§ 6.5.3 Weak Entity Sets

- An entity set that does not have a primary key is referred to as a *weak entity set*
  - e.g. *section(sec_id, semester, year) in* Fig.6.14 in next slide, and Fig. 7.0.10
- A weak entity set E_1 can ***only*** be distinguished through another (strong) entity set E_2, called **identifying（标识）/owner（属主） entity set**, which has association/ relationship **R** with E_1
  - E_2 owns E_1 , E_1 must be related to E_2 via an instance of R, called the *identifying* relationship（标识性联系）.
  - R is ***many-to-one*** (or ***one-to-one***) from the weak entity set $\mathbf{E_1}$ **to** the identity set $\mathbf{E_2}$

Weak Entity Sets (cont.)

- E_1 is of ***total participation in*** the relationship set R
- R is depicted using a *double diamond*
- weak entity set is depicted as *double rectangles*.

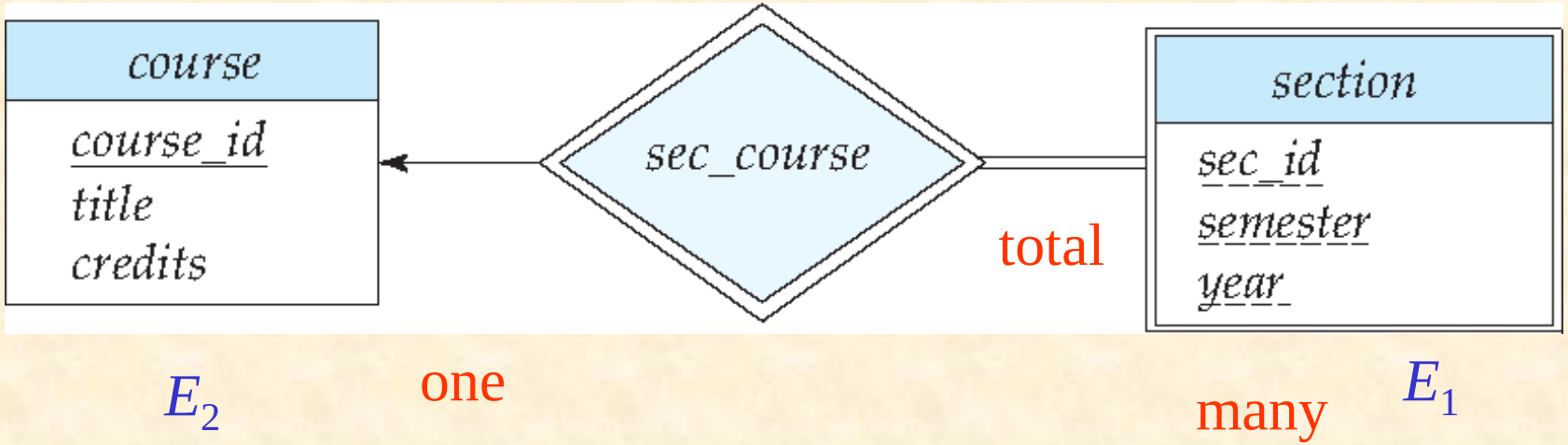


Fig.6.14 E-R diagram with a weak entity set

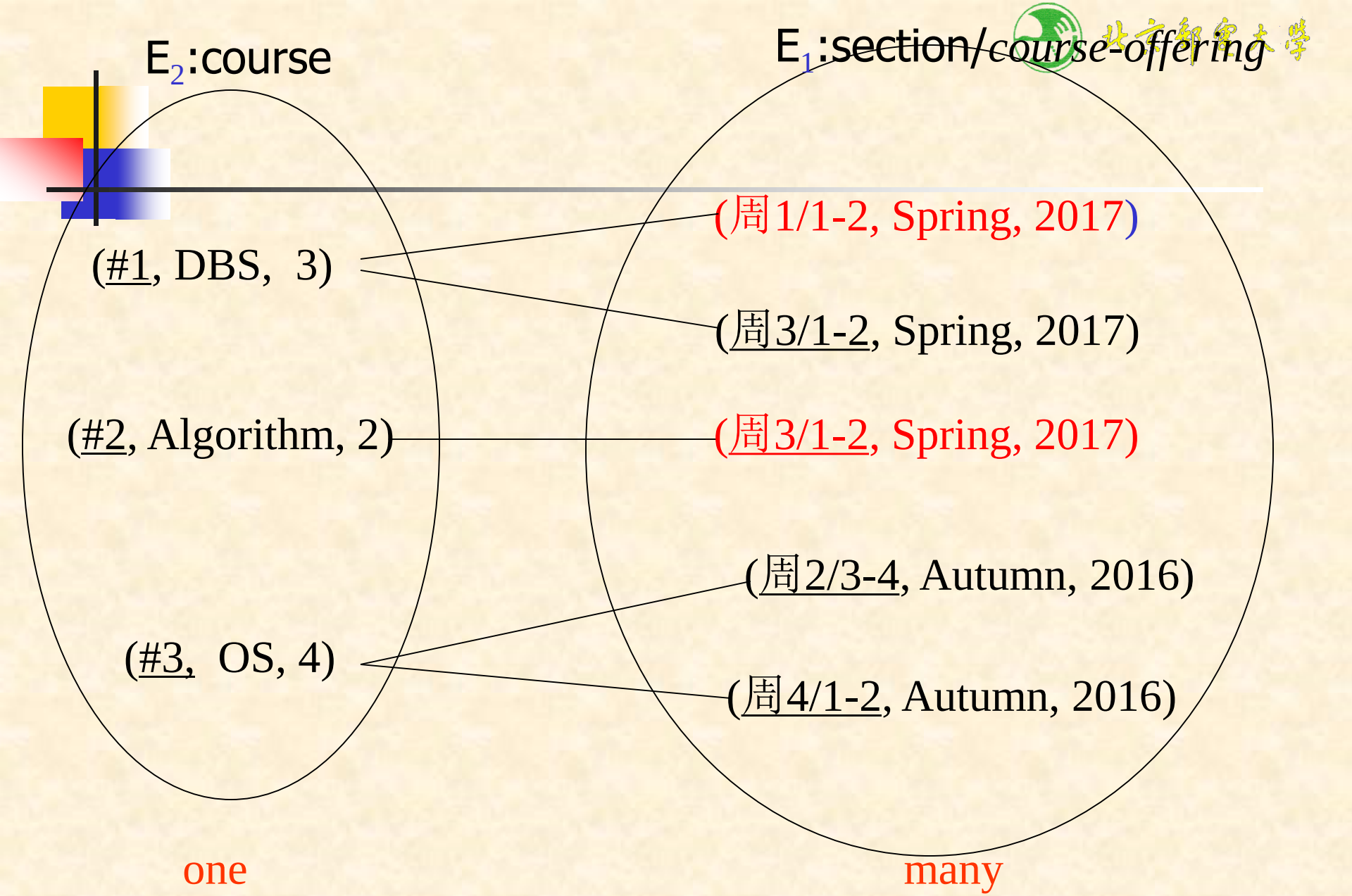


Fig.7.0.10 A example of weak entity set

Weak Entity Sets (cont.)

- The *discriminator （分辨符）(or partial key)* of a weak entity set is the set of attributes that distinguishes among all those entities in the weak entity set that depending on a particular strong entity
  - e.g. ***sec_id, semester, year***
  - the discriminator of a weak entity set is underlined with a *dashed line*（虚线）

Weak Entity Sets (cont.)

- The ***primary key of a weak entity set*** is formed by the primary key of the *strong entity set* on which the weak entity set is existence dependent, plus the weak entity set's discriminator
  - primary key for weak entity set E_1

    = discriminator $\cup$ primary_key(E_2) .
  - e.g. sec_id, semester, year, course_id

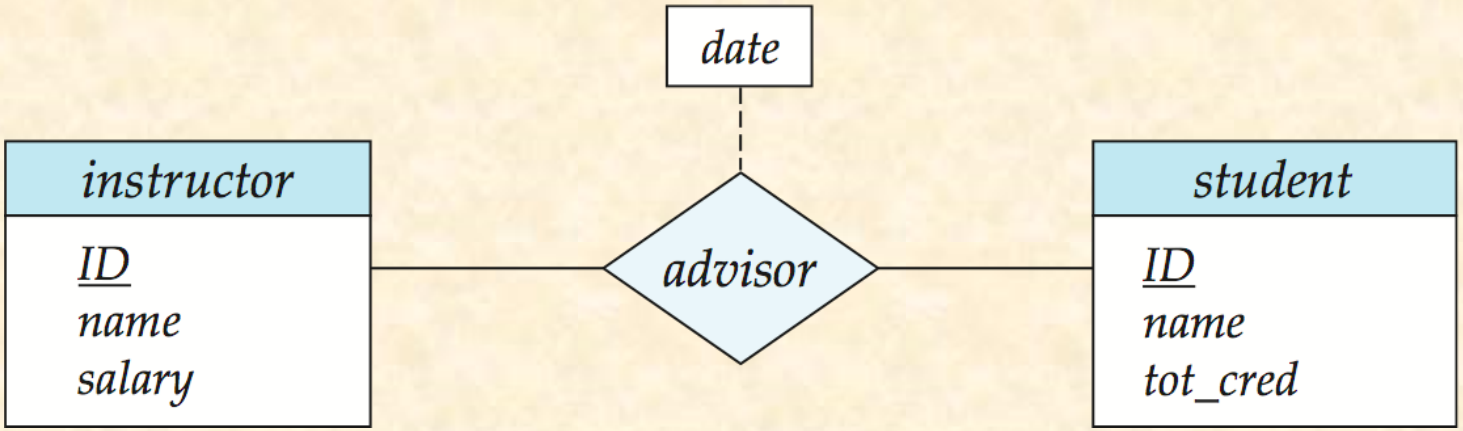
- Weak entity set E_1 with several identifying entity sets E_2, E_3, .., E_n, primary key for weak entity set E_1 =

  discriminator $\cup$ primary_key(E_2) $\cup$ … primary_$\cup$key(E_n) .

E-R diagrams

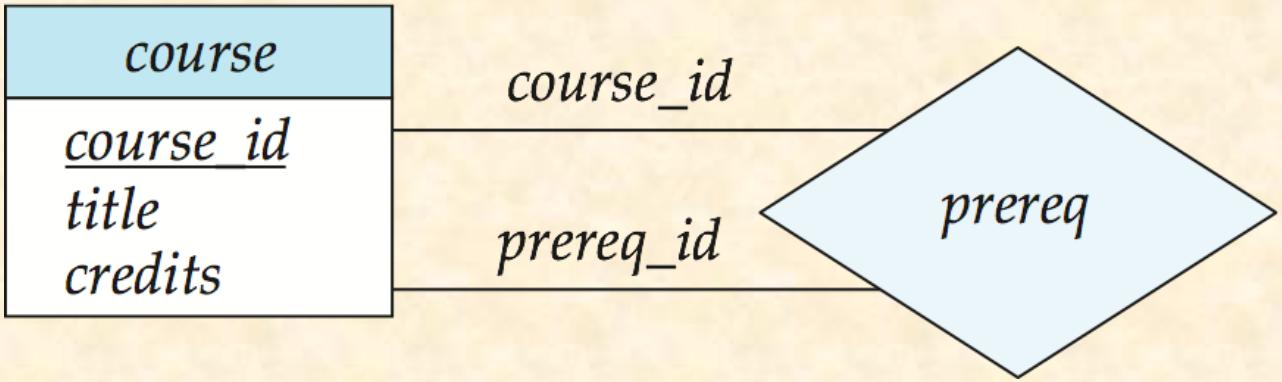
- ***Rectangles***（矩形） represent entity sets
- ***Diamonds***（菱形） represent relationship sets
- ***Lines*** link attributes to entity sets and entity sets to relationship sets.
- ***Ellipses***（椭圆） represent attributes
  - ***double ellipses*** represent multivalued attributes
  - ***Dashed***（虚线） ***ellipses*** denote derived attributes（派生属性）
- ***Underline*** indicates primary key attributes
- Double rectangles represent weak entity sets

Relationship Sets with Attributes



Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a “role” in the relationship
- The labels “*course_id*” and “*prereq_id*” are called **roles**.



Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set.

One-to-Many Relationship

Total and Partial Participation

Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set

participation of *student* in *advisor* relation is total

- every *student* must have an associated instructor

Partial participation: some entities may not participate in any relationship in the relationship set

Example: participation of *instructor* in *advisor* is partial

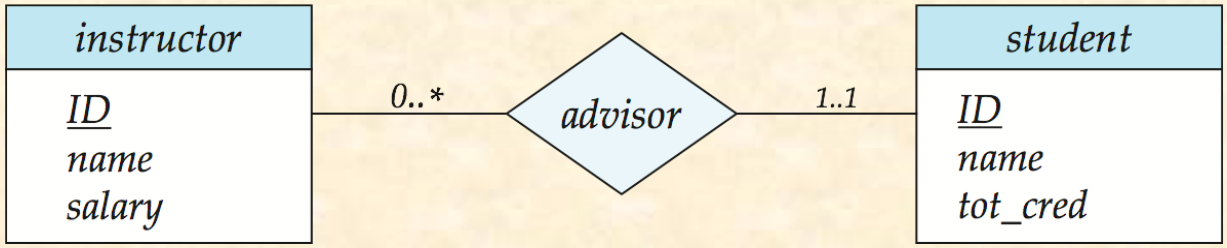
Cardinality Limits

A line may have an associated minimum and maximum cardinality, shown in the form *l..h*, where *l* is the minimum and *h* the maximum cardinality

A minimum value of 1 indicates total participation.

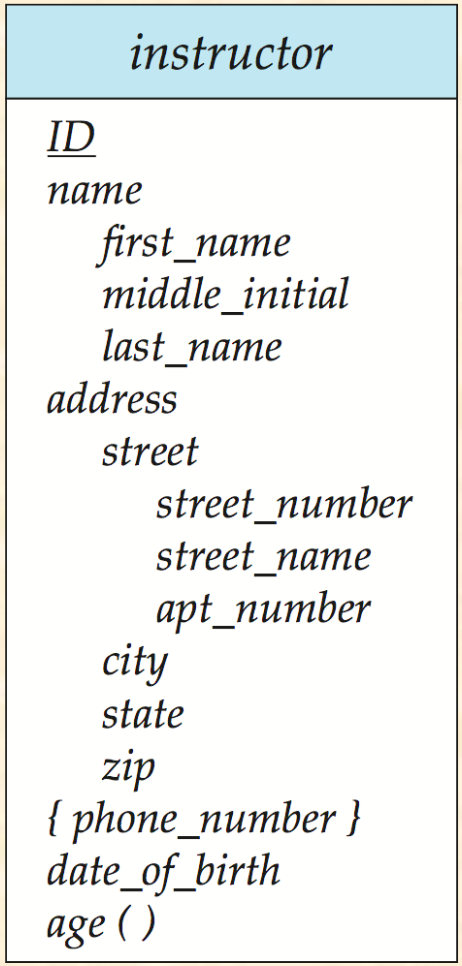
A maximum value of 1 indicates that the entity participates in at most one relationship

A maximum value of * indicates no limit.



Instructor can advise 0 or more students. A student must have 1 advisor; cannot have multiple advisors

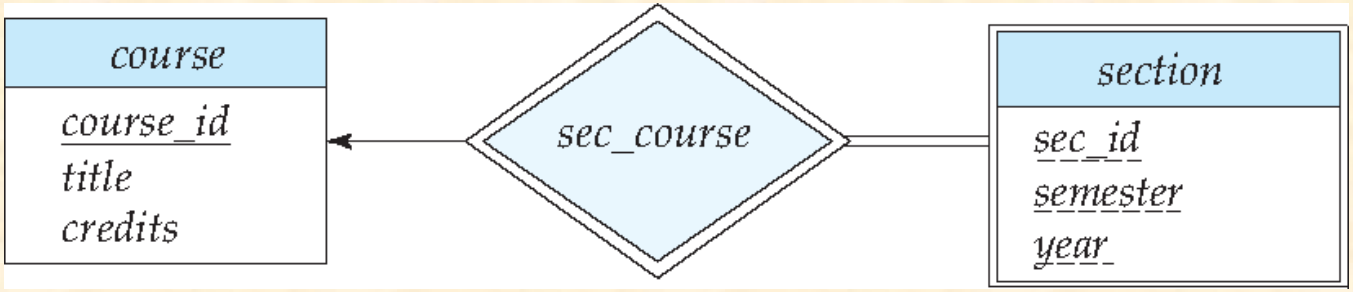
Entity with Complex Attributes



Expressing Weak Entity Sets

- In E-R diagrams, a weak entity set is depicted via a double rectangle.
- We underline the discriminator of a weak entity set with a dashed line.
- The relationship set connecting the weak entity set to the identifying strong entity set is depicted by a double diamond.
- Primary key for *section* :

  (*course_id*, *sec_id*, *semester*, *year*)



E-R diagrams (cont.)

