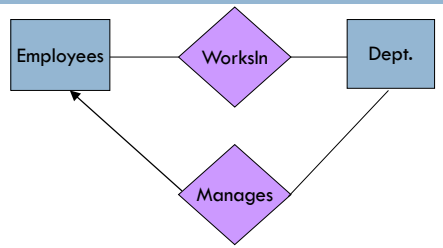
Key Constraints

37



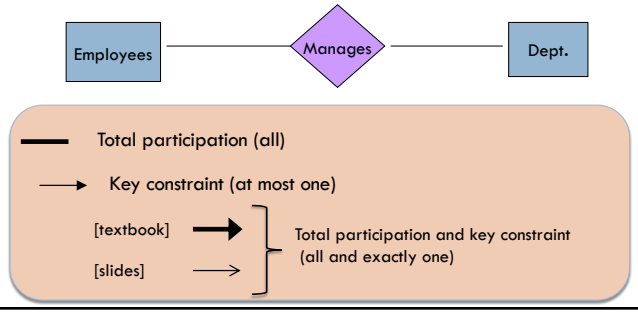
- Many-many: "An employee can work in many depts, and a dept. can have many employees
- One-many: A dept has at most one manager, and employees can manage many departments

37

Participation Constraints

38

- Does every dept. have to have a manager?
  - If yes, then every dept. must appear in the manages relation: total participation (vs. partial)



38

Attributes on Relationships

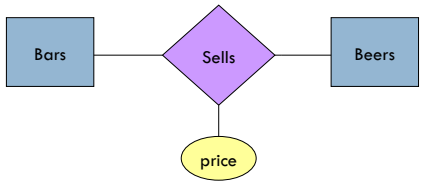
39

- Sometimes it is useful to attach an attribute to a relationship.
- Think of this attribute as a property of tuples in the relationship set.

39

Example: Attribute on Relationship

40



Price is a function of both the bar and the beer, not of one alone.
E.g., "The price of Miller beer at Joe's bar"

40

Equivalent Diagrams Without Attributes on Relationships

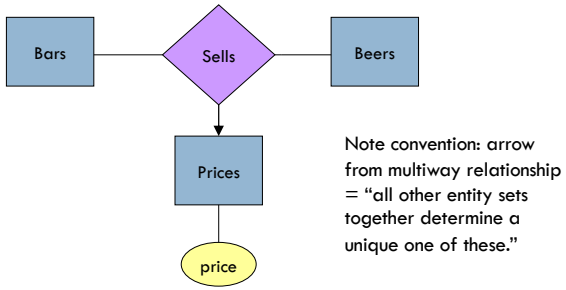
41

- Create an entity set representing values of the attribute.
- Make that entity set participate in the relationship.

41

Example: Removing an Attribute from a Relationship

42



42

Roles

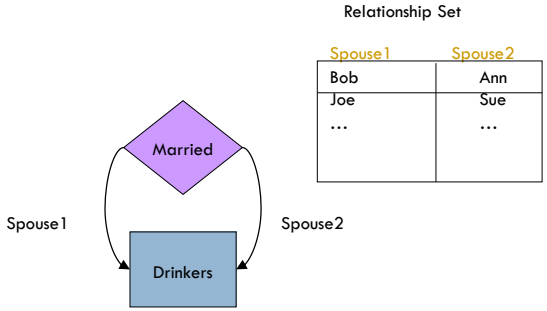
43

- Sometimes an entity set appears more than once in a relationship.
- Label the edges between the relationship and the entity set with names called *roles*.

43

Example: Roles

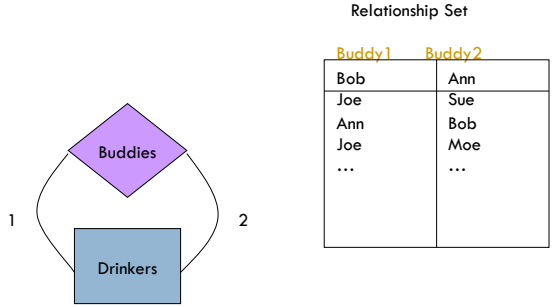
44



44

Example: Roles

45



Relationship Set

Buddy1	Buddy2
Bob	Ann
Joe	Sue
Ann	Bob
Joe	Moe
…	…

45

Subclasses

46

- *Subclass* = special case = more properties.
- Example: Ales are a kind of beer.
  - Not every beer is an ale, but some are.
  - Let us suppose that in addition to all the *properties* (attributes and relationships) of beers, ales also have the attribute color.

46

Subclasses in E/R Diagrams

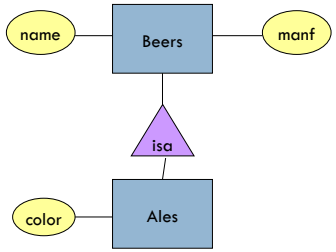
47

- isa triangles indicate the subclass relationship.
  - Point to the superclass.
- Reasons for using isa:
  - To add descriptive attributes specific to a subclass.
  - To identify entities that participate in a relationship.

