طراحی پایگاه داده

Chapter 6: Database Design Using the E-R Model

Database System Concepts, 7th Ed.

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

Outline

- Extended E-R Features
- Entity-Relationship Design Issues
- Alternative Notations for Modeling Data
- Other Aspects of Database Design

Design Phases

- Initial phase -- characterize fully the data needs of the prospective database users.
- Second phase -- choosing a data model
  - Applying the concepts of the chosen data model
  - Translating these requirements into a conceptual schema of the database.
  - A fully developed conceptual schema indicates the functional requirements of the enterprise.
    - Describe the kinds of operations (or transactions) that will be performed on the data.

-
ایجاد خود پایگاه داده:
ابتدا باید نیاز اطلاعاتی اون محیط رو متوجه بشیم ینی چیا میخوایم واسش ثبت کنیم
پایگاه داده همه جا مطرحه
ما باید خواسته ها رو کامل بفهمیم و موجودیت ها روکامل درک کنیم

Design Phases (Cont.)

- Final Phase -- Moving from an abstract data model to the implementation of the database
  - Logical Design – the designer maps the high-level conceptual schema onto the implementation data model of the database system that will be used. The implementation data model is typically the relational data model, and this step typically consists of mapping the conceptual schema defined using the entity-relationship model into a relation schema.
  - Physical Design – Deciding on the physical layout of the database

Design Approaches

- Entity Relationship Model (covered in this chapter)
  - Models an enterprise as a collection of *entities* and *relationships*
    - Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects
      - Described by a set of *attributes*
    - Relationship: an association among several entities
  - Represented diagrammatically by an *entity-relationship diagram:*
- Normalization Theory (Chapter 7)
  - Formalize what designs are bad, and test for them

-
منطقی که میخوایم استفاده بکنیم ER مدل هست--> موجودیت عنصر مهم توی ER است
ینی اول میایم موجودیت ها رو مشخص میکنیم و بعد توی لایه بعدی برای هر کدوم از این موجودیت ها باید بیایم فیلدهامون رو مشخص بکنیم
بین موجودیت ها باید بیایم رابطه برقرار بکنیم

Outline of the ER Model

ER model -- Database Modeling

- The ER data mode was developed to facilitate database design by allowing specification of an **enterprise schema** that represents the overall logical structure of a database.
- The ER data model employs three basic concepts:
  - entity sets,
  - relationship sets,
  - attributes.
- The ER model also has an associated diagrammatic representation, the **ER diagram**, which can express the overall logical structure of a database graphically.

Entity Sets

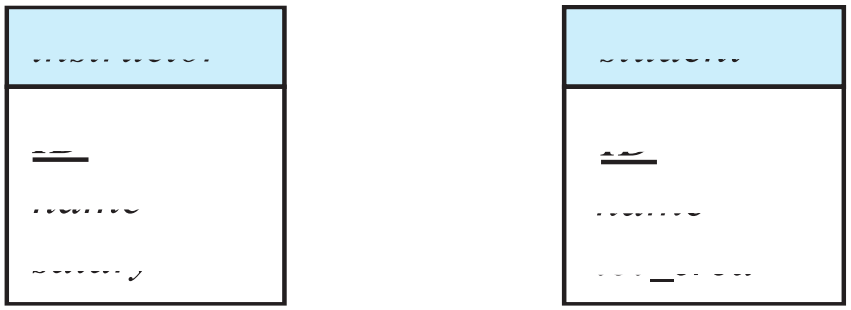
- An **entity** is an object that exists and is distinguishable from other objects.
  - Example: specific person, company, event, plant
- An **entity set** is a set of entities of the same type that share the same properties.
  - Example: set of all persons, companies, trees, holidays
- An entity is represented by a set of attributes; i.e., descriptive properties possessed by all members of an entity set.
  - Example:

    instructor = (*ID, name, salary*)
    course= (*course_id, title, credits*)
- A subset of the attributes form a **primary key** of the entity set; i.e., uniquely identifying each member of the set.

-
نمونه های یک ابجکت هستند <-- entity sets
هر entity یک سری صفت هایی داره
entity یا موجودیت: هر چیزی که ما بتونیم براش جدول ایجاد کنیم میشه موجودیت مثلا درس می تونه موجودیت باشه ولی شماره دانشجویی نه چون برای شماره دانشجویی نمی تونیم جدول بکشیم

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



-
برای هر موجودیت مستقل یک مستطیل داریم

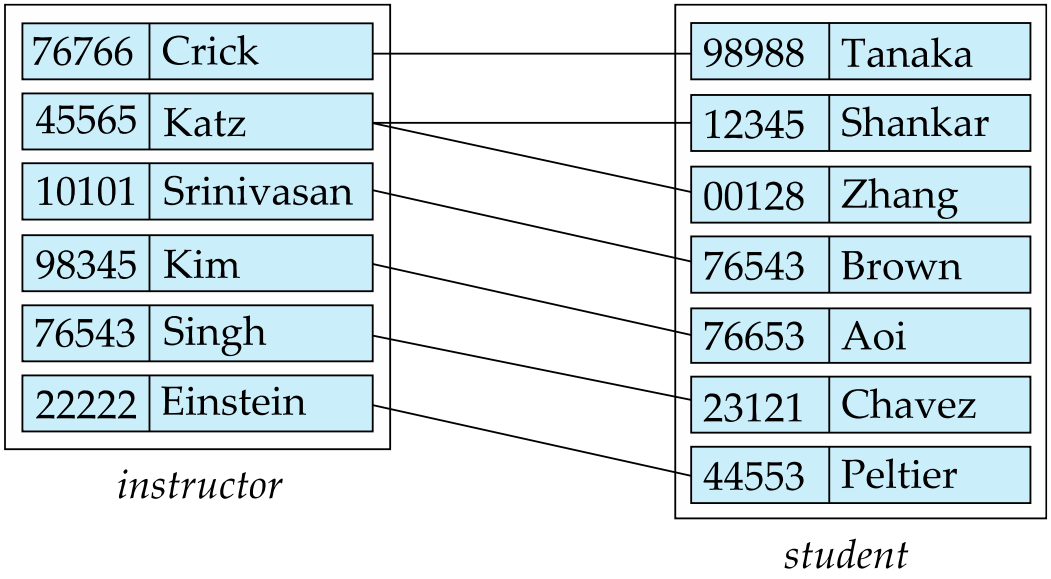
Relationship Sets

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

  Example:

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

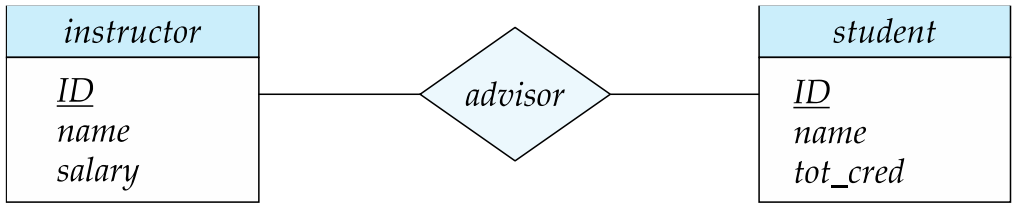
- A **relationship set** is a set of relationships of the same type.
- Example: we define the relationship set *advisor* to denote the associations between students and the instructors who act as their advisors. We draw a line between related entities.



-
رابطه:
بین موجودیت ها می تونیم رابطه داشته باشیم

Representing Relationship Sets via ER Diagrams

- Diamonds represent relationship sets.

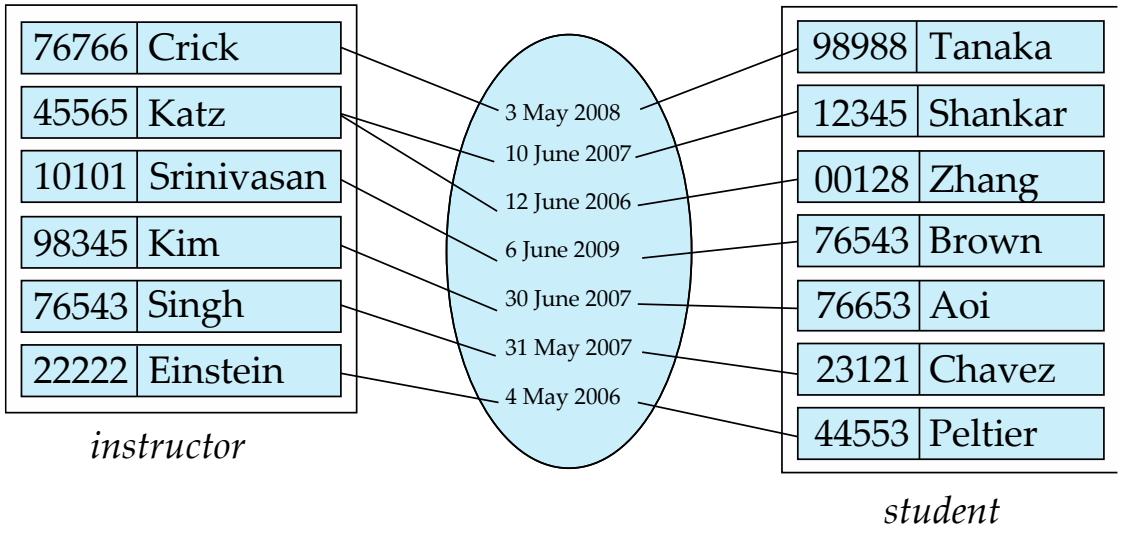


این رابطه بین استاد راهنما و دانشجو است برای پایان نامه

-
اول موجودیت ها رو باید بفهمیم
به محض اینکه فهمیدیم بین موجودیت ها رابطه است میایم یه لوزی بین موجودیت ها می کشیم
نکته: این رابطه خودش میتونه صفت داشته باشه ینی منظورمان همین لوزیه است

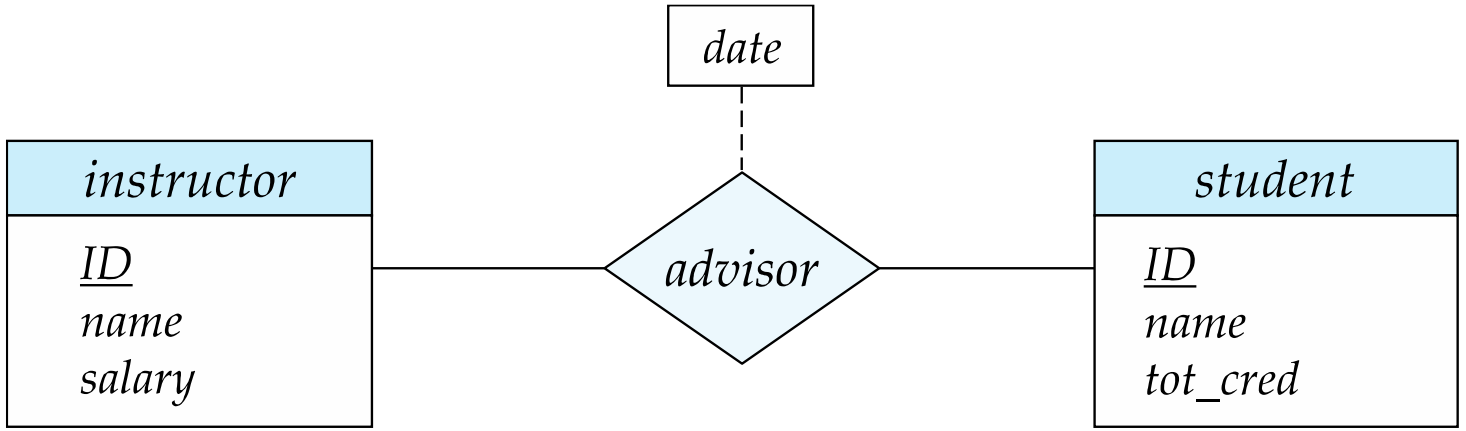
Relationship Sets (Cont.)

