Introduction to Relational Databases

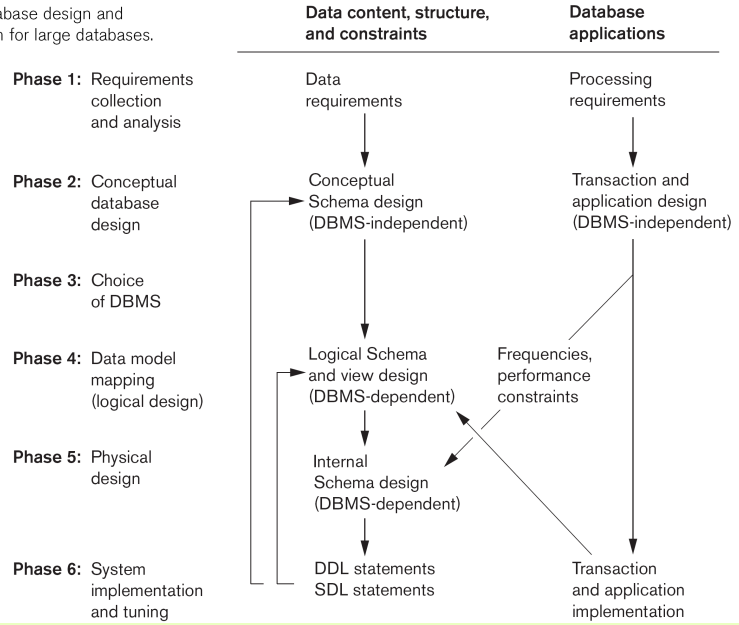
- Bachelor CS, Lille 1 University
- Nov 25th, 2015 (lecture 11/12)

- Topic: Introduction to Database Modeling with UML
  - Conceptual model,
  - Logical model

License

- Auteur original de la version italienne: inconnu
- Traduction italien → anglais: A. Bonifati (relecture: C. Kuttler)
- Corrections et modifications en 2015: C. Kuttler

Figure 10.1
Phases of database design and implementation for large databases.



From : Elmasri & Navathe, Fundamentals of DB systems

Prologue: a Class Diagram in UML

- Today's lecture: where are we?
  - we need to do an analysis of the purpose of a database and the objectives of a database modeling
- Goals of the conceptual model
  - focus on the concepts of the real world
  - avoid to think about the implementation of those concepts (as objects in a programming language or relations in a database)

A Class Diagram in UML

- With UML one can stay with an high level of abstraction
  - no details for the classes
  - no methods for the classes
  - no need to have all attributes in the relations
  - no need to have the types of the attributes
- Whereas all the above is relevant for the logical model (i.e. in SQL creation scripts)

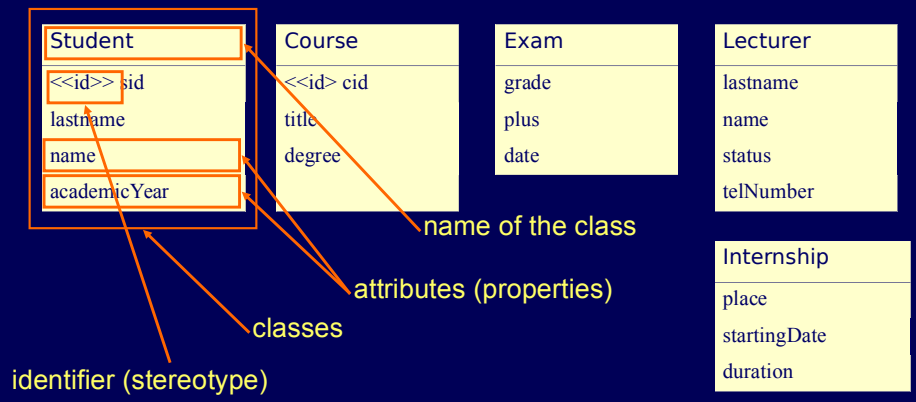
Example: a Database of information about courses

- for administrative staff of a university, for courses in Computer Science
- Manage the data for Bachelor program. For each student, one must register the exams that he/she has passed.
- Also include data about courses and exams.
- For each course one has to keep track of the lecturer, that can be one or more;
- For each lecturer one has to keep track of the phone numbers;
- For the students, it is necessary to record the professor who is responsible of the final internship and also the information about the internship. Only students enrolled into the third year can have such an advisor for the internship period.

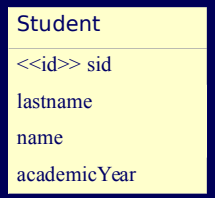
Class

- Abstract "concepts" from real world
  - facts, people, things, that exist autonomously
  - example: student, course, exam, lecturer
- Instance of a class
  - set of objects representing the reality
- Classes have attributes
  - attributes are properties of a class that are relevant for the application
  - stereotype <<id>> for identifiers (UML syntax)
  - stereotype: UML notation for constructs that have a precise role

