Relational translation - 2

Knowledge Objectives

1. Explain the two possible implementations of symmetric reflexive associations in a RDBMS
2. Remember where to place the attributes of the UML associations when they are implemented on a RDBMS, depending on their multiplicity
3. Distinguish associative classes that appear just due to UML syntax constraints from those truly associative classes

Understanding Objectives

1. Translate from a UML class diagram (with around 10 classes, some maybe associative classes, and related by associations, generalizations and aggregations) into an SQL schema

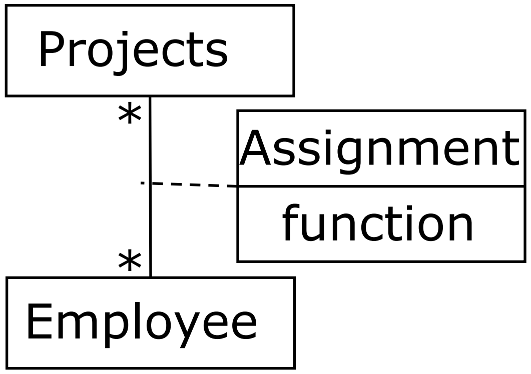
Application Objectives

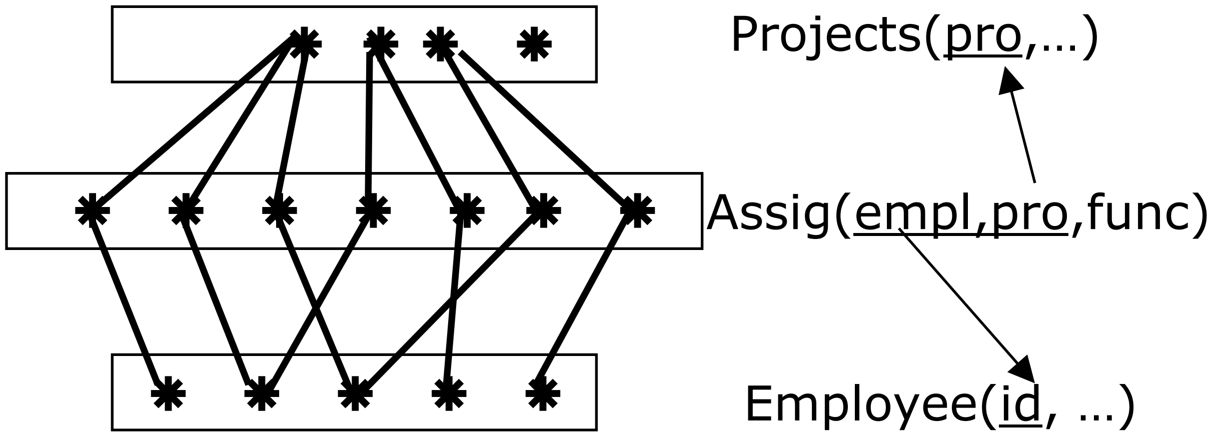
1. Choose and justify the best option to translate from a UML class diagram (with less than 10 classes, some maybe associative classes, related by associations, generalizations and aggregations) into an SQL schema, given the statistics of participation of the instances in the relationships and the queries
2. Given a multivalued attribute and an explanation of its usage, choose and justify the best option to implement it in a RDBMS

Multiplicities

1. Maximum multiplicity:
   - Each one, how many at most? Give raise to:
     - *-*
     - 1-*
     - 1-1
2. Minimum multiplicity:
   - Could zeros exist (<u>**possible**</u> no participation of an instance in the relationship)?
     - Above cases split into subcases
     - If there are zeros, do they give rise to nulls?

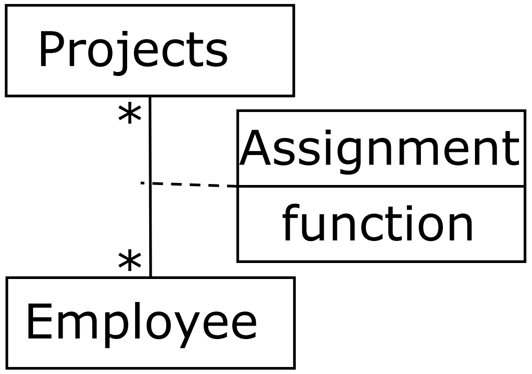
Binary association (*-*)

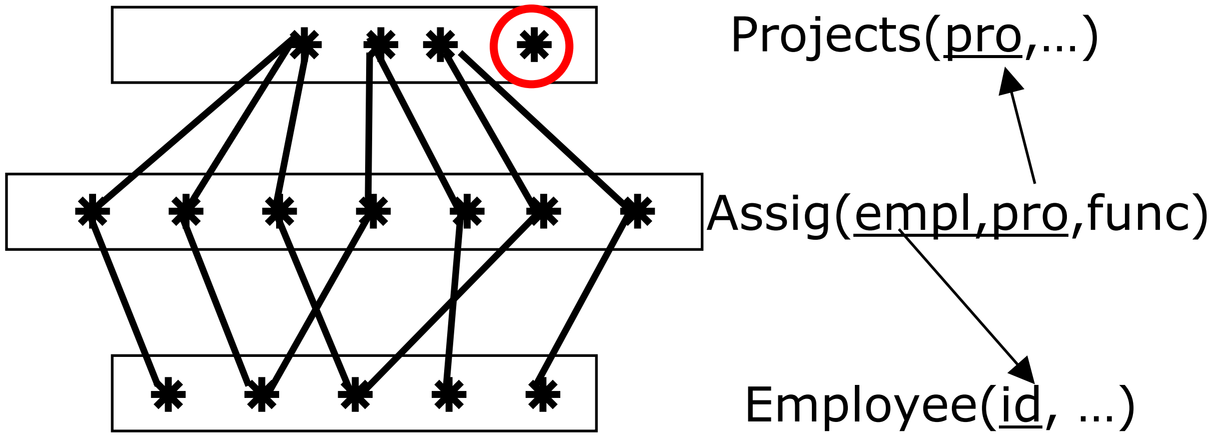




Always a new table!!!

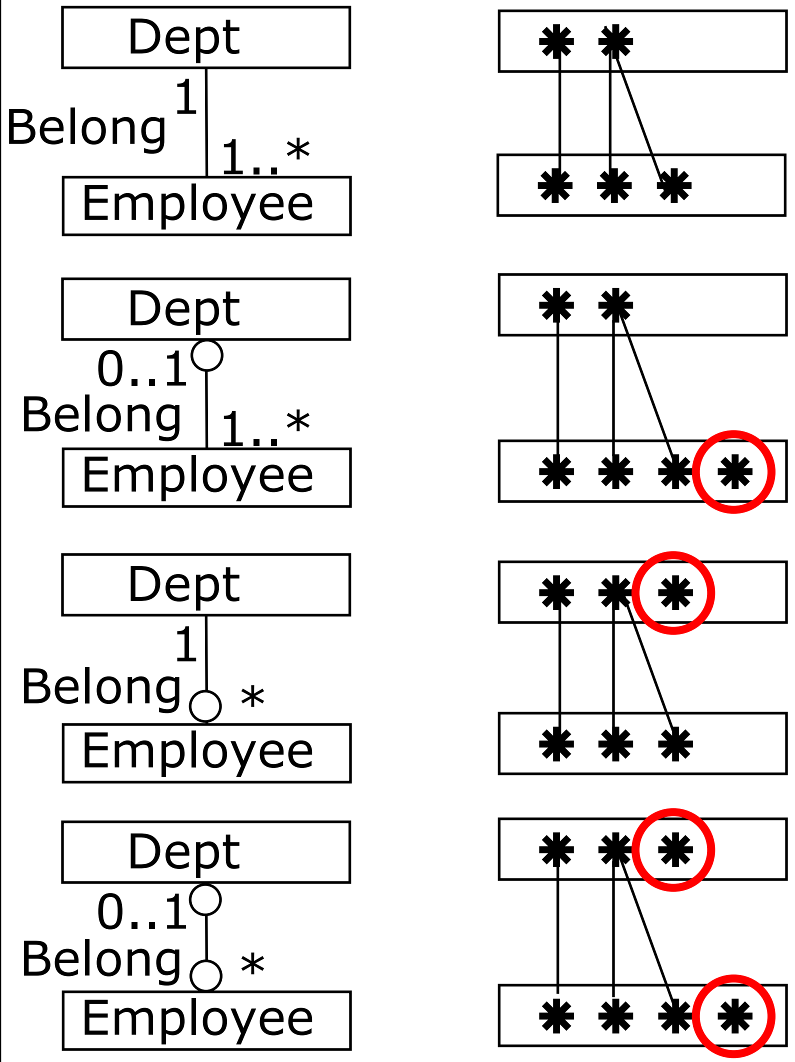
Binary association (*-*)





Always a new table!!!

Binary associations (1-*)



Dept(dpt,…)

Empl(id, …, dpt)

Dept(dpt,…)

Empl(id, …, dpt)

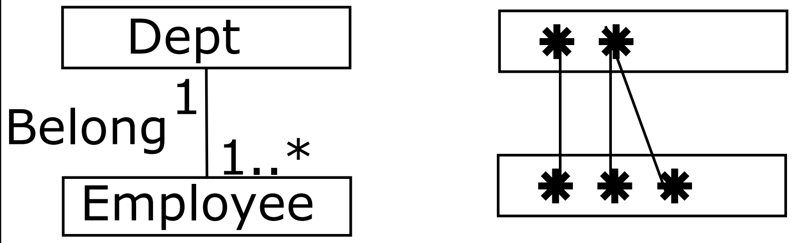
Dept(dpt,…)

Empl(id, …, dpt)

Dept(dpt,…)

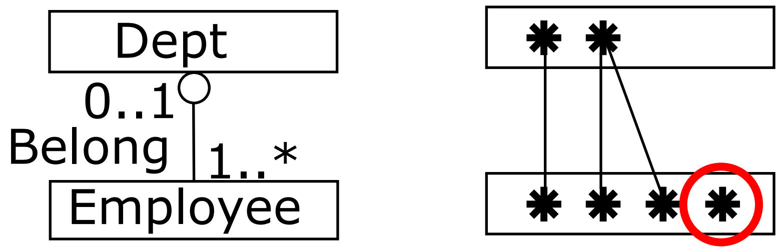
Empl(id, …, dpt)

Binary associations (1-*)



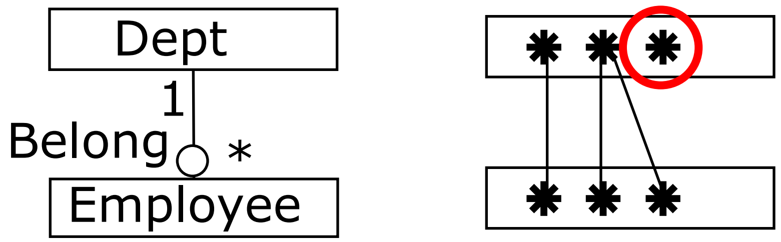
Dept(dpt,…)

Empl(id, …, dpt)
WITHOUT



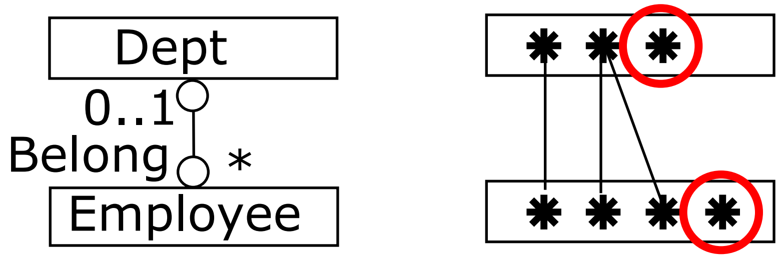
Dept(dpt,…)

Empl(id, …, dpt)



Dept(dpt,…)

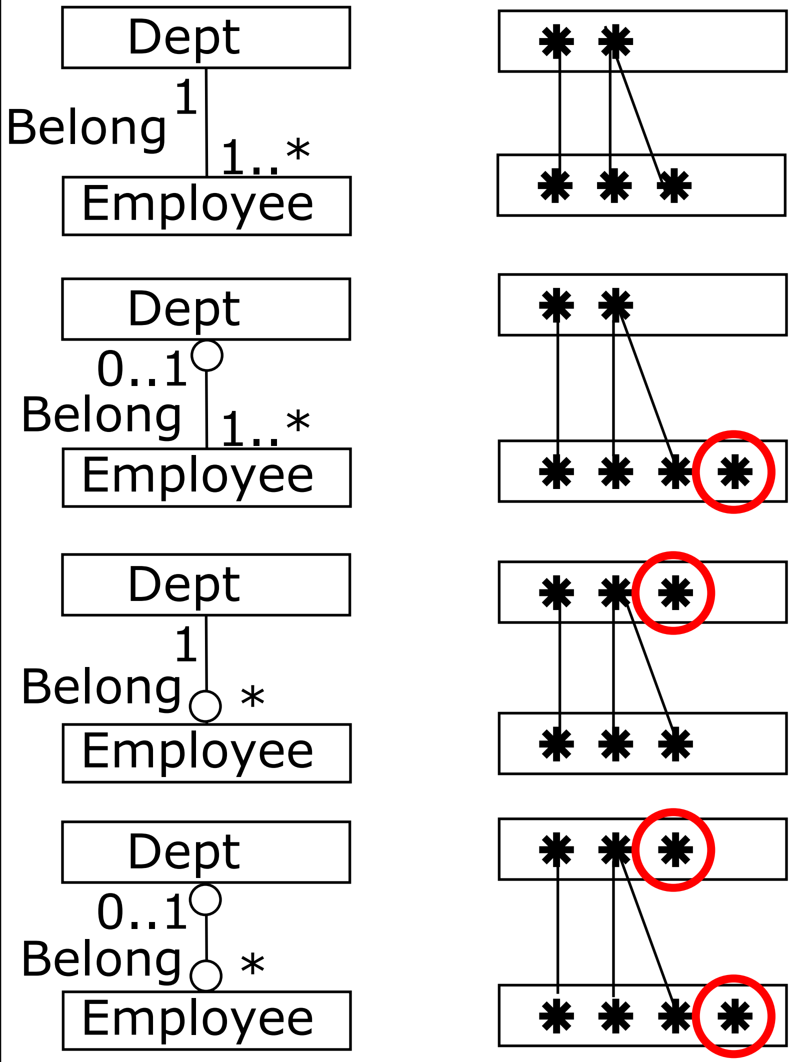
Empl(id, …, dpt)



