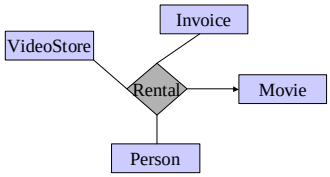
Arrows in Multiway Relationships

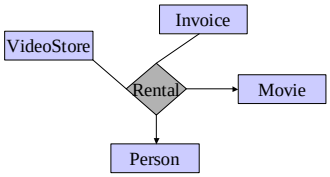
- Q: what does the arrow mean ?



- A: if I know the store, person, invoice, I know the movie too

Arrows in Multiway Relationships

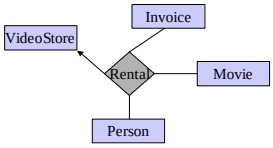
- Q: what do these arrow mean ?



- A: store, person, invoice determines movie and store, invoice, movie determines person

Arrows in Multiway Relationships

- Q: how do I say: "invoice determines store" ?
- A: no good way; best approximation:



- Why is this incomplete ?

Lecture 3: Database Modeling (continued)

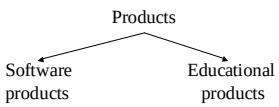
Sept. 29, 2000

Outline

- Subclasses (2.4)
- Constraints: (2.5)
- Weak entity sets (2.6)

Modeling Subclasses

Some objects in a class may be special
- •define a new class
- •better: define a *subclass*



So --- we define subclasses (in ODL and in E/R).

Subclasses in ODL

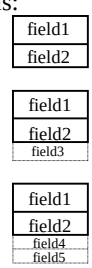
```
interface  SoftwareProduct:  Product{
      attribute  string    platform;
      attribute  integer  requiredMemory;
}

interface EducationalProduct: Product{
      attribute struct Interval {integer begin, integer end} ageGroup;
      attribute string topic;
}
```

The two classes *inherit* all the properties of Product.

Understanding Subclasses

- Think in terms of records:
  - Product
  - SoftwareProduct
  - EducationalProduct



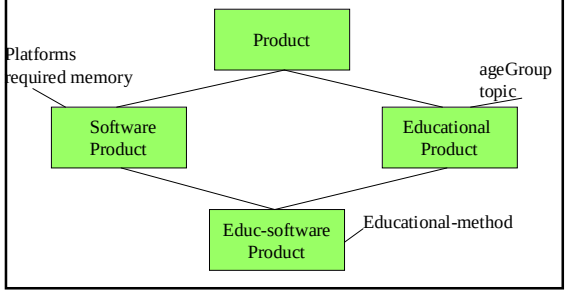
Subclasses for Heterogeneous Collections

Now consider a class referring to Product:

```
interface  Vendor {
      attribute  string    name;
      relationshiop Set<Product> offers;
}
```

Some offers are Product, others are SoftwareProduct, or EducationalProduct

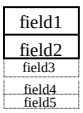
Multiple Inheritance in ODL



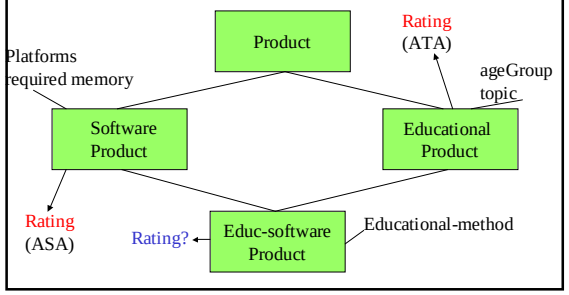

```
interface  EducSoftwareProduct:
      SoftwareProduct, EducationalProduct {
        attribute  string  educational-method;
}
```

Understanding Multiple Inheritance

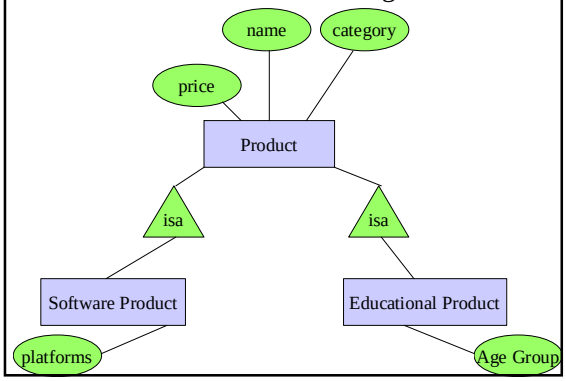
- Think in terms of records:
  - EducSoftwareProduct



How do we resolve conflicts?