- 标注规则！！！
  - 由*联系*发出的有向箭头指向one 所对应的实体，无向线段指向many端所对应的实体。
  - 只有在many-to-many的情况下，E-R图中联系与实体间的线段 全部是无向线段，其他三种情况下必出现有向线段

§ 6.7 E-R Design Issues

6.7.1 Use of entity sets vs. attributes

- Mapping "***thing/object***" in real world into ***entity*** sets or ***attribute*** sets
  - e.g. for *instructor* and *phone* objects in real world
    - model 1. entity:

      instructor(ID, name, salary, phone-number)

      implying that employees have precisely one phone number each, unless *phone-number* is multivalued;

      how to represent the other properties of "phone" object ?

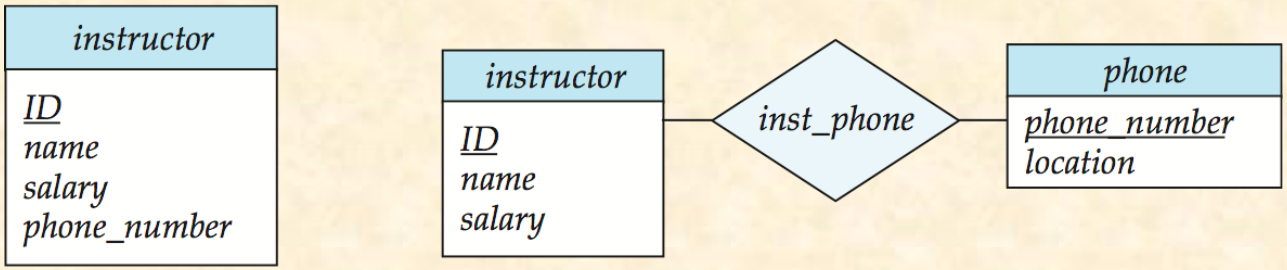


Fig.6.23 Alternatives for adding phone to the instructor entity set

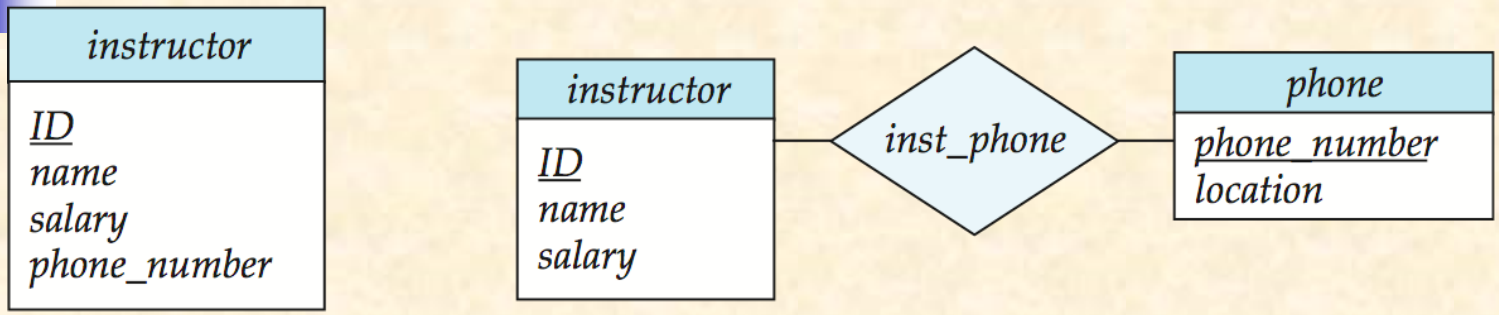


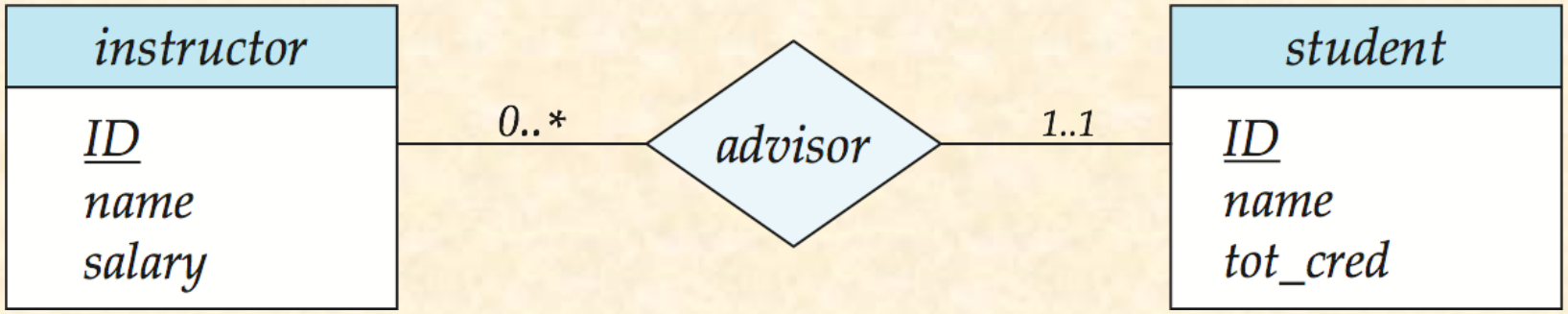
Fig.6.23 Alternatives for adding phone to the instructor entity set

- model 2. entity: *instructor*(*ID*, *name*, *salary*)
  phone(*phone-number*, *location*, …)
  relationship: *inst-phone*

- Treating *phone* as an entity is more general, more information about *phone* can be modeled, e.g.
  - *Location*
  - plus multiple phone numbers

Entity Sets vs. Attributes (cont.)

- Rule1. Not to use the primary-key of an entity set as an attributes of another entity set (to represent *implicitly* association between these two entity sets), it is better to use an relationship set to explicitly show this association
  - e.g. for the entity *student*, do not use *instructor.id* as its attribute, instead of, representing association among *student* and *instructor* using relationship *advisor*



Entity Sets vs. Attributes (cont.)

- Rule2. If ***E*** participate in ***R***, not to designate the primary-key(E) as attributes of ***R***, to avoid *information redundancy*
  - for example, an *ill* modeling of *advisor*

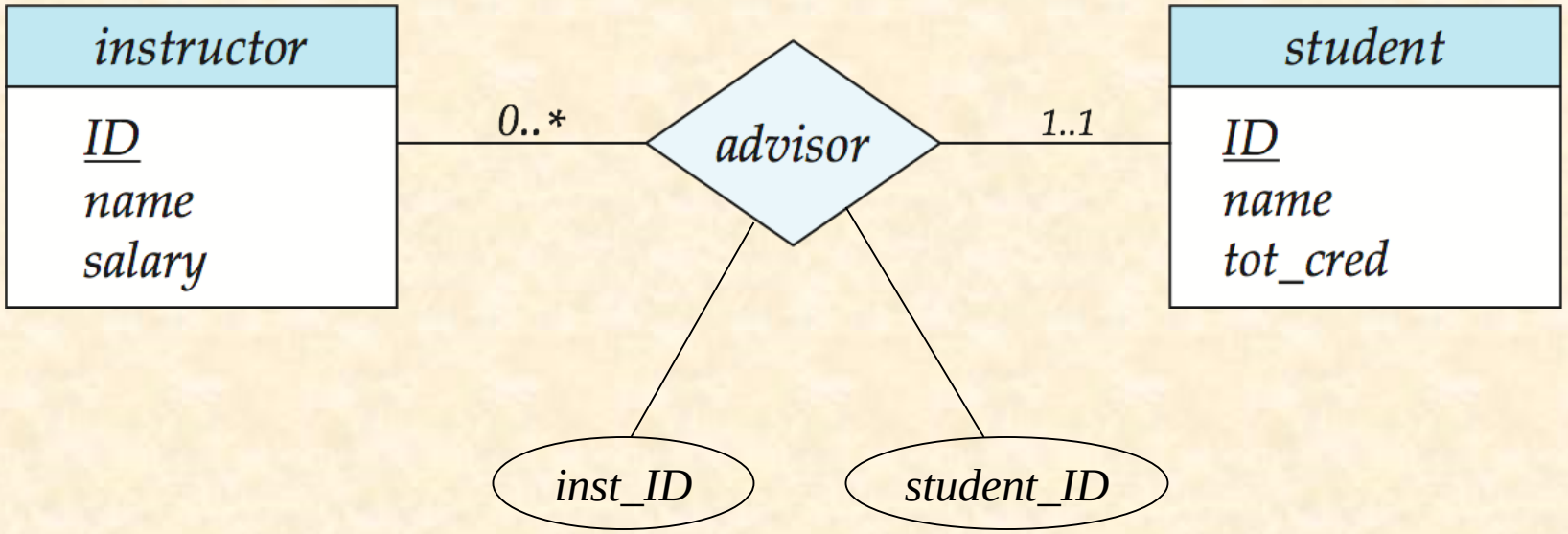
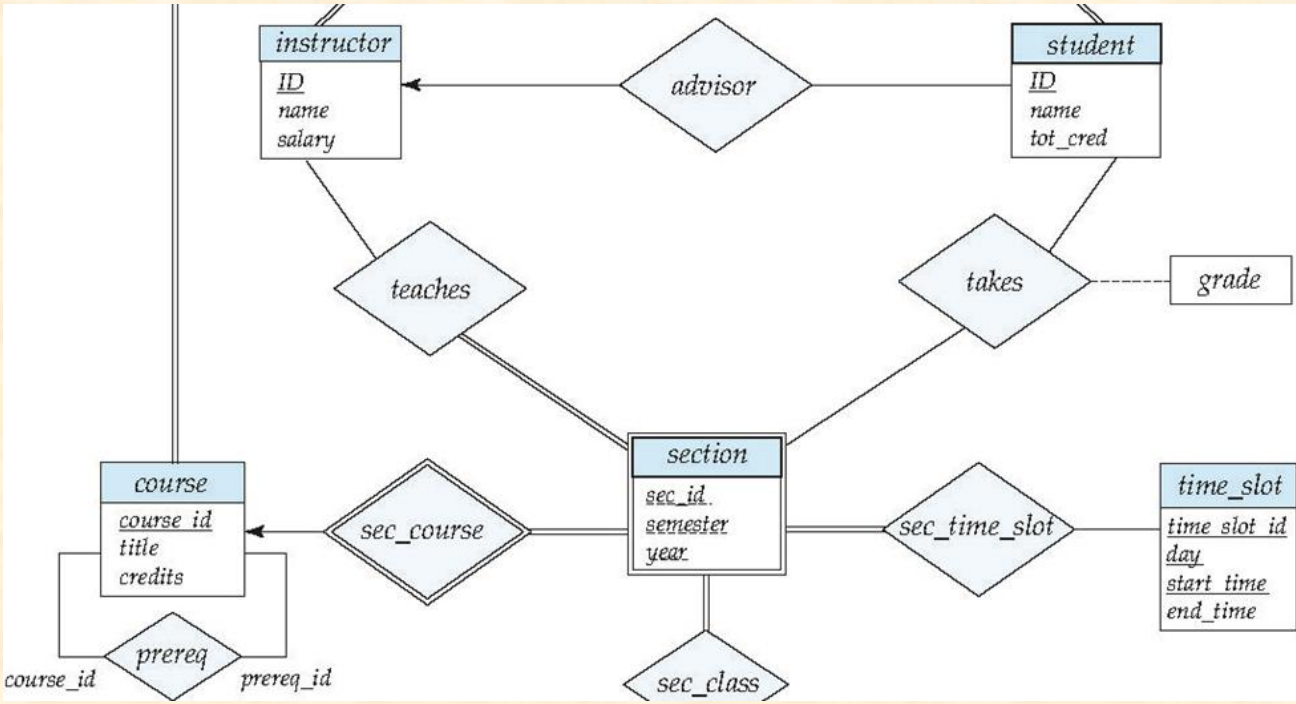


Fig.7.0.7 Anti-example:
Data redundancy in *advisor*

6.7.2 Use of Entity Set vs. Relationship Set

- Rule3. Use the relationship set to describe the *action* occurring between entities
  - e.g. *students* *select* *courses*

- In Fig.6.15 the *takes* relationship is used to model the situation where a *student* takes a *section* of a *course*



Entities vs. Relationship sets

- An alternative is to design a course-registration record, represented by a entity set named *registration*, for each course that each student takes

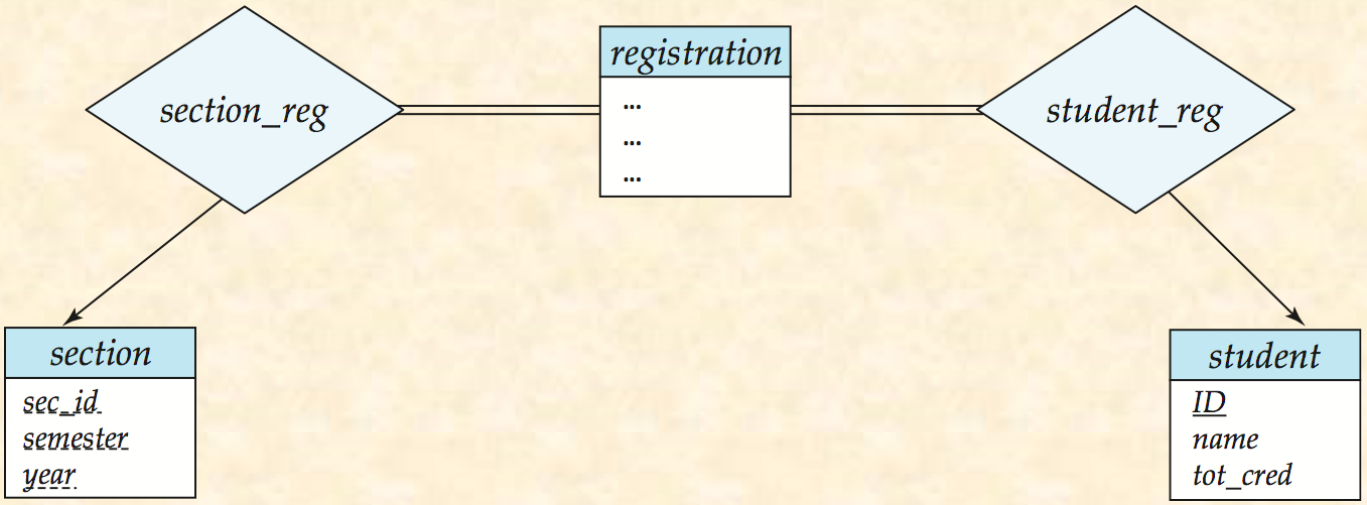


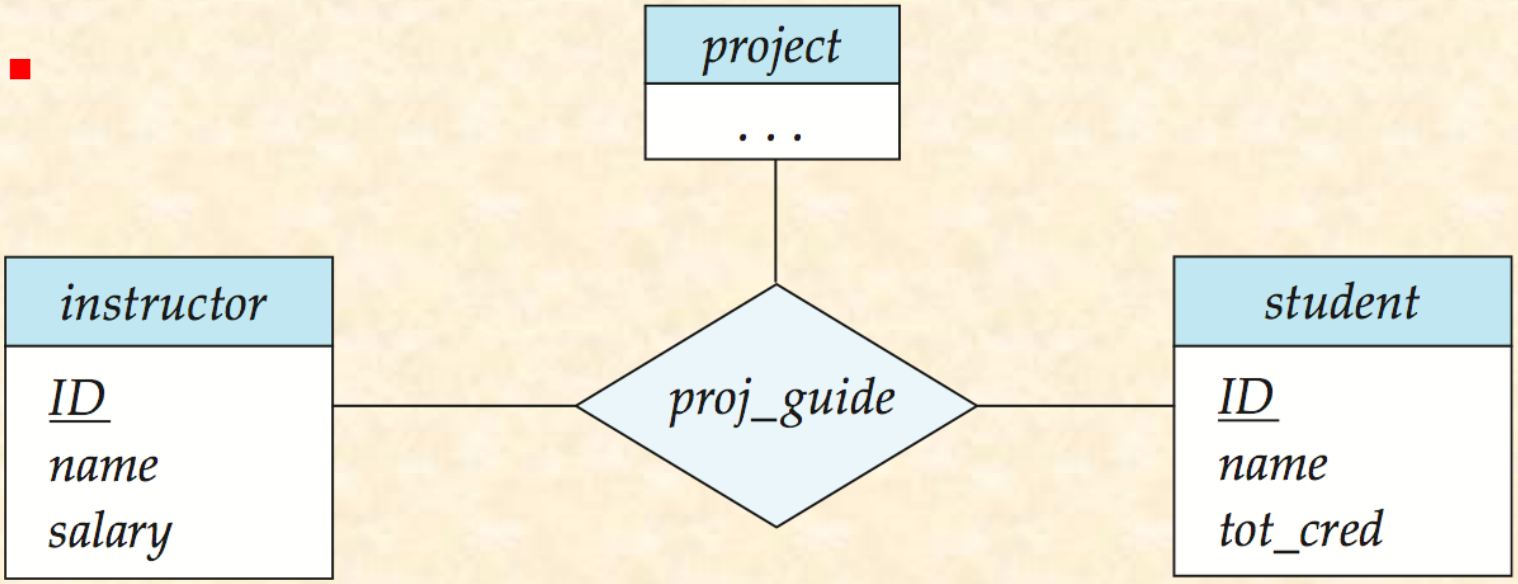
Fig.6.24

- In Fig.6.24, more information can be associated with the entity set *registration* than the relationship set *takes*

