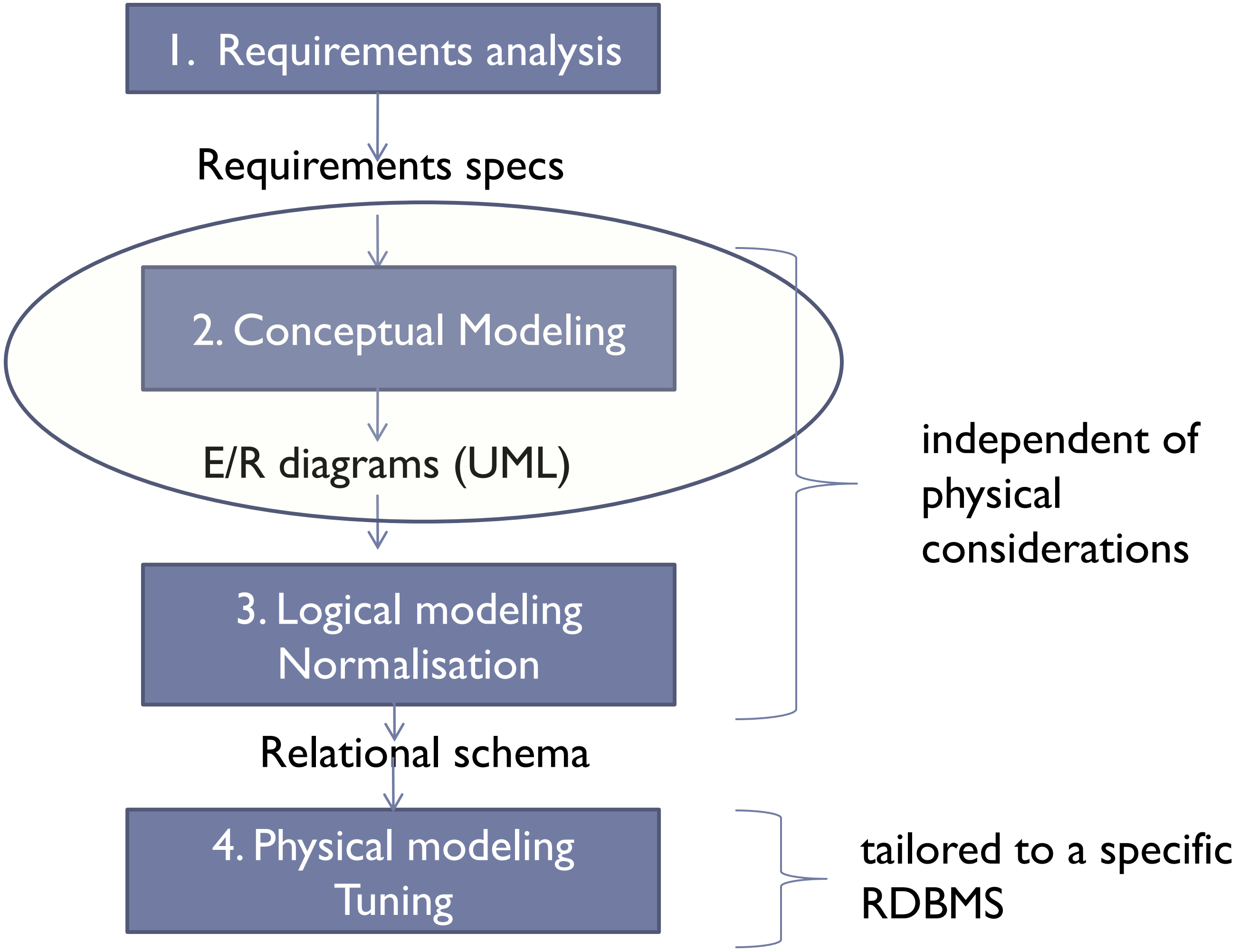
SELECT *lecture*
FROM Databases

WHERE INITCAP(chapter) = 'Relational Database Design',
AND topic = 'The Entity/Relationship Model'

Mihaela Elena Breabăn

Relational Database Design Methodology



1. Requirements analysis

Requirements specs

2. Conceptual Modeling

E/R diagrams (UML)

3. Logical modeling
Normalisation

Relational schema

4. Physical modeling
Tuning

independent of physical considerations

tailored to a specific RDBMS

Plan

- Schema design: motivation
- E/R schema design using Chen's notation
  - E/R concepts
  - Modeling constraints
  - Connection Traps
- E/R schema design in UML
- From E/R and/or UML to the relational schema

Schema design: motivation

- Several schemas may be designed for a database
  - Some are (much) better than others
    - redundancy?
    - efficiency?
    - consistency?

- How generate good schemas?

- Two design approaches:
  - E/R data modeling (Chen,'76)
  - Schema decomposition - normalization (Codd, '70-'74)
- Usually they are applied subsequently

Basic E/R concepts (Chen 1976)

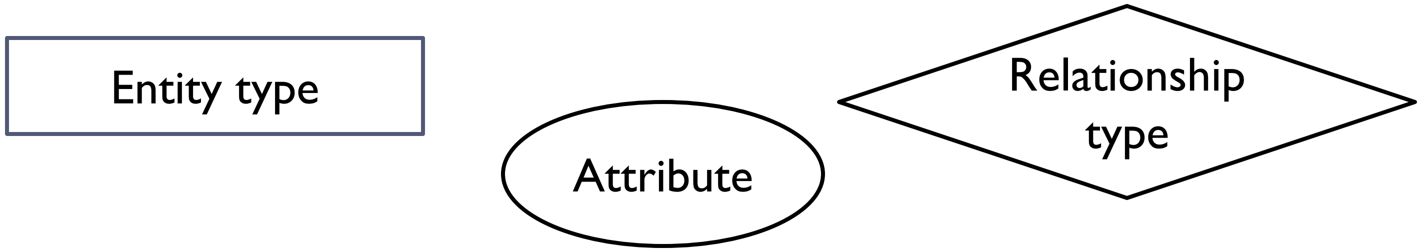
- Entity
  - Data that can be modeled as an object having independent existence
  - An *entity type* groups objects having the same properties; each entity type is described by a name and a list of properties; each object in the group will be called an entity (occurrence/instance)
  - Any entity must be uniquely identifiable within its class

- Relationship
  - Models the interaction/association between entities
  - A *relationship type* is a set of associations among entities types; each relationship type is given a (descriptive) name
  - A relationship (occurrence/instance) is uniquely identifiable by the participating entities
  - Degree of a relationship type = the number of entity types involved
    - binary, ternary…
    - A recursive relationship type involves one entity type more than once, playing distinct roles

- Attributes
  - For entity types these are specific properties describing the independent objects
  - For association types these may be
    - Properties of the involved entity types
    - Specific properties of the relationship type, storing new information related to the association

E/R diagrams

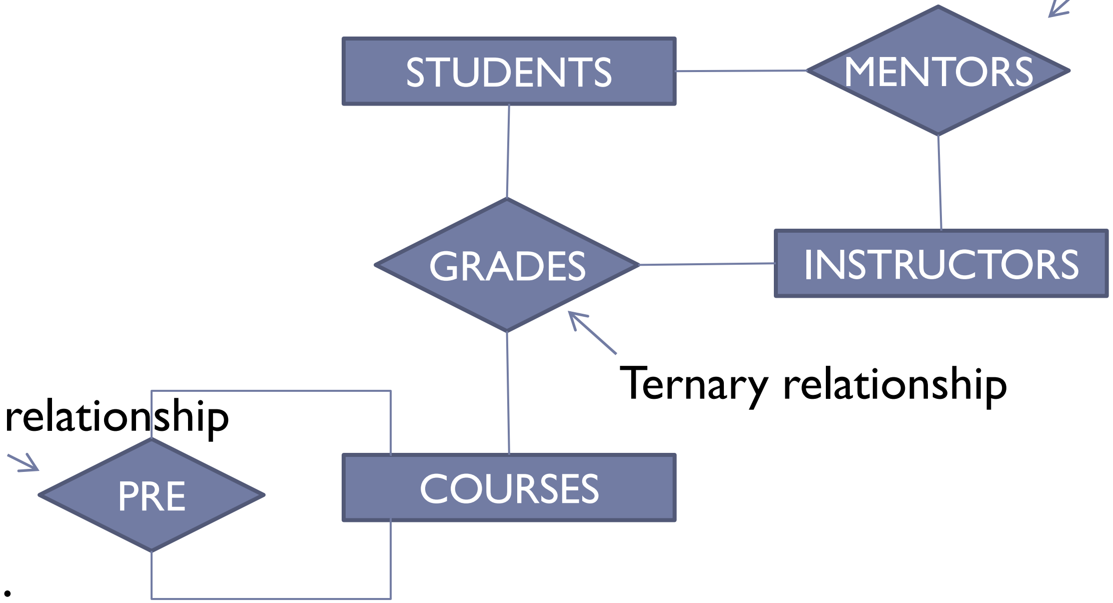
- Graphical notation for E/R concepts
  - There exist several graphical standards, we will use for the moment Chen's notation:



- E/R diagram = a graph where
  - Entity types, relationship types and attributes are vertices/nodes
  - Edges connect
    - Entity type nodes with relationship types nodes
    - Entity type nodes with attribute nodes
    - Relationship type nodes with attribute nodes

