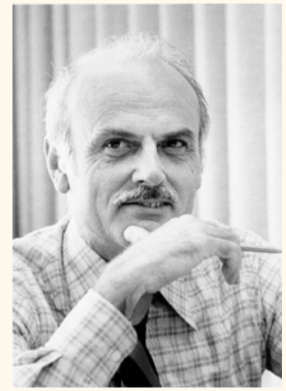
The Entity-Relationship Model

Chapter 2

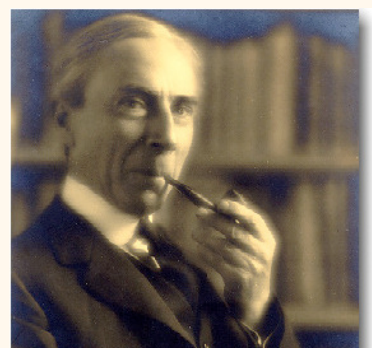
Relational Databases: History

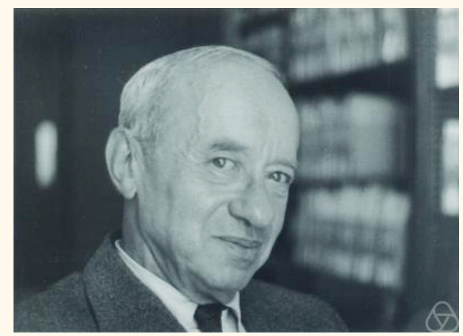
- 1970s: Computers are spreading. Many organizations use them to store their data.
- Ad hoc formats
  - ⇨ hard to build general data management systems.
  - ⇨ lots of duplicated effort.
- The Standardization Dilemma:
  - Too restrictive: doesn’t fit users’ needs.
  - Too loose: back to ad-hoc solutions.

The Relational Format



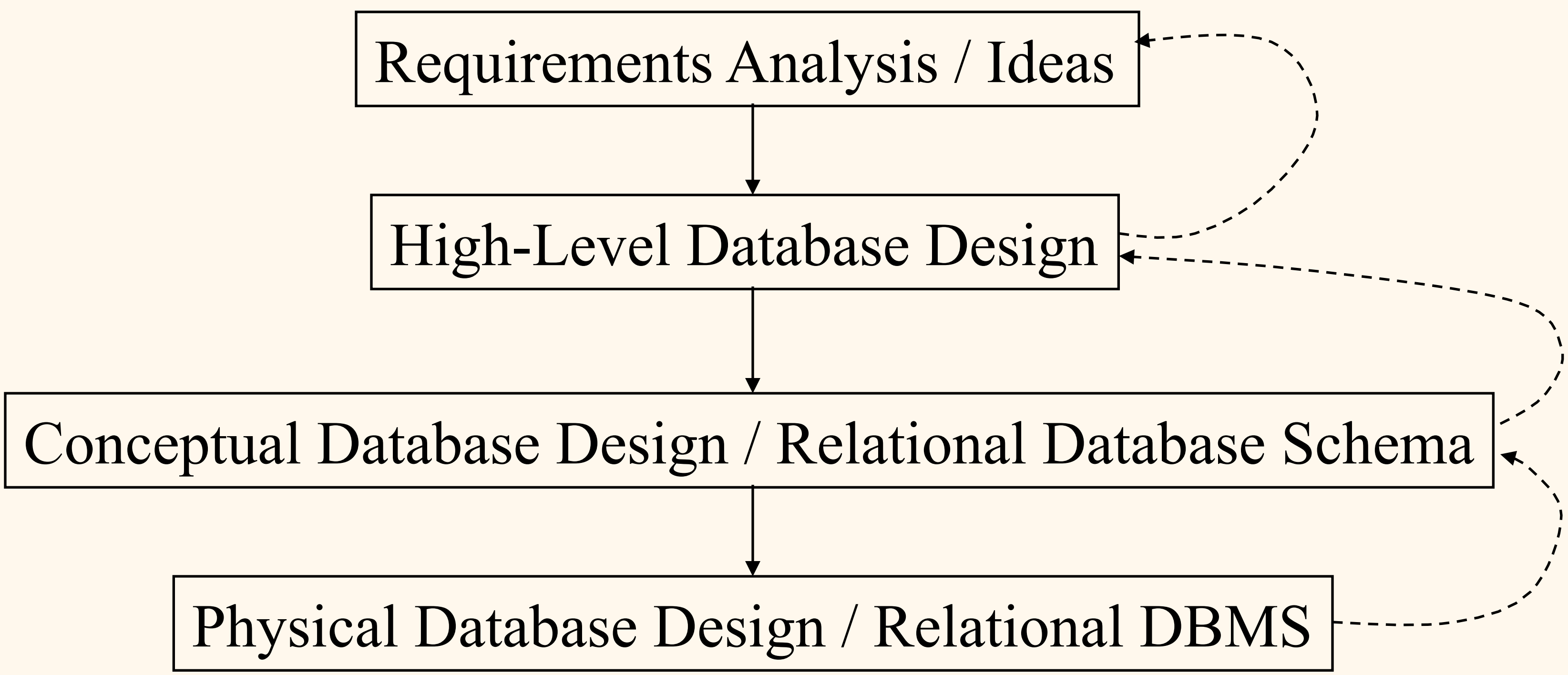
- Codd (IBM Research 1970)
- The fundamental question: *What kinds of information do users need to represent?*
- Answered by 1^{st}-order predicate logic!
  (Russell, Tarski).



