CS3402 Database Systems:

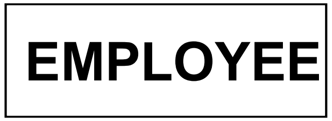
ER , Relational Data Model

ER Model Concepts Revisited -Entity

- **Entity:** specific object represented in the database For example, 'John Smith'

- **Entity type:** Entities with the same attributes are grouped into an entity type

  For example, EMPLOYEE



- **Entity set:** Each entity type has a collection of entities stored in the database called the entity set

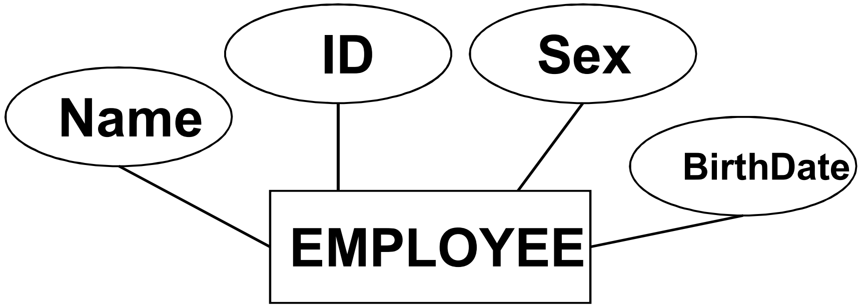
  For example, EMPLOYEE={John Smith, James Black,…}

Note: Same name (EMPLOYEE) used to refer to both the entity type and the entity set

ER Model Concepts Revisited -Attribute

- **Attribute:** properties used to describe an entity

  For example, an EMPLOYEE entity may have Name, ID, Address, Sex, BirthDate



- **Each attribute can take a value from a domain**

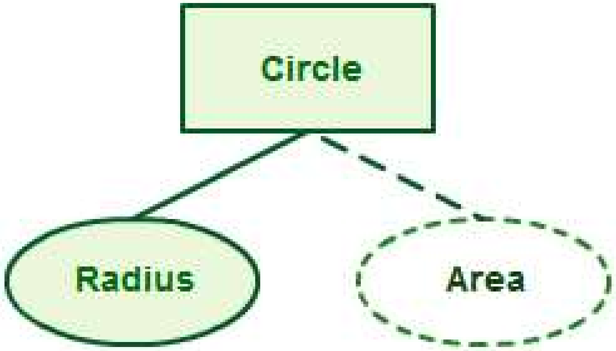
  For example, Name ∈ Character String,

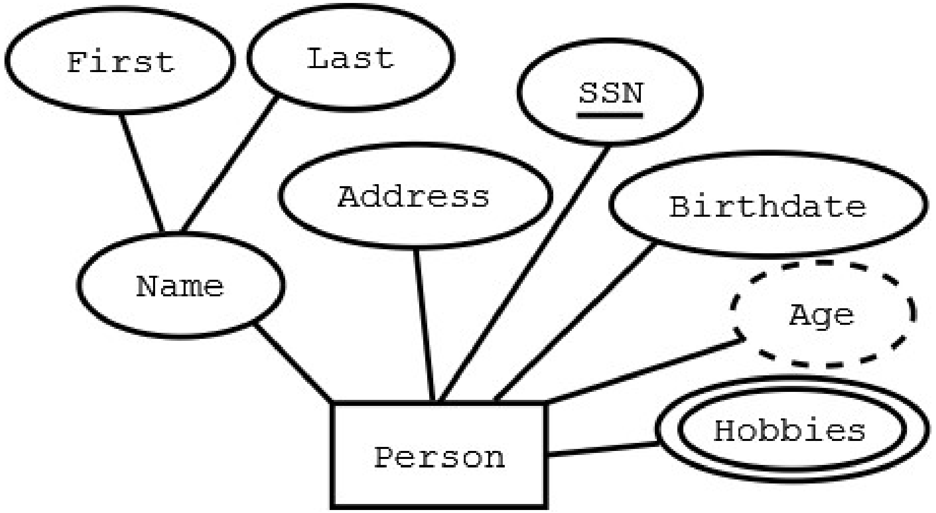
  ID ∈ Integer, ...

- **Attribute type:** Simple, Composite, Multi-valued

ER Model Concepts Revisited -Attribute

- **Derived Attribute:** An **attribute** which can be **derived** from other **attributes** of the entity type is known as **derived attribute**.

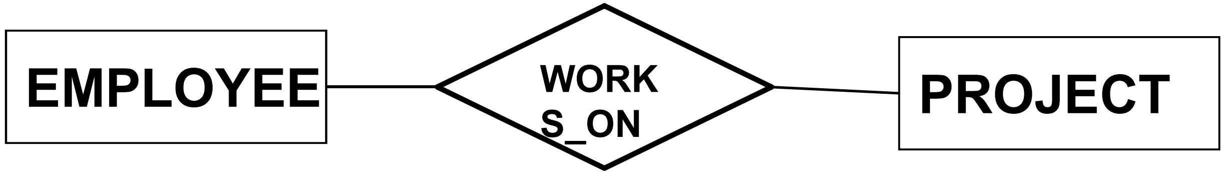




ER Model Concepts Revisited -Relationship

- **Relationship:** relates two or more distinct entities with a specific meaning

  For example, EMPLOYEE John Smith works on the PROJECT 'solar'



- **Relationship Type:**

  Identifies the relationship name and the participating entity types and certain relationship constraints (conditions)

  For example, WORKS_ON relationship type in which EMPLOYEEs and PROJECTs participate

- **Relationship Set:**

  The current set of relationship instances of a relationship type

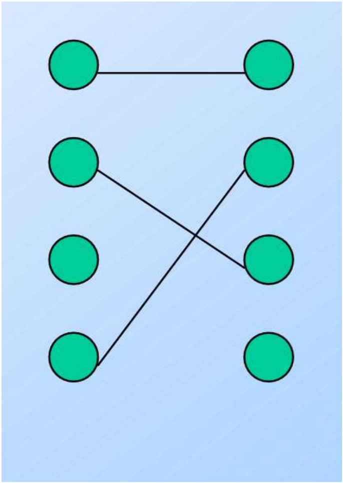
  Note : Similar to: Entity type vs Entity set

Cardinality Constraints of Binary Relationship

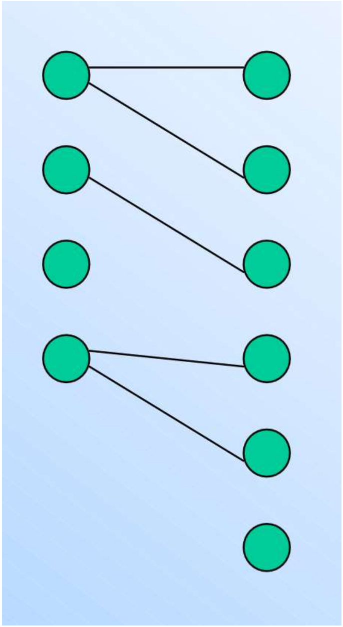
- Suppose R is a relationship connecting sets E and F
- If each member of E can be connected by R to at most one member of F, we say that R is many-to-one from E to F
- Note that in a many-to-one relationship from E to F, each entity in F can be connected to many members of E
- If R is both many-to-one from E to F and many-to-one from F to E, then we say that R is one-to-one
- If R is neither many-to-one from E to F or from F to E, we way R is many-to-many

ER Model Concepts Revisited -Constraints of Relationship

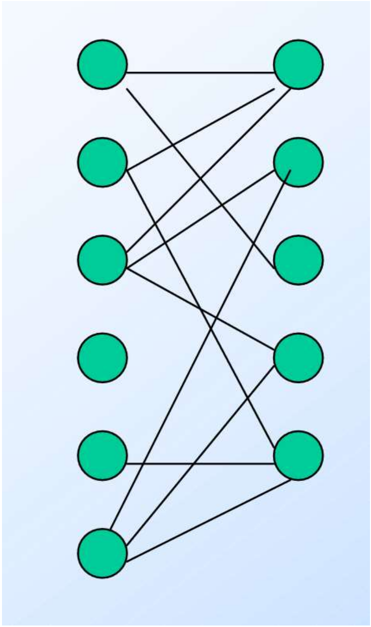
- **Cardinality ratio** (of a binary relationship): 1:1, 1:N, N:1, or M:N



1:1



1:N

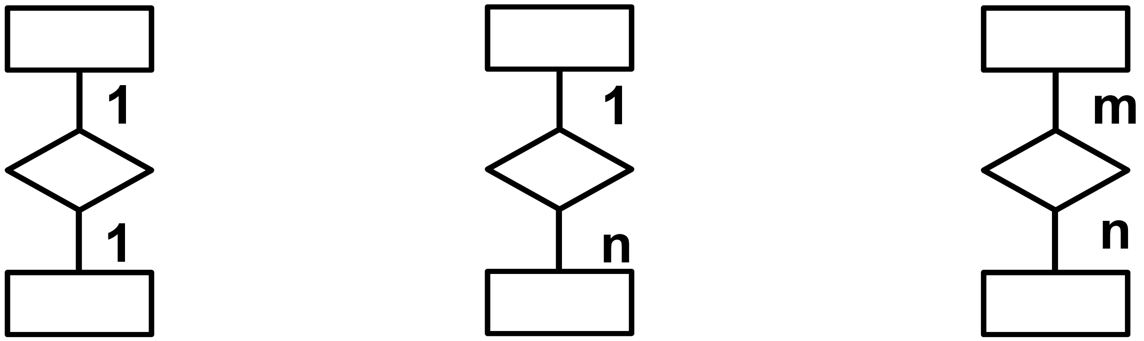


M:N

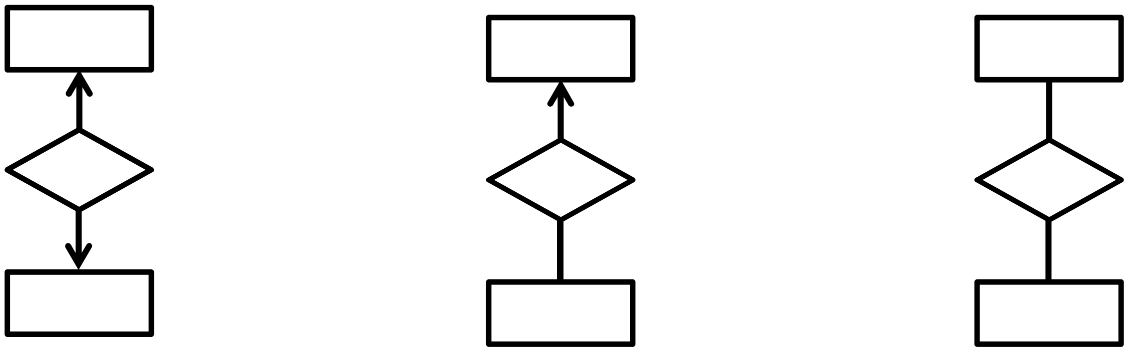
ER Model Concepts Revisited -Constraints of Relationship

