CS 5530

Database Systems
Spring 2020

Entity Relationship Model

Announcements

- Office hours:
  - Mon - Thurs: 2:15p – 3:15p
- CADE help hours posted

Keys

- Remember: keys define the data, not the other way around
  - Sometimes (like in HW), we give backwards exercises

Keys

- In general, we can not infer keys from instance

A_1	A_2	A_3
x	4	q
y	4	p
x	3	x
b	7	q

- Keys apply to **all possible instances**

Foreign Key

- **Foreign Key**:
  - Attribute whose values are a key in another table
  - Think of it as a "pointer"

$\{a, b\}$

$\{a\}$

$\{b\}$

Key

Students

sID	Name	GPA
1	Harry	3.5
2	Hermione	4.0
3	Ron	4.0
4	Malfoy	3.9

Foreign Key

Enrolled

sID	cID	Grade
4	CS3810	B-
3	CS4400	A-
2	CS6016	A+
2	CS3500	A+

A	B	C
1	X	
1	Y	
2	X	

$\{A, B\}$

$\{A\}$

Database Design Steps

1. Requirements Analysis
    - What does user need? What must it do?

Database Design Steps

1. Requirements Analysis
   - What does user need? What must it do?

2. Conceptual Design
   - High level formal description

Database Design Steps

1. Requirements Analysis
   - What does user need? What must it do?

2. Conceptual Design
   - High level formal description

3. Schema Refinement
   - Consistency and normalization

Database Design Steps

1. Requirements Analysis
   - What does user need? What must it do?

2. Conceptual Design
   - High level formal description

3. Schema Refinement
   - Consistency and normalization

4. Physical Design
   - Indexes, disk layout

Database Design Steps

1. Requirements Analysis
    - What does user need? What must it do?

2. Conceptual Design
    - High level formal description

3. Schema Refinement
    - Consistency and normalization

4. Physical Design
    - Indexes, disk layout

5. Security Design
    - Who accesses it, and how?

Database Design Steps

1. Requirements Analysis
   - What does user need? What must it do?

2. Conceptual Design
   - High level formal description

•Using Entity-Relationship (ER) model

Problem → Solution

- Program construction

Problem specification → **C**++ → assembly

Problem → Solution

- Program construction

    Problem specification → **C++** → assembly

- Database construction

    Problem specification → **ER Model** → schemas

Problem → Solution

- Program construction

  Problem specification → **C++** → assembly

- Database construction

  Problem specification → **ER Model** → schemas

- In both of these cases, you *could* skip the middle step

Problem → Solution

- Program construction

  Problem specification → **C++** → assembly

- Database construction

  Problem specification → **ER Model** → schemas

- In both of these cases, you *could* skip the middle step

  - In both cases, that would be a bad idea

Problem → Solution

- There is a mechanical (algorithmic) translation to the final result

ER Model

- What are the **entities**, and their **relationships**?

ER Model

- What are the **entities**, and their **relationships**?

- For the remainder of today, **forget about tables**!

Entity

- A real-world object, distinguishable from other entities
  - `{u0123456, “Danny”}`

- An entity is described by a set of **attributes**
  - `(uID string, name string)`

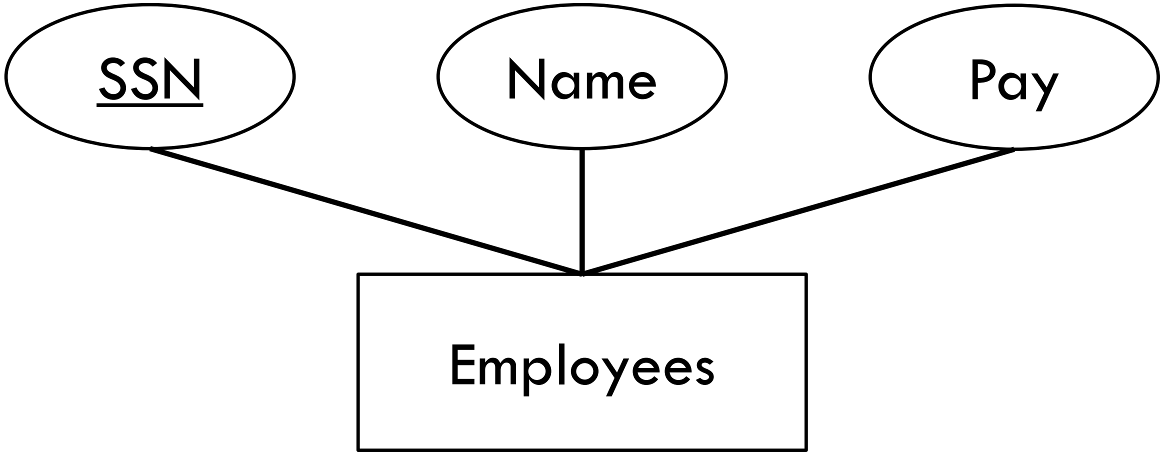
Entity Set

- A collection of entities of the same type, e.g.
  - All students
  - All buildings
  - All people

- All entities in the set have the same attributes

- An entity set has a key attribute

ER Diagram

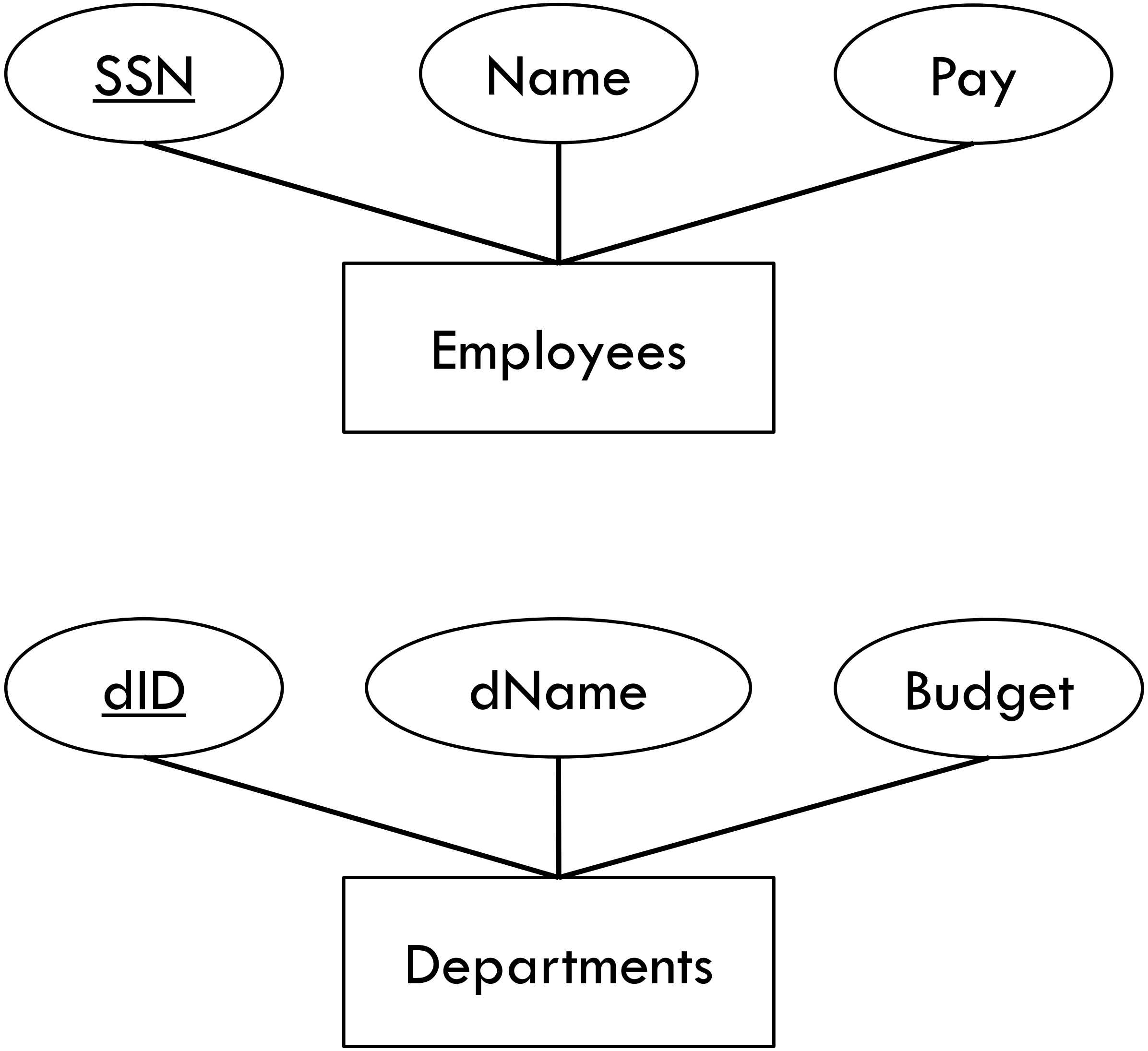


□ = entity set

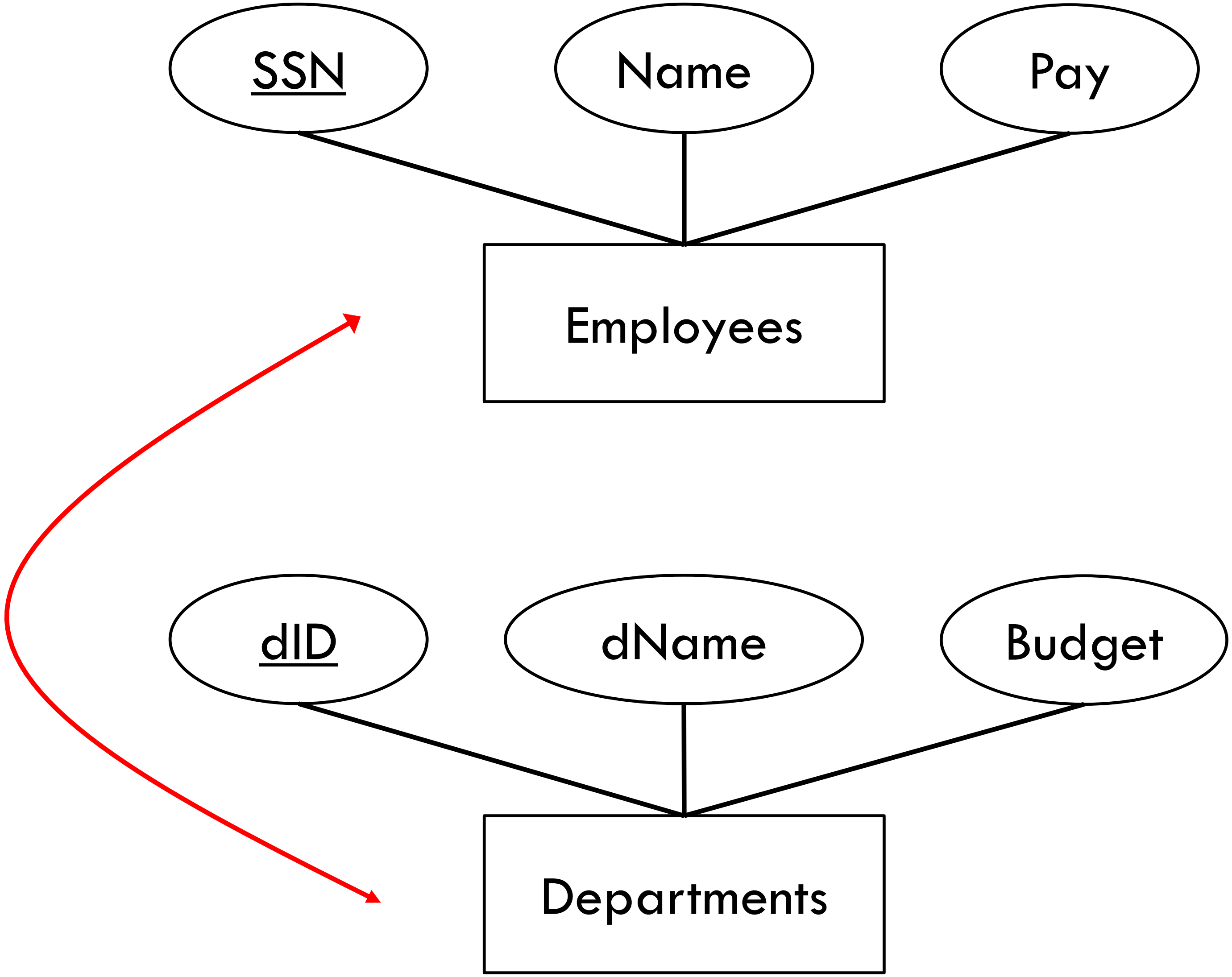
○ = attribute

underline = part of primary key

ER Diagram



ER Diagram



What about relationships?

