The Entity-Relationship Model

Course Instructor-
Dr. Koushlendra Kumar Singh
Assistant Professor
Department of Computer Science and Engineering
National Institute of Jamshedpur- Jharkhand

Note: Students who enrolled in this course, Please mail me for any doubt related to this content-
koushlendra.cse@nitjsr.ac.in

Review

- **Query Optimization**
  - Some resources, see slides
- **Finished discussing SQL**
  - Insert
  - Delete
  - Update
  - Null Values – Outer Joins
  - Views
  - Order By
  - Access Control
  - Integrity Constraints

Review

- **Data Modelling**
  - Relational
  - E-R
- **Storing Data**
  - File Indexes
  - Buffer Pool Management
- **Query Languages**
  - SQL
  - Relational Algebra
  - Relational Calculus
- **Query Optimization**
  - External Sorting
  - Join Algorithms
  - Query Plans, Cost Estimation

Topics to be covered

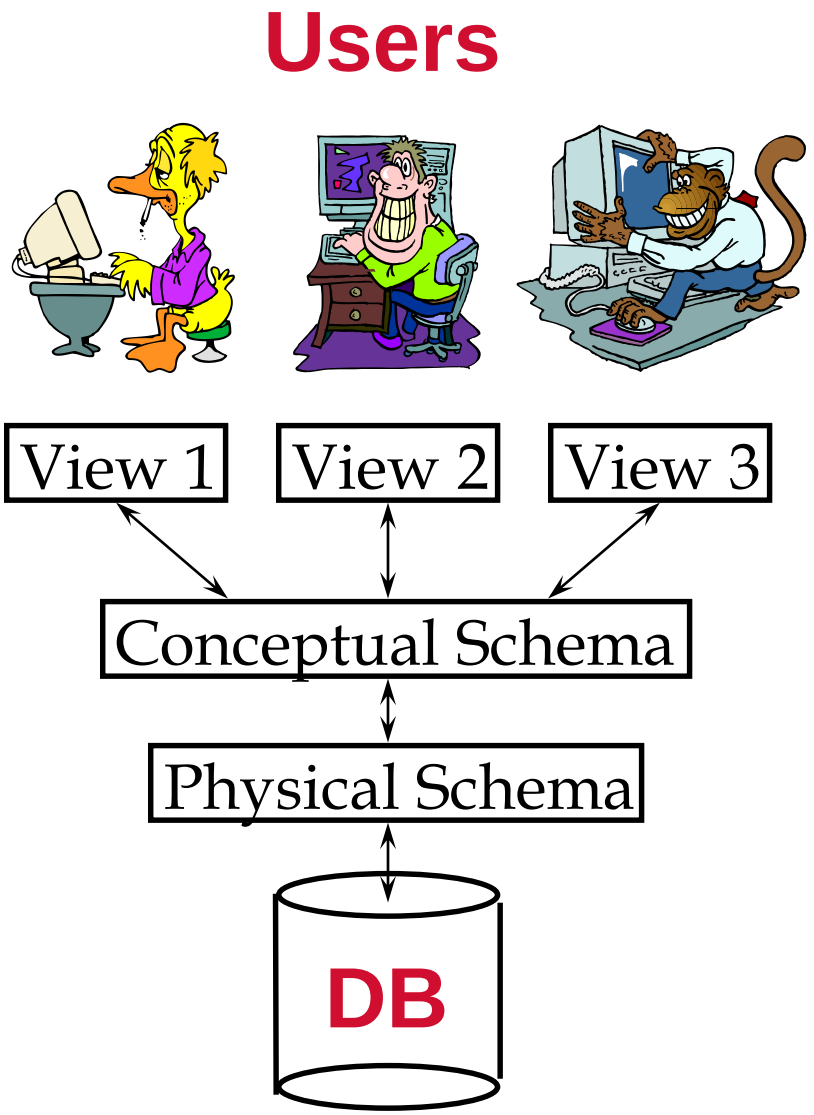
- Concepts/Constructs in ER Approach and diagram
  - Cardinality vs. Participation Constraint
  - Weak Entity Type, EX/ID Relationship Types, generalization and specialization
  - Some extensions: Aggregation, Multiple FDs Representation
- English Sentence Structure and ER Diagram o self study
- ER Construct Notation Comparison
- Database Schema Design using ER Approach
- Translation of a (Normal Form) ER Diagram to a RDB
- A Normal Form for ER Diagram

The ER Model

- **Discussed briefly in Initial Class leactures**
- **A different data model from Relational**
- **Most commonly used for database design**
- **Details of the ER Model**
- **Next, Translating ER Schemas to Relational**

Review: Levels of Abstraction

- Views describe how users see the data.
- Conceptual schema defines logical structure
- Physical schema describes the files and indexes used.
- E-R Model most often appears at the View level, with the Relation Model at the Conceptual level
- Some systems exist that use ER model as Conceptual Model



Databases Model the Real World

- "Data Model" translates 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

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

A Problem with the Relational Model

```
CREATE TABLE Enrolled
      (sid CHAR(20),
       cid CHAR(20),
       grade CHAR(2))
```

```
CREATE TABLE Students
      (sid CHAR(20),
       name CHAR(20),
       login CHAR(10),
       age INTEGER,
       gpa FLOAT)
```

With complicated schemas, it may be hard for a person to understand the structure from the data definition.

Enrolled

cid	grade	sid
Carnatic101	C	53666
Reggae203	B	53666
Topology112	A	53650
History105	B	53666