- Two ways to indicate **Cardinality ratio** in ER Diagram

1 Place appropriate number on the link.

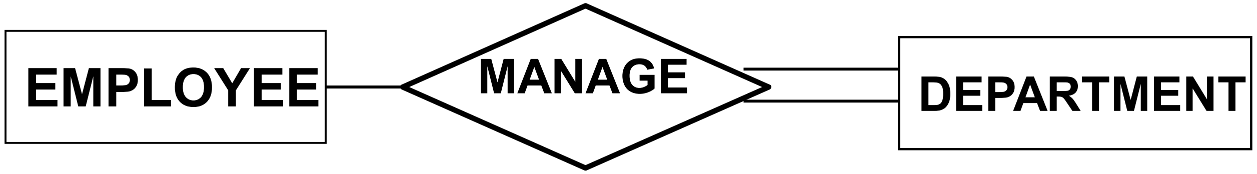


2 Place arrow on the **1** side



ER Model Concepts Revisited -Constraits of Relationship

- **Participation constraint** (on each participating entity type):
  - Specify the minimum no. of relationship instances that each entity can participate in
  - Total (existence dependency) or partial
  - Total shown by **double line**, partial by **single line**
  - E.g., double line: Department to Employee; single line: Employee to Department (not all employees manage department)



(Alternative (min, max) notation for relationship structural constraints:

- Specified on *each participation* of an entity type E in a relationship type R
- Specifies that each entity e in E participates in *at least* min and *at most* max relationship instances in R
- Default (no constraint): min=0, max=n
- Must have min≤max, min≥0, max ≥1

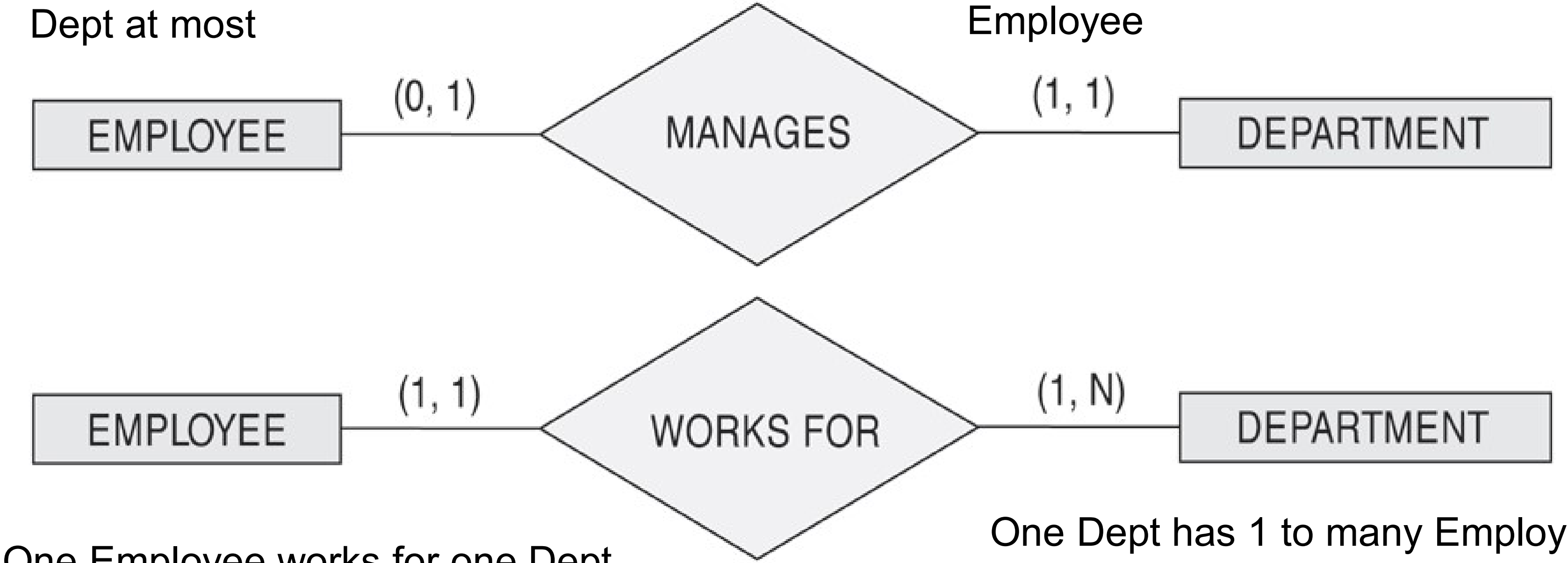
Alternative (min, max) notation for relationship structural constraints:

- Examples:
  - A department has exactly one manager and an employee can manage at most one department
    - Specify (0,1) for participation of EMPLOYEE in MANAGES
    - Specify (1,1) for participation of DEPARTMENT in MANAGES
  - An employee can work for exactly one department but a department can have any number of employees
    - Specify (1,1) for participation of EMPLOYEE in WORKS_FOR
    - Specify (1,n) for participation of DEPARTMENT in WORKS_FOR

