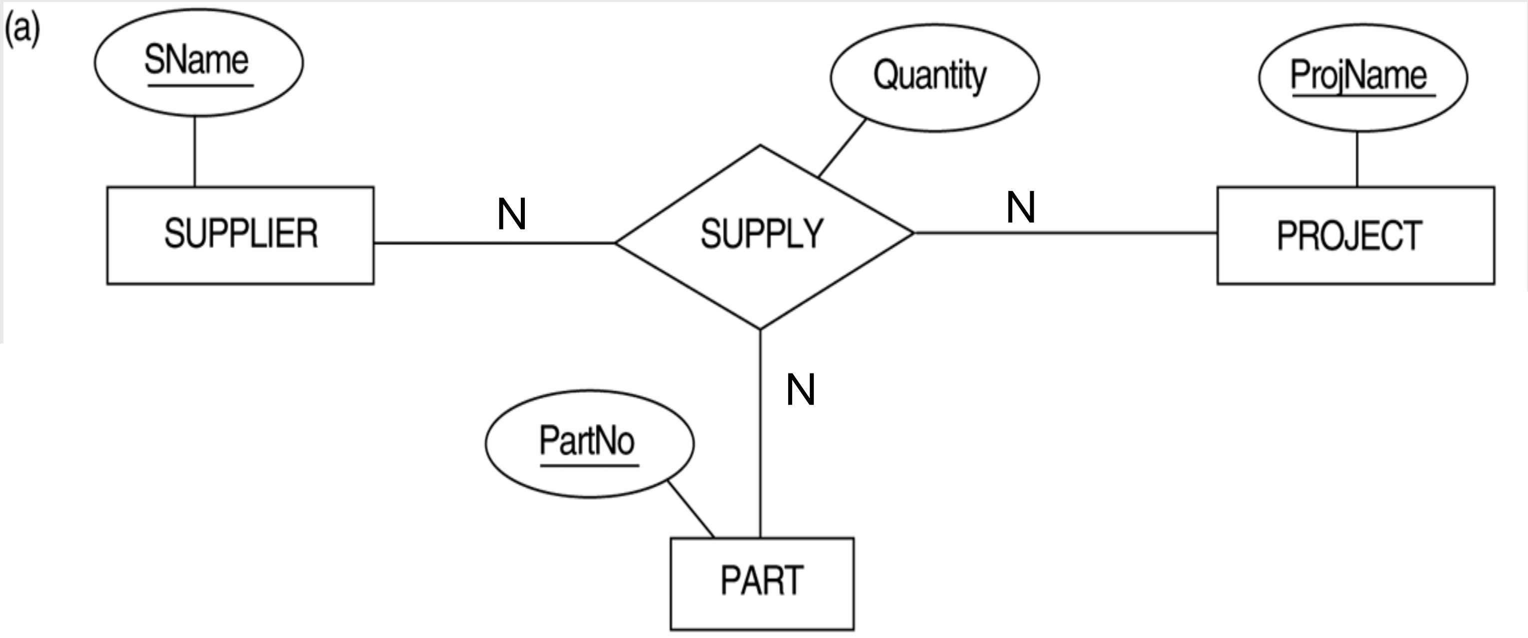
Result of mapping the COMPANY ER schema into a relational schema.



ER-to-Relational Mapping Algorithm (cont.)

- **Step 7: Mapping of N-ary Relationship Types.**
  - For each n-ary relationship type R, where 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.
  - Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.
  - Example: The relationship type SUPPY in the ER below. This can be mapped to the relation SUPPLY shown in the relational schema, whose primary key is the combination of the three foreign keys (SNAME, PARTNO, PROJNAME)

Ternary relationship types.
(a) The SUPPLY relationship.



Mapping the *n*-ary relationship type SUPPLY

SUPPLIER

SNAME	. . .

PROJECT

PROJNAME	. . .

