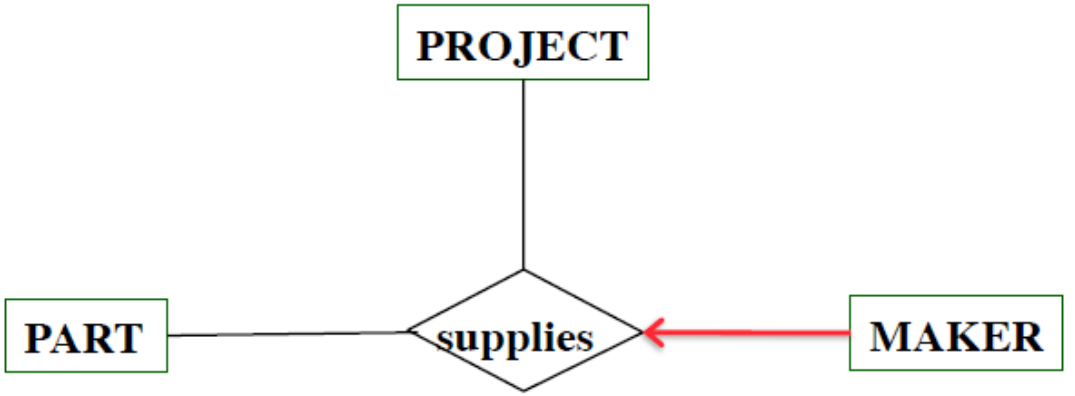
n-ary Relationships

If for each project the employee is supervised by someone else, it is better to have a *ternary* relationship.

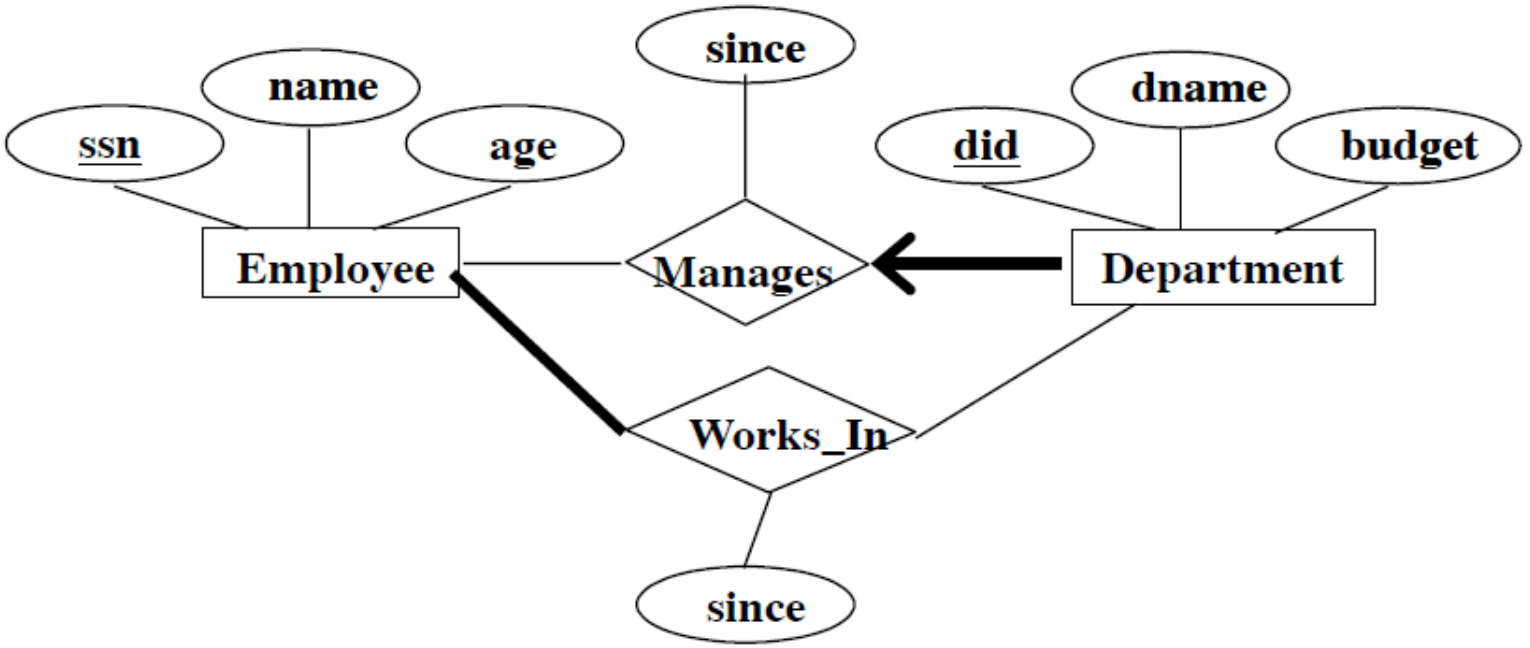


(*Each maker supplies at most one part and it supplies it to at most one project. So in the set of (maker, part, project)-tuples, the* maker *field is a key.)*

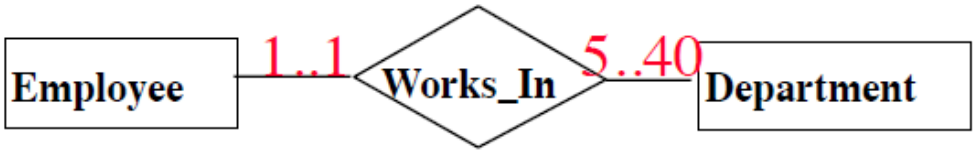
The particular variant of ER diagrams you are being taught does not allow saying that a maker supplies at most one part to each project, but may supply parts to multiple projects

Relationship Participation Constraints

- Does every employee work somewhere? (Is every Employee entity involved in some Works_In relationship?)
  - **If so, this is a *participation constraint*: the participation of Employee in Works_In (vs. Employee in Manages) is said to be *total* (vs. *partial*). [*thick edge* notation]**
  - **Arrow can be thick, to indicate both total and key**



(More general cardinality constraints)



- "Every department participates in 5 to 40 **worksIn** relationships -- (emp, dept) pairs
- "Every employee participates in at least 1 and at most 1 **worksIn** relationships."

