

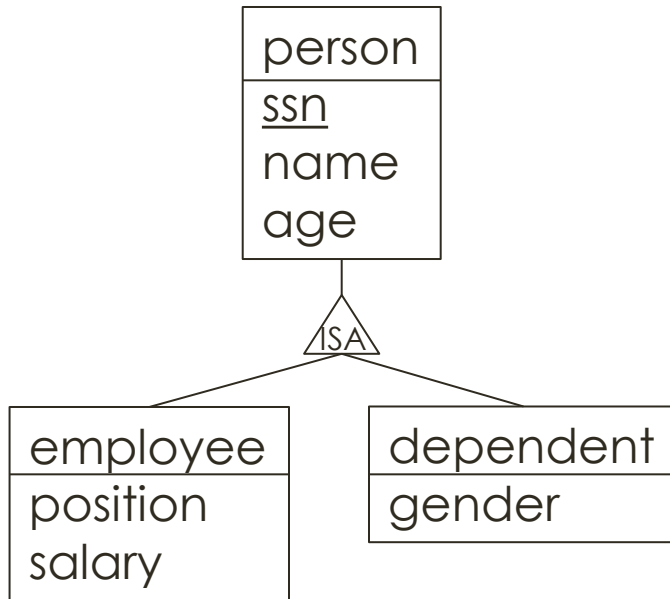


# DB Design

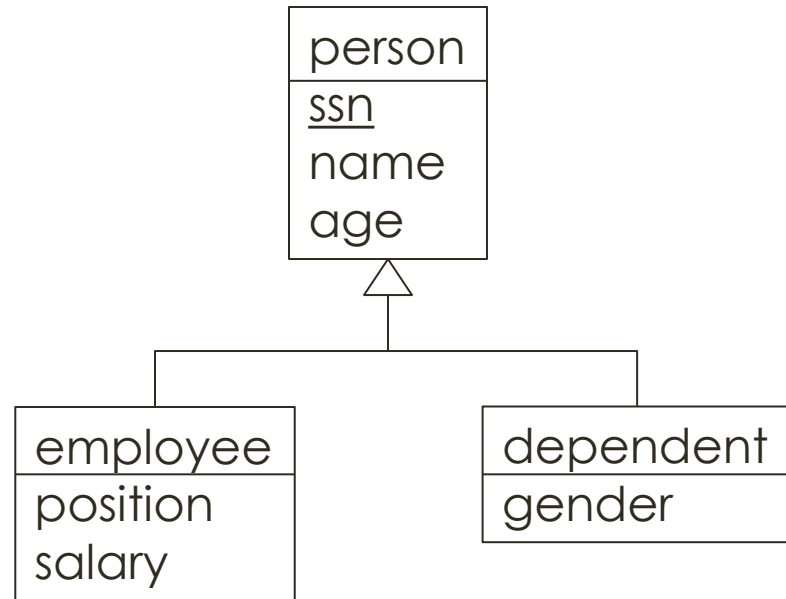
Week 4

# ISA relationship among entities

notation in the book:

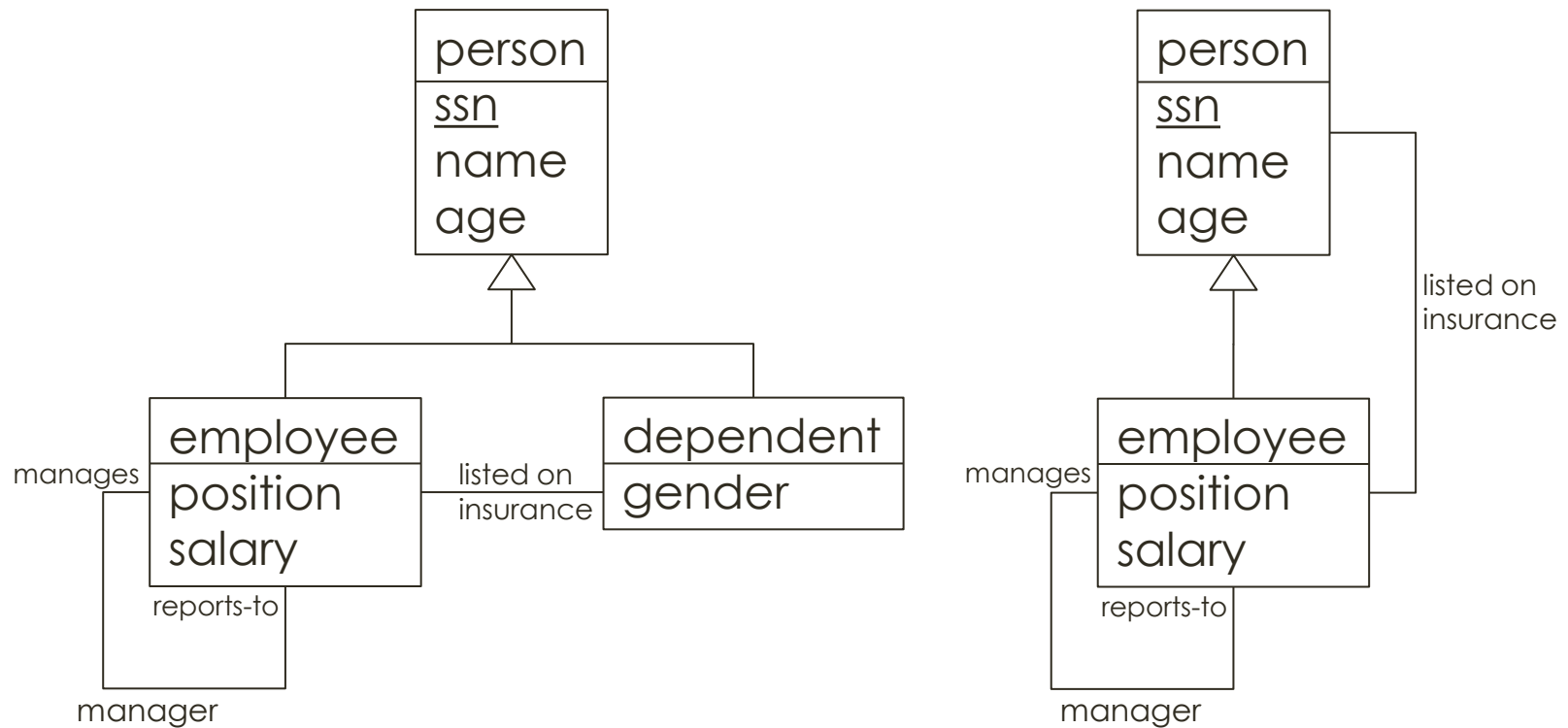


UML notation:



The person entity is the superclass; the employee and dependent entities are the subclasses. They inherit all of the attributes (and relationships) from the person.  
Note: subclasses don't need a key; they'll use key of superclass.

# Several modeling options for dependent



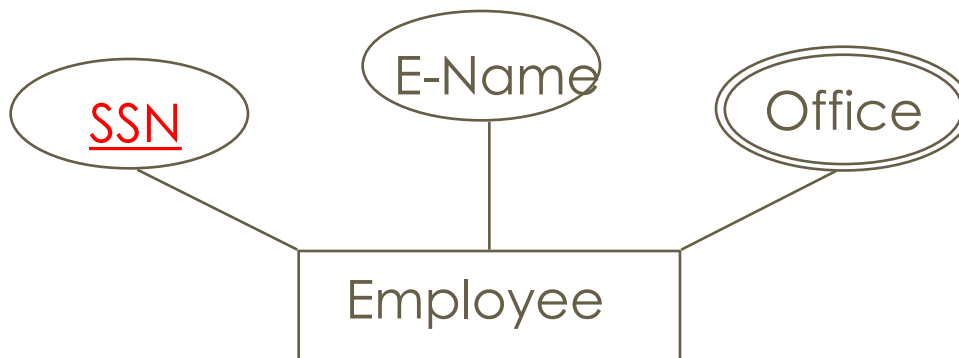
Is dependent modeled separately from person? Or not.

# Multi-valued Attribute

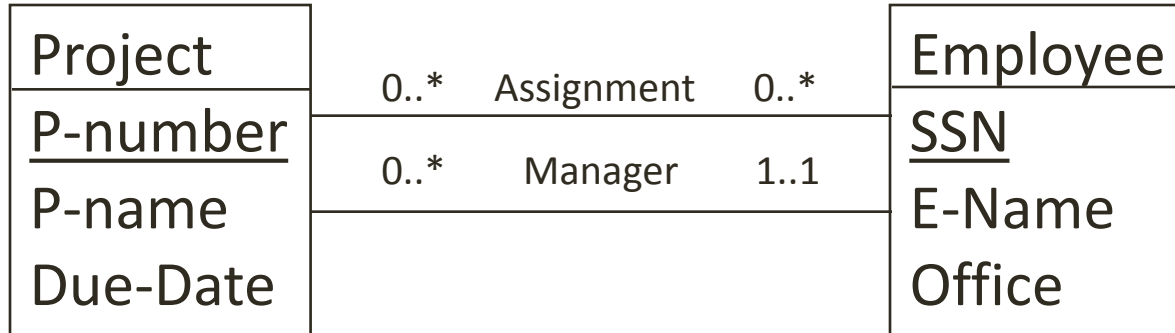
An attribute that can have multiple values for an entity.

Sometimes indicated by double circle, as shown.

This is an extension to the original ER model.



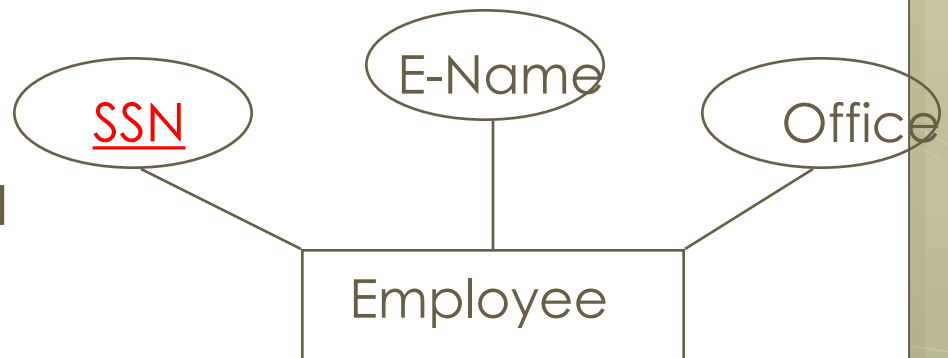
# Converting ER to Relational Schema



# 1. Translate each entity into a table, with keys.

- Entity :

- can be represented as a table in the relational model
- has a **key** ... which becomes a key for the table



```
CREATE TABLE Employee  
(SSN CHAR(11) NOT NULL,  
E-Name CHAR(20),  
Office INTEGER,  
PRIMARY KEY (SSN))
```

## 2. Create a table for the multi-valued attribute.

Note: a relational DBMS may or may not allow multi-valued attributes.