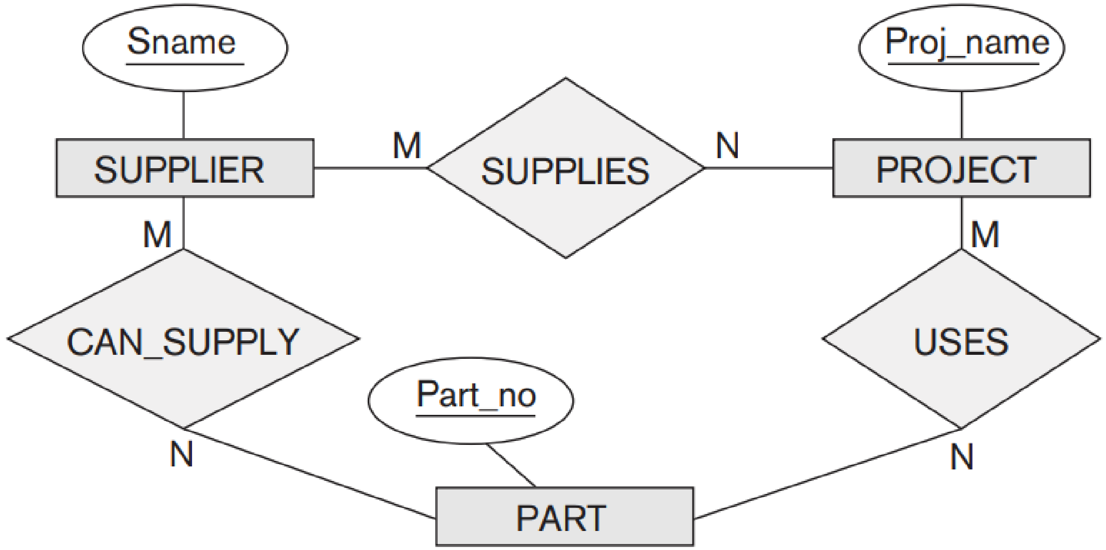
PART

PARTNO	. . .

SUPPLY

SNAME	PROJNAME	PARTNO	QUANTITY

3 Binary relationship types



- SUPPELIER (<u>Sname</u>)
- PROJECT (<u>Proj_name</u>)
- PART (<u>Part_no</u>)
- SUPPLIES (<u>Sname, Proj_name</u>)
- CAN_SUPPLY (<u>Sname, Part_no</u>)
- USES (<u>Proj_name, Part_no</u>)

3 Binary relationship types (cont.)

ตอบได้มั้ยว่า **Project P1** ใช้ยางจาก **Supplier** ใด

SUPPLIER
Sname
S1
S2
S3
S4

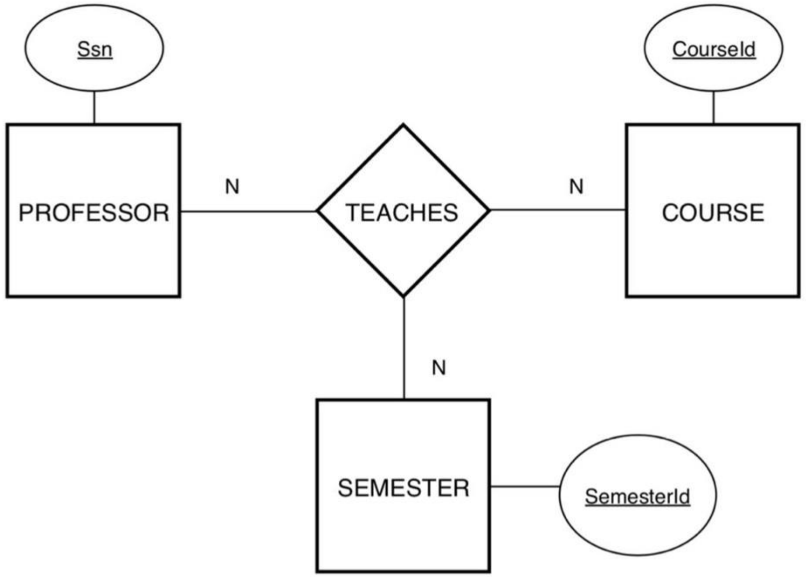
PROJECT
Proj_name
P1
P2
P3

PART
Part_no
ยาง
ล้อ
พวงมาลัย
ท่อไอเสีย

SUPPLIES	
Sname	Proj_name
S1	P1
S2	P1
S1	P2

CAN_SUPPLY	
Sname	Part_no
S1	ยาง
S1	ล้อ
S2	ยาง
S2	พวงมาลัย
S3	ยาง
S3	ท่อไอเสีย
S4	ยาง

