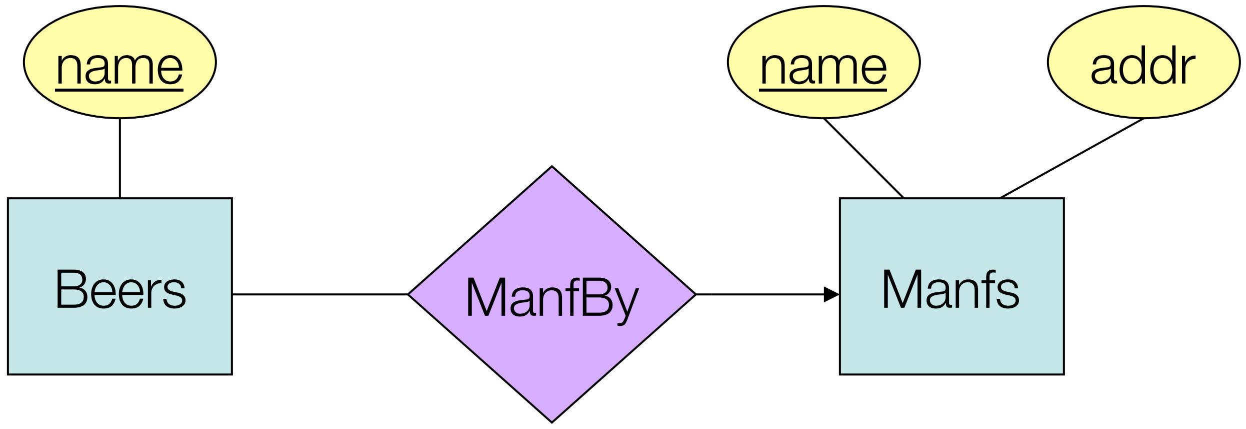
Design Techniques for E/R

- Pretty simple, really
    1. Avoid redundancy (or introducing redundancy)
    2. Do not use an entity set when an attribute will do!
    3. Limit the use of weak entity sets

1. Avoid redundancy

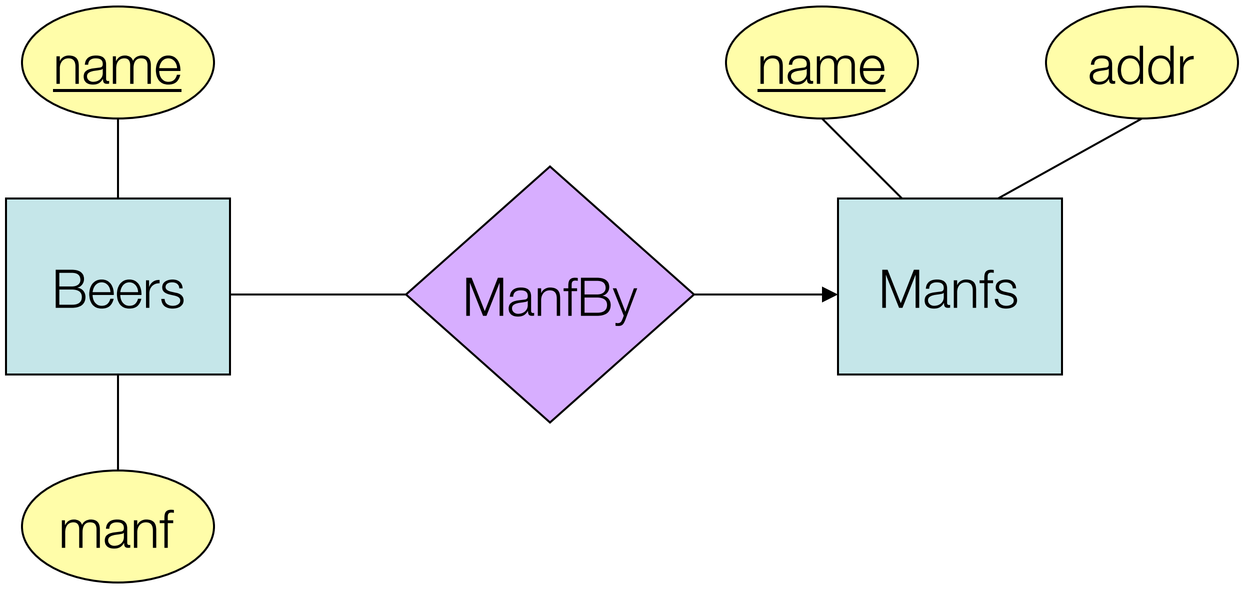
- Our working definition of redundancy
  - Saying the same thing in two or more different places
  - (Bad software engineering technique, let alone database design technique.)
- This wastes space...
- ... but also (which is perhaps worse!) encourages inconsistency
  - Two representations of the same fact become inconsistent if we change it in one spot but forget to the change it in the other spot.
  - (Also recall update anomalies discussed during presentation of Functional Dependencies.)

