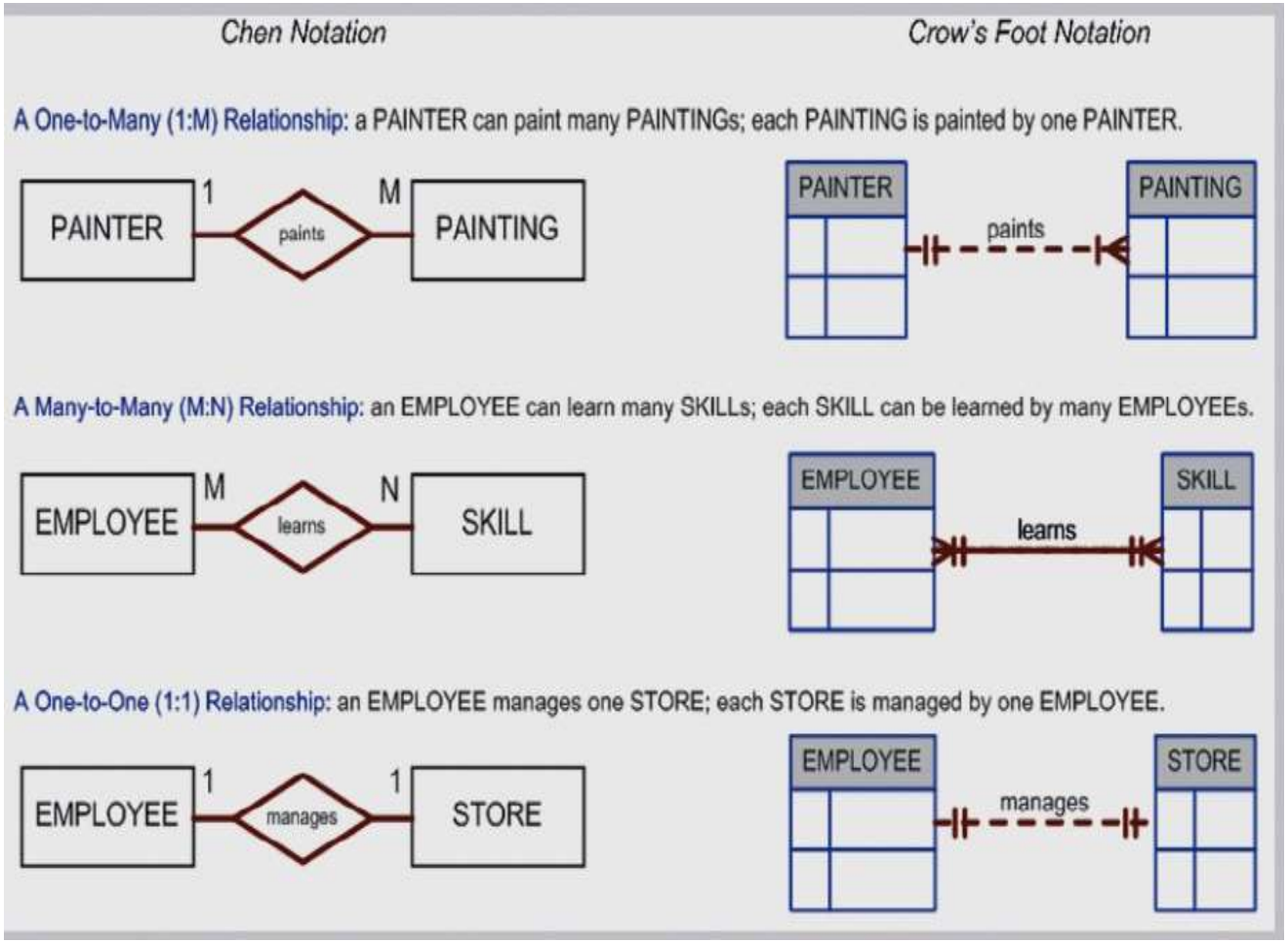
# E-R Modelling

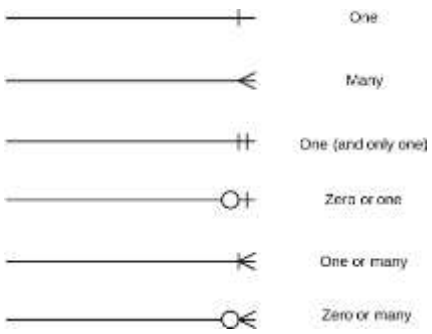
# Building Blocks of ERD

| Type | English Grammar Equivalent | Example |
|---|---|---|
| Entity | Proper Noun | Student, Employee, Instructor, Courses, Room |
| Relationship | Verb | has, teaches, belongs, handles |
| Attribute | Adjective | Height, Age, Gender, Nationality, First name |

# ERD –Two Popular Notation

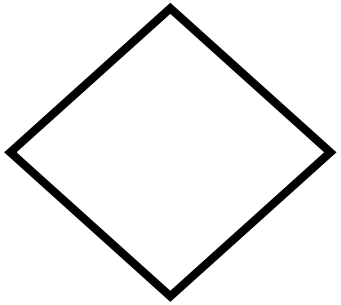
- Chen Notation Crow's Foot Notation



# **Chen Notation** - Symbol

Rectangle represents an **Entity**



Diamond represents a **Relationship**


Lines with labels represents **Cardinality**

# Entity (Chen Notation)

- is a real-world object distinguishable or unique from other objects.
- An entity can be a **concrete** or **physical** object like *employee*, *student*, *faculty*, *customer* etc. Or it could also be **conceptual** or **abstract** like *transaction*, *order*, *course*, *subjects* etc.
- It can be thought of as a noun like *student, employee* etc.
- It is normally represented by a rectangle shape.

# Entity Could be a…

Person (ex. Teacher, Student, Physician)

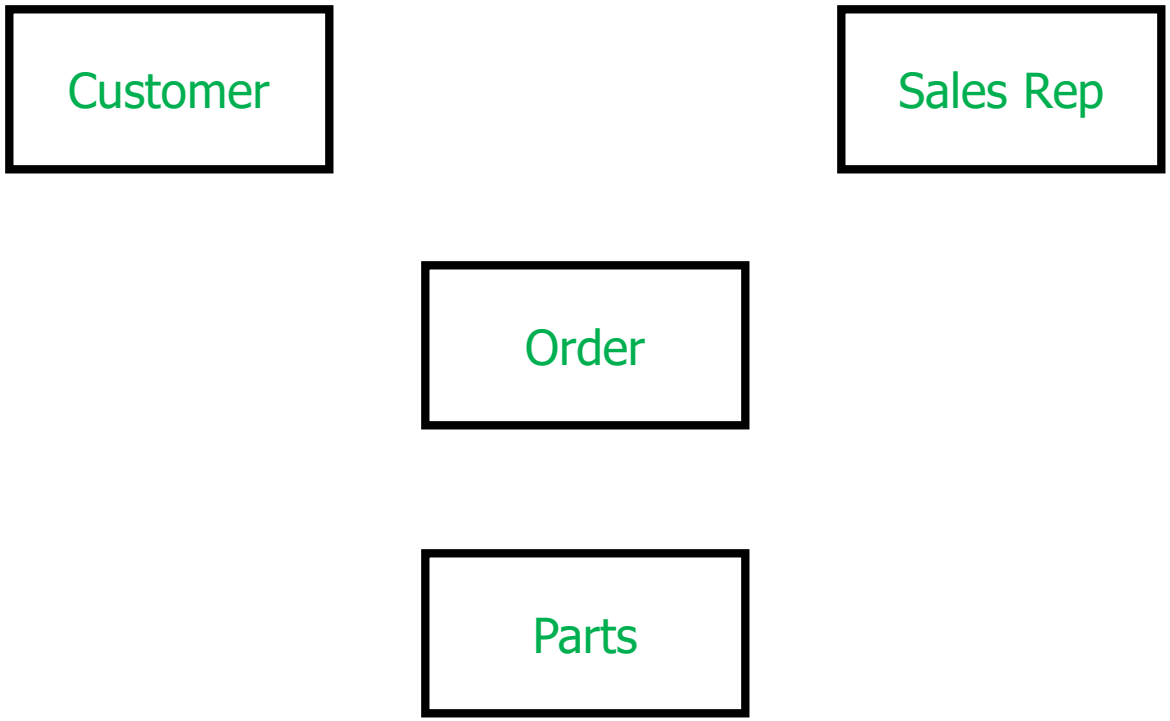
Place (ex. School, Hotel, Store )

Object (ex. Mouse, Books, Bulding )

Event (ex. Enroll, Withdraw, Order )

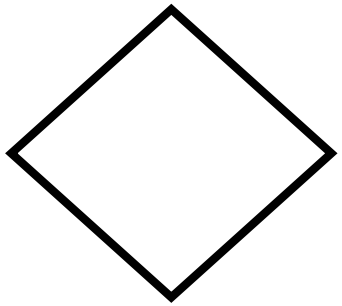
Idea or Concept (ex. Courses, Account, Delivery )

# Entity - Example

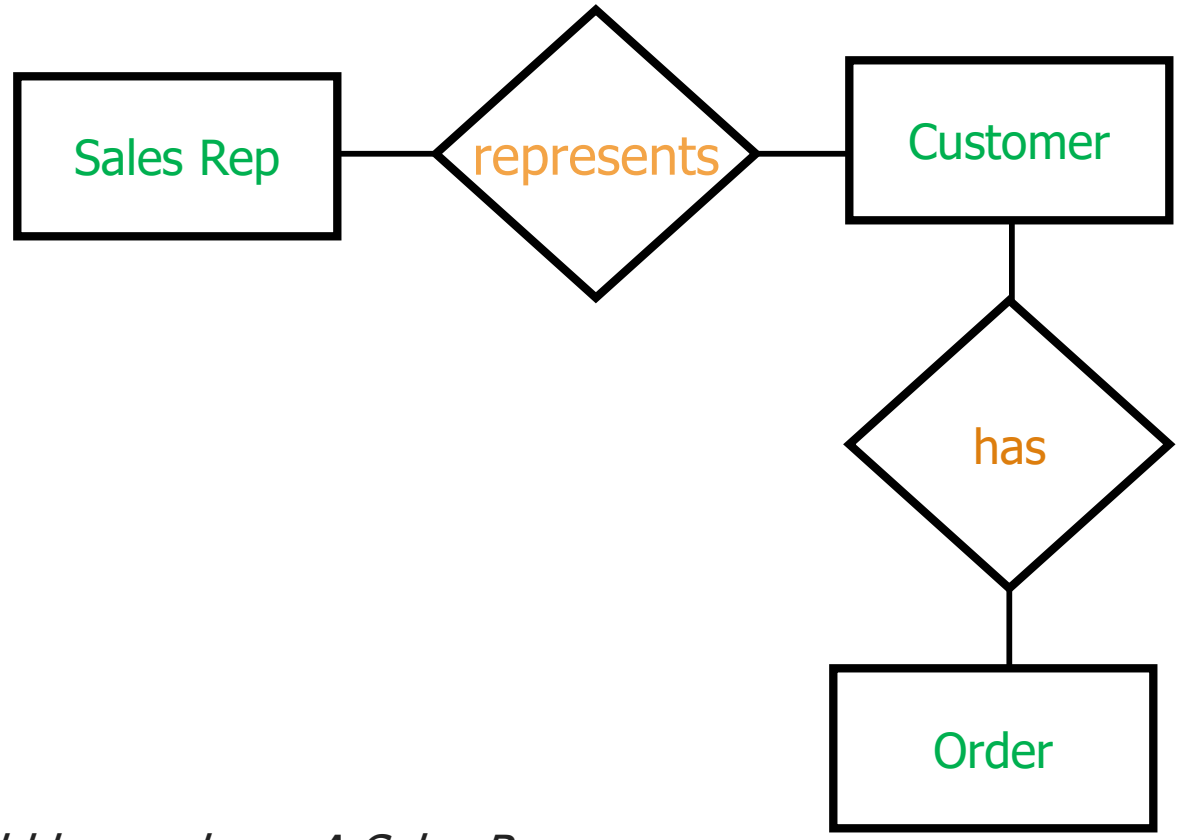


# Relationship

- is a way of relating one entity to another. Entities can therefore participate in a relationship.
- it is commonly thought as a *verb* connecting the *entities* or *nouns*.
- It is normally represented by a diamond shape.