6.7.3 Binary vs N-ary Relationship Sets

- Rule4. *Prefer to* using the binary relationship set, and replace a non-binary relationship set by a number of binary relationship set if possible
  - *child-father-mother* is replaced with

    child-father, *child-mother* and *husband-wife*



Binary vs N-ary Relationship Sets

- Translating a non-binary relationship set into several binary relationship sets
  - for 3-ary relationship set $R = \{(a_i , b_i , c_i) \mid a_i \in A , b_i \in B, c_i \in C \}$among entity sets A, B , C, i.e. $R \subseteq A \times B \times C$, replace R by an entity set $E = \{ e_i \}$ and three binary relationship set R_a, R_b, R_c

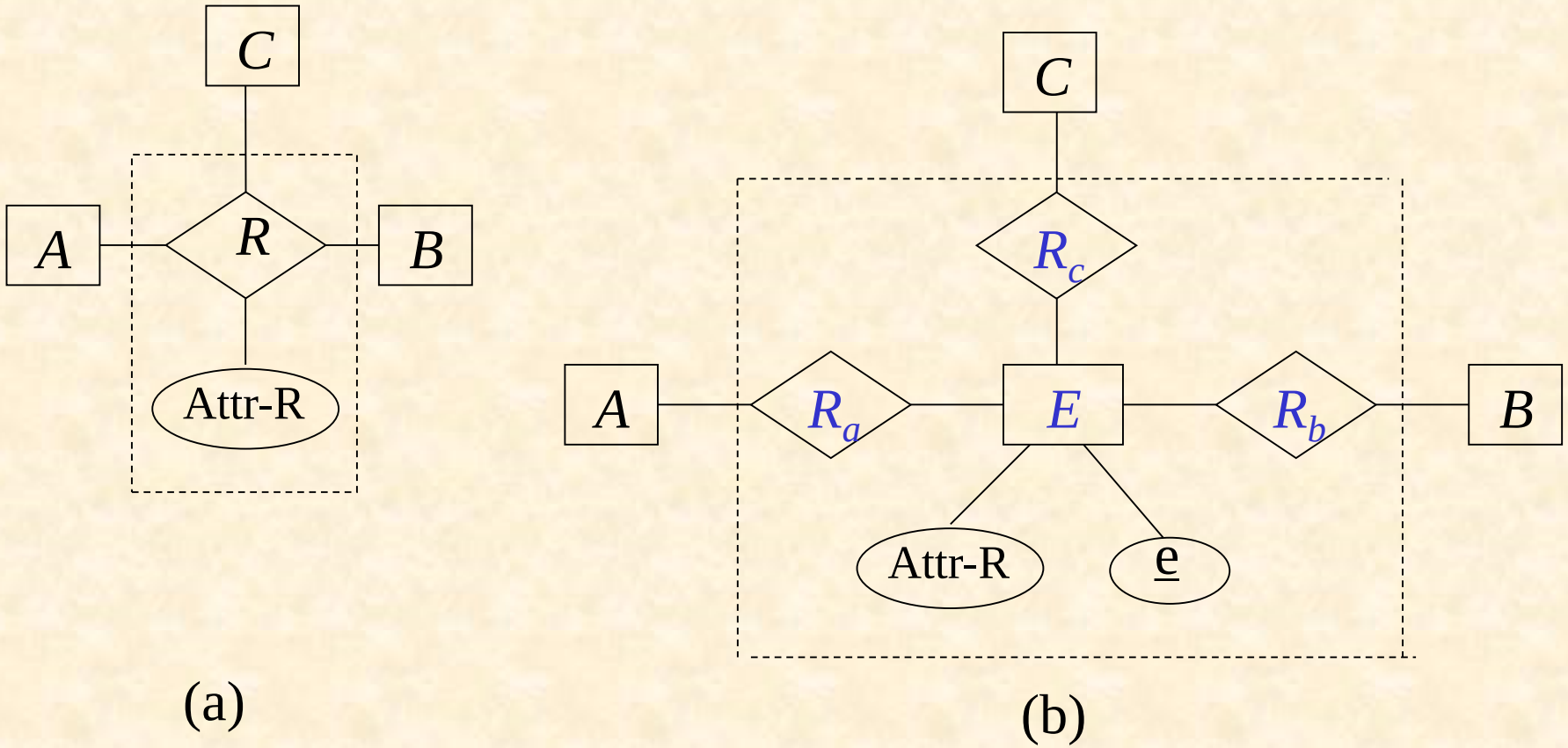


Fig.7.0.8 Non-binary relationship set translating

Binary vs N-ary Relationship Sets (cont.)

- $E = \{ e_i \}$, $|E| = |R|$, i.e. each (a_i , b_i , c_i) in R corresponds to one e_i in E, or $E = \{ e_i \} = R = \{(a_i , b_i , c_i) \}$
- $R_a = \{ (e_i , a_i) \mid e_i \in E , a_i \in A \}$, relating E and A
- $R_b = \{ (e_i , b_i) \mid e_i \in E , b_i \in B \}$, relating E and B
- $R_c = \{ (e_i , c_i) \mid e_i \in E , c_i \in B \}$, relating E and C
- E has an identifying attribute $\underline{e}$ (candidate key) to distinguish each e_i in E
- all attributes of R, i.e. attr-R, are assigned to E

- refer to Fig.7.0.8 (a) , (b)

Cardinality Constraints on Ternary Relationship

We allow at most one arrow out of a ternary (or greater degree) relationship to indicate a cardinality constraint

For exampe, an arrow from *proj_guide* to *instructor* indicates each student has at most one guide for a project

If there is more than one arrow, there are two ways of defining the meaning.

For example, a ternary relationship *R* between *A*, *B* and *C* with arrows to *B* and *C* could mean

1. Each *A* entity is associated with a unique entity from *B* and *C* or
2. Each pair of entities from (*A*, *B*) is associated with a unique *C* entity, and each pair (*A*, *C*) is associated with a unique *B*

Each alternative has been used in different formalisms

To avoid confusion we outlaw more than one arrow

6.7.4 Placement of Relationship Attributes

- The attributes of a relationship sets ***R*** may be associated with one of its participating entity sets
- For the attribute ***attr-A*** in relationship set ***R*** between E_1 and E_2 ($R \subseteq E_1 \times E_2$, if the cardinality of ***R*** is
  - *one-to-one* : ***attr-A*** can be assigned as the attribute of either E_1 or E_2, rather than relationship set ***R***
  - *one-to-many* from E_1 to E_2 : ***attr-A*** can only be assigned as the attribute of the entity set E_2 (the many-side), rather than the relationship set ***R***

Placement of Relationship Attributes

- e.g. the relationship *advisor* in Fig.7.20 from *instructor* to *student* is *one to many*, the attribute *date* of *advisor*, which *specifies when the instructor became the advisor of the student*, can be made as an attribute of *student*, instead of the attribute of the relationship *advisor*

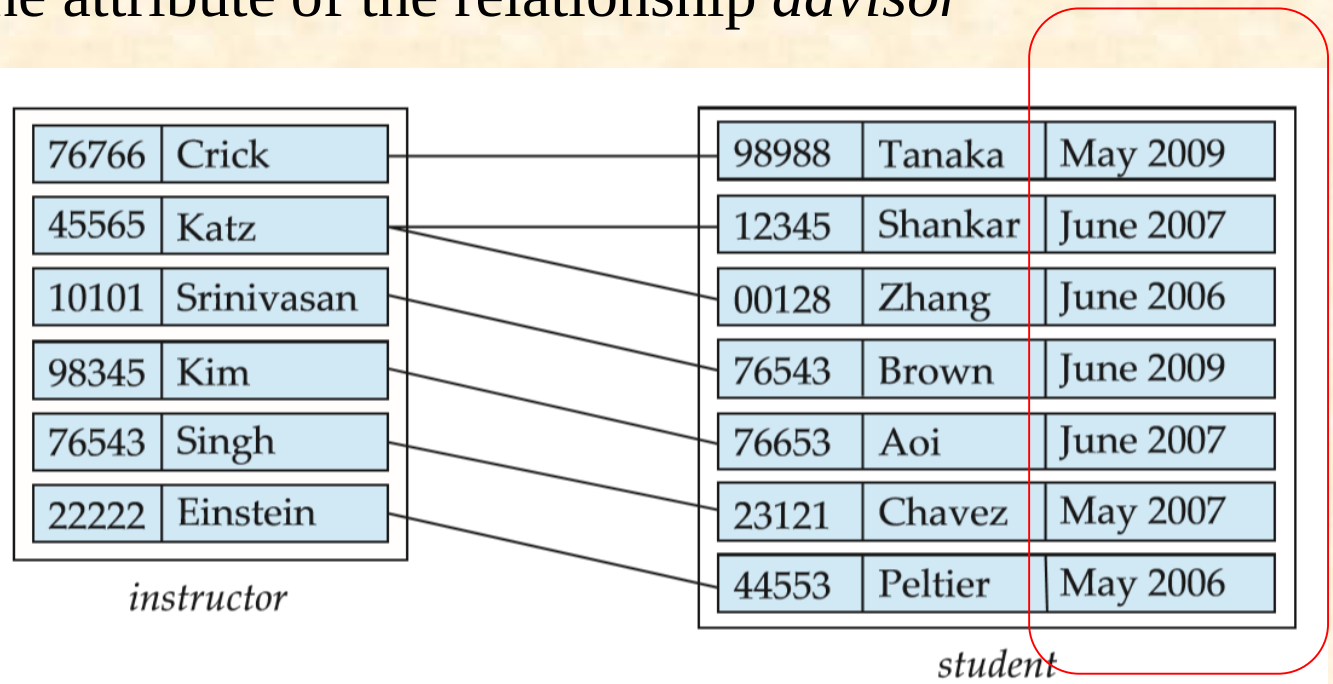


Figure 7.20 *date* as an attribute of the *student* entity set.

Placement of Relationship Attributes

- *many-to-many*: ***attr-A*** may only be assigned as the attribute of ***R***, not of its participating entity sets E_1 or E_2

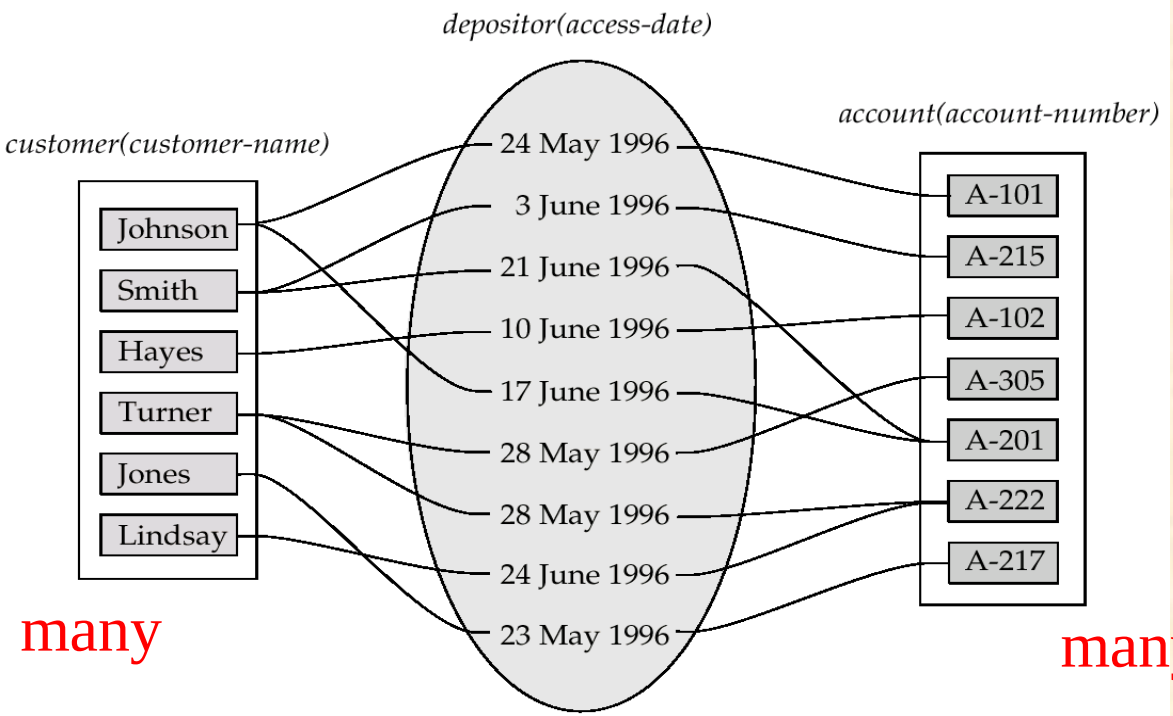


Fig.8.0.9 *Access-date* as attribute of the *depositor* relationship set

§ 6.8 Extended E-R model

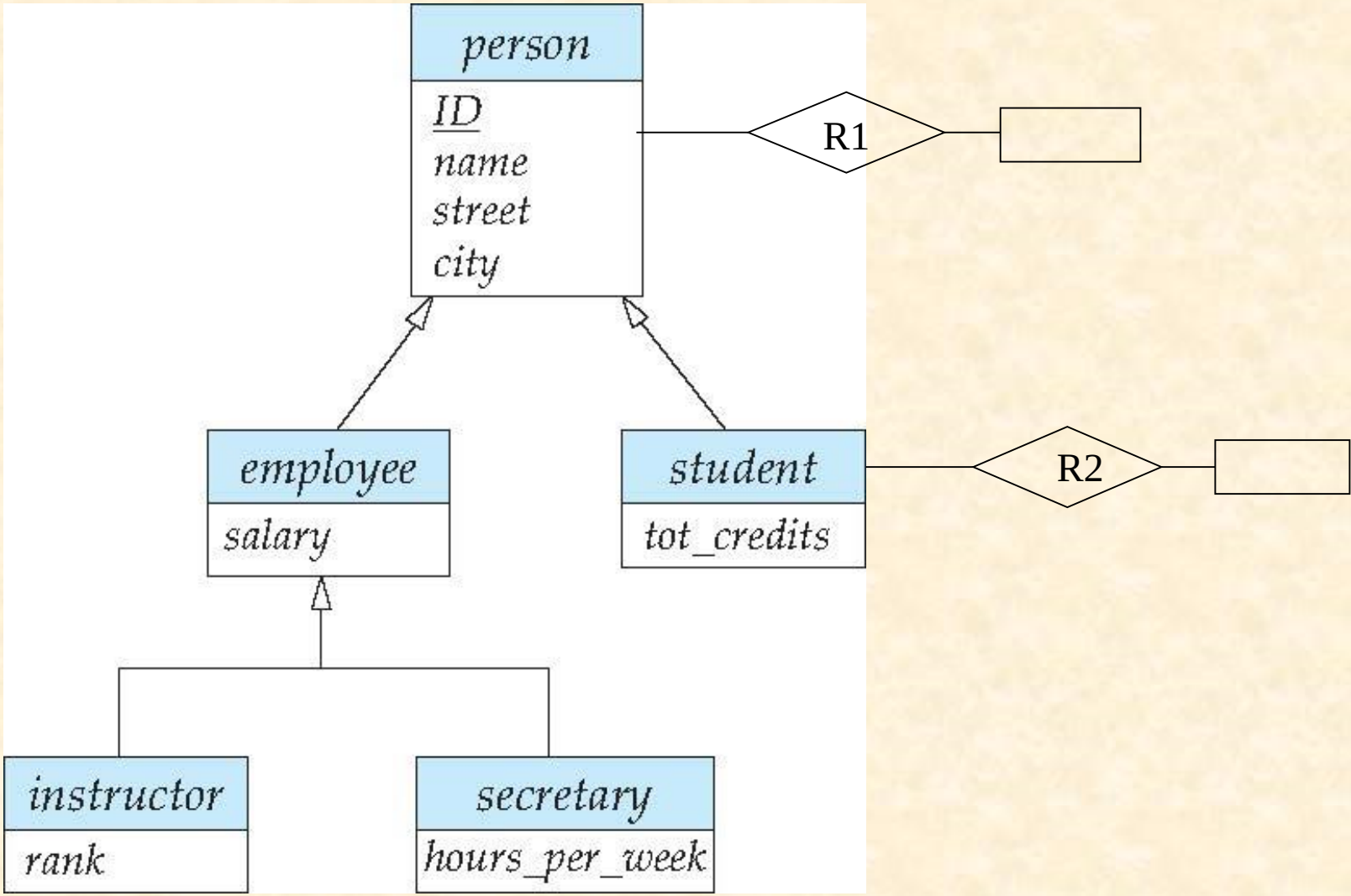
- Object-oriented (OO) E-R
  - specialization (特化，特殊化，例化）
  - generalization (概括化，泛化，普遍化)
  - attributes inheritance (属性继承）

6.8.1 Specialization

- **Specialization**
  - top-down design process of designating subgroups within an entity set, according to differences between entities in the entity set, each subgroup is distinctive from other entities in the set

Specialization (cont.)