USES	
Proj_name	Part_no
P1	ยาง
P1	ล้อ
P2	ยาง
P3	ท่อไอเสีย


Ssn
CourseId
N
N
PROFESSOR
TEACHES
COURSE
N
SEMESTER
SemesterId

Summary of Mapping constructs and constraints

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

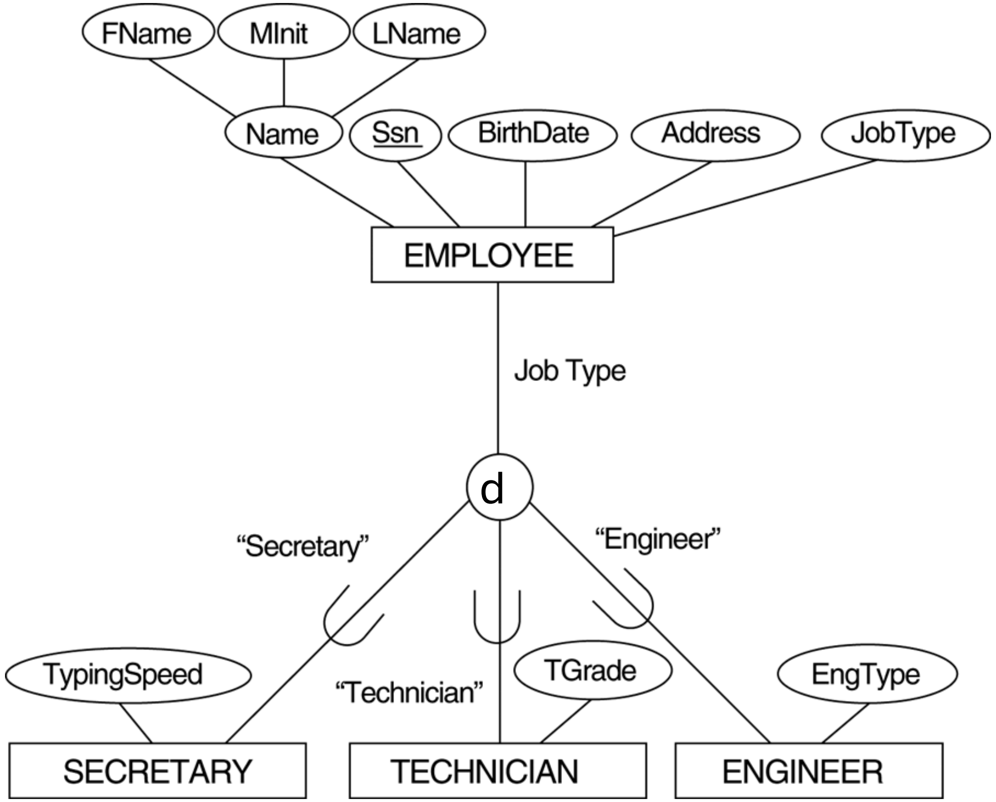
ER-to-Relational Mapping Algorithm (cont.)

- **Step8: Options for Mapping Specialization or Generalization.**
  - Convert each specialization with **m subclasses {S1, S2,….,Sm}** and generalized **superclass C**, **where the attributes of C are {k,a1,…an}** and **k is the (primary) key**, into relational schemas using one of the four following options:

ER-to-Relational Mapping Algorithm (cont.)

- **Option 8A: Multiple relations-Superclass and subclasses.**
  - Create a relation L for C with attributes Attrs(L) = {k,a1,...an} and PK(L) = k. Create a relation Li for each subclass Si, 1 < i < m, with the attributes Attrs(Li) = {k} U {attributes of Si} and PK(Li)=k. This option works for any specialization (total or partial, disjoint of over-lapping).
- **Option 8B: Multiple relations-Subclass relations only**
  - Create a relation Li for each subclass Si, 1 < i < m, with the attributes Attr(Li) = {attributes of Si} U {k,a1...,an} and PK(Li) = k. This option only works for a specialization whose subclasses are total (every entity in the superclass must belong to (at least) one of the subclasses).

EER diagram notation for an attribute-defined specialization on JobType.



Options 8A for Mapping Specialization or Generalization.