# Relationship - Example




*Could be read as : A Sales Rep Represents a Customer. And a Customer has an Order.*

# Cardinality

- **Cardinality**: number of items that must be included in a relationship
  - An entity in a relationship with minimum cardinality of zero plays an **optional role** in the relationship
  - An entity with a minimum cardinality of one plays a **mandatory role** in the relationship

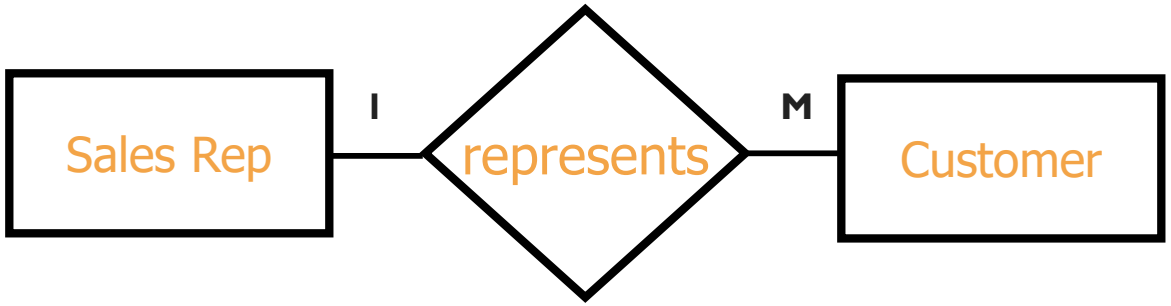
# Cardinality - Symbols


One-to-one Relationship


One-to-many Relationship

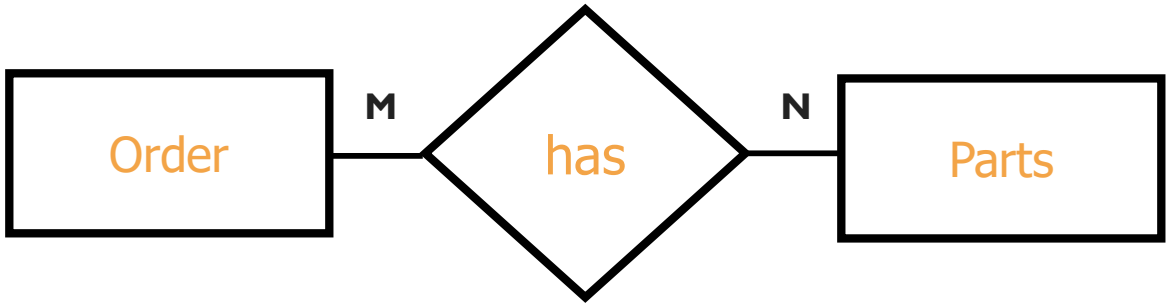

Many-to-many Relationship

# Cardinality Symbols - Example




*Could be read as : A Sales Rep could represent 1 or Many Customers.*

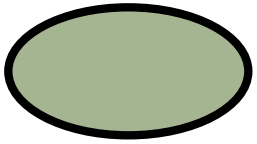
# Cardinality Symbols – Example (Cont'd)




*Could be read as : An Order could have many Parts (e.g. Products Ordered) and a Part could have many Orders.*

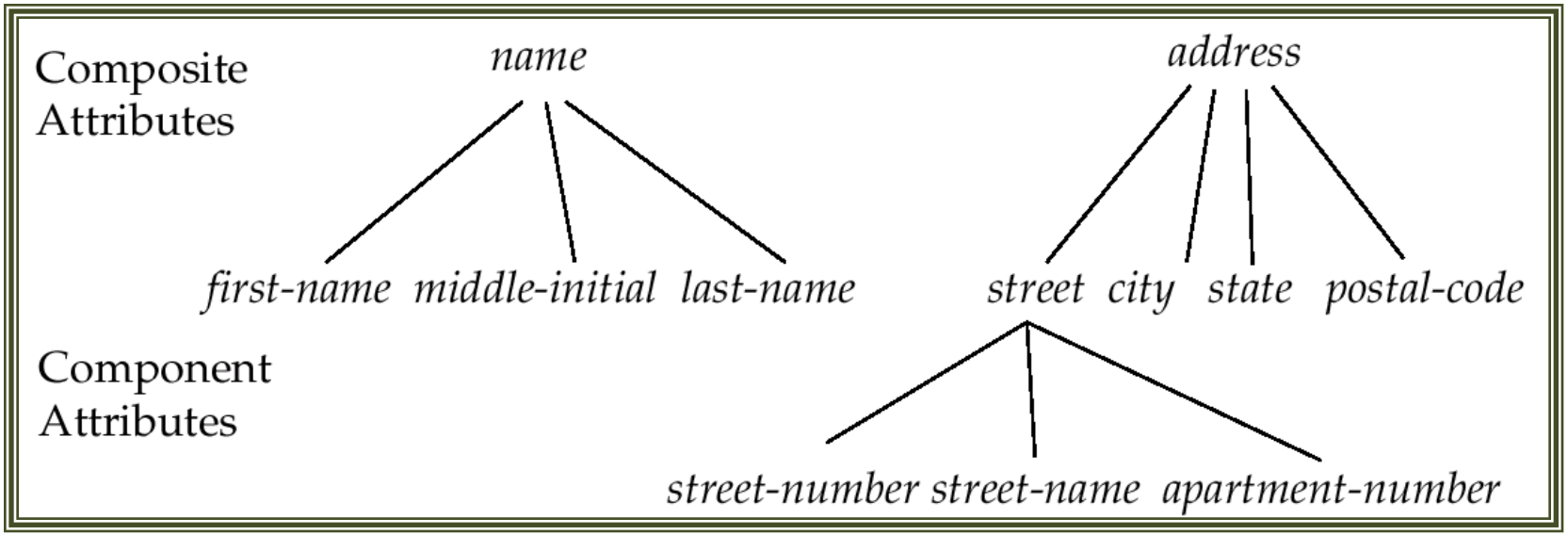
# Attribute

- Refers to the characteristic or basic fact or field of an Entity or Relationship.
- For example a Student entity could have the following **attributes** *ID Number*, *Last Name*, *First Name*, *Address*, *Birth Date* etc.
- A relationship could also have an attribute for example an Entity name **Student** enrolls (relationship) to a **Course/Program**. Now, when you enroll you enroll on a certain date so you will have an attribute of *Enrollment Date* under **Enroll** relationship.
- It is normally represented by an **oval**.

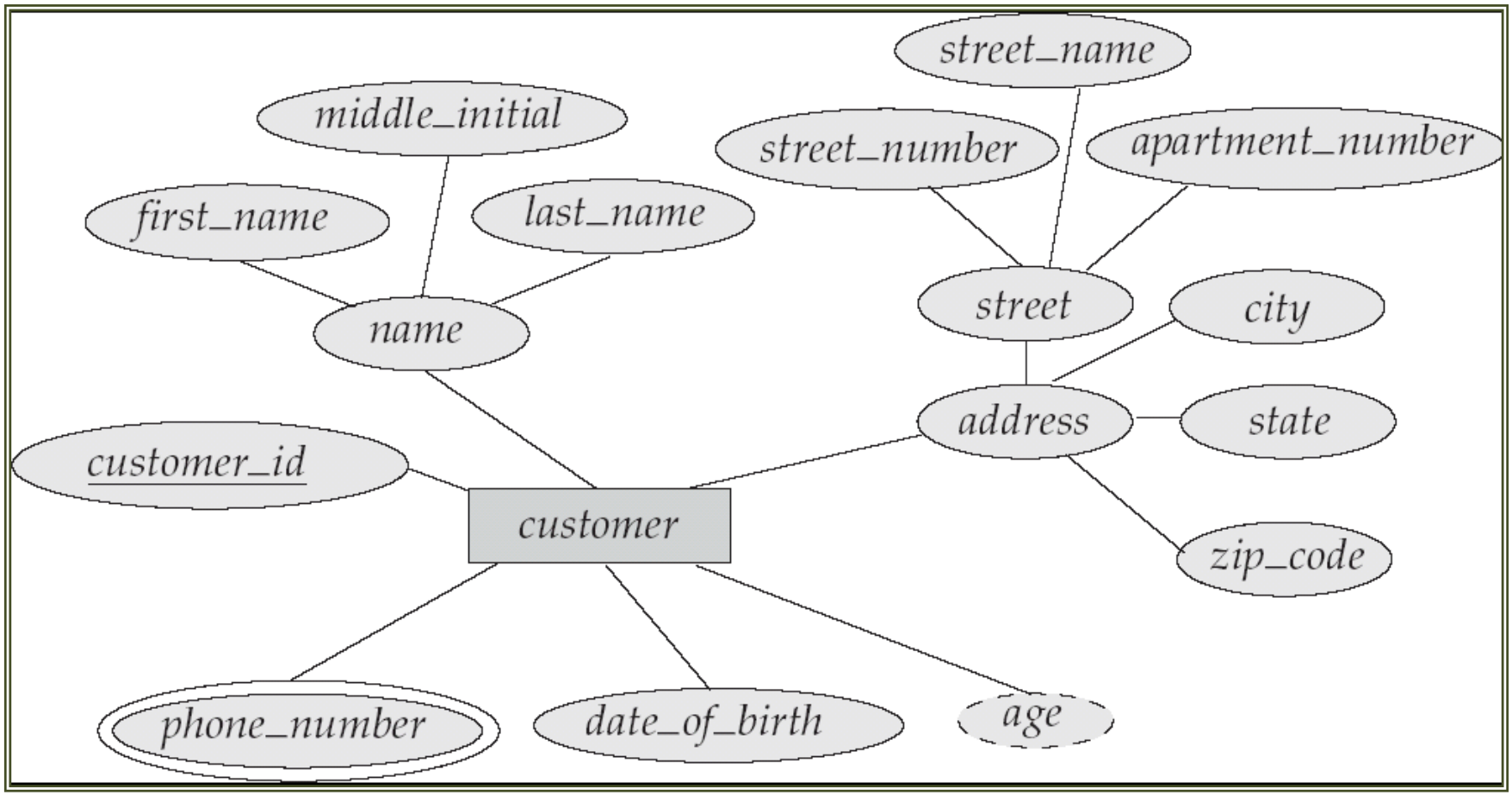


- Attribute types:
  - *Simple* and *composite* (e.g., name (first,middle,last)) attributes
  - *Single-valued* and *multi-valued* attributes (e.g., multi-valued attribute: *phone-numbers, dependents, etc*
  - *Derived* attributes
    - Can be computed from other attributes (e.g., *age*, given date of birth)

