VirginiaTech

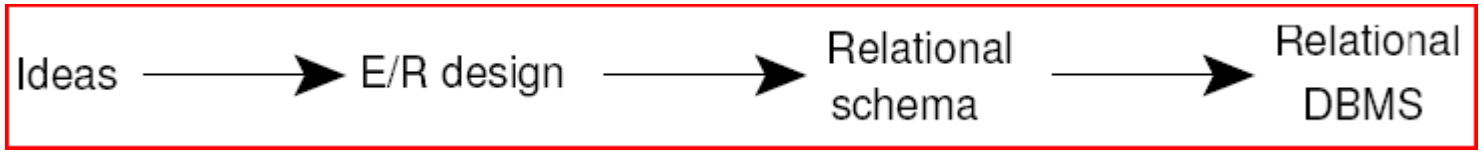
CS 4604: Introduction to Database Management Systems

B. Aditya Prakash

Lecture #11: From E/R to Relations

Purpose of E/R Model

- The E/R model allows us to sketch the design of a database informally.
  - Represent different types of data and how they relate to each other
- Designs are drawings called *entity-relationship diagrams*.
- Fairly mechanical ways to convert E/R diagrams to real implementations like relational databases exist.



Recap: Relational Model

- Built around a single concept for modelling data: the relation or table.
- Supports high-level programming language (SQL).
- Has an elegant mathematical design theory.
- Most current DBMS are relational.

Recap: The Relation

- A relation is a two-dimensional table:
  - Relation ←→ table.
  - Attribute ←→ column name.
  - Tuple ←→ row (not the header row).
  - Database ←→ collection of relations.

CoursesTaken

Student	*Course*	*Grade*
Hermione Grainger	Potions	A-
Draco Malfoy	Potions	B
Harry Potter	Potions	A
Ron Weasley	Potions	C

Recap: The Schema

- The schema of a relation is the name of the relation followed by a paranthetised list of attributes
  - CoursesTaken(Student, Course, Grade)
- A design in a relational model consists of a set of schemas.
  - Such a set of schemas is called a relational database schema

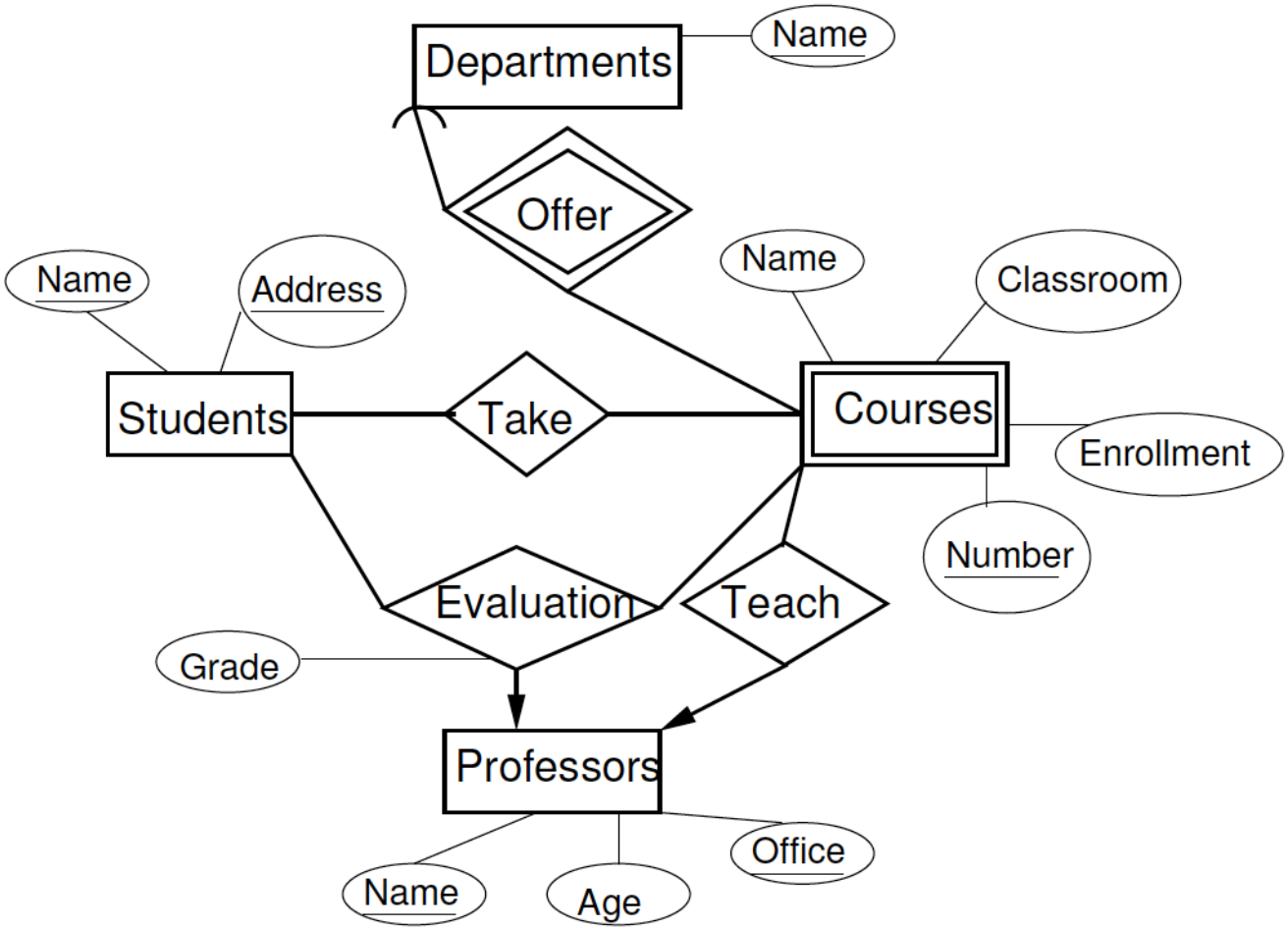
CoursesTaken

Student	*Course*	*Grade*
Hermione Grainger	Potions	A-
Draco Malfoy	Potions	B
Harry Potter	Potions	A
Ron Weasley	Potions	C

