Relational Data Model and **ER/EER-to-Relational Mapping**

Chapter 3

Contents

Contents

Relational Data Model

- Basic Concepts: relational data model, relation schema, domain, tuple, cardinality & degree, database schema, etc.
- Relational Integrity Constraints
  - key, primary key & foreign key
  - entity integrity constraint
  - referential integrity
- Update Operations on Relations

Basic Concepts

- 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
- The 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

Basic Concepts

- **Relational data model:** represents a database in the form of **relations** - 2-dimensional table with rows and columns of data. A database may contain one or more such tables. A relation schema is used to describe a relation
- **Relation schema:** R(A1, A2,..., An) is made up of a relation name R and a list of **attributes** A1, A2, . . ., An. Each attribute Ai is the name of a role played by some domain D in the relation schema R. R is called the **name** of this relation

Basic Concepts

- The **degree of a relation** is the number of attributes n of its relation schema.
- **Domain D**: D is called the domain of Ai and is denoted by dom(Ai). It is a set of atomic values and a set of integrity constraints
  - STUDENT(Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)
  - Degree = ??
  - dom(GPA) = ??

Basic Concepts

- **Tuple**: row/record in table
- **Cardinality**: number of tuples in a table
- **Database schema** S = {R1, R2,…, Rm}

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Schema diagram for the COMPANY relational database schema

Basic Concepts

- A **relation r** (or **relation state, relation instance**) of the relation schema R(A1, A2, . . ., An), also denoted by r(R), is a set of **n-tuples** r = {t1, t2, . . ., tm}.
  - Each n-tuple t is an ordered list of n values t = <v1, v2, . . ., vn>, where each value vi, i=1..n, is an element of dom(Ai) or is a special **null** value. The i^{th} value in tuple t, which corresponds to the attribute Ai, is referred to as t[Ai]

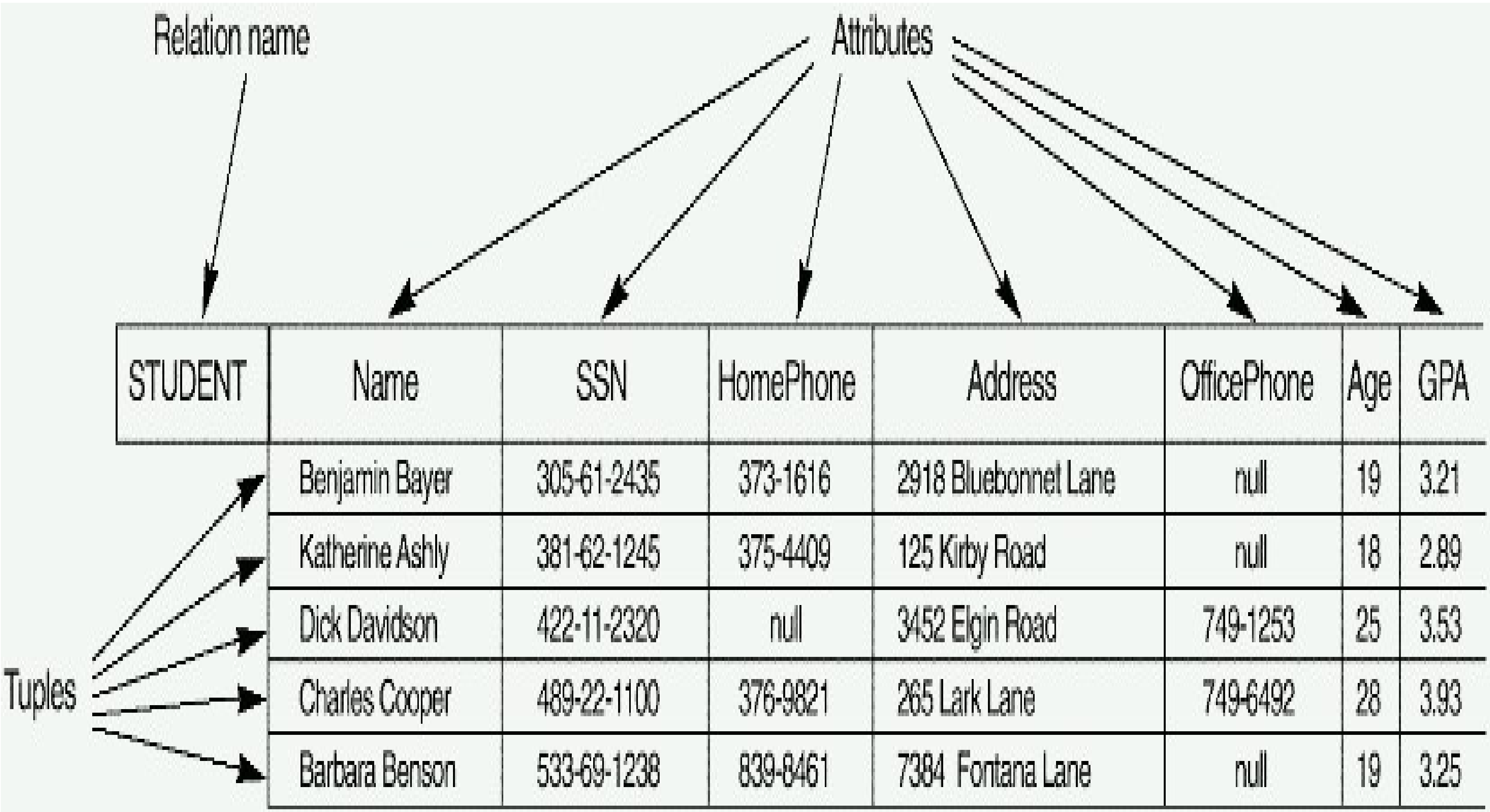
Basic Concepts

Relational data model
Database schema
Relation schema
Relation
Tuple
Attribute

Basic Concepts

- A relation can be conveniently represented by a table, as the example shows
- The columns of the tabular relation represent attributes
- Each attribute has a distinct name, and is always referenced by that name, never by its position
- Each row of the table represents a tuple. The ordering of the tuples is immaterial and all tuples must be distinct

Basic Concepts



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

Alternative Terminology for Relational Model

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

Relational Integrity Constraints

- Constraints are **conditions** that must hold on **all** valid relation instances. There are three main types of constraints:
  - Key constraints
  - Entity integrity constraints
  - Referential integrity constraints
- But ...

Relational Integrity Constraints

- **Null** value
  - Represents value for an attribute that is currently unknown or inapplicable for tuple
  - Deals with incomplete or exceptional data
  - Represents the absence of a value and is not the same as zero or spaces, which are values

Relational Integrity Constraints - Key Constraints

- **Superkey** of R: A set of attributes SK of R such that no two tuples in any valid relation instance r(R) will have the same value for SK. That is, for any distinct tuples t1 and t2 in r(R), t1[SK] $\neq$ t2[SK]
- **Key** of R: A "minimal" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey

Relational Integrity Constraints - Key Constraints

Example: The CAR relation schema:

CAR(State, Reg#, SerialNo, Make, Model, Year) has two keys

Key1 = {State, Reg#}

Key2 = {SerialNo}, which are also superkeys. {SerialNo, Make} is a superkey but not a key

- If a relation has several **candidate** keys, one is chosen arbitrarily to be the **primary** key. The primary key attributes are <u>**underlined**</u>**.**
- **Secondary key**

Relational Integrity Constraints - Key Constraints

- The CAR relation, with two candidate keys: License_Number and Engine_Serial_Number

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

Relational Integrity Constraints - 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 = $\{R_1, R_2, ..., R_n\}$
- **Entity Integrity**: primary key attributes PK of each relation schema R in S cannot have null values in any tuple of r(R) because primary key values are used to identify the individual tuples: $t[PK] \neq null$ 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

Relational Integrity Constraints - Referential Integrity

- A constraint involving two relations (the previous constraints involve a single relation)
- Used to specify a relationship among tuples in two relations: the referencing relation and the referenced relation
- Tuples in the referencing relation R_1 have attributes FK (called foreign key attributes) that reference the primary key attributes PK of the referenced relation R_2. A tuple t_1 in R1 is said to reference a tuple t_2 in R_2 if $t_1[FK] = t_2[PK]$
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R_1.FK to R_2

Relational Integrity Constraints - Referential Integrity

DEPARTMENT

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

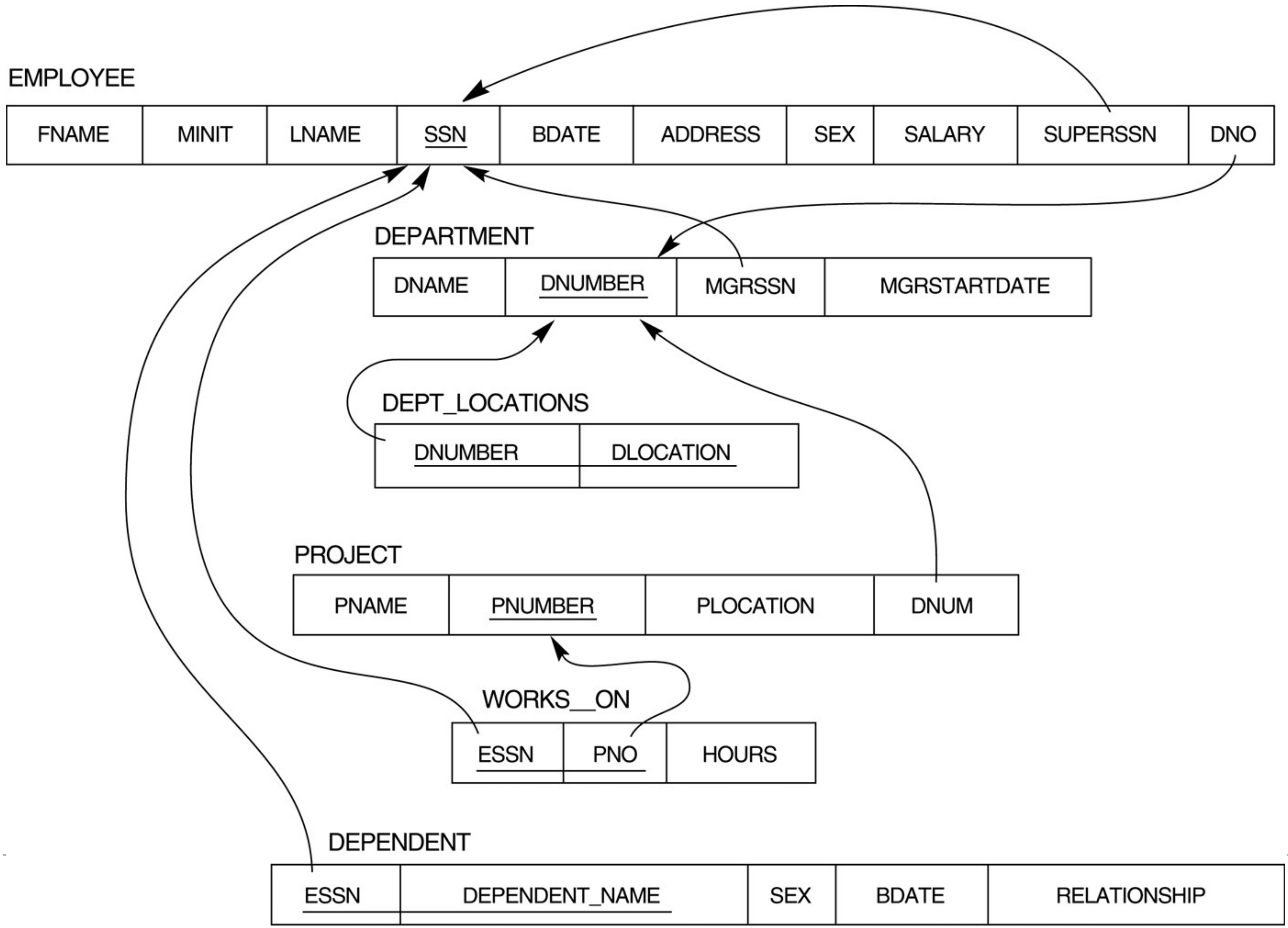
EMPLOYEE

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

Relational Integrity Constraints - Referential Integrity

- The value in the foreign key column (or columns) FK of the referencing relation R_1 can be either:
  - (1) a value of an existing primary key value of the corresponding primary key PK in the referenced relation R_2, or
  - (2) a NULL
- In case (2), the FK in R_1 should not be a part of its own primary key

Referential integrity constraints displayed on the COMPANY relational database schema



Relational Integrity Constraints - Other Types of Constraints

- **Semantic Integrity Constraints**:
  - based on application semantics and cannot be expressed by the model per se
  - E.g., “the max. no. of hours per employee for all projects he or she works on is 56 hrs per week”
  - A constraint specification language may have to be used to express these
  - SQL-99 allows triggers and ASSERTIONS to allow for some of these
- State/static constraints (so far)
- Transition/dynamic constraints: e.g., “the salary of an employee can only increase”

Update Operations on Relations

- INSERT a tuple
- DELETE a tuple
- MODIFY a tuple

- Integrity constraints should not be violated by the update operations

Update Operations on Relations

- **Insertion**: to insert a new tuple t into a relation R. When inserting a new tuple, it should make sure that the database constraints are not violated:
  - The value of an attribute should be of the correct data type (i.e. from the appropriate domain).
  - The value of a prime attribute (i.e. the key attribute) must not be null
  - The key value(s) must not be the same as that of an existing tuple in the same relation
  - The value of a foreign key (if any) must refer to an existing tuple in the corresponding relation
- Options if the constraints are violated: *??? !!*

Update Operations on Relations

- **Deletion**: to remove an existing tuple t from a relation R. When deleting a tuple, the following constraints must not be violated:
  - The tuple must already exist in the database
  - The referential integrity constraint is not violated
- **Modification**: to change values of some attributes of an existing tuple t in a relation R