Classes: examples



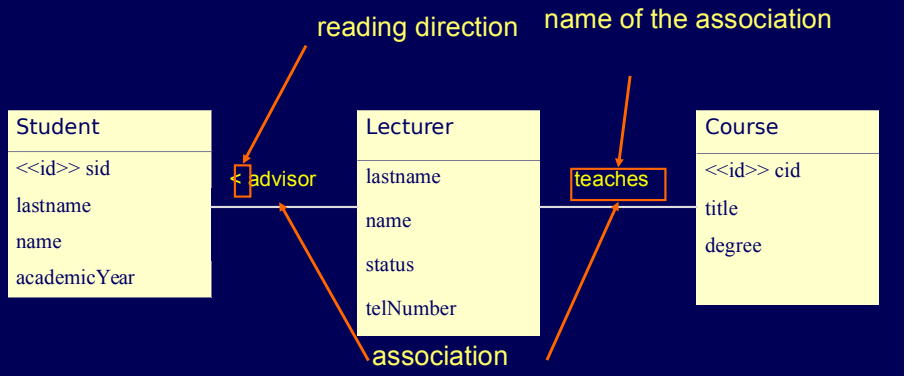
Class student: instance



Associations

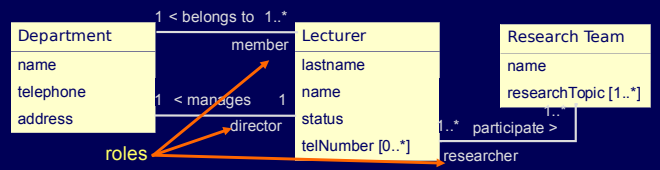
- Relations between classes
  - logical relationship relevant for the application
  - ex: a student has passed an exam
  - ex: a lecturer teaches a course
- Instance of an association
  - set of arrows among instances of the involved classes

Associations: names and reading direction



Association: roles

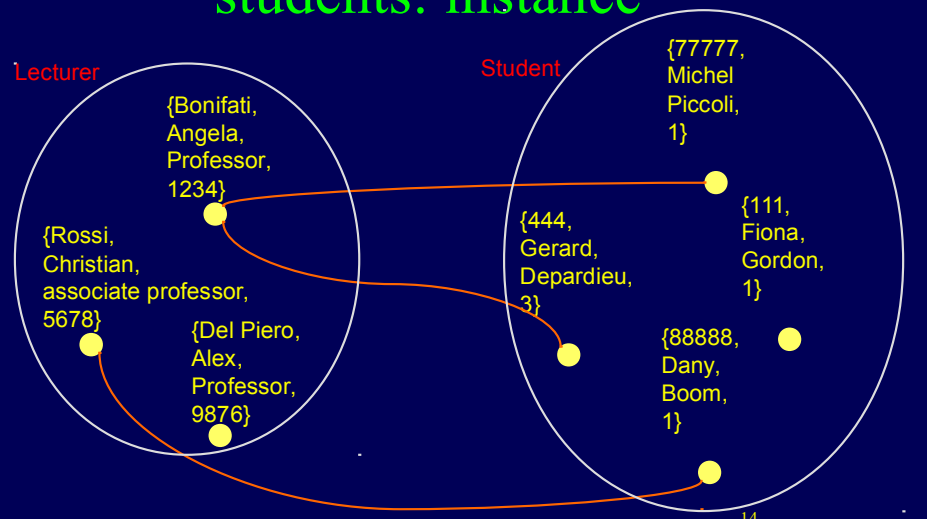
- description of the role of a class in the relationship



Association

- Typically
  - binary associations (between 2 distinct classes)
- But there may exist
  - recursive associations: associations between objects of the same class
  - n-ary associations (seldom used): associations that involve objects of multiple classes (3 or more)

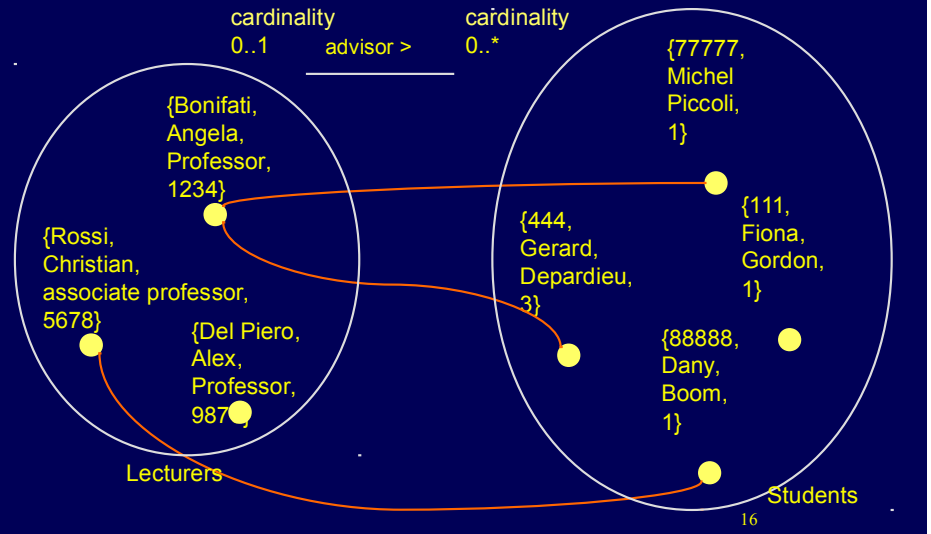
Association between lecturer and students: instance