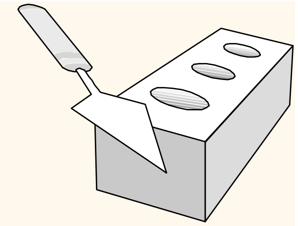


- The world consists of
  - Individuals/entities.
  - Relationships/links among them.

Overview of Database Development



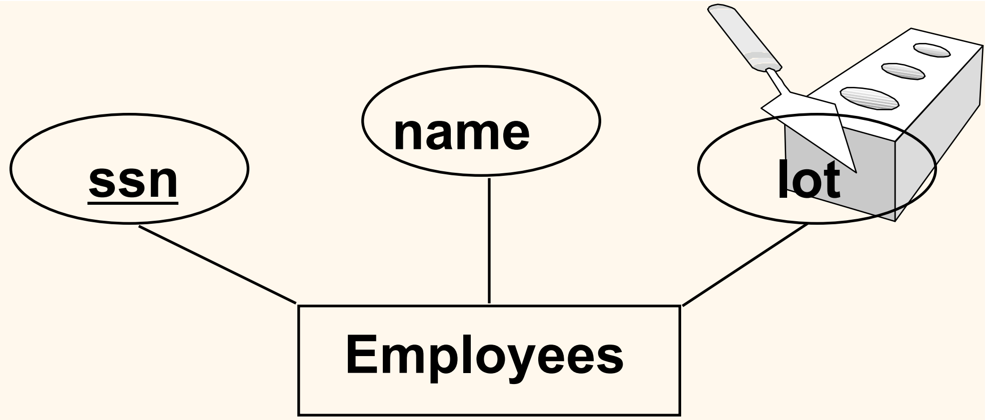
→ Similar to software development



Overview of Database Design

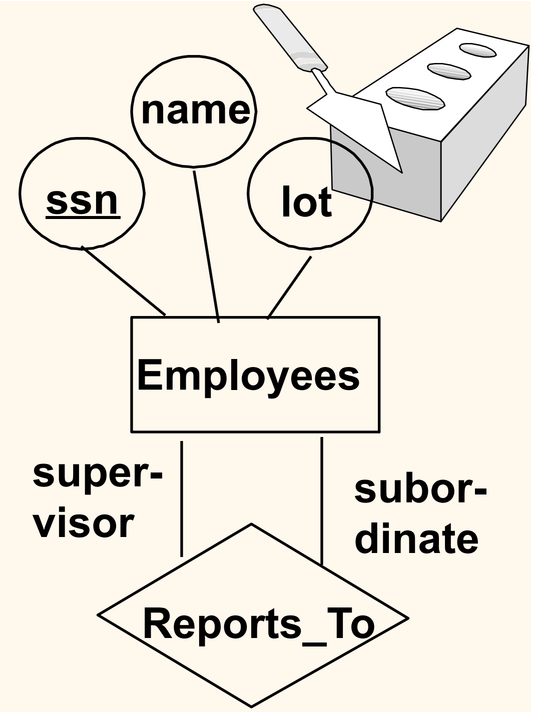
- ❖ *Conceptual design*: (*ER Model is used at this stage.)*
  - What are the *entities* and *relationships* in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?
  - A database `schema' in the ER Model can be represented pictorially (*ER diagrams*).
  - Can map an ER diagram into a relational schema.
  - Can also Unified Modelling Language (UML).
    - ✧ Pictorial Versions of 1st-order logic.

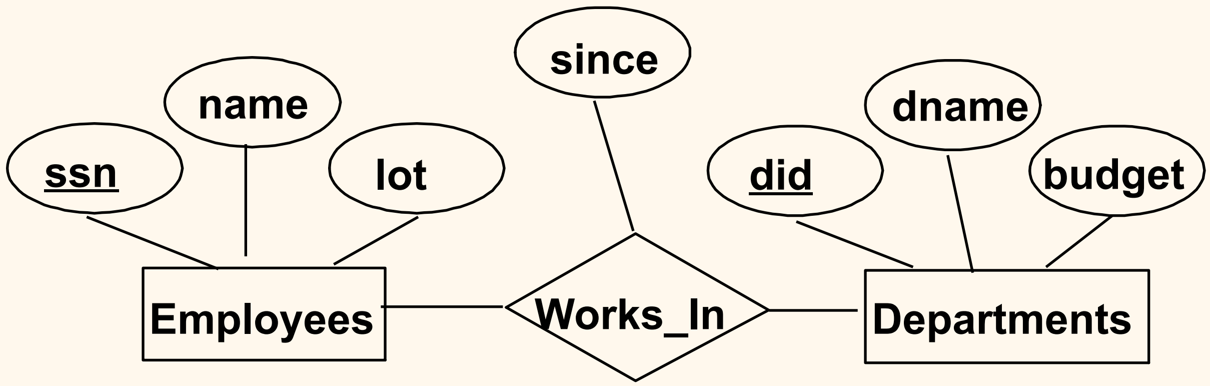
ER Model Basics



- *Entity:* Real-world object distinguishable from other objects. An entity is described (in DB) using a set of *attributes*.
- *Entity Set*: A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. (Until we consider ISA hierarchies, anyway!)
  - Each entity set has a *key*.
  - Each attribute has a *domain*.