# Composite Attributes



# E-R Diagram With Composite, Multivalued, and Derived Attributes

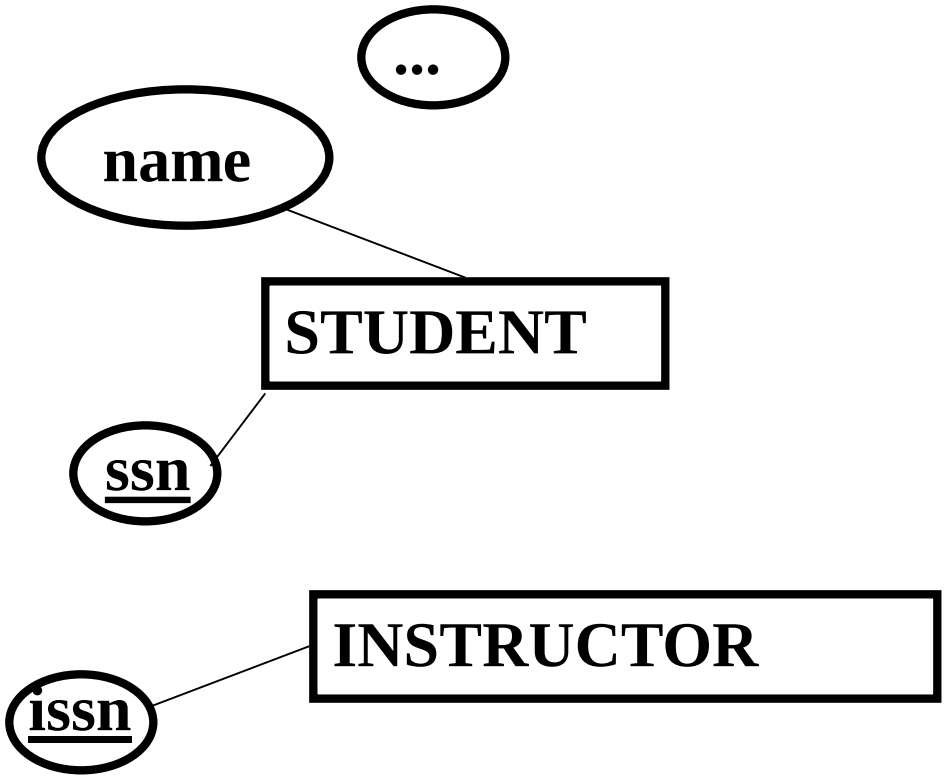


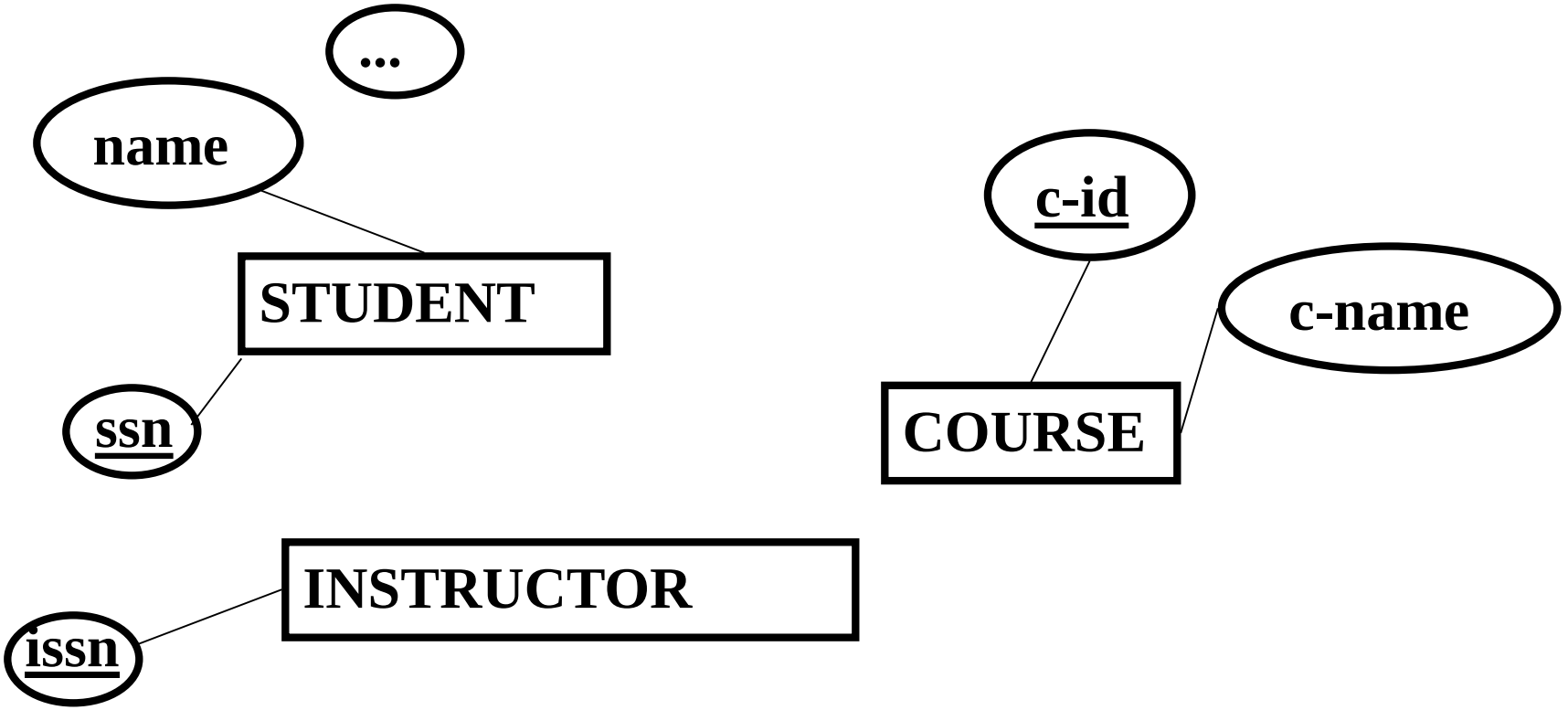
# More examples …

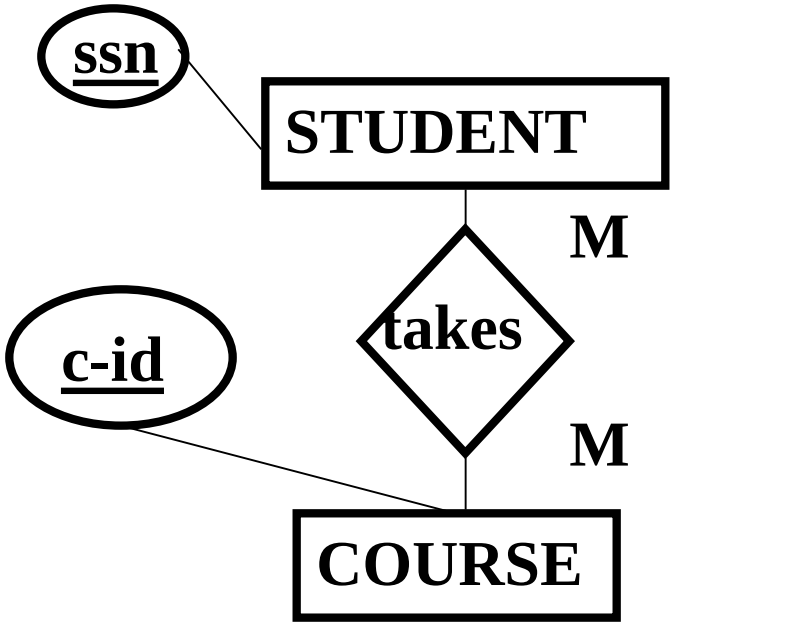
Students, taking courses, offered by instructors; one instructor per course

nouns -> entity sets

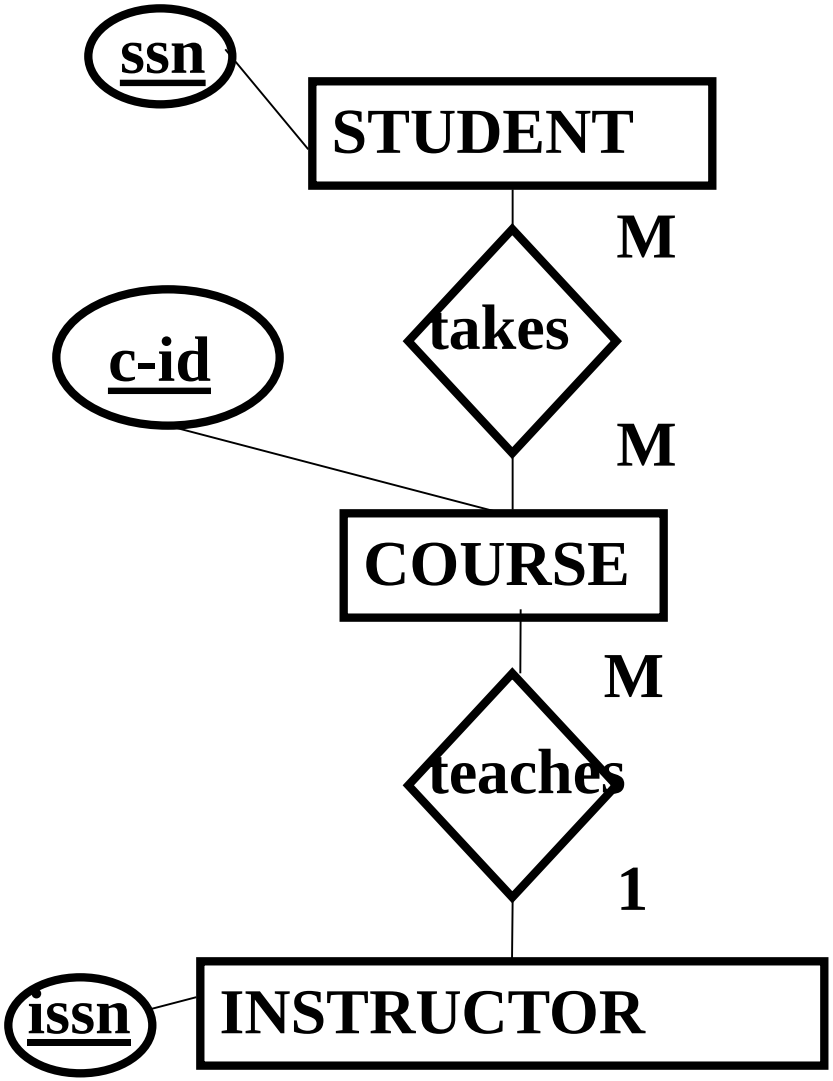
verbs -> relationships


...
name
STUDENT
ssn
INSTRUCTOR
issn


...
name
STUDENT
ssn
c-id
c-name
COURSE
INSTRUCTOR
issn


ssn
STUDENT
M
takes
c-id
M
COURSE

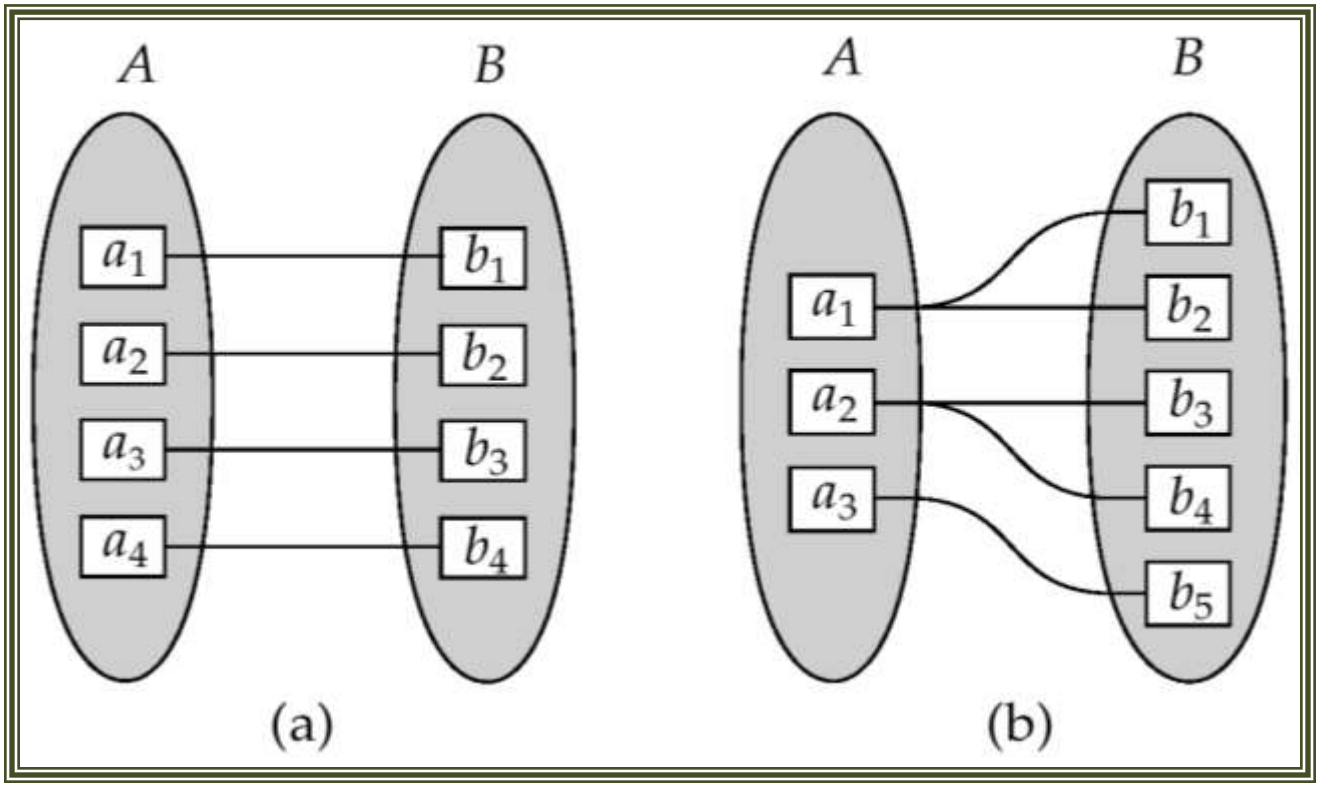

issn
INSTRUCTOR


ssn
STUDENT
M
takes
c-id
M
COURSE
M
teaches
1
issn
INSTRUCTOR

# Mapping Constraints: Cardinalities

- 1 to 1 (example?)
- 1 to M
- M to M

# Mapping Cardinalities

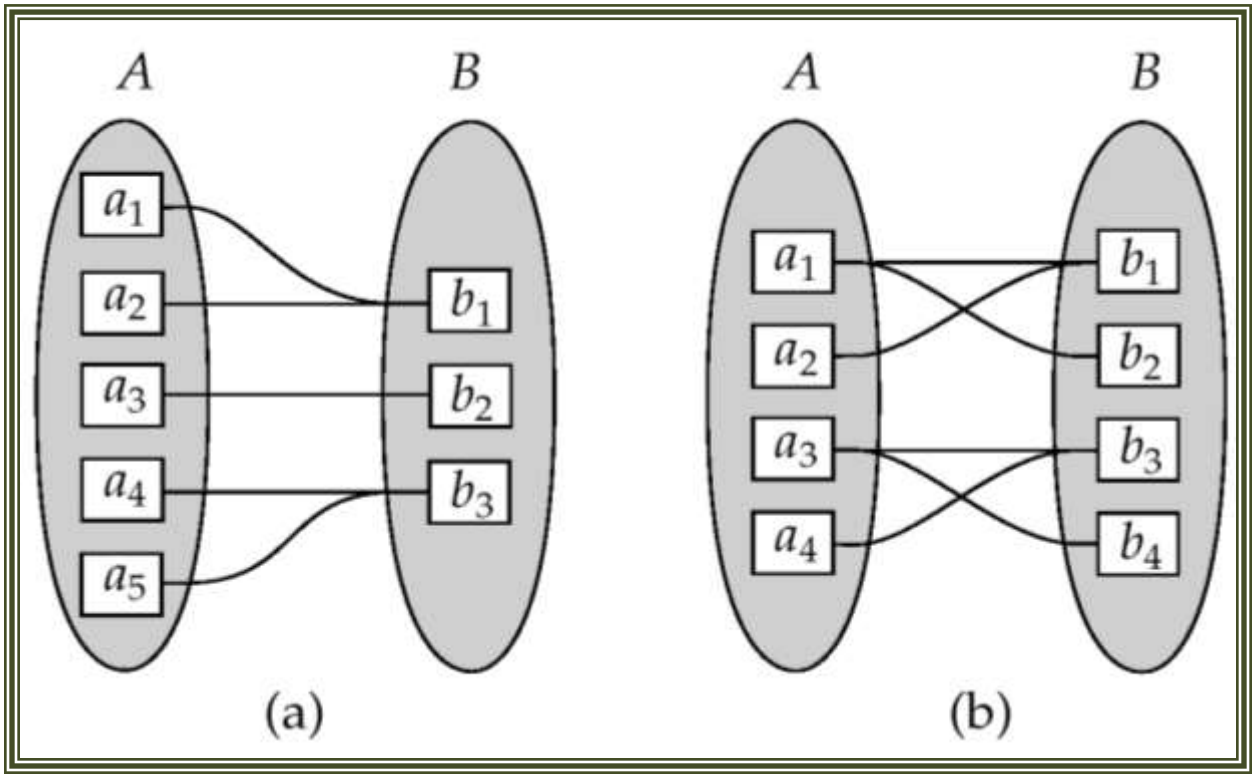



One to one

One to many

Note: Some elements in A and B may not be mapped to any elements in the other set