The (min,max) notation for relationship constraints

One Employee may manage one Dept at most

One Dept is managed by one Employee



One Employee works for one Dept

One Dept has 1 to many Employee

COMPANY ER Schema Diagram using Cardinality ratio notation

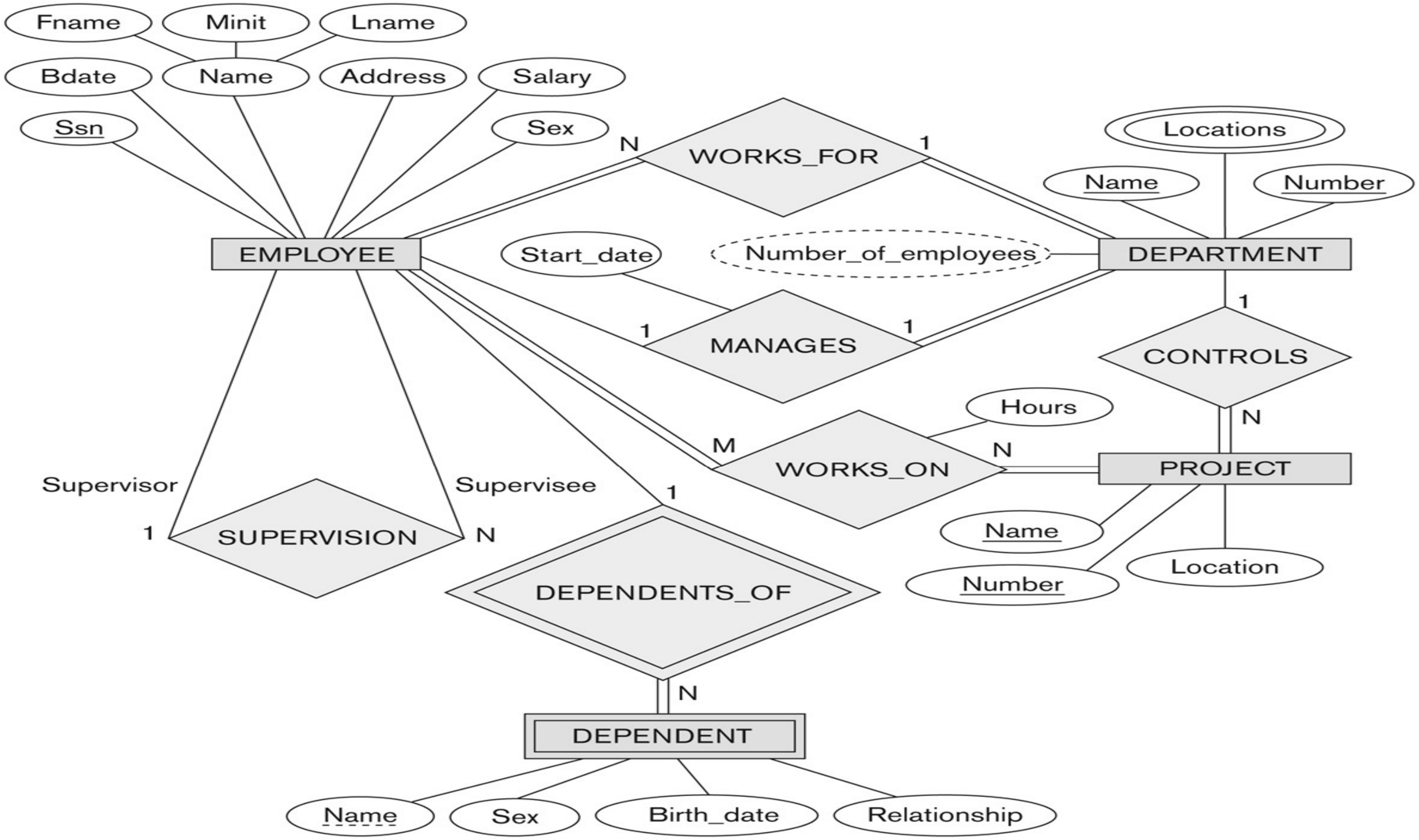


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

COMPANY ER Schema Diagram using (min, max) notation

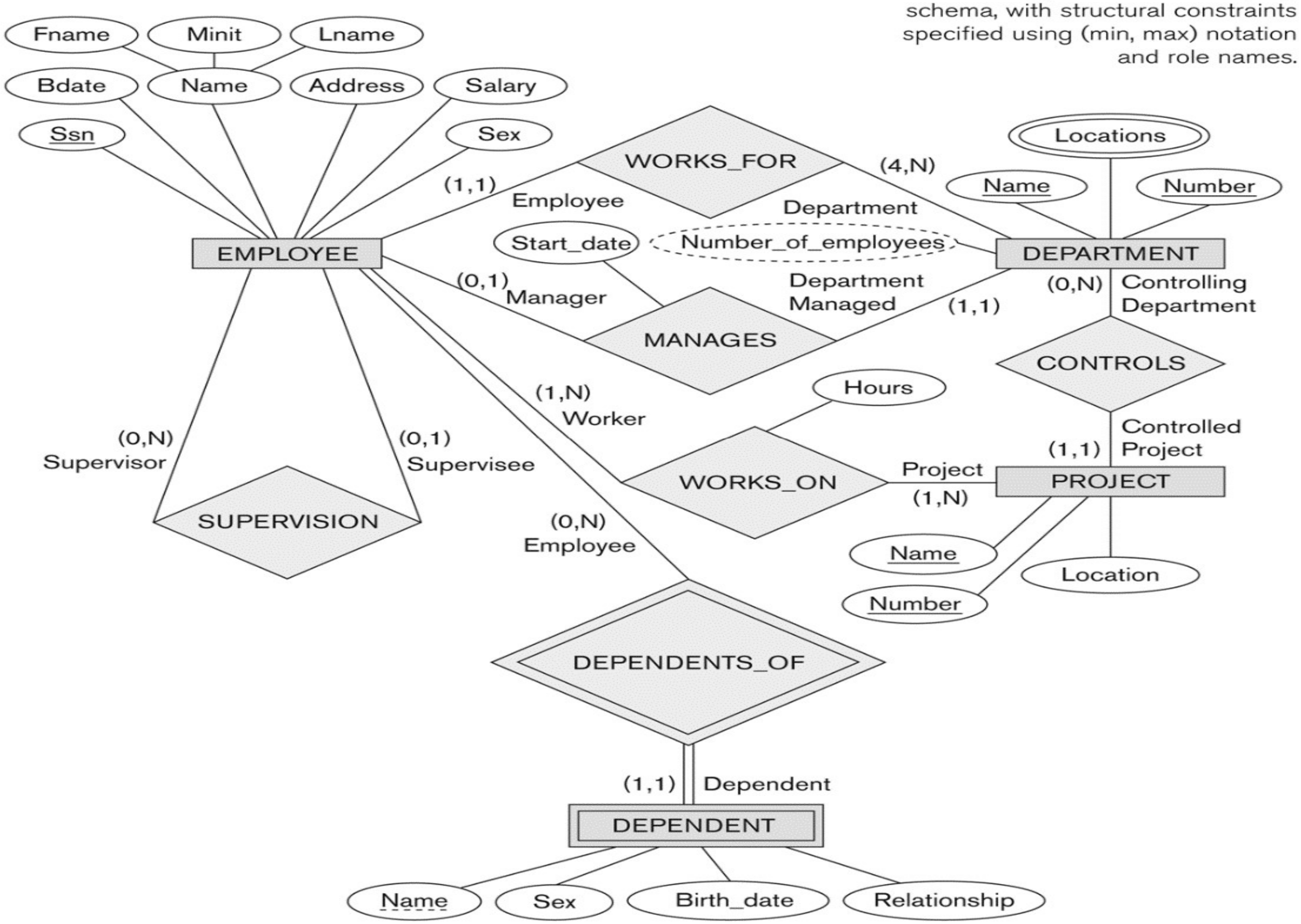


Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

ER Model Concepts Revisited -Constraits of Relationship

- The **degree** of a relationship type is the number of participating entity types.

  Both MANAGES and WORKS_ON are **binary** relationships.

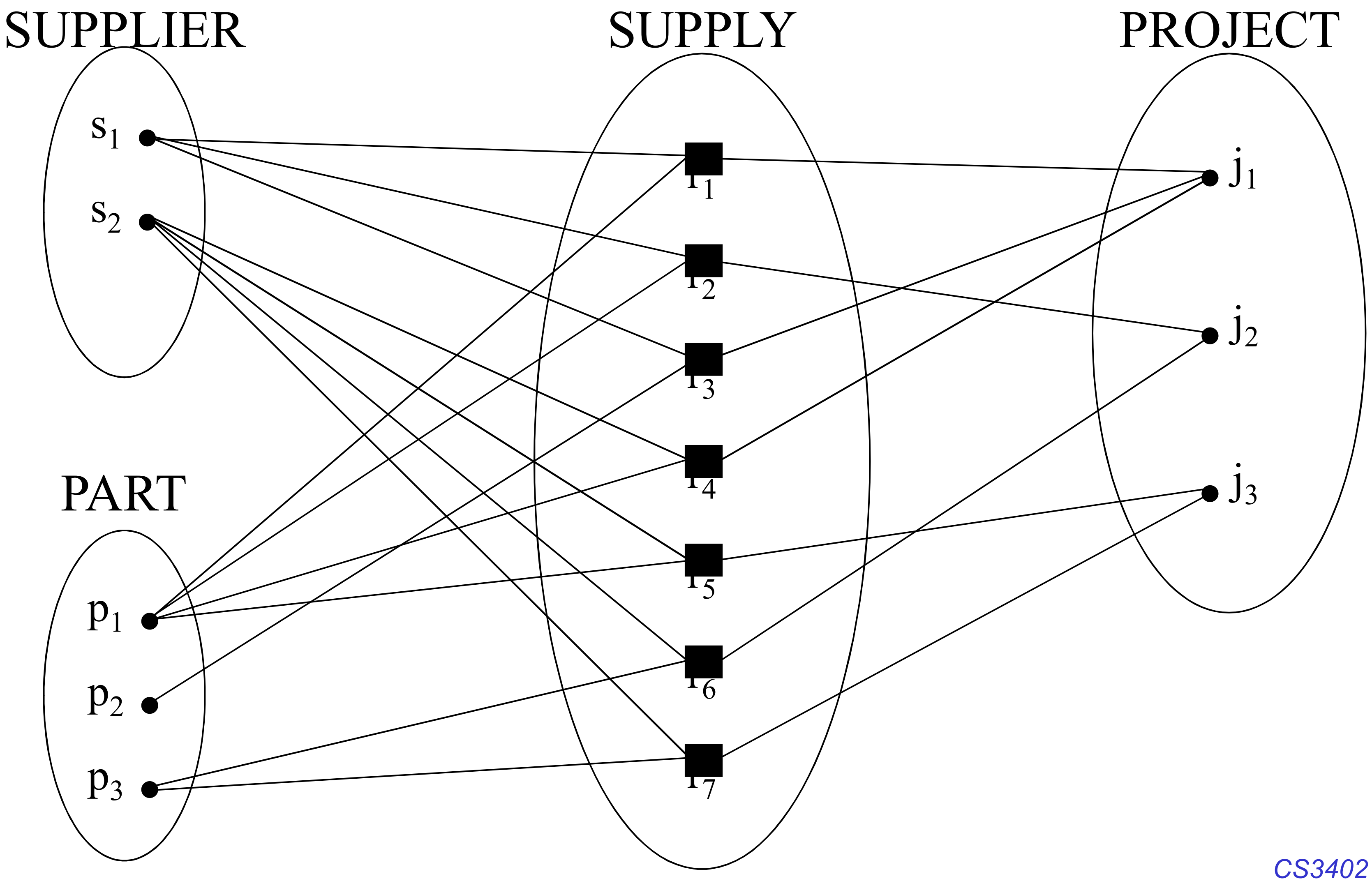
- More than one relationship type can exist with the same participating entity types

For examples, MANAGES and WORKS_FOR are distinct relationships between EMPLOYEE and DEPARTMENT.

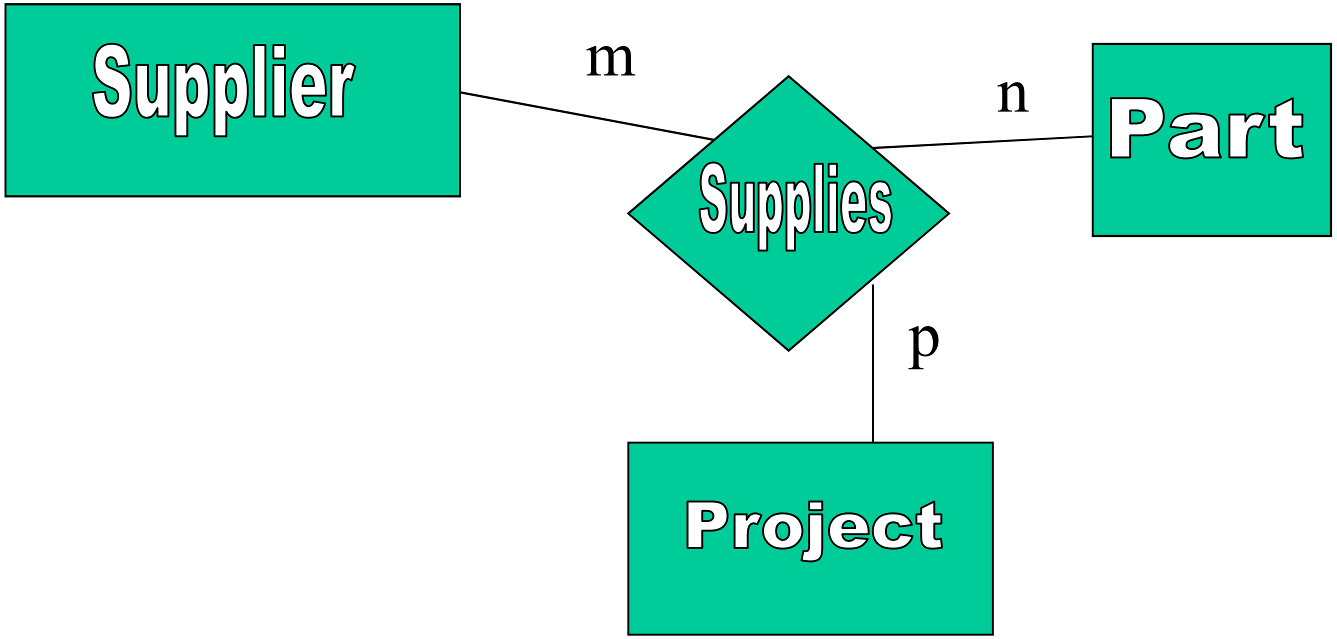
Relationships of Higher Degree

- Relationship types of degree 2 are called binary
- Relationship types of degree 3 are called ternary and of degree n are called n-ary
    - Supplier A supplies part B for project C
- In general, an n-ary relationship is not equivalent to n binary relationships
- Constraints are harder to specify for higher-degree relationships (n > 2) than for binary relationships

Ternary Relationship: Instance Diagram

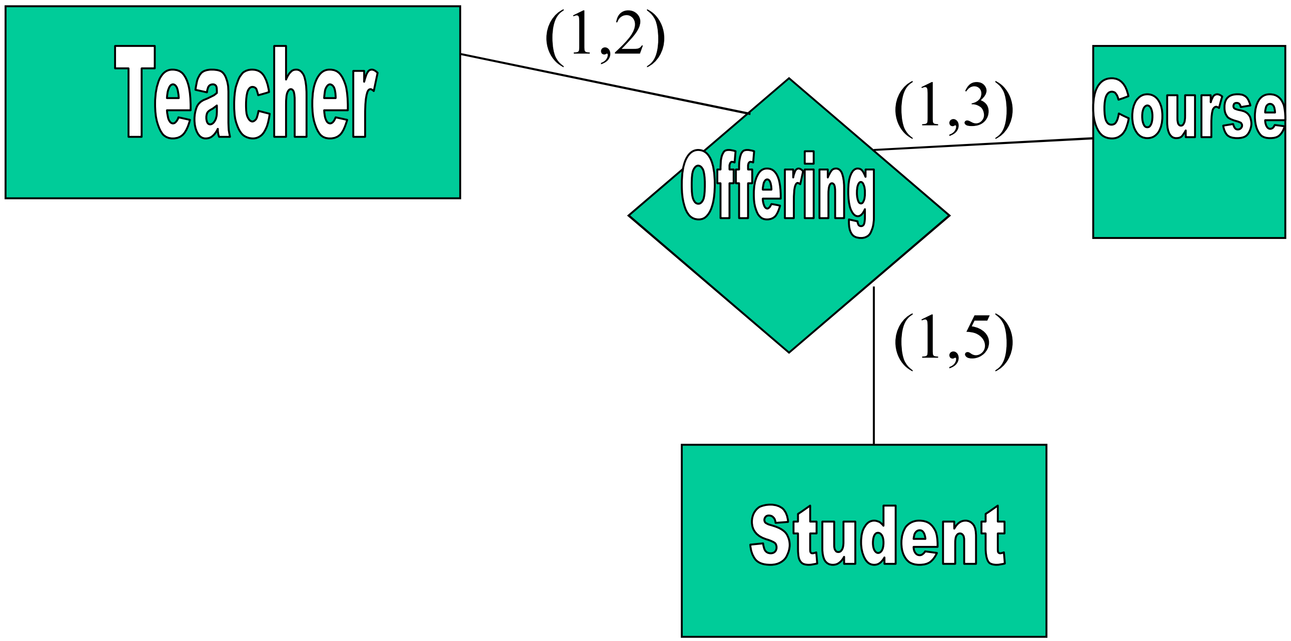


Why not on Higher Order Relationship Types



What does it mean to put m:n:p on the three arms of the relationship? It is essentially meaningless.

