



# *SELECT lecture* **FROM Databases**

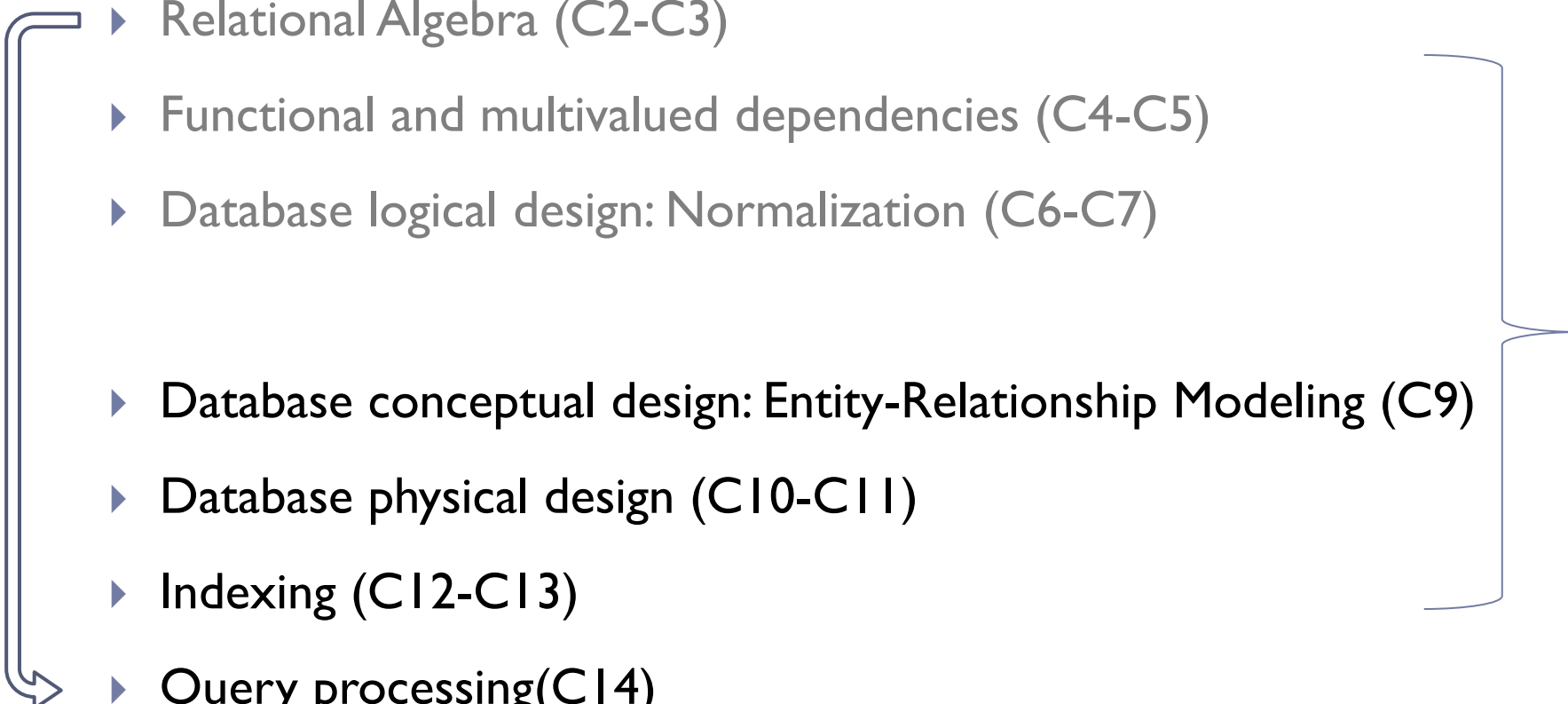
WHERE INITCAP(chapter) = **'Relational Database Design'**,  
AND topic = **'The Entity/Relationship Model'**

Mihaela Elena Breabăn

© FII 2023-2024

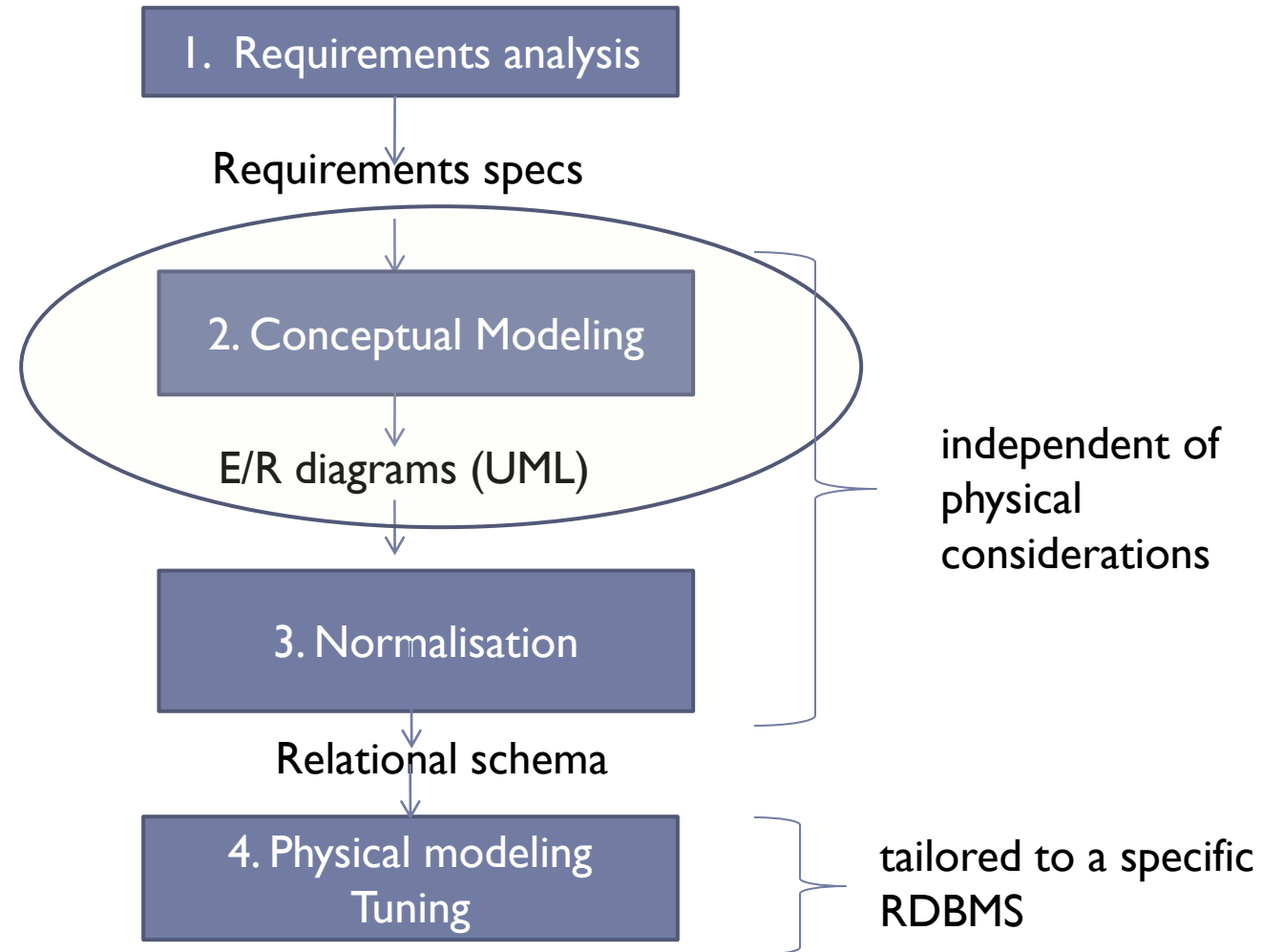
# Databases – a review on course content

---

- ▶ Basic database concepts (C1)
  - ▶ Relational Algebra (C2-C3)
  - ▶ Functional and multivalued dependencies (C4-C5)
  - ▶ Database logical design: Normalization (C6-C7)
  - ▶ Database conceptual design: Entity-Relationship Modeling (C9)
  - ▶ Database physical design (C10-C11)
  - ▶ Indexing (C12-C13)
  - ▶ Query processing (C14)
- 
- Database design and optimization

# Relational Database Design Methodology

---



# Plan for today

---

- ▶ 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:
  - ▶ Schema decomposition - normalization (Codd, '70-'74)
  - ▶ E/R data modeling (Chen,'76)
- ▶ Usually they are applied subsequently: start with E/R and continue with normalization