Cardinalities of associations

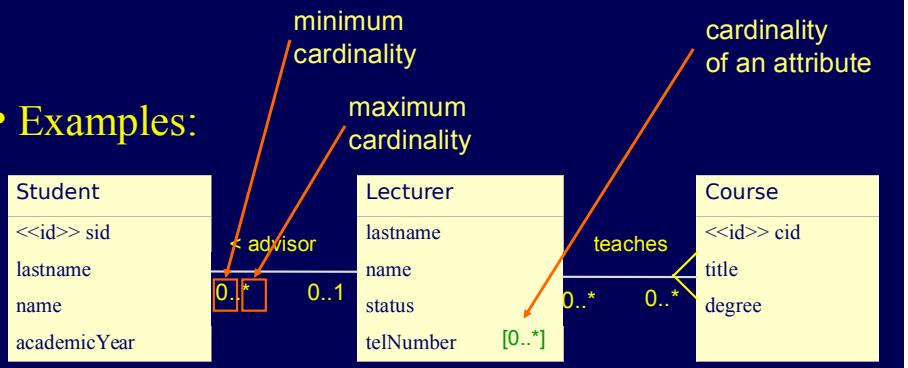
- Constraints
  - Expressed for both classes within association
  - Min/Max number of objects of a class that can be associated to the objects of the other class
- Minimum cardinality
  - typically 0 or 1
- Maximum cardinality
  - typically 1 or * (n) (but also 3 or 5)

Cardinalities of association



Cardinalities in the conceptual model

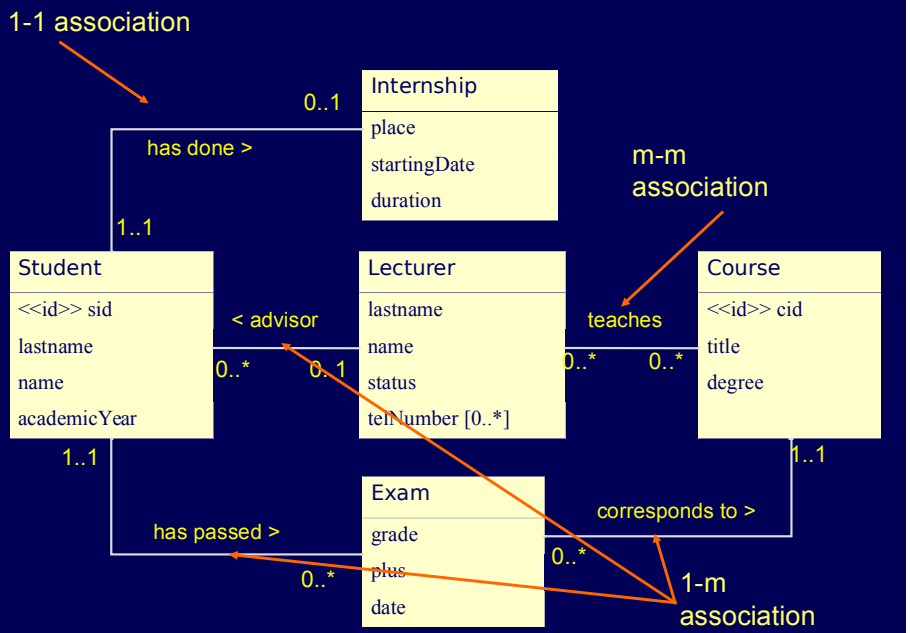
- Examples:



- A lecturer can be the advisor of 0 or many students.
- A student has at most 1 advisor.
- This interpretation in UML is the opposite of that in MERISE.

1-1 , 1-m , or m-m?

- Combinations of max cardinalities
  - 1 to 1: max cardinality equal to 1 on both sides of association
  - 1 to many: max cardinality equal to 1 on one side and * on the other side
  - many to many: max cardinality equal to * on both sides