The (min,max) Notation for Higher Order Relationship Type Constraints



A Teacher can offer 1 to 2 Offerings

A Course may have 1 to 3 Offerings

A Student may enroll in 1 to 5 Offerings

Example of a ternary relationship

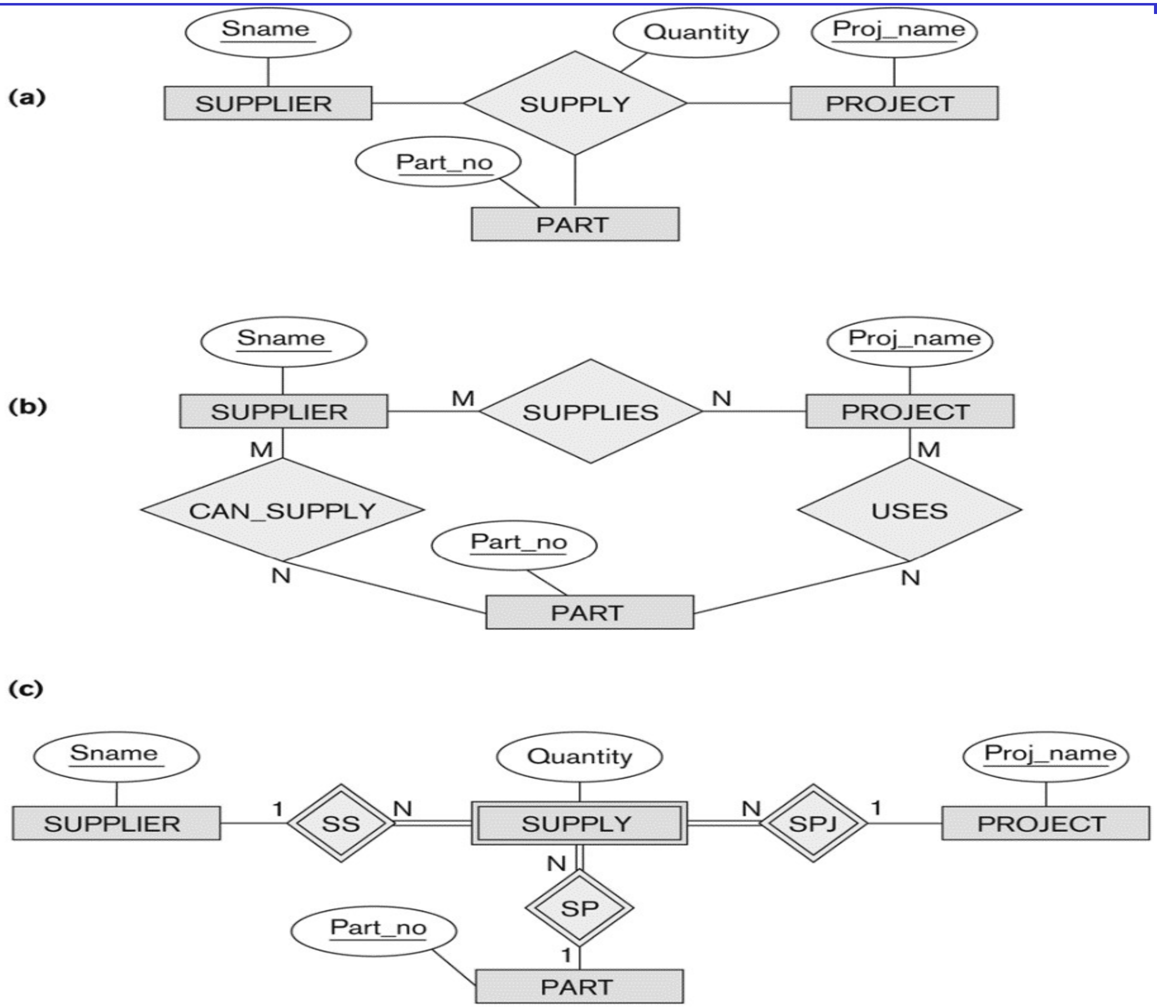


Figure 3.17
Ternary relationship types. (a) The SUPPLY relationship. (b) Three binary relationships not equivalent to SUPPLY. (c) SUPPLY represented as a weak entity type.

(S1, P1, Pj1)

(S1, P1)
(S1, Pj1)
(P1, Pj1)

(S1, Sp1)
(Sp1, Pj1)
(Sp1, P1)

Recursive Relationship

- A relationship type between the same participating entity type in distinct roles (roles in relationships)
- Also called a self-referencing relationship type

- Example: the SUPERVISION relationship

- EMPLOYEE participates twice in two distinct roles:
  - supervisor (or boss) role
  - supervisee (or subordinate) role

- Each relationship instance relates two distinct EMPLOYEE entities:
  - One employee in *supervisor* role
  - One employee in *supervisee* role

A Recursive Relationship Example

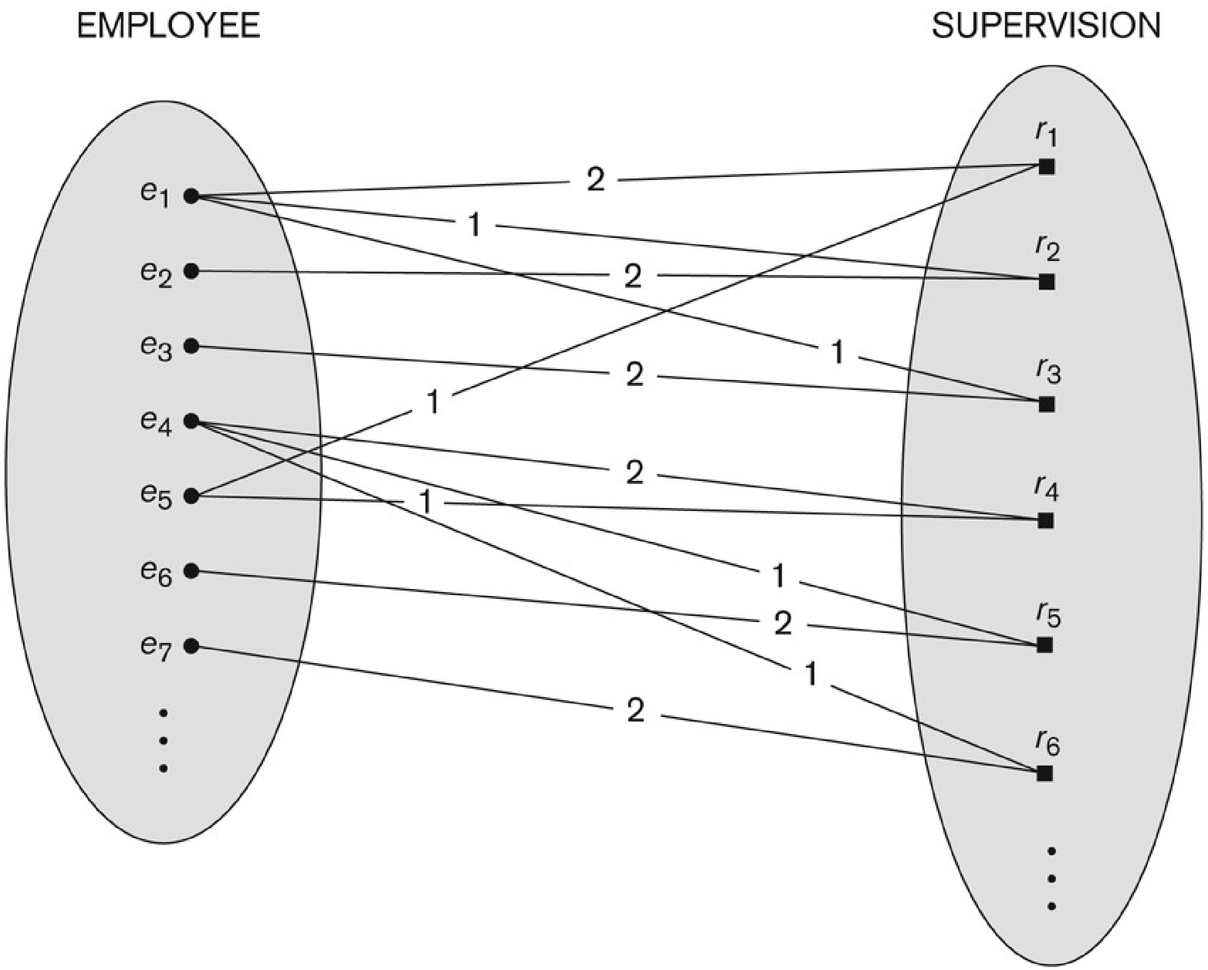
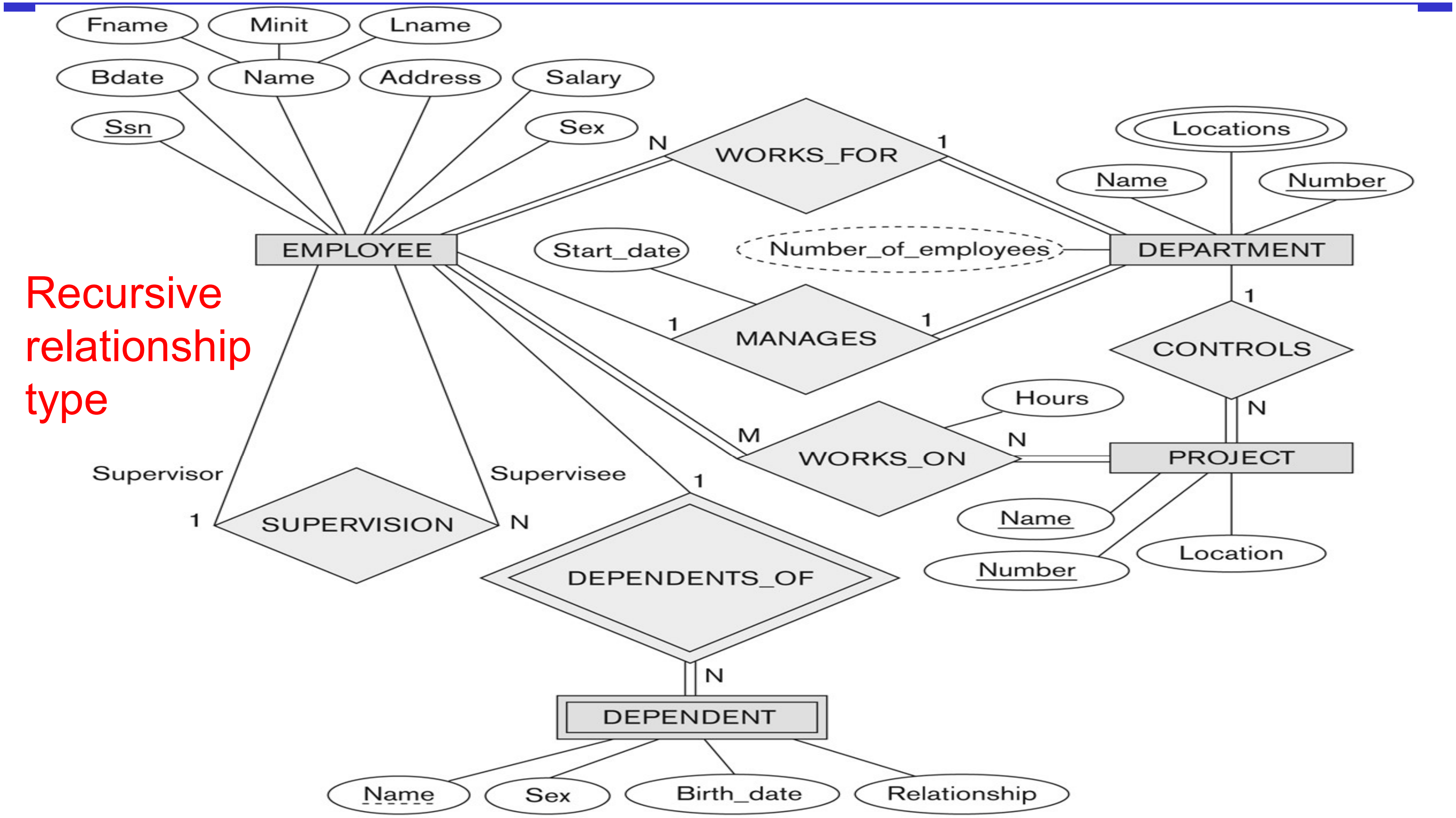