Example

- Let’s design a database to store information about
  - Students
  - Instructors
  - Courses
  - Grades
  - Mentors



Binary relationship

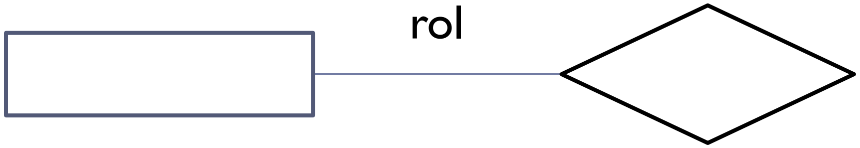
Ternary relationship

Recursive relationship

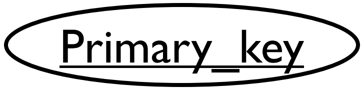
- Requirements:
  - We can determine any student’s grades for the courses he/she attended and the accumulated credits
  - We can determine a student’s mentor
  - We store the prerequisites for each course

More E/R concepts

- Roles
  - Explain the function of the entity type in an association

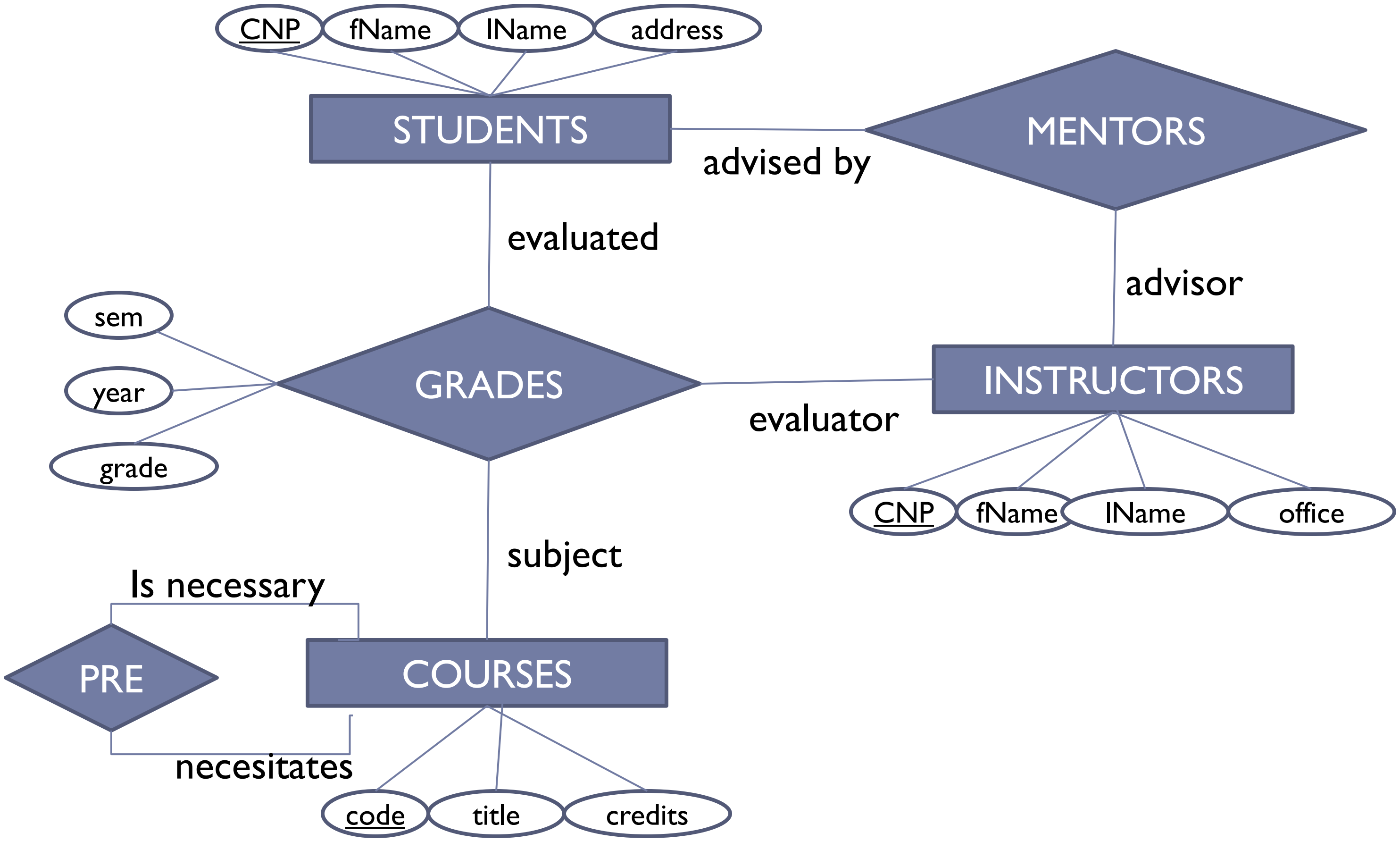


- Primary key
  - The minimal set of attributes that is designated to uniquely identify an entity instance or a relationship instance
  - It is mandatory for entity types in order to identify the entity instances that are involved in relationships



- Foreign key – defined only for relationships!
  - A set of attributes that plays as primary key for the involved entity types

Example

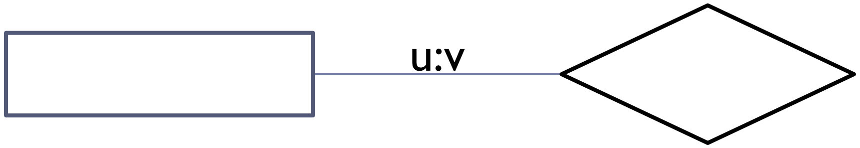


Which are the foreign keys for our relationship types?

Multiplicity constraints

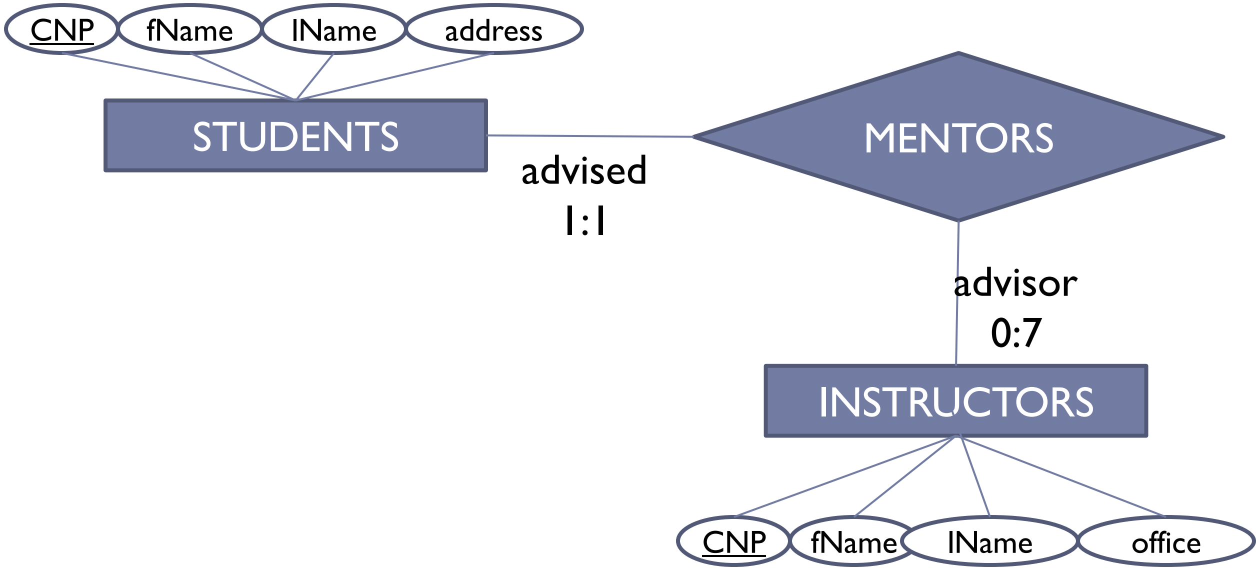
- The E/R model allows us to declare constraints on the number of relationship instances to which an entity instance may participate

- Let R be a relationship type among *n* entity-types denoted E_i, $i=1..n$. The database satisfies the (E_i, u,v,R) constraint if each entity instance of E_i participates in at least *u* and most *v* relationship instances from R.