# Mapping Cardinalities

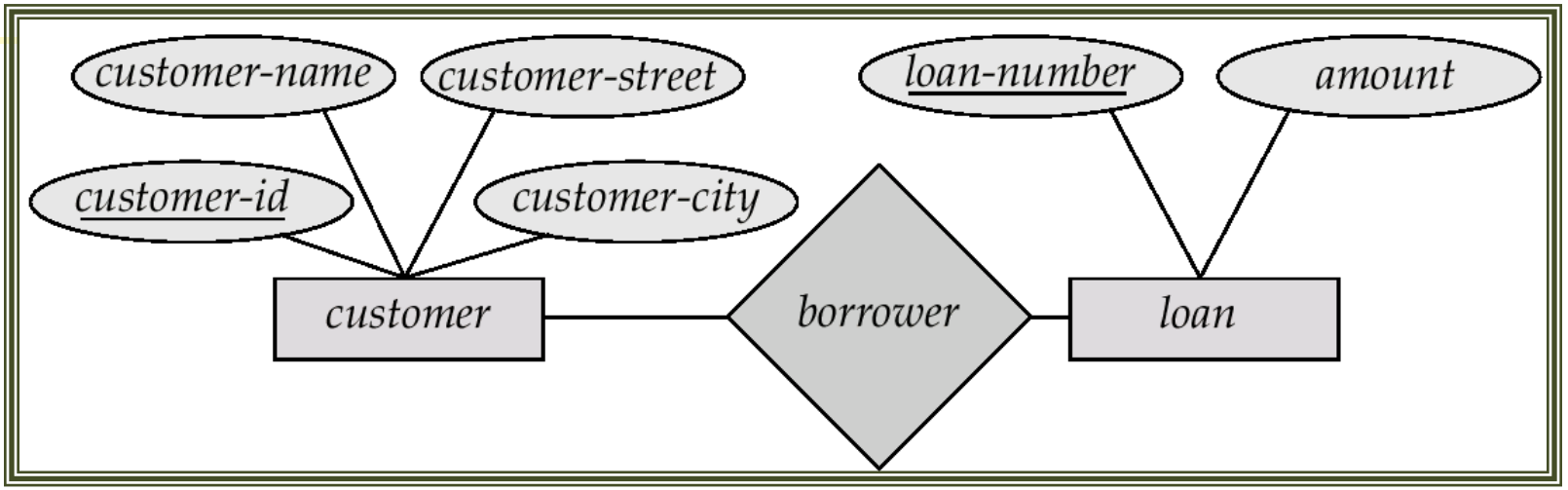



Many to one

Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

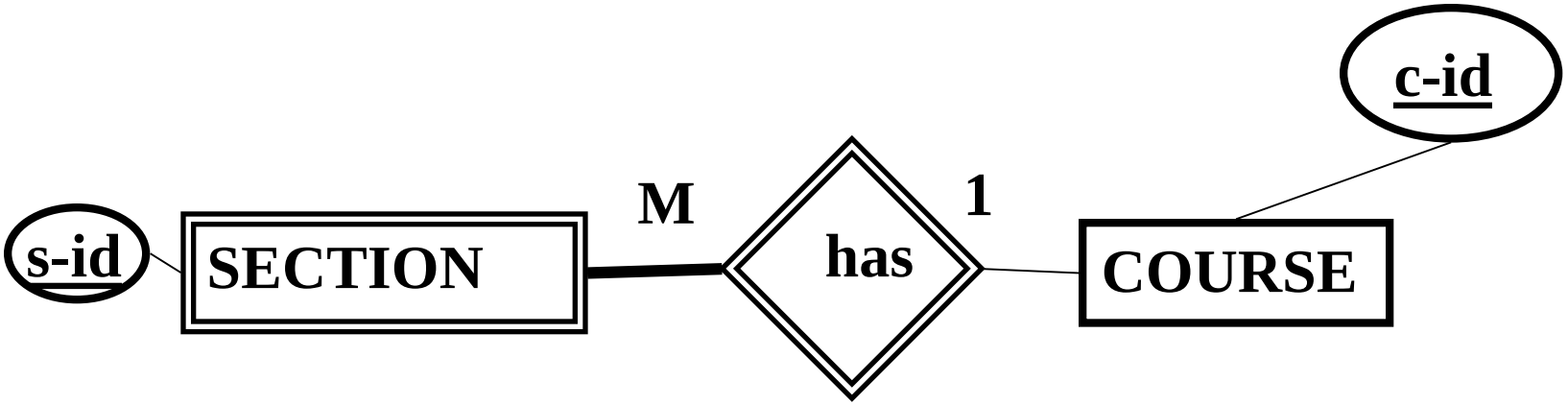
# E-R Diagrams




- **Rectangles** represent entity sets
- **Diamonds** represent relationship sets
- **Lines** link attributes to entity sets and entity sets to relationship sets
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes
  - **Dashed ellipses** denote derived attributes
- **Underline** indicates primary key attributes

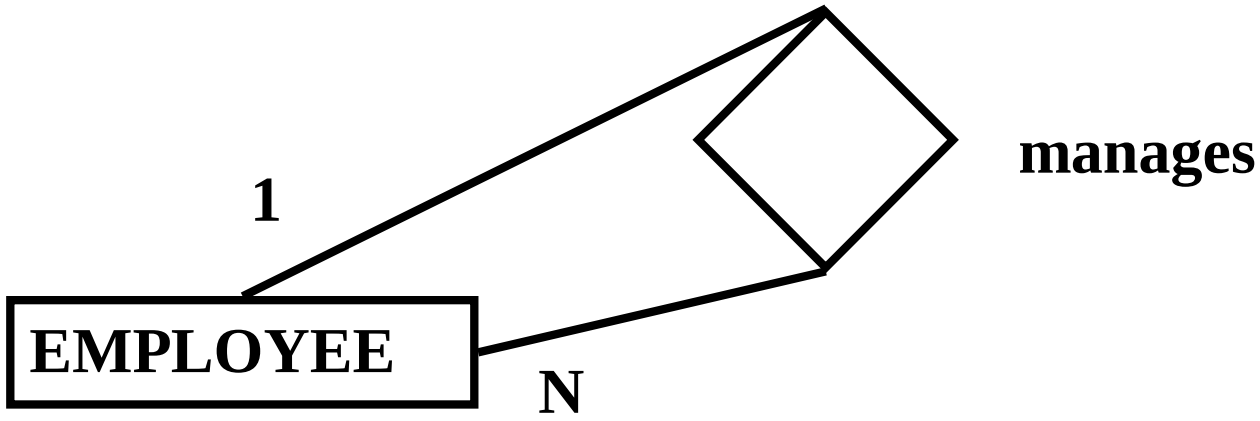
# Strong Vs Weak

- 'weak' entities: if they need to borrow a unique id from a 'strong entity - DOUBLE box.
- 'c-id' + 's-id': unique id for SECTION
- discriminator (e.g., 's-id')



# More details

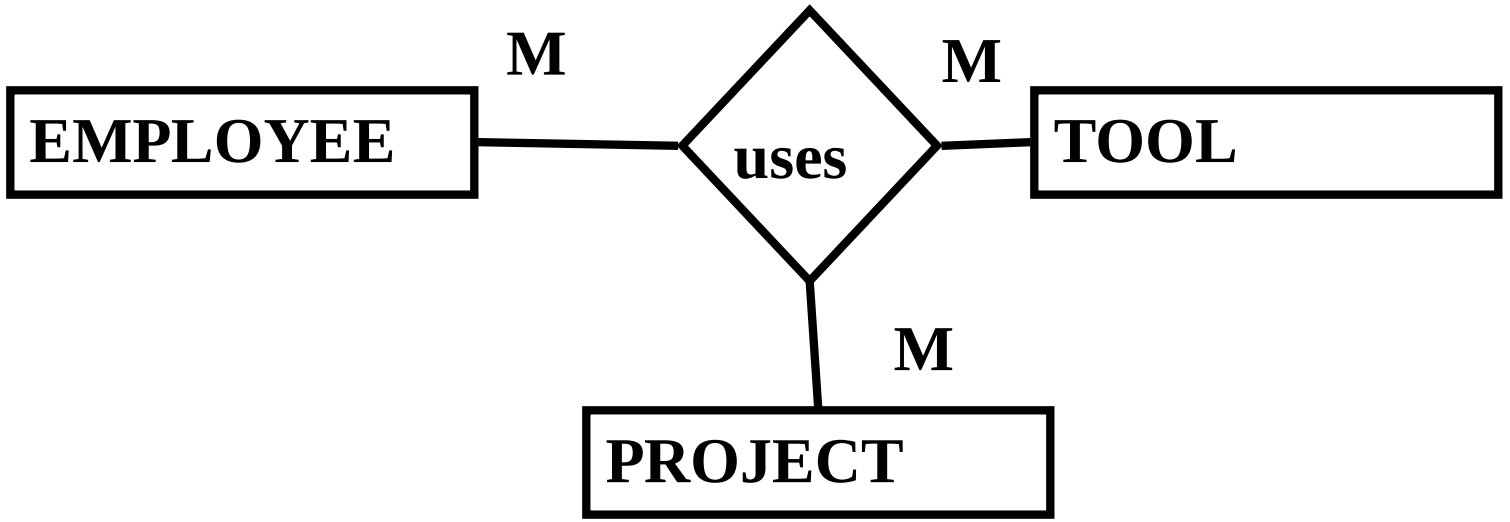
- Entity sets of a relationship need not be distinct
- self-relationships – example ?




- Usually different "roles" are indicated by labeling the lines that connect diamonds to rectangles

# More details

- Binary relationships
- 3-way and k-way relationships?

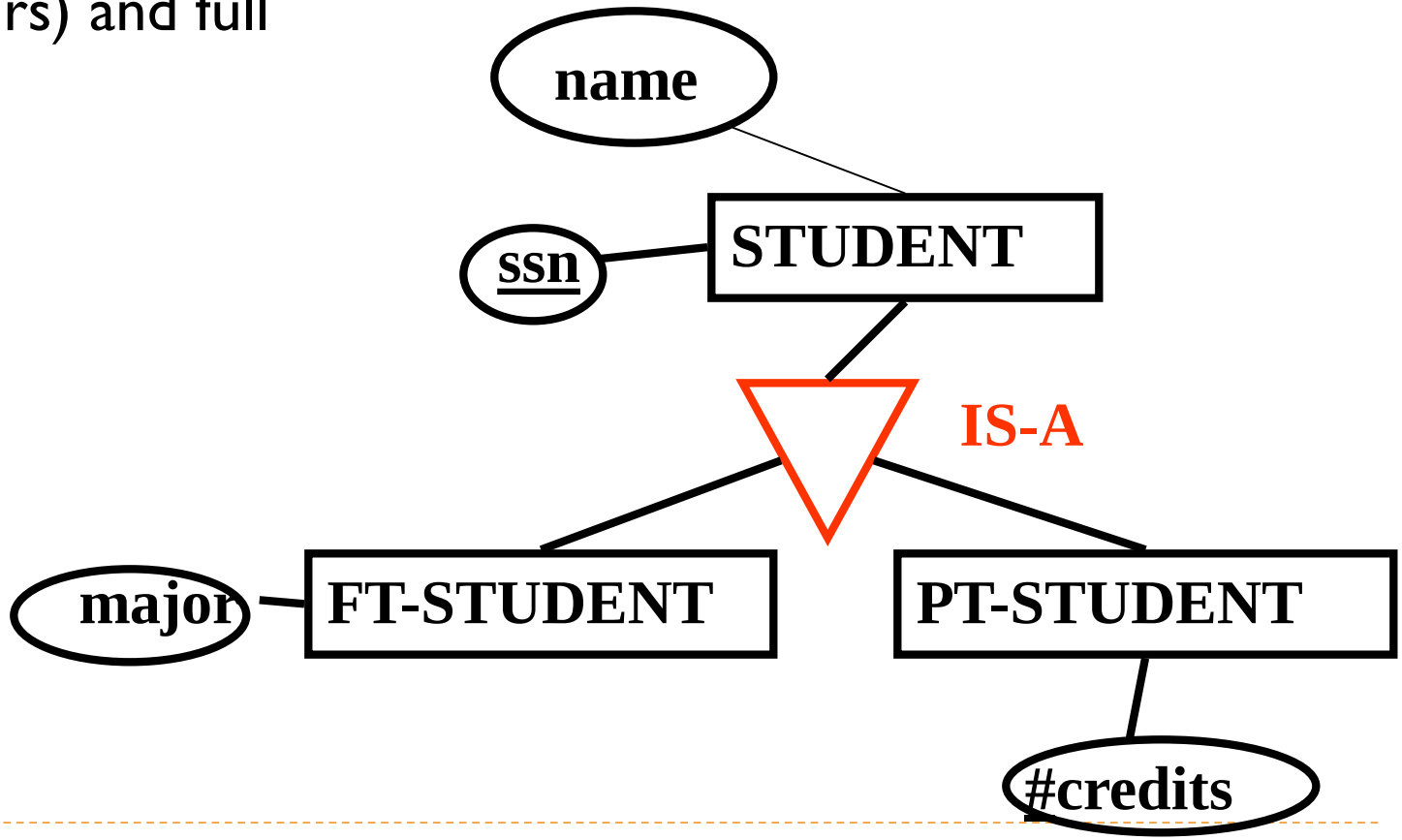


# More details - attributes

- superkey: a set of one or more attributes whose values uniquely determine each entity (e.g., (ssn, address) )
- candidate key: a minimal super key (e.g., ssn; employee#)
- primary key: a cand. key, chosen by DBA
- multivalued or set-valued attributes (e.g., 'dependents' for EMPLOYEE)
- derived attributes (e.g., 15% tip)

