CS 133: Databases

Fall 2019
Lec 20 – 11/19
Database Design
Prof. Beth Trushkowsky

Warm-up Exercise

(See exercise sheet. You can start before class.)

Redundant data storage.
A sailor can't exist without a reservation, same with boats.
Integrity constraints we know should exist, can't (E.g., sid determines sname)

Goals for Today

- Learn about the process of designing a database to model a real-world application
- Understand how to encode an application in an entity-relationship (ER) diagram
- Reason about translating an ER model to a relational model

Database Design

- Requirements Analysis
  - user needs; what must database do?
- Conceptual Design
  - high level description (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

Data Models – Describing Data

- A ***Database design*** encodes some portion of the real world
- A ***Data Model*** is a set of concepts for thinking about this encoding

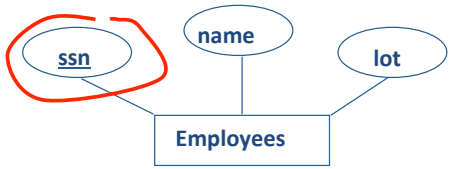
Helpful to start with a graphical representation: the *Entity-Relationship* model!

Integrity Constraints (ICs)

- Remember the "C" in ACID (Consistency)
- Integrity Constraint (IC): condition that must be true for *any* instance of the database
  - e.g., ***domain constraints***, ***keys and foreign key***
  - ICs are specified when schema is defined.
  - ICs are checked when relations are modified.
- Come from semantics of the real world!
  - Should be determined during Requirements Analysis and/or Conceptual Design phases

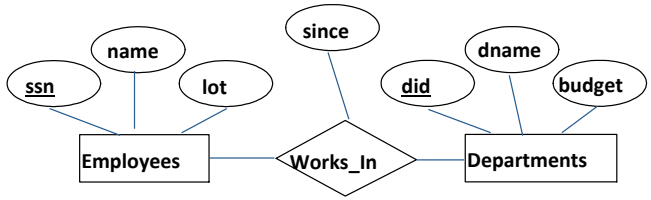
What "business rules" make sense

Entity-Relationship (ER) Model Basics



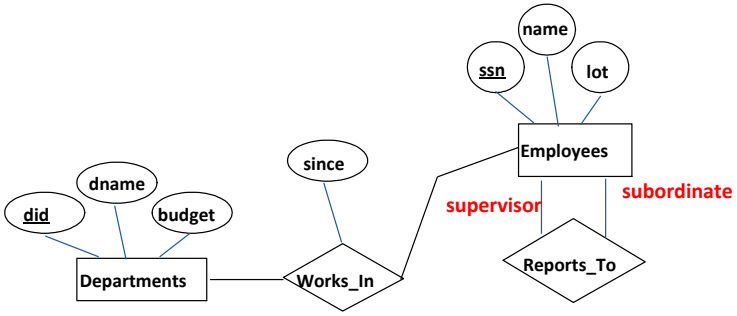
- *Entity:* Real-world object, distinguishable from other objects. An entity can have a set of *attributes*
  - Each **attribute** is atomic (not a list or set)
- *Entity Set*: A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. (Until 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., Alice ***works in*** Pharmacy department.
  - Relationships can have their own attributes
  - Relationships **uniquely identified only by participating entities**, *excluding 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 $e_1 \in E_1, \ldots, e_n \in E_n$

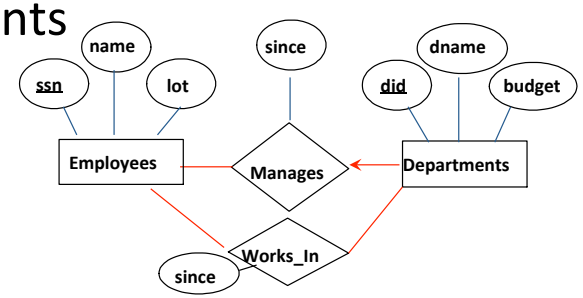
ER Model Basics (Cont.)



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

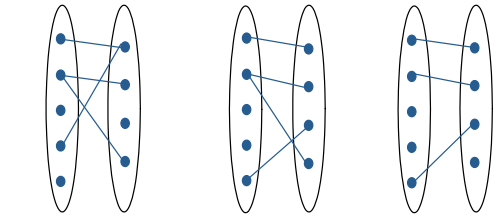
Key Constraints

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



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

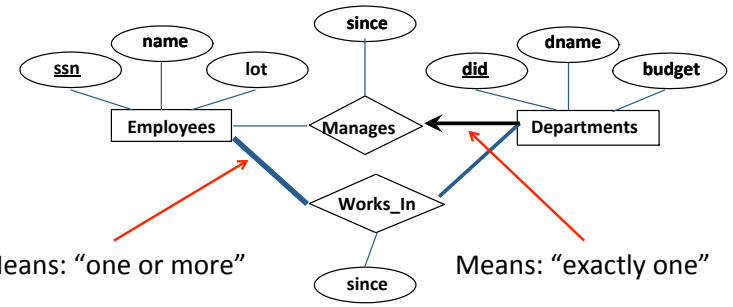
How to characterize Employees and Departments via Manages?



