

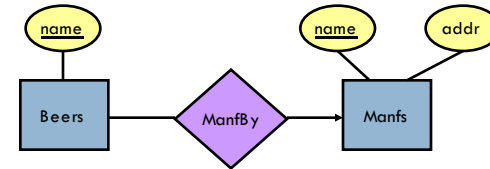
Avoiding Redundancy

1

- **Redundancy** = saying the same thing in two (or more) different ways.
- Wastes space and (more importantly) encourages inconsistency.
 - ▣ Two representations of the same fact become inconsistent if we change one and forget to change the other.

Example: Good

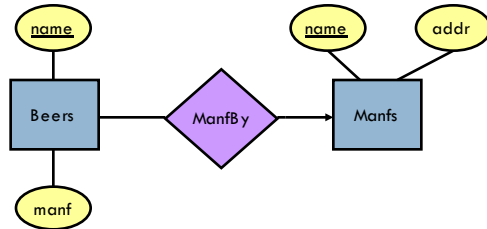
2



This design gives the address of each manufacturer exactly once.

Example: Bad

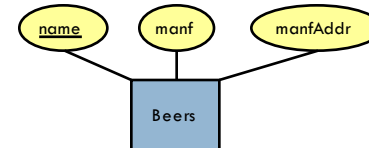
3



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

Example: Bad

4



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

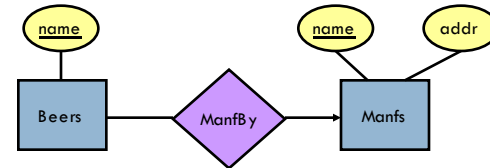
Entity Sets Versus Attributes

5

- An entity set should satisfy at least one of the following conditions:
 - ▣ It is more than the name of something; it has at least one non-key attribute. OR
 - ▣ It is the “many” in a many-one or many-many relationship.
- Depends on the application requirements:
 - If we have several addresses per employee, *address* must be an entity (since attributes cannot be set-valued).
 - If the structure (city, street, etc.) is important, e.g., we want to retrieve employees in a given city, *address* must be modeled as an entity (since attribute values are atomic).

Example: Good

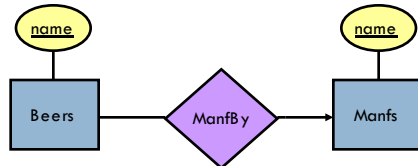
6



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

Example: Bad

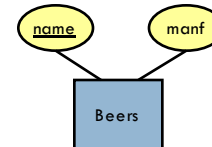
7



Since the manufacturer is nothing but a name, and is not at the “many” end of any relationship, it need not be an entity set.

Example: Good

8



There is no need to make the manufacturer an entity set, because we record nothing about manufacturers besides their name.

Don't Overuse Weak Entity Sets

9

- Beginning database designers often doubt that anything could be a key by itself.
 - ▣ They make all entity sets weak, supported by all other entity sets to which they are linked.
- In reality, we usually create unique ID's for entity sets.
 - ▣ Examples include social-security numbers, automobile VIN's etc.

When Do We Need Weak Entity Sets?

10

- The usual reason is that there is no global authority capable of creating unique ID's.
- **Example:** it is unlikely that there could be an agreement to assign unique player numbers across all football teams in the world.

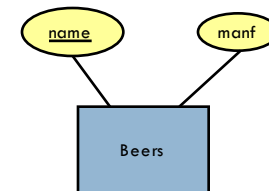
From E/R Diagrams to Relations

11

- Entity set -> relation.
 - ▣ Attributes -> attributes.
- Relationships -> relations whose attributes are only:
 - ▣ The keys of the connected entity sets.
 - ▣ Attributes of the relationship itself.