ER Model Basics (Contd.)





- *Relationship*: Association among two or more entities. E.g., Attishoo works in Pharmacy department.
- *Relationship Set*: Collection of similar relationships.
  - An n-ary relationship set R relates n entity sets $E_1 \ldots E_n$; each relationship in R involves entities $e_1, \ldots, e_n$.
    - Same entity set could participate in different relationship sets, or in different "roles" in same set.

Cartesian or Cross-Products

- A **tuple** $\langle a_1,a_2,\ldots,a_n \rangle$ is just a list with n elements in order.
- A binary tuple <a,b> is called an **ordered pair**.
- Given two sets A,B, we can form a new set A x B containing all ordered pairs <a,b> such that a is a member of A, b is a member of B.
- In set notation: A x B = {<a,b> | a in A, b in B}.
- Example: {1,2,3} x {x,y} = {<1,x>,<1,y>,<2,x>,<2,y>,<3,x>,<3,y>}

Exercise

- Let $A = \{1,2,3\}$, $B = \{x,y\}$.
- Compute $B \times A$.
- Compute $A \times A$.

N-fold cross products

- Given n sets A1, A2, …, A_n, the cross product A1 x A2 x A3 …x A_n is a new set defined by A1 x A2 x A3 …x A_n = {<a1, a2, a3, …, a_n>: a1 in A1, a2 in A2, …, a_n in A_n}.
- Example: {1,2,3} x {x,y} x {1,2,3} has as members <1,x,1> and <1,y,3>.

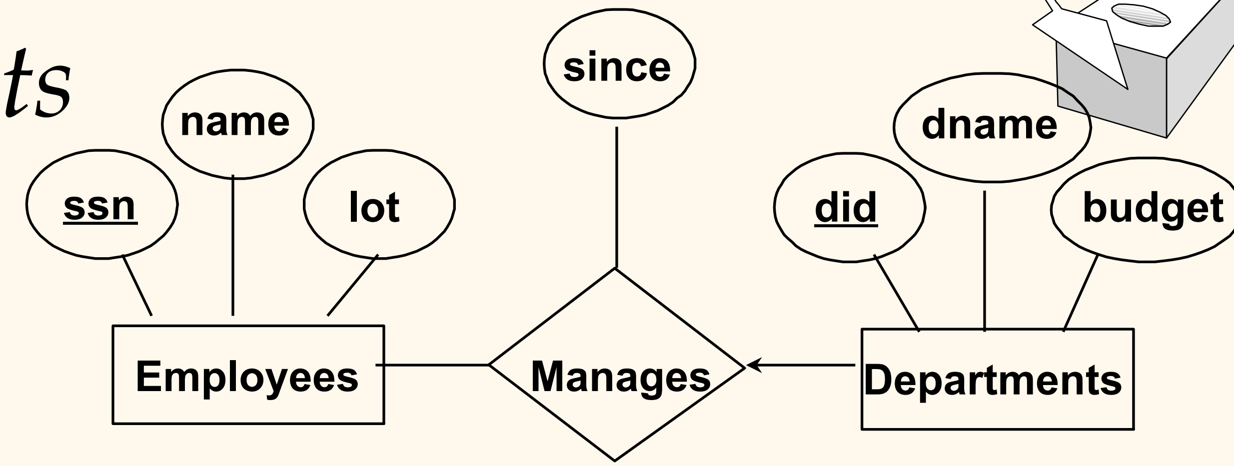
Exercise

- Let A = {1,2,3}, B = {x,y}.
- Compute B x A x B.
- Compute B x B x B.
- If a set C has *n* members, how many are there in C x C? How many in C x C x C? In general, how many in C x C x …x C where we take *k* cross-products?