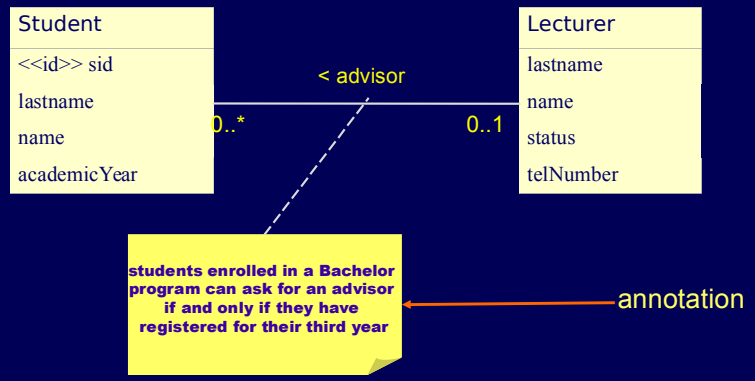


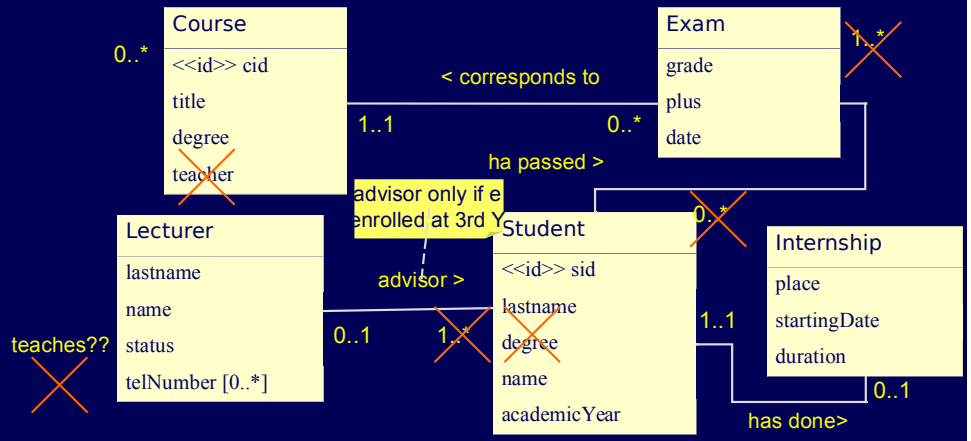
The complete diagram

- The complete diagram
  - contains all the revised constructs
- It is possible to add annotations
  - useful to add comments
  - and to express in UML constraints that are otherwise not expressible, such as: the students enrolled in a Bachelor program can ask for an advisor if and only if they have registered for their third year!

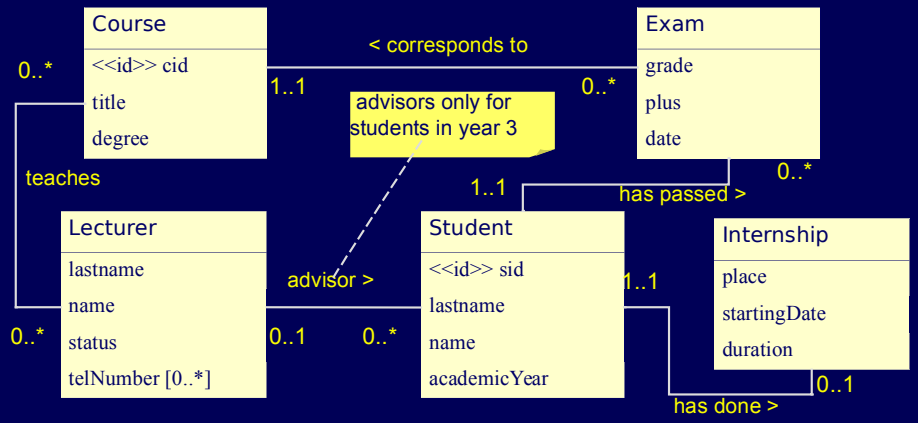
Towards the complete diagram

- Examples:





A diagram with mistakes (done on purpose :-)



Conceptual model (final version)

The interpretation of cardinalities in UML is just the opposite of the MERISE convention.

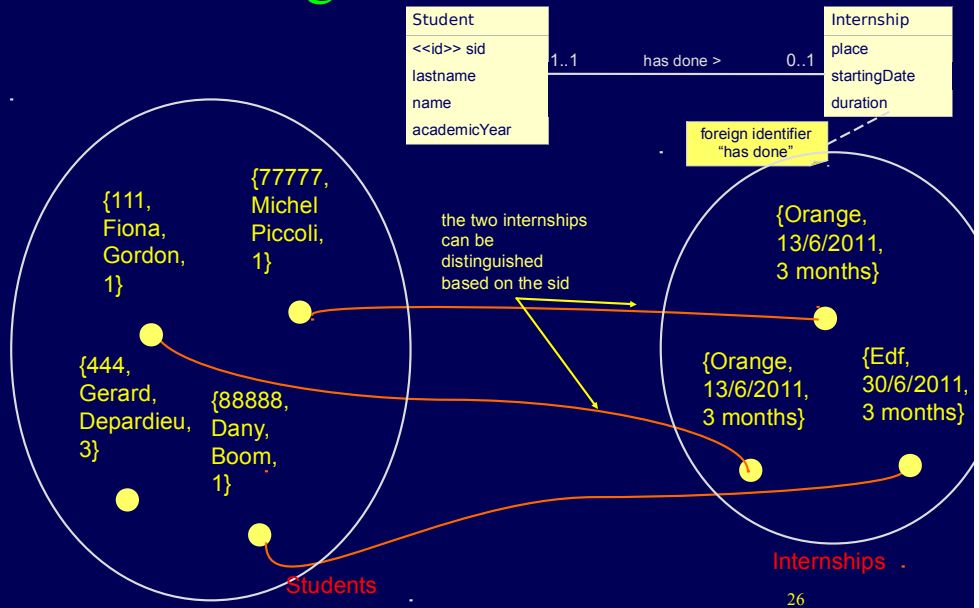
Identifiers

- Within classes
  - not all classes have explicit identifiers
  - but the objects of a class have to be distinct (ex: two apples within the class “Apples”)
  - in a sw program, all objects have an implicit identifier (hidden): OID
  - explicit identifiers (e.g. sid, ssn, plate nr.) exist for a catalog
  - they may become relevant inside a database model

Foreign identifiers of classes

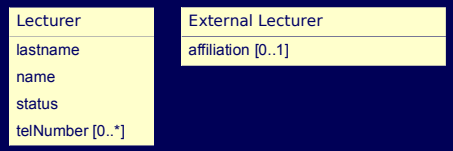
- Some objects are identified thanks to other objects to which they are associated
  - example: a Internship is identified by means of the student ID (the student who has done it)
  - to identify a Internship it is necessary to traverse the edge that represents the association
  - the cardinality on the other side has to be 1
  - Represented as annotation in conceptual model