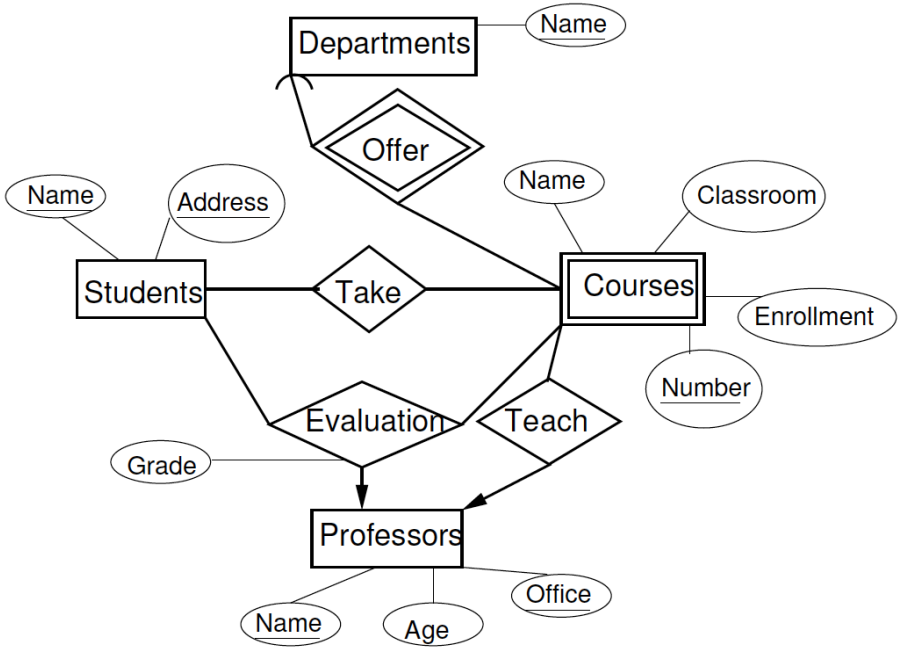
Converting E/R Diagrams to Relational Designs

- Entity Set → Relation
  - Attribute of Entity Set → Attribute of a Relation
- Relationship → relation whose attributes are
  - Attribute of the relationship itself
  - Key attributes of the connected entity sets
- Several special cases:
  - Weak entity sets.
  - Combining relations (especially for many-one relationships)
  - *ISA* relationships and subclasses

Example for Conversion



Schemas for Non-Weak Entity Sets



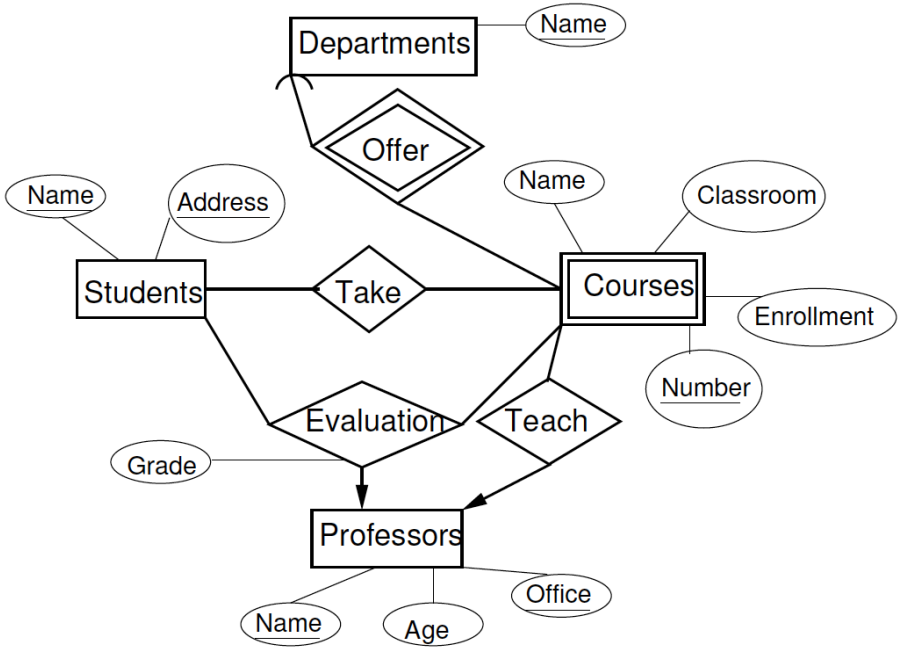
- For each entity set, create a relation with the same name and with the same set of attributes

Students (Name, Address)

Professors (Name, Office, Age)

Departments (Name)

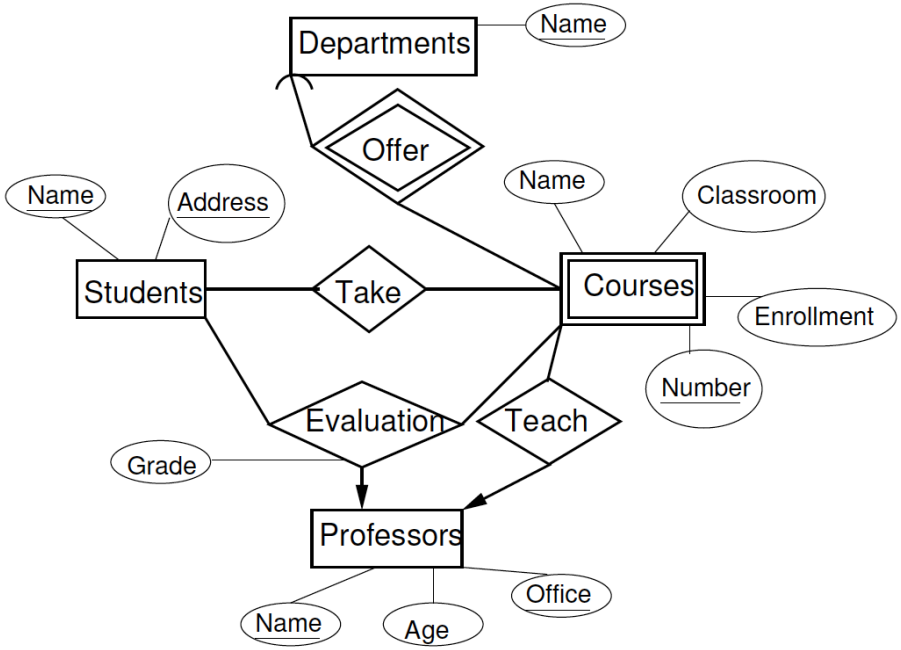
Schemas for Weak Entity Sets



- For each weak entity set W, create a relation with the same name whose attributes are:
  - Attributes of W
  - Key attributes of other entity sets that help form the key for W