- An attribute can also be associated with a relationship set.
- 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



این تاریخ هارو داره ذخیره میکنه یه جایی ینی صفت های توی رابطه رو می تونیم ذخیره کنیم
نکته: یک رابطه ممکن است ویژگی های توصیفی داشته باشد

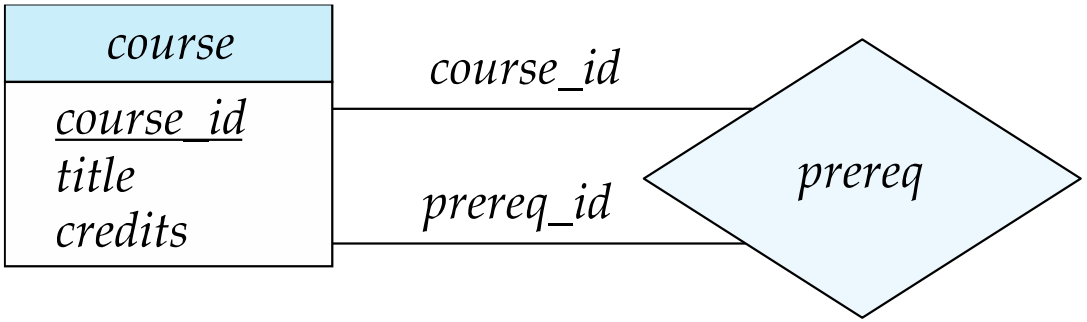
Relationship Sets with Attributes



نکته: بین موجودیت ها می تونیم رابطه یا رابطه ها داشته باشیم و خود رابطه ها هم می تونند ویژگی یا صفت داشته باشند

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**.



توی ER دیاگرام می تونیم رابطه بازگشتی هم داشته باشیم ینی یک موجودیتی می تونه با خودش هم رابطه داشته باشه مثلا اینجا یک درس می تونه پیش نیاز یک درس دیگه باشه
مثلا ریاضی یک پیش نیاز ریاضی دو است و از اونجایی که جفتشون هم درس هستند پس از یک موجودیت دارند نشات می گیرند

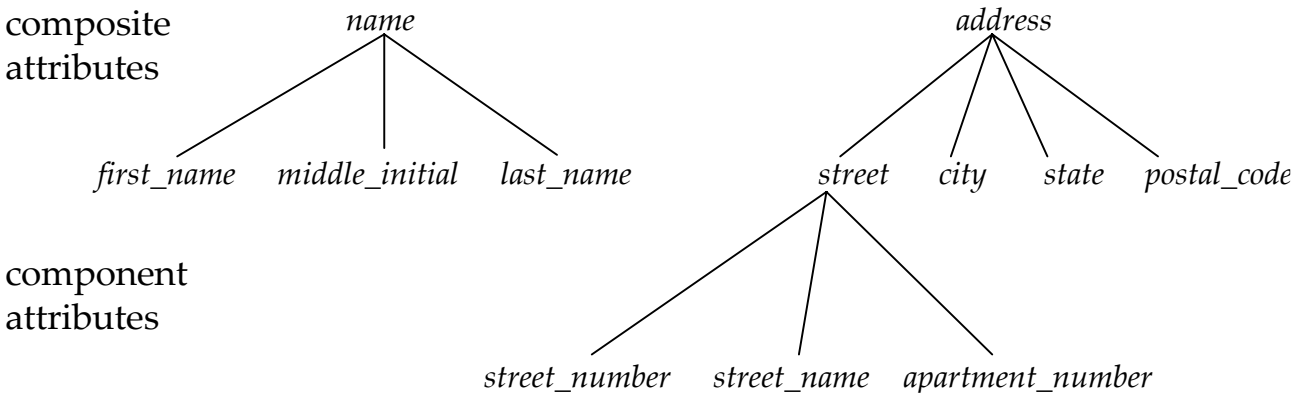
Complex Attributes

- Attribute types:
  - **Simple** and **composite** attributes.
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_numbers {}*
  - **Derived** attributes ()
    - Can be computed from other attributes
    - Example: age, given date_of_birth
- **Domain** – the set of permitted values for each attribute

-
صفت هایی یا ویژگی هایی که میتونیم برای یک موجودیت داشته باشیم:
صفت یا ویژگی ساده یا صفت ترکیبی
صفت چند مقداره یا تک مقداره مثلا شماره تلفن --> یک دانشجو می تونه مثلا چندتا شماره تلفن داشته باشه که این میشود صفت چند مقداره
که این صفتو با {} نشون میدیم
صفت های محاسباتی با نماد () مثل سن فرد یا تاریخ تولد

Composite Attributes

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



اسم رو میتونیم بشکونیم به چندتا چیز پس اسم میشه صفت ترکیبی اما اسم خودش به تنهایی میشه صفت ساده
ادرس هم همین طور
نکته: پس اونایی که می تونن به زیر ویژگی ها تقسیم شوند رو صفت مرکب می گیم

Representing Complex Attributes in ER Diagram

ID یک صفت ساده است و همین طور کلید اصلی است

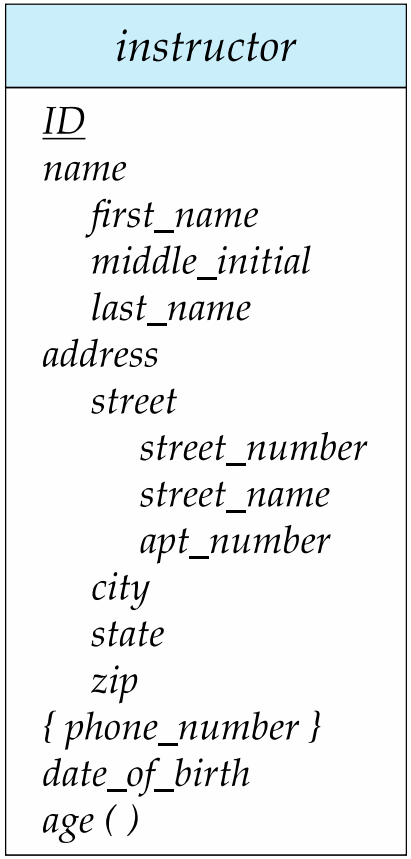
نام صفت مرکب

ادرس صفت مرکب

شماره تلفن چندتا مقدار دارد ینی چند مقداره است

تاریخ تولد صفت ساده است

سن یک صفت محاسباتی است



Mapping Cardinality Constraints

- Express the number of entities to which another entity can be associated via a relationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many

-
بین موجودیت ها رابطه داریم

نوع رابطه بین جدول ها: کاردینالیتی ها:

• یک به یک: یک موجودیت در A حداکثر با یک موجودیت در B ارتباط دارد و یک موجودیت در B با حداکثر یک موجودیت در A مرتبط است

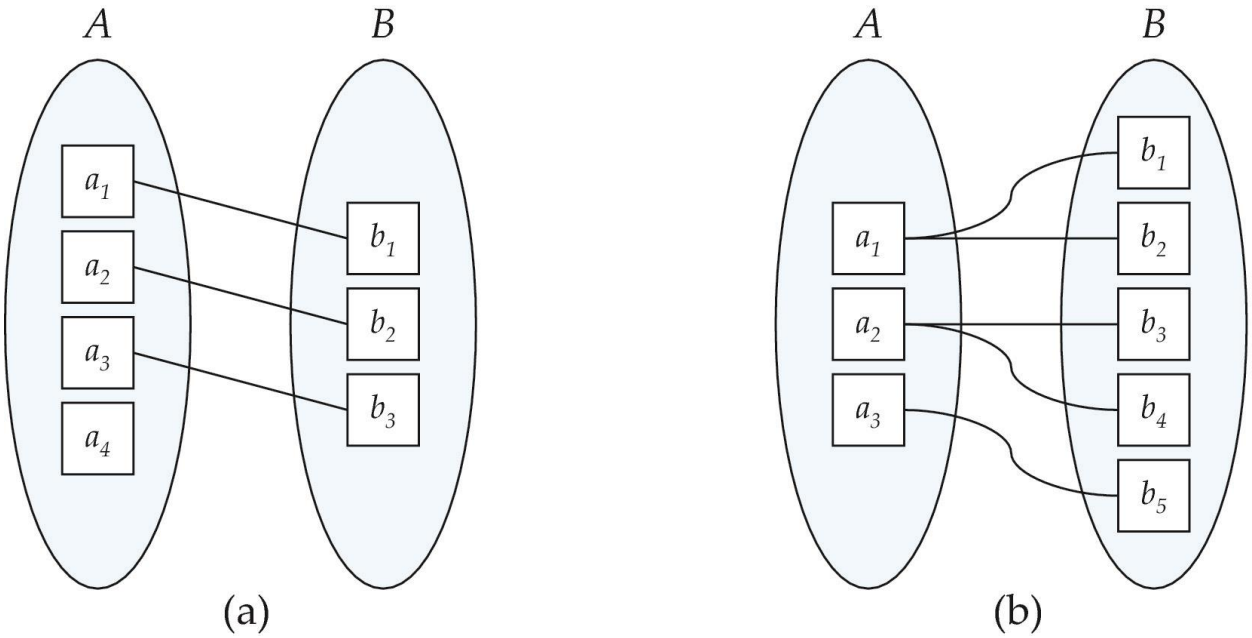
• یک به چند: یک موجودیت در A با هر تعداد (صفر یا چند) موجودیت در B رابطه دارد، اما یک موجودیت در B می تواند با حداکثر یک موجودیت در A مرتبط باشد

• چند به یک: یک موجودیت در A با حداکثر یک موجودیت در B مرتبط است، اما یک موجودیت در B می تواند با هر تعداد (صفر یا چند) موجودیت در A مرتبط باشد

