

# CSC 261/461 – Database Systems

## Lecture 8

Fall 2017

# Announcement

- Project 2 is out.
- Not a group project!

# Agenda

- More about ER model
- ER model to Relation (Table)

# Chen Notation vs Crow's Foot Notation

Textbook inverts the **(min, max)** notations. We are following that for consistency

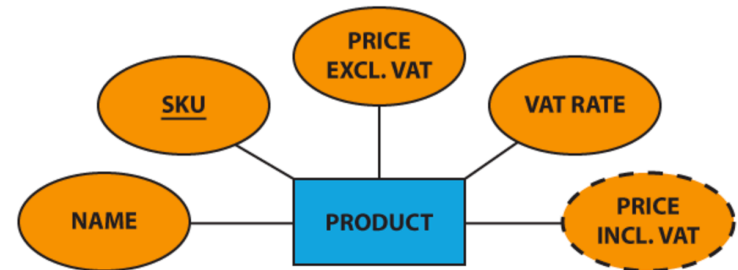
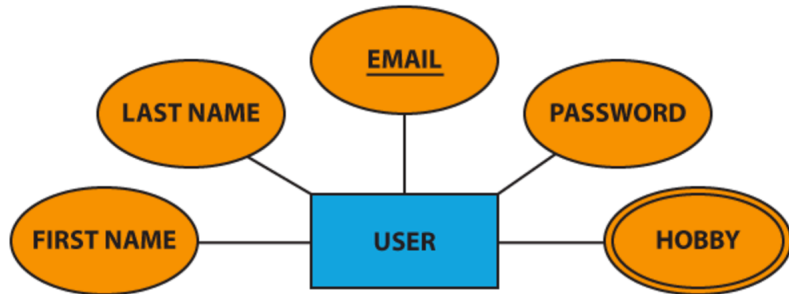
Crow's foot



Chen Notation



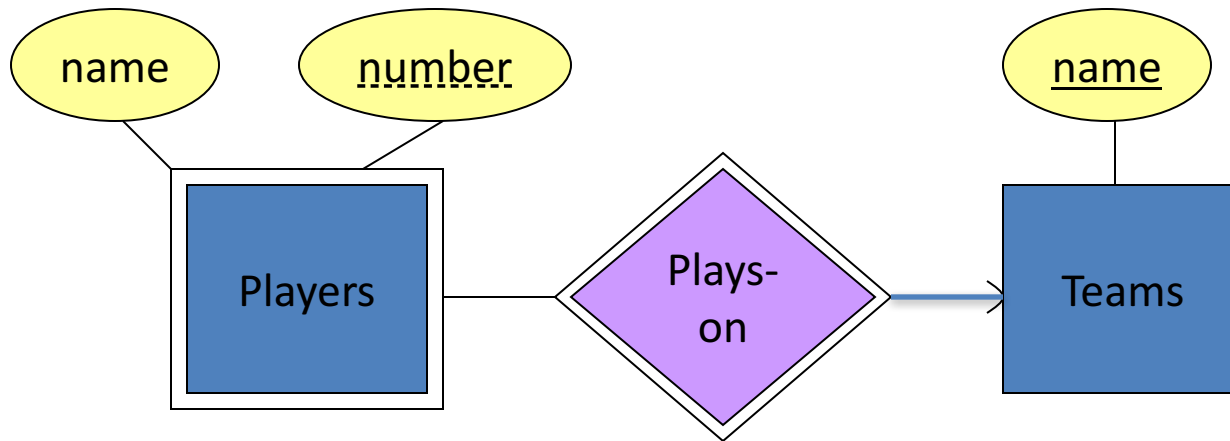
# Special Symbols Used



# Strong and Weak Entities

- A **weak entity** is an entity whose instances **cannot** exist in the database without the existence of an instance of another entity
- Any entity that is not weak entity is called a **strong entity**
  - Instances of a strong entity can exist in the database independently
- The **weak entity's identifier** is a combination of the **identifier** of the owner entity and the **partial key** of the weak entity.

# In E/R Diagrams



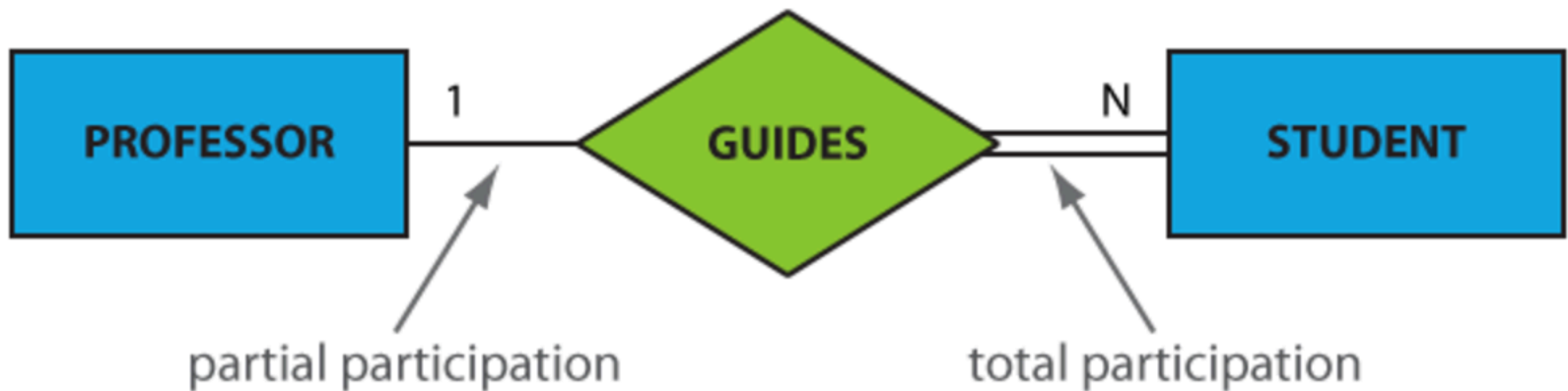
- Double diamond for *supporting* many-one relationship.
- Double rectangle for the weak entity set.

## Example: Weak Entity Set

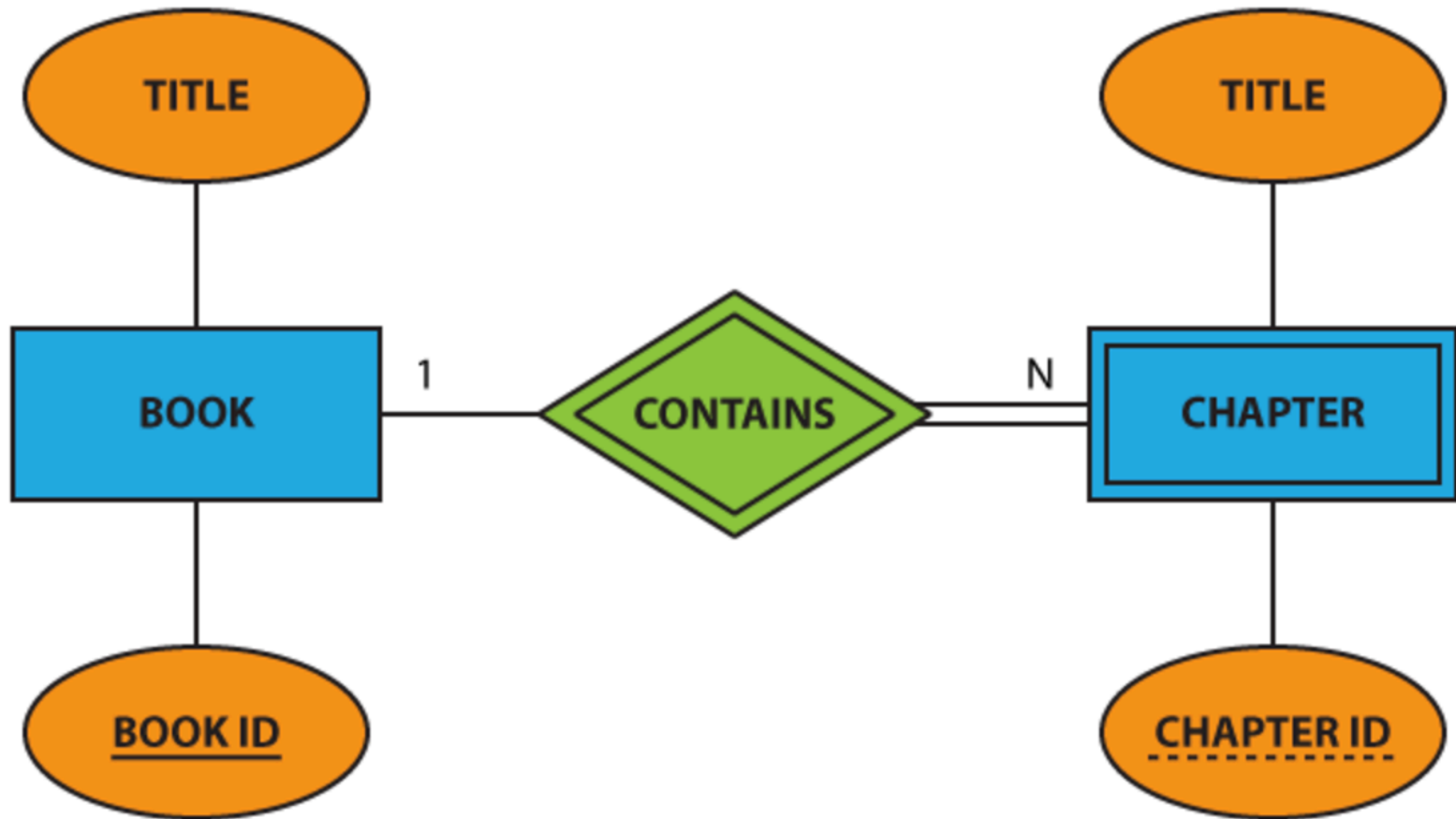
- **name** is almost a key for football players, but there might be two with the same name.
- **number** is certainly not a key, since players on two teams could have the same number.
- But **number**, together with the team **name** related to the player should be unique.