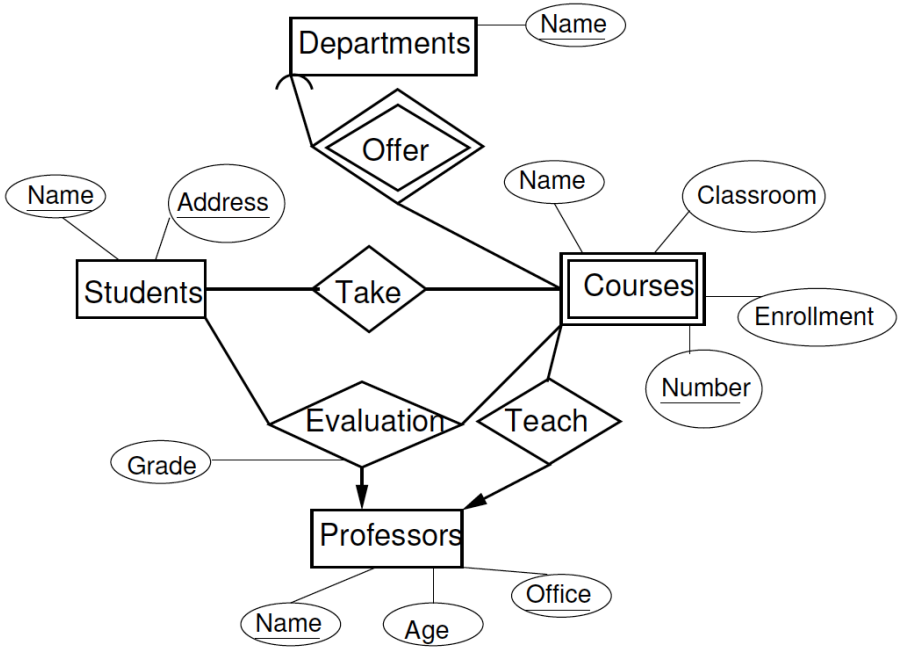
Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)

Schemas for Non-Supporting Relationships

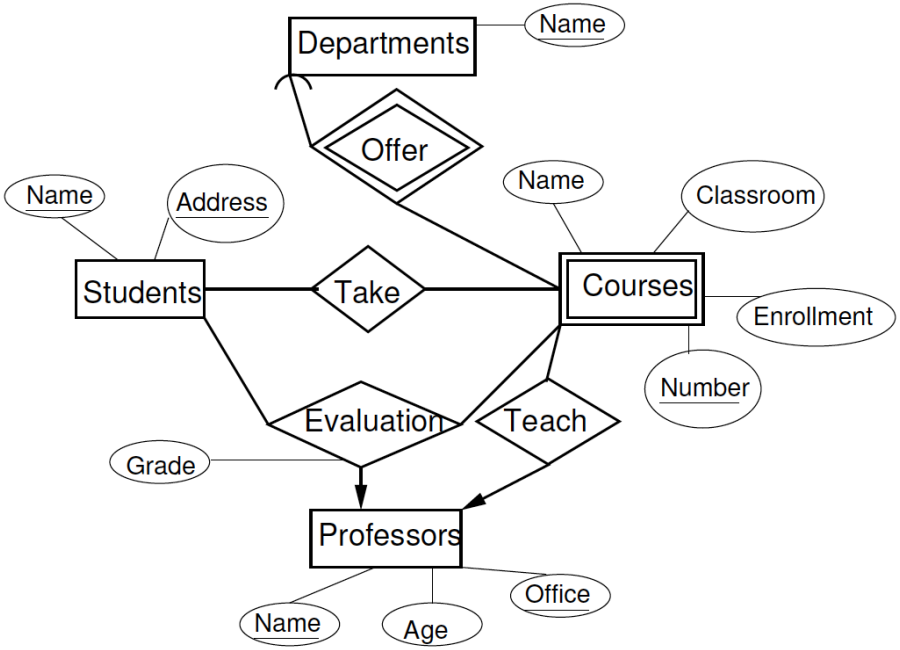


- For each relationship, create a relation with the same name whose attributes are
  - Attributes of the relationship itself.
  - Key attributes of the connected entity sets (even if they are weak)