# Basic E/R concepts (Chen 1976)

---

## Capture the functional and informational needs of a business

### ▶ Entity

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

### ▶ Relationship

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

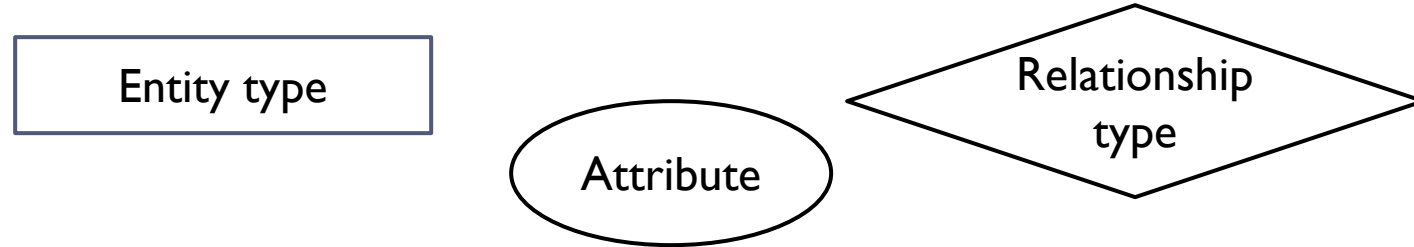
### ▶ Attributes

- ▶ For entities these are specific properties describing the independent objects
- ▶ For associations these may be
  - ▶ Properties of the involved entities
  - ▶ Specific properties of the relationship, 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:

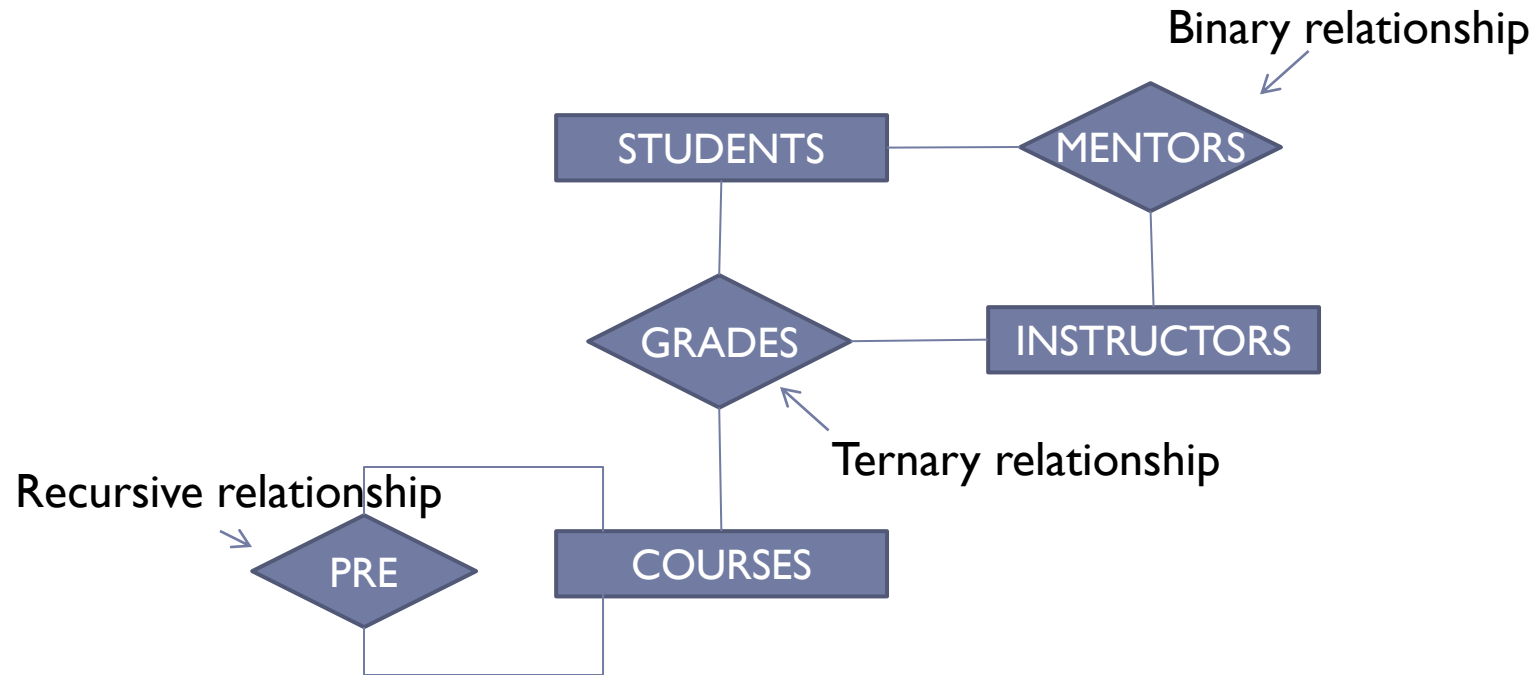


- ▶ E/R diagram = a graph where
  - ▶ Entities, relationships and attributes are vertices/nodes
  - ▶ Edges connect
    - ▶ Entity nodes with relationship nodes
    - ▶ Entity nodes with attribute nodes
    - ▶ Relationship nodes with attribute nodes

# Example

- ▶ Let's design a database to store information about

- ▶ Students
- ▶ Instructors
- ▶ Courses
- ▶ Grades
- ▶ Mentors



- ▶ Requirements:

- ▶ We can determine any student's grades for the courses he/she attended and the instructors who graded them
- ▶ We can determine a student's mentor, meaning the professor supervising him/her
- ▶ We store the prerequisites for each course, corresponding to other courses that must be graduated in advance

