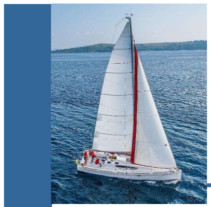
فصل ششم:طراحی پایگاه داده به کمک مدل ER

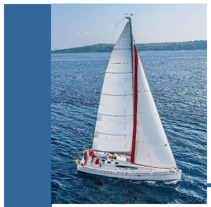
(Database Design Using the E-R Model)

درس پایگاه داده
دانشگاه صنعتی نوشیروانی بابل
مهدی عمادی
m.emadi@nit.ac.ir



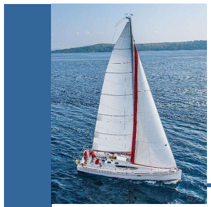
Outline

- Overview of the Design Process
- The Entity-Relationship Model
- Complex Attributes
- Mapping Cardinalities
- Primary Key
- Removing Redundant Attributes in Entity Sets
- Reducing ER Diagrams to Relational Schemas
- Extended E-R Features
- Entity-Relationship Design Issues
- Alternative Notations for Modeling Data
- Other Aspects of Database Design



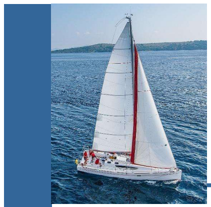
یک مسئله

- می خواهیم سیستم آموزش دانشگاه را پیاده سازی کنیم.
- شما چه می کنید؟



مراحل پیاده‌سازی پایگاه داده

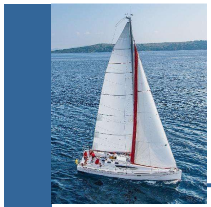
- تحلیل نیازها (Requirements Analysis)
- طراحی
- فاز اولیه: مشخص نمودن نیازهای هر یک از کاربران پایگاه داده مورد نظر
- فاز دوم: انتخاب مدل داده
  - طراحی مفهومی پایگاه داده (Conceptual Database Design)
- فاز نهایی: رفتن از مدل داده انتزاعی (Abstract)به سمت پیاده سازی
  - طراحی منطقی پایگاه داده (Logical Database Design)
  - پالایش شمای داده (Schema Refinement)
  - طراحی فیزیکی پایگاه داده (Physical Database Design)
  - طراحی برنامه و امنیت (Application and Security Design)



تحلیل نیازها (Requirements Analysis)

- چه داده‌هایی باید ذخیره گردد؟
- چه گزارش‌هایی نیاز است؟
- چه برنامه ای با این داده‌ها کار می‌کند؟
- چه قواعدی بر کسب وکار مورد نظر حاکم است؟

- شناخت وضع موجود
- شناخت نیازهای کاربر
  - مصاحبه
- شناخت وضع مطلوب



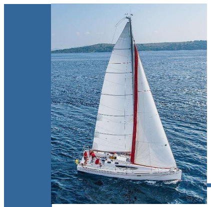
مثال

- ”نیاز به یک پایگاه داده داریم که اطلاعات دانشجو، استاد و درس در آن ذخیره شود. هر دانشجو می تواند چند درس را اخذ کند. هر استاد می تواند چند درس در طول یک ترم ارائه نماید.“

طراحی مفهومی پایگاه داده
(Conceptual Database Design)

- یک توصیف ساده از داده
- زبان مشترک بین پیاده ساز و مشتری (---)

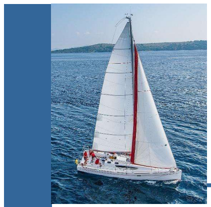
- استفاده از مدل موجودیت و رابطه بین موجودیت ها



مدل داده ای

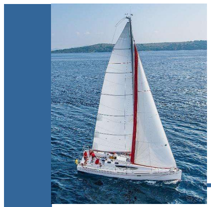
- مجموعه ای از ابزار برای بیان
  - داده ها
  - روابط میان داده ها
  - مفهوم داده ها
  - محدودیت حاکم بر داده ها
- مدلی که در حال حاضر مرسوم است مدل موجودیت ها و رابطه های بین آنها (ER) است.
- مدلهای دیگری نیز وجود دارد، از قبیل
  - مدل شئ گرا
  - سلسله مراتبی
  - و ...

چرا ما از مدل استفاده می کنیم؟



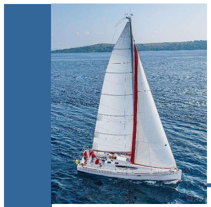
مدل Entity-Relationship

- E-R یک مدل از دنیای واقعی است. در روش ER، سه مفهوم معنایی وجود دارد و معنای داده های هر محیط به کمک همین سه مفهوم نمایش داده می شوند:
- موجودیت (entity)
  - Customer, Account, Student
- رابطه (relationship)
  - Deposit, takes, teaches
- صفت (attributes)
  - Name, age, salary
- ساختار کلی یک مدل رابطه ای را می توان به صورت یک نمودار گرافیکی به نام نمودارموجودیت-رابطه (Entity-Relationship diagram) یا به طور خلاصه ERD نمایش داد.



نوع موجودیت (Entity)

- نوع موجودیت عبارت است از کلی “شیئ”، “چیز”، “پدیده” و بطور کلی هر آنچه که می خواهیم در موردش “اطلاع” داشته باشیم و شناخت خود را در موردش افزایش دهیم، اعم از اینکه وجود فیزیکی یا ذهنی داشته باشد.
  - مانند: دانشجو، جاده، یک پرواز خاص و.....

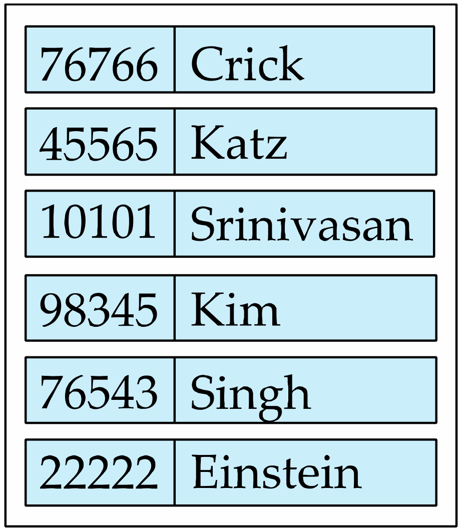


نوع موجودیت (Entity)

- نوع موجودیت عبارت است از کلی “شیئ”، “چیز”، “پدیده” و بطور کلی هر آنچه که می خواهیم در موردش “اطلاع” داشته باشیم و شناخت خود را در موردش افزایش دهیم، اعم از اینکه وجود فیزیکی یا ذهنی داشته باشد.
  - مانند: دانشجو، جاده، یک پرواز خاص و.....
  - مثال یک موجودیت با صفات آن

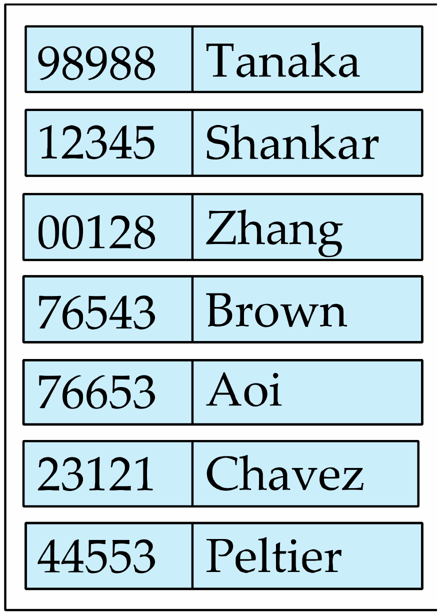
***instructor* = (*ID*, *name*, *salary*)**
***course*= (*course_id*, *title*, *credits*)**

Entity Sets -- *instructor* and *student*



76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

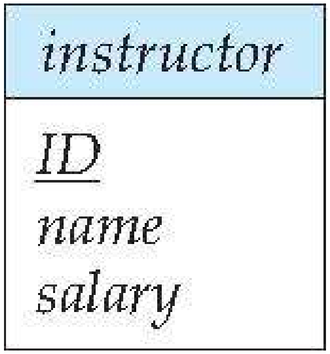


98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

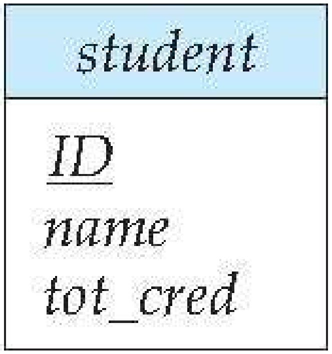
Representing Entity sets in ER Diagram

- Entity sets can be represented graphically as follows:
  - Rectangles represent entity sets.
  - Attributes listed inside entity rectangle
  - Underline indicates primary key attributes



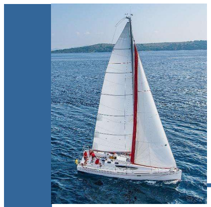
instructor

ID
name
salary



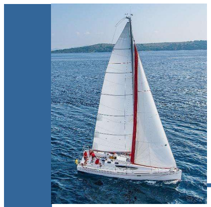
student

ID
name
tot_cred



صفت یا خصیصه (Attribute)

- صفت در واقع خصیصه یا ویژگی یک نوع موجودیت است و هر نوع موجودیت مجموعه ای از صفات دارد. هر صفت از نظر کاربران یک نام، یک نوع(Type) و یک معنای مشخص دارد.
  - مانند: یک شخص دارای یک کد ملی، نام، نام خانوادگی، قد، سن و ... است که این موارد از صفات یک شخص است.
- صفات را از پنج دیدگاه می توان تقصیم بندی کرد.

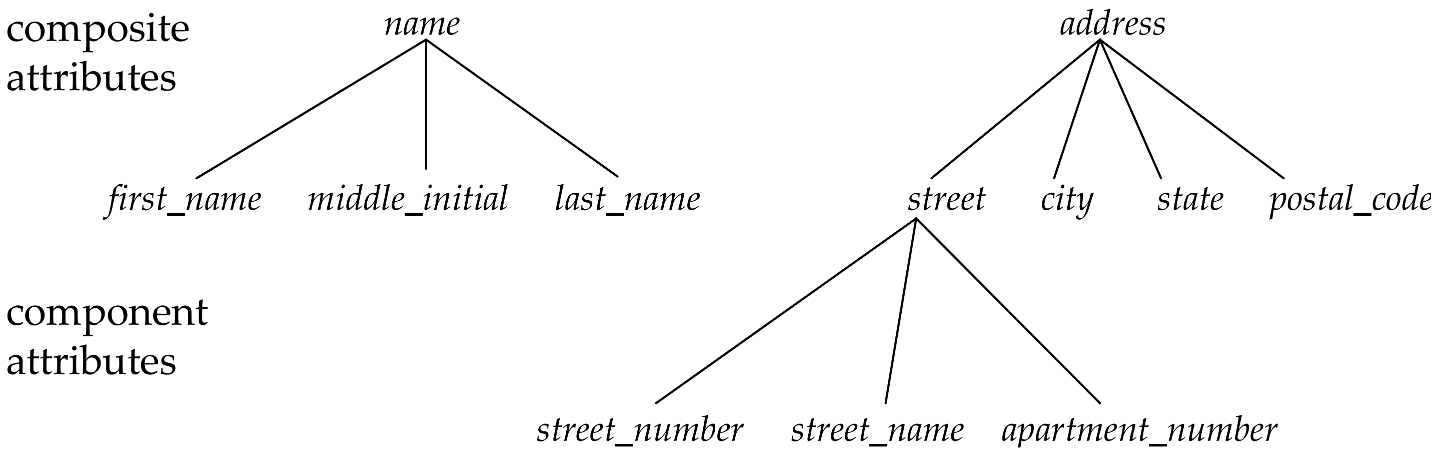


تقسیم بندی صفات از دیدگاه ۱

- ساده (Simple)
  - صفت ساده صفتی است که مقدار آن از لحاظ معنایی ساده یا تجزیه نشدنی باشد، به این معنا که اگر مقدار آن را (دریک حیطه معنایی و کاربرد مشخص) به اجزایی تجزیه کنیم، مقادیر جزیی حاصله فاقد معنا باشند.
  - مثلا صفت نام کوچک شخص یک صفت ساده است.
- مرکب (composite)
  - صفت مرکب صفتی است که از چند صفت ساده تشکیل شده باشد به گونه ای که تجزیه نشدنی باشد و اجزای حاصل از تجزیه، خود صفات ساده (و طبعا دارای معنا در یک کاربرد مشخص) باشند.
  - مثلا صفت تاریخ خود شامل روز، ماه و سال است.

Composite Attributes

- **Composite attributes allow us to divided attributes into subparts (other attributes).**



composite attributes

name

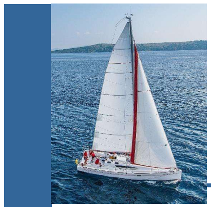
first_name *middle_initial* *last_name*

address

street *city* *state* *postal_code*

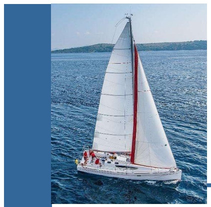
component attributes

street_number *street_name* *apartment_number*



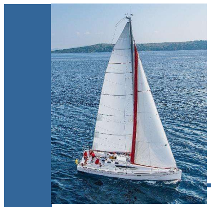
تقسیم بندی صفات از دیدگاه ۲

- تک مقداری (Single-valued)
  - صفت تک مقداری صفتی است که برای یک نمونه از یک موجودیت، حداکثر یک مقدار مقدار از دامنه مقادیر را می گیرد. به بیان ساده تر، به ازای یک نام صفت، حداکثر یک مقدار برای یک نمونه از موجودیت، داشته باشیم.
  - مثلا شماره ملی: یک نمونه از شخص فقط یک شماره ملی می تواند داشته باشد.
- چند مقداری (multivalued)
  - صفت چند مقداری صفتی است که برای بعض یا همه نمونه های نوع موجودیت بیش از یک مقدار، از دامنه مقادیر را می گیرد. به بیان ساده تر، به ازاء یک نام صفت، چند مقدار، برای یک نمونه از موجودیت داشته باشیم.
  - برای مثال اگر شغل فرد را یک صفت برای او در نظر بگیریم آن شخص می تواند چند شغل داشته باشد.