Figure 3.11
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).

Recursive Relationship: SUPERVISION (participation role names are shown)



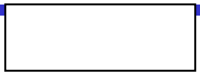
Recursive relationship type

Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

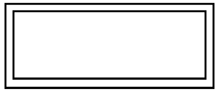
Summary of ER-Diagram Notation

Symbol

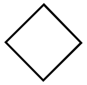
Meaning



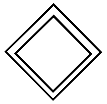
ENTITY TYPE



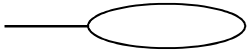
WEAK ENTITY TYPE



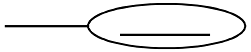
RELATIONSHIP TYPE



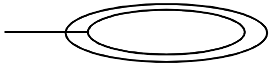
IDENTIFYING RELATIONSHIP TYPE



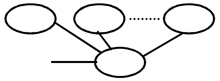
ATTRIBUTE



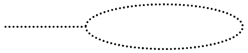
KEY ATTRIBUTE



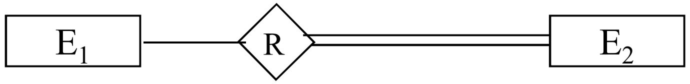
MULTIVALUED ATTRIBUTE



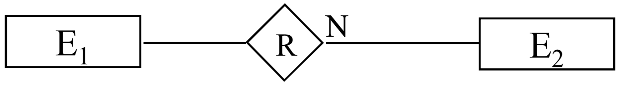
COMPOSITE ATTRIBUTE



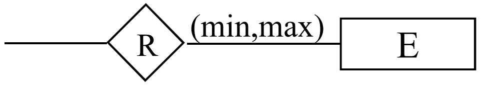
DERIVED ATTRIBUTE



TOTAL PARTICIPATION OF E_2 IN R



CARDINALITY RATIO 1:N FOR E_1:E_2 IN R

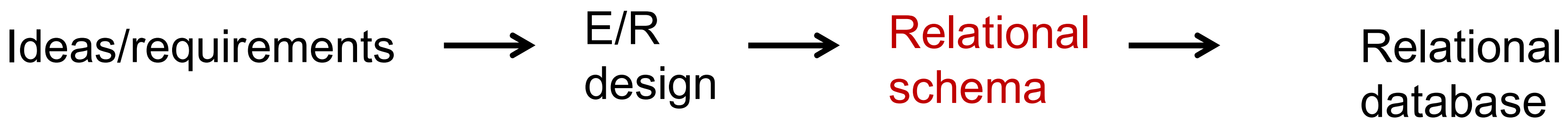


STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R

Relational Model

- Although the E/R approach is a simple and an appropriate way to describe the structure of data, many database implementations are always based on another approach called the relational model
  - E/R diagram -> relation model
- The relational model was first proposed by Dr. E.F. Codd of IBM Research in 1970 in the following paper:
  - "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970
- The above paper caused a major revolution in the field of database management and earned Dr. Codd the coveted ACM Turing Award

Database Modelling and Implementation



Ideas/requirements → E/R design → Relational schema → Relational database

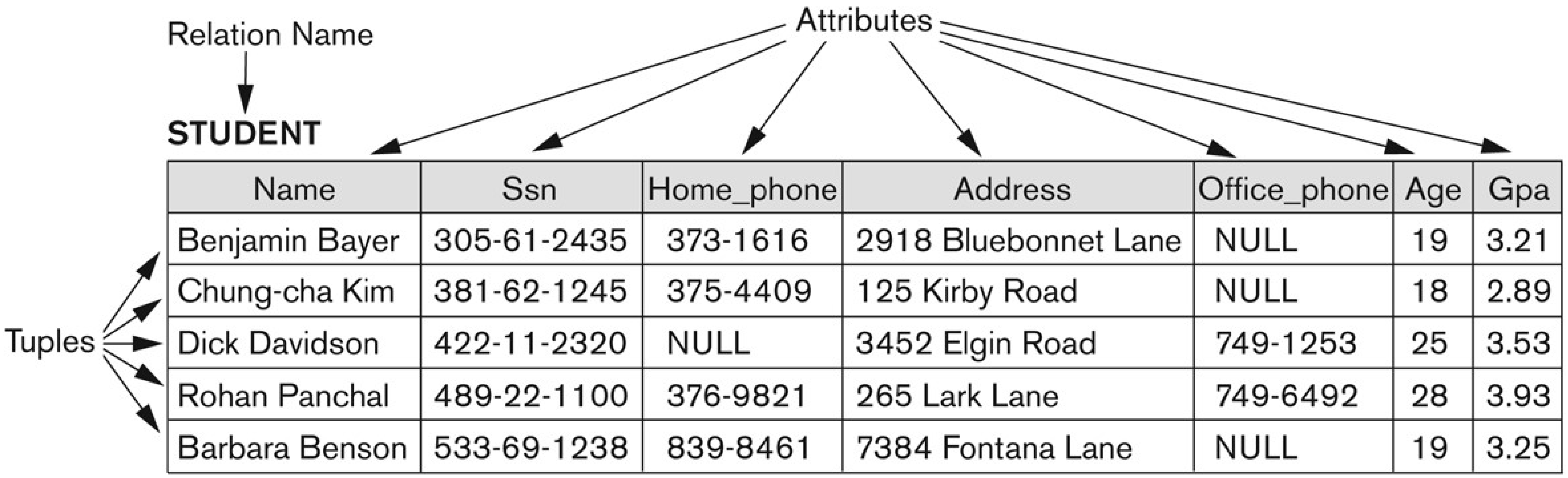
Informal Description

- A relation looks like a table (rows x columns) of values

- A relation contains a set of rows (tuples) and each column (attribute) has a column header that gives an indication of the meaning of the data items in that column
  - Associated with each attribute of a relation is a set of values (domain)
  - Movies(title:string, year:integer, length:integer)

- The data elements in each row (tuple) represent certain facts that correspond to a real-world entity or relationship

Example of a Relation



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

Figure 5.1
The attributes and tuples of a relation STUDENT.

Regular structure makes it easy to be manipulated
What are the operations?

Relational Data Model

- ***Basic Structure***
  - Records
    - Each row/tuple in a relation is a record (an entity)
    - Each attribute in a relation corresponds to a particular field of a record
  - Sample Relational DB **Schema**:

Customer

cname	C#	address

Parts

P#	pname	cost

Orders

order#	C#	P#	quantity

Relational Data Model

- A corresponding DB Instance:

Customer

cname	C#	address
John	41256	8 Blue St., LA
Mary	56437	6 Red Ave, SF
Joe	23986	12 Pink Rd, NY

Parts

P#	pname	cost
301	widget	25,000
111	gadget	17,500
507	screw	5,900

Orders

order#	C#	P#	quantity
21	41256	301	15
26	41256	111	7
27	56437	301	11
32	23986	507	18
...			

Definition Summary

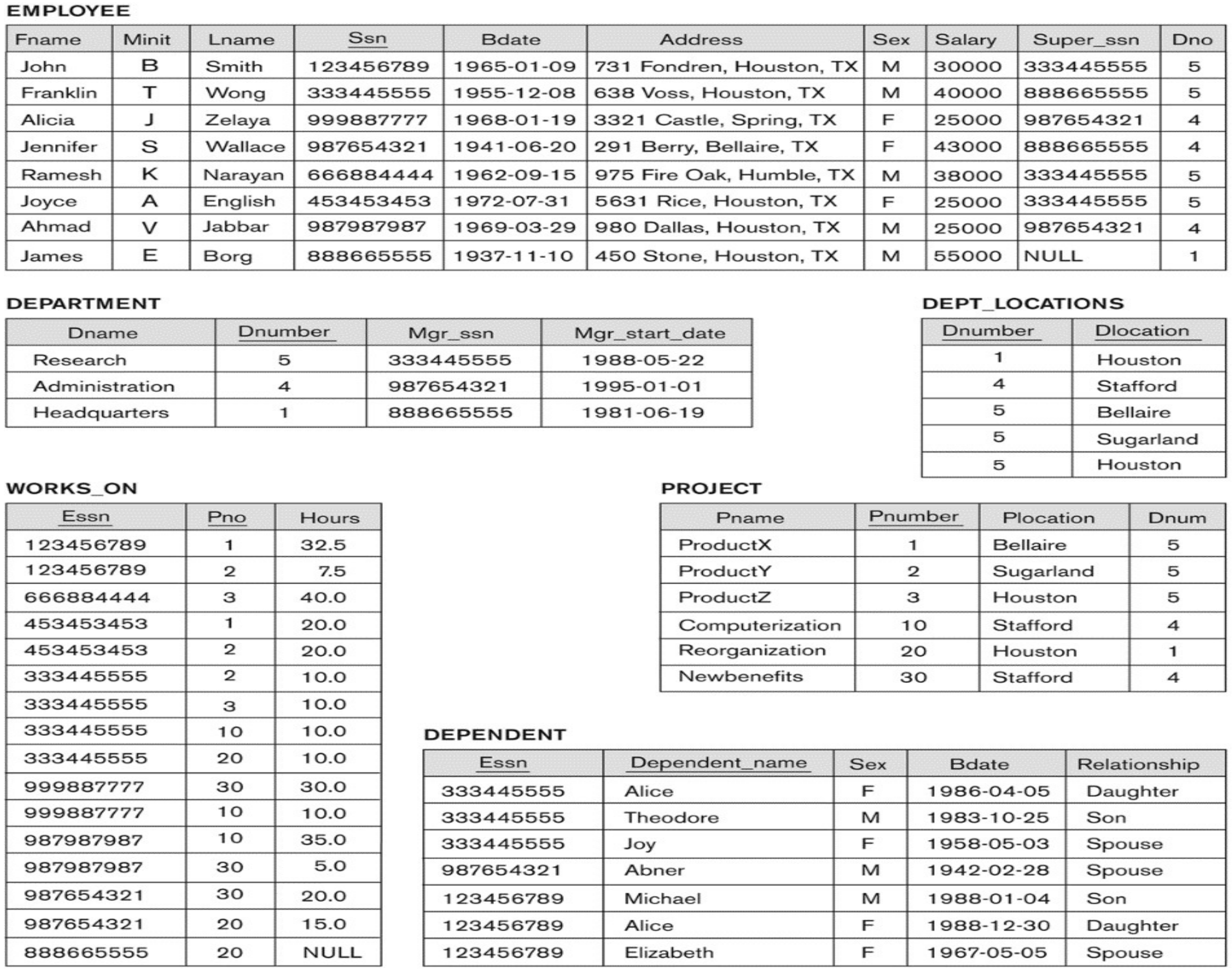
Informal Terms	Formal Terms
Table	Relation
Column Header	Attribute
All possible Column Values	Domain
Row	Tuple
Table Definition	Schema of a Relation
Populated Table	State of the Relation