# Partial vs Total Participation

- An entity set may participate in a relation either totally or partially.
- **Total participation** means that every entity in the set is involved in the relationship
  - depicted as a double line.
- **Partial participation** means that not all entities in the set are involved in the relationship, e.g., not every professor guides a student
  - depicted by a single line.



# Weak Entity , Total Participation and Partial Key

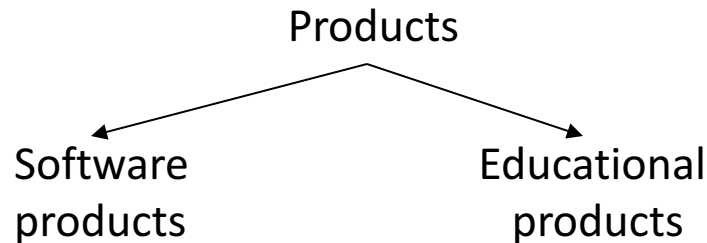


# HAS-A Relationships

- The relationships in the previous slides are called **HAS-A** relationships
- The term is used because each entity instance *has a* relationship to a second entity instance
  - An employee **has** a locker
  - A locker **has** an employee
- There are also **IS-A** relationships

# Modeling Subclasses

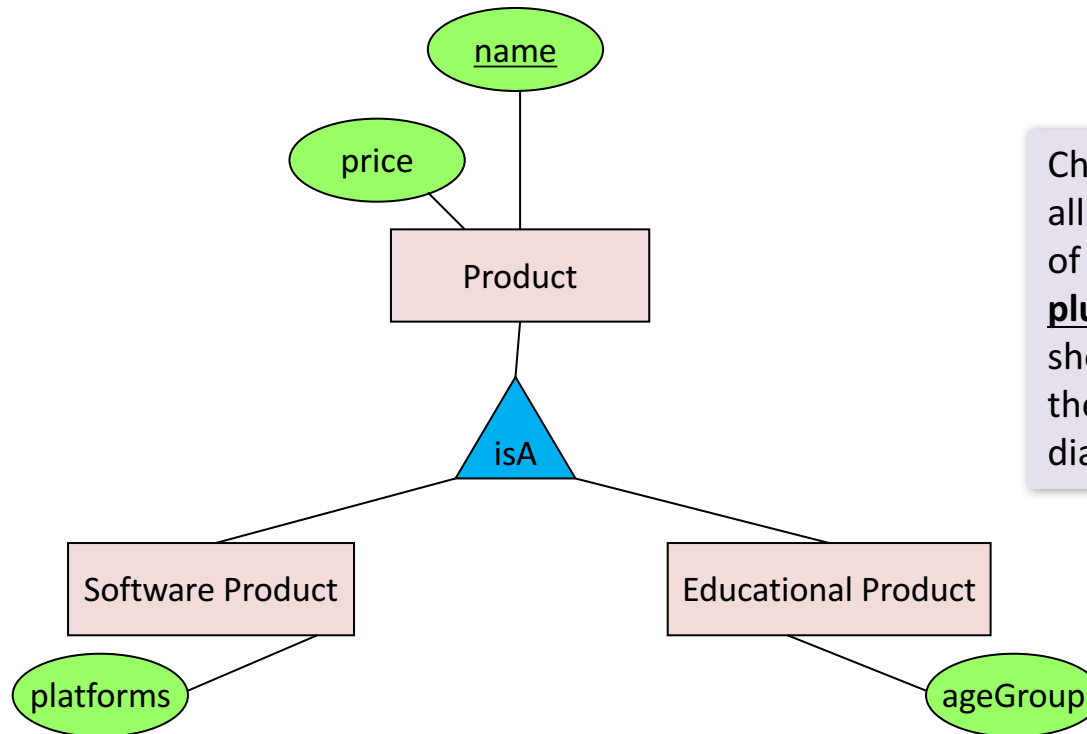
- Some objects in a class may be special, i.e. worthy of their own class
  - Define a new class?
    - *But what if we want to maintain connection to current class?*
  - Better: define a *subclass*
    - *Ex:*



We can define **subclasses** in E/R!



# Modeling Subclasses



Child subclasses contain all the attributes of *all* of their parent classes **plus** the new attributes shown attached to them in the E/R diagram

# Understanding Subclasses

- Think in terms of records; ex:

## – Product

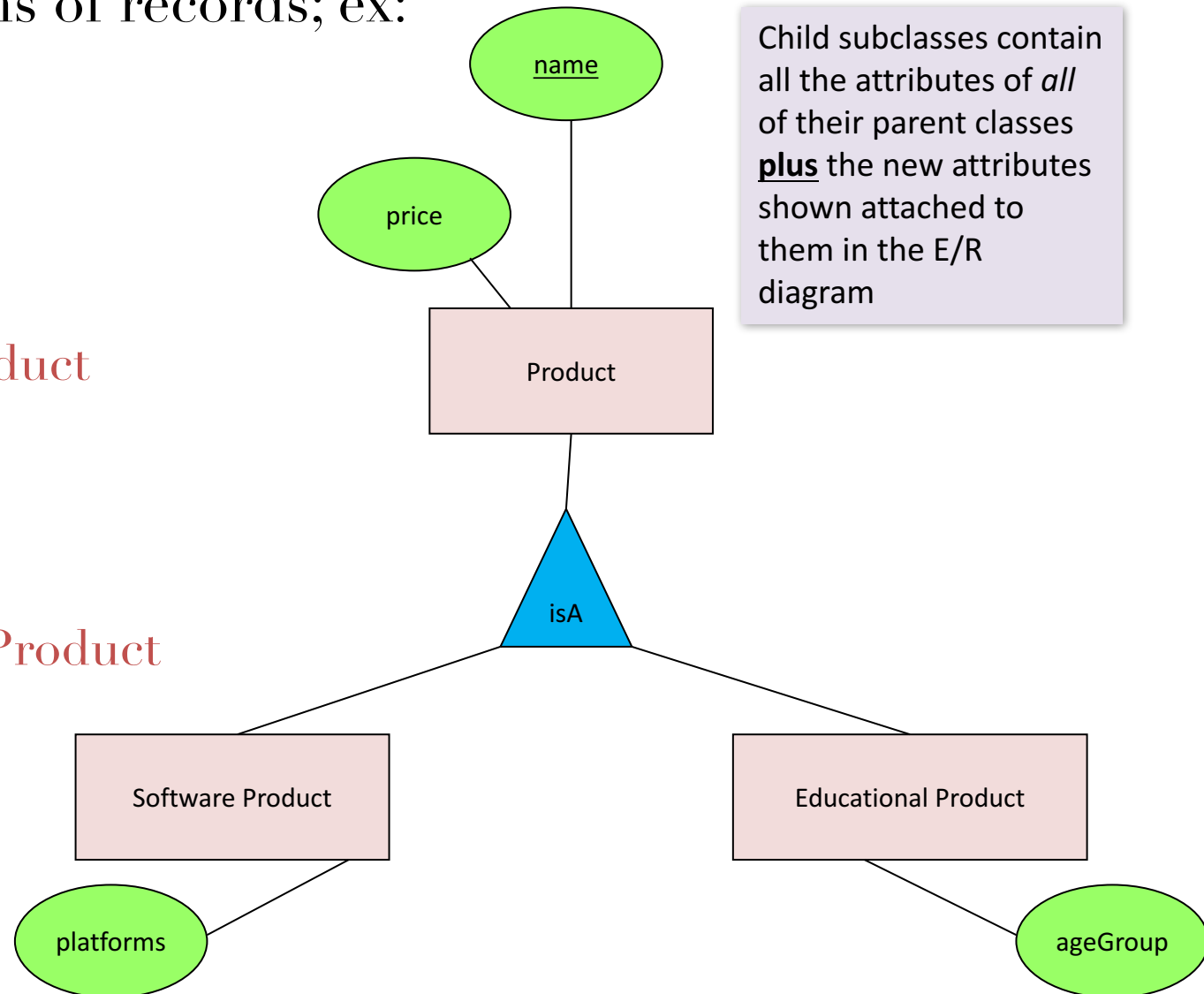
name
price

## – SoftwareProduct

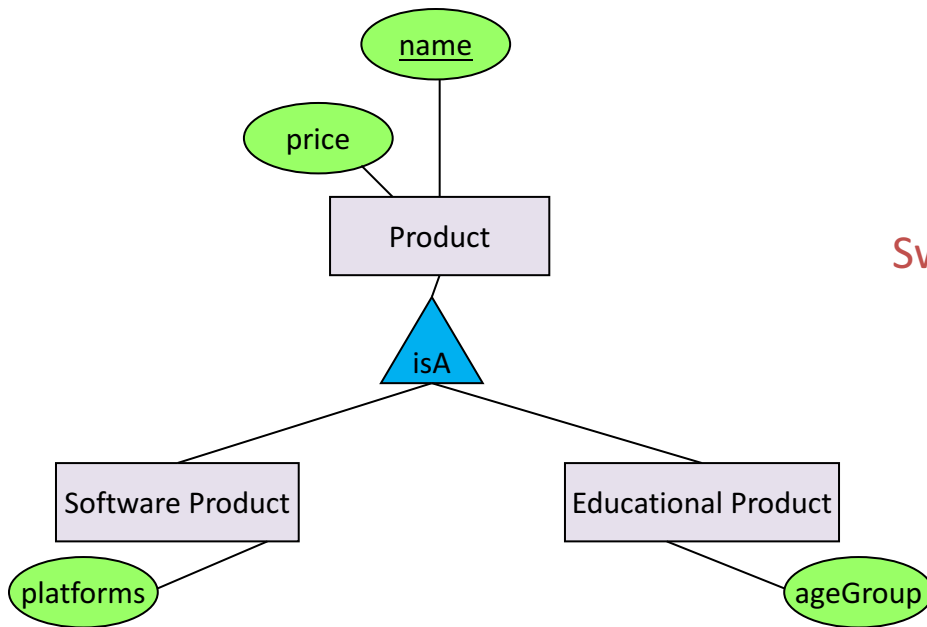
name
price
platforms

## – EducationalProduct

name
price
ageGroup



Think like tables...



Product

<u>name</u>	price	category
Gizmo	99	gadget
Camera	49	photo
Toy	39	gadget

Sw.Product








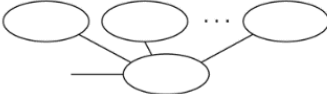



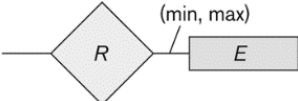
<u>name</u>	platforms
Gizmo	unix

Ed.Product

<u>name</u>	ageGroup
Gizmo	todler
Toy	retired

- Summary

**Figure 3.14**  
Summary of the  
notation for ER  
diagrams.

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of $E_2$ in $R$
	Cardinality Ratio 1: N for $E_1:E_2$ in $R$
	Structural Constraint (min, max) on Participation of $E$ in $R$

# ACTIVITIES

# DRAW AN E/R DIAGRAM FOR FOOTBALL

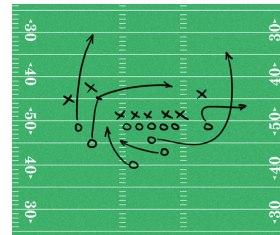
Use the following simplified model of a football season  
(concepts to include are underlined):



Teams play each other in Games.  
Each pair of teams can play each other multiple times



Players belong to Teams



A Game is made up of Plays that result in a yardage gain/loss, and potentially a touchdown



A Play will contain either a Pass from one player to another, or a Run by one player

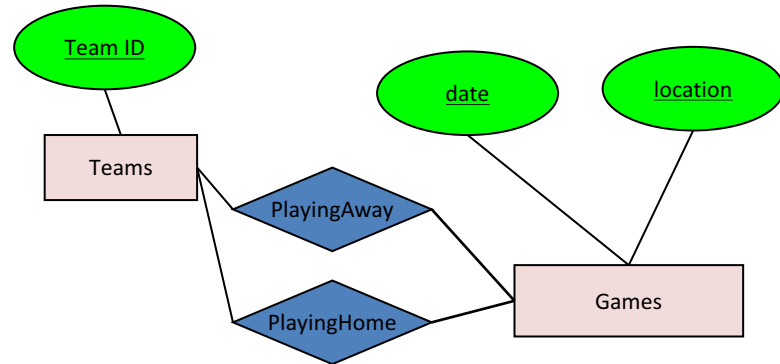
Note that various ER diagrams could work, not just the following one!

## **ACTIVITY**

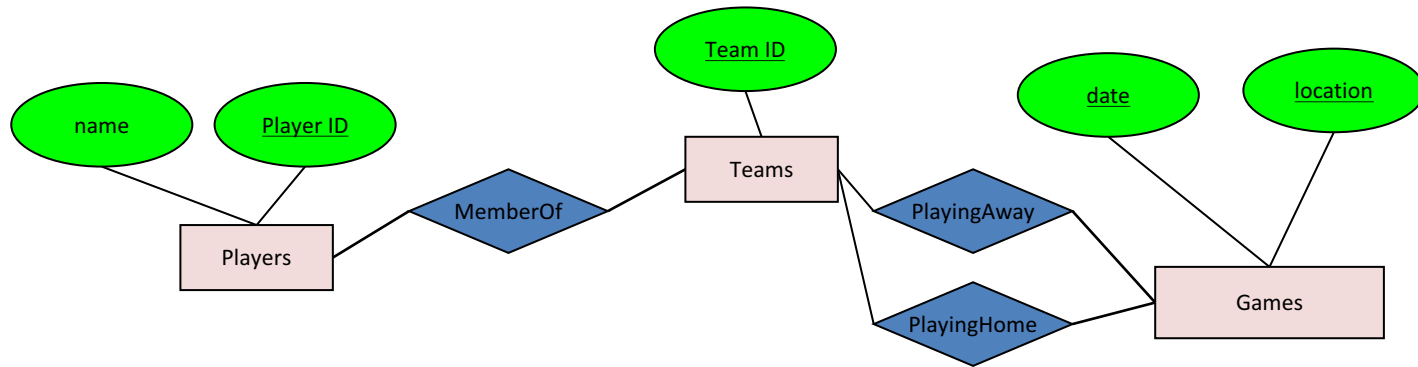
Note two copies of  
the Teams entity  
here!



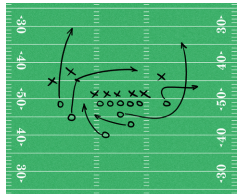
Teams play each  
other in Games.  
Each pair of teams  
can play each other  
multiple times