(a) EMPLOYEE

<u>SSN</u>	FName	MInit	LName	BirthDate	Address	JobType

SECRETARY

<u>SSN</u>	TypingSpeed

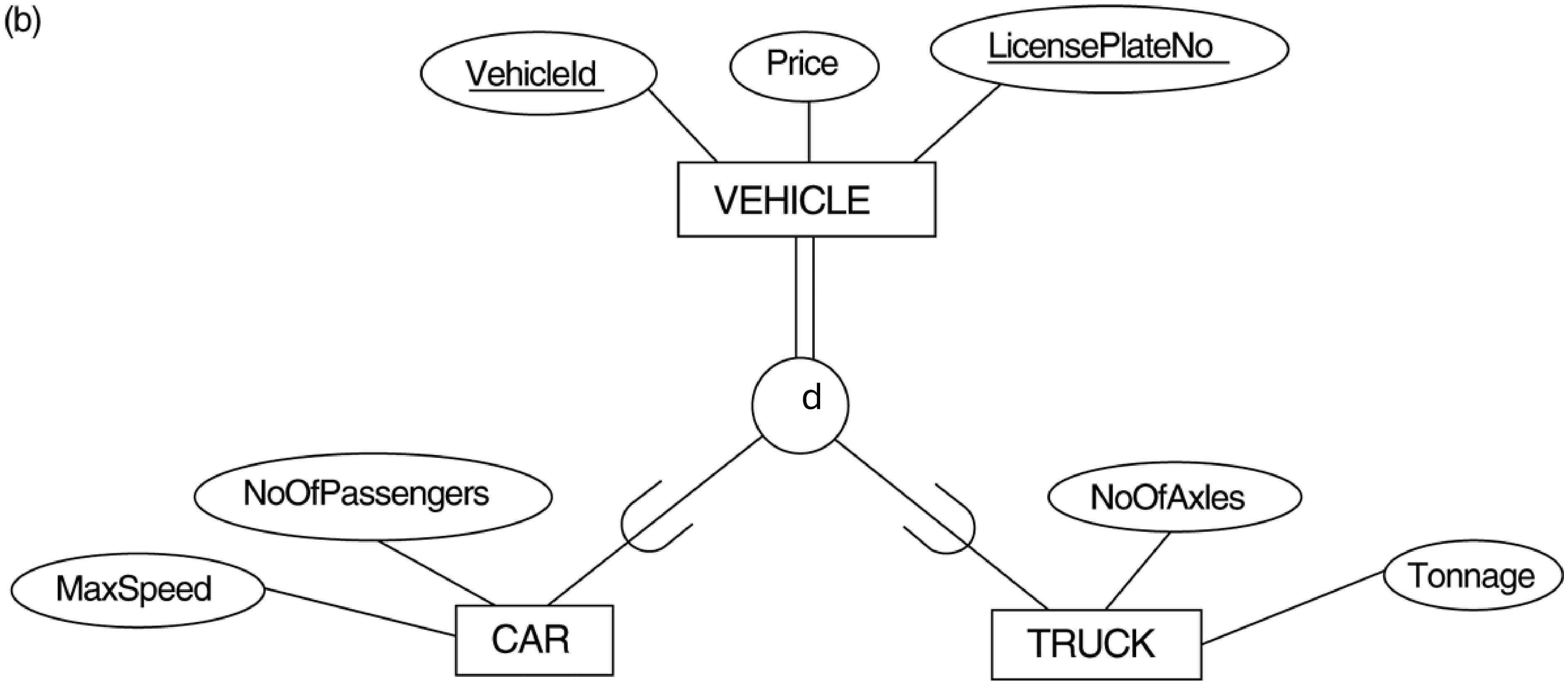
TECHNICIAN

<u>SSN</u>	TGrade

ENGINEER

<u>SSN</u>	EngType

Generalizing CAR and TRUCK into the superclass VEHICLE.



Options 8B for Mapping Specialization or Generalization.