Summary of participation constraint notation

- In a diagram for relationship R, each entity E is considered separately. We record what constraints there are on any *instance* of E, in terms of how many specific relationships/tuples it can participate in.
- The choices are
  - **no constraints**

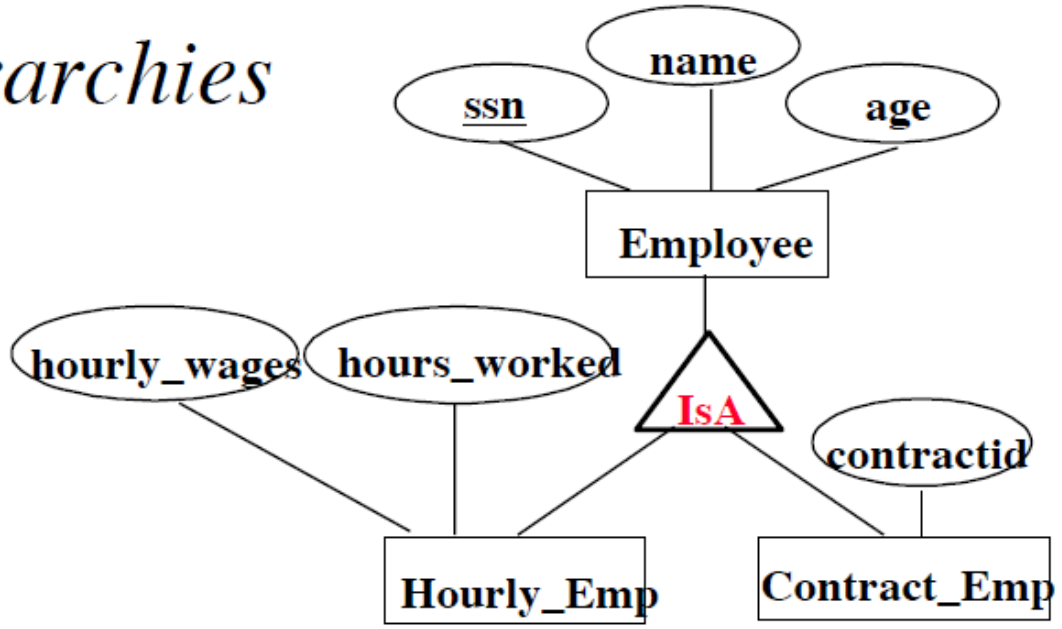
    (notation: simple edge; 0.. in cardinality notation)*
  - **at least one relationship for each instance**

    (notation: thick edge; 1.. in cardinality notation)*
  - **at most one relationship for each instance**

    (notation: edge with arrow; 0..1 in cardinality notation)
  - **at least one relationship for each instance**

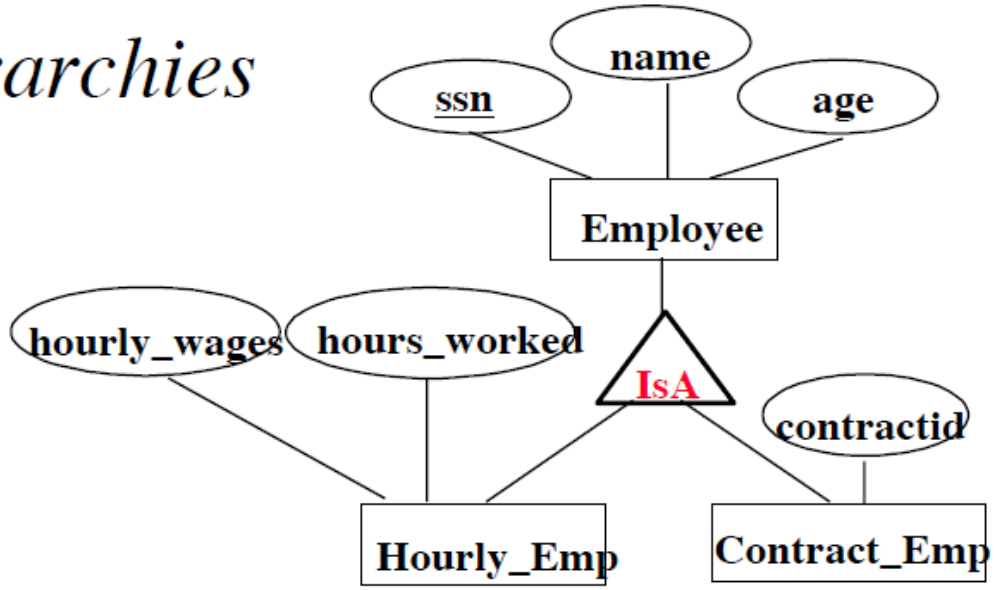
    (notation: thick edge with arrow; 1..1 in cardinality notation)

