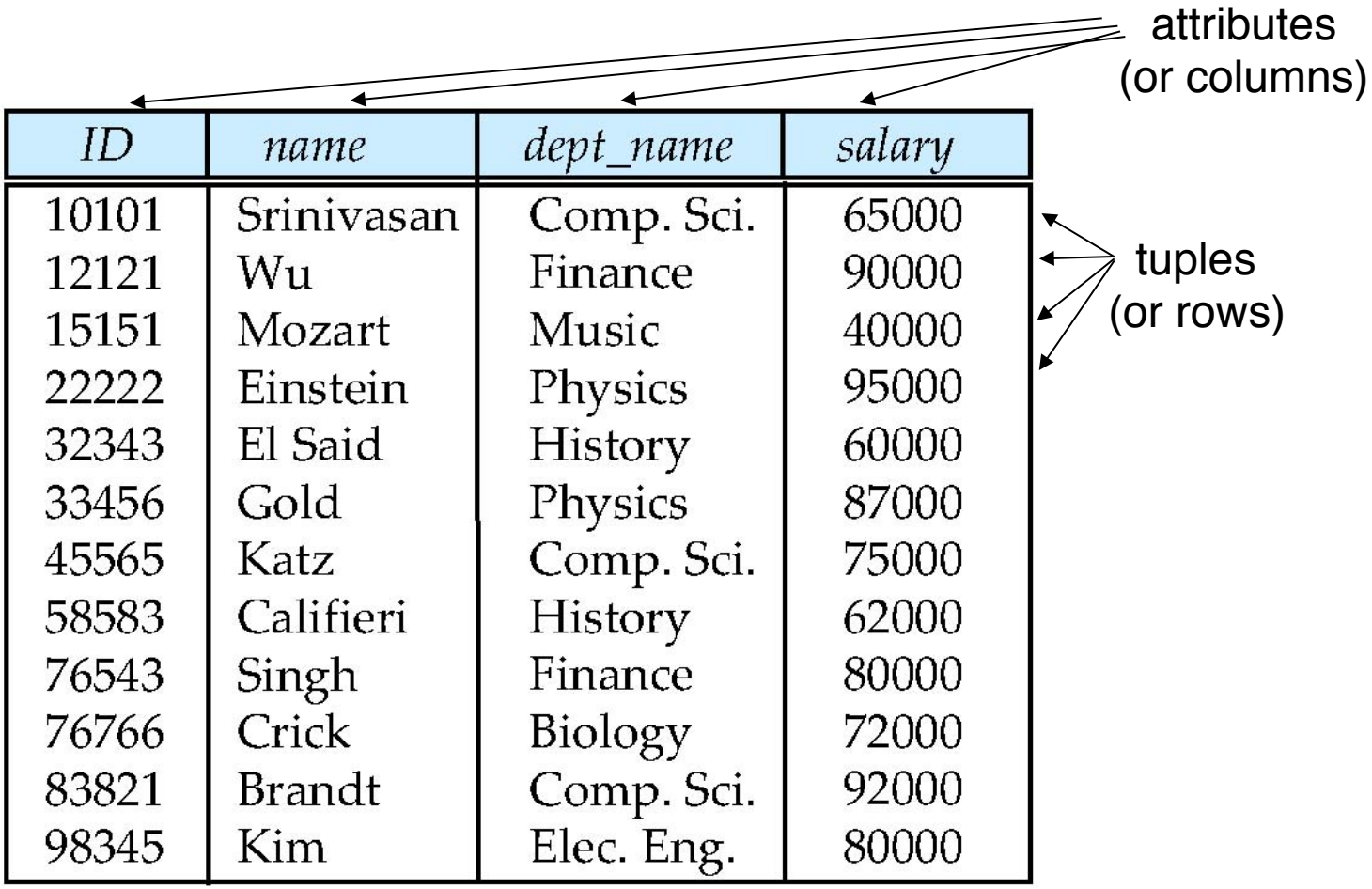
Relational Model and the Entity-Relationship (ER) Model

Professor Jessica Lin

Relations



ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

A relational database consists of a collection of tables (relations).

A **relation** consists of a **relational schema** and a **relational instance**.

A **relation schema** is essentially a list of column names with their data types. In this case…

instructor(ID: string, *name* : string, *dept_name*: string, *salary* : integer)

A **relation instance** is made up of zero or more **tuples** (rows, records)

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

A schema specifies a relation's name.

instructor(ID: string, *name* : string, *dept_name*: string, *salary* : integer)

A schema also specifies the name of each **field**, and its domain.

Fields are often referred to as columns, attributes, dimensions

A minor, but important point about relations, the fields are unordered.

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

This is not a problem, since we refer to fields by name.

Also, the tuples are unordered too!

Note that every tuple in our instance is unique. This is not a coincidence. The definition of relation demands it.

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

The number of fields is called the **degree** (or **arity**, or dimensionality of the relation).

Below we have a table of degree 4.

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

The number of tuples = **cardinality** of the relation

Of course, we don't count the row that has the labels!

To the right we have a table of cardinality 12.