Update Operations on Relations

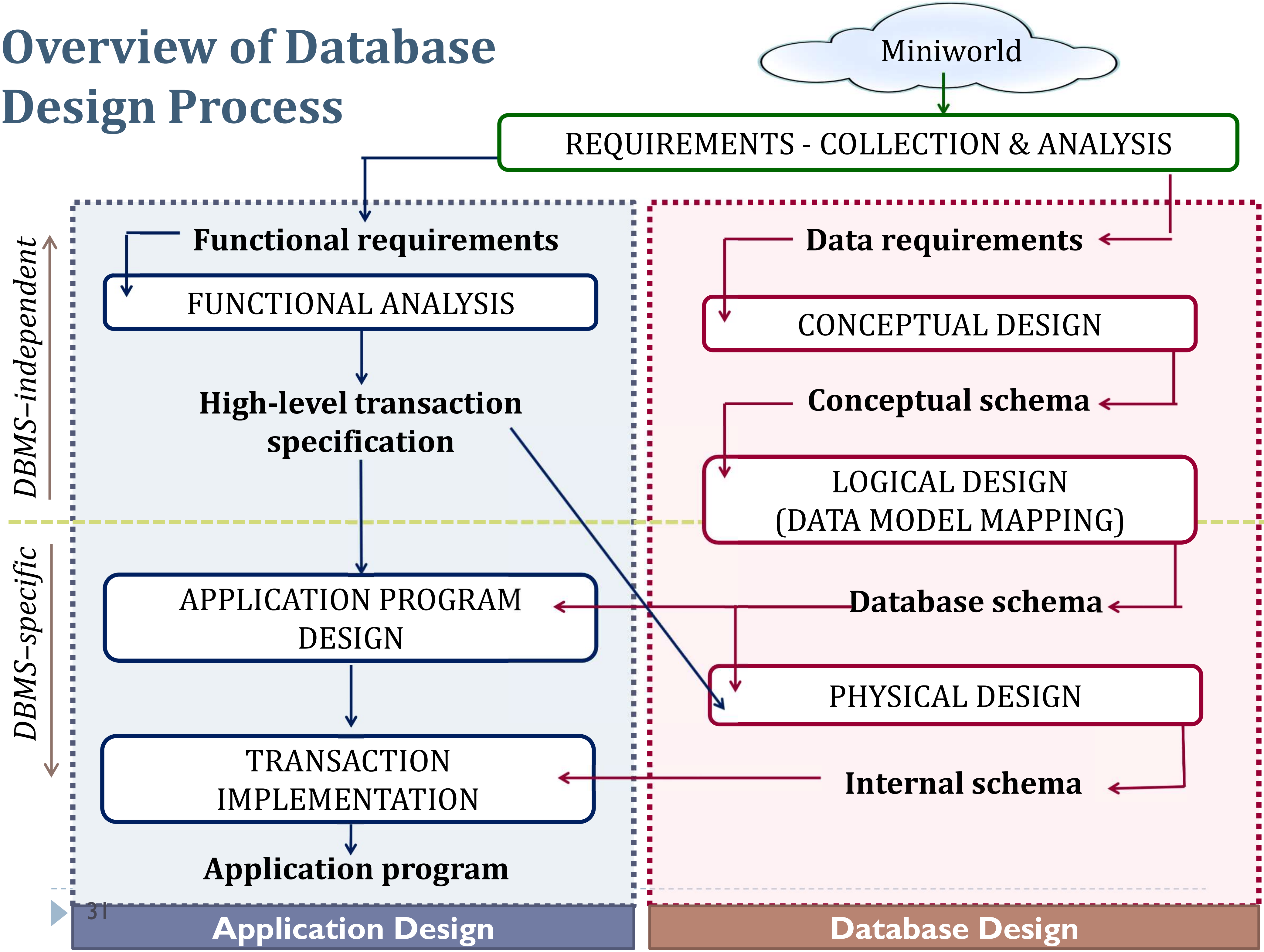
- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine

Contents

Main Phases of Database Design

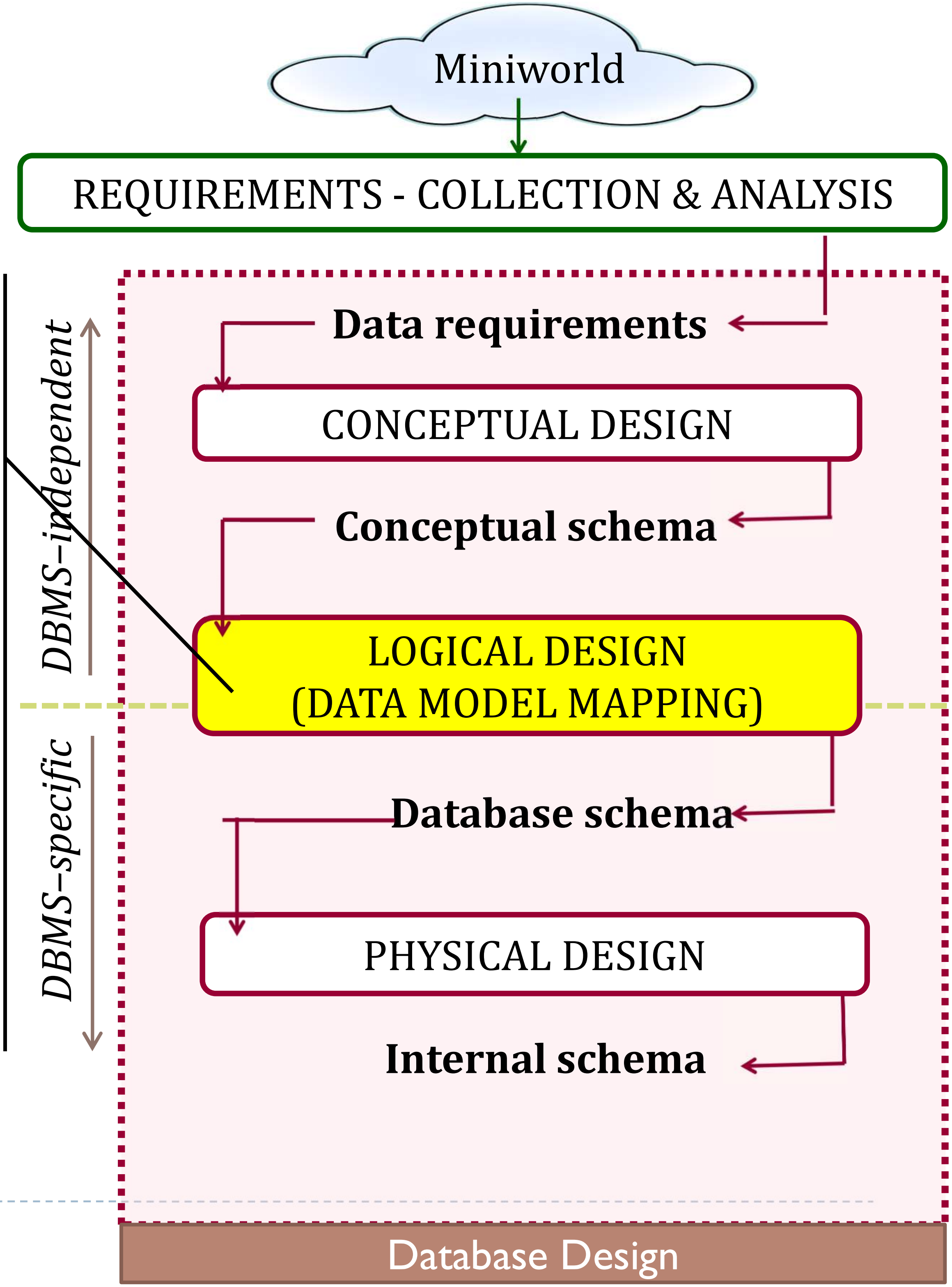
- Three main phases
  - Conceptual database design
  - Logical database design
  - Physical database design

Overview of Database Design Process



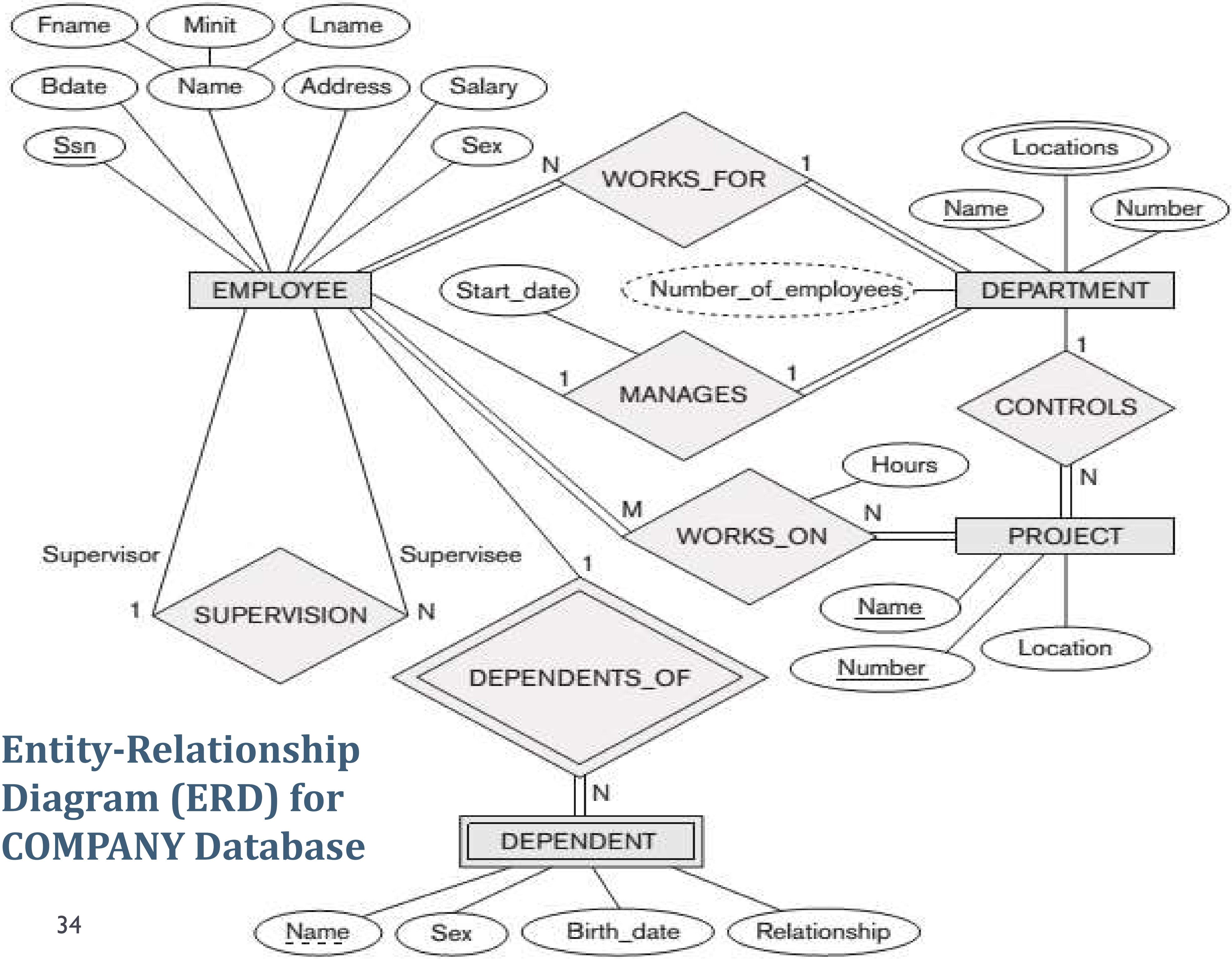
Overview of Database Design Process

- Create a **database schema** in implementation data model of a commercial DBMS
- **Data model mapping** is often automated or semi-automated within the database design tool.



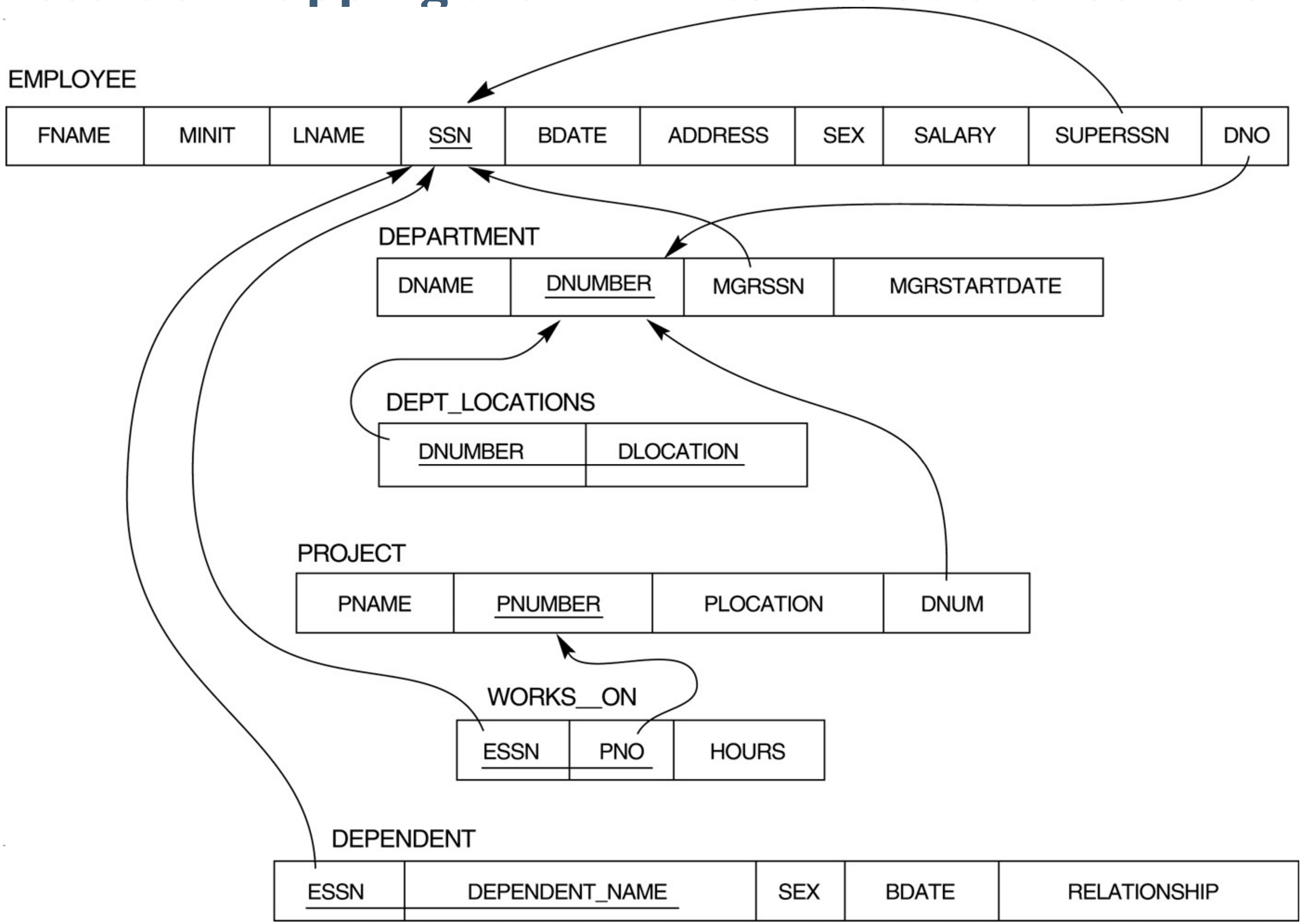
Main Phases of Database Design

- Logical database design
  - The process of constructing a model of the data used in an enterprise based on a specific data model (e.g. relational), but independent of a particular DBMS and other physical considerations
  - ER- & EER-to-Relational Mapping
  - *Normalization*