Relationship

- Relationship between 2 or more entities:
  - “Danny works in School of Computing”

entity relationship entity

Relationship

- Relationship between 2 or more entities:
  - “Danny works in School of Computing”

    entity relationship entity

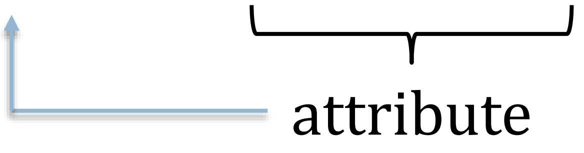
- Relationship Set:
  - Set of relationships between entities of same type
  - e.g. works in relates Employees to Departments

Relationship Attributes

- Relationships can have attributes as well:
  - Danny works in SoC $\underbrace{\text{since 2010}}_{\text{attribute}}$

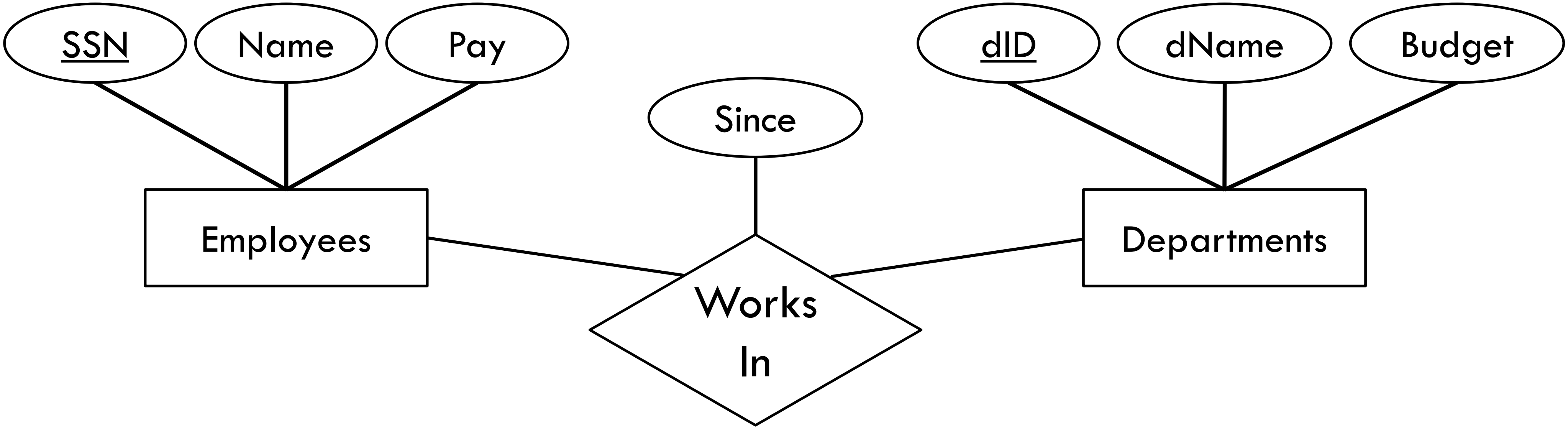
Relationship Attributes

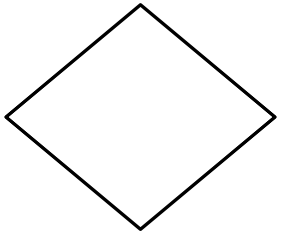
- Relationships can have attributes as well:
  - Danny works in SoC since 2010



- Starting date does not belong to Danny or SoC
  - It belongs to the relationship