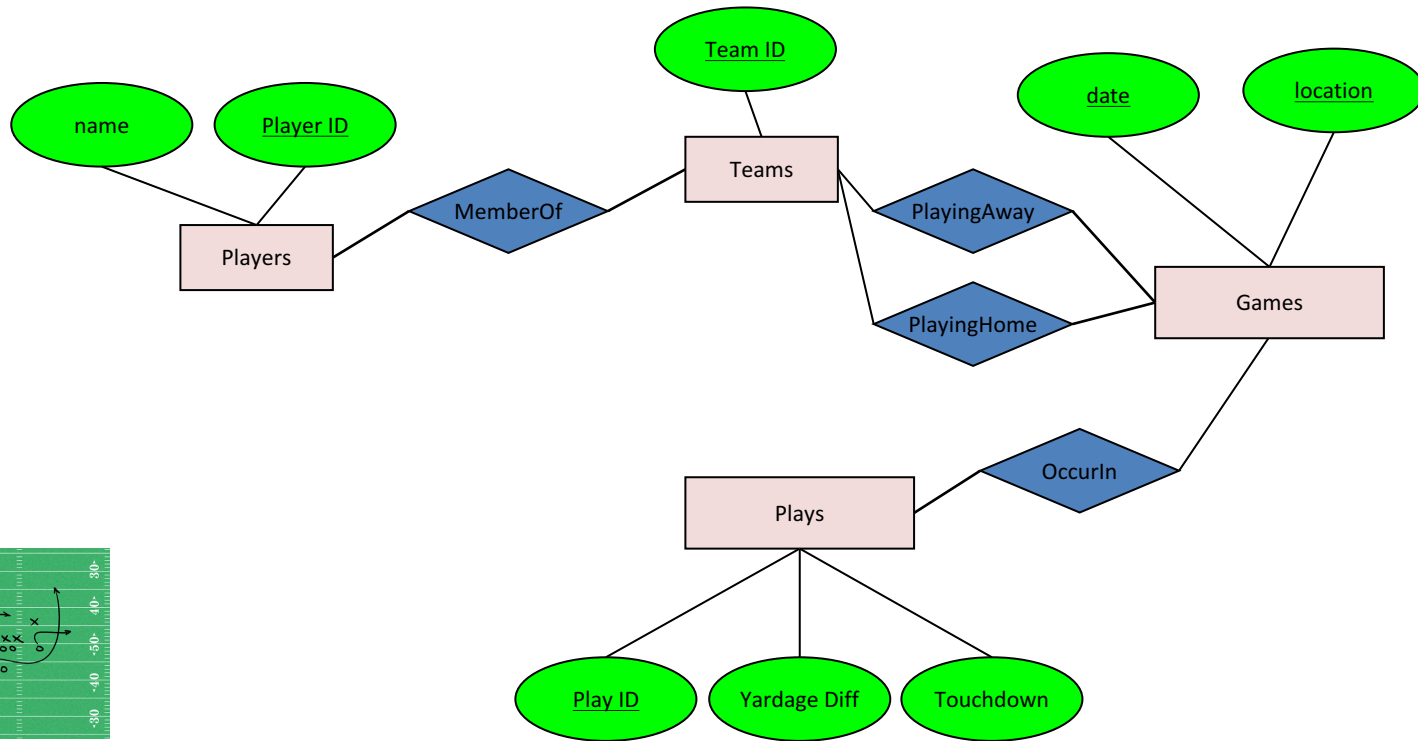


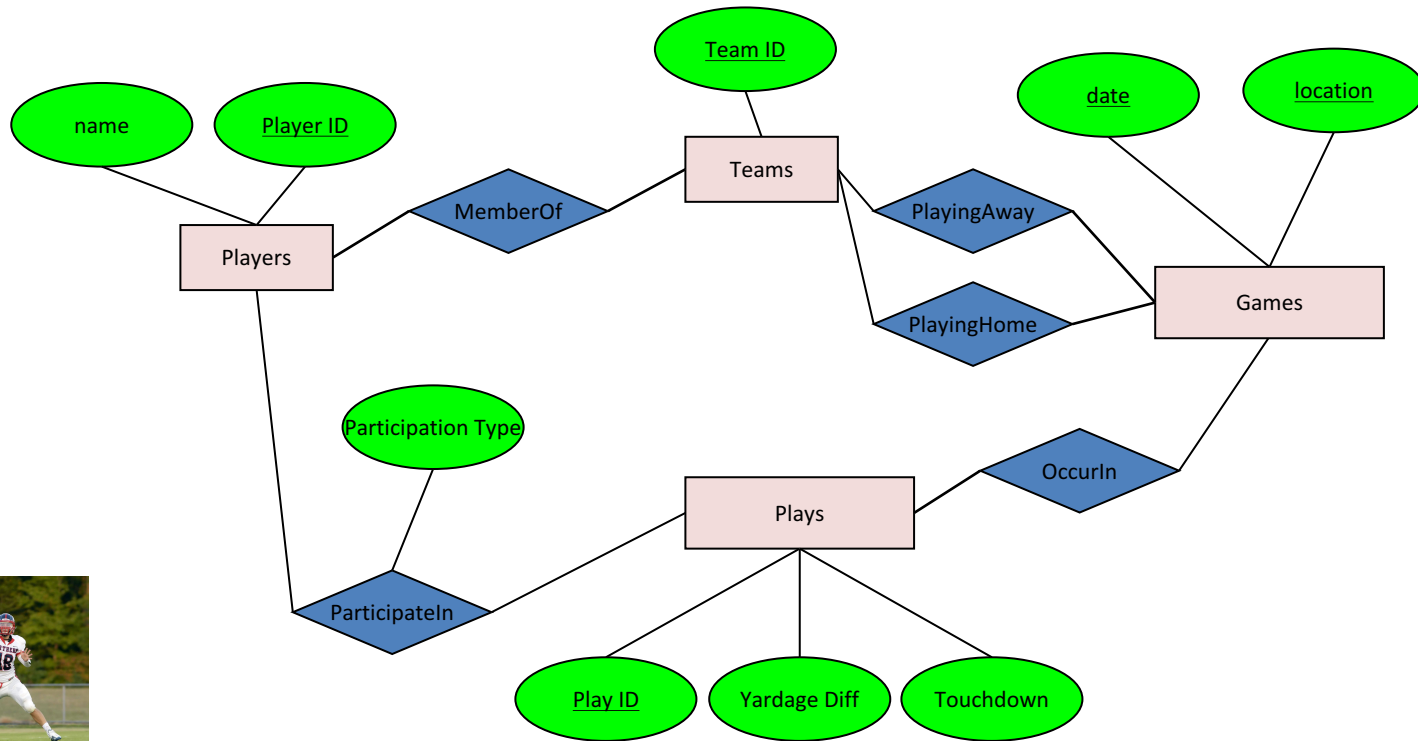


Players belong to  
Teams (assume no  
trades / changes)



A Game is made up of **Plays** that result in a yardage gain/loss, and potentially a touchdown





A Play will contain either a **Pass** from one player to another, or a **Run** by one player

Note that various ER diagrams could work, not just the following one!

## **ACTIVITY 2**

# ENHANCE YOUR E/R DIAGRAM!

Also make sure to add (new concepts underlined):



A player can only belong to one team, a play can only be in one game, a pass/run..?



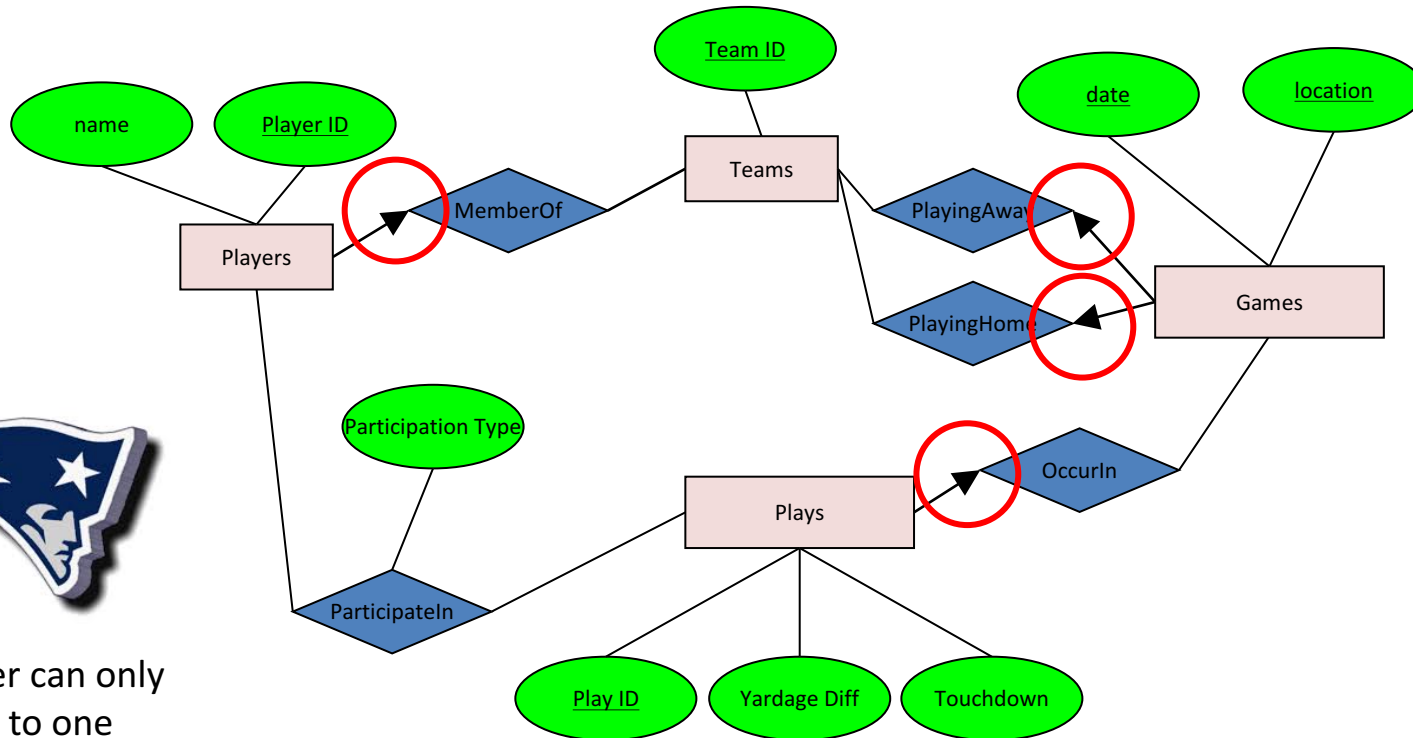
Players can achieve a Personal Record linked to a specific Game and Play