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

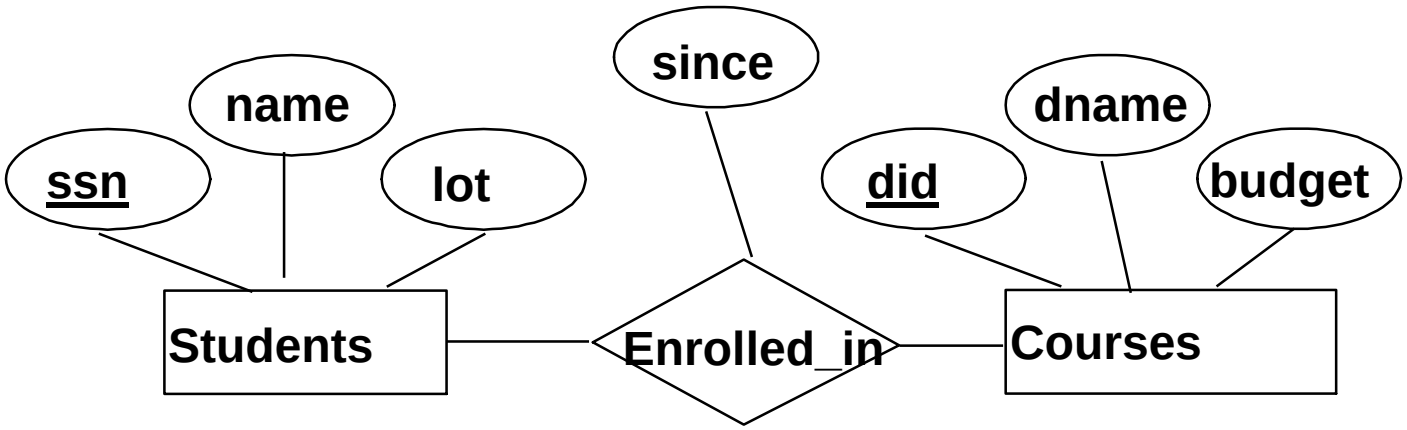
One Solution: The E-R Model

- **Instead of relations, it has:**

  Entities and Relationships

- **These are described with diagrams,**

  both structure, notation more obvious to humans



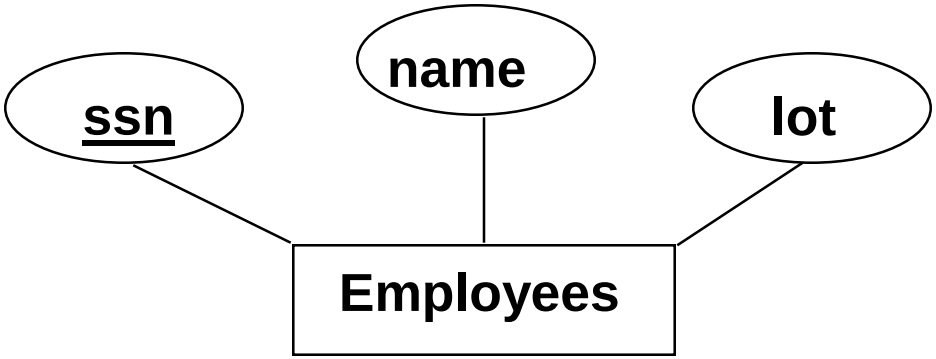
Steps in Database Design

- 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, and how

Conceptual Design

- Define enterprise *entities* and *relationships*
- What information about entities and relationships should be in 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



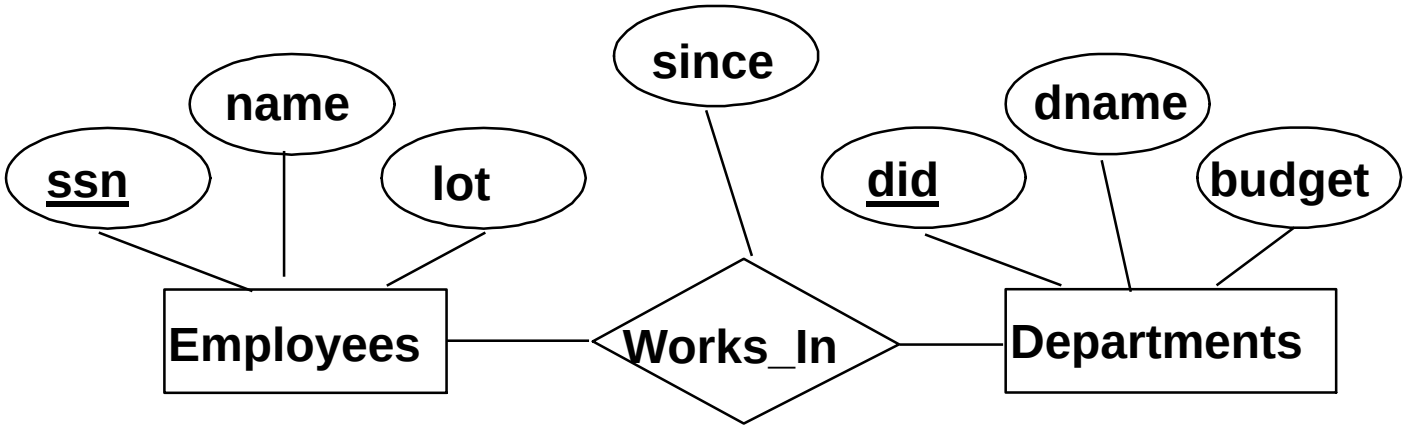
- *Entity:*

  Real-world thing, distinguishable from other objects.

  Entity described by set of *attributes*.

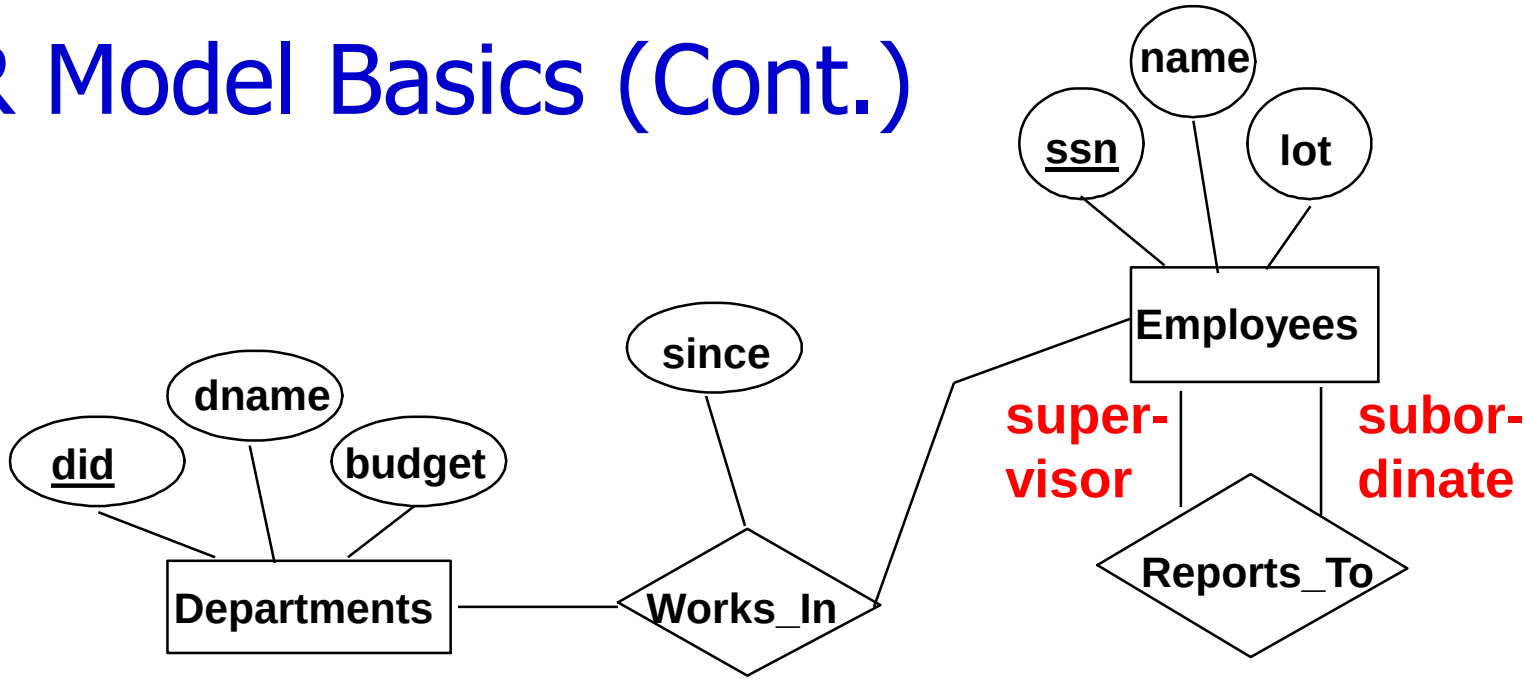
- *Entity Set*: A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. (Until we consider hierarchies, anyway!)
  - Each entity set has a *key (underlined)*.
  - Each attribute has a *domain*.