Populated Relation State

- Each *relation* has many records/tuples in its current relation state/instance
- Whenever the database is changed, a new state arises
- Basic operations for changing the database:
  - INSERT a new tuple in a relation
  - DELETE an existing tuple from a relation
  - MODIFY an attribute of an existing tuple

Populated database state for COMPANY

Figure 5.6
One possible database state for the COMPANY relational database schema.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

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
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

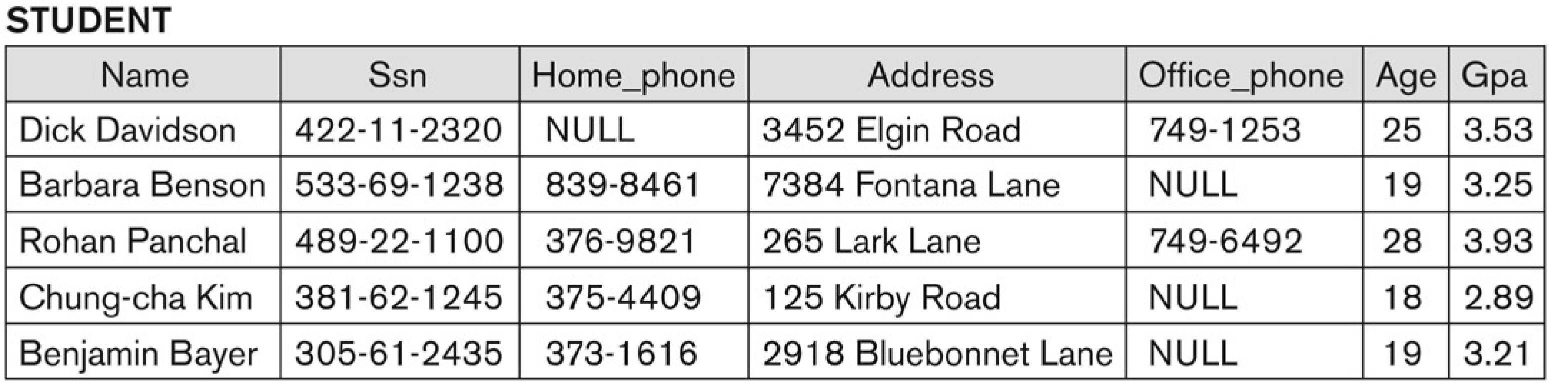
Characteristics Of Relations

- Ordering of tuples in a relation:
  - The tuples are *not considered to be ordered*, even though they appear to be in a tabular form (may have different presentation orders)

- Ordering of attributes in a relation schema R (and of values within each tuple):
  - We consider the attributes in R(A1, A2, ..., An) and the values in t=<v1, v2, ..., vn> to be ordered
    - Example: t= { <name, "John" >, <SSN, 123456789> }
    - This representation may be called as "self-describing"

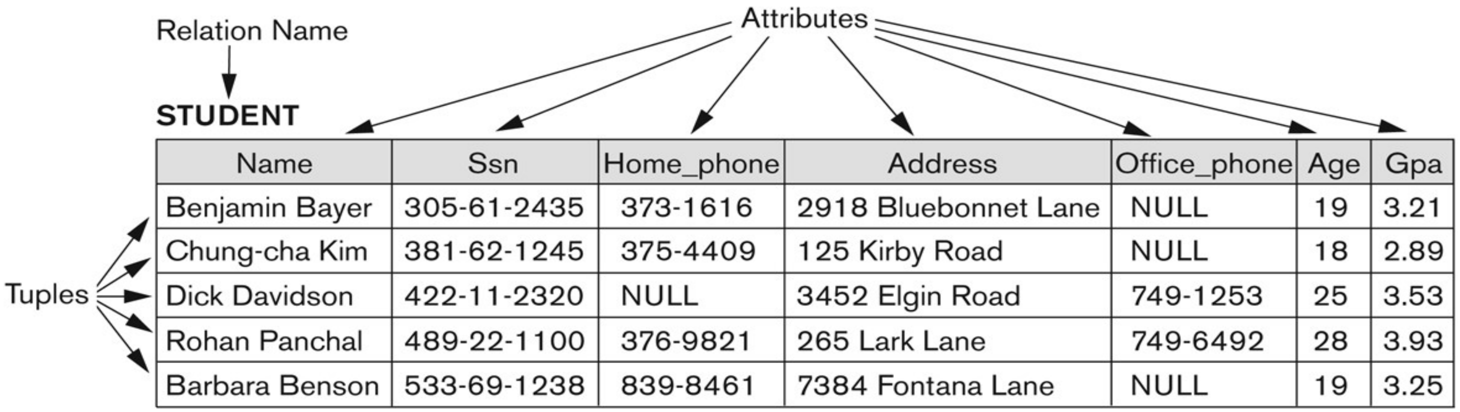
Same state with different order of tuples

Figure 5.2
The relation STUDENT from Figure 5.1 with a different order of tuples.



STUDENT

Name	Ssn	Home_phone	Address	Office_phone	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
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Chung-cha Kim	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



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

Figure 5.1
The attributes and tuples of a relation STUDENT.

Characteristics Of Relations

- Values in a tuple:
  - All values are considered atomic (indivisible)
    - Basic unit for manipulation (add or change)
  - Each value in a tuple must be from the domain (set of values) of the attribute for that column
  - A special null value is used to represent values that are unknown or not available or inapplicable in certain tuples

- We refer to the attribute values of a tuple t by:
  - t[Ai] or t.Ai: the value vi of attribute Ai for tuple t

From E/R Diagrams to Relations

- Converting an E/R design to a relational schema (an approximation approach):
  - Turn each entity set into a relation with the same set of attributes
  - Replace a relationship by a relation whose attributes are the keys for the connected entity sets
  - Note: weak entity sets cannot be translated straightforward to relations