47

Example: Subclasses

48



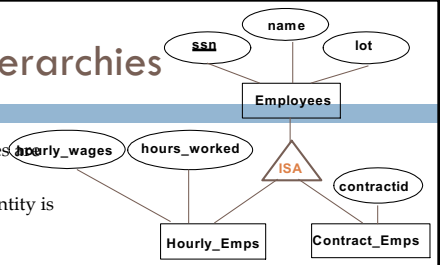
Assume subclasses form a tree.

48

ISA (`is a') Hierarchies

49

- As in C++, or other PLs, attributes are inherited.
- If we declare A **ISA** B, every A entity is also considered to be a B entity.



- *Overlap constraints*: Can two sub-classes contain the same entity? E.g., Can Joe be an Hourly_Emps as well as a Contract_Emps entity?
- *Covering constraints*: Does every Employees entity have to be an Hourly_Emps or a Contract_Emps entity?

R. Ramakrishnan & J. Gehrke

49

Keys

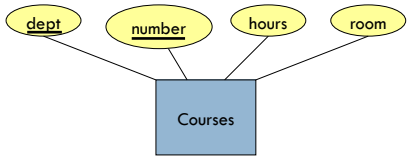
50

- A *key* is a set of attributes for one entity set such that no two entities in this set agree on all the attributes of the key.
  - It is allowed for two entities to agree on some, but not all, of the key attributes.
- We must designate a key for every entity set.
- Underline the key attribute(s).

50

Example: a Multi-attribute Key

51



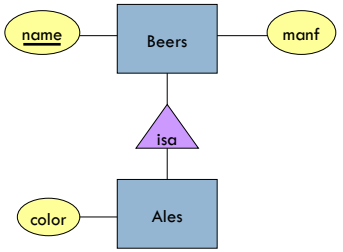
- Note that hours and room could also serve as a key, but we must select only one primary key.

51

Keys

52

In an Isa hierarchy, only the root entity set has a key, and it must serve as the key for all entities in the hierarchy.



52

Weak Entity Sets

53

- Occasionally, entities of an entity set need "help" to identify them uniquely.
- Entity set *E* is said to be *weak* if in order to identify entities of *E* uniquely, we need to follow one or more many-one relationships from *E* and include the key of the related entities from the connected entity sets.

53

Example: Weak Entity Set

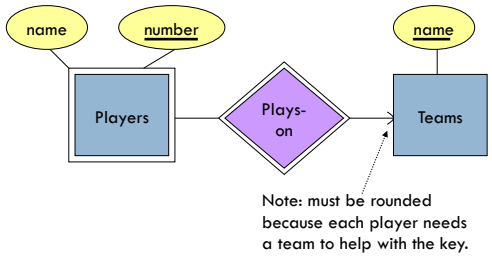
54

- name is almost a key for football players, but there might be two with the same name.
- number is certainly not a key, since players on two teams could have the same number.
- But number, together with the team name related to the player by Plays-on should be unique.

54

In E/R Diagrams

55



- Double diamond for *supporting* many-one relationship.
- Double rectangle for the weak entity set.

55

Weak Entity-Set Rules

56

- A weak entity set has one or more many-one relationships to other (supporting) entity sets.
  - Not every many-one relationship from a weak entity set need be supporting.
  - But supporting relationships must have a rounded arrow (entity at the "one" end is guaranteed).

56

Weak Entity-Set Rules – (2)

57

- The key for a weak entity set is its own underlined attributes and the keys from the supporting entity sets.
  - E.g., (player) number and (team) name is a key for Players in the previous example.

57

Design Techniques

58

1. Avoid redundancy.
2. Limit the use of weak entity sets.
3. Don't use an entity set when an attribute will do.

58

Avoiding Redundancy

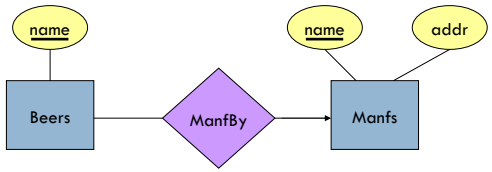
59

- *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.

59

Example: Good

60

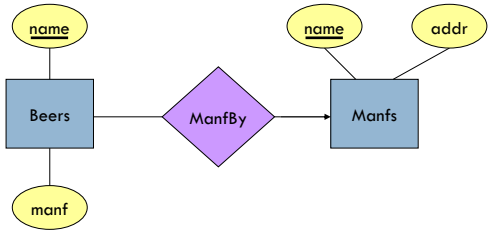


This design gives the address of each manufacturer exactly once.

60

Example: Bad

61

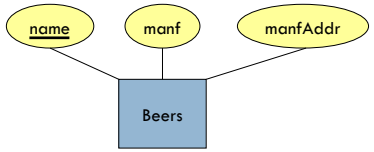


This design states the manufacturer of a beer twice: as an attribute and as a related entity.

61

Example: Bad

62



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

62