Class Hierarchies

- As in C++, or other PLs, attributes are inherited.
- If we declare A ISA B, every A entity is also considered to be a B entity.

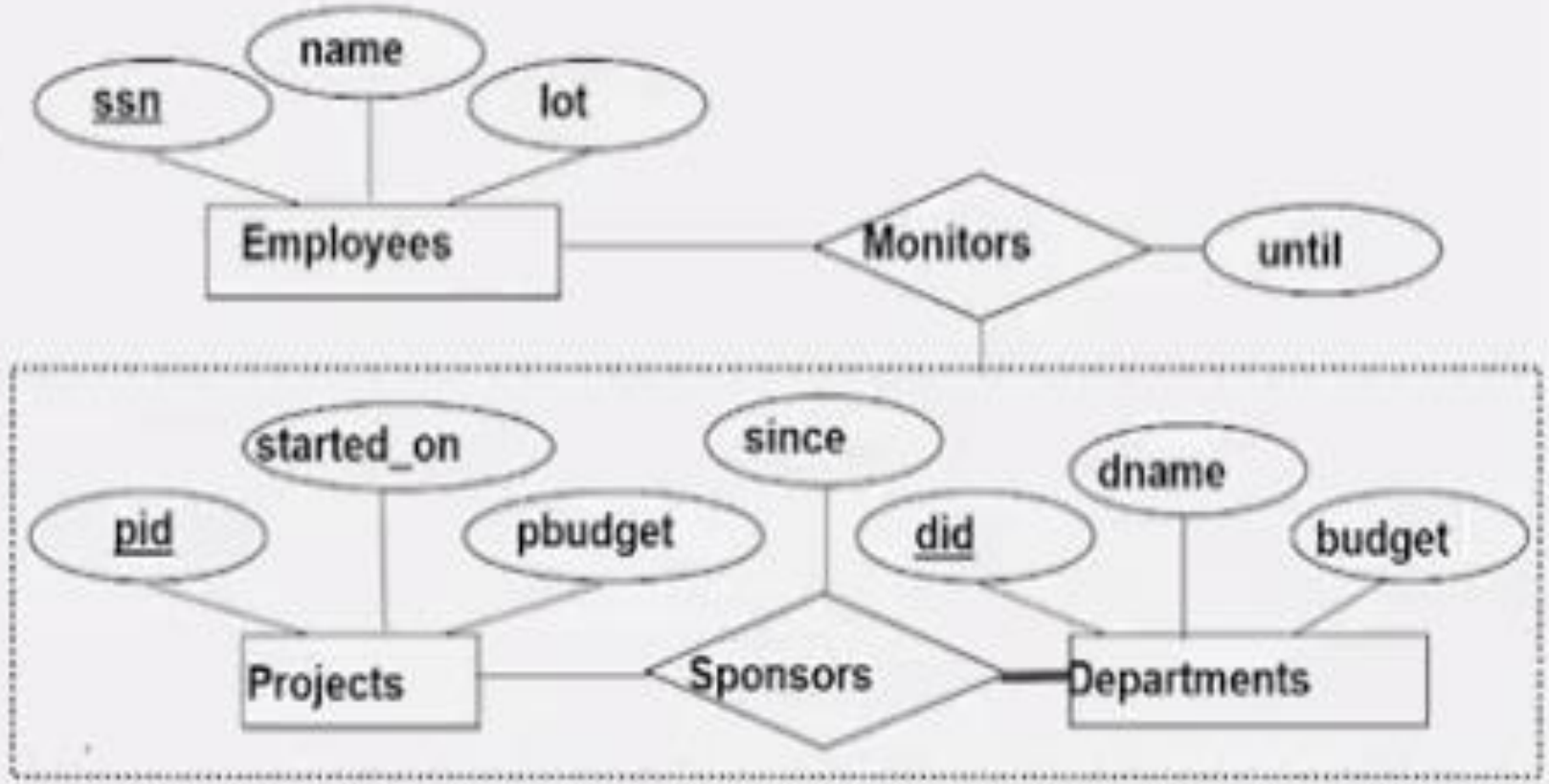
- *Overlap constraints*:
  Can Joe be an Hourly_Emps as well as a Contract_Emps entity? *(Allowed/disallowed)*
- *Covering constraints*: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/no)*
- Reasons for using ISA:
  - To add descriptive attributes specific to a subclass.
  - To identify entities that participate in a relationship.

Where were we?