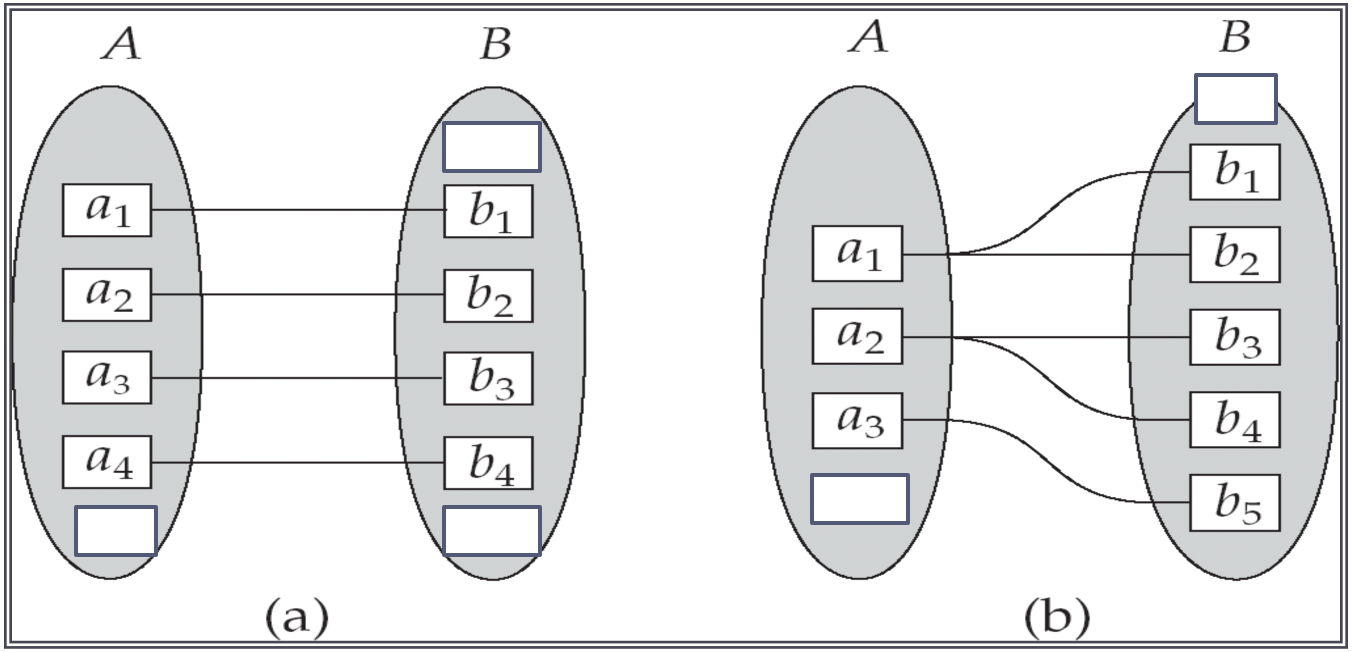


Example

- (Students1,1Mentors)
- (Instructors,0,7,Mentors)
- Interpretation: Every student has exactly one instructor as advisor/mentor and an instructor can advise at most 7 students



Multiplicity constraints for binary relationships (1)

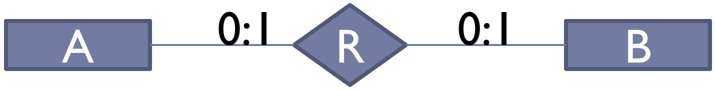


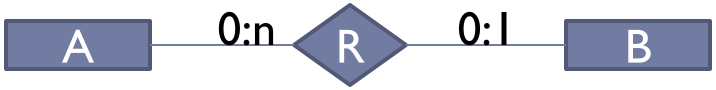
(a)

(b)

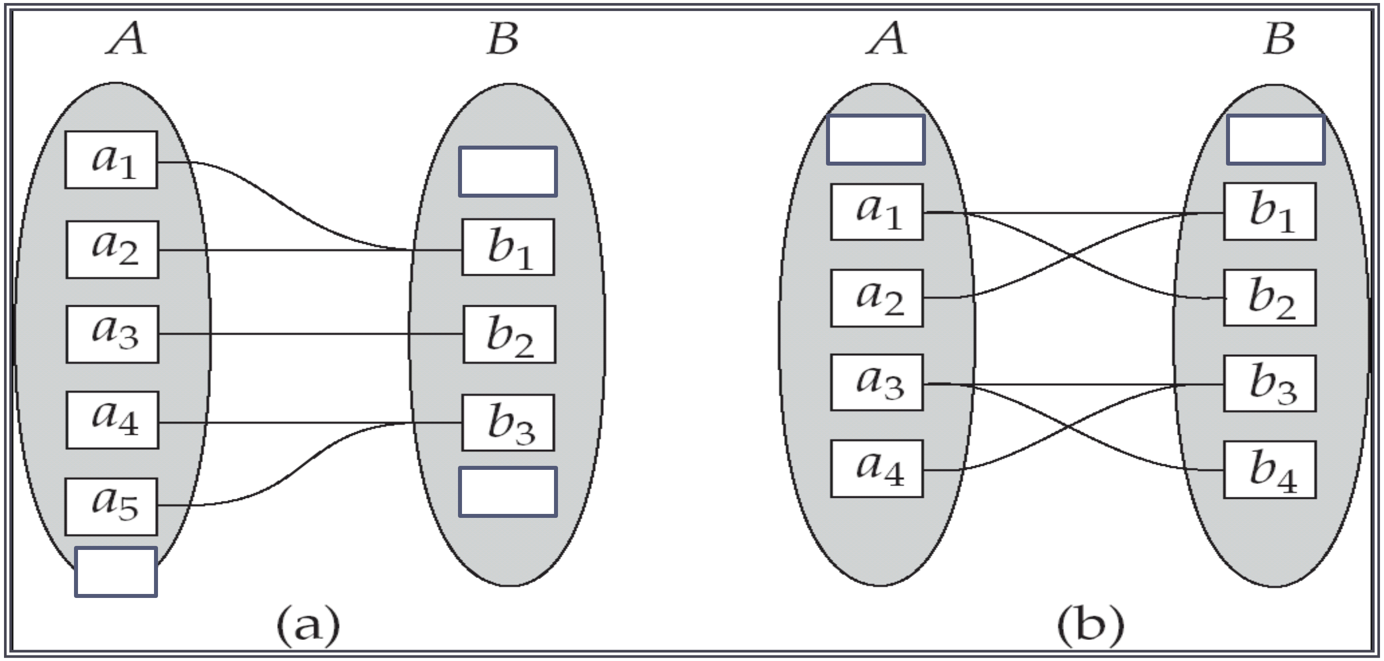
a) One-to-one relationship
(A,0,1,R) (B,0,1,R)

b) One-to-many relationship
(A,0,n,R) (B,0,1,R), n>1



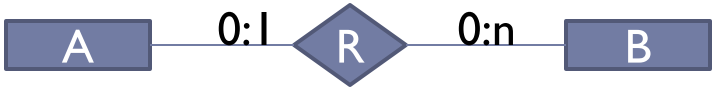


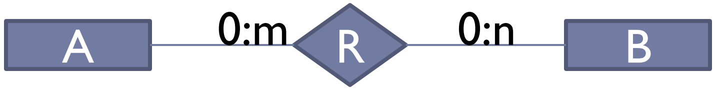
Multiplicity constraints for binary relationships (2)



a) Many-to-one relationship
(A,0,1,R) (B,0,n,R)

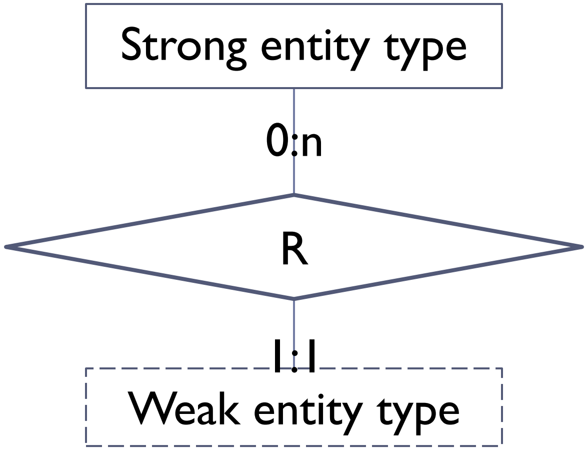
b) Many-to-many relationship
(A,0,m,R) (B,0,n,R), m,n>1





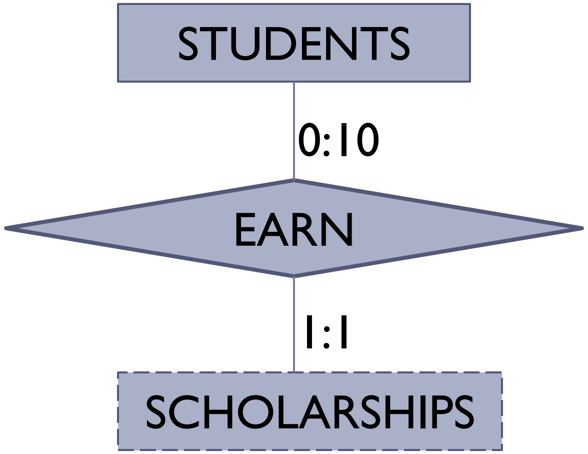
Weak entity type

- An entity type is said to be weak if its instances depend to some extent on the existence of entities from other entity type (existential dependence)