Attribute Types

- The set of allowed values for each attribute is called the **domain** of the attribute
- The special value ***null*** is a member of every domain
- Attribute types:
  - **Simple** and **composite** attributes.
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_numbers*
  - **Derived** attributes
    - Can be computed from other attributes
    - Example: age, given date_of_birth

Database

- A database consists of multiple relations.
- Information about an enterprise is broken up into parts:

  instructor
  student
  advisor

- Bad design:

  univ (*instructor_ID, name, dept_name, salary, student_ID*, ..)

  results in

  – repetition of information (e.g., two students have the same instructor)

  – the need for null values (e.g., represent an student with no advisor)

- Next we will learn to to design a (good) database schema.

The E-R Model

The Entity-Relationship Model

The **E-R** (entity-relationship) data model views the real world as a set of basic **objects** (entities) and **relationships** among these objects.

It is intended primarily for the DB design process by allowing the specification of an **enterprise scheme**. This represents the overall logical structure of the DB.

Entities and Entity Sets

•An **entity** is an object that exists and is distinguishable from other objects. For instance, Michelle Lee with S.S.N. 890-12-3456 is an entity, as she can be uniquely identified as one particular person in the universe.

•An entity may be **concrete** (a person or a book, for example) or **abstract** (like a course or a disease).

•An **entity set** is a set of entities of the same type (e.g., all persons having an account at a bank).

•Entity sets **need not be disjoint**. For example, the entity set *Student* (all students in a university) and the entity set *professor* (all professors in a university) may have members in common. (i.e. a computer science professor might take a class in anthropology).

Entities and Entity Sets Continued

• An entity is represented by a set of **attributes**. (e.g. *name*, *SSN*, *Phone-Num* for "customer" entity.)

• The **domain** of the attribute is the set of permitted values (e.g. the telephone number must be ten positive integers).

• Formally, an attribute is a **function** which maps an entity set into a domain.

• Every entity is described by a set of (attribute, data value) pairs. There is one pair for each attribute of the entity set.

• e.g. a particular *student* entity is described by the set {(name, Lee), (SSN, 890-123-456), (street, Blaine), (city, Riverside)}.

Entity Sets *instructor* and *student*

instructor_ID	instructor_name
76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

instructor

student-ID	student_name
98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Relationships

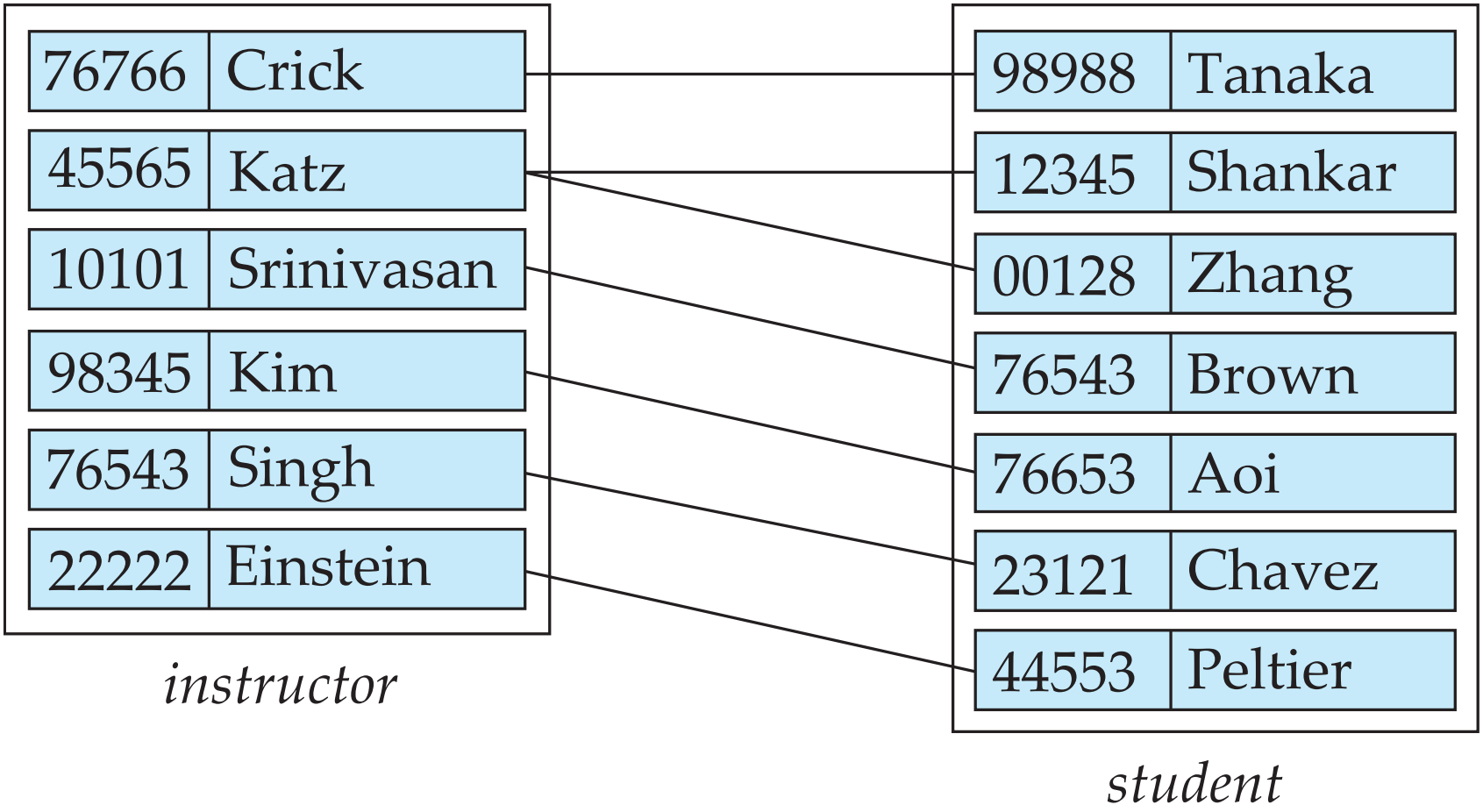
- A **relationship** is an association among several entities.

e.g. “advising” relationship describes an instructor advising a student

- A **relationship set** is a set of relationships of the same type.