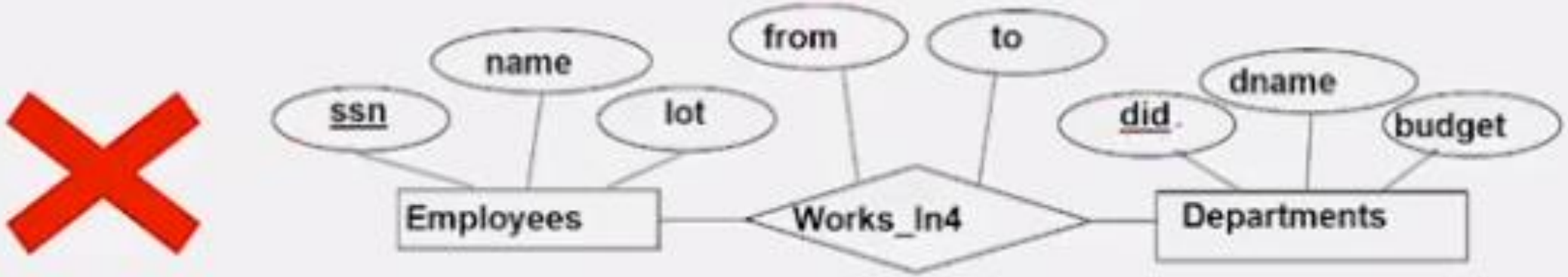
- *Conceptual design* follows *requirements analysis*,
  - Yields a high-level description of data to be stored
- ER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications.
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships).
- Some additional constructs: *weak entities*, *ISA hierarchies*, and *aggregation*.
- Several kinds of integrity constraints can be expressed in the ER model (Key, Participation, Overlap, Covering)

Aggregation

- Used when we have to model a relationship involving (entity sets and) a *relationship set*.
- *Aggregation* allows us to treat a relationship set as an entity set for purposes of participation in (other) relationships.