ER Diagram

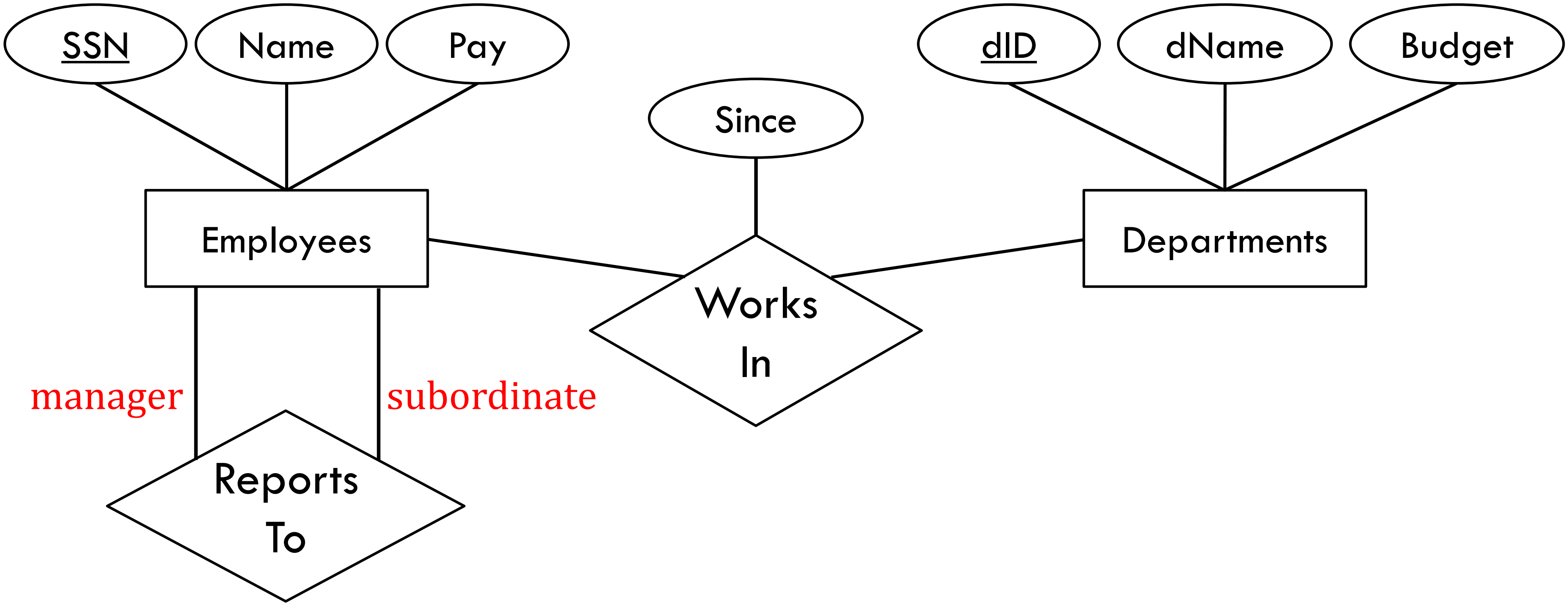




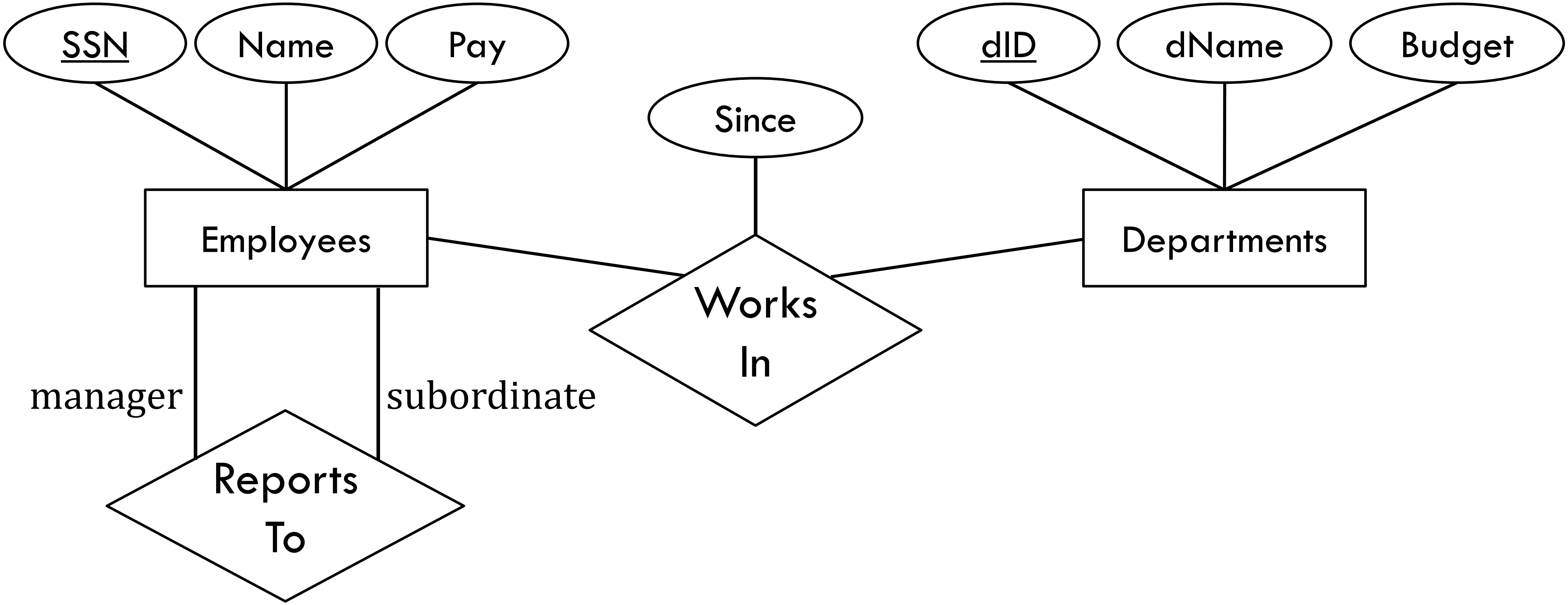
= relationship set

relationship sets do not need a primary key

Inter-Relationship

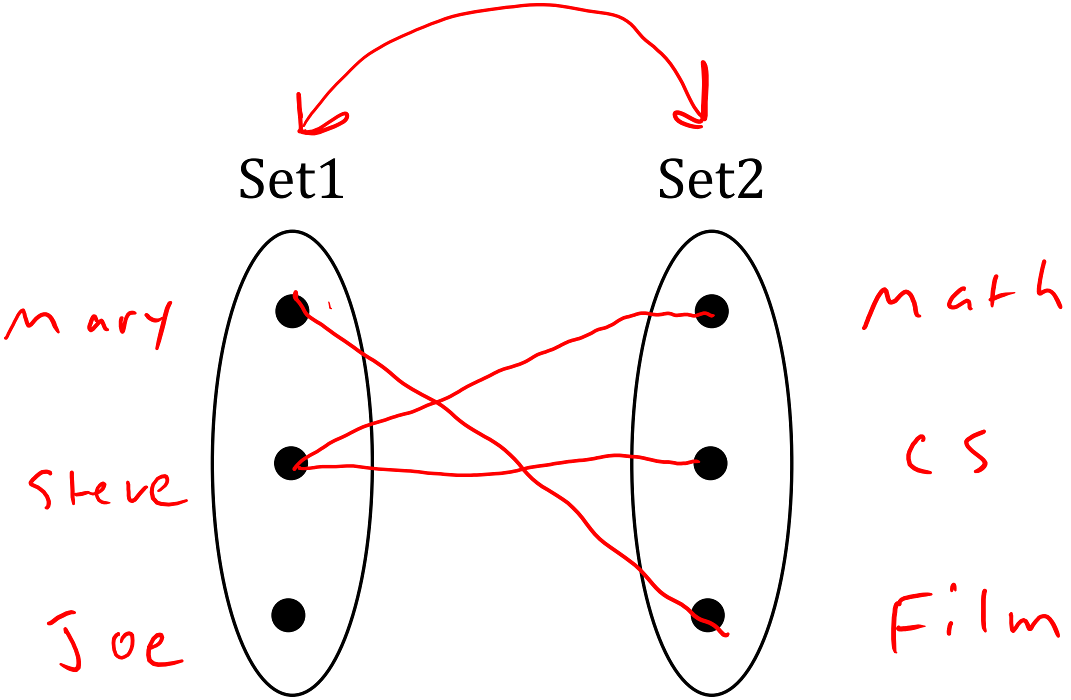


Cardinality



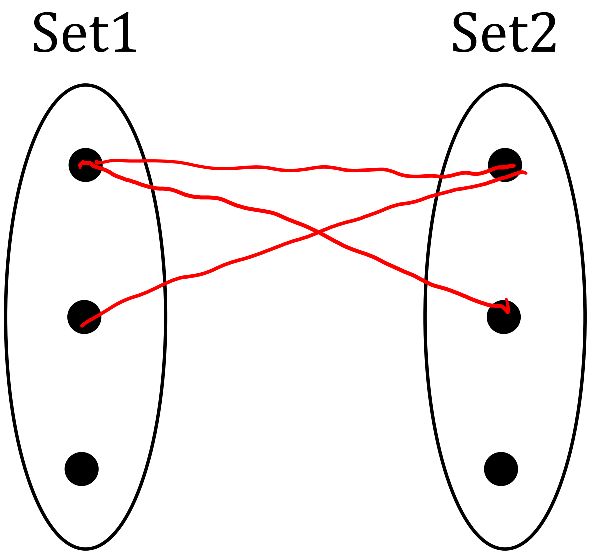
- Can an employee work in multiple departments?
- Can a department have multiple employees?

Cardinality



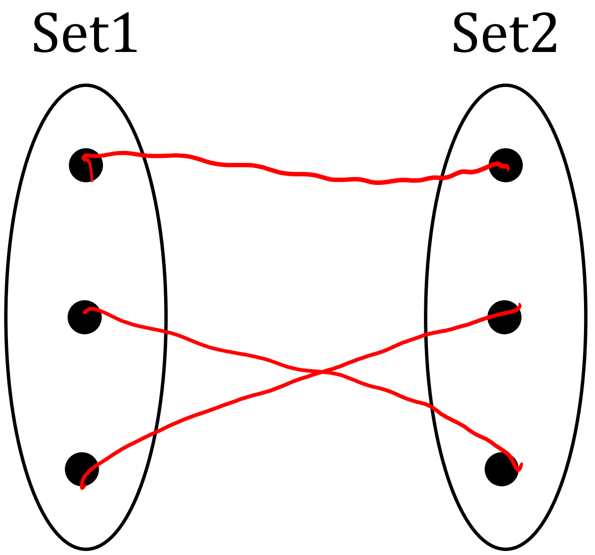
1 - to - many

Cardinality



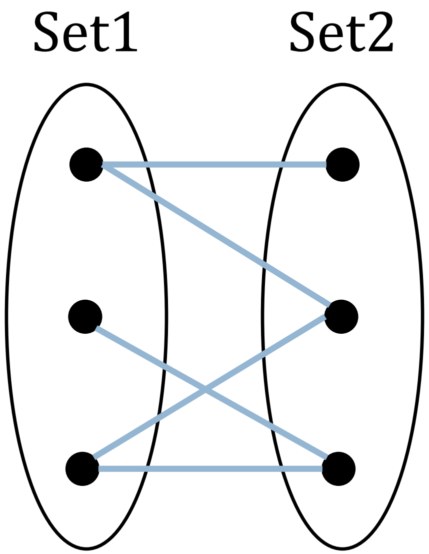
M - to - M

Cardinality

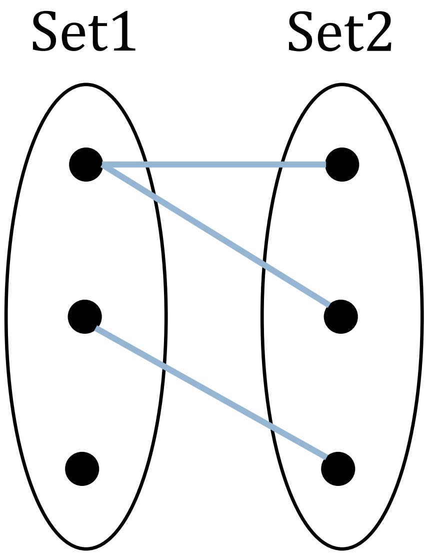


1-to-1

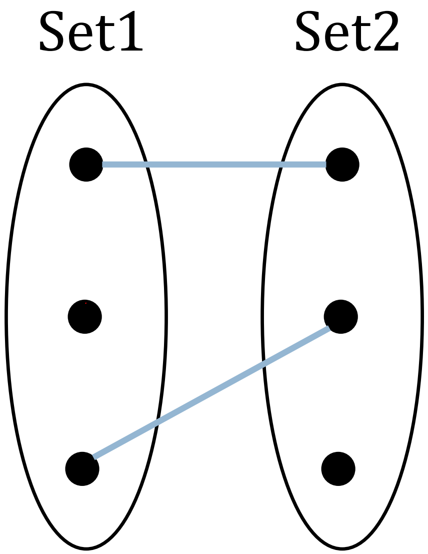
Cardinality



Many-to-Many



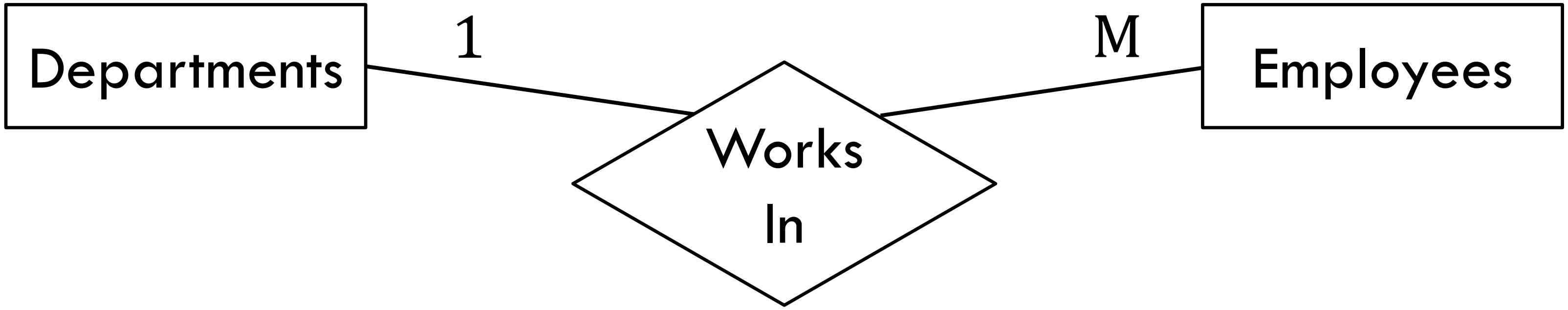
One-to-Many



One-to-One

Cardinality

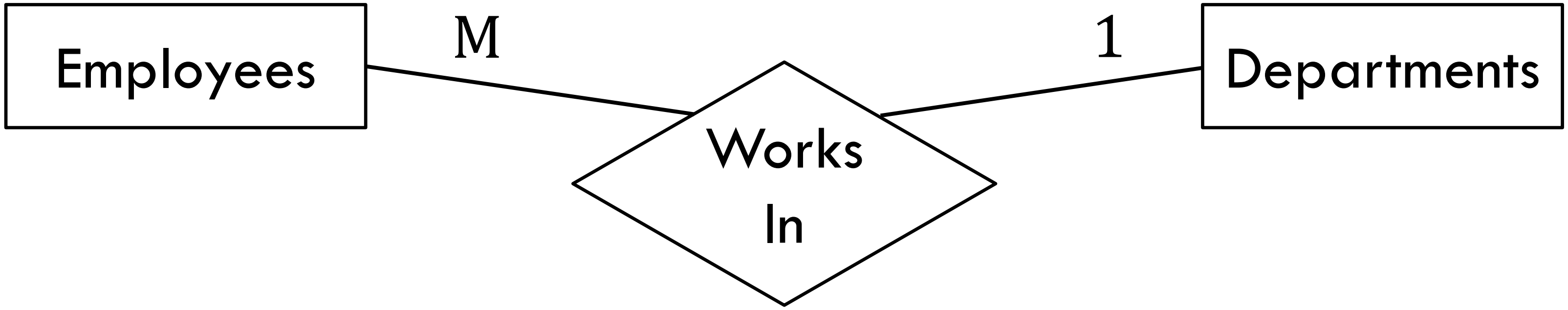
- Annotate opposite edge of relationship with cardinality



- "A department can have many employees, and an employee can work for one department"
  - 1-to-Many

Cardinality

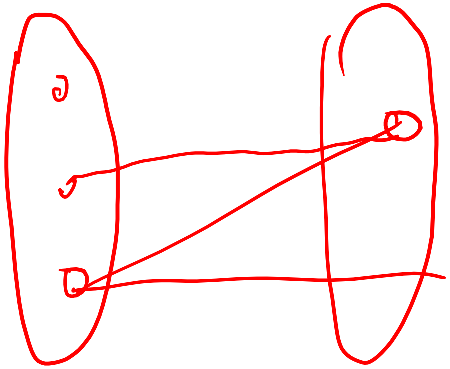
- They can be drawn in either direction
  - Still call it 1-to-M