- Refer to Fig.7.0.21
- These subgroupings become lower-level entity sets that may have attributes or may participate in relationships that do not apply to the higher-level entity set
- In the E-R model, specialization is depicted by a triangle（三角形） component labeled ***ISA***
  - e.g. *student* "is a" *person* in Figure 7.0.21



Overlapping – *employee* and *student*
Disjoint – *instructor* and *secretary*

Fig. 7.0.21 Specialization and generalization

6.8.2 Generalization

- **Generalization**
  - a bottom-up design process that combines a number of entity sets that share the same features into a higher-level entity set
  - e.g. Fig.7.0.21
- Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way
  - the terms *specialization* and *generalization* are used interchangeably

Superclass vs Subclass

- The ***ISA*** relationship, or *containment*, is also referred to as superclass-subclass relationship
- **Superclass and subclass**
  - for the entity sets *A* and *B*, if *A* is the generalization of *B*, i.e. *B* is the specialization of *A* , then *A* is the superclass of *B*, and *B* is the subclass of *A*

6.8.3 Attribute Inheritance

- A lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked
  - a higher-level entity's attributes and relationship can apply to all of its lower-level entities

6.8.4 Constraints on Generation/Specialization

- How to determine the membership of subclass/lower-level entities, i.e., how to divide superclass/higher-level entity set into lower-level entity sets
  - *condition-defined* , attribute-defined generalization
    - on the basis of explicit condition/predicate/attributes-value

    e.g. *student_type*= graduate, undergraduate
  - *user-defined*
    - e.g. all instructors are divided into four teams based on their assigned works, and each team becomes a subclass of the *instructor* entity set

Constraints on Specialization/Generalization

- **Completeness constraint** -- specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization.
  - **total**: an entity must belong to one of the lower-level entity sets, e.g. *student* generalization
  - **partial**: an entity need not belong to one of the lower-level entity sets, e.g. .*person* specialization
- Partial generalization is the default. We can specify total generalization in an ER diagram by adding the keyword **total** in the diagram and drawing a dashed line from the keyword to the corresponding hollow arrow-head to which it applies (for a total generalization), or to the set of hollow arrow-heads to which it applies (for an overlapping generalization).
- The *student* generalization is total: All student entities must be either graduate or undergraduate. Because the higher-level entity set arrived at through generalization is generally composed of only those entities in the lower-level entity sets, the completeness constraint for a generalized higher-level entity set is usually total

Constraints on Generation/Specialization (cont.)

- depicted in E-R diagram as

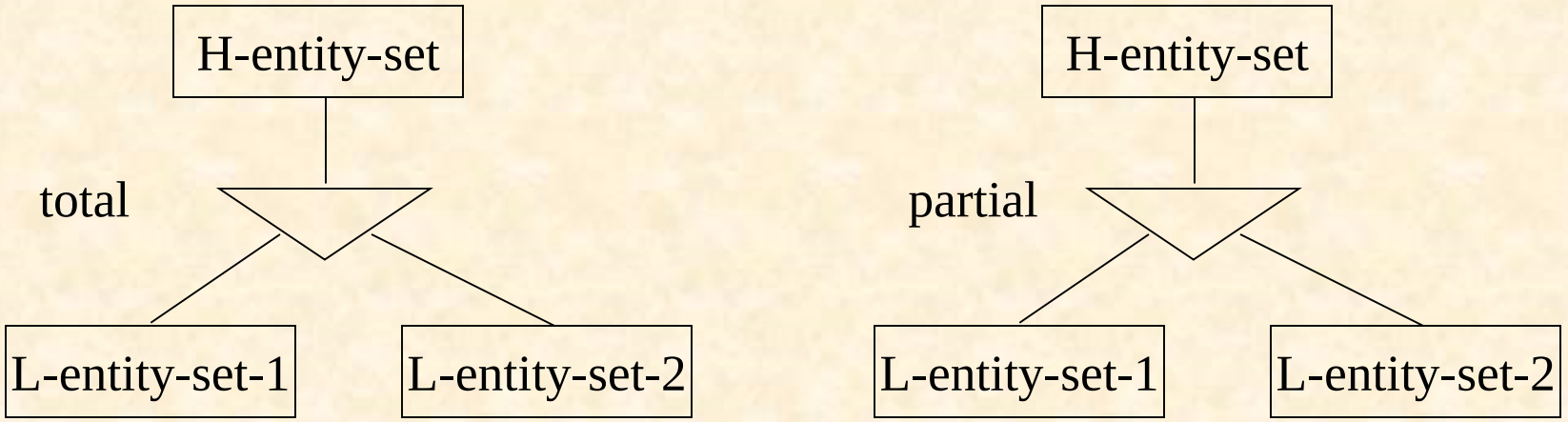


Fig.7.0.12 Completeness constraints

6.8.4 Constraints on Generation/Specialization (cont.)

- Constraints on whether or not a high-level entity may belong to more than one lower-level entity set *within a single specialization*, e.g. in Fig. 7.0.21
  - disjoint
    - a high-level entity can belong to only one lower-level entity set, i.e., L-entity-set-1 ∩ L-entity-set-2 = Φ
    - E.g. *instructor* and *secretary*

Constraints on Generation/Specialization (cont.)

- overlapping
  - a high-level entity can belong to more than one lower-level entity sets
  - L-entity-set-1∩L-entity-set-2 ≠ Φ
  - *E.g. employee* and *student*
- depicted in E-R diagram as

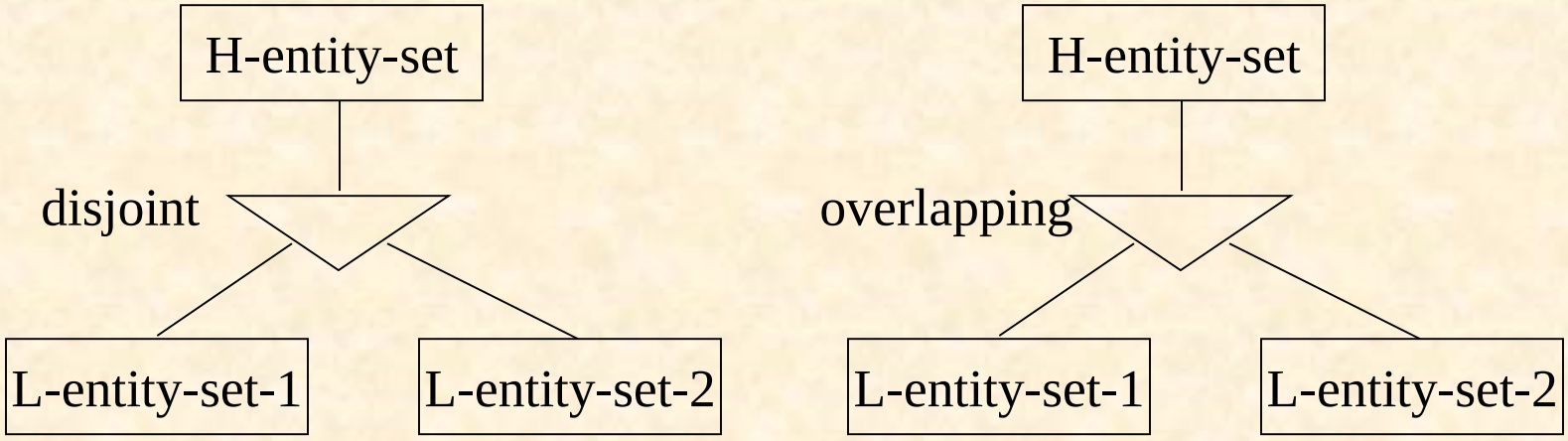
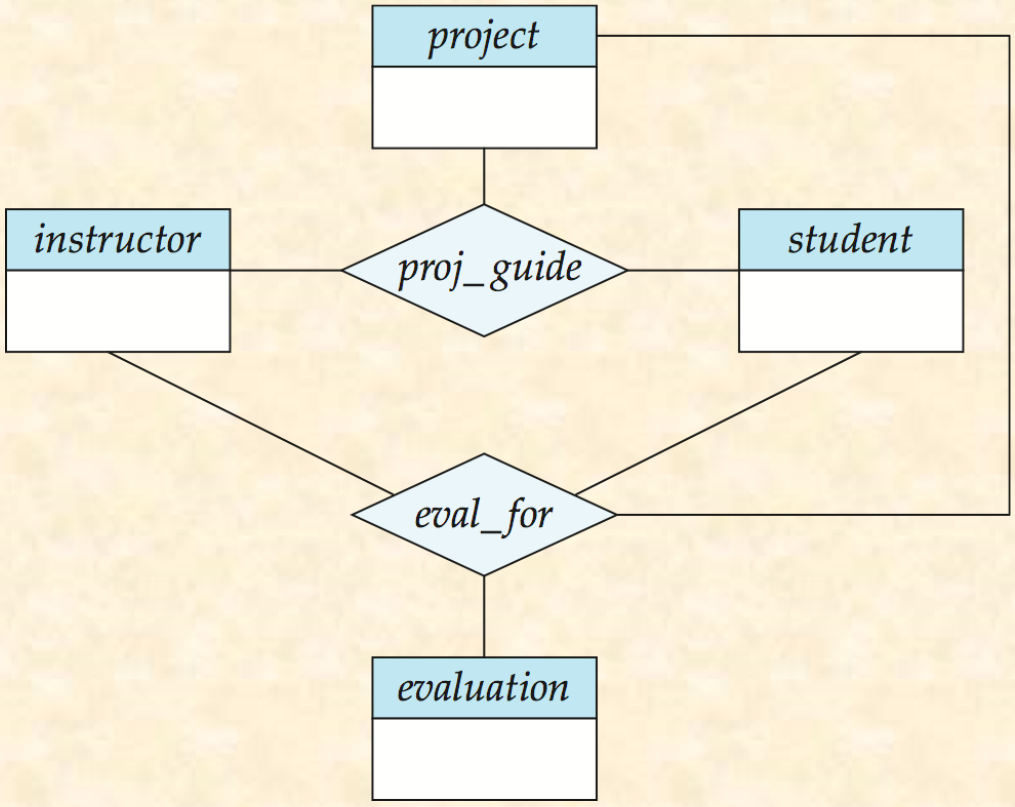


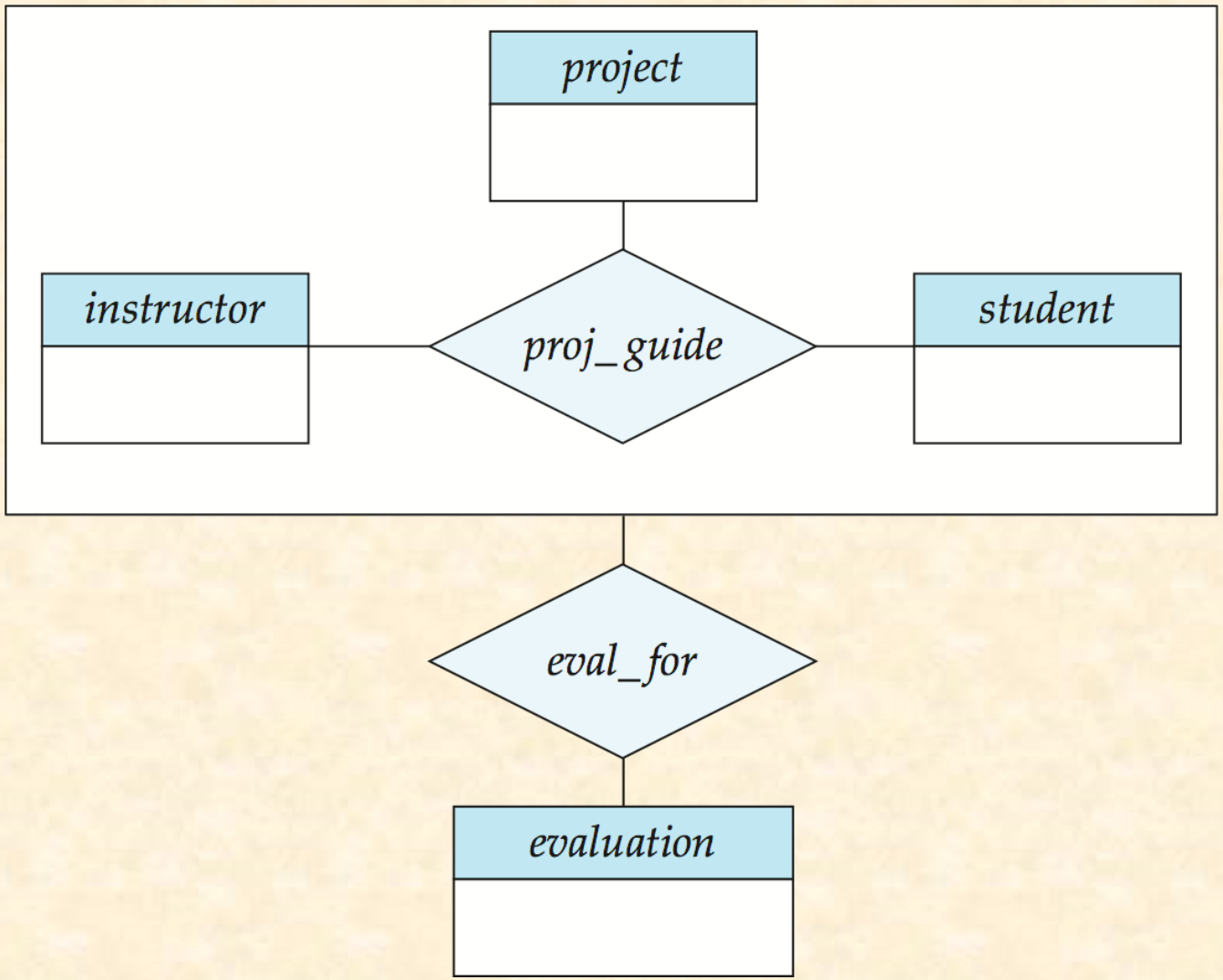
Fig.7.0.11 disjoint and overlapping in generalization

6.8.5 Aggregation

- **Aggregation**
  - to express *relationship among relationships*
  - an abstraction (***methodology***) through which relationships are treated as high-level entities



Aggregation (Cont.)