• چند به چند: یک موجودیت در A با هر تعداد (صفر یا چند) موجودیت در B ارتباط دارد و یک موجودیت در B با هر تعداد (صفر یا چند) موجودیت در A مرتبط است

Mapping Cardinalities

entity set A



One to one

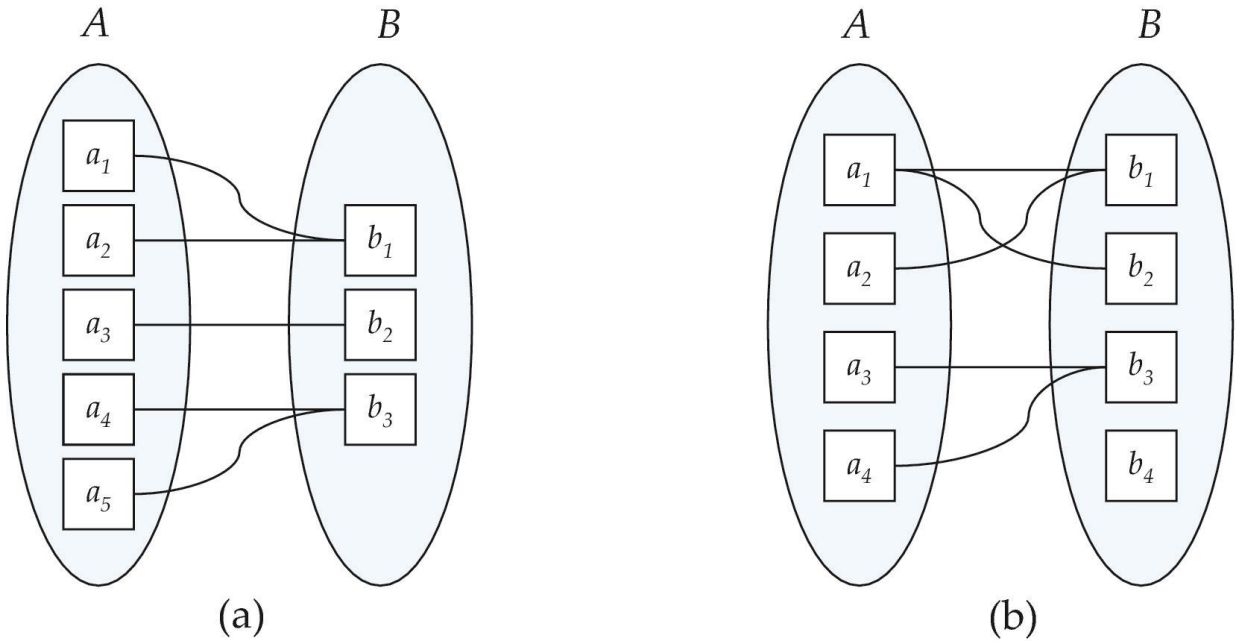
One to many

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

a--> مثالش: دانشکده با رئیس دانشکده ینی یه طرف یک موجودیت داریم از جنس هیئت علمی (A) و اون طرف هم یک موجودیت داریم از جنس دانشکده (B)

b--> مثالش: استاد راهنما و دانشجوها ینی یک استاد می تونه استاد راهنمای چندتا دانشجو باشه ولی برعکسش برقرار نیست

Mapping Cardinalities



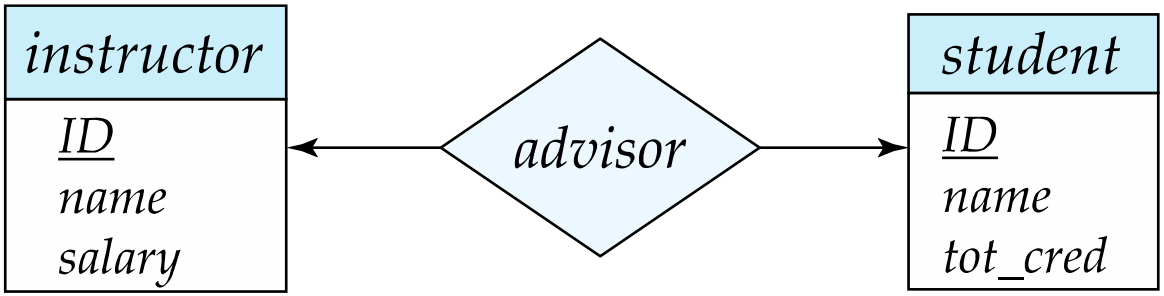
Many to one

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

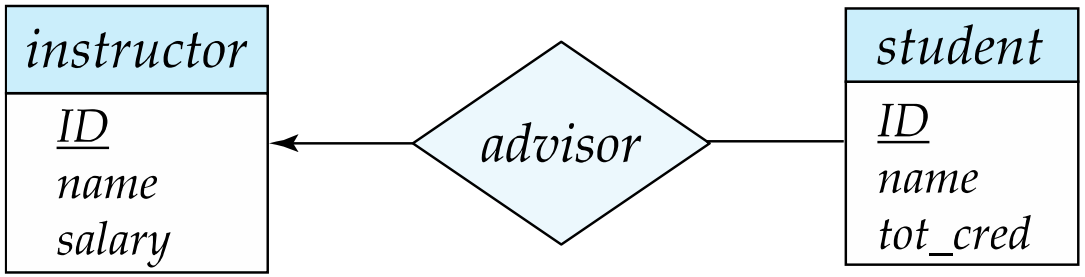
- We express cardinality constraints by drawing either a directed line (→), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set.
- One-to-one relationship between an *instructor* and a *student* :
  - A student is associated with at most one *instructor* via the relationship *advisor*
  - A *student* is associated with at most one *department* via *stud_dept*



-
محدویت هایی که میخوایم نشون بدیم توی دیاگرام:
با پیکان فقط یه بار میتونه ارتباط برقرار کنه ینی در واقع با پیکان محدودیت ها رو نشون میدیم
مثال اینجا:
مثلا یه دانشجو می تونه حداکثر یه استاد واسه راهنمایی داشته باشه و استاد هم می تونه راهنمای حداکثر یه دانشجو باشه

One-to-Many Relationship

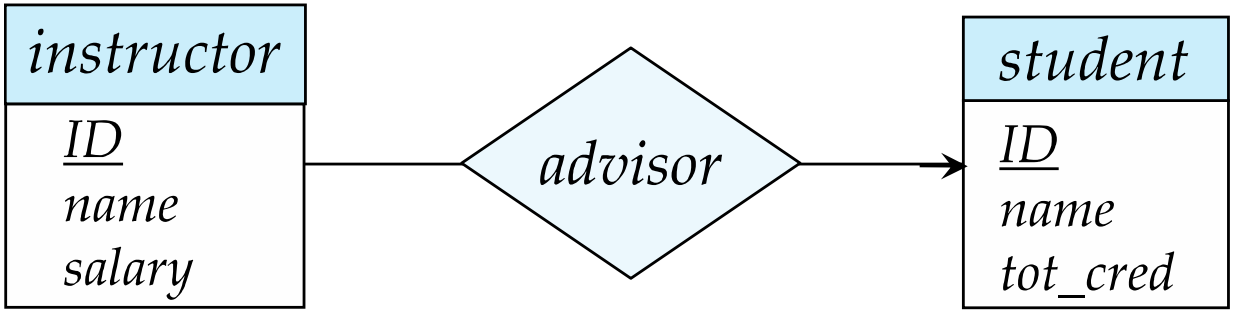
- one-to-many relationship between an *instructor* and a *student*
  - an instructor is associated with several (including 0) students via *advisor*
  - a student is associated with at most one instructor via advisor,



-
مثال:
یک استاد میتونه چندتا دانشجو رو راهنمایی بکنه (ینی این میتونه صفرتا هم باشه) ولی یک دانشجو حداکثر یک استاد میتونه راهنماش باشه

Many-to-One Relationships

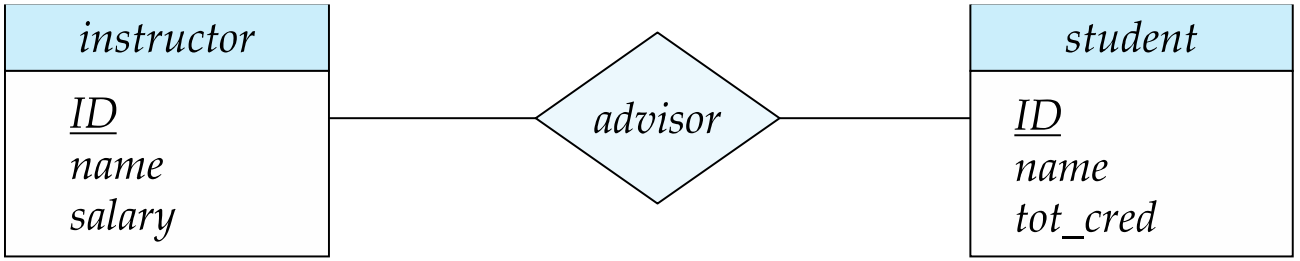
- In a many-to-one relationship between an *instructor* and a *student,*
  - an instructor is associated with at most one student via *advisor*,
  - and a student is associated with several (including 0) instructors via *advisor*



مثال:
یک دانشجو میتونه توسط چندتا استاد راهنمایی بشه ولی یه استاد حداکثر یک دانشجو رو میتونه راهنمایی بکنه

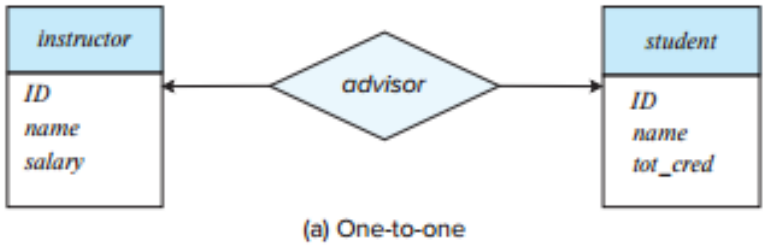
Many-to-Many Relationship

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

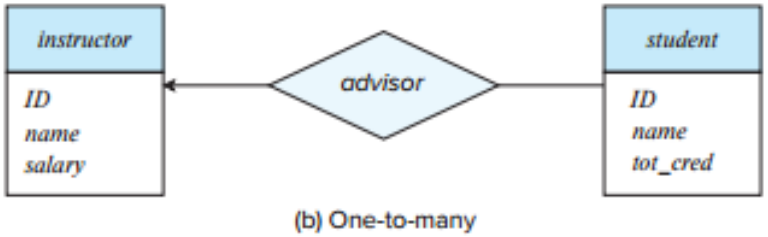


مثال:
اینجا رابطه many to many داریم
یک استاد میتونه چند دانشجو رو راهنمایی بکنه و یک دانشجو هم میتونه چندتا استاد داشته باشه