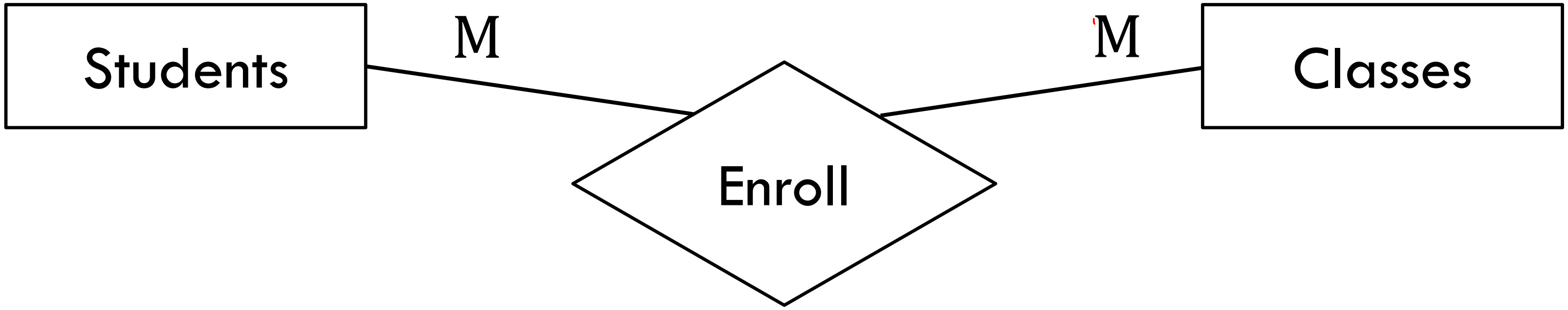


- "A department can have many employees, and an employee can work for one department"
  - 1-to-Many

Cardinality

- Many-to-many





Students

M

Enroll

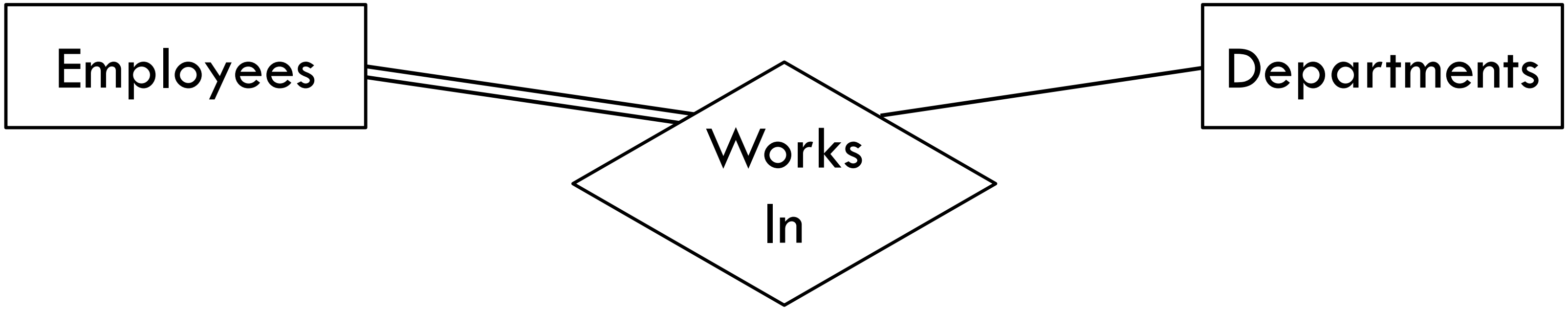
M

Classes

- "A student can take multiple classes, and a class can have multiple students"
  - Many-to-Many

Participation Constraint

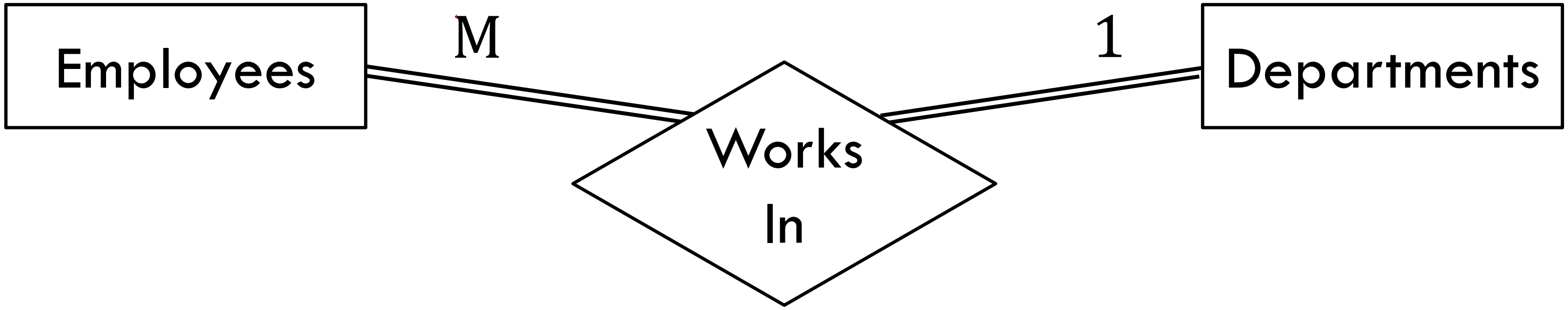
- Double line indicates all entities in the set *must* participate



- “An employee must work in a department, but a department does not necessarily have any employees”

Participation Constraint

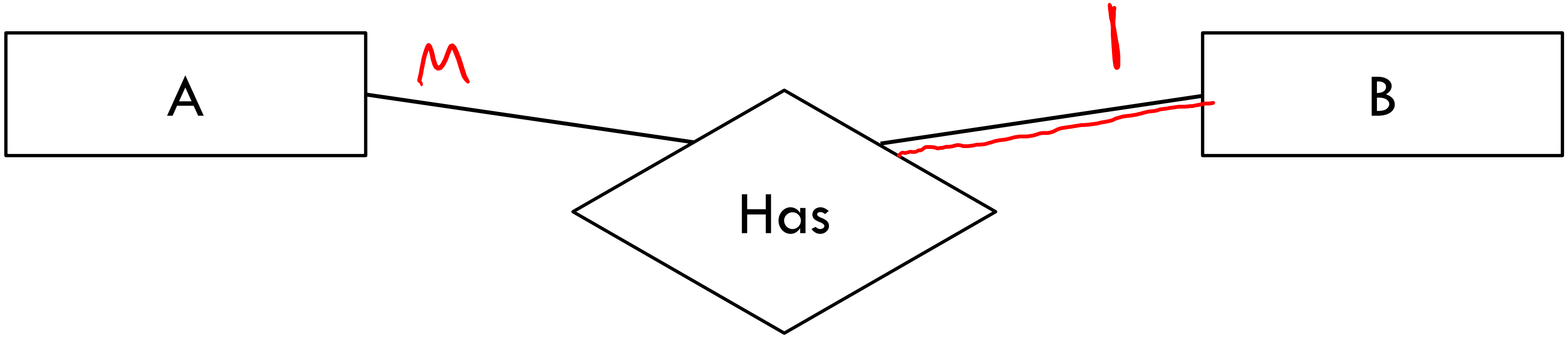
- Bold line indicates all entities in the set *must* participate
  - At least once



- "An employee must work in **one** department, and a department must have **at least** one employee"

