

The Entity-Relationship Model

Chapter 2

Instructor: Vladimir Zadorozhny

vladimir@sis.pitt.edu

Information Science Program

School of Information Sciences,

University of Pittsburgh

Database: a Set of Relations (Tables)

customer_id	customer_name	customer_street	customer_city
192-83-7465	Johnson	12 Alma St.	Palo Alto
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The customer table

account_number	balance
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The account table

customer_id	account_number
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The depositor table

- Find the name of the customer with customer-id 192-83-7465
select customer.customer_name
from customer
where customer.customer_id = '192-83-7465'

Database Design

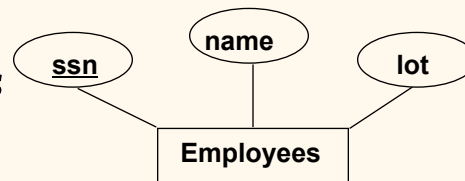
The process of designing the general structure of the database:

- ❖ Requires that we find a “good” collection of relation schemas.
 - Business decision – What attributes should we record in the database?
 - IS decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- ❖ Deciding on the physical layout of the database

Conceptual Database Design

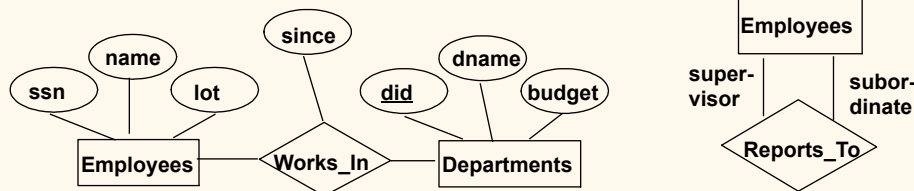
- ❖ Conceptual design: (*ER Model is used at this stage.*)
 - What are the *entities* and *relationships* in the enterprise?
 - What information about these entities and relationships should we store in the database?
 - What are the *integrity constraints* or *business rules* that hold?
 - A database ‘schema’ in the ER Model can be represented pictorially (*ER diagrams*).
 - Can map an ER diagram into a relational schema.

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.)



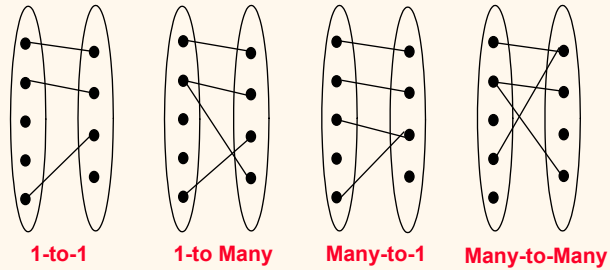
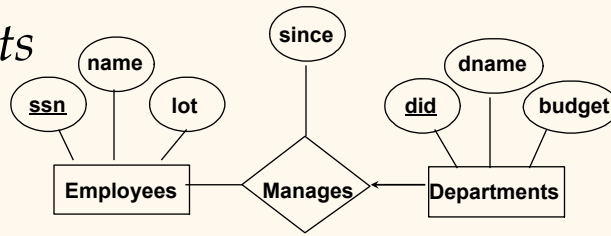
- ❖ **Relationship:** Association among two or more entities. E.g., Attishoo works in Pharmacy department.
- ❖ **Relationship Set:** Collection of similar relationships.
 - An n-ary relationship set R relates n entity sets E1 ... En; each relationship in R involves entities e1 in E1, ..., en in En
 - Same entity set could participate in different relationship sets, or in different “roles” in same set.

Key Constraints

❖ Consider Works_In:

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

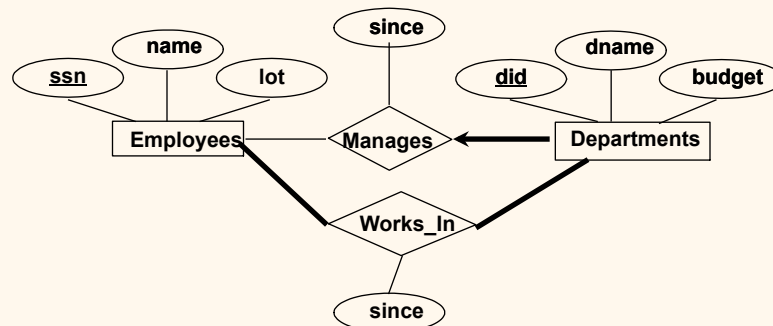
- ❖ In contrast, each dept has at most one manager, according to the key constraint on Manages.