Example: Good



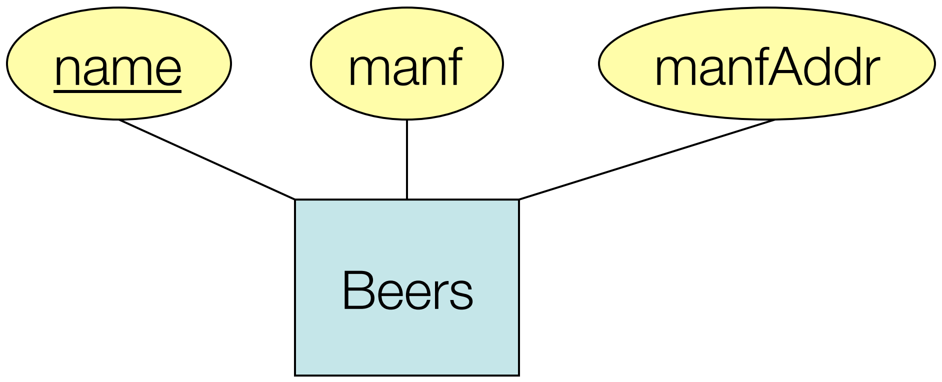
This design gives the **address of each manufacturer** exactly once.

Example: Bad



This design states the manufacturer of a beer **twice**: **as an attribute** and **as a related entity**.

Example: Worsetest

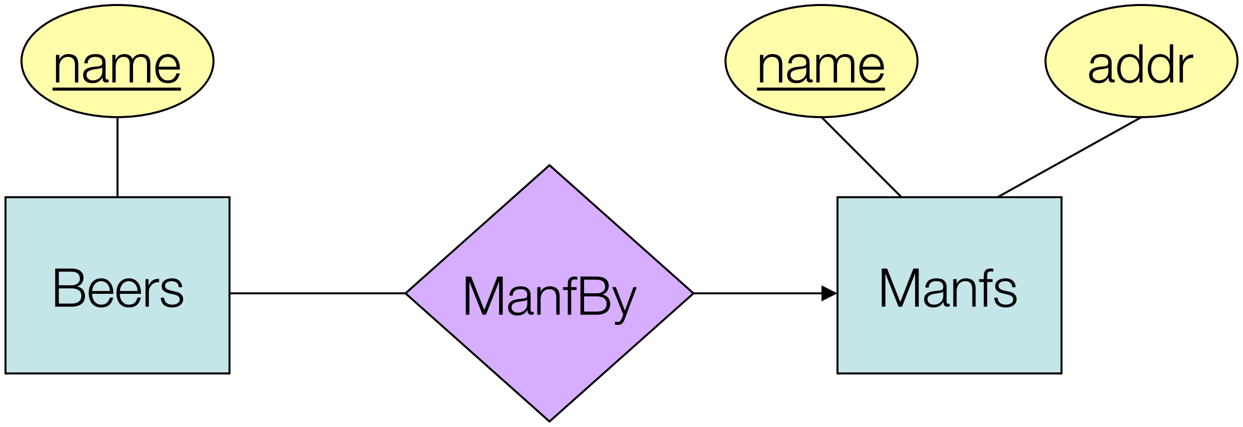


This design **repeats the manufacturer's address once for each beer** and **loses the address if there are temporarily no beers** for a manufacturer.