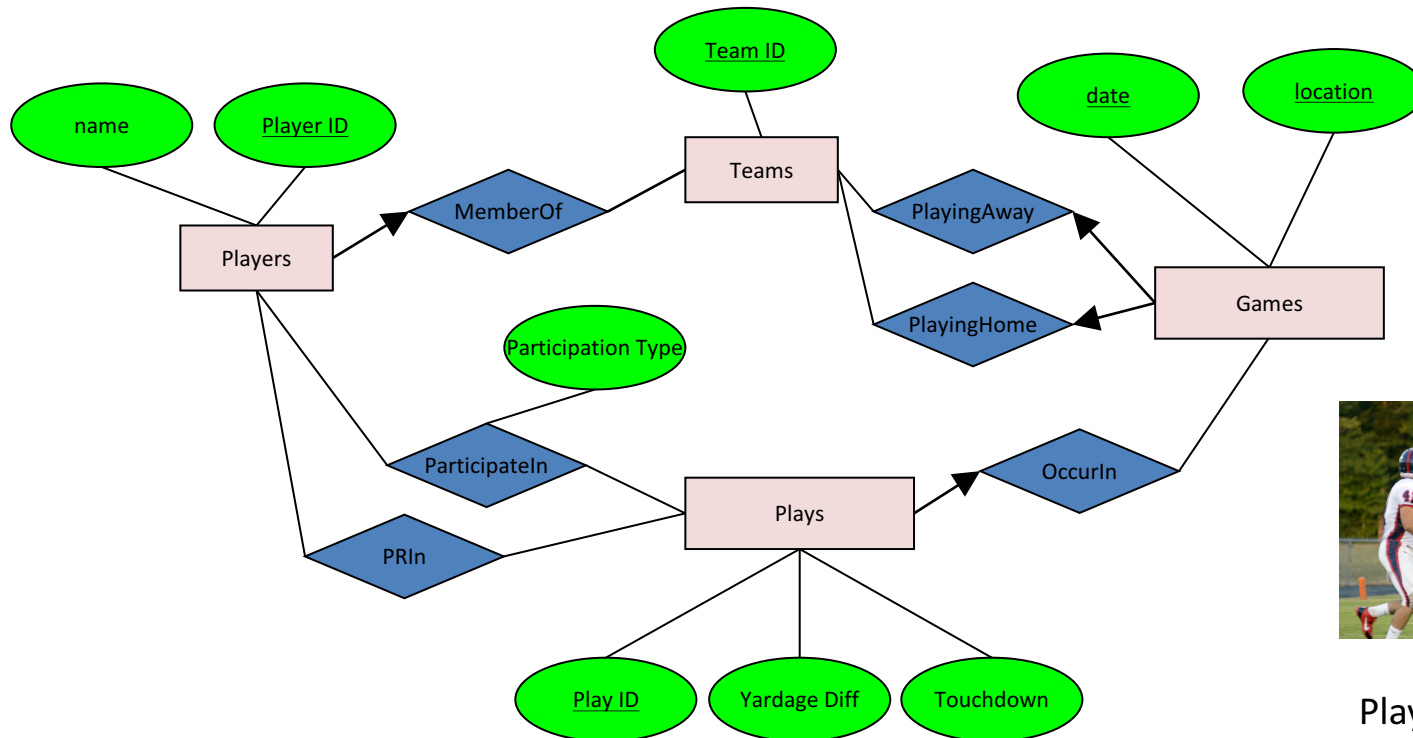


Players have a weight which changes in on vs. off-season

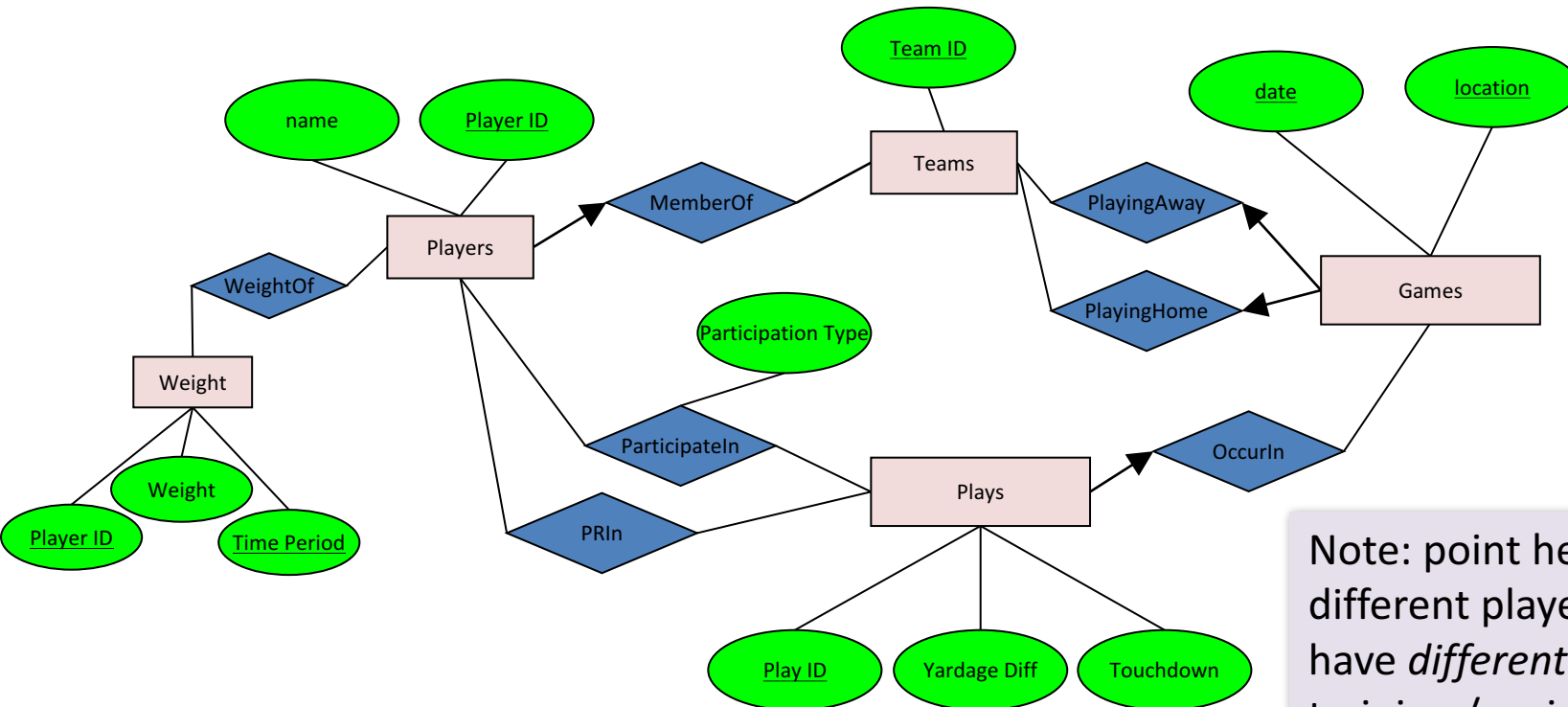


A player can only belong to one team, a play can only be in one game, a pass/run..?





Players can achieve a **Personal Record** linked to a specific Game and Play



Players might have different weights at different times

Note: point here is that different players might have *different numbers* of training / weight phases- hence should represent as new entity!



Note that various ER diagrams could work, not just the following one!

## **ACTIVITY 3**

# ADD IN: SUBCLASSES, CONSTRAINTS, AND WEAK ENTITY SETS

Concepts to include / model:



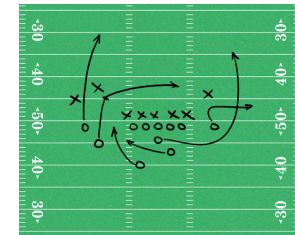
Teams belong to cities- model as ***weak entity sets***



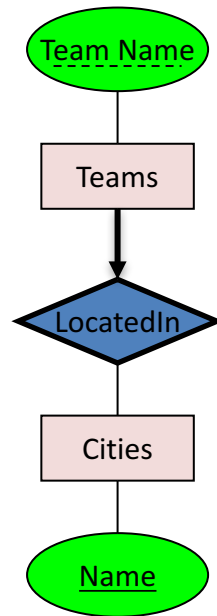
Players are either on Offense or Defense, and are of types (QB, RB, WR, TE, K)