Foreign identifiers



Cardinalities of attributes

- number of possible values of an attribute
- min and max
- standard cardinality1..1
- minimum cardinality 0: optional attribute
- maximum cardinality *: multivalued attribute



Summary of Part 1

- Introduction
  - Requirement Analysis
- Class diagram in UML
  - Class
  - Association
  - Cardinality
- The complete diagram
- Foreign identifiers, cardinalities of associations and attributes

Part 2

- Bachelor CS, Lille 1 University
- Dec 2nd, 2015 (lecture 12/12)
- Topic: Introduction to Database Modeling (aka Logical Modeling, transformation from Conceptual Modeling to Logical Modeling)

Prologue: Logical Model

- Today's lecture: where are we?
  - we have completed the conceptual model; we want to transform it into a logical model
- Goals of the logical model
  - starting from a more abstract (conceptual) model that was conceived before
  - gets closer to the implementation of the database

Summary

- Introduction
  - The design process of a database
- An Algorithm for Logical Modeling
  - Translation of the classes
  - Translation of m-m associations
  - Translation of 1-m associations
  - Translation of 1-1 associations
- The resulting database

The design process of a database

- Starting point
  - the conceptual design
- Logical design
  - derive the logical schema from the conceptual model and, eventually, external schemas (views on the database)
  - standard algorithm
- Physical design
  - verify and, possibly, optimize the logical schéma
  - too difficult to standardize

Translation Algorithm

- For now
  - we will study a simplified version
- Steps of the algorithm
  - initial translation of the classes
  - translation of m-m associations
  - translation of 1-m associations
  - translation of 1-1 associations

Logical model

- Graphical notations
  - suitable stereotype of UML
  - For a class or table, encode
    - its attributes
    - the primary keys
    - the possible foreign keys
    - types

Logical model

- Can be implemented

```
CREATE TABLE Student (
   sid integer PRIMARY KEY,
   lastname char(20),
   name char(20),
   year integer,
   degree char(20),
   advisor char(4) REFERENCES Lecturer(pid));
```

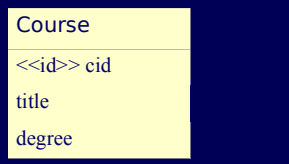
Student	T
sid INTEGER	PK
lastname CHAR(20)	
name CHAR(20)	
year INTEGER	
degree CHAR(20)	
advisor CHAR(4)	FK

Lecturer	T
pid CHAR(4)	PK
…	

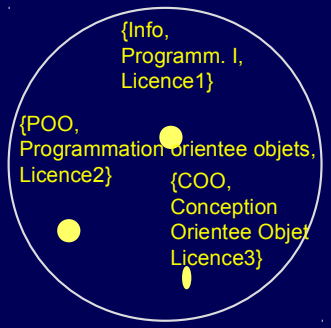
Translation of classes

- Idea
  - each class is converted into a table
  - initially, the same set of attributes
  - other attributes can be added afterwards
- In the logical model, we MUST identify
  - types of attributes
  - primary keys
  - foreign keys
- Primary key
  - it has to be easy to use and concise
  - explicit identifier often exists (sid for Student, cid for Course)
  - otherwise, add a synthetic identifier

Courses

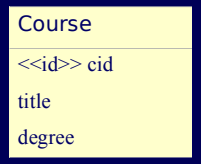


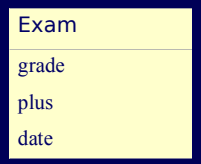
Course	T
cid CHAR(3)	PK
title CHAR(20)	
degree CHAR(20)	



cid	title	degree	…
Info	Programmation	Licence1	…
POO	Programmation orientee objet	Licence2	…
COO	Conception orientee objet	Licence3	…

Classes : conceptual vs logical model





Course	T
cid CHAR(3)	PK
title CHAR(20)	
degree CHAR(20)	

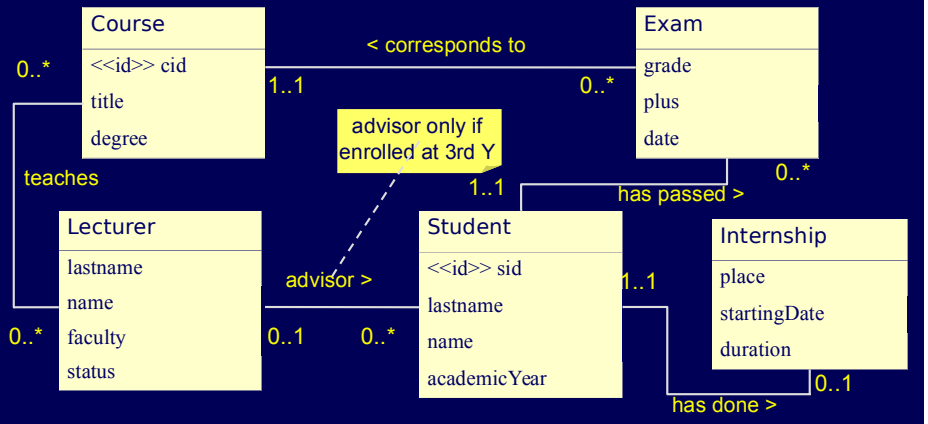
explicit identifier

Exam	T
eid CHAR(5)	PK
grade INTEGER	
plus BOOL	
date DATE	

synthetic identifier (same for internship)

types

How to translate associations ?