(b) CAR

<u>VehicleId</u>	LicensePlateNo	Price	MaxSpeed	NoOfPassengers

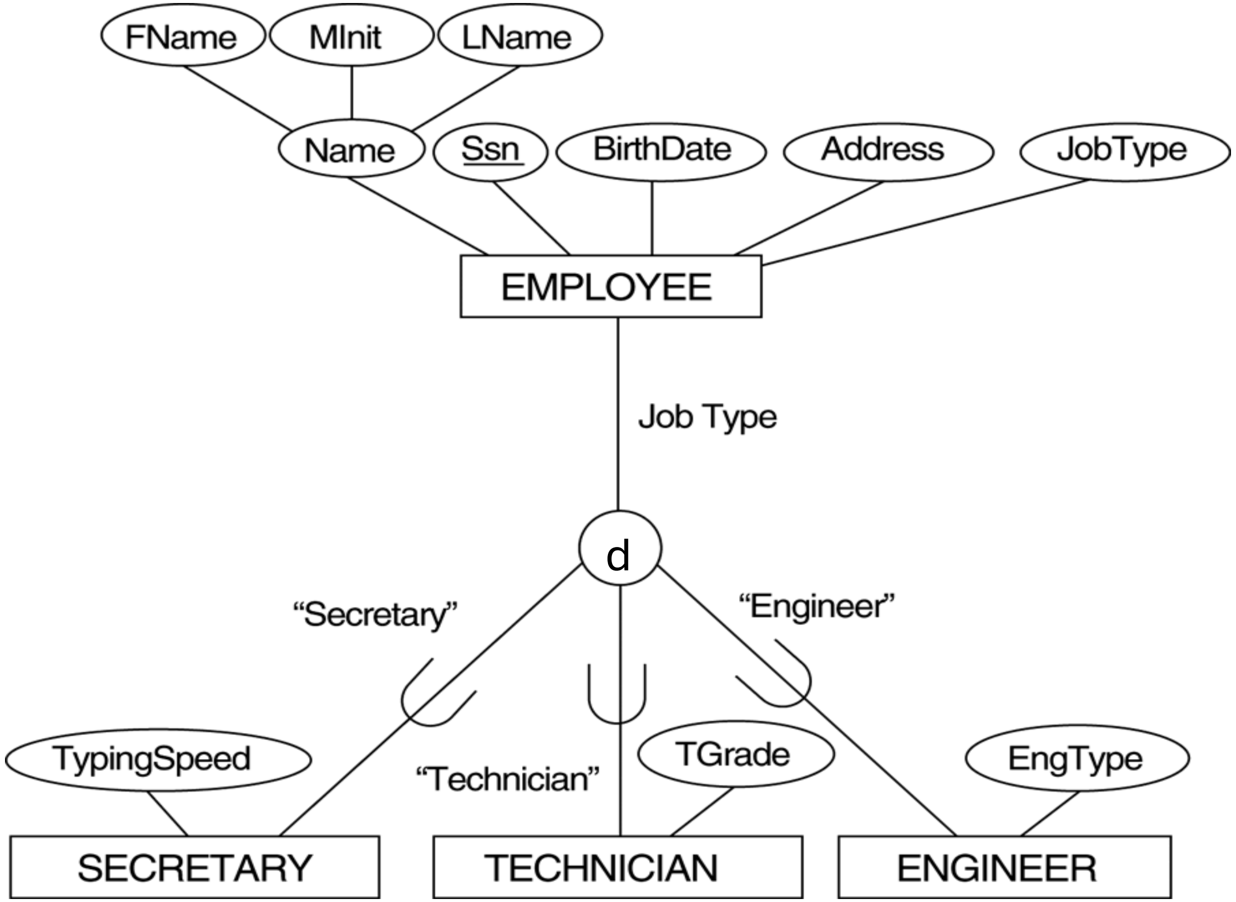
TRUCK

<u>VehicleId</u>	LicensePlateNo	Price	NoOfAxles	**Tonnage**

ER-to-Relational Mapping Algorithm (cont.)

- **Option 8C: Single relation with one type attribute.**
  - Create a single relation L with attributes Attrs(L) = {k,a1,…an} U {attributes of S1} U…U {attributes of Sm} U {t} and PK(L) = k. The attribute t is called a type (or discriminating) attribute that indicates the subclass to which each tuple belongs
- **Option 8D: Single relation with multiple type attributes.**
  - Create a single relation schema L with attributes Attrs(L) = {k,a1,…an} U {attributes of S1} U…U {attributes of Sm} U {t1, t2,…,tm} and PK(L) = k. Each ti, 1 < I < m, is a Boolean type attribute indicating whether a tuple belongs to the subclass Si.

EER diagram notation for an attribute-defined specialization on JobType.