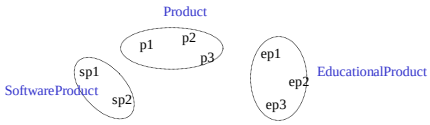


Subclasses in E/R Diagrams



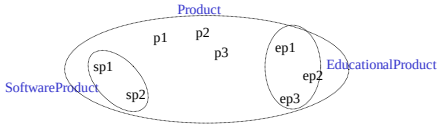
Difference between ODL and E/R inheritance

- ODL: classes are disjoint

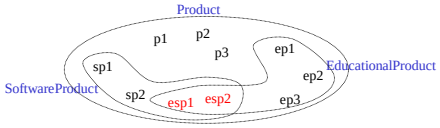


Difference between ODL and E/R inheritance

- E/R: entity sets overlap

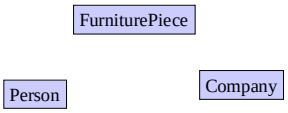


- No need for multiple inheritance in E/R



- we have three entity sets, but four different kinds of objects

Modeling UnionTypes With Subclasses

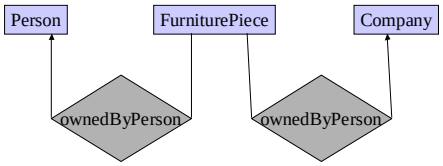


Say: each piece of furniture is owned either by a person, or by a company

Modeling Union Types with Subclasses

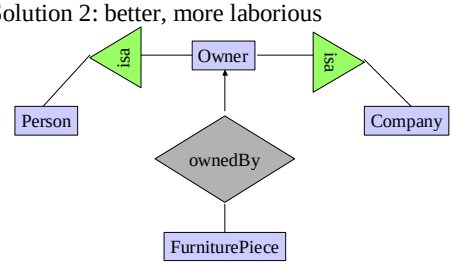
Say: each piece of furniture is owned either by a person, or by a company

Solution 1. Acceptable, imperfect (What's wrong ?)



Modeling Union Types with Subclasses

Solution 2: better, more laborious



Constraints

- A constraint = an assertion about the database that must be true at all times
- part of the db schema
- types in programming languages do not have anything similar
- correspond to *invariants* in programming languages

Modeling Constraints

Finding constraints is part of the modeling process.
Commonly used constraints:

Keys: social security number uniquely identifies a person.

Single-value constraints: a person can have only one father.

Referential integrity constraints: if you work for a company, it must exist in the database.

Domain constraints: peoples' ages are between 0 and 150.
General constraints: all others (at most 50 students enroll in a class)

Keys

A set of attributes that uniquely identify an object or entity:

Person: social security number
name
name + address
name + address + age

Perfect keys are often hard to find, so organizations usually invent something.

An object may have multiple keys:

employee number, social-security number

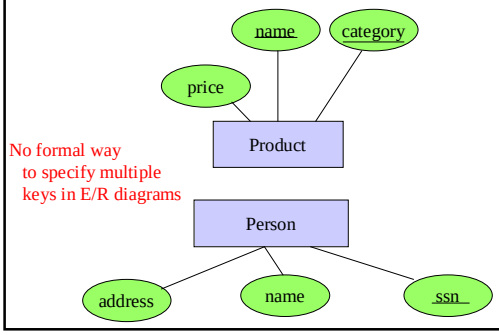
Keys in ODL

```
Interface Person
      (key  ssn)
 {
     attribute string ssn;
     attribute string name;
     ...
  }
```

Defining multiple keys:

```
      (key  ssn employeID  (name address age))
```

Keys in E/R Diagrams



Single Value Constraints

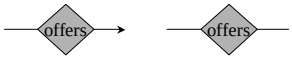
- Sometimes we can choose to allow one or more values
- ODL:
  - attributes are always single value
  - relationships have single or multiple values

    relationship person president;

    relationship set<person> presidents;

Single Value Constraints

- E/R:



Single Value Constraints

Single Value Constraint:

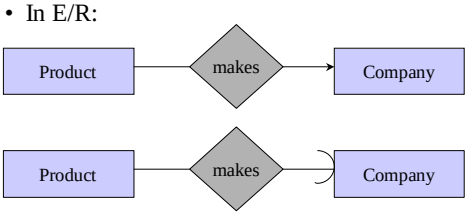
- we explicitly require one value
- two flavors:
  - allow nulls
  - do not allow nulls

Referential Integrity Constraints

- In some formalisms we may refer to other object but get garbage instead
  - e.g. a dangling pointer in C/C++
- the Referential Integrity Constraint explicitly requires a reference to exists

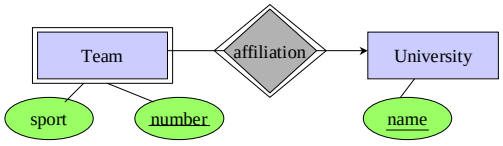
Referential Integrity Constraints

- In ODL:
  - means that a relationship cannot be NULL
- In E/R:



Weak Entity Sets

Entity sets are weak when their key attributes come from other classes to which they are related.

This happens if:

- part-of hierarchies
- splitting n-ary relations to binary.