

# The Entity-Relationship Model

15-415 Spring 2003: Lecture 2  
R &G - Chapter 2

A relationship, I think, is like a shark, you know? It has to constantly move forward or it dies. And I think what we got on our hands is a dead shark.

Woody Allen (from Annie Hall, 1979)



## Administrivia

- Homework 0 is out today.
- This is a pass/fail homework, however, you **MUST** complete it in order to stay in the course.
- Course accounts will be handed out by Minglong Shao.



## Databases Model the Real World

- “Data Model” allows us to translate real world things into structures computers can store
- Many models: Relational, E-R, O-O, Network, Hierarchical, etc.
- **Relational**
  - Rows & Columns

– Keys & Foreign Keys to link Relations

Enrolled			Students				
sid	cid	grade	sid	name	login	age	gpa
53666	Carnatic101	C	53666	Jones	jones@cs	18	3.4
53666	Reggae203	B	53688	Smith	smith@eecs	18	3.2
53650	Topology112	A	53650	Smith	smith@math	19	3.8
53666	History105	B					



## Database Design

- Requirements Analysis
  - user needs; what must database do?
- Conceptual Design
  - high level descr (often done w/ER model)
- Logical Design
  - translate ER into DBMS data model
- Schema Refinement
  - **consistency, normalization**
- Physical Design - indexes, disk layout
- Security Design - who accesses what

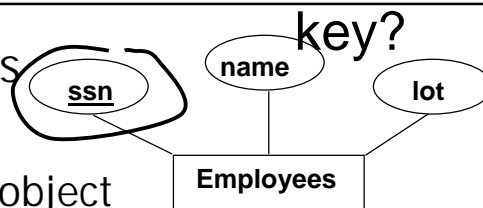


## Conceptual Design

- 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.



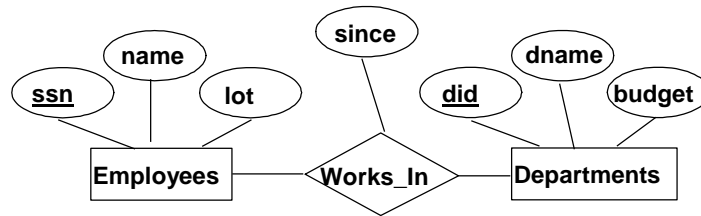
## ER Model Basics



- 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 hierarchies, anyway!)
  - Each entity set has a *key* (underlined).
  - Each attribute has a *domain*.



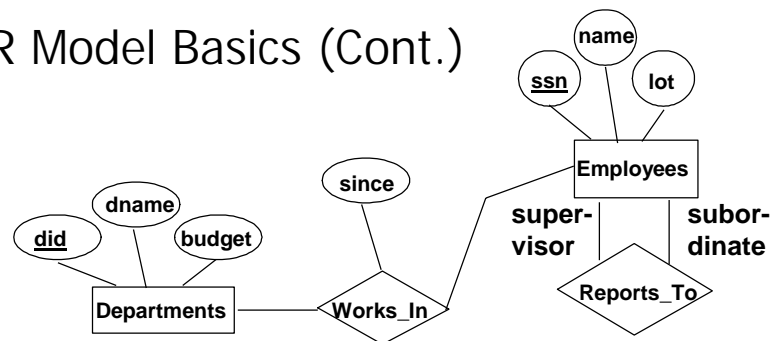
## ER Model Basics (Contd.)



- **Relationship**: Association among two or more entities. E.g., Attishoo works in Pharmacy department.
  - relationships can have their own attributes.
- **Relationship Set**: Collection of similar relationships.
  - An n-ary relationship set R relates n entity sets E1 ... En; each relationship in R involves entities e1 E1, ..., en En



## ER Model Basics (Cont.)



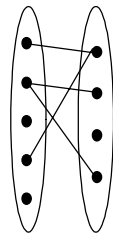
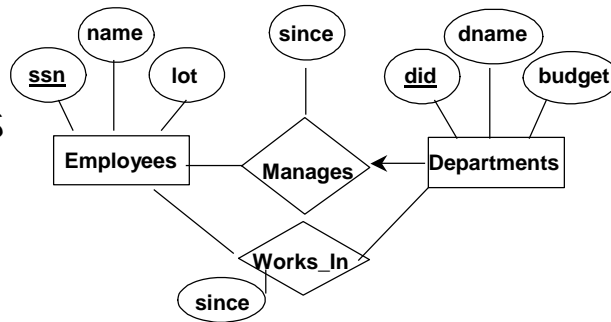
- Same entity set can participate in different relationship sets, or in different “roles” in the same set.