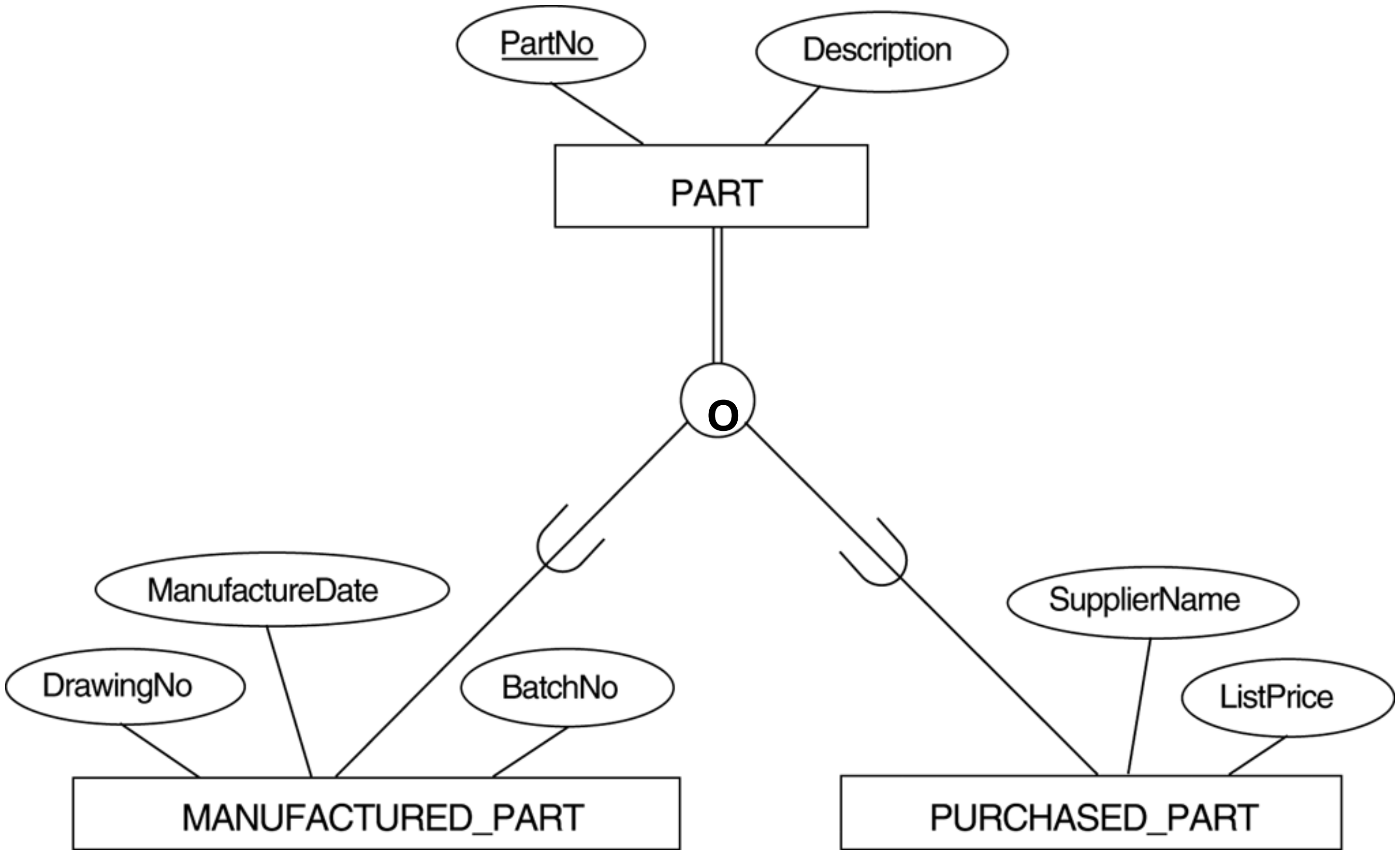


Options 8C for Mapping Specialization or Generalization.

(c) EMPLOYEE

<u>SSN</u>	FName	MInit	LName	BirthDate	Address	JobType	TypingSpeed	TGrade	**EngType**

EER diagram notation for an overlapping (non-disjoint) specialization.



Options 8D with Boolean type fields Mflag and Pflag for Mapping Specialization or Generalization.