2. Entity sets vs. attributes

- An entity set should satisfy at least one of the following:
- It is more than the name of something
  - That is, **it has at least one non-key attribute**!
- It is the **many** in a **many-to-one** or **many-to-many** relationship.

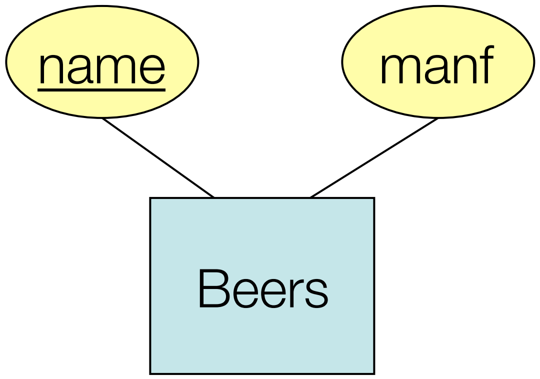
Example: Good



Manfs deserves to be an entity set because of the non-key attribute **addr**.

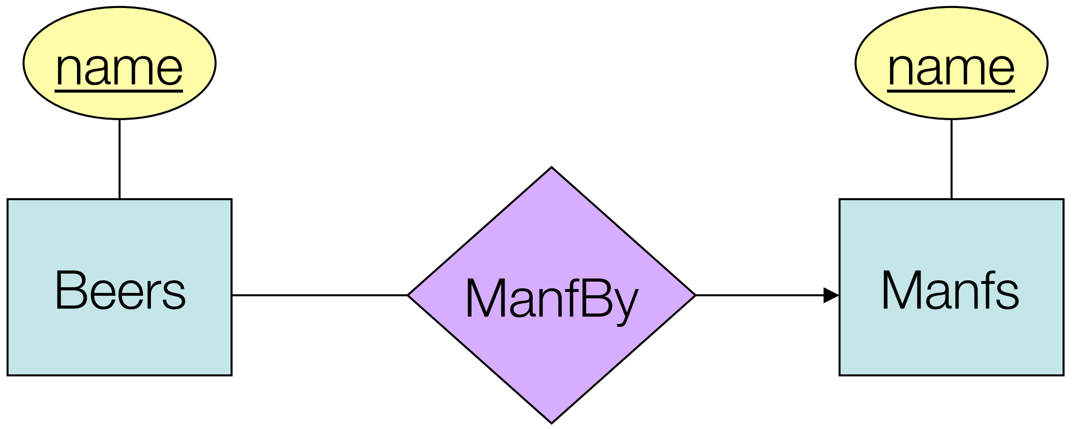
Beers deserves to be an entity set because it is the “many” of the many-one relationship **ManfBy**.

Example: Good



There is no need to make the manufacturer a separate entity set here, because we record nothing about manufacturers besides their name. Therefore it makes sense **manf** becomes an attribute in the entity **Beers**.

Example: Ugh.



Since the manufacturer is nothing but a name, and is not at the “many” end of any relationship, it should not be an entity set. (Repeating what we mentioned on the previous slide...)