How many offices can one employee have?

Just one

Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name, Office)

VS.

More than  
one; define  
extra table.

Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name)  
Office-Assignment(SSN, Office)

# Sample Data

Just one

Project(P-number, P-name, Due-Date)

Employee(SSN, E-Name, **Office**)

12 Smith O-105

15 Wei O-110

More than  
one; define  
extra table.

Project(P-number, P-name, Due-Date)

Employee(SSN, E-Name)

12 Smith

15 Wei

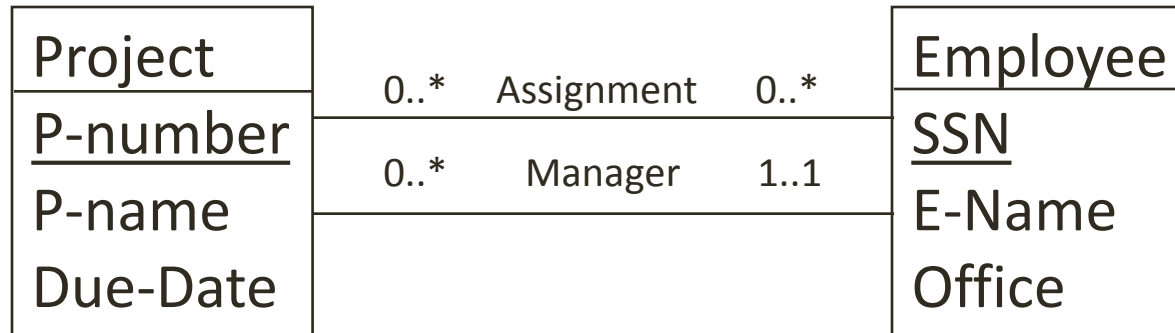
Office-Assignment(SSN, **Office**)

12 O-105

12 O-106

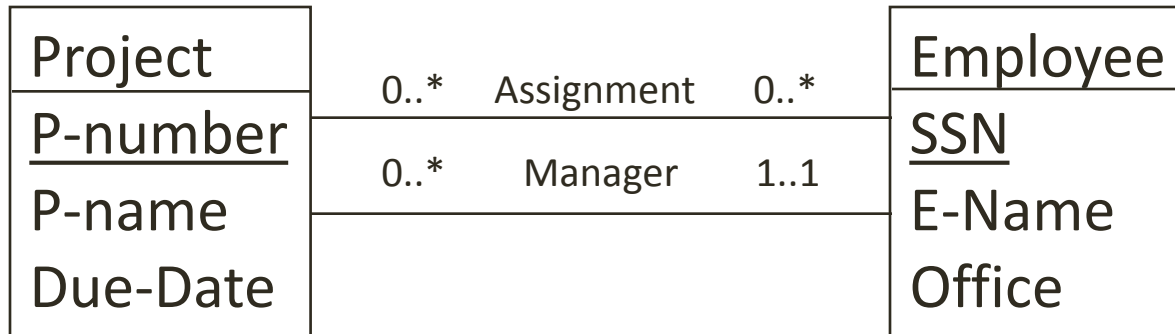
15 O-110

### 3. Translate each **many-to-many** relationship set into a table



**What are the attributes and what is the key for Assignment?**

Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name, Office)



Answer: Assignment(P-Number, SSN)

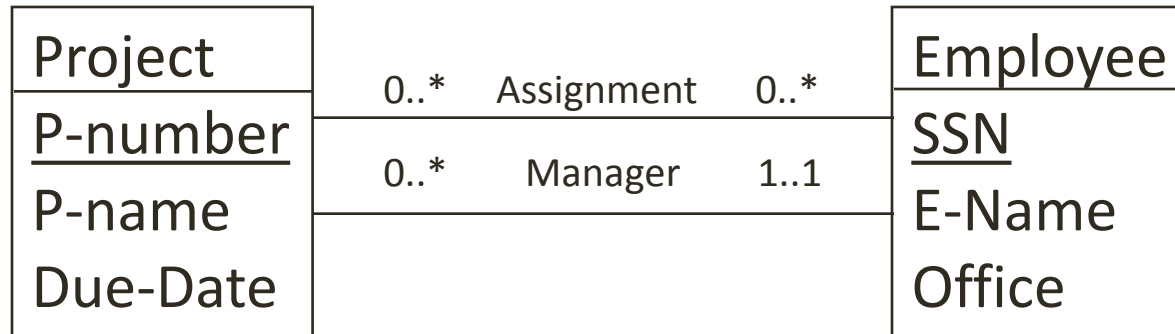
**P-Number** is a foreign key for Project

**SSN** is a foreign key for Employee

Project(P-Number, P-Due-Date)

Employee(SSN, E-Name, Office)