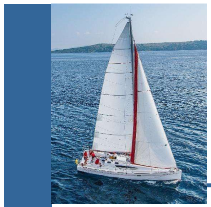
تقسیم بندی صفات از دیدگاه ۳

- شناسه (identifier یا Primary Key)یا ناشناسه موجودیت
  - صفت شناسه موجودیت صفتی است که دو ویژگی داشته باشد:
  - یکتایی مقدار داشته باشد:
    - یعنی در هیچ دو نمونه از یک موجودیت، مقدارش یکسان نباشد. بنابراین عامل تمییز دو نمونه از یک موجودیت است.
    - از این ویژگی ها نتیجه می شود که صفت شناسه، از نظر کاربر، شناسای نوع موجودیت است و متمایز کننده نمونه های آن نوع موجودیت از یکدیگر، و همیشه مقادیرش مشخص و موجود است.
    - مثلا کد ملی
  - حتی الامکان طول مقادیرش کوتاه باشد.
    - مثلا سن افراد



تقسیم بندی صفات از دیدگاه ۴

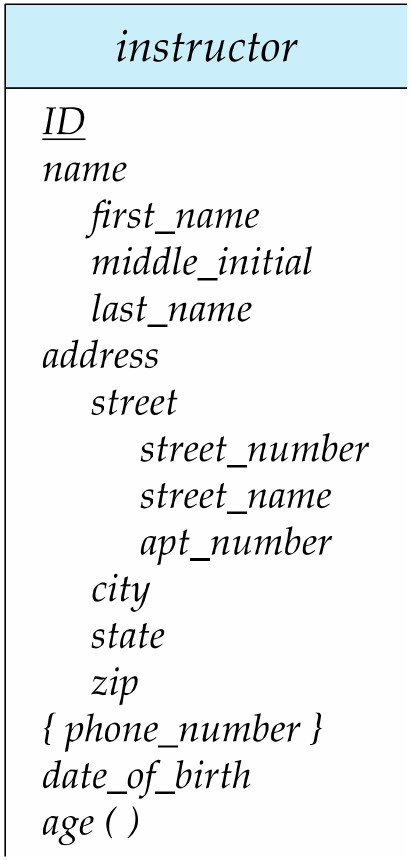
- هیچمقدار پذیر (Nullable) یا هیچمقدار ناپذیر (Not-Nullable)
  - هیچمقدار یعنی: مقدار ناشناخته، مقدار غیر قابل اعمال، مقدار تعریف نشده.
  - همیشه ممکن است مقدار یک صفت برای برخی از نمونه های یک موجودیت، ناشناخته، ناموجود و ... باشد.
  - به زبان دیگری وقتی شما فرم درخواست کاغذی را پر می کنید ممکن از تعدادی از قسمتها را خالی بگذارید.
  - البته بسیاری از طراحان سیستمها معتقدند، باید طراحی را طوری اصلاح کرد که امکان بروز صفت هیچمقدار وجود نداشته باشد.



تقسیم بندی صفات از دیدگاه ۵

- ذخیره شده (واقعی)
  - صفت ذخیره شده صفتی است که مقادیرش در پایگاه داده ذخیره شده باشند (موجود باشند) و البته هیچمقدار هم داشته باشد، اگر شناسه نباشد.
- مشتق (**Derived**)
  - صفت مشتق صفتی است که مقادیرش در پایگاه داده ذخیره نباشند بلکه حاصل یک پردازش روی فقره هایی از داده های ذخیره شده باشند، مثلا از یک محاسبه بدست آید.
    - ما در پایگاه داده نمرات دروس یک دانشجو را داریم. با فرض اینکه معدل کل یک دانشجو یکی از صفات او به حساب آید می توان هر بار که نیاز به آن داشتیم از جمع کل نمرات او و تقسیم آن بر تعداد آنها معدل او را بدست آورد و لزومی ندارد که حتما در پایگاه داده عینا ذخیره شود. (آیا همیشه لزومی ندارد!!؟؟)

Representing Complex Attributes in ER Diagram

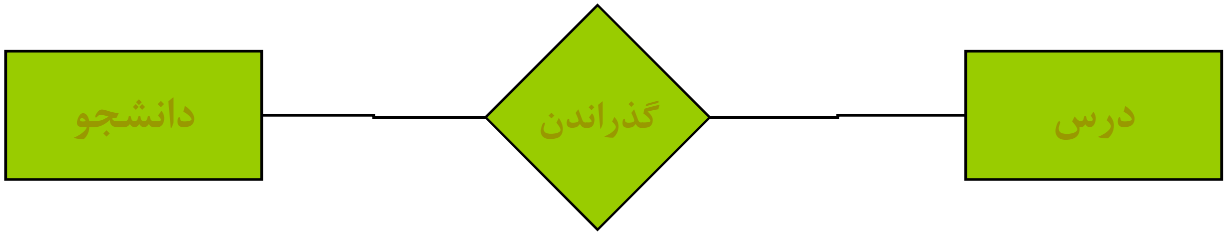


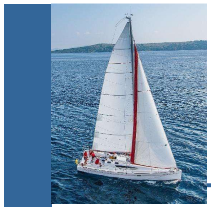
instructor

- *ID*
- *name*
  - *first_name*
  - *middle_initial*
  - *last_name*
- *address*
  - *street*
    - *street_number*
    - *street_name*
    - *apt_number*
  - *city*
  - *state*
  - *zip*
- *{ phone_number }*
- *date_of_birth*
- *age ( )*

ارتباط

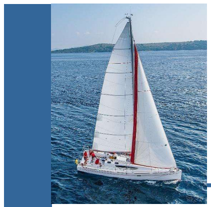
- ارتباط یا بستگی مفهومی است بسیار مهم در مدلسازی معنایی داده ها. بین انواع موجودیتها، معمولا ارتباط (ارتباطی) برقرار است.
- تعریف نوع ارتباط
  - نوع ارتباط عبارت است از اندرکنش (تعامل) بین دو یا بیش از دو نوع موجودیت (و یا بین یک نوع موجودیت و خودش) و ماهیتا نوعی بستگی بین انواع موجودیتهاست.
  - هر نوع ارتباط یک معنای مشخص دارد و با یک نام بیان می شود. و نیز می توان گفت که نوع ارتباط، عملی است که بین انواع موجودیتها جاری بوده، هست و خواهد بود.
  - برای مثال هر وقت که می گوییم شخصی یک شغل دارد یعنی با یک شغل در ارتباط است.





نوع ارتباط

- خصوصیات کلی
  - هر ارتباط یک نام دارد: معمولا کلمه یا عبارتی فعلی و نه اسمی
  - هر نوع ارتباط یک معنای مشخص دارد . این معنا با معنای هر نوع ارتباط دیگر متفاوت است. واقع شدن و در اختیار داشتن
  - هر نوع ارتباط نمونه هایی دارد. یعنی چند مثال در دنیای واقعی



Relationship Sets

- A **relationship** is an association among several entities

  Example:

22222 (Einstein)	*advisor*	44553 (Peltier)
instructor entity	relationship set	*student* entity

- A **relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

$$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

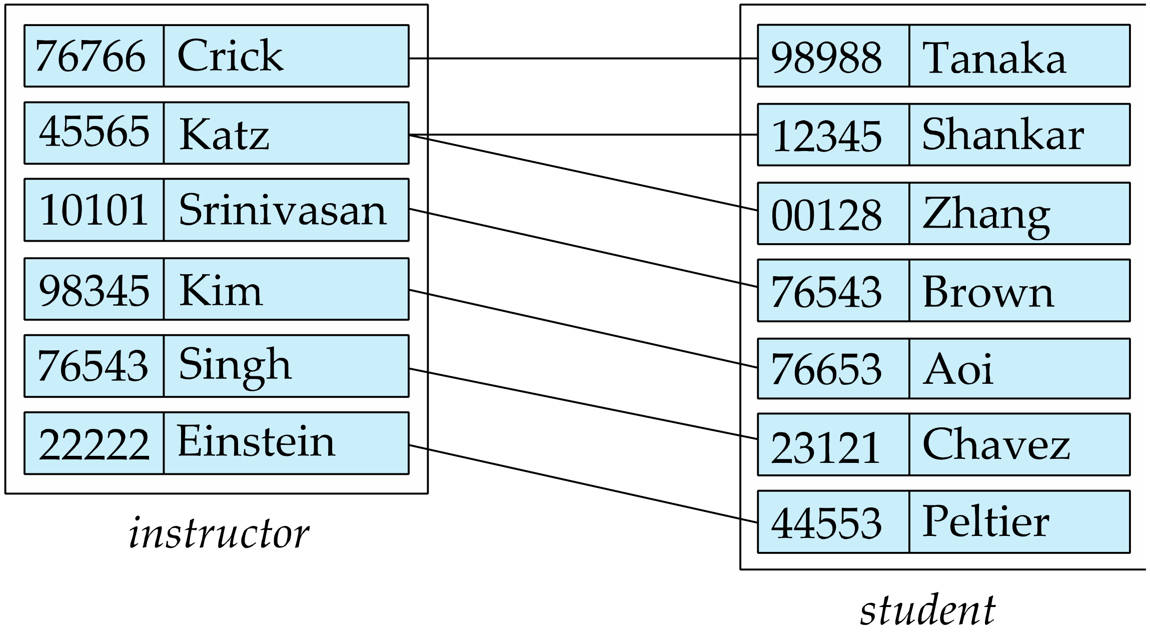
where $(e_1, e_2, \ldots, e_n)$ is a relationship

- Example:

  $(44553,22222) \in$ *advisor*

Relationship Sets (Cont.)

- Example: we define the relationship set *advisor* to denote the associations between students and the instructors who act as their advisors.
- Pictorially, we draw a line between related entities.



76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

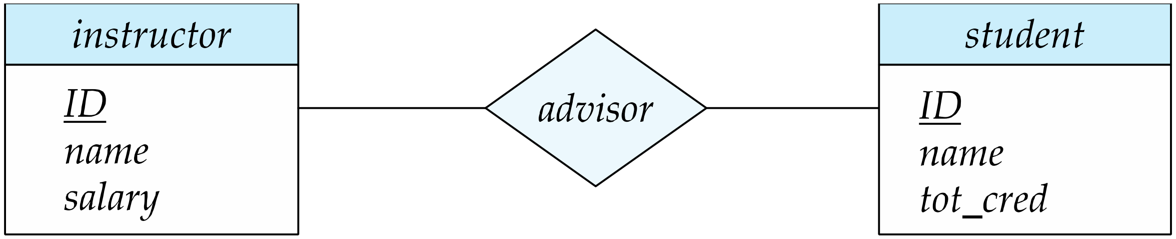
instructor

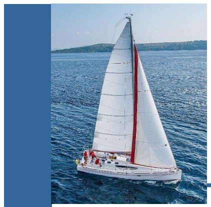
98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Representing Relationship Sets via ER Diagrams

- Diamonds represent relationship sets.

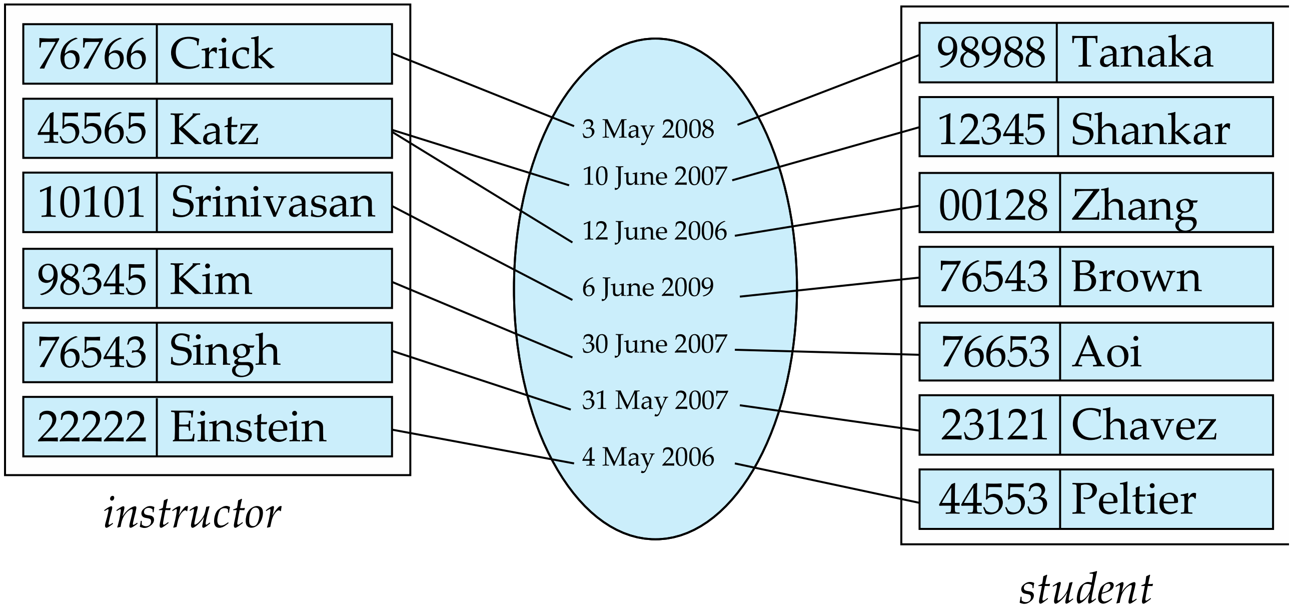




Relationship Sets (Cont.)

- ارتباط می تواند صفت یا صفاتی داشته باشد، اما معمولا فاقد صفت شناسه است.
- مثلا می خواهیم بگوییم یک مدیر در یک سازمان مدیریت می کند، در این مثال زمان شروع مدیریت می تواند صفت این ارتباط باشد.

- **For instance, the *advisor* relationship set between entity sets *instructor* and *student* may have the attribute *date* which tracks when the student started being associated with the advisor**



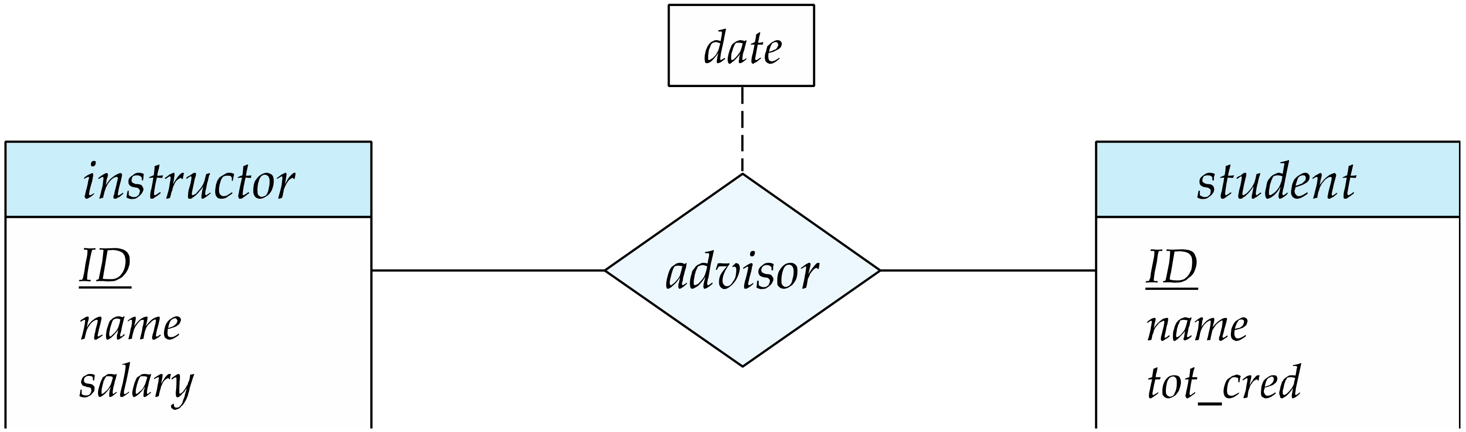
76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