ISA (`is a') Hierarchies



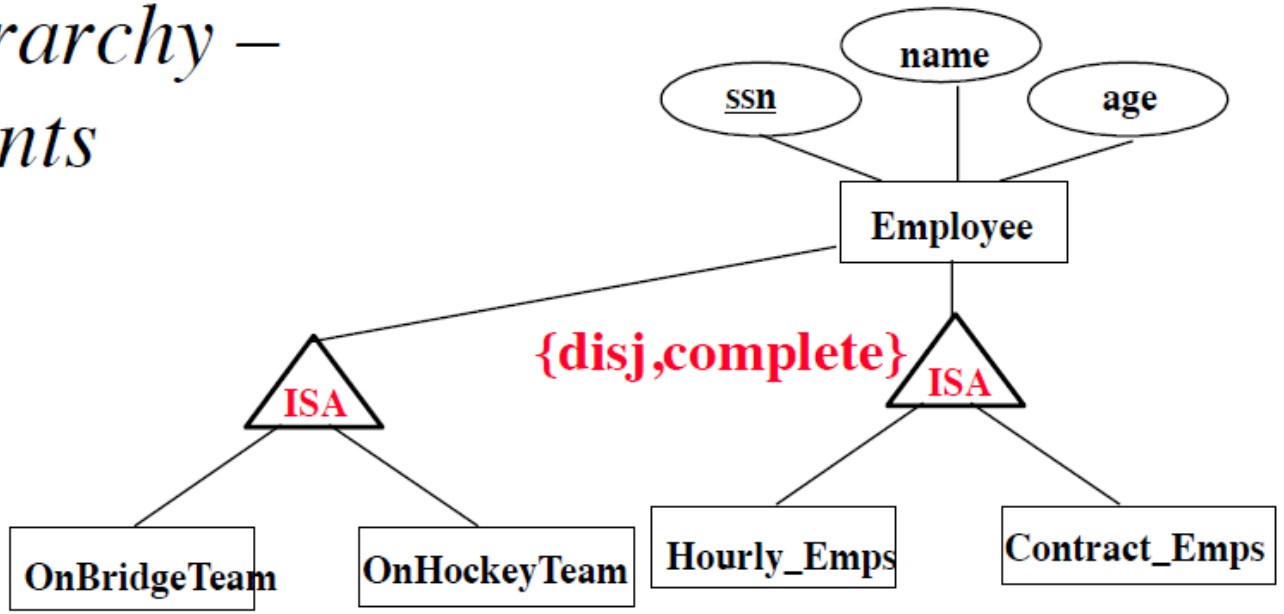
- A **IsA** B *characteristic*: every A entity set instance is also a B entity set instance.
- Attributes are *inherited*. (So are relationships)

ISA (`is a') Hierarchies



- When to use use **IsA:**
  1. **To factor out commonalities and avoid repetition:** eg, *age*
  2. **To more precisely identify subset of entities that are domains/co-domains of a relationship** (*Contract_Emps* participates totally in *ManagerOf* reln, and *Hourly_Emps* do not participate at all) **or have specific attributes** (*contractId*)
  3. **To organize large models and the process of creating/ modifying models** (*inheritance of changes*)

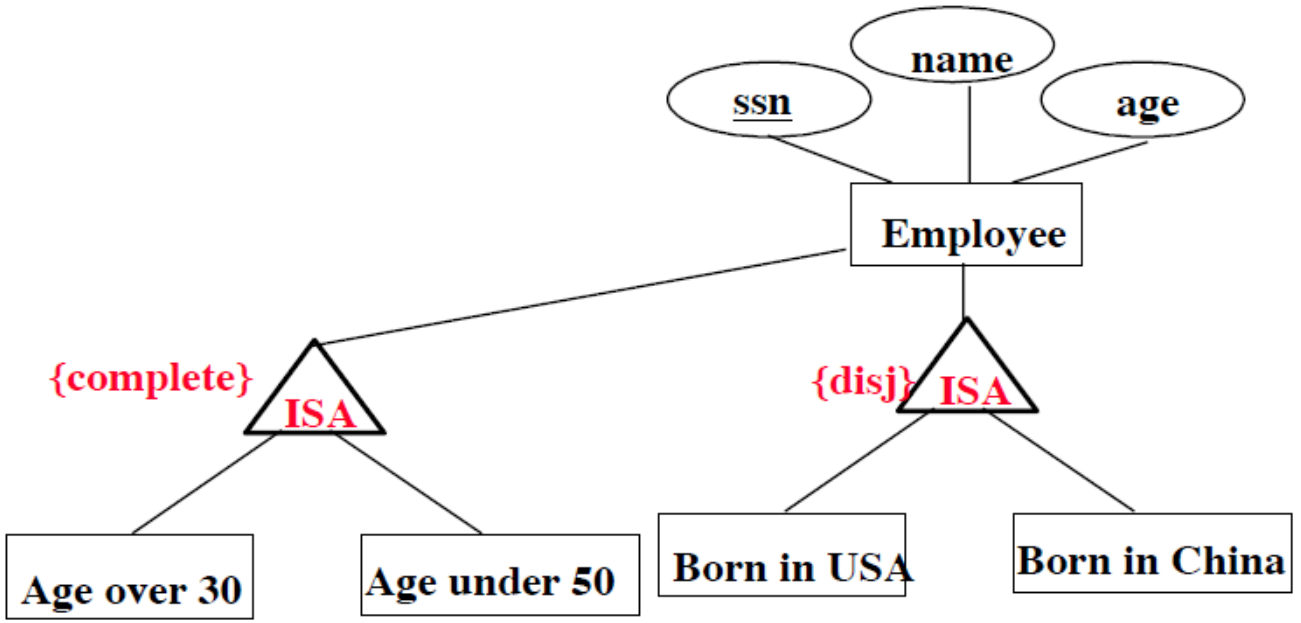
ISA Hierarchy – constraints



Constraints on IS-A hierarchies:

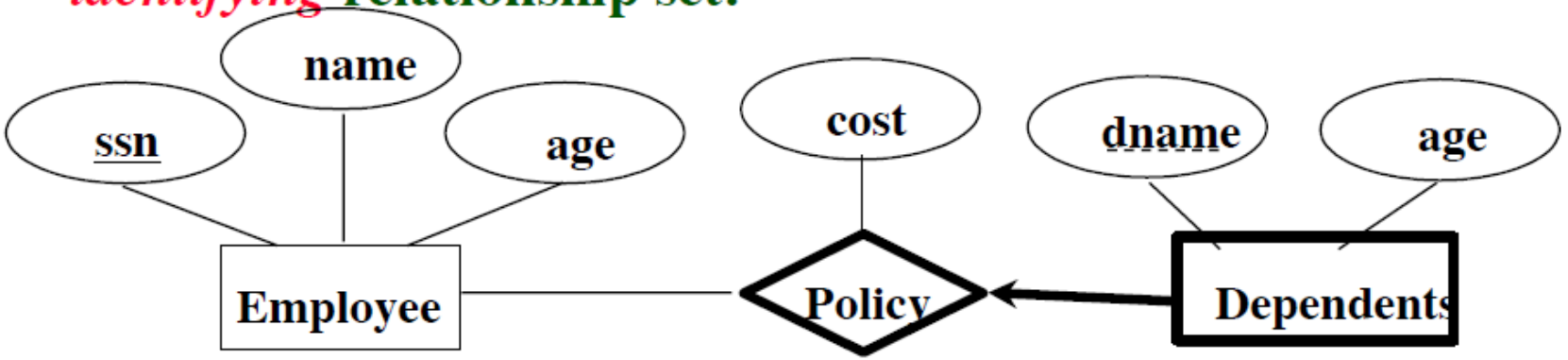
- *Overlap constraint*: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? ***= disjoint*** *(vs. overlapping)*
- *Covering constraint:* : Does every Employee entity also have to be an Hourly_Emps or a Contract_Emps entity? ***= complete*** *(vs. partial) cover*

ISA Hierarchy constraints (cont'd)



Weak Entities

- A *weak entity* does not have enough attributes "stored" to identify it uniquely by their values. It must therefore be identified through its relationship to another (*owner*) entity.
  - **Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).**
  - **Weak entity set must have total participation in this *identifying* relationship set.**



Conceptual Design Using the ER Model

Modeling choices:

1. Should a concept be modeled as an entity or an attribute?
2. Should a concept be modeled as an entity or a relationship?
3. Relationship kinds:
   A. **Binary vs ternary**
   B. **Describing Relationships: Aggregation and Reification**
   C. **Manifestation**
   D. **(partOf)**

1. Entity vs. Attribute

- Should *address* be an attribute of Employee or an entity (connected to Employee by a relationship like ***livesAt***)?
- Depends upon the use we want to make of address information, and the semantics of the data:
  - If your *home* (which is what has an address) participates in relationships (e.g., it is being sold,...) then it becomes of interest in its own right. The decision is really about where you draw the *boundary* of your Universe of Discourse.
  - (In Ramakrishnan's text, since attributes must be single valued and unstructured, also need entities for address if you have *multiple* addresses, or an address *has components*: street, city.)

2. Attribute vs Relationship

- Could think of *manager* as an attribute of an employee
  - **must have a integer/string value** (manager's SS# or EmpId)
  - **cannot have a direct attribute of its own *e.g.*, *mngr_phone*** (must be stored on *all* the people for whom she is a manger!)

3a) Relationships : Binary vs. Ternary

- Consider a ternary relation Supplies relating entity sets Part, Manufacturer and Project, with descriptive attribute *quantity*.
- Is there a combination of binary relationships that represents it?
  - **Manuf "can-supply" Part,**
  - **Proj "needs" Part,**
  - **Proj "deals-with" Manuf**

  But this does not imply that Pr has agreed to buy Pa from Ma. And how would we record *quantity*?
- **So 'supplies' cannot be replaced by 3 binary relationships**

(DRAW THESE)

3a) Relationships : Binary vs. Ternary (cont'd)

BUT, in some cases, 3-argument relationships are *ill-conceived* and should really be represented by binary relationships:

consider **worksIn(Employee,Factory,Company)**

"Employee works in Factory owned by Company"

As you can tell from the English even (2 verbs in this sentence: "works ", "owned"), there are in fact two relationships here

worksIn(Employee,Factory),

ownedBy(Factory,Company)

Avoid such complex relationships (use "minimal information facts")

Binary vs. Ternary Relations will be revisited when we do "*lossless decomposition of relations*" based on "functional dependencies"