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

Participation Constraints

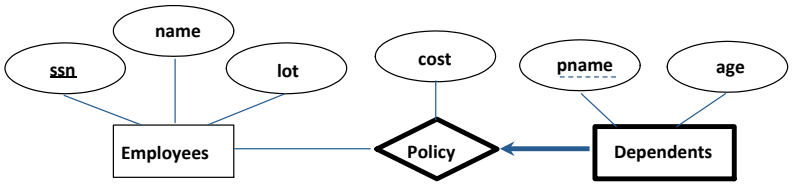
- Does every employee work in a department?
  - If so, the participation of Employees in Works_In is said to be ***total*** (vs. ***partial***)
- What if every department has an employee working in it?
- Use bold edge in ER diagram… basically means "one or more"



Weak Entities

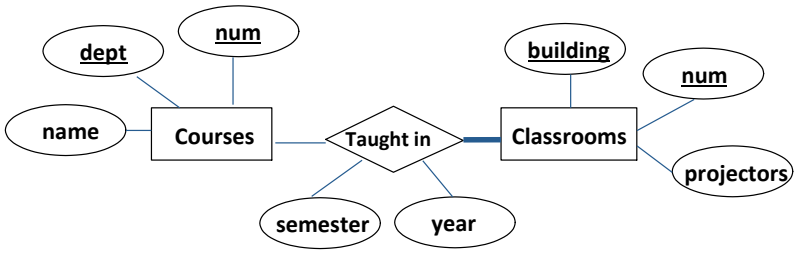
A ***weak entity*** can be identified uniquely only with the primary key of another (***owner***) entity.

- 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)

Exercise 2: Interpret E/R diagram



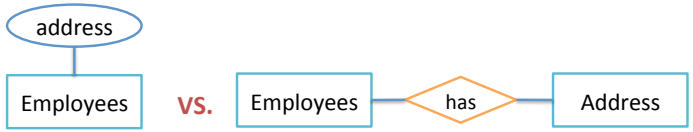
a) Bold edge → Every classroom must be taught in.
   Non-bold edge → Not every class has to be taught (in a classroom)
b) No. A relationship is uniquely identified by its participating entities only

Conceptual Design Using the ER Model

- ER modeling can get tricky!
- Example design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
- 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

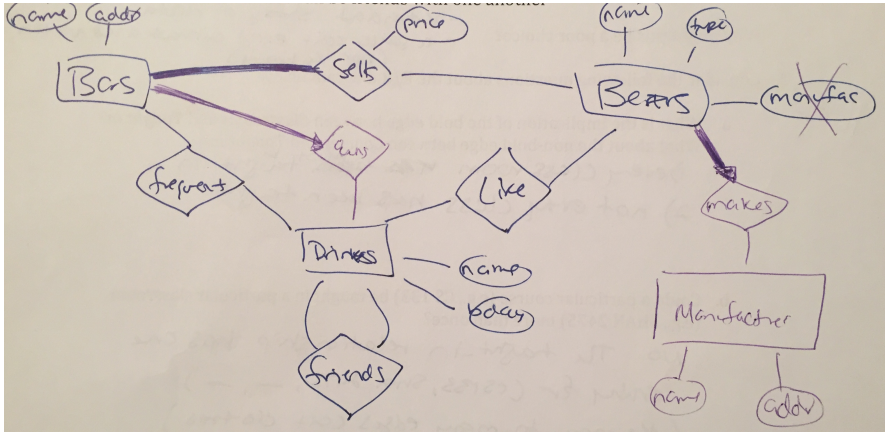
- E.g., capturing employee "Address":



- It depends! Semantics and usage
  - Several addresses per employee?
    - must be an entity!
    - atomic attribute types (no set-valued attributes!)
  - Care about structure? (city, street, etc.)
    - must be an entity!
    - atomic attribute types (no tuple-valued attributes!)