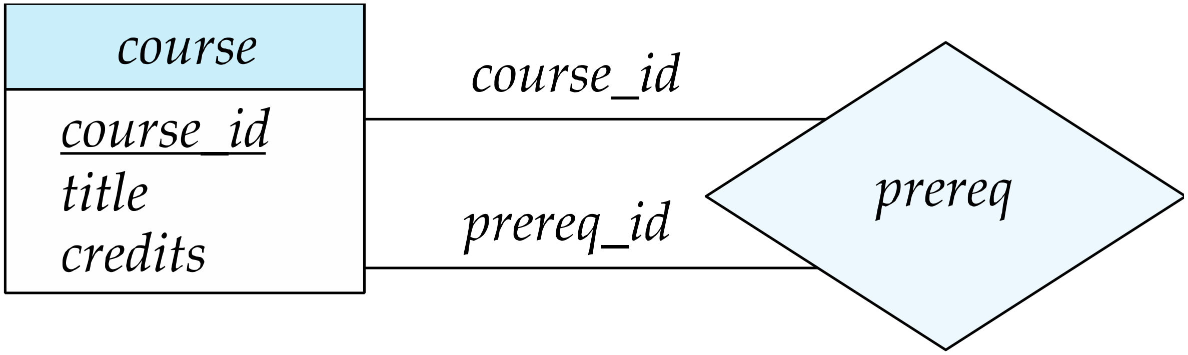
student

Relationship Sets with Attributes

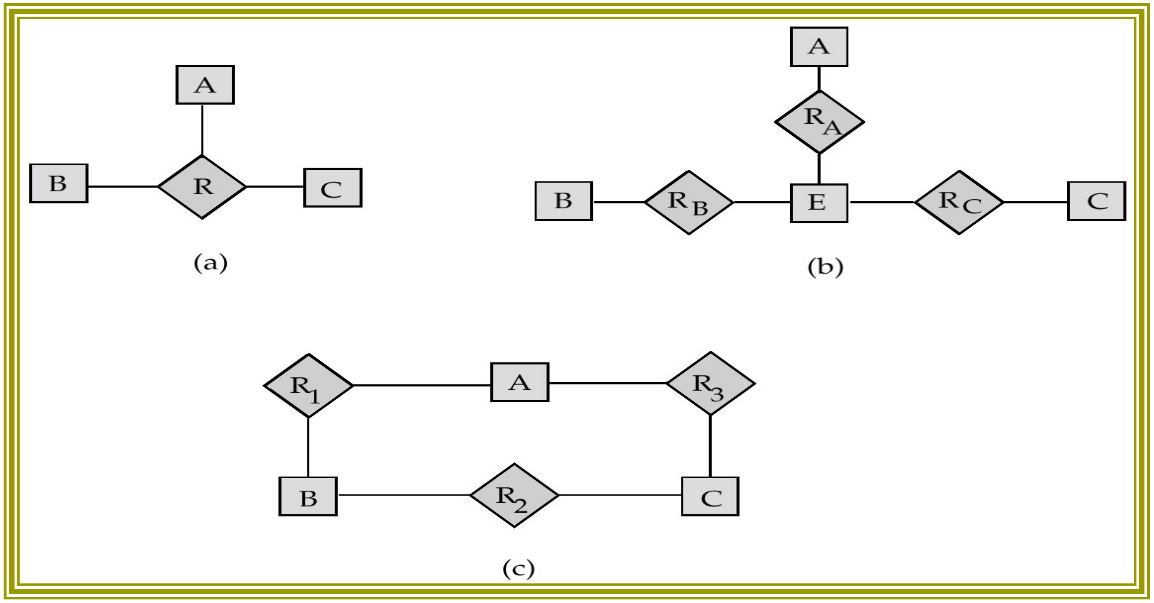


رابطه همانی و نقش (Role) در رابطه

- 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.



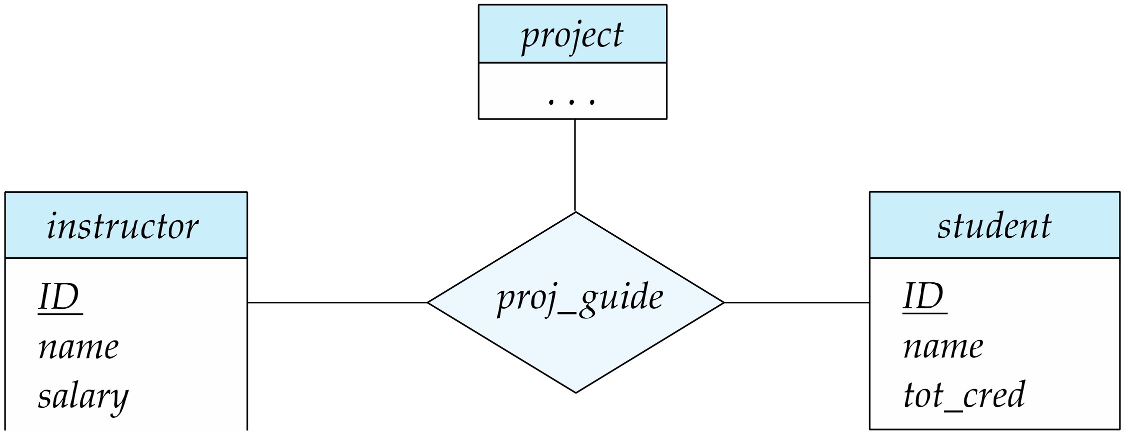
درجه نوع ارتباط (Degree of a Relationship)

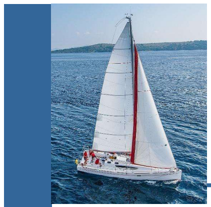


- تعداد شرکت کنندگان در یک نوع ارتباط را درجه آن ارتباط گوییم و این اصطلاحات را داریم: دوگانی(Binary) و سه گانی
- (a) یک رابطه سه گانی (Ternary) را نشان می دهد.
- (b) و (c) سه رابطه دوگانی را نشان می دهند.

Non-binary Relationship Sets

- Most relationship sets are binary
- There are occasions when it is more convenient to represent relationships as non-binary.
- E-R Diagram with a Ternary Relationship

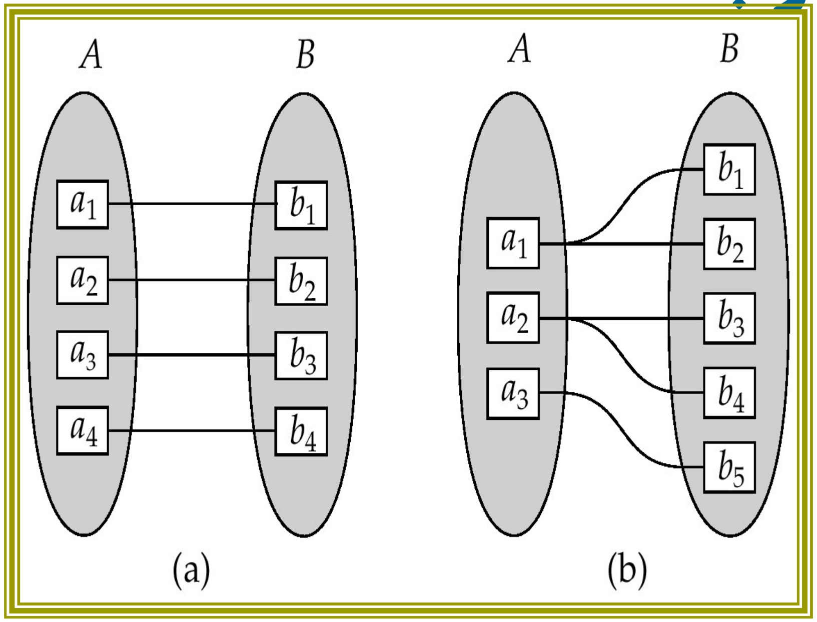


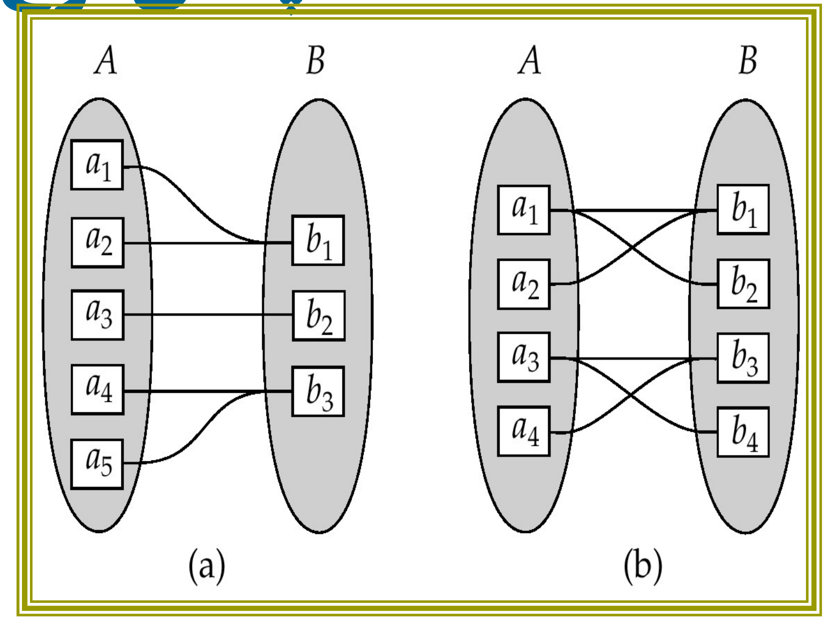


نگاشت محدودیت چندی رابطه
(Mapping Cardinality Constraints)

- چندی و ماهیت نوع ارتباط
  - چندی یک نوع ارتباط (که به آن کاردینالیتی ارتباط هم می گویند) مثلا بین دو موجودیت عبارتست از چگونگی تناظر بین دو مجموعه نمونه های آن دو موجودیت. می دانیم که سه گونه تناظر داریم:
    - تناظر یک به یک 1 : 1
      - یک سازمان یک مدیر کل دارد و هر مدیر کل فقط مدیر کل یک سازمان است.
    - تناظر یک به چند 1 : N
      - یک مادر چند فرزند دارد و هر فرزند یک مادر دارد.
    - تناظر چند به چند N : M
      - هر دانشجو چند درس می گذراند و هر درس توسط چند دانشجو گذرانده می شود.

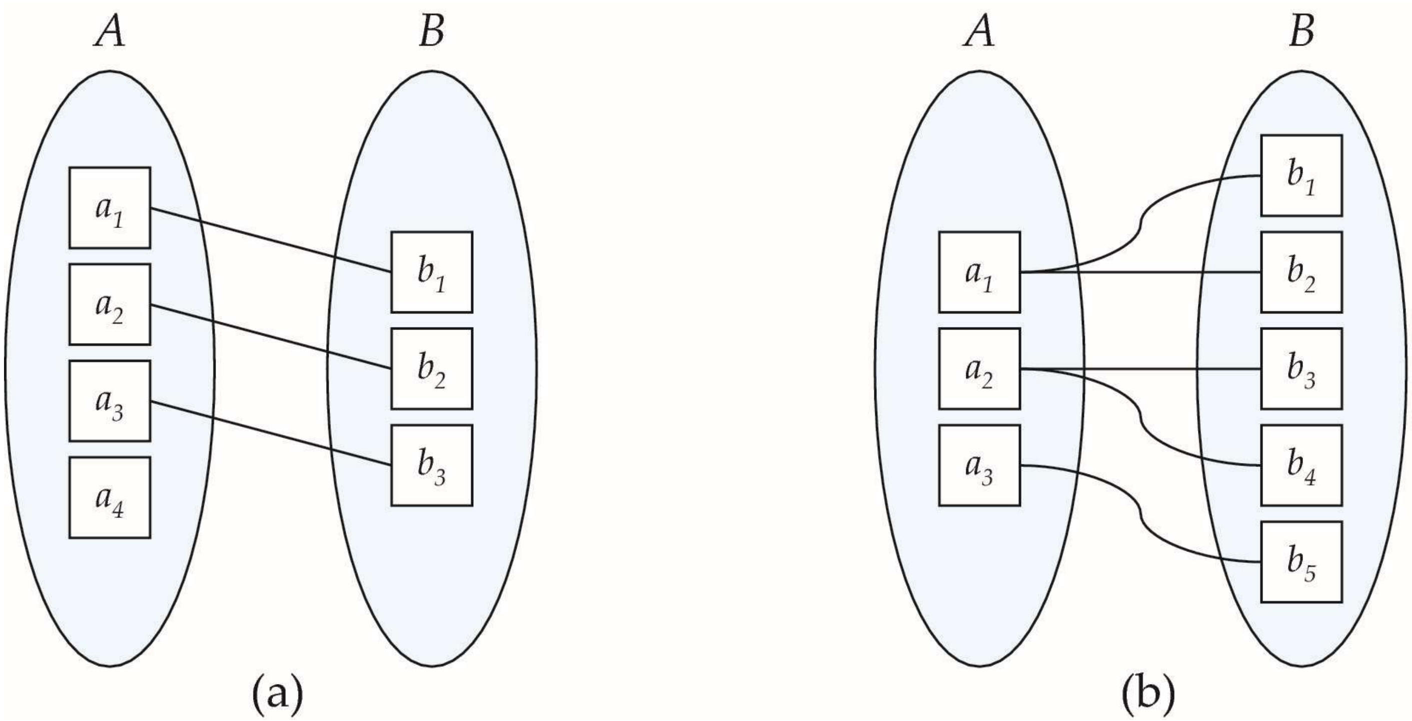
چندی نوع ارتباط





- (a) تناظر یک به یک 1 : 1
- (b) و (c) تناظر یک به چند 1 : N
- (d) تناظر چند به چند N : M

Mapping Cardinalities

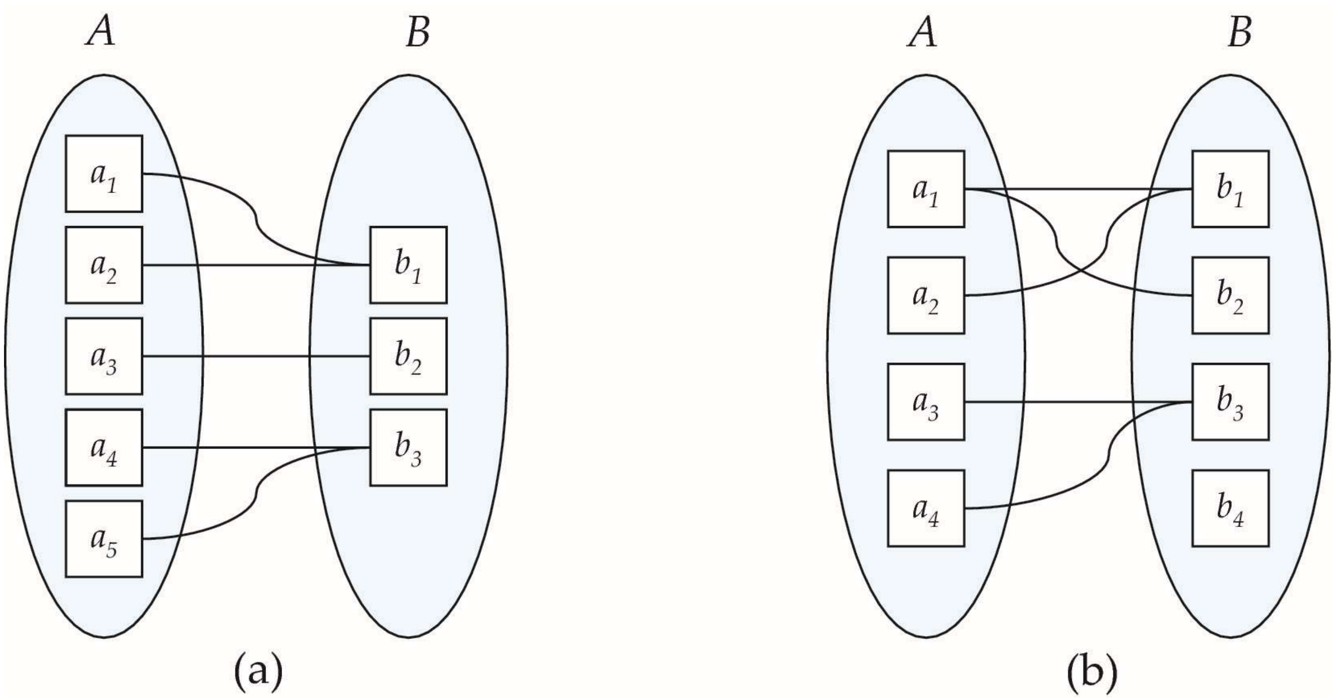


One to one

One to many

Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

Mapping Cardinalities



Many to one

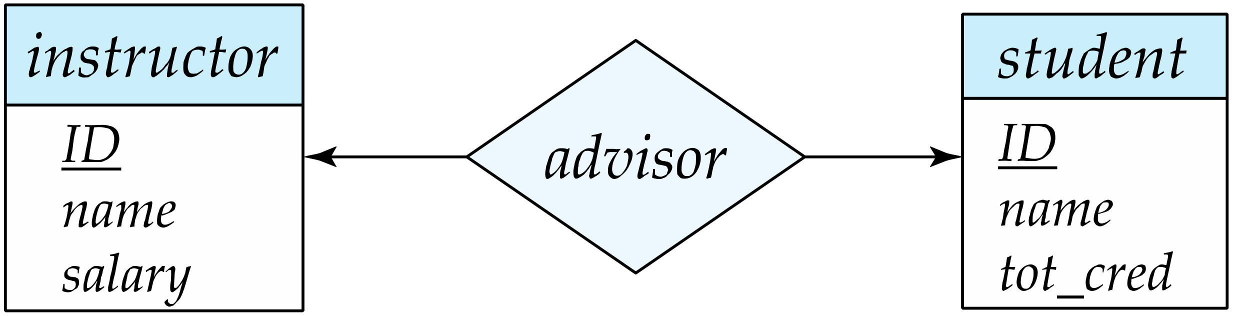
Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