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 $E_1 \ldots E_n$; each relationship in *R* involves entities $e_1 \in E_1, \ldots, e_n \in E_n$

ER Model Basics (Cont.)

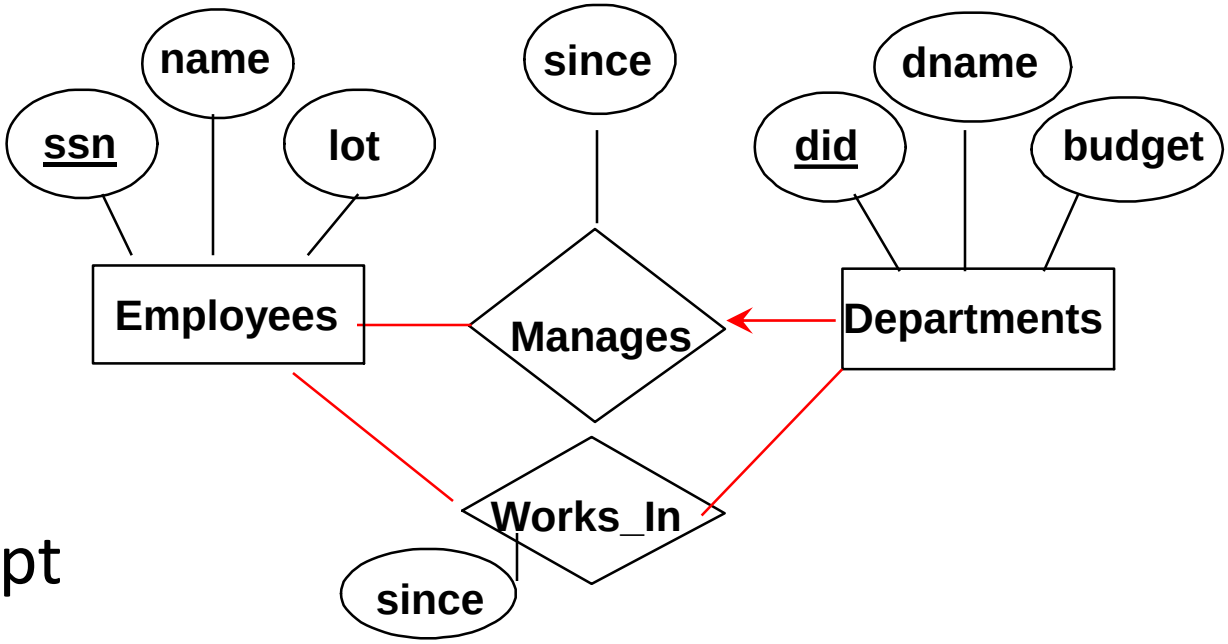


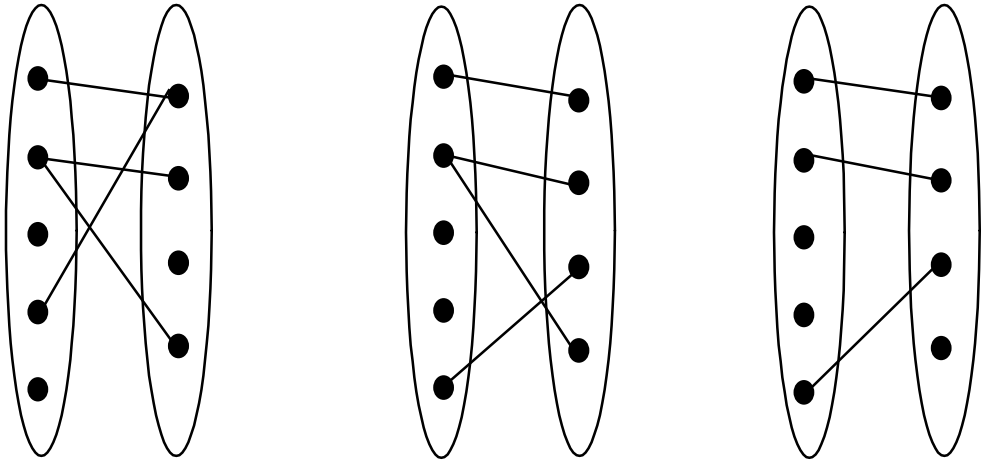
Same entity set can participate in different relationship sets, or in different "roles" in the same set.

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.