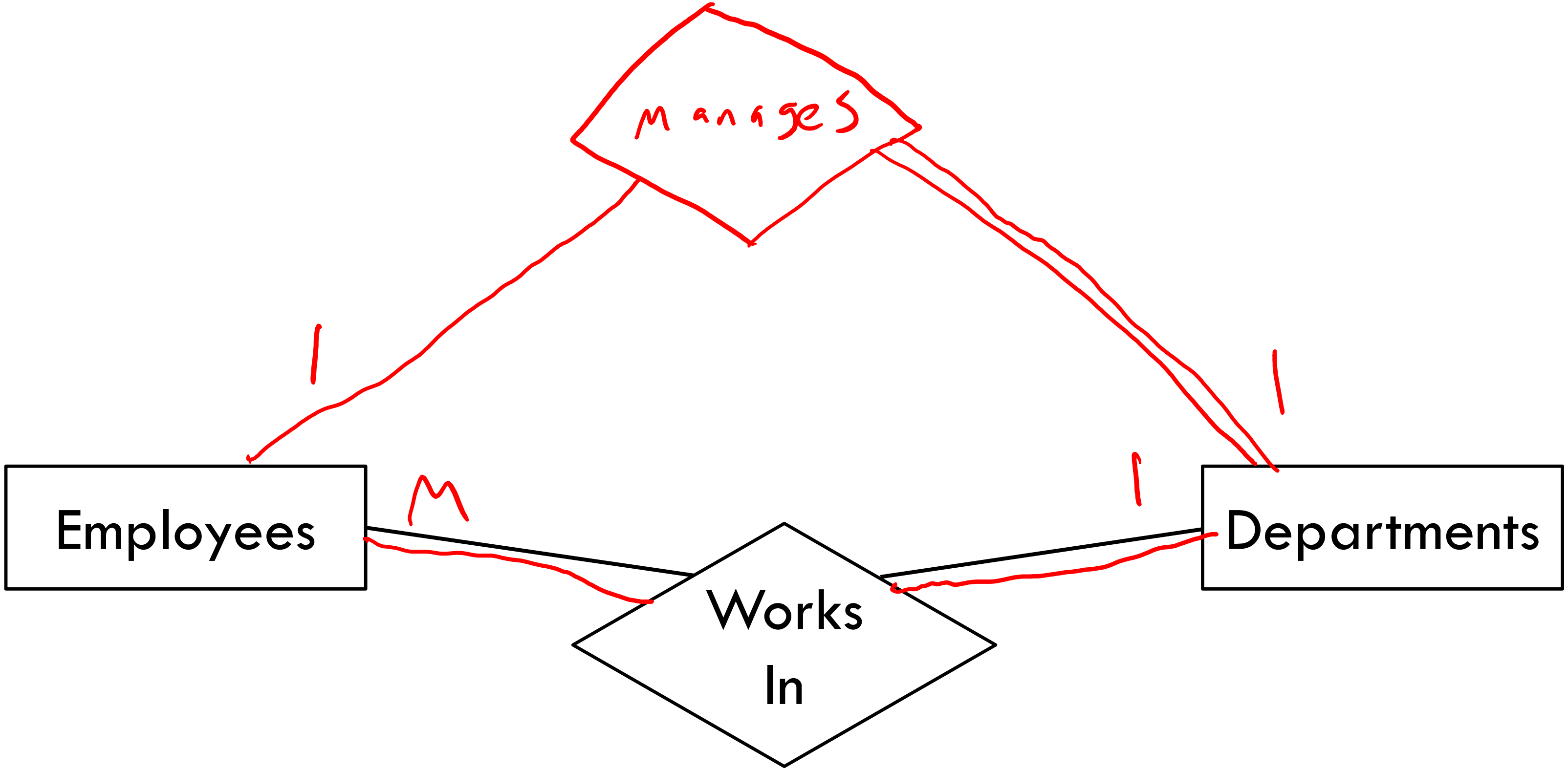
Practice – Annotate Diagram

- An A has *at most* one B
- A B has *at least* one A



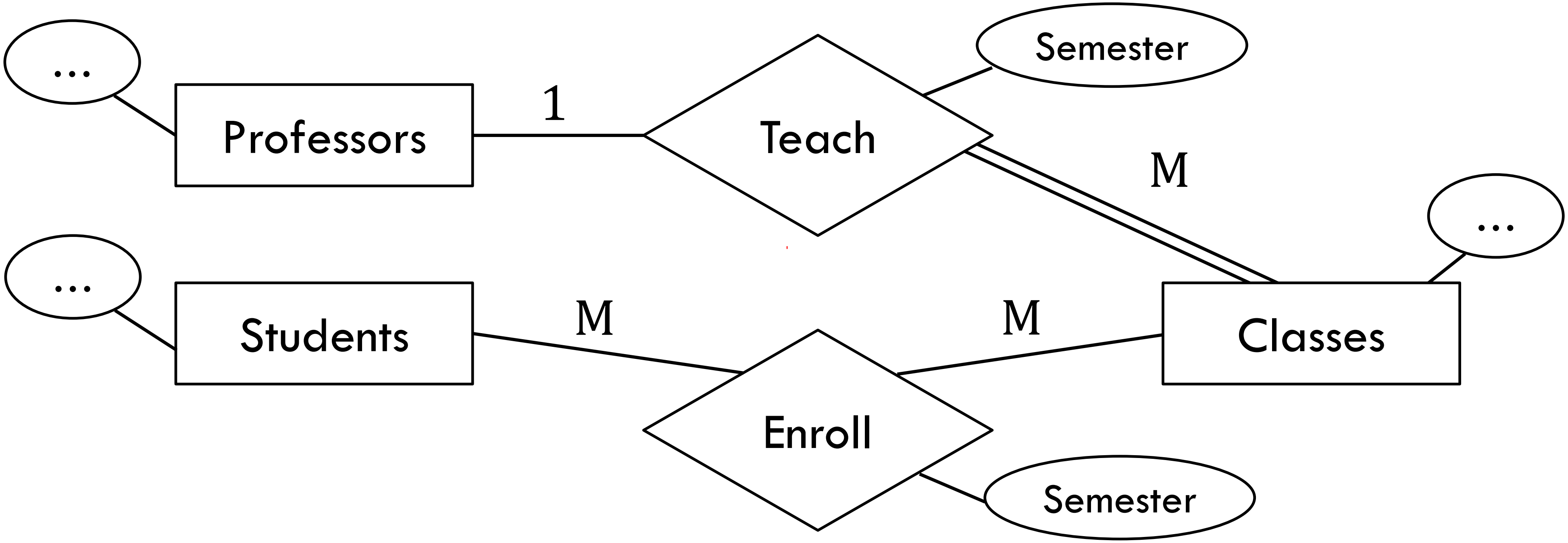
Practice

- A department has at least one employee
- An employee works for exactly one department
- A department has exactly one manager
- An employee can manage up to one department



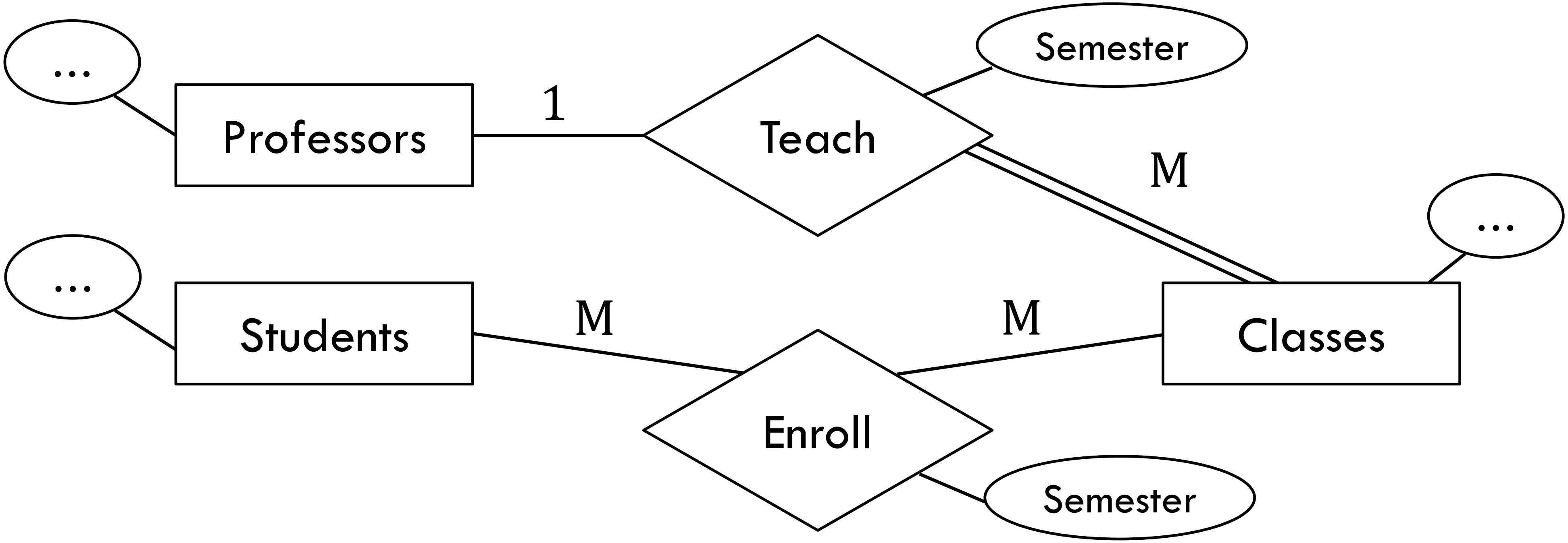
Practice (Translate Diagram)

- Can a professor take sabbatical?
- Can a class be co-taught?
- Can a class have no teacher?
- Can a class have no students?
- Can a student take multiple classes?
- Can a student take the semester off?



Practice (Translate Diagram)

•Can a professor take sabbatical?	yes
•Can a class be co-taught?	no
•Can a class have no teacher?	no
•Can a class have no students?	yes
•Can a student take multiple classes?	yes
•Can a student take the semester off?	yes



Hierarchical Types

```
class Employee { int SSN; string name; }


class HourlyEmployee extends Employee
{  float wage;  }


class SalaryEmployee extends Employee
{  float salary; Benefits b;  }
```

Hierarchical Types

```
class Employee { int SSN; string name; }
```

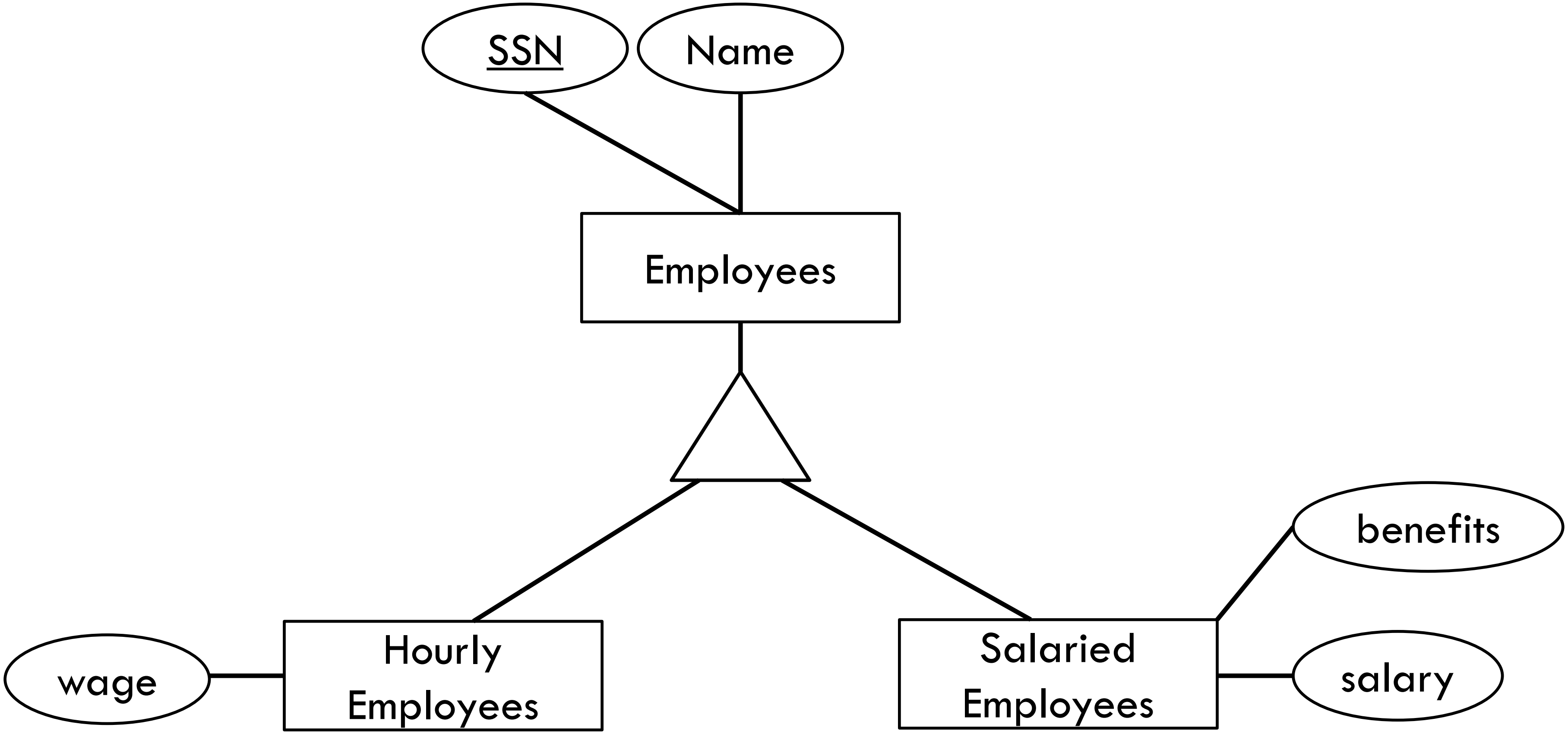
```
class HourlyEmployee extends Employee
{   float wage;   }
```

```
class SalaryEmployee extends Employee
{   float salary; Benefits b;   }
```

- `HourlyEmployee` "IS-A" `Employee`

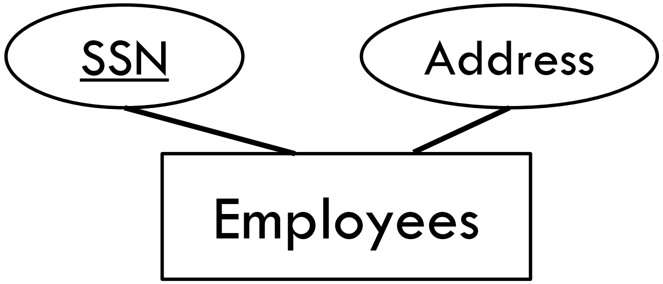
“IS-A”

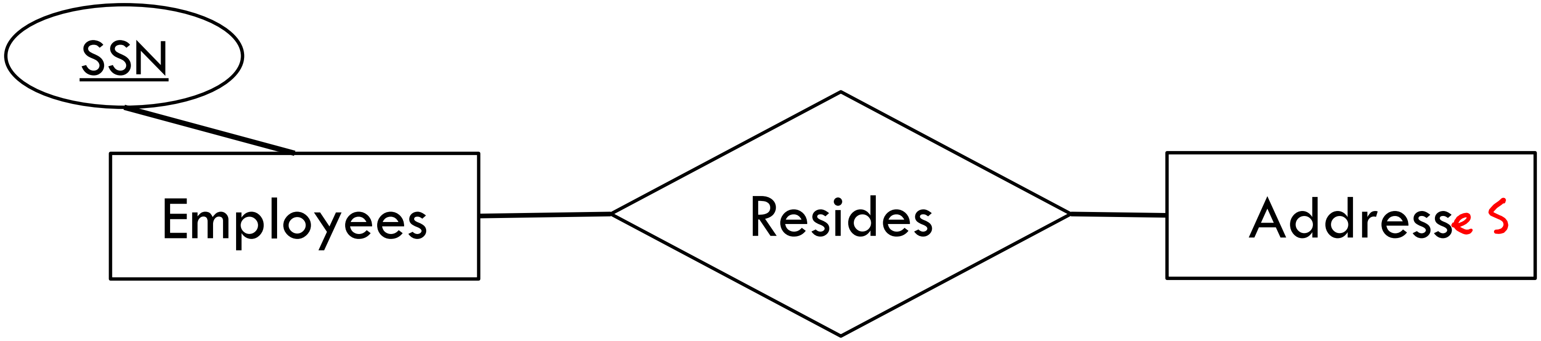
△ = IS-A


SSN
Name
Employees
Hourly Employees
Salaried Employees
wage
benefits
salary

Entity or Attribute?