Representing Cardinality Constraints in ER Diagram

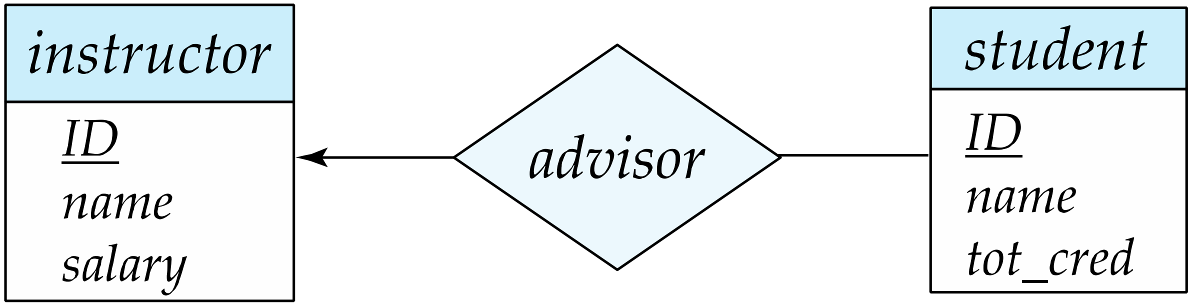
- We express cardinality constraints by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set.

- One-to-one relationship between an *instructor* and a *student* :
  - A student is associated with at most one *instructor* via the relationship *advisor*
  - A *student* is associated with at most one *department* via *stud_dept*



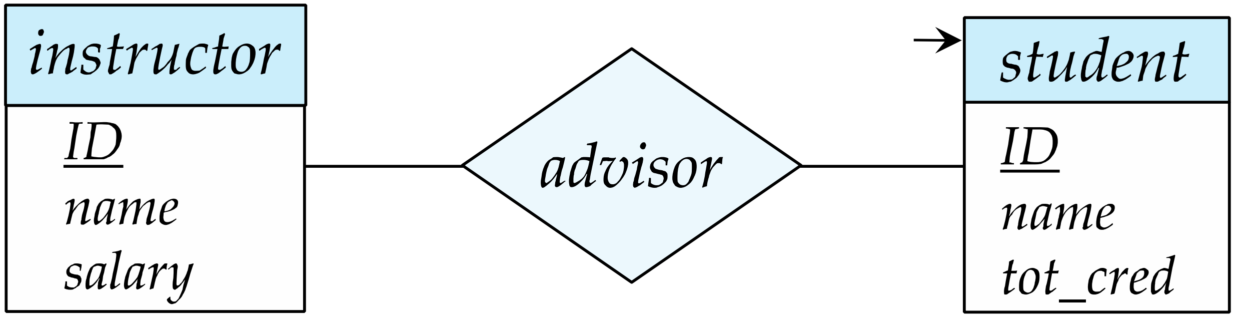
One-to-Many Relationship

- one-to-many relationship between an *instructor* and a *student*
  - an instructor is associated with several (including 0) students via *advisor*
  - a student is associated with at most one instructor via advisor,



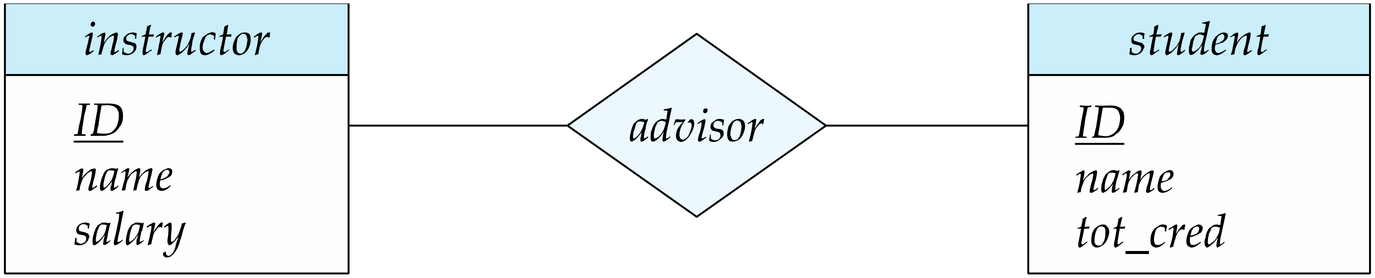
Many-to-One Relationships

- In a many-to-one relationship between an *instructor* and a *student*,
  - an instructor is associated with at most one student via *advisor*,
  - and a student is associated with several (including 0) instructors via *advisor*



Many-to-Many Relationship

- An instructor is associated with several (possibly 0) students via *advisor*
- A student is associated with several (possibly 0) instructors via *advisor*

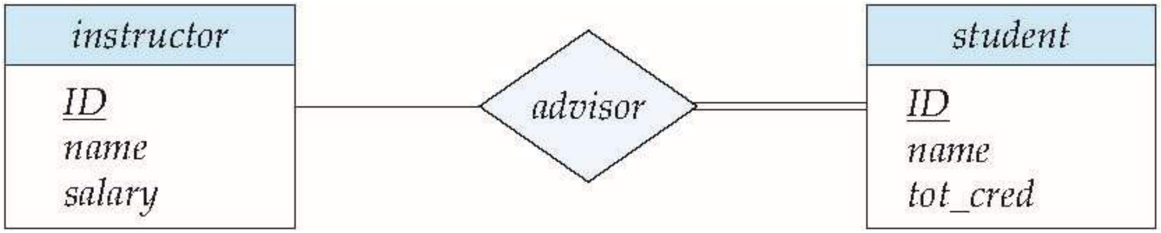


وضع مشارکت در ارتباط

- وضع مشارکت در ارتباط
  - مشارکت یک نوع موجودیت در یک نوع ارتباط ممکن است الزامی (کامل یا Total) یا غیرالزامی (نا کامل یا Partial) باشد.
  - برای مثال هر سازمانی حتما یک مدیر دارد و ما سازمان بدون مدیر نداریم پس این رابطه یک رابه الزامی است.

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



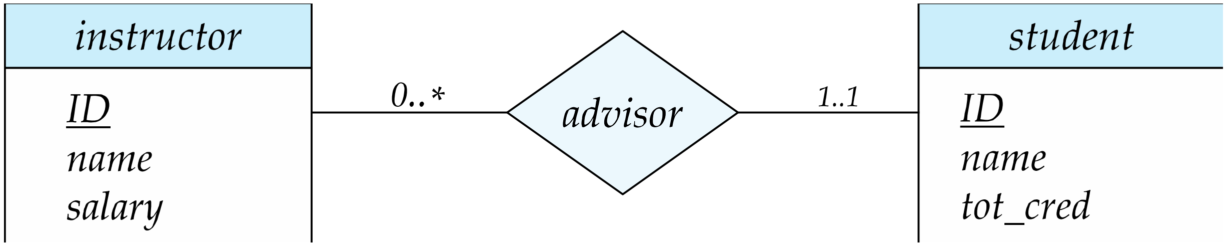
participation of *student* in *advisor* relation is total

  - every *student* must have an associated instructor

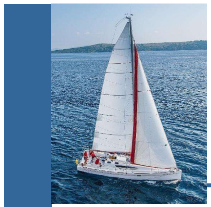
- **Partial participation**: some entities may not participate in any relationship in the relationship set
  - Example: participation of *instructor* in *advisor* is partial

Notation for Expressing More Complex Constraints

- 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

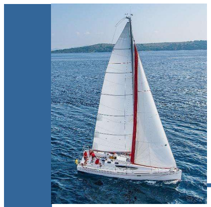


Cardinality Constraints on Ternary Relationship

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

Primary Key

- Primary keys provide a way to specify how entities and relations are distinguished. We will consider:
  - Entity sets
  - Relationship sets.
  - Weak entity sets

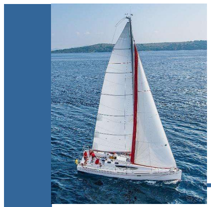


Primary key for Entity Sets

- By definition, individual entities are distinct.
- From database perspective, the differences among them must be expressed in terms of their attributes.
- The values of the attribute values of an entity must be such that they can uniquely identify the entity.
  - No two entities in an entity set are allowed to have exactly the same value for all attributes.
- A key for an entity is a set of attributes that suffice to distinguish entities from each other

Primary Key for Relationship Sets

- To distinguish among the various relationships of a relationship set we use the individual primary keys of the entities in the relationship set.
  - Let *R* be a relationship set involving entity sets E1, E2, .. En
  - The primary key for R is consists of the union of the primary keys of entity sets E1, E2, ..En
  - If the relationship set *R* has attributes a1, a2, .., am associated with it, then the primary key of *R* also includes the attributes a1, a2, .., am
- Example: relationship set "advisor".
  - The primary key consists of *inrsructor.ID* and s*tudent.ID*
- The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set.

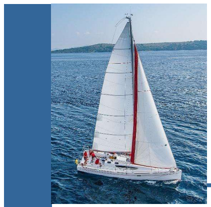


Choice of Primary key for Binary Relationship

- Many-to-Many relationships.   The preceding union of the primary keys is a minimal superkey and is chosen as the primary key.
- One-to-Many relationships . The primary key of the "Many" side is a minimal superkey and is used as the primary key.
- Many-to-one relationships. The primary key of the "Many" side is a minimal superkey and is used as the primary key.
- One-to-one relationships. The primary key of either one of the participating entity sets forms a minimal superkey, and either one can be chosen as the primary key.

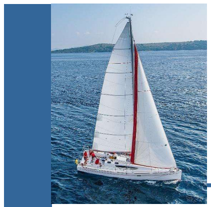
Choice of Primary key for Nonbinary Relationship

- If no cardinality constraints are present, the superkey is formed as described earlier. and it is chosen as the primary key.
- If there are cardinality constraints are present:
  - Recall that we permit at most one arrow out of a relationship set.
  - AVI



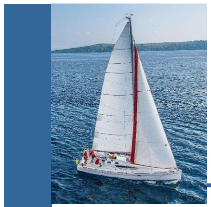
موجودیت ضعیف
(Weak Entity)

- موجودیتی است که در محدوده مدل‌سازی ما به یک موجودیت دیگر وابسته بوده و توسط آن موجودیت (**identifying owner**) شناخته می‌شود.
- نوع مشارکت موجودیت ضعیف در رابطه الزامی می‌باشد
- چندی رابطه همواره یک به چند می‌باشد (در برخی حالات خاص می‌تواند یک به کی باشد)
- **Partial Key**



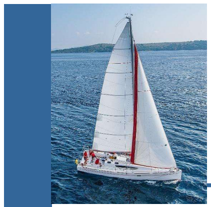
Weak Entity Sets

- Consider a *section* entity, which is uniquely identified by a *course_id*, *semester*, *year*, and *sec_id*.
- Clearly, section entities are related to course entities. Suppose we create a relationship set *sec_course* between entity sets *section* and *course*.
- Note that the information in *sec_course* is redundant, since *section* already has an attribute *course_id*, which identifies the course with which the section is related.
- One option to deal with this redundancy is to get rid of the relationship s*ec_course*; however, by doing so the relationship between *section* and *course* becomes implicit in an attribute, which is not desirable.



Weak Entity Sets (Cont.)

- An alternative way to deal with this redundancy is to not store the attribute *course_id* in the *section* entity and to only store the remaining attributes *section_id*, *year*, and *semester*.
  - However, the entity set *section* then does not have enough attributes to identify a particular *section* entity uniquely
- To deal with this problem, we treat the relationship *sec_course* as a special relationship that provides extra information, in this case, the *course_id*, required to identify *section* entities uniquely.
- A **weak entity set** is one whose existence is dependent on another entity, called its **identifying entity**
- Instead of associating a primary key with a weak entity, we use the identifying entity, along with extra attributes called **discriminator** to uniquely identify a weak entity.

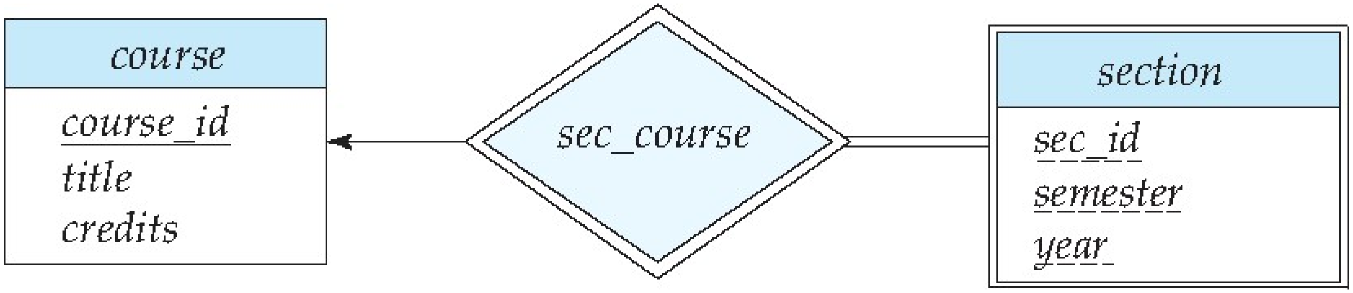


Weak Entity Sets (Cont.)