- Binary relationships are the most common; however, it’s possible to have relationship between more than two entities.
    - “degree of relationship”

Relationship Set *advisor*



Keys

Differences between entities must be expressed in terms of attributes.

• A **superkey** is a set of one or more attributes which, taken collectively, allow us to identify uniquely an entity in the entity set.
• For example, in the entity set *student*, *{name*, *S.S.N.}* is a superkey.
• Note that *name* alone is not, as two students could have the same name.
• A superkey may contain extraneous attributes, and we are often interested in the smallest superkey. A superkey for which no subset is a superkey is called a **candidate key**.

Name	**S.S.N.**
Lisa	111-11-1111
Bart	222-22-2222
Lisa	333-33-3333
Sue	444-44-4444

We can see that {*Name*,*S.S.N.*} is a **superkey**

In this example, *S.S.N.* is a **candidate** key, as it is minimal, and uniquely identifies a students entity.

Keys Cont.

•A **primary key** is a candidate key (there may be more than one) chosen by the DB designer to identify entities in an entity set.

In the example below…

Give an example of a superkey.
Give an example of a candidate key.
What's the logical choice of primary key?

Make	Model	Owner	State	License #	VIN #
Ford	Focus	Mike	CA	SD123	34724
BMW	Z4	Joe	CA	JOE	55725
Ford	Escort	Sue	AZ	TD4352	75822
Honda	Civic	Bert	CA	456GHf	77924

Keys Cont.

•A **primary key** is a candidate key (there may be more than one) chosen by the DB designer to identify entities in an entity set.

In the example below…

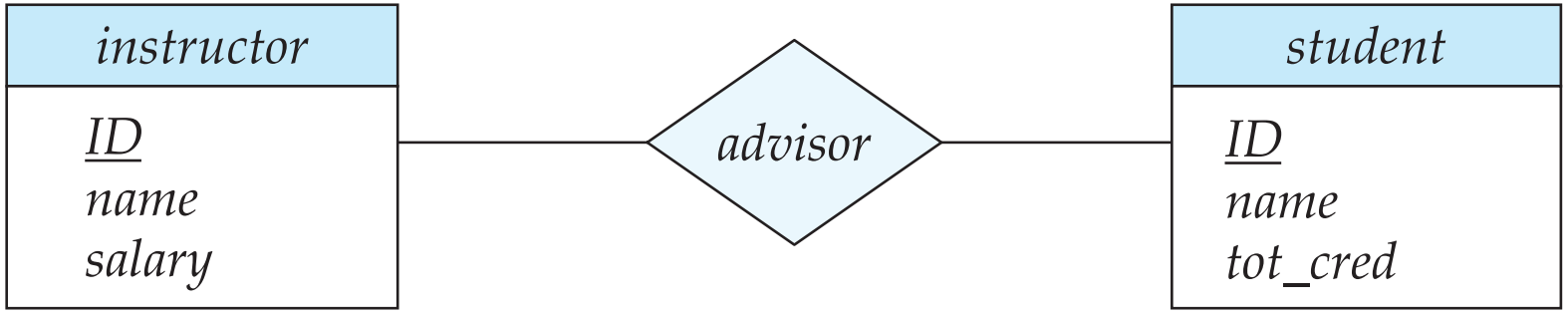
{Make,Model,Owner,State,License#,VIN#} is a superkey
{State,License#,VIN#} is a superkey
{Make,Model,Owner} is *not* a superkey