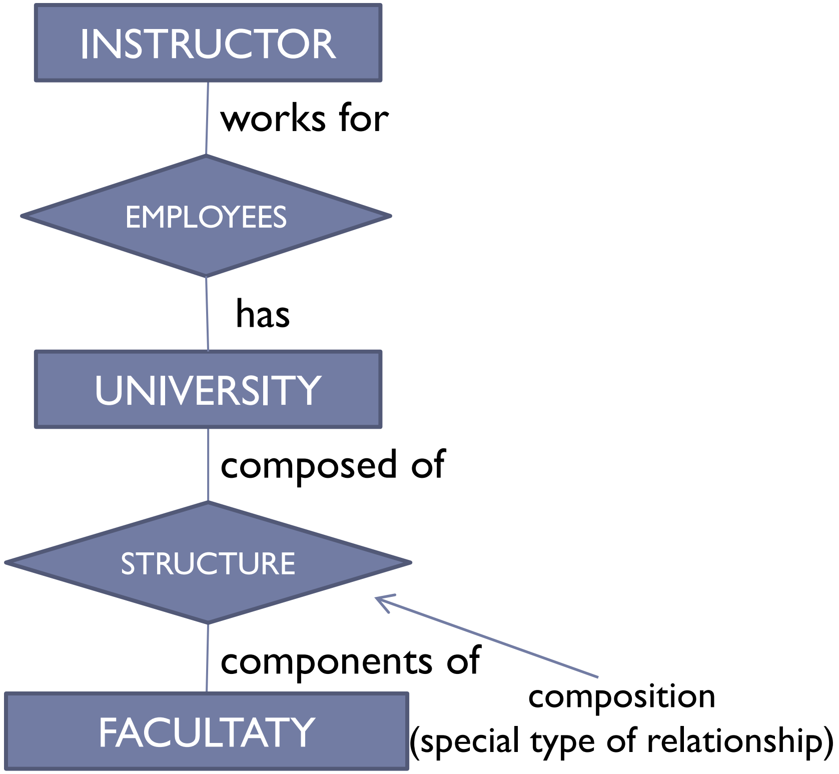


- Has no key
- Must satisfy the multiplicity constraint (Weak-entity,1,1,R), participating in a one-to-one or one-to-many relationship with respect to the strong entity

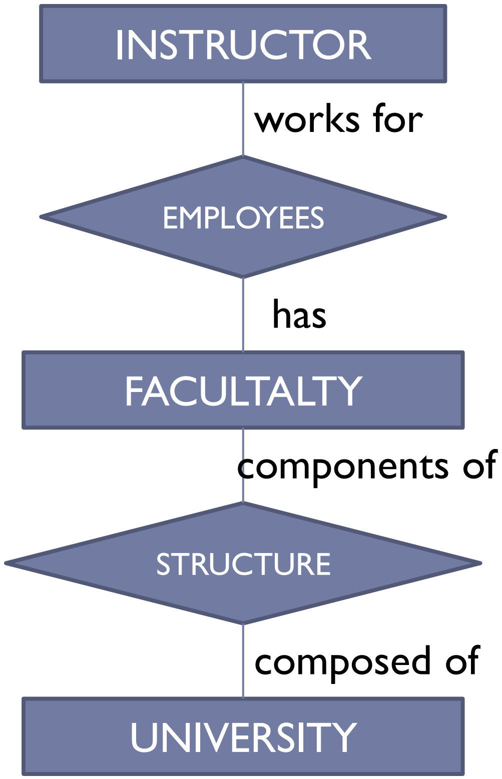
Weak entity - example



Connection traps (Fan traps)



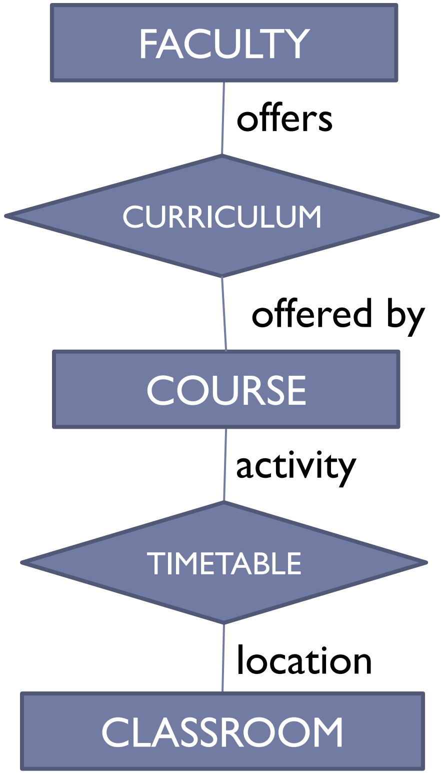
composition
(special type of relationship)

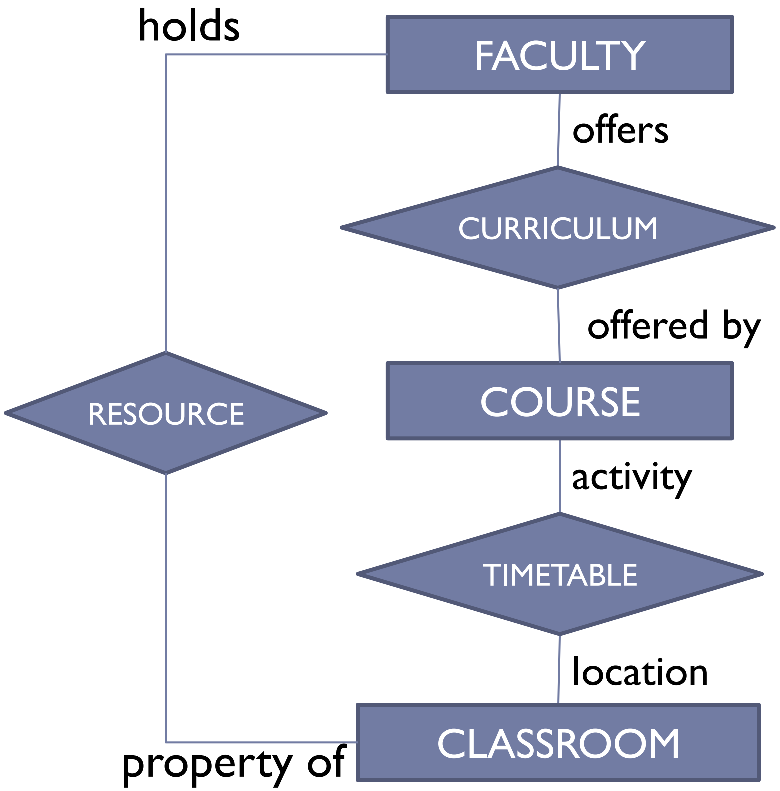


Problem:
In which department profesor X is working?

Solution:
Restructured model

Connection traps (Chasm traps)



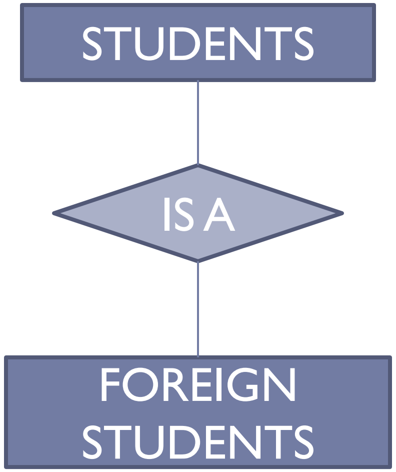


Problem:
Which are the classrooms belonging to some faculty?

Solution:
New relationships

Enhancements of the E/R model Specializations

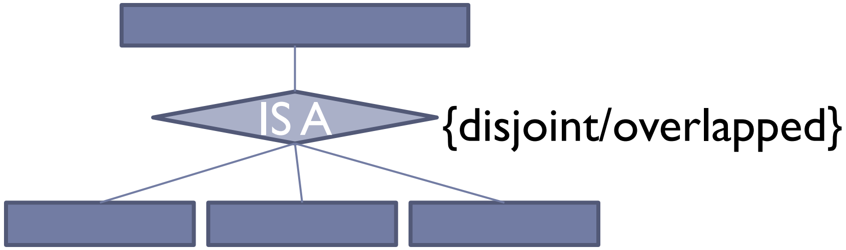
- Subgroups of entity instances
  - Having new distinctive attributes or
  - Participating in relationships not common to all entity instances
  - Correspond to a specialized entity type which is involved in a IS-A relationship type relative to the basic entity type



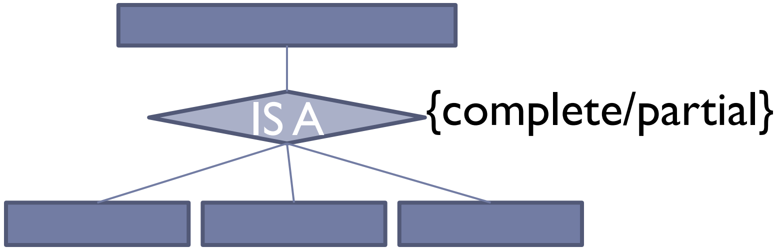
Specialization constraints

- Instances of the specializations inherit all the attributes and the relationships of the basic entity type, including the key