Participation Constraints

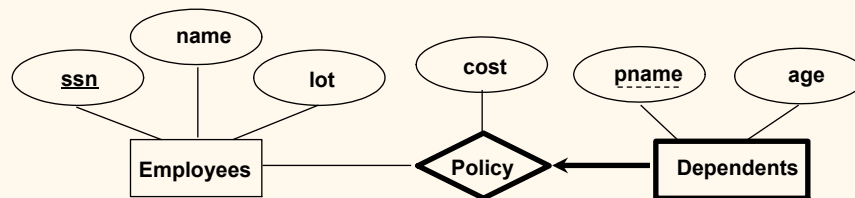
❖ Does every department have a manager?

- 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 row of the Manages table (with a non-null *ssn* value!)



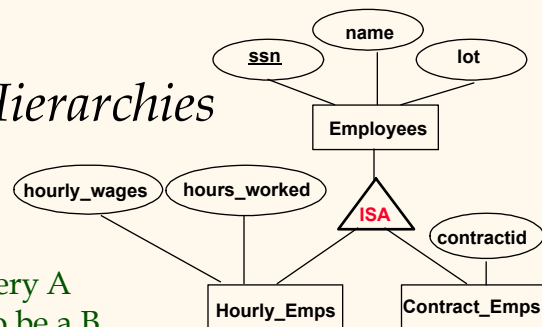
Weak Entities

- ❖ 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?
- Should a concept be modeled as an entity or a relationship?

❖ Constraints in the ER Model:

- A lot of data semantics can (and should) be captured.
- But some constraints cannot be captured in ER diagrams.

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.)

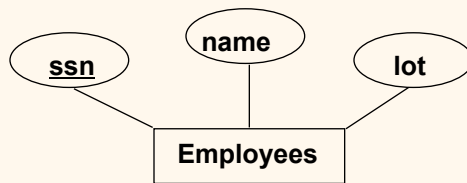
- ❖ Several kinds of integrity constraints can be expressed in the ER model: *key constraints*, *participation constraints*, and *overlap/covering constraints* for ISA hierarchies. Some *foreign key constraints* are also implicit in the definition of a relationship set.
 - Some constraints (notably, *functional dependencies*) cannot be expressed in the ER model.
 - 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, whether or not to use ISA hierarchies.
- ❖ Ensuring good database design: resulting relational schema should be analyzed and refined further. FD information and normalization techniques are especially useful.

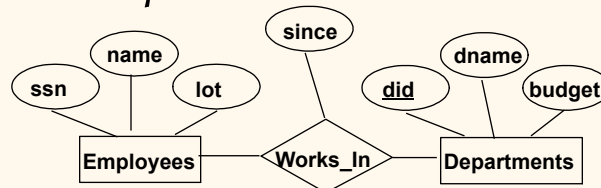
Logical DB Design: ER to Relational

❖ Entity sets to tables:



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

Relationship Sets to Tables



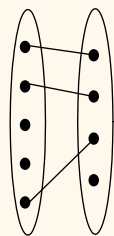
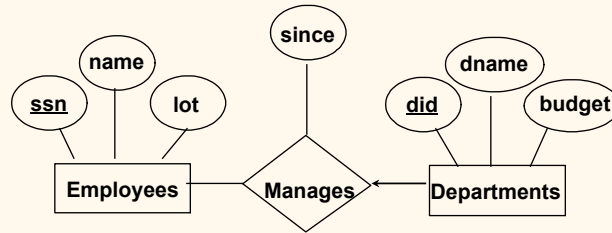
❖ In translating a relationship set to a relation, attributes of the relation must include:

- Keys for each participating entity set (as foreign keys).
 - This set of attributes forms a *superkey* for the relation.
- All descriptive attributes.