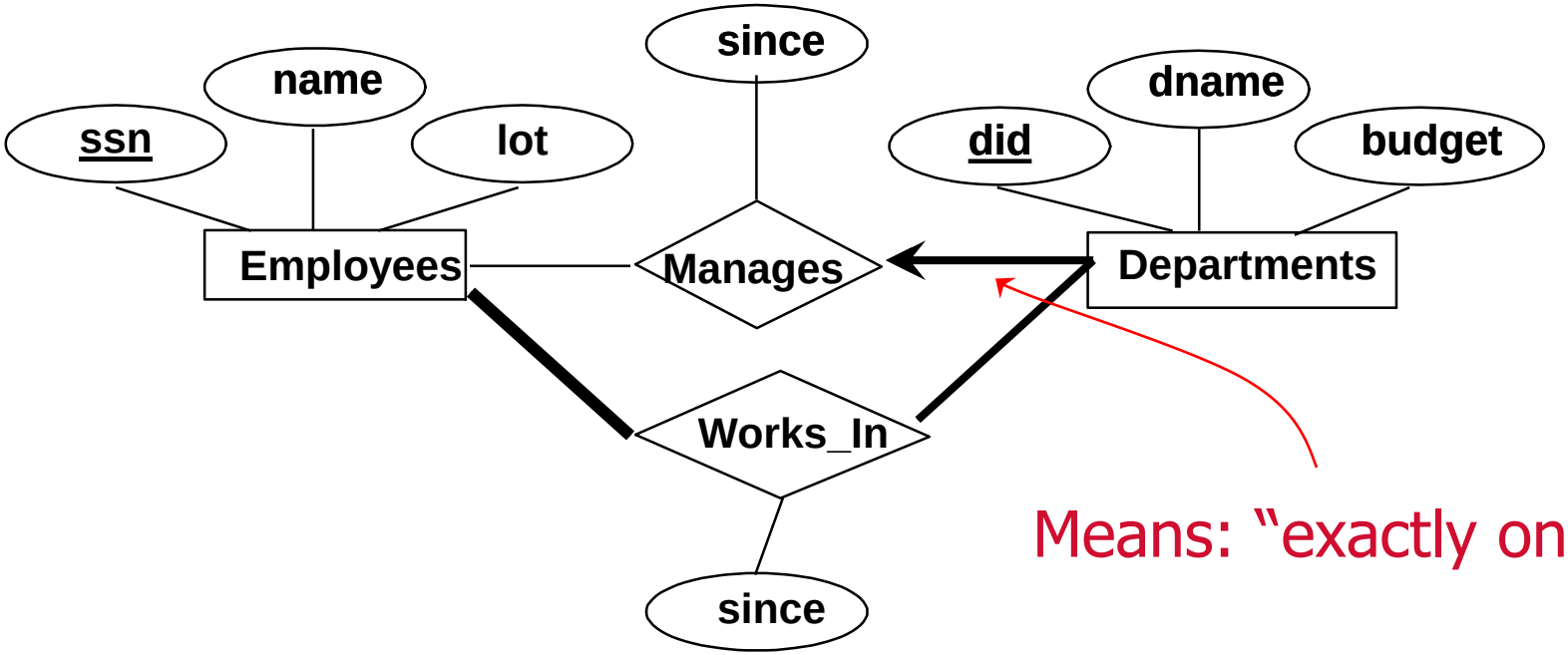
Many-to-Many

1-to Many

1-to-1

Participation Constraints

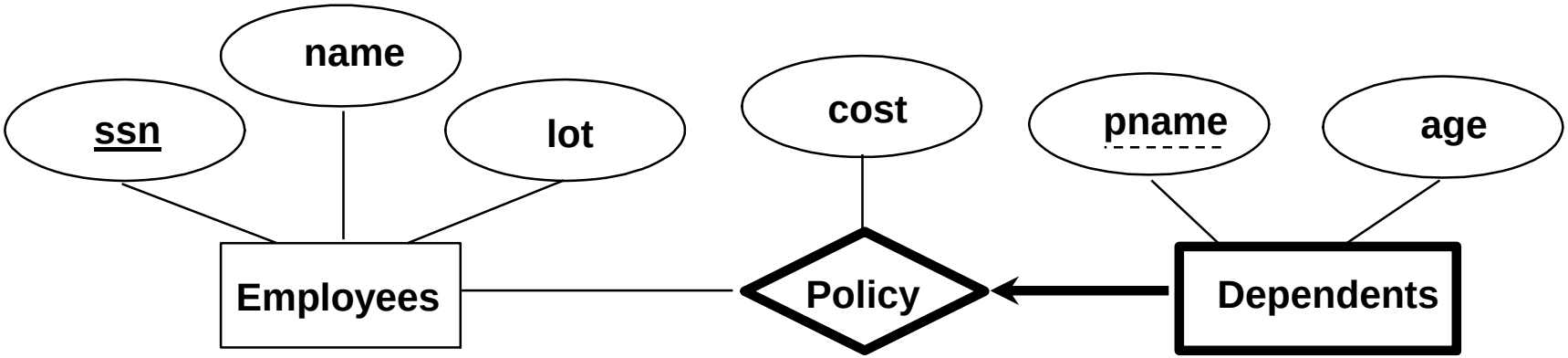
- **Does every employee work in a department?**
- **If so, this is a *participation constraint***
  - the participation of Employees in Works_In is said to be *total* (vs. *partial*)
  - What if every department has an employee working in it?
- **Basically means "at least one"**



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.

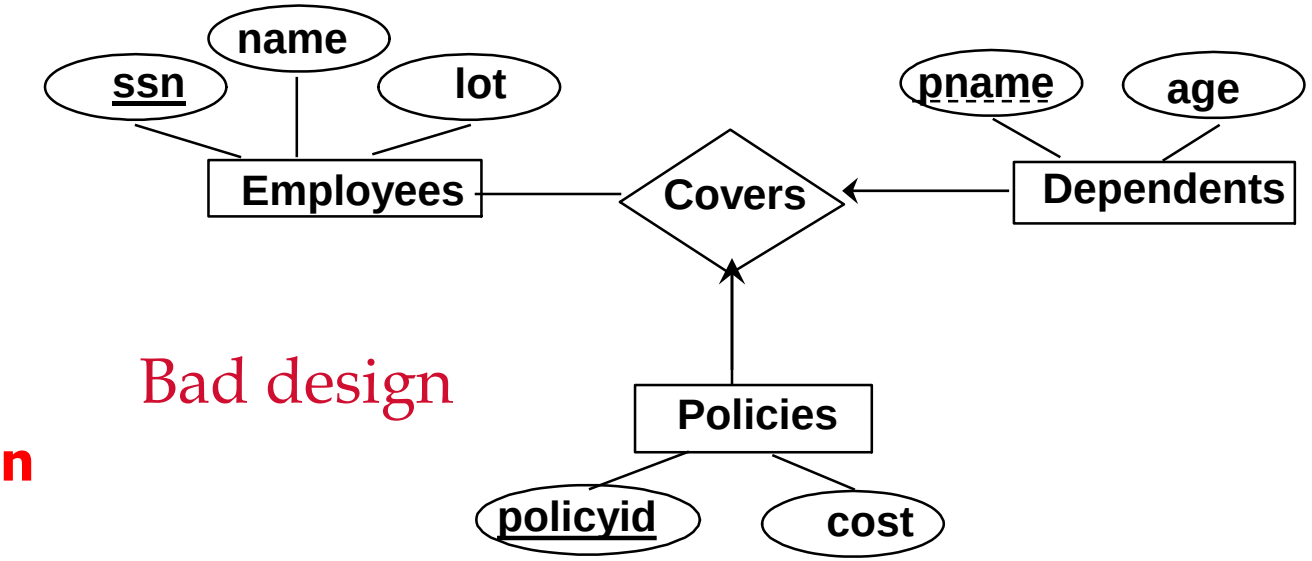


Weak entities have only a "partial key" (dashed underline)