Example: ER of Company

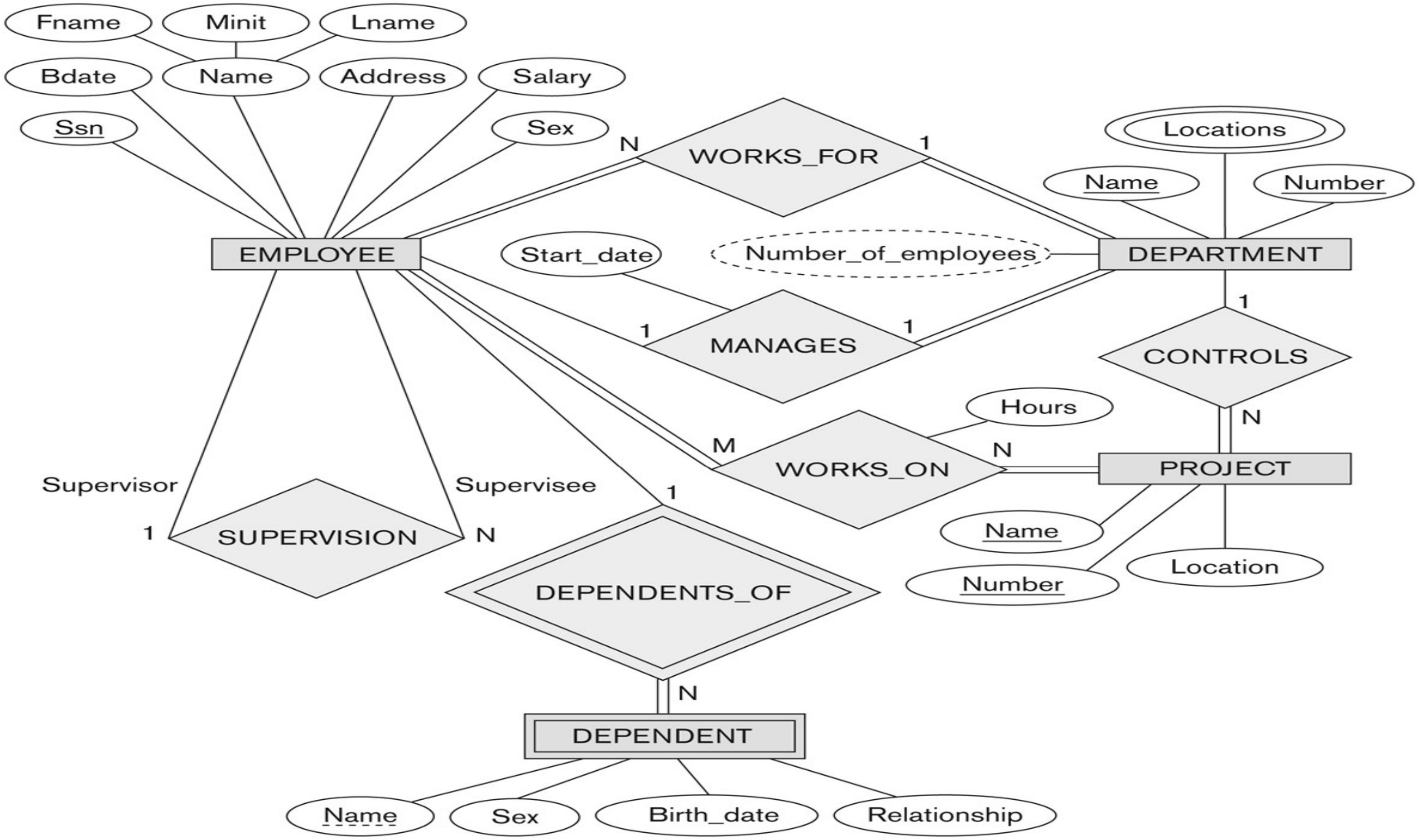
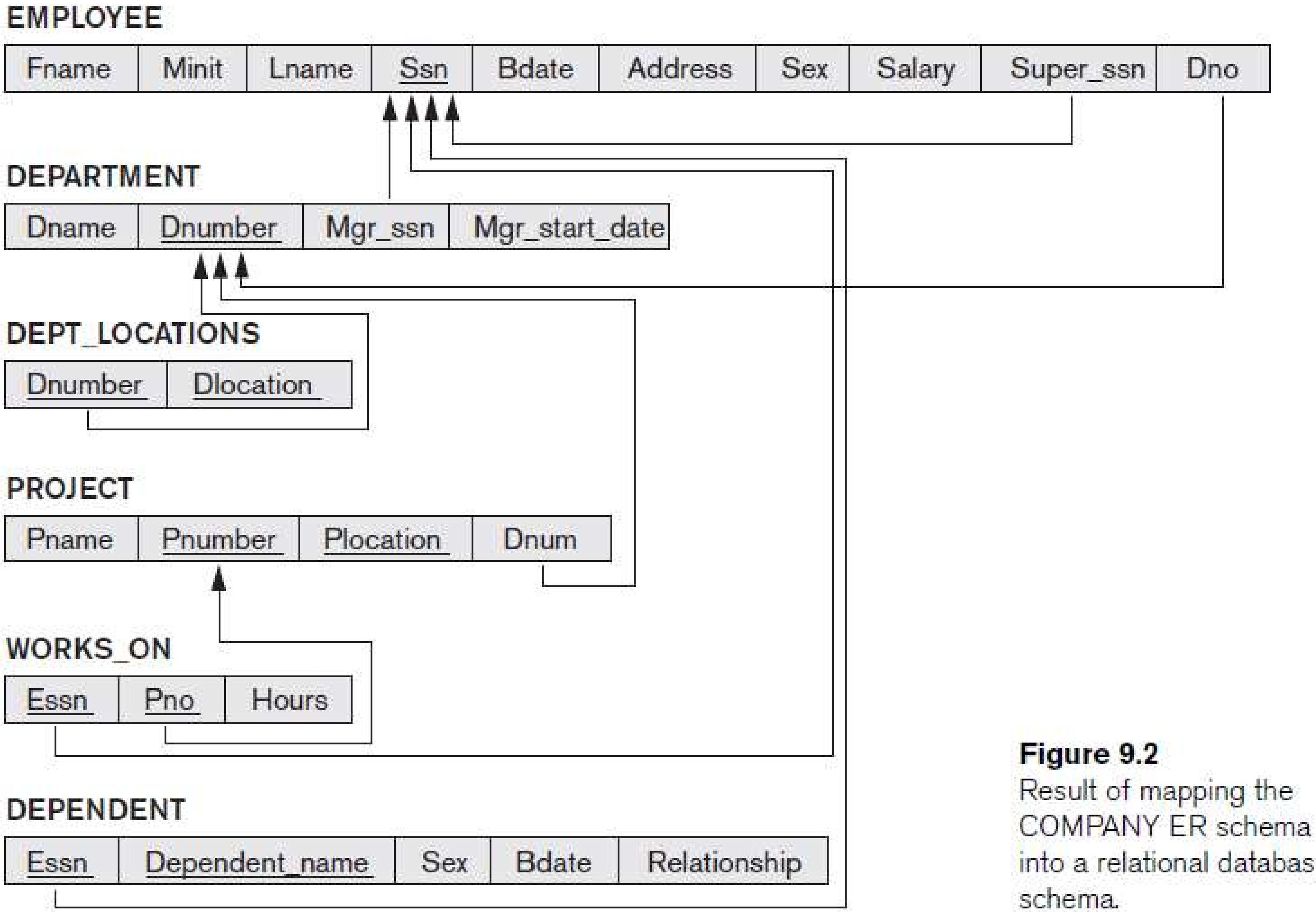


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.



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

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

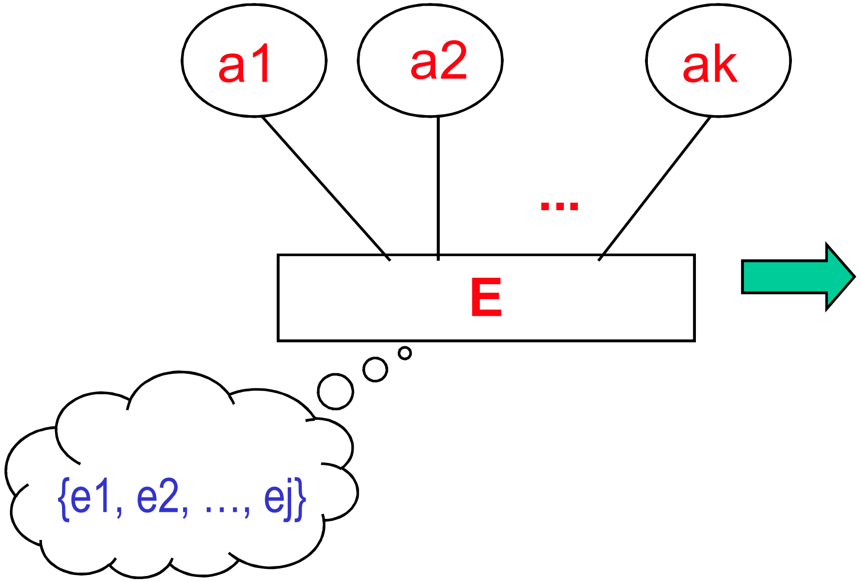
ER-to-Relational Mapping Algorithm

- COMPANY database example
  - Assume that the mapping will create tables with simple single-valued attributes

- Step 1: Mapping of Regular (strong) Entity Types
  - For each regular entity type, create a relation *R* that includes all the simple attributes of *E*
  - Choose one of the key attributes E as the primary key for *R*
  - Called entity relations
    - Each tuple represents an entity instance

ER to Relations (Entity Sets)

- Mapping ER Diagrams into tables (relations)
  - Representation of (Strong) Entity Sets



E

	a1	a2	…	ak
e1	xx	yy	…	zz
e2				
:				

tuples

(entities)

ER-to-Relational Mapping Algorithm

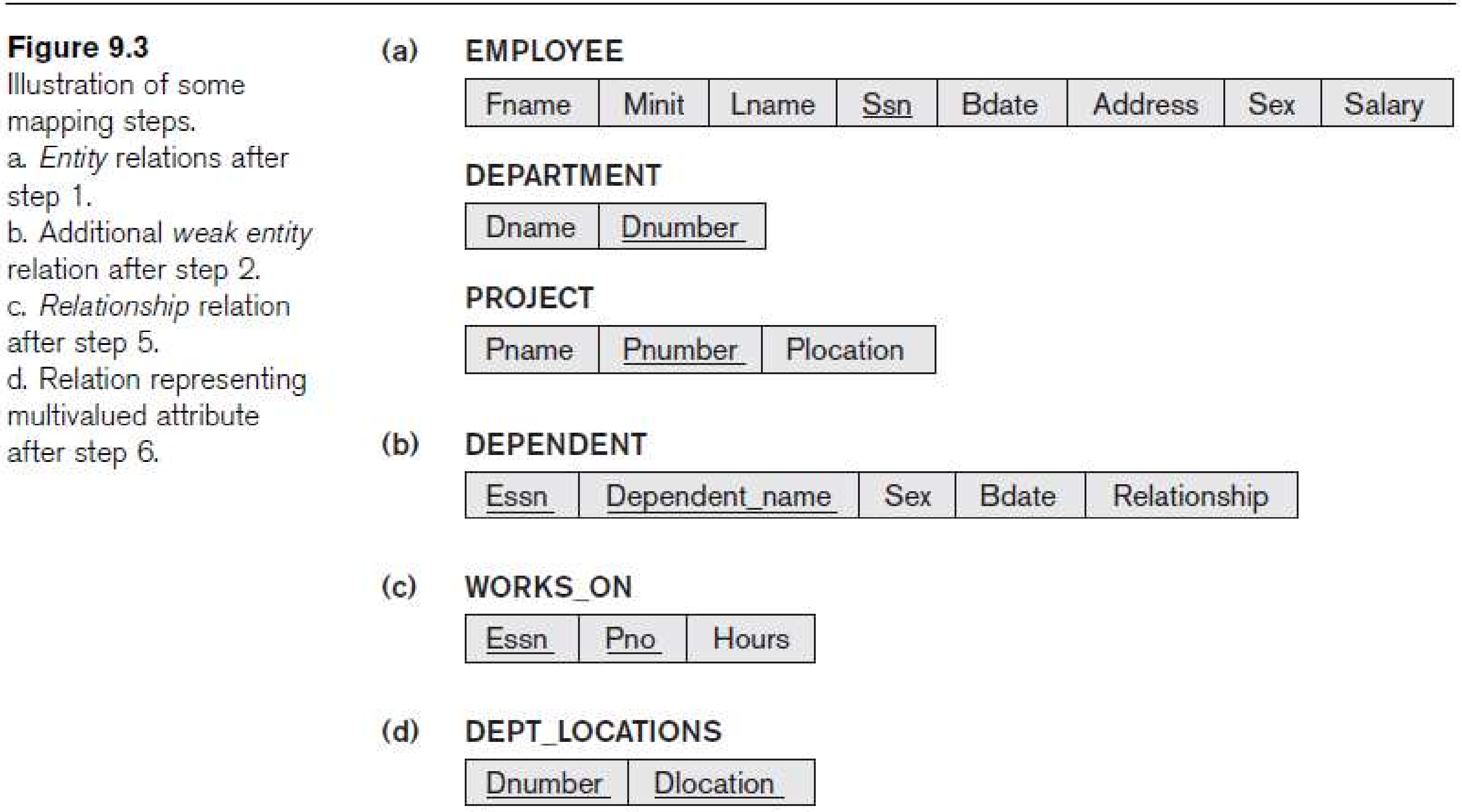


Figure 9.3
Illustration of some mapping steps.
a. *Entity* relations after step 1.
b. Additional *weak entity* relation after step 2.
c. *Relationship* relation after step 5.
d. Relation representing multivalued attribute after step 6.