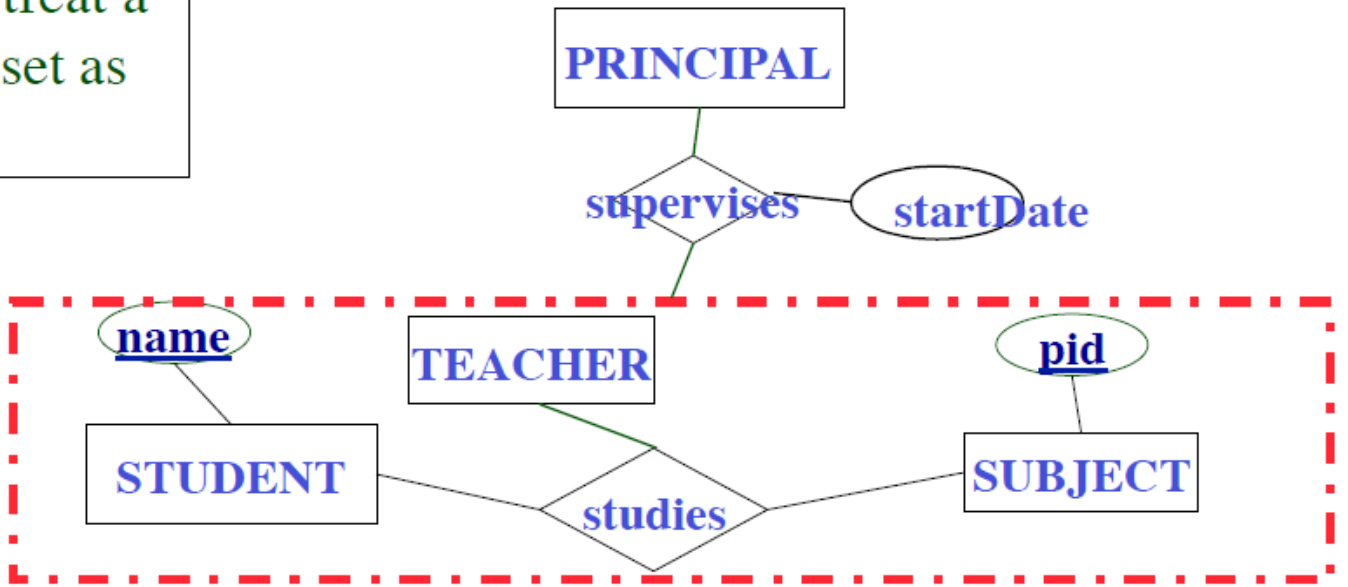
3b) Describing Relationships

- Relationships may need to be "described" by more complex things than simple attributes
  - ***teaches(Teacher,Student,Subject)* is *supervisedBy Principal***
  - do not like the idea of writing a single 4-place relationships ***teachingSupevisedBy(Tchr,Std,Subj,Principal)*** because this is composite!
  - another eg. '**"Anna likes Vronya" was said by Tony**'
- Relationships might participate in IsA
  - ***sonOf* is a specialization of *childOf* is a specialization of *relativeOf***

Describing Relationships in ER: "Aggregation"

- Relationships may need to be "described" by more complex things than simple attributes
  - ***teaches(Teacher,Student,Subject)*** **is** ***supervisedBy Principal***