- Am employee has an address





Entity or Attribute?

- It’s usually obvious:
  - A student is an entity
  - A student ID is **not** an entity

Entity or Attribute?

- If it’s not obvious, a few questions to ask:
  - Is it complex data that needs its own keys?

Entity or Attribute?

- If it’s not obvious, a few questions to ask:
  - Is it complex data that needs its own keys?
  - Can an entity have more than one of them?

Entity or Attribute?

- If it’s not obvious, a few questions to ask:
  - Is it complex data that needs its own keys?
  - Can an entity have more than one of them?
  - Does the data type make sense in another relationship?

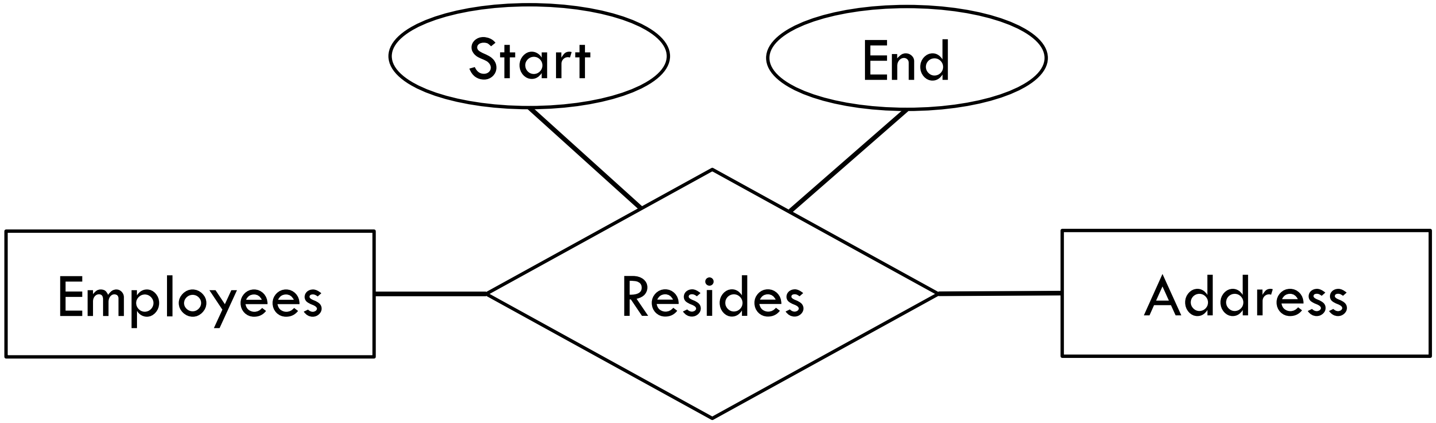
Entity or Attribute?

- If it’s not obvious, a few questions to ask:
  - Is it complex data that needs its own keys?
  - Can an entity have more than one of them?
  - Does the data type make sense in another relationship?

- “yes” to these usually argues for an entity
  - But not always!

Entity or Attribute?

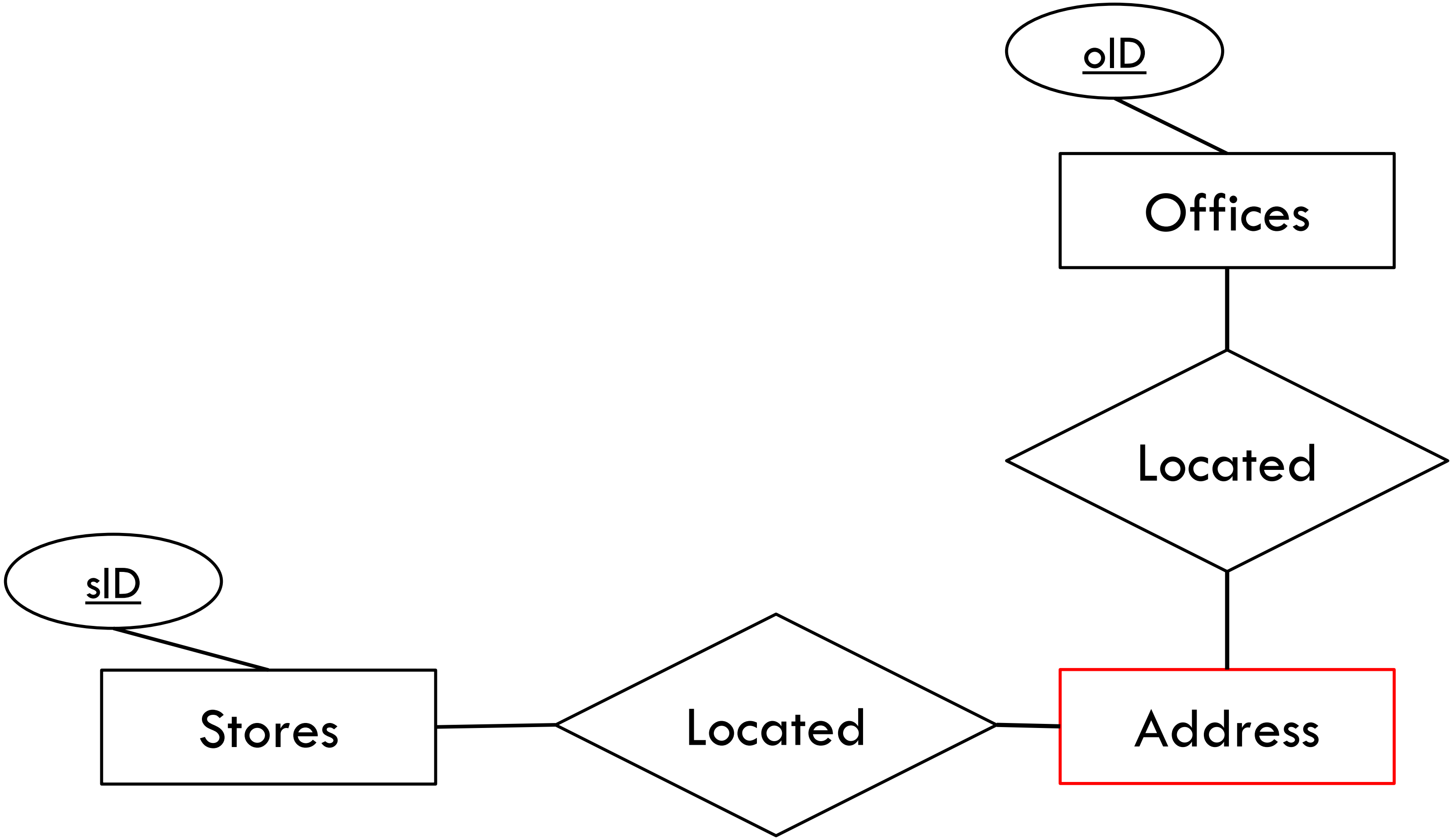
- Track full residence history



- Now an employee can have multiple addresses

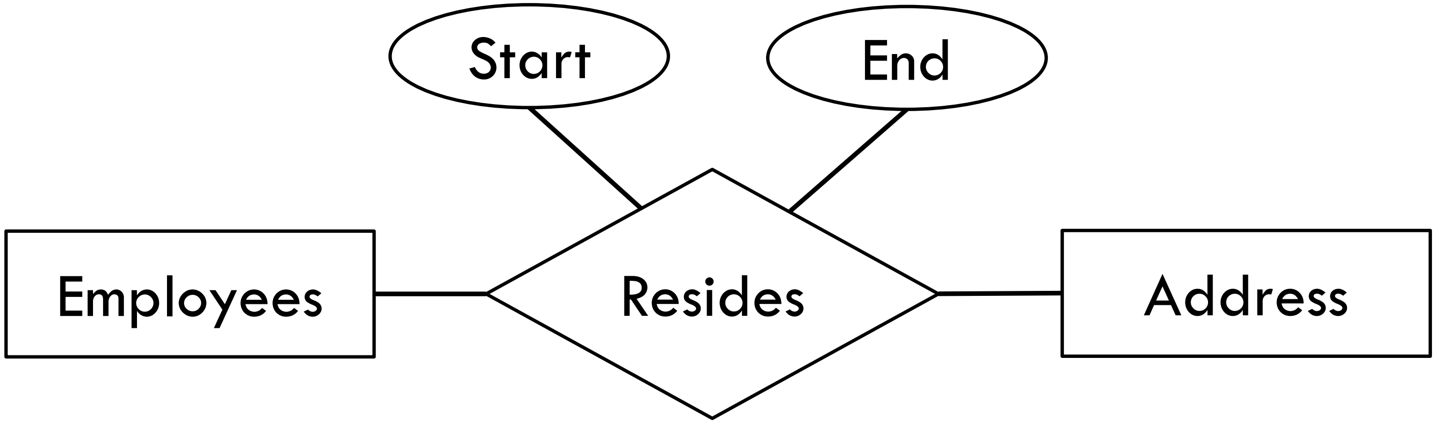
Entity or Attribute?

- A company has retail locations and office locations



Ponder

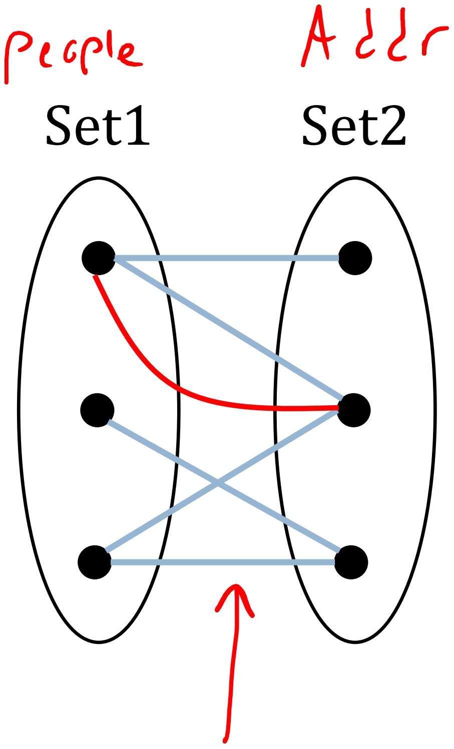
- What if the employee lives at the same residence two different times?