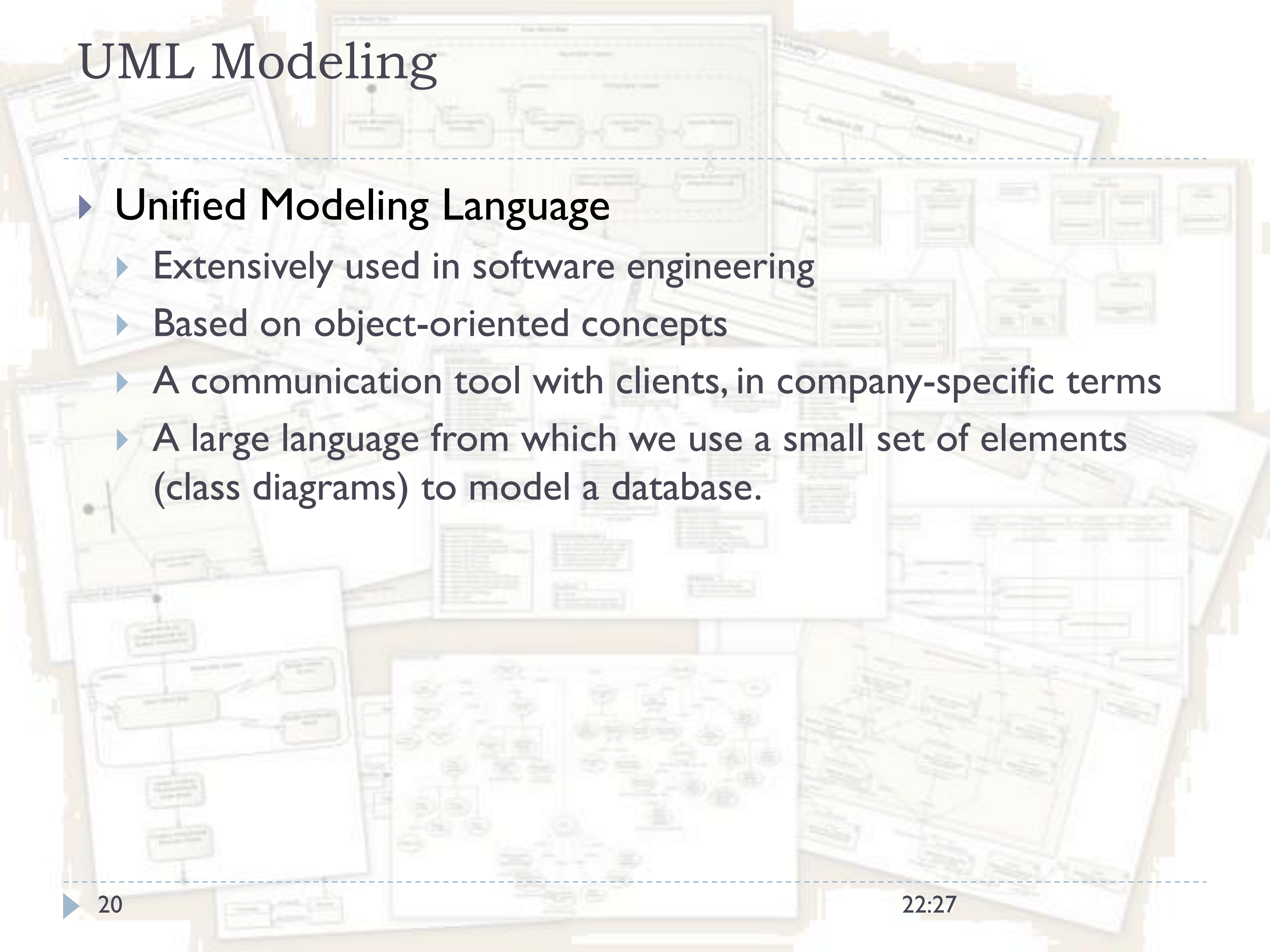
- An instance of an entity type may belong only to one or to several specializations:
  - Disjoint specializations (exclusive)
  - Overlapped specializations



- An instance of an entity type must or must not belong to at least one specialization:
  - Complete specializations
  - Incomplete (partial) specializations



UML Modeling



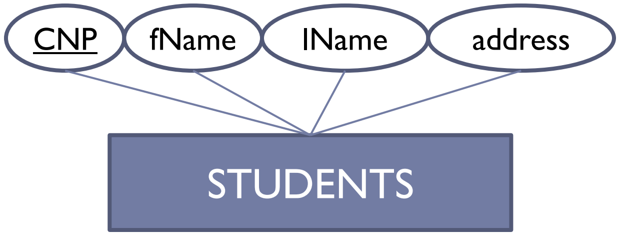
- Unified Modeling Language
  - Extensively used in software engineering
  - Based on object-oriented concepts
  - A communication tool with clients, in company-specific terms
  - A large language from which we use a small set of elements (class diagrams) to model a database.

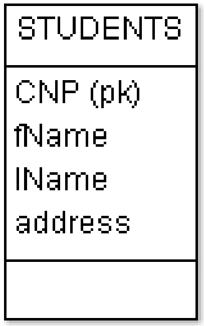
E/R – UML Correspondence

E/R	UML
Entity type	Class
Relationship type NOT having own attributes	Association
Relationship type with own attributes	Association class
Specialization	Subclass
	Composition and Aggregation

Classes

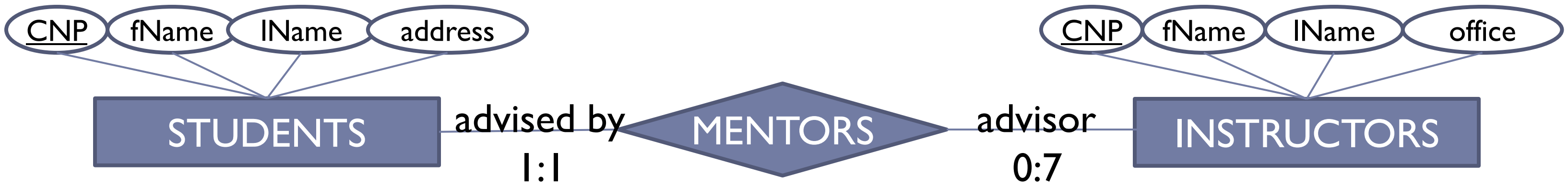
- Components: names, attributes, methods
- DB: name, attribute (primary key)





Associations

- Express the associations between objects belonging to two classes
- BD: relationship types among entity types



CNP fName lName address

STUDENTS

advised by

1:1

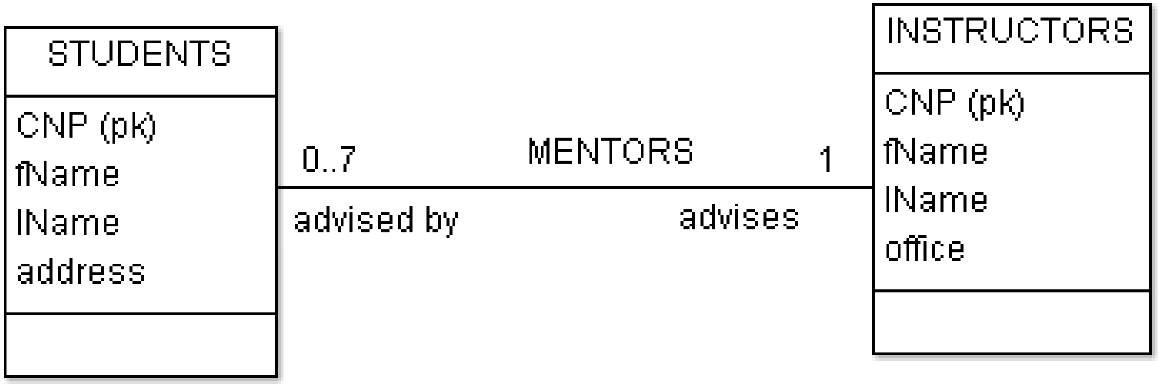
MENTORS

advisor

0:7

INSTRUCTORS

CNP fName lName office



STUDENTS

CNP (pk)
fName
lName
address

0..7

MENTORS

1

advised by

advises

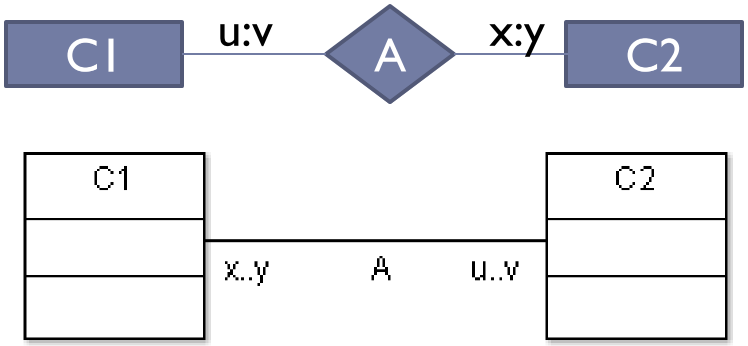
INSTRUCTORS

CNP (pk)
fName
lName
office

- Obs: multiplicity constraints are specified in reverse order compared to E/R diagrams

Associations
Multiplicity constraints

- Constraints
  - (C1,u,v,A)
  - (C2,x,y,A)



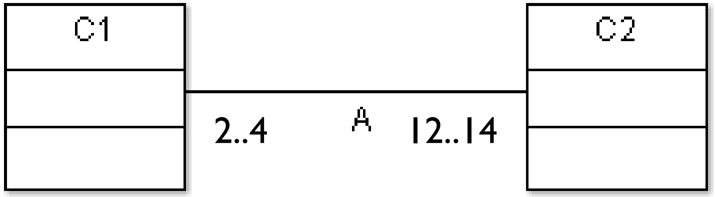
- Every entity in C1 is associated to at least *u* and at most *v* entities in C2
- Every entity in C2 is associated to at least *x* and at most *y* entities in) C1

x..y	**u..v**	**Relationship**
0..1	0..1	One-to-one incomplete
1..1 (1)	1..1 (1)	One-to-one complete (the default)
0..1	0..* (*)	One-to-many incomplete
…	…	…

Quiz

1. Model the relationship between STUDENTS and UNIVERSITIES. A student may study to at most 2 universities and must study to at least one. A university may have at most 10.000 students.