{State,License#} is a candidate key
{VIN#} is a candidate key

VIN# is the logical choice for **primary key**

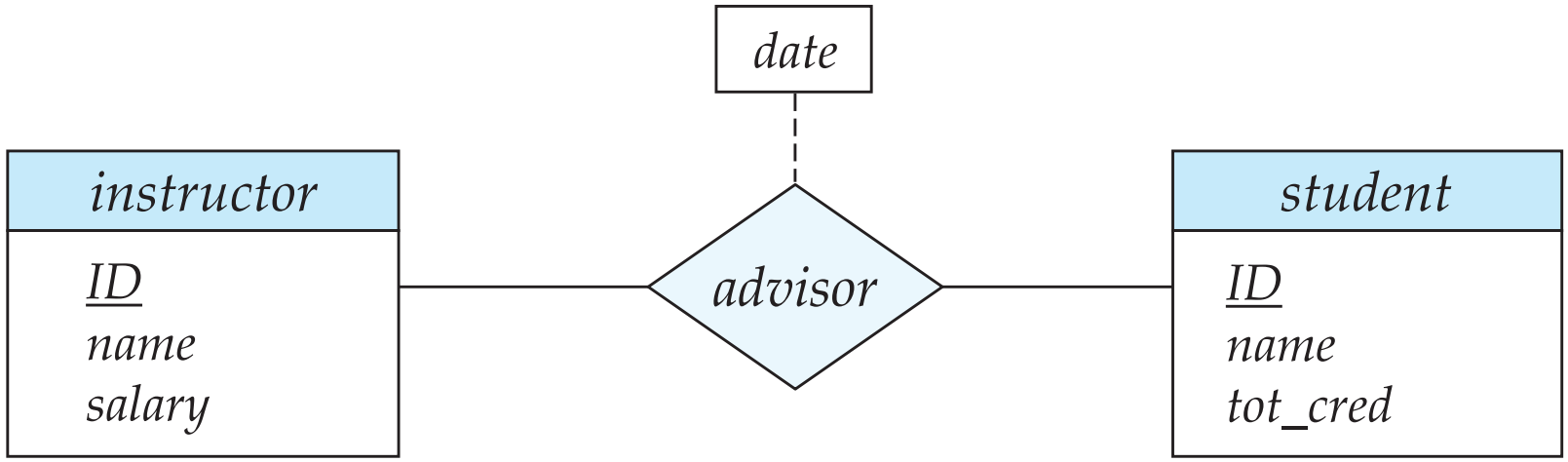
Make	**Model**	**Owner**	**State**	**License #**	**VIN #**
Ford	Focus	Mike	CA	SD123	34724
BMW	Z4	Joe	CA	JOE	55725
Ford	Escort	Sue	AZ	TD4352	75822
Honda	Civic	Bert	CA	456GHf	77924

E-R Diagrams



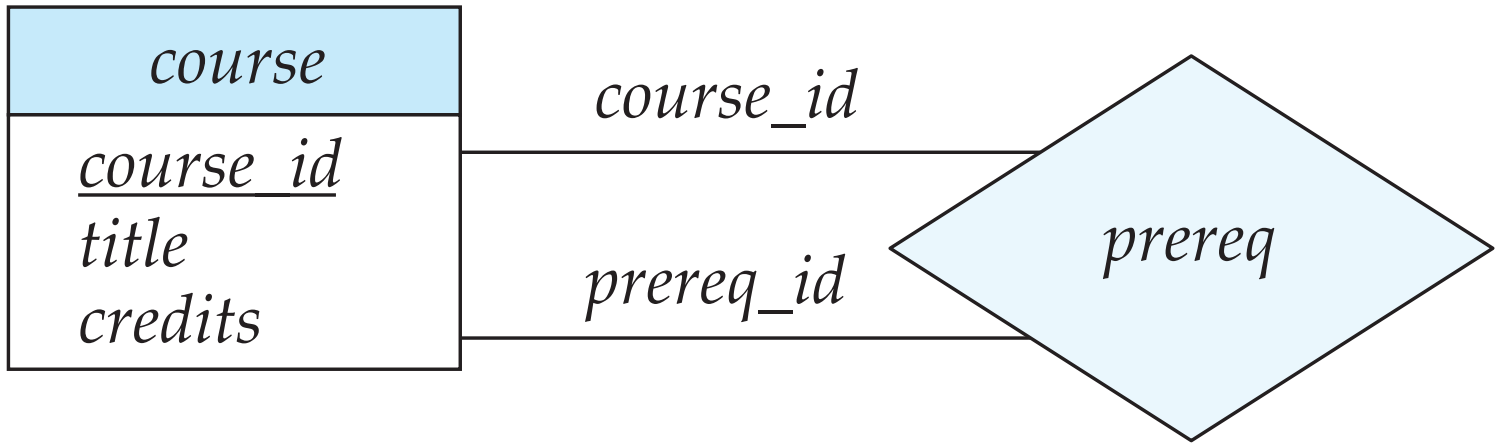
- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Attributes listed inside entity rectangle
- Underline indicates primary key attributes

Relationship Sets with Attributes



Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a "role" in the relationship
- The labels "*course_id*" and "*prereq_id*" are called **roles**.



Key Constraints

We can also use arrows to indicate **key constraints** (often simply referred to as **constraints**)

Suppose the university has the following rule: A student is allowed to be advised by at most one professor. However, a professor is allowed to advise more than one student.

This is an example of a **one-to-many relationship** **(between instructor and student)**, that is, many students can be advised by one professor, but each student can only have (at most) one advisor. We can represent this with an arrow as shown below.

Key Constraints Continued

•There are four possible **key constraints,** they express the number of entities to which another entity can be associated via a relationship. For binary relationship sets between entity sets A and B, the mapping cardinality must be one of:

1.One-to-one: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

2.One-to-many: An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.

3.Many-to-one: An entity in A is associated with at most one entity in B. An entity in B is associated with any number in A.