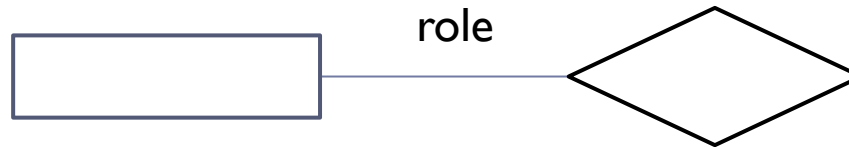


# More E/R concepts

---

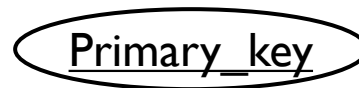
## ▶ Roles

- ▶ Explain the function of the entity in an association



## ▶ Primary key

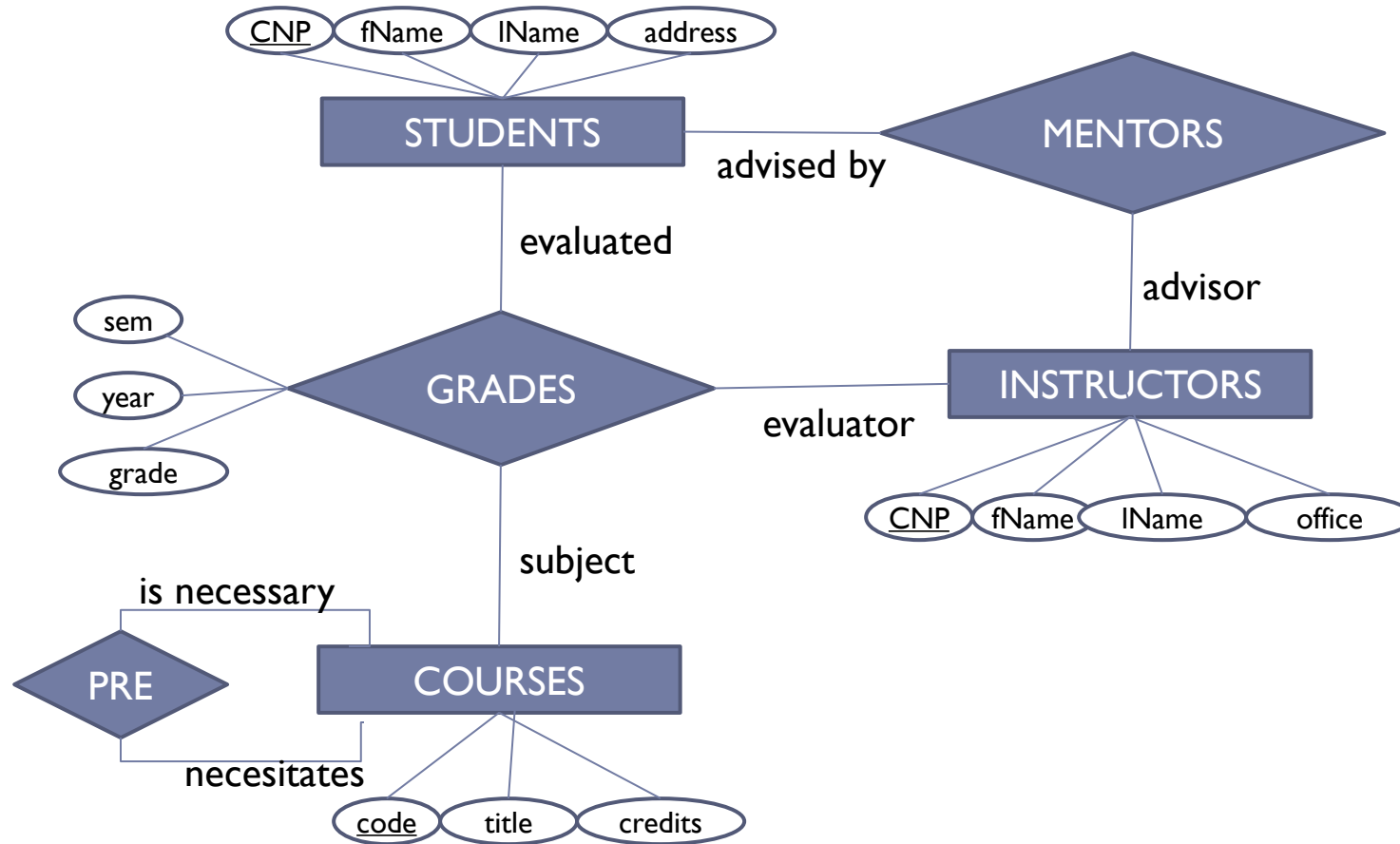
- ▶ The minimal set of attributes that is designated to uniquely identify an entity instance or a relationship instance
- ▶ It is mandatory for entity types in order to identify the entity instances that are involved in relationships



## ▶ Foreign key – defined only for relationships!

- ▶ The set of attributes that behave as primary key for the involved entities

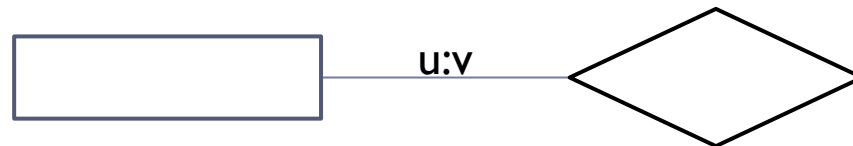
# Example



Which are the foreign keys for our relationship types?

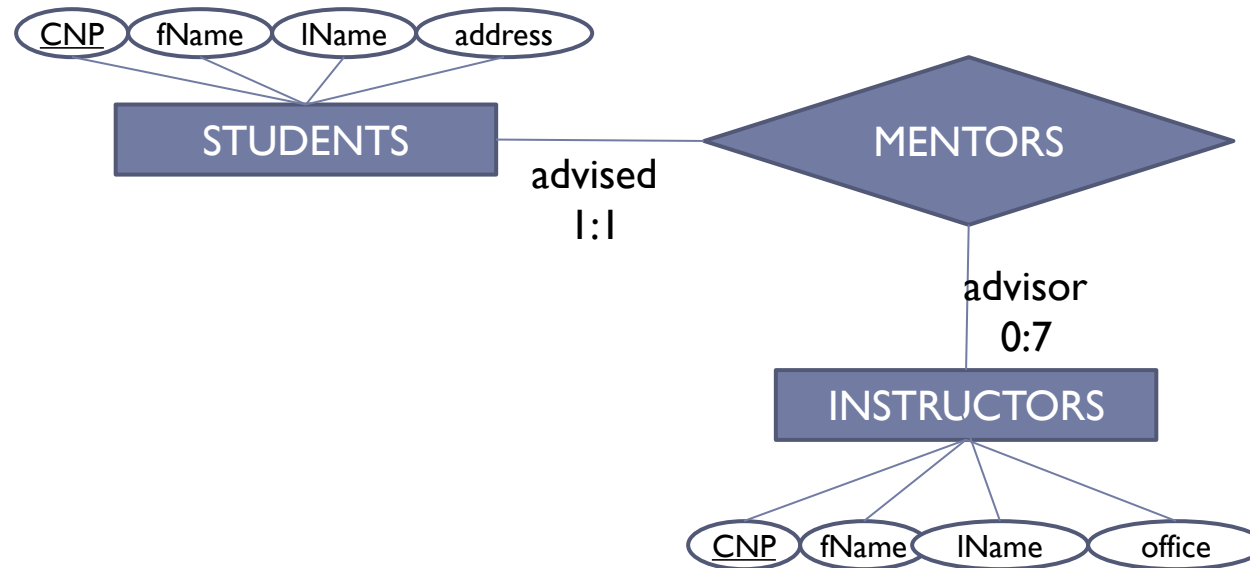
# Multiplicity constraints

- ▶ 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 among  $n$  entities 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  $R$ .

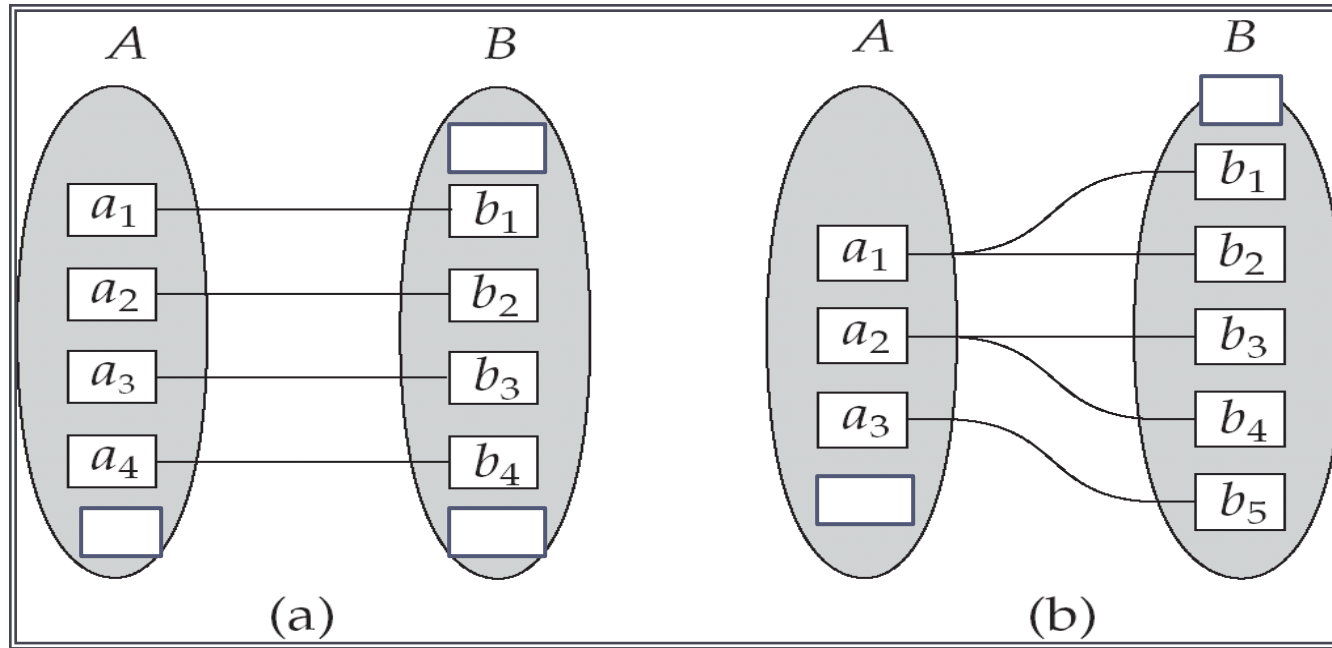


# Example

- ▶ (Students, 1, 1, Mentors)
- ▶ (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)