Dept(dpt,…)

Empl(id, …, dpt)

Binary associations (1-*)



Dept(dpt,…)

Empl(id, …, dpt) WITHOUT

Dept(dpt,…)

Empl(id, …, dpt) WITH

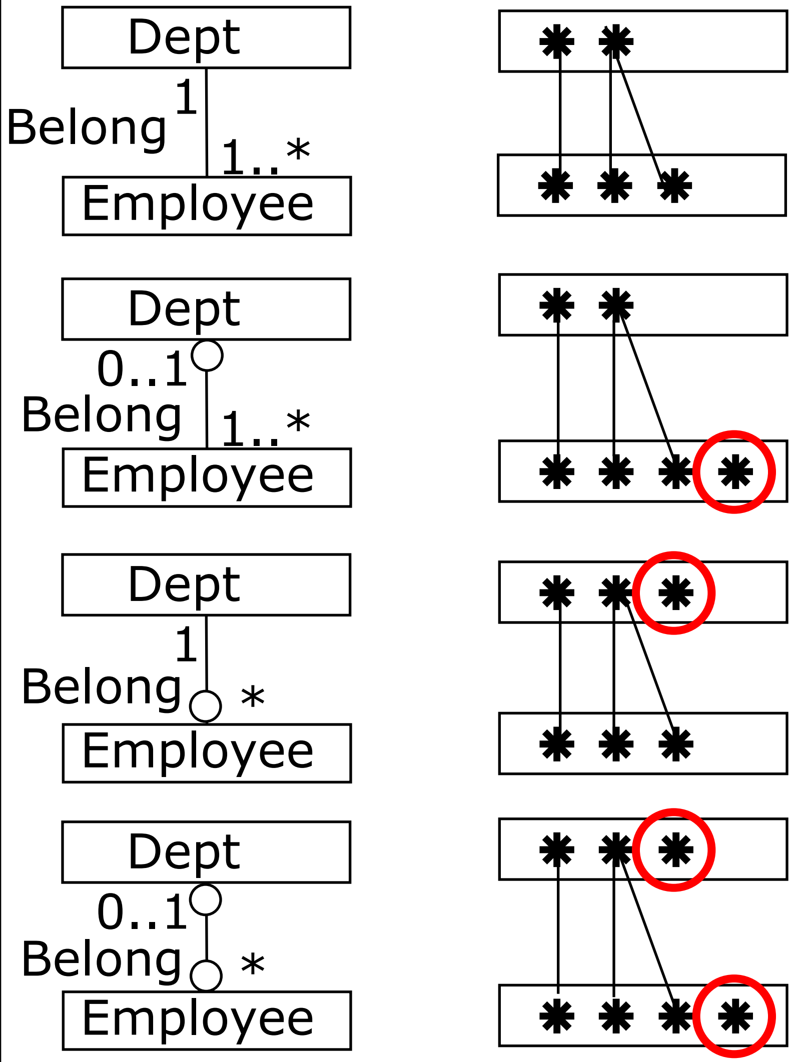
Dept(dpt,…)

Empl(id, …, dpt)

Dept(dpt,…)

Empl(id, …, dpt)

Binary associations (1-*)



Dept(dpt,…)

Empl(id, …, dpt)
WITHOUT

Dept(dpt,…)

Empl(id, …, dpt)
WITH

Dept(dpt,…)
Belong(dpt,empl)
Empl(id,…) WITHOUT

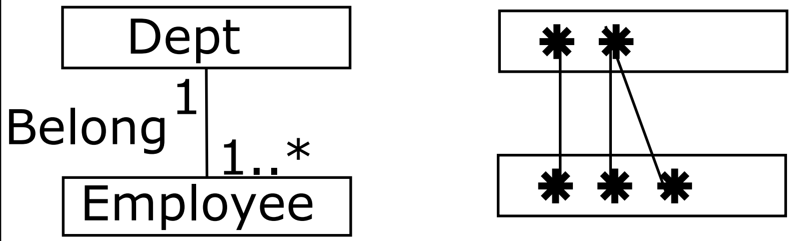
Dept(dpt,…)

Empl(id, …, dpt)

Dept(dpt,…)

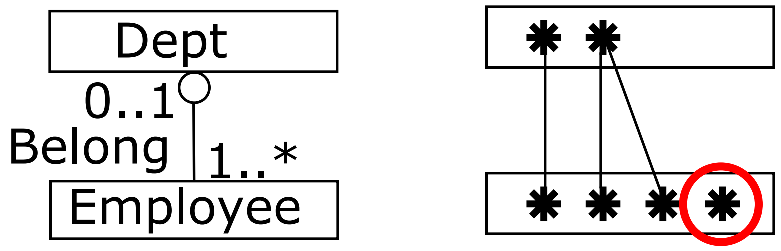
Empl(id, …, dpt)

Binary associations (1-*)



Dept(dpt,…)

Empl(id, …, dpt)
WITHOUT



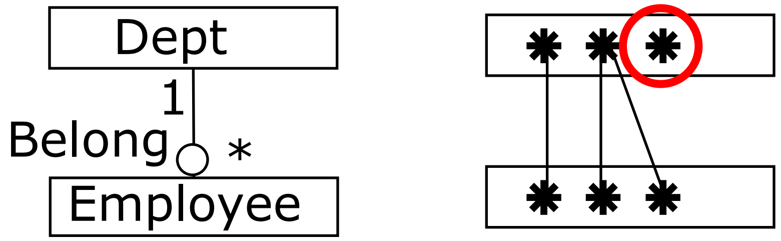
Dept(dpt,…)

Empl(id, …, dpt)
WITH

Dept(dpt,…)

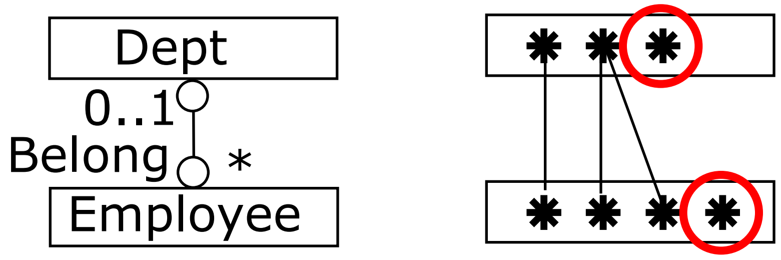
Belong(dpt,empl)

Empl(id,…) WITHOUT



Dept(dpt,…)

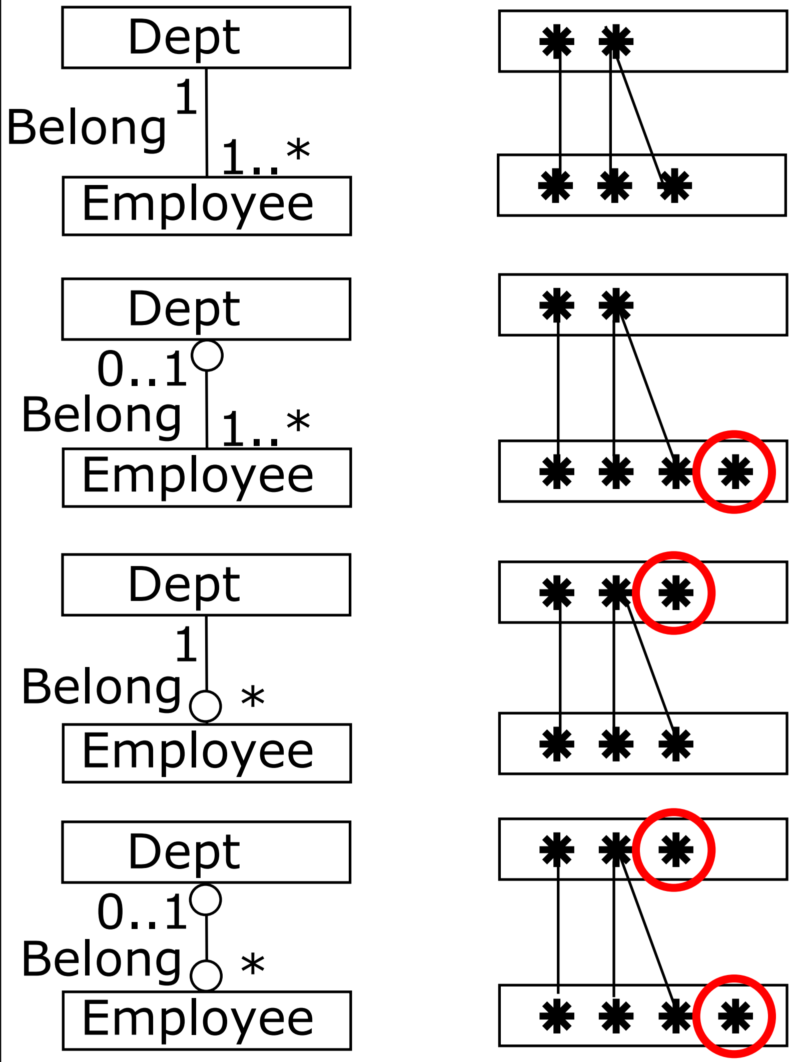
Empl(id, …, dpt)
WITHOUT



Dept(dpt,…)

Empl(id, …, dpt)

Binary associations (1-*)



Dept(dpt,…)

Empl(id, …, dpt)
WITHOUT

Dept(dpt,…)

Empl(id, …, dpt)
WITH

Dept(dpt,…)
Belong(dpt,empl)
Empl(id,…) WITHOUT

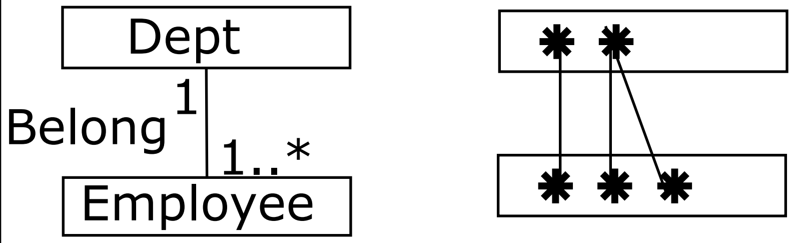
Dept(dpt,…)

Empl(id, …, dpt)
WITHOUT

Dept(dpt,…)

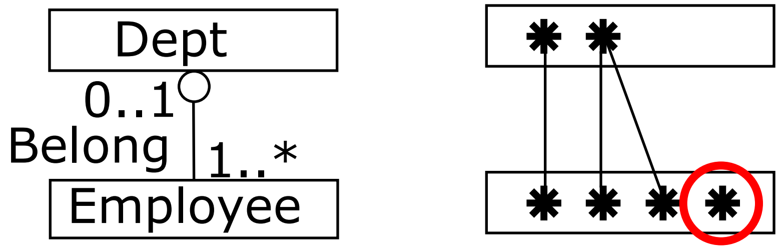
Empl(id, …, dpt)
WITH

Binary associations (1-*)



Dept(dpt,...)

Empl(id, ..., dpt) WITHOUT



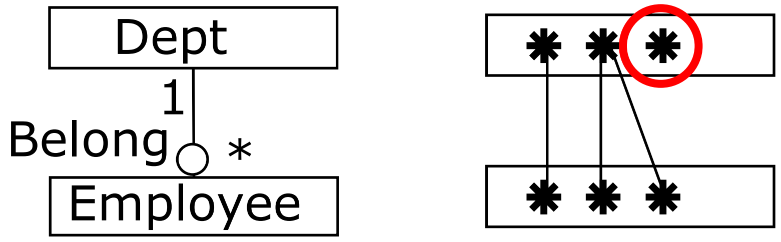
Dept(dpt,...)

Empl(id, ..., dpt) WITH

Dept(dpt,...)

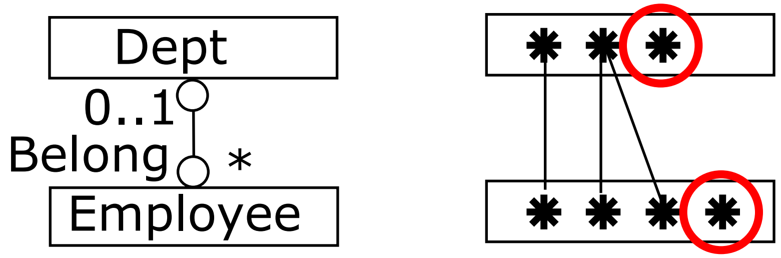
Belong(dpt,empl)

Empl(id,...) WITHOUT



Dept(dpt,...)

Empl(id, ..., dpt) WITHOUT



Dept(dpt,...)