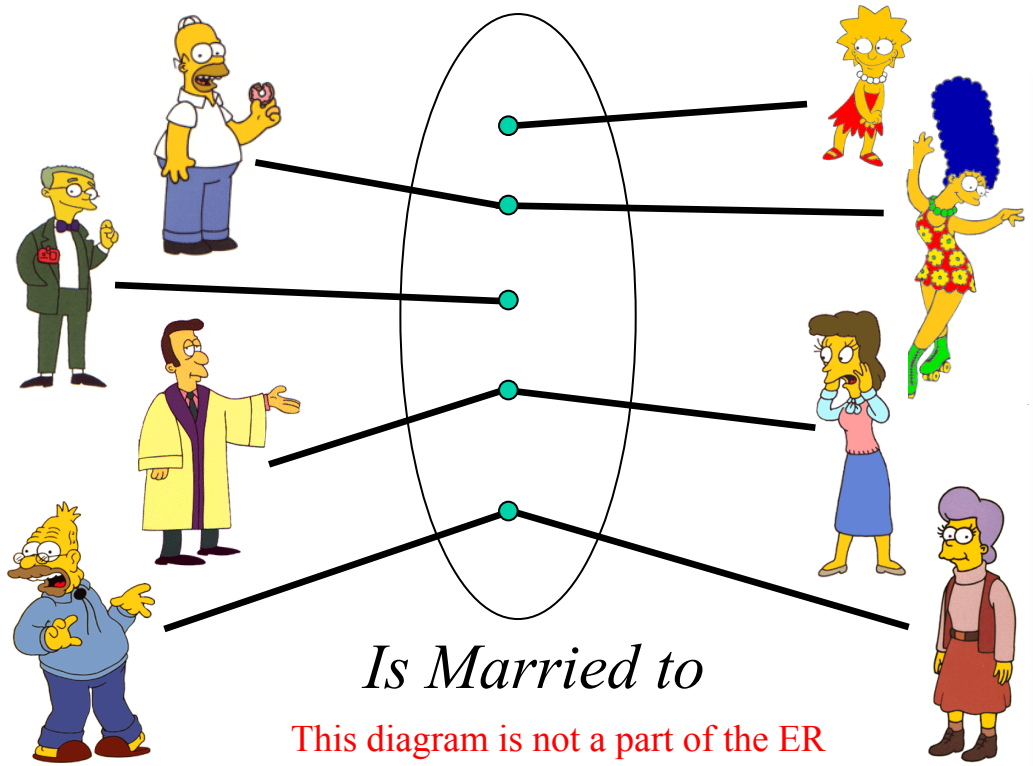
4.Many-to-many: Entities in A and B are associated with any number from each other.

The appropriate **key constraint** for a particular relationship set depends on the real world being modeled.

Key Constraints: Examples

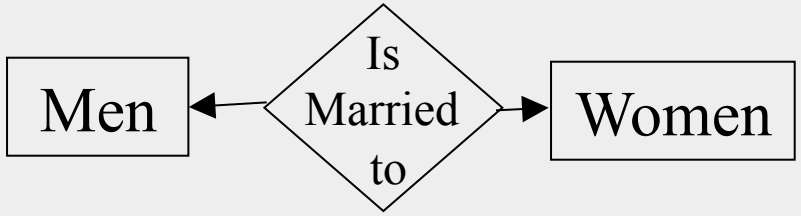
•**One-to-one**: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

A man may be married to at most one woman, and a woman may be married to at most one man (both men and women can be unmarried)



Is Married to

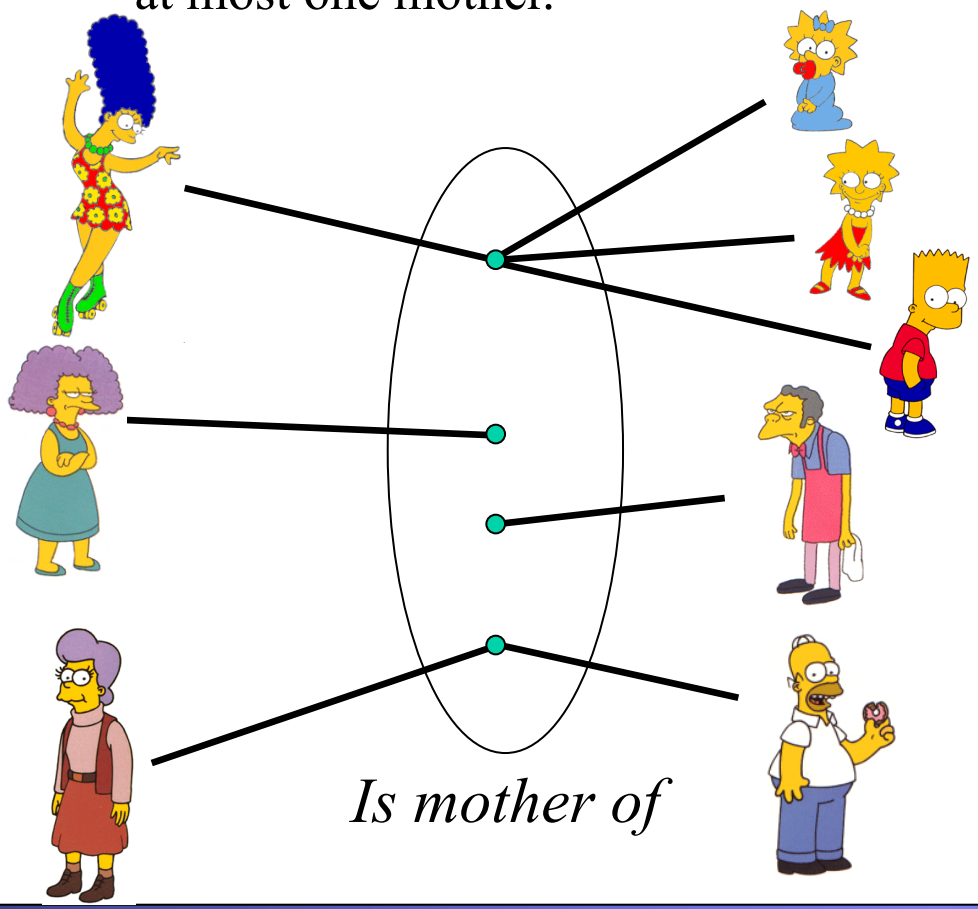
This diagram is not a part of the ER model! It is just an intuitive picture to explain a concept



Key Constraints: Examples

•**One-to-many**: An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.