# Extended ER features: Specialization

- e.g., students: part time (#credit-hours) and full time (major)

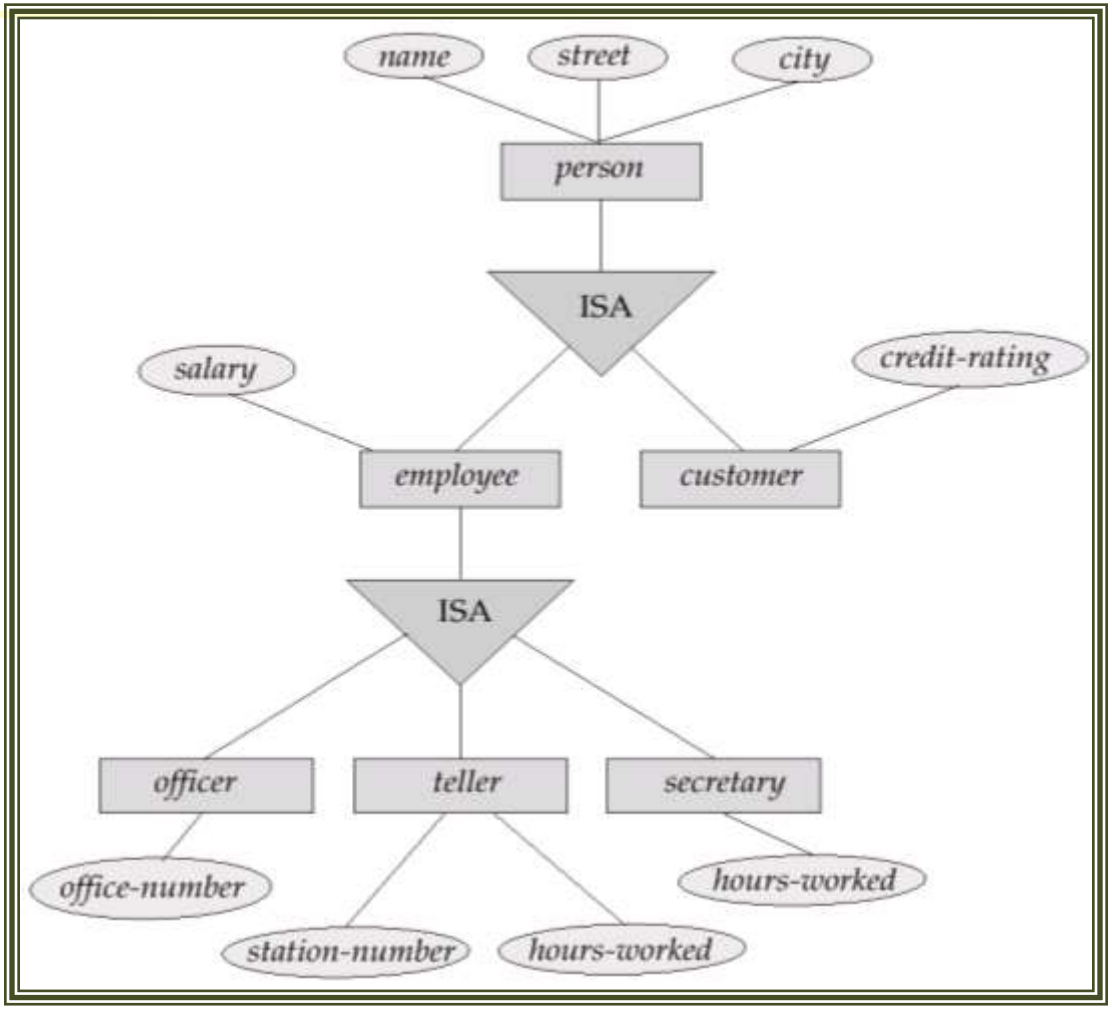


# Specialization

- Top-down design process; designate subgroupings within an entity set that are distinctive from other entities in the set
- These subgroupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set
- Depicted by a *triangle* component labeled ISA
  (E.g. *customer* "is a" *person*)

- **Attribute inheritance** – a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked

# Another specialization example



# Generalization

- … opposite to specialization
- A bottom-up design process – combine a number of entity sets that share the same features into a higher-level entity set
- Specialization and generalization are inversions of each other; they are represented in an E-R diagram in the same way; we use the terms interchangeably
- could have **many** levels of an IS-A hierarchy
- attribute inheritance

# Specialization and generalization (Cont.)

- Can have multiple specializations of an entity set based on different features
- E.g. *permanent-employee* vs. *temporary-employee*, in addition to *officer* vs. *secretary* vs. *teller*
  - Each particular employee would be
    - a member of one of *permanent-employee* or *temporary-employee*,
    - and also a member of one of *officer*, *secretary*, or *teller*
- The ISA relationship also referred to as **superclass - subclass** relationship

# Design Constraints on Specialization/Generalization

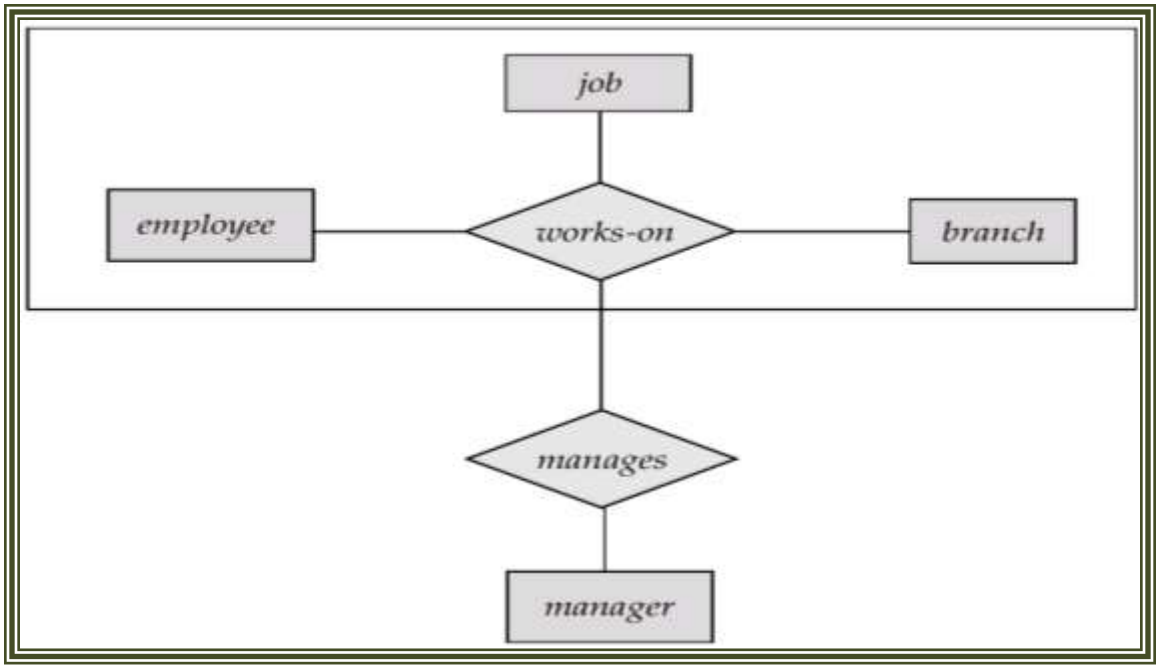
- The constraint on which entities can be members of a given lower-level entity set:
  - Condition-defined
    - E.g. all customers over 65 years are members of *senior-citizen* entity set; *senior-citizen* ISA *person*
  - User-defined
- The constraint on whether or not entities may belong to more than one lower-level entity sets within a single generalization
  - Disjoint
    - an entity can belong to only one lower-level entity set
    - Noted in E-R diagram by writing *disjoint* next to the ISA triangle
  - Overlapping
    - an entity can belong to more than one lower-level entity set

# Design Constraints on a Specialization/Generalization

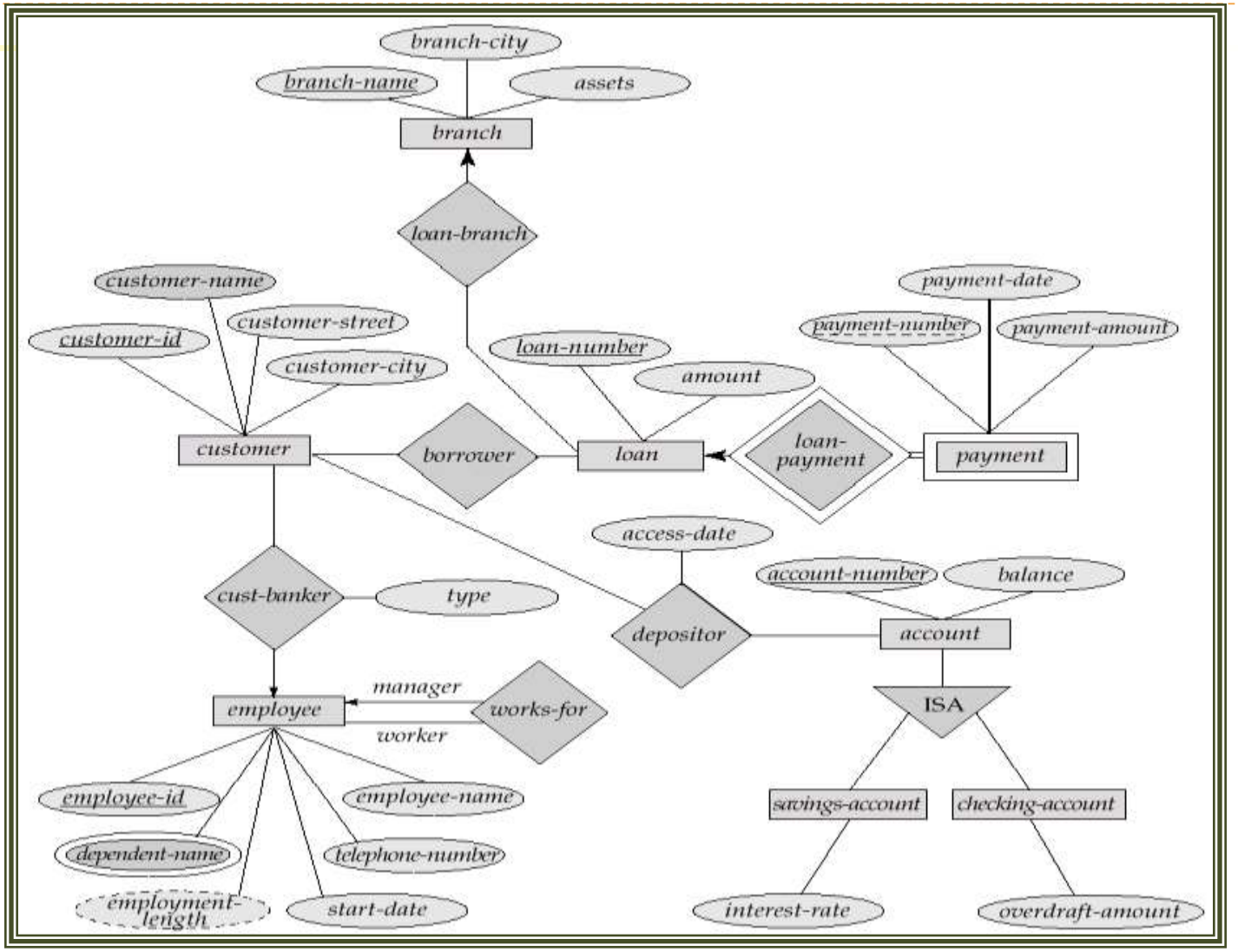
- Completeness constraint -- specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization
  - **total** : an entity must belong to one of the lower-level entity sets
  - **partial**: an entity need not belong to one of the lower-level entity sets

# Aggregation

- treats a relationship as an ‘abstract’ entity
- allows relationships between relationships
- rarely used