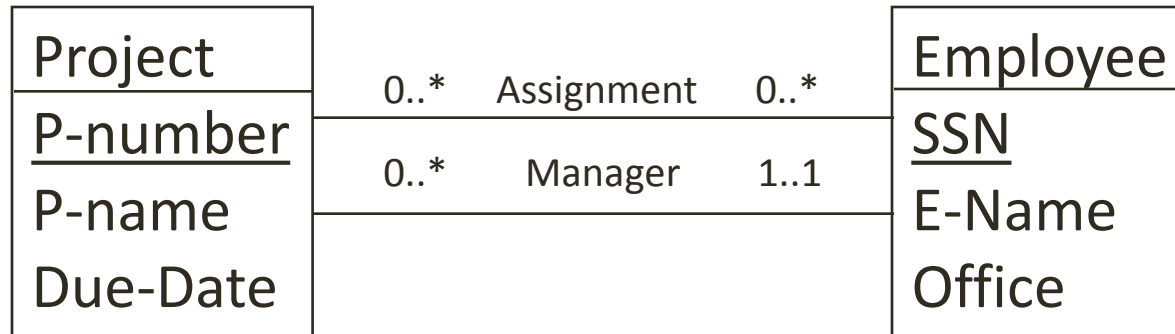
## What do we do with a **one-to-many** relationship?



For example, what do we do with Manager?

Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name, Office)

## 4. Create a foreign key for a 1-to-many relationship.

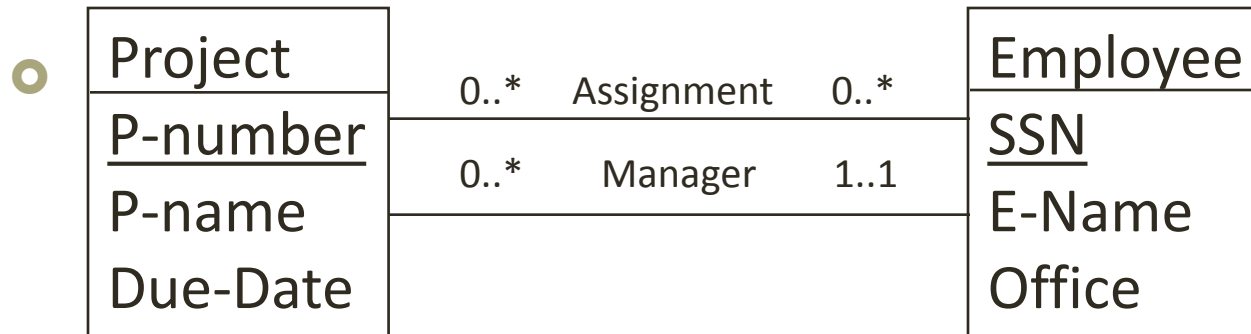


Project(P-number, P-name, Due-Date, **Manager**)  
Employee(SSN, E-Name, Office)

**Manager is a foreign key (referencing the Employee relation)**

value of Manager must match an SSN in Employee

## 4. Or...Create a table for a 1-many relationship.



Project(P-number, P-name, Due-Date, Manager)  
Employee(SSN, E-Name, Office)

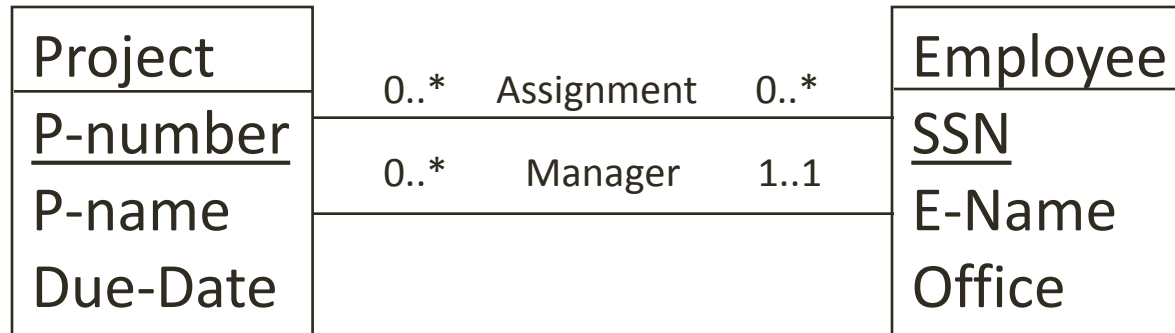
vs.

Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name, Office)  
Manager(P-number, SSN)

What are the tradeoffs between these two?