- An entity set that is not a weak entity set is termed a **strong entity set.**
- Every weak entity must be associated with an identifying entity; that is, the weak entity set is said to be **existence dependent** on the identifying entity set.
- The identifying entity set is said to **own** the weak entity set that it identifies.
- The relationship associating the weak entity set with the identifying entity set is called the **identifying relationship**.
- Note that the relational schema we eventually create from the entity set *section* does have the attribute *course_id*, for reasons that will become clear later, even though we have dropped the attribute *course_id* from the entity set *section.*

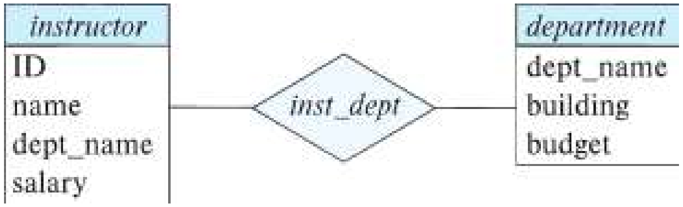
Expressing Weak Entity Sets

- 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*)

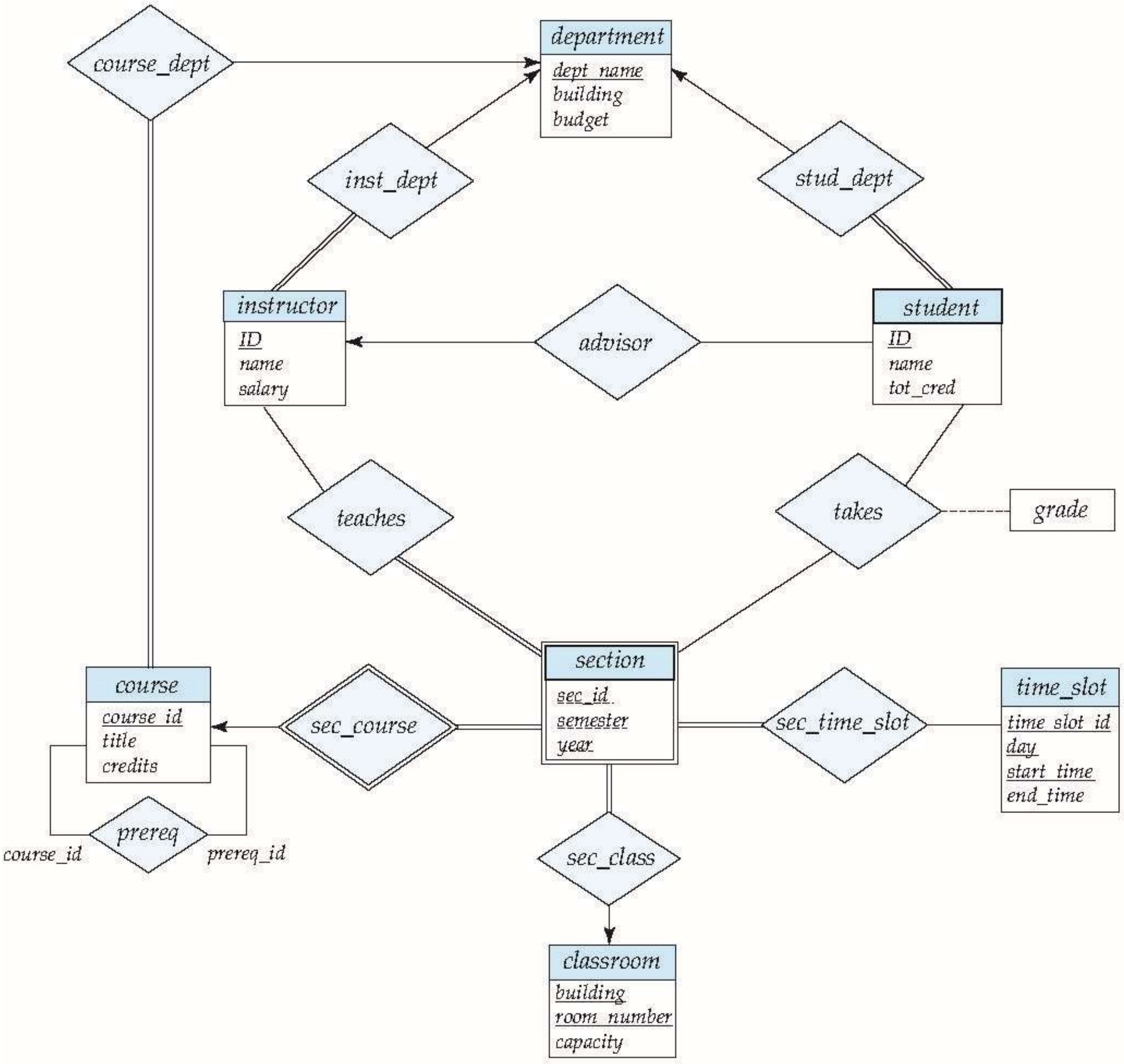


Redundant Attributes

- Suppose we have entity sets:
  - *instructor*, with attributes: *ID*, *name*, *dept_name, salary*
  - *department,* with attributes: *dept_name, building, budget*
- We model the fact that each instructor has an associated department using a relationship set *inst_dept*
- The attribute *dept_name* in *instructor* replicates information present in the relationship and is therefore redundant
  - and needs to be removed.
- BUT: when converting back to tables, in some cases the attribute gets reintroduced, as we will see later.



E-R Diagram for a University Enterprise



Reduction to Relation Schemas

Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collection of schemas.
- For each entity set and relationship set there is a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

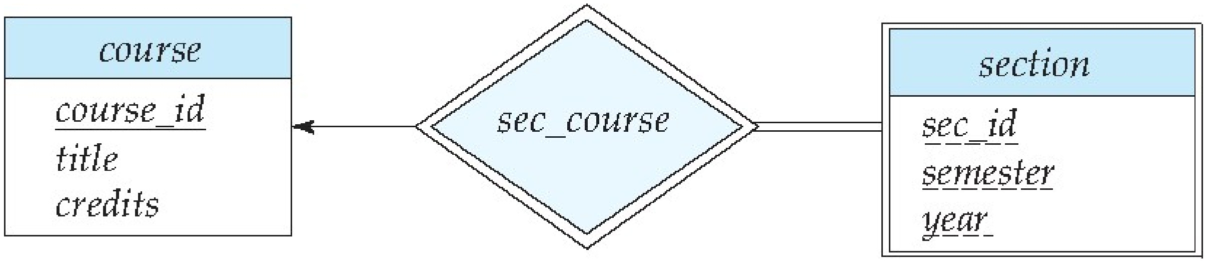
Representing Entity Sets

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

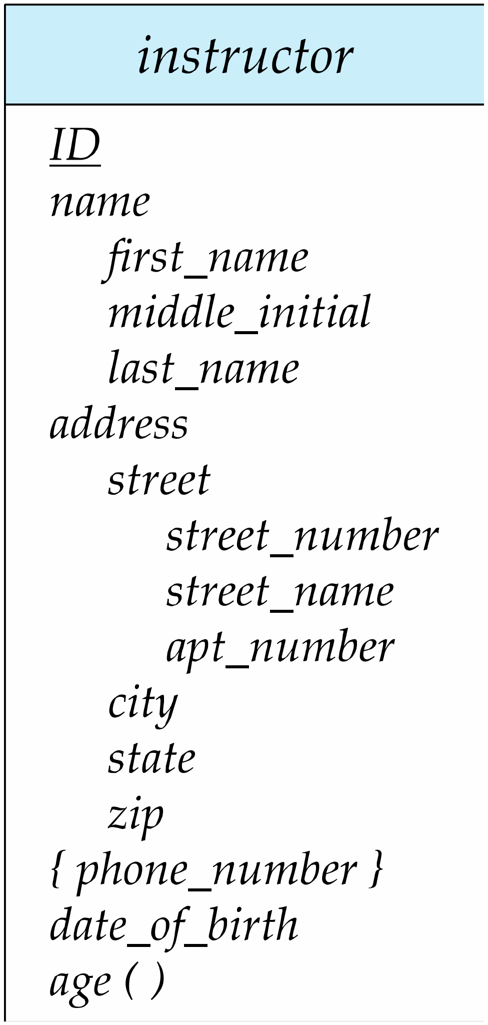
  student(ID, name, tot_cred)

- 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 Entity Sets with Composite Attributes



instructor

- *ID*
- *name*
  - *first_name*
  - *middle_initial*
  - *last_name*
- *address*
  - *street*
    - *street_number*
    - *street_name*
    - *apt_number*
  - *city*
  - *state*
  - *zip*
- *{ phone_number }*
- *date_of_birth*
- *age ( )*

- **Composite attributes are flattened out by creating a separate attribute for each component attribute**
  - **Example: given entity set *instructor* with composite attribute *name* with component attributes *first_name* and *last_name* the schema corresponding to the entity set has two attributes *name_first_name* and *name_last_name***
    - **Prefix omitted if there is no ambiguity (*name_first_name* could be *first_name)***
- **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)

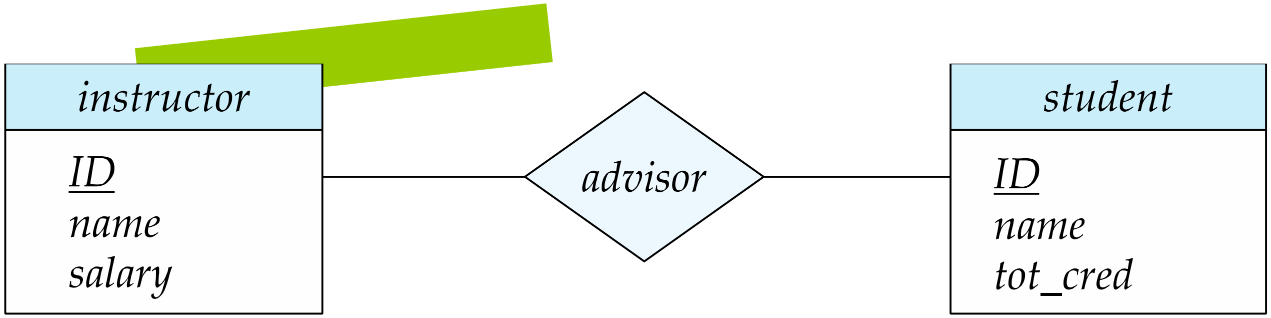
Representation of Entity Sets with Multivalued Attributes

- 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)

Representing Relationship Sets

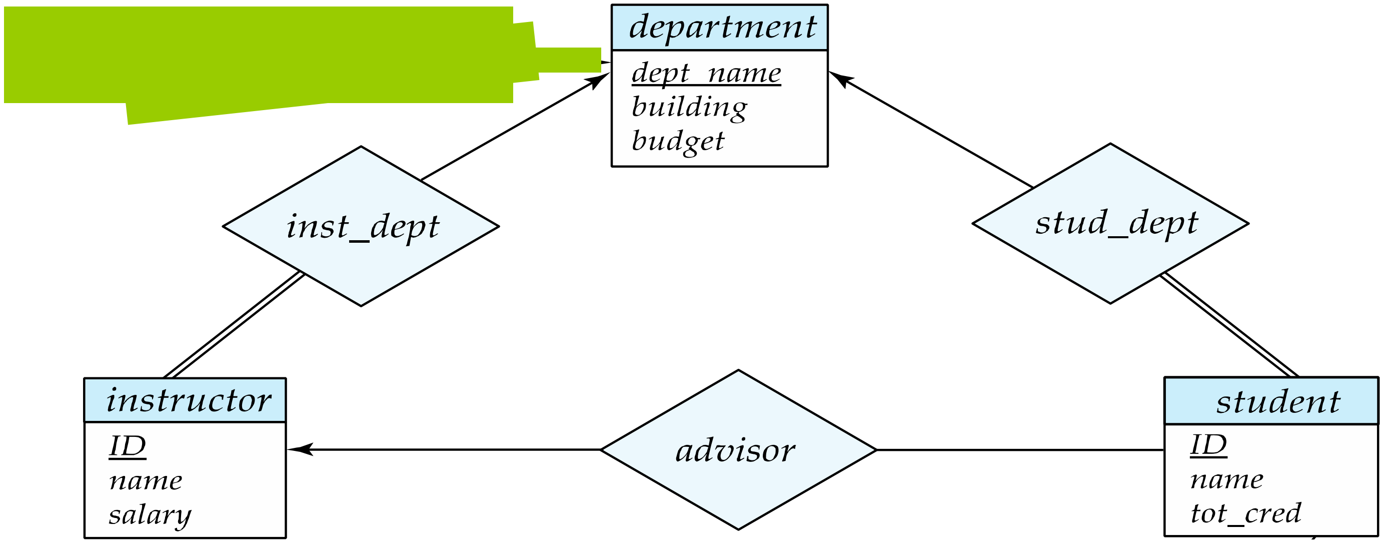
- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set *advisor*

***advisor* = (*<u>s_id, i_id</u>*)**

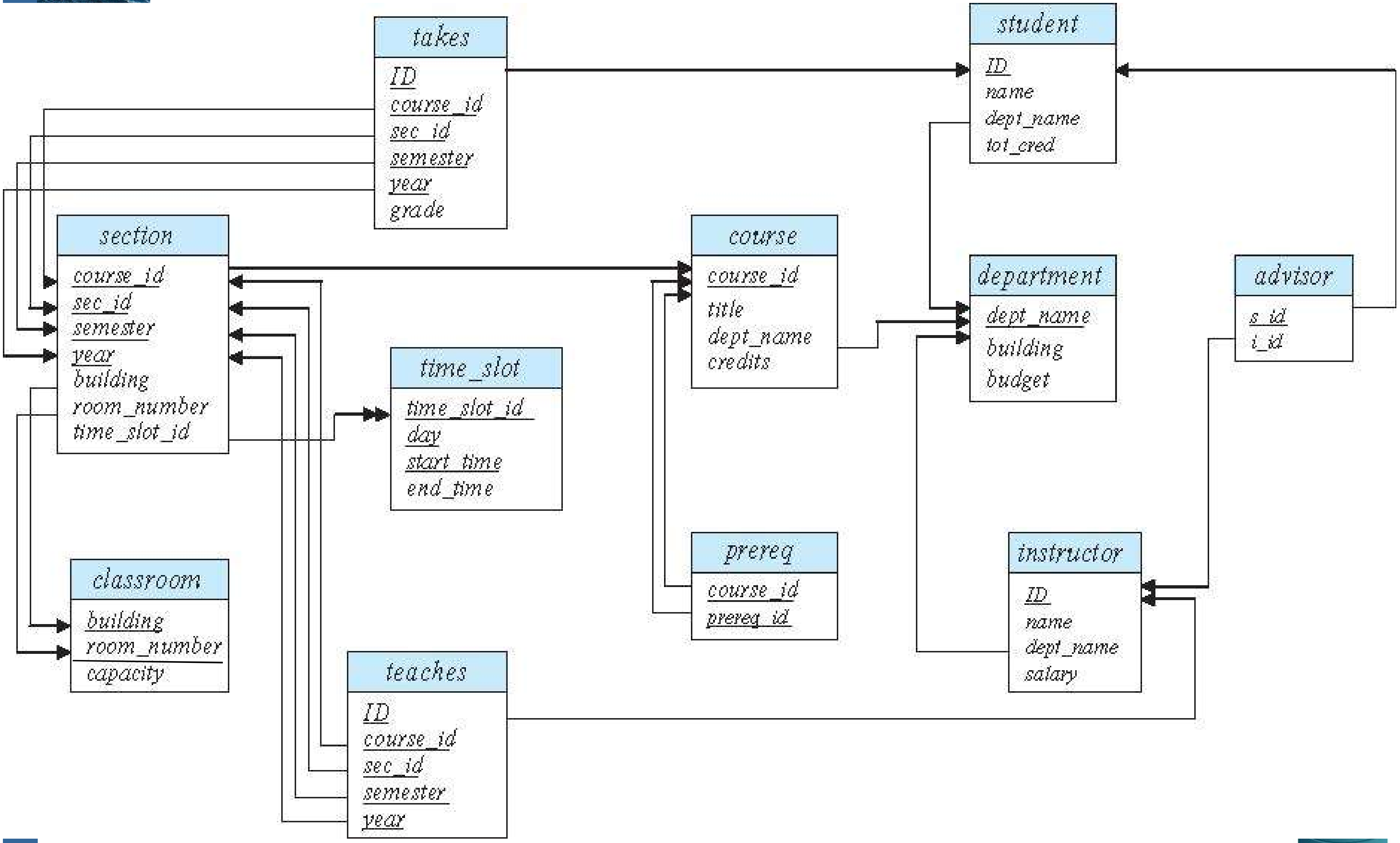


Redundancy of Schemas

- 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
- Example: Instead of creating a schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from entity set *instructor*



Schema Diagram for University Database



takes
ID
course_id
sec_id
semester
year
grade

student
ID
name
dept_name
tot_cred

section
course_id
sec_id
semester
year
building
room_number
time_slot_id

course
course_id
title
dept_name
credits

department
dept_name
building
budget

advisor
s_id
i_id

time_slot
time_slot_id
day
start_time
end_time

classroom
building
room_number
capacity

prereq
course_id
prereq_id

instructor
ID
name
dept_name
salary

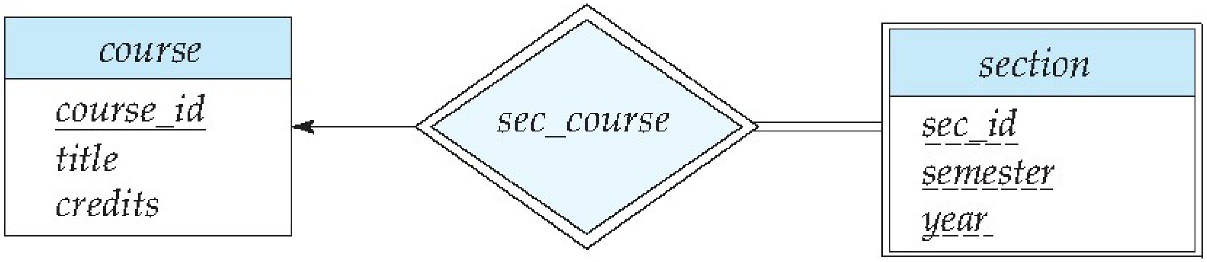
teaches
ID
course_id
sec_id
semester
year

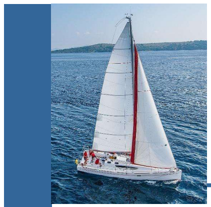
Redundancy of Schemas (Cont.)