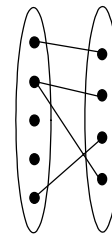
## Key Constraints

An employee can work in many departments; a dept can have many employees.

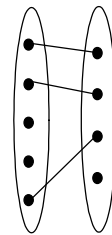
In contrast, each dept has at most one manager, according to the key constraint on Manages.



Many-to-Many



1-to Many

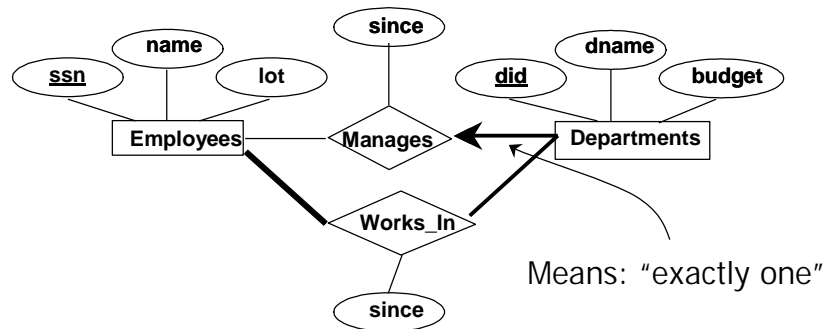


1-to-1



## Participation Constraints

- Does every employee work in a department?
- If so, this is a participation constraint: the participation of Departments in Manages is said to be *total* (vs. *partial*).
- Basically means "at least one"

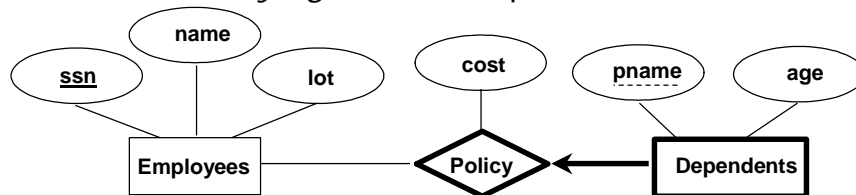




## Weak Entities

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.



Weak entities have only a "partial key" (dashed underline)

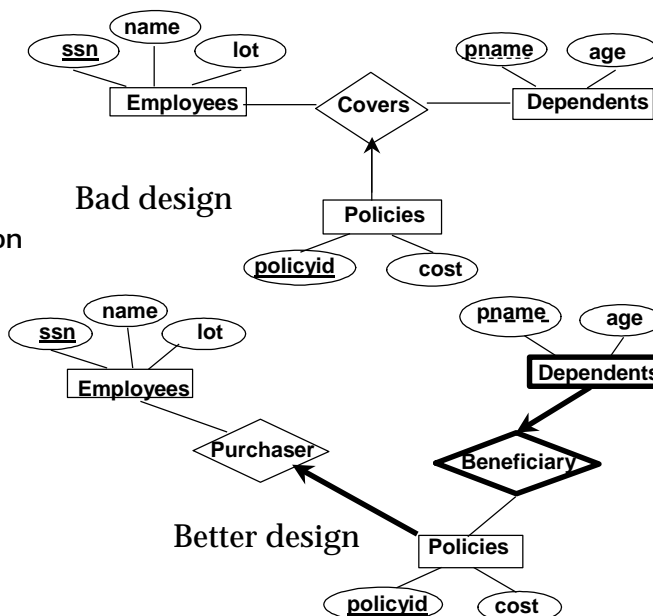


## Binary vs. Ternary Relationships

If each policy is owned by just 1 employee:

Key constraint on Policies would mean policy can only cover 1 dependent!