Note:  
P-number  
is the key  
for Manager

## What if SSN is the key for Manager?

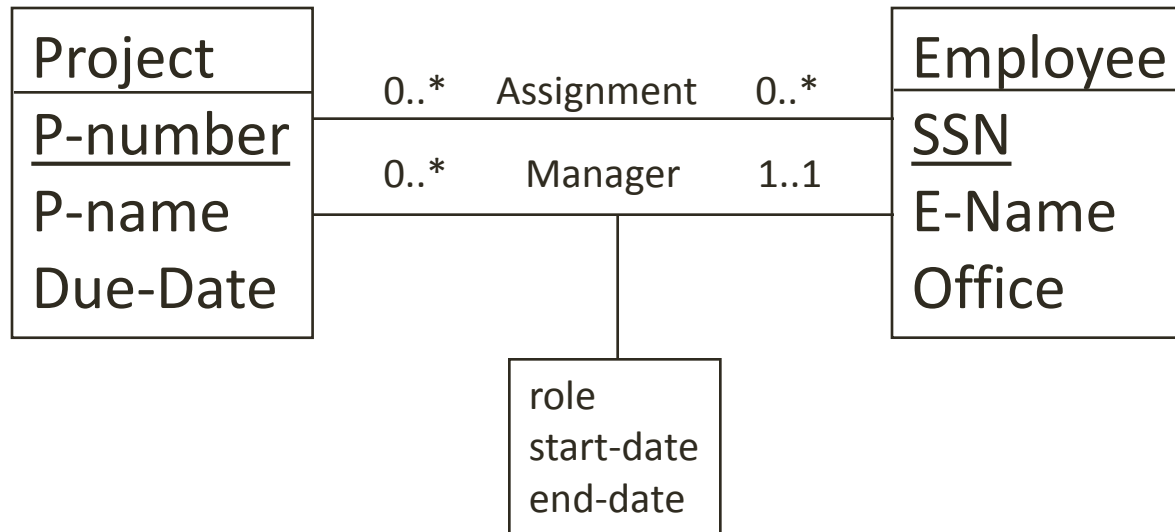


Project (P-number, P-name, Due-Date)  
Employee (SSN, E-Name, Office, Managed-project)

vs.

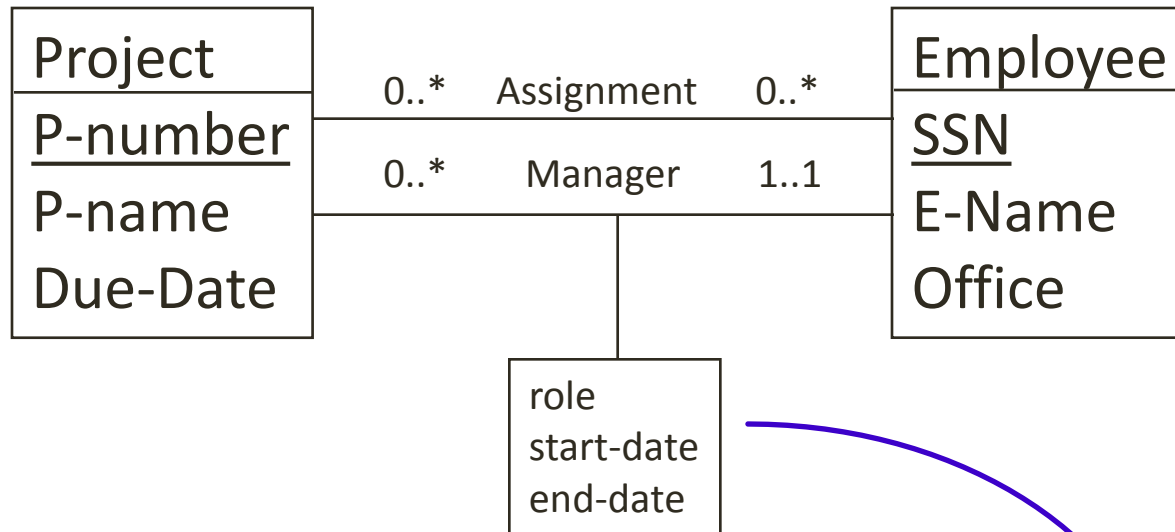
Project (P-number, P-name, Due-Date)  
Employee (SSN, E-Name, Office)  
Manager (P-number, SSN)

## What if a many-to-many relationship has attributes?



Assignment(P-number, SSN) ◉  
Project(P-number, P-name, Due-Date)  
Employee(SSN, E-Name, Office)

# Add attributes to the table for the relationship

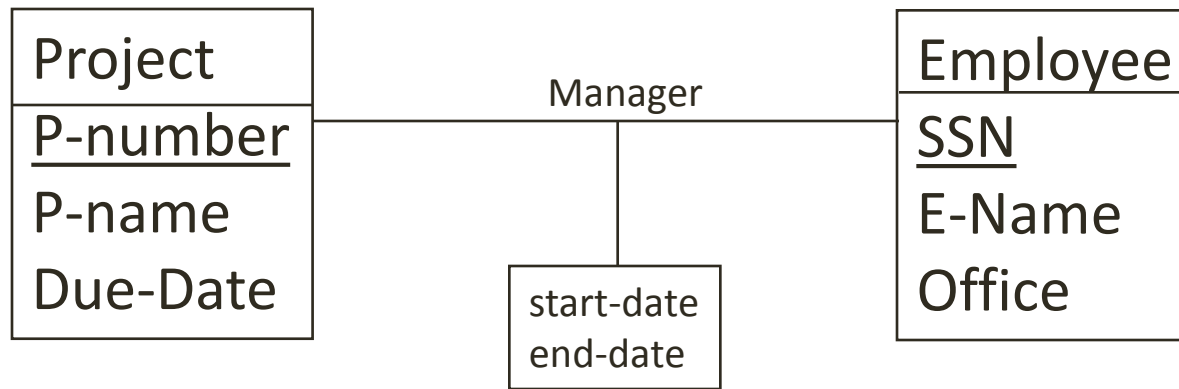


Assignment(P-number, SSN, role, start-date, end-date)

Project(P-number, P-name, Due-Date)

Employee(SSN, E-Name, Office)