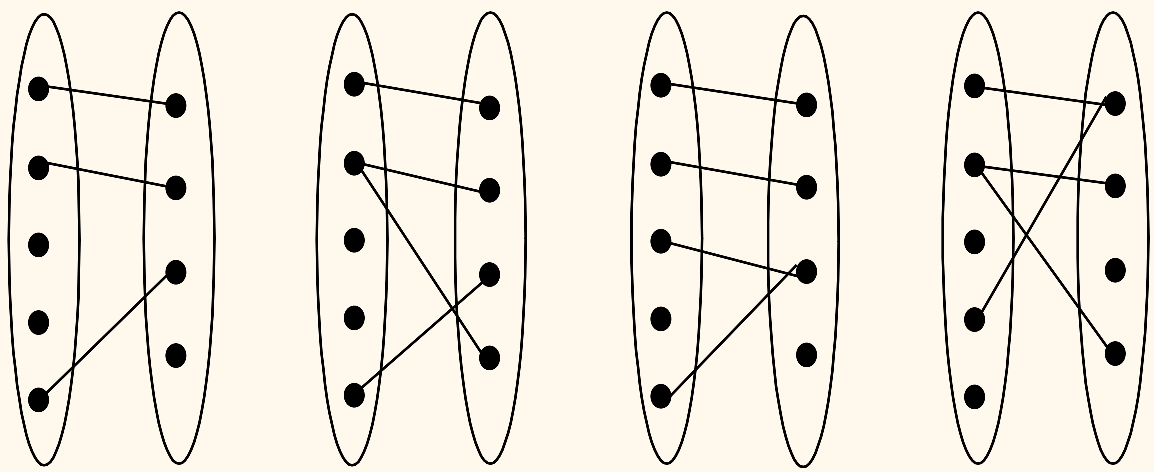
A Formal Treatment of Relation: The Cross Product

- Let E1, E2, E3 be three entity sets.
- A relationship among E1, E2, E3 is a tuple in E1 x E2 x E3.
- A relationship set, or relation, is a set of relationships. So if R is a relation among E1, E2, E3, then R is a subset of E1 x E2 x E3.

Key Constraints



- Consider Works_In: An employee can work in many departments; a dept can have many employees.
- In contrast, each dept has at most one manager, according to the *key constraint* on Manages.



1-to-1 1-to Many Many-to-1 Many-to-Many

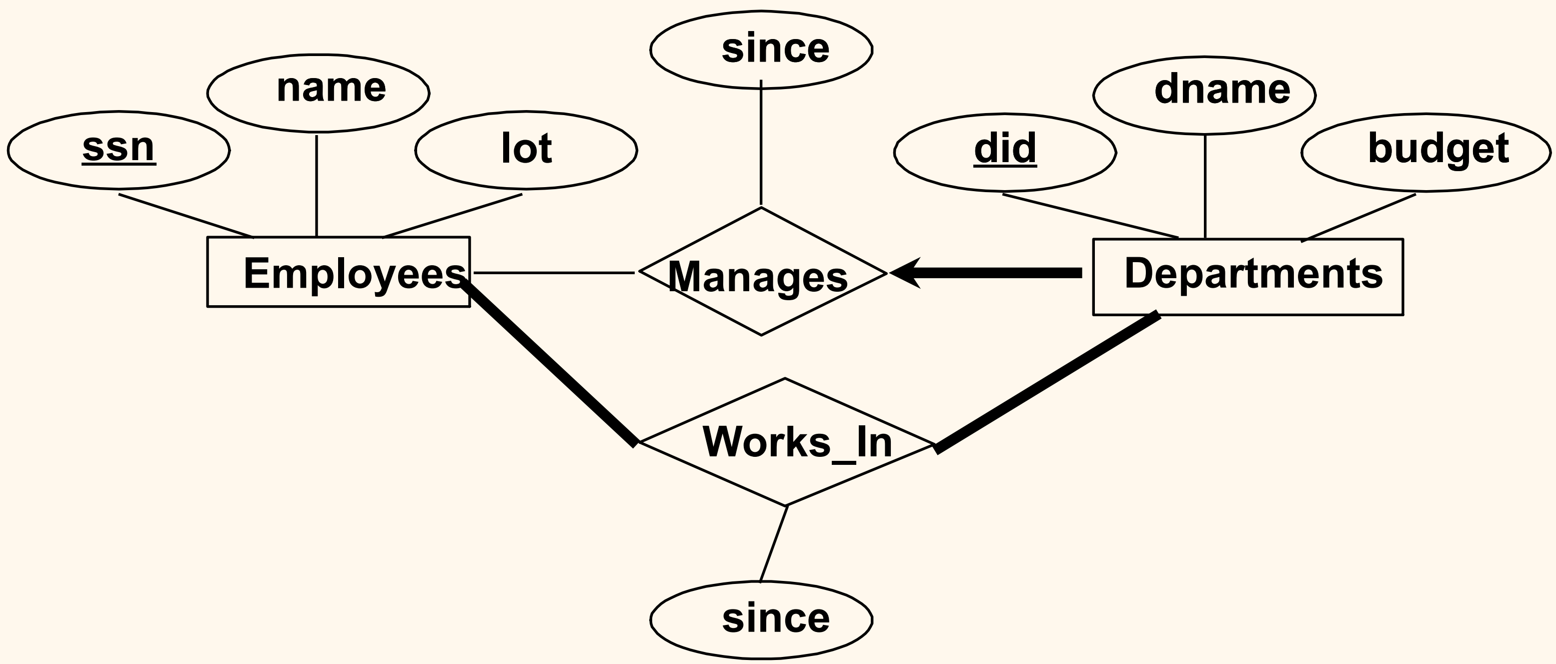
Exercise 2.2

A university database contains information about professors (identified by SSN) and courses (identified by courseid). Professors teach courses; each of the following situations concerns the Teaches relationship set. For each diagram, draw an ER diagram that describes it (assuming no further constraints hold).