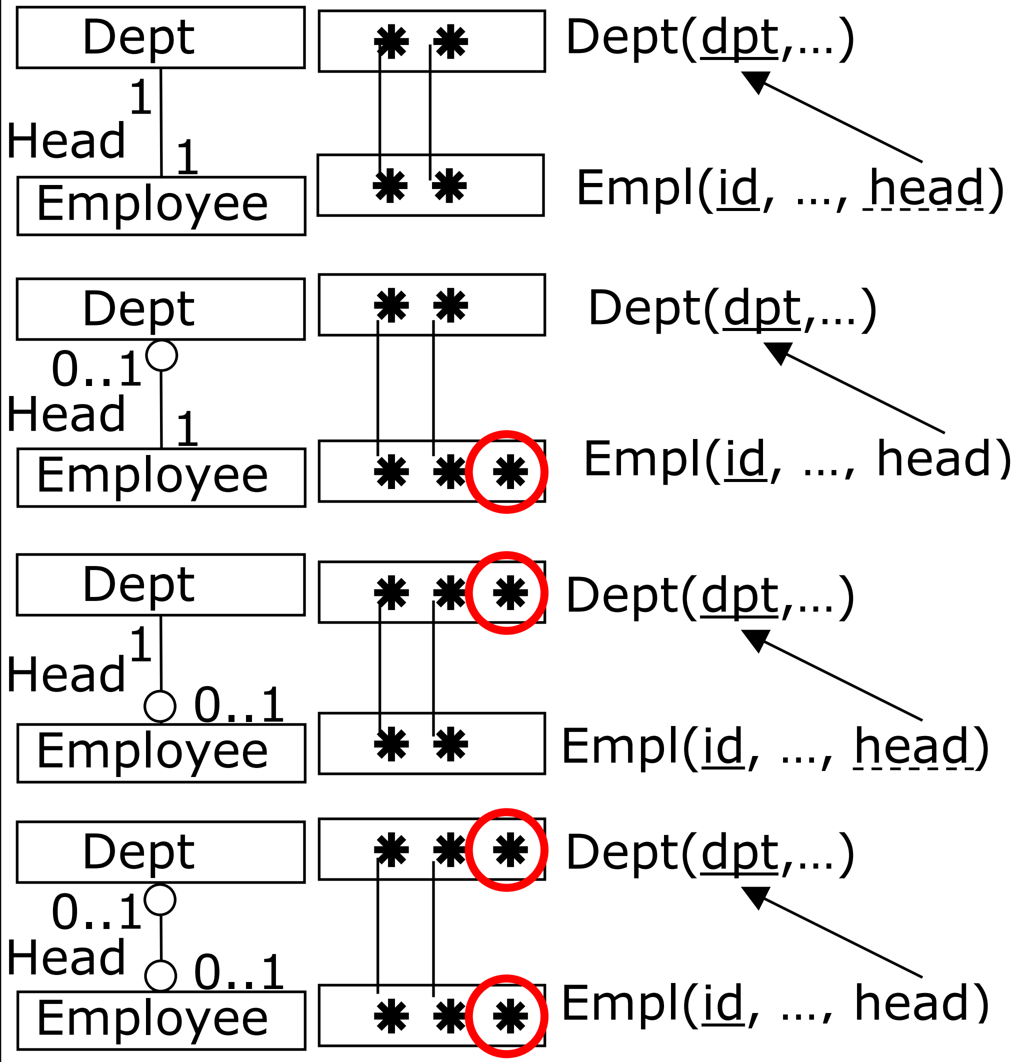
Empl(id, ..., dpt) WITH

Dept(dpt,...)

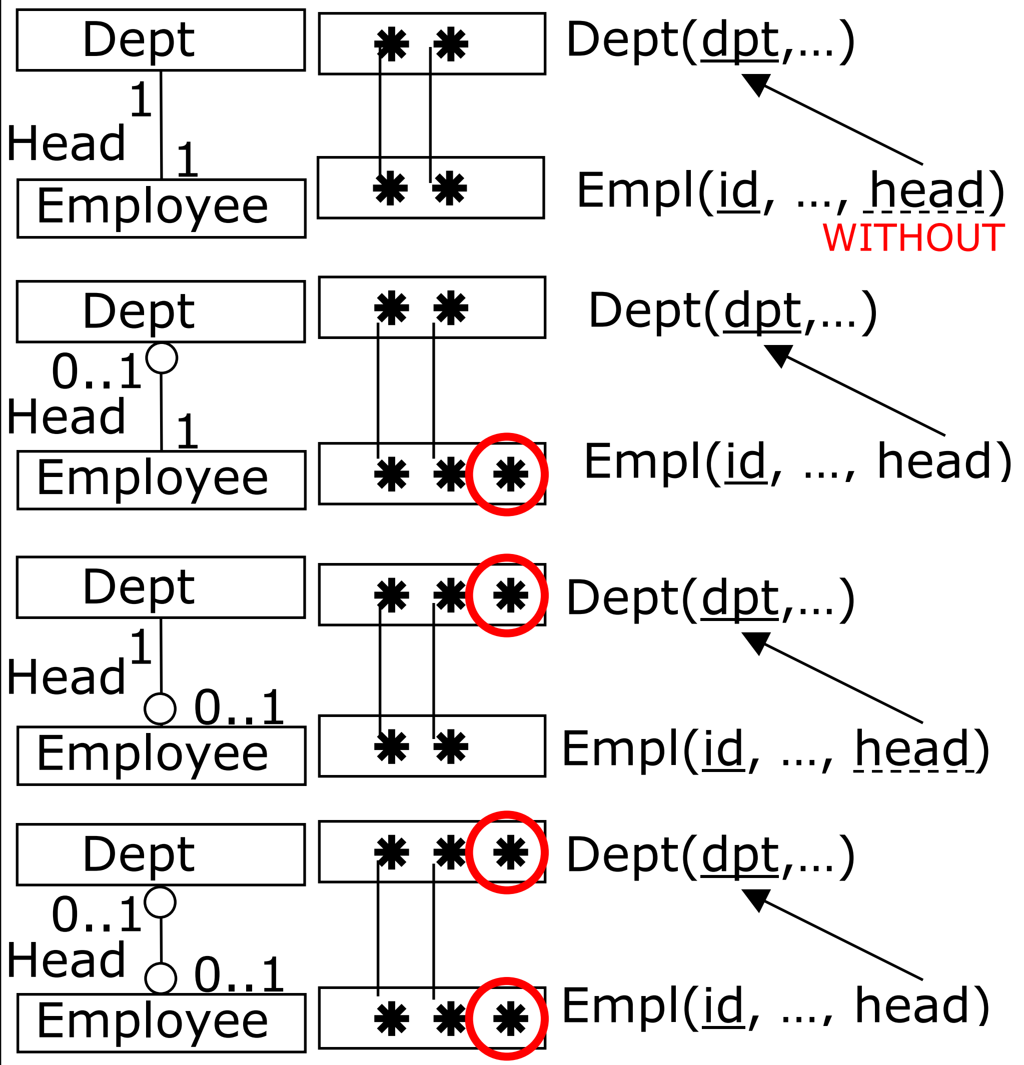
Belong(dpt,empl)

Empl(id,...) WITHOUT

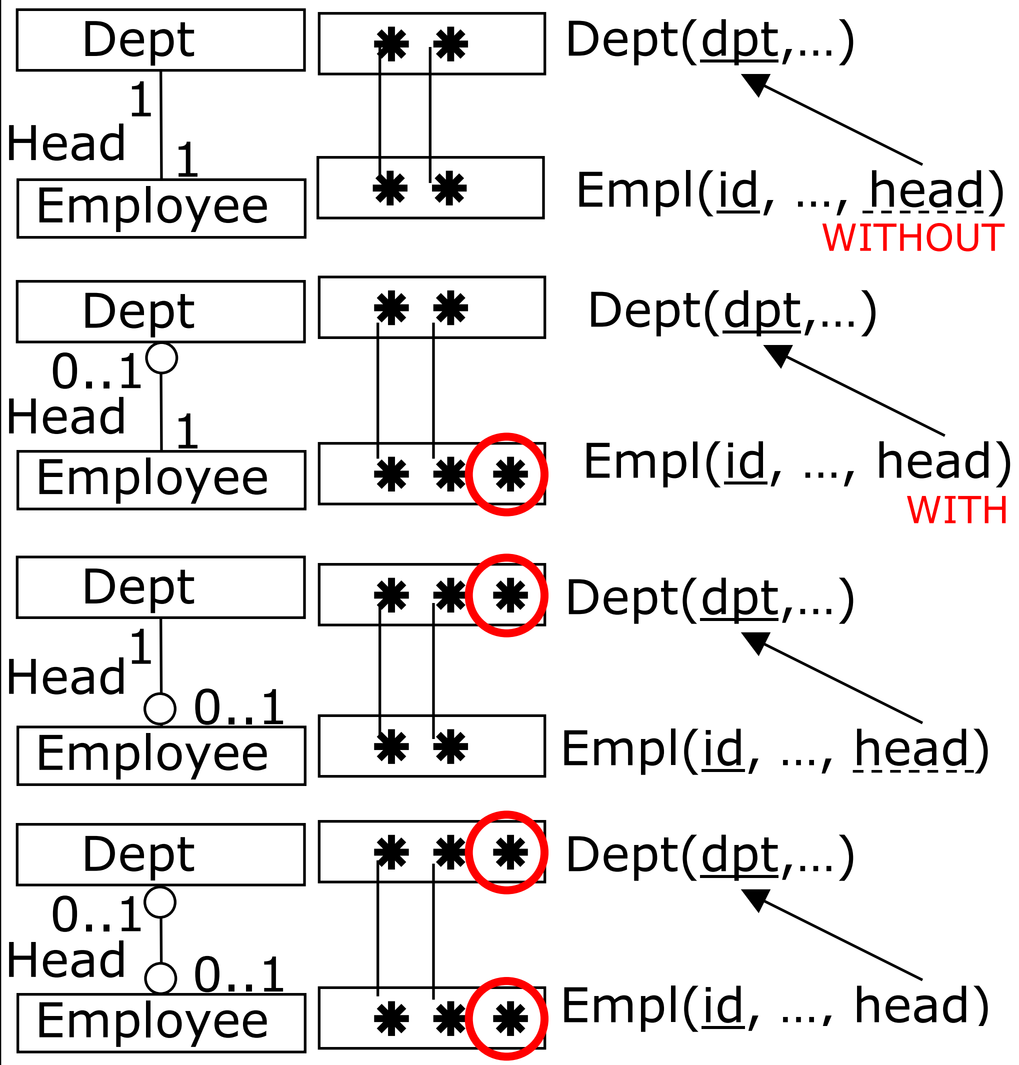
Binary associations (1-1)



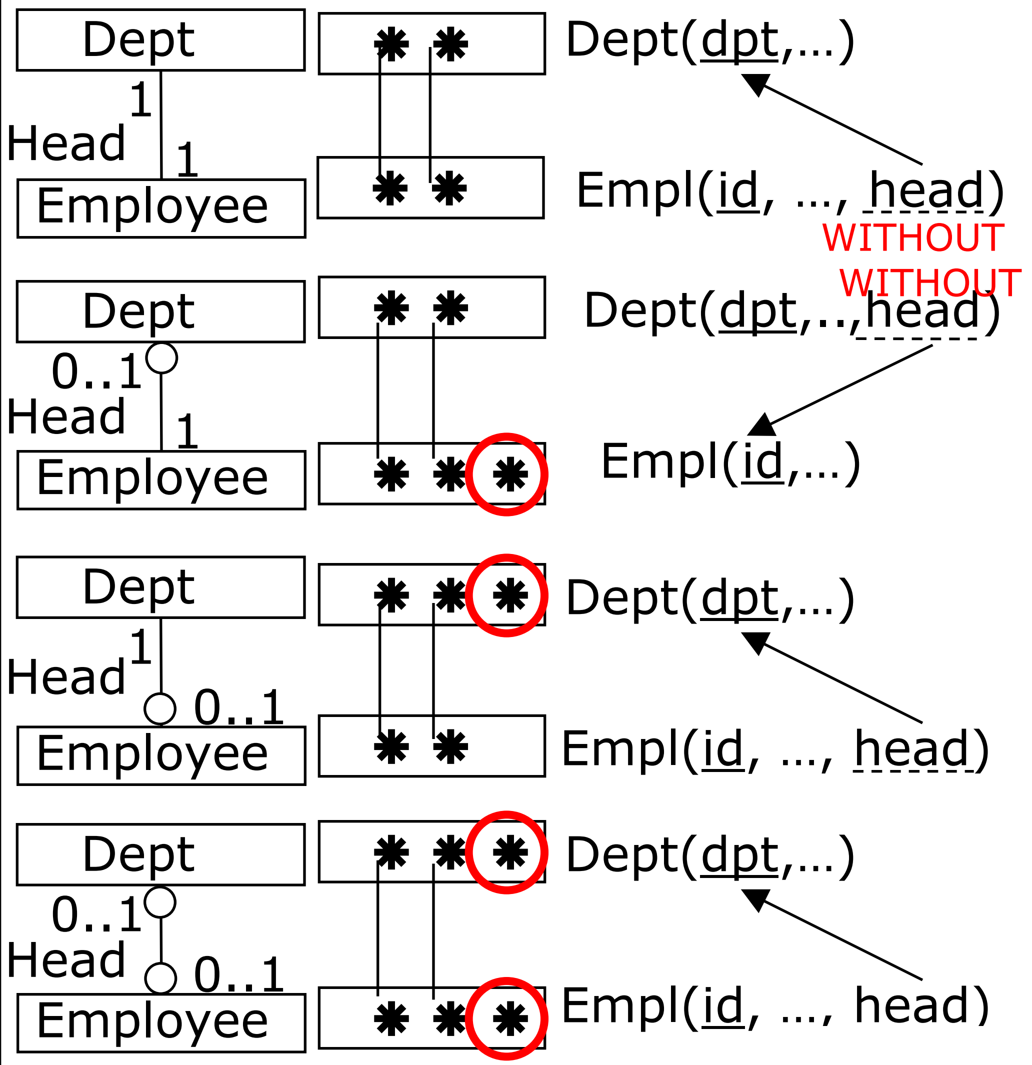
Binary associations (1-1)