•What are the additional constraints in the 2nd diagram?



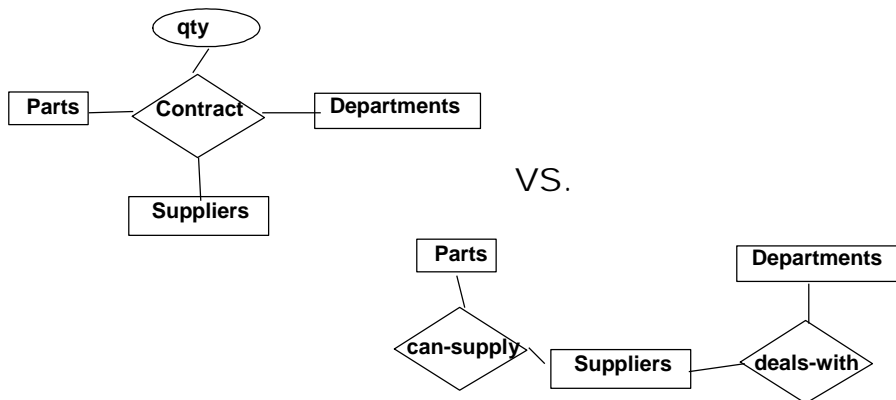


## Binary vs. Ternary Relationships (Contd.)

- 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.



## Binary vs. Ternary Relationships (Contd.)



- 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 so far

- **Entities and Entity Set (boxes)**
- **Relationships and Relationship sets (diamonds)**
  - binary
  - n-ary
- **Key constraints (1-1, M-1, M-M, arrows on 1 side)**
- **Participation constraints (bold for Total)**
- **Weak entities - require strong entity for key**



## Now you try it

### Courses database:

- **Courses, Students, Teachers**
- **Courses have ids, titles, credits, ...**
- **Courses have multiple sections that have time/rm and exactly one teacher**
- **Must track students course schedule and transcript including grades, semester taken, etc.**
- **Must track which classes a professor has taught**
- **Database should work over multiple semesters**





Next...

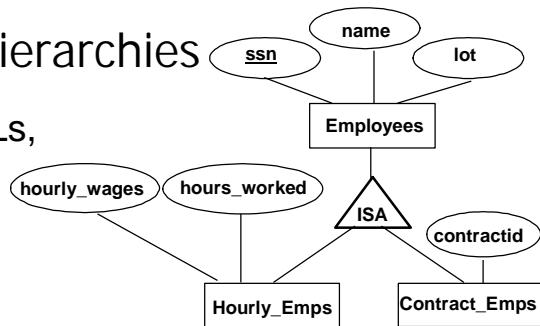
- ... a couple more “advanced” concepts...



## 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.



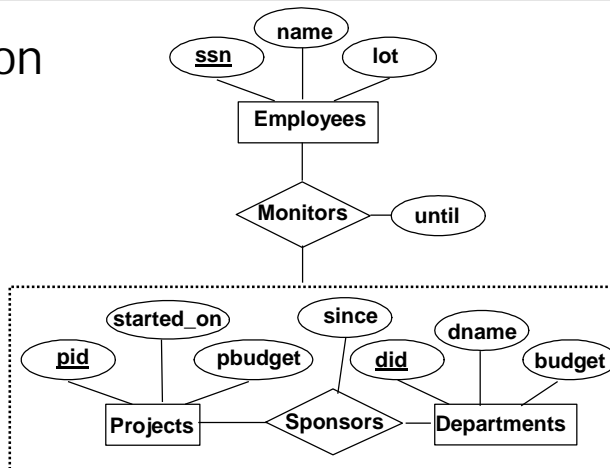
- *Overlap constraints*: Can Joe be an Hourly\_Emps as well as a Contract\_Emps entity? (*Allowed/disallowed*)
- *Covering constraints*: Does every Employees entity also have to be an Hourly\_Emps or a Contract\_Emps entity? (*Yes/no*)
- Reasons for using ISA:
  - To add descriptive attributes specific to a subclass. (i.e. not appropriate for all entities in the superclass)
  - To identify entities that participate in a particular relationship (i.e., not all superclass entities participate).