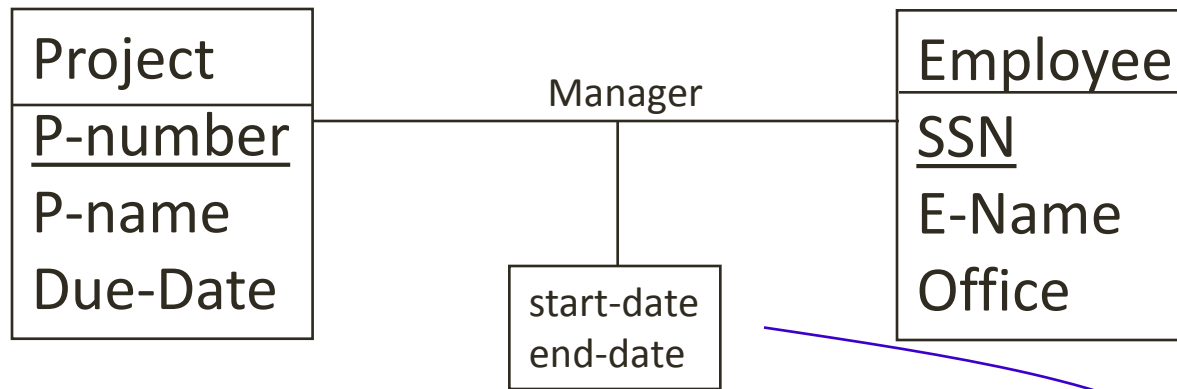
## What if a 1-to-many relationship has an attribute?



Project(P-number, P-name, Due-Date, Manager)

Employee(SSN, E-Name, Office)

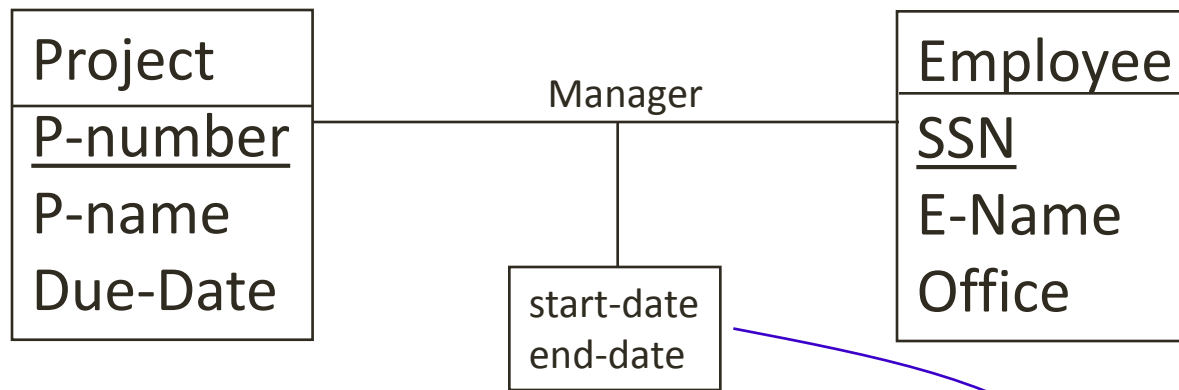
# Add attributes to the table for the relationship?



Project(P-number, P-name, Due-Date, Manager, start-date, end-date)

Employee(SSN, E-Name, Office)

# Add attributes to the table for the relationship?



No, bad idea.

You should use an extra table for the relationship.

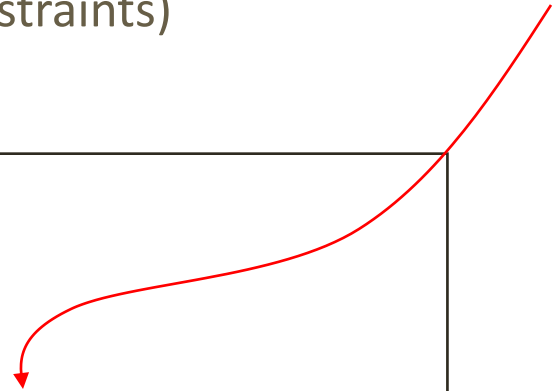
Project(P-number, P-name, Due-Date,  
Manages(P-number, SSN, start-date, end-date)  
Employee(SSN, E-Name, Office)

Notice the key for the Manages table.