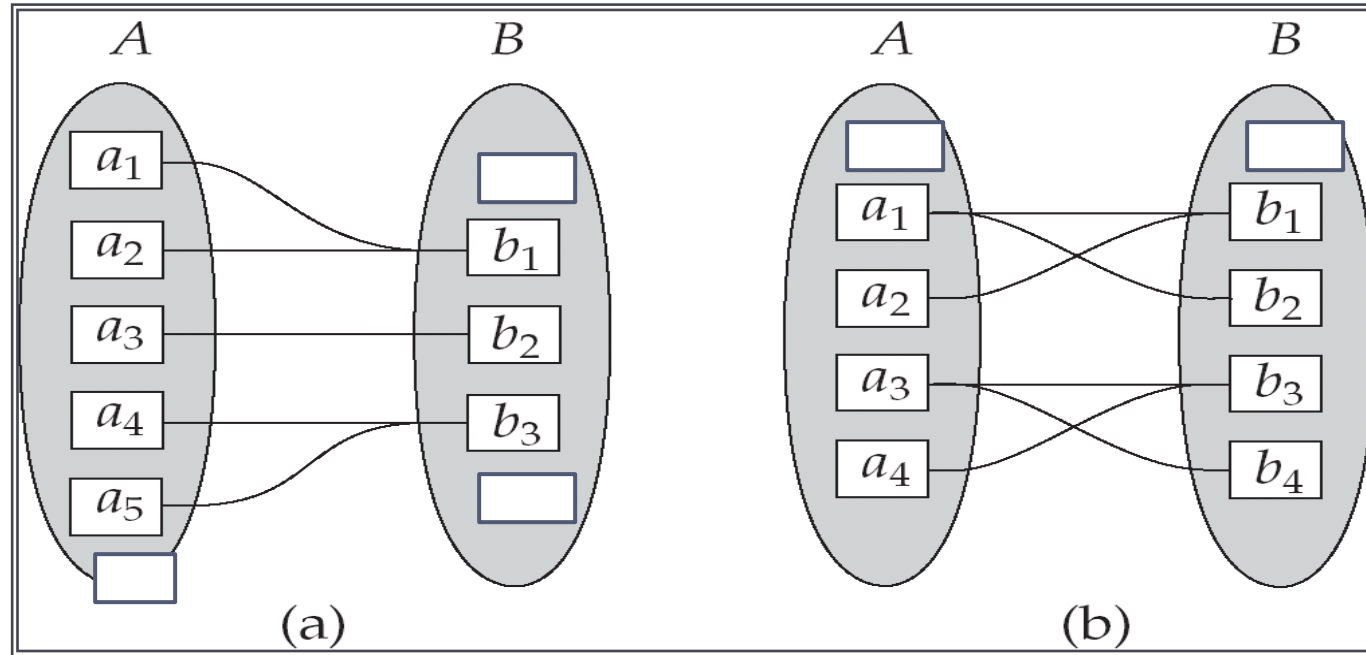
a) One-to-one relationship  
(A,0,1,R) (B,0,1,R) - incomplete



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)

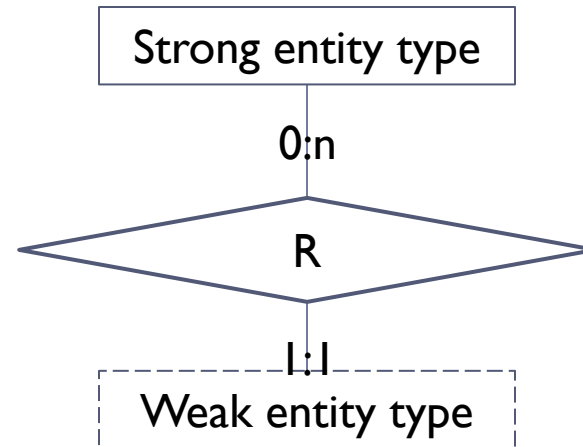


b) Many-to-many relationship  
(A,0,m,R) (B,0,n,R),  $m,n > 1$



# Weak entity type

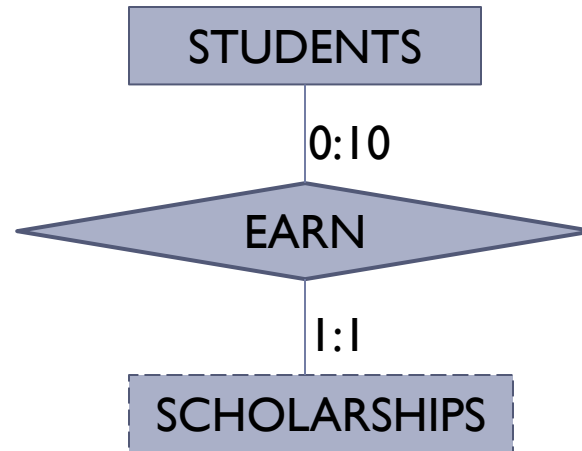
- ▶ An entity type is said to be weak if its instances depend to some extent on the existence of entity instances from other entity type (existential dependence)



- ▶ 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

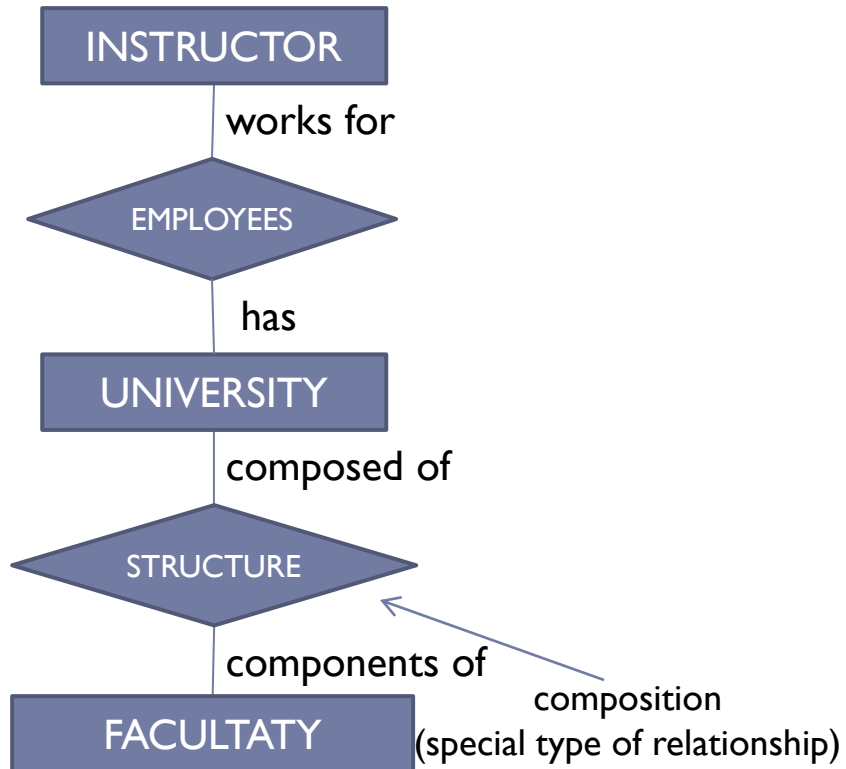
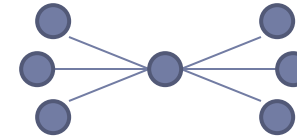
# Weak entity - example