Examples

- A database used in an ordering (订货，订购) system, which contains information about *customers*, *products* and *orders*
- Two steps in DB design
  - E-R modeling
  - table reduction
- For more details about this example, refer to *Appendix D E-R Modeling and Table Reductions in Ordering Database*

6.7 Reduction to Relational Schemas

- ***Table = relation***

- Two steps in logical database design
  - generating of the initial relational schema from the E-R diagram, as illustrated in § 6.6
  - *relational schema normalization*, in Chapter 7
- Converting an E-R diagram to a table format is the basis for deriving a relational database design from an E-R diagram

Principles

- An *E-R diagram* describing a database can be represented by a collection of *relational tables*
- For each *entity set* and *relationship set*, there is a unique table which is assigned the name of the corresponding entity set or relationship set
- Each table has a number of columns; the columns generally corresponding to entities' or relationships' attributes, and have unique names

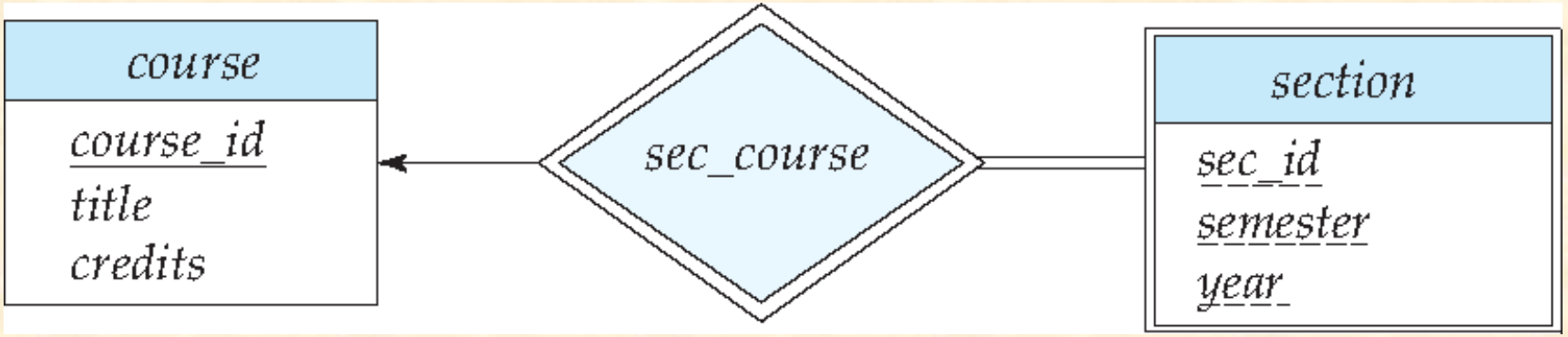
Representation of Strong/Weak Entity Sets

- A strong entity set reduces to a schema with the same attributes

  student(ID, name, tot_cred)

- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set

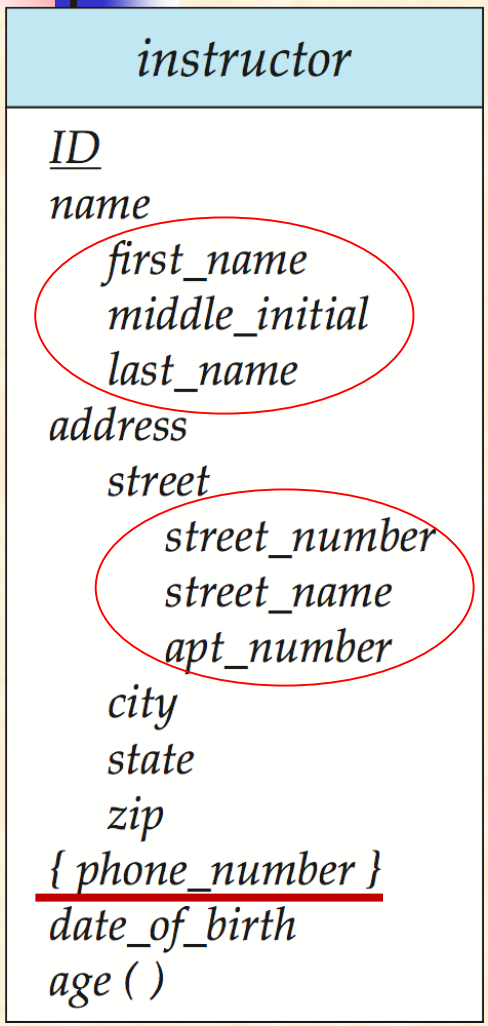
  section (course_id, sec_id, sem, year)



Representation of Weak Entity Sets

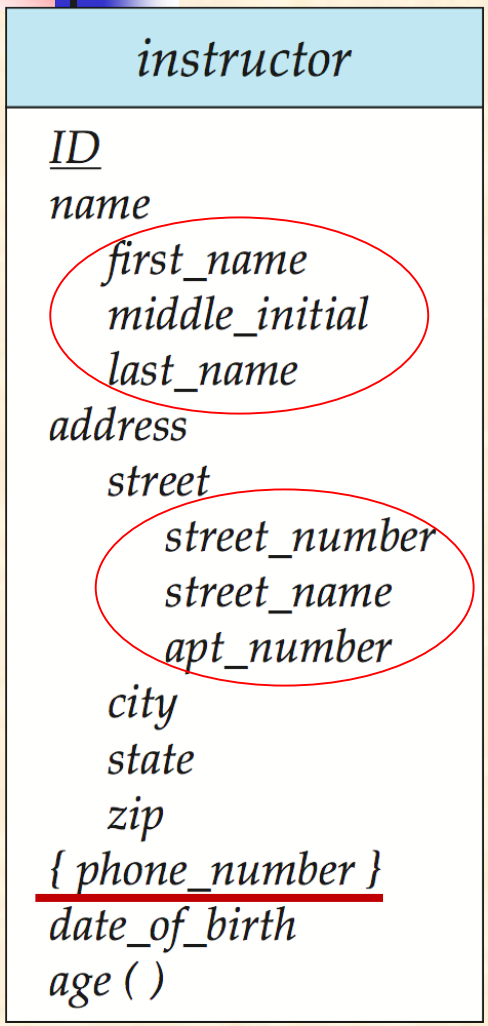
- For a weak entity set ***A***(a_1, a_2,…, a_m), its owner entity set is ***B***, primary-key (***B***) is b_1, b_2, …, b_n, then
  - table ***A*** with one column for each attribute in {a_1, a_2,…, a_m } ∪ {b_1, b_2, …, b_n }
  - the primary key of table ***A*** is the union of *discriminator/partial key* of weak entity set A and primary-key(***B***)

Representation of Entity Sets with Composite Attributes



- Composite attributes（复合属性）are flattened out by creating a separate attribute for each component attribute
- E.g. Given entity set *instructor* with composite attribute *name* with component attributes *first_name*, *middle_initial* and *last_name*, the schema corresponding to the entity set has three attributes:
  - *name_first_name*, *middle_initial* and *name_last_name*
    - Prefix omitted if there is no ambiguity (*name_first_name* could be *first_name)*

Representation of Entity Sets with Composite Attributes



- Ignoring multivalued attributes, extended instructor schema is
  - *instructor(ID, first_name, middle_initial, last_name, street_number, street_name, apt_number, city, state, zip_code, date_of_birth)*

Representation of Entity Sets with Multivalued Attributes

- A multivalued attribute *M* of an entity *E* is represented by a separate schema *EM*
- Schema *EM* has attributes corresponding to the primary key of *E* and an attribute corresponding to multivalued attribute *M*
- Example: Multivalued attribute *phone_number* of *instructor* is represented by a schema:
  inst_phone= (*ID*, *phone_number*)
- Each value of the multivalued attribute maps to a separate tuple of the relation on schema *EM*
  - For example, an *instructor* entity with primary key 22222 and phone numbers 456-7890 and 123-4567 maps to two tuples:
    (22222, 456-7890) and (22222, 123-4567)

Example

- For entity set ***customer***(*c-id*, *c-name*, *d-names*, *street*, *city*), each entity *customer* may have several dependents, so *d-names* (i.e., *dependent-name*) is a multivalued attribute

c-id	*c-name*	*d-names*	*street*	*city*
321-12-2	John	{Hayes, Adams}	North	Rye
322-10-4	Smith	{Mary, Elice}	Spring	Princeton

Fig.6.0.18 Entity set ***customer*** with *multivalued* attribute

- the entity set ***customer*** is reduced into two tables
  - ***c-table,*** which has no multivalued attribute *d-names*

c-id	*c-name*	*street*	*city*
321-12-2	John	North	Rye
322-10-4	Smith	Spring	Princeton

  - ***d-table,*** which has primary key {*c-id*, *d-names*}, and is as follows

Reduction of Multivalued Attributes (cont.)

c-id	*d-names*
321-12-2	Hayes
321-12-2	Adams
322-10-4	Mary
322-10-4	Elice

Representation of Relationship Sets

- Reduction of a relationship set into tables are ***strongly*** dependent on the *mapping cardinality* constraint and *total/partial constraints* related to this relationship set

- A *many-to-many* relationship set is represented as a table with columns for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set

Example

instructor(*ID*, *name*, *salary*)
student(*ID*, *name*, *tot_cred*)
advisor(*instrctor.ID*, *student.ID*)

Fig.7.0.14 Reduced *advisor* table

Many-to-many

- For relationship set ***R*** between entity set ***A*** and ***B***, primary-key(***A***) ∪ primary-key(***B***) = {a_1, a_2,…, a_m}, descriptive attributes of ***R*** are b_1, b_2, …, b_n, then ***R*** can be reduced to
  - table R with one column for each attribute in {a_1, a_2,…, a_m } ∪ {b_1, b_2, …, b_n }

Many-to-one/One-to-many

- Many-to-one and one-to-many relationship sets ***that are total on the many-side*** can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side

- For a relationship sets ***R***, which satisfies that (refer to Fig.7.0.15)
  - *many-to-one* mapping from entity set ***A*** to ***B***, ***and***
  - *the many-side* entity set ***A*** *totally participates in* ***R***

  , then ***R*** should not be represented as an independent table, it can be reduced to the table ***A*** from entity set ***A***, by
    - adding *the primary key of B* into table ***A***
    - adding descriptive attributes of ***R*** into table ***A***
- For a one-to-many relationship sets ***R*** from ***A*** to ***B***, the similar reduction can be conducted on the basis of table ***B***.

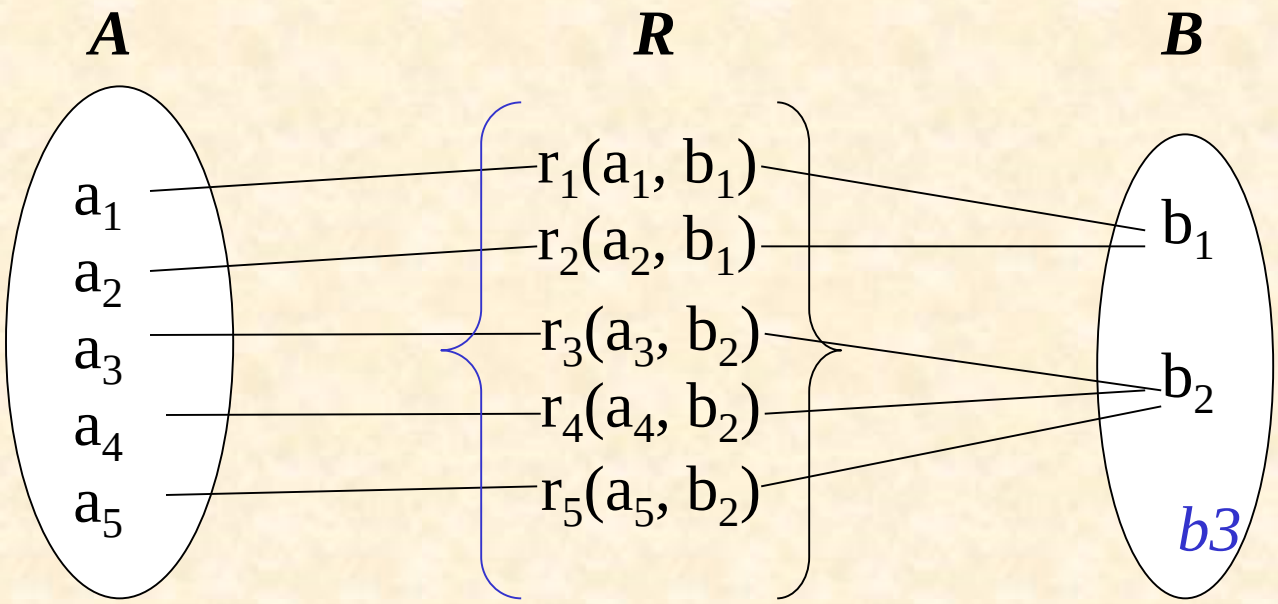


Fig. 7.0.15 A *many-to-one* relationship set ***R***

Example

- Instead of creating a schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from *instructor*
  - *instructor(ID, name, salary, dept_name)*
  - *department(dept_name, building, budget)*