Schemas for Non-Supporting Relationships

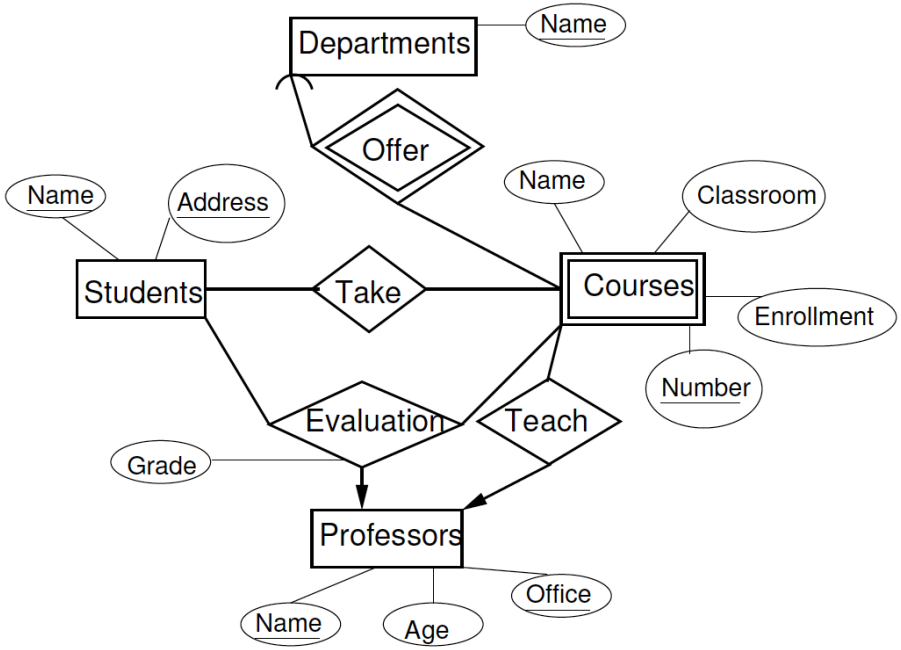


Schemas for Non-Supporting Relationships



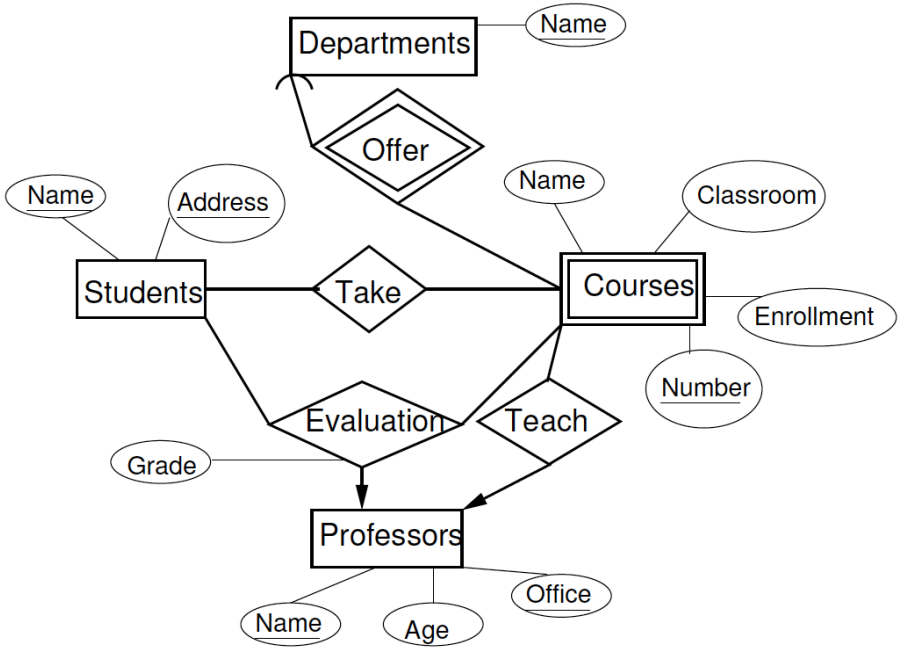
- Take

Schemas for Non-Supporting Relationships



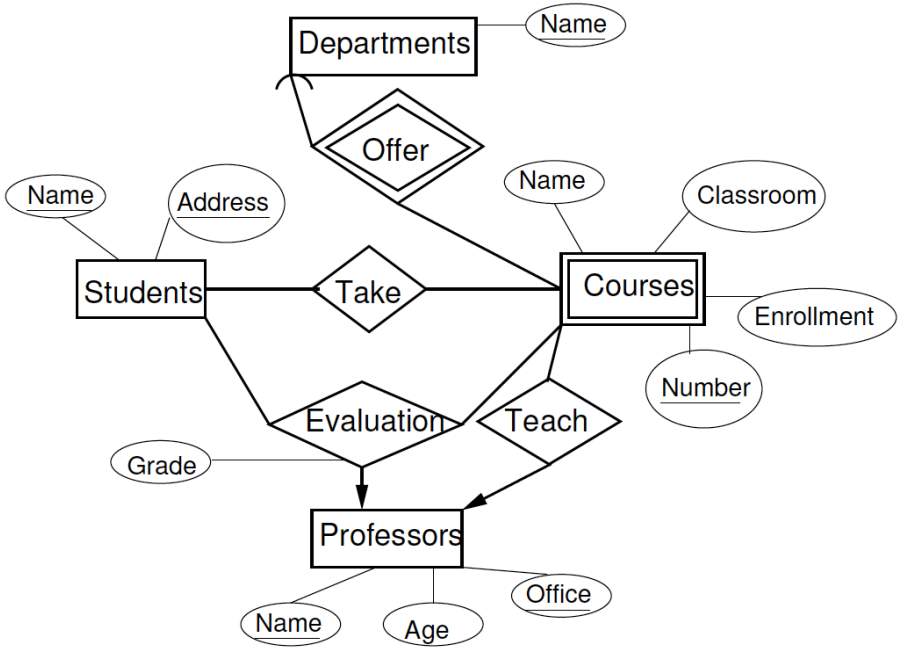
- Take (StudentName, Address, Number, DepartmentName)
- Teach

Schemas for Non-Supporting Relationships



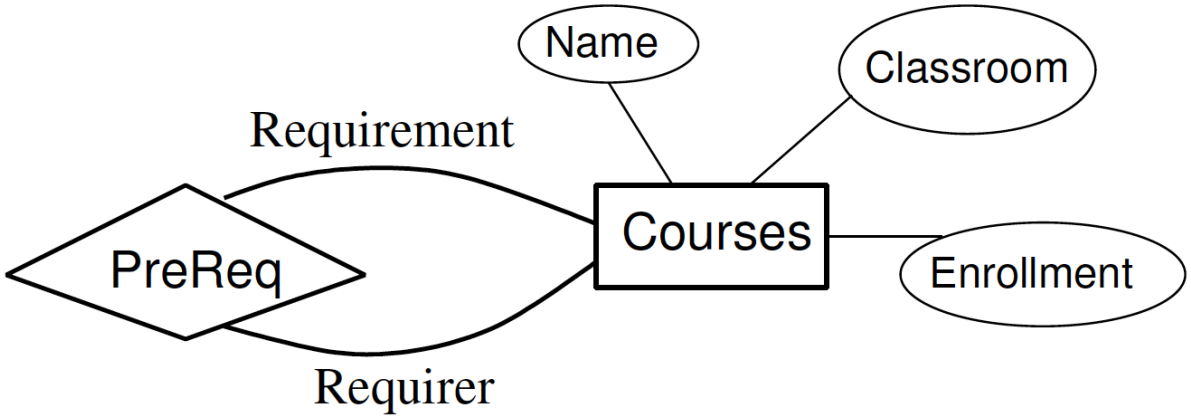
- Take (StudentName, Address, Number, DepartmentName)
- Teach (ProfessorName, Office, Number, DepartmentName)
- Evaluation

Schemas for Non-Supporting Relationships



- Take (StudentName, Address, Number, DepartmentName)
- Teach (ProfessorName, Office, Number, DepartmentName)
- Evaluation (StudentName, Address, ProfessorName, Office, Number, DepartmentName, Grade)

Roles in Relationships



- If an entity set E appears k > 1 times in a relationship R (in different roles), the key attributes for E appear k times in the relation for R, appropriately renamed

PreReq (RequirerNumber, RequirerDeptName, RequirementNumber, RequirementDeptName)

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)

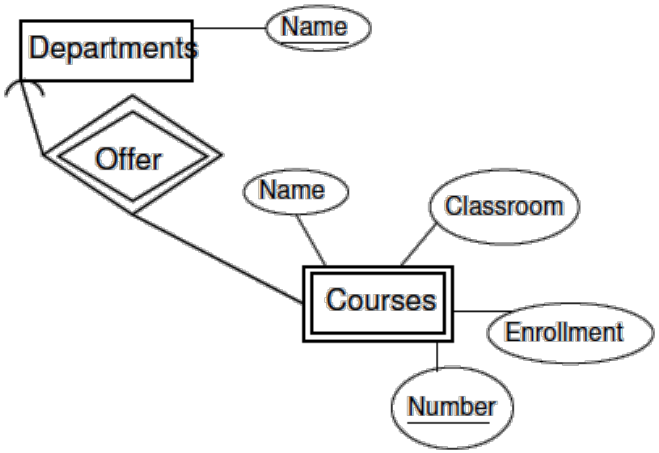
Teach(Number, DepartmentName, ProfessorName, Office)

- 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

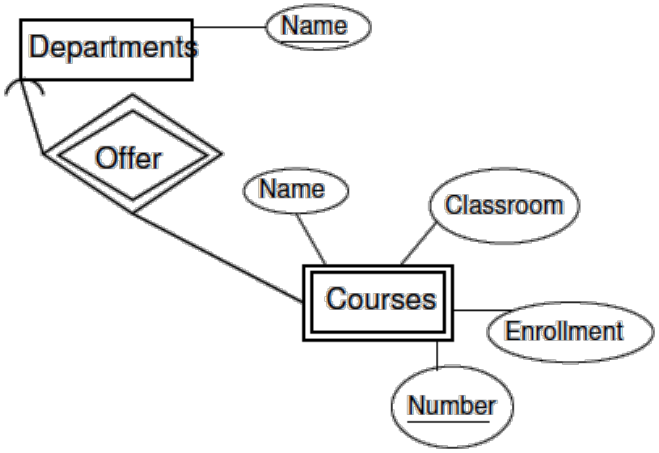
- We can combine into one relation Q
  - The relation for an entity set E
  - all many-to-one relationships R1, R2, …, Rk from E to other entity sets E1, E2, …, Ek respectively
- The attributes of Q are
  - All the attributes of E
  - Any attributes of R1, R2, …, Rk
  - The key attributes of E1, E2, …, Ek
- What if R is a many-many relationship from E to F?

