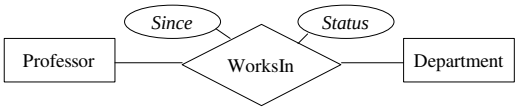
Representation in SQL

- Each role of relationship type produces a foreign key in corresponding relation
  - Foreign key references table corresponding to entity type from which role values are drawn

16

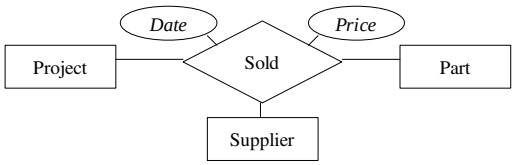
Example 1




```
CREATE TABLE WorksIn (
   Since DATE,              -- attribute
   Status CHAR (10),        -- attribute
   ProfId INTEGER,          -- role (key of Professor)
   DeptId CHAR (4),         -- role (key of Department)
   PRIMARY KEY (ProfId),  -- since a professor works in at most one department
   FOREIGN KEY (ProfId) REFERENCES Professor (Id),
   FOREIGN KEY (DeptId) REFERENCES Department )
```

17

Example 2




```
CREATE TABLE Sold (
   Price INTEGER,              -- attribute
   Date DATE,                  -- attribute
   ProjId INTEGER,             -- role
   SupplierId INTEGER,         -- role
   PartNumber INTEGER,         -- role
   PRIMARY KEY (ProjId, SupplierId, PartNumber, Date),
   FOREIGN KEY (ProjId) REFERENCES Project,
   FOREIGN KEY (SupplierId) REFERENCES Supplier (Id),
   FOREIGN KEY (PartNumber) REFERENCES Part (Number) )
```

18

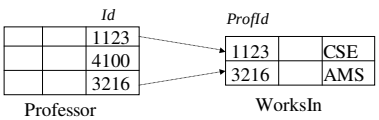
Key Constraint (special case)

- If, for a particular participant entity type, each entity participates in *at most* one relationship, corresponding role is a key of relationship type
  - E.g., *Professor* role is unique in WorksIn
- Representation in E-R diagram: arrow

19

Key Constraint (special case)

- *Relational model representation*: key of relation corresponding to entity type is key of relation corresponding to relationship type
  - *Id* is primary key of Professor; *ProfId* is key of WorksIn. Professor 4100 does not participate.
  - Cannot use foreign key in Professor since some professors do not participate



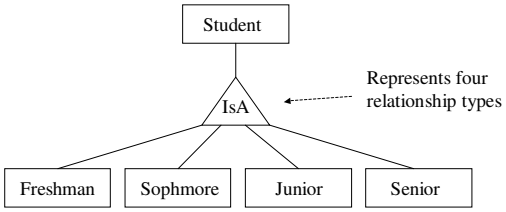
20

Entity Type Hierarchies

- One entity type might be subtype of another
  - Freshman is a subtype of Student
- A relationship exists between a Freshman entity and the corresponding Student entity
  - e.g., Freshman John is related to Student John
- This relationship is called *IsA*
  - Freshman IsA Student
  - The two entities related by IsA are always descriptions of the same real-world object

21

IsA



22

Properties of IsA

- *Inheritance* - Attributes of supertype apply to subtype.
  - E.g., *GPA* attribute of Student applies to Freshman
  - Subtype *inherits* all attributes of supertype.
  - Key of supertype is key of subtype
- *Transitivity* - Hierarchy of IsA
  - Student is subtype of Person, Freshman is subtype of Student, so Freshman is also a subtype of Student

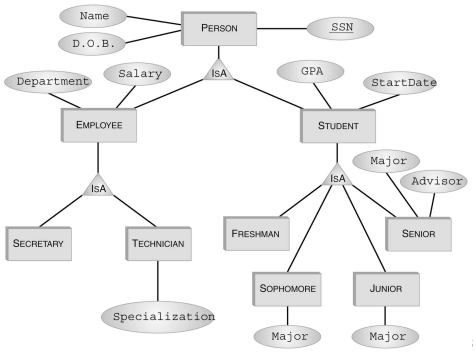
23

IsA

- *Advantage*: Used to create a more concise and readable E-R diagram
  - Attributes common to different entity sets need not be repeated
  - They can be grouped in one place as attributes of supertype
  - Attributes of (sibling) subtypes can be different

24

IsA Hierarchy - Example



Type Hierarchy

- Might have associated constraints:
  - *Covering constraint*: Union of subtype entities is equal to set of supertype entities
    - Employee is either a secretary or a technician (or both)
  - *Disjointness constraint*: Sets of subtype entities are disjoint from one another
    - Freshman, Sophomore, Junior, Senior are disjoint sets
- Might be related to fragmentation of data

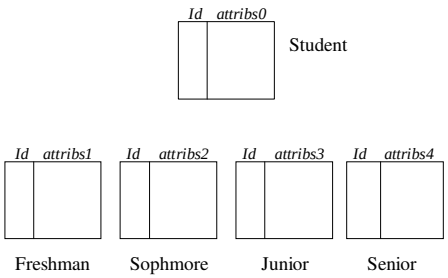
26

Type Hierarchies and Relational Model

- Supertypes and subtypes can be realized as separate relations
  - Need a way of identifying subtype entity with its (unique) related supertype entity
    - *Choose a candidate key and make it an attribute of all entity types in hierarchy*

27

Type Hierarchies and Relational Model



28

Type Hierarchies and Relational Model

- Redundancy eliminated if IsA is not disjoint
  - For individuals who are both employees and students, Name and DOB are stored once

Person

SSN	*Name*	*DOB*
1234	Mary	1950

Employee

SSN	*Department*	*Salary*
1234	Accounting	35000

Student

SSN	*GPA*	*StartDate*
1234	3.5	1997

29