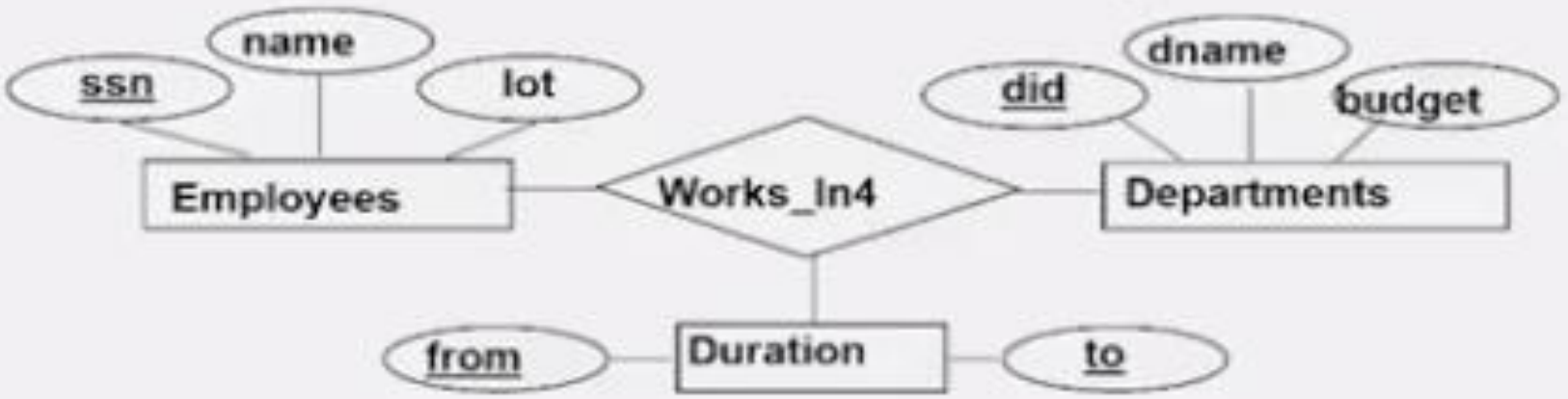


A relationship is identified by its participating Entities



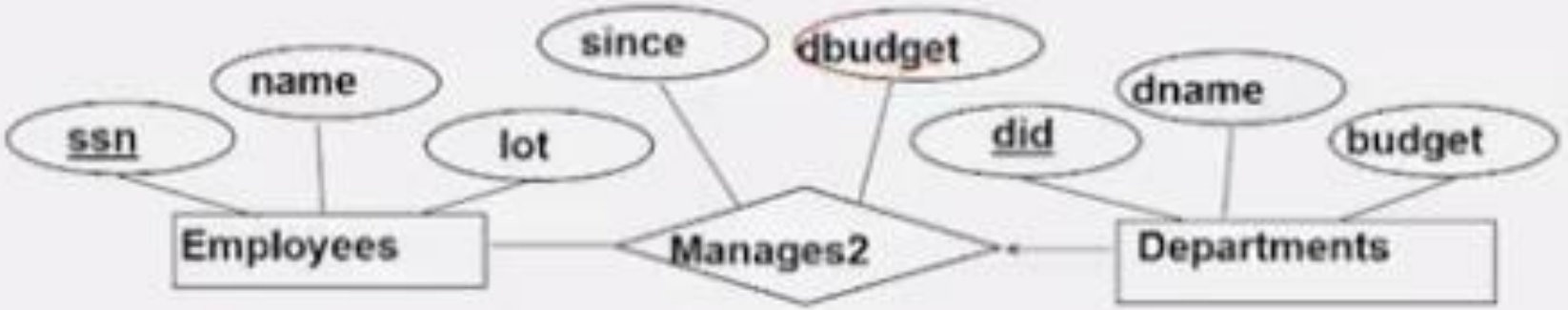
What if an Employee can work in a given Department for more than one period?

Better Design ?!

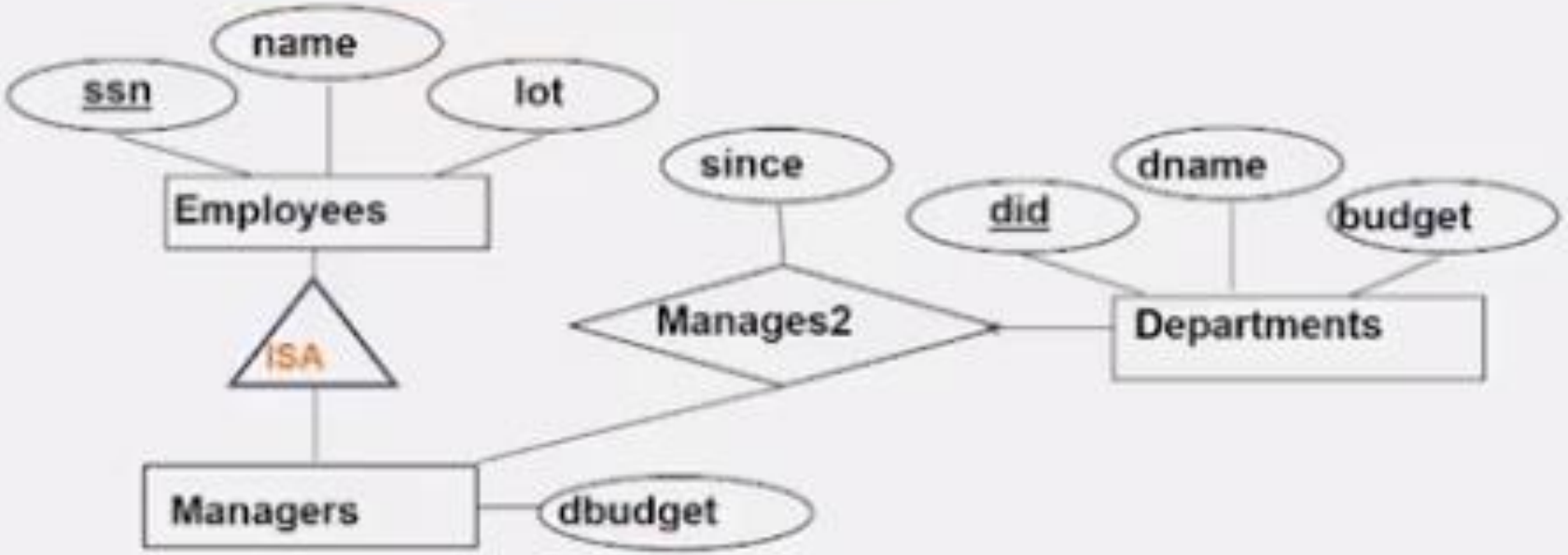


?

1


since
dbudget
name
dname
ssn
did
budget
lot
Employees
Manages2
Departments

2


name
ssn
lot
since
dname
Employees
did
budget
ISA
Manages2
Departments
Managers
dbudget

Binary vs. Ternary Relationships

- If each policy is owned by just 1 employee, and each dependent is tied to the covering policy, first diagram is inaccurate.