Supporting Relationships



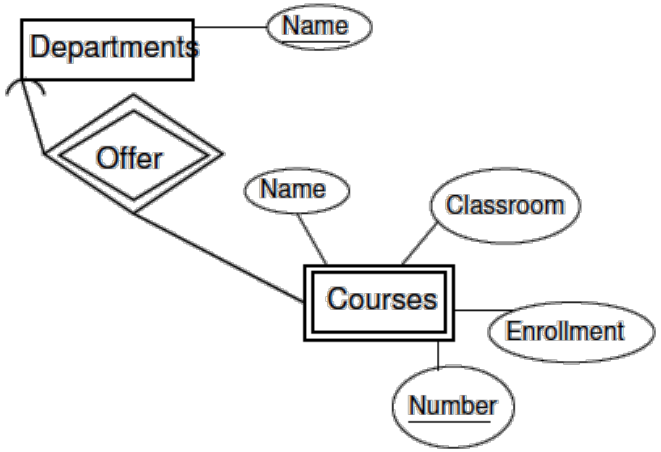
- Schema for Departments is Departments(Name)
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
- What is the schema for offer?

Supporting Relationships



- What is the schema for offer?
  - Offer(Name, Number, DepartmentName)
  - But Name and DepartmentName are identical, so the schema for Offer is Offer(Number, DepartmentName)
  - The schema for Offer is a subset of the schema for the weak entity set, so we can dispense with the relation for Offer

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

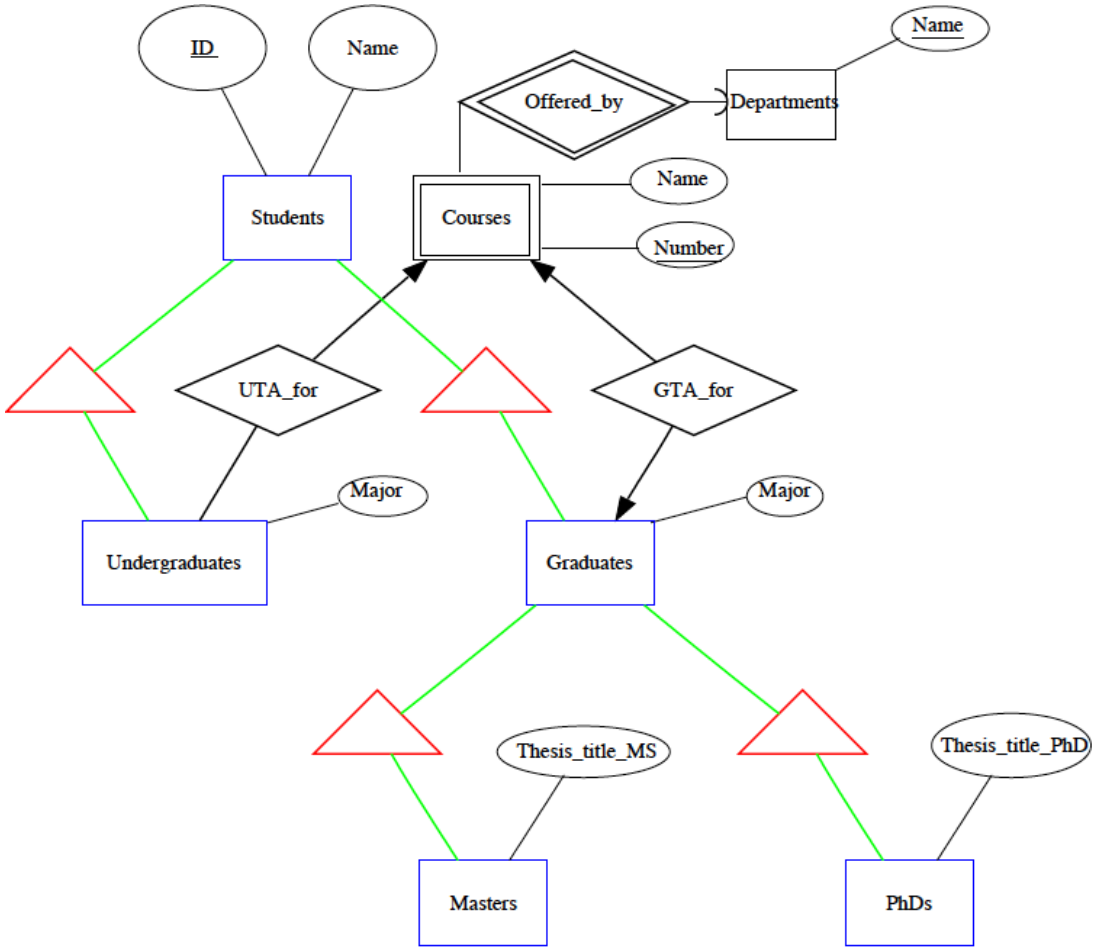
ISA to Relational

- 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

Example: ISA to Relational Method I: E/R Approach

Students(ID, Name)

Undergraduates(ID, Major)