A woman may be the mother of many (or no) children. A person may have at most one mother.

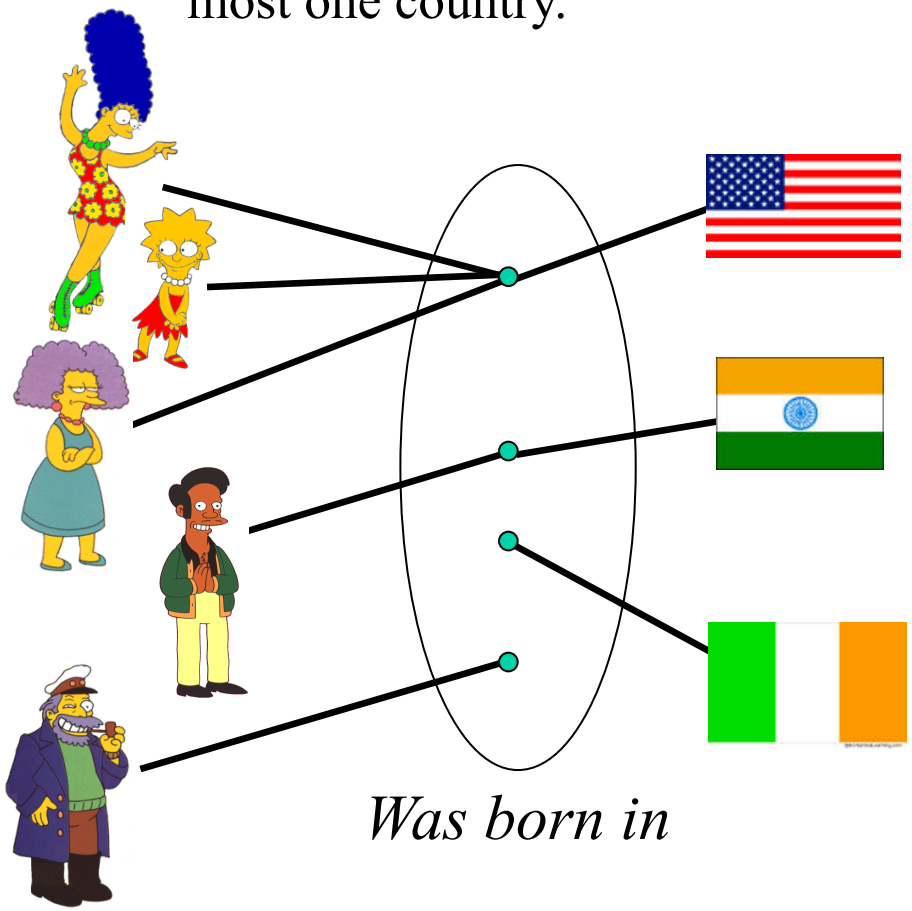


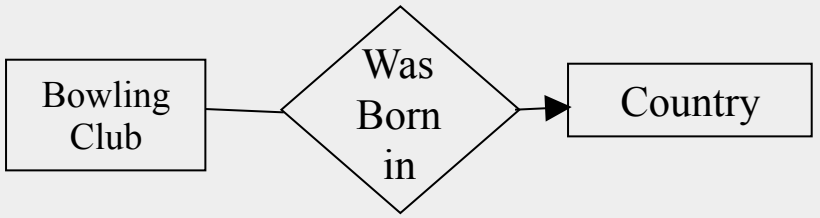
Note that this example is not saying that Moe does not have a mother, since we know as a biological fact that everyone has a mother. It is simply the case that Moe's mom is not a member of the Women's club.

Key Constraints: Examples

•**Many-to-one**: An entity in A is associated with at most one entity in B. An entity in B is associated with any number in A.

Many people can be born in any county, but any individual is born in at most one country.