# E-R Diagram for a Banking Enterprise



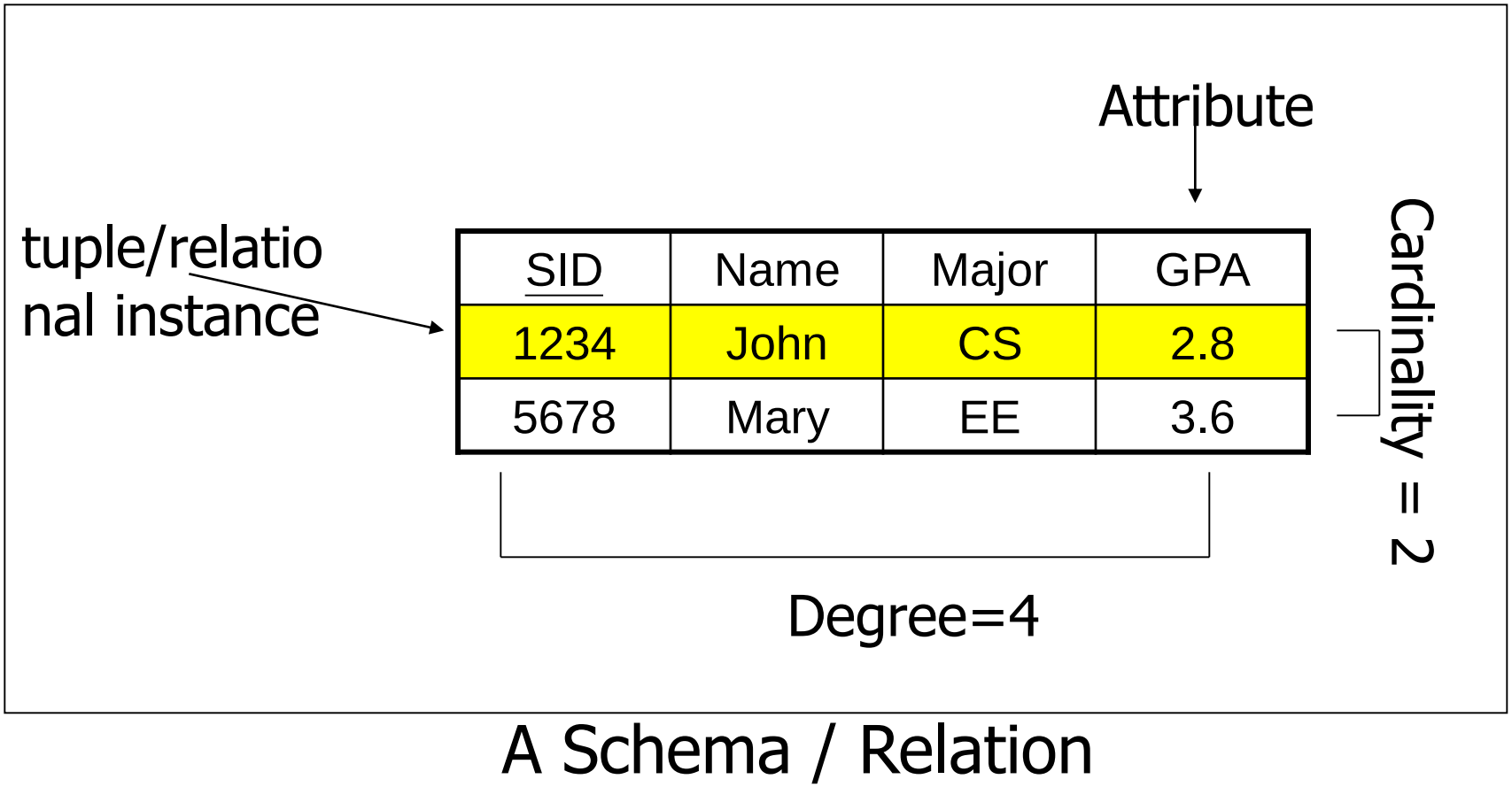
# Reduction of E-R Diagrams to Tables

# Relational Model

**Relational Model is made up of tables**

- A row of table = a relational instance/tuple
- A column of table = an attribute
- A table = a schema/relation
- Cardinality = number of rows
- Degree = number of columns

# Example


Attribute
tuple/relatio
nal instance
SID
Name
Major
GPA
1234
John
CS
2.8
5678
Mary
EE
3.6
Cardinality = 2
Degree=4
A Schema / Relation

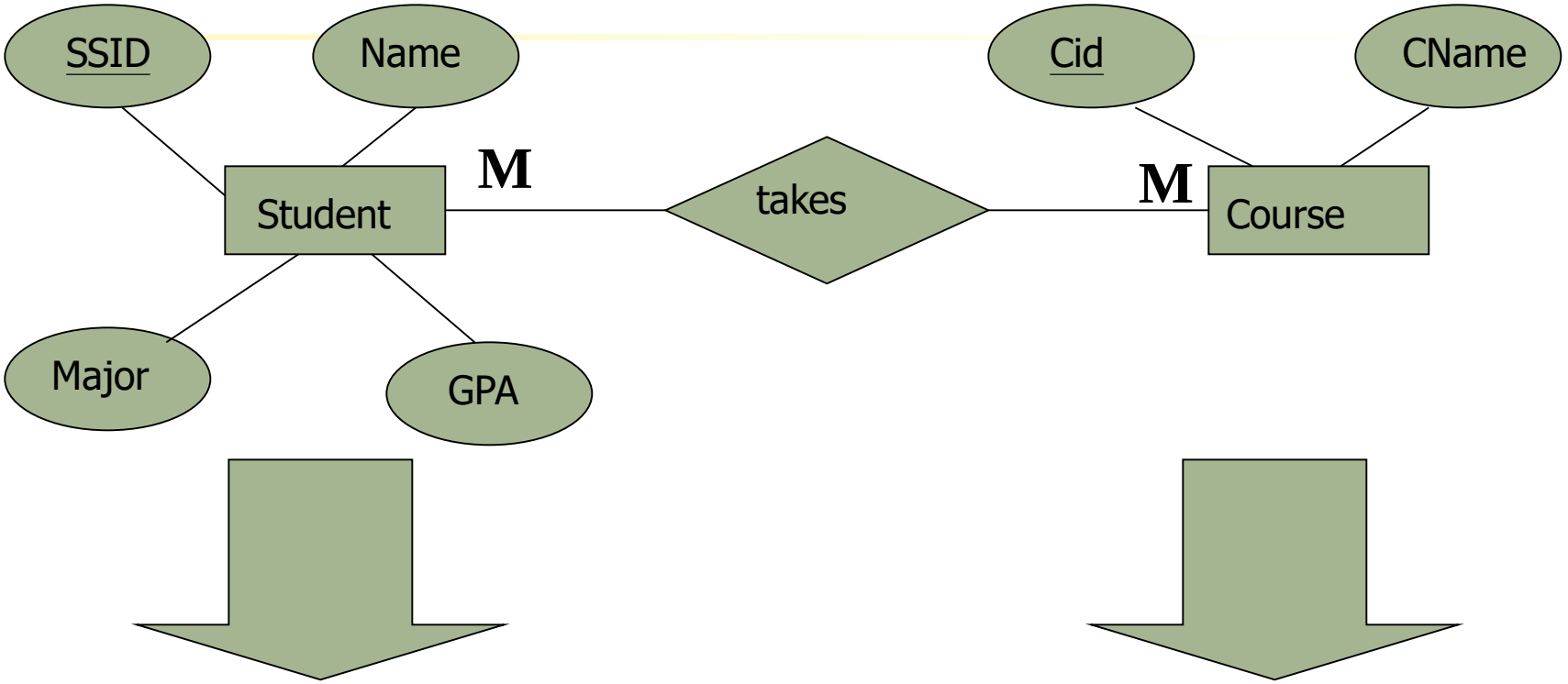
# From ER Model to Relational Model

So… how do we convert an ER diagram into a table??
Simple!!

Basic Ideas:

- Build a table for each entity set
- Make a column in the table for each attribute in the entity set
- Primary Key
- Build a table for each relationship set if necessary

## Example – Strong Entity Set



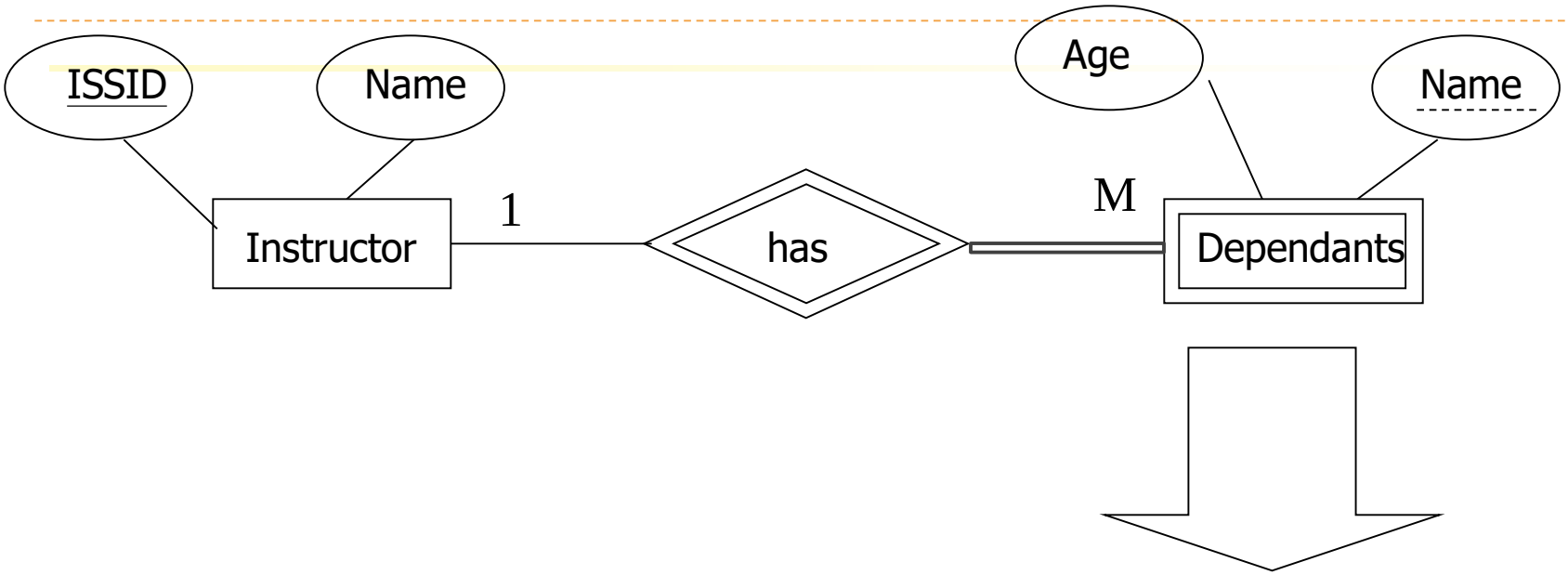

| SID | Name | Major | GPA |
|---|---|---|---|
| 1234 | John | CS | 2.8 |
| 5678 | Mary | EE | 3.6 |

| Cid | CName |
|---|---|
| C1 | C |
| C2 | Java |

# Representation of Weak Entity Set

- Weak Entity Set Cannot exists alone
- To build a table/schema for weak entity set
  - Construct a table with one column for each attribute in the weak entity set
  - Remember to include discriminator
  - Augment one extra column on the right side of the table, put in there the primary key of the Strong Entity Set (the entity set that the weak entity set is depending on)
  - Primary Key of the weak entity set = Discriminator + foreign key

## Example – Weak Entity Set




| Age | Name | Parent_ISSID |
|---|---|---|
| 10 | Bart | 1234 |
| 8 | Lisa | 5678 |

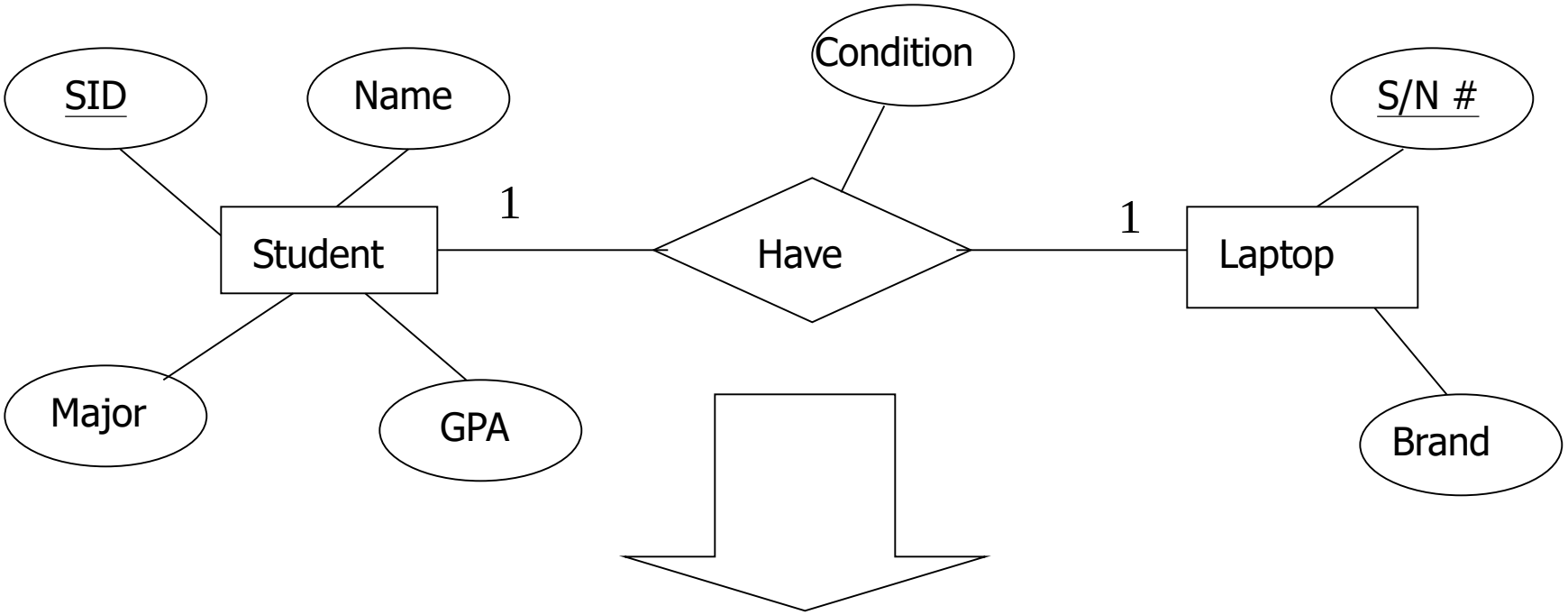
* Primary key of *Children* is *Parent_ISSID* + *Name*