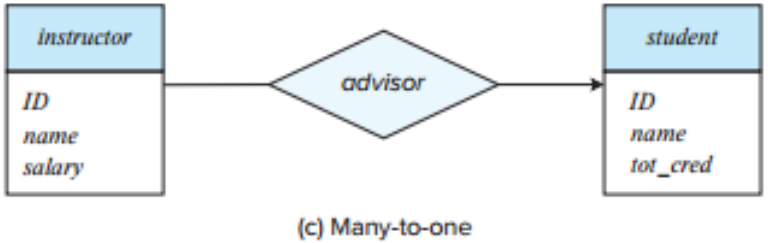
mapping cardinality



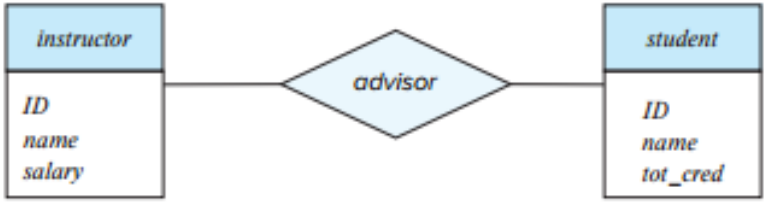
(a) One-to-one



(b) One-to-many



(c) Many-to-one

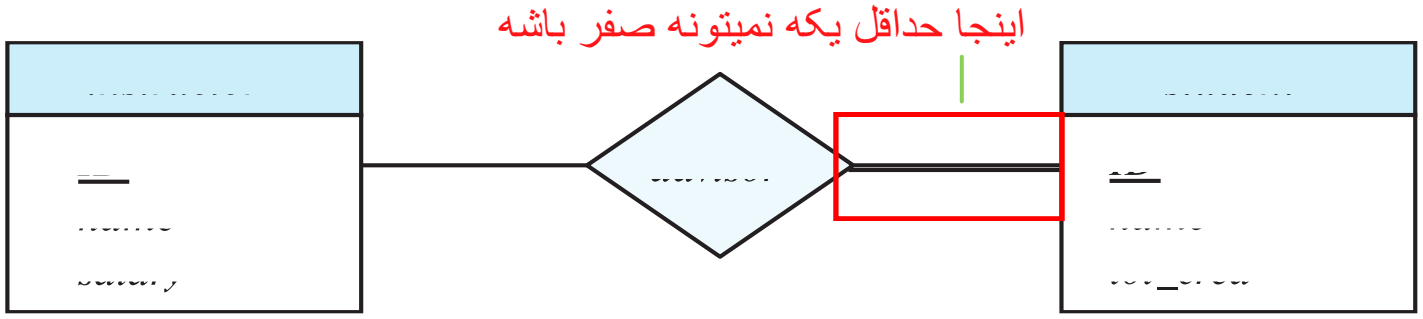


(d) Many-to-many

خلاصه قبلی ها -->

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

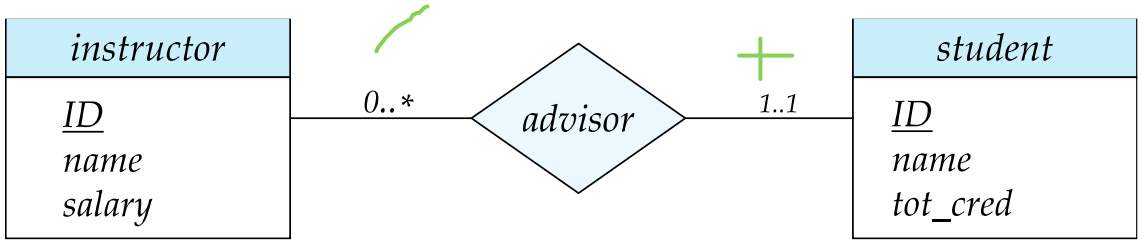
  - every *student* must have an associated instructor

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

-
بحث ما اینجا بازم محدودیت گذاشتن روی رابطه بین موجودیت هاست
اگر موجودیتی که داریم در موردش صبحت میکنیم باید حداقل توی یک ارتباط با موجودیت سمت مقابلش ارتباط برقرار بکنه میگیم total اون رابطه
و سمت عادی هم به صورت partial نشون میدیم

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.
- Example



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

-
ما میتونیم تعداد کاردینالیتی ارتباط بین موجودیت هارو با عدد هم نشان بدیم
دانشجو فقط میتواند با یک استاد مقاله رو برداشته باشه +
اینو استاد می تواند راهنمایی دانشجویی نباشه و در عین حال هم راهنمایی چندتا دانشجو باشه /

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

میخوایم برای موجودیت ها کلید بذاریم
کلید اصلی میشه مجموعه صفت هایی که اولا باید خاصیت سوپر کی رو داشته باشن ینی تکرار پذیر نباشند توی رکوردهای مختلف و کمینه هم باشند

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 union of the primary keys of entity sets E1, E2, ..En describes an individual relationship in set *R*
  - If the relationship set *R* has attributes a1, a2, .., am associated with it, then *primary-key*(*E*1) ∪ *primary-key*(*E*2) ∪ ⋯ ∪ *primary-key*(*En*) ∪ {*a*1, *a*2, … , *am*} describes an individual relationship in set *R*.
  - Thus, in both of the preceding cases, the set of attributes *primary-key*(*E*1) ∪ *primary-key*(*E*2) ∪ ⋯ ∪ *primary-key*(*En*) forms a superkey for the relationship set.
- Example: relationship set “advisor”.
  - The superkey 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.

-
می خوایم مشخص بکنیم خود اون لوزی ها که میشدند رابطه های ما کلید اصلیشون چیه
به لوزی ها میگفتیم relationship
ترکیب کلیدهایی که توی موجودیت اول و دوم باشند می تونند عملا یک سوپرکی باشه برامون ینی اگر کلید اصلی رابطه ها رو برداریم و مجموعشون عملا میشه یک سوپرکی برای relationship ما

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.

-
حالا اگر بخوایم کلید اصلی رو مشخص بکنیم برای این relationship میایم:
رابطه چند به چند: حتما کلید اصلی جدول چپ و راست باید باشه
نکته: رابطه می تونه یک جدول باشه اگر از نوع چند به چند باشه
رابطه یک به چند: کلید اصلی رو می تونیم از حداقلی بودن سوپر کلید سمت جدول چند به دست بیاریم
رابطه چند به یک: کلید اصلیمون میشه کلید اصلی سمت many مثل بالاییه دقیقا
رابطه یک به یک: فرقی نمیکنه یکی از موجودیت ها رو اینجا در نظر میگیریم

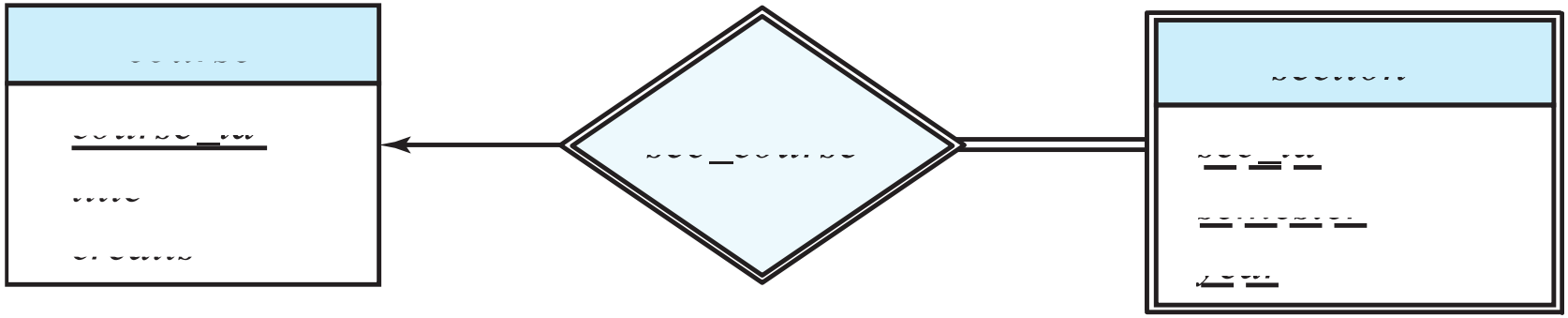
Weak Entity Sets

- 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.
- 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 relationship associating the weak entity set with the identifying entity set is called the **identifying relationship**.

-
موجودیت ضعیف: موجودیتی که به تنهایی عملا معنی نداره ینی چیزی ازش نمی تونیم بفهمیم ینی نمیتونیم بفهمیم درباره چی داره حرف می زنه مثلا دانشجو یه چیز واضحه و ما می فهمیم ولی مثلا والدین دانشجو اگر موجودیت ما باشه کسی اینو نمی شناسه مثلا توی دانشگاه ینی مثلا برای کسی مهم نیست و کسی اطلاعی ازش نداره
نکته: یه سری موجودیت هایی هستند که این موجودیت ها باید به یه چیزی توی اون بیزینس وصل باشند وگرنه به تنهایی معنیه ندارند پس برای این موجودیت باید موجودیت اصلیش باشه