Entity-Relationship Diagram (ERD) for COMPANY Database

Result of mapping the ERD into a relational schema



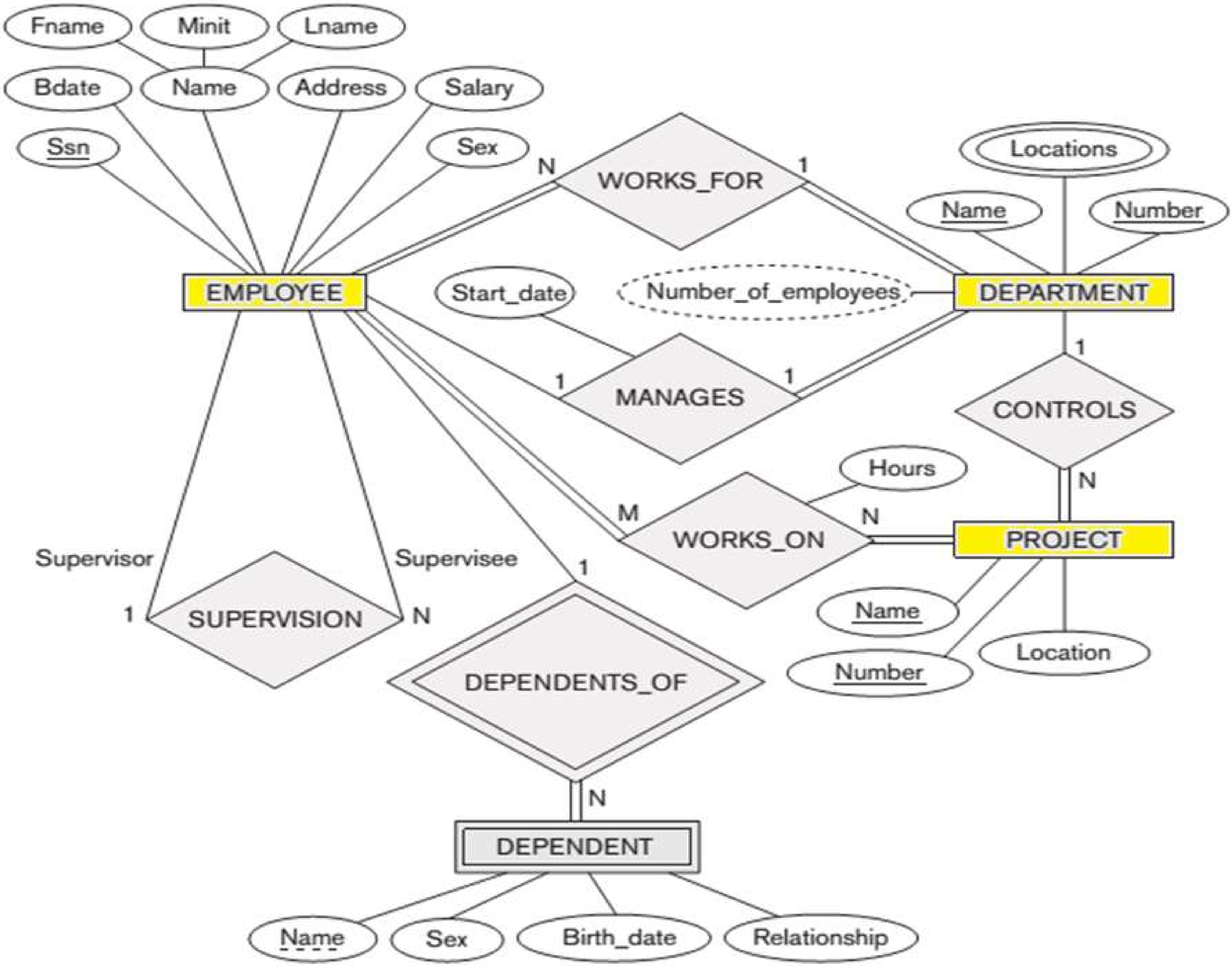
Contents

ER- & EER-to-Relational Mapping

- ER-
  - Step 1: Mapping of Regular Entity Types
  - Step 2: Mapping of Weak Entity Types
  - Step 3: Mapping of Binary 1:1 Relationship 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
- EER-
  - Step 8: Options for Mapping Specialization or Generalization.
  - Step 9: Mapping of Union Types (Categories)

ER-to-Relational Mapping

- **Step 1: Mapping of Regular (strong) Entity Types**
  - Entity --> Relation
  - Attribute of entity --> Attribute of relation
  - Primary key of entity --> Primary key of relation
  - **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



Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Step 1: Mapping of Regular (strong) Entity Types

EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY

DEPARTMENT

DNAME	DNUMBER

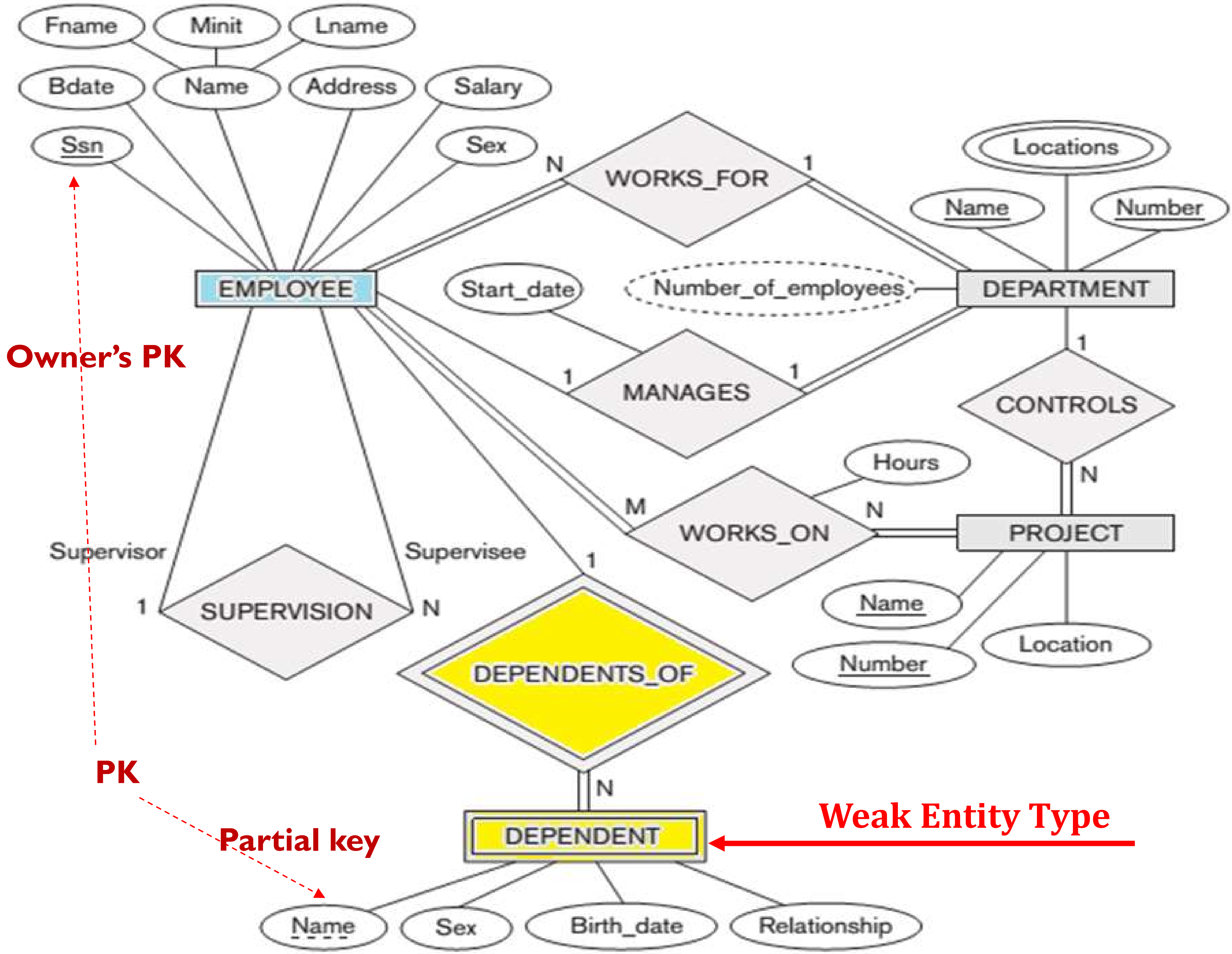
PROJECT

PNAME	PNUMBER	PLOCATION

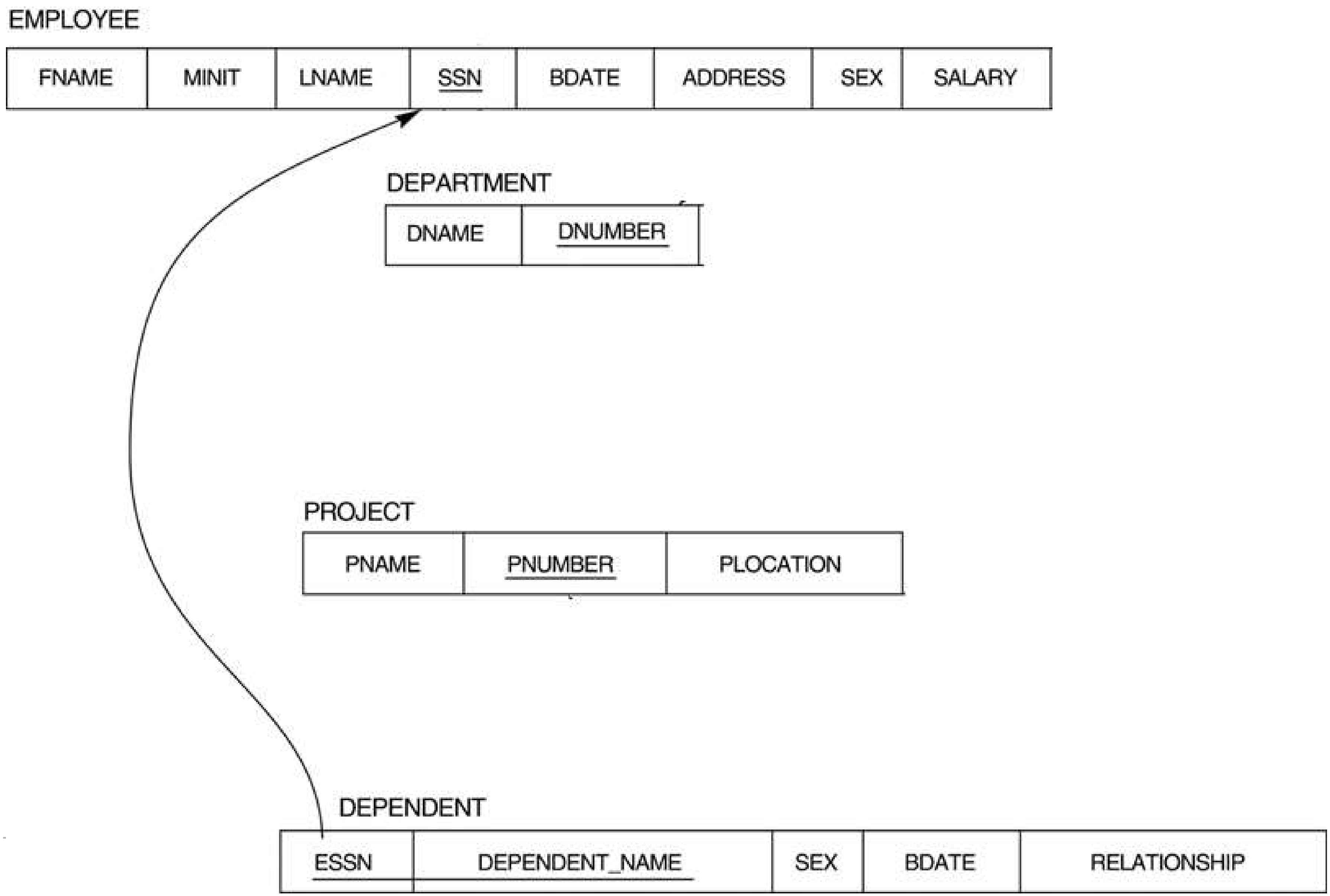
ER-to-Relational Mapping

- **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
  - **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
  - Note: CASCADE option as implemented


Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
Owner's PK
1
MANAGES
1
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
DEPENDENTS_OF
Location
Number
PK
N
Weak Entity Type
Partial key
DEPENDENT
Name
Sex
Birth_date
Relationship

Step 2: Mapping of Weak Entity Types



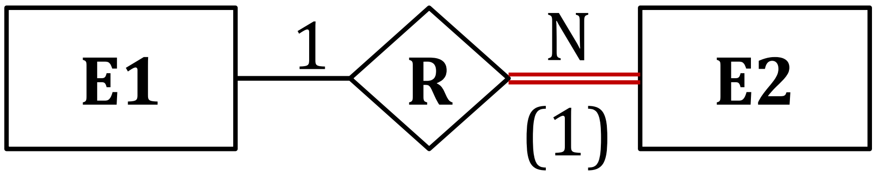
ER-to-Relational Mapping

- ER-
  -
  -
  - Step 3: Mapping of Binary 1:1 Relationship 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
- Transformation of binary relationships - depends on *functionality* of relationship and *membership class* of participating entity types

ER-to-Relational Mapping

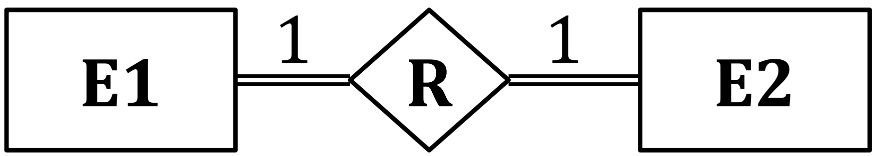
- **Mandatory** membership class
  - For two entity types E1 and E2: If E2 is a mandatory member of an N:1 (or 1:1) relationship with E1, then the relation for E2 will include the prime attributes of E1 as a foreign key to represent the relationship
  - 1:1 relationship: If the membership class for E1 and E2 are both mandatory, a foreign key can be used in either relation
  - N:1 relationship: If the membership class of E2, which is at the N-side of the relationship, is *optional* (i.e. partial), then the above guideline is not applicable

ER-to-Relational Mapping



E1(K1)

E2(K2, K1)



E1(K1)

E2(K2, K1)