Aggregation
allows us to treat a relationship set as an entity set

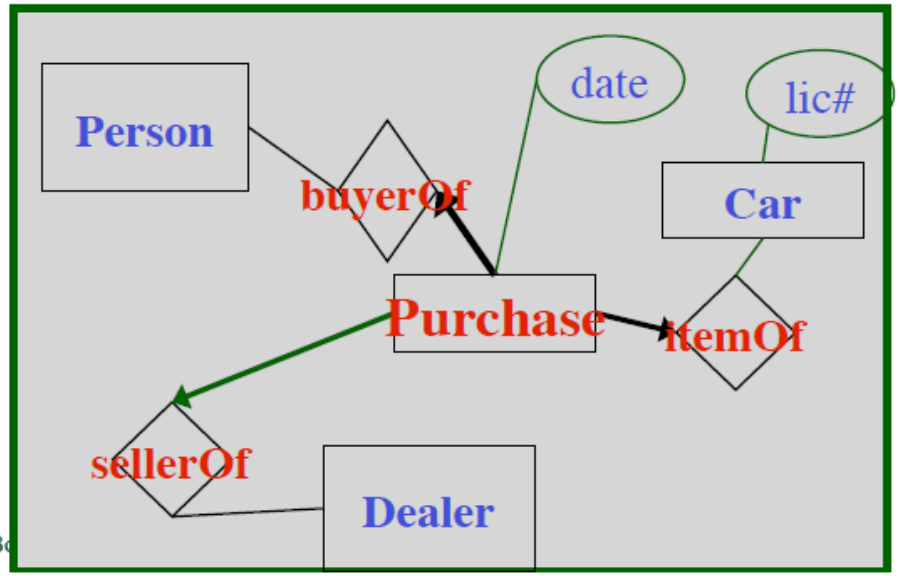


Alternative: **Reifiying ("making real") relationships**

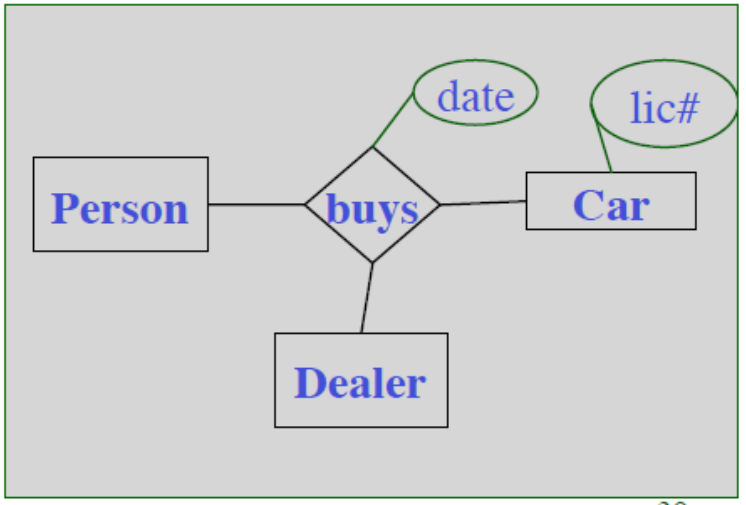
- take a relationship R and think of it as an entity, linked by *simple functional binary relationships* to the *participants* in R.
- Use the role names to label these binary relationships

BUT, beware that 2. allows duplicate purchases not allowed by 1. (purchase_i and purchase_ii entities, both linked to the same person, car and dealer instance.)

2



1

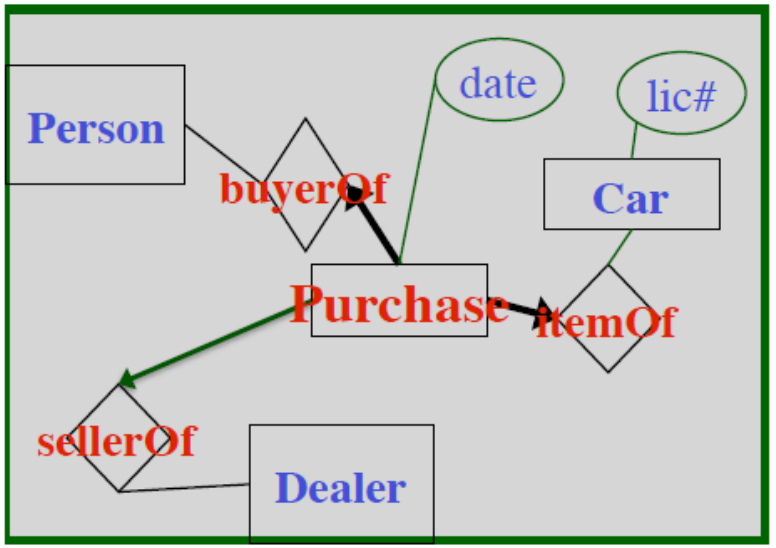


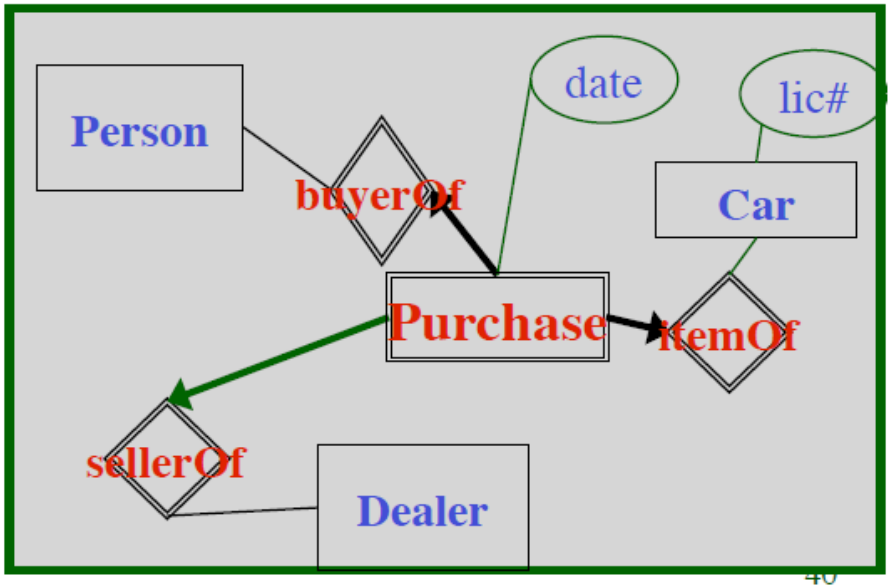
Reification (cont'd)

- Problem 1: (A) allows duplicate purchases, as noted on the previous page
- Problem 2: entity Purchase does not have a key – illegal in ER!