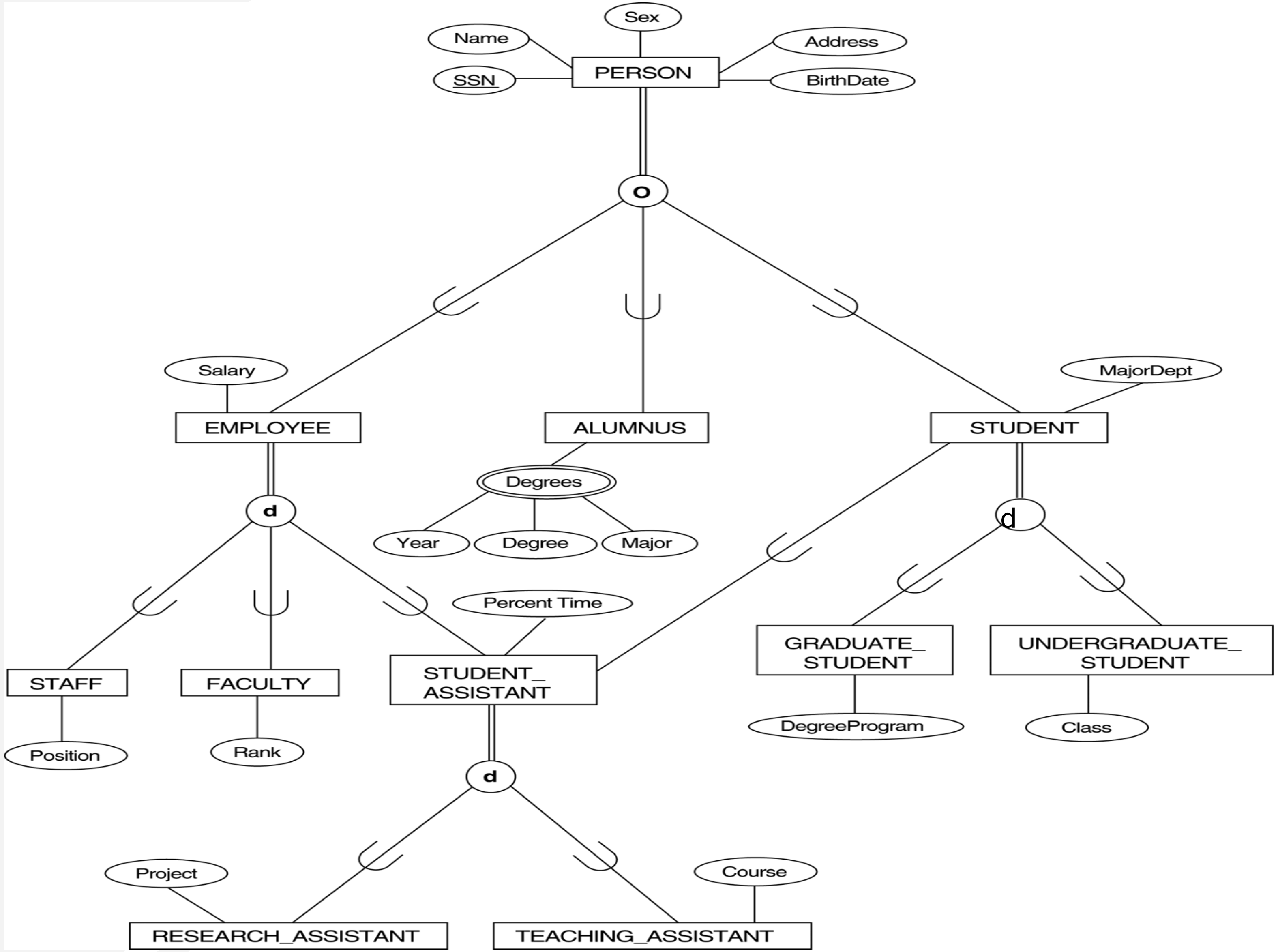
(d) PART

PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

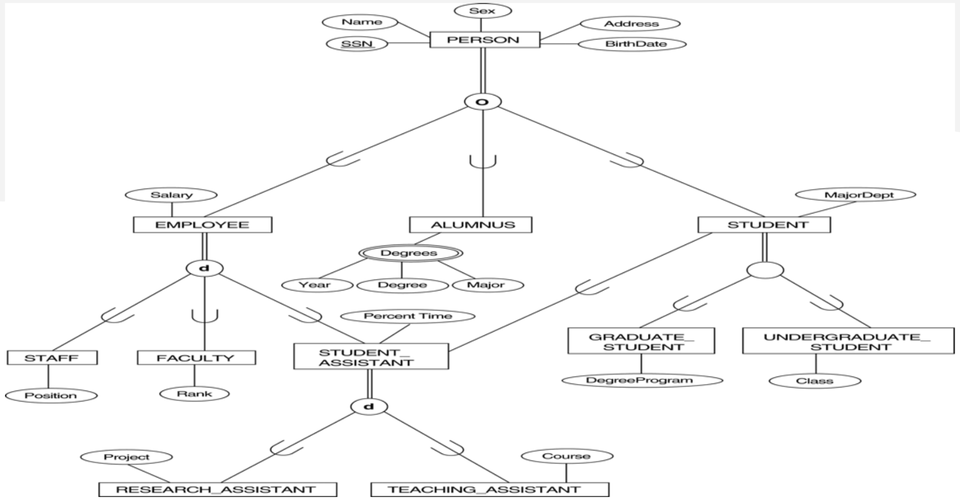
ER-to-Relational Mapping Algorithm (cont.)

- Mapping of Shared Subclasses (Multiple Inheritance)
- A shared subclass, such as STUDENT_ASSISTANT, is a subclass of several classes, indicating multiple inheritance. These classes must all have the same key attribute; otherwise, the shared subclass would be modeled as a category.
- We can apply any of the options discussed in Step 8 to a shared subclass, subject to the restriction discussed in Step 8 of the mapping algorithm. Below both 8C and 8D are used for the shared class STUDENT_ASSISTANT.

A specialization lattice with multiple inheritance for a UNIVERSITY database.



Mapping the EER specialization lattice in using multiple options.



PERSON

SSN	Name	BirthDate	Sex	Address

EMPLOYEE

SSN	Salary	EmployeeType	Position	Rank	PercentTime	RAFlag	TAFlag	Project	Course

ALUMNUS

SSN

ALUMNUS_DEGREES

SSN	Year	Degree	Major

STUDENT

SSN	MajorDept	GradFlag	UndergradFlag	DegreeProgram	Class	StudAssistFlag

ER-to-Relational Mapping Algorithm (cont.)