---

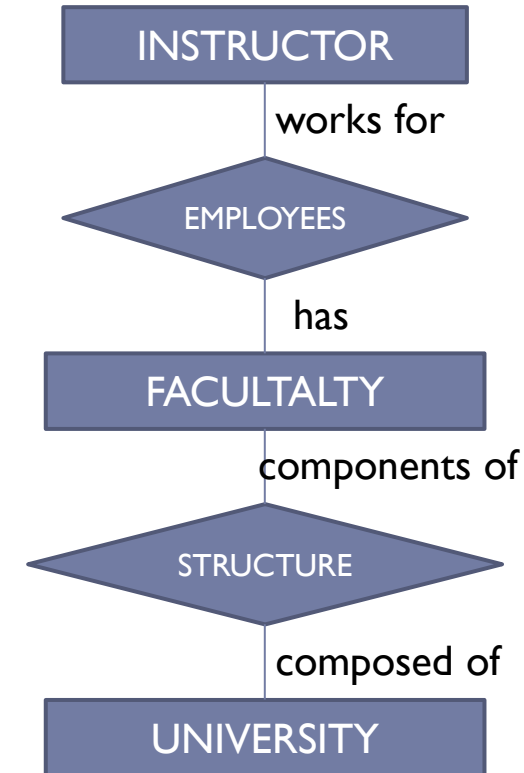




# Connection traps (Fan traps)

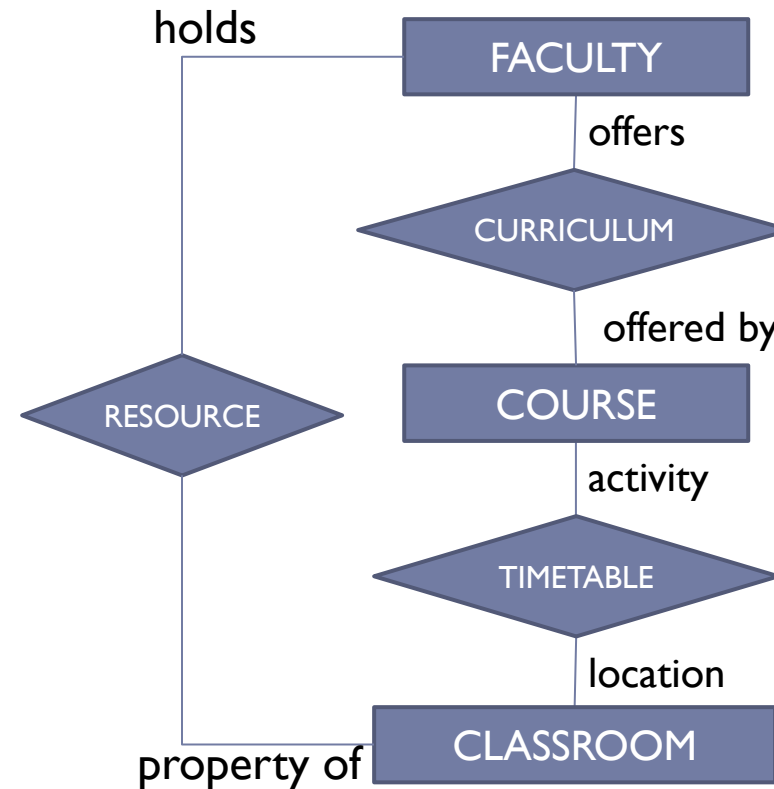
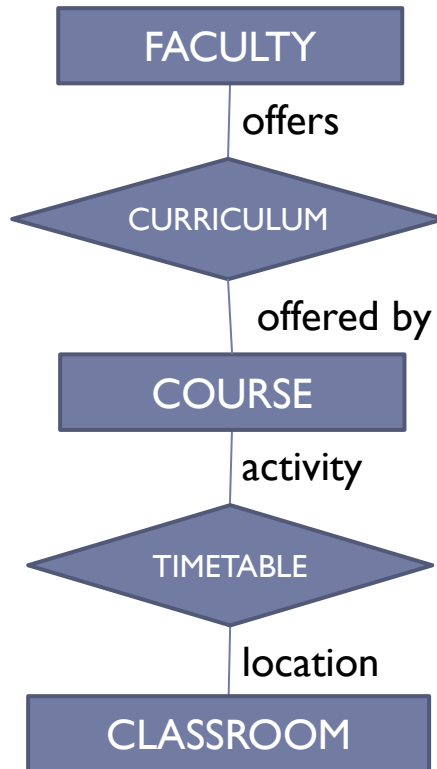
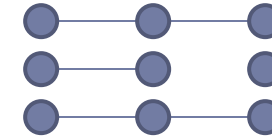


**Problem:**  
In which department/faculty profesor X is working?



**Solution:**  
Restructured model

# Connection traps (Chasm traps)



**Problem:**  
Which are the classrooms belonging to some faculty?

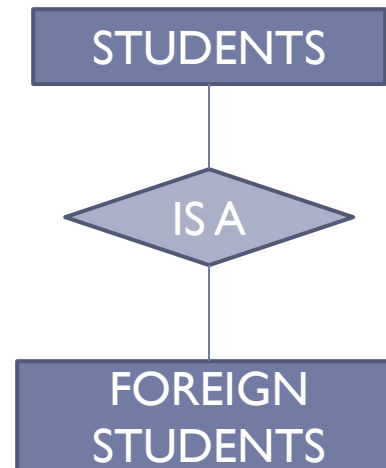
**Solution:**  
New relationships

# Enhancements of the E/R model

## Specializations

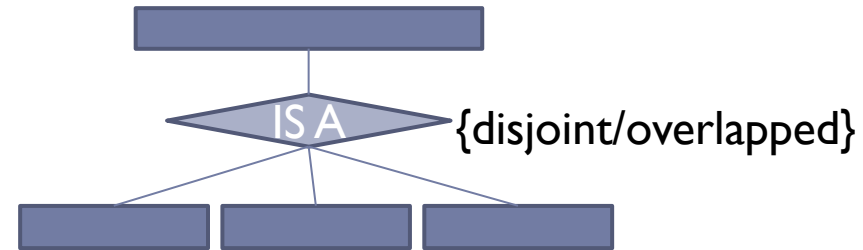
---

- ▶ 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

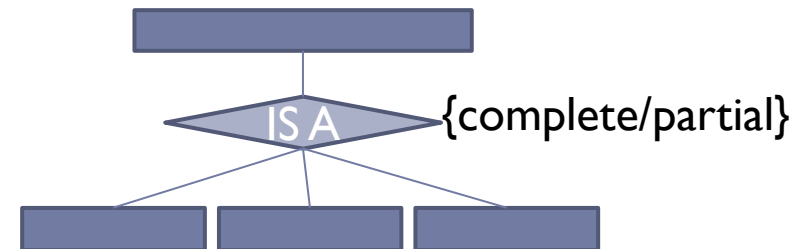


# Specialization constraints

- ▶ 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

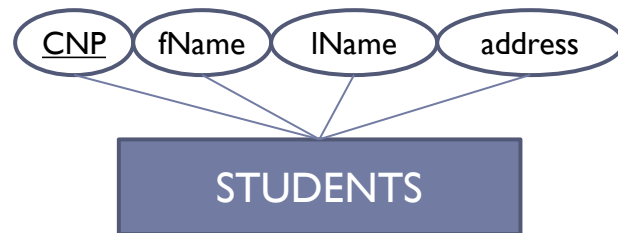
- Extensively used in software engineering
- Based on object-oriented concepts
- A communication tool with clients, in company-specific terms
- A large language from which we use a small set of elements (class diagrams) to model a database.

# 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

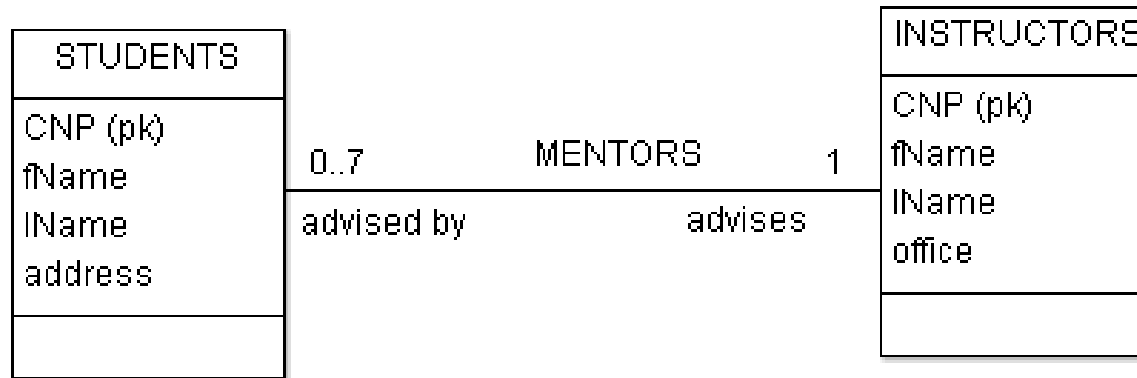
- ▶ Components: names, attributes, methods
- ▶ DB: name, attribute (primary key)



STUDENTS
CNP (pk)
fName
lName
address

# Associations

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



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



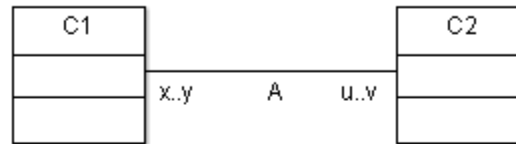
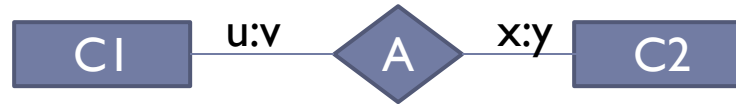
# Associations