# Representation of Relationship Set

- For one-to-one relationship sets, either side can be chosen to act as the "many" side
  - That is, extra attribute can be added to either of the tables corresponding to the two entity sets

## Example – One-to-One Relationship Set

1:1 Relationship



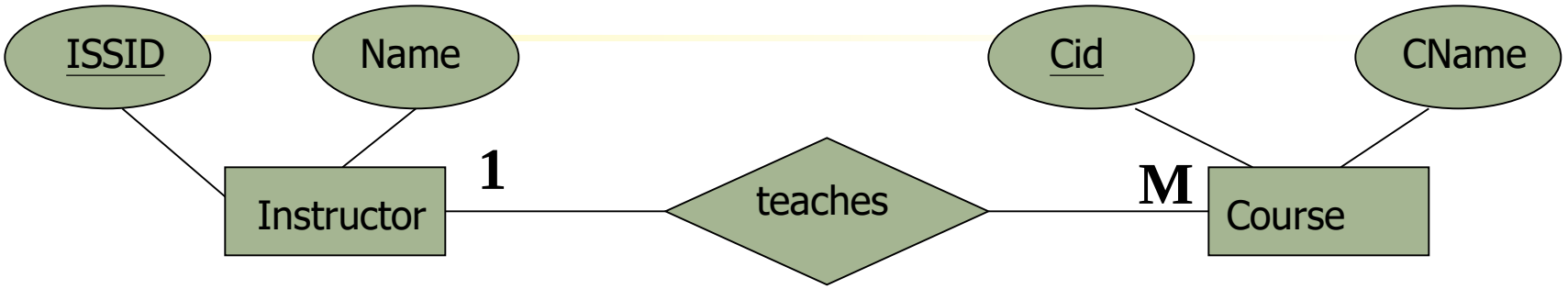

| SID | Name | Major | GPA | LP_S/N | Hav_Cond |
|---|---|---|---|---|---|
| 9999 | Bart | Economy | -4.0 | 123-456 | Own |
| 8888 | Lisa | Physics | 4.0 | 567-890 | Loan |

* Primary key can be either *SID* or *LP_S/N*

# Representing Relationship Set

- Many-to-one and one-to-many relationship sets can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side

## Example – Many-to-One Relationship Set



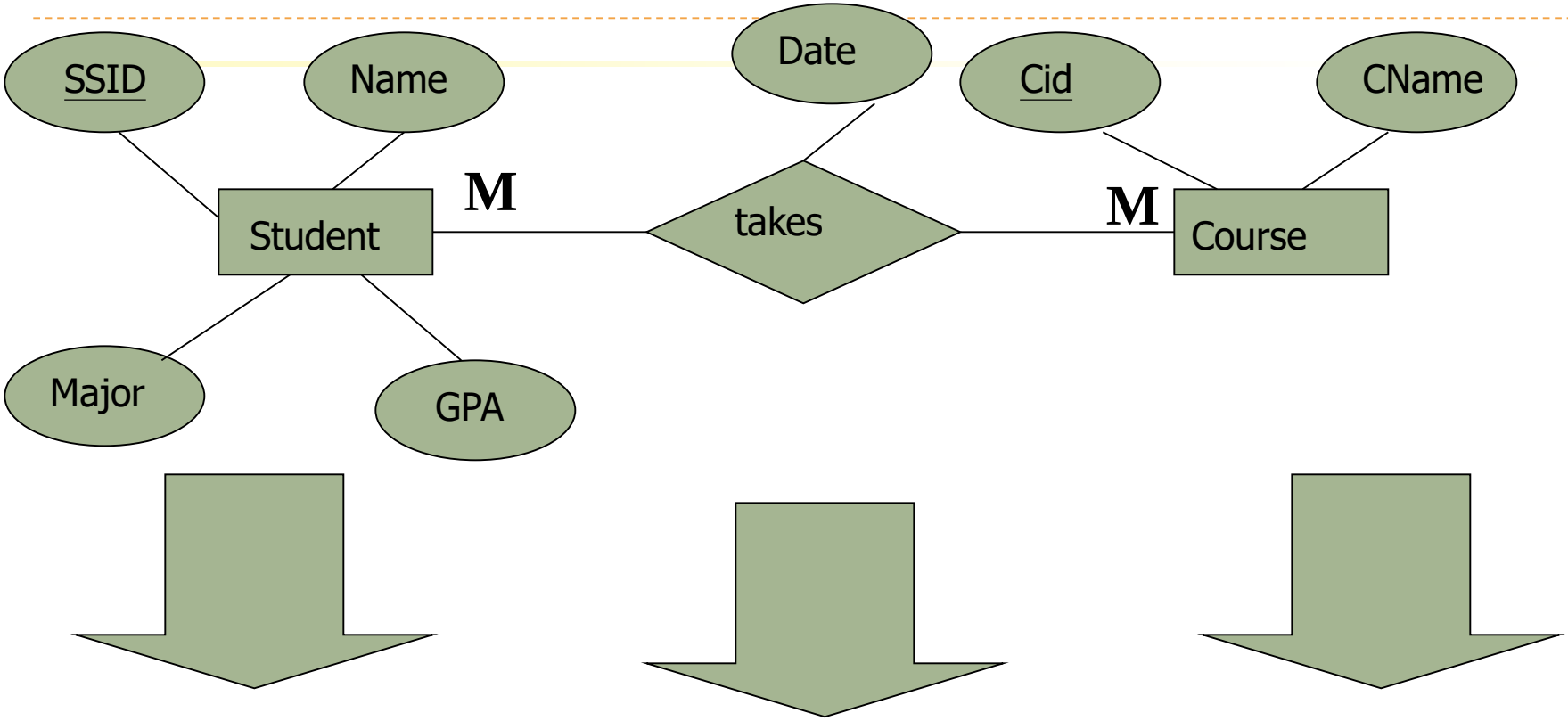

| ISSID | Name |
|---|---|
| 12 | Joan |
| 56 | Martha |

| Cid | CName | ISSID |
|---|---|---|
| C1 | C | 12 |
| C2 | Java | 56 |

# Representing Relationship Set

- For many-to-many relationship
  - Primary key of this new schema is the union of the foreign keys of both entity sets.

## Example – Many-to –Many Relation Ship set



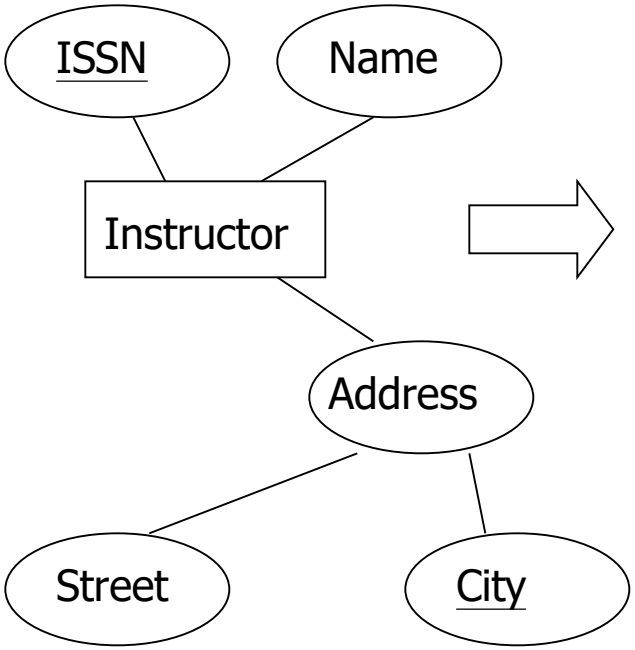

| SID | Name | Major | GPA |
|---|---|---|---|
| 1234 | John | CS | 2.8 |
| 5678 | Mary | EE | 3.6 |

| SID | Cid | Date |
|---|---|---|
| 1234 | C1 | 15/01/2009 |
| 1234 | C2 | 20/12/2010 |

| Cid | CName |
|---|---|
| C1 | C |
| C2 | Java |

# Representing Composite Attribute

- One column for each component attribute
- NO column for the composite attribute itself

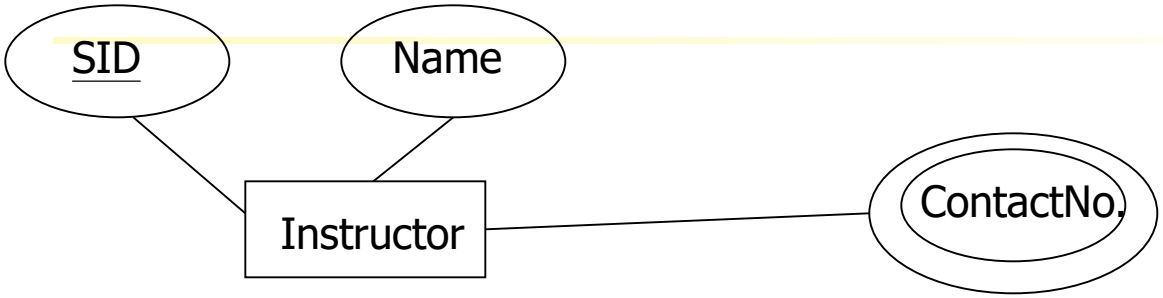



| ISSN | Name | Street | City |
|---|---|---|---|
| 99 | Dr. Smith | 50 1st St. | Fake City |
| 88 | Dr. Lee | 1 B St. | San Jose |

# Representing Multivalued Attribute

- For each multivalue attribute in an entity set/relationship set
  - Build a new relation schema with two columns
  - One column for the primary keys of the entity set/relationship set that has the multivalue attribute
  - Another column for the multivalue attributes. Each cell of this column holds only one value. So each value is represented as an unique tuple
  - Primary key for this schema is the union of all attributes

# Example – Multivalued attribute




The primary key for this table is Student_SID + Children, the union of all attributes

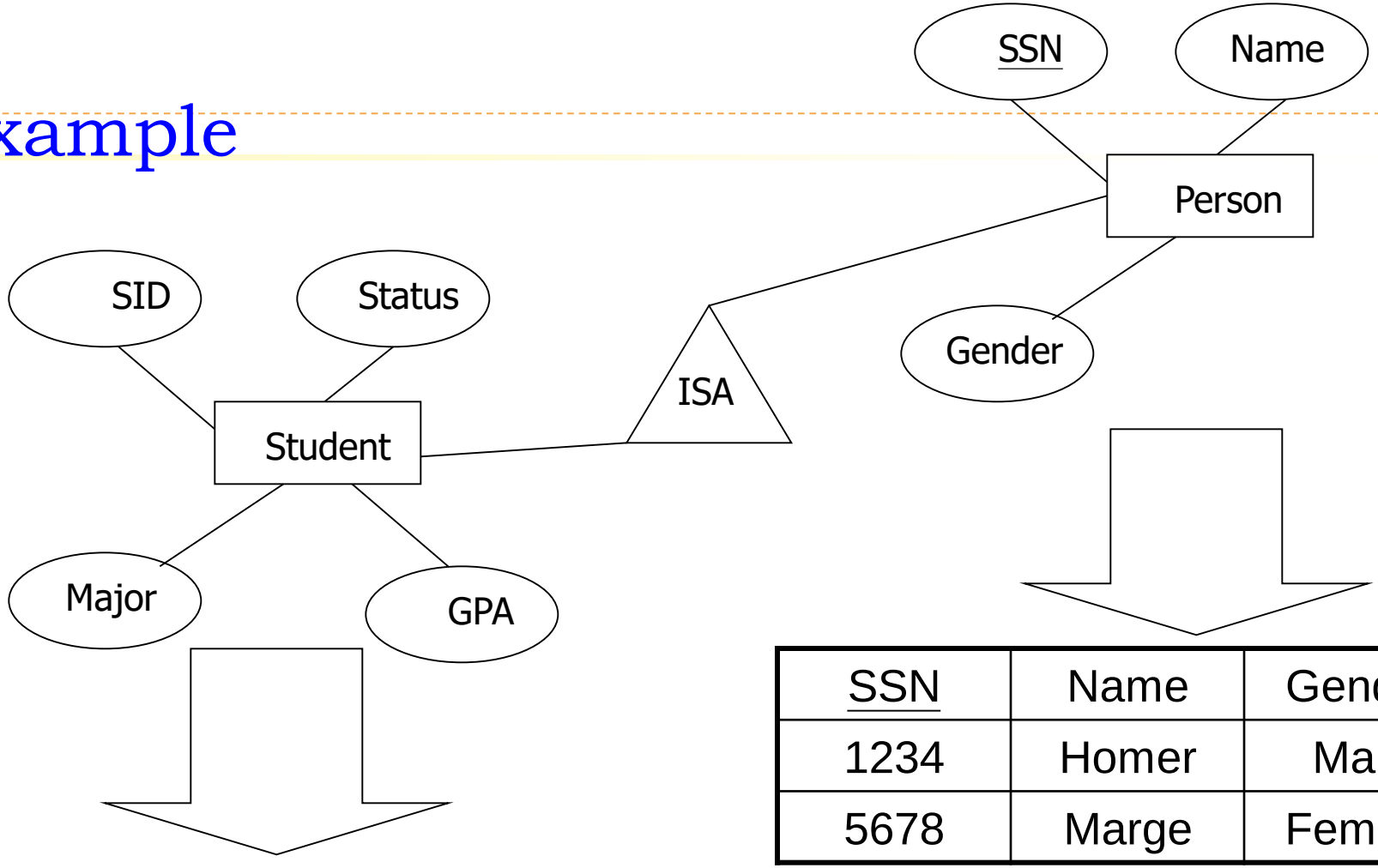
| ISSID | Name | Major | GPA |
|---|---|---|---|
| 12 | Joan | CS | 2.8 |
| 56 | Martha | EE | 3.6 |