5/1/16 – 5/1/17, 723 Evergreen Terrace
5/1/17 – 5/1/18, 555 Creek Rd.
5/1/18 – 5/1/19, 723 Evergreen Terrace

Ponder

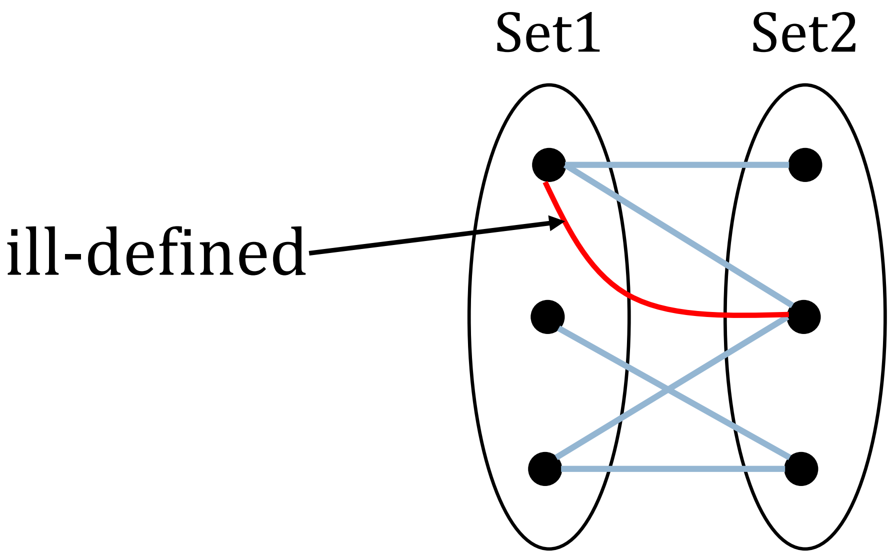
- What if the employee lives at the same residence two different times?



Many-to-Many

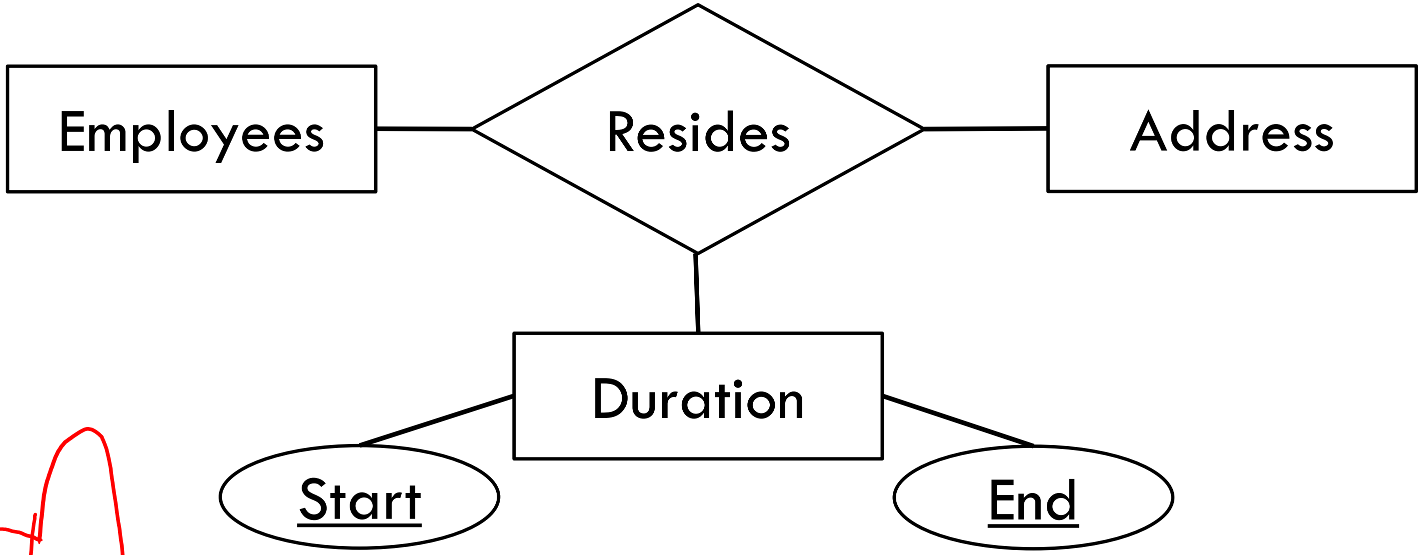
Ponder

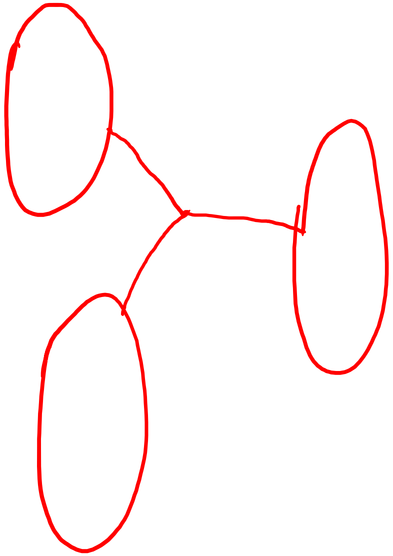
- What if the employee lives at the same residence two different times?



Many-to-Many

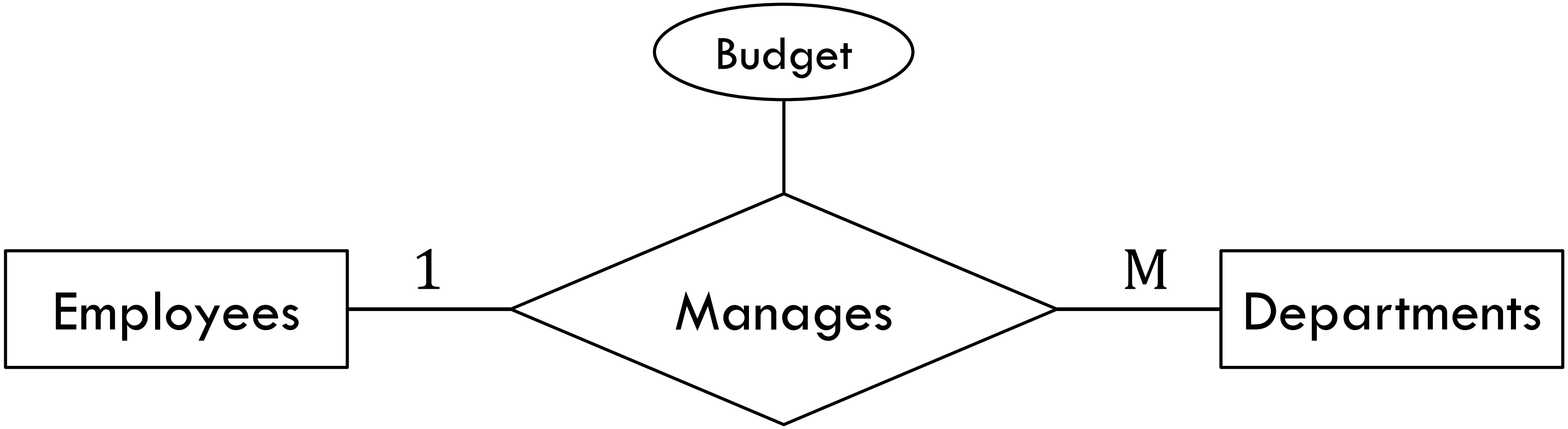
Ternary Relationship





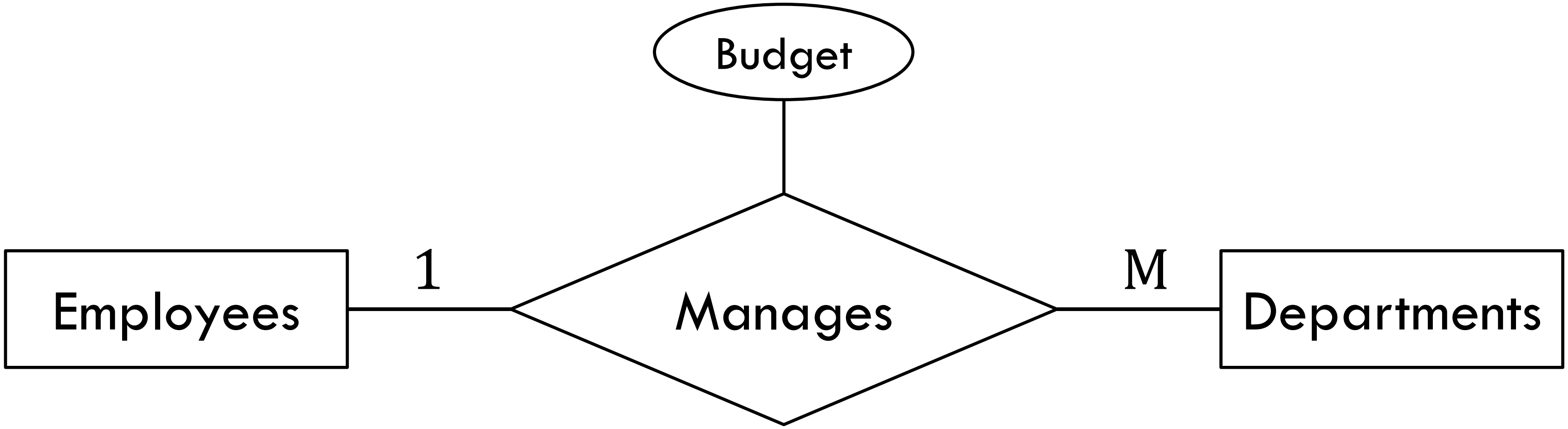
Entity vs. Relationship

- An employee can manage multiple departments
- With a different budget for each department



Entity vs. Relationship

- An employee can manage multiple departments
- With a different budget for each department

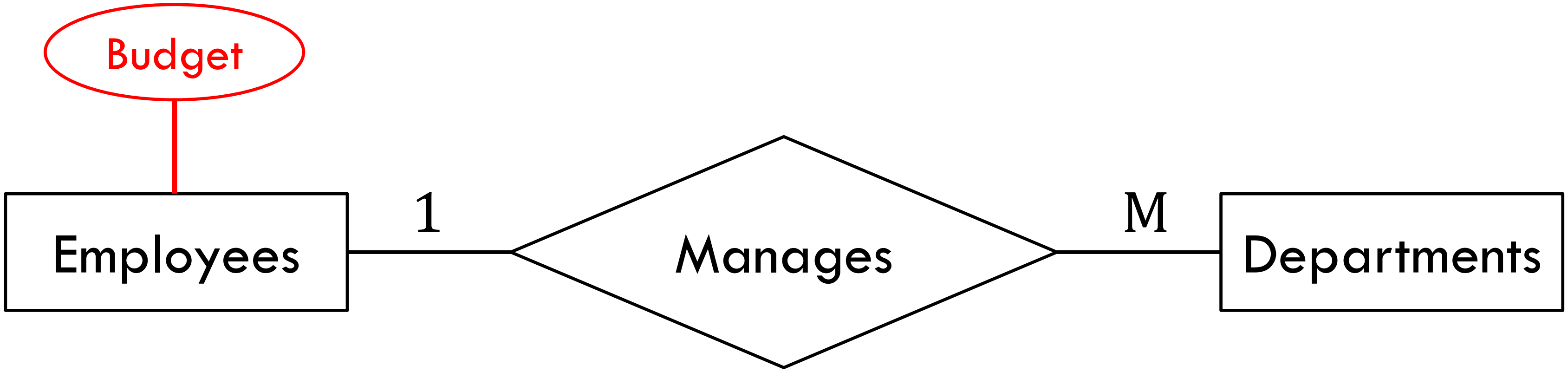


- What if the manager has just one budget to split?