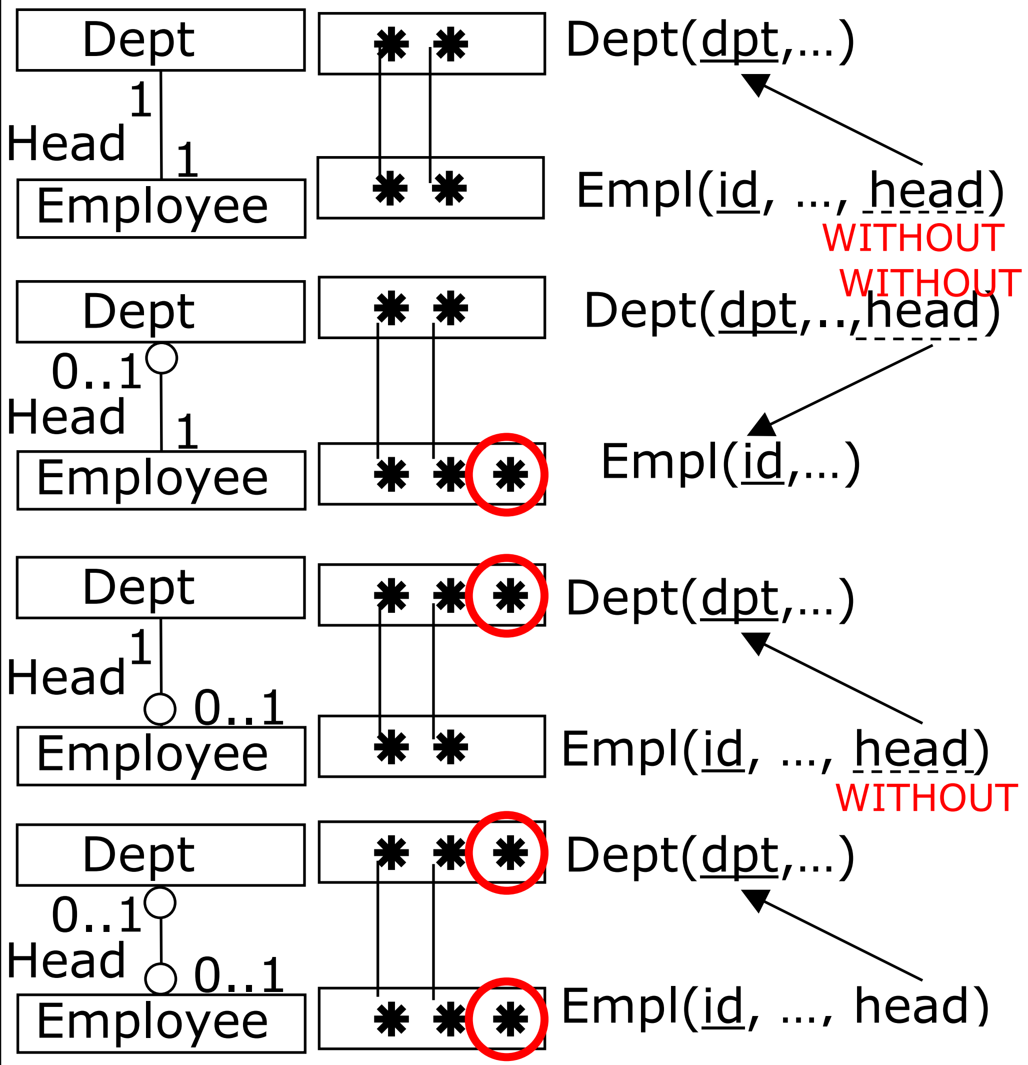
Binary associations (1-1)



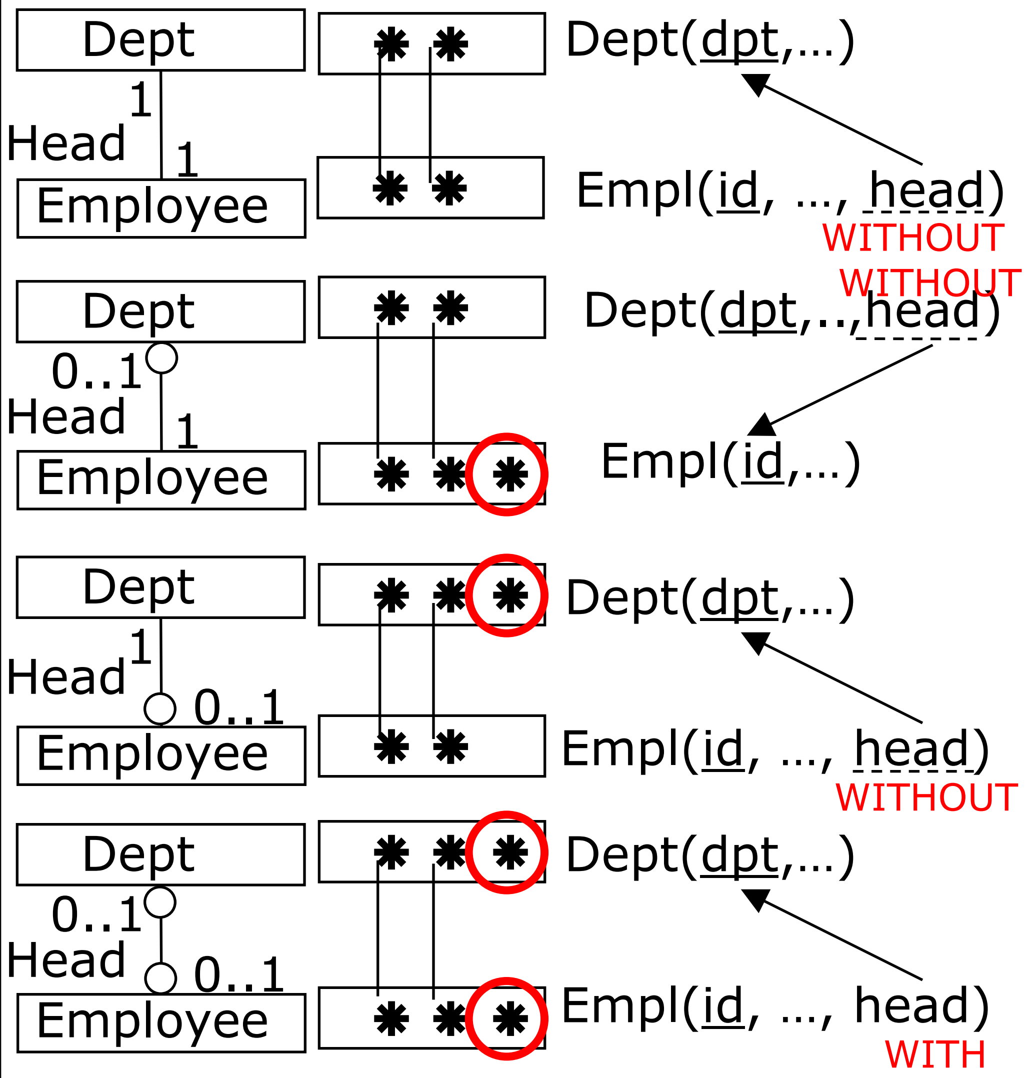
Binary associations (1-1)



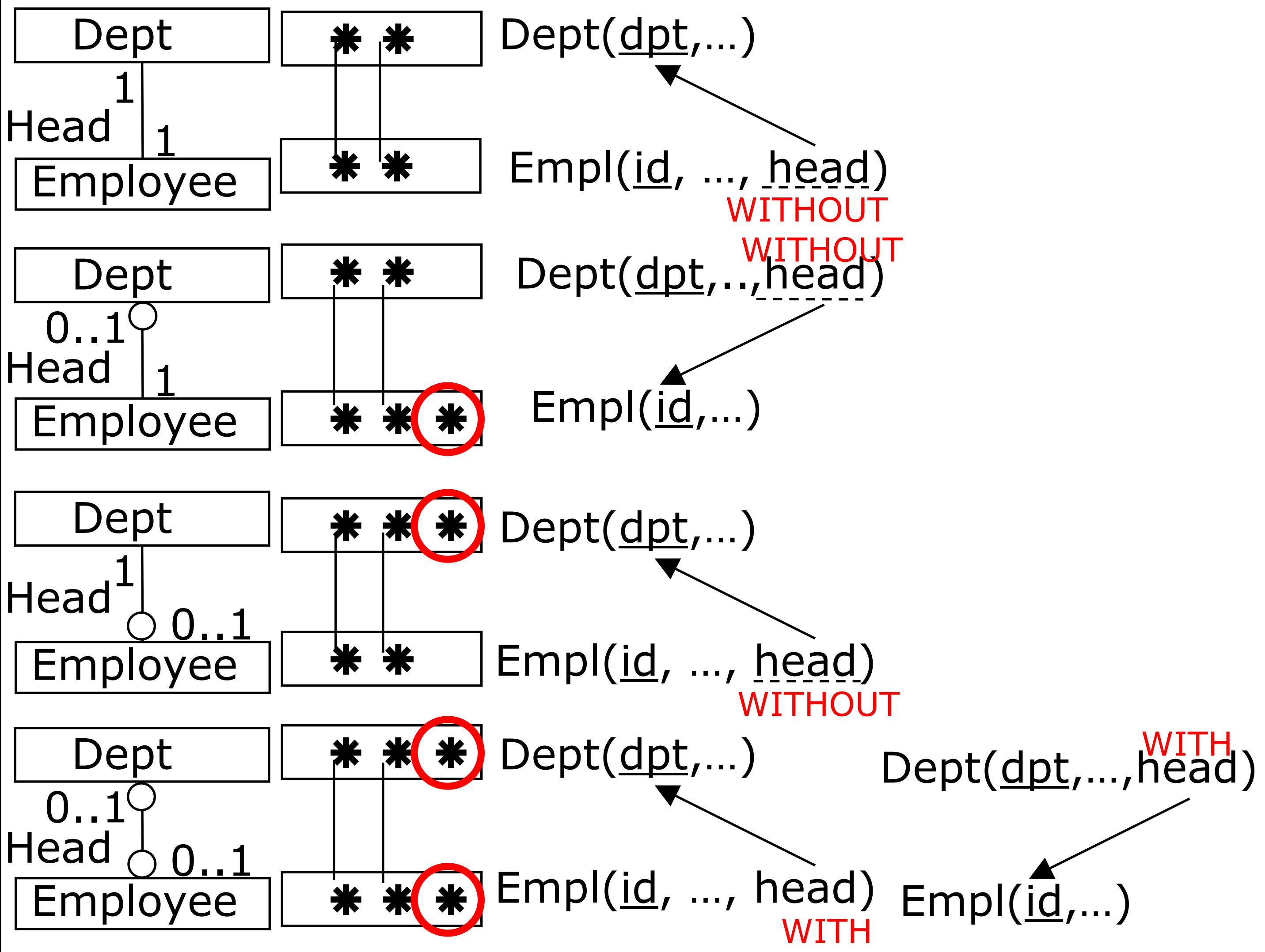
Binary associations (1-1)



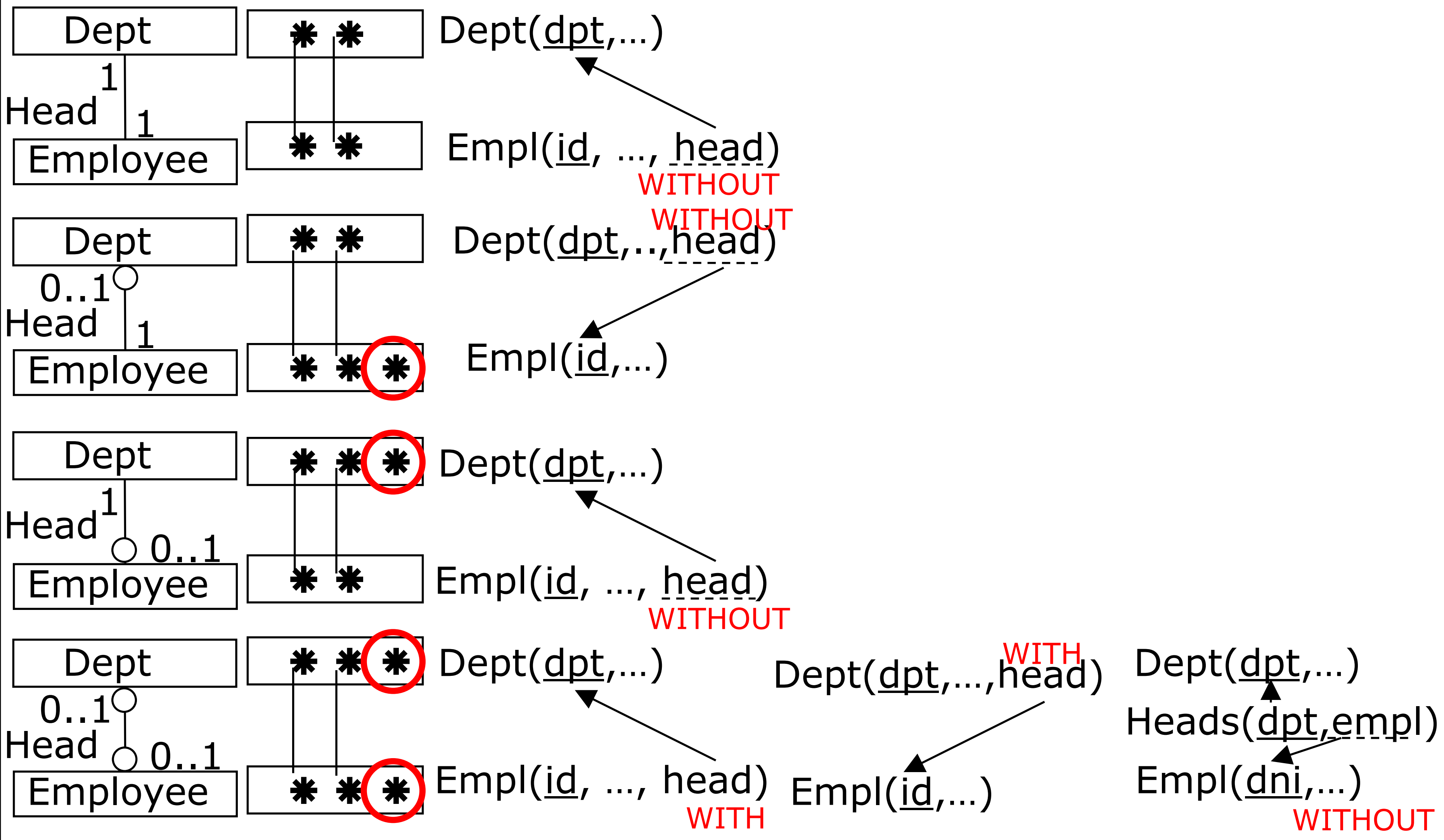
Binary associations (1-1)



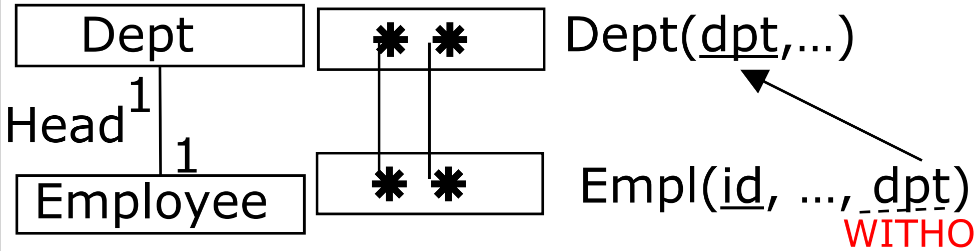
Binary associations (1-1)