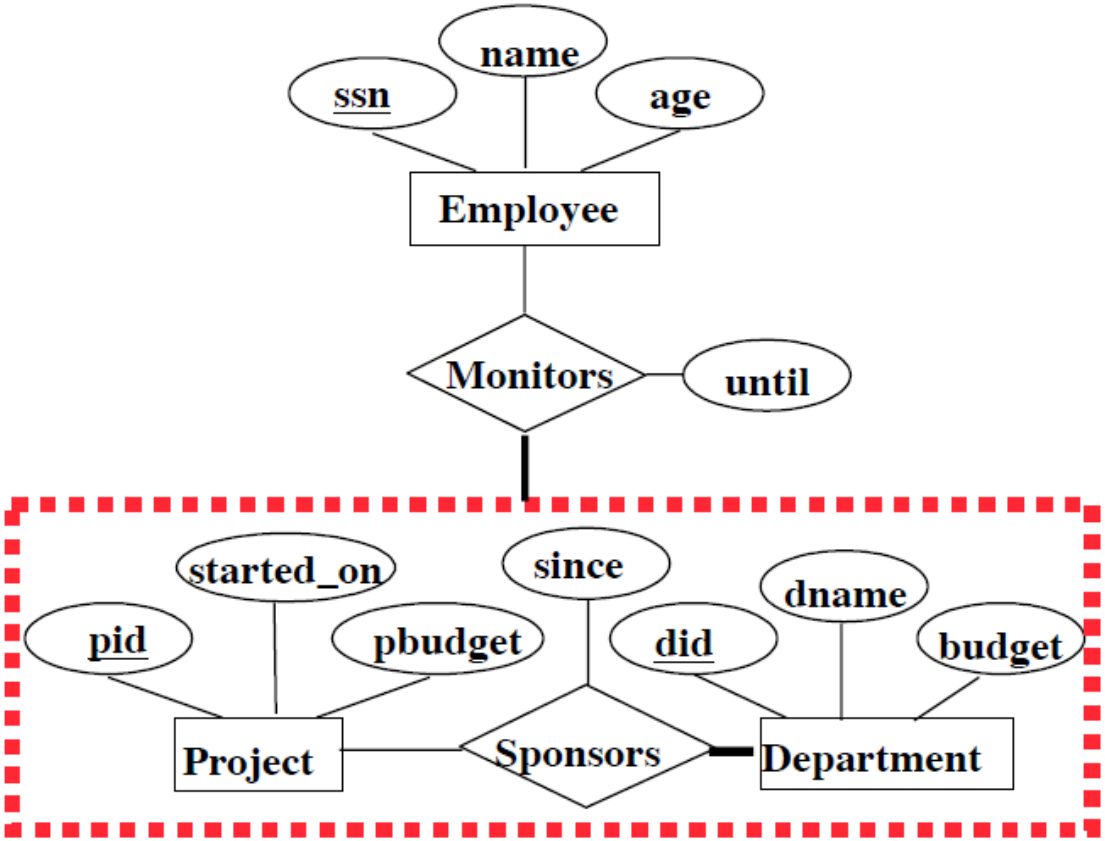
Solution to both: make Purchase be a *weak entity*

2



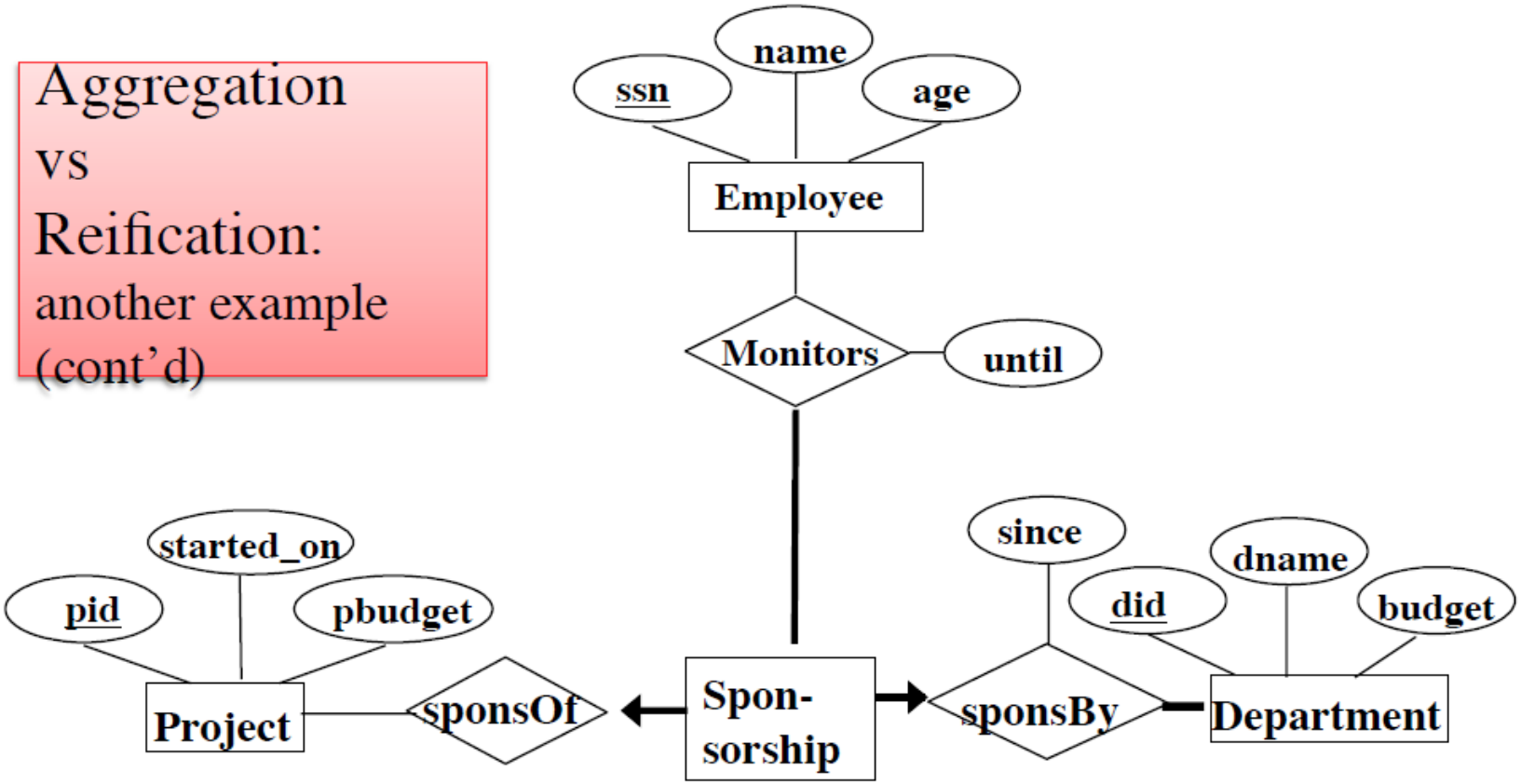


Aggregation
vs
Reification:
another example



Represented as Aggregation

Aggregation vs Reification: another example (cont'd)