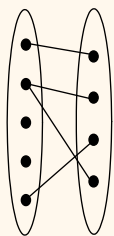
```
CREATE TABLE Works_In(
ssn CHAR(11),
did INTEGER,
since DATE,
PRIMARY KEY (ssn, did),
FOREIGN KEY (ssn)
REFERENCES Employees,
FOREIGN KEY (did)
REFERENCES Departments)
```


Review: Key Constraints

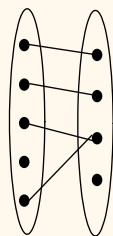
- ❖ Each dept has at most one manager, according to the key constraint on **Manages**.



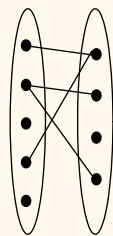
1-to-1



1-to-Many



Many-to-1



Many-to-Many

Translation to relational model?

Translating ER Diagrams with Key Constraints

- ❖ Map relationship to a table:
 - Note that **did** is the key now!
 - Separate tables for Employees and Departments.
- ❖ Since each department has a unique manager, we could instead combine Manages and Departments.

```

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

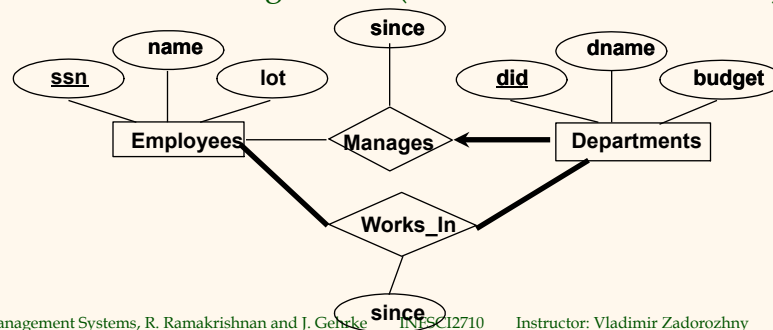
```

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

Review: 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 row of the Manages table (with a non-null *ssn* value!)



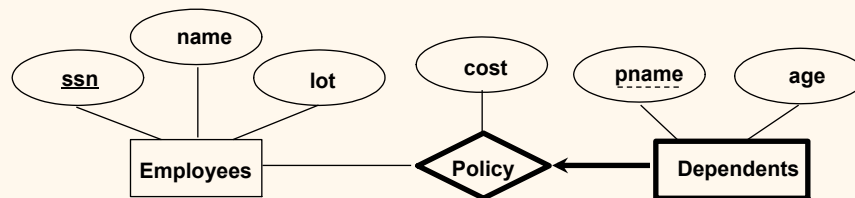
Participation Constraints in SQL

- ### ❖ We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).

```
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)
```

Review: 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 (1 owner, many weak entities).
 - Weak entity set must have total participation in this *identifying* relationship set.

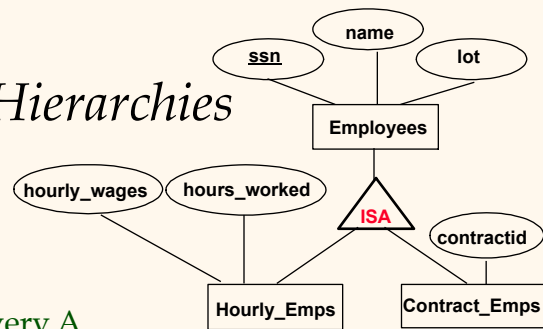


Translating Weak Entity Sets

- ❖ 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)
```

Review: ISA 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*)

Translating ISA Hierarchies to Relations

- ❖ **General approach**:
 - 3 relations: Employees, Hourly_Emps and Contract_Emps.
 - Hourly_Emps: Every employee is recorded in Employees. For hourly emps, extra info recorded in Hourly_Emps (*hourly_wages, hours_worked, ssn*); must delete Hourly_Emps tuple if referenced Employees tuple is deleted).
 - Queries involving all employees easy, those involving just Hourly_Emps require a join to get some attributes.
- ❖ **Alternative: Just Hourly_Emps and Contract_Emps**.
 - Hourly_Emps: *ssn, name, lot, hourly_wages, hours_worked*.
 - Each employee must be in one of these two subclasses.