Binary associations (1-1)



Fusing classes/Choosing PK

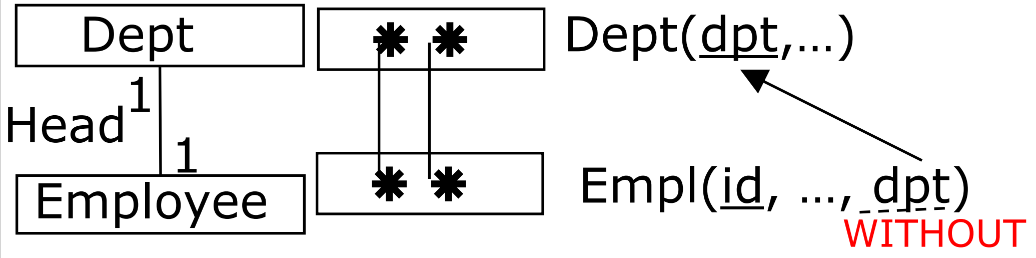


Dept(dpt,…)

Empl(id, …, dpt)

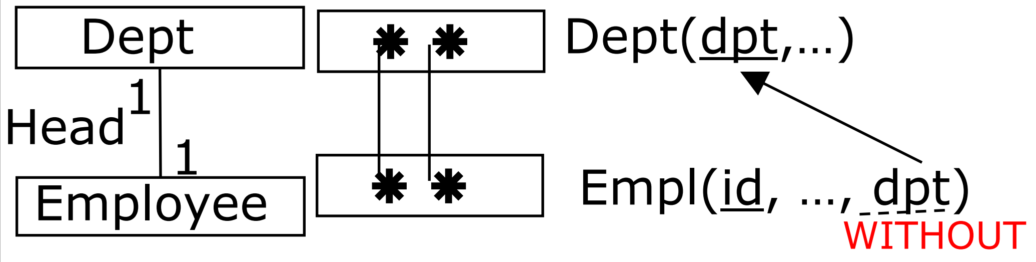
WITHOUT

Fusing classes/Choosing PK



Dept_Empl(dpt,…,empl,…)

Fusing classes/Choosing PK

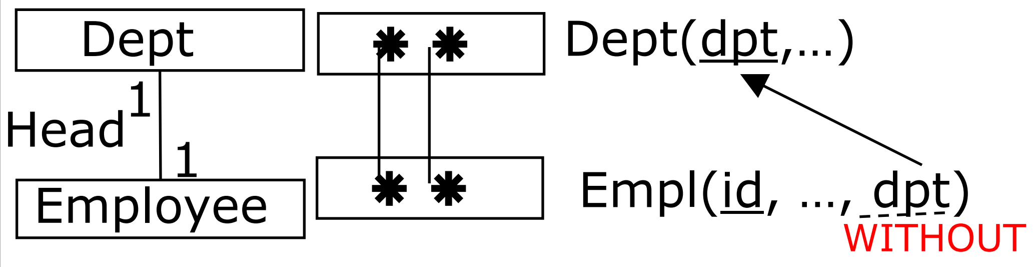


Dept_Empl(dpt,…,empl,…)

Which candidate key would you choose?

- Time
- Space

Fusing classes/Choosing PK



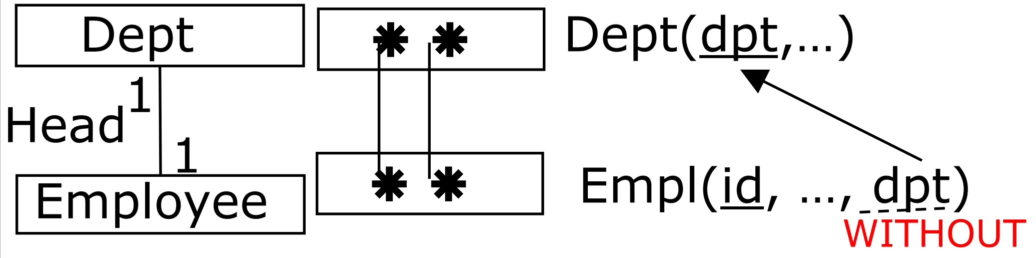
Dept_Empl(dpt,…,empl,…)

Which candidate key would you choose?

Country_President(	country, …,	president,…)
	USA	B. Obama
	Spain	M. Rajoy

- Time
- Space

Fusing classes/Choosing PK



Dept_Empl(dpt,…,empl,…)

Which candidate key would you choose?

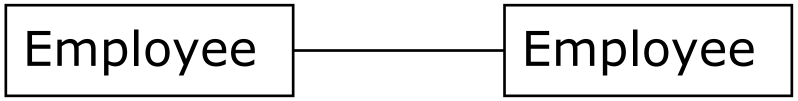
Country_President(	country, …,	president,…)
	USA	B. Obama
	Spain	M. Rajoy

- Time
- Space
- Change frequency

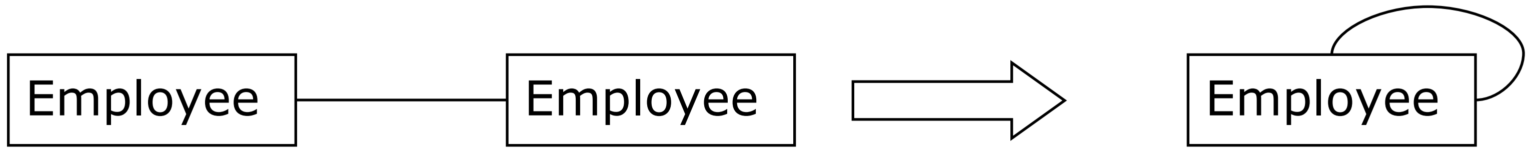
Attributes of relationships

- *-* or n-ary (common)
  - In the table representing the association
- 1-* (uncommon)
  - If any, in the table representing the association
  - Otherwise?
- 1-1 (rare)
  - If any, in the table representing the association
  - If only one table (fusion), in it
  - Otherwise?

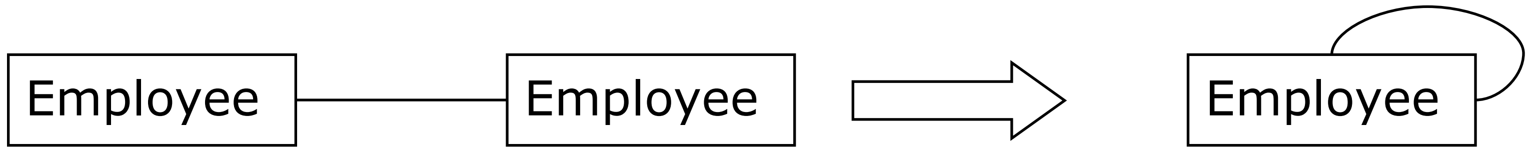
Reflexive associations



Reflexive associations

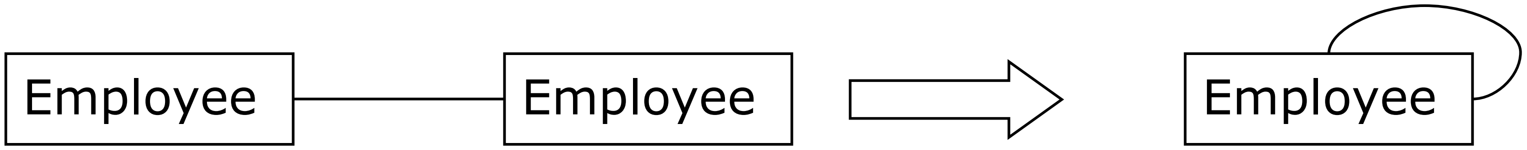


Reflexive associations



- Valid multiplicities:
  - *-* (Relatives)
  - 1-* (Mother)
  - 1-1 (Couple)
- Singularity:

Reflexive associations

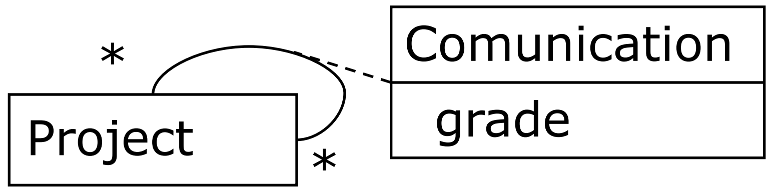


- Valid multiplicities:
  - *-* (Relatives)
  - 1-* (Mother)
  - 1-1 (Couple)
- Singularity:
  - May be symmetric or not

Brother1	Brother2
John	Peter
Peter	John

Friend1	Friend2	Grade
John	Peter	10
Peter	John	2

Reflexive multiplicities



Project(pro,…)

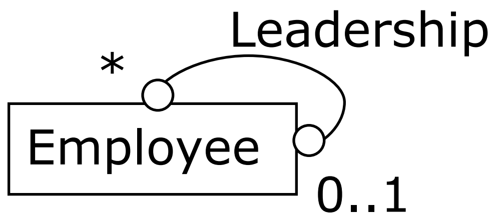
Requires_com(pro,pro-com,grade)

Employee(emp,…,leader) WITH

or

Employee(emp,…)

Leadership(emp,emp-leader) WITHOUT

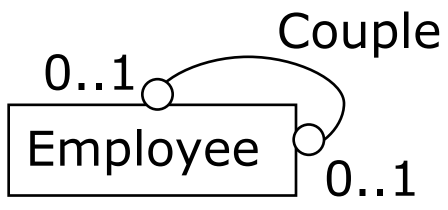


Employee(emp,…,couple) WITH

or

Employee(emp,…)

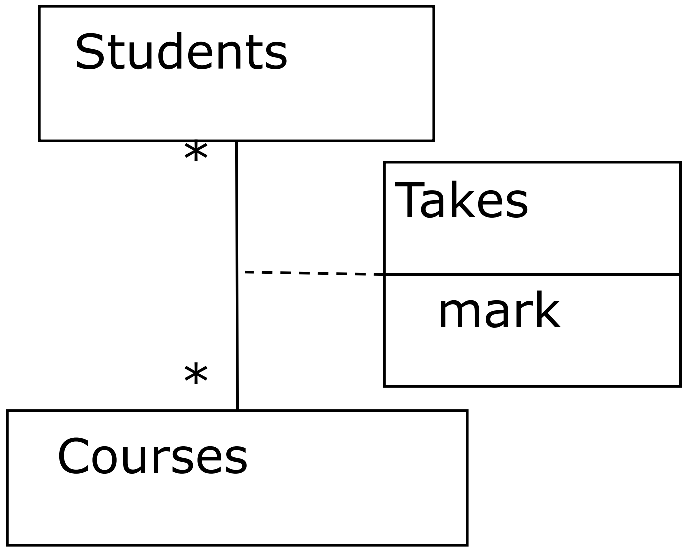
Couple(emp,emp-couple) WITHOUT



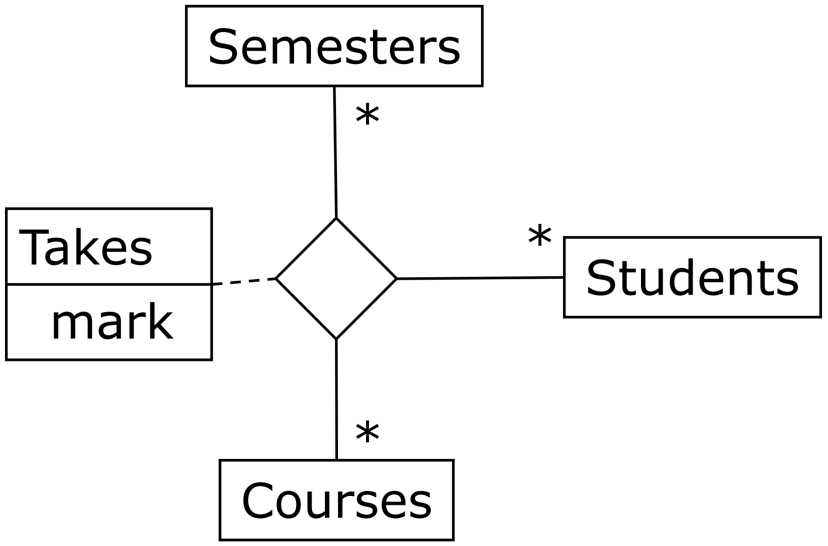
Symmetric reflexive associations

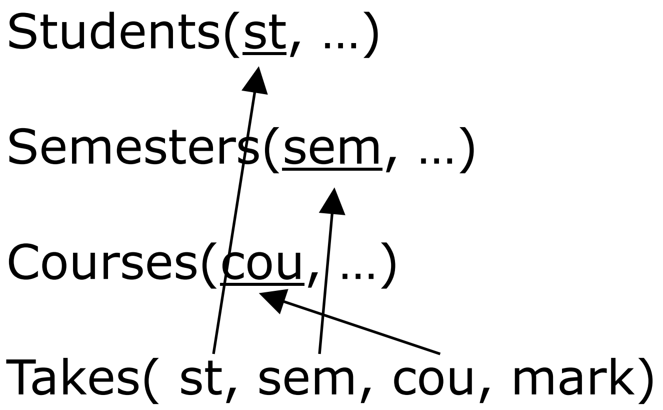
- We must preserve the property. Triggers may bring a satisfactory solution:
  - INSERT (a, b) → INSERT (b, a)
  - DELETE (a, b) → DELETE (b, a)
  - UPDATE…
- We may store only half of the pairs
  - CREATE VIEW to simulate the whole set of pairs
  - Trigger INSERT (a, b) → look if (b, a) already present
  - Trigger DELETE (a, b) → look if (b, a) is present instead
  - Trigger UPDATE…

Can we improve this model?