## Multiplicity constraints

- ▶ Constraints

- ▶  $(C1, u, v, A)$

- ▶  $(C2, x, y, A)$

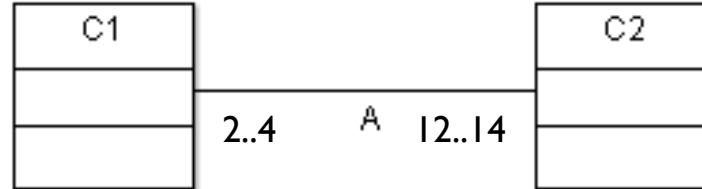


- ▶ Every entity in C1 is associated to at least  $u$  and at most  $v$  entities in C2
- ▶ Every entity in C2 is associated to at least  $x$  and at most  $y$  entities in C1

$x..y$	$u..v$	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 apply to at most 2 universities and must apply to at least one. A university may have at most 10.000 students.
2. Given the relationship

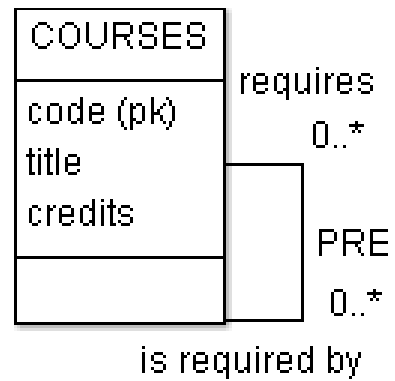
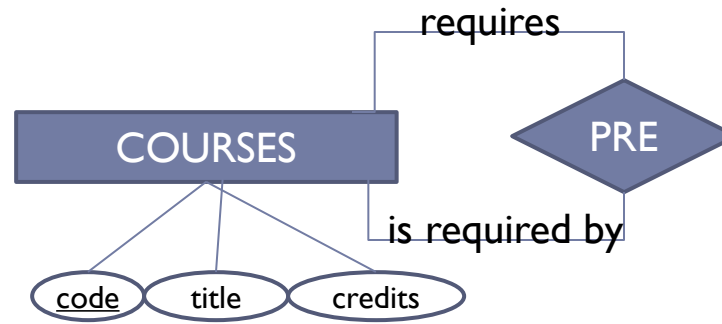


What is the minimum number of entities in C1?

What about C2?

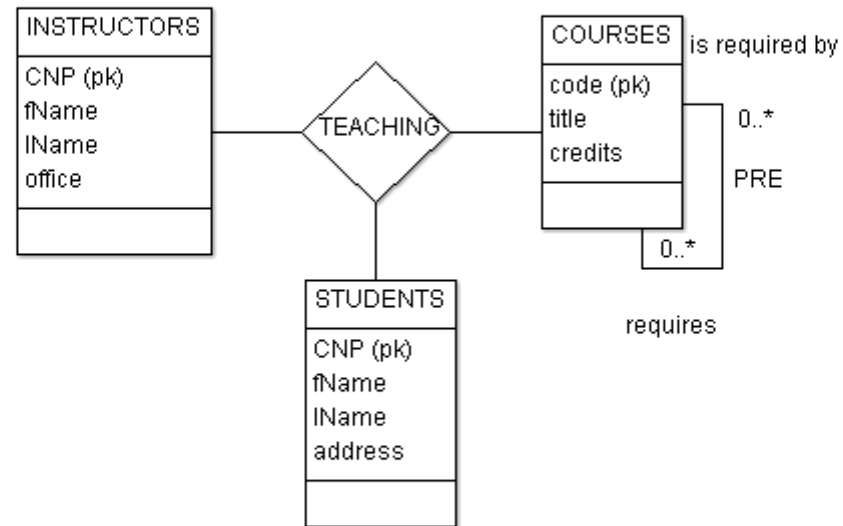
# Recursive relationships

---



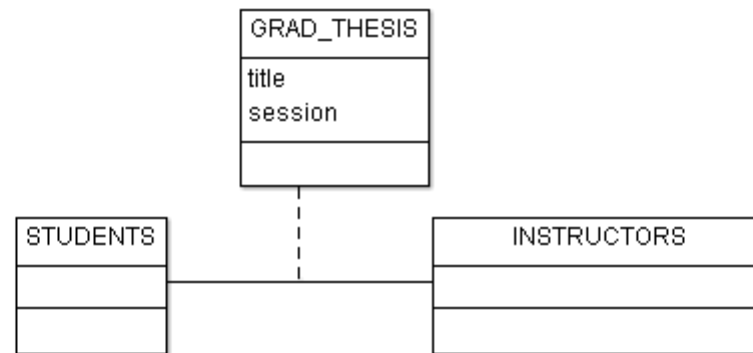
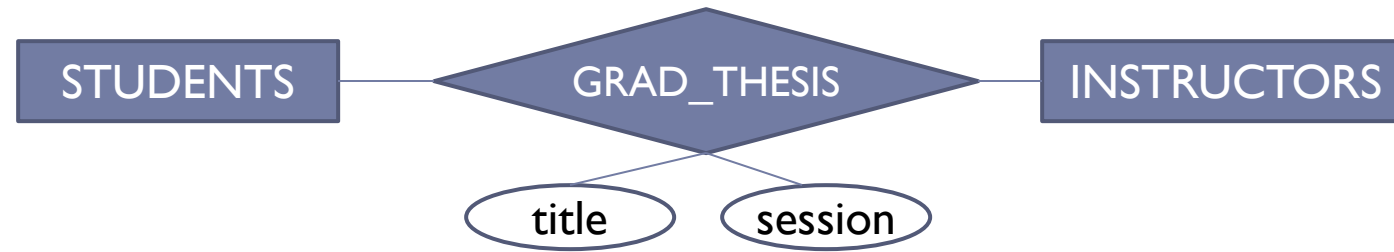
# N-ary relationships