- For one-to-one relationship sets, either side can be chosen to act as the “many” side
  - That is, an extra attribute can be added to either of the tables corresponding to the two entity sets
- If participation is *partial* on the “many” side, replacing a schema by an extra attribute in the schema corresponding to the “many” side could result in null values

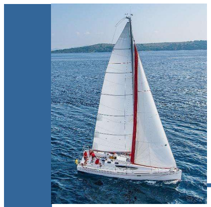
Redundancy of Schemas (Cont.)

- **The schema corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.**
- **Example: The *section* schema already contains the attributes that would appear in the *sec_course* schema**





Extended E-R Features

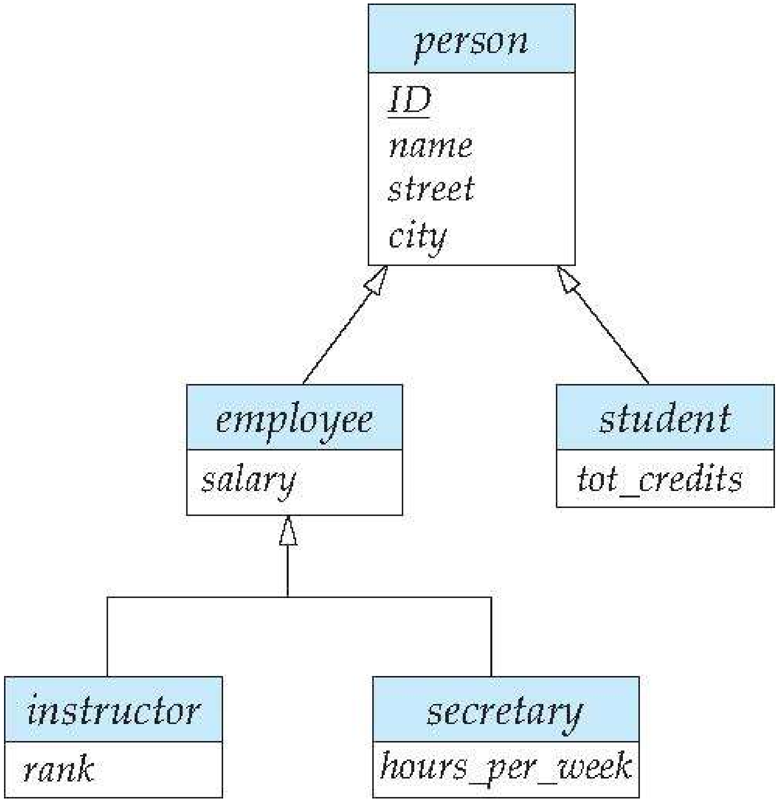


Specialization

- Top-down design process; we designate sub-groupings within an entity set that are distinctive from other entities in the set.
- These sub-groupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set.
- Depicted by a *triangle* component labeled ISA (e.g., *instructor* "is a" *person*).
- 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.

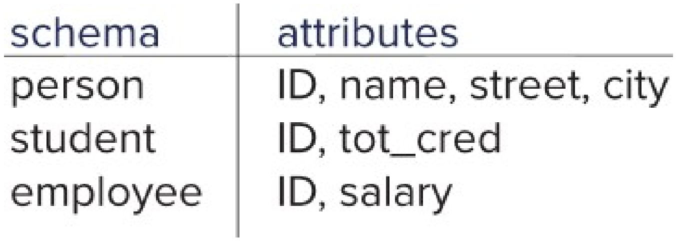
Specialization Example

- **Overlapping – *employee* and *student***
- **Disjoint – *instructor* and *secretary***
- **Total and partial**



Representing Specialization via Schemas

- **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**

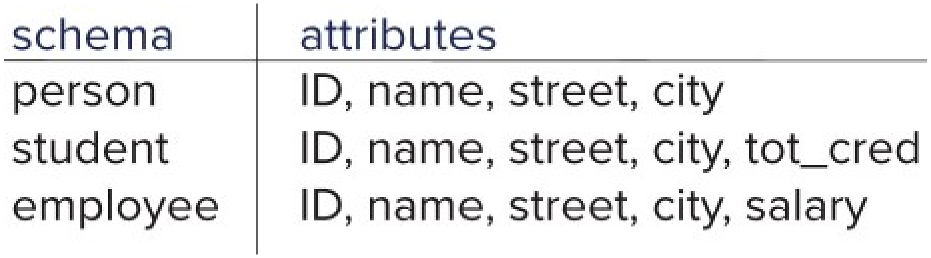


schema	attributes
person	ID, name, street, city
student	ID, tot_cred
employee	ID, salary

  - **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 as Schemas (Cont.)

- **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

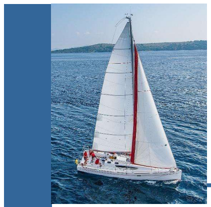
  - **Drawback:** ***name, street*** **and** ***city*** **may be stored redundantly for people who are both students and employees**

Generalization

- **A bottom-up design process – combine a number of entity sets that share the same features into a higher-level entity set.**
- **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.**

Completeness constraint

- **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
  - **partial**: an entity need not belong to one of the lower-level entity sets

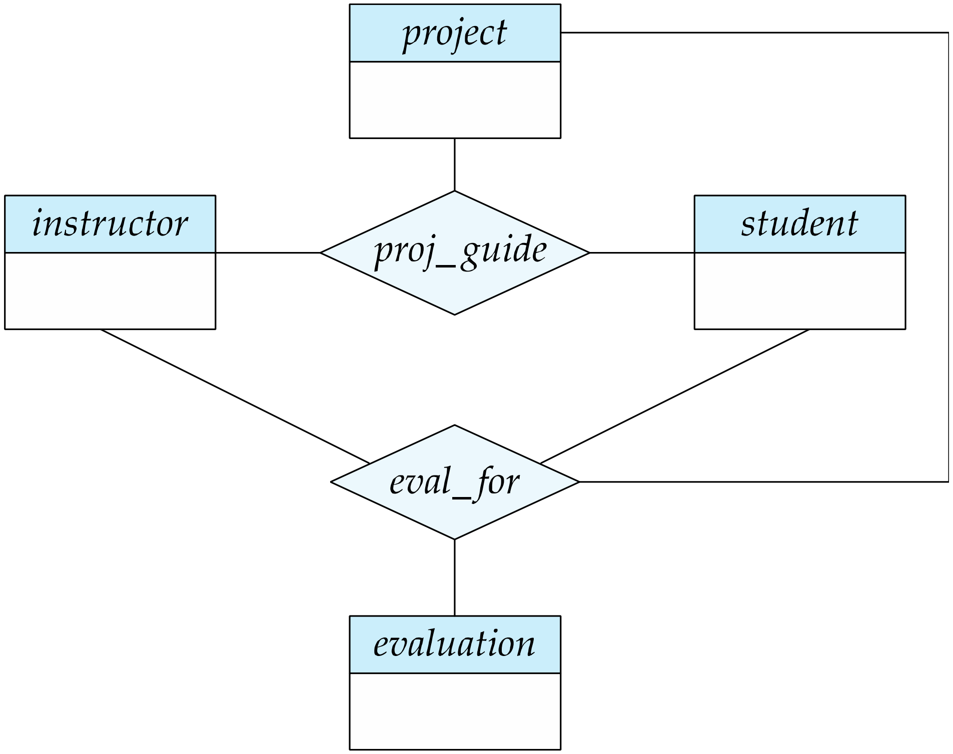


Completeness constraint (Cont.)

- 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

Aggregation

- Consider the ternary relationship *proj_guide*, which we saw earlier
- Suppose we want to record evaluations of a student by a guide on a project

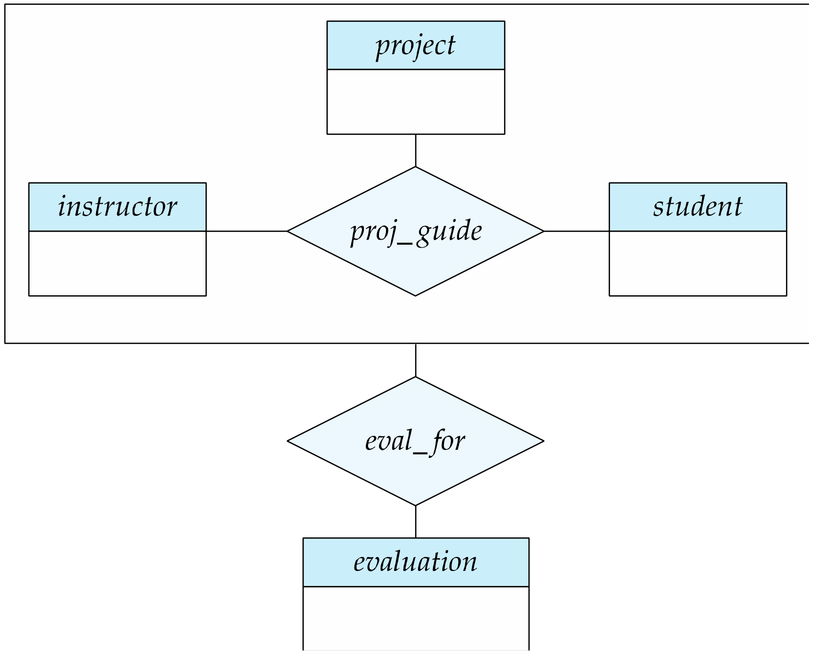


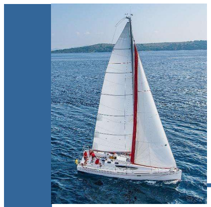
Aggregation (Cont.)

- Relationship sets *eval_for* and *proj_guide* represent overlapping information
  - Every *eval_for* relationship corresponds to a *proj_guide* relationship
  - However, some *proj_guide* relationships may not correspond to any *eval_for* relationships
    - So we can't discard the *proj_guide* relationship
- Eliminate this redundancy via *aggregation*
  - Treat relationship as an abstract entity
  - Allows relationships between relationships
  - Abstraction of relationship into new entity

Aggregation (Cont.)

- Eliminate this redundancy via *aggregation* without introducing redundancy, the following diagram represents:
  - A student is guided by a particular instructor on a particular project
  - A student, instructor, project combination may have an associated evaluation

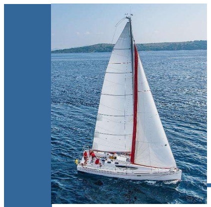




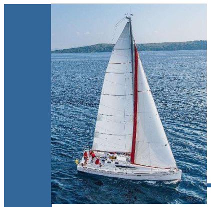
Reduction to Relational Schemas

- To represent aggregation, create a schema containing
  - Primary key of the aggregated relationship,
  - The primary key of the associated entity set
  - Any descriptive attributes
- In our example:
  - The schema *eval_for* is:

    eval_for (*s_ID, project_id, i_ID, evaluation_id*)
  - The schema *proj_guide* is redundant.

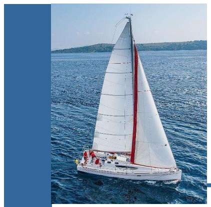


Design Issues



مشخص کردن موجودیت ها

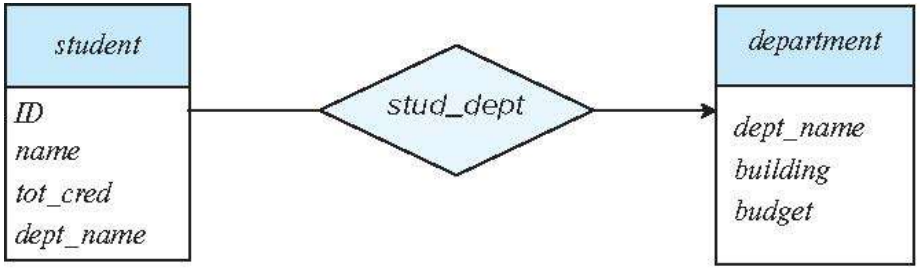
- ”نیاز به یک پایگاه داده داریم که اطلاعات دانشجو، استاد و درس در آن ذخیره شود. هر دانشجو می تواند چند در را اخذ کند. هر استاد می تواند چند درس در طول یک ترم ارائه نماید.“



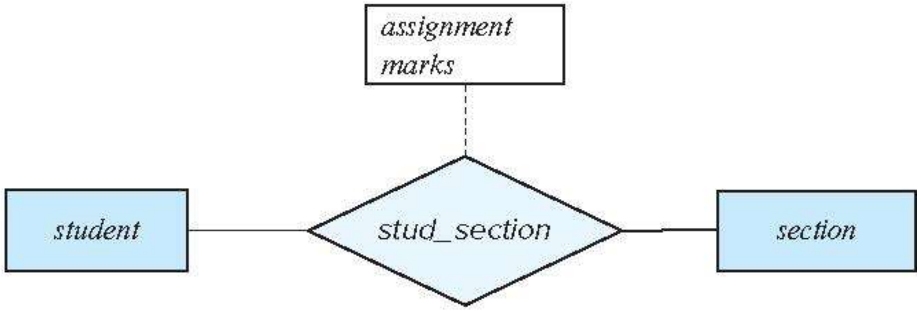
- موجودیت در مقابل صفت
  - صفات چند مقداری (موجودیت ضعیف)
- موجودیت در مقابل رابطه
  - رابطه های صفت دار
- رابطه دوگانی در مقابل رابطه سه گانی (؟؟)
  - همیشه نمی توان به جای یک رابطه سه گانی از دو یا سه رابطه دوگانی استفاده کرد.
- تجمیع در مقابل رابطه سه گانی
- موجودیت به جای رابطه سه گانی

Common Mistakes in E-R Diagrams

- **Example of erroneous E-R diagrams**



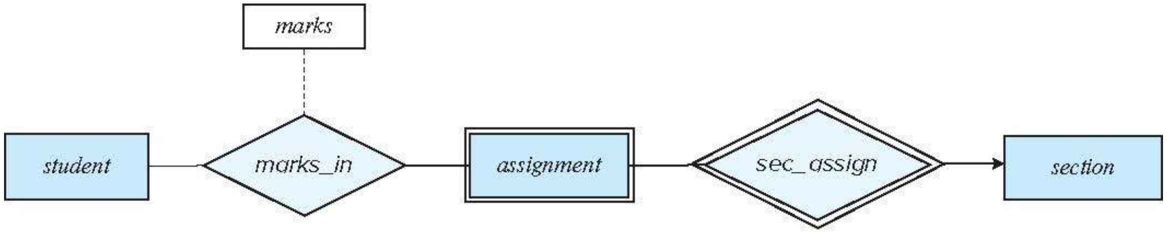
(a) Incorrect use of attribute



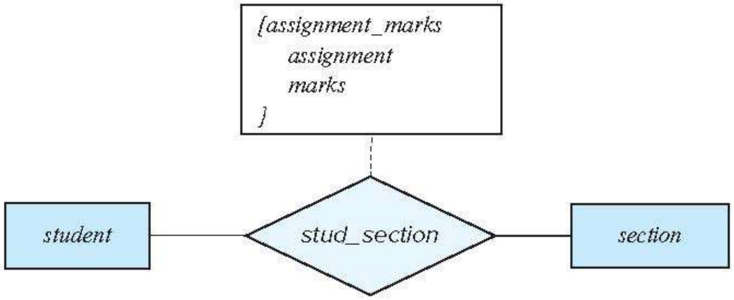
(b) Erroneous use of relationship attributes

Common Mistakes in E-R Diagrams (Cont.)