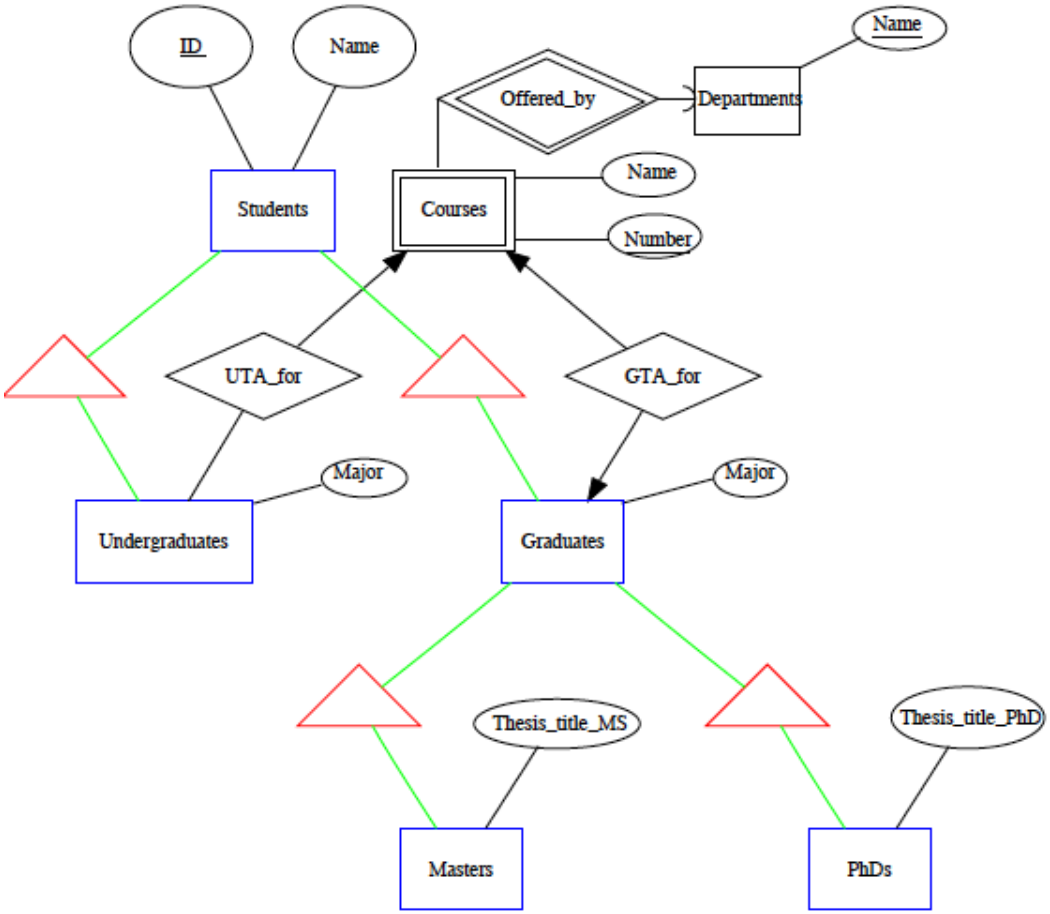
Graduates(ID, Major)

Masters(ID, Thesis_title_MS)

PhDs(ID, Thesis_title_PhD)

UTA_for(ID, CourseNum, DeptName)

GTA_for(ID, CourseNum, DeptName)



ISA to Relational Method II: "Flatten" Approach

- Create a single relation for the entire hierarchy
- Attributes are
  - the key attributes of the root and
  - the attributes of each entity set in the hierarchy
- Handle relationships as before

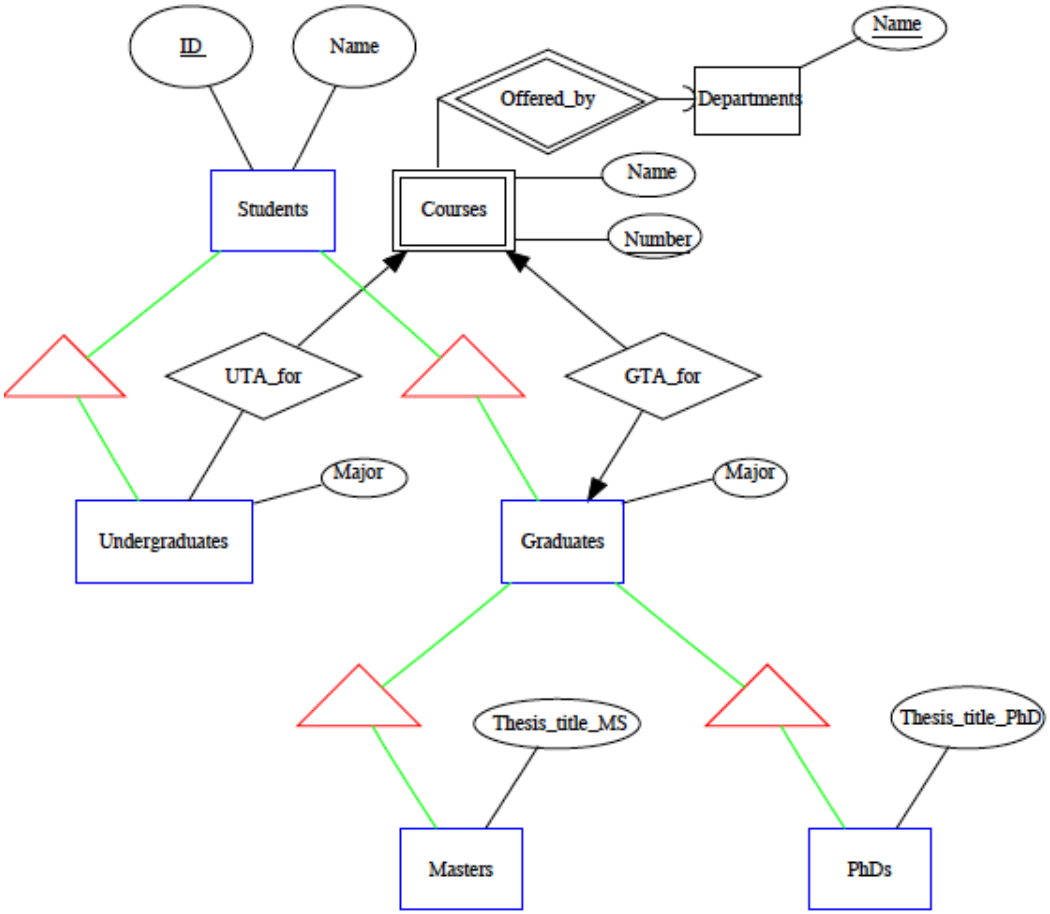
Students(ID, Name, UGMajor, GMajor, Thesis_title_MS, Thesis_title_PhD).

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.
- 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.