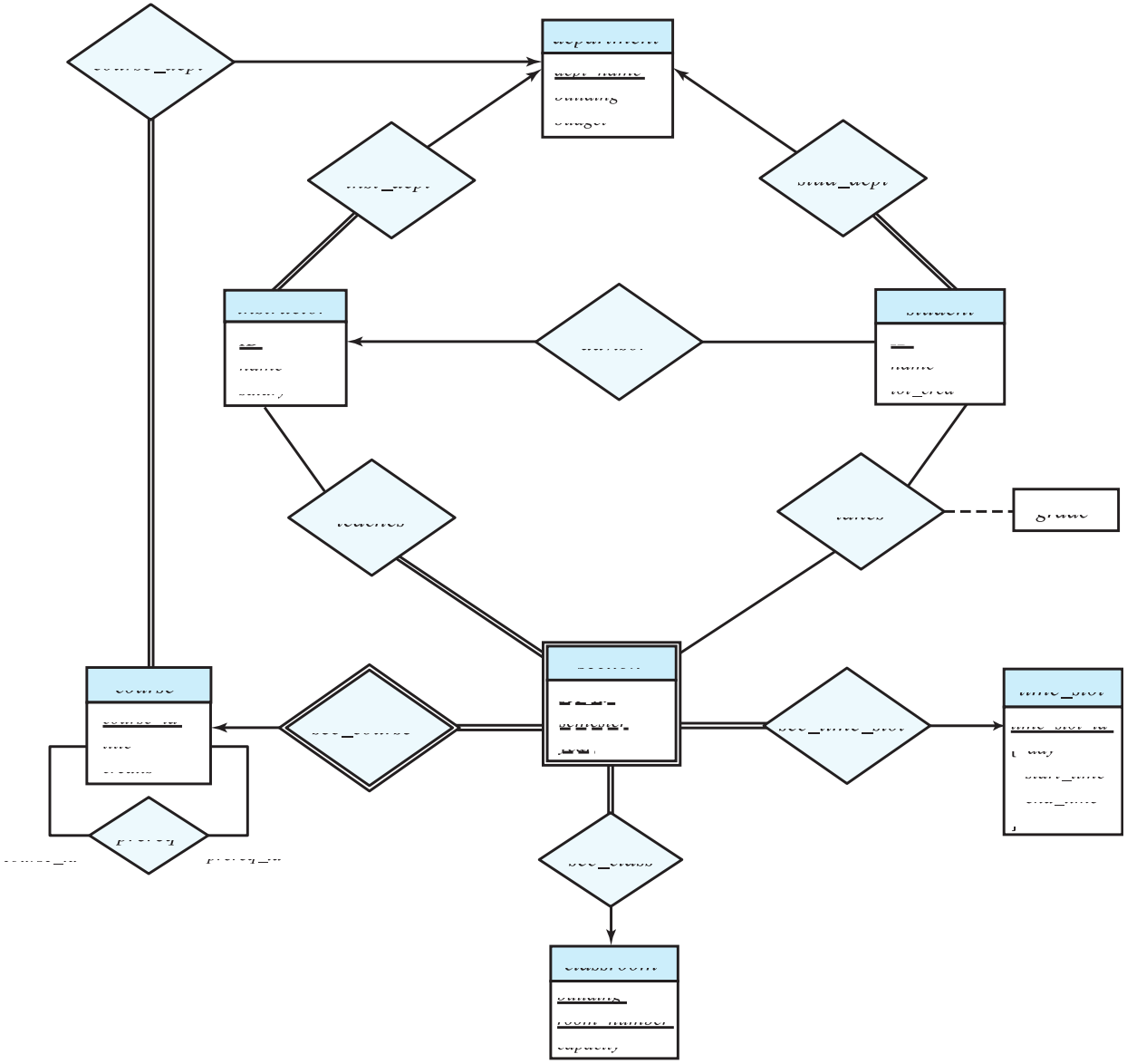
Expressing Weak Entity Sets

- 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*)



-
اون موجودیتی که ضعیف نیست رو میگن strong entity
موجودیت ضعیف رو توی ER دیاگرام به صورت یک مستطیل دوتایی نشون میدن

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 relation schemas.
- For each entity set and relationship set there is a unique relation schema that is assigned the name of the corresponding entity set or relationship set.
- Each relation schema has a number of columns (generally corresponding to attributes), which have unique names.

Representing Strong Entity Sets

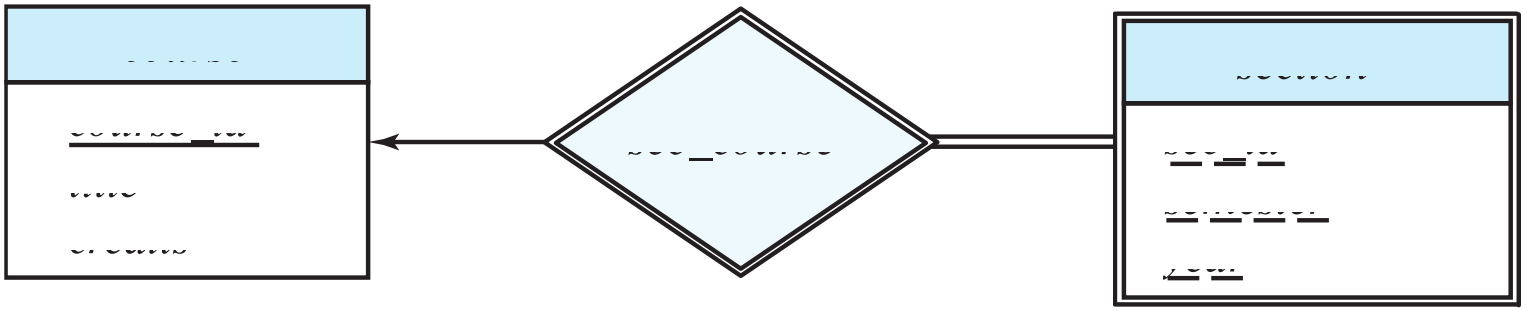
- A strong entity set reduces to a relation schema with the same attributes
- the primary key of the entity set serves as the primary key of the resulting relation schema

  student(ID, name, tot_cred)

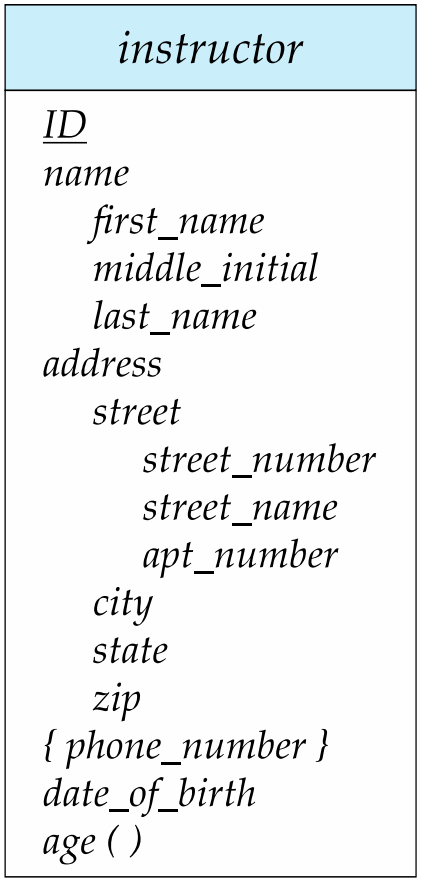
- Continuing with our example, for the University E-R diagram, all the strong entity sets, except *time slot*, have only simple attributes.
- The schemas derived from these strong entity sets are as follow:
  - *classroom*(*building*, *room_number*, *capacity*)
  - *department*(*dept name*, *building*, *budget*)
  - *course*(*course_id*, *title*, *credits*)
  - *instructor*(*ID*, *name*, *salary*)
  - *student*(*ID*, *name*, *tot_cred*)

Representing Weak Entity Sets

- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set
- For tables derived from a weak entity set, the combination of the primary key of the strong entity set and the discriminator of the weak entity set serves as the primary key of the relation schema.
- In addition to creating a primary key, we also create a foreign-key constraint on the weak relation.
- The foreign-key constraint ensures that for each tuple representing a weak entity, there is a corresponding tuple representing the corresponding strong entity.
  section (course_id, sec_id, sem, year)



Representation of Entity Sets with Composite Attributes



- 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)

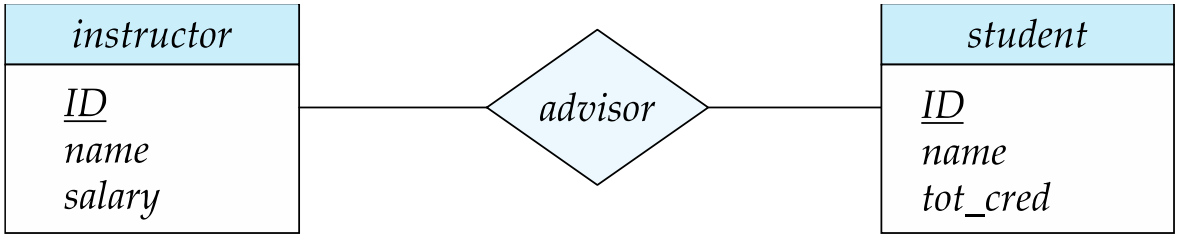
Representation of Entity Sets with Multivalued Attributes

- 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

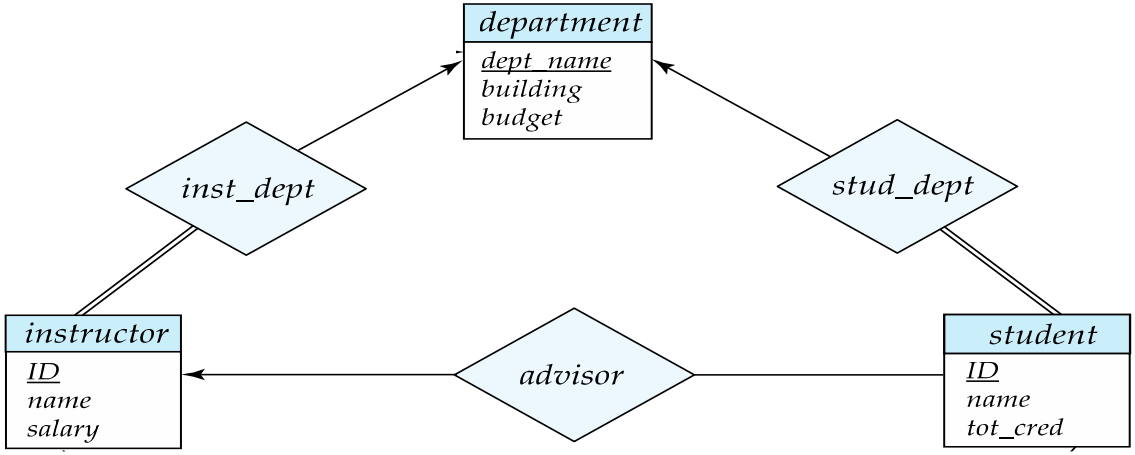
- A many-to-many relationship set is represented as a relation schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set *advisor*

advisor = (*s_id, i_id*)



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 relation schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from entity set *instructor*
- Example

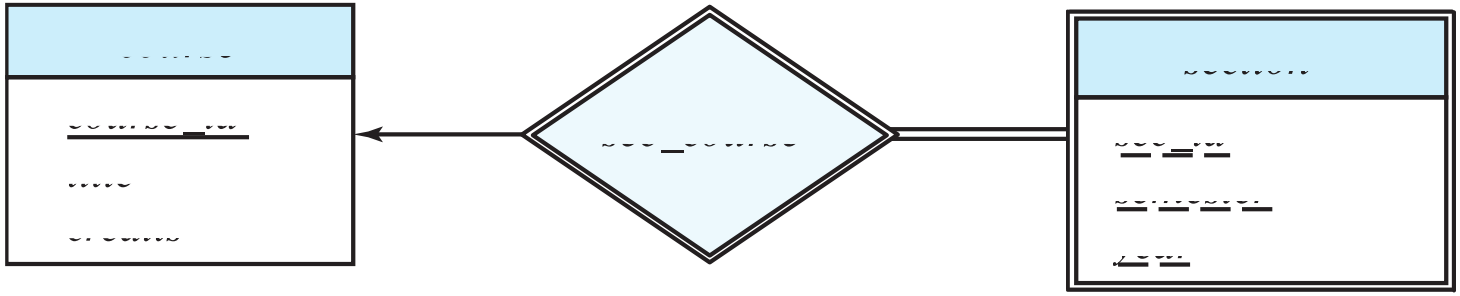


Redundancy of Schemas (Cont.)