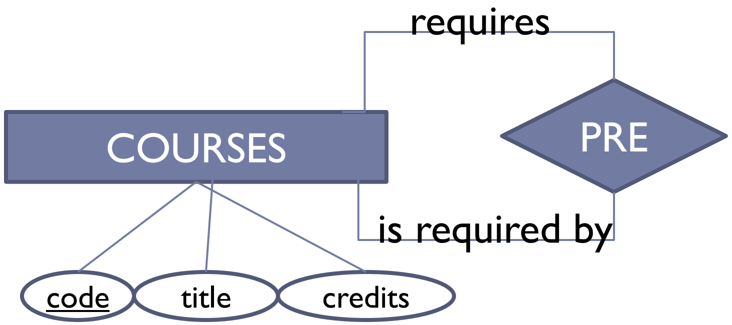
2. Given the relationship

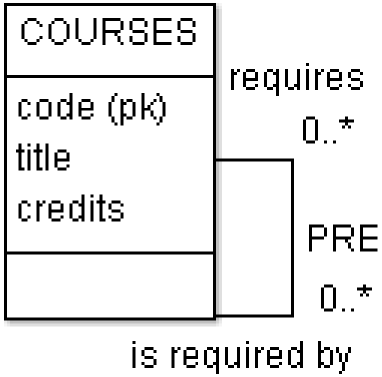


What is the minimum number of entities in C1?

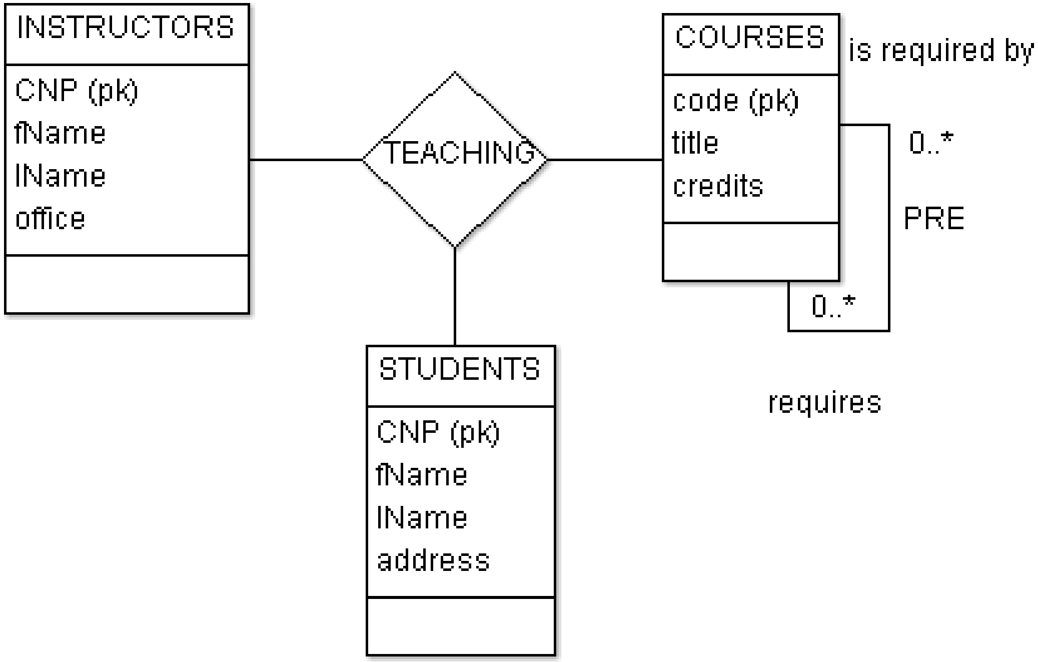
What about C2?