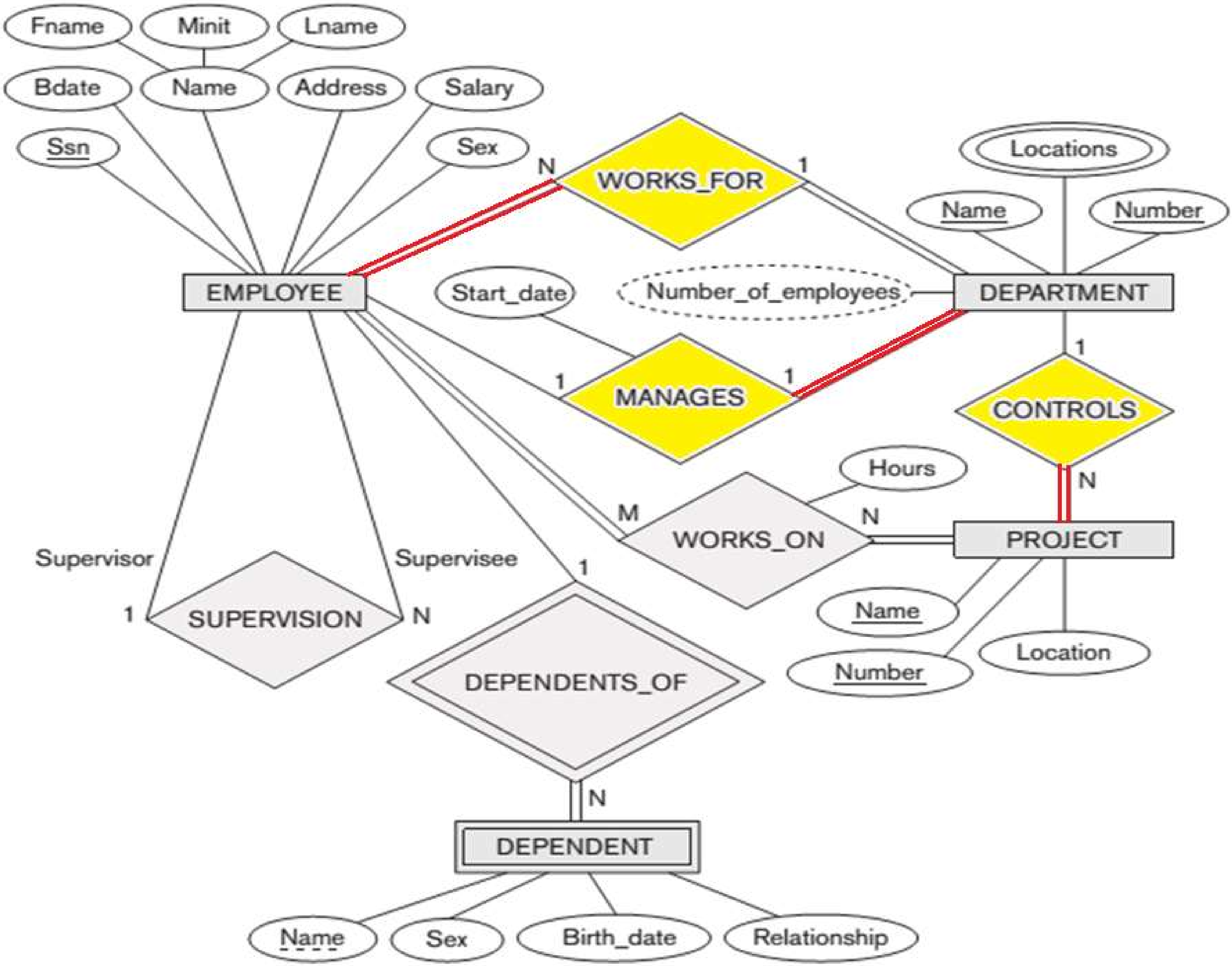
E1(K1, K2)

E2(K2)

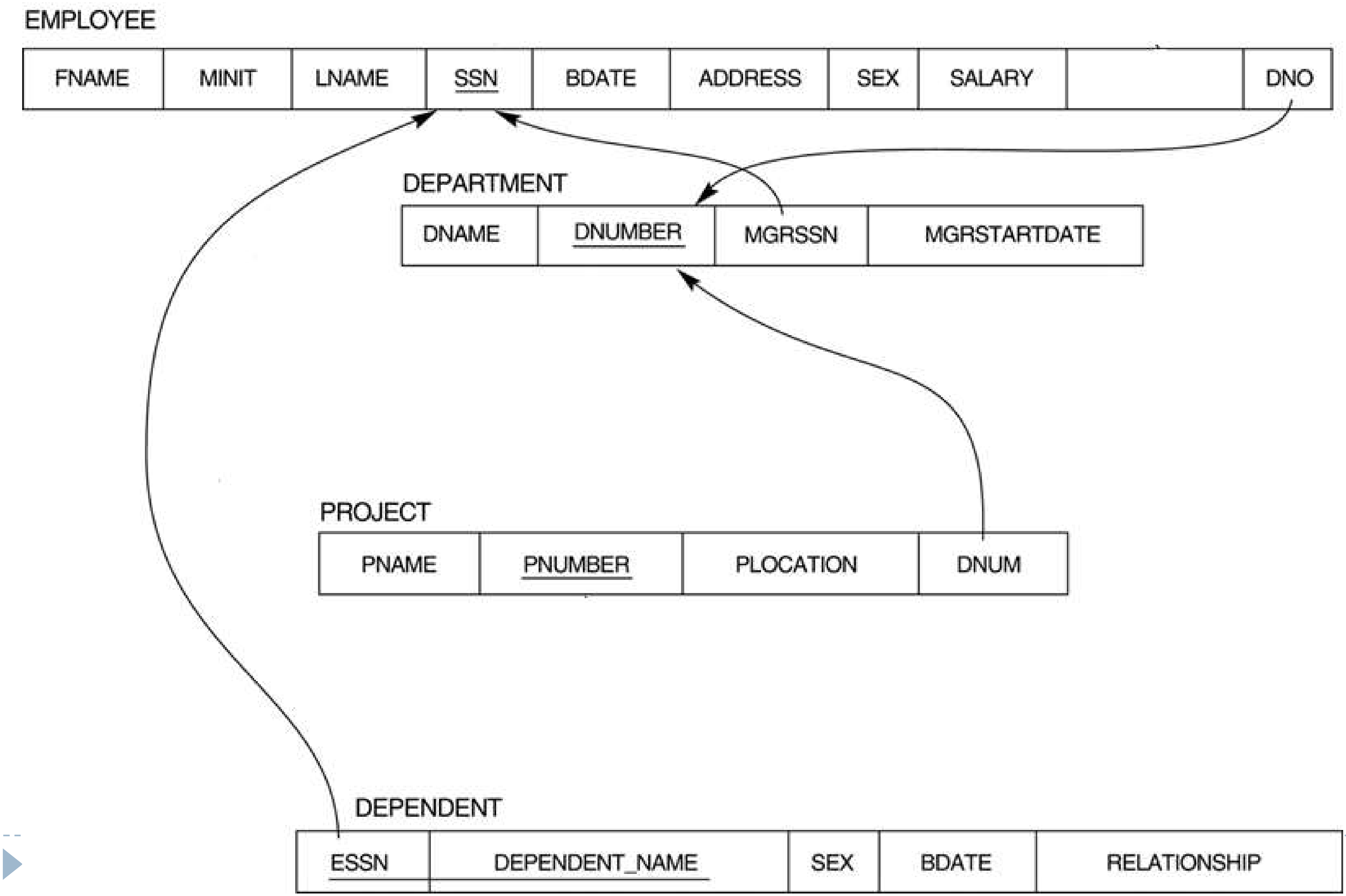
ER-to-Relational Mapping

- Assume every module must be offered by a department, then the entity type MODULE is a **mandatory** member of the relationship OFFER. The relation for MODULE is:

MODULE(<u>MDL-NUMBER</u>, TITLE, TERM, ..., **DNAME**)


Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

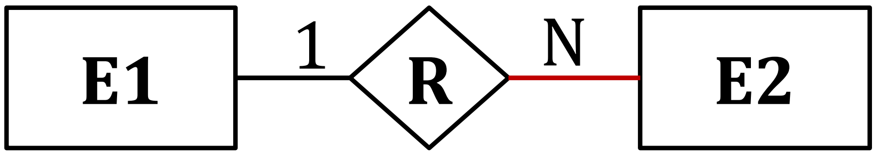
Step 3-4: Mapping of Relationship Types (Mandatory)



ER-to-Relational Mapping

- **Optional** membership classes
  - If entity type E2 is an optional member of the N:1 relationship with entity type E1 (i.e. E2 is at the N-side of the relationship), then the relationship is **usually** represented by a new relation containing the prime attributes of E1 and E2, together with any attributes of the relationship. The key of the entity type at the N-side (i.e. E2) will become the key of the new relation
  - If both entity types in a 1:1 relationship have the optional membership, a new relation is created which contains the prime attributes of both entity types, together with any attributes of the relationship. The prime attribute(s) of either entity type will be the key of the new relation

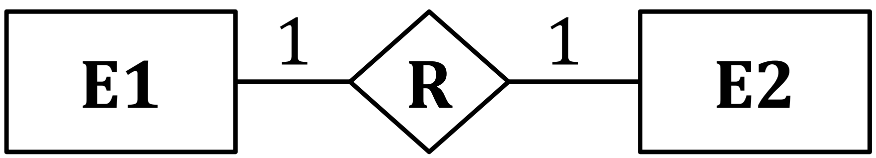
ER-to-Relational Mapping



E1 (K1)
E2 (K2, K1)

E1 (K1)
E2 (K2)
R (K2, K1)

Almost optional
1000Emp -> 10 ko thuộc Dept
-> 990 thuộc Dept



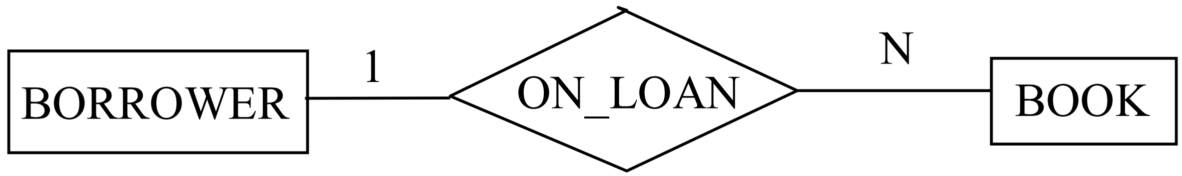
E1 (K1)
E2 (K2)
R (K1,K2)

E1 (K1)
E2 (K2)
R (K1,K2)

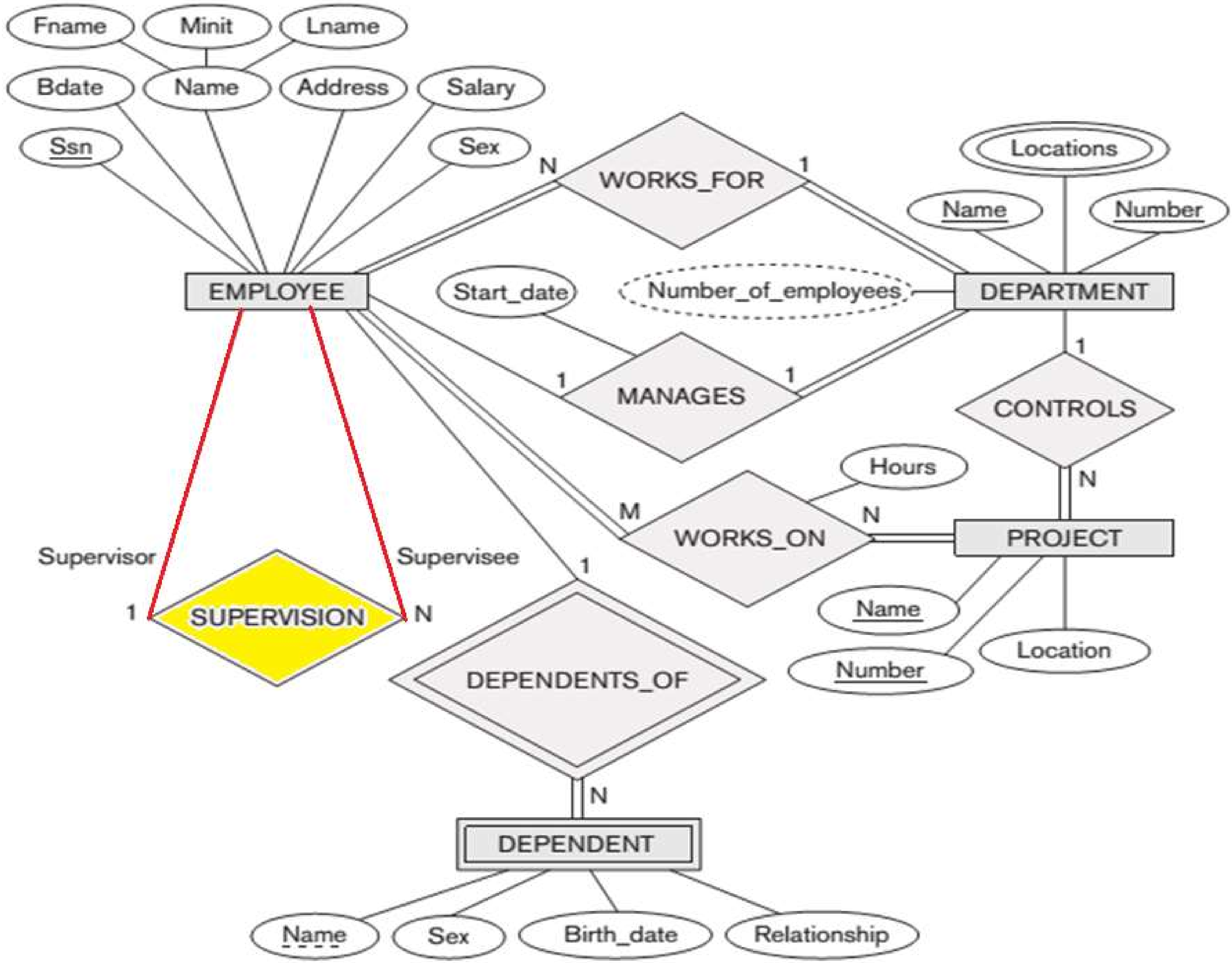
E1 (K1)
E2 (K2, K1)

E1 (K1, K2)
E2 (K2)

