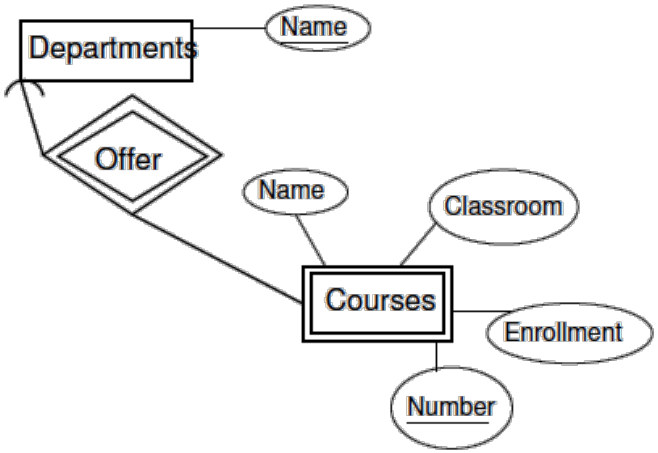
Handling weak entity sets

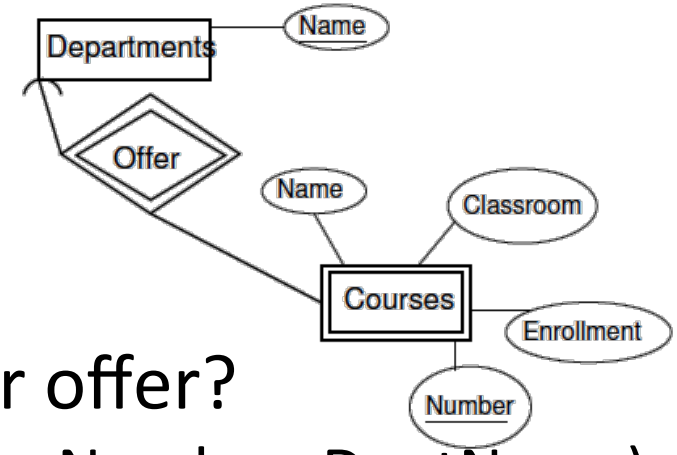
- For each weak entity set W, create a relation with attributes:
  - attributes of W
  - key attributes of supporting entity sets for W

Supporting Relationships



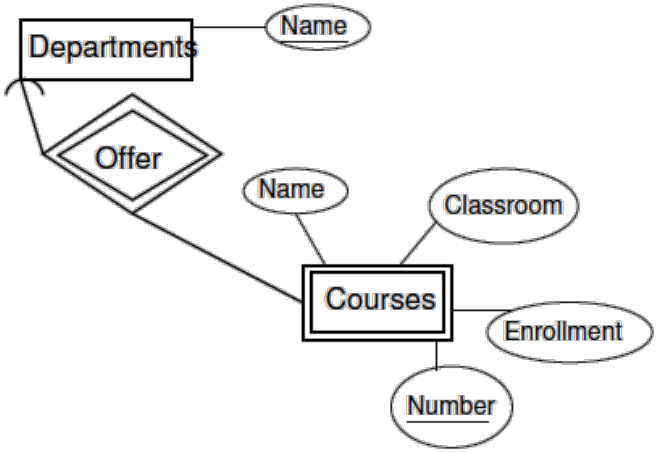
- Schema for Departments is Departments(Name)
- Schema for Courses is Courses(Number, DeptName, CourseName, Classroom, Enrollment)
- What is the schema for Offer?

Supporting Relationships



- What is the schema for offer?
  - Offer(DeptName, CourseNumber, DeptName)
  - But Name and DeptName are identical, so the schema for Offer is Offer(Number, DeptName)
  - The schema for Offer is a subset of the schema for the weak entity set, so ***we can dispense with the relation for Offer.***
  - ***Key point: Don't make a relation for supporting relationships.***

Summary of Weak Entity Sets



- If W is a weak entity set, the relation for W has a schema whose attributes are
  - all attributes of W
  - all attributes of supporting relationships for W
  - for each supporting relationship for W to an entity set E
  - the key attributes of E
- There is no relation for any supporting relationship for W

Combining Relations

- Consider many-one Teach relationship from Courses to Professors
- Schemas are:

  Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)

  Professors(Name, Office, Age)

  Teach(Number, DepartmentName, ProfessorName, Office)

Combining Relations

Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)

Professors(Name, Office, Age)

Teach(Number, DepartmentName, ProfessorName, Office)

- The key for Courses uniquely determines all attributes of Teach
- We can combine the relations for Courses and Teach into a single relation whose attributes are
  - All the attributes for Courses,
  - Any attributes of Teach, and
  - The key attributes of Professors

Rules for Combining Relations

- We can combine into one relation Q
  - The relation for an entity set E
  - all many-to-one relationships R1, R2, …, Rk from E to other entity sets E1, E2, …, Ek respectively
- The attributes of Q are
  - All the attributes of E
  - Any attributes of R1, R2, …, Rk
  - The key attributes of E1, E2, …, Ek
- Combining a many-many relationship with one of its entity sets often leads to redundancy.

ISA to Relational

- Three approaches:
  - E/R viewpoint
  - Object-oriented viewpoint
  - "Flatten" viewpoint

Rules Satisfied by an ISA Hierarchy

- The hierarchy has a root entity set.
- The root entity set has a key that identifies every entity represented by the hierarchy.
- A particular entity can have components that belong to entity sets of any subtree of the hierarchy, as long as that subtree includes the root.

Example ISA hierarchy

ISA to Relational Method I: E/R Approach

- Create a relation for each entity set
- The attributes of the relation for a non-root entity set E are
  - the attributes forming the key (obtained from the root) and
  - any attributes of E itself
- An entity with components in multiple entity sets has tuples in all the relations corresponding to these entity sets
- Do not create a relation for any isa relationship
- Create a relation for every other relationship

ISA to Relational Method III: Object Oriented Approach

- Treat entities as objects belonging to a single class.
- "Class" == subtree of the hierarchy that includes the root.
- e.g., for Movies:
  - Movie, Movie+Cartoon, Movie+MM, Movie +Cartoon+MM.

ISA to Relational Method II: Object Oriented Approach

- Enumerate all subtrees of the hierarchy that contain the root.
- For each such subtree,
  - Create a relation that represents entities that have components in exactly that subtree.
  - The schema for this relation has all the attributes of all the entity sets in that subtree.
- Schema of the relation for a relationship has key attributes of the connected entity sets.

ISA to Relational Method III: “Flatten” Approach (or "NULLs")

- Make one relation for the whole hierarchical structure.
- Use NULL for any attribute that is not defined for a particular entity.

Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations (advantage: flatten)
  - E/R approach works for some queries where info is duplicated among relations.
    - "What 2008 movies were > 150 mins?"
  - E/R approach is hard for other queries.
    - "What weapons were used in cartoons > 150 mins?"

Comparison of the Three Approaches

- Number of relations for n relations in the hierarchy
  - We like to have a small number of relations
  - Flatten
    - 1
  - E/R
    - n
  - OO
    - Can be 2^n

Comparison of the Three Approaches

- Redundancy and space usage
  - Flatten
    - May have a large number of NULLs
    - (also prevents you from using NULL to denote something besides class membership)
  - E/R
    - Several tuples per entity, but only key attributes are repeated
  - OO
    - Only one tuple per entity