- **Correct versions of the E-R diagram of previous slide**



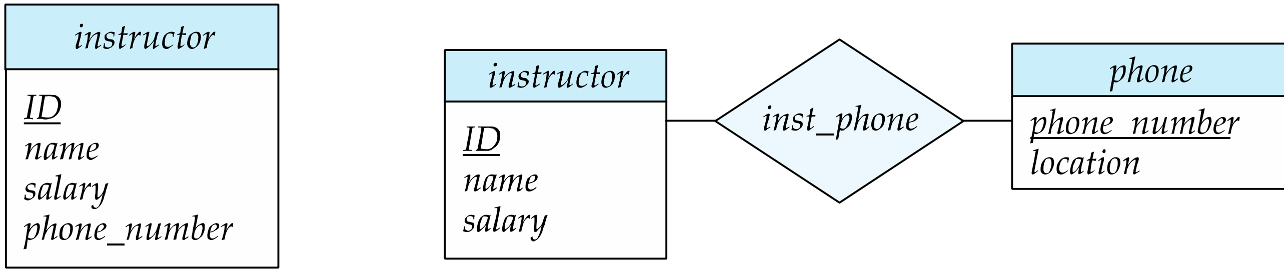
(c) Correct alternative to erroneous E-R diagram (b)



(d) Correct alternative to erroneous E-R diagram (b)

Entities vs. Attributes

- **Use of entity sets vs. attributes**

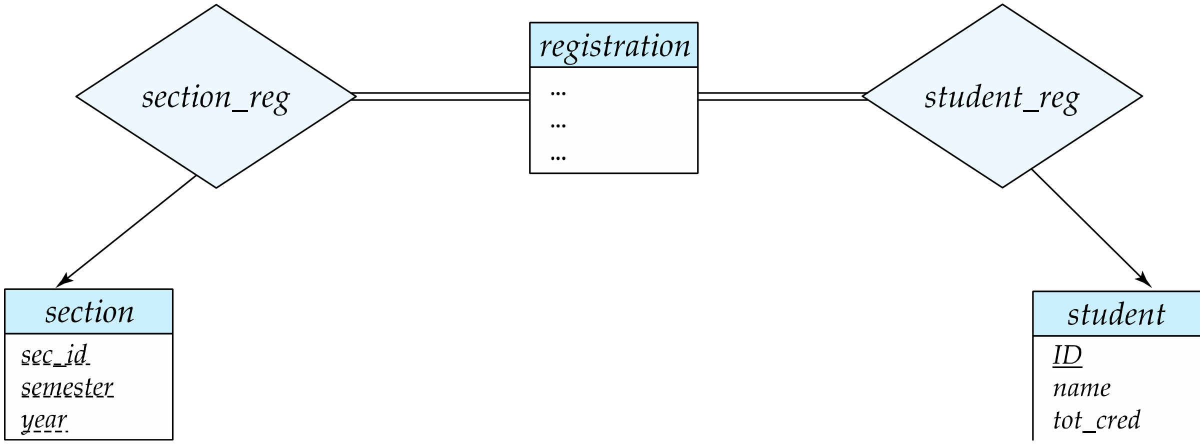


- **Use of phone as an entity allows extra information about phone numbers (plus multiple phone numbers)**

Entities vs. Relationship sets

- **Use of entity sets vs. relationship sets**

  Possible guideline is to designate a relationship set to describe an action that occurs between entities



- **Placement of relationship attributes**

  For example, attribute date as attribute of advisor or as attribute of student

Binary Vs. Non-Binary Relationships

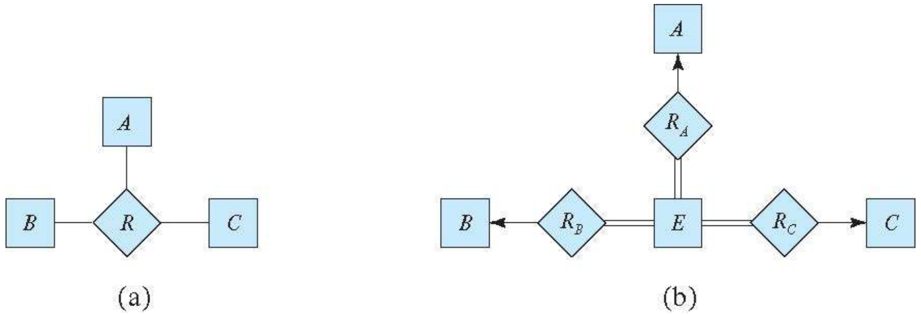
- **Although it is possible to replace any non-binary (*n*-ary, for *n* > 2) relationship set by a number of distinct binary relationship sets, a *n*-ary relationship set shows more clearly that several entities participate in a single relationship.**
- **Some relationships that appear to be non-binary may be better represented using binary relationships**
  - **For example, a ternary relationship *parents*, relating a child to his/her father and mother, is best replaced by two binary relationships, *father* and *mother***
    - **Using two binary relationships allows partial information (e.g., only mother being known)**
  - **But there are some relationships that are naturally non-binary**
    - **Example: *proj_guide***

Converting Non-Binary Relationships to Binary Form

- In general, any non-binary relationship can be represented using binary relationships by creating an artificial entity set.
  - Replace R between entity sets A, B and C by an entity set E, and three relationship sets:

    1. R_A, relating E and A 2. R_B, relating E and B
    3. R_C, relating E and C
  - Create an identifying attribute for E *and* add any attributes of R to E
  - For each relationship (a_i , b_i , c_i) in R, create

    1. a new entity e_i in the entity set E 2. add (e_i , a_i) to R_A
    3. add (e_i , b_i) to R_B 4. add (e_i , c_i) to R_C

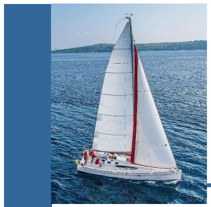


(a)

(b)

Converting Non-Binary Relationships (Cont.)

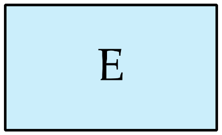
- Also need to translate constraints
  - Translating all constraints may not be possible
  - There may be instances in the translated schema that cannot correspond to any instance of *R*
    - Exercise: *add constraints to the relationships* R_A, R_B *and* R_C to ensure that a newly created entity corresponds to exactly one entity in each of entity sets *A*, *B* and *C*
  - We can avoid creating an identifying attribute by making E a weak entity set (described shortly) identified by the three relationship sets



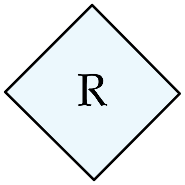
E-R Design Decisions

- The use of an attribute or entity set to represent an object.
- Whether a real-world concept is best expressed by an entity set or a relationship set.
- The use of a ternary relationship versus a pair of binary relationships.
- The use of a strong or weak entity set.
- The use of specialization/generalization – contributes to modularity in the design.
- The use of aggregation – can treat the aggregate entity set as a single unit without concern for the details of its internal structure.

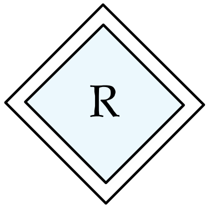
Summary of Symbols Used in E-R Notation



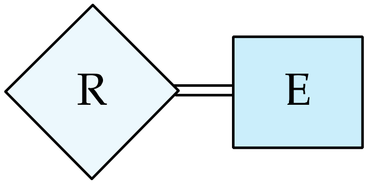
entity set



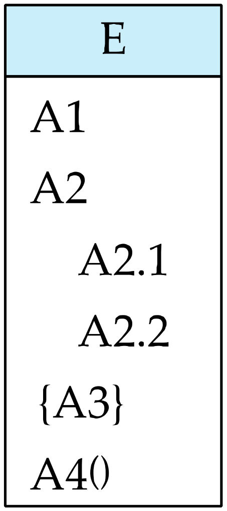
relationship set



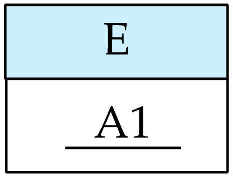
identifying
relationship set
for weak entity set



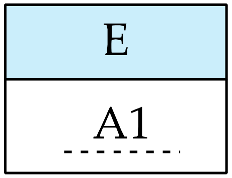
total participation
of entity set in
relationship



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

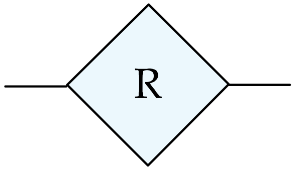


primary key

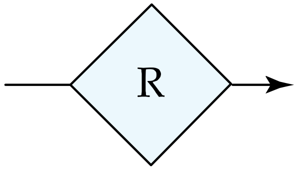


discriminating
attribute of
weak entity set

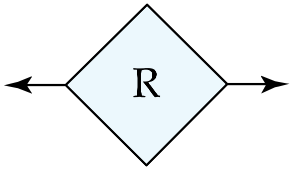
Symbols Used in E-R Notation (Cont.)



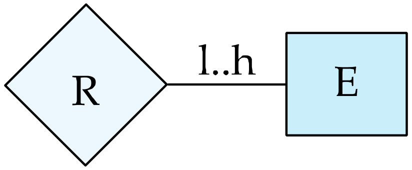
many-to-many relationship



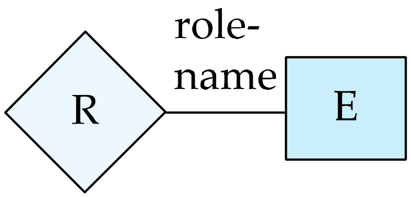
many-to-one relationship



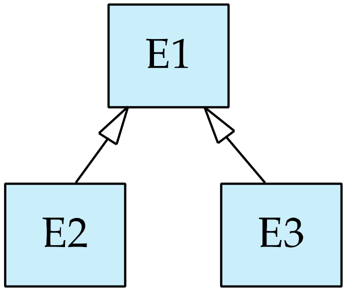
one-to-one relationship



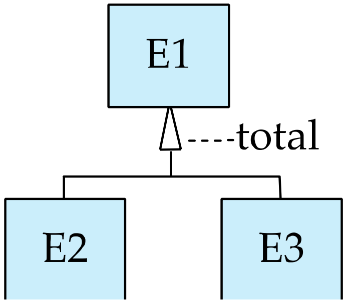
cardinality limits



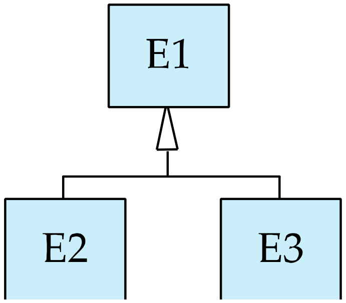
role indicator



ISA: generalization or specialization



total (disjoint) generalization

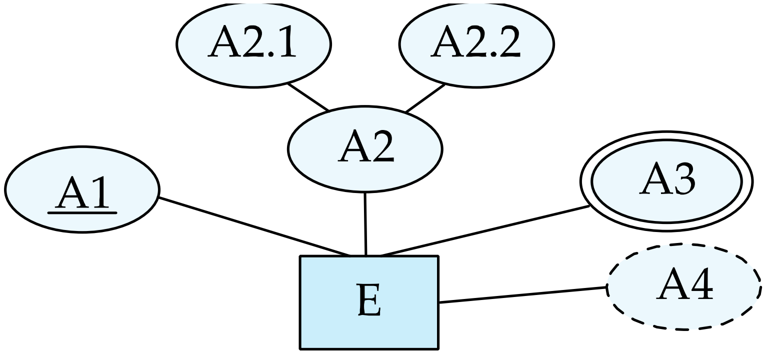


disjoint generalization

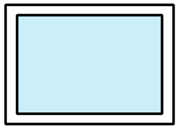
Alternative ER Notations

- Chen, IDE1FX, …

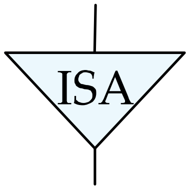
entity set E with
simple attribute A1,
composite attribute A2,
multivalued attribute A3,
derived attribute A4,
and primary key A1