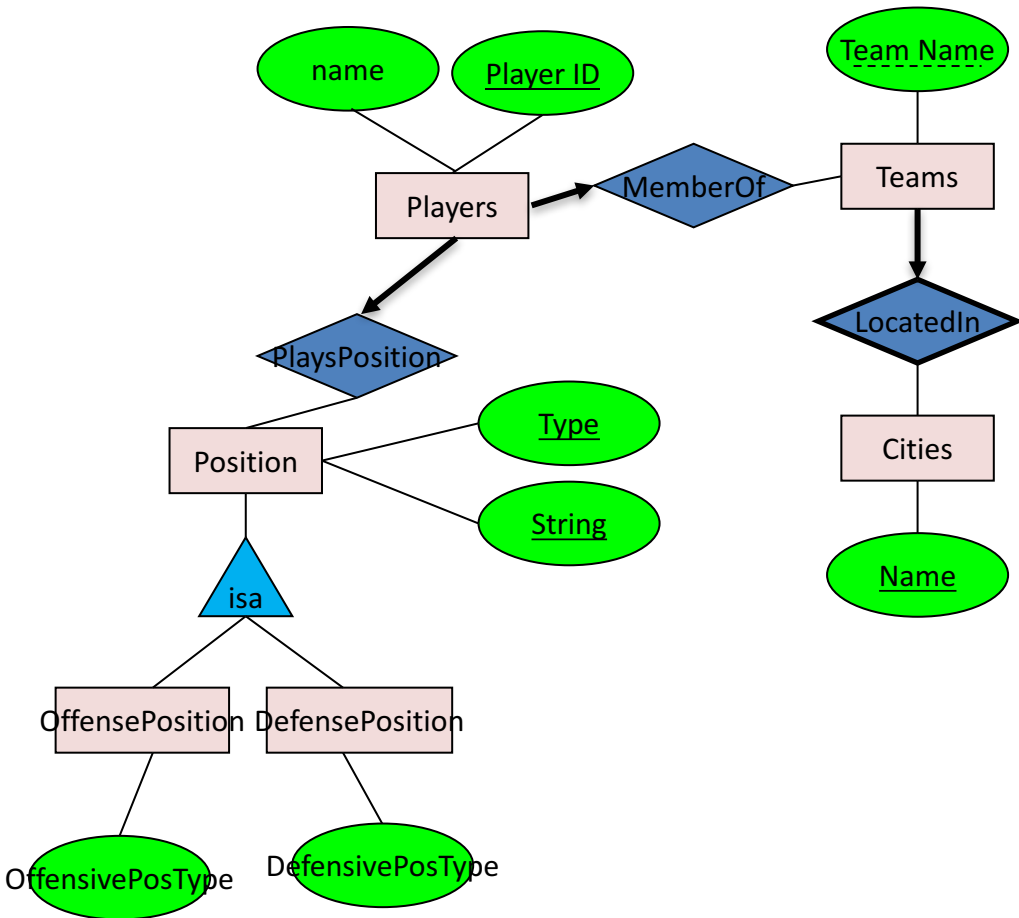
All passes are to exactly one player; all runs include a player



Make sure you have designated keys for all our concepts!



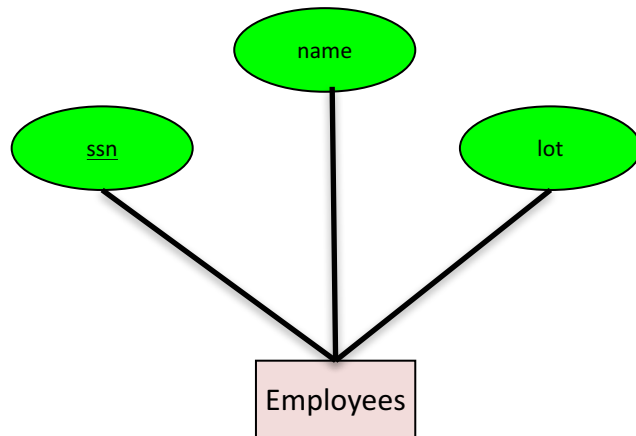
Teams belong to  
cities- model as  
***weak entity sets***



Players are either on Offense or Defense, and are of types (QB, RB, WR, TE, etc.)

## Design Theory (ER model to Relations)

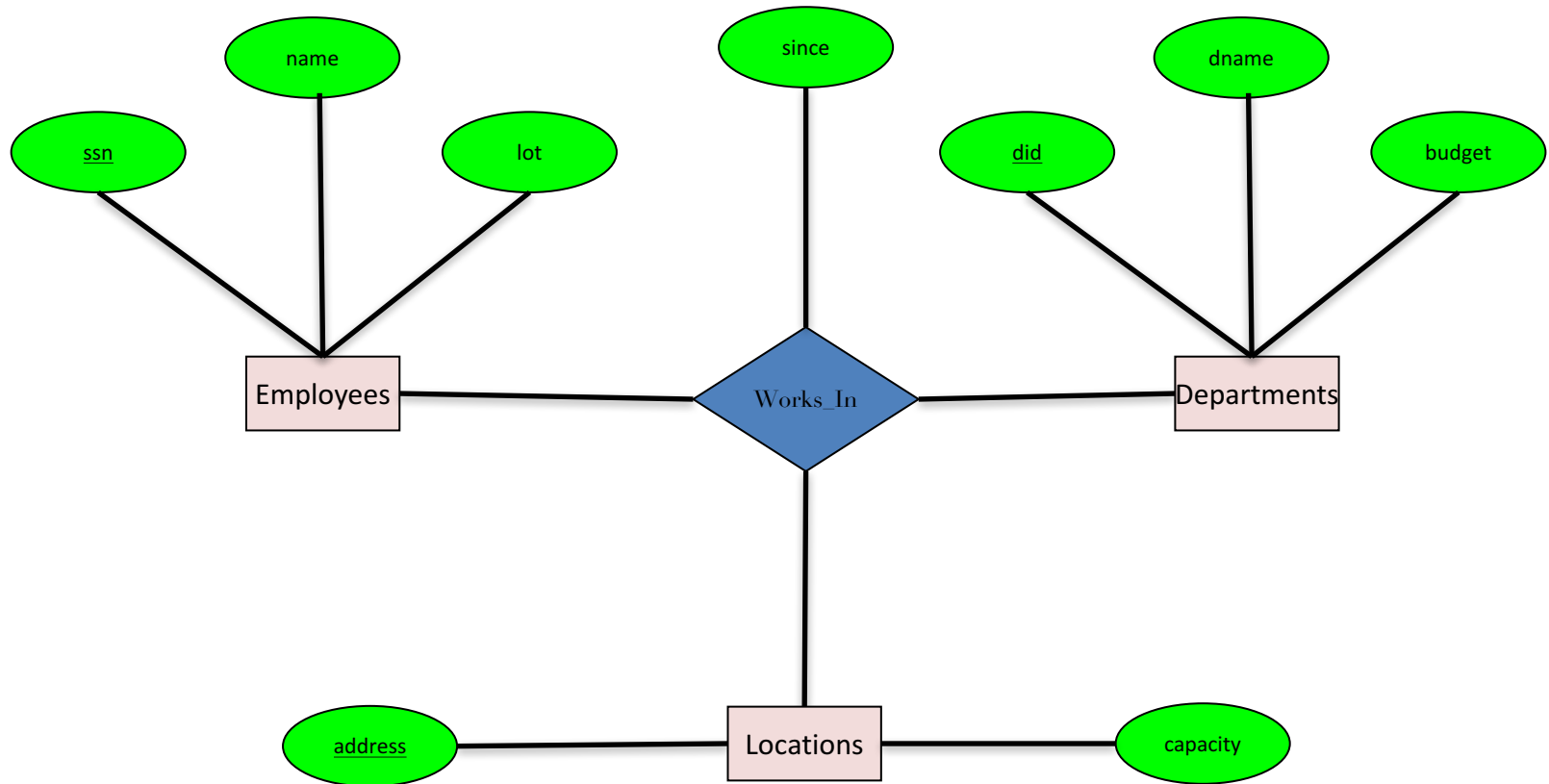
# Entity Sets to Tables



ssn	name	lot
123-22-3333	Alex	23
234-44-6666	Bob	44
567-88-9787	John	12

```
CREATE TABLE Employees (    ssn char(11),
                             name varchar(30),
                             lot Integer,
                             PRIMARY KEY (ssn))
```

# Relationship Sets (without Constraints) to Tables

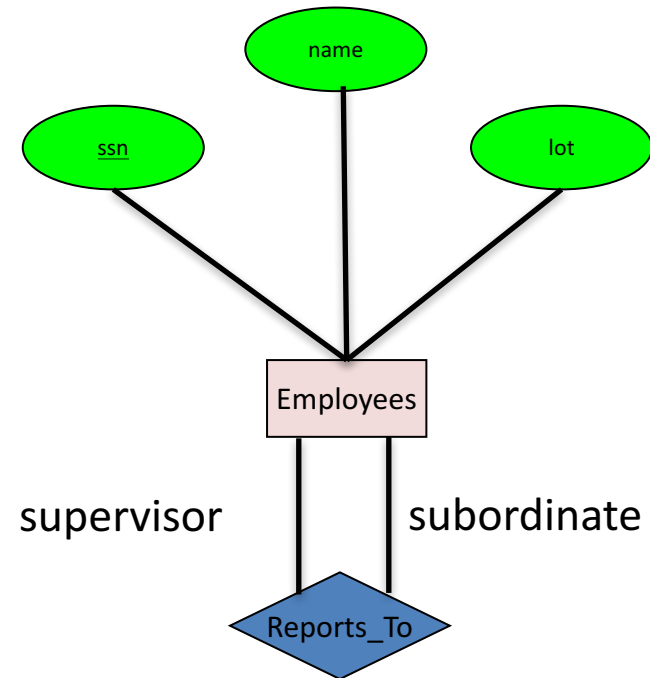


# Relationship Sets (without Constraints) to Tables

```
CREATE TABLE Works_in(      ssn char(11),
                                did integer (30),
                                address varchar(30),
                                since date,
                                PRIMARY KEY (ssn, did, address),
                                FOREIGN KEY (ssn) REFERENCES Employees,
                                FOREIGN KEY (address) REFERENCES Locations,
                                FOREIGN KEY (did) REFERENCES Departments,
                                )
```