Example: ISA to Relational Method III: Object Oriented Approach

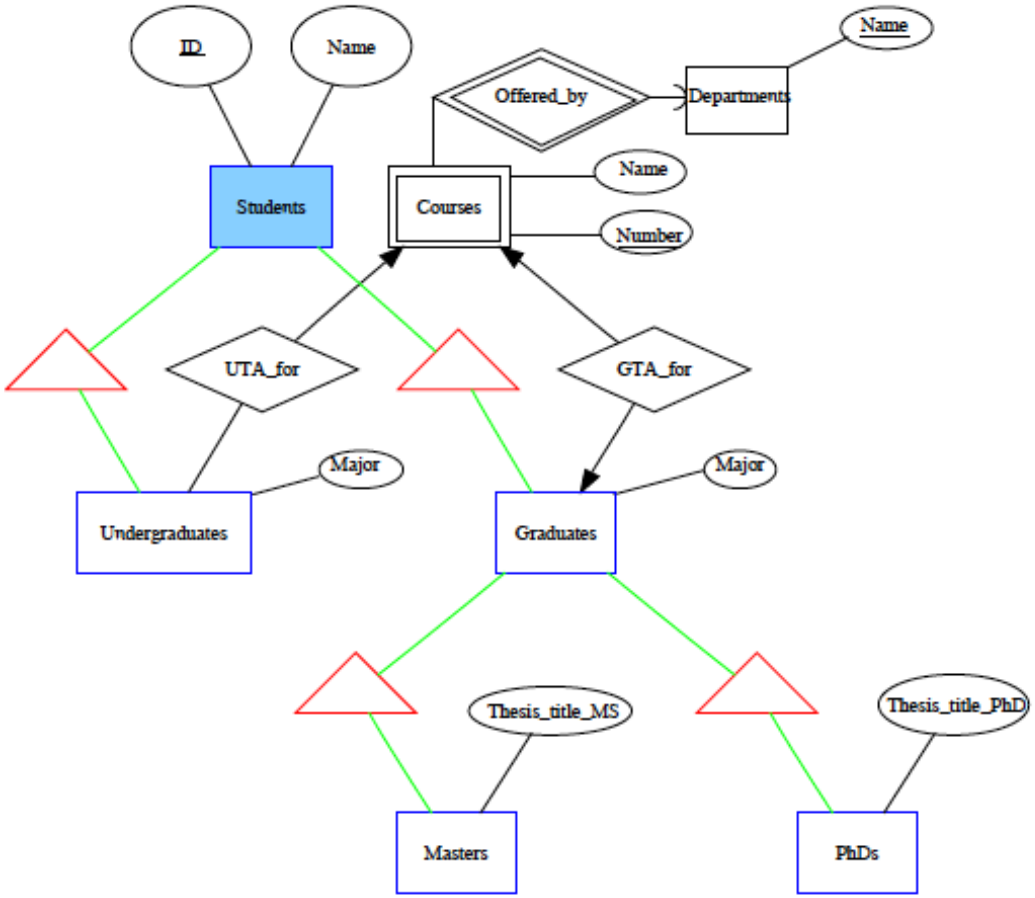
Subtrees are:



Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

Students(ID)



Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

Students(ID)

StudentsUGs(ID, Major)

Example: ISA to Relational Method III: Object Oriented Approach

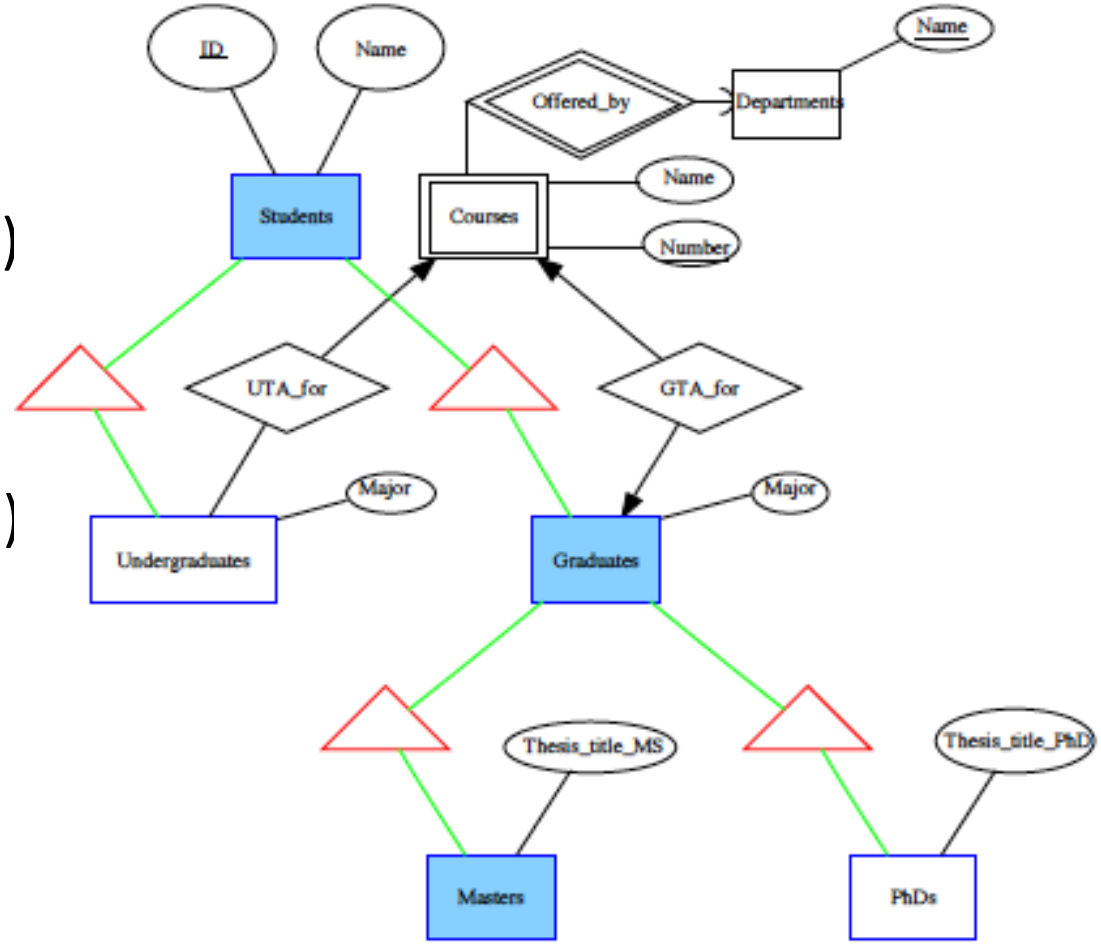
Subtrees are:

- Students(ID)
- StudentsUGs(ID, Major)
- StudentGs(ID, Major)

Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

- Students(ID)
- StudentsUGs(ID, Major)
- StudentGs(ID, Major)
- StudentGsMasters(ID, Major, Thesis_title_MS)



Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

- Students(ID)
- StudentsUGs(ID, Major)
- StudentGs(ID, Major)
- StudentGsMasters(ID, Major, Thesis_title_MS)
- StudentsGsPhDs(ID, Major, Thesis_title_PhD)

Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

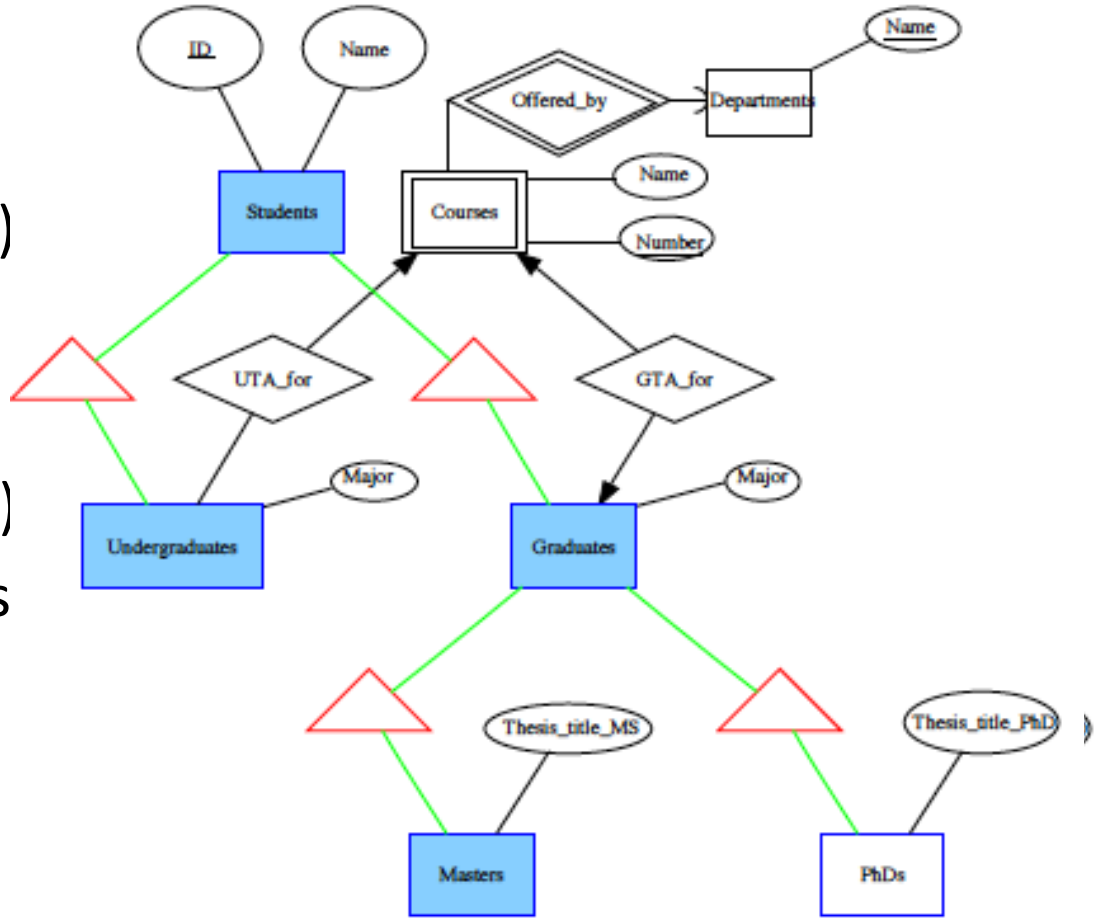
Students(ID)

StudentsUGs(ID, Major)

StudentGs(ID, Major)

StudentGsMasters(ID, Major, Thesis_title_MS)

StudentsUGsGsMasters (ID,UGMinor,GradMinor, Thesis_title_MS)



Example: ISA to Relational Method III: Object Oriented Approach

Subtrees are:

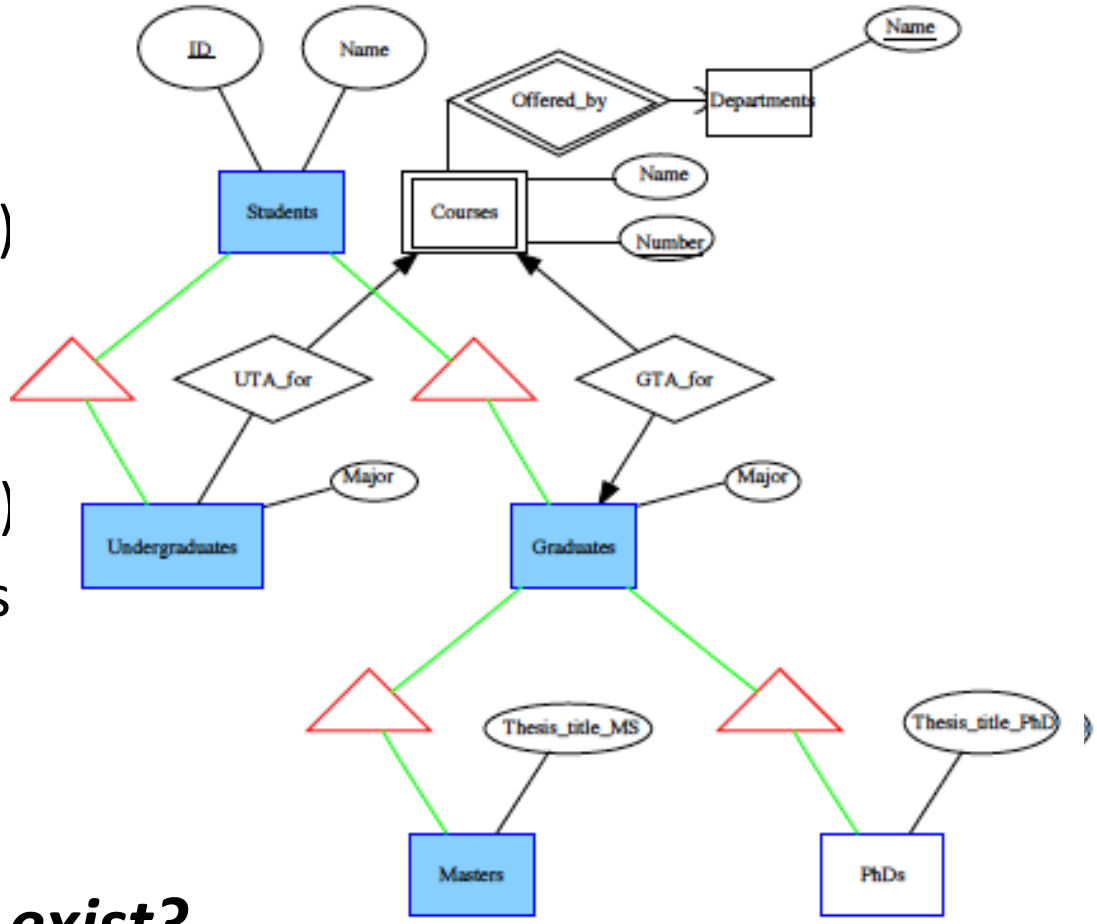
Students(ID)

StudentsUGs(ID, Major)

StudentGs(ID, Major)

StudentGsMasters(ID, Major, Thesis_title_MS)

StudentsUGsGsMasters (ID,UGMinor,GradMinor, Thesis_title_MS)



What other subtrees exist?

Comparison of the Three Approaches

- Answering queries
  - It is expensive to answer queries involving several relations
  - Queries about Students in general
  - Queries about a particular subclass of Students

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
  - E/R
    - Several tuples per entity, but only key attributes are repeated
  - OO
    - Only one tuple per entity