DATABASE MANAGEMENT SYSTEMS
UNIT 1 – TOPIC 7
Conceptual Design Using the ER Model

Developing an ER diagram presents several choices, including the following:

- Should a concept be modeled as an entity or an attribute?
- Should a concept be modeled as an entity or a relationship?
- What are the relationship sets and their participating entity sets?
- Should we use binary or ternary relationships?
- Should we use aggregation?

1. Entity vs. Attribute

While identifying the attributes of an entity set, it is sometimes not clear whether a property should be modeled as an attribute or as an entity set.

Example : Consider the relationship "Employee WorksIn Department", if we want to add address information to the Employee entity set.

Should *address* be an attribute of Employees or an entity (connected to Employees by a relationship)?

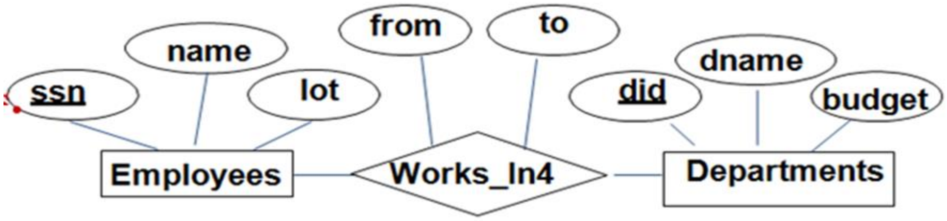
Now this depends upon the use we want to make of address information, and the semantics of the data:

- If we have several addresses per employee, *address* must be an entity (since attributes cannot be set-valued) and an association between employees and addresses using a relationship say Has_address.

- If the structure (city, state, zipcode etc.) is important, e.g., we want to retrieve employees in a given city, *address* must be modeled as an entity with these attributes (since attribute values are atomic).

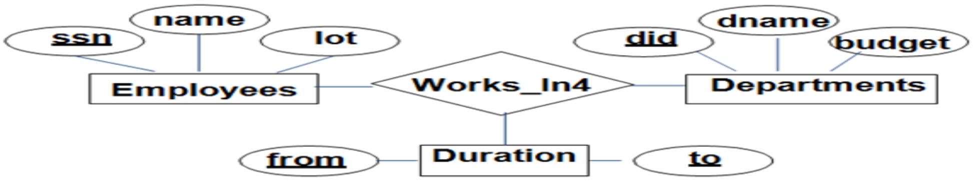
Consider another example below:

In the "Employee WorksIn Department" Relationship, it is possible for an employee to work in a given department over more than one period, which is not possible to accomodate as an attribute.



- Works_In4 does not allow an employee to work in a department for two or more periods.

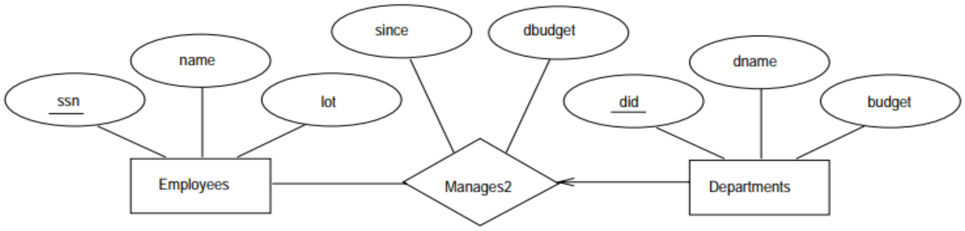
- Similar to the problem of wanting to record several addresses for an employee, We want to record *several values of the descriptive attributes for each instance of this relationship.* Accomplished by introducing new entity set, **Duration.**



-

2. Entity vs. Relationship

Look at the example below : Consider the relationship set called Manages.