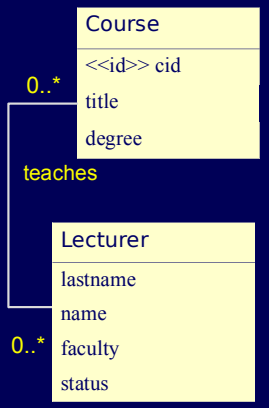


Distinct methods for 1-1, 1-m, m-m

Translation of m-m associations

- For each m-m association
  - Create a new table
  - with foreign keys to the classes connected by the association
  - with the attributes of the association, if any
  - the primary key of the association is composed of both foreign keys of the connecting classes (each instance of the association is identified by a pair of elements)

Translation of m-m associations

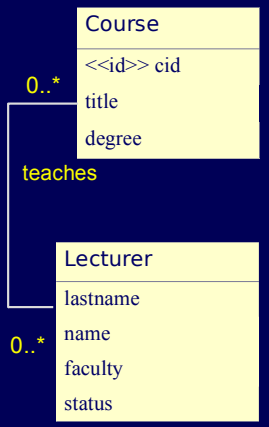


Course	T
cid CHAR(3)	PK
title CHAR(20)	
degree CHAR(20)	

Teaching	T
cid CHAR(3)	PK, FK
pid CHAR(4)	PK, FK

Lecturer	T
pid CHAR(4)	PK
lastname CHAR(20)	
name CHAR(20)	
…	…

Translation of the m-m associations



cid	title	degree
PR1	Programming I	Licence
ASD	Algorithms and DS	Licence
INFT	Theoretical CS	Master

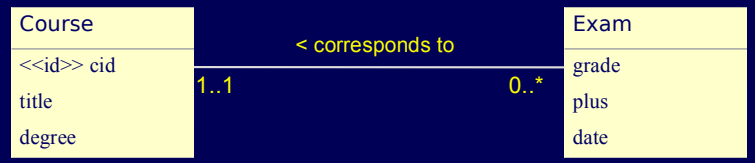
lecturer	course
FT	PR1
CV	ASD
FT	ASD
…	…

pid	lastname	name	...
FT	Totti	Francesco	…
CV	Vieri	Christian	…
ADP	Del Piero	Alessandro	…

Translation of the 1-m associations

- In principle
  - could introduce new tables (as for m-m case)
  - it would be inefficient: we would need to join the obtained tables
- Generate foreign keys
  - In the class with cardinality 1, add a foreign key, that references to the side of the association with cardinality m

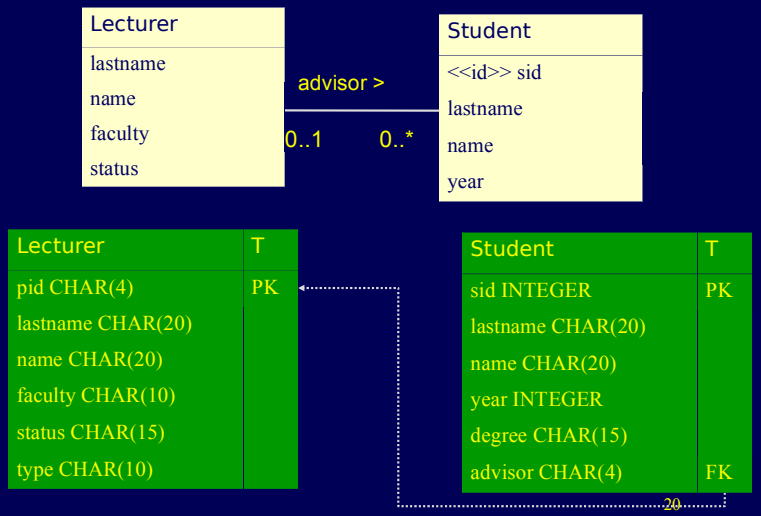
1-m association : generate foreign key in logical model



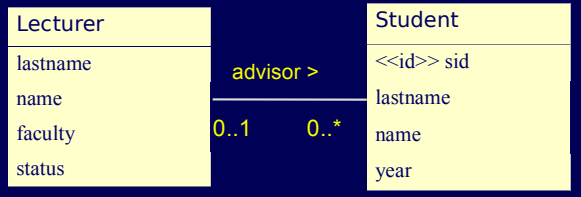
Course	T
cid CHAR(3)	PK
title CHAR(20)	
degree CHAR(20)	