- 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



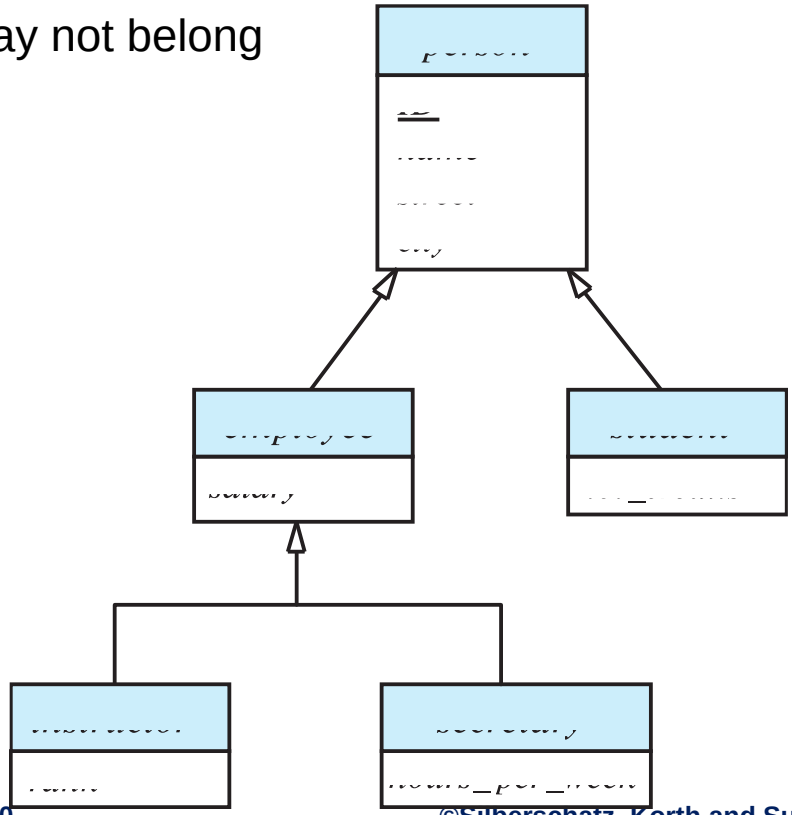
Extended E-R Features

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

- The way we depict specialization in an E-R diagram depends on whether an entity may belong to multiple specialized entity sets or if it must belong to at most one specialized entity set. The former case (multiple sets permitted) is called **overlapping specialization (***employee* and *student***)**, while the latter case (at most one permitted) is called **disjoint specialization (***instructor* and *secretary***)**.
- **Total** : Each higher-level entity must belong to a lower-level entity set.
- **Partial**: Some higher-level entities may not belong to any lower-level entity set.



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.

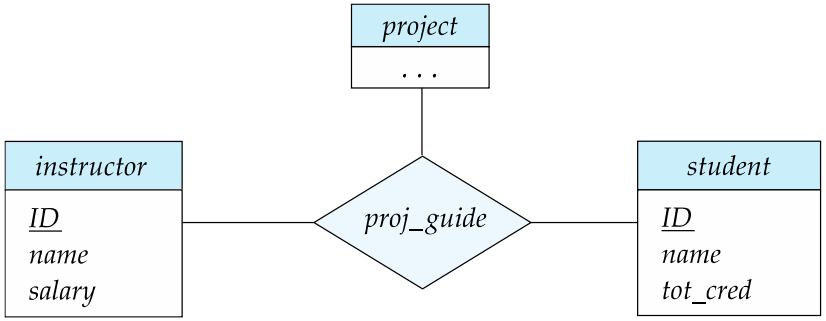
Completeness constraint

- **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

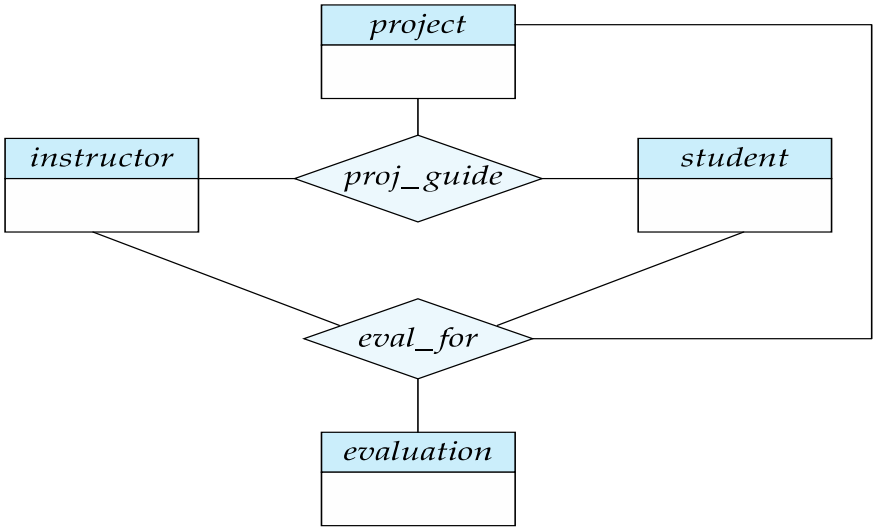
- The specialization of *person* to *student* or *employee* is total if the university does not need to represent any person who is neither a *student* nor an *employee*. However, if the university needs to represent such persons, then the specialization would be partial.

Aggregation

- One limitation of the E-R model is that it cannot express relationships among relationships.
- Consider the ternary relationship *proj_guide. students* work on research *projects* under the guidance of an *instructor*.



- Suppose we want to record evaluations of a student by a guide on a project

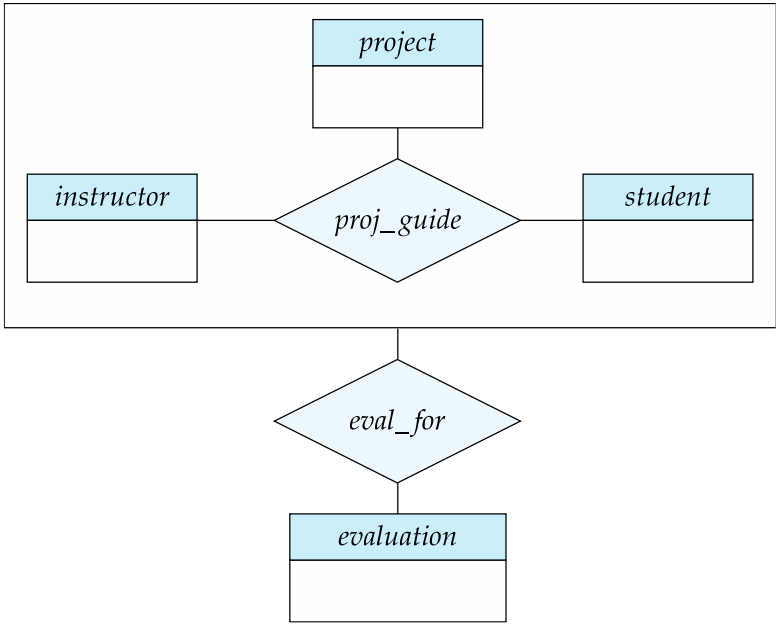


Aggregation (Cont.)

- 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

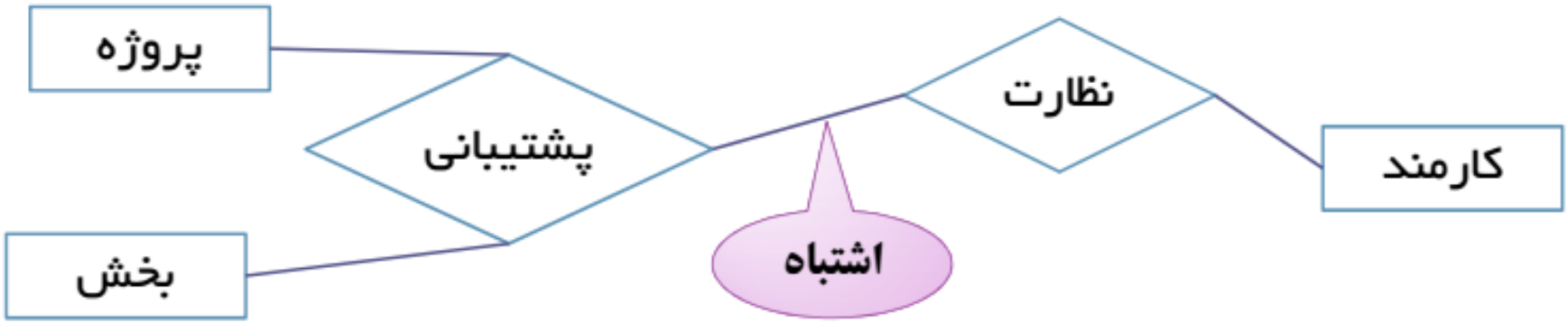
Aggregation (Cont.)