foreign key referencing constraint between *instructor* and *department*

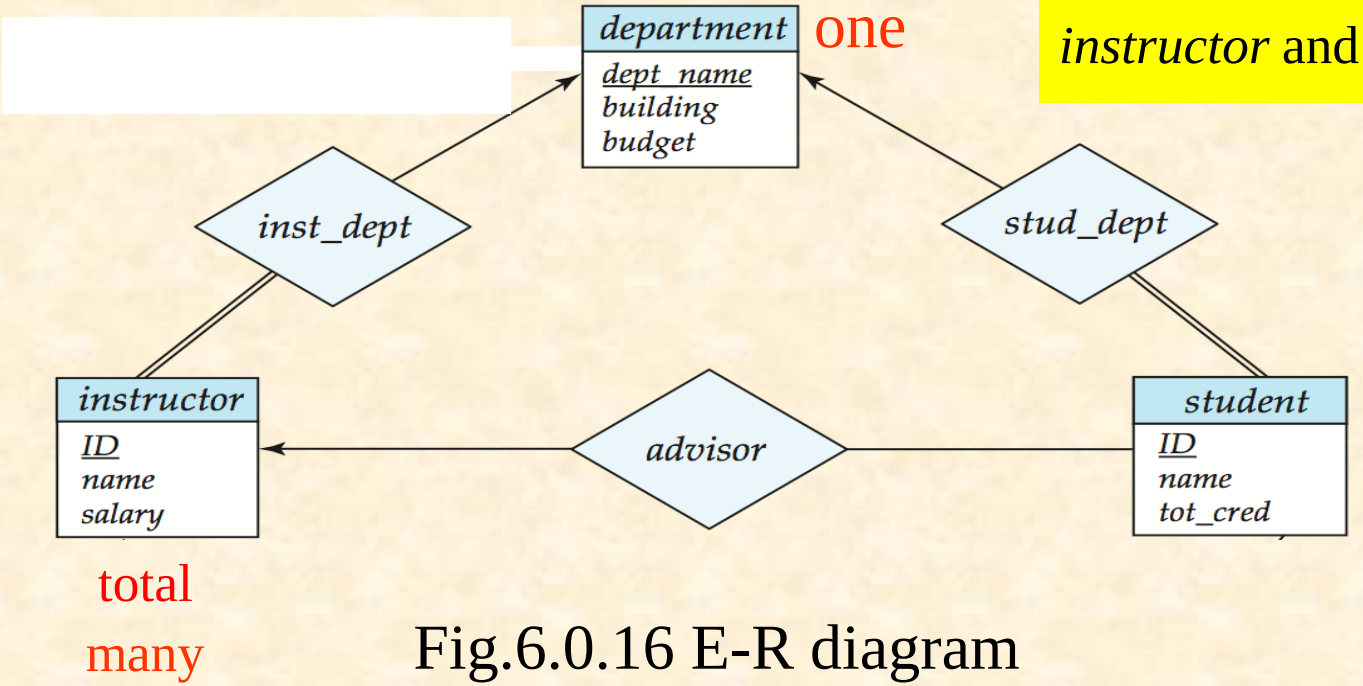


Fig.6.0.16 E-R diagram

Many-to-one Relationship Sets *with partial participation*

The relationship is reduced to a independent table

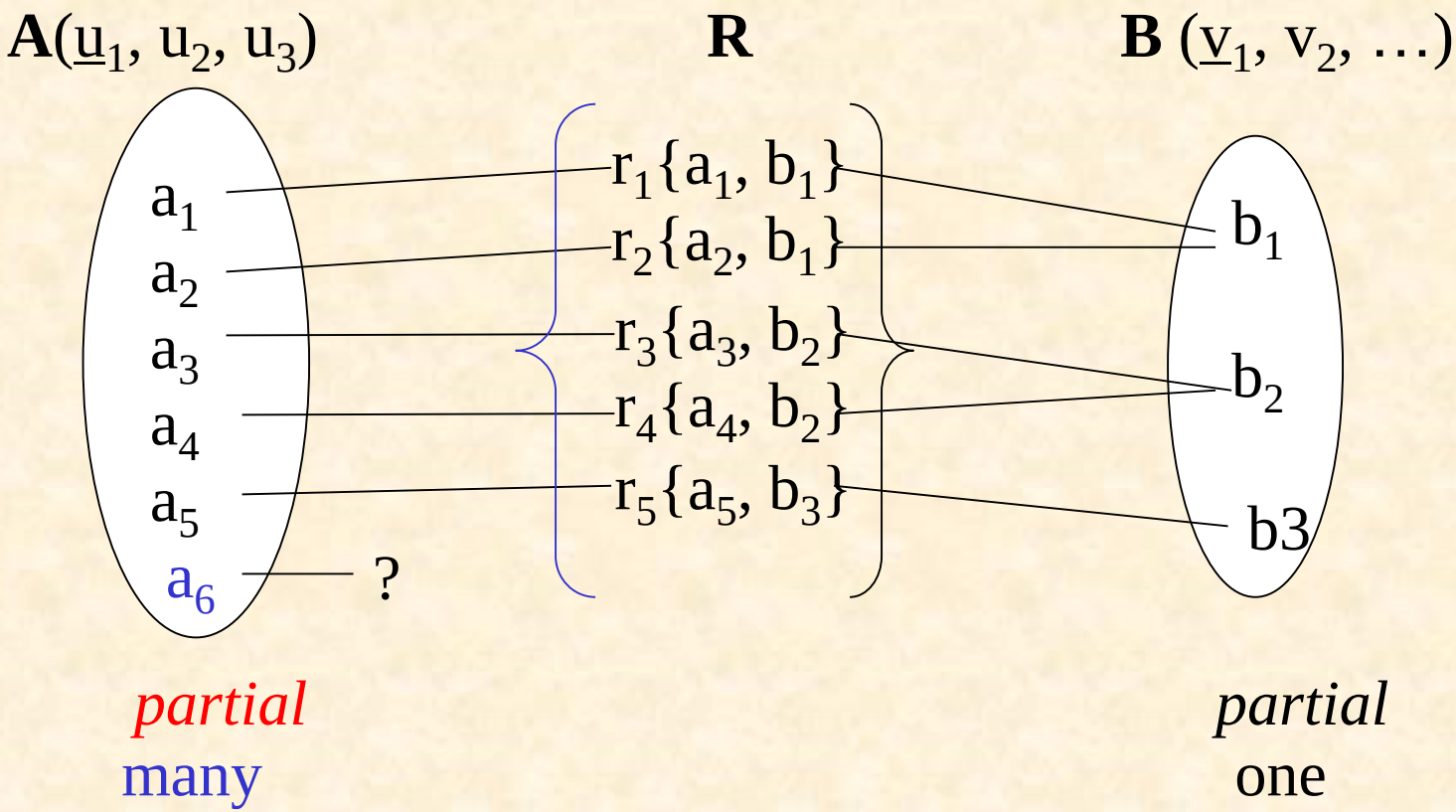


Fig.6.0.17-1 Why not reducing ***R*** into table ***A***

在数据库系统中，尽可能避免关系表中的空值null—用default value 代替null

- If ***R***={r_1, r_2, r_3, r_4, r_5} is reduced to table
  A'($\underline{u_1}$, u_2, u_3 , v_1) ={a_1', a_2', a_3', a_4', a_5', a_6'},
  - a_6 does not participate in ***R***
  - thus, a_6' will has *null* on the attribute v_1

Table ***A***' ={a_1', a_2', a_3', a_4', a_5', a_6'}

	$\underline{u_1}$	$\underline{u_2}$	$\underline{u_3}$	$\underline{v_1}$	
a_1'	*	*	*	*	$r_1\{a_1, b_1\}$
a_2'	*	*	*	*	$r_2\{a_2, b_1\}$
a_3'	*	*	*	*	$r_3\{a_3, b_2\}$
a_4'	*	*	*	*	$r_4\{a_4, b_2\}$
a_5'	*	*	*	*	$r_5\{a_5, b_3\}$
a_6'	*	*	*	*null*	

Fig.7.0.17-2 Why not reducing ***R*** into table ***A***

Reduction of *one-to-one* Relationship Sets

- One-to-one relationship can be viewed as a specialized instance of many-to-one relationship.

- For *one-to-one* relationship set ***R*** relating entity set ***A*** and ***B***, assuming one of two entity sets, i.e. ***A*** is *totally participating* ***R***, then
  - ***R*** is represented by adding *B's primary key* to the table ***A*** of entity set ***A***
  - *R's descriptive attributes* are also added to table ***A***

Reduction of Specialization/Generalization

- Method 1:
  - Form a schema for the higher-level entity
  - Form a schema for each lower-level entity set, include primary key of higher-level entity set and local attributes

schema	attributes
person	ID, name, street, city
student	ID, tot_cred
employee	ID, salary

  - Drawback: getting information about, an *employee* requires accessing two relations, the one corresponding to the low-level schema and the one corresponding to the high-level schema

Representing Specialization via Schemas

- Method 2:
  - Form a schema for each entity set with all local and inherited attributes

schema	attributes
person	ID, name, street, city
student	ID, name, street, city, tot_cred
employee	ID, name, street, city, salary

  - Drawback: *name*, *street* and *city* may be stored redundantly for people who are both students and employees

- Method 1:
  - for the three entity sets, creating three tables

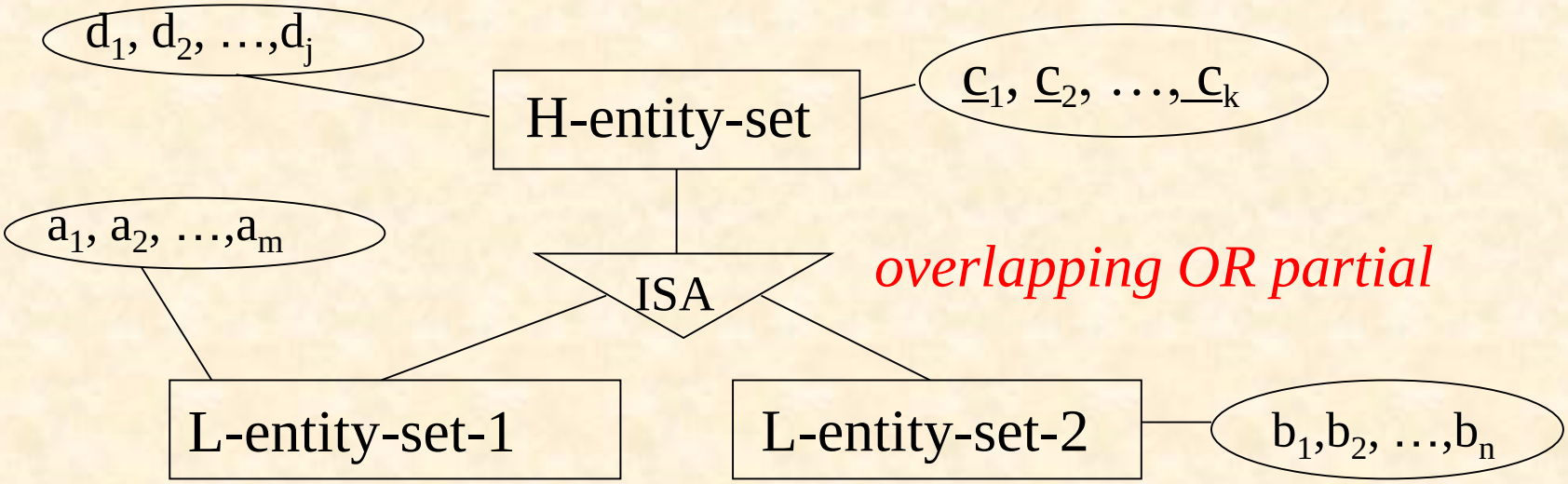


Fig.8.0.19 E-R diagram with *generalization /specialization*

Representation of Generalization (cont.)

- T_1 (c_1, c_2, …, c_k, d_1, d_2, …,d_j) for H-entity-set
- T_2 (a_1, a_2, …,a_m, c_1, c_2, …, c_k) for L-entity-set1
- T_3 (b_1,b_2, …,b_n , c_1, c_2, …, c_k) for L-entity-set2
- note:
  - c_1, c_2, …, c_k are primary attributes of H-entity-set
  - c_1, c_2, …, c_k are the *foreign key* from T_2 and T_3 *referencing* T_1, representing the association between H-entity-set and L-entity-set1/ L-entity-set2.

Representation of Generalization (cont.)

- Method 2. if generalization is ***disjoint*** and ***complete***, creating two tables

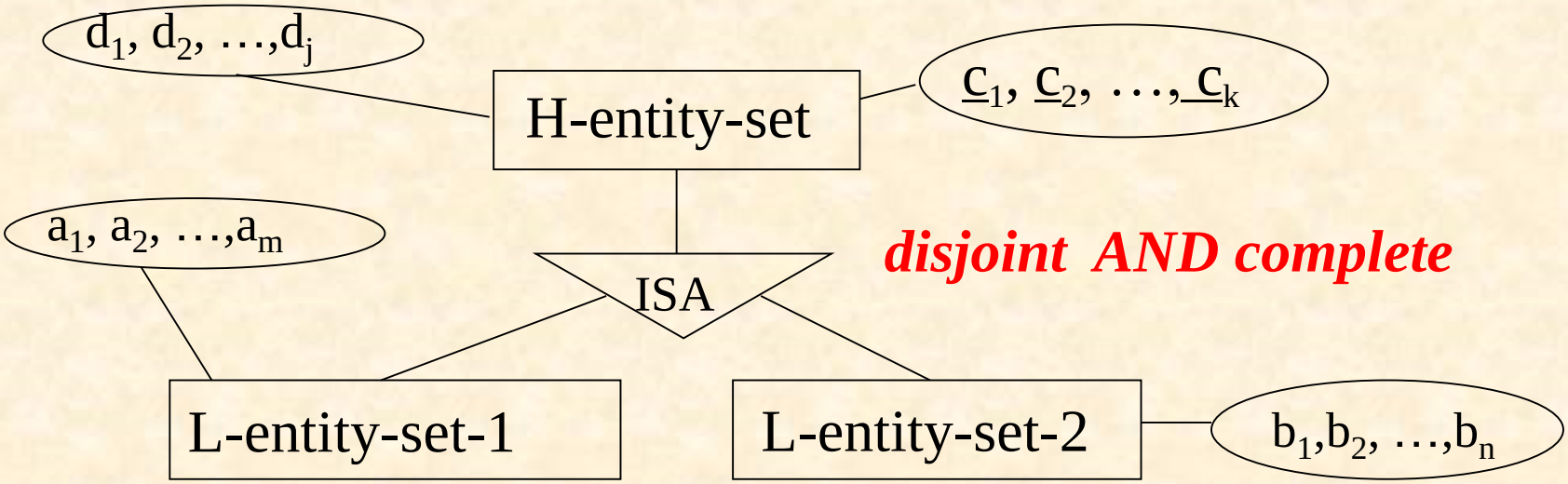
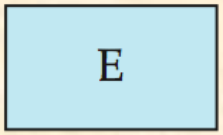


Fig.8.0.20 E-R diagram with *generalization /specialization* (II)

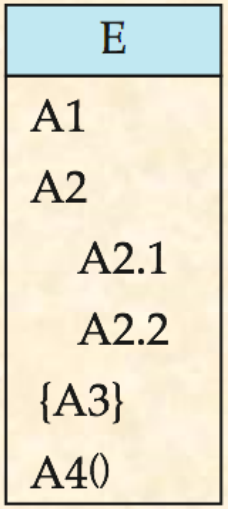
Representation of Generalization (cont.)

- T_2 (a_1, a_2, …,a_m, c_1, c_2, …, c_k, d_1, d_2, …,d_j) for L-entity-set1
- T_3 (b_1,b_2, …,b_n, c_1, c_2, …, c_k, d_1, d_2, …,d_j) for L-entity-set2
- a_1, a_2, …,a_m, are attributes of L-entity-set1, b_1,b_2, …,b_n are attributes of L-entity-set2, and c_1, c_2, …, c_k, d_1, d_2, …,d_j are all attributes of H-entity-set
- *replacing* high-level entity set H-entity-set *with* T2 and T_3 of lower-level entity sets.

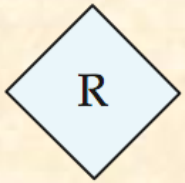
6.9 Symbols Used in E-R Notation



entity set

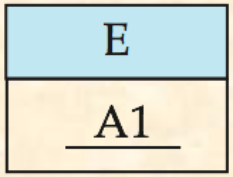


attributes:
simple (A1),
composite (A2) and
multivalued (A3)
derived (A4)

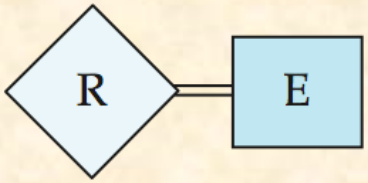


relationship set

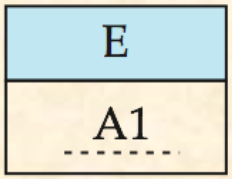
identifying
relationship set
for weak entity set



primary key

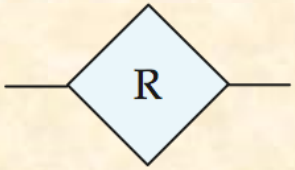


total participation
of entity set in
relationship

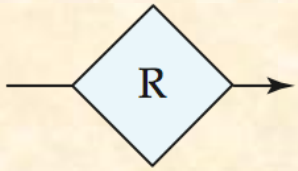


discriminating
attribute of
weak entity set

Symbols Used in E-R Notation (Cont.)

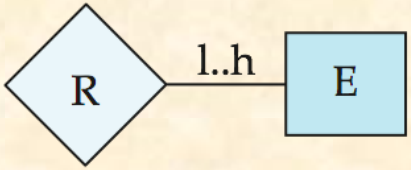


many-to-many relationship

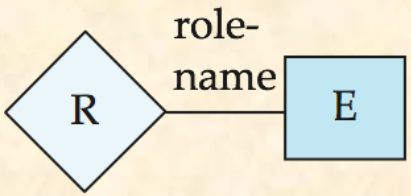


many-to-one relationship

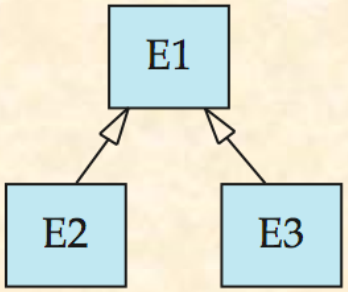
one-to-one relationship



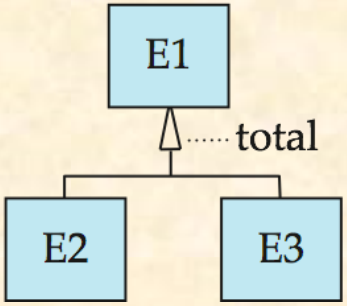
cardinality limits



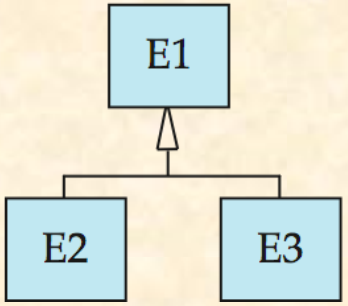
role indicator



ISA: generalization or specialization



total (disjoint) generalization

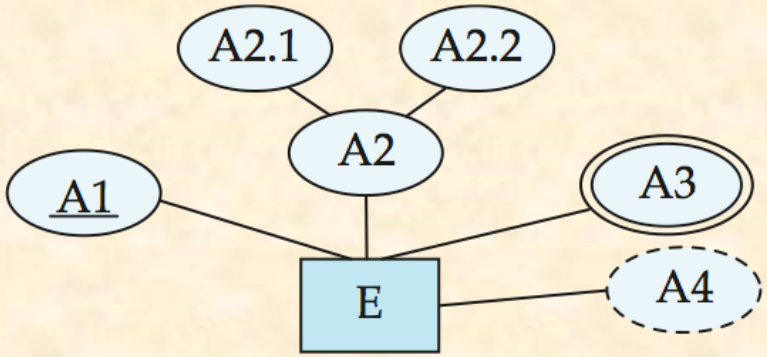


disjoint generalization

Alternative ER Notations

- Chen, IDE1FX, …

entity set E with
simple attribute A1,
composite attribute A2,
multivalued attribute A3,
derived attribute A4,
and primary key A1