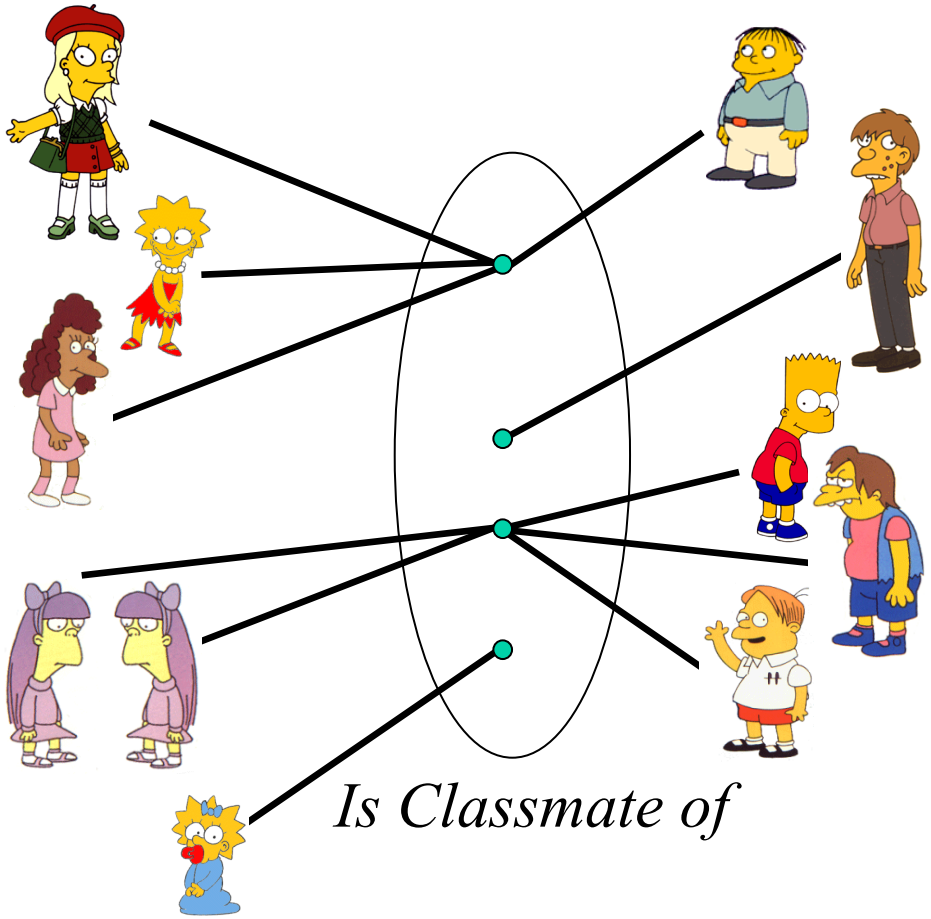


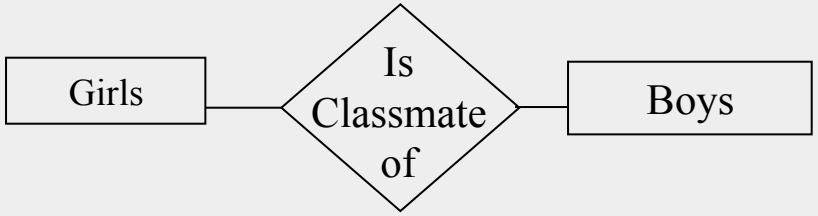
Note that we are not saying that the Sea Captain was not born in any country, he almost certainly was, we just don't know which country, or it is not in our Country entity set.

Also note that we are not saying that no one was born in Ireland, it is just that no one in the Bowling Club was.

Key Constraints: Examples

•**Many-to-many**: Entities in A and B are associated with any number from each other.





Key Constraints

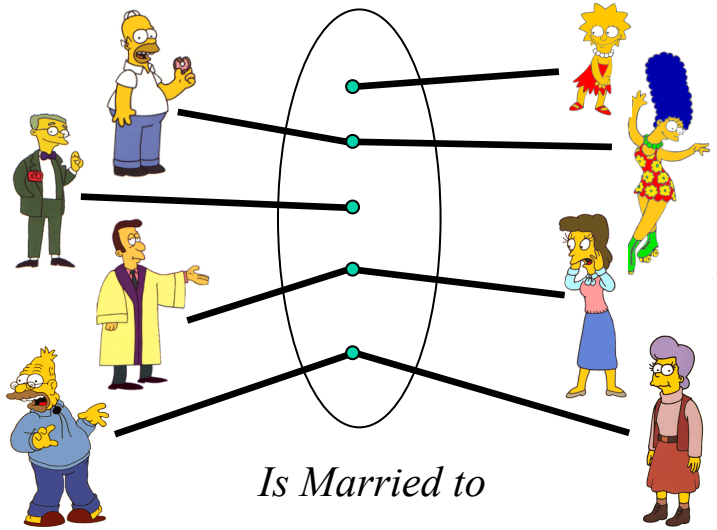
- The arrow positioning is simple once you get it straight in your mind, so do some examples. Think of the arrow head as pointing to the entity that "one" refers to.
- A line with an arrow is equivalent to having the constraint "0:1"
- Some people use the term "**Mapping Cardinalities**" or "**Multiplicity**" to refer to key constraints.

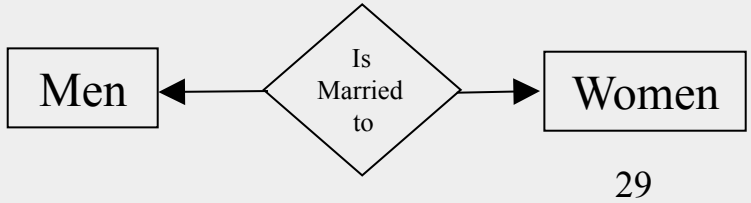
Participation Constraints

Earlier, we saw an example of a one-to-one key constraint, noting that a man may be married to at most one woman, and a woman may be married to at most one man (both men and women can be unmarried).

Suppose we want to build a database for the "Springfield Married Persons Association". In this case *everyone* must be married! In database terms their participation must be **total**. (the previous case that allows unmarried people is said to have **partial** participation.)