ER-to-Relational Mapping

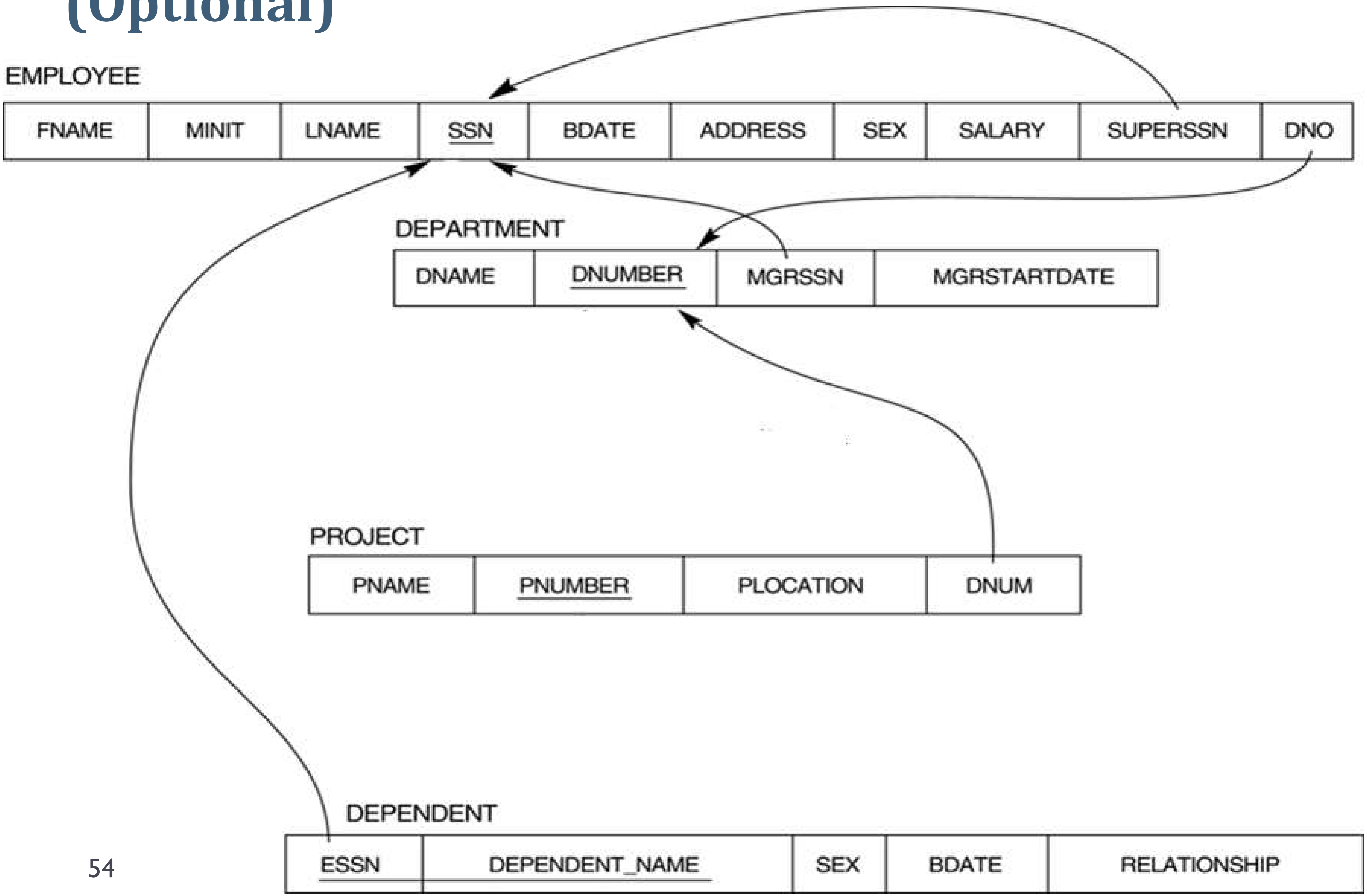


- One possible representation of the relationship:
  - BORROWER(<u>BNUMBER</u>, NAME, ADDRESS, ...)
  - BOOK(<u>ISBN</u>, TITLE, ..., **BNUMBER**)
- A better alternative:
  - BORROWER(<u>BNUMBER</u>, NAME, ADDRESS, ...)
  - BOOK(<u>ISBN</u>, TITLE, ...)
  - ON_LOAN(<u>ISBN</u>, BNUMBER)



Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Step 3-4: Mapping of Relationship Types (Optional)

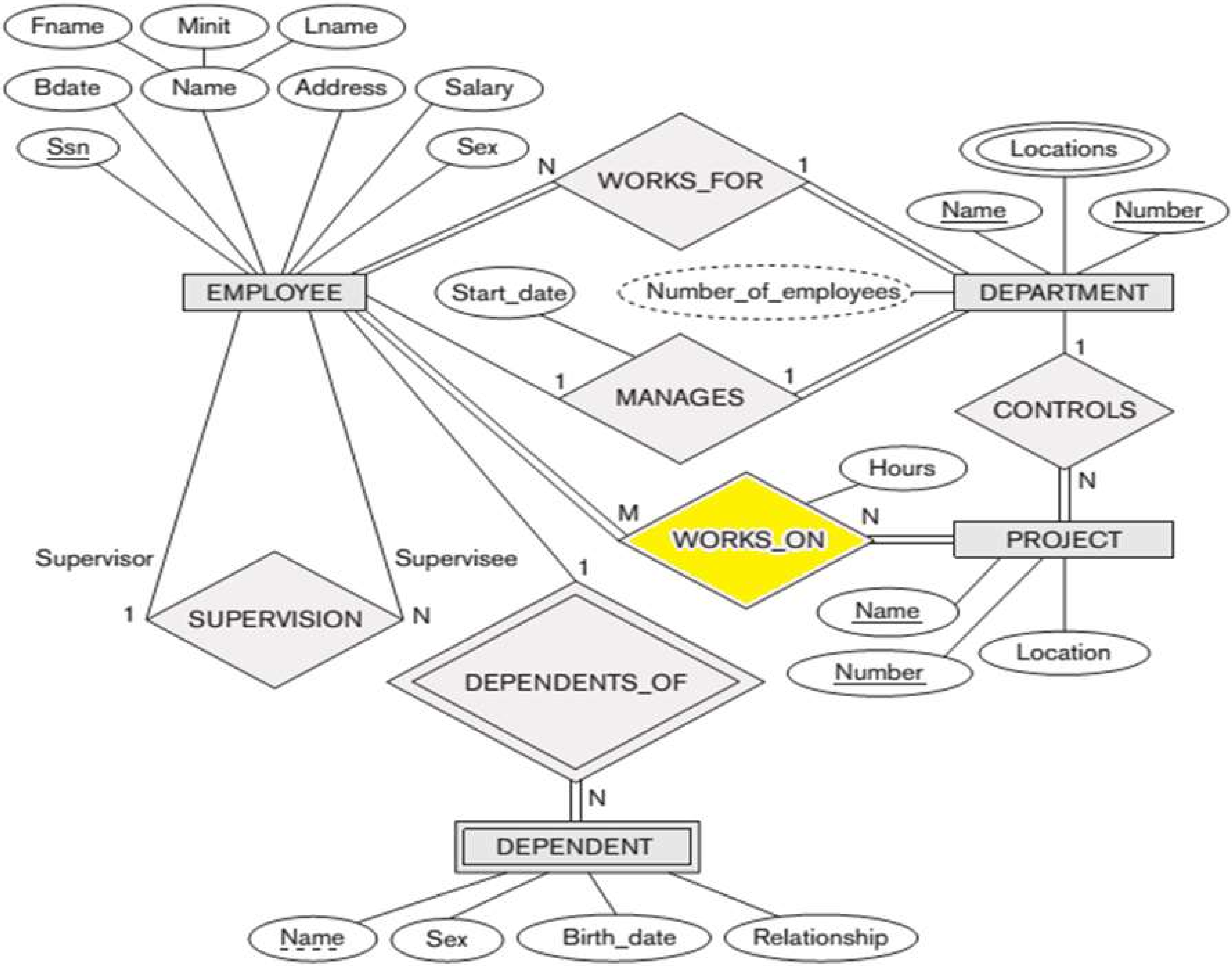


ER-to-Relational Mapping

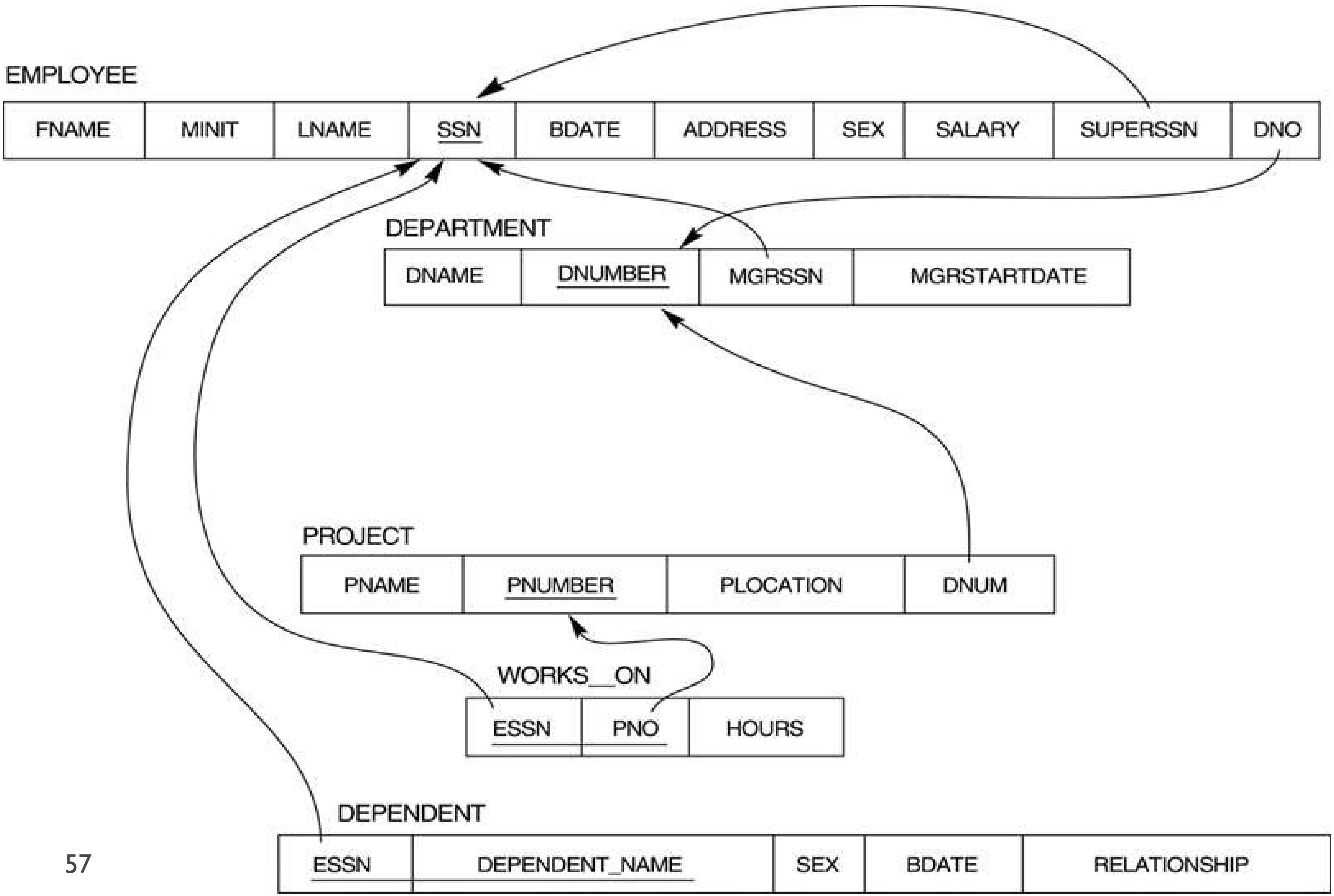
- N:M binary relationships:
  - An N:M relationship is always represented by a new relation which consists of the prime attributes of both participating entity types together with any attributes of the relationship
  - The combination of the prime attributes will form the primary key of the new relation

- **Example:** ENROL is an M:N relationship between STUDENT and MODULE. To represent the relationship, we have a new relation:

  ENROL(<u>SNUMBER, MDL-NUMBER</u>, DATE)

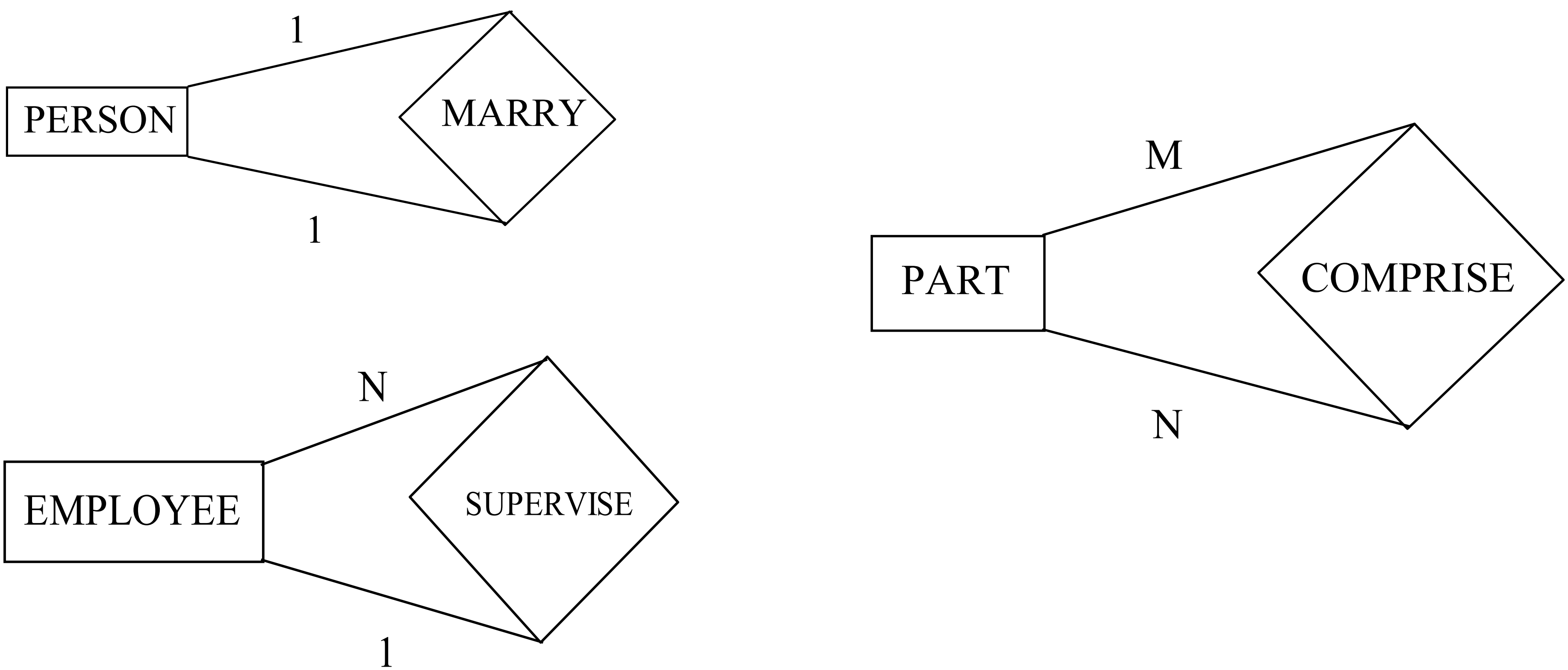

Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Step 5: Mapping M:N Relationship Types



ER-to-Relational Mapping

- Transformation of recursive/involuted relationships
  - Relationship among different instances of the same entity
  - The name(s) of the prime attribute(s) needs to be changed to reflect the role each entity plays in the relationship



ER-to-Relational Mapping

- **Example 1:** 1:1 involuted relationship, in which the memberships for both entities are optional

PERSON(<u>ID</u>, NAME, ADDRESS, …)

MARRY(<u>HUSBAND-ID</u>, WIFE_ID, DATE_OF_MARRIAGE)

ER-to-Relational Mapping

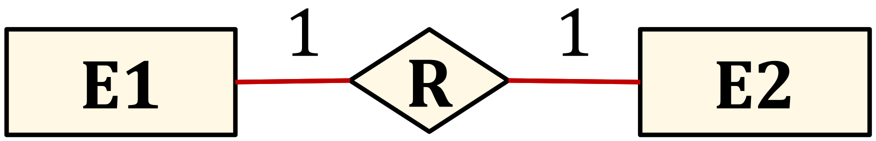
- **Example 2:** 1:M involuted relationship
  - If the relationship is mandatory or almost mandatory:
    EMPLOYEE(<u>ID</u>, ENAME, ..., **SUPERVISOR_ID**)
  - If the relationship is optional:
    EMPLOYEE(<u>ID</u>, ENAME, ...)
    SUPERVISE(<u>ID</u>, START_DATE, ..., **SUPERVISOR_ID**)

- **Example 3:** N:M involuted relationship

  PART(<u>PNUMBER</u>, DESCRIPTION, ...)