(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

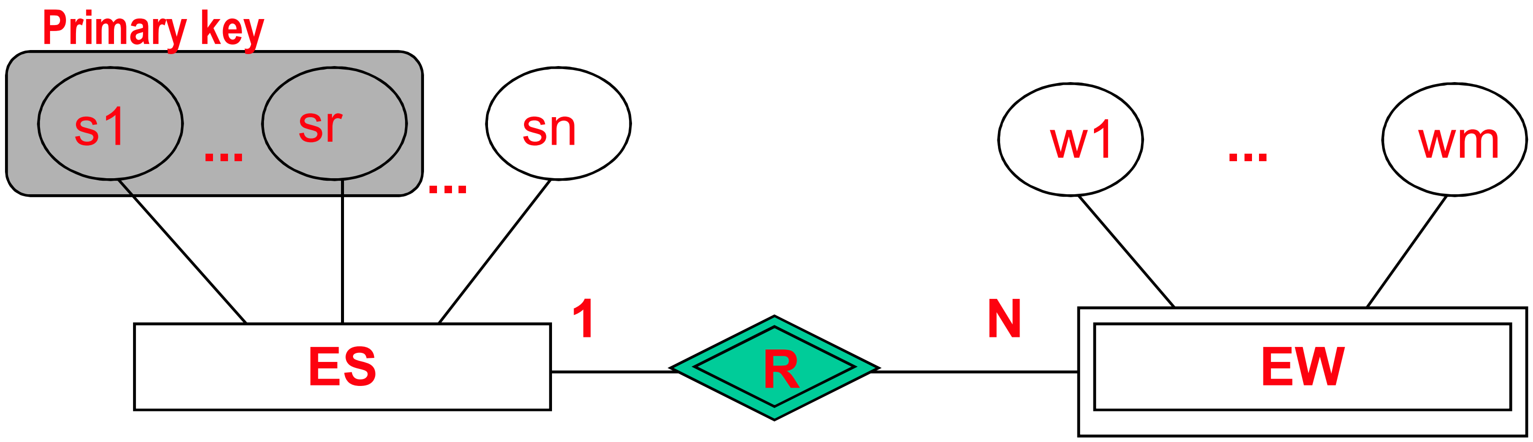
Dnumber	Dlocation

ER-to-Relational Mapping Algorithm

- Step 2: Mapping of Weak Entity Types
  - For each weak entity type, create a relation *R* and include all simple attributes of the entity type as attributes of *R*
  - Include primary key attribute of owner as foreign key attributes of *R*
  - The primary key of R is the combination of the primary key of the owner and the partial key of the weak entity type
  - E.g., Essn and Dependent_Name

ER to Relations (Weak Entity Set)

- Representation of Weak Entity Sets



ES

s1	sr	...	sn

EW

w1	…	wm	s1	…	sr

ER-to-Relational Mapping Algorithm

- Step 3: Mapping of Binary 1:1 Relationship Types
  - For each binary 1:1 relationship type
    - Identify relations that correspond to entity types participating in *R*
  - Possible approaches:
    - Foreign key approach
    - Merged relationship approach
    - Cross reference or relationship relation approach

ER-to-Relational Mapping Algorithm

- Foreign key approach (S –T)
  - Choose one of the relations S and include as a foreign key in S the primary key of T
  - Include all simple attributes as attributes of S
  - E.g., MANAGES (Mgr_ssn and Mgr_start_date) into DEPARTMENT

ER-to-Relational Mapping Algorithm

- Merged relationship approach
  - To merge the two entity types and the relationship into a single relation

- Cross reference or relationship relation approach
  - 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
  - The relation R will include the primary key attributes of S and T as foreign keys to S and T
  - The primary key of R will be one of the two foreign keys

ER-to-Relational Mapping Algorithm

- Step 4: Mapping of Binary 1:*N* Relationship Types
  - For each regular binary 1:*N* relationship type
    - Identify relation that represents participating entity type at *N*-side of relationship type S
    - Include primary key of other entity type as foreign key in *S*
    - Include simple attributes of 1:*N* relationship type as attributes of *S*
    - E.g., for WORK_FOR, we include the primary key Dnumber of the DEPARTMENT as foreign key in EMPLOYEE

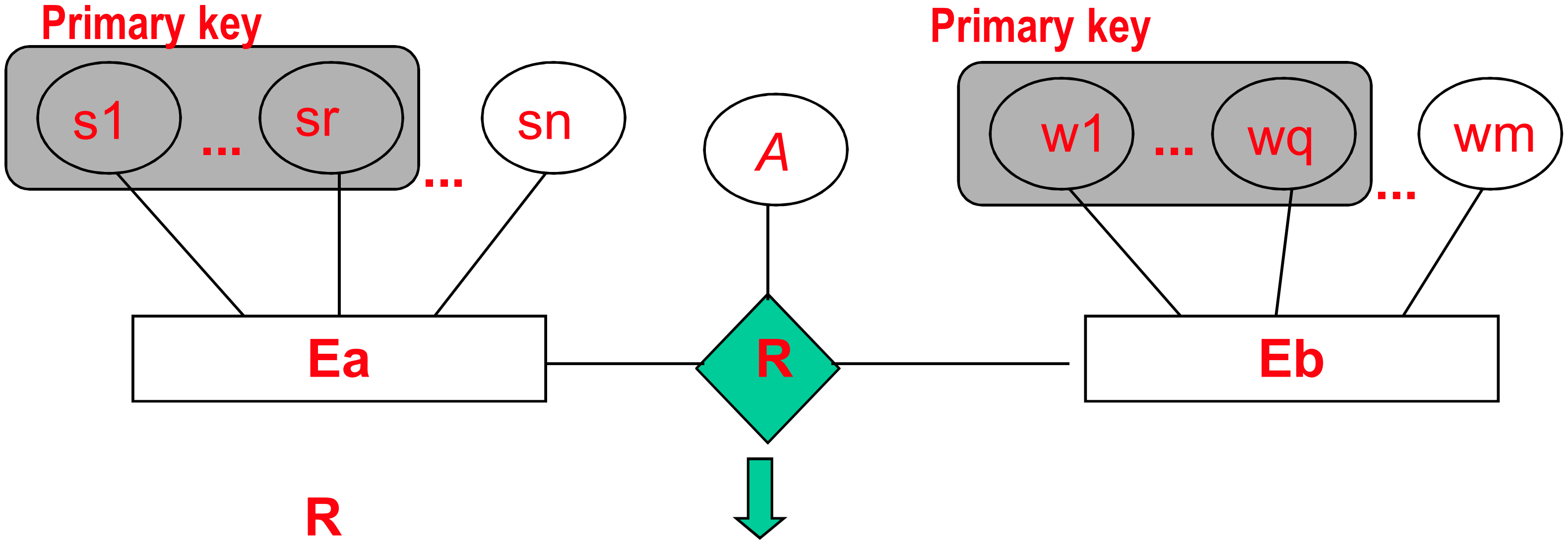
  - Alternative approach

ER-to-Relational Mapping Algorithm

- Step 5: Mapping of Binary *M*:*N* Relationship Types
  - For each binary *M*:*N* relationship type
    - Create a new relation *S*
    - Include as primary key of participating entity types as foreign key attributes in *S*
    - Their combination forms the primary keys of the relations
    - Include any simple attributes of *M*:*N* relationship type
    - E.g., For WORKS_ON, we include the primary keys of PROJECT and EMPLOYEE as foreign keys in WORK_ON and rename them Pno and Essn, respectively

ER to Relations (Many-to-Many)

- Representation of Many to Many Relationship Sets



R

s1	…	sr	w1	…	wq	*A*

ER-to-Relational Mapping Algorithm

- Step 6: Mapping of Multivalued Attributes
  - For each multivalued attribute A
    - Create a new relation *R*
    - Primary key of *R* is the combination of *A* and *K* (the primary key attribute) of the relation that represents the entity type or relationship that has A as a multivalued attribute
    - If the multivalued attribute is composite, include its simple components
    - E.g., we create a relation DEPT_LOCATION. The attribute Dlocation represents the multivalued attributes LOCATIONS of DEPARTMENT
    - The primary key of DEEPT_LOCATIONS is the

ER-to-Relational Mapping Algorithm

- Step 7: Mapping of *N*-ary Relationship Types
  - For each *n*-ary relationship type *R*
    - Create a new relation *S* to represent *R*
    - Include primary keys of participating entity types as foreign keys
    - Include any simple attributes as attributes
    - The primary key of S is a combination of all foreign keys that reference the relations representing the participating entity types

Discussion and Summary of Mapping for ER Model Constructs

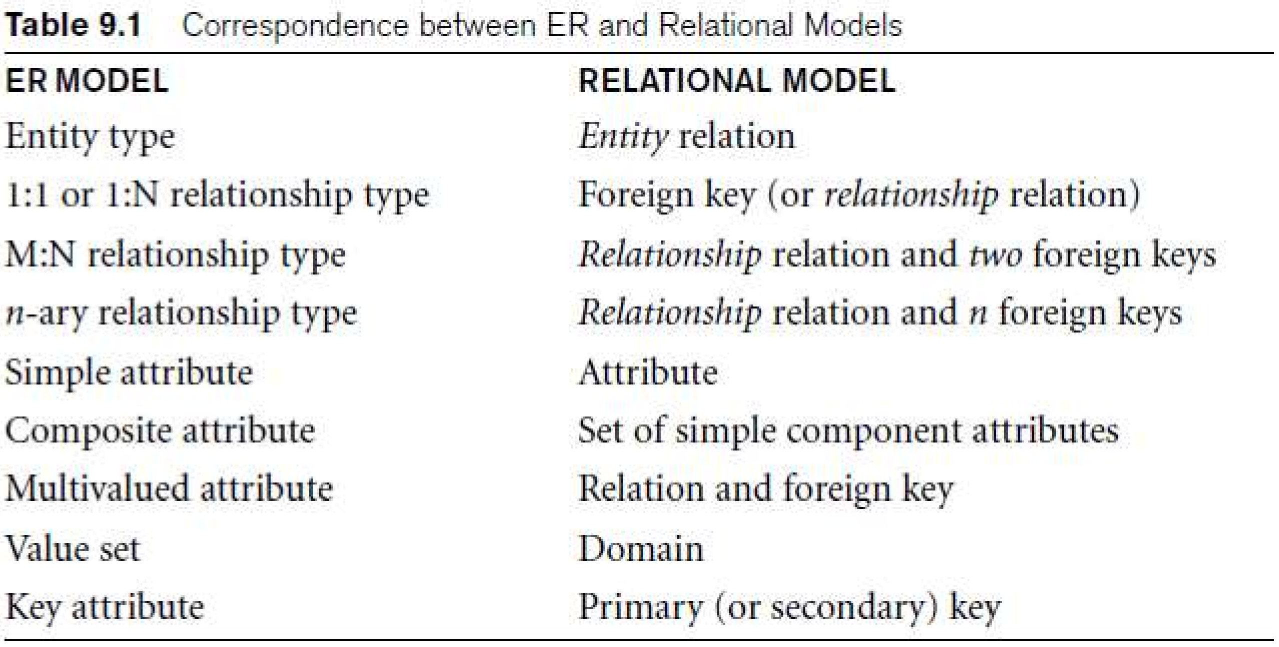


Table 9.1 Correspondence between ER and Relational Models

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

References

- 6e
  - Ch. 3 p.55 – 75
  - Ch. 8, p.270 – 278