Recursive relationships

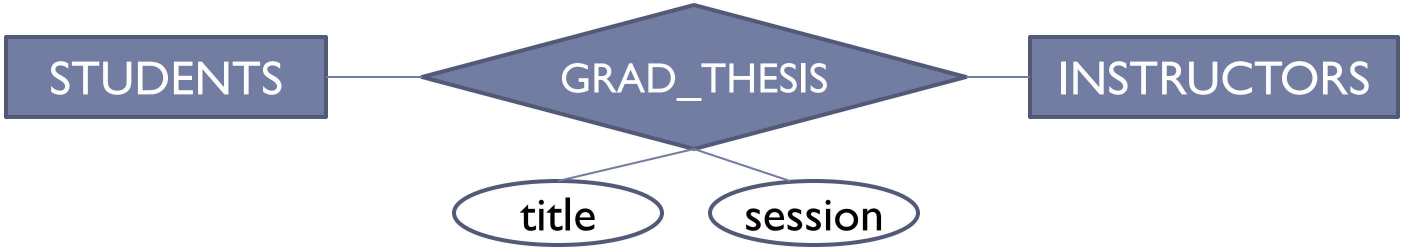


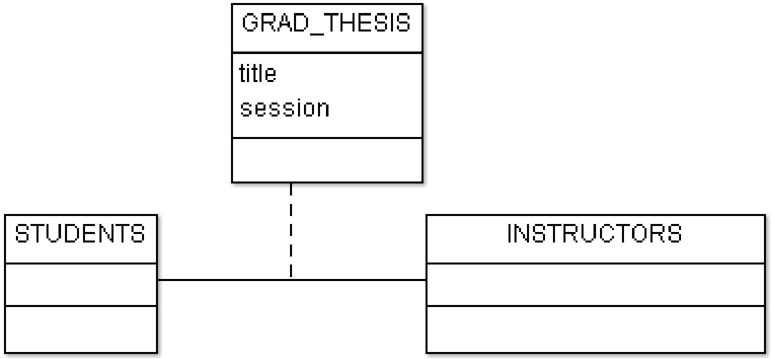


N-ary relationships

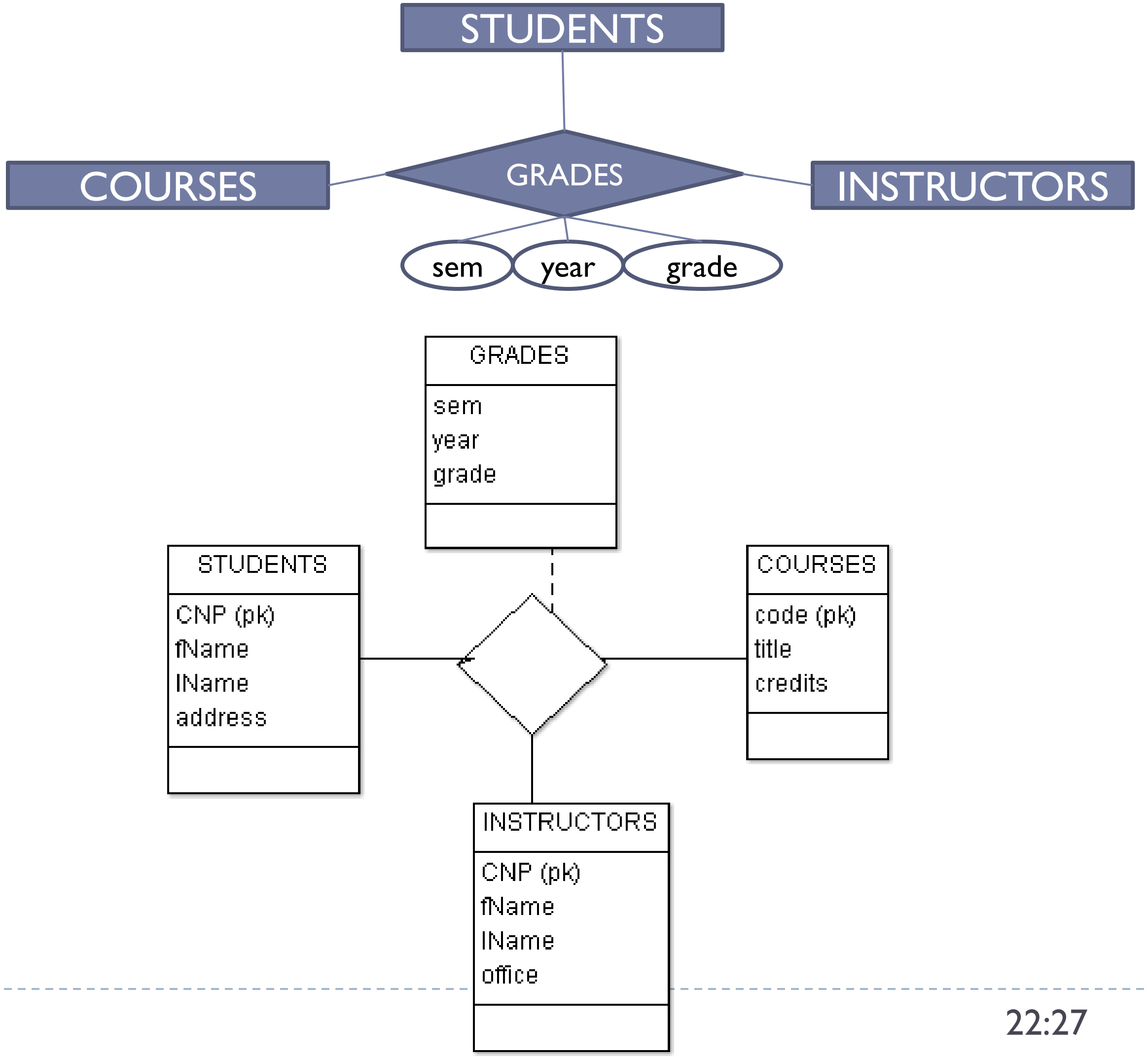


Association classes



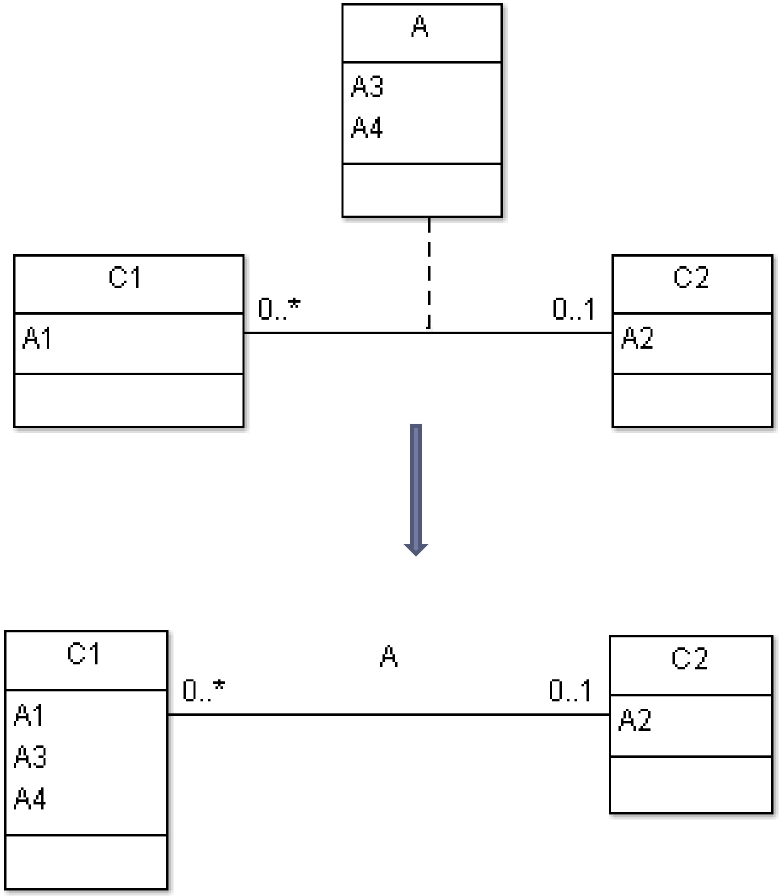


Association classes

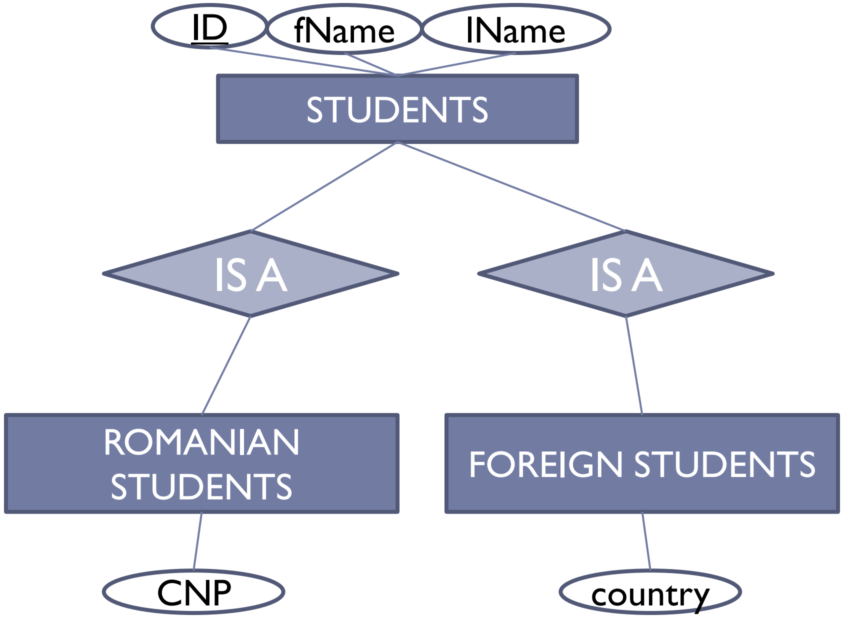


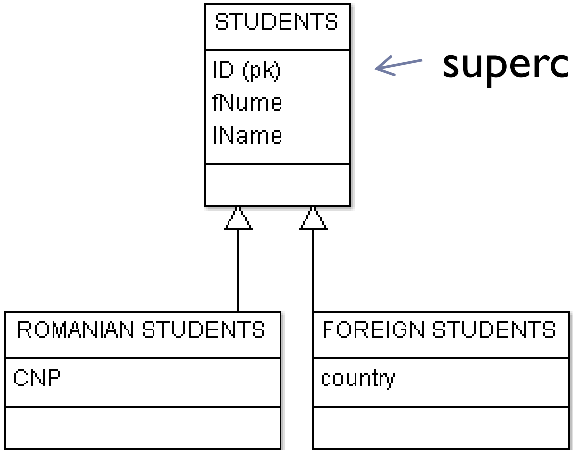
Eliminating association classes

- When we have 0..1 or 1..1 multiplicity:



Subclasses (1)

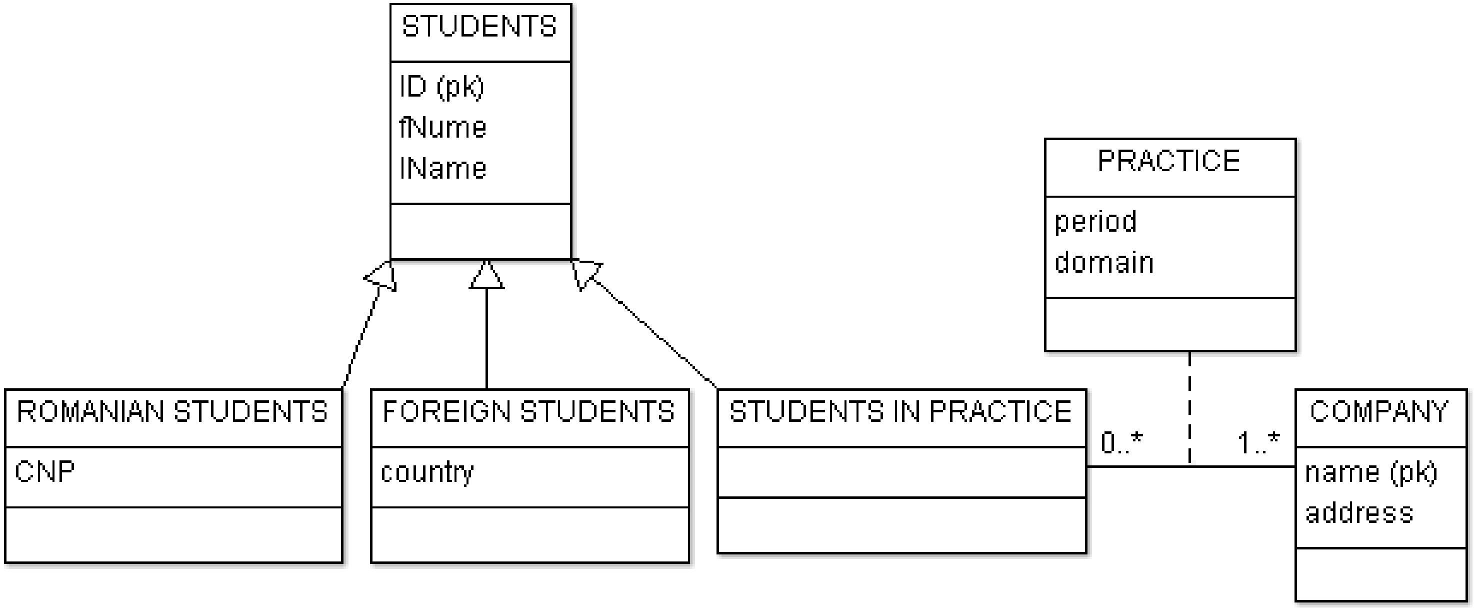




superclass

Complete, disjoint specialization

Subclass (2)



STUDENTS

ID (pk)
fNume
lName

ROMANIAN STUDENTS

CNP

FOREIGN STUDENTS

country

STUDENTS IN PRACTICE

PRACTICE

period
domain

COMPANY

name (pk)
address

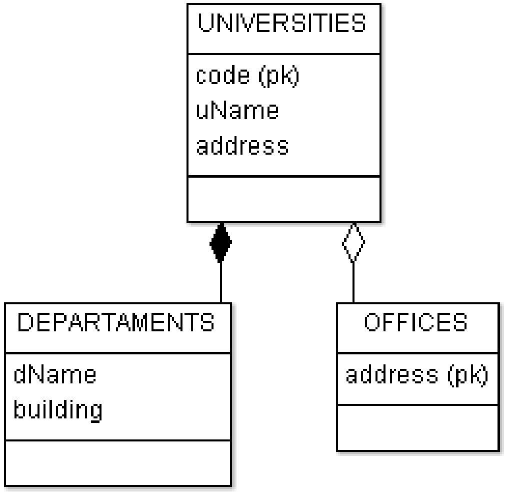
0..*

1..*

Complete overlapping specialization

Compositions and aggregation

- Entities of a type are parts of entities of another type
- Special cases of relationships