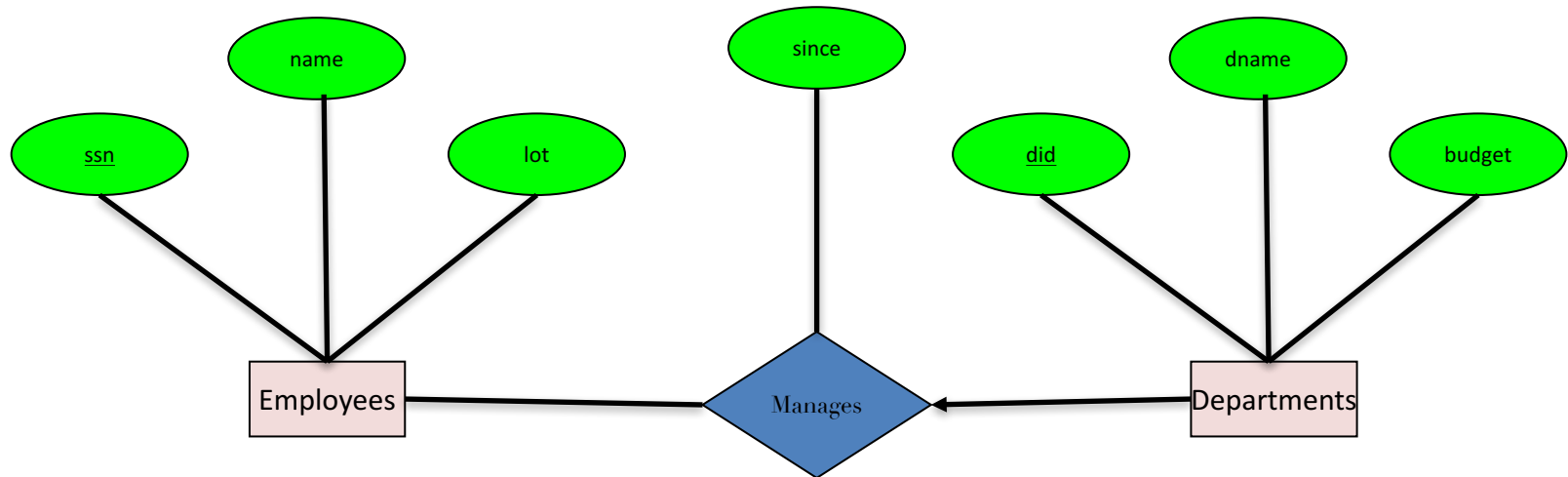


# Relationship Sets (without Constraints) to Tables



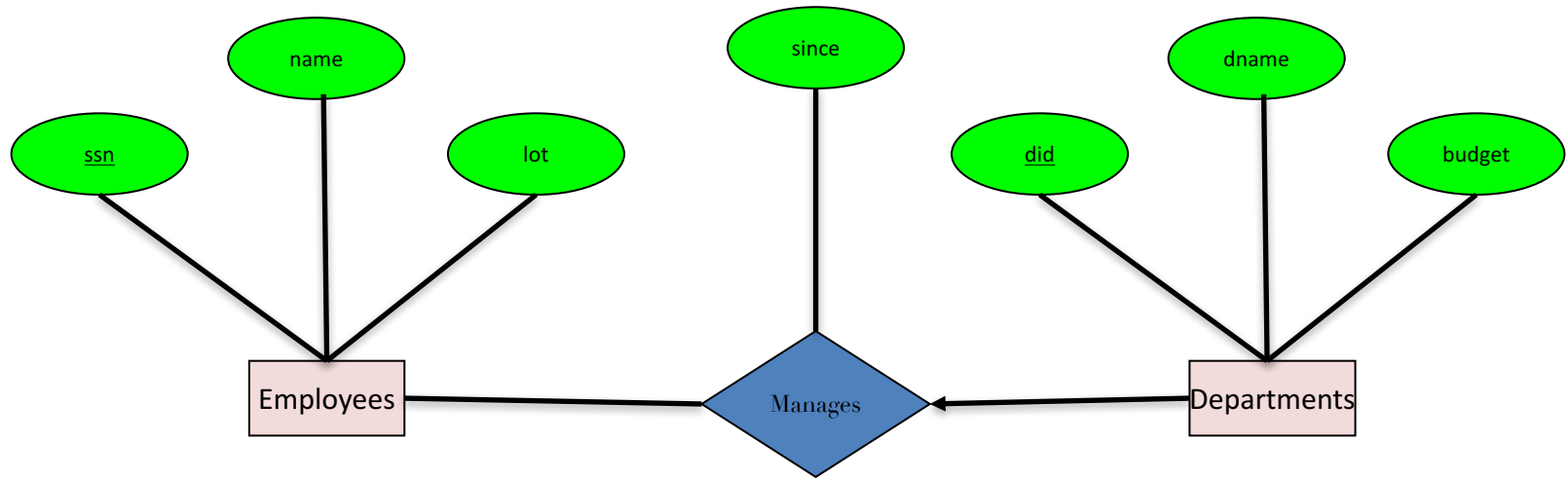
```
CREATE TABLE Reports_To (  
    supervisor_ssn char(11),  
    subordinate_ssn char(11),  
    PRIMARY KEY (supervisor_ssn,  
                subordinate_ssn),  
    FOREIGN KEY (supervisor_ssn )  
        REFERENCES Employees(ssn),  
    FOREIGN KEY (subordinate_ssn )  
        REFERENCES Employees(ssn)  
)
```

# Relationship Sets (with key Constraints) to Tables



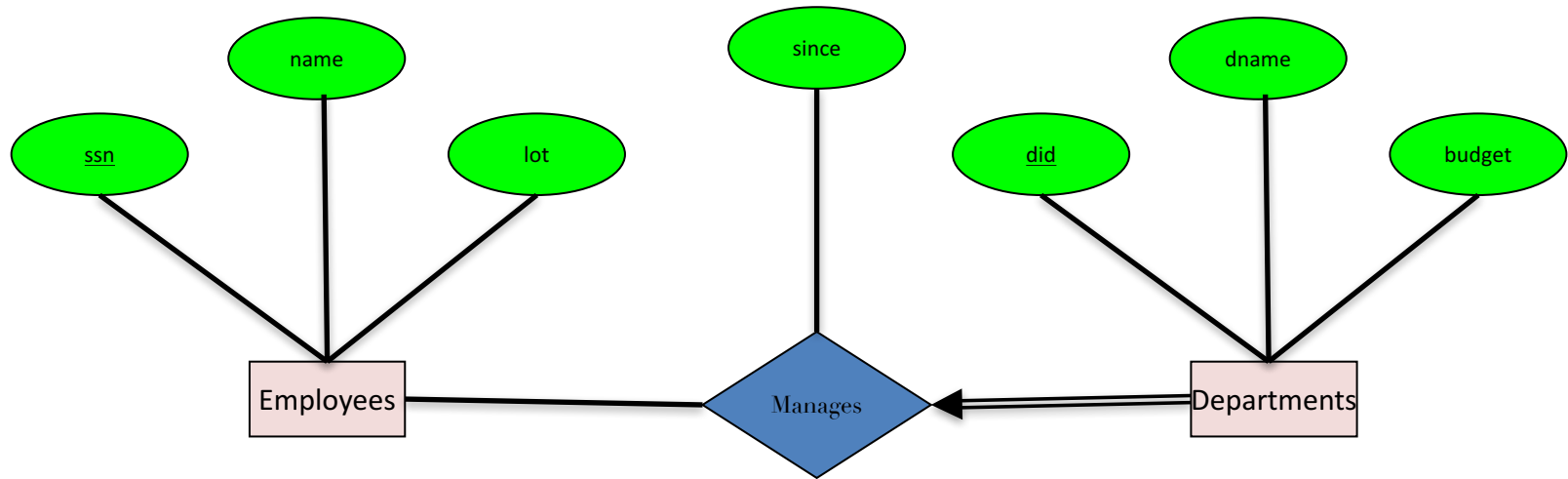
```
CREATE TABLE Manages (  
    ssn char(11),  
    did integer (30),  
    since date,  
  
    PRIMARY KEY (did),  
    FOREIGN KEY (ssn) REFERENCES Employees,  
    FOREIGN KEY (did) REFERENCES Departments,  
)
```

# Better way of doing it



```
CREATE TABLE Dept_Mgr (  
    did integer (30),  
    dname varchar(30),  
    budget float(30),  
    ssn char(11),  
    since date,  
  
    PRIMARY KEY (did),  
    FOREIGN KEY (ssn) REFERENCES Employees,  
)
```