---

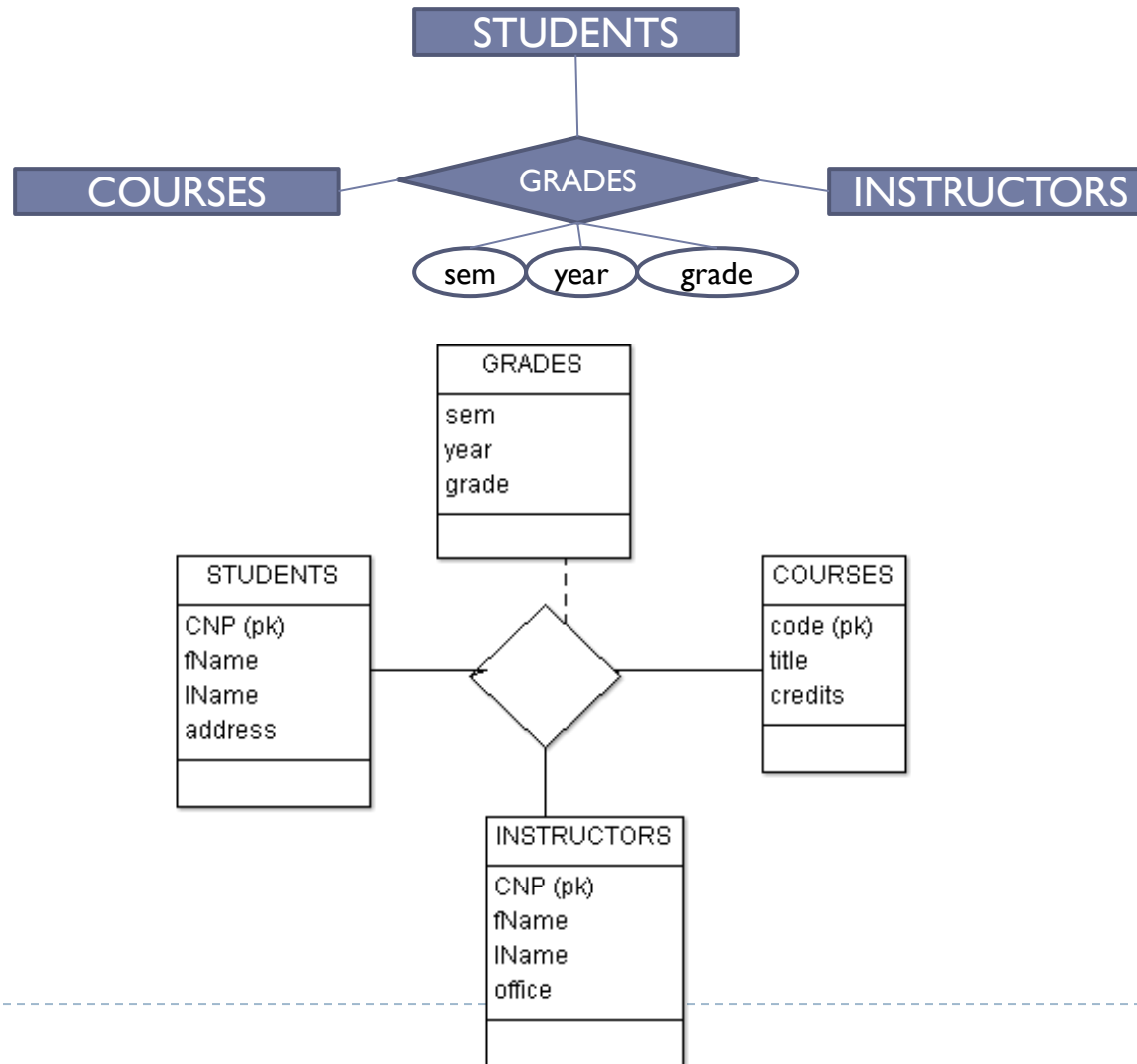


# Association classes

---

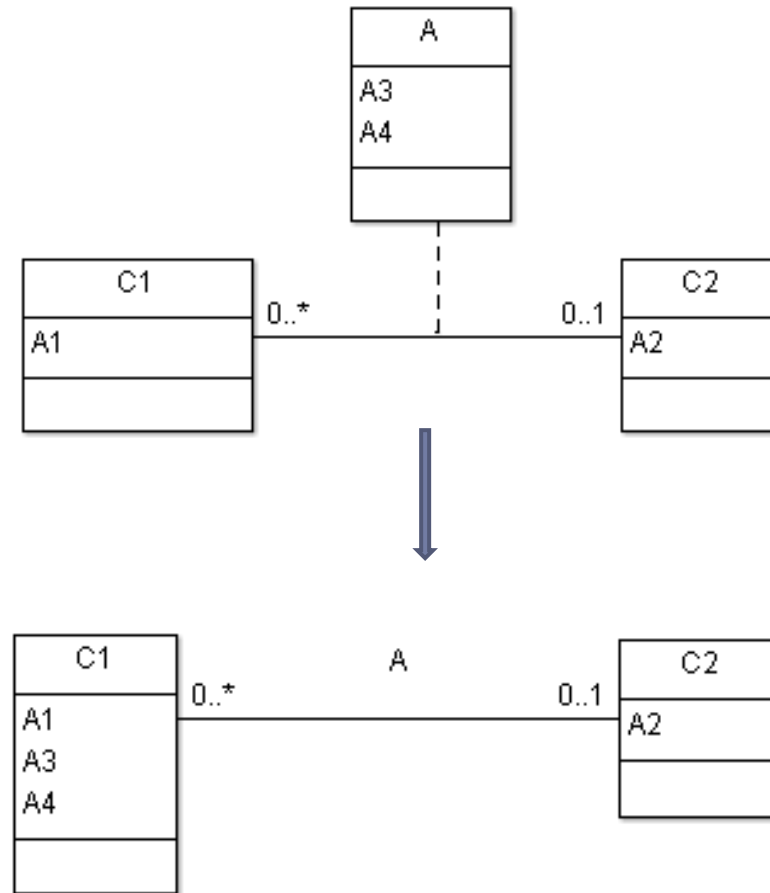


# Association classes



# Eliminating association classes

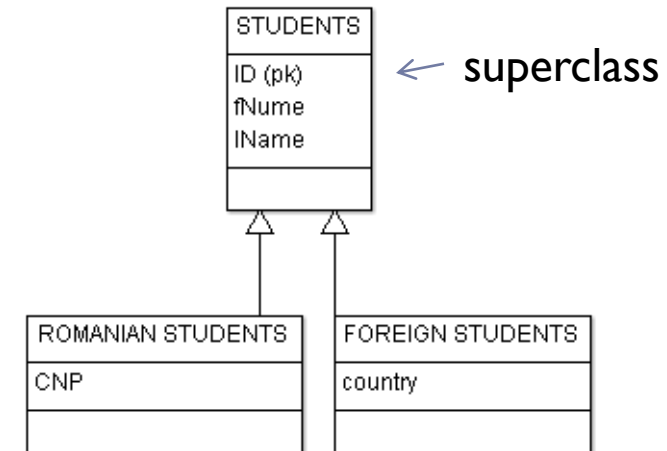
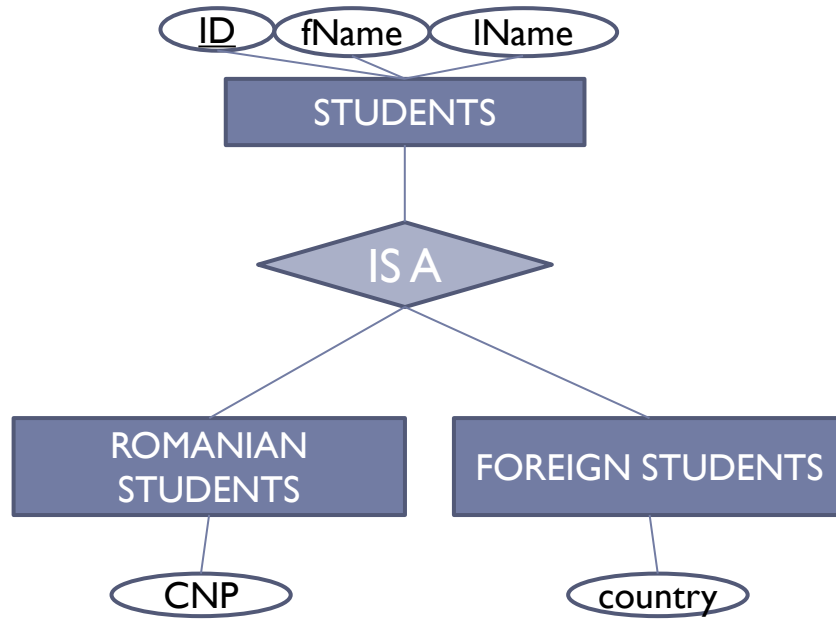
- ▶ When we have 0..1 or 1..1 multiplicity:



# Subclasses

## (1)

---



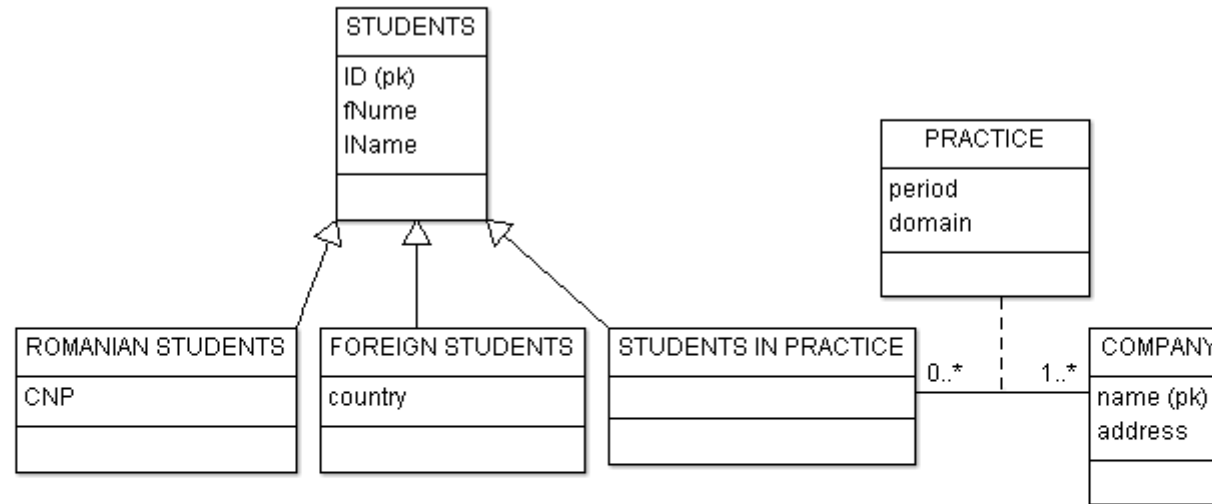
Complete, disjoint specialization



# Subclasses

## (2)

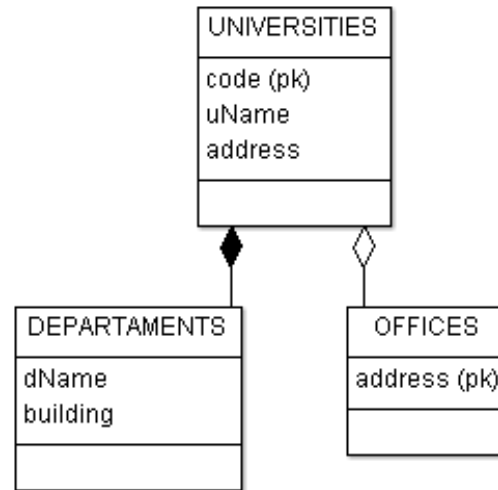
---



Complete overlapping specialization

# Compositions and aggregation

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



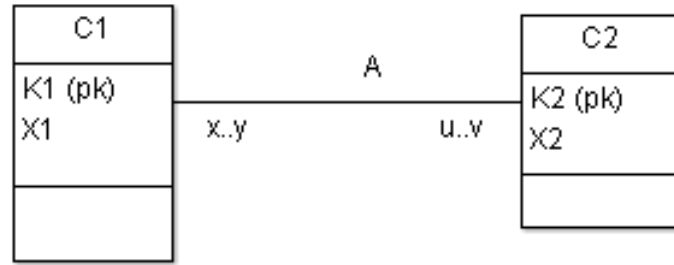
- ▶ Composition: **all** entities of a *partial* class belong to entities of the *composed* class; a partial class usually corresponds to a weak entity type (multiplicity 1..1; no primary key);
- ▶ Aggregation: **some** entities of an entity type belong to entities of another entity type (multiplicity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



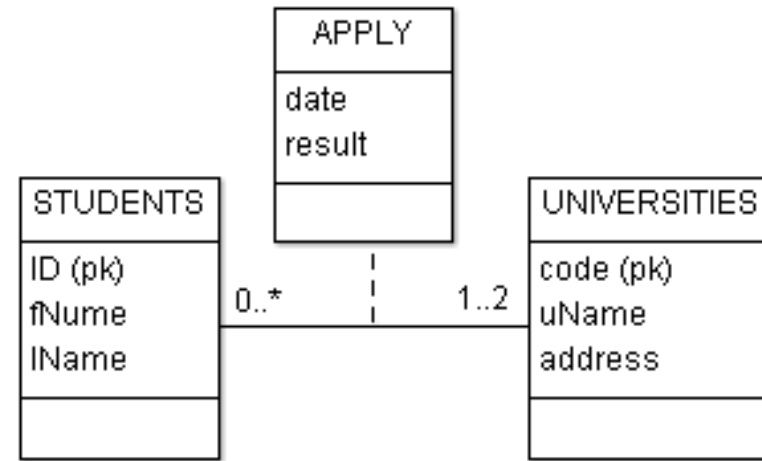
$\{C1(\underline{K1}, X1), C2(\underline{K2}, X2), A(K1, K2)\}$

- The primary key of a relationship depends on multiplicity:

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

# Quiz

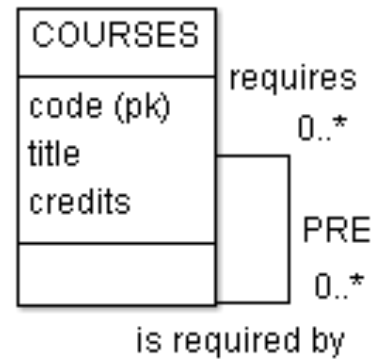
- For the E/R diagram below



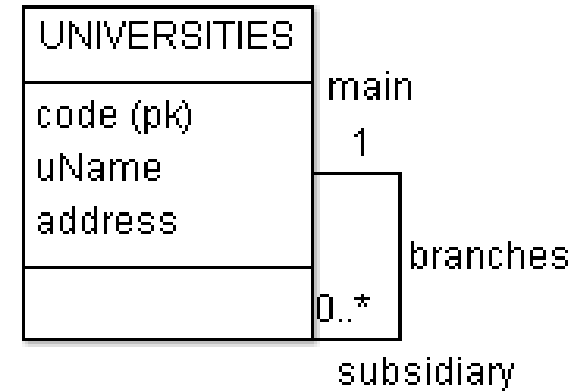
Is it possible to eliminate the relation corresponding to the relationship?

# Recursive relationships

---



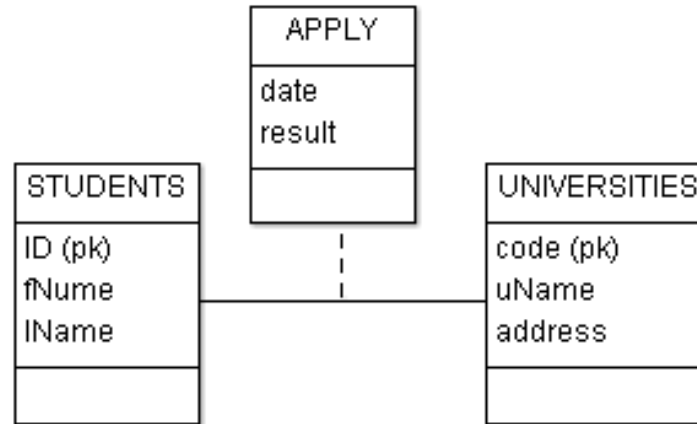
{COURSES (code, title, credits)  
PRE (code1, code2)}



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

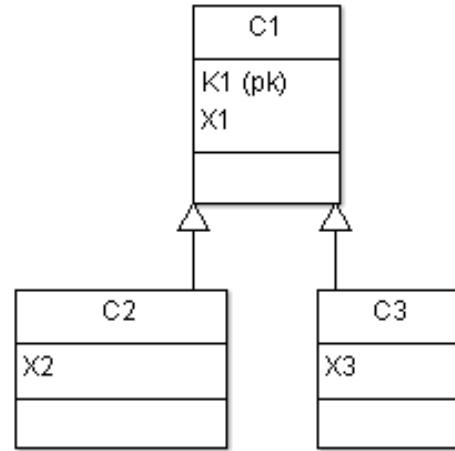
# Association classes

---



{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
  - $C1(\underline{K1}, X1), C2(\underline{K1}, X2), C3(\underline{K1}, X3)$
2. Independent superclass relation + subclass relations containing both general attributes and specialized attributes
  - $C1(\underline{K1}, X1), C2(\underline{K2}, X1, X2), C3(\underline{K2}, X1, X3)$
3. One relation containing everything
  - $C(\underline{K1}, X1, X2, X3)$

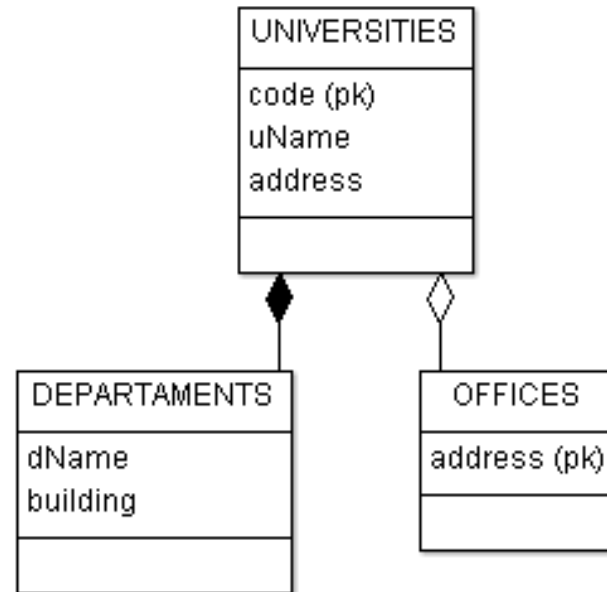


# Quiz

---

- 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?
- $n1 < n2 < n3$
  - $n1 \leq n2 \leq n3$
  - $n3 < n2 < n1$
  - $n3 \leq n2 \leq n1$

# Composition and aggregation



{ UNIVERSITIES(code, uName, address)  
DEPARTMENTS(codeU, dName, building)  
OFFICES (codeU, address)}

← 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, entity / 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*, (5<sup>th</sup> edition) Addison Wesley, 2009
- ▶ Tools:
  - ▶ <https://creately.com> (ER diagrams, UML class diagrams)
  - ▶ <http://diagramo.com/> (ER diagrams)
  - ▶ <https://argouml-tigris-org.github.io/tigris/argouml/> (UML class diagrams)