1. Professors can teach the same course in several semesters, and each offering must be recorded.
2. Professors can teach the same course in several semesters, and only the most recent such offering needs to be recorded.

Participation Constraints

- Does every department have a manager?
  - If so, this is a *participation constraint*: the participation of Departments in Manages is said to be *total* (vs. *partial*).
    - Every *did* value in Departments table must appear in a tuple of the Manages relation.



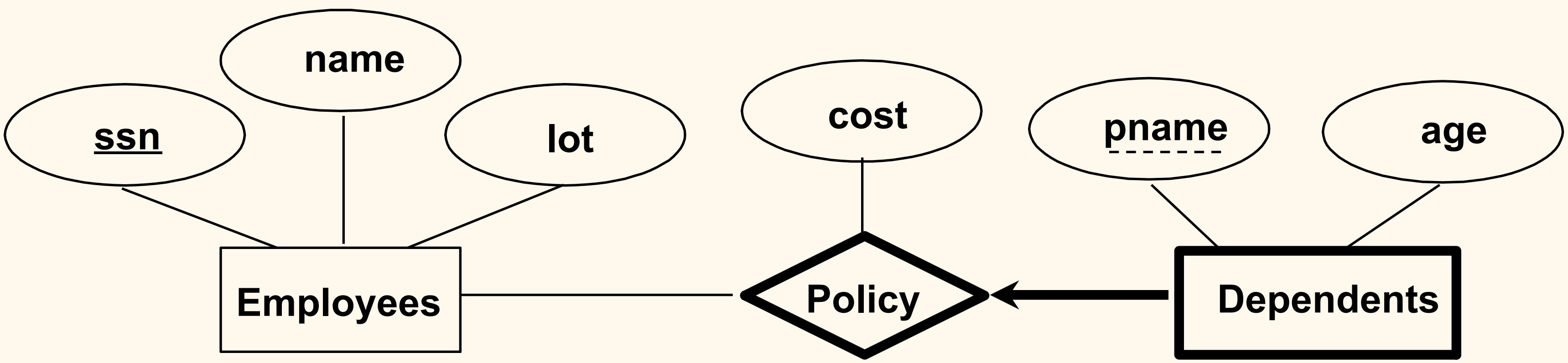
Exercise 2.2 ctd.

Same scenario as before, where we need only the current semester. Draw E-R diagrams for the following constraints.

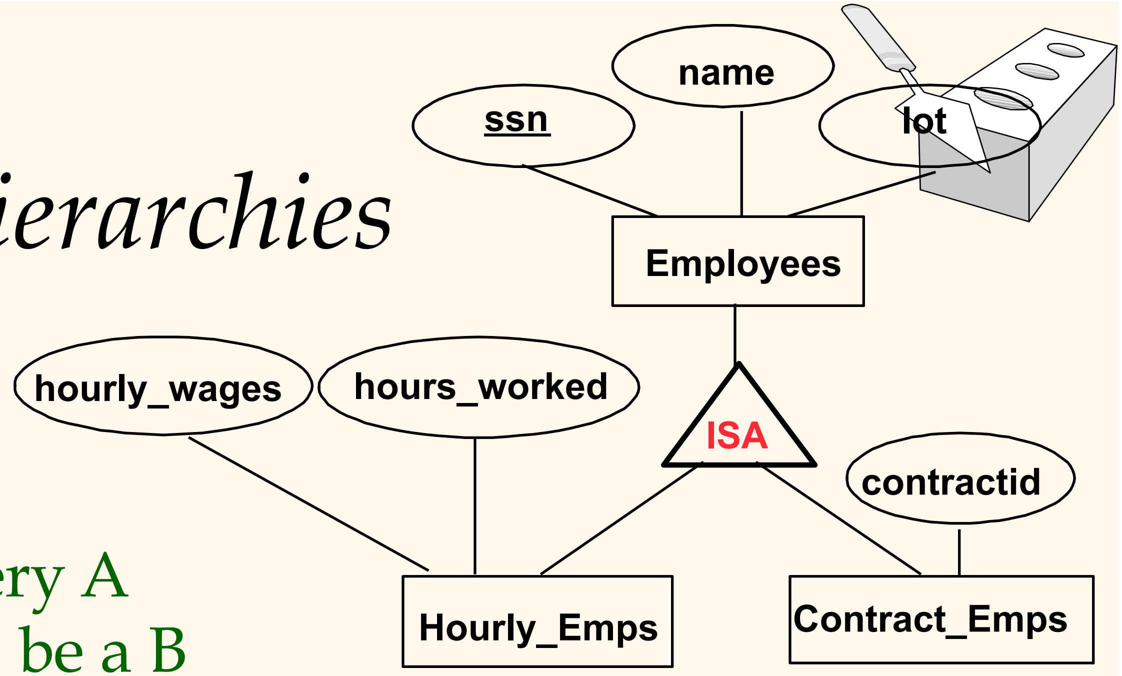
3. Every professor must teach some course.
4. Every professor teaches exactly one course.
5. Every professor teaches exactly one course, and every course must be taught by some professor.