How do we represent this with ER diagrams? (answer on next slide)

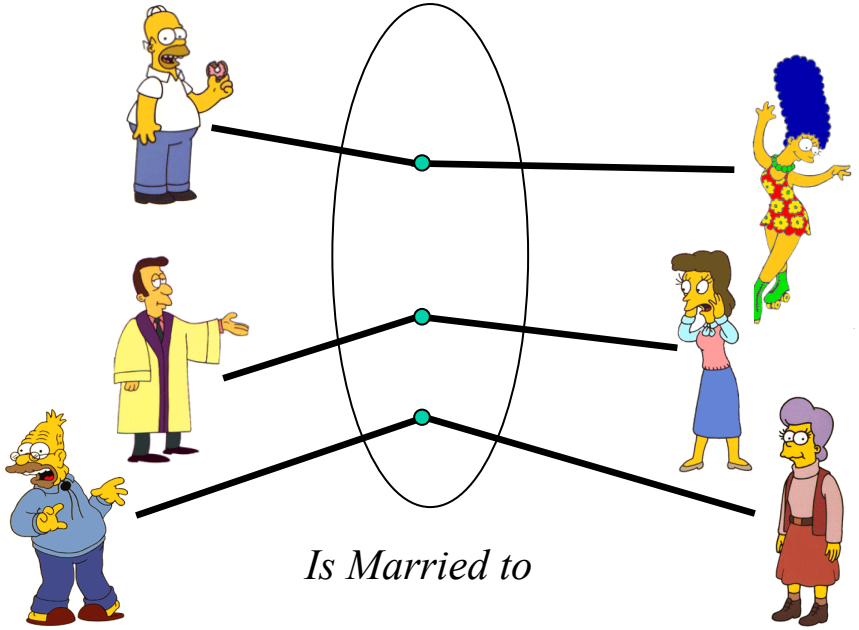


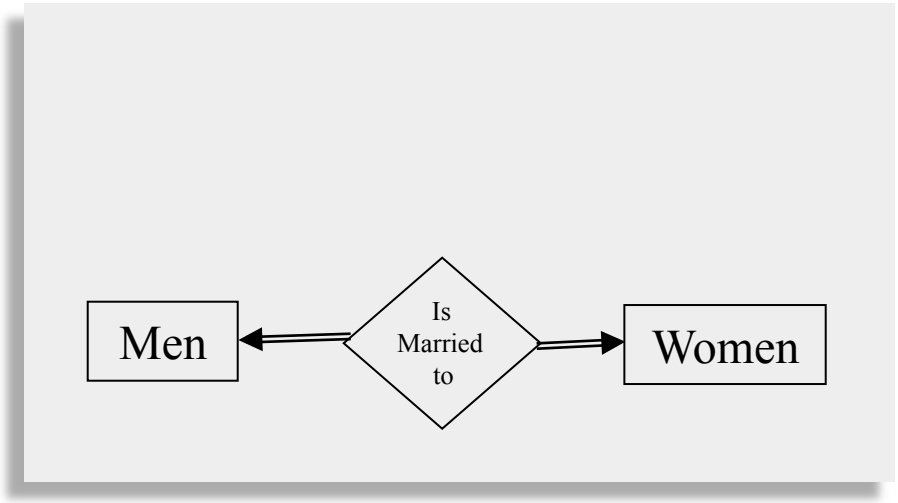


Participation Constraints Cont.

Participation Constraints are indicated by double lines in ER diagrams (some textbooks use bold lines).

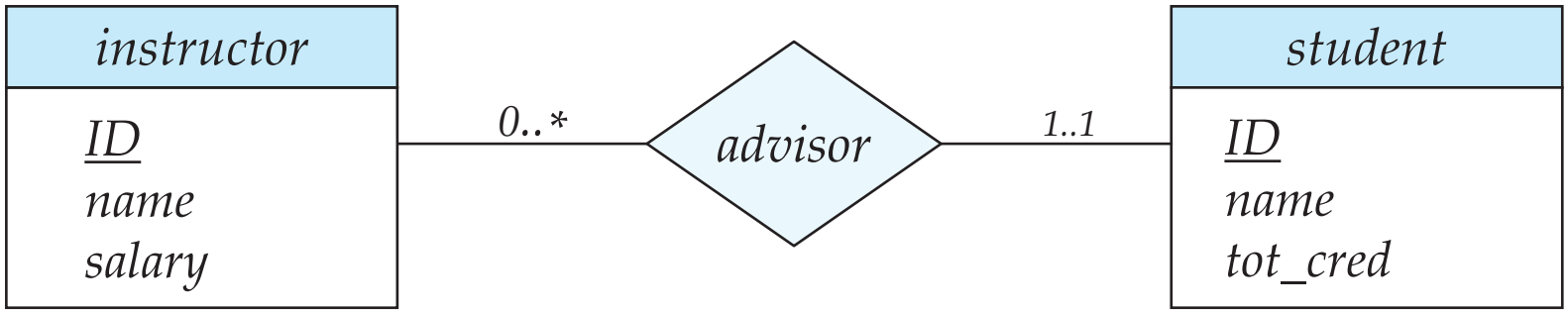
We can use double lines (to indicate participation constraints), and arrow lines (to indicate key constraints) independently of each other to create an expressive language of possibilities.





Alternate Notation for Cardinality Limit

- Cardinality limit can also express participation constraint



Ternary Relationship