Entity vs. Relationship

- An employee can manage multiple departments

- With a different budget for each department

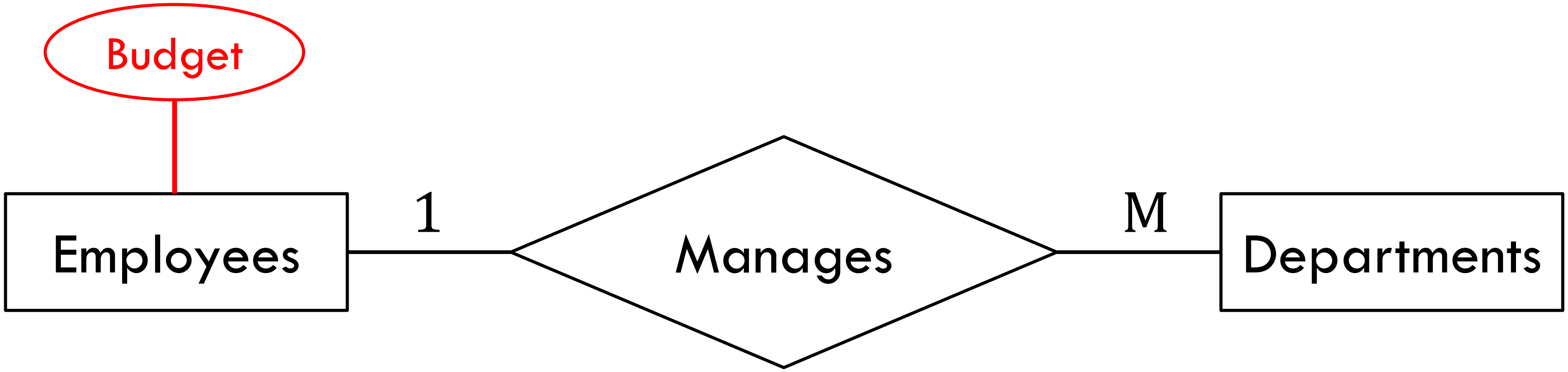


- What if the manager has just one budget to split?

Entity vs. Relationship

- An employee can manage multiple departments

- With a different budget for each department

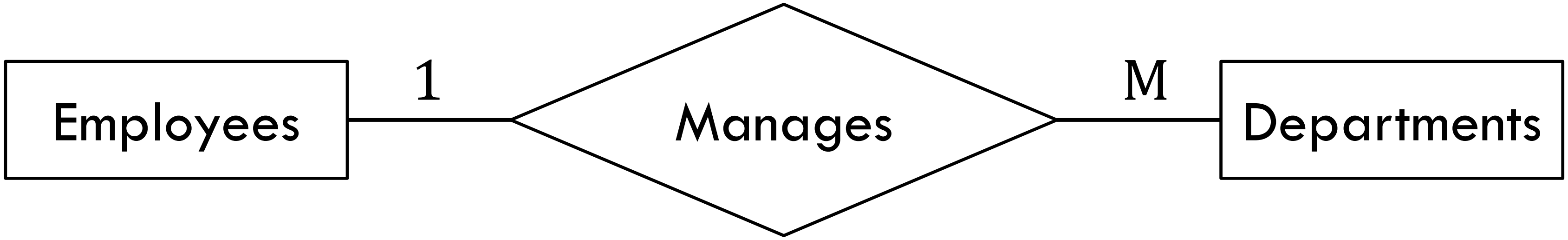


- What if the manager has just one budget to split?
  - Bad: not all employees are managers

Entity vs. Relationship

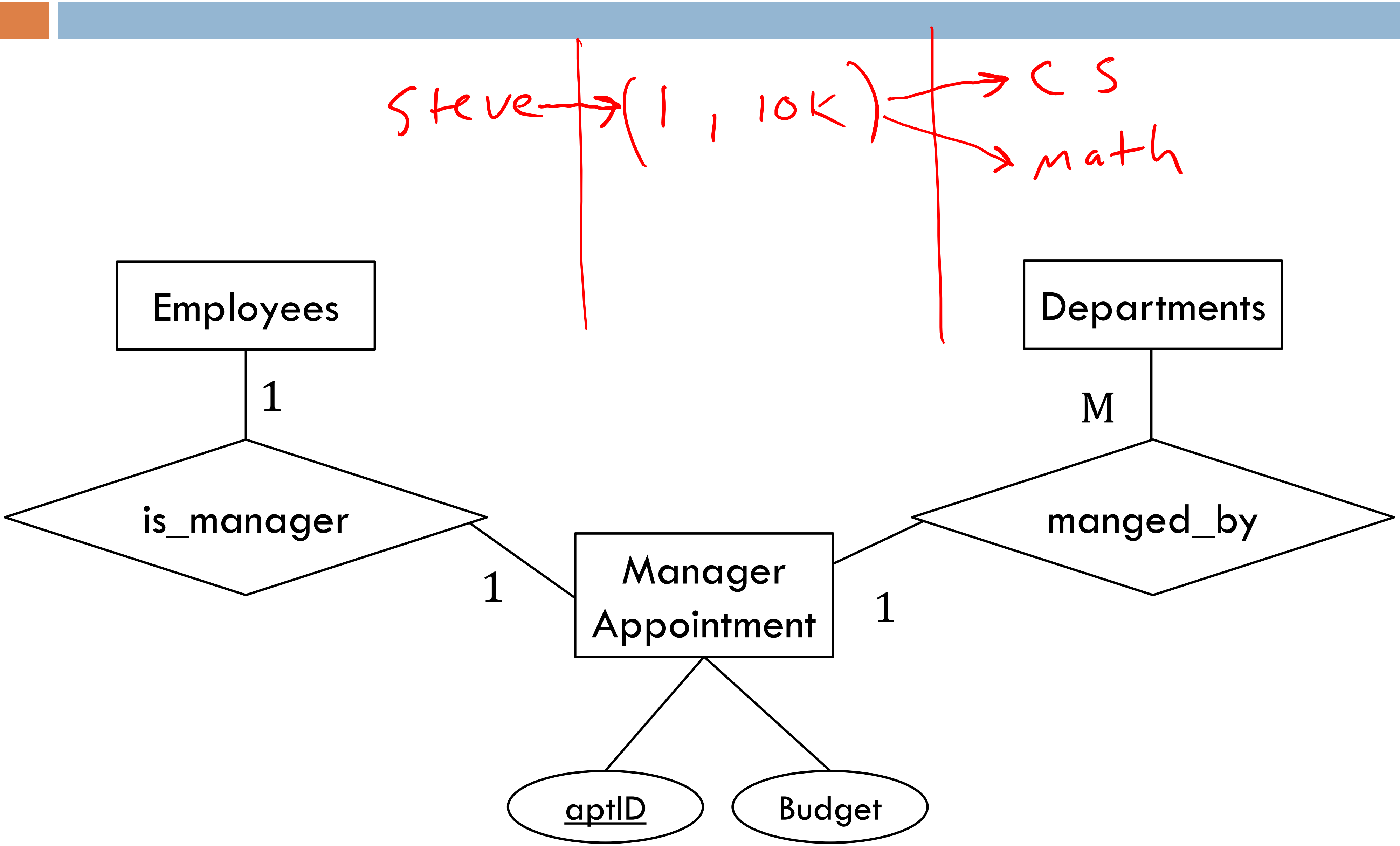
- An employee can manage multiple departments

- With a different budget for each department

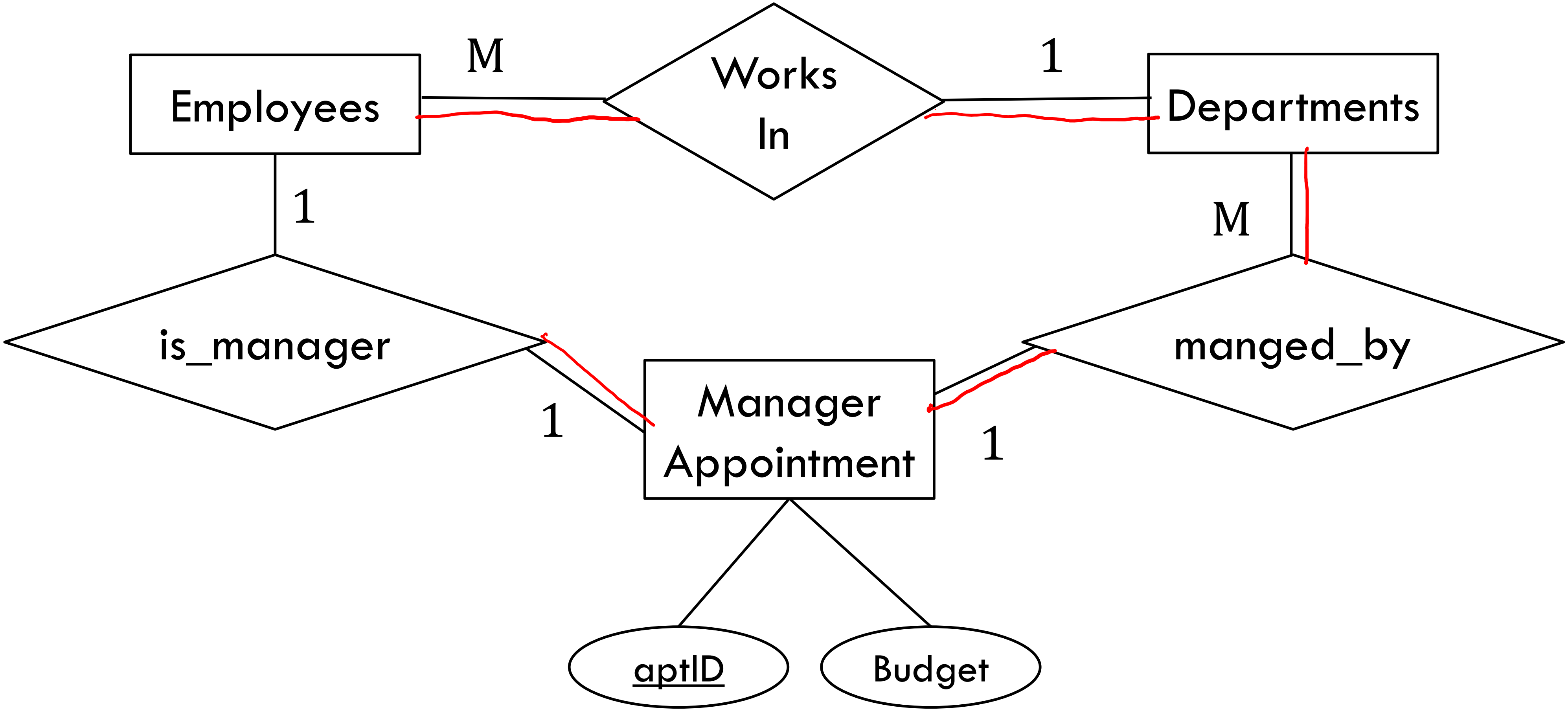


- What we need: an “entity” that defines a management roll

Appointment Entity



Annotate


Employees
M
Works
In
1
Departments
1
M
is_manager
manged_by
1
Manager
Appointment
1
aptID
Budget

ER Diagram Notations

- There are lots of different notations

- We will use Chen notation (what we studied today)

Try it Out

- Good options
  - draw.io – free!
  - lucidchart – better, but not free

- Working options
  - Powerpoint
  - Illustrator
  - MS Paint