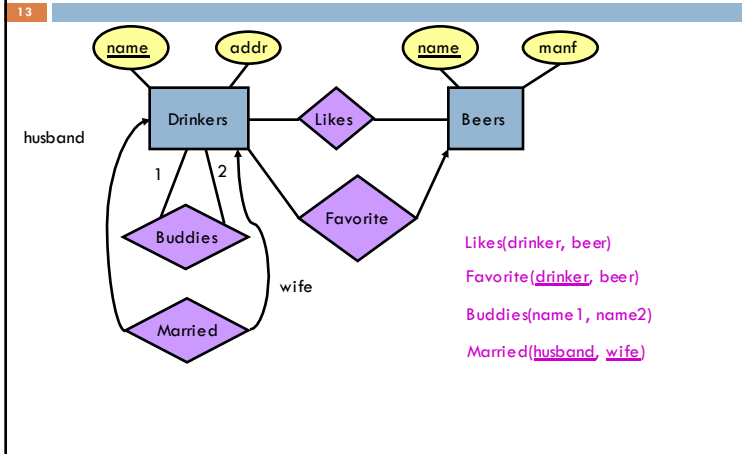
Entity Set -> Relation

12



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

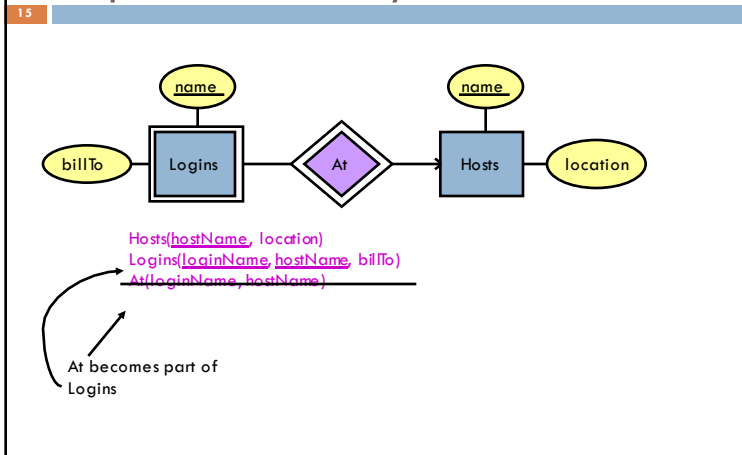
Relationship -> Relation



Handling Weak Entity Sets

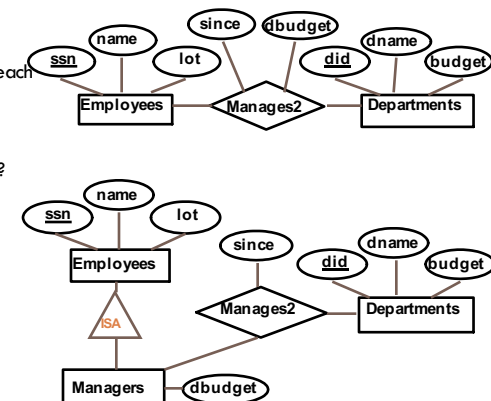
- 14
- Relation for a weak entity set must include attributes for its complete key (including those belonging to other entity sets), as well as its own, nonkey attributes.
 - A supporting relationship is redundant and yields no relation (unless it has attributes).

Example: Weak Entity Set -> Relation



Entity vs. Relationship

- 16
- First ER diagram OK if a manager gets a separate discretionary budget for each dept.
 - What if a manager gets a discretionary budget that covers *all* managed depts?
 - **Redundancy:** dbudget stored for each dept managed by manager.
 - **Misleading:** Suggests dbudget associated with department-mgr combination.



Summary

17

- *Conceptual design* follows *requirements analysis*,
 - ▣ Yields a high-level description of data to be stored
- ER model popular for conceptual design
 - ▣ Constructs are expressive, close to the way people think about their applications.
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships).
- Some additional constructs: *weak entities*, *ISA hierarchies*.

R. Ramakrishnan & J. Gehrke

Summary of ER (cont'd.)

18

- Several kinds of integrity constraints can be expressed in the ER model: *key constraints*, *participation constraints*, and *overlap/covering constraints* for ISA hierarchies.
 - ▣ Constraints play an important role in determining the best database design for an enterprise.

Summary (cont'd)

19

- ER design is *subjective*. There are often many ways to model a given scenario!
- Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:
 - ▣ Entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies
 - ▣ Ensuring good database design: resulting relational schema should be analyzed and refined further.