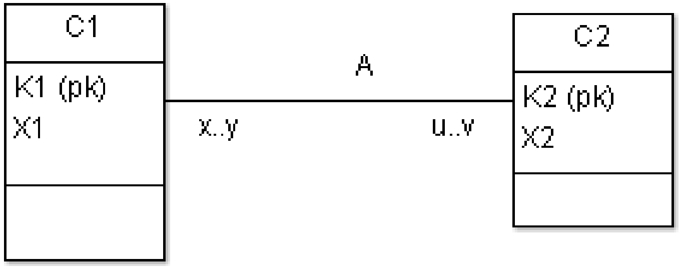


- Composition: **all** entities of a *partial* class belong to entities of the *composed* class; a partial class usually corresponds to a weak entity type (multiplicity 1..1; no primary key);
- Aggregation: **some** entities of an entity type belong to entities of another entity type(multiplicity 0..1)

Transforming E/R, UML into relational schemas

E/R	UML	Relational schema
Entity type	Class	Relation with primary key
Relationship NOT having own attributes	Association	Relation with foreign keys
Relationship having own attributes	Associations class	Relation with foreign keys and other attributes
Specialization	Subclass	Relation with primary key (from superclass) and specialized attributes
	Compositions and aggregation	Relation with foreign key and own attributes

Entity types and relationships



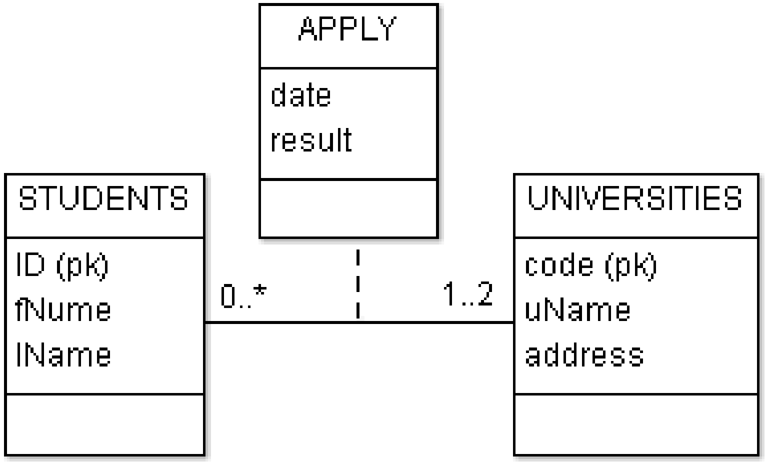
{C1($\underline{\text{K1}}$, X1), C2($\underline{\text{K2}}$, X2), A(K1,K2)}

- The primary key of a relationship depends on multiplicity:

x..y	u..v	Primary key for A	Observations
0..1 1..1	*	K2	There's no need for relation A {C1($\underline{\text{K1}}$, X1), C2($\underline{\text{K2}}$, X2,K1)}
*	0..1 1..1	K1	There's no need for relation A {C1($\underline{\text{K1}}$, X1,K2), C2($\underline{\text{K2}}$, X2)}
*	*	(K1,K2)	

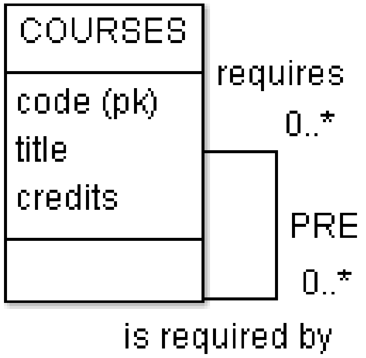
Quiz

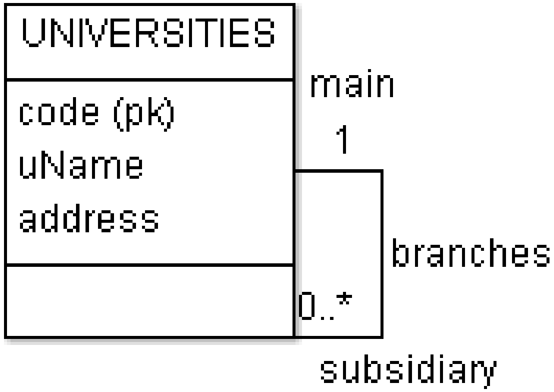
- For the E/R diagram below



Is it possible to eliminate the relation corresponding to the relationship?

Recursive relationships

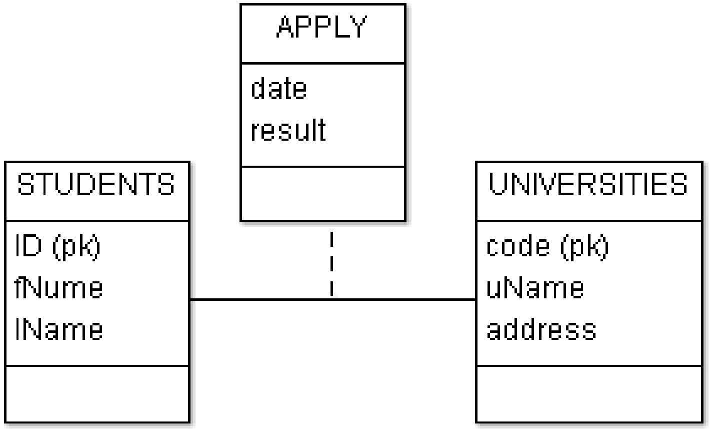




{COURSES (code, title, credits)
PRE (code1, code2)}

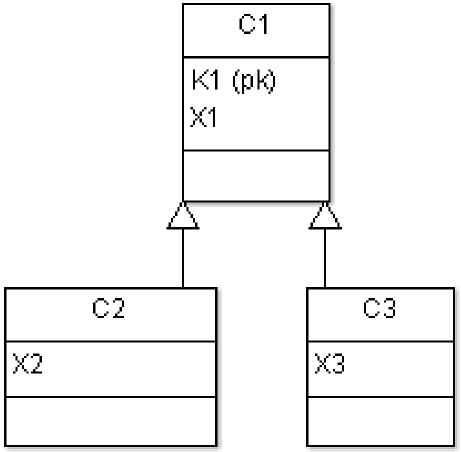
{UNIVERSITIES (code, uName,address)
BRANCHES (subCode, mainCode)}

Association classes



{STUDENTS (ID, fName, lName)
UNIVERSITIES (code, uName, address)
APPLY (ID, code, date, result)}

Specializations/Subclasses



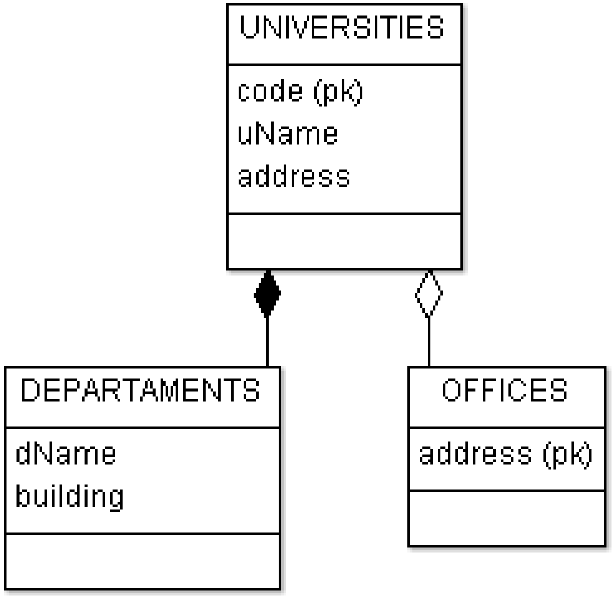
- Choices:
  1. Superclass relation + subclass relations containing foreign key and specialized attributes
     - C1(K1,X1), C2(K1,X2), C3(K1,X3)
  2. Independent superclass relation + subclass relations containing both general attributes and specialized attributes
     - C1(K1,X1), C2(K2,X1,X2), C3(K2,X1,X3)
  3. One relation containing everything
     - C(K1,X1,X2,X3)

Quiz

- Let S be a superclass with a number of subclasses. Consider that the specialization is incomplete and overlapping. If *n1*, *n2* and *n3* represent the total number of tuples which would be generated for each of the three previous translation schemes, which of the following is true?

  - n1<n2<n3
  - n1≤n2≤n3
  - n3<n2<n1
  - n3 ≤n2 ≤n1

Composition and aggregation



{ UNIVERSITIES(<u>code</u>, uName, address)
DEPARTMENTS(codeU, dName, building)
OFFICES (codeU, <u>address</u>)}

does NOT accept NULL

accepts NULL

ER / UML Modeling Summary

- PROS
  - Popular technique in conceptual modeling
  - Constructions which allow us expressing our own personal point of view on data/application
  - Allows expressing some classes of constraints (primary keys, foreign keys, multiplicity…)

- CONS
  - Subjectivity (entity / attribute, entitity / relationship, subclass, composition)
  - Does not allow modeling all kinds of dependencies
  - Necessitates future normalisation steps

Bibliography

- Chapters 11 and 12 in Thomas Connolly, Carolyn Begg: *Database Systems: A Practical Approach to Design, Implementation and Management, (5th edition)* Addison Wesley, 2009

- Tools:
  - https://creately.com (ER diagrams, UML class diagrams)
  - http://diagramo.com/ (ER diagrams)
  - http://argouml-downloads.tigris.org/nonav/argouml-0.32.2/ArgoUML-0.32.2.zip (UML class diagrams)