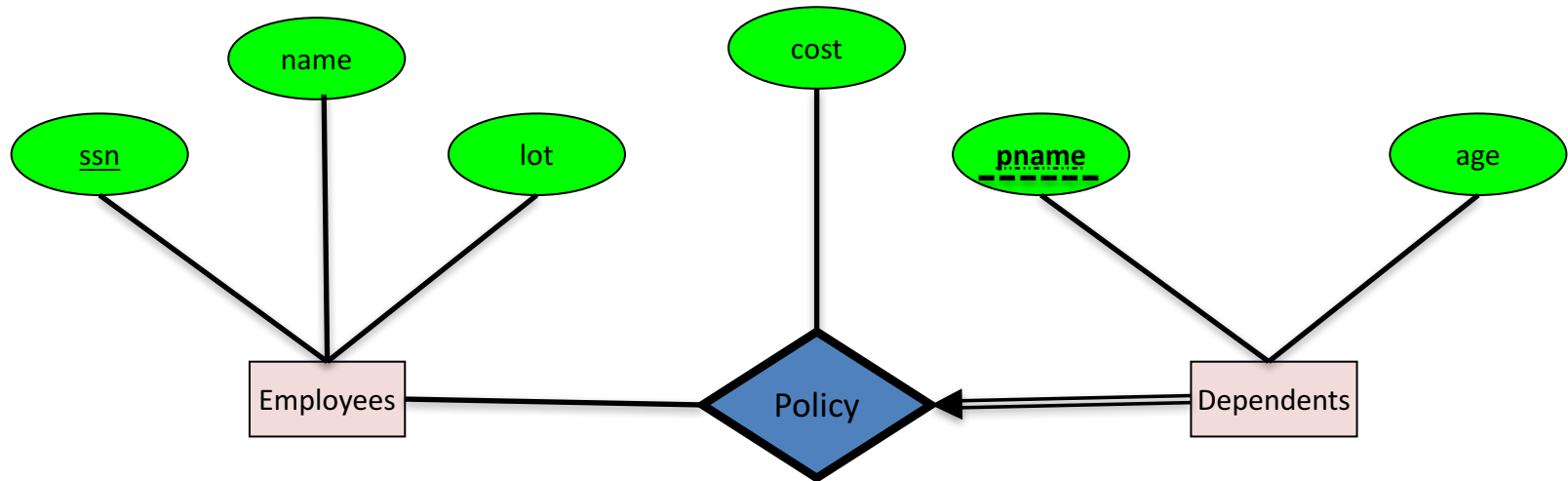
# Relationship Sets (with Participation Constraints) to Tables



```
CREATE TABLE Dept_Mgr (
    did integer (30),
    dname varchar(30),
    budget float(30),
    ssn char(11),
    since date,

    PRIMARY KEY (did),
    FOREIGN KEY (ssn) REFERENCES Employees ON
    DELETE NO ACTION
)
```

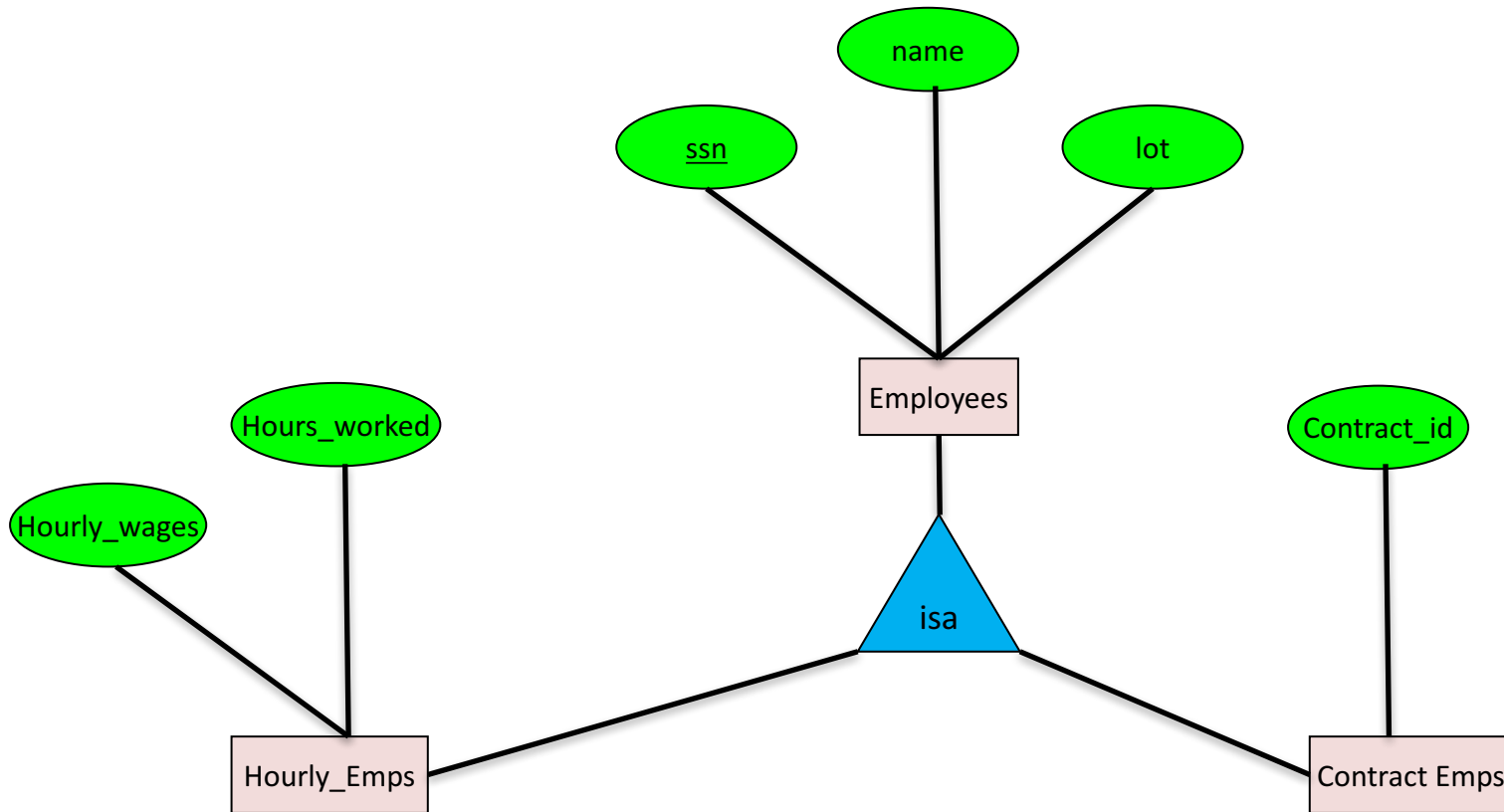
# Translating Weak Entity Sets



```
CREATE TABLE Dept_Policy(  pname varchar(30),
                           age integer,
                           cost float,
                           ssn char(11),

                           PRIMARY KEY (pname, ssn),
                           FOREIGN KEY (ssn) REFERENCES Employees
ON DELETE CASCADE
)
```

# Translating Class Hierarchies



## Two options

1. We can map each of the entity sets Employees, Hourly\_Emps, and Contract\_Emps to a distinct relation.
2. We can create just two relations, corresponding to Hourly\_Emps and Contract\_Emps

Both have their pros and cons

- Redundancy
- Performance

# E/R Summary

- E/R diagrams are a visual syntax that allows technical and non-technical people to talk
  - For conceptual design
- Basic constructs: **entity**, **relationship**, and **attributes**
- A good design is faithful to the constraints of the application, but not overzealous



# Scenario

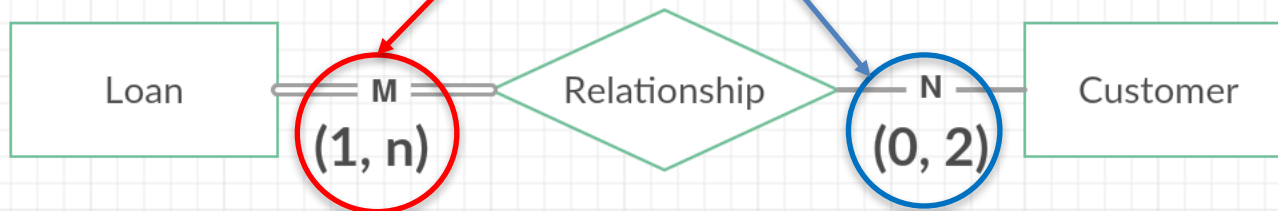
- One customer can have **at max 2 loans**. One loan can be given to **multiple** customers.

What it really means:

- One customer can have **(0,2)** loans
- One loan can be given to **(1,n)** customer
- This is a many to many scenario



Crow's foot Notation



Chen Notation

# Acknowledgement

- Some of the slides in this presentation are taken from the slides provided by the authors.
- Many of these slides are taken from cs145 course offered by Stanford University.
- Thanks to YouTube, especially to [Dr. Daniel Soper](#) for his useful videos.