Exam	T
eid CHAR(5)	PK
grade INTEGER	
plus BOOL	
date DATE	
course CHAR(3)	FK

Another example



Translation of the 1-m associations



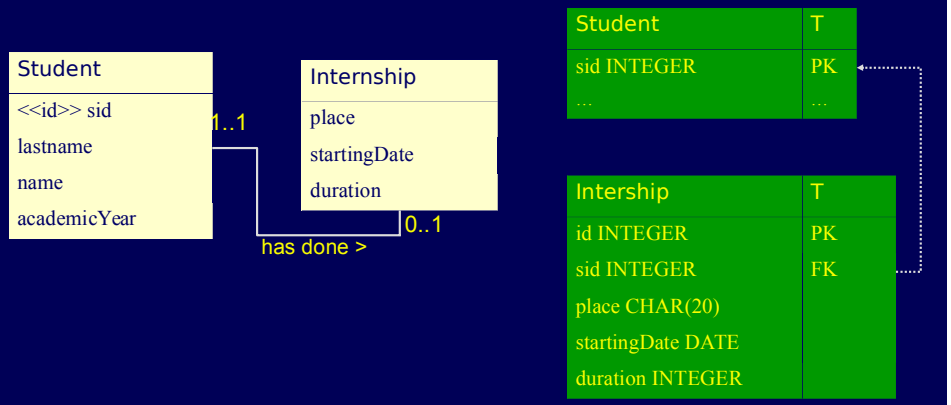
pid	lastname	name	...
FT	Totti	Francesco	…
CV	Vieri	Christian	…
ADP	Del Piero	Alessandro	…

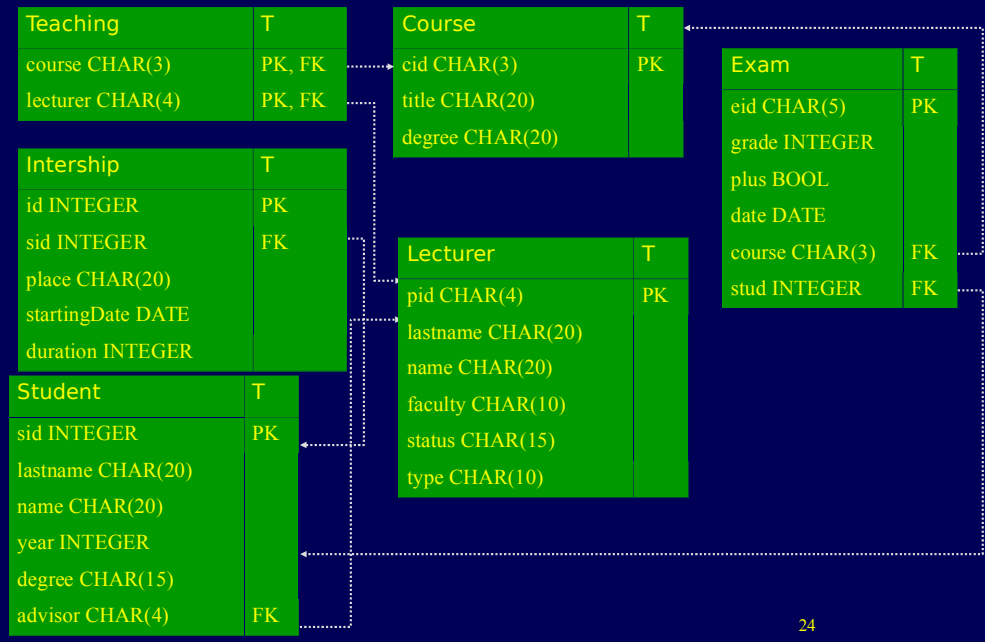
sid	lastname	name	...	advisor
111	Rossi	Mario	…	null
222	Neri	Paolo	…	null
333	Rossi	Maria	…	null
444	Pinco	Palla	…	FT
77777	Bruno	Pasquale	…	FT
88888	Pinco	Pietro	…	CV

Translation of the 1-1associations

- Similar discussion as previously
  - I can choose where to put the foreign key
  - it is preferable to use as foreign key the primary key of the class which is on the side of the association with minimum cardinality equal to 1
- Example
  - Student has done an Internship

Translation of the 1-1associations





The final database schema in SQL

```
CREATE TABLE Lecturer (
   pid char(4) PRIMARY KEY,
   lastname varchar(20) NOT NULL,
   name varchar(20) NOT NULL,
   status char(15),
   faculty char(10)
);


CREATE TABLE Student (
   sid integer PRIMARY KEY,
   lastname varchar(20) NOT NULL,
   name varchar(20) NOT NULL,
   degree char(20),
   year integer,
   advisor char(4) REFERENCES Lecturer(pid)
);
```


The final database schema in SQL

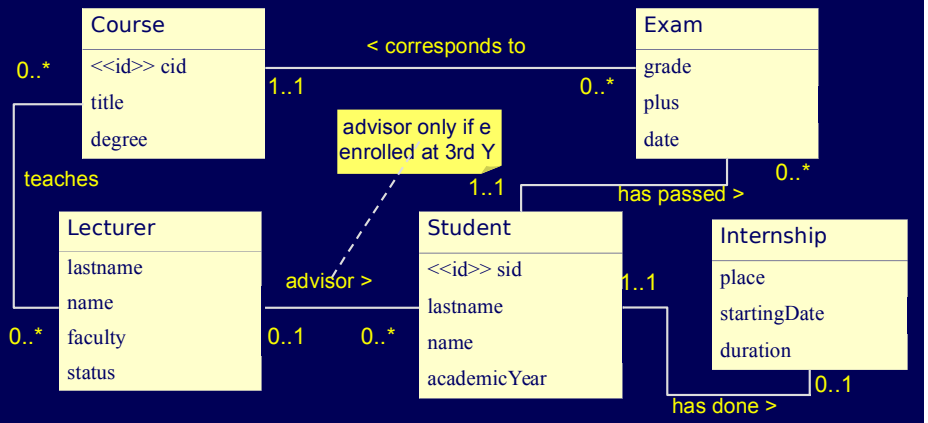
```
CREATE TABLE Course (
   cid char(3) PRIMARY KEY,
   title varchar(20) NOT NULL,
   degree char(20)
);

CREATE TABLE Exam (
    eid char(5) PRIMARY KEY,
    student integer NOT NULL REFERENCES Student(sid)
      ON DELETE cascade ON UPDATE cascade,
    course char(3) NOT NULL REFERENCES Course(cid),
    grade integer,
    plus bool,
    date date
);
```