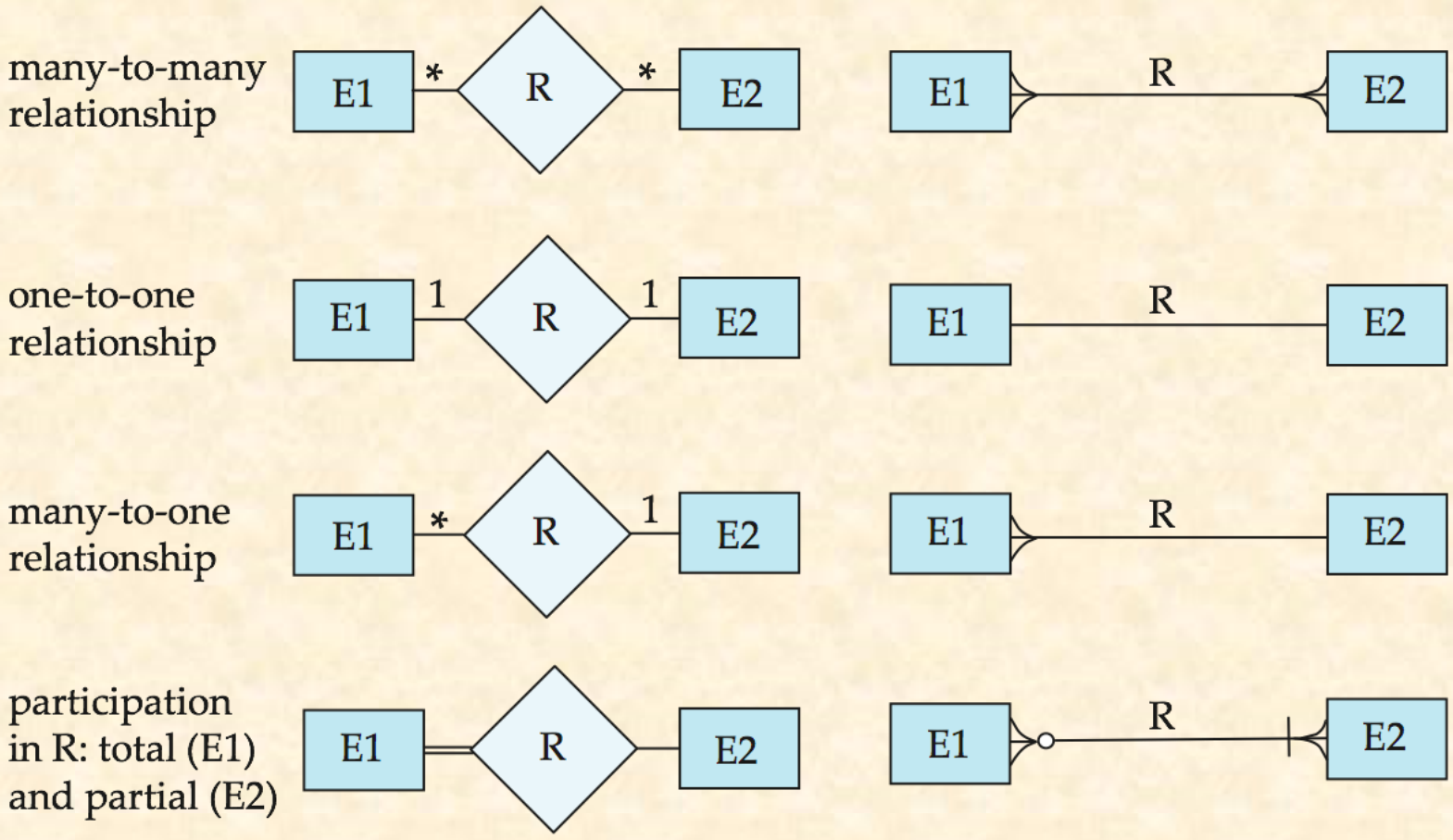
weak entity set 

generalization 

total generalization 

Alternative ER Notations

Chen **IDE1FX (Crows feet notation)**

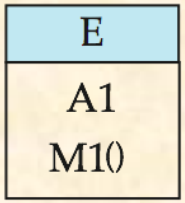


UML

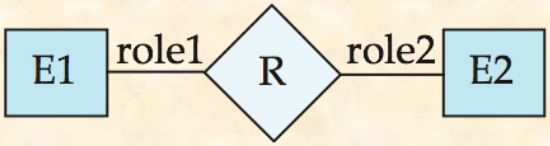
- **UML**: Unified Modeling Language
- UML has many components to graphically model different aspects of an entire software system
- UML Class Diagrams correspond to E-R Diagram, but several differences.

ER vs. UML Class Diagrams

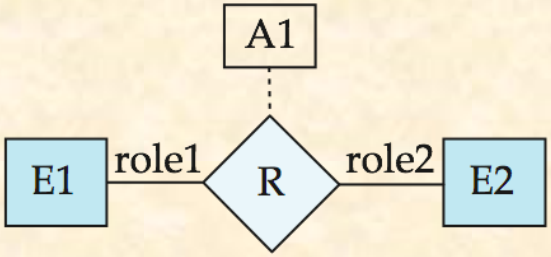
ER Diagram Notation



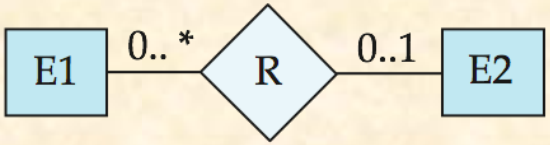
entity with attributes (simple, composite, multivalued, derived)



binary relationship

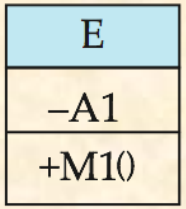


relationship attributes

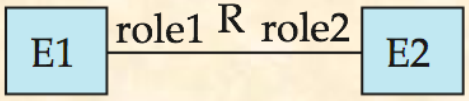


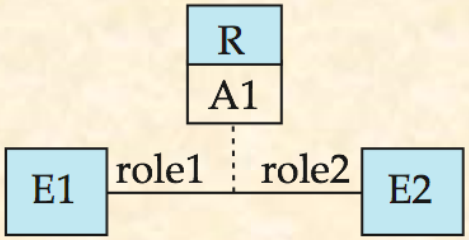
cardinality constraints

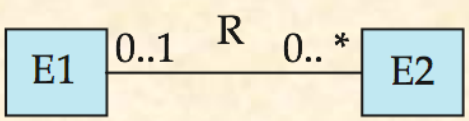
Equivalent in UML



class with simple attributes and methods (attribute prefixes: + = public, – = private, # = protected)





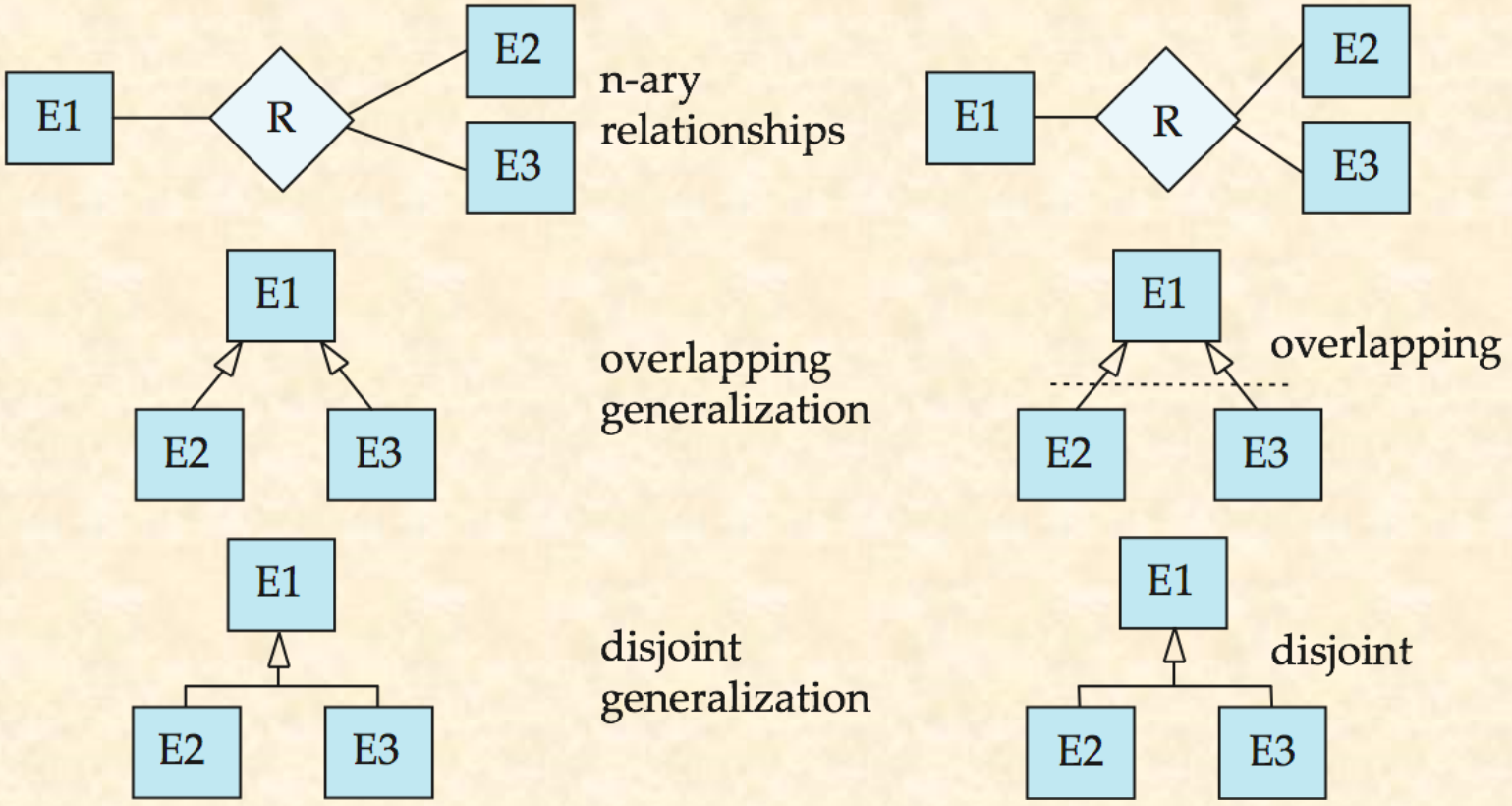


*Note reversal of position in cardinality constraint depiction

ER vs. UML Class Diagrams

ER Diagram Notation

Equivalent in UML



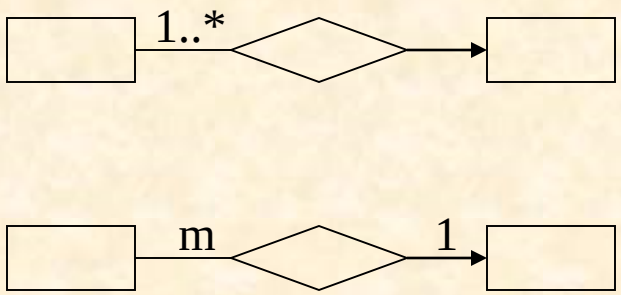
*Generalization can use merged or separate arrows independent of disjoint/overlapping

UML Class Diagrams (Cont.)

- Binary relationship sets are represented in UML by just drawing a line connecting the entity sets. The relationship set name is written adjacent to the line.
- The role played by an entity set in a relationship set may also be specified by writing the role name on the line, adjacent to the entity set.
- The relationship set name may alternatively be written in a box, along with attributes of the relationship set, and the box is connected, using a dotted line, to the line depicting the relationship set.

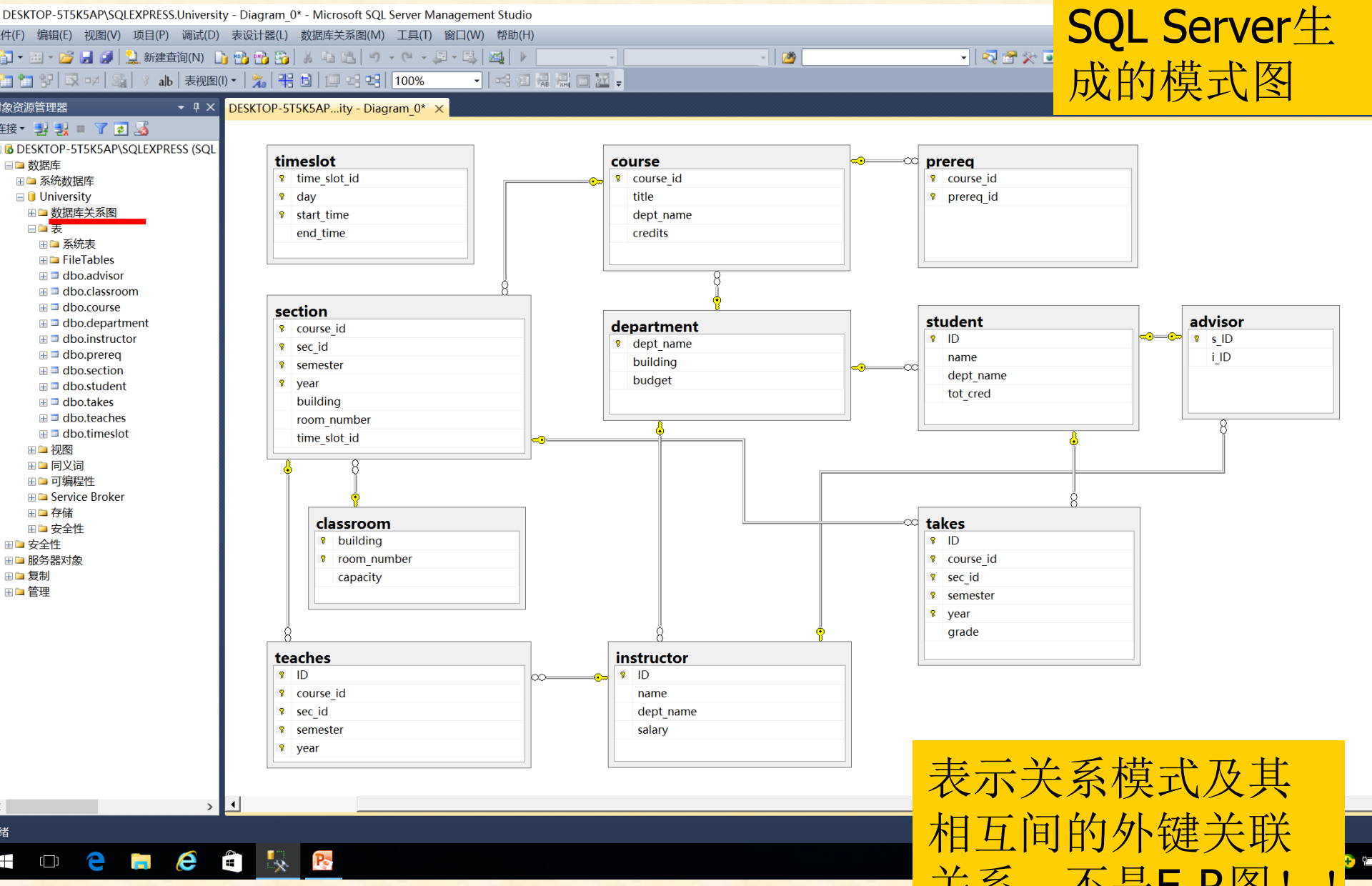
注意

- 画E-R图时，mapping cardinality、participation和cardinality limits约束的画法必须统一，不允许混用
  - 反例



Schema Diagram in Relational Model

SQL Server生成的模式图



表示关系模式及其相互间的外键关联关系，不是E-R图！！

Excises

- 1. 完成教材中习题6.20，画出对应E-R图，并转换为关系模式。
- 2. 将下页PPT中的数据需求用E-R图进行描述，并转化为关系模式。

Consider the following information about the pharmacies, patients and drugs:

- (1) Patients are identified by an SSN, and their names, addresses, and ages must be recorded.
- (2) Doctors are identified by an SSN. For each doctor, the name, specialty, and years of experience must be recorded.
- (3) Each pharmaceutical company (制药公司) is identified by name and has a phone number.
- (4) For each drug, the trade name and formula（成份）must be recorded. Each drug is produced by a given pharmaceutical company, and the trade name identifies a drug uniquely from among the products of that company.
- (5) Each pharmacy(药房) has a name, address, and phone number. Each pharmacy is identified by ID.
- (6) Every patient has a primary doctor. Every doctor has at least one patient.
- (7) Each pharmacy sells several drugs and has a price for each. A drug could be sold at several pharmacies, and the price could vary from one pharmacy to another.
- (8) Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors. Each prescription has a date and a quantity associated with it.
- (9) Pharmaceutical companies have long term contracts with pharmacies. A pharmaceutical company can contract with several pharmacies, and a pharmacy can contract with several pharmaceutical companies. For each contract, you have to store a start date, an end date.