3. Do not overuse weak entity sets

- As we start to design databases for the first time, we may often doubt that anything could, by itself, be a key.
  - As a result, we mistakenly make all entity sets weak.
  - We then support that weak entity set by all other entity sets to which they are linked.
- In practice we usually create unique IDs for entities with entity sets
  - Example: social insurance numbers, student numbers
  - Example: automobile VINs, cell phone IMEI numbers

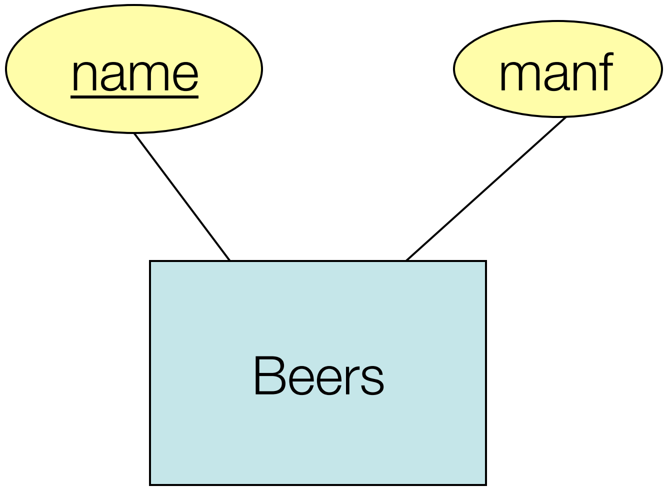
So when **do** we need weak entity sets?

- Usual reason:
  - There is no global authority capable of creating unique IDs for entities in the set
- Example:
  - Unlikely we could convince the International Rugby Union that they assign unique player numbers across all rugby teams in the world.

Translating E/R diagrams into relations

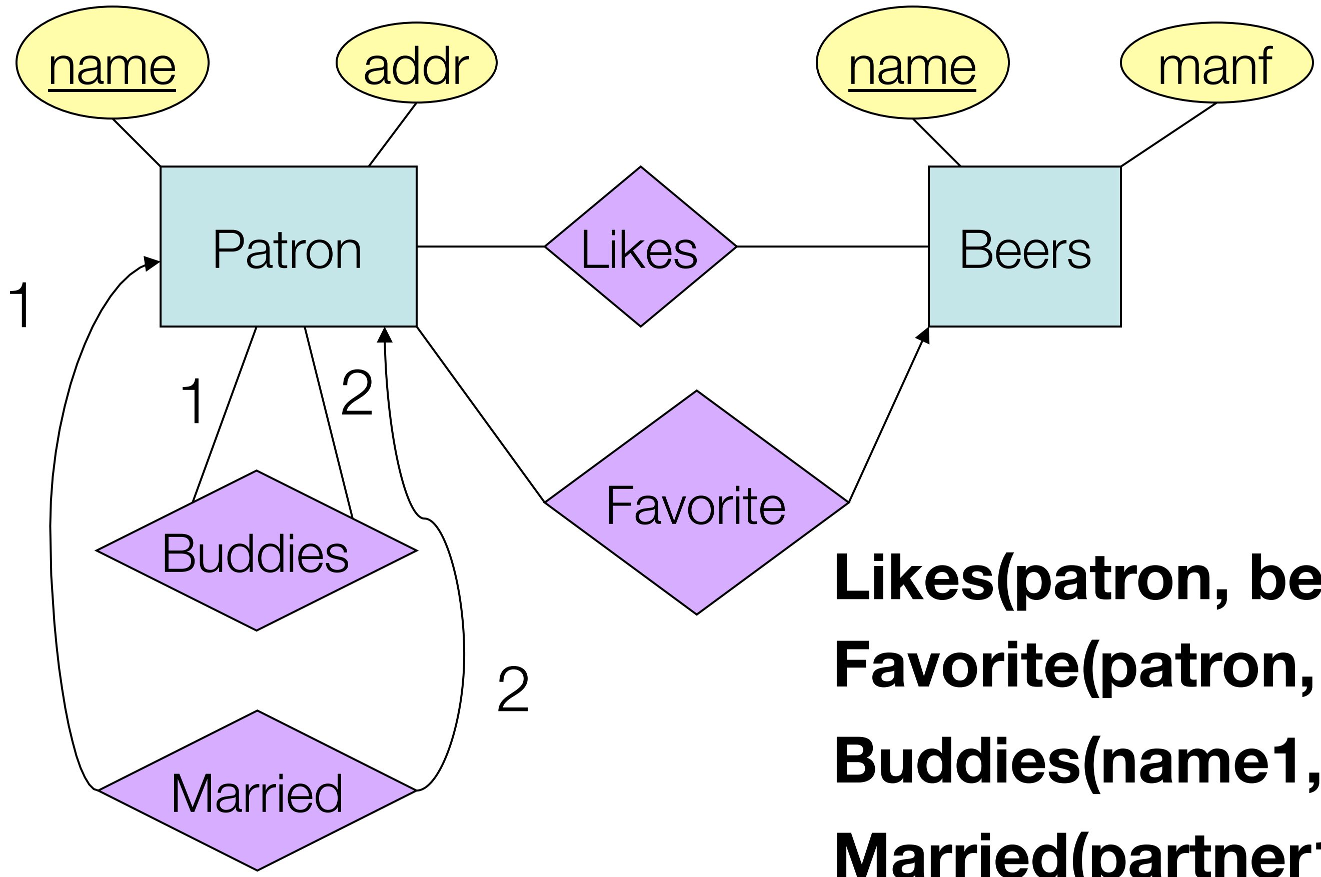
- For the most part the process is straightforward
- Entity sets → relation
  - Entity-set attributes → relation attributes
- Relationships → relations where attributes are:
  - the keys of the connected entity sets, or
  - attributes of the relationship itself.
- Translation to relations only gets particularly tricky when we deal with ISA hierarchies.