- **Step 9: Mapping of Union Types (Categories).**
  - For mapping a category whose defining superclass have different keys, it is customary to specify a new key attribute, called **a surrogate key**, when creating a relation to correspond to the category.
  - In the example below we can create a relation OWNER to correspond to the OWNER category and include any attributes of the category in this relation. The primary key of the OWNER relation is the surrogate key, which we called OwnerId.

Two categories (union types): OWNER and REGISTERED_VEHICLE.

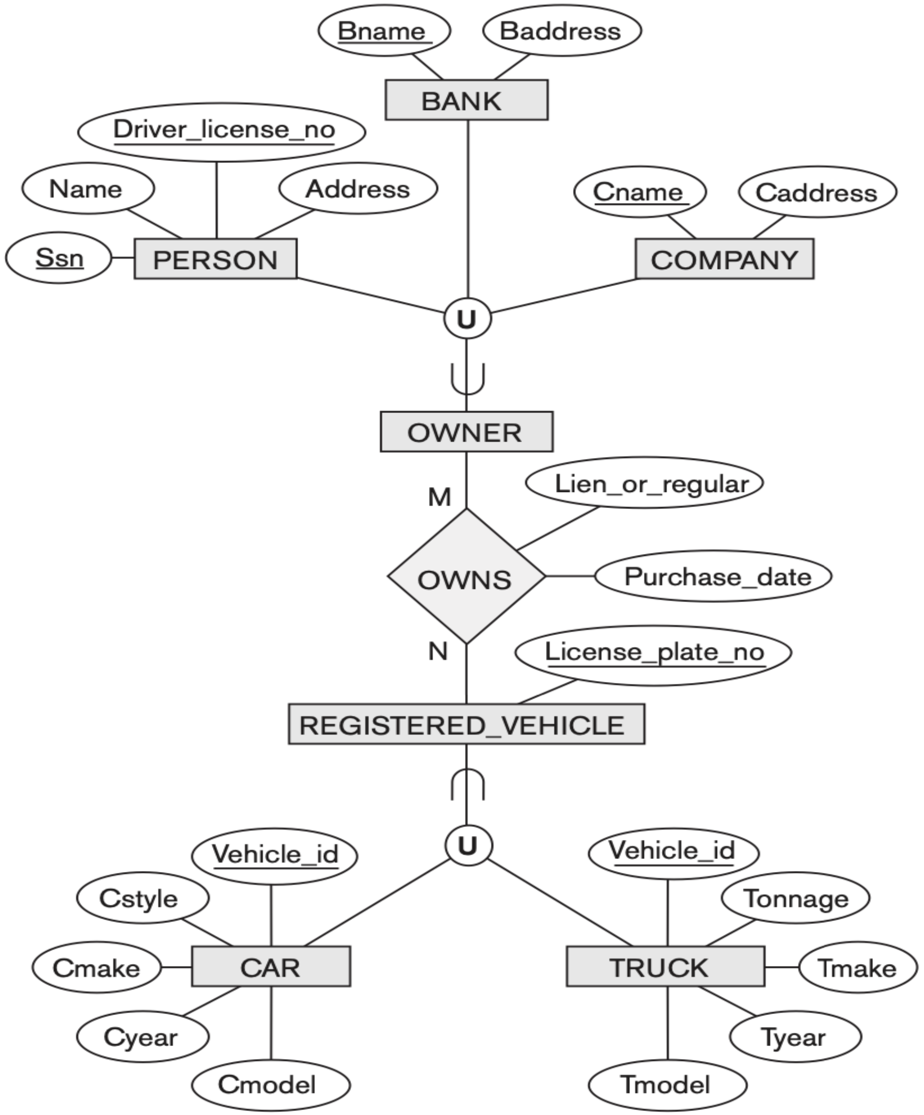


Figure 4.8
Two categories (union types): OWNER and REGISTERED_VEHICLE.

Mapping the EER categories (union types).

PERSON

SSN	DriverLicenseNo	Name	Address	OwnerID

BANK

BName	BAddress	OwnerId

COMPANY

CName	CAddress	OwnerId

OWNER

OwnerId

REGISTERED_VEHICLE

VehicleId	LicensePlateNumber

CAR

VehicleId	CStyle	CMake	CModel	

TRUCK

VehicleId	TMake	TModel	Tonnage	TYear

OWNS

OwnerId	VehicleId	PurchaseDate	LienOrRegular