Weak Entities

- A *weak entity* can be identified uniquely only by considering the primary key of 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.



ISA (`is a') 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.

Conceptual Design Using the ER Model

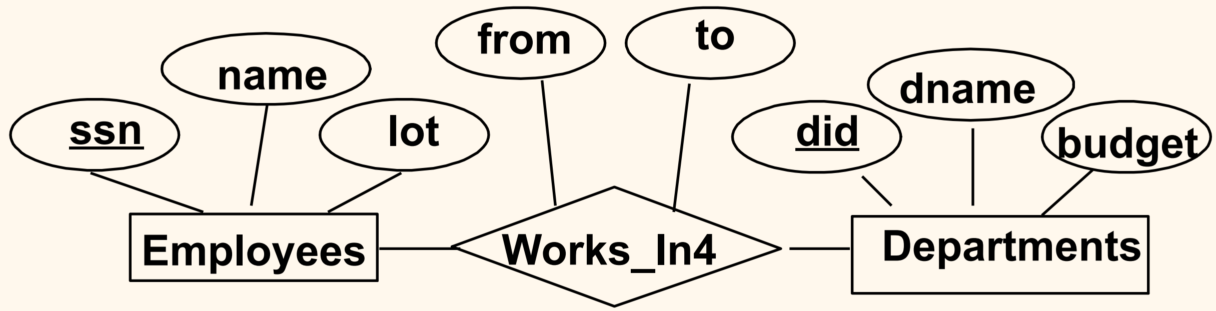
- Design choices:
  - Should a concept be modeled as an entity or an attribute? (e.g., address)
  - Should a concept be modeled as an entity or a relationship? (e.g., address)
  - Identifying relationships: Binary or ternary? Aggregation? (see text)

Entity vs. Attribute

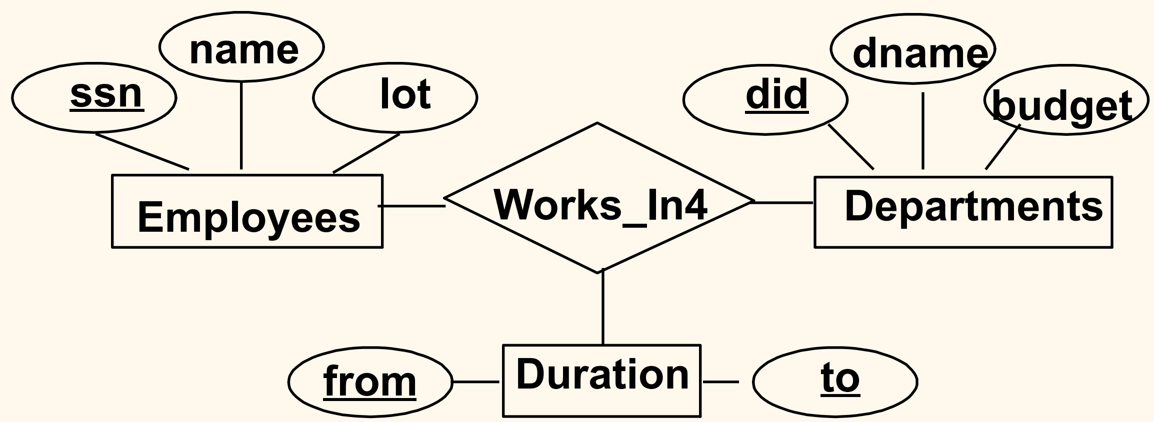
- Should *address* be an attribute of Employees or an entity (connected to Employees by a relationship)?
- Depends upon the use we want to make of address information, and the semantics of the data:
  - If we have several addresses per employee, *address* must be an entity (since attributes cannot be set-valued).
  - If the structure (city, street, etc.) is important, e.g., we want to retrieve employees in a given city, *address* must be modeled as an entity (since attribute values are atomic).

Entity vs. Attribute (Contd.)