- 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

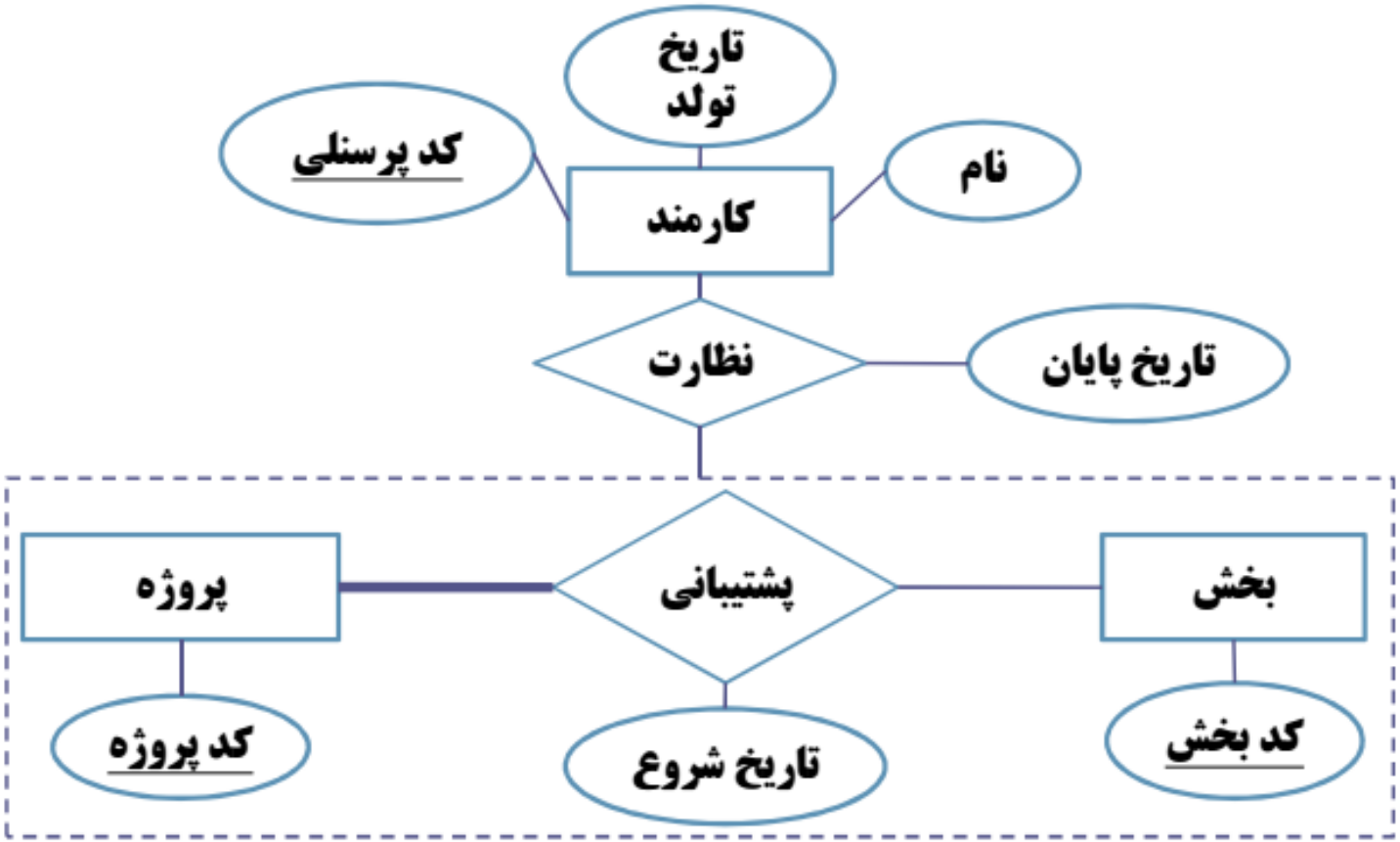


تجمع یا مجتمع سازی یعنی یک نوع موجودیت جدید را بر اساس دو یا چند موجودیت مرتبط با هم، به صورت یکپارچه در یک نوع موجودیت واحد ارائه نماییم. بدیهی است که این نوع موجودیت واحد خود می تواند با نوع موجودیت های دیگر نیز در ارتباط باشد.

مثلاً در یک سازمان رابطه ای به عنوان پشتیبانی وجود دارد که در آن یک پروژه توسط یک بخش پشتیبانی مالی شود. مدیریت، یک کارمند را برای نظارت بر روند پشتیبانی از پروژه انتخاب می کند حال رابطه این کارمند با رابطه پشتیبانی بصورت نظارت است. در حالت عادی نمی توان بین دو رابطه مجزا ارتباط برقرار کرد. مثلا شکل زیر صحیح نیست.



شکل صحیح رابطه نظارت بر رابطه پشتیبانی بصورت زیر است:



Reduction to Relational Schemas

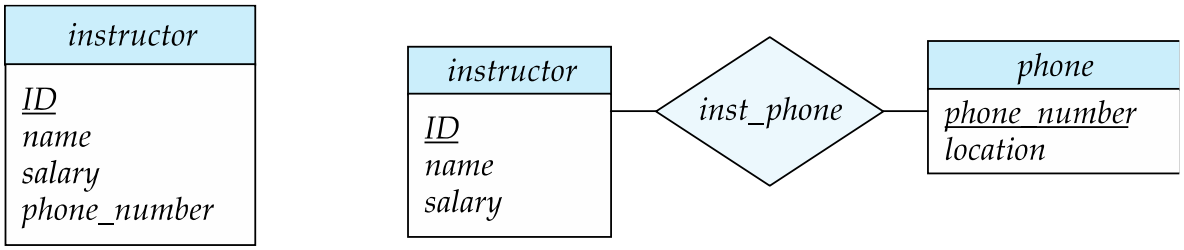
- To represent aggregation, create a schema containing
  - Primary key of the aggregated relationship,
  - The primary key of the associated entity set
  - Any descriptive attributes
- In our example:
  - The schema *eval_for* is:

    eval_for (*s_ID, project_id, i_ID, evaluation_id*)

Design Issues

Entities vs. Attributes

- Use of entity sets vs. attributes

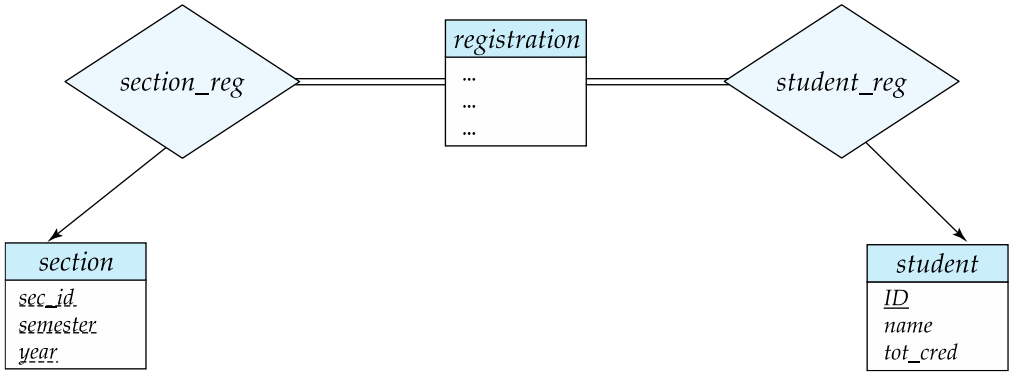


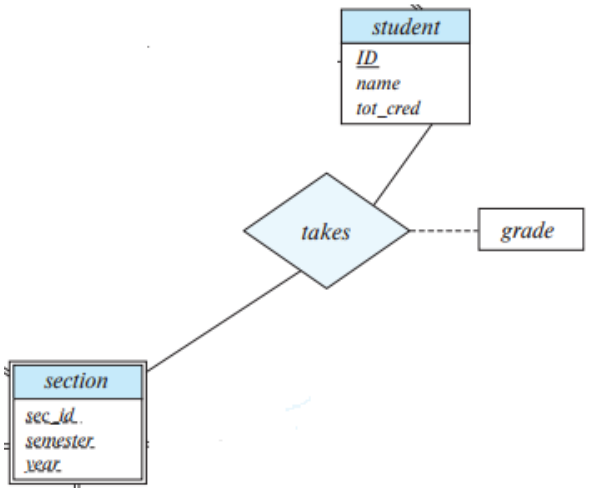
- Use of phone as an entity allows extra information about phone numbers (plus multiple phone numbers)

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



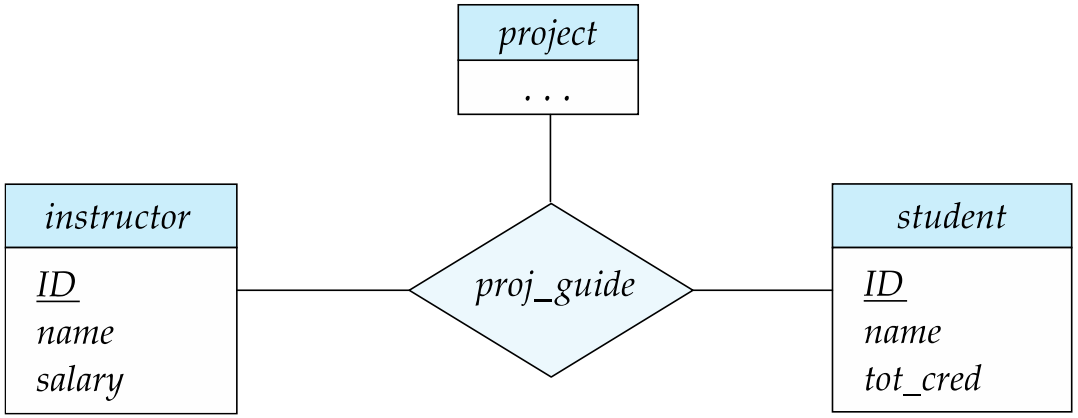


Degree of a Relationship Set

- Binary relationship
  - involve two entity sets (or degree two).
  - most relationship sets in a database system are binary.
- Relationships between more than two entity sets are rare. Most relationships are binary. (More on this later.)
  - Example: *students* work on research *projects* under the guidance of an *instructor*.
  - relationship *proj_guide* is a ternary relationship between *instructor*, *student,* and *project*

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



Binary Vs. Non-Binary Relationships

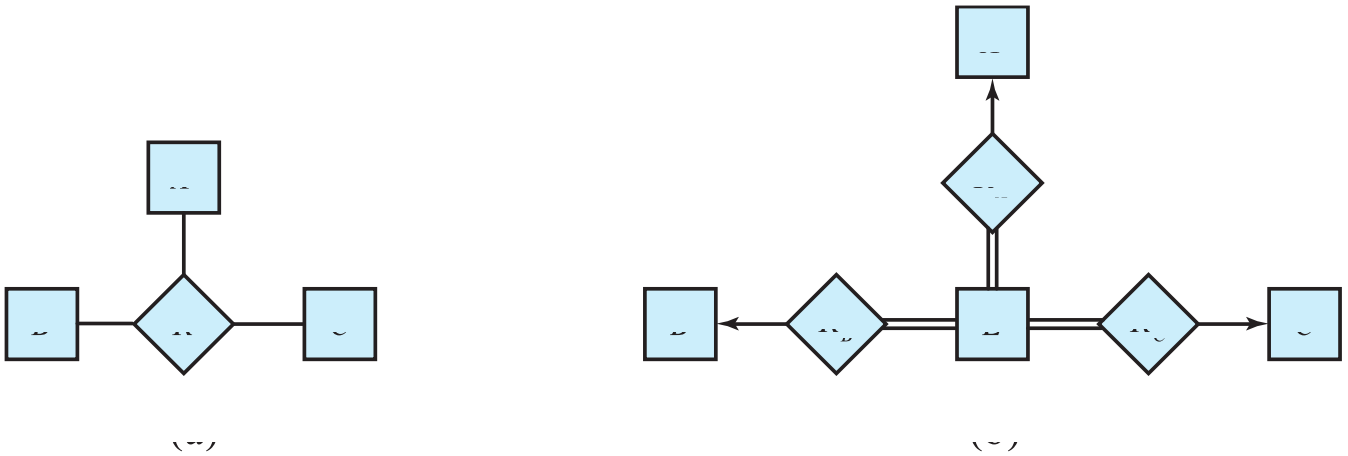
- Although it is possible to replace any non-binary (*n*-ary, for $n > 2$) relationship set by a number of distinct binary relationship sets, a *n*-ary relationship set shows more clearly that several entities participate in a single relationship.
- Some relationships that appear to be non-binary may 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