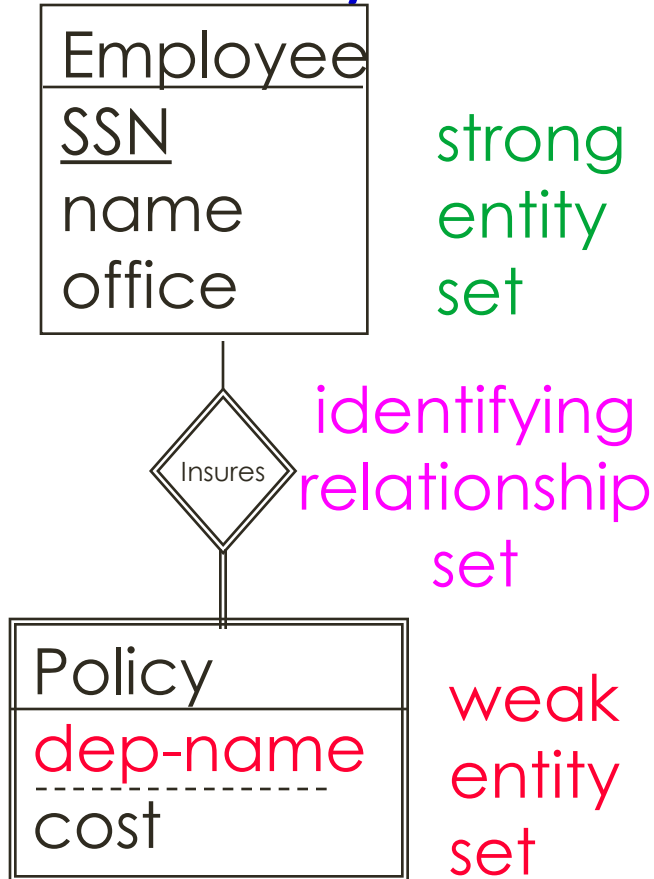
# Participation Constraints in SQL

We can require any table to be in a binary relationship using a foreign key which is required to be NOT NULL (but little else without resorting to CHECK constraints)

```
CREATE TABLE Department (  
  d-code          INTEGER,  
  d-name          CHAR(20),  
  manager-ssn    CHAR(9) NOT NULL,  
  since          DATE,  
  PRIMARY KEY (d-code),  
  FOREIGN KEY (manager-ssn) REFERENCES Employee,  
  ON DELETE NO ACTION)
```



# Weak Entity Sets



# Translating Weak Entity Sets

- Weak entity sets and identifying relationship sets are translated into a single table. Must include **key of strong entity set**, as a foreign key.
- When the owner entity is deleted, all owned weak entities must also be deleted.

```
CREATE TABLE Insurance_Policy (  
  dep-name      CHAR(20),  
  cost          REAL,  
  ssn          CHAR(11) NOT NULL,  
  PRIMARY KEY  (dep-name, ssn),  
  FOREIGN KEY  (ssn) REFERENCES Employee, ON DELETE CASCADE)
```

# Translation Steps: ER to Tables

- Create table and choose key for each entity; include single-valued attributes.
- Create table for each weak entity; include single-valued attributes. Include key of owner as a foreign key in the weak entity. Set key as foreign key of owner plus local, partial key.
- For each 1:1 relationship, add a foreign key to one of the entity sets involved in the relationship (a foreign key to the other entity in the relationship)\*.
- For each 1:N relationship, add a foreign key to the entity set on the N-side of the relationship (to reference the entity set on the 1-side of the relationship)\*.
- \* Unless relationship set has attributes. If it does, create a new table for the relationship set.

## Translation Steps: ER to Tables (cont.)

- For each M:N relationship set, create a new table. Include a foreign key for each participant entity set, in the relationship set. The key for the new table is the set of all such foreign keys.
- For each multi-valued attribute, construct a separate table. Repeat the key for the entity in this new table. It will serve as both the key for this table as well as a foreign key to the original table for the entity.
- ◉ This algorithm from Elmasri & Navathe, *Fundamentals of Database Systems*

Note UML Diagrams can be at two levels:

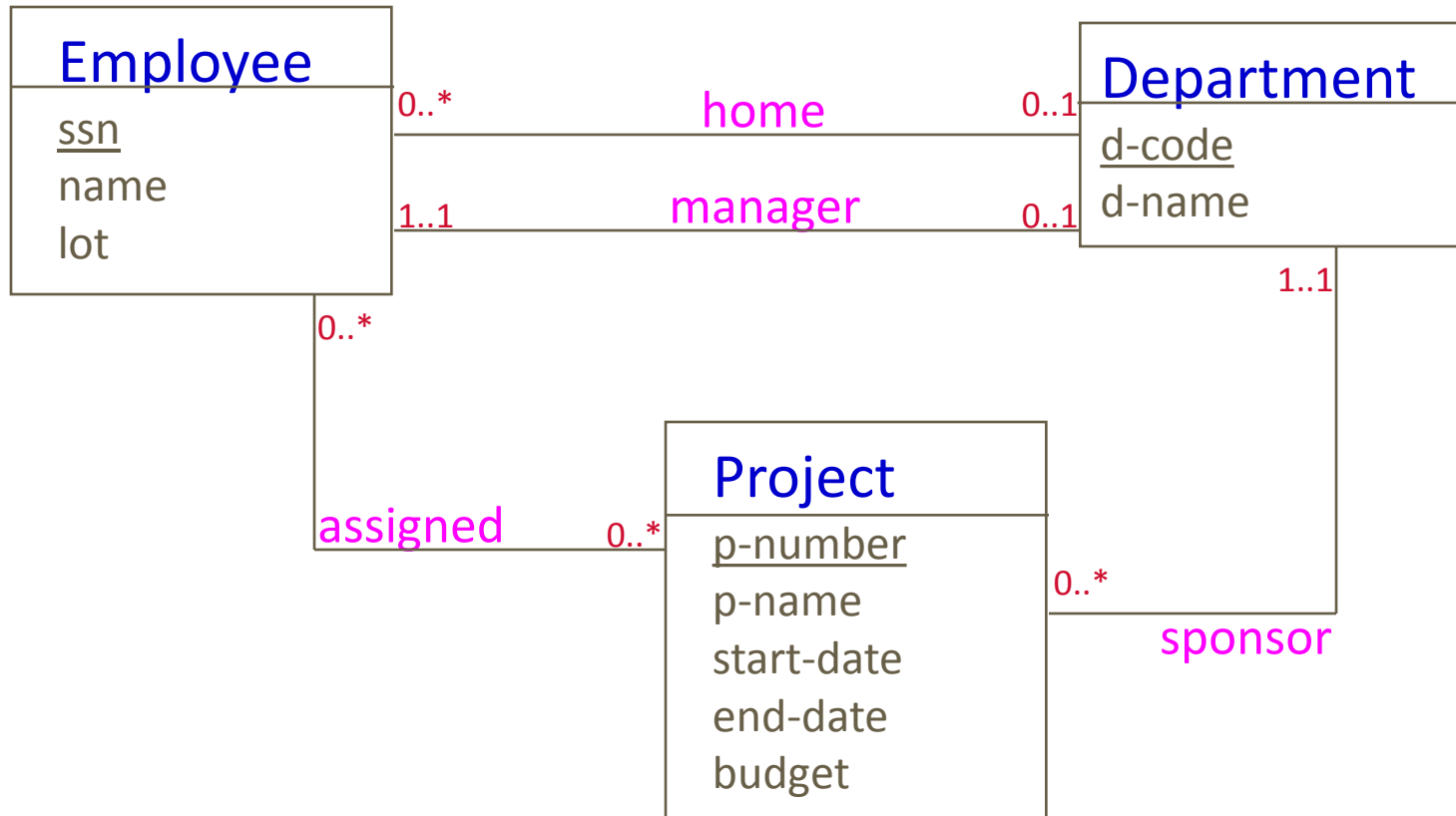
the Conceptual Level

and

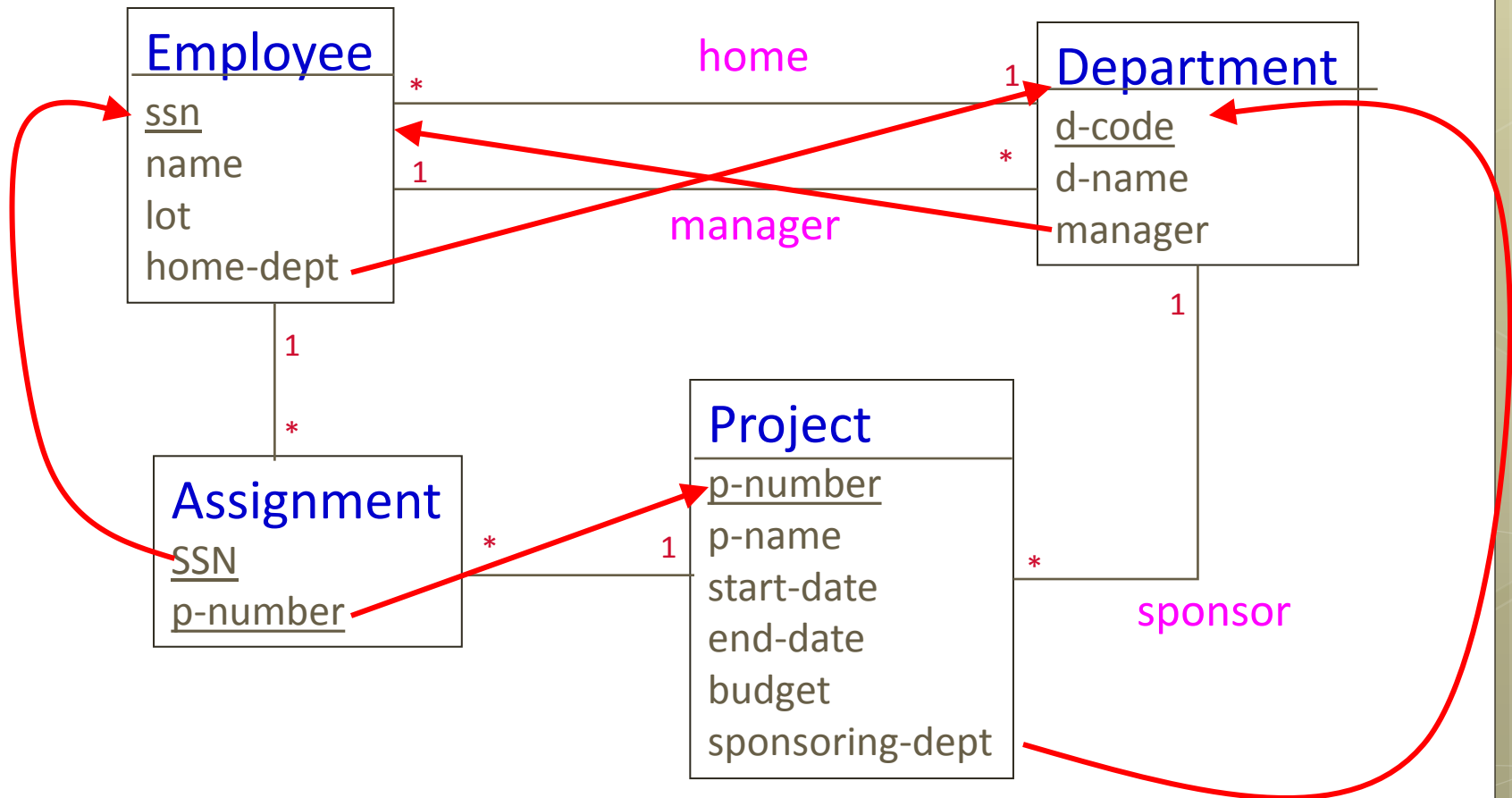
the Relational Table level.

The difference is primarily with the many-to-many relationship sets.

# Entity-Relationship Diagram



# Equivalent Relational Schema



Notice that the relationship shown in this diagram aren't really needed. *Foreign keys* reference other tables.