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

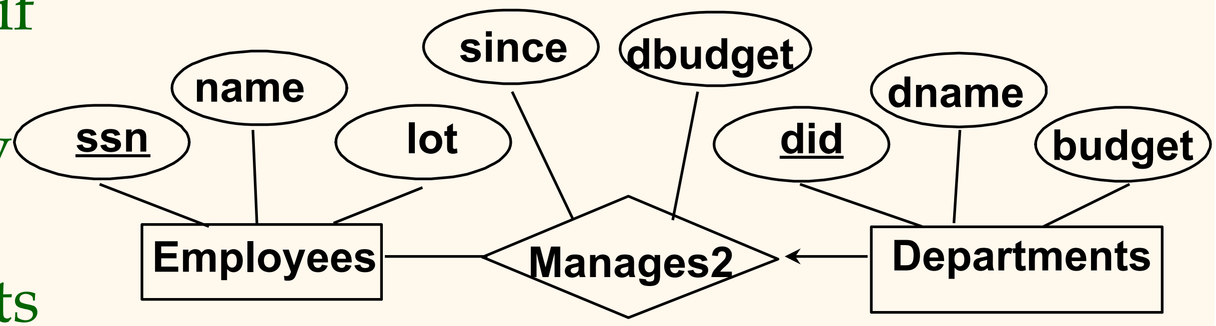


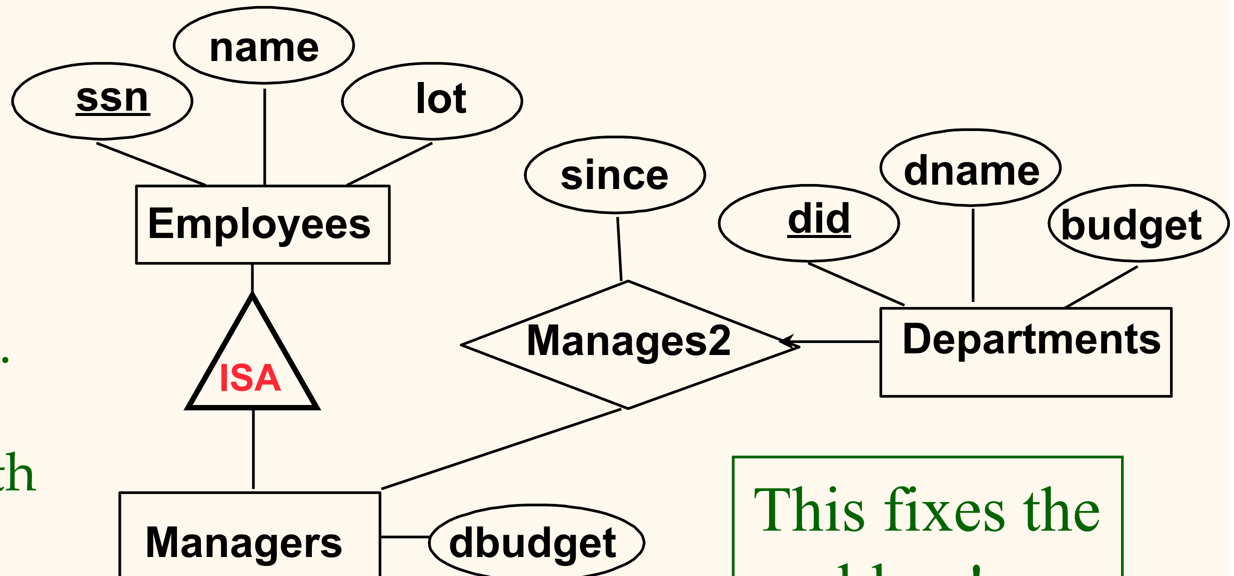
- Similar to the problem of wanting to record several addresses for an employee: We want to record *several values of the descriptive attributes for each instance of this relationship.* Accomplished by introducing new entity set, Duration.



Entity vs. Relationship

- First ER diagram OK if a manager gets a separate discretionary budget for each dept.
- What if a manager gets a discretionary budget that covers *all* managed depts?
  - Redundancy: *dbudget* stored for each dept managed by manager.
  - Misleading: Suggests *dbudget* associated with department-mgr combination.