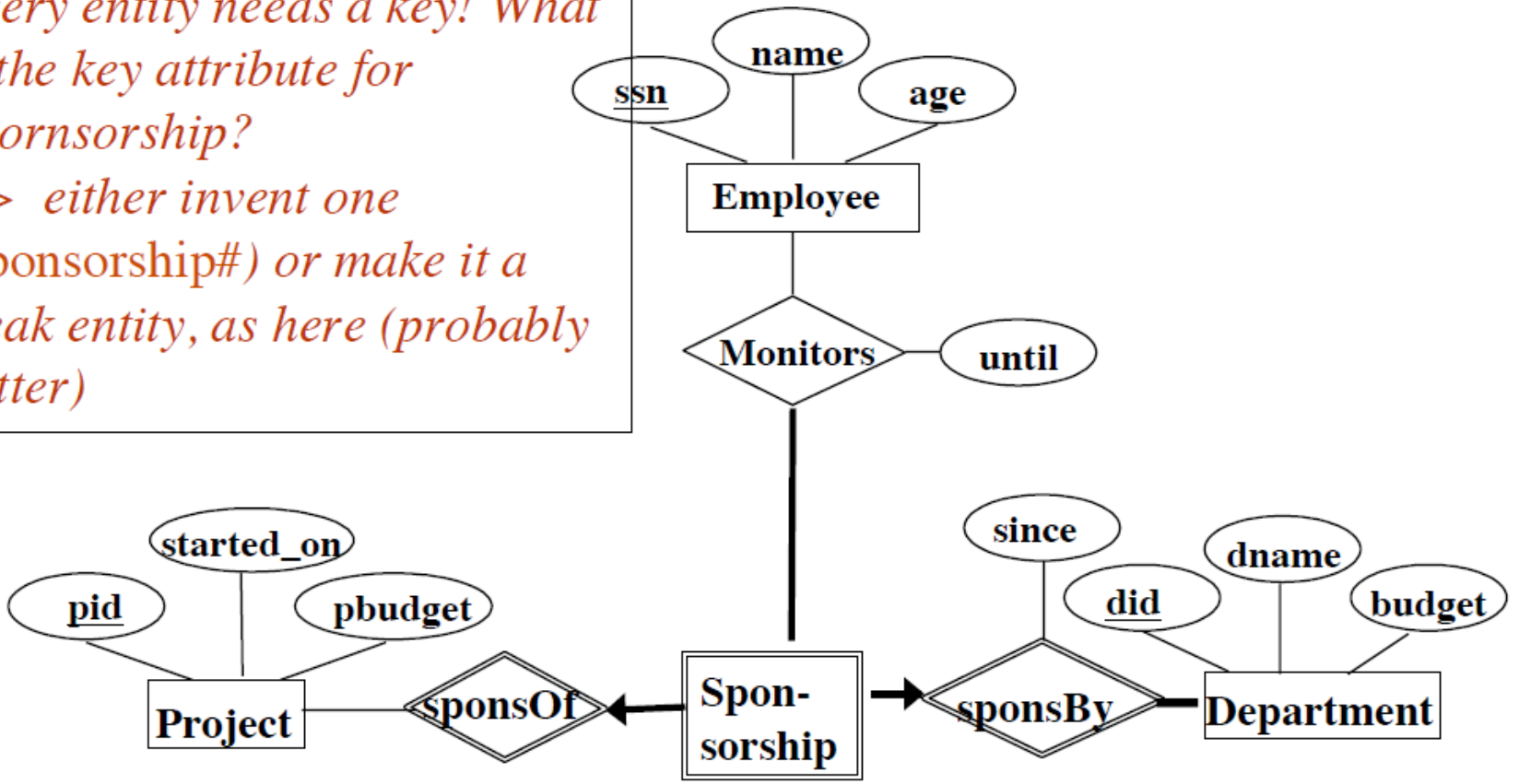


Attempted Represented by Reification

[Note how the reified entity is labeled by a noun ("Sponsorship") derived from the verb ("Sponsors") naming the relationship.]

Reification: keys

Every entity needs a key! What is the key attribute for Spornsorship?
>> either invent one (sponsorship#) or make it a weak entity, as here (probably better)



(Another example of Reifying Relationships)

Instead of Lend(Library,Material,ToPatron,...)

Loan **with relationships**

--<*lentTo*>-- Patron

--<*onLoan*>-- Material

--<*lentOn*>-- Date

--<*dueOn*>-- Date

--<*NrOfRenewals*>-- {0,1,2,3}

BENEFITS: This allows you to declare *specialized/sub relations like "lending for short time":*

ShortermLoan IsA Loan