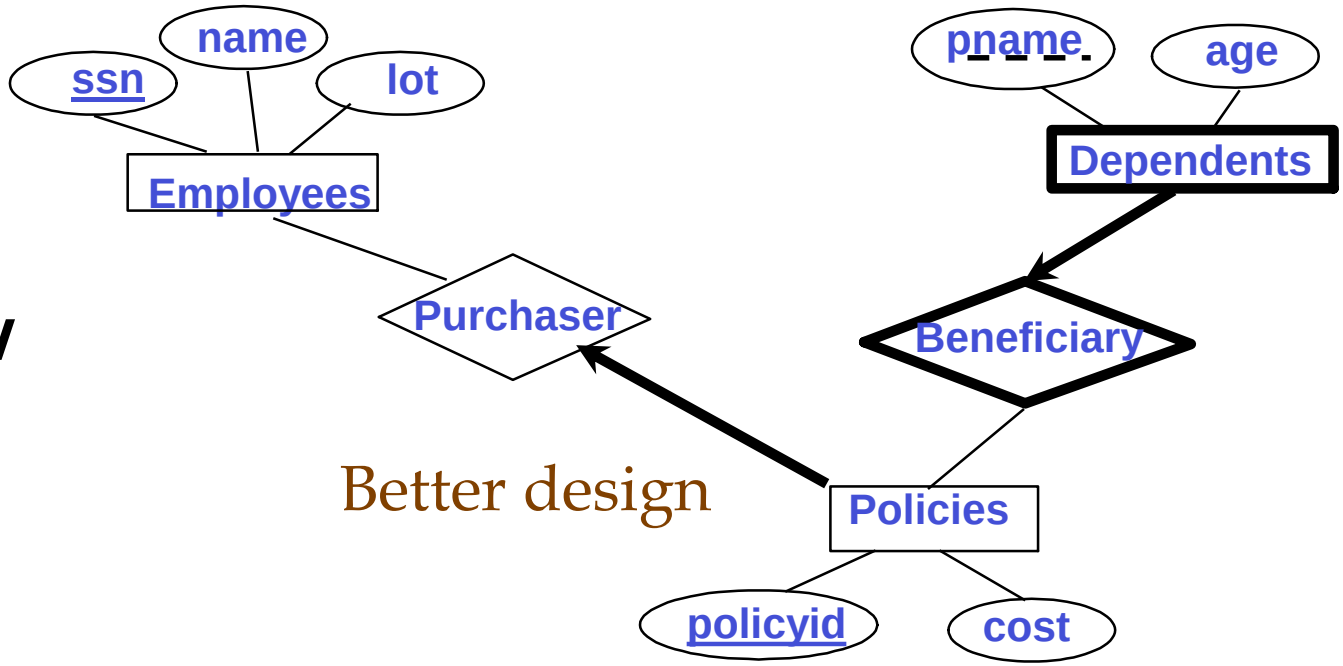
Binary vs. Ternary Relationships

If each policy is owned by just 1 employee:

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



Bad design



Better design

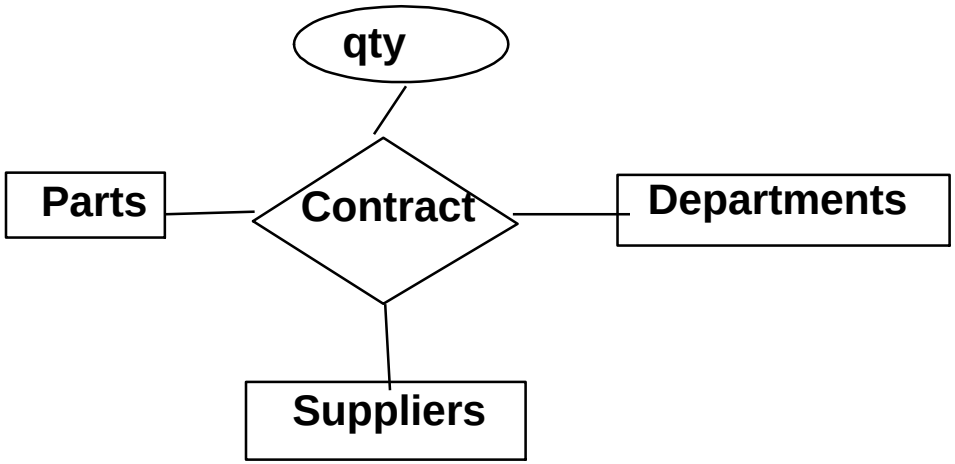
- **Think through *all* the constraints in the 2nd diagram!**

Binary vs. Ternary Relationships (Contd.)

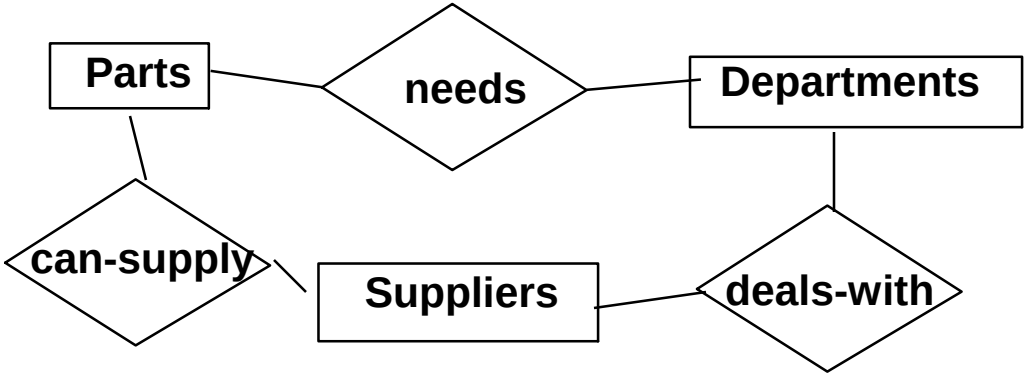
- Previous example illustrated case when two binary relationships were better than one ternary relationship.