Entity Set → Relation



Relation: **Beers(name, manf)**

Relationship → Relation



Likes(patron, beer)

Favorite(patron, beer)

Buddies(name1, name2)

Married(partner1, partner2)

Combining relations

- On occasion it is OK to combine into one relation:
  - The relation for an entity-set E
  - The relations for many-to-one relationships of which E is on the "many" end.
- Example:
  - Patrons(name, addr) & Favorite(patron, beer)...
  - ... can be combined into Patrons1(name, addr, favBeer)
  - Here E = Patrons; relation on the "one" end of "many-to-one" is Beers.

Combining relations

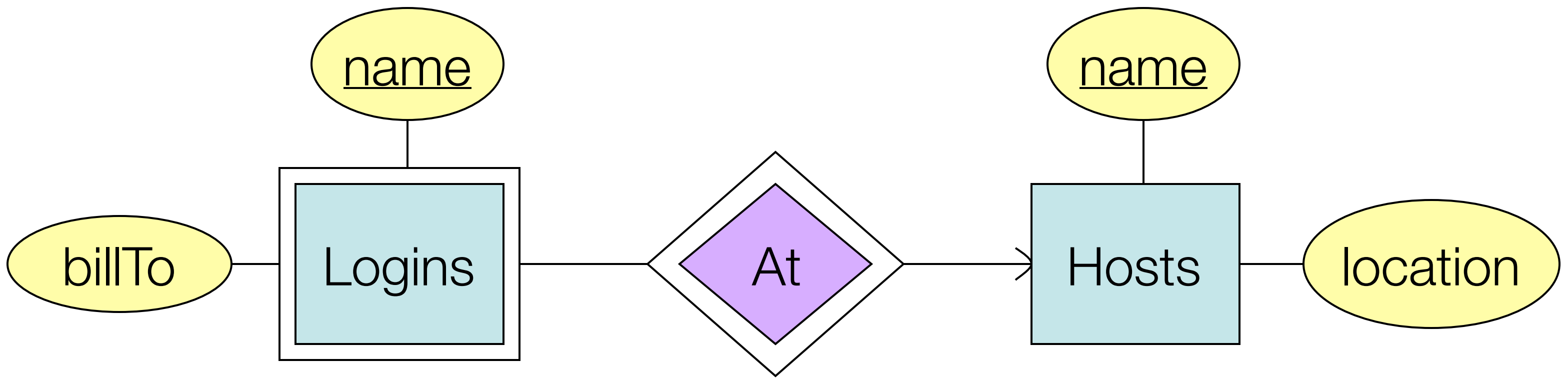
- However, the approach does not work when E is in a many-to-many relationship
- Example:
  - Combining Patrons with Likes would be a mistake.
  - This would lead to redundancy
  - (Recall that a Patron can have multiple beers that they "like".)