The final database schema in SQL

```
CREATE TABLE Internship (
   id INTEGER PRIMARY KEY,
   student integer REFERENCES Student(sid),
   place char(20) NOT NULL,
   startingDate date,
   duration integer
);


CREATE TABLE Teaching (
   lecturer char(4) REFERENCES Lecturer(pid),
   course char(3) REFERENCES Course(cid),
   PRIMARY KEY (lecturer, course)
);
```



The Complete Algorithm

- I step
  - initial translation of classes
- II step
  - translation of hierarchies (omitted in our course)
- III step
  - translation of multi-valued attributes
- IV step
  - translation of m-m associations

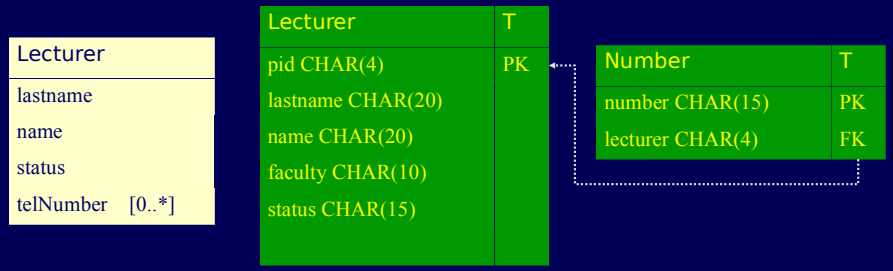
The Complete Algorithm

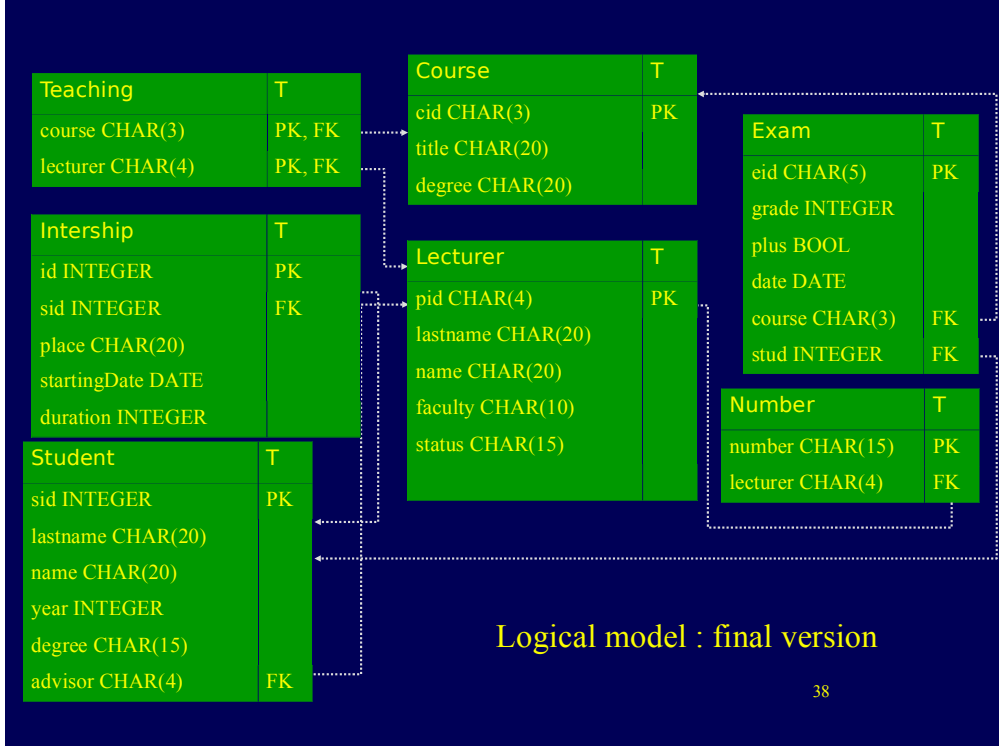
- V step
  - translation of 1-m associations
- VI step
  - translation of 1-1 associations
- VII step
  - introduction of further constraints (omitted)
- VIII step
  - design of external schemas (omitted)

III Step: Multi-valued attributes

- Multi-valued attributes
  - do not have ‘atomic’ values
  - require extra tables
- A table for a multi-valued attribute
  - has an attribute that represents the values
  - a foreign key to the class to which the attribute belongs to

III Step: Multi-valued Attributes





Logical model : final version

Summary of part 2

- Introduction
  - The design process of a database
- An algorithm for the design of the logical model
  - Translation of classes
  - Translation of multi-valued attributes
  - Translation of m-m associations
  - Translation of 1-m associations
  - Translation of 1-1 associations
- The resulting database