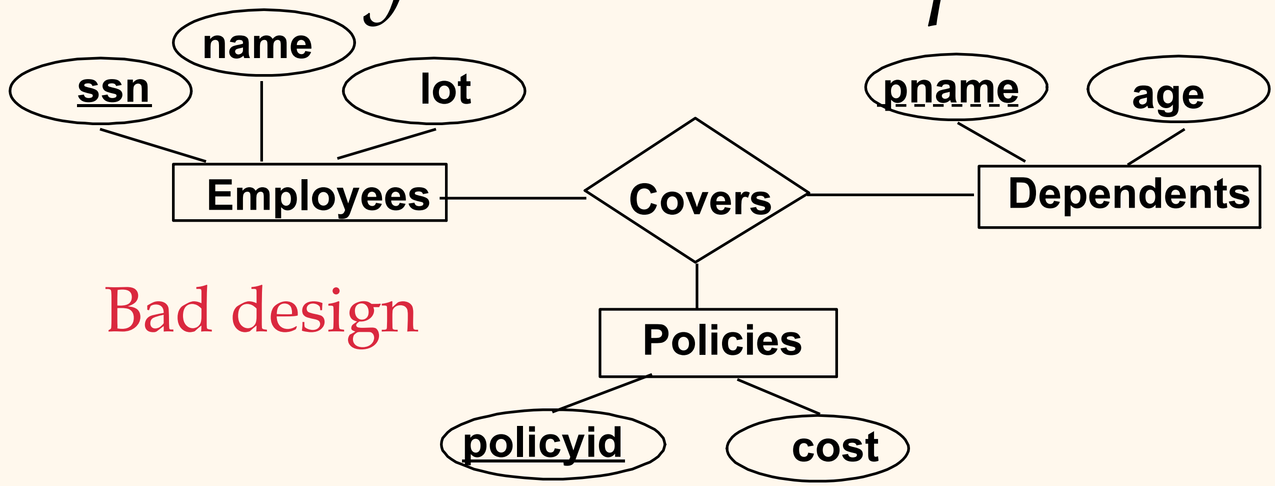




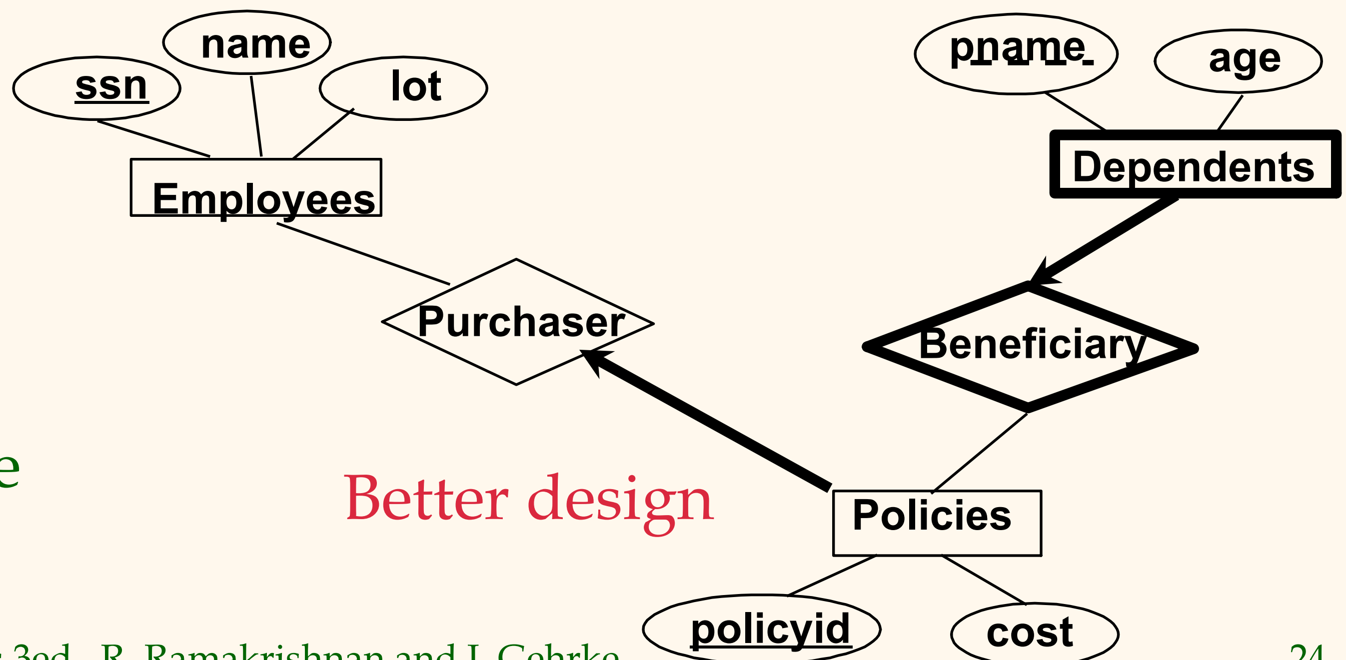
This fixes the problem!

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.
- What are the additional constraints in the 2nd diagram?



Bad design



Better design

Binary vs. Ternary Relationships (Contd.)

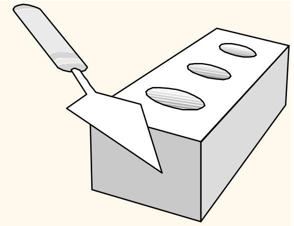
- Previous example illustrated a case when two binary relationships were better than one ternary relationship.
- An example in the other direction: a ternary relation Contracts relates entity sets Parts, Departments and Suppliers, and has descriptive attribute *qty*. No combination of binary relationships is an adequate substitute:
  - S "can-supply" P, D "needs" P, and D "deals-with" S does not imply that D has agreed to buy P from S.
  - How do we record *qty*?

Summary of Conceptual Design

- *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*.
- Note: There are many variations on ER model.

Summary of ER (Contd.)

- Several kinds of integrity constraints can be expressed in the ER model: *key constraints*, *participation constraints*, and *overlap/covering constraints* for ISA hierarchies.
  - Some constraints (notably, *functional dependencies*) cannot be expressed in the ER model. (e.g., z = x + y)
  - Constraints play an important role in determining the best database design for an enterprise.



Summary of ER (Contd.)

- ER design is *subjective*. There are often many ways to model a given scenario! Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:
  - Entity vs. attribute.
  - entity vs. relationship
  - binary or n-ary relationship
  - ISA hierarchies.
- Ensuring good database design: resulting relational schema should be analyzed and refined further. See Ch. 19.