- Opposite example: 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.)



vs.

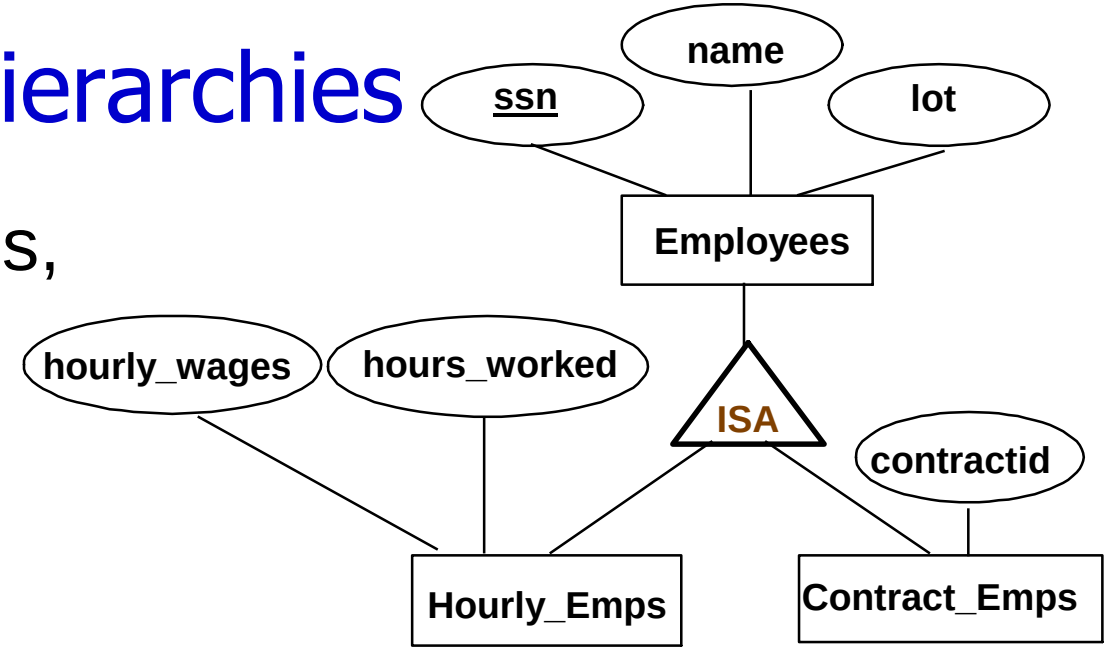


- S "can-supply" P, D "needs" P, and D "deals-with" S does not imply that D has agreed to buy P from S.
- How do we record *qty*?

Summary so far

- **Entities and Entity Set (boxes)**
- **Relationships and Relationship sets (diamonds)**
  - binary
  - n-ary
- **Key constraints (1-1,1-M, M-M, arrows on 1 side)**
- **Participation constraints (bold for Total)**
- **Weak entities - require strong entity for key**
- **Next, a couple more “advanced” concepts…**

ISA (`is a') Hierarchies



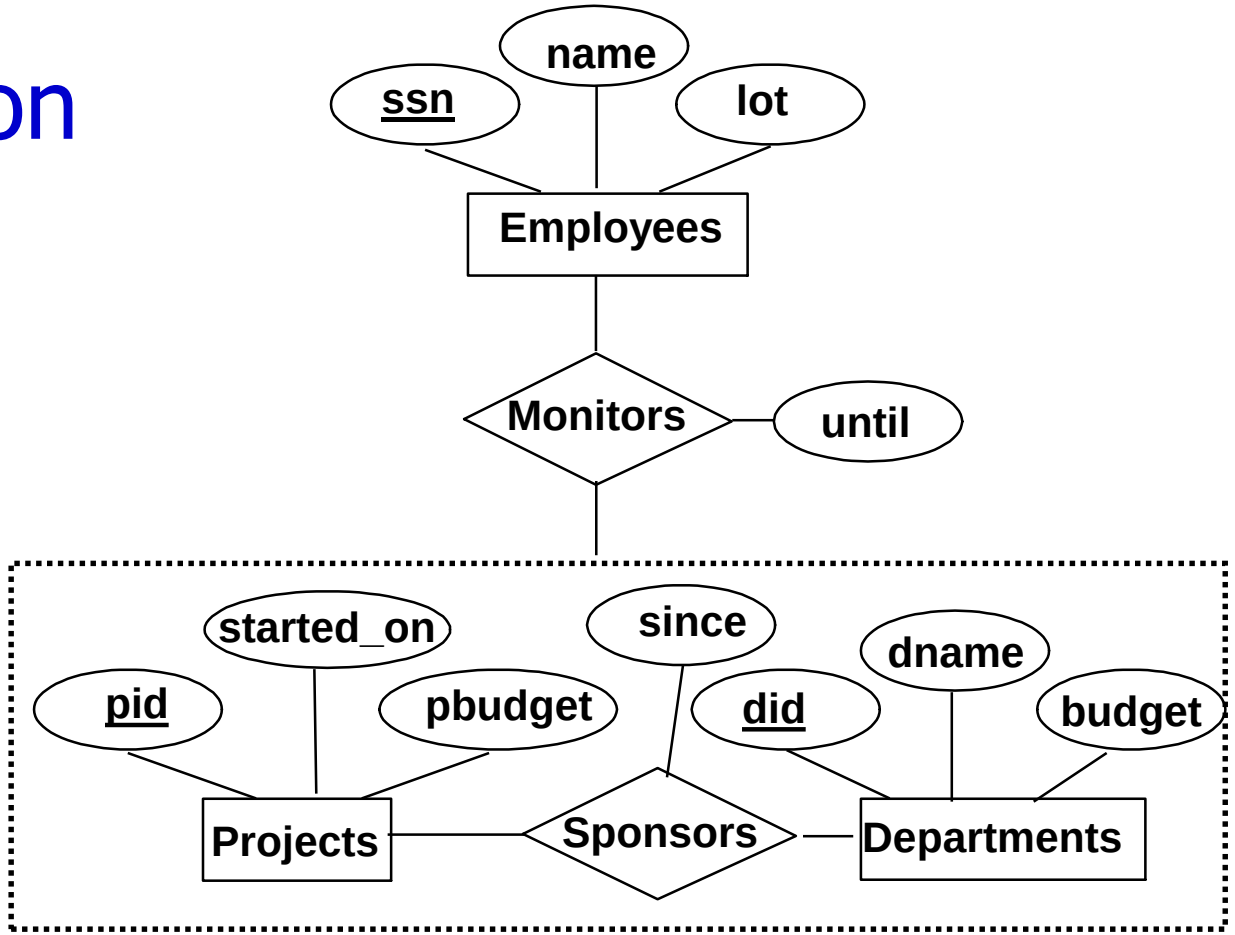
- As in C++, or other PLs, attributes are inherited.
- If we declare A **ISA** B, every A entity is also considered to be a B entity.