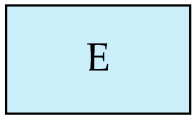
- In general, any non-binary relationship can be represented using binary relationships by creating an artificial entity set.
  - Replace R between entity sets A, B and C by an entity set E, and three relationship sets:

    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

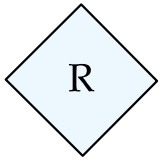
    1. a new entity e_i in the entity set E
    2. add (e_i , a_i) to R_A
    3. add (e_i , b_i) to R_B
    4. add (e_i , c_i) to R_C



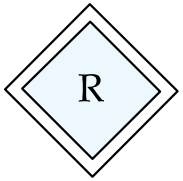
Summary of Symbols Used in E-R Notation



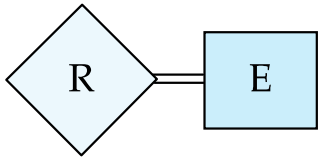
entity set



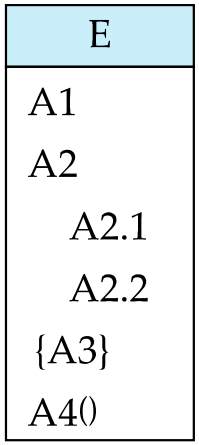
relationship set



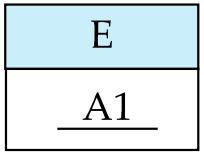
identifying relationship set for weak entity set



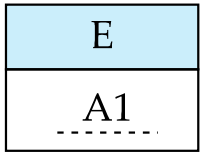
total participation of entity set in relationship



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

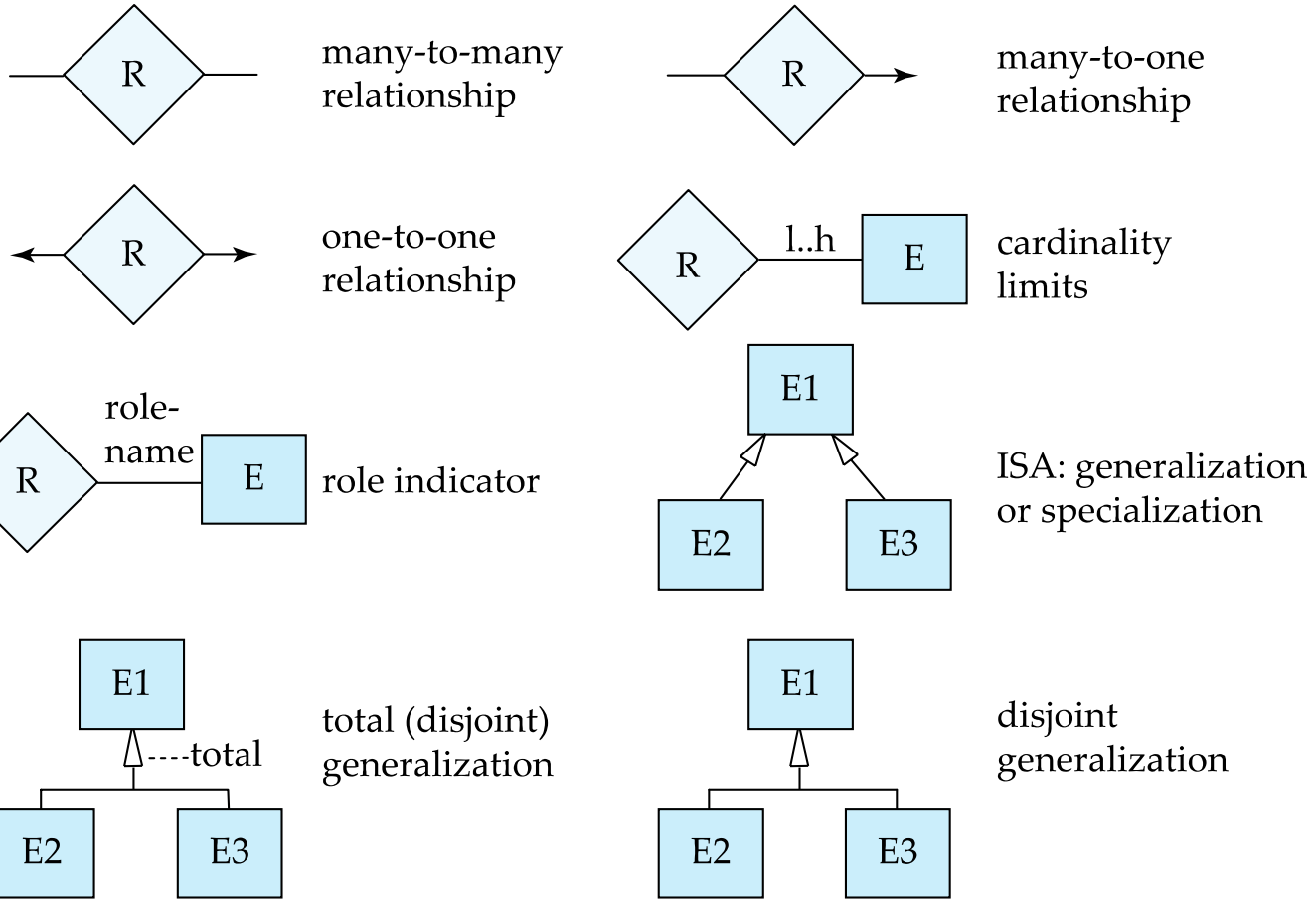


primary key



discriminating attribute of weak entity set

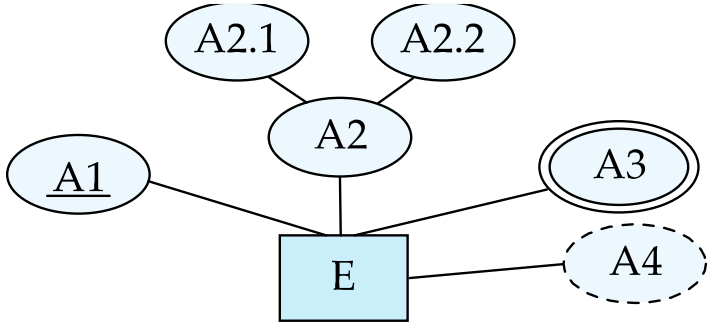
Symbols Used in E-R Notation (Cont.)

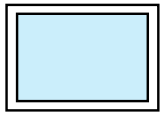


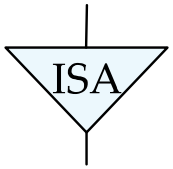
Alternative ER Notations

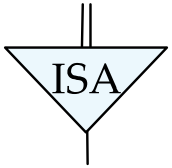
- Chen, IDE1FX, …

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

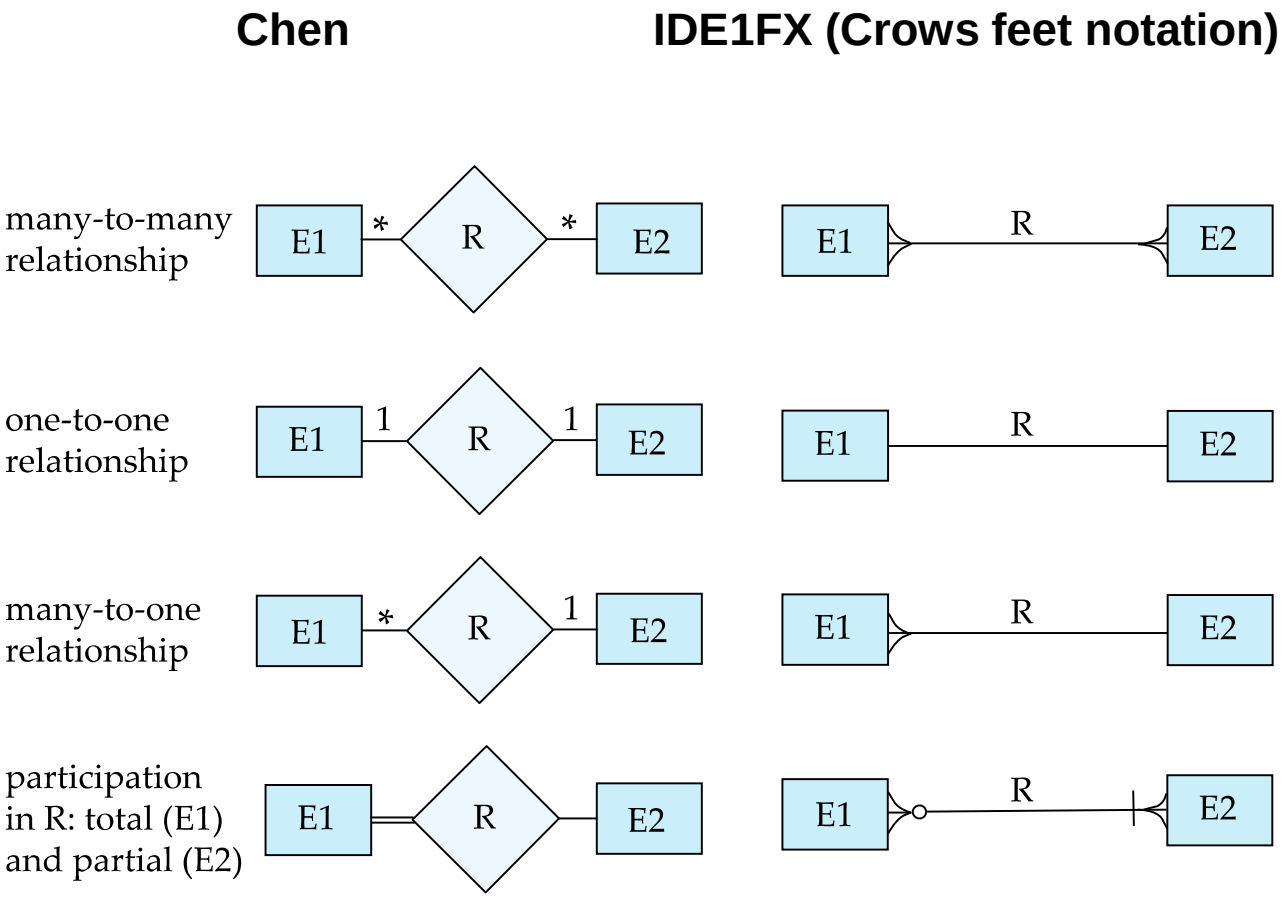


weak entity set 

generalization 

total generalization 

Alternative ER Notations



End of Chapter 6