Ternary associations (*-*-*)




Students(<u>st</u>, …)

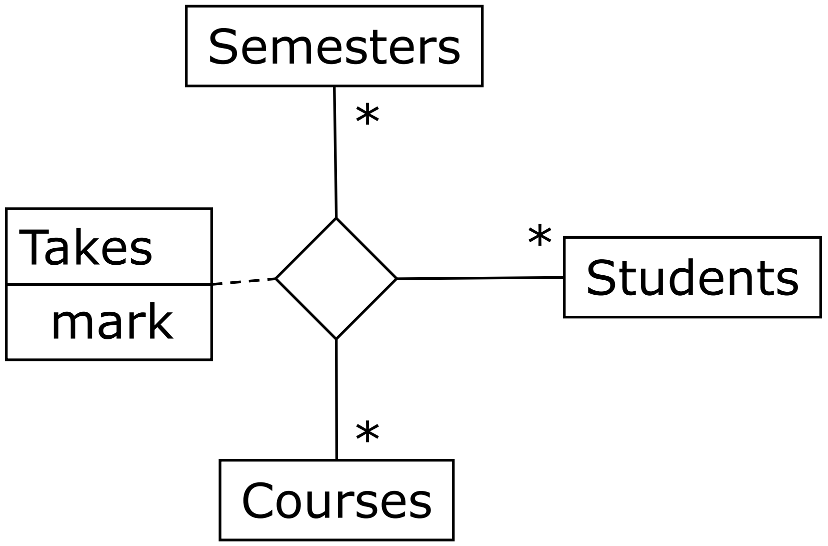
Semesters(<u>sem</u>, …)

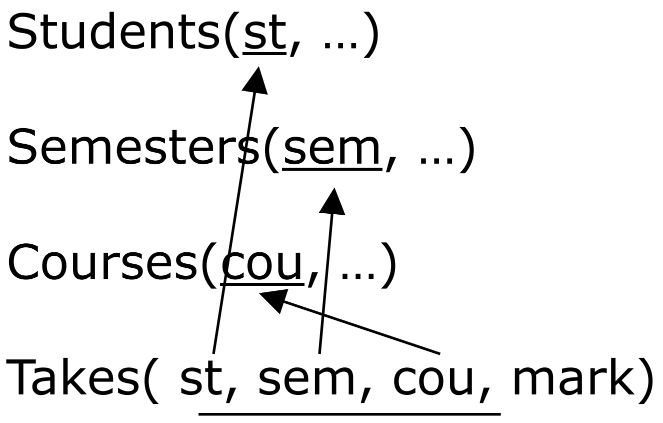
Courses(<u>cou</u>, …)

Takes(st, sem, cou, mark)

Always a new table!!!

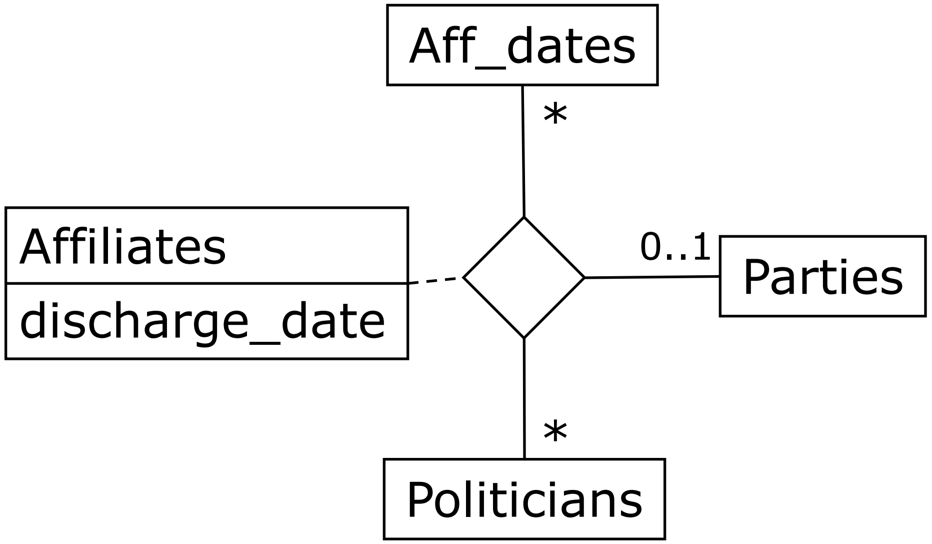
Ternary associations (*-*-*)





Always a new table!!!

Ternary associations (*-*-1)



Parties(acronym, …)

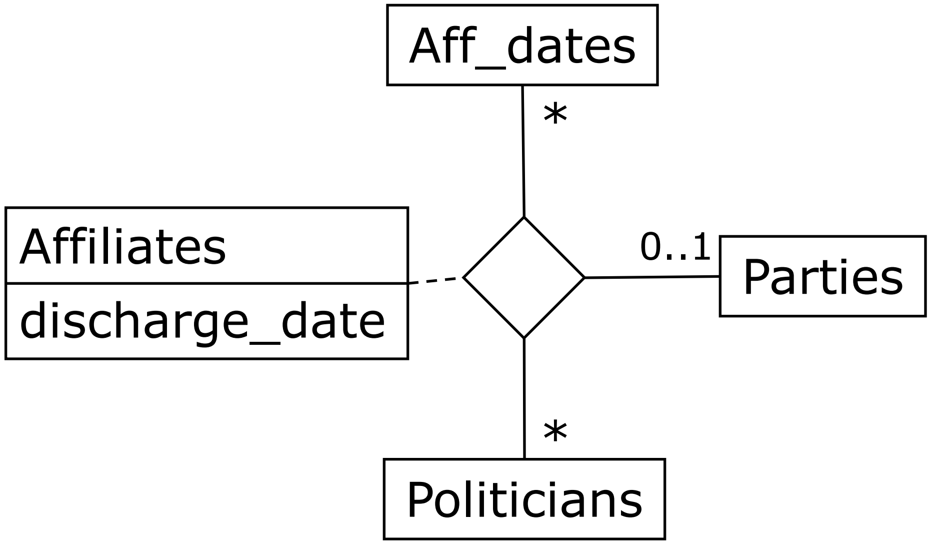
Aff_dates(date, …)

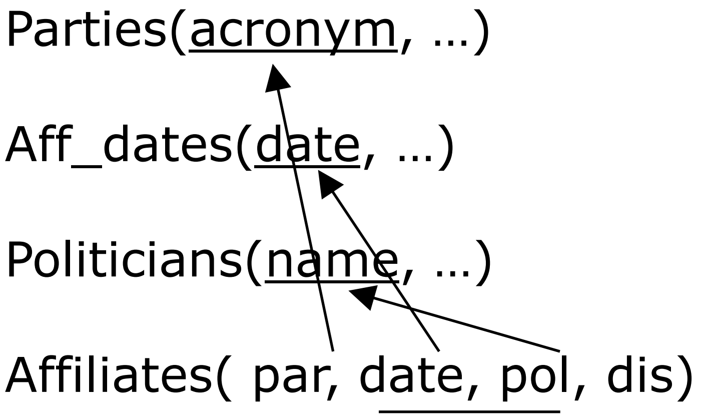
Politicians(name, …)

Affiliates(par, date, pol, dis)

Always a new table!!!

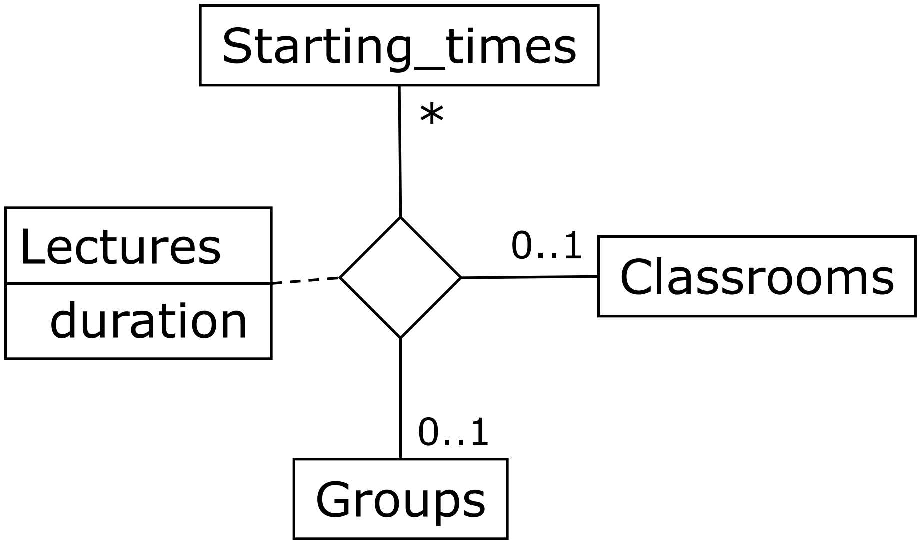
Ternary associations (*-*-1)

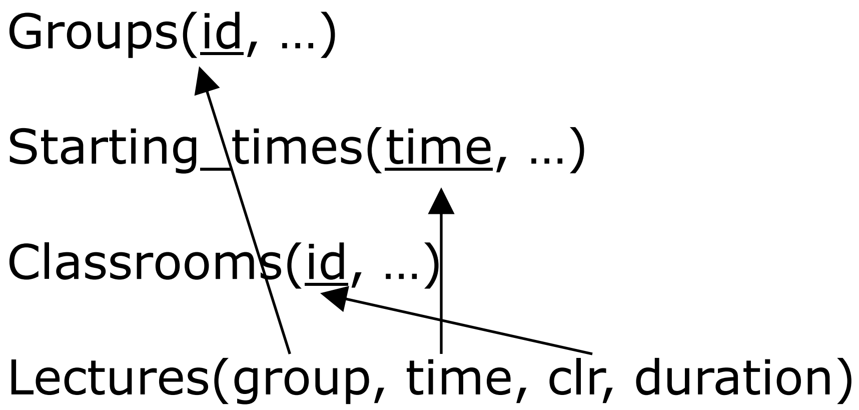




Always a new table!!!

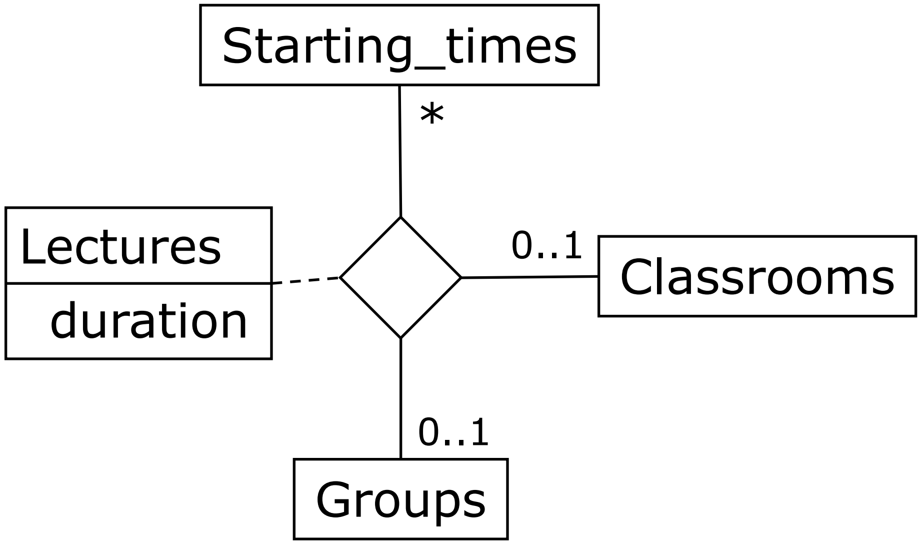
Ternary associations (*-1-1)

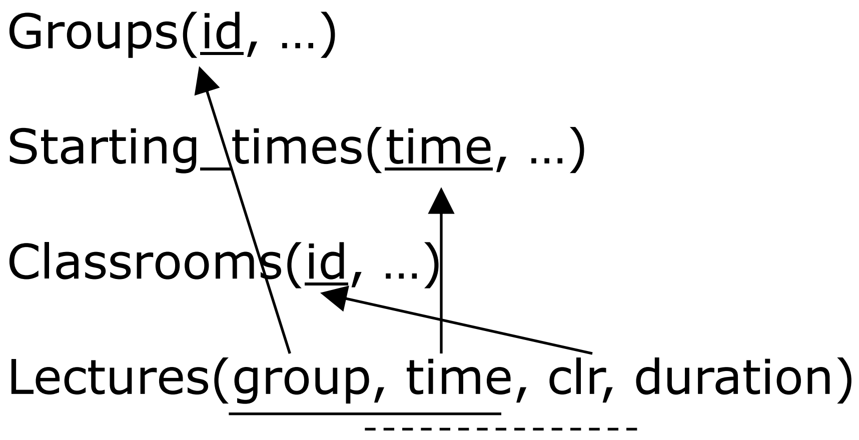




Always a new table!!!

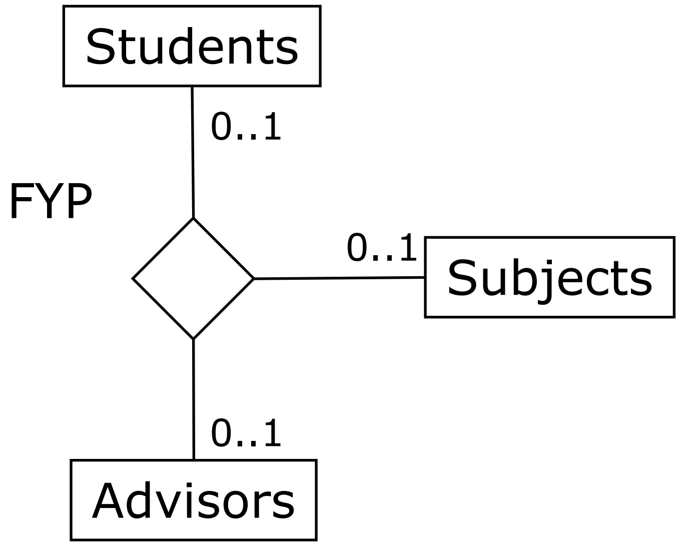
Ternary associations (*-1-1)





Always a new table!!!