  COMPRISE(<u>MAJOR-PNUMBER, MINOR-PNUMBER</u>, QUANTITY)

ER-to-Relational Mapping – Step 3: 1:1

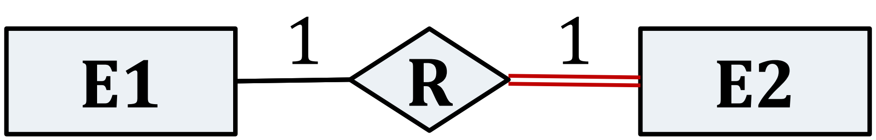


E1 (K1)
E2 (K2, K1)

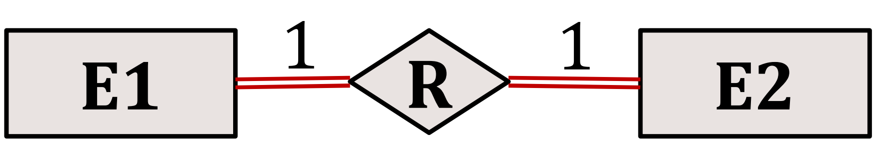
E1 (K1, K2)
E2 (K2)

E1 (K1)
E2 (K2)
R (K1,K2)

E1 (K1)
E2 (K2)
R (K1,K2)



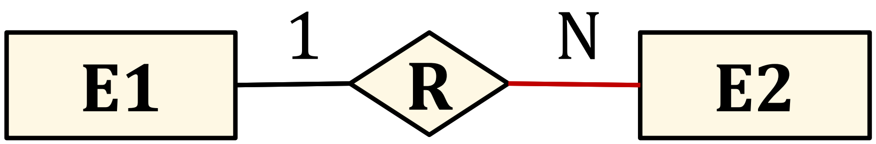
E1 (K1)
E2 (K2, K1)



E1 (K1)
E2 (K2, K1)

E1 (K1, K2)
E2 (K2)

ER-to-Relational Mapping – Step 4: 1:N



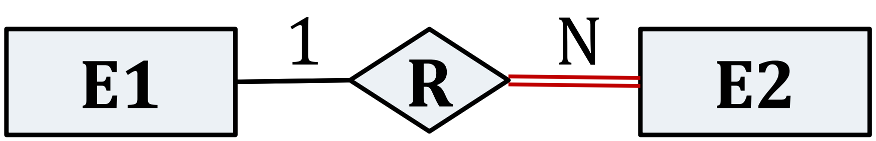
E1 (K1)

E2 (K2, K1)

E1 (K1)

E2 (K2)

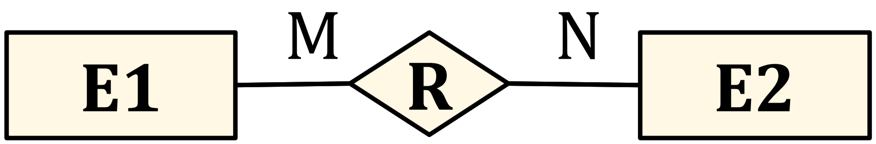
R (K1,K2)



E1 (K1)

E2 (K2, K1)

ER-to-Relational Mapping – Step 5: M:N



E1 (<u>K1</u>)

E2 (<u>K2</u>)

R (<u>K1, K2</u>)

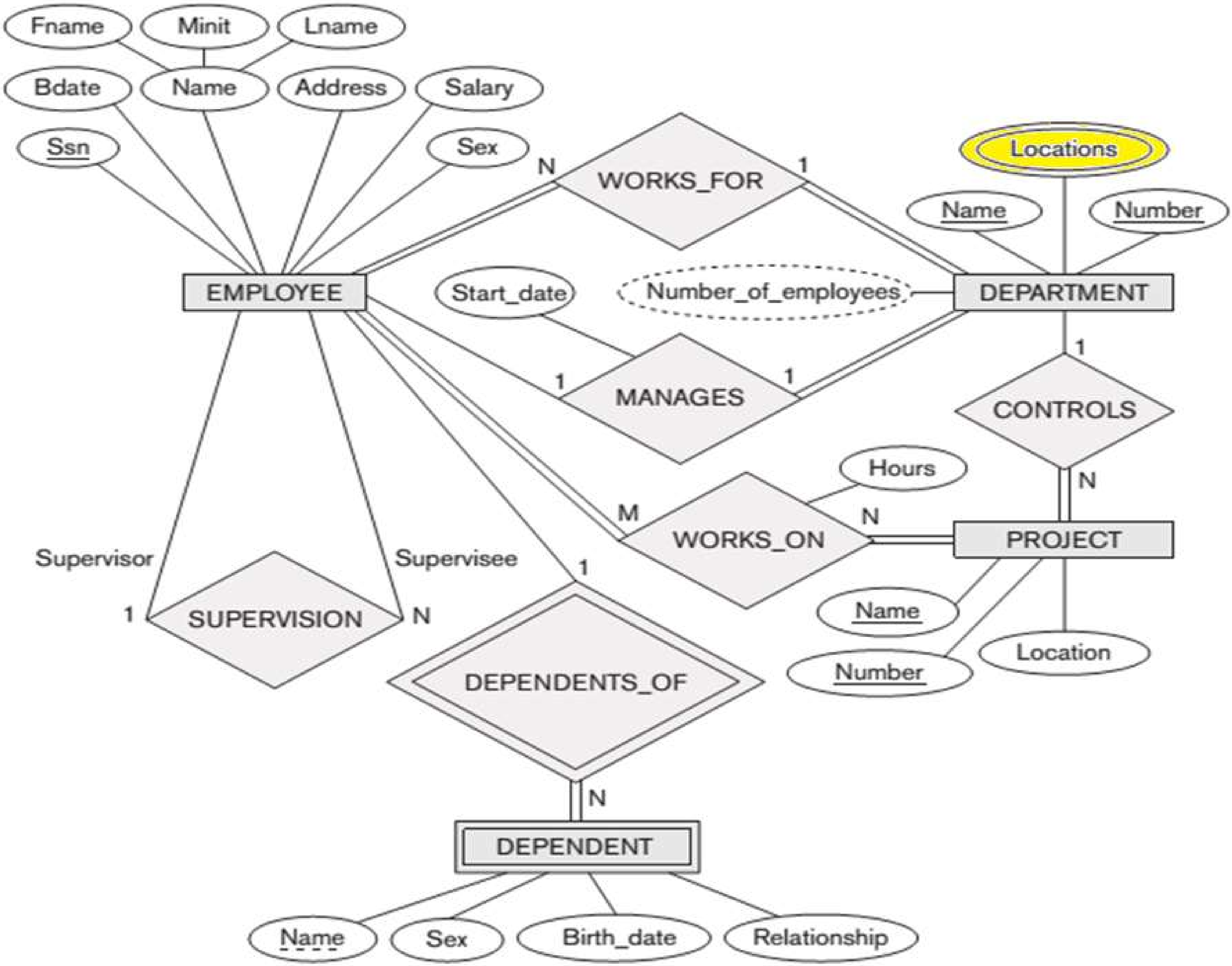
ER-to-Relational Mapping

- ER-
  -
  -
  -
  -
  -
  - Step 6: Mapping of Multivalued attributes
  - Step 7: Mapping of N-ary Relationship Types

ER-to-Relational Mapping

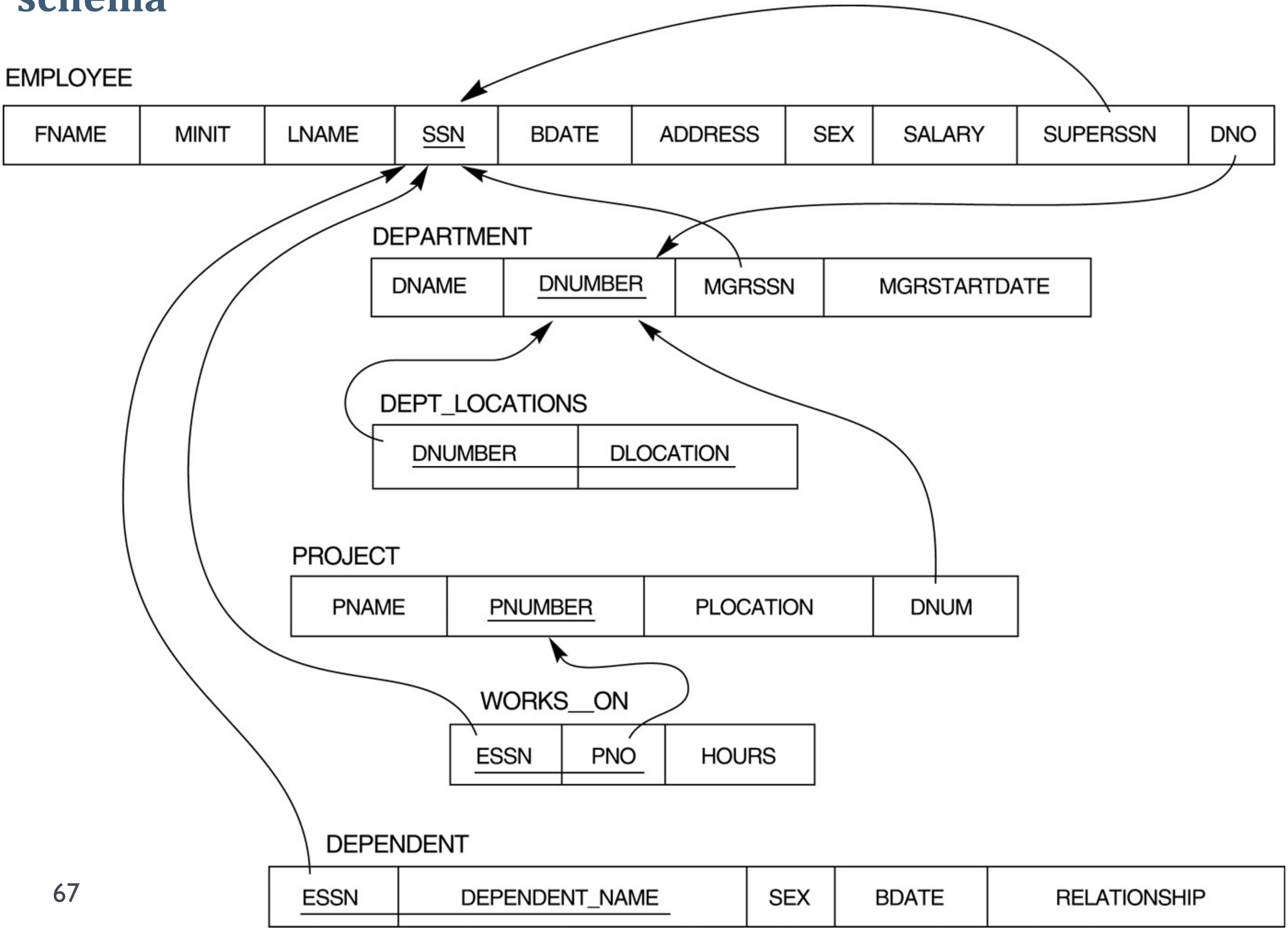
- Step 6: Mapping of Multivalued attributes
  - For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type or relationship type that has 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: 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}



Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
N
WORKS_ON
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Result of mapping the COMPANY ER schema into a relational schema



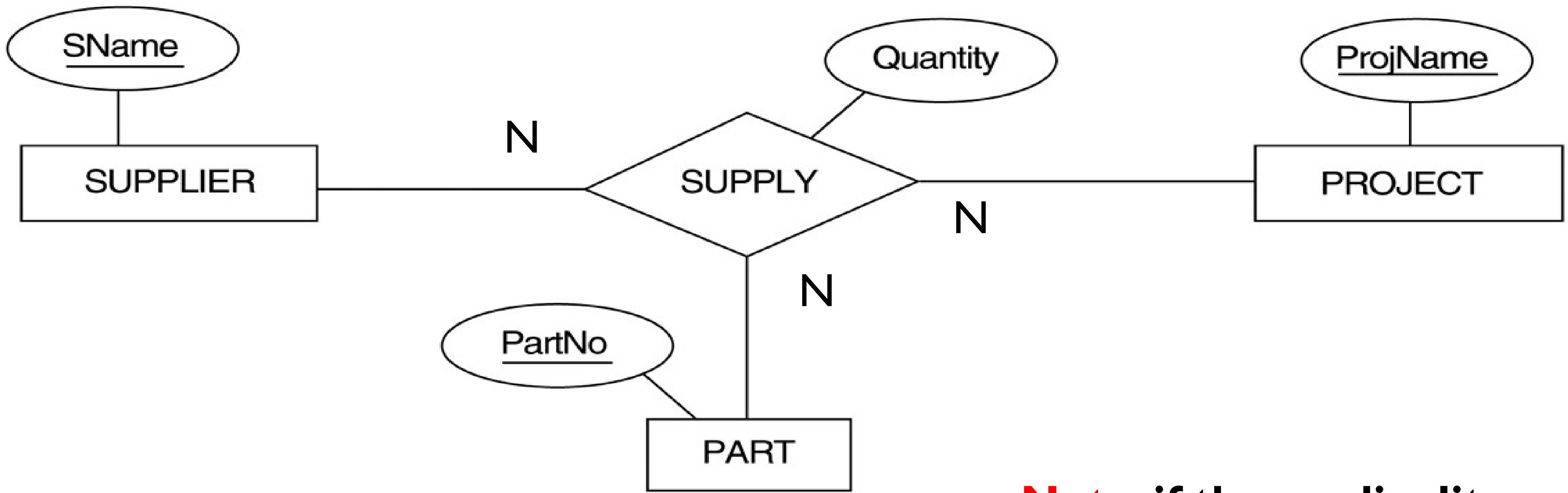
ER-to-Relational Mapping

- 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}

ER-to-Relational Mapping

Ternary relationship types: The SUPPLY relationship



SUPPLIER

SNAME	. . .

PROJECT

PROJNAME	. . .

PART

PARTNO	. . .