- ***Overlap constraints*: Can Simon 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

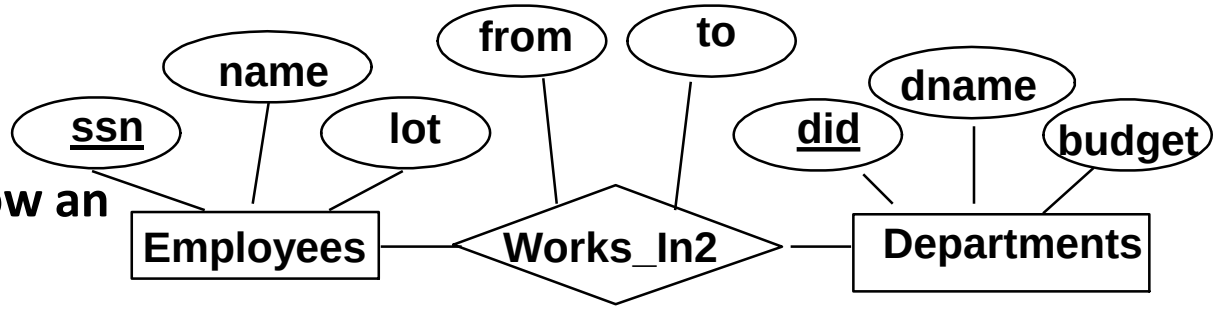
- **ER modeling *can* get tricky!**
- **Design choices:**
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary? Aggregation?
- **Note constraints of the ER Model:**
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.
    - We'll refine things in our logical (relational) design

Entity vs. Attribute

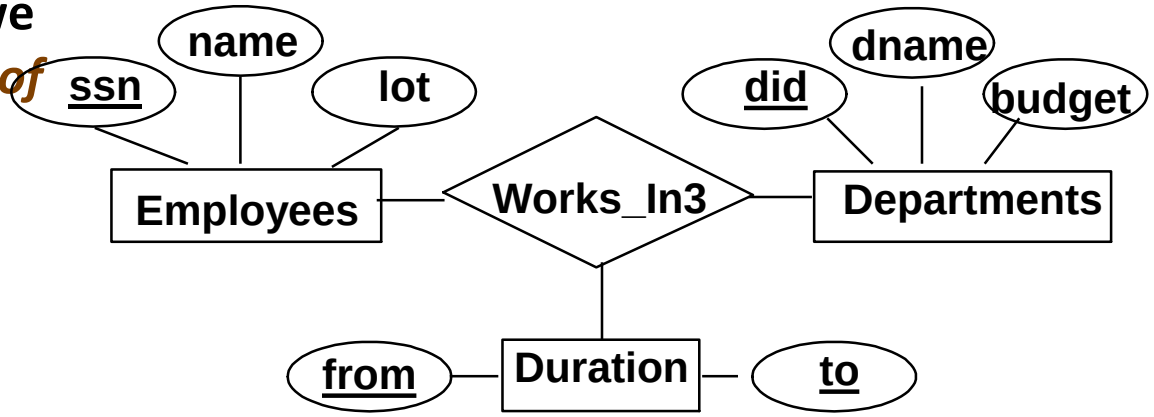
- **Should *address* be:**
  - attribute of Employees or
  - an entity (related to Employees)?

- **Depends upon use of address information, and the semantics of the data:**
  - If several addresses per employee, *address* must be an entity (since attributes cannot be set-valued).
  - If structure (city, street, etc.) is important, *address* must be modeled as an entity (since attribute values are atomic).

Entity vs. Attribute (Cont.)

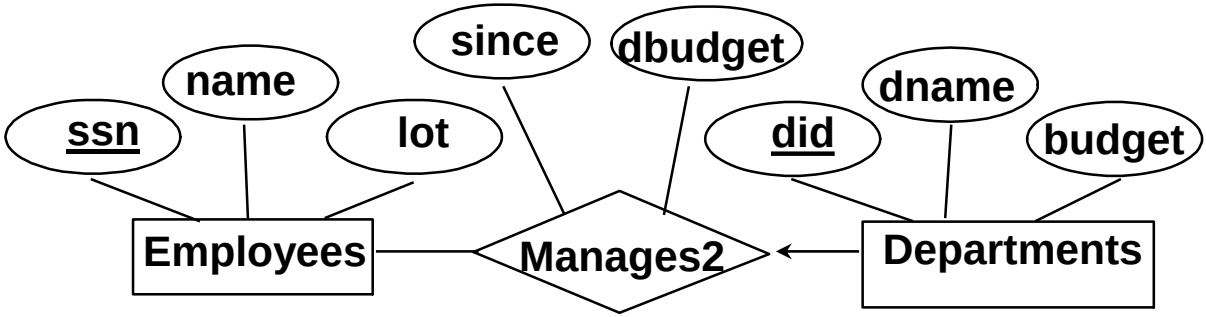


- Works_In2 does not allow an employee to work in a department for two or more periods.
- Similar to the problem of wanting to record several addresses for an employee: we want to record *several values of the descriptive attributes for each instance of this relationship.*