Ternary associations (1-1-1)



Subjects(<u>subjects</u>, …)

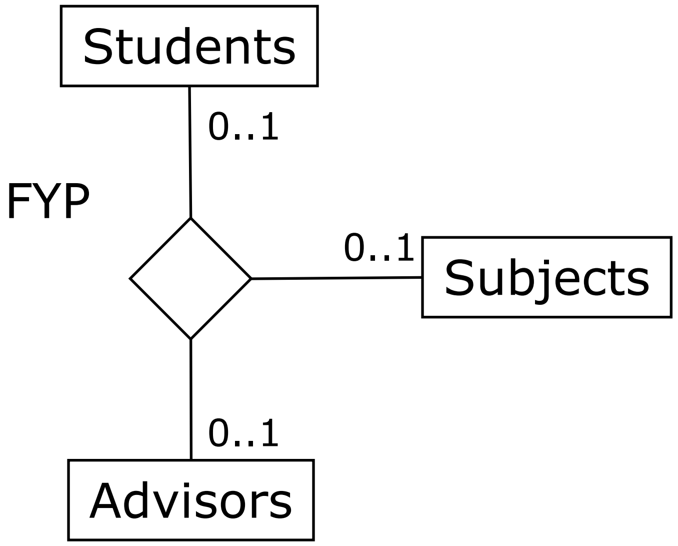
Students(<u>name</u>, …)

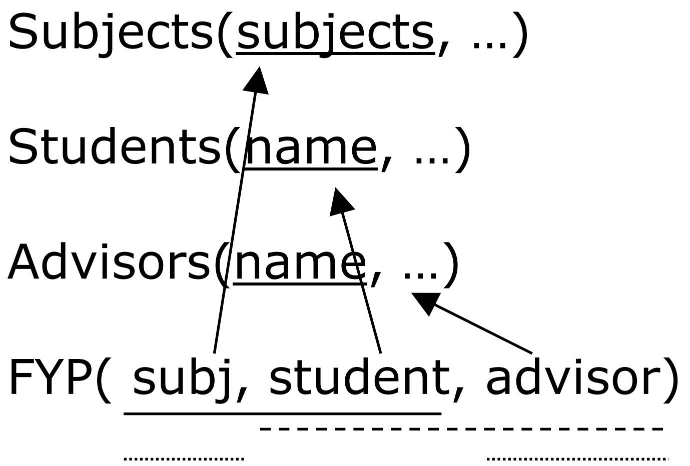
Advisors(<u>name</u>, …)

FYP(subj, student, advisor)

Always a new table!!!

Ternary associations (1-1-1)




Subjects(subjects, …)

Students(name, …)

Advisors(name, …)

FYP(subj, student, advisor)

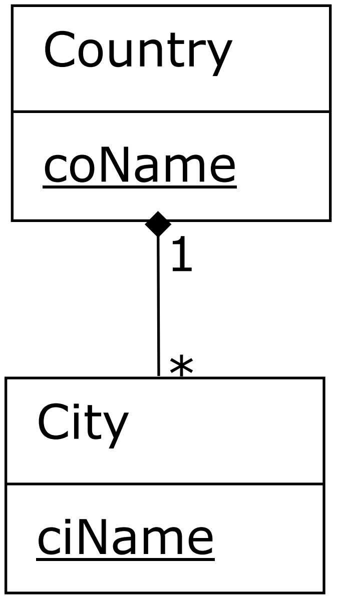
Always a new table!!!

N-ary associations

Binary:	A new table or foreign key
Ternary:	A new table
Quaternary:	A new table
…	

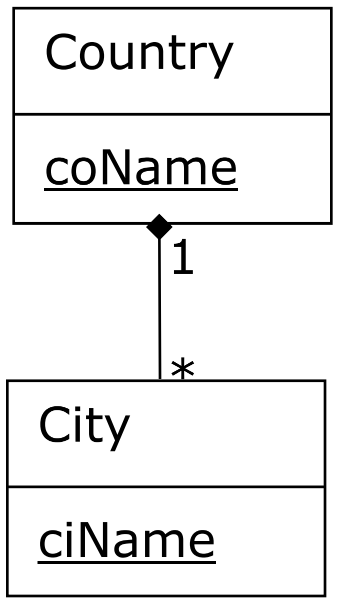
Compound aggregation (I)

- Weak class, with regard to the external key of the classical relational model



Compound aggregation (I)

- Weak class, with regard to the external key of the classical relational model



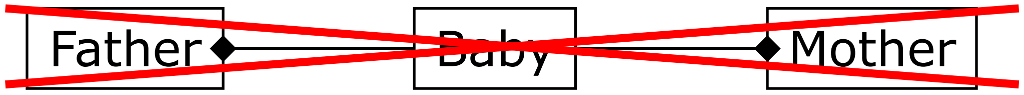
Country(<u>coName</u>, …)

$\nearrow$

City(<u>coName, ciName</u>, …)

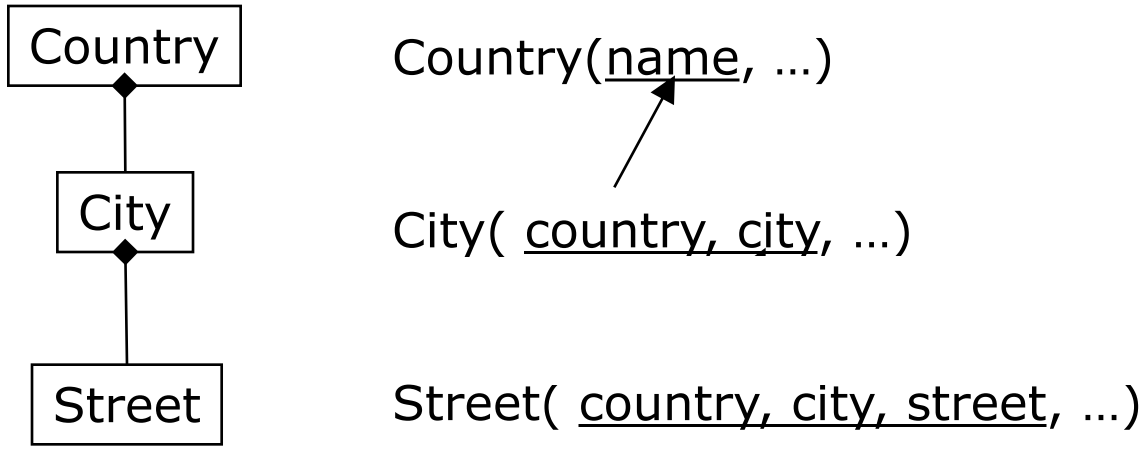
Compound aggregation (II)

- A given class cannot be part of two



- Composition cannot have zeros at "to-one" side

- Compositions can be chained



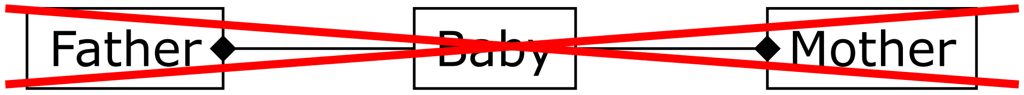
Country(<u>name</u>, …)

City(<u>country, city</u>, …)

Street(<u>country, city, street</u>, …)

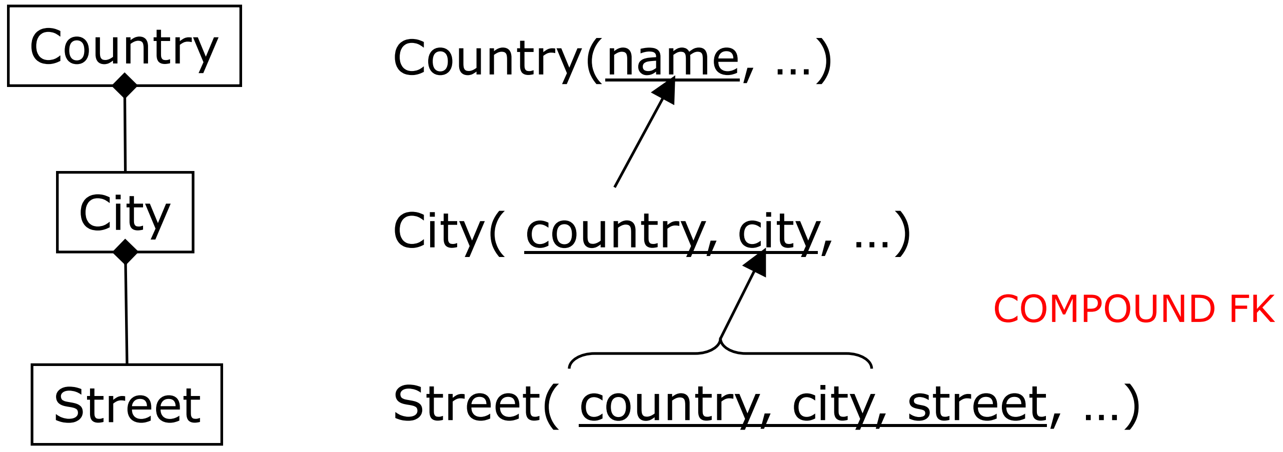
Compound aggregation (II)

- A given class cannot be part of two



- Composition cannot have zeros at "to-one" side

- Compositions can be chained



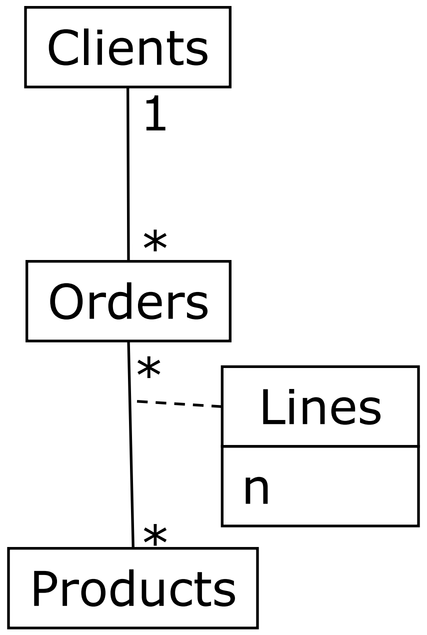
Country(<u>name</u>, …)

City(<u>country, city</u>, …)

Street(<u>country, city, street</u>, …)

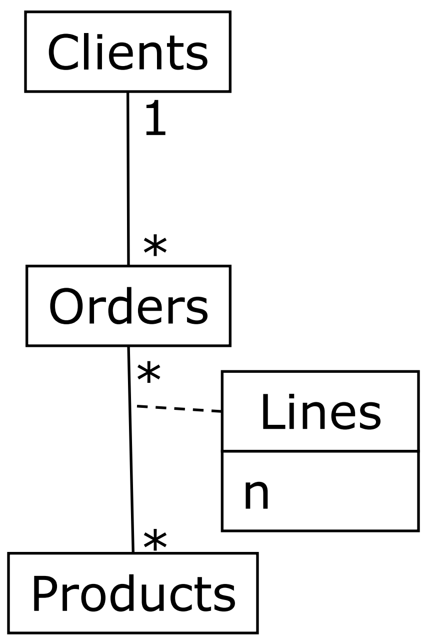
Class vs Association

ASSOCIATION

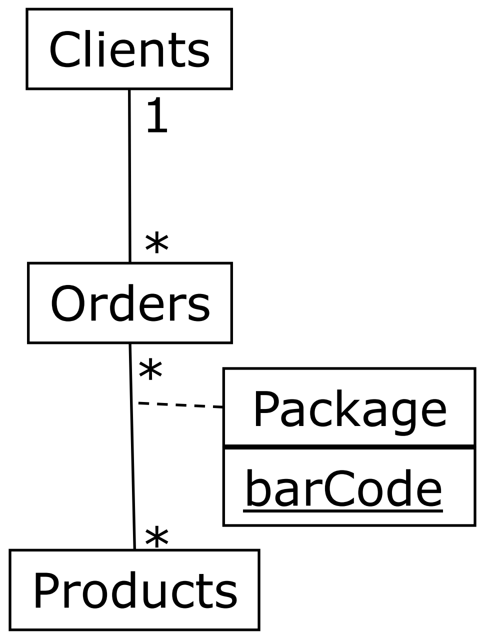


Class vs Association

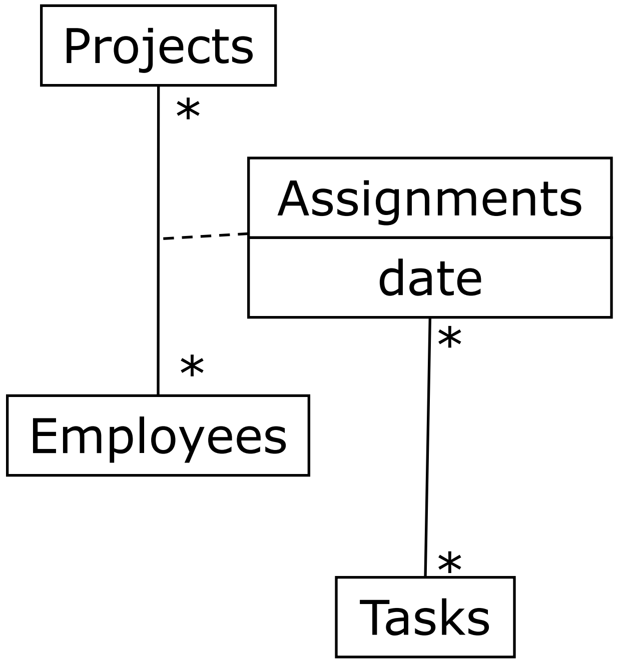
ASSOCIATION

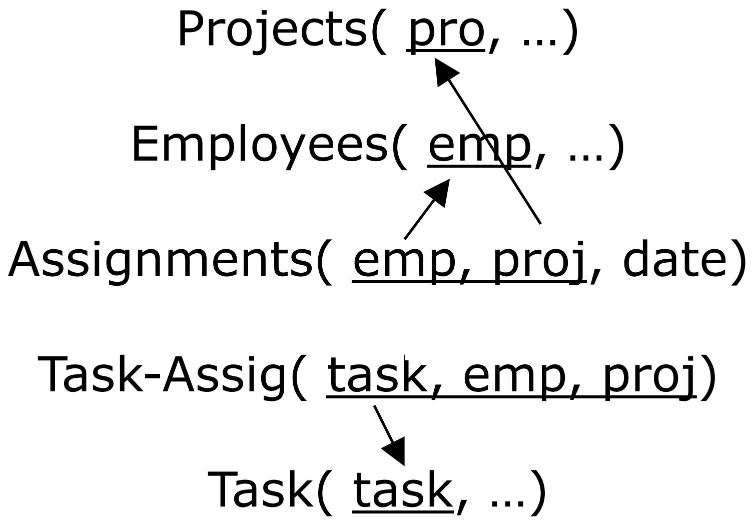


ASSOCIATION-CLASS



Association classes (I)