name	addr	beer
Cliff	Tyndall	Blue
Cliff	Tyndall	Bud Light
...	...	...

Handling weak entity sets

- The relation for a weak entity set must:
  - include attributes for its complete key (including those belong to other entity sets)
  - include its own (by definition) non-key attributes
- The supporting relationship is redundant, however, and yields no relation.
  - Only exception: if the relationship itself has attributes).

Example: Weak Entity Set -> Relation



Hosts(hostName, location)

Logins(loginName, hostName, billTo)

~~**At(loginName, hostName, hostName2)**~~

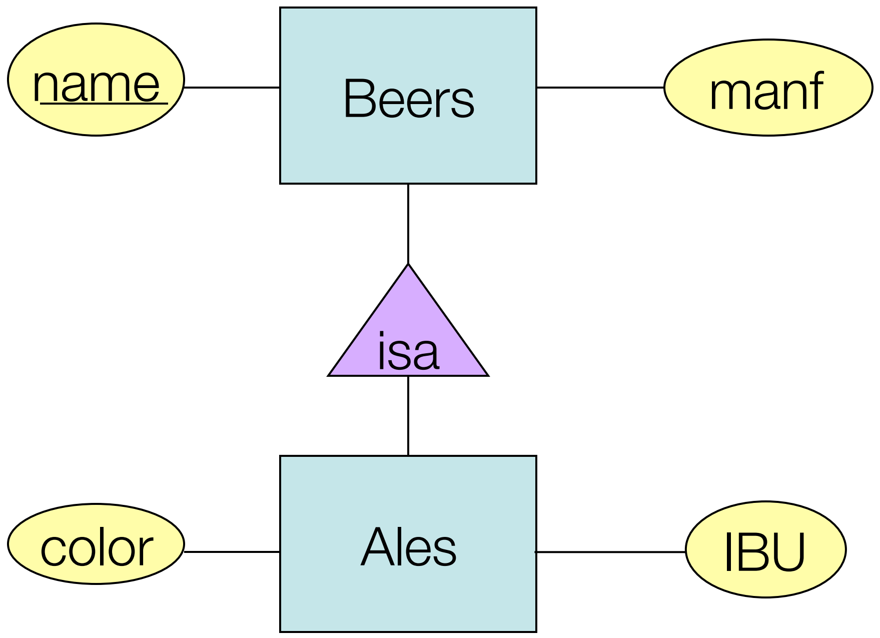
At becomes part of **Logins**

Must be the same

Converting ISAs into relations: three ways

- Object-oriented:
  - One relation per subset of subclasses
  - Includes all relevant attributes
- Use nulls:
  - One relation
  - Relation tuples for an entity how have NULL in those attributes that do not belong to the ISA entity.
- E/R style:
  - One relation for each subclass (key attributes; attributes for that subclass)

Example: ISA $\rightarrow$ Relations



Object-Oriented

Beers

name	manf
Bud	Anheuser-Busch

Ales

name	manf	color	IBU
Longboat	Philips	dark	40

Good for queries like “find the color of ales made by Phillips”

E/R Style

Beers

name	manf
Longboat	Philips
Bud	Anheuser-Busch

Ales

name	color	IBU
Longboat	dark	40

Good for queries like “find all beers (including ales) made by Phillips”

Using Nulls

Beers

name	manf	color	IBU
Longboat	Philips	dark	40
Bud	Anheuser-Busch	NULL	NULL