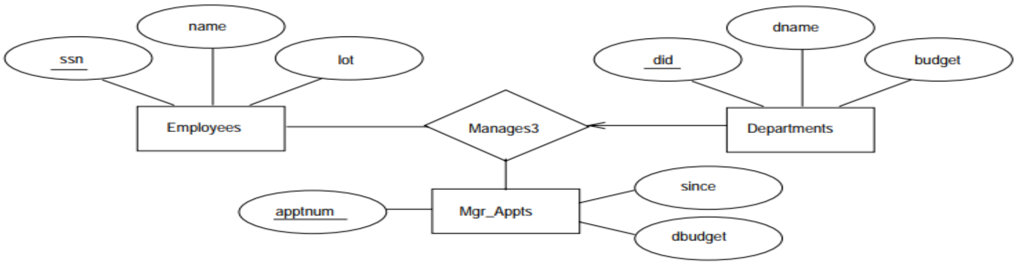
Suppose that each department manager is given a discretionary budget. There is at most one employee managing a department, but a given employee could manage several departments; we store the starting date and discretionary budget for each manager-department pair.

What if a manager manages more than one department and the discretionary budget is a sum of all the departments he manages.

Redundancy: given employee will have the same value in the dbudget field

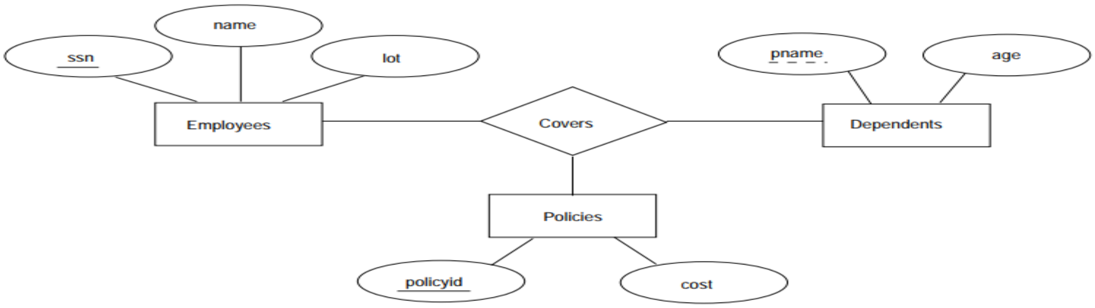
Misleading: Suggests dbudget associated with department - mgr combination.

We can address these problems by associating dbudget with the appointment of the employee as manager of a group of departments. In this approach, we model the appointment as an entity set, say Mgr_Appt, and use a ternary relationship, say Man ages3, to relate a manager, an appointment, and a department.



3. Binary vs. Ternary Relationships

Consider an ER Diagram that models a situation in which an employee can own several policies, each policy can be owned by several employees, and each dependent can be covered by several policies.



If each policy is owned by just 1 employee, and each dependent is tied to the covering policy, the ternary relationship as above is inaccurate.

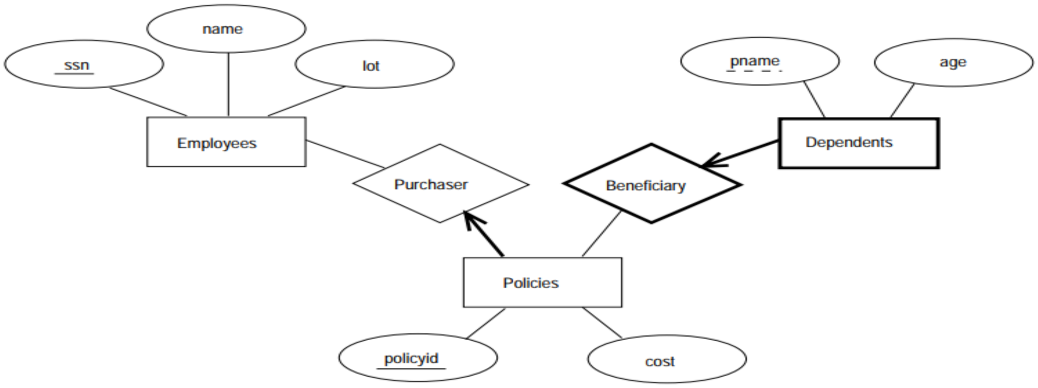
Suppose that we have the following additional requirements:

- A policy cannot be owned jointly by two or more employees.
- Every policy must be owned by some employee.
- each dependent entity is uniquely identified by taking pname in conjunction with the policyid of a policy entity

Solution can be:

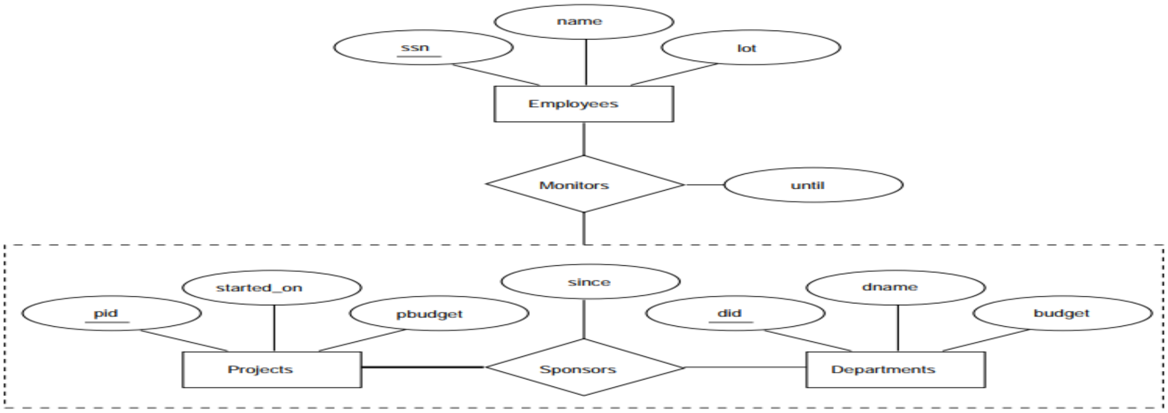
- The first requirement suggests that we impose a key constraint on Policies with respect to Covers, but this constraint has the unintended side effect that a policy can cover only one dependent.
- The second requirement suggests that we impose a total participation constraint on Policies. This solution is acceptable if each policy covers at least one dependent.
- The third requirement forces us to introduce an identifying relationship that is binary

The best way to model this situation is to use two binary relationships, as shown in Figure

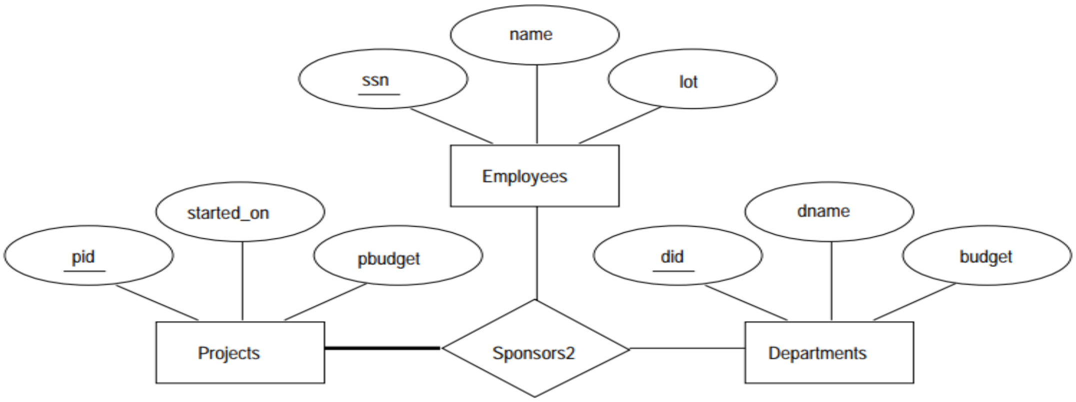


4. Aggregation v/s Ternary relationship

- The choice between using aggregation or ternary relationship is mainly determined by existence of a relationship that relates relationship set to entity set.
- The choice may also be guided by certain integrity constraints that we want to express.



According to the ER diagram shown in Figure above, a project can be sponsored by any number of departments, a department can sponsor one or more projects, and each sponsorship is monitored by one or more employees. If we don't need to record the '**until'** attribute of Monitors, then we might reasonably use a ternary relationship as shown below without the need for aggregation.



Consider the constraint that each sponsorship (of a project by a department) be mon itored by at most one employee. We cannot express this constraint in terms of the Sponsors2 relationship set. On the other hand, we can easily express the constraint by drawing an arrow from the aggregated relationship Sponsors to the relationship Monitors in the aggregation ER diagram.