SUPPLY

SNAME	PROJNAME	PARTNO	QUANTITY

Note: if the cardinality constraint on any of the entity types E participating in the relationship is 1, the PK should not include the FK attributes that reference the relation E' corresponding to E

ER-to-Relational Mapping

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 & 2 foreign keys
n-ary relationship type	"Relationship" relation & 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- & EER-to-Relational Mapping

- ER-
  - Step 1: Mapping of Regular Entity Types
  - Step 2: Mapping of Weak Entity Types
  - Step 3: Mapping of Binary 1:1 Relationship 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
- EER-
  - Step 8: Options for Mapping Specialization or Generalization.
  - Step 9: Mapping of Union Types (Categories)

EER-to-Relational Mapping

- **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:

  - Option 8A: Multiple relations-Superclass and subclasses
  - Option 8B: Multiple relations-Subclass relations only
  - Option 8C: Single relation with one type attribute
  - Option 8D: Single relation with multiple type attributes

EER-to-Relational Mapping

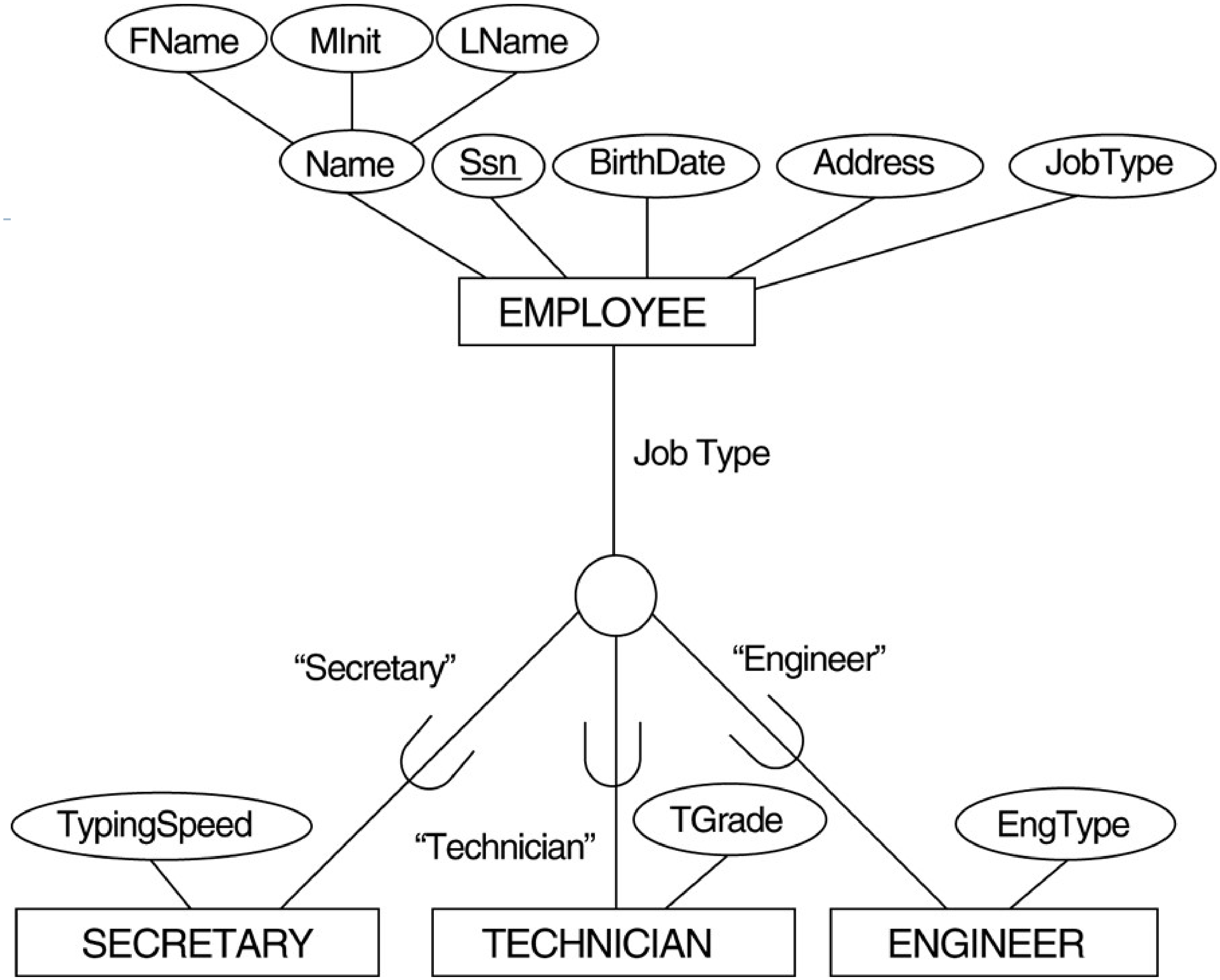
- **Option 8A: Multiple relations-Superclass and subclasses**

  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 attributesAttrs(Li) = {k} U {attributes of Si} and PK(Li)=k. This option works for any specialization (total or partial, disjoint or 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).

Example: Option 8A



EMPLOYEE

SSN	FName	MInit	LName	BirthDate	Address	JobType

SECRETARY

SSN	TypingSpeed

TECHNICIAN

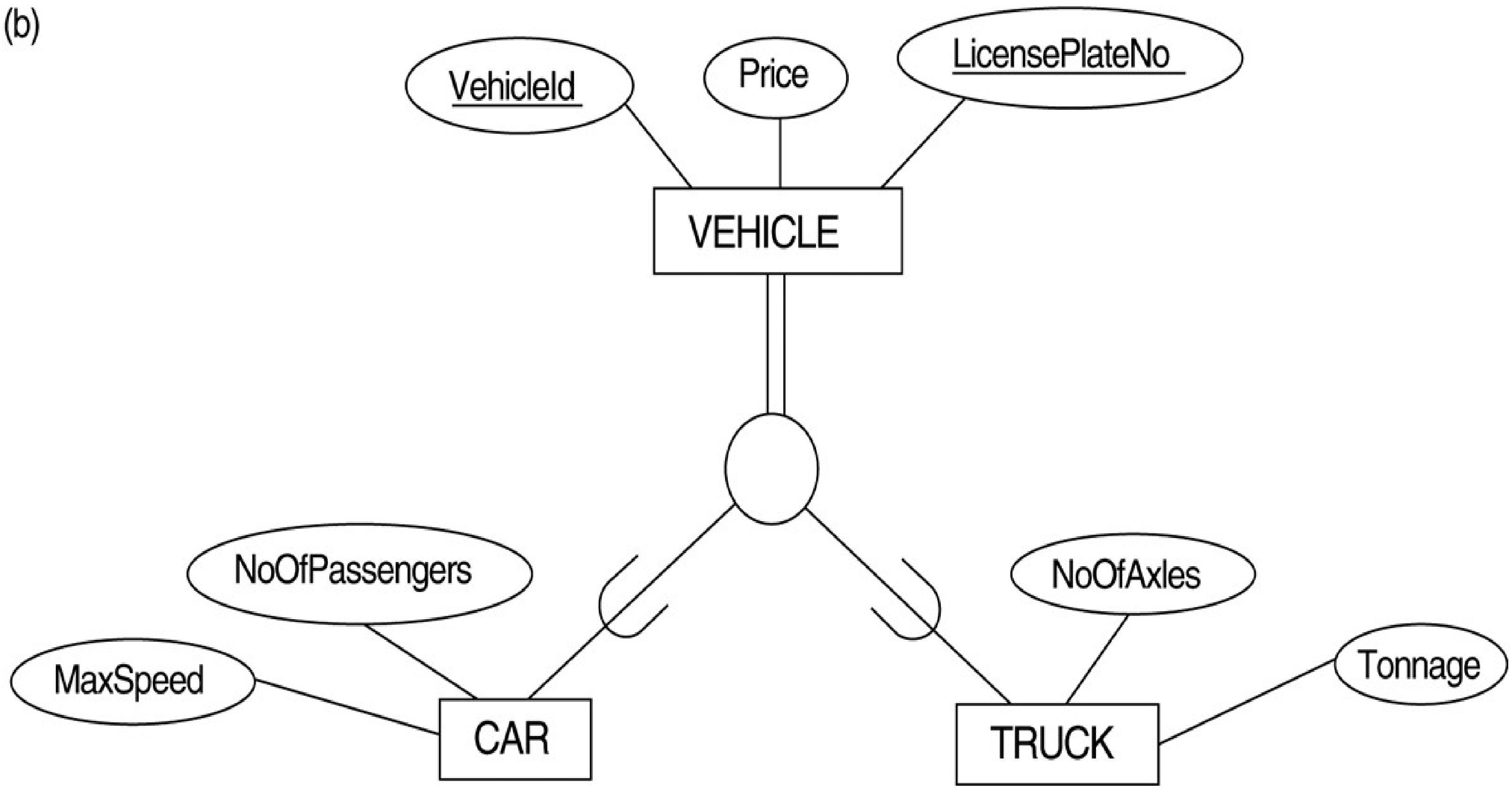
SSN	TGrade

ENGINEER

SSN	EngType

Example: Option 8B

(b)



CAR

VehicleId	LicensePlateNo	Price	MaxSpeed	NoOfPassengers

TRUCK

VehicleId	LicensePlateNo	Price	NoOfAxles	**Tonnage**

EER-to-Relational Mapping

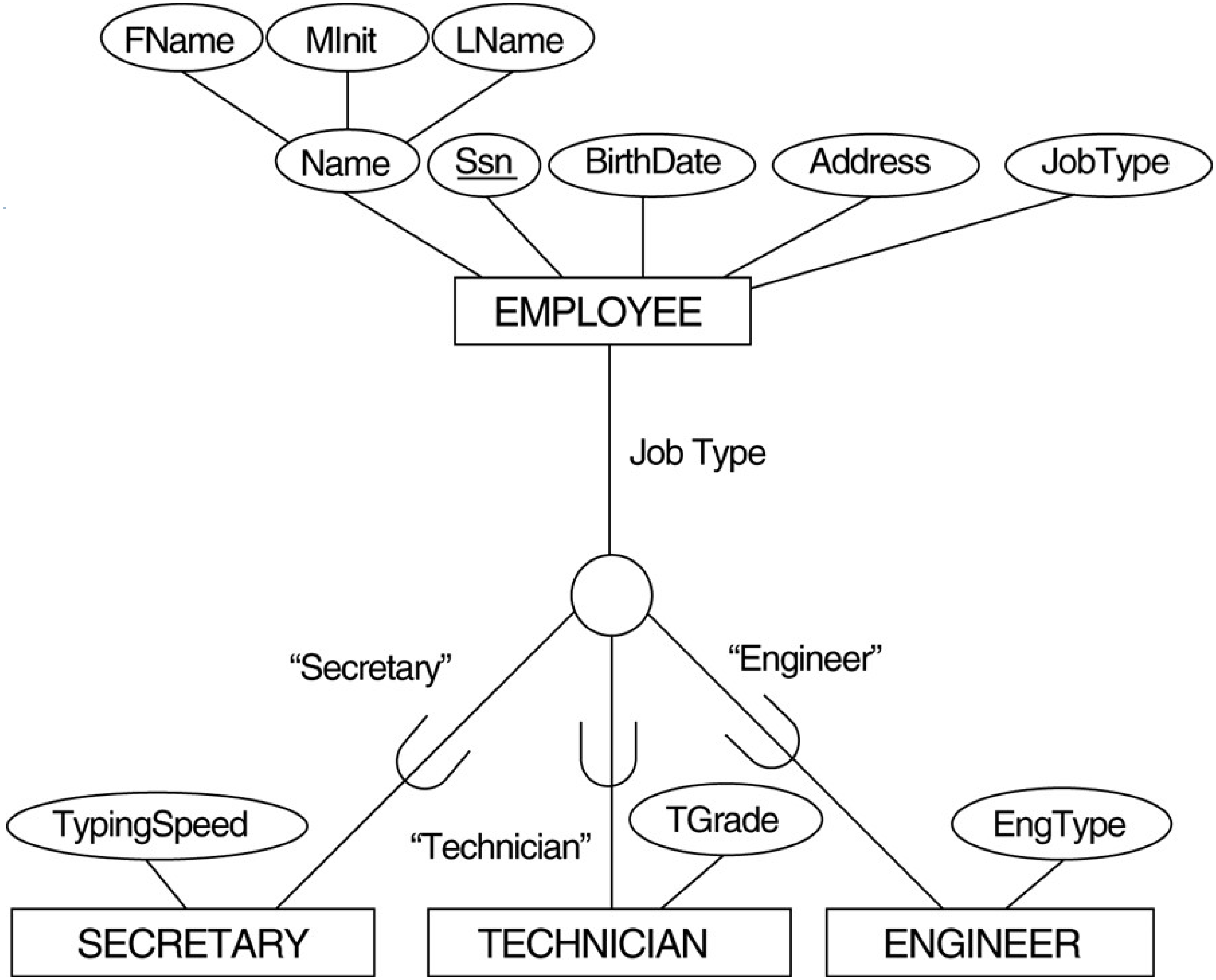
- **Option 8C: Single relation with one type attribute**

  Create a single relation L with attributes Attrs(L) = $\{k, a_1, \ldots a_n\}$ U {attributes of S_1} U…U {attributes of S_m} 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. This option works only for a specialization whose subclasses are *disjoint,*

- **Option 8D: Single relation with multiple type attributes**

  Create a single relation schema L with attributes Attrs(L) = $\{k, a_1, \ldots a_n\}$ U {attributes of S_1} U…U {attributes of S_m} U $\{t_1, t_2, \ldots, t_m\}$ and PK(L) = k. Each t_i, 1 < I < m, is a Boolean type attribute indicating whether a tuple belongs to the subclass S_i.