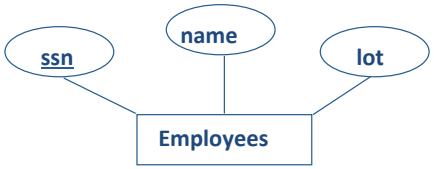
Exercise 3

- Example answer:



Logical DB Design: ER to Relational

Entity sets to tables




```
CREATE TABLE Employees
  (ssn CHAR(11),
   name CHAR(20),
   lot  INTEGER,
   PRIMARY KEY  (ssn))
```

Relationship Sets to Tables

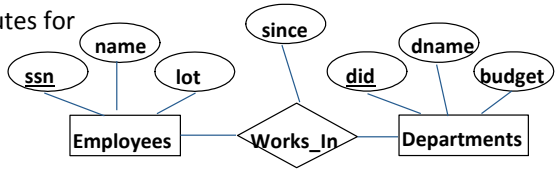
- In translating a **many-to-many** relationship set to a relation, attributes of the relation must include:

  1) Keys for each participating entity set (as ***foreign keys***).

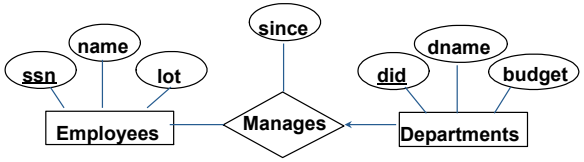
     This set of attributes forms a ***super key*** for the relation. Also a ***candidate key*** if no key constraints.

  2) All descriptive attributes for the relationship set.

```
CREATE TABLE Works_In(
  ssn  CHAR(1),
  did  INTEGER,
  since  DATE,
 PRIMARY KEY (ssn, did),
 FOREIGN KEY (ssn)
    REFERENCES Employees,
 FOREIGN KEY (did)
  REFERENCES Departments)
```



Translating ER with Key Constraints

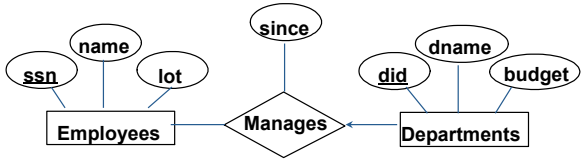


One way to translate the Manages Relationship (one-to-many):

```
CREATE TABLE  Manages(
 ssn  CHAR(11),
 did  INTEGER,
 since  DATE,
 PRIMARY KEY  (did),
 FOREIGN KEY (ssn) REFERENCES Employees,
 FOREIGN KEY (did) REFERENCES Departments
)
```

Primary key is from the "many" side of the relationship

Translating ER with Key Constraints (Take 2)

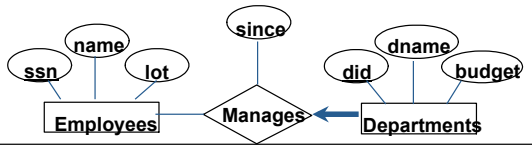


Since each department has a unique manager, we could instead **combine Manages and Departments** as **Dept_Mgr**:

```
CREATE TABLE  Dept_Mgr(
 did  INTEGER,
 dname  CHAR(20),
 budget  REAL,
 ssn  CHAR(11),
 since  DATE,
 PRIMARY KEY  (did),
 FOREIGN KEY (ssn) REFERENCES Employees)
```

Participation Constraints in SQL

- Capturing participation constraints more difficult
- Example involving one entity set in a binary relationship:




```
CREATE TABLE  Dept_Mgr(
   did  INTEGER,
   dname  CHAR(20),
   budget  REAL,
   ssn  CHAR(11) NOT NULL,
   since  DATE,
   PRIMARY KEY  (did),
   FOREIGN KEY  (ssn) REFERENCES Employees,
      ON DELETE NO ACTION)
```

Exercise 4 (attribute types omitted)

- CREATE TABLE **Drinkers** (
  ssn,
  name,
  birthday,
  PRIMARY KEY(ssn))

- CREATE TABLE **Friends** (
  friend1,
  friend2,
  PRIMARY KEY(friend1,friend2),

  FOREIGN KEY(friend1)
  REFERENCES Drinkers(ssn),

  FOREIGN KEY(friend2)
  REFERENCES Drinkers(ssn))

- CREATE TABLE **Bars_owns** (
  name,
  address,
  phone,
  ssn_owner NOT NULL,
  PRIMARY KEY(name),

  FOREIGN KEY(ssn_owner)
  REFERENCES Drinkers(ssn)

Translating Weak Entity Sets

- Weak entity set and identifying relationship set are translated into a single table.
  - When the owner entity is deleted, all owned weak entities must also be deleted.

```
CREATE TABLE  Dep_Policy (
   pname  CHAR(20),
   age  INTEGER,
   cost  REAL,
   ssn  CHAR(11) NOT NULL,
   PRIMARY KEY  (pname, ssn),
   FOREIGN KEY  (ssn) REFERENCES Employees,
      ON DELETE CASCADE)
```

… and more

- See textbook for a few more details, e.g.,