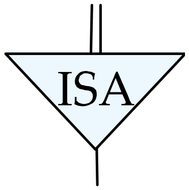
weak entity set



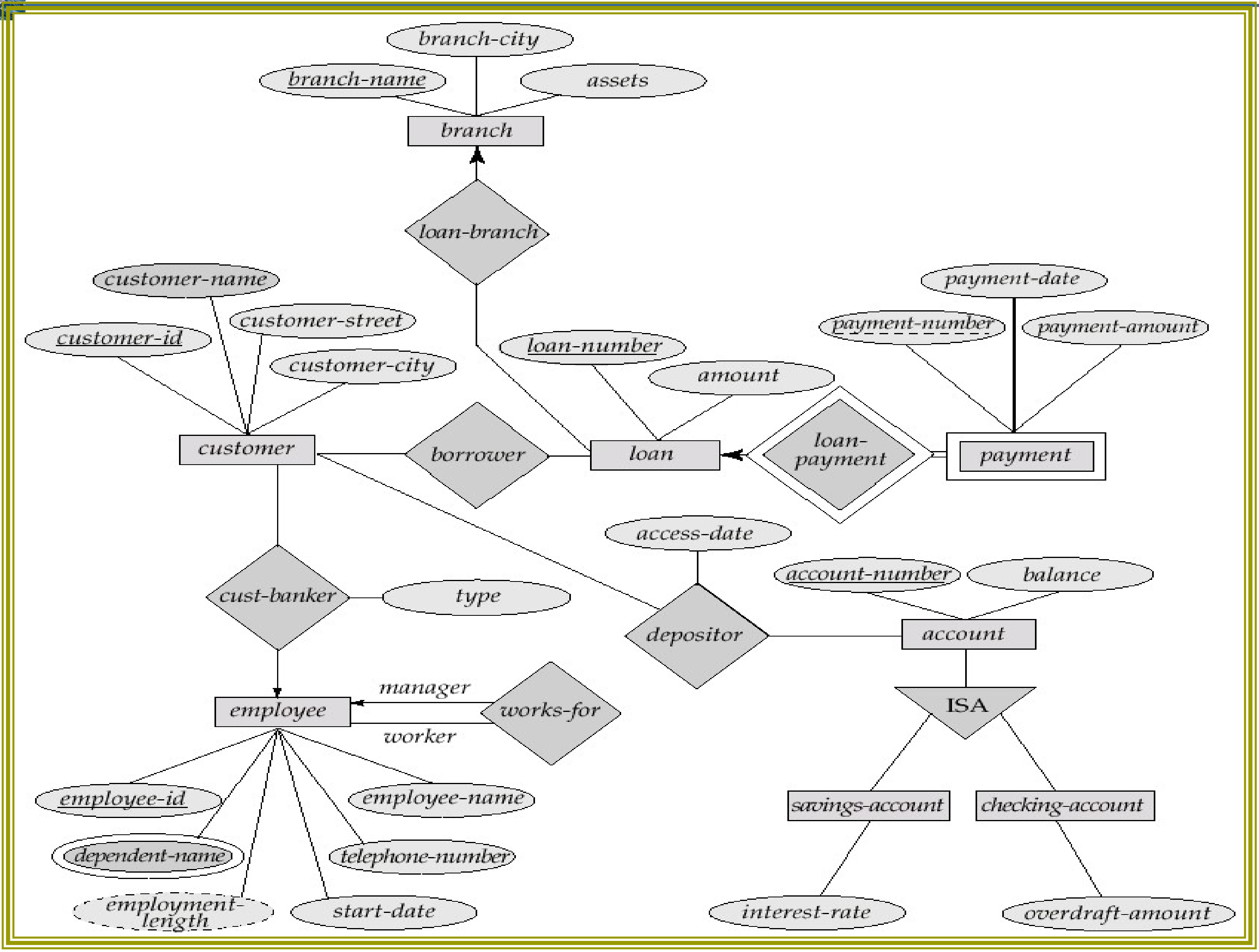
generalization



total generalization



نمونه ERD برای گوشه ای از سیستم بانکداری



Alternative ER Notations

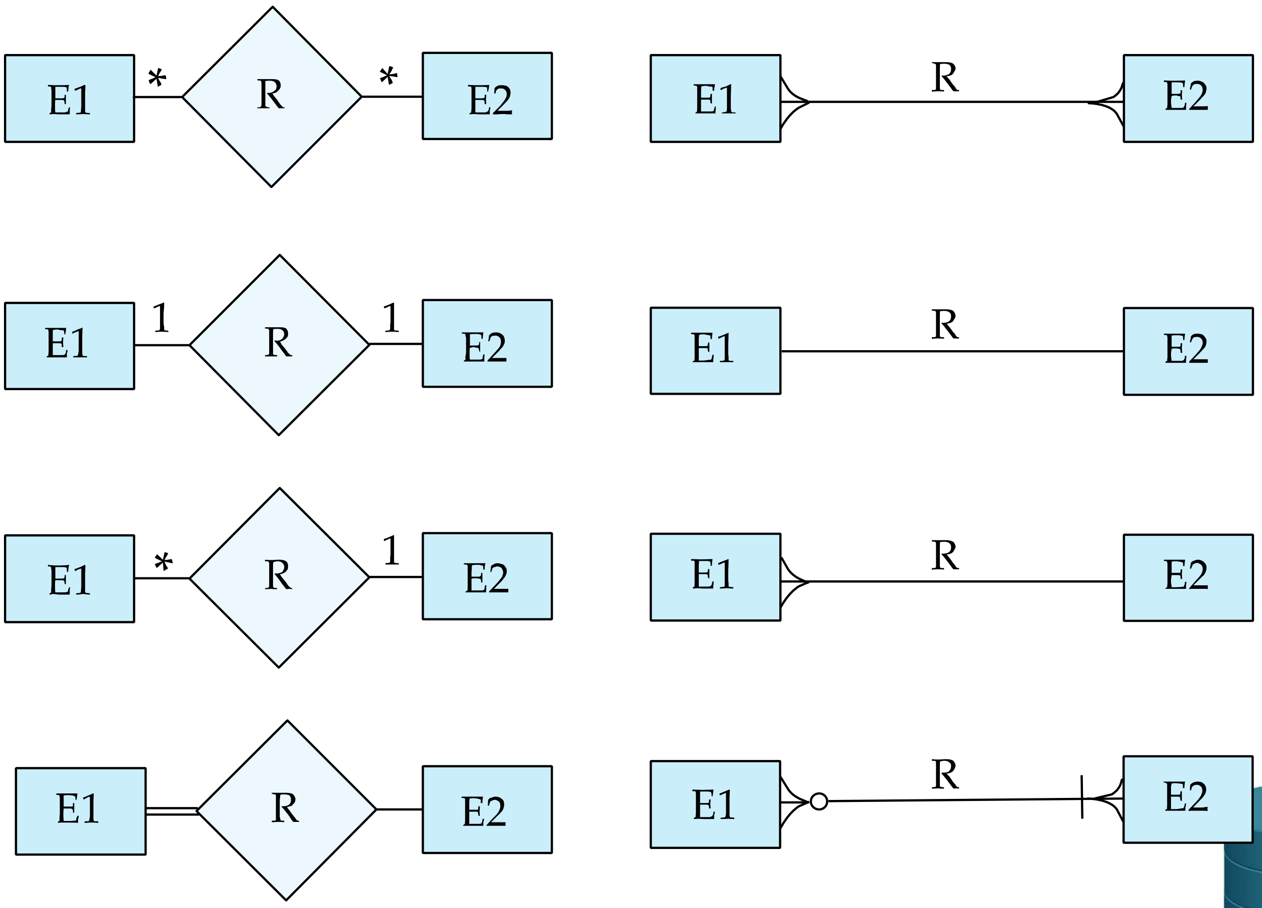
Chen **IDE1FX (Crows feet notation)**

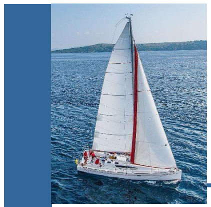
many-to-many relationship

one-to-one relationship

many-to-one relationship

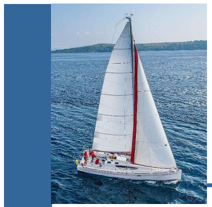
participation in R: total (E1) and partial (E2)





UML

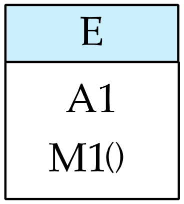
- **UML: Unified Modeling Language**
- **UML has many components to graphically model different aspects of an entire software system**
- **UML Class Diagrams correspond to E-R Diagram, but several differences.**



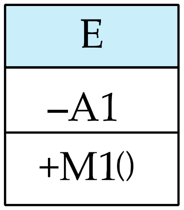
ER vs. UML Class Diagrams

ER Diagram Notation

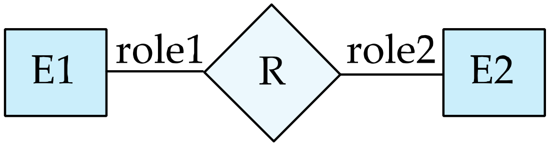
Equivalent in UML



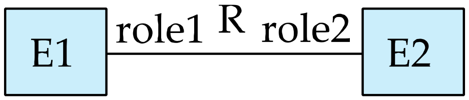
entity with attributes (simple, composite, multivalued, derived)

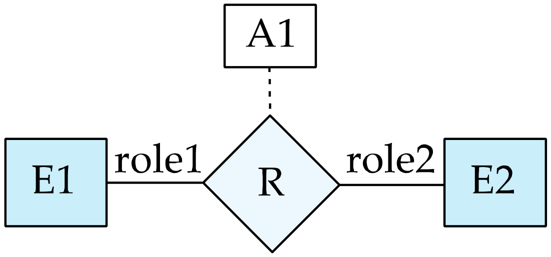


class with simple attributes and methods (attribute prefixes: + = public, – = private, # = protected)

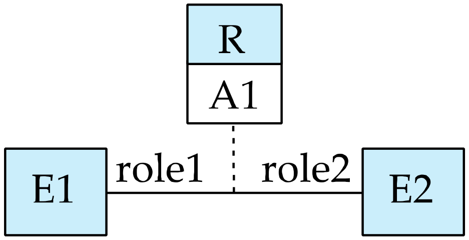


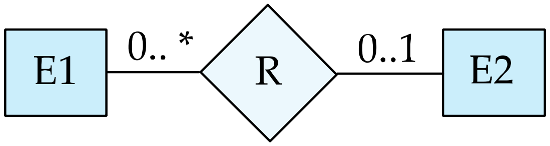
binary relationship



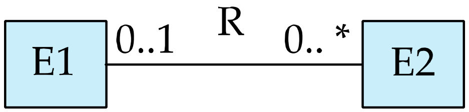


relationship attributes





cardinality constraints

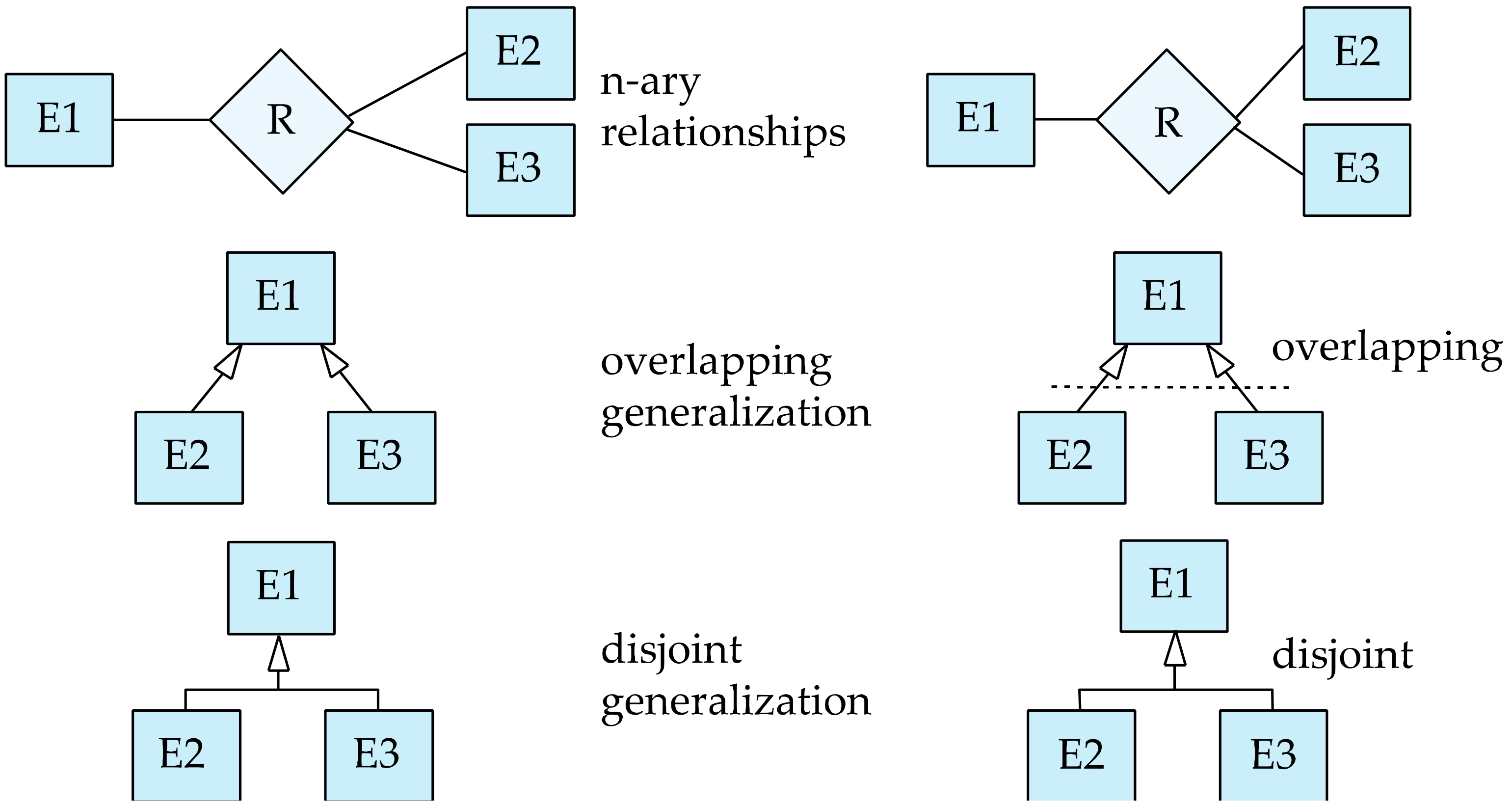


*Note reversal of position in cardinality constraint depiction

ER vs. UML Class Diagrams

ER Diagram Notation

Equivalent in UML



n-ary relationships

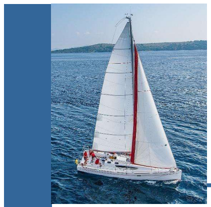
overlapping generalization

disjoint generalization

overlapping

disjoint

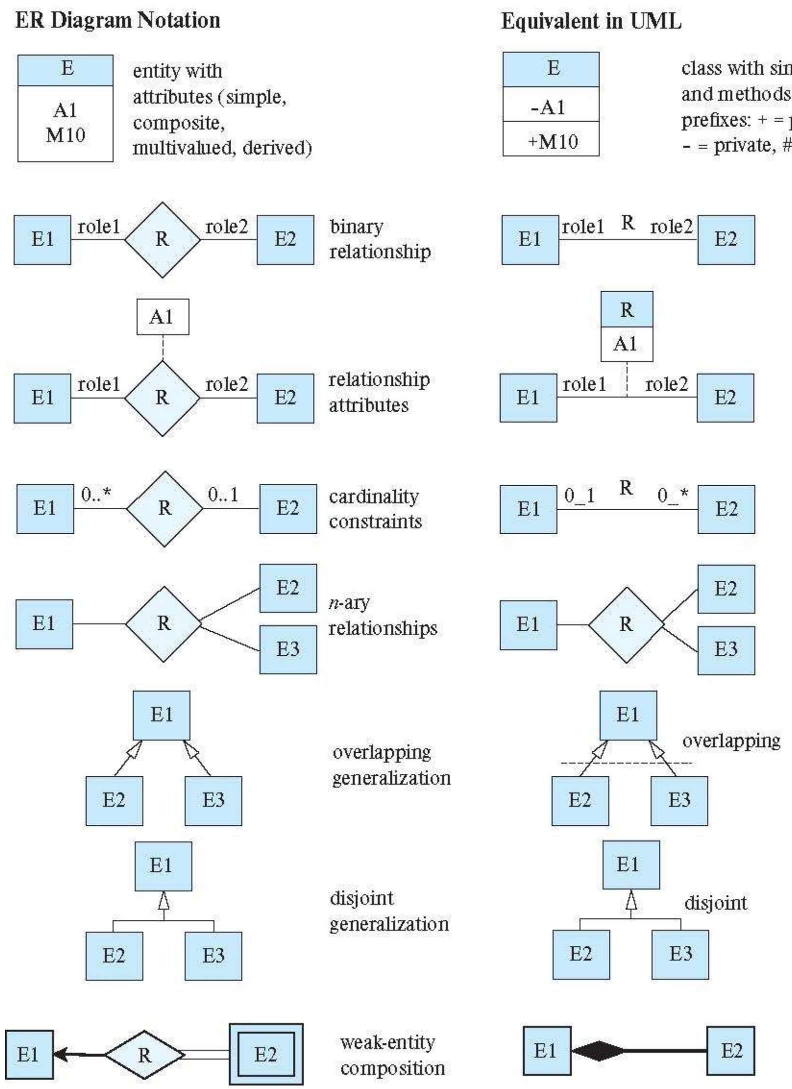
*Generalization can use merged or separate arrows independent of disjoint/overlapping



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.

ER vs. UML Class Diagrams



پایان فصل ششم