| ISSID | ContactNo |
|---|---|
| 12 | 222222 |
| 12 | 345677 |
| 56 | 777799 |
| 56 | 998765 |
| 56 | 333455 |

# Representing Class Hierarchy

- Two general approaches depending on disjointness and completeness
  - For non-disjoint and/or non-complete class hierarchy:
    - create a table for each super class entity set according to normal entity set translation method.
    - Create a table for each subclass entity set with a column for each of the attributes of that entity set plus one for each attributes of the primary key of the super class entity set
    - This primary key from super class entity set is also used as the primary key for this new table

# Example




| SSN | Name | Gender |
|---|---|---|
| 1234 | Homer | Male |
| 5678 | Marge | Female |

| SSN | SID | Status | Major | GPA |
|---|---|---|---|---|
| 1234 | 9999 | Full | CS | 2.8 |
| 5678 | 8888 | Part | EE | 3.6 |

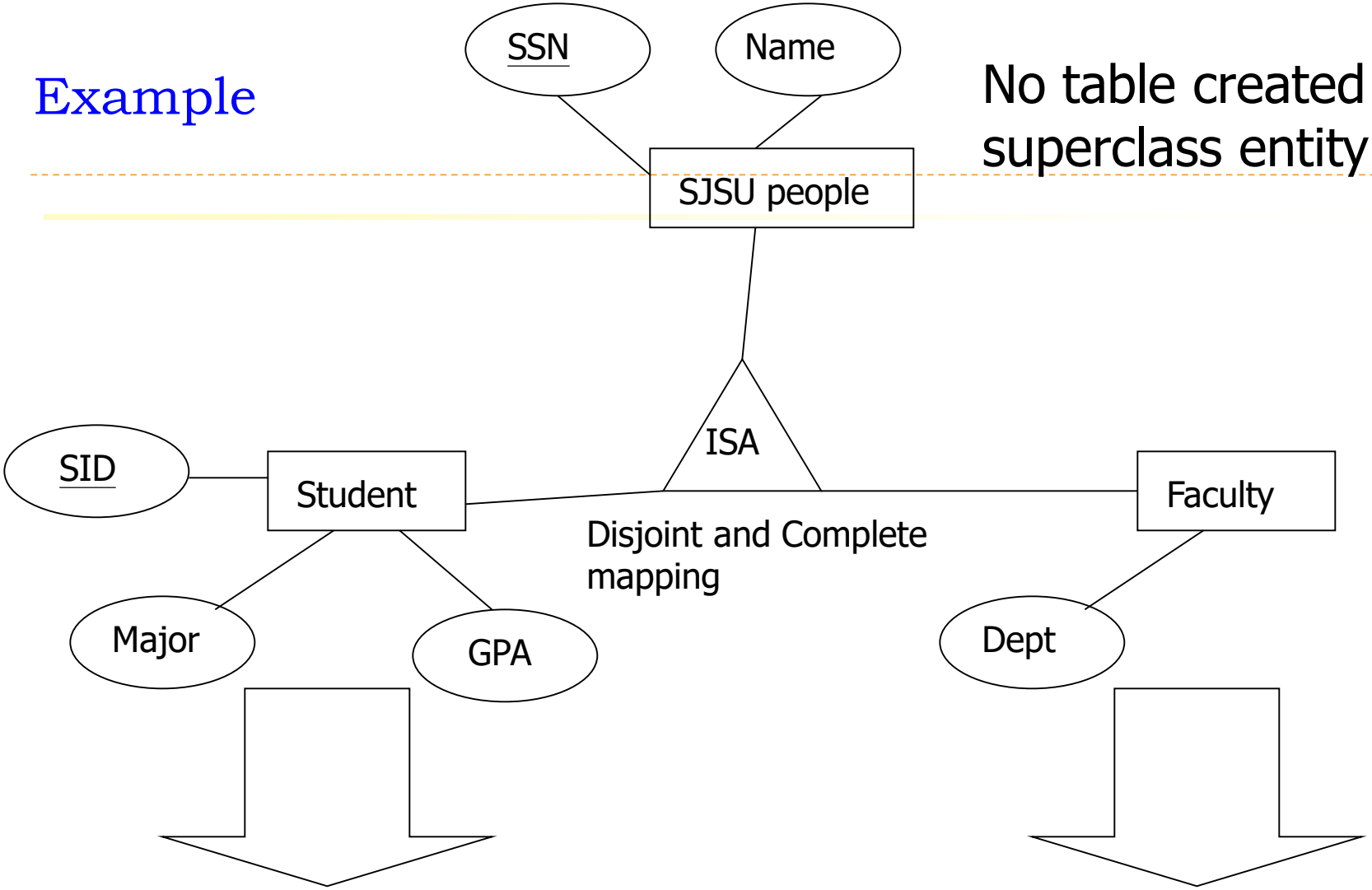
# Representing Class Hierarchy

- Two general approaches depending on disjointness and completeness
  - For disjoint **AND** complete mapping class hierarchy:
  - DO NOT create a table for the super class entity set
  - Create a table for each subclass entity set include all attributes of that subclass entity set and attributes of the superclass entity set

  - Simple and Intuitive enough, need example?

# Example

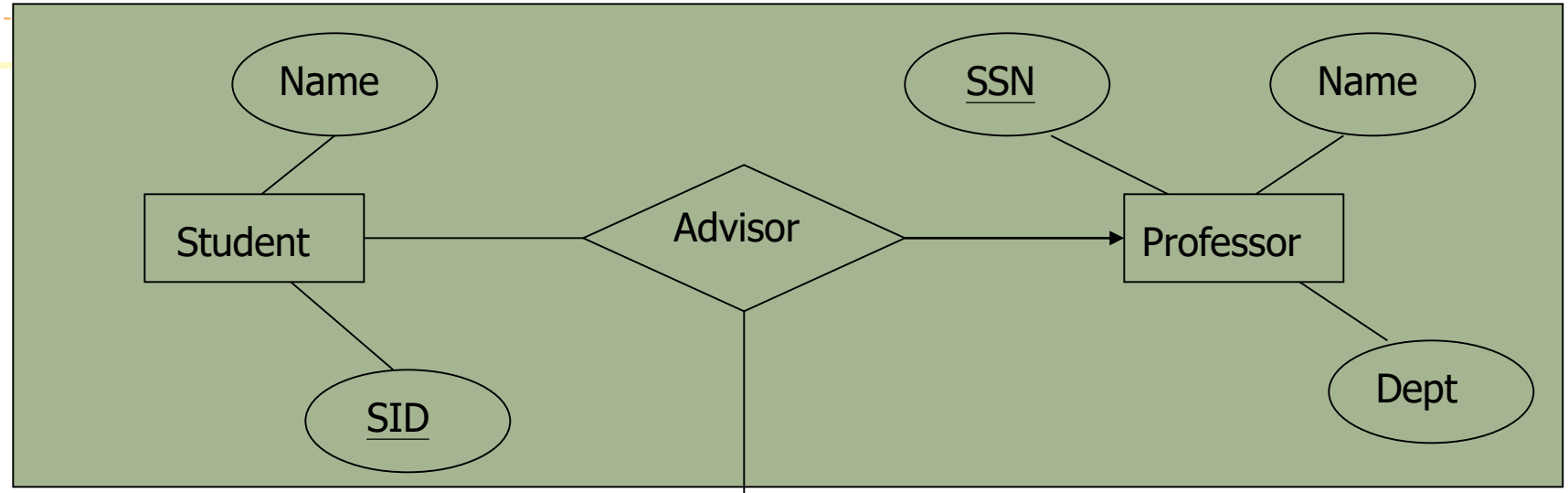
No table created for superclass entity set

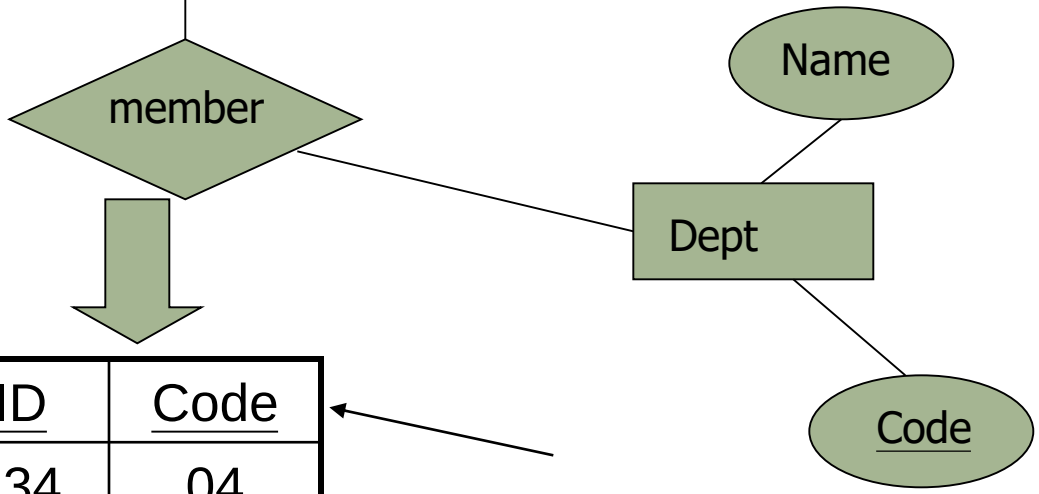



| SSN | Name | SID | Major | GPA |
|---|---|---|---|---|
| 1234 | John | 9999 | CS | 2.8 |
| 5678 | Mary | 8888 | EE | 3.6 |

| SSN | Name | Dept |
|---|---|---|
| 1234 | Homer | C.S. |
| 5678 | Marge | Math |

# Representing Aggregation




Primary Key of *Advisor*

| SID | Code |
|---|---|
| 1234 | 04 |
| 5678 | 08 |

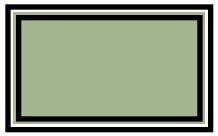
Primary key of *Dept*

# Representing aggregation

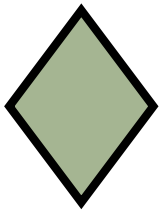
- make table, with primary keys of all involved entities

# Summary - cont'd

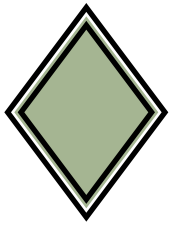
**(strong) entity set**



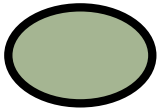
**weak entity set**



**relationship set**

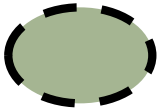


**identifying rel. set for weak entity**



**attribute**

**multivalued attribute**

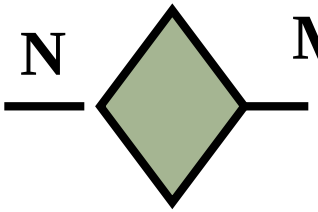


**derived attribute**

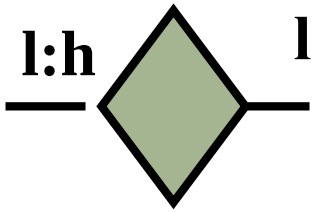
# Summary - cont'd


**primary key**


**discriminator**

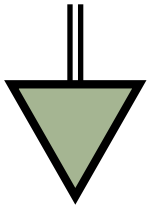



**cardinalities**




**cardinalities with limits**

**IS-A**



**total generalization**