Appendix D E-R Modeling and Table Reductions in Ordering Database

- A database used in an ordering (订货/订购) system contains information about *customers*, *products* and *orders*. The following information is to be included:
  - for each ***customer***
    - *customer-number*, *ship-to-addresses* (several per customer), *name*, *balance*, *discount*
  - for each ***order***（订单）
    - *order number*, *customer number*, *ship-to address* (one per order), *date of ordering* (including *year*, *month*, *and day*), *products* (several kinds per order), and *quantity* ordered of each product

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

- for each kind of ***product***
  - *product-number, name, manufacturing-plant, quantity on hand*
- Entity sets: ***Customer, Order, Product*** and their *attributes.*

  In ***Customer,*** *ship-to addresses* is a multi-valued attribute, but in ***Order***, *ship-to addresses* is a single-valued attribute. In ***Order***, *date* is a composite attribute

- Some constraints(i.e., integrity constraints) about this ordering system are as follows
  - a customer may *make* one or more orders, and some customers may make no orders
  - each order must be for one (and only one) customer

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

- ***relationship sets***
  - relationship set ***make*** between ***Customer*** and ***Order***
  - ***Customer*** partly participate ***make,*** and ***Order*** totally participate ***make***
  - in relationship set ***make***, mapping cardinality from ***Customer*** to ***Order*** is one-to-many
- each order must *contain* at least one kind of products
- a kind of product may appear in several orders, but some kinds of products may not be ordered by any customers
- ***relationship set***
  - relationship set ***contain*** between ***Order*** and ***Product***, and

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

descriptive attributes ***quantity of ordered*** for relationship set ***contain***

- ***Product*** partly participate ***contain,*** and ***Order*** totally participate ***contain***
- In relationship set ***contain***, mapping cardinality from ***Order*** to ***Product*** is many-to-many

- E-R diagram of the ordering DB is shown in Fig. C.1

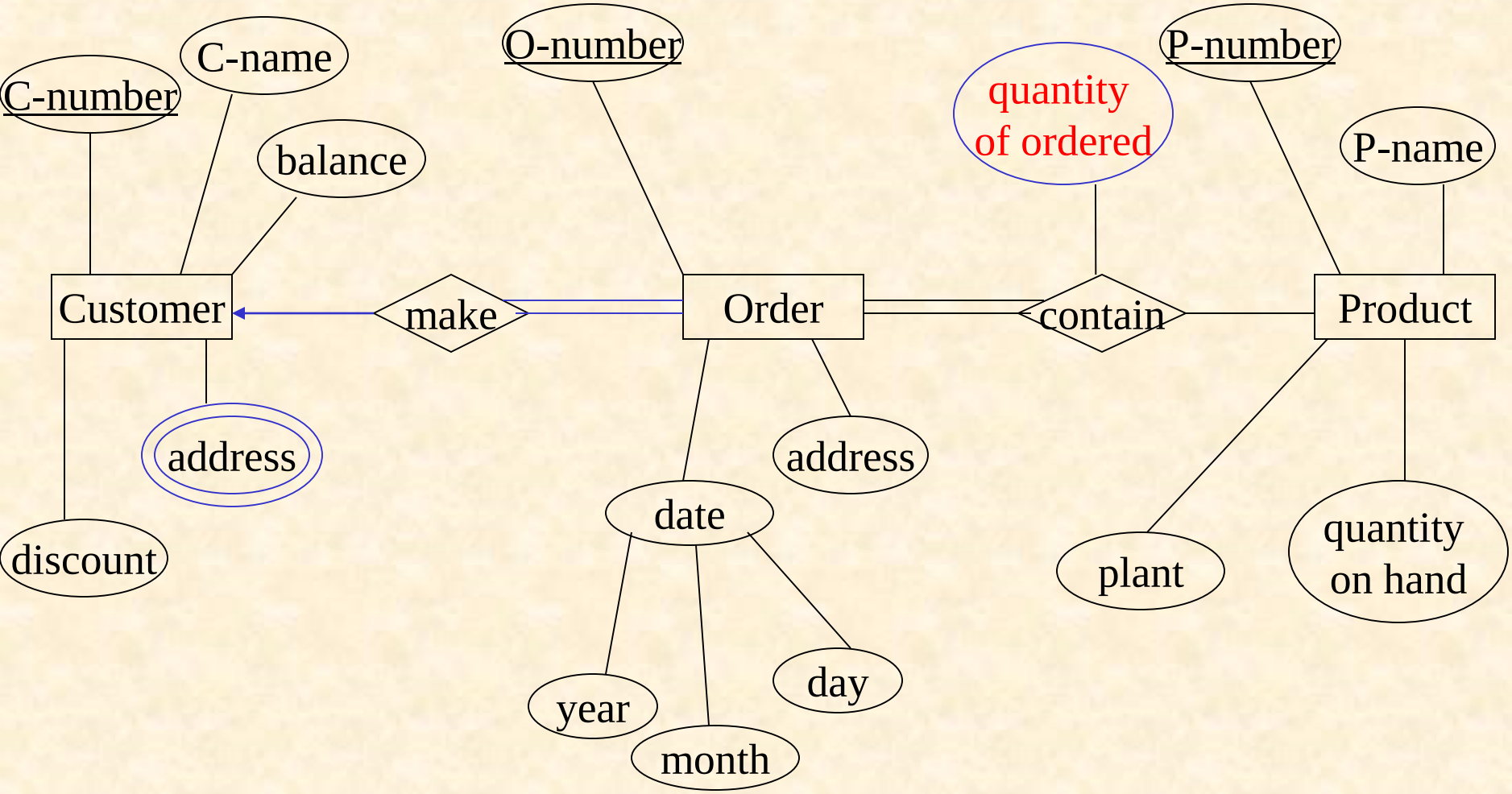


Fig. C.1. E-R diagram of the ordering DB is shown in

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

- The E-R diagram in Fig. C.1 is then reduced to some relational tables, according to the methodologies mentioned in § 6.9
- For entity set ***Customer*** and its multi-valued attribute *ship-to addresses*, there are two tables
  - ***customer***=(C-number, c-name, discount, balance)
  - ***ship-to-addresses*** = (C-number, ship-to-addresses)
- For entity set ***Order*** and its composite attribute ***date*** which should be decomposed , there is a table
  - ***order1***=(O-number, ship-to-address, year, month, day)

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

- For entity set ***Product***, there is a table
  - ***product***=(P-number, p-name, plant, quantity-on-hand)
- For relationship set ***make***, there are no descriptive attributes, mapping cardinality from ***Customer*** to ***Order*** is one-to-many, and ***Order*** totally participate ***make***. So, ***make*** should not be reduced into a separate table, and can be represented by adding the primary key C-number of ***Customer*** , that is *C-number* into the table ***order***
  - ***order2***=(O-number, C-number, ship-to-address, year, month, day)
  - note: the primary key C-number is already in the table ***order***

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

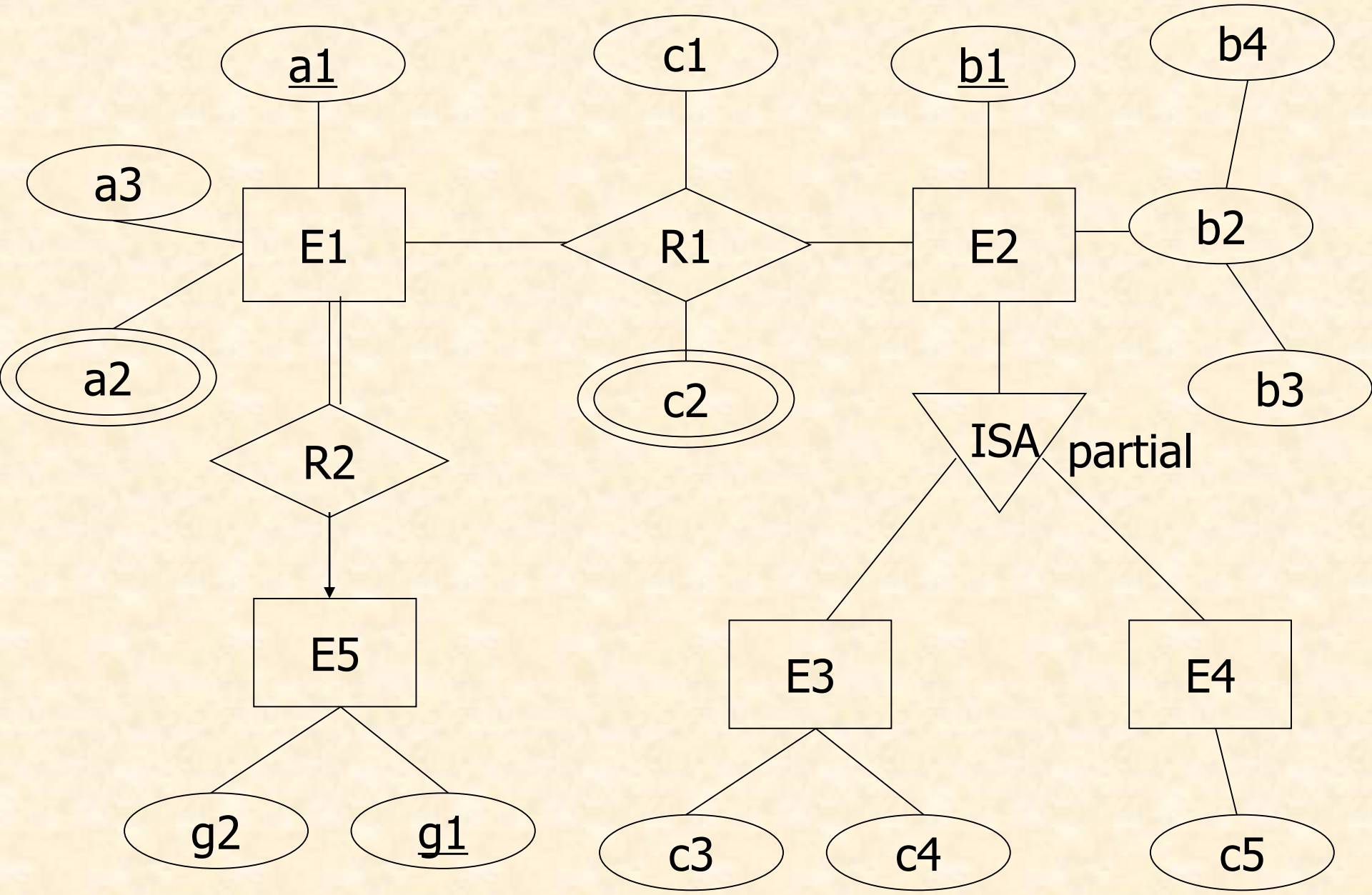
- Supposing mapping cardinality from ***Order*** to ***Product*** is many-to-many, for relationship set ***contain***, there is a descriptive attribute ***quantity of ordered***, ***contain*** should be represented as a separate table with columns for the primary keys of the two participating entity and descriptive attribute
  - ***contain***=(O-number, P-number, quantity of ordered))
- Supposing mapping cardinality from ***Order*** to ***Product*** is one-to-many, because ***Product*** at the many side only partly participate relationship set ***contain*** , ***contain*** should also be represented as a separate table
  - ***contain***=(O-number, P-number, quantity of ordered))

Appendix D E-R Modeling and Table Reductions in Ordering Database (cont.)

- So, the E-R diagram in Fig. C.1 can be reduced into the following five tables
  - ***customer***=(C-number, c-name, discount, balance)
  - ***ship-to-addresses*** = (C-number, ship-to-addresses)
  - ***order***=(O-number, C-number, ship-to-address, year, month, day)
  - ***product***=(P-number, p-name, plant, quantity-on-hand)
  - ***contain=***(O-number, P-number, quantity of ordered))

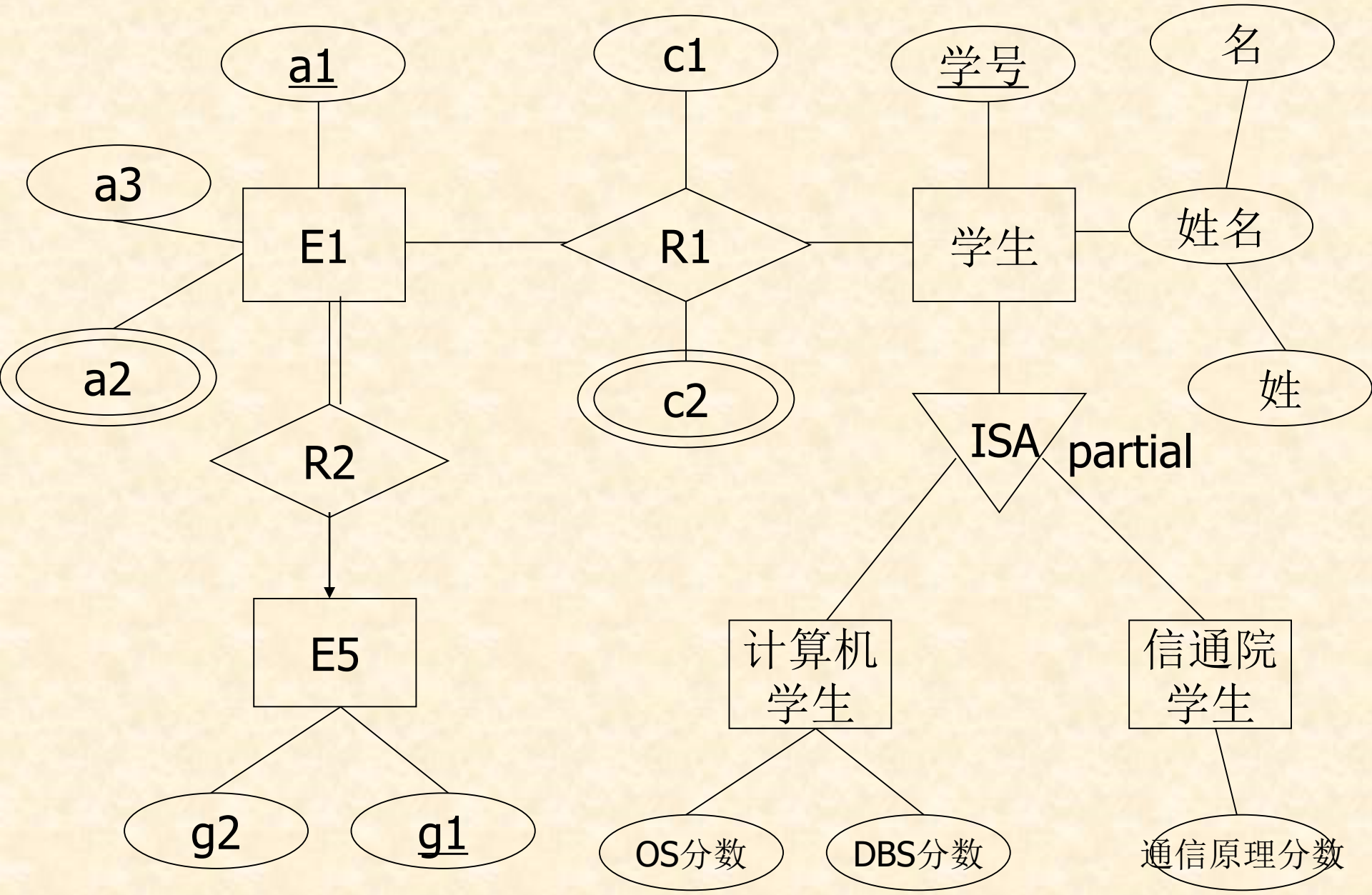
Appendix E Example

- Convert the following E-R diagram to the proper relation schemas and identify the primary key of each relation


a1
c1
b1
b4
a3
E1
R1
E2
b2
a2
c2
b3
R2
ISA
partial
E5
E3
E4
g2
g1
c3
c4
c5

Appendix E Example (cont.)

- E1(a1, a3, g1), E12(a1, a2)
- E2(b1, b3, b4)
- R1(a1, b1, c1), R12(a1, b1, c2)
- E3(b1, c3, c4) , E4(b1, c5)
- E5(g1, g2)


a1
c1
学号
名
a3
E1
R1
学生
姓名
a2
c2
姓
R2
ISA
partial
E5
计算机学生
信通院学生
g2
g1
OS分数
DBS分数
通信原理分数

Have a break