Projects(pro, …)

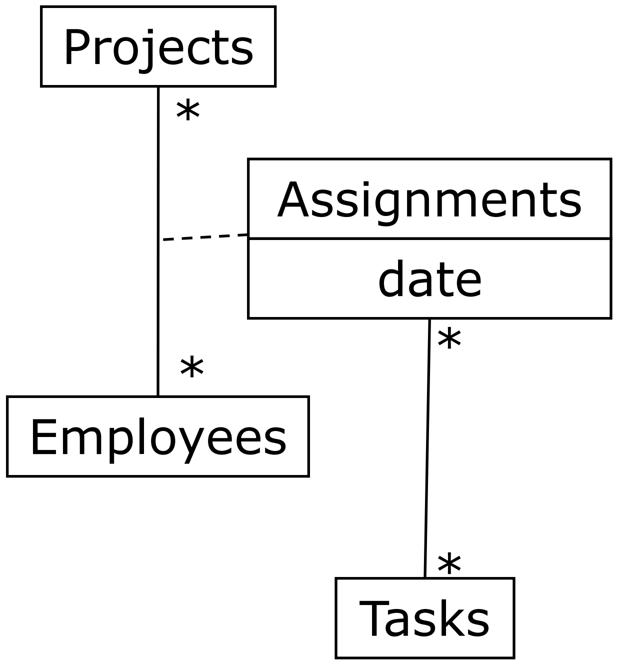
Employees(emp, …)

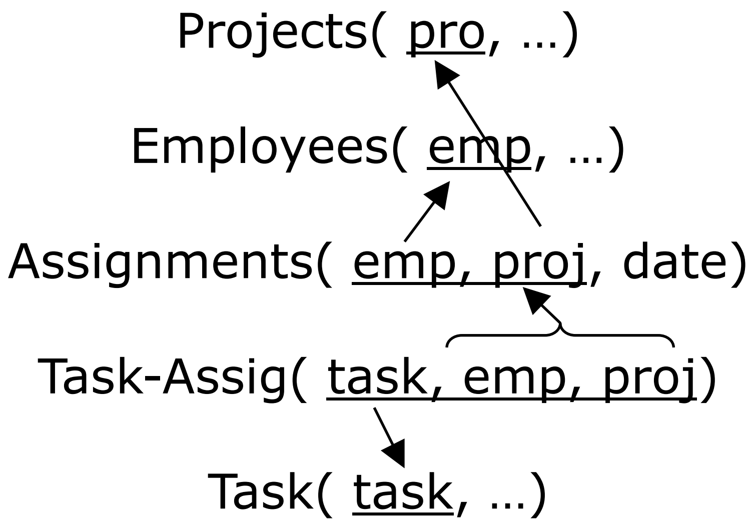
Assignments(emp, proj, date)

Task-Assig(task, emp, proj)

Task(task, …)

Association classes (I)





Projects(pro, …)

Employees(emp, …)

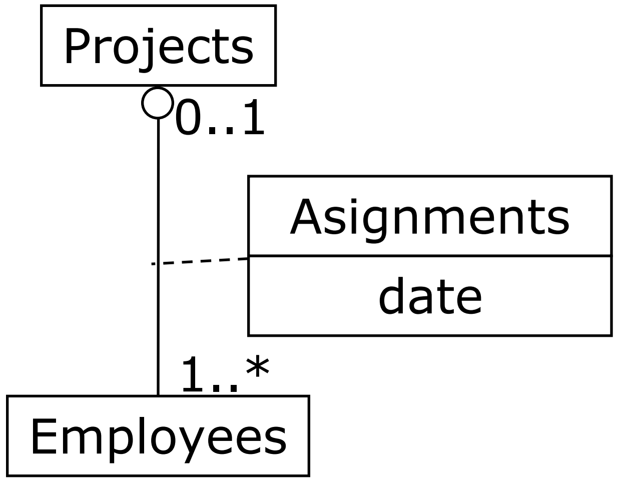
Assignments(emp, proj, date)

Task-Assig(task, emp, proj)

Task(task, …)

FOREIGN KEY (emp, proj) REFERENCES Asignaments

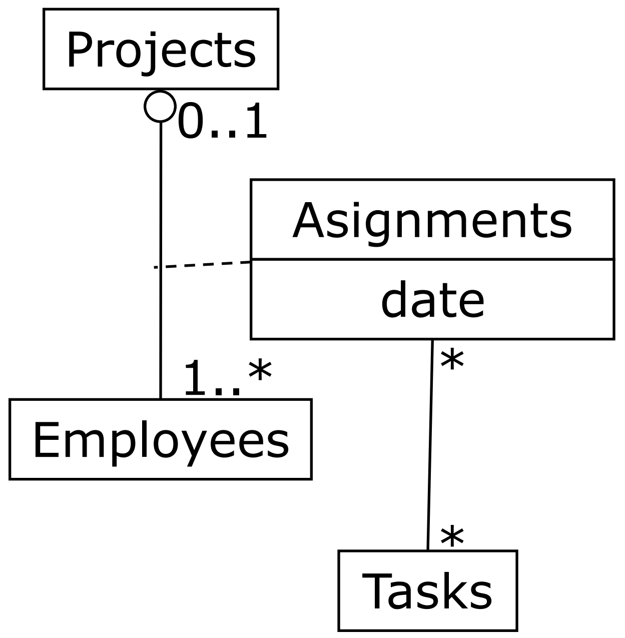
Association classes (II)



Projects(<u>pro</u>, …)

Employees(<u>emp</u>,… , proj, date)

Association classes (II)



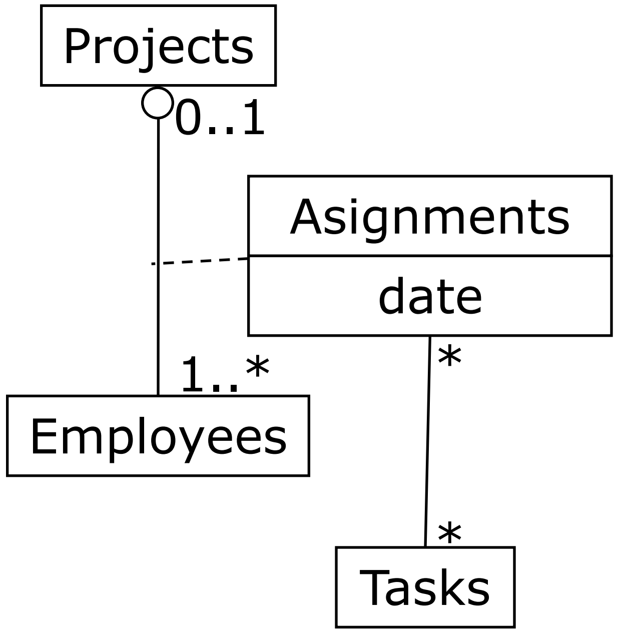
Projects(pro, …)

Employees(emp,… , proj, date)

Task-Assig(tas, emp)

Tasks(tas, …)

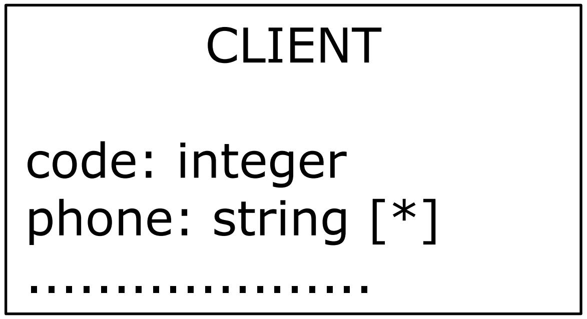
Association classes (II)



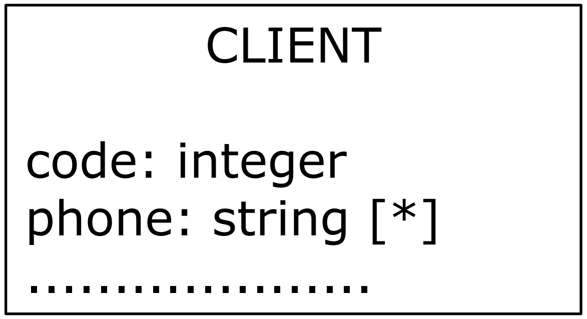
$$\begin{tikzcd}
\text{Projects}(\underline{\text{pro}}, \dots) \\
\text{Employees}(\underline{\text{emp}}, \dots) \\
\text{Assignment}(\text{pro}, \underline{\text{emp}}, \text{date}) \arrow[uu] \arrow[u] \\
\text{Task-Assig}(\underline{\text{tas, emp}}) \arrow[u] \arrow[d] \\
\text{Tasks}(\underline{\text{task}}, \dots)
\end{tikzcd}$$

FOREIGN KEY (emp) REFERENCES Assignments

Multivalued attributes (I)



Multivalued attributes (I)



ONE VALUE PER COLUMN

```
Client( code, office-phone, secretary-phone, cell-phone, ...)
        C1    933333333    933333331       666666666   null
```

Multivalued attributes (I)

CLIENT
code: integer phone: string [*] …………………

ONE VALUE PER COLUMN

Client(code, office-phone, secretary-phone, cell-phone, ...)

C1	933333333	933333331	666666666	null

ONE VALUE PER ROW

Client(code, ...)

$\uparrow$

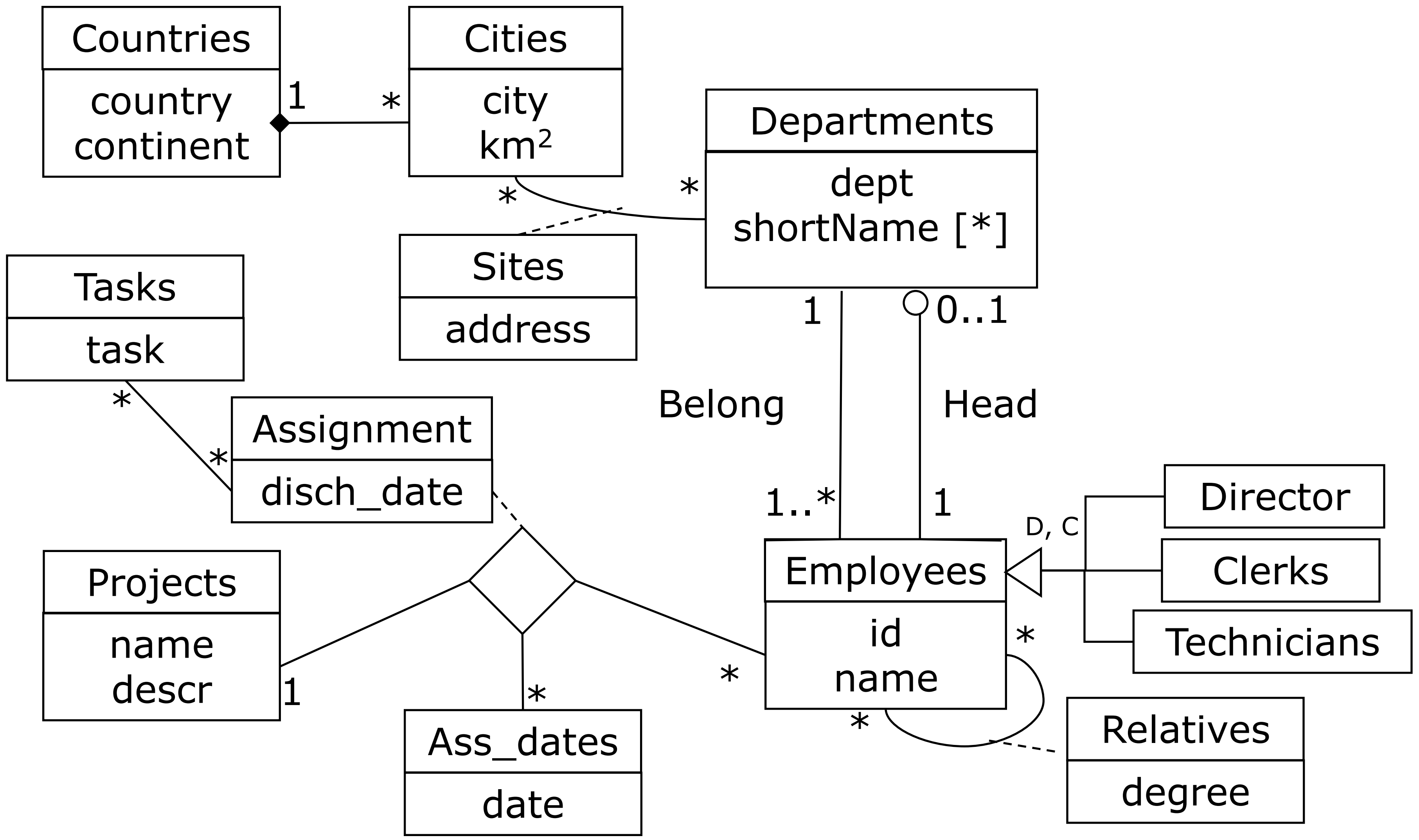
ClientTelephones(code, place, telephone)

C1	Office	933333333
C1	Secretary	933333331
C1	Cellular	666666666

Multivalued attributes (II)

Per column	Per row
Fixed number of values	Variable number of values
Few values	Many values
Generates nulls	There are no null values
One I/O	Many I/O
Global processing	Partial processing
Natural PK	Artificial PK
Less space	More space
Hard to aggregate	Easy to aggregate
Many CHECKs	One CHECK
Lower concurrency	Higher concurrency

Example: Conceptual schema



Summary

- Translation of relationships
- Possible overlooking of classes
- Attributes of relationships
- Class or relationship
- Multivalued attributes

Bibliography (I)

- Jaume Sistac et al. *Disseny de bases de dades*. Editorial UOC, 2002. Col·lecció Manuals, number 43
- T. Teorey et al. *Database modeling and design*. Morgan Kaufmann Publishers, 2006. 4th edition
- R. Elmasri and B. Nabathe. *Fundamentals of Database Systems*. Addison-Wesley, fourth edition, 2003