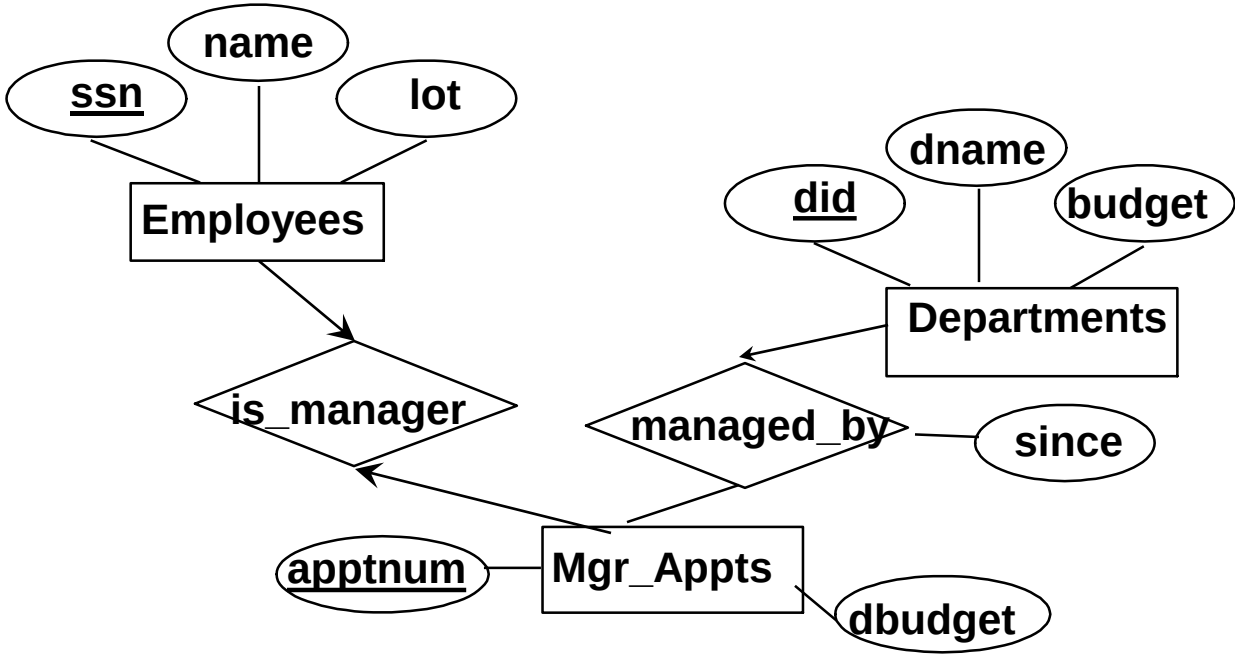
Entity vs. Relationship

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)

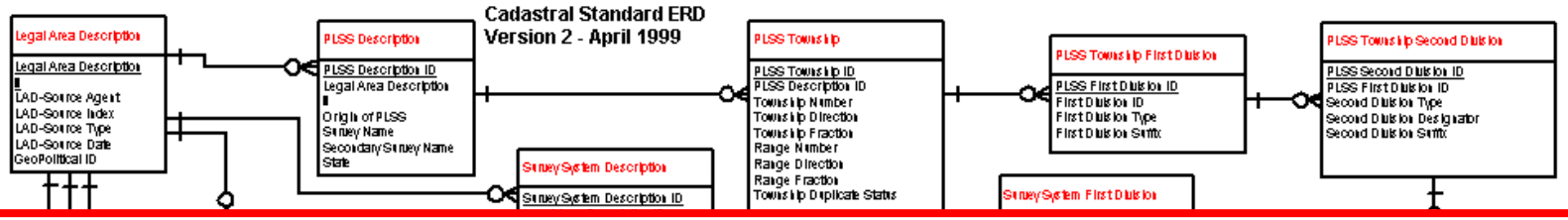


Now you try it

Try this at home - Courses database:

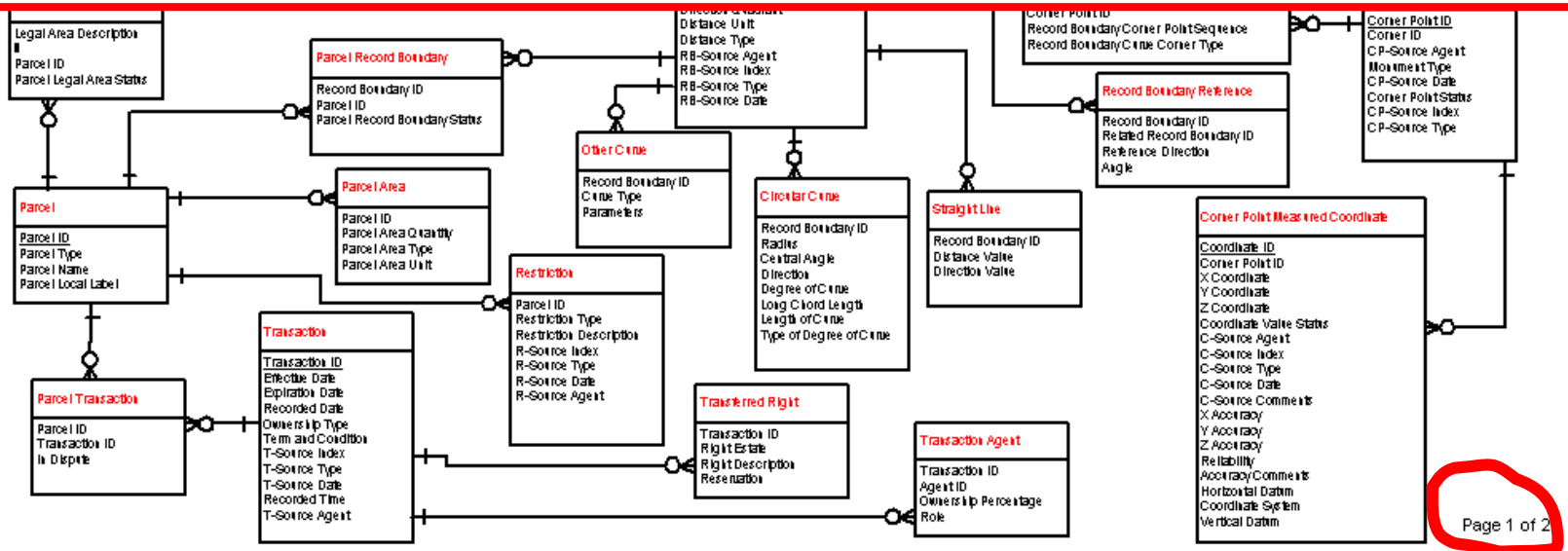
- **Courses, Students, Teachers**
- **Courses have ids, titles, credits, …**
- **Courses have multiple sections that have time/rm and exactly one teacher**
- **Must track students' course schedules and transcripts including grades, semester taken, etc.**
- **Must track which classes a professor has taught**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

A Cadastral E-R Diagram



cadastral: showing or recording property boundaries, subdivision lines, buildings, and related details

Source: US Dept. Interior Bureau of Land Management,
Federal Geographic Data Committee Cadastral Subcommittee
http://www.fairview-industries.com/standardmodule/cad-erd.htm



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
  - expressive constructs
  - close to how people think
  - Note: Many variations on ER model, Both graphically and conceptually
- 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*.
  - 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.

References:

1. A. Silberschatz, H.F. Korth, S.Sudarshan Data Base System and Concepts, fifth edition: Mc GrawHil

2. Tok Wang Ling National University of Singapore

3. Conceptual Design Using the ER Model inst.eecs.berkeley.edu › lecs › 11-ER