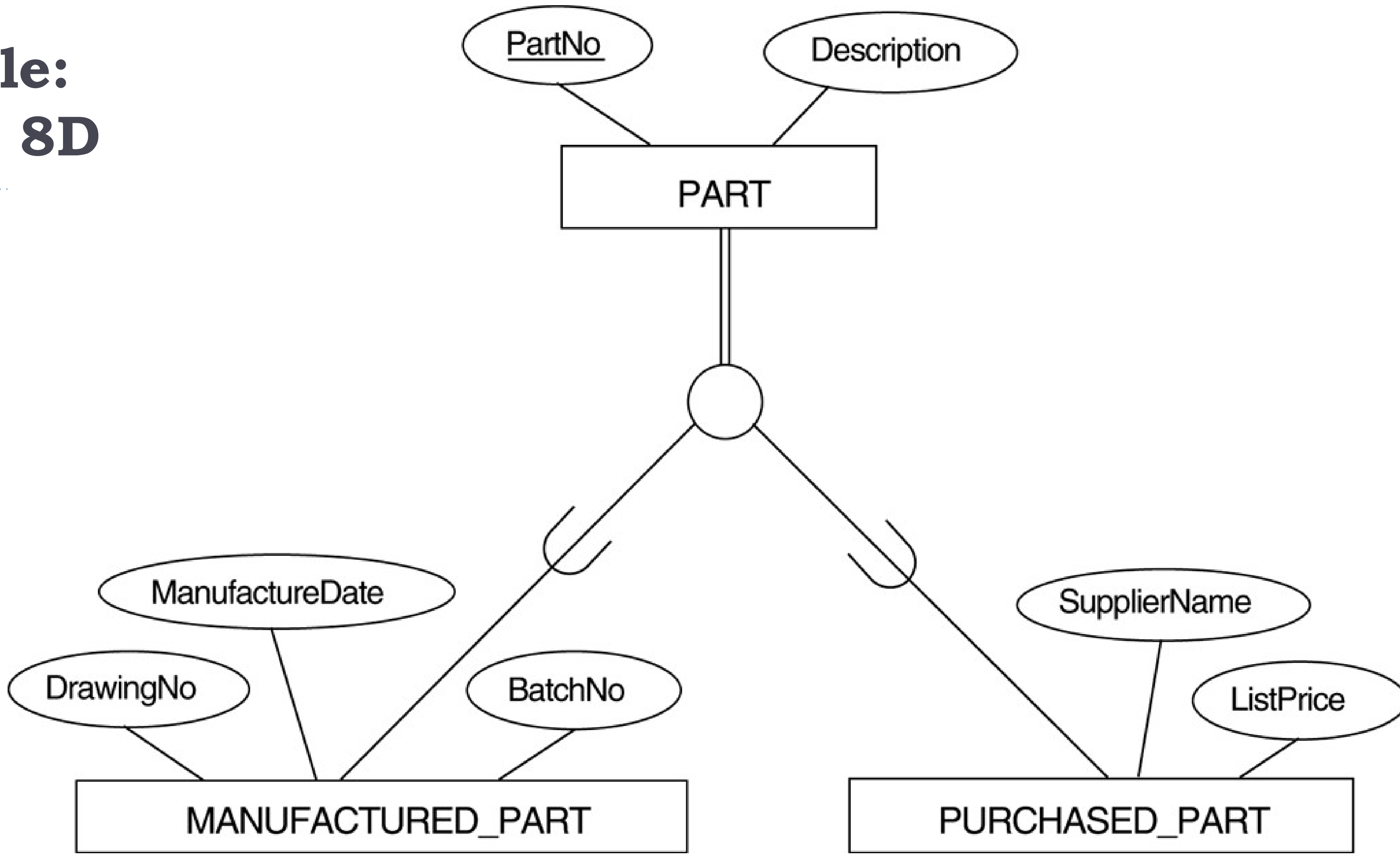
Example: Option 8C



EMPLOYEE

SSN	FName	MInit	LName	BirthDate	Address	JobType	TypingSpeed	TGrade	**EngType**

Example: Option 8D



PART

PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

EER-to-Relational Mapping

- 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.

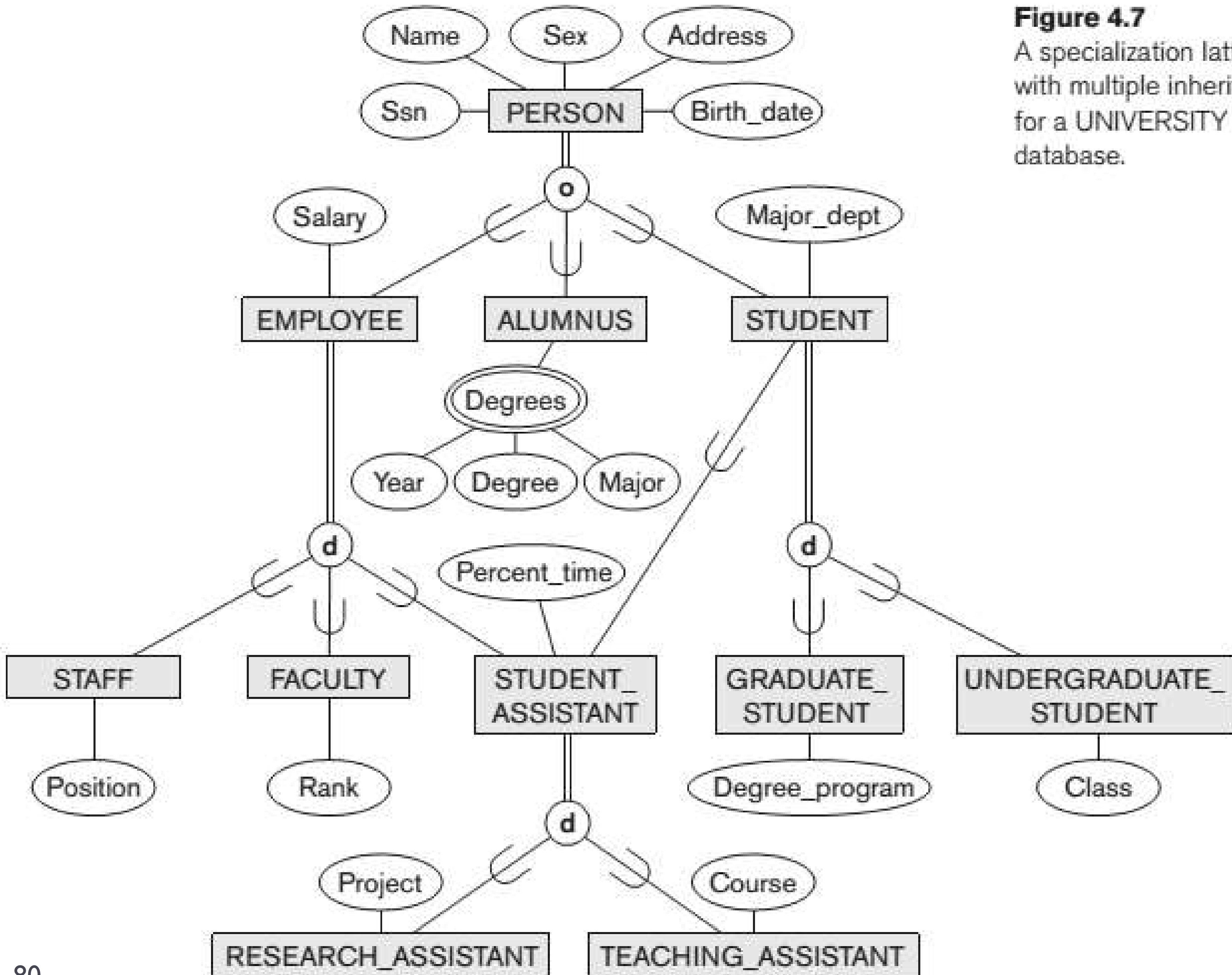


Figure 4.7
A specialization lattice with multiple inheritance for a UNIVERSITY database.

Example: Mapping of Shared Subclasses

Figure 9.6
Mapping the EER specialization lattice in Figure 4.8 using multiple options.

PERSON

Ssn	Name	Birth_date	Sex	Address

EMPLOYEE

Ssn	Salary	Employee_type	Position	Rank	Percent_time	Ra_flag	Ta_flag	Project	Course

ALUMNUS

Ssn

ALUMNUS_DEGREES

Ssn	Year	Degree	Major

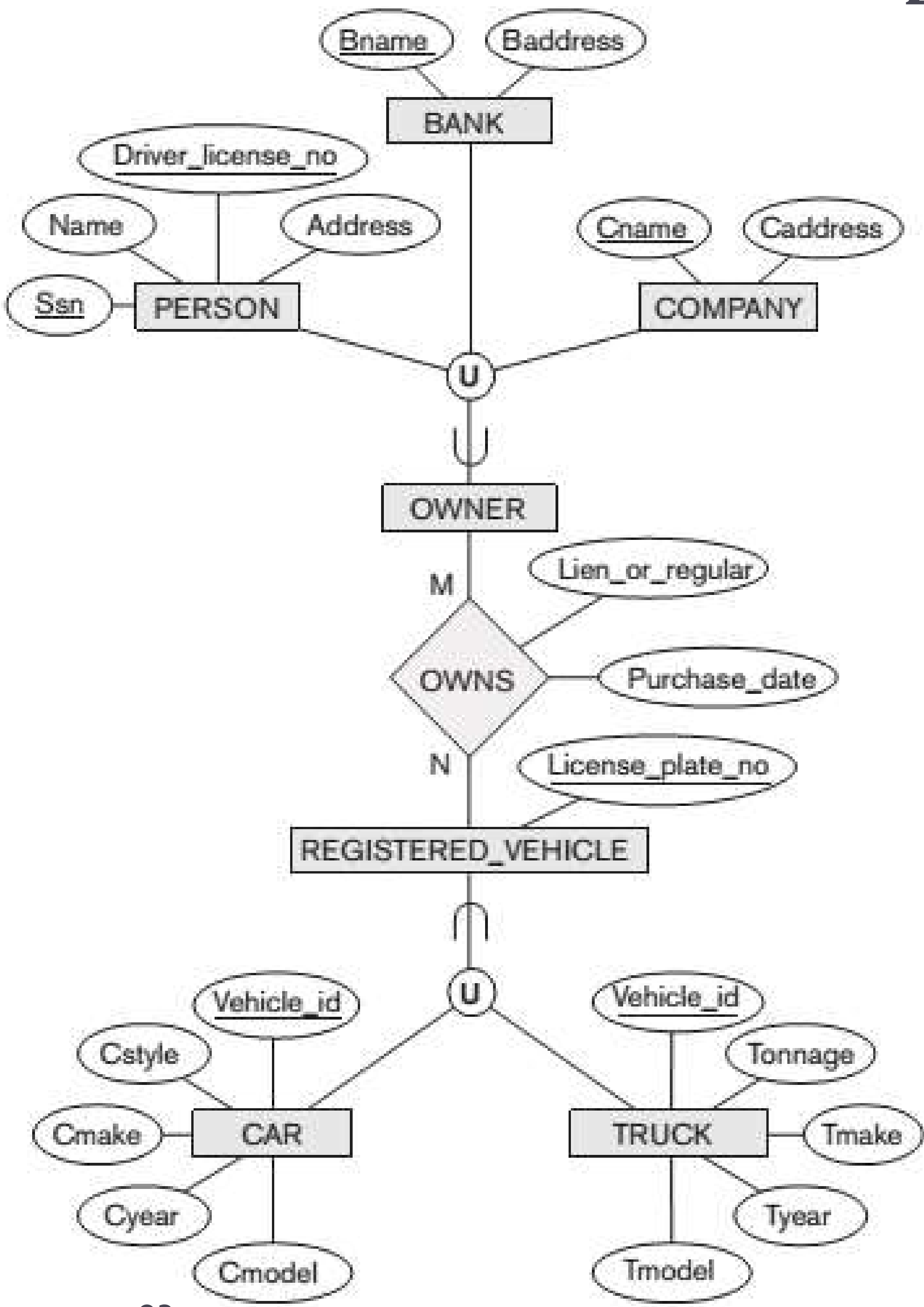
STUDENT

Ssn	Major_dept	Grad_flag	Undergrad_flag	Degree_program	Class	Student_assist_flag

EER-to-Relational Mapping

- **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.

Example: Mapping of Union Types



PERSON

Ssn	Driver_license_no	Name	Address	Owner_id

BANK

Bname	Baddress	Owner_id

COMPANY

Cname	Caddress	Owner_id

OWNER

Owner_id

REGISTERED_VEHICLE

Vehicle_id	License_plate_number

CAR

Vehicle_id	Cstyle	Cmake	Cmodel	Cyear

TRUCK

Vehicle_id	Tmake	Tmodel	Tonnage	Tyear

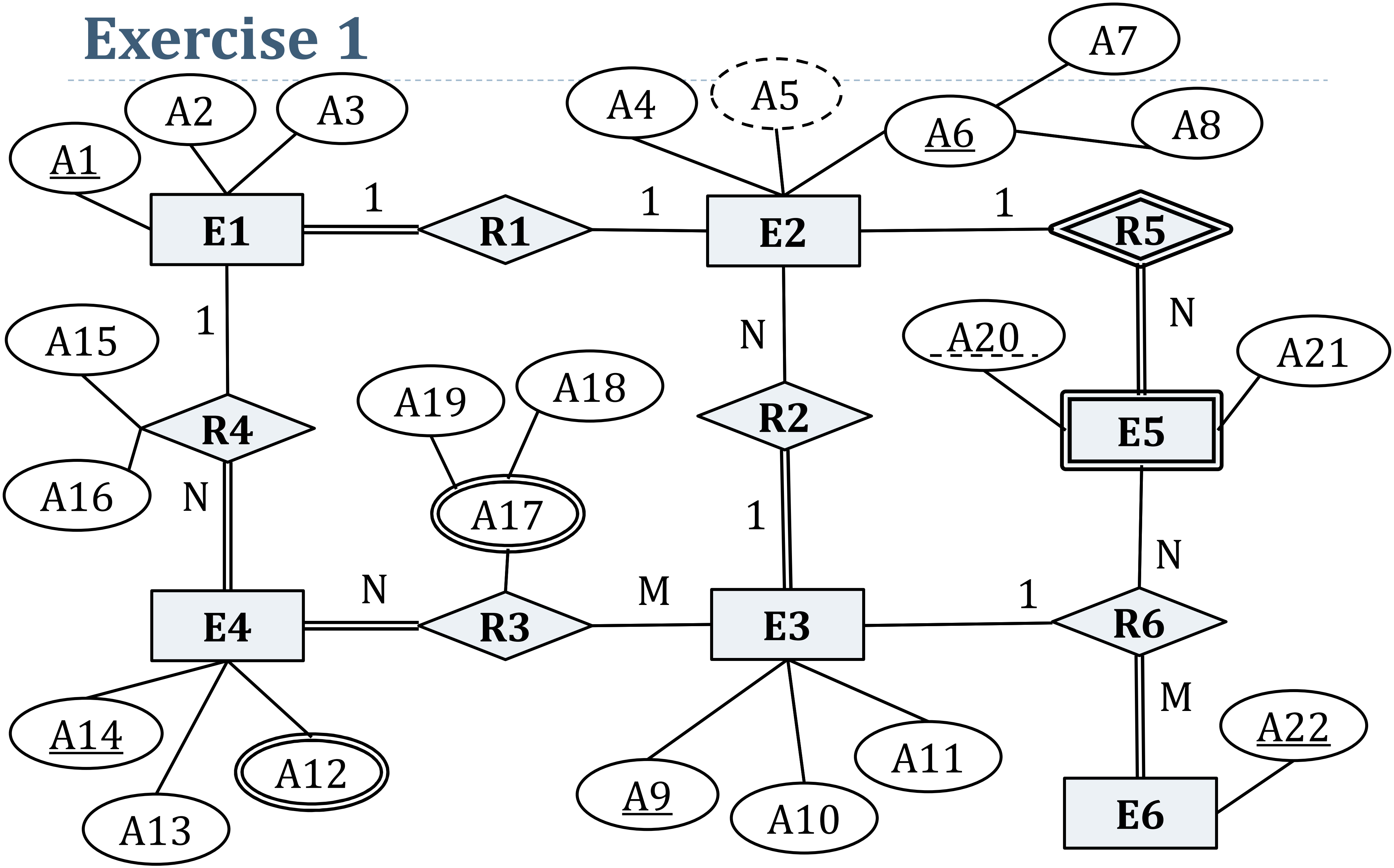
OWNS

Owner_id	Vehicle_id	Purchase_date	Lien_or_regular

Contents


Q
A

Exercise 1


A1
A2
A3
A4
A5
A6
A7
A8
E1
1
R1
1
E2
1
R5
N
A15
1
N
A20
A21
R4
A19
A18
R2
E5
A16
N
A17
1
N
N
M
1
E4
R3
E3
R6
M
A14
A22
A12
A11
A9
A10
E6
A13