## Aggregation

Used to model a relationship involving a *relationship set*.

Allows us to treat a relationship set as an entity set for purposes of participation in (other) relationships.



### **Aggregation vs. ternary relationship?**

- ❖ **Monitors** is a distinct relationship, with a descriptive attribute.
- ❖ Also, can say that each sponsorship is monitored by at most one employee.



## Conceptual Design Using the ER Model

- **Design choices:**
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary? Aggregation?
- **Constraints in the ER Model:**
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.



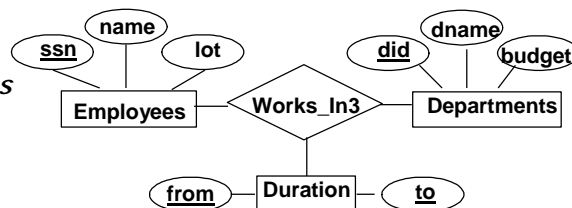
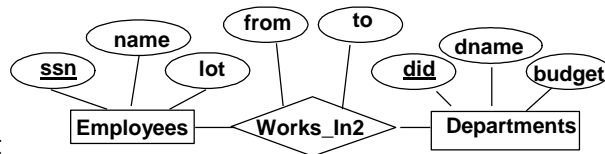
## Entity vs. Attribute

- Should *address* be an attribute of Employees or an entity (related to Employees)?
- Depends upon how we want to use 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, *address* must be modeled as an entity (since attribute values are atomic).



## Entity vs. Attribute (Cont.)

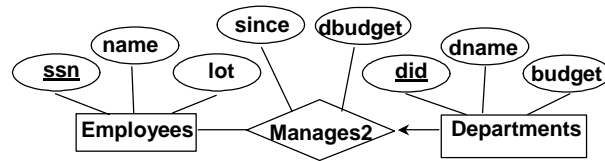
- Works\_In2 does not allow an employee to work in a department for two or more periods.
- Similar to the problem of wanting to record several addresses for an employee: we want to record *several values of the descriptive attributes for each instance of this relationship*.





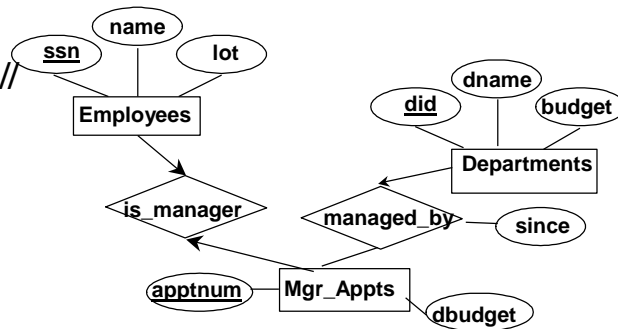
## Entity vs. Relationship

OK as long as a manager gets a separate discretionary budget (*dbudget*) for each dept.



What if manager's *dbudget* covers *all* managed depts?

(can repeat value, but such redundancy is problematic)



## 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.
  - Note: There are many variations on ER model.
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships).
- Some additional constructs: *weak entities*, *ISA hierarchies*, and *aggregation*.



## Summary of ER (Cont.)

- Several kinds of integrity constraints:
  - *key constraints*
  - *participation constraints*
  - *overlap/covering* for ISA hierarchies.
- Some *foreign key constraints* are also implicit in the definition of a relationship set.
- Many other constraints (notably, *functional dependencies*) cannot be expressed.
- Constraints play an important role in determining the best database design for an enterprise.



## Summary of ER (Cont.)

- 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, whether or not to use ISA hierarchies, aggregation.
- Ensuring good database design: resulting relational schema should be analyzed and refined further.
  - Functional Dependency information and normalization techniques are especially useful.